



E-MapReduce SmartData

文档版本: 20220629



# 法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。 如果您阅读或使用本文档,您的阅读或使用行为将被视为对本声明全部内容的认可。

- 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档,且仅能用 于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息,您应当严格 遵守保密义务;未经阿里云事先书面同意,您不得向任何第三方披露本手册内容或 提供给任何第三方使用。
- 未经阿里云事先书面许可,任何单位、公司或个人不得擅自摘抄、翻译、复制本文 档内容的部分或全部,不得以任何方式或途径进行传播和宣传。
- 由于产品版本升级、调整或其他原因,本文档内容有可能变更。阿里云保留在没有 任何通知或者提示下对本文档的内容进行修改的权利,并在阿里云授权通道中不时 发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠 道下载、获取最新版的用户文档。
- 4. 本文档仅作为用户使用阿里云产品及服务的参考性指引,阿里云以产品及服务的"现状"、"有缺陷"和"当前功能"的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引,但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的,阿里云不承担任何法律责任。在任何情况下,阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害,包括用户使用或信赖本文档而遭受的利润损失,承担责任(即使阿里云已被告知该等损失的可能性)。
- 5. 阿里云网站上所有内容,包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计,均由阿里云和/或其关联公司依法拥有其知识产权,包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意,任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外,未经阿里云事先书面同意,任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称(包括但不限于单独为或以组合形式包含"阿里云"、"Aliyun"、"万网"等阿里云和/或其关联公司品牌,上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司)。
- 6. 如若发现本文档存在任何错误,请与阿里云取得直接联系。

# 通用约定

格式	说明	样例
⚠ 危险	该类警示信息将导致系统重大变更甚至故 障,或者导致人身伤害等结果。	
▲ 警告	该类警示信息可能会导致系统重大变更甚 至故障,或者导致人身伤害等结果。	會告 重启操作将导致业务中断,恢复业务 时间约十分钟。
〔〕) 注意	用于警示信息、补充说明等,是用户必须 了解的内容。	大意 权重设置为0,该服务器不会再接受新 请求。
? 说明	用于补充说明、最佳实践、窍门等 <i>,</i> 不是 用户必须了解的内容。	⑦ 说明 您也可以通过按Ctrl+A选中全部文 件。
>	多级菜单递进。	单击设置> 网络> 设置网络类型。
粗体	表示按键、菜单、页面名称等UI元素。	在 <b>结果确认</b> 页面,单击 <b>确定</b> 。
Courier字体	命令或代码。	执行 cd /d C:/window 命令,进入 Windows系统文件夹。
斜体	表示参数、变量。	bae log listinstanceid
[] 或者 [alb]	表示可选项,至多选择一个。	ipconfig [-all -t]
{} 或者 {a b}	表示必选项,至多选择一个。	switch {active stand}

# 目录

1.7	概述	16
2.	JindoFS介绍和使用	18
3.	SmartData 3.8.x	21
	3.1. SmartData 3.8.x版本简介	21
	3.2. JindoFS Block模式	21
	3.2.1. Block模式使用说明	21
	3.2.2. 使用RocksDB作为元数据后端	23
	3.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	26
	3.2.4. AuditLog使用说明	31
	3.2.5. 访问JindoFS Web UI	35
	3.2.6. 权限功能	36
	3.2.7. 数据管理策略	39
	3.2.8. 文件元数据离线分析	40
	3.2.9. JindoFS Credential Provider使用说明	44
	3.2.10. JindoFS Block模式加密使用说明	46
	3.3. JindoFS Cache模式	49
	3.3.1. Cache模式使用说明	49
	3.3.2. 使用JindoFS SDK免密功能	52
	3.3.3. AuditLog使用说明	54
	3.3.4. Jindo Job Committer使用说明	58
	3.3.5. JindoFS OSS Credential Provider使用说明	60
	3.3.6. 访问JindoFS Web UI	64
	3.3.7. 权限功能	66
	3.4. JindoTable	69
	3.4.1. 开启native查询加速	69
	3.4.2. JindoTable使用说明	72

3.4.3. JindoTable SDK模式归档和解冻命令介绍	75
3.4.4. JindoTable MoveTo命令介绍	78
3.4.5. JindoTable表或分区访问热度收集	81
3.4.6. JindoTable表或分区访问冷度收集	83
3.5. 工具集	86
3.5.1. Jindo sql命令介绍	86
3.5.2. FUSE使用说明	93
3.5.3. Jindo DistCp使用说明	94
3.5.4. Jindo DistCp场景化使用指导	101
3.5.5. 分层存储命令使用说明	107
4.SmartData 3.7.x	109
4.1. SmartData 3.7.x版本简介	109
4.2. JindoFS Block模式	109
4.2.1. Block模式使用说明	109
4.2.2. 使用RocksDB作为元数据后端	111
4.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	115
4.2.4. AuditLog使用说明	119
4.2.5. 访问JindoFS Web UI	123
4.2.6. 权限功能	124
4.2.7. 数据管理策略	127
4.2.8. 文件元数据离线分析	128
4.2.9. JindoFS Credential Provider使用说明	132
4.2.10. JindoFS Block模式加密使用说明	134
4.3. JindoFS Cache模式	137
4.3.1. Cache模式使用说明	137
4.3.2. 使用JindoFS SDK免密功能	140
4.3.3. AuditLog使用说明	142
4.3.4. Jindo Job Committer使用说明	146

4.3.5. JindoFS OSS Credential Provider使用说明	148
4.3.6. 访问JindoFS Web UI	151
4.3.7. 权限功能	153
4.4. JindoTable	156
4.4.1. 开启native查询加速	156
4.4.2. JindoTable使用说明	159
4.4.3. JindoTable SDK模式归档和解冻命令介绍	162
4.4.4. JindoTable MoveTo命令介绍	165
4.4.5. JindoTable表或分区访问热度收集	168
4.4.6. JindoTable表或分区访问冷度收集	170
4.5. 工具集	173
4.5.1. Jindo sql命令介绍	173
4.5.2. FUSE使用说明	180
4.5.3. Jindo DistCp使用说明	181
4.5.4. Jindo DistCp场景化使用指导	188
4.5.5. 分层存储命令使用说明	194
5.SmartData 3.6.x	196
5.1. SmartData 3.6.x版本简介	196
5.2. JindoFS Block模式	197
5.2.1. Block模式使用说明	197
5.2.2. 使用RocksDB作为元数据后端	199
5.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	202
5.2.4. AuditLog使用说明	207
5.2.5. 访问JindoFS Web UI	210
5.2.6. 权限功能	212
5.2.7. 数据管理策略	215
5.2.8. 文件元数据离线分析	216
5.2.9. JindoFS Credential Provider使用说明	219

5.2.10. JindoFS Block模式加密使用说明	221
5.3. JindoFS Cache模式	225
5.3.1. Cache模式使用说明	225
5.3.2. 使用JindoFS SDK免密功能	228
5.3.3. AuditLog使用说明	230
5.3.4. Jindo Job Committer使用说明	234
5.3.5. JindoFS OSS Credential Provider使用说明	236
5.3.6. 访问JindoFS Web UI	239
5.3.7. 权限功能	241
5.4. JindoTable	244
5.4.1. 开启native查询加速	244
5.4.2. JindoTable使用说明	247
5.4.3. JindoTable SDK模式归档和解冻命令介绍	250
5.4.4. JindoTable MoveTo命令介绍	253
5.4.5. JindoTable表或分区访问热度收集	256
5.4.6. JindoTable表或分区访问冷度收集	258
5.5. 工具集	261
5.5.1. Jindo sql命令介绍	261
5.5.2. FUSE使用说明	268
5.5.3. Jindo DistCp使用说明	270
5.5.4. Jindo DistCp场景化使用指导	276
5.5.5. 分层存储命令使用说明	282
6.SmartData 3.5.x	284
6.1. SmartData 3.5.x版本简介	284
6.2. JindoFS Block模式	284
6.2.1. Block模式使用说明	284
6.2.2. 使用RocksDB作为元数据后端	286
6.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	290

6.2.4. AuditLog使用说明	294
6.2.5. 访问JindoFS Web UI	200
6.2.6. 权限功能	298 299
6.2.7. 数据管理策略	302
6.2.8. 文件元数据离线分析	303
6.2.9. JindoFS Credential Provider使用说明	307
6.2.10. JindoFS Block模式加密使用说明	309
6.3. JindoFS Cache模式	312
6.3.1. Cache模式使用说明	312
6.3.2. 使用JindoFS SDK免密功能	315
6.3.3. AuditLog使用说明	317
6.3.4. Jindo Job Committer使用说明	321
6.3.5. JindoFS OSS Credential Provider使用说明	323
6.3.6. 访问JindoFS Web UI	326
6.3.7. 权限功能	328
6.4. JindoTable	331
6.4.1. 开启native查询加速	331
6.4.2. JindoCube使用说明	333
6.4.3. JindoTable使用说明	340
6.4.4. JindoTable表或分区访问热度收集	344
6.4.5. JindoTable表或分区访问冷度收集	346
6.5. 工具集	349
6.5.1. FUSE使用说明	349
6.5.2. Jindo DistCp使用说明	351
6.5.3. Jindo DistCp场景化使用指导	358
6.5.4. 分层存储命令使用说明	363
7.SmartData 3.4.x	365
7.1. SmartData 3.4.x版本简介	365

7.2. JindoFS Block模式	365
7.2.1. Block模式使用说明	365
7.2.2. 使用RocksDB作为元数据后端	367
7.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	371
7.2.4. AuditLog使用说明	375
7.2.5. 访问JindoFS Web UI	379
7.2.6. 权限功能	380
7.2.7. 数据管理策略	383
7.2.8. 文件元数据离线分析	384
7.2.9. JindoFS Credential Provider使用说明	388
7.2.10. JindoFS Block模式加密使用说明	390
7.3. JindoFS Cache模式	393
7.3.1. Cache模式使用说明	393
7.3.2. 使用JindoFS SDK免密功能	396
7.3.3. AuditLog使用说明	398
7.3.4. Jindo Job Committer使用说明	402
7.3.5. JindoFS OSS Credential Provider使用说明	404
7.3.6. 访问JindoFS Web UI	407
7.3.7. 权限功能	409
7.4. JindoTable	412
7.4.1. 开启ORC查询加速	412
7.4.2. JindoTable使用说明	414
7.4.3. JindoCube使用说明	417
7.4.4. JindoTable表或分区访问热度收集	425
7.5. 工具集	427
7.5.1. FUSE使用说明	427
7.5.2. Jindo DistCp使用说明	429
7.5.3. Jindo DistCp场景化使用指导	436

7.5.4. 分层存储命令使用说明	442
8.SmartData 3.2.x	444
8.1. SmartData 3.2.x版本简介	444
8.2. JindoFS Block模式	444
8.2.1. Block模式使用说明	444
8.2.2. 使用RocksDB作为元数据后端	446
8.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	450
8.2.4. AuditLog使用说明	454
8.2.5. 访问JindoFS Web UI	458
8.2.6. 权限功能	459
8.2.7. 数据管理策略	462
8.2.8. 文件元数据离线分析	463
8.3. JindoFS Cache模式	467
8.3.1. Cache模式使用说明	467
8.3.2. 使用JindoFS SDK免密功能	470
8.3.3. AuditLog使用说明	472
8.3.4. Jindo Job Committer使用说明	475
8.3.5. Credential Provider使用说明	477
8.3.6. 访问JindoFS Web UI	480
8.3.7. 权限功能	482
8.4. JindoTable	485
8.4.1. 开启ORC查询加速	485
8.4.2. JindoTable使用说明	487
8.4.3. JindoCube使用说明	490
8.4.4. JindoTable表或分区访问热度收集	498
8.5. 工具集	500
8.5.1. FUSE使用说明	500
8.5.2. Jindo DistCp使用说明	502

8.5.3. Jindo DistCp场景化使用指导	509
8.5.4. 分层存储命令使用说明	515
9.SmartData 3.1.x	517
9.1. SmartData 3.1.x版本简介	517
9.2. JindoFS Block模式	518
9.2.1. Block模式使用说明	518
9.2.2. 使用RocksDB作为元数据后端	520
9.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	523
9.2.4. AuditLog使用说明	528
9.2.5. 访问JindoFS Web UI	531
9.2.6. 权限功能	533
9.2.7. 数据管理策略	536
9.2.8. 文件元数据离线分析	537
9.3. JindoFS Cache模式	540
9.3.1. Cache模式使用说明	540
9.3.2. 使用JindoFS SDK免密功能	543
9.3.3. AuditLog使用说明	546
9.3.4. Jindo Job Committer使用说明	549
9.3.5. JindoFS OSS Credential Provider使用说明	551
9.3.6. 访问JindoFS Web UI	554
9.3.7. 权限功能	555
9.4. JindoTable	558
9.4.1. JindoTable使用说明	558
9.4.2. JindoTable表或分区的访问热度收集	562
9.4.3. JindoCube使用说明	563
9.5. 工具集	571
9.5.1. FUSE使用说明	571
9.5.2. Jindo DistCp使用说明	573

9.5.3. Jindo DistCp场景化使用指导	580
9.5.4. 分层存储命令使用说明	586
10.SmartData 3.0.x	588
10.1. SmartData 3.0.x版本简介	588
10.2. JindoFS Block模式	588
10.2.1. Block模式使用说明	589
10.2.2. 使用RocksDB作为元数据后端	591
10.2.3. 使用Raft-RocksDB-Tablestore作为存储后端	594
10.2.4. 访问JindoFS Web UI	599
10.2.5. 权限功能	600
10.2.6. AuditLog使用说明	603
10.2.7. 文件元数据离线分析	607
10.3. JindoFS Cache模式	610
10.3.1. JindoFS缓存模式使用说明	610
10.3.2. 使用JindoFS SDK免密功能	613
10.3.3. 访问JindoFS Web UI	615
10.3.4. 权限功能	617
10.3.5. Jindo Job Committer使用说明	620
10.3.6. JindoFS AuditLog使用说明	622
10.3.7. Credential Provider使用说明	625
10.4. JindoTable	627
10.4.1. JindoTable使用说明	627
10.4.2. JindoCube使用说明	631
10.5. 工具集	638
10.5.1. JindoFS FUSE使用说明	638
10.5.2. 分层存储命令使用说明	640
10.5.3. Jindo DistCp使用说明	641
10.5.4. Jindo DistCp场景化使用指导	648

11.SmartData 2.7.3-2.7.4	655
11.1. JindoFS Block模式	655
11.1.1. Block模式使用说明	655
11.1.2. 使用Tablestore作为存储后端	657
11.1.3. 使用RocksDB作为元数据后端	659
11.1.4. 使用Raft-RocksDB-Tablestore作为存储后端	660
11.1.5. JindoFS权限功能	665
11.1.6. Jindo AuditLog使用说明	668
11.2. JindoFS Cache模式	670
11.2.1. JindoFS缓存模式使用说明	670
11.2.2. 使用JindoFS SDK免密功能	674
11.2.3. Jindo Job Committer使用说明	676
11.2.4. JindoFS权限功能	678
11.2.5. Jindo AuditLog使用说明	681
11.3. JindoTable	683
11.3.1. JindoCube使用说明	683
11.4. 工具集	691
11.4.1. Jindo DistCp使用说明	691
11.4.2. FUSE使用说明	698
12.SmartData 2.6.0-2.7.2	701
12.1. SmartData 2.6.0-2.7.2版本简介	701
12.2. JindoFS Block模式	701
12.2.1. JindoFS块存储模式使用说明	701
12.2.2. 使用Tablestore作为存储后端	703
12.2.3. 使用RocksDB作为元数据后端	706
12.2.4. 使用Raft-RocksDB-Tablestore作为存储后端	707
12.2.5. JindoFS权限功能	712
12.3. JindoFS Cache模式	714

12.3.1. JindoFS缓存模式使用说明	714
12.3.2. JindoFS权限功能	717
12.3.3. Jindo Job Committer使用说明	720
12.4. JindoTable	722
12.4.1. JindoCube使用说明	722
12.5. 工具集	730
12.5.1. Jindo DistCp使用说明	730
13.SmartData 2.2.x及之前版本	738
13.1. SmartData使用说明(EMR-3.20.0~3.22.0版本)	738
13.2. SmartData使用说明(EMR-3.22.0~3.25.1版本)	742
13.3. JindoFS块存储模式	749
13.4. JindoFS缓存模式	752
13.5. 使用JindoFS SDK免密功能	754
13.6. JindoFS外部客户端	756
14.最佳实践	758
14.1. 迁移Hadoop文件系统数据至JindoFS	758
14.2. 使用MapReduce处理JindoFS上的数据	758
14.3. 使用Hive查询JindoFS上的数据	759
14.4. 使用Spark处理JindoFS上的数据	760
14.5. 使用Flink处理JindoFS上的数据	761
14.6. 使用Impala或Presto查询JindoFS上的数据	761
14.7. 使用JindoFS作为HBase的底层存储	762
14.8. 基于JindoFS存储YARN MR或SPARK作业日志	765
14.9. 将Kafka数据导入JindoFS	768
14.10. 跨集群访问JindoFS	768
14.11. 改写Jindo HDFS客户端路径	769
14.12. 支持Flink可恢复性写入JindoFS或OSS	771
14.13. 使用Flume写入JindoFS	772

15.SmartData常见问题		775
------------------	--	-----

# 1.概述

Smart Dat a是E-MapReduce(简称EMR)产品的核心自研组件,为EMR各个计算引擎提供统一的存储优化、缓存优化、计算加速 优化和多个存储功能扩展,涵盖数据访问、数据治理和数据安全。

Smart Dat a组件在EMR产品中的位置如下所示。



Smart Dat a组件包括:

- JindoFS核心子系统:为各种远端存储系统提供缓存和缓存加速,详情请参见JindoFS介绍和使用。
- JindoTable核心子系统:为表格数据源(例如Hive数仓)提供表和分区级别的优化和治理,详情请参见JindoTable使用说明。
- JindoManager:提供JindoFS&JindoTable相关服务和功能的管理页面,例如,查看文件和表在缓存上的各种统计指标。
- JindoSDK:为EMR各种开源计算引擎提供统一的SDK,支持Java、C、C++和Python语言,提供多种访问和API接口,包括HCFS 文件系统接口、POSIX接口和Table表格接口。
- 工具集: 提供相关的工具集, 例如Jindo tool和迁移工具Jindo Dist Cp。
- 各种Connectors:包括Hadoop connector、Flink connector和TensorFlow connector,支持Kite SDK、Apache Beams、Flume、Sqoop和Kaf ka。

Smart Dat a目前通过JindoFS和JindoTable支持的数据源,包括阿里云OSS、Apache Hadoop HDFS、Hive数仓和阿里云 MaxCompute。

Smart Dat a作为EMR产品核心自研组件,独立开发与版本发布,详细版本请参见版本概述。

Smart Data详细使用,请查看相应文档:

- Smart Data 3.8.x版本简介
- Smart Dat a 3.7.x版本简介
- Smart Data 3.6.x版本简介
- Smart Data 3.5.x版本简介
- Smart Data 3.4.x版本简介
- Smart Data 3.2.x版本简介
- Smart Data 3.1.x版本简介
- Smart Dat a 3.0.x版本简介
- Smart Dat a 2.7.3-2.7.4版本
- Smart Dat a 2.6.0-2.7.2版本简介

- Smart Dat a使用说明(EMR-3.22.0~3.25.1版本)
- Smart Dat a使用说明(EMR-3.20.0~3.22.0版本)

# 2.JindoFS介绍和使用

JindoFS是基于阿里云对象存储OSS,为开源大数据生态构建的Hadoop兼容文件系统(Hadoop Compatible File System,HCFS)。JindoFS提供兼容对象存储的纯客户端模式(SDK)和缓存模式(Cache),以支持与优化Hadoop和Spark生 态大数据计算对OSS的访问;提供块存储模式(Block),以充分利用OSS的海量存储能力和优化文件系统元数据的操作。

## JindoFS纯客户端模式(SDK)

JindoFS纯客户端模式为Hive和Spark等计算框架提供了访问阿里云OSS及其各种操作的优化,类似Hadoop社区的OSS FileSystem 或S3A FileSystem。此模式不改变文件或对象在OSS上的组织方式,文件还是保存在OSS上,JindoFS只是提供面向Hadoop生态的客户端连接、扩展、适配和优化访问。您可以使用此模式,上传JindoFS SDK的JAR包至组件的*classpath*目录,简单易用,无需 部署分布式服务。



## JindoFS缓存模式(Cache)

JindoFS缓存模式(Cache)兼容JindoFS纯客户端模式(SDK),同时利用Jindo分布式缓存能力在计算侧为OSS提供缓存加速,以 满足大规模的分析和训练吞吐需求。在纯客户端模式(SDK)基础上,Cache模式支持可选的元数据缓存和数据分布式缓存,同 时保持数据跟OSS兼容和同步。数据缓存可以基于内存、SSD和普通磁盘,以适用不同的计算场景。



JindoFS块存储模式(Block)

JindoFS存储模式(Block),不仅提供缓存加速能力,还可以组织、存储数据和管理文件元数据,类似Apache Hadoop HDFS。 在此模式下JindoFS是个独立的存储系统,只是文件块数据存储在OSS上。



## Cache模式和Block模式对比

两种模式都把数据存储在OSS上,同时根据本地缓存空间剩余情况确定是否在本地也放置一份以用于缓存加速。

两种模式的本质区别在于,块存储模式可以管理目录和文件元数据,文件是分成多个块存储在OSS上,所以写到OSS上的是一个 一个的文件块,而缓存模式存储的是整个文件对象。

### 三种模式对比

以下从多个维度上介绍JindoFS的纯客户端模式(SDK)、缓存模式(Cache)和块存储模式(Block)的差异。

维度	JindoFS SDK	JindoFS Cache	JindoFS Block
存储成本	● 全量数据OSS存储。 ● 支持归档。	<ul> <li>全量数据OSS存储。</li> <li>热数据使用缓存(20%)。</li> <li>支持归档。</li> </ul>	<ul> <li>全量数据OSS存储。</li> <li>温数据和热数据(60%)本 地缓存1备份。</li> <li>支持归档。</li> <li>支持透明压缩。</li> </ul>
弹性	高	较高	支持
吞吐	OSS带宽。	OSS带宽和热数据缓存带宽。	OSS带宽、温数据和热数据缓存 带宽。
元数据	<ul> <li>模拟HDFS,不支持目录和文件语义。</li> <li>规模超大,支持EB级别。</li> </ul>	<ul> <li>模拟HDFS,不支持目录和文件语义,支持文件数据缓存。</li> <li>规模超大,支持EB级别。</li> </ul>	<ul> <li>兼容性接近HDFS,性能最高。</li> <li>规模大,支持10亿以上的文件数。</li> </ul>
运维	低	一般 需要维护缓存系统能力。	较高 需要维护文件系统元数据服务和 缓存系统。

#### Smart Dat a· JindoFS介绍和使用

#### E-MapReduce

维度	JindoFS SDK	JindoFS Cache	JindoFS Block
安全	<ul> <li>支持AccessKey认证。</li> <li>支持RAM鉴权。</li> <li>支持OSS访问日志。</li> <li>支持OSS数据加密。</li> </ul>	<ul> <li>支持AccessKey认证。</li> <li>支持RAM鉴权。</li> <li>支持OSS访问日志。</li> <li>支持OSS数据加密。</li> </ul>	<ul> <li>支持AccessKey认证。</li> <li>支持Unix权限和Ranger权限。</li> <li>支持审计日志 (AuditLog)。</li> <li>支持数据加密。</li> </ul>
	仅支 持 <i>oss://<oss_bucket>/<oss_ dir&gt;/方式,支持跨产品访问该 文件路径。</oss_ </oss_bucket></i>	<ul> <li>默认使用方式oss://<oss_b ucket&gt;/<oss_dir>/,支持跨 产品访问该文件路径,可以 打开缓存开关。</oss_dir></oss_b </li> <li>支持多Namespace使用方式/ fs://<your_namespace>/&lt; path_of_file&gt;,不支持跨产</your_namespace></li> </ul>	仅支持多Namespace使用方 式 <i>jfs://<your_namespace>/&lt; path_of_file&gt;,</your_namespace></i> 不支持跨产品 访问该文件路径,可以打开缓存 开关。
使用方式		品访问该文件路径,可以打 开缓存开关。	⑦ 说明 Block模式使 用详情请参见各版本下
		⑦ 说明 Cache模式使 用详情请参见各版本下 LindoFS Cache模式的内	JindoFS Block模式的内 容。
		容。	

#### 常见问题

• Q: 针对典型的数据湖场景, 建议采用什么模式?

A:因为JindoFS SDK和Cache模式完全兼容OSS对象存储语义,具有完全的存储分离架构和弹性灵活性,所以,针对典型的数 据湖场景,推荐您使用SDK或者Cache模式以支持大数据分析和A训练加速。

• Q: 为什么Block模式跟HDFS相比, 是更好的HDFS?

A:

- HDFS的常规限制为4亿,而Block模式元数据规模上支撑10亿以上的文件数,大于HDFS的限制,而且在集群高峰期时性能更为稳定。
- HDFS有Java onheap限制,而Block模式没有Java onheap和内存限制,可以支持更大的数据规模。
- Block模式轻运维,不用担心坏盘或坏节点,数据1备份放置在OSS上,支持上下线节点。
- 支持对冷数据做透明压缩和归档,使用多种手段进行成本优化,对接对象存储,支持EB级数据规模。
- Block模式支持HDFS的一些重要特性。例如,HDFS Audit Log、Ranger集成和数据加密。
- Q: Block模式具有哪些特别的优势?

A:

- Block模式可以管理文件元数据和组织文件数据,因此可以不局限于OSS对象存储,完全可以满足各种大数据引擎对存储接口的需求。这些接口包括但不限于Rename的原子性和事务性能力、高性能本地写入、透明压缩、truncate、append、flush、sync和snapshot等。这些高阶存储接口对实现完整的POSIX和对接更多的大数据引擎到OSS是不可或缺的,例如,Flink、HBase、Kafka和Kudu。其他两种方式使用OSS也可以对接部分接口,但是能力和优势会有所不足。
- Block模式在费用上优于其他两种方式使用OSS。Block模式中,因为全部数据中占比60%的温数据和热数据都在本地有缓存 备份,大部分读请求都不会通过OSS,所以可以节省一部分费用。

# 3.SmartData 3.8.x 3.1. SmartData 3.8.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS、JindoT able和相关工具集。本文为您介绍Smart Dat a(3.8.x)版本更新的内容。

#### JindoSDK

特性	描述
支持AssumeRoleStsCredentialsProvider	适用于获取一个扮演RAM角色的临时AccessKey访问OSS的情况,详细信息请参 见J <mark>indoFS OSS Credential Provider使用说明</mark> 。
支持JindoRangerCredentialsProvider	适用于通过配置Ranger来控制用户访问OSS权限的情况,详细信息请参见JindoFS OSS Credential Provider使用说明。

# 3.2. JindoFS Block模式

# 3.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

### 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

## 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面, 单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	tia			服务配置
配置搜	索		Q	全部 smartdata-site namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

- i. 修改jfs.namespaces为test。
- test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。
- ii. 单击自定义配置, 在新增配置项对话框中增加以下参数, 单击确定。

参数	描述	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>

#### ⅲ. 单击确定。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

### 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在**服务配置**区域的storage页签,修改如下参数。

服务配置				
全部 smartdata-site namespace	client storage			
	storage.handler.threads	40		0
	storage.watermark.low.ratio	0.2		0
	storage.watermark.high.ratio	0.4		0
	storage.oss.upload.threads	20		0
参数		描述		
storage.watermark.high.ratio		表示磁盘( 用的磁盘3	吏用量的上水位比例,每块数据盘的J 空间到达上水位即会触发清理。默认(	indoFS数据目录占 值: 0.4。
storage.watermark.low.ratio		表示使用 JindoFS数	量的下水位比例,触发清理后会自动 据目录占用空间清理到下水位。默认	青理冷数据,将 值:0.2。

```
⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。
```

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 3.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

#### 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见<mark>使用Raft-RocksDB-</mark> Tablestore作为存储后端。

单机RocksDB作为元数据服务的架构图如下所示。



#### 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面, 单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。

#### ii. 单击namespace。

状态     部署拓扑     配置修改历史       配置过滤     服务配置       配置搜索     全部   smartdata-site     namespace       client   storage	< 返回	a Smart	Data 🗸	●正常				
配置过滤 服务配置 配置搜索 全部 smartdata-site namespace client storage	状态	部署拓扑	配置	配置修改历史				
配置搜索 全部 smartdata-site namespace client storage	配置过	滤			服务配置			
	配置搜	素			全部 smartdata-site nan	mespace	client	storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中,输入执行原因,开启自动更新配置。
  - ⅲ. 单击**确定**。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

7. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 8. 单击右上角的操作 > 启动All Components。

#### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. 准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809 25 jfs://test/
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 \_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ jindo jfs -metaStatus
<pre>==== emr-header-1:8101 ====</pre>
OtsUploader: _synced=1
[RocksDB Row Counts of each CF(Table)]

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。

3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: ° true ° false	true

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

iii. 单击确定。

- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

<pre>==== emr-header-1:8101 ====</pre>	
[Recovery From OTS Status]	
state: FINISH	
total 22855 rows.	

(可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。
 此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一致
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
         1596 1482809
                                                    25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items

      drwxrwxr-x
      - root
      root
      0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-

      -rw-r----
      1 hadoop hadoop
      5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt

      -rw-r-----
      1 hadoop hadoop
      20 2020-03-25 15:07 jfs://test/testfile

drwxrwxr-x - root root
                                              0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021 , ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面, 单击相应集群所在行的更多 > 重启。

# 3.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。Raft元数据实例的 OTS(TableStore)备份为可选的容灾操作,Raft元数据本身具有一定的容灾能力,可根据实际情况判断是否需要开启。

#### 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





#### 前提条件

• (可选)创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

⑦ 说明 需要开启OTS(Tablestore)的事务特性。

• 创建3 Master的EMR集群,详情请参见创建集群。



⑦ 说明 如果没有部署方式,请提交工单处理。

## 配置本地raft后端

- 1. 进入SmartData服务页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 暂停SmartData所有服务。
  - i. 选择右上角的操作 > 停止All Components。
  - ii. 在执行集群操作对话框,填写执行原因,单击确定。

- iii. 在确认对话框, 单击确定。
- 3. 根据使用需求,添加需要的namespace。
- 4. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏,选择**集群服务 > SmartData**。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击namespace页签。
- 5. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ **说明** 如果不需要使用OTS远端存储,直接执行步骤7和步骤9;如果需要使用OTS远端存储,请执行步骤6~步骤 9。

#### 6. (可选)配置远端OTS异步存储。

#### 在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。 ⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	true

- 7. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

- 8. 启动SmartData所有服务。
  - i. 在SmartData服务页面,选择右上角的操作 > 启动All Components。
  - ii. 在执行集群操作对话框,填写执行原因,单击确定。
  - iii. 在确认对话框,单击确定。
- 9. 通过SSH方式连接集群,然后执行以下命令查看状态。

jindo jfs -metaStatus -detail

如果返回信息中包含raft字样,则表示配置生效。

#### 从Tablestore恢复元数据信息

↓ 注意 需要开启OTS (TableStore) 的备份功能。

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. 查看原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

#### 返回信息如下所示。

1596 1482809

25 jfs://test/

⑦ 说明 返回信息中的1596表示文件夹个数,1482809为文件个数。

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 \_\_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail



iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。

2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: ° true ° false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	true

4. 保存配置。

i. 单击右上角的保存。

- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 5. 启动SmartData所有服务。
  - i. 在SmartData服务页面,选择右上角的操作 > 启动All Components。
  - ii. 在执行集群操作对话框,填写执行原因,单击确定。
  - iii. 在确认对话框, 单击确定。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus -detail

如图所示,LEADER节点的state为FINISH表示恢复完成。



- 7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。
  - 执行以下命令,查看文件数量是否与原集群一致。

hadoop fs -count jfs://test/

返回信息如下所示。

1596 1482809

25 jfs://test/

• 执行以下命令, 查看文件是否可以正常读取。

hadoop fs -cat jfs://test/testfile

• 执行以下命令, 查看文件目录。

hadoop fs -ls jfs://test/

返回信息如下所示。

```
Found 3 items
drwxrwxr-x - root root
-rw-r---- 1 hadoop hadoop
-rw-r---- 1 hadoop hadoop
```

0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt 20 2020-03-25 15:07 jfs://test/testfile

#### • 执行以下命令,验证文件是否不可修改。

hadoop fs -rm jfs://test/testfile

#### 返回如下提示信息。

java.io.IOException: ErrorCode : 25021 , ErrorMsg: Namespace is under recovery mode, and is read-only.

⑦ 说明 此时的集群为恢复模式,也是只读模式,不可修改文件

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	false

#### 9. 重启集群。

- i. 单击上方的集群管理页签。
- ii. 在集群管理页面,选择相应集群所在行的更多 > 重启。
- iii. 在确认重启集群对话框中, 单击确定。

## 3.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功能,所以您可以通过Shell命令分析指定的日志文件。

### 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

### 使用AuditLog

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜查	<u></u>		Q	全部 smartdata-site namespace client storage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> show tables;	
database tableName	isTemporary
default audit_log false	
default audit_log_source	false
default fs_image false	

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken: 0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sql>	select * from	n audit_log_sour	ce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-20	10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null per	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-20	10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null per	rm=null	2020-10-20						
2020-10-20	11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null per	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-20	11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null per	rm=null	2020-10-20						
2020-10-20	11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s dst	t=null	<pre>perm=root:root:</pre>	rwxr-xx	2020-10	-20			
2020-10-20	11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null per	rm=null	2020-10-20						
2020-10-20	11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s dst	t=null	perm=root:root:	rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	m audit_log	lim	it 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true roo	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	0:50:11.950	true roo	ot (	auth:SIMPL	LE)	192.	- 10 C	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true roo	ot (	auth:SIMPL	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	1:26:06.469	true roo	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true roc	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:11.320	true roc	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:14.368	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:14.393	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:16.230	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:16.255	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ds, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-sql> select cmd	,count(*) f	from audit_le	og where	date="2020	-10-20"	group	by cmd	order	by	cmd;
cmd count(1)										
getFileStatusRequest	387									
listFileletRequest	387									
Time taken: 5,767 seco	nds. Fetche	ed 2 row(s)								

# 3.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

#### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

#### 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web U功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview	
Start Time:	Fri Oct 16 12:29:24 2020
Status:	Active
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)
Node:	Live Nodes: [2], Decommission Nodes: [0]
Version:	3.0.0
Build No:	fa0ea608a42

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)					
Namespace:	jfs://test/				
Namespaces:	test				
Mode:	BLOCK_MODE				
Backend URI:	oss://				
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27				

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)								
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version	
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4	
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4	

Overview								
Start Time:	Fri Oct 16 12:29		9:25 2020					
Version:		3.0.0						
Build Version:		fa0ea608a42a	5e0e4ebcdbbfc3c041fe49f8e82	le				
Storage Lists (4)								
Directory	Stora	ageType	Capacity	Last Eviction Time	Partition Id			
/mnt/d	Disk		28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31			
/mnt/d	Disk		19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21			
/mnt/d								
	Disk		51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1			

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 3.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

### 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限:您可以设置文件的777权限,以及Owner和Group。
  - 。 Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。


# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	â Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 志			服务配置
配置搜	素			全部 i smartdata-site <b>namespace</b> client i storage

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager 🗅 Audit 🔹	Settings			🔒 admin
Service Manager				
Service Manager				🛛 Import 🖾 Export
C HDES		F + 50		+ 22
			emphie	
	enr-noase	· · · · · · · · · · · · · · · · · · ·	em-nive	

iii. 配置相关参数。

参数	描述
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。
Username	自定义。
Password	自定义。
Namenode URL	输入jfs://{namespace_name}/。
Authorization Enabled	使用默认值No。
Authentication Type	使用默认值Simple。
dfs.datanode.kerberos.principal	
dfs.namenode.kerberos.principal	<b>太</b> 指它
dfs.secondary.namenode.kerberos.principal	د بهرسا و د ا
Add New Configurations	

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。 以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

#### 3. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 3.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

# 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 令使用说明。

# 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。

策略名称	策略说明
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令,获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

### 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用 于一些高压缩比的文件。支持以下两种压缩策略。

策略名称	策略说明
NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path>  $\ldots$ 

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path>: 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

# 3.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

### 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

#### 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

#### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

bin jindo jfs -dumpMetadata test-block
SLF4J: CLass path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.htmlamut.ple\_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4].impl.Log4jLoggerFactory]
SLF4J: Actual binding is of type [org.slf4].impl.Log4jLoggerFactory]

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

```
Sucessfully upload namespace metadata to OSS.
```

# 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如, 配置的 namespace.sysinfo.oss.uri 为 oss://abc/ ,则上传的文件会在 oss://abc/metadataDump 子目录中。

参数	说明
namespace.sysinfo.oss.uri	存储Bucket和路径。
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinf o.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

← → ↑ 1 oss://emrtest/svsinfd	/metadataDump test-block/
▲上传 + 创建目录 □	更多▼
2 名称	类型 / 大小
☐ ≥ 2020_09_14_18_58_16	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

#### 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

#### E-MapReduce

{			
	"type":"string",	/*INode <b>类型,</b> FILE <b>文件</b> DIRECTORY <b>目录</b> */	
	"id": "string",	/*INode id*/	
	"parentId" :"string",	/* <b>父节点</b> id*/	
	"name":"string",	/*INode <b>名称</b> */	
	"size": "int",	/*INode <b>大小,</b> bigint*/	
	"permission":"int",	/*permission <b>以</b> int <b>格式存放</b> */	
	"owner":"string",	/*owner <b>名称</b> */	
	"ownerGroup":"string",	/*owner <b>组名称</b> */	
	"mtime":"int",	/*inode <b>修改时间,</b> bigint*/	
	"atime":"int",	/*inode <b>最近访问时间,</b> bigint*/	
	"attributes":"string",	/*文件相关属性*/	
	"state":"string",	/*INode <b>状态</b> */	
	"storagePolicy":"string	; <b>",</b> /*存储策略*/	
	"etag":"string"	/*etag*/	
}			

# 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 audit\_log和fs\_image。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta data 生成的数据。

```
示例如下。
```

```
jindo-sql> show partitions fs_image;
partition
namespace=k /datetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)
```

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@er Spark m	nr-worke aster: y	r-2 hado arn, App	op]# jind lication	lo sql Id: app	,											
jindo-se	ql> show	tables;														
databas	e	tableNa	me	isTempo	orary											
default	audit_L	og	talse													
default	audit_l	og_sourc	e	false												
default	fs_imag	e	false													
Time ta	ken: 0.3	45 secon	ids, Fetch	ied 3 ro	m(s)											
jindo-se	ql> sele	ct * fro	m fs_imag	e limit	: 10;											
atime	attr	etag	id	mtime	name	owner	ownerGr	roup	parentId	permission	size	state	storage	Policy	type	name
space	datetim															
0			73110760	0505189	9448	1603084	070081	/tpcds/	'orc/5000/web	_returns/wr_returned	d_date_s	k=2450819	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			16534448	0419066	75495	1603084	071350	/tpcds/	orc/5000/web	_returns/wr_returned	_date_s	k=2450820	root	root	334790	833296
5855433	489		Finalize	!d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			73110760	0505189	9470	1603084	070185	/tpcds/	orc/5000/web	_returns/wr_returned	l_date_s	k=2450821	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			11922762	0234792	87249	1603084	069581	/tpcds/	'orc/5000/web	_returns/wr_returned	_date_s	k=2450822	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			10769840	5188724	41036	1603084	073592	/tpcds/	orc/5000/web	_returns/wr_returned	_date_s	k=2450823	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			26993899	8662451	1354	1603084	068996	/tpcds/	orc/5000/web	_returns/wr_returned	_date_s	k=2450824	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			11922762	0234792	87307	1603084	069875	/tpcds/	orc/5000/web	_returns/wr_returned	date_s	k-2450825	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			15464684	8201766	5002	1603084	072440	/tpcds/	/orc/5000/web	_returns/wr_returned	date_s	k=2450826	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	rv	kugou	2020_10_20_	10_50_36						
0			16534448	0419066	75460	1603084	071170	/tpcds/	orc/5000/web	_returns/wr_returned	date_s	k=2450827	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	r'v	kugou	2020 10 20	10 50 36						
0			73110760	0505189	9544	1603084	070572	/tpcds/	orc/5000/web	_returns/wr_returned	date_s	k=2450828	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

jindo-sql> count(1)	select count(*)	from fs_image where	e type = "Directory"	and namespace="kugou"	and datetime="2020_10_20_10_47_14";	
11837 Time taken:	6.852 seconds,∣	Fetched 1 row(s)				

# 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table\_name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

示例如下。

[hive> select ^ from inode_metadata_test8 limit 100;								
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futu	ns. Conside		different execut	ion engine (i.e.	spark; te	z) or using Hiv		eleases.
Query ID = root_202089								
Total jobs = 1								
Launching Job 1 out of 1								
Number of reduce tasks is set to 0 since there's no reduce operator								
Starting Job = job_1 , Tracking URL = http://emr-heade	0888/proxy/	applica						
Kill Command = /usr/lib/hadoop-current/bin/hadoop job -kill job_1599								
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0								
[2020-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0%								
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.22 se								
MapReduce Total cumulative CPU time: 1 seconds 220 msec								
Ended Job = job_;								
MapReduce Jobs Launched:								
Stage-Stage-1: Map: 1 Cumulative CPU: 1.22 sec HDFS Read: 6867 HDFS Write: 1								
Total MapReduce CPU Time Spent: 1 seconds 220 msec								
0K								
Directory 11274334386847219714 11274334386847219713 /uttest/oss	11 caoj	ie staff	1599545017615	1599545017615		Finalized	WARM	
Directory 11274334386847219719 11274334386847219713 /uttest/oss2	11 caoj	ie staff	1599545017654	1599545017654		Finalized	WARM	
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir			1599545017636	1599545017636		Finalized	WARM	
File 11274334386847219715 11274334386847219714 /uttest/oss/file1	20 caoj		1599545017632	1599545017632		Finalized	WARM	
File 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2			1599545017642	1599545017642		Finalized	WARM	
File 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3		ie staff	1599545017651	1599545017651		Finalized	WARM	
Directory 11274334386847219728 11274334386847219719 /uttest/oss2/dir			staff 159954	5017654 1599545	017654		ed	WARM
File 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2			1599545017658	1599545017658		Finalized	WARM	
File 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3			1599545017666	1599545017666		Finalized	WARM	
Directory 11274334386847219713 17672823557433851905 /uttest 0	aojie staf	f 1599545	5017615 159954	5017615		d WARM		
Time taken: 10.734 seconds, Fetched: 10 row(s)								

# 3.2.9. JindoFS Credential Provider使用说明

JindoFS的数据存储在OSS中,如果您需要访问JindoFS的数据,需要提供OSS的AccessKey才能访问。Smartdata 3.4.0及后续版 本支持JindoFS Credential Provider,您可以通过配置JindoFS Credential Provider,将加密后的AccessKey信息添加至文件中,以 避免泄露AccessKey信息。

# 配置JindoFS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smartdata-site服务配置。
  - i. 单击配置页签。
  - ii. 在服务配置区域, 单击smart dat a-sit e页签。
- 3. 添加配置信息。
  - i. 在smartdata-site页签,单击右上角的自定义配置。
  - ii. 在新增配置项对话框中,新增如下配置。

参数	描述
fs.jfs.credentials.provider	配置com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential。例如, com.aliyun. emr.fs.auth.TemporaryAliyunCredentialsProvider,c om.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,c vider,com.aliyun.emr.fs.auth.EnvironmentVariable CredentialsProvider 。 Provider详情请参见Provider类型。

#### ⅲ. 单击确定。

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

### Provider类型

• TemporaryAliyunCredentialsProvider

适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsPr ovider
fs.jfs.accessKeyId	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。
fs.jfs.securityToken	OSS的SecurityToken(临时安全令牌)。

• SimpleAliyunCredentialsProvider

适合使用长期有效的AccessKey访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvid er
fs.jfs.accessKeyld	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。

#### • Environment VariableCredent ialsProvider

### 该方式需要在环境变量中配置以下参数。

参数	参数说明		
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EnvironmentVariableCredential sProvider		
ALIYUN_ACCESS_KEY_ID	OSS的AccessKey ID。		
ALIYUN_ACCESS_KEY_SECRET	OSS的AccessKey Secret。		
	OSS的SecurityToken(临时安全令牌)。		
ALIYUN_SECURITY_TOKEN	⑦ 说明 仅配置有时效Token时需要。		

#### • JindoCommonCredentialsProvider

该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.JindoCommonCredentialsProvi der
jindo.common.accessKeyld	OSS的AccessKey ID。
jindo.common.accessKeySecret	OSS的AccessKey Secret。
jindo.common.securityToken	OSS的SecurityToken(临时安全令牌)。

#### • EcsStsCredentialsProvider

该方式无需配置AccessKey,可以免密方式访问OSS。

参数	参数说明
fs.jfs.credentials.provider	${\tt com.aliyun.emr.fs.auth.EcsStsCredentialsProvider}$

# 3.2.10. JindoFS Block模式加密使用说明

JindoFS Block模式支持文件加密,加密机制和使用方法与Apache HDFS的Encryption Zone类似。加密密钥通过密钥管理服务 (KMS)统一管理,您可以对有敏感数据的目录设置加密策略,然后就可以透明地在该目录下加密写入的数据和解密读取的数据,无需更改您的代码。

# 前提条件

- 已创建集群,详情请参见创建集群。
- 已开通密钥管理服务(KMS),详情请参见开通密钥管理服务。

# 背景信息

Block模式加密架构图如下:



# 配置JindoFS使用阿里云KMS

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击namespace页签。
- 3. 添加配置信息。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下配置。

参数	描述
crypto.provider.type	Provider的类型,仅支持ALIYUN。
crypto.provider.endpoint	KMS的公网接入地址。详情请参见 <mark>调用方式</mark> 。
crypto.provider.kms.accessKeyld	访问KMS的AccessKey ID。
crypto.provider.kms.accessKeySecret	访问KMS的AccessKey Secret。

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 在右上角选择操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅳ. 在确认对话框中, 单击确定。

# 使用JindoFS KeyProvider

Jindo KeyProvider负责对接KMS,加密密钥存储在KMS。KeyProvider基于KMS提供新增密钥、查询密钥和轮换密钥等功能。

• 新增密钥: 传入keyldName, 创建一个新的密钥。

jindo key -create -keyIdName <keyIdName>

⑦ 说明 本文示例中的<keyIdName>为您创建的密钥名称。

例如,执行以下命令新增policy\_test的密钥。

jindo key -create -keyIdName policy\_test

在阿里云KMS控制台,您可以看到新增了一个名为policy\_test的密钥。

密钥管理服务控制台	创建凭据
用户主密钥	凭据管家最新上线全托管RDS凭据,支持定时更新RDS数据库口令,减少数据库口令泄漏风险。
凭据	名称
证书	policy_test 普通凭据

• 查询密钥: 查看当前存在的密钥名。

jindo key -list

#### 返回信息如下:

Listing Keys: policy\_test policy\_test2

• 轮换密钥:您可以根据Key ID定期更换密钥。更新密钥后Key Version会随之发生变化,即文件在加密时,使用最新的密钥进行

#### 加密, 文件在解密时使用现有文件的密钥版本进行解密。

jindo key -roll -keyIdName <keyIdName>

#### 例如,执行以下命令轮换密钥policy\_test。

jindo key -roll -keyIdName policy\_test

# 在阿里云KMS控制台,您可以看到密钥*policy\_test*的版本状态已经更新,之前的版本状态变成了ACSPrevious,新的版本状态为ACSCurrent。

密钥管理服务控制台	← policy_test			
用户主密钥	任据法律			
凭据	九酒井旧			
证书	凭据信息	名称	policy_test	
		ARN	acs:kms:cn-hangzh	policy_test
		加密密钥	系统托管密钥	
		描述信息	Encryption Key for JindoFS File E	<u>/</u>
	◇ 版本列表 ⑦			
	存入凭据值			
	版本号			版本状态
	1613802530			ACSCurrent   状态管理
	1613801564			ACSPrevious   状态管理

# 管理JindoFS加密策略

您可以根据以下命令,设置和查看加密策略:

• 设置加密策略

jindo jfs -setCryptoPolicy -keyIdName <keyIdName> <path>

⑦ 说明 本示例的 <path> 为您访问JindoFS上文件的路径。例如*jfs://test/*。

● 查看加密策略

jindo jfs -getCryptoPolicy <path>

#### 设置和查看加密策略示例如下所示:

#### 1. 查看*jfs://test/*路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回信息显示为 {NONE} 。

2. 设置*jfs://test/*的加密策略。

jindo jfs -setCryptoPolicy -keyIdName policy\_test jfs://test/

#### 3. 进入bigboot目录,再次查看jfs://test/路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回如下信息。

SLF4J: Class path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/b2jindosdk/3.4.0-hadoop3.1/package/b2jindosdk-3.4 .0-hadoop3.1/lib/jindo-distcp-3.4.0.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/hadoop/3.2.1-1.0.1/package/hadoop-3.2.1-1.0.1/sha re/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html#multiple bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] 21/03/12 13:52:34 WARN: Unable to load native-hadoop library for your platform... using builtin-java clas ses where applicable 21/03/12 13:52:35 INFO: Jboot log name is /var/log/bigboot/jboot-20210312-135234-12953.LOG 21/03/12 13:52:35 INFO: Write buffer size 1048576, logic block size 134217728 21/03/12 13:52:35 INFO: cmd=getFileStatus, src=jfs://test/, dst=null, size=0, parameter=null, time-in-ms= 7. version=3.4.0 21/03/12 13:52:35 INFO: cmd=getCryptoPolicy, src=jfs://test/, dst=null, size=0, parameter=, time-in-ms=2, version=3.4.0 The crypto policy of path: jfs://test/ is {cipherSuite: AES\_CTR\_NOPADDING\_256, keyIdName: policy test2, k eyIdVersion: null, edek: , iv: } 21/03/12 13:52:35 INFO: Read total statistics: oss read average <none>, cache read average <none>, read o ss percent <none>

#### 设置完成后即可正常读写该路径下的文件。

◦ 拷贝本地文件至HDFS。

hadoop fs -put test.log jfs://test/

○ 展示文件内容。

hadoop fs -cat jfs://test/test.log

# 3.3. JindoFS Cache模式

# 3.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓 存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主 要介绍JindoFS的缓存模式及其使用方式。

#### 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

### 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

#### 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

#### 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的集群管理页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smar	tData ∽	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 述			服务配置
配置搜	素 \		a	全部 smartdata-site namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例		
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>		
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。		
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache		

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。

此配置无需重启Smart Dat a服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后, Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

服务配置					
全部 smartdata-site namespace	client <b>storage</b>				
	0				
	storage.watermark.low.ratio	0.2	0		
	storage.watermark.high.ratio	0.4	0		
	storage.oss.upload.threads	20	0		
参数		描述			
storage.watermark.high.ratio		表示磁盘使用量的上水位比例 用的磁盘空间到达上水位即4	列,每块数据盘的JindoFS数据目录占 会触发清理。默认值:0.4。		
storage.watermark.low.ratio		表示使用量的下水位比例,触 JindoFS数据目录占用空间清	触发清理后会自动清理冷数据,将 理到下水位。默认值:0.2。		

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

#### 2. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击**确定**。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - Ⅳ. 在确认对话框中,单击确定。

#### 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。

参数	参数说明
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击namespace页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。

# 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在服务配置区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

● 在服务配置区域的smart dat a-sit e页签, 配置以下参数。

参数	参数说明	
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。	
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。	
fs.oss.committer.magic.enabled	② 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。	

# 3.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce (简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

# 前提条件

适用环境:ECS(EMR环境外)+Hadoop+JavaSDK。

### 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smartdata-aliyun-jfs-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

#### 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击**完成**。

#### 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击确定。

#### 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏,单击实例与镜像 > 实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

# 步骤四:在ECS上设置环境变量

#### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP CLASSPATH=\$HADOOP CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

# 步骤五:测试免密方式访问的方法

#### 1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

### JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
        while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
    }
}
```

# 3.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

#### 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 ismartdata-site namespace client istorage

3. 配置如下参数。

i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: I true:打开AuditLog功能。 I false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏,单击基础设置 > 生命周期,在生命周期单击设置。
- iv. 单击创建规则,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

#### 示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> s	how tables	;		
database	tableN	ame	isTemporary	
default audi	t_log	false		
default audi	t_log_sour	ce	false	
default fs_i	mage	false		

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sq	l> select * from	n audit_log_sou	urce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	m audit_log	lim	it 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true roo	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	0:50:11.950	true roo	ot (	auth:SIMPL	LE)	192.	- 10 C	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true roo	ot (	auth:SIMPL	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	1:26:06.469	true roo	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true roc	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:11.320	true roc	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:14.368	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:14.393	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:16.230	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:16.255	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ds, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 3.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

### 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

### ↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后,MapReduce Application Master或Spark Driver执行最终的Job Commit操作时,会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

#### 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.jindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

置mapreduce.outputcommitter.factory.scheme.oss为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。

- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

#### 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

### 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

#### 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。

```
ii. 单击配置页签。
```

- iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

```
默认值为8。
```

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

# 3.3.5. JindoFS OSS Credential Provider使用说明

Smart Dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

# 使用限制

JindoRangerCredentialsProvider和AssumeRoleStsCredentialsProvider仅适用于SmartData 3.8.0及后续版本。

# 配置JindoFS OSS Credential Provider

- 1. 进入SmartData服务的配置页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,单击**集群服务 > SmartData**。
  - vi. 单击**配置**页签。
- 2. 根据配置方式修改或新增配置信息。

配置方式	描述
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数fs.jfs.cache.oss.credentials.provider,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。

配置方式	描述					
	新增配置项的操作步骤如下: i.在smartdata-site页签,单击右上角的自定义配置。					
	↓ 注意 JindoRangerCredentialsProvider类型需要在namespace页签添加自定义配置。					
	ii. 在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。					
按照Bucket配置	例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。					
	⑦ 说明 fs.jfs.cache.oss.bucket.XXX.credentials.provider中的XXX为OSS的 Bucket名称。					
	iii. 单击确定。					

#### 3. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	<ul> <li>适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需在smartdata-site页签新增以下配置:</li> <li>fs.jfs.cache.oss.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>fs.jfs.cache.oss.accurityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需在 <b>smartdata-site</b> 页签新 增以下配置: • fs.jfs.cache.oss.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。

### Smart Dat a Smart Dat a 3.8.x

# E-MapReduce

类型	描述
Environment Variable Credentials Provi der	<ul> <li>该方式需要在环境变量中配置以下参数:</li> <li>fs.jfs.cache.oss.credentials.provider: 设置 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效的Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需在smartdata-site页签新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。
JindoRangerCredent ialsProvider	该方式适用于通过配置Ranger来控制用户访问OSS权限的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.JindoRangerCredentialsProvider,并需在 <b>namespace</b> 页签新增参 数namespace.oss.permission.method,参数值为ranger的配置项。 ⑦ 说明 JindoRangerCredentialsProvider类型,添加完自定义配置后,必须启动 JindoFS Namespace服务。重启服务详情请参见启动JindoFS Namespace服务。
AssumeRoleStsCredentialsProvider	<ul> <li>该方式适用于获取一个扮演RAM角色的临时AccessKey访问OSS的情况。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.AssumeRoleStsCredentialsProvider,并需在smartdata-site页签 新增以下配置:</li> <li>assume.role.sts.accessKeyId: 阿里云STS (Security Token Service)的AccessKey ID。</li> <li>assume.role.sts.accessKeySecret: 阿里云STS的AccessKey Secret。</li> <li>assume.role.sts.endpoint: 阿里云STS的Endpoint,详情请参见接入地址。</li> <li>assume.role.roleArn: 要扮演RAM角色ARN。格式为 acs:ram::\$accountID:role/\$roleName。查看ARN详情请参见如何查看RAM角色的ARN?</li> <li>assume.role.roleSessionName:角色会话名称。该参数为自定义参数,例如:用户名。</li> </ul>

# 按照Bucket配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型

描述

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需在 <b>smartdata-site</b> 页 签新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken(临时安全令 牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需在 <b>smartdata-site</b> 页签新 增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
EnvironmentVariableCredentialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数:</li> <li>fs.jfs.cache.oss.bucket.XXX.credentials.provider:设置 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需在smartdata-site页签</li> <li>新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。
JindoRangerCredentialsProvider	该方式适用于通过配置Ranger来控制用户访问OSS权限的情况。 需要在fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.JindoRangerCredentialsProvider,并需在 <b>namespace</b> 页签新增参 数namespace.oss.permission.method,参数值为ranger的配置项。

类型	描述
AssumeRoleStsCredentialsProvider	该方式适用于获取一个扮演RAM角色的临时AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.AssumeRoleStsCredentialsProvider,并需在 <b>smartdata-site</b> 页签 新增以下配置: • assume.role.sts.accessKeyId:阿里云STS(SecurityTokenService)的AccessKeyID。 • assume.role.sts.accessKeySecret:阿里云STS的AccessKeySecret。 • assume.role.sts.endpoint:阿里云STS的Endpoint,详情请参见接入地址。 • assume.role.roleArn:要扮演的RAM角色ARN。格式为 acs:ram::\$accountID:role/\$roleName。查看ARN详情请参见如何查看RAM角色的ARN? • assume.role.roleSessionName:角色会话名称。该参数为自定义参数,例如:用户名。

# 启动JindoFS Namespace服务

JindoRangerCredentialsProvider类型配置完成后,必须启动JindoFS Namespace服务。

- 1. 在SmartData服务的配置页面,选择右上角的操作 > 重启Jindo Namespace Service。
- 2. 在执行集群操作对话框中,输入执行原因,单击确定。
- 3. 在弹出的确认对话框中, 单击确定。

# 3.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

### 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web U功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview								
Start Time:	Fri Oct 16 12:29:24 2020							
Status:	Active							
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)							
Node:	Live Nodes: [2], Decommission Nodes: [0]							
Version:	3.0.0							
Build No:	fa0ea608a4;							

Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespa	Namespace Info (1)							
Namespace:	Namespace: jfs://test/							
Namespaces:	Namespaces: test							
Mode:	BLOCK_MODE							
Backend URI:	oss://							
Summary: Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27								

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

# Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

#### • 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 3.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

### 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



### 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜	素			全部 smartdata-site namespace client storage
请输入			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。

#### 5. 重启配置。

- i. 单击右上角的操作 > 重启 Jindo Namespace Service。
- ii. 输入执行原因, 单击**确定**。
- 开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见概述。

ii. Ranger UI添加HDFS service。

Ranger	Access Manager	🗅 Audit	Settings				👸 admin
Service Ma	nager						
Service Ma	nager						🛛 İmport
B	IDFS		+ 4 2	B HBASE	+ 20	I HIVE	+ 2 2
				emr-hbase	• 7	emr-hive	<ul> <li>Image: A state of the state of</li></ul>
				emr-noase	· · · · · · · · · · · · · · · · · · ·	emr-nive	

#### iii. 配置相关参数。

参数	描述	
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}/。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal	天指它	
dfs.namenode.kerberos.principal		
dfs.secondary.namenode.kerberos.principal	1.秋一。	
Add New Configurations		

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

# 3.4. JindoTable

# 3.4.1. 开启native查询加速

JindoTable通过Native Engine,支持对Spark、Hive或Presto上ORC或Parquet格式文件进行加速。本文为您介绍如何开启native 查询加速,以提升Spark、Hive和Presto的性能。

#### 前提条件

已创建集群,且ORC或Parquet文件已存放至JindoFS或OSS,创建集群详情,请参见创建集群。

#### 使用限制

- 不支持对Binary类型文件进行加速。
- 不支持分区列的值存储在文件中的分区表。
- 不支持EMR-5.X系列及后续版本的E-MapReduce集群。
- 不支持代码spark.read.schema (userDefinedSchema)。
- 支持Date类型区间为1400-01-01到9999-12-31。
- 同一个表中查询列不支持区分大小写。例如, NAME和name两个列在同一个表中无法使用查询加速。
- Spark、Hive和Presto服务支持的引擎和存储格式如下所示。

引擎	ORC	Parquet
Spark2	支持	支持
Spark3	支持	支持
Presto	支持	支持
Hive2	不支持	支持
Hive3	不支持	支持

• Spark、Hive和Presto服务支持的引擎和存储文件系统如下所示。

引擎	OSS	JFS	HDFS
Spark2	支持	支持	支持
Presto	支持	支持	支持
Hive2	支持	支持	不支持
Hive3	支持	支持	不支持

# 提升Spark性能

1. 开启JindoTable ORC或Parquet加速。

? 说明

- 因为查询加速使用的是堆外内存,所以在Spark任务中建议添加配置 --conf spark.executor.memoryOverhead=
   4g ,提高Spark申请额外资源用来进行加速。
- Spark读取ORC或Parquet时,需要使用DataFrame API或者Spark-SQL。

0	全局设置	
---	------	--

- a. 进入详情页面。
  - a. 登录阿里云E-MapReduce控制台。
  - b. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - c. 单击上方的集群管理页签。
  - d. 在集群管理页面,单击相应集群所在行的详情。
- b. 修改配置。
  - a. 在左侧导航栏,选择集群服务 > Spark。
  - b. 在Spark服务页面,单击配置页签。
  - c. 在搜索区域,搜索参数spark.sql.extensions,修改参数值 为io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtension。
- c. 保存配置。
  - a. 单击保存。
  - b. 在确认修改对话框中, 输入执行原因, 单击确定。
- d. 重启ThriftServer。
  - a. 在右上角选择操作 > 重启ThriftServer。
  - b. 在执行集群操作对话框中,输入执行原因,单击确定。
  - c. 在确认对话框中, 单击确定。
- Job级别设置

使用spark-shell或者spark-sql时,可以添加Spark的启动参数。

作业详情请参见Spark Shell作业配置或Spark SQL作业配置。

- 2. 检查开启情况。
  - i. 登录Spark History Server UI页面。 登录详情请参见访问链接与端口。

#### ii. 在Spark的SQL页面, 查看执行任务。

当出现JindoDataSourceV2Scan时,表示开启成功。否则,请排查步骤1中的操作。



# 提升Presto性能

□ 注意 Presto查询并发较高,且查询加速使用堆外内存,因此使用查询加速时内存配置必须大于10 GB。

因为Presto已经内置JindoTable native加速的 catalog: hive-acc ,所以您可以直接使用 catalog: hive-acc 来启用查询加 速。

示例如下。

presto --server emr-header-1:9090 --catalog hive-acc --schema default

⑦ 说明 目前使用Presto查询加速功能时,暂不支持读取复杂的数据类型,例如Map、Struct或Array。

# 提升Hive性能

↓ 注意 如果您对作业稳定性要求较高时,建议不要开启native查询加速。

您可以通过以下两种方式提升Hive性能:

• 控制台方式

在控制台Hive服务的配置页面,搜索并修改自定义参数hive.jindotable.native.enabled为true,保存配置后,重启服务使 配置生效,此方式适用于Hive on MR和Hive on Tez。

<返回 🙀 Hive マ ●正常		O 查看操作历史
状态: 部署拓扑 配置 配置修改历史 元数据		
配置过滤	服务配置	
配置搜索 hive.jindotable.native.enabled	全部 hive-site	
配置范围	hive.jindotable.native.enabled true	\$ Ø

• 命令行方式

您可以直接在命令行中设置 hive.jindotable.native.enabled 为 true 来启用查询加速。因为EMR-3.35.0及后续版本已 经内置JindoTable Parquet加速的插件,所以您可以直接设置该参数。

set hive.jindotable.native.enabled=true;

⑦ 说明 目前使用Hive查询加速功能时,暂不支持读取复杂的数据类型,例如Map、Struct或Array。

# 3.4.2. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

# 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partitionCol2=2,... 的格式。

#### -accessStat

● 语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。
```
jindo table -accessStat -d 7 -n 20
```

- -cache
- 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

```
表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。
```

• 示例:缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

- -uncache
- 语法

jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示删除集群本地磁盘上指定表或分区的缓存数据。

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partition Col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

- -archive
- 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

- -unarchive
- 语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName> [-p] <partitionSpec>

功能

```
表示将归档数据转为标准存储。
```

-o 将归档数据转为解冻, -i 将归档数据转为低频。

示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

```
jindo table -unarchive -i -t db1.t2
```

- -status
- 语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

。 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

### -optimize

● 语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

### -showTable

- 语法
  - jindo table -showTable {-t} <dbName.tableName>
- 功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例:展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

```
-showPartition
```

语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

jindo table -showPartition -t db1.t1 -p date=2020-10-12

- -listTables
- 语法

jindo table -listTables [-db] [dbName]

功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

• 示例:

> 文档版本: 20220629

#### ○ 展示default库中的表。

jindo table -listTables

○ 列出数据库db1中的表。

```
jindo table -listTables -db db1
```

### -dumpmc

语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} table} <tablename> {-p} <partitionSpec> {-f} <csv tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络Tunnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-p	分区信息 。例如 pt=xxx , 多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

功能

- 表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。
- 示例:
  - Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord
```

• Dumpmc Maxcompute表(CSV格式)至OSS存储。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv

## 3.4.3. JindoTable SDK模式归档和解冻命令介绍

JindoTable SDK模式提供archiveTable和unarchiveTable命令,可以在不依赖Jindo Namespace Service的情况下进行归档和解冻 等操作。本文为您介绍archiveTable和unarchiveTable命令的使用方法。

#### 前提条件

- 已创建集群,详情请参见创建集群。
- 待归档的数据必须是表数据(可以是分区表或非分区表),且已经位于阿里云对象存储OSS。

### 背景信息

JindoTable原有archive和unarchive命令可以对OSS上的表或分区进行归档或解冻等操作,但archive和unarchive命令依赖 Smart Dat a组件Jindo Namespace Service。现在新增的archiveTable和unarchiveTable命令,可以在不依赖Jindo Namespace Service的情况下进行归档和解冻等操作。

新增的archiveTable和unarchiveTable命令与原有archive和unarchive命令的主要区别为:

- 可以在未部署SmartData服务的集群上执行。例如,非EMR的用户自建集群。
- 可以通过传入过滤参数,一次应用于大量分区,多线程执行。如果本地多线程仍不能满足需求,还可以启动MapReduce任务 在整个集群上执行。

原有archive和unarchive命令的详细信息,请参见JindoTable使用说明。

### 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持新增的archiveTable和unarchiveTable命令。

### archiveTable命令

archiveTable命令可以对OSS上的表或分区进行归档。

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,获取帮助信息。

jindo table -help archiveTable

### archiveTable命令语句格式如下所示。

```
-archiveTable -t <dbName.tableName> \
-i/-a/-ca \
[-c "<condition>" | -fullTable] \
[-b/-before <before days>] \
[-p/-parallel <parallelism>] \
[-mr/-mapReduce] \
[-e/-explain] \
[-w/-workingDir <working directory>] \
[-1/-logDir <log directory>]
```

参数	描述	是否必选参数
-t <dbname.tablename &gt;</dbname.tablename 	待归档的表名称,格式为 数据库名.表名 。 数据库和表名之间以半角句号(.)分隔。表可以是分区表或非分区表。	是
-i/-a/-ca	目标存储方式。支持如下方式: • -i: 低频 (Infrequent Access, IA)存储。 • -a: 归档 (Archive)存储。 • -ca: 冷归档 (Cold Archive)存储。 如果指定-i则表示低频存储,会跳过已经处于归档或冷归档存储的文件。如果指定-a则 表示归档存储,会跳过冷归档的文件。	是
-c " <condition>"   - fullTable</condition>	<ul> <li>-fullTable 和 -c "<condition>" 只需提供一个,即要么指定 -c "<condition>" ,要么指定 -fullTable 。</condition></condition></li> <li>* 指定 -fullTable 时,则为归档整表,既可以是非分区表也可以是分区表。</li> <li>* 指定 -c "<condition>" 时,则提供了一个过滤条件,用来选择希望归档的分区,支持常见运算符,例如大于号 (&gt;)。</condition></li> <li>例如,数据类型为String的分区ds,希望分区名大于 'd',则代码为 -c " ds &gt; ' d' " 。</li> </ul>	是

参数	描述	是否必选参数
-b/before <before days&gt;</before 	只有创建时间距离现在超过一定天数的表或分区才会被归档。	否
-p/-parallel <parallelism></parallelism>	归档操作的并行度。	否
-mr/-mapReduce	使用Hadoop MapReduce而非本地多线程来归档数据。	否
-e/-explain	如果出现该选项,则为解释(explain)模式,只会显示待移动的分区列表,而不会真 正移动数据。	否
-w/-workingDir	只在MapReduce作业时使用,为MapReduce作业的工作目录。必须有读写权限,工作 目录可以非空(作业执行过程中会创建临时文件,执行完毕会清理临时文件)。	否
-l/-logDir <log directory&gt;</log 	指定Log文件的目录。	否

### unarchiveTable命令

unarchiveTable命令与archiveTable命令格式基本一致,但效果相反。unarchiveTable命令可以对OSS上的表或分区进行解冻。

- 1. 通过SSH方式登录集群,详情请参见<mark>登录集群</mark>。
- 2. 执行以下命令, 获取帮助信息。

jindo table -help unarchiveTable

unarchiveTable命令语句格式如下所示。

```
-archiveTable -t <dbName.tableName> \
-i/-a/-o/-cr \
[-notWait] \
[-c "<condition>" | -fullTable] \
[-b/-before <before days>] \
[-p/-parallel <parallelism>] \
[-mr/-mapReduce] \
[-e/-explain] \
[-w/-workingDir <working directory>] \
[-1/-logDir <log directory>]
```

unarchiveTable命令与archiveTable命令参数只有以下两处区别:

- 没有必选参数 -i/-a/-ca, 而被可选参数-i/-a/-o/-cr替代。
- 多了可选参数-notWait。

参数	描述	是否必选参数
-i/-a/-o/-cr	转换存储类型,均适用于冷归档。 • 如果不指定-i/-a/-o/-cr参数,则转换存储格式为标准(Standard)存储。	
	⑦ 说明 只有当冷归档文档处于完全解冻状态时,才可以转换到标准 (Standard)、低频(Infrequent Access,IA)或归档(Archive)存储类型。	
	<ul> <li>如果指定-i/-a/-o/-cr参数,相关参数描述如下:</li> <li>指定-i参数,则转换存储格式为低频 (Infrequent Access, IA)存储,跳过原本为标 准存储的文件。对冷归档 (Cold Archive)有效。</li> <li>指定-a参数,则转换存储格式为归档,唯一作用是将冷归档 (Cold Archive)文件改为 归档,跳过原本为标准或低频存储的文件。</li> <li>指定-o参数,则仅做解冻 (Restore)操作。跳过原本为标准存储或低频存储的文件。已经处于解冻状态的文件也会被跳过,即不会重复解冻。</li> <li>指定-cr参数,则用来检查分区下文件的解冻任务是否完成。</li> </ul>	否
	- 旧龙 51岁双,则用不恒星刀 617 入下的府位力定百元成。	

参数	描述	是否必选参数
-notWait	只对解冻(Restore)操作有效,如果指定该参数,则只发送解冻命令,而不等待解冻任务 完成。通常用于冷归档 (Cold Archive) 文件的解冻。	否

## 3.4.4. JindoTable MoveTo命令介绍

MoveTo命令可以实现表和分区数据的迁移功能。本文为您介绍MoveTo命令的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 背景信息

MoveTo命令可以在拷贝底层数据结束后,自动更新元数据,使表和分区的数据完整地迁移到新路径;可以通过条件筛选,一次 拷贝大量分区。在数据迁移过程中,还使用了多种措施保护数据的完整性,确保数据安全。

### 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持使用MoveTo命令。

### 使用MoveTo命令

○ 注意 集群上每次仅允许运行一个MoveTo进程。如果集群上有正在运行的MoveTo进程,启动新的MoveTo进程时会因为获取不到配置锁而退出,并告知正在运行的MoveTo进程。此时,您可以终止掉正在运行的MoveTo进程,启动新的MoveTo进程,或者等待正在运行的MoveTo进程结束。

- 1. 通过SSH方式登录集群,详情请参见<mark>登录集群</mark>。
- 2. 执行以下命令, 获取帮助信息。

jindo table -help moveTo

帮助信息类似如下所示。

<dbname.tablename< th=""><th>&gt; The table to move.</th></dbname.tablename<>	> The table to move.
<destination path<="" td=""><td>&gt; The destination base directory which is always at the</td></destination>	> The destination base directory which is always at the
	same level of a 'table location', where the moved
	partitions or un-partitioned data would located in.
<condition>/-full</condition>	Table A filter condition to determine which partitions should
	be moved, supporting common operators (like $'>'$ ) and
	built-in UDFs (like to_date) (UDFs not supported
	yet), while -fullTable means that all partitions (or
	a whole un-partitioned table) should be moved. One but
	only one option must be specified among $-c$
	" <condition>" and -fullTable.</condition>
<before days=""></before>	Optional, saying that table/partitions should be moved
	only when they are created (not updated or modified)
	more than some days before from now.
<parallelism></parallelism>	The maximum concurrency when copying partitions, 1 by
	default.
<oss stor<="" td=""><td>age policy&gt;: Storage policy for OSS destination, which can be Standard</td></oss>	age policy>: Storage policy for OSS destination, which can be Standard
(by default), I	A, Archive, or ColdArchive. Not applicable for destinations other
than OSS. NOTE:	if you are willing to use ColdArchive storage policy, please
make sure that	Cold Archive has been enabled for your OSS bucket.
-o/-overWrite	Overwriting the final paths where the data would be moved.
	For partitioned tables this overwrites partitions' locations
	which are subdirectories of <destination path="">; for</destination>
	un-partitioned table this overwrites the <destination path=""></destination>
	itself.
-r/-removeSource	Let the source data be removed when the corresponding
	table/partition is successfully moved to the new destination.
	Otherwise (by default), the source data would be left as it
	was.
-skipTrash	Applicable only when [-r/-removeSource] is enabled. If
	present, source data would be immediately deleted from the
	file system, bypassing the trash.
-e/-explain	If present, the command would not really move data, but only
	prints the table/partitions that would be moved for given
	conditions.
<log directory=""></log>	A directory to locate log files, '/tmp/ <current user="">/' by</current>
	default.

### MoveTo命令语句如下所示。

```
jindo table -moveTo \
  -t <dbName.tableName> \
  -d <destination path> \
  [-c "<condition>" | -fullTable] \
  [-b/-before <before days>] \
  [-p/-parallel <parallelism>] \
  [-s/-storagePolicy <OSS storage policy>] \
  [-o/-overWrite] \
  [-r/-removeSource] \
  [-skipTrash] \
  [-e/-explain] \
  [-1/-logDir <log directory>]
```

参数	描述	是否必选参数
-t <dbname.tablename></dbname.tablename>	待移动的表名称,格式为数据库名.表名。 数据库和表名之前以半角句号(.)分隔。表可以是分区表或非分 区表。	是

参数	描述	是否必选参数
-d <destination path=""></destination>	待移动的目标位置。无论是移动分区还是移动非分区表的整表, 该位置都对应 "表" 一级的位置。如果移动的是分区,则分区的完 整路径是该路径+分区名。例如 <destination path="">/p1=v1/ p2=v2/ 。</destination>	是
-c " <condition>"   -fullTable</condition>	<ul> <li>两者必须且只能提供一个,即要么指定 -c "</li> <li><condition>",要么指定 -fullTable 。</condition></li> <li>指定 -fullTable 时,则为移动整表,既可以是非分区表也可以是分区表。</li> <li>指定 -c "<condition>"时,则提供了一个过滤条件,用来选择希望移动的分区,支持常见运算符,例如大于号(&gt;)。</condition></li> <li>例如,数据类型为String的分区ds,希望分区名大于'd',则代码为 -c "ds &gt; 'd' "。</li> </ul>	否
-b/before <before days=""></before>	仅创建时间距离现在超过一定天数的表或者分区才会被移动。	否
-p/-parallel <parallelism></parallelism>	迁移操作的并行度。	否
-s/-storagePolicy <oss storage<br="">policy&gt;</oss>	<ul> <li>拷贝到OSS时,在OSS上的存储策略。存储策略如下:</li> <li>Standard:归档存储。</li> <li>IA:低频 (Infrequent Access)存储。</li> <li>Archive:标准存储。</li> <li>ColdArchive:冷归档存储。</li> <li>⑦ 说明 使用前请确保OSS Bucket开通了该功能。</li> </ul>	否
-o/-overWrite	是否强制覆盖目标写入路径。如果是分区表,则只会清空待移动 分区的分区路径,不会清空整个表路径。	否
-r/-removeSource	移动完成,元数据也同步更新后,是否清理源路径。如果是分区 表,则只会清理成功移动的分区的源路径。	否
-skipT rash	清理源路径时是否跳过Trash。 ⑦ 说明 在指定了参数-r/-removeSource时适用。	쥼
-e/-explain	如果出现该选项,则为解释(explain )模式,只会显示待移动的 分区列表,而不会真正移动数据。	否
-l/-logDir <log directory=""></log>	指定Log文件目录。	否

### 配置锁目录

MoveTo工具实现了进程锁,需要提供一个HDFS的路径放置锁文件。默认情况下,该路径为hdfs:///tmp/jindotable-lock/。

↓ 注意 放置锁文件的路径只能是HDFS路径。如果您对该路径无操作权限时,可以按照如下步骤添加自定义配置,配置 该路径。

1. 进入HDFS服务页面。

i.

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的**集群管理**页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。

v. 在左侧导航栏,选择集群服务 > HDFS。

2. 修改配置。

i. 在HDFS服务的配置页面,单击hdfs-site或core-site页签。

ii. 单击右上角的自定义配置。

< 返回 <li> ↔ HDFS &gt; ● 正常</li>		◎ 查看操作历史 I 快捷链接 > ₽ 操作
状态 部署拓扑 配置修改历史		
<b>配置过述</b> 配置提素 端命入 Q	服务配置 全部 <mark>hdfs-site</mark>   kms-site   httpfs-site   hadoop-env   httpfs-env   ssl-client <mark>core-site</mark>	●         部署客户选配量         保77           目定义配量         日定义配量
配置范围	dfs.datanode.cache.revocation.timeout.ms 900000	Ø

iii. 在**新增配置项**对话框中,添加配置项jindotable.moveto.tablelock.base.dir,参数值为一个已存在的HDFS路径。

↓ 注意 自定义配置锁目录时,请确保整个集群的所有节点上不存在正在运行的MoveTo进程,否则可能导致 MoveTo执行失败,甚至导致数据污染。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。

## 3.4.5. JindoTable表或分区访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

#### 前提条件

已创建集群,详情请参见创建集群。

### 背景信息

JindoTable支持收集访问Hive表的记录,收集的数据保存在SmartData服务的Namespace中。

Smart Dat a 3.2.x版本开始支持Spark、Hive和Presto引擎,Spark和Presto的数据收集默认是打开的,如果需要关闭,请参见关闭 <mark>热度收集</mark>。Hive的数据收集默认是关闭的,如果需要打开,请参见<mark>开启Hive热度收集</mark>。

### 数据查询

JindoTable提供了命令方式查询热度信息。

语法

```
jindo table -accessStat <-d [days]> <-n [topNums]>
```

days和 topNums为正整数。当只设置天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

```
jindo table -accessStat -d 7 -n 20
```

JindoTable使用详情,请参见JindoTable使用说明。

### 开启Hive热度收集

1.

- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的**集群管理**页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改Hive的参数值。

- i. 在左侧导航栏,选择**集群服务 > Hive**。
- ii. 在Hive服务页面,单击配置页签。
- iii. 搜索参数hive.exec.post.hooks,在参数值后追加com.aliyun.emr.table.hive.HivePostHook。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - i. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
  - ii. 在执行集群操作对话框,输入执行原因。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

### 关闭热度收集

1.

- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。
- 。 Hive服务:
  - a. 在左侧导航栏,选择集群服务 > Hive。
  - b. 在Hive服务页面,单击配置页签。
  - c. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据		
配置过滤 配置搜索 hive.exec.post.hooks	服务配置 全部 l hive-site	
	hive-exec.post.hooks	k.HiveHook, com.aliyun.emr.table.hive.HivePostHook

- Spark服务:
  - a. 在左侧导航栏,选择**集群服务 > Spark**。
  - b. 在Spark服务页面,单击配置页签。
  - c. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSQLQueryListener。

状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置
記宣視家 spark.sql.queryExecutionListeners ② Q	全部 / spark-defaults
配置范围	spark.sql.queryExecutionListeners com.aliyun.emr.table.spark.SparkSQLQueryListener

- Presto服务:
  - a. 在左侧导航栏,选择集群服务 > Presto。
  - b. 在Presto服务页面, 单击配置页签。
  - c. 搜索参数event-listener.name, 删除参数值中的内容。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

- 7. 重启服务。
  - 。 Hive服务:
    - a. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中,单击确定。
  - Spark服务:
    - a. 在Spark服务页面,选择右上角的操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Presto服务:
    - a. 在Presto服务页面,选择右上角的操作 > 重启All Components。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中,单击确定。

## 3.4.6. JindoTable表或分区访问冷度收集

JindoTable表或分区的访问冷度收集功能可以为您维护表或分区上次的访问时间,从而筛选出最近没有被访问的数据,帮助您优 化数据存储方式,节约成本。例如,在数据分析中,您可以把部分不常用的分区数据移动到成本更低的存储介质以节约成本。

### 前提条件

已创建EMR-3.35.0及后续版本或EMR-4.9.0及后续版本,创建详情请参见创建集群。

### 背景信息

Smart Dat a 3.5.x版本开始支持Hive、Spark和Presto组件的冷度收集功能。该功能目前默认不开启,如果需要开启,请参见开启 Spark冷度收集、开启Hive冷度收集和开启Presto冷度收集。

⑦ 说明 因为冷度收集与热度收集使用相同的hooks或Listeners,所以开启组件的冷度收集时会同时打开热度收集功能。 表或分区访问热度收集的详情,请参见jindoTable表或分区访问热度收集。

### 使用限制

- 不支持DLF数据湖元数据。
- Hive CLI、HiveServer2、Spark SQL CLI、Spark Thrift server和Presto服务所在IP需要有权限访问元数据底层存储(MySQL或 RDS)。
- 仅支持Hive、Spark和Presto组件的冷度收集。

#### 数据查询

JindoTable提供了命令方式查询冷度信息。

语法

jindo table -leastUseStat -n <num> [-i/-ignoreNever]

num是显示的条目数量,应为正整数。-i/-ignoreNever为可选参数,如果设置该参数,则会过滤掉从未被访问过的表或分区。

功能

展示最久未被访问的表或分区。

• 示例:查询最久未被访问的表或分区的20条记录。

```
jindo table -leastUseStat -n 20
```

返回如下图所示三列结果。

tdb.t1 pid=20/qid=ten 2021-03-26 13:53:52 tdb.t2 2021-03-26 13:53:58

- 第一列为表的名字,格式:数据库名.表名。
- 。 第二列为分区名字,格式:第一分区列=列值/第二分区列=列值/...,如果表为非分区表则为空。
- 第三列为最近一次访问的时间,格式: yyyy-MM-dd HH:mm:ss。
  - ⑦ 说明 如果为分区表,则只显示到分区级别,表本身不会单独显示。

JindoTable使用详情,请参见JindoTable使用说明。

#### 开启Spark冷度收集

- 1. 进入Spark页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Spark。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.queryExecutionListeners,确保参数值包含com.aliyun.emr.table.sparkSparkSQLQueryListener,如果存 在多个Listeners时使用英文逗号(,)隔开。

< 返回 🖍 Spark 🗸 • 正常		④ 查看操作
状态 部署拓扑 配置 配置修改历史		
配置过滤	服务配置	
配置搜索 spark.sql.queryExecutionListeners Q Q	全部   spark-defaults	
配置范围	spark.sql.queryExecutionListeners com.aliyun.emr.table.spark.SparkSQLQueryLis	:ener,c 🛛

- 4. 添加自定义配置。
  - i. 在服务配置页面, 单击spark-defaults页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为spark.sql.query.update.access.time.enabled,Value为true。

新増配置项			×
* Key	* Value	描述	攝作
spark.sql.query.update.access.time.enabled	true		删除
添加			
			确定取消

- iv. 单击确定。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。

iii. 在确认对话框中, 单击确定。

### 开启Hive冷度收集

- 1. 进入Hive页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
  - ⅲ. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Hive。
- 2. 在Hive服务页面,单击配置页签。
- 3. 搜索参数hive.exec.post.hooks,确保参数值包含com.aliyun.emr.table.hive.HivePostHook,如果存在多个hooks时使用英文逗号(,)隔开。

<返回 🖓 Hive ➤ ●正常	
状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤 配置接索 hive.exec.post.hooks Q	服务配置 全部   hive-site
配置范围	hive.exec.post.hooks com.aliyun.emr.meta.hive.hook.LineageLoggerHook

- 4. 添加自定义配置。
  - i. 在服务配置页面,单击hive-site页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为hive.hook.update.access.time.enabled, Value为true。

新	新増配置项				
	* Кеу	* Value	海道	操作	
	hive.hook.update.access.time.enabled	true		删除	
漆加					
				确定取消	

- iv. 单击**确定**。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

#### 开启Presto冷度收集

- 1. 进入Presto页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > Presto**。
- 2. 在Presto服务页面,单击**配置**页签。

3. 搜索参数event-listener.name,确保参数值包含jindo-presto-post-event-listener。

<返回	
状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置
配置搜索	全部   event-listener,properties
event-listener.name © Q	
配置范围	event-listener.name jindo-presto-post-event-listener

- 4. 添加自定义配置。
  - i. 在服务配置页面,单击event-listener.properties页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为listener.update.access.time.enabled,Value为true。

亲	新增配置项 ×		×	
	* Key	* Value	描述	撮作
	listener.update.access.time.enabled	true		删除
	添加			
				确定取消

- iv. 单击**确定**。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中,输入执行原因,单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在确认对话框中,单击确定。

## 3.5. 工具集

## 3.5.1. Jindo sql命令介绍

Jindo sql命令是JindoFS自带的工具,方便您分析JindoFS访问日志、元数据和OSS访问日志。本文为您介绍如何使用Jindo sql命 令,分析JindoFS访问日志、元数据和OSS访问日志的数据。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 背景信息

您可以使用Jindo sql命令分析以下数据:

- 使用Jindo sql分析JindoFS访问日志
- 使用Jindo sql分析元数据
- 使用Jindo sql分析OSS访问日志

### 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持使用Jindo sql命令。

### 使用Jindo sql命令

1. 通过SSH方式登录集群,详情请参见登录集群。

### 2. 执行以下命令,启动jindo sql。

jindo sql

### jindo sql支持以下常用参数。

参数	描述
-f	指定运行的SQL文件。
-i	启动Jindo sql后自动运行初始化SQL脚本。
-d	参数设置为键值对的形式。例如, -d A=B 。

### Jindo sql内置表结构

● audit\_log\_source (分区表)

audit\_log\_source表用作JindoFS访问日志原始表。

参数	描述
datetime	时间格式yyyy-MM-dd HH:mm:ss。
allowed	本次操作是否被允许,取值如下: • true:允许本次操作。 • false:不允许本次操作。
ugi	操作用户(包含认证方式信息)。
ip	Client IP地址。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dst	目标路径,可以为空。
perm	操作文件的Permission信息。
date (分区列)	日志日期,格式为YYYY-mm-DD。

### audit\_log

audit\_log允许使用分区列进行分区过滤,用作JindoFS访问日志表。

参数	描述
datetime	时间格式yyyy-MM-dd HH:mm:ss。
allowed	本次操作是否被允许,取值如下: • true:允许本次操作。 • false:不允许本次操作。
ugi	操作用户(包含认证方式信息)。
ip	Client IP地址。
ns	Block模式namespace的名称。
cmd	操作命令。

### E-MapReduce

参数	描述
src	源路径。
dst	目标路径,可以为空。
perm	操作文件的Permission信息。
date (分区列)	日志日期,格式为YYYY-mm-DD。

### ● fs\_image (分区表)

### fs\_image用作转存image信息

参数	描述
atime	INode最近访问时间。
attr	文件相关属性。
etag	OSS的ETag值。
id	INode的ID。
mtime	INode的修改时间。
name	INode的名称。
owner	owner名称。
ownerGroup	owner组名称。
parentId	父节点的ID。
permission	操作文件的Permission信息。
size	INode的大小。
state	INode的状态。
type	INode的类型。
storagePolicy	存储策略。
namespace (分区列)	namespace名称。
datetime (分区列)	转存时间。

### oss\_access\_log\_source

如果开启分区表模式,则为分区表。oss\_access\_log\_source表用作OSS访问日志原始表。

参数	描述
line	原始日志。
bucket(分区列)	Bucket名称。
partition_date (分区列)	日志日期格式为YYYY-mm-DD。

### oss\_access\_log

如果开启分区表模式,允许使用分区列进行分区过滤。oss\_access\_log表用作OSS访问日志。

参数	描述
Remote_IP	请求者的IP地址。
Reserved	保留字段,固定值为-。
Reserved1	保留字段,固定值为-。
Time	OSS收到请求的时间。
Request_URI	包含query string的请求URL。OSS会忽略以x-开头的query string参数,但这个参数会被记录在访问日志中。所以您可以使用x-开头query string参数标记一个请求,然后使用这个标记快速查找该请求对应的日志。
HTTP_Status	OSS返回的HTTP状态码。
SentBytes	请求产生的下行流量。单位:Byte。
RequestTime	完成本次请求耗费的时间。单位:ms。
Referer	请求的HTTP Referer。
User_Agent	HTTP的User-Agent头。
HostName	请求访问的目标域名。
Request_ID	请求的Request ID。
LoggingFlag	是否已开启日志转存。
Requester	请求者的用户ID。取值-表示匿名访问。
Operation	请求类型。
Bucket	请求的目标Bucket名称。
Кеу	请求的目标Object名称。
Object Size	目标Object大小。单位:Byte。
Server_Cost_Time	OSS处理本次请求所花的时间。单位:毫秒。
ErrorCode	OSS返回的错误码。取值-表示未返回错误码。
RequestLength	请求的长度。单位: Byte。
UserlD	Bucket拥有者ID。
Delta_DataSize	Bucket大小的变化量。取值-表示此次请求不涉及Object的写入操作。
SyncRequest	请求是否为CDN回源请求。取值如下: • cdn: 请求是CDN回源请求。 • -: 请求不是CDN回源请求。
StorageClass	<ul> <li>目标Object的存储类型。取值如下:</li> <li>Standard:标准存储。</li> <li>IA:低频访问存储。</li> <li>Archive:归档存储。</li> <li>Cold Archive:冷归档存储。</li> <li>-:未获取Object存储类型。</li> </ul>

参数	描述
TargetStorageClass	是否通过生命周期规则或CopyObject转换了Object的存储类型。取值如下: o Standard:转换为标准存储。 o IA:转换为低频访问存储。 o Archive:转换为归档存储。 o Cold Archive:转换为冷归档存储 o -:请求不涉及Object存储类型转换操作。
TransmissionAccelerationAccessPoint	通过传输加速域名访问目标Bucket时使用的传输加速接入点。取值-表示未使用传 输加速域名或传输加速接入点与目标Bucket所在地域相同。 例如,请求者通过华东1(杭州)的接入点访问目标Bucket时,值为cn- hangzhou。
AccessKeyID	访问的AccessKey ID。
bucket (分区列)	Bucket名称。
partition_date (分区列)	日志日期格式为YYYY-mm-DD。

### 使用Jindo sql分析JindoFS访问日志

JindoFS为存储在OSS上的JindoFS访问日志文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃IP分析。您可以通过 jindo sql 命令,使用该功能。

⑦ 说明 已开启AuditLog功能,详情请参见AuditLog使用说明。

Jindo SQL相关命令示例如下:

• 执行如下命令,显示表。

show tables;

⑦ 说明 表结构信息,请参见Jindo sql内置表结构。

返回信息如下图所示。



• 执行如下命令,显示分区。

show partitions audit\_log\_source;

返回信息类似如下图所示。

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken: 0.031 seconds, Fetched 1 row(s)
```

• 执行如下命令,查询数据。

select \* from audit\_log\_source limit 10;

#### 返回信息类似如下图所示。

jindo-sql> select * from	n audit_log_sourc	e limit 1:	.0;				
datetime allowed	ugi ip	ns c	md src	dst perm	date		
2020-10-20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	0				
2020-10-20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null perm=null	2020-10-20						
2020-10-20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	0				
2020-10-20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null perm=null	2020-10-20						
2020-10-20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s dst=null	perm=root:root:r	wxr-xx	2020-10-	-20			
2020-10-20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null perm=null	2020-10-20						
2020-10-20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s dst=null	perm=root:root:r	wxr-xx	2020-10-	-20			

select \* from audit log limit 10;

返回信息类似如下。

jindo-sql> se	elect * from	n audit_l	og li	mit 10										
datetime	allowed	ugi	ip	ns	cmd	src	dst	perm	date					
2020-10-20 10	0:50:11.924	true	root	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	rwxrwx
r-x 2020-	-10-20													
2020-10-20 10	0:50:11.950	true	root	(auth:	SIMPLE)	192.	10.00	kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20 1:	1:26:06.445	true	root	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	rwxrwx
r-x 2020-	-10-20													
2020-10-20 12	1:26:06.469	true	root	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20 1:	1:26:11.295	true	root	(auth:	SIMPLE)	192.	1. A.	kugou	getFileStatusRequest	jfs://k		null	root:ro	oot:rw
xr-xx 2020-	-10-20													
2020-10-20 1:	1:26:11.320	true	root	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 12	1:26:14.368	true	root	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:rc	oot:rw
xr-xx 2020	-10-20													
2020-10-20 12	1:26:14.393	true	root	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 12	1:26:16.230	true	root	(auth:	SIMPLE)	192.	- 10 A	kugou	getFileStatusRequest	jfs://k		null	root:rc	oot:rw
xr-xx 2020-	-10-20													
2020-10-20 12	1:26:16.255	true	root	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
<b>T</b> 1 1 1 1	4 040		1 40											

• 执行如下命令,统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

### 使用Jindo sql分析元数据

JindoFS为JindoFS上的元数据文件提供SQL的分析功能,通过SQL分析相关表。您可以通过 jindo sql 命令,使用该功能。

```
⑦ 说明 已开启AuditLog功能,详情请参见AuditLog使用说明。
```

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,启动Jindo sql。

jindo sql

- 3. 查询Jindo SQL可以分析的表格。
  - 使用 show tables 命令,可以查看支持查询分析的表格。Jindo Sql内置了审计和元数据信息的分析功能,对应 audit\_log和fs\_image。

代码示例如下图所示。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

使用 show partitions fs\_image
 命令,可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -du
 mpMetadata 生成的数据。

代码示例如下图所示。

jindo-sql> show partitions fs_image;	
partition	
namespace=k /datetime=2020_10_20_10_47_14	
namespace=k /datetime=2020_10_20_10_50_36	
namespace=k /datetime=2020_10_20_10_52_06	
Time taken: 0.045 seconds, Fetched 3 row(s)	

4. 查询分析元数据信息。

Jindo SQL使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

代码示例如下图所示。

[root@en	nr-worke	-2 hadoo	op]# jin	ndo sql													
Spark ma	aster: ye	arn, Appi	lication	n Id: app													
jindo-se	al> show	tables;															
database		tableNar	me	isTempo	rary												
default	audit_lo	xg	false														
default	audit_le	g_source	e	false													
default	fs_image		false														
Time tak	cen: 0.34	15 second	ds, Fet	ched 3 ro	m(s)												
jindo-so	ql> sele	t * from	m fs_im	age limit	10;												
atime	attr	etag	id	mtime	name	owner	ownerGr	oup	parentId	permiss	ion	size	state	storage	Policy	type	name
space	datetim																
0			731107	600505189	9448	1603084	4070081	/tpcds.	/orc/5000/web	_returns/wr_	returned	_date_	sk=2450819	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			1653444	480419066	75495	160308	1071350	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_!	sk=2450820	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			731107	600505189	9470	1603084	4070185	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450821	root	root	334790	833296
5855433	489	0	Finali	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			119227	620234792	87249	1603084	4069581	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450822	root	root	334790	833296
5855433	489	Ø	Finali	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			107698	405188724	41036	1603084	4073592	/tpcds.	/orc/5000/web	_returns/wr_	returned	_date_	sk=2450823	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	pry	kugou	2020_10_20_	10_50_36							
0			269938	998662451	1354	160308	1068996	/tpcds	/orc/5000/web	_returns/wr_	returned	_date_s	sk=2450824	root	root	334790	833296
5855433	489	Ø	Finalia	zed	WARM	Directo	pry	kugou	2020_10_20_	10_50_36							
0			1192270	620234792	87307	160308	4069875	/tpcds.	/orc/5000/web	_returns/wr_	returned_	_date_:	sk=2450825	root	root	334790	833296
5855433	489	ø	Finali	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			154646	848201766	500Z	1603084	4072440	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450826	root	root	334790	833296
5855433	489	ø	Finali	zed	WARM	Directo	bry	kugou	2020_10_20_	10_50_36							
0			1653444	480419066	75460	1603084	4071170	/tpcds.	/orc/5000/web	_returns/wr_	returned,	_date_	sk=2450827	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	bry	kugou	2020_10_20_	10_50_36						224200	
0			7311070	600505189	9244	160308	1070572	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_s	SK=2450828	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							

例如: 根据某次转存的元数据信息统计该namespace下的目录个数。

jindo-sql> select count(\*) from fs\_image where type = "Directory" and namespace="kugou" and datetime="2020\_10\_20\_10\_47\_14"; count(1) 11837 Time taken:\_6.852 seconds, Fetched 1 row(s)

### 使用Jindo sql分析OSS访问日志

↓ 注意 分析OSS访问日志需要指定OSS访问日志目录和指定是否为分区表,指定分区表会自动按照Bucket或date进行日志归档,能够支持使用过滤语句指定查询某个分区,极大的提升了查询效率,但是开启分区表之后必须每次使用分区表模式,否则文件会被归档到目录导致部分数据无法查询。

JindoFS为存储在OSS上的OSS访问日志文件提供SQL的分析功能,通过SQL分析相关表。您可以通过\_\_\_jindo\_sql\_\_命令,使用该 功能。

⑦ 说明 namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

```
⑦ 说明 已开启日志转存,详情请参见日志转存。
```

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,启动Jindo sql。

jindo sql

3. 执行以下命令,指定日志存储路径和表类型。

jindo sql -d access\_log\_path=oss://test-sh/oss-accesslog -d partition.table.enabled=true

```
代码中的 access_log_path 为OSS访问日志存储路径, partition.table.enabled 指定是否为分区表, true表示为分区
表。
```

### 常见问题

• Q: 如何修改初始资源Jindo sql的启动参数?

A: 因为 Jindo sql 基于Spark的程序,所以初始资源可能较小,您可以通过环境变量 JINDO\_SPARK\_OPTS 来修改初始资源 Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.exec utor.cores=5 --conf spark.executor.memory=20G"

• Q: 如何使用Hive分析表?

A:为了避免污染Hive元数据,默认Hive看不到Default下的几个表,如果想使用Hive分析这些表,可以通过语句 show create table {table name} 查看表语句或者使用SQL创建新表,Hive需要执行加载外部表。

## 3.5.2. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

### 前提条件

已创建集群,详情请参见创建集群。

### 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

### 挂载

? 说明 依次在每个节点上执行挂载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

#### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

返回用户在服务端配置的所有命名空间列表。

test testcache

#### 2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

### 3. 创建目录。

mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/

### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

如果您想使用Python方式写入和读取文件,请参见如下示例:

### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

#### 返回如下信息。

my first file This file

### 卸载

```
? 说明 依次在每个节点上执行卸载操作。
```

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

## 3.5.3. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见<mark>登录集群</mark>。

2. 执行以下命令,获取帮助信息。

jindo distcp --help

#### 返回信息如下。

help - Print help text
src=VALUE - Directory to copy files from
dest=VALUE - Directory to copy files to
parallelism=VALUE - Copy task parallelism
outputManifest=VALUE - The name of the manifest file
previousManifest=VALUE - The path to an existing manifest file
requirePreviousManifest=VALUE - Require that a previous manifest is present if specified
copyFromManifest - Copy from a manifest instead of listing a directory
srcPrefixesFile=VALUE - File containing a list of source URI prefixes
srcPattern=VALUE - Include only source files matching this pattern
deleteOnSuccess - Delete input files after a successful copy
outputCodec=VALUE - Compression codec for output files
groupBy=VALUE - Pattern to group input files by
targetSize=VALUE - Target size for output files
enableBalancePlan - Enable plan copy task to make balance
enableDynamicPlan - Enable plan copy task dynamically
enableTransaction - Enable transation on Job explicitly
diff - show the difference between src and dest filelist
ossKey=VALUE - Specify your oss key if needed
ossSecret=VALUE - Specify your oss secret if needed
ossEndPoint=VALUE - Specify your oss endPoint if needed
policy=VALUE - Specify your oss storage policy
cleanUpPending - clean up the incomplete upload when distcp job finish
queue=VALUE - Specify yarn queuename if needed
bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed
s3Key=VALUE - Specify your s3 key
s3Secret=VALUE - Specify your s3 Sercet
s3EndPoint=VALUE - Specify your s3 EndPoint

### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp

② 说明 本文示例中的yourBucket Name是您OSS Bucket 的名称。

### --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制*/data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定\_\_\_\_srcPattern\_的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Yound 6 items									
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst		
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log		
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log		
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001		
09									
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt		
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt		

#### 执行以下命令,复制以log结尾的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPattern .\*\.lo
g --parallelism 20

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-

      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-

      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --deleteOnSuccess -parallelism 20

### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下:

Found 6 items								
-rw-rw-rw-	1	938 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/000151.sst.gz</yourbucketname>						
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/1.log.gz</yourbucketname>						
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/2.log.gz</yourbucketname>						
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/OPTIONS-000109.g</yourbucketname>						
Z								
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/emp01.txt.gz</yourbucketname>						
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly table/2017-02-01/03/emp06.txt.gz</yourbucketname>						

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。

### --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

### --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录, 查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly_table/manifest-2020-04-17.gz --parallel ism 20
```

#### 执行以下命令,查看运行期间复制的文件。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst

#### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

#### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
        Found 4 items

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

例如,执行以下命令,查看如下文件夹中的数据。

```
hdfs dfs -ls /data/incoming/hourly_table/2017-02-01/03
```

### 返回信息如下。

\_ ...

Found 8 items	S	
-rw-r	2 root hadoop	2252 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2 root hadoop	4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2 root hadoop	4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2 root hadoop	4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r	2 root hadoop	4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r	2 root hadoop	4891 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09		
-rw-r	2 root hadoop	1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2 root hadoop	1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

#### --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为DistCp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo DistCp的作业分配计划,以达到更好的DistCp性能。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --enableBalancePlan
--parallelism 20

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

### --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定 --enableDynamicPlan 来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

### --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的DistCp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis m 20

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yam队列的名称。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

#### --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

#### 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

● 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive --parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

#### 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

#### 命令示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --cleanUpPending --
parallelism 20
```

### 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里,避免每次使用时填写Accesskey。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

#### 此时代码示例如下。

jindo distcp /tmp/jindo-distcp-2.7.3.jar --src s3://smartdatal/ --dest s3://smartdatal/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

### 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG REDUCE=0
```

⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的 大小可能不相等。

## 3.5.4. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

### 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-distcp-<version>.jar。
   Hadoop 2.7及后续版本,请下载jindo-distcp-3.0.0.jar。

• Hadoop 3.x系列版本,请下载jindo-dist cp-3.0.0.jar。

#### 场景预览

Jindo DistCp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二: 使用Jindo Dist Cp成功导完数据后, 如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?
- 场景四: 成功导入HDFS数据至OSS, 数据不断增量增加, 在Dist cp过程中可能已经产生了新文件, 该使用哪些参数处理?
- 场景五:如果需要指定Jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六: 当通过低频或者归档形式写入OSS, 该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九:如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时,该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二: 如果Copy完文件, 需要删除原文件, 只保留目标文件时, 该使用哪些参数?
- 场景十三: 如果不想将OSS AccessKey这种参数写在命令行里, 该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret ),以及Endpoint信息,且该AccessKey具有写目标Bucket的权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

#### 本场景示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10
```

⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch

### 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

#### Distcp Counters

```
Bytes Destination Copied=11010048000
Bytes Source Read=11010048000
Files Copied=1001
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG REDUCE=0
```

#### 参数含义如下:

- Bytes Destination Copied: 表示目标端写文件的字节数大小。
- Bytes Source Read: 表示源端读文件的字节数大小。
- Files Copied:表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

### 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

## 在<mark>场景</mark>一的基础上,如果您的Dist cp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Dist cp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文 件的Copy。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table le --dest oss://yang-hhht/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa nifest --parallelism 20

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

### 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 参数含义如下:

- --outputManifest
   : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如manifest-2020-04-17.gz,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Distcp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife
st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest-2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

### 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid th 6 --parallelism 10

#### 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

### 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

#### 优化对比如下。



### 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly\_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10

### 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

### 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

### 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则, 正则表达式。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.*/([a-z]+).*.txt' --parallelism 20
```

### 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10
```

### 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

### Smart Dat a Smart Dat a 3.8.x

```
<configuration>
configuration>
 cname>fs.jfs.cache.oss-accessKeyId</name>
        <value>xxx</value>
        </property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
        </property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
        </property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
        </property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
        </property>
        configuration>
```

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

## 3.5.5. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存储成本,或者加速访问数据的速度。

### 使用Jindo jfs

执行以下命令,获取帮助信息。

jindo jfs -help archive

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

### Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p参数可以保证本地数据不受磁盘水位清理。

### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

### Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参 见对象存储OSS的存储类型介绍。

jindo jfs -archive -i|-a|-c <path>

指定以下参数时:

- -i: 表示可以归档数据至低频存储类型。
- -a: 表示可以归档数据至归档存储类型。
- -c: 表示可以归档数据至冷归档存储类型。

#### Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

Unarchive默认可以将数据恢复成标准存储,指定以下参数时:

- -i: 表示可以恢复数据至低频存储类型。
- -o: 表示可以临时解冻归档存储类型, 使数据临时可读。

### Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

#### 指定以下参数时:

- -detail: 表示可以查看文件进度信息。
- -sync: 表示该命令需要同步等待分层存储任务结束才会退出。

### ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -ls2 <path>

返回信息会包含文件的存储类型,示例如下。

drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse -rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia\_data.csv -rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia\_data.json
# **4.SmartData 3.7.x** 4.1. SmartData 3.7.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS、JindoTable和相关工具集。本文为您介绍Smart Dat a(3.7.x)版本的新增内容。

## JindoFS

此版本中JindoFS的新特性如下表所示。

特性	描述
JindoFS支持展示统计信息	SmartData 3.7.2及后续版本支持该特性。 JindoFS服务收集汇总了一些重要的指标信息,例如OSS读写吞吐、缓存读写吞吐、 缓存使用率等,可用于对接Prometheus,进行可视化监控。
JindoFS分层存储支持冷归档	SmartData 3.7.3及后续版本支持该特性。 分层存储命令支持了OSS冷归档存储类型,可以对冷数据进一步节省成本,详情请 参见 <mark>分层存储命令使用说明。</mark>

## JindoSDK

此版本中JindoSDK的新特性如下表所示。

特性	描述
支持访问AWS S3文件系统	JindoFS客户端支持了AWS S3文件系统,可以用来访问S3上的数据。
支持OSS PrefixLink特性	JindoFS客户端支持了OSS PrefixLink特性,通过提升Rename性能,能够有效加速 Hive作业,特别对于Hive ETL场景有明显优化效果。
支持OSS原子Rename	SmartData 3.7.2及后续版本支持该特性。 JindoFS客户端利用OSS新特性实现了原子Rename,适用于Delta场景。

## JindoTable

此版本中JindoTable的新特性如下表所示。

特性	描述
归档、解冻功能支持冷归档	SmartData 3.7.2及后续版本支持该特性。 JindoTable SDK模式归档和解冻命令支持了OSS冷归档存储类型,详情请参 见J <mark>indoTable SDK模式归档和解冻命令介绍</mark> 。

# 4.2. JindoFS Block模式

## 4.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

## 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别

对于一写多读的场景效果显著。

- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

#### 配置使用方式

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	:Data 🗸	● 正常		
状态	部署拓扑	配置	配置修改历史		
配置过	地				服务配置
配置搜	<b>索</b>			Q	全部   smartdata-site namespace client   storage

#### 3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	描述	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>

ⅲ. 单击**确定**。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

#### 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

client storage			
storage.handler.threads	40		0
storage watermark low ratio	0.2		0
storage.watermark.iow.ratio	0.2		
storage.watermark.high.ratio	0.4		0
storage.oss.upload.threads	20		0
	描述		
storage.watermark.high.ratio		吏用量的上水位比例,每块数据盘的 空间到达上水位即会触发清理。默认	JindoFS数据目录占 值: 0.4。
storage.watermark.low.ratio		量的下水位比例,触发清理后会自动 据目录占用空间清理到下水位。默认	清理冷数据,将 \值:0.2。
	client storage storage.handler.threads storage.watermark.low.ratio storage.watermark.high.ratio storage.oss.upload.threads	client storage storage.handler.threads 40 storage.watermark.low.ratio 0.2 storage.watermark.high.ratio 0.4 storage.oss.upload.threads 20 描述 最示磁盘位用的磁盘雪	client storage         storage.handler.threads         40         storage.watermark.low.ratio         0.2         storage.watermark.low.ratio         0.4         storage.oss.upload.threads         20         描述         最示磁盘使用量的上水位比例,每块数据盘的,用的磁盘空间到达上水位即会触发清理。默认         表示使用量的下水位比例,触发清理后会自动,JindoFS数据目录占用空间清理到下水位。默认

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

## 4.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

## 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见<mark>使用Raft-RocksDB-</mark> Tablestore作为存储后端。

#### 单机RocksDB作为元数据服务的架构图如下所示。



### 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > Smart Dat a**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	● 正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<b>索</b>		Q	全部 smartdata-site namespace client storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
    - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX

namespace.ots.endpointTablestore实例的Endpoint地址,推荐使 用VPC地址。http://emr-ifs.cn-hangzhou.vpc.tables tore.aliyuncs.comReare是否开启OTS异步上传,包括: o true o false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	参数	参数说明	示例
Ramespace.backend.rocksdb.asyn c.ots.enabled	namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使 用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
	namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。 ⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	true

#### 7. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 8. 单击右上角的操作 > 启动All Components。

## 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. 准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809
```

25 jfs://test/

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 \_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ jindo jfs -metaStatus
emr-header-1:8101
)tsUploader: _synced=1
RocksDB Row Counts of each CF(Table)]

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。
  - 新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。
- 3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false

参数 描还 示例	参数	描述	示例

	是否开启从OTS恢复元数据,包括:	
namespace.backend.rocksdb.reco very.mode	• true • false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

==== emr-header-1:8101 ==== [Recovery From OTS Status] state: FINISH total 22855 rows.

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一致
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
       1596 1482809 25 jfs://test/
# 文件可正常读取 (cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root
                                  0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r----- 1 hadoop hadoop
-rw-r----- 1 hadoop hadoop
# 口诗壯太 了可你怎么 !!
                                   5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
                                  20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021 , ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: ° true ° false	false

- 9. 重启集群。
  - i. 单击上方的**集群管理**页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

## 4.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

### 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

? 说明	需要开启事务功能。
• 创建3 Mast	er的EMR集群,详情请参见创建集群。
🛯 高可用	高可用: 🛛
	部署方式: 👩 🔷 2 Master
	② ③ 3 Master 查看服务部署
? 说明	如果没有部署方式,请 <mark>提交工单</mark> 处理。

## 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





## 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,单击**集群服务 > SmartData**。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏,单击**集群服务 > SmartData**。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤 7。

#### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com

参数	参数说明	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 7. 单击右上角的操作 > 启动All Components。

#### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. (可选)准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。



ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 \_\_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail



- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

```
jindo jfs -metaStatus -detail
```

如图所示, LEADER节点的state为FINISH表示恢复完成。



7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

#### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

## 4.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

## 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

## Smart Dat a Smart Dat a 3.7.x

## E-MapReduce

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

## 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 ismartdata-site namespace client istorage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏,单击基础设置 > 生命周期,在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> show tables;	
database tableName	isTemporary
default audit_log	se
default audit_log_source	false
default fs_image t	se

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sql>	select * from	n audit_log_sour	ce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-20	10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null per	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-20	10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null per	rm=null	2020-10-20						
2020-10-20	11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null per	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-20	11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null per	rm=null	2020-10-20						
2020-10-20	11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s dst	t=null	<pre>perm=root:root:</pre>	rwxr-xx	2020-10	-20			
2020-10-20	11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null per	rm=null	2020-10-20						
2020-10-20	11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s dst	t=null	perm=root:root:	rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	m audit_log	lim	it 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true roo	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	0:50:11.950	true roo	ot (	auth:SIMPL	LE)	192.	- 10 C	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true roo	ot (	auth:SIMPL	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	1:26:06.469	true roo	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true roc	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:11.320	true roc	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:14.368	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:14.393	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:16.230	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:16.255	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ds, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-	sql>	select	cmd	,count(*)	) from	audit_log	where	date="	2020-	10-20"	group	by	cmd	order	by	cmd;
cmd	cou	int(1)														
getFil	.eStat	usReque	est	387												
listFi	lelet	Request		387												
Time t	aken:	5.767	seco	nds. Feta	hed 2	row(s)										

## 4.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

## 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

## 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web U功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview									
Start Time:	Fri Oct 16 12:29:24 2020								
Status:	Active								
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)								
Node:	Live Nodes: [2], Decommission Nodes: [0]								
Version:	3.0.0								
Build No:	fa0ea608a42								

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)									
Namespace:	jfs://test/								
Namespaces:	test								
Mode:	BLOCK_MODE								
Backend URI:	oss://								
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27								

StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)											
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version				
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4				
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4				

Overview											
Start Time:         Fri Oct 16 12:29:25 2020											
Version:	3.0.0										
Build Version:	fa0ea608a	42a5e0e4ebcdbbfc3c041fe49f8	e82e								
Storage Lists (	Storage Lists (4)										
Directory	StorageType	Capacity	Last Eviction Time	Partition Id							
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31							
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21							
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1							
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11							

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

## 4.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

## 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限:您可以设置文件的777权限,以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



## 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	a∰o Smart	tData ~	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	<sup>t</sup> 滤			服务配置
配置搜	素		_	全部 smartdata-site namespace client storage
清 論 ノ	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager 🗈 Audit 💠 S	ettings				🔒 admin
Service Manager					
Service Manager					🛛 Import 🖉 Export
🗁 HDFS	+22	🗁 HBASE	+ 🛙 🖸	🕞 HIVE	+ 🛛 🖸
		emr-hbase	• 7 8	emr-hive	• 2 8

iii. 配置相关参数。

参数	描述				
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。				
Username	自定义。				
Password	自定义。				
Namenode URL	输入 <i>jfs://{namespace_name</i> }/。				
Authorization Enabled	使用默认值No。				
Authentication Type	使用默认值Simple。				
dfs.datanode.kerberos.principal					
dfs.namenode.kerberos.principal	<b>太</b> 指它				
dfs.secondary.namenode.kerberos.principal	小☆→。				
Add New Configurations					

iv. 单击Add。

## 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。 以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

#### 3. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

## 4.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

## 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 令使用说明。

## 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。

策略名称	策略说明
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令,获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

### 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用 于一些高压缩比的文件。支持以下两种压缩策略。

策略名称	策略说明
NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path>  $\ldots$ 

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path>: 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

## 4.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

## 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

### 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

#### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

bin jindo jfs -dumpMetadata test-block
SLF4J: CLass path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4j.org/codes.htmlamut.ple\_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4].impl.Log4jLoggerFactory]
SLF4J: Actual binding is of type [org.slf4].impl.Log4jLoggerFactory]

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

```
Sucessfully upload namespace metadata to OSS.
```

## 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如, 配置的 namespace.sysinfo.oss.uri 为 oss://abc/ ,则上传的文件会在 oss://abc/metadataDump 子目录中。

参数	说明
namespace.sysinf o.oss.uri	存储Bucket和路径。
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinfo.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

	//emrtest/sysinfc /metadataDu	ump test-block/
▲上传 + 创建目录 □	更多▼	
名称		类型 / 大小
2020_09_14_18_58	16 3	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

#### 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

### E-MapReduce

{		
	"type":"string",	/*INode <b>类型,</b> FILE <b>文件</b> DIRECTORY <b>目录</b> */
	"id": "string",	/*INode id*/
	"parentId" :"string",	/* <b>父节点</b> id*/
	"name":"string",	/*INode <b>名称</b> */
	"size": "int",	/*INode <b>大小,</b> bigint*/
	"permission":"int",	/*permission <b>以</b> int <b>格式存放</b> */
	"owner":"string",	/*owner <b>名称</b> */
	"ownerGroup":"string",	/*owner <b>组名称</b> */
	"mtime":"int",	/*inode <b>修改时间,</b> bigint*/
	"atime":"int",	/*inode <b>最近访问时间,</b> bigint*/
	"attributes":"string",	/* <b>文件相关属性</b> */
	"state":"string",	/*INode <b>状态</b> */
	"storagePolicy":"string	", /*存储策略*/
	"etag":"string"	/*etag*/
}		

## 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 audit\_log和fs\_image。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta data 生成的数据。

```
示例如下。
```

```
jindo-sql> show partitions fs_image;
partition
namespace=k /datetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)
```

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@em Spark ma jindo-sa database default default default	nr-worke ster: y ql> show audit_l audit_l	r-2 hada arn, App tables; tableNa og og_sourc	op]# jindo sql lication Id: a me isTem false se false false	pp porary	-										
Time tak	cen: 0.3	45 secon	ds, Fetched 3	row(s)											
jindo-sa	al> sele	ct * fro	m fs_image lim	it 10;											
atime	attr	etag	id mtime	name	owner	ownerGr	•oup	parentId	permission	size	state	storage	Policy	type	name
space	datetim														
0			7311076005051	899448	1603084	070081	/tpcds/	/orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450819	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			1653444804190	6675495	1603084	071350	/tpcds/	/orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450820	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			7311076005051	899470	1603084	070185	/tpcds/	/orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450821	root	root	334790	833296
5855433	489	ø	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			1192276202347	9287249	1603084	069581	/tpcds/	/orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450822	root	root	334790	833296
5855433	489	ø	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36					100000	
0			1076984051887	2441036	1603084	073592	/tpcds/	/orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450823	root	root	334790	833296
5855433	489	Ø	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			2699389986624	511354	1603084	068996	/tpcds/	/orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450824	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			11922/620234/	9287307	1603084	069875	/tpcds/	orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450825	root	root	334790	\$33296
5855433	489	0	Finalized	MAKM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			1546468482017	665002	1603084	072440	/tpcds/	orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450826	root	root	334790	833296
5855433	489	ø	Finalized	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			1653444804190	6675460	1603084	071170	/tpcds/	orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450827	root	root	334790	833296
5855433	469	0	Finalized	RAKM	Directo	ry operation	kugou	2020_10_20_	10_20_30					224200	000000
0	100		7511076005051	899544	1003084	070572	/tpcds/	orc/5000/web	_returns/wr_retu	urned_date_s	sk=2450828	root	root	334790	833296
5855433	489	0	Findlized	MARM	Directo	ry	kugou	2020_10_20_	10_50_36						

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

jindo-sql> count(1)	<pre>select count(*)</pre>	) from fs_image where	type = "Directory"	and namespace="kugou"	and datetime="2020_10_20_10_47_14";
11837 Time taken:	6.852 seconds,	Fetched 1 row(s)			

## 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table\_name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

示例如下。

hive> select * from inode_metadata_test8 limit 100;						
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futu			ing a different execu	tion engine (i.e. spar	k, tez) or using I	Hive 1.X releases
Query ID = root_202089						
Total jobs = 1						
Launching Job 1 out of 1						
Number of reduce tasks is set to 0 since there's no reduce operator						
Starting Job = job_1 , Tracking URL = http://emr-heade	:20888	/proxy/appl				
Kill Command = /usr/lib/hadoop-current/bin/hadoop job -kill job_1599						
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0						
[2020-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0%						
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.22 se						
MapReduce Total cumulative CPU time: 1 seconds 220 msec						
Ended Job = job_:						
MapReduce Jobs Launched:						
Stage-Stage-1: Map: 1 Cumulative CPU: 1.22 sec HDFS Read: 6867 HDFS Write: 1						
Total MapReduce CPU Time Spent: 1 seconds 220 msec						
0K						
Directory 11274334386847219714 11274334386847219713 /uttest/oss			staff 1599545017615	1599545017615		WARM
Directory 11274334386847219719 11274334386847219713 /uttest/oss2			staff 1599545017654	1599545017654		WARM
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir			staff 1599545017636	1599545017636		WARM
File 11274334386847219715 11274334386847219714 /uttest/oss/file1		caojie	staff 1599545017632	1599545017632		WARM
File 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2			staff 1599545017642	1599545017642		WARM
File 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3			staff 1599545017651	1599545017651		WARM
Directory 11274334386847219728 11274334386847219719 /uttest/oss2/dir			caojie staff 15995	45017654 15995450176	554 Fina	lized WARM
File 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2			staff 1599545017658	1599545017658		WARM
File 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3			staff 1599545017666	1599545017666		WARM
Directory 11274334386847219713 17672823557433851905 /uttest 0	511 caoji		1599545017615 15995	45017615 Fir	nalized WARM	
Time taken: 10.734 seconds, Fetched: 10 row(s)						

## 4.2.9. JindoFS Credential Provider使用说明

JindoFS的数据存储在OSS中,如果您需要访问JindoFS的数据,需要提供OSS的AccessKey才能访问。Smartdata 3.4.0及后续版 本支持JindoFS Credential Provider,您可以通过配置JindoFS Credential Provider,将加密后的AccessKey信息添加至文件中,以 避免泄露AccessKey信息。

## 配置JindoFS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smartdata-site服务配置。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击smartdata-site页签。
- 3. 添加配置信息。
  - i. 在smartdata-site页签,单击右上角的自定义配置。
  - ii. 在新增配置项对话框中,新增如下配置。

参数	描述
fs.jfs.credentials.provider	配置com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential。例如, com.aliyun. emr.fs.auth.TemporaryAliyunCredentialsProvider,c om.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,c vider,com.aliyun.emr.fs.auth.EnvironmentVariable CredentialsProvider 。 Provider详情请参见Provider类型。

#### ⅲ. 单击确定。

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## Provider类型

• TemporaryAliyunCredentialsProvider

适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsPr ovider
fs.jfs.accessKeyId	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。
fs.jfs.securityToken	OSS的SecurityToken(临时安全令牌)。

• SimpleAliyunCredentialsProvider

适合使用长期有效的AccessKey访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvid er
fs.jfs.accessKeyld	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。

#### • Environment VariableCredent ialsProvider

### 该方式需要在环境变量中配置以下参数。

参数	参数说明	
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EnvironmentVariableCredential sProvider	
ALIYUN_ACCESS_KEY_ID	OSS的AccessKey ID。	
ALIYUN_ACCESS_KEY_SECRET	OSS的AccessKey Secret。	
	OSS的SecurityToken(临时安全令牌)。	
ALIYUN_SECURITY_T OKEN	⑦ 说明 仅配置有时效Token时需要。	

#### • JindoCommonCredentialsProvider

该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.JindoCommonCredentialsProvi der
jindo.common.accessKeyld	OSS的AccessKey ID。
jindo.common.accessKeySecret	OSS的AccessKey Secret。
jindo.common.securityToken	OSS的SecurityToken(临时安全令牌)。

#### • EcsStsCredentialsProvider

该方式无需配置AccessKey,可以免密方式访问OSS。

参数	参数说明
fs.jfs.credentials.provider	${\tt com.aliyun.emr.fs.auth.EcsStsCredentialsProvider}$

## 4.2.10. JindoFS Block模式加密使用说明

JindoFS Block模式支持文件加密,加密机制和使用方法与Apache HDFS的Encryption Zone类似。加密密钥通过密钥管理服务 (KMS)统一管理,您可以对有敏感数据的目录设置加密策略,然后就可以透明地在该目录下加密写入的数据和解密读取的数据,无需更改您的代码。

## 前提条件

- 已创建集群,详情请参见创建集群。
- 已开通密钥管理服务(KMS),详情请参见开通密钥管理服务。

## 背景信息

Block模式加密架构图如下:



## 配置JindoFS使用阿里云KMS

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击namespace页签。
- 3. 添加配置信息。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下配置。

参数	描述
crypto.provider.type	Provider的类型,仅支持ALIYUN。
crypto.provider.endpoint	KMS的公网接入地址。详情请参见 <mark>调用方式</mark> 。
crypto.provider.kms.accessKeyld	访问KMS的AccessKey ID。
crypto.provider.kms.accessKeySecret	访问KMS的AccessKey Secret。

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 在右上角选择操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅳ. 在确认对话框中, 单击确定。

## 使用JindoFS KeyProvider

Jindo KeyProvider负责对接KMS,加密密钥存储在KMS。KeyProvider基于KMS提供新增密钥、查询密钥和轮换密钥等功能。

• 新增密钥: 传入keyldName, 创建一个新的密钥。

jindo key -create -keyIdName <keyIdName>

⑦ 说明 本文示例中的<keyIdName>为您创建的密钥名称。

例如,执行以下命令新增policy\_test的密钥。

jindo key -create -keyIdName policy\_test

在阿里云KMS控制台,您可以看到新增了一个名为policy\_test的密钥。

密钥管理服务控制台	创建凭据
用户主密钥	凭据管家最新上线全托管RDS凭据,支持定时更新RDS数据库口令,减少数据库口令泄漏风险。
凭据	名称
证书	policy_test 普通凭据

• 查询密钥: 查看当前存在的密钥名。

jindo key -list

#### 返回信息如下:

Listing Keys: policy\_test policy\_test2

• 轮换密钥:您可以根据Key ID定期更换密钥。更新密钥后Key Version会随之发生变化,即文件在加密时,使用最新的密钥进行

#### 加密, 文件在解密时使用现有文件的密钥版本进行解密。

jindo key -roll -keyIdName <keyIdName>

#### 例如,执行以下命令轮换密钥policy\_test。

jindo key -roll -keyIdName policy\_test

## 在阿里云KMS控制台,您可以看到密钥*policy\_test*的版本状态已经更新,之前的版本状态变成了ACSPrevious,新的版本状态为ACSCurrent。

密钥管理服务控制台	← policy_test			
用户主密钥	任招兴法律			
凭据	汽炉中间			
证书	凭据信息	名称	policy_test	
		ARN	acs:kms:cn-hangzh	policy_test
		加密密钥	系统托管密钥	
		描述信息	Encryption Key for JindoFS File E	1
	∨ 版本列表 ⑦			
	存入凭据值			
	版本号			版本状态
	1613802530			ACSCurrent   状态管理
	1613801564			ACSPrevious   状态管理

## 管理JindoFS加密策略

您可以根据以下命令,设置和查看加密策略:

• 设置加密策略

jindo jfs -setCryptoPolicy -keyIdName <keyIdName> <path>

⑦ 说明 本示例的 <path> 为您访问JindoFS上文件的路径。例如jfs://test/。

● 查看加密策略

jindo jfs -getCryptoPolicy <path>

#### 设置和查看加密策略示例如下所示:

#### 1. 查看*jfs://test/*路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回信息显示为 {NONE} 。

2. 设置*jfs://test/*的加密策略。

jindo jfs -setCryptoPolicy -keyIdName policy\_test jfs://test/

#### 3. 进入bigboot目录,再次查看jfs://test/路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回如下信息。

SLF4J: Class path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/b2jindosdk/3.4.0-hadoop3.1/package/b2jindosdk-3.4 .0-hadoop3.1/lib/jindo-distcp-3.4.0.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/hadoop/3.2.1-1.0.1/package/hadoop-3.2.1-1.0.1/sha re/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html#multiple bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] 21/03/12 13:52:34 WARN: Unable to load native-hadoop library for your platform... using builtin-java clas ses where applicable 21/03/12 13:52:35 INFO: Jboot log name is /var/log/bigboot/jboot-20210312-135234-12953.LOG 21/03/12 13:52:35 INFO: Write buffer size 1048576, logic block size 134217728 21/03/12 13:52:35 INFO: cmd=getFileStatus, src=jfs://test/, dst=null, size=0, parameter=null, time-in-ms= 7. version=3.4.0 21/03/12 13:52:35 INFO: cmd=getCryptoPolicy, src=jfs://test/, dst=null, size=0, parameter=, time-in-ms=2, version=3.4.0 The crypto policy of path: jfs://test/ is {cipherSuite: AES\_CTR\_NOPADDING\_256, keyIdName: policy test2, k eyIdVersion: null, edek: , iv: } 21/03/12 13:52:35 INFO: Read total statistics: oss read average <none>, cache read average <none>, read o ss percent <none>

#### 设置完成后即可正常读写该路径下的文件。

◦ 拷贝本地文件至HDFS。

hadoop fs -put test.log jfs://test/

• 展示文件内容。

hadoop fs -cat jfs://test/test.log

# 4.3. JindoFS Cache模式

## 4.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓 存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主 要介绍JindoFS的缓存模式及其使用方式。

#### 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

## 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

#### 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

#### 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的集群管理页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 述			服务配置
配置搜	素 \		0	全部 smartdata-site namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

## 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。

此配置无需重启SmartData服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后, Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

## 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

服务配置				
全部   smartdata-site   namespace   client storage				
	storage.handler.threads	40	0	
	storage.watermark.low.ratio	0.2	0	
	storage.watermark.high.ratio	0.4	0	
	storage.oss.upload.threads	20	0	
参数		描述		
storage.watermark.high.ratio		表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。		
storage.watermark.low.ratio		表示使用量的下水位比例,触 JindoFS数据目录占用空间清	触发清理后会自动清理冷数据,将 理到下水位。默认值:0.2。	

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

#### 2. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

#### 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。

参数	参数说明
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击namespace页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>	
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。	

## 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在**服务配置**区域的client页签,配置以下参数。

参数	参数说明	
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。	
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。	

● 在服务配置区域的smart dat a-sit e页签,配置以下参数。

参数说明	
文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。	
启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。	
② 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。	

## 4.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce (简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

## 前提条件

适用环境:ECS(EMR环境外)+Hadoop+JavaSDK。

### 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smartdata-aliyun-jfs-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

#### 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击**完成**。

#### 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击确定。

## 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏, 单击实例与镜像 > 实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

## 步骤四:在ECS上设置环境变量

#### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

## 步骤五:测试免密方式访问的方法

#### 1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

### JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
        while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
    }
}
```

## 4.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

## 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

## 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 ismartdata-site namespace client istorage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: I true:打开AuditLog功能。 I false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏,单击基础设置 > 生命周期,在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。
• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

isTemporary
false

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sql>	<ul> <li>select * from</li> </ul>	n audit_log_sour	ce limit 1	.0;				
datetime	allowed	ugi ip	ns c	md src	dst perm	date		
2020-10-20	10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null pe	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	0				
2020-10-20	10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null pe	rm=null	2020-10-20						
2020-10-20	11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null pe	rm=hadoop:hado	pop:rwxrwxr-x	2020-10-2	.0				
2020-10-20	11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null pe	rm=null	2020-10-20						
2020-10-20	11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s ds	t=null	perm=root:root:	rwxr-xx	2020-10-	-20			
2020-10-20	11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null pe	erm=null	2020-10-20						
2020-10-20	11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s ds	t=null	perm=root:root:	rwxr-xx	2020-10-	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> se	lect * from	n audit_log	li	mit 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 10	:50:11.924	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	wxrwx
r-x 2020-	10-20													
2020-10-20 10	:50:11.950	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	:26:06.445	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	wxrwx
r-x 2020-	10-20													
2020-10-20 11	:26:06.469	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	:26:11.295	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20													
2020-10-20 11	:26:11.320	true ro	ot	(auth:SIMPL	LE)	192.	1818 - I	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 11	:26:14.368	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20													
2020-10-20 11	:26:14.393	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 11	:26:16.230	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20													
2020-10-20 11	:26:16.255	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken: 1	.918 second	ds, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-sql> select cmd	,count(*)	from audit_	og where	date="2020	-10-20"	group	by cn	nd order	by	cmd;
cmd count(1)										
getFileStatusRequest	387									
listFileletRequest	387									
Time taken: 5.767 seco	nds. Fetch	row(s)								

# 4.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

#### 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

#### ↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后,MapReduce Application Master或Spark Driver执行最终的Job Commit操作时,会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

#### 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.jindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

置mapreduce.outputcommitter.factory.scheme.oss为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > SmartData。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。

- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

#### 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

#### 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

#### 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。

```
ii. 单击配置页签。
```

- iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

```
默认值为8。
```

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# 4.3.5. JindoFS OSS Credential Provider使用说明

Smart dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

# 配置JindoFS OSS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入smartdata-site页面。
  - i. 单击**配置**页签。
  - ii. 在**服务配置**区域,单击smartdata-site页签。
- 3. 在smartdata-site页签,根据配置方式修改或新增配置信息。

配置方式	描述
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数 <b>fs.jfs.cache.oss.credentials.provider</b> ,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。

配置方式	描述					
	<ul> <li>新增配置信息:</li> <li>i. 在smartdata-site页签,单击右上角的自定义配置。</li> <li>ii. 在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。</li> <li>例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> </ul>					
按照Bucket配置	<ul> <li>⑦ 说明 XXX为OSS的Bucket名称。</li> <li>iii. 单击确定。</li> </ul>					

#### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。

## E-MapReduce

类型	描述
Environment Variable Credentials Provi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 按照Bucket配置

## 您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken (临时安全令 牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment VariableCredent ialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>

类型	描述
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 4.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

#### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

#### 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息(Overview)

```
包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。
```

Overview					
Start Time:	Fri Oct 16 12:29:24 2020				
Status:	Active				
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)				
Node:	Live Nodes: [2], Decommission Nodes: [0]				
Version:	3.0.0				
Build No:	fa0ea608a4;				

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)			
Namespace:	jfs://test/		
Namespaces:	test		
Mode:	BLOCK_MODE		
Backend URI:	oss://		
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27		

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

# Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 4.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

#### 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



#### 启用JindoFS Unix权限

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	素			全部   smartdata-site <b>namespace</b> client   storage
请输入	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

#### 5. 重启配置。

- i. 单击右上角的操作 > 重启 Jindo Namespace Service。
- ii. 输入执行原因, 单击**确定**。
- 开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

#### 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger UI添加HDFS service。

Iger  QAccess Manager	🗅 Audit	Settings				🔒 admin
rvice Manager						
ice Manager						🛛 Import
-			-		-	
B HDFS		+22	B HBASE	+ 22	▷ HIVE	+ 🖬 🖻
			emr-hbase	• 7	emr-hive	• 7 8
	vice Manager ice Manager	Iger ØAccess Manager D Audit vice Manager ice Manager	Iger     Ø.Access Manager       Vice Manager       ice Manager       HDFS	Iger OAccess Manager DAudit O Settings vice Manager HDFS ← HBASE emr-hbase	Iger VAccess Manager Audit ● Settings vice Manager  HDFS  HBASE  emr-hbase	Iger VAccess Manager D Audit ● Settings vice Manager  HDFS  HBASE +22  HIVE em/hbase

iii. 配置相关参数。

参数	描述
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。
Username	自定义。
Password	自定义。
Namenode URL	输入jfs://{namespace_name}/。
Authorization Enabled	使用默认值No。
Authentication Type	使用默认值Simple。
dfs.datanode.kerberos.principal	
dfs.namenode.kerberos.principal	不肯它
dfs.secondary.namenode.kerberos.principal	د بهرم و د ا
Add New Configurations	

iv. 单击Add。

### 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

# 4.4. JindoTable

# 4.4.1. 开启native查询加速

JindoTable通过Native Engine,支持对Spark、Hive或Presto上ORC或Parquet格式文件进行加速。本文为您介绍如何开启native 查询加速,以提升Spark、Hive和Presto的性能。

## 前提条件

已创建集群,且ORC或Parquet文件已存放至JindoFS或OSS,创建集群详情,请参见创建集群。

#### 使用限制

- 不支持对Binary类型文件进行加速。
- 不支持分区列的值存储在文件中的分区表。
- 不支持EMR-5.X系列及后续版本的E-MapReduce集群。
- 不支持代码spark.read.schema (userDefinedSchema)。
- 支持Date类型区间为1400-01-01到9999-12-31。
- 同一个表中查询列不支持区分大小写。例如, NAME和name两个列在同一个表中无法使用查询加速。
- Spark、Hive和Presto服务支持的引擎和存储格式如下所示。

引擎	ORC	Parquet
Spark2	支持	支持
Spark3	支持	支持
Presto	支持	支持
Hive2	不支持	支持
Hive3	不支持	支持

• Spark、Hive和Presto服务支持的引擎和存储文件系统如下所示。

引擎	OSS	JFS	HDFS
Spark2	支持	支持	支持
Presto	支持	支持	支持
Hive2	支持	支持	不支持
Hive3	支持	支持	不支持

# 提升Spark性能

1. 开启JindoTable ORC或Parquet加速。

? 说明

- 因为查询加速使用的是堆外内存,所以在Spark任务中建议添加配置 --conf spark.executor.memoryOverhead=
   4g ,提高Spark申请额外资源用来进行加速。
- Spark读取ORC或Parquet时,需要使用DataFrame API或者Spark-SQL。

- 。 全局设置
  - a. 进入详情页面。
    - a. 登录阿里云E-MapReduce控制台。
    - b. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
    - c. 单击上方的集群管理页签。
    - d. 在集群管理页面,单击相应集群所在行的详情。
  - b. 修改配置。
    - a. 在左侧导航栏,选择**集群服务 > Spark**。
    - b. 在Spark服务页面,单击配置页签。
    - c. 在搜索区域,搜索参数spark.sql.extensions,修改参数值 为io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtension。
  - c. 保存配置。
    - a. 单击保存。
    - b. 在确认修改对话框中, 输入执行原因, 单击确定。
  - d. 重启ThriftServer。
    - a. 在右上角选择操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框中,输入执行原因,单击确定。
    - c. 在确认对话框中, 单击确定。
- Job级别设置

使用spark-shell或者spark-sql时,可以添加Spark的启动参数。

作业详情请参见Spark Shell作业配置或Spark SQL作业配置。

- 2. 检查开启情况。
  - i. 登录Spark History Server UI页面。 登录详情请参见访问链接与端口。



# 提升Presto性能

↓ 注意 Presto查询并发较高,且查询加速使用堆外内存,因此使用查询加速时内存配置必须大于10 GB。

因为Presto已经内置JindoTable native加速的 catalog: hive-acc ,所以您可以直接使用 catalog: hive-acc 来启用查询加 速。

示例如下。

presto --server emr-header-1:9090 --catalog hive-acc --schema default

⑦ 说明 目前使用Presto查询加速功能时,暂不支持读取复杂的数据类型,例如Map、Struct或Array。

## 提升Hive性能

↓ 注意 如果您对作业稳定性要求较高时,建议不要开启native查询加速。

您可以通过以下两种方式提升Hive性能:

• 控制台方式

在控制台Hive服务的配置页面,搜索并修改自定义参数hive.jindotable.native.enabled为true,保存配置后,重启服务使 配置生效,此方式适用于Hive on MR和Hive on Tez。

<返回 🛛 🍇 Hive ➤ ●正常		① 查看操作历史
状态 部署拓扑 配置 配置修改历史 元数据		
配置过滤	服务配置	
配置搜索 hive.jindotable.native.enabled	全部   hive-site	
配置范围	hive.jindotable.native.enabled true	\$ Ø

• 命令行方式

您可以直接在命令行中设置 hive.jindotable.native.enabled 为 true 来启用查询加速。因为EMR-3.35.0及后续版本已 经内置JindoTable Parquet加速的插件,所以您可以直接设置该参数。

set hive.jindotable.native.enabled=true;

⑦ 说明 目前使用Hive查询加速功能时,暂不支持读取复杂的数据类型,例如Map、Struct或Array。

# 4.4.2. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

#### 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partitionCol2=2,... 的格式。

#### -accessStat

语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

```
jindo table -accessStat -d 7 -n 20
```

#### -cache

● 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

```
表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。
```

• 示例: 缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

#### -uncache

● 语法

jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示删除集群本地磁盘上指定表或分区的缓存数据。

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partition Col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

- -archive
- 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

- -unarchive
- 语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName> [-p] <partitionSpec>

- 功能
  - 表示将归档数据转为标准存储。

-o 将归档数据转为解冻, -i 将归档数据转为低频。

示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

```
jindo table -unarchive -i -t db1.t2
```

- -status
- 语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

。 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

#### -optimize

● 语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

#### -showTable

- 语法
  - jindo table -showTable {-t} <dbName.tableName>
- 功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例:展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

```
-showPartition
```

● 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

jindo table -showPartition -t db1.t1 -p date=2020-10-12

- -listTables
- 语法

jindo table -listTables [-db] [dbName]

功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

• 示例:

#### 。 展示default 库中的表。

jindo table -listTables

○ 列出数据库db1中的表。

```
jindo table -listTables -db db1
```

### -dumpmc

● 语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} table} <tablename> {-p} <partitionSpec> {-f} <csv{tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络Tunnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-p	分区信息 。例如 pt=xxx , 多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

功能

- 表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。
- 示例:
  - Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord
```

• Dumpmc Maxcompute表(CSV格式)至OSS存储。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv

# 4.4.3. JindoTable SDK模式归档和解冻命令介绍

JindoTable SDK模式提供archiveTable和unarchiveTable命令,可以在不依赖Jindo Namespace Service的情况下进行归档和解冻 等操作。本文为您介绍archiveTable和unarchiveTable命令的使用方法。

#### 前提条件

- 已创建集群,详情请参见创建集群。
- 待归档的数据必须是表数据(可以是分区表或非分区表),且已经位于阿里云对象存储OSS。

### 背景信息

JindoTable原有archive和unarchive命令可以对OSS上的表或分区进行归档或解冻等操作,但archive和unarchive命令依赖 Smart Dat a组件Jindo Namespace Service。现在新增的archiveTable和unarchiveTable命令,可以在不依赖Jindo Namespace Service的情况下进行归档和解冻等操作。

新增的archiveTable和unarchiveTable命令与原有archive和unarchive命令的主要区别为:

- 可以在未部署Smart Data服务的集群上执行。例如,非EMR的用户自建集群。
- 可以通过传入过滤参数,一次应用于大量分区,多线程执行。如果本地多线程仍不能满足需求,还可以启动MapReduce任务 在整个集群上执行。

原有archive和unarchive命令的详细信息,请参见JindoTable使用说明。

## 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持新增的archiveTable和unarchiveTable命令。

## archiveTable命令

archiveTable命令可以对OSS上的表或分区进行归档。

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,获取帮助信息。

jindo table -help archiveTable

#### archiveTable命令语句格式如下所示。

```
-archiveTable -t <dbName.tableName> \
-i/-a/-ca \
[-c "<condition>" | -fullTable] \
[-b/-before <before days>] \
[-p/-parallel <parallelism>] \
[-mr/-mapReduce] \
[-e/-explain] \
[-w/-workingDir <working directory>] \
[-1/-logDir <log directory>]
```

参数	描述	是否必选参数
-t <dbname.tablename &gt;</dbname.tablename 	待归档的表名称,格式为 数据库名.表名 。 数据库和表名之间以半角句号(.)分隔。表可以是分区表或非分区表。	是
-i/-a/-ca	目标存储方式。支持如下方式: • -i: 低频 (Infrequent Access, IA)存储。 • -a: 归档 (Archive)存储。 • -ca: 冷归档 (Cold Archive)存储。 如果指定-i则表示低频存储,会跳过已经处于归档或冷归档存储的文件。如果指定-a则 表示归档存储,会跳过冷归档的文件。	是
-c " <condition>"   - fullTable</condition>	<ul> <li>-fullTable 和 -c "<condition>" 只需提供一个,即要么指定 -c "<condition>" ,要么指定 -fullTable 。</condition></condition></li> <li>* 指定 -fullTable 时,则为归档整表,既可以是非分区表也可以是分区表。</li> <li>* 指定 -c "<condition>" 时,则提供了一个过滤条件,用来选择希望归档的分区,支持常见运算符,例如大于号(&gt;)。</condition></li> <li>例如,数据类型为String的分区ds,希望分区名大于 'd',则代码为 -c " ds &gt; ' d' " 。</li> </ul>	是

参数	描述	是否必选参数
-b/before <before days&gt;</before 	只有创建时间距离现在超过一定天数的表或分区才会被归档。	否
-p/-parallel <parallelism></parallelism>	归档操作的并行度。	否
-mr/-mapReduce	使用Hadoop MapReduce而非本地多线程来归档数据。	否
-e/-explain	如果出现该选项,则为解释(explain)模式,只会显示待移动的分区列表,而不会真 正移动数据。	否
-w/-workingDir	只在MapReduce作业时使用,为MapReduce作业的工作目录。必须有读写权限,工作 目录可以非空(作业执行过程中会创建临时文件,执行完毕会清理临时文件)。	否
-l/-logDir <log directory&gt;</log 	指定Log文件的目录。	否

#### unarchiveTable命令

unarchiveTable命令与archiveTable命令格式基本一致,但效果相反。unarchiveTable命令可以对OSS上的表或分区进行解冻。

- 1. 通过SSH方式登录集群,详情请参见<mark>登录集群</mark>。
- 2. 执行以下命令,获取帮助信息。

jindo table -help unarchiveTable

unarchiveTable命令语句格式如下所示。

```
-archiveTable -t <dbName.tableName> \
-i/-a/-o/-cr \
[-notWait] \
[-c "<condition>" | -fullTable] \
[-b/-before <before days>] \
[-p/-parallel <parallelism>] \
[-mr/-mapReduce] \
[-e/-explain] \
[-w/-workingDir <working directory>] \
[-1/-logDir <log directory>]
```

unarchiveTable命令与archiveTable命令参数只有以下两处区别:

- 没有必选参数 -i/-a/-ca, 而被可选参数-i/-a/-o/-cr替代。
- 多了可选参数-notWait。

参数	描述	是否必选参数
-i/-a/-o/-cr	转换存储类型,均适用于冷归档。 • 如果不指定-i/-a/-o/-cr参数,则转换存储格式为标准(Standard)存储。	
	⑦ 说明 只有当冷归档文档处于完全解冻状态时,才可以转换到标准 (Standard)、低频 (Infrequent Access,IA)或归档(Archive)存储类型。	
	<ul> <li>如果指定-i/-a/-o/-cr参数,相关参数描述如下:</li> <li>指定-i参数,则转换存储格式为低频 (Infrequent Access, IA)存储,跳过原本为标准存储的文件。对冷归档 (Cold Archive)有效。</li> <li>指定-a参数,则转换存储格式为归档,唯一作用是将冷归档 (Cold Archive)文件改为归档,跳过原本为标准或低频存储的文件。</li> <li>指定-o参数,则仅做解冻 (Restore)操作。跳过原本为标准存储或低频存储的文件。已经处于解冻状态的文件也会被跳过,即不会重复解冻。</li> </ul>	否
	◦ 指定-cr参数,则用来检查分区下文件的解冻任务是否完成。	

参数	描述	是否必选参数
-notWait	只对解冻(Restore)操作有效,如果指定该参数,则只发送解冻命令,而不等待解冻任务 完成。通常用于冷归档 (Cold Archive) 文件的解冻。	否

# 4.4.4. JindoTable MoveTo命令介绍

MoveTo命令可以实现表和分区数据的迁移功能。本文为您介绍MoveTo命令的使用方法。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

#### 背景信息

MoveTo命令可以在拷贝底层数据结束后,自动更新元数据,使表和分区的数据完整地迁移到新路径;可以通过条件筛选,一次 拷贝大量分区。在数据迁移过程中,还使用了多种措施保护数据的完整性,确保数据安全。

#### 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持使用MoveTo命令。

#### 使用MoveTo命令

○ 注意 集群上每次仅允许运行一个MoveTo进程。如果集群上有正在运行的MoveTo进程,启动新的MoveTo进程时会因为获取不到配置锁而退出,并告知正在运行的MoveTo进程。此时,您可以终止掉正在运行的MoveTo进程,启动新的MoveTo进程,或者等待正在运行的MoveTo进程结束。

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令, 获取帮助信息。

jindo table -help moveTo

帮助信息类似如下所示。

<dbname.tablename></dbname.tablename>	The table to move.
<destination path=""></destination>	The destination base directory which is always at the
	same level of a 'table location', where the moved
	partitions or un-partitioned data would located in.
<condition>/-fullTak</condition>	le A filter condition to determine which partitions should
	be moved, supporting common operators (like '>') and
	built-in UDFs (like to_date) (UDFs not supported
	yet), while -fullTable means that all partitions (or
	a whole un-partitioned table) should be moved. One but
	only one option must be specified among -c
	" <condition>" and -fullTable.</condition>
<before days=""></before>	Optional, saying that table/partitions should be moved
	only when they are created (not updated or modified)
	more than some days before from now.
<parallelism></parallelism>	The maximum concurrency when copying partitions, 1 by
	default.
<oss storage<="" td=""><td>e policy&gt;: Storage policy for OSS destination, which can be Standard</td></oss>	e policy>: Storage policy for OSS destination, which can be Standard
(by default), IA,	Archive, or ColdArchive. Not applicable for destinations other
than OSS. NOTE: if	you are willing to use ColdArchive storage policy, please
make sure that Col	d Archive has been enabled for your OSS bucket.
-o/-overWrite Ov	rerwriting the final paths where the data would be moved.
	For partitioned tables this overwrites partitions' locations
	which are subdirectories of <destination path="">; for</destination>
	un-partitioned table this overwrites the <destination path=""></destination>
	itself.
-r/-removesource Le	the source data be removed when the corresponding
	Otherwise (by default) the service data would be left as it
	otherwise (by default), the source data would be felt as it
-ckinTrach Ar	was.
-Skipilasii kp	present source data would be immediately deleted from the
	file system humassing the trach
-e/-explain If	The system, sypassing the trash.
0, onpiain 11	prints the table/partitions that would be moved for given
	conditions.
<log directory=""> A</log>	directory to locate log files, '/tmp/ <current user="">/' by</current>
	default

#### MoveTo命令语句如下所示。

```
jindo table -moveTo \
  -t <dbName.tableName> \
  -d <destination path> \
  [-c "<condition>" | -fullTable] \
  [-b/-before <before days>] \
  [-p/-parallel <parallelism>] \
  [-s/-storagePolicy <OSS storage policy>] \
  [-o/-overWrite] \
  [-r/-removeSource] \
  [-skipTrash] \
  [-e/-explain] \
  [-1/-logDir <log directory>]
```

参数	描述	是否必选参数
-t <dbname.tablename></dbname.tablename>	待移动的表名称,格式为数据库名.表名。 数据库和表名之前以半角句号(.)分隔。表可以是分区表或非分 区表。	是

参数	描述	是否必选参数
-d <destination path=""></destination>	待移动的目标位置。无论是移动分区还是移动非分区表的整表, 该位置都对应 "表" 一级的位置。如果移动的是分区,则分区的完 整路径是该路径+分区名。例如 <destination path="">/p1=v1/ p2=v2/ 。</destination>	是
-c " <condition>"   -fullTable</condition>	<ul> <li>两者必须且只能提供一个,即要么指定 -c "</li> <li><condition>",要么指定 -fullTable 。</condition></li> <li>指定 -fullTable 时,则为移动整表,既可以是非分区表也可以是分区表。</li> <li>指定 -c "<condition>"时,则提供了一个过滤条件,用来选择希望移动的分区,支持常见运算符,例如大于号(&gt;)。</condition></li> <li>例如,数据类型为String的分区ds,希望分区名大于'd',则代码为 -c " ds &gt; 'd' "。</li> </ul>	否
-b/before <before days=""></before>	仅创建时间距离现在超过一定天数的表或者分区才会被移动。	否
-p/-parallel <parallelism></parallelism>	迁移操作的并行度。	否
-s/-storagePolicy <oss storage<br="">policy&gt;</oss>	<ul> <li>拷贝到OSS时,在OSS上的存储策略。存储策略如下:</li> <li>Standard:归档存储。</li> <li>IA:低频 (Infrequent Access)存储。</li> <li>Archive:标准存储。</li> <li>ColdArchive:冷归档存储。</li> <li>⑦ 说明 使用前请确保OSS Bucket开通了该功能。</li> </ul>	否
-o/-overWrite	是否强制覆盖目标写入路径。如果是分区表,则只会清空待移动 分区的分区路径,不会清空整个表路径。	否
-r/-removeSource	移动完成,元数据也同步更新后,是否清理源路径。如果是分区 表,则只会清理成功移动的分区的源路径。	否
-skipTrash	清理源路径时是否跳过Trash。 ⑦ 说明 在指定了参数-r/-removeSource时适用。	否
-e/-explain	如果出现该选项,则为解释(explain )模式,只会显示待移动的 分区列表,而不会真正移动数据。	否
-l/-logDir <log directory=""></log>	指定Log文件目录。	否

## 配置锁目录

MoveTo工具实现了进程锁,需要提供一个HDFS的路径放置锁文件。默认情况下,该路径为hdfs:///tmp/jindotable-lock/。

↓ 注意 放置锁文件的路径只能是HDFS路径。如果您对该路径无操作权限时,可以按照如下步骤添加自定义配置,配置 该路径。

1. 进入HDFS服务页面。

i.

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的**集群管理**页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。

v. 在左侧导航栏,选择**集群服务 > HDFS**。

2. 修改配置。

i. 在HDFS服务的配置页面,单击hdfs-site或core-site页签。

ii. 单击右上角的自定义配置。

< 返回 <sup>②</sup> HDFS → ● 正常		◎ 查看操作历史
状态 部署拓扑 配置 配置修改历史		
配置过渡 配置接索 端約入 Q	服务配置 全部 <mark>hdfs-site</mark>   kms-site   httpfs-site   hadoop-env   httpfs-env   ssl-client core-site	部署寄户挑配置         保存           自定义配置
配置范围	dfs.datanode.cache.revocation.timeout.ms 900000	Ø

iii. 在**新增配置项**对话框中,添加配置项jindotable.moveto.tablelock.base.dir,参数值为一个已存在的HDFS路径。

↓ 注意 自定义配置锁目录时,请确保整个集群的所有节点上不存在正在运行的MoveTo进程,否则可能导致 MoveTo执行失败,甚至导致数据污染。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。

# 4.4.5. JindoTable表或分区访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

#### 前提条件

已创建集群,详情请参见创建集群。

#### 背景信息

JindoTable支持收集访问Hive表的记录,收集的数据保存在SmartData服务的Namespace中。

Smart Dat a 3.2.x版本开始支持Spark、Hive和Presto引擎,Spark和Presto的数据收集默认是打开的,如果需要关闭,请参见关闭 <mark>热度收集</mark>。Hive的数据收集默认是关闭的,如果需要打开,请参见<mark>开启Hive热度收集</mark>。

#### 数据查询

JindoTable提供了命令方式查询热度信息。

● 语法

```
jindo table -accessStat <-d [days]> <-n [topNums]>
```

days和 topNums为正整数。当只设置天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

```
jindo table -accessStat -d 7 -n 20
```

JindoTable使用详情,请参见JindoTable使用说明。

#### 开启Hive热度收集

1.

- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的**集群管理**页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改Hive的参数值。

- i. 在左侧导航栏,选择**集群服务 > Hive**。
- ii.在Hive服务页面,单击配置页签。
- iii. 搜索参数hive.exec.post.hooks,在参数值后追加com.aliyun.emr.table.hive.HivePostHook。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - i. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
  - ii. 在执行集群操作对话框,输入执行原因。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

#### 关闭热度收集

1.

- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。
- 。 Hive服务:
  - a. 在左侧导航栏,选择**集群服务 > Hive**。
  - b. 在Hive服务页面,单击配置页签。
  - c. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据		
配置过滤 配置搜索 hive.exec.post.hooks	服务配置 全部 □ hive-site	
	hive.exec.post.hooks	k.HiveHook, com.aliyun.emr.table.hive.HivePostHook

- Spark服务:
  - a. 在左侧导航栏,选择集群服务 > Spark。
  - b. 在Spark服务页面,单击配置页签。
  - c. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSQLQueryListener。

状态 部署拓扑 配置 配置修改历史	
配置过速	服务配置
spark.sql.queryExecutionListeners O Q	± = p   spark-detaults
配置范围	sparksql.queryExecutionListeners comaliyun.emr.table.sparkSQLQueryListenet

- Presto服务:
  - a. 在左侧导航栏,选择集群服务 > Presto。
  - b. 在Presto服务页面, 单击配置页签。
  - c. 搜索参数event-listener.name, 删除参数值中的内容。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

# ⅲ. 单击**确定**。

- 7. 重启服务。
  - ∘ Hive服务:
    - a. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Spark服务:
    - a. 在Spark服务页面,选择右上角的操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Presto服务:
    - a. 在Presto服务页面,选择右上角的操作 > 重启All Components。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。

# 4.4.6. JindoTable表或分区访问冷度收集

JindoTable表或分区的访问冷度收集功能可以为您维护表或分区上次的访问时间,从而筛选出最近没有被访问的数据,帮助您优 化数据存储方式,节约成本。例如,在数据分析中,您可以把部分不常用的分区数据移动到成本更低的存储介质以节约成本。

#### 前提条件

已创建EMR-3.35.0及后续版本或EMR-4.9.0及后续版本,创建详情请参见创建集群。

#### 背景信息

Smart Dat a 3.5.x版本开始支持Hive、Spark和Presto组件的冷度收集功能。该功能目前默认不开启,如果需要开启,请参见开启 Spark冷度收集、开启Hive冷度收集和开启Presto冷度收集。

⑦ 说明 因为冷度收集与热度收集使用相同的hooks或Listeners,所以开启组件的冷度收集时会同时打开热度收集功能。 表或分区访问热度收集的详情,请参见jindoTable表或分区访问热度收集。

#### 使用限制

- 不支持DLF数据湖元数据。
- Hive CLI、HiveServer2、Spark SQL CLI、Spark Thrift server和Presto服务所在IP需要有权限访问元数据底层存储(MySQL或 RDS)。
- 仅支持Hive、Spark和Presto组件的冷度收集。

#### 数据查询

JindoTable提供了命令方式查询冷度信息。

语法

jindo table -leastUseStat -n <num> [-i/-ignoreNever]

num是显示的条目数量,应为正整数。-i/-ignoreNever为可选参数,如果设置该参数,则会过滤掉从未被访问过的表或分区。

功能

展示最久未被访问的表或分区。

• 示例:查询最久未被访问的表或分区的20条记录。

```
jindo table -leastUseStat -n 20
```

返回如下图所示三列结果。

tdb.t1 pid=20/qid=ten 2021-03-26 13:53:52 tdb.t2 2021-03-26 13:53:58

- 第一列为表的名字,格式:数据库名.表名。
- 。 第二列为分区名字,格式:第一分区列=列值/第二分区列=列值/...,如果表为非分区表则为空。
- 第三列为最近一次访问的时间,格式: yyyy-MM-dd HH:mm:ss。
  - ⑦ 说明 如果为分区表,则只显示到分区级别,表本身不会单独显示。

JindoTable使用详情,请参见JindoTable使用说明。

#### 开启Spark冷度收集

- 1. 进入Spark页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Spark。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.queryExecutionListeners,确保参数值包含com.aliyun.emr.table.sparkSparkSQLQueryListener,如果存 在多个Listeners时使用英文逗号(,)隔开。

< 返回 🖓 Spark 🗸 🔍 正常		④ 查看操作
状态 部署拓扑 配置修改历史		
配置过滤 配置搜索	服务配置 全部   spark-defaults	
Spendounder, inceedementerers 0 0 00	spark.sql.queryExecutionListeners com.aliyun.emr.table.spark.SparkSQLQueryList	ener,c 🛛

- 4. 添加自定义配置。
  - i. 在服务配置页面, 单击spark-defaults页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为spark.sql.query.update.access.time.enabled, Value为true。

新増配置项			×
* Key	* Value	描述	操作
spark.sql.query.update.access.time.enabled	true		删除
洒加			
			确定取消

- ⅳ. 单击确定。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。

iii. 在确认对话框中, 单击确定。

## 开启Hive冷度收集

- 1. 进入Hive页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - ⅲ. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Hive。
- 2. 在Hive服务页面,单击配置页签。
- 3. 搜索参数hive.exec.post.hooks,确保参数值包含com.aliyun.emr.table.hive.HivePostHook,如果存在多个hooks时使用英文逗号(,)隔开。

< 返回   🙀 Hive マ ● 正常	
状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤 配置搜索	服务配置 全部 I hive-site
TWEERELIDURS 0 4	hive.exec.post.hooks com.aliyun.emr.meta.hive.hook.LineageLoggerHook

4. 添加自定义配置。

i. 在服务配置页面,单击hive-site页签。

- ii. 单击右上角的自定义配置。
- iii. 在新增配置项对话框中,设置Key为hive.hook.update.access.time.enabled, Value为true。

新	新增配圈项 ×			
	* Key	* Value	描述	操作
	hive.hook.update.access.time.enabled	true		删除
汤加				
				确定取消

- iv. 单击**确定**。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

#### 开启Presto冷度收集

- 1. 进入Presto页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Presto。
- 2. 在Presto服务页面,单击**配置**页签。

3. 搜索参数event-listener.name,确保参数值包含jindo-presto-post-event-listener。

<返回   ≫ Presto ~ ● 正常	
状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置
event-listener.name	
配置范围	event-listener.name jindo-presto-post-event-listener

- 4. 添加自定义配置。
  - i. 在服务配置页面,单击event-listener.properties页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为listener.update.access.time.enabled,Value为true。

휫	新增配置项 ×			
	* Key	* Value	描述	撮作
	listener.update.access.time.enabled	true		删除
	添加			
				确定 取消

- iv. 单击**确定**。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在确认对话框中,单击确定。

# 4.5. 工具集

# 4.5.1. Jindo sql命令介绍

Jindo sql命令是JindoFS自带的工具,方便您分析JindoFS访问日志、元数据和OSS访问日志。本文为您介绍如何使用Jindo sql命 令,分析JindoFS访问日志、元数据和OSS访问日志的数据。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

#### 背景信息

您可以使用Jindo sql命令分析以下数据:

- 使用Jindo sql分析JindoFS访问日志
- 使用Jindo sql分析元数据
- 使用Jindo sql分析OSS访问日志

#### 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持使用Jindo sql命令。

#### 使用Jindo sql命令

1. 通过SSH方式登录集群,详情请参见登录集群。

#### 2. 执行以下命令,启动jindo sql。

jindo sql

jindo sql支持以下常用参数。

参数	描述
-f	指定运行的SQL文件。
-i	启动Jindo sql后自动运行初始化SQL脚本。
-d	参数设置为键值对的形式。例如, -d A=B 。

# Jindo sql内置表结构

● audit\_log\_source (分区表)

audit\_log\_source表用作JindoFS访问日志原始表。

参数	描述
datetime	时间格式yyyy-MM-dd HH:mm:ss。
allowed	本次操作是否被允许,取值如下: • true: 允许本次操作。 • false: 不允许本次操作。
ugi	操作用户(包含认证方式信息)。
ip	Client IP地址。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dst	目标路径,可以为空。
perm	操作文件的Permission信息。
date (分区列)	日志日期,格式为YYYY-mm-DD。

## audit\_log

audit\_log允许使用分区列进行分区过滤,用作JindoFS访问日志表。

参数	描述
datetime	时间格式yyyy-MM-dd HH:mm:ss。
allowed	本次操作是否被允许,取值如下: • true:允许本次操作。 • false:不允许本次操作。
ugi	操作用户(包含认证方式信息)。
ip	Client IP地址。
ns	Block模式namespace的名称。
cmd	操作命令。

参数	描述
src	源路径。
dst	目标路径,可以为空。
perm	操作文件的Permission信息。
date (分区列)	日志日期,格式为YYYY-mm-DD。

## ● fs\_image(分区表)

## fs\_image用作转存image信息

参数	描述
atime	INode最近访问时间。
attr	文件相关属性。
etag	OSS的ETag值。
id	INode的ID。
mtime	INode的修改时间。
name	INode的名称。
owner	owner名称。
ownerGroup	owner组名称。
parentId	父节点的ID。
permission	操作文件的Permission信息。
size	INode的大小。
state	INode的状态。
type	INode的类型。
storagePolicy	存储策略。
namespace(分区列)	namespace名称。
datetime (分区列)	转存时间。

#### oss\_access\_log\_source

如果开启分区表模式,则为分区表。oss\_access\_log\_source表用作OSS访问日志原始表。

参数	描述
line	原始日志。
bucket(分区列)	Bucket名称。
partition_date (分区列)	日志日期格式为YYYY-mm-DD。

# oss\_access\_log

如果开启分区表模式,允许使用分区列进行分区过滤。oss\_access\_log表用作OSS访问日志。

参数	描述
Remote_IP	请求者的IP地址。
Reserved	保留字段,固定值为-。
Reserved 1	保留字段,固定值为-。
Time	OSS收到请求的时间。
Request_URI	包含query string的请求URL。OSS会忽略以x-开头的query string参数,但这个参 数会被记录在访问日志中。所以您可以使用x-开头query string参数标记一个请 求,然后使用这个标记快速查找该请求对应的日志。
HTTP_Status	OSS返回的HTTP状态码。
SentBytes	请求产生的下行流量。单位:Byte。
RequestTime	完成本次请求耗费的时间。单位:ms。
Referer	请求的HTTP Referer。
User_Agent	HTTP的User-Agent头。
HostName	请求访问的目标域名。
Request_ID	请求的Request ID。
LoggingFlag	是否已开启日志转存。
Requester	请求者的用户ID。取值-表示匿名访问。
Operation	请求类型。
Bucket	请求的目标Bucket名称。
Кеу	请求的目标Object名称。
Object Size	目标Object大小。单位:Byte。
Server_Cost_Time	OSS处理本次请求所花的时间。单位:毫秒。
ErrorCode	OSS返回的错误码。取值-表示未返回错误码。
RequestLength	请求的长度。单位: Byte。
UserlD	Bucket拥有者ID。
Delta_DataSize	Bucket大小的变化量。取值-表示此次请求不涉及Object的写入操作。
SyncRequest	请求是否为CDN回源请求。取值如下: • cdn: 请求是CDN回源请求。 • -: 请求不是CDN回源请求。
StorageClass	目标Object的存储类型。取值如下: o Standard:标准存储。 o IA:低频访问存储。 o Archive:归档存储。 o Cold Archive:冷归档存储。 o -:未获取Object存储类型。

参数	描述
TargetStorageClass	是否通过生命周期规则或CopyObject转换了Object的存储类型。取值如下: • Standard:转换为标准存储。 • IA:转换为低频访问存储。 • Archive:转换为归档存储。 • Cold Archive:转换为冷归档存储 • -:请求不涉及Object存储类型转换操作。
T ransmissionAccelerat ionAccessPoint	通过传输加速域名访问目标Bucket时使用的传输加速接入点。取值-表示未使用传 输加速域名或传输加速接入点与目标Bucket所在地域相同。 例如,请求者通过华东1(杭州)的接入点访问目标Bucket时,值为cn- hangzhou。
AccessKeyID	访问的AccessKey ID。
bucket(分区列)	Bucket名称。
partition_date (分区列)	日志日期格式为YYYY-mm-DD。

# 使用Jindo sql分析JindoFS访问日志

JindoFS为存储在OSS上的JindoFS访问日志文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃IP分析。您可以通过 jindo sql 命令,使用该功能。

⑦ 说明 已开启AuditLog功能,详情请参见AuditLog使用说明。

Jindo SQL相关命令示例如下:

• 执行如下命令,显示表。

show tables;

⑦ 说明 表结构信息,请参见Jindo sql内置表结构。

返回信息如下图所示。



• 执行如下命令,显示分区。

show partitions audit\_log\_source;

返回信息类似如下图所示。

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken: 0.031 seconds, Fetched 1 row(s)
```

• 执行如下命令,查询数据。

select \* from audit\_log\_source limit 10;

返回信息类似如下图所示。

jindo-s	ql> select * fr	om audit_log_sou	rce limit	10;				
datetim	e allowe	dugi ip	ns	cmd src	dst perm	date		
2020-10	-20 10:50:11.92	4 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:ha	doop:rwxrwxr-x	2020-10-	20				
2020-10	-20 10:50:11.95	0 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10	-20 11:26:06.44	5 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:ha	doop:rwxrwxr-x	2020-10-	20				
2020-10	-20 11:26:06.46	9 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10	-20 11:26:11.29	5 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s	dst=null	perm=root:root	rwxr-xx	2020-10	-20			
2020-10	-20 11:26:11.32	0 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null	perm=null	2020-10-20						
2020-10	-20 11:26:14.36	8 allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	<pre>src=jfs: d</pre>
s	dst=null	perm=root:root	rwxr-xx	2020-10	-20			

select \* from audit log limit 10;

返回信息类似如下。

jindo-sql> s	select * from	1 audit_l	og li	mit 10										
datetime	allowed	ugi	ip	ns	cmd	src	dst	perm	date					
2020-10-20 1	10:50:11.924	true	root	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	wxrwx
r-x 2020	0-10-20													
2020-10-20 1	10:50:11.950	true	root	(auth:	SIMPLE)	192.	10.00	kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20 1	11:26:06.445	true	root	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	wxrwx
r-x 2020	0-10-20													
2020-10-20 1	11:26:06.469	true	root	(auth:	SIMPLE)	192.	10.00 C	kugou	listFileletRequest	jfs://k	11	null	2020-10	9-20
2020-10-20 1	11:26:11.295	true	root	(auth:	SIMPLE)	192.	11 A 11	kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	0-10-20													
2020-10-20 1	11:26:11.320	true	root	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	11:26:14.368	true	root	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	0-10-20													
2020-10-20 1	11:26:14.393	true	root	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	11:26:16.230	true	root	(auth:	SIMPLE)	192.	11 A M	kuaou	getFileStatusRequest	ifs://k		null	root:ro	ot:rw
xr-xx 2020	0-10-20													
2020-10-20 1	11:26:16.255	true	root	(auth:	SIMPLE)	192.		kuqou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time believes	1 010	In Easterly	- 1 10											

• 执行如下命令,统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 使用Jindo sql分析元数据

JindoFS为JindoFS上的元数据文件提供SQL的分析功能,通过SQL分析相关表。您可以通过 jindo sql 命令,使用该功能。

⑦ 说明 已开启AuditLog功能,详情请参见AuditLog使用说明。

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,启动Jindo sql。

jindo sql

- 3. 查询Jindo SQL可以分析的表格。
  - 使用 show tables 命令,可以查看支持查询分析的表格。Jindo Sql内置了审计和元数据信息的分析功能,对应 audit\_log和fs\_image。

代码示例如下图所示。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

使用 show partitions fs\_image
 命令,可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -du
 mpMetadata 生成的数据。

代码示例如下图所示。

jindo-sql> show partitions fs_image;	
partition	
namespace=k /datetime=2020_10_20_10_47_14	
namespace=k /datetime=2020_10_20_10_50_36	
namespace=k /datetime=2020_10_20_10_52_06	
Time taken: 0.045 seconds, Fetched 3 row(s)	

4. 查询分析元数据信息。

Jindo SQL使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

代码示例如下图所示。

[root@en	nr-worke	-2 hadoo	op]# jin	ndo sql													
Spark ma	aster: ye	arn, Appi	lication	n Id: app													
jindo-se	al> show	tables;															
database		tableNar	me	isTempo	rary												
default	audit_lo	xg	false														
default	audit_le	g_source	e	false													
default	fs_image		false														
Time tak	cen: 0.34	15 second	ds, Fet	ched 3 ro	m(s)												
jindo-so	ql> sele	t * from	m fs_im	age limit	10;												
atime	attr	etag	id	mtime	name	owner	ownerGr	oup	parentId	permiss	ion	size	state	storage	Policy	type	name
space	datetim																
0			731107	600505189	9448	1603084	4070081	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_	sk=2450819	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			1653444	480419066	75495	160308	1071350	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_!	sk=2450820	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			731107	600505189	9470	1603084	4070185	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450821	root	root	334790	833296
5855433	489	0	Finali	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			119227	620234792	87249	1603084	4069581	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450822	root	root	334790	833296
5855433	489	Ø	Finali	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			107698	405188724	41036	1603084	4073592	/tpcds.	/orc/5000/web	_returns/wr_	returned	_date_	sk=2450823	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	pry	kugou	2020_10_20_	10_50_36							
0			269938	998662451	1354	160308	1068996	/tpcds	/orc/5000/web	_returns/wr_	returned	_date_s	sk=2450824	root	root	334790	833296
5855433	489	Ø	Finalia	zed	WARM	Directo	pry	kugou	2020_10_20_	10_50_36							
0			1192270	620234792	87307	160308	4069875	/tpcds.	/orc/5000/web	_returns/wr_	returned_	_date_:	sk=2450825	root	root	334790	833296
5855433	489	ø	Finali	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			154646	848201766	500Z	1603084	4072440	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450826	root	root	334790	833296
5855433	489	ø	Finali	zed	WARM	Directo	bry	kugou	2020_10_20_	10_50_36							
0			1653444	480419066	75460	1603084	4071170	/tpcds.	/orc/5000/web	_returns/wr_	returned,	_date_	sk=2450827	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	bry	kugou	2020_10_20_	10_50_36						224200	
0			7311070	600505189	9244	160308	1070572	/tpcds.	/orc/5000/web	_returns/wr_	returned.	_date_s	SK=2450828	root	root	334790	833296
5855433	489	0	Finalia	zed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							

例如: 根据某次转存的元数据信息统计该namespace下的目录个数。

jindo-sql> select count(\*) from fs\_image where type = "Directory" and namespace="kugou" and datetime="2020\_10\_20\_10\_47\_14"; count(1) 11837 Time taken:\_6.852 seconds, Fetched 1 row(s)

The careful store seconds, recence i ron(s)

⑦ 说明 namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

# 使用Jindo sql分析OSS访问日志

○ 注意 分析OSS访问日志需要指定OSS访问日志目录和指定是否为分区表,指定分区表会自动按照Bucket或date进行日志归档,能够支持使用过滤语句指定查询某个分区,极大的提升了查询效率,但是开启分区表之后必须每次使用分区表模式,否则文件会被归档到目录导致部分数据无法查询。

JindoFS为存储在OSS上的OSS访问日志文件提供SQL的分析功能,通过SQL分析相关表。您可以通过\_\_\_jindo\_sql\_\_命令,使用该 功能。

```
⑦ 说明 已开启日志转存,详情请参见日志转存。
```

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,启动Jindo sql。

jindo sql

3. 执行以下命令,指定日志存储路径和表类型。

jindo sql -d access\_log\_path=oss://test-sh/oss-accesslog -d partition.table.enabled=true

```
代码中的 access_log_path 为OSS访问日志存储路径, partition.table.enabled 指定是否为分区表, true表示为分区
表。
```

#### 常见问题

• Q: 如何修改初始资源Jindo sql的启动参数?

A: 因为 Jindo sql 基于Spark的程序,所以初始资源可能较小,您可以通过环境变量 JINDO\_SPARK\_OPTS 来修改初始资源 Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.exec utor.cores=5 --conf spark.executor.memory=20G"

• Q: 如何使用Hive分析表?

A:为了避免污染Hive元数据,默认Hive看不到Default下的几个表,如果想使用Hive分析这些表,可以通过语句 show create table {table\_name} 查看表语句或者使用SQL创建新表,Hive需要执行加载外部表。

# 4.5.2. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

#### 前提条件

已创建集群,详情请参见创建集群。

#### 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

#### 挂载

? 说明 依次在每个节点上执行挂载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

#### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

返回用户在服务端配置的所有命名空间列表。
test testcache

#### 2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

### 3. 创建目录。

mkdir /mnt/jfs/test/dirl
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/

### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

### 如果您想使用Python方式写入和读取文件,请参见如下示例:

#### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

#### 返回如下信息。

my first file This file

### 卸载

```
? 说明 依次在每个节点上执行卸载操作。
```

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 4.5.3. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见<mark>登录集群</mark>。

2. 执行以下命令,获取帮助信息。

jindo distcp --help

#### 返回信息如下。

help - Print help text
src=VALUE - Directory to copy files from
dest=VALUE - Directory to copy files to
parallelism=VALUE - Copy task parallelism
outputManifest=VALUE - The name of the manifest file
previousManifest=VALUE - The path to an existing manifest file
requirePreviousManifest=VALUE - Require that a previous manifest is present if specified
copyFromManifest - Copy from a manifest instead of listing a directory
srcPrefixesFile=VALUE - File containing a list of source URI prefixes
srcPattern=VALUE - Include only source files matching this pattern
deleteOnSuccess - Delete input files after a successful copy
outputCodec=VALUE - Compression codec for output files
groupBy=VALUE - Pattern to group input files by
targetSize=VALUE - Target size for output files
enableBalancePlan - Enable plan copy task to make balance
enableDynamicPlan - Enable plan copy task dynamically
enableTransaction - Enable transation on Job explicitly
diff - show the difference between src and dest filelist
ossKey=VALUE - Specify your oss key if needed
ossSecret=VALUE - Specify your oss secret if needed
ossEndPoint=VALUE - Specify your oss endPoint if needed
policy=VALUE - Specify your oss storage policy
cleanUpPending - clean up the incomplete upload when distcp job finish
queue=VALUE - Specify yarn queuename if needed
bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed
s3Key=VALUE - Specify your s3 key
s3Secret=VALUE - Specify your s3 Sercet
s3EndPoint=VALUE - Specify your s3 EndPoint

### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo DistCp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo DistCp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp

② 说明 本文示例中的yourBucket Name是您OSS Bucket 的名称。

### --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制/data/incoming/hourly\_table/2017-02-01/03下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Found 6 items	3						
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09							
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,复制以log结尾的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPattern .\*\.lo
g --parallelism 20

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-

      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-

      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --deleteOnSuccess -parallelism 20

#### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下:

Found 6 item	S		
-rw-rw-rw-	1	938 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/000151.sst.gz</yourbucketname>	
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/1.log.gz</yourbucketname>	
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/2.log.gz</yourbucketname>	
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/OPTIONS-000109.c</yourbucketname>	g
Z			
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/emp01.txt.gz</yourbucketname>	
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly table/2017-02-01/03/emp06.txt.gz</yourbucketname>	

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。

### --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

### --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录, 查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly_table/manifest-2020-04-17.gz --parallel ism 20
```

#### 执行以下命令,查看运行期间复制的文件。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst

#### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly table","size":4891)

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
      Found 4 items

      drwxrwxrwx -
      0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

      drwxrwxrwx -
      0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

#### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

例如,执行以下命令,查看如下文件夹中的数据。

```
hdfs dfs -ls /data/incoming/hourly_table/2017-02-01/03
```

### 返回信息如下。

Found 8 item	s						
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09							
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

#### --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为DistCp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo DistCp的作业分配计划,以达到更好的DistCp性能。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --enableBalancePlan
--parallelism 20

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

### --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定 --enableDynamicPlan 来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

### --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的Dist Cp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis m 20

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yam队列的名称。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

#### --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

#### 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

● 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive --parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

#### 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

#### 命令示例如下。

```
jindo distop --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --cleanUpPending --
parallelism 20
```

### 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的*core-site.xm*文件里 ,避免每次使用时填写Accesskey。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

#### 此时代码示例如下。

jindo distcp /tmp/jindo-distcp-2.7.3.jar --src s3://smartdatal/ --dest s3://smartdatal/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

### 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG REDUCE=0
```

⑦ **说明** 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的 大小可能不相等。

# 4.5.4. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

### 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-distcp-<version>.jar。
   Hadoop 2.7及后续版本,请下载jindo-distcp-3.0.0.jar。

• Hadoop 3.x系列版本,请下载jindo-dist cp-3.0.0.jar。

#### 场景预览

Jindo Dist Cp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二: 使用Jindo Dist Cp成功导完数据后, 如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?
- 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Dist cp过程中可能已经产生了新文件,该使用哪些参数处理?
- 场景五:如果需要指定Jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六: 当通过低频或者归档形式写入OSS, 该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九:如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时,该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二: 如果Copy完文件, 需要删除原文件, 只保留目标文件时, 该使用哪些参数?
- 场景十三: 如果不想将OSS AccessKey这种参数写在命令行里, 该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret ),以及Endpoint信息,且该AccessKey具有写目标Bucket的权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

#### 本场景示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10
```

⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch

### 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

```
您可以在MapReduce任务结束的Counter信息中,获取Dist cp Counters的信息。
```

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

#### 参数含义如下:

- Bytes Destination Copied: 表示目标端写文件的字节数大小。
- Bytes Source Read: 表示源端读文件的字节数大小。
- Files Copied:表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

### 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

# 在<mark>场景</mark>一的基础上,如果您的Dist cp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Dist cp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文 件的Copy。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table le --dest oss://yang-hhht/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa nifest --parallelism 20

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

### 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 参数含义如下:

- --outputManifest
   : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如manifest-2020-04-17.gz,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Dist cp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest-2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

### 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid th 6 --parallelism 10

#### 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

### 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

#### 优化对比如下。



### 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly\_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10

### 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

### 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

### 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则, 正则表达式。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.*/([a-z]+).*.txt' --parallelism 20
```

### 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10
```

### 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

### E-MapReduce

```
<configuration>
       <property>
          <name>fs.jfs.cache.oss-accessKeyId</name>
          <value>xxx</value>
       </property>
       <property>
           <name>fs.jfs.cache.oss-accessKeySecret</name>
           <value>xxx</value>
       </property>
       <property>
           <name>fs.jfs.cache.oss-endpoint</name>
           <value>oss-cn-xxx.aliyuncs.com</value>
       </property>
   </configuration>
• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在 core-site.xml中。
   <configuration>
       <property>
           <name>fs.s3a.access.key</name>
```

```
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

# 4.5.5. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存储成本,或者加速访问数据的速度。

### 使用Jindo jfs

执行以下命令,获取帮助信息。

jindo jfs -help archive

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

### Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p参数可以保证本地数据不受磁盘水位清理。

### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

### Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参见对象存储OSS的存储类型介绍。

jindo jfs -archive -i|-a|-c <path>

指定以下参数时:

- -i: 表示可以归档数据至低频存储类型。
- -a: 表示可以归档数据至归档存储类型。
- -c: 表示可以归档数据至冷归档存储类型。

#### Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

Unarchive默认可以将数据恢复成标准存储,指定以下参数时:

- -i: 表示可以恢复数据至低频存储类型。
- -o: 表示可以临时解冻归档存储类型, 使数据临时可读。

### Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

#### 指定以下参数时:

- -detail: 表示可以查看文件进度信息。
- -sync: 表示该命令需要同步等待分层存储任务结束才会退出。

### ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -1s2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse -rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia\_data.csv -rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia\_data.json

# 5.SmartData 3.6.x 5.1. SmartData 3.6.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS、JindoTable和相关工具集。本文为您介绍Smart Dat a(3.6.x)版本的更新内容。

### JindoFS

### 此版本中JindoFS的新特性如下表所示。

特性	描述
JindoFS支持多云和S3协议及缓存加速	JindoFS实现对S3协议的支持,具备访问亚马逊S3及其他S3协议系统 的能力,并且还支持缓存加速功能,使访问更高效。
JindoFS支持HDFS缓存加速	JindoFS访问HDFS新增支持缓存加速,使访问更高效。
JindoFS支持MetaSync和Data Cache事务性	JindoFS元数据缓存上支持了事务性加载的可选项,能够保证整个预 加载任务的事务性,保证加载过程中不会在元数据视图上出现非完整 的中间状态。
JindoFS优化缓存预加载机制	<ul> <li>优化了元数据和数据缓存的预加载机制,提升了执行效率。</li> <li>优化进度反馈。</li> <li>支持了针对超大目录的任务。</li> </ul>

### JindoSDK

### 此版本中JindoSDK的新特性如下表所示。

特性	描述
JindoSDK支持本地缓存(Local)策略	JindoSDK支持本地缓存策略,使得在未部署SmartData服务的情况 下,也能支持本地数据缓存,提高OSS数据的访问效率。
JindoSDK支持和FileSystem平级的Object Store API	JindoSDK在已有FileSystem语义支持的基础上,新增支持平级的 Object Store API语义,与对象存储系统OSS等有更直接的对应关 系,方便使用。Object Store API优化了Copy和Rename等操作,提 升了执行效率。
JindoSDK支持OSS服务端缓存优化	JindoSDK支持OSS加速器功能,开启后,您可以根据加速器的容量定 制OSS的访问带宽。
JindoSDK支持OSS二级域名Endpoint	JindoSDK新功能,开启后,支持在特定环境下,使用二级域名或者IP 地址方式访问OSS服务。默认不开启二级域名Endpoint访问。

### JindoTable

### 此版本中JindoTable的新特性如下表所示。

特性	描述
JindoTable支持HDFS数据分层存储和归档到OSS	JindoTable新增命令,能够批量移动表或分区到OSS,并同步更新元 数据。支持通过过滤条件选择分区,支持指定拷贝到OSS的存储策 略。对于已经在OSS的数据,支持批量归档。
JindoTable支持OSS归档数据解冻和取回	JindoTable新增命令,对于存储在OSS的数据,能够批量进行数据的 解冻和取回。
JindoTable支持Presto查询加速Parquet数据	JindoTable NativeEngine查询加速引擎能够支持Presto和Parquet 组合的场景,显著提升Presto查询Parquet数据效率。

特性	描述
JindoTable支持Spark 3查询加速Parquet和ORC数据	JindoTable NativeEngine查询加速引擎新增支持Spark 3计算引擎, 同时支持Parquet和ORC格式的数据,全面提升Spark3查询效率。
JindoTable 支持查询加速HDFS存储上的数据	JindoTable NativeEngine查询加速引擎新增支持HDFS存储,Spark 和Presto能够通过NativeEngine高效读取HDFS文件。
JindoTable支持分析OSS访问日志	支持使用SQL语句分析OSS访问日志。

### JindoFuse

#### 此版本中JindoFuse的新特性如下表所示。

特性	描述
JindoFuse完善支持训练和在线场景	JindoFuse新增支持指定JindoFS命名空间挂载,或者以SDK模式指定 OSS目录挂载。

# 5.2. JindoFS Block模式

# 5.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

### 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

### 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	● 正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	素			全部 i smartdata-site namespace client i storage
(清頼)	λ		Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

- i. 修改jfs.namespaces为test。
- test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。
- ii. 单击自定义配置, 在新增配置项对话框中增加以下参数, 单击确定。

参数	描述	示例		
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>		
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。		
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block		
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX		
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>		

### ⅲ. 单击确定。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

### 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在**服务配置**区域的storage页签,修改如下参数。

服务配置							
全部   smartdata-site   namespace   client   storage							
	storage.handler.threads	40		0			
	storage.watermark.low.ratio	0.2		0			
	storage.watermark.high.ratio	0.4		0			
	storage.oss.upload.threads	20		0			
参数		描述					
storage.watermark.high.ratio			表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。				
storage.watermark.low.ratio			表示使用量的下水位比例,触发清理后会自动清理冷数据,将 JindoFS数据目录占用空间清理到下水位。默认值:0.2。				

```
⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。
```

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 5.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

### 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见<mark>使用Raft-RocksDB-</mark> Tablestore作为存储后端。

单机RocksDB作为元数据服务的架构图如下所示。



### 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。

#### ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过滤				服务配置	
配置搜	卖				全部   smartdata-site   namespace   client   storage
清諭)				Q	

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中,输入执行原因,开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例	
namespace.ots.instance	Tablestore实例名称。	emr-jf s	
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk	
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX	
namespace.ots.endpoint	Tablestore实例的Endpoint地址, 推荐使 用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com	
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true	
	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。		

7. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 8. 单击右上角的操作 > 启动All Components。

### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. 准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809 25 jfs://test/
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 \_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ jindo jfs -metaStatus				
<pre>==== emr-header-1:8101 =====</pre>				
OtsUploader: _synced=1				
[RocksDB Row Counts of each CF(Table)]				

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。

3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: ° true ° false	true

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

iii. 单击确定。

- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

==== emr-header-1:8101 ====	
[Recovery From OTS Status]	
state: FINISH	
total 22855 rows.	

(可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。
 此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一数
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
    1596 1482809 25 jfs://test/
# 文件可正常读取(cat, get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: o true o false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: o true o false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面, 单击相应集群所在行的更多 > 重启。

# 5.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

#### 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

```
⑦ 说明 需要开启事务功能。
```

• 创建3 Master的EMR集群,详情请参见创建集群。

🖾 高可用	高可用: 🕜		
	部署方式: 🕜	🔾 2 Master	
	0	3 Master	查看服务部署
? 说明	如果没有部署	方式,请 <mark>提交</mark>	工单处理。

### 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





### 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏, 单击集群服务 > Smart Data。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏, 单击集群服务 > SmartData。
  - ii. 单击**配置**页签。
  - iii. 在**服务配置**区域,单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: 。 rocksdb 。 ots 。 raft 默认为rocksdb。	raft

参数	描述	示例
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤 7。

#### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jf s
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.raft.async.ot s.enabled	<ul> <li>是否开启OTS异步上传,包括:</li> <li>true</li> <li>false</li> <li>当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。</li> <li>⑦ 说明 如果SmartData服务已</li> </ul>	true
	完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

#### 6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

7. 单击右上角的操作 > 启动All Components。

#### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
(文件夹个数) (文件个数)
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 \_\_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail

[RaftPeerImpl]
peer 1d:
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers: In the second
changing_conf: N0 stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
<pre>vote_timer: timeout(5000ms) STOPPED</pre>
<pre>stepdown_timer: timeout(5000ms) SCHEDULING(in 2335ms)</pre>
snapshot_timer: timeout(3600000ms) SCHEDULING(in 150305ms)
storage: [1, 624625]
disk_index: 624625
known_applied_index: 624625
last_log_id: (index=624625,term=2)
first_index_pinned: 624625
state_machine: Idle
last_committed_index: 624625
last_snapshot_index: 0
last_snapshot_term: 0
snapshot_status: IDLE
replicator_2576980378901 replicator_2
replicator_329853488332 next_index=624626 flying_append_entries_size=0 idle hc=2301 ac=623564 ic=0
OtsUploader: _lastStopIndex=624624, _synced=1
[DeckenD Daw County of each (C(Table)]

iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。

#### 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

#### 3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

#### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus -detail

如图所示, LEADER节点的state为FINISH表示恢复完成。

```
[RaftPeerImpl]
peer_id: _____ :8103:0
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: NO stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
vote_timer: timeout(5000ms) STOPPED
stepdown_timer: timeout(5000ms) SCHEDULING(in 3382ms)
snapshot_timer: timeout(600000ms) SCHEDULING(in 474855ms)
storage: [1, 153]
disk_index: 153
known_applied_index: 153
last_log_id: (index=153,term=2)
first_index_pinned: 1
state_machine: Idle
 last_committed_index: 153
last_snapshot_index: 1
last_snapshot_term: 2
 snapshot_status: IDLE
replicator_1116691496965@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0 replicator_3311419785217@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0
 [Recovery From OTS Status]
 state: FINISH
 total rows: 1484409
 table `jfs_block_test` 2 rows.
table `jfs_namespace_cache_ns` 1 rows.
 table `jfs_namespace_test` 1484406 rows.
```

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

#### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

## 5.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功能,所以您可以通过Shell命令分析指定的日志文件。

### 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

### 使用AuditLog

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面, 单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。

### i. 单击**配置**页签。

ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置援	<b>康</b>			全部 I smartdata-site namespace client I storage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。
  - ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: Itrue:打开AuditLog功能。 Ifalse:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

### iii. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在确认对话框中, 单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

#### • jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

• show partitions table\_name 获取所有分区。

● desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

#### 示例如下:

#### • 执行如下命令显示表。

show tables;

jindo-sql> show	tables;	
database	tableName	isTemporary
default audit_lo	g false	
default audit_lo	g_source	false
default fs_image	false	

• 执行如下命令显示分区。

show partitions audit\_log\_source;

返回信息类似如下。

jindo-sql> show partitions audit\_log\_source; partition date=2020-10-20 Time taken: 0.031 seconds, Fetched 1 row(s)

• 执行如下查询数据。

select \* from audit\_log\_source limit 10;

返回信息类似如下。

jindo-s	ql> select * fro	om audit_lo	g_source limit	10;				
datetim	e allowe	dugi i	p ns	cmd src	dst per	m date		
2020-10	-20 10:50:11.92	1 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	doop:rwxrwx	r-x 2020-10-	20				
2020-10	-20 10:50:11.95	0 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=listFileletRequest src	=jfs://kugou/ =
null	perm=null	2020-10-2	0					
2020-10	-20 11:26:06.44	5 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	doop:rwxrwx	r-x 2020-10-	20				
2020-10	-20 11:26:06.46	9 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=listFileletRequest src	=jfs://kugou/ =
null	perm=null	2020-10-2	0					
2020-10	-20 11:26:11.29	5 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s	dst=null	perm=root	:root:rwxr-xx	2020-10	0-20			
2020-10	-20 11:26:11.32	0 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=listFileletRequest src	=jfs://kugou/=
null	perm=null	2020-10-2	0					
2020-10	-20 11:26:14.36	3 allowed=t	rue ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root	:root:rwxr-xx	2020-10	0-20			

select \* from audit\_log limit 10;

返回信息类似如下。

jindo-sql>	select * from	n audit_lo	bg li	mit 10;										
datetime	allowed	ugi i	.p	ns	cmd	src	dst	perm	date					
2020-10-20	10:50:11.924	true r	oot	(auth:SI	IMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	hadoop:ri	wxrwx
r-x 202	10-10-20													
2020-10-20	10:50:11.950	true r	oot	(auth:SI	IMPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20	11:26:06.445	true r	oot	(auth:SI	IMPLE)	192.		kugou	getFileStatusRequest	jfs://k	<b></b> 11	hadoop:h	hadoop:ri	wxrwx
r-x 202	20-10-20													
2020-10-20	11:26:06.469	true r	oot	(auth:SI	IMPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20	11:26:11.295	true r	oot	(auth:SI	IMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 202	20-10-20													
2020-10-20	11:26:11.320	true r	oot	(auth:SI	IMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20	11:26:14.368	true r	oot	(auth:SI	IMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 202	20-10-20													
2020-10-20	11:26:14.393	true r	oot	(auth:SI	IMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20	11:26:16.230	true r	oot	(auth:SI	IMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 202	20-10-20													
2020-10-20	11:26:16.255	true r	oot	(auth:SI	IMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ls. Fetche	ed 10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-sql> select cmd	,count(*)	from audit_log	where dat	e="2020-10-20"	group by	cmd order	by cmd;	
cmd count(1)								
getFileStatusRequest	387							
listFileletRequest	387							
Time taken: 5.767 seco	onds. Fetch	ned 2 row(s)						

# 5.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

### 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

总览信息(Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview	
Start Time:	Fri Oct 16 12:29:24 2020
Status:	Active
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)
Node:	Live Nodes: [2], Decommission Nodes: [0]
Version:	3.0.0
Build No:	fa0ea608a4/

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespa	Namespace Info (1)				
Namespace:	jfs://test/				
Namespaces:	test				
Mode:	BLOCK_MODE				
Backend URI:	oss://				
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27				

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

## Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

#### • 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

## 5.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

### 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



### 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜	素			全部 smartdata-site namespace client storage
请输入			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。

### 5. 重启配置。

```
i. 单击右上角的操作 > 重启 Jindo Namespace Service。
```

ii. 输入执行原因, 单击确定。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

### 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见概述。

ii. Ranger UI添加HDFS service。

Ranger	Access Manager	🗅 Audit	Settings				👸 admin
Service Ma	nager						
Service Ma	nager						🛛 İmport
B	IDFS		+ 4 2	B HBASE	+ 20	I HIVE	+ 2 2
				emr-hbase	• 7	emr-hive	<ul> <li>Image: A state of the state of</li></ul>
				emr-noase	· · · · · · · · · · · · · · · · · · ·	emr-nive	

#### iii. 配置相关参数。

参数	描述	
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}/。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal		
dfs.namenode.kerberos.principal	不填写。	
dfs.secondary.namenode.kerberos.principal		
Add New Configurations		

iv. 单击Add。

### 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hado op.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

## 5.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

### 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 <del>令使用说明</del>。

### 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

### OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

#### 其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令,获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

### 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用于一些高压缩比的文件。支持以下两种压缩策略。

|--|

NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path> ...

#### 其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path> : 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

# 5.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

### 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

### 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

#### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

( :bin jindo jfs -dumphetadata test-block SLF4J: Elass path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.httpl#uucuple\_bindings for explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] Sucessfully upload namespace metadata to 0SS.

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

Sucessfully upload namespace metadata to OSS.

### 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如,配置的	namespace.sysinfo.oss.uri	为 oss://abc/	,则上传的又件会在	oss://abc/metadataDump	子目录中。
参数			说明		
namespace.s	sysinfo.oss.uri		存储Bucket和路径。		

\_\_\_\_
参数	说明
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinfo.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

← → ↑ 1 oss://emrtest/sysinfc/metadataDump	test-block/
▲上传 + 创建目录 □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	
□ 名称	类型 / 大小
□ <b>□</b> 2020_09_14_18_58_16 <b>3</b>	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

## 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

```
{
                    /*INode类型,FILE文件DIRECTORY目录*/
 "type":"string",
                          /*INode id*/
 "id": "string",
 "parentId":"string", /*父节点id*/
"name":"string", /*INode名称*/
"size": "int", /*INode大小, bigint*/
 "permission":"int",
                            /*permission以int格式存放*/
                      /*permiee.
/*owner名称*/
---组夕称:
 "owner":"string",
  "ownerGroup":"string", /*owner组名称*/
                 /*inode修改时间,bigint*/
/*inode最近访问时间,bigint*/
  "mtime":"int",
 "atime":"int",
                            /*文件相关属性*/
/*INode状态*/
 "attributes":"string",
 "state":"string",
 "storagePolicy":"string", /*存储策略*/
 "etag":"string" /*etag*/
}
```

## 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 *audit\_log*和*fs\_image*。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta

data <mark>生成的数据。</mark>

示例如下。

jindo-sql> show partitions fs_image; partition	
namespace=k /datetime=2020_10_20_10_47_14	
namespace=k /datetime=2020_10_20_10_50_36	
namespace=k /datetime=2020_10_20_10_52_06	
Time taken: 0.045 seconds, Fetched 3 row(s)	

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@em	ir-worker	-2 hadoo	op]# jin	do sql													
Spark ma	ster: yo	arn, Appl	lication	Id: app													
jindo-sq	1> show	tables;															
database		tableNam	ne	isTempo	rary												
default	audit_la	xg	false														
default	audit_la	g_source		false													
default	fs_image		false														
Time tak	en: 0.34	15 second	is, Fetc	hed 3 ro	n(s)												
jindo-sq	l> seled	t * from	n fs_ima	ge limit	10;												
atime	attr	etag	id	mtime	natte	owner	ownerGr	oup	parentId	permiss	ion	size	state	storage	Policy	type	name
space	datetime																
0			7311076	00505189	9448	1603084	070081	/tpcds/	orc/5000/web.	returns/wr_	returned,	_date_:	sk=2450819	root	root	3347908	33296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	0_50_36							
0			1653444	80419066	75495	1603084	071350	/tpcds/	orc/5000/web	_returns/wr_	returned.	_date_:	sk=2450820	root	root	3347908	33296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	10_50_36							
0			7311076	00505189	9470	1603084	070185	/tpcds/	orc/5000/web.	_returns/wr_	returned.	_date_:	sk=2450821	root	root	3347908	33296
5855433	489	Ø	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	10_50_36							
0			1192276	20234792	87249	1603084	069581	/tpcds/	orc/5000/web.	_returns/wr_	returned,	_date_:	sk=2450822	root	root	3347908	33296
5855433	489	Ø	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	0_50_36							
0			1076984	05188724	41036	1603084	073592	/tpcds/	orc/5000/web.	returns/wr_	returned	_date_:	sk=2450823	root	root	3347908	33296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	10_50_36							
0			2699389	98662451	1354	1603084	068996	/tpcds/	orc/5000/web_	_returns/wr_	returned	_date_:	sk=2450824	root	root	3347908	33296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	0_50_36							
0			1192276	20234792	87307	1603084	069875	/tpcds/	orc/5000/web_	_returns/wr_	returned.	_date_:	sk=2450825	root	root	3347908	33296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	10_50_36							
0			1546468	48201766	5002	1603084	072440	/tpcds/	orc/5000/web.	_returns/wr_	returned.	_date_	sk=2450826	root	root	3347908	33296
5855433	489	ø	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	10_50_36							
0			1653444	80419066	75460	1603084	071170	/tpcds/	orc/5000/web.	_returns/wr_	returned,	_date_:	sk=2450827	root	root	3347908	33296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	10_50_36							
0			7311076	00505189	9544	1603084	070572	/tpcds/	orc/5000/web	_returns/wr_	returned	_date_:	sk=2450828	root	root	3347908	33296
5855433	489		Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_1	0_50_36							
Time tak	en: 5 76	A second	te Eatel	hed 10 n	come a b												

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

```
jindo-sql> select count(*) from fs_image where type = "Directory" and namespace="kugou" and datetime="2020_10_20_10_47_14";
count(1)
11837
Time taken:_6.852 seconds, Fetched 1 row(s)
```

## 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table\_name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

#### 示例如下。

nive> select * from inode_metadata_test8 limit 100; /ARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futu			ising a differe	nt execution en	gine (i.e. spark,	tez) or using		eleases.
Query ID = root_202089								
Fotal jobs = 1								
aunching Job 1 out of 1.								
umber of reduce tasks is set to 0 since there's no reduce operator								
Starting Job = job_1 , Tracking URL = http://emr-heade	: 2088	18/proxy/app						
<pre><ill -kill="" command="/usr/lib/hadoop-current/bin/hadoop" job="" job_1595<="" pre=""></ill></pre>								
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: U								
2020-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0% Completion CDU 4 22								
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Lumulative LPU 1.22 se								
hapkeouce lotal cumulative LPU time: I seconds 220 msec								
hared Job -								
Hapkeoute Jobs Launcheo.	EDA CHECESE							
Stage-Stage-1, hap, 1 - Cumulative (10, 1,2 Sec. HDFS Read, 000) HDFS write, 1	.924 SOCCESS							
NK								
Directory 11274334386847219714 11274334386847219713 /uttest/oss		caoiie	staff 15995	45017615 1599	545017615	Finalized	WARM	
Directory 11274334386847219719 11274334386847219713 /uttest/oss2	0 511	caoiie	staff 15995	45017654 1599	545017654	Finalized	WARM	
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir		caojie	staff 15995	45017636 1599	545017636	Finalized		
ile 11274334386847219715 11274334386847219714 /uttest/oss/file1		caojie	staff 15995	45017632 1599	545017632		WARM	
ile 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2		caojie	staff 15995	45017642 1599	545017642		WARM	
ile 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3		caojie	staff 15995	45017651 1599	545017651		WARM	
Directory 11274334386847219720 11274334386847219719 /uttest/oss2/dir				159954501765			alized	WARM
ile 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2			staff 15995		545017658			
ile 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3			staff 15995		545017666		WARM	
Directory 11274334386847219713 17672823557433851905 /uttest 0			1599545017615	159954501761		zed WAR		
Fime taken: 10.734 seconds, Fetched: 10 ro⊮(s)								

## 5.2.9. JindoFS Credential Provider使用说明

JindoFS的数据存储在OSS中,如果您需要访问JindoFS的数据,需要提供OSS的AccessKey才能访问。Smartdata 3.4.0及后续版 本支持JindoFS Credential Provider,您可以通过配置JindoFS Credential Provider,将加密后的AccessKey信息添加至文件中,以 避免泄露AccessKey信息。

## 配置JindoFS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入smartdata-site服务配置。
  - i. 单击配置页签。
  - ii. 在服务配置区域,单击smartdata-site页签。
- 3. 添加配置信息。
  - i. 在smartdata-site页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下配置。

参数	描述
fs.jfs.credentials.provider	配置com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现炎,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential。例如, com.aliyun. emr.fs.auth.TemporaryAliyunCredentialsProvider,c om.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,c vider,com.aliyun.emr.fs.auth.EnvironmentVariable CredentialsProvider 。 Provider详情请参见Provider类型。

#### ⅲ. 单击确定。

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## Provider类型

• TemporaryAliyunCredentialsProvider

适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsPr ovider
fs.jfs.accessKeyId	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。
fs.jfs.securityToken	OSS的SecurityToken(临时安全令牌)。

• SimpleAliyunCredentialsProvider

#### 适合使用长期有效的AccessKey访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvid er
fs.jfs.accessKeyld	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。

EnvironmentVariableCredentialsProvider

#### 该方式需要在环境变量中配置以下参数。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EnvironmentVariableCredential sProvider
ALIYUN_ACCESS_KEY_ID	OSS的AccessKey ID。
ALIYUN_ACCESS_KEY_SECRET	OSS的AccessKey Secret。

参数	参数说明				
	OSS的SecurityToken(临时安全令牌)。				
ALIYUN_SECURITY_TOKEN	⑦ 说明 仅配置有时效Token时需要。				

#### • JindoCommonCredentialsProvider

该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.JindoCommonCredentialsProvi der
jindo.common.accessKeyId	OSS的AccessKey ID。
jindo.common.accessKeySecret	OSS的AccessKey Secret。
jindo.common.securityToken	OSS的SecurityToken(临时安全令牌)。

#### • EcsStsCredentialsProvider

该方式无需配置AccessKey,可以免密方式访问OSS。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EcsStsCredentialsProvider

## 5.2.10. JindoFS Block模式加密使用说明

JindoFS Block模式支持文件加密,加密机制和使用方法与Apache HDFS的Encryption Zone类似。加密密钥通过密钥管理服务 (KMS)统一管理,您可以对有敏感数据的目录设置加密策略,然后就可以透明地在该目录下加密写入的数据和解密读取的数据,无需更改您的代码。

## 前提条件

- 已创建集群,详情请参见创建集群。
- 已开通密钥管理服务(KMS),详情请参见<mark>开通密钥管理服务</mark>。

## 背景信息

#### Block模式加密架构图如下:



## 配置JindoFS使用阿里云KMS

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 在**服务配置**区域,单击namespace页签。
- 3. 添加配置信息。
  - i. 在namespace页签, 单击右上角的自定义配置。
  - ii. 在新增配置项对话框中,新增如下配置。

参数	描述
crypto.provider.type	Provider的类型,仅支持ALIYUN。
crypto.provider.endpoint	KMS的公网接入地址。详情请参见 <mark>调用方式</mark> 。
crypto.provider.kms.accessKeyId	访问KMS的AccessKey ID。
crypto.provider.kms.accessKeySecret	访问KMS的AccessKey Secret。

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 在右上角选择操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击**确定**。

ⅳ. 在确认对话框中, 单击确定。

## 使用JindoFS KeyProvider

Jindo KeyProvider负责对接KMS,加密密钥存储在KMS。KeyProvider基于KMS提供新增密钥、查询密钥和轮换密钥等功能。

• 新增密钥:传入keyldName,创建一个新的密钥。

jindo key -create -keyIdName <keyIdName>

⑦ 说明 本文示例中的<keyIdName>为您创建的密钥名称。

#### 例如,执行以下命令新增policy\_test的密钥。

jindo key -create -keyIdName policy\_test

#### 在阿里云KMS控制台,您可以看到新增了一个名为policy\_test的密钥。

密钥管理服务控制台	创建凭据 区刷新 接入指南 ②了解凭据
用户主密钥	凭据管家最新上线全托管RDS凭据,支持定时更新RDS数据库口令,减少数据库口令泄漏风险。
凭据	名称
证书	policy_test 普通凭据

#### • 查询密钥: 查看当前存在的密钥名。

jindo key -list

#### 返回信息如下:

Listing	Keys:	
	policy_	test
	policy	test2

轮换密钥:您可以根据Key ID定期更换密钥。更新密钥后Key Version会随之发生变化,即文件在加密时,使用最新的密钥进行加密,文件在解密时使用现有文件的密钥版本进行解密。

jindo key -roll -keyIdName <keyIdName>

#### 例如,执行以下命令轮换密钥policy\_test。

jindo key -roll -keyIdName policy\_test

在阿里云KMS控制台,您可以看到密钥*policy\_test*的版本状态已经更新,之前的版本状态变成了ACSPrevious,新的版本状态为ACSCurrent。

密钥管理服务控制台	← policy_test			
用户主密钥	<b>任</b>			
凭据	汽店评1頁			
证书	凭据信息	名称	policy_test	
		ARN	acs:kms:cn-hangzh	policy_test
		加密密钥	系统托管密钥	
		描述信息	Encryption Key for JindoFS File E	<u>_</u>
	∨ 版本列表 ⑦			
	存入凭据值			
	版本号		脂	反本状态
	1613802530			ACSCurrent   状态管理
	1613801564			ACSPrevious   状态管理

## 管理JindoFS加密策略

您可以根据以下命令,设置和查看加密策略:

● 设置加密策略

jindo jfs -setCryptoPolicy -keyIdName <keyIdName> <path>

⑦ 说明 本示例的 <path> 为您访问JindoFS上文件的路径。例如*jfs://test/*。

#### ● 查看加密策略

jindo jfs -getCryptoPolicy <path>

#### 设置和查看加密策略示例如下所示:

1. 查看*jfs://test/*路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回信息显示为 {NONE} 。

2. 设置*jfs://test/*的加密策略。

jindo jfs -setCryptoPolicy -keyIdName policy\_test jfs://test/

3. 进入bigboot目录,再次查看jfs://test/路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回如下信息。

SLF4J: Class path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/b2jindosdk/3.4.0-hadoop3.1/package/b2jindosdk-3.4 .0-hadoop3.1/lib/jindo-distcp-3.4.0.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/hadoop/3.2.1-1.0.1/package/hadoop-3.2.1-1.0.1/sha re/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html#multiple bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] 21/03/12 13:52:34 WARN: Unable to load native-hadoop library for your platform... using builtin-java clas ses where applicable 21/03/12 13:52:35 INFO: Jboot log name is /var/log/bigboot/jboot-20210312-135234-12953.LOG 21/03/12 13:52:35 INFO: Write buffer size 1048576, logic block size 134217728 21/03/12 13:52:35 INFO: cmd=getFileStatus, src=jfs://test/, dst=null, size=0, parameter=null, time-in-ms= 7. version=3.4.0 21/03/12 13:52:35 INFO: cmd=getCryptoPolicy, src=jfs://test/, dst=null, size=0, parameter=, time-in-ms=2, version=3.4.0 The crypto policy of path: jfs://test/ is {cipherSuite: AES\_CTR\_NOPADDING\_256, keyIdName: policy test2, k eyIdVersion: null, edek: , iv: } 21/03/12 13:52:35 INFO: Read total statistics: oss read average <none>, cache read average <none>, read o ss percent <none>

#### 设置完成后即可正常读写该路径下的文件。

◦ 拷贝本地文件至HDFS。

hadoop fs -put test.log jfs://test/

○ 展示文件内容。

hadoop fs -cat jfs://test/test.log

# 5.3. JindoFS Cache模式

## 5.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓 存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主 要介绍JindoFS的缓存模式及其使用方式。

#### 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

## 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

#### 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

#### 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的集群管理页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smar	tData ∽	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 述			服务配置
配置搜	素 \		a	全部 smartdata-site namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

## 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。

此配置无需重启SmartData服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后, Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

## 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

服务配置			
全部   smartdata-site   namespace	client storage		
	storage.handler.threads	40	0
	storage.watermark.low.ratio	0.2	0
	storage.watermark.high.ratio	0.4	0
	storage.oss.upload.threads	20	0
参数		描述	
storage.watermark.high.ratio		表示磁盘使用量的上水位比例 用的磁盘空间到达上水位即会	J,每块数据盘的JindoFS数据目录占 ≿触发清理。默认值:0.4。
storage.watermark.low.ratio		表示使用量的下水位比例,触 JindoFS数据目录占用空间清理	患发清理后会自动清理冷数据,将 理到下水位。默认值:0.2。

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

#### 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面, 单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。

参数	参数说明
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击namespace页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。

## 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在**服务配置**区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

● 在服务配置区域的smart dat a-sit e页签, 配置以下参数。

参数	参数说明
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。
fs.oss.committer.magic.enabled	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。
	② 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。

## 5.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce (简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

## 前提条件

适用环境:ECS(EMR环境外)+Hadoop+JavaSDK。

### 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smartdata-aliyun-jfs-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

#### 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击**完成**。

#### 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击确定。

#### 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏, 单击实例与镜像 > 实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

## 步骤四:在ECS上设置环境变量

#### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP CLASSPATH=\$HADOOP CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

## 步骤五:测试免密方式访问的方法

#### 1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
        while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
    }
}
```

## 5.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

### 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

## 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 i smartdata-site <b>namespace</b> client i storage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: I true:打开AuditLog功能。 I false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中,单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

isTemporary
false

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sa	<pre>l&gt; select * fror</pre>	n audit_log_sour	ce limit 1	0;				
datetime	allowed	ugi ip	ns ci	nd src	dst perr	n date		
2020-10-	20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	0				
2020-10-	20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=j</pre>	fs://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	0				
2020-10-	20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=listFileletRequest src=j	fs://kugou/ ===
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s	dst=null	<pre>perm=root:root:</pre>	rwxr-xx	2020-10-	-20			
2020-10-	20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=j</pre>	fs://kugou/===
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root:	rwxr-xx	2020-10-	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	1 audit_log	limit 10;										
datetime	allowed	ugi ip	ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true roo	t (auth:SI	MPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	hadoop:r	wxrwx
r-x 2020	-10-20												
2020-10-20 1	0:50:11.950	true roo	t (auth:SI	MPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true roo	t (auth:SI	MPLE)	192.		kugou	getFileStatusRequest	jfs://k	-11	hadoop:h	hadoop:r	wxrwx
r-x 2020	-10-20												
2020-10-20 1	1:26:06.469	true roo	t (auth:SI	MPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true roo	t (auth:SI	MPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20												
2020-10-20 1	1:26:11.320	true roo	t (auth:S]	MPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20													
2020-10-20 1	1:26:14.368	true roo	t (auth:S]	MPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20												
2020-10-20 1	1:26:14.393	true roo	t (auth:S]	MPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20													
2020-10-20 1:	1:26:16.230	true roo	t (auth:SI	MPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20												
2020-10-20 1	1:26:16.255	true roo	t (auth:SI	MPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20													
Time taken:	1.918 second	s. Fetched	10  row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

## 5.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

## 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

## ↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后, MapReduce Application Master或Spark Driver执行最终的Job Commit操作时, 会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

#### 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.jindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

置**mapreduce.outputcommitter.factory.scheme.oss**为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > SmartData。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。

- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

#### 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

## 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

### 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。

```
ii. 单击配置页签。
```

- iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

```
默认值为8。
```

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## 5.3.5. JindoFS OSS Credential Provider使用说明

Smart dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

## 配置JindoFS OSS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入smartdata-site页面。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击smartdata-site页签。
- 3. 在smartdata-site页签,根据配置方式修改或新增配置信息。

配置方式	描述
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数fs.jfs.cache.oss.credentials.provider,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如, com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。

配置方式	描述
	<ul> <li>新增配置信息:</li> <li>i. 在smartdata-site页签,单击右上角的自定义配置。</li> <li>ii. 在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。</li> <li>例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> </ul>
按照Bucket配置	<ul> <li>⑦ 说明 XXX为OSS的Bucket名称。</li> <li>iii. 单击确定。</li> </ul>

### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

## 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。

## E-MapReduce

类型	描述
Environment VariableCredent ialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

## 按照Bucket配置

## 您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken (临时安全令 牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment VariableCredent ialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>

类型	描述
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

## 5.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

## 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息(Overview)

```
包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。
```

Overview				
Start Time:	Fri Oct 16 12:29:24 2020			
Status:	Active			
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)			
Node:	Live Nodes: [2], Decommission Nodes: [0]			
Version:	3.0.0			
Build No:	fa0ea608a42			

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)				
Namespace:	jfs://test/			
Namespaces:	test			
Mode:	BLOCK_MODE			
Backend URI:	oss://			
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27			

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

## Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

## 5.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

## 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



## 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述			服务配置
配置搜	素			全部 smartdata-site namespace client storage
请输入	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

#### 5. 重启配置。

- i. 单击右上角的操作 > 重启 Jindo Namespace Service。
- ii. 输入执行原因, 单击**确定**。
- 开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

## 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger UI添加HDFS service。

Ranger	Access Manager	🗅 Audit	O Settings				💡 admin
Service Ma	anager						
Service Ma	nager						🛛 Import
			<b>—</b> ———————————————————————————————————			-	
B	HDFS		+ 🛛 🖸	B HBASE	+ 22	I HIVE	+ 2 2
				emr-hbase	<ul> <li>Image: Image: Ima</li></ul>	emr-hive	• 7 8
				emr-hbase		emr-hive	

iii. 配置相关参数。

参数	描述		
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。		
Username	自定义。		
Password	自定义。		
Namenode URL	输入jfs://{namespace_name}/。		
Authorization Enabled	使用默认值No。		
Authentication Type	使用默认值Simple。		
dfs.datanode.kerberos.principal			
dfs.namenode.kerberos.principal	了地名		
dfs.secondary.namenode.kerberos.principal	د بهرم و د ا		
Add New Configurations			

iv. 单击Add。

## 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

# 5.4. JindoTable

## 5.4.1. 开启native查询加速

JindoTable通过Native Engine,支持对Spark、Hive或Presto上ORC或Parquet格式文件进行加速。本文为您介绍如何开启native 查询加速,以提升Spark、Hive和Presto的性能。

## 前提条件

已创建集群,且ORC或Parquet文件已存放至JindoFS或OSS,创建集群详情,请参见创建集群。

### 使用限制

- 不支持对Binary类型文件进行加速。
- 不支持分区列的值存储在文件中的分区表。
- 不支持EMR-5.X系列及后续版本的E-MapReduce集群。
- 不支持代码spark.read.schema (userDefinedSchema)。
- 支持Date类型区间为1400-01-01到9999-12-31。
- 同一个表中查询列不支持区分大小写。例如, NAME和name两个列在同一个表中无法使用查询加速。
- Spark、Hive和Presto服务支持的引擎和存储格式如下所示。

引擎	ORC	Parquet
Spark2	支持	支持
Spark3	支持	支持
Presto	支持	支持
Hive2	不支持	支持
Hive3	不支持	支持

• Spark、Hive和Presto服务支持的引擎和存储文件系统如下所示。

引擎	OSS	JFS	HDFS
Spark2	支持	支持	支持
Presto	支持	支持	支持
Hive2	支持	支持	不支持
Hive3	支持	支持	不支持

## 提升Spark性能

1. 开启JindoTable ORC或Parquet加速。

? 说明

- 因为查询加速使用的是堆外内存,所以在Spark任务中建议添加配置 --conf spark.executor.memoryOverhead=
   4g ,提高Spark申请额外资源用来进行加速。
- Spark读取ORC或Parquet时,需要使用DataFrame API或者Spark-SQL。

- 。 全局设置
  - a. 进入详情页面。
    - a. 登录阿里云E-MapReduce控制台。
    - b. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
    - c. 单击上方的集群管理页签。
    - d. 在集群管理页面,单击相应集群所在行的详情。
  - b. 修改配置。
    - a. 在左侧导航栏,选择**集群服务 > Spark**。
    - b. 在Spark服务页面,单击配置页签。
    - c. 在搜索区域,搜索参数spark.sql.extensions,修改参数值 为io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtension。
  - c. 保存配置。
    - a. 单击保存。
    - b. 在确认修改对话框中, 输入执行原因, 单击确定。
  - d. 重启ThriftServer。
    - a. 在右上角选择操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框中, 输入执行原因, 单击确定。
    - c. 在确认对话框中, 单击确定。
- Job级别设置

使用spark-shell或者spark-sql时,可以添加Spark的启动参数。

作业详情请参见Spark Shell作业配置或Spark SQL作业配置。

- 2. 检查开启情况。
  - i. 登录Spark History Server UI页面。 登录详情请参见访问链接与端口。



## 提升Presto性能

↓ 注意 Presto查询并发较高,且查询加速使用堆外内存,因此使用查询加速时内存配置必须大于10 GB。

因为Presto已经内置JindoTable native加速的 catalog: hive-acc ,所以您可以直接使用 catalog: hive-acc 来启用查询加 速。

示例如下。

presto --server emr-header-1:9090 --catalog hive-acc --schema default

⑦ 说明 目前使用Presto查询加速功能时,暂不支持读取复杂的数据类型,例如Map、Struct或Array。

## 提升Hive性能

↓ 注意 如果您对作业稳定性要求较高时,建议不要开启native查询加速。

您可以通过以下两种方式提升Hive性能:

• 控制台方式

在控制台Hive服务的配置页面,搜索并修改自定义参数hive.jindotable.native.enabled为true,保存配置后,重启服务使 配置生效,此方式适用于Hive on MR和Hive on Tez。

< 返回 🛛 🍇 Hive ➤ 🔍 正常		查看操作历史
状态 部署拓扑 配置 配置修改历史 元数据		
配置过滤	服务配置	
配置搜索 hive.jindotable.native.enabled 0 Q	全部 hive-site	
配置范围	hive,jindotable.native.enabled true	\$ Ø

• 命令行方式

您可以直接在命令行中设置 hive.jindotable.native.enabled 为 true 来启用查询加速。因为EMR-3.35.0及后续版本已 经内置JindoTable Parquet加速的插件,所以您可以直接设置该参数。

set hive.jindotable.native.enabled=true;

⑦ 说明 目前使用Hive查询加速功能时,暂不支持读取复杂的数据类型,例如Map、Struct或Array。

## 5.4.2. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

## 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partitionCol2=2,... 的格式。

#### -accessStat

● 语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

```
jindo table -accessStat -d 7 -n 20
```

#### -cache

● 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

```
表或分区的路径需要位于OSS或jindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。
```

• 示例: 缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

#### -uncache

● 语法

jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示删除集群本地磁盘上指定表或分区的缓存数据。

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partition Col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

- -archive
- 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

- -unarchive
- 语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName> [-p] <partitionSpec>

- 功能
  - 表示将归档数据转为标准存储。

-o 将归档数据转为解冻, -i 将归档数据转为低频。

示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

```
jindo table -unarchive -i -t db1.t2
```

- -status
- 语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

。 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

#### -optimize

● 语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

## -showTable

- 语法
  - jindo table -showTable {-t} <dbName.tableName>
- 功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例:展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

```
-showPartition
```

● 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

jindo table -showPartition -t db1.t1 -p date=2020-10-12

- -listTables
- 语法

jindo table -listTables [-db] [dbName]

功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

• 示例:

#### ◦ 展示default库中的表。

jindo table -listTables

◦列出数据库db1中的表。

```
jindo table -listTables -db db1
```

## -dumpmc

● 语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} table} <tablename> {-p} <partitionSpec> {-f} <csv tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络Tunnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-p	分区信息 。例如 pt=xxx ,多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

功能

- 表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。
- 示例:
  - Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord
```

○ Dumpmc Maxcompute表(CSV格式)至OSS存储。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv

## 5.4.3. JindoTable SDK模式归档和解冻命令介绍

JindoTable SDK模式提供archiveTable和unarchiveTable命令,可以在不依赖Jindo Namespace Service的情况下进行归档和解冻 等操作。本文为您介绍archiveTable和unarchiveTable命令的使用方法。

## 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

• 待归档的数据必须是表数据(可以是分区表或非分区表),且已经位于阿里云对象存储OSS。

### 背景信息

JindoTable原有archive和unarchive命令可以对OSS上的表或分区进行归档或解冻等操作,但archive和unarchive命令依赖 Smart Dat a组件Jindo Namespace Service。现在新增的archiveTable和unarchiveTable命令,可以在不依赖Jindo Namespace Service的情况下进行归档和解冻等操作。

新增的archiveTable和unarchiveTable命令与原有archive和unarchive命令的主要区别为:

- 可以在未部署SmartData服务的集群上执行。例如,非EMR的用户自建集群。
- 可以通过传入过滤参数,一次应用于大量分区,多线程执行。如果本地多线程仍不能满足需求,还可以启动MapReduce任务 在整个集群上执行。

原有archive和unarchive命令的详细信息,请参见JindoTable使用说明。

### 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持新增的archiveTable和unarchiveTable命令。

## archiveTable命令

archiveTable命令可以对OSS上的表或分区进行归档。

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令, 获取帮助信息。

jindo table -help archiveTable

返回如下信息。

<dbname.tablename></dbname.tablename>	The table to archive.
-a/-i	storage policy, -a for Archive and -i for IA
	(Infrequent Access).
<condition>/-fullTable</condition>	A filter condition to determine which partitions should
	be archived, supporting common operators (like '>'),
	while -fullTable means that all partitions (or a whole
	un-partitioned table) should be archived. One but only
	one option must be specified among -c " <condition>" and</condition>
	-fullTable.
<before days=""></before>	Optional, saying that table/partitions should be
	archived only when they are created (not updated or
	modified) more than some days before from now.
<parallelism></parallelism>	The maximum concurrency when archiving partitions, 1 by
	default.
-mr/-mapReduce	Archive table/partitions using cluster-level MapReduce
	job instead of local-level multi-thread.
-e/-explain	If present, the command would not really archive data,
	but only prints the table/partitions that would be
	archived for given conditions.
<working director<="" td=""><td>y&gt;: A directory to locate map-reduce temp files. Must not be</td></working>	y>: A directory to locate map-reduce temp files. Must not be
local file system direc	tory. 'hdfs:///tmp/ <current user="">/jindotable-policy/' by</current>
default.	
<log directory=""> A dire</log>	ctory to locate log files, '/tmp/ <current user="">/' by</current>
dofaul	+

archiveTable命令语句格式如下所示。

```
-archiveTable -t <dbName.tableName> \
-a/-i \
[-c "<condition>" | -fullTable] \
[-b/-before <before days>] \
[-p/-parallel <parallelism>] \
[-mr/-mapReduce] \
[-e/-explain] \
[-w/-workingDir <working directory>] \
[-1/-logDir <log directory>]
```

参数	描述	是否必选参数
-t <dbname.tablename></dbname.tablename>	待归档的表名称,格式为 数据库名.表名 。 数据库和表名之间以半角句号(.)分隔。表可以是分区表或非分 区表。	是
-a/-i	目标存储方式。支持如下方式: •: 归档(Archive)存储。 •: 低频(Infrequent Access,IA)存储。 如果使用-i即表示低频存储,会跳过已经处于归档存储的文件。	是
-c " <condition>"   -fullTable</condition>	-fullTable 和 -c " <condition>" 只需提供一个,即要         么指定 -c "<condition>" ,要么指定 -fullTable 。         • 指定 -fullTable 时,则为移动整表,既可以是非分区表也可以是分区表。         • 指定 -c "<condition>" 时,则提供了一个过滤条件,用         来选择希望移动的分区,支持常见运算符,例如大于号(&gt;)。         例如,数据类型为String的分区ds,希望分区名大于'd',则代码为 -c " ds &gt; 'd' " 。</condition></condition></condition>	否
-b/before <before days=""></before>	只有创建时间距离现在超过一定天数的表或分区才会被归档。	否
-p/-parallel <parallelism></parallelism>	归档操作的并行度。	否
-mr/-mapReduce	使用Hadoop MapReduce而非本地多线程来归档数据。	否
-e/-explain	如果出现该选项,则为解释(explain)模式,只会显示待移动的 分区列表,而不会真正移动数据。	否
-w/-workingDir	只在MapReduce作业时使用,为MapReduce作业的工作目录。必 须有读写权限,工作目录可以非空(作业执行过程中会创建临时 文件,执行完毕会清理临时文件)。	否
-l/-logDir <log directory=""></log>	指定Log文件的目录。	否

## unarchiveTable命令

unarchiveTable命令与archiveTable命令格式基本一致,但效果相反。unarchiveTable命令可以对OSS上的表或分区进行解冻。

- 1. 通过SSH方式登录集群,详情请参见<del>登录集群</del>。
- 2. 执行以下命令,获取帮助信息。

jindo table -help unarchiveTable

返回如下信息。
<dbname.tablename></dbname.tablename>	The table to unarchive.
-i	unarchive to IA (Infrequent Access).
-0	restore to make archived data accessible temporarily.
<condition>/-fullTable</condition>	A filter condition to determine which partitions should
	be unarchived, supporting common operators (like '>'),
	while -fullTable means that all partitions (or a whole
	un-partitioned table) should be unarchived. One but
	only one option must be specified among -c
	" <condition>" and -fullTable.</condition>
<before days=""></before>	Optional, saying that table/partitions should be
	unarchived only when they are created (not updated or
	modified) more than some days before from now.
<parallelism></parallelism>	The maximum concurrency when unarchiving partitions, 1
	by default.
-mr/-mapReduce	Unarchive table/partitions using cluster-level
	MapReduce job instead of local-level multi-thread.
-e/-explain	If present, the command would not really unarchive
	data, but only prints the table/partitions that would
	be unarchived for given conditions.
<working director<="" td=""><td>y&gt;: A directory to locate map-reduce temp files. Must not be a</td></working>	y>: A directory to locate map-reduce temp files. Must not be a
local file system direc	tory. 'hdfs:///tmp/ <current user="">/jindotable-policy/' by</current>
default.	
<log directory=""> A dire</log>	ectory to locate log files, '/tmp/ <current user="">/' by</current>
defaul	t.

#### unarchiveTable命令语句格式如下所示。

```
-unarchiveTable -t <dbName.tableName> \
[-i/-0] \
[-c "<condition>" | -fullTable] \
[-b/-before <before days>] \
[-p/-parallel <parallelism>] \
[-mr/-mapReduce] \
[-e/-explain] \
[-w/-workingDir <working directory>] \
[-1/-logDir <log directory>]
```

unarchiveTable命令与archiveTable命令参数只有一处区别,即没有必选参数-a/-i,而被可选参数-i/-o替代。

可选参数-i/-o描述如下:

- 如果不指定-i/-o参数,则转换存储格式为标准(Standard)存储。
- 如果指定-i参数,则转换存储格式为低频(Infrequent Access, IA)存储,原本为标准存储的文件被跳过。
- 如果指定-o参数,则仅做解冻(Restore)操作。原本为标准存储或低频存储的文件均被跳过。已经处于解冻状态的文件也会 被跳过,即不会重复解冻。

# 5.4.4. JindoTable MoveTo命令介绍

MoveTo命令可以实现表和分区数据的迁移功能。本文为您介绍MoveTo命令的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

# 背景信息

MoveTo命令可以在拷贝底层数据结束后,自动更新元数据,使表和分区的数据完整地迁移到新路径;可以通过条件筛选,一次 拷贝大量分区。在数据迁移过程中,还使用了多种措施保护数据的完整性,确保数据安全。

# 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持使用MoveTo命令。

# 使用MoveTo命令

○ 注意 集群上每次仅允许运行一个MoveTo进程。如果集群上有正在运行的MoveTo进程,启动新的MoveTo进程时会因为获取不到配置锁而退出,并告知正在运行的MoveTo进程。此时,您可以终止掉正在运行的MoveTo进程,启动新的MoveTo进程,或者等待正在运行的MoveTo进程结束。

#### 1. 通过SSH方式登录集群,详情请参见登录集群。

#### 2. 执行以下命令, 获取帮助信息。

jindo table -help moveTo

#### 帮助信息类似如下所示。

<dbname.tablename< th=""><th>&gt; The table to move.</th></dbname.tablename<>	> The table to move.
<destination path:<="" td=""><td>&gt; The destination base directory which is always at the</td></destination>	> The destination base directory which is always at the
	same level of a 'table location', where the moved
	partitions or un-partitioned data would located in.
<condition>/-full</condition>	Table A filter condition to determine which partitions should
	be moved, supporting common operators (like '>') and
	built-in UDFs (like to date) (UDFs not supported
	yet), while -fullTable means that all partitions (or
	a whole un-partitioned table) should be moved. One but
	only one option must be specified among -c
	" <condition>" and -fullTable.</condition>
<before days=""></before>	Optional, saying that table/partitions should be moved
-	only when they are created (not updated or modified)
	more than some days before from now.
<parallelism></parallelism>	The maximum concurrency when copying partitions, 1 by
-	default.
<oss store<="" td=""><td>age policy&gt;: Storage policy for OSS destination, which can be Standar</td></oss>	age policy>: Storage policy for OSS destination, which can be Standar
(by default), I	A, Archive, or ColdArchive. Not applicable for destinations other
than OSS. NOTE:	if you are willing to use ColdArchive storage policy, please
make sure that (	Cold Archive has been enabled for your OSS bucket.
-o/-overWrite	Overwriting the final paths where the data would be moved.
	For partitioned tables this overwrites partitions' locations
	which are subdirectories of <destination path="">; for</destination>
	un-partitioned table this overwrites the <destination path=""></destination>
	itself.
-r/-removeSource	Let the source data be removed when the corresponding
	table/partition is successfully moved to the new destination.
	Otherwise (by default), the source data would be left as it
	was.
-skipTrash	Applicable only when [-r/-removeSource] is enabled. If
1	present, source data would be immediately deleted from the
	file system, bypassing the trash.
-e/-explain	If present, the command would not really move data, but only
-, - 1 -	prints the table/partitions that would be moved for given
	conditions.
<log directorv=""></log>	A directory to locate log files, '/tmp/ <current user="">/' by</current>
. yy,	default.

MoveTo命令语句如下所示。

jindo table -moveTo \
 -t <dbName.tableName> \
 -d <destination path> \
 [-c "<condition>" | -fullTable] \
 [-b/-before <before days>] \
 [-p/-parallel <parallelism>] \
 [-s/-storagePolicy <OSS storage policy>] \
 [-o/-overWrite] \
 [-r/-removeSource] \
 [-skipTrash] \
 [-e/-explain] \
 [-1/-logDir <log directory>]

参数	描述	是否必选参数
-t <dbname.tablename></dbname.tablename>	待移动的表名称,格式为 数据库名.表名 。 数据库和表名之前以半角句号(.)分隔。表可以是分区表或非分 区表。	是
-d <destination path=""></destination>	待移动的目标位置。无论是移动分区还是移动非分区表的整表, 该位置都对应 "表" 一级的位置。如果移动的是分区,则分区的完 整路径是该路径+分区名。例如 <destination path="">/p1=v1/ p2=v2/ 。</destination>	是
-c " <condition>"   -fullTable</condition>	<ul> <li>两者必须且只能提供一个,即要么指定 -c "</li> <li><condition>",要么指定 -fullTable 。</condition></li> <li>指定 -fullTable 时,则为移动整表,既可以是非分区表也可以是分区表。</li> <li>指定 -c "<condition>"时,则提供了一个过滤条件,用来选择希望移动的分区,支持常见运算符,例如大于号(&gt;)。</condition></li> <li>例如,数据类型为String的分区ds,希望分区名大于'd',则代码为 -c " ds &gt; 'd' "。</li> </ul>	否
-b/before <before days=""></before>	仅创建时间距离现在超过一定天数的表或者分区才会被移动。	否
-p/-parallel <parallelism></parallelism>	迁移操作的并行度。	否
-s/-storagePolicy <oss storage<br="">policy&gt;</oss>	<ul> <li>拷贝到OSS时,在OSS上的存储策略。存储策略如下:</li> <li>Standard:归档存储。</li> <li>IA:低频 (Infrequent Access)存储。</li> <li>Archive:标准存储。</li> <li>ColdArchive:冷归档存储。</li> <li>⑦ 说明 使用前请确保OSS Bucket开通了该功能。</li> </ul>	否
-o/-overWrite	是否强制覆盖目标写入路径。如果是分区表,则只会清空待移动 分区的分区路径,不会清空整个表路径。	否
-r/-removeSource	移动完成,元数据也同步更新后,是否清理源路径。如果是分区 表,则只会清理成功移动的分区的源路径。	否
-skipT rash	清理源路径时是否跳过Trash。 ⑦ 说明 在指定了参数-r/-removeSource时适用。	否

参数	描述	是否必选参数
-e/-explain	如果出现该选项,则为解释(explain )模式,只会显示待移动的 分区列表,而不会真正移动数据。	否
-l/-logDir <log directory=""></log>	指定Log文件目录。	否

# 配置锁目录

MoveTo工具实现了进程锁,需要提供一个HDFS的路径放置锁文件。默认情况下,该路径为hdfs:///tmp/jindotable-lock/。

↓ 注意 放置锁文件的路径只能是HDFS路径。如果您对该路径无操作权限时,可以按照如下步骤添加自定义配置,配置 该路径。

- 1. 进入HDFS服务页面。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > HDFS。
- 2. 修改配置。
  - i. 在HDFS服务的配置页面,单击hdfs-site或core-site页签。
  - ii. 单击右上角的自定义配置。

< 返回 🖉 HDFS 🗸 🌢 正常		© 查看操作历史 d	快捷链接 💙 🤌 操作 💙
状态 部署拓扑 配置 配置修改历史			
配置过速 配置接索	服务配置 全部 <mark>hdfs-site</mark> kms-site httpfs-site   hadoop-env   httpfs-env   ssl-client core-site		部署客户洪配置保存 自定义配置
<ul> <li>请输入</li> <li>Q</li> <li>配置范围</li> </ul>	dfs.datanode.cache.revocation.timeout.ms 900000	0	

iii. 在**新增配置项**对话框中,添加配置项jindotable.moveto.tablelock.base.dir,参数值为一个已存在的HDFS路径。

○ 注意 自定义配置锁目录时,请确保整个集群的所有节点上不存在正在运行的MoveTo进程,否则可能导致 MoveTo执行失败,甚至导致数据污染。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。

# 5.4.5. JindoTable表或分区访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

### 前提条件

已创建集群,详情请参见创建集群。

# 背景信息

JindoTable支持收集访问Hive表的记录,收集的数据保存在SmartData服务的Namespace中。

Smart Dat a 3.2.x版本开始支持Spark、Hive和Presto引擎,Spark和Presto的数据收集默认是打开的,如果需要关闭,请参见关闭 <mark>热度收集</mark>。Hive的数据收集默认是关闭的,如果需要打开,请参见<mark>开启Hive热度收集</mark>。

#### 数据查询

JindoTable提供了命令方式查询热度信息。

● 语法

<sup>&</sup>gt; 文档版本: 20220629

jindo table -accessStat <-d [days]> <-n [topNums]>

days和 topNums为正整数。当只设置天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

JindoTable使用详情,请参见JindoTable使用说明。

#### 开启Hive热度收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改Hive的参数值。
  - i. 在左侧导航栏,选择**集群服务 > Hive**。
  - ii. 在Hive服务页面,单击配置页签。
  - iii. 搜索参数hive.exec.post.hooks,在参数值后追加com.aliyun.emr.table.hive.HivePostHook。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 7. 重启服务。
  - i. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
  - ii. 在执行集群操作对话框,输入执行原因。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 关闭热度收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。
  - Hive服务:
    - a. 在左侧导航栏,选择集群服务 > Hive。
    - b. 在Hive服务页面,单击配置页签。
    - c. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤 配置搜索	服务配置 全部   hive-site
	hive.exec.post.hooks kHiveHook com.aliyun.emr.table.hive.HivePostHook

- 。 Spark服务:
  - a. 在左侧导航栏,选择集群服务 > Spark。

- b. 在Spark服务页面,单击**配置**页签。
- c. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSQLQueryListener。

状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置
配置搜索	全部   spark-defaults
spark.sql.queryExecutionListeners	spark.sql.queryExecutionListeners comaliyun.emr.table.spark.SparkSQLQueryListener

○ Presto服务:

- a. 在左侧导航栏,选择集群服务 > Presto。
- b. 在Presto服务页面, 单击配置页签。

c. 搜索参数event-listener.name, 删除参数值中的内容。

- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 7. 重启服务。
  - Hive服务:
    - a. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - o Spark服务:
    - a. 在Spark服务页面,选择右上角的操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Presto服务:
    - a. 在Presto服务页面,选择右上角的操作 > 重启All Components。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。

# 5.4.6. JindoTable表或分区访问冷度收集

JindoTable表或分区的访问冷度收集功能可以为您维护表或分区上次的访问时间,从而筛选出最近没有被访问的数据,帮助您优 化数据存储方式,节约成本。例如,在数据分析中,您可以把部分不常用的分区数据移动到成本更低的存储介质以节约成本。

#### 前提条件

已创建EMR-3.35.0及后续版本或EMR-4.9.0及后续版本,创建详情请参见创建集群。

#### 背景信息

Smart Dat a 3.5.x版本开始支持Hive、Spark和Presto组件的冷度收集功能。该功能目前默认不开启,如果需要开启,请参见<mark>开启</mark> Spark冷度收集、开启Hive冷度收集和开启Presto冷度收集。

② 说明 因为冷度收集与热度收集使用相同的hooks或Listeners,所以开启组件的冷度收集时会同时打开热度收集功能。 表或分区访问热度收集的详情,请参见jindoTable表或分区访问热度收集。

### 使用限制

- 不支持DLF数据湖元数据。
- Hive CLI、HiveServer2、Spark SQL CLI、Spark Thriftserver和Presto服务所在IP需要有权限访问元数据底层存储(MySQL或 RDS)。
- 仅支持Hive、Spark和Presto组件的冷度收集。

#### 数据查询

JindoTable提供了命令方式查询冷度信息。

● 语法

jindo table -leastUseStat -n <num> [-i/-ignoreNever]

num是显示的条目数量,应为正整数。-i/-ignoreNever为可选参数,如果设置该参数,则会过滤掉从未被访问过的表或分区。

功能

展示最久未被访问的表或分区。

• 示例:查询最久未被访问的表或分区的20条记录。

jindo table -leastUseStat -n 20

返回如下图所示三列结果。

tdb.t1	pid=20/qid=ten	2021-03-26	13:53:52
tdb.t2		2021-03-26	13:53:58

- 第一列为表的名字,格式:数据库名.表名。
- 。 第二列为分区名字,格式:第一分区列=列值/第二分区列=列值/...,如果表为非分区表则为空。
- 第三列为最近一次访问的时间,格式: yyyy-MM-dd HH:mm:ss。

⑦ 说明 如果为分区表,则只显示到分区级别,表本身不会单独显示。

JindoTable使用详情,请参见JindoTable使用说明。

#### 开启Spark冷度收集

- 1. 进入Spark页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Spark。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.queryExecutionListeners,确保参数值包含com.aliyun.emr.table.spark.SparkSQLQueryListener,如果存 在多个Listeners时使用英文逗号(,)隔开。

<返回 🖍 Spark 🗸 • 正常		O 查看操作
状态 部署拓扑 配置 配置修改历史		
配置过滤	服务配置	
配置搜索	全部 I spark-defaults	
spark.sql.queryExecutionListeners 0 0	spark.sql.queryExecutionListeners com.aliyum.emr.table.spark.SparkSQLQueryLis	tener,c 🕐

- 4. 添加自定义配置。
  - i. 在服务配置页面, 单击spark-defaults页签。
  - ii. 单击右上角的自定义配置。

iii. 在新增配置项对话框中,设置Key为spark.sql.query.update.access.time.enabled, Value为true。

新增配置项 ×			
* Key	* Value	描述	攝作
spark.sql.query.update.access.time.enabled	true		删除
汤加			
			确定 取消

- iv. 单击**确定**。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

### 开启Hive冷度收集

- 1. 进入Hive页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Hive。
- 2. 在Hive服务页面,单击配置页签。
- 3. 搜索参数hive.exec.post.hooks,确保参数值包含com.aliyun.emr.table.hive.HivePostHook,如果存在多个hooks时使用英文逗号(,)隔开。

<返回 < 🎬 Hive → ●正常	
状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤	服务配置
	全部   hive-site
nive.exec.post.nooks	hive ever nost books rom alivius emmeta hive book Lineanel oggerHook
配置范围	inverse comony one many one of the other of

4. 添加自定义配置。

新增配罢顶

- i. 在服务配置页面, 单击hive-site页签。
- ii. 单击右上角的自定义配置。
- iii. 在新增配置项对话框中,设置Key为hive.hook.update.access.time.enabled, Value为true。

MARKED X			~
* Key	* Value	描述	操作
hive.hook.update.access.time.enabled	true		删除
添加			
			确定 取消

- i∨. 单击**确定**。
- 5. 保存配置。

- i. 单击保存。
- ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

# 开启Presto冷度收集

- 1. 进入Presto页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Presto。
- 2. 在Presto服务页面,单击配置页签。
- 3. 搜索参数event-listener.name,确保参数值包含jindo-presto-post-event-listener。

<返回 Presto ≻ ●正常	
状态 部署拓扑 配置 配置修改历史	
配置过速 配置搜索	服务配置 全部 event-listener.properties
event-listener.name © Q 配置范围	event-listener.name jindo-presto-post-event-listener

- 4. 添加自定义配置。
  - i. 在服务配置页面, 单击event-listener.properties页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为listener.update.access.time.enabled, Value为true。

岃	新門自選項 >>			
	* Key	* Value	描述	操作
	listener.update.access.time.enabled	true		删除
	添加			
				确定 取消

iv. 单击**确定**。

- 5. 保存配置。
  - i. 单击保存。

ii. 在确认修改对话框中, 输入执行原因, 单击确定。

- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

# 5.5. 工具集

# 5.5.1. Jindo sql命令介绍

Jindo sql命令是JindoFS自带的工具,方便您分析JindoFS访问日志、元数据和OSS访问日志。本文为您介绍如何使用Jindo sql命 令,分析JindoFS访问日志、元数据和OSS访问日志的数据。

# 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

# 背景信息

您可以使用Jindo sql命令分析以下数据:

- 使用Jindo sql分析JindoFS访问日志
- 使用Jindo sql分析元数据
- 使用Jindo sql分析OSS访问日志

## 使用限制

EMR-3.36.0及后续版本或EMR-5.2.0及后续版本的集群,支持使用Jindo sql命令。

# 使用Jindo sql命令

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,启动jindo sql。

jindo sql

jindo sql支持以下常用参数。

参数	描述
-f	指定运行的SQL文件。
-i	启动Jindo sql后自动运行初始化SQL脚本。
-d	参数设置为键值对的形式。例如, -d A=B 。

# Jindo sql内置表结构

• audit\_log\_source (分区表)

audit\_log\_source表用作JindoFS访问日志原始表。

参数	描述
datetime	时间格式yyyy-MM-dd HH:mm:ss。
allowed	本次操作是否被允许,取值如下: o true:允许本次操作。 o false:不允许本次操作。
ugi	操作用户(包含认证方式信息)。
ip	Client IP地址。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dst	目标路径,可以为空。
perm	操作文件的Permission信息。

参数	描述
date (分区列)	日志日期,格式为YYYY-mm-DD。

# audit\_log

# audit\_log允许使用分区列进行分区过滤,用作JindoFS访问日志表。

参数	描述
datetime	时间格式yyyy-MM-dd HH:mm:ss。
allowed	本次操作是否被允许,取值如下: • true:允许本次操作。 • false:不允许本次操作。
ugi	操作用户(包含认证方式信息)。
ip	Client IP地址。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dst	目标路径,可以为空。
perm	操作文件的Permission信息。
date (分区列)	日志日期,格式为YYYY-mm-DD。

# ● fs\_image(分区表)

# fs\_image用作转存image信息

参数	描述
atime	INode最近访问时间。
attr	文件相关属性。
etag	OSS的ETag值。
id	Node的D。
mtime	INode的修改时间。
name	INode的名称。
owner	owner名称。
ownerGroup	owner组名称。
parentId	父节点的ID。
permission	操作文件的Permission信息。
size	INode的大小。
state	INode的状态。
type	INode的类型。

参数	描述
storagePolicy	存储策略。
namespace (分区列)	namespace名称。
datetime (分区列)	转存时间。

#### oss\_access\_log\_source

如果开启分区表模式,则为分区表。oss\_access\_log\_source表用作OSS访问日志原始表。

参数	描述
line	原始日志。
bucket(分区列)	Bucket名称。
partition_date (分区列)	日志日期格式为YYYY-mm-DD。

# oss\_access\_log

如果开启分区表模式,允许使用分区列进行分区过滤。oss\_access\_log表用作OSS访问日志。

参数	描述
Remote_IP	请求者的IP地址。
Reserved	保留字段,固定值为-。
Reserved1	保留字段,固定值为-。
Time	OSS收到请求的时间。
Request_URI	包含query string的请求URL。OSS会忽略以x-开头的query string参数,但这个参数会被记录在访问日志中。所以您可以使用x-开头query string参数标记一个请求,然后使用这个标记快速查找该请求对应的日志。
HTTP_Status	OSS返回的HTTP状态码。
SentBytes	请求产生的下行流量。单位:Byte。
RequestTime	完成本次请求耗费的时间。单位:ms。
Referer	请求的HTTP Referer。
User_Agent	HTTP的User-Agent头。
HostName	请求访问的目标域名。
Request_ID	请求的Request ID。
LoggingFlag	是否已开启日志转存。
Requester	请求者的用户ID。取值-表示匿名访问。
Operation	请求类型。
Bucket	请求的目标Bucket名称。
Кеу	请求的目标Object名称。
Object Size	目标Object大小。单位:Byte。

参数	描述
Server_Cost_Time	OSS处理本次请求所花的时间。单位:毫秒。
ErrorCode	OSS返回的错误码。取值-表示未返回错误码。
RequestLength	请求的长度。单位: Byte。
UserlD	Bucket拥有者ID。
Delta_DataSize	Bucket大小的变化量。取值-表示此次请求不涉及Object的写入操作。
SyncRequest	请求是否为CDN回源请求。取值如下: • cdn: 请求是CDN回源请求。 • -: 请求不是CDN回源请求。
StorageClass	<ul> <li>目标Object的存储类型。取值如下:</li> <li>Standard:标准存储。</li> <li>IA:低频访问存储。</li> <li>Archive:归档存储。</li> <li>Cold Archive:冷归档存储。</li> <li>-:未获取Object存储类型。</li> </ul>
TargetStorageClass	是否通过生命周期规则或CopyObject转换了Object的存储类型。取值如下: • Standard:转换为标准存储。 • IA:转换为低频访问存储。 • Archive:转换为归档存储。 • Cold Archive:转换为冷归档存储 • -:请求不涉及Object存储类型转换操作。
TransmissionAccelerationAccessPoint	通过传输加速域名访问目标Bucket时使用的传输加速接入点。取值-表示未使用传 输加速域名或传输加速接入点与目标Bucket所在地域相同。 例如,请求者通过华东1(杭州)的接入点访问目标Bucket时,值为cn- hangzhou。
AccessKeyID	访问的AccessKey ID。
bucket(分区列)	Bucket名称。
partition_date (分区列)	日志日期格式为YYYY-mm-DD。

# 使用Jindo sql分析JindoFS访问日志

JindoFS为存储在OSS上的JindoFS访问日志文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃IP分析。您可以通过 jindo sql 命令,使用该功能。

⑦ 说明 已开启AuditLog功能,详情请参见AuditLog使用说明。

Jindo SQL相关命令示例如下:

• 执行如下命令,显示表。

show tables;

⑦ 说明 表结构信息,请参见Jindo sql内置表结构。

返回信息如下图所示。

jindo-so	1> show t	tables;		ľ
database	e t	tableName	isTemporary	
default	audit_log	g false		
default	audit_log	_source	false	
default	fs_image	false		

• 执行如下命令,显示分区。

show partitions audit\_log\_source;

#### 返回信息类似如下图所示。

jindo-sql> show partitions audit\_log\_source; partition date=2020-10-20 Time taken: 0.031 seconds, Fetched 1 row(s)

• 执行如下命令,查询数据。

select \* from audit\_log\_source limit 10;

返回信息类似如下图所示。

jindo-sql> select * fro	m audit_log_sour	ce limit 10	ð;					
datetime allowed	ugi ip	ns cn	nd src	dst pe	erm o	date		
2020-10-20 10:50:11.924	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null perm=hadoop:had	oop:rwxrwxr-x	2020-10-20	0					
2020-10-20 10:50:11.950	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.	1813 P	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null perm=null	2020-10-20							
2020-10-20 11:26:06.445	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.	1993 P	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null perm=hadoop:had	oop:rwxrwxr-x	2020-10-20	0					
2020-10-20 11:26:06.469	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/
null perm=null	2020-10-20							
2020-10-20 11:26:11.295	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s dst=null	perm=root:root:	rwxr-xx	2020-10-	-20				
2020-10-20 11:26:11.320	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null perm=null	2020-10-20							
2020-10-20 11:26:14.368	allowed=true	ugi=root (	(auth:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=getFileStatusRequest</pre>	src=jfs: d
s dst=null	perm=root:root:	rwxr-xx	2020-10-	-20				

select \* from audit\_log limit 10;

返回信息类似如下。

jindo-sql> select * from audit_log limit 10;									
datetime allowed ugi	ip ns cmd	src dst	perm	date					
2020-10-20 10:50:11.924 true	root (auth:SIMPLE)	192.	kugou	getFileStatusRequest	jfs://k ll	hadoop:	hadoop:rwxrwx		
r-x 2020-10-20									
2020-10-20 10:50:11.950 true	root (auth:SIMPLE)	192.	kugou	listFileletRequest	jfs://k ll	null	2020-10-20		
2020-10-20 11:26:06.445 true	root (auth:SIMPLE)	192.	kugou	getFileStatusRequest	jfs://k ll	hadoop:	hadoop:rwxrwx		
r-x 2020-10-20									
2020-10-20 11:26:06.469 true	root (auth:SIMPLE)	192.	kugou	listFileletRequest	jfs://k ll	null	2020-10-20		
2020-10-20 11:26:11.295 true	root (auth:SIMPLE)	192.	kugou	getFileStatusRequest	jfs://k	null	root:root:rw		
xr-xx 2020-10-20									
2020-10-20 11:26:11.320 true	root (auth:SIMPLE)	192.	kugou	listFileletRequest	jfs://k	null	null 2020		
-10-20									
2020-10-20 11:26:14.368 true	root (auth:SIMPLE)	192.	kugou	getFileStatusRequest	jfs://k	null	root:root:rw		
xr-xx 2020-10-20									
2020-10-20 11:26:14.393 true	root (auth:SIMPLE)	192.	kugou	listFileletRequest	jfs://k	null	null 2020		
-10-20									
2020-10-20 11:26:16.230 true	root (auth:SIMPLE)	192.	kugou	getFileStatusRequest	jfs://k	null	root:root:rw		
xr-xx 2020-10-20									
2020-10-20 11:26:16.255 true	root (auth:SIMPLE)	192.	kugou	listFileletRequest	jfs://k	null	null 2020		
-10-20									
Time taken: 1 018 seconds For	tchod 10 now(c)								

• 执行如下命令,统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 使用Jindo sql分析元数据

JindoFS为JindoFS上的元数据文件提供SQL的分析功能,通过SQL分析相关表。您可以通过 jindo sql 命令,使用该功能。

⑦ 说明 已开启AuditLog功能,详情请参见AuditLog使用说明。

- 1. 通过SSH方式登录集群,详情请参见<del>登录集群</del>。
- 2. 执行以下命令,启动Jindo sql。

jindo sql

- 3. 查询Jindo SQL可以分析的表格。
  - 使用 show tables 命令,可以查看支持查询分析的表格。Jindo Sql内置了审计和元数据信息的分析功能,对应 audit\_log和fs\_image。

代码示例如下图所示。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

使用 show partitions fs\_image
 命令,可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -du
 mpMetadata 生成的数据。

代码示例如下图所示。

jindo-sql> show partitions fs_image;
partition
namespace=k /datetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)

4. 查询分析元数据信息。

Jindo SQL使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

代码示例如下图所示。

[root@em Spark ma	root@emr-worker-2 hadoop]# jindo sql park master: yarn, Application Id: app														
jindo-sa	1> show	tables;													
database		tablena	me isi	emporary											
default	audit_1	og sa sourse	raise												
default	fe image	og_sourc	e rai	26											
Time tok	in A 3	e 15 secon	de Eatchad	3 montes											
iindo-so	als sele	rt * fro	m fs image 1	imit 10:											
atime	atte	etaa	id mti	ne name	owner	ownerGr	nun	narentId	nermission	size	state	storage	Policy	type	name
space	datetim	e										o cor age			manne
0			73110760050	51899448	1603084	4070081	/tocds/	orc/5000/web		date sk	=2450819	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	or v	kugou	2020_10_20_	10_50_36						
0			16534448041	986675495	1603084	1071350	/tpcds/	orc/5000/web	_returns/wr_returned	d_date_sk	=2450820	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	ory	kugou	2020_10_20_	10_50_36						
			73110760050	51899470	1603084	4070185	/tpcds/	orc/5000/web	_returns/wr_returned	l_date_sk	-2450821	root	root	334790	833296
5855433	489		Finalized	WARM	Directe	ory	kugou	2020_10_20_	10_50_36						
			11922762023	479287249	1603084	4069581	/tpcds/	orc/5000/web	_returns/wr_returned	d_date_sk	=2450822	root	root	334790	833296
5855433	489		Finalized	WARM	Directo	ory	kugou	2020_10_20_	10_50_36						
			10769840518	872441036	1603084	4073592	/tpcds/	orc/5000/web	_returns/wr_returned	d_date_sk	=2450823	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	pry	kugou	2020_10_20_	10_50_36						
0			26993899866	24511354	1603084	1068996	/tpcds/	orc/5000/web	_returns/wr_returned	d_date_sk	=2450824	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	ory	kugou	2020_10_20_	10_50_36						
0			11922762023	479287307	160308	4069875	/tpcds/	orc/5000/web	_returns/wr_returned	l_date_sk	=2450825	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directe	ory	kugou	2020_10_20_	10_50_36						
0			15464684820	17665002	1603084	4072440	/tpcds/	orc/5000/web	_returns/wr_returned	_date_sk	=2450826	root	root	334790	833296
5855433	489	ø	Finalized	WARM	Directo	bry	kugou	2020_10_20_	10_50_36						
0			16534448041	906675460	160308	4071170	/tpcds/	orc/5000/web	_returns/wr_returned	d_date_sk	=2450827	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directo	ory	kugou	2020_10_20_	10_50_36					224200	
0	100		73110760050	51899544	1603084	10/05/2	/tpcds/	orc/5000/web	_returns/wr_returned	_date_sk	=2450828	root	root	334790	833296
5855433 Time tel	407	64	da Fatabad	10 RAKM	prrecto	bry	kugou	2020_10_20_	19_59_50						
Time tak	ten: 6.76	o4 secon	as, retched	10 row(S)											

例如:根据某次转存的元数据信息统计该namespace下的目录个数。

jindo-sql> count(1) 11837 Time taken:	select 6.852	: count(*) seconds,	from Fetche	fs_imag d 1 row	e where (s)	type =	= "Directo	ory" and	namespace	⊨"kugou	" and	datetime⊨	="2020_1	10_20_1	0_47_1	4";	
? 说即	月 r	namesp	ace利	date	et ime >	5 Jind	o Sql增	加的两	ī列,分	别对应	ī于n	amespa	ace名	称和」	上传元	こ数据的	勺时间看

# 使用Jindo sql分析OSS访问日志

↓ 注意 分析OSS访问日志需要指定OSS访问日志目录和指定是否为分区表,指定分区表会自动按照Bucket或date进行日志归档,能够支持使用过滤语句指定查询某个分区,极大的提升了查询效率,但是开启分区表之后必须每次使用分区表模式,否则文件会被归档到目录导致部分数据无法查询。

JindoFS为存储在OSS上的OSS访问日志文件提供SQL的分析功能,通过SQL分析相关表。您可以通过\_\_jindo\_sql\_\_命令,使用该 功能。

⑦ 说明 已开启日志转存,详情请参见日志转存。

- 1. 通过SSH方式登录集群,详情请参见登录集群。
- 2. 执行以下命令,启动Jindo sql。

jindo sql

3. 执行以下命令,指定日志存储路径和表类型。

```
jindo sql -d access_log_path=oss://test-sh/oss-accesslog -d partition.table.enabled=true
```

代码中的 access\_log\_path 为OSS访问日志存储路径, partition.table.enabled 指定是否为分区表, true表示为分区 表。

## 常见问题

• Q: 如何修改初始资源Jindo sql的启动参数?

A: 因为 Jindo sql 基于Spark的程序,所以初始资源可能较小,您可以通过环境变量 JINDO\_SPARK\_OPTS 来修改初始资源 Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.exec utor.cores=5 --conf spark.executor.memory=20G"

#### • Q: 如何使用Hive分析表?

A:为了避免污染Hive元数据,默认Hive看不到Default下的几个表,如果想使用Hive分析这些表,可以通过语句 show create table {table\_name} 查看表语句或者使用SQL创建新表,Hive需要执行加载外部表。

# 5.5.2. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

#### 前提条件

已创建集群,详情请参见创建集群。

#### 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

# 挂载

- ⑦ 说明 依次在每个节点上执行挂载操作。
- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

#### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

返回用户在服务端配置的所有命名空间列表。

test testcache

2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

3. 创建目录。

mkdir /mnt/jfs/test/dirl
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

### 如果您想使用Python方式写入和读取文件,请参见如下示例:

#### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

#### 2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

#### 返回如下信息。

```
my first file
This file
```

# 卸载

⑦ 说明 依次在每个节点上执行卸载操作。

```
1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
```

```
2. 执行如下命令, 卸载FUSE。
```

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 5.5.3. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

# 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

# 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见登录集群。

2. 执行以下命令, 获取帮助信息。

jindo distcp --help

#### 返回信息如下。

help - Print help text
src=VALUE - Directory to copy files from
dest=VALUE - Directory to copy files to
parallelism=VALUE - Copy task parallelism
outputManifest=VALUE - The name of the manifest file
previousManifest=VALUE - The path to an existing manifest file
requirePreviousManifest=VALUE - Require that a previous manifest is present if specified
copyFromManifest - Copy from a manifest instead of listing a directory
srcPrefixesFile=VALUE - File containing a list of source URI prefixes
srcPattern=VALUE - Include only source files matching this pattern
deleteOnSuccess - Delete input files after a successful copy
outputCodec=VALUE - Compression codec for output files
groupBy=VALUE - Pattern to group input files by
targetSize=VALUE - Target size for output files
enableBalancePlan - Enable plan copy task to make balance
enableDynamicPlan - Enable plan copy task dynamically
enableTransaction - Enable transation on Job explicitly
diff - show the difference between src and dest filelist
ossKey=VALUE - Specify your oss key if needed
ossSecret=VALUE - Specify your oss secret if needed
ossEndPoint=VALUE - Specify your oss endPoint if needed
policy=VALUE - Specify your oss storage policy
cleanUpPending - clean up the incomplete upload when distcp job finish
queue=VALUE - Specify yarn queuename if needed
bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed
s3Key=VALUE - Specify your s3 key
s3Secret=VALUE - Specify your s3 Sercet
s3EndPoint=VALUE - Specify your s3 EndPoint

# --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo DistCp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录, Jindo DistCp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket 的名称。

#### --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制*/data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定\_\_\_\_srcPattern\_的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

返回信息如下。

```
      Found 6 items

      -rw-r-----
      2 root hadoop
      2252 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/000151.sst

      -rw-r-----
      2 root hadoop
      4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/1.log

      -rw-r-----
      2 root hadoop
      4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/2.log

      -rw-r-----
      2 root hadoop
      4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001

      09
      -

      -rw-r-----
      2 root hadoop
      1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp01.txt

      -rw-r-----
      2 root hadoop
      1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp01.txt

      -rw-r-----
      2 root hadoop
      1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp01.txt
```

#### 执行以下命令,复制以log结尾的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPattern .\*\.lo
g --parallelism 20

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

# --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --deleteOnSuccess -parallelism 20

#### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下:

```
      Found 6 items

      -rw-rw-rw-1
      938 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/000151.sst.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/OPTIONS-000109.g

      z
      -rw-rw-rw-1

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz
```

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。

# --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前outputManifest文件默认且必须为gz类型 压缩文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20
```

#### 查看outputManifest文件内容。

```
hadoop fs -text oss://<yourBucketName>/hourly_table/manifest-2020-04-17.gz > before.lst
cat before.lst
```

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst", "baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table", "size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

```
{"path":"oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly_table","size":1016}
```

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

# --outputManifest和--previousManifest

```
--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件
的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录,查看运行期间复制的文
件。
```

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --outputManifest=ma
nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly_table/manifest-2020-04-17.gz --parallel
ism 20
```

#### 执行以下命令,查看运行期间复制的文件。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst

#### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"

:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly table","size":4891)

# --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
        Found 4 items

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20
```

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

```
hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly_table/2017-02-01
hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly table/2017-02-02
```

#### --groupBy和-targetSize

# 因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

#### 例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly table/2017-02-01/03

```
返回信息如下。
```

Found 8 items								
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst	
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log	
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log	
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/5.log	
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/6.log	
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001	
09								
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt	
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt	

#### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --targetSize=10 --g
roupBy='.\*/([a-z]+).\*.txt' --parallelism 20

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

# --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为Dist Cp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo Dist Cp的作业分配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableBalancePlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

#### --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定\_\_\_enableDynamicPlan\_来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

### --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

# --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

#### 例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

#### 如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的Dist Cp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --dest oss://<yourB ucketName>/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis m 20

② 说明 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yam队列的名称。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

# --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

### 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

#### 命令示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20
```

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

# 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

● 使用归档 ( --archive ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive --parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

#### 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

#### 命令示例如下。

```
jindo distop --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --cleanUpPending --
parallelism 20
```

# 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里,避免每次使用时填写Accesskey。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

#### 此时代码示例如下。

```
jindo distop /tmp/jindo-distop-2.7.3.jar --src s3://smartdatal/ --dest s3://smartdatal/tmp --s3EndPoint s3-u s-west-1.amazonaws.com
```

# 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的 大小可能不相等。

# 5.5.4. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

#### 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-distcp-<version>.jar。
  - Hadoop 2.7及后续版本,请下载jindo-dist cp-3.0.0.jar。
  - Hadoop 3.x系列版本,请下载jindo-dist cp-3.0.0.jar。

#### 场景预览

Jindo Dist Cp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二: 使用Jindo Dist Cp成功导完数据后, 如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时, Dist Cp任务存在随时失败的情况,该使用哪些参数支持断点续传?
- 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Dist cp过程中可能已经产生了新文件,该使用哪些参数处理?
- 场景五:如果需要指定jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九: 如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时, 该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?
- 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret ),以及Endpoint信息,且该AccessKey具有写目标Bucket的 权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载 Jindo Dist Cp JAR包。

本场景示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10

⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch

# 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

#### 您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_REDUCE=0
```

参数含义如下:

- 。 Bytes Destination Copied: 表示目标端写文件的字节数大小。
- Bytes Source Read: 表示源端读文件的字节数大小。
- Files Copied:表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

# 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

在<mark>场景一</mark>的基础上,如果您的Distcp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Distcp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文件的Copy。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --dest oss://yang-hhht/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa nifest --parallelism 20

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

# 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信 息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

参数含义如下:

- --outputManifest : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如*manifest-2020-04-17.gz*,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Distcp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife
st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest-2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

# 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小,单位为MB。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid
th 6 --parallelism 10

# 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

#### 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

### 优化对比如下。



# 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey Ⅳ。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10
```

# 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。

需要在场景一的基础上增加 --outputCodec 参数, 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10
```

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

② 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

# 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录, 需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

#### 示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

# 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则,正则表达式。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.\*/([a-z]+).\*.txt' --parallelism 20

# 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10

# 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

```
<configuration>
<property>
<name>fs.jfs.cache.oss-accessKeyId</name>
<value>xxx</value>
</property>
<name>fs.jfs.cache.oss-accessKeySecret</name>
<value>xxx</value>
</property>
<property>
<name>fs.jfs.cache.oss-endpoint</name>
<value>oss-cn-xxx.aliyuncs.com</value>
</property>
</configuration>
```

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在 core-sit e.xml中。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

# 5.5.5. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存 储成本,或者加速访问数据的速度。

# 使用Jindo jfs

#### 执行以下命令,获取帮助信息。

```
jindo jfs -help archive
-archive -i/a <path> ... :
Archive commands.
```

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

#### 常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p 参数可以保证本地数据不受磁盘水位清理。

#### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

```
jindo jfs -uncache <path>
```

#### Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参 见对象存储OSS的存储类型介绍。

```
jindo jfs -archive -i|-a <path>
```

-i 参数可以归档数据至OSS低频存储类型。 -a 参数可以归档数据至OSS归档存储类型。

# Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

Unarchive默认可以将数据恢复成标准存储, -i 参数可以恢复数据至低频存储类型。 -o 参数可以临时解冻归档存储类型, 使数据临时可读。

#### Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

-detail 参数可以查看文件进度信息。 -sync 参数表示该命令需要同步等待分层存储任务结束才会退出。

#### ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -1s2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

```
drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse
-rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia_data.csv
-rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia_data.json
```

# 6.SmartData 3.5.x 6.1. SmartData 3.5.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS、JindoT able和相关工具集。本文介绍Smart Dat a(3.5.x)版本的更新内容。

# JindoFS OSS扩展和支持

优化OSS删除目录的性能。

# JindoSDK

- Java使用JindoSDK时, JindoSDK日志输出到Java日志中, 以提高可诊断性。
- 新增SDK端使用内存统计日志,可以看到当前JindoSDK使用的内存大小。

# JindoTable计算优化

- JindoTable新增native加速功能,可以对使用Spark、Hive或Presto读取存储在OSS和JindoFS上的ORC或Parquet格式的文件进行加速,详情请参见开启native查询加速。
- Hive支持JindoTable冷度统计,以统计Hive表访问频次,详情请参见JindoTable表或分区访问冷度收集。

# JindoFS工具集

增强JindoDistcp,支持通过阿里云监控(CloudMonitor)服务监控告警失败任务、移除了对AVX指令集的依赖、并新增使用冷归 档方式写入OSS等功能,详情请参见Jindo DistCp使用说明。

# 6.2. JindoFS Block模式

# 6.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

# 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

# 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。

ii. 单击namespace。

< 返回	a Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	滤				服务配置
配置搜	表				全部 i smartdata-site namespace client i storage
清瀬/				Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前jindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	描述	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>

# ⅲ. 单击确定。

- 4. 单击右上角的**保存**。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

	服务配置					
	全部   smartdata-site   namespace	client storage				
		storage.handler.threads	40		0	
		storage.watermark.low.ratio	0.2		0	
		storage.watermark.high.ratio	0.4		0	
		storage.oss.upload.threads	20		0	
参	数		描述			
si	orage.watermark.high.ratio		表示磁盘 用的磁盘:	表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。		
si	torage.watermark.low.ratio 表示使用量的下水位比例,触发清理后会自动清理》。 JindoFS数据目录占用空间清理到下水位。默认值:					

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

#### 2. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 6.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

# 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见使用Raft-RocksDB-Tablestore作为存储后端。

#### 单机RocksDB作为元数据服务的架构图如下所示。



# 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > Smart Dat a**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	● 正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<b>索</b>		Q	全部 smartdata-site namespace client storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
    - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX

## Smart Dat a Smart Dat a 3.5.x

参数	参数说明	示例
namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使 用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
	② 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

#### 7. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 8. 单击右上角的操作 > 启动All Components。

# 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. 准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809
```

25 jfs://test/

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 \_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ ji	ndo jfs -metaStatus.
==== emr-header-1:8101 ====	=
OtsUploader: _synced=1	
[RocksDB Row Counts of each	CF(Table)]

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。
  - 新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。
- 3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
参数	描述	示例
----	----	----

	是否开启从OTS恢复元数据,包括:	
namespace.backend.rocksdb.reco very.mode	• true • false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

==== emr-header-1:8101 ==== [Recovery From OTS Status] state: FINISH total 22855 rows.

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

[hadoop@emr-header-1 ~]\$ hadoop fs -cat jfs://test/testfile this is a test file

#### # 查看目录

```
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: ° true ° false	false

9. 重启集群。

i. 单击上方的**集群管理**页签。

ii. 在集群管理页面, 单击相应集群所在行的更多 > 重启。

# 6.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

# 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

? 说明	需要开启事务功能。
● 创建3 Mast	er的EMR集群,详情请参见 <mark>创建集群</mark> 。
🛚 高可用	高可用: 🕢 🦲
	部署方式: 🥝 🔷 2 Master
	<ul> <li>③ 3 Master 查看服务部署</li> </ul>
? 说明	如果没有部署方式 <i>,</i> 请 <mark>提交工单</mark> 处理。

# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,单击**集群服务 > SmartData**。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏,单击**集群服务 > SmartData**。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值) 。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤 7。

#### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jf s
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com

参数	参数说明	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 7. 单击右上角的操作 > 启动All Components。

## 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

[hadoop@emr-heade	r-1 ~]\$ hadoop fs -count	jfs://test/
1596	1482809	25 jfs://test/
(文件夹个数)	(文件个数)	

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail



- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

```
jindo jfs -metaStatus -detail
```

如图所示, LEADER节点的state为FINISH表示恢复完成。



7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式, 也是只读模式。

```
# 对比文件数量一致
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
    1596 1482809 25 jfs://test/
# 文件可正常读取(cat, get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

# 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 6.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

## 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

## 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 ismartdata-site namespace client istorage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则, 在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

#### 示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> s	how tables	;		
database	tableN	ame	isTemporary	
default audi	t_log	false		
default audi	t_log_sour	ce	false	
default fs_i	mage	false		

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sq	l> select * from	n audit_log_sou	urce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> select	* from audi	t_log li	imit 10;										
datetime al	lowed ugi	ip	ns	cmd	src	dst	perm	date					
2020-10-20 10:50:1	1.924 true	root	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	iadoop:r	wxrwx
r-x 2020-10-20													
2020-10-20 10:50:1	1.950 true	root	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11:26:0	6.445 true	root	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	hadoop:r	wxrwx
r-x 2020-10-20													
2020-10-20 11:26:0	6.469 true	root	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11:26:1	1.295 true	root	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-10-20													
2020-10-20 11:26:1	1.320 true	root	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20													
2020-10-20 11:26:1	4.368 true	root	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-10-20													
2020-10-20 11:26:1	4.393 true	root	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20													
2020-10-20 11:26:1	6.230 true	root	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-10-20													
2020-10-20 11:26:1	6.255 true	root	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20													
Time taken: 1.918	seconds. Fet	tched 10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 6.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

#### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview					
Start Time:	Fri Oct 16 12:29:24 2020				
Status:	Active				
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)				
Node:	Live Nodes: [2], Decommission Nodes: [0]				
Version:	3.0.0				
Build No:	fa0ea608a4;				

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)						
Namespace: jfs://test/						
Namespaces:	test					
Mode:	BLOCK_MODE					
Backend URI:	oss://					
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27					

StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

"卑击INOOEN" 应以 算有母	却个噬盆的全间使用情况。
-------------------	--------------

Overview							
Start Time: Fri Oct 16 12:29:		9:25 2020					
Version:		3.0.0					
Build Version:		fa0ea608a42a	je0e4ebcdbbfc3c041fe49f8e82e				
Storage Lists (4	.)						
Directory	Stora	ідеТуре	Capacity	Last Eviction Time	Partition Id		
/mnt/d	Disk		28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31		
/mnt/d	Disk		19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21		
/mnt/d	Disk		51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1		
/mnt/d	Disk		24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11		

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 6.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - 。 Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。

#### E-MapReduce



# 启用JindoFS Unix权限

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	a∰o Smart	tData ~	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 志			服务配置
配置搜	素			全部 I smartdata-site <b>namespace</b> client I storage
请蝓ノ	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager 🗈 Audit 💠 S	ettings				🔒 admin
Service Manager					
Service Manager					🛛 Import 🖉 Export
🗁 HDFS	+22	🗁 HBASE	+ 🛛 🖸	🕞 HIVE	+ 🛛 🖸
		emr-hbase	• 7 8	emr-hive	• 2 8

iii. 配置相关参数。

参数	描述
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test 。
Username	自定义。
Password	自定义。
Namenode URL	输入jfs://{namespace_name}/。
Authorization Enabled	使用默认值No。
Authentication Type	使用默认值Simple。
dfs.datanode.kerberos.principal	
dfs.namenode.kerberos.principal	石塘史
dfs.secondary.namenode.kerberos.principal	小炭司。
Add New Configurations	

#### iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hado op.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

# 3. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 6.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

# 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 令使用说明。

# 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。

策略名称	策略说明
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令, 获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

## 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用于一些高压缩比的文件。支持以下两种压缩策略。

策略名称	策略说明
NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path>: 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

# 6.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

# 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

#### 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

#### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

bin jindo jfs -dumpMetadata test-block
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class]
SLF4J: See http://www.slf4J.org/codes.htmlamut.ple\_bindings for an explanation.
SLF4J: Actual binding is of type [org.slf4j.impl.Log4]LoggerFactory]
SUcessfully upload newspace metadata to 055.

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

```
Sucessfully upload namespace metadata to OSS.
```

# 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如, 配置的 namespace.sysinfo.oss.uri 为 oss://abc/ ,则上传的文件会在 oss://abc/metadataDump 子目录中。

参数	说明
namespace.sysinfo.oss.uri	存储Bucket和路径。
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinfo.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

← → ↑ ↑ 1 oss://emrtest/svsin	fc/metadataDump test-block/
▲上传 + 创建目录 □	更多▼
□ 名称	类型 / 大小
□	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

#### 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

{		
	"type":"string",	/*INode <b>类型,</b> FILE <b>文件</b> DIRECTORY <b>目录</b> */
	"id": "string",	/*INode id*/
	"parentId" :"string",	/* <b>父节点</b> id*/
	"name":"string",	/*INode <b>名称</b> */
	"size": "int",	/*INode <b>大小,</b> bigint*/
	"permission":"int",	/*permission <b>以</b> int <b>格式存放</b> */
	"owner":"string",	/*owner <b>名称</b> */
	"ownerGroup":"string",	/*owner <b>组名称</b> */
	"mtime":"int",	/*inode <b>修改时间,</b> bigint*/
	"atime":"int",	/*inode <b>最近访问时间,</b> bigint*/
	"attributes":"string",	/* <b>文件相关属性</b> */
	"state":"string",	/*INode <b>状态</b> */
	"storagePolicy":"string	", /*存储策略*/
	"etag":"string"	/*etag*/
1		

# 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 audit\_log和fs\_image。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta data 生成的数据。

```
示例如下。
```

```
jindo-sql> show partitions fs_image;
partition
namespace=k /datetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)
```

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@e	mr-worke	r-2 had	doop]# jindo sql											
iindo-s	als show	table	C:	r										
databas	e	table	Name isTemp	orary										
default	audit_1	oa	false											
default	audit 1	og sour	rce false											
default	fs_imag	e	false											
Time to	ken: 0.3	45 seco	onds, Fetched 3 r	on(s)										
jindo-s	ql> sele	ct + fi	rom fs_image limi	t 10;										
atime	attr	etag	id mtime	natte	owner	ownerGroup	parentId	permission	size	state	storag	ePolicy	type	name
space	datetim	ie												
0			73110760050518	99448	16030840	70081 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_s	sk=2450819	root	root	334790	833296
5855433	489		Finalized	WARM	Director	'y kugou	2020_10_20	_10_50_36						
0			16534448041906	675495	16030840	071350 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_s	sk=2450820	root	root	334790	833296
5855433	489	0	Finalized	WARM	Director	'y kugou	2020_10_20	_10_50_36						
0			73110760050518	99470	16030840	070185 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_	sk=2450821	root	root	334790	833296
5855433	489	0	Finalized	WARM	Director	°y kugou	2020_10_20	_10_50_36						
0			11922762023479	287249	16030840	069581 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_:	sk=2450822	root	root	334790	833296
5855433	489		Finalized	WARM	Director	°y kugou	2020_10_20	_10_50_36						
0			10769840518872	441036	16030840	073592 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_	sk=2450823	root	root	334790	833296
5855433	489	0	Finalized	WARM	Director	•y kugou	2020_10_20	_10_50_36						
0			26993899866245	11354	16030840	068996 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_s	sk=2450824	root	root	334790	0833296
5855433	489	0	Finalized	WARM	Director	ry kugou	2020_10_20	_10_50_36						
0			11922762023479	287307	16030840	069875 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_	sk=2450825	root	root	334790	833296
5855433	489	0	Finalized	WARM	Director	ry kugou	2020_10_20	_10_50_36						
0			15464684820176	6500Z	16030840	072440 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_	sk=2450826	root	root	334790	0833296
5855433	489	ø	Finalized	WARM	Director	ry kugou	2020_10_20	_10_50_36						
0			16534448041906	675460	16030840	071170 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_	sk=2450827	root	root	334790	0833296
5855433	489	ø	Finalized	WARM	Director	'y kugou	2020_10_20	_10_50_36						
0			73110760050518	99544	16030840	070572 /tpcds	/orc/5000/we	b_returns/wr_return	ed_date_	sk=2450828	root	root	334790	0833296
5855433	489	0	Finalized	WARM	Director	'y kugou	2020_10_20	_10_50_36						

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

jindo-sql>	select cour	nt(*) from t	fs_image where	type =	"Directory"	and namespace="kugou"	and datetime="2020_10_20_	10_47_14";
count(1)								
11837								
Time taken:	6.852 secon	nds, Fetchee	d 1 row(s)					

# 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table\_name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

示例如下。

[hive> select * from inode_metadata_test8 limit 100;						
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futu			ing a different execut	ion engine (i.e. spark,	tez) or using Hive	1.X releases.
Query ID = root_202009						
Total jobs = 1						
(Launching Job 1 out of 1						
Number of reduce tasks is set to 0 since there's no reduce operator						
Starting Job = job_1 , Tracking URL = http://emr-heade	:20888	/proxy/appl				
Kill Command = /usr/lib/hadoop-current/bin/hadoop job -kill job_1599						
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0						
(2020-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0%						
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.22 se						
MapReduce Total cumulative CPU time: 1 seconds 220 msec						
Ended Job = job_:						
MapReduce Jobs Launched:						
Stage-Stage-1: Map: 1 Cumulative CPU: 1.22 sec HDFS Read: 6867 HDFS Write: 1						
Total MapReduce CPU Time Spent: 1 seconds 220 msec						
Directory 11274334386847219714 11274334386847219713 /uttest/oss			staff 1599545017615	1599545017615		WARM
Directory 11274334386847219719 11274334386847219713 /uttest/oss2			staff 1599545017654	1599545017654		WARM
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir			staff 1599545017636	1599545017636		WARM
File 11274334386847219715 11274334386847219714 /uttest/oss/file1			staff 1599545017632	1599545017632		WARM
File 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2			staff 1599545017642	1599545017642		WARM
File 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3			staff 1599545017651	1599545017651		WARM
Directory 11274334386847219728 11274334386847219719 /uttest/oss2/dir			caojie staff 159954	5017654 1599545017654		d WARM
File 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2			staff 1599545017658	1599545017658	Finalized 4	WARM
File 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3			staff 1599545017666	1599545017666	Finalized 4	WARM
Directory 11274334386847219713 17672823557433851905 /uttest 0		e staff :	1599545017615 159954	5017615 Final	lized WARM	
Time taken: 10.734 seconds, Fetched: 10 row(s)						

# 6.2.9. JindoFS Credential Provider使用说明

JindoFS的数据存储在OSS中,如果您需要访问JindoFS的数据,需要提供OSS的AccessKey才能访问。Smartdata 3.4.0及后续版 本支持JindoFS Credential Provider,您可以通过配置JindoFS Credential Provider,将加密后的AccessKey信息添加至文件中,以 避免泄露AccessKey信息。

# 配置JindoFS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smartdata-site服务配置。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击smart dat a-sit e页签。
- 3. 添加配置信息。
  - i. 在smartdata-site页签,单击右上角的自定义配置。
  - ii. 在**新增配置项**对话框中,新增如下配置。

参数	描述
fs.jfs.credentials.provider	配置com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential。例如, com.aliyun. emr.fs.auth.TemporaryAliyunCredentialsProvider,c om.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,c vider,com.aliyun.emr.fs.auth.EnvironmentVariable CredentialsProvider 。 Provider详情请参见Provider类型。

- ⅲ. 单击确定。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# Provider类型

• TemporaryAliyunCredentialsProvider

## 适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsPr ovider
fs.jfs.accessKeyld	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。
fs.jfs.securityToken	OSS的SecurityToken(临时安全令牌)。

#### • SimpleAliyunCredentialsProvider

## 适合使用长期有效的AccessKey访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvid er
fs.jfs.accessKeyld	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。

#### Environment VariableCredent ialsProvider

#### 该方式需要在环境变量中配置以下参数。

参数	参数说明		
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EnvironmentVariableCredential sProvider		
ALIYUN_ACCESS_KEY_ID	OSS的AccessKey ID。		
ALIYUN_ACCESS_KEY_SECRET	OSS的AccessKey Secret。		
ALIYUN_SECURITY_TOKEN	OSS的SecurityToken(临时安全令牌)。		
	⑦ 说明 仅配置有时效Token时需要。		

#### • JindoCommonCredentialsProvider

该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.JindoCommonCredentialsProvi der
jindo.common.accessKeyld	OSS的AccessKey ID。
jindo.common.accessKeySecret	OSS的AccessKey Secret。
jindo.common.securityToken	OSS的SecurityToken(临时安全令牌)。

#### • EcsStsCredentialsProvider

该方式无需配置AccessKey,可以免密方式访问OSS。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EcsStsCredentialsProvider

# 6.2.10. JindoFS Block模式加密使用说明

JindoFS Block模式支持文件加密,加密机制和使用方法与Apache HDFS的Encryption Zone类似。加密密钥通过密钥管理服务 (KMS)统一管理,您可以对有敏感数据的目录设置加密策略,然后就可以透明地在该目录下加密写入的数据和解密读取的数据,无需更改您的代码。

# 前提条件

- 已创建集群,详情请参见创建集群。
- 已开通密钥管理服务(KMS),详情请参见开通密钥管理服务。

# 背景信息

Block模式加密架构图如下:



# 配置JindoFS使用阿里云KMS

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击namespace页签。
- 3. 添加配置信息。
  - i. 在namespace页签, 单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下配置。

参数	描述
crypto.provider.type	Provider的类型,仅支持ALIYUN。
crypto.provider.endpoint	KMS的公网接入地址。详情请参见 <mark>调用方式</mark> 。
crypto.provider.kms.accessKeyld	访问KMS的AccessKey ID。
crypto.provider.kms.accessKeySecret	访问KMS的AccessKey Secret。

4. 保存配置。

```
i. 单击右上角的保存。
```

- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 5. 重启配置。
  - i. 在右上角选择操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅳ. 在确认对话框中, 单击确定。

# 使用JindoFS KeyProvider

Jindo KeyProvider负责对接KMS,加密密钥存储在KMS。KeyProvider基于KMS提供新增密钥、查询密钥和轮换密钥等功能。

• 新增密钥: 传入keyldName, 创建一个新的密钥。

jindo key -create -keyIdName <keyIdName>

⑦ 说明 本文示例中的<keyIdName>为您创建的密钥名称。

例如,执行以下命令新增policy\_test的密钥。

jindo key -create -keyIdName policy\_test

在阿里云KMS控制台,您可以看到新增了一个名为policy\_test的密钥。

密钥管理服务控制台	创建凭据
用户主密钥	凭据管家最新上线全托管RDS凭据,支持定时更新RDS数据库口令,减少数据库口令泄漏风险。
凭据	名称
证书	policy_test 普通凭据

• 查询密钥: 查看当前存在的密钥名。

jindo key -list

返回信息如下:

Listing Keys: policy\_test policy\_test2

• 轮换密钥:您可以根据Key ID定期更换密钥。更新密钥后Key Version会随之发生变化,即文件在加密时,使用最新的密钥进行

#### 加密, 文件在解密时使用现有文件的密钥版本进行解密。

jindo key -roll -keyIdName <keyIdName>

#### 例如,执行以下命令轮换密钥policy\_test。

jindo key -roll -keyIdName policy\_test

# 在阿里云KMS控制台,您可以看到密钥*policy\_test*的版本状态已经更新,之前的版本状态变成了ACSPrevious,新的版本状态为ACSCurrent。

密钥管理服务控制台	← policy_test			
用户主密钥	任根料体			
凭据	尤姞许頂			
证书	凭据信息	名称	policy_test	
		ARN	acs:kms:cn-hangzh	policy_test
		加密密钥	系统托管密钥	
		描述信息	Encryption Key for JindoFS File E	1
	∨ 版本列表 ⑦			
	存入凭据值			
	版本号			版本状态
	1613802530			ACSCurrent   状态管理
	1613801564			ACSPrevious   状态管理

# 管理JindoFS加密策略

您可以根据以下命令,设置和查看加密策略:

• 设置加密策略

jindo jfs -setCryptoPolicy -keyIdName <keyIdName> <path>

⑦ 说明 本示例的 <path> 为您访问JindoFS上文件的路径。例如*jfs://test/*。

● 查看加密策略

jindo jfs -getCryptoPolicy <path>

#### 设置和查看加密策略示例如下所示:

#### 1. 查看*jfs://test/*路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回信息显示为 {NONE} 。

2. 设置*jfs://test/*的加密策略。

jindo jfs -setCryptoPolicy -keyIdName policy\_test jfs://test/

#### 3. 进入bigboot目录,再次查看jfs://test/路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回如下信息。

SLF4J: Class path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/b2jindosdk/3.4.0-hadoop3.1/package/b2jindosdk-3.4 .0-hadoop3.1/lib/jindo-distcp-3.4.0.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/hadoop/3.2.1-1.0.1/package/hadoop-3.2.1-1.0.1/sha re/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html#multiple bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] 21/03/12 13:52:34 WARN: Unable to load native-hadoop library for your platform... using builtin-java clas ses where applicable 21/03/12 13:52:35 INFO: Jboot log name is /var/log/bigboot/jboot-20210312-135234-12953.LOG 21/03/12 13:52:35 INFO: Write buffer size 1048576, logic block size 134217728 21/03/12 13:52:35 INFO: cmd=getFileStatus, src=jfs://test/, dst=null, size=0, parameter=null, time-in-ms= 7. version=3.4.0 21/03/12 13:52:35 INFO: cmd=getCryptoPolicy, src=jfs://test/, dst=null, size=0, parameter=, time-in-ms=2, version=3.4.0 The crypto policy of path: jfs://test/ is {cipherSuite: AES\_CTR\_NOPADDING\_256, keyIdName: policy test2, k eyIdVersion: null, edek: , iv: } 21/03/12 13:52:35 INFO: Read total statistics: oss read average <none>, cache read average <none>, read o ss percent <none>

#### 设置完成后即可正常读写该路径下的文件。

◦ 拷贝本地文件至HDFS。

hadoop fs -put test.log jfs://test/

• 展示文件内容。

hadoop fs -cat jfs://test/test.log

# 6.3. JindoFS Cache模式

# 6.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓 存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主 要介绍JindoFS的缓存模式及其使用方式。

#### 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

# 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

#### 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

#### 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的集群管理页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜索	<u>ج</u>		0	全部 smartdata-site namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例	
jfs.namespaces.test.oss.uri		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>	
	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。	
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache	

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。

此配置无需重启Smart Dat a服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后, Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

服务配置					
全部 smartdata-site namespace	client storage				
	storage.handler.threads	40		0	
	storage.watermark.low.ratio	0.2		0	
	storage.watermark.high.ratio	0.4		0	
	storage.oss.upload.threads	20		0	
参数		描述			
storage.watermark.high.ratio 表示磁盘使用量的上水位比例,每块用的磁盘空间到达上水位即会触发清		用量的上水位比例,每块数据盘的Ji 间到达上水位即会触发清理。默认值	ndoFS数据目录占 查: 0.4。		
storage.watermark.low.ratio		表示使用量 JindoFS数据	的下水位比例,触发清理后会自动滞 5目录占用空间清理到下水位。默认	青理冷数据 <i>,</i> 将 值:0.2。	

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅰ٧. 在确认对话框中,单击确定。

#### 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明		
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。		

参数	参数说明		
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。		
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。		

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击namespace页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明		
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>		
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。		
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。		

# 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在服务配置区域的client页签,配置以下参数。

参数	参数说明		
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。		
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。		

● 在服务配置区域的smart dat a-sit e页签, 配置以下参数。

参数	参数说明				
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。				
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。				
fs.oss.committer.magic.enabled	② 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。				

# 6.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce (简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

## 前提条件

适用环境: ECS(EMR环境外)+Hadoop+JavaSDK。

## 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smartdata-aliyun-jfs-\*,jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

#### 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击**完成**。

#### 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击确定。

# 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏,单击实例与镜像 > 实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

# 步骤四:在ECS上设置环境变量

#### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP CLASSPATH=\$HADOOP CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

# 步骤五:测试免密方式访问的方法

1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
        while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
    }
}
```

# 6.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

## 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

# Smart Dat a Smart Dat a 3.5.x

# E-MapReduce

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		0	全部 i smartdata-site namespace client i storage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql	<pre>&gt; show tables;</pre>			
database	tableNam	1e	isTemporary	
default d	udit_log	false		
default d	udit_log_source	2	false	
default f	s_image	false		

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sql>	select * from	n audit_log_sour	ce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-20	10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null per	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-20	10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null per	rm=null	2020-10-20						
2020-10-20	11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null per	rm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-20	11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null per	rm=null	2020-10-20						
2020-10-20	11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s dst	t=null	<pre>perm=root:root:</pre>	rwxr-xx	2020-10	-20			
2020-10-20	11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null per	rm=null	2020-10-20						
2020-10-20	11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s dst	t=null	perm=root:root:	rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	n audit_log	lin	mit 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	nadoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	0:50:11.950	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true ro	ot (	(auth:SIMPL	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:h	nadoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	1:26:06.469	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true ro	ot (	(auth:SIMPl	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:11.320	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:14.368	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:14.393	true ro	ot (	(auth:SIMPL	LE)	192.	- 10 C	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:16.230	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:16.255	true ro	ot (	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ds. Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo	-sql>	select	cmd	,count(*	) from	audit_log	where	date=	"2020-	-10-20"	group	by	cmd	order	by	cmd;
cmd	COL	unt(1)														
getFi	leStat	tusReque	est	387												
listF	ilelet	Request	t	387												
Time	taken:	5.767	seco	nds. Fet	ched 2	row(s)										

# 6.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

# 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

# ○ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后,MapReduce Application Master或Spark Driver执行最终的Job Commit操作时,会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

#### 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域, 单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.jindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

置mapreduce.outputcommitter.factory.scheme.oss为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击smartdata-site页签。

- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

#### 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

# 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

#### 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。

- ii. 单击**配置**页签。
- iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

默认值为8。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

# 6.3.5. JindoFS OSS Credential Provider使用说明

Smart dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

# 配置JindoFS OSS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入smartdata-site页面。
  - i. 单击配置页签。
  - ii. 在**服务配置**区域,单击smartdata-site页签。
- 3. 在smartdata-site页签,根据配置方式修改或新增配置信息。

配置方式	描述
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数fs.jfs.cache.oss.credentials.provider,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如, com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。

配置方式	描述						
	<ul> <li>新增配置信息:</li> <li>i. 在smartdata-site页签,单击右上角的自定义配置。</li> <li>ii. 在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。</li> <li>例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> </ul>						
按照Bucket配置	<ul> <li>⑦ 说明 XXX为OSS的Bucket名称。</li> <li>ii. 单击确定。</li> </ul>						

## 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。
类型	描述
---	--
Environment Variable Credentials Provi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 按照Bucket配置

# 您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken (临时安全令 牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment VariableCredent ialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>

类型	描述
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 6.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

# 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息(Overview)

```
包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。
```

Overview	
Start Time:	Fri Oct 16 12:29:24 2020
Status:	Active
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)
Node:	Live Nodes: [2], Decommission Nodes: [0]
Version:	3.0.0
Build No:	fa0ea608a4/

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)					
Namespace:	jfs://test/				
Namespaces:	test				
Mode:	BLOCK_MODE				
Backend URI:	oss://				
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27				

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

# Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

#### • 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 6.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限:您可以设置文件的777权限,以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜	素			全部 smartdata-site namespace client storage
请输入			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。

#### 5. 重启配置。

- i. 单击右上角的操作 > 重启 Jindo Namespace Service。
- ii. 输入执行原因, 单击**确定**。
- 开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见概述。

ii. Ranger UI添加HDFS service。

Ranger	Access Manager	🗅 Audit	Settings				👸 admin
Service Mar	ager						
Service Man	ager						🛛 Import 🛛 🖾 Export
BH	DFS		+ 4 2	HBASE	+ 20	E HIVE	+ 2 2
				emr-hbase	Ø 8 8	emr-hive	• 8
	Dr5			emr-hbase	• 2 8	emr-hive	

#### iii. 配置相关参数。

参数	描述		
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。		
Username	自定义。		
Password	自定义。		
Namenode URL	输入jfs://{namespace_name}/。		
Authorization Enabled	使用默认值No。		
Authentication Type	使用默认值Simple。		
dfs.datanode.kerberos.principal			
dfs.namenode.kerberos.principal	7.14日		
dfs.secondary.namenode.kerberos.principal	۰ (۲-کپر. ا		
Add New Configurations			

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hado op.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

# 6.4. JindoTable

# 6.4.1. 开启native查询加速

JindoTable提供Native Engine,支持查询加速。系统默认不开启加速,开启之后可以提升在Spark、Hive或Presto上ORC或 Parquet格式文件的查询速度。

# 前提条件

已创建EMR-3.35.0及后续版本或EMR-4.9.0及后续版本,且ORC或Parquet文件已存放至JindoFS或OSS,创建集群详情请参见创建 集群。

# 背景信息

Spark、Hive和Presto上服务支持的引擎和存储格式如下所示。

引擎	ORC	Parquet
Spark2	支持	支持
Presto	支持	不支持
Hive2	不支持	支持

# 使用限制

- 不支持Binary类型。
- 不支持分区列存储在文件中的分区表。
- 不支持EMR 5.X及后续版本的EMR集群。
- 不支持代码spark.read.schema (userDefinedSchema), userDefinedSchema不同于文件schema issue。
- 支持Date类型区间为1400-01-01到9999-12-31。
- 同一个表中查询列不支持区分大小写。例如,NAME和name两个列在同一个表中无法使用查询加速。

# 提升Spark性能

1. 开启JindoTable ORC或Parquet加速。

#### ⑦ 说明

- 因为查询加速使用的是堆外内存,所以在Spark任务中建议添加配置 --conf spark.executor.memoryOverhead=
   4g 提高Spark申请额外资源用来进行加速。
- Spark调用读取ORC或Parquet时,需要使用native来启用加速。

#### 全局设置

详细请参见全局设置Spark。

○ Job级别设置

使用spark-shell或者spark-sql时可以添加Spark的启动参数。

spark.sql.extensions==io.delta.sql.DeltaSparkSessionExtension, com.aliyun.emr.sql.JindoTableExtension

作业详情请参见Spark Shell作业配置或Spark SQL作业配置。

2. 检查开启情况。



# 提升Presto性能

因为Presto已经内置JindoTable native加速的 catalog: hive-acc ,所以您可以直接使用 catalog: hive-acc 来启用查询加 速。

示例如下。

presto --server emr-header-1:9090 --catalog hive-acc --schema default

⑦ 说明 目前使用Presto查询加速功能时,暂不支持复杂数据类型,例如Map、Struct或Array。

# 提升Hive性能

↓ 注意 如果您对作业稳定性要求较高时,建议不要开启native查询加速。

因为EMR Hive 2.3.7(EMR-3.35.0)已经内置JindoTable Parquet加速的插件,所以您可以直接设 置hive.jindotable.native.enabled来启用查询加速,或者可以在控制台配置页面的*hive-site.xml*页签,添加自定义参 数**hive.jindotable.native.enabled为true**,来开启查询加速并重启Hive,此方式适用于Hive on MR和Hive on Tez。

### 代码示例如下。

set hive.jindotable.native.enabled=true;

⑦ 说明 目前使用Hive查询加速功能时,暂不支持复杂数据类型,例如Map、Struct或Array。

# 全局设置Spark

#### 1. 进入Spark页面。

- i. 登录阿里云E-MapReduce控制台。
- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的**集群管理**页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > Spark**。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.extensions,修改参数值 为io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtension。
- 4. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 5. 重启ThriftServer。
  - i. 在右上角选择操作 > 重启ThriftServer。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

# 6.4.2. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

# 前提条件

已创建表或者视图。

# 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的DataSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

# JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dataset、DataFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spark 2.4.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manager	nent									
New Cache										

JindoCube还提供了**spark.sql.cache.useDatabase**参数,可以针对业务方向,按不同的业务建立database,把需要建 cache的view放在这个database中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPartition**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true

参数	说明	示例值
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为**true**。

# JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击**集群管理**页签。
  - iii. 单击待操作集群所在行的集群ID。
  - iv. 单击左侧导航栏的**访问链接与端口**。
  - v. 在公网访问链接页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。
    - Knox相关使用说明请参见Knox。
  - vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
  - vii. 单击上方的Cube Management页签。
  - viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:

Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。 在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选

参数	描述	是否必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Table and View Information			
dataBase	tableName	CubeCache	action
ssb_10	customer	c_name c_address	raw cache cube cache
ssb_10	dates	© C_CIIV © C_nation © C_region	raw cache cube cache
ssb_10	lineorder	c_phone c_mktsegment	raw cache cube cache
ssb_10	lineorder_flatten	□ p_parkey □ p_name □ p_mfar	raw cache cube cache
ssb_10	p_lineorder	p_category p_brand	raw cache cube cache
ssb_10	part	_ p_color _ p_type _ o size	raw cache cube cache
ssb_10	supplier	Description	raw cache cube cache
		lo_quantity   MAX	
		lo_tax	
		Cancle OK	

JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

```
SELECT c_city, c_nation, c_region, MAX(lo_quantity), SUM(lo_tax)
FROM lineorder_flatten
GROUP BY c_city, c_nation, c_region;
```

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
МАХ	МАХ
MIN	MIN
AVG	COUNT, SUM
COUNT (DISTINCT)	PRE_COUNT_DISTINCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Det ail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作,Cache表示的数据并未持久化,需要继续构建Cache,从而持久化Cache数据到 HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新,需要更新Cache中的数据从而保持一致。JindoCube 支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spork 2.4.3	Jobs Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	nent Build	Page						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>							
Optional Filter	۵							
Column:	s_region	;	•					
Filter Type:	<ul> <li>Fixed Values</li> <li>Range Values</li> </ul>							
Values:	ASIA, AMERICA		Filter view data wit	th column value eq	uals to inpu	ut values.		
	Submit							

在构建JindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。 Column:过滤字段。 Filter Type:过滤类型,支持固定值和范围值两种。 Fixed Values:指定过滤值,可以多个,以","分隔。 Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

SELECT \* FROM lineorder\_flatten
WHERE s\_region == 'ASIA' OR s\_region == 'AMERICA';

单击**Submit**,提交构建任务,返回到Cache详细页面,对应的构建任务会提交到Spark集群中执行,在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

Spork 2.4.3	Jobs Stages Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	ment Timer Trigger I	Build Pag	e				
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Trigger Strategy:							
Start At:	2019-12-30 11:00:00						
Period:	1	Hour		\$			
Optional Step:	۵						
Step By:	TimeStamp Column(Long Type)     DateTime Column(String Type)						
Column Name:	lo_orderdate \$						
DateTime Format	Such as 'yyyy-MM-dd HI If availat	ole, filter would com	pare column as Tirr	estampT	ype value.		
	Submit						

定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由SparkDriver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、 构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之 释放。

3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

● 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3 Jobs	Stages Storage Environn	nent Executors SQL	Cube Management Streaming	JDBC/ODBC Server		Thr	ift JDBC/ODBC Server application UI
Cube Management							
viewDataBase	viewName	cacheName	enableRewrite	cacheType	cacheDataSize	lastUpdataTime	action
ssb_10	lineorder_flatten	lo_raw_cache	false	Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache							

○ 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 24.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application	tion UI
Cube Manage	ment [	Detail F	Page								
Name									Value		
Database									ssb_10		
View Name									lineorder_fatten		
Cache Name									lo_raw_cache		
Enabled									true		
Cache Type									RAW		
Cache Columns									C. catalogi, c. pameruber, b. catalogi, b. parkoy, b. support, b. catalogi, b. catalogi, b. parkoy, p. and catalogi, c. parkoy, b. and catalogi, c. parkoy, b. and catalogi, c. parkoy, b. and catalogi, c. parkoy, c. pa	ppriotity, Io_quantity, Io_exténdedprice, opkey, s_name, s_address, s_city, s_nation onth, d_daynuminweek, d_daynuminmonth dayinmonthfi, d_holidayfi, d_weekdayfi, ame, p_mfgr, p_category, p_brand, p_color	n, n, r,
Location									/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw		
Data Size									None		
Provider									PARQUET		
Partition By									lo_orderdate		
ZOrder By											
Last Update Time									2019/12/28 17:52:55		
Enabled  Partition Information Show 10 \$ entries										Search:	
Path						Size	•		Action		
								No data av	valiable in table		
Showing 0 to 0 of 0 entries	5									Previous	Next
Build Information     Latest cache building is fire	hished at 20	19-12-28 17	:52:55								
Build Cache Trigger	Period Build	1									

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delet**e删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ 说明 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

+ col name	data type		+
+		+	1 +
lo_orderkey	bigint	NULL	
lo custkev	int	NULL	
o partkey	int	NULL	
Lo_suppkey	int	NULL	
lo_orderdate	int	NULL	
lo_orderpriotity	string	NULL	
lo_shippriotity	int	NULL	
lo_quantity	bigint	NULL	
o_extendedprice	bigint	NULL	
lo_ordtotalprice	bigint	NULL	
	bigint	NULL	
o supplycost	bigint	NULL	
lo tax	bigint	NULL	
Lo_commitdate	int	NULL	
lo_shipmode	string	NULL	
_suppkey	int	NULL	
_name	string	NULL	
_address	string	NULL	
s_city	string	NULL	
s_nation	string	NULL	
_ region	string	NULL	
datekev	int	NULL	
date	string	NULL	
dayofweek	string	NULL	
_month	string	NULL	
_year	int	NULL	
i_yearmonthnum	int	NULL	
_yearmonth	string	NULL	
_daynuminweek	int	NULL	
_daynuminmonth	int	NULL	
_daynuminyear	int	NULL	
i_montnnuminyear	int	NULL	
i sellingseason	etring	NULL	
lastdavinweekfl	int	NULL	
lastdavinmonthfl	int	NULL	
holidayfl	int	NULL	
_weekdayfl	int	NULL	
_custkey	int	NULL	
_name	string	NULL	
_address	string	NULL	
city	string	NULL	
_nation	string	NULL	
phone	string	NULL	
mktsegment	string	NULL	
partkey	int	NULL	
name	string	NULL	
mfgr	string	NULL	
_category	string	NULL	
_brand	string	NULL	
_color	string	NULL	
_type	string	NULL	
size	int	NULL	
_concarner	serrinà	NULL	
Detailed Table Information			
Database	ssb_10	i	
Table	lineorder_flatten	i	
Dwner	hadoop	I	
Created Time	Sat Dec 28 17:30:02 CST 2019	I	
ast Access	Thu Jan 01 08:00:00 CST 1970		
Created By	Spark 2.2 or prior		
ype /	VIEW		) Adamandan' Ala avakiny' Addamadan' Adama at A
/lew lext	SELECI lineorder . lo_orderkey , lineorder . lo_l:	inenumber ,	lineorder . lo_custkey , lineorder . lo_partkey ,
leorder , ro_supprey / rineord	er`.`lo_ordtotalprice`, `lineorder`.`lo_discount`, `:	lineorder`.	`lo_revenue`, `lineorder`.`lo_supplycost`, `lineorder`. Jiar``s name` `supplier``s address` `supplier``s
<pre>c'.lo_extendedprice', 'lineord lo_tax', 'lineorder'.lo_commit y', 'supplier'.s_nation', 'su n', 'dates'.d_wear', 'dates'.' 'dates'.'d_monthnuminyear', 'da i_holidayfl', 'dates'.d_weekda customer'.'c_region', 'custome t'.'o brand.'nart'.n color'</pre>	<pre>date ', lineorder'.lo_shipmode ', 'supplier'.s_suppl pplier'.s_region', 'supplier'.s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.d_yearmonth', 'dates'.d_di tes'.d_weeknuminyear', 'dates'.d_sellingseason', ' yfl', customer'.c_ustkey', 'customer'.c_name', 'r'.c_phone', 'customer'.c_mktsegment', 'part'.p_p, 'part'.p_type'.bart'.n_sip'.nart'.p_contail</pre>	_datekey`, supp. _datekey`, aynuminweek dates`.`d_la customer`.`d artkey`, `pa ner` FROM `d	<pre>dates.'.d_date', 'dates'.d_dayofweek', 'dates'.d , 'dates'.d_daynuminmonth', 'dates'.d_daynuminyear astdayinweekfl', 'dates'.d_lastdayinmonthfl', 'dates c_address', customer'.c_city', customer'.c_nation art'.p_name', part'.p_mfgr', part'.p_category', sb 10'.linoorder'.sb 10'.sunnlier'.seh 10'.d</pre>
c'.lo_extendedprice', 'lineord lo_tax', lineordor.lo_commit ty', 'supplier'.'s_nation', 'su 'dates'.d_year', dates'. dates'.d_monthnuminyear', 'da _holidayfl', 'dates'.d_weekda customer'.c_region', 'custome customer'.c_region', 'custome customer'.c_region', 'custome'.	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppl pplier'.'s_region', 'supplier'.'s_phone', 'dates'.'d_ d_yearmonthnum', 'dates'.'d_yearmonth', 'dates'.'d_ tes'.'d_weeknuminyear', 'dates'.'d_sellingseason', ' yfl', 'customer'.'c_ustkey', 'customer'.c_name', ' r'.'c_phone', 'customer'.'c_mktsegment', 'part'.'p_ , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contain a'.'nart'.'</pre>	_datekey`, Supp _datekey`, aynuminweek dates`.`d_la customer`.`d artkey`, `pa ner` FROM `s	dates'.d_date', dates'.d_dayofweek', 'dates'.d , 'dates'.d_dayofweek', 'dates'.d sates'.d_daynumimonth, 'dates'.d_daynuminyear satdayinweekfl, 'dates'.d_lastdayinmonth1, 'dates _address', 'customer'.c_eity', 'customer'.c_nation art'.p_name', 'part'.p_mfgr', 'part'.p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.d
<pre>c'.lo_extendedprice', 'lineord Lo_tax', 'lineorder'.lo_commit y', 'supplier'.s_nation', 'su 'dates'.d_year', 'dates'. 'dates'.d_monthnuminyear', 'da j_holidayfl', 'dates'.d_weekda customer'.c_region', 'custome rt'.p_brand', 'part'.p_color' 's', 'ssb_10'.customer', 'ssb_1 EE lineorder'.'lo_rderdate'</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppl pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d d_yaarmonthnum', 'dates'.d_yaarmonth', 'dates'.d_d tes'.'d_weeknuminyear', 'dates'.d_sellingseason', ' yf'.'customer'.c_ustkey', 'customer'.c_name', ' r'.'c_phone', 'customer'.c_mktsegment', 'part'.p_p, , 'part'.p_type', 'part'.p_size', 'part'.p_contai 0'.part' = 'dates'.'d'datekey' AND 'lineorder'.'lo_custev'</pre>	_datekey`, Supp. _datekey`, ` aynuminweek dates`.`d_la customer`.`d artkey`, `pa ner` FROM `s `customer`	<pre>dates '.d_date', 'dates'.d_dayofweek', 'dates'.d dates '.d_date', 'dates'.d_dayofweek', 'dates'.d_ satdayinwoekfl, 'dates'.d_lastdayinmonth1, 'dates c_address', 'customer'.c_city', 'customer'.c_nation art'.p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.supplier', 'ssb_10'.d .c custkev' AND 'lineorder'.'lo supplier', 'solition'.</pre>
<pre>c'.lo_extendedprice', 'lineord lo_tax', 'lineorder'.lo_commit y', 'supplier'.s_nation', 'su i', 'dates'.d_wear', 'dates'.' 'dates'.d_monthnuminyear', 'da i_holidayfl', 'dates'.d_weekda customer'.c_region', 'custome rt'.'p_brand', 'part'.p_color' s', 'ssb_10'.'customer', 'ssb_1 RE lineorder'.lo_orderdate' s_suppkey'AND 'lineorder'.lo</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppl pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.'d_sellingseason', ' yfl', 'customer'.'c_ustkey, 'customer'.c_name', ' yfl', 'customer'.'c_mktsegment', 'part'.'p_p, , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contain 0'.part e' dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_artkey'  </pre>	_datekey`, `supp. _datekey`, ` aynuminweek dates`.`d_la customer`.`d artkey`, `pp her` FROM `s `customer`.	<pre>dates.'.d_date', 'dates'.d_dayofweek', 'dates'.d dates'.'d_date', 'dates'.d_dayofweek', 'dates'.d_ stdayinwekfl', 'dates'.'d_lastdayinmothfl', 'dates c_address', 'customer'.'c_city', 'customer'.'c_nation art'.'p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'d .'c_custkey' AND 'lineorder'.'lo_suppkey' = 'supplier</pre>
c'.lo_extendedprice', 'lineord lo_tax', 'lineordr'.lo_commit ty', 'supplier'.'s_nation', 'su 'dates'.dyear', 'dates'.' 'dates'.dymontnuminyear', 'da Jolidayfl', 'dates'.dywekda customer'.c_region', 'custome rt'.p_brand', 'part'.'p_color' ', 'ssb_10'.customer', 'ssb_1 ERE 'lineorder'.lo_orderdate' _suppkey' AND 'lineorder'.lo_ Table Properties	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppl pplier'.'s_region', 'supplier'.'s_phone', 'dates'.'d_ d_yearmonthnum', 'dates'.'d_yearmonth', 'dates'.'d_ tes'.'d_weeknuminyear', 'dates'.'d_sellingseason', ' yfl', 'customer'.c_ustkey', 'customer'.c_name', ' r'.'c_phone', 'customer'.'c_mktsegment', 'part'.'p_pn , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contain 0'.part' e 'dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_partkey'   [transient_lastDdf!me=157767599]</pre>	_datekey`, `supp_ _datekey`, ` aynuminweek dates`.`d_li customer`.`d artkey`, `p her` FROM `s `customer`.	<pre>dates.'d_date', 'dates'.d_dayofweek', 'dates'.d , 'dates'.d_dates'.d_dayofweek', 'dates'.d sates'.'d_daynuminmonth', 'dates'.'d_daynuminyear satdayinweekfl', 'dates'.d_lastdayinmonthf', 'dates _address', 'customer'.c_eity', 'customer'.c_nation art'.p_name', 'part'.p_erategory', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'d .'c_custkey' AND 'lineorder'.'lo_suppkey' = `supplier  </pre>
<pre>c'.lo_extendedprice', 'lineord Lo_tax', 'lineorder'.lo_commit y', 'supplier'.s_nation', 'su 'dates'.d_year', 'dates'. 'dates'.d_monthnuminyear', 'da j.holidayfl', 'dates'.d_weekda 'customer'.c_region', 'custome rt'.p_brand', 'part'.p_color' 's', 'ssb_18'.customer', 'ssb_1 ERE 'lineorder'.lo_orderdate' s_suppkey' AND 'lineorder'.lo_ Table Properties erde Library</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppl pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.d_yearmonth', 'dates'.d_ tes'.d_weeknuminyear', 'dates'.d_sellingseason', ' yf'l', customer'.c_ustkey', 'customer'.c_name', ' r'.c_phone', 'customer'.c_mktsegment', 'part'.p_p, 'part'.p_type', 'part'.'p_size', 'part'.'p_part' 0'.part'.'p_part'.'p_size', 'part'.'p_contain 0'.part' = 'dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_partkey'   [transient_lastDdlTime=1577677899] org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe</pre>	datekey`, Supp. _datekey`, ` daynuminweek dates`.`d_li customer`.`d artkey`, `pa her` FROM `s `customer`. 	<pre>dates.'d_date', 'dates'.d_dayofweek', 'dates'.d_ dates.'d_date', 'dates'.d_dayofweek', 'dates'.d_ satdayinwoekfl', 'dates'.d_lastdayinmonth1, 'dates _address', 'customer'.c_city', 'customer'.c_nation art'.p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.' .'c_custkey' AND 'lineorder'.'lo_suppkey' = 'supplier  </pre>
<pre>c'.lo_extendedprice', 'lineord Lo_tax', lineordr.lo_commit ty', 'supplier'.'s_nation', 'su ', 'dates'.d_worthnuminyear', 'da _holidayfl', 'dates'.d_weekda customer'.c_region', custome rt'.p_brand', 'part'.p_color' i, 'ssb_10'.customer', 'ssb_1 ERE 'lineorder'.lo_orderdate' _suppkey' AND 'lineorder'.lo_ 'sble Properties erde Library InputFormat</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppl pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.'d_sellingseason', ' yfl', 'customer'.c_custkey', 'customer'.c_name', ' yfl', 'customer'.c_mktsegment', 'part'.'p_p, 'part'.'p_type', 'part'.'p_size', 'part'.'p_contain 0'.part e' dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_partkey'   [transient_lastDdlTime=157767759] org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe org.apache.hadoop.mapred.TextInputFormat</pre>	datekey', supp. datekey', ' aynuminweek dates'.`d_l customer`.`d artkey`, `pa ner` FROM `s `customer`.   	dates'. d_date', 'dates'. d_dayofweek', 'dates'. d , 'dates'. d_daynumimonth', 'dates'.'d_daynuminyear satdayinwekfl', 'dates'.'d_lastdayinmonth', 'dates _address', 'customer'.c_city', 'customer'.c_nation art'.p_name', 'part'.p_mfgr', 'part.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'d .'c_custkey' AND 'lineorder'.'lo_suppkey' = `supplier
<pre>c'.lo_extendedprice', 'lineord lo_tax', 'lineorder'.lo_commit ty', 'supplier'.'s_nation', 'su 'dates'.dyear', 'dates'.' 'dates'.dymontnuminyear', 'da Jolidayfl', 'dates'.'dweekda customer'.c_region', 'customer', 'ss5_10'.customer', 'so5_1 REP lineorder'.'lo_orderdate' _suppkey' AND 'lineorder'.'lo_ Table Properties Serde Library InputFormat UtputFormat</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_suppli pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d d_yaarmonthnum', 'dates'.d_yaarmonth', 'dates'.d_di tes'.'d_weeknuminyear', 'dates'.d_sellingseason', ' yf'.'customer'.c_nste', 'sustomer'.c_name', ' r'.'c_phone', 'customer'.'c_mktsegment, 'part'.p_pr ', part'.p_type', part'.'p_size', 'part'.p_contai 0'.part' = 'dates'.d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.p_partkey'   [transient_lastDdTime=1577677599] org.apache.hadoop.mapred.TextInputFormat org.apache.hadoop.mapred.TextInputFormat</pre>	datekey', supp. aynuminweek' dates'.d_li customer'.'d ner' FROM ': `customer'.                   	<pre>dates . d_date', 'dates'. d_dayofweek', 'dates'. d m, 'dates'.'d_dates'. d_dayofweek', 'dates'.d stdayinwekfl', 'dates'.'d_lastdayinmonth', 'dates'. _address', 'customer'.'c_city', 'customer'.'c_nation art'.'p_name', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'d .'c_custkey' AND 'lineorder'.'lo_suppkey' = `supplier </pre>

0: jdbc:hive2://localhost:10001/ssb\_10>

基于lineorder\_flatten视图简单查询的执行计划如下:

<pre>[0: jdb:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10;</pre>	
19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.support.soll11.reserved keywords does not exist	
++ / /////////////////////////////	
plan l	
<pre>/ == Physical Plan ==</pre>	
CollectLimit 10	
+- *(1) SortMergeJoin [lo_partkey#313], [p_partkey#359], Inner	
:- *(0) SOT [10_par(key#313 ASC NULS F1XS1], Talse, 0 : +- Exchange hashpartitioning(10 partkey#313, 200)	
: +- *(7) SortMergeJoin [lo_custkey#312], [c_custkey#351], Inner	
: :- *(4) Sort [lo_custkey#312 ASC NULLS FIRST], false, 0	
: : +- Exchange hashpartitioning(o_custkey#siz, 200) : : +- + (3) BroadcastHashDoin [lo orderdat#315]. [d datekey#334]. Inner, BuildRight	
: : :- *(3) BroadcastHashJoin [lo_suppkey#314], [s_suppkey#327], Inner, BuildRight	
: : : : : : : : : : : : : : : : : : :	
<pre>xey#313/) : : : +- Scan hive ssb 10.lineorder [lo orderkev#310L, lo linenumber#311L, lo custkev#312, lo partkev#313, lo suppkev#314, lo ord</pre>	
erdate#315, lo_orderpriotity#316, lo_shippriotity#317, lo_quantity#318L, lo_extendedprice#319L, lo_ordtotalprice#320L, lo_discount#321L, lo_revenue#32	
2L, lo_supplycost#323L, lo_tax#324L, lo_commitdate#325, lo_shipmode#326], HiveTableRelation `ssb_10`.'lineorder`, org.apache.hadoop.hive.serde2.lazy.L	
azysimpleserue, [lo_orderkey#site], lo_linenumber#sitL, lo_custkey#aiz, lo_parkkey#site, lo_suppkey#site, lo_orderdate#site, lo_orderpriotity#site, lo_briderpriotity#site, lo_suppkey#site, lo_s	
ommitdate#325, lo_shipmode#326]	
: : : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))	
: : : +- *(1) Filter isnotnuli(s_suppKey#32/) : : : : +- Scan hive ssh 10, supplier (s suppkey#327, s name#328, s address#329, s citv#330, s nation#331, s region#332, s phone#	
333], HiveTableRelation `ssb_10`.`supplier`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [s_suppkey#327, s_name#328, s_address#329, s_city#330	
, s_nation#331, s_region#332, s_phone#333]	
: : +- broadcastexchange HashedkelationbroadcastMode(List(cast(input(e, int, taise) as bigint))) : : : +- *(2) Filter isnotnull(d datekev#334)	
: : + Scan hive ssb_10.dates [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearm	
onth#340, d_daynuminweek#341, d_daynuminmonth#342, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweek	
T1#34/, d_lastdaylmmontnt#348, d_nolldayT1#349, d_weekdayT1#3509, htvelablekelation ssp_10 . dates , org.apache.nadoop.nive.serde2.lazy.lazySimpleSer The [d datakeu#335, d data#35, d data%], d_month#337, d_var#338, d_var#monthmu#339, d_varmonth#34, d_davlmmimonth#34, datakeu#338, d_datakeu#335, d_davld_week#345, d_month#337, d_var#monthmu#339, d_varmonth#34, d_davlmimonth#34,	
2, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweekf#347, d_lastdayinmonthf1#348, d_hollidayf1#349,	
d_weekdayf1#350]	
: +- *(6) Sort [c_custkey#351 ASC NULLS FIRST], false, 0 +- Evchange hashpartitioning(c_custkey#351 200)	
: +- *(5) Filter ((isnotnull(c_region#356) & (c_region#356 = ASIA)) && isnotnull(c_custkey#351))	
: +- Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk	
tsegment#358], HiveTableRelation ssb_10 . customer , org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [c_custkey#351, c_name#352, c_address#353, c cit##354 . c nation#355 . c region#3562 . c mbreamont#358]	
+- +(10) Sort [m_parkey#359 ASC NULLS FIRST], false, 0	
+- Exchange hashpartitioning(p_partkey#359, 200)	
+- $*(9)$ Filter isnotnull(p_partkey#359) +- scap bige sch la part (n_partkey#359) n_pame#360 n_mfnr#361 n_category#362 n_brand#363 n_calor#366 n_type#365 n_category#366 n_conta	
iner#367], HiveTableRelation 'ssb_10', part, org., apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [p_partkey#356, p_name#366, p_mgr#361, p_category#3	
62, p_brand#363, p_color#364, p_type#365, p_size#366, p_container#367]	1
+	
1 job state 10. 50 state 10. 90 stat	
在为line order flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计	
划。伏化后的中行计划加下,	
יו אצע או נואננים באיאר א , יא	
0: idbc:hive2://localhost:10001/ssb 10> explain select * from lineorder flatten where c region = 'ASIA' limit 10:	
19/12/30 14:17:47 INFO [main] HiveConf: Found configuration file file:/etc/ecm/spark-conf-2.4.3-hadoop2.8-1.4.2/hive-site.xml	
19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive enforce bucketing does not exist	
17/12/30 14:1/:4/ WRAW [main] Hivedon: i Hivedon of name five.support.sqlii.reserved.keywords does not exist	
plan	
+	
Collectimit 10	
+- *(1) Project [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderdate#133, lo_orderpriotity#134, lo_ship	
priotity#136, lo_quantity#1361, lo_extendedprice#137L, lo_ordtotalprice#138L, lo_discount#139L, lo_revenue#140L, lo_supplycost#141L, lo_tax#142L, lo_c	
<pre>vumacuacumary, n_artymoutering, s_ouppreyming, s_name#ing, s_dutressmith, s_oityming, s_nation#ing, s_region#ing, s_phone#io1, 34 more fields) +- *(1) Filter (isnothull(c.region#174)&amp;&amp; (c.region#174 = ASIA))</pre>	
+- *(1) FileScan parquet [lo_orderkey#128L,lo_linenumber#129L,lo_custkey#130,lo_partkey#131,lo_suppkey#132,lo_orderpriotity#134,lo_shippriotity#	
135, lo_quantity#136(,lo_extendedprice#137), lo_ordtotalprice#1381, lo_discount#1391, lo_revenue#1461, lo_supplycost#1411, lo_tax#1421, lo_commitdate#143, lo_ bienede#1460, puepeeu#1460, puepeeu#1460, poete 1400, poete 1400, poete 1400, poete 1400, poete 1400, poete 140	
sinjmuderine, suppreyring, and saddress i 4/, scity#140, sciton#149, scregion#120, sphone#121, catekey#122, 34 MOYE 1121031 BATCheC: TUP, Format: Parquet, Location: IMMenoryFileIndex[/user/hive/warehouse/sb 10/ CACHE/lineorder flatten/raw], PartitionFileIndex[/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex[/user/lineors/ileIndex]/user/lineors/ileIndex[/user/lineors/ileIndex[/user/lindex]/user/lineors/ileIn	
shedFilters: [IsNotNull(c_region), EqualTo(c_region,ASIA)], ReadSchema: struct <lo_orderkey:bigint,lo_linenumber:bigint,lo_custkey:int,lo_partkey:int,l< td=""><td></td></lo_orderkey:bigint,lo_linenumber:bigint,lo_custkey:int,lo_partkey:int,l<>	
o_suppkey:int,lo_or	

可以看到,优化后的执行计划省去了lineorder flatten视图的所有计算逻辑,直接访问HDFS中Cache的数据。

#### 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性, 而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据, 用户需要根据查询对于数据一致性的需求, 触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

# 6.4.3. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

# 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。

- -accessStat
- 语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

- -cache
- 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。

• 示例: 缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

- -uncache
- 语法

jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示删除集群本地磁盘上指定表或分区的缓存数据。

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partition col2=2,... 的格式。

● 示例:

```
○ 删除集群本地磁盘上表db1.t2的缓存数据。
```

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

#### -archive

● 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

#### -unarchive

● 语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示将归档数据转为标准存储。

-o 将归档数据转为解冻, -i 将归档数据转为低频。

• 示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

jindo table -unarchive -i -t db1.t2

#### -status

语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - ∘ 查看表db1.t2的状态。

jindo table -status -t db1.t2

○ 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

```
-optimize
```

语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

- -showTable
- 语法

jindo table -showTable {-t} <dbName.tableName>

功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例: 展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

- -showPartition
- 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

jindo table -showPartition -t db1.t1 -p date=2020-10-12

- -listTables
- 语法
  - jindo table -listTables [-db] [dbName]
- 功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

- 示例:
  - 。 展示default 库中的表。

jindo table -listTables

○ 列出数据库db1中的表。

jindo table -listTables -db db1

# -dumpmc

● 语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-table} <tablename> {-p} <partitionSpec> {-f} <csv[tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络Tunnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是

参数	描述	是否必选
-p	分区信息 。例如 pt=xxx ,多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

```
• 功能
```

表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。

• 示例:

○ Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord

• Dumpmc Maxcompute表(CSV格式)至OSS存储。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv

# 6.4.4. JindoTable表或分区访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

#### 前提条件

```
已创建集群,详情请参见创建集群。
```

#### 背景信息

JindoTable支持收集访问Hive表的记录,收集的数据保存在SmartData服务的Namespace中。

Smart Dat a 3.2.x版本开始支持Spark、Hive和Presto引擎,Spark和Presto的数据收集默认是打开的,如果需要关闭,请参见关闭 <mark>热度收集</mark>。Hive的数据收集默认是关闭的,如果需要打开,请参见<mark>开启Hive热度收集</mark>。

# 数据查询

JindoTable提供了命令方式查询热度信息。

● 语法

jindo table -accessStat <-d [days]> <-n [topNums]>

days 和 topNums 为正整数。当只设置天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

```
jindo table -accessStat -d 7 -n 20
```

JindoTable使用详情,请参见JindoTable使用说明。

#### 开启Hive热度收集

1.

2. 在顶部菜单栏处,根据实际情况选择地域和资源组。

- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改Hive的参数值。
  - i. 在左侧导航栏,选择**集群服务 > Hive**。
  - ii. 在Hive服务页面,单击配置页签。

iii. 搜索参数hive.exec.post.hooks, 在参数值后追加com.aliyun.emr.table.hive.HivePostHook。

- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - i. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
  - ii. 在执行集群操作对话框,输入执行原因。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

#### 关闭热度收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。
  - Hive服务:
    - a. 在左侧导航栏,选择集群服务 > Hive。
    - b. 在Hive服务页面, 单击配置页签。
    - c. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤	服务配置
配置搜索	全部   hive-site
hive.exec.post.hooks © Q	
配置范围	hive.exec.post.hooks

- Spark服务:
  - a. 在左侧导航栏,选择集群服务 > Spark。
  - b. 在Spark服务页面,单击配置页签。
  - c. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSQLQueryListener。

状态 部署拓扑 配置 配置修改历史	
配置边读 配置搜索 spark.sql.queryExecutionListeners Q Q	服务配置 全部   spark-defaults
配置范围	spark.sql.queryExecutionListeners <b>com.aliyun.emr.table.sparkSQLQueryListener</b>

- Presto服务:
  - a. 在左侧导航栏,选择集群服务 > Presto。
  - b. 在Presto服务页面, 单击配置页签。
  - c. 搜索参数event-listener.name, 删除参数值中的内容。

```
6. 保存配置。
```

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 7. 重启服务。
  - 。 Hive服务:
    - a. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Spark服务:
    - a. 在Spark服务页面,选择右上角的操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Presto服务:
    - a. 在Presto服务页面,选择右上角的操作 > 重启All Components。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。

# 6.4.5. JindoTable表或分区访问冷度收集

JindoTable表或分区的访问冷度收集功能可以为您维护表或分区上次的访问时间,从而筛选出最近没有被访问的数据,帮助您优 化数据存储方式,节约成本。例如,在数据分析中,您可以把部分不常用的分区数据移动到成本更低的存储介质以节约成本。

### 前提条件

已创建EMR-3.35.0及后续版本或EMR-4.9.0及后续版本,创建详情请参见创建集群。

# 背景信息

Smart Dat a 3.5.x版本开始支持Hive、Spark和Presto组件的冷度收集功能。该功能目前默认不开启,如果需要开启,请参见<mark>开启</mark> Spark冷度收集、开启Hive冷度收集和开启Presto冷度收集。

⑦ 说明 因为冷度收集与热度收集使用相同的hooks或Listeners,所以开启组件的冷度收集时会同时打开热度收集功能。 表或分区访问热度收集的详情,请参见JindoTable表或分区访问热度收集。

# 使用限制

- 不支持DLF数据湖元数据。
- Hive CLI、HiveServer2、Spark SQL CLI、Spark Thrift server和Presto服务所在IP需要有权限访问元数据底层存储(MySQL或 RDS)。
- 仅支持Hive、Spark和Presto组件的冷度收集。

# 数据查询

JindoTable提供了命令方式查询冷度信息。

语法

```
jindo table -leastUseStat -n <num> [-i/-ignoreNever]
```

num是显示的条目数量,应为正整数。-i/-ignoreNever为可选参数,如果设置该参数,则会过滤掉从未被访问过的表或分区。

功能

展示最久未被访问的表或分区。

• 示例:查询最久未被访问的表或分区的20条记录。

jindo table -leastUseStat -n 20

返回如下图所示三列结果。

tdb.t1	pid=20/qid=ten	2021-03-26	13:53:52
tdb.t2		2021-03-26	13:53:58

- 第一列为表的名字,格式:数据库名.表名。
- 。 第二列为分区名字,格式:第一分区列=列值/第二分区列=列值/...,如果表为非分区表则为空。
- 第三列为最近一次访问的时间,格式: yyyy-MM-dd HH:mm:ss。

⑦ 说明 如果为分区表,则只显示到分区级别,表本身不会单独显示。

JindoTable使用详情,请参见JindoTable使用说明。

### 开启Spark冷度收集

- 1. 进入Spark页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Spark。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.queryExecutionListeners,确保参数值包含com.aliyun.emr.table.spark.SparkSQLQueryListener,如果存在多个Listeners时使用英文逗号(,)隔开。

<返回 ✔ Spark ➤ ●正常	0 查看
状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置
配置搜索 spark.sql.queryExecutionListeners	全部   spark-defaults
配置范围	spark.sql.queryExecutionListeners com.aliyun.emr.table.sparkSQLQueryListener, C

- 4. 添加自定义配置。
  - i. 在服务配置页面,单击spark-defaults页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为spark.sql.query.update.access.time.enabled,Value为true。

亲	新增配置项 × ×					
	* Key	* Value	描述	撮作		
	spark.sql.query.update.access.time.enabled	true		删除		
	添加					
			ſ	确定 取消		

- Ⅳ. 单击确定。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。

- i. 在右上角选择**操作 > 重启All Component s**。
- ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- iii. 在确认对话框中,单击确定。

#### 开启Hive冷度收集

- 1. 进入Hive页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Hive。
- 2. 在Hive服务页面,单击配置页签。
- 3. 搜索参数hive.exec.post.hooks,确保参数值包含com.aliyun.emr.table.hive.HivePostHook,如果存在多个hooks时使用英文逗号(,)隔开。

<返回 <sup>℃</sup> ‱ Hive ~ ● 正常	
状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤	服务配置
配置搜索 hive.exec.post.hooks	全部   hive-site
配置范围	hive.exec.post.hooks com.aliyun.emr.meta.hive.hook.LineageLoggerHook

- 4. 添加自定义配置。
  - i. 在服务配置页面, 单击hive-site页签。
  - ii. 单击右上角的自定义配置。
  - iii. 在新增配置项对话框中,设置Key为hive.hook.update.access.time.enabled, Value为true。

新	所增配置项 X						
* Key		* Value	拉法	操作			
	hive.hook.update.access.time.enabled	true		删除			
	添加						
				确定取消			

- ⅳ. 单击确定。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

# 开启Presto冷度收集

- 1. 进入Presto页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。

v. 在左侧导航栏,选择集群服务 > Presto。

- 2. 在Presto服务页面,单击配置页签。
- 3. 搜索参数event-list ener.name,确保参数值包含jindo-prest o-post-event-list ener。

< 返回 🧼 Presto 🖌 🍨 正常	
状态 部署拓扑 配置 配置修改历史	
配置过滤 配置搜索	服务配置 全部   event-listener.properties
event-listener.name	event-listener.name jindo-presto-post-event-listener

4. 添加自定义配置。

i. 在服务配置页面,单击event-listener.properties页签。

- ii. 单击右上角的自定义配置。
- iii. 在新增配置项对话框中,设置Key为listener.update.access.time.enabled,Value为true。

狮	新哨配直坝 X						
* Кеу		* Value	描述	操作			
	listener.update.access.time.enabled	true		删除			
	添加						
				确定 取消			

- iv. 单击**确定**。
- 5. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 6. 重启所有组件。
  - i. 在右上角选择操作 > 重启All Components。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - ⅲ. 在**确认**对话框中,单击**确定**。

# 6.5. 工具集

# 6.5.1. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

#### 前提条件

已创建集群, 详情请参见创建集群。

## 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

# 挂载

⑦ 说明 依次在每个节点上执行挂载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,新建目录。

mkdir /mnt/jfs

#### 3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

返回用户在服务端配置的所有命名空间列表。

test testcache

#### 2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

#### 3. 创建目录。

mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dir1/

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

#### 如果您想使用Python方式写入和读取文件,请参见如下示例:

# 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

# 2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

#### 返回如下信息。

my first file This file

# 卸载

```
⑦ 说明 依次在每个节点上执行卸载操作。
```

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 6.5.2. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见登录集群。

2. 执行以下命令, 获取帮助信息。

jindo distcp --help

#### 返回信息如下。

help - Print help text
src=VALUE - Directory to copy files from
dest=VALUE - Directory to copy files to
parallelism=VALUE - Copy task parallelism
outputManifest=VALUE - The name of the manifest file
previousManifest=VALUE - The path to an existing manifest file
requirePreviousManifest=VALUE - Require that a previous manifest is present if specified
copyFromManifest - Copy from a manifest instead of listing a directory
srcPrefixesFile=VALUE - File containing a list of source URI prefixes
srcPattern=VALUE - Include only source files matching this pattern
deleteOnSuccess - Delete input files after a successful copy
outputCodec=VALUE - Compression codec for output files
groupBy=VALUE - Pattern to group input files by
targetSize=VALUE - Target size for output files
enableBalancePlan - Enable plan copy task to make balance
enableDynamicPlan - Enable plan copy task dynamically
enableTransaction - Enable transation on Job explicitly
diff - show the difference between src and dest filelist
ossKey=VALUE - Specify your oss key if needed
ossSecret=VALUE - Specify your oss secret if needed
ossEndPoint=VALUE - Specify your oss endPoint if needed
policy=VALUE - Specify your oss storage policy
cleanUpPending - clean up the incomplete upload when distcp job finish
queue=VALUE - Specify yarn queuename if needed
bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed
s3Key=VALUE - Specify your s3 key
s3Secret=VALUE - Specify your s3 Sercet
s3EndPoint=VALUE - Specify your s3 EndPoint

### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo DistCp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo DistCp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

#### --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制*/data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定\_\_\_\_srcPattern\_的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Found 6 items							
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09							
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,复制以log结尾的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPattern .\*\.lo
g --parallelism 20

#### 执行以下命令,查看目标bucket的内容。

```
hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03
```

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-

      1
      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

# --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --deleteOnSuccess -parallelism 20

#### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下:

Found 6 item	IS	
-rw-rw-rw-	1	938 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/000151.sst.gz</yourbucketname>
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/1.log.gz</yourbucketname>
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/2.log.gz</yourbucketname>
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/OPTIONS-000109.g</yourbucketname>
Z		
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/emp01.txt.gz</yourbucketname>
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/emp06.txt.gz</yourbucketname>

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

② 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。

#### --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

# --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录, 查看运行期间复制的文 件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --parallel ism 20

#### 执行以下命令,查看运行期间复制的文件。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst

#### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

#### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

## --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
      Found 4 items

      drwxrwxrwx -
      0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

      drwxrwxrwx -
      0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20
```

### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

```
hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly_table/2017-02-01
hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly_table/2017-02-02
```

#### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

例如,执行以下命令,查看如下文件夹中的数据。

```
hdfs dfs -ls /data/incoming/hourly table/2017-02-01/03
```

#### 返回信息如下。

Found 8 items				
-rw-r	2 root hadoop	2252 2020-04-17 20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst	
-rw-r	2 root hadoop	4891 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/1.log	
-rw-r	2 root hadoop	4891 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/2.log	
-rw-r	2 root hadoop	4891 2020-04-17 21:08	/data/incoming/hourly_table/2017-02-01/03/5.log	
-rw-r	2 root hadoop	4891 2020-04-17 21:08	/data/incoming/hourly_table/2017-02-01/03/6.log	
-rw-r	2 root hadoop	4891 2020-04-17 20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001	
09				
-rw-r	2 root hadoop	1016 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt	
-rw-r	2 root hadoop	1016 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt	

#### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --targetSize=10 --g
roupBy='.\*/([a-z]+).\*.txt' --parallelism 20

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

# --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为Dist Cp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo Dist Cp的作业分配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableBalancePlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

### --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定 --enableDynamicPlan 来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

# --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --enableTransaction
--parallelism 20

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

#### 如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的Dist Cp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --dest oss://<yourB ucketName>/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis m 20

② 说明 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerip:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yam队列的名称。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

# --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

### 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

# 命令示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --ossKey <yourAcces
sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20
```

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

● 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive --parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --policy ia --par
allelism 20
```

#### 清理残留文件

在您的DistCp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

# 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

### 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里 , 避免每次使用时填写Accesskey。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

#### 此时代码示例如下。

jindo distop /tmp/jindo-distop-2.7.3.jar --src s3://smartdatal/ --dest s3://smartdatal/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

### 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distep Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

```
② 说明 如果您的Dist Cp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的
大小可能不相等。
```

# 6.5.3. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

### 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-dist cp-<version>.jar。
  - Hadoop 2.7及后续版本,请下载jindo-dist cp-3.0.0.jar。
  - 。 Hadoop 3.x系列版本,请下载jindo-distcp-3.0.0.jar。

#### 场景预览

Jindo DistCp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二: 使用Jindo Dist Cp成功导完数据后, 如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?
- 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新文件,该使用哪些参数处理?
- 场景五:如果需要指定jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六: 当通过低频或者归档形式写入OSS, 该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九:如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时,该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二: 如果Copy完文件, 需要删除原文件, 只保留目标文件时, 该使用哪些参数?
- 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret ),以及Endpoint信息,且该AccessKey具有写目标Bucket的 权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

#### 本场景示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10

⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启 -enableBatch 参数来进行优化。优化命令如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch

#### 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_REDUCE=0
```

参数含义如下:

- Bytes Destination Copied: 表示目标端写文件的字节数大小。
- 。 Bytes Source Read: 表示源端读文件的字节数大小。
- 。 Files Copied: 表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

# 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

在<mark>场景一</mark>的基础上,如果您的Distcp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Distcp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文件的Copy。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table le --dest oss://yang-hhht/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa nifest --parallelism 20 file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

# 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信 息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 参数含义如下:

- --outputManifest: 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如manifest-2020-04-17.gz,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Distcp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife
st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest-2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

# 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid
th 6 --parallelism 10
```

### 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

● 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table
--ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism
20
```

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

# 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。
如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

#### 优化对比如下。



# 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

示例如下。

hadoop	jar jindo	-distcp- <ve< th=""><th>ersion&gt;.jar</th><th>src</th><th>s3a:/</th><th>/yourbucket/</th><th>dest</th><th>oss:/</th><th>/yang-hh</th><th>ht/hourly</th><th>_table</th><th>s3Key</th><th>you</th></ve<>	ersion>.jar	src	s3a:/	/yourbucket/	dest	oss:/	/yang-hh	ht/hourly	_table	s3Key	you
rkey -	-s3Secret	yoursecret	s3EndPoin	t s3-u	is-wes	t-1.amazonaws	s.com -	-paral	lelism 1	0			

场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则jindo DistCp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

② 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

• 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

#### 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

--targetSize:合并文件的最大大小,单位MB。

• --groupBy : 合并规则, 正则表达式。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.\*/([a-z]+).\*.txt' --parallelism 20

场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10

# 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

```
<configuration>
<property>
<name>fs.jfs.cache.oss-accessKeyId</name>
<value>xxx</value>
</property>
<name>fs.jfs.cache.oss-accessKeySecret</name>
<value>xxx</value>
</property>
<property>
<name>fs.jfs.cache.oss-endpoint</name>
<value>oss-cn-xxx.aliyuncs.com</value>
</property>
</configuration>
```

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

# 6.5.4. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存储成本,或者加速访问数据的速度。

# 使用Jindo jfs

执行以下命令,获取帮助信息。

```
jindo jfs -help archive
-archive -i/a <path> ... :
Archive commands.
```

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令

#### E-MapReduce

- Unarchive命令
- Status命令
- ls2命令

#### Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p 参数可以保证本地数据不受磁盘水位清理。

#### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

#### Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参见对象存储OSS的存储类型介绍。

jindo jfs -archive -i|-a <path>

-i 参数可以归档数据至OSS低频存储类型。 -a 参数可以归档数据至OSS归档存储类型。

# Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

```
Unarchive默认可以将数据恢复成标准存储, -i 参数可以恢复数据至低频存储类型。 -o 参数可以临时解冻归档存储类型, 使数据临时可读。
```

## Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

-detail 参数可以查看文件进度信息。 -sync 参数表示该命令需要同步等待分层存储任务结束才会退出。

#### ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -ls2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse -rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia\_data.csv -rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia\_data.json

# 7.SmartData 3.4.x 7.1. SmartData 3.4.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS、JindoT able和相关工具集。本文介绍Smart Dat a(3.4.x)版本的更新内容。

# JindoFS OSS扩展和支持

- 新增OSS recoverable Output Stream功能,支持Flush和Recover API。适用于高可靠写入场景,例如Flume。
- 优化OSS Rename操作性能,结合OSS服务端提升Rename操作的执行时间。
- 优化OSS多版本下的List操作性能,避免Bucket多版本下大量临时文件影响目录的List性能。
- 优化OSS多版本JindoMagicCommitter性能,新增JindoDirectCommitter。
- 增强Credentials Provider框架,新增JindoCommonCredentialsProvider。
- 优化文件Create操作的性能,去掉OSS写入时的冗余检查。

# JindoFS存储优化

JindoFS Block模式支持数据加密,加密密钥支持阿里云密钥管理服务KMS(Key Management Service)和国际AES加密算法。

# JindoTable计算优化

完善Native Orc Reader, Block模式支持新的免密方式。

# JindoFS工具集

增强JindoDistcp,优化增量迁移场景。例如,迁移HDFS数据至OSS时,实现迁移路径的Checksum比对。

# JindoFS生态支持

新增Python版本的Jindo OSS SDK,支持基本的OSS操作,兼容OSS2 Python库。

# 7.2. JindoFS Block模式

# 7.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

## 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

# 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。

ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	」 述				服务配置
配置搜	素				全部 smartdata-site <b>namespace</b> client storage
清輸	λ.			Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前jindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

描述	示例
	oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
表示test命名空间的后端存储。	② 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。
表示test命名空间为块存储模式。	block
表示存储后端OSS的AccessKey ID。	XXXX
表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>
	描述 表示test命名空间的后端存储。 表示test命名空间为块存储模式。 表示存储后端OSS的AccessKey ID。 表示存储后端OSS的AccessKey zecret。

#### ⅲ. 单击确定。

- 4. 单击右上角的**保存**。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

	服务配置 全部   smartdata-site   namespace   client storage						
		storage.handler.threads	40		0		
		storage.watermark.low.ratio	0.2		0		
		storage.watermark.high.ratio	0.4		0		
		storage.oss.upload.threads	20		0		
, Wi	参数		描述				
storage.watermark.high.ratio			表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。				
storage.watermark.low.ratio			表示使用 JindoFS数	量的下水位比例,触发清理后会自动 据目录占用空间清理到下水位。默认	清理冷数据,将 \值:0.2。		

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

#### 2. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 7.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

## 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见使用Raft-RocksDB-Tablestore作为存储后端。

#### 单机RocksDB作为元数据服务的架构图如下所示。



#### 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	â Smart	tData 🗸	● 正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<b>索</b>		Q	全部 smartdata-site namespace client storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
    - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX

参数	参数说明	示例
namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使 用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: <ul> <li>true</li> <li>false</li> <li>当设置为true时,需要在SmartData服务</li> <li>完成初始化前,开启OTS异步上传功能。</li> </ul> <li>⑦ 说明 如果SmartData服务已</li> <li>完成初始化,则不能再开启该功能。</li> <li>因为OTS的数据已经落后于本地</li>	true
	RocksDB的数据。	

#### 7. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 8. 单击右上角的操作 > 启动All Components。

## 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. 准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809
```

25 jfs://test/

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 \_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ jindo jfs -metaStatus
emr-header-1:8101
)tsUploader: _synced=1
RocksDB Row Counts of each CF(Table)]

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。
  - 新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。
- 3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false

参数	描述	示例

	是否开启从OTS恢复元数据,包括:	
namespace.backend.rocksdb.reco very.mode	• true	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

```
==== emr-header-1:8101 ====
[Recovery From OTS Status]
state: FINISH
total 22855 rows.
```

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一致
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
       1596 1482809 25 jfs://test/
# 文件可正常读取 (cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root
                                  0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r----- 1 hadoop hadoop
-rw-r----- 1 hadoop hadoop
# 只读状本 不可你对去你
                                  5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
                                  20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021 , ErrorMsg: Namespace is under recovery mode, and is read-only.
```

8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的**集群管理**页签。
  - ii. 在集群管理页面, 单击相应集群所在行的更多 > 重启。

# 7.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

## 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。



# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,单击**集群服务 > SmartData**。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏,单击**集群服务 > SmartData**。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值) 。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤 7。

#### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例	
namespace.ots.instance	Tablestore实例名称。	emr-jf s	
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk	
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX	
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com	

参数	参数说明	示例
namespace.backend.raft.async.ot	是否开启OTS异步上传,包括: o true o false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	
s.enabled	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 7. 单击右上角的操作 > 启动All Components。

#### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. (可选)准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。



jindo jfs -metaStatus -detail



- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: ° true ° false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

```
jindo jfs -metaStatus -detail
```

如图所示, LEADER节点的state为FINISH表示恢复完成。



7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

#### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/

1596 1482809 25 jfs://test/

# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 7.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

#### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

#### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

# Smart Dat a Smart Dat a 3.4.x

## E-MapReduce

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		0	全部 i smartdata-site namespace client i storage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中,输入执行原因,单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> show tables	s;		
database table	Name	isTemporary	
default audit_log	false		
default audit_log_sour	rce	false	
default fs_image	false		

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sa	<pre>l&gt; select * from</pre>	n audit_log_	source limit	10;				
datetime	allowed	ugi ip	ns	cmd src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	oop:rwxrwxr-	x 2020-10-	20				
2020-10-	20 10:50:11.950	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	oop:rwxrwxr-	x 2020-10-	20				
2020-10-	20 11:26:06.469	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=listFileletRequest src=jfs	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s	dst=null	perm=root:r	oot:rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:r	oot:rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	m audit_log	lim	it 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true roo	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	0:50:11.950	true roo	ot (	auth:SIMPL	LE)	192.	- 10 C	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true roo	ot (	auth:SIMPL	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	1:26:06.469	true roo	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true roc	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:11.320	true roc	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:14.368	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:14.393	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:16.230	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:16.255	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ds, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-	sql>	select	cmd	,count(*)	) from	audit_log	where	date="	2020-	10-20"	group	by	cmd	order	by	cmd;
cmd	cou	int(1)														
getFil	.eStat	usReque	est	387												
listFi	lelet	Request		387												
Time t	aken:	5.767	seco	nds. Feta	hed 2	row(s)										

# 7.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

#### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

## 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web U功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview						
Start Time:	Fri Oct 16 12:29:24 2020					
Status:	Active					
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)					
Node:	Live Nodes: [2], Decommission Nodes: [0]					
Version:	3.0.0					
Build No:	fa0ea608a4;					

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)						
Namespace: jfs://test/						
Namespaces:	test					
Mode:	BLOCK_MODE					
Backend URI:	oss://					
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27					

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

Overview								
Start Time:		Fri Oct 16 12:29	ri Oct 16 12:29:25 2020					
Version: 3.0.0								
Build Version: fa0ea608a			a5e0e4ebcdbbfc3c041fe49f8e82e					
Storage Lists (4)								
Directory	Stora	адеТуре	Capacity	Last Eviction Time	Partition Id			
/mnt/d	Disk		28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31			
/mnt/d	Disk		19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21			
/mnt/d	Disk		51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1			

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 7.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	a∰o Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	t滤			服务配置
配置搜	索		_	全部 smartdata-site namespace client storage
请输入	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的**保存**。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager B Audit Ø	Settings				🔐 admin
Service Manager					
Service Manager					Import Export
	+22	🗁 HBASE	+ 22	🕞 HIVE	+ 🛛 🕰
		emr-hbase	• 6 =	emr-hive	

iii. 配置相关参数。

参数	描述		
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。		
Username	自定义。		
Password	自定义。		
Namenode URL	输入jfs://{namespace_name}/。		
Authorization Enabled	使用默认值No。		
Authentication Type	使用默认值Simple。		
dfs.datanode.kerberos.principal			
dfs.namenode.kerberos.principal	<b>太</b> 指写		
dfs.secondary.namenode.kerberos.principal	小県う。		
Add New Configurations			

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。 以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

#### 3. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 7.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

# 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 令使用说明。

## 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。

策略名称	策略说明
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令,获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

### 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用 于一些高压缩比的文件。支持以下两种压缩策略。

策略名称	策略说明
NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path>  $\ldots$ 

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path>: 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

# 7.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

# 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

#### 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

#### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

ibin jindo jfs -dumpMetadata test-block SLF4J: Class path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.htal&mu.cuple\_bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] Sucessfully upload namespace metadata to 055.

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

```
Sucessfully upload namespace metadata to OSS.
```

# 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如, 配置的 namespace.sysinfo.oss.uri 为 oss://abc/ ,则上传的文件会在 oss://abc/metadataDump 子目录中。

参数	说明
namespace.sysinfo.oss.uri	存储Bucket和路径。
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinf o.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

← → ↑ 1 oss://emrtest/svsinfd/metadataDump te	est-block/
▲上传 + 创建目录 □	
□ 名称	类型 / 大小
2020_09_14_18_58_16	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

#### 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

#### E-MapReduce

{		
	"type":"string",	/*INode <b>类型,</b> FILE <b>文件</b> DIRECTORY <b>目录</b> */
	"id": "string",	/*INode id*/
	"parentId" :"string",	/* <b>父节点</b> id*/
	"name":"string",	/*INode <b>名称</b> */
	"size": "int",	/*INode <b>大小,</b> bigint*/
	"permission":"int",	/*permission <b>以</b> int <b>格式存放</b> */
	"owner":"string",	/*owner <b>名称</b> */
	"ownerGroup":"string",	/*owner <b>组名称</b> */
	"mtime":"int",	/*inode <b>修改时间,</b> bigint*/
	"atime":"int",	/*inode <b>最近访问时间,</b> bigint*/
	"attributes":"string",	/* <b>文件相关属性</b> */
	"state":"string",	/*INode <b>状态</b> */
	"storagePolicy":"string	", /*存储策略*/
	"etag":"string"	/*etag*/
}		

# 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 audit\_log和fs\_image。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta data 生成的数据。

```
示例如下。
```

```
jindo-sql> show partitions fs_image;
partition
namespace=k /datetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)
```

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@er Spark m	nr-worke aster: y	r-2 hado arn, App	op]# jind lication	lo sql Id: app	,											
jindo-se	ql> show	tables;														
databas	e	tableNa	me	isTempo	orary											
default	audit_L	og	talse													
default	audit_l	og_sourc	e	false												
default	fs_imag	e	false													
Time ta	ken: 0.3	45 secon	ids, Fetch	ied 3 ro	m(s)											
jindo-se	ql> sele	ct * fro	m fs_imag	e limit	: 10;											
atime	attr	etag	id	mtime	name	owner	ownerGr	roup	parentId	permission	size	state	storage	Policy	type	name
space	datetim															
0			73110760	0505189	9448	1603084	070081	/tpcds/	'orc/5000/web	_returns/wr_returned	d_date_s	k=2450819	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			16534448	0419066	75495	1603084	071350	/tpcds/	orc/5000/web	_returns/wr_returned	_date_s	k=2450820	root	root	334790	833296
5855433	489		Finalize	!d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			73110760	0505189	9470	1603084	070185	/tpcds/	orc/5000/web	_returns/wr_returned	l_date_s	k=2450821	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			11922762	0234792	87249	1603084	069581	/tpcds/	'orc/5000/web	_returns/wr_returned	_date_s	k=2450822	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			10769840	5188724	41036	1603084	073592	/tpcds/	orc/5000/web	_returns/wr_returned	_date_s	k=2450823	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			26993899	8662451	1354	1603084	068996	/tpcds/	orc/5000/web	_returns/wr_returned	_date_s	k=2450824	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			11922762	0234792	87307	1603084	069875	/tpcds/	orc/5000/web	_returns/wr_returned	date_s	k=2450825	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						
0			15464684	8201766	5002	1603084	072440	/tpcds/	/orc/5000/web	_returns/wr_returned	date_s	k=2450826	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	rv	kugou	2020_10_20_	10_50_36						
0			16534448	0419066	75460	1603084	071170	/tpcds/	orc/5000/web	_returns/wr_returned	date_s	k=2450827	root	root	334790	833296
5855433	489	0	Finalize	d	WARM	Directo	r'v	kudou	2020 10 20	10 50 36						
0			73110760	0505189	9544	1603084	070572	/tpcds/	orc/5000/web	_returns/wr_returned	date_s	k=2450828	root	root	334790	833296
5855433	489		Finalize	d	WARM	Directo	ry	kugou	2020_10_20_	10_50_36						

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

jindo-sql> count(1)	select	count(*)	from f	s_image	where	type =	"Directory"	and	namespace="kugou"	and	datetime="2020_10_20_10_47_14";	
11837 Time taken:	6.852 s	econds,	Fetched	1 row(	5)							

# 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table\_name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

示例如下。

hive> select * from inode_metadata_test8 limit 100;	nive> select * from inode metadata test8 limit 100;								
MARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X releases.									
Query ID = root_202009/									
Total jobs = 1									
Launching Job 1 out of 1									
Number of reduce tasks is set to 0 since there's no reduce operator									
Starting Job = job 1 , Tracking URL = http://emr-heade		0888/proxy.	applica						
Kill Command = /usr/lib/hadoop-current/bin/hadoop job -kill job 1599									
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0									
2020-09-08 14:57:26.112 Stage-1 wap = 0%, reduce = 0%									
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.22 se									
MapReduce Total cumulative CPU time: 1 seconds 220 msec									
Ended Job = job :									
MapReduce Jobs Launched:									
Stage-Stage-1: Map: 1 Cumulative CPU: 1.22 sec HDFS Read: 6867 HDFS Write: 1	524 SUCCE								
Total MapReduce CPU Time Spent: 1 seconds 220 msec									
Directory 11274334386847219714 11274334386847219713 /uttest/oss		11 cao	jie sta	f 159954501761	5 1599545017615		Finalized	WARM	
Directory 11274334386847219719 11274334386847219713 /uttest/oss2		11 cao	jie sta	f 159954501765	4 1599545017654		Finalized		
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir		11 cao	iie sta	1 159954501763	5 1599545017636		Finalized		
File 11274334386847219715 11274334386847219714 /uttest/oss/file1		20 cao	iie sta	1 159954501763	2 1599545017632		Finalized		
File 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2		20 cao	iie sta	f 159954501764	2 1599545017642		Finalized	WARM	
File 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3		20 cao	iie sta	f 159954501765	1 1599545017651		Finalized	WARM	
Directory 11274334386847219728 11274334386847219719 /uttest/oss2/dir		511	cao	ie staff 1599	545817654 159954	5017654	Finali:	zed	WARM
File 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2		20 cao	iie sta	f 159954501765	8 1599545017658		Finalized	WARM	
File 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3		20 cao	iie sta	1 159954501766	5 1599545017666		Finalized	WARM	
Directory 11274334386847219713 17672823557433851905 /uttest 0		aoiie sta	ff 159	545017615 1599	545017615	Finaliz			
Time taken: 10.734 seconds, Fetched: 10 row(s)									

# 7.2.9. JindoFS Credential Provider使用说明

JindoFS的数据存储在OSS中,如果您需要访问JindoFS的数据,需要提供OSS的AccessKey才能访问。Smartdata 3.4.0及后续版 本支持JindoFS Credential Provider,您可以通过配置JindoFS Credential Provider,将加密后的AccessKey信息添加至文件中,以 避免泄露AccessKey信息。

# 配置JindoFS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smartdata-site服务配置。
  - i. 单击配置页签。
  - ii. 在服务配置区域, 单击smartdata-site页签。
- 3. 添加配置信息。
  - i. 在smartdata-site页签,单击右上角的自定义配置。
  - ii. 在**新增配置项**对话框中,新增如下配置。

参数	描述
fs.jfs.credentials.provider	配置com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential。例如, com.aliyun. emr.fs.auth.TemporaryAliyunCredentialsProvider,c om.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,c vider,com.aliyun.emr.fs.auth.EnvironmentVariable CredentialsProvider 。 Provider详情请参见Provider类型。

#### ⅲ. 单击确定。

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

#### Provider类型

• TemporaryAliyunCredentialsProvider

适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsPr ovider
fs.jfs.accessKeyId	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。
fs.jfs.securityToken	OSS的SecurityToken(临时安全令牌)。

• SimpleAliyunCredentialsProvider

适合使用长期有效的AccessKey访问OSS的情况。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvid er
fs.jfs.accessKeyld	OSS的AccessKey ID。
fs.jfs.accessKeySecret	OSS的AccessKey Secret。

#### • Environment VariableCredent ialsProvider

#### 该方式需要在环境变量中配置以下参数。

参数	参数说明					
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.EnvironmentVariableCredential sProvider					
ALIYUN_ACCESS_KEY_ID	OSS的AccessKey ID。					
ALIYUN_ACCESS_KEY_SECRET	OSS的AccessKey Secret。					
	OSS的SecurityToken(临时安全令牌)。					
ALIYUN_SECURITY_T OKEN	⑦ 说明 仅配置有时效Token时需要。					

#### • JindoCommonCredentialsProvider

该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。

参数	参数说明
fs.jfs.credentials.provider	com.aliyun.emr.fs.auth.JindoCommonCredentialsProvi der
jindo.common.accessKeyld	OSS的AccessKey ID。
jindo.common.accessKeySecret	OSS的AccessKey Secret。
jindo.common.securityToken	OSS的SecurityToken(临时安全令牌)。

• EcsStsCredentialsProvider

该方式无需配置AccessKey,可以免密方式访问OSS。

参数	参数说明
fs.jfs.credentials.provider	${\tt com.aliyun.emr.fs.auth.EcsStsCredentialsProvider}$

# 7.2.10. JindoFS Block模式加密使用说明

JindoFS Block模式支持文件加密,加密机制和使用方法与Apache HDFS的Encryption Zone类似。加密密钥通过密钥管理服务 (KMS)统一管理,您可以对有敏感数据的目录设置加密策略,然后就可以透明地在该目录下加密写入的数据和解密读取的数据,无需更改您的代码。

### 前提条件

- 已创建集群,详情请参见创建集群。
- 已开通密钥管理服务(KMS),详情请参见开通密钥管理服务。

# 背景信息

Block模式加密架构图如下:



## 配置JindoFS使用阿里云KMS

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 在**服务配置**区域,单击namespace页签。
- 3. 添加配置信息。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下配置。

参数	描述
crypto.provider.type	Provider的类型,仅支持ALIYUN。
crypto.provider.endpoint	KMS的公网接入地址。详情请参见 <mark>调用方式</mark> 。
crypto.provider.kms.accessKeyld	访问KMS的AccessKey ID。
crypto.provider.kms.accessKeySecret	访问KMS的AccessKey Secret。

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 在右上角选择操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅳ. 在确认对话框中, 单击确定。

# 使用JindoFS KeyProvider

Jindo KeyProvider负责对接KMS,加密密钥存储在KMS。KeyProvider基于KMS提供新增密钥、查询密钥和轮换密钥等功能。

• 新增密钥: 传入keyldName, 创建一个新的密钥。

jindo key -create -keyIdName <keyIdName>

⑦ 说明 本文示例中的<keyIdName>为您创建的密钥名称。

例如,执行以下命令新增policy\_test的密钥。

jindo key -create -keyIdName policy\_test

在阿里云KMS控制台,您可以看到新增了一个名为policy\_test的密钥。

密钥管理服务控制台	创建凭据
用户主密钥	凭据管家最新上线全托管RDS凭据,支持定时更新RDS数据库口令,减少数据库口令泄漏风险。
凭据	名称
证书	policy_test 普通凭据

• 查询密钥: 查看当前存在的密钥名。

jindo key -list

返回信息如下:

Listing Keys: policy\_test policy\_test2

• 轮换密钥:您可以根据Key ID定期更换密钥。更新密钥后Key Version会随之发生变化,即文件在加密时,使用最新的密钥进行

#### 加密, 文件在解密时使用现有文件的密钥版本进行解密。

jindo key -roll -keyIdName <keyIdName>

#### 例如,执行以下命令轮换密钥policy\_test。

jindo key -roll -keyIdName policy\_test

# 在阿里云KMS控制台,您可以看到密钥*policy\_test*的版本状态已经更新,之前的版本状态变成了ACSPrevious,新的版本状态为ACSCurrent。

密钥管理服务控制台	← policy_test			
用户主密钥	任据详情			
凭据				
证书	凭据信息	名称	policy_test	
		ARN	acs:kms:cn-hangzh	policy_test
		加密密钥	系统托管密钥	
		描述信息	Encryption Key for JindoFS File E	<u>/</u>
	∨ 版本列表 ⑦			
	存入凭据值			
	版本号			版本状态
	1613802530			ACSCurrent   状态管理
	1613801564			ACSPrevious   状态管理

# 管理JindoFS加密策略

您可以根据以下命令,设置和查看加密策略:

• 设置加密策略

jindo jfs -setCryptoPolicy -keyIdName <keyIdName> <path>

⑦ 说明 本示例的 <path> 为您访问JindoFS上文件的路径。例如*jfs://test/*。

● 查看加密策略

jindo jfs -getCryptoPolicy <path>

#### 设置和查看加密策略示例如下所示:

#### 1. 查看*jfs://test/*路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回信息显示为 {NONE} 。

2. 设置*jfs://test/*的加密策略。

jindo jfs -setCryptoPolicy -keyIdName policy\_test jfs://test/

#### 3. 进入bigboot目录,再次查看jfs://test/路径的加密策略。

jindo jfs -getCryptoPolicy jfs://test/

返回如下信息。

SLF4J: Class path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/b2jindosdk/3.4.0-hadoop3.1/package/b2jindosdk-3.4 .0-hadoop3.1/lib/jindo-distcp-3.4.0.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/opt/apps/ecm/service/hadoop/3.2.1-1.0.1/package/hadoop-3.2.1-1.0.1/sha re/hadoop/common/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html#multiple bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] 21/03/12 13:52:34 WARN: Unable to load native-hadoop library for your platform... using builtin-java clas ses where applicable 21/03/12 13:52:35 INFO: Jboot log name is /var/log/bigboot/jboot-20210312-135234-12953.LOG 21/03/12 13:52:35 INFO: Write buffer size 1048576, logic block size 134217728 21/03/12 13:52:35 INFO: cmd=getFileStatus, src=jfs://test/, dst=null, size=0, parameter=null, time-in-ms= 7. version=3.4.0 21/03/12 13:52:35 INFO: cmd=getCryptoPolicy, src=jfs://test/, dst=null, size=0, parameter=, time-in-ms=2, version=3.4.0 The crypto policy of path: jfs://test/ is {cipherSuite: AES\_CTR\_NOPADDING\_256, keyIdName: policy test2, k eyIdVersion: null, edek: , iv: } 21/03/12 13:52:35 INFO: Read total statistics: oss read average <none>, cache read average <none>, read o ss percent <none>

#### 设置完成后即可正常读写该路径下的文件。

◦ 拷贝本地文件至HDFS。

hadoop fs -put test.log jfs://test/

○ 展示文件内容。

hadoop fs -cat jfs://test/test.log

# 7.3. JindoFS Cache模式

# 7.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓 存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主 要介绍JindoFS的缓存模式及其使用方式。

#### 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

## 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

#### 配置OSS Scheme (推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

#### 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的集群管理页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smar	tData ∽	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 述			服务配置
配置搜	素 \		a	全部 smartdata-site namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。

此配置无需重启SmartData服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后, Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

	服务配置					
	全部 🗆 smartdata-site 🗏 namespace	client storage				
		storage.handler.threads	40		0	
		storage.watermark.low.ratio	0.2		0	
		storage.watermark.high.ratio	0.4		0	
		storage.oss.upload.threads	20		0	
参数		描述				
storage.watermark.high.ratio		表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。		录占		
st	storage.watermark.low.ratio		表示使用 JindoFS数	量的下水位比例,触发清理后会自动 据目录占用空间清理到下水位。默认	青理冷数据 <i>,</i> 料 .值:0.2。	冬

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - Ⅳ. 在确认对话框中,单击确定。

#### 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面, 单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。

参数	参数说明
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击namespace页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。

# 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在**服务配置**区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

● 在服务配置区域的smart dat a-sit e页签, 配置以下参数。

参数	参数说明
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。
fs.oss.committer.magic.enabled	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。
	② 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。

# 7.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce (简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

# 前提条件

适用环境:ECS(EMR环境外)+Hadoop+JavaSDK。
# 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smartdata-aliyun-jfs-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

### 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击**完成**。

### 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击确定。

# 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏, 单击实例与镜像 > 实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

# 步骤四:在ECS上设置环境变量

### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

# 步骤五:测试免密方式访问的方法

### 1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
        while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
    }
}
```

# 7.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

### Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 i smartdata-site <b>namespace</b> client i storage

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则, 在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

### 示例如下:

• 执行如下命令显示表。

show tables;

<pre>&gt; show tables;</pre>			
tableNa	me	isTemporary	
udit_log	false		
udit_log_sourc	е	false	
s image	false		
	> show tables; tableNan udit_log udit_log_source s image	<pre>&gt; show tables; tableName udit_log false udit_log_source 's imaae false</pre>	> show tables; tableName isTemporary udit_log false udit_log_source false s image false

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sq	l> select * from	n audit_log_sou	urce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	= //kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> se	lect * from	n audit_lo	g li	.mit 10;										
datetime	allowed	ugi i	p	ns	cmd	src	dst	perm	date					
2020-10-20 10	:50:11.924	true r	oot	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:	nadoop:r	wxrwx
r-x 2020-	10-20													
2020-10-20 10	:50:11.950	true r	oot	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	:26:06.445	true r	oot	(auth:SIMP	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:	nadoop:r	wxrwx
r-x 2020-	10-20													
2020-10-20 11	:26:06.469	true r	oot	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	:26:11.295	true r	oot	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20													
2020-10-20 11	:26:11.320	true r	oot	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 11	:26:14.368	true r	oot	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20													
2020-10-20 11	:26:14.393	true r	oot	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 11	:26:16.230	true r	oot	(auth:SIMP	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20													
2020-10-20 11	:26:16.255	true r	oot	(auth:SIMP	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken: 1	.918 second	ds, Fetche	d 10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 7.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

# 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

# ↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后, MapReduce Application Master或Spark Driver执行最终的Job Commit操作时, 会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

### 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.jindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

置**mapreduce.outputcommitter.factory.scheme.oss**为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。

- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

### 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

### 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。

```
ii. 单击配置页签。
```

- iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

```
默认值为8。
```

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# 7.3.5. JindoFS OSS Credential Provider使用说明

Smart dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

# 配置JindoFS OSS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入smartdata-site页面。
  - i. 单击**配置**页签。
  - ii. 在**服务配置**区域,单击smartdata-site页签。
- 3. 在smartdata-site页签,根据配置方式修改或新增配置信息。

配置方式	描述
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数fs.jfs.cache.oss.credentials.provider,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如, com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。

配置方式	描述				
	<ul> <li>新增配置信息:</li> <li>i. 在smartdata-site页签,单击右上角的自定义配置。</li> <li>ii. 在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。</li> <li>例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> </ul>				
按照Bucket配置	<ul> <li>说明 XXX为OSS的Bucket名称。</li> <li>iii. 单击确定。</li> </ul>				

# 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。

# E-MapReduce

类型	描述
Environment Variable Credentials Provi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 按照Bucket配置

# 您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken (临时安全令 牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment VariableCredent ialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>

类型	描述
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 7.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

# 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息(Overview)

```
包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。
```

Overview					
Start Time:	Fri Oct 16 12:29:24 2020				
Status:	Active				
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)				
Node:	Live Nodes: [2], Decommission Nodes: [0]				
Version:	3.0.0				
Build No:	fa0ea608a42				

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespa	Namespace Info (1)					
Namespace:	jfs://test/					
Namespaces:	test					
Mode:	BLOCK_MODE					
Backend URI:	oss://					
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27					

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

# Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 7.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜索			全部 smartdata-site namespace client storage	
请输入			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

### 5. 重启配置。

- i. 单击右上角的操作 > 重启 Jindo Namespace Service。
- ii. 输入执行原因, 单击**确定**。
- 开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger UI添加HDFS service。

Range	Access Manager	🗅 Audit	Settings				🔒 admin
Service	lanager						
Service N	anager						🛛 Import
						-	. = -
2	HDFS		+ 🛛 🖸	HBASE	+ 22	E HIVE	+ 2 2
				emr-hbase	۲ کا 👔	emr-hive	• 7 8

iii. 配置相关参数。

参数	描述		
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。		
Username	自定义。		
Password	自定义。		
Namenode URL	输入 <i>jfs://{namespace_name}/。</i>		
Authorization Enabled	使用默认值No。		
Authentication Type	使用默认值Simple。		
dfs.datanode.kerberos.principal			
dfs.namenode.kerberos.principal			
dfs.secondary.namenode.kerberos.principal	小块山。		
Add New Configurations			

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

# 7.4. JindoTable

# 7.4.1. 开启ORC查询加速

JindoTable提供Native ORC Reader,支持查询加速。系统默认不开启加速,开启之后可以提升Spark或Presto读取ORC文件的性 能。

# 前提条件

ORC文件已存放至JindoFS或OSS。

? 说明 暂不支持HDFS加速。

# 提升Spark性能

1. 开启JindoTable ORC加速。

⑦ 说明 Spark调用读取ORC时,需要使用DataFrame或者Spark-SQLAPI来启用加速。

。 全局设置

详细请参见全局设置Spark。

○ Job级别设置

使用spark-shell或者spark-sql时可以添加Spark的启动参数。

--conf spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtens ion

作业详情请参见Spark Shell作业配置或Spark SQL作业配置。

- 2. 检查开启情况。
  - i. 登录Spark History Server UI页面。

登录详情请参见访问链接与端口。

### ii. 在Spark的SQL页面, 查看执行任务。

当出现JindoDataSourceV2Scan时,表示开启成功。否则,请排查步骤1中的操作。



# 提升Presto性能

因为Presto已经内置JindoTable ORC加速的 catalog: hive-acc ,所以您可以直接使用 catalog: hive-acc 来启用查询加 速。

#### 示例如下。

presto --server https://emr-header-1.cluster-xxx:7778/ --catalog hive-acc --schema default

⑦ 说明 emr-header-1.cluster-xxx 是emr-header-1节点的hostname。

# 全局设置Spark

- 1. 进入Spark页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Spark。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.extensions, 修改参数值 为io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtension。
- 4. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 5. 重启ThriftServer。

- i. 在右上角选择**操作 > 重启ThriftServer**。
- ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- iii. 在确认对话框中, 单击确定。

# 7.4.2. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

# 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

# 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。

### -accessStat

● 语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

### -cache

● 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。

• 示例:缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

### -uncache

- 语法
  - jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>
- 功能

```
表示删除集群本地磁盘上指定表或分区的缓存数据。
```

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partition Col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

### -archive

● 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

```
    功能
```

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

### -unarchive

语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示将归档数据转为标准存储。

-o 将归档数据转为解冻, -i 将归档数据转为低频。

示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

jindo table -unarchive -i -t db1.t2

- -status
- 语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

### 。 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

- -optimize
- 语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t dbl.tl

- -showTable
- 语法

jindo table -showTable {-t} <dbName.tableName>

功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例:展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

### -showPartition

● 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

```
jindo table -showPartition -t db1.t1 -p date=2020-10-12
```

### -listTables

- 语法
  - jindo table -listTables [-db] [dbName]
- 功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

- 示例:
  - 。 展示default 库中的表。

jindo table -listTables

○ 列出数据库db1中的表。

jindo table -listTables -db db1

- -dumpmc
- 语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} <table} <tablename> {-p} <partitionSpec> {-f} <csv|tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络T unnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-p	分区信息 。例如 pt=xxx ,多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

```
• 功能
```

表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。

• 示例:

```
• Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。
```

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord

○ Dumpmc Maxcompute表(CSV格式)至OSS存储。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv
```

# 7.4.3. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

# 前提条件

已创建表或者视图。

# 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的DataSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

### JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dataset、DataFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spark 24.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manage	ment									
New Cache										

JindoCube还提供了**spark.sql.cache.useDatabase**参数,可以针对业务方向,按不同的业务建立database,把需要建 cache的view放在这个database中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPartition**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为**true**。

# JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击集群管理页签。
  - iii. 单击待操作集群所在行的集群ID。
  - Ⅳ. 单击左侧导航栏的**访问链接与端口**。
  - v. 在**公网访问链接**页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。 Knox相关使用说明请参见Knox。
  - vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
  - vii. 单击上方的Cube Management页签。
  - viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:

Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

# ■ Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。

在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Society 24.3 Jobs Stages Storage Enviro	nment Executors SQL	Cube Management Streaming JDBC/ODBC Server	
Table and View Information	tablaNama	CubeCache	action
ssb_10	customer	c_name caddress	raw cache cube cache
ssb_10	dates	≤ c_city ≤ c_nation ⇒ c_region	raw cache cube cache
ssb_10	lineorder	c_phone c_mktsegment	raw cache cube cache
ssb_10	lineorder_flatten	p_parkey p_name p mfar	raw cache cube cache
ssb_10	p_lineorder	p_category p_brand	raw cache cube cache
ssb_10	part	_ p_color _ p_type _ p size	raw cache cube cache
ssb_10	supplier	Decontainer	raw cache cube cache
		lo_quantity	
		lo_tax	
		Cancie OK	

JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

SELECT c\_city, c\_nation, c\_region, MAX(lo\_quantity), SUM(lo\_tax)
FROM lineorder\_flatten
GROUP BY c\_city, c\_nation, c\_region;

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
MAX	MAX
MIN	MIN
AVG	COUNT, SUM
COUNT ( DIST INCT )	PRE_COUNT_DIST INCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Det ail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作,Cache表示的数据并未持久化,需要继续构建Cache,从而持久化Cache数据到HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新,需要更新Cache中的数据从而保持一致。JindoCube支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spork 2.4.3	Jobs Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	ment Build Pa	age						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>							
Optional Filter								
Column:	s_region	÷						
Filter Type:	<ul> <li>Fixed Values</li> <li>Range Values</li> </ul>							
Values:	ASIA, AMERICA		Filter view data wit	h column value eq	uals to inpu	ut values.		
	Submit							

在构建JindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。  Column:过滤字段。  Filter Type:过滤类型,支持固定值和范围值两种。  Fixed Values:指定过滤值,可以多个,以","分隔。  Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

```
SELECT * FROM lineorder_flatten
WHERE s_region == 'ASIA' OR s_region == 'AMERICA';
```

单击**Submit**,提交构建任务,返回到Cache详细页面,对应的构建任务会提交到Spark集群中执行,在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

Spork 2.4.3	Jobs Stages Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manage	ment Timer Trigger E	Build Page	e				
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Trigger Strategy:							
Start At:	2019-12-30 11:00:00						
Period:	1	Hour		\$			
Optional Step:	۵						
Step By:	<ul> <li>TimeStamp Column(Long Type)</li> <li>DateTime Column(String Type)</li> </ul>						
Column Name:	lo_orderdate \$						
DateTime Format	Such as 'yyyy-MM-dd HI If availab	le, filter would comp	oare column as Tim	estampTy	ype value.		
	Submit						

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 ■ Overwrite: 会覆盖之前曾经构建的Cache数据。 ■ Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由Spark Driver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、 构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之 释放。

3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

● 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3 Jobs	Stages Storage Environm	nent Executors SQL	Cube Management Streaming	JDBC/ODBC Server		Thr	ift JDBC/ODBC Server application UI
Cube Managemen	t						
viewDataBase	viewName	cacheName	enableRewrite	cacheType	cacheDataSize	lastUpdataTime	action
ssb_10	lineorder_flatten	lo_raw_cache	false	Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache							

○ 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 24.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manage	ment [	Detail F	Page							
Name									Value	
Database									ssb_10	
View Name									lineorder_flatten	
Cache Name									lo_raw_cache	
Enabled									true	
Cache Type									RAW	
Cache Columns									Londrawick, Damunber, D., custere, D., parkey, D., supplier, D., contrainte, D., and protecting, Lo, 201, Lo, et al. (201), Contrainter, D., Sphone, S., and C., custere, D., dataset, A., dataset, A., and M., and C., custere, D., dataset, A., dataset, A., and M., and S., and C., custere, C., dataset, C., dataset, A., and M., and S., and and S., and and S., and and S., and and S., and and S., and and S., and and and and and and and and and and	opriotity, lo_quantity, lo_extendedprice, pkeys, s_name, s_address, s_city, s_nation, onth, d_daynuminevek, d_daynuminnonth, daynmonthif, d_holidayft, d_weekdayff, ime, p_mfgr, p_category, p_brand, p_color,
Location									/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw	
Data Size									None	
Provider									PARQUET	
Partition By									lo_orderdate	
ZOrder By										
Last Update Time									2019/12/28 17:52:55	
Enabled Partition Information Show 10 ¢ entries										Search:
Path						Siz	0		Action	
								No data av	allable in table	
Showing 0 to 0 of 0 entries	3									Previous Next
Build Information     Latest cache building is fir	ished at 20	19-12-28 17	:52:55							
Build Cache Trigger	Period Build	i								

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delet**e删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ 说明 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

<pre>[0: jdbc:hive2://localhost:10001/s 19/12/30 14:08:16 WARN [main] Hiv 19/12/30 14:08:16 WARN [main] Hiv</pre>	ssb_10> desc extended lineorder_flatten; /eConf: HiveConf of name hive.enforce.bucketing does /eConf: HiveConf of name hive.support.sql11.reserved.	not exist keywords do	es not exist			
col_name	data_type	comment	+ 			
lo_orderkey	bigint	NULL	+			
lo_linenumber	bigint	NULL	Ì			
lo_custkey	int	NULL				
lo_partkey	int	NULL				
lo_suppkey	int	NULL				
lo_orderdate	int	NULL				
lo_orderpriotity	string	NULL				
lo_shippriotity	int	NULL				
lo_quantity	bigint	NULL				
lo_extendedprice	bigint	NULL				
lo_ordtotalprice	bigint	NULL				
lo_discount	bigint	NULL				
lo_revenue	bigint	NULL				
lo_supplycost	bigint	NULL				
lo_tax	bigint	NULL				
lo_commitdate	1nt	NULL				
10_shipmode	string	NULL				
s_suppkey		NULL				
s_name	string	NULL				
s_address	string	NULL				
s_city	string	NULL				
s_nation	string	NULL				
s_region	string	NULL				
d datekev	lint	NULL				
d_date	string	NULL				
d_davofweek	string	NULL				
d_duyormeek	string	NULL				
d vear	int	NULL				
d_yearmonthnum	lint	NULL				
d vearmonth	string	NULL				
d davnuminweek	lint	NULL				
d daynuminmonth	int	NULL				
d davnuminvear	lint	NULL				
d monthnuminyear	int	NULL				
d weeknuminvear	int	NULL				
d sellingseason	string	NULL				
d lastdavinweekfl	int	NULL				
d lastdavinmonthfl	int	NULL				
d_holidayfl	int	NULL				
d weekdavfl	int	NULL				
c custkev	int	NULL				
c name	string	NULL				
c address	string	NULL				
c_city	string	NULL				
c_nation	string	NULL	i de la constante de			
c_region	string	NULL	İ. Alaşı da karalışı da kar			
c_phone	string	NULL	i			
c_mktsegment	string	NULL	ĺ			
p_partkey	int	NULL	i i i i i i i i i i i i i i i i i i i			
p_name	string	NULL				
p_mfgr	string	NULL				
p_category	string	NULL				
p_brand	string	NULL				
p_color	string	NULL				
p_type	string	NULL				
p_size	int	NULL				
p_container	string	NULL				
		1				
# Detailed Table Information		1				
Lacabase	sspin	1				
Owner		1	1			
Created Time	Hadoop   Sat Dec 28 17:30:02 CST 2010	1				
	Thu lan 01 08:00:02 C31 2017	1				
Created By	Snark 2 2 or prior	1				
	VTEW	1				
View Text	SELECT `lineorder` `lo orderkey`. `lineorder` `lo l	inenumber`.	`lineorder` `lo custkev`. `lineorder` `lo partkev`. `			
lineorder``lo sunnkey`. `lineord	der``lo orderdate`.`lipeorder``lo orderpriotity`.	`lineorder`	'lo shippriotity', 'lineorder' 'lo quantity', 'lineor			
der``lo extendedprice`, `lineoro	der` `lo_ordtotalprice`, `lipeorder` `lo_discount`, `	lineorder`	'lo revenue'. 'lineorder' 'lo supplycost'. 'lineorder'			
`lo tax`. `lineorder` `lo commit	tdate`, `lineorder` `lo shinmode`, `supplier` `s supp	key`. `sunn	lier``s name`. `sunnlier``s address`. `sunnlier``s			
city`, `supplier`,`s nation`, `su	unlier`.`s region`. `supplier`.`s phone`. `dates`.`d	datekev`.	'dates', 'd date', 'dates', 'd davofweek', 'dates', 'd mo			
nth`, `dates` `d year`, `dates`	d vearmonthnum`, `dates` `d vearmonth`, `dates` `d d	_uacekey , avnuminweek	` `dates` `d davnuminmonth`, `dates` `d davnuminvear`			
, dates . d. monthnuminyear, 'dates . d. weeknuminyear, 'dates . d. sellingseason', 'dates . d. lastdayinweekfi', 'dates . d. lastdayinmonthfi'. 'dates						
. d_holidayfl', dates'. d_weekdayfl', customer'. c_custkey', customer'. c_name'. customer'. c_address'. customer'. c_nation', dates'.						
part. p brand, `part. `p color`, `part.`p type', `part.`p container` FROM `ssb 10'. `lineorder`. `ssb 10'.`sinnlier`.`ssb 10'.						
tes, ssb.10. customer, ssb.10. part						
WHERE 'lineorder'.'lo_orderdate' = 'dates'.'d_datekey' AND `lineorder'.'lo_custkey' = 'customer'.'c custkey' AND `lineorder'.'lo suppkey' = `supplier'						
.`s_suppkey` AND `lineorder`.`lo	partkey' = 'part'.'p_partkey'					
Table Properties	[transient_lastDdlTime=1577677599]	1				
Serde Library	org.apache.hadoop.hive.serde2.lazv.LazvSimpleSerDe	i	İ			
InputFormat	org.apache.hadoop.mapred.TextInputFormat	i	i i i i i i i i i i i i i i i i i i i			
OutputFormat	org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutpu	tFormat	·			
Storage Properties	[serialization.format=1]	i .				
+		+	+			
73 rows selected (0.057 seconds)						

0: jdbc:hive2://localhost:10001/ssb\_10>

### 基于lineorder\_flatten视图简单查询的执行计划如下:

_
<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:19:32 WARN [main] HiveConf; HiveConf; of name hive.enforce.buckting does not exist 20/02/00 Version Versio</pre>
19/12/30 14:19:32 WAKN [main] HiveCont: HiveCont of name hive.support.sqlil.reserved.keywords does not exist
plan
+
+- *(11) SortMergeJoin [lo_partkey#313], [p_partkey#359], Inner
:- *(8) Sort [lo_partkey#313 ASC NULLS FIRST], false, 0 : +- Exchance hashpartitioning(lo partkey#313, 200)
: +- *(7) SortMergeJoin [lo_custkey#312], [c_custkey#351], Inner
: :- *(4) Sort [lo_custkey#312 ASC NULLS FIRS]], taise, 0 : : +- Exchange hashpartitioning(lo_custkey#312, 200)
: +- *(3) BroadcastHashJoin [lo_orderdate#315], [d_datekey#334], Inner, BuildRight
: : - *(3) BroadcastHashoon [lo_suppkey#314], [s_suppkey#32/], Inner, BulldRight : : : - *(3) Filter ((isnothull(lo_suppkey#314) && isnothull(lo_crate#315)) && isnothull(lo_custkey#312)) && isnothull(lo_prat
key#313))
: : : + - Scan hive ssp_ue.lineorder [lo_orderkey#310], lo_linenumber#311, lo_custkey#312, lo_patrkey#313, lo_suppkey#314, lo_ord erdate#315, lo_orderpriotity#316, lo_shippiotity#317, lo_quantity#3181, lo_extendedprice#3191, lo_ordtotalprice#3208, lo_discount#3211, lo_revenue#32 21, lo_supplycost#3231, lo_tax#3244, lo_commitdate#325, lo_shipmode#3261, lo_tax#3194, lo_ordtotalprice#3208, lo_lo_orderkey#312, lo_supplycost#3231, lo_tax#3244, lo_cotderdedprice#311, lo_overtex#312, lo_suppkey#314, lo_orderderde#315, lo_shipmode#3261, lo_shipmode#3261, lo_suppkey#317, lo_quantity#3181, lo_orderkpriotity#3194, lo_ordtotalprice#3204, lo_lo_shipmode#3251, lo_suppkey#312,
<pre>: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) : : +- *(1) Filter isnotnull(s_suppkey#327)</pre>
: : : +- Scan hive ssb_10.supplier [s_suppkey#327, s_name#328, s_address#329, s_city#330, s_nation#331, s_region#332, s_phone# 333], HiveTableRelation `ssb_10`.`supplier`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [s_suppkey#327, s_name#328, s_address#329, s_city#330 . s_nation#331. s_region#332. s_bhone#333]
: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) : : +- *(2) Filter isnotnull(d_datekey#334)
: : +- Scan hive ssb_10.dates [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearm ont#340, d_daynuminweek#341, d_daynuminmonth#342, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweek fl#347, d_lastdayinmonthfl#348, d_holidayfl#349, d_weekdayfl#350, HiveTableRelation `ssb_10`.dates', org.apache.hadoop.hive.serde2.lazy.LazySimpleSe rDe, [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearmonth#346, d_daynuminweek#341, d_daynuminmonth#34 2, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweekfl#347, d_lastdayinmonthfl#348, d_holidayfl#349, d_weekdayfl#350]
: +- *(6) Sort [c_custkey#351 ASC NULLS FIRST], false, 0 += Forbanda herbartitionic(c_custkey#351 _ 200)
<ul> <li>+ Exchange mashparticiping(c_usikeys351, 200)</li> <li>+ +(5) Filter ((isnortull(c_region#356) &amp;&amp; (c_region#356 = ASIA)) &amp;&amp; isnortull(c_custkey#351))</li> <li>+ - Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk</li> <li>+ - Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk</li> </ul>
<pre></pre>
+- Exchange hashpartitioning(p_partkey#359, 200)
+- *(9) Filter isnotnulitp.partkey#359) +- Scan hive ssb_16. part (p_partkey#359, p_name#360, p_mfgr#361, p_category#362, p_brand#363, p_color#364, p_type#365, p_size#366, p_conta
iner#367], HiveTableRelation `ssb_10`.`part`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [p_partkey#359, p_name#360, p_mfgr#361, p_category#3 62, p_brand#363, p_color#364, p_type#365, p_size#366, p_container#367]
++ 1 row selected (0.435 seconds)
0: jdbc:hive2://localhost:10001/ssb_10>
在为line order_flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计
划,优化后的执行计划如下:
<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:17:47 INFO [main] HiveConf: Found configuration file file:/etc/scm/spark-conf-2.4.3-hadoop2.8-1.4.2/hive-site.xml 19/12/30 14:17:47 UMRN [main] HiveConf of name hive.enforce.bucketing does not exist</pre>
19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist
plan   +
== Physical Plan == Collecting 1 A
<pre>bolicitum10 project [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderdate#133, lo_orderpriotity#134, lo_ship priotity#135, lo_quantity#136L, lo_extendedprice#137L, lo_ordtotalprice#138L, lo_discount#139L, lo_revenue#148L, lo_supplycost#141L, lo_tax#142L, lo_c ommitdate#143, lo_shipmode#144, s_suppkey#145, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#150, s_phone#151, 34 more fields] + = *(1) Filter (isnothull(c_region#174) &amp;&amp; (c_region#174 = ASIA))</pre>
+- *(1) FileScan parquet [lo_orderkey#128L,lo_linenumber#129L,lo_custkey#130,lo_partkey#131,lo_suppkey#132,lo_orderpriotity#134,lo_shippriotity# 135,lo_quantity#136L,lo_extendedprice#137L,lo_ordtotalprice#138L,lo_discount#139L,lo_revenue#140L,lo_supplycost#141L.lo_tax#142L.lo_commitdate#143.lo

可以看到,优化后的执行计划省去了lineorder\_flatten视图的所有计算逻辑,直接访问HDFS中Cache的数据。

### 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性,而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据,用户需要根据查询对于数据一致性的需求,触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

# 7.4.4. JindoTable表或分区访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

### 前提条件

已创建集群,详情请参见创建集群。

### 背景信息

JindoTable支持收集访问Hive表的记录,收集的数据保存在SmartData服务的Namespace中。

Smart Dat a 3.2.x版本开始支持Spark、Hive和Presto引擎,Spark和Presto的数据收集默认是打开的,如果需要关闭,请参见关闭 <mark>热度收集</mark>。Hive的数据收集默认是关闭的,如果需要打开,请参见<mark>开启Hive热度收集</mark>。

### 数据查询

JindoTable提供了命令方式查询热度信息。

● 语法

jindo table -accessStat <-d [days]> <-n [topNums]>

days 和 topNums 为正整数。当只设置天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

JindoTable使用详情,请参见JindoTable使用说明。

### 开启Hive热度收集

1.

- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改Hive的参数值。
  - i. 在左侧导航栏,选择集群服务 > Hive。
  - ii. 在Hive服务页面, 单击配置页签。

iii. 搜索参数hive.exec.post.hooks,在参数值后追加com.aliyun.emr.table.hive.HivePostHook。

- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - i. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
  - ii. 在执行集群操作对话框, 输入执行原因。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

### 关闭热度收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。
  - o Hive服务:
    - a. 在左侧导航栏,选择集群服务 > Hive。
    - b. 在Hive服务页面, 单击配置页签。

c. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据		
配置过滤 配置接索 hive.exec.posthooks © Q	服务配置 全部   hive-site	
配置范围	hive.exec.post.hooks	k.HiveHook, com.aliyun.emr.table.hive.HivePostHook

- Spark服务:
  - a. 在左侧导航栏,选择集群服务 > Spark。
  - b. 在Spark服务页面,单击配置页签。
  - c. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSQLQueryListener。

状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置 全部   spark-defaults
spark.sql.queryExecutionListeners Q 配置范围	spark.sql.queryExecutionListeners comaliyun.emr.table.sparkSQLQueryListener

- Presto服务:
  - a. 在左侧导航栏,选择集群服务 > Presto。
  - b. 在Presto服务页面, 单击配置页签。
  - c. 搜索参数event-listener.name, 删除参数值中的内容。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - Hive服务:
    - a. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Spark服务:
    - a. 在Spark服务页面,选择右上角的操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Presto服务:
    - a. 在Presto服务页面,选择右上角的操作 > 重启All Components。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。

# 7.5. 工具集

# 7.5.1. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

# 前提条件

已创建集群,详情请参见创建集群。

# 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

### 挂载

⑦ 说明 依次在每个节点上执行挂载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

#### 返回用户在服务端配置的所有命名空间列表。

test testcache

### 2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

### 3. 创建目录。

```
mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/
```

### 4. 写入文件。

```
echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/
```

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

### 返回如下信息。

hello world

如果您想使用Python方式写入和读取文件,请参见如下示例:

### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

# 返回如下信息。

my first file This file

# 卸载

? 说明 依次在每个节点上执行卸载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 7.5.2. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见<del>登录集群</del>。

2. 执行以下命令,获取帮助信息。

jindo distcp --help

返回信息如下。

--help - Print help text --src=VALUE - Directory to copy files from --dest=VALUE - Directory to copy files to --parallelism=VALUE - Copy task parallelism --outputManifest=VALUE - The name of the manifest file --previousManifest=VALUE - The path to an existing manifest file --requirePreviousManifest=VALUE - Require that a previous manifest is present if specified --copyFromManifest - Copy from a manifest instead of listing a directory --srcPrefixesFile=VALUE - File containing a list of source URI prefixes --srcPattern=VALUE - Include only source files matching this pattern --deleteOnSuccess - Delete input files after a successful copy --outputCodec=VALUE - Compression codec for output files --groupBy=VALUE - Pattern to group input files by --targetSize=VALUE - Target size for output files --enableBalancePlan - Enable plan copy task to make balance --enableDynamicPlan - Enable plan copy task dynamically --enableTransaction - Enable transation on Job explicitly --diff - show the difference between src and dest filelist --ossKey=VALUE - Specify your oss key if needed --ossSecret=VALUE - Specify your oss secret if needed --ossEndPoint=VALUE - Specify your oss endPoint if needed --policy=VALUE - Specify your oss storage policy --cleanUpPending - clean up the incomplete upload when distcp job finish --queue=VALUE - Specify yarn queuename if needed --bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed --s3Key=VALUE - Specify your s3 key --s3Secret=VALUE - Specify your s3 Sercet --s3EndPoint=VALUE - Specify your s3 EndPoint

### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

```
jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp
```

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

# --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distop --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制/*data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

```
hdfs dfs -ls /data/incoming/hourly table/2017-02-01/03
```

### 返回信息如下。

Found 6 items							
-rw-r	2 root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst	
-rw-r	2 root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log	
-rw-r	2 root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log	
-rw-r	2 root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001	
09							
-rw-r	2 root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt	
-rw-r	2 root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt	

#### 执行以下命令,复制以log结尾的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPattern .*\.lo
g --parallelism 20
```

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-

      1
      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1
      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

#### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --deleteOnSuccess -
-parallelism 20
```

### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03
```

#### 返回信息如下:

```
      Found 6 items

      -rw-rw-rw-
      1
      938 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/000151.sst.gz

      -rw-rw-rw-
      1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log.gz

      -rw-rw-rw-
      1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log.gz

      -rw-rw-rw-
      1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/OPTIONS-000109.g

      z
      -

      -rw-rw-rw-
      1

      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-
      1

      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-
      1

      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp06.txt.gz
```

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

 none表示保存为未压缩的文件。如果文件已压缩,则Jindo DistCp会将其解压缩。

### • keep表示不更改文件压缩形态,按原样复制。

```
② 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。
```

### --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

### --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录,查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --parallel ism 20

### 执行以下命令,查看运行期间复制的文件。

```
hadoop fs -text oss://<yourBucketName>/hourly_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst
```

### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}
#### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

#### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
        Found 4 items

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

#### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

#### 例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

```
Found 8 items
-rw-r---- 2 root hadoop
                              2252 2020-04-17 20:42 /data/incoming/hourly table/2017-02-01/03/000151.sst
-rw-r---- 2 root hadoop
                              4891 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/1.log
-rw-r---- 2 root hadoop
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/2.log
            2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/emp01.txt
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp06.txt
```

执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

### --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为Dist Cp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo Dist Cp的作业分配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableBalancePlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

# --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定\_\_\_enableDynamicPlan\_来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

# --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

#### 如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的DistCp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --dest oss://<yourB
ucketName>/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis
m 20
```

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yarn队列的名称。

命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

# --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

# 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

# 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

• 使用归档( --archive ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive
--parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

# 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

#### 命令示例如下。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

# 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

```
jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your_bucket>/hourly_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com
```

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里 ,避免每次使用时填写Accesskey。

#### E-MapReduce

<configuration></configuration>
<property></property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>

#### 此时代码示例如下。

jindo distop /tmp/jindo-distop-2.7.3.jar --src s3://smartdata1/ --dest s3://smartdata1/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

# 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters
Bytes Destination Copied=11010048000
Bytes Source Read=11010048000
Files Copied=1001
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
```

```
⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的
大小可能不相等。
```

# 7.5.3. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

### 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-dist cp-<version>.jar。
  - Hadoop 2.7及后续版本,请下载jindo-dist cp-3.0.0.jar。
  - Hadoop 3.x系列版本,请下载jindo-dist cp-3.0.0.jar。

# 场景预览

Jindo Dist Cp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二:使用Jindo Dist Cp成功导完数据后,如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?

- 场景四: 成功导入HDFS数据至OSS, 数据不断增量增加, 在Dist cp过程中可能已经产生了新文件, 该使用哪些参数处理?
- 场景五:如果需要指定jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九: 如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时, 该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?
- 场景十三: 如果不想将OSS AccessKey这种参数写在命令行里, 该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret ),以及Endpoint信息,且该AccessKey具有写目标Bucket的 权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

本场景示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10

```
⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。
```

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch
```

# 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

```
Distcp Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

参数含义如下:

○ Bytes Destination Copied:表示目标端写文件的字节数大小。

- Bytes Source Read:表示源端读文件的字节数大小。
- 。 Files Copied: 表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

# 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

在<mark>场景</mark>一的基础上,如果您的Distcp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Distcp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文 件的Copy。示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table
le --dest oss://yang-hhht/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa
nifest --parallelism 20
```

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

# 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信息, 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

参数含义如下:

- --outputManifest : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如*manifest-2020-04-17.gz*,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Dist cp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest=2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

# 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid
th 6 --parallelism 10
```

# 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

# 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

#### 优化对比如下。



# 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly\_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10

# 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

# 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

# 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则, 正则表达式。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.*/([a-z]+).*.txt' --parallelism 20
```

### 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10
```

# 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

#### E-MapReduce

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在 core-site.xml中。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

# 7.5.4. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存储成本,或者加速访问数据的速度。

# 使用Jindo jfs

执行以下命令,获取帮助信息。

```
jindo jfs -help archive
-archive -i/a <path> ... :
Archive commands.
```

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

# Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p 参数可以保证本地数据不受磁盘水位清理。

### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

### Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参 见对象存储OSS的存储类型介绍。

jindo jfs -archive -i|-a <path>

-i 参数可以归档数据至OSS低频存储类型。 -a 参数可以归档数据至OSS归档存储类型。

# Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

Unarchive默认可以将数据恢复成标准存储, -i 参数可以恢复数据至低频存储类型。 -o 参数可以临时解冻归档存储类型, 使数据临时可读。

# Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

-detail 参数可以查看文件进度信息。 -sync 参数表示该命令需要同步等待分层存储任务结束才会退出。

# ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -ls2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse -rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia\_data.csv -rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia\_data.json

# 8.SmartData 3.2.x 8.1. SmartData 3.2.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS、JindoTable和相关工具集。本文介绍Smart Dat a(3.2.x)版本的更新内容。

# JindoFS OSS扩展和支持

- 支持OSS多种免密获取Token的方式,允许自定义和扩展。
- 通过阿里云TableStore实现对Rename的并发操作的互斥。
- 支持通过Delta或Hudi写入数据至OSS。

# JindoFS缓存优化

优化在Al训练场景下小文件元数据的缓存,提升元数据预加载操作和List操作的性能。

# JindoTable计算优化

- JindoTable集成了AliORC,提供Native ORC Reader。JindoTable支持Spark和Presto使用Native ORC Reader读取ORC文件,以提升计算读取性能。
- Presto支持JindoTable访问热度统计,统计Hive表访问频次。

# JindoFS生态支持

Spark写入OSS文件时, 支持配置 spark.hadoop.mapreduce.fileoutputcommitter.marksuccessfuljobs=false , 允许作业不 生成\_*SUCCESS*文件。

# 8.2. JindoFS Block模式

# 8.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

# 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

# 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。

ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过渡			服务配置		
配置搜索			全部 i smartdata-site namespace client i storage		
清瀬/				Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前jindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	描述	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	⑦ 说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。

# ⅲ. 单击确定。

- 4. 单击右上角的**保存**。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

	服务配置				
	全部   smartdata-site   namespace	client storage			
		storage.handler.threads	40		0
		storage.watermark.low.ratio	0.2		0
		storage.watermark.high.ratio	0.4		0
		storage.oss.upload.threads	20	-	0
参	数		描述		
storage.watermark.high.ratio		表示磁盘 用的磁盘:	使用量的上水位比例,每块数据盘的 空间到达上水位即会触发清理。默认	JindoFS数据目录占 值: 0.4。	
storage.watermark.low.ratio		表示使用 JindoFS数	量的下水位比例,触发清理后会自动 (据目录占用空间清理到下水位。默认	清理冷数据,将 人值:0.2。	

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

#### 2. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 8.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

# 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见使用Raft-RocksDB-Tablestore作为存储后端。

#### 单机RocksDB作为元数据服务的架构图如下所示。



# 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	â Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<b>ξ</b>		Q	全部 smartdata-site namespace client storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
    - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX

# Smart Dat a Smart Dat a 3.2.x

参数	参数说明	示例
namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使 用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.rocksdb.asyn c.ots.enabled	<ul> <li>是否开启OTS异步上传,包括:</li> <li>true</li> <li>false</li> <li>当设置为true时,需要在SmartData服务</li> <li>完成初始化前,开启OTS异步上传功能。</li> <li>⑦ 说明 如果SmartData服务已</li> </ul>	true
	完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

#### 7. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 8. 单击右上角的操作 > 启动All Components。

# 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. 准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809
```

25 jfs://test/

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 \_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。



- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。
  - 新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。
- 3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false

参数	描述	示例

	是否开启从OTS恢复元数据,包括:	
namespace.backend.rocksdb.reco very.mode	• true • false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

==== emr-header-1:8101 ==== [Recovery From OTS Status] state: FINISH total 22855 rows.

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一致
```

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat.get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

9. 重启集群。

i. 单击上方的**集群管理**页签。

ii. 在集群管理页面, 单击相应集群所在行的更多 > 重启。

# 8.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

# 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。



# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,单击**集群服务 > SmartData**。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏,单击**集群服务 > SmartData**。
  - ii. 单击配置页签。
  - iii. 在服务配置区域, 单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤 7。

#### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jf s
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com

参数	参数说明	示例
namespace.backend.raft.async.ot	是否开启OTS异步上传,包括: o true o false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
s.enabled	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 7. 单击右上角的操作 > 启动All Components。

#### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. (可选)准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。



jindo jfs -metaStatus -detail



- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

```
jindo jfs -metaStatus -detail
```

如图所示, LEADER节点的state为FINISH表示恢复完成。



7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一致
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
    1596 1482809 25 jfs://test/
# 文件可正常读取(cat, get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

# 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 8.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

# 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

# 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

# Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false

参数	描述
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	虱 💼 SmartData ∽ ●正常			
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	<u>索</u>		Q	全部 ismartdata-site namespace client istorage

3. 配置如下参数。

i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: I true:打开AuditLog功能。 I false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中,输入执行原因,单击确定。
- v. 在确认对话框中,单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击**创建规则**,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

# 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

#### 示例如下:

• 执行如下命令显示表。

show tables;

isTemporary
false

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sq	l> select * from	n audit_log_sou	urce limit :	10;				
datetime	allowed	ugi ip	ns (	cmd src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:hado	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root	:rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> se	elect * from	n audit_log	li	mit 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 10	0:50:11.924	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020-	-10-20													
2020-10-20 10	0:50:11.950	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	L:26:06.445	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020-	-10-20													
2020-10-20 11	1:26:06.469	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	<b>1</b> 1	null	2020-10	-20
2020-10-20 11	1:26:11.295	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	-10-20													
2020-10-20 11	L:26:11.320	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 11	1:26:14.368	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	-10-20													
2020-10-20 11	L:26:14.393	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 11	1:26:16.230	true ro	ot	(auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	-10-20													
2020-10-20 11	L:26:16.255	true ro	ot	(auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken: 1	.918 second	ls, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 8.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview	
Start Time:	Fri Oct 16 12:29:24 2020
Status:	Active
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)
Node:	Live Nodes: [2], Decommission Nodes: [0]
Version:	3.0.0
Build No:	fa0ea608a42

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)		
Namespace:	jfs://test/	
Namespaces:	test	
Mode:	BLOCK_MODE	
Backend URI:	oss://	
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27	

StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

Overview						
Start Time:	Fri Oct 16 1	Fri Oct 16 12:29:25 2020				
Version:	3.0.0					
Build Version:	fa0ea608a	42a5e0e4ebcdbbfc3c041fe49f8	Be82e			
Storage Lists (4)						
Directory	StorageType	Capacity	Last Eviction Time	Partition Id		
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31		
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21		
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1		

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 8.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - 。 Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。

### E-MapReduce



# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	a∰o Smart	tData ~	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 志			服务配置
配置搜	素			全部 I smartdata-site <b>namespace</b> client I storage
请蝓ノ	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager 🗅 Audit 💠 S	ettings				🔒 admin
Service Manager					
Service Manager					🛛 Import
🗁 HDFS	+22	🗁 HBASE	+ 🛛 🖸	🕞 HIVE	+ 🛛 🖸
		emr-hbase	• 7 8	emr-hive	• 2 8

iii. 配置相关参数。

参数	描述	
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}/。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal	了店兒	
dfs.namenode.kerberos.principal		
dfs.secondary.namenode.kerberos.principal	小夾司。	
Add New Configurations		

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hado op.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

# 3. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因,单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 8.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

# 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 令使用说明。

# 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。

策略名称	策略说明
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令,获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

# 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用于一些高压缩比的文件。支持以下两种压缩策略。

策略名称	策略说明
NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path> ...

其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path>: 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

# 8.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

# 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

# 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

#### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

ibin jindo jfs -dumpMetadata test-block SLF4J: CLass path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html&mu.ruple\_bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory] Sucessfully upload namespace metadata to 055.

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

```
Sucessfully upload namespace metadata to OSS.
```

# 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如, 配置的 namespace.sysinfo.oss.uri 为 oss://abc/ ,则上传的文件会在 oss://abc/metadataDump 子目录中。

参数	说明
namespace.sysinf o.oss.uri	存储Bucket和路径。
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinfo.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

← → ↑ 1 oss://emrtest/svsinfd/metadataDump te	st-block/
▲上传 + 创建目录 □	
□ 名称	类型 / 大小
2020_09_14_18_58_16	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

### 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

{		
	"type":"string",	/*INode <b>类型,</b> FILE <b>文件</b> DIRECTORY <b>目录</b> */
	"id": "string",	/*INode id*/
	"parentId" :"string",	/* <b>父节点</b> id*/
	"name":"string",	/*INode <b>名称</b> */
	"size": "int",	/*INode <b>大小,</b> bigint*/
	"permission":"int",	/*permission <b>以</b> int <b>格式存放</b> */
	"owner":"string",	/*owner <b>名称</b> */
	"ownerGroup":"string",	/*owner <b>组名称</b> */
	"mtime":"int",	/*inode <b>修改时间,</b> bigint*/
	"atime":"int",	/*inode <b>最近访问时间,</b> bigint*/
	"attributes":"string",	/* <b>文件相关属性</b> */
	"state":"string",	/*INode <b>状态</b> */
	"storagePolicy":"string	", /*存储策略*/
	"etag":"string"	/*etag*/
}		

# 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 audit\_log和fs\_image。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta data 生成的数据。

```
示例如下。
```

```
jindo-sql> show partitions fs_image;
partition
namespace=k /datetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)
```

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@er	nr-worke	r-2 had	loop]# jindo sql			والمتعادية والمستحدين								
Spark m	aster: y	arn, Ap	plication Id: d	pp										
jindo-se	ql> show	tables												
databas	e	tableN	lame isTer	porary										
default	audit_1	og	false											
default	audit_l	og_sour	ce false											
default	fs_imag	e	false											
Time tal	ken: 0.3	45 seco	nds, Fetched 3	row(s)										
jindo-se	ql> sele	ct * fr	om fs_image lin	rit 10;										
atime	attr	etag	id mtime	e name	owner o	ownerGroup	parentId	permission	size	state	storag	ePolicy	type	name
space	datetim	ie												
0			7311076005051	1899448	160308407	70081 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450819	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			1653444804190	6675495	160308407	71350 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450820	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			731107600505:	1899470	160308407	70185 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450821	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			1192276202347	9287249	160308400	69581 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450822	root	root	334790	833296
5855433	489		Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			1076984051887	2441036	160308407	73592 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450823	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			2699389986624	511354	160308406	68996 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450824	root	root	334790	833296
5855433	489		Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			1192276202347	9287307	160308406	69875 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450825	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			1546468482017	665002	160308407	72440 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450826	root	root	334790	833296
5855433	489	ø	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			1653444804190	6675460	160308407	71170 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450827	root	root	334790	833296
5855433	489		Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						
0			731107600505	1899544	160308407	70572 /tpcds	/orc/5000/we	b_returns/wr_return	ned_date_s	sk=2450828	root	root	334790	833296
5855433	489	0	Finalized	WARM	Directory	y kugou	2020_10_20	_10_50_36						

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

jindo-sql>	select count(*)	) from fs_image where	e type = "Directory"	and namespace="kugou"	and datetime="2020_10_20_10_47_14";
11837					
Time taken:	6.852 seconds,	Fetched 1 row(s)			

# 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table\_name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

```
select * from table_name limit 200;
```

示例如下。

[hive> select * from inode_metadata_test8 limit 100;						
VARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the futu				ion engine (i.e. spark,	tez) or using Hive 1.	X releases.
Query ID = root_202009						
Total jobs = 1						
Launching Job 1 out of 1						
Number of reduce tasks is set to 0 since there's no reduce operator						
Starting Job = job_1 , Tracking URL = http://emr-heade	:208	388/proxy/app				
Kill Command = /usr/lib/hadoop-current/bin/hadoop job -kill job_1599						
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0						
[2028-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0%						
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.22 se						
MapReduce Total cumulative CPU time: 1 seconds 220 msec						
Ended Job = job_:						
MapReduce Jobs Launched:						
Stage-Stage-1: Map: 1 Cumulative CPU: 1.22 sec HDFS Read: 6867 HDFS Write: 1	524 SUCCESS					
Total MapReduce CPU Time Spent: 1 seconds 220 msec						
0K						
Directory 11274334386847219714 11274334386847219713 /uttest/oss			staff 1599545017615	1599545017615	Finalized WAR	М
Directory 11274334386847219719 11274334386847219713 /uttest/oss2			staff 1599545017654	1599545017654	Finalized WAR	М
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir			staff 1599545017636	1599545017636	Finalized WAR	М
File 11274334386847219715 11274334386847219714 /uttest/oss/file1	0 420		staff 1599545017632	1599545017632	Finalized WAR	М
File 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2	0 420		staff 1599545017642	1599545017642	Finalized WAR	М
File 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3	0 420		staff 1599545017651	1599545017651	Finalized WAR	М
Directory 11274334386847219728 11274334386847219719 /uttest/oss2/dir			caojie staff 159954	15017654 1599545017654		WARM
File 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2	0 420		staff 1599545017658	1599545017658	Finalized WAR	М
File 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3	0 420		staff 1599545017666	1599545017666	Finalized WAR	М
Directory 11274334386847219713 17672823557433851905 /uttest 0		ojie staff	1599545017615 159954	15017615 Finali	ized WARM	
Time taken: 10.734 seconds, Fetched: 10 row(s)						

# 8.3. JindoFS Cache模式

# 8.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓 存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主 要介绍JindoFS的缓存模式及其使用方式。

# 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

# 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

# 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

### 配置JFS Scheme

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。

ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	滤				服务配置
配置搜	素				全部 i smartdata-site <b>namespace</b> client i storage
清揃)				Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

- i. 修改jfs.namespaces为test。
- test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。
- ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例			
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>			
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。			
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache			

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

```
重启后即可通过 jfs://test/<path_of_file> 的形式访问JindoFS上的文件。
```

# 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true, 表示启用缓存模式。

此配置无需重启SmartData服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后,Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。
在**服务配置**区域的storage页签,修改如下参数。

	<b>.</b>								
	服务配置								
	全部 smartdata-site namespace	client storage							
		storage.handler.threads	40	-	0				
		storage.watermark.low.ratio	0.2		0				
		storage.watermark.high.ratio	0.4		0				
		storage.oss.upload.threads	20		0				
1	数		描述						
storage.watermark.high.ratio			表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。						
storage.watermark.low.ratio				量的下水位比例,触发清理后会自动; 据目录占用空间清理到下水位。默认	青理冷数据,将 值:0.2。				

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅳ. 在确认对话框中, 单击确定。

## 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > Smart Dat a的配置页面,单击smart dat a-sit e页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > Smart Data的配置页面,单击namespace页签。

#### ii. 修改jfs.namespaces为test。

iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。

# 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在**服务配置**区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

#### • 在服务配置区域的smartdata-site页签,配置以下参数。

参数	参数说明
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。
fs.oss.committer.magic.enabled	② 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。

# 8.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时,E-MapReduce(简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

## 前提条件

适用环境: ECS(EMR环境外)+Hadoop+JavaSDK。

#### 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smart dat a-aliyun-jf s-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

## 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。

6. 单击**完成**。

## 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击确定。

## 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏,单击实例与镜像>实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

## 步骤四:在ECS上设置环境变量

#### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

## 步骤五:测试免密方式访问的方法

1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

#### 2. 使用Hadoop FileSystem访问OSS。

```
JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。
```

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
        while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
            Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
   }
}
```

# 8.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

# 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

## 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

```
2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x
```

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

<返回 📾 SmartData 🗸 🌢 正常				
状态	部署拓扑	配置	配置修改历史	
配置过滤			服务配置	
配置搜	素 		Q	全部 is smartdata-site namespace client is storage

#### 3. 配置如下参数。

- i. 在namespace页签, 单击右上角的自定义配置。
- ii. 在新增配置顶对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: I true:打开AuditLog功能。 I false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

- ⅲ. 单击部署客户端配置。
- iv. 在执行集群操作对话框中,输入执行原因,单击确定。
- v. 在**确认**对话框中,单击**确定**。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。

- ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- iii. 在**确认**对话框中,单击**确定**。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏,单击基础设置 > 生命周期,在生命周期单击设置。
- iv. 单击创建规则,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

## 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

- show partitions table\_name 获取所有分区。
- desc formatted table name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

#### 示例如下:

• 执行如下命令显示表。

show tables;

jindo-so	l> show tab	les;		
database	tab	LeName	isTemporary	
default	audit_log	false		
default	audit_log_s	ource	false	
default	fs_image	false		

• 执行如下命令显示分区。

show partitions audit\_log\_source;

返回信息类似如下。

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken: 0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

select \* from audit\_log\_source limit 10;

返回信息类似如下。

jindo-sql	l> select * fro	m audit_1	log_source limit	10;						
datetime	allowed	ugi	ip ns	cmd	src	dst	perm	date		
2020-10-2	20 10:50:11.924	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.	10. H. L	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null p	perm=hadoop:had	oop:rwxrv	vxr-x 2020-10	-20						
2020-10-2	20 10:50:11.950	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.	10. ALC	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null p	perm=null	2020-10-	-20							
2020-10-2	20 11:26:06.445	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null p	perm=hadoop:had	oop:rwxrv	vxr-x 2020-10	-20						
2020-10-2	20 11:26:06.469	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null p	perm=null	2020-10-	-20							
2020-10-2	20 11:26:11.295	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s c	dst=null	perm=roo	ot:root:rwxr-x	ĸ	2020-10-	-20				
2020-10-2	20 11:26:11.320	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null p	perm=null	2020-10-	-20							
2020-10-2	20 11:26:14.368	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.	100 ALC: N	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s c	dst=null	perm=roo	ot:root:rwxr-x	×	2020-10-	-20				

```
select * from audit_log limit 10;
```

返回信息类似如下。

		1 1									
jindo-sql> select * from audit_log limit 10;											
datetime allowed ugi	ip	ns	cmd	src dst	perm	date					
2020-10-20 10:50:11.924 tru	e root	(auth:SIM	PLE)	192.	kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:	rwxrwx
r-x 2020-10-20											
2020-10-20 10:50:11.950 tru	e root	(auth:SIM	PLE)	192.	kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20 11:26:06.445 tru	e root	(auth:SIM	PLE)	192.	kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:	rwxrwx
r-x 2020-10-20											
2020-10-20 11:26:06.469 tru	e root	(auth:SIM	PLE)	192.	kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20 11:26:11.295 tru	e root	(auth:SIM	PLE)	192.	kugou	getFileStatusRequest	jfs://k		null	root:re	oot:rw
xr-xx 2020-10-20											
2020-10-20 11:26:11.320 tru	e root	(auth:SIM	PLE)	192.	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20											
2020-10-20 11:26:14.368 tru	e root	(auth:SIM	PLE)	192.	kugou	getFileStatusRequest	jfs://k		null	root:re	oot:rw
xr-xx 2020-10-20											
2020-10-20 11:26:14.393 tru	e root	(auth:SIM	PLE)	192.	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20											
2020-10-20 11:26:16.230 tru	e root	(auth:SIM	PLE)	192.	kugou	getFileStatusRequest	jfs://k		null	root:re	oot:rw
xr-xx 2020-10-20											
2020-10-20 11:26:16.255 tru	e root	(auth:SIM	PLE)	192.	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20											
Time taken: 1.918 seconds.	Fetched 1	0 row(s)									

• 执行如下命令统计2020-10-20日不同命令的使用频次。

<pre>jindo-sql&gt; select cmd ,</pre>	count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)	
getFileStatusRequest	387
listFileletRequest	387
Time taken: 5 767 secon	hds Eatchad 2 row(s)

# 8.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

## 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后, MapReduce Application Master或Spark Driver执行最终的Job Commit操作时, 会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

# 在MapReduce中使用Jindo Job Committer

1. 进入YARN服务的mapred-site页签。

- i. ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的**集群管理**页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏单击集群服务 > YARN。
- vi. 单击配置页签。
- vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - ∘ Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.JindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.factory.scheme.oss**为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.jindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

# 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击**配置**页签。
  - iii. 在**服务配置**区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

```
    ⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。
```

- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。
  - 默认值为8。
- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

# 8.3.5. Credential Provider使用说明

Smart dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

### 配置JindoFS OSS Credential Provider

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smart dat a-sit e页面。
  - i. 单击**配置**页签。
  - ii. 在服务配置区域, 单击smart dat a-site页签。

## 3. 在smartdata-site页签,根据配置方式修改或新增配置信息。

配置方式	描述			
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数fs.jfs.cache.oss.credentials.provider,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。			
按照Bucket配置	<ul> <li>新增配置信息:         <ol> <li>在smartdata-site页签,单击右上角的自定义配置。</li> <li>在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。</li> <li>例如,com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider, aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider.</li> </ol> </li> <li>③ 说明 XXX为OSS的Bucket名称。</li> </ul>			
	iii. 单击确定。			

#### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。

类型	描述		
Environment VariableCredentialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>		
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>		
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。		

# 按照Bucket配置

# 您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken(临时安全令 牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment VariableCredentialsProvi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>

类型	描述
JindoCommonCredent ialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 8.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

# 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息(Overview)

```
包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。
```

Overview	
Start Time:	Fri Oct 16 12:29:24 2020
Status:	Active
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)
Node:	Live Nodes: [2], Decommission Nodes: [0]
Version:	3.0.0
Build No:	fa0ea608a42

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespa	Namespace Info (1)		
Namespace:	jfs://test/		
Namespaces:	test		
Mode:	BLOCK_MODE		
Backend URI:	oss://		
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27		

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

# Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

#### • 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 8.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

## 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限:您可以设置文件的777权限,以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



## 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜	素			全部 smartdata-site namespace client storage
请输入			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。

5. 重启配置。

```
i. 单击右上角的操作 > 重启 Jindo Namespace Service。
```

ii. 输入执行原因, 单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见概述。

ii. Ranger UI添加HDFS service。

Ranger	Access Manager	🗅 Audit	Settings				👸 admin
Service Mar	ager						
Service Man	ager						🛛 Import 🛛 🖾 Export
BH	DFS		+ 4 2	HBASE	+ 20	E HIVE	+ 2 2
				emr-hbase	Ø 8 8	emr-hive	• 8
	Drs			emr-hbase		emr-hive	

#### iii. 配置相关参数。

参数	描述
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。
Username	自定义。
Password	自定义。
Namenode URL	输入jfs://{namespace_name}/。
Authorization Enabled	使用默认值No。
Authentication Type	使用默认值Simple。
dfs.datanode.kerberos.principal	
dfs.namenode.kerberos.principal	<b>太</b> 指写
dfs.secondary.namenode.kerberos.principal	1,24,219
Add New Configurations	

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击**确定**。
- 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。
   详情请参见Ranger Usersync集成LDAP。

# 8.4. JindoTable

# 8.4.1. 开启ORC查询加速

JindoTable提供Native ORC Reader,支持查询加速。系统默认不开启加速,开启之后可以提升Spark或Presto读取ORC文件的性 能。

# 前提条件

ORC文件已存放至JindoFS或OSS。

? 说明 暂不支持HDFS加速。

# 提升Spark性能

1. 开启JindoTable ORC加速。

⑦ 说明 Spark调用读取ORC时,需要使用DataFrame或者Spark-SQLAPI来启用加速。

。 全局设置

详细请参见全局设置Spark。

○ Job级别设置

使用spark-shell或者spark-sql时可以添加Spark的启动参数。

--conf spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtens ion

作业详情请参见Spark Shell作业配置或Spark SQL作业配置。

- 2. 检查开启情况。
  - i. 登录Spark History Server UI页面。

登录详情请参见访问链接与端口。



# 提升Presto性能

因为Presto已经内置JindoTable ORC加速的 catalog: hive-acc ,所以您可以直接使用 catalog: hive-acc 来启用查询加 速。

#### 示例如下。

presto --server https://emr-header-1.cluster-xxx:7778/ --catalog hive-acc --schema default

⑦ 说明 emr-header-1.cluster-xxx 是emr-header-1节点的hostname。

# 全局设置Spark

- 1. 进入Spark页面。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Spark。
- 2. 在Spark服务页面,单击配置页签。
- 3. 搜索参数spark.sql.extensions, 修改参数值 为io.delta.sql.DeltaSparkSessionExtension,com.aliyun.emr.sql.JindoTableExtension。
- 4. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 5. 重启ThriftServer。

- i. 在右上角选择**操作 > 重启ThriftServer**。
- ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- iii. 在确认对话框中, 单击确定。

# 8.4.2. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

## 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

## 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。

### -accessStat

● 语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

#### -cache

● 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。

• 示例:缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

### -uncache

- 语法
  - jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>
- 功能

```
表示删除集群本地磁盘上指定表或分区的缓存数据。
```

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partition col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

- -archive
- 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

### -unarchive

● 语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示将归档数据转为标准存储。

-o 将归档数据转为解冻, -i 将归档数据转为低频。

• 示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

jindo table -unarchive -i -t db1.t2

- -status
- 语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

#### 。 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

- -optimize
- 语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

- -showTable
- 语法

jindo table -showTable {-t} <dbName.tableName>

功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例:展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

#### -showPartition

● 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

```
jindo table -showPartition -t db1.t1 -p date=2020-10-12
```

#### -listTables

- 语法
  - jindo table -listTables [-db] [dbName]
- 功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

- 示例:
  - 展示default库中的表。

jindo table -listTables

◦ 列出数据库db1中的表。

jindo table -listTables -db db1

- -dumpmc
- 语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} <table} <tablename> {-p} <partitionSpec> {-f} <csv|tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络T unnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-p	分区信息 。例如 pt=xxx ,多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

#### 功能

表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。

• 示例:

```
• Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。
```

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord

○ Dumpmc Maxcompute表(CSV格式)至OSS存储。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv
```

# 8.4.3. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

## 前提条件

已创建表或者视图。

### 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的DataSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

### JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dataset 、DataFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spark 2.4.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manager	ment									
New Cache										

JindoCube还提供了**spark.sql.cache.useDatabase**参数,可以针对业务方向,按不同的业务建立database,把需要建 cache的view放在这个database中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPartition**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为**true**。

## JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击集群管理页签。
  - iii. 单击待操作集群所在行的集群ID。
  - Ⅳ. 单击左侧导航栏的**访问链接与端口**。
  - v. 在**公网访问链接**页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。 Knox相关使用说明请参见Knox。
  - vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
  - vii. 单击上方的Cube Management页签。
  - viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:

Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

# ■ Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。

在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Spork 24.3 Jobs Stages S	Storage Environment Executors SQL	Cube Management Streaming JDBC/ODBC Server	
dataBase	tableName	CubeCache	action
ssb_10	customer	c_name c_address	raw cache
ssb_10	dates	c_city c_nation c_ration	raw cache cube cache
ssb_10	lineorder	c_phone c_mktsegment	raw cache cube cache
ssb_10	lineorder_flatten	p_partkey p_name	raw cache cube cache
ssb_10	p_lineorder	p_ingr	raw cache cube cache
ssb_10	part	p_color p_type	raw cache cube cache
ssb_10	supplier	p_osce	raw cache cube cache
		lo_quantity   MAX	
		lo_tax \$ SUM \$ to	
		Cancle OK	

JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

SELECT c\_city, c\_nation, c\_region, MAX(lo\_quantity), SUM(lo\_tax)
FROM lineorder\_flatten
GROUP BY c\_city, c\_nation, c\_region;

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
MAX	MAX
MIN	MIN
AVG	COUNT, SUM
COUNT ( DIST INCT )	PRE_COUNT_DIST INCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Det ail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作,Cache表示的数据并未持久化,需要继续构建Cache,从而持久化Cache数据到HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新,需要更新Cache中的数据从而保持一致。JindoCube支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spork 2.4.3	Jobs Stages Storage	e Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manage	ment Build Page						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Optional Filter							
Column:	s_region	\$					
Filter Type:	<ul> <li>Fixed Values</li> <li>Range Values</li> </ul>						
Values:	ASIA, AMERICA	Filter view data w	ith column value equ	uals to inpu	t values.		
	Submit						

在构建JindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。 Column:过滤字段。 Filter Type:过滤类型,支持固定值和范围值两种。 Fixed Values:指定过滤值,可以多个,以","分隔。 Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

SELECT \* FROM lineorder\_flatten
WHERE s\_region == 'ASIA' OR s\_region == 'AMERICA';

单击**Submit**,提交构建任务,返回到Cache详细页面,对应的构建任务会提交到Spark集群中执行,在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

Spork 2.4.3	Jobs Stages Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	ment Timer Trigger I	Build Pag	e				
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Trigger Strategy:							
Start At:	2019-12-30 11:00:00						
Period:	1	Hour		\$			
Optional Step:	۵						
Step By:	TimeStamp Column(Long Type)     DateTime Column(String Type)						
Column Name:	lo_orderdate \$						
DateTime Format	Such as 'yyyy-MM-dd HI If availat	ole, filter would com	pare column as Tirr	estampT	ype value.		
	Submit						

定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由Spark Driver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、 构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之 释放。

3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

● 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3 Jo	bs Stages Storage Env	onment Executors SQL	Cube Management Streamin	g JDBC/ODBC Server		Th	rift JDBC/ODBC Server application UI
Cube Manageme	nt						
viewDataBase	viewName	cacheName	enableRewrite	cacheType	cacheDataSize	lastUpdataTime	action
ssb_10	lineorder_flatten	lo_raw_cache	false	Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache							

○ 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 24.3	Jobs Stages	Storage Er	ivironment E	Executors	SQL Cube	Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manage	ment Detail F	Page							
Basic Cache Information	1								
Name								Value	
Database								ssb_10	
View Name								lineorder_tlatten	
Gacne Name								IO_raw_cacne	
Enabled								true	
Cache Type								Raw	
Gache Columns								lo creterery lo Jinenumber (o. zustrey, lo partey), supprey, lo creterate lo, corterprotty, lo zin (c) cretatopite (o. b) document, prevenous (o. posphycost, lo zustre), lo cormitato (o. stephymote), sup s. region, s. phone, d. datelev, d. date, d. date, d. date, d. datelev, d. prott, d. year, d. year, d. yearmonthrum, d. yearmont d. domunimiyer, d. monthrumiyera, d. wearburningera, d. stephysics, on J. statistications, d. partey c. custre, c. name, c. address, c. Ohr, c. nation, c. region, c. phone, c. mitstegment, p. particely, p. na p. phys. p. size, no container	priotity, lo_quantity, lo_extendedprice, byev, s. name, s. address, s_pt(t), s_nation, nth, d_daynuminweek, d_daynuminmonth, tayinmonthfi, d_holidayfi, d_weekdayfi, me, p_mfgr, p_category, p_brand, p_color,
Location								/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw	
Data Size								None	
Provider								PARQUET	
Partition By								lo_orderdate	
ZOrder By									
Last Update Time								2019/12/28 17:52:55	
Enabled  Partition Information Show 10 ¢ entries									Search:
Path					Size			Action	
							No data av	allable in table	
Showing 0 to 0 of 0 entries	3								Previous Next
Build Information     Latest cache building is fir	ished at 2019-12-28 17	:52:55							
Build Cache Trigger	Period Build								

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delet**e删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ 说明 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

: jdbc:hive2://localhost:10001/ 9/12/30 14:08:16 WARN [main] Hi 9/12/30 14:08:16 WARN [main] Hi	ssb_10> desc extended lineorder_flatten; veConf: HiveConf of name hive.enforce.bucketing does veConf: HiveConf of name hive.support.sql11.reserved.	not exist keywords do	es not exist
col_name	+data_type	+	+ 
lo_orderkey	+	+	+
lo_linenumber	bigint	NULL	ĺ
lo_custkey	int	NULL	İ
lo_partkey	int	NULL	
lo_suppkey	int	NULL	i i i i i i i i i i i i i i i i i i i
lo_orderdate	int	NULL	ĺ
lo_orderpriotity	string	NULL	i
lo_shippriotity	int	NULL	i
lo_quantity	bigint	NULL	i
lo_extendedprice	bigint	NULL	i
lo_ordtotalprice	bigint	NULL	i
lo_discount	bigint	NULL	
lo revenue	bigint	NULL	
lo supplycost	bigint	NULL	
lo tax	bigint	NULL	
lo commitdate	lint	NULL	
lo shipmode	string	NULL	
s suppkey	lint	NULL	
s pame	l string	NULL	
e address	etring	NULL	
e city	string	NULL	
s nation	string	NULL	
s_netion	string	NULL	
s phone	string	NULL	
s_phone	stilly	NULL	
u_ualekey	int	NULL	
u_uate	string	NULL	
a_aayotweek	string	NULL	
a_month	string	NULL	
d_year	1 int	NULL	
d_yearmonthnum	int	NULL	
d_yearmonth	string	NULL	
d_daynuminweek	int	NULL	
d_daynuminmonth	int	NULL	
d_daynuminyear	int	NULL	
d_monthnuminyear	int	NULL	İ
d_weeknuminyear	int	NULL	İ. Alaşı da karalışı da karalışı da karalışı da karalışı da karalışı da karalışı da karalışı da karalışı da kar
d_sellingseason	string	NULL	i
d_lastdayinweekfl	l int	NULL	i
d_lastdayinmonthfl	int	NULL	i
d_holidayfl	l int	NULL	i
d weekdavfl	lint	NULL	
c custkey	lint	NULL	
c name	string	NULL	
c address	l string	NULL	
c city	string	NULL	
c nation	string	NULL	
c_region	string	NULL	
c phone	etring	NULL	
c_phone	string	NULL	
c_mkcsegmenc	string	NULL	
p_parckey	inc	NULL	
p_name	l string	NULL	
p_mrgr	string	NULL	
p_category	string	NULL	
p_orang	string	NULL	
p_color	string	NULL	
p_type	string	NULL	
p_size	1 Inc	NULL	
p_container	l strruñ	NULL	
# Detailed Table T Council		1	
# Detailed Table Information	l ach 10	1	
Database	SSD_10	1	
labte	lineorder_flatten	!	
Owner	hadoop	!	
created lime	Sat Dec 28 17:30:02 CSI 2019	!	
Last Access	Thu Jan 01 08:00:00 CST 1970	1	
Created By	Spark 2.2 or prior	1	ļ
Туре	VIEW	1	
View Text	SELECT `lineorder`.`lo_orderkey`, `lineorder`.`lo_l	inenumber`,	`lineorder`.`lo_custkey`, `lineorder`.`lo_partkey`, `
ineorder`.`lo_suppkey`, `lineor er` `lo_extendedprice`, `lineor	der`.`lo_orderdate`, `lineorder`.`lo_orderpriotity`, der`.`lo_ordtotalprice`, `lineorder`.`lo_discount`.`	`lineorder`	.`lo_shippriotity`, `lineorder`.`lo_quantity`, `lineor `lo_revenue`, `lineorder` `lo_supplycost`, `lineorder`
`lo_tax`, `lineorder`.`lo_commi	tdate`, `lineorder`.`lo_shipmode`, `supplier`.`s_supp	key`, `supp	lier`.`s_name`, `supplier`.`s_address`, `supplier`.`s
ity`, `supplier`.`s_nation`, `s	upplier`.`s_region`, `supplier`.`s_phone`, `dates`.`d	_datekey`,	`dates`.`d_date`, `dates`.`d_dayofweek`, `dates`.`d_m
th`, `dates`.`d_year`, `dates`.	`d_yearmonthnum`, `dates`.`d_yearmonth`, `dates`.`d_d	aynuminweek	`, `dates`.`d_daynuminmonth`, `dates`.`d_daynuminvear
`dates`.`d_monthnuminyear`, `d	ates`.`d_weeknuminyear`, `dates`.`d_sellingseason`, `	dates`.`d_l	astdayinweekfl`, `dates`.`d_lastdayinmonthfl`, `dates
`d_holidayfl`, `dates`.`d_weekd	ayfl`, `customer`.`c_custkey`, `customer`.`c_name`, `	customer`.`	c_address`, `customer`.`c_city`, `customer`.`c_nation
`customer`.`c_region`, `custom	er`.`c_phone`, `customer`.`c_mktsegment`, `part`.`p_p	artkey`, `p	art`.`p_name`, `part`.`p_mfgr`, `part`.`p_category`,
art`.`p_brand`, `part`.`p_color	`, `part`.`p_type`, `part`.`p_size`, `part`.`p_contai	ner`FROM`	ssb_10`.`lineorder`, `ssb_10`.`supplier`, `ssb_10`.`da
es`, `ssb_10`.`customer`, `ssb	10`.`part`		
HERE `lineorder`.`lo_orderdate`	= `dates`.`d_datekey` AND `lineorder`.`lo_custkev` =	`customer`	.`c_custkey` AND `lineorder`.`lo_suppkev` = `supplier
`s_suppkey` AND `lineorder`.`lo	_partkey` = `part`.`p_partkev`		
Table Properties	[transient_lastDdlTime=1577677599]	1	
Serde Library	org.apache.hadoop.hive.serde2.lazv.LazvSimpleSerDe	i	i
InputFormat	org.apache.hadoop.mapred.TextInputFormat	i	i
OutputFormat	org.apache.hadoop.hive.gl.io.HiveIgnoreKevTextOutput	tFormat	
Storage Properties	[serialization.format=1]	1	1
	+	+	+
3 rows selected (0.057 seconds)			

0: jdbc:hive2://localhost:10001/ssb\_10>

基于lineorder\_flatten视图简单查询的执行计划如下:

<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10;</pre>	
19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.support.soll1.reserved.keywords does not exist 19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.support.soll1.reserved.keywords does not exist	
+	
plan l	
<pre></pre>	
CollectLimit 10	
++ *(11) SortMergeJoin [lo_partkey#313], [p_partkey#359], Inner	
:- *(0) SOT [10_DATKEY#JI3 ASS NULLS FINS[], Talse, 0 : +- Exchange hashpartitioning(10 partkey#JI3, 200)	
: +- *(7) SortMergeJoin [lo_custkey#312], [c_custkey#351], Inner	
: :- *(4) Sort [lo_custkey#312 ASC NULLS FIRST], false, 0	
: : + - Exchange hashpartitioning(io_custkey#aiz, 200) : : + -+ *(3) BroadcastHashDoin [lo orderdate#315], [d datekev#334], Inner, BuildRight	
: : :- *(3) BroadcastHashJoin [lo_suppkey#314], [s_suppkey#327], Inner, BuildRight	
: : : :- *(3) Filter (((isnotnull(lo_suppkey#314) && isnotnull(lo_orderdate#315)) && isnotnull(lo_custkey#312)) && isnotnull(lo_part	
Key#315)) : : : : +- Scan hive ssb 10.lineorder [lo orderkev#310L. lo linenumber#311L. lo custkev#312. lo partkev#313. lo suppkev#314. lo ord	
erdate#315, lo_orderpriotity#316, lo_shippriotity#317, lo_quantity#318L, lo_extendedprice#319L, lo_ordtotalprice#320L, lo_discount#321L, lo_revenue#32	
2L, lo_supplycost#323L, lo_tax#324L, lo_commitdate#325, lo_shipmode#326], HiveTableRelation `ssb_10'.`lineorder`, org.apache.hadoop.hive.serde2.lazy.L	
azysimpleserue, [lo_orderkey#simu, lo_linenumber#silt, lo_oustkey#sit, lo_partkey#sit, lo_suppkey#sit, lo_orderdate#sit, lo_orderpriotity#sit, lo_suppkey#sit, lo_orderate#sit, lo_orderpriotity#sit, lo_suppkey#sit,	
omitdate#25, 10_shipmode#326]	
: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))	
: : : +- *(1) Filter isnotnull(s_suppkey#327) • · · · · · · · - Scan hive ssh Na supplier (s suppkey#327, s name#328, s address#329, s city#338, s nation#331, s renion#332, s nbne#	
333], HiveTableRelation `ssb_10`. Supplier', org.apache.hadoop.hive.serde2.lazy.lazySimpleSerDe, [s_supky#327, s_name#328, s_eldress429, s_elty#330	
, s_nation#331, s_region#332, s_phone#333]	
: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) +- +(2) Eilter isnotnull(d datakew#34)	
: : +- Scan hive ssb_10.idates (d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearm	
onth#340, d_daynuminweek#341, d_daynuminmonth#342, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweek	
T1#347, d_lastdayinmonthT#348, d_holidayT1#349, d_weekdayT1#350], HiveTableRelation ssb_10. dates , org.apache.hadoop.hive.serde2.lazy.lazySimpleSe The [d databau#335 d databa#355 d databau#335 d vas#837 d vas#837 d vas#837 d vas#monthB#36 d datumingak#841 d datuminganth#36	
10 - U_uatersos, Quatersos, Quayo metersos, Quantinaso, Quatarsos, Quata	
d_weekdayf1#350]	
: ++*(6) Sort [c_custkey#351 ASC NULLS FIRST], false, 0	
: +- Exchange hashpartitioning(c_cuskey#ssi, 200) : +- *(5) Filter ((isnothull(c_cuskey#ssi, 200) & (cregion#356 = ASIA)) && isnothull(c_cuskey#351))	
: +- Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk	
tsegment#358], HiveTableRelation 'ssb_10'.`customer', org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [c_custkey#351, c_name#352, c_address#353, c	
+- Exchange hashpartitioning(p_partkey#359, 200)	
+- *(9) Filter isnotnull(p_partkey#359)	
ingr#367]. HiveTableRelation 'ssb 10', bart', org.apache.hadoon.hive.serde2.lazv.LazvSimoleSerDe. [p bartkev#359, p name#360, p fartegory#361, biveTableRelation 'ssb 10', bart', org.apache.hadoon.hive.serde2.lazv.LazvSimoleSerDe. [p bartkev#359, p name#361, p tart861, p t	
62, p_brand#363, p_color#364, p_type#365, p_size#366, p_container#367]	1
++ ++	
1 row selected (0.435 seconds) 0. idhc:hive2://localhostil0001/ssh 105	
	ī
在为Line order flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计	
[4. idhashiya2://lacalhast:10001/sch 10: avalain coloct + from linearday flatten where a varian = 188741 limit 10:	
19/12/30 14:17:47 INFO [main] HiveConf: Found configuration file file:/etc/cem/spark-conf-2.4.3-hadoop2.8-1.4.2/hive-site.xml	
19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.enforce.bucketing does not exist	
19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist	
plan	
++	
== Physical Plan ==	
++*(1) Project [lo orderkey#128L, lo linenumber#129L, lo custkey#130, lo partkey#131, lo suppkey#132, lo orderdate#133, lo orderpriotity#134, lo ship	
priotity#135, lo_quantity#136L, lo_extendedprice#137L, lo_ordtotalprice#138L, lo_discount#139L, lo_revenue#140L, lo_supplycost#141L, lo_tax#142L, lo_c	
ommitdate#143, lo_shipmode#144, s_suppkey#145, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#150, s_phone#151, 34 more fields]	
+ +(1) FileS (shothoft(c_top)(0)#2/4) as (c_top)(0)#1/4 = ASLA) + +(1) FileScan parquet [lo orderprive] to linenumber129(.lo custkey#130,lo partkey#131,lo suppkey#132,lo orderprive)#144.lo shipprive)#144.lo	
135, lo_quantity#136L, lo_extendedprice#137L, lo_ordtotalprice#138L, lo_discount#139L, lo_revenue#140L, lo_supplycost#141L, lo_tax#142L, lo_commitdate#143, lo_	
shipmode#144, s_suppkey#146, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#150, s_phone#151, d_datekey#152, 34 more fields] Batched: true,	
rumat: requet, Lucation: inmemoryFileIndex[/user/nive/warenouse/ssg_ive_uACHE/lineorder_fileTten/raw], PartItionCount: 2406, PartItionFilters: [], Pu shedFilters: [IsNotNull(c region), Equalfo(c region, ASIA), ReadSchema: structlo orderKavibigint. to linenumberibigint. In custavint in carter	
o_suppkey:int,lo_or	
+	

## 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性, 而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据, 用户需要根据查询对于数据一致性的需求, 触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

# 8.4.4. JindoTable表或分区访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

# 前提条件

已创建集群,详情请参见创建集群。

## 背景信息

JindoTable支持收集访问Hive表的记录,收集的数据保存在SmartData服务的Namespace中。

Smart Dat a 3.2.x版本开始支持Spark、Hive和Presto引擎,Spark和Presto的数据收集默认是打开的,如果需要关闭,请参见关闭 <mark>热度收集</mark>。Hive的数据收集默认是关闭的,如果需要打开,请参见<mark>开启Hive热度收集</mark>。

#### 数据查询

JindoTable提供了命令方式查询热度信息。

● 语法

jindo table -accessStat <-d [days]> <-n [topNums]>

days和 topNums为正整数。当只设置天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

JindoTable使用详情,请参见JindoTable使用说明。

### 开启Hive热度收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改Hive的参数值。
  - i. 在左侧导航栏, 选择集群服务 > Hive。
  - ii. 在Hive服务页面, 单击配置页签。

iii. 搜索参数hive.exec.post.hooks,在参数值后追加com.aliyun.emr.table.hive.HivePostHook。

- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 7. 重启服务。
  - i. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
  - ii. 在执行集群操作对话框,输入执行原因。
  - iii. 单击确定。
  - iv. 在确认对话框中, 单击确定。

### 关闭热度收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。
  - 。 Hive服务:
    - a. 在左侧导航栏,选择集群服务 > Hive。
    - b. 在Hive服务页面, 单击配置页签。

c. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据	
配置过滤 配置接索 hive.exec.post.hooks Q	服务配置 全部   hive-site
配置范围	hive.exec.post.hooks kHiveHook comalyun.em.table.hive.HivePostHook

#### • Spark服务:

- a. 在左侧导航栏,选择集群服务 > Spark。
- b. 在Spark服务页面,单击配置页签。
- c. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSQLQueryListener。

状态 部署拓扑 配置 配置修改历史	
配置过滤	服务配置
配置搜索	全部 spark-defaults
spark.sql.queryExecutionListeners 🛛 🛇 🔍	
配置范围	sparksql.queryExecutionListeners comaliyunemr.table.sparkSparkSQLQueryListener

- Presto服务:
  - a. 在左侧导航栏,选择集群服务 > Presto。
  - b. 在Presto服务页面, 单击配置页签。
  - c. 搜索参数event-listener.name, 删除参数值中的内容。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - Hive服务:
    - a. 在Hive服务页面,选择右上角的操作 > 重启HiveServer2。
    - b. 在执行集群操作对话框,输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Spark服务:
    - a. 在Spark服务页面,选择右上角的操作 > 重启ThriftServer。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - Presto服务:
    - a. 在Presto服务页面,选择右上角的操作 > 重启All Components。
    - b. 在执行集群操作对话框, 输入执行原因。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。

# 8.5. 工具集

# 8.5.1. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

## 前提条件

已创建集群,详情请参见创建集群。

## 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

### 挂载

⑦ 说明 依次在每个节点上执行挂载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

## 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

#### 返回用户在服务端配置的所有命名空间列表。

test testcache

### 2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

#### 3. 创建目录。

```
mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/
```

#### 4. 写入文件。

```
echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/
```

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

如果您想使用Python方式写入和读取文件,请参见如下示例:

#### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

## 返回如下信息。

my first file This file

## 卸载

⑦ 说明 依次在每个节点上执行卸载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 8.5.2. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

### 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见<sup>登录集群</sup>。

2. 执行以下命令,获取帮助信息。

jindo distcp --help

返回信息如下。

```
--help
              - Print help text
                 - Directory to copy files from
--src=VALUE
--dest=VALUE
                        - Directory to copy files to
--parallelism=VALUE
                          - Copy task parallelism
--outputManifest=VALUE
                          - The name of the manifest file
--previousManifest=VALUE - The path to an existing manifest file
--requirePreviousManifest=VALUE - Require that a previous manifest is present if specified
--copyFromManifest - Copy from a manifest instead of listing a directory
--srcPrefixesFile=VALUE - File containing a list of source URI prefixes
--srcPattern=VALUE - Include only source files matching this pattern
--deleteOnSuccess - Delete input files after a successful copy
--outputCodec=VALUE - Compression codec for output files
--groupBy=VALUE - Pattern to group input files by
--targetSize=VALUE - Target size for output files
--enableBalancePlan - Enable plan copy task to make balance
--enableDynamicPlan - Enable plan copy task dynamically
--enableTransaction - Enable transation on Job explicitly
--diff - show the difference between src and dest filelist
--ossKey=VALUE - Specify your oss key if needed
--ossSecret=VALUE - Specify your oss secret if needed
--ossEndPoint=VALUE - Specify your oss endPoint if needed
--policy=VALUE - Specify your oss storage policy
--cleanUpPending - clean up the incomplete upload when distcp job finish
--queue=VALUE - Specify yarn queuename if needed
--bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed
--s3Key=VALUE - Specify your s3 key
--s3Secret=VALUE - Specify your s3 Sercet
--s3EndPoint=VALUE - Specify your s3 EndPoint
```

#### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

```
jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp
```

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

### --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制/*data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Found 6 items	3					
-rw-r	2 root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2 root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2 root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2 root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09						
-rw-r	2 root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2 root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,复制以log结尾的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPattern .*\.lo
g --parallelism 20
```

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-

      1
      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

#### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --deleteOnSuccess -
-parallelism 20
```

### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03
```

#### 返回信息如下:

```
      Found 6 items

      -rw-rw-rw-1
      938 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/000151.sst.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/0PTIONS-000109.g

      z
      -rw-rw-rw-1

      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz
```

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

 none表示保存为未压缩的文件。如果文件已压缩,则jindo DistCp会将其解压缩。
#### • keep表示不更改文件压缩形态,按原样复制。

② 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。

### --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

### --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录,查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --parallel ism 20

### 执行以下命令,查看运行期间复制的文件。

```
hadoop fs -text oss://<yourBucketName>/hourly_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst
```

### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

### 返回信息如下。

```
        Found 4 items

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20

### 查看folders.txt文件的内容。

cat folders.txt

### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

### 例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

### 返回信息如下。

```
Found 8 items
-rw-r---- 2 root hadoop
                              2252 2020-04-17 20:42 /data/incoming/hourly table/2017-02-01/03/000151.sst
-rw-r---- 2 root hadoop
                              4891 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/1.log
-rw-r---- 2 root hadoop
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/2.log
            2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r---- 2 root hadoop
                                4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/emp01.txt
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp06.txt
```

### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --targetSize=10 --g
roupBy='.\*/([a-z]+).\*.txt' --parallelism 20

### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

# --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为Dist Cp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo Dist Cp的作业分配计划,以达到更好的Dist Cp性能。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --enableBalancePlan
--parallelism 20

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

# --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定\_\_\_enableDynamicPlan\_来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

# --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

#### 如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的DistCp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --dest oss://<yourB
ucketName>/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis
m 20
```

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

### --queue

您可以使用--queue来指定本次DistCp任务所在Yarn队列的名称。

命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

# --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

# 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

# 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

• 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive
--parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

# 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

# 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

# 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

### 代码示例如下。

jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里 ,避免每次使用时填写Accesskey。

# Smart Dat a Smart Dat a 3.2.x

<configuration></configuration>
<property></property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>

### 此时代码示例如下。

jindo distop /tmp/jindo-distop-2.7.3.jar --src s3://smartdata1/ --dest s3://smartdata1/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

# 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_NAP=0

WRONG_REDUCE=0
```

```
⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的
大小可能不相等。
```

# 8.5.3. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

## 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-distcp-<version>.jar。
  - Hadoop 2.7及后续版本,请下载jindo-dist cp-3.0.0.jar。
  - Hadoop 3.x系列版本,请下载jindo-dist cp-3.0.0.jar。

## 场景预览

Jindo DistCp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二:使用Jindo Dist Cp成功导完数据后,如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?

- 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Dist cp过程中可能已经产生了新文件,该使用哪些参数处理?
- 场景五:如果需要指定Jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六: 当通过低频或者归档形式写入OSS, 该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九: 如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时, 该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?
- 场景十三: 如果不想将OSS AccessKey这种参数写在命令行里, 该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret),以及Endpoint信息,且该AccessKey具有写目标Bucket的 权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

本场景示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10

```
⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。
```

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch
```

# 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

参数含义如下:

○ Bytes Destination Copied:表示目标端写文件的字节数大小。

- 。 Bytes Source Read:表示源端读文件的字节数大小。
- 。 Files Copied: 表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

# 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

在<mark>场景</mark>一的基础上,如果您的Distcp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Distcp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文 件的Copy。示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table
le --dest oss://yang-hhht/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa
nifest --parallelism 20
```

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

# 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信息, 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

参数含义如下:

- --outputManifest : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如*manifest-2020-04-17.gz*,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Distcp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife
st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest=2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

# 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid
th 6 --parallelism 10
```

# 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

# 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

### 优化对比如下。



### • 文件总体均衡,大小差不多情况。

### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

### 优化对比如下。



# 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly\_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10

# 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

# 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

# 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则, 正则表达式。

### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.*/([a-z]+).*.txt' --parallelism 20
```

### 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10
```

# 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

### Smart Dat a Smart Dat a 3.2.x

```
<configuration>
configuration>
 cname>fs.jfs.cache.oss-accessKeyId</name>
        <value>xxx</value>
</property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
</property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
</property>
        <name>fs.jfs.cache.oss-endpoint</name>
        <value>oss-cn-xxx.aliyuncs.com</value>
</property>
</configuration>
```

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

```
<configuration>
<configuration>
cname>fs.s3a.access.key</name>
<value>xxx</value>
</property>
cname>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
configuration>
```

# 8.5.4. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存 储成本,或者加速访问数据的速度。

# 使用Jindo jfs

执行以下命令,获取帮助信息。

```
jindo jfs -help archive
-archive -i/a <path> ... :
Archive commands.
```

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

# Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p 参数可以保证本地数据不受磁盘水位清理。

### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

# Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参 见对象存储OSS的存储类型介绍。

jindo jfs -archive -i|-a <path>

-i 参数可以归档数据至OSS低频存储类型。 -a 参数可以归档数据至OSS归档存储类型。

# Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

Unarchive默认可以将数据恢复成标准存储, -i 参数可以恢复数据至低频存储类型。 -o 参数可以临时解冻归档存储类型, 使数据临时可读。

# Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

-detail 参数可以查看文件进度信息。 -sync 参数表示该命令需要同步等待分层存储任务结束才会退出。

# ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -ls2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse -rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia\_data.csv -rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia\_data.json

# 9.SmartData 3.1.x 9.1. SmartData 3.1.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS,JindoT able和相关工具集。本文介绍Smart Dat a(3.1.x)版本的更新内容。

# 背景信息

Smart Data 3.1.x版本使用时,限制信息如下:

- JindoFS Cache模式支持元数据缓存,修改met a-cache开关,即可启用缓存模式,但仅建议在训练场景下打开使用,不建议在 分析场景下使用(避免因配置使用不当导致跟其他写入路径出现不同步的情况)。
- JindoFS Namespace名称,仅可使用字母、数字和中划线(-)。
- Jindo DistCp目前支持的大文件最大不能超过78 GB。
- JindoFS Block模式虽然支持checksum功能,但Jindo DistCp暂不支持checksum功能。

### 功能变更

- JindoFS存储优化
- JindoFS缓存优化
- JindoTable计算优化
- JindoManager系统管理
- JindoTools工具集
- JindoFS生态支持

# JindoFS存储优化

- 支持文件的checksum功能,对齐开源HDFS checksum相关接口,支持MD5MD5CRC和COMPOSITE\_CRC两种算法;并且针对 MD5MD5CRC算法实现了可传入block size的扩展接口以及支持相应的Shell命令,从而能够更好地支持和HDFS文件的比对。
- 文件透明压缩功能,支持对目录设置压缩策略,对目录下新写入的文件数据块进行压缩后存储到OSS后端存储上,对于一些高压缩比的数据,可以大幅节省存储空间以及读写数据量。
- 支持写文件flush语义,调用flush接口后能够保证文件数据持久化到当前位置,并且可读。
- 解决了 hadoop fs -1s -R 命令在文件目录层级深,目录很多的情况下,出现由于线程处于等待状态致使命令无法执行的问题。
- 增强了 hadoop fs -stat 命令, 支持显示atime和privilege等。
- 增加了Jindo HDFS客户端路径改写功能,以减少集群迁移时修改路径的工作量。
   详情请参见改写Jindo HDFS客户端路径。

# JindoFS缓存优化

- 针对机器学习训练场景提供小文件缓存优化,大幅提升海量小文件的缓存效率和读取性能。
- 提供小文件目录预加载 cache 命令, 大幅提升预加载效率。
- 支持数据缓存自动触发功能,您可以通过设置需要跟踪的目标目录以及时间间隔,每隔相应的时间间隔,系统自动发现用户目录下的新增文件,并自动触发Cache操作。

# JindoTable计算优化

- JindoTable Dump TF格式支持二维数组。
- Jindo mc dump支持Gzip压缩,可以使用 -c 参数。

## JindoManager系统管理

增加了JindoManager服务,集中负责Jindo系统的运维管理以及状态监控等附加功能,提供了Web UI服务,以及查看各项Jindo系 统状态。

# JindoTools工具集

● Jindo DistCp工具针对小文件优化了Job Commiter的逻辑,大幅减少OSS的请求次数,提升大量小文件情况下DistCp的性能。

• Jindo Dist Cp工具优化了文件分批,实现了更加合理的分批策略,提升了整体性能。

# JindoFS生态支持

- Flink流式作业可恢复性地写入JindoFS,支持Block与Cache两种模式。结合可重发的数据源(例如Kafka),可以实现 Exact ly\_Once语义。
- Flink实现熵注入功能。流式作业写入OSS或jindoFS时(Block与Cache两种模式均可),支持写入路径的熵注入(entropy injection)功能,即可以使用随机字符串匹配替换路径中的特定部分。该功能有利于提高写入效率。

详情请参见支持Flink可恢复性写入JindoFS或OSS。

• 支持jindoFS Tensorflow Connector,通过实现Tensorflow Filesystem,支持原生的Tensorflow IO接口。支持Tensorflow 1.15及后续版本和Tensorflow 2.3后续版本。

# 9.2. JindoFS Block模式

# 9.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

# 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

# 配置使用方式

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。

### ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	● 正常		
状态	部署拓扑	配置	配置修改历史		
配置过	过滤			服务配置	
配置搜	索 λ		Q	全部 I smartdata-site	namespace client storage

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

### ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	描述	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	⑦ 说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。

### iii. 单击确定。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

服务配置			
全部 i smartdata-site i namespace	client storage		
	storage.handler.threads	40	0
	storage.watermark.low.ratio	0.2	0
	storage.watermark.high.ratio	0.4	0
	storage.oss.upload.threads	20	0
参数		描述	
storage.watermark.high.ratio		表示磁盘使用量的上水位 用的磁盘空间到达上水位	比例,每块数据盘的JindoFS数据目录占 即会触发清理。默认值:0.4。
storage.watermark.low.ratio		表示使用量的下水位比例 JindoFS数据目录占用空间	,触发清理后会自动清理冷数据,将 ]清理到下水位。默认值:0.2。

```
⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。
```

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 9.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

# 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见<mark>使用Raft-RocksDB-</mark> Tablestore作为存储后端。

单机RocksDB作为元数据服务的架构图如下所示。



# 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。

### ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜	素			全部 smartdata-site <b>namespace</b> client storage
清諭/			Q	

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中,输入执行原因,开启自动更新配置。
  - ⅲ. 单击**确定**。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。 ? 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	true

- 7. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 8. 单击右上角的操作 > 启动All Components。

# 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. 准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809 25 jfs://test/
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ jindo jfs -metaStatus
<pre>==== emr-header-1:8101 ====</pre>
OtsUploader: _synced=1
[RocksDB Row Counts of each CF(Table)]

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。

3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: ° true ° false	true

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。



(可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。
 此时的集群为恢复模式,也是只读模式。

### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/

1596 1482809 25 jfs://test/

# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

# 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: o true o false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 9.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

## 前提条件

- 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。
  - ? 说明 需要开启事务功能。
- 创建3 Master的EMR集群,详情请参见创建集群。

🖾 高可用	高可用: 🕜		
	部署方式: 🕜	🔿 2 Master	
	0	3 Master	查看服务部署
? 说明	如果没有部署)	方式,请提交	丁单处理。

# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。



元数据服务-多机Raft-RocksDB-Tablestore+HA如下图所示。

# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏, 单击集群服务 > Smart Data。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏, 单击**集群服务 > SmartData**。
  - ii. 单击**配置**页签。
  - iii. 在**服务配置**区域,单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft

参数	描述	示例
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤
 7。

### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。 ⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	true

- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

7. 单击右上角的操作 > 启动All Components。

### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
(文件夹个数) (文件个数)
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 \_\_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail

[RaftPeerImpl]
peer id:
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: N0 stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
<pre>vote_timer: timeout(5000ms) STOPPED</pre>
<pre>stepdown_timer: timeout(5000ms) SCHEDULING(in 2335ms)</pre>
snapshot_timer: timeout(3600000ms) SCHEDULING(in 150305ms)
storage: [1, 624625]
disk_index: 624625
known_applied_index: 624625
last_log_id: (index=624625,term=2)
first_index_pinned: 624625
state_machine: Idle
last_committed_index: 624625
last_snapshot_index: 0
last_snapshot_term: 0
snapshot_status: IDLE
replicator_25769803789@ext_index=624626 flying_append_entries_size=0 idle hc=2301 ac=624261 ic=0
replicator_329853488332 next_index=624626 flying_append_entries_size=0 idle hc=2301 ac=623564 ic=0
OtsUploader: _lastStopIndex=624624, _synced=1
Flacker DR Down Country of arch (ECTAHIA)]

iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。

### 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

### 3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus -detail

如图所示, LEADER节点的state为FINISH表示恢复完成。

```
[RaftPeerImpl]
peer_id: _____:8103:0
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: NO stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
vote_timer: timeout(5000ms) STOPPED
stepdown_timer: timeout(5000ms) SCHEDULING(in 3382ms)
snapshot_timer: timeout(600000ms) SCHEDULING(in 474855ms)
storage: [1, 153]
disk_index: 153
known_applied_index: 153
last_log_id: (index=153,term=2)
first_index_pinned: 1
state_machine: Idle
last_committed_index: 153
last_snapshot_index: 1
last_snapshot_term: 2
snapshot_status: IDLE
replicator_1116691496965@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0 replicator_3311419785217@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0
 [Recovery From OTS Status]
 state: FINISH
total rows: 1484409
table `jfs_block_test` 2 rows.
table `jfs_namespace_cache_ns` 1 rows.
table `jfs_namespace_test` 1484406 rows.
```

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

# 9. 重启集群。

- i. 单击上方的**集群管理**页签。
- ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 9.2.4. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

# 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

# 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面, 单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。

- i. 单击**配置**页签。
- ii. 单击namespace。

< 返回	💼 Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	土滤			服务配置
配置搜	索			全部 smartdata-site namespace client storage

- 3. 配置如下参数。
  - i. 在namespace页签, 单击右上角的自定义配置。
  - ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

# iii. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则,在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

### • jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

• show partitions table\_name 获取所有分区。

● desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

### 示例如下:

### • 执行如下命令显示表。

show tables;

jindo-sql> show	tables;	
database	tableName	isTemporary
default audit_lo	g false	
default audit_lo	g_source	false
default fs_image	false	

• 执行如下命令显示分区。

show partitions audit\_log\_source;

返回信息类似如下。

jindo-sql> show partitions audit\_log\_source; partition date=2020-10-20 Time taken: 0.031 seconds, Fetched 1 row(s)

• 执行如下查询数据。

select \* from audit\_log\_source limit 10;

返回信息类似如下。

jindo-so	l> select * fro	m audit_log_sour	ce limit 1	L0;				
datetime	e allowed	ugi ip	ns o	and src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 10:50:11.950	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	oop:rwxrwxr-x	2020-10-2	20				
2020-10-	20 11:26:06.469	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s	dst=null	perm=root:root:	rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=true	ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:root:	rwxr-xx	2020-10	-20			

select \* from audit\_log limit 10;

返回信息类似如下。

jindo-sql> se	lect * fro	m audit_le	og li	mit 10	;											
datetime	allowed	ugi i	ip	ns	Cff	nd sn	rc	d	lst	perm	date					
2020-10-20 10	:50:11.924	true	root	(auth:	SIMPLE)	) 19	92.			kugou	getFileStatusRequest	jfs://k	11	hadoop:h	hadoop:r	wxrwx
r-x 2020-	10-20															
2020-10-20 10	:50:11.950	true	root	(auth:	SIMPLE)	) 19	92.			kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	:26:06.445	true	root	(auth:	SIMPLE)	) 19	92.			kugou	getFileStatusRequest	jfs://k	11	hadoop:h	hadoop:r	wxrwx
r-x 2020-	10-20															
2020-10-20 11	:26:06.469	true	root	(auth:	SIMPLE)	) 19	92.			kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11	:26:11.295	true	root	(auth:	SIMPLE)	) 19	92.			kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20															
2020-10-20 11	:26:11.320	true	root	(auth:	SIMPLE)	) 19	92.			kugou	listFileletRequest	jfs://k		null	null	2020
-10-20																
2020-10-20 11	:26:14.368	true	root	(auth:	SIMPLE)	) 19	92.			kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20															
2020-10-20 11	:26:14.393	true	root	(auth:	SIMPLE)	) 19	92.			kugou	listFileletRequest	jfs://k		null	null	2020
-10-20																
2020-10-20 11	:26:16.230	true	root	(auth:	SIMPLE)	) 19	92.			kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-	10-20															
2020-10-20 11	:26:16.255	true	root	(auth:	SIMPLE)	) 19	92.			kugou	listFileletRequest	jfs://k		null	null	2020
-10-20																
Time taken: 1	918 secon	ds Fetch	ed 10	row(s	5											

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-sql> select cmd	,count(*) from	audit_log where	date="2020-10-20"	group by cmd	order by cmd;	ľ
cmd count(1)						
getFileStatusRequest	387					
listFileletRequest	387					
Time taken: 5.767 sec	onds. Fetched 2	row(s)				

# 9.2.5. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

# 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview		
Start Time:	Fri Oct 16 12:29:24 2020	
Status:	Active	
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)	
Node:	Live Nodes: [2], Decommission Nodes: [0]	
Version:	3.0.0	
Build No:	fa0ea608a42	

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)			
Namespace:	jfs://test/		
Namespaces:	test		
Mode:	BLOCK_MODE		
Backend URI:	oss://		
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27		

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview	
Start Time:	Fri Oct 16 12:29:25 2020
Version:	3.0.0
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e

# Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 9.2.6. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述			服务配置
配置搜	素			全部 smartdata-site namespace client storage
请输入	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

### 5. 重启配置。

- i. 单击右上角的操作 > 重启 Jindo Namespace Service。
- ii. 输入执行原因, 单击**确定**。
- 开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger UI添加HDFS service。

Iger  QAccess Manager	🗅 Audit	Settings				🔒 admin
rvice Manager						
ice Manager						🛛 Import
-			-		-	
B HDFS		+22	B HBASE	+ 22	▷ HIVE	+ 🖬 🖻
			emr-hbase	• 7	emr-hive	• 7 8
	vice Manager ice Manager ice Manager ice HDFS	Iger ØAccess Manager D Audit vice Manager ice Manager	Iger     Ø.Access Manager       Vice Manager       ice Manager       HDFS	Iger OAccess Manager DAudit O Settings vice Manager HDFS ← HBASE emr-hbase	Iger VAccess Manager Audit ● Settings vice Manager  HDFS  HBASE  emr-hbase	Iger VAccess Manager D Audit ● Settings vice Manager  HDFS  HBASE +22  HIVE em/hbase

iii. 配置相关参数。

参数	描述	
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}/。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal	7.博尼	
dfs.namenode.kerberos.principal		
dfs.secondary.namenode.kerberos.principal	小块山。	
Add New Configurations		

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 9.2.7. 数据管理策略

JindoFS块存储模式对文件数据管理提供了高级策略,以满足不同情形下的存储需求,主要包括存储策略(Storage Policy)和压 缩策略(Compression Policy)。本文详细介绍相关策略及其使用方式。

# 使用限制

存储策略和压缩策略都是针对目录设置的,仅对目录下新写入的文件有效。如果是设置策略之前已存在的文件或者使用rename 和mv命令移动来的文件,更新压缩策略时需要重新写入,更新存储策略需要执行分层存储命令进行归档,详情请参见分层存储命 令使用说明。

# 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略以适应不同的存储需求。支持设置以下五种存储策略。

策略名称	策略说明
AR	数据仅在OSS上有一个备份,并且使用OSS归档存储(Archive)类型 存储。
IA	数据仅在OSS上有一个备份,并且使用OSS低频访问(Infrequent Access)类型存储。
COLD	数据仅在OSS上有一个备份,并且使用OSS标准存储(Standard)类 型存储。
WARM	数据在OSS和本地分别有一个备份,本地备份能够有效的提供后续 的读取加速。 默认策略。
НОТ	数据在OSS和本地分别有一个备份,并且本地备份强制锁定,不受自 动缓存清理影响,针对一些最热的数据提供更加高优先级的加速效 果。

### OSS存储类型的详细介绍,请参见存储类型介绍。

示例,新增的文件将会以父目录所指定的Storage Policy进行存储。

• 您可以通过以下命令,设置存储类型。

jindo jfs -setStoragePolicy [-R] <StoragePolicy>(AR/IA/COLD/WARM/HOT) <path> ...

### 其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- <path> : 设置Storage Policy的路径名称。
- 您通过以下命令,获取某个目录的存储策略。

jindo jfs -getStoragePolicy <path>

# 压缩策略

JindoFS提供了Compression Policy功能,可以针对数据块进行压缩后存储,能够有效地减少存储空间和提高数据读写效率,适用 于一些高压缩比的文件。支持以下两种压缩策略。

策略名称	策略说明
------	------

NONE	不对数据块进行压缩。 默认策略。
ZSTD	对数据块使用ZSTD(Zstandard)压缩算法。

示例,新增的文件将会以父目录所指定的Compression Policy进行压缩后存储。

• 您可以通过以下命令,设置压缩类型。

jindo jfs -setCompressionPolicy [-R] <CompressionPolicy>(NONE/ZSTD) <path> ...

### 其中,涉及参数如下:

- [-R] : 递归设置该路径下的所有路径。
- o <path> : 设置Compression Policy的路径名称。
- 您通过以下命令,获取某个目录的压缩策略。

jindo jfs -getCompressionPolicy <path> ...

# 9.2.8. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信 息。

# 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为f simage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个f simage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

# 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

### 例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

(iii) jindo jfs -dumpMetadata test-block SLF4J: Elass path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ct code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-distcp-2.7.4.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html>multiple\_bindings for an explanation. SLF4J: Actual binding is of type [org.slf4j.impl.Log4]LoggerFactory] Sucessfully upload namespace metadata to OSS.

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

Sucessfully upload namespace metadata to OSS.

# 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如,配置的	namespace.sysinfo.oss.uri	为 oss://abc/ ,	则上传的文件会在	oss://abc/metadataDump	子目录中。
参数			说明		
namespace.s	sysinfo.oss.uri		存储Bucket和路径。		

### E-MapReduce

参数	说明
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinfo.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

← → ↑ 1 oss://emrtest/sysi	nfc /metadataDump test-block/
▲上传 + 创建目录 □	更多▼
□ 名称	类型 / 大小
2020_09_14_18_58_16	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

# 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

ł		
	"type":"string",	/*INode <b>类型,</b> FILE <b>文件</b> DIRECTORY <b>目录</b> */
	"id": "string",	/*INode id*/
	"parentId" :"string",	/* <b>父节点</b> id*/
	"name":"string",	/*INode <b>名称</b> */
	"size": "int",	/*INode <b>大小,</b> bigint*/
	"permission":"int",	/*permission <b>以</b> int <b>格式存放</b> */
	"owner":"string",	/*owner <b>名称</b> */
	"ownerGroup":"string",	/*owner <b>组名称</b> */
	"mtime":"int",	/*inode <b>修改时间,</b> bigint*/
	"atime":"int",	/*inode <b>最近访问时间,</b> bigint*/
	"attributes":"string",	/* <b>文件相关属性</b> */
	"state":"string",	/*INode <b>状态</b> */
	"storagePolicy":"string	3 <b>",</b> /* <b>存储策略</b> */
	"etag":"string"	/*etag*/
}		

# 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 *audit\_log*和*fs\_image*。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta

data <mark>生成的数据。</mark>

示例如下。

jindo-sql> show partitions fs_image; partition	
namespace=k /datetime=2020_10_20_10_47_14	
namespace=k /datetime=2020_10_20_10_50_36	
namespace=k /datetime=2020_10_20_10_52_06	
Time taken: 0.045 seconds, Fetched 3 row(s)	

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@em Spark ma	ir-worke ister: y	r-2 hado arn, App	op]# jin lication	do sql Id: app	,		-										
jindo-sa	l> show	tables;															
database		Cablena	me	tstempo	rary												
default	aualt_l	0g	Talse	6-1													
default	auait_L	og_sourc	e	false													
default	ts_imag	8	false														
lime tak	en: 0.5	45 secon	ds, Fetc	hed 3 ro	m(s)												
jindo-sa	l> sele	ct • fro	m fs_ima	ge limit	10;										-		
atime	attr	etag	10	mtime	name	owner	ownerGr	oup	parentId	permiss	sion	size	state	storage	Policy	type	name
space	datetim	e															
0			7311076	00505189	9448	1603084	070081	/tpcds/	orc/5000/web	_returns/wr	returned	_date_	sk=2450819	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0			1653444	80419066	75495	1603084	071350	/tpcds/	orc/5000/web	_returns/wr_	_returned	_date_	sk=2450820	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0			7311076	00505189	9470	1603084	070185	/tpcds/	orc/5000/web.	_returns/wr_	_returned	_date_	sk=2450821	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_3	10_50_36							
0			1192276	20234792	87249	1603084	069581	/tpcds/	orc/5000/web.	_returns/wr_	_returned	_date_	sk=2450822	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							an an an an an an an an an an an an an a
0			1076984	05188724	41036	1603084	073592	/tpcds/	orc/5000/web,	_returns/wr_	returned	_date_	sk=2450823	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_3	10_50_36							
			2699389	98662451	1354	1603084	068996	/tpcds/	orc/5000/web	_returns/wr_	returned	_date_	sk=2450824	root	root	334790	833296
5855433	489		Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_:	10_50_36							
			1192276	20234792	87307	1603084	069875	/tpcds/	orc/5000/web.	_returns/wr_	returned	_date_	sk=2450825	root	root	334790	833296
5855433	489		Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_:	10_50_36							
			1546468	48201766	500Z	1603084	072440	/tpcds/	orc/5000/web.	_returns/wr_	returned	_date_	sk=2450826	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0			1653444	80419066	75460	1603084	071170	/tpcds/	orc/5000/web	returns/wr	returned	_date_	sk=2450827	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0			7311076	00505189	9544	1603084	070572	/tpcds/	orc/5000/web	_returns/wr_	returned	_date_	sk=2450828	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	rv	kugou	2020_10_20_	10_50_36							
Time tak	en: 6 7	64 secon	ds Fetc	hed 10 r	(s)												

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

```
jindo-sql> select count(*) from fs_image where type = "Directory" and namespace="kugou" and datetime="2020_10_20_10_47_14";
count(1)
11837
Time taken:_6.852 seconds, Fetched 1 row(s)
```

# 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

CREATE EXTERNAL TABLE `table name` (`type` string, `id` string, `parentId` string, `name` string, `size` bigint, `permission` int, `owner` string, `ownerGroup` string, `mtime` bigint, `atime` bigint, `attr` string, `state` string, `storagePolicy` string, `etag` string) ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe' STORED AS TEXTFILE LOCATION '文件上传的OSS路径';

### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

### 示例如下。

	Consider u	sing a differen	t execution engin	e (i.e. spark, t		Hive 1.X r	eleases.				
:2088	8/proxy/app										
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 0											
2020-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0%											
524 SUCCESS											
		staff 159954	5017615 1599545	017615		WARM					
		staff 159954	5017654 1599545	017654		WARM					
			5017636 1599545	017636		WARM					
	caojie	staff 159954	5017632 1599545	017632		WARM					
		staff 159954	5017642 1599545	017642		WARM					
		staff 159954	5017651 1599545	017651		WARM					
			1599545017654	1599545017654	Fina		WARM				
			5017658 1599545	017658		WARM					
			5017666 1599545	017666		WARM					
					ed WARM						
	re versions. :2000 c 524 SUCCESS 0 511 0 511 0 511 0 511 0 511 0 420 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	re versions. Consider u :20088/proxy/app c 524 SUCCESS 0 511 Caojie 9 511 Caojie 0 420 Caojie 0 420 Caojie 0 420 Caojie 0 420 Caojie 0 420 Caojie 0 420 Caojie 0 420 Caojie 511 Caojie Staff	re versions. Consider using a differen :20080/proxy/applic: :20080/proxy/applic: c 524 SUCCESS 0 511 caojie staff 159954 0 511 caojie staff 159954 0 420 caojie staff 159954 0 420 caojie staff 159954 0 420 caojie staff 159954 0 420 caojie staff 159954 0 420 caojie staff 159954 0 420 caojie staff 159955 1 caojie staff 1599545017615	re versions. Consider using a different execution engin :200800/proxy/applic :200800/p	re versions. Consider using a different execution engine (i.e. spark, f :200808/proxy/applic: :200808/proxy/applic: c 524 SUCCESS 0 511 caojie staff 1599545017615 1599545017615 0 511 caojie staff 1599545017615 1599545017654 0 420 caojie staff 1599545017615 1599545017654 0 420 caojie staff 1599545017634 159954501763 0 420 caojie staff 1599545017634 159954501763 0 420 caojie staff 1599545017634 1599545017634 0 420 caojie staff 1599545017634 1599545017634 0 420 caojie staff 1599545017635 1599545017634 0 420 caojie staff 1599545017635 1599545017636 0 420 caojie staff 1599545017635 1599545017666 0 420 caojie staff 1599545017635 1599545017666 511 caojie staff 1599545017615 599545017666	re versions. Consider using a different execution engine (1.e. spark, tez) or using :200806/proxy/applics :200806/prox/applics :200806/prox/applics :200806/	re versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X n :200806/proxy/applic: :200806/proxy/applic: c 524 SUCCESS 0 511 caojie staff 1599545017615 1599545017615 Finalized VARM 0 511 caojie staff 1599545017654 1599545017656 Finalized VARM 0 420 caojie staff 1599545017654 1599545017652 Finalized VARM 0 420 caojie staff 1599545017654 1599545017652 Finalized VARM 0 420 caojie staff 1599545017654 1599545017652 Finalized VARM 0 420 caojie staff 1599545017654 1595545017654 Finalized VARM 0 420 caojie staff 1599545017654 1595545017654 Finalized VARM 0 420 caojie staff 1599545017654 1595545017654 Finalized VARM 0 420 caojie staff 1599545017654 1595545017656 Finalized VARM 0 420 caojie staff 1599545017654 1595545017656 Finalized VARM 0 420 caojie staff 1599545017651 1599545017656 Finalized VARM 0 420 caojie staff 1599545017651 Finalized VARM 0 420 caojie staff 1599545017651 Finalized VARM				

# 9.3. JindoFS Cache模式

# 9.3.1. Cache模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主要介绍JindoFS的缓存模式及其使用方式。

## 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

# 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。
# 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改 即可运行。

# 配置JFS Scheme

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过滤			服务配置	
<b>配置搜索</b> 法输入		Q	全部 i smartdata-site namespace client i storage	

#### 3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache

- 4. 单击确定。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

#### 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

1. 在集群服务 > Smart Dat a的配置页面, 单击client 页签。

- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。
  - 此配置无需重启SmartData服务。
- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后,Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

#### 在**服务配置**区域的storage页签,修改如下参数。

服务配置			
全部 smartdata-site namespace	client storage		
	storage.handler.threads	40	?
	storage.watermark.low.ratio	0.2	0
	storage.watermark.high.ratio	0.4	?
	storage.oss.upload.threads	20	0

参数	描述
storage.watermark.high.ratio	表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。
storage.watermark.low.ratio	表示使用量的下水位比例,触发清理后会自动清理冷数据,将 JindoFS数据目录占用空间清理到下水位。默认值:0.2。

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

#### 2. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面, 单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyId	表示存储后端OSS的AccessKey ID。
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

- ? 说明 兼容EMR-3.30.0之前版本的配置项。
- JFS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击namespace页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>	
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。	

### 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在服务配置区域的client页签, 配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

● 在**服务配置**区域的smart dat a-sit e页签,配置以下参数。

参数	参数说明	
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。	
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。	
fs.oss.committer.magic.enabled	⑦ 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。	

9.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时,E-MapReduce(简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

#### 前提条件

适用环境: ECS(EMR环境外)+Hadoop+JavaSDK。

# 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smartdata-aliyun-jfs-\*.jar*。如果要使用Spark则 需要把/*opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

#### 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击完成。

# 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击**确定**。

### 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏, 单击**实例与镜像 > 实例**。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。

付费方式	t -	连接状态	2	操作
按量 2019年7	7月26日 09:16 创建	-	看 更改实(	管理   远程连接 列规格   <u>更多</u> ▼
按量		_	购买相同配	置
2019年7	7月25日 09:47 创建		实例状态	•
按量-托 2019年	修改实例属性		实例设置	
按量	设置用户数据		密码/密钥	
2019年	授予/收回RAM角色	<u>a</u>	资源变配	•
<del>按量</del> 2019年	编辑标签		磁盘和镜像	
校旦	连接帮助		网络和安全	组 🔰
<sup>19</sup> 重 2019年	调整宿主机部署		运维和诊断	• • •
按量	调整实例所属部署	集	1	
2019年	保存为启动模板		更改实行	刘规格   更多 ▼
按量-抢	占式实例		Ê	管理   远程连接

5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

#### 步骤四:在ECS上设置环境变量

执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

# 步骤五:测试免密方式访问的方法

1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
       while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
       }
   }
}
```

# 9.3.3. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

### 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功能,所以您可以通过Shell命令分析指定的日志文件。

### 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。

#### ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	● 正常			
状态	部署拓扑	配置	配置修改历史			
配置过	滤				服务配置	
配置搜索			全部 i smartdata-site namespace client i storage			
清諭)	(			Q		

#### 3. 配置如下参数。

- i. 在namespace页签,单击右上角的自定义配置。
- ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在**确认**对话框中,单击**确定**。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则,在创建生命周期规则配置各项参数。
  - 详情请参见<mark>设置生</mark>命周期规则。
- v. 单击**确定**。

# 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。 jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

• show partitions table\_name 获取所有分区。

● desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

jindo-sal> show tables;	
database tableName	isTemporary
default audit_log false	
default audit_log_source	false
default fs_image false	

• 执行如下命令显示分区。

show partitions audit\_log\_source;

返回信息类似如下。

jindo	o-sql>	show	partitions	audit_lo	g_source;
parti	ition				
date=	=2020-1	0-20			
Time	taken:	0.03	1 seconds.	Fetched	1 row(s)

• 执行如下查询数据。

select \* from audit\_log\_source limit 10;

返回信息类似如下。

jindo-sql> sel	ect * from	n audit_log_sour	ce limit :	10;							
datetime	allowed	ugi ip	ns	cmd sr	°C	dst	perm	date			
2020-10-20 10:	50:11.924	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.	BR 815	ns=kugou	cmd=getFileStatusRequest	<pre>src=jfs:</pre>	
st=null perm=h	adoop:had	pop:rwxrwxr-x	2020-10-	20							
2020-10-20 10:	50:11.950	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.	BR 811	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/	
null perm=n	<b>111</b>	2020-10-20									
2020-10-20 11:	26:06.445	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	<pre>src=jfs:</pre>	
st=null perm=h	adoop:had	pop:rwxrwxr-x	2020-10-	20							
2020-10-20 11:	26:06.469	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/	
null perm=n	ull	2020-10-20									
2020-10-20 11:	26:11.295	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	<pre>src=jfs:</pre>	d
s dst=nu	11	perm=root:root:	rwxr-xx	20	020-10-	20					
2020-10-20 11:	26:11.320	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/	
null perm=n	ull	2020-10-20									
2020-10-20 11:	26:14.368	allowed=true	ugi=root	(auth:SIN	(PLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	<pre>src=jfs:</pre>	d
s dst=nu	11	<pre>perm=root:root:</pre>	rwxr-xx	20	020-10-	20					

select \* from audit\_log limit 10;

返回信息类似如下。

jindo-sql>	select * from	m audit_lo	og li	mit 10	;									
datetime	allowed	ugi i	.p	ns	cmd	src	dst	perm	date					
2020-10-20	10:50:11.924	true r	oot	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:b	hadoop:r	wxrwx
r-x 202	0-10-20													
2020-10-20	10:50:11.950	true r	oot	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20	11:26:06.445	true r	oot	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:b	hadoop:r	wxrwx
r-x 202	0-10-20													
2020-10-20	11:26:06.469	true r	oot	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	0-20
2020-10-20	11:26:11.295	true r	oot	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 202	0-10-20													
2020-10-20	11:26:11.320	true r	oot	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20	11:26:14.368	true r	oot	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 202	0-10-20													
2020-10-20	11:26:14.393	true r	oot	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20	11:26:16.230	true r	oot	(auth:	SIMPLE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 202	0-10-20													
2020-10-20	11:26:16.255	true r	oot	(auth:	SIMPLE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1 918 50000	de Enteha	d 10	now	5									

• 执行如下命令统计2020-10-20日不同命令的使用频次。

```
jindo-sql> select cmd ,count(*) from audit_log where date="2020-10-20" group by cmd order by cmd;
cmd count(1)
getFileStatusRequest 387
listFileletRequest 387
Time taken: 5.767 seconds, Fetched 2 row(s)
```

# 9.3.4. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

### 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后,MapReduce Application Master或Spark Driver执行最终的Job Commit操作时,会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

#### 在MapReduce中使用Jindo Job Committer

1. 进入YARN服务的mapred-site页签。

```
i.
```

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的**集群管理**页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏单击集群服务 > YARN。
- vi. 单击配置页签。
- vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。

```
。 Hadoop 2.x版本
```

```
在YARN服务的mapred-site页签,设
置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter。
```

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

 $\label{eq:compareduce.output committer.factory.scheme.oss} \texttt{b} \texttt{com.aliyun.emr.fs.oss.commit.jindoOssCommitterFactory}_{\circ}$ 

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > SmartData。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.jindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

#### 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的**spark-defaults**页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击smartdata-site页签。

5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时,Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和
 FileOutputCommitter行为一样。

- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

### 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > SmartData。
  - ii. 单击配置页签。
  - iii. 在服务配置区域, 单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。
- 默认值为8。
- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# 9.3.5. JindoFS OSS Credential Provider使用说明

Smart dat a 3.4.0及后续版本支持JindoFS OSS Credential Provider,您可以通过配置JindoFS OSS Credential Provider,将加密后的AccessKey信息添加至文件中,以避免泄露AccessKey信息。

#### 配置JindoFS OSS Credential Provider

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smartdata-site页面。
  - i. 单击**配置**页签。
  - ii. 在服务配置区域,单击smartdata-site页签。
- 3. 在smartdata-site页签,根据配置方式修改或新增配置信息。

配置方式	描述
全局方式配置(所有Bucket使用同一 种方式)	在配置搜索区域,搜索参数fs.jfs.cache.oss.credentials.provider,在参数值后追加 AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential,需要添加的参数详情,请参见全局方式配置。 例如, com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。

配置方式	描述
按照Bucket配置	<ul> <li>新增配置信息:</li> <li>i. 在smartdata-site页签,单击右上角的自定义配置。</li> <li>ii. 在新增配置项对话框中,设 置Key为fs.jfs.cache.oss.bucket.XXX.credentials.provider,Value为 com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号 (,)隔开,按照先后顺序读取Credential直至读到有效的Credential,其余需要添加的 参数详情,请参见按照Bucket配置。</li> <li>例如, com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,com. aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider。</li> </ul>
	⑦ 说明 XXX为OSS的Bucket名称。
	iii. 单击 <b>确定</b> 。

# 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击**确定**。

# 全局方式配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
TemporaryAliyunCredentialsProvider	适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。 • fs.jfs.cache.oss.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.credentials.provider: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment Variable Credentials Provi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>

类型	描述
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>需要在fs.jfs.cache.oss.credentials.provider的参数值中追加com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 需要在fs.jfs.cache.oss.credentials.provider的参数值中追 加com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 按照Bucket配置

您可以根据情况,选择不同的Provider。Provider类型如下表。

类型	描述
T emporaryAliyunCredent ialsProvider	<ul> <li>适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsProvider,并需新增以下配置:</li> <li>fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>fs.jfs.cache.oss.bucket.XXX.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>
SimpleAliyunCredentialsProvider	适合使用长期有效的AccessKey访问OSS的情况。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvider,并需新增以下配置: • fs.jfs.cache.oss.bucket.XXX.accessKeyId: OSS Bucket的AccessKey ID。 • fs.jfs.cache.oss.bucket.XXX.accessKeySecret: OSS Bucket的AccessKey Secret。
Environment Variable Credentials Provi der	<ul> <li>该方式需要在环境变量中配置以下参数。</li> <li>设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EnvironmentVariableCredentialsProvider,并需新增以下配置:</li> <li>ALIYUN_ACCESS_KEY_ID: OSS Bucket的AccessKey ID。</li> <li>ALIYUN_ACCESS_KEY_SECRET: OSS Bucket的AccessKey Secret。</li> <li>ALIYUN_SECURITY_TOKEN: OSS Bucket的SecurityToken(临时安全令牌)。</li> <li>⑦ 说明 仅配置有时效Token时需要。</li> </ul>
JindoCommonCredentialsProvider	<ul> <li>该方式为通用配置,配置后JindoOSS和JindoFS均可以使用。</li> <li>设置fs.jfs.cache.oss.credentials.provider的参数值</li> <li>为com.aliyun.emr.fs.auth.JindoCommonCredentialsProvider,并需新增以下配置:</li> <li>jindo.common.accessKeyId: OSS Bucket的AccessKey ID。</li> <li>jindo.common.accessKeySecret: OSS Bucket的AccessKey Secret。</li> <li>jindo.common.securityToken: OSS Bucket的SecurityToken(临时安全令牌)。</li> </ul>

类型	描述
EcsStsCredentialsProvider	该方式无需配置AccessKey,可以免密方式访问OSS。 设置fs.jfs.cache.oss.bucket.XXX.credentials.provider的参数值 为com.aliyun.emr.fs.auth.EcsStsCredentialsProvider。

# 9.3.6. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

# 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

您可以通过*http://emr-header-1:8104/*访问JindoFS Web UI功能。JindoFS 3.1.x版本提供总览信息(Overview)、Namespace 信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview						
Start Time:	Fri Oct 16 12:29:24 2020					
Status:	Active					
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)					
Node:	Live Nodes: [2], Decommission Nodes: [0]					
Version:	3.0.0					
Build No:	fa0ea608a42					

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Na	Namespace Info (1)							
Nan	nespace:	jfs://test/						
Nar	mespaces:	test						
Мо	de:	BLOCK_MODE						
Bac	kend URI:	oss://						
Sur	nmary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27						

• StorageService信息

包含当前集群的StorageService列表,以及对应StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

"卑击INOOEN" 应以 算有母	却个噬盆的全间使用情况。
-------------------	--------------

Overview								
Start Time:		Fri Oct 16 12:29	9:25 2020					
Version: 3.0.0								
Build Version:	9							
Storage Lists (4	Storage Lists (4)							
Directory	Stora	ідеТуре	Capacity	Last Eviction Time	Partition Id			
/mnt/d	Disk		28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31			
/mnt/d	Disk		19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21			
/mnt/d	Disk		51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1			
/mnt/d	Disk		24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11			

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 9.3.7. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - 。 Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。

#### E-MapReduce



# 启用JindoFS Unix权限

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	a∰o Smart	tData ~	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	述 志			服务配置
配置搜	素			全部 I smartdata-site <b>namespace</b> client I storage
请蝓ノ	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager 🗅 Audit 💠 S	ettings				🔒 admin
Service Manager					
Service Manager					🛛 Import
🗁 HDFS	+22	🗁 HBASE	+ 🛛 🖸	🕞 HIVE	+ 🛛 🖸
		emr-hbase	• 7 8	emr-hive	• 2 8

iii. 配置相关参数。

参数	描述	
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test 。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}/。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal		
dfs.namenode.kerberos.principal	石塘空	
dfs.secondary.namenode.kerberos.principal	小论)。	
Add New Configurations		

#### iv. 单击Add。

## 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hado op.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

# 3. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 9.4. JindoTable

# 9.4.1. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

# 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

# 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition
- -listTables
- -dumpmc

○ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partitionCol2=2,... 的格式。

- -accessStat
- 语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

- -cache
- 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。

• 示例:缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

#### -uncache

语法

jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示删除集群本地磁盘上指定表或分区的缓存数据。

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionCol1=1,partition Col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

- -archive
- 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

• 示例:指定表db1.t1缓存至本地磁盘上。

jindo table -archive -t db1.t1 -p date=2020-10-12

#### -unarchive

- 语法
  - jindo table -unarchive [-o|-i] {-t} <dbName.tableName>[-p] <partitionSpec>
- 功能

表示将归档数据转为标准存储。

• 示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

jindo table -unarchive -i -t db1.t2

- -status
- 语法

jindo table -status {-t} <dbName.tableName> [-p] <partitionSpec>

功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

○ 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

#### -optimize

语法

jindo table -optimize {-t} <dbName.tableName>

功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

#### -showTable

- 语法
  - jindo table -showTable {-t} <dbName.tableName>
- 功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例: 展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

- -showPartition
- 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

#### • 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

jindo table -showPartition -t db1.t1 -p date=2020-10-12

- -listTables
- 语法

jindo table -listTables [-db] [dbName]

功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

- 示例:
  - 。 展示default 库中的表。

jindo table -listTables

○ 列出数据库db1中的表。

jindo table -listTables -db db1

#### -dumpmc

语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} <table} <tablename> {-p} <partitionSpec> {-f} <csv|tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络Tunnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-р	分区信息 。例如 <sub>pt=xxx</sub> ,多个分区时 用英文逗号 (,) 分 开 <sub>pt=xxx</sub> ,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

功能

表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。

- 示例:
  - Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord

○ Dumpmc Maxcompute表(CSV格式)至OSS存储。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv
```

# 9.4.2. JindoTable表或分区的访问热度收集

您可以通过JindoTable表或分区的访问热度收集功能来区分冷热数据,从而节约整体的存储成本,提高缓存利用效率。

#### 数据收集

JindoTable支持收集访问Hive表的记录,目前支持的引擎有Spark和Hive。收集的数据保存在集群SmartData服务的Namespace中。

数据收集是默认打开的。如果需要关闭,请参见关闭数据收集。

#### 数据查询

JindoTable提供了命令方式查询热度信息。

● 语法

jindo table -accessStat <-d [days]> <-n [topNums]>

days 和 topNums 为正整数。当天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

功能

查询在指定时间范围内,访问最多表或分区的指定条数。

• 示例,查询近七天访问最多的表或分区的20条访问记录。

jindo table -accessStat -d 7 -n 20

JindoTable使用详情,请参见JindoTable使用说明。

### 关闭数据收集

- 1.
- 2. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- 3. 单击上方的集群管理页签。
- 4. 在集群管理页面,单击相应集群所在行的详情。
- 5. 修改参数值。

删除如下参数值中的部分内容:

- Hive服务:
  - a. 在左侧导航栏单击集群服务 > Hive。
  - b. 单击配置页签。
  - c. 单击hive-site页签。
  - d. 搜索参数hive.exec.post.hooks, 删除参数值中的com.aliyun.emr.table.hive.HivePostHook。

状态 部署拓扑 配置 配置修改历史 元数据		
配置过滤 配置搜索 hive.exec.posthooks	服务配置 全部   hive-site	
配置范围	hive.exec.post.hooks	k.HiveHook, com.aliyun.emr.table.hive.HivePostHook

- 。 Spark服务:
  - a. 在左侧导航栏单击集群服务 > Spark。
  - b. 单击配置页签。
  - c. 单击spark-defaults页签。

d. 搜索参数spark.sql.queryExecutionListeners, 删除参数值中的com.aliyun.emr.table.spark.SparkSOLOueryListener。

状态	部署拓扑	配置	配置修改历史				
配置進	<b>过滤</b> 索			服务配置 全部 spark-defaults			
sparl 配置范	k.sql.queryExect	utionListen	ers © Q		spark.sql.queryExecutionListeners	com.aliyun.emr.table.spark.SparkSQLQueryListener	

- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 7. 重启服务。
  - Hive服务:
    - a. 单击右上角的操作 > 重启 HiveServer2。
    - b. 在执行集群操作对话框,设置相关参数。
    - c. 单击确定。
    - d. 在确认对话框中, 单击确定。
  - o Spark服务:
    - a. 单击右上角的操作 > 重启 ThriftServer。
    - b. 在执行集群操作对话框,设置相关参数。
    - c. 单击确定。
    - d. 在确认对话框中,单击确定。

# 9.4.3. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

#### 前提条件

已创建表或者视图。

#### 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的DataSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

#### JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dataset、DataFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spork 2.4.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manager	nent									
New Cache										

JindoCube还提供了**spark.sql.cache.useDatabase**参数,可以针对业务方向,按不同的业务建立database,把需要建 cache的view放在这个database中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPartition**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

#### 2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为**true**。

# JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击**集群管理**页签。
  - iii. 单击待操作集群所在行的集群ID。
  - Ⅳ. 单击左侧导航栏的**访问链接与端口**。
  - v. 在**公网访问链接**页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。 Knox相关使用说明请参见Knox。
  - vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
  - vii. 单击上方的Cube Management页签。
  - viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:

- Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。
  - 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。
 在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选



JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

SELECT c\_city, c\_nation, c\_region, MAX(lo\_quantity), SUM(lo\_tax)
FROM lineorder\_flatten
GROUP BY c\_city, c\_nation, c\_region;

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
MAX	MAX
MIN	MIN
AVG	COUNT, SUM

聚合函数类型	预计算函数
COUNT (DISTINCT)	PRE_COUNT_DISTINCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Detail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作, Cache表示的数据并未持久化, 需要继续构建Cache, 从而持久化Cache数据到 HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新, 需要更新Cache中的数据从而保持一致。JindoCube 支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spork 2.4.3	Jobs Sta	iges Sto	orage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manage	ment Bui	ld Pag	е						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>								
Optional Filter	۵								
Column:	s_region		4						
Filter Type:	<ul> <li>Fixed Value</li> <li>Range Value</li> </ul>	es							
Values:	ASIA, AMERIO	CA		Filter view data wit	th column value eq	uals to inpu	ut values.		
	Submit								

在构建lindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 <ul> <li>Overwrite:会覆盖之前曾经构建的Cache数据。</li> <li>Append:会新增数据到Cache中。</li> </ul>
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。 ■ Column:过滤字段。 ■ Filter Type:过滤类型,支持固定值和范围值两种。 ■ Fixed Values:指定过滤值,可以多个,以","分隔。 ■ Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

SELECT \* FROM lineorder\_flatten
WHERE s\_region == 'ASIA' OR s\_region == 'AMERICA';

单击**Submit** , 提交构建任务, 返回到Cache详细页面, 对应的构建任务会提交到Spark集群中执行, 在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

Spork 2.4.3	Jobs Stages Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	ment Timer Trigger B	Build Page	е				
Save Mode:	Overwrite     Append						
Trigger Strategy:							
Start At:	2019-12-30 11:00:00						
Period:	1	Hour		\$			
Optional Step:							
Step By:	<ul> <li>TimeStamp Column(Long Type)</li> <li>DateTime Column(String Type)</li> </ul>						
Column Name:	lo_orderdate						
DateTime Format	Such as 'yyyy-MM-dd Hł If availab	e, filter would comp	oare column as Tir	nestampT	ype value.		
	Submit						

#### 定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 ■ Overwrite: 会覆盖之前曾经构建的Cache数据。 ■ Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由SparkDriver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之释放。
- 3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

● 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3	Jobs	Stages	Storage	Environme	nt Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server		т	hrift JDBC/ODBC Server application UI
Cube Management												
viewDataBase		viewName			acheName		enableRewrite		cacheType	cacheDataSize	lastUpdataTime	action
ssb_10		lineorder_fla	tten	b	o_raw_cache		false		Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache												

。 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 2.4.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manage	ment D	Detail F	Page							
Basic Cache Information										
Name									Value	
Database									ssb_10	
View Name									lineorder_flatten	
Cache Name									lo_raw_cache	
Enabled									true	
Cache Type									RAW	
Cache Columns									b, ordeteky, b, Inenumber, bc, custeky, b, parkey, b, suppkey, b, ordetalta, b, ordetprotity, b, al, c, ordetakyte, b, discourt, b, orverval, b, supphyces, b, back, b, committale, b, parkeynob, s, su s, region, s, phone, d, dateky, d, date, d, danydweek, d, month, d, yae, d, yearmonthrum, d, yaem (dynumilyse), c, mane, c, addetese, c, chi, c, nation, c, region, c, phone, c, mistegment, b, parkey, p, m Jype, p, zaes, orothiner	Ippriotity, Io_quantity, Io_extendedprice, ppkey, s_name, s_address, s_city, s_nation, onth, d_daynuminwoek', d_daynuminmonth, ddaynmonthif, d_holidayfi, d_weekdayfi, ame, p_mfgr, p_category, p_brand, p_color,
Location									/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw	
Data Size									None	
Provider									PARQUET	
Partition By									lo_orderdate	
ZOrder By										
Last Update Time									2019/12/28 17:52:55	
Enabled Partition Information Show 10 ¢ entries										Search:
Path						Siz	e		Action	
								No data av	valiable in table	
Showing 0 to 0 of 0 entries										Previous Next
Build Information     Latest cache building is fire	ished at 201	19-12-28 17	:52:55							
Build Cache Trigger	Period Build									

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delete**删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ 说明 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

<pre>[0: jdbc:hive2://localhost:10001/s 19/12/30 14:08:16 WARN [main] Hiv 19/12/30 14:08:16 WARN [main] Hiv</pre>	ssb_10> desc extended lineorder_flatten; /eConf: HiveConf of name hive.enforce.bucketing does /eConf: HiveConf of name hive.support.sql11.reserved.	not exist keywords do	es not exist				
col_name	data_type	comment	+				
lo_orderkey	bigint	NULL	+				
lo_linenumber	bigint	NULL					
lo_custkey	int	NULL					
lo_partkey	int	NULL					
lo_suppkey	int	NULL					
lo_orderdate	int	NULL					
lo_orderpriotity	string	NULL					
lo_shippriotity	int	NULL					
lo_quantity	bigint	NULL					
lo_extendedprice	bigint	NULL					
lo_ordtotalprice	bigint	NULL					
lo_discount	bigint	NULL					
lo_revenue	bigint	NULL					
lo_supplycost	bigint	NULL					
lo_tax	bigint	NULL					
lo_commitdate	1nt	NULL					
10_shipmode	string	NULL					
s_suppkey		NULL					
s_name	string	NULL					
s_address	string	NULL					
s_city	string	NULL					
s_nation	string						
s_region	string						
d datekev	lint	NULL					
d_date	string	NULL					
d_davofweek	string	NULL					
d_duyormeek	string	NULL					
d vear	int	NULL					
d_yearmonthnum	lint	NULL					
d vearmonth	string	NULL					
d davnuminweek	lint	NULL					
d daynuminmonth	int	NULL					
d davnuminvear	lint	NULL					
d monthnuminyear	int	NULL					
d weeknuminvear	int	NULL					
d sellingseason	string	NULL					
d lastdavinweekfl	int	NULL					
d lastdavinmonthfl	int	NULL					
d_holidayfl	int	NULL					
d weekdavfl	int	NULL					
c custkev	int	NULL					
c name	string	NULL					
c address	string	NULL					
c_city	string	NULL					
c_nation	string	NULL					
c_region	string	NULL					
c_phone	string	NULL					
c_mktsegment	string	NULL					
p_partkey	int	NULL					
p_name	string	NULL					
p_mfgr	string	NULL					
p_category	string	NULL					
p_brand	string	NULL					
p_color	string	NULL					
p_type	string	NULL					
p_size	int	NULL					
p_container	string	NULL					
# Detailed Table Information							
Lacabase	sspin						
Owner		1					
Created Time	Hadoop   Sat Dec 28 17-30-02 CST 2010						
	Thu Jan 01 08:00:02 C31 2017						
Created By	Spark 2 2 or prior						
	VTEW	1					
View Text	SELECT `lineorder` `lo orderkev`. `lineorder` `lo l	inenumber`.	`lineorder` `lo custkey`, `lineorder` `lo partkey`, `				
lineorder``lo sunnkey`. `lineord	der``lo orderdate`.`lipeorder``lo orderpriotity`.	`lineorder`	'lo shippriotity', 'lineorder' 'lo quantity', 'lineor				
der``lo extendedprice`, `lineoro	der` `lo_ordtotalprice`, `lipeorder` `lo_discount`, `	lineorder`	`lo revenue`. `lineorder` `lo sunnlycost`. `lineorder`				
`lo tax`. `lineorder` `lo commit	tdate`, `lineorder` `lo shinmode`, `supplier` `s supp	kev`. `sunn	lier``s name`.`sunnlier``s address`.`sunnlier``s				
city`, `supplier`,`s nation`, `su	unlier`.`s region`. `supplier`.`s phone`. `dates`.`d	datekev`.	`dates`.`d date`. `dates`.`d davofweek`. `dates`.`d mo				
nth`, `dates` `d year`, `dates`	d vearmonthnum`, `dates` `d vearmonth`, `dates` `d d	_uacekey ; avnuminweek	` `dates` `d davnuminmonth`, `dates` `d davnuminvear`				
nn, accos e_you, accos ayearmoninam, accos e_youmonin, actos au_adynumineer, actos augynuminennin, actos augynu . 'datas' 'd monthuminyaar'. 'datas' 'd waakuminyaar'. 'datas' 'd sacason' 'datas' 'd sactaaimaastafi' 'datas' i							
, were smontentamentyeer, weres sweennamentyeer, weres u_seriingseason, weres s_iastuayinweenti, weres s_iastuayinmonthi, detes .d holidayfi. datas d weekdayfi. customer customer constance customer conducationer conducationer conducationer							
· university dates · uneerdayit, customer · custokey, customer · uname, customer · customer · customer · City, customer ·							
, outcome greaters, castomer, castomer, castomer, cancesquent, part, pratter, pratter, print, part, pratter, part, product, print, product, castomer, c							
tes, isb 10', customer', isb 10', bart'							
WHERE `lineorder`, `lo orderdate`	= `dates`,`d datekev` AND `lineorder`,`lo custkev` =	`customer`	.`c custkey` AND `lineorder`.`lo suppkey` = `supplier`				
.`s_suppkey` AND `lineorder`.`lo	partkey' = `part`.`p_partkey`						
Table Properties	[transient_lastDdlTime=1577677599]	1					
Serde Library	org.apache.hadoop.hive.serde2.lazv.LazvSimpleSerDe	i					
InputFormat	org.apache.hadoop.mapred.TextInputFormat	i					
OutputFormat	org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutpu	tFormat	·				
Storage Properties	[serialization.format=1]	i .					
+		+	+				
73 rows selected (0.057 seconds)							

0: jdbc:hive2://localhost:10001/ssb\_10>

#### 基于lineorder\_flatten视图简单查询的执行计划如下:

-
<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.enforce.bucketing does not exist 19/12/30 14:19:32 WARN [main] HiveConf: fname hive.support.sql11.reserved.keywords does not exist</pre>
   plan
== Physical Plan ==
<pre>UblectLimit 10 LollectLimit re>
<pre>i : : +- Scan hive ssb_10.lineorder [lo_orderkey#310L, lo_linenumber#311L, lo_custkey#312, lo_partkey#313, lo_suppkey#314, lo_ord erdat#315, lo_orderpriotity#316, lo_shippriotity#317, lo_quantity#318L, lo_extendedprice#319L, lo_ordtotalprice#320L, lo_discount#321L, lo_revenue#32 21, lo_supplycost#3232L, lo_tax#324L, lo_commitdate#325, lo_shipmode#326J, HiveFableRelation `ssb_10'. `lineorder', org.apache.hadoop.hive.serde2.lazy.L azySimpleSerDe, [lo_orderkey#310L, lo_linenumber#311L, lo_custkey#312, lo_partkey#313, lo_suppkey#314, lo_orderdate#315, lo_orderpriotity#316, lo_ship priotity#317, lo_quantity#318L, lo_extendedprice#319L, lo_ordtotalprice#320L, lo_discount#321L, lo_revenue#322L, lo_supplycost#323L, lo_tax#324L, lo_c</pre>
<pre>: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) : : : +- *(1) Filter isnothull(s_suppkey#327) :</pre>
333], HiveTableRelation `ssb_10`.`supplier`, org.apache.hadoop.hive.serde2.lazySimpleSerDe, [s_suppkey#327, s_name#328, s_address#329, s_city#330 , s_nation#331, s_region#332, s_phone#333]
: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) : : +- *(2) Filter isnotnull(d_datekey#334) : : : : : : : : : : : : : : : : : : :
: - Scan nive ssp_te.dates [u_datekey#344, o_datekey#344, o_date#345, d_date#345, d_datekey#346, d_datekey#347,
<pre>: +- *(6) Sort [c_custkey#351 ASC NULLS FIRST], false, 0 : +- Exchange hashpartitioning(c_custkey#351, 200)</pre>
<pre>. += *(a) Filter (lishothull(_legion#360 / ad (_legion#360 - AirA) ad Ishothull(_ustkey#351) : +- Scan hive sb_10.customer (_custkey#351, _c_name#353, _c_ity#354, _c_nation#355, c_region#356, c_phone#357, c_mk tsegment#358], HiveTableRelation `ssb_10`.`customer`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [c_custkey#351, c_name#352, c_address#353, c _city#354, c_nation#355, c_region#356, c_phone#357, c_mktsegment#358] +- *(10) Sort [p_partkey#359 ASC NULLS FIRST], false, 0 +- Exchange hashpartitioning(p_partkey#359, 200)</pre>
<pre>+- *(9) Filter isnotnul1(p_partkey#359) +- Scan hive sbs_l0.part [p_partkey#359, p_name#360, p_mfgr#361, p_category#362, p_brand#363, p_color#364, p_type#365, p_size#366, p_conta iner#367], HiveTableRelation `ssb_10`.`part`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [p_partkey#359, p_name#360, p_mfgr#361, p_category#3</pre>
02, p_DTand#363, p_C0107#364, p_Type#360, p_S12e#366, p_container#36/] +
0: jdbc:hive2://localhost:10001/ssb_10>
在为line order_flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计 划,优化后的执行计划如下:
<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:17:47 INFO [main] HiveConf: Found configuration file file://tc/ccm/spark-conf-2.4.3-hadoop2.8-1.4.2/hive-site.xml 19/12/30 14:17:47 WARN [main] HiveConf: file: file://tiveConf of name hive.enforce.bucketing does not exist 19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist</pre>
plan
== Physical Plan == CollectLimit 10
+- *(1) Project [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderdate#133, lo_orderpriotity#134, lo_ship priotity#135, lo_quantity#136L, lo_extendedprice#137L, lo_ordtolaprice#138L, lo_discount#139L, lo_revenue#146L, lo_supplycost#141L, lo_tax#142L, lo_c ommitdate#143, lo_shipmed#144, s_suppkey#145, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#156, s_phone#151, 34 more fields] + *(1) Filter (isnotnull(c_region#174) && (c_region#174 = ASIA)) - *(1) Filter (isnotnull(c_region#174) && (c_region#174 = ASIA)
+ +(1) illocan parquet [10_0rderRey#120t,10_1inenumber#127t,10_custkey#130,10_partkey#131,10_suppkey#132,10_0rderpriotity#134,10_snippriotity#

可以看到,优化后的执行计划省去了lineorder\_flatten视图的所有计算逻辑,直接访问HDFS中Cache的数据。

#### 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性,而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据,用户需要根据查询对于数据一致性的需求,触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

# 9.5. 工具集

# 9.5.1. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

#### 前提条件

已创建集群,详情请参见创建集群。

### 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

#### 挂载

⑦ 说明 依次在每个节点上执行挂载操作。

#### 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。

2. 执行如下命令, 新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

#### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

#### 返回用户在服务端配置的所有命名空间列表。

test testcache

2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

### 3. 创建目录。

mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

#### 如果您想使用Python方式写入和读取文件,请参见如下示例:

#### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

### 返回如下信息。

my first file This file

# 卸载

? 说明 依次在每个节点上执行卸载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 9.5.2. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

#### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

#### 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见登录集群。

2. 执行以下命令,获取帮助信息。

jindo distcp --help

返回信息如下。

--help - Print help text --src=VALUE - Directory to copy files from --dest=VALUE - Directory to copy files to --parallelism=VALUE - Copy task parallelism --outputManifest=VALUE - The name of the manifest file --previousManifest=VALUE - The path to an existing manifest file --requirePreviousManifest=VALUE - Require that a previous manifest is present if specified --copyFromManifest - Copy from a manifest instead of listing a directory --srcPrefixesFile=VALUE - File containing a list of source URI prefixes --srcPattern=VALUE - Include only source files matching this pattern --deleteOnSuccess - Delete input files after a successful copy --outputCodec=VALUE - Compression codec for output files --groupBy=VALUE - Pattern to group input files by --targetSize=VALUE - Target size for output files --enableBalancePlan - Enable plan copy task to make balance --enableDynamicPlan - Enable plan copy task dynamically --enableTransaction - Enable transation on Job explicitly --diff - show the difference between src and dest filelist --ossKey=VALUE - Specify your oss key if needed --ossSecret=VALUE - Specify your oss secret if needed --ossEndPoint=VALUE - Specify your oss endPoint if needed --policy=VALUE - Specify your oss storage policy --cleanUpPending - clean up the incomplete upload when distcp job finish --queue=VALUE - Specify yarn queuename if needed --bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed --s3Key=VALUE - Specify your s3 key --s3Secret=VALUE - Specify your s3 Sercet --s3EndPoint=VALUE - Specify your s3 EndPoint

#### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

```
jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp
```

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

### --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制/*data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

```
hdfs dfs -ls /data/incoming/hourly table/2017-02-01/03
```

#### 返回信息如下。

Found 6 items							
-rw-r	2 root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst	
-rw-r	2 root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log	
-rw-r	2 root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log	
-rw-r	2 root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001	
09							
-rw-r	2 root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt	
-rw-r	2 root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt	

#### 执行以下命令,复制以log结尾的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPattern .*\.lo
g --parallelism 20
```

#### 执行以下命令,查看目标bucket的内容。

```
hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03
```

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

#### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --deleteOnSuccess -
-parallelism 20
```

### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03
```

#### 返回信息如下:

```
      Found 6 items

      -rw-rw-rw-1
      938 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/000151.sst.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/0PTIONS-000109.g

      z
      -rw-rw-rw-1

      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz
```

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

 none表示保存为未压缩的文件。如果文件已压缩,则Jindo DistCp会将其解压缩。

#### • keep表示不更改文件压缩形态,按原样复制。

```
② 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。
```

#### --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma
nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

#### --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录,查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --parallel ism 20

#### 执行以下命令,查看运行期间复制的文件。

```
hadoop fs -text oss://<yourBucketName>/hourly_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst
```

#### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}
#### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

#### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
        Found 4 items

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

#### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

#### 例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

```
Found 8 items
-rw-r---- 2 root hadoop
                              2252 2020-04-17 20:42 /data/incoming/hourly table/2017-02-01/03/000151.sst
-rw-r---- 2 root hadoop
                              4891 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/1.log
-rw-r---- 2 root hadoop
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/2.log
            2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/emp01.txt
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp06.txt
```

执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

#### --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为Dist Cp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo Dist Cp的作业分配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableBalancePlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

# --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定\_\_\_enableDynamicPlan\_来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

### --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

#### 如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的DistCp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --dest oss://<yourB
ucketName>/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis
m 20
```

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yarn队列的名称。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

#### --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

#### 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

# 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

• 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive
--parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

# 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

#### 命令示例如下。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

#### 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distop jindo-distop-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里 ,避免每次使用时填写Accesskey。

#### E-MapReduce

<configuration></configuration>
<property></property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>

#### 此时代码示例如下。

jindo distop /tmp/jindo-distop-2.7.3.jar --src s3://smartdatal/ --dest s3://smartdatal/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

# 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters
Bytes Destination Copied=11010048000
Bytes Source Read=11010048000
Files Copied=1001
Shuffle Errors
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
```

```
⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的
大小可能不相等。
```

# 9.5.3. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

#### 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-dist cp-<version>.jar。
  - Hadoop 2.7及后续版本,请下载jindo-dist cp-3.0.0.jar。
  - Hadoop 3.x系列版本,请下载jindo-dist cp-3.0.0.jar。

### 场景预览

Jindo DistCp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二:使用Jindo Dist Cp成功导完数据后,如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?

- 场景四: 成功导入HDFS数据至OSS, 数据不断增量增加, 在Dist cp过程中可能已经产生了新文件, 该使用哪些参数处理?
- 场景五:如果需要指定Jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九: 如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时, 该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?
- 场景十三: 如果不想将OSS AccessKey这种参数写在命令行里, 该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret),以及Endpoint信息,且该AccessKey具有写目标Bucket的 权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

本场景示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10

```
⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。
```

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch
```

# 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

```
Distcp Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

参数含义如下:

○ Bytes Destination Copied: 表示目标端写文件的字节数大小。

- 。 Bytes Source Read: 表示源端读文件的字节数大小。
- 。 Files Copied: 表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

# 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

在<mark>场景</mark>一的基础上,如果您的Distcp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Distcp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文 件的Copy。示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table
le --dest oss://yang-hhht/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa
nifest --parallelism 20
```

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

# 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信息, 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 参数含义如下:

- --outputManifest : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如*manifest-2020-04-17.gz*,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Dist cp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest=2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

### 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid
th 6 --parallelism 10
```

### 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

#### 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

#### 优化对比如下。



# 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly\_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10

### 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

# 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table
--ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .*\.log --p
arallelism 10
```

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

### 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则, 正则表达式。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.*/([a-z]+).*.txt' --parallelism 20
```

#### 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10
```

# 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

#### E-MapReduce

```
<configuration>
<configuration>
cname>fs.jfs.cache.oss-accessKeyId</name>
<value>xxx</value>
</property>
cname>fs.jfs.cache.oss-accessKeySecret</name>
<value>xxx</value>
</property>
cname>fs.jfs.cache.oss-endpoint</name>
cvalue>oss-cn-xxx.aliyuncs.com</value>
</property>
</configuration>
```

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

# 9.5.4. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存 储成本,或者加速访问数据的速度。

# 使用Jindo jfs

执行以下命令,获取帮助信息。

```
jindo jfs -help archive
-archive -i/a <path> ... :
Archive commands.
```

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

# Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p 参数可以保证本地数据不受磁盘水位清理。

#### Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

# Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参 见对象存储OSS的存储类型介绍。

```
jindo jfs -archive -i|-a <path>
```

-i 参数可以归档数据至OSS低频存储类型。 -a 参数可以归档数据至OSS归档存储类型。

### Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

jindo jfs -unarchive -i/-o <path>

Unarchive默认可以将数据恢复成标准存储, -i 参数可以恢复数据至低频存储类型。 -o 参数可以临时解冻归档存储类型, 使数据临时可读。

### Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

jindo jfs -status -detail/-sync <path>

-detail 参数可以查看文件进度信息。 -sync 参数表示该命令需要同步等待分层存储任务结束才会退出。

### ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -ls2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse -rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia\_data.csv -rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia\_data.json

# 10.SmartData 3.0.x 10.1. SmartData 3.0.x版本简介

Smart Dat a组件是EMR Jindo引擎的存储部分,为EMR各个计算引擎提供统一的存储、缓存、计算优化以及功能扩展。Smart Dat a 组件主要包括JindoFS,JindoTable和相关工具集。本文介绍Smart Dat a(3.0.x)版本的更新内容。

# JindoFS存储优化

- 改进Jindo Namespace服务单机配置,单机情况下也可以更新并异步写入元数据至Tablestore。
- 移除jindo Namespace服务的Tablestore作为元数据后端的配置,不再支持基于Tablestore的HA方案。
- 支持归档存储,允许文件数据按照OSS归档类型进行存储,以节省成本。
- 提供JindoFS分层存储的Archive、Unarchive和Status命令,允许归档至指定目录,查看归档操作进度和相关状态。
- 提供JindoFS ls2命令,允许查看文件信息。
- 支持JindoFS存储系统fsimage的离线导出和分析查询。
- 支持跨集群访问JindoFS存储系统。

JindoFS分层存储命令详情请参见<mark>分层存储命令使用说明</mark>。

### JindoFS缓存优化

- 改进缓存数据磁盘组织, 解除对系统盘的依赖, 实现数据盘之间完全独立, 增强磁盘下线操作。
- 改进缓存服务, 增强节点容错处理和节点下线操作。
- 改进缓存块写入磁盘的选择策略,默认支持轮询(Round Robin)。
- 改进读写流程, 增强容错处理。
- 提供JindoFS分层存储的Cache、Uncache和Status命令,允许缓存至指定目录,支持数据预加载,查看缓存进度和相关状态。
- 优化小文件占用缓存空间的问题,准确地统计相关指标。

# JindoTable计算优化

- 提供JindoTable Optimize命令,支持优化Hive表操作,例如分区小文件合并。
- 提供jindoTable Archive、Unarchive和Status命令,允许归档至指定表和分区,查看归档操作进度和相关状态。
- 支持JindoTable Cache、Uncache和Status命令,允许缓存至指定表和分区,支持数据预加载,查看缓存进度和相关状态。
- 支持导出MaxCompute表至JindoFS缓存系统上,以实现机器学习训练前结构化数据的预加载机制。

JindoTable详情请参见JindoTable使用说明。

# JindoFS OSS扩展和支持

- 支持在客户端进行Ranger权限集成,获取OSS各种操作,通过JindoFS服务记录进行Ranger权限检查。
- 支持在客户端进行操作审计,获取OSS各种操作,通过JindoFS服务记录操作记录,作为审计用途。
- 支持Hadoop Credentials Provider框架,允许按照Hadoop常用方式指定OSS的AccessKey配置。
- 支持Flink Connector, 允许Flink引擎将OSS作为source、sink和checkpoint存储。
- 提供JindoFS OSS SDK(Hadoop Connector)轻量版本(lite),主要适用于非标准环境,例如用户的IDC(Internet Data Center)集群环境。

# JindoManager系统管理

支持通过UI来查看JindoFS存储系统上的系统状态、文件统计和缓存系统上的缓存指标统计。

# JindoTools工具集

改进Jindo DistCp工具的分发机制,针对EMR集群内使用场景和非EMR集群环境使用场景,分别使用不同的发行包。

Jindo Dist Cp提供轻量版本(lite),主要适用于非标准环境,例如用户的IDC集群环境。

# 10.2. JindoFS Block模式

# 10.2.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

## 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

### 配置使用方式

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	过滤				服务配置
配置搜	索				全部 I smartdata-site namespace client I storage
<b>清頼</b>	λ			Q	

#### 3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

- i. 修改jfs.namespaces为test。
  - test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

#### ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

描述	示例
表示test命名空间的后端存储。	oss:// <oss_bucket>/<oss_dir>/ ⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。</oss_dir></oss_bucket>
表示test命名空间为块存储模式。	block
表示存储后端OSS的AccessKey ID。	XXXX
表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>
	描述 表示test命名空间的后端存储。 表示test命名空间为块存储模式。 表示存储后端OSS的AccessKey ID。 表示存储后端OSS的AccessKey Secret。

#### iii. 单击确定。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在**服务配置**区域的storage页签,修改如下参数。

服务配置				
全部 smartdata-site namespace	client storage			
	storage.handler.threads	40		0
	storage.watermark.low.ratio	0.2		0
	storage.watermark.high.ratio	0.4		0
	storage.oss.upload.threads	20		0
参数		描述		
storage.watermark.high.ratio		表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。		
storage.watermark.low.ratio			表示使用量的下水位比例,触发清理后会自动清理冷数据,将 JindoFS数据目录占用空间清理到下水位。默认值:0.2。	

```
⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。
```

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 10.2.2. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

#### 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Raft作为元数据后端,详情请参见使用Raft-RocksDB-Tablestore作为存储后端。

单机RocksDB作为元数据服务的架构图如下所示。



### 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。

#### ii. 单击namespace。

状态     部署拓扑     配置修改历史       配置过滤     服务配置       配置搜索     全部   smartdata-site       namespace     client   storage	< 返回	â Smart	Data 🗸	●正常				
配置过滤 服务配置 配置搜索 全部 smartdata-site namespace client storage	状态	部署拓扑	配置	配置修改历史				
配置搜索 全部 smartdata-site namespace client storage	配置过	滤			服务配置			
	配置搜	素			全部 smartdata-site nan	mespace	client	storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中,输入执行原因,开启自动更新配置。
  - ⅲ. 单击**确定**。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。
- 6. (可选)配置远端Tablestore(OTS)异步存储。

您可以给集群绑定一个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储介质,本地的元数据信息会异步地同步 至您的Tablestore实例上。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jf s
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,推荐使http://emr-jfs.cn-hangzhou.vp用VPC地址。tore.aliyuncs.com	
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

- 7. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 8. 单击右上角的操作 > 启动All Components。

# 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有一份完整的JindoFS元数据的副本。您可以在停止或释 放原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

1. 准备工作。

i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

hadoop fs -count jfs://test/

返回信息类似如下。

```
1596 1482809 25 jfs://test/
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果显示 synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus

返回信息类似如下所示。

[hadoop@emr-header-1 ~]\$ jindo jfs -metaStatus
<pre>==== emr-header-1:8101 ====</pre>
OtsUploader: _synced=1
[RocksDB Row Counts of each CF(Table)]

- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。

3. 初始化配置。

在SmartData服务的namespace页签,添加如下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: ° true ° false	false
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	true

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

iii. 单击确定。

- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus

如图所示, state为FINISH时表示恢复完成。

==== emr-header-1:8101 ====
[Recovery From OTS Status]
state: FINISH
total 22855 rows.

(可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。
 此时的集群为恢复模式,也是只读模式。

```
# 对比文件数量一致
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
         1596 1482809
                                                   25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root
                                             0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087

      drwxrwxr-x
      - root
      root
      0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-

      -rw-r----
      1 hadoop hadoop
      5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt

      -rw-r-----
      1 hadoop hadoop
      20 2020-03-25 15:07 jfs://test/testfile

-rw-r---- 1 hadoop hadoop
                                           20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021 , ErrorMsg: Namespace is under recovery mode, and is read-only.
```

### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.rocksdb.asyn c.ots.enabled	是否开启OTS异步上传,包括: o true o false	true
namespace.backend.rocksdb.reco very.mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

- 9. 重启集群。
  - i. 单击上方的集群管理页签。
  - ii. 在集群管理页面, 单击相应集群所在行的更多 > 重启。

# 10.2.3. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

#### 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

```
⑦ 说明 需要开启事务功能。
```

• 创建3 Master的EMR集群,详情请参见创建集群。

🖽 高可用	高可用: 🕜		
	部署方式: 🕜	🔿 2 Master	
	0	3 Master	查看服务部署
? 说明	如果没有部署	方式,请 <mark>提交</mark>	工单处理。

# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。





# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏, 单击集群服务 > Smart Data。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏, 单击**集群服务 > SmartData**。
  - ii. 单击**配置**页签。
  - iii. 在**服务配置**区域,单击namespace页签。
- 4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft

参数	描述	示例	
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0	
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101	

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤 7。

#### 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例		
namespace.ots.instance	Tablestore实例名称。	emr-jf s		
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk		
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX		
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常http://emr-jfs.cn-hangzhouEMR集群,推荐使用VPC地址。tore.aliyuncs.com			
namespace.backend.raft.async.ot s.enabled	<ul> <li>是否开启OTS异步上传,包括:</li> <li>true</li> <li>false</li> <li>当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。</li> <li>⑦ 说明 如果SmartData服务已</li> </ul>	true		
	完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。			

#### 6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

7. 单击右上角的操作 > 启动All Components。

#### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
(文件夹个数) (文件个数)
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 \_\_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail

RANCESSON 192, 148, RANCESSON	
ONE	
PPED	
EDULING(in 2335ms)	
SCHEDULING(in 150305ms)	
ext index_624626 flying annend entrie	$a_{1} = a_{1} = a_{1} = a_{2} = a_{1} = a_{2} = a_{1} = a_{2} = a_{2} = a_{2} = a_{1} = a_{2$
next index_624626 flying append_entre	$s_s s_s z_{e=0}$ fulle $nc_{e=2301}$ $uc_{e=024201}$ $(c_{e=0})$
hext_thuex=024020 rtythg_uppenu_entr	Tes_size=0 tale nc=2501 ac=625564 tc=0
syncod_1	
	KONE JPPED JEDULING(in 2335ms) SCHEDULING(in 150305ms) ext_index=624626 flying_append_entric next_index=624626 flying_append_entric 

iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。

#### 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

#### 3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: o true o false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: o true o false	true

#### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus -detail

如图所示, LEADER节点的state为FINISH表示恢复完成。

```
[RaftPeerImpl]
peer_id: _____ :8103:0
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: NO stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
vote_timer: timeout(5000ms) STOPPED
stepdown_timer: timeout(5000ms) SCHEDULING(in 3382ms)
snapshot_timer: timeout(600000ms) SCHEDULING(in 474855ms)
storage: [1, 153]
disk_index: 153
known_applied_index: 153
last_log_id: (index=153,term=2)
first_index_pinned: 1
state_machine: Idle
last_committed_index: 153
last_snapshot_index: 1
last_snapshot_term: 2
 snapshot_status: IDLE
replicator_1116691496965@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0 replicator_3311419785217@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0
 [Recovery From OTS Status]
 state: FINISH
 total rows: 1484409
 table `jfs_block_test` 2 rows.
table `jfs_namespace_cache_ns` 1 rows.
 table `jfs_namespace_test` 1484406 rows.
```

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

#### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

#### 9. 重启集群。

- i. 单击上方的集群管理页签。
- ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 10.2.4. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

### 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

# 访问JindoFS Web UI

打通SSH隧道后,您可以通过*http://emr-header-1:8101/*访问JindoFS Web U功能。JindoFS 3.0版本提供总览信息 (Overview)、Namespace信息、存储节点信息以及专家功能(Advanced)。

● 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

# Overview

Start Time:	Fri Oct 16 12:29:24 2020
Status:	Active
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)
Node:	Live Nodes: [2], Decommission Nodes: [0]
Version:	3.0.0
Build No:	fa0ea608a42

#### • Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespace Info (1)						
Namespace:	jfs://test/					
Namespaces:	test					
Mode:	BLOCK_MODE					
Backend URI:	oss://					
Summary:	Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27					

• StorageService信息

包含当前集群的 StorageService 列表,以及对应 StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

Overview								
Start Time:	Start Time: Fri Oct 16 12:29:25 2020							
Version:		3.0.0						
Build Version:		fa0ea608a42a	5e0e4ebcdbbfc3c041fe49f8e82	?e				
Storage Lists (4	Storage Lists (4)							
Directory	Storag	еТуре	Capacity	Last Eviction Time	Partition Id			
/mnt/d	Disk		28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31			
/mnt/d	Disk		19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21			
/mnt/d	Disk Disk		19.59 GB/58.91 GB 51.95 GB/58.91 GB	Fri Oct 16 20:25:43 2020 Fri Oct 16 20:19:48 2020	21			

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

# 10.2.5. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

# 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	a∰o Smart	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	t滤			服务配置
配置搜	索		_	全部 smartdata-site namespace client storage
请输入	λ		Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因,单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace\_name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - ⅳ. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

Ranger ØAccess Manager 🗅 Audi	it 🗢 Settings				🔒 admin
Service Manager					
Service Manager					Import Export
🗁 HDFS	+32	🗁 HBASE	+ 🛛 🖸	🕞 HIVE	+ 🛛 🖸
		emr-hbase	• 2 😭	emr-hive	• (2) 😭

iii. 配置相关参数。

参数	描述				
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。				
Username	自定义。				
Password	自定义。				
Namenode URL	输入 <i>jfs://{namespace_name</i> ]/。				
Authorization Enabled	使用默认值No。				
Authentication Type	使用默认值Simple。				
dfs.datanode.kerberos.principal					
dfs.namenode.kerberos.principal	云档它				
dfs.secondary.namenode.kerberos.principal	小块司。				
Add New Configurations					

iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信 息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。 以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hado op.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

### 3. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因,单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 10.2.6. AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

#### 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

# 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。

# Smart Dat a Smart Dat a 3.0.x

# E-MapReduce

参数	描述
ip	Client IP.
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

# 使用AuditLog

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常				
状态	部署拓扑	配置	配置修改历史				
配置过	滤				服务配置		
配置搜	ξ.				全部 smartdata-site	namespace	client storage
清諭)				Q			

- 3. 配置如下参数。
  - i. 在namespace页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

- ⅲ. 单击部署客户端配置。
- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在确认对话框中, 单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在确认对话框中,单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则, 在创建生命周期规则配置各项参数。
- 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

#### 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

show partitions table\_name 获取所有分区。

• desc formatted table\_name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> show tables	s;		
database table	Name	isTemporary	
default audit_log	false		
default audit_log_sour	rce	false	
default fs_image	false		

• 执行如下命令显示分区。

show partitions audit\_log\_source;

```
返回信息类似如下。
```

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken:_0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

```
返回信息类似如下。
```

jindo-sa	<pre>l&gt; select * from</pre>	n audit_log_	source limit	10;				
datetime	allowed	ugi ip	ns	cmd src	dst perm	date		
2020-10-	20 10:50:11.924	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	oop:rwxrwxr-	x 2020-10-	20				
2020-10-	20 10:50:11.950	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:06.445	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null	perm=hadoop:had	oop:rwxrwxr-	x 2020-10-	20				
2020-10-	20 11:26:06.469	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=listFileletRequest src=jfs	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:11.295	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs:d
s	dst=null	perm=root:r	oot:rwxr-xx	2020-10	-20			
2020-10-	20 11:26:11.320	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null	perm=null	2020-10-20						
2020-10-	20 11:26:14.368	allowed=tru	e ugi=root	(auth:SIMPLE)	ip=192.	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s	dst=null	perm=root:r	oot:rwxr-xx	2020-10	-20			

```
select * from audit_log limit 10;
```

```
返回信息类似如下。
```

jindo-sql> s	elect * from	m audit_log	lim	it 10;										
datetime	allowed	ugi ip		ns	cmd	src	dst	perm	date					
2020-10-20 1	0:50:11.924	true roo	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	0:50:11.950	true roo	ot (	auth:SIMPL	LE)	192.	- 10 C	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:06.445	true roo	ot (	auth:SIMPL	LE)	192.	100	kugou	getFileStatusRequest	jfs://k	11	hadoop:h	adoop:r	wxrwx
r-x 2020	-10-20													
2020-10-20 1	1:26:06.469	true roo	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 1	1:26:11.295	true roc	ot (	auth:SIMPL	LE)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:11.320	true roc	ot (	auth:SIMPL	LE)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:14.368	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:14.393	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
2020-10-20 1	1:26:16.230	true roc	ot (a	auth:SIMPL	.E)	192.		kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020	-10-20													
2020-10-20 1	1:26:16.255	true roc	ot (	auth:SIMPL	.E)	192.		kugou	listFileletRequest	jfs://k		null	null	2020
-10-20														
Time taken:	1.918 second	ds, Fetched	10	row(s)										

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-sql> select cmd ,	count(*) from	audit_log w	where date="	2020-10-20"	group b	by cmd	order	by cm	d;
cmd count(1)									
getFileStatusRequest	387								
listFileletRequest	387								
Time taken: 5 767 secon	ds Fatchad 2	row(s)							

# 10.2.7. 文件元数据离线分析

EMR-3.30.0及后续版本的Block模式,支持dump整个namespace的元数据信息至OSS中,并通过Jindo Sql工具直接分析元数信息。

# 背景信息

在HDFS文件系统中,整个分布式文件的元数据存储在名为fsimage的快照文件中。文件中包含了整个文件系统的命名空间、文件、Block和文件系统配额等元数据信息。HDFS支持通过命令行下载整个fsimage文件(xml形式)到本地,以便离线分析元数据 信息,而JindoFS无需下载元数据信息至本地。

# 上传文件系统元数据至OSS

使用Jindo命令行工具上传命名空间的元数据至OSS,命令格式如下。

jindo jfs -dumpMetadata <nsName>

<nsName> 为Block模式对应的namespace名称。

例如,上传并离线分析test-block的元数据。

jindo jfs -dumpMetadata test-block

SLF4J: Elass path contains multiple SLF4J bindings. SLF4J: Found binding in [jar:file:/Users/ci code/bigboot-3rdparty/bigboot/output/sdk/lib/bigboot-emr-cli.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ci code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ci code/bigboot-3rdparty/bigboot/output/sdk/lib/jindo-auditlog-full.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: Found binding in [jar:file:/Users/ci code/bigboot-3rdparty/bigboot/output/sdk/lib/jboot.jar!/org/slf4j/impl/StaticLoggerBinder.class] SLF4J: See http://www.slf4j.org/codes.html#mutiple\_bindings for an explanation. SLF4J: Actual binding is of type [org.slf4].impl.Log4jLoggerFactory] Sucessfull yupload namespace metadata to 055.

当提示如下信息时,表示上传成功并以JSON格式的文件存放在OSS中。

```
Sucessfully upload namespace metadata to OSS.
```

# 元数据上传路径

元数据信息上传的路径为JindoFS中配置的sysinfo的子目录下的metadataDump子目录。

例如, 配置的 namespace.sysinfo.oss.uri 为 oss://abc/ ,则上传的文件会在 oss://abc/metadataDump 子目录中。

参数	说明
namespace.sysinfo.oss.uri	存储Bucket和路径。
namespace.sysinfo.oss.endpoint	对应Endpoint信息,支持跨Region。
namespace.sysinfo.oss.access.key	阿里云的AccessKey ID。
namespace.sysinfo.oss.access.secret	阿里云的AccessKey Secret。

批次信息:因为分布式文件系统的元数据会跟随用户的使用发生变化,所以我们每次对元数据进行分析是基于命令执行当时的元 数据信息的快照进行的。每次运行Jindo命令进行上传会在目录下,根据上传时间生成对应批次号作为本次上传文件的根目录, 以保证每次上传的数据不会被覆盖,您可以根据需要删除历史数据。

< > * <1	oss://emrtest/sysinfc/metadataDump	test-block/
▲上传 + 创建目录	□ 更多 <b>▼</b>	
□ 名称		类型 / 大小
□	18_58_16 3	目录

- ①表示OSS系统信息配置路径。
- ②表示namespce。
- ③表示批次号。

#### 元数据Schema

上传至OSS的文件系统元信息以JSON文件格式存放。其Schema信息如下。

```
{
  "type":"string", /*INode类型, FILE文件DIRECTORY目录*/
  "id": "string", /*INode id*/
  "parentId" :"string", /*文节点id*/
  "name":"string", /*INode名称*/
  "size": "int", /*INode大小, bigint*/
  "permission":"int", /*permission以int格式存放*/
  "owner":"string", /*owner名称*/
  "ownerGroup":"string", /*owner组名称*/
  "mtime":"int", /*inode修改时间, bigint*/
  "attributes":"string", /*文件相关属性*/
  "state":"string", /*INode状态*/
  "storagePolicy":"string", /*etag*/
}
```

# 使用Jindo Sql分析元数据

1. 执行如下命令,启动Jindo Sql。

```
[root@emr-header-1 ~]# jindo sql
Spark master: yarn, Application Id: application_1603081647416_0050
jindo-sql> show tables;
database tableName isTemporary
default audit_log false
default audit_log_source false
default fs_image false
Time taken: 0.33 seconds, Fetched 3 row(s)
```

- 2. 查询Jindo Sql可以分析的表格。
  - 使用 show tables 可以查看支持查询分析的表格。目前Jindo Sql内置了审计和元数据信息的分析功能,对 应 audit\_log和fs\_image。
  - 使用 show partitions fs\_image 可以查看表的fs\_image分区信息。每一个分区对应于一次上传 jindo jfs -dumpMeta data 生成的数据。

示例如下。

<pre>jindo-sql&gt; show partitions fs_image; partition</pre>
namena le (detetime 2020 10 20 10 47 14
namespace=k / aatetime=2020_10_20_10_47_14
namespace=k /datetime=2020_10_20_10_50_36
namespace=k /datetime=2020_10_20_10_52_06
Time taken: 0.045 seconds, Fetched 3 row(s)

3. 查询分析元数据信息。

Jindo Sql使用Spark-SQL语法。您可以使用SQL进行分析和查询fs\_image表。

示例如下。

[root@em Spark ma	ir-worker ister: yd	-2 hadoo irn, Appl	op]# jin lication	do sql Id: app													
jindo-se	1> show	tables;															
database		tableNam	ne	isTempo	rary												
default	audit_le	g	false														
default	audit_lo	g_source	8	false													
default	fs_image		false														
Time tak	en: 0.34	5 second	ds, Fetc	hed 3 ro	m(s)												
jindo-so	<pre>pl&gt; selec</pre>	t * from	n fs_ima	ge limit	10;												
atime	attr	etag	id	mtime	nane	owner	ownerGr	oup	parentId	permiss	sion	size	state	storage	Policy	type	name
space	datetime																
0			7311076	00505189	9448	1603084	1070081	/tpcds/	'orc/5000/web	_returns/wr_	returned	_date_:	sk=2450819	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0			1653444	80419066	75495	1603084	071350	/tpcds/	orc/5000/web	_returns/wr_	returned	_date_:	sk=2450820	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			7311076	00505189	9470	1603084	070185	/tpcds/	'orc/5000/web	_returns/wr_	_returned	_date_:	sk=2450821	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			1192276	20234792	87249	1603084	069581	/tpcds/	'orc/5000/web	_returns/wr_	.returned	_date_:	sk=2450822	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ny	kugou	2020_10_20_	10_50_36							
0			1076984	05188724	41036	1603084	073592	/tpcds/	orc/5000/web	_returns/wr_	returned	_date_:	sk=2450823	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0			2699389	98662451	1354	1603084	068996	/tpcds/	orc/5000/web	_returns/wr_	_returned	_date_:	sk=2450824	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			1192276	20234792	87307	1603084	069875	/tpcds/	orc/5000/web	_returns/wr_	_returned	_date_:	sk=2450825	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			1546468	48201766	500Z	1603084	072440	/tpcds/	orc/5000/web	_returns/wr_	.returned.	_date_:	sk=2450826	root	root	334790	833296
5855433	489	ø	Finaliz	ed	WARM	Directo	ory	kugou	2020_10_20_	10_50_36							
0			1653444	80419066	75460	1603084	071170	/tpcds/	orc/5000/web	_returns/wr_	returned,	_date_:	sk=2450827	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ry	kugou	2020_10_20_	10_50_36							
0	-		7311076	00505189	9544	1603084	070572	/tpcds/	orc/5000/web	_returns/wr_	_returned	_date_	sk=2450828	root	root	334790	833296
5855433	489	0	Finaliz	ed	WARM	Directo	ny	kugou	2020_10_20_	10_50_36							

namespace和datetime为Jindo Sql增加的两列,分别对应于namespace名称和上传元数据的时间戳。

例如:根据某次dump的元数据信息统计该namespace下的目录个数。

jindo-sql> count(1)	select	count(*)	from	fs_image	where	type	= "Directory	and namespace="kugou	" and	datetime="2020_10_20_10_47_14";	
11837 Time taken:	6 852 5	econds I	Fetche	d 1 row(s	0						
rune cuken.	0.052 5	econus, i	eccne	a r rom(s	·/						

# 使用Hive分析元数据

1. 在Hive中创建Table Schema。

在Hive中创建对应的元信息以供查询,您可以参考下面的格式在Hive中创建文件系统元信息对应表的Schema。

```
CREATE EXTERNAL TABLE `table_name`
(`type` string,
 `id` string,
`parentId` string,
 `name` string,
`size` bigint,
`permission` int,
 `owner` string,
 `ownerGroup` string,
 `mtime` bigint,
 `atime` bigint,
 `attr` string,
 `state` string,
 `storagePolicy` string,
`etag` string)
ROW FORMAT SERDE 'org.apache.hive.hcatalog.data.JsonSerDe'
STORED AS TEXTFILE
LOCATION '文件上传的OSS路径';
```

#### 2. 使用Hive进行离线分析。

创建完Hive表后,您可以使用Hive SQL分析元数据。

select \* from table\_name limit 200;

示例如下。

hive> select * from inode_metadata_test8 limit 100;												
WARNING: Hive-on-MR is deprecated in Hive 2 and may not be available in the future versions. Consider using a different execution engine (i.e. spark, tez) or using Hive 1.X relea	ases.											
Query ID = root_202009/												
Total jobs = 1												
(Launching Job 1 out of 1												
Number of reduce tasks is set to 0 since there's no reduce operator												
Starting Job = job 1 , Tracking URL = http://emr-heade :20080/proxy/applic:												
Kill Command = /usr/lib/hadoop-current/bin/hadoop jobkill job 1599												
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: θ												
2020-09-08 14:57:26,112 Stage-1 map = 0%, reduce = 0%												
2020-09-08 14:57:31,263 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 1.22 sec												
MapReduce Total cumulative CPU time: 1 seconds 220 msec												
Ended Job = job_;												
MapReduce Jobs Launched:												
Stage-Stage-1: Map: 1 Cumulative CPU: 1.22 sec   HDFS Read: 6867 HDFS Write: 1524 SUCCESS												
Total MapReduce CPU Time Spent: 1 seconds 220 msec												
Directory 11274334386847219714 11274334386847219713 /uttest/oss 0 511 caojie staff 1599545017615 1599545017615 Finalized WARM												
Directory 11274334386847219719 11274334386847219713 /uttest/oss2 0 511 caojie staff 1599545017654 1599545017654 Finalized WARM												
Directory 11274334386847219716 11274334386847219714 /uttest/oss/dir 0 511 caojie staff 1599545017636 1599545017636 Finalized WARM												
File 11274334386847219715 11274334386847219714 /uttest/oss/file1 0 420 caojie staff 1599545017632 1599545017632 Finalized WARM												
File 11274334386847219717 11274334386847219716 /uttest/oss/dir/file2 0 420 caojie staff 1599545017642 1599545017642 Finalized WARM												
File 11274334386847219718 11274334386847219716 /uttest/oss/dir/file3 0 420 caojie staff 1599545017651 1599545017651 Finalized WARM												
Directory 11274334386847219728 11274334386847219719 /uttest/oss2/dir 8 511 caojie staff 1599545817654 1599545817654 Finalized WAF	RM											
File 11274334386847219721 11274334386847219720 /uttest/oss2/dir/file2 0 420 caojie staff 1599545017658 1599545017658 Finalized WARM												
File 11274334386847219722 11274334386847219720 /uttest/oss2/dir/file3 0 420 caojie staff 1599545017666 1599545017666 Finalized WARM												
Directory 11274334386847219713 17672823557433851905 /uttest 0 511 caojie staff 1599545017615 1599545017615 Finalized WARM												
Time taken: 10.734 seconds, Fetched: 10 row(s)												

# 10.3. JindoFS Cache模式

# 10.3.1. JindoFS缓存模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主要介绍JindoFS的缓存模式及其使用方式。

# 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

# 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

- OSS Scheme
   详情请参见配置OSS Scheme(推荐)。
- JFS Scheme
   详情请参见配置JFS Scheme。

# 配置OSS Scheme (推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改 即可运行。

# 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。

ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	● 正常		
状态	部署拓扑	配置	配置修改历史		
配置过	述				服务配置
配置搜	素				全部   smartdata-site   namespace   client   storage
清輪ノ				Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

- i. 修改jfs.namespaces为test。
- test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。
- ii. 单击自定义配置, 在新增配置项对话框中增加以下参数。

参数	参数说明	示例				
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>				
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。				
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache				

- 4. 单击**确定**。
- 5. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 6. 选择右上角的操作 > 重启 Jindo Namespace Service。

```
重启后即可通过 jfs://test/<path_of_file> 的形式访问JindoFS上的文件。
```

### 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面, 单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为true,表示启用缓存模式。

此配置无需重启SmartData服务。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

缓存模式启用后,Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情 请参见磁盘空间水位控制。

#### 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在服务配置区域的storage页签,修改如下参数。

	服务配置 全部   smartdata-site   namespace	client <b>storage</b>						
		storage.handler.threads	40	0				
		storage.watermark.low.ratio	0.2		0			
		storage.watermark.high.ratio	0.4		0			
		storage.oss.upload.threads	20		0			
×	数		描述					
storage.watermark.high.ratio				表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据目录占 用的磁盘空间到达上水位即会触发清理。默认值:0.4。				
storage.watermark.low.ratio			表示使用 JindoFS数	量的下水位比例,触发清理后会自动》 '据目录占用空间清理到下水位。默认	清理冷数据 <i>,</i> 将 、值:0.2。			

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - ⅳ. 在确认对话框中, 单击确定。

# 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > Smart Dat a的配置页面,单击smart dat a-sit e页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss.accessKeyld	表示存储后端OSS的AccessKey ID。
fs.jfs.cache.oss.accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss.endpoint	表示存储后端OSS的endpoint。

⑦ 说明 兼容EMR-3.30.0之前版本的配置项。

- JFS Scheme
  - i. 在集群服务 > Smart Data的配置页面,单击namespace页签。
#### ii. 修改jfs.namespaces为test。

iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。

## 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在**服务配置**区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

#### • 在服务配置区域的smartdata-site页签,配置以下参数。

参数	参数说明
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为 8MB, 如果作业同时打开的写入流较多导致内存使用过大,可以适当调小 此参数。默认值:1048576。
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。
fs.oss.committer.magic.enabled	⑦ 说明 针对Cache模式下,这类OSS对象存储rename操 作性能较差的问题,推出了Jindo Job Committer。

## 10.3.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时,E-MapReduce(简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

## 前提条件

适用环境: ECS(EMR环境外)+Hadoop+JavaSDK。

## 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smart dat a-aliyun-jf s-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current/jars/*里面的包也删除,从而可以正常使用。

## 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。

### 6. 单击完成。

## 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击**确定**。

## 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏,单击实例与镜像>实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

## 步骤四:在ECS上设置环境变量

### 执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

## 步骤五:测试免密方式访问的方法

1. 使用Shell访问OSS。

```
hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>
```

#### 2. 使用Hadoop FileSystem访问OSS。

```
JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。
```

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
       while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
            Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
        }
   }
}
```

## 10.3.3. 访问JindoFS Web UI

JindoFS提供了Web UI服务,您可以快速查看集群当前的状态。例如,当前的运行模式、命名空间、集群StorageService信息和 启动状态等。

## 前提条件

通过SSH隧道方式才能访问Web UI,详情请参见通过SSH隧道方式访问开源组件Web UI。

## 访问JindoFS Web UI

打通SSH隧道后,您可以通过*http://emr-header-1:8101/*访问JindoFS Web U功能。JindoFS 3.0版本提供总览信息 (Overview)、Namespace信息、存储节点信息以及专家功能(Advanced)。

• 总览信息 (Overview)

包含Namespace启动时间、当前状态、元数据后端、当前Storage服务数量和版本信息等。

Overview						
Start Time:	Fri Oct 16 12:29:24 2020					
Status:	Active					
Meta Backend:	RocksDB (Standalone) emr-header-1.clust (Active)					
Node:	Live Nodes: [2], Decommission Nodes: [0]					
Version:	3.0.0					
Build No:	fa0ea608a4;					

• Namespace信息

包含当前节点可用的Namespace以及对应的模式和后端。Block模式的Namespace支持查看当前Namespace的统计信息,包括目录数、文件数以及文件总大小等。

Namespa	Namespace Info (1)					
Namespace:	jfs://test/					
Namespaces:	test					
Mode:	BLOCK_MODE					
Backend URI:	oss://					
Summary:	Summary: Directory Count: [2], File Count: [47701], File Size: [166725951374], Task Count: [0], Computed at 2020-10-20 10:49:27					

• StorageService信息

包含当前集群的 StorageService 列表,以及对应 StorageService的地址、状态、使用量、最近连接时间、启动时间、 StorageService编号和内部版本信息等。

StorageService (2)							
Node	Status	Capacity	Last contact	Start Time	Storage Id	Version	Build Version
emr-worker-2.cluster-	Healthy	125.70 GB/235.63 GB	2 sec	Fri Oct 16 12:29:30 2020	0	3.0.0	fa0ea608a4
emr-worker-1.cluster-	Healthy	124.69 GB/235.63 GB	1 sec	Fri Oct 16 12:29:25 2020	5	3.0.0	fa0ea608a4

单击Node对应链接,可以查看每个磁盘的空间使用情况。

Overview					
Start Time:	Fri Oct 16 12:29:25 2020				
Version:	3.0.0				
Build Version:	fa0ea608a42a5e0e4ebcdbbfc3c041fe49f8e82e				

## Storage Lists (4)

Directory	StorageType	Capacity	Last Eviction Time	Partition Id
/mnt/d	Disk	28.49 GB/58.91 GB	Fri Oct 16 20:23:51 2020	31
/mnt/d	Disk	19.59 GB/58.91 GB	Fri Oct 16 20:25:43 2020	21
/mnt/d	Disk	51.95 GB/58.91 GB	Fri Oct 16 20:19:48 2020	1
/mnt/d	Disk	24.67 GB/58.91 GB	Fri Oct 16 20:24:40 2020	11

• 专家功能 (Advanced)

专家功能目前仅用于JindoFS开发人员排查问题。

## 10.3.4. 权限功能

本文介绍JindoFS的namespace的存储模式(Block或Cache)支持的文件系统权限功能。Block模式和Cache模式不支持切换。

## 背景信息

根据您namespace的存储模式, JindoFS支持的系统权限如下:

- 当您namespace的存储模式是Block模式时,支持Unix和Ranger权限。
  - 。 Unix权限: 您可以设置文件的777权限, 以及Owner和Group。
  - Ranger权限:您可以执行复杂或高级操作。例如使用路径通配符。
- 当您namespace的存储模式是Cache模式时,仅支持Ranger权限。
   您可以执行复杂或高级操作。例如使用路径通配符。



## 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	a∰o Smar	tData 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜索			全部 smartdata-site namespace client storage	
请输入			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## 5. 重启配置。

```
i. 单击右上角的操作 > 重启 Jindo Namespace Service。
```

ii. 输入执行原因, 单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dirl/file1 hadoop fs -chown john:staff jfs://{namespace name}/dirl/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/ ls: java.security.AccessControlException: Permission denied. Server Exception

## 启用JindoFS Ranger权限

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger UI添加HDFS service。

Ra	nger  ©Access Manager	🗅 Audit 🔹 Settings					💡 admin
S	rvice Manager						
Ser	vice Manager						🛛 Import 🛛 🖾 Export
	HDFS		+ 🛛 🖾	HBASE	+ 20	I HIVE	+ 2 2
				emr-hbase	• 7	emr-hive	<ul> <li>Image: Image: ></ul>

iii. 配置相关参数。

参数	描述		
Service Name	固定格式:jfs-{namespace_name}。 例如:jfs-test。		
Username	自定义。		
Password	自定义。		
Namenode URL	输入jfs://{namespace_name}/。		
Authorization Enabled	使用默认值No。		
Authentication Type	使用默认值Simple。		
dfs.datanode.kerberos.principal			
dfs.namenode.kerberos.principal	不肯它		
dfs.secondary.namenode.kerberos.principal	د بهرم و د ا		
Add New Configurations			

iv. 单击Add。

## 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,以获取LDAP的用户组信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,参见以下示例设置参数来配置LDAP,单击确定。

以下配置项请遵循开源HDFS内容,详情请参见core-default.xml。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击**确定**。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

## 10.3.5. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

## 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口,结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后, MapReduce Application Master或Spark Driver执行最终的Job Commit操作时, 会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

## 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - ∘ Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.JindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的mapred-site页签,设

置**mapreduce.outputcommitter.factory.scheme.oss**为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域, 单击smartdata-site页签。

- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

⑦ 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

## 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

## 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

## 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。

- ii. 单击配置页签。
- iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

默认值为8。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

## 10.3.6. JindoFS AuditLog使用说明

Jindo AuditLog提供缓存和Block模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

## 前提条件

- 已创建EMR-3.30.0版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

## 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数和清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

## 审计信息

Block模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许,取值如下: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	Block模式namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

## 使用AuditLog

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。

- ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
- iii. 单击上方的集群管理页签。
- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	<sup>t</sup> 滤			服务配置
配置搜	素		Q	全部 smartdata-site namespace client storage

#### 3. 配置如下参数。

- i. 在namespace页签, 单击右上角的自定义配置。
- ii. 在新增配置项对话框中,新增如下参数。

参数	描述	是否必填
jfs.namespaces.{ns}.auditlog.enable	打开指定namespaces的AuditLog开 关,取值如下: true:打开AuditLog功能。 false:关闭AuditLog功能。	是
namespace.sysinfo.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.sysinfo.oss.access.key	存储OSS的AccessKey ID。	否
namespace.sysinfo.oss.access.secret	存储OSS的AccessKey Secret。	否
namespace.sysinfo.oss.endpoint	存储OSS的Endpoint。	否

- ⅲ. 单击部署客户端配置。
- iv. 在执行集群操作对话框中, 输入执行原因, 单击确定。
- v. 在**确认**对话框中,单击确定。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中,输入执行原因,单击确定。
  - iii. 在确认对话框中, 单击确定。
- 5. 配置清理策略。
  - OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。
  - i. 登录 OSS管理控制台。
  - ii. 单击创建的存储空间。
  - iii. 在左侧导航栏,单击基础设置 > 生命周期,在生命周期单击设置。

- iv. 单击创建规则,在创建生命周期规则配置各项参数。
  - 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

## 使用Jindo AuditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供SQL的分析功能,通过SQL分析相关表,提供Top-N活跃操作命令分析和Top-N活跃 IP分析。您可以使用 jindo sql 命令,使用该功能。

jindo sql 使用Spark-SQL语法,内部嵌入了audit\_log\_source(auditlog原始数据)、audit\_log(auditlog清洗后数据)和 fs\_image(fsimage日志数据)三个表,audit\_log\_source和fs\_image均为分区表。使用方法如下:

• jindo sql --help 查看支持参数的详细信息。常用参数如下。

参数	描述
-f	指定运行的SQL文件。
-i	启动jindo sql后自动运行初始化SQL脚本。

- show partitions table\_name 获取所有分区。
- desc formatted table name 查看表结构。

因为Jindo sql基于Spark的程序,所以初始资源可能较小,您可以通过环境变量JINDO\_SPARK\_OPTS来修改初始资源Jindo sql的启动参数,修改示例如下。

export JINDO\_SPARK\_OPTS="--conf spark.driver.memory=4G --conf spark.executor.instances=20 --conf spark.executor.cores=5 --conf spark.executor.memory=20G"

#### 示例如下:

• 执行如下命令显示表。

show tables;

jindo-sql> show to	ables;	
database to	ableName	isTemporary
default audit_log	false	
default audit_log	_source	false
default fs_image	false	

• 执行如下命令显示分区。

show partitions audit\_log\_source;

返回信息类似如下。

```
jindo-sql> show partitions audit_log_source;
partition
date=2020-10-20
Time taken: 0.031 seconds, Fetched 1 row(s)
```

• 执行如下查询数据。

```
select * from audit_log_source limit 10;
```

返回信息类似如下。

jindo-sql	l> select * fro	m audit_1	log_source limit	10;						
datetime	allowed	ugi	ip ns	cmd	src	dst	perm	date		
2020-10-2	20 10:50:11.924	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.	10. H I I	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
st=null p	perm=hadoop:had	oop:rwxrv	vxr-x 2020-10	-20						
2020-10-2	20 10:50:11.950	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.	10. ALC	ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null p	perm=null	2020-10-	-20							
2020-10-2	20 11:26:06.445	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	<pre>src=jfs: d</pre>
st=null p	perm=hadoop:had	oop:rwxrv	vxr-x 2020-10	-20						
2020-10-2	20 11:26:06.469	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/ =
null p	perm=null	2020-10-	-20							
2020-10-2	20 11:26:11.295	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s c	dst=null	perm=roo	ot:root:rwxr-x	ĸ	2020-10-	-20				
2020-10-2	20 11:26:11.320	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.		ns=kugou	<pre>cmd=listFileletRequest src=jfs</pre>	://kugou/=
null p	perm=null	2020-10-	-20							
2020-10-2	20 11:26:14.368	allowed=	=true ugi=roo	t (auth	:SIMPLE)	ip=192.	100 ALC: N	ns=kugou	cmd=getFileStatusRequest	src=jfs: d
s c	dst=null	perm=roo	ot:root:rwxr-x	ĸ	2020-10-	-20				

```
select * from audit_log limit 10;
```

返回信息类似如下。

jindo-sql> select * from a	udit_log li	.mit 10;									
datetime allowed ug	ji ip	ns	cmd	src dst	perm	date					
2020-10-20 10:50:11.924 tr	ue root	(auth:SIMP	LE)	192.	kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	wxrwx
r-x 2020-10-20											
2020-10-20 10:50:11.950 tr	rue root	(auth:SIMP	LE)	192.	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11:26:06.445 tr	rue root	(auth:SIMP	LE)	192.	kugou	getFileStatusRequest	jfs://k	11	hadoop:	hadoop:r	wxrwx
r-x 2020-10-20											
2020-10-20 11:26:06.469 tr	rue root	(auth:SIMP	LE)	192.	kugou	listFileletRequest	jfs://k	11	null	2020-10	-20
2020-10-20 11:26:11.295 tr	ue root	(auth:SIMP	LE)	192.	kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-10-20											
2020-10-20 11:26:11.320 tr	rue root	(auth:SIMP	LE)	192.	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20											
2020-10-20 11:26:14.368 tr	ue root	(auth:SIMP	LE)	192.	kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-10-20											
2020-10-20 11:26:14.393 tr	rue root	(auth:SIMP	LE)	192.	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20											
2020-10-20 11:26:16.230 tr	ue root	(auth:SIMP	LE)	192.	kugou	getFileStatusRequest	jfs://k		null	root:ro	ot:rw
xr-xx 2020-10-20											
2020-10-20 11:26:16.255 tr	rue root	(auth:SIMP	LE)	192.	kugou	listFileletRequest	jfs://k		null	null	2020
-10-20											
Time taken: 1.918 seconds,	Fetched 10	row(s)									

• 执行如下命令统计2020-10-20日不同命令的使用频次。

jindo-sql> select cmd	,count(*)	from audit_log	where date="2020	0-10-20" group by	cmd order by	cmd;
cmd count(1)						
getFileStatusRequest	387					
listFileletRequest	387					
Time taken: 5 767 seco	nds Fotol	hed 2 row(s)				

## 10.3.7. Credential Provider使用说明

您可以使用Credential Provider配置加密后的AccessKey信息至文件中,避免泄露AccessKey信息。

## 背景信息

EMR-3.30.0版本支持JindoOSS Credential Provider功能。您可以通过使用Hadoop Credential Provider将加密后的AccessKey信息 存入文件,从而避免配置明文AccessKey,根据不同情况选择合适的JindoOSS Credential Provider。

## 配置JindoOSS Credential Provider

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入smartdata-site服务配置。
  - i. 单击**配置**页签。
  - ii. 在**服务配置**区域,单击smartdata-site页签。
- 3. 添加配置信息。
  - i. 在smartdata-site页签,单击右上角的自定义配置。

#### ii. 在新增配置项对话框中,新增如下配置。

参数	描述
fs.jfs.cache.credentials.provider	配置com.aliyun.emr.fs.auth.AliyunCredentialsProvider的实现类,多个类时使用英文逗号(,)隔开,按照先后顺序读取 Credential直至读到有效的Credential。例如, com.aliyun. emr.fs.auth.TemporaryAliyunCredentialsProvider, com.aliyun.emr.fs.auth.SimpleAliyunCredentialsPr ovider, com.aliyun.emr.fs.auth.EnvironmentVariabl eCredentialsProvider 。 Provider详情请参见Provider类型。

## 使用Hadoop Credential Providers存储AccessKey信息

⑦ 说明 Hadoop Credential Provider详情的使用方法,请参见CredentialProvider API Guide。

#### fs.jfs.cache.oss.accessKeyId、fs.jfs.cache.oss.accessKeySecret和fs.jfs.cache.oss.securityToken可以存储至 Hadoop Credential Providers。

使用Hadoop提供的命令,存储AccessKey和SecurityToken信息至Credential文件中。命令格式如下。

hadoop credential <subcommand> [options]

#### 例如,存储AccessKey和Token信息至JECKS文件中,除了使用文件权限保护该文件外,您也可以指定密码加密存储信息,如果不 指定密码则使用默认字符串加密。

hadoop credential create fs.jfs.cache.oss.accessKeyId -value AAA -provider jceks://file/root/oss.jceks hadoop credential create fs.jfs.cache.oss.accessKeySecret -value BBB -provider jceks://file/root/oss.jceks hadoop credential create fs.jfs.cache.oss.securityToken -value CCC -provider jceks://file/root/oss.jceks

#### 生成Credential文件后,您需要配置下面的参数来指定Provider的类型和位置。

参数	描述
fs.jfs.cache.oss.security.credential.provider.path	配置存储AccessKey的Credential文件。 例如 <i>,jceks://file/\${user.home}/oss.jceks</i> 为HOME下 的 <i>oss.jceks</i> 文件。

## Provider类型

• TemporaryAliyunCredentialsProvider

适合使用有时效性的AccessKey和SecurityToken访问OSS的情况。

参数	参数说明
fs.jfs.cache.oss.credentials.provider	com.aliyun.emr.fs.auth.TemporaryAliyunCredentialsPr ovider
fs.jfs.cache.oss.accessKeyld	OSS的AccessKey ID。
fs.jfs.cache.oss.accessKeySecret	OSS的AccessKey Secret。
fs.jfs.cache.oss.securityToken	OSS的SecurityToken(临时安全令牌)。

SimpleAliyunCredentialsProvider

适合使用长期有效的AccessKey访问OSS的情况。

参数	参数说明
fs.jfs.cache.oss.credentials.provider	com.aliyun.emr.fs.auth.SimpleAliyunCredentialsProvid er
fs.jfs.cache.oss.accessKeyld	OSS的AccessKey ID。
fs.jfs.cache.oss.accessKeySecret	OSS的AccessKey Secret。

#### • Environment VariableCredent ialsProvider

#### 该方式需要在环境变量中配置以下参数。

参数	参数说明				
fs.jfs.cache.oss.credentials.provider	com.aliyun.emr.fs.auth.EnvironmentVariableCredential sProvider				
ALIYUN_ACCESS_KEY_ID	OSS的AccessKey ID。				
ALIYUN_ACCESS_KEY_SECRET	OSS的AccessKey Secret。				
	OSS的SecurityToken(临时安全令牌)。				
ALIYUN_SECURITY_T OKEN	⑦ 说明 仅配置有时效Token时需要。				

## • InstanceProfileCredentialsProvider

该方式无需配置AccessKey,可以免密方式访问OSS。

参数	参数说明
fs.jfs.cache.oss.credentials.provider	com.aliyun.emr.fs.auth.InstanceProfileCredentialsProvider

# **10.4. JindoTable** 10.4.1. JindoTable使用说明

JindoTable提供表或分区级别的热度统计、存储分层和表文件优化的功能。本文为您介绍JindoTable的使用方法。

## 前提条件

- 本地安装了Java JDK 8。
- 已创建EMR-3.30.0或后续版本的集群,详情请参见创建集群。

## 使用JindoTable

常见命令如下:

- -accessStat
- -cache
- -archive
- -unarchive
- -uncache
- -status
- -optimize
- -showTable
- -showPartition

## E-MapReduce

- -listTables
- -dumpmc

↓ 注意 指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。

#### -accessStat

语法

jindo table -accessStat {-d} <days>{-n} <topNums>

功能

查询在指定时间范围内,访问最多的N条表或分区的记录。

<days>和<topNums>应为正整数。天数为1时,表示查询从本地时间当天0:00开始到现在的所有访问记录。

• 示例:查询近七天,访问最多的20条表或分区的记录。

jindo table -accessStat -d 7 -n 20

- -cache
- 语法

jindo table -cache {-t} <dbName.tableName>[-p] <partitionSpec>[-pin]

功能

表示缓存指定表或分区的数据至集群本地磁盘上。

表或分区的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partitionCol2=2,... 的格式。指定 -pin 时,在缓存空间不足时尽量不删除相关数据。

• 示例:缓存2020-03-16日db1.t1表的数据至本地磁盘上。

jindo table -cache -t db1.t1 -p date=2020-03-16

#### -uncache

● 语法

jindo table -uncache {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示删除集群本地磁盘上指定表或分区的缓存数据。

对应的路径需要位于OSS或JindoFS。指定表时使用 database.table 的格式,指定分区时使用 partitionColl=1,partition col2=2,... 的格式。

- 示例:
  - 删除集群本地磁盘上表db1.t2的缓存数据。

jindo table -uncache -t db1.t2

○ 删除集群本地磁盘上表db1.t1中指定分区的缓存数据。

jindo table -uncache -t db1.t1 -p date=2020-03-16,category=1

- -archive
- 语法

jindo table -archive {-a|i} {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示降低表或者分区的存储策略级别,默认改为归档存储。

加上-i使用低频存储。指定表时使用database.table的格式,指定分区时使用`partitionCol1=1,partitionCol2=2,...`的格式。

```
• 示例:指定表db1.t1缓存至本地磁盘上。
```

jindo table -archive -t db1.t1 -p date=2020-10-12

- -unarchive
- 语法

jindo table -unarchive [-o|-i] {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示将归档数据转为标准存储。

-o 将归档数据转为解冻, -i 将归档数据转为低频。

● 示例

jindo table -unarchive -o -t db1.t1 -p date=2020-03-16,category=1

jindo table -unarchive -i -t db1.t2

## -status

● 语法

jindo table -status {-t} <dbName.tableName>[-p] <partitionSpec>

● 功能

表示查看指定表或者分区的存储状态。

- 示例:
  - 。 查看表db1.t2的状态。

jindo table -status -t db1.t2

○ 查看表db1.t1在2020-03-16日的状态。

jindo table -status -t db1.t1 -p date=2020-03-16

## -optimize

● 语法

jindo table -optimize {-t} <dbName.tableName>

• 功能

优化表在存储层的数据组织。

• 示例:优化表db1.t1在存储层的数据组织。

jindo table -optimize -t db1.t1

## -showTable

● 语法

jindo table -showTable {-t} <dbName.tableName>

功能

如果是分区表,则展示所有分区;如果是非分区表,则返回表的存储情况。

• 示例: 展示db1.t1分区表的所有分区。

jindo table -showTable -t db1.t1

- -showPartition
- 语法

jindo table -showPartition {-t} <dbName.tableName>[-p] <partitionSpec>

功能

表示返回分区的存储情况。

• 示例: 返回分区表db1.t1在2020-10-12日的存储情况。

jindo table -showPartition -t dbl.tl -p date=2020-10-12

## -listTables

● 语法

jindo table -listTables [-db] [dbName]

● 功能

展示指定数据库中的所有表。不指定 [-db] 时默认展示default库中的表。

- 示例:
  - 展示default库中的表。

jindo table -listTables

○ 列出数据库db1中的表。

jindo table -listTables -db db1

## -dumpmc

● 语法

jindo table -dumpmc {-i} <accessId> {-k} <accessKey> {-m} <numMaps> {-t} <tunnelUrl> {-project} <projectName> {-t} <table} <tablename> {-p} <partitionSpec> {-f} <csv{tfrecord> {-o} <outputPath>

参数	描述	是否必选
-i	阿里云的AccessKey ID。	是
-k	阿里云的AccessKey Secret。	是
-m	map任务数。	是
-t	MaxCompute的VPC网络Tunnel Endpoint。	是
-project	Maxcompute的项目空间名。	是
-table	Maxcompute的表名。	是
-p	分区信息 。例如 pt=xxx , 多个分区时 用英文逗号 (,) 分 开 pt=xxx,dt=xxx 。	否
-f	文件格式。包括: 。 tfrecord 。 csv	是
-0	目的路径。	是

• 功能

表示Dumpmc Maxcompute表至EMR集群或OSS存储。支持CSV格式和TFRECORD格式。

• 示例:

○ Dumpmc Maxcompute表(TFRECORD格式)至EMR集群。

jindo table -dumpmc -m 10 -project mctest\_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o /tmp/outputtf1 -f tfrecord

• Dumpmc Maxcompute表(CSV格式)至OSS存储。

```
jindo table -dumpmc -m 10 -project mctest_project -table t1 -t http://dt.xxx.maxcompute.aliyun-inc.com -k xxxxxxxx -i XXXXXX -o oss://bucket1/tmp/outputcsv -f csv
```

## 10.4.2. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

## 前提条件

已创建表或者视图。

## 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的DataSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

## JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dataset、DataFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spark 2.4.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manage	ment									

JindoCube还提供了**spark.sql.cache.useDatabase**参数,可以针对业务方向,按不同的业务建立database,把需要建 cache的view放在这个database中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPartition**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为t**rue**。

## JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击**集群管理**页签。
  - iii. 单击待操作集群所在行的集群ID。

- iv. 单击左侧导航栏的**访问链接与端口**。
- v. 在**公网访问链接**页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。 Knox相关使用说明请参见Knox。
- vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
- vii. 单击上方的Cube Management页签。
- viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:

Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。 在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Table and View Informatio	n		_	
dataBase	tableName	CubeCache	ac	tion
ssb_10	customer	c_name c_address	rav	w cache Ibe cache
ssb_10	dates	c_city c_nation	rav	w cache Jbe cache
ssb_10	lineorder	c_phone c_mktsegment	rav	w cache ibe cache
ssb_10	lineorder_flatten	□ p_partkey □ p_name	rav	w cache Jbe cache
ssb_10	p_lineorder	p_ingi p_category p_brand	rav	w cache ibe cache
ssb_10	part	p_color p_type	rav	w cache Jbe cache
ssb_10	supplier	p_size p_container	rav	w cache Ibe cache
		lo_quantity   MAX    MAX		
		lo_tax + SUM + -		
		Cancle OK		

JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

SELECT c\_city, c\_nation, c\_region, MAX(lo\_quantity), SUM(lo\_tax)
FROM lineorder\_flatten
GROUP BY c\_city, c\_nation, c\_region;

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
MAX	MAX
MIN	MIN
AVG	COUNT, SUM
COUNT (DISTINCT)	PRE_COUNT_DISTINCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Det ail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作,Cache表示的数据并未持久化,需要继续构建Cache,从而持久化Cache数据到 HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新,需要更新Cache中的数据从而保持一致。JindoCube 支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spark 2.4.3	Jobs Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	nent Build P	age						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>							
Optional Filter								
Column:	s_region	÷						
Filter Type:	<ul> <li>Fixed Values</li> <li>Range Values</li> </ul>							
Values:	ASIA,AMERICA		Filter view data wit	h column value eq	uals to inpu	ut values.		
	Submit							

在构建JindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。 ■ Column:过滤字段。 ■ Filter Type:过滤类型,支持固定值和范围值两种。 ■ Fixed Values:指定过滤值,可以多个,以","分隔。 ■ Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

SELECT \* FROM lineorder\_flatten
WHERE s\_region == 'ASIA' OR s\_region == 'AMERICA';

单击**Submit**,提交构建任务,返回到Cache详细页面,对应的构建任务会提交到Spark集群中执行,在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

Spork 2.4.3	Jobs Stages Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manage	ment Timer Trigger B	Build Page	е				
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Trigger Strategy:							
Start At:	2019-12-30 11:00:00						
Period:	1	Hour		\$			
Optional Step:	۵						
Step By:	TimeStamp Column(Long Type)     DateTime Column(String Type)						
Column Name:	lo_orderdate \$						
DateTime Format	Such as 'yyyy-MM-dd HI If availab	ole, filter would com	oare column as Tin	nestamp	Type value.		
	Submit						

定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由SparkDriver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、 构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之 释放。

3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

◦ 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3 Jobs	Stages Storage Environn	nent Executors SQL	Cube Management Streaming	JDBC/ODBC Server		Thr	ift JDBC/ODBC Server application UI
Cube Management	:						
viewDataBase	viewName	cacheName	enableRewrite	cacheType	cacheDataSize	lastUpdataTime	action
ssb_10	lineorder_flatten	lo_raw_cache	false	Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache							

○ 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 24.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manage	ment	Detail I	Page							
<ul> <li>Basic Cache Information</li> </ul>	1									
Name									Value	
Database									ssb_10	
View Name									lineorder_flatten	
Cache Name									lo_raw_cache	
Enabled									true	
Cache Type									RAW	
Cache Columns									lo, orderkey, lo, linenumber, lo, cuativey, lo, cupriley, lo, cupriley, lo, currentaria, lo, orderprototi, lo, and lo, carditalapito, e, al solacunti, la, verevano, lo, supplycost, lo, alca, lo, committatio, lo, safimode, s. sup e, region, s, phone, d_ catalexe, d_ catale, d_ algoviewei, d_, anonth, d_, year, d_ yearmonthmum, d_ yearmo (_dynuminyee), c, name, c_ adateses, c, chy, c_ nation, c_ region, c_ phone, c_ mitsegment, p_ partikey, p_ na p_ type, p. Size, container	priotity, io_quantity, io_extendedprice, pkeys, a_name, s_address, s_city, s_nation, nth, d_daynuminwoek, d_daynuminmonth, daynmonthit, d_holidayfi, d_weekdayfi, ime, p_mfgr, p_category, p_brand, p_color,
Location									/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw	
Data Size									None	
Provider									PARQUET	
Partition By									lo_orderdate	
ZOrder By										
Last Update Time									2019/12/28 17:52:55	
Enabled Partition Information Show 10 ¢ entries										Search:
Path						Siz	e		Action	
								No data av	allable in table	
Showing 0 to 0 of 0 entries	3									Previous Next
Build Information     Latest cache building is fin	ished at 20	019-12-28 1	7:52:55							
Build Cache Trigger	Period Buil	d								

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delet**e删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ 说明 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

col_name	data_type	comment	
o orderkev	higint		
o_linenumber	bigint	NULL	
o_custkey	int	NULL	
o_partkey	int	NULL	
o_suppkey	int	NULL	
o_orderdate	int	NULL	
o_orderpriotity	string	NULL	
o quantity	hight	NULL	
o extendedprice	bigint	NULL	
o_ordtotalprice	bigint	NULL	
o_discount	bigint	NULL	
o_revenue	bigint	NULL	
o_supplycost	bigint	NULL	
.o_tax	bigint	NULL	
o shipmode	string	NULL	
suppkey	int	NULL	
name	string	NULL	
address	string	NULL	
_city	string	NULL	
_nation	string	NULL	
_region	string	NULL	
_pnone	string	NULL	
	int etring	NULL	
davofweek	string	NULL	
month	string	NULL	
vear	int	NULL	
_yearmonthnum	int	NULL	
_yearmonth	string	NULL	
_daynuminweek	int	NULL	
_daynuminmonth	int	NULL	
_daynuminyear	int	NULL	
_monthnuminyear	int	NULL	
sellingseason	string	NULL	
lastdavinweekfl	int	NULL	
_lastdayinmonthfl	int	NULL	
L_holidayfl	int	NULL	
_weekdayfl	int	NULL	
_custkey	int	NULL	
_name	string	NULL	
address	string	NULL	
nation	string	NULL	
region	string	NULL	
phone	string	NULL	
_mktsegment	string	NULL	
_partkey	int	NULL	
_name	string	NULL	
_mtgr	string	NULL	
brand	string	NULL	
color	string	NULL	
type	string	NULL	
size	int	NULL	
_container	string	NULL	
Detailed Table Information			
Jatabase	ssb_10		
able	hadoon		
Created Time	Sat Dec 28 17:30:02 CST 2019		
ast Access	Thu Jan 01 08:00:00 CST 1970	i i	
created By	Spark 2.2 or prior	i	
ype	VIEW	i i	
/iew Text	SELECT `lineorder`.`lo_orderkey`, `lineorder`.`lo_1:	inenumber`,	`lineorder`.`lo_custkey`, `lineorder`.`lo_partkey`, `
eorder'. lo_suppkey', 'lineord '. lo_extendedprice', 'lineord .o_tax', 'lineorder'. lo_commit	er`.`lo_orderdate`, `lineorder`.`lo_orderpriotity`, er`.`lo_ordtotalprice`, `lineorder`.`lo_discount`, ` date`, `lineorder`.`lo_shipmode`, `supplier`.`s_suppl	lineorder`. lineorder`.` key`, `suppl	`lo_shippriotity`, `lineorder`.`lo_quantity`, `lineor `lo_revenue`, `lineorder`.`lo_supplycost`, `lineorder` Lier`.`s_name`, `supplier`.`s_address`, `supplier`.`s_
y , supplier . s_nation , sup ), 'dates`.`d_year`, 'dates`.`d dates`.`d_monthnuminyear`, 'dat	oplier . s_region , supplier . s_phone , dates . d d_yearmonthnum`, `dates`.`d_yearmonth`, `dates`.`d_d tes`.`d_weeknuminyear`, `dates`.`d_sellingseason`, `u	_datekey , aynuminweek` dates`.`d_la	dates.d_date, dates.d_dayorweek, dates.d_mo , dates`.d_daynuminmonth', dates`.d_daynuminyear` astdayinweekf1`, `dates`.`d_lastdayinmonthf1`, `dates`
noiidayTi , dates . d_weekday customer`.`c_region`, `customer t`.`p_brand`, `part`.`p_color`,	yrı, customer.c_custkey, customer.c_name', r`.`c_phone`, `customer`.`c_mktsegment`, `part`.`p_p; , `part`.`p_type`, `part`.`p_size`, `part`.`p_contain	artkey`, `pa ner` FROM `s	auoress , customer . c_city , customer . c_nation' srt`.p_name`, part``p_mfgr`, part``p_category`, ssb_10`.`lineorder`, `ssb_10`.`supplier`, `ssb_10`.`da
; ssb_10`.`customer`, `ssb_10 RE `lineorder`.`lo_orderdate` = s_suppkey` AND `lineorder`.`lo r	ð. part = `dates`.`d_datekey` AND `lineorder`.`lo_custkey` = ɔartkey` = `part`.`p_partkey`	`customer`.	`c_custkey` AND `lineorder`.`lo_suppkey` = `supplier`
able Properties	[transient_lastDdlTime=1577677599] org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe		
PAPALLE MARK PROVIDE	AND A REAL AND A		

0: jdbc:hive2://localhost:10001/ssb\_10>

基于lineorder\_flatten视图简单查询的执行计划如下:

<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.enforce.bucketing does not exist</pre>	
19/12/30 14:19:32 WARN [min] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist	
plan	
== Physical Plan ==	
Unietchinit (lo_partkey#313], [p_partkey#359], Inner	
:- ★(8) Sort [lo_partkey#313 ASC NULLS FIRS]], talse, 0 : → - Exchange hashpartitioning(lo_partkey#313, 200)	
: +- *(7) SortMergeJoin [lo_custkey#312], [c_custkey#351], Inner : *(4) Sort [lo_custkey#312 Sort [lo_custkey#351], Sort [lo_custkey#351], Sort [lo_custkey#312 Sort [lo_custkey#351], Sort [lo_custkey#351	
: +- Exchange hashpartitioning(lo_custkey#312, 200)	
: : +- *(3) BroadcastHashJoin [lo_orderdate#315], [d_datekey#334], Inner, BuildRight : : :- *(3) BroadcastHashJoin [lo supneke#324], [s supneke#327]. Inner, BuildRight	
: : :- *(3) Filter (((isnotnull(lo_suppkey#314) && isnotnull(lo_orderdate#315)) && isnotnull(lo_custkey#312)) && isnotnull(lo_part key#313))	
: : + Scan hive ssb_10.lineorder [lo_orderkey#310L, lo_linenumber#311L, lo_custkey#312, lo_partkey#313, lo_suppkey#314, lo_ord	
erdate#31b, lo_orderpriotity#310, lo_shippriotity#317, lo_quantity#318, lo_extendedprice#3194, lo_ordtotalprice#3294, lo_ascount#3211, lo_revenue#32 21, lo_supplycost#32331, lo_tax#3244, lo_commitdate#3255, lo_shipmade#3261, HiveFableRelation `sbz_18' `lineorder', org.apache.hadoop.hive.serde2.lazy.l	
azySimpleSerDe, [lo_orderkey#316], lo_linenumber#311], lo_custkey#312, lo_bartkey#313, lo_suppkey#314, lo_orderdste#315, lo_orderpriotity#316, lo_ship priotity#317, lo_quartity#3181, lo_avtenderice#3301, lo_avtender331, lo_discourt#3211, lo_avenum#3231, lo_tax#324	
profity#23, 10_dbit(1)#36c, 10	
: : : + → BroadcastExchange HashedHelaIiohBroadcastMode(List(Cast(input[@, int, Taise] as Digint))) : : : +→ +(1) Filter isinothull(s_suppkey#327)	
: : : + - Scan hive ssb_10.supplier [s_suppkey#327, s_name#328, s_address#329, s_city#330, s_nation#331, s_region#332, s_phone# 333], HiveTableRelation 'ssb_10'.supplier', org.apache.hadoop.hive.serde2.lazy.lazySimpleSerDe, [s_uppkey#327, s_name#328, s_address#329, s_city#330	
, s_nation#331, s_region#332, s_phone#333) : : + - BroadcastFxchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))	
: +- *(2) Filter isotnull(d_dateksy836)	
i : → Scan nive ssc_iolaxes io_aktes io_akterysa, io_adversas, o_aoyoiwerkasa, o_montinesas, o_yearmontinnumesas	
f1#347, d_lastdayinmonthf1#348, d_holidayf1#349, d_weekdayf1#350], HiveTableRelation `ssb_10'. `dates', org.apache.hadoop.hive.serde2.lazy.lazySimpleSe TDe. [d dateke#334. d date#335. d datofweek#336. d wear#338. d vear#338. d vearmonth#349. d vearmonth#340. d datem315.	
2, d_daymuminybar#343, d_monthnuminybar#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweekfl#347, d_lastdayinwonthfl#348, d_holidayfl#349,	
u_weekuay1⊥#500j : +- *(6) Sort [c_custkey#351 ASC NULLS FIRST], false, 0	
: +- Exchange hashpartitioning(c_custkey#351, 200) +- +{5}. Filter ((isnotuul)(c_region#356) & (c_region#356 = ASTA)) && isnotnul](c_custkey#351))	
+- Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk	
Isegment#3b8], HivelableHelation ssb_10 - Customer , org.apache.hadoop.nive.serde2.lazy.LazySimpleSerDe, Lc_custKey#3b1, c_name#3b2, c_address#3b3, c _city#354, c_nation#355, c_region#356, c_mktsegment#358]	
+- *(10) Sort [p_partkey#359 ASC NULLS FIRST], false, 0 +- Evchance heshpartification(n_partkew#359, 200)	
+- rely Filter isothull(p_parkey#359)	
+- Scan hive ssb_10.part (p_partkey#359, p_name#360, p_mfgr#361, p_category#362, p_brand#363, p_color#364, p_type#365, p_size#366, p_conta iner#367], HiveTableRalation "ssb_10" - partkey#350, p_name#361, p_category#362, p_partkey#359, p_name#360, p_mfgr#361, p_category#3	
62, p_brand#363, p_color#364, p_type#365, p_size#366, p_container#367]	1
1 row selected (0.435 seconds)	
0: Jabc:hive2://localnost:10001/SSD_10>	
在为line order_flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计	
划,优化后的执行计划如下:	
(a. idhaibina //iarabash 10001/ab 10. analis salash t fann liarandan flabban abana a marin - 10011 liais 10.	
or jadochivez/jadochiv	
19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.enforce.buckting does not exist 19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist	
== rnysical rian == Collectimit 10	
++ +(1) Project [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderdate#133, lo_orderpriotity#134, lo_ship priotity#135, lo quantity#136L, lo extendedrice#131L, lo orderdalprice#138L, lo discourt#139L, lo revenue#144L, lo suppkey#132, lo tax#1402. lo c	
ommitdate#143, lo_shipmode#144, s_suppkey#145, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#156, s_phone#151, 34 more field]	
+- *(1) Filter (15 notnull(c_region#1/4) && (c_region#1/4 = ASIA))	

可以看到,优化后的执行计划省去了lineorder flatten视图的所有计算逻辑,直接访问HDFS中Cache的数据。

## 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性,而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据,用户需要根据查询对于数据一致性的需求,触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

# 10.5. 工具集

## 10.5.1. JindoFS FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

## 前提条件

已创建集群,详情请参见创建集群。

## 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

#### 挂载

⑦ 说明 依次在每个节点上执行挂载操作。

#### 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。

2. 执行如下命令, 新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

#### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

## 返回用户在服务端配置的所有命名空间列表。

test testcache

2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

## 3. 创建目录。

mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

## 返回如下信息。

hello world

#### 如果您想使用Python方式写入和读取文件,请参见如下示例:

#### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

2. 使用Python读文件。创建脚本read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

### 返回如下信息。

my first file This file

## 卸载

? 说明 依次在每个节点上执行卸载操作。

- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令, 卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

## 10.5.2. 分层存储命令使用说明

EMR-3.30版本JindoFS引入分层存储功能。通过该功能您可以根据数据冷热程度选择不同的存储介质来存储数据,以减少数据存 储成本,或者加速访问数据的速度。

## 使用Jindo jfs

执行以下命令,获取帮助信息。

```
jindo jfs -help archive
-archive -i/a <path> ... :
Archive commands.
```

JindoFS分层存储命令均为异步执行,分层存储命令只是启动相关任务执行。

常用命令如下:

- Cache命令
- Uncache命令
- Archive命令
- Unarchive命令
- Status命令
- ls2命令

## Cache命令

Cache命令可以备份对应路径的数据至本集群的磁盘,以便于后续可以读取本地数据,无需读取OSS上的数据。

jindo jfs -cache -p <path>

-p 参数可以保证本地数据不受磁盘水位清理。

## Uncache命令

Uncache命令可以删除本地集群中的本地备份,只存储数据在OSS标准存储上,以便于后续读取OSS上的数据。

jindo jfs -uncache <path>

## Archive命令

Archive命令可以归档存储数据,删除本地磁盘上的数据备份,归档OSS上的数据至低频访问存储或者归档存储上。存储类型请参 见对象存储OSS的存储类型介绍。

```
jindo jfs -archive -i|-a <path>
```

-i 参数可以归档数据至OSS低频存储类型。 -a 参数可以归档数据至OSS归档存储类型。

## Unarchive命令

Unarchive命令可以将数据从归档存储类型恢复到低频存储或者标准存储,同时可以临时解冻归档存储类型,使数据临时可读, 有效时间为1天。

```
jindo jfs -unarchive -i/-o <path>
```

Unarchive默认可以将数据恢复成标准存储, -i 参数可以恢复数据至低频存储类型。 -o 参数可以临时解冻归档存储类型, 使数据临时可读。

## Status命令

Status命令可以查看任务进度信息,默认会统计该路径需要执行分层存储的文件数目以及已经完成的数据。

```
jindo jfs -status -detail/-sync <path>
```

-detail 参数可以查看文件进度信息。 -sync 参数表示该命令需要同步等待分层存储任务结束才会退出。

## ls2命令

JindoFS扩展hadoop ls相关操作,提供ls2命令可以查看文件存储状态。

hadoop fs -1s2 <path>

#### 返回信息会包含文件的存储类型,示例如下。

```
drwxrwxrwx - - 0 2020-06-05 04:27 oss://xxxx/warehouse
-rw-rw-rw- 1 Archive 1484 2020-09-23 16:40 oss://xxxx/wikipedia_data.csv
-rw-rw-rw- 1 Standard 1676 2020-06-07 20:04 oss://xxxx/wikipedia_data.json
```

## 10.5.3. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

## 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

## 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见登录集群。

2. 执行以下命令,获取帮助信息。

jindo distcp --help

```
返回信息如下。
```

--help - Print help text --src=VALUE - Directory to copy files from --dest=VALUE - Directory to copy files to --parallelism=VALUE - Copy task parallelism --outputManifest=VALUE - The name of the manifest file --previousManifest=VALUE - The path to an existing manifest file --requirePreviousManifest=VALUE - Require that a previous manifest is present if specified --copyFromManifest - Copy from a manifest instead of listing a directory --srcPrefixesFile=VALUE - File containing a list of source URI prefixes --srcPattern=VALUE - Include only source files matching this pattern --deleteOnSuccess - Delete input files after a successful copy --outputCodec=VALUE - Compression codec for output files --groupBy=VALUE - Pattern to group input files by --targetSize=VALUE - Target size for output files --enableBalancePlan - Enable plan copy task to make balance --enableDynamicPlan - Enable plan copy task dynamically --enableTransaction - Enable transation on Job explicitly --diff - show the difference between src and dest filelist --ossKey=VALUE - Specify your oss key if needed --ossSecret=VALUE - Specify your oss secret if needed --ossEndPoint=VALUE - Specify your oss endPoint if needed --policy=VALUE - Specify your oss storage policy --cleanUpPending - clean up the incomplete upload when distcp job finish --queue=VALUE - Specify yarn queuename if needed --bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed --s3Key=VALUE - Specify your s3 key --s3Secret=VALUE - Specify your s3 Sercet --s3EndPoint=VALUE - Specify your s3 EndPoint

### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录,Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

```
jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp
```

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

## --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制/*data/incoming/hourly\_table/2017-02-01/03*下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

```
hdfs dfs -ls /data/incoming/hourly table/2017-02-01/03
```

#### 返回信息如下。

Found 6 items	3		
-rw-r	2 root hadoop	2252 2020-04-17 20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2 root hadoop	4891 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2 root hadoop	4891 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2 root hadoop	4891 2020-04-17 20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09			
-rw-r	2 root hadoop	1016 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2 root hadoop	1016 2020-04-17 20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,复制以log结尾的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPattern .*\.lo
g --parallelism 20
```

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-

      1
      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

#### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --deleteOnSuccess -
-parallelism 20
```

## --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03
```

#### 返回信息如下:

```
      Found 6 items

      -rw-rw-rw-1
      938 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/000151.sst.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log.gz

      -rw-rw-rw-1
      1956 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/0PTIONS-000109.g

      z
      -rw-rw-rw-1

      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz

      -rw-rw-rw-1
      506 2020-04-17 20:58 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp01.txt.gz
```

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

 none表示保存为未压缩的文件。如果文件已压缩,则Jindo DistCp会将其解压缩。

## • keep表示不更改文件压缩形态,按原样复制。

```
② 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。
```

## --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

## --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录,查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --parallel ism 20

#### 执行以下命令,查看运行期间复制的文件。

```
hadoop fs -text oss://<yourBucketName>/hourly_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst
```

#### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

#### --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

## --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
        Found 4 items

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

        drwxrwxrwx -
        0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

#### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

#### 例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

```
Found 8 items
-rw-r---- 2 root hadoop
                              2252 2020-04-17 20:42 /data/incoming/hourly table/2017-02-01/03/000151.sst
-rw-r---- 2 root hadoop
                              4891 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/1.log
-rw-r---- 2 root hadoop
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/2.log
            2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 21:08 /data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r---- 2 root hadoop
                               4891 2020-04-17 20:42 /data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09
-rw-r---- 2 root hadoop
                              1016 2020-04-17 20:47 /data/incoming/hourly table/2017-02-01/03/emp01.txt
-rw-r---- 2 root hadoop
                               1016 2020-04-17 20:47 /data/incoming/hourly_table/2017-02-01/03/emp06.txt
```

执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

## --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为Dist Cp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo Dist Cp的作业分配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableBalancePlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

## --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定\_\_\_enableDynamicPlan\_来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

## --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

#### 如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的DistCp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --dest oss://<yourB
ucketName>/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis
m 20
```

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yarn队列的名称。

命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

## --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

## 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

## 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

• 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive
--parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

## 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

### 命令示例如下。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

## 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distop jindo-distop-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的core-site.xml文件里 ,避免每次使用时填写Accesskey。

### E-MapReduce

<configuration></configuration>
<property></property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
<property></property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>

## 此时代码示例如下。

jindo distop /tmp/jindo-distop-2.7.3.jar --src s3://smartdatal/ --dest s3://smartdatal/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

## 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

```
⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的
大小可能不相等。
```

## 10.5.4. Jindo DistCp场景化使用指导

本文通过场景化为您介绍如何使用Jindo Dist Cp。

## 前提条件

- 已创建相应版本的集群,详情请参见创建集群。
- 已安装JDK 1.8。
- 根据您使用的Hadoop版本,下载jindo-dist cp-<version>.jar。
  - Hadoop 2.7及后续版本,请下载jindo-dist cp-3.0.0.jar。
  - Hadoop 3.x系列版本,请下载jindo-distcp-3.0.0.jar。

## 场景预览

Jindo Dist Cp常用使用场景如下所示:

- 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?
- 场景二:使用Jindo Dist Cp成功导完数据后,如何验证数据完整性?
- 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断点续传?
- 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Dist cp过程中可能已经产生了新文件,该使用哪些参数处理?
- 场景五:如果需要指定Jindo Dist Cp作业在Yarn上的队列以及可用带宽,该使用哪些参数?
- 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?
- 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?
- 场景八: 如果需要使用S3作为数据源, 该使用哪些参数?
- 场景九: 如果需要写入文件至OSS上并压缩(LZO和GZ格式等)时, 该使用哪些参数?
- 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy对象,该使用哪些参数?
- 场景十一: 如果想合并符合一定规则的文件, 以减少文件个数, 该使用哪些参数?
- 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?
- 场景十三: 如果不想将OSS AccessKey这种参数写在命令行里, 该如何处理?

# 场景一:导入HDFS数据至OSS,需要使用哪些参数?如果数据量很大、文件很多(百万千万级别)时,该使用哪些参数优化?

如果您使用的不是EMR环境,当从HDFS上往OSS传输数据时,需要满足以下几点:

- 可以访问HDFS,并有读数据权限。
- 需要提供OSS的AccessKey (AccessKey ID和AccessKey Secret),以及Endpoint信息,且该AccessKey具有写目标Bucket的 权限。
- OSS Bucket不能为归档类型。
- 环境可以提交MapReduce任务。
- 已下载Jindo Dist Cp JAR包。

本场景示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 10

```
⑦ 说明 各参数含义请参见Jindo Dist Cp使用说明。
```

当您的数量很大,文件数量很多,例如百万千万级别时,您可以增大parallelism,以增加并发度,还可以开启\_\_\_ enableBatch 参数来进行优化。优化命令如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 500 --enableB
atch
```

# 场景二:使用Jindo DistCp成功导完数据后,如何验证数据完整性?

您可以通过以下两种方式验证数据完整性:

• Jindo Dist Cp Counters

您可以在MapReduce任务结束的Counter信息中,获取Distcp Counters的信息。

```
Distcp Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0
```

参数含义如下:

○ Bytes Destination Copied: 表示目标端写文件的字节数大小。

- Bytes Source Read:表示源端读文件的字节数大小。
- 。 Files Copied: 表示成功Copy的文件数。
- Jindo Dist Cp -- diff

您可以使用 --diff 命令,进行源端和目标端的文件比较,该命令会对文件名和文件大小进行比较,记录遗漏或者未成功传 输的文件,存储在提交命令的当前目录下,生成manifest文件。

在场景一的基础上增加 --diff 参数即可,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

当全部文件传输成功时,系统返回如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

# 场景三:导入HDFS数据至OSS时,DistCp任务存在随时失败的情况,该使用哪些参数支持断 点续传?

在<mark>场景</mark>一的基础上,如果您的Distcp任务因为各种原因中间失败了,而此时您想支持断点续传,只Copy剩下未Copy成功的文件,此时需要您在进行上一次Distcp任务完成后进行如下操作:

1. 增加一个 --diff 命令, 查看所有文件是否都传输完成。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --diff

#### 当所有文件都传输完成,则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely.

2. 文件没有传输完成时会生成manifest文件,您可以使用 --copyFromManifest 和 --previousManifest 命令进行剩余文 件的Copy。示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table
le --dest oss://yang-hhht/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromMa
nifest --parallelism 20
```

file:///opt/manifest-2020-04-17.gz 为您当前执行命令的本地路径。

# 场景四:成功导入HDFS数据至OSS,数据不断增量增加,在Distcp过程中可能已经产生了新 文件,该使用哪些参数处理?

1. 未产生上一次Copy的文件信息,需要指定生成manifest文件,记录已完成的文件信息。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --requirePreviousManifest=false 两个信息, 示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

参数含义如下:

- --outputManifest : 指定生成的manifest文件,文件名称自定义但必须以gz结尾,例如*manifest-2020-04-17.gz*,该 文件会存放在 --dest 指定的目录下。
- --requirePreviousManifest : 无已生成的历史manifest文件信息。
- 2. 当前一次Dist cp任务结束后,源目录可能已经产生了新文件,这时候需要增量同步新文件。

在场景一的基础上增加 --outputManifest=manifest-2020-04-17.gz 和 --previousManifest=oss://yang-hhht/hourly\_ table/manifest-2020-04-17.gz 两个信息,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_tab le --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

hadoop jar jindo-distcp-2.7.3.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputManifest=manife
st-2020-04-18.gz --previousManifest=oss://yang-hhht/hourly\_table/manifest-2020-04-17.gz --parallelism 10

3. 重复执行步骤2,不断同步增量文件。

# 场景五:如果需要指定Jindo DistCp作业在Yarn上的队列以及可用带宽,该使用哪些参数?

在场景一的基础上需要增加两个参数。两个参数可以配合使用,也可以单独使用。

- --queue : 指定Yarn队列的名称。
- --bandwidth : 指定带宽的大小, 单位为MB。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --queue yarnqueue --bandwid
th 6 --parallelism 10
```

# 场景六:当通过低频或者归档形式写入OSS,该使用哪些参数?

• 当通过归档形式写入OSS时,需要在在场景一的基础上增加 --archive 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --archive --parallelism 20

• 当通过低频形式写入OSS时,需要在在场景一的基础上增加 --ia 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --ia --parallelism 20

## 场景七:针对小文件比例和文件大小情况,该使用哪些参数来优化传输速度?

• 小文件较多,大文件较大情况。

如果要Copy的所有文件中小文件的占比较高,大文件较少,但是单个文件数据较大,在正常流程中是按照随机方式来进行 Copy文件分配,此时如果不做优化很可能造成一个Copy进程分配到大文件的同时也分配到很多小文件,不能发挥最好的性能。

在场景一的基础上增加 --enableDynamicPlan 开启优化选项,但不能和 --enableBalancePlan 一起使用。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableDynamicPlan --pa rallelism 10

#### 优化对比如下。



#### • 文件总体均衡,大小差不多情况。

#### 如果您要Copy的数据里文件大小总体差不多,比较均衡,您可以使用 --enableBalancePlan 优化。示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --enableBalancePlan --pa rallelism 10

#### 优化对比如下。



# 场景八:如果需要使用S3作为数据源,该使用哪些参数?

需要在场景一的基础上替换OSS的AccessKey和EndPoint信息转换成S3参数:

- --s3Key : 连接S3的AccessKey ID。
- --s3Secret : 连接S3的AccessKey Secret。
- --s3EndPoint : 连接S3的EndPoint信息。

#### 示例如下。

hadoop jar jindo-distcp-<version>.jar --src s3a://yourbucket/ --dest oss://yang-hhht/hourly\_table --s3Key you rkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com --parallelism 10

# 场景九:如果需要写入文件至OSS并压缩文件(LZO和GZ格式等)时,该使用哪些参数?

如果您想压缩写入的目标文件,例如LZO和GZ等格式,以降低目标文件的存储空间,您可以使用 --outputCodec 参数来完成。 需要在场景一的基础上增加 --outputCodec 参数,示例如下。 hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table -ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --outputCodec=gz --parallel
ism 10

Jindo DistCp支持编解码器GZIP、GZ、LZO、LZOP和SNAPPY以及关键字none和keep(默认值)。这些关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

② 说明 如您在开源Hadoop集群环境中使用LZO压缩功能,则您需要安装gplcompression的native库和hadoop-lzo包,

# 场景十:如果需要把本次Copy中符合特定规则或者同一个父目录下的部分子目录作为Copy 对象,该使用哪些参数?

● 如果您需要将Copy列表中符合一定规则的文件进行Copy,需要在场景一的基础上增加 --srcPattern 参数,示例如下。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPattern .\*\.log --p arallelism 10

--srcPattern : 进行过滤的正则表达式,符合规则进行Copy,否则抛弃。

● 如果您需要Copy同一个父目录下的部分子目录,需要在场景一的基础上增加 --srcPrefixesFile 参数。

hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly\_table --dest oss://yang-hhht/hourly\_table --ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --srcPrefixesFile file:/ //opt/folders.txt --parallelism 20

--srcPrefixesFile :存储需要Copy的同父目录的文件夹列表的文件。

示例中的folders.txt内容如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly\_table/2017-02-02

# 场景十一:如果想合并符合一定规则的文件,以减少文件个数,该使用哪些参数?

需要在场景一的基础上增加如下参数:

- --targetSize : 合并文件的最大大小, 单位MB。
- --groupBy : 合并规则, 正则表达式。

#### 示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --targetSize=10 --groupBy='
.*/([a-z]+).*.txt' --parallelism 20
```

### 场景十二:如果Copy完文件,需要删除原文件,只保留目标文件时,该使用哪些参数?

需要在<mark>场景一的基础上,增加</mark> --deleteOnSuccess 参数,示例如下。

```
hadoop jar jindo-distcp-<version>.jar --src /data/incoming/hourly_table --dest oss://yang-hhht/hourly_table -
-ossKey yourkey --ossSecret yoursecret --ossEndPoint oss-cn-hangzhou.aliyuncs.com --deleteOnSuccess --paralle
lism 10
```

# 场景十三:如果不想将OSS AccessKey这种参数写在命令行里,该如何处理?

通常您需要将OSS AccessKey和endPoint信息写在参数里,但是Jindo DistCp可以将OSS的AccessKey ID、AccessKey Secret和 Endpoint预先写在Hadoop的*core-site.xm*l文件里,以避免使用时多次填写的问题。

• 如果您需要保存OSS的AccessKey相关信息,您需要将以下信息保存在core-site.xml中。

# E-MapReduce

```
<configuration>
configuration>
 cname>fs.jfs.cache.oss-accessKeyId</name>
        <value>xxx</value>
</property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
</property>
        <name>fs.jfs.cache.oss-accessKeySecret</name>
        <value>xxx</value>
</property>
        <name>fs.jfs.cache.oss-endpoint</name>
        <value>oss-cn-xxx.aliyuncs.com</value>
</property>
</configuration>
```

• 如果您需要保存S3的AccessKey相关信息,您需要将以下信息保存在*core-site.xml*中。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

# 11.SmartData 2.7.3-2.7.4 11.1. JindoFS Block模式

# 11.1.1. Block模式使用说明

Block模式提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元 数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的Block模式及其使用方式。

# 背景信息

JindoFS Block模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

# 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	地				服务配置
配置搜	索				全部   smartdata-site namespace client   storage
「清榆」	λ			Q	

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

#### ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	描述	示例	
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。	
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block	
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX	
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。</li> </ul>	

#### iii. 单击确定。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

在**服务配置**区域的storage页签,修改如下参数。

服务配置						
全部   smartdata-site   namespace   client   storage						
storage.handler.threads 40						
	storage.watermark.low.ratio	0.2		0		
	storage.watermark.high.ratio	0.4		0		
	storage.oss.upload.threads	20		0		
参数		描述				
storage.watermark.high.ratio			吏用量的上水位比例,每块数据盘的」 空间到达上水位即会触发清理。默认	lindoFS数据目录占 值: 0.4。		
storage.watermark.low.ratio			量的下水位比例,触发清理后会自动 据目录占用空间清理到下水位。默认	清理冷数据,将 \值:0.2。		

```
⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。
```

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - iv. 在确认对话框中, 单击确定。

# 11.1.2. 使用Tablestore作为存储后端

JindoFS元数据服务支持不同的存储后端,本文介绍使用Tablestore(OTS)作为元数据后端时需要进行的配置。

#### 前提条件

● 已创建EMR集群。

详情请参见<mark>创建集群</mark>。

● 已创建Tablestore实例,推荐使用高性能实例。

详情请参见<del>创建实例</del>。

? 说明 需要开启事务功能。

# 背景信息

JindoFS在新版本中,支持使用Tablestore作为JindoFS元数据服务(Namespace Service)的存储。一个EMR JindoFS集群可以绑 定一个Tablestore实例(Instance)作为JindoFS元数据服务的存储介质,元数据服务会自动为每个Namespace创建独立的 Tablestore表进行管理和存储元数据信息。

元数据服务(双机Tablestore和HA)架构图如下所示。



# 配置Tablestore

使用Tablestore功能,需要把创建的Tablestore实例和JindoFS的Namespace服务进行绑定,详细步骤如下:

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	● 正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	k λ		Q	全部   smartdata-site namespace client   storage

3. 配置以下参数。

例如,在华东1(杭州)地域下,创建了emr-jfs的Tablestore实例,EMR集群使用VPC网络,访问Tablestore的AccessKey ID 为kkkkkk,Access Secret为XXXXXX。

参数	参数说明	是否必选	示例
namespace.backend.typ e	设置namespace后端存储类 型,支持: o rocksdb o ots o raft 默认为rocksdb。	是	ots
namespace.ots.instance	Tablestore实例名称。	是	emr-jfs
namespace.ots.accessKe y	Tablestore实例的AccessKey ID。	否	kkkkk
namespace.ots.accessSe cret	Tablestore实例的AccessKey Secret。	否	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地 址,普通EMR集群,推荐使用 VPC地址。	是	http://emr-jfs.cn-hangzhou .vpc.tablestore.aliyuncs.co m

## 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击**确定**。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。

# 配置Tablestore(高可用方案)

针对EMR的高可用集群,可以通过配置开启Namespace高可用模式。



Namespace高可用模式采用Active和Standby互备方式,支持自动故障转移,当Active Namespace出现异常或者异常中止时, 客户端可以请求自动切换到新的Active节点。



- 1. 进入SmartData的namespace服务配置,配置以下参数。
  - i. 修改jfs.namespace.server.rpc-address值为emr-header-1:8101,emr-header-2:8101。
  - ii. 单击右上角的自定义配置,添加namespace.backend.ots.ha为true。
  - ⅲ. 单击确定。
  - iv. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
- 2. 单击右上角的操作 > 重启Jindo Namespace Service。
- 3. 单击右上角的操作 > 重启Jindo Storage Service。

# 11.1.3. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进行的相关配置。

# 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Tablestore(OTS)或者Raft作为元数据后端,详情请参见使用Tablestore作为存储后端和使用Raft-RocksDB-Tablestore作为存储后端。

#### 单机RocksDB作为元数据服务的架构图如下所示。



# 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	● 正常	
状态	部署拓扑	配置	配置修改历史	
配置过	过滤			服务配置
配置搜	索 入		Q	全部 smartdata-site namespace client storage

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。

# 11.1.4. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

# 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

⑦ 说明 需要开启事务功能。

• 创建3 Master的EMR集群,详情请参见创建集群。

₩ 高可用	高可用: 🕢		
	部署方式: 📀	🔾 2 Master	
	0	3 Master	查看服务部署
? 说明	如果没有部署方	5式,请 <mark>提交</mark>	工单处理。

# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。



元数据服务-多机Raft-RocksDB-Tablestore+HA如下图所示。

# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在**集群管理**页面,单击相应集群所在行的**详情**。
  - v. 在左侧导航栏, 单击集群服务 > Smart Data。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的namespace页签。
  - i. 在左侧导航栏, 单击集群服务 > Smart Data。
  - ii. 单击**配置**页签。

iii. 在**服务配置**区域,单击namespace页签。

4. 在SmartData服务的namespace页签,设置如下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: o rocksdb o ots o raft 默认为rocksdb。	raft
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值) 。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤
 7。

# 5. (可选)配置远端OTS异步存储。

在SmartData服务的namespace页签,设置如下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret	Tablestore实例的AccessKey Secret。	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。 ⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	true

6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 7. 单击右上角的操作 > 启动All Components。

## 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
(文件夹个数) (文件个数)
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail



- iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。
- 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端。

3. 初始化配置。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	true

- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中,输入执行原因,开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 启动All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus -detail

如图所示, LEADER节点的state为FINISH表示恢复完成。

```
[RaftPeerImpl]
peer_id: _____ :8103:0
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: NO stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
vote_timer: timeout(5000ms) STOPPED
stepdown_timer: timeout(5000ms) SCHEDULING(in 3382ms)
snapshot_timer: timeout(600000ms) SCHEDULING(in 474855ms)
storage: [1, 153]
disk_index: 153
known_applied_index: 153
last_log_id: (index=153,term=2)
first_index_pinned: 1
state_machine: Idle
last_committed_index: 153
last_snapshot_index: 1
last_snapshot_term: 2
 snapshot_status: IDLE
replicator_1116691496965@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0 replicator_3311419785217@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0
 [Recovery From OTS Status]
 state: FINISH
 total rows: 1484409
 table `jfs_block_test` 2 rows.
table `jfs_namespace_cache_ns` 1 rows.
 table `jfs_namespace_test` 1484406 rows.
```

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

#### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021, ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的namespace页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
namespace.backend.raft.recovery. mode	是否开启从OTS恢复元数据,包括: 。 true 。 false	false

#### 9. 重启集群。

- i. 单击上方的集群管理页签。
- ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 11.1.5. JindoFS权限功能

本文介绍JindoFS的Block模式支持的文件系统权限功能,包括Unix权限和Ranger权限两种。

# 背景信息

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。



Block模式支持Unix权限和Ranger权限两种文件系统权限功能:

- Unix权限:可以使用文件的777权限。
- Ranger权限:可以使用Ranger路径通配符等高级配置。

# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	● 正常		
状态	部署拓扑	配置	配置修改历史		
配置过	过滤				服务配置
配置搜索			全部 smartdata-site <b>namespace</b> client storage		
<b>遺輸</b>	λ		C	2	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

# ⅲ. 单击确定。

5. 重启配置。

i. 单击右上角的操作 > 重启 Jindo Namespace Service。

ii. 输入执行原因, 单击确定。

开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

```
hadoop fs -chmod 777 jfs://{namespace_name}/dir1/file1
hadoop fs -chown john:staff jfs://{namespace_name}/dir1/file1
```

如果用户对某一个文件没有权限,将返回如下错误信息。

```
[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/
ls: java.security.AccessControlException: Permission denied. Server Exception
```

# 启用JindoFS Ranger权限

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因,单击**确定**。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见概述。

ii. Ranger UI添加HDFS service。

Ranger	Access Manager	🗅 Audit	© Settings				🔂 admin
Service Mar	Service Manager						
Service Man	ager						Import Export
Вн	DFS		+22	HBASE	+ 22	🕞 HIVE	+ 🛛 🖸
				emr-hbase	• 2 🕯	emr-hive	

iii. 配置相关参数。

参数	说明	
Service Name	jfs-{namespace_name}。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal	不填写。	
dfs.namenode.kerberos.principal		
dfs.secondary.namenode.kerberos.principal		
Add New Configurations		

#### iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,才能获取LDAP的用户组 信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在namespace页签, 单击自定义配置。
- 2. 在新增配置项对话框中,设置以下参数配置LDAP,单击确定。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- ⑦ 说明 配置项请遵循开源HDFS内容。
- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。

详情请参见Ranger Usersync集成LDAP。

# 11.1.6. Jindo AuditLog使用说明

Jindo AuditLog提供块存储模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

# 前提条件

- 已创建EMR-3.29.x版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

# 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件 不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数,清理策略等。因为JindoFS AuditLog提供分析功 能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

#### 块存储模式记录的审计信息参数如下所示。

参数	描述
时间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	块存储模式Namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

# 使用AuditLog

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。

- 2. 进入namespace服务配置。
  - i. 单击**配置**页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	过滤			服务配置	
配置搜	康			全部 smartdata-site n	amespace client storage

#### 3. 配置如下参数。

- i. 在namespace页签, 单击右上角的自定义配置。
- ii. 在新增配置顶对话框中,新增如下参数。

参数	描述	是否必填
namespace.auditlog.enable	<ul> <li>true:打开AuditLog功能。</li> <li>false:关闭AuditLog功能。</li> </ul>	是
namespace.auditlog.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.auditlog.oss.accessKey	存储OSS的AccessKey ID。	否
namespace.auditlog.oss.accessSecre t	存储OSS的AccessKey Secret。	否
namespace.auditlog.oss.endpoint	存储OSS的Endpoint。	否

#### iii. 单击部署客户端配置。

- iv. 在执行集群操作对话框中,输入执行原因,单击确定。
- ∨. 在**确认**对话框中,单击**确定**。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则,在创建生命周期规则配置各项参数。
  - 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击确定。

# 使用Jindo auditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供Shell命令的分析功能,通过MR任务分析Log文件,提供Top-N活跃操作命令分析、 Top-N活跃IP分析。您可以使用 jindo auditlog 命令,使用该功能。

## Jindo Auditlog的参数说明如下表。

参数	描述	是否必填
src	存储AuditLog的OSS Bucket。默认为 <mark>步骤</mark> <mark>3</mark> 中namespace.auditlog.oss.uri的值,您 也可以自定义该参数。	否
ns	指定待分析的Namespace。默认为block模 式下所有ns。	否
type	指定分析: • ip: IP地址活跃度。 • cmd: 操作命令活跃度。	是
min	指定时间范围,分钟级别。	否
day	指定时间范围,天级别。 day 1,表示当天。	<ul><li>⑦ 说明min和day, 需要二选</li><li>一。</li></ul>

#### 在E-MapReduce控制台,创建MR类型作业,作业内容示例如下。

jindo auditlog --src oss://<yourbucket>/auditlog/ --ns test --type ip --day 1 --top 2

#### 返回信息如下。

```
16 openFileStatusRequest
```

```
6 deleteFileletRequest
```

# 11.2. JindoFS Cache模式

# 11.2.1. JindoFS缓存模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主要介绍JindoFS的缓存模式及其使用方式。

## 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

# 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

• JFS Scheme

详情请参见配置JFS Scheme。

# 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改 即可运行。

# 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面, 单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	●正常				
状态	部署拓扑	配置	配置修改历史				
配置过	过滤			服务配置			
配置搜	<b>ξ</b>		Q	全部   smartdata-site	namespace	client ∣ storag	e

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例	
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。	
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache	

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面,单击client 页签。
- 2. 修改jfs.cache.data-cache.enable为1,表示启用缓存。

此配置为客户端配置,不需要重启SmartData服务。

缓存启用后,Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情请参 见<mark>磁盘空间水位控制</mark>。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

#### 在**服务配置**区域的storage页签,修改如下参数。

服务配置 全部   smartdata-site   namespace	client <b>storage</b>			
	storage.handler.threads	40		0
	storage.watermark.low.ratio	0.2		0
	storage.watermark.high.ratio	0.4		0
	storage.oss.upload.threads	20		0
参数		描述		
storage.watermark.high.ratio		表示磁盘( 用的磁盘3	吏用量的上水位比例,每块数据盘的J 空间到达上水位即会触发清理。默认(	indoFS数据目 值: 0.4。
storage.watermark.low.ratio		表示使用 JindoFS数	量的下水位比例,触发清理后会自动; 据目录占用空间清理到下水位。默认	清理冷数据 <i>,</i> 将 、值:0.2。

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 单击右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数。
  - ⅲ. 单击确定。
  - Ⅳ. 在确认对话框中,单击确定。

# 访问OSS bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面,单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss-accessKeyId	表示存储后端OSS的AccessKey ID。
fs.jfs.cache.oss-accessKeySecret	表示存储后端OSS的AccessKey Secret。
fs.jfs.cache.oss-endpoint	表示存储后端OSS的endpoint。

- JFS Scheme
  - i. 在集群服务 > Smart Dat a的配置页面,单击bigboot页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。

- OSS Scheme
- JFS Scheme

```
i. 在集群服务 > Smart Dat a的配置页面,单击namespace页签。
```

ii.

iii.

# 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在服务配置区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

● 在服务配置区域的smart dat a-sit e页签, 配置以下参数。

参数	参数说明			
	rename过程使用普通copy接口的文件大小上限(小于阈值的使用 普通 copy接口,大于阈值的使用multipart copy接口以提高copy 效率)。			
fs.jfs.cache.copy.simple.max.byte	⑦ 说明 如果确认已开通OSS fast copy功能,参数值设为-1,表示所有大小均使用普通copy接口,从而有效利用fast copy获得最优的rename性能。			
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为8MB,如 果作业同时打开的写入流较多导致内存使用过大,可以适当调小此 参数。默认值:1048576。			
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。			
fs.oss.committer.magic.enabled	文件写入流的buffer大小,参数值必须为2的幂次,最大为8MB,如 果作业同时打开的写入流较多导致内存使用过大,可以适当调小此 参数。默认值:1048576。			

# 11.2.2. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce(简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

## 前提条件

适用环境: ECS(EMR环境外)+Hadoop+JavaSDK。

#### 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smart dat a-aliyun-jf s-\*.jar*。如果要使用Spark则 需要把*/opt/apps/spark-current /jars/*里面的包也删除,从而可以正常使用。

# 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下**一步**。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击**完成**。

# 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。
- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为**系统策略**或自定义策略。
- 6. 输入策略名称。
- 7. 单击**确定**。

## 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏,单击**实例与镜像 > 实例**。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。

付费方式	t -	连接状态	2	操作
按量 2019年7	7月26日 09:16 创建	-	看 更改实例	管理   远程连接 列规格   <u>更多</u> ▼
按量		_	购买相同配	置
2019年7	7月25日 09:47 创建		实例状态	•
按量-托 2019年	修改实例属性		实例设置	
按量	设置用户数据		密码/密钥	•
2019年	授予/收回RAM角色	5	资源变配	•
按量 2019年	编辑标签		磁盘和镜像	
本目	连接帮助		网络和安全	组 🕨
2019年	调整宿主机部署		运维和诊断	•
按量	调整实例所属部署	集		管理
2019年	保存为启动模板		更改实例	列规格   更多 ▼
按量-抢	占式实例		行	管理   远程连接

5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

## 步骤四:在ECS上设置环境变量

执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

# 步骤五:测试免密方式访问的方法

1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

2. 使用Hadoop FileSystem访问OSS。

JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
       FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
       while (iterator.hasNext()) {
           LocatedFileStatus fileStatus = iterator.next();
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
       }
   }
}
```

# 11.2.3. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

# 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后,MapReduce Application Master或Spark Driver执行最终的Job Commit操作时,会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

# 在MapReduce中使用Jindo Job Committer

- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处, 根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在服务配置区域, 单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.JindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的**mapred-site**页签,设

置mapreduce.outputcommitter.factory.scheme.oss为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

#### ⅲ. 单击确定。

② 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.jindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

# 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

#### 6. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击**确定**。

# 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。

默认值为8。

3. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 11.2.4. JindoFS权限功能

本文介绍JindoFS的Block模式支持的文件系统权限功能,包括Unix权限和Ranger权限两种。

# 背景信息

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。



Block模式支持Unix权限和Ranger权限两种文件系统权限功能:

- Unix权限:可以使用文件的777权限。
- Ranger权限:可以使用Ranger路径通配符等高级配置。

# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	Data 🗸	●正常	
状态	部署拓扑	配置	配置修改历史	
配置过	滤			服务配置
配置搜	<b>秦</b>		Q	全部   smartdata-site namespace client   storage

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的**保存**。

ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

- iii. 单击确定。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因, 单击**确定**。

开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

```
hadoop fs -chmod 777 jfs://{namespace_name}/dirl/file1
hadoop fs -chown john:staff jfs://{namespace_name}/dirl/file1
```

如果用户对某一个文件没有权限,将返回如下错误信息。

```
[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/
ls: java.security.AccessControlException: Permission denied. Server Exception
```

# 启用JindoFS Ranger权限

- 1. 添加Ranger。
  - i. 在namespace页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见<mark>概述</mark>。

ii. Ranger Ul添加HDFS service。

R	anger	Access Manager	🗅 Audit	Settings				🔥 admin
1	Service Man	ager						
S	Service Mana	ager						C Import Export
	CS HI	255				± 20		+ <b>2</b> ຄ
		013		TOO	IDASE	- G G	CHIVE	+ 3 2
					emr-hbase	<ul> <li>Image: Image: ></ul>	emr-hive	• 7 8

#### iii. 配置相关参数。

参数	说明	
Service Name	jfs-{namespace_name}。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal		
dfs.namenode.kerberos.principal	7.结束	
dfs.secondary.namenode.kerberos.principal	小浴山。	
Add New Configurations		

#### iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,才能获取LDAP的用户组 信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在namespace页签,单击**自定义配置**。
- 2. 在新增配置项对话框中,设置以下参数配置LDAP,单击确定。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

- ⑦ 说明 配置项请遵循开源HDFS内容。
- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击确定。
- 5. 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。 详情请参见Ranger Usersync集成LDAP。

# 11.2.5. Jindo AuditLog使用说明

Jindo AuditLog提供块存储模式的审计功能,记录Namespace端的增加、删除和重命名操作信息。

# 前提条件

- 已创建EMR-3.29.x版本的集群,详情请参见创建集群。
- 已创建存储空间,详情请参见创建存储空间。

# 背景信息

AuditLog可以分析Namespace端访问信息、发现异常请求和追踪错误等。JindoFS AuditLog存储日志文件至OSS,单个Log文件不超过5 GB。基于OSS的生命周期策略,您可以自定义日志文件的保留天数,清理策略等。因为JindoFS AuditLog提供分析功能,所以您可以通过Shell命令分析指定的日志文件。

# 审计信息

#### 审计信息示例。

2020-07-09 18:29:24.689 allowed=true ugi=hadoop (auth:SIMPLE) ip=127.0.0.1 ns=test-block cmd=CreateFileletReq uest src=jfs://test-block/test/test.snappy.parquet dst=null perm=::rwxrwxr-x

#### 块存储模式记录的审计信息参数如下所示。

参数	描述
间	时间格式yyyy-MM-dd hh:mm:ss.SSS。
allowed	本次操作是否被允许: • true • false
ugi	操作用户(包含认证方式信息)。
ip	Client IP。
ns	块存储模式Namespace的名称。
cmd	操作命令。
src	源路径。
dest	目标路径,可以为空。
perm	操作文件Permission信息。

# 使用AuditLog

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。

- 2. 进入namespace服务配置。
  - i. 单击配置页签。
  - ii. 单击namespace。

< 返回	💼 Smart	tData 🗸	● 正常			
状态	部署拓扑	配置	配置修改历史			
피우					叩성되었	
配直接	135.			0	全部 smartdata-site	namespace client storage

#### 3. 配置如下参数。

- i. 在namespace页签,单击右上角的自定义配置。
- ii. 在新增配置顶对话框中,新增如下参数。

参数	描述	是否必填
namespace.auditlog.enable	■ true: 打开AuditLog功能。 ■ false: 关闭AuditLog功能。	是
namespace.auditlog.oss.uri	存储AuditLog的OSS Bucket。 请参 见oss:// <yourbucket>/auditLog格式 配置。 <yourbucket>请替换为待存储的Bucket 的名称。</yourbucket></yourbucket>	是
namespace.auditlog.oss.accessKey	存储OSS的AccessKey ID。	否
namespace.auditlog.oss.accessSecre t	存储OSS的AccessKey Secret。	否
namespace.auditlog.oss.endpoint	存储OSS的Endpoint。	否

#### ⅲ. 单击部署客户端配置。

- iv. 在执行集群操作对话框中,输入执行原因,单击确定。
- ∨. 在**确认**对话框中,单击**确定**。
- 4. 重启服务。
  - i. 单击右上角的操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。
- 5. 配置清理策略。

OSS提供了lifeCycle功能来管理OSS上文件的生命周期,您可以利用该功能来自定义Log文件的清理或者保存时间。

- i. 登录 OSS管理控制台。
- ii. 单击创建的存储空间。
- iii. 在左侧导航栏, 单击基础设置 > 生命周期, 在生命周期单击设置。
- iv. 单击创建规则, 在创建生命周期规则配置各项参数。
  - 详情请参见<mark>设置生命周期规则</mark>。
- v. 单击**确定**。

# 使用Jindo auditLog分析功能

JindoFS为存储在OSS上的AuditLog文件提供Shell命令的分析功能,通过MR任务分析Log文件,提供Top-N活跃操作命令分析、 Top-N活跃IP分析。您可以使用 jindo auditlog 命令,使用该功能。

## Jindo Audit log的参数说明如下表。

参数	描述	是否必填
src	存储AuditLog的OSS Bucket。默认为 <mark>步骤</mark> 3中namespace.auditlog.oss.uri的值,您 也可以自定义该参数。	否
ns	指定待分析的Namespace。默认为block模 式下所有ns。	否
type	指定分析: • ip: IP地址活跃度。 • cmd: 操作命令活跃度。	是
min	指定时间范围,分钟级别。	否
day	指定时间范围,天级别。 day 1,表示当天。	<ul><li>⑦ 说明min和day, 需要二选</li><li>一。</li></ul>

#### 在E-MapReduce控制台,创建MR类型作业,作业内容示例如下。

jindo auditlog --src oss://<yourbucket>/auditlog/ --ns test --type ip --day 1 --top 2

#### 返回信息如下。

```
16 openFileStatusRequest
```

```
6 deleteFileletRequest
```

# 11.3. JindoTable

# 11.3.1. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

# 前提条件

已创建表或者视图。

# 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的DataSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

## JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dataset、DataFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spark 24.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Management										
New Cache										

JindoCube还提供了**spark.sql.cache.useDat abase**参数,可以针对业务方向,按不同的业务建立dat abase,把需要建 cache的view放在这个dat abase中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPart it ion**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为**true**。

# JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击集群管理页签。
  - iii. 单击待操作集群所在行的集群ID。
  - Ⅳ. 单击左侧导航栏的**访问链接与端口**。
  - v. 在**公网访问链接**页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。 Knox相关使用说明请参见Knox。
  - vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
  - vii. 单击上方的Cube Management页签。
  - viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:
Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

# ■ Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。

在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Table and View Information			
dataBase	tableName	CubeCache	action
ssb_10	customer	c_name c_address	raw cache cube cache
ssb_10	dates	c_ctiy c_nation c_radion	raw cache cube cache
ssb_10	lineorder	c_phone c_mktsegment	raw cache cube cache
ssb_10	lineorder_flatten	p_parkey p_name p_name	raw cache cube cache
ssb_10	p_lineorder	p_category p_brand	raw cache cube cache
ssb_10	part	p_color p_type	raw cache cube cache
ssb_10	supplier	p_scet	raw cache cube cache
		lo_quantity   MAX    MAX	

JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

SELECT c\_city, c\_nation, c\_region, MAX(lo\_quantity), SUM(lo\_tax)
FROM lineorder\_flatten
GROUP BY c\_city, c\_nation, c\_region;

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
MAX	MAX
MIN	MIN
AVG	COUNT, SUM
COUNT (DISTINCT)	PRE_COUNT_DIST INCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Det ail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作,Cache表示的数据并未持久化,需要继续构建Cache,从而持久化Cache数据到 HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新,需要更新Cache中的数据从而保持一致。JindoCube 支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spork 2.4.3	Jobs Stages Storage	e Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manage	ment Build Page						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Optional Filter							
Column:	s_region	\$					
Filter Type:	<ul> <li>Fixed Values</li> <li>Range Values</li> </ul>						
Values:	ASIA, AMERICA	Filter view data w	ith column value equ	uals to inpu	t values.		
	Submit						

在构建JindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。 Column:过滤字段。 Filter Type:过滤类型,支持固定值和范围值两种。 Fixed Values:指定过滤值,可以多个,以","分隔。 Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

```
SELECT * FROM lineorder_flatten
WHERE s_region == 'ASIA' OR s_region == 'AMERICA';
```

单击**Submit**,提交构建任务,返回到Cache详细页面,对应的构建任务会提交到Spark集群中执行,在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

Spark 2.4.3	Jobs Stages Storage Environment Executors SQL Cube Management Streaming JDBC/ODBC Server
Cube Manage	ment Timer Trigger Build Page
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>
Trigger Strategy:	
Start At:	2019-12-30 11:00:00
Period:	1 Hour ¢
Optional Step:	0
Step By:	TimeStamp Column(Long Type)     DateTime Column(String Type)
Column Name:	lo_orderdate \$
DateTime Format	Such as 'yyyy-MM-dd HI If available, filter would compare column as TimestampType value.
	Submit

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由Spark Driver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、 构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之 释放。

3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

● 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3 Joi	s Stages Storage Environ	ment Executors SQL	Cube Management Streaming	JDBC/ODBC Server		Th	rift JDBC/ODBC Server application UI
Cube Manageme	ıt						
viewDataBase	viewName	cacheName	enableRewrite	cacheType	cacheDataSize	lastUpdataTime	action
ssb_10	lineorder_flatten	lo_raw_cache	false	Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache							

○ 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 2.4.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application	n UI
Cube Manage	ment	Detail I	Page								
<ul> <li>Basic Cache Information</li> </ul>	1										
Name									Value		
Database									ssb_10		
View Name									lineorder_flatten		
Cache Name									lo_raw_cache		
Enabled									true		
Cache Type									RAW		
Cache Columns									lo, ordeteke, lo, inenumber, lo, custeke, lo, partike, lo, suppkes, lo, ordetalite, lo, ordetprotito, lo, all, lo, ordetalisho, el, discunt, lo, avrevuo, lo, supplocot, D, such, conmittatino, lo, partenoto, su, a, region, a, phone, d_atakee, d_atake, d_atende, d_asende, d_aveamonthrum, d_yeam (_depunum)yea, lo, anothrum/mayea, d_atakee, d_asende, d_asende, d_aveamonthrum, d_yeam (_depunum)yea, lo, anothrum/mayea, d_asenderum/season, d_atastastiveet, d_asa (_custakee, c_name, c_adatese, c_chik, c_nation, c_negion, c_phone, c_mixtegment, p_artikey, p, m (_physe, p.sza, p.continer	opriotity, lo_quantity, lo_extendedprice, opkey, s_name, s_address, s_city, s_nation, onth, d_daynuminweek, d_daynuminmonth, dayinmonthfl, d_holidayfl, d_weekdayfl, imme, p_mfgr, p_category, p_brand, p_color,	
Location									/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw		
Data Size									None		
Provider									PARQUET		
Partition By									lo_orderdate		
ZOrder By											
Last Update Time									2019/12/28 17:52:55		
Enabled  Partition Information Show 10 ¢ entries										Search:	
Path						Siz	0		Action		
								No data av	allable in table		
Showing 0 to 0 of 0 entries	3									Previous Net	axt
Build Information     Latest cache building is fin	ished at 20	019-12-28 1	7:52:55								
Build Cache Trigger	Period Buil	d									

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delet**e删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ **说明** 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

col_name	data_type	comment	-
o_orderkey	bigint	NULL	
o_linenumber	bigint	NULL	
o_custkey	int	NULL	
o_partkey	int	NULL	
o_suppkey	int		
o orderpriotity	string	NULL	
o shippriotity	int	NULL	
o_quantity	bigint	NULL	
o_extendedprice	bigint	NULL	
o_ordtotalprice	bigint	NULL	
.o_discount	bigint	NULL	
o_revenue	bigint	NULL	
o_supplycost	bigint	NULL	
o commitdate	int	NULL	
o shipmode	string	NULL	
suppkev	int	NULL	
name	string	NULL	
_address	string	NULL	
_city	string	NULL	
_nation	string	NULL	
_region	string	NULL	
_phone	int		
date	string	NULL	
davofweek	string	NULL	
month	string	NULL	
year	int	NULL	
_yearmonthnum	int	NULL	
_yearmonth	string	NULL	
_daynuminweek	int	NULL	
_daynuminmonth	int	NULL	
_daynuminyear	int	NULL	
_monthnuminyear	int	NULL	
sellingseason	string	NULL	
lastdavinweekfl	int	NULL	
lastdayinmonthfl	int	NULL	
_holidayfl	int	NULL	
_weekdayfl	int	NULL	
_custkey	int	NULL	
_name	string	NULL	
_address	string	NULL	
_city	string	NULL	
_nation	string	NULL	
phone	string	NULL	
mktsegment	string	NULL	
partkey	int	NULL	
_name	string	NULL	
_mfgr	string	NULL	
_category	string	NULL	
_brand	string	NULL	
	string	NULL	
_type	int	NULL	
	string	NULL	
	501119		
Detailed Table Information		i	
atabase	ssb_10	i	
able	lineorder_flatten	1	
wner	hadoop	1	
reated Time	Sat Dec 28 17:30:02 CST 2019		
ast Access	Thu Jan 01 08:00:00 CST 1970		
ivea l	Spark 2.2 of prior		
/iew Text	SELECT `lineorder`.`lo orderkev`. `lineorder`.`lo l	inenumber`.	'lineorder`.`lo custkev`. `lineorder`.`lo partkev`.
eorder`.`lo suppkey`, `lineord	er'.'lo orderdate', 'lineorder'.'lo orderpriotity',	`lineorder`	.'lo shippriotity', 'lineorder'.'lo quantity', 'lineo
	er`.`lo_ordtotalprice`, `lineorder`.`lo_discount`, `	lineorder`.	`lo_revenue`, `lineorder`.`lo_supplycost`, `lineorder
`.`lo_extendedprice`, `lineord		kev`, `supp	lier`.`s_name`, `supplier`.`s_address`, `supplier`.`s
`.`lo_extendedprice`, `lineord o_tax`, `lineorder`.`lo_commit	date`, `lineorder`.`lo_shipmode`, `supplier`.`s_supp		
<pre>`.`lo_extendedprice`, `lineord o_tax`, `lineorder`.`lo_commit y`, `supplier`.`s_nation`, `su</pre>	date`, `lineorder`.`lo_shipmode`, `supplier`.`s_supp pplier`.`s_region`, `supplier`.`s_phone`, `dates`.`d	_datekey`,	dates . d_date , dates . d_dayofweek , dates . d_r
<pre>:`.`lo_extendedprice`, `lineord .o_tax`, `lineorder`.`lo_commit y`, `supplier`.`s_nation`, `su `, `dates`.`d_year`, `dates`.`</pre>	date`, `lineorder`.`lo_shipmode`, `supplier`.`s_supp pplier`.`s_region`, `supplier`.`s_phone`, `dates`.`d d_yearmonthnum`, `dates`.`d_yearmonth`, `dates`.`d_d	_datekey`, aynuminweek	dates . d_date , dates . d_dayofweek , dates . d_r `, `dates`.`d_daynuminmonth`, `dates`.`d_daynuminyea
<pre>:`.lo_extendedprice`, `lineord .o_tax`, `lineorder`.`lo_commit y`, `supplier`.`s_nation`, `su `, `dates`.`d_year`, `dates`.` dates`.`d_monthnuminyear`, 'da belidere</pre>	date', `lineorder`.lo_shipmode', `supplier`.'s_supp pplier`.'s_region`, `supplier`.'s_phone`, `dates`.'d d_yearmonthnum`, dates`.d_yearmonth', 'dates`.d_ tes`.'d_weeknumiyear`, 'dates`.'d_sellingseason`,	_datekey`, aynuminweek dates`.`d_1	dates . d_date , dates . d_dayotweek , dates . d_r `, `dates`.`d_daynuminmonth`, `dates`.`d_daynuminyea astdayinweekfl`, `dates`.`d_lastdayinmonthfl`, `dates
<pre>c`.lo_extendedprice', 'lineord .o_tax', 'lineorder', lo_commit y', 'supplier'.'s_nation', 'su ', 'dates'.'d_year', 'dates'.' dates'.'d_monthnuminyear', 'da _holidayfl', 'dates'.'d_weekda '</pre>	<pre>date', 'lineorder'.lo_shipmode', 'supplier'.s_supp pplier'.s_region', 'supplier'.s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.d_yearmonth', 'dates'.d_ ds'.d_weeknuminyear', 'dates'.d_sellingseason', yfl', 'custemer'.c_custkey', 'custemer'.c_name',</pre>	_datekey`, aynuminweek dates`.`d_1 customer`.`	<pre>dates . d_date , dates . d_dayotweek , dates . d_r `, 'dates`.'d_daynuminmonth`, 'dates`.'d_daynuminyeas satdayinweekfl`, 'dates`.'d_lastdayinmonthfl`, 'dates c_address`, 'customer`.'c_city', 'customer`.'c_nation ''''''''''''''''''''''''''''''''''''</pre>
(*) Lo_extendedprice', 'lineord o_tax', 'lineorder'.'lo_commit y', 'supplier'.'s_nation', 'su 'dates'.'d_word, 'dates'.' dates'.'d_monthnuminyear', 'da _holidayfl', 'dates'.'d_weedda customer'.'c_region', 'custome '' o brand''.c.''	<pre>date , 'lineorder'.lo_shipmode , 'supplier'.s_supp pplier'.s_region', 'supplier'.s_phone', 'dates'.d d_yearmonthnum', 'dates'.d_yearmonth', 'dates'.d_d tes'.d_weeknuminyear', 'dates'.d_sellingseason', yfl, 'customer'.c_ustkey, 'customer'.c_name', r'.c_phone', 'customer'.c_mktsegment', 'part'.p_p 'aart'' - ture'</pre>	_datekey`, aynuminweek dates`.`d_l customer`.` artkey`, `p per` FPOM	<pre>dates . d_date , dates . d_dayotweek , dates . d_daynuminyea , dates . d_daynuminnth , dates . d_daynuminyea astdayinweekf1 , dates . d_lastdayinmonthf1 , date c_address , customer . c_city , customer . c_natio art . p_name , part . p_mfgr , part . p_category , set 10 . linearder</pre>
'.lo_extendedprice', 'lineord o_tax', 'lineorder'.lo_commit y', 'supplier'.s_nation', 'su ', 'dates'.d_year', 'dates'.' d_holidayfl', 'dates'.d_weekda customer'.c_region', 'custome t'.p_brand', part'.p_color' 'ssh 10' customer'.ech 1	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.'d d_yearmonthnum', 'dates'.'d_yearmonth', 'dates'.'d tes'.'d_weeknuminyear', 'dates'.'d_sellingseason', 'yfl', 'customer'.'c_dates'.'d_sellingseason', 'yfl', 'customer'.'c_mktsegment', 'part'.p_ , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contai A' 'nart'.'</pre>	_datekey`, aynuminweek dates`.`d_1 customer`.` artkey`,`p ner` FROM	<pre>dates . d_date , dates . d_dayotweek , dates . d_daynuminyea , 'dates'. 'd_daynuminonth', 'dates'. 'dates'. astdayinweekfl', 'dates'. 'd_lastdayinmonthfl', 'date c_address', 'customer'. 'c_city', 'customer'. 'c_nation art'. p_name', 'part'. 'p_mfgr', 'part'. 'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'dates', 'ssb_10'.'</pre>
<pre>'.'lo_extendedprice', 'lineord .o_tax', 'lineordr'.'lo_commit y', 'supplier'.'s_nation', 'su ', 'dates'.d_year', 'dates'.' dates'.d_monthnuminyear', 'da _holidayfl', 'dates'.d_weekda customer'.'c_region', 'custome t'.'p_brand', 'part'.p_color' ', 'ssb_10'.customer', 'ssb_1 RE 'lineorder'.'lo_rdredret</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d d_yaarmonthnum', 'dates'.d_yaarmonth', 'dates'.d_ tes'.'d_weeknuminyear', 'dates'.d_sellingseason', ' yfl', 'customer'.c_natkey', 'customer'.c_name', r'.'c_phone', 'customer'.c_mktsegment', 'part'.p_p , 'part'.p_type', 'part'.'p_size', 'part'.'p_contai e' 'dates'.'d datekey' AND 'lineorder'.'lo custere'</pre>	_datekey`, aynuminweek dates`.`d_1 customer`.` artkey`,`p ner` FROM `customer`	<pre>dates . d_date , dates . d_dayotweek , dates . d_daynuminyea , 'dates'.'d_daynuminonth', 'dates'.'d_daynuminyea astdayinweekfl', 'dates'.'d_lastdayinmonthfl', 'dates _address', 'customer'.'c_city', 'customer'.'c_nation art'.'p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'c .'c custkey' AND 'lineorder'.'lo sunnkey' = 'sunnlier'.</pre>
<pre>lo_extendedprice', 'lineord .o_tax', lineordor'.lo_commit y', 'supplier'.s_nation', 'su ', 'dates'.d_year', 'dates'. dates'.d_monthnuminyear', 'da '_holidayfl', 'dates'.d_weekda customer'.c_region', 'custome t'.p_brand', 'part'.p_color' ', 'ssb_10'.customer', 'sbp_11 RE 'lineorder'.lo_orderdate' suppkey'AND 'lineorder'.</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.'d_ d_yearmonthnum', 'dates'.'d_yearmonth', 'dates'.'d_ tes'.'d_weeknuminyear', 'dates'.'d_sellingseason', 'yfl', 'customer'.c_ustkey, 'customer'.c_name', 'r'.'c_phone', 'customer'.'c_mktsegment', 'part'.'p_p , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contai 0'.part' e'dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_partkey'  </pre>	_datekey`, aynuminweek dates`.`d_l customer`.` artkey`, `p ner` FROM `customer`	dates . d_date , dates . d_dayotweek , dates . d_daynuminyea , 'dates .'.d_daynuminonth', 'dates .'.daynuminyea astdayinweekf1', 'dates'.'d_lastdayinmonthf1', 'date c_address', 'customer'.'c_city', 'customer'.c_natio art'.`p_name', 'part'.`p_mfgr', 'part'.`p_category', ssb_10'.`lineorder', 'ssb_10'.`supplier', 'ssb_10'.' .`c_custkey` AND 'lineorder`.`lo_suppkey` = `supplier
<pre>`.'.lo_extendedprice', 'lineord .o_tax', 'lineorder'.lo_commit y', 'supplier'.'s_nation', 'su ', 'dates'.d_worthnuminyear', 'da _holidayfl', 'dates'.d_weekda customer'.c_region', 'custome t'.'p_brand', 'part'.'p_color' ', 'ssb_10'.customer', 'ssb_1 RE 'lineorder'.'lo_orderdate' _suppkey' AND 'lineorder'.'lo_ able Properties</pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.d_yearmonth', 'dates'.d_ dts'.d_weeknuminyear', 'dates'.d_sellingseason', ' yfl', 'customer'.c_custkey', 'customer'.c_name', ' r'.c_phone', 'customer'.c_mktsegment', 'part'.p_p , 'part'.'p_type', 'part'.'p_size', 'part'.p_contai 0'.part' = 'dates'.d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.p_partkey'   [transient_lastDdlTime=15776759]</pre>	_datekey`, aynuminweek dates`.`d_l customer`.` artkey`, `p ner` FROM `customer` 	<pre>dates . d_date , dates . d_dayotweek , dates . d_daynuminyea , 'dates'.'d_daynuminonth', 'dates'.'d_lastdayinwnekf1', 'date c_address', 'customer'.'c_city', 'customer'.'c_nation art'.p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'c .'c_custkey` AND 'lineorder'.'lo_suppkey` = 'supplier  </pre>
<pre>'.'lo_extendedprice', 'lineord .o_tax', 'lineordr'.'lo_commit y', 'supplier'.'s_nation', 'su ', 'dates'.d_year', 'dates'.' dates'.d_monthnuminyear', 'da _holidayfl', 'dates'.d_weekda customer'.c_region', 'custome t'.p_brand', 'part'.p_color' ', 'ssb_10'.customer', 'ssb_1 RE 'lineorder'.'lo_orderdate' _suppkey' AND 'lineorder'.'lo_ able Properties   erde Library  </pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d d_yaarmonthnum', 'dates'.d_yaarmonth', 'dates'.d_ tes'.'d_weeknuminyear', 'dates'.d_sellingseason', ' yf'.'customer'.c_custkey', 'customer'.c_name', r'.'c_phone', 'customer'.c_mktsegment', 'part'.'p_p , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contai 0'.'part' = 'dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_partkey'   [transient_lastDdlTime=1577677599] org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe</pre>	_datekey`, aynuminweek dates`.`d_l customer`.` artkey`, `p ner` FROM `customer`   	<pre>dates . d_date , dates . d_dayotweek , datesd_daynuminyea , 'dates '.d_daynuminonth', 'dates '.daynuminyea astdayinweekf1', 'dates '.d_lastdayinmonthf1', 'date c_address', 'customer'.c_city', 'customer'.c_natio art '.p_name', 'part '.p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'d .'c_custkey` AND 'lineorder'.'lo_suppkey` = `supplier  </pre>
<pre>'.'lo_extendedprice', 'lineord .o_tax', 'lineordr'.lo_commit y', 'supplier'.'s_nation', 'su ', 'dates'.'d_year', 'dates'.' dates'.'d_monthnuminyear', 'da _holidayfl', 'dates'.'d_weekda customer'.'c_region', 'custome t'.'p_brand', 'part'.'p_color' ', 'ssb_10'.'customer', 'ssb_1 RE 'lineorder'.'lo_orderdate' _suppkey' AND 'lineorder'.'lo_ able Properties erde Library   nputFormat  </pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.'d_ d_yearmonthnum', 'dates'.'d_yearmonth', 'dates'.'d_ tes'.'d_weeknuminyear', 'dates'.'d_sellingseason', ' yf'l, 'customer'.c_ustkey', 'customer'.c_name', r'.'c_phone', 'customer'.'c_mktsegment', 'part'.'p_p , 'part'.'p_type', 'part'.'p_size', 'part'.'p_contai 0'.part e' dates'.'d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.p_partkey'   [transient_lastDdlTime=1577677599] org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe org.apache.hadoop.mapred.TextInputFormat</pre>	_datekey`, aynuminweek dates`.`d_l customer`.` artkey`, `p ner` FROM `customer`     	<pre>dates . d_date , dates . d_dayotweek , datesd_dayonwinyea astdayinweekf1', 'dates'.'d_lastdayinmonthf1', 'dates c_address', 'customer'.'c_city', 'customer'.c_natio art'.'p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.' .'c_custkey' AND 'lineorder'.'lo_suppkey' = 'supplier    </pre>
<pre>'.'lo_extendedprice', 'lineord .o_tax', 'lineorder'.lo_commit y', 'supplier'.'s_nation', 'su ', 'dates'.d_vear', 'dates'.' dates'.d_monthnuminyear', 'da _holidayfl', 'dates'.d_weekda customer'.c_region', 'custome t'.'p_brand', 'part'.'p_color' ', 'ssb_10'.customer', 'ssb_1 RE 'lineorder'.'lo_orderdate' _suppkey' AND 'lineorder'.'lo_ able Properties erde Library   nputFormat   utputFormat    </pre>	<pre>date', 'lineorder'.'lo_shipmode', 'supplier'.'s_supp pplier'.'s_region', 'supplier'.'s_phone', 'dates'.d_ d_yearmonthnum', 'dates'.d_yearmonth', 'dates'.d_ tes'.d_weeknuminyear', 'dates'.d_sellingseason', ' yfl', 'customer'.'c_custkey', 'customer'.c_name', ', 'part'.'p_type', 'part'.'p_size', 'part'.'p_contai 0'.part' = 'dates'.d_datekey' AND 'lineorder'.'lo_custkey' = partkey' = 'part'.'p_partkey'   [transier_lastDdTime=157767759] org.apache.hadoop.mapred.TextIngutFormat org.apache.hadoop.mapred.TextIngutFormat</pre>	_datekey`, aynuminweek dates`.`d_l artkey`, `p ner` FROM `customer`     tFormat	<pre>dates . d_date , dates . d_dayotweek , dates . d_daynuminyea , 'dates'.'d_daynuminonth', 'dates'.'d_lastdayinwnoehfl', 'dates c_address', 'customer'.c_city', 'customer'.c_nation art'.p_name', 'part'.'p_mfgr', 'part'.'p_category', ssb_10'.'lineorder', 'ssb_10'.'supplier', 'ssb_10'.'d .'c_custkey' AND 'lineorder'.'lo_suppkey' = 'supplier      </pre>

0: jdbc:hive2://localhost:10001/ssb\_10>

#### 基于lineorder\_flatten视图简单查询的执行计划如下:

_
<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.enforce.bucketing does not exist 19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist</pre>
plan
<pre>+</pre>
<pre>+- *(1) SoctmetgeJoin [Lo_partkey#313], [b_partkey#359], inner :- *(8) Sort [lo_partkey#313], [c_partkey#313, 200) : +- Exchange hashpartitioning(lo_partkey#312], [c_custkey#351], Inner : :- *(4) Sort [lo_custkey#312], [c_custkey#312], [a_custkey#351], Inner : :- *(4) Sort [lo_custkey#312], [c_custkey#312], [a_distantian [lo_sustkey#312], [d_datekey#334], Inner, BuildRight : : +- *(3) BroadcastHashDoin [lo_supkey#314], [s_suppkey#327], Inner, BuildRight : : :- *(3) BroadcastHashDoin [lo_supkey#314], &amp;&amp; isnotnull(lo_custkey#312]) &amp;&amp; isnotnull(lo_part key#313))</pre>
: : : +- Scan hive ssb_t0.lineorder [10_orderke#310L, lo_linenumber#311L, lo_custkey#312, lo_partkey#312, lo_suppkey#313, lo_ord erdate#315, lo_orderpriotity#315, lo_supriotity#317, lo_quantity#318L, lo_extendedprice#319L, lo_ordtotalprice#220L, lo_discount#321L, lo_revenue#32 2L, lo_supplycost#323L, lo_tax#324L, lo_commitdate#325, lo_shipmode#326], HiveTableRelation `ssb_10`.`lineorder`, org.apache.hadoop.hive.serde2.lazy.L azySimpleSerDe, [lo_orderkey#316L, lo_linenumber#311L, lo_custkey#312, lo_partkey#313, lo_suppkey#314, lo_orderdet#315, lo_orderpriotity#316, lo_ship priotity#317, lo_quantity#318L, lo_extendedprice#319L, lo_ordtotalprice#320L, lo_discount#321L, lo_revenue#322L, lo_supplycost#323L, lo_tax#324L, lo_c ommitdate#325, lo_shipmode#326]
<pre>: : + → BroadcastExchange HashedRelationBroadcastMode(list(cast(input[0, int, false] as bigint))) : : : + → [1] Filter isonthull(s_suppkey#327) : : + → Scan hive ssb_10.supplier [s_suppkey#327] 33], HiveTableRelation 'ssb 10', 'supplier', ora.aache.hadoop.hive.serde2.lazv.LazvismbleSerDe.[s_supkey#327, s_name#328, s_ddress#329, s_nate#328, s_ddress#329, s_nate#328, s_ddress#329, s_nate#328, s_ddress#329, s_nate#328, s_ddress#329]</pre>
, s_nation#331, s_region#332, s_phone#333] : : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint)))
: : +- *(2) Filter isnotnull(d_atekey#334) : : Scan hive ssb_10.dates [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearm onth#340, d_daynuminweek#341, d_daynuminmonth#342, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweek fl#347, d_lastdayinmonthfl#348, d_holidayfl#349, d_weekdayfl#3501, HiveTableRelation `sb_10`.'dates', org.apache.hadoop.hive.serde2.lazyl.azySimpleSe rDe, [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_yearmonthnum#339, d_yearmonth#340, d_daynuminweek#341, d_daynuminwonth#32 , d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweekfl#347, d_lastdayinmonthfl#348, d_holidayfl#349, d_weekdayfl#350]
<pre>: +- Exchange hashpartitioning(c_custkey#351, 200) : +- *(5) Filter ((isnotnull(c_region#366) &amp;&amp; (c_region#366 = ASIA)) &amp;&amp; isnotnull(c_custkey#351)) : +- Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk tsegment#358], HiveTableRelation 'ssb_10'.'customer [, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [c_custkey#351, c_name#352, c_address#353, c _city#354, c_nation#355, c_region#356, c_cphone#357, c_mktsegment#358] +- *(10) Sort [p_partkey#359 ASC NULLS FIRST], false, 0 +- Exchange hashpartitioning(p_partkey#359, 200) +- *(9) Filter isnotnul(p_partkey#359)</pre>
+- Scan hive ssb_10.part [p_partkey#359, p_name#360, p_mfgr#361, p_category#362, p_brand#363, p_color#364, p_type#365, p_size#366, p_conta iner#367], HiveTableRelation `ssb_10'.`part`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [p_partkey#359, p_name#360, p_mfgr#361, p_category#3 62, p_brand#363, p_color#364, p_type#365, p_size#366, p_container#367]   +
1 row selected (0.435 seconds) 0: jdbc:hive2://localhost:10001/ssb_10>
在为line order_flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计
划,10.化后的执行计划如下:
(0: jdbc:hive2://localhost:18001/ssb_18> explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:17:47 INFO [main] HiveConf: Found configuration file file:/etc/ecm/spark-conf-2.4.3-hadoop2.8-1.4.2/hive-site.xml 19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.enforce.bucketing does not exist 19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist
plan
== Physical Plan == CollectLimit 10
<pre>+- *(1) Project [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderdate#133, lo_orderpriotity#134, lo_ship priotity#136, lo_quantity#136L, lo_extendedprice#137L, lo_ordtotalprice#138L, lo_discount#139L, lo_revenue#140L, lo_supplycost#141L, lo_tax#142L, lo_c ommitdate#143, lo_shipmode#144, s_suppkey#145, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#150, s_phone#151, 34 more fields] +- *(1) Filter (isonthull(c_region#174) &amp;&amp; (c_region#174 = ASIA)) +- *(1) Filter(sam parquet [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderpriotity#134, lo_shippriotity# 25 lo_europity#142(lo_orderderige#1271) lo_orderigerigerigerigerigerigerigerigerigerig</pre>

可以看到,优化后的执行计划省去了lineorder\_flatten视图的所有计算逻辑,直接访问HDFS中Cache的数据。

## 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性,而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据,用户需要根据查询对于数据一致性的需求,触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

# 11.4. 工具集

# 11.4.1. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

### 前提条件

- 本地安装了Java JDK 8。
- 已创建集群,详情请参见创建集群。

# 使用Jindo Distcp

1. 通过SSH方式连接集群。

```
详情请参见登录集群。
```

2. 执行以下命令, 获取帮助信息。

jindo distcp --help

#### 返回信息如下。

help - Print help text
src=VALUE - Directory to copy files from
dest=VALUE - Directory to copy files to
parallelism=VALUE - Copy task parallelism
outputManifest=VALUE - The name of the manifest file
previousManifest=VALUE - The path to an existing manifest file
requirePreviousManifest=VALUE - Require that a previous manifest is present if specified
copyFromManifest - Copy from a manifest instead of listing a directory
srcPrefixesFile=VALUE - File containing a list of source URI prefixes
srcPattern=VALUE - Include only source files matching this pattern
deleteOnSuccess - Delete input files after a successful copy
outputCodec=VALUE - Compression codec for output files
groupBy=VALUE - Pattern to group input files by
targetSize=VALUE - Target size for output files
enableBalancePlan - Enable plan copy task to make balance
enableDynamicPlan - Enable plan copy task dynamically
enableTransaction - Enable transation on Job explicitly
diff - show the difference between src and dest filelist
ossKey=VALUE - Specify your oss key if needed
ossSecret=VALUE - Specify your oss secret if needed
ossEndPoint=VALUE - Specify your oss endPoint if needed
policy=VALUE - Specify your oss storage policy
cleanUpPending - clean up the incomplete upload when distcp job finish
queue=VALUE - Specify yarn queuename if needed
bandwidth=VALUE - Specify bandwidth per map/reduce in MB if needed
s3Key=VALUE - Specify your s3 key
s3Secret=VALUE - Specify your s3 Sercet
s3EndPoint=VALUE - Specify your s3 EndPoint

### --src和--dest

--src 表示指定源文件的路径, --dest 表示目标文件的路径。

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录, Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

## --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

#### --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

例如,如果您需要复制/data/incoming/hourly\_table/2017-02-01/03下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Found 6 items							
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09							
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,复制以log结尾的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPattern .\*\.lo
g --parallelism 20

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
Found 2 items

-rw-rw-rw- 1 4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

-rw-rw-rw- 1 4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

#### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --deleteOnSuccess -parallelism 20

#### --outputCodec

--outputCodec 可以在线高效地存储数据和压缩文件。

jindo distop --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下:

Found 6 item	S		
-rw-rw-rw-	1	938 2020-04-17 20:58 oss:// <youre< td=""><td>ucketName&gt;/hourly_table/2017-02-01/03/000151.sst.gz</td></youre<>	ucketName>/hourly_table/2017-02-01/03/000151.sst.gz
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <youre< td=""><td>ucketName&gt;/hourly_table/2017-02-01/03/1.log.gz</td></youre<>	ucketName>/hourly_table/2017-02-01/03/1.log.gz
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <youre< td=""><td>ucketName&gt;/hourly_table/2017-02-01/03/2.log.gz</td></youre<>	ucketName>/hourly_table/2017-02-01/03/2.log.gz
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <youre< td=""><td>ucketName&gt;/hourly_table/2017-02-01/03/OPTIONS-000109.g</td></youre<>	ucketName>/hourly_table/2017-02-01/03/OPTIONS-000109.g
Z			
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <youre< td=""><td>ucketName&gt;/hourly_table/2017-02-01/03/emp01.txt.gz</td></youre<>	ucketName>/hourly_table/2017-02-01/03/emp01.txt.gz
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <youre< td=""><td>ucketName&gt;/hourly table/2017-02-01/03/emp06.txt.gz</td></youre<>	ucketName>/hourly table/2017-02-01/03/emp06.txt.gz

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则Jindo Dist Cp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

⑦ 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。

## --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

# --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录, 查看运行期间复制的文件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --outputManifest=ma nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly_table/manifest-2020-04-17.gz --parallel ism 20
```

#### 执行以下命令,查看运行期间复制的文件。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-18.gz > current.lst diff before.lst current.lst

#### 返回信息如下。

3a4.5

> {"path":"oss://<yourBucketName>/hourly table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir" :"oss://<yourBucketName>/hourly table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir" :"oss://<yourBucketName>/hourly\_table","size":4891}

# --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

#### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

#### 示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly table

#### 返回信息如下。

```
Found 4 items
drwxrwxrwx -
                       0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01
                       0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
drwxrwxrwx -
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

jindo distcp --src /data/incoming/hourly table --dest oss://<yourBucketName>/hourly table --srcPrefixesFile f ile:///opt/folders.txt --parallelism 20

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly table/2017-02-01 hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly table/2017-02-02

#### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Found 8 items							
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09							
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

#### --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为DistCp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo DistCp的作业分配计划,以达到更好的DistCp性能。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --enableBalancePlan
--parallelism 20

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

## --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定 --enableDynamicPlan 来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

# --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的Dist Cp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --dest oss://<yourB ucketName>/hourly\_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis m 20

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

#### --queue

您可以使用--queue来指定本次DistCp任务所在Yarn队列的名称。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<your\_bucket>/hourly\_table --queue yarnqueue

#### --bandwidth

您可以使用--bandwidth来指定本次DistCp任务所用的带宽(以MB为单位),避免占用过大带宽。

#### 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

#### 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

● 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive --parallelism 20

• 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

#### 清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

#### 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

# 使用s3作为数据源

您可以在命令中使用--s3Key、--s3Secret、--s3EndPoint选项来指定连接s3的相关信息。

#### 代码示例如下。

jindo distcp jindo-distcp-2.7.3.jar --src s3a://yourbucket/ --dest oss://<your\_bucket>/hourly\_table --s3Key y ourkey --s3Secret yoursecret --s3EndPoint s3-us-west-1.amazonaws.com

您可以配置s3Key、s3Secret、s3EndPoint在Hadoop的*core-site.xm*文件里 ,避免每次使用时填写Accesskey。

```
<configuration>
<property>
<name>fs.s3a.access.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3a.secret.key</name>
<value>xxx</value>
</property>
<property>
<name>fs.s3.endpoint</name>
<value>s3-us-west-1.amazonaws.com</value>
</property>
</configuration>
```

#### 此时代码示例如下。

jindo distcp /tmp/jindo-distcp-2.7.3.jar --src s3://smartdata1/ --dest s3://smartdata1/tmp --s3EndPoint s3-u s-west-1.amazonaws.com

## 查看Distcp Counters

```
执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。
```

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG REDUCE=0
```

⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的 大小可能不相等。

# 11.4.2. FUSE使用说明

本文介绍如何通过FUSE客户端访问JindoFS。FUSE支持Block和JFS Scheme的Cache两种模式。

#### 前提条件

已创建集群,详情请参见创建集群。

# 背景信息

FUSE是Linux系统内核提供的一种挂载文件系统的方式。通过JindoFS的FUSE客户端,将JindoFS集群上的文件映射到本地磁盘,您可以像访问本地磁盘一样访问JindoFS集群上的数据,无需再使用 hadoop fs -ls jfs://<namespace>/ 方式访问数据。

#### 挂载

- ⑦ 说明 依次在每个节点上执行挂载操作。
- 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。
- 2. 执行如下命令,新建目录。

mkdir /mnt/jfs

3. 执行如下命令, 挂载目录。

jindofs-fuse /mnt/jfs

/mnt/jfs作为FUSE的挂载目录。

#### 读写文件

1. 列出/mnt/jfs/下的所有目录。

ls /mnt/jfs/

#### 返回用户在服务端配置的所有命名空间列表。

test testcache

#### 2. 列出命名空间test下面的文件列表。

ls /mnt/jfs/test/

#### 3. 创建目录。

mkdir /mnt/jfs/test/dir1
ls /mnt/jfs/test/

#### 4. 写入文件。

echo "hello world" > /tmp/hello.txt
cp /tmp/hello.txt /mnt/jfs/test/dirl/

#### 5. 读取文件。

cat /mnt/jfs/test/dir1/hello.txt

#### 返回如下信息。

hello world

### 如果您想使用Python方式写入和读取文件,请参见如下示例:

#### 1. 使用Python写write.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'w',encoding = 'utf-8') as f:
    f.write("my first file\n")
    f.write("This file\n\n")
    f.write("contains three lines\n")
```

#### 2. 使用Python读文件。创建脚本 read.py文件,包含如下内容。

```
#!/usr/bin/env python36
with open("/mnt/jfs/test/test.txt",'r',encoding = 'utf-8') as f:
    lines = f.readlines()
    [print(x, end = '') for x in lines]
```

#### 读取写入test.txt文件的内容。

[hadoop@emr-header-1 ~]\$ ./read.py

#### 返回如下信息。

my first file This file

# 卸载

? 说明 依次在每个节点上执行卸载操作。

#### 1. 使用SSH方式登录到集群主节点,详情请参见登录集群。

# 2. 执行如下命令, 卸载FUSE。

umount jindofs-fuse

如果出现 target is busy 错误,请切换到其它目录,停止所有正在读写FUSE文件的程序,再执行卸载操作。

# 12.SmartData 2.6.0-2.7.2 12.1. SmartData 2.6.0-2.7.2版本简介

Smart Dat a的2.6.0-2.7.2版本,包含多个重大特性的发布以及大幅的性能优化。例如,Namespace服务后端存储支持 Tablest ore(OTS)以及Raft、Namespace服务支持HA、读写性能优化、块存储模式和缓存模式使用方式优化等。

# 元数据服务后端存储方案升级

在原有RocksDB方案的基础上,新版本推出了Tablestore和Raft的后端存储方案,实现元数据上云。

针对使用Cache模式且对于元数据存储以及HA没有高要求的场景,默认的RocksDB是一种简单、实用而且高效的方案。 Tablest ore和Raft的方案,实现了元数据服务的高可用,可以通过多个Namespace服务提供HA方案。

各方案详情请参见:

- 使用Tablestore作为存储后端
- 使用Raft-RocksDB-Tablestore作为存储后端
- 使用RocksDB作为元数据后端

# 使用模式优化

支持块存储模式和缓存模式两种使用模式:

- 块存储模式(Block):详情请参见JindoFS块存储模式使用说明。
- 缓存模式(Cache):支持多种使用方式。例如,既支持与Block模式一致的使用方式,也支持原有OSS文件系统的使用方式, 以满足用户不同的需要,详情请参见jindoFS缓存模式使用说明。

# 支持权限

Block模式支持Unix权限和Ranger权限两种文件系统权限功能:

- Unix权限:可以使用文件的777权限。
- Ranger权限:可以使用Ranger路径通配符等高级配置。

权限功能详细请参见JindoFS权限功能。

# 12.2. JindoFS Block模式

# 12.2.1. JindoFS块存储模式使用说明

块存储模式(Block)提供了最为高效的数据读写能力和元数据访问能力。数据以Block形式存储在后端存储OSS上,本地提供缓存加速,元数据则由本地Namespace服务维护,提供高效的元数据访问性能。本文主要介绍JindoFS的块存储模式及其使用方式。

# 背景信息

JindoFS块存储模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

### 配置使用方式

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。

- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入bigboot服务配置页面。
  - i. 单击**配置**页签。
  - ii. 单击bigboot。

< 返回             SmartData →   ●正常									
状态	部署拓扑	配置	鼠 配置修改历史						
配置达	tia				服务配置				
配置搜	嫊				全部 「 smartdata-site 」 client 「 storage 」 bigboot				
请输	λ			Q					

#### 3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	描述	示例	
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 推荐配置到OSS Bucket下的某一个具体目录,该命 名空间即会将Block模式的数据块存 放在该目录下。	
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block	
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX	
jfs.namespaces.test.oss.access.secre t	表示存储后端OSS的AccessKey Secret。	⑦ 说明 考虑到性能和稳定性, 推荐使用同账户、同Region下的 OSS Bucket作为存储后端,此 时, E-MapReduce集群能够免密访 问OSS,无需配置AccessKey ID和 AccessKey Secret。	

# ⅲ. 单击确定。

- 4. 单击右上角的保存。
- 5. 选择右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_of\_file> 的形式访问JindoFS上的文件。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

#### 可以在**服务配置**区域的storage页签,修改以下参数。

storage.watermark.low.ratio	0.2	0					
storage.watermark.high.ratio	0.4	0					
参数	描述						
storage.watermark.high.ratio	表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据 用的磁盘空间到达上水位即会触发清理。默认值:0.4。	目录占					
storage.watermark.low.ratio	表示使用量的下水位比例,触发清理后会自动清理冷数据, JindoFS数据目录占用空间清理到下水位。默认值:0.2。	将					

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

- 2. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 选择右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数,单击确定。
  - iii. 在确认对话框中,单击确定。

# 12.2.2. 使用Tablestore作为存储后端

JindoFS元数据服务支持不同的存储后端,本文介绍使用Tablestore(OTS)作为元数据后端时需要进行的配置。

#### 前提条件

- 已创建EMR集群。
   详情请参见创建集群。
- 已创建Tablestore实例,推荐使用高性能实例。

详情请参见创建实例。

? 说明 需要开启事务功能。

# 背景信息

JindoFS在新版本中,支持使用Tablestore作为JindoFS元数据服务(Namespace Service)的存储。一个EMR JindoFS集群可以绑 定一个Tablestore实例(Instance)作为JindoFS元数据服务的存储介质,元数据服务会自动为每个Namespace创建独立的 Tablestore表进行管理和存储元数据信息。 元数据服务(双机Tablestore和HA)架构图如下所示。



# 配置Tablestore

使用Tablestore功能,需要把创建的Tablestore实例和JindoFS的Namespace服务进行绑定,详细步骤如下:

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入bigboot服务配置页面。
  - i. 单击**配置**页签。
  - ii. 单击bigboot。

< 返回      ●● SmartData ~  ● 正常							
状态 部署拓扑 配置 配置修改历史	配置修改历史						
配置过滤      服务配置							
配置搜索               全部  sma	rtdata-site   client   storage   <mark>bigboo</mark> t						

3. 配置以下参数。

例如,在华东1(杭州)地域下,创建了emr-jfs的Tablestore实例,EMR集群使用VPC网络,访问Tablestore的AccessKey ID 为kkkkkk,Access Secret为XXXXXX。

参数	参数说明	是否必选	示例
----	------	------	----

参数	参数说明	是否必选	示例
namespace.backend.typ e	设置namespace后端存储类 型,支持: o rocksdb o ots o raft 默认为rocksdb。	是	ots
namespace.ots.instance	Tablestore实例名称。	是	emr-jfs
namespace.ots.accessKe y	Tablestore实例的AccessKey ID。	否	kkkkk
namespace.ots.accessSe cret	Tablestore实例的AccessKey Secret。	否	XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地 址,普通EMR集群,推荐使用 VPC地址。	是	http://emr-jfs.cn-hangzhou .vpc.tablestore.aliyuncs.co m

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。

# 配置Tablestore(高可用方案)

针对EMR的高可用集群,可以通过配置开启Namespace高可用模式。

🔤 高可用 🛛 高可用: 👩 🧲

Namespace高可用模式采用Active和Standby互备方式,支持自动故障转移,当Active Namespace出现异常或者异常中止时, 客户端可以请求自动切换到新的Active节点。



- 1. 进入SmartData的bigboot服务配置,配置以下参数。
  - i. 修改jfs.namespace.server.rpc-address值为emr-header-1:8101,emr-header-2:8101。
  - ii. 单击右上角的自定义配置,添加namespace.backend.ots.ha为true。
  - iii. 单击确定。
  - iv. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
- 2. 单击右上角的操作 > 重启Jindo Namespace Service。
- 3. 单击右上角的操作 > 重启Jindo Storage Service。

# 12.2.3. 使用RocksDB作为元数据后端

JindoFS元数据服务支持不同的存储后端,默认配置RocksDB为元数据存储后端。本文介绍使用RocksDB作为元数据后端时需要进 行的相关配置。

# 背景信息

RocksDB作为元数据后端时不支持高可用。如果需要高可用,推荐配置Tablestore(OTS)或者Raft作为元数据后端,详情请参 见使用Tablestore作为存储后端和使用Raft-RocksDB-Tablestore作为存储后端。

单机RocksDB作为元数据服务的架构图如下所示。



# 配置RocksDB

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入bigboot服务配置页面。
  - i. 单击配置页签。

ii. 单击bigboot。

< 返回           SmartData ~ ● 正常								
状态	部署拓扑	配置	配置修改历史					
配置过	罅				服务配置			
配置搜	溸				全部 「 smartdata-site 」 client 「 storage 」 <mark>bigboot</mark>			
请输	λ			Q				

- 3. 设置namespace.backend.type为rocksdb。
- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 5. 单击右上角的操作 > 重启 Jindo Namespace Service。

# 12.2.4. 使用Raft-RocksDB-Tablestore作为存储后端

JindoFS在EMR-3.27.0及之后版本中支持使用Raft-RocksDB-OTS作为Jindo元数据服务(Namespace Service)的存储。1个EMR JindoFS集群创建3个Master节点组成1个Raft实例,实例的每个Peer节点使用本地RocksDB存储元数据信息。

#### 前提条件

• 创建Tablestore实例,推荐使用高性能实例,详情请参见创建实例。

```
? 说明 需要开启事务功能。
```

• 创建3 Master的EMR集群,详情请参见创建集群。

🖾 高可用	高可用: 🕢		
	部署方式: 👩	O 2 Master	
	0	3 Master	查看服务部署
? 说明	如果没有部署	方式,请 <mark>提交</mark>	工单处理。

# 背景信息

RocksDB通过Raft协议实现3个节点之间的复制。集群可以绑定1个Tablestore(OTS)实例,作为Jindo的元数据服务的额外存储 介质,本地的元数据信息会实时异步地同步到用户的Tablestore实例上。



元数据服务-多机Raft-RocksDB-Tablestore+HA如下图所示。

# 配置本地raft后端

- 1. 新建EMR集群后,暂停SmartData所有服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏, 单击集群服务 > Smart Data。
  - vi. 单击右上角的操作 > 停止 All Components。
- 2. 根据使用需求,添加需要的namespace。
- 3. 进入SmartData服务的bigboot页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击bigboot页签。
- 4. 在SmartData服务的bigboot页签,设置以下参数。

参数	描述	示例
namespace.backend.type	设置namespace后端存储类型,支持: 。 rocksdb 。 ots 。 raft 默认为rocksdb。	raft

参数	描述	示例
namespace.backend.raft.initial- conf	部署raft实例的3个Master地址(固定 值)。	emr-header-1:8103:0,emr-header- 2:8103:0,emr-header-3:8103:0
jfs.namespace.server.rpc-address	Client端访问raft实例的3个Master地址 (固定值)	emr-header-1:8101,emr-header- 2:8101,emr-header-3:8101

⑦ 说明 如果不需要使用OTS远端存储,直接执行步骤6和步骤7;如果需要使用OTS远端存储,请执行步骤5~步骤
 7。

#### 5. (可选)配置远端OTS异步存储。

#### 在SmartData服务的bigboot页签,设置以下参数。

参数	参数说明	示例
namespace.ots.instance	Tablestore实例名称。	emr-jfs
namespace.ots.accessKey	Tablestore实例的AccessKey ID。	kkkkk
namespace.ots.accessSecret Tablestore实例的AccessKey Secret。		XXXXXX
namespace.ots.endpoint	Tablestore实例的Endpoint地址,通常 EMR集群,推荐使用VPC地址。	http://emr-jfs.cn-hangzhou.vpc.tables tore.aliyuncs.com
namespace.backend.raft.async.ot	是否开启OTS异步上传,包括: • true • false 当设置为true时,需要在SmartData服务 完成初始化前,开启OTS异步上传功能。	true
s.enabled	⑦ 说明 如果SmartData服务已 完成初始化,则不能再开启该功能。 因为OTS的数据已经落后于本地 RocksDB的数据。	

#### 6. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。

ⅲ. 单击确定。

7. 单击右上角的操作 > 启动All Components。

### 从Tablestore恢复元数据信息

如果您在原始集群开启了远端Tablestore异步存储,则Tablestore上会有1份完整的JindoFS元数据的副本。您可以在停止或释放 原始集群后,在新创建的集群上恢复原先的元数据,从而继续访问之前保存的文件。

- 1. (可选)准备工作。
  - i. (可选)统计原始集群的元数据信息(文件和文件夹数量)。

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
(文件夹个数) (文件个数)
```

ii. 停止原始集群的作业,等待30~120秒左右,等待原始集群的数据已经完全同步到Tablestore。执行以下命令查看状态。如果LEADER节点显示 \_\_synced=1 ,则表示Tablestore为最新数据,同步完成。

jindo jfs -metaStatus -detail

[RaftPeerImpl]
peer id:
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: N0 stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
<pre>vote_timer: timeout(5000ms) STOPPED</pre>
<pre>stepdown_timer: timeout(5000ms) SCHEDULING(in 2335ms)</pre>
snapshot_timer: timeout(3600000ms) SCHEDULING(in 150305ms)
storage: [1, 624625]
disk_index: 624625
known_applied_index: 624625
last_log_id: (index=624625,term=2)
first_index_pinned: 624625
state_machine: Idle
last_committed_index: 624625
last_snapshot_index: 0
last_snapshot_term: 0
snapshot_status: IDLE
replicator_25769803789@ext_index=624626 flying_append_entries_size=0 idle hc=2301 ac=624261 ic=0
replicator_329853488332 next_index=624626 flying_append_entries_size=0 idle hc=2301 ac=623564 ic=0
OtsUploader: _lastStopIndex=624624, _synced=1
Flacker DR Down Country of arch (ECTAHIA)]

iii. 停止或释放原始集群,确保没有其它集群正在访问当前的Tablestore实例。

#### 2. 创建新集群。

新建与Tablestore实例相同Region的EMR集群,暂停SmartData所有服务。详情请参见配置本地raft后端中的步骤1。

#### 3. 初始化配置。

在SmartData服务的bigboot页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	false
namespace.backend.raft.recovery. mode 是否开启从OTS恢复元数据,包括: o true o false		true

#### 4. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 5. 单击右上角的操作 > 启动 All Components。
- 6. 新集群的SmartData服务启动后,自动从OTS恢复元数据到本地Raft-RocksDB上,可以通过以下命令查看恢复进度。

jindo jfs -metaStatus -detail

如图所示, LEADER节点的state为FINISH表示恢复完成。

```
[RaftPeerImpl]
peer_id: _____:8103:0
state: LEADER
readonly: 0
term: 2
conf_index: 1
peers:
changing_conf: NO stage: STAGE_NONE
election_timer: timeout(5000ms) STOPPED
vote_timer: timeout(5000ms) STOPPED
stepdown_timer: timeout(5000ms) SCHEDULING(in 3382ms)
snapshot_timer: timeout(600000ms) SCHEDULING(in 474855ms)
storage: [1, 153]
disk_index: 153
known_applied_index: 153
last_log_id: (index=153,term=2)
first_index_pinned: 1
state_machine: Idle
last_committed_index: 153
last_snapshot_index: 1
last_snapshot_term: 2
snapshot_status: IDLE
replicator_1116691496965@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0 replicator_3311419785217@1 : next_index=154 flying_append_entries_size=0 idle hc=262 ac=154 ic=0
 [Recovery From OTS Status]
 state: FINISH
total rows: 1484409
table `jfs_block_test` 2 rows.
table `jfs_namespace_cache_ns` 1 rows.
table `jfs_namespace_test` 1484406 rows.
```

7. (可选)执行以下操作,可以比较一下文件数量与原始集群是否一致。

此时的集群为恢复模式,也是只读模式。

#### # 对比文件数量一致

```
[hadoop@emr-header-1 ~]$ hadoop fs -count jfs://test/
1596 1482809 25 jfs://test/
# 文件可正常读取(cat、get命令)
[hadoop@emr-header-1 ~]$ hadoop fs -cat jfs://test/testfile
this is a test file
# 查看目录
[hadoop@emr-header-1 ~]$ hadoop fs -ls jfs://test/
Found 3 items
drwxrwxr-x - root root 0 2020-03-25 14:54 jfs://test/emr-header-1.cluster-50087
-rw-r---- 1 hadoop hadoop 5 2020-03-25 14:50 jfs://test/haha-12096RANDOM.txt
-rw-r---- 1 hadoop hadoop 20 2020-03-25 15:07 jfs://test/testfile
# 只读状态,不可修改文件
[hadoop@emr-header-1 ~]$ hadoop fs -rm jfs://test/testfile
java.io.IOException: ErrorCode : 25021 , ErrorMsg: Namespace is under recovery mode, and is read-only.
```

#### 8. 修改配置,将集群设置为正常模式,开启OTS异步上传功能。

在SmartData服务的bigboot页签,设置以下参数。

参数	描述	示例
namespace.backend.raft.async.ot s.enabled	是否开启OTS异步上传,包括: 。 true 。 false	true
anamespace.backend.raft.recovery. mode e false 是否开启从OTS恢复元数据,包括: o true o false		false

### 9. 重启集群。

- i. 单击上方的集群管理页签。
- ii. 在集群管理页面,单击相应集群所在行的更多 > 重启。

# 12.2.5. JindoFS权限功能

本文介绍JindoFS的Block模式支持的文件系统权限功能,包括Unix权限和Ranger权限两种。

# 背景信息

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。



Block模式支持Unix权限和Ranger权限两种文件系统权限功能:

- Unix权限:可以使用文件的777权限。
- Ranger权限:可以使用Ranger路径通配符等高级配置。

# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入bigboot服务配置页面。
  - i. 单击配置页签。
  - ii. 单击bigboot。

返回 📾 SmartData 🗸	●正常	
态 部署拓扑 配置	配置修改历史	
配置过滤		服务配置
配置搜索 请输入	Q	全部 「 smartdata-site 「 client 「 storage 「 <mark>bigboot</mark>

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。

#### 5. 重启配置。

i. 单击右上角的操作 > 重启 Jindo Namespace Service。

ii. 输入执行原因, 单击**确定**。

```
开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。
```

hadoop fs -chmod 777 jfs://{namespace\_name}/dir1/file1 hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/
ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

- 1. 添加Ranger。
  - i. 在bigboot页签, 单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的保存。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击确定。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。
- 2. 配置Ranger。
  - i. 进入Ranger UI页面。

详情请参见概述。

ii. Ranger UI添加HDFS service。

Ran	ger ØAccess Manager	🗅 Audit	Settings				😥 admin
Serv	ice Manager						
Servio	e Manager						🖬 Import 🚺 Export
	🗁 HDFS		+22	🕞 HBASE	+ 🖸 🖬	▷ HIVE	+ 🖬 😡
				emr-hbase	۲ 👔	emr-hive	

iii. 配置相关参数。

参数	说明	
Service Name	jfs-{namespace_name}。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal		
dfs.namenode.kerberos.principal	石塘空	
dfs.secondary.namenode.kerberos.principal	○ 小項与。	
Add New Configurations		

#### iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,才能获取LDAP的用户组 信息,从而对当前用户组进行Ranger权限的校验。

- 1. 在bigboot页签,单击自定义配置。
- 2. 在新增配置项对话框中,设置以下参数配置LDAP,单击确定。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	$org. a pache. hado op. security. Shell Based {\tt UnixGroupsMapping}$
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

⑦ 说明 配置项请遵循开源HDFS内容。

#### 3. 保存配置。

- i. 单击右上角的**保存**。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

### 4. 重启配置。

- i. 单击右上角的操作 > 重启 All Components。
- ii. 输入执行原因, 单击**确定**。
- 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。
   详情请参见Ranger Usersync集成LDAP。

# 12.3. JindoFS Cache模式

# 12.3.1. JindoFS缓存模式使用说明

缓存模式(Cache)主要兼容原生OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行缓存,提升EMR集群内访问OSS的效率,同时兼容了原有OSS原有文件形式,数据访问上能够与其他OSS客户端完全兼容。本文主要介绍JindoFS的缓存模式及其使用方式。

# 背景信息

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此和其他的各种OSS客户端是完全兼容的, 对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时集群中的缓存也能一定程度上提升数据访问性能,缓 解读写OSS的带宽压力。

# 配置使用方式

JindoFS缓存模式提供了以下两种基本使用方式,以满足不同的使用需求。

OSS Scheme

详情请参见配置OSS Scheme(推荐)。

 JFS Scheme 详情请参见配置JFS Scheme。

# 配置OSS Scheme(推荐)

OSS Scheme保留了原有OSS文件系统的使用习惯,即直接通过 oss://<bucket\_name>/<path\_of\_your\_file> 的形式访问OSS 上的文件。使用该方式访问OSS,无需进行额外的配置,创建EMR集群后即可使用,对于原有读写OSS的作业也无需做任何修改即可运行。

# 配置JFS Scheme

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > Smart Dat a**。
- 2. 进入bigboot服务配置页面。
  - i. 单击**配置**页签。
  - ii. 单击bigboot。

< 返回	📾 Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置过	t)逮				服务配置
配置換	<b>膝</b> 入			Q	全部 「 smartdata-site 」 client 「 storage 」 <mark>bigboot</mark>

3. 配置以下参数。

JindoFS支持多命名空间,本文命名空间以test为例。

i. 修改jfs.namespaces为test。

test表示当前JindoFS支持的命名空间,多个命名空间时以逗号(,)隔开。

ii. 单击自定义配置,在新增配置项对话框中增加以下参数。

参数	参数说明	示例	
		oss:// <oss_bucket>/<oss_dir>/</oss_dir></oss_bucket>	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。	⑦ 说明 该配置必须配置到OSS Bucket下的具体目录,也可以直接 使用根目录。	
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache	

### 4. 单击右上角的保存。

5. 单击右上角的操作 > 重启 Jindo Namespace Service。

重启后即可通过 jfs://test/<path\_to\_your\_file> 的形式访问,该命名空间下的文件会 以jfs.namespaces.test.oss.uri所配置的目录作为根目录进行组织,例如 jfs://test/hello.txt 对应实际OSS上的文 件为*oss://<oss\_bucket>/<oss\_dir>/hello.txt*。

# 启用缓存

启用缓存会利用本地磁盘对访问的热数据块进行缓存,默认状态为禁用,即所有OSS读取都直接访问OSS上的数据。

- 1. 在集群服务 > Smart Dat a的配置页面, 单击client 页签。
- 修改jfs.cache.data-cache.enable为1,表示启用缓存。
   此配置为客户端配置,不需要重启SmartData服务。

缓存启用后,Jindo服务会自动管理本地缓存备份,通过水位清理本地缓存,请您根据需求配置一定的比例用于缓存,详情请参 见磁盘空间水位控制。

# 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了 storage.watermark.high.ratio 和 storage.watermark.low.ratio 两个参数来调节本地存储的使用容量,值均为0 ~1的小数,表示使用磁盘空间的比例。

1. 修改磁盘水位配置。

可以在**服务配置**区域的storage页签,修改以下参数。

服务配置 全部   smartdata-site   client   storage   bigboot				
storage.watermark.low.ratio	0.2	0		
storage.watermark.high.ratio	0.4	0		
参数	描述			
storage.watermark.high.ratio	表示磁盘使用量的上水位比例,每块数据盘的JindoFS数据 用的磁盘空间到达上水位即会触发清理。默认值:0.4。	目录占		
storage.watermark.low.ratio	表示使用量的下水位比例,触发清理后会自动清理冷数据, JindoFS数据目录占用空间清理到下水位。默认值:0.2。	将		

⑦ 说明 您可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理的值即可。

### 2. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。
- 3. 重启Jindo Storage Service使配置生效。
  - i. 选择右上角的操作 > 重启Jindo Storage Service。
  - ii. 在执行集群操作对话框中,设置相关参数,单击确定。
  - iii. 在确认对话框中, 单击确定。

# 访问OSS Bucket

在EMR集群中访问同账号、同区域的OSS Bucket时,默认支持免密访问,即无需配置任何AccessKey即可访问。如果访问非以上 情况的OSS Bucket需要配置相应的AccessKey ID、AccessKey Secret以及Endpoint,针对两种使用方式相应的配置分别如下:

- OSS Scheme
  - i. 在集群服务 > SmartData的配置页面, 单击smartdata-site页签。
  - ii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明
fs.jfs.cache.oss-accessKeyld	表示存储后端OSS的AccessKey ID。
fs.jfs.cache.oss-accessKeySecret	表示存储后端OSS的AccessKey Secret。

参数	参数说明
fs.jfs.cache.oss-endpoint	表示存储后端OSS的endpoint。

- JFS Scheme
  - i. 在集群服务 > Smart Dat a的配置页面,单击bigboot页签。
  - ii. 修改jfs.namespaces为test。
  - iii. 单击自定义配置,在新增配置项对话框中增加以下参数,单击确定。

参数	参数说明	
jfs.namespaces.test.oss.uri	表示test命名空间的后端存储。示例: <i>oss://<oss_bucket.end point&gt;/<oss_dir< i="">&gt;。 endpoint信息直接配置在oss.uri中。</oss_dir<></oss_bucket.end </i>	
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。	

# 高级配置

Cache模式还包含一些高级配置,用于性能调优,以下配置均为客户端配置,修改后无需重启SmartData服务。

● 在服务配置区域的client页签,配置以下参数。

参数	参数说明
client.oss.upload.threads	每个文件写入流的OSS上传线程数。默认值:4。
client.oss.upload.max.parallelism	进程级别OSS上传总并发度上限,防止过多上传线程造成过大的带 宽压力以及过大的内存消耗。默认值:16。

● 在**服务配置**区域的smart dat a-sit e页签, 配置以下参数。

参数	参数说明
	rename过程使用普通copy接口的文件大小上限(小于阈值的使用 普通 copy接口,大于阈值的使用multipart copy接口以提高copy 效率)。
fs.jfs.cache.copy.simple.max.byte	⑦ 说明 如果确认已开通OSS fast copy功能,参数值设为-1,表示所有大小均使用普通copy接口,从而有效利用fast copy获得最优的rename性能。
fs.jfs.cache.write.buffer.size	文件写入流的buffer大小,参数值必须为2的幂次,最大为8MB,如 果作业同时打开的写入流较多导致内存使用过大,可以适当调小此 参数。默认值:1048576。
	启用Jindo Job Committer,避免Job Committer的rename操作, 来提升性能。默认值:true。
fs.oss.committer.magic.enabled	② 说明 针对Cache模式下,由于OSS这类对象存储 rename操作性能较差的问题,推出了Jindo Job Committer。

# 12.3.2. JindoFS权限功能

本文介绍JindoFS的Block模式支持的文件系统权限功能,包括Unix权限和Ranger权限两种。

# 背景信息

您可以在Apache Ranger组件上配置用户权限,在JindoFS上开启Ranger插件后,就可以在Ranger上对JindoFS权限(和其它组件 权限)进行一站式管理。



Block模式支持Unix权限和Ranger权限两种文件系统权限功能:

- Unix权限:可以使用文件的777权限。
- Ranger权限:可以使用Ranger路径通配符等高级配置。

# 启用JindoFS Unix权限

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入bigboot服务配置页面。
  - i. 单击配置页签。
  - ii. 单击bigboot。

< 返回	🔿 Smart	Data 🗸	●正常		
状态	部署拓扑	配置	配置修改历史		
配置达	土滤				服务配置
配置指	螦				全部 「 smartdata-site 」 client 「 storage 」 <mark>bigboot</mark>
请输	λ			Q	

- 3. 单击**自定义配置**,在新增配置项对话框中,设置Key为jfs.namespaces. <namespace>.permission.method, Value为unix,单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击**确定**。
- 5. 重启配置。
  - i. 单击右上角的操作 > 重启 Jindo Namespace Service。
  - ii. 输入执行原因, 单击**确定**。

开启文件系统权限后,使用方式跟HDFS一样。支持以下命令。

```
hadoop fs -chmod 777 jfs://{namespace_name}/dir1/file1
hadoop fs -chown john:staff jfs://{namespace name}/dir1/file1
```

如果用户对某一个文件没有权限,将返回如下错误信息。

[root@emr-header-1 ~]# hadoop fs -ls jfs://test/user/
ls: java.security.AccessControlException: Permission denied. Server Exception

# 启用JindoFS Ranger权限

- 1. 添加Ranger。
  - i. 在bigboot页签,单击自定义配置。
  - ii. 在新增配置项对话框中,设置Key为jfs.namespaces.<namespace>.permission.method, Value为ranger。
  - iii. 保存配置。
    - a. 单击右上角的**保存**。
    - b. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
    - c. 单击**确定**。
  - iv. 重启配置。
    - a. 单击右上角的操作 > 重启 Jindo Namespace Service。
    - b. 输入执行原因, 单击确定。

#### 2. 配置Ranger。

- i. 进入Ranger UI页面。
- 详情请参见<mark>概述</mark>。
- ii. Ranger Ul添加HDFS service。

Rang	er ØAccess Manager	🗅 Audit	Settings				🔐 admin
Servio	e Manager						
Service	Manager						C Import Export
					- <b>2</b> 0		+ <b>2</b> 0
e e				e monse			100
				emr-hbase	۲ کا ک	emr-hive	

iii. 配置相关参数。

参数	说明	
Service Name	jfs-{namespace_name}。	
Username	自定义。	
Password	自定义。	
Namenode URL	输入jfs://{namespace_name}。	
Authorization Enabled	使用默认值No。	
Authentication Type	使用默认值Simple。	
dfs.datanode.kerberos.principal		
dfs.namenode.kerberos.principal	了结束	
dfs.secondary.namenode.kerberos.principal	个具与。	
Add New Configurations		

### iv. 单击Add。

# 启用JindoFS Ranger权限+LDAP用户组

如果您在Ranger UserSync上开启了从LDAP同步用户组信息的功能,则JindoFS也需要修改相应的配置,才能获取LDAP的用户组 信息,从而对当前用户组进行Ranger权限的校验。

1. 在bigboot页签,单击自定义配置。

#### 2. 在新增配置项对话框中,设置以下参数配置LDAP,单击确定。

参数	示例
hadoop.security.group.mapping	org.apache.hadoop.security.CompositeGroupsMapping
hadoop.security.group.mapping.providers	shell4services,ad4users
hadoop.security.group.mapping.providers.combined	true
hadoop.security.group.mapping.provider.shell4services	org.apache.hadoop.security.ShellBasedUnixGroupsMapping
hadoop.security.group.mapping.provider.ad4users	org.apache.hadoop.security.LdapGroupsMapping
hadoop.security.group.mapping.ldap.url	ldap://emr-header-1:10389
hadoop.security.group.mapping.ldap.search.filter.user	(&(objectClass=person)(uid={0}))
hadoop.security.group.mapping.ldap.search.filter.group	(objectClass=groupOfNames)
hadoop.security.group.mapping.ldap.base	o=emr

⑦ 说明 配置项请遵循开源HDFS内容。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 重启配置。
  - i. 单击右上角的操作 > 重启 All Components。
  - ii. 输入执行原因, 单击确定。
- 通过SSH登录emr-header-1节点,配置Ranger UserSync并启用LDAP选项。
   详情请参见Ranger Usersync集成LDAP。

# 12.3.3. Jindo Job Committer使用说明

本文主要介绍JindoOssCommitter的使用说明。

### 背景信息

Job Committer是MapReduce和Spark等分布式计算框架的一个基础组件,用来处理分布式任务写数据的一致性问题。

Jindo Job Committer是阿里云E-MapReduce针对OSS场景开发的高效Job Committer的实现,基于OSS的Multipart Upload接口, 结合OSS Filesystem层的定制化支持。使用Jindo Job Committer时,Task数据直接写到最终目录中,在完成Job Commit前,中间 数据对外不可见,彻底避免了Rename操作,同时保证数据的一致性。

↓ 注意

- OSS拷贝数据的性能,针对不同的用户或Bucket会有差异,可能与OSS带宽以及是否开启某些高级特性等因素有关, 具体问题可以咨询OSS的技术支持。
- 在所有任务都完成后,MapReduce Application Master或Spark Driver执行最终的Job Commit操作时,会有一个短暂的时间窗口。时间窗口的大小和文件数量线性相关,可以通过增大 fs.oss.committer.threads 可以提高并发处理的速度。
- Hive和Presto等没有使用Hadoop的Job Committer。
- E-MapReduce集群中默认打开Jindo Oss Committer的参数。

# 在MapReduce中使用Jindo Job Committer
- 1. 进入YARN服务的mapred-site页签。
  - i.
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - ⅲ. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏单击集群服务 > YARN。
  - vi. 单击配置页签。
  - vii. 在**服务配置**区域,单击mapred-site页签。
- 2. 针对Hadoop不同版本,在YARN服务中配置以下参数。
  - Hadoop 2.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.class**为com.aliyun.emr.fs.oss.commit.JindoOssCommitter。

○ Hadoop 3.x版本

在YARN服务的**mapred-site**页签,设 置**mapreduce.outputcommitter.factory.scheme.oss**为com.aliyun.emr.fs.oss.commit.JindoOssCommitterFactory。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域, 单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。
- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - iii. 单击确定。

② 说明 在设置mapreduce.outputcommitter.class为com.aliyun.emr.fs.oss.commit.JindoOssCommitter后,可以通 过开关fs.oss.committer.magic.enabled便捷地控制所使用的Job Committer。当打开时,MapReduce任务会使用无需 Rename操作的Jindo Oss Magic Committer,当关闭时,JindoOssCommitter和FileOutputCommitter行为一样。

## 在Spark中使用Jindo Job Committer

- 1. 进入Spark服务的spark-defaults页签。
  - i. 在左侧导航栏单击集群服务 > Spark。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击spark-defaults页签。
- 2. 在Spark服务的spark-defaults页签,设置以下参数。

参数	参数值
spark.sql.sources.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.parquet.output.committer.class	com.aliyun.emr.fs.oss.commit.JindoOssCommitter
spark.sql.hive.outputCommitterClass	com.aliyun.emr.fs.oss.commit.JindoOssCommitter

这三个参数分别用来设置写入数据到Spark DataSource表、Spark Parquet格式的DataSource表和Hive表时使用的Job Committer。

- 3. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。
- 4. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > SmartData。
  - ii. 单击**配置**页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 5. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.magic.enabled为true。

⑦ 说明 您可以通过开关 fs.oss.committer.magic.enabled 便捷地控制所使用的Job Committer。当打开时, Spark任务会使用无需Rename操作的Jindo Oss Magic Committer,当关闭时, JindoOssCommitter和 FileOutputCommitter行为一样。

- 6. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## 优化Jindo Job Committer性能

当MapReduce或Spark任务写大量文件的时候,您可以调整MapReduce Application Master或Spark Driver中并发执行Commit相 关任务的线程数量,提升Job Commit性能。

- 1. 进入SmartData服务的smartdata-site页签。
  - i. 在左侧导航栏单击集群服务 > Smart Data。
  - ii. 单击配置页签。
  - iii. 在服务配置区域,单击smartdata-site页签。
- 2. 在SmartData服务的smartdata-site页签,设置fs.oss.committer.threads为8。
  - 默认值为8。
- 3. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

## 12.4. JindoTable

## 12.4.1. JindoCube使用说明

JindoCube在E-MapReduce 3.24.0及之后版本中可用。本文主要介绍E-MapReduce JindoCube的安装、部署和使用等。

## 前提条件

已创建表或者视图。

## 概述

JindoCube是E-MapReduce Spark支持的高级特性,通过预计算加速数据处理,实现十倍甚至百倍的性能提升。您可以将任意 View表示的数据进行持久化,持久化的数据可以保存在HDFS或OSS等任意Spark支持的Dat aSource中。EMR Spark自动发现可用 的已持久化数据,并优化执行计划,对用户完全透明。JindoCube主要用于查询模式相对比较固定的业务场景,通过提前设计 JindoCube,对数据进行预计算和预组织,从而加速业务查询的速度,常见的使用场景包括MOLAP多维分析、报表生成、数据 Dashboard和跨集群数据同步等。

## JindoCube的安装与部署

JindoCube作为EMR Spark组件的高级特性,所有使用EMR Spark提交的Dat aset 、Dat aFrame API、SQL任务,均可以基于 JindoCube进行加速,无须额外的组件部署与维护。

1. UI页面展示。

JindoCube主要通过Spark的UI页面进行管理,包括JindoCube的创建、删除和更新等。通过UI创建JindoCube完成后,即可自动用于该集群所有Spark任务的查询加速。通过spark.sql.cache.tab.display参数可以控制是否在Spark UI页面展示 JindoCube的Tab,可以通过EMR控制台在Spark服务中配置相关参数,或者在Spark提交命令中指定参数值,该参数默认值为false。

Spark 24.3	Jobs	Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application UI
Cube Manager	nent									

JindoCube还提供了**spark.sql.cache.useDatabase**参数,可以针对业务方向,按不同的业务建立database,把需要建 cache的view放在这个database中。对于分区表JindoCube还提供了**spark.sql.cache.cacheByPartition**参数,可指定 cache使用分区字段进行存储。

参数	说明	示例值
spark.sql.cache.tab.display	显示Cube Mangament页面。	true
spark.sql.cache.useDatabase	cube存储数据库。	db1,db2,dbn
spark.sql.cache.cacheByPartition	按照分区字段存储cube。	true

#### 2. 优化查询。

**spark.sql.cache.queryRewrite**用于控制是否允许使用JindoCube中的Cache数据加速Spark查询任务,用户可以在集群、 session、SQL等层面使用该配置,默认值为t**rue**。

## JindoCube的使用

- 1. 创建JindoCube。
  - i. 通过阿里云账号登录阿里云 E-MapReduce 控制台。
  - ii. 单击**集群管理**页签。
  - iii. 单击待操作集群所在行的集群ID。
  - iv. 单击左侧导航栏的**访问链接与端口**。
  - v. 在**公网访问链接**页面,单击YARN UI所在行的链接,进入Knox代理的YARN UI页面。 Knox相关使用说明请参见Knox。
  - vi. 单击Name为Thrift JDBC/ODBC Server, Application Type为SPARK所在行的ApplicationMaster。
  - vii. 单击上方的Cube Management页签。
  - viii. 单击New Cache。

您可以选择某一个表或视图,单击action中的链接继续创建Cache。可以选择的Cache类型分为两类:

Raw Cache: 某一个表或者视图的raw cache,表示将对应表或视图代表的表数据按照指定的方式持久化。 在创建Raw Cache时,需要指定如下信息:

参数	描述	是否必选
Cache Name	指定Cache的名字,支持字母、数字、 连接号(-)和下划线(_)的组合。	必选
Column Selector	选择需要Cache哪些列的数据。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

## ■ Cube Cache:基于某一个表或者视图的原始数据,按照用户指定的方式构建cube,并将cube数据持久化。

在创建Cube Cache时,用户需要指定如下信息:

参数	描述	是否必选
Cache Name		必选
Dimension Selector	选择构建Cube时的维度字段。	必选
Measure Selector	选择构建Cube时的measure字段和 measure预计算函数。	必选
Rewrite	是否允许该Cache被用作后续查询的执 行计划优化。	必选
Provider	Cache数据的存储格式,支持JSON、 PARQUET、ORC等所有Spark支持的数 据格式。	必选
Partition Columns	Cache数据的分区字段。	可选
ZOrder Columns	ZOrder是一种支持多列排序的方 法,Cache数据按照ZOrder字段排序 后,对于基于ZOrder字段过滤的查询会 有更好的加速效果。	可选

Table and View Information	n	
dataBase	tableName	CubeCache action
ssb_10	customer	c_name     raw cache       c_address     cube cache
ssb_10	dates	C_crity raw cache C_nation cube cache c region cube cache
ssb_10	lineorder	c_phone raw cache c_mktsegment cube cache
ssb_10	lineorder_flatten	p_parkey     raw cache     p_name     cube cache     cube cache
ssb_10	p_lineorder	p_name         raw cache           p_category         raw cache           cube cache         cube cache
ssb_10	part	p_color     raw cache       p_lype     cube cache
ssb_10	supplier	D_size D_container raw cache cube cache
		Masure selector     Io_quantity
		lo_tax \$ SUM \$ ₽ ■

JindoCube通过用户指定的Dimension和Measure信息来构建Cube,对于上图的示例,创建的Cube Cache可以用SQL 表示为:

SELECT c\_city, c\_nation, c\_region, MAX(lo\_quantity), SUM(lo\_tax)
FROM lineorder\_flatten
GROUP BY c\_city, c\_nation, c\_region;

JindoCube计算Cube的最细粒度维度组合,在优化使用更粗粒度的维度组合的查询时,基于Spark强大的现场计算能力,通过重聚合实现。在定义Cube Cache时,必须使用JindoCube支持的预计算函数。JindoCube支持的预计算函数和其对应的聚合函数类型如下:

聚合函数类型	预计算函数
COUNT	COUNT
SUM	SUM
MAX	MAX
MIN	MIN
AVG	COUNT, SUM
COUNT ( DIST INCT )	PRE_COUNT_DIST INCT
APPROX_COUNT_DISTINCT	PRE_APPROX_COUNT_DISTINCT

在**Cube Management**页面,展示所有的Cache列表。单击**Det ail**进入Cache的详细页面,在Cache详细页会展示 Cache的详细信息、包括基本信息、Cache数据分区信息、构建Cache信息以及构建历史信息等。

2. 构建JindoCube。

创建JindoCube Cache只是进行元数据操作,Cache表示的数据并未持久化,需要继续构建Cache,从而持久化Cache数据到 HDFS或OSS等存储中。此外Cache对应的源表数据可能会新增或者更新,需要更新Cache中的数据从而保持一致。JindoCube 支持两类构建操作:

• Build Cache。

通过Build Cache链接,用户可以主动触发一次构建操作,构建页面相关信息如下:

Spark 2.4.3	Jobs Stages Stora	ge Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manager	ment Build Page						
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Optional Filter							
Column:	s_region	\$					
Filter Type:	<ul> <li>Fixed Values</li> <li>Range Values</li> </ul>						
Values:	ASIA,AMERICA	Filter view data w	vith column value equ	als to input v	values.		
	Submit						

在构建JindoCube的Cache时,相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Optional Filter	用户可以选择额外的过滤条件,在构建时,将该Cache表示的数 据过滤后再持久化。 Column:过滤字段。 Filter Type:过滤类型,支持固定值和范围值两种。 Fixed Values:指定过滤值,可以多个,以","分隔。 Range Values:指定范围值的最小和最大值,最大值可以 为空,过滤条件包含最小值,不包含最大值。

上图中构建任务想要构建lineorder\_flatten视图的Raw Cache数据,要写入Cache中的数据可以使用如下SQL表示:

```
SELECT * FROM lineorder_flatten
WHERE s_region == 'ASIA' OR s_region == 'AMERICA';
```

单击**Submit**,提交构建任务,返回到Cache详细页面,对应的构建任务会提交到Spark集群中执行,在Build Information 中可以看到当前是否正在构建Cache的信息。在Cache构建完成后,可以在Build History中看到相关的信息。

⑦ 说明 Cache数据由Spark任务写到一个指定目录中,和普通的Spark写表或者写目录一样,对于Parquet、 Json、ORC等数据格式,并发构建同一个Cache可能导致Cache数据不准确,不可用,应避免这种情况。如果无法避免 并发构建、更新Cache,可以考虑使用delta等支持并发写的数据格式。

• Trigger Period Build。

定期更新功能可以方便用户设置自动更新Cache的策略,保持Cache数据和源表数据的一致。相关页面如下:

Spork 2.4.3	Jobs Stages Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server
Cube Manage	ment Timer Trigger I	Build Page	е				
Save Mode:	<ul> <li>Overwrite</li> <li>Append</li> </ul>						
Trigger Strategy:							
Start At:	2019-12-30 11:00:00						
Period:	1	Hour		\$			
Optional Step:	۵						
Step By:	TimeStamp Column(Long Type)     DateTime Column(String Type)						
Column Name:	lo_orderdate \$						
DateTime Format	Such as 'yyyy-MM-dd Hł If availat	ble, filter would com	oare column as Tirr	estamp	Type value.		
	Submit						

定期更新的相关用户选项如下:

参数	描述
Save Mode	支持Overwrite和Append两种模式。 Overwrite: 会覆盖之前曾经构建的Cache数据。 Append: 会新增数据到Cache中。
Trigger Strategy	触发策略,设置触发构建任务的开始时间和间隔时间。 Start At:通过时间控件选择或者手工输入第一次触发构建任 务的时间点,日期格式为yyyy-MM-dd hh:mm:ss。 Period:设置触发构建任务的间隔时间。
Optional Step	设置每次触发构建任务的数据筛选条件,通过指定时间类型的字段,配合触发策略中的间隔时间,可以实现按照时间间隔增量的更新Cache。如果不选择,每次全量更新Cache。 Step By:选择增量更新字段类型,只支持时间类型字段,包括Long类型的timestamp字段,以及指定dateformat信息的String类型字段。 Column Name:增量更新字段名称。

在Cache详细页面中,可以看到当前设置的定期更新策略,用户可以随时通过Cancel Period Build取消定期更新。所有触发的构建任务信息在完成后也可以在Build History列表中看到。

? 说明

- 定期更新任务是Spark集群级别的,相关设置保存在SparkContext中,并由SparkDriver定期触发,当Spark集群关闭后,定期更新任务也随之关闭。
- 当前Spark集群所有的构建任务完成后,都会展示在Build History列表中,包含开始/结束时间、SaveMode、 构建条件,任务最终状态等。Build History也是Spark集群级别的信息,当Spark集群关闭后,相关信息也随之 释放。

3. 管理JindoCube。

创建和构建JindoCube的Cache数据后,通过Cube Management的UI页面,可以对JindoCube的Cache数据进行进一步的管理。

● 删除cache。

在JindoCube Cache列表页面,可以通过action列的**Drop**删除对应Cache,删除成功后,Cache的相关元数据和存储数据 都会被清理。

Spork 2.4.3 Job	Stages Storage Environ	ment Executors SQL	Cube Management Streaming	JDBC/ODBC Server		Thr	ift JDBC/ODBC Server application UI
Cube Management							
viewDataBase	viewName	cacheName	enableRewrite	cacheType	cacheDataSize	lastUpdataTime	action
ssb_10	lineorder_flatten	lo_raw_cache	false	Raw Cache	28 GB	2019/12/28 17:52:55	Detail Drop
New Cache							

○ 开启或关闭Cache优化。

JindoCube支持在Cache级别,设置是否允许用于Spark查询的优化,在Cache的详细页面,您可以通过基本信息中的Enabled或Disabled,启用或者停用该Cache,控制是否允许该Cache用于查询加速。

Spork 24.3	Jobs Stages	Storage	Environment	Executors	SQL	Cube Management	Streaming	JDBC/ODBC Server	Thrift JDBC/ODBC Server application	UI
Cube Manage	ment Detail F	Page								
Name								Value		
Database								ssb_10		
View Name								lineorder_flatten		
Cache Name								lo_raw_cache		
Enabled								true		
Cache Type								RAW		
Cache Columns								Is, orderwey, lo, linerumber, lo, custery, lo, parkey, lo, support, lo, orderateta lo, orderprofito, lo, and lo, contratato, lo, addisorda, lo, ander lo, lo, contratato, lo, addisorda, lo, ander lo, custeratori, lo, addisorda, lo,	ppriotity, lo_quantity, lo_extendedprice, ppkey,name, s_address, s_city, s_nation, phth,daynuminmovek,daynuminnonth, dayinmonthfi, d_holidayfi, d_weekdayfi, ame, p_mfgr, p_category, p_brand, p_color,	
Location								/user/hive/warehouse/ssb_10/_CACHE/lineorder_flatten/raw		
Data Size								None		
Provider								PARQUET		
Partition By								lo_orderdate		
ZOrder By										
Last Update Time								2019/12/28 17:52:55		
Enabled > Partition Information Show 10 ¢ entries									Search:	
Path					Size			Action		
							No data av	allable in table		
Showing 0 to 0 of 0 entries	s								Previous Next	
Build Information     Latest cache building is fire	nished at 2019-12-28 17	:52:55								
Build Cache Trigger	Period Build									

○ 删除分区数据。

如果Cache的数据是按照分区存储的,当确认某些分区数据不再需要时,删除这些分区数据可以节省大量存储空间。在 Cache的详细页面,分区Cache的相关分区会通过列表展示,用户可以通过**Delet**e删除特定分区的数据。

Path	Size	Action
lo_orderdate=19920101	12 MB	Delete
lo_orderdate=19920102	12 MB	Delete
lo_orderdate=19920103	12 MB	Delete
lo_orderdate=19920104	12 MB	Delete
lo_orderdate=19920105	12 MB	Delete
lo_orderdate=19920106	12 MB	Delete
lo_orderdate=19920107	12 MB	Delete
lo_orderdate=19920108	12 MB	Delete
lo_orderdate=19920109	12 MB	Delete
lo_orderdate=19920110	12 MB	Delete

⑦ 说明 在删除Cache分区数据之前,请谨慎确认,确保该分区数据不会被使用。如果用户的查询经过优化需要用 到该Cache被删除的分区数据,会导致错误的查询结果。

4. 查询优化。

目前JindoCube支持基于View的查询优化,当用户使用某个视图创建了Raw Cache或者Cube Cache后,后续的查询使用到了 该视图,EMR Spark会在满足逻辑语义的前提下,尝试使用Cache重写查询的执行计划,新的执行计划直接访问Cache数据, 从而加速查询速度。以如下场景为例,lineorder\_flatten视图是将lineorder和其他维度表关联之后的大宽表视图,其相关定 义如下:

<pre>[0: jdbc:hive2://localhost:10001/s 19/12/30 14:08:16 WARN [main] Hiv 19/12/30 14:08:16 WARN [main] Hiv</pre>	ssb_10> desc extended lineorder_flatten; /eConf: HiveConf of name hive.enforce.bucketing does /eConf: HiveConf of name hive.support.sql11.reserved.	not exist keywords do	es not exist
col_name	data_type	comment	
lo_orderkey	bigint	NULL	
lo_linenumber	bigint	NULL	
lo_custkey	int	NULL	
lo_partkey	int	NULL	
lo_suppkey	1nt	NULL	
lo orderpriotity	1nt	NULL	
lo shippriotity	lint	NULL	
lo quantity	bigint	NULL	
lo_extendedprice	bigint	NULL	
lo_ordtotalprice	bigint	NULL	
lo_discount	bigint	NULL	
lo_revenue	bigint	NULL	
lo_supplycost	bigint	NULL	
lo_tax	bigint	NULL	
lo shipmode	1nt etring	NULL	
s suppkey	string	NULL	
s name	string	NULL	
s_address	string	NULL	
s_city	string	NULL	
s_nation	string	NULL	
s_region	string	NULL	
s_phone	string	NULL	
d date	LINL Letring	NULL	
d davofweek	string	NULL	
d month	string	NULL	
d vear	int	NULL	
d_yearmonthnum	int	NULL	
d_yearmonth	string	NULL	
d_daynuminweek	int	NULL	
d_daynuminmonth	int	NULL	
d_daynuminyear	int	NULL	
d_monthnuminyear	int int	NULL	
d_weekhuminyear	string	NULL	
d lastdavinweekfl	int	NULL	
d_lastdayinmonthfl	int	NULL	
d_holidayfl	int	NULL	
d_weekdayfl	int	NULL	
c_custkey	int	NULL	
c_name	string	NULL	
c_address	string	NULL	
c_city	string	NULL	
c region	string	NULL	
c_phone	string	NULL	
c_mktsegment	string	NULL	
p_partkey	int	NULL	
p_name	string	NULL	
p_mfgr	string	NULL	
p_category	string	NULL	
p_brand	string	NULL	
p type	string	NULL	
p_size	int	NULL	
p_container	string	NULL	
		!	
# Detailed Table Information		1	
Database	SSD_10		
Owner	hadoon		
Created Time	Sat Dec 28 17:30:02 CST 2019		
Last Access	Thu Jan 01 08:00:00 CST 1970	i	
Created By	Spark 2.2 or prior	i	
Туре	VIEW	j .	
View Text	SELECT `lineorder`.`lo_orderkey`, `lineorder`.`lo_l	inenumber`,	`lineorder`.`lo_custkey`, `lineorder`.`lo_partkey`, `
lineorder'.'lo_suppkey', 'lineord	der'.'lo_orderdate', 'lineorder'.'lo_orderpriotity',	'lineorder'	. lo_shippriotity', 'lineorder'. 'lo_quantity', 'lineor
der . 10_extendedprice , lineoro	der . 10_ordtotalprice , lineorder . 10_discount ,	lineorder .	lo_revenue , lineorder . lo_supplycost , lineorder
city` `eupplier` `e pation` `eu	upplier``s region```supplier``s phone```dates``d	datekey`	'dates' `d date` `dates` `d davofwaek` `dates` `d mo
nth`, `dates`.`d year`, `dates`.	d vearmonthnum`, `dates`,`d vearmonth`, `dates`,`d d	avnuminweek	`. `dates`.`d davnuminmonth`. `dates`.`d davnuminvear`
, `dates`.`d_monthnuminyear`, `da	ates`.`d_weeknuminyear`, `dates`.`d_sellingseason`, `	dates`.`d_l	astdayinweekfl`, `dates`.`d_lastdayinmonthfl`, `dates`
.`d_holidayfl`, `dates`.`d_weekda	ayfl`, `customer`.`c_custkey`, `customer`.`c_name`, `	customer`.`	c_address`, `customer`.`c_city`, `customer`.`c_nation`
, `customer`.`c_region`, `custome	er`.`c_phone`, `customer`.`c_mktsegment`, `part`.`p_p	artkey`, `p	art`.`p_name`, `part`.`p_mfgr`, `part`.`p_category`, `
part`.`p_brand`, `part`.`p_color	`, `part`.`p_type`, `part`.`p_size`, `part`.`p_contai	ner`FROM `	ssb_10`.`lineorder`, `ssb_10`.`supplier`, `ssb_10`.`da
tes , ssb_10 .customer , ssb_1	LU . part	·	The supplication of the su
""" s suppkey AND lipeorder 'lo	<pre>- uales . u_ualekey AND lineorder . lo_custkey = partkey = `part'.`p partkey`  </pre>	cuscomer	. c_custkey AND iineorder . io_suppkey = supplier
Table Properties	[transient lastDdlTime=1577677599]	1	
Serde Library	org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe	i	
InputFormat	org.apache.hadoop.mapred.TextInputFormat	i	
OutputFormat	org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutpu	tFormat	
Storage Properties	[serialization.format=1]	1	
73 rows selected (0.057 seconds)		+	•

0: jdbc:hive2://localhost:10001/ssb\_10>

基于lineorder\_flatten视图简单查询的执行计划如下:

[0: jdbc:hive2://localhost:10001/ssb_10> explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:19:32 WARN [main] HiveConf: HiveConf of name hive.enforce.bucksting does not exist 19/12/30 4:19:32 WARN [main] HiveConf: HiveConf of name hive.support.sull.reserved.keywords does not exist	
+   plan	
Collectimit 10 +- *(11) SortMegJoin [lo_partkey#313], [p_partkey#359], Inner	
:- *(8) Sort [10_partkey#313 ASC WULLS F1KS]], Talse, 0 : +- Exchange hashpartitioning[10_partkey#313, 200]	
: +- *(/) SortWergeJoin [io_custkey#312], [c_custkey#351], inner : :- *(4) Sort [lo_custkey#312 ASC NULLS FIRST], false, 0	
: +- Exchange hashpartitioning(io_custkey#312, 200) : +- *(3) BroadcastHashJoin [lo_orderdate#315], [d_datekey#334], Inner, BuildRight	
: : :- *(3) proadcastmasnJoin [io_suppkey#324], [s_suppkey#32/], inner, Bulldkight : : :- *(3) Filter (((isnotnull(lo_suppkey#314) & isnotnull(lo_orderdate#315)) & isnotnull(lo_custkey#312)) & isnotnull(lo_part key#313))	
: : : + → Scan hive ssb_10.lineorder [lo_orderkey#310], lo_linenumber#311, lo_custkey#312, lo_partkey#312, lo_suppkey#314, lo_creater erdate#315, lo_orderpriotity#316, lo_shippriotity#317, lo_quantity#3181, lo_extendedprice#3104, lo_ordtotalprice#3204, lo_discount#3211, lo_revenue#32 21, lo_supplycost#3231, lo_tax#3241, lo_commitdate#325, lo_shipmode#3261, loight, HiveTableRelation `ssb_10'.lineorder`, org.apache.hadoop.hive.serde2.lazy.L azySimpleSerDe, [lo_orderkey#31841, lo_ienumber#3111, lo_custkey#312, lo_partkey#3124, lo_crederdate#315, lo_orderprivity#316, lo_ship priotity#317, lo_quantity#3181, lo_textendedprice#319L, lo_ordtotalprice#320L, lo_discount#321L, lo_revenue#322L, lo_supplycost#3231, lo_tax#3241, lo_c	
<pre>: : +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) : : +- *(1) Filter isnotnull(s suppkev#327)</pre>	
: : +- Scan hive ssb_10.supplier [s_suppkey#327, s_name#328, s_address#329, s_city#330, s_nation#331, s_region#332, s_phone# 333], HiveTableRelation `ssb_10`.`supplier`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [s_suppkey#327, s_name#328, s_address#329, s_city#330 . s nation#331. s region#332. s phone#333]	
<pre>: +- BroadcastExchange HashedRelationBroadcastMode(List(cast(input[0, int, false] as bigint))) : +- *(2) Filter isnotnull(d_datekey#334)</pre>	
: Scan hive ssb_10.dates [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearm onth#340, d_daynuminwek#341, d_daynuminmonth#342, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinwek fl#347, d_lastdayinmonthfl#348, d_holidayfl#349, d_weekdayfl#350], HiveTableRelation `ssb_10'.dates', org.apache.hadoop.hive.sorde2.lazy.tazySimpleSe rDe, [d_datekey#334, d_date#335, d_dayofweek#336, d_month#337, d_year#338, d_yearmonthnum#339, d_yearmonth#340, d_daynuminweek#341, d_daynuminmonth#34 2, d_daynuminyear#343, d_monthnuminyear#344, d_weeknuminyear#345, d_sellingseason#346, d_lastdayinweekfl#347, d_lastdayinmonthfl#348, d_holidayfl#349, d_weekdayfl#3560]	
: +- *(6) Sort [c_custkey#351 ASC NULLS FIRST], false, 0 : +- Exchange hashpartitioning(c_custkey#351, 200)	
: +- *(5) Filter ((isnotnull(c_region#356) && (c_region#356 = ASIA)) && isnotnull(c_custkey#351)) : +- Scan hive ssb_10.customer [c_custkey#351, c_name#352, c_address#353, c_city#354, c_nation#355, c_region#356, c_phone#357, c_mk tsegment#358], HiveTableRelation `ssb_10`.`customer`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [c_custkey#351, c_name#352, c_address#353, c _city#354, c_nation#355, c_region#356, c_phone#357, c_mktsegment#358]	
+- *(10) Sort [p_partkey#359 ASC NULLS FIRST], false, 0 +- Exchange hashpartitioning(p_partkey#359, 200)	
+- *(9) Filter isnotnull(p_partkey#359) +- Scan hive ssb_10.part [p_partkey#359, p_name#360, p_mfgr#361, p_category#362, p_brand#363, p_color#364, p_type#365, p_size#366, p_conta	
<pre>iner#367], HiveTableRelation `ssb_10`.`part`, org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe, [p_partkey#359, p_name#360, p_mfgr#361, p_category#3 62, p_brand#363, p_color#364, p_type#365, p_size#366, p_container#367]</pre>	1
1 row selected (0.435 seconds)	
0: jobc:nive2://localnost:10001/ssb_10>	ï
在为line order_flatten视图创建Raw Cache并构建完成后,执行相同查询,EMR Spark会自动使用Cache数据优化执行计	
划,优化后的执行计划如下:	
<pre>[0: jdbc:hive2://localhost:10001/ssb_10&gt; explain select * from lineorder_flatten where c_region = 'ASIA' limit 10; 19/12/30 14:17:47 INFO [main] HiveConf: Found configuration file file:/ecm/spark-conf-2.4.3-hadoop2.8-1.4.2/hive-site.xml 19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.enforce.bucketing does not exist 19/12/30 14:17:47 WARN [main] HiveConf: HiveConf of name hive.support.sql11.reserved.keywords does not exist</pre>	
+	
<pre>++   == Physical Plan ==</pre>	
<pre>Collectimit 10 + *(1) Project [lo_orderkey#128L, lo_linenumber#129L, lo_custkey#130, lo_partkey#131, lo_suppkey#132, lo_orderdate#133, lo_orderpriotity#134, lo_ship priotity#135, lo_quantity#136L, lo_extendedprice#137L, lo_ordetalprice#138L, lo_discount#139L, lo_revenue#140L, lo_suppkey#141L, lo_tax#142L, lo_c ommtdate#143, lo_shipmode#144, s_uppkey#145, s_name#146, s_address#147, s_city#148, s_nation#149, s_region#156, s_phone#151, 34 more fields] + *(1) Filter (isnotnull(c_region#174) &amp;&amp; (c_region#174 = ASIA)) </pre>	
+- +(1) FileScan parquet [10_0fderkey#126L,10_inenumber#124L,10_custkey#130,10_partkey#131,10_suppkey#132,10_ofderpriotity#134,10_shippriotity#	

可以看到,优化后的执行计划省去了lineorder\_flatten视图的所有计算逻辑,直接访问HDFS中Cache的数据。

## 注意事项

- 1. JindoCube并不保证Cache数据和源表数据的一致性,而是需要用户通过手工触发或者设置定期策略触发更新任务同步Cache中的数据,用户需要根据查询对于数据一致性的需求,触发Cache的更新任务。
- 2. 在对查询的执行计划进行优化的时候, JindoCube根据视图的元数据判断是否可以使用Cache优化查询的执行计划。优化后,如果Cache的数据不完整,可能会影响查询结果的完整性或正确性。可能导致Cache数据不完整的情况包括:用户在Cache详情页主动删除查询需要的Cache Partition数据,构建、更新Cache时指定的过滤条件过滤掉了查询需要的数据,查询需要的数据还未及时更新到Cache等。

## 12.5. 工具集

## 12.5.1. Jindo DistCp使用说明

本文介绍JindoFS的数据迁移工具Jindo DistCp的使用方法。

## 前提条件

已创建集群,详情请参见创建集群。

## 使用Jindo Distcp

1. 通过SSH方式连接集群。

详情请参见登录集群。

2. 执行以下命令, 获取帮助信息。

jindo distcp --help

#### Jindo Dist Cp参数详细信息如下:

- --src和--dest
- --parallelism
- --srcPattern
- --deleteOnSuccess
- --outputCodec
- --out put Manifest和--requirePreviousManifest
- --out put Manifest 和--previousManifest
- --copyFromManifest
- --srcPrefixesFile
- --groupBy和-targetSize
- --enableBalancePlan
- --enableDynamicPlan
- --enableTransaction
- --diff
- 查看Dist cp Counters
- 使用OSS AccessKey
- 使用归档或低频写入OSS
- 清理残留文件

#### --src和--dest

```
--src 表示指定源文件的路径, --dest 表示目标文件的路径。
```

Jindo Dist Cp默认将 --src 目录下的所有文件拷贝到指定的 --dest 路径下。您可以通过指定 --dest 路径来确定拷贝后的 文件目录,如果不指定根目录, Jindo Dist Cp会自动创建根目录。

例如,您可以执行以下命令,将/opt/tmp下的文件拷贝到OSS Bucket。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp

⑦ 说明 本文示例中的yourBucket Name是您OSS Bucket的名称。

## --parallelism

--parallelism 用于指定MapReduce作业里的mapreduce.job.reduces参数。该参数默认为7,您可以根据集群的资源情况,通过自定义 --parallelism 大小来控制Dist Cp任务的并发度。

例如,将HDFS上/opt/tmp目录拷贝到OSS Bucket,可以执行以下命令。

jindo distcp --src /opt/tmp --dest oss://<yourBucketName>/tmp --parallelism 20

## --srcPattern

--srcPattern 使用正则表达式,用于选择或者过滤需要复制的文件。您可以编写自定义的正则表达式来完成选择或者过滤操 作,正则表达式必须为全路径正则匹配。

## 例如,如果您需要复制/data/incoming/hourly\_table/2017-02-01/03下所有log文件,您可以通过指定 --srcPattern 的正则 表达式来过滤需要复制的文件。

#### 执行以下命令,查看/data/incoming/hourly\_table/2017-02-01/03下的文件。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

#### 返回信息如下。

Found 6 item	ound 6 items							
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst	
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log	
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log	
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001	
09								
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt	
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly table/2017-02-01/03/emp06.txt	

#### 执行以下命令,复制以log结尾的文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --srcPattern .\*\.lo
g --parallelism 20

#### 执行以下命令,查看目标bucket的内容。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下,显示只复制了以log结尾的文件。

```
      Found 2 items

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/1.log

      -rw-rw-rw-
      1

      4891 2020-04-17 20:52 oss://<yourBucketName>/hourly_table/2017-02-01/03/2.log
```

#### --deleteOnSuccess

--deleteOnSuccess 可以移动数据并从源位置删除文件。

例如,执行以下命令,您可以将/data/incoming/下的hourly\_table文件移动到OSS Bucket中,并删除源位置文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --deleteOnSuccess -
-parallelism 20
```

### --outputCodec

```
--outputCodec 可以在线高效地存储数据和压缩文件。
```

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputCodec=gz -parallelism 20

#### 目标文件夹中的文件已经使用gz编解码器压缩了。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table/2017-02-01/03

#### 返回信息如下:

Found 6 item	ns	
-rw-rw-rw-	1	938 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/000151.sst.gz</yourbucketname>
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/1.log.gz</yourbucketname>
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/2.log.gz</yourbucketname>
-rw-rw-rw-	1	1956 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/OPTIONS-000109.g</yourbucketname>
Z		
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/emp01.txt.gz</yourbucketname>
-rw-rw-rw-	1	506 2020-04-17 20:58 oss:// <yourbucketname>/hourly_table/2017-02-01/03/emp06.txt.gz</yourbucketname>

Jindo DistCp当前版本支持编解码器gzip、gz、lzo、lzop、snappy以及关键字none和keep(默认值)。关键字含义如下:

- none表示保存为未压缩的文件。如果文件已压缩,则jindo DistCp会将其解压缩。
- keep表示不更改文件压缩形态,按原样复制。

```
② 说明 如果您想在开源Hadoop集群环境中使用编解码器Izo,则需要安装gplcompression的native库和hadoop-Izo包。
```

## --outputManifest和--requirePreviousManifest

--outputManifest 可以指定生成DistCp的清单文件,用来记录copy过程中的目标文件、源文件和数据量大小等信息。

如果您需要生成清单文件,则指定 --requirePreviousManifest 为 false 。当前output Manifest文件默认且必须为gz类型 压缩文件。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --outputManifest=ma nifest-2020-04-17.gz --requirePreviousManifest=false --parallelism 20

#### 查看outputManifest文件内容。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz > before.lst
cat before.lst

#### 返回信息如下。

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/000151.sst","baseName":"2017-02-01/03/000151.sst",
"srcDir":"oss://<yourBucketName>/hourly\_table","size":2252}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/1.log","baseName":"2017-02-01/03/1.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/2.log","baseName":"2017-02-01/03/2.log","srcDir":"
oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/OPTIONS-000109","baseName":"2017-02-01/03/OPTIONS-000109","srcDir":"oss://<yourBucketName>/hourly\_table","size":4891}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp01.txt","baseName":"2017-02-01/03/emp01.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

{"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/emp06.txt","baseName":"2017-02-01/03/emp06.txt","s
rcDir":"oss://<yourBucketName>/hourly\_table","size":1016}

#### --outputManifest和--previousManifest

--outputManifest 表示包含所有已复制文件(旧文件和新文件)的列表, --previousManifest 表示只包含之前复制文件 的列表。您可以使用 --outputManifest 和 --previousManifest 重新创建完整的操作历史记录, 查看运行期间复制的文 件。

#### 例如,在源文件夹中新增加了两个文件,命令如下所示。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --outputManifest=ma
nifest-2020-04-18.gz --previousManifest=oss://<yourBucketName>/hourly_table/manifest-2020-04-17.gz --parallel
ism 20
```

执行以下命令,查看运行期间复制的文件。

hadoop fs -text oss://<yourBucketName>/hourly\_table/manifest-2020-04-18.gz > current.lst
diff before.lst current.lst

### 返回信息如下。

3a4,5

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/5.log","baseName":"2017-02-01/03/5.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

> {"path":"oss://<yourBucketName>/hourly\_table/2017-02-01/03/6.log","baseName":"2017-02-01/03/6.log","srcDir"
:"oss://<yourBucketName>/hourly\_table","size":4891}

## --copyFromManifest

使用 --outputManifest 生成清单文件后,您可以使用 --copyFromManifest 指定 --outputManifest 生成的清单文件,并 将dest目录生成的清单文件中包含的文件信息拷贝到新的目录下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --previousManifest= oss://<yourBucketName>/hourly\_table/manifest-2020-04-17.gz --copyFromManifest --parallelism 20

#### --srcPrefixesFile

--srcPrefixesFile 可以一次性完成多个文件夹的复制。

示例如下,查看hourly\_table下文件。

hdfs dfs -ls oss://<yourBucketName>/hourly\_table

#### 返回信息如下。

```
      Found 4 items

      drwxrwxrwx -
      0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-01

      drwxrwxrwx -
      0 1970-01-01 08:00 oss://<yourBucketName>/hourly_table/2017-02-02
```

#### 执行以下命令,复制hourly\_table下文件到folders.txt。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --srcPrefixesFile f
ile:///opt/folders.txt --parallelism 20
```

#### 查看folders.txt文件的内容。

cat folders.txt

#### 返回信息如下。

```
hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly_table/2017-02-01
hdfs://emr-header-1.cluster-50466:9000/data/incoming/hourly_table/2017-02-02
```

### --groupBy和-targetSize

因为Hadoop可以从HDFS中读取少量的大文件,而不再读取大量的小文件,所以在大量小文件的场景下,您可以使用Jindo Dist Cp将小文件聚合为指定大小的大文件,以便于优化分析性能和降低成本。

例如,执行以下命令,查看如下文件夹中的数据。

hdfs dfs -ls /data/incoming/hourly\_table/2017-02-01/03

返回信息如下。

\_ . . . .

Found & Items	5						
-rw-r	2	root	hadoop	2252	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/000151.sst
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/1.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/2.log
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/5.log
-rw-r	2	root	hadoop	4891	2020-04-17	21:08	/data/incoming/hourly_table/2017-02-01/03/6.log
-rw-r	2	root	hadoop	4891	2020-04-17	20:42	/data/incoming/hourly_table/2017-02-01/03/OPTIONS-0001
09							
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp01.txt
-rw-r	2	root	hadoop	1016	2020-04-17	20:47	/data/incoming/hourly_table/2017-02-01/03/emp06.txt

#### 执行以下命令,将如下文件夹中的TXT文件合并为不超过10M的文件。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --targetSize=10 --g
roupBy='.*/([a-z]+).*.txt' --parallelism 20
```

#### 经过合并后,可以看到两个TXT文件被合并成了一个文件。

```
hdfs dfs -ls oss://<yourBucketName>/hourly_table/2017-02-01/03/
Found 1 items
-rw-rw-rw- 1 2032 2020-04-17 21:18 oss://<yourBucketName>/hourly_table/2017-02-01/03/emp2
```

### --enableBalancePlan

在您要拷贝的数据大小均衡、小文件和大文件混合的场景下,因为DistCp默认的执行计划是随机进行文件分配的,所以您可以指定 --enableBalancePlan 来更改Jindo DistCp的作业分配计划,以达到更好的DistCp性能。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --enableBalancePlan
--parallelism 20

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 同时使用。

## --enableDynamicPlan

当您要拷贝的数据大小分化严重、小文件数据较多的场景下,您可以指定 --enableDynamicPlan 来更改Jindo Dist Cp的作业分 配计划,以达到更好的Dist Cp性能。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableDynamicPlan
--parallelism 20
```

⑦ 说明 该参数不支持和 --groupby 或 --targetSize 参数一起使用。

## --enableTransaction

--enableTransaction 可以保证Job级别的完整性以及保证Job之间的事务支持。示例如下。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --enableTransaction
--parallelism 20
```

#### --diff

Dist Cp任务完成后,您可以使用 --diff 查看当前Dist Cp的文件差异。

例如,执行以下命令,查看/data/incoming/。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --diff

如果全部任务完成则会提示如下信息。

INFO distcp.JindoDistCp: distcp has been done completely

如果src的文件未能同步到dest上,则会在当前目录下生成*manifest*文件,您可以使用 --copyFromManifest 和 -previousManifest 拷贝剩余文件,从而完成数据大小和文件个数的校验。如果您的Dist Cp任务包含压缩或者解压缩,则 -diff 不能显示正确的文件差异,因为压缩或者解压缩会改变文件的大小。

```
jindo distcp --src /data/incoming/hourly_table --dest oss://<yourBucketName>/hourly_table --dest oss://<yourB
ucketName>/hourly_table --previousManifest=file:///opt/manifest-2020-04-17.gz --copyFromManifest --parallelis
m 20
```

② **说明** 如果您的 --dest 为HDFS路径,目前仅支持/path、hdfs://hostname:ip/path和hdfs://headerlp:ip/path的 写法,暂不支持hdfs:///path、hdfs:/path和其他自定义写法。

## 查看Distcp Counters

执行以下命令,在MapReduce的Counter信息中查找Distcp Counters的信息。

```
Distop Counters

Bytes Destination Copied=11010048000

Bytes Source Read=11010048000

Files Copied=1001

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_REDUCE=0
```

⑦ 说明 如果您的DistCp操作中包含压缩或者解压缩文件,则 Bytes Destination Copied 和 Bytes Source Read 的 大小可能不相等。

## 使用OSS AccessKey

在E-MapReduce外或者免密服务出现问题的情况下,您可以通过指定AccessKey来获得访问OSS的权限。您可以在命令中使用--ossKey、--ossSecret、--ossEndPoint选项来指定AccessKey。

命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --ossKey <yourAcces sKeyId> --ossSecret <yourAccessKeySecret> --ossEndPoint oss-cn-hangzhou.aliyuncs.com --parallelism 20

本文示例中的yourAccessKeyId是您阿里云账号的AccessKey ID, yourAccessKeySecret是您阿里云账号的AccessKey Secret。

## 使用归档或低频写入OSS

在您的Dist cp任务写入OSS时,您可以通过如下模式写入OSS,数据存储:

● 使用归档( --archive )示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy archive --parallelism 20

● 使用低频 ( --ia ) 示例命令如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --policy ia --par allelism 20

清理残留文件

在您的Dist Cp任务过程中,由于某种原因在您的目标目录下,产生未正确上传的文件,这部分文件通过uploadld的方式由OSS管理,并且对用户不可见时,您可以通过指定--cleanUpPending选项,指定任务结束时清理残留文件,或者您也可以通过OSS控制 台进行清理。

## 命令示例如下。

jindo distcp --src /data/incoming/hourly\_table --dest oss://<yourBucketName>/hourly\_table --cleanUpPending -parallelism 20

# 13.SmartData 2.2.x及之前版本 13.1.SmartData使用说明(EMR-3.20.0~3.22.0 版本)

本文主要介绍JindoFS的配置使用方式,以及一些典型的应用场景。

## 概述

JindoFS是一种云原生的文件系统,结合OSS和本地存储,成为E-MapReduce产品的新一代存储系统,为上层计算提供了高效可 靠的存储。

JindoFS提供了块存储模式(Block)和缓存模式(Cache)的存储模式。

JindoFS 采用了本地存储和OSS的异构多备份机制,Storage Service提供了数据存储能力,首先使用OSS作为存储后端,保证数据的高可靠性,同时利用本地存储实现冗余备份,利用本地的备份,可以加速数据读取;另外,JindoFS 的元数据通过本地服务 Namespace Service管理,从而保证了元数据操作的性能(和HDFS元数据操作性能相似)。

? 说明

- E-MapReduce-3.20.0及以上版本支持Jindo FS,您可以在创建集群时勾选相关服务来使用JindoFS。
- 本文主要是E-MapReduce-3.20.0及以上版本至E-MapReduce-3.22.0(但不包括)版本的介绍; E-MapReduce-3.22.0及以上版本的JindoFS使用说明,请参见SmartData使用说明(EMR-3.22.0~3.25.1版本)。



## 应用场景

E-MapReduce目前提供了三种大数据存储系统,E-MapReduce OssFileSystem、E-MapReduce HDFS和E-MapReduce JindoFS, 其中OssFileSystem和JindoFS都是云上存储的解决方案,下表为这三种存储系统和开源OSS各自的特点。

特点	开源OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
存储空间	海量	海量	取决于集群规模	海量
可靠性	高	高	高	高

特点	开源OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
吞吐率因素	服务端	集群内磁盘缓存	集群内磁盘	集群内磁盘
元数据效率	慢	<b>ф</b>	快	快
扩容操作	容易	容易	容易	容易
缩容操作	容易	容易	需Decommission	容易
数据本地化	无	ड्रेड्र	强	较强

JindoFS块存储模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

### 环境准备

• 创建集群

选择E-MapReduce-3.20.0及以上版本至E-MapReduce-3.22.0(但不包括)版本,勾选可选服务中 的SmartData和Bigboot,创建集群详情请参见创建集群。Bigboot 服务提供了E-MapReduce平台上的基础的分布式数据管 理交互服务以及一些组件管理监控和支持性服务,SmartData服务基于Bigboot之上对应用层提供了JindoFS文件系统。

1 \$	次件配置	2 硬件配置		3 基础配置
₩ 软件配置	集群类型: Hadoop	Kafka ZooKeeper Data Science	Druid	
	(C)	rz 🖗 🕜		
	开源大数	居离线、实时、Ad-hoc查询场景		
	Hadoop是	宅全使用开源Hadoop生态,采用YARN管理集	]群资源,提供Hive、Spark离线大规模分布式	、数据存储和计算,SparkStreaming、
	Flink, Stor	m流式数据计算,Presto、Impala交互式查询	l,Oozie、Pig等Hadoop生态圈的组件,支持	OSS存储,支持Kerberos用户认证和数据
	加密。			
	产品版本: EMR-3.20.	0 ~		
	必选服务: HDFS (2.8	.5) YARN (2.8.5) Hive (3.1.1) Sp	oark (2.4.2) Knox (1.1.0) Zeppelin (0.	8.1) Tez (0.9.1)
	ApacheD	(2.0.0) Ganglia (3.7.2) Pig (0.14.0)	Sqoop (1.4.7) Hue (4.1.0)	
	可选服务: HBase (1.	4.9) ZooKeeper (3.4.13) Presto (0.2)	13) Impala (2.12.2) Flume (1.8.0)	Livy (0.6.0) Superset (0.28.1)
	Ranger (1	.2.0) Flink (1.7.2) Storm (1.2.2)	Phoenix (4.14.1) SmartData (1.0.0	Bigboot (1.0.0
	请点击选择			

#### ● 配置集群

Smart Dat a提供的JindoFS文件系统使用OSS作为存储后端,因此在使用JindoFS之前需配置一些OSS相关参数。下面提供两种配置方式,第一种是先创建好集群,修改Bigboot相关参数,需重启Smart Dat a服务生效;第二种是创建集群过程中添加自定义配置,这样集群创建好后相关服务就能按照自定义参数启动。

#### 。 集群创建好后参数初始化

- oss.access.bucket 为OSS bucket的名称。
- oss.data-dir 为JindoFS在OSS bucket中所使用的目录(注:该目录为 JindoFS后端存储目录,生成的数据不能人为破坏,并且保证该目录仅用于JindoFS后端存储, JindoFS在写入数据时会自动创建用户所配置的目录,无需在OSS上事先创建)。
- oss.access.endpoint 为bucket所在的区域。
- oss.access.key 为存储后端OSS的AccessKey ID。
- oss.access.secret 为存储后端OSS的AccessKey Secret。

考虑到性能和稳定性,推荐使用同region下的OSS bucket作为存储后端,此时,E-MapReduce集群能够免密访问OSS,无 需配置AccessKey ID和AccessKey Secret。

所有JindoFS相关配置都在Bigboot组件中,配置如下图所示,红框中为必填的配置项。

<返回 正常 BIGBOOT ▼		
状态 部署拓扑 配置 配置修改历史		
快速配置	服务配置 集群默认配置 ~	
基础配置 高级配置 只读配置 数据路径 日志路径	bigboot	
日志相关         JVM相关         数据相关         数据库相关         性能相关           时间相关         编解码相关         OSS相关         地址端口         内存配置	oss.access.secret	~ 0
磁盘相关 网络相关 文件路径 URL或URI	node.data-dirs.watermark.low.ratio 0.3	0
配置文件:	oss.data-dir jindoFS-1	0
bigboot	oss.access.endpoint	0
	node.data-dirs.watermark.high.ratio 0.6	0
	max.blockletBuffer.group.num 16	0
	oss.access.key	0
	oss.access.bucket	0

⑦ 说明 JindoFS支持多命名空间,本文命名空间以test为例。

#### 配置完成后保存并部署,然后在SmartData服务中重启所有组件,即开始使用JindoFS。

	正常	55 Ganglia	⇒	cpu_system(%)	۸		
	正常	🞝 Spark	⇒	2.5% 2% 1.5%	hw		
	正常	Hue	⇒	1% 0.5% 0%			
	正常	🥏 Zeppelin	<b>₩</b> 配置	所有组件			
	TH	r01	①重启	所有组件	_		
	止常	lez	①重启	JindoFS Namespace Service	5-19 2		
	正常	Sqoop	① 重启	JindoFS Storage Service	• 平均		
	正堂	Pig	▶ 启动 所有组件				
			■ 停止	止所有组件			
	正常	SmartData	⇒	disk_partition_capacity_m	ax_usec		
	[正常]	Knox	<u>=&gt;</u>	30%			
。 创致	建集群时添加自定义酝	置					

E-MapReduce集群在创建集群时支持添加自定义配置,以同region下免密访问OSS为例,如下图勾选**软件自定义配置**,添加如下配置,配置 oss.data-dir 和 oss.access.bucket 。

```
[
    {
    "ServiceName":"BIGBOOT",
   "FileName":"bigboot",
   "ConfigKey":"oss.data-dir",
    "ConfigValue":"jindoFS-1"
    },
    {
    "ServiceName":"BIGBOOT",
    "FileName":"bigboot",
    "ConfigKey":"oss.access.bucket",
    "ConfigValue":"oss-bucket-name"
]
11 软件配置
                    集群类型:
                                        Druid
                                                Kafka
                                                        ZooKeeper
                                                                    Data science
                             开源大数据离线、实时、Ad-hoc查询场景
                             E-MapReduce Hadoop是完全使用开源Hadoop生态,采用YARN管理集群资源,提供Hive、Spark离线大规模分
                             布式数据存储和计算,SparkStreaming、Flink、Storm流式数据计算,Presto、Impala交互式查询,Oozie、Pig
                             等Hadoop生态圈的组件,支持OSS存储,支持Kerberos的数据认证与加密。
                    产品版本: EMR-3.20.0
                                                                    v
            资源管理类型: ?
                                       全托管
                    必洗服务:
                                          ApacheDS (2.0.0) Ganglia (3.7.2)
                    可选服务: HBase (1.4.9) ZooKeeper (3.4.13) Presto (0.213) Impala (2.12.2) Flume (1.8.0)
                             Livy (0.6.0) Superset (0.28.1) Ranger (1.2.0) Flink (1.7.2) Storm (1.2.2)
                             Phoenix (4.14.1) SmartData (1.0.0 Bigboot (1.0.0 Oozie (5.1.0)
                             请点击选择
   * 高级设置
              Kerberos集群模式: 👔 💿 高安全集群中的各组件会通过Kerberos进行认证,详细信息参考Kerberos简介 🗗
                软件自定义配置: ?
                                            新建集群创建前,可以通过json文件定义集群组件的参数配置,详细信息参考 软件配
                                    置で
                                    [{"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"oss.data-
dir","ConfigValue":"<u>lindoF</u>S"},
{"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"<u>oss.access.bucket</u>","Co
nfigValue":"<u>oss-bucket-name</u>"}]
```

## 使用JindoFS

JindoFS使用上与HDFS类似,提供ifs前缀,将ifs替代hdfs即可使用。简单示例:

hadoop fs -ls jfs:/// hadoop fs -mkdir jfs:///test-dirhadoop fs -put test.log jfs:///test-dir/

目前, JindoFS能够支持 E-MapReduce 集群上的 Hadoop、Hive、Spark的作业进行访问,其余组件尚未完全支持。

#### 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了node.data-dirs.watermark.high.ratio和node.data-dirs.watermark.low.ratio这两个参数用来调节本地存储的使用容量, 值均为0~1的小数表示使用比例,JindoFS默认使用所有数据盘,每块盘的使用容量默认即为数据盘大小。前者表示使用量上水 位比例,每块数据盘的JindoFS占用的空间到达上水位即会开始清理淘汰;后者表示使用量下水位比例,触发清理后会将JindoFS 的占用空间清理到下水位。用户可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理 的值即可。

## 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略适应不同的存储需求,可以对目录设置以下四种存储策略。

策略	策略说明
COLD	表示数据仅在OSS上有一个备份,没有本地备份,适用于冷数据存储。
WARM	默认策略。 表示数据在OSS和本地分别有一个备份, 本地备份能够有效的提供后续的读取加速。
НОТ	表示数据在OSS上有一个备份,本地有多个备份,针对一些最热的数据提供更进一步的加速效果。
ТЕМР	表示数据仅有一个本地备份,针对一些临时性数据,提供高性能的读写,但降低了数据的高可靠性,适用 于一些临时数据的存取。

JindoFS提供了Admin工具设置目录的Storage Policy(默认为 WARM),新增的文件将会以父目录所指定的Storage Policy进行 存储,使用方式如下所示。

jindo dfsadmin -R -setStoragePolicy [path] [policy]

#### 通过以下命令,获取某个目录的存储策略。

jindo dfsadmin -getStoragePolicy [path]

⑦ 说明 其中[path]为设置policy的路径名称, -R表示递归设置该路径下的所有路径。

#### Admin工具还提供archive命令,实现对冷数据的归档。

此命令提供了一种用户显式淘汰本地数据块的方式。Hive分区表按天分区,假如业务上对一周前的分区数据认为不会再经常访问,那么就可以定期将一周前的分区目录执行archive,淘汰本地备份,文件备份将仅仅保留在后端OSS上。

Archive命令的使用方式如下:

jindo dfsadmin -archive [path]

⑦ 说明 [path]为需要归档文件的所在目录路径。

## 13.2. SmartData使用说明(EMR-3.22.0~3.25.1 版本)

JindoFS是一种云原生的文件系统,结合OSS和本地存储,成为E-MapReduce产品的新一代存储系统,为上层计算提供了高效可 靠的存储。本文主要说明JindoFS的配置使用方式,以及介绍一些典型的应用场景。

## 概述

JindoFS提供了块存储模式(Block)和缓存模式(Cache)的存储模式。

JindoFS采用了本地存储和OSS的异构多备份机制,Storage Service提供了数据存储能力,首先使用OSS作为存储后端,保证数据 的高可靠性,同时利用本地存储实现冗余备份,利用本地的备份,可以加速数据读取;另外,JindoFS的元数据通过本地服务 Namespace Service管理,从而保证了元数据操作的性能(和HDFS元数据操作性能相似)。

### ? 说明

- E-MapReduce-3.20.0及以上版本支持Jindo FS,您可以在创建集群时勾选相关服务来使用JindoFS。
- 本文主要是E-MapReduce-3.22.0及以上版本的介绍; E-MapReduce-3.20.0及以上版本至E-MapReduce-3.22.0(但不包括)版本的JindoFS使用说明,请参见Smart Data使用说明(EMR-3.20.0~3.22.0版本)。



## 环境准备

创建集群

选择E-MapReduce-3.22.0及以上版本,勾选可选服务中的SmartData,创建集群详情请参见创建集群。

1 4	次件配置	2 硬件配置	3 基础配置
🗰 软件配置	集群类型: Hadoop	Kafka ZooKeeper Data Science Druid	
	开源大数据 Hadoop是完全	<ul> <li></li></ul>	Hive、Spark离线大规模分布式数据存储和计算,SparkStreaming、Flink、
	Storm流式数据 产品版本: EMR-3.22.1	計算,Presto、Impala交互式查询,Oozie、Pig等Hadoo	op生态圈的组件,支持OSS存储,支持Kerberos用户认证和数据加密。
	必选服务: HDFS (2.8.5) Pig (0.14.0)	YARN (2.8.5)         Hive (3.1.1)         Spark (2.4.3)           Sqoop (1.4.7)         Bigboot (2.0.1)         OpenLDAP (2	Knox (1.1.0) Zeppelin (0.8.1) Tez (0.9.1) Ganglia (3.7.2) 4.44) Hue (4.4.0)
	可选服务: HBase (1.4.9) Ranger (1.2.0 Oozie (5.1.0) 请点击选择	) ZooKeeper (3.5.5) Presto (0.221) Impala ( )) Flink (1.7.2) Storm (1.2.2) Phoenix (4.14.1	2.12.2)       Flume (1.8.0)       Livy (0.6.0)       Superset (0.28.1)         )       Analytics Zoo (0.5.0)       SmartData (2.0.0)       Kudu (1.10.0)

● 配置集群

Smart Dat a提供的JindoFS文件系统使用OSS作为存储后端,因此在使用JindoFS之前需配置一些OSS相关参数。下面提供两种配置方式,第一种是先创建好集群,修改Bigboot相关参数,需重启Smart Dat a服务生效;第二种是创建集群过程中添加自定义配置,这样集群创建好后相关服务就能按照自定义参数启动:

○ 集群创建好后初始化参数

所有JindoFS相关配置都在Bigboot组件中,配置如下所示:

a. 在服务配置页面,单击bigboot页签。

服务配置	0	部署客户端配置保存
全部 bigboot		自定义配置
storage.data-dirs.watermark.low.ratio	0.3	0
jfs.namespaces	test	5 🕐
storage.data-dirs.watermark.high.ratio	0.6	0

## b. 单击**自定义配置**。

Кеу	* Value	描述	操作
jfs.namespaces.test.uri	oss://oss-bucket/oss-dir		删除
jfs.namespaces.test.mode	block		删除
jfs.namespaces.test.oss.acc	XXXX		删除
jfs.namespaces.test.oss.acc	XXXX		删除
添加			

## ? 说明

- 红框中为必填的配置项。
- JindoFS支持多命名空间,本文命名空间以test为例。

参数	参数说明	示例
jfs.namespaces	表示当前JindoFS支持的命名空间,多个命 名空间时以逗号隔开。	test
		oss://oss-bucket/oss-dir
jfs.namespaces.test.uri	表示test命名空间的后端存储。	⑦ 说明 该配置也可以配置到OSS bucket下的具体目录,该命名空间即 以该目录作为根目录来读写数据。
		block
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	② 说明 JindoFS支持block和 cache两种存储模式。
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。	? 说明 考虑到性能和稳定性, 推荐使用同账户、同region下的OSS bucket作为存储后端,此时,E- MapReduce集群能够免密访问OSS, 无需配置AccessKey ID和AccessKey Secret。

正常	5 <sup>5</sup> Ganglia	⇒	cpu_system(%)
正常	🞝 Spark	⇒	2.5% 2% 1.5%
正常	Hue	⇒	1% - 0.5% - 0%
正常	🥭 Zeppelin	↓fî 配置	<b>1</b> 所有组件
正常	Tez	① 重启 ① 重启	音所有组件 目 JindoFS Namespace Service 5-19 2
正常	Sqoop	① 重启	l JindoFS Storage Service
正常	○ Pig	▶ 启云	力所有组件
正常	SmartData	⇒	disk_partition_capacity_max_used
正常	Knox	<u>=&gt;</u>	30%

配置完成后保存并部署,然后在SmartData服务中重启所有组件,即开始使用JindoFS。

## ◦ 创建集群时添加自定义配置

E-MapReduce集群在创建集群时支持添加自定义配置,以同region下免密访问OSS为例,如下图勾选**软件自定义配置**,配置命名空间test的相关配置,详情如下。

<pre>[ {    "ServiceName":"BIG    "FileName":"bigboo    "ConfigKey":"jfs.n    },{    "ServiceName":"BIG    "FileName":"bigboo    "ConfigKey":"jfs.n    "ConfigValue":"oss    },{    "ServiceName":"BIG    "FileName":"bigboo    "ConfigKey":"jfs.n</pre>	BOOT", t", amespaces","ConfigValue":"test" BOOT", t", amespaces.test.uri", ://oss-bucket/oss-dir" BOOT", t", amespaces.test.mode", ck"
}	
]	
<b>軟件配置</b> 集群	Hadoop Kafka ZooKeeper Data Science Druid
	开源大数据离线、实时、Ad-hoc查询场景
	Hadoop是完全使用开源Hadoop生态,采用YARN管理集群资源,提供Hive、Spark离线大规模分布式数据存储和计算,SparkStreaming、 Flink Storm运式数据计算 Dracto Impala成石式查询 Opria Dio英山adoot生太网的说什,支持OSS存储,支持Karbaror田白门近和数据
	Films, Stormunizaedan)异,Fresto, Impana文王之直间,Oozie、Fig等Fradoop王改國的近代,文は4055分別後,文は40eros内アル近代40da 加密。
产品	温版本: EMR-3.22.1 ~
1247	田奈- 山口EC /2 & S1 VADN /2 & S1 Uius (2.1.1) Seart (2.4.2) Keav (1.1.0) Zamalia (0.0.1) Tax (0.0.1) Gamalia (2.7.2)
202	Pig (0.14.0)         Sqoop (1.4.7)         Bigboot (2.0.1)         OpenLDAP (2.4.44)         Hue (4.4.0)
可送	服务: [HBase (1.4.9) ] ZooKeeper (3.5.5) [Presto (0.221)] [Impala (2.12.2)] [Flume (1.8.0)] [Livy (0.6.0)] [Superset (0.28.1)]
	Ranger (1.2.0)         Flink (1.7.2)         Storm (1.2.2)         Phoenix (4.14.1)         Analytics Zoo (0.5.0)         SmartData (2.0.0)         Kudu (1.10.0)
	Oozie (5.1.0) 诸白市玩経
	HBUILLAN KUTA
× 高级设置	
Kerberos集	群模式: 😰 高安全集群中的各组件会通过Kerberos进行认证,详细信息参考Kerberos简介 🗗
软件自定	义配置: 2 新建集群创建前,可以通过json文件定义集群组件的参数配置,详细信息参考 软件配置 2
	[{"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"jfs.namespaces","ConfigValue":"test"}, {"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"jfs.namespaces.test.uri","ConfigValue":"oss://oss- bucket/oss-dir"), {"ServiceName":"BIGBOOT","FileName":"bigboot","ConfigKey":"jfs.namespaces.test.mode","ConfigValue":"block"}]

## 使用JindoFS

JindoFS使用上与HDFS类似,提供jfs前缀,将jfs替代hdfs即可使用。

目前,JindoFS能够支持EMR集群上的大部分计算组件,包括Hadoop、Hive、Spark、Flink、Presto和Impala。 简单示例:

• Shell命令

```
hadoop fs -ls jfs://your-namespace/
hadoop fs -mkdir jfs://your-namespace/test-dir
hadoop fs -put test.log jfs://your-namespace/test-dir/
hadoop fs -get jfs://your-namespace/test-dir/test.log ./
```

• MapReduce作业

hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.8.5.jar teragen -Dmap red.map.tasks=1000 10737418240 jfs://your-namespace/terasort/input hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.8.5.jar terasort -Dma pred.reduce.tasks=1000 jfs://your-namespace/terasort/input jfs://your-namespace/terasort/output

• Spark-SQL

CREATE EXTERNAL TABLE IF NOT EXISTS src\_jfs (key INT, value STRING) location 'jfs://your-namespace/Spark\_sq l\_test/';

## 磁盘空间水位控制

JindoFS后端基于OSS,可以提供海量的存储,但是本地盘的容量是有限的,因此JindoFS会自动淘汰本地较冷的数据备份。我们 提供了node.data-dirs.watermark.high.ratio和node.data-dirs.watermark.low.ratio这两个参数用来调节本地存储的使用容量, 值均为0~1的小数表示使用比例,JindoFS默认使用所有数据盘,每块盘的使用容量默认即为数据盘大小。前者表示使用量上水 位比例,每块数据盘的JindoFS占用的空间到达上水位即会开始清理淘汰;后者表示使用量下水位比例,触发清理后会将JindoFS 的占用空间清理到下水位。用户可以通过设置上水位比例调节期望分给JindoFS的磁盘空间,下水位必须小于上水位,设置合理 的值即可。

### 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略适应不同的存储需求,可以对目录设置以下四种存储策略。

策略	策略说明
COLD	表示数据仅在OSS上有一个备份,没有本地备份,适用于冷数据存储。
WARM	默认策略。 表示数据在OSS和本地分别有一个备份, 本地备份能够有效的提供后续的读取加速。
НОТ	表示数据在OSS上有一个备份,本地有多个备份,针对一些最热的数据提供更进一步的加速效果。
TEMP	表示数据仅有一个本地备份,针对一些临时性数据,提供高性能的读写,但降低了数据的高可靠性,适用 于一些临时数据的存取。

JindoFS提供了Admin工具设置目录的Storage Policy(默认为 WARM),新增的文件将会以父目录所指定的Storage Policy进行存储,使用方式如下。

jindo dfsadmin -R -setStoragePolicy [path] [policy]

#### 通过以下命令,获取某个目录的存储策略:

jindo dfsadmin -getStoragePolicy [path]

⑦ 说明 其中[path]为设置policy的路径名称, -R表示递归设置该路径下的所有路径。

## Admin工具

JindoFS提供了Admin工具的archive和jindo命令。

• Admin工具提供archive命令,实现对冷数据的归档。

此命令提供了一种用户显式淘汰本地数据块的方式。Hive分区表按天分区,假如业务上对一周前的分区数据认为不会再经常访问,那么就可以定期将一周前的分区目录执行archive,淘汰本地备份,文件备份将仅仅保留在后端OSS上。

Archive命令的使用方式如下。

jindo dfsadmin -archive [path]

⑦ 说明 [path]为需要归档文件的所在目录路径。

• Admin工具提供jindo命令,为Namespace Service提供了一些管理员功能命令。

jindo dfsadmin [-options]

⑦ 说明 可以通过 jindo dfsadmin --help 命令获取帮助信息。

Admin工具对Cache模式提供了diff和sync命令。

• diff命令主要用来显示本地数据与后端存储系统数据之间的差异。

jindo dfsadmin -R -diff [path]

⑦ 说明 默认情况下比较 [path] 目录的子目录中元数据之间的差异, -R 选项表示递归比较 [path] 目录下所有的路径。

• sync命令用于同步本地与后端存储之前的元数据。

jindo dfsadmin -R -sync [path]

⑦ 说明 [path] 表示需要同步元数据的路径,默认只会同步 [path] 的下一级目录, -R 选项表示递归比较 [path] 目录下所有的路径。

## 13.3. JindoFS块存储模式

本文主要介绍JindoFS的块存储模式(Block),以及一些典型的应用场景。

#### 概念

块存储模式提供了最为高效的数据读写能力和元数据访问能力,并且能够支持更加全面的Hadoop文件系统语义。同时,JindoFS 也提供了外部客户端,能够从集群外部访问建立在E-MapReduce集群内的JindoFS文件系统。

数据以Block形式存储在后端存储OSS上,本地Namespace服务维护元数据信息,该模式在性能上较优,无论是数据性能还是元 数据性能。

## 应用场景

E-MapReduce目前提供了三种大数据存储系统, E-MapReduce OssFileSystem、E-MapReduce HDFS和E-MapReduce JindoFS, 其中OssFileSystem和JindoFS都是云上存储的解决方案, 下表为这三种存储系统和开源OSS各自的特点。

特点	开源OSS	E-MapReduce OssFileSystem	E-MapReduce HDFS	E-MapReduce JindoFS
存储空间	海量	海量	取决于集群规模	海量
可靠性	高	高	高	高
吞吐率因素	服务端	集群内磁盘缓存	集群内磁盘	集群内磁盘
元数据效率	慢	<b>ф</b>	快	快
扩容操作	容易	容易	容易	容易
缩容操作	容易	容易	需Decommission	容易
数据本地化	无	弱	强	较强

JindoFS块存储模式具有以下几个特点:

- 海量弹性的存储空间,基于OSS作为存储后端,存储不受限于本地集群,而且本地集群能够自由弹性伸缩。
- 能够利用本地集群的存储资源加速数据读取,适合具有一定本地存储能力的集群,能够利用有限的本地存储提升吞吐率,特别 对于一写多读的场景效果显著。
- 元数据操作效率高,能够与HDFS相当,能够有效规避OSS文件系统元数据操作耗时以及高频访问下可能引发不稳定的问题。
- 能够最大限度保证执行作业时的数据本地化,减少网络传输的压力,进一步提升读取性能。

### 配置集群

所有JindoFS相关配置都在Bigboot组件中, 配置如下图所示。

修改配置项

务配置			0	部署客户端配置
è部 bigboot				自定义
storage.d	ata-dirs.watermark.low.ratio	0.3		0
	jfs.namespaces	test		5 0
storage.da	ta-dirs.watermark.high.ratio	0.6		0
配置项				
冒配置项			×	-
* Key	* Value	描述	操作	
jfs.namespaces.test.uri	oss://oss-bucket/oss-dir	r	删除	
jfs.namespaces.test.mode	block		删除	
jfs.namespaces.test.oss.acc	XXXX		删除	
jfs.namespaces.test.oss.acc	XXXX		删除	
添加				
			确定取消	
			确定取消	

## ? 说明

- 红框中为必填的配置项。
- JindoFS支持多命名空间,本文命名空间以test为例。

参数	参数说明	示例
jfs.namespaces	表示当前JindoFS支持的命名空间,多个命名 空间时以逗号隔开。	test

参数	参数说明	示例
		oss://oss-bucket/oss-dir
jfs.namespaces.test.uri	表示test命名空间的后端存储。	⑦ 说明 该配置也可以配置到OSS bucket下的具体目录,该命名空间即以 该目录作为根目录来读写数据。
jfs.namespaces.test.mode	表示test命名空间为块存储模式。	block
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。	<ul> <li>说明 考虑到性能和稳定性,推荐使用同账户、同region下的OSS bucket作为存储后端,此时,E- MapReduce集群能够免密访问OSS,无 需配置AccessKey ID和AccessKey Secret。</li> </ul>

配置完成后保存并部署,然后在SmartData服务中重启Namespace Service,即可开始使用JindoFS。

₩ 配置 所有组件
③ 重启 所有组件
① 重启 JindoFS Namespace Service
③ 重启 JindoFS Storage Service
▶ 启动 所有组件
■ 停止 所有组件

## 存储策略

JindoFS提供了Storage Policy功能,提供更加灵活的存储策略适应不同的存储需求,可以对目录设置以下四种存储策略。

策略	策略说明
COLD	表示数据仅在OSS上有一个备份,没有本地备份,适用于冷数据存储。
WARM	默认策略。 表示数据在OSS和本地分别有一个备份, 本地备份能够有效的提供后续的读取加速。
НОТ	表示数据在OSS上有一个备份,本地有多个备份,针对一些最热的数据提供更进一步的加速效果。
TEMP	表示数据仅有一个本地备份,针对一些临时性数据,提供高性能的读写,但降低了数据的高可靠性,适用 于一些临时数据的存取。

JindoFS提供了Admin工具设置目录的Storage Policy(默认为 WARM),新增的文件将会以父目录所指定的Storage Policy进行存储,使用方式如下所示。

jindo dfsadmin -R -setStoragePolicy [path] [policy]

## 通过以下命令,获取某个目录的存储策略。

jindo dfsadmin -getStoragePolicy [path]

⑦ 说明 其中[path]为设置policy的路径名称, - R表示递归设置该路径下的所有路径。

Admin工具还提供archive命令,实现对冷数据的归档。

此命令提供了一种用户显式淘汰本地数据块的方式。Hive分区表按天分区,假如业务上对一周前的分区数据认为不会再经常访问,那么就可以定期将一周前的分区目录执行archive,淘汰本地备份,文件备份将仅仅保留在后端OSS上。

Archive命令的使用方式如下:

```
jindo dfsadmin -archive [path]
```

⑦ 说明 [path]为需要归档文件的所在目录路径。

# 13.4. JindoFS缓存模式

本文主要介绍JindoFS的缓存模式(Cache),以及一些典型的应用场景。

## 概述

缓存模式兼容现有OSS存储方式,文件以对象的形式存储在OSS上,每个文件根据实际访问情况会在本地进行数据和元数据的缓存,从而提高访问数据以及元数据的性能,Cache模式提供不同元数据同步策略以满足您在不同场景下的需求。

#### 应用场景

缓存模式最大的特点就是兼容性,保持了OSS原有的对象语义,集群中仅做缓存,因此JindoFS和OSS客户端、OssFileSystem 等,或者其他的各种OSS的交互程序是完全兼容的,对原有OSS上的存量数据也不需要做任何的迁移、转换工作即可使用。同时 集群中的数据和元数据缓存也能一定程度上提升数据访问性能。

## 配置集群

所有JindoFS相关配置都在Bigboot组件中,配置如下图所示。

#### 修改配置项

服务配置			?	部署客户端配置保存
全部 bigboot				自定义配置
	storage.data-dirs.watermark.low.ratio	0.3		0
	jfs.namespaces	test		5 🕜
	storage.data-dirs.watermark.high.ratio	0.6		0

新增配置项

' Key	* Value	描述	操作
jfs.namespaces.test.uri	oss://oss-bucket/oss-dir		删除
jfs.namespaces.test.mode	block		删除
jfs.namespaces.test.oss.acc	XXXX		删除
jfs.namespaces.test.oss.acc	XXXX		删除

## ? 说明

- 红框中为必填的配置项。
- JindoFS支持多命名空间,本文命名空间以test为例。

参数	参数说明	示例
jfs.namespaces	表示当前JindoFS支持的命名空间,多个命名 空间时以逗号隔开。	test
jfs.namespaces.test.uri	表示test命名空间的后端存储。	oss://oss-bucket/ <b>? 说明</b> 该配置也可以配置到OSS bucket下的具体目录,该命名空间即以 该目录作为根目录来读写数据,但一般 情况下配置bucket即可,这样路径就和 原生OSS保持一致。
jfs.namespaces.test.mode	表示test命名空间为缓存模式。	cache
jfs.namespaces.test.oss.access.key	表示存储后端OSS的AccessKey ID。	XXXX
jfs.namespaces.test.oss.access.secret	表示存储后端OSS的AccessKey Secret。	⑦ 说明 考虑到性能和稳定性,推荐使用同账户、同region下的OSS bucket作为存储后端,此时,E- MapReduce集群能够免密访问OSS,无 需配置AccessKey ID和AccessKey Secret。

配置完成后保存并部署,然后在SmartData服务中重启Namespace Service,即可开始使用JindoFS。

₩ 配置 所有组件
③ 重启 所有组件
① 重启 JindoFS Namespace Service
① 重启 JindoFS Storage Service
▶ 启动 所有组件
■ 停止 所有组件

## 元数据同步策略

缓存模式下可能存在JindoFS集群构建之前,您已经在OSS上保存了大量数据的场景,对于这种场景,后续的数据访问会同步数据 和元数据到JindoFS集群,数据同步策略为了访问数据都会在本地保留一份;元数据同步策略分为两部分,包括元数据同步间隔 策略和元数据load策略:

• 元数据同步间隔策略:

配置参数为namespace.sync.interval, 该参数默认值为-1, 表示不会同步OSS上的元数据。

- 当namespace.sync.interval=0时,表示每次操作都会同步OSS上的元数据。
- 当namespace.sync.interval>0时,表示会以固定的时间间隔来同步OSS上的元数据。

⑦ 说明 例如当namespace.sync.interval=5时,表示每隔5秒会去同步OSS上的元数据。

• 元数据Load策略:

配置参数为namespace.sync.loadtype,该配置参数为枚举类型{never,once,always},never表示从不同步OSS上的元数据;once为默认配置,表示只从OSS同步一次元数据;always表示每次操作都会同步OSS上的元数据。

⑦ 说明 当不配置namespace.sync.interval参数时,才会去使用Load策略;如果已配置namespace.sync.interval参数,则Load策略配置不生效。

## 13.5. 使用JindoFS SDK免密功能

本文介绍使用JindoFS SDK时, E-MapReduce(简称EMR)集群外如何以免密方式访问E-MapReduce JindoFS的文件系统。

#### 前提条件

适用环境: ECS(EMR环境外)+Hadoop+JavaSDK。

## 背景信息

使用JindoFS SDK时,需要把环境中相关Jindo的包从环境中移除,如*jboot.jar、smart dat a-aliyun-jf s-\*,jar*。如果要使用Spark则 需要把*/opt/apps/spark-current /jars/*里面的包也删除,从而可以正常使用。

## 步骤一: 创建实例RAM角色

- 1. 使用云账号登录RAM的控制台。
- 2. 单击左侧导航栏的RAM角色管理。
- 3. 单击创建 RAM 角色,选择当前可信实体类型为阿里云服务。
- 4. 单击下一步。
- 5. 输入角色名称,从选择授信服务列表中,选择云服务器。
- 6. 单击完成。

## 步骤二:为RAM角色授予权限

- 1. 使用云账号登录RAM的控制台。
- 2. (可选)如果您不使用系统权限,可以参见账号访问控制创建自定义权限策略章节创建一个自定义策略。
- 3. 单击左侧导航栏的RAM角色管理。

- 4. 单击新创建RAM角色名称所在行的精确授权。
- 5. 选择权限类型为系统策略或自定义策略。
- 6. 输入策略名称。
- 7. 单击**确定**。

## 步骤三:为实例授予RAM角色

- 1. 登录ECS管理控制台。
- 2. 在左侧导航栏,单击实例与镜像 > 实例。
- 3. 在顶部状态栏左上角处,选择地域。
- 4. 找到要操作的ECS实例,选择更多 > 实例设置 > 授予/收回RAM角色。



5. 在弹窗中,选择创建好的实例RAM角色,单击确定完成授予。

## 步骤四:在ECS上设置环境变量

执行如下命令,在ECS上设置环境变量。

export CLASSPATH=/xx/xx/jindofs-2.5.0-sdk.jar

#### 或者执行如下命令。

HADOOP\_CLASSPATH=\$HADOOP\_CLASSPATH:/xx/xx/jindofs-2.5.0-sdk.jar

## 步骤五:测试免密方式访问的方法

1. 使用Shell访问OSS。

hdfs dfs -ls/-mkdir/-put/..... oss://<ossPath>

使用Hadoop FileSystem访问OSS。
 JindoFS SDK支持使用Hadoop FileSystem访问OSS,示例代码如下。

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSvstem;
import org.apache.hadoop.fs.LocatedFileStatus;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.fs.RemoteIterator;
import java.net.URI;
public class test {
   public static void main(String[] args) throws Exception {
        FileSystem fs = FileSystem.get(new URI("ossPath"), new Configuration());
       RemoteIterator<LocatedFileStatus> iterator = fs.listFiles(new Path("ossPath"), false);
       while (iterator hasNext()) {
           LocatedFileStatus fileStatus = iterator.next():
           Path fullPath = fileStatus.getPath();
           System.out.println(fullPath);
       }
   }
}
```

## 13.6. JindoFS外部客户端

本文主要介绍JindoFS的外部客户端,以及一些典型的应用场景。

## 概述

JindoFS外部客户端,主要是为E-MapReduce集群外部访问JindoFS集群提供一种可行的方法。现在JindoFS外部客户端只能访问块存储模式下的JindoFS,不支持访问缓存模式下的JindoFS。实际上,缓存模式兼容OSS原始语义,因此外部访问仅需用普通OSS客户端即可。

### 应用场景

JindoFS外部客户端实现了Hadoop文件系统的接口,在用户程序跟E-MapReduce JindoFS Namespace服务网络相通的情况下, 用户可以通过JindoFS外部客户端去访问JindoFS上存储的数据, 但外部客户端不能利用E-MapReduce JindoFS的数据缓存能力, 相比E-MapReduce集群内部访问JindoFS集群,性能有所损失。

## 配置外部客户端

已配置JindoFS块存储模式的Namespace,详情请参见JindoFS块存储模式。

1. 获取Bigboot程序包。

在E-MapReduce集群内部/usr/lib/bigboot-current路径下,获取Bigboot程序包。

```
⑦ 说明 一般情况下,程序使用Native开发,若实际系统类型与E-MapReduce集群差别较大,相关的程序需重新编译,可以通过联系我们处理。
```

2. 配置环境。

设置环境变量BIGBOOT\_HOME为程序安装根目录,将程序根目录下*ext和lib*的路径,添加到用户使用的大数据组件(Hadoop或Spark等)的Classpath中。

- 3. 从E-MapReduce集群内部拷贝配置文件/usr/lib/bigboot-current/conf/bigboot.cfg.external,到用户客户机上对应的安装目录 conf/bigboot.cfg。
- 4. 配置Namespace Service。
  - o client.namespace.rpc.port : 配置JindoFS Namespace Service的监听端口。
  - client.namespace.rpc.address : 配置JindoFS Namespace Service的监听地址。

⑦ 说明 默认E-MapReduce集群中的配置文件已经配置好这两项。

```
5. 配置数据访问相关的配置项。
```

• client.namespaces.{YourNamespace}.oss.access.bucket :配置OSS bucket选项。
- client.namespaces.{YourNamespace}.oss.access.endpoint : 配置OSS endpoint选项。
- client.namespaces.{YourNamespace}.oss.access.key : 配置OSS的AccessKey ID。
- o client.namespaces.{YourNamespace}.oss.access.secret : 配置OSS的AccessKey Secret。

```
② 说明 其中 {YourNamespace} 为外部客户端要访问的Namespace的名称,本文Namespace的名称以test为例。
```

#### 配置示例如下。

```
client.namespace.rpc.port = 8101
client.namespace.rpc.address = {RPC_Address}
client.namespaces.test.oss.access.bucket = {YourOssBucket}
client.namespaces.test.oss.access.endpoint = {YourOssEndpoint}
client.namespaces.test.oss.access.key = {YourOssAccessKeyID}
client.namespaces.test.oss.access.secret = {YourOssAccessKeySecret}
```

# 配置验证

验证如下信息:

• 通过以下命令,验证Namespace是否正确。

hdfs dfs -ls jfs://test/

#### • 通过以下命令,验证数据是否可以上传或者下载。

hdfs dfs -put /etc/hosts jfs://test/ hdfs dfs -get jfs://test/hosts

# 14.最佳实践 14.1. 迁移Hadoop文件系统数据至JindoFS

本文以OSS为例,介绍如何将Hadoop文件系统上的数据迁移至JindoFS。

# 迁移数据

• Hadoop FsShell

对于文件较少或者数据量较小的场景,可以直接使用Hadoop的FsShell进行同步:

hadoop dfs -cp hdfs://emr-cluster/README.md jfs://emr-jfs/

hadoop dfs -cp oss://oss\_bucket/README.md jfs://emr-jfs/

• Dist Cp

对于文件较多或者数据量较大的场景,推荐使用Hadoop内置的Dist Cp进行同步:

hadoop distcp hdfs://emr-cluster/files jfs://emr-jfs/output/

hadoop distcp oss://oss\_bucket/files jfs://emr-jfs/output/

⑦ 说明 Dist Cp参数详情,请参见Dist Cp Version2 Guide。

# 利用JindoFS缓存模式

缓存模式是兼容现有OSS的存储方式,文件会以原生对象的形式存储在OSS上,同时OSS文件通过JindoFS缓存模式访问时,也有 机会在本地进行数据和元数据的缓存、加速访问,具体请参见JindoFS缓存模式。

# 14.2. 使用MapReduce处理JindoFS上的数据

本文介绍如何使用MapReduce读写JindoFS上的数据。

# JindoFS配置

以EMR-3.35版本为例,创建名为emr-jfs的命名空间,相关配置参数示例如下:

- jfs.namespaces=emr-jfs
- jfs.namespaces.emr-jfs.oss.uri=oss://oss-bucket/oss-dir
- jfs.namespaces.emr-jfs.mode=block

# MapReduce简介

Hadoop MapReduce作业通常是通过HDFS进行读写,JindoFS目前已兼容大部分HDFS接口,只需要将MapReduce作业的输入、 输出目录配置到JindoFS,即可实现读写JindoFS上的文件 。

Hadoop MapReduce是一个使用简易的软件框架,基于它写出来的应用程序能够运行在由上千个商用机器组成的大型集群上,并以一种可靠容错的方式并行处理上T级别的数据集。一个MapReduce作业通常会把输入的数据集切分为若干独立的数据块,由 Map任务以完全并行的方式处理它们。框架会对map的输出先进行排序,然后把结果输入给reduce任务。通常作业的输入和输 出都会被存储在文件系统中。整个框架负责任务的调度和监控,以及重新执行已经失败的任务。

# 作业的输入和输出

MapReduce作业通常会指明输入或输出的位置(路径),并通过实现合适的接口或抽象类提供map和reduce函数。Hadoop的 job client再加上其他作业的参数提交给ResourceManager,进行调度执行。这种情况下,我们直接修改作业的输入和输出目录 即可实现JindoFS的读写。

# MapReduce on JindoFS样例

以下是MapReduce作业通过修改输入输出实现JindoFS的读写的例子。

#### • Teragen数据生成样例

Teragen是Example中生成随机数据演示程序,在指定目录上生成指定行数的数据,具体命令如下:

hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-\*.jar teragen <num rows
> <output dir>

#### 替换输出路径,可以把数据输出到JindoFS上:

hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-\*.jar teragen 100000 jf s://emr-jfs/teragen data 0

#### • Terasort数据生成样例

#### Terasort是Example中数据排序演示样例,有输入和输出目录,具体命令如下:

hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-\*.jar terasort <in> <ou
t>

#### 替换输入和输出路径,即可处理lindoFS上的数据:

```
hadoop jar /usr/lib/hadoop-current/share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar terasort jfs://em
r-jfs/teragen_data_0/ jfs://emr-jfs/terasort_data_0
```

# 14.3. 使用Hive查询JindoFS上的数据

Apache Hive是Hadoop生态中广泛使用的SQL引擎之一,让用户可以使用SQL实现分布式的查询,Hive中数据主要以 undefinedDatabase、Table和Partition的形式进行管理,通过指定位置(Location)对应到后端的数据。

#### JindoFS配置

以EMR-3.35版本为例,创建名为emr-jfs的命名空间,相关配置参数示例如下:

- jfs.namespaces=emr-jfs
- jfs.namespaces.emr-jfs.oss.uri=oss://oss-bucket/oss-dir
- jfs.namespaces.emr-jfs.mode=block

# Warehouse、Database、Table或Partition的Location

• Warehouse的Location

Hive的hive-site中有hive.metastore.warehouse.dir,表示Hive数仓存放数据的默认路径,例如配置成: jfs://emr-jfs /user/hive/warehouse 。

• Database的Location

Hive的Database会有一个Location属性,database的Location作为下属Table的默认路径。默认情况下,创建Database不是必须指定Location,默认会使用hive-site中hive.metastore.warehouse.dir的值加上database的名字作为路径。通过下面的命令可以指定Database的Location到JindoFS:

。 创建Database时指定Location到JindoFS。

```
CREATE DATABASE database_name
LOCATION
'jfs://namespace/database_dir';
```

例如,创建名为database\_on\_jindofs, location为 jfs://emr-jfs/warehouse/database\_on\_jindofs 的Hive数据库。

```
CREATE DATABASE database_on_jindofs
LOCATION
'jfs://emr-jfs/hive/warehouse/database_on_jindofs';
```

- 修改Database的Location到JindoFS。
  - a. 通过SHOW CREAT E语句查看Database的Location。

SHOW CREATE DATABASE database\_name;

通常情况下,默认为warehouse目录。

LOCATION 'hdfs://emr-jfs/user/hive/warehouse/database\_name.db'

b. 通过修改Location,可以把默认路径指定到jindoFS上。此操作不会影响存量表,当新建表没有指定默认Location时, 才会使用此目录。

ALTER DATABASE database\_name SET LOCATION jfs\_path;

• Table/Partition的Location

Table/Partition的Location与Database类似,对于非Partition表,数据直接存放在Table Location下,Partition表的数据存放 在Partition目录下,相关操作如下:

。 创建Table时指定Location到JindoFS。

```
CREATE [EXTERNAL] TABLE table_name
[(col_name data_type,...)]
LOCATION 'jfs://emr-jfs/database_dir/table_dir';
```

- 。 修改Table/Partition指定Location到 JindoFS。
  - a. 通过DESCRIBE语句查看Table/Partition的 location。

DESCRIBE FORMATTED [PARTITION partition\_spec] table\_name;

b. 通过修改Location,可以把默认路径指定到JindoFS上。

ALTER TABLE table\_name [PARTITION partition\_spec] SET LOCATION "jfs\_path";

例如, 查看表 jfs\_table\_name下的某个Partition。

DESCRIBE FORMATTED jfs\_table\_name PARTITION (partition\_key1=123,partition\_key2='xxxx');

# Hive scratch目录

Hive会把一些临时输出文件和作业计划存储在scratch目录,可以通过设置hive-site的hive.exec.scratchdir把地址指向到 JindoFS,也可以通过命令行传参。

bin/hive --hiveconf hive.exec.scratchdir=jfs://emr-jfs/scratch\_dir

#### 或者

```
set hive.exec.scratchdir=jfs://emr-jfs/scratch_dir;
```

# 14.4. 使用Spark处理JindoFS上的数据

Spark处理JindoFS上的数据,主要有两种方式,一种是直接调用文件系统接口使用;一种是通过SparkSQL读取存在JindoFS的数 据表。

# JindoFS配置

以EMR-3.35版本为例,创建名为emr-jfs的命名空间,相关配置参数示例如下:

- jfs.namespaces=emr-jfs
- jfs.namespaces.emr-jfs.oss.uri=oss://oss-bucket/oss-dir
- jfs.namespaces.emr-jfs.mode=block

# 处理JindoFS上的数据

#### • 调用文件系统

Spark中读写JindoFS上的数据,与处理其他文件系统的数据类似,以RDD操作为例,直接使用jfs的路径即可:

```
val a = sc.textFile("jfs://emr-jfs/README.md")
```

写入数据:

scala> a.collect().saveAsTextFile("jfs://emr-jfs/output")

• SparkSQL

创建数据库、数据表以及分区时指定Location到JindoFS即可,SparkSQL处理JindoFS上的数据与HiveSQL类似,详情请参见使用 Hive查询JindoFS上的数据。对于已经创建好的存储在JindoFS上的数据表,直接查询即可。

# 14.5. 使用Flink处理JindoFS上的数据

本文介绍如何使用Flink处理JindoFS上的数据。

### JindoFS配置

以EMR-3.35版本为例,创建名为emr-jfs的命名空间,相关配置参数示例如下:

- jfs.namespaces=emr-jfs
- jfs.namespaces.emr-jfs.oss.uri=oss://oss-bucket/oss-dir
- jfs.namespaces.emr-jfs.mode=block

# 使用JindoFS

Flink作业同样可以将作业的输入输出指定为JindoFS相应Namespace下的路径,即可实现Flink作业对JindoFS数据的交互。

例如, HDFS 上的作业命令如下。

```
flink run -m yarn-cluster -yD taskmanager.network.memory.fraction=0.4 -yD akka.ask.timeout=60s -yjm 2048 -ytm 2048 -ys 4 -yn 14 -c xxx.xxx.FlinkWordCount -p 56 XXX.jar --input hdfs:///test/large-input-flink --output hd fs:///runjob/test/large-output-flink"
```

#### 相应的改成如下命令即可:

• Flink 1.10及之前版本

```
flink run -m yarn-cluster -yD taskmanager.network.memory.fraction=0.4 -yD akka.ask.timeout=60s -yjm 2048 -y
tm 2048 -ys 4 -yn 14 -c xxx.xxx.FlinkWordCount -p 56 XXX.jar --input jfs://emr-jfs/test/large-input-flink --output jfs://emr-jfs/test/large-output-flink"
```

• Flink 1.10之后版本

flink run -m yarn-cluster -yD taskmanager.network.memory.fraction=0.4 -yD akka.ask.timeout=60s -yjm 2048 -y tm 2048 -ys 4 -c xxx.xxx.FlinkWordCount -p 56 XXX.jar --input jfs://emr-jfs/test/large-input-flink --output jfs://emr-jfs/test/large-output-flink"

# 14.6. 使用Impala或Presto查询JindoFS上的数据

本文介绍如何使用Impala或Presto查询JindoFS上的数据。

# JindoFS配置

以EMR-3.35版本为例, 创建名为*emr-jfs*的命名空间, 相关配置参数示例如下:

- jfs.namespaces=emr-jfs
- jfs.namespaces.emr-jfs.oss.uri=oss://oss-bucket/oss-dir
- jfs.namespaces.emr-jfs.mode=block

# 使用JindoFS

目前,在E-MapReduce 3.22.0及以上版本,Impala或Presto支持Hive元数据的读取,对于存储在JindoFS的Hive数据表,E-MapReduce Impala/Presto可以直接读取。

同样,也可以将建表语句的Location设置为JindoFS路径,即可实现表数据落在JindoFS上。

#### 例如,原有HDFS上的建表语句如下:

Create external table lineitem (L\_ORDERKEY INT, L\_PARTKEY INT, L\_SUPPKEY INT, L\_LINENUMBER INT, L\_QUANTITY DO UBLE, L\_EXTENDEDPRICE DOUBLE, L\_DISCOUNT DOUBLE, L\_TAX DOUBLE, L\_RETURNFLAG STRING, L\_LINESTATUS STRING, L\_SH IPDATE STRING, L\_COMMITDATE STRING, L\_RECEIPTDATE STRING, L\_SHIPINSTRUCT STRING, L\_SHIPMODE STRING, L\_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION 'hdfs:///tpch\_impala/lineit em';

#### 相应的改成如下命令即可:

Create external table lineitem (L\_ORDERKEY INT, L\_PARTKEY INT, L\_SUPPKEY INT, L\_LINENUMBER INT, L\_QUANTITY DO UBLE, L\_EXTENDEDPRICE DOUBLE, L\_DISCOUNT DOUBLE, L\_TAX DOUBLE, L\_RETURNFLAG STRING, L\_LINESTATUS STRING, L\_SH IPDATE STRING, L\_COMMITDATE STRING, L\_RECEIPTDATE STRING, L\_SHIPINSTRUCT STRING, L\_SHIPMODE STRING, L\_COMMENT STRING) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|' STORED AS TEXTFILE LOCATION 'jfs://emr-jfs/tpch\_impala/ lineitem';

# 14.7. 使用JindoFS作为HBase的底层存储

本文为您介绍如何使用JindoFS作为HBase的底层存储。

# 背景信息

HBase是Hadoop生态中的实时数据库,有很高的写入性能,E-MapReduce HBase支持使用JindoFS或OSS作为底层存储,相对于HDFS存储,使用更加灵活。

⑦ 说明 建议您使用EMR-3.36.0及后续版本的集群。

# JindoFS配置

以EMR-3.36.0版本为例,创建名为emr-jfs的命名空间,Bucket名称为oss-bucket,相关配置参数示例如下:

- jfs.namespaces=emr-jfs
- jfs.namespaces.emr-jfs.oss.uri=oss://<oss-bucket>/oss-dir
- jfs.namespaces.emr-jfs.mode=block

# 指定HBase的存储路径

指定HBase的存储路径,是需要修改hbase-site配置文件中的参数hbase.rootdir的值为JindoFS或OSS地址,修改参 数hbase.wal.dir的值为本地的HDFS地址,通过本地HDFS集群存储WAL文件。如果要释放集群,需要先Disable table,确保 WAL文件已经完全更新到HFile。

参数

描述

参数	描述	
	指定HBase的ROOT存储目录到JindoFS或OSS。 参数值为 <i>jfs://emr-jfs/hbase-root-dir</i> 。	
hbase.rootdir	⑦ 说明 参数值中的emr-jfs为您配置的命名空间。	
hbase.wal.dir	指定HBase的WAL存储目录到本地HDFS集群。 参数值为: • HA集群: hdfs://emr-cluster/hbase	
	• 非HA集群: hdfs://emr-header-1:9000/hbase	

# 创建集群

在创建集群时,添加软件自定义配置,创建集群详情请参见创建集群。

产品版本:	EMR-3.36.0 > 产品发行版本说明 d
必选服务:	HDFS (2.8.5) YARN (2.8.5) Hive (2.3.8) Spark (2.4.7) Knox (1.1.0) Tez (0.9.2) Ganglia (3.7.2) Sqoop (1.4.7) SmartData (3.6.0) Bigboot (3.6.0)
	Hudi (0.8.0) OpenLDAP (2.4.44) Hue (4.9.0)
可选服务:	HBase (14.9)         ZooKeeper (3.6.2)         Presto (338)         Impala (3.4.0)         Zeppelin (0.9.0)         Flume (1.9.0)         Livy (0.7.1)         Superset (0.36.0)         Ranger (1.2.0)
	Storm (1.2.2)         Phoenix (4.14.1)         ESS (1.0.0)         Alluxio (2.5.0)         Kudu (1.10.0)         Oozie (5.2.1)
《 高级设置	
Kerberos集群模式:	◎ ②
软件自定义配置:	◎ ● 新建集群创建前,可以通过json文件定义集群组件的参数配置,详细信息参考 软件配置 ♂
	[ { "ServiceName":"SMARTDATA",
	"FileName", "namespace", "ConfigKey',"jfs.namespaces", "ConfigValue": emr: jfs
	), {

以JindoFS作为HBase后端为例,需要替换 oss\_bucket 及对应路径,自定义配置如下。

#### E-MapReduce

```
[
 {
       "ServiceName":"SMARTDATA",
      "FileName":"namespace",
      "ConfigKey":"jfs.namespaces",
      "ConfigValue":"emr-jfs"
 },
  {
       "ServiceName":"SMARTDATA",
       "FileName":"namespace",
       "ConfigKey":"jfs.namespaces.emr-jfs.oss.uri",
       "ConfigValue":"oss://oss-bucket/jindoFS"
 },
  {
       "ServiceName":"SMARTDATA",
       "FileName":"namespace",
       "ConfigKey":"jfs.namespaces.emr-jfs.mode",
       "ConfigValue":"block"
 },
  {
       "ServiceName":"HBASE",
       "FileName":"hbase-site",
       "ConfigKey": "hbase.rootdir",
       "ConfigValue":"jfs://emr-jfs/hbase-root-dir"
 },
  {
       "ServiceName":"HBASE",
       "FileName":"hbase-site",
       "ConfigKey":"hbase.wal.dir",
       "ConfigValue": "hdfs://emr-cluster/hbase"
  }
1
```

## 常见问题

• 问题现象: MapReduce程序使用TableSnapshotInputFormat读取Hbase数据时,报错信息如下。

- 问题分析:因为HBase提供的开源MapReduce程序存在缺陷,所以会校验HBase的数据路径跟HDFS中fs.defaultFS路径是否一致。
- 解决方法:
  - 使用ExportSnapshot可以正常读取并导出Hbase数据。
  - 使用TableSnapshotInputFormat读取Hbase数据时,需要您在阿里云E-MapReduce控制台,HDFS服务的配置页面, 在core-site页签,修改fs.defaultFS的参数值为jfs前缀的根目录。例如,修改fs.defaultFS的参数值为上述步骤中配置的jfs://emr-jfs/。

服务配置		
全部 core-site		
214.4.4.4	hdfs://arra handas 1 alustas 247 da0000	0
TS.defaultFS	hdfs://emr-header-1.cluster-24/ 4:9000	•

# 14.8. 基于JindoFS存储YARN MR或SPARK作业日 志

本文为您介绍如何将MapReduce和Spark作业日志配置到JindoFS或OSS上。

# 背景信息

E-MapReduce集群支持按量计费以及包年包月的付费方式,满足不同用户的使用需求。对于按量计费的集群随时会被释放,而 Hadoop默认会把日志存储在HDFS上,当集群释放以后,按量计费的用户就无法查询作业的日志了,因此这也给按量计费用户排 查作业问题带来了困难。因此,您可以将MapReduce和Spark作业日志配置到JindoFS或OSS上,待重新创建集群后,可以继续查 询之前作业相关的日志。

# JindoFS、YARN Container日志和Spark HistoryServer配置

#### ● JindoFS配置

梦蚁	描述	示例
jfs.namespaces	表示当前JindoFS支持的命名空 间,多个命名空间时以逗号隔 开。	emr-jfs
		oss://oss-bucket/oss-dir
jfs.namespaces.emr- jfs.oss.uri	表示emr-jfs命名空间的后端存 储。	<b>⑦ 说明</b> OSS- bucket为您Bucket的名称。
jfs.namespaces.test.mode		block
	表示emr-jfs命名空间为块存储 模式。	<ul> <li>⑦ 说明 JindoFS支持</li> <li>block和cache两种存储模</li> <li>式。</li> </ul>
	jfs.namespaces jfs.namespaces.emr- jfs.oss.uri jfs.namespaces.test.mode	jfs.namespaces 標子的 jfs.namespaces.emr- jfs.oss.uri

#### • YARN Container日志配置

配置文件	参数	描述	示例
yarn-site	yarn.nodemanager.remote- app-log-dir	当应用程序运行结束后,日志 聚合的存储位置,YARN日志聚 合功能默认已打开。	<ul> <li><i>ifs://emr-ifs/emr-cluster-log/yarn-apps-logs</i></li> <li><i>oss://\${oss-bucket}/emr-cluster-log/yarn-apps-logs</i></li> </ul>
	mapreduce.jobhistory.done- dir	JobHistory存放已经运行完的 Hadoop作业记录的目录。	<ul> <li>jfs://emr-jfs/emr-cluster- log/jobhistory/done</li> <li>oss://\${oss-bucket}/emr- cluster-log/jobhistory/do ne</li> </ul>
mapred-site	mapreduce.jobhistory.interm ediate-done-dir	JobHistory存放未归档的 Hadoop作业记录的目录。	<ul> <li>ifs://emr-jfs/emr-cluster- log/jobhistory/done_inter mediate</li> <li>oss://\${oss-bucket}/emr- cluster-log/jobhistory/do ne_intermediate</li> </ul>

## • Spark HistoryServer配置

配置文件	参数	描述	示例
spark-defaults	spark_eventlog_dir	存放Spark作业历史的目录。	<ul> <li>ifs://emr-jfs/emr-cluster- log/spark-history</li> <li>oss://\$[oss-bucket]/emr- cluster-log/spark-history</li> </ul>

# 创建集群

# 在创建集群时,添加软件自定义配置,创建集群详情请参见创建集群。

产品版本:	IR-3.36.0     >     产品发行版本说明 I	
必选服务:	DFS (2.8.5)         YARN (2.8.5)         Hive (2.3.8)         Spark (2.4.7)         Knox (1.1.0)         Tez (0.9.2)         Ganglia (3.7.2)         Sqoop (1.4.7)         SmartData (3.6.0)         Bigboot (3.6.0)           udi (0.8.0)         OpenLDAP (2.4.44)         Hue (49.0)         Hue (	
可选服务:	Base (14.9)       ZooKeeper (3.6.2)       Presto (338)       Impala (3.4.0)       Zeppelin (0.9.0)       Flume (1.9.0)       Livy (0.7.1)       Superset (0.36.0)       Ranger (1.2.0)         orm (12.2)       Phoenix (4.14.1)       ESS (1.0.0)       Alluxio (2.5.0)       Kudu (1.10.0)       Oozie (5.2.1)	
《 高级设置		
Kerberos集群模式:	高安全集群中的各组件会通过Kerberos进行认证,详细信息参考 Kerberos简介 d	
软件自定义配置:	新建集群创建前,可以通过json文件定义集群组件的参数配置,详细信息参考 软件配置 d*	
	<pre>{     {         {</pre>	
⑦ 说明 本文以S	art Dat a 3.6.0为例,如果是Smart Dat a 2.2.3及之前版本,需要修	
改 <mark>ServiceName</mark> Smart Data 2.7.3), 版本,可以直接使用	BIGBOOT , FileName 为 bigboot ;如果是SmartData 2.2.3到SmartData 2.7.3版本(包括 需要修改 ServiceName 为 SMARTDATA , FileName 为 bigboot ;如果是SmartData 2.7.3之后 实示例。	

# 软件自定义配置示例如下:

• 以JindoFS存储日志为例, 替换 oss\_bucket 及对应路径。

### Smart Dat a·最佳实践

```
[
  {
         "ServiceName":"SMARTDATA",
      "FileName":"namespace",
      "ConfigKey":"jfs.namespaces",
       "ConfigValue":"emr-jfs"
  },
  {
          "ServiceName":"SMARTDATA",
      "FileName":"namespace",
       "ConfigKey":"jfs.namespaces.emr-jfs.oss.uri",
       "ConfigValue":"oss://oss-bucket/jindoFS"
  },
  {
          "ServiceName":"SMARTDATA",
      "FileName":"namespace",
       "ConfigKey":"jfs.namespaces.emr-jfs.mode",
       "ConfigValue":"block"
  },
  {
         "ServiceName":"YARN",
      "FileName":"mapred-site",
      "ConfigKey": "mapreduce.jobhistory.done-dir",
      "ConfigValue":"jfs://emr-jfs/emr-cluster-log/jobhistory/done"
  },
  {
          "ServiceName":"YARN",
      "FileName":"mapred-site",
       "ConfigKey": "mapreduce.jobhistory.intermediate-done-dir",
       "ConfigValue":"jfs://emr-jfs/emr-cluster-log/jobhistory/done_intermediate"
  },
  {
          "ServiceName":"YARN",
      "FileName":"yarn-site",
      "ConfigKey":"yarn.nodemanager.remote-app-log-dir",
       "ConfigValue":"jfs://emr-jfs/emr-cluster-log/yarn-apps-logs"
  },
  {
          "ServiceName":"SPARK",
      "FileName":"spark-defaults",
       "ConfigKey":"spark_eventlog_dir",
       "ConfigValue":"jfs://emr-jfs/emr-cluster-log/spark-history"
 }
]
```

• 以OSS存储日志为例, 替换 oss\_bucket 及对应路径。

```
[
  {
          "ServiceName":"YARN",
       "FileName":"mapred-site",
       "ConfigKey": "mapreduce.jobhistory.done-dir",
       "ConfigValue":"oss://oss bucket/emr-cluster-log/jobhistory/done"
  },
  {
          "ServiceName":"YARN",
       "FileName":"mapred-site",
       "ConfigKey": "mapreduce.jobhistory.intermediate-done-dir",
       "ConfigValue":"oss://oss_bucket/emr-cluster-log/jobhistory/done_intermediate"
  },
  ł
           "ServiceName":"YARN".
       "FileName":"yarn-site",
       "ConfigKey":"yarn.nodemanager.remote-app-log-dir",
       "ConfigValue":"oss://oss bucket/emr-cluster-log/yarn-apps-logs"
  },
  {
           "ServiceName":"SPARK",
       "FileName":"spark-defaults",
       "ConfigKey":"spark eventlog dir",
       "ConfigValue":"oss://oss_bucket/emr-cluster-log/spark-history"
   1
1
```

# 14.9. 将Kafka数据导入JindoFS

Kafka广泛用于日志收集、监控数据聚合等场景,支持离线或流式数据处理、实时数据分析等。本文主要介绍Kafka数据导入到 JindoFS的几种方式。

# 常见Kafka数据导入方式

### • 通过Flume导入

推荐使用Flume方式导入到JindoFS,利用Flume对HDFS的支持,替换路径到JindoFS即可完成。

```
al.sinks = emr-jfs
...
al.sinks.emr-jfs.type = hdfs
al.sinks.emr-jfs.hdfs.path = jfs://emr-jfs/kafka/%{topic}/%y-%m-%d
al.sinks.emr-jfs.hdfs.rollInterval = 10
al.sinks.emr-jfs.hdfs.rollSize = 0
al.sinks.emr-jfs.hdfs.rollCount = 0
al.sinks.emr-jfs.hdfs.fileType = DataStream
```

● 通过调用Kafka API导入

对于MapReduce、Spark以及其他调用Kafka API导入数据的方式,只需引用Hadoop FileSystem,然后使用JindoFS的路径写入 即可。

• 通过Kafka Connector导入

使用Kafka HDFS Connect or也可以把Kafka数据导入到Hadoop生态,将sink的输出路径替换成JindoFS的路径即可。

# 14.10. 跨集群访问JindoFS

通常E-MapReduce集群之间相互独立,每个集群的客户端只能连接并访问本集群内配置的namespace。在多集群的情况下,您可以通过配置JindoFS实现跨集群互访。本文以集群A访问集群B为例,介绍如何跨集群访问JindoFS。

#### 前提条件

- 已创建EMR-3.30.0及后续版本的同一VPC下的集群A和B, 详情请参见创建集群。
- 配置/etc/hosts文件,同步B集群所有节点的hosts至A集群。

# 修改配置

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 进入client服务配置。
  - i. 单击**配置**页签。
  - ii. 在服务配置区域,单击client页签。
- 3. 修改配置信息,实现跨集群访问。

根据B集群的namespace.backend.type参数配置A集群:

- 。 当B集群的namespace.backend.type为rocksdb时,执行如下操作:
  - a. 单击右上角的自定义配置。
  - b. 在新增配置项对话框中,添加client.external.namespace.rpc.addresses为 emr-header-1.<cluster-B>:8101 , 单击确定。

⑦ 说明 <cluster-B>为集群B的集群ID。

- 当B集群的namespace.backend.type为raft时,执行如下操作:
  - a. 单击右上角的自定义配置。
  - b. 在新增配置项对话框中,添加client.external.namespace.rpc.addresses为 emr-header-1.<cluster-B>:8101, emr-header-2.<cluster-B>:8101,emr-header-3.<cluster-B>:8101 , 单击确定。
- 4. 保存配置。
  - i. 单击右上角的保存。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# 关联多个集群

client.external.namespace.rpc.addresses配置多个远端地址时,即可实现关联多个集群,不同的集群地址通过英文分号 (;)隔开。

例如,集群A需要关联集群B和集群C,B集群(rocksdb实现)地址为 emr-header-1.<cluster-B>:8101 ,C集群(raft实现) 地址为 emr-header-1.<cluster-C>:8101,emr-header-2.<cluster-C>:8101,emr-header-3.<cluster-C>:8101 那A集群需要添 加的配置信息为 client.external.namespace.rpc.addresses=emr-header-1.<cluster-B>:8101;emr-header-1.<cluster-C>:8101,emr-header-2.<cluster-C>:8101,emr-header-3.<cluster-C>:8101 。

# 14.11. 改写Jindo HDFS客户端路径

Smart Dat a 3.1.x版本支持改写Jindo HDFS客户端级别的路径,以减少集群迁移时修改路径的工作量。例如,通过将HDFS地址重 写至OSS地址,方便您迁移HDFS中的数据至OSS后,无需改动业务逻辑中的数据地址,即可访问数据。

#### 使用限制

仅支持Hadoop 2.x版本,不支持Hadoop 3.x版本。

#### 开启路径改写功能

- 1. 进入SmartData服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。

- iv. 在集群管理页面,单击相应集群所在行的详情。
- v. 在左侧导航栏,选择**集群服务 > SmartData**。
- 2. 单击**配置**页签。
- 3. 在服务配置区域, 单击smart dat a-sit e页签。
- 4. 在服务配置区域,单击右侧的自定义配置。
- 5. 在新增配置项对话框中,添加Key为fs.hdfs.impl, Value为com.aliyun.emr.fs.hdfs.jindoHdfsShimsFileSystem的配置项。

新增配置项			×
* Key	* Value	描述	操作
fs.hdfs.impl	com.aliyun.emr.fs.hdfs.JindoHdfsShimsFileSystem		删除
添加			
			<b>确定</b> 取消

- 6. 单击**确定**。
- 7. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# 改写配置路径

- 1. 进入Smart Dat a服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的**集群管理**页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > SmartData。
- 2. 单击配置页签。
- 3. 在服务配置区域,单击smartdata-site页签。
- 4. 在服务配置区域,单击右侧的自定义配置。
- 5. 在新增配置项对话框中,添加如下两个配置项。

参数	描述	参数值
fs.jindo.shim.path-rewrite. <rule- NAME&gt;.source</rule- 	路径重写的挂载点。	<ul> <li>HA集群 hdfs://emr-cluster/<osspath></osspath></li> <li>非HA集群 hdfs://<your_hostname>:9000/<os spath&gt;</os </your_hostname></li> <li>② 说明 <your_hostname>您 可以通过SSH登录主节点,执行 h ostname 命令获取,SSH登录主 节点详情请参见登录集群。</your_hostname></li> </ul>
fs.jindo.shim.path-rewrite. <rule- NAME&gt;.target</rule- 	您实际访问的路径。	oss:// <your_bucket>/<testpath></testpath></your_bucket>

#### RULE-NAME需要您自定义。

6. 单击**确定**。

### 7. 保存配置。

- i. 单击右上角的保存。
- ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
- ⅲ. 单击确定。

# 示例

HA集群,在smart dat a-sit e页签,添加如下参数后,您访问 hdf s://emr-clust er/osspat h时实际访问的是 oss://jindo-bucket/<testpath>的数据。

参数	参数值
fs.jindo.shim.path-rewrite.testrule.source	hdfs://emr-cluster/osspath
fs.jindo.shim.path-rewrite.testrule.target	oss://jindo-bucket/ <testpath></testpath>

#### 您可以通过SSH登录集群的主节点,执行如下命令,查看改写情况。

hadoop fs -ls /

#### 通过如下信息,看到osspath已经挂载在根目录下。

[root@emr-hea	ader-l ~]# ha	adoop fs -ls /		
Found 7 items	3			
drwxr-xr-x	<ul> <li>hadoop</li> </ul>	hadoop	0 2020-11-24	21:12 /apps
drwxrwxrwx	<ul> <li>flowagent</li> </ul>	hadoop	0 2020-11-24	21:11 /emr-flow
drwxr-xx	- root	hadoop	0 2020-11-24	21:12 /emr-sparksql-udf
drwxrwxrwx			0 2020-11-25	16:58 /osspath
drwxr-xx	- hadoop	hadoop	0 2020-11-25	16:56 /spark-history
drwxrwxrwx	- root	hadoop	0 2020-11-25	16:55 /tmp
drwxr-xt	- hadoop	hadoop	0 2020-11-24	21:14 /user
r				

# 14.12. 支持Flink可恢复性写入JindoFS或OSS

Smart Dat a 3.0.x版本支持Flink可恢复性写入OSS, Smart Dat a 3.1.x版本支持Flink可恢复性写入JindoFS或OSS。通过Flink自有的 检查点(Checkpoint)机制,当写入存储介质的作业发生局部失败时,作业可以迅速自动恢复,并继续写入。

# 背景信息

可恢复性写入功能支持将数据以EXACTLY\_ONCE语义写入存储介质,在大数据场景下保证了数据的安全性和一致性。

# 在Flink作业中的用法

- 通用配置
  - 为了支持EXACTLY\_ONCE语义写入JindoFS或OSS,您需要执行如下配置:
    - i. 打开Flink的检查点(Checkpoint)。

示例如下。

a. 通过如下方式建立的StreamExecutionEnvironment。

StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();

b. 执行如下命令, 启动Checkpoint。

env.enableCheckpointing(<userDefinedCheckpointInterval>, CheckpointingMode.EXACTLY\_ONCE);

- ii. 使用可以重发的数据源,例如Kafka。
- 便捷使用

您无需额外引入依赖,只需使用带有*jfs://或oss://*前缀的路径,就可以使用该功能。JindoFS可以自动识别*jfs://或oss://*前 缀,并启用该功能。

例如,以DataStream<String>的对象OutputStream为例。

i. 添加Sink。

#### ■ 将其写入JindoFS时,您可以执行如下命令添加Sink。

#### ■ 将其写入OSS,您可以执行如下命令添加Sink。

ii. 使用 env.execute() 执行Flink作业即可。

# 自定义配置

您在提交Flink作业时,可以自定义参数,以开启或控制特定功能。

例如,以yarn-cluster模式提交Flink作业时,通过 -yD 配置。示例如下。

<flink\_home>/bin/flink run -m yarn-cluster -yD key1=value1 -yD key2=value2 ...

Smart Dat a 支持通过配置开启熵注入(Entropy Injection)功能或控制分片上传(Multipart Upload)的并行度。

• 熵注入 (Smart Dat a 3.1.x及其后续版本)

该功能可以匹配写入路径的一段特定字符串,用一段随机的字符串进行替换,以削弱所谓片区效应,提高写入效率。

○ 如果是写入JindoFS(Block或Cache模式),则需要提供下列配置。

jfs.entropy.key=<user-defined-key>
jfs.entropy.length=<user-defined-length>

○ 如果是写入OSS,则需要提供下列配置。

oss.entropy.key=<user-defined-key>
oss.entropy.length=<user-defined-length>

• 分片上传 (Smart Dat a 3.0.x及其后续版本)

当写入场景为OSS或JindoFS Cache模式时,可恢复性读写功能会自动调用高效的分片上传机制,将待上传的文件分为多个数据 块分别上传,最后组合。目前支持配置参数*oss.upload.max.concurrent.uploads*,用来控制上传数据块的并行度,如果设置 较高的数值则可能会提高写入效率(但也会占用更多资源)。默认情况下,该值为当前可用的处理器数量。

# 14.13. 使用Flume写入JindoFS

Apache Flume是一个分布式、可靠和高可用的系统,用于从大量不同的数据源有效地收集、聚合和移动大量日志数据,进行集中式的数据存储。Flume的核心是Agent,Agent中包含Source、Channel和Sink。本文为您介绍如何使用HDFS Sink写入数据至 JindoFS。

#### 前提条件

已创建集群,并选择了Flume服务。创建集群详情,请参见创建集群。

#### 背景信息

Apache Flume的HDFS Sink通过Flush接口保证事务性写入。JindoFS Block模式从Smart Dat a 3.2.x及后续版本开始默认支持Flush 接口,您可以直接配置Sink。

OSS本身不支持Flush接口,Smart Data 3.4.x及后续版本支持Flume可恢复性写入JindoFS Cache模式或OSS。通过支持Flush接口,虽然不能让数据立刻可见,但是可以确保数据暂存在云端,当写入程序发生Crash时,您可以通过调用SDK或命令行,恢复程序在Crash之前已经写入JindoFS Cache模式或OSS的数据。

# 使用Flume写入JindoFS Block模式

您需要配置Sink, 各参数含义请参见Apache Flume。配置Sink示例如下:

#### # 配置JFS Sink

```
xxx.sinks.jfs_sink.hdfs.path = jfs://${your_ns_name}/flume_dir/%H%M/%S
xxx.sinks.jfs_sink.hdfs.round =true
xxx.sinks.jfs_sink.hdfs.roundValue = 15
xxx.sinks.jfs_sink.hdfs.Unit = minute
xxx.sinks.jfs_sink.hdfs.filePrefix = your_topic
# Sink参数, batchSize需要设置大一些,建议每次Flush的量在32MB以上,否则会影响性能。
xxx.sinks.jfs_sink.hdfs.batchSize = 100000
...
xxx.sinks.jfs_sink.rollSize = 3600
xxx.sinks.jfs_sink.threadsPoolSize = 30
xxx.sinks.jfs_sink.fileType = DataStream
xxx.sinks.jfs_sink.writeFormat = Text
```

⑦ 说明 本文示例中的 your\_ns\_name 为您NameSpace的名称。

### 使用Flume写入JindoFS Cache模式或OSS

- 1. 开启Flush和Recover功能。
  - 具体步骤,请参见开启Flush和Recover功能。
- 2. 配置Sink。

各参数含义请参见Apache Flume。配置Sink示例如下:

```
# 配置OSS Sink
xxx.sinks.oss_sink.hdfs.path = oss://${your_bucket}/flume_dir/%H%M/%S
xxx.sinks.oss_sink.hdfs.round = true
xxx.sinks.oss_sink.hdfs.roundValue = 15
xxx.sinks.oss_sink.hdfs.filePrefix = <your_topic>
# Sink参数, batchSize需要设置大一些,建议每次Flush的量在32MB以上,否则会影响性能。
xxx.sinks.oss_sink.hdfs.batchSize = 100000
...
xxx.sinks.oss_sink.rollSize = 3600
xxx.sinks.oss_sink.threadsPoolSize = 30
```

⑦ 说明 本文示例中的 your\_bucket 为您OSS Bucket的名称。

3. 恢复文件Flush的数据。

具体步骤,请参见恢复文件Flush的数据。

# 开启Flush和Recover功能

使用Flume写入JindoFS Cache模式时,需要开启Flush和Recover功能。

- 1. 进入Smart Data服务。
  - i. 登录阿里云E-MapReduce控制台。
  - ii. 在顶部菜单栏处,根据实际情况选择地域和资源组。
  - iii. 单击上方的集群管理页签。
  - iv. 在集群管理页面,单击相应集群所在行的详情。
  - v. 在左侧导航栏,选择集群服务 > Smart Data。
- 2. 进入smartdata-site页面。
  - i. 单击配置页签。
  - ii. 在服务配置页面, 单击storage。

#### 3. 单击右上角的自定义配置,添加如下参数。

参数	描述
fs.jfs.cache.oss.flush.enable	是否开启Flush和Recover功能,开启时需要设置为true。
fs.jfs.cache.flush.staging.path	Flush的数据和Manifest的暂存区,默认值为/tmp。 例如:在使用默认值时,如果文件的写入路径是: <i>oss://test-bucket/dir1/file1</i> ,则Staging的路径为 <i>oss://test-bucket/tmp/dir1/file1</i> 。

- 4. 保存配置。
  - i. 单击右上角的**保存**。
  - ii. 在确认修改对话框中, 输入执行原因, 开启自动更新配置。
  - ⅲ. 单击确定。

# 恢复文件Flush的数据

您可以使用Recover命令恢复数据。

jindo jfs -	recover	[-R]				
		[-flushStagingPath {flushStagingPath}]				
		[-accessKeyId \${accessKeyId}]				
		[-accessKeySecret \${accessKeySecret}]				
		<path></path>				

参数	描述
-R	是否递归Recover,恢复文件夹时需要添加该参数。
-flushStagingPath	Flush的数据和Manifest的暂存区,默认值为/tmp。 例如:在使用默认值时,如果文件的写入路径是: <i>oss://test-bucket/dir1/file1</i> ,则Staging的 路径为 <i>oss://test-bucket/tmp/dir1/file1</i> 。
-accessKeyld	阿里云账号的AccessKey ID。
-accessKeySecret	阿里云账号的AccessKey Secret。
path	Flush的数据的写入路径。 例如: <i>oss://test-bucket/dir1/file1或oss://test-bucket/dir1</i> 。

#### 您也可以通过JindoOssFileSystem的Recover接口恢复文件或目录。本示例为恢复文件夹。

JindoOssFileSystem jindoFileSystem = (JindoOssFileSystem) fs; boolean isFolder = true; jindoFileSystem.recover(path, isFolder);

⑦ 说明 代码中的path表示待恢复文件的路径, isFolder表示是否需要递归恢复。如果是恢复文件夹, 需要设置为true。 如果是恢复文件, 需要设置为false。

# 15.SmartData常见问题

本文汇总了使用Smart Dat a时的常见问题。

### 基本概念

- 什么是JindoFS?
- 已经有阿里云OSS,为什么还要使用JindoFS?
- JindoFS有哪些使用方式? 使用场景是什么?
- JindoFS SDK和缓存模式的区别是什么?
- JindoFS缓存模式和Block模式的区别是怎么?
- JindoFS Block模式可以通过OSS API读取数据吗?
- 对象存储OSS不支持Rename操作,那JindoFS支持Rename操作吗?
- JindoFS的Rename性能如何?
- JindoFS支持类似于Hadoop S3A的Magic Committer吗?
- JindoFS对百万千万级文件数目录的支持情况如何?
- JindoFS是如何保证数据可靠性的?
- JindoFS支持文件和目录操作的一致性吗?
- JindoFS支持文件和目录操作的原子性吗?
- JindoFS Block模式如何保证HA?
- JindoFS Block模式在集群上保存文件元数据,重建集群时数据怎么处理?
- EMR已经支持HDFS,为什么还要有JindoFS Block模式?
- JindoFS和Alluxio相比有什么技术差异和优势?
- 跟HDFS相比,使用JindoFS和OSS能节省成本吗?
- Hadoop社区版本也提供OSS支持, JindoFS有什么优势?
- JindoFS提供Fuse支持吗?和OSS自带的Fuse有什么优势?
- EMR中的Smart Dat a和JindoFS是什么关系?
- EMR中的Bigboot和JindoFS是什么关系?

# 开源和生态

- JindoFS支持哪些开源组件?
- JindoFS吞吐如何? 会不会影响Spark或Hive大规模分析计算?
- JindoFS写性能如何? Flume或Kafka在写入数据时碰到瓶颈如何处理?
- JindoFS支持Flink实时计算场景吗?
- JindoFS和OSS场景下,可以使用Presto做交互式分析吗?
- 如果使用JindoFS,如何迁移HDFS上的数据?
- 使用Impala时,可以通过JindoFS查询OSS上的数据吗?
- JindoFS支持使用Delta Lake,或者Hudi和IceBerg时,存放数据在OSS上吗?
- 数据存放在OSS上, JindoFS支持机器学习训练吗?
- 基于MaxCompute数仓上的数据, JindoFS如何帮助机器学习训练?
- 基于Hive数仓上的数据, JindoFS如何帮助机器学习训练?
- 如何处理Bigboot日志占用过大的问题?

#### 升级和迁移

- 如果使用JindoFS,如何迁移HDFS上的数据?
- JindoFS在新版本才有,如果需要在EMR集群上使用JindoFS,该如何处理?
- JindoFS支持哪些Hadoop版本和发行厂商?
- JindoFS可以在ECS自建集群上使用吗?
- JindoFS可以在阿里云ACK环境上使用吗?

- 使用JindoFS会被阿里云E-MapReduce绑定吗?
- JindoFS可以在IDC机房的Hadoop集群使用吗?

# 已知版本问题汇总

- /opt/bignode目录占用巨大并持续增长,该如何处理?
- 读缓存时数据出错,该如何处理?

# OSS相关

- 如何查看JindoFS上的数据量?
- JindoFS查看的数据量和OSS产品控制台上看到的数据量不一致时如何处理?
- 什么情况下建议打开OSS Bucket 的多版本控制?
- 打开OSS Bucket多版本控制对EMR和JindoFS的影响是什么?
- OSS归档存储可以大量节省存储成本, JindoFS提供相应的支持吗?

# 安全相关

- 使用JindoFS, 会泄露AccessKey吗?
- 什么是AccessKey免密?
- 如果支持AccessKey免密,那如何区分不同的用户和权限限制?
- 如何使用不同的AccessKey, 通过JindoFS访问不同的OSS Bucket?
- 在无EMR管控支持情况下,想使用自建的IDC集群,又不想在集群节点上配置AccessKey,该如何处理?
- JindoFS支持Auditlog吗?
- JindoFS支持Ranger集成吗?

# 使用问题汇总

JindoFS如何手动下线节点?

# 什么是JindoFS?

JindoFS是阿里云开源大数据E-MapReduce产品提供的一套Hadoop文件系统,主要对Hadoop和Spark大数据生态系统使用阿里 云OSS提供多层次的封装支持和优化。

基础功能提供适配OSS和支持访问,您可以直接使用JindoFS SDK;标准功能针对OSS提供分布式缓存优化,以对应JindoFS缓存模式;高级功能上针对使用OSS一些特殊或重要场景进行了深度定制,例如,JindoFS Block模式。

# 已经有阿里云OSS,为什么还要使用JindoFS?

阿里云OSS是对象存储系统,提供基于对象语义的REST API和各种语言SDK封装。JindoFS主要是对阿里云OSS提供 HCFS(Hadoop Compatible FileSystem)接口封装,并且在此基础上提供缓存加速能力和高级优化定制的功能。因为Hadoop 和Spark生态组件依赖HCFS的抽象接口,所以需要使用JindoFS。

# JindoFS有哪些使用方式?使用场景是什么?

JindoFS使用方式包括JindoFS SDK(*jindo-sdk\_xxx.jar*)、缓存和Block模式。

针对三种方式,使用场景如下:

- JindoFS SDK模式:简单情况下,您可以使用此模式,上传JindoFS SDK的JAR包至组件的classpath目录。
- 缓存模式:当计算性能受限于IO和存储吞吐时,您可以使用此模式,在计算集群的Core节点上增加或扩容磁盘,以开启数据缓存。
- Block模式:特殊场景,例如对元数据操作性能和一致性要求高时,使用此模式。

#### JindoFS SDK和缓存模式的区别是什么?

JindoFS SDK和缓存模式完全兼容阿里云OSS,通过这两种方式您可以通过OSS产品提供的API和SDK,直接读取写入OSS的文件。

缓存模式需要部署和配置Jindo分布式缓存服务,打开数据缓存开关,而JindoFS SDK则不需要。如果缓存服务出现故障,系统自 动切换至JindoFS SDK方式,直接读写OSS文件。

#### JindoFS缓存模式和Block模式的区别是怎么?

Block模式可以管理文件的元数据,组织数据的块,把OSS作为磁盘来使用,类似HDFS。读写Block模式的数据需要通过JindoFS SDK 客户端。

缓存模式兼容OSS,可以直接读写数据。例如,您通过缓存模式写一个大文件,可以在OSS控制台对应目录下找到这个大文件。 如果是块缓存模式时,您只能找到很多文件块,这些块只能通过JindoFS SDK 客户端拼接成大文件。

#### 如果更新了OSS上的数据,如何保证JindoFS缓存数据的一致性?

如果OSS对象被修改、覆盖或删除,JindoFS在读取OSS对象的时候,首先会检查OSS对象的Meta信息和MD5,然后对比本地缓存的信息,检查是否发生了变化。如果发生了变化,本次读取放弃本地缓存直接读取OSS,并更新缓存。

#### JindoFS Block模式可以通过OSS API读取数据吗?

不可以。只能通过JindoFS SDK客户端读取数据。

#### 对象存储OSS不支持Rename操作,那JindoFS支持Rename操作吗?

支持。因为JindoFS支持HCFS文件系统接口,所以支持文件和目录的Rename操作。

对象存储OSS因为没有文件和目录的概念,所以不支持文件和目录的Rename操作,需要通过模拟文件系统的方式来实现Rename 操作(先拷贝对象至新位置,再删除旧的对象)。

#### JindoFS的Rename性能如何?

JindoFS的Rename性能优于社区版本。如果是文件,OSS支持大对象 Fast Copy 优化,JindoFS 利用该优化做到比社区版本快很 多;如果是目录,涉及到很多文件,JindoFS通过充分并发优化,也比社区版本快多倍。

#### JindoFS支持类似于Hadoop S3A的Magic Committer吗?

JindoFS支持无需Rename操作的Magic Committer。

#### JindoFS对百万千万级文件数目录的支持情况如何?

针对百万千万级文件数的大目录,JindoFS支持并发访问和内存优化,不会出现OOM(Out Of Memory)或者挂起。

#### JindoFS是如何保证数据可靠性的?

因为jindoFS无论使用哪种方式,数据都存放在OSS上,本地磁盘只缓存数据,所以数据可靠性是由OSS来保证的。

#### JindoFS支持文件和目录操作的一致性吗?

支持。JindoFS Block模式实现HDFS文件系统语义,支持强一致性。

#### JindoFS支持文件和目录操作的原子性吗?

JindoFS Cache模式不支持原子性。JindoFS Cache模式因为要兼容OSS,受限于OSS对象存储限制,不支持跨对象操作的原子性。 例如Rename操作,至少涉及到源和目标两个对象,如果是目录的Rename,涉及的对象更多。

JindoFS Block模式严格实现HDFS文件系统语义,支持原子性,包括Rename操作。

#### JindoFS Block模式如何保证HA?

JindoFS Block模式基于Raft分布式一致性协议可以部署多个Jindo NamespaceService节点,并且元数据的更新支持异步备份至阿 里云T ablest ore数据库上。

#### JindoFS Block模式在集群上保存文件元数据,重建集群时数据怎么处理?

JindoFS Block模式的元数据的更新支持异步备份至阿里云Tablestore数据库上,在确保生产集群停止更新,所有修改同步至 Tablestore后,可以停掉JindoFS集群,此时,所有数据在OSS和Tablestore上。重建集群时,恢复OSS和Tablestore上数据至重 建集群。

⑦ 说明 重建集群时,需要考虑版本的兼容性。例如,EMR-2.7.x版本之间都是兼容的,但EMR-2.7.x和EMR-2.6.x之间则不一定。如果是升级到不兼容的大版本时,建议通过Jindo Dist Cp同步Block模式数据至OSS。

# EMR已经支持HDFS,为什么还要有JindoFS Block模式?

JindoFS Block模式从技术架构和功能上确实和HDFS相似,都是自定义管理文件元数据并组织数据,具有强一致性。

JindoFS Block模式的优势在于,数据备份至OSS上,支持弹性扩展、低成本且无需维护磁盘。

# JindoFS和Alluxio相比有什么技术差异和优势?

对比项	JindoFS	Alluxio	
相同点	JindoFS缓存模式在技术架构上与Alluxio类似,都提供对OSS的缓存加速能力,支持Master + Workers形式,Master维护缓存块的位置信息,Workers提供缓存块的管理和读写能力。		
	JindoFS不需要挂载,可以直接访 问 <i>oss://</i> 路径,只需打开数据缓存开关即 可。	Alluxio需要先挂载OSS Bucket位置至名字空 间 <i>,</i> 再使用 <i>alluxio://</i> 访问	
不同点	JindoFS核心支持的是OSS,性能极致优化。	Alluxio支持数据源很多,可以同时挂载到统 一的名字空间。	
	JindoFS提供基础的SDK模式支持访问适配 OSS,全面对接各种开源引擎。	无	

# 跟HDFS相比,使用JindoFS和OSS能节省成本吗?

HDFS存储时,不能弹性伸缩,预算不足就会面临存储空间不足,或者存在空间浪费的情况。

阿里云OSS是海量对象存储,支持弹性伸缩,具有归档存储功能,可以备份冷数据。JindoFS基于OSS,支持数据冷热分层和数据 归档存储策略,使用得当,整体上可以降低成本。

# Hadoop社区版本也提供OSS支持, JindoFS有什么优势?

Hadoop社区版本支持的OSS,受到社区整体约束,只具备OSS基本适配功能。

JindoFS对OSS的支持优势如下:

- 更全面:对接各种开源引擎。
- 更活跃:对OSS的新功能提供同步更新和升级。
- 更高级:具有高阶缓存加速能力和高级定制功能Block模式。
- 更快:性能更优。JindoFS核心代码采用C++ native代码开发,各种基本操作性能优于社区版本。

# JindoFS提供Fuse支持吗?和OSS自带的Fuse有什么优势?

提供。JindoFS提供的Fuse优势在于能够利用JindoFS分布式缓存和Block模式功能。

# JindoFS支持哪些开源组件?

支持按照HCFS接口读写数据的组件,例如,Hadoop、Hive、Spark、Flink、Presto、HBase、Impala、Druid、Kafka和Flume。

# JindoFS吞吐如何? 会不会影响Spark或Hive大规模分析计算?

JindoFS在适配上充分发挥OSS并发吞吐能力,实现异步并发读取,利用Concurrent Multipart Upload特性进行并发分块写入,在 读写吞吐上面比社区版本具有较大优势。

JindoFS缓存模式和Block模式可以利用集群本地磁盘或内存来缓存数据,对于新写入的数据和重复读取的数据具有显著加速效果。在同样集群条件下,对于Spark或Hive分析计算,跟HDFS相比集群吞吐是相当的,甚至优于HDFS。

#### JindoFS写性能如何? Flume或Kafka在写入数据时碰到瓶颈如何处理?

因为HDFS需要写三备份才算写入成功,JindoFS只需写入OSS一备份就算写入成功,所以通常情况下,JindoFS写性能优于HDFS。 如果集群的Flume或Kafka在写入数据时碰到瓶颈,请提交工单处理。

#### JindoFS支持Flink实时计算场景吗?

支持。JindoFS支持Flink读OSS source, checkpoint和sink到OSS以及Exactly-Once语义。

# JindoFS和OSS场景下,可以使用Presto做交互式分析吗?

可以。JindoFS缓存模式和Block模式都支持Presto交互式分析,且性能稳定。

#### 使用Impala时,可以通过JindoFS查询OSS上的数据吗?

可以。Impala 3.4及后续版本支持JindoFS,可以读写OSS。

# JindoFS支持使用Delta Lake,或者Hudi和IceBerg时,存放数据在OSS上吗?

支持。

### 数据存放在OSS上, JindoFS支持机器学习训练吗?

支持。您可以使用JindoFS缓存模式,通过预加载将OSS数据提前写入内存或者SSD做缓存,然后训练引擎可以通过JindoFuse支持 直接读取。

### 基于MaxCompute数仓上的数据, JindoFS如何帮助机器学习训练?

有如下两种方式:

- MaxCompute数仓作业将数据通过MaxCompute外表方式写入至OSS,然后在训练集群通过JindoFS缓存模式和JindoFuse来加载训练。
- 通过JindoTable从MaxCompute拉取数据写入至JindoFS缓存模式,然后使用JindoFuse来加载训练。

#### 基于Hive数仓上的数据, JindoFS如何帮助机器学习训练?

类似于MaxCompute数仓上的数据处理方式,方式详情请参见基于MaxCompute数仓上的数据,JindoFS如何帮助机器学习训练? 。

#### 如果使用JindoFS,如何迁移HDFS上的数据?

您可以使用Jindo Dist Cp同步HDFS数据至JindoFS或OSS。Jindo Dist Cp比Hadoop Dist Cp性能高,且支持OSS归档。

### JindoFS在新版本才有,如果需要在EMR集群上使用JindoFS,该如何处理?

如果集群规模不大,建议重建集群来使用JindoFS和EMR新版本。如果规模较大,请提交工单处理。

#### JindoFS支持哪些Hadoop版本和发行厂商?

JindoFS SDK提供OSS适配功能,明确支持Hadoop 2.7后续版本和Hadoop 3.x版本。

Hortonworks版本 (Hortonworks Data Platform, 简称HDP)和Cloudera版本 (Cloudera's Distribution Including Apache Hadoop, 简称CDH)都可以使用,但可能会存在冲突,需要修改配置 fs.oss.impl = JindoOssFileSystem 。

# JindoFS可以在ECS自建集群上使用吗?

可以。需要您下载jindoFS SDK手工部署即可。如果您需要使用jindoFS缓存模式和Block模式功能,建议您登录阿里云E-MapReduce控制台,直接使用E-MapReduce产品。

#### lindoFS可以在阿里云ACK环境上使用吗?

可以。

#### 使用JindoFS会被阿里云E-MapReduce绑定吗?

不会。JindoFS遵循标准HCFS接口,兼容和支持全面开源生态,不会绑定。

#### JindoFS可以在IDC机房的Hadoop集群使用吗?

可以。您可以直接下载开源JindoFS SDK按照文档部署使用即可。如果集群出现兼容性问题,请提交工单处理。

#### 如何查看JindoFS上的数据量?

您可以直接使用如下命令查看统计情况。

hadoop dfs -du/count

#### JindoFS查看的数据量和OSS产品控制台上看到的数据量不一致时如何处理?

通常是因为在OSS Bucket打开了多版本,导致历史版本数据未清理,或者JindoFS Block模式较早版本,因为内存泄露,导致残留 数据,此时您可以<mark>提交工单</mark>。

#### 什么情况下建议打开OSS Bucket的多版本控制?

对于重要的数据建议打开OSS Bucket多版本,防止误删时数据不丢失。

# 打开OSS Bucket多版本控制对EMR和JindoFS的影响是什么?

对于Hive或Spark中间结果存放的数据以及频繁修改的数据,不建议使用多版本Bucket,会影响计算性能。

### OSS归档存储可以大量节省存储成本, JindoFS提供相应的支持吗?

JindoFS支持相应的OSS归档存储。Block模式上,提供专门的存储策略与OSS归档匹配。

### 使用JindoFS, 会泄露AccessKey吗?

JindoFS支持在集群上配置使用AccessKey,但存在泄漏AccessKey的风险。在EMR集群里或者在ECS环境,如果节点绑定了ECS role,您可以使用权限管理,不使用AccessKey。

# 什么是AccessKey免密?

EMR集群提供AccessKey免密,该功能通过EMR管控得到用户授权,方便用户拿到具有权限的阿里云STS (Securit y Token Service) ,然后使用该 Token访问阿里云资源,例如OSS。

#### 如果支持AccessKey免密,那如何区分不同的用户和权限限制?

AccessKey免密不是适用于所有的场景。

如果有多个用户需要区分权限,有如下两种方式:

- 您可以通过RAM用户权限控制,每个用户使用RAM用户来访问OSS。
- 您可以使用JindoFS权限控制,通过Ranger来授权。

↓ 注意 JindoFS仅能在Namespace上设定权限控制。

# 如何使用不同的AccessKey,通过JindoFS访问不同的OSS Bucket?

您可以使用JindoFS multinamespace,每个Namespace配置不同的OSS bucket和对应的AccessKey信息。

# 在无EMR管控支持情况下,想使用自建的IDC集群,又不想在集群节点上配置AccessKey,该 如何处理?

您可以使用Hadoop Crendential Provider机制,详情请参见Credential Provider使用说明

#### JindoFS支持Auditlog吗?

支持。JindoFS支持MultiNamespaces,每个Namespace上可以设定Auditlog,默认不打开。

#### JindoFS支持Ranger集成吗?

支持。JindoFS支持Multi Namespaces,每个Namespace上可以设定支持Ranger,默认不打开。

#### EMR中的SmartData和JindoFS是什么关系?

Smart Dat a是产品组件,该组件包括JindoFS服务。

#### EMR中的Bigboot和JindoFS是什么关系?

Bigboot是SmartData组件的基础设施,对该组件所包含服务提供毫秒级进程监测和日志清理等功能。

#### 如何处理Bigboot日志占用过大的问题?

EMR-3.36.1或EMR-5.2.1之前的版本,会出现Bigboot日志占用过大的问题。当您觉得Bigboot占用日志过大时,针对已有的日志 文件需要您手动删除,后续您可以参照以下步骤新增配置,将日志级别由INFO修改为WARN,以减少打印过多的日志信息。

1. 在EMR控制台新增配置。

- i. 在Bigboot服务的配置页面, 单击namespace页签。
- ii. 单击自定义配置。

iii. 在新增配置项对话框中,添加参数名为logger.level,参数值为1的配置项。

<返回 💼 Smart	Data 🖌 🍨 正常				◎ 查看操作历史 首 快	2015 - 🖉 接作 - 🗸
状态 部署拓扑	配置 配置修改历史					
配置过渡		服务配置			0	部署客户端配置 保存
新増配置项						× 自定义配置
* Key		* Value		描述	操作	
logger.level		1			删除	
添加						
					確定	取消
? 说明	此处的日志级别默·	认为0表示IN	FO,修改为1表示WA	\RN。		

- ⅳ. 单击确定。
- 2. 保存配置。
  - i. 单击保存。
  - ii. 在确认修改对话框中, 输入执行原因, 单击确定。
- 3. 重启Jindo Namespace Service。
  - i. 在右上角选择操作 > 重启Jindo Namespace Service。
  - ii. 在执行集群操作对话框中, 输入执行原因, 单击确定。
  - iii. 在确认对话框中, 单击确定。

# /opt/bignode目录占用巨大并持续增长,该如何处理?

- 问题现象:通常情况下, /opt/bignode目录正常大小应该是几个GB,但Smart Data 2.x版本, /opt/bignode目录占用巨大, 并且出现磁盘占用过大并持续增长的情况。
- 问题原因:可能是由于监控进程异常退出,导致对Jindo Storage Service进程状态监控异常,从而不断重复拉起进程导致的。
- 处理方法: 您可以通过以下步骤排查并处理。

在出现问题的节点上,以root用户执行以下命令,查看进程信息。

ps -aux | grep b2-storageservice | grep -v grep

通常情况下应该有两个进程,分别是xxxx/services/b2-storageservice.spec和xxxx/bin/b2-storageservice。如果缺少 xxxx/services/b2-storageservice.spec进程,则可以确认是监控进程异常退出导致的,需手动终止xxxx/bin/b2-storageservice进程。

kill -9 <PID of b2-storageservice>

② 说明 <PID of b2-storageservice>为您查看进程信息时,查询到的xxxx/bin/b2-storageservice进程的进程ID。

执行以上命令后,监控进程会自动拉起新的Storage Service服务,目录占用会自动恢复正常。如果以上方式仍旧无法解决该问题,请提交工单处理。

#### 读缓存时数据出错,该如何处理?

- 问题现象:Smart Data 3.0至Smart Data 3.7版本,使用JindoFS Block模式(Block模式默认启用缓存),或者使用Cache模式并 打开了缓存的情况下,生成数据时没有问题,但读取数据时出现内容污染的现象,例如,作业对格式源数据(ORC或 Parquet)报格式错误、HBase报HFile格式错误等。
- 问题原因:可能是遇到了缓存数据块读取流程上的程序已知缺陷,误读了错误的缓存数据块导致的。
- 处理方法:您可以通过以下方式进行快速恢复处理,并联系EMR对历史版本程序进行升级修复。

请根据实际需求选择以下方式,快速规避该缺陷,避免对业务产生持续影响。

#### ○ 方式一:关闭缓存。

通常该缺陷是缓存数据导致的,通过关闭缓存可以彻底避免该问题。您可以在SmartData服务的client配置页面通过配置关闭缓存,Block模式配置jfs.data-cache.enable为false,Cache模式配置jfs.cache.data-cache.enable为false。

• 方式二: 对报错文件进行缓存清理。

如果出于性能等方面考虑,不适合关闭缓存,可以选择此方式。

a. 执行以下命令, 提交缓存清理任务。

jindo jfs -uncache <**报错文件全路径**>

b. 执行以下命令, 查看任务状态, 并且确定报错文件的缓存已清理完成。

jindo jfs -status <报错文件全路径>

c. 在SmartData服务的client配置页面,添加自定义配置项storage.compaction.enable为false。

添加参数的详情,请参见添加组件参数。

d. 重启Jindo Storage Service服务。

重启Jindo Storage Service服务的详情,请参见重启服务。

#### JindoFS如何手动下线节点?

手动下线节点的命令格式如下。

jindo jfs -decommission <excludeFile>

② 说明 命令中的 <excludeFile> 为新建的文件,文件中的内容是待下线节点的主机名称,每行代表一个待下线的节点,节点主机名称您可以通过 hostname 命令获取。文件内容格式如下。

emr-worker-1.cluster-29\*\*\*\*
emr-worker-2.cluster-29\*\*\*\*