Alibaba Cloud E-MapReduce

User Guide

Issue: 20190220

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades , adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults " and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity , applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use

or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.

5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified , reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates . The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).
6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
⛔	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	⛔ Danger: Resetting will result in the loss of user configuration data.
⚠	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	⚠ Warning: Restarting will cause business interruption. About 10 minutes are required to restore business.
📋	This indicates warning information, supplementary instructions, and other content that the user must understand.	ⓘ Notice: Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructions, best practices, tips, and other content that is good to know for the user.	📋 Note: You can use Ctrl + A to select all files.
>	Multi-level menu cascade.	Settings > Network > Set network type
Bold	It is used for buttons, menus, page names, and other UI elements.	Click OK.
`Courier font`	It is used for commands.	Run the `cd /d C:/windows` command to enter the Windows system folder.
`Italics`	It is used for parameters and variables.	`bae log list --instanceid` *`Instance_ID`*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *`[-all|-t]`*

Style	Description	Example
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich` *{stand | slave}*

Contents

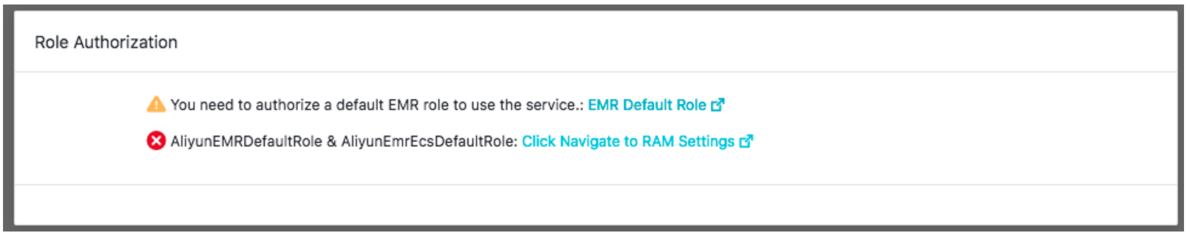
1 Role authorization

When you activate the E-MapReduce service, a default system role named AliyunEMRDefaultRole must be granted to the E-MapReduce service account. Once assigned, E-MapReduce can call the relevant services (such as ECS and OSS), create clusters, save logs, and perform other related tasks.

> Notice:
>
> If you are activating E-MapReduce for the first time, you must authorize roles by using the primary account. Otherwise, the primary and user accounts cannot access or use E-MapReduce.

Role authorization process

1. When you create a cluster or an on-demand execution plan, if a default role is not authorized to the E-MapReduce service account, the following prompt is displayed. Click Go to RAM for authorization to authorize the role.

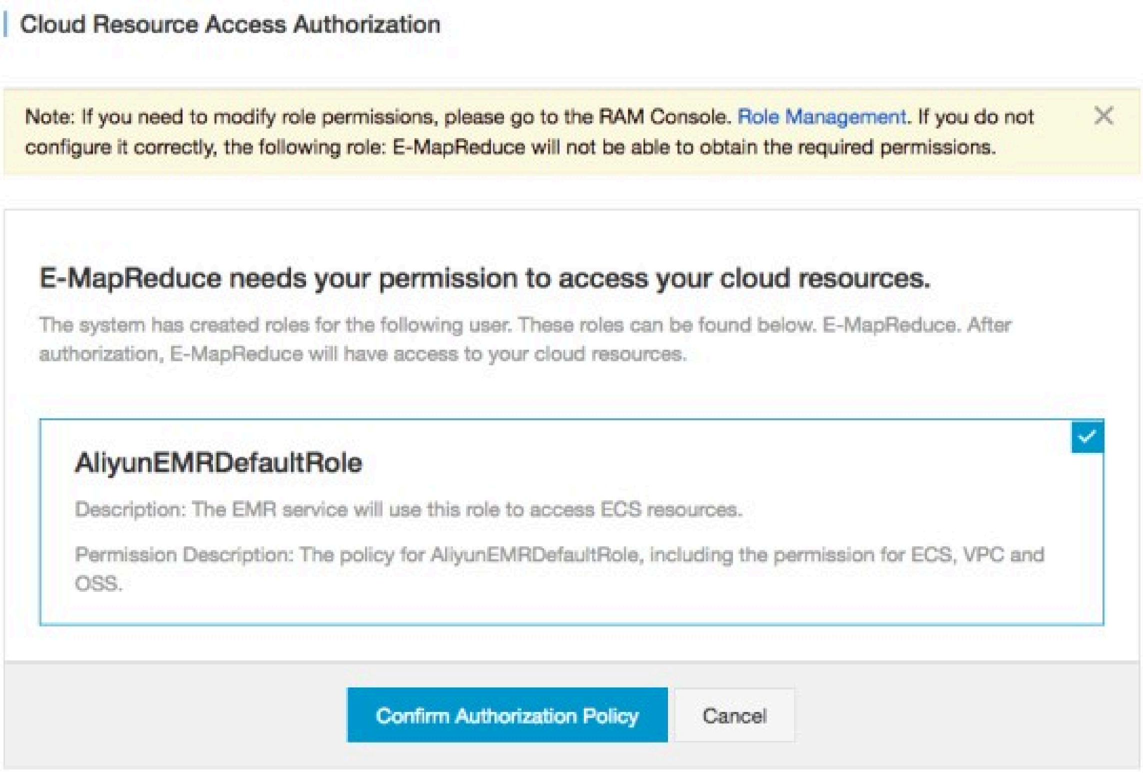


Role Authorization

⚠ You need to authorize a default EMR role to use the service.: EMR Default Role

☒ AliyunEMRDefaultRole & AliyunEmrEcsDefaultRole: Click Navigate to RAM Settings

2. On the RAM authorization page, click Confirm Authorization Policy to authorize the default role AliyunEMRDefaultRole to the E-MapReduce service account.



Cloud Resource Access Authorization

Note: If you need to modify role permissions, please go to the RAM Console. Role Management. If you do not configure it correctly, the following role: E-MapReduce will not be able to obtain the required permissions.

E-MapReduce needs your permission to access your cloud resources.

The system has created roles for the following user. These roles can be found below. E-MapReduce. After authorization, E-MapReduce will have access to your cloud resources.

AliyunEMRDefaultRole

Description: The EMR service will use this role to access ECS resources.

Permission Description: The policy for AliyunEMRDefaultRole, including the permission for ECS, VPC and OSS.

3. Refresh the E-MapReduce console, and then perform relevant operations.

   If you want to view relevant detailed policy information of AliyunE-MapReduceDefaultRole, log on to the RAM console, or click *View Link*.

Default role permissions

The default role AliyunEMRDefaultRole has the following permissions

- ECS related permissions:

Permission name (Action)	Description
ecs:CreateInstance	Create ECS instances
ecs:RenewInstance	Renew ECS instances.
ecs:DescribeRegions	Query ECS region information.
ecs:DescribeZones	Query zone information.
ecs:DescribeImages	Query image information.
ecs:CreateSecurityGroup	Create security groups.
ecs:AllocatePublicIpAddress	Allocate a public network IP address.
ecs:DeleteInstance	Delete machine instances.
ecs:StartInstance	Start machine instances.

Permission name (Action)	Description
ecs:StopInstance	Stop machine instances.
ecs:DescribeInstances	Query machine instances.
ecs:DescribeDisks	Query the machine's relevant disk information.
ecs:AuthorizeSecurityGroup	Set security group input rules.
ecs:AuthorizeSecurityGroupEgress	Set security group output rules.
ecs:DescribeSecurityGroupAttribute	Query security group details.
ecs:DescribeSecurityGroups	Query security group list information.

- OSS related permissions

Permission name (Action)	Description
oss: PutObject	Upload file or folder objects.
oss: GetObject	Get file or folder objects.
oss: ListObjects	Query file list information.

Grant permissions to a RAM user account

To ensure that user accounts can access the E-MapReduce service, you need to log on with your primary account to the *RAM console* and set AliyunEMRFullAccess or AliyunEMRDevelopAccess policies to grant user accounts access to E-MapReduce.

Note:

- The AliyunEMRFullAccess policy grants user accounts with full access permissions to E-MapReduce and E-MapReduce resources.
- The AliyunEMRDevelopAccess policy grants user accounts the E-MapReduce Developer permission, but, unlike AliyunEMRFullAccess, it does not grant users with other access permissions, such as the permissions for creating or releasing E-MapReduce clusters.

For information about RAM user accounts, policies, and roles, see *RAM*.

2 Configure clusters

2.1 Instance types

There are three types of node instances in an E-MapReduce cluster: master, core, and task.

Different service processes are deployed on each instance type. For example, with Hadoop, the HDFS NameNode and YARN ResourceManager services are deployed on master instances, while the HDFS DataNode and YARN NodeManager services are deployed on core instances. For task instances, because they are only used in computing tasks, only YARN NodeManager is deployed, and not HDFS-related services.

When you create a cluster, you must determine the ECS specifications for each instance type. ECS instances of the same type must be in the same instance group. If you increase the number of hosts in a core or task instance group, you can scale the cluster up at a later date. This does not apply to master instance groups.

Note:

Task instances are supported in version 3.2.0 or later.

Master instance

The master instance is where the management and control components of the cluster service are deployed. You can connect to the master instance using SSH and check service statuses in the cluster through the software's Web UI.

If you want to perform a test or run a job, log on to the master instance and submit jobs directly at the command line. By default, only one master instance is used. However, if the cluster's high availability feature is enabled, two are used.

Core instance

Core instances, which are managed by master instances, store all of the data in the cluster. They also deploy computing services to perform computing tasks. If you need more data storage or are experiencing heavier workloads, you can scale core instances up at any time without impacting the operations of the cluster.

Task instance

Task instances are responsible for computing and can quickly add computing power to a cluster. They can also scale up and down at any time without impacting the operations of the cluster. However, this instance type is optional, and if the core instance has enough computing power, task instances are not necessary. Depending on the fault tolerance (or retries) of the computing service, a reduction in the number of task instance nodes may cause MapReduce and Spark jobs to fail.

2.2 Gateway clusters

A gateway cluster is an independent cluster that consists of multiple nodes of the same configuration.

When you create a gateway cluster, you can associate it with an existing Hadoop cluster. To facilitate cluster operations, it is recommended that you associate it with a cluster on which Hadoop (HDFS and YARN), Hive, Spark, Sqoop, Pig, or other clients have been deployed. It is an independent submission point and does not use up cluster resources, especially when you submit jobs on the Master node. This improves the stability of the Master node. If there are too many jobs to submit, you can add nodes as and when you need them.

You can also create multiple gateway clusters for different users, allowing them to use their own environment to meet different service requirements.

2.3 ECS instances

This section contains information on the different ECS instance types.

ECS instance types supported by E-MapReduce

- General-purpose

  This type uses cloud disks as storage. The ratio between vCPUs and memory is 1:4. For example, 32 cores and 128 GB memory.
- Compute

  This type uses cloud disks as storage and provides more computing resources. The ratio between vCPUs and memory is 1:2. For example, 32 cores and 64 GB memory.
- Memory

This type uses cloud disks as storage and provides more memory resources. The ratio between vCPUs and memory is 1:8. For example, 32 cores and 256 GB memory .

- Big data

  This type uses local SATA disks as storage, which is highly cost-effective. If you want to store massive amounts of data (TB-level), it is recommended that you use this type.
- Ephemeral SSD

  This type uses ephemeral SSDs as storage, which provides high local IOPS and throughput.
- Shared (entry level)

  This type shares CPUs and is not stable enough for most scenarios. It is applicable for entry-level users, not enterprise customers.
- GPU

  This type is a heterogeneous GPU-based model applicable in machine learning scenarios.

ECS instance types applicable in different scenarios

- Master instances

  General-purpose and memory types are applicable in master instances, where data is directly stored on Alibaba Cloud's cloud disks. There are also three backups to guarantee high data reliability.
- Core instances

  General-purpose, compute, and memory types are applicable for small data volumes (not TB-level) or when OSS is used as the primary data storage. When the amount of data is large (10 TB or more), it is recommended that you use the big data type, which is more cost-effective. Using an ephemeral disk makes it harder to ensure data reliability, but this can be maintained and guaranteed by the E-MapReduce platform.
- Task instances

  All types except the big data type are suitable for task instances to give additional computing power to the cluster. Currently, the ephemeral SSD type is not supported, but will be added soon.

2.4 Storage guide

There are two types of disks on a node: the system disk, which is used to install operating systems, and the data disk, which is used to store data.

A node typically has one system disk by default, which must be a cloud disk. However , you can have more than one data disk (currently, up to sixteen on a single node). Each data disk can have different configurations, including having a different type or capacity. In E-MapReduce, a cluster's system disks are SSD cloud disks by default , and four are used by default. Considering current intranet bandwidth, this default configuration of four cloud disks is sufficient.

Cloud and ephemeral disks

Two types of disk are available for data storage.

- Cloud disks

  Includes SSD, ultra, and basic cloud disks.

  Cloud disks are not attached directly to the local computing node. Instead, they access a remote storage node through the network. Each piece of data has two real-time backups at the backend, meaning that there are three identical copies in total . When one is corrupted (due to disk damage), a backup is used automatically for recovery.
- Ephemeral disks

  Includes ephemeral SATA disks in the big data type and ephemeral SSD disks used in the ephemeral SSD type.

  Ephemeral disks are attached directly to the computing node and have a better performance than cloud disks. You cannot change the number of ephemeral disks. As with offline physical hosts, there is no data backup at the backend, meaning that upper-layer software is required to guarantee data reliability.

Usage scenarios

In E-MapReduce, when the hosting node is released, all of the data in the cloud and ephemeral disks is cleared. The disks can also not be kept independently and used again. Hadoop HDFS uses all data disks for data storage. Hadoop YARN uses all data disks as on-demand data storage for computing.

If you do not have massive amounts of data (below TB-level), you can use cloud disks , as the IOPS and throughput are smaller than local disks. In the event that you have large amounts of data, it is recommend that you use local disks whose data reliability is guaranteed by E-MapReduce. If you find the throughput to be insufficient, switch to ephemeral disks.

OSS

OSS can be used as HDFS in E-MapReduce, and you can have easy read and write access to OSS. All code that uses HDFS can also be easily modified to access data on OSS. Below you can find a number of examples:

Reading data from Spark

```
sc.textfile("hdfs://user/path")
```

Changing the storage type from HDFS to OSS

```
sc.textfile("oss://user/path")
```

This is the same for Map Reduce and Hive jobs.

HDFS commands process OSS data directly:

```
hadoop fs -ls oss://bucket/path
hadoop fs -cp hdfs://user/path  oss://bucket/path
```

In this process, you do not need to enter the AK or endpoint. E-MapReduce automatically completes your information using the current cluster owner.

However, as OSS does not have high IOPS, it is not suitable for usage scenarios that require high IOPS, such as Spark Streaming or HBase.

2.5 D1 series

To meet the demand for storage in big data scenarios, Alibaba Cloud has launched the D1 series on the cloud.

Instead of using cloud disks in its data storage, the D1 series uses ephemeral disks , which solves the problem of high costs caused by keeping multiple copies of redundant data in cloud disks. Data also no longer needs to be transferred over the network, which improves disk throughput. Furthermore, with the D1 series, you can also take advantage of Hadoop's proximity computing.

Compared with cloud disks, the series greatly enhances storage performance while reducing prices. Indeed, the cost is almost the same as offline physical hosts.

Despite their advantages, however, ephemeral disks still cannot ensure data reliability, and upper-layer software is required to guarantee it. If a disk or node fails, operations and maintenance must be performed manually. Cloud disks, meanwhile, guarantee data reliability automatically, meaning that you do not need to worry about disk damage. Alibaba Cloud's default multi-disk backup policy is also helpful in this regard.

E-MapReduce + D1 solution

A complete set of automated O&M solutions, such as the D1 series, is now available for ephemeral disks in E-MapReduce. This allows Alibaba Cloud users to use ephemeral disks conveniently and reliably, without having to worry about the entire O&M process. Data reliability and service availability are guaranteed.

The main advantages are as follows:

- Highly reliable distribution of required nodes
- Ephemeral disk and node fault monitoring
- Automatic determination of data migration opportunities
- Automatic failed node migration and data balancing
- Automatic HDFS data detection
- Network topology optimization

With the automated O&M of the entire back-end management and control system, E-MapReduce helps you make better use of ephemeral disks and develop a cost-effective big data system.

Note:

If you want to set up a Hadoop cluster using the D1 series, submit a ticket. We will then be able to assist you in your operations.

2.6 Enable access between classic networks and VPCs

This section describes how to enable inter-access between ECS on classic networks and E-MapReduce clusters on VPC networks.

ClassicLink

Alibaba Cloud currently provides two types of cloud network: classic and VPC. While some users still use classic networks, E-MapReduce clusters use VPCs.

To grant access between ECS on a classic network and an E-MapReduce cluster on a VPC network, Alibaba Cloud launches *ClassicLink*. Follow these steps:

1. Create a vSwitch according to the CIDR block specified in ClassicLink.
2. To deploy a cluster that you have created, use the vSwitch of the CIDR block.
3. Connect the corresponding classic network node to the VPC in the ECS console.
4. Set the security group rules.

3 Clusters

3.1 Create a cluster

In this tutorial, you will learn how to create an Alibaba Cloud E-MapReduce (EMR) cluster.

Go to the EMR cluster creation page

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. Complete RAM authorization. For details, see *Role authorization*.
3. Select a region for the cluster. The region cannot be changed once the cluster is created.
4. Click Create Cluster to go to the cluster creation page.

Create a cluster

> Notice:
> After you create an EMR cluster, the only thing that can be changed is its name.

To create a cluster, follow these three steps:

1. Configure the software.
   - EMR version: The main version of E-MapReduce represents a complete open source software environment and can be upgraded regularly based on upgrades made to the internal component software. If the software related to Hadoop is upgraded, the main version of E-MapReduce is also upgraded. Clusters from an earlier version cannot be upgraded to a later version.
   - Cluster type: Currently, E-MapReduce provides four cluster types.
     - Hadoop clusters, which provide the following semi-managed ecosystem components:
       - Hadoop, Hive, and Spark for large-scale offline distributed data storage and computing.
       - Spark Streaming, Flink, and Storm for stream processing.
       - Presto and Impala for running interactive analytics.
       - Oozie and Pig.

  - Druid clusters, which provide semi-managed, real-time interactive analysis services, query large amounts of data at millisecond latency, and support multiple data intake methods. When used with services such as EMR Hadoop , EMR Spark, OSS, and RDS, Druid clusters offer real-time query solutions.
  - Data Science clusters, which are mainly applicable in big data and AI scenarios, providing Hive and Spark offline big data, and TensorFlow model training.
  - Kafka clusters, which are semi-managed distributed message systems that feature high throughput and high scalability, providing a complete service monitoring system that can keep a stable running environment.
- Required Services: Displays a list of all software components under the selected cluster type, including their name and version number.
- Optional Services:You can select different components as required. The selected components start relevant service processes by default.

  Note:

  The more components you select, the higher the requirements are for configuration, as there may be insufficient resources to run these services.
- High security mode: In this mode, you can set the cluster's Kerberos authentication. This feature is unnecessary for clusters used by individual users and is turned off by default.
- Enable custom setting: Before you start a cluster, you can specify a JSON file to change the software configuration.

2. Configure the hardware.
   - Billing method
     - Like with ECS, both Subscription and Pay-As-You-Go modes are supported. If you select Subscription mode, you must also select the duration. You can select 1, 2, 3, 6, or 9 months, or 1, 2, or 3 years. This mode is applicable to short-term testing or flexible dynamic tasks, but is relatively expensive.
   - Cluster network configuration
     - Zone: Select the zone where the cluster is to be located. If better network connectivity is required, we recommend selecting the same availability zone. However, this increases the risk of failure when creating a cluster, as the

availability zone's storage may be insufficient. If you need a large number of nodes, please submit a ticket.
  - Network type: The Virtual Private Cloud (VPC) network is selected by default, which requires you to enter a VPC and a VSwitch. If you have not created a network, go to the *VPC console* to create one. For more information about E-MapReduce VPC, see *VPC*.
  - VPC: Select the region of the VPC network.
  - VSwitch: Select a zone for VSwitch under the corresponding VPC. If no VSwitch is available in this zone, you must create a new one.
  - Security group name: A security group does not typically exist when you first create a cluster. To create a new security group, enter a name. If you already have a security group, you can select it here.
- Cluster configuration
  - High availability: When enabled, two master instances in the Hadoop cluster are used to ensure the availability of the Resource Manager and Name Node. HBase clusters support high availability by default.
  - Node type: The three types of node supported are as follows:
    - Master, which is mainly responsible for the deployment of control processes such as Resource Manager and Name Node.
    - Core, which is mainly responsible for the storage of all data in the cluster, and can be scaled up as required.
    - Task, which is the node used for computing. It does not store data and is used to adjust the computing capacity of the cluster.
  - Node configuration: Select different node types. Different types of nodes have different application scenarios.
  - Data disk type: The data disks used by a cluster node are either standard cloud disks, high-efficiency cloud disks, or SSD cloud disks. This varies between machine type and region. When the user selects different regions, disks that are supported by those regions are displayed in the drop-down list. By default, data disks are released when the cluster is released. The ephemeral disk type is set by default and cannot be changed.
  - Data disk volume: The recommended minimum cluster volume for a single machine is 40 G, and the maximum is 8000 G. The capacity of the ephemeral disk is set by default and cannot be changed.

    - Instance quantity: This indicates the number of instances of all required nodes. A cluster requires at least three instances. However, high availability clusters require at least four, and therefore add one master node.

3. Configure the basic information.

    - Basic information

        - Cluster name: The cluster name can contain Chinese characters, English letters (uppercase and lowercase), numbers, hyphens (-), and underscores (_), with a length of between 1-64 characters.

    - Running logs

        - Running logs: The function for saving running logs is turned on by default. In the default state, you can select the OSS directory as a location to save running logs, but you must have activated OSS before using this function. The cost depends on the number of uploaded files. We recommend that you open the OSS log saving function, which helps in debugging and error screening.
        - Log path: OSS path for saving logs.
        - Uniform Meta Database: This is provided by E-MapReduce to store all Hive metadata in the external database of the cluster. We recommend that you use this function when the cluster uses OSS as the main storage.

    - Permission settings

        - EMR role: This role authorizes E-MapReduce to use other Alibaba Cloud services, such as ECS and OSS.
        - ECS role: This role allows your programs running on the E-MapReduce computing nodes to access cloud services like OSS without providing the Alibaba Cloud AccessKey. E-MapReduce automatically applies for an on-demand AccessKey to authorize access. The AccessKey permission is controlled by this role.

    - Logon settings

        - Remote logon: It is turned on by default to enable security group port 22.
        - Logon password: Set the logon password at the master node. The logon password must contain English letters (both uppercase and lowercase letters), numbers, and special characters (!@#$%^&*) with a length of between 8-30 characters.

- (Optional) Bootstrap actions: Before Hadoop is enabled in the cluster, you can run the customized script. For more information, see *Bootstrap actions*.

Purchase lists and cluster costs

The cost of the cluster is displayed in the Configuration List pane. The price varies with the type of payment. For Subscription clusters, the total expense is shown. For Pay-As-You-Go clusters, the hourly cost is shown.

Confirm creation

Once you have entered all the necessary information, the Create button is highlighted. Click Create to create the cluster.

> Note:
>
> - If your cluster is Pay-As-You-Go, it is created immediately, and you are taken back to the Overview page. Here, your cluster is displayed with the status Initializing. It can take several minutes to finish creating the cluster. After the cluster is created, its status is switched to Idle.
> - Subscription clusters are not created until the order is generated and paid.

Log on to the core node

To log on to the core node, perform the following steps:

1. Switch to the Hadoop account on the master node.

   ```
   su hadoop
   ```

2. Log on to the core node through SSH without a key.

   ```
   ssh emr-worker-1
   ```

3. Get root permissions through the sudo command.

   ```
   sudo vi /etc/hosts
   ```

Failure during cluster creation

If a cluster fails to be created, the message Cluster creation failed is displayed on the cluster list page. If you hover your cursor over the red exclamation point, the reason for the failure is displayed.

You do not need to perform any additional operations because the corresponding computing resources are not created. The cluster is automatically hidden after three days.

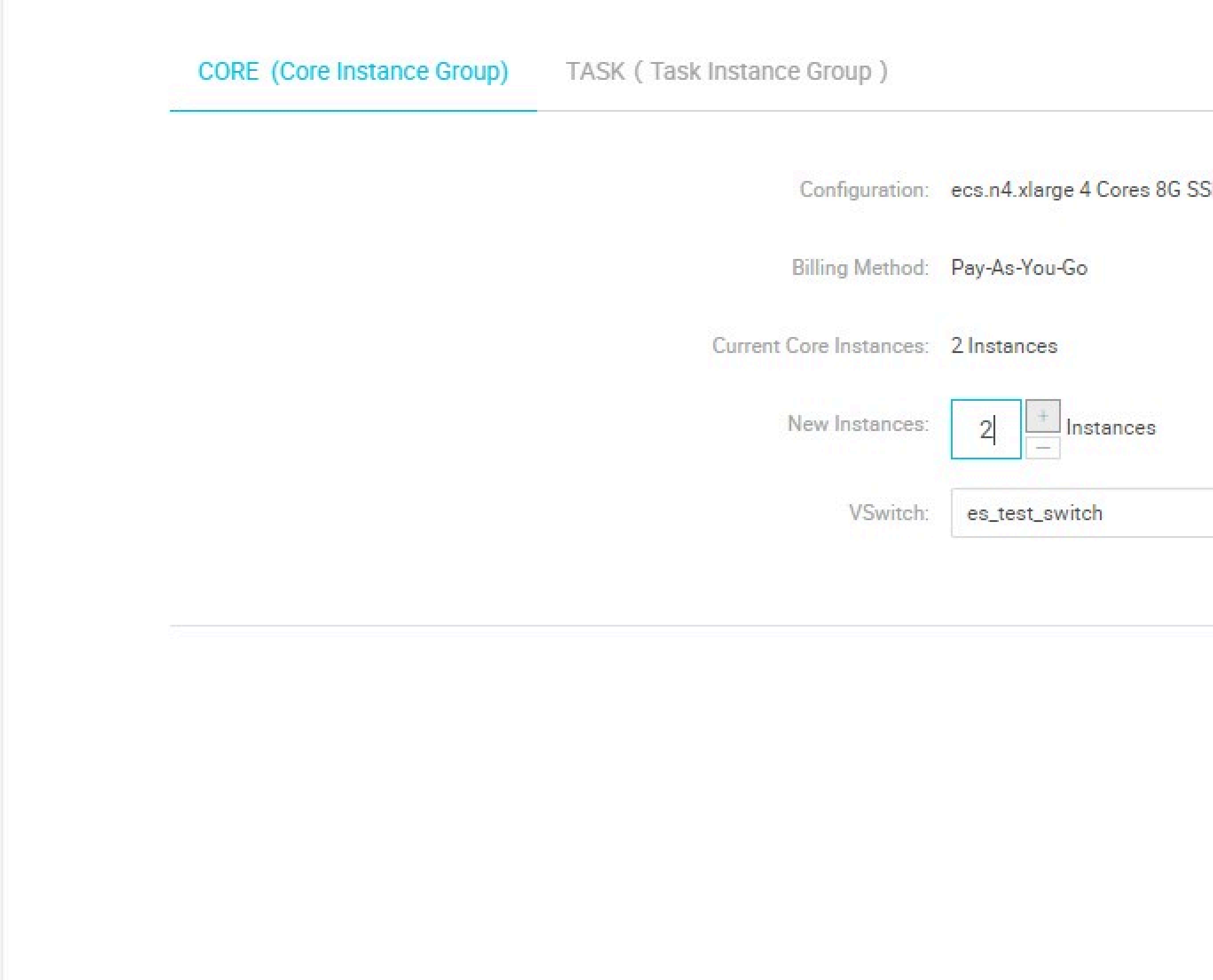
3.2 Expand a cluster

If your cluster does not have enough resources, you can expand it horizontally. Only core and task nodes can be expanded. When expanding a cluster, configurations default to be consistent with the ECS instance purchased previously.

Enter expansion interface

Select the cluster you want to expand from the list of clusters, click More, and select Scale Up/Out. You can also click View Details to the right of the cluster, and click Scale Up/Out in the upper right corner.

Expansion interface



Configuration: ecs.n4.xlarge 4 Cores 8G SS

Billing Method: Pay-As-You-Go

Current Core Instances: 2 Instances

New Instances: 2 Instances

VSwitch: es_test_switch

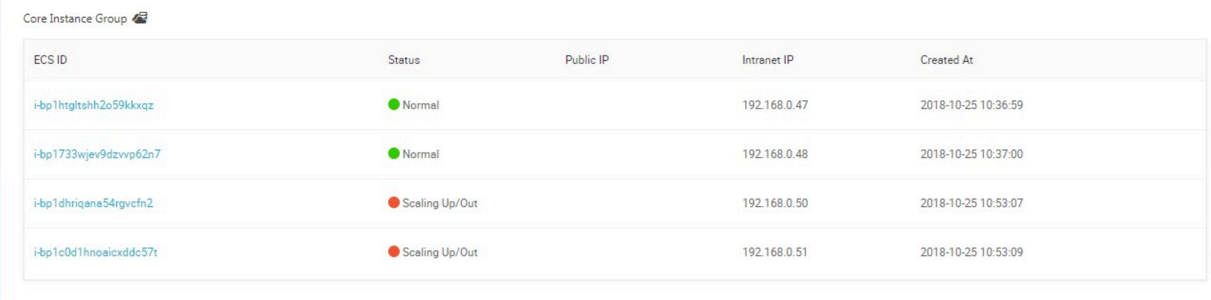
Note:

Although expansion is supported, reduction is not.

- Configuration: Displays the configurations of the current instance.
- Billing Method: Displays the payment method of the current cluster.
- Current Core Instances: Displays the total number of your current core nodes.
- New Instances: Enter the quantity of instances that you want to add.
- VSwitch: Displays the VSwitch of the current cluster.

Expansion status

In the following figure, the statuses of the cluster expansions are shown.



Core Instance Group

ECS ID	Status	Public IP	Intranet IP	Created At
i-bp1htgltshh2o59kkxqz	● Normal		192.168.0.47	2018-10-25 10:36:59
i-bp1733wjev9dzvvp62n7	● Normal		192.168.0.48	2018-10-25 10:37:00
i-bp1dhriqana54rgvcfn2	● Scaling Up/Out		192.168.0.50	2018-10-25 10:53:07
i-bp1c0d1hnoaicxddc57t	● Scaling Up/Out		192.168.0.51	2018-10-25 10:53:09

To view the expansion status of a cluster, click Core Instance Group (CORE) in the Cluster Overview panel. Nodes that are being expanded are displayed as Scaling Up/Out. When the status of an ECS instance changes to Normal, ECS has been added to the cluster and can now provide services.

Change password

After you expand a cluster, you can log on to the expanded node with SSH and change your root password. To do so, follow these steps:

1. Log on the master host with SHH using the following command and obtain the public IP of the master cluster in the *Cluster Overview* panel.

```
ssh root@ip.of.master
```

2. Switch to the hadoop user.

```
su hadoop
```

3. Log on to the expanded node and obtain the intranet IP of the expanded node in the *Cluster Overview* panel.

```
ssh ip.of.worker
```

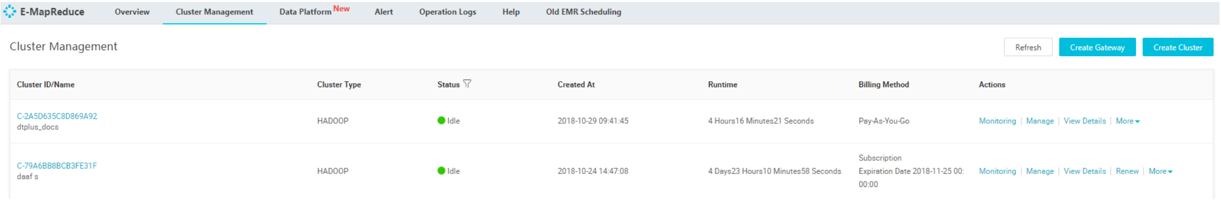
4. Change your root password using the following command.

```
sudo passwd root
```

3.3 Cluster list

The Cluster Management page, displays basic information about all of your clusters.

On the Cluster Management page, information of clusters are displayed as follows:



Cluster ID/Name	Cluster Type	Status	Created At	Runtime	Billing Method	Actions
C-2A5D635C8D869A92 dtplus_docs	HADOOP	● Idle	2018-10-29 09:41:45	4 Hours16 Minutes21 Seconds	Pay-As-You-Go	Monitoring | Manage | View Details | More
C-79A6BB8BCB3FE31F daaf s	HADOOP	● Idle	2018-10-24 14:47:08	4 Days23 Hours10 Minutes58 Seconds	Subscription Expiration Date 2018-11-25 00:00:00	Monitoring | Manage | View Details | Renew | More

The items in the cluster list are as follows:

- Cluster ID/Name: The ID and name of a cluster. Move your cursor over a cluster's name to modify it.
- Cluster Type: Hadoop is the only cluster type available.
- Status: The status of a cluster. For more information, see *Cluster statuses*. If a cluster experiences an abnormality, such as a creation failure, prompt information appears on the right. If you hover your cursor over it, you can view detailed error information. You can also sort the statuses by clicking Status.
- Created At: Time at which a cluster was created.
- Runtime: The time from the point of creation to the current time. Once the cluster is released, the timing is terminated.
- Billing Method: The billing method of the cluster.
- Actions: Operations that can be performed on clusters, including the following:

- Monitoring: Monitors the CPU usage rate, memory capacity, and disk capacity of E-MapReduce clusters to help users monitor the running status of the cluster.
- Manage: Enters the Clusters and Services panel.
- View details: Enters the Cluster Overview panel and view detailed information after the cluster is created.
- More:
  - Scale Up/Out: Expands the cluster.
  - Release: Releases a cluster. For more information, see *Release a cluster*.
  - Restart: Restarts a cluster.

3.4 Release a cluster

Pay-As-You-Go clusters can be released on the Cluster Management page. Only those with the following statuses can be released:

- Creating
- Running
- Idle

Common release

Before releasing a cluster, you are prompted to confirm the operation. Once the release is confirmed, the following occurs:

- All jobs in the cluster are forcibly terminated.
- If you have selected to save logs to OSS, all current job logs are saved to OSS. This may take several minutes.
- Clusters are released. Depending on their size, this may take seconds or minutes. ECS clusters are billed before they are released.

> ⚠ Warning:
> To save money, make sure to release a cluster before a new billable hour starts.

Forcible release

If you no longer need logs and want to immediately terminate running clusters, perform a forcible release. Logs are not collected and clusters are released directly.

Cluster release failure

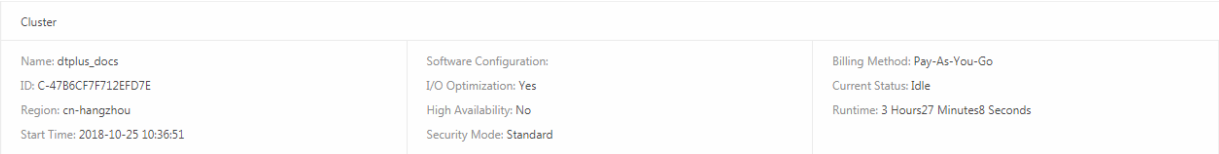
Due to system errors or other causes, clusters may fail to be released. If a failure occurs, E-MapReduce starts background protection until the cluster is finally released .

3.5 Cluster details

Cluster details display detailed information about a cluster.

Detailed information is provided in the following areas: cluster, software, network, and host.

Cluster



Cluster

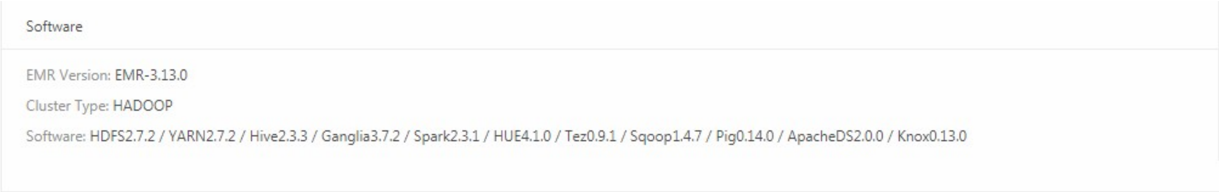
Name: dtplus_docs
ID: C-47B6CF7F712EFD7E
Region: cn-hangzhou
Start Time: 2018-10-25 10:36:51

Software Configuration:
I/O Optimization: Yes
High Availability: No
Security Mode: Standard

Billing Method: Pay-As-You-Go
Current Status: Idle
Runtime: 3 Hours27 Minutes8 Seconds

- Name: The name of a cluster.
- ID: The instance ID of a cluster.
- Region: The region where a cluster is located.
- Start Time: The time at which a cluster is created.
- Software Configuration: Software configurations.
- I/O Optimization: Whether the I/O optimization setting is enabled.
- High Availability: Whether high-availability clusters are enabled.
- Security Mode: Software in clusters is started in Kerberos secure mode. For more information about Kerberos, see *Introduction to Kerberos*.
- Billing Method: Cluster billing method.
- Current Status: For more information, see *Cluster statuses*.
- Runtime: The time from the point of creation to the current time.
- Bootstrap: The names, paths, and parameters of all configured bootstrap actions are listed here.
- ECS Role: When your program runs on an E-MapReduce compute node, you can access the related Alibaba Cloud services, such OSS, without an AccessKey. E-MapReduce automatically requests a temporary AccessKey to authorize this access. The permission control of this temporary AccessKey is controlled by this role.

Software



Software

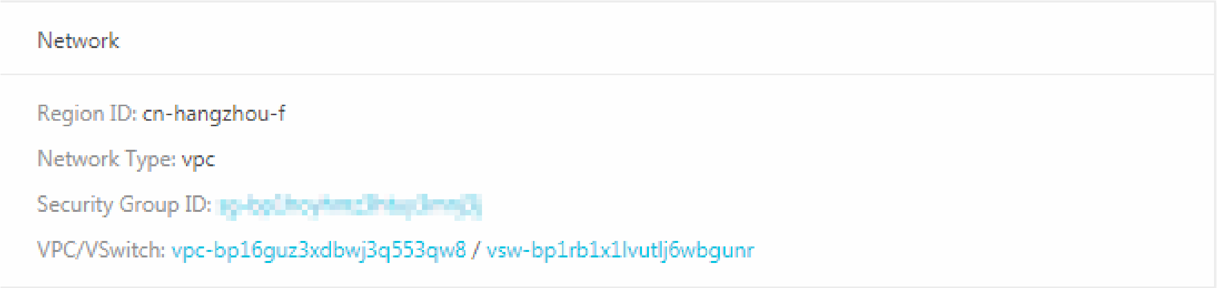
EMR Version: EMR-3.13.0

Cluster Type: HADOOP

Software: HDFS2.7.2 / YARN2.7.2 / Hive2.3.3 / Ganglia3.7.2 / Spark2.3.1 / HUE4.1.0 / Tez0.9.1 / Sqoop1.4.7 / Pig0.14.0 / ApacheDS2.0.0 / Knox0.13.0

- Main Version: The main version of E-MapReduce.
- Cluster Type: The selected cluster type.
- Software: All application programs installed are listed here with their versions, such as HDFS2.7.2, Hive 2.3.3, or Spark 2.3.1.

Network



Network

Region ID: cn-hangzhou-f

Network Type: vpc

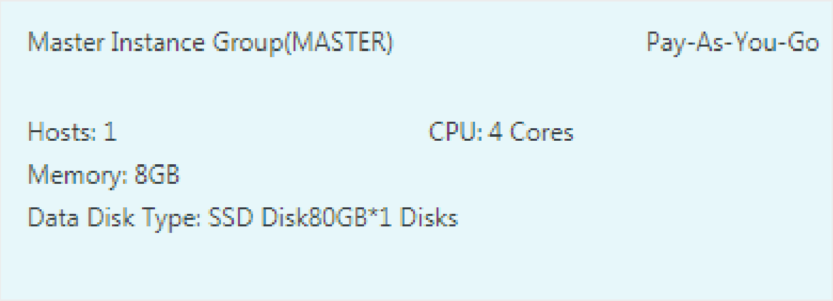
Security Group ID: [redacted]

VPC/VSwitch: vpc-bp16guz3xdbwj3q553qw8 / vsw-bp1rb1x1lvutlj6wbgunr

- Region ID: The region where a cluster is located, such as cn-hangzhou-b, which is the same as ECS.
- Network Type: The network type of a cluster.
- Security Group ID: The ID of the security group that a cluster joined.
- VPC/VSwitch: The VPC and VSwitch IDs of a cluster.

Host

- Master Instance Group (Master): Configurations of all master nodes.



Master Instance Group(MASTER) Pay-As-You-Go

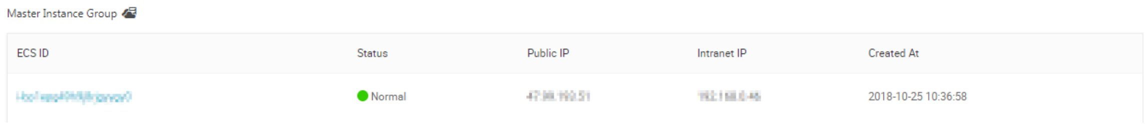
Hosts: 1 CPU: 4 Cores

Memory: 8GB

Data Disk Type: SSD Disk80GB*1 Disks

  - Hosts: The number of current nodes. During the creation process, the number of current nodes is less than the number of nodes you applied for until the creation is complete.

- CPU: The number of cores in a node's CPU.
- Memory: Memory capacity of a node.
- Data Disk Type: Data disk type and capacity of a node.
- ECS ID: The ID of the ECS instances purchased.

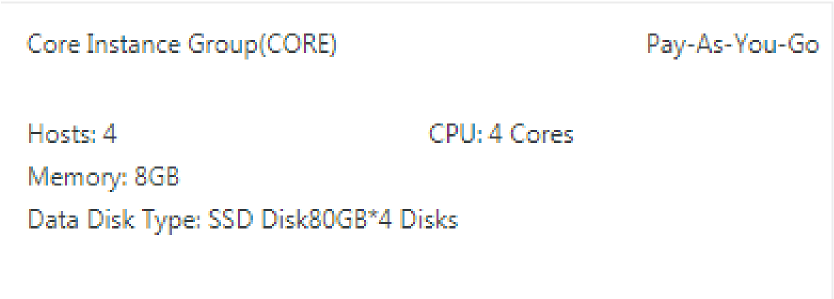


Master Instance Group

ECS ID	Status	Public IP	Intranet IP	Created At
[redacted]	● Normal	[redacted]	[redacted]	2018-10-25 10:36:58

- Status: Includes Creating, Normal, Expanding, and Released.
- Public IP : The public IP of master nodes.
- Intranet IP: The internal network IP of the machine that can be accessed by all nodes in the cluster.
- Created At: The creation time of the ECS instance purchased.

· Core Instance Group (Core) : Configurations of all core nodes.



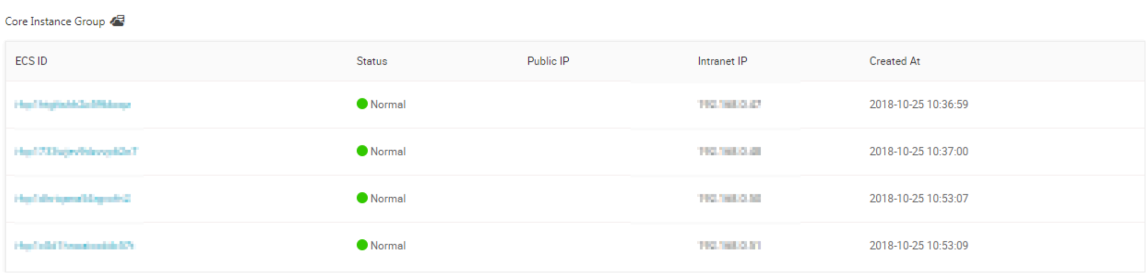
Core Instance Group(CORE) Pay-As-You-Go

Hosts: 4 CPU: 4 Cores

Memory: 8GB

Data Disk Type: SSD Disk80GB*4 Disks

- Hosts: The number of current nodes. This is the same as the number of the nodes you applied for.
- CPU: The number of cores in a node's CPU.
- Memory: Memory capacity of a node.
- Data Disk Type: Data disk type and capacity of a node.
- ECS ID: The ID of the ECS instances purchased.



Core Instance Group

ECS ID	Status	Public IP	Intranet IP	Created At
[redacted]	● Normal		[redacted]	2018-10-25 10:36:59
[redacted]	● Normal		[redacted]	2018-10-25 10:37:00
[redacted]	● Normal		[redacted]	2018-10-25 10:53:07
[redacted]	● Normal		[redacted]	2018-10-25 10:53:09

- Status: Includes Creating, Normal, Expanding, and Released.
- Intranet IP: The internal network IP of the machine that can be accessed by all nodes in the cluster.

- Created At: The creation time of the ECS instance purchased.

3.6 User management

User management allows you to manage the accounts required to create services on specified clusters. E-MapReduce currently supports the creation of two types of accounts: Knox and Kerberos. This topic explains how to manage Knox accounts.

Create a RAM account

1. Log on to the *Alibaba Cloud E-MapReduce console* and go to the Cluster Management page.
2. Click Manage on the right side of the target cluster ID.
3. In the navigation panel on the left, click User Management.
4. In the upper-right corner of the page, click Create RAM User.

Add a Knox account

1. In the User Management page, select the account you want to add to a cluster, and then click Set Knox Account Password in the Actions column.
2. In the Add Knox User dialog box, enter a password to use for logon and click OK.
3. Refresh the User Management page. When Synchronized is displayed in the Knox Account column, you have successfully added the Knox account.

   You can then sign in to Knox using the User Name and the password set in *Step 2*.

Delete a Knox account

1. In the User Management panel, select the account you want to delete from a cluster, and then click Delete Knox Account in the Actions column.
2. Refresh the User Management page. When Unsynchronized is displayed in the Knox Account column, you have successfully deleted the Knox account.

FAQs

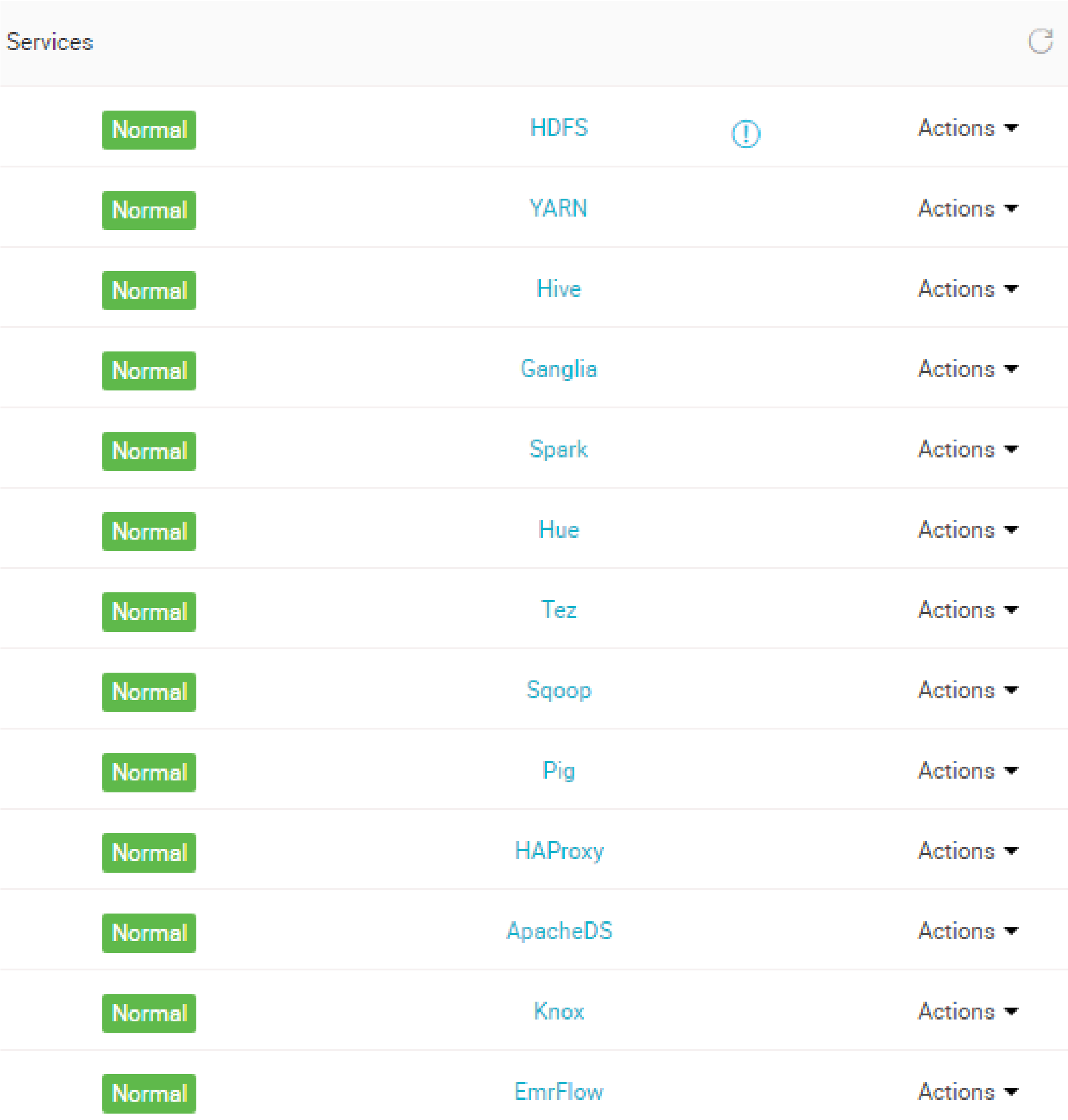
- Different clusters cannot share the same Knox account. For example, Knox account A that you added to cluster-1 cannot be used in cluster-2. If you want to use Knox account A in cluster-2, you must re-add account A to cluster-2.
- If the message An error occurred while synchronizing the status is displayed when you add a Knox account, click Retry to add it again.

- If you try to add an account multiple times but it fails each time, click Clusters and Services on the left side of the page to check if ApacheDS is stopped. If it is, start ApacheDS and go back to User Management to try again.

3.7 Service list

HDFS, YARN, and other services are listed on the Clusters and Services tab of the cluster management page. You can view the running statuses of these services.

The service list shows the following information.



Services

Normal	HDFS	ⓘ	Actions ▾
Normal	YARN		Actions ▾
Normal	Hive		Actions ▾
Normal	Ganglia		Actions ▾
Normal	Spark		Actions ▾
Normal	Hue		Actions ▾
Normal	Tez		Actions ▾
Normal	Sqoop		Actions ▾
Normal	Pig		Actions ▾
Normal	HAProxy		Actions ▾
Normal	ApacheDS		Actions ▾
Normal	Knox		Actions ▾
Normal	EmrFlow		Actions ▾

A service's running status is only showed for clusters that are in the Idle or Running status. Services that are not checked when creating a cluster, such as Storm, are not listed.

Click a service to view the corresponding tabs, including Status, Component Topology, Configuration, and Configuration Change History. The status can either be Normal or Error. If the status of a service on a node is Error, you can use the master node to log on to the node and check the service process.

3.8 Cluster scripts

To modify a cluster's running environment, you can install third-party software in the cluster. This mostly applies to Subscription clusters. After a cluster is created, the cluster script feature allows you to select nodes in batches and run your specified script to meet your own requirements.

Role of a cluster script

A cluster script is similar to a bootstrap action. After creating a cluster, you can install packages that are unavailable to your cluster before, such as:

- YUM, which is used to install software already provided.
- Public software packages from the public network.
- Software that allows you to read your data from OSS.
- Services like Flink or Impala, which require a more complex script.

We strongly recommend that you test the cluster script on a node first. After the script has been verified, you can perform operations on the whole cluster.

Create and run a cluster script

1. A cluster script can run on an idle cluster or a running cluster. On the Cluster Management page, click View Details next to the cluster you want to run a script on.
2. On the menu on the left, click Cluster Scripts to enter the cluster script execution interface.
3. In the upper-right corner, click Create and Run to enter the creation interface.
4. Enter configurations in the script creation pane, select a node for execution, and click OK.

To update the running status of the cluster script on each node, click Refresh. To display the running status of the cluster script on each node, click View Details.

Cluster scripts are applicable for long-standing clusters and can only run on available clusters that are idle or running. To initialize on-demand clusters, perform a bootstrap action.

The cluster script feature downloads a script from the OSS and runs it on a specified node. If the returned value is 0, the execution has failed. If the execution fails, you can log on to the node to check the running log. The running log for each node is located at `/var/log/cluster-scripts/clusterScriptId`. If the cluster is configured with an OSS log directory, the running log is also uploaded to `osslogpath/clusterId/ip/cluster-scripts/clusterScriptId`.

By default, the root account is used to run the specified script. In the script, you can use `su hadoop` to switch to a Hadoop account.

A cluster script can successfully run on some nodes, but fail on others. For example , restarting a node can lead to a failure in script operation. After resolving the error, you can run the cluster script again. After a cluster is expanded, you can specify the expanded node for separate execution of the cluster script.

Only one cluster script can run on a cluster at a time. For each cluster, you can retain up to ten cluster script records. If you have ten records and want to create a new cluster script, you must first delete the previous records.

Script example

For a script similar to a bootstrap action, you can specify the file in the script that you want to download from OSS. In the following example, the file `oss://yourbucket/myfile.tar.gz` is downloaded and decompressed to the directory `/yourdir:`

```
#! #!/bin/bash
osscmd --id=<yourid> --key=<yourkey> --host=oss-cn-hangzhou-internal.
aliyuncs.com get oss://<yourbucket>/<myfile>.tar.gz ./<myfile>.tar.gz
mkdir -p /<yourdir>
tar -zxvf <myfile>.tar.gz -C /<yourdir>
```

OSSCMD is pre-installed on the node and can be called directly to download the file.

> Note:
>
> The OSS host address can be an intranet address, Internet address, or VPC network address. If a classic network is used, you must specify an intranet address. If

the network is located in Hangzhou, the intranet address is oss-cn-hangzhou-internal.aliyuncs.com. If a VPC network is used, you must specify a domain name that can be accessed from the VPC intranet. If the network is located in Hangzhou, the domain name is vpc100-oss-cn-hangzhou.aliyuncs.com.

Additional system software packages can be installed on the script using YUM. For example, ld-linux.so. 2:

```
#! /bin/bash
yum install -y ld-linux.so. 2
```

3.9 Cluster renewal

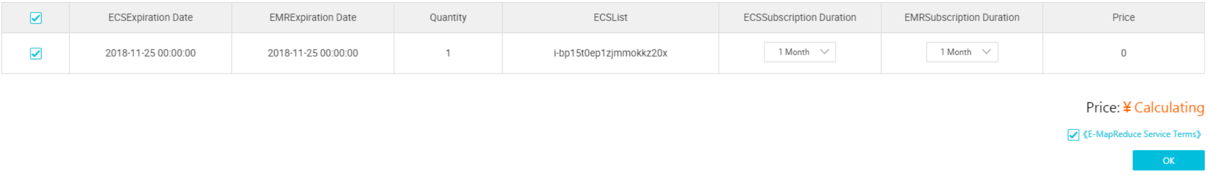
When your Subscription cluster is about to expire, you have to renew it in order to continue using E-MapReduce cluster services. Cluster renewal includes the renewal of E-MapReduce services and ECS instances.

Enter the renewal page

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the page, click Cluster Management.
3. In the cluster list, select the cluster that you want to renew.
4. To the right the cluster, click Renew to enter the cluster renewal page.

Renewal page

As shown in the following figure, the renewal page contains a number of columns, which are detailed below.



☑	ECSExpiration Date	EMRExpiration Date	Quantity	ECSList	ECSSubscription Duration	EMRSubscription Duration	Price
☑	2018-11-25 00:00:00	2018-11-25 00:00:00	1	i-bp15t0ep1zjmmokkz20x	1 Month	1 Month	0

Price: ¥ Calculating

☑ 《E-MapReduce Service Terms》

- ECS Expiration Date: The expiration date of an ECS instance.
- EMR Expiration Date: The expiration date of E-MapReduce services.
- Quantity: The number of machines for instance groups.
- ECS List: The ECS instance ID of the machine in the cluster.
- ECS Subscription Duration: The duration of your ECS subscription. You can select from one to nine months, or one, two, or three years.

- EMR Subscription Duration: The duration of your E-MapReduce subscription. We recommend that you keep it consistent with ECS.
- Price: The price of the renewal of E-MapReduce services and ECS instances.

Make a payment

> Notice:
> The fees consist of the combined costs of ECS renewal and E-MapReduce service products. If there are unpaid orders in the cluster list, you cannot expand or renew any clusters.

1. Click OK to view the prompt box for a successful order placement.
2. Click Go to the payment page. The payment page displays the total amount to be paid as well as the order details.
3. Click Confirm payment.
4. After you make the payment, click Payment completed to return to the cluster list page.

If an expired cluster is successfully renewed, its expiration date is updated to reflect the renewal. If an expired ECS instance is successfully renewed, its expiration date is usually updated around three to five minutes later.

If you confirm the renewal, but fail to pay for it, Cancel order and Make the payment are displayed on the right side of the cluster. Click Make the payment to complete the payment or Cancel to cancel the renewal.

3.10 Security groups

Security groups created in E-MapReduce can be used during the creation of clusters.

Only port 22 is accessible in clusters created by E-MapReduce. We recommend that you divide ECS instances by function, and put them into different user security groups. For example, name the security group of E-MapReduce E-MapReduce security group, and name the security group that you create User security group. Each security group is provided with unique access control as required.

If you need to link a security group with a cluster that has already been created, follow these steps.

Add an E-MapReduce cluster to an existing security group

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the page, click Cluster Management.
3. Click View Details.
4. In the Network tab, find Security Group ID and click the ID link.
5. In the menu on the left, click Instances in Security Group to view the security group names of all ECS instances.
6. Log on to the *Alibaba Cloud ECS console* and click Security Group in the navigation panel on the left to find the security group as viewed in the preceding step.
7. Click Manage Instances in a security group and view ECS instance names that start with emr-xxx. These are the ECS instances in an E-MapReduce cluster.
8. Select all of these instances, click Move to security group, and then select a security group to move the E-MapReduce cluster to.

Add an existing cluster to an E-MapReduce security group

1. Find the security group where the existing cluster is located.
2. Repeat the preceding operations, and move to the E-MapReduce security group.
3. Select scattered machines in the ECS console directly and move the clusters to the E-MapReduce security group in batches.

Security group rules

When an ECS instance is in several different security groups, the security group rules are subject to the OR relationship. For example, only port 22 of the E-MapReduce security group is accessible, whereas all ports of the User security group are accessible. After the E-MapReduce cluster is added to the User security group, all ports of the machine in E-MapReduce are open.

3.11 Access links and ports

This section serves as a quick portal for components.

When a cluster is created, several domain names are bound to it by default so that you can access the following open source components:

- HDFS
- YARN
- Spark

- Hue
- Ganglia

In the Cluster Management page, click Manage or View Details, and find the Access Links and Ports tab where you can see links for these components.

By default, there is no user name or password required for access, which means that the access request cannot pass the HTTP authentication. Click Set User Name and Password to set a user name and password to access your component interface.

Only one user name and one password can be used. A new user name and password will always replace previous ones.

Note:

Currently, this function is only supported by version 2.3 and later.

3.12 Node upgrade

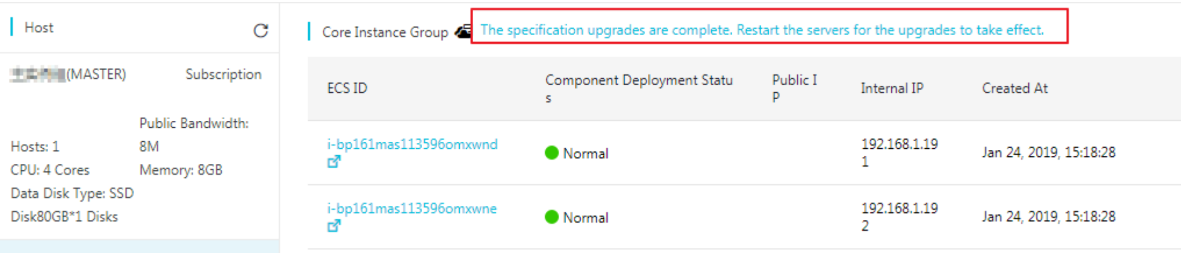
In real scenarios, the CPU or memory of a cluster node, especially master nodes, may be insufficient.

Note:

- Only E-MapReduce Subscription clusters can be upgraded.
- Ephemeral disk instances (such as instance types of D1 and I2) do not support upgrade configurations. You can only add nodes.
- Non-ephemeral disk instances only support upgrade configurations, and do not support downgrade configurations.

Procedure

1. In the Cluster Management panel, select a cluster, and click View Details to go to the Cluster Overview pane.
2. In the upper-right corner, click Renew & Upgrade > Upgrade.
3. Configure the nodes that you want to upgrade.
4. Click OK.
5. Pay for your order.
6. Return to the Cluster Management page, refresh the page to make sure that the node configuration has become the target specification. The following figure displays the upgraded node information.



Host

[redacted](MASTER) Subscription

Hosts: 1
CPU: 4 Cores
Data Disk Type: SSD
Disk80GB*1 Disks

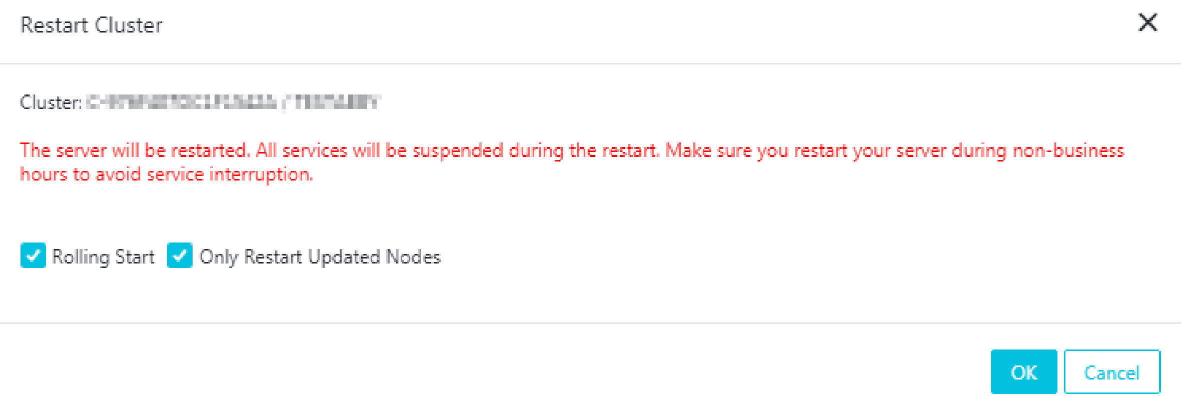
Public Bandwidth: 8M
Memory: 8GB

Core Instance Group

The specification upgrades are complete. Restart the servers for the upgrades to take effect.

ECS ID	Component Deployment Status	Public IP	Internal IP	Created At
i-bp161mas113596omxwnd	● Normal		192.168.1.191	Jan 24, 2019, 15:18:28
i-bp161mas113596omxwne	● Normal		192.168.1.192	Jan 24, 2019, 15:18:28

7. Click The specification upgrades are complete. Restart the server for the upgrades to take effect to view to the following figure.



Restart Cluster

Cluster: [redacted]

The server will be restarted. All services will be suspended during the restart. Make sure you restart your server during non-business hours to avoid service interruption.

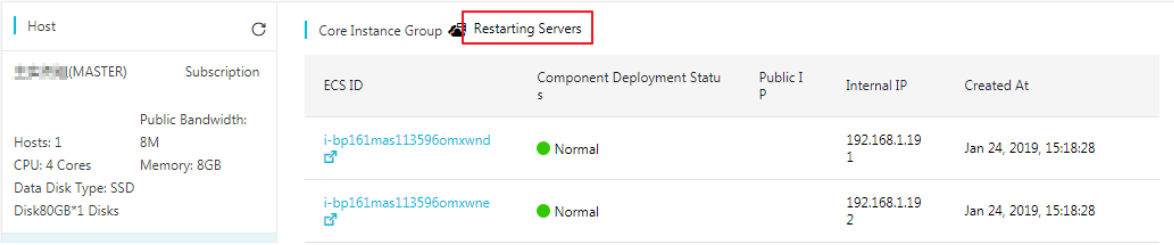
☑ Rolling Start ☑ Only Restart Updated Nodes

8. Click OK.

Notice:

- If you restart a cluster, you also restart ECS instances. ECS instance data is unavailable during restarting.
- Rolling Start
  - If selected, only ECS instances you select is restarted. Instances are restarted one after another. Each restart takes about five minutes.
  - If not selected, all ECS instances are restarted.
- Only Restart Updated Nodes
  - Configuration-updated nodes are nodes whose disks or configurations have been expanded or upgraded.
  - If selected, only configuration-updated nodes are restarted. For example, if you expanded a core group node, but did not expand a master group node's disk or upgrade its configurations, only the ECS instance in the core group is restarted, and the ECS instance in the master group will not be restarted.
  - If not selected, all nodes (all instances in the cluster) are restarted.

9. During the restart process, the prompt Restarting Servers in the following figure is displayed for the corresponding node group (such as a core group).



Host

[redacted](MASTER) Subscription

Hosts: 1
CPU: 4 Cores
Data Disk Type: SSD
Disk80GB*1 Disks
Public Bandwidth: 8M
Memory: 8GB

Core Instance Group Restarting Servers

ECS ID	Component Deployment Status	Public IP	Internal IP	Created At
i-bp161mas113596omxwnd	● Normal		192.168.1.191	Jan 24, 2019, 15:18:28
i-bp161mas113596omxwne	● Normal		192.168.1.192	Jan 24, 2019, 15:18:28

10. Log on to the EMR cluster to check upgrades. After the prompt in step 9 is not displayed, all upgraded configurations take effect.

11. If you just upgraded the CPU and did not upgrade the memory, ignore this step. If you upgraded the memory, you need to modify cluster configurations so that YARN can use new resources.

    a. On the Clusters and Services pane, click YARN.
    b. Click the Configuration tab, locate `yarn.nodemanager.resource.memory-mb` and `yarn.scheduler.maximum-allocation-mb` two configuration items, and change the value of the two configuration items to machine memory * 0.8. The unit is MB. For example, in this new configurations, the memory is 32 GB, configure the following value 26214 (32*1024*0.8).

    ```
    yarn.nodemanager.resource.memory-mb=26214
    yarn.scheduler.maximum-allocation-mb=26214
    ```

    c. In the upper right corner, click Save.
    d. In the upper right corner, click Actions > CONFIGURE All Components.
    e. In the upper right corner, click View Operation Logs and wait for the status of the CONFIGURE YARN operation type to Successful.
    f. In the upper right corner, click Actions > RESTART All Components.
    g. Click View Operation Logs, wait for the status of the RESTART YARN operation type to Successful, and new resources can be used by YARN.

3.13 Disk expansion

In the event that you do not have enough data storage space, cluster disks of E-MapReduce 3.11.0 or later can be expanded on the cluster overview page.

Enter the expansion interface

On the Cluster Management page, find the target cluster. Click View Details, and in the upper-right corner, click Disk Capacity Expansion.

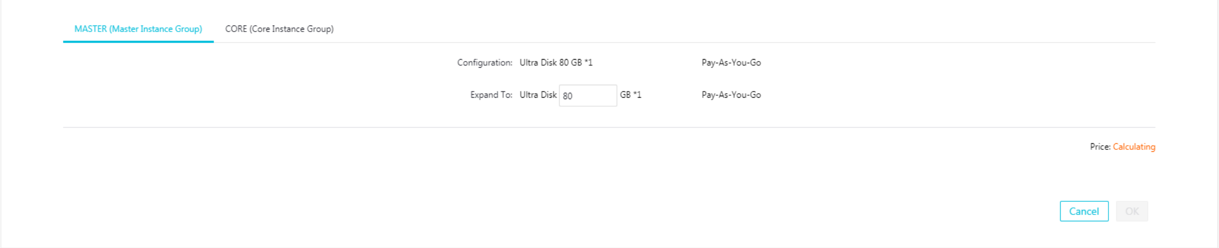
Select the instance group that needs to be expanded.

Notice:

Make sure your account balance is sufficient. The fee of expanding a data disk is deducted automatically. When your account balance is insufficient the, the process of expanding a data disk will be interrupted.

Expansion interface

The following figure shows the disk expansion interface displayed.



Configuration: Ultra Disk 80 GB *1 Pay-As-You-Go

Expand To: Ultra Disk 80 GB *1 Pay-As-You-Go

Price: Calculating

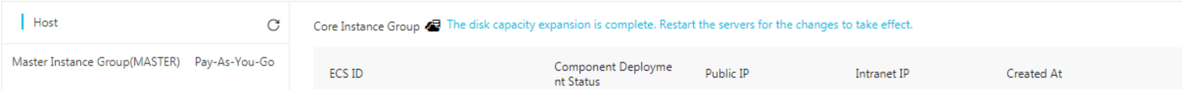
- Configuration: Data disk configurations of the current instance group.
- Expand to: Data disk size after the disk is expanded.

Note:

- E-MapReduce currently supports the expansion disks, but not their reduction.
- The expansion of disks is only supported for data disks.

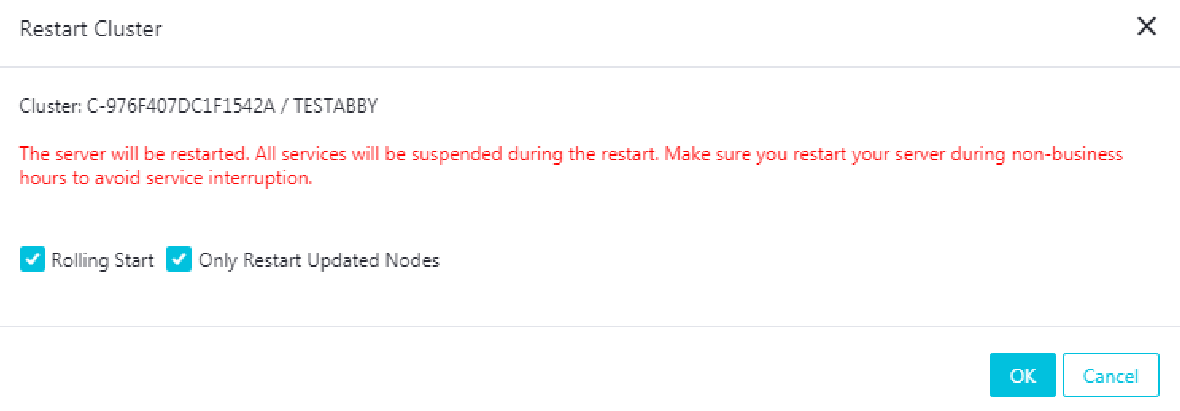
Restart a cluster

1. After completing the disk expansion, the corresponding core instance group displays The disk capacity expansion is complete. Restart the servers for the changes to take effect, as shown in the following figure.



The disk capacity expansion is complete. Restart the servers for the changes to take effect.

2. Click The disk capacity expansion is complete. Restart the servers for the changes to take effect, as shown in the following figure.



Restart Cluster

Cluster: C-976F407DC1F1542A / TESTABBY

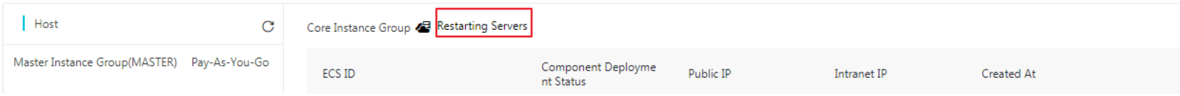
The server will be restarted. All services will be suspended during the restart. Make sure you restart your server during non-business hours to avoid service interruption.

☑ Rolling Start ☑ Only Restart Updated Nodes

> **Note:**
> If you restart a cluster, you also restart ECS instances. ECS instance data is unavailable during restarting.

- Rolling Start
  - If selected, only ECS instances you select is restarted. Instances are restarted one after another. Each restart takes about five minutes.
  - If not selected, all ECS instances are restarted.
- Only Restart Updated Nodes
  - Configuration-updated nodes are nodes whose disks or configurations have been expanded or upgraded.
  - If selected, only configuration-updated nodes are restarted. For example, if you expanded a core group node, but did not expand a master group node's disk or upgrade its configurations, only the ECS instance in the core group is restarted, and the ECS instance in the master group will not be restarted.
  - If not selected, all nodes (all instances in the cluster) are restarted.

3. During the restart process, the prompt in the following figure is displayed for the corresponding node group (such as a core group).



Host

Master Instance Group(MASTER) Pay-As-You-Go

Core Instance Group Restarting Servers

ECS ID	Component Deployment Status	Public IP	Intranet IP	Created At

4. After the prompt in step 3 is no longer displayed, all data disks are expanded and available to use.

Node data disk expansion

1. Log on to the E-MapReduce console, find the target cluster, and click View Details to go to the cluster overview page.
2. View the `ECS ID` of the core node in the cluster you want to expand, such as `i-bp1bsithym5hh9h93xxx`. All the node disks in the cluster are expanded uniformly by default to make sure the disk spaces of all nodes in the cluster are consistent.
3. Copy the `ECS ID` and go to the ECS console. Select Instance Details in the pane on the left, then select the instance ID and enter the ECS ID in the search box. Note that the same region must be selected.
4. When the corresponding ECS node is found, click Manage, and click Disks in the navigation pane on the left.
5. Expand the data disks. You must repeat the following steps to scale out each disk, because you cannot expand multiple disks in batches.
6. Expand all the disks in the ECS console and then restart the nodes.

   See *Linux _ Resize a data disk* for more information on expanding data disks.

   > Note:
   >
   > If the unmount operation fails, YARN and HDFS services must be disabled in the cluster. Disk1 operations may encounter unmount failures because of log writing by ilogtail. ilogtail must be temporarily killed using the command `sudo pgrep ilogtail | sudo xargs kill`. ilogtail can then be recovered by restarting the node.

7. You can then see that all disks are expanded using the `df -h` command on the node.

Node system disk expansion

Cluster disks of E-MapReduce 3.11.0 or earlier can be scaled out in the ECS console.

1. In the E-MapReduce console, enter the details page of the cluster you want to expand.
2. View the `ECS ID` of the master node in the cluster you want to expand, such as `i-bp1bsithym5hh9h93xxx`. All the node disks in the cluster are expanded uniformly by default to make sure the disk spaces of all nodes in the cluster are consistent.

3. Copy the `ECS ID` and go to the ECS console. Select Instance Details in the pane on the left, then select the instance ID and enter the ECS ID in the search box. Note that the same region must be selected.
4. When the corresponding ECS node is found, click Manage, and click Disks in the navigation pane on the left.
5. Find the system disk (there is only one).
6. See *Increase system disk size* for more information on expanding system disks.

> Notice:
>
> - For non-HA clusters, while you are expanding the system disk, the cluster becomes unavailable.
> - ECS disk operations may have changed the */etc/hosts* file in the node. After the disk has been expanded, the file must be repaired. SSH log-on-free is also damaged, but the service remains unaffected and can be repaired manually.

3.14 Convert your billing method

There are two billing methods available for E-MapReduce: Pay-As-You-Go and Subscription. We recommend that you use Pay-As-You-Go, as it is more cost-effective. If you are sure that you want to use E-MapReduce for a longer period of time, you can change to the Subscription method.

Convert Pay-As-You-Go to Subscription

You can select the Pay-As-You-Go method at the beginning during your E-MapReduce trial and at any time after that. In the Cluster Overview panel, click the Switch to Subscription button to switch your payment method from Pay-As-You-Go to Subscription. The payment method for the entire cluster will be converted.

Convert Subscription to Pay-As-You-Go

You cannot currently convert from Subscription to Pay-As-You-Go.

3.15 Remove abnormal nodes

If an ECS node in an E-MapReduce cluster is abnormal, and if you do not need this node and want to remove it, you can do so.

To remove an abnormal node, follow these steps:

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the page, click Cluster Management.
3. Click the ID link of the cluster that includes the node that you want to remove.
4. In the left-side navigation pane, click Hosts.
5. Find the instance that you want to remove, and in the action column, click Remove.

   ECS instances can only be removed when they are in a stopped or released status.
6. Click OK to remove the instance.

4 Alarm management

4.1 Cluster alarm management

To help you monitor cluster operations, CloudMonitor offers multiple monitoring metrics for E-MapReduce clusters, including CPU idleness, memory capacity, and disk capacity. It also allows you to set alarm rules for these monitoring metrics. Once an alarm rule is triggered, CloudMonitor promptly notifies the contacts in the notification group, allowing you to deal with the problem in time.

Configure alarm rules

To set an alarm rule for an E-MapReduce cluster, perform the following steps:

1. Log on to *CloudMonitor console*.
2. In the navigation pane on the left, click Cloud Service Monitoring > E-MapReduce to go to the E-MapReduce monitoring list page.
3. Click the Alarm Rules tab.
4. In the upper-right corner of the page, click Create Alarm Rule to set an alarm rule for a monitoring metric.
5. In the Related Resource area, set Products and Resource Range.
   - Products: Select E-MapReduce in the drop-down list.
   - Resource Range: The scope of the alarm rule, which is divided into three areas: all resources, application group, and cluster. When all resources is selected, the maximum number of resources that can be monitored is 1,000. If you have more than 1,000 resources, an alarm may not occur when the threshold is reached. We recommend that you set the alarm rules first before using application group to divide resources according to your needs.
     - All Resources: Indicates that the rule works on all E-MapReduce instances under the account. For example, if you set an alarm for when the CPU usage becomes greater than 80%, as long as there is an instance whose CPU usage is greater than 80% in the account, the rule is triggered.
     - ***Application Group***: Indicates that the rule works on all instances of a certain application group. For example, if you set an alarm for when the CPU usage

becomes greater than 80%, as long as there is a host whose CPU usage is greater than 80% in this group, the rule is triggered.

- Cluster: Indicates that the rule only works on a specific E-MapReduce cluster. For example, if you set an alarm for when the CPU usage becomes greater than 80%, as long as there is an instance whose CPU usage is greater than 80% in this cluster, the rule is triggered.

6. Configure the Set Alarm Rules area.

   - Alarm Rule: Sets the name of an alarm rule.
   - Rule Description: The part of the alarm rule that defines whether the conditions have been met to trigger the alarm rule. If they have been met, the alarm rule is triggered. For example, if the rule is set as an average CPU usage within 1 minute of greater than or equal to 90%, the rule is triggered if the average CPU usage within 1 minute is greater than or equal to 90%. For more information about the monitoring metrics in E-MapReduce clusters, see *E-MapReduce monitoring*.
   - Role: By default, any role is applicable.

     If you click Add Alarm Rule, you can set multiple alarm rules (which are billed as such). If one of the rules is triggered, the system sends notifications to the notification group.
   - Mute for: The interval at which alarm notifications are sent again in the event that the monitoring metrics are not returned to normal after an alarm occurs.
   - Triggered when threshold is exceeded for: Number of times the rule is triggered to send alarm notifications. For example, if the rule is set as an average CPU usage within 1 minute of greater than or equal 90%, and this happens on 3 occasions, the rule is triggered if the average CPU usage within 1 minute is greater than or equal to 90% on 3 occasions.
   - Effective Period: The effective time of the alarm rule. CloudMonitor only checks whether the monitoring metric triggers an alarm rule during this effective period. The system also only checks whether the monitoring data requires an alarm during the effective time.

7. Configure the Notification Method area.

   - Notification Contact: The contact group that receives the notification. Enter a keyword for the contact group in the search box to quickly locate the group that you want to associate. By clicking the right arrow icon, the contact group is added to the contact list on the right. If you have not yet created a contact group, click Quickly create a contact group. If you want to delete a contact group from the contact list on the right, click the left arrow icon.

- Notification Methods: Alarm information is divided into three levels: critical, warning, and information. These different levels correspond to different notification methods. Before you can configure the critical level, you must first purchase the phone alarm resource package.
- Email Remark (Optional): Customizes additional information for alarm emails. The remarks you enter are included in the notification message sent to contacts.
- HTTP Callback (Optional): This feature allows you to integrate the alarm notifications sent by CloudMonitor into existing maintenance systems or message notification systems. CloudMonitor pushes alarm notifications to a specified public URL through the POST request method supported by HTTP. When you receive an alarm notification, you can process it further according to its content. For more information, see *Alarm callback*.

8. Click Confirm to complete the alarm rule configuration

5 Software configuration

Software such as Hadoop, Hive, and Pig contain a large number of configurations that can be modified using the software configuration function. For example, you may want to increase the number of service threads in the dfs.namenode.handler.count HDFS server from 10 to 50, or decrease the default size of HDFS file block dfs.blocksize from 128 MB to 64 MB because the system only contains small files. The software configuration function can only be performed once when a cluster is started.

Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. Select a region. The cluster associated with this region is listed.
3. Click Create Cluster to enter the cluster creation page.
4. All of the software and the corresponding versions can be seen in the Software Configuration step during cluster creation. If you want to change the configuration of the cluster, select a corresponding .json configuration file in the Enable Custom Setting box. You can then override or add to the default cluster parameters. An example of a .json file is as follows:

```
[
    {
        "ServiceName":"YARN",
        "FileName":"yarn-site",
        "ConfigKey":"yarn.nodemanager.resource.cpu-vcores",
        "ConfigValue":"8"
    },
    {
        "ServiceName":"YARN",
        "FileName":"yarn-site",
        "ConfigKey":"aaa",
        "ConfigValue":"bbb"
    }
]
```

Note:

- The suffix of the actual incoming FileName parameter needs to be removed.
- Capitalize all letters of the ServiceName parameter name.

ConfigKey is the name of a configuration, whereas ConfigValue is a specific value.

Configuration files of all services are as follows:

- Hadoop

Filename:

  - core-site.xml
  - log4j.properties
  - hdfs-site.xml
  - mapred-site.xml
  - yarn-site.xml
  - httpsfs-site.xml
  - capacity-scheduler.xml
  - hadoop-env.sh
  - httpfs-env.sh
  - mapred-env.sh
  - yarn-env.sh
- Pig

  Filename:

  - pig.properties
  - log4j.properties
- Hive

  Filename:

  - hive-env.sh
  - hive-site.xml
  - hive-exec-log4j.properties
  - hive-log4j.properties

After setting configuration files, click Next to complete software configurations.

6 Bootstrap actions

Bootstrap actions are used to run a customized script before the cluster starts Hadoop. The customized script is used to install your required third-party software or change the cluster operating environment.

Function

With bootstrap actions, you can install and perform a variety of things to your cluster that are not currently supported by clusters, such as:

- Install software with Yum.
- Directly download open software from a public network.
- Read your data from OSS.
- Install and operate a service, such as Flink or Impala. (The script for this is more complex.)

We strongly recommend that you test the bootstrap action with a Pay-As-You-Go cluster and create a Subscription cluster only after the test is successful.

Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. Select a region. The cluster associated with this region is listed.
3. Click Create Cluster to enter the cluster creation page.
4. At the bottom of the basic configuration page, click Add to enter the operation page.
5. Enter the configuration items.

You can add up to 16 bootstrap actions to be performed during cluster initialization in a specified sequence. By default, your customized script is run with the root account. Use `su hadoop` in the script to switch to a Hadoop account.

It is possible for a bootstrap action to fail, but this does not affect the creation of a cluster. After a cluster is created successfully, you can view all abnormalities in the bootstrap/software configuration column in the cluster details page. In the event of an abnormality, you can log on to all nodes to view the operation logs in the *`/var/log/bootstrap-actions`* directory.

Bootstrap action type

Bootstrap actions are categorized into customized bootstrap actions and operating -condition bootstrap action. The main difference is that the operating-condition bootstrap action can only operate your operation in nodes that meet the requirements .

Customized bootstrap action

For customized bootstrap actions, the position of the bootstrap action name and the execution script in OSS must be specified and, where necessary, optional parameters filled in. During cluster initialization, all nodes download the specified OSS scripts and runs them directly or after adding the optional parameters.

You can specify the files that need to be downloaded from OSS in the script. In the following example, the *oss://yourbucket/myfile.tar.gz* file is downloaded locally and extracted to the */yourdir* directory:

```
#! #!/bin/bash
osscmd --id=<yourid> --key=<yourkey> --host=oss-cn-hangzhou-internal.
aliyuncs.com get oss://<yourbucket>/<myfile>.tar.gz ./<myfile>.tar.gz
mkdir -p /<yourdir>
tar -zxvf <myfile>.tar.gz -C /<yourdir>
```

The osscmd is preinstalled on the node and can be invoked directly to download the file.

> Notice:
> OSS address host has an intranet address, an Internet address, and a VPC network address. If you use a classic network, you need to specify the intranet address. For Hangzhou, this is oss-cn-hangzhou-internal.aliyuncs.com. If you use a VPC network, you need to specify the domain name that the VPC intranet can access. For Hangzhou, this is vpc100-oss-cn-hangzhou.aliyuncs.com.

Bootstrap actions can install additional system software packages through Yum. The following example shows the installation of ld-linux.so. 2:

```
#! #!/bin/bash
yum install -y ld-linux.so. 2
```

Operating-condition bootstrap action

The execution script of an operating-condition bootstrap action is predefined. This means that you only need to specify its name and optional parameters. Operating-condition bootstrap actions must provide optional parameters, including operating conditions and commands (separated by spaces). The operating conditions support instance.isMaster=true/false, which specifies that it only operates on master or non-master nodes. The following example shows that the optional parameters of an operating-condition bootstrap action are specified to only create a directory on a master node:

```
instance.isMaster=true mkdir -p /tmp/abc
```

If you need to specify multiple operating commands, you can divide them by using a semicolon (;). For example: `instance.isMaster=true mkdir -p /tmp/abc;mkdir -p /tmp/def`.

7 VPC

Virtual Private Cloud (VPC) helps you build an isolated network environment, including customizing the IP address range, network segment, routing table, and gateway.

For more information, see *What is VPC*. VPC can be interconnected with physical IDC equipment rooms using *Express Connect*.

Create a VPC cluster

When you create a cluster in E-MapReduce, you can select from two types of network: classic and VPC. If you select VPC, complete the following operations:

- Subordinate VPC: Select a VPC where the current E-MapReduce cluster is located. If you have not yet created a VPC, log on to the *VPC console* and create one.
- VSwitch: An ECS instance in the E-MapReduce cluster communicates through a VSwitch. If you have not yet created a VSwitch, log on to the *VPC console* and create one. Because a VSwitch has the properties of an availability zone, when you create a cluster in E-MapReduce, the VSwitch you create must also belong to the availability zone selected.
- Security group: The security group the cluster belongs to. Currently, only the security group of a VPC can be used, not the security group of a classic network. To ensure security, a security group created outside of E-MapReduce cannot be selected. Enter a security group name to create a security group.

Example

The following example shows how to enable Hive to access HBase clusters in E-MapReduce in different VPCs.

1. Create clusters.

   Create two clusters in E-MapReduce. Hive cluster C1 is located in VPC1, whereas HBase cluster C2 is located in VPC2. Both clusters are located in the cn-hangzhou region.
2. Configure the high-speed channel.

   For more information, see *Establish an intranet connection between VPCs under the same account*. Select the same region.

3. Log on to the HBase cluster through SSH. Create a table through HBase Shell.

```
hbase(main):001:0> create 'testfromHbase','cf'
```

4. Log on to Hive through SSH.

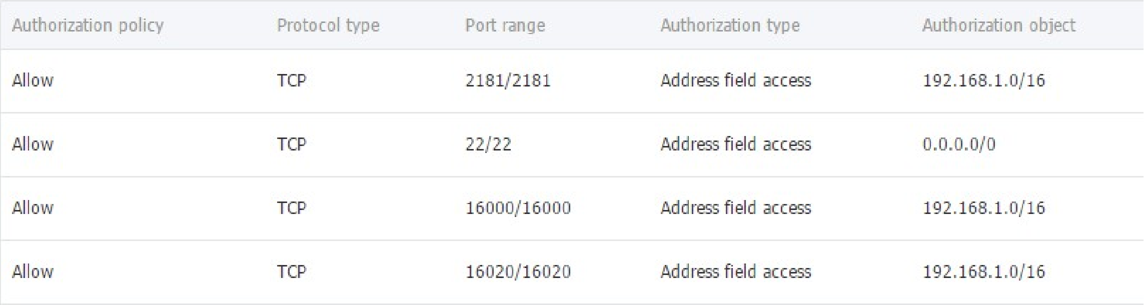
   a. Modify the hosts and add the following line:

   ```
   $zk_ip emr-cluster //$zk_ip is the zk node IP of Hbase cluster.
   ```

   b. Access HBase through Hive Shell.

   ```
   hive> set hbase.zookeeper.quorum=172.16.126.111,172.16.126.112,172
   .16.126.113;
   hive> CREATE EXTERNAL TABLE IF NOT EXISTS testfromHive (rowkey
   STRING, pageviews Int, bytes STRING) STORED BY 'org.apache.hadoop
   .hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES ('hbase.
   columns.mapping' = ':key,cf:c1,cf:c2') TBLPROPERTIES ('hbase.table
   .name' = 'testfromHbase');
   ```

   At this point, the java.net.SocketTimeoutException exception is reported. This is because the security group where the HBase cluster's ECS is located limits access to E-MapReduce at the related port. By default, security groups created by E-MapReduce only open port 22. Therefore, a security group rule must be added to the HBase cluster's security group so as to open a port for the Hive cluster, as shown in the following figure.



Authorization policy	Protocol type	Port range	Authorization type	Authorization object
Allow	TCP	2181/2181	Address field access	192.168.1.0/16
Allow	TCP	22/22	Address field access	0.0.0.0/0
Allow	TCP	16000/16000	Address field access	192.168.1.0/16
Allow	TCP	16020/16020	Address field access	192.168.1.0/16

8 Python instructions

This section provides an overview of different Python instructions.

Python 2.7

Python 2.7 is supported in E-MapReduce 2.0.0 and later. The files are located at usr/local/Python-2.7.11/. Numpy is included.

Python 3.6

Python 3.6.4 is supported in EMR 2.10.0, 3.10.0 and later. The files are located at /usr/bin/python3. Earlier versions do not support Python 3 by default. If you want to install it, complete the following steps:

- *Download the Python 3 package*.
- Unzip the downloaded file using the following commands:

```
tar zxvf Python-3.6.4.tgz
cd Python-3.6.4 ./configure --prefix=/usr/local/Python-3.6.4
make && make install
ln -s /usr/local/Python-3.6.4/bin/python3.6 /bin/python3
ln -s /usr/local/Python-3.6.4/bin/pip3 /bin/pip3
```

- Verify the environment:

```
[root@emr-header-1 bin]# python3
```

```
Python 3.6.4 (default, Mar 12 2018, 14:03:26)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-16)] on linuxType “help”, “
copyright”, “credits” or “license” for more information.
```

```
[root@emr-header-1 bin]# pip3 -V
```

```
pip 9.0.1 from /usr/local/Python-3.6.4/lib/python3.6/site-packages (
python 3.6)
```

If the preceding command information is displayed, you have successfully installed Python 3.

9 Open source components

9.1 Hue

E-MapReduce currently supports *Hue*, which you can access through Apache Knox. The following section provides an overview of how to use Hue.

Preparation

In the *Security groups* cluster, *set the security group rules*, and open port 8888.

> Notice:
>
> Set security group rules for limited IP ranges. IP 0.0.0.0/0 is not allowed to add into the security group.

Access Hue

To access Hue, complete the following steps:

1. In the EMR console, click Manage to the right of the cluster ID.
2. On the left side of the Configuration page, click Access Links and Ports.

View the password

If Hue does not have an administrator after the first running, the first user to log on is set automatically to administrator. For security, E-MapReduce generates an administrator account and password by default. The administrator account is admin. To view the password, complete the following steps:

1. Click Manage to the right of the cluster ID.
2. In the Clusters and Services panel, click Hue.
3. Click the Configuration tab to go to the `admin_pwd` parameter. It is a random password.

Create a new account if you forget your password

If you forget your password for your Hue account, you can create a new account by completing the following steps:

1. In the cluster list page, click Manage next to the target cluster.
2. In the navigation panel on the left, click Cluster Overview.

3. In the Core Instance Group, obtain the public network IPs of some master nodes.
4. Log on to the master node through SSH.
5. Execute the following command:

   ```
   /opt/apps/hue/build/env/bin/hue createsuperuser
   ```

6. Enter a new user name, e-mail, and password, and press Enter.

   If Superuser created successfully is displayed, you have successfully created a new account. You can now log on to Hue with the new account.

Add or modify a configuration

1. In the cluster list page, click Manage next to the target cluster.
2. In the service list, click Hue, and then click the Configuration tab.
3. In the upper-right corner of the page, click Custom Configuration, and configure the Key and Value fields. The key must adhere to the following specifications:

   ```
   $section_path.$real_key
   ```

   Note:

   - `$real_key` is the actual key to be added, such as `hive_server_host`.
   - In the *hue.ini* file, you can view the `$section_path` before the `$real_key`.

     For example, if the `hive_server_host` belongs to the `[beeswax]` section, this means that the `$section_path` is `beeswax`. If this is the case, the key to be added is `beeswax.hive_server_host`.
   -
   - If you need to modify the multilevel section `[desktop] -> [[ldap]] -> [[[ldap_servers]]] -> [[[[users]]]] ->user_name_attr` value in the *hue.ini* file, the key to be configured is `desktop.ldap.ldap_servers.users.user_name_attr`.

9.2 Oozie

The following section provides an overview of how to use Oozie in a E-MapReduce cluster.

Note:

> E-MapReduce version 2.0.0 and later support Oozie. If you need to use Oozie in a cluster, make sure that the version you are using is 2.0.0 or higher.

Preparations

Before you create a cluster, you must first open an SSH tunnel. For more information, see *Connect to clusters using SSH*.

In the following, which uses a MAC environment as an example, the IP address of the public network for the cluster's master node is assumed to be xx.xx.xx.xx:

1. Log on to the master node.

   ```
   ssh root@xx.xx.xx.xx
   ```

2. Enter your password.
3. Check the `id_rsa.pub` content of the local machine. Note that this is executed on the local machine, not the remote master node.

   ```
   cat ~/.ssh/id_rsa.pub
   ```

4. Write the `id_rsa.pub` content of the local machine in *`~/.ssh/authorized_keys`* on the local master node, which is executed on the remote master node.

   ```
   mkdir ~/.ssh/
   vim ~/.ssh/authorized_keys
   ```

5. Copy and paste the content observed in *Step 2*. You should now be able to log on to the master node without a password using `ssh root@xx.xx.xx.xx`.
6. Execute the following command on the local machine to perform port forwarding:

   ```
   ssh -i ~/.ssh/id_rsa -ND 8157 root@xx.xx.xx.xx
   ```

7. Execute the following command to enable Chrome to in the new terminal on the local machine:

   ```
   /Applications/Google\ Chrome.app/Contents/MacOS/Google\ Chrome --
   proxy-server="socks5://localhost:8157" --host-resolver-rules="MAP *
   0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmp
   ```

Access the Oozie UI interface

Access the following in Chrome to perform port forwarding: xx.xx.xx.xx:11000/oozie, localhost:11000/oozie, or intranet ip: 11000/oozie.

Submit a workflow job

Before you run Oozie, you first have to install *Oozie's ShareLib*.

In E-MapReduce clusters, ShareLib is installed by default for Oozie users. If you are using Oozie to submit a workflow job, you do not need to install ShareLib again.

Clusters with HA enabled use different methods to access NameNode and ResourceManager than clusters with HA disabled. Therefore, when you submit an Oozie workflow job, you need to specify a different NameNode and JobTracker (ResourceManager) in job.properties files. To do so, complete the following steps:

- Non-HA clusters

  ```
  nameNode=hdfs://emr-header-1:9000
  jobTracker=emr-header-1:8032
  ```

- HA clusters

  ```
  nameNode=hdfs://emr-cluster
  jobTracker=rm1,rm2
  ```

In the following examples, configurations are made for both non-HA and HA clusters. For operations that do not require modification, the sample code can be used directly. For the specific format of a workflow file, see the relevant documentation on the *official Oozie website*.

- Submit a workflow job on a non-HA cluster

  1. Log on to the main master node of the cluster.

     ```
     ssh root@publicIp_of_master
     ```

  2. Download the sample code.

     ```
     [root@emr-header-1 ~]# su oozie
     [oozie@emr-header-1 root]$ cd /tmp
     [oozie@emr-header-1 tmp]$ wget http://emr-sample-projects.oss-cn-
     hangzhou.aliyuncs.com/oozie-examples/oozie-examples.zip
     ```

```
[oozie@emr-header-1 tmp]$ unzip oozie-examples.zip
```

3. Synchronize the Oozie workflow code to HDFS.

```
[oozie@emr-header-1 tmp]$ hadoop fs -copyFromLocal examples/ /user
/oozie/examples
```

4. Submit a sample Oozie workflow job.

```
[oozie@emr-header-1 tmp]$ $OOZIE_HOME/bin/oozie job -config
examples/apps/map-reduce/job.properties -run
```

   After submitting the job successfully, a jobId is returned, for example:

```
job: 0000000-160627195651086-oozie-oozi-W
```

5. Go to the Oozie UI page to view the submitted Oozie workflow job.

- Submit a workflow job on an HA cluster

1. Log on to the main master node of the HA cluster.

```
ssh root@main_master_ip
```

   To determine the current main master node, check whether the Oozie UI can be accessed or not. By default, the Oozie server service is enabled on the main master node `xx.xx.xx.xx:11000/oozie`.

2. Download the sample code.

```
[root@emr-header-1 ~]# su oozie
[oozie@emr-header-1 root]$ cd /tmp
[oozie@emr-header-1 tmp]$ wget http://emr-sample-projects.oss-cn-
hangzhou.aliyuncs.com/oozie-examples/oozie-examples-ha.zip
[oozie@emr-header-1 tmp]$ unzip oozie-examples-ha.zip
```

3. Synchronize the Oozie workflow code to HDFS.

```
[oozie@emr-header-1 tmp]$ hadoop fs -copyFromLocal examples/ /user
/oozie/examples
```

4. Submit a sample Oozie workflow job.

```
[oozie@emr-header-1 tmp]$ $OOZIE_HOME/bin/oozie job -config
examples/apps/map-reduce/job.properties -run
```

   After submitting the job successfully, a jobId is returned. This should be similar to:

```
job: 0000000-160627195651086-oozie-oozi-W
```

5. Go to the Oozie UI page to view the submitted Oozie workflow job.

9.3 Presto

The following section provides an overview of how to use Presto.

E-MapReduce versions 2.0 and later support *Presto*. Presto can be used in E-MapReduce by checking the Presto software box when you select a mirror image.

After you create a cluster, log on to the master node. The Presto software can be found in the `/usr/lib/presto-current` directory, and the PrestoServer processes can be viewed using the jps command.

PrestoServer processes can be divided into coordinator and worker processes. The coordinator process is started on the master node (the HA cluster is the master node whose hostname starts with emr-header-1), and the worker process is started on the core node. The service process configuration can be found in the `/usr/lib/presto-current/etc` directory. Coordinator uses coordinator-config.properties, whereas worker uses worker-config.properties. Other configuration files are shared. The web port is set as 9090.

By default, Presto services are supported by Hive. You can connect Hive's metastore on the cluster to read Hive table information and query it. The cluster is pre-installed with Presto CLI and can execute the following command to check Hive tables:

```
presto -server localhost:9090 -catalog hive -schema default -user
hadoop -execute 'show tables'
```

Note:

There is a delay of several seconds when Hive tables are synchronized.

9.4 Zeppelin

E-MapReduce can access Zeppelin through Apache Knox.

Preparation

1. In the *Security groups* cluster, *set the security group rules*, and open port 8080.
2. In Knox, add a user name and password. For more information on how to set Knox users, see *Knox*. The user name and password are only used to log on to the various Knox services. They are not related to Alibaba Cloud RAM user names.

Notice:

Set security group rules for limited IP ranges. IP 0.0.0.0/0 is not allowed to add into the security group.

Access Zeppelin

To view the access links for Zeppelin, complete the following steps:

1. On the right of the cluster list page, click Manage.
2. In the pane on the left, click Access Links and Ports.

9.5 ZooKeeper

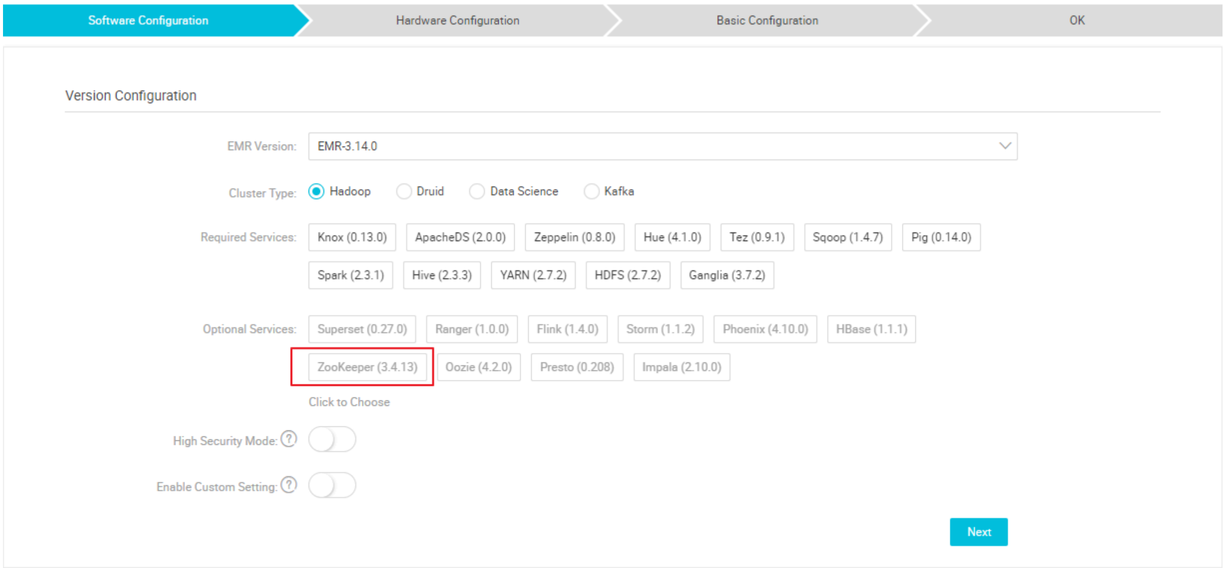
The *ZooKeeper* service is enabled in E-MapReduce clusters by default.

Note:

ZooKeeper only has 3 nodes, regardless of how many machines are currently in the cluster. More nodes are not currently supported.

Create a cluster

When you create a cluster, select the Zookeeper service in the software configuration page.



Version Configuration

EMR Version: EMR-3.14.0

Cluster Type: ● Hadoop ○ Druid ○ Data Science ○ Kafka

Required Services: Knox (0.13.0) ApacheDS (2.0.0) Zeppelin (0.8.0) Hue (4.1.0) Tez (0.9.1) Sqoop (1.4.7) Pig (0.14.0) Spark (2.3.1) Hive (2.3.3) YARN (2.7.2) HDFS (2.7.2) Ganglia (3.7.2)

Optional Services: Superset (0.27.0) Ranger (1.0.0) Flink (1.4.0) Storm (1.1.2) Phoenix (4.10.0) HBase (1.1.1) ZooKeeper (3.4.13) Oozie (4.2.0) Presto (0.208) Impala (2.10.0)

Click to Choose

High Security Mode:

Enable Custom Setting:

Node information

After you have created a cluster and its status is idle, in the Clusters and Services page, select ZooKeeper, and then click Component Topology to view ZooKeeper nodes. E-MapReduce enables 3 ZooKeeper nodes. The corresponding intranet IP address (2181 is the default port) of ZooKeeper nodes are indicated in the IP column for access to the ZooKeeper service.

9.6 Kafka

9.6.1 Quick start

E-MapReduce 3.4.0 and later support Kafka.

Create a Kafka cluster

When creating a cluster on E-MapReduce, set the cluster type to Kafka. A cluster containing only Kafka components is created by default. The components include basic components, as well as Zookeeper, Kafka, and KafkaManager components. Only one Kafka broker is deployed on each node. We recommend that you use a dedicated Kafka cluster instead of mixing with Hadoop services.

Ephemeral disk Kafka clusters

To better reduce unit costs and respond to larger storage needs, E-MapReduce 3.5.1 supports Kafka clusters on local disks (D1 cluster models). For more information, see *ECS models*. Compared to cloud disks, local disk Kafka clusters have the following features:

- High-volume local SATA HDD disks with high I/O throughput, sequential read and write performance on a single disk of 190 MB/s, and up to 5 GB/s of storage I/O capability.
- Cost of local storage is 97% lower than that of SSD cloud disks.
- Higher network performance, with up to 17 Gbit/s instances of network bandwidth . This meets data interaction requirements for peak business instances.

Local disk models also have the following features:

Operation	Ephemeral disk data status	Description
Restart within the operating system/restart or force restart in the ECS console	Retained	The local ephemeral disk's storage volume is retained . Data is also retained.
Shut down within the operating system/Stop or force stop in the ECS console	Retained	The local ephemeral disk's storage volume is retained . Data is also retained.
Release (instances) on the console	Erased	The local ephemeral disk's storage volume is erased. Data is not retained.

Notice:

- When the host is down or the disk is corrupted, the data on the disk is lost.
- Do not store business data on a local ephemeral disk for a long period of time. Back up data in a timely manner and adopt a high-availability architecture. For long-term storage, we recommend that you store data on a cloud disk.

To be able to deploy Kafka on a local disk, E-MapReduce has the following default requirements:

1. `default.replication.factor = 3` indicates that the number of partitions and replicas in the topic is at least three. If a smaller number of replicas is set, the risk of data loss is increased.
2. `min.insync.replicas = 2` indicates that when the producer is required to set acks to all (-1), it is considered successful to write at least two replicas at a time.

When a local disk corruption occurs, E-MapReduce performs the following:

1. Removes the bad disk from the broker configuration, restarts Broker, and recovers the lost data from the bad disk on the other available local disks. The time it takes to perform data recovery varies according to the amount of data that has been written on the broken disk.
2. When the number of damaged machine disks is over 20%, E-MapReduce takes the initiative to migrate the machine and restore the abnormal disk.
3. If there is not enough disk space available on the current machine to recover lost data on the damaged disk, Broker is shut down abnormally. If this is the case, you can choose to clean some data, free up disk space, or restart the Broker service. You can also open a ticket with E-MapReduce for machine migration and to recover abnormal disks.

Parameter description

You can check Kafka software configurations on the E-MapReduce cluster configuration management interface.

Configuration item	Description
zookeeper.connect	Zookeeper connection address configured on Kafka.
kafka.heap.opts	Size of the heap memory of the Kafka broker.

Configuration item	Description
num.io.threads	Number of the Kafka broker's I/O threads, which by default is twice the number of CPU cores.
num.network.threads	Number of the Kafka broker's network threads, which by default is the same as the number of CPU cores.

9.6.2 Cross-cluster Kafka access

An independent Kafka cluster is deployed to provide the Kafka service. Therefore, you may need to access this service across clusters.

Cross-cluster access to Kafka

Cross-cluster access to Kafka consists of two types:

- Accessing E-MapReduce Kafka clusters from the Alibaba Cloud intranet network.
- Accessing E-MapReduce Kafka clusters from the public network.

Different solutions are prepared for different E-MapReduce versions.

EMR-3.11.x and later

- Access Kafka from the Alibaba Cloud intranet network

  You can access Kafka by using the intranet IP address of a Kafka cluster node. Use port 9092 to access Kafka from the intranet network.

  Make sure that the networks are accessible before you access Kafka:

  - For more information about how to access a VPC from a classic network, see *Enable access between classic networks and VPCs* here.
  - For more information about how to access a VPC from another VPC, see *Configure a VPC-to-VPC connection*.
- Access Kafka in the public network

  The core node of the Kafka cluster is unable to access the public network by default. To access the Kafka cluster in the public network, complete the following steps:

  1. Interconnect Kafka clusters with the public network.
     - If Kafka clusters are deployed in a VPC environment, there are two ways to interconnect them:

        - ■ Deploy Express Connect to interconnect the VPC with the public network. For details, see *Express Connect*.
        - ■ Bind EIPs to cluster core nodes. For details, see *EIP*. The following steps bind the EIP to the ECS:
    - If Kafka is deployed in a classic network, there are two ways to interconnect them:
        - ■ To create a Pay-As-You-Go cluster, use ECS APIs. For details, see *API*.
        - ■ To create a Subscription cluster, you can directly assign a public IP address to the relevant host in the ECS console.

2. Create an EIP in the *VPC console* and purchase the relevant EIPs based on the number of core nodes in the Kafka cluster.
3. Configure security group rules that allow the Kafka cluster to control public network access to the cluster's IP addresses. This improves the security of the Kafka cluster exposed in the public network. You can view the security group to which the cluster belongs in the E-MapReduce console, and configure security group rules based on security group IDs. For more information, see *Security group rules*.
4. On the Cluster Management page of the E-MapReduce console, click Manage next to the specified cluster, select Cluster Overview on the left side of the page, and then click Sync Cluster Host Info in the upper-right corner.
5. Restart the Kafka cluster.
6. Use the EIP of the Kafka cluster node to access Kafka in the public network. Use port 9093 to access Kafka from the public network.

Versions earlier than EMR-3.11.x

- Access Kafka from the Alibaba Cloud intranet network

  You must configure the host information of the Kafka cluster node on the client host. The Long domain of the Kafka cluster node must also be configured. For example:

  ```
  ／etc/hosts
  # kafka cluster
  10.0.1.23 emr-header-1.cluster-48742
  10.0.1.24 emr-worker-1.cluster-48742
  10.0.1.25 emr-worker-2.cluster-48742
  10.0.1.26 emr-worker-3.cluster-48742
  ```
- Access Kafka in the public network

The core node of the Kafka cluster is unable to access the public network by default. To access the Kafka cluster in the public network, complete the following steps:

1. Interconnect Kafka clusters with the public network.
   - If Kafka clusters are deployed in a VPC environment, there are two ways to interconnect them:
     - Deploy Express Connect to interconnect the VPC with the public network. For details, see *Express Connect*.
     - Bind EIPs to cluster core nodes. For details, see *EIP*. Complete the following steps to bind the EIP to the ECS.
   - If Kafka is deployed in a classic network, there are two ways to interconnect them:
     - To create a Pay-As-You-Go cluster, use ECS APIs. For details, see *API*.
     - To create a Subscription cluster, you can directly assign a public IP address to the relevant host in the ECS console.
2. Create an EIP in the *VPC console* and purchase the relevant EIPs based on the number of core nodes in the Kafka cluster.
3. Configure security group rules that allow the Kafka cluster to control public network access to the cluster's IP addresses. This improves the security of the Kafka cluster exposed in the public network. You can view the security group to which the cluster belongs in the E-MapReduce console, and configure security group rules based on security group IDs. For more information, see *Security group rules*.
4. Modify the Kafka cluster's `listeners.address.principal` software configuration to `HOST`, and restart the Kafka cluster.
5. Configure the *`hosts`* file on the local client host.

9.6.3 Kafka Ranger

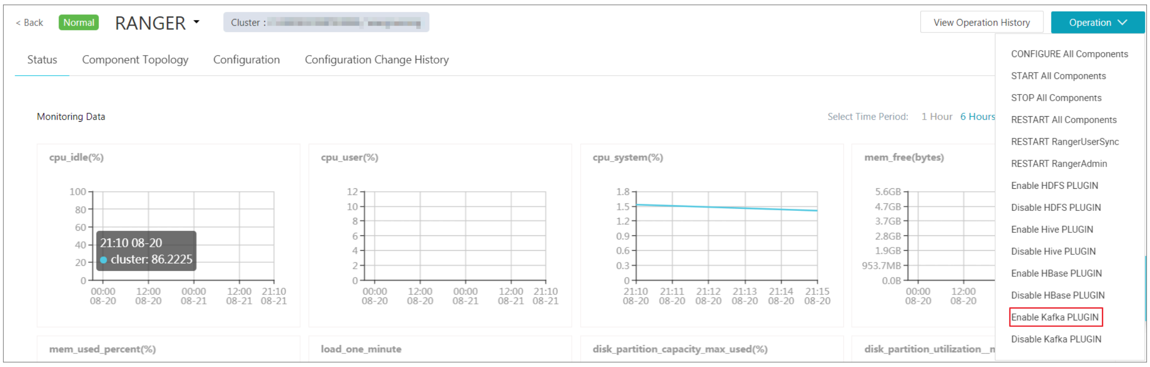
With E-MapReduce 3.12.0 and later, Kafka allows you to configure permissions with Ranger.

Integrate Ranger into Kafka

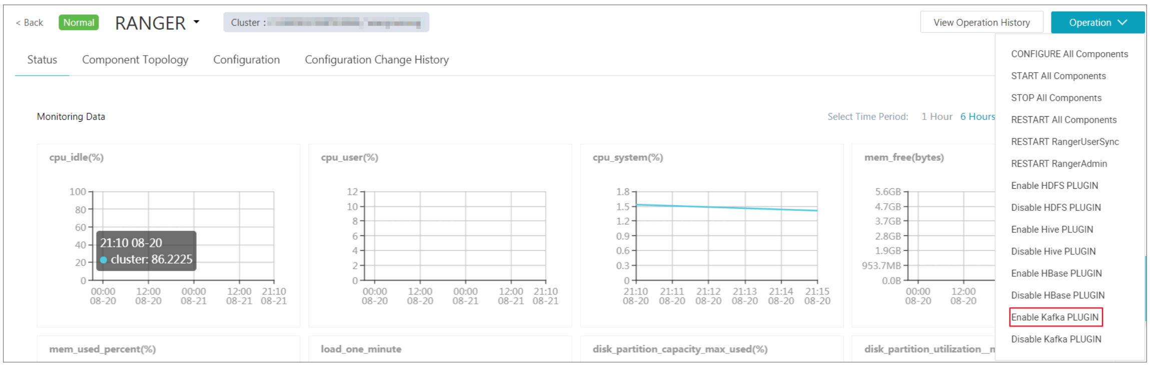
To integrate Ranger into Kafka, complete the following steps:

- Enable Kakfa Plugin

1. On the Cluster Management page, click Ranger in the service list to enter the Ranger Management page. Click Operation in the upper-right corner and select Enable Kafka PLUGIN.



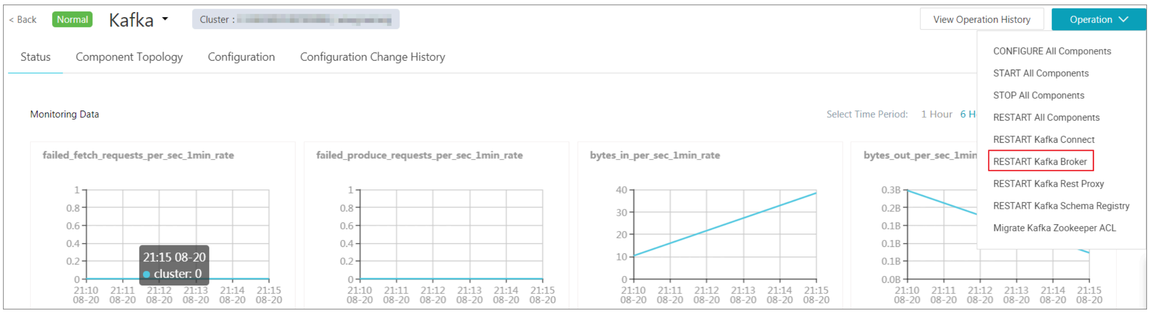
2. You can check the progress by clicking View Operation History in the upper-right corner of the page.



- Restart Kafka broker

After enabling the Kafka plugin, you must restart the broker to make it take effect.

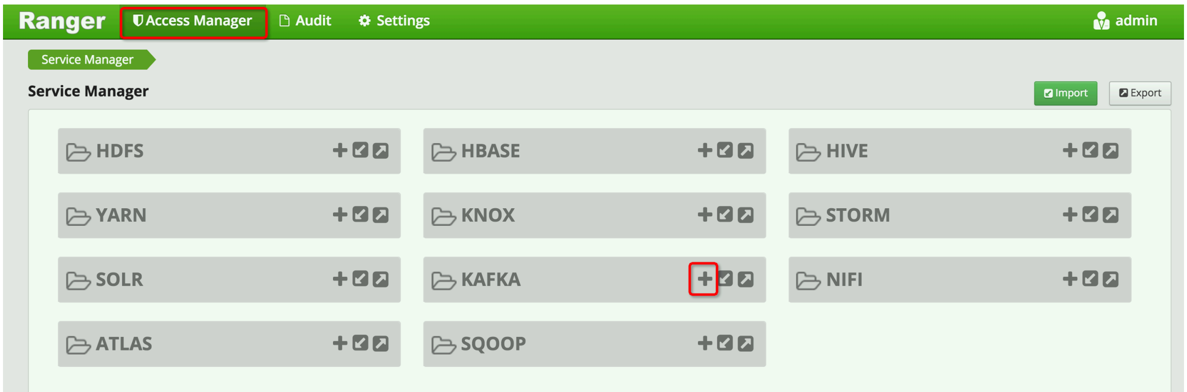
1. On the Cluster Management page, click the inverted triangle icon behind RANGER in the upper-left corner to switch to Kafka.
2. Click Actions in the upper-right corner of the page and select RESTART Broker.
3. You can check the progress by clicking View Operation History in the upper-right corner of the page.



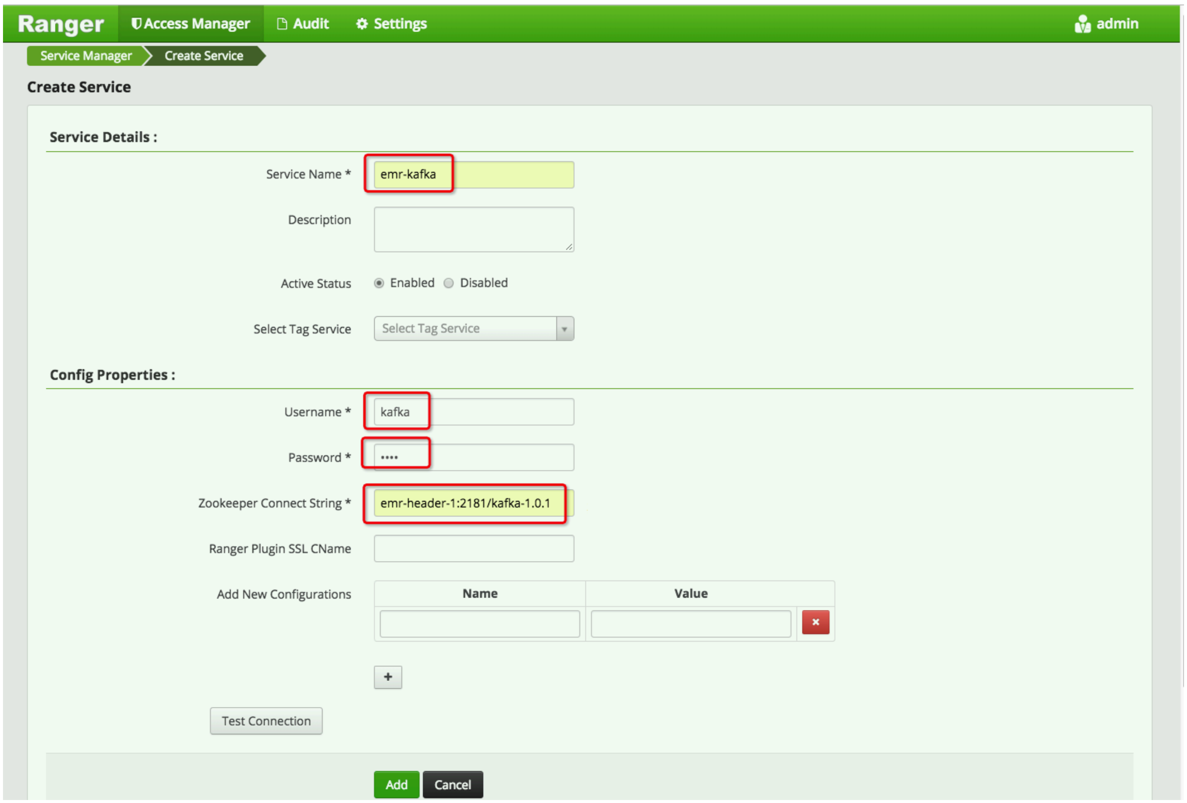
- Add Kafka service on the Ranger WebUI

  For more information about how to go to the Ranger WebUI, see *Ranger Introduction*.

  Add the Kafka service on the WebUI:



  Configure the Kafka service:



Create Service

Service Details :

Service Name * emr-kafka

Description

Active Status ● Enabled ○ Disabled

Select Tag Service: Select Tag Service

Config Properties :

Username * kafka

Password * ····

Zookeeper Connect String * emr-header-1:2181/kafka-1.0.1

Ranger Plugin SSL CName

Add New Configurations

Name	Value

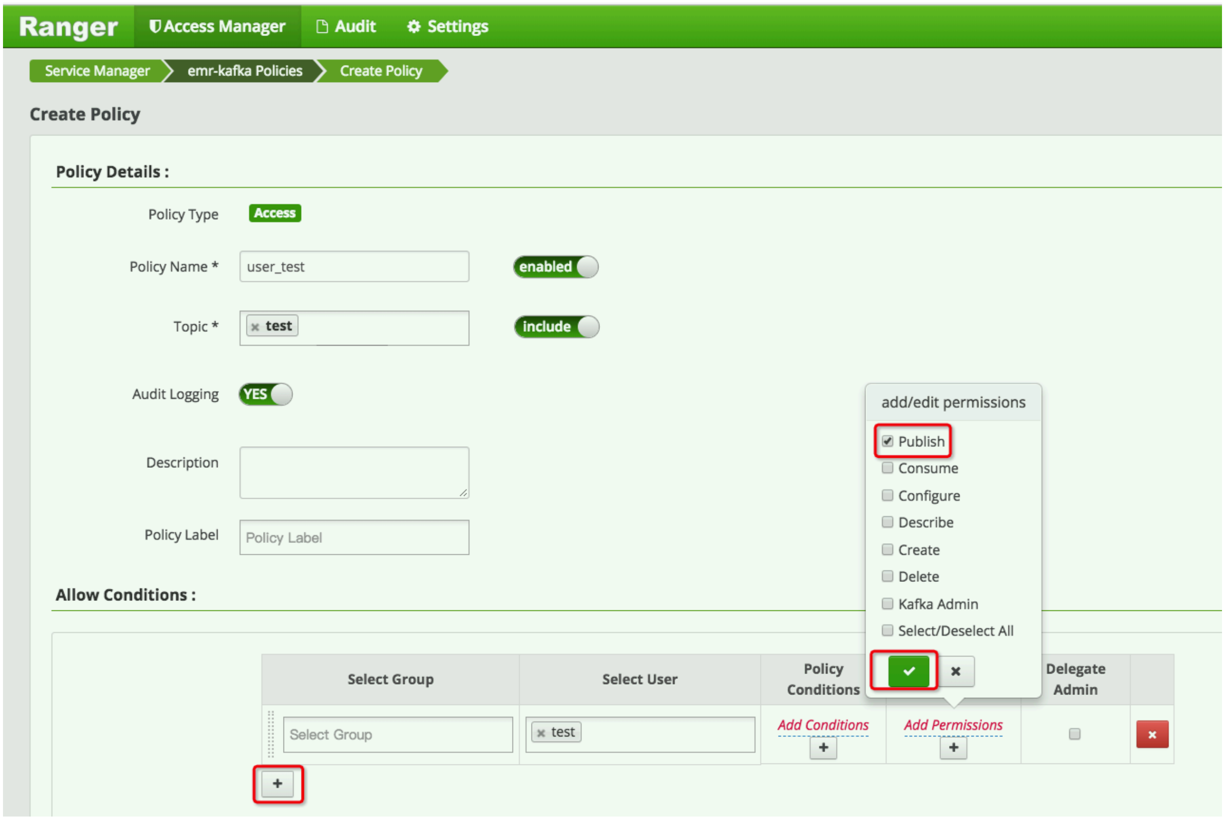
Configure permissions

After integrating Ranger into Kafka, you can set the relevant permissions.

> Notice:
>
> In a standard cluster, Ranger automatically generates the all - topic rule after the Kafka service is added. This rule indicates that there are no restrictions on

permissions. All users can perform all actions. In this case, Ranger cannot identify permissions through the user.

Here, user_test is used as an example to add the Publish permission:



Create Policy

Policy Details :

Policy Type: Access

Policy Name *: user_test — enabled

Topic *: test — include

Audit Logging: YES

Description:

Policy Label: Policy Label

Allow Conditions :

add/edit permissions

- ☑ Publish
- □ Consume
- □ Configure
- □ Describe
- □ Create
- □ Delete
- □ Kafka Admin
- □ Select/Deselect All

Select Group	Select User	Policy Conditions	Permissions	Delegate Admin	
Select Group	test	Add Conditions	Add Permissions	□	×

After you add a policy, the permissions are granted to the `test` user. This user can then perform the write operation for `test`.

Note:

The policy takes effect one minute later after it is added.

9.6.4 Kafka SSL

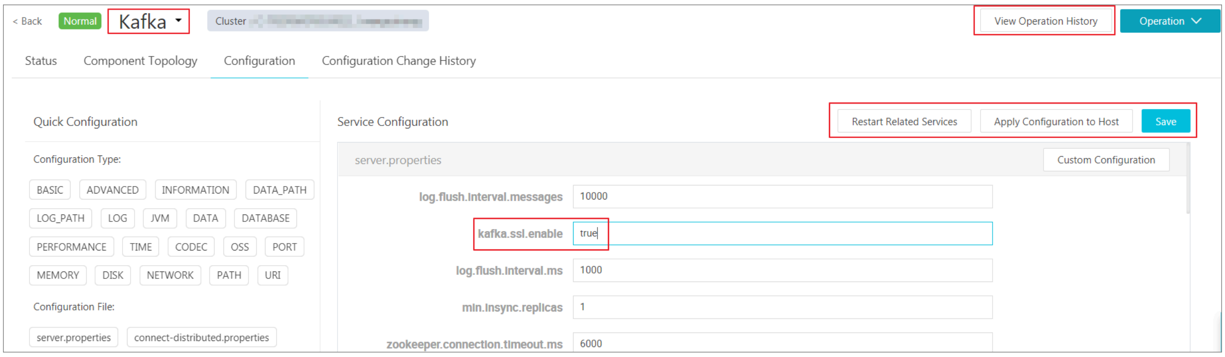
E-MapReduce Kafka supports the SSL function in E-MapReduce 3.12.0 and later.

Create a cluster

For details about how to create a cluster, see *Create a cluster*.

Enable the SSL service

By default, the SSL function is not enabled for the Kafka cluster. You can enable it on the configuration page of the Kafka service.



log.flush.interval.messages 10000

kafka.ssl.enable true

log.flush.interval.ms 1000

min.insync.replicas 1

zookeeper.connection.timeout.ms 6000

As shown in the preceding figure, change `kafka.ssl.enable` to `true` and then restart the component.

Access Kafka from the client

You need to configure `security.protocol`, `truststore`, and `keystore` when you access Kafka through SSL. Take a standard mode cluster as an example. To run a job in a Kafka cluster, you can configure the cluster as follows:

```
security.protocol=SSL
ssl.truststore.location=/etc/ecm/kafka-conf/truststore
ssl.truststore.password=${password}
ssl.keystore.location=/etc/ecm/kafka-conf/keystore
ssl.keystore.password=${password}
```

If you are running a job in an environment other than a Kafka cluster, copy the truststore and keystore files (in the *`/etc/ecm/kafka-conf/`* directory on any node of the cluster) in the Kafka cluster to the running environment and add configurations accordingly.

Take the producer and consumer programs in Kafka as an example.

1. Create the configuration file *`ssl.properties`* and add configuration items.

```
security.protocol=SSL
 ssl.truststore.location=/etc/ecm/kafka-conf/truststore
 ssl.truststore.password=${password}
 ssl.keystore.location=/etc/ecm/kafka-conf/keystore
 ssl.keystore.password=${password}
```

2. Create a topic.

```
kafka-topics.sh --zookeeper emr-header-1:2181/kafka-1.0.1 --
replication-factor 2 --
```

```
partitions 100 --topic test --create
```

3. Use an SSL configuration file to generate data.

```
kafka-producer-perf-test.sh --topic test --num-records 123456 --
throughput 10000 --record-size 1024 --producer-props bootstrap.
servers=emr-worker-1:9092 --producer.config ssl.properties
```

4. Use an SSL configuration file to consume data.

```
kafka-consumer-perf-test.sh --broker-list emr-worker-1:9092 --
messages 100000000 --topic test --consumer.config ssl.properties
```

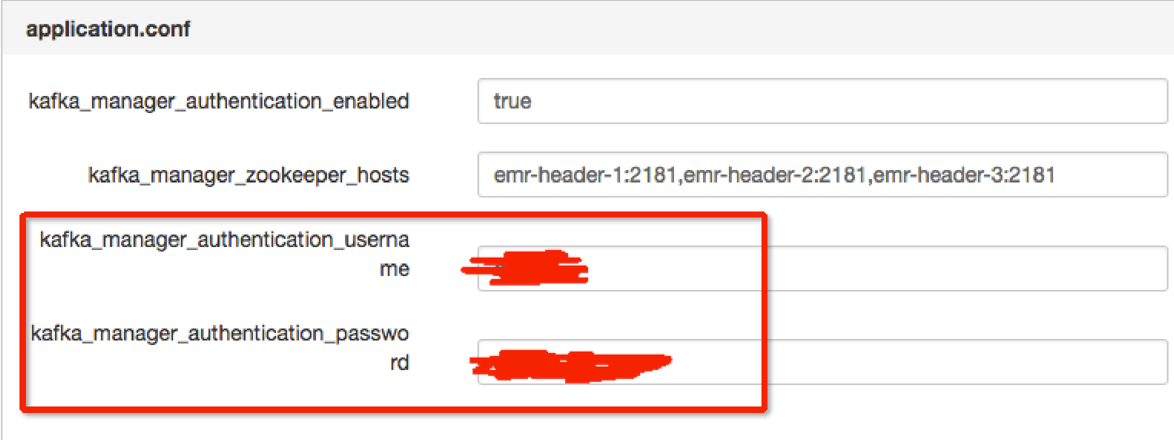
9.6.5 Kafka Manager

E-MapReduce 3.4.0 and later support Kafka Manager for use in managing Kafka clusters.

Procedure

> Notice:
> Kafka Manager software is installed by default and the Kafka Manager authentication function is enabled when a Kafka cluster is created. We strongly recommend that you change the default password when using Kafka Manager for the first time and access Kafka Manager through the SSH tunnel. We do not recommend that you expose Port 8085 to the public network unless an IP address whitelist is configured to avoid data leakage.

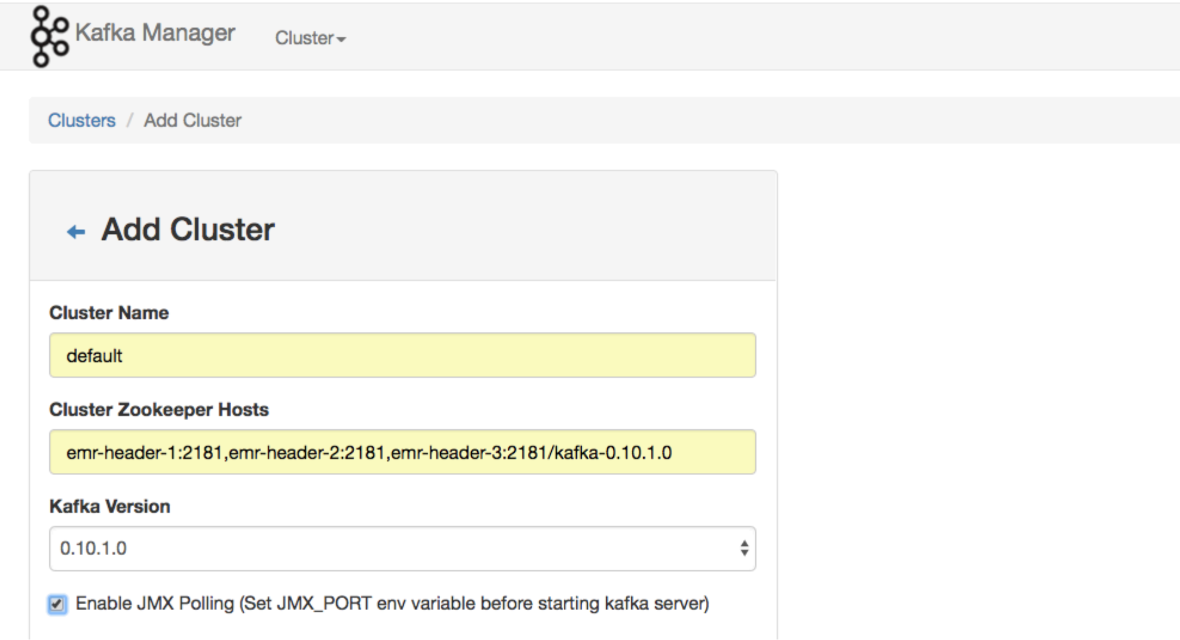
- We recommend that you access the web page through the SSH tunnel. For more information, see *Connect to clusters using SSH*.
- Access *http://localhost:8085*.
- Enter your user name and password. Refer to the configuration information of Kafka Manager.



application.conf

kafka_manager_authentication_enabled	true
kafka_manager_zookeeper_hosts	emr-header-1:2181,emr-header-2:2181,emr-header-3:2181
kafka_manager_authentication_username	[redacted]
kafka_manager_authentication_password	[redacted]

- Add an existing Kafka cluster and make sure that the Zookeeper address of the Kafka cluster is correct. For more information, see the configuration information of Kafka Manager. Select the corresponding Kafka version. We recommend that you enable the JMX function.



Add Cluster

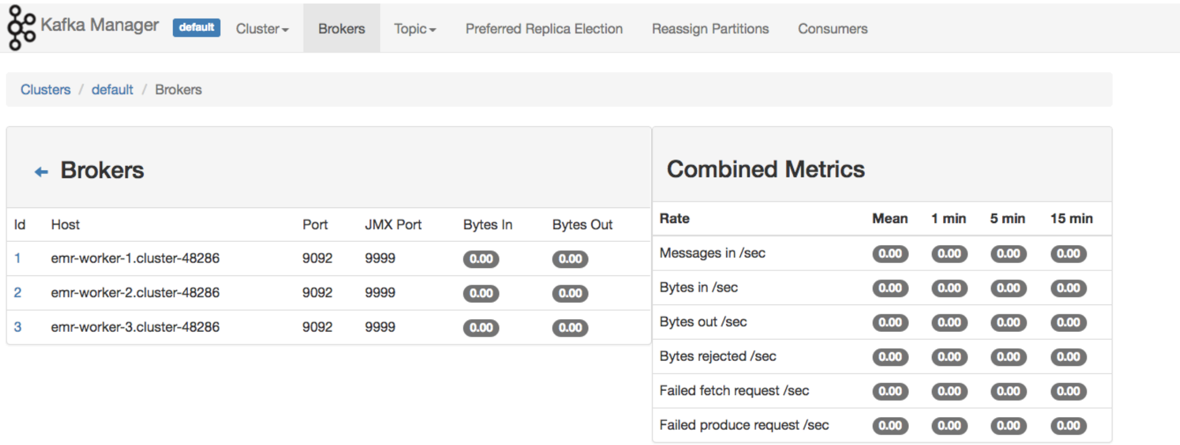
Cluster Name: default

Cluster Zookeeper Hosts: emr-header-1:2181,emr-header-2:2181,emr-header-3:2181/kafka-0.10.1.0

Kafka Version: 0.10.1.0

☑ Enable JMX Polling (Set JMX_PORT env variable before starting kafka server)

- Common Kafka functions are available immediately after you create a Kafka cluster.



Brokers

Id	Host	Port	JMX Port	Bytes In	Bytes Out
1	emr-worker-1.cluster-48286	9092	9999	0.00	0.00
2	emr-worker-2.cluster-48286	9092	9999	0.00	0.00
3	emr-worker-3.cluster-48286	9092	9999	0.00	0.00

Combined Metrics

Rate	Mean	1 min	5 min	15 min
Messages in /sec	0.00	0.00	0.00	0.00
Bytes in /sec	0.00	0.00	0.00	0.00
Bytes out /sec	0.00	0.00	0.00	0.00
Bytes rejected /sec	0.00	0.00	0.00	0.00
Failed fetch request /sec	0.00	0.00	0.00	0.00
Failed produce request /sec	0.00	0.00	0.00	0.00

9.6.6 Common Kafka problems

This section describes two common issues with Kafka.

- `Error while executing topic command : Replication factor: 1 larger than available brokers: 0.`

  Common causes:

  - A fault occurs in the Kafka service and the cluster broker process exits. You need to use logs to troubleshoot the fault.
  - The ZooKeeper address of the Kafka service is incorrect. View and use the Zookeeper.connect configuration item on the Kafka configuration management page.
- `java.net.BindException: Address already in use (Bind failed)`

  You may encounter this exception when you use Kafka command line tools. This is typically caused by the unavailability of the JMX port. You can specify a JMX port manually before using the command line. For example:

  ```
  JMX_PORT=10101 kafka-topics --zookeeper emr-header-1:2181/kafka-1.0.
  0 --list
  ```

9.7 Druid

9.7.1 Introduction to Druid

Druid is a column-oriented, open-source, distributed data store used to query and analyze issues in large data sets in real time.

Basic features

Druid has the following features:

- Sub-second OLAP queries, including multi-dimensional filtering, ad-hoc attribute grouping, and fast data aggregation.
- Real-time data consumption, collection, and querying.
- Efficient multi-tenant capability, which enables thousands of users to perform searches online at the same time.
- Strong scalability, which supports the fast processing of PB-level data, 100 billion-level events, and thousands of concurrent queries per second.
- Extremely high availability and support for rolling upgrades.

Usage scenarios

Real-time data analysis is the most typical usage scenario for Druid and covers a wide range of areas, including:

- Real-time indicator monitoring
- Model recommendations

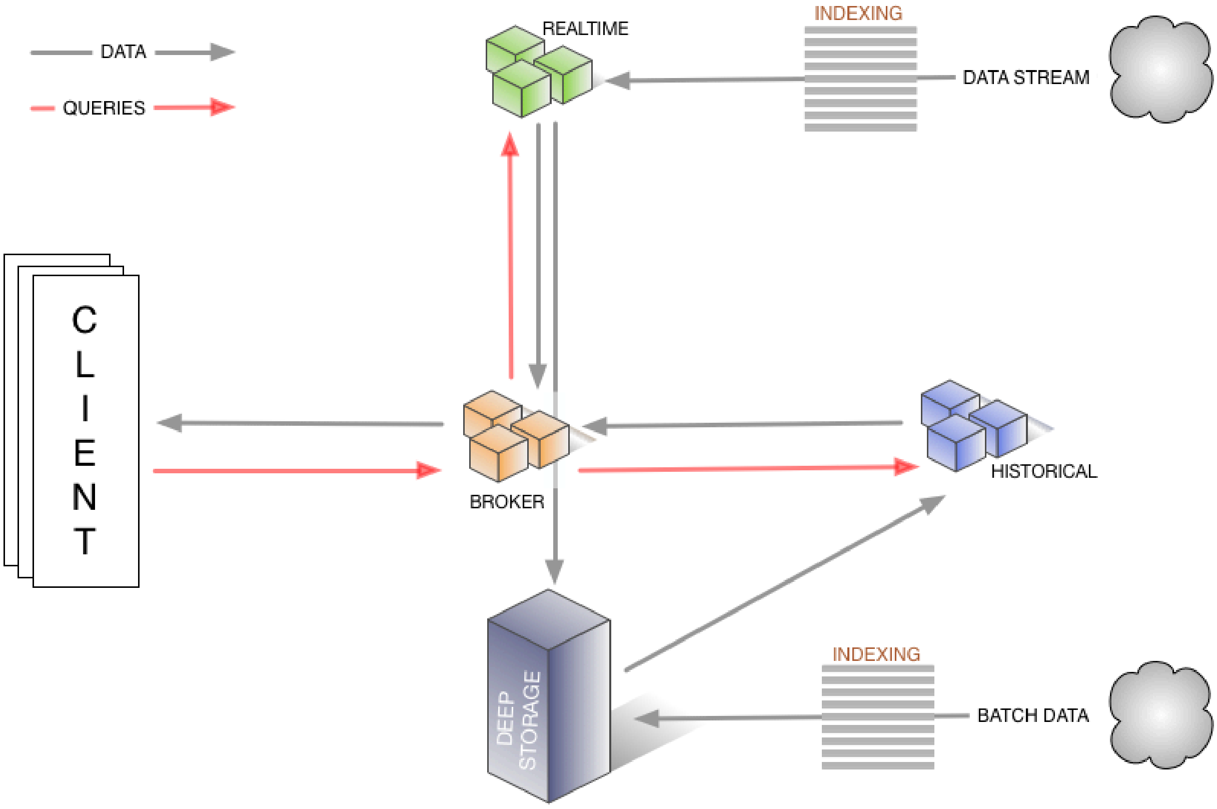
- Advertisement platforms
- Model searches

These scenarios involve large amounts of data, and the requirement for time delay in data querying is high. In real-time indicator monitoring, problems need to be detected at the moment of occurrence so that you can be warned as soon as possible . In the recommendation model, user behavior data needs to be collected in real time and sent to the recommendation system promptly. In just a few clicks, the system is able to identify your search intent and recommend more appropriate results in future searches.

Architecture

Druid has an excellent architectural design with multiple components working together to complete a series of processes, such as data collection, indexing, storage, and querying.

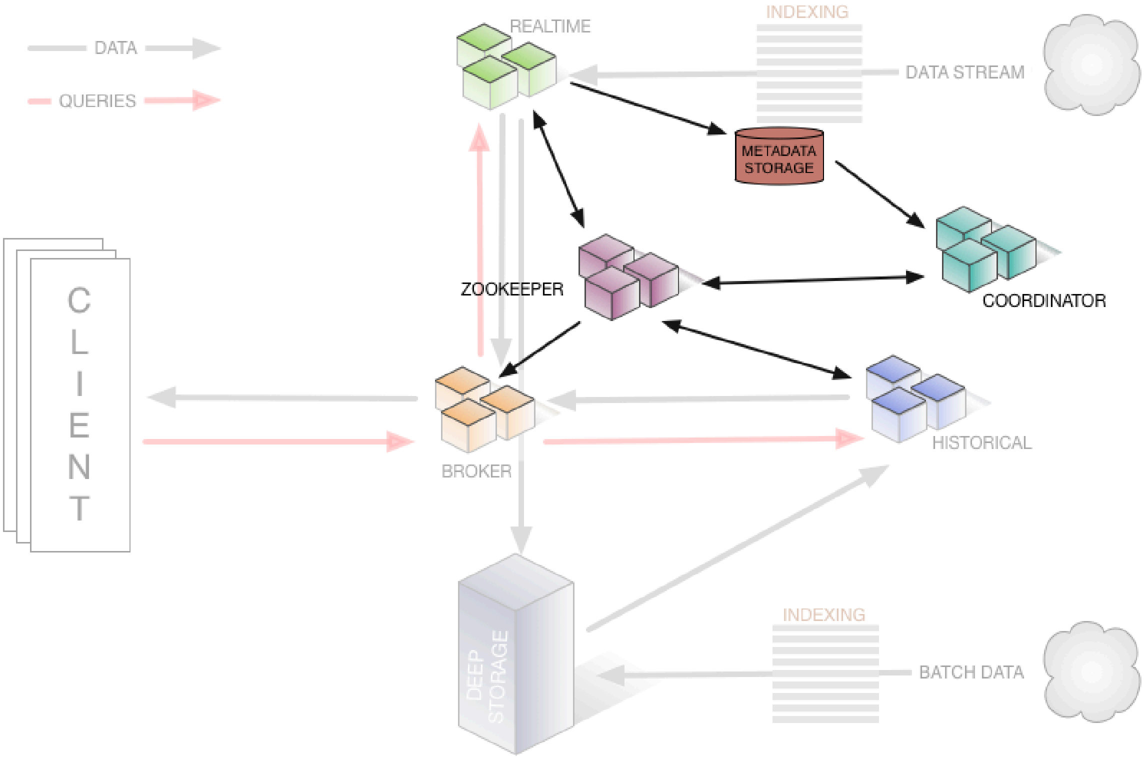
The following figure shows the components contained in the Druid working-layer (for data indexing and data querying).



- The real-time component is responsible for the real-time data collection.
- In the broker phase, query tasks are distributed, and the results are collected and returned to you.

- The historical node is responsible for the storage of historical data after indexing. The data is stored in deep storage. Deep storage can be either local or a distributed file system, such as HDFS.
- The indexing service consists of two components (not shown in the figure).
  - The Overlord component is responsible for managing and distributing indexing tasks.
  - The MiddleManager component is responsible for executing indexing tasks.

The following figure shows the components involved in the management layer of Druid segments (Druid index file).



- The ZooKeeper component is responsible for storing the status of the cluster and discovering components, such as the topology information of the cluster, election of the Overlord leader, and management of the indexing task.
- The Coordinator component is responsible for managing segments, such as the downloading and deletion of the segments and balancing them with historical components.
- The Metadata storage component is responsible for storing the meta-information of segments and managing all kinds of persistent or temporary data in the cluster, such as configuration information and audit information.

Product advantages

E-MapReduce Druid has improved a lot based on open-source Druid, including integration with E-MapReduce and the peripheral Alibaba Cloud ecosystem, easy monitoring and operation support, and easy-to-use product interfaces. You can use it immediately after purchase. It does not need 24/7 operation and maintenance.

E-MapReduce Druid supports the following features:

- Using OSS as deep storage
- Using OSS files as data sources for indexing in batches
- Using RDS to store metadata
- Integrating with Superset tools
- Easy scale up and scale down (scale down is for task node)
- Diversified monitoring indicators and alarm rules
- Bad node migration
- High-security mode
- HA

9.7.2 Quick start

E-MapReduce 3.11.0 and later support Druid as a cluster type.

Druid is used as a separate cluster type (instead of being added to a Hadoop cluster) for the following reasons:

- Druid can be used independently of Hadoop.
- Druid has high memory requirements when there is a large amount of data, especially for Broker and Historical nodes. Druid is not controlled by YARN, and will compete for resources during multi-service operation.
- As an infrastructure, the number of nodes in a Hadoop cluster can be relatively large, whereas a Druid cluster can be relatively small. Using them together results in greater flexibility.

Create a Druid cluster

Select the Druid cluster type when you create a cluster. You can select HDFS and YARN when creating a Druid cluster for testing only. We strongly recommend that you use a dedicated Hadoop cluster as the production environment.

Configure a cluster

- Configure the cluster to use HDFS as deep storage for Druid

  For a standalone Druid cluster, you may need to store your index data in the HDFS of another Hadoop cluster. Therefore, you need to complete the settings for the connectivity between the two clusters. For details, see Interaction with *Hadoop clusters*. You then need to configure the following items on the Druid configuration page and restart the service. The configuration items are in common.runtime on the configuration page.

  - druid.storage.type: HDFS
  - druid.storage.storageDirectory: The HDFS directory must be complete, such as hdfs://emr-header-1.cluster-xxxxxxxx:9000/druid/segments.

  Note:

  If the Hadoop cluster is an HA cluster, you must change emr-header-1.cluster-xxxxx:9000 to emr-cluster, or change port 9000 to port 8020.

- Use OSS as deep storage for Druid

  E-MapReduce Druid supports the use of OSS as deep storage. Due to the AccessKey-free capability of E-MapReduce, Druid can automatically get access to OSS without having to configure the AccessKey. Because the OSS function of HDFS enables Druid to have access to OSS, druid.storage.type still needs to be configured as HDFS: during the configuration process.

  - druid.storage.type: HDFS
  - druid.storage.storageDirectory: For example, oss://emr-druid-cn-hangzhou/segments.

  Because the OSS function of HDFS enables Druid to have access to OSS, you need to select one of the following two scenarios:

  - Install HDFS when you create a cluster. The system is then configured automatically. (After HDFS is installed, you can choose not to use it, disable it, or use it for testing purposes only.)

- Create *`hdfs-site.xml`* in the Druid configuration directory *`/etc/ecm/druid-conf/druid/_common/`*. The content is as follows. Copy the file to the same directory of all nodes:

```
&lt;? xml version="1.0"? >
  <configuration>
    <property>
      <name>fs.oss.impl</name>
      <value>com.aliyun.fs.oss.nat.NativeOssFileSystem</value>
    </property>
    <property>
      <name>fs.oss.buffer.dirs</name>
      <value>file:///mnt/disk1/data,...</value>
    </property>
    <property>
      <name>fs.oss.impl.disable.cache</name>
      <value>true</value>
    </property>
  </configuration>
```

The `fs.oss.buffer.dirs` can be set to multiple paths.

- Use RDS to save Druid metadata

Use the MySQL database on the header-1 node to save Druid metadata. You can also use Alibaba Cloud RDS to save the metadata.

The following uses RDS MySQL as an example to demonstrate the configuration. Before you configure it, make sure that:

- The RDS MySQL instance has been created.
- A separate account has been created for Druid to access RDS MySQL (the root account is not recommended). This example uses account name druid and password druidpw.
- Create a separate MySQL database for Druid metadata. Suppose the database is called druiddb.
- Make sure that account druid has permission to access druiddb.

In the E-MapReduce console, click Manage next to the Druid cluster you want to configure. Click the Druid service, and then select the Configuration tab to find the *`common.runtime`* configuration file. Click Custom Configuration to add the following three configuration items:

- druid.metadata.storage.connector.connectURI, where the value is: jdbc:mysql://rm-xxxxx.mysql.rds.aliyuncs.com:3306/druiddb.
- druid.metadata.storage.connector.user, where the value is druid.

  - druid.metadata.storage.connector.password, where the value is druidpw.

  Click the Save, Configuration to Host, and then Restart Related Services in the upper-right corner to make the configuration take effect.

  Log on to the RDS console to view the tables created by druiddb. You will find tables automatically created by druid.

- Service memory configuration

  The memory of the Druid service consists of heap memory (configured through jvm. config) and direct memory (configured through jvm. config and runtime. properteis). E-MapReduce automatically generates a set of configurations when you create a cluster. However, in some cases, you may still need to configure the memory.

  To adjust the service memory configuration, you can access the cluster services through the E-MapReduce console and perform related operations on the page.

  Note:

  For direct memory, make sure that:

  ```
  -XX:MaxDirectMemorySize is greater than or equal to druid.
  processing.buffer.sizeBytes * (druid.processing.numMergeBuffers +
  druid.processing.numThreads + 1).
  ```

Batch index

- Interaction with Hadoop clusters

  If you select HDFS and YARN (with their own Hadoop clusters) when creating your Druid clusters, the system automatically configures the interaction between HDFS and YARN. The following example shows how to configure the interaction between a standalone Druid cluster and a standalone Hadoop cluster. It is assumed that the Druid cluster ID is 1234 and the Hadoop cluster ID is 5678. If your clusters do not work as expected, this may be because of a slightly inaccurate operation.

  For the interaction with standard-mode Hadoop clusters, complete the following operations:

  1. Ensure the communication between the two clusters. (Each cluster is associated with a different security group, and access rules are configured for these security groups.)

2. Put core-site.xml, hdfs-site.xml, yarn-site.xml, mapred-site.xml of `/etc/ecm/hadoop-conf` of the Hadoop cluster in the `/etc/ecm/duird-conf/druid/_common` directory on each node of the Druid cluster. (If you select built-in Hadoop when creating the cluster, several soft links in this directory will map to the configuration of the Hadoop service of E-MapReduce. Remove these soft links first.)
3. Write the hosts of the Hadoop cluster to the hosts list on the Druid cluster. Note that the hostname of the Hadoop cluster should be a long name, such as emr-header-1.cluster-xxxxxxxx. We recommend that you put the Hadoop hosts after the hosts of the Druid cluster, such as:

```
...
10.157.201.36   emr-as.cn-hangzhou.aliyuncs.com
10.157.64.5     eas.cn-hangzhou.emr.aliyuncs.com
192.168.142.255 emr-worker-1.cluster-1234 emr-worker-1 emr-header-
2.cluster-1234 emr-header-2 iZbp1h9g7boqo9x23qbifiZ
192.168.143.0   emr-worker-2.cluster-1234 emr-worker-2 emr-header-
3.cluster-1234 emr-header-3 iZbp1eaa5819tkjx55yr9xZ
192.168.142.254 emr-header-1.cluster-1234 emr-header-1 iZbp1e3zwu
vnmakmsjer2uZ
For Hadoop clusters in high-security mode, perform the following
operations:
192.168.143.6   emr-worker-1.cluster-5678 emr-worker-1 emr-header-
2.cluster-5678 emr-header-2 iZbp195rj7zvx8qar4f6b0Z
192.168.143.7 emr-worker-2.cluster-5678 emr-worker-2 emr-header-3.
cluster-5678 emr-header-3 iZbp15vy2rsxoegki4qhdpZ
192.168.143.5 emr-header-1.cluster-5678 emr-header-1 iZbp10tx4e
gw3wfnh5oii1Z
```

For Hadoop clusters in high-security mode, complete the following operations:

1. Ensure the communication between the two clusters. (Each cluster is associated with a different security group, and access rules are configured for these security groups.)
2. Put core-site.xml, hdfs-site.xml, yarn-site.xml, mapred-site.xml of `/etc/ecm/hadoop-conf` of the Hadoop cluster in the `/etc/ecm/duird-conf/druid/_common` directory on each node of the Druid cluster. (If you select built-in Hadoop when creating a cluster, several soft links in this directory will point to the configuration with Hadoop. Remove these soft links first.) Modify `hadoop.security.authentication.use.has` in core-site.xml to `false`. (This configuration is completed on the client to enable AccessKey authentication for users. If Kerberos authentication is used, disable AccessKey authentication.)
3. Write the hosts of the Hadoop cluster to the hosts list of each node on the Druid cluster. Note that the hostname of the Hadoop cluster should be a long name

, such as emr-header-1.cluster-xxxxxxxx. We recommend that you put the Hadoop hosts after the hosts of the Druid cluster.

4. Set Kerberos cross-domain mutual trust between the two clusters. For more details, see *Cross-region access*.
5. Create a local Druid account (useradd-m-g hadoop) on all nodes of the Hadoop cluster, or set druid.auth.authenticator.kerberos.authtomate to create a mapping rule for the Kerberos account to the local account. For specific pre-release rules, see *here*. This method is recommended because it is easy to operate without errors.

   Note:

   In high-security Hadoop clusters, all Hadoop commands must be run from a local account. By default, this local account needs to have the same name as the principal. YARN also supports mapping a principal to a local account.
6. Restart the Druid service.

- Use Hadoop to index batch data

Druid has an example named wikiticker located in `${DRUID_HOME}/quickstart`. By default, `${DRUID_HOME}` is /usr/lib/ druid-current. Each line of the wikiticker document (wikiticker-2015-09-12-sampled.json.gz) is a record. Each record is a json object. The format is as follows:

~~~~
```json
{
    "Time": "2015-09-12T00: 46: 58.771Z ",
    "channel": "#en.wikipedia",
    "cityName": null,
    "comment": "added project",
    "countryIsoCode": null,
    "countryName": null,
    "isAnonymous": false,
    "isMinor": false,
    "isNew": false,
    "isRobot": false,
    "isUnpatrolled": false,
    "metroCode": null,
    "namespace": "Talk",
    "page": "Talk:Oswald Tilghman",
    "regionIsoCode": null,
    "regionName": null,
    "user": "GELongstreet",
    "delta": 36,
    "added": 36,
    "deleted": 0
}
~~~~

~~~
```
~~~

To use Hadoop to index batch data, complete the following steps:

1. Decompress the compressed file and place it in a directory of HDFS (such as: *hdfs://emr-header-1.cluster-5678:9000/druid*). Run the following command on the Hadoop cluster:

   ```
   ### If you are operating on a standalone Hadoop cluster, copy
    a druid.keytab to Hadoop cluster after the mutual trust is
   established between the two clusters, and run the kinit command.
    kinit -kt /etc/ecm/druid-conf/druid.keytab druid
    ###
    hdfs dfs -mkdir hdfs://emr-header-1.cluster-5678:9000/druid
    hdfs dfs -put ${DRUID_HOME}/quickstart/wikiticker-2015-09-16-
   sampled.json hdfs://emr-header-1.cluster-5678:9000/druid
   ```

   Note:

   - Modify `hadoop.security.authentication.use.has` in */etc/ecm/hadoop-conf/core-site.xml* to `false` before running the HDFS command for a high-security cluster.
   - Make sure that you have created a Linux account named Druid on each node of the Hadoop cluster.

2. Modify Druid cluster *${DRUID_HOME}/quickstart/wikiticker-index.json*, as shown below:

   ```
   {
        "type" : "index_hadoop",
        "spec" : {
            "ioConfig" : {
                "type" : "hadoop",
                "inputSpec" : {
                    "type" : "static",
                    "paths" : "hdfs://emr-header-1.cluster-5678:9000/
   druid/wikiticker-2015-09-16-sampled.json"
                }
            },
            "dataSchema" : {
                "dataSource" : "wikiticker",
                "granularitySpec" : {
                    "type" : "uniform",
                    "segmentGranularity" : "day",
                    "queryGranularity" : "none",
                    "intervals" : ["2015-09-12/2015-09-13"]
                },
                "parser" : {
                    "type" : "hadoopyString",
                    "parseSpec" : {
                        "format" : "json",
                        "dimensionsSpec" : {
                            "dimensions" : [
                                "channel",
   ```

```
                             "cityName",
                             "comment",
                             "countryIsoCode",
                             "countryName",
                             "isAnonymous",
                             "isMinor",
                             "isNew",
                             "isRobot",
                             "isUnpatrolled",
                             "metroCode",
                             "namespace",
                             "page",
                             "regionIsoCode",
                             "regionName",
                             "user"
                         ]
                     },
                     "timestampSpec" : {
                         "format" : "auto",
                         "column" : "time"
                     }
                 }
             },
             "metricsSpec" : [
                 {
                     "name" : "count",
                     "type" : "count"
                 },
                 {
                     "name" : "added",
                     "type" : "longSum",
                     "fieldName" : "added"
                 },
                 {
                     "name" : "deleted",
                     "type" : "longSum",
                     "fieldName" : "deleted"
                 },
                 {
                     "name" : "delta",
                     "type" : "longSum",
                     "fieldName" : "delta"
                 },
                 {
                     "name" : "user_unique",
                     "type" : "hyperUnique",
                     "fieldName" : "user"
                 }
             ]
         },
         "tuningConfig" : {
             "type" : "hadoop",
             "partitionsSpec" : {
                 "type" : "hashed",
                 "targetPartitionSize" : 5000000
             },
             "jobProperties" : {
                 "mapreduce.job.classloader": "true"
             }
         }
     },
     "hadoopDependencyCoordinates": ["org.apache.hadoop:hadoop-
client:2.7.2"]
```

```
  }
```

Note:

- `spec.ioConfig.type` is set to `hadoop`.
- `spec.ioConfig.inputSpec.paths` is the path of the input file.
- `tuningConfig.type` is `hadoop`.
- `tuningConfig.jobProperties` sets the classloader of the MapReduce job.
- `hadoopDependencyCoordinates` develops the version of Hadoop client.

3. Run the batch index command on the Druid cluster.

```
cd ${DRUID_HOME}
 curl --negotiate -u:druid -b ~/cookies -c ~/cookies -XPOST -H '
Content-Type:application/json' -d @quickstart/wikiticker-index.
json http://emr-header-1.cluster-1234:18090/druid/indexer/v1/task
```

   Note that items such as `- -negotiate`, `-u`, `-b`, and `-c` are for high-security mode Druid clusters. The Overlord port number is 18090 by default.

4. View the running state of the jobs.

   Enter http://emr-header-1.cluster-1234:18090/console.html into your browser to view how the jobs run. To access the page properly, you need to open an SSH tunnel in advance (see *Connect to clusters using SSH*), and start a Chrome agent. If the high-security mode is enabled for the Druid cluster, you have to configure your browser to support the Kerberos authentication process. For more information, see *here*.

5. Query the data based on Druid syntax.

   Druid has its own query syntax. You need to prepare a json-formatted query file that describes how you want to query. A topN query to the wikiticker data is as follows *`${DRUID_HOME}/quickstart/wikiticker-top-pages.json`*:

```
{
     "queryType" : "topN",
     "dataSource" : "wikiticker",
     "intervals" : ["2015-09-12/2015-09-13"],
     "granularity" : "all",
     "dimension" : "page",
     "metric" : "edits",
     "threshold" : 25,
     "aggregations" : [
         {
             "type" : "longSum",
             "name" : "edits",
             "fieldName" : "count"
         }
```

```
    ]
}
```

You can check the results of the query by running the following command:

```
cd ${DRUID_HOME}
 curl --negotiate -u:druid -b ~/cookies -c ~/cookies -XPOST -H '
Content-Type:application/json' -d @quickstart/wikiticker-top-pages
.json 'http://emr-header-1.cluster-1234:18082/druid/v2/?pretty'
```

Note that items such as`- -negotiate`, `-u`, `-b`, and`-c` are for Druid clusters in high-security mode. You can check the results of a specific query.

- Real-time index

  We recommend that you use *Tranquility client* to send real-time data to Druid. Tranquility supports sending data to Druid in a variety of ways, such as Kafka, Flink, Storm, and Spark Streaming. For more information about the Kafka method, see *Tranquility*. Druid also follows the Kafka section in Tranquility. For more information about how to use Tranquility and SDKs, see *Tranquility Help Document*.

  For Kafka, you can also use the kafka-indexing-service extension. For details, see *Kafka Indexing Service*.

- Troubleshoot index failures

  When the index fails, complete the following steps to troubleshoot the failure:

  - For the index of batch data

    1. If the curl command output displays an error or does not display any information, check the file format. Alternatively, add the `-v` parameter to the curl command to check the value returned from the RESTful API.
    2. Observe the execution of jobs on the Overlord page. If the execution fails, view the logs on that page.
    3. In many cases, logs are not generated. In the case of a Hadoop job, open the YARN page to check whether an index job has been generated.
    4. If no errors are found, you need to log on to the Druid cluster, and view the execution logs of Overlord at *`/mnt/disk1/log/druid/overlord—emr-header-1.cluster-xxxx.log`*. In the case of an HA cluster, check the Overlord that you submitted the job to.
    5. If the job has been submitted to Middlemanager, but a failure is returned from Middlemanager, you need to view the worker that the job is submitted to

in Overlord, and log on to the worker to view the Middlemanager logs (at */mnt/disk1/log/druid/middleManager-emr-header-1.cluster-xxxx.log*).

- For real-time index of Tranquility

  View the Tranquility log to check whether the message was received or dropped.

  The remaining troubleshooting steps are the same as steps 2 to 5 for batch indexing.

  Most errors are about cluster configurations and jobs. Cluster configuration errors are about memory parameters, cross-cluster connection, access to clusters in high-security mode, and principals. Job errors are about the format of the job description files, input data parsing, and other job-related configuration issues (such as ioConfig).

9.7.3 Ingestion Spec

This section briefly introduces Ingestion Spec, the description file of the index data.

Ingestion Spec is a unified description of the format of the data being indexed and how it is indexed by Druid. It is a JSON file, which consists of three parts:

```
{
    "dataSchema" : {...},
    "ioConfig" : {...},
    "tuningConfig" : {...}
}
```

Key	Format	Description	Required
dataSchema	JSON object	Describes the schema information of the data you want to consume. dataSchema is fixed and does not change with the way in which data is consumed.	Yes
ioConfig	JSON object	Describes the source and destination of the data you want to consume. If the consumption method of the data is different, ioConfig is also different.	Yes
tuningConfig	JSON object	Configures the parameters of the data you want to consume. If the consumption method of the data is different, the adjustable parameters are also different.	No

dataSchema

dataSchema describes the format of the data and how to parse the data. The typical structure is as follows:

```
{
    "dataSrouce": <name_of_dataSource>,
    "parser": {
        "type": <>,
        "parseSpec": {
            "format": <>,
            "timestampSpec": {},
            "dimensionsSpec": {}
        }
    },
    "metricsSpec": {},
    "granularitySpec": {}
}
```

Key	Format	Description	Required
dataSource	String	Name of the data source.	Yes
parser	JSON object	How the data is parsed.	Yes
metricsSpec	Array of JSON objects	Aggregator list.	Yes
granularitySpec	JSON object	Data aggregation settings, such as creating segments and aggregation granularity.	Yes

- parser

  parser determines how your data is parsed correctly. metricsSpec defines how the data is clustered for calculation. granularitySpec defines the granularity of the data fragmentation and the granularity of the query.

  There are two types of parser: string and hadoopstring. The latter is used for Hadoop index jobs. ParseSpec is a specific definition of data format resolution.

Key	Format	Description	Required
type	String	The data format can be json, jsonLowercase, csv, or tsv.	Yes
timestampSpec	JSON object	Timestamp and timestamp type .	Yes
dimensionsSpec	JSON object	The dimension of the data (columns are included).	Yes

For different data formats, additional parseSpec options may exist. The following table describes timestampSpec and dimensionsSpec.

Key	Format	Description	Required
column	String	Columns corresponding to the timestamp.	Yes
format	String	The timestamp type can be ISO, millis, POSIX, auto, or whatever is supported by *joda time*.	Yes

Key	Format	Description	Required
dimensions	JSON array	Describes which dimensions the data contains. Each dimension can be just a string . You can also specify the attribute for the dimension . For example, the type of dimensions: [dimenssion1, dimenssion2, {type: long, name : dimenssion3}] is string by default.	Yes
dimensionExclusions	Array of JSON strings	Dimension to be deleted when data is consumed.	No
spatialDimensions	Array of JSON objects	Spatial dimension.	No

- metricsSpec

MetricsSpec is an array of JSON objects. It defines several aggregators. Aggregators typically have the following structures:

~~~
```json
{
    "type": <type>,
    "name": <output_name>,
    "fieldName": <metric_name>
}
```
~~~

The following commonly used aggregators are provided:

Type	Type optional
count	count

Type	Type optional
sum	longSum, doubleSum, floatSum
min/max	longMin/longMax, doubleMin/doubleMax, floatMin/floatMax
first/last	longFirst/longLast, doubleFirst/doubleLast, floatFirst/floatLast
javascript	javascript
cardinality	cardinality
hyperUnique	hyperUnique

Note:

The last three types in the table are advanced aggregators. For information about how to use them, see *Druid official documents*.

- granularitySpec

  Two aggregation modes are supported: uniform and arbitrary. The uniform mode aggregates data with a fixed interval of time. The arbitrary mode tries to make sure that each of the segments has the same size, but the time interval for aggregation is not fixed. Uniform is the current default option.

Key	Format	Description	Required
segmentGranularity	String	Segment granularity Uniform type.	No. The default is DAY.
queryGranularity	String	Minimum data aggregation granularity for query.	No
rollup	Bool value	Aggregate or not.	No. The default is true.
intervals	String	Time interval of data consumption.	It is Yes for batch and No for realtime.

ioConfig

ioConfig describes the data source. An example of Hadoop index is as follows:

```
{
    "type": "hadoop",
    "inputSpec": {
        "type": "static",
        "paths": "hdfs://emr-header-1.cluster-6789:9000/druid/
quickstart/wikiticker-2015-09-16-sampled.json"
    }
}
```

This part is not required for streaming data that is processed through Tranquility.

TuningConfig

TuningConfig refers to additional settings. For example, you can specify MapReduce parameters to use Hadoop to create an index for batch data. The contents of tuningConfig may vary based on the data source. For more information, see the example file or official document of this service.

9.7.4 Tranquility

This section uses Kafka as an example and describes how to use Tranquility in E-MapReduce to capture data from the Kafka cluster and push it to the Druid cluster in real time.

Tranquility is an application that sends data to Druid in real-time in push mode. It solves many issues, such as multiple partitions, multiple copies, service discovery, and data loss. It simplifies the use of Druid and supports a wide range of data sources , including Samza, Spark, Storm, Kafka, and Fink.

Interaction with the Kafka cluster

The first interaction is between the Druid cluster and the Kafka cluster. The interaction configuration of the two clusters is similar to that of the Hadoop cluster. You have to set the connectivity and hosts. For standard mode Kafka clusters, complete the following steps:

1. Ensure the communication between clusters. (The two clusters are either in the same security group, or each cluster is associated with a different security group and access rules are configured for these security groups.)
2. Write the hosts of the Kafka cluster to the hosts list of each node on the Druid cluster. Note that the hostname of the Kafka cluster should be a long name, such as emr-header-1.cluster-xxxxxxxx.

For high-security mode Kafka clusters, complete the following operations (the first two steps are the same as those for standard mode clusters):

1. Ensure the communication between the two clusters (The two clusters are in the same security group, or each cluster is associated with a different security group and access rules are configured for these security groups).
2. Write the hosts of the Kafka cluster to the hosts list of each node on the Druid cluster. Note that the hostname of the Kafka cluster should be a long name, such as emr-header-1.cluster-xxxxxxxx.
3. Set Kerberos cross-domain mutual trust between the two clusters. For details, see *Cross-region access*. Bidirectional mutual trust is recommended.
4. Prepare a client security configuration file:

   ```
   KafkaClient {
         com.sun.security.auth.module.Krb5LoginModule required
         useKeyTab=true
         storeKey=true
         keyTab="/etc/ecm/druid-conf/druid.keytab"
         principal="druid@EMR. 1234. COM";
     };
   ```

   Synchronize the configuration file to all nodes in the Druid cluster and place it in a specific directory, such as */tmp/kafka/kafka_client_jaas.conf*.
5. In overlord.jvm of the Druid configuration page:

   ```
   Add Djava.security.auth.login.config=/tmp/kafka/kafka_client_jaas.
   conf
   ```
6. Configure the following option in middleManager.runtime on the Druid configuration page: `druid.indexer.runner.javaOpts=-Djava.security.auth.login.confi=/tmp/kafka/kafka_client_jaas.conf` and other jvm startup parameters.
7. Restart the Druid service.

Use Tranquility Kafka

Because Tranquility is a service, it is a consumer for Kafka and a client for Druid. You can use a neutral machine to run Tranquility, as long as this machine is able to connect to the Kafka and the Druid clusters simultaneously.

1. Create a topic named pageViews on the Kafka side.

   ```
   --If the Kafka high-security mode is enabled:
    export KAFKA_OPTS="-Djava.security.auth.login.config=/etc/ecm/kafka
   -conf/kafka_client_jaas.conf"
    --
   ```

```
 ./bin/kafka-topics.sh --create --zookeeper emr-header-1:2181,
emr-header-2:2181,emr-header-3:2181/kafka-1.0.1 --partitions 1 --
replication-factor 1 --topic pageViews
```

2. Download the Tranquility installation package and decompress it to a path.
3. Configure the dataSource.

   It is assumed that your topic name is pageViews, and each topic is a JSON file.

```
{"time": "2018-05-23T11:59:43Z", "url": "/foo/bar", "user": "alice",
 "latencyMs": 32}
 {"time": "2018-05-23T11:59:44Z", "url": "/", "user": "bob", "
latencyMs": 11}
 {"time": "2018-05-23T11:59:45Z", "url": "/foo/bar", "user": "bob",
 "latencyMs": 45}
```

   The configuration of the corresponding dataSource is as follows:

```
{
   "dataSources" : {
     "pageViews-kafka" : {
       "spec" : {
         "dataSchema" : {
           "dataSource" : "pageViews-kafka",
           "parser" : {
             "type" : "string",
             "parseSpec" : {
               "timestampSpec" : {
                 "column" : "time",
                 "format" : "auto"
               },
               "dimensionsSpec" : {
                 "dimensions" : ["url", "user"],
                 "dimensionExclusions" : [
                   "timestamp",
                   "value"
                 ]
               },
               "format" : "json"
             }
           },
           "granularitySpec" : {
             "type" : "uniform",
             "segmentGranularity" : "hour",
             "queryGranularity" : "none"
           },
           "metricsSpec" : [
             {"name": "views", "type": "count"},
             {"name": "latencyMs", "type": "doubleSum", "fieldName":
 "latencyMs"}
           ]
         },
         "ioConfig" : {
           "type" : "realtime"
         },
         "tuningConfig" : {
           "type" : "realtime",
           "maxRowsInMemory" : "100000",
           "intermediatePersistPeriod" : "PT10M",
           "windowPeriod" : "PT10M"
         }
       },
       "properties" : {
         "task.partitions" : "1",
         "task.replicants" : "1",
         "topicPattern" : "pageViews"
       }
     }
   },
   "properties" : {
     "zookeeper.connect" : "localhost",
     "druid.discovery.curator.path" : "/druid/discovery",
     "druid.selectors.indexing.serviceName" : "druid/overlord",
     "commit.periodMillis" : "15000",
     "consumer.numThreads" : "2",
     "kafka.zookeeper.connect" : "emr-header-1.cluster-500148518:
2181,emr-header-2.cluster-500148518:2181,   emr-header-3.cluster-
500148518:2181/kafka-1.0.1",
     "kafka.group.id" : "tranquility-kafka",
   }
 }
```

4. Run the following command to start Tranquility.

```
./bin/tranquility kafka -configFile
```

5. Start the producer and configure it to send data.

```
./bin/kafka-console-producer.sh --broker-list emr-worker-1:9092,emr-
worker-2:9092,emr-worker-3:9092 --topic pageViews
```

Enter the following codes:

```
{"time": "2018-05-24T09:26:12Z", "url": "/foo/bar", "user": "alice",
 "latencyMs": 32}
 {"time": "2018-05-24T09:26:13Z", "url": "/", "user": "bob", "
latencyMs": 11}
 {"time": "2018-05-24T09:26:14Z", "url": "/foo/bar", "user": "bob",
 "latencyMs": 45}
```

You can now view specific information in the Tranquility log. The corresponding real-time indexing task has also been started on the Druid side.

9.7.5 Kafka Indexing Service

This section describes how to use Druid Kafka Indexing Service in E-MapReduce to ingest Kafka data in real time.

The Kafka Indexing Service is an extension launched by Druid to ingest Kafka data in real time using Druid's indexing service. The extension enables supervisors in Overlord which start some indexing tasks in Middlemanager. These tasks connect to the Kafka cluster to ingest the topic data and complete the index creation. You need to prepare a data ingestion format file and manually start the supervisor through the RESTful API.

Interaction with the Kafka cluster

See the introduction in *Tranquility*.

Use Druid's Kafka Indexing Service to ingest Kafka data in real time

1. Run the following command on the Kafka cluster (or gateway) to create a topic named metrics.

```
--If the Kafka high-security mode is enabled:
 export KAFKA_OPTS="-Djava.security.auth.login.config=/etc/ecm/kafka
-conf/kafka_client_jaas.conf"
 --
 kafka-topics.sh --create --zookeeper emr-header-1:2181,emr-header-
2,emr-header-3/kafka-1.0.0 --partitions 1 --replication-factor 1 --
topic metrics
```

   You can adjust the parameters based on your needs. The `/kafka-1.0.0` section of the `- -zookeeper` parameter is path, and you can see the value of the zookeeper.connect on the Kafka service Configuration page of the Kafka cluster. If you build your own Kafka cluster, the parmname `–zookeeper` parameter can be changed according to your actual configuration.

2. Define the data format description file for the data source. Name it metrics-kafka.json and place it in the current directory (or another directory that you have specified).

```
{
     "type": "kafka",
     "dataSchema": {
         "dataSource": "metrics-kafka",
         "parser": {
             "type": "string",
             "parseSpec": {
                 "timestampSpec": {
                     "column": "time",
                     "format": "auto"
                 },
                 "dimensionsSpec": {
                     "dimensions": ["url", "user"]
                 },
                 "format": "json"
             }
         },
         "granularitySpec": {
             "type": "uniform",
             "segmentGranularity": "hour",
             "queryGranularity": "none"
         },
         "metricsSpec": [{
                 "type": "count",
                 "name": "views"
             },
             {
                 "name": "latencyMs",
                 "type": "doubleSum",
                 "fieldName": "latencyMs"
             }
         ]
     },
     "ioConfig": {
         "topic": "metrics",
         "consumerProperties": {
             "bootstrap.servers": "emr-worker-1.cluster-xxxxxxxx:
9092 (the bootstrap.servers of your Kafka clusters)",
             "group.id": "kafka-indexing-service",
             "security.protocol": "SASL_PLAINTEXT",
             "sasl.mechanism": "GSSAPI"
         },
         "taskCount": 1,
         replicas: 1
         "taskDuration": "PT1H"
     },
     "tuningConfig": {
         "type": "Kafka",
         "maxRowsInMemory": "100000"
     }
 }
```

Note:

`ioConfig.consumerProperties.security.protocol` and `ioConfig.consumerProperties.sasl.mechanism` are security-related options and are not required for standard mode Kafka clusters.

3. Run the following command to add a Kafka supervisor.

```
curl --negotiate -u:druid -b ~/cookies -c ~/cookies -XPOST -H '
Content-Type: application/json' -d @metrics-kafka.json http://emr-
header-1.cluster-1234:18090/druid/indexer/v1/supervisor
```

The `–negotiate`, `-u`, `-b`, and `-c` options are for high-security mode Druid clusters.

4. Enable a console producer on the Kafka cluster.

```
--If the high-security mode of Kafka is enabled:
 export KAFKA_OPTS="-Djava.security.auth.login.config=/etc/ecm/kafka
-conf/kafka_client_jaas.conf"
 echo -e "security.protocol=SASL_PLAINTEXT\nsasl.mechanism=GSSAPI"
 > /tmp/Kafka/producer.conf
 --
 Kafka-console-producer.sh --producer.config /tmp/kafka/producer.
conf --broker-list emr-worker-1:9092, emr-worker-2:9092, emr-worker-3
:9092 --topic metrics
 >
```

The *–producer.config /tmp/Kafka/producer.conf* option is for high-security mode Kafka clusters.

5. Enter data at the command prompt of kafka_console_producer.

```
{"time": "2018-03-06T09:57:58Z", "url": "/foo/bar", "user": "alice",
 "latencyMs": 32}
 {"time": "2018-03-06T09:57:59Z", "url": "/", "user": "bob", "
latencyMs": 11}
 {"time": "2018-03-06T09:58:00Z", "url": "/foo/bar", "user": "bob",
 "latencyMs": 45}
```

   The timestamp can be generated with the following Python command:

```
python -c 'import datetime; print(datetime.datetime.utcnow().
strftime("%Y-%m-%dT%H:%M:%SZ"))'
```

6. Prepare a query file named metrics-search.json.

```
{
     "queryType" : "search",
     "dataSource" : "metrics-kafka",
     "intervals" : ["2018-03-02T00:00:00.000/2018-03-08T00:00:00.000
"],
     "granularity" : "all",
     "searchDimensions": [
         "url",
         "user"
     ],
     "query": {
         "type": "insensitive_contains",
         "value": "bob"
     }
 }
```

7. Execute the query on the master node of the Druid cluster.

```
curl --negotiate -u:Druid -b ~/cookies -c ~/cookies -XPOST -H '
Content-Type: application/json' -d @metrics-search.json http://emr-
header-1.cluster-1234:8082/druid/v2/?pretty
```

   The `–negotiate`, `-u`, `-b`, and `-c` options are for high-security mode Druid clusters.

8. You will see a query result similar to the following:

```
[ {
   "timestamp" : "2018-03-06T09:00:00.000Z",
   "result": {
     "dimension" : "user",
     "value" : "bob",
     "count": 2,
   } ]
```

```
  } ]
```

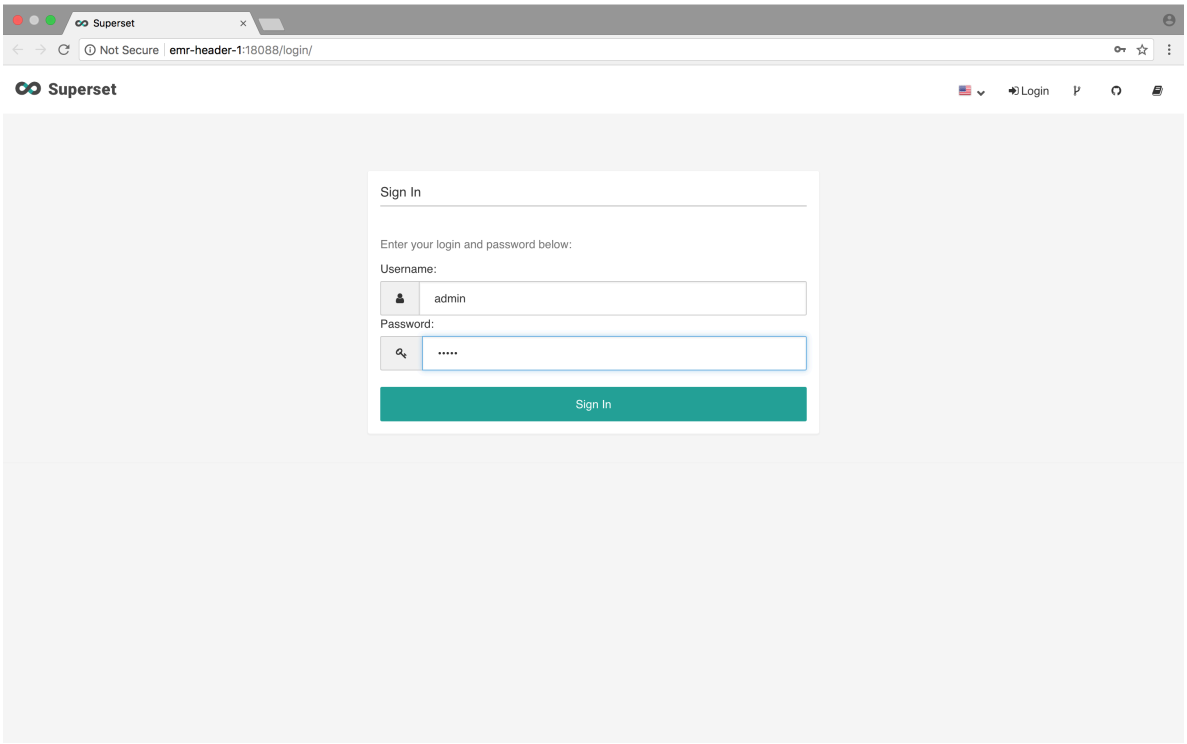
9.7.6 Superset

The Druid cluster integrates the Superset tool, which is integrated with Druid and supports a variety of relational databases. Because Druid supports SQL, you can access Druid through Superset in two ways: Druid's native query language or SQL.

Superset is installed in emr-header-1 by default and does not support high availability at present. Before you use this tool, make sure that your host can access emr-header-1. You can connect to the host by establishing the *SSH tunnel*.

1. Log on to the Superset

   Enter http://emr-header-1:18088 in your browser to go to the Superset logon page. The default username is admin and the default password is admin. When you log on for the first time, we strongly recommend changing your password.

   Not Secure | emr-header-1:18088/login/

   Sign In

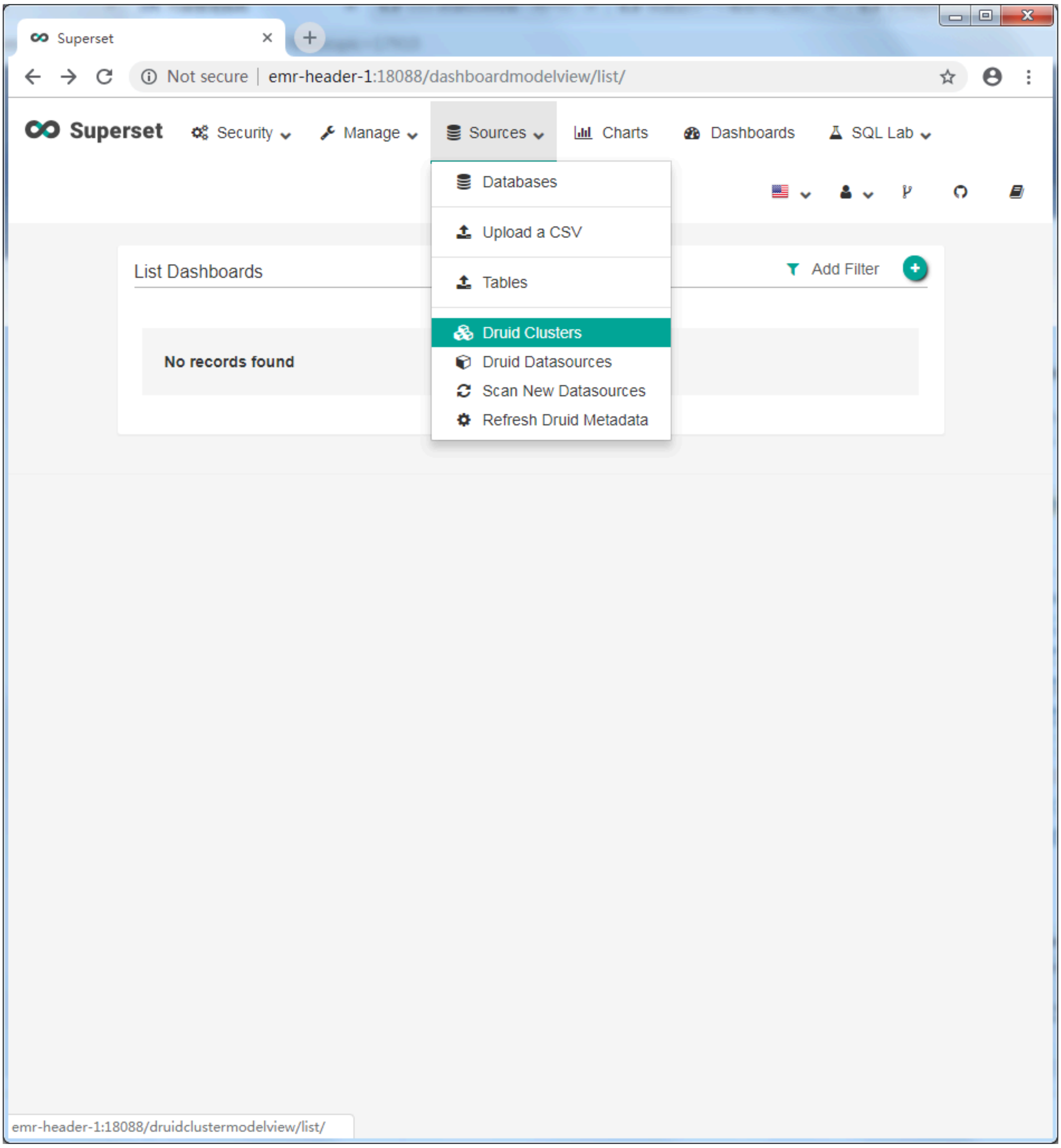
   Enter your login and password below:

   Username: admin

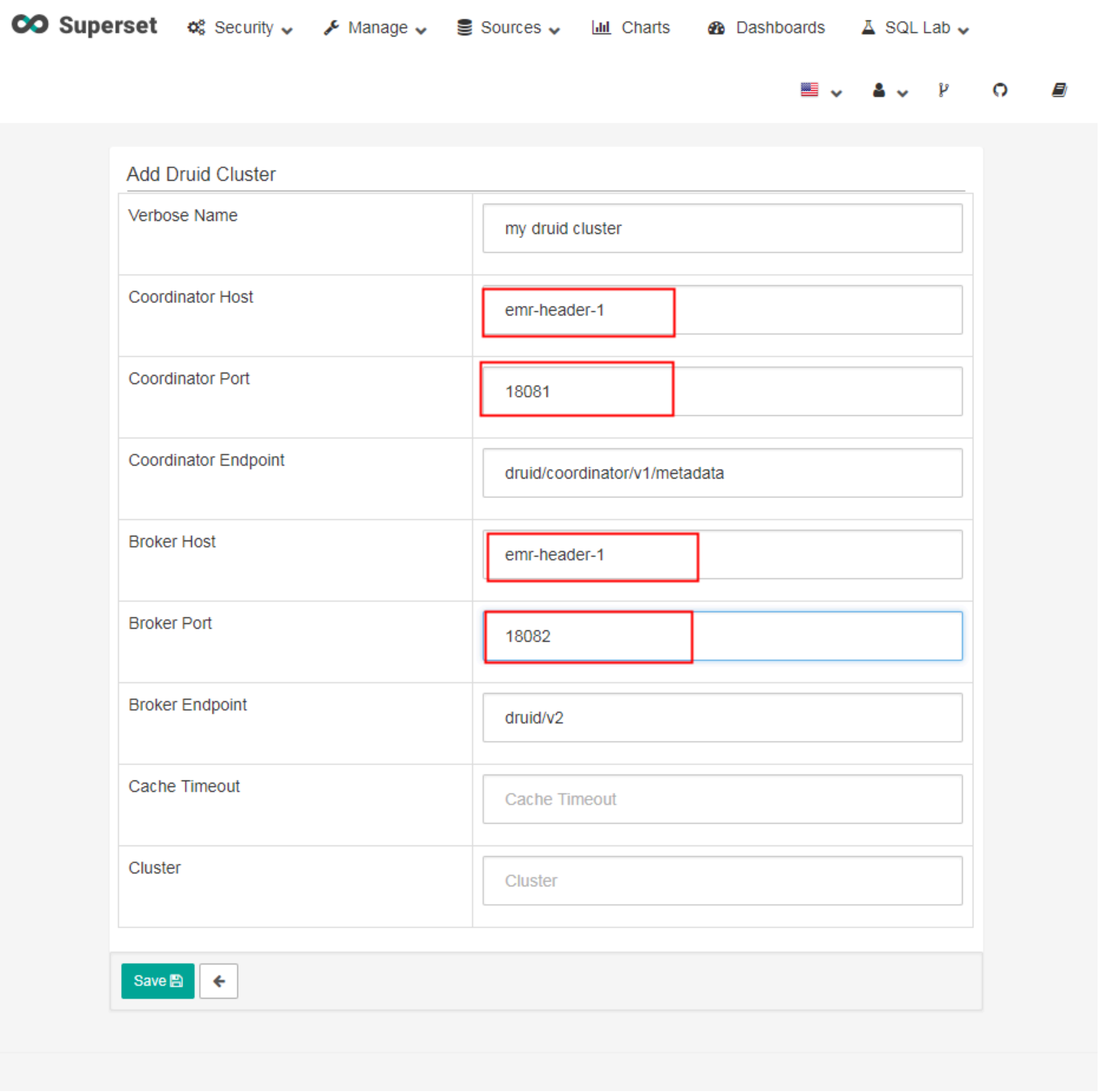
   Password: •••••

2. Add a Druid cluster

   The English interface is displayed by default. You can select the appropriate language by clicking the flag icon in the upper-right corner. In the menu bar along the top, select Data Source > Druid Cluster to add a Druid cluster.



Configure the addresses of the coordinator and broker. The default port number in E-MapReduce is the corresponding open source port number with "1" added in front. For example, if the open-source broker port number is 8082, the port number in E-MapReduce is 18082.



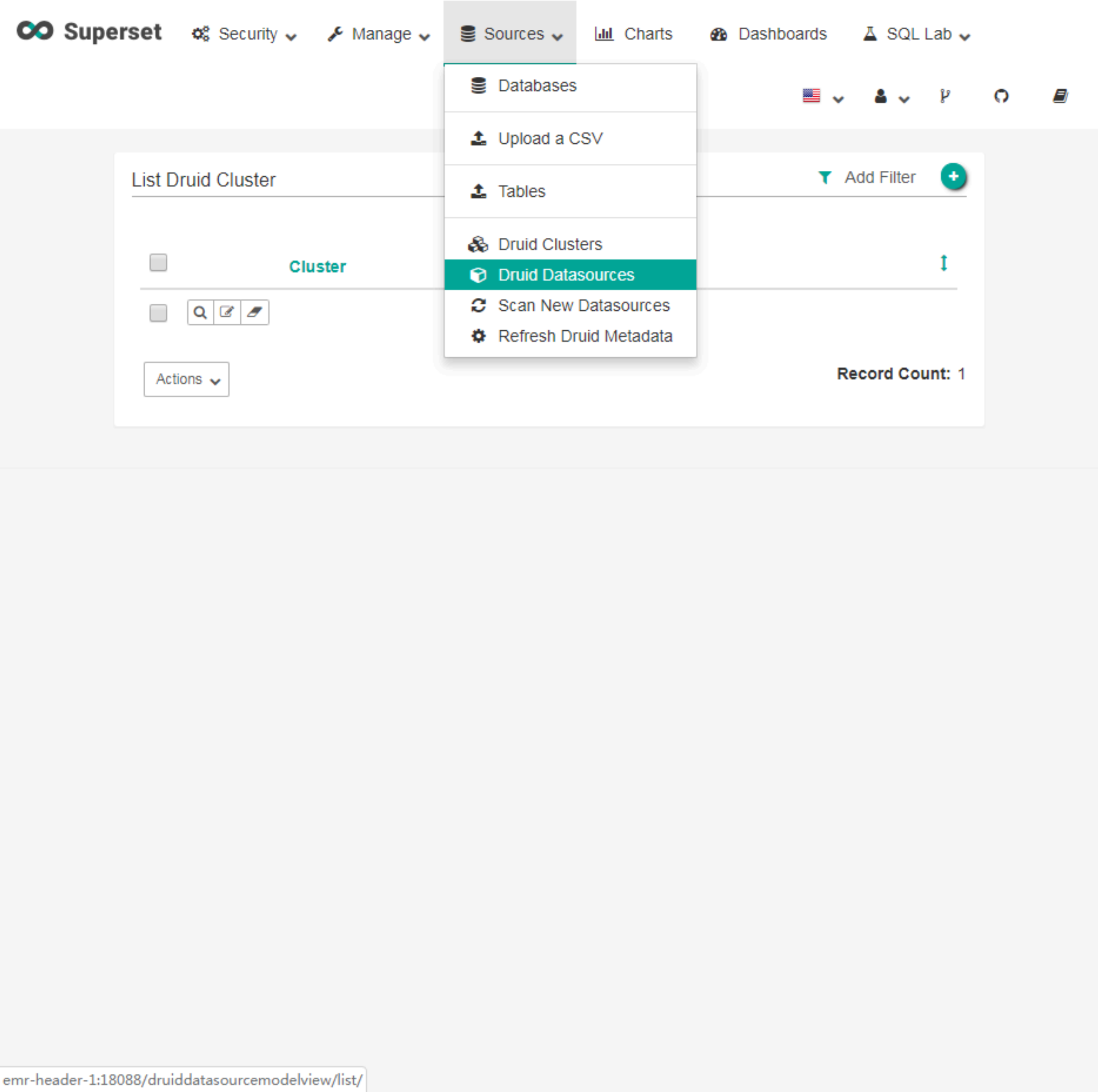
Add Druid Cluster

Verbose Name	my druid cluster
Coordinator Host	emr-header-1
Coordinator Port	18081
Coordinator Endpoint	druid/coordinator/v1/metadata
Broker Host	emr-header-1
Broker Port	18082
Broker Endpoint	druid/v2
Cache Timeout	Cache Timeout
Cluster	Cluster

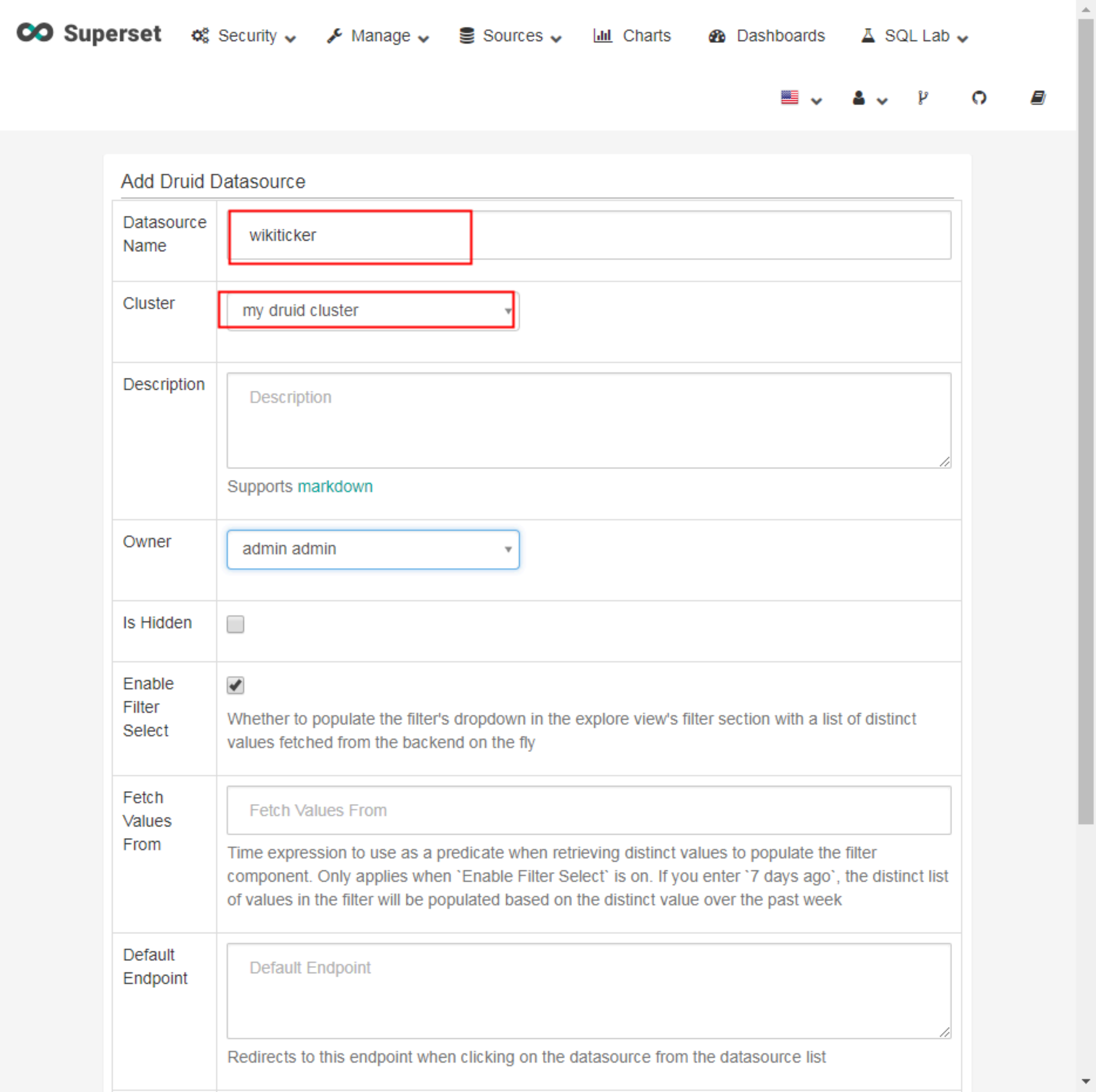
3. Refresh or add a new data source

   After adding the Druid cluster, you can click Data Source > Scan to add new data sources. The data sources on the Druid cluster loaded automatically.

   You can also customize a new data source by clicking Sources > Druid Datasources on the interface. (This operation is equivalent to writing a JSON file for data source ingestion.)



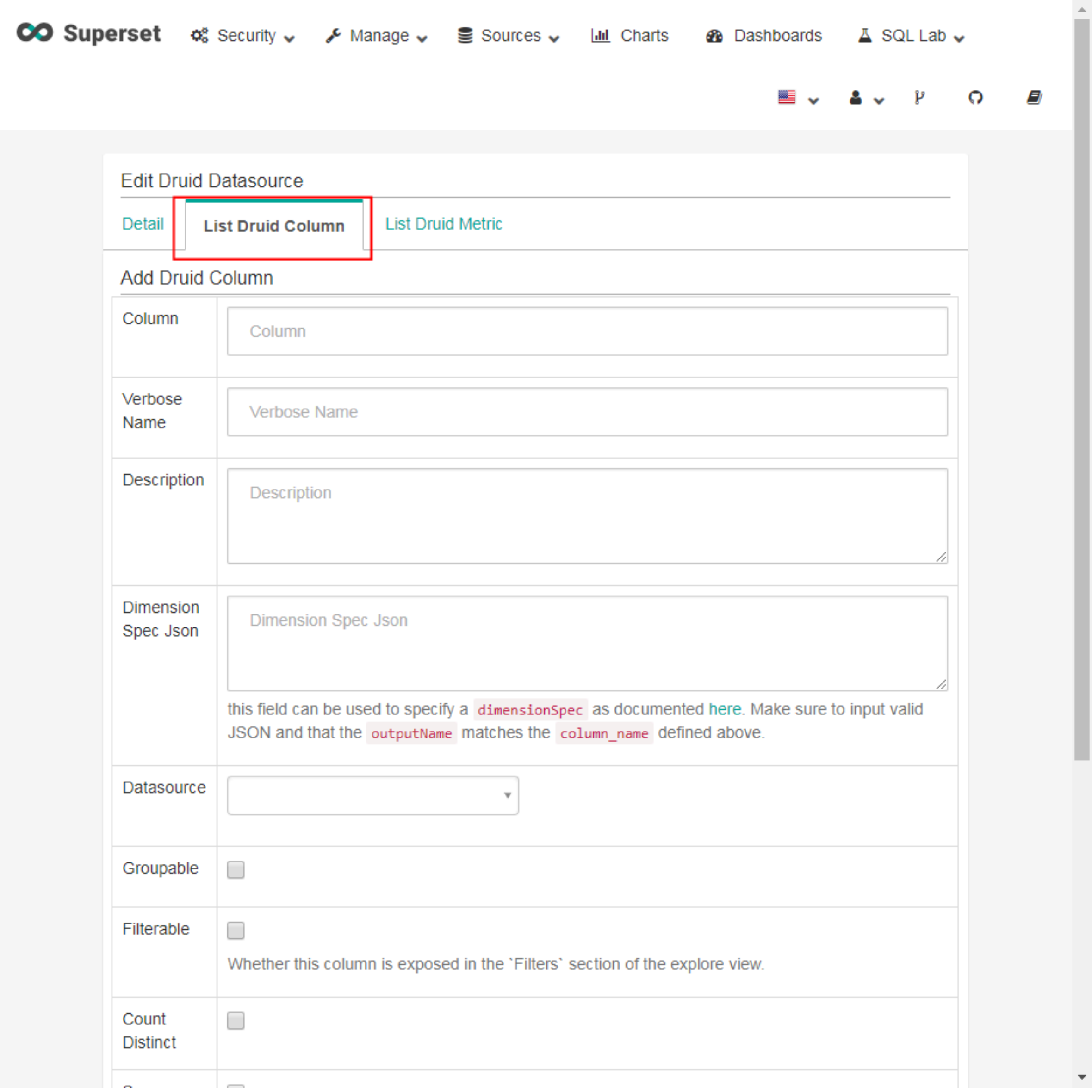
Enter the necessary information for custom data sources, and save it.



Add Druid Datasource

Datasource Name	wikiticker
Cluster	my druid cluster
Description	Description Supports markdown
Owner	admin admin
Is Hidden	□
Enable Filter Select	☑ Whether to populate the filter's dropdown in the explore view's filter section with a list of distinct values fetched from the backend on the fly
Fetch Values From	Fetch Values From Time expression to use as a predicate when retrieving distinct values to populate the filter component. Only applies when `Enable Filter Select` is on. If you enter `7 days ago`, the distinct list of values in the filter will be populated based on the distinct value over the past week
Default Endpoint	Default Endpoint Redirects to this endpoint when clicking on the datasource from the datasource list

Click the second of the three small icons on the left side to edit the data source. Enter the appropriate information, such as dimensions and metrics.



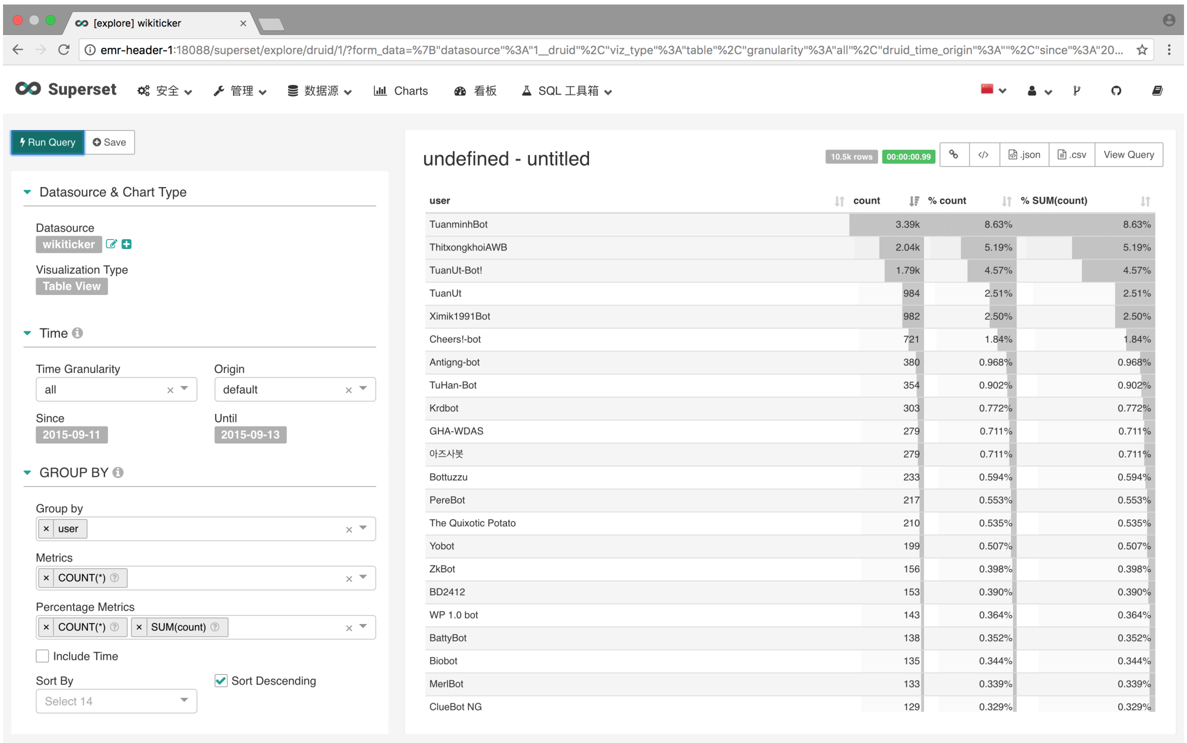
Edit Druid Datasource

Add Druid Column

Column	Column
Verbose Name	Verbose Name
Description	Description
Dimension Spec Json	Dimension Spec Json this field can be used to specify a `dimensionSpec` as documented here. Make sure to input valid JSON and that the `outputName` matches the `column_name` defined above.
Datasource	
Groupable	□
Filterable	□ Whether this column is exposed in the `Filters` section of the explore view.
Count Distinct	□

4. Query Druid

   After the data source has been added successfully, click it to go to the details page.


[explore] wikiticker
emr-header-1:18088/superset/explore/druid/1/?form_data=%7B"datasource"%3A"1__druid"%2C"viz_type"%3A"table"%2C"granularity"%3A"all"%2C"druid_time_origin"%3A""%2C"since"%3A"20...
Superset
安全
管理
数据源
Charts
看板
SQL 工具箱
Run Query
Save
10.5k rows
00:00:00.99
.json
.csv
View Query

Datasource & Chart Type

Datasource: wikiticker

Visualization Type: Table View

Time

Time Granularity: all

Origin: default

Since: 2015-09-11

Until: 2015-09-13

GROUP BY

Group by: user

Metrics: COUNT(*)

Percentage Metrics: COUNT(*) SUM(count)

☐ Include Time

Sort By: Select 14

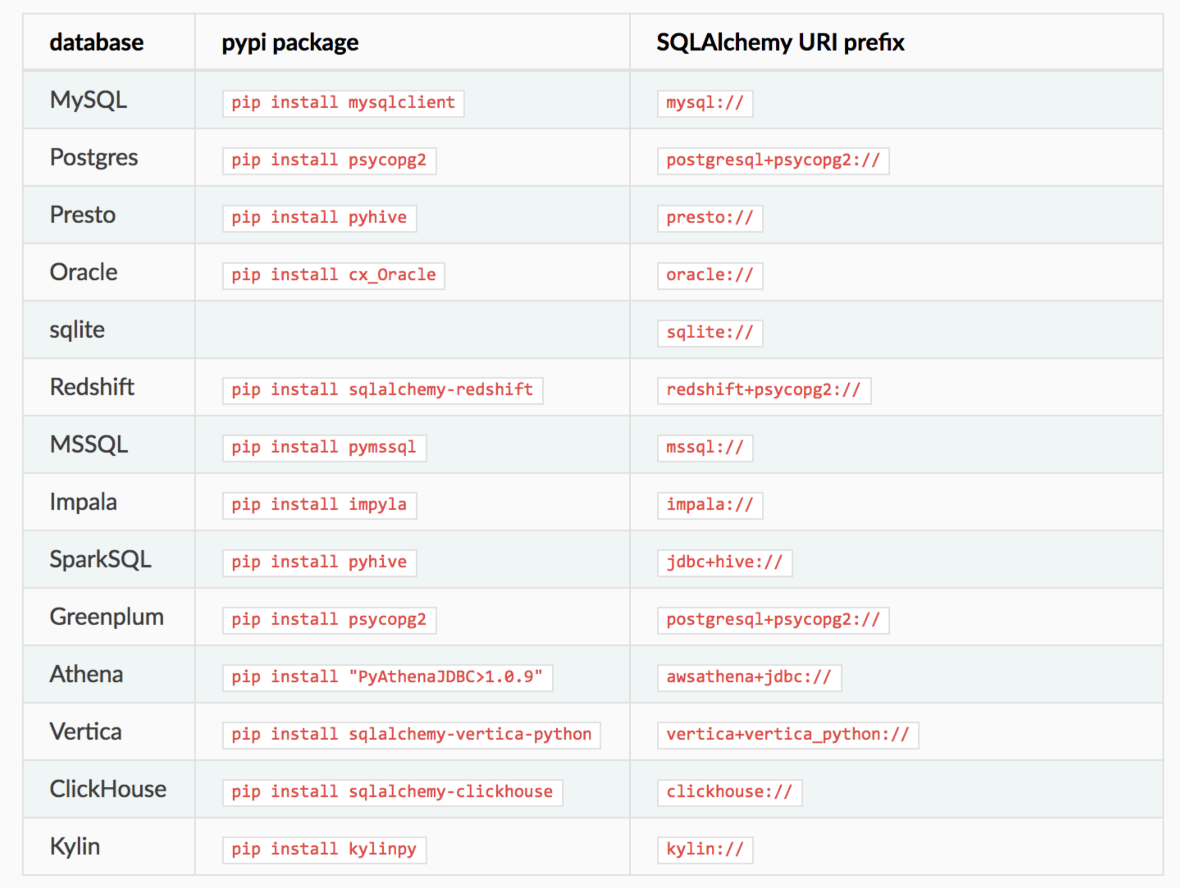
☑ Sort Descending

undefined - untitled

user	count	% count	% SUM(count)
TuanminhBot	3.39k	8.63%	8.63%
ThitxongkhoiAWB	2.04k	5.19%	5.19%
TuanUt-Bot!	1.79k	4.57%	4.57%
TuanUt	984	2.51%	2.51%
Ximik1991Bot	982	2.50%	2.50%
Cheers!-bot	721	1.84%	1.84%
Antigng-bot	380	0.968%	0.968%
TuHan-Bot	354	0.902%	0.902%
Krdbot	303	0.772%	0.772%
GHA-WDAS	279	0.711%	0.711%
아즈사봇	279	0.711%	0.711%
Bottuzzu	233	0.594%	0.594%
PereBot	217	0.553%	0.553%
The Quixotic Potato	210	0.535%	0.535%
Yobot	199	0.507%	0.507%
ZkBot	156	0.398%	0.398%
BD2412	153	0.390%	0.390%
WP 1.0 bot	143	0.364%	0.364%
BattyBot	138	0.352%	0.352%
Biobot	135	0.344%	0.344%
MerlBot	133	0.339%	0.339%
ClueBot NG	129	0.329%	0.329%

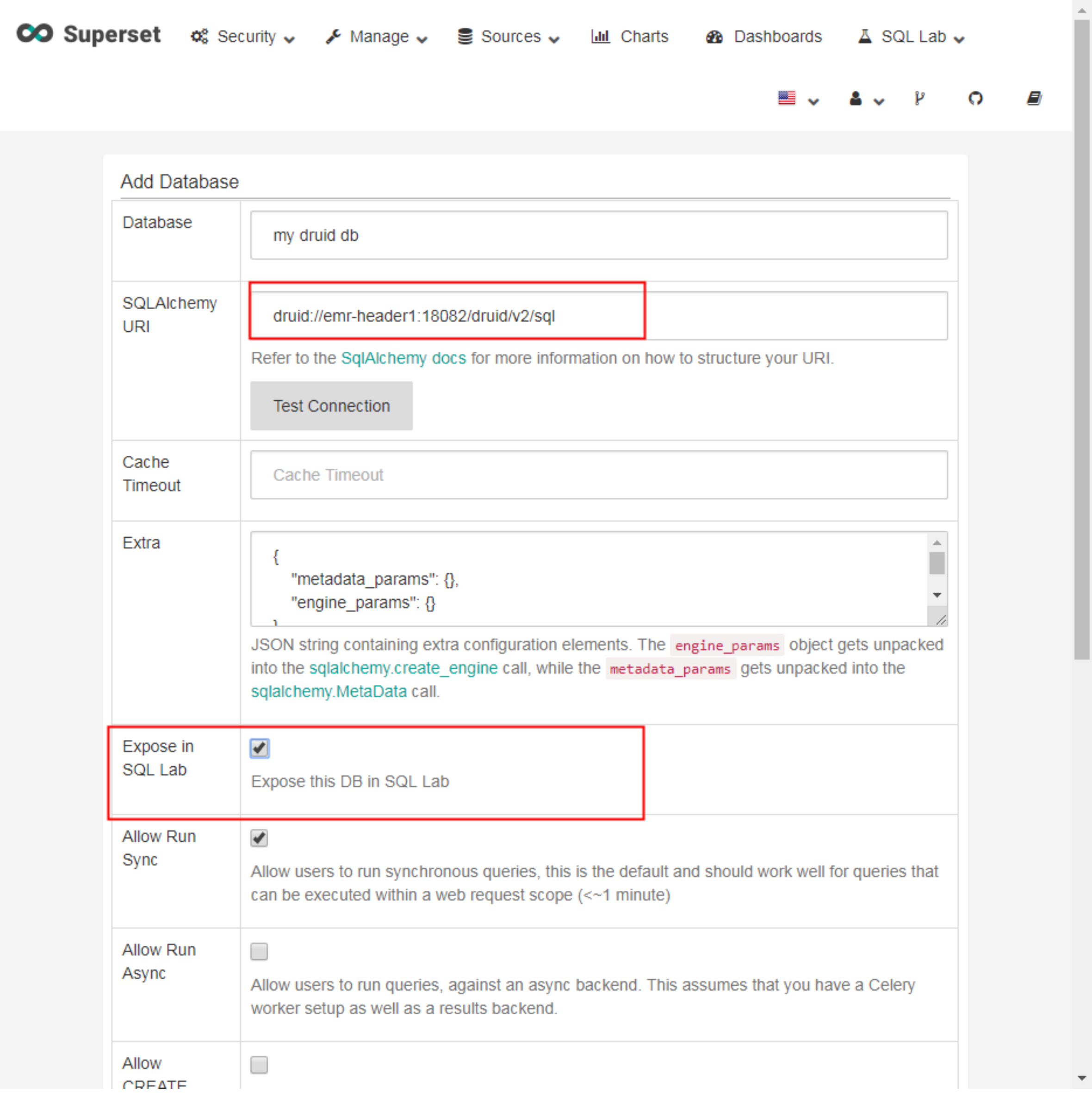
5. (Optional) Use Druid as a database

Superset provides SQLAlchemy to support a wide variety of databases with various dialects, as shown in the following figure.



database	pypi package	SQLAlchemy URI prefix
MySQL	`pip install mysqlclient`	`mysql://`
Postgres	`pip install psycopg2`	`postgresql+psycopg2://`
Presto	`pip install pyhive`	`presto://`
Oracle	`pip install cx_Oracle`	`oracle://`
sqlite		`sqlite://`
Redshift	`pip install sqlalchemy-redshift`	`redshift+psycopg2://`
MSSQL	`pip install pymssql`	`mssql://`
Impala	`pip install impyla`	`impala://`
SparkSQL	`pip install pyhive`	`jdbc+hive://`
Greenplum	`pip install psycopg2`	`postgresql+psycopg2://`
Athena	`pip install "PyAthenaJDBC>1.0.9"`	`awsathena+jdbc://`
Vertica	`pip install sqlalchemy-vertica-python`	`vertica+vertica_python://`
ClickHouse	`pip install sqlalchemy-clickhouse`	`clickhouse://`
Kylin	`pip install kylinpy`	`kylin://`

Superset also supports accessing Druid in this way. The corresponding SQLAlchemy URI of Druid is druid://emr-header-1:18082/druid/v2/sql. When you add Druid as a database, check the "Expose in SQL Lab" check box.



Add Database

Database	my druid db
SQLAlchemy URI	druid://emr-header1:18082/druid/v2/sql Refer to the SqlAlchemy docs for more information on how to structure your URI.
Cache Timeout	Cache Timeout

Extra	{
	"metadata_params": {},
	"engine_params": {}
	JSON string containing extra configuration elements. The `engine_params` object gets unpacked into the sqlalchemy.create_engine call, while the `metadata_params` gets unpacked into the sqlalchemy.MetaData call.

Expose in SQL Lab	☑ Expose this DB in SQL Lab
Allow Run Sync	☑ Allow users to run synchronous queries, this is the default and should work well for queries that can be executed within a web request scope (<~1 minute)
Allow Run Async	□ Allow users to run queries, against an async backend. This assumes that you have a Celery worker setup as well as a results backend.
Allow CREATE	□

You can now use SQL to query in the SQL toolkit.

9.7.7 Common Druid problems

This section describes some of the common problems you may encounter with Druid.

Analyze the indexing failure

If indexing fails, complete the following steps to troubleshoot the failure:

- For batch data indexing

1. If the curl command output displays an errort or does not display any information, check the file format. Alternatively, add the `-v` parameter to the curl command to check the value returned from the RESTful API.
2. Observe the execution of jobs on the Overlord page. If the execution fails, view the logs on that page.
3. In many cases, logs are not generated. In the case of a Hadoop job, open the YARN page to check whether an index job has been generated.
4. If no errors are found, you need to log on to the Druid cluster and view the execution logs of Overlord at */mnt/disk1/log/druid/overlord—emr-header-1.cluster-xxxx.log*. In the case of an HA cluster, check the Overlord that you submitted the job to.
5. If the job has been submitted to Middlemanager but a failure is returned from Middlemanager, you need to view the worker that the job is submitted to in Overlord, and log on to the worker node to view the Middlemanager logs (at */mnt/disk1/log/druid/middleManager-emr-header-1.cluster-xxxx.log*).

- For real-time Tranquility indexing

  Check the Tranquility log to see if the message was received or dropped.

  The remaining troubleshooting steps are the same as steps 2 to 5 of batch indexing.

  Most errors are about cluster configurations and jobs. Cluster configuration errors are about memory parameters, cross-cluster connection, access to clusters in high-security mode, and principals. Job errors are about the format of the job descriptio n files, input data parsing, and other job-related configuration issues (such as ioConfig).

Obtain the FAQ list

- Service startup fails.

  Most of these problems are due to configuration problems with the running parameters of the JVM component. For example, the machine may not have a large memory, but it is configured with a larger JVM memory or a larger number of threads.

  To resolve this issue, view the component logs and adjust the relevant parameters. JVM memory involves heap memory and direct memory. For more information, go to *Druid Performance FAQ*.

- The YARN task fails during indexing and shows a JAR package conflict error like this: `Error: class com.fasterxml.jackson.datatype.guava.deser.HostAndPortDeserializer overrides final method deserialize.( Lcom/fasterxml/jackson/core/JsonParser;Lcom/fasterxml/jackson/databind/DeserializationContext;)Ljava/lang/Object;`.

  To resolve this issue, add the following content to the indexing job configuration file:

  ```
  "tuningConfig" : {
        ...
        "jobProperties" : {
            "mapreduce.job.classloader": "true"
            or
            "mapreduce.job.user.classpath.first": "true"
        }
        ...
   }
  ```

  The parameter `mapreduce.job.classloader` allows MapReduce jobs to use standalone classloaders, and the parameter `mapreduce.job.user.classpath.first` gives MapReduce the priority to use your JAR packages. You can select one of these two configuration items. For more information, go to *Druid documents*.
- The logs of the index task report that the reduce task cannot create the segments directory.

  To resolve this issue, complete the following:

  - Check the settings for deep storage, including type and directory. If the type is local, pay attention to the permission settings of the directory. If the type is HDFS, the directory should be written as the full HDFS path, such as hdfs://:9000/. For hdfs_master, IP is recommended. If you want to use a domain name, use the full domain name, such as emr-header-1.cluster-xxxxxxxx rather than emr-header-1.
  - If you are using Hadoop for batch indexing, you must set the deep storage of segments as "hdfs". The local type may cause the MapReduce job to be in an unidentified state, because the remote YARN cluster cannot create the segments directory in the reduce task. (This is only applicable to standalone Druid clusters.)
- Failed to create directory within 10,000 attempts.

This issue occurs typically because the path set by java.io.tmp in the JVM configuration file does not exist. Set the path and make sure that the Druid account has permission to access it.

- com.twitter.finagle.NoBrokersAvailableException: No hosts are available for disco! firehose:druid:overlord

  This issue is typically due to ZooKeeper connection issues. Make sure that Druid and Tranquility have the same connection string for ZooKeeper. Because the default ZooKeeper path for Druid is /druid, make sure that zookeeper.connect in the Tranquility settings includes /druid. (Two ZooKeeper settings exist in Tranquility Kafka. One is zookeeper.connect used to connect the ZooKeeper of the Druid cluster, and the other is kafka.zookeeper.connect used to connect the ZooKeeper of the Kafka cluster. These two ZooKeepers may not belong to the same ZooKeeper cluster.)
- The MiddleManager reports that the com.hadoop.compression.lzo.LzoCodec class cannot be found during indexing.

  This is because the Hadoop cluster of E-MapReduce is configured with lzo compression.

  To resolve this issues, copy the JAR package and the native file under the EMR HADOOP_HOME/lib directory to Druid's druid.extensions.hadoopDependenciesDir (by default, DRUID_HOME/hadoop-dependencies).
- The following error is reported during indexing:

  ```
  2018-02-01T09:00:32,647 ERROR [task-runner-0-priority-0] com.hadoop.
  compression.lzo.GPLNativeCodeLoader - could not unpack the binaries
    java.io.IOException: No such file or directory
            at java.io.UnixFileSystem.createFileExclusively(Native
  Method) ~[?:1.8.0_151]
            at java.io.File.createTempFile(File.java:2024) ~[?:1.8.
  0_151]
            at java.io.File.createTempFile(File.java:2070) ~[?:1.8.
  0_151]
            at com.hadoop.compression.lzo.GPLNativeCodeLoader.
  unpackBinaries(GPLNativeCodeLoader.java:115) [hadoop-lzo-0.4.21-
  SNAPSHOT.jar:?]
  ```

  This issue occurs because the java.io.tmp path does not exist. Set the path and make sure that the Druid account has permission to access it.

9.8 Presto

9.8.1 Connector

System connector

- Overview

  You can use SQL to query the basic information and measurements of a Presto cluster through the connector.

- Configuration

  All information can be obtained through a catalog called "system". Configuration is not necessary.

  Examples:

```
--- List all supported data entries
SHOW SCHEMAS FROM system;
--- List all data entries in the project during runtime
SHOW TABLES FROM system.runtime;
--- Obtain node status
SELECT * FROM system.runtime.nodes;
     node_id |         http_uri         | node_version | coordinator
 | state
-------------+--------------------------+--------------
+-------------+--------
 3d7e8095-...| http://192.168.1.100:9090| 0.188        | false
 | active
 7868d742-...| http://192.168.1.101:9090| 0.188        | false
 | active
 7c51b0c1-...| http://192.168.1.102:9090| 0.188        | true
 | active
--- Force cancel a query
CALL system.runtime.kill_query('20151207_215727_00146_tx3nr');
```

- Data tables

  The connector provides the following data tables:

Table	Schema	Description
catalogs	metadata	This table contains the list of all catalogs supported in the connector.
schema_properties	metadata	This table contains the list of available properties that can be set when creating a schema.
table_properties	metadata	This table contains the list of available properties that can be set when creating a table.

Table	Schema	Description
nodes	runtime	This table contains the list of all visible nodes and their statuses in the Presto cluster.
queries	runtime	This table contains information queries currently and recently initiated in the Presto cluster, including the original query texts (SQL), identities of the users who initiate the queries , and information about query performances, such as query queues and analysis times.
tasks	runtime	This table contains information on the task involved in the queries in the Presto cluster, including the locations and numbers of lines and bytes processed in each task.
transactions	runtime	This table contains the list of currently opened transactions and related metadata. The data includes information such as creation time, idle time, initiation parameters, and access catalogs.

- Stored procedures

  The connector supports the following stored procedures:

  `runtime.kill_query(id)` cancels queries from a specified ID.

JMX connector

- Overview

  You can query JMX information for all nodes in the Presto cluster through the JMX connector. The connector is generally used for system monitoring and debugging. To implement regular dumps of JMX information, modify the configuration of the connector.

- Configuration

  Create file *etc/catalog/jmx.properties*, add the following content, and enable the JMX connector.

  ```
  connector.name=jmx
  ```

  If a regular dump of JMX data is expected, the following content can be added to the configuration file:

  ```
  connector.name=jmx
  jmx.dump-tables=java.lang:type=Runtime,com.facebook.presto.execution
  .scheduler:name=NodeScheduler
  ```

```
jmx.dump-period=10s
jmx.max-entries=86400
```

Where:

- `dump-tables` is a list of MBeans (Managed Beans) separated by commas. This configuration specifies which MBeans are sampled and stored in the memory for each sample period.
- `dump-period` is used for setting the sample period, which is 10s by default.
- `max-entries` is used for setting the maximum length of the history, which is 86400 by default.

If the name of a metric contains a comma, use `\,` to escape as follows:

```
connector.name=jmx
jmx.dump-tables=com.facebook.presto.memory:type=memorypool\\,name=
general,\
   com.facebook.presto.memory:type=memorypool\\,name=system,\
   com.facebook.presto.memory:type=memorypool\\,name=reserved
```

- Data tables

  JMX connector provides 2 schemas, current and history, where:

  - current contains the current MBean in each node, the name of which is the table name in current. If the bean name contains non-standard characters, the table name must be in quotation marks for the query. The table name can be obtained through the following statement:

    ```
    SHOW TABLES FROM jmx.current;
    ```

    Examples:

    ```
    --- Obtain jvm information for each node
    SELECT node, vmname, vmversion
    FROM jmx.current."java.lang:type=runtime";
          node    |              vmname               | vmversion
    --------------+-----------------------------------+-----------
     ddc4df17-xxx | Java HotSpot(TM) 64-Bit Server VM | 24.60-b09
    (1 rows)
    ```

    ```
    --- Obtain the metrics of max and min file descriptor counts for
    each node
    SELECT openfiledescriptorcount, maxfiledescriptorcount
    FROM jmx.current."java.lang:type=operatingsystem";
     openfiledescriptorcount | maxfiledescriptorcount
    -------------------------+------------------------
                         329 |                  10240
    ```

```
(1 rows)
```

- history contains the data table corresponding to the metrics to be dumped in the configuration file. Use the following statement to query:

```
SELECT "timestamp", "uptime" FROM jmx.history."java.lang:type=
runtime";
        timestamp        | uptime
-------------------------+--------
 2016-01-28 10:18:50.000 |  11420
 2016-01-28 10:19:00.000 |  21422
 2016-01-28 10:19:10.000 |  31412
(3 rows)
```

Kafka connector

- Overview

  The connector maps topics in Kafka to tables in Presto. Each record in Kafka is mapped to a message in a Presto table.

- Configuration

  Create file *etc/catalog/kafka.properties*, add the following content, and enable the Kafka connector.

```
connector.name=kafka
kafka.table-names=table1,table2
kafka.nodes=host1:port,host2:port
```

Note:
Presto can connect to multiple Kafka clusters by adding a new properties file to the configuration catalog. The file name is then mapped to the Presto catalog. For example, when configuration file *orders.properties* is added, Presto creates a catalog named "orders".

```
## orders.properties
connector.name=kafka  # It denotes the connector type, which cannot
 be changed
kafka.table-names=tableA,tableB
kafka.nodes=host1:port,host2:port
```

Kafka connector provides the following properties:

- kafka.table-names

  Description: Required. This defines the list of tables supported in the connector.

Details: The file name can be modified using the schema name, with forms like *{schema_name}.{table_name}*. If the file name is not modified using the schema name, the table is mapped to the schema defined in *kafka.default-schema* .

- kafka.default-schema

  Description: Optional. This is the default schema name, with the default value `default`.

- kafka.nodes

  Description: Required. This is the node list in the Kafka cluster.

  Details: The configuration form is `hostname:port[,hostname:port…]`. You can only configure a part of the Kafka nodes here, but Presto must be capable of connecting to all nodes in the Kafka cluster. Otherwise, some data may not be obtained.

- kafka.connect-timeout

  Description: Optional. This is the timeout for the connector and the Kafka cluster, which is 10s by default.

  Details: If there is a large amount of pressure on the Kafka cluster, it may take a long time to create a connection, causing a timeout when executing the query by Presto. If this is the case, we recommend that you add the configured value.

- kafka.buffer-size

  Description: Optional. This is the read buffer size, which is 64 KB by default.

  Details: You can use this to set the size of the buffer for internal data reads from Kafka. The data buffer must be larger than a message. A data buffer is distribute d for each worker and data node.

- kafka.table-description-dir

  Description: Optional. This is the catalog of topic (table) description files, which is *etc/kafka* by default.

  Details: Data table definition files in JSON format are stored in this directory (. json has to be used as a suffix).

- kafka.hide-internal-columns

  Description: Optional. This is the list of preset columns to be hidden, the value of which is `true` by default.

Details: In addition to data columns defined in the table description file, the connector also maintains a number of extra columns for each table. The property is used to control whether these columns are shown in the execution result of the `DESCRIBE` and `SELECT *` statements. Regardless of the setting in the configuration, these columns are involved in the query process.

The Kafka connector provides the following internal columns:

Column name	Type	Description
_partition_id	BIGINT	The ID of the Kafka partition where the record is located.
_partition_offset	BIGINT	The offset of the Kafka partition where the record is located.
_segment_start	BIGINT	The lowest offset containing this data segment. This offset is for each partition.
_segment_end	BIGINT	The largest offset containing this data segment (which is the start offset of the next segment). This offset is for each partition.
_segment_count	BIGINT	The serial number of the column in this segment. For an uncompressed topic, `_segment_start` + `_segment_count` = `_partition_offset`.
_message_corrupt	BOOLEAN	This field is set to `TRUE` if a decoder cannot decode the record.
_message	VARCHAR	A string coded with UTF-8 from the message bytes. If the type of the topic message is text, this field is useful.
_message_length	BIGINT	The byte length of the message.
_key_corrupt	BOOLEAN	This field is set to `TRUE` if a key decoder cannot decode the record.
_key	VARCHAR	A string coded with UTF-8 from the key bytes. If the type of the topic message is text, this field will be useful.
_key_length	BIGINT	The byte length of the key.

Note:

For tables without definition files, `_key_corrupt` and `_message_corrupt` remain `FALSE`.

- Table definition files

  Kafka is a schema-less message system. The formats of the messages must defined by the producers and consumers. Presto also requires that data be capable of being mapped into tables. Therefore, you must provide corresponding table definition files based on the actual use of the messages. For messages in JSON format, if a definition file is not provided, JSON functions in Presto can be used for resolution in queries. While the method is flexible, it increases the difficulty of writing SQL statements.

  When JSON is used to define a table in a table definition file, the file name can be customized. The extension must be *.json.

```
{
    "tableName": ...,
    "schemaName": ...,
    "topicName": ...,
    "key": {
        "dataFormat": ...,
        "fields": [
            ...
        ]
    },
    "message": {
        "dataFormat": ...,
        "fields": [
            ...
       ]
    }
}
```

Field	Required	Type	Description
tableName	Yes	string	The name of the Presto table.
schemaName	No	string	The name of the schema where the table is located.
topicName	Yes	string	The name of the Kafka topic.
key	No	JSON object	The rules for mapping from message keys to columns.
message	No	JSON object	The rules for mapping from messages to columns.

  The mapping rules for keys and messages use the following fields for description.

Field	Required	Type	Description
dataFormat	Yes	string	A decoder for setting a group of columns.
fields	Yes	JSON array	A column definition list.

`fields` here is a JSON array, and each element is a JSON object as follows:

```
{
    "name": ...,
    "type": ...,
    "dataFormat": ...,
    "mapping": ...,
    "formatHint": ...,
    "hidden": ...,
    "comment": ...
}
```

Field	Required	Type	Description
name	Yes	string	The name of the column.
type	Yes	string	The column data type.
dataFormat	No	string	The column data decoder.
mapping	No	string	The decoder parameters.
formatHint	No	string	The hint set for the column , which can be used by the decoder.
hiddent	No	boolean	Indicates whether it is hidden or not.
comment	No	string	The description of the column.

- Decoder

  The function of the decoder is to map Kafka messages (key + message) to the columns in the data tables. In the absence of table definition files, Presto uses the dummy decoder.

  The Kafka connector provides the following decoders:

  - raw: Original bytes are used without conversion.
  - csv: Messages are processed as strings in CSV format.
  - json: Messages are processed as strings in JSON format.

9.8.2 What is Presto?

Presto is an open-source distributed SQL-on-Hadoop query engine powered by Facebook. It is currently maintained by the open source community and Facebook engineers, and has derived multiple commercial versions.

Basic features

Presto is implemented in Java. It is easy to use and offers high performance and strong scalability. Its other features are as follows:

- Fully supports ANSI SQL.
- Supports rich data sources, accessing them as follows:
  - Interaction with Hive
  - Cassandra
  - Kafka
  - MongoDB
  - MySQL
  - PostgreSQL
  - SQL Server
  - Redis
  - Redshift
  - Local files
- Supports advanced data structures.
  - Array and map data
  - JSON data
  - GIS data
  - Color data
- Presto provides the following expansion configurations:
  - Data connector expansion
  - Custom data types
  - Custom SQL functions

  To achieve efficient service processes, you can expand the corresponding modules according to your own service features.
- Based on the Pipeline process model, data is returned to you in real time.
- Improved monitoring interfaces.

  - Friendly WebUI is provided to present the execution processes of the query tasks visually.
  - Supports the JMX protocol.

Scenarios

Presto is a distributed SQL engine that is well-suited to the following scenarios:

- ETL
- Ad-hoc queries
- Massive structured and semi-structured data analysis
- Massive multi-dimensional data aggregation/reports

In particular, Presto is a data warehouse product, which is not designed to replace traditional RDBMS databases such as MySQL and PostgreSQL. It has limited support for transactions and is not suitable for online service scenarios.

Benefits

In addition to being open source, the E-MapReduce Presto product comes with the following advantages:

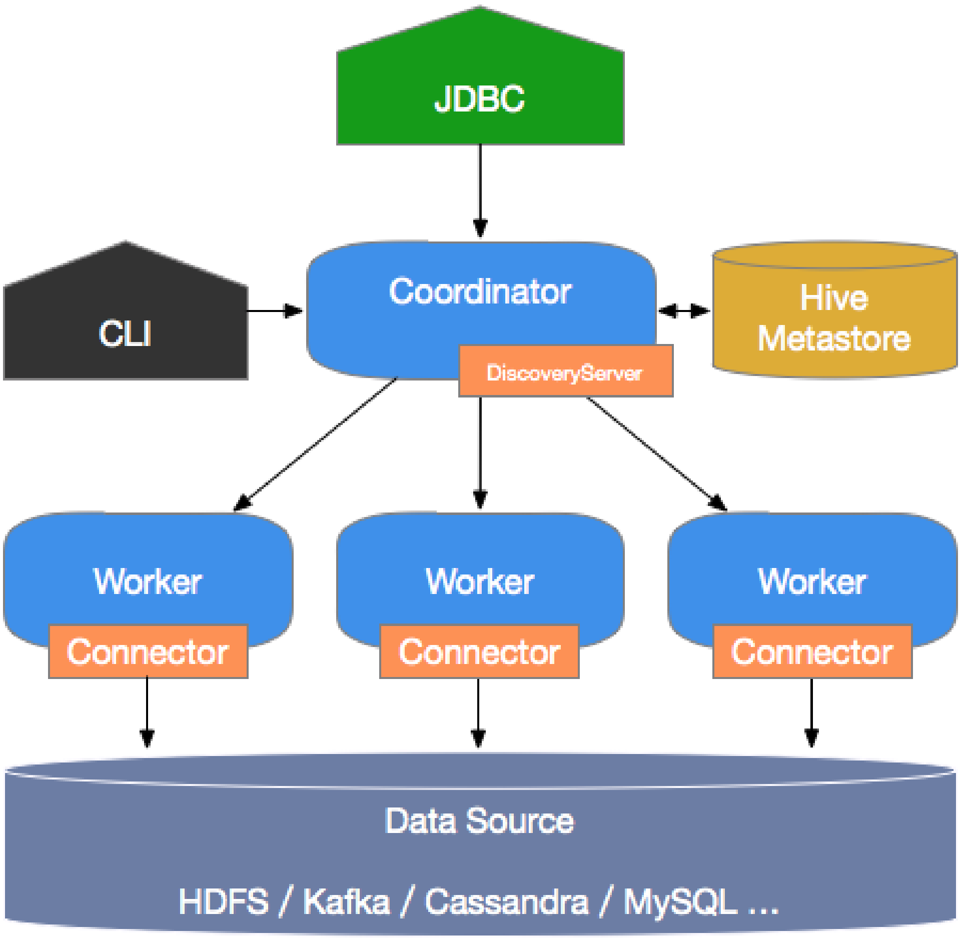
- You can purchase it for immediate use to build a Presto cluster with hundreds of nodes in minutes.
- It supports elastic scalability, meaning that you can scale the cluster up and down with simple operations.
- It works perfectly in connection with the E-MapReduce software stacks, and supports the processing of data stored in OSS.
- O&M is free 24/7, providing an all-in-one service.

9.8.3 Quick start

This section provides an overview of Presto and describes to use it to develop applications.

Architecture

The following figure shows the architecture of Presto:



Presto has a typical mobile/server architecture comprising a coordinator node and multiple worker nodes. Coordinator is responsible for the following:

- Receiving and parsing your query requests, generating execution plans, and sending the execution plans to the worker nodes for execution.
- Monitoring the running status of the worker nodes. Each worker node maintains a heartbeat connection with the coordinator node, reporting the node statuses.
- Maintaining the metastore data

Worker nodes run the tasks assigned by the coordinator node, read data from external storage systems through connectors, process the data, and send the results to the coordinator node.

Basic concepts

The basic concepts of Presto are as follows:

- Data model

  The data model indicates the data organization form. To manage data, Presto uses a three-level structure that consists of catalogs, schemas, and tables.

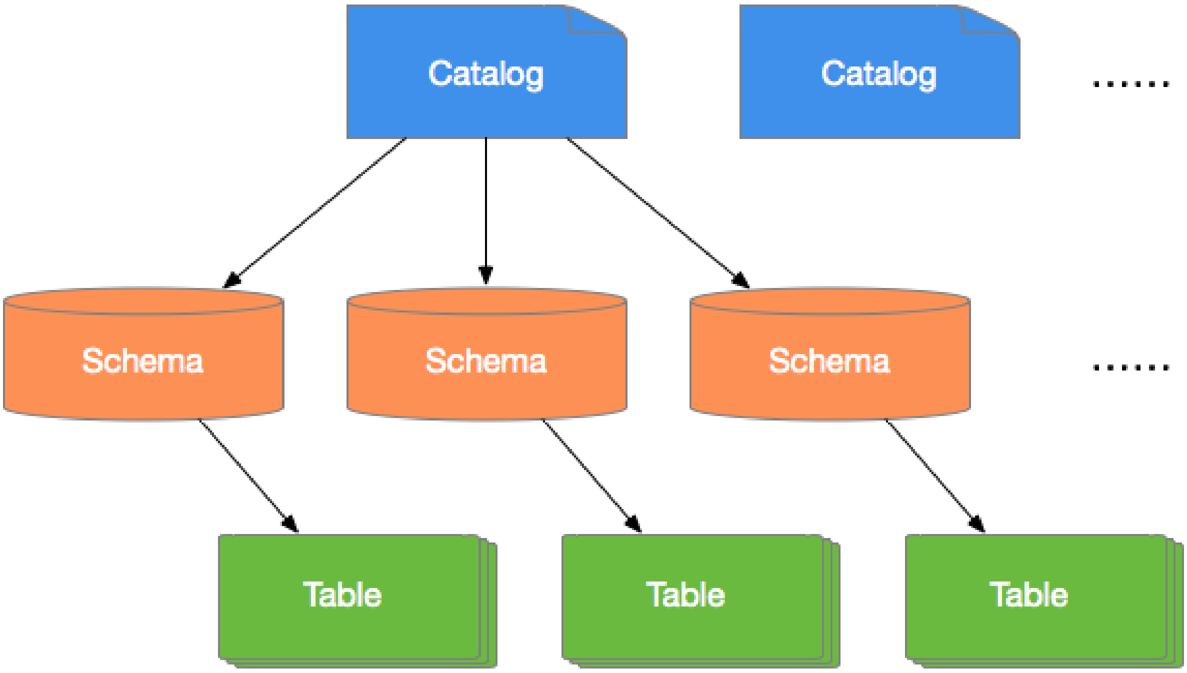
- Catalog

  A catalog contains multiple schemas and is physically directed to an external data source, which can be accessed through connectors. When you run an SQL statement in Presto, you are running it against one or more catalogs.
- Schema

  A schema is a database instance that contains multiple data tables.
- Table

  A data table is the same as a general database table.

The relationships between catalogs, schemas, and tables are shown in the following figure.



- Connector

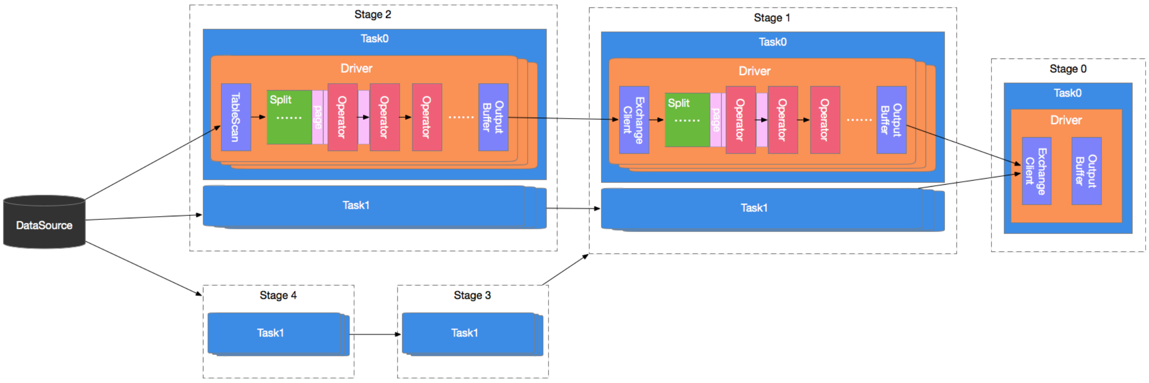
  Presto uses connectors to connect to various external data sources. To access customized data sources, Presto provides a standard *SPI*, which allows you to develop your own connectors using this standard API.

  A catalog is typically associated with a specific connector (which can be configured in the Properties file of the catalog). Presto contains multiple built-in connectors.
- Query-related concepts
  - Statement

    Statement refers to an SQL statement that you enter via JDBC or CLI.

- Query

  Query refers to the execution process of a query. When Presto receives an SQL statement, the coordinator parses this statement, generates an execution plan, and sends this plan to a worker for execution. A query is logically made up of several components, namely stages, tasks, drivers, splits, operators, and data sources, which are shown in the following figure.



- Stage

  A Presto query contains multiple stages. A stage is a logical concept that indicates the stage of a query process, and comprises one or more execution tasks. Presto uses a tree-like structure to organize stages, the root node of which is Single Stage. This stage aggregates data output from the upstream stages and sends the results to the coordinator. The leaf node of this tree is Source Stage. This stage receives data from the connector for processing.

- Task

  A task refers to a specific task to be executed and is the smallest Presto task scheduling unit. During the execution process, the Presto task scheduler distributes these tasks to individual workers for execution. Tasks in one stage can be executed in parallel. Tasks in two different stages transmit data by means of the exchange module. `Task` is also a logical concept that contains the parameters and contents of the task. The actual task execution is done by the driver.

- Driver

  Driver is responsible for executing the specific tasks. A task may contain multiple driver instances so as to achieve parallel processing within the same task. Each driver processes a split. A driver is made up of a set of operators and is responsible for specific data operations, such as conversion and filtering.

- Operator

  The operator is the smallest execution unit and is responsible for processing each page of a split, such as weighting and conversion. It is similar to a logical operator in concept. A page is a column-based data structure, and is the smallest data unit that an operator can process. A page object consists of multiple blocks , with each block representing multiple data rows of a field. Pages can be of a maximum of 1 MB and can contain data of up to 16 x 1024 rows.
- Exchange

  Two stages exchange data through the exchange module. The data transmission process is completed between two tasks. A downstream task typically fetches data from the output buffer of an upstream task using an exchange client. The fetched data is then transmitted to driver in splits for processing.

Command line tool

The command line tool uses *SSH to log on to an EMR cluster* and executes the following command to enter the Presto console:

```
$ presto --server emr-header-1:9090 --catalog hive --schema default --
user hadoop
```

High-security clusters use the following command:

```
$ presto  --server https://emr-header-1:7778  \
          --enable-authentication \
          --krb5-config-path /etc/krb5.conf \
          --krb5-keytab-path  /etc/ecm/presto-conf/presto.keytab \
          --krb5-remote-service-name presto \
          --keystore-path /etc/ecm/presto-conf/keystore \
          --keystore-password 81ba14ce6084 \
          --catalog hive --schema default \
          --krb5-principal  presto/emr-header-1.cluster-XXXX@EMR.XXXX.
COM
```

- *`XXXX`* is the ECM ID of the cluster, a string of numbers that can be obtained through *`cat /etc/hosts`*.
- *`81ba14ce6084`* is the default password of *`/etc/ecm/presto-conf/keystore`*. We recommend that you use your own keystore after deployment.

You can execute the following command from the console:

```
Presto: Default> show schemas;
    schema.
--------------------
default
Hive
```

```
information_schema
tpch_100gb_orc
tpch_10gb_orc
tpch_10tb_orc
tpch_1tb_orc
(7 rows)
```

You can then execute the `presto --help` command to obtain help from the console. The parameters and definitions are as follows:

```
--server <server>                        # Specifies the URI of a
Coordinator
--user <user>                            # Sets the username
--catalog <catalog>                      # Specifies the default
Catalog
--schema <schema>                        # Specifies the default Schema
--execute <execute>                      # Executes a statement and
then exits
-f <file>, --file <file>                 # Executes an SQL statement
and then exits
--debug                                  # Shows debugging information
--client-request-timeout <timeout>       # Specifies the client timeout
 value, which is 2 minutes by default
--enable-authentication                  # Enables client authentica
tion
--keystore-password <keystore password> # KeyStore password
--keystore-path <keystore path>          # KeyStore path
--krb5-config-path <krb5 config path>    # Kerberos configuration file
path (default: /etc/krb5.conf)
--krb5-credential-cache-path <path>      # Kerberos credential cache
path
--krb5-keytab-path <krb5 keytab path>    # Kerberos Key table path
--krb5-principal <krb5 principal>        # Kerberos principal to be
used
--krb5-remote-service-name <name>        # Remote Kerberos node name
--log-levels-file <log levels>           # Configuration file path for
debugging logs
--output-format <output-format>          # Bulk export data format,
which is CSV by default
--session <session>                      # Specifies the session
attribute, in the format key=value
--socks-proxy <socks-proxy>              # Sets the proxy server
--source <source>                        # Sets query source
--version                                # Shows version info
-h, --help                               # Shows help info
```

Uses JDBC

Java applications can access databases using the JDBC driver provided by Presto. The procedure is the same as that for general RDBMS databases.

- **Introduction to Maven**

  You can add the following configuration to the POM file to introduce the Presto JDBC driver:

```
<dependency>
    <groupId>com.facebook.presto</groupId>
    <artifactId>presto-jdbc</artifactId>
    <version>0.187</version>
</dependency>
```

- Driver class name

  The Presto JDBC driver class is `com.facebook.presto.jdbc.PrestoDriver`.

- Connection string

  The following connection string format is supported.

  ```
  jdbc:presto://<COORDINATOR>:<PORT>/[CATALOG]/[SCHEMA]
  ```

  For example:

  ```
  jdbc:presto://emr-header-1:9090               # Connects to data
  base, using the default Catalog and Schema
  jdbc:presto://emr-header-1:9090/hive          # Connects to data
  base, using Catalog(hive) and the default Schema
  jdbc:presto://emr-header-1:9090/hive/default  # Connects to data
  base, using Catalog(hive) and Schema(default)
  ```

- Connection parameters

  The Presto JDBC driver supports various parameters that may be set as URL parameters or as `Properties` and passed to DriverManager.

  Example of passing parameters to DriverManager as `Properties`:

  ```
  String url = "jdbc:presto://emr-header-1:9090/hive/default";
  Properties properties = new Properties();
  properties.setProperty("user", "hadoop");
  Connection connection = DriverManager.getConnection(url, properties
  );
  ......
  ```

  Example of passing parameters to DriverManager as URL parameters:

  ```
  String url = "jdbc:presto://emr-header-1:9090/hive/default? user=
  hadoop";
  Connection connection = DriverManager.getConnection(url);
  ......
  ```

  The parameters are described as follows:

Parameter name	Format	Description
user	STRING	User name.
password	STRING	Password.
Socksproxy	\:\	SOCKS proxy server address and port. For example, localhost:1080.

Parameter name	Format	Description
httpProxy	\:\	HTTP proxy server address and port. For example, localhost:8888.
SSL	true\	Whether or not to use HTTPS for connections. This is false by default.
SSLTrustStorePath	STRING	Java TrustStore file path.
SSLTrustStorePassword	STRING	Java TrustStore password.
KerberosRemoteServiceName	STRING	Kerberos service name.
KerberosPrincipal	STRING	Kerberos principal.
KerberosUseCanonicalHostname	true\	Whether or not to use the canonical hostname. This is false by default.
KerberosConfigPath	STRING	Kerberos configuration file path.
KerberosKeytabPath	STRING	Kerberos KeyTab file path.
KerberosCredentialCachePath	STRING	Kerberos credential cache path

- Java example

  The following is an example of using the Presto JDBC driver with Java.

```
.....
// Loads the JDBC Driver class
try {
    Class.forName("com.facebook.presto.jdbc.PrestoDriver");
} catch(ClassNotFoundException e) {
    LOG.ERROR("Failed to load presto jdbc driver.", e);
    System.exit(-1);
}
Connection connection = null;
Statement stmt = null;
try {
    String url = "jdbc:presto://emr-header-1:9090/hive/default";
    Properties properties = new Properties();
    properties.setProperty("user", "hadoop");
    // Creates the connection object
    Connection = drivermanager. getconnection (URL, properties );
    // Creates the Statement object
    statement = connection.createStatement();
    Executes the query
    ResultSet rs = statement.executeQuery("select * from t1");
    Returns results
    int columnNum = rs.getMetaData().getColumnCount();
    int rowIndex = 0;
    while (rs.next()) {
        rowIndex++;
        for(int i = 1; i <= columnNum; i++) {
            System.out.println("Row " + rowIndex + ", Column " + i +
 ": " + rs.getInt(i));
```

```
        }
    }
} catch(SQLException e) {
    LOG.ERROR("Exception thrown.", e);
} finally {
  // Destroys Statement object
  If (statement! = null) {
      try {
        statement.close();
    } catch(Throwable t) {
        // No-ops
    }
  }
  Closes connection
  if (connection ! = null) {
      try {
        connection.close();
    } catch(Throwable t) {
        // No-ops
    }
  }
}
```

9.8.4 Data types

Presto supports multiple common data types by default, such as Boolean, Integer, Floating-Point, String, and Date and Time. You can also add customized data types using plug-ins. Customized Presto connectors are not required to support all data types.

Data types

Presto's set of built-in data types are as follows:

- Boolean

  Represents two option with a value of true or false.
- Tinyint

  A two's complement 8-bit signed integer.
- Smallint

  A two's complement 16-bit signed integer.
- Integer

  A two's complement 32-bit signed integer.
- Bigint

  A two's complement 64-bit signed integer.
- Real

A 32-bit inexact, variable-precision floating point which implements *IEEE Standard 754* for Floating-Point Arithmetic.

- Double

  A 64-bit multi-precision floating point which implements *IEEE Standard 754* for Floating-Point Arithmetic.

- Decimal

  A fixed precision decimal number. Precision up to 38 digits is supported but performance is best up to 17 digits. It takes two literal parameters to define the DECIMAL type :A fixed-precision decimal number. Precision up to 38 digits is supported but performance is optimal up to 18 digits. The following two literal parameters are required to define decimal:

  - precision: The total number of digits, excluding symbols.
  - scale: The number of digits in a fractional part. Scale is optional and defaults to 0.

    For example, `DECIMAL '-10.7'` can be expressed with the `DECIMAL(3,1)` type.

  The following table describes the bits and value range of the integer type.

Value type	Bits	Minimum value	Maximum value
TINYINT	8 bit	-2^7	2^7 - 1
SMALLINT	16 bit	2^15	2^15 - 1
INTEGER	32 bit	-2^31	-2^31 - 1
BIGINT	64 bit	-2^63	-2^63 - 1

String type

Presto supports the following built-in string types:

- Varchar

  Variable length character data with an optional maximum length.

  For example, `VARCHAR` or `VARCHAR(10)`.

- Char

  Fixed length character data. If no length is specified, the default length is 1.

  For example, `CHAR` or `CHAR(10)` .

> Note:
> A string with a specified length always has the number of characters equal to this length. Where the string length is smaller than the specified length, leading and trailing spaces are included in comparisons of the string value. As a result, two character values of different lengths can never be equal.

- Varbinary

  Indicates variable length binary data.

Date and time

Presto supports the following built-in date and time types:

- Date

  Refers to a calendar date (year, month, day) without a time.

  For example, `DATE '1988-01-30'`.

- Time

  Refers to a time, including the hour, minute, second, and millisecond. This value can be modified in the time zone.

  For example:

  - `TIME '18:01:02.345'` does not have a time zone definition and is therefore parsed using the system time zone.
  - `TIME '18:01:02.345 Asia/Shanghai'` has a time zone definition and is therefore parsed using the defined time zone.

- Timestamp

  Refers to a point in time that includes the date and time of day. The value range is from `'1970-01-01 00:00:01' UTC` to `'2038-01-19 03:14:07' UTC`, which can be modified in the time zone.

  For example, `TIMESTAMP '1988-01-30 01:02:03.321'` or `TIMESTAMP '1988-01-30 01:02:03.321 Asia/Shanghai'`.

- Interval

  Mainly used in time calculated expressions to refer to a time span, the unit of which can be:

- Year
- Quarter (of the year)
- Month
- Day
- Hour
- Minute
- Second
- Millisecond

For example, `DATE '2012-08-08' + INTERVAL '2' DAY`.

Complex types

Presto supports multiple complex built-in data types to support more complex business scenarios. These data types include the following:

- JSON

  This can be a JSON object, array, number, or string, as well as the Boolean type in the TRUE, FALSE or NULL state.

  For example:

  - `JSON '[1, null, 1988]'`
  - `JSON '{"K1": 1, "K2": "ABC "}'`
- Array

  An array of the given component type. The types of elements in an array must be consistent.

  For example, `ARRAY[1, 2, 3]`.
- Map

  Represents a mapping relationship that consists of a key array and a value array.

  For example, `MAP(ARRAY['foo', 'bar'], ARRAY[1, 2])`
- Row

  A structure made up of named fields. The fields may be accessed with field reference operator `.` and the field names. You can use the operator and the method of column names to access data columns.

For example, `CAST(ROW(1988, 1.0, 30) AS ROW(y BIGINT, m DOUBLE, d TINYINT ))`.

- IP address

  An IP address that can represent either an IPv4 or IPv6 address. Internally, the type is a IPv6 address. Support for IPv4 is handled using the *IPv4-mapped IPv6 address range*.

  For example, `IPADDRESS '0.0.0.0'` or `IPADDRESS '2001:db8::1'`.

9.8.5 Common functions and operators

This section describes common Presto functions and operators.

Logical operators

Presto supports AND, OR, and NOT logical operators, and supports NULL in logical computation. For example:

```
SELECT CAST(null as boolean) AND true; --- null
SELECT CAST(null AS boolean) AND false; -- false
SELECT CAST(null AS boolean) AND CAST(null AS boolean); -- null
SELECT NOT CAST(null AS boolean); -- null
```

A complete truth table is shown as follows:

a	b	a AND b	A or B
TRUE	TRUE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	NULL	NULL	TRUE
FALSE	TRUE	FALSE	TRUE
FALSE	FALSE	FALSE	FALSE
FALSE	NULL	FALSE	NULL
NULL	TRUE	NULL	TRUE
NULL	FALSE	FALSE	NULL
NULL	FALSE	NULL	NULL

The result of NOT FALSE is TRUE, the result of NOT TRUE is FALSE, and the result of NOT NULL is NULL.

This section details the comparison functions and operators.

- Comparison operators

Presto provides the following comparison operations:

Operator	Description
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
=	Equal to
<>/! =	Not equal to
[NOT] BETWEEN	Value X is [not] between the minimum and the maximum values.
IS [NOT] NULL	Tests whether a value is NULL.
IS [NOT] DISTINCT FROM	Determines whether two values are identical. NULL typically signifies an unknown value, which means that any comparison involving a NULL will produce NULL. However, the IS [NOT] DISTINCT FROM operator treats NULL as a known value, and consequently returns a TRUE or FALSE result.

- Comparison functions

  Presto provides the following comparison related functions:

  - GREATEST

    Returns the largest of the provided values.

    For example, `GREATEST(1, 2)`.
  - LEAST

    Returns the smallest of the provided values.

    For example, `LEAST(1, 2)`.
- Quantified comparison predicates

  Presto also provides several quantified comparison predicates to enhance the comparison expressions. These are used as follows:

```
<EXPRESSION> <OPERATOR> <QUANTIFIER> (<SUBQUERY>)
```

For example:

```
SELECT 21 < ALL (VALUES 19, 20, 21); -- false
SELECT 42 >= SOME (SELECT 41 UNION ALL SELECT 42 UNION ALL SELECT 43
); -- true
```

ALL, ANY and SOME are quantified comparison predicates.

- `A = ALL (…)`: If A is equal to all values, TRUE is returned. For example, if `SELECT 21 = ALL (VALUES 20, 20, 20);`, TRUE is returned.
- `A <> ALL (…)`: If A does not match any values, TRUE is returned. For example, if `SELECT 21 <> ALL (VALUES 19, 20, 22);`, TRUE is returned.
- `A < ALL (…)`: If A is smaller than the smallest value, TRUE is returned. For example, if `SELECT 18 < ALL (VALUES 19, 20, 22);`, TRUE is returned.
- `A = ANY (…)`: If A is equal to any of the values, TRUE is returned. This form is equivalent to `A IN (…)`. For example, if `SELECT ‘hello’ = ANY (VALUES ‘hello’, ‘world’);`, TRUE is returned.
- `A <> ANY (…)`: If A does not match one or more values, TRUE is returned. This form is equivalent to `A IN (…)`. For example, if `SELECT 21 &lt;&gt; ALL (VALUES 19, 20, 21);`, TRUE is returned.
- `A < ANY (…)`: If A is smaller than the biggest value, true is returned. For example, if `SELECT 21 < ALL (VALUES 19, 20, 22);`, TRUE is returned.

ANY and SOME have the same meaning and can be used interchangeably.

Conditional expressions

Conditional expressions are mainly used to express branch logic. Presto supports the following conditional expressions:

- CASE expression

  The standard SQL CASE expression has two different forms:

```
CASE expression
    WHEN <value|condition> THEN result
    [ WHEN ... ]
    [ ELSE result]
END
```

The *expression* statement compares the expression and the value and condition in *value|condition*. If the same value is found or the condition is met, a result is returned.

For example:

```
--- Compare value
```

```
SELECT a,
       CASE a
           WHEN 1 THEN 'one'
           WHEN 2 THEN 'two'
           ELSE 'many'
       END
```

```
--- Compare conditional expression
SELECT a, b,
       CASE
           WHEN a = 1 THEN 'aaa'
           WHEN b = 2 THEN 'bbb'
           ELSE 'ccc'
       END
```

- IF function

  The IF function is a simple comparison function used to simplify the writing method for the comparison logic of two values. Its expression form is as follows:

  ```
  IF(condition, true_value, [false_value])
  ```

  If *`condition`* is true, true_value is returned. Otherwise, false_value is returned. false_value is optional. If it is not specified and if *`condition`* is not true, NULL is returned.

- COALESCE

  The COALESCE function returns the first non-null value in the argument list. Its expression form is as follows:

  ```
  COALESCE(value1, value2[, ...])
  ```

- NULLIF

  The NULLIF function returns null if value1 equals value2. Otherwise, value1 is returned. Its expression form is as follows:

  ```
  NULLIF(value1, value2)
  ```

- TRY

  The TRY function evaluates an expression and handles certain types of errors by returning NULL. The following errors are handled by TRY:

  - Division by zero
  - Invalid cast or function argument
  - Numeric value out of range

In the event of errors, it is typically used in conjunction with COALESCE to return the default value. Its expression form is as follows:

```
TRY(expression)
```

For example:

```
--- When COALESCE and TRY are used in conjunction, if packages=0,
and a "division by zero" error is thrown, the default value (0) will
 be returned.
SELECT COALESCE(TRY(total_cost / packages), 0) AS per_package FROM
shipping;
per_package
-------------
           4
          14
           0
          19
(4 rows)
```

Conversion functions

Presto provides the following explicit conversion functions:

- CAST

  Explicitly casts a value as a type, and raises an error if the cast fails. Use it is as follows:

  ```
  CAST(value AS type) -> value1:type
  ```

- TRY_CAST

  Similar to cast, but returns null if the cast fails. Use it as follows:

  ```
  TRY_CAST(value AS TYPE) -> value1:TYPE | NULL
  ```

- TYPEOF

  Returns the type of the provided parameter or expression value. Use it as follows:

  ```
  TYPEOF(expression) -> type:VARCHAR
  ```

  For example:

  ```
  SELECT TYPEOF(123); -- integer
  SELECT TYPEOF('cat'); -- varchar(3)
  SELECT TYPEOF(cos(2) + 1.5); -- double
  ```

Mathematical functions and operators

- Mathematical operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division (dividing integers results in truncation)
%	Modulus (remainder)

- Mathematical functions

  Presto provides a wealth of mathematical functions, as shown in the following table:

Function	Syntax	Description
abs	abs(x) →	Returns the absolute value of x.
cbrt	cbrt(x) → double	Returns the cube root of x.
ceil	ceil(x)	Returns x rounded up to the nearest integer. This is an alias for ceiling.
ceiling	ceiling(x)	Returns x rounded up to the nearest integer.
cosine_similarity	cosine_similarity(x, y) → double	Returns the cosine similarity between the sparse vectors x and y.
degrees	degress(x) -> double	Converts angle x in radians to degrees.
e	e()->double	Returns the constant Euler's number.
exp	exp(x)->double	Returns Euler's number raised to the power of x.
floor	floor(x)	Returns x rounded down to the nearest integer.
from_base	from_base(string, radix) → bigint	Returns the value of string interpreted as a base-radix number.
inverse_normal_cdf	inverse_normal_cdf(mean,sd,p)->double	Computes the inverse of the Normal cdf with a given mean and standard deviation (sd) for the cumulative probability.
ln	ln(x)->double	Returns the natural logarithm of x.
log2	log2(x)->double	Returns the base 2 logarithm of x.

Function	Syntax	Description
log10	log10(x)->double	Returns the base 10 logarithm of x.
log	log(x,b) -> double	Returns the base b logarithm of x.
mod	mod(n,m)	Returns the modulus (remainder) of n divided by m.
pi	pi()->double	Returns the constant pi.
pow	pow(x,p)->double	Returns x raised to the power of p. This is an alias for power.
power	power(x,p)->double	Returns x raised to the power of p.
radians	radians(x)->double	Converts angle x in degrees to radians.
rand	rand()->double	Returns a pseudo-random value in the range 0.0 <= x < 1.0. This is an alias for random.
random	random()->double	Returns a pseudo-random value in the range 0.0 <= x < 1.0.
random	random(n)	Returns a pseudo-random number between 0 and n (exclusive).
round	round(x)	Returns x rounded to the nearest integer.
round	round(x, d)	Returns x rounded to d decimal places.
sign	sign(x)	Returns the signum function of x. In other words, if the argument is 0, 0 is returned. If the argument is greater than 0, 1 is returned. If the argument is less than 0, -1 is returned. For double arguments, the function also returns NaN if the argument is NaN, 1 if the argument is +Infinity, and -1 if the argument is -Infinity.
sqrt	sqrt(x)->double	Returns the square root of x.
to_base	to_base(x, radix)-> varchar	Returns the radix representation of x.
truncate	truncate(x) → double	Returns x rounded to an integer by dropping digits after the decimal point.

Function	Syntax	Description
width_bucket	width_bucket(x, bound1, bound2, n) → bigint	Returns the bin number of x in an equi-width histogram with the specified bound1 and bound2 bounds and n number of buckets.
width_bucket	width_bucket(x, bins)	Returns the bin number of x according to the bins specified by the array bins.
acos	acos(x)->double	Returns the arc cosine of x, which is a radian.
asin	asin(x)->double	Returns the arc sine of x, which is a radian.
atan	atan(x)->double	Returns the arc tangent of x, which is a radian.
atan2	atan2(y,x)->double	Returns the arc tangent of y / x, which is a radian.
cos	cos(x)->double	Returns the cosine of x, which is a radian.
cosh	cosh(x)->double	Returns the hyperbolic cosine of x, which is a radian.
sin	sin(x)->double	Returns the sine of x, which is a radian.
tan	tan(x)->double	Returns the tangent of x, which is a radian.
tanh	tanh(x)->double	Returns the hyperbolic tangent of x, which is a radian.
infinity	infinity() → double	Returns the constant representing positive infinity.
is_finite	is_finite(x) → boolean	Determines if x is finite.
is_infinite	is_infinite(x) → boolean	Determines if x is infinite.
is_nan	is_nan(x) → boolean	Determines if x is not-a-number.
nan	nan()	Returns the constant representing not-a-number.

Bitwise functions

Presto provides the following bitwise functions:

Function	Syntax	Description
bit_count	bit_count(x, bits) → bigint	Returns the number of bits set in x at position 1 in two's complement representation.
bitwise_and	bitwise_and(x, y) → bigint	The bitwise AND function
bitwise_not	bitwise_not(x) → bigint	The bitwise NOT function
bitwise_or	bitwise_or(x, y) → bigint	The bitwise OR function
bitwise_xor	bitwise_xor(x, y) → bigint	The bitwise XOR function
bitwise_and_agg	bitwise_and_agg(x) → bigint	Returns the bitwise AND of all input values in two's complement representation. x is an array.
bitwise_or_agg	bitwise_or_agg(x) → bigint	Returns the bitwise OR of all input values in two's complement representation. x is an array.

For example:

```
SELECT bit_count(9, 64); -- 2
SELECT bit_count(9, 8); -- 2
SELECT bit_count(-7, 64); -- 62
SELECT bit_count(-7, 8); -- 6
```

Decimal functions and operators

- Decimal literals

  Use the following syntax to define the literal of the DECIMAL type:

  ```
  DECIMAL 'xxxx.yyyyy'
  ```

  The *precision* of the DECIMAL type is equal to the number of digits in the literal (including trailing and leading zeros). The *scale* is equal to the number of digits in the fractional part (including trailing zeros). For example:

Example literal	Data type
DECIMAL ‘0’	DECIMAL(1)
DECIMAL ‘12345’	DECIMAL(5)
DECIMAL ‘0000012345.1234500000’	DECIMAL(20, 10)

- Operators

  Arithmetic operators

Assuming that x is of the DECIMAL(xp, xs) type and y is of the DECIMAL(yp, ys) type,

- x: DECIMAL(xp,xs)
- y: DECIMAL(yp,ps)

and they observe the following rules when used in arithmetic operations:

- x + y or x - y
  - precision = min(38, 1 + min(xs, ys) + min(xp-xs, yp-ys))
  - scale = max(xs, ys)
- x * y
  - precision = min(38, xp + yp)
  - scale = xs + ys
- x / y
  - precision = min(38, xp + ys + max(0, ys-xs))
  - scale = max(xs, ys)
- x % y
  - precision = min(xp - xs, yp - ys) + max(xs, bs)
  - scale = max(xs, ys)
- Comparison operators

  All standard comparison operators and BETWEEN operator can be used for the DECIMAL type.
- Unary decimal operators

  The - operator performs negation for the DECIMAL type.

String functions and operators

- Concatenation operator

  The `||` operator performs concatenation.
- String functions

  The string functions supported by Presto are listed in the following table:

Function name	Syntax	Description
chr	chr(n) → varchar	Returns the Unicode code point `n` as a single character string.
codepoint	codepoint(string) → integer	Returns the Unicode code point of the only character of `string`.
concat	concat(string1, ⋯, stringN) → varchar	Returns the concatenation of string1 , string2, ⋯, stringN. This function provides the same functionality as the SQL-standard concatenation operator.
hamming_distance	hamming_distance(string1, string2) → bigint	Returns the *Hamming distance* of string1 and string2, in other words the number of positions at which the corresponding characters are different. Note that the two strings must have the same length.
levenshtein_distance	levenshtein_distance(string1, string2) → bigint	Returns the *Levenshtein distance* of string1 and string2.
lower	lower(string) → varchar	Converts string to lowercase.
upper	upper(string) → varchar	Converts string to uppercase.
replace	replace(string, search) → varchar	Removes all instances of `search` from `string`.
replace	replace(string, search, replace) → varchar	Replaces all instances of `search` with `replace` in `string`.
reverse	reverse(string) → varchar	Returns string with the characters in reverse order.
lpad	lpad(string, size, padstring) → varchar	Left pads string to size characters with `padstring`. If size is less than the length of `string`, the result is truncated to `size` characters. `size` must not be negative and `padstring` cannot be empty.

Function name	Syntax	Description
rpad	rpad(string, size, padstring) → varchar	Right pads string to size characters with `padstring`. If size is less than the length of `string`, the result is truncated to `size` characters. `size` must not be negative and `padstring` cannot be empty.
ltrim	ltrim(string) → varchar	Removes leading whitespace from string.
rtrim	rtrim(string) → varchar	Removes trailing whitespace from string.
split	split(string, delimiter) → array	Splits string on delimiter and returns an array.
split	split(string, delimiter, limit) → array	Splits string on delimiter and returns an array of size at the maximum limit.
split_part	split_part(string, delimiter, index) → varchar	Splits string on delimiter and returns the field `index`. Field indexes start with 1.
split_to_map	split_to_map(string , entryDelimiter, keyValueDelimiter) → map	Splits string by entryDelimiter and keyValueDelimiter and returns a map.
strpos	strpos(string, substring) → bigint	Returns the starting position of the first instance of substring in string. Positions start with 1. If it cannot be found, 0 is returned.
position	position(substring IN string) → bigint	Returns the starting position of the first instance of substring in string.
substr	substr(string, start, [length]) → varchar	Returns a substring from string of [length] length from the starting position `start`. Positions start with 1. The length parameter is optional.

- Unicode functions
  - normalize(string) → varchar

    Transforms string with the *NFC normalization form*.
  - normalize(string, form) → varchar

Transforms string with the specified normalization form. `form` must be one of the following keywords:

- `NFD` canonical decomposition
- `NFC` canonical decomposition, followed by canonical composition
- `NFKD` compatibility decomposition
- `NFKC` compatibility decomposition, followed by canonical composition

- to_utf8(string) → varbinary

  Encodes string into a UTF-8 varbinary representation.
- from_utf8(binary, [replace]) → varchar

  Decodes a UTF-8 encoded string from binary. Invalid UTF-8 sequences are replaced with `replace`, which is the Unicode replacement character `U+FFFD` by default. Note that the replacement string `replace` must either be a single character or empty.

Regular expression functions

Presto supports all of the regular expression functions that use the *Java Pattern* syntax, with a few notable exceptions:

- When in multi-line mode:
  - Enabled through the `? m` flag.
  - `\n` is recognized as a line terminator.
  - The `? d` flag is not supported.
- Case-sensitive matching:
  - Enabled through the `?i` flag.
  - The `?u` flag is not supported.
  - Context-sensitive matching is not supported.
  - Local-sensitive matching is not supported.
- Surrogate pairs are not supported

  For example, `\uD800\uDC00` is not treated as `U+10000` and must be specified as `\x{10000}`.
- Boundaries `\b` are incorrectly handled for a non-spacing mark without a base character.

- `\Q` and `\E` are not supported in character classes (such as `[A-Z123]`) and are instead treated as literals.
- Unicode character classes (`\p{prop}`) are supported with the following differences:
  - All underscores in names must be removed. For example, use `OldItalic` instead of `Old_Italic`.
  - Scripts must be specified directly, without the `Is`, `script=`, or `sc=` prefixes. For example, use `\p{Hiragana}` instead of `\p{script=Hiragana}`.
  - Blocks must be specified with the `In` prefix. The `block=` and `blk=` prefixes are not supported. For example, `\p{InMongolia}`.
  - Categories must be specified directly, without the `Is`, `general_category=` or `gc=` prefixes. For example, `\p{L}`.
  - Binary properties must be specified directly, without the `Is` prefix. For example, use `\p{NoncharacterCodePoint}` instead of `\p{IsNoncharacterCodePoint}`.

The regular expression functions provided by Presto are as follows:

- regexp_extract_all(string, pattern, [group]) → array

  Returns the substring(s) matched by the regular expression `pattern` in `string`. If the `pattern` expression uses the grouping function, then the `group` parameter can be set to specify the *capturing group*.

  For example:

  ```
  SELECT regexp_extract_all('1a 2b 14m', '\d+'); -- [1, 2, 14]
  SELECT regexp_extract_all('1a 2b 14m', '(\d+)([a-z]+)', 2); -- ['a',
   'b', 'm']
  ```
- regexp_extract(string, pattern, [group]) → varchar

  The function and usage is similar to those of `regexp_extract_all`. The difference is that this function only returns the first substring matched by the regular expression.

  For example:

  ```
  SELECT regexp_extract_all('1a 2b 14m', '\d+'); -- [1, 2, 14]
  SELECT regexp_extract_all('1a 2b 14m', '(\d+)([a-z]+)', 2); -- ['a',
   'b', 'm']
  ```
- regexp_extract_all(string, pattern, [group]) → array

Returns the substring(s) matched by the regular expression `pattern` in `string`. If the `pattern` expression uses the grouping function, then the `group` parameter can be set to specify the *capturing group* to be matched by the regular expression.

For example:

```
SELECT regexp_extract('1a 2b 14m', '\d+'); -- 1
SELECT regexp_extract('1a 2b 14m', '(\d+)([a-z]+)', 2); -- 'a'
```

- regexp_like(string, pattern) → boolean

  Evaluates the regular expression `pattern` and determines if it is contained within `string`. If it is, TRUE is returned. If not, False is returned. This function is similar to the LIKE operator, expect that the pattern only needs to be contained within string, rather than needing to match all of string.

  For example:

  ```
  SELECT regexp_like('1a 2b 14m', '\d+b'); -- true
  ```

- regexp_replace(string, pattern, [replacement]) → varchar

  Replaces every instance of the substring matched by the regular expression `pattern` in `string` with `replacement`. `replacement` is optional and is replaced by ‘’ (deleting the matched substrings) if it is not specified.

  Capturing groups can be referenced in `replacement` using *`$g`* (g is the ordinal number, starting at one) for a numbered group or *`${name}`* for a named group. A dollar sign ($) may be included in the `replacement` by escaping it with a backslash `\$`.

  For example:

  ```
  SELECT regexp_replace('1a 2b 14m', '\d+[ab] '); -- '14m'
  SELECT regexp_replace('1a 2b 14m', '(\d+)([ab]) ', '3c$2 '); -- '3ca
   3cb 14m'
  ```

- regexp_split(string, pattern) → array

  Splits string using the regular expression `pattern` and returns an array. Trailing empty strings are preserved.

  For example:

  ```
  SELECT regexp_split('1a 2b 14m', '\s*[a-z]+\s*'); -- ['1', '2', '14
  ', ''] 4 elements
  ```

```
                                                    -- The last one is
 an empty string
```

Binary functions and operators

- Concatenation operator

  The `||` operator performs binary concatenation.
- Binary functions

Function	Syntax	Description
length	length(binary) → bigint	Returns the length of binary in bytes.
concat	concat(binary1, ⋯, binaryN) → varbinary	Returns the concatenation of binary1, binary2, ⋯, binaryN.
to_base64	to_base64(binary) → varchar	Encodes binary into a base64 string representation.
from_base64	from_base64(string) → varbinary	Decodes binary data from the base64 encoded string.
to_base64url	to_base64url(binary) → varchar	Encodes binary into a base64 string representation using the URL safe alphabet.
from_base64url	from_base64url(string) → varbinary	Decodes binary data from the base64 encoded string using the URL safe alphabet.
to_hex	to_hex(binary) → varchar	Encodes binary into a hex string representation.
from_hex	from_hex(string) → varbinary	Decodes binary data from the hex encoded string.
to_big_endian_64	to_big_endian_64(bigint) → varbinary	Encodes bigint in a 64-bit two's complement big endian format.
from_big_endian_64	from_big_endian_64(binary) → bigint	Decodes bigint value from a 64-bit two's complement big endian binary.
to_ieee754_32	to_ieee754_32(real) → varbinary	Encodes real in a 32-bit big-endian binary according to *IEEE 754* single-precision floating-point format.
to_ieee754_64	to_ieee754_64(double) → varbinary	Encodes double in a 64-bit big-endian binary according to *IEEE 754* double-precision floating-point format.
crc32	crc32(binary) → bigint	Computes the CRC-32 of binary.

Function	Syntax	Description
md5	md5(binary) → varbinary	Computes the md5 hash of binary.
sha1	sha1(binary) → varbinary	Computes the sha1 hash of binary.
sha256	sha256(binary) → varbinary	Computes the sha256 hash of binary.
sha512	sha512(binary) → varbinary	Computes the sha512 hash of binary.
xxhash64	xxhash64(binary) → varbinary	Computes the xxhash64 hash of binary.

Date and time functions and operators

- Date and time operators

  Presto supports two date and time operators: `+` and `-`.

  For example:

```
--- +
 date '2012-08-08' + interval '2' day                --- 2012-08-10
 time '01:00' + interval '3' hour                    --- 04:00:00.
000
 timestamp '2012-08-08 01:00' + interval '29' hour    --- 2012-08-09
 06:00:00.000
 timestamp '2012-10-31 01:00' + interval '1' month    --- 2012-11-30
 01:00:00.000
 interval '2' day + interval '3' hour                --- 2 03:00:00.
000
 interval '3' year + interval '5' month                --- 3-5
 --- -
 date '2012-08-08' - interval '2' day                --- 2012-08-06
 time '01:00' - interval '3' hour                    --- 22:00:00.
000
 timestamp '2012-08-08 01:00' - interval '29' hour    --- 2012-08-06
 20:00:00.000
 timestamp '2012-10-31 01:00' - interval '1' month    --- 2012-09-30
 01:00:00.000
 interval '2' day - interval '3' hour                --- 1 21:00:00.
000
 interval '3' year - interval '5'                   --- month    2-7
```

- Time zone conversion

  The `AT TIME ZONE` operator sets the time zone of a timestamp.

  For example:

```
SELECT timestamp '2012-10-31 01:00 UTC';
 --- 2012-10-31 01:00:00.000 UTC
 SELECT timestamp '2012-10-31 01:00 UTC' AT TIME ZONE 'America/
Los_Angeles';
```

```
--- 2012-10-30 18:00:00.000 America/Los_Angeles
```

- Date and time functions

  - Basic functions

Function	Syntax	Description
current_date	current_date -> date	Returns the current date as of the start of the query.
current_time	current_time -> time with time zone	Returns the current time as of the start of the query.
current_timestamp	current_timestamp -> timestamp with time zone	Returns the current timestamp as of the start of the query.
current_timezone	current_timezone() → varchar	Returns the current time zone.
date	date(x) → date	Parses a date literal into a date.
from_iso8601_timestamp	from_iso8601_timestamp(string) → timestamp with time zone	Parses the ISO 8601 formatted string into a timestamp with time zone.
from_iso8601_date	from_iso8601_date(string) → date	Parses the ISO 8601 formatted string into a date.
from_unixtime	from_unixtime(unixtime, [timezone_str]) → timestamp	Returns the UNIX timestamp as a timestamp. The timestamp option is allowed.
from_unixtime	from_unixtime(unixtime, hours, minutes) → timestamp with time zone	Returns the UNIX timestamp as a timestamp with a time zone using `hours` and `minutes` for the time zone offset.
localtime	localtime -> time	Returns the current time as of the start of the query.
localtimestamp	localtimestamp -> timestamp	Returns the current timestamp as of the start of the query.
now	now() → timestamp with time zone	Returns the current time. This is an alias for `current_time`.
to_iso8601	to_iso8601(x) → varchar	Formats `x` as an ISO 8601 string. `x` can either be `DATE` or `TIMESTAMP [with time zone]`.

Function	Syntax	Description
to_milliseconds	to_milliseconds(interval) → bigint	Returns the day-to-second interval as milliseconds.
to_unixtime	to_unixtime(timestamp) → double	Returns the timestamp as a UNIX timestamp.

Note:

The following SQL-standard functions do not use parentheses:

- current_data
- current_time
- current_timestamp
- localtime
- localtimestamp

- Truncation function

  The truncation function truncates the date and time value by a specified unit, and returns the date and time value of this unit. Use it as follows:

  ```
  date_trunc(unit, x) -> [same as x]
  ```

  where `unit` is one of the following:

  - `second`
  - `minute`
  - `hour`
  - `day`
  - `week`
  - `month`
  - `quarter`
  - `year`

- Interval functions

  Presto provides the following two functions for interval calculation:

  - date_add(unit, value, timestamp) → [same as input]

    Adds an interval value of unit to timestamp. Subtraction can be performed by using a negative value with a unit.

  - date_diff(unit, timestamp1, timestamp2) → bigint

    Returns interval between two timestamps expressed in terms of unit, where unit is one of the following:

    - `ns`: nanoseconds
    - `us`: microseconds
    - `ms`: milliseconds
    - `s`: seconds
    - `m`: minutes
    - `h`: hours
    - `d`: days

- Date and time extraction functions

  Presto provides an `extract` function to extract the specified fields from a date and time value. The function is as follows:

  extract(field FROM x) → bigint

  where `x` is the date and time value and `field` is the field to be extracted, which can be one of the following values:

  - `YEAR`
  - `QUARTER`: Quarter of a year.
  - `MONTH`
  - `WEEK`
  - `DAY`
  - `DAY_OF_MONTH`
  - `DAY_OF_WEEK`
  - `DOW`: This is an alias for DAY_OF_WEEK.
  - `DAY_OF_YEAR`
  - `DOY`: This is an alias for `DAY_OF_YEAR`.
  - `YEAR_OF_WEEK`: Year of an *ISO Week*
  - `YOW`: This is an alias for `YEAR_OF_WEEK`.
  - `HOUR`
  - `MINUTE`

- `SECOND`
- `TIMEZONE_HOUR`: Hour, with a time zone
- `TIMEZONE_MINUTE`: Minute, with a time zone

For the sake of convenience, Presto provides the following helper functions:

Function	Syntax	Description
day	day(x) → bigint	Returns the day of the month from x.
day_of_month	day_of_month(x) → bigint	This is an alias for `day`.
dayofweek	day_of_week(x) → bigint	Returns the ISO day of the week from x.
day_of_year	day_of_year(x) → bigint	Returns the day of the year from x.
dow	dow(x) → bigint	This is an alias for `day_of_week`.
doy	doy(x) → bigint	This is an alias for `day_of_year`.
hour	hour(x) → bigint	Returns the hour of the day from x. The value ranges from 0 to 23.
minute	minute(x) → bigint	Returns the minute from x. The value ranges from 0 to 59.
month	month(x) → bigint	Returns the month of the year from x. The value ranges from 1 to 12.
quarter	quarter(x) → bigint	Returns the quarter of the year from x.
second	second(x) → bigint	Returns the second from x. The value ranges from 0 to 59.
timezone_hour	timezone_hour(timestamp) → bigint	Returns the hour of the time zone offset from timestamp.
timezone_minute	timezone_minute(timestamp) → bigint	Returns the minute of the time zone offset from timestamp.
week	week(x) → bigint	Returns the ISO week of the year from x. The value ranges from 1 to 53.
week_of_year	week_of_year(x) → bigint	This is an alias for `week`.
year	year(x) → bigint	Returns the year from x.
year_of_week	year_of_week(x) → bigint	Returns the year of a week from `x`. (For more information, see *ISO Week*.)

Function	Syntax	Description
yow	yow(x) → bigint	This is an alias for `year_of_week`.

- MySQL date functions

Presto uses format strings that are compatible with MySQL `date_parse` and `str_to_date` functions. The format strings are as follows:

■ date_format(timestamp, format) → varchar

Formats `timestamp` as a string using `format`.

■ date_parse(string, format) → timestamp

Parses strings into a timestamp using `format`.

MySQL format specifiers supported by Presto are shown in the following table:

Specifier	Description
%a	Abbreviated weekday name (Mon, Tue, ⋯, Sun).
%b	Abbreviated month name (Jan, Feb, ⋯, Dec).
%c	Month, numeric (1, 2, ⋯, 12). This cannot be zero.
%d	Day of the month, numeric (01, 02, ⋯, 31). This cannot be zero.
%e	Day of the month, numeric (1, 2, ⋯, 31). This cannot be zero.
%f	Fraction of second (6 digits for printing: 000000 ⋯ 999000; 1 to 9 digits for parsing: 0 ⋯ 999999999).
%H	Hour (00, 01, ⋯, 23).
%h	Hour (01, 02, ⋯, 12).
%I	Hour (01, 02, ⋯, 12).
%i	Minutes, numeric (00, 01, ⋯, 59).
%j	Day of year (001, 002, ⋯, 365).
%k	Hour (0, 1, ⋯, 23).
%l	Hour (1, 2, ⋯, 12).
%M	Month name (January, February, ⋯, December).
%m	Month, numeric (01, 02, ⋯, 12) [4].
%p	AM/PM

Specifier	Description
%r	Time, 12-hour (hh:mm:ss AM/PM).
%S	Seconds (00, 01, …, 59).
%s	Seconds (00, 01, …, 59).
%T	Time, 24-hour (hh:mm:ss).
%v	Week (01, 02, …, 53), where Monday is the first day of the week. Used with `%X`.
%W	Weekday name (Monday, Tuesday, …, Sunday).
%x	Year of the week, where Monday is the first day of the week, numeric, four digits.
%Y	Year, numeric, four digits.
%y	Year, numeric (two digits). When parsing, the two-digit year format assumes the range [1970 … 2069].
%%	A literal '%' character.

Note:

The following specifiers are not currently supported: %D, %U, %u, %V, %w, and %X.

- Java date functions

  The functions in this section use a format string that is compatible with *Joda-Time's DateTimeFormatter pattern*.

  - format_datetime(timestamp, format) → varchar: Formats timestamps.
  - parse_datetime(string, format) → timestamp with time zone: Parses strings into a timestamp.

Aggregate functions

Aggregate functions have the following features:

- Input data sets
- Output single computation results.

Almost all of these aggregate functions ignore `null` values and return `null` for rows with no input or when all values are `null`. However, there are a few notable exceptions:

- count
- count_if
- max_by
- min_by
- approx_distinct

- Basic aggregate functions

Function	Syntax	Description
arbitrary	arbitrary(x) → [same as input]	Returns an arbitrary non-null value of x .
array_agg	array_agg(x) → array<[same as input]>	Returns an array created from the input x elements.
avg	avg(x) → double	Returns the mean of all input values.
avg	avg(time interval type) → time interval type	Returns the average interval length of all input values.
bool_and	bool_and(boolean) → boolean	Returns TRUE if every input value is TRUE, otherwise FALSE.
bool_or	bool_or(boolean) → boolean	Returns TRUE if any input value is TRUE , otherwise FALSE.
checksum	checksum(x) → varbinary	Returns an order-insensitive checksum of the given values.
count	count(*) → bigint	Returns the number of input rows.
count	count(x) → bigint	Returns the number of non-null input values.
count_if	count_if(x) → bigint	Returns the number of TRUE input values. This function is equivalent to `count(CASE WHEN x THEN 1 END)`.
every	every(boolean) → boolean	This is an alias for bool_and.
geometric_mean	geometric_mean(x) → double	Returns the geometric mean of all input values.
max_by	max_by(x, y) → [same as x]	Returns the value of x associated with the maximum value of y over all input values.
max_by	max_by(x, y, n) → array<[same as x]>	Returns n values of x associated with the n largest of all input values of y in descending order of y.

Function	Syntax	Description
min_by	min_by(x, y) → [same as x]	Returns the value of x associated with the minimum value of y over all input values.
min_by	min_by(x, y, n) → array<[same as x]>	Returns n values of x associated with the n smallest of all input values of y in ascending order of y.
max	max(x) → [same as input]	Returns the maximum value of all input values.
max	max(x, n) → array<[same as x]>	Returns n largest values of all input values of x.
min	min(x) → [same as input]	Returns the minimum value of all input values.
min	min(x, n) → array<[same as x]>	Returns n smallest values of all input values of x.
sum	sum(x) → [same as input]	Returns the sum of all input values.

- Bitwise aggregate functions

  For bitwise aggregate functions, refer to the `bitwise_and_agg` and `bitwise_or_agg` functions described in *General aggregate functions*.

- Map aggregate functions

Function	Syntax	Description
histogram	histogram(x) → map	Returns a map containing the count of the number of times each input value occurs.
map_agg	map_agg(key, value) → map	Returns a map created from the input key/value pairs.
map_union	map_union(x) → map	Returns the union of all the input maps . If a key is found in multiple input maps, that key's value in the resulting map comes from an arbitrary input map .
multimap_agg	multimap_agg(key, value) → map>	Returns a multimap created from the input key/value pairs.

- Close aggregate function

Function	Syntax	Description
approx_distinct	approx_distinct(x, [e]) → bigint	Returns the approximate number of distinct input values. This function provides an approximation of `count (DISTINCT x)`. Zero is returned if all input values are null. This function should produce a standard error of no more than `e`, which is the standard deviation of the (approximately normal) error distribution over all possible sets. It is optional, and is 2.3% by default. The current implementation of this function requires that `e` be in the range of [0.01150, 0.26000]. It does not guarantee an upper bound of the error for any specific input set.
approx_percentile	approx_percentile(x, percentage) → [same as x]	Returns the approximate percentile for all input values of x at the given percentage.
approx_percentile	approx_percentile(x, percentages) → array<[same as x]>	Similar to the preceding function, percentages is an array, and returns constant values for all input rows.
approx_percentile	approx_percentile(x, w, percentage) → [same as x]	Similar to the preceding function, w is the weighted value of x.
approx_percentile	approx_percentile(x, w, percentage, accuracy) → [same as x]	Similar to the preceding function, `accuracy` is the upper bound of the estimation accuracy, and the value must be in the range of [0, 1].
approx_percentile	approx_percentile(x, w , percentages) → array<[same as x]>	Similar to the preceding function, percentages is an array, and returns constant values for all input rows.
numeric_histogram	numeric_histogram(buckets, value, [weight]) → map	Computes an approximate histogram with up to a given number of buckets. `buckets` must be a BIGINT. `value` and `weight` must be numeric. `weight` is optional, and is 1 by default.

- Statistical aggregate functions

Function	Syntax	Description
corr	corr(y, x) → double	Returns the correlation coefficient of input values.
covar_pop	covar_pop(y, x) → double	Returns the population covariance of input values.
covar_samp	covar_samp(y, x) → double	Returns the sample covariance of input values.
kurtosis	kurtosis(x) → double	Returns the excess kurtosis of all input values. Unbiased estimate using the following expression: kurtosis(x) = n(n+1)/((n-1)(n-2)(n-3))sum[(x_i-mean)^4]/sttdev(x)^4-3(n-1)^2/((n-2)(n-3))
regr_intercept	regr_intercept(y, x) → double	Returns the linear regression intercept of input values. *y* is the dependent value, whereas *x* is the independent value.
regr_slope	regr_slope(y, x) → double	Returns the linear regression slope of input values. *y* is the dependent value, whereas *x* is the independent value.
skewness	skewness(x) → double	Returns the skewness of all input values.
sttdev_pop	sttdev_pop(x) → double	Returns the population standard deviation of all input values.
sttdev_samp	sttdev_samp(x) → double	Returns the sample standard deviation of all input values.
sttdev	sttdev(x) → double	This is an alias for `sttdev_samp`.
var_pop	var_pop(x) → double	Returns the population variance of all input values.
var_samp	var_samp(x) → double	Returns the sample variance of all input values.
variance	variance(x) → double	This is an alias for `var_samp`.

9.8.6 SQL statements

This section provides an overview of SQL statements.

ALTER SCHEMA

- Synopsis

```
ALTER SCHEMA name RENAME TO new_name
```

- Description

  Renames schemas.

- Examples

```
ALTER SCHEMA web RENAME TO traffic -- Renames Schema 'web' as '
traffic'
```

ALTER TABLE

- Synopsis

```
ALTER TABLE name RENAME TO new_name
ALTER TABLE name ADD COLUMN column_name data_type
ALTER TABLE name DROP COLUMN column_name
ALTER TABLE name RENAME COLUMN column_name TO new_column_name
```

- Description

  Changes the definition of an existing table.

- Examples

```
ALTER TABLE users RENAME TO people; --- Rename
ALTER TABLE users ADD COLUMN zip varchar; --- Add column
ALTER TABLE users DROP COLUMN zip; --- Drop column
ALTER TABLE users RENAME COLUMN id TO user_id; --- Rename column
```

CALL

- Synopsis

```
CALL procedure_name ( [ name => ] expression [, ...] )
```

- Description

  Calls a stored procedure. Stored procedures can be provided by connectors to perform data manipulation or administrative tasks. Some connectors, such as the PostgreSQL connector, are for systems that have their own stored procedures. These systems must use the stored procedures provided by the connectors to access their own stored procedures, which are not directly callable through CALL.

- **Examples**

```
CALL test(123, 'apple'); --- Call a stored procedure using
positional arguments
CALL test(name => 'apple', id => 123); --- Call a stored procedure
using named arguments
CALL catalog.schema.test(); --- Call a stored procedure using a
fully qualified name
```

COMMIT

- **Synopsis**

```
COMMIT [WORK]
```

- **Description**

  Commits the current transaction.

- **Examples**

```
COMMIT;
COMMIT WORK;
```

CREATE SCHEMA

- **Synopsis**

```
CREATE SCHEMA [ IF NOT EXISTS ] schema_name
[ WITH ( property_name = expression [, ...] ) ]
```

- **Description**

  Creates a new schema. A schema is a container that holds tables, views, and other database objects.

  - The optional `IF NOT EXISTS` clause causes the error to be suppressed if the schema already exists.
  - The optional `WITH` clause can be used to set properties on the newly created schema. To list all available schema properties, run the following query:

    ```
    SELECT * FROM system.metadata.schema_properties;
    ```

- **Examples**

```
CREATE SCHEMA web;
CREATE SCHEMA hive.sales;
```

```
CREATE SCHEMA IF NOT EXISTS traffic;
```

CREATE TABLE

- **Synopsis**

```
CREATE TABLE [ IF NOT EXISTS ]
table_name (
  { column_name data_type [ COMMENT comment ]
  | LIKE existing_table_name [ { INCLUDING | EXCLUDING } PROPERTIES
 ] }
  [, ...]
)
[ COMMENT table_comment ]
[ WITH ( property_name = expression [, ...] ) ]
```

- **Description**

  Creates an empty table. Use `CREATE TABLE AS` to create a table from an existing data set.

  - **The optional `IF NOT EXISTS` clause causes the error to be suppressed if the table already exists.**
  - **The optional `WITH` clause can be used to set properties on the newly created table. To list all available table properties, run the following query:**

```
SELECT * FROM system.metadata.table_properties;
```

  - **The `LIKE` clause can be used to include all the column definitions from an existing table in the new table. Multiple `LIKE` clauses may be specified.**
  - **If `INCLUDING PROPERTIES` is specified, all of the table properties are copied to a new table. If the `WITH` clause specifies the same property name as one of the copied properties using `INCLUDING PROPERTIES`, the value from the `WITH` clause is used. The default behavior is `EXCLUDING PROPERTIES`.**

- **Examples**

```
--- Create a new table orders:
CREATE TABLE orders (
  orderkey bigint,
  orderstatus varchar,
  totalprice double,
  orderdate date
)
WITH (format = 'ORC')
--- Create the table orders if it does not already exist, adding a
table comment and a column comment:
CREATE TABLE IF NOT EXISTS orders (
  orderkey bigint,
  orderstatus varchar,
  totalprice double COMMENT 'Price in cents.',
  orderdate date
)
COMMENT 'A table to keep track of orders.'
Create the table bigger_orders, using some column definitions from
orders:
CREATE TABLE bigger_orders (
  another_orderkey bigint,
  LIKE orders,
  another_orderdate date
)
```

CREATE TABLE AS

- **Synopsis**

  ```
  CREATE TABLE [ IF NOT EXISTS ] table_name [ ( column_alias, ... ) ]
  [ COMMENT table_comment ]
  [ WITH ( property_name = expression [, ...] ) ]
  AS query
  [ WITH [ NO ] DATA ]
  ```

- **Description**

  Creates a new table containing the result of a `SELECT` query.

  - **The optional `IF NOT EXISTS` clause causes the error to be suppressed if the table already exists.**
  - **The optional `WITH` clause can be used to set properties on the newly created table. To list all available table properties, run the following query:**

    ```
    SELECT * FROM system.metadata.table_properties;
    ```

- **Examples**

  ```
  --- Select two columns from orders to create a new table
  CREATE TABLE orders_column_aliased (order_date, total_price)
  AS
  SELECT orderdate, totalprice
  FROM orders
  --- Create a new table using the aggregate function
  CREATE TABLE orders_by_date
  COMMENT 'Summary of orders by date'
  WITH (format = 'ORC')
  AS
  SELECT orderdate, sum(totalprice) AS price
  FROM orders
  GROUP BY orderdate
  --- Create a new table, using the **IF NOT EXISTS** clause
  CREATE TABLE IF NOT EXISTS orders_by_date AS
  SELECT orderdate, sum(totalprice) AS price
  FROM orders
  GROUP BY orderdate
  --- Create a new table with the same schema as nation and no data
  Create Table maid
  SELECT *
  FROM nation
  ```

```
WITH NO DATA
```

CREATE VIEW

- **Synopsis**

```
CREATE [ OR REPLACE ] VIEW view_name AS query
```

- **Description**

  Creates a view. The view is a logic table that does not contain any data. It can be referenced by future queries. The query stored by the view is run every time the view is referenced by another query.

  The optional `OR REPLACE` clause causes the view to be replaced if it already exists. It does not raise an error.

- **Examples**

```
--- Create a simple view
CREATE VIEW test AS
SELECT orderkey, orderstatus, totalprice / 2 AS half
FROM orders
--- Create view using the aggregate function
CREATE VIEW orders_by_date AS
SELECT orderdate, sum(totalprice) AS price
FROM orders
GROUP BY orderdate
--- Create a view that replaces an existing view
CREATE OR REPLACE VIEW test AS
SELECT orderkey, orderstatus, totalprice / 4 AS quarter
FROM orders
```

DEALLOCATE PREPARE

- **Synopsis**

```
DEALLOCATE PREPARE statement_name
```

- **Description**

  Removes a statement with the name statement_name from the list of prepared statements in a session.

- **Examples**

```
--- Deallocate a statement named my_query
```

```
DEALLOCATE PREPARE my_query;
```

DELETE

- Synopsis

  ```
  DELETE FROM table_name [ WHERE condition ]
  ```

- Description

  If the `WHERE` clause is specified, delete the matching rows from the table. If it is not specified, all rows from the table are deleted.

- Examples

  ```
  --- Delete the matching row
  DELETE FROM lineitem WHERE shipmode = 'AIR';
  --- Delete the matching row
  DELETE FROM lineitem
  WHERE orderkey IN (SELECT orderkey FROM orders WHERE priority = 'LOW
  ');
  --- Clear the table
  DELETE FROM orders;
  ```

- Limitations

  Some connectors have limits or do not support `DELETE`.

DESCRIBE

- Synopsis

  ```
  DESCRIBE table_name
  ```

- Description

  Retrieves the table definitions, and is an alias for *SHOW COLUMNS*.

- Examples

  ```
  DESCRIBE orders;
  ```

DESCRIBE INPUT

- Synopsis

  ```
  DESCRIBE INPUT statement_name
  ```

- Description

  Lists the input parameters of a prepared statement along with the position and type of each parameter.

- **Examples**

  ```
  --- Create a pre-compiled query 'my_ select1'
  PREPARE my_select1 FROM
  SELECT ? From nation where regionkey =? AND name < ? ;
  --- Get the descriptive information of this prepared statement
  DESCRIBE INPUT my_select1;
  ```

  DESCRIBE INPUT my_select1;

  ```
  Position | Type
  --------------------
          0 | unknown
          1 | bigint
          2 | varchar
  (3 rows)
  ```

DESCRIBE OUTPUT

- **Synopsis**

  ```
  DESCRIBE OUTPUT statement_name
  ```

- **Description**

  Lists the output columns of a prepared statement, including the column name (or alias), catalog, schema, table name, type, type size in bytes, and a boolean indicating if the column is aliased.

- **Examples**

  - **Example one**

    Prepare a prepared statement:

    ```
    PREPARE my_select1 FROM
    SELECT * FROM nation;
    ```

    Execute `DESCRIBE OUTPUT`, which outputs the following:

    ```
    DESCRIBE OUTPUT my_select1;
     Column Name | Catalog | Schema | Table  |  Type   | Type Size |
    Aliased
    -------------+---------+--------+--------+---------+-----------
    +---------
     nationkey   | tpch    | sf1    | nation | bigint  |         8 |
    false
     name        | tpch    | sf1    | nation | varchar |         0 |
    false
     regionkey   | tpch    | sf1    | nation | bigint  |         8 |
    false
     comment     | tpch    | sf1    | nation | varchar |         0 |
    false
    ```

```
(4 rows)
```

- **Example two**

  ```
  PREPARE my_select2 FROM
  SELECT count(*) as my_count, 1+2 FROM nation
  ```

  Execute `DESCRIBE OUTPUT`, which outputs the following:

  ```
  DESCRIBE OUTPUT my_select2;
   Column Name | Catalog | Schema | Table |  Type  | Type Size |
  Aliased
  -------------+---------+--------+-------+--------+-----------
  +---------
   my_count    |         |        |       | bigint |         8 |
  true
   _col1       |         |        |       | bigint |         8 |
  false
  (2 rows)
  ```

- **Example three:**

  ```
  PREPARE my_create FROM
  CREATE TABLE foo AS SELECT * FROM nation;
  ```

  Execute `DESCRIBE OUTPUT`, which outputs the following:

  ```
  DESCRIBE OUTPUT my_create;
   Column Name | Catalog | Schema | Table |  Type  | Type Size |
  Aliased
  -------------+---------+--------+-------+--------+-----------
  +---------
   rows        |         |        |       | bigint |         8 |
  false
  (1 row)
  ```

DROP SCHEMA

- **Synopsis**

  ```
  DROP SCHEMA [ IF EXISTS ] schema_name
  ```

- **Description**

  Drops an existing schema.

  - **The schema must be empty.**
  - **The optional `IF EXISTS` clause causes the error to be suppressed if the schema does not exist.**

- **Examples**

  ```
  DROP SCHEMA web;
  ```

```
DROP TABLE IF EXISTS sales;
```

DROP TABLE

- **Synopsis**

  ```
  DROP TABLE [ IF EXISTS ] table_name
  ```

- **Description**

  Drops an existing table. The optional `IF EXISTS` clause causes the error to be suppressed if the table does not exist.

- **Examples**

  ```
  DROP TABLE orders_by_date;
  DROP TABLE IF EXISTS orders_by_date;
  ```

DROP VIEW

- **Synopsis**

  ```
  DROP VIEW [ IF EXISTS ] view_name
  ```

- **Description**

  Drops an existing view. The optional `IF EXISTS` clause causes the error to be suppressed if the view does not exist.

- **Examples**

  ```
  DROP VIEW orders_by_date;
  DROP VIEW IF EXISTS orders_by_date;
  ```

EXECUTE

- **Synopsis**

  ```
  EXECUTE statement_name [ USING parameter1 [ , parameter2, ... ] ]
  ```

- **Description**

  Executes a prepared statement. Parameter values are defined in the `USING` clause.

- **Examples**

  - **Example one**

    ```
    PREPARE my_select1 FROM
    SELECT name FROM nation;
    --- Execute a prepared statement
    ```

```
EXECUTE my_select1;
```

- Example two

```
PREPARE my_select2 FROM
SELECT name FROM nation WHERE regionkey = ? and nationkey < ? ;
--- Execute a prepared statement
EXECUTE my_select2 USING 1, 3;
--- The preceding statement is equivalent to executing the
following statement:
SELECT name FROM nation WHERE regionkey = 1 AND nationkey < 3;
```

EXPLAIN

- Synopsis

```
EXPLAIN [ ( option [, ...] ) ] statement
where option can be one of:
    FORMAT { TEXT | GRAPHVIZ }
    TYPE { LOGICAL | DISTRIBUTED | VALIDATE }
```

- Description

  Achieves one of the following functions based on the option used:

  - Shows the logical plan of a query statement
  - Shows the distributed execution plan of a query statement
  - Validates a query statement

  Use the `TYPE DISTRIBUTED` option to display plan fragments. Each fragment is executed by one or more Presto nodes. Fragment separation represents the data exchange between Presto nodes. Fragment type specifies how the fragment is executed by Presto nodes and how the data is distributed between fragments. Fragment types are as follows:

  - `SINGLE`: Fragments are executed on a single node.
  - `HASH`: Fragments are executed on a fixed number of nodes with the input data distributed using a hash function.
  - `ROUND_ROBIN`: Fragments are executed on a fixed number of nodes with the input data distributed in a `ROUND-ROBIN` fashion.
  - `BROADCAST`: Fragments are executed on a fixed number of nodes with the input data broadcast to all nodes.
  - `SOURCE`: Fragments are executed on nodes where input splits are accessed.
- Examples

  - Example one

Logical plan:

```
presto:tiny> EXPLAIN SELECT regionkey, count(*) FROM nation GROUP
BY 1;
                                                Query Plan
----------------------------------------------------------------------------
 - Output[regionkey, _col1] => [regionkey:bigint, count:bigint]
         _ Col1: = count?
     - RemoteExchange[GATHER] => regionkey:bigint, count:bigint
         - Aggregate(FINAL)[regionkey] => [regionkey:bigint, count
:bigint]
                count := "count"("count_8")
             - LocalExchange[HASH][$hashvalue] ("regionkey") =>
regionkey:bigint, count_8:bigint, $hashvalue:bigint
                 - RemoteExchange[REPARTITION][$hashvalue_9] =>
regionkey:bigint, count_8:bigint, $hashvalue_9:bigint
                     - Project[] => [regionkey:bigint, count_8:
bigint, $hashvalue_10:bigint]
                             $hashvalue_10 := "combine_hash"(
BIGINT '0', COALESCE("$operator$hash_code"("regionkey"), 0))
                         - Aggregate(PARTIAL)[regionkey] => [
regionkey:bigint, count_8:bigint]
                                 count_8 := "count"(*)
                             - TableScan[tpch:tpch:nation:sf0.1,
originalConstraint = true] => [regionkey:bigint]
                                     regionkey := tpch:regionkey
```

- **Example two**

Distributed plan:

```
presto:tiny> EXPLAIN (TYPE DISTRIBUTED) SELECT regionkey, count
(*) FROM nation GROUP BY 1;
                                          Query Plan
----------------------------------------------------------------------------
 Fragment 0 [SINGLE]
     Output layout: [regionkey, count]
     Output partitioning: SINGLE []
     - Output[regionkey, _col1] => [regionkey:bigint, count:bigint
]
             _col1 := count
         - RemoteSource[1] => [regionkey:bigint, count:bigint]
 Fragment 1 [HASH]
     Output layout: [regionkey, count]
     Output partitioning: SINGLE []
     - Aggregate(FINAL)[regionkey] => [regionkey:bigint, count:
bigint]
             count := "count"("count_8")
         - LocalExchange[HASH][$hashvalue] ("regionkey") =>
regionkey:bigint, count_8:bigint, $hashvalue:bigint
             - RemoteSource[2] => [regionkey:bigint, count_8:
bigint, $hashvalue_9:bigint]
 Fragment 2 [SOURCE]
     Output layout: [regionkey, count_8, $hashvalue_10]
     Output partitioning: HASH [regionkey][$hashvalue_10]
     - Project[] => [regionkey:bigint, count_8:bigint, $hashvalue_
10:bigint]
             $hashvalue_10 := "combine_hash"(BIGINT '0', COALESCE
("$operator$hash_code"("regionkey"), 0))
         - Aggregate(PARTIAL)[regionkey] => [regionkey:bigint,
count_8:bigint]
```

```
                 count_8 := "count"(*)
             - TableScan[tpch:tpch:nation:sf0.1, originalCo
nstraint = true] => [regionkey:bigint]
                     regionkey := tpch:regionkey
```

- **Example three:**

  Validation:

  ```
  presto:tiny> EXPLAIN (TYPE VALIDATE) SELECT regionkey, count(*)
  FROM nation GROUP BY 1;
   Valid
  -------
   true
  ```

EXPLAIN ANALYZE

- **Synopsis**

  ```
  EXPLAIN ANALYZE [VERBOSE] statement
  ```

- **Description**

  Executes the statement and shows its distributed execution plan along with the cost of each operation. The `VERBOSE` option gives more detailed information and low-level statistics.

- **Examples**

  In the following example, you can see the CPU time spent in each stage, as well as the relative cost of each plan node in the stage. Note that the relative cost of the plan nodes is based on wall time, which may or may not be correlated to CPU time . For each plan node, you can see additional statistics, which are useful if you want to detect data anomalies for a query such as skewness or abnormal hash collisions.

  ```
  presto:sf1> EXPLAIN ANALYZE SELECT count(*), clerk FROM orders WHERE
   orderdate > date '1995-01-01' GROUP BY clerk;
                                            Query Plan
  ------------------------------------------------------------------------------
  Fragment 1 [HASH]
      Cost: CPU 88.57ms, Input: 4000 rows (148.44kB), Output: 1000
  rows (28.32kB)
      Output layout: [count, clerk]
      Output partitioning: SINGLE []
      - Project[] => [count:bigint, clerk:varchar(15)]
              Cost: 26.24%, Input: 1000 rows (37.11kB), Output: 1000
  rows (28.32kB), Filtered: 0.00%
              Input avg.: 62.50 lines, Input std.dev.: 14.77%
          - Aggregate(FINAL)[clerk][$hashvalue] => [clerk:varchar(15),
   $hashvalue:bigint, count:bigint]
                  Cost: 16.83%, Output: 1000 rows (37.11kB)
                  Input avg.: 250.00 lines, Input std.dev.: 14.77%
                  count := "count"("count_8")
              - LocalExchange[HASH][$hashvalue] ("clerk") => clerk:
  varchar(15), count_8:bigint, $hashvalue:bigint
  ```

```
                    Cost: 47.28%, Output: 4000 rows (148.44kB)
                    Input avg.: 4000.00 lines, Input std.dev.: 0.00%
                - RemoteSource[2] => [clerk:varchar(15), count_8:
bigint, $hashvalue_9:bigint]
                        Cost: 9.65%, Output: 4000 rows (148.44kB)
                        Input avg.: 4000.00 lines, Input std.dev.: 0
.00%
Fragment 2 [tpch:orders:1500000]
    Cost: CPU 14.00s, Input: 818058 rows (22.62MB), Output: 4000
rows (148.44kB)
    Output layout: [clerk, count_8, $hashvalue_10]
    Output partitioning: HASH [clerk][$hashvalue_10]
    - Aggregate(PARTIAL)[clerk][$hashvalue_10] => [clerk:varchar(15
), $hashvalue_10:bigint, count_8:bigint]
            Cost: 4.47%, Output: 4000 rows (148.44kB)
            Input avg.: 204514.50 lines, Input std.dev.: 0.05%
            Collisions avg.: 5701.28 (17569.93% est.), Collisions
std.dev.: 1.12%
            count_8 := "count"(*)
        - ScanFilterProject[table = tpch:tpch:orders:sf1.0,
originalConstraint = ("orderdate" > "$literal$date"(BIGINT '9131
')), filterPredicate = ("orderdate" > "$literal$date"(BIGINT '9131
'))] => [cler
                Cost: 95.53%, Input: 1500000 rows (0B), Output:
818058 rows (22.62MB), Filtered: 45.46%
                Input avg.: 375000.00 lines, Input std.dev.: 0.00%
                $hashvalue_10 := "combine_hash"(BIGINT '0', COALESCE
("$operator$hash_code"("clerk"), 0))
                orderdate := tpch:orderdate
                clerk := tpch:clerk
```

If the `VERBOSE` option is used, some operators may report additional information.

```
EXPLAIN ANALYZE VERBOSE SELECT count(clerk) OVER() FROM orders WHERE
 orderdate > date '1995-01-01';
                                          Query Plan
-------------------------------------------------------------------------------
  ...
         - Window[] => [clerk:varchar(15), count:bigint]
                 Cost: {rows: ?, bytes: ?}
                 CPU fraction: 75.93%, Output: 8130 rows (230.24kB)
                 Input avg.: 8130.00 lines, Input std.dev.: 0.00%
                 Active Drivers: [ 1 / 1 ]
                 Index size: std.dev.: 0.00 bytes , 0.00 rows
                 Index count per driver: std.dev.: 0.00
                 Rows per driver: STD. Dev.: 0.00
                 Size of partition: std.dev.: 0.00
                 count := count("clerk")
 ...
```

GRANT

- **Synopsis**

```
GRANT ( privilege [, ...] | ( ALL PRIVILEGES ) )
ON [ TABLE ] table_name TO ( grantee | PUBLIC )
[ WITH GRANT OPTION ]
```

- **Description**

Grants specified privileges to the specified grantee.

- Specifying `ALL PRIVILEGES` grants `DELETE`, `INSERT` and `SELECT` privileges.
- Specifying `PUBLIC` grants privileges to the `PUBLIC` role and in doing so to all users.
- The optional `WITH GRANT OPTION` clause allows the grantee to grant these same privileges to others.

- Examples

```
GRANT INSERT, SELECT ON orders TO alice; --- Grant privileges to
user alice
GRANT SELECT ON nation TO alice WITH GRANT OPTION; --- Grant SELECT
 privilege to user alice, additionally allowing alice to grant **
SELECT** privilege to others
GRANT SELECT ON orders TO PUBLIC; --- Grant **SELECT** privilege on
the table order to everyone
```

- Limitations

Some connectors do not support `GRANT`.

INSERT

- Synopsis

```
INSERT INTO table_name [ ( column [, ... ] ) ] query
```

- Description

Inserts new rows into a table. If the list of column names is specified, they must be identical to the list of columns produced by the `query`. Each column in the table not present in the column list is filled with a `null` value.

- Examples

```
INSERT INTO orders SELECT * FROM new_orders; --- Insert the SELECT
results into the orders table.
INSERT INTO cities VALUES (1, 'San Francisco'); --- Insert a single
row
INSERT INTO cities VALUES (2, 'San Jose'), (3, 'Oakland'); ---
Insert multiple rows
INSERT INTO nation (nationkey, name, regionkey, comment) VALUES (26,
 'POLAND', 3, 'no comment'); --- Insert a single row
```

```
INSERT INTO nation (nationkey, name, regionkey) VALUES (26, 'POLAND
', 3); --- Inserts a single row (only includes some columns)
```

PREPARE

- **Synopsis**

```
PREPARE statement_name FROM statement
```

- **Description**

  Prepares a statement for execution at a later time. Prepared statements are queries saved in a session with a given name. The statement can include parameters in place of literals to be replaced at the time of execution. Parameters are represented by `?`.

- **Examples**

```
--- Prepare a query that does not include parameters
PREPARE my_select1 FROM
SELECT * FROM nation;
--- Prepare a query that includes parameters
PREPARE my_select2 FROM
SELECT name FROM nation WHERE regionkey = ? AND nationkey < ? ;
--- Prepare an insert statement that does not include parameters
PREPARE my_insert FROM
INSERT INTO cities VALUES (1, 'San Francisco');
```

RESET SESSION

- **Synopsis**

```
RESET SESSION name
RESET SESSION catalog.name
```

- **Description**

  Resets a session property value to the default value.

- **Examples**

```
RESET SESSION optimize_hash_generation;
RESET SESSION hive.optimized_reader_enabled;
```

REVOKE

- **Synopsis**

```
REVOKE [ GRANT OPTION FOR ]
(Privilege [,...] | ALL PRIVILEGES )
ON [ TABLE ] table_name FROM ( grantee | PUBLIC )
```

- **Description**

  Revokes specified privileges from the specified grantee.

  - Specifying `ALL PRIVILEGE` revokes `SELECT`, `INSERT` and `DELETE` privileges.
  - Specifying `PUBLIC` revokes privileges from the `PUBLIC` role. You retain privileges assigned to you directly or through other roles.
  - The optional `GRANT OPTION FOR` clause also revokes the privilege to `GRANT` specified privileges.
  - Usage of the term `grantee` denotes both users and roles.
- Examples

```
--- Revoke INSERT and SELECT privileges on the table orders from
user alice
REVOKE INSERT, SELECT ON orders FROM alice;
--- Revoke SELECT privilege on the table nation from everyone,
--- additionally revoking the privilege to grant SELECT privilege to
 others
REVOKE GRANT OPTION FOR SELECT ON nation FROM PUBLIC;
--- Revoke all privileges on the table test from user alice
REVOKE ALL PRIVILEGES ON test FROM alice;
```

- Limitations

  Some connectors do not support `REVOKE`.

ROLLBACK

- Synopsis

```
ROLLBACK [ WORK ]
```

- Description

  Rolls back the current transaction.

- Examples

```
ROLLBACK;
ROLLBACK WORK;
```

SELECT

- Synopsis

```
[ WITH with_query [, ...] ]
SELECT [ ALL | DISTINCT ] select_expr [, ...]
[ FROM from_item [, ...] ]
[ WHERE condition ]
[ GROUP BY [ ALL | DISTINCT ] grouping_element [, ...] ]
[ HAVING condition]
[ { UNION | INTERSECT | EXCEPT } [ ALL | DISTINCT ] select ]
[ ORDER BY expression [ ASC | DESC ] [, ...] ]
[ LIMIT [ count | ALL ] ]
```

where `from_item` is either:

```
Table_name [[as] alias [(column_alias [,...] ) ] ]
```

```
From_item join_type from_item [ON join_condition | using (join_colum
n [,...] ) ]
```

and `join_type` is either:

- [INNER] JOIN
- LEFT [OUTER] JOIN
- RIGHT [OUTER] JOIN
- FULL [OUTER] JOIN
- CROSS JOIN

and `grouping_element` is either:

- ()
- expression
- GROUPING SETS ((column [, …]) [, …])
- CUBE (column [, …])
- ROLLUP (column [, …])

- Description

  Retrieves rows from zero or more tables to get data sets.
- WITH clause
  - Basic functions

    The WITH clause defines named relations for use within a query. It allows flattening nested queries or simplifying subqueries. For example, the following queries are equivalent:

    ```
    --- The WITH clause is not used
    SELECT a, b
    FROM (
      SELECT a, MAX(b) AS b FROM t GROUP BY a
    ) AS x;
    --- The WITH clause is used, and the query statement looks to be
    much clearer
    WITH x AS (SELECT a, MAX(b) AS b FROM t GROUP BY a)
    SELECT a, b FROM x;
    ```
  - Define multiple subqueries

The WITH clause can be used to define multiple subqueries:

```
WITH
  t1 AS (SELECT a, MAX(b) AS b FROM x GROUP BY a),
  t2 AS (SELECT a, AVG(d) AS d FROM y GROUP BY a)
SELECT t1.*, t2. *
FROM t1
JOIN t2 ON t1.a = t2.a;
```

- Form a chain structure

  Additionally, the relations within a WITH clause can chain:

```
WITH
  x AS (SELECT a FROM t),
  y AS (SELECT a AS b FROM x),
  z AS (SELECT b AS c FROM y)
SELECT c FROM z;
```

- GROUP BY clause

  - Basic functions

    The `GROUP BY` clause divides the output of a `SELECT` statement into groups of rows containing matching values. A simple `GROUP BY` clause may contain any expression composed of input columns or it may be an ordinal number selecting an output column by position (starting at one).

    The following queries are equivalent (the position for the `nationkey` column is two).

```
--- Using the ordinal number
SELECT count(*), nationkey FROM customer GROUP BY 2;
--- Using the input column name
SELECT count(*), nationkey FROM customer GROUP BY nationkey;
```

    `GROUP BY` clauses can group output by input column names that do not appear in the output of a select statement. For example:

```
--- The mktsegment column has not been specified in the SELECT
list.
--- The result set does not contain content of the mktsegment
column.
SELECT count(*) FROM customer GROUP BY mktsegment;
 _col0
-------
 29968
 30142
 30189
 29949
 29752
```

```
(5 rows)
```

Note:

When a `GROUP BY` clause is used in a `SELECT` statement, all output expressions must be either aggregate functions or columns present in the `GROUP BY` clause.

- Complex grouping operations

  Presto supports the following three complex aggregation syntaxes. This allows you to perform analysis that requires aggregation on multiple sets of columns in a single query:

  - GROUPING SETS

    `CUBE` `ROLLUP`

    The shipping table is a data table with five columns as follows:

```
SELECT * FROM shipping;
 origin_state | origin_zip | destination_state | destinatio
n_zip | package_weight
--------------+------------+-------------------
+-----------------+----------------
 California   |      94131 | New Jersey        |
8648 |             13
 California   |      94131 | New Jersey        |
8540 |             42
 New Jersey   |       7081 | Connecticut       |
6708 |            225
 California   |      90210 | Connecticut       |
6927 |           1337
 California   |      94131 | Colorado          |
80302 |             5
 New York     |      10002 | New Jersey        |
8540 |              3
(6 rows)
```

    It is possible to retrieve the following grouping results using a single query statement:

    - Group by origin_state and get the total package_weight.
    - Group by origin_state and origin_zip and get the total package_weight.
    - Group by destination_state and get the total package_weight.

    `GROUPING SETS` allows you to retrieve the result set of the above three groups with a single query statement, as shown below:

```
SELECT origin_state, origin_zip, destination_state, sum(
package_weight)
FROM shipping
GROUP BY GROUPING SETS (
    (origin_state),
    (origin_state, origin_zip),
    (destination_state));
 origin_state | origin_zip | destination_state | _col0
--------------+------------+-------------------+-------
 New Jersey   | NULL       | NULL              |   225
 California   | NULL       | NULL              |  1397
 New York     | NULL       | NULL              |     3
 California   |      90210 | NULL              |  1337
 California   |      94131 | NULL              |    60
 New Jersey   |       7081 | NULL              |   225
 New York     |      10002 | NULL              |     3
 NULL         | NULL       | Colorado          |     5
 NULL         | NULL       | New Jersey        |    58
 NULL         | NULL       | Connecticut       |  1562
(10 rows)
```

The preceding query may be considered logically equivalent to a `UNION ALL` of multiple `GROUP BY` queries:

```
SELECT origin_state, NULL, NULL, sum(package_weight)
FROM shipping GROUP BY origin_state
UNION ALL
SELECT origin_state, origin_zip, NULL, sum(package_weight)
FROM shipping GROUP BY origin_state, origin_zip
UNION ALL
SELECT NULL, NULL, destination_state, sum(package_weight)
FROM shipping GROUP BY destination_state;
```

However, queries with complex grouping syntax (such as `GROUPING SETS`) only read from the underlying data source once, whereas queries with the `UNION ALL` read the underlying data three times. This is why queries with a `UNION ALL` may produce inconsistent results when the data source is not deterministic.

- CUBE

The CUBE operator generates all possible grouping sets for a given set of columns. For example, the query:

```
SELECT origin_state, destination_state, sum(package_weight)
FROM shipping
Group by cube (glas_state, destiny _ State );
 origin_state | destination_state | _col0
--------------+-------------------+-------
 California   | New Jersey        |    55
 California   | Colorado          |     5
 New York     | New Jersey        |     3
 New Jersey   | Connecticut       |   225
 California   | Connecticut       |  1337
 California   | NULL              |  1397
 New York     | NULL              |     3
 New Jersey   | NULL              |   225
 NULL         | New Jersey        |    58
 NULL         | Connecticut       |  1562
 NULL         | Colorado          |     5
 NULL         | NULL              |  1625
(12 rows)
```

is equivalent to:

```
SELECT origin_state, destination_state, sum(package_weight)
FROM shipping
GROUP BY GROUPING SETS (
    (origin_state, destination_state),
    (origin_state),
    (destination_state),
    ());
```

- ROLLUP

  The ROLLUP operator generates all possible subtotals for a given set of columns. For example, the query:

```
SELECT origin_state, origin_zip, sum(package_weight)
FROM shipping
GROUP BY ROLLUP (origin_state, origin_zip);
 origin_state | origin_zip | _col2
--------------+------------+-------
 California   |      94131 |    60
 California   |      90210 |  1337
 New Jersey   |       7081 |   225
 New York     |      10002 |     3
 California   | NULL       |  1397
 New York     | NULL       |     3
 New Jersey   | NULL       |   225
 NULL         | NULL       |  1625
(8 rows)
```

  is equivalent to:

```
SELECT origin_state, origin_zip, sum(package_weight)
FROM shipping
GROUP BY GROUPING SETS ((origin_state, origin_zip), (origin_sta
te), ());
```

- Combining multiple grouping expressions

  The following three statements are equivalent:

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY
    GROUPING SETS ((origin_state, destination_state)),
    ROLLUP (origin_zip);
```

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY
    GROUPING SETS ((origin_state, destination_state)),
    GROUPING SETS ((origin_zip), ());
```

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY GROUPING SETS (
    (origin_state, destination_state, origin_zip),
    (origin_state, destination_state));
```

The output results are as follows:

```
origin_state | destination_state | origin_zip | _col3
--------------+-------------------+------------+-------
 New York     | New Jersey        |      10002 |     3
 California   | New Jersey        |      94131 |    55
 New Jersey   | Connecticut       |       7081 |   225
 California   | Connecticut       |      90210 |  1337
 California   | Colorado          |      94131 |     5
 New York     | New Jersey        | NULL       |     3
 New Jersey   | Connecticut       | NULL       |   225
 California   | Colorado          | NULL       |     5
 California   | Connecticut       | NULL       |  1337
 California   | New Jersey        | NULL       |    55
(10 rows)
```

In a `GROUP BY` clause, the `ALL` and `DISTINCT` quantifiers determine whether duplicate grouping sets each produce distinct output rows. For example, the query:

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY ALL
    CUBE (origin_state, destination_state),
    ROLLUP (origin_state, origin_zip);
```

is equivalent to:

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY GROUPING SETS (
    (origin_state, destination_state, origin_zip),
    (origin_state, origin_zip),
    (origin_state, destination_state, origin_zip),
    (origin_state, origin_zip),
    (origin_state, destination_state),
    (origin_state),
    (origin_state, destination_state),
    (origin_state),
    (origin_state, destination_state),
    (origin_state),
    (destination_state),
    ());
```

Multiple duplicate grouping sets are available. However, if the query uses the `DISTINCT` quantifier, only unique grouping sets are generated. For example, the query:

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY DISTINCT
    CUBE (origin_state, destination_state),
    ROLLUP (origin_state, origin_zip);
```

is equivalent to:

```
SELECT origin_state, destination_state, origin_zip, sum(
package_weight)
FROM shipping
GROUP BY GROUPING SETS (
    (origin_state, destination_state, origin_zip),
    (origin_state, origin_zip),
    (origin_state, destination_state),
    (origin_state),
    (destination_state),
    ());
```

Note:

The default set quantifier for `GROUP BY` is `ALL`.

- GROUPING operations

  Presto provides a `grouping` operation that returns a bit set converted to decimal, indicating which columns are present in a grouping. The semantics are demonstrated as follows:

```
grouping(col1, ..., colN) -> bigint
```

  `grouping` is used in conjunction with `GROUPING SETS`, `ROLLUP`, `CUBE`, or `GROUP BY`. `grouping` columns must be identical to the columns referenced in the corresponding `GROUPING SETS`, `ROLLUP`, `CUBE`, or `GROUP BY` clause.

```
SELECT origin_state, origin_zip, destination_state, sum(package_we
ight),
       grouping(origin_state, origin_zip, destination_state)
FROM shipping
GROUP BY GROUPING SETS (
        (origin_state),
        (origin_state, origin_zip),
        (destination_state));
origin_state | origin_zip | destination_state | _col3 | _col4
--------------+------------+-------------------+-------+-------
```

```
California   | NULL       | NULL              |  1397 |     3
 --- 011
New Jersey   | NULL       | NULL              |   225 |     3
 --- 011
New York     | NULL       | NULL              |     3 |     3
 --- 011
California   |      94131 | NULL              |    60 |     1
 --- 001
New Jersey   |       7081 | NULL              |   225 |     1
 --- 001
California   |      90210 | NULL              |  1337 |     1
 --- 001
New York     |      10002 | NULL              |     3 |     1
 --- 001
NULL         | NULL       | New Jersey        |    58 |     6
 --- 100
NULL         | NULL       | Connecticut       |  1562 |     6
 --- 100
NULL         | NULL       | Colorado          |     5 |     6
 --- 100
(10 rows)
```

As shown in the preceding table, bits are assigned to the argument columns, with the rightmost column being the least significant bit. For a given `grouping`, a bit is set to 0 if the corresponding column is included in the grouping. If it is not included, the bit is set to 1.

- HAVING clause

The `HAVING` clause is used in conjunction with aggregate functions and the `GROUP BY` clause to control which groups are selected. A `HAVING` clause is executed after grouping and aggregation are complete, which eliminates groups that do not satisfy the given conditions.

The following example selects user groups with an account balance greater than 5700000:

```
SELECT count(*), mktsegment, nationkey,
       CAST(sum(acctbal) AS bigint) AS totalbal
FROM customer
GROUP BY mktsegment, nationkey
HAVING sum(acctbal) > 5700000
ORDER BY totalbal DESC;
```

The output is as follows:

```
_col0 | mktsegment | nationkey | totalbal
-------+------------+-----------+----------
  1272 | AUTOMOBILE |        19 |  5856939
  1253 | FURNITURE  |        14 |  5794887
  1248 | FURNITURE  |         9 |  5784628
  1243 | FURNITURE  |        12 |  5757371
  1231 | HOUSEHOLD  |         3 |  5753216
  1251 | MACHINERY  |         2 |  5719140
  1247 | FURNITURE  |         8 |  5701952
```

```
(7 rows)
```

- Set operations

  Presto supports three set operations, namely `UNION`, `INTERSECT`, and `EXCEPT`. These clauses are used to combine the results of more than one query statement into a single result set. Use it as follows:

  ```
  query UNION [ALL | DISTINCT] query
  query INTERSECT [DISTINCT] query
  query EXCEPT [DISTINCT] query
  ```

  The `ALL` and `DISTINCT` arguments control which rows are included in the final result set. The default is `DISTINCT`.

  - `ALL`: Duplicated rows may be returned.
  - `DISTINCT`: Duplicated rows are eliminated.

  The `ALL` argument is not supported for `INTERSECT` or `EXCEPT`.

  The three set operations above are processed from left to right, with `INTERSECT` having the highest priority. This means `A UNION B INTERSECT C EXCEPT D` is the same as `A UNION (B INTERSECT C) EXCEPT D`.

- UNION

  `UNION` combines two query result sets and uses the `ALL` and `DISTINCT` arguments to control whether or not to remove duplicates.

  - Example one:

    ```
    SELECT 13
    UNION
    Select 42;
     _col0
    -------
        13
        42
    (2 rows)
    ```

  - Example two:

    ```
    SELECT 13
    UNION
    SELECT * FROM (VALUES 42, 13);
     _col0
    -------
        13
        42
    ```

```
(2 rows)
```

- **Example three:**

```
SELECT 13
UNION ALL
SELECT * FROM (VALUES 42, 13);
 _col0
-------
    13
    42
    13
(3 rows)
```

- **INTERSECT**

  `INTERSECT` **returns only the rows that are in both query result sets.**

  Examples

```
SELECT * FROM (VALUES 13, 42)
INTERSECT
SELECT 13;
 _col0
-------
   13
(1 row)
```

- **EXCEPT**

  `EXCEPT` **returns the rows that are in the result set of the first query, but not the second.**

```
SELECT * FROM (VALUES 13, 42)
EXCEPT
SELECT 13;
 _col0
-------
   42
(1 row)
```

- **ORDER BY clause**

  The `ORDER BY` **clause is used to sort a result set. The semantics are as follows:**

```
ORDER BY expression [ ASC | DESC ] [ NULLS { FIRST | LAST } ]
 [, ...]
```

  Where:

  - **Each** `expression` **may comprise output columns or it may be an ordinal number selecting an output column by position (starting at one).**
  - **The** `ORDER BY` **clause is the last step of a query after any** `GROUP BY` **or** `HAVING` **clause.**

- `NULLS { FIRST | LAST }` is used to control the sorting method of the `NULL` value (regardless of `ASC` or `DESC`). The default null ordering is `LAST`.

- **LIMIT clause**

  The `LIMIT` clause restricts the number of rows in the result set. `LIMIT ALL` is the same as omitting the `LIMIT` clause.

  Examples

  ```
  In this example, because the query lacks an ORDER BY, exactly which
  rows are returned is arbitrary.
  SELECT orderdate FROM orders LIMIT 5;
   orderdate
  -------------
   1996-04-14
   1992-01-15
   1995-02-01
   1995-11-12
   1992-04-26
  (5 rows)
  ```

- **TABLESAMPLE**

  Presto provides two sampling methods, namely `BERNOULLI` and `SYSTEM`. However, neither of them allow deterministic bounds on the number of rows returned.

  - `BERNOULLI`:

    Each row is selected to be in the table sample with a probability of the sample percentage. When a table is sampled using the Bernoulli method, all of its physical blocks are scanned and certain rows are skipped based on a comparison between the sample percentage and a random value calculated at runtime.

    The probability of a row being included in the result is independent from any other row. This does not reduce the time required to read the sampled table from the disk. It may have an impact on the total query time if the sampled output is processed further.

  - `SYSTEM`

    This sampling method divides the table into logical segments of data and samples the table at this granularity. This sampling method either selects all the rows from a particular segment of data or skips it (based on a comparison between the sample percentage and a random value calculated at runtime).

The rows selected in a system sampling is dependent on which connector is used . For example, when used with Hive, it is dependent on how the data is laid out in HDFS. This method does not guarantee independent sampling probabilities.

Examples

```
--- Using BERNOULLI sampling
SELECT *
FROM users TABLESAMPLE BERNOULLI (50);
--- Using system sampling
SELECT *
FROM users TABLESAMPLE SYSTEM (75);
Using sampling with joins:
--- Using sampling with JOIN
SELECT o.*, i. *
FROM orders o TABLESAMPLE SYSTEM (10)
JOIN lineitem i TABLESAMPLE BERNOULLI (40)
  ON o.orderkey = i.orderkey;
```

- **UNNEST**

`UNNEST` can be used to expand an array or map into a relation. Arrays are expanded into a single column, and maps are expanded into two columns (key, value). `UNNEST` can also be used with multiple arrays and maps, in which case they are expanded into multiple columns, with as many rows as the highest cardinality argument (the other columns are padded with nulls). A `WITH ORDINALITY` clause is an option for `UNNEST`. If it is implemented, an additional ordinal column is added to the end. `UNNEST` is normally used with a `JOIN` and can reference columns from relations on the left side of the join.

- Example one:

```
--- Using a single column
SELECT student, score
FROM tests
CROSS JOIN UNNEST(scores) AS t (score);
```

- Example two:

```
--- Using multiple columns
SELECT numbers, animals, n, a
FROM (
  VALUES
    (ARRAY[2, 5], ARRAY['dog', 'cat', 'bird']),
    (ARRAY[7, 8, 9], ARRAY['cow', 'pig'])
) AS x (numbers, animals)
CROSS JOIN UNNEST(numbers, animals) AS t (n, a);
  numbers  |     animals      |  n   |  a
-----------+------------------+------+------
 [2, 5]    | [dog, cat, bird] |    2 | dog
 [2, 5]    | [dog, cat, bird] |    5 | cat
 [2, 5]    | [dog, cat, bird] | NULL | bird
 [7, 8, 9] | [cow, pig]       |    7 | cow
 [7, 8, 9] | [cow, pig]       |    8 | pig
 [7, 8, 9] | [cow, pig]       |    9 | NULL
(6 rows)
```

- Example three:

```
--- Using a WITH ORDINALITY clause
SELECT numbers, n, a
FROM (
  VALUES
    (ARRAY[2, 5]),
    (ARRAY[7, 8, 9])
) AS x (numbers)
CROSS JOIN UNNEST(numbers) WITH ORDINALITY AS t (n, a);
  numbers   | n | a
-----------+---+---
 [2, 5]    | 2 | 1
 [2, 5]    | 5 | 2
 [7, 8, 9] | 7 | 1
 [7, 8, 9] | 8 | 2
 [7, 8, 9] | 9 | 3
(5 rows)
```

- Joins

Joins allow you to combine data from multiple relations. A `CROSS JOIN` returns the *Cartesian product* of two relations (all combinations). `CROSS JOIN` can be specified using either:

- the explicit `CROSS JOIN` syntax, or
- by specifying multiple relations in the `FROM` clause.

Both of the following queries are equivalent:

```
--- using the explicit **CROSS JOIN** syntax
SELECT *
FROM nation
CROSS JOIN region;
--- specifying multiple relations in the **FROM** clause
VALUES
FROM nation, region;
```

In this example, the nation table contains 25 rows and the region table contains 5, so a cross join between the two tables produces 125 rows:

```
SELECT n.name AS nation, r.name AS region
FROM nation AS n
CROSS JOIN region AS r
ORDER BY 1, 2;
     nation     |   region
----------------+-------------
 ALGERIA        | AFRICA
 ALGERIA        | AMERICA
 ALGERIA        | ASIA
 ALGERIA        | EUROPE
 ALGERIA        | MIDDLE EAST
 ARGENTINA      | AFRICA
 ARGENTINA      | AMERICA
...
(125 rows)
```

When two relations in a join have columns with the same name, the column references must be qualified using the relation name (or alias).

```
--- Correct
SELECT nation.name, region.name
FROM nation
CROSS JOIN region;
--- Correct
SELECT n.name, r.name
FROM nation AS n
CROSS JOIN region AS r;
--- Correct
SELECT n.name, r.name
FROM nation n
CROSS JOIN region r;
--- Wrong, it will raise the "Column 'name' is ambiguous" error
SELECT name
FROM nation
CROSS JOIN region;
```

- Subqueries

A subquery is an expression which is composed of a query. The subquery is correlated when it refers to columns outside of the subquery. Presto has limited support for correlated subqueries.

■ EXISTS

The `EXISTS` predicate determines if a subquery returns any rows. If a subquery returns rows, the `WHERE` expression is TRUE. Otherwise, the expression is FALSE.

Examples

```
SELECT name
FROM nation
WHERE EXISTS (SELECT * FROM region WHERE region.regionkey =
nation.regionkey);
```

■ IN

The IN predicate determines if any columns specified by `WHERE` are included in the result set produced by the subquery. It only returns results if columns are included in the result set. The subquery must produce exactly one column.

Examples

```
SELECT name
FROM nation
WHERE regionkey IN (SELECT regionkey FROM region);
```

- Scalar subquery

  A scalar subquery is a non-correlated subquery that returns either one row or none. The subquery cannot produce more than one row. If the subquery produces no rows, the returned value is NULL.

  Examples

```
SELECT name
FROM nation
WH ERregionkey = (SELECT max(regionkey) FROM regio;);
```

SET SESSION

- Synopsis

```
SET SESSION name = expression
SET SESSION catalog.name = expression
```

- Description

  Sets a session property value.

- Examples

```
SET SESSION optimize_hash_generation = true;
SET SESSION hive.optimized_reader_enabled = true;
```

SHOW CATALOGS

- Synopsis

```
SHOW CATALOGS [ LIKE pattern ]
```

- Description

  Lists the available catalogs. The `LIKE` clause can be used to filter the catalog names.

- **Examples**

```
SHOW CATALOGS;
```

SHOW COLUMNS

- **Synopsis**

```
SHOW COLUMNS FROM table
```

- **Description**

  Lists the columns in a given table along with their data type and other attributes.

- **Examples**

```
SHOW COLUMNS FROM orders;
```

SHOW CREATE TABLE

- **Synopsis**

```
SHOW CREATE TABLE table_name
```

- **Description**

  Shows the SQL statement that creates the specified table.

- **Examples**

```
SHOW CREATE TABLE sf1.orders;
-----------------------------------------
 CREATE TABLE tpch.sf1.orders (
    orderkey bigint,
    orderstatus varchar,
    totalprice double,
    orderdate varchar
 )
 WITH (
    format = 'ORC',
    partitioned_by = ARRAY['orderdate']
 )
(1 row)
```

SHOW CREATE VIEW

- **Synopsis**

```
SHOW CREATE VIEW view_name
```

- **Description**

  Shows the SQL statement that creates the specified view.

- **Examples**

```
SHOW CREATE VIEW view1;
```

SHOW FUNCTIONS

- **Synopsis**

```
SHOW FUNCTIONS
```

- **Description**

  List all the functions available for use in queries.
- **Examples**

```
SHOW FUNCTIONS
```

SHOW GRANTS

- **Synopsis**

```
SHOW GRANTS [ ON [ TABLE ] table_name ]
```

- **Description**

  Lists the grants for the current user on the specified table in the current catalog.
- **Examples**

```
--- List the grants for the current user on table orders
SHOW GRANTS ON TABLE orders;
--- List the grants for the current user on all the tables in all
schemas of the current catalog
SHOW GRANTS;
```

- **Limitations**

  Some connectors do not support `SHOW GRANTS`.

SHOW SCHEMAS

- **Synopsis**

```
SHOW SCHEMAS [ FROM catalog ] [ LIKE pattern ]
```

- **Description**

  Lists all schemas in the specified catalog, or if no catalog has been specified, in the current catalog. The `LIKE` clause can be used to filter the schema names.

- Examples

  ```
  SHOW SCHEMAS;
  ```

SHOW SESSION

- Synopsis

  ```
  SHOW SESSION
  ```

- Description

  Lists the current session properties.

- Examples

  ```
  SHOW SESSION
  ```

SHOW TABLES

- Synopsis

  ```
  SHOW TABLES [ FROM schema ] [ LIKE pattern ]
  ```

- Description

  Lists all tables in the specified schema, or if no schema has been specified, in the current schema. The `LIKE` clause can be used to filter the table name.

- Examples

  ```
  SHOW TABLES;
  ```

START TRANSACTION

- Synopsis

  ```
  START TRANSACTION [ mode [, ...] ]
  where **mode** is one of:
  ISOLATION LEVEL { READ UNCOMMITTED | READ COMMITTED | REPEATABLE
  READ | SERIALIZABLE }
  READ { ONLY | WRITE }
  ```

- Description

  Starts a new transaction for the current session.

- Examples

  ```
  START TRANSACTION;
  START TRANSACTION ISOLATION LEVEL REPEATABLE READ;
  START TRANSACTION READ WRITE;
  START TRANSACTION ISOLATION LEVEL READ COMMITTED, READ ONLY;
  ```

```
START TRANSACTION READ WRITE, ISOLATION LEVEL SERIALIZABLE;
```

USE

- **Synopsis**

```
USE catalog.schema
USE schema
```

- **Description**

Updates the session to use the specified catalog and schema. If a catalog is not specified, the schema is resolved relative to the current catalog.

- **Examples**

```
USE hive.finance;
USE information_schema;
```

VALUES

- **Synopsis**

```
VALUES row [, ...]
where **row** is a single expression or
( column_expression [, ...] )
```

- **Description**

Defines a literal inline table.

- `VALUE` can be used anywhere a query can be used. For example, behind the `FROM` clause of a `SELECT`, in an `INSERT`, or even at the top level.
- `VALUE` creates an anonymous table without column names by default. The table and columns can be named using an `AS` clause.

- **Examples**

```
--- Return a table with one column and three rows
VALUES 1, 2, 3
--- Return a table with two columns and three rows
VALUES
    (1, 'a'),
    (2, 'b'),
    (3, 'c')
--- Using in a query statement:
SELECT * FROM (
    VALUES
        (1, 'a'),
        (2, 'b'),
        (3, 'c')
) AS t (id, name)
--- Create a table
CREATE TABLE example AS
SELECT * FROM (
    VALUES
        (1, 'a'),
        (2, 'b'),
        (3, 'c')
) AS t (id, name)
```

9.8.7 Technical support

This section provides details on technical support.

If you have any questions, please contact our technical support:

- *Submit a ticket*

9.9 TensorFlow

TensorFlow is supported by E-MapReduce 3.13.0 and later. You can add TensorFlow from the available services in your software configurations. If you are using TensorFlow in E-MapReduce to perform high-performance computing, you can allocate CPU and GPU resources through YARN.

Prerequisites

- On the software side, an E-MapReduce cluster installs TensorFlow and a TensorFlow on YARN (TOY) toolkit.
- On the hardware side, E-MapReduce supports computing using both CPU and GPU resources. If you need to use GPU computing, you can choose ECS instances from compute optimized families with GPU, such as gn5 and gn6, for the core and task nodes in the cluster. Compute optimized families with GPU support heterogeneous computing. After determining the instance type, choose the CUDA toolkit and cuDNN versions as required.

Submit TensorFlow jobs

You can log on to the master node in the E-MapReduce cluster to submit TensorFlow jobs using the command line. For example:

```
el_submit [-h] [-t APP_TYPE] [-a APP_NAME] [-m MODE] [-m_arg MODE_ARG]
[-interact INTERACT] [-x EXIT]
[-enable_tensorboard ENABLE_TENSORBOARD]
[-log_tensorboard LOG_TENSORBOARD] [-conf CONF] [-f FILES]
[-pn PS_NUM] [-pc PS_CPU] [-pm PS_MEMORY] [-wn WORKER_NUM]
[-wc WORKER_CPU] [-wg WORKER_GPU] [-wm WORKER_MEMORY]
```

```
[-wnpg WNPG] [-ppn PPN] [-c COMMAND [COMMAND ...]]
```

The basic parameters are described as follows:

- -t APP_TYPE: Specifies the type of task to be submitted. The supported types are tensorflow-ps, tensorflow-mpi, and standalone. They are used in conjunction with the following –m MODE parameter.
  - tensorflow-ps: Uses a parameter server for the communication of data, which is the PS mode of native TensorFlow.
  - tensorflow-mpi: Uses Horovod, an open source framework from Uber, which relies on message passing interface (MPI) primitives for the communication of data.
  - standalone: Users assign tasks to one instance in the YARN cluster for execution . This is similar to standalone execution.
- –a APP_NAME: Specifies the name of the submitted TensorFlow job. You can name jobs as required.
- –m MODE: Specifies the runtime environment for submitted TensorFlow jobs. E-MapReduce supports the following environments: local, virtual-env, and docker.
  - local: Uses Python runtime environments set up in the E-MapReduce worker nodes. If you want to use third-party Python packages, you need to install the packages on all the nodes manually.
  - docker: Uses the Docker containers installed on the E-MapReduce worker nodes . TensorFlow runs in Docker containers.
  - virtual-env: Uses isolated Python environments created by users. You can install Python libraries in Python environments. These libraries can be different from those installed in the environments that are set up in the worker nodes.
- -m_arg MODE_ARG: Specifies the supplemental parameter for the –m MODE. If the runtime environment is docker, set the value to the docker image name. If the runtime environment is virtual-env, set the value to the name of Python environment tar.gz file.
- –x Exit: You need to exit the parameter servers manually for certain distributed TensorFlow APIs. To exit parameter servers automatically when worker servers finish training their models, specify the -x option.
- -enable_tensorboard: Specifies whether to enable TensorBoard when TensorFlow starts training models.

- -log_tensorboard: Specifies the location of TensorBoard logs in HDFS. If TensorBoard is enabled when TensorFlow starts training models, this parameter is required.
- -conf CONF: Specifies the location of the Hadoop configuration. Setting the value is optional. The default E-MapReduce configuration is used.
- –f FILES: Specifies all dependent files and folders for TensorFlow to run, including executable scripts. If virtual-env files that are executed in a virtual environmen t are specified, you can put all dependencies in one folder. The script then automatically uploads the folders into HDFS according to the folder hierarchy.
- -pn TensorFlow: Specifies the number of parameter servers to start.
- -pc: Specifies the number of CPU cores that each parameter server requests.
- -pm: Specifies the memory size that each parameter server requests.
- -wn: Specifies the number of worker nodes started by TensorFlow.
- -wc: Specifies the number of CPU cores that each worker requests.
- -wg: Specifies the number of GPU cores that each worker requests.
- -wm: Specifies the memory usage that each worker requests.
- -c COMMAND: Specifies the command to run. For example, pythoncensus.py.

Advanced options. We recommend that you use advanced options with care, as they may result in job failures.

- -wnpg: Specifies the number of workers that use a GPU simultaneously (for tensorflow-ps).
- -ppn: Specifies the number of workers that use a GPU simultaneously (for Horovod). The preceding options refer to multitasking on a single GPU. Thresholds should be set to avoid GPU running out of memory.

9.10 Knox

E-MapReduce supports *Apache Knox*. If you select a Knox-supported image to create a cluster, you can access the Web UI from the public network to use services such as YARN, HDFS, and SparkHistory.

Preparations

- Enable Knox access using a public IP address

1. The service port of Knox on E-MapReduce is 8443. In the cluster details, find the ECS security group in which the cluster is located.
2. Change the corresponding security group in the ECS console and add a rule in Internet inbound to enable port 8443.

> Notice:
>
> - For security reasons, the authorization object must be your limited IP address range. 0.0.0.0/0 is forbidden.
> - After port 8443 of the security group is enabled, all nodes (including non-E-MapReduce ECS nodes) in the security group enable port 8443 at the ingress of the public network.

- Set a Knox user

  Accessing Knox requires a user name and password for authentication. The authentication is based on LDAP. You can use your own LDAP service or the LDAP service of Apache Directory Server in the cluster.

  - Use the LDAP service in the cluster

    Method one (recommended):

    Add a Knox account in the *User Management* page.

    Method two:

    1. Log on to the cluster through SSH. For more information, see *Connect to clusters using SSH*.
    2. Prepare your user data. Here, Tom is used as the user name. In the file, replace all `emr-guest` with `Tom` and `cn:EMR GUEST` with `cn:Tom`, and set `userPassword` to your password.

       ```
       su knox
       cd /usr/lib/knox-current/templates
       vi users.ldif
       ```

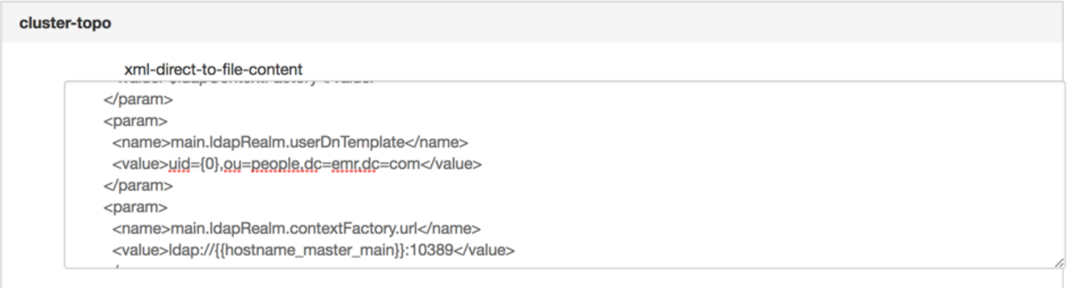
       > Notice:
       >
       > For security reasons, before you export your user data to LDAP, change the password of users.ldif by changing `userPassword` to your password.

3. Export to LDAP.

   ```
   su knox
   cd /usr/lib/knox-current/templates
   sh ldap-sample-users.sh
   ```

- Use your own LDAP service

1. Enter the cluster configuration management page. In the cluster-topo configuration, set `main.ldapRealm.userDnTemplate` to your user DN template and `main.ldapRealm.contextFactory.url` to your LDAP server domain name and port. Then, save the settings and restart Knox.



cluster-topo

xml-direct-to-file-content

```
</param>
<param>
  <name>main.ldapRealm.userDnTemplate</name>
  <value>uid={0},ou=people,dc=emr,dc=com</value>
</param>
<param>
  <name>main.ldapRealm.contextFactory.url</name>
  <value>ldap://{{hostname_master_main}}:10389</value>
```

2. Your LDAP service does not typically run in the cluster. You must enable the Knox port to access the LDAP service in the public network, such as port 10389. For more information, see the preceding steps for enabling port 8443. Then, select Internet outbound.

> **Notice:**
>
> For security reasons, the authorization object must be the public IP address of your Knox cluster. 0.0.0.0/0** is forbidden.

Access Knox

- Access using the E-MapReduce shortcut link

  1. Log on to the *E-MapReduce console*.
  2. Click the ID link of the target cluster.
  3. In the navigation pane on the left, click Clusters and Services.
  4. Click the relevant services on the E-MapReduce services page, such as HDFS and YARN.
  5. In the upper-right corner, click Quick Link.

- Access using the public IP address of the cluster

  1. Check the public IP address in the cluster details.

2. Access the URLs of the relevant services in the browser.

   - HDFS UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/hdfs/*
   - YARN UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/yarn/*
   - SparkHistory UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/sparkhistory/*
   - Ganglia UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/ganglia/*
   - Storm UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/storm/*
   - Oozie UI: *https://{cluster_access_ip}:8443/gateway/cluster-topo/oozie/*
3. website is not security is displayed in your browser because the Knox service uses a self-signed certificate. Confirm that the accessed IP address is the same as that of your cluster and the port is 8443. Click advance > continue.
4. Enter the user name and password set in LDAP in the logon dialog box.

Access control lists

Knox provides service-level permission management to limit service access to specific users, user groups, or IP addresses. See *Apache Knox Authorization*.

- Example
  - Scenario: The YARN UI only allows access by user Tom.
  - Steps: Enter the cluster configuration management page. In the cluster-topo configuration, add access control list (ACL) code between the `<gateway>...</gateway>` labels.

```
<provider>
      <role>authorization</role>
      <name>AclsAuthz</name>
      <enabled>true</enabled>
      <param>
          <name>YARNUI.acl</name>
          <value>Tom;*;*</value>
      </param>
</provider>
```

- Notes

  Knox provides RESTful APIs for operating a range of services, including adding or deleting HDFS files. For security reasons, make sure that when you enable port 8443 of the security group in the ECS console, the authorization object is your limited IP address range. 0.0.0.0/0 is forbidden. Do not use the LDAP user name and password in the Knox installation directory to access Knox.

9.11 Instructions for using Flume

E-MapReduce version 3.16.0 and later support Apache Flume. This topic describes how to use Flume to synchronize E-MapReduce Kafka cluster data to HDFS, Hive, and HBase running on E-MapReduce Hadoop clusters, and to Alibaba Cloud OSS.

Prerequisites

- You must have selected Flume in the Optional Service menu when you created a Hadoop cluster.
- You must have created a Kafka cluster and created a topic named flume-test to generate data.

> Note:
>
> - If you have created a high security mode Hadoop cluster to consume standard Kafka cluster data, and you need to configure Kerberos authentication on the Hadoop cluster, see *Authentication method compatible with MIT Kerberos*.
> - If you have created a high security mode Kafka cluster and you need to write data to a standard Hadoop cluster using Flume, see *Kerberos Kafka Source in this topic*.
> - If you have created a high security mode Hadoop cluster and a high security mode Kafka cluster, and you need to configure Kerberos, see *Cross-region access* and *Cross-region access using Flume*.

Kafka->HDFS

- Configure Flume

  Create a configuration file named *flume.properties*, and add the following configurations for the file, where `a1.sources.source1.kafka.bootstrap.servers` indicates the host and port for a Kafka broker, `a1.sources.source1.kafka.topics` indicates the Kafka topic where Flume is used to consume data, and `a1.sinks.k1.hdfs.path` indicates the path where Flume writes data to HDFS.

```
a1.sources = source1
a1.sinks = k1
a1.channels = c1

a1.sources.source1.type = org.apache.flume.source.kafka.KafkaSource
a1.sources.source1.channels = c1
a1.sources.source1.kafka.bootstrap.servers = kafka-host1:port1,kafka
-host2:port2...
a1.sources.source1.kafka.topics = flume-test
a1.sources.source1.kafka.consumer.group.id = flume-test-group
```

```
# Describe the sink
a1.sinks.k1.type = hdfs
a1.sinks.k1.hdfs.path = /tmp/flume/test-data
a1.sinks.k1.hdfs.fileType=DataStream

# Use a channel which buffers events in memory
a1.channels.c1.type = memory
a1.channels.c1.capacity = 100
a1.channels.c1.transactionCapacity = 100

# Bind the source and sink to the channel
a1.sources.source1.channels = c1
a1.sinks.k1.channel = c1
```

- Start Flume

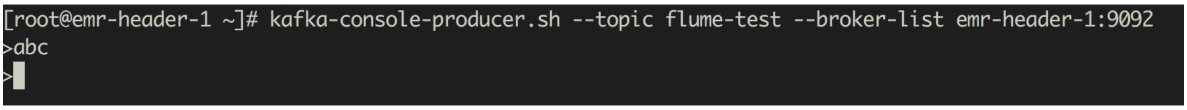
  Flume's default configuration file is stored in `/etc/ecm/flume-conf`. Use the following configuration file to start a Flume agent.

  ```
  flume-ng agent --name a1 --conf /etc/ecm/flume-conf  --conf-file
  flume.properties
  ```

  After the agent is started, the log logs/flume.log will be generated in the current path due to *log4j.properties* being in */etc/ecm/flume-conf*. You can configure *log4j.properties* according to your requirements.

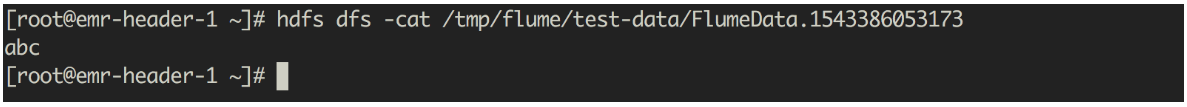
- Test

  Use the kafka-console-producer.sh command and input the test data abc in your Kafka cluster.

  ```
  [root@emr-header-1 ~]# kafka-console-producer.sh --topic flume-test --broker-list emr-header-1:9092
  >abc
  >
  ```

  Flume generates a file FlumeData.xxx with a timestamp (in milliseconds) suffix based on the current time. When you view the file content, you can see the data that you input in Kafka.

  ```
  [root@emr-header-1 ~]# hdfs dfs -cat /tmp/flume/test-data/FlumeData.1543386053173
  abc
  [root@emr-header-1 ~]#
  ```

Kafka->Hive

- Create a Hive table

  Before Flume writes data into Hive using transactions, you need to set the `transactional` property when creating a Hive table. The following example shows how to create a table named flume_test table.

  ```
  create table flume_test (id int, content string)
  ```

```
clustered by (id) into 2 buckets
stored as orc  TBLPROPERTIES('transactional'='true');
```

- **Configure Flume**

  Create a configuration file flume.properties and add the following configurations for the file, where `a1.sources.source1.kafka.bootstrap.servers` indicates the host and port for a Kafka broker and `a1.sinks.k1.hive.metastore` indicates a Hive metastore URI. Then, configure the value of `hive.metastore.uris` in the *`hive-site.xml`* file:

```
a1.sources = source1
a1.sinks = k1
a1.channels = c1

a1.sources.source1.type = org.apache.flume.source.kafka.KafkaSource
a1.sources.source1.channels = c1
a1.sources.source1.kafka.bootstrap.servers = kafka-host1:port1,kafka
-host2:port2...
a1.sources.source1.kafka.topics = flume-test
a1.sources.source1.kafka.consumer.group.id = flume-test-group

# Describe the sink
a1.sinks.k1.type = hive
a1.sinks.k1.hive.metastore = thrift://xxxx:9083
a1.sinks.k1.hive.database = default
a1.sinks.k1.hive.table = flume_test
a1.sinks.k1.serializer = DELIMITED
a1.sinks.k1.serializer.delimiter = ","
a1.sinks.k1.serializer.serdeSeparator = ','
a1.sinks.k1.serializer.fieldnames =id,content

a1.channels.c1.type = memory
a1.channels.c1.capacity = 100
a1.channels.c1.transactionCapacity = 100

a1.sources.source1.channels = c1
a1.sinks.k1.channel = c1
```

- **Start Flume**

```
flume-ng agent --name a1 --conf /etc/ecm/flume-conf  --conf-file
flume.properties
```

- **Generate data**

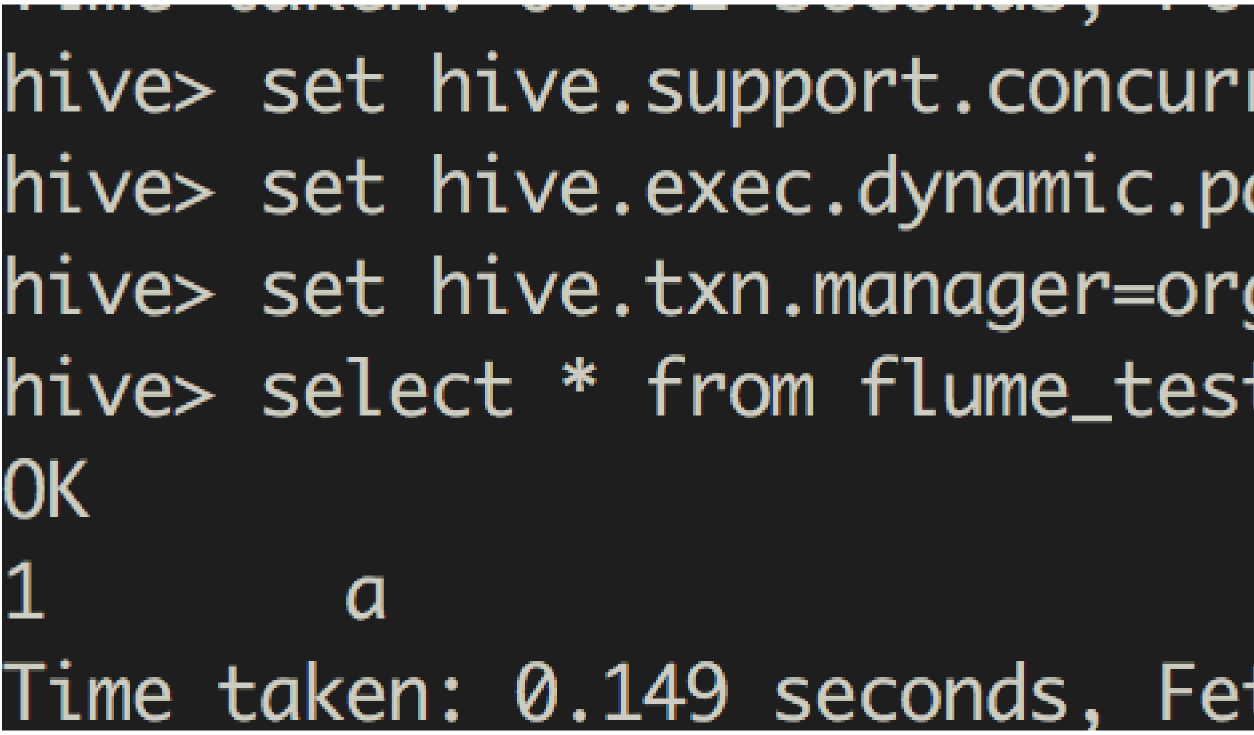
  Use the *`kafka-console-producer.sh`* command and input the comma-separated test data 1,a in your Kafka cluster.

- **Verify the input data**

  Note that quering Hive transaction tables require configuration on the Hive client:

```
hive.support.concurrency - true
hive.exec.dynamic.partition.mode - nonstrict
hive.txn.manager - org.apache.hadoop.hive.ql.lockmgr.DbTxnManager
```

After the preceding configurations are set, you can query data in the flume_test table.

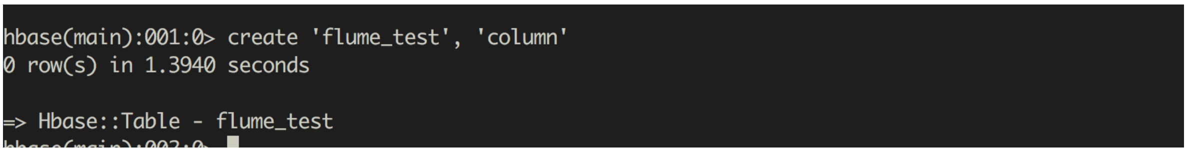


```
hive> set hive.support.concurr
hive> set hive.exec.dynamic.p
hive> set hive.txn.manager=org
hive> select * from flume_test
OK
1       a
Time taken: 0.149 seconds, Fe
```

Kafka->HBase

- Create a HBase table

  Create a HBase table flume_test and a column family column.



```
hbase(main):001:0> create 'flume_test', 'column'
0 row(s) in 1.3940 seconds

=> Hbase::Table - flume_test
hbase(main):002:0>
```

- Configure Flume

  Create a configuration file *flume.properties* and add the following configurations, where `a1.sources.source1.kafka.bootstrap.servers` indicates the host and port for a Kafka broker, `a1.sinks.k1.table` indicates the name of the HBase table, and `a1.sinks.k1.columnFamily` indicates the name of the column family:

```
a1.sources = source1
a1.sinks = k1
a1.channels = c1

a1.sources.source1.type = org.apache.flume.source.kafka.KafkaSource
a1.sources.source1.channels = c1
a1.sources.source1.kafka.bootstrap.servers = kafka-host1:port1,kafka
-host2:port2...
a1.sources.source1.kafka.topics = flume-test
```

```
a1.sources.source1.kafka.consumer.group.id = flume-test-group

a1.sinks.k1.type = hbase
a1.sinks.k1.table = flume_test
a1.sinks.k1.columnFamily = column


# Use a channel which buffers events in memory
a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

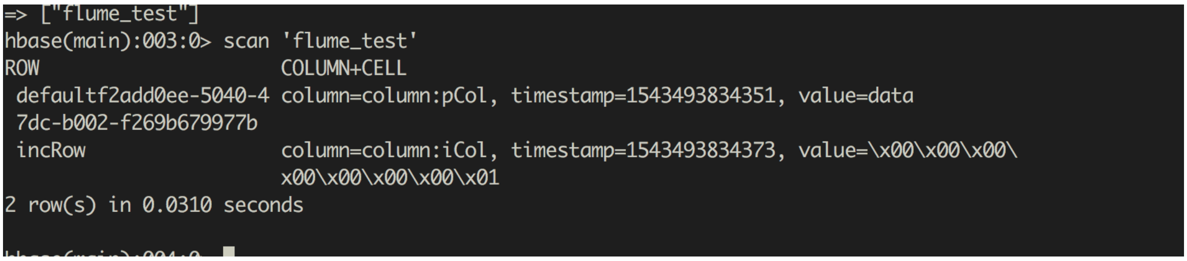
# Bind the source and sink to the channel
a1.sources.source1.channels = c1
a1.sinks.k1.channel = c1
```

- Start Flume

  ```
  flume-ng agent --name a1 --conf /etc/ecm/flume-conf  --conf-file
  flume.properties
  ```

- Test

  After data is generated using kafka-console-producer.sh in your Kafka cluster, you can query data in HBase.

  ```
  => ["flume_test"]
  hbase(main):003:0> scan 'flume_test'
  ROW                        COLUMN+CELL
   defaultf2add0ee-5040-4 column=column:pCol, timestamp=1543493834351, value=data
   7dc-b002-f269b679977b
   incRow                    column=column:iCol, timestamp=1543493834373, value=\x00\x00\x00\
                             x00\x00\x00\x00\x01
  2 row(s) in 0.0310 seconds

  hbase(main):004:0>
  ```

Kafka->OSS

- Create an OSS path

  Create an OSS bucket and directory, such as *oss://flume-test/result*.

- Configure Flume

  Flume requires a large amount of JVM memory when writing data to OSS. To resolve this issue, you can:

  - Reduce the OSS cache size

    Copy the file hdfs-site.xml from */etc/ecm/hadoop-conf* to */etc/ecm/flume-conf*, and reduce the value of the configuration term `smartdata.cache.buffer.size`, for example, to 1048576.

  - Increase the Flume agent's heap size (Xmx)

In the Flume configuration path */etc/ecm/flume-conf*, copy configuration file *flume-env.sh.template*, paste it to the /etc/ecm/flume-conf path , rename it *flume-env.sh*, and set Xmx, for example, to 1G:

```
export JAVA_OPTS="-Xmx1g"
```

Create a configuration file *flume.properties* and add the following configurations, where *a1.sources.source1.kafka.bootstrap.servers* indicates the host and port for a Kafka broker, and *a1.sinks.k1.hdfs.path* indicates an OSS path:

```
a1.sources = source1
a1.sinks = k1
a1.channels = c1

a1.sources.source1.type = org.apache.flume.source.kafka.KafkaSource
a1.sources.source1.channels = c1
a1.sources.source1.kafka.bootstrap.servers = kafka-host1:port1,kafka
-host2:port2...
a1.sources.source1.kafka.topics = flume-test
a1.sources.source1.kafka.consumer.group.id = flume-test-group

a1.sinks.k1.type = hdfs
a1.sinks.k1.hdfs.path = oss://flume-test/result
a1.sinks.k1.hdfs.fileType=DataStream

# Use a channel which buffers events in memory
a1.channels.c1.type = memory
a1.channels.c1.capacity = 100
a1.channels.c1.transactionCapacity = 100

# Bind the source and sink to the channel
a1.sources.source1.channels = c1
a1.sinks.k1.channel = c1
```

- Start Flume

  If you modified the OSS cache size when configuring Flume, use the classpath parameter to pass OSS-related dependencies and configurations to Flume:

```
flume-ng agent --name a1 --conf /etc/ecm/flume-conf  --conf-file
flume.properties  --classpath "/opt/apps/extra-jars/*:/etc/ecm/flume
-conf/hdfs-site.xml"
```

  If you modified the Flume agent's Xmx, you only need to pass OSS-related dependencies:

```
flume-ng agent --name a1 --conf /etc/ecm/flume-conf  --conf-file
flume.properties  --classpath "/opt/apps/extra-jars/*"
```

- Test

After data is generated using `kafka-console-producer.sh` in your Kafka cluster, in the OSS path `oss://flume-test/result`, a file `FlumeData.xxxx` is generated with a timestamp (in milliseconds) suffix based on the current time.

Kerberos Kafka source

If high security Kafka cluster data is consumed, you must configure the following variables:

- In your Kafka cluster, configure Kerberos authentication and copy the generated keytab file `test.keytab` to the Hadoop cluster path `/etc/ecm/flume-conf`, and copy the Kafka cluster file `/etc/ecm/has-conf/krb5.conf` to the Hadoop cluster path /etc/ecm/flume-conf. For more information, see *Authentication method compatible with MIT Kerberos*.
- Configure `flume.properties`

  In the `flume.properties` file, add the following configurations:

  ```
  a1.sources.source1.kafka.consumer.security.protocol = SASL_PLAINTEXT
  a1.sources.source1.kafka.consumer.sasl.mechanism = GSSAPI
  a1.sources.source1.kafka.consumer.sasl.kerberos.service.name = kafka
  ```
- Configure the Kafka client
  - In `/etc/ecm/flume-conf`, create the file `flume\_jaas.conf` with the following content:

    ```
    KafkaClient {
      com.sun.security.auth.module.Krb5LoginModule required
      useKeyTab=true
      storeKey=true
      keyTab="/etc/ecm/flume-conf/test.keytab"
      serviceName="kafka"
      principal="test@EMR.${realm}. COM";
    };
    ```

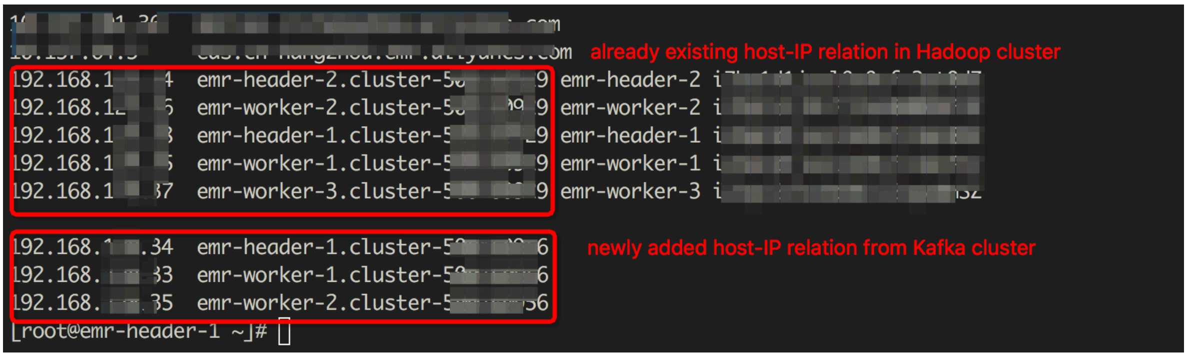
    where, `${realm}` is replaced with a Kerberos realm of your Kafka cluster. To obtain a Kerberos realm, in your Kafka cluster, run the command `hostname` to obtain a hostname in the form of `emr-header-1.cluster-xxx`, such as `mr-header-1.cluster-123456`. The last numeric string 123456 is a Kerberos realm.
  - Change `/etc/ecm/flume-conf/flume-env.sh`

By default, in */etc/ecm/flume-conf/*, the file *flume-env.sh* does not exist. You need to copy *flume-env.sh.template*, paste it to /etc/ecm/flume-conf/, rename it *flume-env.sh*, and add the following content:

```
export JAVA_OPTS="$JAVA_OPTS -Djava.security.krb5.conf=/etc/ecm/
flume-conf/krb5.conf"
export JAVA_OPTS="$JAVA_OPTS -Djava.security.auth.login.config=/
etc/ecm/flume-conf/flume_jaas.conf"
```

- Set domain name

  Add domain names and IP binding information of each Kafka cluster node to */etc/hosts* of the Hadoop cluster. The form of the long domain name is *emr-header-1.cluster-123456*.



```
[redacted].com
[redacted]com
192.168.1[redacted]4   emr-header-2.cluster-5[redacted]9 emr-header-2 i[redacted]
192.168.1[redacted]6   emr-worker-2.cluster-5[redacted]9 emr-worker-2 i[redacted]
192.168.1[redacted]3   emr-header-1.cluster-5[redacted]9 emr-header-1 i[redacted]
192.168.1[redacted]5   emr-worker-1.cluster-5[redacted]9 emr-worker-1 i[redacted]
192.168.1[redacted]37  emr-worker-3.cluster-5[redacted]9 emr-worker-3 i[redacted]

192.168.[redacted].34  emr-header-1.cluster-5[redacted]6
192.168.[redacted]33   emr-worker-1.cluster-5[redacted]6
192.168.[redacted]35   emr-worker-2.cluster-5[redacted]56
[root@emr-header-1 ~]#
```

already existing host-IP relation in Hadoop cluster

newly added host-IP relation from Kafka cluster

Cross-region access using Flume

After cross-region access is configured, you need to set other configurations as follows:

- In your Kafka cluster, configure Kerberos authentication and copy the generated keytab file *test.keytab* to the Hadoop cluster path */etc/ecm/flume-conf*. For more information, see *Authentication method compatible with MIT Kerberos*.
- Configure *flume.properties*

  In the *flume.properties* file, add the following configurations:

```
a1.sources.source1.kafka.consumer.security.protocol = SASL_PLAINTEXT
a1.sources.source1.kafka.consumer.sasl.mechanism = GSSAPI
a1.sources.source1.kafka.consumer.sasl.kerberos.service.name = kafka
```

- Configure the Kafka client
  - In */etc/ecm/flume-conf*, create the file *flume_jaas.conf* with the following content:

```
KafkaClient {
  com.sun.security.auth.module.Krb5LoginModule required
  useKeyTab=true
  storeKey=true
  keyTab="/etc/ecm/flume-conf/test.keytab"
  serviceName="kafka"
  principal="test@EMR.${realm}. COM";
};
```

where, `${realm}` is replaced with a Kerberos realm of your Kafka cluster. To obtain a Kerberos realm, in your Kafka cluster, run the command `hostname` to obtain a hostname in the form of *emr-header-1.cluster-xxx*, such as *emr-header-1.cluster-123456*. The last numeric string 123456 is a Kerberos realm.

- Change */etc/ecm/flume-conf/flume-env.sh*

  By default, in */etc/ecm/flume-conf/*, the file *flume-env.sh* does not exist. You need to copy *flume-env.sh.template*, paste it to */etc/ecm/flume-conf/*, rename it flume-env.sh, and add the following content:

```
export JAVA_OPTS="$JAVA_OPTS -Djava.security.auth.login.config=/
etc/ecm/flume-conf/flume_jaas.conf""
```

10 OSS ACL

This article introduces how to use RAM Access control to isolate the data of different sub-account.

Procedure

E-MapReduce supports using RAM to isolate the data of different sub-accounts. To do so, complete the following steps:

1. Log on to the *Alibaba Cloud RAM console*.
2. Create a sub-account in RAM. For more information, see *Create a RAM user*.
3. In the navigation pane on the left, click Policies to enter the policy management page.
4. Click Custom Policy.
5. In the upper-right corner, click Create Authorization Policy. Follow the steps there to create a policy. You can create as many policies as the number of authorization control sets that you need.

   In this example, it is assumed that you need the following two sets of data control policies:

   - Bucketname of the testing environment: test-bucket. The corresponding policy is as follows:

     ```
     {
     "Version": "1",
     "Statement": [
     {
     "Effect": "Allow",
     "Action": [
       "oss:ListBuckets"
     ],
     "Resource": [
       "acs:oss:*:*:*"
     ]
     },
     {
     "Effect": "Allow",
     "Action": [
       "oss:Listobjects",
       "oss:GetObject",
       "oss:PutObject",
       "oss:DeleteObject"
     ],
     "Resource": [
       "acs:oss:*:*:test-bucket",
       "acs:oss:*:*:test-bucket/*"
     ]
     ```

```
}
]
}
```

- Bucketname of the production environment: prod-bucket. The corresponding policy is as follows:

```
{
"Version": "1",
"Statement": [
{
"Effect": "Allow",
"Action": [
  "oss:ListBuckets"
],
"Resource": [
  "acs:oss:*:*:*"
]
},
{
"Effect": "Allow",
"Action": [
  "oss:Listobjects",
  "oss:GetObject",
  "oss:PutObject"
],
"Resource": [
  "acs:oss:*:*:prod-bucket",
  "acs:oss:*:*:prod-bucket/*"
]
}
]
}
```

6. Click Users.
7. Find the sub-account item which the policy is given to and click Manage.
8. In the navigation pane on the left, click User Authorization Policy.
9. In the upper-right corner, click Edit Authorization Policy.
10. Select and add authorization policies.
11. Click OK to complete the policy authorization of the sub-account.
12. In the upper-right corner, click User Details to enter the user details page of the sub-account.
13. Click Start Console Logon in the console logon management bar to initialize the authorization of the sub-account logon console.

Complete and use

Once you have completed all of the preceding steps, you can use the sub-account to log on to E-MapReduce. Note the following limitations:

- All buckets are displayed in the OSS interface for clusters, operations, and plan executions. However, you can only enter the authorized bucket.
- Only the content under the authorized bucket is displayed.
- The authorized bucket can only be read and written. Otherwise, an error is reported.

11 Connect to clusters using SSH

If your existing jobs and execution plans do not meet your complex application requirements, you can log on to the master node of the E-MapReduce cluster and navigate to the cluster details page, which contains the public network IP address of the cluster's master node. You can then log on to the master node through SSH to view various settings and states.

Common variables

Some relevant environment variables have been set on the machine, including the following common ones:

- JAVA_HOME
- HADOOP_HOME
- HADOOP_CONF_DIR
- HADOOP_LOG_DIR
- YARN_LOG_DIR
- HIVE_HOME
- HIVE_CONF_DIR
- PIG_HOME
- PIG_CONF_DIR

You can quote these variables in scripts. However, to avoid unexpected E-MapReduce errors, we recommend that you do not change them.

Connect to the master node

1. Enter the following command for SSH to log on to the master node. You can obtain the public network IP of the cluster's master node in the host information column on the *Cluster details* page.

   ```
   ssh root@ip.of.master
   ```

2. Enter the password set during creation.

Connect to a cluster using SSH without a password

To be able to manage or perform operations on a cluster, you first need to log on. To connect to the master node, you can use the passwordless SSH logon on the master machine. By default, the master node opens up the public network IP. The procedure is as follows:

1. Connect to the master node in the root and password mode.
2. Change to a Hadoop or HDFS user.

## SSH mode of Linux

1. Copy the private key to the local machine.

   ```
   sz ~/.ssh/id_rsa
   ```

2. Return to your local machine and attempt to connect to the master again.

   ```
   ssh -i private_key_path/id_rsa hadoop@server_ip_address
   ```

   If only one private key exists, put it in `~/.ssh/`. By default, this private key does not need to be specified by `-i`.

Connect to the master node using SSH on Windows

There are two ways in Windows of connecting to the master node through SSH without a password:

- Using PuTTY

  1. Click to enter the *PuTTY download* page.
  2. Download PuTTYgen.
  3. Open PuTTYgen and load your private key.

     Notice:
     Keep the private key safe. In the event of accidental disclosure, generate a new private key immediately.

  4. Use the default configuration and save the private key. Obtain a secret PuTTY key file with the .ppk suffix.
  5. Operate PuTTY and select Session on the configuration page.
  6. Enter the public network IP address of the target machine and add the user name (for example, hadoop@MasterNodeIP).
  7. Select Connetion > SSH > Auth on the configuration page.
  8. Select the generated PPK file.
  9. Click Open to log on to the master node.
- Using Cygwin | MinGW

This convenient tool simulates the Linux environment in Windows.

To use this method, see the aforementioned SSH method of Linux.

The MinGW method is recommended because it is the most compact. If the official website cannot be opened, download a Git client. The default Git Bash can be used.

View the web UI of Hadoop, Spark, Ganglia, and other systems

> Note:
>
> You must have finished *the passwordless SSH logon* before performing this step.

For safety reasons, the ports of the web UI monitoring system for Hadoop, Spark, Ganglia, and other systems in the E-MapReduce cluster are not open to the public. If you want to visit these web UIs, you must build an SSH tunnel for port forwarding. The following two methods are available:

> Notice:
>
> The following operations should be completed on your local machine, and not on the machine in the cluster.

- Dynamic port forwarding

  Create a SSH tunnel that can connect certain dynamic port connections between the local and master machines in the E-MapReduce cluster.

  ```
  ssh -i /path/id_xxx -ND 8157 hadoop@masterNodeIP
  ```

  8157 is any port not used in the local machine. You can customize this.

  After dynamic forwarding, you can view the following:

  - Recommended methods

    We recommend that you use Chrome as your browser. View the web UI as follows:

    ```
    chrome --proxy-server="socks5://localhost:8157" --host-resolver-
    rules="MAP * 0.0.0.0 , EXCLUDE localhost" --user-data-dir=/tmp/
    ```

    In Windows, the *tmppath* can be written into a similar *d:/tmppath*. In Linux or OSX, */tmp/* can be written directly.

    The location of Chrome depends on the operating system, as shown in the following table.

Operating system	Chrome location
Mac OS X	/Applications/Google Chrome.app/Contents/MacOS /Google Chrome
Linux	/usr/bin/google-chrome
Windows	C:\Program Files (x86)\Google\Chrome\Application\ chrome.exe

- Plug-in method

  If your local machine is connected to the master machine of the E-MapReduce cluster through an SSH tunnel, you need to configure a local agent to view the web UI of Hadoop, Spark, or Ganglia in your browser. Complete the following steps:

  1. For Chrome or Firefox, click here to *download the FoxyProxy Standard* software.
  2. After you have installed this and restarted your browser, open a text editor and edit the following content:

     ```
     <?<!--?--> xml version="1.0" encoding="UTF-8"? >
     <foxyproxy>
     <proxies>
     <proxy name="aliyun-emr-socks-proxy" id="2322596116" notes
     ="" fromSubscription="false" enabled="true" mode="manual"
     selectedTabIndex="2" lastresort="false" animatedIcons="true
     " includeInCycle="true" color="#0055E5" proxyDNS="true"
     noInternalIPs="false" autoconfMode="pac" clearCacheBeforeUse
     ="false" disableCache="false" clearCookiesBeforeUse="false"
     rejectCookies="false">
     <matches>
     <match enabled="true" name="120.*" pattern="http://120.*"
     isRegEx="false" isBlackList="false" isMultiLine="false"
     caseSensitive="false" fromSubscription="false" ></match>
     </matches>
     <manualconf host="localhost" port="8157" socksversion="5"
     isSocks="true" username="" password="" domain="" ></manualconf>
     </proxy>
     </proxies>
     </foxyproxy>
     ```

     Specifically:

     - `Port 8157` is a port that your local machine uses to build an SSH connection with the cluster's master machine. It is the same port used in the SSH command executed in the terminal.
     - `120.*` is used to match with the IP address in the master machine. Confirm it according to the master IP address.
  3. In a browser, click the FoxyProxy button and then select Options.

4. Select Import/Export.
5. Select the XML file you edited in Step 2 and click Open.
6. In the Import FoxyProxy Setting dialog box, click Add.
7. In a browser, click the FoxyProxy button and select Use Proxy aliyun-emr-socks-proxy for all URLs.
8. Enter localhost:8088 in your browser to open the Hadoop interface.

- Local port forwarding

  Notice:

  One disadvantage of local port forwarding is that only the interface in the outermost layer can be seen. The viewing of detailed job information results in an error.

  ```
  ssh -i /path/id_rsa -N -L 8157:masterNodeIP:8088 hadoop@masterNodeIP
  ```

  The parameters are described as follows:

  - `path`: Private key storage path.
  - `masterNodeIP`: IP address of the master node to be connected.
  - `8088`: Access port number of ResourceManager on the master node.

12 Create a gateway

A gateway is an ECS server in the same intranet as an E-MapReduce cluster. You can use gateways for load balancing and security isolation or to submit jobs to E-MapReduce clusters.

You can create a gateway in the following two ways:

- In *E-MapReduce* (recommended).
- Manually.

Create a gateway on the E-MapReduce console

E-MapReduce gateways only support Hadoop clusters. Before you create an E-MapReduce gateway, you first need to create a Hadoop cluster. To create an E-MapReduce gateway, complete the following steps.

1. Log on to the *E-MapReduce console* .
2. Click Create Gateway.
3. Enter the required information on the Create Gateway page.
    - Billing Method:
        - Subscription: Charges for a specified period of time. This method is more cost-effective than Pay-As-You-Go, especially when you pay for three years at a time, as you get a larger discount.
        - Pay-As-You-Go: Charges by the hour. This method calculates fees based on the number of hours that you use the product.
    - Cluster: Create a gateway that the gateway can submit jobs to. Gateways automatically configure the Hadoop environment that is consistent with the cluster.
    - Configuration: The available ECS instance specifications in the zone.
    - System Disk Type: The system disk type of the gateway node. There are two types of system disk: SSD cloud disk and efficient cloud disk. The displayed type varies according to the server model and region. By default, the system disk is released when the cluster is released.
    - System Disk Size: The minimum size is 40 GB and the maximum is 500 GB. The default value is 300 GB.

  - Data Disk Type: The data disk type of the gateway node. There are two types of data disk: SSD cloud disk and efficient cloud disk. The displayed type varies according to the server model and region. By default, the data disk is released when the cluster is released.
  - Data Disk Size: The minimum size is 200 GB and the maximum is 4000 GB. The default value is 300 GB.
  - Quantity: The number of data disks. The minimum is 1 and the maximum is 10.
  - Cluster Name: The name of a gateway. It can contain between 1 and 64 characters. Only Chinese characters, letters, numbers, hyphens (-), and underscores (_) are allowed.
  - Password/Key Pair:
    - Password Mode: Enter the password for logging on to the gateway in the text box.
    - Key Pair Mode: Select the key pair name for logging on to the gateway in the drop-down menu. If no key pair has been created yet, click Create Key Pair on the right to go to the ECS console. Do not disclose private key files in the .pem format that correspond to the key pair. After the gateway is created, the public key of the key pair is automatically bound to the ECS where the gateway is located. When you log on to the gateway through SSH, enter the private key in the private key file.

4. Click Create to save the configurations.

   If the creation is successful, the newly created gateway is displayed in the cluster list and the status in the Status column becomes Idle.

Create a gateway manually

- Network environment

  Make sure that the gateway machine is in the security group of the corresponding E-MapReduce cluster. This allows the gateway nodes to have easy access to the E-MapReduce cluster. For more information about setting the security group of a machine, see *Create a security group*.

- Software environment
  - System environment: CentOS 7.2+ is recommended.
  - Java environment: JDK 1.7 or later must be installed. OpenJDK 1.8.0 is recommended.

- **Procedure**
  - **E-MapReduce 2.7 or later, 3.2 or later**

    To create a gateway in these versions, we recommend that you use the E-MapReduce console.

    If you want to set up a gateway manually, copy the following script to the gateway host and run it: `sh deploy.sh <masteri_ip> master_password_file`.

    - *`deploy.sh`* **is the script name.**
    - *`masteri_ip`* **is the IP address of the master node in the cluster, which needs to be accessible.**
    - *`master_password_file`* **is the file for storing the password of the master node, which is written in the file.**

```
#! /usr/bin/bash
If [$ #! = 2]
then
   echo "Usage: $0 master_ip master_password_file"
   exit 1;
fi
masterip=$1
masterpwdfile=$2
if ! type sshpass >/dev/null 2>&1; then
   yum install -y sshpass
fi
if ! type java >/dev/null 2>&1; then
   yum install -y java-1.8.0-openjdk
fi
mkdir -p /opt/apps
mkdir -p /etc/ecm
echo "Start to copy package from $masterip to local gateway(/opt/
apps)"
echo " -copying hadoop-2.7.2"
sshpass -f $masterpwdfile scp -r -o 'StrictHostKeyChecking no'
root@$masterip:/usr/lib/hadoop-current /opt/apps/
echo " -copying hive-2.0.1"
sshpass -f $masterpwdfile scp -r root@$masterip:/usr/lib/hive-
current /opt/apps/
echo " -copying spark-2.1.1"
sshpass -f $masterpwdfile scp -r root@$masterip:/usr/lib/spark-
current /opt/apps/
echo "Start to link /usr/lib/\${app}-current to /opt/apps/\${app}"
if [ -L /usr/lib/hadoop-current ]
then
   unlink /usr/lib/hadoop-current
fi
ln -s /opt/apps/hadoop-current  /usr/lib/hadoop-current
if [ -L /usr/lib/hive-current ]
then
   unlink /usr/lib/hive-current
fi
ln -s /opt/apps/hive-current  /usr/lib/hive-current
if [ -L /usr/lib/spark-current ]
then
```

```
   unlink /usr/lib/spark-current
fi
ln -s /opt/apps/spark-current /usr/lib/spark-current
echo "Start to copy conf from $masterip to local gateway(/etc/ecm
)"
sshpass -f $masterpwdfile scp -r root@$masterip:/etc/ecm/hadoop-
conf  /etc/ecm/hadoop-conf
sshpass -f $masterpwdfile scp -r root@$masterip:/etc/ecm/hive-conf
 /etc/ecm/hive-conf
sshpass -f $masterpwdfile scp -r root@$masterip:/etc/ecm/spark-
conf /etc/ecm/spark-conf
echo "Start to copy environment from $masterip to local gateway(/
etc/profile.d)"
sshpass -f $masterpwdfile scp root@$masterip:/etc/profile.d/hdfs.
sh /etc/profile.d/
sshpass -f $masterpwdfile scp root@$masterip:/etc/profile.d/yarn.
sh /etc/profile.d/
sshpass -f $masterpwdfile scp root@$masterip:/etc/profile.d/hive.
sh /etc/profile.d/
sshpass -f $masterpwdfile scp root@$masterip:/etc/profile.d/spark.
sh /etc/profile.d/
if [ -L /usr/lib/jvm/java ]
then
   unlink /usr/lib/jvm/java
fi
echo "" >>/etc/profile.d/hdfs.sh
echo export JAVA_HOME=/usr/lib/jvm/jre-1.8.0 >>/etc/profile.d/hdfs
.sh
echo "Start to copy host info from $masterip to local gateway(/etc
/hosts)"
sshpass -f $masterpwdfile scp root@$masterip:/etc/hosts /etc/
hosts_bak
cat /etc/hosts_bak | grep emr | grep cluster >>/etc/hosts
if ! id hadoop >& /dev/null
then
   useradd hadoop
fi
```

- E-MapReduce 2.7 or earlier, 3.2 or earlier

  Copy the following script to the gateway host and run it: `sh deploy.sh <masteri_ip> master_password_file`.

  - *`deploy.sh`* is the script name.
  - *`masteri_ip`* is the IP address of the master node in the cluster, which needs to be accessible.
  - *`master_password_file`* is the file for storing the password of the master node, which is written in the file.

```
! /usr/bin/bash
if [ $# ! = 2 ]
then
   echo "Usage: $0 master_ip master_password_file"
   exit 1;
fi
masterip=$1
masterpwdfile=$2
if ! type sshpass >/dev/null 2>&1; then
```

```
   yum install -y sshpass
fi
if ! type java >/dev/null 2>&1; then
   yum install -y java-1.8.0-openjdk
fi
mkdir -p /opt/apps
mkdir -p /etc/emr
echo "Start to copy package from $masterip to local gateway(/opt/
apps)"
echo " -copying hadoop-2.7.2"
sshpass -f $masterpwdfile scp -r -o 'StrictHostKeyChecking no'
root@$masterip:/usr/lib/hadoop-current /opt/apps/
echo " -copying hive-2.0.1"
sshpass -f $masterpwdfile scp -r root@$masterip:/usr/lib/hive-
current /opt/apps/
echo " -copying spark-2.1.1"
sshpass -f $masterpwdfile scp -r root@$masterip:/usr/lib/spark-
current /opt/apps/
echo "Start to link /usr/lib/\${app}-current to /opt/apps/\${app}"
if [ -L /usr/lib/hadoop-current ]
then
   Unlink/usr/lib/hadoop-Current
fi
ln -s /opt/apps/hadoop-current  /usr/lib/hadoop-current
if [ -L /usr/lib/hive-current ]
then
   unlink /usr/lib/hive-current
fi
ln -s /opt/apps/hive-current  /usr/lib/hive-current
if [ -L /usr/lib/spark-current ]
then
   unlink /usr/lib/spark-current
fi
Ln-S/opt/apps/spark-current/usr/lib/spark-Current
echo "Start to copy conf from $masterip to local gateway(/etc/emr
)"
sshpass -f $masterpwdfile scp -r root@$masterip:/etc/emr/hadoop-
conf  /etc/emr/hadoop-conf
sshpass -f $masterpwdfile scp -r root@$masterip:/etc/emr/hive-conf
 /etc/emr/hive-conf
sshpass -f $masterpwdfile scp -r root@$masterip:/etc/emr/spark-
conf /etc/emr/spark-conf
Echo "start to copy environment from $ masterip to local Gateway
 (/etc/profile. d )"
sshpass -f $masterpwdfile scp root@$masterip:/etc/profile.d/hadoop
.sh /etc/profile.d/
if [ -L /usr/lib/jvm/java ]
then
   unlink /usr/lib/jvm/java
fi
ln -s /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.131-3.b12.el7_3.x86_64
/jre /usr/lib/jvm/java
echo "Start to copy host info from $masterip to local gateway(/etc
/hosts)"
sshpass -f $masterpwdfile scp root@$masterip:/etc/hosts /etc/
hosts_bak
cat /etc/hosts_bak | grep emr | grep cluster >>/etc/hosts
if ! id hadoop >& /dev/null
then
   useradd hadoop
fi
```

- **Test**

- **Hive**

```
[hadoop@iZ23bc05hrvZ ~]$ hive
hive> show databases;
OK
default
Time taken: 1.124 seconds, Fetched: 1 row(s)
hive> create database school;
OK
Time taken: 0.362 seconds
hive>
```

- **Run the Hadoop job**

```
[hadoop@iZ23bc05hrvZ ~]$ hadoop  jar /usr/lib/hadoop-current/share
/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar pi 10 10
Number of Maps  = 10
Samples per Map = 10
Wrote input for Map #0
Wrote input for Map #1
Wrote input for Map #2
Wrote input for Map #3
Wrote input for Map #4
Wrote input for Map #5
Wrote input for Map #6
Wrote input for Map #7
Wrote input for Map #8
Wrote input for Map #9
  File Input Format Counters
      Bytes Read=1180
  File Output Format Counters
      Bytes Written=97
Job Finished in 29.798 seconds
Estimated value of Pi is 3.20000000000000000000
```

13 MetaService

MetaService allows you to access Alibaba Cloud resources in the E-MapReduce cluster without using an AccessKey (AK).

Default roles

When creating a cluster, you must authorize an application role (AliyunEmrEcsDefaultRole) to E-MapReduce. After you do so, you can perform operations on E-MapReduce to access Alibaba Cloud resources without using an AK. By default, the following permission policies are granted to AliyunEmrEcsDefaultRole:

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "oss:GetObject",
        "oss:ListObjects",
        "oss:PutObject",
        "oss:DeleteObject",
        "oss:ListBuckets",
        "oss:AbortMultipartUpload"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

By default, operations based on MetaService can only access OSS data. If you want to use MetaService to access other Alibaba Cloud resources, such as LogService, you must grant permissions to AliyunEmrEcsDefaultRole. Perform the preceding operations on the *RAM console*.

Notice:

MetaService only supports AK-free operations on OSS, LogService, and MNS data. Modify and delete the default role with caution. Otherwise, you may fail to create or perform operations on clusters.

Custom application roles

When creating a cluster, you can use a default role or create your own application role. In most cases, you only need to use or modify the default role. For more information about how to create and authorize a role to E-MapReduce, see *RAM*.

Access to MetaService

MetaService is an HTTP service that can be accessed directly to obtain metadata. For example, by using the `curl http://localhost:10011/cluster-region` command, you can obtain the region where the current cluster is located.

MetaService supports the following types of information:

- Region: /cluster-region
- Role name: /cluster-role-name
- AccessKeyId: /role-access-key-id
- AccessKeySecret: /role-access-key-secret
- SecurityToken: /role-security-token
- Network type: /cluster-network-type

Use MetaService

You can use MetaService to access Alibaba Cloud resources without using an AK. This has the following advantages:

- Reduces the risk of an AK leak. The use of RAM also minimizes security risks, as only the required permissions are granted to the role.
- Improves user experience. For instance, when you access OSS resources interactively, you no longer need to write a long string of OSS paths.

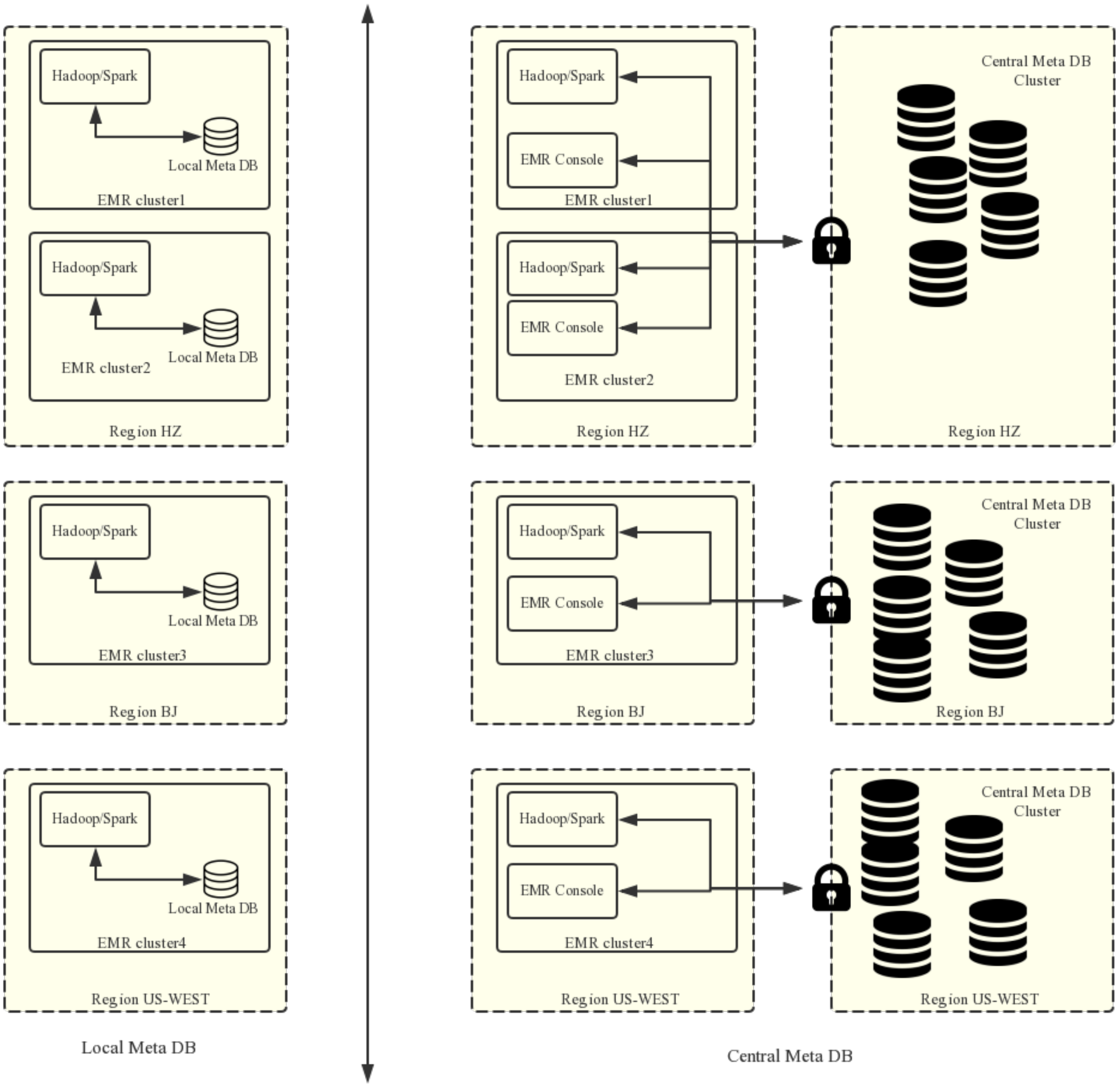
The usage methods are as follows:

```
I. Using the Hadoop command line to display OSS data
    Previously, we used: hadoop fs -ls oss://ZaH******As1s:Ba23N
**************sdaBj2@bucket.oss-cn-hangzhou-internal.aliyuncs.com/a/b/
c
    Now, we use: hadoop fs -ls oss://bucket/a/b/c
II. Using Hive to create a table
    Previously, we used:
        CREATE EXTERNAL TABLE test_table(id INT, name string)
        ROW FORMAT DELIMITED
        FIELDS TERMINATED BY '/t'
        LOCATION 'oss://ZaH******As1s:Ba23N**************sdaBj2@bucket
.oss-cn-hangzhou-internal.aliyuncs.com/a/b/c';
    Now, we use:
        CREATE EXTERNAL TABLE test_table(id INT, name string)
        ROW FORMAT DELIMITED
        FIELDS TERMINATED BY '/t'
        LOCATION 'oss://bucket/a/b/c';
III. Spark
    Previously, we used: val data = sc.textFile("oss://ZaH******As1s:
Ba23N**************sdaBj2@bucket.oss-cn-hangzhou-internal.aliyuncs.com
/a/b/c")
    Now, we use: val data = sc.textFile("oss://bucket/a/b/c")
```

14 Table management

E-MapReduce 2.4.0 and later support the central management of metadata in the highly reliable Hive metastore. In earlier versions, clusters use the local MySQL database.

Introduction



When you create a cluster, you can enable the central metastore function so that the cluster uses an external metastore.

Note:

- The current metastore needs to be connected using the public IP address. The cluster must therefore have a public IP address. Do not change this IP address, otherwise the corresponding database whitelist becomes invalid.
- The table management function can only be used if the central metastore function was enabled when a cluster was created. Local metastores do not currently support table management. If you have a local metastore, use the Hue tool in the cluster to manage your tables.

The central metadata management function performs the following:

1. Provides long-term metadata storage.

   If metadata is stored in the local MySQL database of the cluster, it is lost when the cluster is released. Clusters can be created and released at any time as required, especially when E-MapReduce supports flexible creation. To retain metadata, you must log on to the cluster and manually export the data. This issue is resolved with the central metadata management function.
2. Separates computing and storage.

   E-MapReduce supports storing data in Alibaba Cloud OSS, which reduces usage costs, especially when dealing with high volumes of data. Meanwhile , E-MapReduce clusters are mainly used as computing resources and can be released at any time after use. Since data is stored in OSS, problems with metadata migration are non-existent.
3. Implements data sharing.

   With the central metastore, if all data is stored in OSS, all clusters can access the data without migrating or restructuring it. This enables E-MapReduce clusters to provide different services, while still ensuring direct data sharing.

Notice:

Before central metadata management is supported, metadata is stored in the local MySQL database of each cluster and is lost when the cluster is released. With central metadata management, releasing clusters does not clean up metadata. Before you delete the data in OSS or in the HDFS of a cluster or you release a cluster, make sure that the corresponding metadata is already deleted. That means the tables and database that store the data have been dropped. This prevents dirty metadata in the database.

Table management operations

Before E-MapReduce clusters can support metadata management, you have to first log on to the internal environment of a cluster to check, add, or delete tables. If more than one cluster exists, you have to log on to each, one by one. This is inconvenient. With the central metadata management function, E-MapReduce enables table management on the console. This includes checking the list of databases and tables, checking table details, creating and deleting databases and tables, and previewing data.

- Create a database or table list
- View table details
- Preview data
- Create a database
- Create a table

  There are two ways of creating a table: manually and from a file.

  - Manual creation: If no service data exists, you can manually input the table structure to create an empty table.
  - Creation from a file: If service data already exists, you can use it as a table by parsing the table interface from the file. Make sure that the separators used for creating a table correspond to those used in the data file. This guarantees a proper table structure.

  The separators can be common characters such as commas and spaces, or special characters TAB, ^A, ^B, and ^C.

  Notice:

  1. Databases and tables can only be created and deleted in E-MapReduce clusters.
  2. The HDFS is the internal file system of each cluster and does not support cross-cluster communication without special network settings. Therefore, the E-MapReduce table management function only supports creating databases and tables based on the OSS file system.
  3. The location of a database or table must be in a directory under the OSS bucket, rather than the OSS bucket.

Common issues

1. Wrong FS: `oss://yourbucket/xxx/xxx/xxx`.

   This error occurs when the table data on OSS is deleted but the table metadata is not. The table schema continues to exist, but the actual data does not or is moved to another location. In this case, you can change the table location to an existing path and delete the table again.

   ```
   alter table test set location 'oss://your_bucket/your_folder'
   ```

   You can complete this operation on the E-MapReduce interactive console.

   > Note:
   >
   > *oss://your_bucket/your_folder* must be an existing OSS path.

2. Wrong FS: hdfs://yourhost:9000/xxx/xxx/xxx.

   This error occurs when the table data in the HDFS is deleted but the table schema is not. The error can be removed by following the preceding solution.

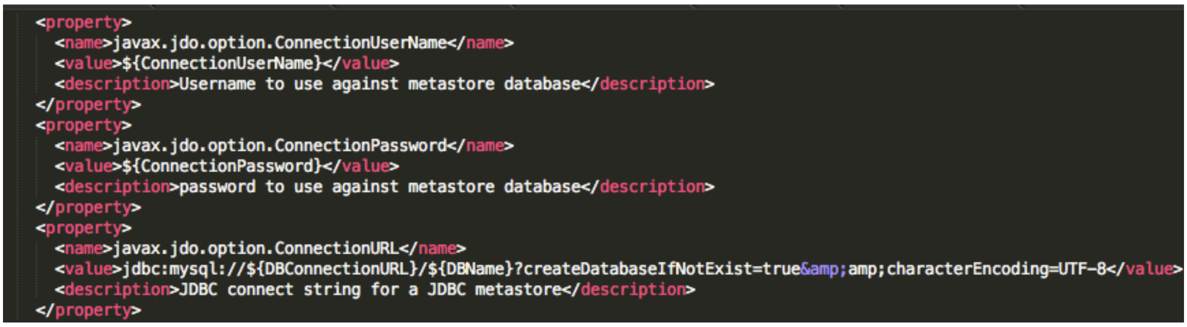
3. The message java.lang.IllegalArgumentException: java.net.UnknownHostException: xxxxxxx is displayed when the Hive database is deleted.

   This error occurs because the Hive database is created in the HDFS of a cluster and is not cleaned when the cluster is released. As a result, its data in the HDFS of the released cluster cannot be accessed after a new cluster is created. Therefore, when releasing a cluster, remember to clean the databases and tables that are manually created in the HDFS of the cluster.

   To resolve this problem, log on to the master node of the cluster using the command line, and find the address, user name, and password used to access the Hive metastore in *$HIVE_CONF_DIR/hive-site.xml*.

   ```
   javax.jdo.option.ConnectionUserName //Username for accessing the
   database;
   javax.jdo.option.ConnectionPassword //Password for accessing the
   database;
   ```

```
javax.jdo.option.ConnectionURL //Address and name of the database ;
```

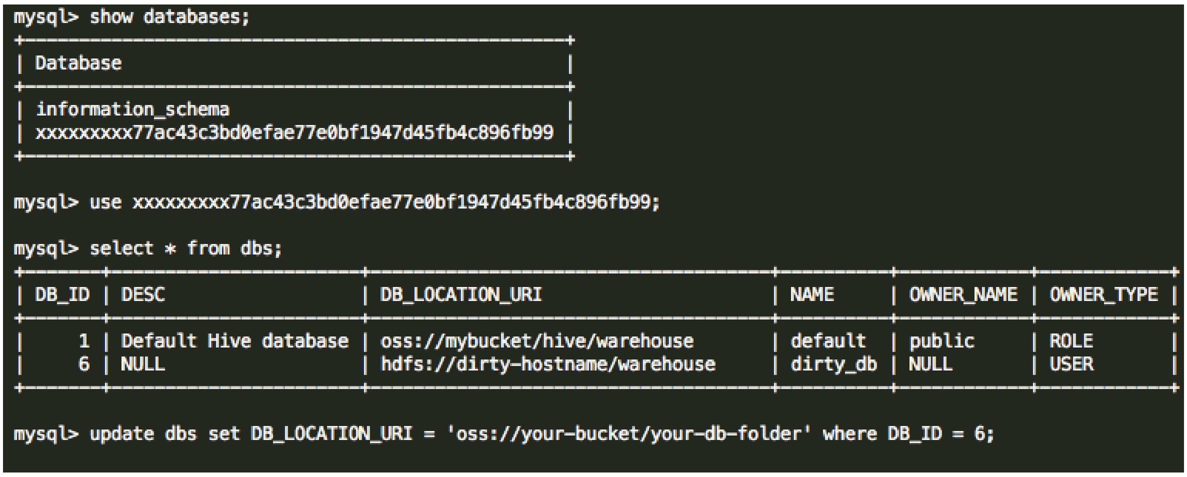


```
<property>
  <name>javax.jdo.option.ConnectionUserName</name>
  <value>${ConnectionUserName}</value>
  <description>Username to use against metastore database</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionPassword</name>
  <value>${ConnectionPassword}</value>
  <description>password to use against metastore database</description>
</property>
<property>
  <name>javax.jdo.option.ConnectionURL</name>
  <value>jdbc:mysql://${DBConnectionURL}/${DBName}?createDatabaseIfNotExist=true&amp;amp;characterEncoding=UTF-8</value>
  <description>JDBC connect string for a JDBC metastore</description>
</property>
```

Log on to the Hive metastore on the master node of the cluster:

```
mysql -h ${DBConnectionURL} -u ${ConnectionUserName} -p [Press Enter
]
[Enter the password]${ConnectionPassword}
```

After logging on, change its location to an existing OSS path in the region:



```
mysql> show databases;
+--------------------------------------------------+
| Database                                         |
+--------------------------------------------------+
| information_schema                               |
| xxxxxxxxx77ac43c3bd0efae77e0bf1947d45fb4c896fb99 |
+--------------------------------------------------+

mysql> use xxxxxxxxx77ac43c3bd0efae77e0bf1947d45fb4c896fb99;

mysql> select * from dbs;
+-------+-----------------------+------------------------------------+----------+------------+------------+
| DB_ID | DESC                  | DB_LOCATION_URI                    | NAME     | OWNER_NAME | OWNER_TYPE |
+-------+-----------------------+------------------------------------+----------+------------+------------+
|     1 | Default Hive database | oss://mybucket/hive/warehouse      | default  | public     | ROLE       |
|     6 | NULL                  | hdfs://dirty-hostname/warehouse    | dirty_db | NULL       | USER       |
+-------+-----------------------+------------------------------------+----------+------------+------------+

mysql> update dbs set DB_LOCATION_URI = 'oss://your-bucket/your-db-folder' where DB_ID = 6;
```

15 Kerberos authentication

15.1 Introduction to Kerberos

Kerberos is a network authentication protocol that allows nodes communicating over a non-secure network to securely prove their identity. From versions 2.7.x and 3.5.x onwards, E-MapReduce supports the creation of clusters in which open source components are started in the Kerberos security mode. In this mode, only authenticated clients can access the cluster service, such as HDFS.

Prerequisites

The Kerberos components supported by the latest E-MapReduce version are shown in the following table:

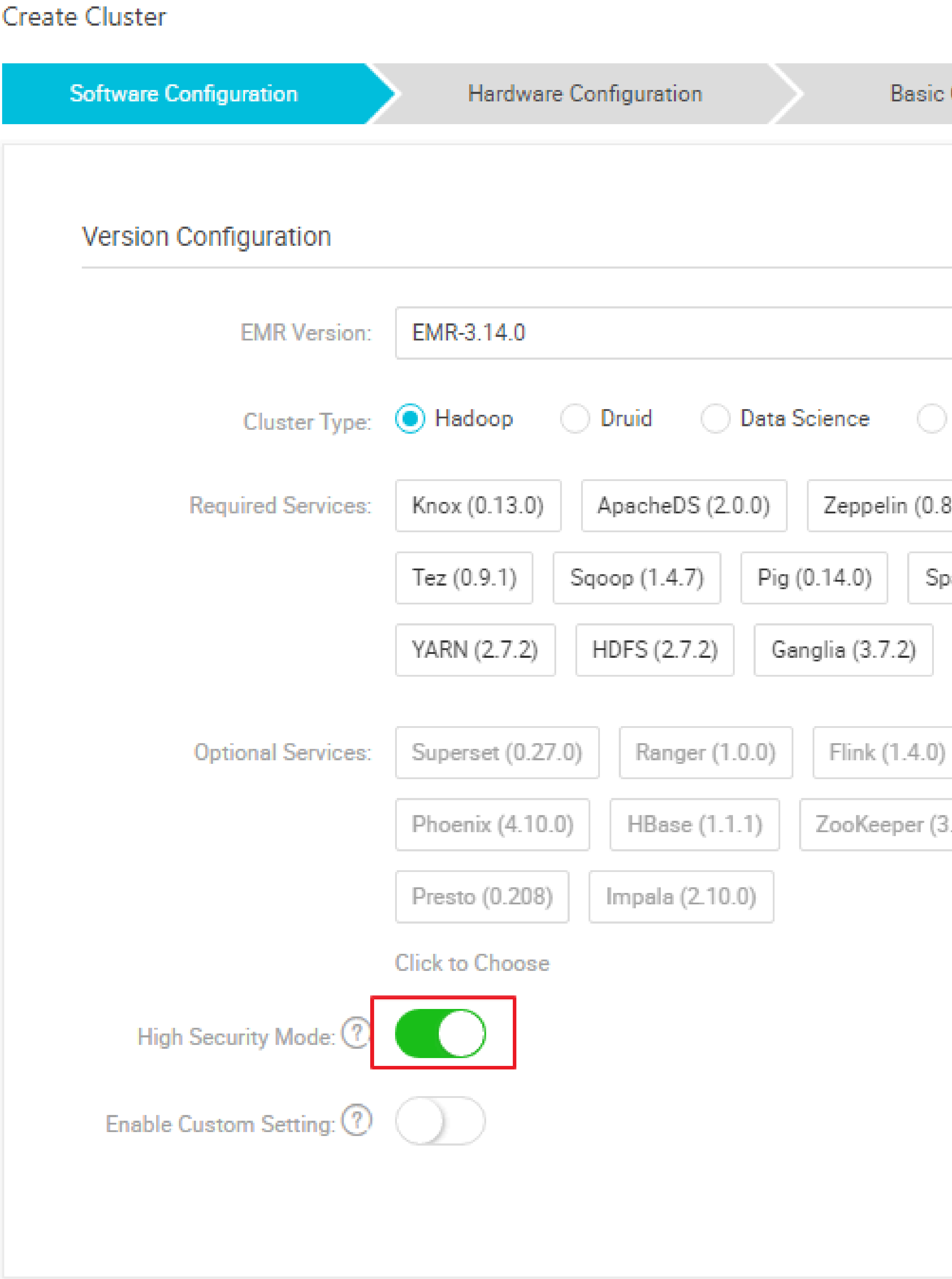
Component name	Component version
YARN	2.7.2
Spark	2.1.1/1.6.3
Hive	2.0.1
Tez	0.8.4
ZooKeeper	3.4.6
Hue	3.12.0
Zeppelin	0.7.1
Oozie	4.2.0
Sqoop	1.4.6
HBase	1.1.1
Phoenix	4.7.0

Note:

Kafka, Presto, and Storm do not currently support Kerberos.

Create a security cluster

In the software configuration tab on the cluster creation page, you can turn on High Security Mode, as shown in the following figure:



Create Cluster

Version Configuration

EMR Version: EMR-3.14.0

Cluster Type: ● Hadoop ○ Druid ○ Data Science ○

Required Services: Knox (0.13.0) ApacheDS (2.0.0) Zeppelin (0.8 Tez (0.9.1) Sqoop (1.4.7) Pig (0.14.0) Sp YARN (2.7.2) HDFS (2.7.2) Ganglia (3.7.2)

Optional Services: Superset (0.27.0) Ranger (1.0.0) Flink (1.4.0) Phoenix (4.10.0) HBase (1.1.1) ZooKeeper (3. Presto (0.208) Impala (2.10.0)

Click to Choose

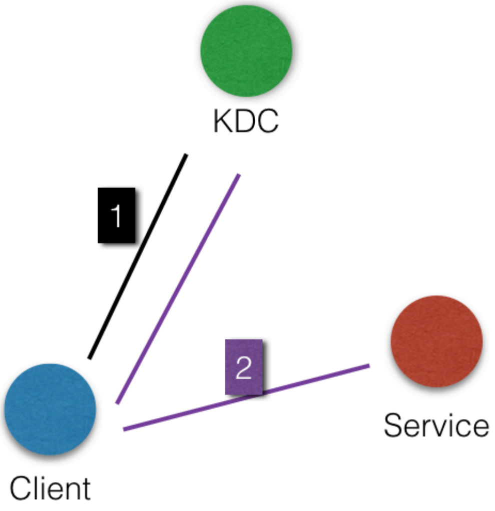
High Security Mode: ? ●

Enable Custom Setting: ? ○

Kerberos authentication

Kerberos is an identity authentication protocol based on symmetric key cryptography. As a third-party authentication service, Kerberos can provide its authentication function for other services. It also supports SSO, and the client can access multiple services, such as HBase and HDFS, after authentication.

The Kerberos protocol process is mainly divided into two stages: the KDC authenticates the client ID, and the service authenticates the client ID.



- KDC

  Kerberos server
- Client

  If a user (principal) needs to access the service, the KDC and service authenticate the principal's identity.
- Service

  Services that have integrated with Kerberos include HDFS, YARN, and HBase.
- KDC ID authentication

  Before a principal can access a service integrated with Kerberos, it must first pass KDC ID authentication.

  After doing so, the client receives a ticket-granting ticket (TGT), which can be used to access a service that has integrated Kerberos.
- Service ID authentication

When a principal receives the TGT, it can access the service. It uses the TGT and the name of the service that it must access (such as HDFS) to obtain a service-granting ticket (SGT) from the KDC, and uses the SGT to access the service. This then uses the relevant information to conduct ID authentication on the client. After passing authentication, the client can access the service as normal.

Kerberos and E-MapReduce

When you create a cluster, services in the E-MapReduce Kerberos security cluster start in the Kerberos security mode.

- The Kerberos server is a HasServer
  - Log on to the *Alibaba Cloud E-MapReduce console*, choose Cluster > > Configuration Management > HAS, and conduct operations such as view, modify configuration, and restart.
  - Non-HA clusters are deployed on the emr-header-1 node, whereas HA clusters are deployed on both the emr-header-1 and emr-header-2 nodes.
- Supports four ID authentication methods

  HasServer supports the following four ID authentication methods. The client can specify the method that is used by HasServer by configuring the relevant parameters.

  - ID authentication compatible with MIT Kerberos

    Client configuration:

    ```
    If you want to execute a client request on a cluster node, you
    must set
    hadoop.security.authentication.use.has in /etc/ecm/hadoop-conf/
    core-site.xml to false.
    In case of any jobs are running through the execution plan of the
     console, then values in the /etc/ecm/hadoop-conf/core-site.xml
    file on the master node must not be modified. Otherwise, the job
    in the execution plan fails because of the authentication failure
    . You can follow these steps:
    export HADOOP_CONF_DIR=/etc/has/hadoop-conf Export a temporary
    environment variable. The hadoop.security.authentication.use.has
    value under this path has already been set to false.
    ```

    Access method: You can use open source clients to access the service, such as an HDFS client. For more information, *click here*.
  - RAM ID authentication

Client configuration:

```
If you want to run a client request on a cluster node, you must
set
hadoop.security.authentication.use.has in /etc/ecm/hadoop-conf/
core-site.xml to true, and auth_type in /etc/has/has-client.conf
to RAM.
In case of any jobs are running through the execution plan of the
 console, then values in the /etc/ecm/hadoop-conf/core-site.xml
and /etc/has/has-client.conf files on the master node must not be
 modified. Otherwise, the job in the execution plan fails because
of the authentication failure. You can use the following method:
export HADOOP_CONF_DIR=/etc/has/hadoop-conf; export HAS_CONF_DIR
=/path/to/has-client.conf Export a temporary environment variable
, and then set the auth_type in the has-client.conf file of the
HAS_CONF_DIR folder to RAM.
```

Access method: The client must use a software package of the cluster, such as Hadoop or HBase. For more information, *click here*.

- LDAP ID authentication

  Client configuration:

```
If you want to execute a client request on a cluster node, you
must set
hadoop.security.authentication.use.has in /etc/ecm/hadoop-conf/
core-site.xml to true, and auth_type in /etc/has/has-client.conf
to LDAP.
In case of any jobs are running through the execution plan of the
 console, then values in the /etc/ecm/hadoop-conf/core-site.xml
and /etc/has/has-client.conf files on the master node must not be
 modified. Otherwise, the job in the execution plan fails because
of the authentication failure. You can follow these steps:
export HADOOP_CONF_DIR=/etc/has/hadoop-conf; export HAS_CONF_DIR
=/path/to/has-client.conf Export temporary environment viarables
, and then set the auth_type in the has-client.conf file of the
HAS_CONF_DIR folder to LDAP.
```

  Access method: The client must use a software package of the cluster, such as Hadoop or HBase. For more information, *click here*.

- Execution plan authentication

  If you have jobs submitted through the execution plan of the E-MapReduce console, you must not modify the default configuration of the emr-header-1 node.

  Client configuration:

```
Set hadoop.security.authentication.use.has in /etc/ecm/hadoop-conf
/core-site.xml to true, and auth_type in /etc/has/has-client.conf
on emr-header-1 to EMR.
```

  For more information, *click here*.

- Others

  Log on to the master node to access the cluster

  The administrator can use the Has account (the default logon method is MIT-Kerberos-compatible) to log on to the master node and access the cluster service. This facilitates troubleshooting and O&M tasks.

  ```
  &gt;sudo su has
  &gt;hadoop fs -ls /
  ```

  > **Note:**
  >
  > Other accounts can also be used to log on to the master node, provided that such accounts have already passed Kerberos authentication. In addition, if you have to use the MIT-Kerberos-compatible method on the master node, you must first export an environment variable under this account.

  `export HADOOP_CONF_DIR=/etc/has/hadoop-conf/`

15.2 Authentication compatible with MIT Kerberos

This section details the MIT Kerberos authentication process through the HDFS service.

Authentication method

The Kerberos server in the E-MapReduce cluster is started at the master node. Some management operations must be performed with the root account of the master node (emr-header-1).

In the following example, the test user accesses the HDFS service to introduce relevant procedures.

- Execute `hadoop fs -ls /` on the gateway.

  - Configure krb5.conf.

    ```
    Use root account on the Gateway
     scp root@emr-header-1:/etc/krb5.conf /etc/
    ```

  - Add principal.

    ■ Log on to the emr-header-1 node and switch to the root account.

- Open the admin tool in Kerberos.
-
  ```
    sh /usr/lib/has-current/bin/hadmin-local.sh /etc/ecm/has-conf
   -k /etc/ecm/has-conf/admin.keytab
    HadminLocalTool.local: #Press Enter to view the use of the
  commands
    HadminLocalTool.local: addprinc #Input the command and press
  Enter to view the use of the specific command
    HadminLocalTool.local: addprinc -pw 123456 test #Add
  principal for the user test, and set the password to 123456
  ```

- Export the keytab file.

  Use the Kerberos admin tool to export the keytab file that corresponds to the principal.

  ```
  HadminLocalTool.local: ktadd -k /root/test.keytab test #Export the
   keytab file, which can be used subsequently
  ```

- Use kinit to obtain the ticket.

  On the client machine where HDFS commands are executed, such as the gateway :

  - Add the Linux account test `useradd test`.
  - Install MIT Kerberos client tools.

    MIT Kerberos tools can be used for related operations (such as kinit and klist). For more information, see *MIT Kerberos*.

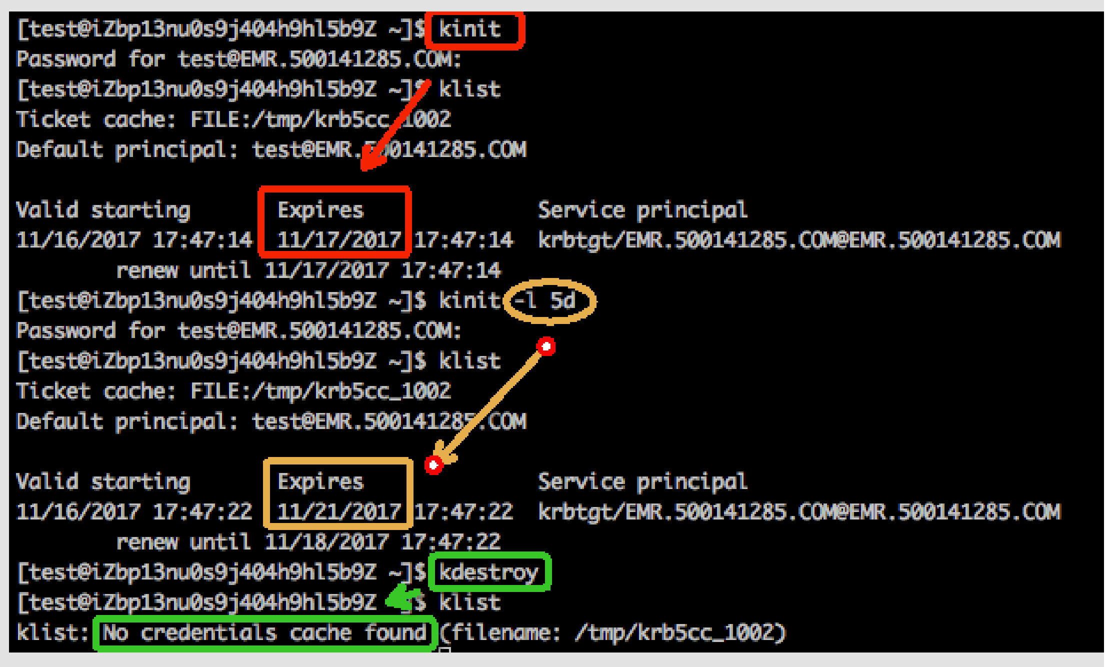
    `yum install krb5-libs krb5-workstation -y`

  - Switch to the test account to execute kinit.

    ```
          su test
          #If the keytab file does not exist, execute
          kinit #Press Enter
          Password for test: 123456 #Done
          #the keytab file exists, execute
          kinit -kt test.keytab test
          #View the ticket
          klist
    ```

    Note:

    Application of MIT Kerberos tools



```
[test@iZbp13nu0s9j404h9hl5b9Z ~]$ kinit
Password for test@EMR.500141285.COM:
[test@iZbp13nu0s9j404h9hl5b9Z ~]$ klist
Ticket cache: FILE:/tmp/krb5cc_1002
Default principal: test@EMR.500141285.COM

Valid starting       Expires              Service principal
11/16/2017 17:47:14  11/17/2017 17:47:14  krbtgt/EMR.500141285.COM@EMR.500141285.COM
        renew until 11/17/2017 17:47:14
[test@iZbp13nu0s9j404h9hl5b9Z ~]$ kinit -l 5d
Password for test@EMR.500141285.COM:
[test@iZbp13nu0s9j404h9hl5b9Z ~]$ klist
Ticket cache: FILE:/tmp/krb5cc_1002
Default principal: test@EMR.500141285.COM

Valid starting       Expires              Service principal
11/16/2017 17:47:22  11/21/2017 17:47:22  krbtgt/EMR.500141285.COM@EMR.500141285.COM
        renew until 11/18/2017 17:47:22
[test@iZbp13nu0s9j404h9hl5b9Z ~]$ kdestroy
[test@iZbp13nu0s9j404h9hl5b9Z ~]$ klist
klist: No credentials cache found (filename: /tmp/krb5cc_1002)
```

- Execute HDFS commands.

  When a ticket is obtained, HDFS commands can be executed as usual.

```
hadoop fs -ls /
      Found 5 items
     drwxr-xr-x     - hadoop hadoop          0 2017-11-12 14:23 /
apps
     drwx------      - hbase  hadoop          0 2017-11-15 19:40
 /hbase
     drwxrwx--t+   - hadoop hadoop         0 2017-11-15 17:51 /
spark-history
     drwxrwxrwt   - hadoop hadoop         0 2017-11-13 23:25 /tmp
     drwxr-x--t      - hadoop hadoop          0 2017-11-13 16:12 /
user
```

Note:

To run a YARN job, you need to add the corresponding Linux accounts to all of the nodes in the cluster in advance. For more information, see *Add test account to the E-MapReduce cluster*.

- Use Java to access HDFS.

  - Use a local ticket cache.

Note:

To obtain the ticket, you need to execute kinit in advance. When the ticket expires, the application is not accessed.

```
public static void main(String[] args) throws IOException {
   Configuration conf = new Configuration();
   //Load the HDFS configuration, which is copied from the EMR
cluster
   conf.addResource(new Path("/etc/ecm/hadoop-conf/hdfs-site.xml
"));
   conf.addResource(new Path("/etc/ecm/hadoop-conf/core-site.xml
"));
   //kinit needs to be executed in advance to obtain the ticket
with the Linux account of the application
   UserGroupInformation.setConfiguration(conf);
   UserGroupInformation.loginUserFromSubject(null);
   FileSystem fs = FileSystem.get(conf);
   FileStatus[] fsStatus = fs.listStatus(new Path("/"));
   for(int i = 0; i < fsStatus.length; i++){
       System.out.println(fsStatus[i].getPath().toString());
   }
}
```

- **(Recommended) Use a keytab file.**

Note:

The keytab is valid for a long time and has nothing to do with the local ticket.

```
public static void main(String[] args) throws IOException {
  String keytab = args[0];
  String principal = args[1];
  Configuration conf = new Configuration();
  //Load the HDFS configuration, which is copied from the EMR
cluster
  conf.addResource(new Path("/etc/ecm/hadoop-conf/hdfs-site.xml
"));
  conf.addResource(new Path("/etc/ecm/hadoop-conf/core-site.xml
"));
  //Directly use keytab file, which is obtained through executing
relevant commands on master-1 in the EMR cluster [the commands are
 introduced earlier in this article]
  UserGroupInformation.setConfiguration(conf);
  UserGroupInformation.loginUserFromKeytab(principal, keytab);
  FileSystem fs = FileSystem.get(conf);
  FileStatus[] fsStatus = fs.listStatus(new Path("/"));
  for(int i = 0; i < fsStatus.length; i++){
      System.out.println(fsStatus[i].getPath().toString());
  }
  }
```

POM dependencies are attached:

```
<dependencies>
            <dependency>
                <groupId>org.apache.hadoop</groupId>
                <artifactId>hadoop-common</artifactId>
                <version>2.7.2</version>
            </dependency>
            <dependency>
                <groupId>org.apache.hadoop</groupId>
                <artifactId>hadoop-hdfs</artifactId>
                <version>2.7.2</version>
            </dependency>
```

```
        </dependencies>
```

15.3 RAM authentication

In addition to supporting an authentication method compatible with MIT Kerberos, the Kerberos server in the E-MapReduce cluster also supports using Alibaba Cloud Resource Access Management (RAM) as the identity information to perform authentication.

RAM ID authentication

RAM supports creating and managing RAM user accounts, as well as using these accounts to control access to various resources on the cloud.

The administrator of the master account can create RAM users on the RAM user management page (the user name must comply with Linux user name specifications) and download their AccessKey for the corresponding developer. The developer can then configure the AccessKey to pass Kerberos authentication and access the cluster service.

Unlike the MIT Kerberos authentication, RAM identity authentication does not require adding principals to the Kerberos server in advance.

The following example uses a RAM user account that has already been created to access a gateway.

- Add the RAM user to the E-MapReduce cluster.

  The E-MapReduce security cluster's YARN uses LinuxContainerExecutor. Running the YARN job on a cluster requires all cluster nodes to add the user account that is going to run the job. LinuxContainerExecutor conducts the related permission validation based on the user account during the execution process.

  The E-MapReduce cluster administrator executes the following code on the cluster 's master node:

  ```
  sudo su hadoop
  sh adduser.sh test 1 2
  ```

  The adduser.sh code is attached:

  ```
  # Username
   user_name=$1
   # Master node count in the cluster. For example, the HA cluster has
   two master nodes.
   master_cnt=$2
  ```

```
 # Worker node count in the cluster
 worker_cnt=$3
 for((i=1;i<=$master_cnt;i++))
 do
   ssh -o StrictHostKeyChecking=no emr-header-$i sudo useradd $
user_name
 done
 for((i=1;i<=$worker_cnt;i++))
 do
   ssh -o StrictHostKeyChecking=no emr-worker-$i sudo useradd $
user_name
done
```

- **The gateway administrator adds the test user account on the gateway machine.**

```
useradd test
```

- **The gateway administrator configures the basic Kerberos environment.**

```
sudo su root
 sh config_gateway_kerberos.sh 10.27.230.10 /pathto/emrheader1
_pwd_file
 # Ensures the value of the /etc/ecm/hadoop-conf/core-site.xml file
on the Gateway is true
 <property>
    <name>hadoop.security.authentication.use.has</name>
    <value>true</value>
 property
```

The config_gateway_kerberos.sh script is attached:

```
# IP address of the emr-header-1 in the EMR cluster
 masterip=$1
 # Saves the corresponding root logon password file for masterip
 masterpwdfile=$2
 if ! type sshpass >/dev/null 2>&1; then
    yum install -y sshpass
fi
  ## Kerberos conf
 sshpass -f $masterpwdfile scp root@$masterip:/etc/krb5.conf /etc/
 mkdir /etc/has
 sshpass -f $masterpwdfile scp root@$masterip:/etc/has/has-client.
conf /etc/has
 sshpass -f $masterpwdfile scp root@$masterip:/etc/has/truststore /
etc/has/
 sshpass -f $masterpwdfile scp root@$masterip:/etc/has/ssl-client.
conf /etc/has/
 # Modifies Kerberos client configuration, changing the default
auth_type from EMR to RAM
 # This file can be manually modified
 sed -i 's/EMR/RAM/g' /etc/has/has-client.conf
```

- **The test user logs on to the gateway and configures the AccessKey.**

```
Log on the test account of Gateway
 # Run the script
 sh add_accesskey.sh test
```

The add_accesskey.sh script is attached to modify the AccessKey:

```
user=$1
 if [[ `cat /home/$user/.bashrc | grep 'export AccessKey'` == "" ]];
then
 echo "
 # Change to the test user's AccessKeyId/AccessKeySecret
 export AccessKeyId=YOUR_AccessKeyId
 export AccessKeySecret=YOUR_AccessKeySecret
 " >>~/.bashrc
 else
    echo $user AccessKey has been added to .bashrc
 fi
```

- The test user executes the command.

  The test user is now able to execute the relevant commands to access the cluster service.

  Execute HDFS commands:

```
[test@gateway ~]$ hadoop fs -ls /
  17/11/19 12:32:15 INFO client.HasClient: The plugin type is: RAM
  Found 4 items
  drwxr-x---   - has    hadoop          0 2017-11-18 21:12 /apps
  drwxrwxrwt   - hadoop hadoop          0 2017-11-19 12:32 /spark-
history
  drwxrwxrwt   - hadoop hadoop          0 2017-11-18 21:16 /tmp
  drwxrwxrwt   - hadoop hadoop          0 2017-11-18 21:16 /user
```

  Run the Hadoop job:

```
[test@gateway ~]$ hadoop jar /usr/lib/hadoop-current/share/hadoop/
mapreduce/hadoop-mapreduce-examples-2.7.2.jar pi 10 1
```

  Run the Spark job:

```
[test@gateway ~]$ spark-submit --conf spark.ui.view.acls=* --class
 org.apache.spark.examples.SparkPi --master yarn-client --driver-
memory 512m --num-executors 1 --executor-memory 1g --executor-cores
```

```
2 /usr/lib/spark-current/examples/jars/spark-examples_2.11-2.1.1.jar
 10
```

15.4 LDAP authentication

E-MapReduce clusters also support authentication based on LDAP, which manages the account system through LDAP. The Kerberos client uses LDAP account information as identity information for authentication.

LDAP identity authentication

LDAP accounts can be shared with other services, such as Hue. You can use an LDAP service (in ApacheDS) configured in the E-MapReduce cluster or use an existing LDAP service, and you only need to configure it on the Kerberos server.

In the following example, an LDAP service (in ApacheDS) has been started by default in a cluster:

- Configure the basic environment in gateway management. (This is the same as the second part of RAM. If it has already been configured, this step can be skipped).

  The only difference is that `auth_type` in `/etc/has/has-client.conf` needs to be modified in LDAP.

  You may also not modify `/etc/has/has-client.conf`. The test user can copy the file, modify `auth_type` with their account, and specify the path through environment variables. For example:

  ```
  export HAS_CONF_DIR=/home/test/has-conf
  ```
- Configure the LDAP administrator user name/password to Kerberos server (HAS) in the E-MapReduce console.

  On the E-MapReduce console, enter Configuration Management > HAS Software, configure the LDAP administrator user name and password in the corresponding `bind_dn` and `bind_password` fields, and restart the HAS service.

  In this example, the LDAP service is the ApacheDS service in the E-MapReduce cluster. Related fields can be obtained from ApacheDS.
- The E-MapReduce cluster administrator adds user information to LDAP.
  - Obtain the administrator user name and password for the ApacheDS LDAP service. `manager_dn` and `manager_password` can be seen in the E-MapReduce console's Configuration Management/ApacheDS Configuration page.

- Add the test user and password to ApacheDS.

```
Log on to root account in the cluster emr-header-1 node
 Create a file test.ldif with the following content:
 dn: cn=test,ou=people,o=emr
 objectclass: inetOrgPerson
 objectclass: organizationalPerson
 objectclass: person
 objectclass: top
 cn: test
 sn: test
 mail: test@example.com
 userpassword: test1234
 #Add to LDAP, in which -w denotes that password is changed to
manager_password
 ldapmodify -x -h localhost -p 10389 -D "uid=admin,ou=system" -w "
Ns1aSe" -a -f test.ldif
 #Delete test.ldif
 rm test.ldif
```

Provide added user name/password to the test user.

- The test user configures the LDAP information.

```
Log on the test account of Gateway
 # Run the script
 sh add_ldap.sh test
```

The add_ldap.sh script is attached to modify the LDAP account information:

```
user=$1
 if [[ `cat /home/$user/.bashrc | grep 'export LDAP_'` == "" ]];then
 echo "
 #Modify to the user test's LDAP_USER/LDAP_PWD
 export LDAP_USER=YOUR_LDAP_USER
 export LDAP_PWD=YOUR_LDAP_USER
 " >>~/.bashrc
 else
    echo $user LDAP user info has been added to .bashrc
 fi
```

- The test user accesses the cluster services.

Execute HDFS commands.

```
[test@iZbp1cyio18s5ymggr7yhrZ ~]$ hadoop fs -ls /
  17/11/19 13:33:33 INFO client.HasClient: The plugin type is: LDAP
  Found 4 items
  drwxr-x---   - has    hadoop          0 2017-11-18 21:12 /apps
  drwxrwxrwt   - hadoop hadoop          0 2017-11-19 13:33 /spark-
history
  drwxrwxrwt   - hadoop hadoop          0 2017-11-19 12:41 /tmp
  drwxrwxrwt   - hadoop hadoop          0 2017-11-19 12:41 /user
```

Run the Hadoop/Spark job.

15.5 Execution plan authentication

E-MapReduce clusters support execution plan authentication. You can authorize Alibaba Cloud Resource Access Management (RAM) user accounts to access execution plans using the master account.

Master account access

After logging on to the E-MapReduce console with the master account, you can run execution plans on the Execution plan page. Submit jobs to the security cluster for execution and access the related open source services involved in the jobs using the Hadoop user name.

RAM user account access

After logging on to the E-MapReduce console with the RAM user account, you can run execution plans on the Execution plan page. Submit jobs to the security cluster for execution and access the related open-source component services involved in the jobs using the user name of the RAM user account.

Examples

- The master account administrator can create multiple RAM user accounts as required and grant them permissions from the RAM console. The RAM users can then log on to the E-MapReduce console and use the related functions.
- The master account administrator provides RAM user accounts to developers.
- After creating jobs and execution plans, developers start running them to submit jobs to the cluster. They can then access the relevant component services in the cluster using the user names that correspond to the RAM user accounts.

  Note:
  Periodic execution plans are currently uniformly executed using the Hadoop account.
- Relevant permission control for component services, such as whether account A is permitted to access a file in HDFS, is performed using the user name of a RAM user
.

15.6 Cross-region access

Kerberos in E-MapReduce supports cross-region access, meaning that different Kerberos clusters can access each other. This section describes cross-region access using cluster A and cluster B as an example.

- Hostname of emr-header-1 in cluster A → emr-header-1.cluster-1234. Region → EMR.1234.COM
- Hostname of emr-header-1 in cluster B → emr-header-1.cluster-6789. Region → EMR.6789.COM

Note:

- The hostname can be obtained by executing the hostname command on emr-header-1.
- The region can be obtained in /etc/krb5.conf on emr-header-1.

Add principal

The emr-header-1 nodes in cluster A and cluster B both run the following command:

```
# root account
       sh /usr/lib/has-current/bin/hadmin-local.sh /etc/ecm/has-conf -
k /etc/ecm/has-conf/admin.keytab
       HadminLocalTool.local: addprinc -pw 123456 krbtgt/EMR.6789.COM@
EMR.1234.COM 6789. COM@EMR. 1234. Com
```

Note:

- The password is 123456. This can be modified.
- The region of cluster B is EMR.6789.COM. This is the region of the cluster being accessed.
- The region of cluster A is EMR.1234.COM. This is the region of the cluster that initiates access.

Configure /etc/krb5.conf for cluster A

Configure [regions]/[domain_region]/[capaths] on cluster A as follows:

```
[libdefaults]
    kdc_realm = EMR. 1234. COM
    default_realm = EMR. 1234. COM
    udp_preference_limit = 4096
    kdc_tcp_port = 88
    kdc_udp_port = 88
    dns_lookup_kdc = false
```

```
[realms]
    EMR. 1234. COM = {
                kdc = 10.81.49.3:88
    }
    EMR. 6789. COM = {
                kdc = 10.81.49.7:88
    }
[domain_realm]
    .cluster-1234 = EMR. 1234. COM
    .cluster-6789 = EMR. 6789. COM
[capaths]
    EMR. 1234. COM = {
       EMR. 6789. COM = .
    }
    EMR. 6789. COM = {
       EMR. 1234. COM = .
    }
```

Synchronize /etc/krb5.conf to all cluster A nodes.

Copy the binding information (only the long domain name emr-xxx-x.cluster-xxx is required) from cluster B's /etc/hosts file to /etc/hosts for all cluster A nodes.

```
10.81.45.89  emr-worker-1.cluster-xxx
 10.81.46.222  emr-worker-2.cluster-xx
 10.81.44.177  emr-header-1.cluster-xxx
```

Note:

- If you want to run a job on cluster A to access cluster B, you must first restart YARN.
- Configure cluster B's host binding information for all cluster A nodes.

Access services in cluster B

You can use cluster A's Kerberos keytab file /ticket as a cache on cluster A to access services in cluster B.

For example, access the HDFS service in cluster B as follows:

```
su has;
hadoop fs -ls hdfs://emr-header-1.cluster-6789:9000/
Found 4 items
-rw-r-----   2 has    hadoop         34 2017-12-05 18:15 hdfs://emr-
header-1.cluster-6789:9000/abc
drwxrwxrwt   - hadoop hadoop          0 2017-12-05 18:32 hdfs://emr-
header-1.cluster-6789:9000/spark-history
drwxrwxrwt   - hadoop hadoop          0 2017-12-05 17:53 hdfs://emr-
header-1.cluster-6789:9000/tmp
drwxrwxrwt   - hadoop hadoop          0 2017-12-05 18:24 hdfs://emr-
header-1.cluster-6789:9000/user
```

16 Component authorization

16.1 HDFS authorization

After HDFS has been enabled, you need permission to access it in order to perform operations, such as read data or create folders.

Add a configuration

The configurations related to HDFS permission are as follows:

- dfs.permissions.enabled

  Enable permission check. Even if the value is false, chmod/chgrp/chown/setfacl performs a permission check.
- dfs.datanode.data.dir.perm

  The permission of the local folder used by datanode, which is 755 by default.
- fs.permissions.umask-mode
  - Permission mask (default permission settings when creating a new file/folder)
  - File creation: 0666 & ^umask
  - Folder creation: 0777 & ^umask
  - Default umask value is 022. This means that the permission for file creation is 644 (666&^022 = 644), and permission of folder creation is 755 (777&^022 = 755).
  - The default setting of the Kerberos security cluster in E-MapReduce is 027. The permission for file creation is 640. For folder creation, it is 750.
- dfs.namenode.acls.enabled
  - Enable ACL control. This gives you permission control for owners/groups, and you can also set it for other users.
  - Commands for setting ACL:

    ```
    hadoop fs -getfacl [-R] <path>
    hadoop fs -setfacl [-R] [-b |-k -m |-x <acl_spec> <path>] |[--set
     <acl_spec>   <path>]
    ```

    For example:

    ```
    su test
     #The user test creates a folder
     hadoop fs -mkdir /tmp/test
     #View the permission of the created folder
    ```

```
 hadoop fs -ls /tmp
 drwxr-x---   - test   hadoop          0 2017-11-26 21:18 /tmp/
test
 #Set ACL and grant rwx permissions to user foo
 hadoop fs -setfacl -m user:foo:rwx /tmp/test
 #View the permission of the file (+ means that ACL is set)
 hadoop fs -ls  /tmp/
 drwxrwx---+  - test   hadoop          0 2017-11-26 21:18 /tmp/
test
 #View ACL
  hadoop fs -getfacl  /tmp/test
  # file: /tmp/test
# owner: test
# group: hadoop
user::rwx
user:foo:rwx
group::r-x
mask::rwx
other::---
```

- dfs.permissions.superusergroup

  Super user group. Users in this group have super user permissions.

Restart the HDFS service

For Kerberos security clusters, HDFS permissions have been set by default (umask is set to 027). Configuration and service restart are not necessary.

For non-Kerberos security clusters, a configuration must be added and the service must be restarted.

Other

- The umask value can be modified as required.
- HDFS is a basic service, and Hive/HBase are based on HDFS. Therefore, HDFS permission control must be configured in advance when configuring other upper-layer services.
- When permissions are enabled for HDFS, the services must be set up (such as /spark-history for Spark and /tmp/$user/ for YARN).
- Sticky bit:

  Sticky bit can be set for a folder to prevent anyone other than super user/file owner/directory owner from deleting files or folders in the folder (even if other users have rwx permissions for that folder). For example:

  ```
  #That is, adding numeral 1 as the first digit
  hadoop fs -chmod 1777 /tmp
  hadoop fs -chmod 1777 /spark-history
  ```

```
hadoop fs -chmod 1777 /user/hive/warehouse
```

16.2 YARN authorization

YARN authorization can be divided into service-level and queue-level authorization.

Service-level authorization

For more information, see Hadoop's *Service Level Authorization Guide*.

- Controls users' access to cluster services, such as job submission.
- Configures hadoop-policy.xml.
- Service-level permission checks are performed before other permission checks (such as for HDFS permission and YARN job submission)

Note:

Typically, if HDFS permission checks and YARN job submission controls have been set up, you may choose not to set the service-level permission control. Perform the relevant configurations as required.

Queue-level authorization

YARN supports permission control for resources by means of queues, and provides two queue scheduling methods: Capacity Scheduler and Fair Scheduler. Capacity Scheduler is used as an example here.

- Add a configuration

  A queue also has two levels of authorization: for job submission (submitting a job to the queue) and for queue management.

  Note:

  - The ACL control object for a queue is a user or group. When you are defining the related parameters, users and groups can be set at the same time, separated by spaces. You can use a comma to separate different users or groups. Only one space indicates that no one has permission.
  - ACL inheritance for a queue: If a user or group can submit an application to a queue, they can also submit applications to all of its sub-queues. The ACL that manages queues can also be inherited. If you want to prevent a user or group

from submitting jobs to a queue, you must set the ACL for this queue and all its parent queues to restrict the job submission permission for this user or group.

- yarn.acl.enable

  Set the ACL switch to true.
- yarn.admin.acl
  - The YARN administrator setting, which enables or disables the execution of `yarn rmadmin/yarn kill` and other commands. This value must be configured. If not, the subsequent queue-based ACL administrator settings do not take effect.
  - You can set the user or group when setting the parameter values:

    ```
    user1,user2 group1,group2 #users and groups are separated by a
    space
      group1,group2 #In case there are only groups, a leading space
     is required.
    ```

    In an E-MapReduce cluster, you must configure the ACL permission for `has` as administrator.
- yarn.scheduler.capacity.${queue-name}.acl_submit_applications
  - Set the user or group that can submit jobs to this queue.
  - If ${queue-name} is the queue name, multi-level queues are supported. Note that ACL is inherited in multi-level queues. For example:

    ```
    #queue-name=root
      <property>
          <name>yarn.scheduler.capacity.root.acl_submit_applicati
    ons</name>
          <value> </value> # Space means no one can submit jobs to
    the root queue
      </property>
     #queue-name=root.testqueue
     <property>
       <name>yarn.scheduler.capacity.root.testqueue.acl_submit
    _applications</name>
          <value>test testgrp</value> #testqueue only allows the
    test user and testgrp group to submit jobs
      </property>
    ```
- yarn.scheduler.capacity.${queue-name}.acl_administer_queue
  - Set some users or groups to manage the queue, such as killing jobs in the queue.

- Multi-level queue names are supported. Note that ACL is inherited in multi-level queues.

```
#queue-name=root
  <property>
      <name>yarn.scheduler.capacity.root.acl_administer_queue</
name>
      <value> </value>
  </property>
 #queue-name=root.testqueue
 <property>
   <name>yarn.scheduler.capacity.root.testqueue.acl_admini
ster_queue</name>
      <value>test testgrp</value>
  </property>
```

- Restart the YARN service
  - Kerberos secure clusters have ACL enabled by default. You can configure the relevant ACL permissions for queues as required.
  - For non-Kerberos secure clusters, enable ACL and configure the permission control for queues in accordance with the preceding instructions. Then restart the YARN service.
- Configuration example
  - yarn-site.xml

```
<property>
        <name>yarn.acl.enable</name>
        <value>true</value>
 </property>
<property>
        <name>yarn.admin.acl</name>
        <value>has</value>
 </property>
```

  - capacity-scheduler.xml
  - Default queue: Disables the default queue and does not allow users to submit jobs or manage the queue.
  - Q1 queue: Only allows the test user to submit jobs and manage the queue (such as killing jobs).
  - Q2 queue: Only allows the foo user to submit jobs and manage the queue.

```
<configuration>
    <property>
        <name>yarn.scheduler.capacity.maximum-applications</name>
        <value>10000</value>
        <description>Maximum number of applications that can be
pending and running.</description>
    </property>
    <property>
```

```
        <name>yarn.scheduler.capacity.maximum-am-resource-percent</
name>
        <value>0.25</value>
        <description>Maximum percent of resources in the cluster
which can be used to run application masters i.e.
            controls number of concurrent running applications.
        </description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.resource-calculator</name>
        <value>org.apache.hadoop.yarn.util.resource.DefaultRes
ourceCalculator</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.queues</name>
        <value>default,q1,q2</value>
        <! --3 queues->
        <description>The queues at the this level (root is the root
queue).</description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.default.capacity</name>
        <value>0</value>
        <description>Default queue target capacity.</description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.default.user-limit-factor
</name>
        <value>1</value>
        <description>Default queue user limit a percentage from 0.0
to 1.0.</description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.default.maximum-capacity
</name>
        <value>100</value>
        <description>The maximum capacity of the default queue.</
description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.default.state</name>
        <value>STOPPED</value>
        <! -- Status of the default queue is set as STOPPED-->
        <description>The state of the default queue. State can be
one of RUNNING or STOPPED.</description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.default.acl_submit
_applications</name>
        <value> </value>
        <! -- The default queue does not allow job submission-->
        <description>The ACL of who can submit jobs to the default
queue.</description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.default.acl_admini
ster_queue</name>
        <value> </value>
        <! -- Prevent users/groups to manage the default queue-->
        <description>The ACL of who can administer jobs on the
default queue.</description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.node-locality-delay</name>
        <value>40</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.queue-mappings</name>
        <value>u:test:q1,u:foo:q2</value>
        <! -- Queue mapping, automatically maps the test user to Q1
queue-->
        <description>A list of mappings that will be used to assign
jobs to queues. The syntax for this list is
            [u|g]:[name]:[queue_name][,next mapping]* Typically this
 list will be used to map users to queues,for
            example, u:%user:%user maps all users to queues with the
 same name as the user.
        </description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.queue-mappings-override.enable
</name>
        <value>true</value>
        <! -- Whether or not allow the above queue-mapping to
overwrite the queue parameters set up by the client-->
        <description>If a queue mapping is present, will it override
 the value specified by the user? This can be used
            by administrators to place jobs in queues that are
different than the one specified by the user. The default
            is false.
        </description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.acl_submit_applications</
name>
        <value> </value>
        <! -- ACL inheritance, the parent queue must have the admin
permissions-->
        <description>
            The ACL of who can submit jobs to the root queue.
        </description>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q1.acl_submit_applicati
ons</name>
        <value>test</value>
        <! -- q1 only allows the test user to submit jobs-->
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q2.acl_submit_applicati
ons</name>
        <value>foo</value>
        <! -- q2 only allows the foo user to submit jobs-->
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q1.maximum-capacity</name
>
        <value>100</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q2.maximum-capacity</name
>
        <value>100</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q1.capacity</name>
        <value>50</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q2.capacity</name>
        <value>50</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.acl_administer_queue</
name>
        <value> </value>
        <! -- ACL inheritance, the parent queue must have the admin
permissions-->
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q1.acl_administer_queue</
name>
        <value>test</value>
        <! -- q1 only allow the test user to manage the queue, such
as killing the jobs-->
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q2.acl_administer_queue</
name>
        <value>foo</value>
        <! -- q2 only allow the foo user to manage the queue, such
as killing the jobs-->
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q1.state</name>
        <value>RUNNING</value>
    </property>
    <property>
        <name>yarn.scheduler.capacity.root.q2.state</name>
        <value>RUNNING</value>
    </property>
</configuration>
```

16.3 Hive authorization

Hive has two authorization modes: one based on storage and the other based on SQL standards. For more information, see Hive's *Authorization guide*.

Note:

Both means of authorization can be configured at the same time without conflict.

Storage based authorization (for Hive metastore)

If a user in a cluster has direct access to data in Hive through an HDFS or Hive client, a permission control must be performed on Hive data in HDFS. By doing so, operation permissions related to Hive SQL can be controlled.

For more information, see Hive's *Storage Based Authorization* guide.

Add configuration

In the cluster Configuration Management page, click Hive > Configuration > hive-site.xml > Add Custom Configuration.

```
<property>
<name>hive.metastore.pre.event.listeners</name>
    <value>org.apache.hadoop.hive.ql.security.authorization.Authorizat
ionPreEventListener</value>
</property>
<property>
<name>hive.security.metastore.authorization.manager</name>
    <value>org.apache.hadoop.hive.ql.security.authorization.StorageBas
edAuthorizationProvider</value>
</property>
<property>
<name>hive.security.metastore.authenticator.manager</name>
    <value>org.apache.hadoop.hive.ql.security.HadoopDefaultMetasto
reAuthenticator</value>
</property>
```

Restart Hive metastore

Restart the Hive metastore in the cluster's Configuration Management page.

HDFS permission control

For Kerberos security clusters in E-MapReduce, HDFS permissions for the Hive warehouse are set.

For non-Kerberos security clusters, you must complete the following steps to set the basic HDFS permission:

- Enable HDFS permissions
- Configure permissions for the Hive warehouse

  ```
  hadoop fs -chmod 1771 /user/hive/warehouse
  It can be set as follows, in which 1 denotes stick bit (i.e. cannot
  delete files/folders created by others)
  hadoop fs -chmod 1777 /user/hive/warehouse
  ```

With the basic permission set, users and user groups can create, read, and write tables as usual by authorizing the folder warehouse.

```
sudo su has
      #Grant rwx permission of folder warehouse to user test
      hadoop fs -setfacl -m user:test:rwx /user/hive/warehouse
      #Grant rwx permission of folder warehouse to user hivegrp
```

```
       hadoo fs -setfacl -m group:hivegrp:rwx /user/hive/warehouse
```

With HDFS authorized, users and user groups can create, read, and write tables as usual. Data in Hive tables that is created by different users in HDFS can only be accessed by the users themselves.

Verification

- The test user creates a table testtbl.

```
hive> create table testtbl(a string);
FAILED: Execution Error, return code 1 from org.apache.hadoop.hive
.ql.exec.DDLTask. MetaException(message:Got exception: org.apache.
hadoop.security.AccessControlException Permission denied: user=test
, access=WRITE, inode="/user/hive/warehouse/testtbl":hadoop:hadoop:
drwxrwx--t
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(
FSPermissionChecker.java:320)
at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.check(
FSPermissionChecker.java:292)
```

An error occurs due to the lack of permissions. Permissions should be granted to the test user.

```
#Switch from root account to has account
su has
#Add ACL and grant rwx permissions of the directory warehouse to the
 account test.
 hadoop fs -setfacl -m user:test:rwx /user/hive/warehouse
```

The test account recreates the database successfully.

```
hive> create table testtbl(a string);
OK
Time taken: 1.371 seconds
#View the directory of testtbl in HDFS. From the permissions it can
be seen that only the groups test and hadoop can read data from the
table created by the user test, while other users have no permission
s
hadoop fs -ls /user/hive/warehouse
drwxr-x---   - test hadoop          0 2017-11-25 14:51 /user/hive/
warehouse/testtbl
#Insert a record
hive>insert into table testtbl select "hz"
```

- User foo accesses the table testtbl.

```
#drop table
hive> drop table testtbl;
FAILED: Execution Error, return code 1 from org.apache.hadoop.hive.
ql.exec.DDLTask. MetaException(message:Permission denied: user=foo,
access=READ, inode="/user/hive/warehouse/testtbl":test:hadoop:drwxr-
x---
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
check(FSPermissionChecker.java:320)
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
checkPermission(FSPermissionChecker.java:219)
```

```
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
checkPermission(FSPermissionChecker.java:190)
#alter table
hive> alter table testtbl add columns(b string);
FAILED: SemanticException Unable to fetch table testtbl. java.
security.AccessControlException: Permission denied: user=foo, access
=READ, inode="/user/hive/warehouse/testtbl":test:hadoop:drwxr-x---
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
check(FSPermissionChecker.java:320)
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
checkPermission(FSPermissionChecker.java:219)
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
checkPermission(FSPermissionChecker.java:190)
    at org.apache.hadoop.hdfs.server.namenode.FSDirectory.checkPermi
ssion(FSDirectory.java:1720)
#select
hive> select * from testtbl;
FAILED: SemanticException Unable to fetch table testtbl. java.
security.AccessControlException: Permission denied: user=foo, access
=READ, inode="/user/hive/warehouse/testtbl":test:hadoop:drwxr-x---
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
check(FSPermissionChecker.java:320)
    at org.apache.hadoop.hdfs.server.namenode.FSPermissionChecker.
checkPermission(FSPermissionChecker.java:219)
```

Foo cannot perform operations on the table created by the test user. HDFS authorization is needed to grant permissions to foo.

```
su has
#Only read permission is granted, and write permission can also be
granted as needed (for example, alter)
#Note: -R: Set files in the folder testtbl to readable
hadoop fs -setfacl -R -m user:foo:r-x /user/hive/warehouse/testtbl
#The table can be selected successfully
hive> select * from testtbl;
OK
hz
Time taken: 2.134 seconds, Fetched: 1 row(s)
```

Note:

You can create a Hive user group, authorize it, and then add new users it.

SQL Standard Based Authorization

- Scenario

  If a cluster user cannot access data in Hive through an HDFS or Hive client, and the only way is to run Hive related commands through `HiveServer (beeline, jdbc, and so on)`, SQL standard based authorization can be used.

  If users can use the Hive shell or similar methods and as long as hive-site.xml in the user's client has not been configured, Hive can still be accessed as usual, even if the following settings are implemented.

For more information, see Hive's *SQL Standard Based Authorization* guide.

- Add configuration

  - The configuration is provided to HiveServer.
  - In the cluster Configuration Management page, click Hive > Configuration > hive-site.xml > Add Custom Configuration.

```
<property>
<name>hive.security.authorization.enabled</name>
 <value>true</value>
</property>
 <property>
<name>hive.users.in.admin.role</name>
 <value>hive</value>
</property>
 <property>
<name>hive.security.authorization.createtable.owner.grants</name>
 <value>ALL</value>
</property>
```

- Restart HiveServer2

  Restart HHiveServer2 in the cluster Configuration Management page.

- Permission operation commands

  For more information on command operations, click *here*.

- Verification

  - User foo accesses the test user's table, testtbl, through beeline.

```
2: jdbc:hive2://emr-header-1.cluster-xxx:10> select * from testtbl
;
Error: Error while compiling statement: FAILED: HiveAccess
ControlException Permission denied: Principal [name=foo, type=USER
] does not have following privileges for operation QUERY [[SELECT
] on Object [type=TABLE_OR_VIEW, name=default.testtbl]] (state=
42000,code=40000)
```

  - Grant permissions.

```
Switch to account test to grant select permission to user foo
hive> grant select on table testtbl to user foo;
 OK
 Time taken: 1.205 seconds
```

  - Foo can select as usual.

```
0: jdbc:hive2://emr-header-1.cluster-xxxxx:10> select * from
testtbl;
INFO  : OK
+------------+---+
| testtbl.a  |
+------------+---+
| hz         |
+-------------+--+
1 row selected (0.787 seconds)
```

- **Revoke permissions.**

```
Switch to account test, and revoke the select permission from user
 foo
  hive> revoke select from user foo;
    OK
    Time taken: 1.094 seconds
```

- **Foo cannot select testtbl data.**

```
User foo cannot select data from table testtbl.
Error: Error while compiling statement: FAILED: HiveAccess
ControlException Permission denied: Principal [name=foo, type=USER
] does not have following privileges for operation QUERY [[SELECT
] on Object [type=TABLE_OR_VIEW, name=default.testtbl]] (state=
42000,code=40000)
```

16.4 HBase authorization

Any account can perform any operation on an HBase cluster without being authorized, including disabling and dropping tables and performing major compaction.

Note:

For clusters that do not have Kerberos authentication, users can forge identities to access the cluster service, even when HBase authorization is enabled. Therefore, we recommend that you create a high-security cluster (for example, supporting Kerberos) as detailed in the *Introduction to Kerberos*.

Add configuration

In the Configuration Management page, choose HBase > Configuration > hbase-site > Custom Configuration in the HBase cluster.

Add the following parameters:

```
<property>
     <name>hbase.security.authorization</name>
     <value>true</value>
</property>
<property>
     <name>hbase.coprocessor.master.classes</name>
     <value>org.apache.hadoop.hbase.security.access.AccessController</
value>
</property>
<property>
     <name>hbase.coprocessor.region.classes</name>
```

```
 <value>org.apache.hadoop.hbase.security.token.TokenProvider,org.
apache.hadoop.hbase.security.access.AccessController</value>
</property>
<property>
  <name>hbase.coprocessor.regionserver.classes</name>
  <value>org.apache.hadoop.hbase.security.access.AccessController,org.
apache.hadoop.hbase.security.token.TokenProvider</value>
</property>
```

Restart the HBase cluster

In the HBase cluster's Configuration Management page, click HBase > Operations > RESTART All Components.

Authorization (ACL)

- Basic concepts

  In HBase, authorization consists of three elements: the granting of operational permissions for a certain scope of resources to a certain entity.

  - Resources in a certain scope

    - Superuser

      A superuser can perform any operations. The account that runs the HBase service is the superuser by default. You can also add superusers by configuring the value of hbase.superuser in hbase-site.xml.

    - Global

      Global Scope has administrator permissions for all tables in the cluster.

    - Namespace

      This has permission control in Namespace Scope.

    - Table

      This has permission control in Table Scope.

    - ColumnFamily

      This has permission control in ColumnFamily Scope.

    - Cell

      This has permission control in Cell Scope.

  - Operational permissions

    - Read (R)

      Read data from resources in a certain scope.

- Write (W)

  Write data to resources in a certain scope.
- Execute (X)

  Execute co-processor in a certain scope.
- Create (C)

  Create or delete a table in a certain scope.
- Admin (A)

  Perform cluster-related operations in a certain scope, such as balance or assign.

- Entity
  - User

    Authorize a user.
  - Group

    Authorize a user group.

- Authorization command
  - grant

    ```
    grant <user> <permissions> [<@namespace> [<table> [<column family>
     [<column qualifier>]]]
    ```

    Note:

    - The authorization methods for users and groups are the same. The prefix @ needs to be added for groups.

      ```
      grant 'test','R','tbl1'   #grant the read permission of the
      table tb11 to the user test.
        grant '@test','R','tbl1'   #grant the read permission of the
      table tb11 to the user group test.
      ```
    - The prefix @ needs to be added for namespace.

      ```
      grant 'test 'C','@ns_1'  #grant the create permission of the
      namespace @ns_1 to the user test.
      ```
  - revoke
  - user_permissions (view permissions)

16.5 Kafka authorization

If Kafka authentication (for example, Kerberos authentication or another simple authorization based on a user name and password) is disabled, users can access services with forged identities, even if Kafka authorization is enabled. Therefore, we recommend that you create a high-security Kafka cluster. For more information, see *Introduction to Kerberos*.

Note:

The permission configurations detailed in this section are for high-security E-MapReduce clusters only (Kafka is started in Kerberos).

Add configurations

1. On the Cluster Management page, click View Details next to the Kafka cluster.
2. In the navigation pane on the left, click the Clusters and Services tab, and click Kafka in the service list.
3. At the top of the page, click the Configuration tab.
4. In the upper-right corner of the Service Configuration list, click Custom Configuration and add the following parameters:

Key	Value	Description
authorizer.class.name	kafka.security.auth.SimpleAclAuthorizer	N/A
super.users	User:kafka	User:kafka is required. Other users can be added and separated by semicolons (;).

Note:

zookeeper.set.acl is used to set the permissions for Kafka to operate data in ZooKeeper. It is already set to true in the E-MapReduce cluster, so you do not need to add this configuration here. With the configuration set to true, only users named Kafka who have passed the Kerberos authentication can run the kafka-topics.sh command in the Kerberos environment. Kafka-topics.sh can read, write, and modify data in ZooKeeper.

Restart a Kafka cluster

1. On the Cluster Management page, click View Details next to the Kafka cluster you want to operate in the Operation column.
2. In the navigation pane on the left, click the Clusters and Services tab, and click Actions to the right of Kafka on the service list.
3. In the drop-down menu, select RESTART All Components. Enter the record information and click OK.

Authorization (ACL)

- Basic concepts

  Definition in official Kafka documentation:

  ```
  Kafka ACLs are defined in the general format of "Principal P is [
  Allowed/Denied] Operation O From Host H On Resource R"
  ```

  This indicates that the ACL process relates to Principal, Allowed/Denied, Operation Host, and Resource.

  - Principal: username

Security protocol	Value
PLAINTEXT	ANONYMOUS
SSL	ANONYMOUS
SASL_PLAINTEXT	If the mechanism is PLAIN, the user name is specified by client_jaas.conf. If the mechanism is GSSAPI, the user name is principal specified by client_jaas.conf.
SASL_SSL	

  - Allowed/Denied
  - Operation: Operations include Read, Write, Create, DeleteAlter, Describe, ClusterAction, AlterConfigs, DescribeConfigs, IdempotentWrite, and All.
  - Host: The target machine.
  - Resource: Resource objects, including Topic, Group, Cluster, and TransactionalId.

  For detailed mapping relationships between operations and resources, such as the supporting relationships between resources and the authorization of operations, see *KIP-11 - Authorization Interface*.

- Authorization command

Perform authorization using the kafka-acls.sh script (/usr/lib/kafka-current/bin/kafka-acls.sh). For more information about how to use this script to authorize Kafka, run the `kafka-acls.sh --help` command.

Procedure

Complete the following operations on the master node of the high-security Kafka cluster you created in E-MapReduce.

1. Create a user named test.

   ```
   useradd test
   ```

2. Create a topic.

   zookeeper.set.acl is set to true, and kafka-topics.sh must be run under a Kafka account. The Kafka account must pass Kerberos authentication.

   ```
   #The Kerberos authentication information related to the kafka
   account is set in kafka_client_jaas.conf.
   export KAFKA_HEAP_OPTS="-Djava.security.auth.login.config=/etc/ecm/
   kafka-conf/kafka_client_jaas.conf"
   # Change the ZooKeeper address to the actual address (run hostnamed
   to acquire) of your Kafka cluster.
   kafka-topics.sh --create --zookeeper emr-header-1:2181/kafka-1.0.0
    --replication-factor 3 --partitions 1 --topic test
   ```

3. Run kafka-console-producer.sh with the test user.

   a. Create a keytab file for the test user to authenticate ZooKeeper and Kafka.

   ```
   su root
   sh /usr/lib/has-current/bin/hadmin-local.sh /etc/ecm/has-conf -k /
   etc/ecm/has-conf/admin.keytab
   HadminLocalTool.local: # Press Enter to display the usage
   instructions on some commands.
   HadminLocalTool.local: addprinc # Enter a command and press Enter
   to display the usage instructions on the command.
   HadminLocalTool.local: addprinc -pw 123456 test # Add a principal
   for the test user and set the password to 123456.
   HadminLocalTool.local: ktadd -k /home/test/test.keytab test #
   Export the keytab file for later use.
   ```

   b. Add a kafka_client_test.conf file.

   Put the file in */home/test/kafka_client_test.conf*. The content of the file is as follows:

   ```
   KafkaClient {
   com.sun.security.auth.module.Krb5LoginModule required
   useKeyTab=true
   storeKey=true
   serviceName="kafka"
   keyTab="/home/test/test.keytab"
   ```

```
principal="test";
};
// Zookeeper client authentication
Client {
com.sun.security.auth.module.Krb5LoginModule required
useKeyTab=true
useTicketCache=false
serviceName="zookeeper"
keyTab="/home/test/test.keytab"
principal="test";
};
```

c. **Add producer.conf.**

Put the file in `/home/test/producer.conf`. The content of the file is as follows:

```
security.protocol=SASL_PLAINTEXT
sasl.mechanism=GSSAPI
```

d. **Run kafka-console-producer.sh.**

```
su test
export KAFKA_HEAP_OPTS="-Djava.security.auth.login.config=/home/
test/kafka_client_test.conf"
kafka-console-producer.sh --producer.config /home/test/producer.
conf --topic test --broker-list emr-worker-1:9092
```

Because no ACL is set, an error is reported after the preceding command is run:

```
org.apache.kafka.common.errors.TopicAuthorizationException: Not
authorized to access topics: [test]
```

e. **Set an ACL.**

Similarly, the `kafka-acls.sh` command must be run under the Kafka account.

```
su kafka
export KAFKA_OPTS="-Djava.security.auth.login.config=/etc/ecm/
kafka-conf/kafka_client_jaas.conf"
kafka-acls.sh --authorizer-properties zookeeper.connect=emr-header
-1:2181/kafka-1.0.0 --add --allow-principal User:test --operation
Write --topic test
```

f. **Run kafka-console-producer.sh again.**

```
su test
export KAFKA_HEAP_OPTS="-Djava.security.auth.login.config=/home/
test/kafka_client_test.conf"
kafka-console-producer.sh --producer.config /home/test/producer.
conf --topic test --broker-list emr-worker-1:9092
```

Normal output is as follows:

```
[2018-02-28 22:25:36,178] INFO Kafka commitId : aaa7af6d4a11b29d (
org.apache.kafka.common.utils.AppInfoParser)
>alibaba
>E-MapReduce
```

```
>
```

4. Run `kafka-console-consumer.sh` with the test user.

   After kafka-console-producer.sh is successfully run and data is written to the topic, you can run `kafka-console-consumer.sh` to perform a consumption test.

   a. Add consumer.conf.

      Put the file in */home/test/consumer.conf*. The content of the file is as follows:

      ```
      security.protocol=SASL_PLAINTEXT
      sasl.mechanism=GSSAPI
      ```

   b. Run kafka-console-consumer.sh.

      ```
      su test
      # Kafka_client_test.conf is used in the same way as kafka-console-producer.sh.
      export KAFKA_HEAP_OPTS="-Djava.security.auth.login.config=/home/test/kafka_client_test.conf"
      kafka-console-consumer.sh --consumer.config consumer.conf  --topic test --bootstrap-server emr-worker-1:9092 --group test-group  --from-beginning
      ```

      Because no permissions are set, an error is reported:

      ```
      org.apache.kafka.common.errors.GroupAuthorizationException: Not authorized to access group: test-group
      ```

   c. Set an ACL.

      ```
      su kafka
      export KAFKA_HEAP_OPTS="-Djava.security.auth.login.config=/etc/ecm/kafka-conf/kafka_client_jaas.conf"
      # test-group permission
      kafka-acls.sh --authorizer-properties zookeeper.connect=emr-header-1:2181/kafka-1.0.0 --add --allow-principal User:test --operation Read --group test-group
      # topic permission
      kafka-acls.sh --authorizer-properties zookeeper.connect=emr-header-1:2181/kafka-1.0.0 --add --allow-principal User:test --operation Read --topic test
      ```

   d. Run `kafka-console-consumer.sh` again.

      ```
      su test
      # Kafka_client_test.conf is used in the same way as kafka-console-producer.sh.
      export KAFKA_HEAP_OPTS="-Djava.security.auth.login.config=/home/test/kafka_client_test.conf"
      ```

```
kafka-console-consumer.sh --consumer.config consumer.conf  --topic
 test --bootstrap-server emr-worker-1:9092 --group test-group  --
from-beginning
```

Normal output is as follows:

```
alibaba
E-MapReduce
```

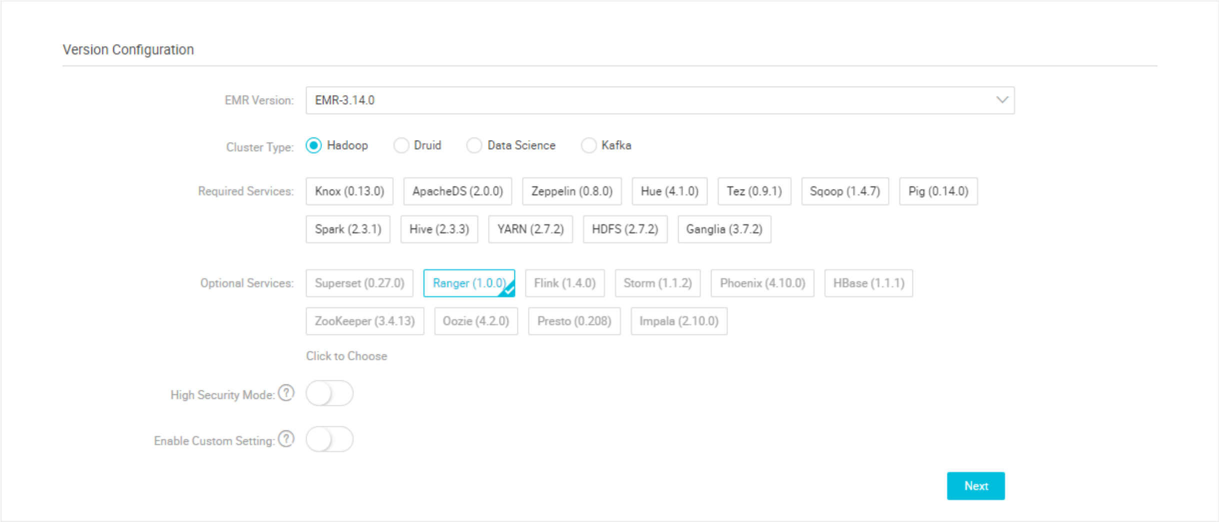
16.6 Ranger

16.6.1 Introduction to Ranger

Apache Ranger provides a centralized framework for permission management, implementing fine-grained access control for components in the Hadoop ecosystem, such as HDFS, Hive, YARN, Kafka, Storm, and Solr. It also provides a UI that allows administrators to perform operations more conveniently.

Create a cluster

Select the Ranger service when you create a cluster in E-MapReduce 2.9.2/3.9.0 or later on the E-MapReduce console.



Version Configuration

EMR Version: EMR-3.14.0

Cluster Type: ● Hadoop ○ Druid ○ Data Science ○ Kafka

Required Services: Knox (0.13.0) ApacheDS (2.0.0) Zeppelin (0.8.0) Hue (4.1.0) Tez (0.9.1) Sqoop (1.4.7) Pig (0.14.0) Spark (2.3.1) Hive (2.3.3) YARN (2.7.2) HDFS (2.7.2) Ganglia (3.7.2)

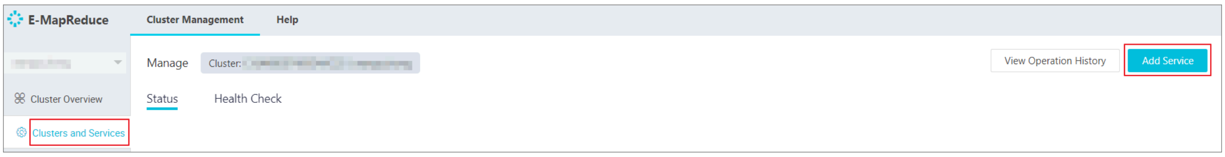
Optional Services: Superset (0.27.0) Ranger (1.0.0) ✓ Flink (1.4.0) Storm (1.1.2) Phoenix (4.10.0) HBase (1.1.1) ZooKeeper (3.4.13) Oozie (4.2.0) Presto (0.208) Impala (2.10.0)

Click to Choose

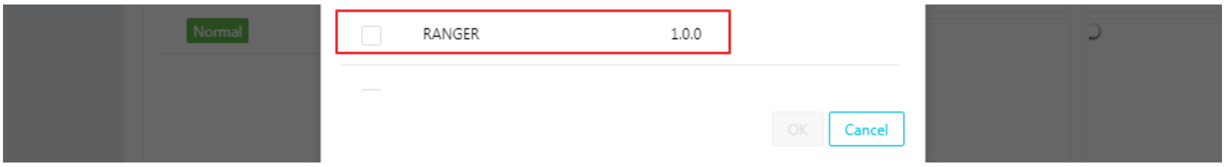
High Security Mode: ⓘ ○

Enable Custom Setting: ⓘ ○

If an E-MapReduce cluster 2.9.2/3.9.0 or later has been created without Ranger, you can go to the Clusters and Services page to add it.



Manage Cluster: [redacted]

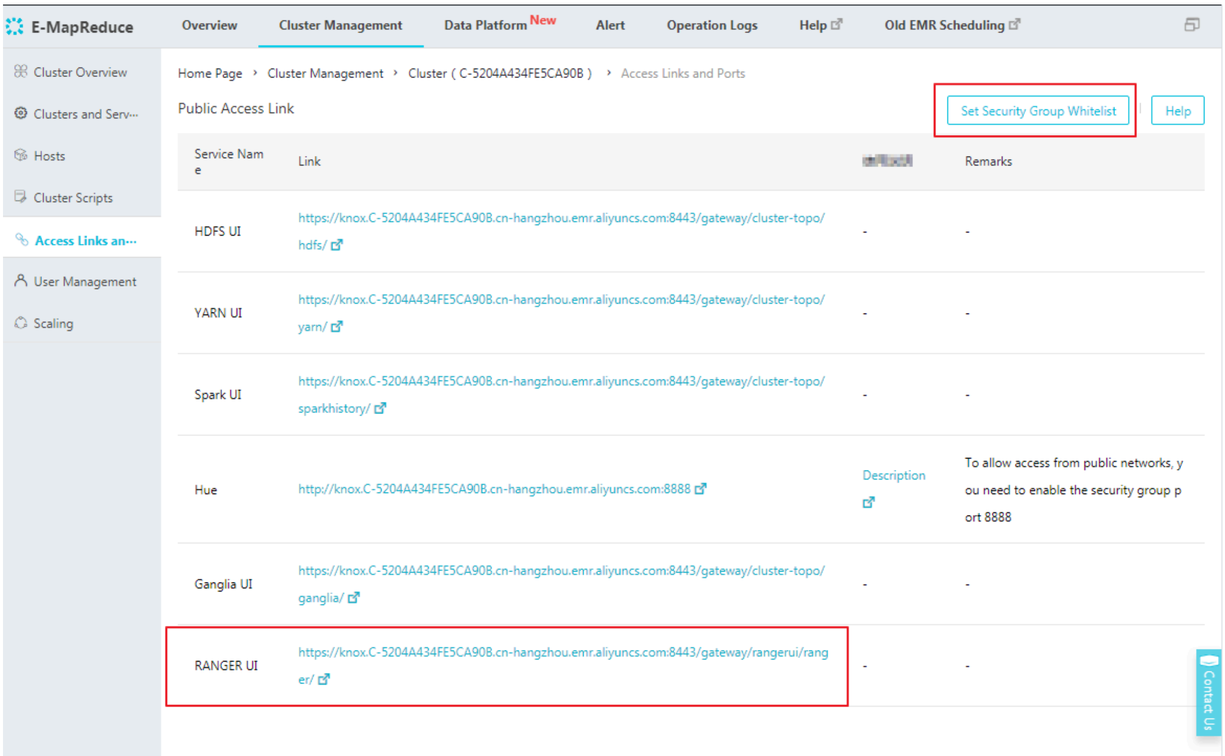


Note:

- When Ranger is enabled, there is no impact or limitation on the application until the security control policy is set.
- The user policy set in Ranger is the cluster Hadoop account.

Ranger UI

After installing Ranger on the cluster, click Manage in the Actions column, and then click Access Links and Ports in the navigation pane on the left. You can then access the Ranger UI by clicking on the link, as shown in the following figure.

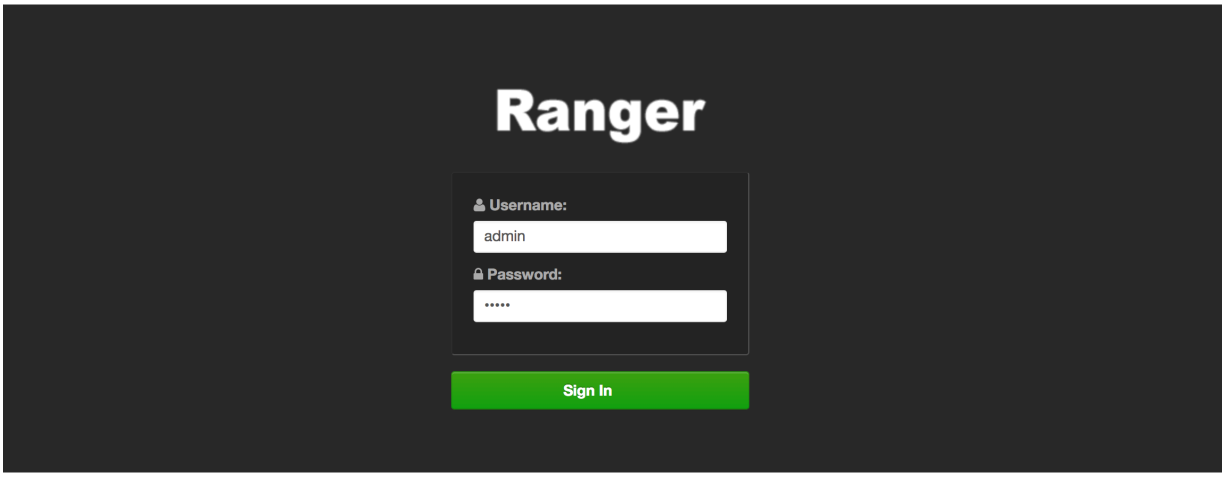


Home Page › Cluster Management › Cluster (C-5204A434FE5CA90B) › Access Links and Ports

Public Access Link

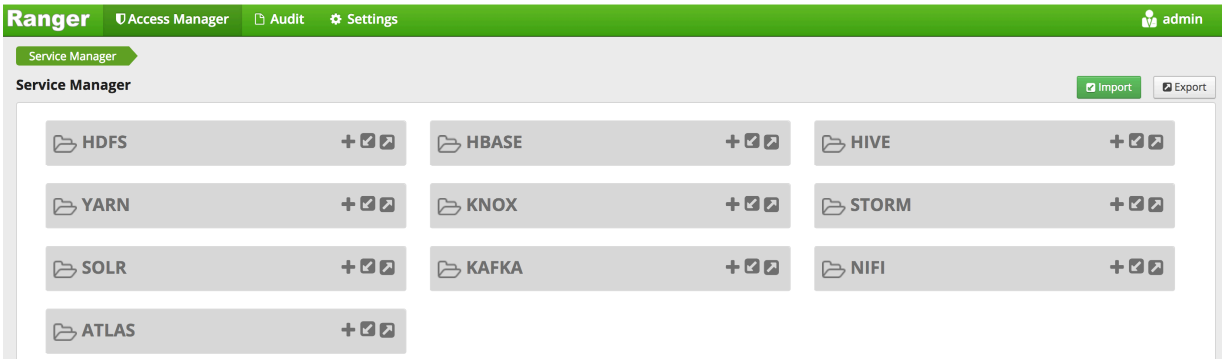
Service Name	Link	[redacted]	Remarks
HDFS UI	https://knox.C-5204A434FE5CA90B.cn-hangzhou.emr.aliyuncs.com:8443/gateway/cluster-topo/hdfs/	-	-
YARN UI	https://knox.C-5204A434FE5CA90B.cn-hangzhou.emr.aliyuncs.com:8443/gateway/cluster-topo/yarn/	-	-
Spark UI	https://knox.C-5204A434FE5CA90B.cn-hangzhou.emr.aliyuncs.com:8443/gateway/cluster-topo/sparkhistory/	-	-
Hue	http://knox.C-5204A434FE5CA90B.cn-hangzhou.emr.aliyuncs.com:8888	Description	To allow access from public networks, you need to enable the security group port 8888
Ganglia UI	https://knox.C-5204A434FE5CA90B.cn-hangzhou.emr.aliyuncs.com:8443/gateway/cluster-topo/ganglia/	-	-
RANGER UI	https://knox.C-5204A434FE5CA90B.cn-hangzhou.emr.aliyuncs.com:8443/gateway/rangerui/ranger/	-	-

Enter the Ranger UI. The default user name and password are both admin, as shown in the following figure.



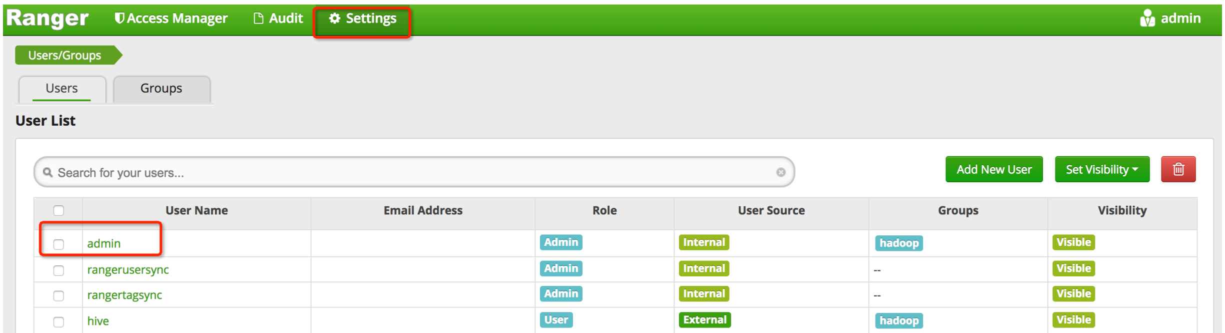
Username: admin

Password: •••••



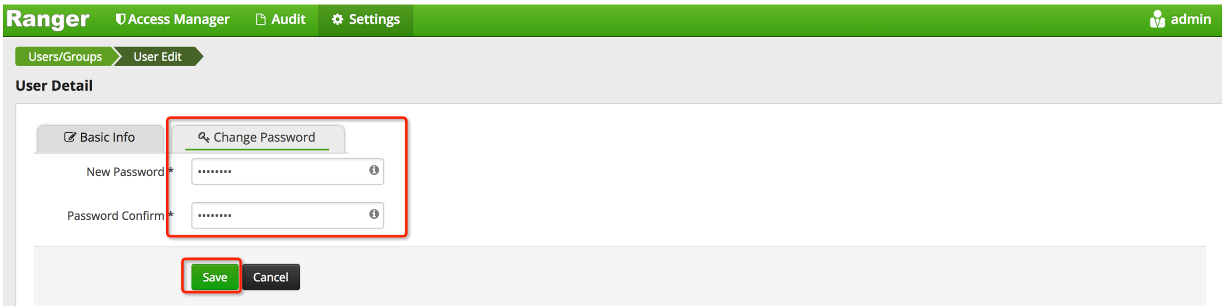
Modify the password

After you first log on, the administrator needs to modify the password of the admin account, as shown in the following figure.



User List

□	User Name	Email Address	Role	User Source	Groups	Visibility
□	admin		Admin	Internal	hadoop	Visible
□	rangerusersync		Admin	Internal	--	Visible
□	rangertagsync		Admin	Internal	--	Visible
□	hive		User	External	hadoop	Visible



User Detail

New Password * ••••••••

Password Confirm * ••••••••

After you change the admin password, in the admin drop-down list in the upper-right corner, click Log Out. You can then log on again with the new password.

Integrate Ranger into other services

After completing the preceding steps, you can integrate Ranger into the services in the cluster to control the relevant permissions. For more information, see the following:

- *Integrate Ranger into HDFS*
- *Integrate Ranger into Hive*
- *Integrate Ranger into HBase*

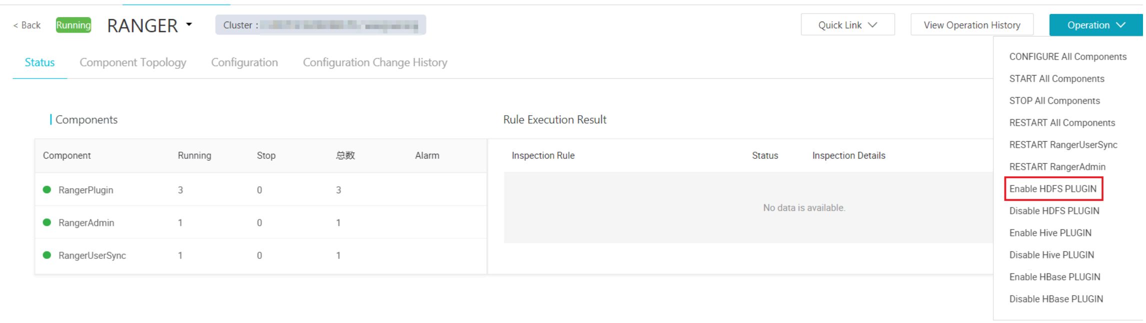
16.6.2 Integrate Ranger into HDFS

Procedure

This section describes the step-by-step process for integrating Ranger into HDFS.

- Enable the HDFS plug-in

  1. On the Cluster Management page, click Manage next to the cluster you want to operate in the Actions column.
  2. Click Ranger in the service list to enter the Ranger Management page.
  3. On the Ranger Configuration page, click the Actions drop-down menu in the upper-right corner, select Enable HDFS PLUGIN, and click OK.



Components

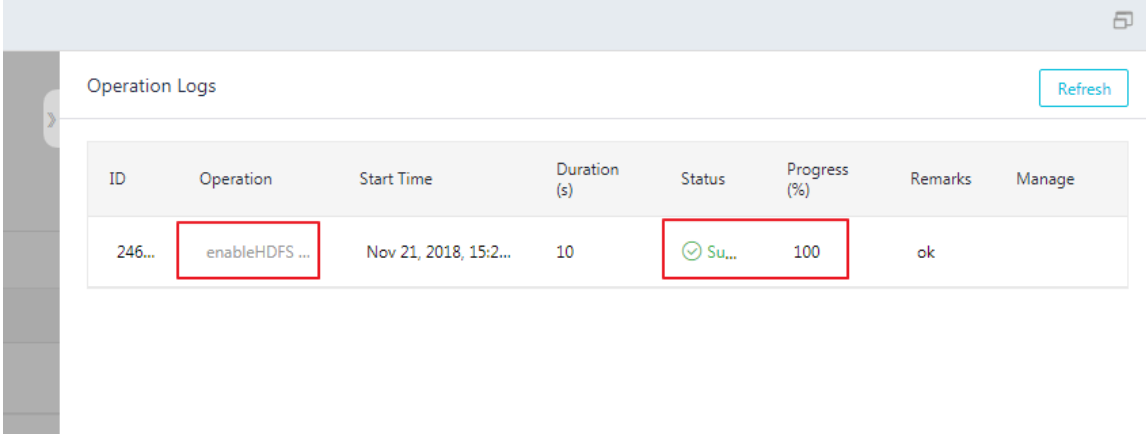
Component	Running	Stop	总数	Alarm
● RangerPlugin	3	0	3	
● RangerAdmin	1	0	1	
● RangerUserSync	1	0	1	

Rule Execution Result

Inspection Rule	Status	Inspection Details
No data is available.		

  4. Enter the record information in the prompt box and click OK.

     You can check the progress by clicking View Operation Logs in the upper-right corner of the page.

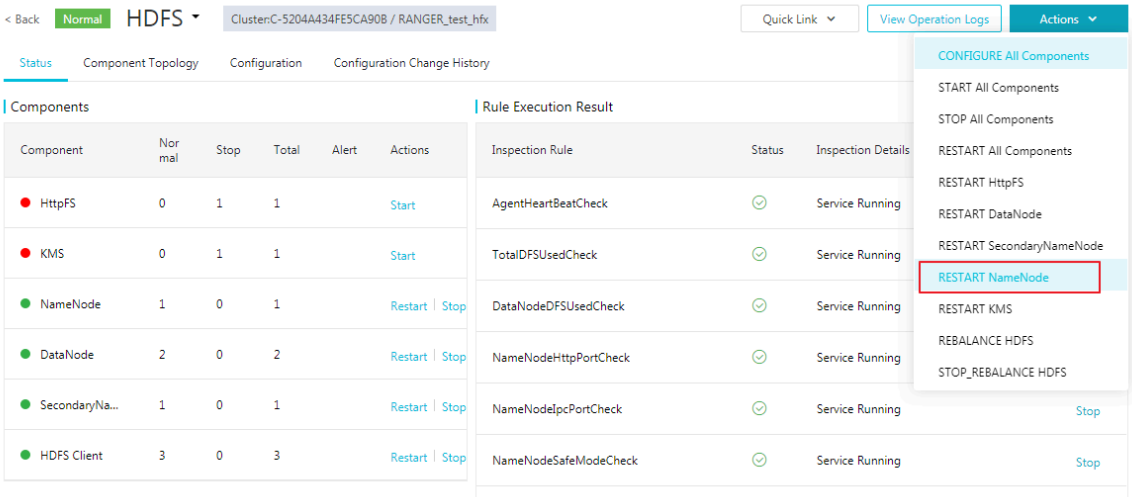


ID	Operation	Start Time	Duration (s)	Status	Progress (%)	Remarks	Manage
246...	enableHDFS ...	Nov 21, 2018, 15:2...	10	✓ Su...	100	ok	

- Restart NameNode

  After enabling the HDFS plug-in, you need to restart NameNode. To do so, complete the following steps:

  1. In the Ranger Management page, click the Ranger drop-down menu in the upper -left corner, and select HDFS.
  2. Click Actions in the upper-right corner of the page and select RESTART NameNode.
  3. You can check the progress by clicking View Operation Logs in the upper-right corner of the page.



Components

Component	Normal	Stop	Total	Alert	Actions
● HttpFS	0	1	1		Start
● KMS	0	1	1		Start
● NameNode	1	0	1		Restart | Stop
● DataNode	2	0	2		Restart | Stop
● SecondaryNa...	1	0	1		Restart | Stop
● HDFS Client	3	0	3		Restart | Stop

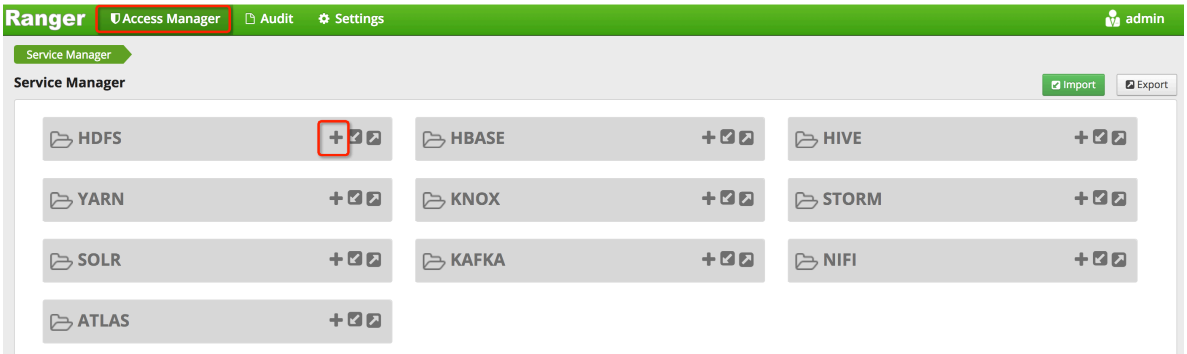
Rule Execution Result

Inspection Rule	Status	Inspection Details	
AgentHeartBeatCheck	✓	Service Running	
TotalDFSUsedCheck	✓	Service Running	
DataNodeDFSUsedCheck	✓	Service Running	
NameNodeHttpPortCheck	✓	Service Running	
NameNodeIpcPortCheck	✓	Service Running	Stop
NameNodeSafeModeCheck	✓	Service Running	Stop

- Add the HDFS service to Ranger UI

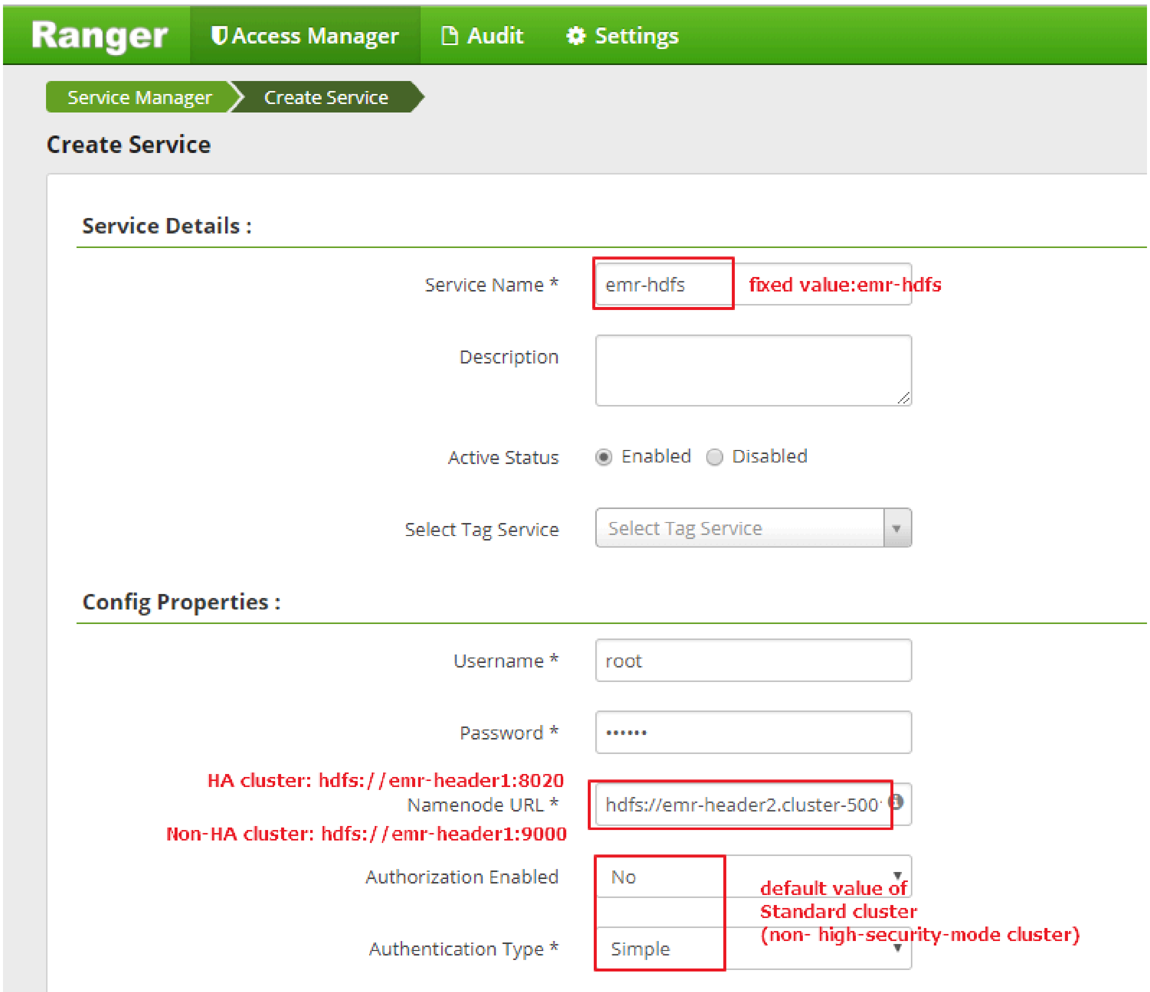
  For more information about how to access the Ranger UI, see *Introduction to Ranger*.

  Add the HDFS service.



- Standard cluster

  To check the mode of the cluster you created, go to the Cluster Overview page. If your cluster is in standard mode, configure it as follows:



Create Service

Service Details :

Service Name * emr-hdfs fixed value:emr-hdfs

Description

Active Status ● Enabled ○ Disabled

Select Tag Service: Select Tag Service

Config Properties :

Username * root

Password * ••••••

HA cluster: hdfs://emr-header1:8020
Namenode URL * hdfs://emr-header2.cluster-500
Non-HA cluster: hdfs://emr-header1:9000

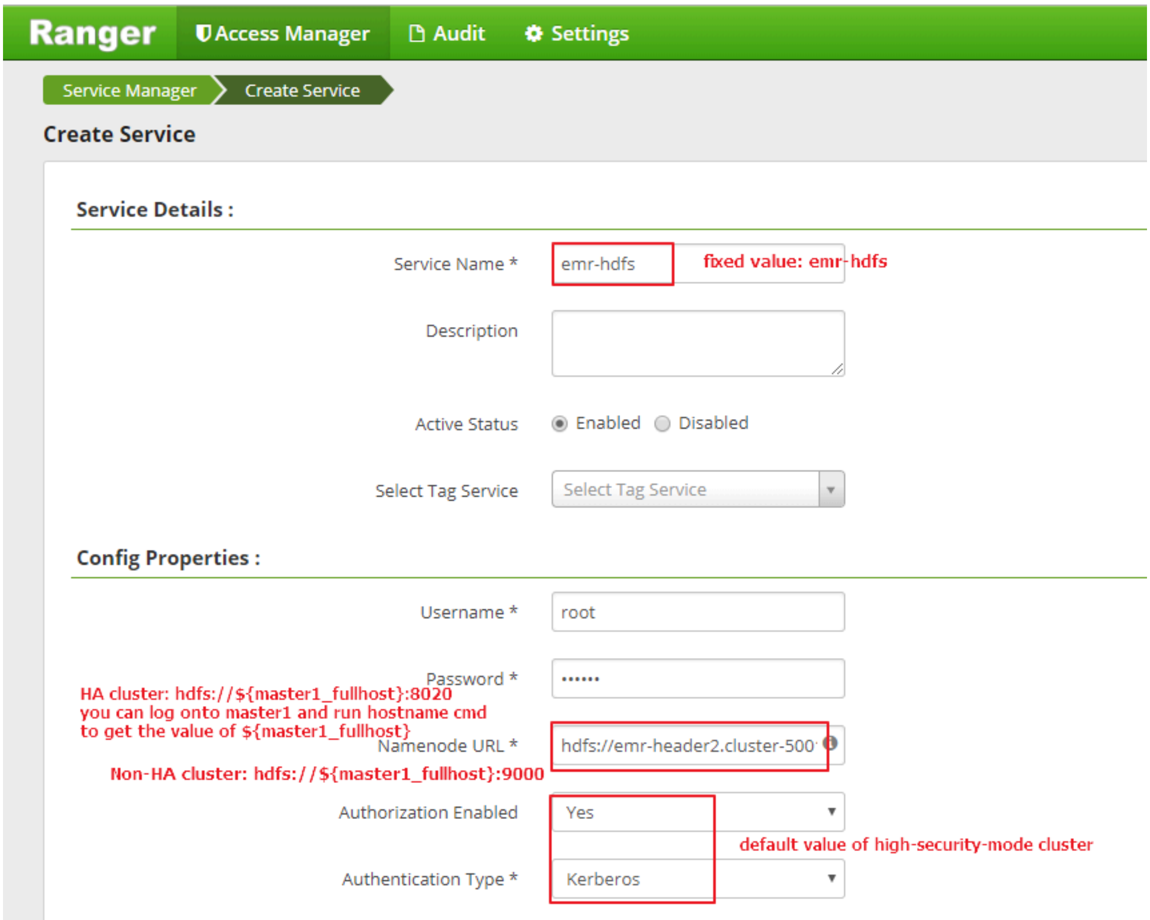
Authorization Enabled: No

Authentication Type * Simple

default value of Standard cluster (non- high-security-mode cluster)

- High-security-mode cluster

  To check the mode of the cluster you created, go to the Cluster Overview page. If your cluster is in high-security mode, configure it as follows:



Create Service

Service Details :

Service Name * emr-hdfs

fixed value: emr-hdfs

Description

Active Status ● Enabled ○ Disabled

Select Tag Service: Select Tag Service

Config Properties :

Username * root

Password * ••••••

HA cluster: hdfs://${master1_fullhost}:8020
you can log onto master1 and run hostname cmd
to get the value of ${master1_fullhost}

Namenode URL * hdfs://emr-header2.cluster-500

Non-HA cluster: hdfs://${master1_fullhost}:9000

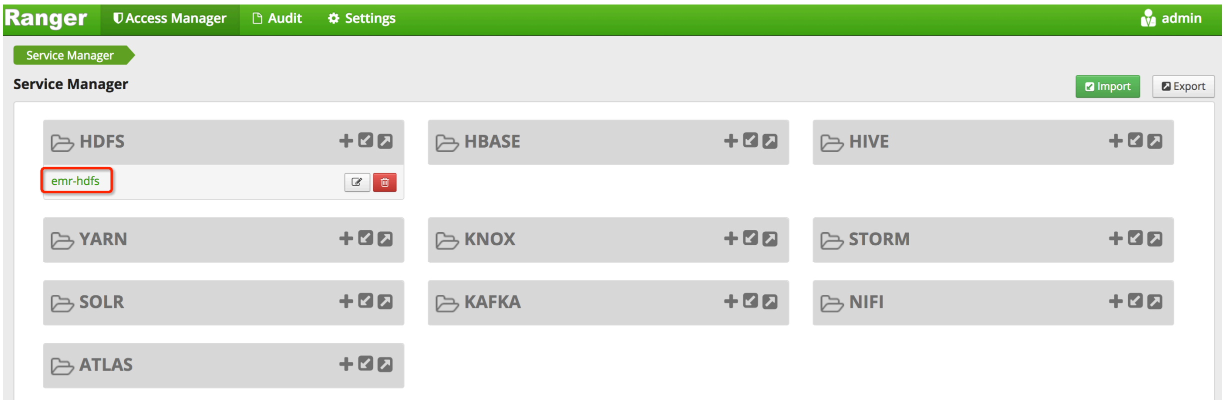
Authorization Enabled: Yes

default value of high-security-mode cluster

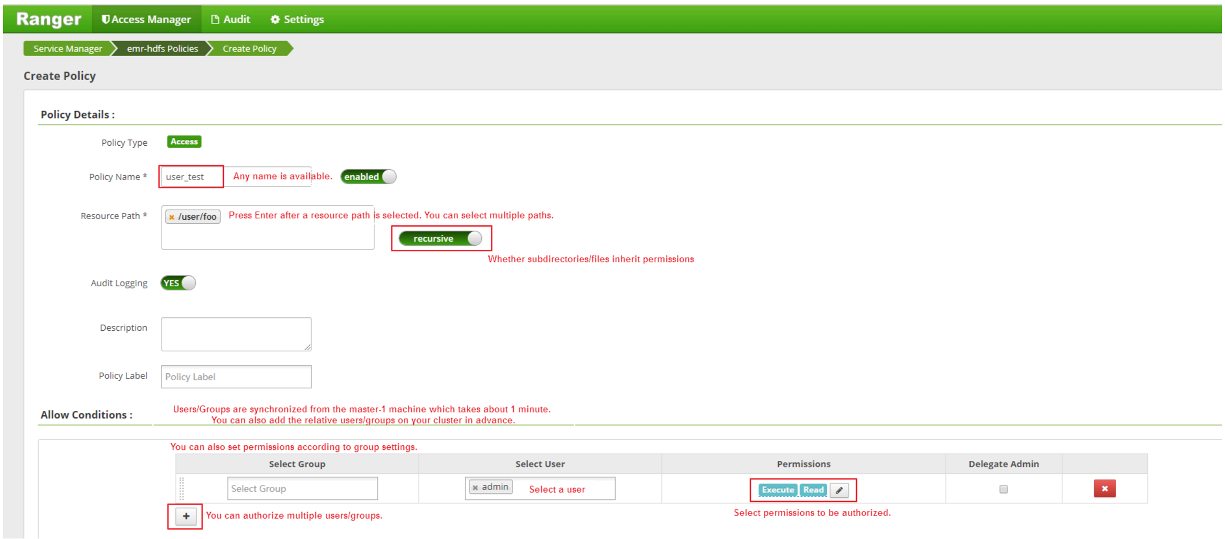
Authentication Type *: Kerberos

Permission configuration

After integrating Ranger into HDFS, you can set permissions, such as granting the test user the write or execute permission for `/user/foo`.



In the preceding figure, click emr-hdfs to enter the policy configuration page.



Create Policy

Policy Details :

Policy Type: Access

Policy Name *: user_test — Any name is available. — enabled

Resource Path *: /user/foo — Press Enter after a resource path is selected. You can select multiple paths.

recursive — Whether subdirectories/files inherit permissions

Audit Logging: YES

Description:

Policy Label: Policy Label

Allow Conditions : Users/Groups are synchronized from the master-1 machine which takes about 1 minute. You can also add the relative users/groups on your cluster in advance.

You can also set permissions according to group settings.

Select Group	Select User	Permissions	Delegate Admin	
Select Group	admin Select a user	Execute Read	□	✖

+ You can authorize multiple users/groups.

Select permissions to be authorized.

Permissions are granted to the test user. They can now access the HDFS path of `/user/foo`.

Note:

The policy takes effect one minute after it is added.

16.6.3 Integrate Ranger into Hive

Procedure

This section describes the step-by-step process for integrating Ranger into Hive.

- Hive access model

  You can access Hive data in three ways: HiveServer2, Hive client, and HDFS.

  - HiveServer2

    - Mode: Use the Beeline client or the JDBC code to run the related Hive script through HiveServer2.
    - Permission settings:

      Hive's *SQL Standard Based Authorization* is used to control the permissions of HiveServer2.

      Hive's table- and column-level permission control in Ranger is also used for HiveServer2. However, if you are still able to access Hive data though a Hive client or HDFS, table- or column-level permission control is insufficient and further control is required.

  - Hive client

- Mode: Access from a Hive client.
- Permission settings:

  The Hive client requests the metastore to perform DDL operations, such as altering tables, adding columns, and reading and processing data in HDFS, by submitting MapReduce jobs.

  Hive's *Storage Based Authorization* is used to control the permissions of Hive clients. It determines whether a user can perform DDL and DML operations based on the read and write permissions of the HDFS path where the table involved in SQL is located, such as `ALTER TABLE test ADD COLUMNS(b STRING)`.

  In Ranger, you can control the permissions of the HDFS path in Hive tables . This, in combination with the Hive metastore which is configured with storage-based authorization, enables you to implement permission control over the access of the Hive client.

  > Note:
  > The DDL operation permissions of a Hive client are actually controlled by the underlying HDFS permissions. If you have HDFS permissions, you also have DDL permissions for tables, such as dropping and altering tables.

- HDFS

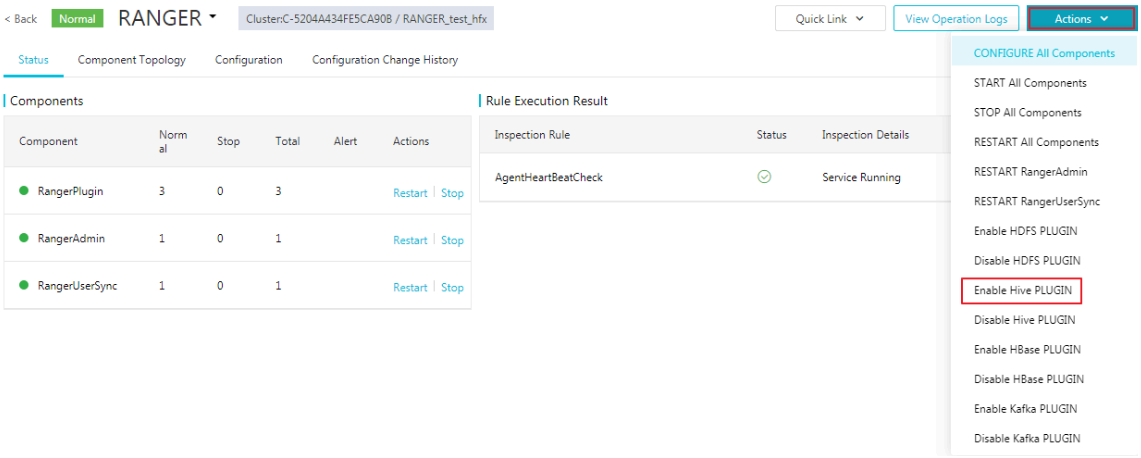
  - Mode: HDFS client and code.
  - Permission settings:

    To have direct access to HDFS, you need to add permission control for HDFS on the underlying HDFS data of the Hive tables.

    You can use Ranger to perform permission control for the underlying HDFS path of Hive tables.

- Enable the Hive plug-in

1. On the Cluster Management page, click Manage next to the cluster you want to operate in the Actions column.
2. Click Ranger in the service list to enter the Ranger Management page.
3. On the Ranger Configuration page, click the Actions drop-down menu in the upper-right corner, select Enable Hive PLUGIN, and click OK.



Components

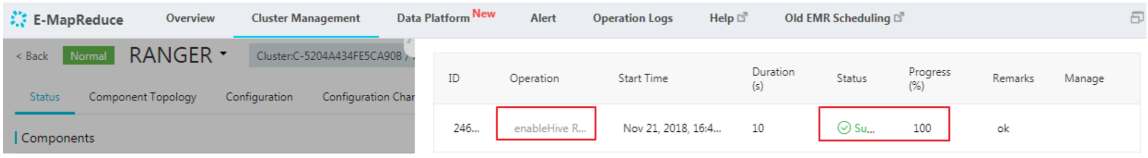
Component	Normal	Stop	Total	Alert	Actions
● RangerPlugin	3	0	3		Restart | Stop
● RangerAdmin	1	0	1		Restart | Stop
● RangerUserSync	1	0	1		Restart | Stop

Rule Execution Result

Inspection Rule	Status	Inspection Details
AgentHeartBeatCheck	✓	Service Running

4. Enter the record information in the prompt box and clickOK.

   You can check the progress by clicking View Operation Logs in the upper-right corner of the page.



ID	Operation	Start Time	Duration (s)	Status	Progress (%)	Remarks	Manage
246...	enableHive R...	Nov 21, 2018, 16:4...	10	✓ Su...	100	ok	

Note:

After you enable the Hive plug-in and restart Hive, HiveServer2 and Hive client scenarios are configured accordingly. For more information about HDFS permissions, see *Integrate Ranger into HDFS*.

- Restart Hive

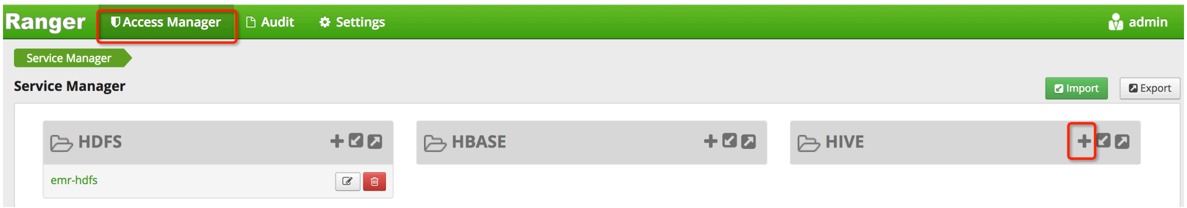
  After enabling the Hive plug-in, you need to restart Hive. To do so, complete the following steps:

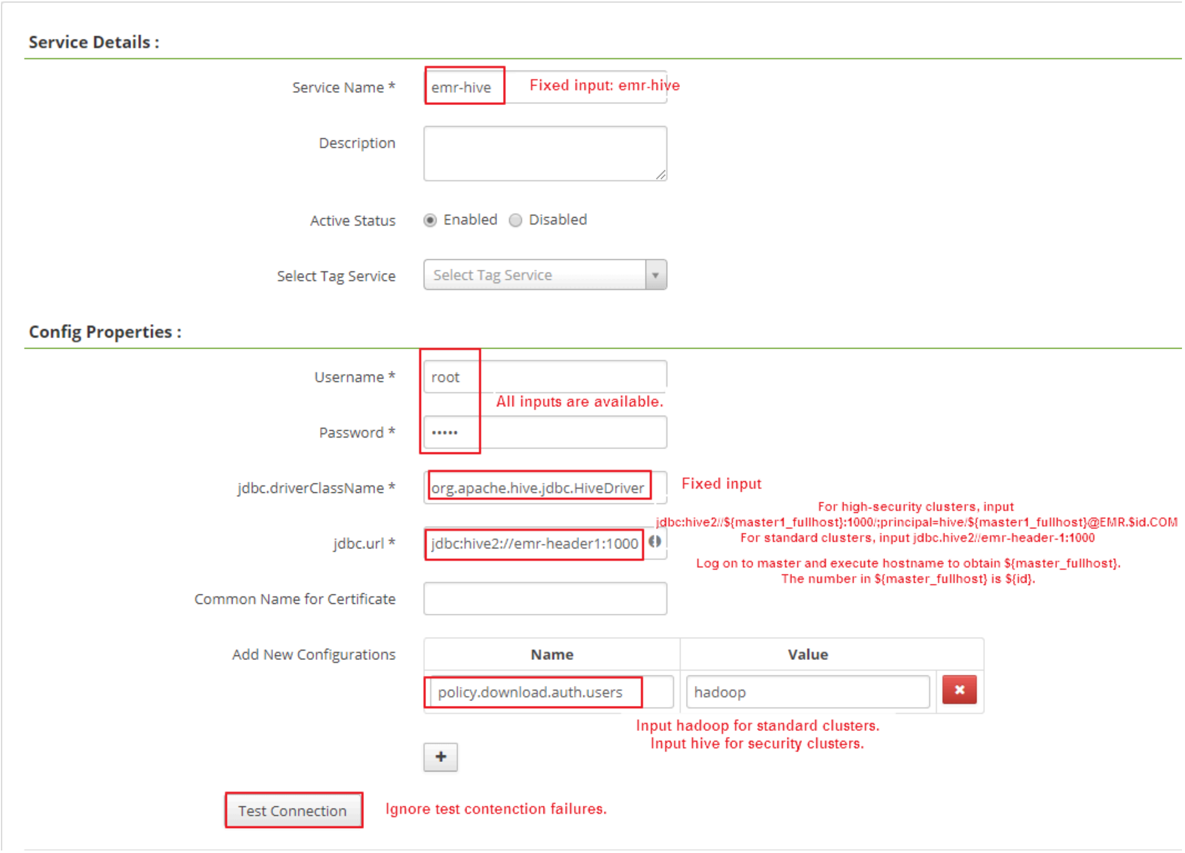
  1. In the Ranger Management page, click the Ranger drop-down menu in the upper -left corner, and select Hive.
  2. Click Actions in the upper-right corner, select RESTART All Components from the drop-down menu, and click OK.
  3. You can check the progress by clicking View Operation Logs in the upper-right corner of the page.

- Add the Hive service to the Ranger UI

  For more information about how to access the Ranger UI page, see *Introduction to Ranger*.

Add the Hive service.





Service Details :

Service Name * emr-hive
Fixed input: emr-hive

Description

Active Status ● Enabled ○ Disabled

Select Tag Service: Select Tag Service

Config Properties :

Username * root

Password * •••••

All inputs are available.

jdbc.driverClassName * org.apache.hive.jdbc.HiveDriver
Fixed input

jdbc.url * jdbc:hive2://emr-header1:1000

For high-security clusters, input
jdbc:hive2//${master1_fullhost}:1000/;principal=hive/${master1_fullhost}@EMR.$id.COM
For standard clusters, input jdbc.hive2//emr-header-1:1000

Log on to master and execute hostname to obtain ${master_fullhost}.
The number in ${master_fullhost} is ${id}.

Common Name for Certificate

Add New Configurations

Name	Value
policy.download.auth.users	hadoop

Input hadoop for standard clusters.
Input hive for security clusters.

Test Connection
Ignore test contention failures.

- Instructions

  Enter a fixed value for the following configuration items:

Name	Value
Service Name	emr-hive
jdbc.driverClassName	org.apache.hive.jdbc.HiveDriver

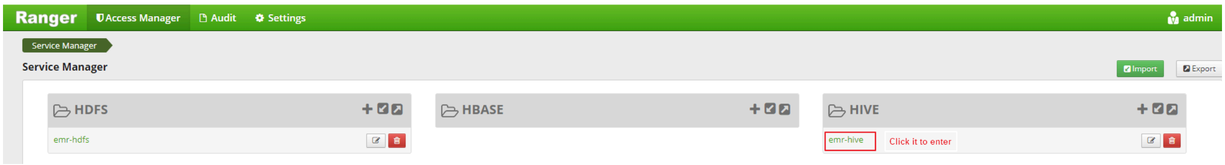
- Enter a variable value for the following configuration items:

Name	Value
jdbc.url	Standard cluster: jdbc:hive2://emr-header-1:10000/ high-High-security cluster: jdbc:hive2://${master1_fullhost}:10000/;principal=hive/${master1_fullhost}@EMR.$id.COM
policy.download.auth.users	Standard cluster: hadoop High-security cluster: hive

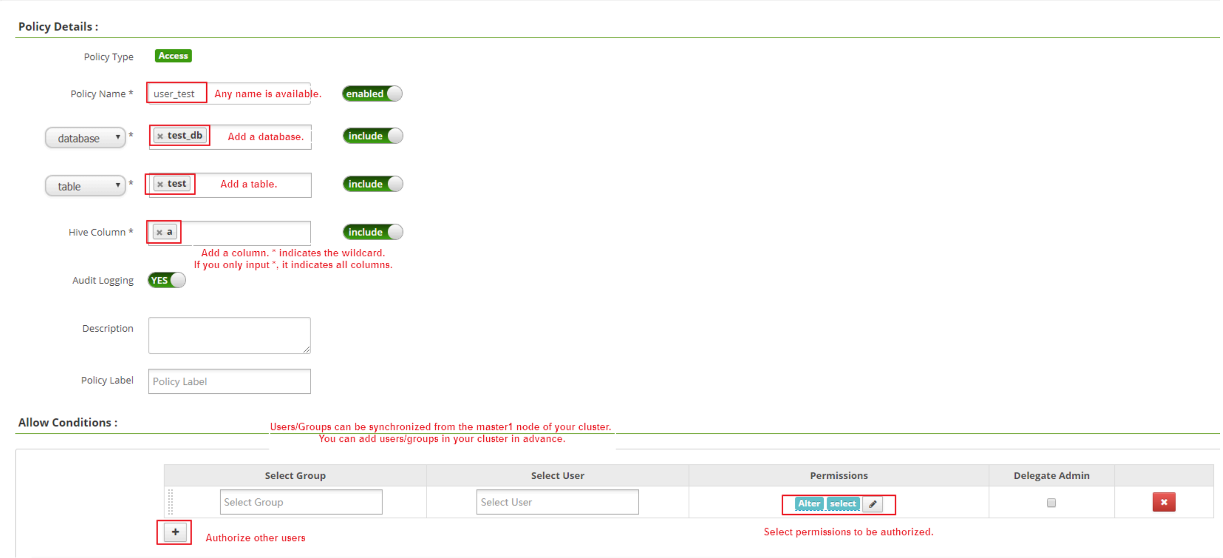
${master1_fullhost} is the long domain name of master1. To obtain this name, log on to master1 and run the `hostname` command. The number in ${master1_fullhost} is the value of $id.

Permission configuration

After integrating Ranger into Hive, you can set permissions, such as granting user foo the Select permission for column A in the testdb.test table.



In the preceding figure, click emr-hive to enter the policy configuration page.


Policy Details :

Policy Type: Access

Policy Name *: user_test — Any name is available. — enabled

database *: test_db — Add a database. — include

table *: test — Add a table. — include

Hive Column *: a — include

Add a column. * indicates the wildcard.
If you only input *, it indicates all columns.

Audit Logging: YES

Description

Policy Label: Policy Label

Allow Conditions :

Users/Groups can be synchronized from the master1 node of your cluster.
You can add users/groups in your cluster in advance.

Select Group	Select User	Permissions	Delegate Admin
Select Group	Select User	Alter select	□

Authorize other users

Select permissions to be authorized.

Permissions are granted to user foo. They can now access the testdb.test table.

> **Note:**
>
> The policy takes effect one minute after it is added.

17 Disaster recovery

17.1 Disaster recovery in E-MapReduce clusters

This article will introduce disaster recovery of data and dervices in E-MapReduce clusters

Data

HDFS stores the data of each file in blocks, with each block holding multiple copies (three by default). HDFS also makes sure that these copies are stored in different frameworks. In most situations, HDFS stores the first copy in the local framework, the second in the same framework as the first but in different nodes, and the last copy in a different framework.

HDFS scans the data copies regularly. If it finds that a data copy has been lost, HDFS makes another to make sure the number of copies is stable. If a node that stores a copy has been lost, HDFS makes another node to recover the data in that node. In Alibaba Cloud, if you use cloud disks, each cloud disk has three data copies in the back-end. If any of them has an issue, the copies exchange and recover data to ensure reliability.

HDFS is a highly reliable file storage system that can store massive amounts of data . Based on the features of Alibaba Cloud, HDFS can also make backups of the data stored in OSS, providing even greater data reliability.

Services

The core components of HDFS guarantee high availability by making sure that there are at least two nodes to back each other up, such as YARN, HDFS, HiveServer, or Hive Meta. In this way, whenever a node experiences an issue, the nodes can exchange and recover data to ensure that services are not impacted.

18 Auto Scaling

18.1 Introduction to Auto Scaling

This section introduces how to enable and disable Alibaba Cloud Auto Scaling in E-MapReduce.

By implementing E-MapReduce Auto Scaling, you can cut costs and improve efficiency in the following scenarios:

- You need to temporarily add computing nodes (according to the time) to supplement computing power.
- You need to make sure that important jobs are completed on time and computing nodes are expanded according to certain cluster indicators.

Note:

- Auto Scaling can only expand or reduce the number of task nodes.
- Auto Scaling is available in both Subscription and Pay-As-You-Go clusters.

Enable Auto Scaling

1. Log on to the *Alibaba Cloud E-MapReduce console* and enter the Cluster Management page.
2. Click Manage next to the target cluster ID.
3. In the navigation pane on the left, click Scaling to enter the Auto Scaling page.
4. In the top right corner of the page, click Enable Auto Scaling.

   If it is your first time using Auto Scaling with your account, you first need to authorize its default role in your E-MapReduce account.
5. Click Confirm on the Auto Scaling authorization page.

Disable Auto Scaling

After you click Disable Auto Scaling, all expanded task nodes will be released. The data stored in HDFS is located in a core node and is not affected.

18.2 Configure Auto Scaling by time

If the computing capability of a Hadoop cluster sees significant peaks and troughs over a specified period of time, you can set a fixed time frame within which a certain number of task nodes supplement the computing capability. This not only ensures that jobs are completed, but it also saves you money.

The expansion nodes are billed in Pay-As-You-Go mode. However, for the same computing capability, the price ratio between Pay-As-You-Go and Subscription modes is around 3:1. Therefore, it is necessary to design a ratio for both modes based on the time needed. For example, if there are 8 peak business hours a day, the price for Pay-As-You-Go is essentially the same as that for Subscription. If peak hours last longer than 8 hours, the Subscription mode is more cost-effective than Pay-As-You-Go.

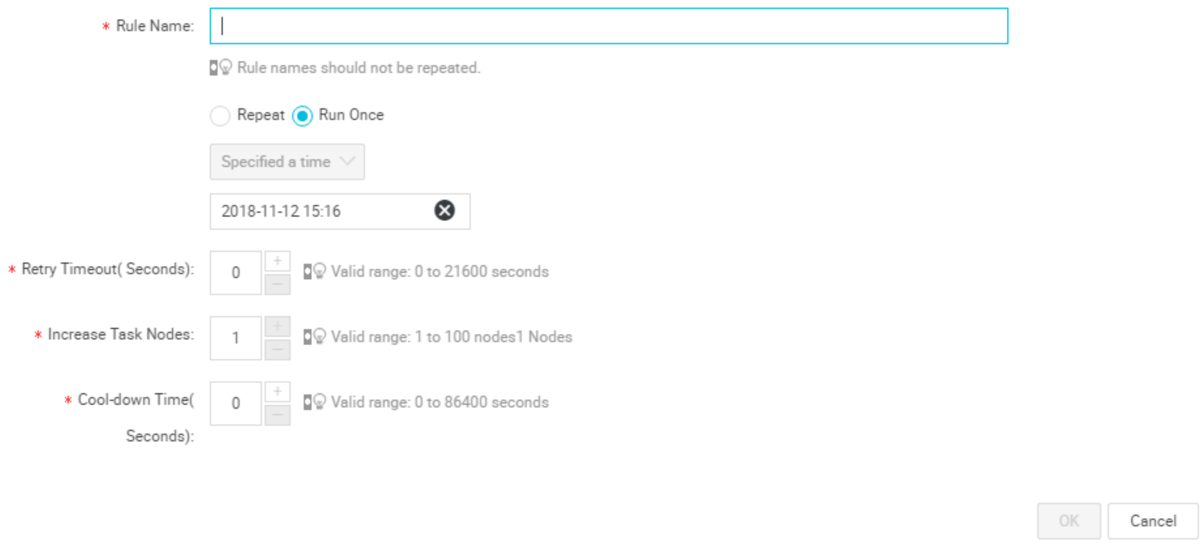
Configure the number of scaling instances

- Maximum number of nodes: The maximum number of nodes that can be expanded. Once this number is reached, even if the Auto Scaling rule is met, expansion and contraction will stop. Currently, you can set up to 1,000 task nodes.
- Minimum number of nodes: The minimum number of nodes that can be expanded. If the number of task nodes set in the Auto Scaling rule is less than this minimum number of nodes, the cluster scales based on the minimum number of nodes at the first execution.

  For example, if the Auto Scaling rule is set to expand 1 node at 00:00:00 every day and the minimum number of nodes is 3, then the system expands 3 nodes at the 00:00:00 on the first day.

Configure scaling rules

Auto Scaling rules include expansion rules and contraction rules. When Auto Scaling is disabled, all rules are cleared. If it is enabled again, the scaling rules need to be reconfigured.



* Rule Name:

Rule names should not be repeated.

○ Repeat ● Run Once

Specified a time

2018-11-12 15:16

* Retry Timeout(Seconds): 0 Valid range: 0 to 21600 seconds

* Increase Task Nodes: 1 Valid range: 1 to 100 nodes1 Nodes

* Cool-down Time(Seconds): 0 Valid range: 0 to 86400 seconds

- Rule Name: In a cluster, the scaling rule names (including expansion rules and contraction rules) cannot be repeated.
- Execution cycle:
  - Run Once: The cluster performs a scaling operation at a specified time.
  - Repeat: You can choose to perform a scaling operation at a specific time every day, every week, or every month.
- Retry Timeout: When the specified time is reached, scaling cannot be performed. By setting the retry timeout, the system can detect that scaling can be performed every 30 seconds in the time range. Once the condition is met, scaling is performed. The range is 0 to 21600 seconds.

  It is assumed that if expansion A needs to be performed in the specified time period, but expansion B is being performed or the cluster is in cool-down time, expansion A cannot be performed. During the retry timeout you set, the system detects that expansion A can be performed every 30 seconds. Once the conditions are met, the cluster immediately performs scaling.
- Increase Task Nodes: The number of task nodes to be increased or decreased by the cluster each time the rule is triggered.
- Cool-down Time: The interval between a scaling operation being completed and the same operation being performed again. Scaling operations are not performed during cool-down time.

Configure scaling instance specifications

When Auto Scaling is enabled, you can specify the hardware specifications for the scaling nodes. The specifications cannot be modified after the configuration is saved. If you need to modify them, you can disable Auto Scaling and then enable it again.

- When you select specifications for vCPU and memory, the system automatically matches the instances that meet the criteria and displays them in the instance list below. For the cluster to be able scale according to the selected instance specificat ions, you need to add an optional instance to the list on the right.
- To avoid scaling failures caused by insufficient ECS instance storage, you can choose up to 3 ECS instance types.
- Regardless of whether you choose an efficient cloud disk or a SSD cloud disk, the data disk is set to a minimum of 40G.

18.3 Preemptible instances in Auto Scaling

E-MapReduce *preemptible instances* are suitable for scenarios where there is no requirement for the successful execution of big data jobs and where the cost of computing resources plays an important role. By using Auto Scaling, you can purchase preemptible instances to increase the computing resources of your clusters.

Enable Auto Scaling

To enable Auto Scaling and set scaling rules, complete the following steps:

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. Click Cluster Management.
3. Find the cluster you want to add a preemptible instance to and click Manage.
4. In the navigation pane on the left, click Scaling.
5. Click Enable Auto Scaling.
6. Configure scaling rules. For more information, see *Configure Auto Scaling by time*.
7. In the scaling configuration area, select Preemptible instance.

Configure a preemptible instance

Note:

Preemptible instances are more cost-effective than Pay-As-You-Go instances. However, Alibaba Cloud may release your preemptive instances at any time based on changes in demand resources or market rates.

To configure your preemptible instance, complete the following steps:

1. Select the vCPU and memory for your instance.
2. Select instance types. You can select up to three instance types. E-MapReduce filters out all other instance types to ensure that you purchase a preemptible instance that meets your requirements.
3. After you select the instance types, click the maximum price of each type and click OK. The instance types appear in the selected instances list. If you want to modify the price of a selected instance type, select the target one in the selected instances list and change the price (by hour). Your instance will run when your bid is higher than the current market rate. Your final instance type is billed at the market rate.
4. The system disk is used for deploying basic services such as the OS and EMR, which are set by default. You can set the data disk type and size according to your requirements.
5. The final configuration price includes the maximum bid price, system disk price, and data disk price. Click Save.

For more information about preemptible instances, see *FAQs*.

18.4 Auto Scaling records

After completing an Auto Scaling operation, click the Scaling Records tab at the top of the Scaling page to see the records of the operation and the number of nodes.

Auto Scaling statuses comprise the following four types:

- Running: The operation is being implemented.
- Success: All of the specified nodes involved in the scaling rule have been added to or removed from the cluster.
- Partial success: Some nodes were successfully added to or removed from the cluster, but others failed due to the disk or ECS instance storage.
- Failure: No node was added to or removed from the cluster.

19 Workflow development

19.1 Manage a workflow project

After creating an E-MapReduce cluster, you can create workflow projects so that multiple jobs can be run simultaneously or sequentially.

Create a project

1. At the top of the page, click the Data Platform tab to enter the Projects page.

   Under the master account, you can view all of its projects and RAM user accounts. RAM users can only view projects if they have development permissions. The granting of project development permissions must be configured in the master account. For more information about authorization, see *User management* below.
2. In the upper-right corner, click New Project. The New Project dialog box is displayed.
3. Enter the project name and description and click Create.

   Note:

   You can only create a project with the master account. New Project is only visible to the administrator of the master account.

User management

After creating a new project, you can grant operational permissions for the project to RAM user accounts.

1. In the Project List page, click View Details in the Actions column.
2. Click the User Management tab.
3. Click Add User to add RAM users to the project under the master account.

   The added RAM users become members of the project and are able to view and develop the jobs and workflows under the project. If you remove a RAM user from a project, click Delete in the Actions column.

   Note:

   You can only add project members with the master account. The User Management tab is only visible to the administrator of the master account.

Associate clusters

After creating a new project, you need to associate it with a cluster so that the workflow in the project can run on it.

1. In the Projects page, click View Details in the Actions column.
2. Click the Cluster Settings tab.
3. Click Add Cluster. From the drop-down menu, you can select a Subscription or Pay-As-You-Go cluster. (Clusters created by temporary jobs are not listed here.)
4. Click OK.

   To disassociate the cluster, click Delete in the Operation column.

   > Note:
   >
   > You can only associate cluster with the master account. The Cluster Settings tab is only visible to the administrator of the master account.

To set both the queue and user to submit jobs to the cluster, click Modify Configuration in the Operation column. The configuration items are as follows:

- Default Submit Job User: Sets the default Hadoop user who submits the job to the selected cluster in the project. The default value is hadoop. There can only be one default user.
- Default Submit Job Queue: Sets the default queue that the jobs are submitted to in the project. If you leave this blank, the job will be submitted to the default queue.
- Submit Job User Whitelist: Sets Hadoop users who can submit jobs to the selected cluster in the project. If there is more than one user, they can be separated by a comma (,).
- Submit Job Queue Whitelist: Sets the queue of the selected cluster that jobs in the project can run in. If there is more than one queue, they can be separated by a comma (,).
- Client whitelist: Configures the client that can submit jobs. You can select either the E-MapReduce master node or the E-MapReduce gateway. Gateways that you have built are not currently listed here.

19.2 Job operations

In a project, you can create jobs such as Shell, Hive, Spark, SparkSQL, MapReduce, Sqoop, Pig and Spark Streaming jobs.

Create a job

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. Click the Data Platform tab to enter the Projects page.
3. Click Design Workflow next to the target project in the Actions column.
4. On the left side of the job editing page, right-click the folder you want to operate on and select New Job.
5. In the New Job dialog box, enter a name and description for the job and select a job type.

   Once the job type is selected, it cannot be modified.
6. Click OK.

> Note:
>
> You can also create subfolders, rename folders, and delete folders by right-clicking on them.

Develop a job

For more information about the various types of jobs, see *Jobs*.

> Note:
>
> When you insert an OSS path, if you select the OSSREF file prefix, the OSS file is downloaded to the cluster and added to the classpath.

- Basic settings

  Click the Job Settings in the upper-right corner of the page to enter the Job Running Configuration page.

  - Number of Retries: Sets the number of retries for if a job fails during a workflow. This option does not take effect if you run the job directly on the Job Editing page.
  - Failure Policy: Sets whether to run the next job or suspend the current workflow in the event that a job fails during a workflow.

- Resource File: If you add resources such as JAR packages or UDFs that the current job depends on, you need to upload the resources to OSS first. After doing so, you can reference the resources directly in the job code.
- Parameter Configuration: Specifies the value of the variable referenced in the job code. You can reference variables in the code with the format `${variable name}`. Click the plus icon (+) on the right to add the key and value. The key is the variable name, whereas the value is the value of the variable. You can also customize the time variable according to the schedule time. The rules are as follows:

  - yyyy indicates the year (4-digit format).
  - MM indicates the month.
  - dd indicates the day.
  - HH24 indicates the hour. If the 12-hour clock is used, this is displayed as hh.
  - mm indicates the minute.
  - ss indicates the second.

    The time variable can be any combination of time containing yyyy. You can also use the plus symbol (+) to advance time and the minus symbol (-) to delay time. For example, if `${yyyy-MM-dd}` indicates the current date, then:

    - One year from now is: `${yyyy+1y}` or `${yyyy-MM-dd hh:mm:ss+1y}`.
    - Three months from now is: `${yyyyMM+3m}` or `${hh:mm:ss yyyy-MM-dd +3m}`.
    - Five days ago is: `${yyyyMMdd-5d}` or `${hh:mm:ss yyyy-MM-dd-5d}`.

- Advanced settings

  To configure advanced settings, click the Advanced tab on the Job Settings page.

  - Mode: Job running modes, including YARN and LOCAL. In YARN mode, the job is submitted on YARN by the Launcher. In LOCAL mode, jobs run directly on the assigned host.
  - Environment Variables: Add environment variables to run jobs, or export environment variables directly in the job script.
  - Scheduling Parameters: Set job configurations, such as the YARN queue, CPU, memory, and Hadoop users. If you do not set this parameter, the job adopts the default value of the Hadoop cluster.

Run a job

Once a job has been developed and configured, you can click Run in the top right corner to run the job.

View a log

After the job has been set to run, you can view its running log in the View Records tab at the bottom. Click Workflow to enter the detailed log page. Here, you can see information such as the job's submitting log and the YARN Container log.

19.3 Ad hoc queries

Only three types of ad hoc query are supported: HiveSQL, SparkSQL, and Shell. When you execute an ad hoc query statement, the log and query results are displayed at the bottom of the log and query page.

Create a job

When you execute a job on the Edit Jobs page and click Details, you are directed to the Details page that shows the operation logs and run logs of this job. Ad hoc queries and jobs are used in different places. Ad hoc queries are usually used by data analysts . You also need to use SQL as a tool to implement an ad hoc query.

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. Click the Data Platform tab to enter the Projects page.
3. Click Design Workflow next to the target project to enter the Edit Jobs page.
4. In the navigation pane on the left, click the Query tab to enter the Query page.
5. In the navigation pane on the left, right-click a folder as required and select New Job.
6. In the New Job dialog box, enter the job name and description, and select a job type.

   The job type cannot be modified once the job has been created.
7. Click OK.

> Note:
> By right-clicking a folder, you can rename it, delete it, and create new subfolders.

Develop a job

For more information, see *HiveSQL*, *SparkSQL*, and *Shell* job types.

Note:

When you insert an OSS UNI and select OSSREF as the prefix, E-MapReduce downloads OSS files to your cluster and adds them to the classpath.

- Basic job settings

  In the top-right corner, click Configure Jobs. The Job Settings dialog box is displayed.

  - Resource File: If you want to add resources such as JAR packages or UDFs that a job execution depends on, you must upload these files to OSS. When you select a resource, you can use this resource in a job.
  - Parameter Configuration: Specifies the values of variables used in a job. You can use variables in your code. The format is `${variable name}`. Click the plus icon (+) on the right to add key-value pairs. The key is the name of the variable, whereas value is the value of the variable. You can also customize the time variable according to the schedule time. The rules are as follows:

    - yyyy indicates the year (4-digit format).
    - MM indicates the month.
    - dd indicates the day.
    - HH 24 indicates the hour. If the 12-hour clock is used, this is displayed as hh.
    - mm indicates the minute.
    - ss indicates the second.
    - The time variable can be any combination of time containing yyyy. You can also use the plus symbol (+) to advance time and the minus symbol (-) to delay time. For example, if `${yyyy-MM-dd}` indicates the current date, then:

      - One year from now is: `${yyyy+1y}` or `${yyyy-MM-dd hh:mm:ss+1y}`.
      - Three months from now is: `${yyyyMM+3m}` or `${hh:mm:ss yyyy-MM-dd +3m}`.
      - Five days ago is: `${yyyyMMdd-5d}` or `${hh:mm:ss yyyy-MM-dd-5d}`.

- Advanced job settings

  To configure advanced settings, click the Advanced tab on the Job Settings page.

- Mode: Job running modes, including YARN and LOCAL. In YARN mode, the job is submitted on YARN by the Launcher. In LOCAL mode, jobs run directly on the assigned host.
- Scheduling Parameters: Set job configurations, such as the YARN queue, CPU, memory, and Hadoop users. If you do not set this parameter, the job adopts the default value of the Hadoop cluster.
-

Execute a job

Once a job has been developed and configured, you can click Run in the top right corner to run the job.

View logs

After you execute a job, you can view its running logs on the Log tab at the bottom of the query page.

19.4 Manage a workflow

E-MapReduce workflows support the parallel execution of big data jobs based on DAG. You can also suspend, stop, rerun workflows, and view their running statuses in the web UI.

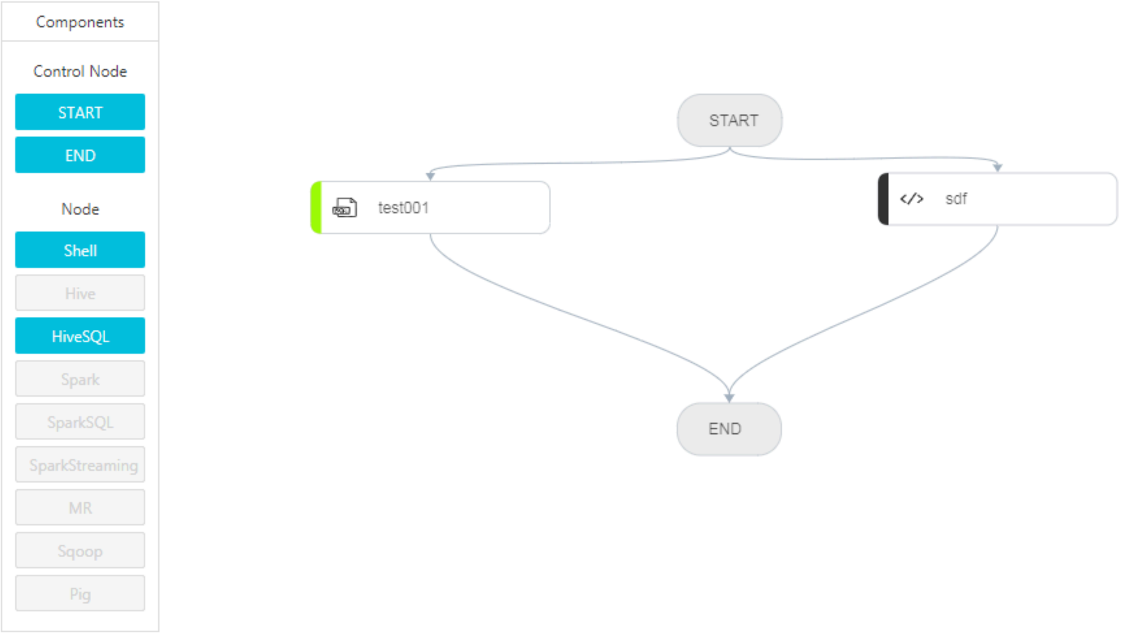
Create a workflow

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the page, click the Data Platform tab.
3. Click Design Workflow next to the target project in the Actions column. Then select the Design Workflow tab.
4. On the left side, right-click the folder you want to operate on and select New Workflow.
5. In the New Workflow dialog box, enter the workflow name and description, and select the E-MapReduce cluster where you want to run the workflow.

   You can select a Subscription or Pay-As-You-Go E-MapReduce cluster that has been created and associated with the project. Alternatively, you can create a new temporary cluster using the cluster template.
6. Click OK.

Edit a workflow

You can drag different types of jobs to the workflow editing canvas and specify the order of job instances by curve. After the job has been dragged, drag the END component from the control node area to the canvas. This indicates that the entire workflow is complete.



Configure a workflow

On the right of the Workflow Design page, click Configure to configure the workflow scheduling.

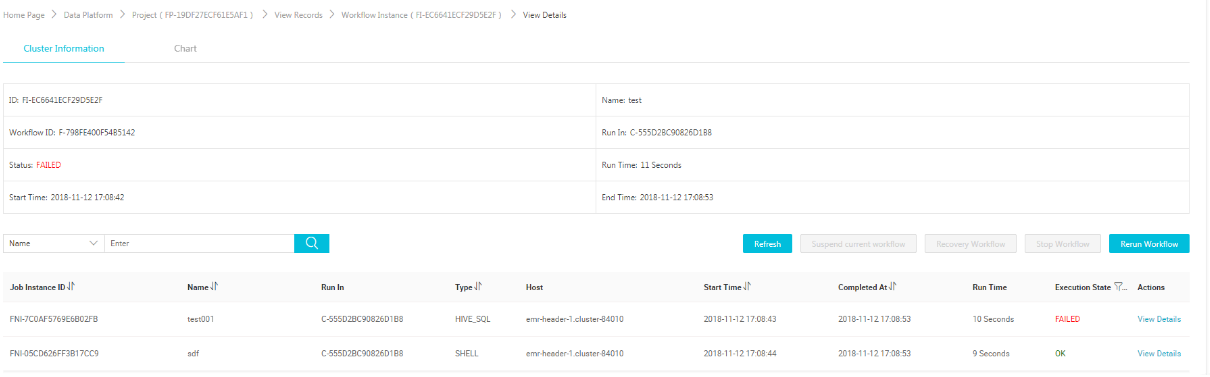
- Run In: The E-MapReduce cluster where the workflow is to run can be modified.
- Scheduling Policy: After workflow scheduling has been enabled, you can choose from two scheduling policies: time scheduler and dependency.
  - Time Scheduler: Sets the start and end times for the workflow scheduling. The system then runs the workflow according to the schedule you set.
  - Dependency: Select the dependency workflow of the current workflow from the selected project. After the dependency workflow has been completed, the current workflow is scheduled to run. Currently, only one workflow can be selected.

Run a workflow

Once a workflow has been developed and configured, you can click Run in the top right corner to run the workflow.

View and operate workflow instances

After the workflow is running, click the View Records tab on the left to view the running status of the workflow instance. Click View Details next to the workflow instance to view the running status of the job instance. You can also suspend, resume, stop, and rerun workflow instances.


Home Page > Data Platform > Project (FP-19DF27ECF61E5AF1) > View Records > Workflow Instance (FI-EC6641ECF29D5E2F) > View Details
Cluster Information
Chart
Name
Enter
Refresh
Suspend current workflow
Recovery Workflow
Stop Workflow
Rerun Workflow

ID: FI-EC6641ECF29D5E2F	Name: test
Workflow ID: F-798FE400F54B5142	Run In: C-555D2BC90826D1B8
Status: FAILED	Run Time: 11 Seconds
Start Time: 2018-11-12 17:08:42	End Time: 2018-11-12 17:08:53

Job Instance ID	Name	Run In	Type	Host	Start Time	Completed At	Run Time	Execution State	Actions
FNI-7C0AF5769E6B02FB	test001	C-555D2BC90826D1B8	HIVE_SQL	emr-header-1.cluster-84010	2018-11-12 17:08:43	2018-11-12 17:08:53	10 Seconds	FAILED	View Details
FNI-05CD626FF3B17CC9	sdf	C-555D2BC90826D1B8	SHELL	emr-header-1.cluster-84010	2018-11-12 17:08:44	2018-11-12 17:08:53	9 Seconds	OK	View Details

- Suspend workflow instance: The job instance continues to run, but subsequent instances do not. By clicking Resume Workflow, the system continues to run the subsequent jobs.
- Stop workflow instance: All running job instances stop immediately.
- Rerun workflow instance: The system runs the workflow from the start component
.

19.5 Jobs

19.5.1 Configure a Hadoop MapReduce job

Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Data Platform.
3. In the Actions column, click Design Workflow next to the specified project.
4. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.
5. In the New Job dialog box, enter the job name and description.

6. Select a Hadoop job type to create a Hadoop MapReduce job. This type of job is submitted in the background using the following process.

   ```
   hadoop jar xxx.jar [MainClass] -Dxxx ....
   ```
7. Click OK.

   > Note:
   >
   > You can also create subfolders, rename folders, and delete folders by right-clicking on them.
8. Enter the parameters in the Content field that are required to submit this job. Enter the parameters after the Hadoop jar, followed by other command line parameters.

   For instance, if you want to submit a Hadoop sleep job that does not read or write any data, this will only succeed if you submit Mapper/Reducer tasks to the cluster and wait for each task to sleep for a while. In Hadoop, this job is packaged in the Hadoop release version's hadoop-mapreduce-client-jobclient-2.6.0-tests.jar. If this job is submitted from the command line, the command should read as follows.

   ```
   hadoop jar /path/to/hadoop-mapreduce-client-jobclient-2.6.0-tests.
   jar sleep -m 3 -r 3 -mt 100 -rt 100
   ```

   To configure this job in E-MapReduce, enter the following content in the Content field.

   ```
   /path/to/hadoop-mapreduce-client-jobclient-2.6.0-tests.jar sleep -m
   3 -r 3 -mt 100 -rt 100
   ```

   > Note:
   >
   > The jar package path used here is an absolute path on the E-MapReduce host. However, the user may put these jar packages anywhere, and as clusters are created and released, the packages become unavailable. Therefore, upload the jar package by performing the following steps:
   >
   > a. Users send their own jar packages to the bucket of OSS for storage. When you configure the parameters for Hadoop, click Select OSS path to select and execute the jar package you want from the OSS directory. Thes system will then auto-complete the OSS address for jar packages. Be sure to switch the prefix of the jar to ossref by clicking Switch resource type. This ensures that the jar package is downloaded correctly by MapReduce.

b. Click OK. The OSS path for this package will be auto-completed in the Content field. When a job is submitted, the system will find the corresponding jar packages automatically based on this path.

c. Behind the jar package path for this OSS, other command line parameters for running jobs will be filled in further.

9. Click Save.

In the example above, the sleep job has no data input/output. If you want the job to read data and process input results, such as word counts, the data input and output paths need to be specified. You can read/write data on the HDFS of the E-MapReduce cluster as well as on OSS. To read/write data on OSS, write the data path as the OSS path when specifying the input and output paths. For instance:

```
jar ossref://emr/checklist/jars/chengtao/hadoop/hadoop-mapreduce-
examples-2.6.0.jar randomtextwriter -D mapreduce.randomtextwriter.
totalbytes=320000 oss://emr/checklist/data/chengtao/hadoop/Wordcount/
Input
```

19.5.2 Configure a Hive job

When you apply for clusters in E-MapReduce, you are provided with a Hive environment by default. Using Hive, you can create and operate tables and data.

Procedure

1. Prepare the Hive script in advance. For example:

```
USE DEFAULT;
 DROP TABLE uservisits;
 CREATE EXTERNAL TABLE IF NOT EXISTS uservisits  (sourceIP STRING,
destURL STRING,visitDate STRING,adRevenue DOUBLE,user
 Agent STRING,countryCode STRING,languageCode STRING,searchWord
STRING,duration INT ) ROW FORMAT DELIMITED FIELDS TERMI
 NATED BY ',' STORED AS SEQUENCEFILE LOCATION '/HiBench/Aggregation/
Input/uservisits';
 DROP TABLE uservisits_aggre;
 CREATE EXTERNAL TABLE IF NOT EXISTS uservisits_aggre ( sourceIP
STRING, sumAdRevenue DOUBLE) STORED AS SEQUENCEFILE LO
 CATION '/HiBench/Aggregation/Output/uservisits_aggre';
 INSERT OVERWRITE TABLE uservisits_aggre SELECT sourceIP, SUM(
adRevenue) FROM uservisits GROUP BY sourceIP;
```

2. Save this script into a script file, such as *uservisits_aggre_hdfs.hive*, and upload it to an OSS directory (for example, *oss://path/to/uservisits_aggre_hdfs.hive*).
3. Log on to the *Alibaba Cloud E-MapReduce console*.
4. At the top of the navigation bar, click Data Platform.

5. In the Actions column, click Design Workflow next to the specified project.
6. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.
7. In the New Job dialog box, enter the job name and description.
8. Select the Hive job type to create a Hive job. This type of job is submitted in the background using the following method.

   ```
   hive [user provided parameters]
   ```

9. Click OK.

   > **Note:**
   > You can also create subfolders, rename folders, and delete folders by right-clicking on them.

10. Enter the parameters in the Content field after the Hive commands. For example, if you want to use a Hive script uploaded to OSS, enter the following.

    ```
    -f ossref://path/to/uservisits_aggre_hdfs.hive
    ```

    You can also click Select OSS path to view and select from OSS. The system will automatically complete the path of the Hive script on OSS. Switch the Hive script prefix to ossref by clicking Switch resource type. This ensures that the file is correctly downloaded by E-MapReduce.

11. Click Save to complete the Hive job configuration.

19.5.3 Configure a Pig job

When you apply for clusters in E-MapReduce, a Pig environment is provided by default. Using Pig, you can create and operate tables and data.

Procedure

1. Prepare the Pig script in advance. For example:

   ~~~
   ```shell
    /*
    * Licensed to the Apache Software Foundation (ASF) under one
    * or more contributor license agreements.  See the NOTICE file
    * distributed with this work for additional information
    * regarding copyright ownership.  The ASF licenses this file
    * to you under the Apache License, Version 2.0 (the
    * "License"); you may not use this file except in compliance
    * with the License.  You may obtain a copy of the License at
    *
    *     http://www.apache.org/licenses/LICENSE-2.0
    *
   ~~~

```
 * Unless required by applicable law or agreed to in writing,
software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
implied.
 * See the License for the specific language governing permissions
and
 * limitations under the License.
 */
 -- Query Phrase Popularity (Hadoop cluster)
 -- This script processes a search query log file from the Excite
search engine and finds search phrases that occur with particular
high frequency during certain times of the day.
 -- Register the tutorial JAR file so that the included UDFs can be
called in the script.
 REGISTER oss://emr/checklist/jars/chengtao/pig/tutorial.jar;
 -- Use the  PigStorage function to load the excite log file into
the “raw” bag as an array of records.
 -- Input: (user,time,query)
 raw = LOAD 'oss://emr/checklist/data/chengtao/pig/excite.log.bz2'
USING PigStorage('\t') AS (user, time, query);
 -- Call the NonURLDetector UDF to remove records if the query field
 is empty or a URL.
 clean1 = FILTER raw BY org.apache.pig.tutorial.NonURLDetector(query
);
 -- Call the ToLower UDF to change the query field to lowercase.
 clean2 = FOREACH clean1 GENERATE user, time, org.apache.pig.
tutorial.ToLower(query) as query;
 -- Because the log file only contains queries for a single day, we
are only interested in the hour.
 -- The excite query log timestamp format is YYMMDDHHMMSS.
 -- Call the ExtractHour UDF to extract the hour (HH) from the time
field.
 houred = FOREACH clean2 GENERATE user, org.apache.pig.tutorial.
ExtractHour(time) as hour, query;
 -- Call the NGramGenerator UDF to compose the n-grams of the query.
 ngramed1 = FOREACH houred GENERATE user, hour, flatten(org.apache.
pig.tutorial.NGramGenerator(query)) as ngram;
 -- Use the  DISTINCT command to get the unique n-grams for all
records.
 ngramed2 = DISTINCT ngramed1;
 -- Use the  GROUP command to group records by n-gram and hour.
 hour_frequency1 = GROUP ngramed2 BY (ngram, hour);
 -- Use the  COUNT function to get the count (occurrences) of each n
-gram.
 hour_frequency2 = FOREACH hour_frequency1 GENERATE flatten($0),
COUNT($1) as count;
 -- Use the  GROUP command to group records by n-gram only.
 -- Each group now corresponds to a distinct n-gram and has the
count for each hour.
 uniq_frequency1 = GROUP hour_frequency2 BY group::ngram;
 -- For each group, identify the hour in which this n-gram is used
with a particularly high frequency.
 -- Call the ScoreGenerator UDF to calculate a "popularity" score
for the n-gram.
 uniq_frequency2 = FOREACH uniq_frequency1 GENERATE flatten($0),
flatten(org.apache.pig.tutorial.ScoreGenerator($1));
 -- Use the  FOREACH-GENERATE command to assign names to the fields
.
 uniq_frequency3 = FOREACH uniq_frequency2 GENERATE $1 as hour, $0
as ngram, $2 as score, $3 as count, $4 as mean;
 -- Use the  FILTER command to move all records with a score less
than or equal to 2.0.
 filtered_uniq_frequency = FILTER uniq_frequency3 BY score > 2.0;
 -- Use the  ORDER command to sort the remaining records by hour and
 score.
 ordered_uniq_frequency = ORDER filtered_uniq_frequency BY hour,
score;
 -- Use the  PigStorage function to store the results.
 -- Output: (hour, n-gram, score, count, average_counts_among
_all_hours)
 STORE ordered_uniq_frequency INTO 'oss://emr/checklist/data/
chengtao/pig/script1-hadoop-results' USING PigStorage();
 ```
```

2. Save this script into a script file, such as *script1-hadoop-oss.pig*, and upload it to an OSS directory (for example, *oss://path/to/script1-hadoop-oss.pig*).
3. Log on to the *Alibaba Cloud E-MapReduce console*.
4. At the top of the navigation bar, click Data Platform.
5. In the Actions column, click Design Workflow next to the specified project.
6. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.
7. In the New Job dialog box, enter the job name and description.
8. Select the Pig job type to create a Pig job. This type of job is submitted in the background using the following method.

```
pig [user provided parameters]
```

9. Click OK.

> Note:
>
> You can also create subfolders, rename folders, and delete folders by right-clicking on them.

10. Enter the parameters in the Content field after the Pig commands. For example, if you want to use a Pig script uploaded to OSS, enter the following.

```
-x mapreduce ossref://emr/checklist/jars/chengtao/pig/script1-hadoop
-oss.pig
```

You can click Select OSS path to view and select from OSS. The system will automatically complete the path of Pig script on OSS. Switch the Pig script prefix to ossref by clicking Switch resource type. This ensures that the file is correctly downloaded by E-MapReduce.

11. Click Save to complete the Pig job configuration.
```

## 19.5.4 Configure a Spark job

In this tutorial, you will learn how to configure a Spark job.

### Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Data Platform.
3. In the Actions column, click Design Workflow next to the specified project.
4. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.
5. In the New Job dialog box, enter the job name and description.
6. Click OK.

   > Note:
   > You can also create subfolders, rename folders, and delete folders by right-clicking on them.

7. Select the Spark job type to create a Spark job. This type of job is submitted in the background using the following method.

   ```
   spark-submit [options] --class [MainClass] xxx.jar args
   ```

8. Enter the parameters in the Content field that are required to submit this job. Only the parameters after `spark-submit` can be entered. The following example shows how to enter the parameters for creating a Spark job and a PySpark job.

   - Create a Spark job

     Create a Spark WordCount job:

     - Job name: WordCount
     - Type: Select Spark
     - Parameters:
       - Enter the following command:

         ```
         spark-submit --master yarn-client --driver-memory 7G --
         executor-memory 5G --executor-cores 1 --num-executors 32 --
         class com.aliyun.emr.checklist.benchmark.SparkWordCount emr
         -checklist_2.10-0.1.0.jar oss://emr/checklist/data/wc oss://
         emr/checklist/data/wc-counts 32
         ```

       - Enter the following in the E-MapReduce job Content field:

         ```
         --master yarn-client --driver-memory 7G --executor-memory 5G
          --executor-cores 1 --num-executors 32 --class com.aliyun.emr
         ```

```
.checklist.benchmark.SparkWordCount ossref://emr/checklist/
jars/emr-checklist_2.10-0.1.0.jar oss://emr/checklist/data/wc
 oss://emr/checklist/data/wc-counts 32
```

> Notice:
>
> Job jar packages are saved in OSS. In the example above, the way to reference the Jar package is *`ossref://emr/checklist/jars/emr-checklist_2.10-0.1.0.jar`*. Click Select OSS path to view and select one from OSS. The system will automatically complete the absolute path of the Spark script on OSS. Switch the default OSS protocol to the ossref protocol.

- Create a PySpark job

  In addition to Scala and Java job types, E-MapReduce also supports Python job types in Spark. Create a Spark K-means job for the Python script:

  - Job name: Python-Kmeans
  - Type: Spark
  - Parameters:

    ```
    --master yarn-client --driver-memory 7g --num-executors 10
     --executor-memory 5g --executor-cores 1 --jars ossref://emr/
    checklist/jars/emr-core-0.1.0.jar ossref://emr/checklist/python
    /wordcount.py oss://emr/checklist/data/kddb 5 32
    ```

  - References of Python script resources are supported, and the ossref protocol is used.
  - For PySpark, the online Python installation kit is not supported.

9. Click Save to complete the Spark job configuration.

19.5.5 Configure a Spark SQL

In this tutorial, you will learn how to configure a Spark SQL job.

> Note:
>
> By default, the mode of Spark SQL used for submitting a job is YARN.

Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Data Platform.
3. In the Actions column, click Design Workflow next to the specified project.
4. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.

5. In the New Job dialog box, enter the job name and description.
6. Click OK.

> Note:
> You can also create subfolders, rename folders, and delete folders by right-clicking on them.

7. Select the Spark SQL job type to create a Spark SQL job. This type of job is submitted in the background using the following method.

```
spark-sql [options] [cli option]
```

8. Enter the parameters in the Content field after the Spark SQL commands.

   - -e option

     -e options can be written to the running SQL by inputting them into the Content field of the job. For example:

     ```
     -e "show databases;"
     ```

   - -f option

     -f options can be used to specify a Spark SQL script file. Uploading well-prepared Spark SQL script files to OSS can provide greater flexibility. We recommend that you use this operation mode. For example:

     ```
     -f ossref://your-bucket/your-spark-sql-script.sql
     ```

9. Click Save to complete Spark SQL job configuration.

19.5.6 Configure a Shell job

In this tutorial, you will learn how to configure a Shell job.

> Notice:
> By default, Shell scripts are currently run by Hadoop. If you need to use the root user, sudo can be used. Use Shell script jobs with caution.

Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Data Platform.
3. In the Actions column, click Design Workflow next to the specified project.

4. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.
5. In the New Job dialog box, enter the job name and description.
6. Select the Shell job type to create a Bash Shell job.
7. Click OK.

> Note:
> You can also create subfolders, rename folders, and delete folders by right-clicking on them.

8. Enter the parameters in the Content field after the Shell commands.

    - -c option

      -c options can be used to set Shell scripts to run by inputting them into the Content field of the job. For example:

      ```
      -c "echo 1; sleep 2; echo 2; sleep 4; echo 3; sleep 8; echo 4;
      sleep 16; echo 5; sleep 32; echo 6; sleep 64; echo 8; sleep 128;
      echo finished"
      ```

    - -f option

      -f options can be used to run Shell script files. By uploading a Shell script file to OSS, Shell scripts on OSS can be defined in the job parameters, making it more flexible than the -c option. For example:

      ```
      -f ossref://mxbucket/sample/sample-shell-job.sh
      ```

9. Click Save to complete Shell job configurations.

19.5.7 Configure a Sqoop job

In this tutorial, you will learn how to configure a Sqoop job.

> Note:
> Only E-MapReduce products with version V1.3.0 or later support the Sqoop job type. Running a Sqoop job on lower versions will fail and errlog will report "Not supported" errors. For more information on parameters, see *Sqoop*.

Procedure

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Data Platform.

3. In the Actions column, click Design Workflow next to the specified project.
4. On the left of the Job Editing page, right-click the folder you want to operate and select New Job.
5. In the New Job dialog box, enter the job name and description.
6. Select the Sqoop job type to create a Sqoop job. This type of job is submitted in the background using the following method.

   ```
   sqoop [args]
   ```
7. Click OK.

   > **Note:**
   >
   > You can also create subfolders, rename folders, and delete folders by right-clicking on them.
8. Enter the parameters in the Content field after the Sqoop commands.
9. Click Save to complete Sqoop job configuration.

19.5.8 Job operations

You can create, clone, modify, and delete jobs.

Job creation

A new job can be created at any time. Currently, a job can only be used in the region where it is created.

Job cloning

Configurations that already exist for a job can be cloned. A cloned job can also only be used in the region where it is created.

Job modification

Before you can modify a job that needs to be added to an execution plan, you must first ensure that the execution plan is not running and that its periodic scheduling is not in progress.

Before you can modify a job that needs to be added to several execution plans, you must first ensure that none of the execution plans are running and that none of their periodic scheduling is in progress. Modifying a job may result in changes to all of the execution plans that use this job.

If you need to debug, we recommend that you perform cloning instead. After you debug, the original jobs in the execution plan are replaced.

Job deletion

As with modification, a job can only be deleted when the execution plan where the job is located is not running and its periodic scheduling is not in progress.

19.5.9 Time and date variables

When you are creating a job, variable wildcards are supported in the job parameters for both time and date.

Variable wildcard format

The format of the variable wildcards supported by E-MapReduce is either `${dateexpr-1d}` or `${dateexpr-1h}`. For example, assuming the current date and time is `2016/04/27 12:08:01`:

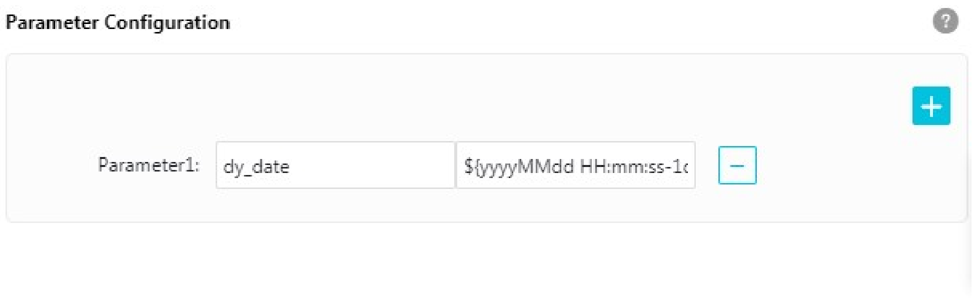
- If `${yyyyMMdd HH:mm:ss-1d}` is displayed, the parameter wildcard is replaced with `20160426 12:08:01` when executed, which is the current date minus one day, and time accurate to the second.
- If `${yyyyMMdd-1d}` is displayed, the parameter wildcard is replaced with `20160426` when executed, which is the current date minus one day.
- If `${yyyyMMdd}` is displayed, the parameter wildcard is replaced with `20160427`, which is the current date.

dateexpr represents the standard format of expressing time. Time is therefore formatted according to this expression and is followed by the amount of time that you want to add or deduct, which can be written as N. For example, `${yyyyMMdd-5d}`, `${yyyyMMdd+5d}`, `${yyyyMMdd+5h}`, or `${yyyyMMdd-5h}`.

> **Note:**
> E-MapReduce currently supports the addition and deduction of hours and days only.

Example

1. Click Job Settings on the top right of the Edit Jobs page.
2. Click the add icon to add new parameters on the Parameter Configuration part, and fill in the parameter according to the Variable wildcard format that mentioned above.



Parameter Configuration

Parameter1: dy_date ${yyyyMMdd HH:mm:ss-1c

3. You can now use the reference of the parameter key in the job editing.

19.6 Old EMR Scheduling (Soon will be unavailable)

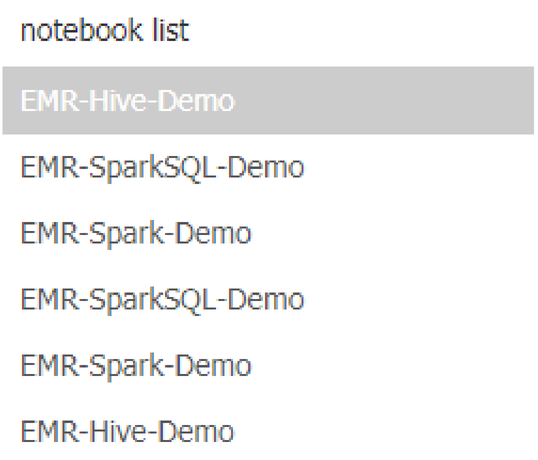
19.6.1 Notebooks

19.6.1.1 Introduction

Notebooks allow you to compile and run Spark, Spark SQL, and Hive SQL tasks directly on the E-MapReduce console. You can then view the running results in the notebook. Notebooks are ideal for processing debugging tasks that require a shorter runtime and whose results need to be viewed directly. For tasks that have a longer runtime and require regular execution, the job and execution plan function must be used. This section describes how to create and run a notebook demo task.

Create a demo task

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Old EMR Scheduling.
3. In the navigation bar on the left, click Notebook.
4. Click New notebook demo.



notebook list

EMR-Hive-Demo

EMR-SparkSQL-Demo

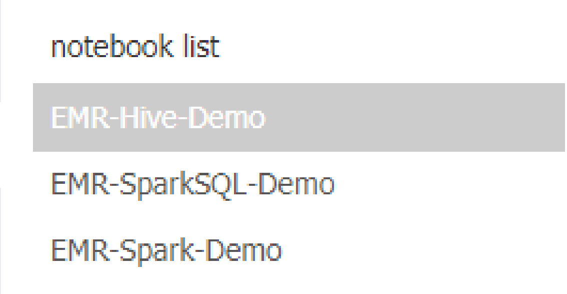
EMR-Spark-Demo

EMR-SparkSQL-Demo

EMR-Spark-Demo

EMR-Hive-Demo

5. A confirmation box is displayed, indicating the required cluster environment. Click OK to create a demo task. Three examples of interactive tasks are created.



notebook list

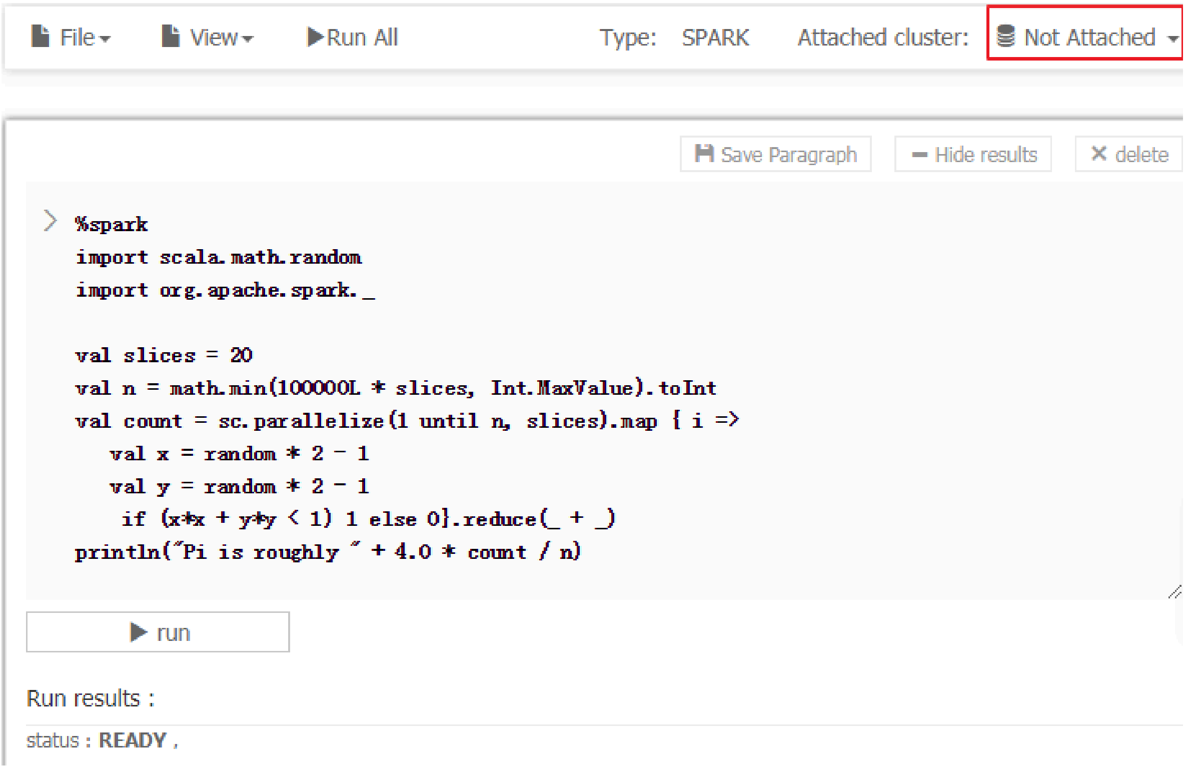
EMR-Hive-Demo

EMR-SparkSQL-Demo

EMR-Spark-Demo

Run a Spark demo task

1. Click EMR-Spark-Demo to display the example of a Spark notebook. Before running the notebook, you need to associate the task to a created cluster. Select a created cluster in the list of available clusters. Note that the associated cluster must be E-MapReduce 2.3 or later and have no less than three nodes, each with at least 4 cores and 8 GB of memory.



Type: SPARK Attached cluster: Not Attached

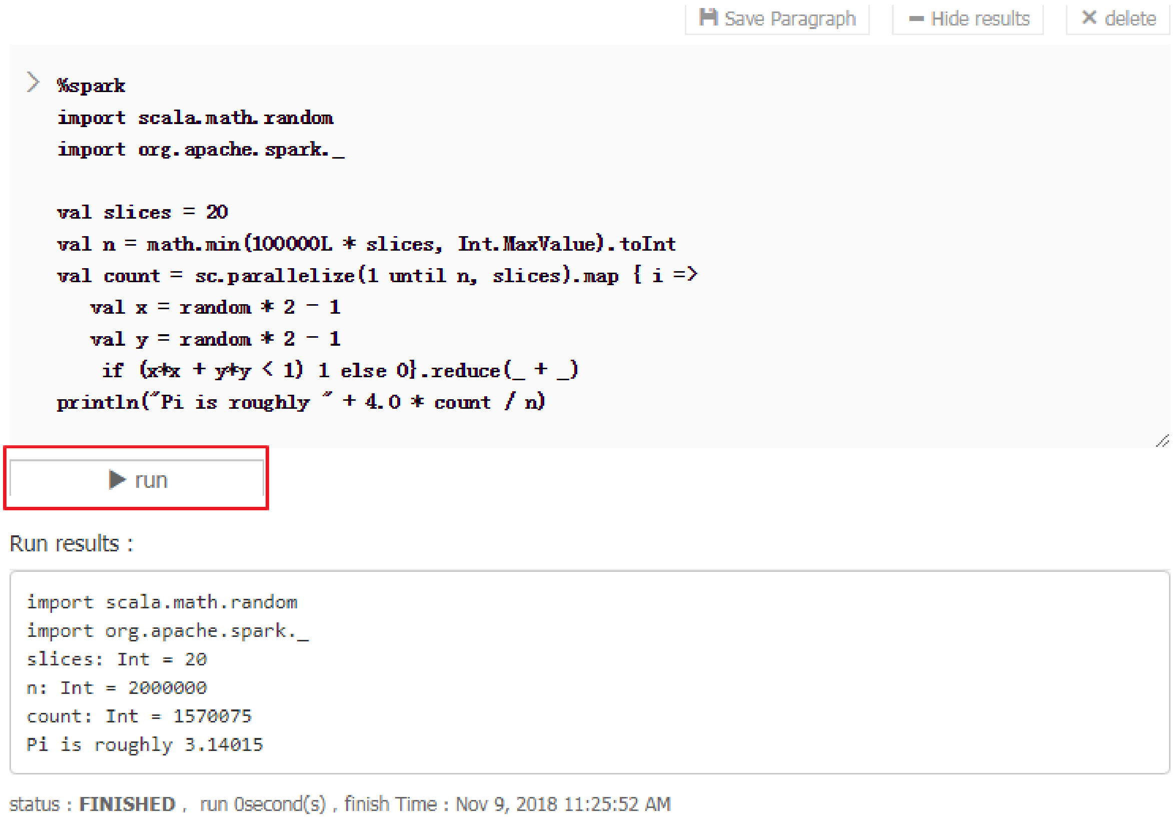
```
%spark
import scala.math.random
import org.apache.spark._

val slices = 20
val n = math.min(100000L * slices, Int.MaxValue).toInt
val count = sc.parallelize(1 until n, slices).map { i =>
   val x = random * 2 - 1
   val y = random * 2 - 1
    if (x*x + y*y < 1) 1 else 0}.reduce(_ + _)
println("Pi is roughly " + 4.0 * count / n)
```

Run results :

status : READY ,

2. After a cluster is associated, click Run. When the associated cluster executes the Spark or Spark SQL notebook for the first time, it takes about one minute to build the Spark context and running environment. It does not need to be built in subsequent executions. The running result is displayed under the Run button.



```
%spark
import scala.math.random
import org.apache.spark._

val slices = 20
val n = math.min(100000L * slices, Int.MaxValue).toInt
val count = sc.parallelize(1 until n, slices).map { i =>
   val x = random * 2 - 1
   val y = random * 2 - 1
    if (x*x + y*y < 1) 1 else 0}.reduce(_ + _)
println("Pi is roughly " + 4.0 * count / n)
```

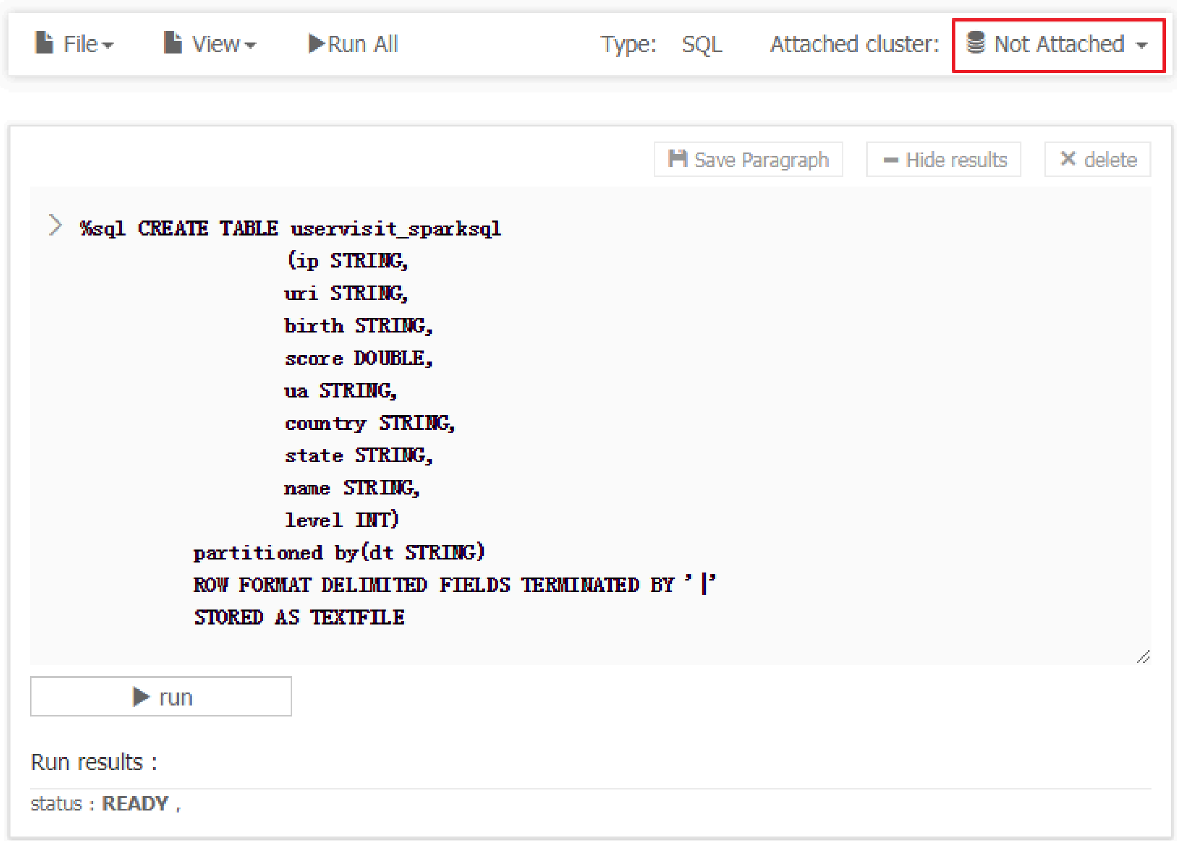
Run results :

```
import scala.math.random
import org.apache.spark._
slices: Int = 20
n: Int = 2000000
count: Int = 1570075
Pi is roughly 3.14015
```

status : **FINISHED** , run 0second(s) , finish Time : Nov 9, 2018 11:25:52 AM

Run a SparkSQL demo task

1. Click EMR-Spark-Demo to display the SparkSQL notebook example. Before running the notebook, you need to associate it to a created cluster. In the upper-right corner, select a created cluster from the list of available clusters.



```
%sql CREATE TABLE uservisit_sparksql
                  (ip STRING,
                  uri STRING,
                  birth STRING,
                  score DOUBLE,
                  ua STRING,
                  country STRING,
                  state STRING,
                  name STRING,
                  level INT)
          partitioned by(dt STRING)
          ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
          STORED AS TEXTFILE
```

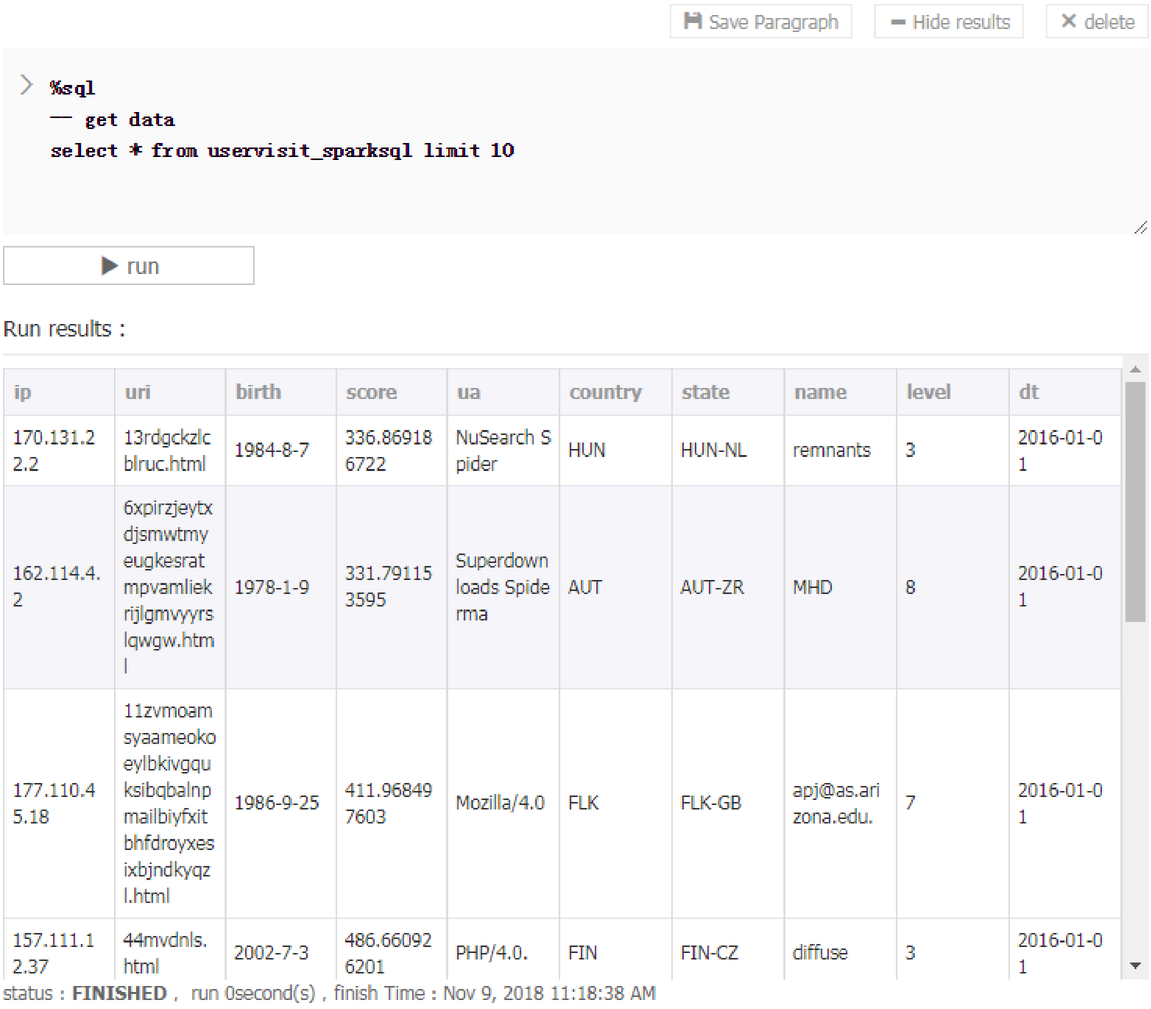
Run results :

status : READY ,

2. The SparkSQL demo contains several demo sections that can be run individually or together by clicking Run All. After running, you can see the returned data results of each section.

Note:

If the section for creating a table is run multiple times, an error is reported indicating that the table already exists.



```
%sql
— get data
select * from uservisit_sparksql limit 10
```

Run results :

ip	uri	birth	score	ua	country	state	name	level	dt
170.131.22.2	13rdgckzlcblruc.html	1984-8-7	336.869186722	NuSearch Spider	HUN	HUN-NL	remnants	3	2016-01-01
162.114.4.2	6xpirzjeytxdjsmwtmyeugkesratmpvamliekrijlgmvyyrslqwgw.html	1978-1-9	331.791153595	Superdownloads Spiderma	AUT	AUT-ZR	MHD	8	2016-01-01
177.110.45.18	11zvmoamsyaameokoeylbkivgquksibqbalnpmailbiyfxitbhfdroyxesixbjndkyqzl.html	1986-9-25	411.968497603	Mozilla/4.0	FLK	FLK-GB	apj@as.arizona.edu.	7	2016-01-01
157.111.12.37	44mvdnls.html	2002-7-3	486.660926201	PHP/4.0.	FIN	FIN-CZ	diffuse	3	2016-01-01

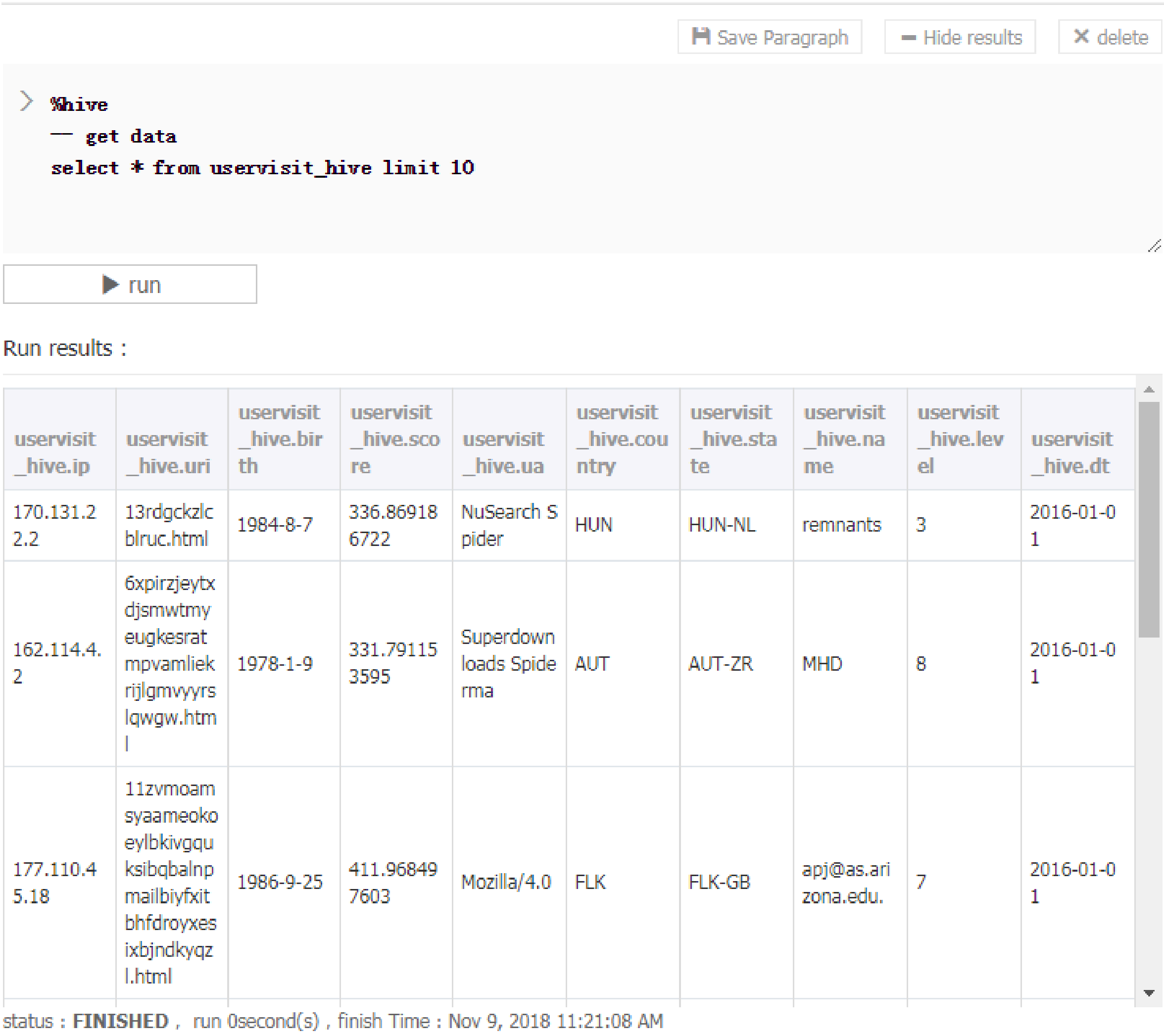
status : **FINISHED** , run 0second(s) , finish Time : Nov 9, 2018 11:18:38 AM

Run a Hive demo task

1. Click EMR-Hive-Demo to display the Hive notebook example. Before running the notebook, you need to associate it to a created cluster. In the upper-right corner, select a created cluster from the list of available clusters.
2. The Hive demo task contains several demo sections that can be run individually or together by clicking Run All. After running, you can see the returned data results of each section.

Note:

- When the associated cluster executes the Hive notebook for the first time, it takes a few seconds to build the Hive client running environment. It does not need to be built in subsequent executions.
- If the section for creating a table is run multiple times, an error is reported indicating that the table already exists.



```
%hive
— get data
select * from uservisit_hive limit 10
```

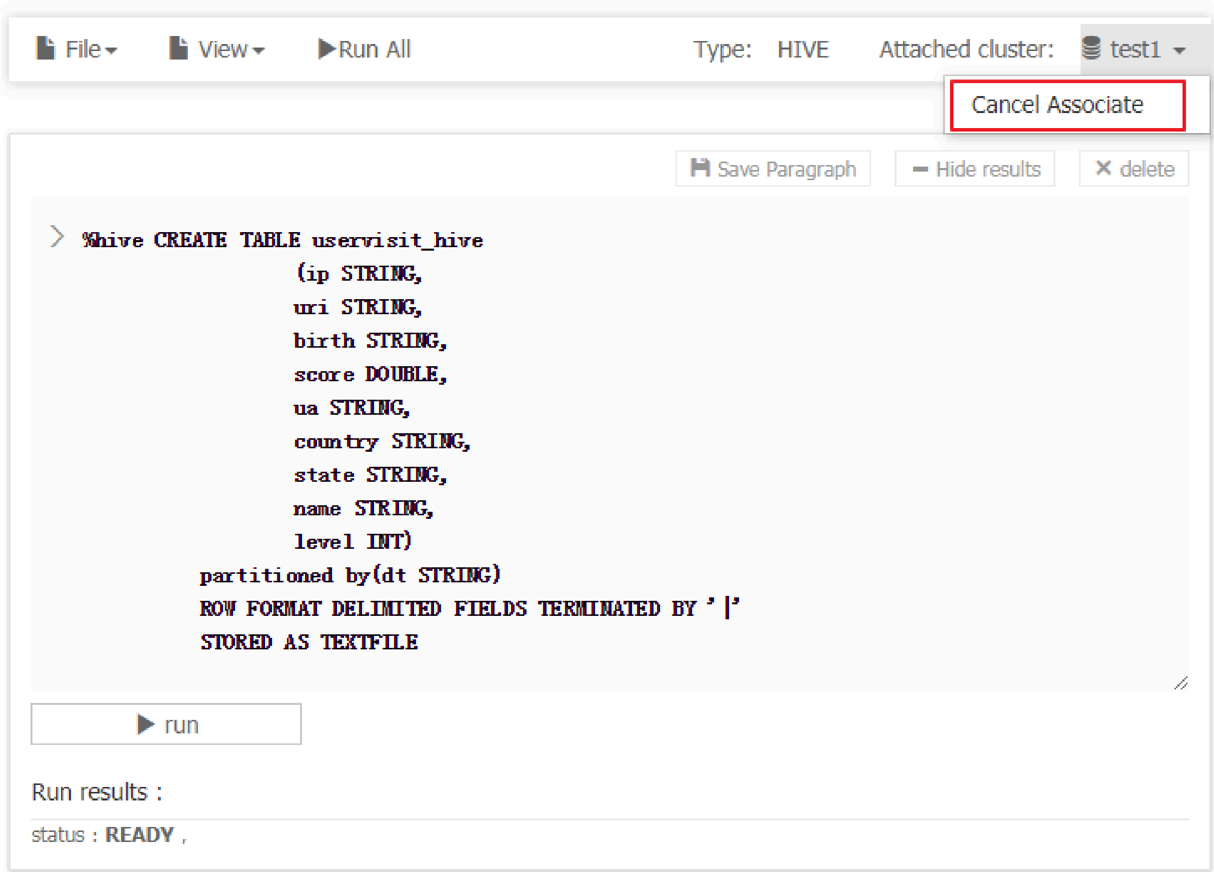
Run results :

uservisit_hive.ip	uservisit_hive.uri	uservisit_hive.birth	uservisit_hive.score	uservisit_hive.ua	uservisit_hive.country	uservisit_hive.state	uservisit_hive.name	uservisit_hive.level	uservisit_hive.dt
170.131.22.2	13rdgckzlcblruc.html	1984-8-7	336.869186722	NuSearch Spider	HUN	HUN-NL	remnants	3	2016-01-01
162.114.4.2	6xpirzjeytxdjsmwtmyeugkesratmpvamliekrijlgmvyyrslqwgw.html	1978-1-9	331.791153595	Superdownloads Spiderma	AUT	AUT-ZR	MHD	8	2016-01-01
177.110.45.18	11zvmoamsyaameokoeylbkivgquksibqbalnpmailbiyfxitbhfdroyxesixbjndkyqzl.html	1986-9-25	411.968497603	Mozilla/4.0	FLK	FLK-GB	apj@as.arizona.edu.	7	2016-01-01

status : **FINISHED** , run 0second(s) , finish Time : Nov 9, 2018 11:21:08 AM

Cancel the association with clusters

After a notebook is run in a cluster, the cluster creates a process for caching some context running environments in order to ensure a quick response upon re-execution. If you do not need to execute other notebooks, and you want to release the cluster resources occupied by caching, you can disassociate all interactive tasks that have been run from the associated clusters. In this way, you can release the memory resources occupied on the original associated clusters.



```
%hive CREATE TABLE uservisit_hive
                 (ip STRING,
                 uri STRING,
                 birth STRING,
                 score DOUBLE,
                 ua STRING,
                 country STRING,
                 state STRING,
                 name STRING,
                 level INT)
         partitioned by(dt STRING)
         ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
         STORED AS TEXTFILE
```

Run results :

status : READY ,

19.6.1.2 Operations

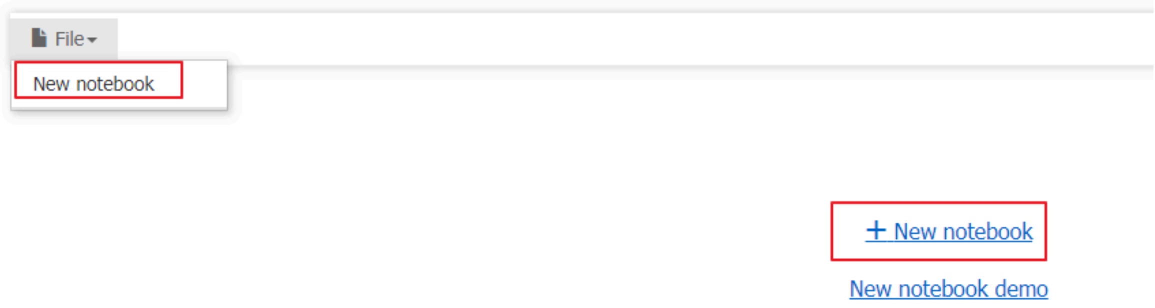
This section details how to perform a number of notebook operations, including how to create a new notebook task on the E-MapReduce console.

Create a new notebook task

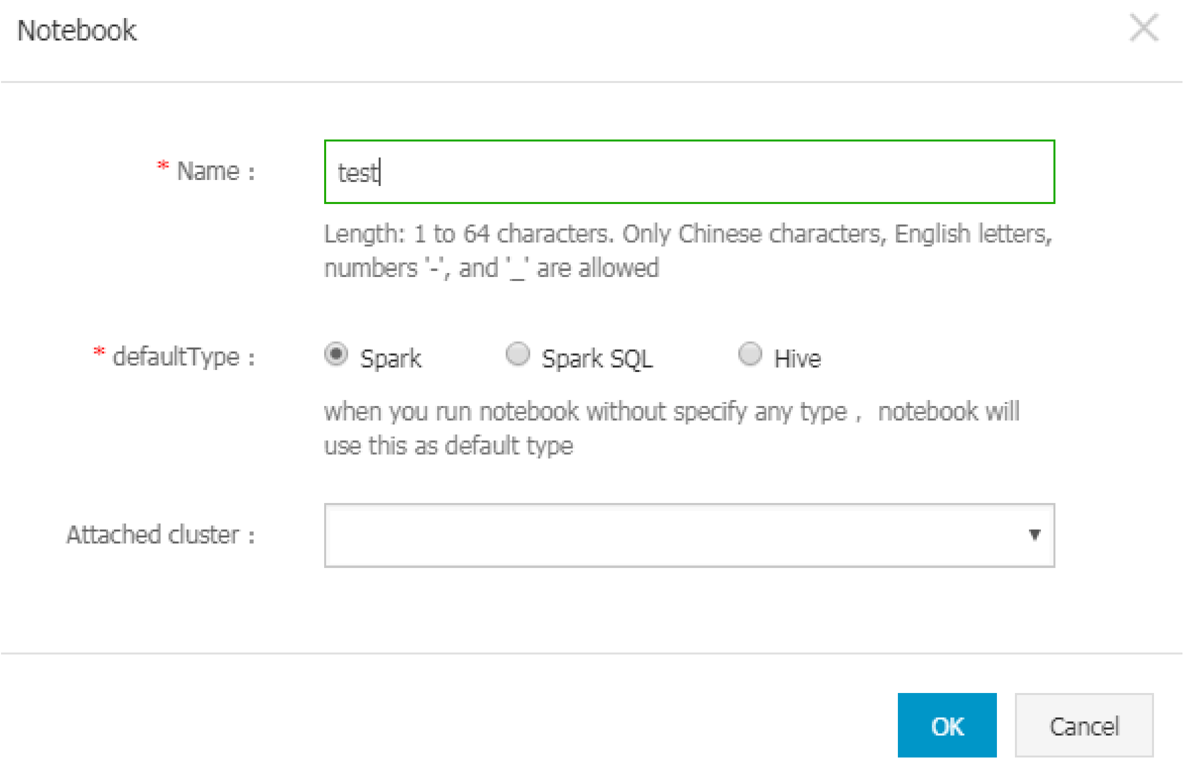
Note:

The cluster on which an interactive task is run must be E-MapReduce 2.3 or later and have no less than three nodes, each with at least 4 cores and 8 GB of memory.

1. Log on to the *Alibaba Cloud E-MapReduce console*.
2. At the top of the navigation bar, click Old EMR Scheduling.
3. In the navigation bar on the left, click Notebook.
4. Click New notebook or File > -New notebook.



5. Enter a name and select the default type. Associating a cluster is optional. Click OK to create a notebook.



Notebook

* Name : test

Length: 1 to 64 characters. Only Chinese characters, English letters, numbers '-', and '_' are allowed

* defaultType : ● Spark ○ Spark SQL ○ Hive

when you run notebook without specify any type , notebook will use this as default type

Attached cluster :

Three types of notebook task are supported. Spark can be used to write Scala code, Spark SQL can be used to write SQL statements supported by Spark, and Hive can be used to write SQL statements supported by Hive.

6. An associated cluster must be E-MapReduce 2.3 or later and have no less than three nodes, each with at least 4 cores and 8 GB of memory. You can also associate the cluster before running the task.

Up to 20 interactive tasks can be created in one account.

Enter and save a section

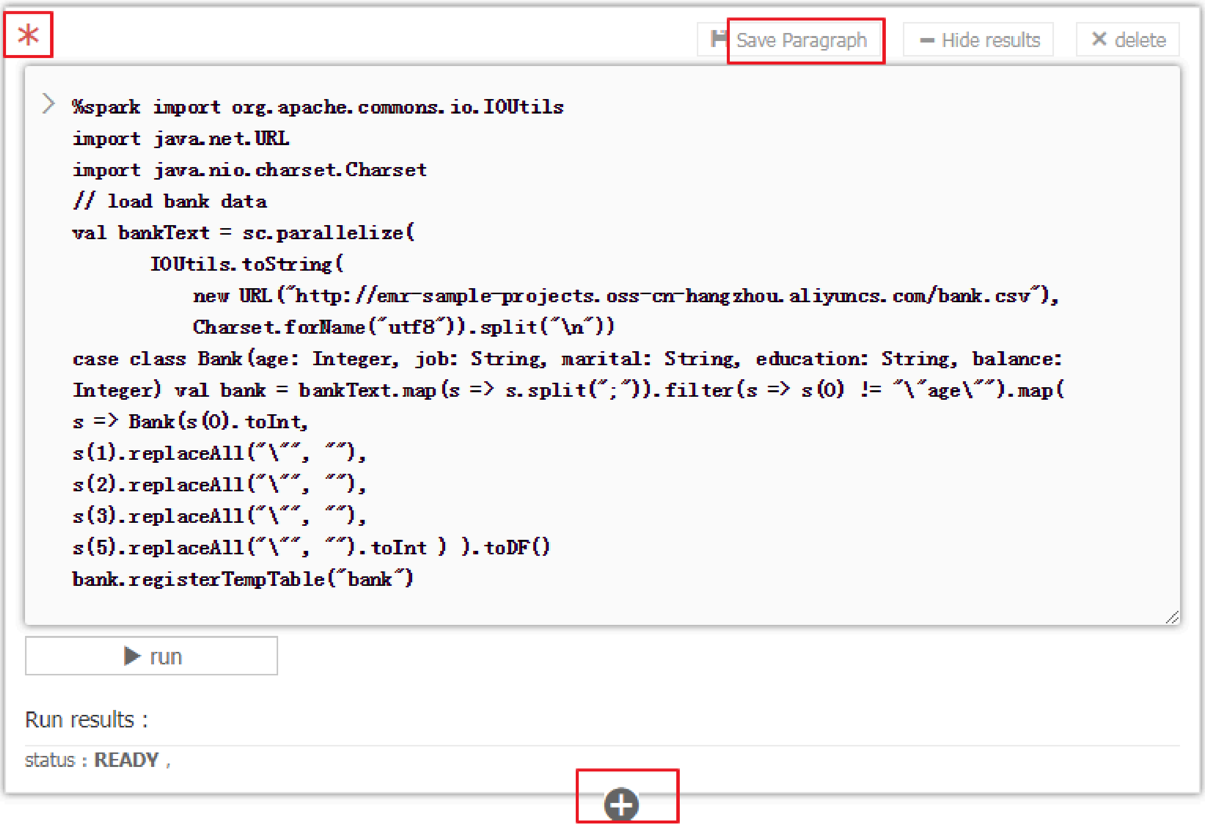
A paragraph is the smallest unit for running a notebook. Multiple paragraphs can be entered into a notebook. Each paragraph starts with either `%spark`, `%sql`, or `%hive`, indicating whether it is a Scala code paragraph, Spark SQL paragraph, or Hive SQL paragraph. The type prefix is separated by a blank space or by line feed and actual content. If the type prefix is not specified, the default type of the interactive task is used as the run type of this paragraph.

The following example shows how to create a temporary Spark table:

Paste the following code into the section and a red * symbol is displayed, indicating that this notebook has been changed. Click Save Paragraph or run to save the modifications to the paragraph. Click + under the paragraph to create a new paragraph. Up to 30 paragraphs can be created in one notebook.

```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
// load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.
com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education
: String, balance: Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) ! = "\"age
\"").map(
    s => Bank(s(0).toInt,
            s(1).replaceAll("\"", ""),
            s(2).replaceAll("\"", ""),
            s(3).replaceAll("\"", ""),
            s(5).replaceAll("\"", "").toInt
        )
).toDF()
```

```
bank.registerTempTable("bank")
```



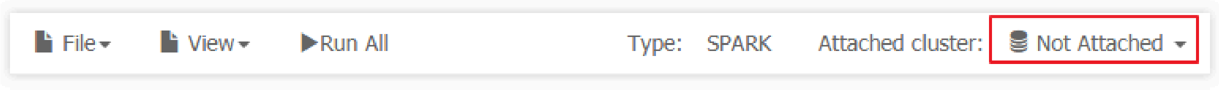
```
%spark import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
// load bank data
val bankText = sc.parallelize(
       IOUtils.toString(
           new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
           Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance:
Integer) val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
s => Bank(s(0).toInt,
s(1).replaceAll("\"", ""),
s(2).replaceAll("\"", ""),
s(3).replaceAll("\"", ""),
s(5).replaceAll("\"", "").toInt ) ).toDF()
bank.registerTempTable("bank")
```

Run results :

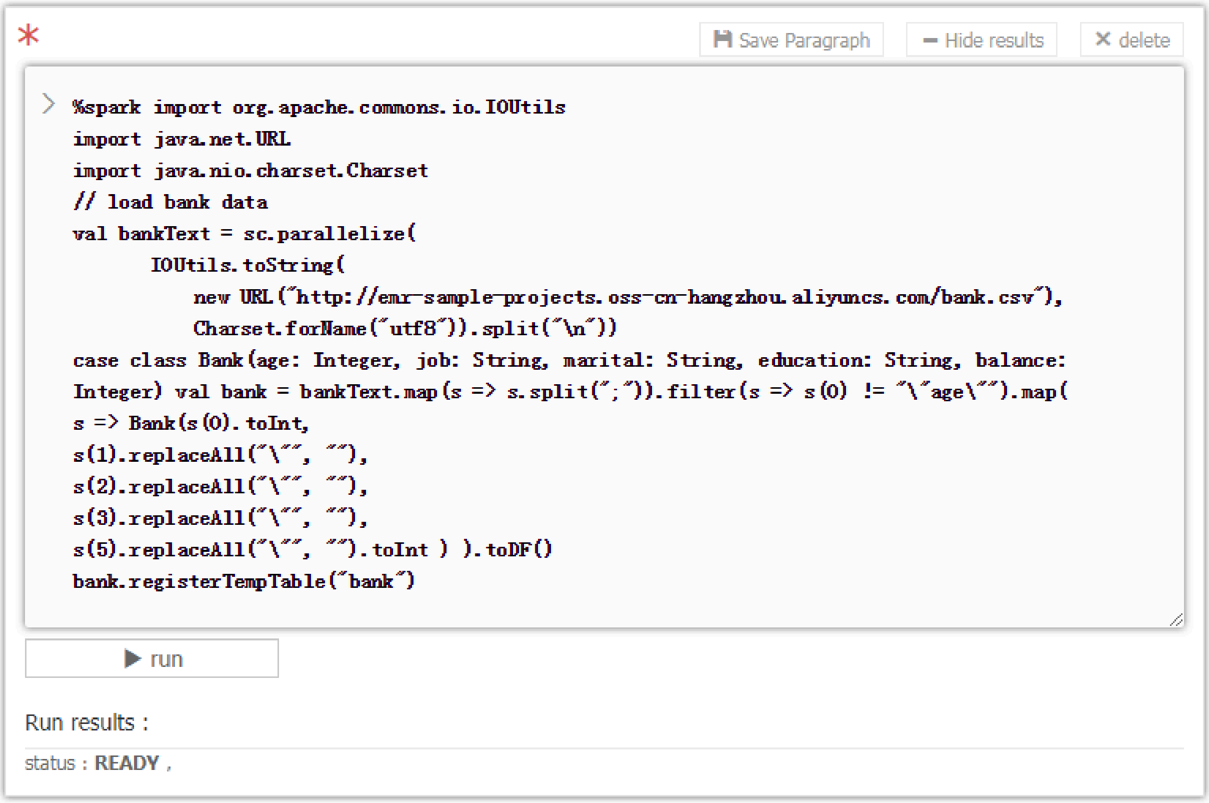
status : READY ,

Run a paragraph

Before running a notebook, you must first associate it to a created cluster. If a created notebook is not associated with a cluster, Not Attached is displayed in the upper-right corner of the page. Click it to select a cluster from the list of available clusters. Note that the associated cluster must be E-MapReduce 2.3 or later and have no less than three nodes, each with at least 4 cores and 8 GB of memory.



Type: SPARK Attached cluster: Not Attached



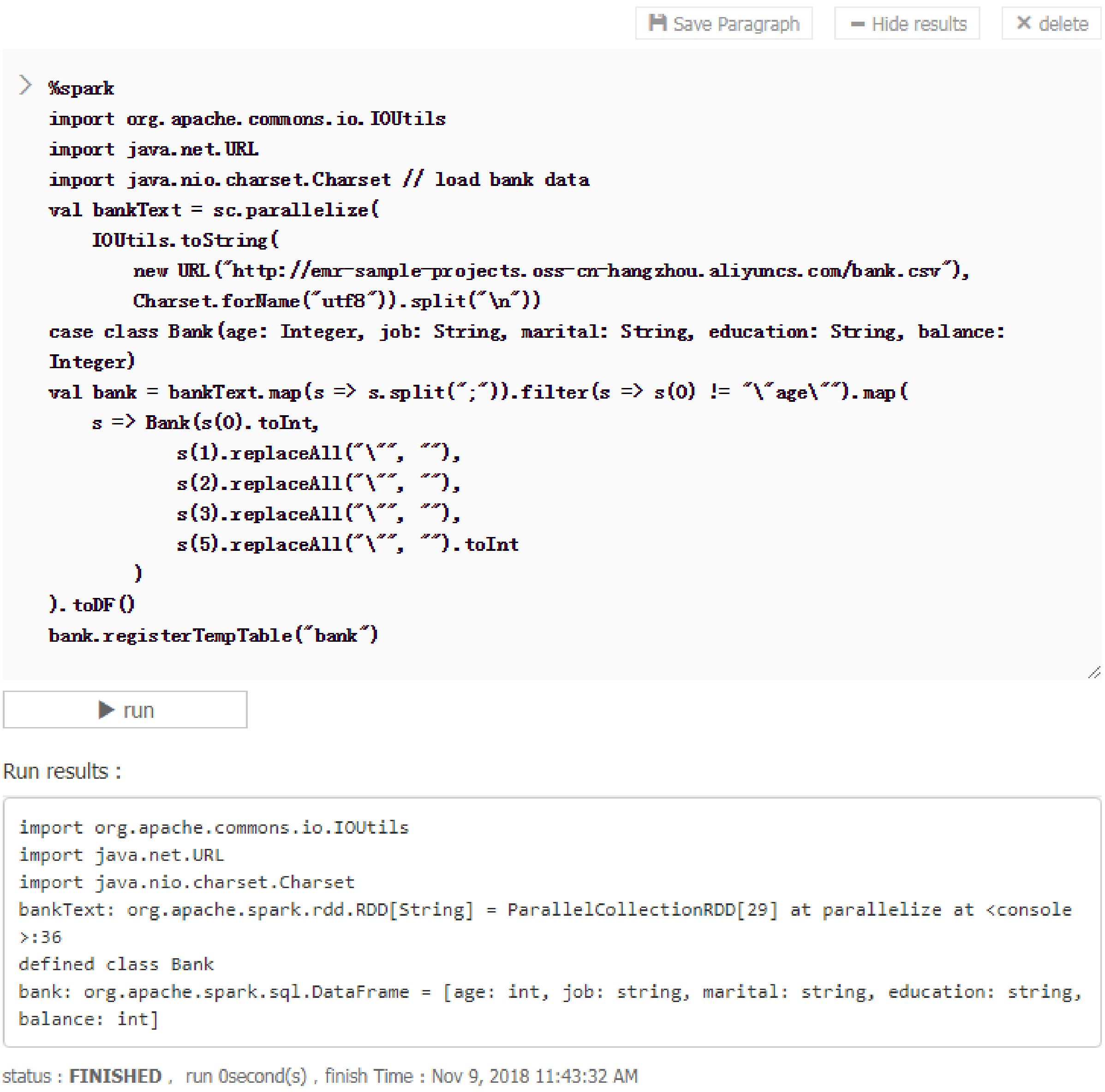
```
%spark import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
// load bank data
val bankText = sc.parallelize(
        IOUtils.toString(
            new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
            Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance:
Integer) val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
s => Bank(s(0).toInt,
s(1).replaceAll("\"", ""),
s(2).replaceAll("\"", ""),
s(3).replaceAll("\"", ""),
s(5).replaceAll("\"", "").toInt ) ).toDF()
bank.registerTempTable("bank")
```

Run results :

status : READY ,

Click Run to save the current paragraph and run the content. If this is the last paragraph, a new paragraph is created automatically.

PENDING indicates that the paragraph has not run yet, RUNNING indicates that the paragraph is running, FINISHED indicates that the running has finished, and ERROR indicates that an error has occurred. The running result is displayed beneath the Run button. During running, you can click Cancel beneath the Run button to cancel running. ABORT is displayed after running has been canceled.



```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset // load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance:
Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
    s => Bank(s(0).toInt,
            s(1).replaceAll("\"", ""),
            s(2).replaceAll("\"", ""),
            s(3).replaceAll("\"", ""),
            s(5).replaceAll("\"", "").toInt
        )
).toDF()
bank.registerTempTable("bank")
```

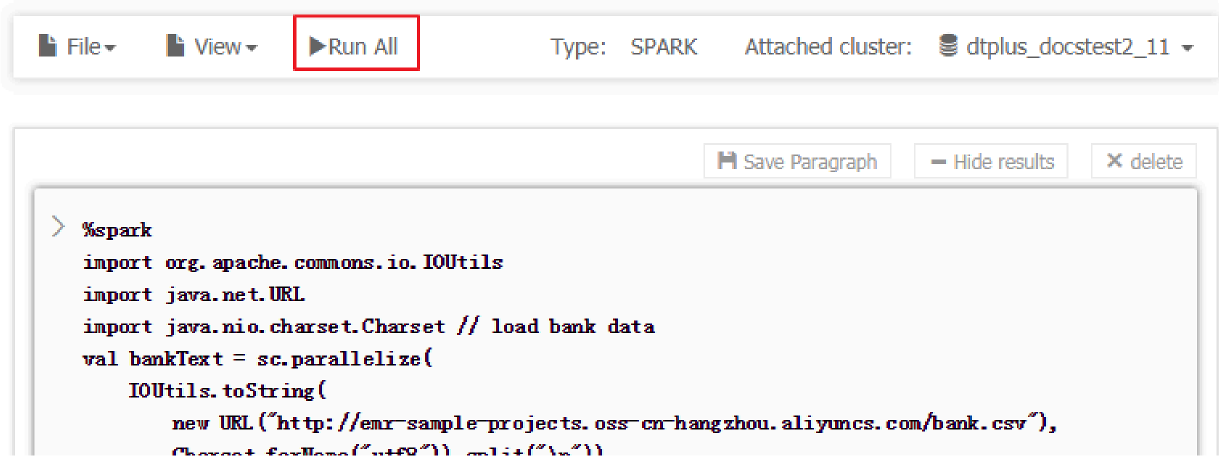
```
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
bankText: org.apache.spark.rdd.RDD[String] = ParallelCollectionRDD[29] at parallelize at <console
>:36
defined class Bank
bank: org.apache.spark.sql.DataFrame = [age: int, job: string, marital: string, education: string,
balance: int]
```

status : **FINISHED** , run 0second(s) , finish Time : Nov 9, 2018 11:43:32 AM

The paragraph can be run multiple times, but only the result of the last running is retained. You cannot modify the content of a paragraph while it is running. It can only be modified after the running has finished.

Run all

For a notebook, you can click Run All on the menu bar to run all paragraphs. The paragraphs are then submitted sequentially for running. Different types have independent execution queues. If a notebook contains multiple paragraph types, the order for executing them on the cluster is decided based on type after they have been submitted sequentially. Spark and Spark SQL support one-by-one execution. Hive supports concurrent execution, with the maximum number of concurrently executed interactive paragraphs on the same cluster is 10. Note that all concurrently executed paragraphs are restricted by cluster resources. If the cluster size is small and many paragraphs need to be executed concurrently, the paragraphs still need to queue in YARN.

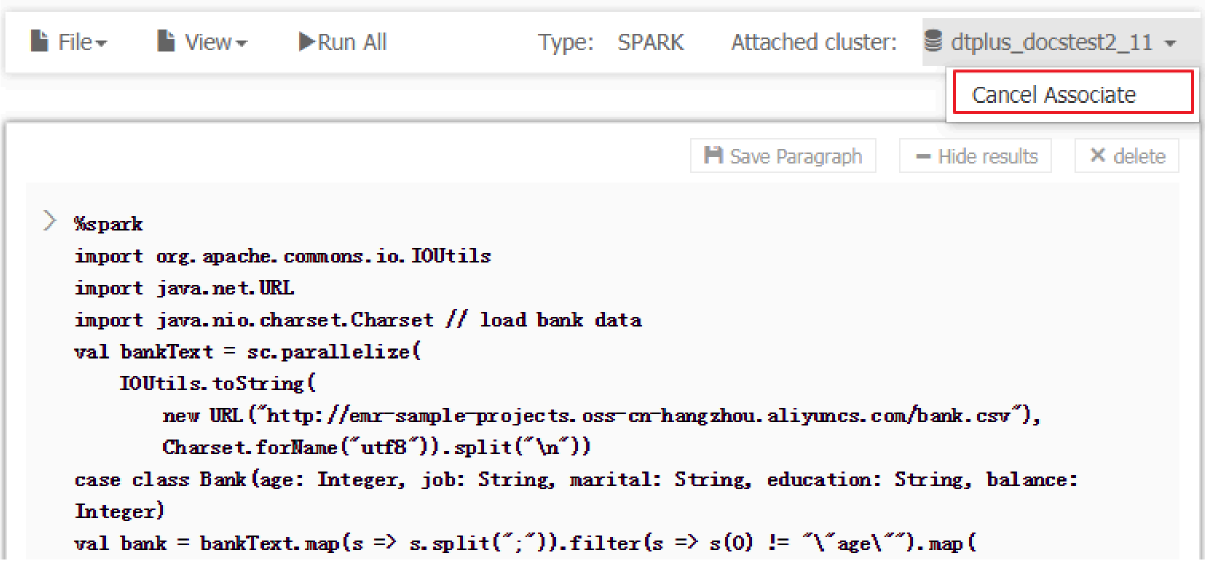


Type: SPARK Attached cluster: dtplus_docstest2_11

```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset // load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
```

Cancel the association with clusters

After a notebook is run in a cluster, the cluster creates a process for caching some context running environments to ensure a quick response upon re-execution. If you do not need to run other notebooks, and you want to release the cluster resources occupied by caching, you can disassociate all notebooks that have been run from the associated clusters. In this way, you can release the memory resources occupied on the original associated clusters.

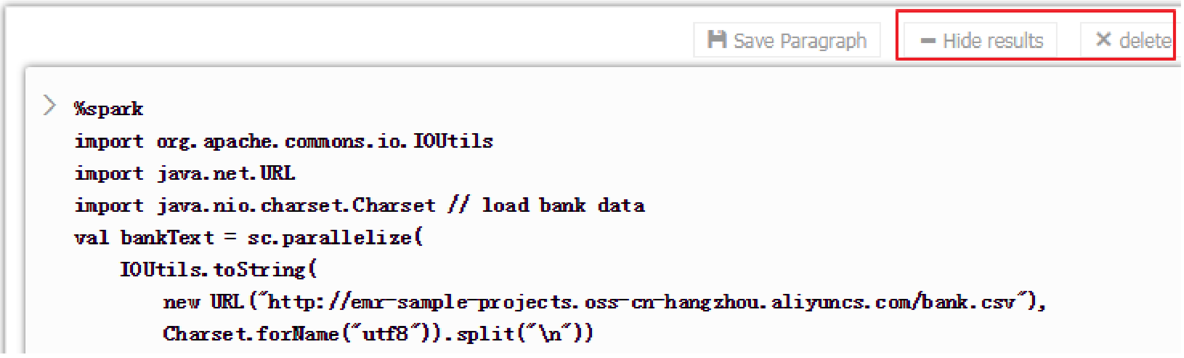


Type: SPARK Attached cluster: dtplus_docstest2_11

```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset // load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance:
Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
```

Other operations

- Paragraph operations



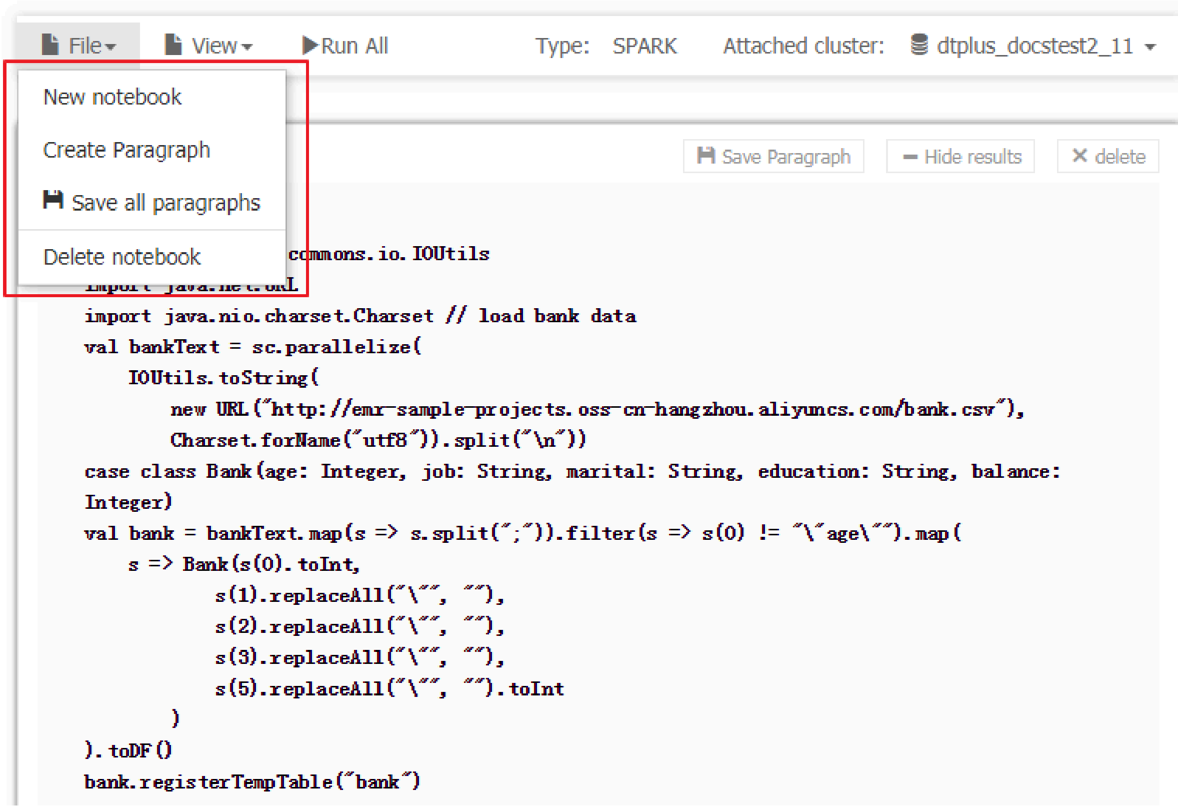
```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset // load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
```

- Hide and display the results

  Hide the paragraph results and only display the entered content of the paragraph.
- Delete a paragraph

  Delete the current paragraph. Paragraphs that are running can also be deleted.

· File menu



```
commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset // load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education: String, balance:
Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) != "\"age\"").map(
    s => Bank(s(0).toInt,
            s(1).replaceAll("\"", ""),
            s(2).replaceAll("\"", ""),
            s(3).replaceAll("\"", ""),
            s(5).replaceAll("\"", "").toInt
        )
).toDF()
bank.registerTempTable("bank")
```

- New notebook

  Create a notebook and switch to the created notebook interface.
- Create Paragraph

Add a new paragraph to the end of a notebook. A notebook can have up to 30 paragraphs.

- Save all paragraphs

  Save all modified paragraphs.

- Delete notebook

  Delete the current notebook. If a cluster has been associated, it will be disassociated.

- View

  Only display codes or display codes and results.

19.6.1.3 Examples

19.6.1.3.1 Query bank employee information

1. Create a temporary table

```
%spark
import org.apache.commons.io.IOUtils
import java.net.URL
import java.nio.charset.Charset
// Zeppelin creates and injects sc (SparkContext) and sqlContext (
HiveContext or SqlContext)
// So you don't need create them manually
// load bank data
val bankText = sc.parallelize(
    IOUtils.toString(
        new URL("http://emr-sample-projects.oss-cn-hangzhou.aliyuncs.
com/bank.csv"),
        Charset.forName("utf8")).split("\n"))
case class Bank(age: Integer, job: String, marital: String, education
: String, balance: Integer)
val bank = bankText.map(s => s.split(";")).filter(s => s(0) ! = "\"age
\"").map(
    s => Bank(s(0).toInt,
            s(1).replaceAll("\"", ""),
            s(2).replaceAll("\"", ""),
            s(3).replaceAll("\"", ""),
            s(5).replaceAll("\"", "").toInt
        )
).toDF()
bank.registerTempTable("bank")
```

2. Query the table structure

```
%sql
```

```
desc bank
```

3. Query the number of employees in each age group under 30

```
%sql select age, count(1) value from bank where age < 30 group by age
order by age
```

4. Query the information of employees younger than or equal to 20

```
%sql select * from bank where age <= 20
```

19.6.1.3.2 Video playback data

Preparations

In this example, you need to download data from OSS and upload it to your OSS bucket. This data includes:

- *User table sample data*
- *Video table sample data*
- *Video playback table sample data*

Upload this sample data respectively to the specified UserInfo, Videoinfo, and Playvideo on your OSS bucket. For example, upload the data to the Demo or UserInfo directory under Bucket Example.

In the following table, replace the SQL [bucketname] with your bucket name, replace [region] with your OSS region name, and replace [bucketpath] with your specified OSS path prefix, such as Demo.

1. Create a user table

```
%hive
CREATE EXTERNAL TABLE user_info(id int,sex int,age int, marital_st
atus int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION 'oss
://[bucketname].oss-cn-[region]-internal.aliyuncs.com/[bucketpath]/
userinfo'
```

2. Create a video table

```
%hive
CREATE EXTERNAL TABLE video_info(id int,title string,type string) ROW
 FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION 'oss://[bucketname
].oss-cn-[region]-internal.aliyuncs.com/[bucketpath]/videoinfo'
```

3. Create a video playback table

```
%hive
CREATE EXTERNAL TABLE play_video(user_id int,video_id int, play_time
 bigint) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION 'oss
```

```
://[bucketname].oss-cn-[region]-internal.aliyuncs.com/[bucketpath]/
playvideo'
```

4. Count the user tables

```
%sql select count(*) from user_info
```

5. Count the video tables

```
%sql select count(*) from video_info
```

6. Count the video playback tables

```
%sql select count(*) from play_video
```

7. Count the video playbacks for each video type

```
%sql select video.type, count(video.type) as count from play_video
 play join video_info video on (play.video_id = video.id) group by
video.type order by count desc
```

8. Display the video information for the top 10 video playbacks

```
%sql select video.id, video.title, video.type, video_count.count from
 (select video_id, count(video_id) as count from play_video group by
 video_id order by count desc limit 10) video_count join video_info
video on (video_count.video_id = video.id) order by count desc
```

9. Display the age of the viewers watching the video with the most video playbacks

```
%sql select age , count(*) as count  from (select distinct(user_id)
from play_video  where video_id =49 ) play join user_info userinfo on
 (play.user_id = userinfo.id) group by userinfo.age
```

10. Display the gender, age, and marital status of the viewers watching the video with the most video playbacks

```
%sql select if(sex=0,'Female','Male') as title, count(*) as count,
 'Gender' as type from (select distinct(user_id) from play_video
where video_id =49 ) play join user_info userinfo on (play.user_id =
userinfo.id) group by userinfo.sex
union all
select case when userinfo.age<15 then 'Less than 15' when age<25 then
 '15-25' when age<35 then '25-35' else 'More than 35' end  , count
(*) as count, 'Age Group' as type from (select distinct(user_id) from
 play_video  where video_id =49) play join user_info userinfo on (play
.user_id = userinfo.id) group by case when userinfo.age<15 then 'Less
than 15' when age<25 then '15-25' when age<35 then '25-35' else 'More
than 35' end
union all
select if(marital_status=0,'Unmarried','Married') as title, count(*)
as count, 'Marital Status' as type from (select distinct(user_id) from
 play_video  where video_id =49 ) play join user_info userinfo on (
play.user_id = userinfo.id) group by marital_status
```

19.6.2 Execution plans

19.6.2.1 Create an execution plan

An execution plan is a set of jobs that can be executed either at one time or periodically by means of scheduling. It can be executed on an existing E-MapReduce cluster and can also create a temporary cluster to execute the jobs dynamically. Its biggest advantage is that it only uses the resources it needs during execution.

Procedure

To create an execution plan, follow these steps:

1. Log on to the *Alibaba Cloud E-MapReduce console* .
2. Select a region.
3. In the upper-right corner, click Old MER Scheduling to go to the Jobs page.
4. In the navigation panel on the left, click Execution plan.
5. In the upper-right corner, click Create an execution plan.
6. In the Create an execution plan page, select between Create as needed and Existing clusters.

   a. Create as needed: Create a new cluster to run jobs.

      - Execution plan for one-time scheduling: Clusters with corresponding configurations are created when the execution starts and are then released upon completion of the operation. For more information about creation parameters, see *Create a cluster*.
      - Execution plan for periodic scheduling: A new cluster is created based on the scheduling settings you define and is then released upon completion of the operation.

   b. Existing clusters: Use an existing cluster that complies with the following requirement:

      - Execution plans can only be added to clusters that are Running or Idle.

      Select Existing clusters and then enter the Select Cluster page. Here, you can select a cluster to associate with the execution plan.
7. Click Next to enter the job configuration page. All user jobs are listed in the table on the left. You can select jobs for execution from this table. By clicking the right-

facing button, the checked jobs are added to the job queue. Jobs in the queue are then submitted to the cluster for execution in order. The same job can be added and executed several times. If you have not created any jobs, see *Jobs*.

8. Click Next to enter the scheduling mode configuration page. The configuration items are as follows:

    a. Name: Must be between 1-64 characters and may only consist of Chinese characters, English letters, numbers, hyphens (-), and underscores (_).

    b. Scheduling policy

        - Manual execution: The execution plan is not executed automatically after it is created. Instead, it must be executed manually. Once the execution is in progress, it cannot be executed again.
        - Periodic scheduling: If you select this function, it is enabled immediately after the execution plan is created. The execution then begins from the configured scheduling time. Periodic scheduling can be disabled in the list page. If a scheduling execution starts, but its last execution is not completed, the scheduling is ignored.

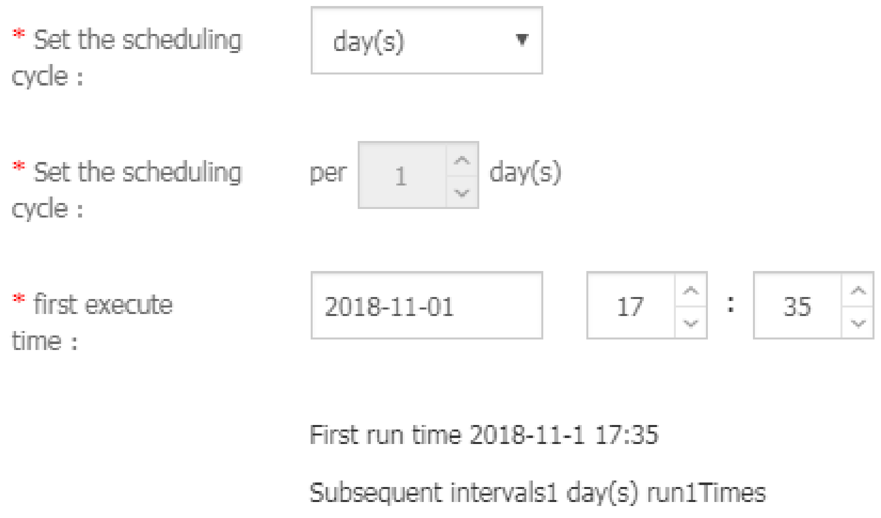
    c. Set the scheduling cycle: There are two scheduling periods: days and hours. The day cycle is one day by default and cannot be changed. However, you can set a specific time interval for hours. The range must be 1-23.

    d. First execution time: The effective start-time of the scheduling. From this point onwards, periodic scheduling is conducted according to the intervals specified.

9. Click OK to complete the creation of the execution plan.

Other information

- Example of periodic scheduling



* Set the scheduling cycle : day(s)

* Set the scheduling cycle : per 1 day(s)

* first execute time : 2018-11-01 17 : 35

First run time 2018-11-1 17:35

Subsequent intervals1 day(s) run1Times

These configurations indicate that the scheduling started on 11/01/2018 at 17:35 with an interval of one day. This means that the second scheduling was conducted on 11/02/2018, at 17:35.

- Sequence of jobs

  Jobs in the execution plan are executed from first to last according to the sequence that you defined in the job list.

- Sequence of multiple execution plans

  When multiple execution plans are submitted to the same cluster, each one submits jobs from its own job sequence. This means that jobs run parallel with each other.

- Example of early job debugging

  During the debugging of a job, it may take some time to create and start a cluster on demand. We recommend that you create a cluster manually first, select Associate the cluster in the execution plan to run jobs, and then set the scheduling mode to Execute immediately. During debugging, you can view the results by clicking Run now on the execution plan list page. Once the debugging is finished, modify the execution plan, modify the way you associate an existing cluster to create a new cluster on demand, and then modify the scheduling mode to periodic scheduling as required. Jobs are then executed automatically on demand.

19.6.2.2 Manage an execution plan

You can view, manage, and modify your execution plans as follows.

1. Log on to the *Alibaba Cloud E-MapReduce console* .

2. Select a region.
3. In the upper-right corner, click Old MER Scheduling to go to the Jobs page.
4. In the navigation panel on the left, click Execution plan.
5. Click Manage next to a plan to go to the execution plan detail page. Here, you can perform the following operations:

    - View details of the execution plan

      You can view the basic information of the execution plan, such as its name, associated clusters, job configurations, scheduling mode and status, and alarm information.
    - Modify the execution plan

      > Notice:
      > Jobs can only be modified if they are not currently running or being scheduled. For an execution plan to be executed immediately, it can only be modified when it is not currently running. If the execution plan is scheduled periodically, wait for the completion of its current operation and verify whether it is in periodical scheduling. If it is, click Stop scheduling before modifying it.

      Each separate module can be modified independently. Click the pen icon to modify information.
    - Configure alarm notifications

      There are three types of alarm notifications:

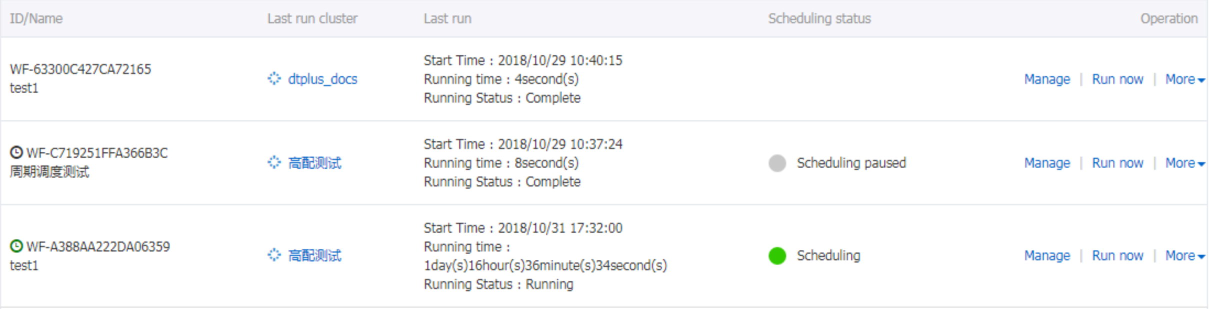
        - Booting timeout: If the periodical scheduling has not been conducted correctly at the specified time and is not executed within 10 minutes of timeout, an alarm is sent.
        - Failed execution: If any job in the execution plan fails, an alarm is sent.
        - Successful execution: If all jobs in the execution plan are executed successful ly, a notification is sent.
    - Run and view results

      If the execution plan can be run, in Basic Information, there will be a Run now button to the right of Scheduling status. If you click this button, a schedule will be executed.

At the bottom of the page, there are running records displaying the execution plan instances executed each time, making it easy to view the corresponding job list and logs.

19.6.2.3 Execution plan list

An execution plan list displays basic information about all of your execution plans, as shown in the following figure.



ID/Name	Last run cluster	Last run	Scheduling status	Operation
WF-63300C427CA72165 test1	dtplus_docs	Start Time : 2018/10/29 10:40:15 Running time : 4second(s) Running Status : Complete		Manage | Run now | More
WF-C719251FFA366B3C 周期调度测试	高配测试	Start Time : 2018/10/29 10:37:24 Running time : 8second(s) Running Status : Complete	● Scheduling paused	Manage | Run now | More
WF-A388AA222DA06359 test1	高配测试	Start Time : 2018/10/31 17:32:00 Running time : 1day(s)16hour(s)36minute(s)34second(s) Running Status : Running	● Scheduling	Manage | Run now | More

- ID/Name: The ID and name of the execution plan.
- Last run cluster: The last cluster to execute this execution plan. This can either be a cluster created on demand or an existing associated cluster. If a cluster is created automatically on demand, (Automatically created) is displayed beneath it, indicating that the cluster was created on demand by E-MapReduce and will be released automatically after running.
- Last run: The running status of the last execution plan.
  - Start time: The time at which the last execution plan started.
  - Running time: The duration for which the last plan ran.
  - Running status: The running status of the last execution plan.
- Scheduling status: This indicates whether scheduling is in progress or has been stopped. Only periodic jobs have a scheduling status.
- Operation
  - Manage: View and modify execution plans.
  - Run now: A job can only be run manually when it is neither running nor being scheduled. Click Run now to run the execution plan immediately.
  - More
    - Start/Stop scheduling: If the scheduling is stopped, Enable scheduling is displayed, which you can click to start the scheduling. If Stop scheduling is

displayed during scheduling, you can click it to stop the scheduling. This button is only available for periodic execution plans.

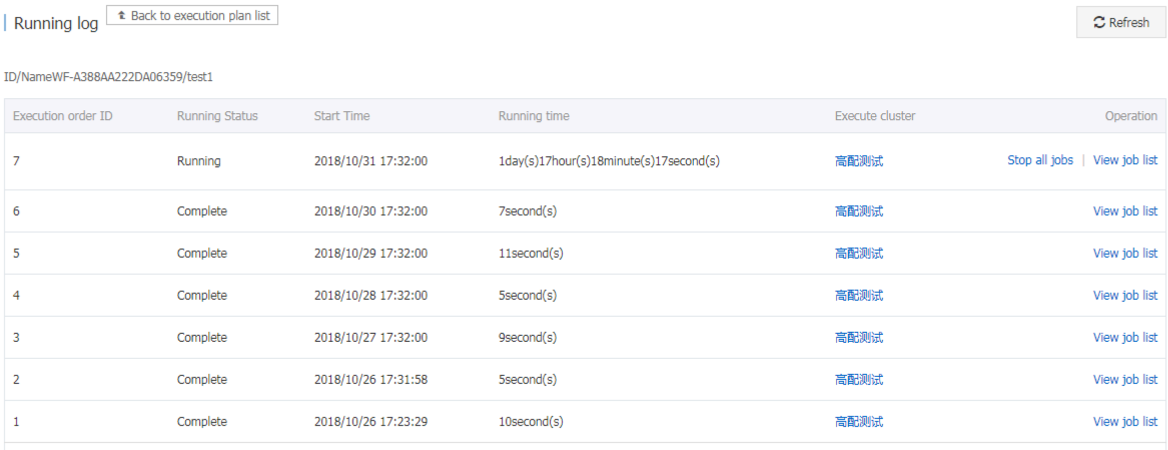
- Running log: Click to enter the job log viewing page.
- Delete: Deletes an execution plan. A running execution plan or one in the process of scheduling cannot be deleted.

19.6.2.4 View job results and logs

In this tutorial, you will learn how to view job results and logs.

View execution records

1. Log on to the *Alibaba Cloud E-MapReduce console* .
2. Select a region.
3. In the upper-right corner, click Old MER Scheduling to go to the Jobs page.
4. In the navigation panel on the left, click Execution plan.
5. To the right of the execution plan, click More > Running log.



ID/NameWF-A388AA222DA06359/test1

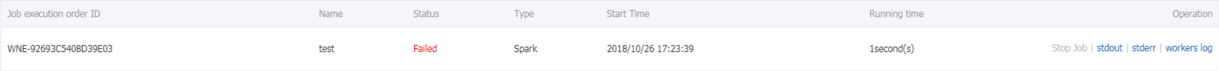
Execution order ID	Running Status	Start Time	Running time	Execute cluster	Operation
7	Running	2018/10/31 17:32:00	1day(s)17hour(s)18minute(s)17second(s)	高配测试	Stop all jobs | View job list
6	Complete	2018/10/30 17:32:00	7second(s)	高配测试	View job list
5	Complete	2018/10/29 17:32:00	11second(s)	高配测试	View job list
4	Complete	2018/10/28 17:32:00	5second(s)	高配测试	View job list
3	Complete	2018/10/27 17:32:00	9second(s)	高配测试	View job list
2	Complete	2018/10/26 17:31:58	5second(s)	高配测试	View job list
1	Complete	2018/10/26 17:23:29	10second(s)	高配测试	View job list

- Execution order ID: The sequence of execution for the execution record, which indicates its position in the execution queue. For example, 1 stands for the first position.
- Running status: The running status of each execution record.
- Start time The time at which the execution plan starts.
- Running time: The total running time until the page is viewed.
- Execute cluster: The cluster run by the execution plan can either be created on demand or it can be an existing associated cluster. Click to view the cluster details page.
- Operation

View job list: Click to enter the job list page.

View job records

On the Job list page, you can view the job list in the execution records of a single execution plan as well as the details of each job, as shown in the following figure.



Job execution order ID	Name	Status	Type	Start Time	Running time	Operation
WNE-92693C5408D39E03	test	Failed	Spark	2018/10/26 17:23:39	1second(s)	Stop Job | stdout | stderr | workers log

- Job execution order ID: After a job is executed, a corresponding ID is created, which is different from the job ID. The job execution ID is the unique identifier for viewing logs on OSS.
- Name: The name of the job.
- Status: The running status of the job.
- Type: The type of job.
- Start time: The time at which the job starts. This is converted into local time.
- Running time: The total running time of the job, in seconds.
- Operation
  - Stop job: You can stop a job if it is in the process of submission or running. If a job is in submission, stopping it will cancel execution. If the job is running, it will be killed.
  - stdout: Records all output content from the standard output (Channel 1) of the master process. If log saving is not enabled for the cluster where jobs are run, this function cannot be executed.
  - stderr: Records all output content from the diagnostic output (Channel 2) of the master process. If log saving is not enabled for the cluster where jobs are run, this function cannot be executed.
  - Workers log: Views the logs of all job worker nodes. If log saving is not enabled for the cluster where jobs are run, this function cannot be executed.

View job worker logs

- Cloud server instance IP: The ECS instance ID of a running job and the corresponding intranet IP address.
- Container ID: The container ID that YARN runs.
- Type: Different log types. stdout and stderr come from different outputs.
- Operation

View the log: Click different types to view the corresponding logs.

19.6.2.5 Parallel execution of multiple execution plans

To maximize the use of a cluster's available computing resources, multiple execution plans can be associated to the same cluster and executed in parallel.

The main points are summarized as follows:

- Jobs in the same execution plan are executed in sequence. By default, preceding jobs are executed before new jobs can be submitted and executed.
- If you have enough cluster resources, you can create multiple different execution plans and associate them to the same cluster to run and execute jobs in parallel. Clusters support a maximum of 20 execution plans by default.
- The management and control system currently supports the submission to YARN of multiple execution plans associated to the same cluster. However, if the cluster itself has insufficient resources, it may take some time for jobs in the YARN queue to wait for scheduling.

For more information on how to create execution plans and associate them to a cluster, see *Create an execution plan*.