ALIBABA CLOUD

阿里云

云数据库HBase版

HBase Ganos 时空引擎

文档版本：20200902

阿里云

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。

1. 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。

2. 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。

3. 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。	危险 重置操作将丢失用户配置数据。
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。	警告 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。	注意 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，不是用户必须了解的内容。	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。	**在结果确认页面，单击确定。**
`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 list --instanceid` *Instance_ID*
[] 或者 [a|b]	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.开通指南

HBase Ganos作为HBase的一个组件，本身不收取任何额外费用，只收取HBase费用。用户在购买HBase之后，即可在控制台中开通。

开通方法

2.Java SDK

2.1. 导入数据

构建SimpleFeature

HBase Ganos通过SimpleFeature类表示空间要素。每个SimpleFeature由ID，Geometry对象与其他属性构成，GeoTools API提供了SimpleFeatureBuilder类帮助用户创建SimpleFeature对象：

```
SimpleFeatureType sft = ....;
SimpleFeatureBuilder sfBuilder = new SimpleFeatureBuilder(sft);
builder.set("属性名", 属性值);
...
builder.set("geom", Geometry);
SimpleFeature feature = builder.buildFeature(object_id + "_" + date.getTime());
```

> 说明 构建SimpleFeature时，Ganos会默认生成128位的UUID作为Feature ID。为了节省存储空间，用户可以自己指定ID，具体方式为：
>
> ```
> SimpleFeature feature =...
> feature.getUserData().put(Hints.USE_PROVIDED_FID, java.lang.Boolean.TRUE);
> ```

创建Geometry对象

每个SimpleFeature包含一个Geometry对象用来表示要素的空间对象。Geometry的各个空间实体对象定义具体请参见官方文档。

GeoTools API提供了GeometryFactory工具类帮助用户创建Geometry对象，具体方法如下：

- 点要素
  - 通过Coordinate对象

    ```
    GeometryFactory geometryFactory = JTSFactoryFinder.getGeometryFactory();
    Coordinate coord = new Coordinate(1, 1);
    Point point = geometryFactory.createPoint(coord);
    ```

  - 通过WKT描述：WKT(Well-known text)是一种文本标记语言，用于表示矢量空间对象、空间参照系统及空间参照系统之间的转换

    ```
    GeometryFactory geometryFactory = JTSFactoryFinder.getGeometryFactory();
    WKTReader reader = new WKTReader(geometryFactory);
    Point point = (Point) reader.read("POINT (1 1)");
    ```

- 线要素

- 通过Coordinate对象

```
GeometryFactory geometryFactory = JTSFactoryFinder.getGeometryFactory();
Coordinate[] coords =
new Coordinate[] {new Coordinate(0, 2), new Coordinate(2, 0), new Coordinate(8, 6) };
LineString line = geometryFactory.createLineString(coordinates);
```

- 通过WKT描述

```
GeometryFactory geometryFactory = JTSFactoryFinder.getGeometryFactory();
WKTReader reader = new WKTReader( geometryFactory );
LineString line = (LineString) reader.read("LINESTRING(0 2, 2 0, 8 6)");
```

- 面要素
  - 通过Coordinate对象

```
GeometryFactory geometryFactory = JTSFactoryFinder.getGeometryFactory();
Coordinate[] coords =
new Coordinate[] {new Coordinate(4, 0), new Coordinate(2, 2),
new Coordinate(4, 4), new Coordinate(6, 2), new Coordinate(4, 0) };
LinearRing ring = geometryFactory.createLinearRing( coords );
LinearRing holes[] = null; // use LinearRing[] to represent holes
Polygon polygon = geometryFactory.createPolygon(ring, holes );
```

  - 通过WKT描述

```
GeometryFactory geometryFactory = JTSFactoryFinder.getGeometryFactory( null );
WKTReader reader = new WKTReader( geometryFactory );
Polygon polygon = (Polygon) reader.read("POLYGON((20 10, 30 0, 40 10, 30 20, 20 10))");
```

数据入库

HBase Ganos通过GeoTools API中的SimpleFeatureWriter写入数据，SimpleFeatureWriter支持事务，可以通过DataStore的getFeatureWriterAppend方法获取。

- 插入单个SimpleFeature

```
SimpleFeatureType sft = ....;
SimpleFeatureWriter writer=(SimpleFeatureWriter)ds.getFeatureWriterAppend(sft.getTypeName(), Tr
ansaction.AUTO_COMMIT);
SimpleFeature toWrite=writer.next();
toWrite.setAttributes(feature.getAttributes());
toWrite.getUserData().putAll(feature.getUserData());
writer.write();
writer.close();
```

- 批量插入SimpleFeature

HBase Ganos支持批量插入SimpleFeature，通过GeoTools API中的SimpleFeatureStore类实现。

```
List<SimpleFeature> features=...
SimpleFeatureStore featureStore = (SimpleFeatureStore) ds.getFeatureSource(sft.getTypeName());
List<FeatureId> featureIds = featureStore.addFeatures(new ListFeatureCollection(sft,features));
```

2.2. 数据查询

一般地，使用Ganos API进行时空查询时，首先需要构建一个org.geotools.data.Query对象，并指定过滤条件、返回列名，排序等相关参数，然后通过DataStore提交Ganos集群运行，最后查询结果会以SimpleFeature集合的形式返回给用户。

1.CQL通用查询语言

Query对象中的查询条件是通过构建CQL查询语句进行的。CQL（Common Query Language，通用查询语言）是OGC为目录web服务规范创建的查询语言，不像基于XML的编码语言，CQL使用我们更熟悉的文本语法编写，具有更好的可读性和适应性。

- 比较运算符包括： =，<>，>，>=，<，<=，如要选取人口大于1500万的城市，条件为PERSONS>15000000，其中PERSONS为人口数量的字段，后面不再单独说明。
- BETWEEN表示从一个范围到另一个范围，如PERSONS BETWEEN 1000000 AND 3000000。
- 比较运算符支持文本值，可在运算符=的右侧指定文本值，如 CITY_NAME='Beijing'，表示选择北京市；也可以使用LIKE操作符，与SQL中的用法一样，如 CITY_NAME LIKE 'N%'会选择所有以N开头的城市。
- 对两个属性进行比较，如 MALE > FEMALE，选择男性多余女性的城市。
- 算术表达式可以使用 +, -, *, /构成，如过滤条件 UNEMPLOY/(EMPLOYED + UNEMPLOY) > 0.07选择所有失业率大于7%的城市。
- 使用IN操作符，可选择属性在给定值范围内的，与SQL用法相同，如 ID IN ('cities.1', 'cities.12')，或查找名字在给定值内的城市，可使用 CITY_NAME IN ('Beijing','Shanghai', 'Guangzhou')。
- 可使用Geoserver中的任何过滤函数，如 strToLowerCase（CITY_NAME） like '%m%'，表示名字中包含m的城市，不区分M的大小写。
- 几何过滤，使用BBOX，如 BBOX(the_geom, -90, 40, -60, 45)表示选择在（-90, 40, -60, 45)范围内的城市。 关于CQL的详细介绍请参见ECQL参考。

2. 空间关系查询

CQL中定义的空间关系查询谓词如下表所示：

语法	描述
INTERSECTS(Expression , Expression)	判断两个空间要素是否相交
DISJOINT(Expression , Expression)	判断两个空间要素是否相离
CONTAINS(Expression , Expression)	判断第一个要素是否在拓扑关系上包含第二个要素
WITHIN(Expression , Expression)	判断第一个要素是否在拓扑关系上被包含在第二个要素之中
TOUCHES(Expression , Expression)	判断两要素是否相接，即至少有一个公共点要素
CROSSES(Expression , Expression)	判断两个空间要素是否相交，要求仅有部分交集而不是完全包含
EQUALS(Expression , Expression)	判断两个空间要素是否相等
BBOX (Expression , Number , Number , Number , Number [, CRS])	测试空间要素是否与由其最小和最大X和Y值指定的边界框相交。可选的CRS是包含SRS代码的字符串（例如，'EPSG：1234'。默认使用查询图层的CRS）

比如获取所有位于空间范围：（120E，30N，130E，40N）中的所有要素，可以：

```
DataStore ds = DataStoreFinder.getDataStore(params);
SimpleFeatureType schema=...
String stFilter = "bbox(geom, 120,30,130,40)"
Query query = new Query(schema, ECQL.toFilter(stFilter));
SimpleFeatureCollection features=ds.getFeatureSource(schema).getFeatures(query);
```

或者获取(46.9 48.9, 47.1 48.9, 47.1 49.1, 46.9 49.1, 46.9 48.9)构建的多边形中的所有元素：

```
String stFilter = "contains('POLYGON ((46.9 48.9, 47.1 48.9, 47.1 49.1, 46.9 49.1, 46.9 48.9))', geom)
Query query = new Query(schema, ECQL.toFilter(stFilter));
```

3. 时空查询

HBase Ganos支持的时间查询谓词如下：

语法	语法
Expression BEFORE Time	时间在Time之前
Expression BEFORE OR DURING Time Period	时间在 Time Period之前或包含在 Time Period之中
Expression DURING Time Period	时间在Time Period之中
Expression DURING OR AFTER Time Period	时间在Time Period之中或之后
Expression AFTER Time	时间在Time Period之后

HBase Ganos支持的时间表达方式有多种：

语法	描述
Time / Time	由起始时间和结束时间定义的区间
Duration / Time	在Time之前的时间区间
Time / Duration	在Time之后的时间区间

注意：目前HBase Ganos还不支持单独时间查询，需要配合空间查询一起进行。

如用户想查询位于（120E，30N，130E，40N）之中，时间介于2014-01-01T11:45:00与2014-01-01T12:15:00之间的要素：

```
String stFilter = "bbox(geom, 120,30,130,40) AND dtg DURING 2014-01-01T11:45:00.000Z/2014-01-01T12:15:00.000Z";
Query query = new Query(schema, ECQL.toFilter(stFilter));
SimpleFeatureCollection features=ds.getFeatureSource(schema).getFeatures(query);
```

4. 属性查询

如3.1节所示，当用户指定针对某一属性字段创建索引之后，就可以对该列进行属性查询：

```
String filter = " name = 'bob'"
val q = new Query(sft.getTypeName, ECQL.toFilter(filter))
SimpleFeatureCollection features=ds.getFeatureSource(schema).getFeatures(query);
```

上例查询中对name字段的值做了限制。

5. 指定返回列名

用户可以通过配置Query对象参数指定具体返回哪些列，如：

```
String[] returnFields=... //指定返回字段名
Query query = new Query(schema, ECQL.toFilter(ecqlPredicate));
query.setPropertyNames(returnFields);
SimpleFeatureCollection features=ds.getFeatureSource(schema).getFeatures(query);
```

6. 指定排序方式

用户可以通过构建SortBy对象参数指定具体返回哪些列，如：

```
String sortField=... //指定排序字段名
FilterFactory2 ff = CommonFactoryFinder.getFilterFactory2();
SortBy[] sort = new SortBy[]{ff.sort(sortField, order)};
query.setSortBy(sort);
SimpleFeatureCollection features=ds.getFeatureSource(schema).getFeatures(query);
```

3.RESTful API

3.1. query

Ganos支持属性查询、id查询、时空范围查询等，数据查询的方式为：

URL	/index/:alias/:index/features
方法	GET
URL参数	必要:alias=[alphanumeric] ds名称index=[alphanumeric] index 名称可选:q=[alphanumeric] JSON 表示的查询条件
成功信息	Code: 200 Content: GeoJSON 格式的要素集合
失败信息Code: 400 - 需要的参数未指定Content: empty	

```
curl\
'localhost:8080/geoserver/geomesa/geojson/index/:alias/:index/features' \ --get --data-urlencode 'q=JSON格式查询条件'
```

HBase Ganos支持的属性查询谓词：

$lt	小于
$lte	小于等于
$gt	大于
$gte	大于等于

以下分别介绍几种常用的查询方式：

（1） 属性查询 HBase Ganos 属性查询通过谓词运算:

示例 1: 查询 id=0 的要素

```
curl \
'localhost:8080/geoserver/geomesa/geojson/index/my_ds/test/features' \
--get --data-urlencode
'q={
"properties.id":"0"
}'
```

示例 2: 查询 name=n1 的要素

```
curl \
'localhost:8080/geoserver/geomesa/geojson/index/my_ds/test/features' \
--get --data-urlencode
'q={"properties.name":"n1"}'
```

示例 3: 查询 age<30 的所有要素

```
curl \
'localhost:8080/geoserver/geomesa/geojson/index/my_ds/test/features' \
--get --data-urlencode
'q={"properties.age":{"$lt":30}}'
```

（2）空间查询

示例 1: 通过外包框 BBOX 查询:

```
q={
"geometry" :
{
"$bbox" : [-180, -90, 180, 90]
}
}
```

示例 2: 空间相交(intersection)查询:

```
q={
"geometry" : {
"$intersects" : {
"$geometry" : {
"type" : "Point",
"coordinates" : [30, 10]
}
}
}
}
```

示例 3: 空间包含(within)查询:

```
q={
"geometry" : {
"$within" : { "$geometry" : {
"type" : "Polygon",
"coordinates": [ [ [0,0], [3,6], [6,1], [0,0] ] ]
}}
}
}
```

示例 4: 空间包含(contains)查询:

```
q={
"geometry" : {
"$contains" : {
"$geometry" : {
"type" : "Point",
"coordinates" : [30, 10]
}
}
}
}
```

（3）时间查询

示例 1:使用$during 谓词指定时间区间

```
curl \
'localhost:8080/geoserver/geomesa/geojson/index/myds/test/features' \
--get --data-urlencode
'q={"dtg":{"$during":"2018-01-02T08:00:00Z/2018-03-02T10:00:00Z"}}'
```

示例 2: 使用 $lt(), $gt $lte 和 $gte

```
curl \ 'localhost:8080/geoserver/geomesa/geojson/index/myds/test/features' \
--get --data-urlencode
'q={"dtg":{"$lt" : "2013-01-02 00:00:00"}}'
```

（4）组合查询

AND： 创建查询条件: a=5 AND b=6,

```
{ "a" : 5, "b" : 6 }
```

OR： 创建查询条件: a=5 OR a=6，

```
{ "$or" : [ { "a" : 5 }, { "b" : 6 } ] }
```

示例 1: 时空查询配合属性查询:

```
curl /
'http://localhost:8080/geoserver/geomesa/geojson/index/tdrive_ds/tdrive_index/featu res'
--get --data-urlencode
'q={
"geometry":{"$bbox":[116.3383,39.8291,116.3384,39.8292]},
"properties.taxi_num":"1131",
"properties.dtg":{"$gt" : "2008-02-08T08:00:00.000+0000"},
"properties.dtg":{"$lt" : "2008-02-08T12:21:16.000+0000"}
}'
```