ALIBABA CLOUD

Alibaba Cloud

DataWorks

Best Practices

Document Version: 20220419

Alibaba Cloud

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 because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should 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 this document based on the "status quo", "being defective", and "existing functions" of its products and services. 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 take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, 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 contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, 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 this document 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 directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice 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.
Warning	A warning notice 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 restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	**Notice:** If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	**Note:** You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.Data migration

1.1. Automatically synchronize IoT data to the cloud

The Internet of Things (IoT) is a network that carries data based on the Internet and traditional telecommunication networks. The IoT enables connections among all physical objects that are independently addressable. This topic describes how to automatically synchronize IoT data to MaxCompute on the cloud.

Context

The IoT aims to collect required information in real time by using various technologies and devices such as sensors. The IoT can connect objects on various types of networks to enable communications among the objects and interaction between the objects and human beings. On the IoT, objects and processes can be detected, identified, and managed in an intelligent manner.

As the three representative technologies of the third information and communications technology (ICT) wave, IoT, big data analytics, and cloud computing will have a wide influence in the future. IoT focuses on connecting things. Big data analytics aims to exploit the data value. Cloud computing provides service support such as computing resources for big data analytics and IoT.

Big data is an important part of the IoT system. The IoT system consists of devices, networks, platforms, applications, and features such as big data analytics and security assurance. Big data analytics is an important means to exploit the value of big data. To conduct big data analytics, synchronize data to the cloud first.

Solution

The solution for automatically synchronizing IoT data to the cloud includes storage and synchronization of raw data to a data analytics system.

Large amounts of raw data from IoT devices are generally stored in semi-structured form. For example, the raw data is stored in Comma-Separated Values (CSV) files in Object Storage Service (OSS).

To synchronize raw data to a big data analytics system or a traditional database, a professional data synchronization system is required. The following figure shows the process of synchronizing raw data from OSS to MaxCompute by using DataWorks Data Integration.

1. Create a batch sync node. For more information, see Configure a synchronization node by using the

codeless UI.

2. Configure the batch sync node to read data from OSS. For more information, see Configure OSS Reader.
3. Configure the batch sync node to write data to MaxCompute. You can also write data to a data store of another type. For more information, see Supported data stores and plug-ins.

Configure automatic data synchronization

To use DataWorks to read CSV files from OSS, you must specify the names of the files to read. IoT devices constantly generate raw data and store the raw data in CSV files. If you want to manually synchronize the raw data to the cloud, complex operations are required. This section describes how to configure automatic synchronization of data to MaxCompute.

When you use this solution, note the following points:

- CSV files must be periodically generated in OSS.

  DataWorks can periodically run a sync node as scheduled. You can configure the scheduling system to run the sync node at the interval at which CSV files are generated in OSS. For example, if a new CSV file is generated in OSS every 15 minutes, you can configure the scheduling system to run the sync node every 15 minutes.

- Each file obtained from OSS must be named based on the corresponding timestamp.

  After a sync node reads a file from OSS, DataWorks must name the file based on the timestamp of the file. You can specify a variable for DataWorks to dynamically generate file names to make sure that each file name is the same as the corresponding file name in OSS.

  > **Note** We recommend that you specify a variable for DataWorks to generate file names with a timestamp in the yyyymmddhhmm format, for example, iot_log_201911062315.csv.

1. Log on to the DataWorks console. In the left-side navigation pane, click Workspaces. On the Workspaces page, find the target workspace and click **Data Integration** in the Actions column.
2. Add connections to OSS and MaxCompute. For more information, see Add an OSS data source and Add a MaxCompute data source.
3. Click the DataWorks icon in the upper-left corner and choose **All Products > DataStudio**. On the page that appears, create a workflow. For more information, see Create a workflow.
4. Create a batch sync node. For more information, see Create a batch synchronization node.
5. On the configuration tab of the batch sync node, specify the Connection parameter and set Object Name Prefix to a file name format that contains a variable to indicate the timestamp.

   As shown in the preceding figure, specify the variable in the ${Variable} format in the Object Name Prefix field. You can customize the variable name, for example, filename.

   Click the **Properties** tab in the right-side navigation pane. In the Properties pane that appears, assign a value to the variable filename in the **Arguments** field in the **General** section, for example, `filename=$[yyyymmddhh24mi]`. For more information, see Configure scheduling parameters.

   The value $[yyyymmddhh24mi] indicates that the timestamp is accurate to minute. For example, 201911062315, 202005250843, and 201912012207 represent 23:15 on November 6, 2019, 08:43 on May 25, 2020, and 22:07 on December 1, 2019, respectively.

6. In the **Properties** pane, set **Instance Recurrence** in the **Schedule** section.

   Set **Instance Recurrence** to **Minute**. Then, set **Start From**, **Interval**, and **End At** as required.

> **Notice** The value of Interval must be the same as the interval at which files are generated in OSS. For example, if a new file is generated in OSS every 15 minutes, you must set Interval to 15 minutes.

7. Commit and deploy the batch sync node. For more information, see Deploy a node.
8. After you deploy the batch sync node, click **Operation Center** in the upper-right corner. On the page that appears, choose Cycle Task Maintenance > **Cycle Task** or **Cycle Task Maintenance** > **Cycle Instance** and check whether the generated node or instance meets your needs. For more information, see View auto triggered nodes.

1.2. Use Data Integration to import data to Elasticsearch

This topic describes how to use Data Integration to offline import data to Elasticsearch.

Prerequisites

1. An Alibaba Cloud account and its AccessKey pair are created.
2. MaxCompute is activated. A default MaxCompute connection is automatically created. The Alibaba Cloud account is used to log on to DataWorks.
3. A workspace is created so that you can collaboratively develop workflows and maintain data and nodes in the workspace. For more information, see Create a workspace.

> **Note** If you want to create a data integration node as a RAM user, grant the required permissions to the RAM user. For more information, see Prepare a RAM user and Manage workspace-level roles and members.

4. The required connection is created. For more information, see Configure a connection.

Create a batch sync node

1. Log on to the DataWorks console.
2. In the left-side navigation pane, click **Workspaces**.
3. Select the region where the required workspace resides. Find the required workspace and click **Data Integration**.
4. On the homepage of Data Integration, click **New offline synchronization** to go to the **DataStudio** page.
5. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

> **Note**
> - The node name must be 1 to 128 characters in length.
> - Set the **Location** parameter to the directory where the created workflow resides. For more information, see Create a workflow.

6. Click **Commit**.

Configure the batch sync node

1. After you create the batch sync node, click the **Switch to Code Editor** icon in the toolbar.
2. In the **Confirm** message, click **OK** to switch to the code editor.
3. Edit code in the code editor based on the following sample code:

```
{
"configuration": {
"setting": {
  "speed": {
    "concurrent": "1", // The number of concurrent threads.
    "mbps": "1" // The maximum transmission rate.
  }
},
"reader": {
  "parameter": {
    "connection": [
      {
        "table": [
          "`es_table`" // The name of the source table.
        ],
        "datasource": "px_mysql_OK" // The name of the connection. We recommend that yo
u use the name of the created connection.
      }
    ],
    "column": [ // The columns in the source table.
      "col_ip",
      "col_double",
      "col_long",
      "col_integer",
      "col_keyword",
      "col_text",
      "col_geo_point",
      "col_date"
    ],
    "where": "", // The WHERE clause.
  },
  "plugin": "mysql"
},
"writer": {
  "parameter": {
    "cleanup": true, // Specifies whether to clear the original data each time you impo
rt data to Elasticsearch. Set the parameter to true if you import all data or recreate
an index and to false if you import incremental data.
    "accessKey": "nimda", // If the X-PACK plug-in is used, enter the AccessKey secret.
Otherwise, enter a null string. The X-PACK plug-in is used for Alibaba Cloud Elasticsea
rch. Therefore, you must enter the AccessKey secret.
    "index": "datax_test", // The index of Elasticsearch. If no index is available, the
plug-in automatically creates one.
    "alias": "test-1-alias", // The alias to be added after data is imported.
    "settings": {
      "index": {
        "number_of_replicas": 0,
        "number_of_shards": 1
```

```
      }
    },
    "batchSize": 1000, // The number of data entries to write at a time.
    "accessId": "default", // If the X-PACK plug-in is used, enter the AccessKey ID. Ot
herwise, enter a null string. The X-PACK plug-in is used for Alibaba Cloud Elasticsearc
h. Therefore, you must enter the AccessKey ID.
    "endpoint": "http://xxx.xxxx.xxx:xxxx", // The endpoint that can be used to access
Elasticsearch. You can view the endpoint in the Elasticsearch console.
    "splitter": ",", // Specify a delimiter if you import arrays.
    "indexType": "default", // The type name in the index of Elasticsearch.
    "aliasMode": "append", // The mode in which an alias is added after the data is imp
orted. append: Add a new alias. exclusive: Retain only the new alias.
    "column": [ // The columns in Elasticsearch. The sequence of the columns is the sam
e as that of the columns in the reader.
      {
        "name": "col_ip",// This column corresponds to the name attribute column in Tab
lestore.
        "type": "ip"// The text type. The default analyzer is used.
      },
      {
        "name": "col_double",
        "type": "string"
      },
      {
        "name": "col_long",
        "type": "long"
      },
      {
        "name": "col_integer",
        "type": "integer"
      },
      {
        "name": "col_keyword",
        "type": "keyword"
      },
      {
        "name": "col_text",
        "type": "text"
      },
      {
        "name": "col_geo_point",
        "type": "geo_point"
      },
      {
        "name": "col_date",
        "type": "date"
      }
    ],
    "discovery": false// Specifies whether to enable automatic discovery. Set the param
eter to true.
  },
  "plugin": "elasticsearch"// The writer name. The name is ElasticsearchWriter. You do
not need to change the value.
}
```

```
},
"type": "job",
"version": "1.0"
}
```

4. Click the icon and the icon.

> **Note**
>
> - You can import data to Elasticsearch only in the code editor.
> - After you save the sync node, click the icon. The node is immediately run.
>
> You can also click the icon to commit the sync node to the scheduling system. The scheduling system then automatically runs the node from the next day based on scheduling parameters.

Subsequent steps

For more information about how to configure sync nodes that use other types of connections, see the topics in Reader configuration and Writer configuration.

1.3. Use Data Integration to ship data collected by LogHub to destinations

This topic describes how to use Data Integration to ship data collected by LogHub of Log Service to destinations that Data Integration supports, such as MaxCompute, Object Storage Service (OSS), Table Store, relational database management systems (RDBMSs), and DataHub. In this topic, MaxCompute is used as an example.

Context

You may need to synchronize data collected by LogHub with destinations in the following scenarios:

- Synchronize data between LogHub and connections such as MaxCompute across regions.
- Synchronize data between LogHub and connections such as MaxCompute across Alibaba Cloud accounts.
- Synchronize data between LogHub and connections such as MaxCompute under the same Alibaba Cloud account.
- Synchronize data between LogHub and connections such as MaxCompute across an Alibaba Cloud account and a Finance Cloud account.

If you have two Alibaba Cloud accounts A and B, you can use account B to create a sync node in Data Integration. Then, you can synchronize LogHub data under account A to MaxCompute under account B. The procedure is as follows:

1. Use the AccessKey ID and AccessKey secret of account A to create a LogHub connection.

   Account B has the permission to access all Log Service projects created by account A.

2. Use the AccessKey ID and AccessKey secret of Resource Access Management (RAM) user A1 under account A to create a LogHub connection.

- Use account A to grant RAM user A1 the `AliyunLogFullAccess` and `AliyunLogReadOnlyAccess` permissions on Log Service. For more information, see Create a RAM user and authorize the RAM user to access Log Service.
- Use account A to grant RAM user A1 custom permissions on Log Service.

  Use account A to log on to the RAM console. In the left-side navigation pane, choose **Permissions > Policies**. On the **Policies** page that appears, click **Create Policy**.

  For more information about authorization, see Authorization - Overview and Overview.

  If the following policy is applied to RAM user A1, account B can only synchronize data of project_name1 and project_name2 in Log Service through RAM user A1:

```
{
"Version": "1",
"Statement": [
{
"Action": [
"log:Get*",
"log:List*",
"log:CreateConsumerGroup",
"log:UpdateConsumerGroup",
"log:DeleteConsumerGroup",
"log:ListConsumerGroup",
"log:ConsumerGroupUpdateCheckPoint",
"log:ConsumerGroupHeartBeat",
"log:GetConsumerGroupCheckPoint"
],
"Resource": [
"acs:log:*:*:project/project_name1",
"acs:log:*:*:project/project_name1/*",
"acs:log:*:*:project/project_name2",
"acs:log:*:*:project/project_name2/*"
],
"Effect": "Allow"
}
]
}
```

Add a LogHub connection

1. Log on to the DataWorks console. In the left-side navigation pane, click Workspaces. On the Workspaces page, find the target workspace and click **Data Integration** in the Actions column.
2. On the **Data Integration** page, click **Connection** in the left-side navigation pane. The **Data Source** page appears.
3. On the **Data Source** page, click **Add Connection** in the upper-right corner.
4. In the **Add Connection** dialog box that appears, click **LogHub**.
5. In the **Add LogHub Connection** dialog box, set parameters for the LogHub connection.

Parameter	Description

Parameter	Description
Connection Name	The name of the connection. The name can contain letters, digits, and underscores (_) and must start with a letter.
Description	The description of the connection. The description can be up to 80 characters in length.
LogHub Endpoint	The endpoint of LogHub, in the format of `http://example.com` For more information, see Endpoints.
Project	The name of the Log Service project.
AccessKey ID	The AccessKey ID used as the logon credential. You can copy the AccessKey ID on the **Security Management** page.
AccessKey Secret	The AccessKey Secret used as the logon credential.

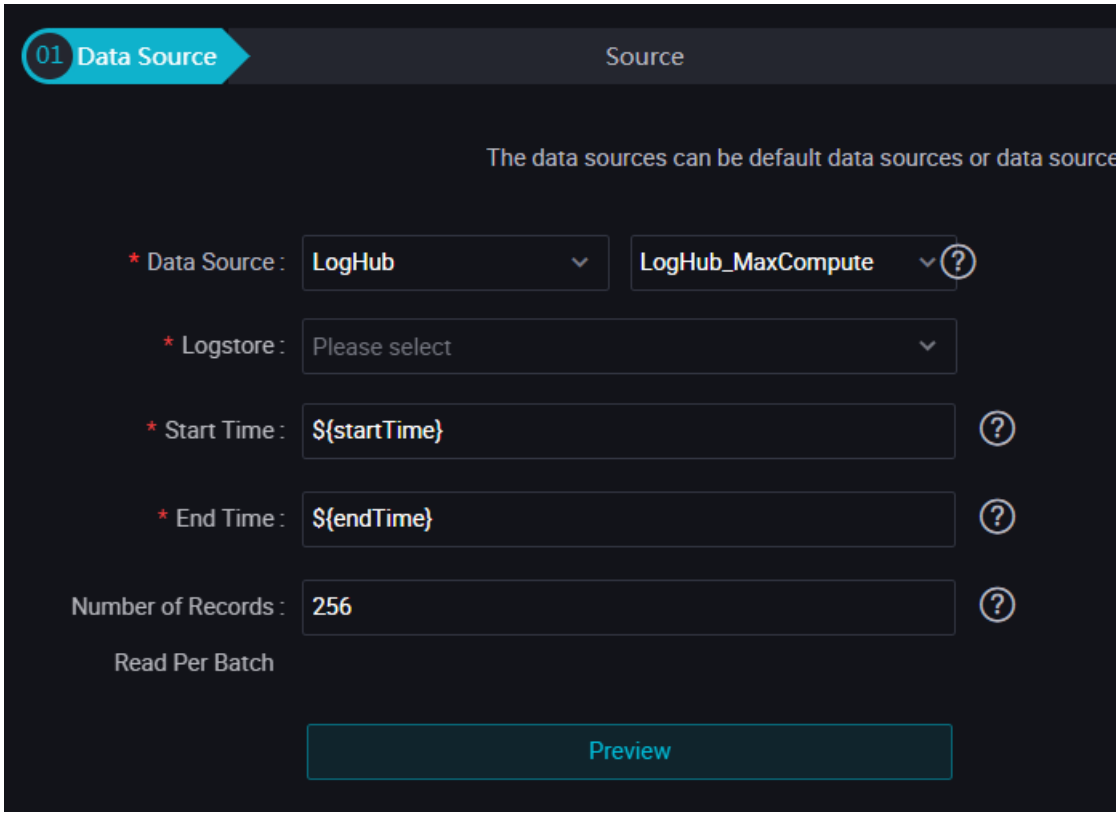
6. Click **Test Connection**.
7. After the connectivity test is passed, click **Complete**.

Create a batch sync node

1. On the **Data Source** page, click the DataWorks icon in the upper-left corner and choose **All Products > DataStudio**. The DataStudio page appears.
2. In the **Data Analytics** section, move the pointer over the icon and select **Workflow**.
3. In the **Create Workflow** dialog box that appears, set **Workflow Name** and **Description** and click **Create**.
4. Expand the created workflow in the left-side workflow list, right-click **Data Integration**, and then choose **Create > Batch Synchronization**.
5. In the **Create Node** dialog box that appears, set **Node Name** and **Location**.
6. Click **Commit**. The node editing tab appears.

Configure the batch sync node on the codeless user interface (UI)

1. In the Connections step, set parameters in the Source section.



Parameter	Description
Connection	The name of the connection. Select LogHub and enter the name of the LogHub connection.
Logstore	The name of the table from which incremental data is exported. You must enable the Stream feature for a table when creating the table, or by calling the UpdateTable operation after creating the table.
Start Timestamp	The start time of data consumption. This parameter defines the left boundary of an interval (left-closed and right-open) in the format of yyyyMMddHHmmss, for example, 20180111013000. The parameter can work with the scheduling time parameters in DataWorks.
End Timestamp	The end time of data consumption. This parameter defines the right boundary of an interval (left-closed and right-open) in the format of yyyyMMddHHmmss, for example, 20180111013010. The parameter can work with the scheduling time parameters in DataWorks.
Records per Batch	The number of data entries read at a time. The default value is 256.

Note You can click Preview to preview a small number of LogHub data entries in the preview box. Based on the start time and end time you specify for data synchronization, these data entries may be different from the actual data to be synchronized.

2. Select a MaxCompute connection and a destination table in the Target section.
3. In the Mappings step, configure the mapping between the fields in the source and destination tables.
4. In the **Channel** step, configure the maximum transmission rate and dirty data check rules.
5. Verify that the preceding configuration is correct and click the **Save** icon in the upper-left corner of the tab.
6. Run the batch sync node.

   You can run the batch sync node in one of the following ways:

   - One-time running

     Click the **Run** icon in the toolbar to run the node on the node editing tab.

     > **Note** After you click the Run icon, set the bizdate parameter in the dialog box that appears.

   - Scheduled running

     Click the **Submit** icon in the toolbar to submit the node to the scheduling system. The scheduling system then automatically runs the node from the next day based on the scheduling time parameters.

     In the Properties dialog box that appears, set Arguments to startTime=$[yyyymmddhh24miss-10/24/60] endTime=$[yyyymmddhh24miss-5/24/60] in the General section, as shown in the preceding figure. The value indicates that the start time of the node is 10 minutes before the scheduling time, and the end time is 5 minutes before the scheduling time.

     In the Schedule section, set Instance Recurrence to Minute, Start From to 00:00, Interval to 05, and End At to 23:59. Then the node is scheduled to run every 5 minutes from 00:00 to 23:59.

Configure the batch sync node in the code editor

1. After creating the batch sync node, click the **Switch to Code Editor** icon in the toolbar on the node editing tab.
2. In the **Confirm** dialog box that appears, click **OK** to switch to the code editor.
3. Click the **Apply Template** icon in the toolbar.
4. In the **Apply Template** dialog box that appears, set Source Connection Type to LogHub and Target Connection Type to ODPS, select the source and destination connections, and then click **OK** to apply a template.
5. Edit code as required in the code editor. The sample code is as follows:

```
{
"type": "job",
"version": "1.0",
"configuration": {
"reader": {
"plugin": "loghub",
"parameter": {
"datasource": "loghub_lzz",// The name of the source connection. Use the name of the co
nnection that you have added.
"logstore": "logstore-ut2",// The name of the source Logstore. A Logstore is a Log Serv
ice unit for collecting, storing, and querying log data.
"beginDateTime": "${startTime}",// The start time of data consumption. This parameter d
efines the left boundary of an interval (left-closed and right-open).
"endDateTime": "${endTime}",// The end time of data consumption. This parameter defines
the right boundary of an interval (left-closed and right-open).
"batchSize": 256,// The number of data entries read at a time. The default value is 256
.
"splitPk": "",
"column": [
"key1",
"key2",
"key3"
]
}
},
"writer": {
"plugin": "odps",
"parameter": {
"datasource": "odps_first",// The name of the destination connection. Use the name of t
he connection that you have added.
"table": "ok",// The name of the destination table.
"truncate": true,
"partition": "",// The partition information.
"column": [// The name of the destination column.
"key1",
"key2",
"key3"
]
}
},
"setting": {
"speed": {
"mbps": 8,// The maximum transmission rate.
"concurrent": 7// The maximum number of concurrent threads.
}
}
}
}
```

1.4. Use Data Integration to import data to DataHub

This topic describes how to use Data Integration to import offline data to DataHub. In this example, a sync node is configured to synchronize data from Stream to DataHub in the code editor.

Prerequisites

1. An Alibaba Cloud account and its AccessKey pair are created.
2. MaxCompute is activated. A default MaxCompute connection is automatically created. The Alibaba Cloud account is used to log on to DataWorks.
3. A workspace is created so that you can collaboratively develop workflows and maintain data and nodes in the workspace. For more information, see Create a workspace.

   > **Note** If you want to create a data integration node as a RAM user, grant the required permissions to the RAM user. For more information, see Prepare a RAM user and Manage workspace-level roles and members.

Context

Data Integration is a data synchronization platform that is provided by Alibaba Cloud. The platform is reliable, secure, cost-efficient, and scalable. It can be used across heterogeneous data storage systems and provides offline data access channels in diverse network environments for more than 20 types of connections.

In this example, a DataHub connection is configured. For information about how to use other types of connections to configure sync nodes, see Reader configuration and Writer configuration.

Procedure

1. Go to the **Data Integration** page.
   i. Log on to the DataWorks console.
   ii. In the left-side navigation pane, click **Workspaces**.
   iii. After you select the region where the required workspace resides, find the workspace and click **Data Integration**.
2. On the **Data Integration** homepage, click **New synchronization task** to go to the **DataStudio** page.
3. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters and click **Commit**.

   > **Note**
   > - The node name must be 1 to 128 characters in length.
   > - Set the **Location** parameter to the directory where the created workflow resides. For more information, see Create a workflow.

4. After the batch sync node is created, click the icon in the top toolbar.
5. In the **Confirm** message, click **OK**. The code editor appears.
6. Click the icon in the top toolbar.
7. In the **Apply Template** dialog box, set the Source Connection Type parameter to Stream and the Target Connection Type parameter to DataHub, and click **OK**.

8. After the template is applied, edit code as required.

```
{
"type": "job",
"version": "1.0",
"configuration": {
 "setting": {
   "errorLimit": {
     "record": "0"
   },
   "speed": {
     "mbps": "1",
     "concurrent": 1, // The number of concurrent threads.
     "throttle": false
   }
 },
 "reader": {
   "plugin": "stream",
   "parameter": {
     "column": [ // The columns in the source table.
       {
         "value": "field", // The column attribute.
         "type": "string"
       },
       {
         "value": true,
         "type": "bool"
       },
       {
         "value": "byte string",
         "type": "bytes"
       }
     ],
     "sliceRecordCount": "100000"
   }
 },
 "writer": {
   "plugin": "datahub",
   "parameter": {
     "datasource": "datahub", // The name of the connection.
     "topic": "xxxx", // The minimum unit for data subscription and publication in Data
Hub. You can use topics to distinguish different types of streaming data.
     "mode": "random", // Data is randomly written.
     "shardId": "0", // Shards are concurrent channels that are used for data transmiss
ion in a topic. Each shard has a unique ID.
     "maxCommitSize": 524288, // The amount of data, in MB, that DataHub Writer buffers
before DataHub Writer sends the data to the destination for the purpose of improving wr
iting efficiency. Default value: 1048576, in bytes, namely 1 MB.
     "maxRetryCount": 500
   }
 }
}
}
```

9. After the configuration is completed, click the and icons.

> Note
> - You can import data to DataHub only in the code editor.
> - If you want to change the template, click the icon in the top toolbar. The original content is overwritten after you apply the new template.
> - After you save the sync node, click the icon. The node is immediately run.
>
> You can also click the icon to commit the sync node to the scheduling system. The scheduling system runs the node at the scheduled time from the next day based on your configurations.

1.5. Migrate data across DataWorks workspaces

This topic describes how to migrate data across DataWorks workspaces in the same region.

Prerequisites

All the steps in the tutorial Build an online operation analysis platform are completed. For more information, see Business scenarios and development process.

Context

This topic uses the bigdata_DOC workspace created in the tutorial Build an online operation analysis platform as the source workspace. You need to create a destination workspace to store the tables, resources, configurations, and data synchronized from the source workspace.

Procedure

1. Create a destination workspace.
   i. Log on to the DataWorks console. In the left-side navigation pane, click **Workspaces**.
   ii. On the Workspaces page that appears, select the **China (Hangzhou)** region in the upper-left corner and click **Create Workspace**.
   iii. In the **Create Workspace** pane that appears, set parameters in the **Basic Settings** step and click **Next**.

Section	Parameter	Description
	Workspace Name	The name of the workspace. The name must be 3 to 23 characters in length and can contain letters, underscores (_), and digits. The name must start with a letter.

Section	Parameter	Description
Basic Information	Display Name	The display name of the workspace. The display name can be a maximum of 23 characters in length. It can contain letters, underscores (_), and digits and must start with a letter.
	Mode	The mode of the workspace. Valid values: **Basic Mode (Production Environment Only)** and **Standard Mode (Development and Production Environments)**. ■ **Basic Mode (Production Environment Only)**: A workspace in basic mode is associated with only one MaxCompute project. Workspaces in basic mode do not isolate the development environment from the production environment. In these workspaces, you can perform only basic data development and cannot strictly control the data development process and the permissions on tables. ■ **Standard Mode (Development and Production Environments)**: A workspace in standard mode is associated with two MaxCompute projects. One serves as the development environment, and the other serves as the production environment. Workspaces in standard mode allow you to develop code in a standard way and strictly control the permissions on tables. These workspaces impose limits on table operations in the production environment for data security. For more information, see Basic mode and standard mode.
	Description	The description of the workspace.
Advanced Settings	Download SELECT Query Result	Specifies whether the query results that are returned by SELECT statements in DataStudio can be downloaded. If you turn off this switch, the query results cannot be downloaded. You can change the setting of this parameter for the workspace in the Workspace Settings panel after the workspace is created. For more information, see Configure security settings.

The source workspace bigdata_DOC is in the basic mode. For convenience, set Mode to **Basic Mode (Production Environment Only)** in the Basic Settings step when you create a destination workspace.

Set Workspace Name to a globally unique name. We recommend that you use a name that is easy to distinguish. In this example, set Workspace Name to clone_test_doc.

iv. In the Select Engines and Services step, select the MaxCompute check box and Pay-As-You-Go in the **Compute Engines** section and click **Next**.

v. In the **Engine Details** step, set the required parameters and click **Create Workspace**.

Compute engine	Parameter	Description
MaxCompute	**Instance Display Name**	The display name of the compute engine instance. The display name must be 3 to 27 characters in length, and can contain only letters, underscores (_), and digits. It must start with a letter.
	MaxCompute Project Name	The name of the MaxCompute project. By default, the name is the same as that of the DataWorks workspace.
	Account for Accessing MaxCompute	The identity used to access the MaxCompute project. For the development environment, the value is fixed to **Task owner**. For the production environment, the valid values are **Alibaba Cloud primary account** and **Alibaba Cloud sub-account**.
	Resource Group	The quotas of computing resources and disk spaces for the compute engine instance.

2. Clone node configurations and resources across workspaces.

You can use the **cross-workspace cloning** feature of DataWorks to clone the node configurations and resources from the bigdata_DOC workspace to the clone_test_doc workspace. For more information, see Clone nodes across workspaces.

> **Note**
> - The cross-workspace cloning feature cannot clone table schemas or data.
> - The cross-workspace cloning feature cannot clone combined nodes. If the destination workspace needs to use the combined nodes that exist in the source workspace, you need to manually create the combined nodes in the destination workspace.

i. Go to the bigdata_DOC workspace and click **Cross-project cloning** in the upper-right corner. The Create Clone Task page appears.

ii. Set **Target Workspace** to clone_test_doc and **Workflow** to Workshop that needs to be cloned. Select all the nodes in the workflow and click **Add to List**. Click **To-Be-Cloned Node List** in the upper-right corner.

iii. In the Nodes to Clone pane that appears, click **Clone All**. The selected nodes are cloned to the clone_test_doc workspace.

iv. Go to the destination workspace and check whether the nodes are cloned.

3. Create tables.

   The cross-workspace cloning feature cannot clone table schemas. Therefore, you need to manually create required tables in the destination workspace.

   - For non-partitioned tables, we recommend that you use the following SQL statement to synchronize the table schema from the source workspace:

     ```
     create table table_name as select * from Source workspace. Table name;
     ```

   - For partitioned tables, we recommend that you use the following SQL statement to synchronize the table schema from the source workspace:

     ```
     create table table_name partitioned by (Partition key column string);
     ```

   Commit the tables to the production environment. For more information, see Create tables.

4. Synchronize data.

   The cross-workspace cloning feature cannot clone data from the source workspace to the destination workspace. You need to manually synchronize required data to the destination workspace. To synchronize the data of the rpt_user_trace_log table from the source workspace to the destination workspace, follow these steps:

   i. Create a connection.

      a. Go to the **Data Integration** page and click **Connection** in the left-side navigation pane.

      b. On the **Data Source** page that appears, click **Add a Connection** in the upper-right corner. In the Add Connection dialog box that appears, select **MaxCompute(ODPS)** in the Big Data Storage section.

      c. In the Add MaxCompute(ODPS) Connection dialog box that appears, set **Connection Name**, **MaxCompute Project Name**, **AccessKey ID**, and **AccessKey Secret**, and click **Complete**. For more information, see Add a MaxCompute data source.

   ii. Create a batch sync node.

      a. Go to the **DataStudio** page, click the Data Analytics tab, and then click Workshop under **Business Flow**. Right-click **Data Integration** and choose **Create > Batch Synchronization** to create a batch sync node.

      b. On the configuration tab of the batch sync node, set the required parameters. In this example, set Connection under Source to bigdata_DOC and that under Target to **odps_first**. Set Table to rpt_user_trace_log. After the configuration is completed, click the **Properties** tab in the right-side navigation pane.

      c. Click **Use Root Node** in the Dependencies section and commit the batch sync node.

iii. Generate retroactive data for the batch sync node.

a. On the DataStudio page, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**.

b. On the page that appears, choose **Cycle Task Maintenance > Cycle Task** in the left-side navigation pane.

c. On the page that appears, find the batch sync node you created in the node list and click the node name. On the canvas that appears on the right, right-click the batch sync node and choose **Run > Current Node Retroactively**.

d. In the Patch Data dialog box that appears, set the required parameters. In this example, set Data Timestamp to Jun 11, 2019 - Jun 17, 2019 to synchronize data from multiple partitions. Click **OK**.

e. On the **Patch Data** page that appears, check the running status of the retroactive instances that are generated. If **Successful** appears in the **STATUS** column of a retroactive instance, the instance is run and the corresponding data is synchronized.

iv. Verify the data synchronization.

On the Data Analytics tab of the DataStudio page, right-click the Workshop workflow under Business Flow and choose **Create > MaxCompute > ODPS SQL** to create an ODPS SQL node. On the configuration tab of the ODPS SQL node, run the following SQL statement to check whether data is synchronized to the destination workspace:

```
select * from rpt_user_trace_log where dt BETWEEN '20190611' and '20190617';
```

1.6. Migrate data from a user-created MySQL database on an ECS instance to MaxCompute

This topic describes how to use an exclusive resource group for Data Integration to migrate data from a user-created MySQL database on an Elastic Compute Service (ECS) instance to MaxCompute.

Prerequisites

- An ECS instance is purchased and bound to a virtual private cloud (VPC) but not the classic network. A MySQL database that stores test data is deployed on the ECS instance. An account used to connect to the database is created. In this example, use the following statements to create a table in the MySQL database and insert test data to the table:

```
CREATE TABLE IF NOT EXISTS good_sale(
   create_time timestamp,
   category varchar(20),
   brand varchar(20),
   buyer_id varchar(20),
   trans_num varchar(20),
   trans_amount DOUBLE,
   click_cnt varchar(20)
   );
insert into good_sale values('2018-08-21','coat','brandA','lilei',3,500.6,7),
('2018-08-22','food','brandB','lilei',1,303,8),
('2018-08-22','coat','brandC','hanmeimei',2,510,2),
('2018-08-22','bath','brandA','hanmeimei',1,442.5,1),
('2018-08-22','food','brandD','hanmeimei',2,234,3),
('2018-08-23','coat','brandB','jimmy',9,2000,7),
('2018-08-23','food','brandA','jimmy',5,45.1,5),
('2018-08-23','coat','brandE','jimmy',5,100.2,4),
('2018-08-24','food','brandG','peiqi',10,5560,7),
('2018-08-24','bath','brandF','peiqi',1,445.6,2),
('2018-08-24','coat','brandA','ray',3,777,3),
('2018-08-24','bath','brandG','ray',3,122,3),
('2018-08-24','coat','brandC','ray',1,62,7) ;
```

- The private IP address, VPC, and vSwitch of your ECS instance are noted.
- A security group rule is added for the ECS instance to allow access requests on the port used by the MySQL database. By default, the MySQL database uses port 3306. For more information, see 添加安全组规则. The name of the security group is noted.
- A DataWorks workspace is created. In this example, create a DataWorks workspace that is in basic mode and uses a MaxCompute compute engine. Make sure that the created DataWorks workspace belongs to the same region as the ECS instance. For more information about how to create a workspace, see Create a workspace.
- An exclusive resource group for Data Integration is purchased and bound to the VPC where the ECS instance resides. The exclusive resource group and the ECS instance are in the same zone. For more information, see Create and use an exclusive resource group for Data Integration. After the exclusive resource group is bound to the VPC, you can view information about the exclusive resource group on the **Resource Groups** page.
- Check whether the VPC, vSwitch, and security group of the exclusive resource group are the same as those of the ECS instance.

Context

An exclusive resource group can transmit your data in a fast and stable manner. Make sure that the exclusive resource group for Data Integration belongs to the same zone in the same region as the data store that needs to be accessed. Make sure that the exclusive resource group for Data Integration belongs to the same region as the DataWorks workspace. In this example, the data store that needs to be accessed is a user-created MySQL database on an ECS instance.

Procedure

1. Create a connection to the MySQL database in the DataWorks console.
   i. Log on to the DataWorks console by using your Alibaba Cloud account.

ii. On the **Workspaces** page, find the required workspace and click **Data Integration**.

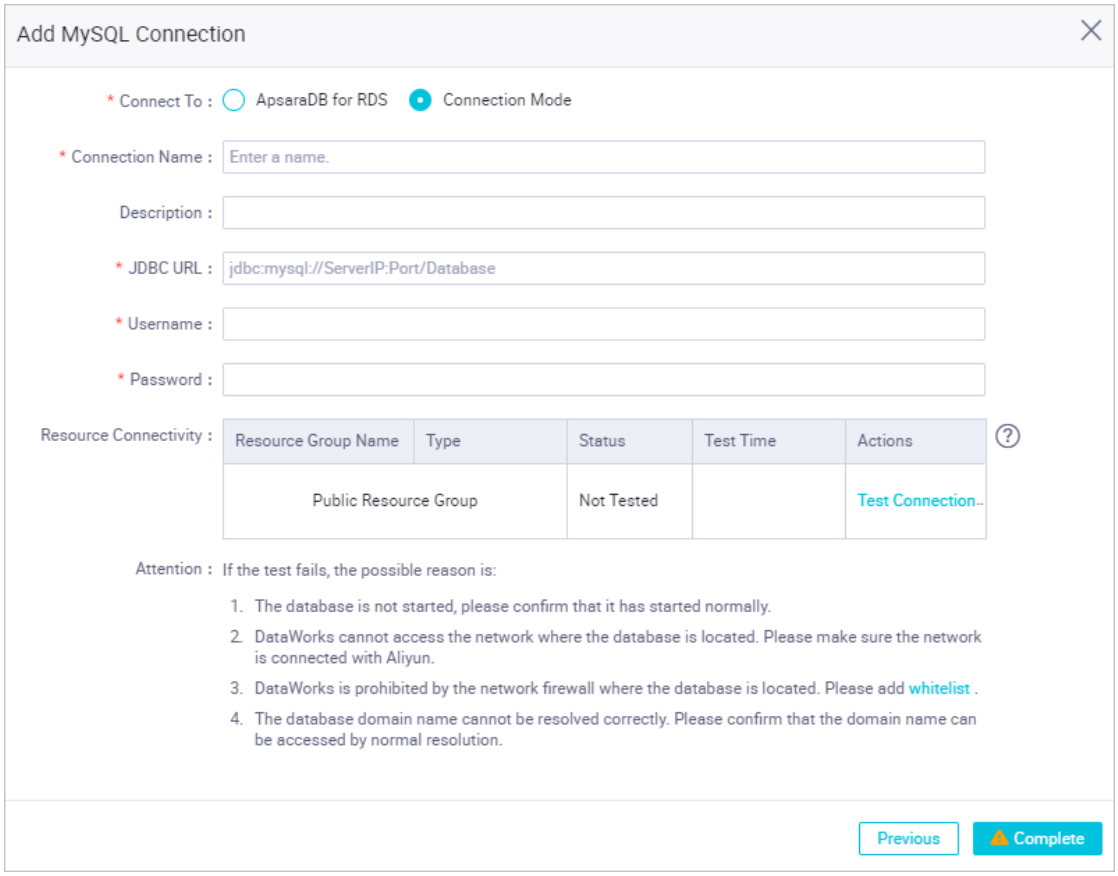
iii. In the left-side navigation pane, click **Connection**.

iv. On the **Data Source** page, click **New data source** in the upper-right corner.

v. In the **Add data source** dialog box, select **MySQL**.

vi. In the **Add MySQL data source** dialog box, set the parameters. For more information, see Add a MySQL data source.

For example, set the Data source type parameter to **Connection string mode**. Use the private IP address of the ECS instance and the default port number 3306 of the MySQL database when you specify the Java Database Connectivity (JDBC) URL.



> **Note** DataWorks cannot test the connectivity of a user-created MySQL database in a VPC. Therefore, it is normal that a connectivity test fails.

vii. Find the required resource group and click **Test connectivity**.

During data synchronization, a sync node uses only one resource group. You must test the connectivity of all the resource groups for Data Integration on which your sync nodes will be run and make sure that the resource groups can connect to the data store. This ensures that your sync nodes can be run as expected. For more information, see Select a network connectivity solution.

viii. After the connection passes the connectivity test, click **Complete**.

2. Create a MaxCompute table.

You must create a table in DataWorks to receive test data from the MySQL database.

i. Click the icon in the upper-left corner and choose **All Products > DataStudio**.

ii. Create a workflow. For more information, see Create a workflow.

iii. Right-click the created workflow and choose **Create > MaxCompute > Table**.

iv. Enter a name for your MaxCompute table. In this example, set the Table Name parameter to good_sale, which is the same as the name of the table in the MySQL database. Click **DDL Statement**, enter the table creation statement, and then click **Generate Table Schema**.

In this example, enter the following table creation statement. Pay attention to data type conversion.

```
CREATE TABLE IF NOT EXISTS good_sale(
   create_time string,
   category STRING,
   brand STRING,
   buyer_id STRING,
   trans_num BIGINT,
   trans_amount DOUBLE,
   click_cnt BIGINT
   );
```

v. Set the **Display Name** parameter and click **Commit to Production Environment**. The MaxCompute table named good_sale is created.

3. Configure a data integration node.

i. Right-click the workflow you just created and choose **Create > Data Integration > Batch Synchronization** to create a data integration node.

ii. Set the Connection parameter under Source to the created MySQL connection and the Connection parameter under Target to odps_first. Click the **Switch to Code Editor** icon to switch to the code editor.

If you cannot set the **Table** parameter under Source or an error is returned when you attempt to switch to the code editor, ignore the issue.

iii. Click the **Resource Group configuration** tab in the right-side navigation pane and select an exclusive resource group that you have purchased.

If you do not select the exclusive resource group as the resource group for Data Integration of your node, the node may fail to be run.

iv. Enter the following code for the data integration node:

```
{
    "type": "job",
    "steps": [
        {
            "stepType": "mysql",
            "parameter": {
                "column": [// The columns in the source table.
                    "create_time",
                    "category",
                    "brand",
                    "buyer_id",
                    "trans_num",
                    "trans_amount",
                    "click_cnt"
```

```
                ],
                "connection": [
                    {
                        "datasource": "shuai",// The source connection.
                        "table": [
                            "good_sale"// The name of the table in the source datab
ase. The name must be enclosed in brackets [].
                        ]
                    }
                ],
                "where": "",
                "splitPk": "",
                "encoding": "UTF-8"
            },
            "name": "Reader",
            "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
                "partition": "",
                "truncate": true,
                "datasource": "odps_first",// The destination connection.
                "column": [// The columns in the destination table.
                    "create_time",
                    "category",
                    "brand",
                    "buyer_id",
                    "trans_num",
                    "trans_amount",
                    "click_cnt"
                ],
                "emptyAsNull": false,
                "table": "good_sale"// The name of the destination table.
            },
            "name": "Writer",
            "category": "writer"
        }
    ],
    "version": "2.0",
    "order": {
        "hops": [
            {
                "from": "Reader",
                "to": "Writer"
            }
        ]
    },
    "setting": {
        "errorLimit": {
            "record": "0"
        },
        "speed": {
            "throttle": false,
```

```
            "concurrent": 2
        }
    }
}
```

v. Click the Run icon. You can view the **Runtime Log** tab in the lower part of the page to check whether the test data is synchronized to MaxCompute.

Result

To query data in the MaxCompute table, create an **ODPS SQL** node.Enter the statement `select * from good_sale ;` , and click the **Run** icon. If the test data appears, it is synchronized to the MaxCompute table.

1.7. Best practice to migrate data from Oracle to MaxCompute

This topic describes how to use the data integration feature of DataWorks to migrate data from Oracle to MaxCompute.

Prerequisites

- The DataWorks environment is ready.
  i. Activate MaxCompute and DataWorks.
  ii. Create a workspace. In this example, a workspace in basic mode is used.
  iii. A workflow is created in the DataWorks console. For more information, see Create a workflow.
- The Oracle database is ready.

  In this example, the Oracle database is installed on an Elastic Compute Service (ECS) instance. To enable network communication, you must configure a public IP address for the ECS instance. In addition, you must configure a security group rule for the ECS instance to ensure that the common port 1521 of the Oracle database is accessible. For more information about how to configure a security group rule for an ECS instance, see Modify security group rules.

  In this example, the type of the ECS instance is **ecs.c5.xlarge**. The ECS instance resides in a virtual private cloud (VPC) in the China (Hangzhou) region.

Context

In this example, DataWorks Oracle Reader is used to read test data from the Oracle database. For more information, see Oracle Reader.

Prepare test data in the Oracle database

1. In the Oracle database, create the DTSTEST.GOOD_SALE table that contains the CREATE_TIME, CATEGORY, BRAND, BUYER_ID, TRANAS_NUM, TRANS_AMOUNT, and CLICK_CNT columns.
2. Insert test data to the DTSTEST.GOOD_SALE table. In this example, the following statements are executed to insert test data:

```
insert into good_sale values('28-December-19','Kitchenware','Brand A','hanmeimei','6','
80.6','4');
insert into good_sale values('21-December-19','Fresh food','Brand B','lilei','7','440.6
','5');
insert into good_sale values('29-December-19','Clothing','Brand C','lily','12','351.9',
'9');
commit;
```

3. After data insertion, execute the following statement to view the data in the table:

```
select * from good_sale;
```

Use DataWorks to migrate data from the Oracle database to MaxCompute

1. Go to the **DataStudio** page.
   i. Log on to the DataWorks console.
   ii. In the left-side navigation pane, click **Workspaces**.
   iii. Select the region where the required workspace resides. Find the required workspace and click **Data Analytics**.
2. On the **DataStudio** page, create a destination table to receive data migrated from the Oracle database.
   i.
   ii.
   iii.
   iv. In the **DDL Statement** dialog box, enter the following statement and click **Generate Table Schema**:

```
CREATE TABLE good_sale
(
    create_time     string,
    category        string,
    brand           string,
    buyer_id        string,
    trans_num       bigint,
    trans_amount    double,
    click_cnt       bigint
) ;
```

   When you create the MaxCompute table, make sure that the data types of the MaxCompute table match those of the Oracle table. For more information about the data types supported by Oracle Reader, see Data types.

   v.
3. Create an Oracle connection. For more information, see Add an Oracle data source.
4. Create a batch sync node.
   i.
   ii.

iii. After you create the batch sync node, set the **Connection** parameter to the created Oracle connection and the **Table** parameter to the Oracle table that you have created. Click **Map Fields with the Same Name**. Use the default values for other parameters.

iv.

v.

Verify the result

1.
2.
3. On the configuration tab of the ODPS SQL node, enter the following statement:

```
-- Check whether the data is written to MaxCompute.
select * from good_sale;
```

4.
5.

1.8. Migrate data from Kafka to MaxCompute

This topic describes how to use DataWorks Data Integration to migrate data from a Kafka cluster to MaxCompute.

Prerequisites

- MaxCompute is activated. For more information, see Activate MaxCompute and DataWorks.
- A workflow is created in DataWorks. In this example, a DataWorks workspace in basic mode is used. For more information, see Create a workflow.
- A Kafka cluster is created.

  Before data migration, make sure that your Kafka cluster works as expected. In this example, Alibaba Cloud E-MapReduce (EMR) is used to automatically create a Kafka cluster. For more information, see Kafka quick start.

  In this example, the following version of EMR Kafka is used:

  - EMR version: V3.12.1
  - Cluster type: Kafka
  - Software: Ganglia 3.7.2, ZooKeeper 3.4.12, Kafka 2.11-1.0.1, and Kafka Manager 1.3.3.16

  The Kafka cluster is deployed in a virtual private cloud (VPC) in the China (Hangzhou) region. The Elastic Compute Service (ECS) instances in the primary instance group of the Kafka cluster are configured with public and private IP addresses.

Context

Kafka is distributed middleware that is used to publish and subscribe to messages. Kafka is widely used because of its high performance and high throughput. Kafka can process millions of messages per second. Kafka is applicable to streaming data processing, and is used in scenarios such as user behavior tracing and log collection.

A typical Kafka cluster contains several producers, brokers, consumers, and a ZooKeeper cluster. A Kafka cluster uses ZooKeeper to manage configurations and coordinate services in the cluster.

A topic is the most commonly used collection of messages in a Kafka cluster, and is a logical concept for message storage. Topics are not stored on physical disks. Instead, messages in each topic are stored on the disks of each cluster node by partition. Multiple producers can publish messages to a topic, and multiple consumers can subscribe to messages in a topic.

When a message is stored to a partition, the message is allocated an offset. The offset is the unique ID of the message in the partition. The offsets of messages in each partition start from 0.

Step 1: Prepare Kafka data

You must prepare test data in the Kafka cluster. Configure a security group rule for the header node of the EMR cluster to allow requests on TCP ports 22 and 9092. This way, you can log on to the header node of the EMR cluster and MaxCompute and DataWorks can communicate with the header node.

1. Log on to the header node of the EMR cluster.
   i. Log on to the EMR console.
   ii. In the top navigation bar, click **Cluster Management**.
   iii. On the page that appears, find the cluster for which you want to prepare test data and go to the details page of the cluster.
   iv. On the details page of the cluster, click Instances. Find the IP address of the header node of the E-MapReduce cluster and use the IP address to remotely log on to the header node by using Secure Shell (SSH).
2. Create a test topic.

   Run the following command to create a test topic named testkafka:

   ```
   [root@emr-header-1 ~]# kafka-topics.sh --zookeeper emr-header-1:2181/kafka-1.0.1 --part
   itions 10 --replication-factor 3 --topic testkafka  --create
   Created topic "testkafka".
   ```

3. Write test data.

   Run the following command to simulate a producer to write data to the testkafka topic. Kafka is used to process streaming data. You can continuously write data to the topic. To ensure that test results are valid, we recommend that you write more than 10 records.

   ```
   [root@emr-header-1 ~]# kafka-console-producer.sh --broker-list emr-header-1:9092 --topi
   c testkafka
   >123
   >abc
   >
   ```

   To simulate a consumer to check whether data is written to Kafka, open another SSH window and run the following command. If the data that is written appears, the data is written to the topic.

   ```
   [root@emr-header-1 ~]# kafka-console-consumer.sh --bootstrap-server emr-header-1:9092 -
   -topic testkafka --from-beginning
   123
   abc
   ```

Step 2: Create a destination table in DataWorks

Create a destination table in DataWorks to receive data from Kafka.

1.
2.
3.
4. Click **DDL Statement**. In the **DDL Statement** dialog box, enter the following CREATE TABLE statement and click **Generate Table Schema**:

```
CREATE TABLE testkafka
(
 key             string,
 value           string,
 partition1      string,
 timestamp1      string,
 offset          string,
 t123            string,
 event_id        string,
 tag             string
) ;
```

   Each column in the statement corresponds to a default column of Kafka Reader that is provided by DataWorks Data Integration.

   - __key__: the key of the message.
   - __value__: the complete content of the message.
   - __partition__: the partition where the message resides.
   - __headers__: the header of the message.
   - __offset__: the offset of the message.
   - __timestamp__: the timestamp of the message.

   You can customize a column. For more information, see Kafka Reader.

5.

Step 3: Synchronize the data

1. Create an exclusive resource group for Data Integration.

   The Kafka plug-in cannot run on the default resource group of DataWorks as expected. You must use an exclusive resource group for Data Integration to synchronize data. For more information, see Create and use an exclusive resource group for Data Integration.

2.
3.
4.
5. Configure the script. In this example, enter the following code:

```
{
    "type": "job",
    "steps": [
        {
            "stepType": "kafka",
            "parameter": {
```

```
                "server": "47.xxx.xxx.xxx:9092",
                "kafkaConfig": {
                    "group.id": "console-consumer-83505"
                },
                "valueType": "ByteArray",
                "column": [
                    "__key__",
                    "__value__",
                    "__partition__",
                    "__timestamp__",
                    "__offset__",
                    "'123'",
                    "event_id",
                    "tag.desc"
                ],
                "topic": "testkafka",
                "keyType": "ByteArray",
                "waitTime": "10",
                "beginOffset": "0",
                "endOffset": "3"
            },
            "name": "Reader",
            "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
                "partition": "",
                "truncate": true,
                "compress": false,
                "datasource": "odps_first",
                "column": [
                    "key",
                    "value",
                    "partition1",
                    "timestamp1",
                    "offset",
                    "t123",
                    "event_id",
                    "tag"
                ],
                "emptyAsNull": false,
                "table": "testkafka"
            },
            "name": "Writer",
            "category": "writer"
        }
    ],
    "version": "2.0",
    "order": {
        "hops": [
            {
                "from": "Reader",
                "to": "Writer"
```

```
            }
        ]
    },
    "setting": {
        "errorLimit": {
            "record": ""
        },
        "speed": {
            "throttle": false,
            "concurrent": 1,
        }
    }
}
```

To view the values of the group.id parameter and the names of consumer groups, run the **kafka-consumer-groups.sh --bootstrap-server emr-header-1:9092 --list** command on the header node.

```
[root@emr-header-1 ~]#  kafka-consumer-groups.sh  --bootstrap-server emr-header-1:9092
--list
Note: This will not show information about old Zookeeper-based consumers.
_emr-client-metrics-handler-group
console-consumer-69493
console-consumer-83505
console-consumer-21030
console-consumer-45322
console-consumer-14773
```

In this example, console-consumer-83505 is used. Run the **kafka-consumer-groups.sh --bootstrap-server emr-header-1:9092 --describe --group console-consumer-83505** command on the header node to obtain the values of the beginOffset and endOffset parameters.

```
[root@emr-header-1 ~]# kafka-consumer-groups.sh --bootstrap-server emr-header-1:9092 --
describe --group console-consumer-83505
Note: This will not show information about old Zookeeper-based consumers.
Consumer group 'console-consumer-83505' has no active members.
TOPIC                          PARTITION  CURRENT-OFFSET  LOG-END-OFFSET  LAG        CO
NSUMER-ID                                       HOST                           CLIENT-I
D
testkafka                      6          0               0               0          -
-                              -
test                           6          3               3               0          -
-                              -
testkafka                      0          0               0               0          -
-                              -
testkafka                      1          1               1               0          -
-                              -
testkafka                      5          0               0               0          -
-                              -
```

6. Configure a resource group for scheduling.
   i. On the node configuration tab, click the **Properties** tab in the right-side navigation pane.

ii. In the **Resource Group** section, set the **Resource Group** parameter to the exclusive resource group for Data Integration that you have created.

> **Note** Assume that you want to write Kafka data to MaxCompute at a regular interval, for example, on an hourly basis. You can use the beginDateTime and endDateTime parameters to set the interval for data reading to 1 hour. Then, the data integration node is scheduled to run once per hour. For more information, see Kafka Reader.

7.

8.

What's next

You can create a data development job and run SQL statements to check whether the data has been synchronized from Message Queue for Apache Kafka to the current table. This topic uses the `select * from testkafka` statement as an example. Specific steps are as follows:

1. In the left-side navigation pane, choose **Data Development > Business Flow**.
2. Right-click and choose **Data Development > Create Data Development Node ID > ODPS SQL**.
3. In the **Create Node** dialog box, enter the node name, and then click **Submit**.
4. On the page of the created node, enter `select * from testkafka` and then click the **Run** icon.

1.9. Migrate JSON data from OSS to MaxCompute

This topic describes how to use the data integration feature of DataWorks to migrate JSON data from Object Storage Service (OSS) to MaxCompute and use the GET_JSON_OBJECT function to extract JSON objects.

Prerequisites

- MaxCompute is activated.
- DataWorks is activated.
- A workflow is created in the DataWorks console. In this example, a workflow is created in a DataWorks workspace in basic mode. For more information, see Create a workflow.
- A TXT file that contains JSON data is uploaded to an OSS bucket. In this example, the OSS bucket is in the China (Shanghai) region. The TXT file contains the following JSON data:

```
{
    "store": {
        "book": [
             {
                "category": "reference",
                "author": "Nigel Rees",
                "title": "Sayings of the Century",
                "price": 8.95
             },
             {
                "category": "fiction",
                "author": "Evelyn Waugh",
                "title": "Sword of Honour",
                "price": 12.99
             },
             {
                 "category": "fiction",
                 "author": "J. R. R. Tolkien",
                 "title": "The Lord of the Rings",
                 "isbn": "0-395-19395-8",
                 "price": 22.99
             }
          ],
          "bicycle": {
              "color": "red",
              "price": 19.95
          }
    },
    "expensive": 10
}
```

Migrate JSON data from OSS to MaxCompute

1. Add an OSS connection. For more information, see Add an OSS data source.
2. Create a table in DataWorks to store the JSON data to be migrated from OSS.
   i.
   ii.
   iii.
   iv. In the **DDL Statement** dialog box, enter the following statement and click **Generate Table Schema**:

   ```
   create table mqdata (mq_data string);
   ```

   v.
3. Create a batch synchronization node.
   i.
   ii.
   iii.
   iv.

v.

vi. Modify JSON code and click the icon.

Sample code:

```
{
    "type": "job",
    "steps": [
        {
            "stepType": "oss",
            "parameter": {
                "fieldDelimiterOrigin": "^",
                "nullFormat": "",
                "compress": "",
                "datasource": "OSS_userlog",
                "column": [
                    {
                        "name": 0,
                        "type": "string",
                        "index": 0
                    }
                ],
                "skipHeader": "false",
                "encoding": "UTF-8",
                "fieldDelimiter": "^",
                "fileFormat": "binary",
                "object": [
                    "applog.txt"
                ]
            },
            "name": "Reader",
            "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
                "partition": "",
                "isCompress": false,
                "truncate": true,
                "datasource": "odps_first",
                "column": [
                    "mqdata"
                ],
                "emptyAsNull": false,
                "table": "mqdata"
            },
            "name": "Writer",
            "category": "writer"
        }
    ],
    "version": "2.0",
    "order": {
        "hops": [
            {
```

```
                "from": "Reader",
                "to": "Writer"
            }
        ]
    },
    "setting": {
        "errorLimit": {
            "record": ""
        },
        "speed": {
            "concurrent": 2,
            "throttle": false,
        }
    }
}
```

Use the GET_JSON_OBJECT function to extract JSON objects

1. Create an ODPS SQL node.
   i.
   ii.
   iii. On the configuration tab of the ODPS SQL node, enter the following statements:

```
--Query data in the mqdata table.
SELECT * from mqdata;
--Obtain the value of the expensive field.
SELECT GET_JSON_OBJECT(mqdata.MQdata,'$.expensive') FROM mqdata;
```

   iv.
   v.

1.10. Migrate JSON-formatted data from MongoDB to MaxCompute

This topic describes how to use the Data Integration service of DataWorks to migrate JSON-formatted fields from MongoDB to MaxCompute.

Prerequisites

- MaxCompute is activated. For more information, see Activate MaxCompute and DataWorks.
- DataWorks is activated.
- A workflow is created in DataWorks. In this example, a DataWorks workspace in basic mode is used. For more information, see Create a workflow.

Prepare test data in MongoDB

1. Prepare an account.

   Create a user in your database to prepare information for creating a connection in DataWorks. In this example, run the following command:

```
db.createUser({user:"bookuser",pwd:"123456",roles:["root"]})
```

The username is bookuser, the password is 123456, and the permission is root.

2. Prepare data.

   Upload the data to the MongoDB database. In this example, an ApsaraDB for MongoDB instance in a virtual private cloud (VPC) is used. You must apply for a public endpoint for the ApsaraDB for MongoDB instance to communicate with the default resource group of DataWorks. The following test data is uploaded:

```
{
    "store": {
        "book": [
            {
                "category": "reference",
                "author": "Nigel Rees",
                "title": "Sayings of the Century",
                "price": 8.95
                },
            {
                "category": "fiction",
                "author": "Evelyn Waugh",
                "title": "Sword of Honour",
                "price": 12.99
                },
            {
                "category": "fiction",
                "author": "J. R. R. Tolkien",
                "title": "The Lord of the Rings",
                "isbn": "0-395-19395-8",
                "price": 22.99
                }
                    ],
        "bicycle": {
            "color": "red",
            "price": 19.95
                }
                    },
        "expensive": 10
            }
```

3. Log on to the MongoDB database in the Data Management (DMS) console. In this example, the name of the database is admin, and the name of the collection is userlog. You can run the following command to view the uploaded data:

```
db.userlog.find().limit(10)
```

Migrate JSON-formatted data from MongoDB to MaxCompute by using DataWorks

1.

2. Create a destination table in DataWorks. This table is used to store the data that is migrated from MongoDB.

i.

ii.

iii.

iv. In the **DDL Statement** dialog box, enter the following statement and click **Generate Table Schema**:

```
create table mqdata (mqdata string);
```

v. Click **Commit to Production Environment**.

3. Create a MongoDB connection. For more information, see Add a MongoDB data source.
4. Create a batch sync node.

i.

ii.

iii.

iv.

v.

vi. Enter the following script:

```
{
    "type": "job",
    "steps": [
    {
        "stepType": "mongodb",
        "parameter": {
            "datasource": "mongodb_userlog", // The name of the connection.
            "column": [
                {
                "name": "store.bicycle.color", // The path of the JSON-formatted fi
eld. In this example, the color field is extracted.
                "type": "document.String" // For fields other than top-level fields
, the data type of such a field is the type that is finally obtained. If the specif
ied JSON-formatted field is a top-level field, such as the expensive field in this
example, enter string.
                }
              ],
            "collectionName": "userlog" // The name of the collection.
            },
        "name": "Reader",
        "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
            "partition": "",
            "isCompress": false,
            "truncate": true,
            "datasource": "odps_first",
            "column": [
            "mqdata" // The name of the column in the MaxCompute table.
            ],
```

```
            "emptyAsNull": false,
            "table": "mqdata"
            },
            "name": "Writer",
            "category": "writer"
            }
            ],
            "version": "2.0",
            "order": {
            "hops": [
            {
            "from": "Reader",
            "to": "Writer"
            }
            ]
            },
            "setting": {
            "errorLimit": {
            "record": ""
            },
            "speed": {
            "concurrent": 2,
            "throttle": false,
            }
            }
        }
```

vii.

viii.

Verify the result

1.

2.

3. On the configuration tab of the ODPS SQL node, enter the following statement:

```
SELECT * from mqdata;
```

4.

5.

1.11. Migrate data from Elasticsearch to MaxCompute

This topic describes how to use the data synchronization feature of DataWorks to migrate data from an Alibaba Cloud Elasticsearch cluster to MaxCompute.

Prerequisites

- MaxCompute is activated. For more information, see Activate MaxCompute and DataWorks.
- DataWorks is activated.

- A workflow is created in DataWorks. In this example, a DataWorks workspace in basic mode is used. For more information, see Create a workflow.
- An Alibaba Cloud Elasticsearch cluster is created.

  Before you migrate data, make sure that your Alibaba Cloud Elasticsearch cluster works as expected. For more information about how to create an Alibaba Cloud Elasticsearch cluster, see Quick start.

  An Alibaba Cloud Elasticsearch cluster with the following configurations is used in this example:

  - Region: China (Shanghai)
  - Zone: Zone B
  - Version: Elasticsearch 5.5.3 with Commercial Feature

Context

Elasticsearch is a Lucene-based search server. It provides a distributed multi-tenant search engine that supports full-text search. Elasticsearch is an open source service that complies with the Apache open standards. It is a mainstream enterprise-class search engine.

Alibaba Cloud Elasticsearch includes Elasticsearch 5.5.3 with Commercial Feature, Elasticsearch 6.3.2 with Commercial Feature, and Elasticsearch 6.7.0 with Commercial Feature. It also contains the commercial X-Pack plug-in. You can use Alibaba Cloud Elasticsearch in scenarios such as data analysis and search. Based on open source Elasticsearch, Alibaba Cloud Elasticsearch provides enterprise-class access control, security monitoring and alerting, and automatic reporting.

Procedure

1. Create a source table in Elasticsearch. For more information, see Use DataWorks to synchronize data from MaxCompute to an Alibaba Cloud Elasticsearch cluster.
2. Create a destination table in MaxCompute.
   i.
   ii.
   iii.
   iv.
   v. In the **DDL Statement** dialog box, enter the following CREATE TABLE statement and click **Generate Table Schema**:

      ```
      create table elastic2mc_bankdata
      (
      age             string,
      job             string,
      marital         string,
      education       string,
      default         string,
      housing         string,
      loan            string,
      contact         string,
      month           string,
      day of week     string
      );
      ```

   vi.

3. Synchronize data.
   i.
   ii.
   iii.
   iv.
   v.
   vi. Configure the script.

   In this example, enter the following code. For more information about the code description, see Elasticsearch Reader.

```
{
    "type": "job",
    "steps": [
        {
            "stepType": "elasticsearch",
            "parameter": {
                "retryCount": 3,
                "column": [
                    "age",
                    "job",
                    "marital",
                    "education",
                    "default",
                    "housing",
                    "loan",
                    "contact",
                    "month",
                    "day_of_week",
                    "duration",
                    "campaign",
                    "pdays",
                    "previous",
                    "poutcome",
                    "emp_var_rate",
                    "cons_price_idx",
                    "cons_conf_idx",
                    "euribor3m",
                    "nr_employed",
                    "y"
                ],
                "scroll": "1m",
                "index": "es_index",
                "pageSize": 1,
                "sort": {
                    "age": "asc"
},
                "type": "elasticsearch",
                "connTimeOut": 1000,
                "retrySleepTime": 1000,
                "endpoint": "http://es-cn-xxxx.xxxx.xxxx.xxxx.com:9200",
                "password": "xxxx",
```

```
                "search": {
                    "match_all": {}
                },
                "readTimeOut": 5000,
                "username": "xxxx"
            },
            "name": "Reader",
            "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
                "partition": "",
                "truncate": true,
                "compress": false,
                "datasource": "odps_first",
                "column": [
                    "age",
                    "job",
                    "marital",
                    "education",
                    "default",
                    "housing",
                    "loan",
                    "contact",
                    "month",
                    "day_of_week",
                    "duration",
                    "campaign",
                    "pdays",
                    "previous",
                    "poutcome",
                    "emp_var_rate",
                    "cons_price_idx",
                    "cons_conf_idx",
                    "euribor3m",
                    "nr_employed",
                    "y"
                ],
                "emptyAsNull": false,
                "table": "elastic2mc_bankdata"
            },
            "name": "Writer",
            "category": "writer"
        }
    ],
    "version": "2.0",
    "order": {
        "hops": [
            {
                "from": "Reader",
                "to": "Writer"
            }
        ]
    }
```

```
    },
    "setting": {
        "errorLimit": {
            "record": "0"
        },
        "speed": {
            "throttle": false,
            "concurrent": 1,
            "dmu": 1
        }
    }
}
```

> **Note** On the **Basic Information** page of the created Alibaba Cloud Elasticsearch cluster, you can view the public IP address and port number in the Public Network Access and Public Network Port fields.

vii. Click the icon to run the code.

viii. You can view the running result on the **Runtime Log** tab.

4. View the result.

i.

ii.

iii. On the configuration tab of the ODPS SQL node, enter the following statement:

```
SELECT * FROM elastic2mc_bankdata;
```

iv.

v.

1.12. Use OTSStream Reader to configure a sync node

OTSStream Reader allows you to export incremental data from Tablestore. This topic describes how to configure a sync node by using OTSStream Reader.

Context

Unlike plug-ins that are used to export full data, OTSStream Reader supports only the multi-version mode and cannot be used to export data in specified columns. Incremental data can be considered as operations logs that include data and operation information. For more information, see OTSStream Reader.

Note When you use OTSStream Reader to configure a sync node, take note of the following items:

- If the node is scheduled to run by day, the node reads the data that is generated during the last 24 hours, but does not read the data that is generated in the last 5 minutes. We recommend that you schedule the node to run by hour.
- The end time that you specify cannot be later than the current system time. Therefore, the end time must be at least 5 minutes earlier than the scheduled time to run the node.
- When the node is scheduled to run by day, the data that is read may be incomplete.
- The node cannot be scheduled to run by week or month.

The time period from the start time to the end time must include the time when operations are performed on the Tablestore table. Assume that you inserted two data records to a Tablestore table at 16:20:00 on October 19, 2017. You can set the start time to 20171019161000 and the end time to 20171019162600.

Create a connection

1. Log on to the DataWorks console. Find the required workspace and click **Data Integration**.

   If you are using another service of DataWorks, click the More icon in the upper-left corner and choose **All Products > Data Integration** to go to the **Data Integration** page.
2. Click **Connection** in the left-side navigation pane to go to the **Data Source** page under **Workspace Management**.
3. Click **New data source** in the upper-right corner.
4. In the **Add data source** dialog box, select **OTS** as the connection type.
5. Set the parameters for the Tablestore connection.

Parameter	Description
Data Source Name	The name of the connection. The name can contain letters, digits, and underscores (_), and must start with a letter.
Data source description	The description of the connection. The description can be up to 80 characters in length.
Applicable environment	Valid values: **Development** and **Production**. Note This parameter is displayed only when the workspace is in standard mode.
Endpoint	The endpoint of Tablestore.
Table Store Instance name	The ID of the Tablestore instance.
AccessKey ID	The AccessKey ID of the AccessKey pair. You can copy the AccessKey ID on the **User Management** page.

Parameter	Description
AccessKey Secret	The AccessKey secret of the AccessKey pair, which is equivalent to the logon password.

6. Click **Test connectivity**.
7. After the connection passes the connectivity test, click **Complete**.

Configure a sync node on the codeless UI

1. On the **Data Source** page, click the More icon in the upper-left corner and choose **All Products > Data Integration**.
2. Click **New offline synchronization** on the homepage.
3. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters and click **Commit**.
4. On the configuration tab of the batch sync node, configure the connection to the source data store.

Parameter	Description
Connection	The name of the connection.
Table	The name of the table from which incremental data is exported. You must enable the Stream feature for the table when you create the table, or by calling the UpdateTable operation after you create the table.
Start Timestamp	The start time (included) in milliseconds of the incremental data. The format is yyyymmddhh24miss.
End Timestamp	The end time (excluded) in milliseconds of the incremental data. The format is yyyymmddhh24miss.
State Table	The name of the table that is used to store status records.
Maximum Retries	The maximum number of retries of each request that is used to read incremental data from Tablestore. Default value: *30*.
Export Time Information	Specifies whether to export time series information, including the time when data is written.

5. Configure the connection to the destination data store.

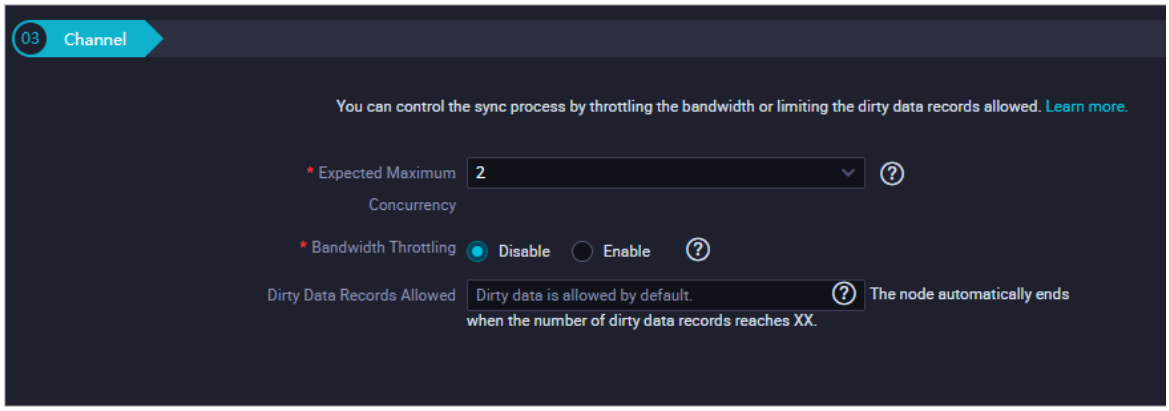
Parameter	Description
Connection	The name of the connection that you have configured.
Table	The table that you want to synchronize.
Partition Key Column	The table to be synchronized is a non-partitioned table. No partition key column is displayed.

Parameter	Description
Writing Rule	○ **Write with Original Data Deleted (Insert Overwrite)**: All data in the table or partition is deleted before data import. This rule is equivalent to the `INSERT OVERWRITE` statement. ○ **Write with Original Data Retained (Insert Into)**: No data is deleted before data import. New data is always appended. This rule is equivalent to the `INSERT INTO` statement.
Convert Empty Strings to Null	Default value: **No**.

6. Configure field mappings.

   Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field. To remove a field, move the pointer over the field and click the **Remove** icon.

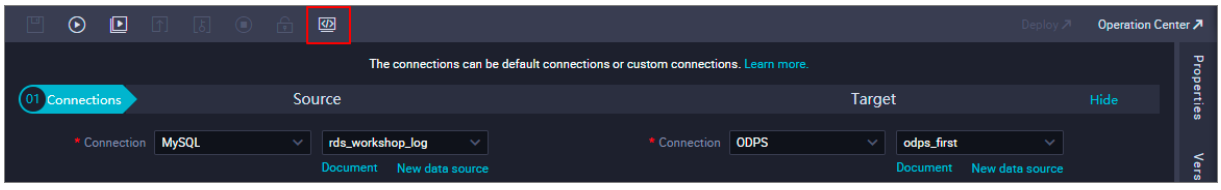
7. Configure channel control policies.



8. Click the **Save** icon in the top toolbar.
9. Click the **Run** icon in the top toolbar. You must set custom parameters before you run the sync node.

Configure a sync node by using the code editor

To configure the sync node by using the code editor, click the **Switch to Code Editor** icon in the top toolbar and click **OK**. The code editor appears.



Configure the sync node as required. You can use the following sample script:

```
{
  "type": "job",
  "version": "1.0",
  "configuration": {
    "reader": {
      "plugin": "otsstream",
      "parameter": {
        "datasource": "otsstream", // The name of the connection. Use the name of the conne
ction that you have created.
        "dataTable": "person", // The name of the table from which the incremental data is
exported. You must enable the Stream feature for the table when you create the table, or by
calling the UpdateTable operation after you create the table.
        "startTimeString": "${startTime}", // The start time (included) in milliseconds of
the incremental data. The format is yyyymmddhh24miss.
        "endTimeString": "${endTime}", // The end time (excluded) in milliseconds of the in
cremental data.
        "statusTable":"TableStoreStreamReaderStatusTable", // The name of the table that is
used to store status records.
        "maxRetries": 30, // The maximum number of retries of each request.
        "isExportSequenceInfo": false,
      }
    },
    "writer": {
      "plugin": "odps",
      "parameter": {
        "datasource":"odps_first", // The name of the connection.
        "table": "person", // The name of the destination table.
        "truncate": true,
        "partition": "pt=${bdp.system.bizdate}", // The information about the partition.
        "column": [ // The column to which data is written.
          "id",
          "colname",
          "version",
          "colvalue",
          "optype",
          "sequenceinfo"
        ]
      }
    },
    "setting": {
      "speed": {
        "mbps": 7, // The maximum transmission rate.
        "concurrent": 7 // The maximum number of concurrent threads.
      }
    }
  }
}
```

You can configure the start time and end time by using one of the following methods:

- `"startTimeString": "${startTime}"` : the start time (included) in milliseconds of the incremental data. The format is yyyymmddhh24miss.

`"endTimeString": "${endTime}"`: the end time (excluded) in milliseconds of the incremental data. The format is yyyymmddhh24miss.

- `"startTimestampMillis":""`: the start time (included) in milliseconds of the incremental data.

  OTSStream Reader searches for the status records in the table that is specified by the statusTable parameter based on the time that is specified by the startTimestampMillis parameter. Then, OTSStream Reader reads data from this point in time.

  If OTSStream Reader cannot find status records of this point in time in the table that is specified by the statusTable parameter, OTSStream Reader reads incremental data that is retained by the system from the first entry, and skips the data that is written earlier than the time that is specified by the startTimestampMillis parameter.

  `"endTimestampMillis":" "`: the end time (excluded) in milliseconds of the incremental data.

  OTSStream Reader reads data from the time that is specified by the startTimestampMilli parameter and stops when it reads the first entry that is written later than or equal to the time that is specified by the endTimestampMilli parameter.

  When OTSStream Reader reads all the incremental data, the reading process ends even if the time that is specified by the endTimestampMillis parameter has not arrived.

If the isExportSequenceInfo parameter is set to *true* (`"isExportSequenceInfo": true`), the system exports an extra column for time series information. The time series information contains the time when data is written. The default value is false, which indicates that no time series information is exported.

1.13. Connect an exclusive resource group for Data Integration to a data store that is deployed in a VPC

This topic uses ApsaraDB RDS as an example to describe how to connect an exclusive resource group for Data Integration to a data store that is deployed in a virtual private cloud (VPC).

Prerequisites

- An ApsaraDB RDS for MySQL instance is purchased. In this example, the ApsaraDB RDS for MySQL instance uses MySQL 5.7. You can configure the instance based on your business needs. For more information, see Create an ApsaraDB RDS for MySQL instance.
- A database is created in the ApsaraDB RDS for MySQL instance. The username and password that are used to log on to the database are obtained. For more information, see Create databases and accounts for an ApsaraDB RDS for MySQL instance.
- A VPC is created. For more information, see Create and manage a VPC.
- A vSwitch is created. For more information, see Work with vSwitches.
- A security group is created. For more information, see Create a security group.

Context

Assume that a data store has no public endpoint and resource groups for Data Integration cannot connect to the data store by using a VPC. When you configure a connection to the data store, the shared resource group or an exclusive resource group for Data Integration fails the connectivity test. This topic describes how to connect an exclusive resource group for Data Integration to a data store

that is deployed in a VPC.

Procedure

1. Obtain the information about the ApsaraDB RDS for MySQL instance.

   When you configure a connection to a database in the ApsaraDB RDS for MySQL instance, you must provide the instance ID, database name, username, and password.

   i. Log on to the ApsaraDB RDS console.

   ii. In the left-side navigation pane, click **Instances**.

   iii. On the **Instances** page, find the required instance and obtain the instance ID.

   iv. Find the required instance and click **Manage**.

   v. In the left-side navigation pane, click **Databases**. On the page that appears, find the database to which you want to create a connection and obtain the database name.

   vi. In the left-side navigation pane, click **Accounts**. Then, obtain the username of the account that you can use to connect to the database. You can enter the username when you create a connection to the database.

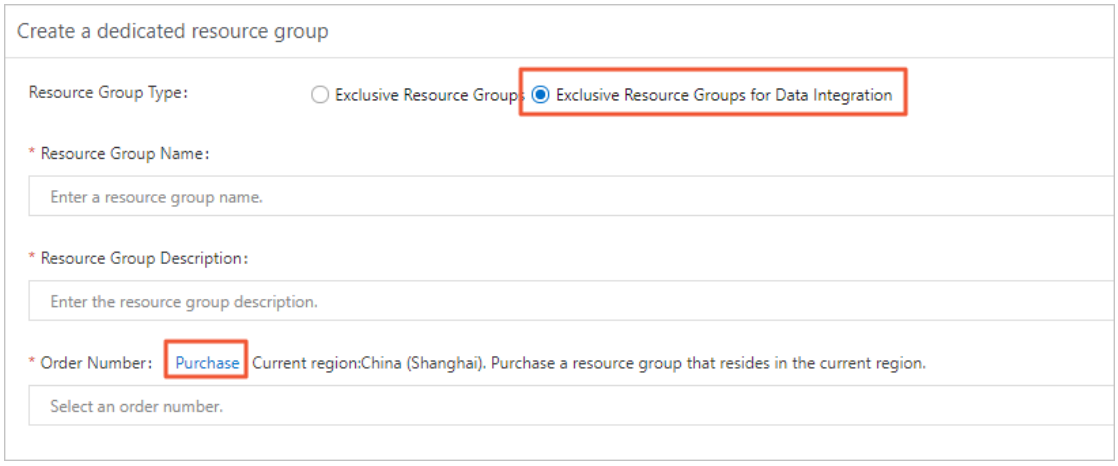
2. Create an exclusive resource group for Data Integration.

   i. Log on to the DataWorks console.

   ii. Select a region. In the left-side navigation pane, click **Resource Groups**.

   iii. On the **Exclusive Resource Groups** tab, click **Create a dedicated resource group**.

   iv. In the **Create a dedicated resource group** panel, set the **Resource Group Type** parameter to **Exclusive Resource Groups for Data Integration**. Then, click **Purchase** next to **Order Number** to go to the buy page.

   v. On the buy page, set the **Region**, **Type**, **Exclusive data integration resources**, **Units**, and **Duration** parameters as required. Then, click **Buy Now**.

   In this example, the ApsaraDB RDS for MySQL instance is purchased in the China (Shanghai) region. Therefore, you must set the **Region** parameter to **China (Shanghai)**. You can select specifications for the exclusive resource group for Data Integration based on your needs. For more information, see Performance metrics and pricing of exclusive resource groups for Data Integration.

   vi. After you confirm that the order information is correct, read and agree to **DataWorks Exclusive Resources Agreement of Service** by selecting the check box and click **Pay**.

vii. Go back to the **Create a dedicated resource group** panel and set the parameters as required.

Parameter	Description
Resource Group Type	The type of the resource group. In this example, select **Exclusive Resource Groups for Data Integration**.
Resource Group Name	The name of the resource group. The name must be unique within all resource groups of a tenant. **Note** A tenant indicates an Alibaba Cloud account. Multiple RAM users may exist for a tenant.
Resource Group Description	The description of the resource group.
Order Number	The order number of the exclusive resources that you purchase.

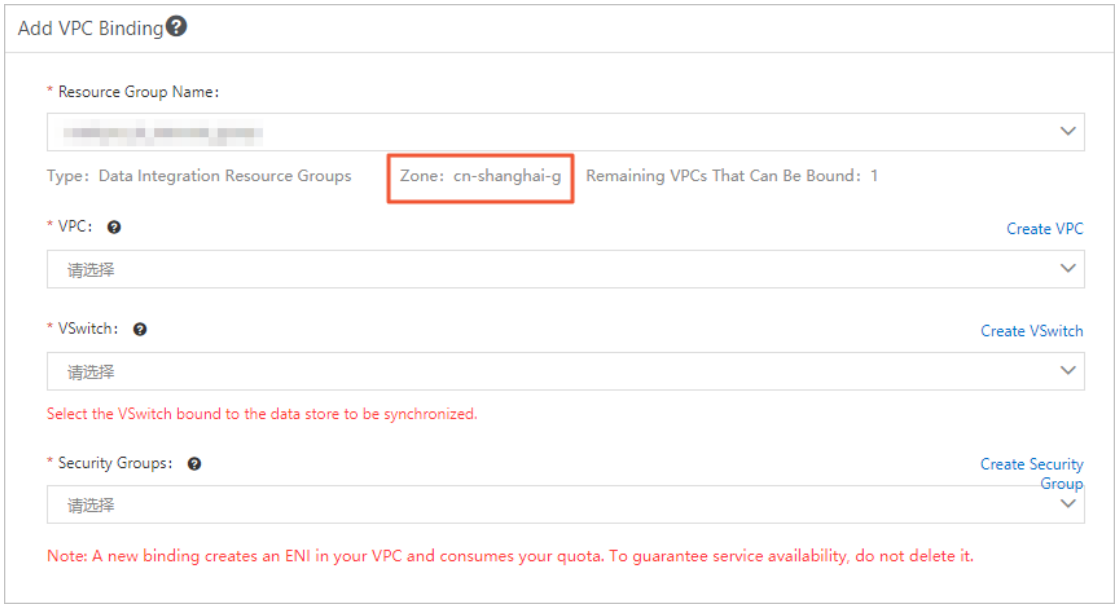
viii. Click **OK**.

3. Bind the exclusive resource group for Data Integration to the VPC.

i. On the **Exclusive Resource Groups** tab of the **Resource Groups** page, find the exclusive resource group for Data Integration and click **Add VPC Binding**.

Notice Before you bind the exclusive resource group for Data Integration to the VPC as a RAM user, make sure that the RAM user is authorized to access the ApsaraDB RDS for MySQL instance.

ii. Click **Add Binding** in the upper-right corner. In the **Add VPC Binding** panel, set the parameters as required.



Parameter	Description
Resource Group Name	Select the resource group that you want to bind to the VPC from the **Resource Group Name** drop-down list.
VPC	Select the VPC where the ApsaraDB RDS for MySQL instance is deployed from the **VPC** drop-down list.
VSwitch	Select the vSwitch where the ApsaraDB RDS for MySQL instance resides from the **VSwitch** drop-down list.
Security Groups	Select the required security group from the **Security Groups** drop-down list.

iii. Click **OK**.

4. (Optional)Add a route to the VPC.

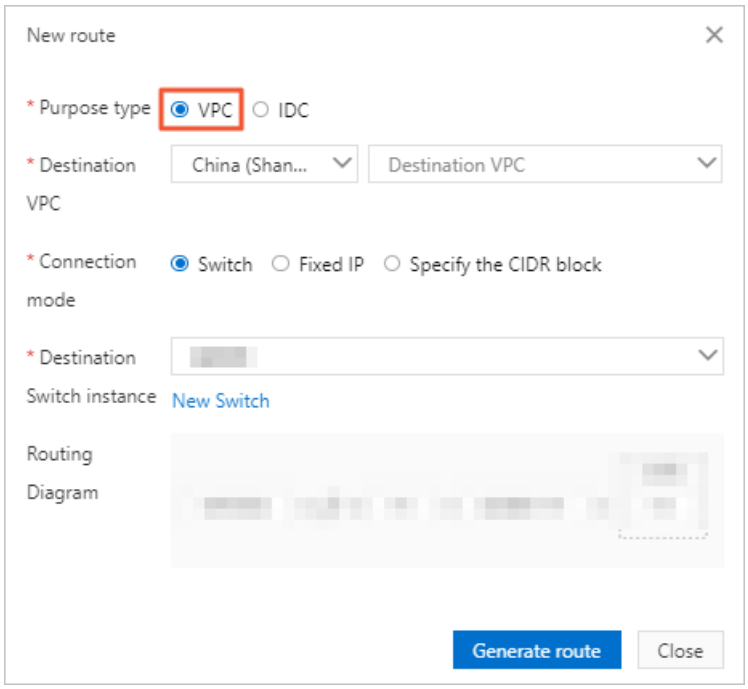
   The exclusive resource group for Data Integration must be in the same zone as the ApsaraDB RDS for MySQL instance. If they are in the same zone, skip this step. If they are in different zones, you must add a route.

   i. On the **Exclusive Resource Groups** tab, find the exclusive resource group for Data Integration and click **Add VPC Binding**.

   ii. Find the required resource group and click **Custom Route**.

   iii. In the **Custom Route** panel, click **Add Route**.

iv. In the **Add Route** dialog box, set the parameters as required.



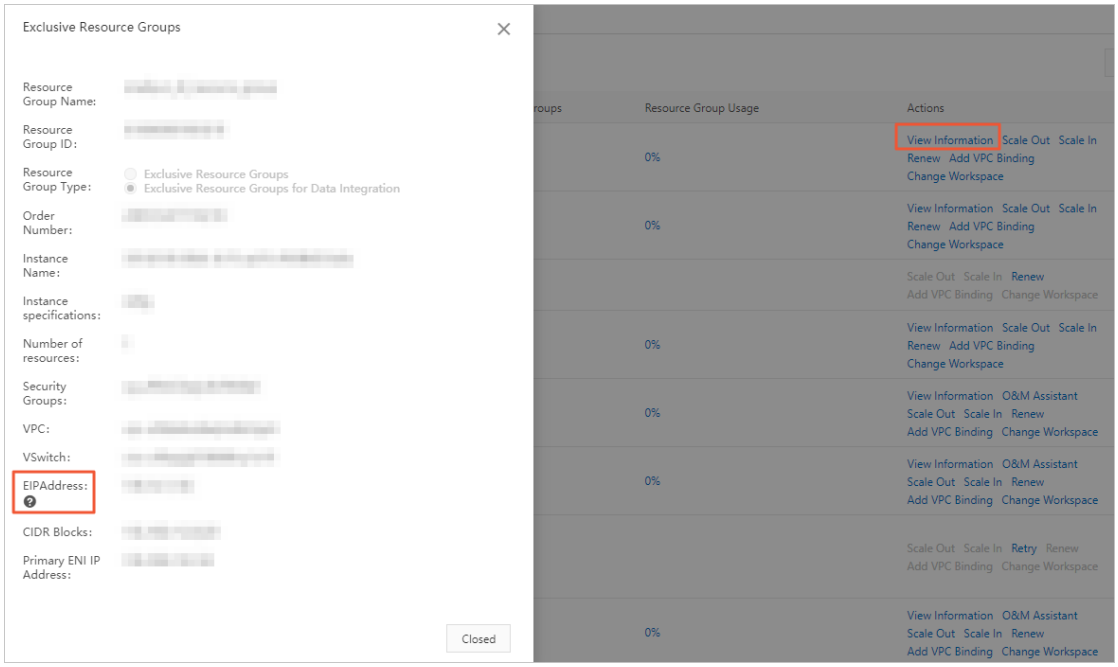
Parameter	Description
Destination Type	The data store in this example is deployed in the VPC. Set this parameter to **VPC**.
Destination VPC	The region and name of the VPC where the data store is deployed. **Note** This parameter is displayed only when the **Destination Type** parameter is set to **VPC**.
Connection Method	Valid values: **Switch**, **Fixed IP Address**, and **CIDR Block**. In this example, set this parameter to **Switch**.
Destination VSwitch	Select a specific vSwitch in the VPC as the destination vSwitch for routing.
Routing	By default, the value cannot be changed.

v. Click **Generate Route**.

5. Add the elastic IP address (EIP) and the elastic network interface (ENI) IP address of the exclusive resource group for Data Integration to the whitelist that allows access to the ApsaraDB RDS for MySQL instance.

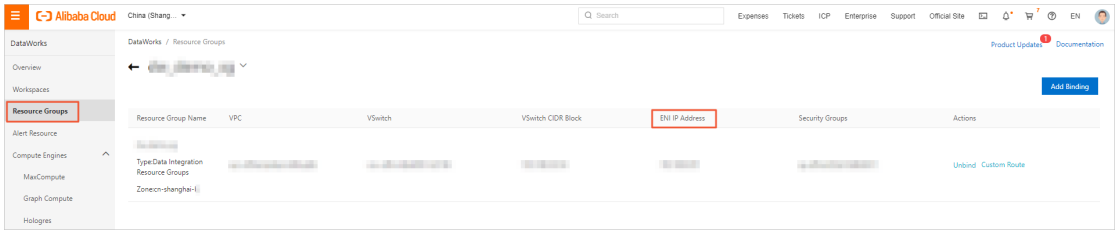
i. On the **Exclusive Resource Groups** tab, find the exclusive resource group for Data Integration and click **View Information**.

ii. In the **Exclusive Resource Groups** dialog box, view the EIP that is indicated by the **EIPAddress** parameter.



iii. On the **Exclusive Resource Groups** tab of the **Resource Groups** page, find the exclusive resource group for Data Integration and click **Add VPC Binding**.

iv. On the details page of the resource group, view the ENI IP address in the **ENI IP Address** column.



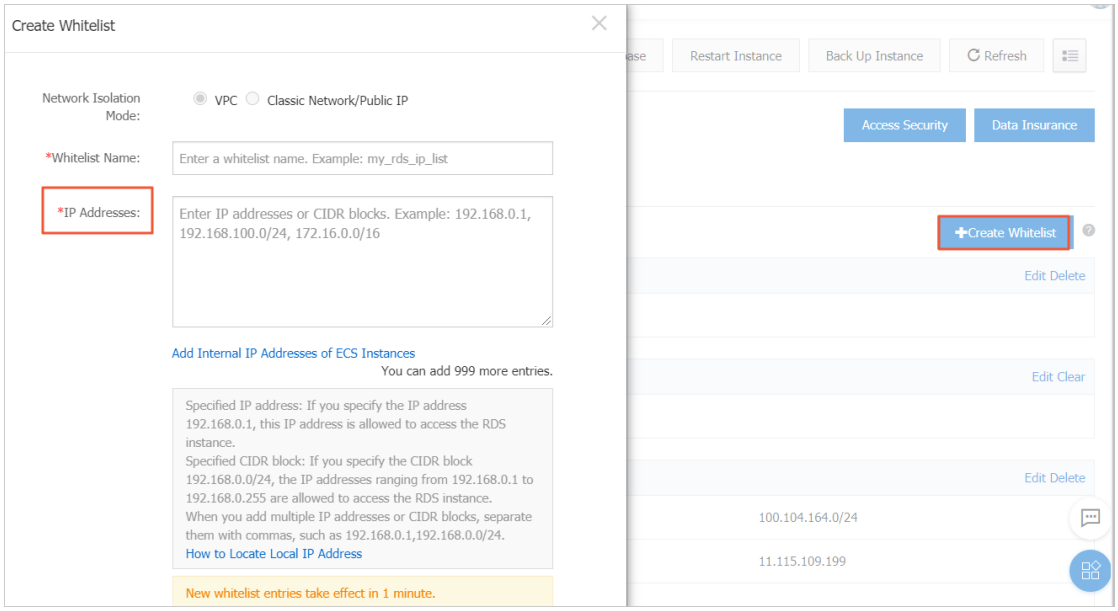
v. Log on to the ApsaraDB RDS console.

vi. In the upper-left corner of the **Instances** page, check whether the selected region is the region where the ApsaraDB RDS for MySQL instance resides. If not, select the required region. Find the ApsaraDB RDS for MySQL instance and click **Manage**.

vii. In the left-side navigation pane, click **Data Security**.

viii. On the **Data Security** page, click **Create Whitelist**.

ix. In the **Create Whitelist** dialog box, set the parameters as required and click **Add**.



Parameter	Description
Network Isolation Mode	By default, the value cannot be changed.
Whitelist Name	The name of the whitelist. The name must comply with the following conventions: ■ It consists of lowercase letters, digits, and underscores (_). ■ It must start with a lowercase letter and end with a digit or lowercase letter. ■ It must be 2 to 32 characters in length.
IP Addresses	Use the EIP and ENI IP address of the exclusive resource group for Data Integration.

6. Create a connection to the specified database in the ApsaraDB RDS for MySQL instance and test the connectivity.
   i. Log on to the DataWorks console.
   ii. In the left-side navigation pane, click **Workspaces**.
   iii. After you select the region where the required workspace resides, find the workspace and click **Data Integration**.
   iv. In the left-side navigation pane, click **Connection** to go to the **Data Source** page.
   v. On the **Data Source** page, click **New data source** in the upper-right corner.
   vi. In the **Add data source** dialog box, select **MySQL** as the connection type.

vii. In the **Add MySQL data source** dialog box, set the parameters as required.

You can set the Data source type parameter to **Alibaba Cloud instance mode** or **Connection string mode**. In this example, select **Alibaba Cloud instance mode**.

Parameter	Description
Data source type	The type of the connection. Set this parameter to **Alibaba Cloud instance mode**.
Data Source Name	The name of the connection. The name can contain letters, digits, and underscores (_), and must start with a letter.
Data source description	The description of the connection. The description can be up to 80 characters in length.
Environment	Valid values: **Development** and **Production**. **Note** This parameter is displayed only when the workspace is in standard mode.
Region	The region of the ApsaraDB RDS for MySQL instance.
RDS instance ID	The ID of the ApsaraDB RDS for MySQL instance.
RDS instance account ID	The ID of the Alibaba Cloud account that is used to purchase the ApsaraDB RDS for MySQL instance. Log on to the DataWorks console. Move the pointer over the profile picture in the upper-right corner and select **Security Settings**. Then, view the ID of your Alibaba Cloud account.
Database name	The name of the database.
User name	The username that is used to log on to the database.
Password	The password that is used to log on to the database.

viii. On the **Data Integration** tab, find the required resource group and click **Test connectivity**.

ix. After the connection passes the connectivity test, click **Complete**.

1.14. Call an API to change the source of a data synchronization node from a MySQL data source to a PolarDB data source

This topic describes how to change the source of a synchronization node whose destination is MaxCompute from a MySQL data source to a PolarDB data source.

Sample code of POM dependencies

```
<dependency>
        <groupId>com.aliyun</groupId>
        <artifactId>aliyun-java-sdk-core</artifactId>
        <version>4.5.20</version>
</dependency>
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-dataworks-public</artifactId>
    <version>3.4.4</version>
 </dependency>
```

Sample code of the SDK for Java

```
package com.alibaba.eas;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.dataworks_public.model.v20200518.*;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.List;
public class updateOfflineTask {
    public static ListFilesResponse.Data.File ListFiles(String filePath, String fileName) t
hrows Exception {
        ListFilesRequest request = new ListFilesRequest();
        request.setProjectId(1911L);
        request.setFileFolderPath(filePath);
        request.setKeyword(fileName);
        ListFilesResponse response1 = client.getAcsResponse(request);
        for(int i = 0 ; i < response1.getData().getFiles().size(); ++i) {
            return response1.getData().getFiles().get(i);
        }
        return null;
    }
    public static String GetFiles(Long fileId) throws Exception {
        GetFileRequest request = new GetFileRequest();
        request.setProjectId(1911L);
        request.setFileId(fileId);
        GetFileResponse response1 = client.getAcsResponse(request);
        return response1.getData().getFile().getContent();
    }
    public static void UpdateDISyncTask(Long fileId, String content) throws Exception {
        UpdateDISyncTaskRequest request = new UpdateDISyncTaskRequest();
        request.setProjectId(1911L);
        request.setFileId(fileId);
        request.setTaskContent(content);
        request.setTaskType("DI_OFFLINE");
        UpdateDISyncTaskResponse response1 = client.getAcsResponse(request);
    }
    public static  Long submitFile(Long fileId) throws  Exception {
        SubmitFileRequest request = new SubmitFileRequest();
```

```
        request.setProjectId(1911L);
        request.setFileId(fileId);
        SubmitFileResponse acsResponse = client.getAcsResponse(request);
        Long deploymentId = acsResponse.getData();
        return deploymentId;
    }
    public static  void getDeployment(Long deploymentId) throws Exception {
        GetDeploymentRequest request = new GetDeploymentRequest();
        request.setProjectId(1911L);
        request.setDeploymentId(deploymentId);
        GetDeploymentResponse acsResponse = client.getAcsResponse(request);
        System.out.println(acsResponse.getData().getDeployment().getStatus());
    }
    public static  Long deploy(Long fileId) throws Exception {
        DeployFileRequest request = new DeployFileRequest();
        request.setProjectId(1911L);
        request.setFileId(fileId);
        DeployFileResponse acsResponse = client.getAcsResponse(request);
        Long deploymentId = acsResponse.getData();
        return deploymentId;
    }
    public static Long listNode(String nodeName) throws Exception {
        ListNodesRequest request = new ListNodesRequest();
        request.setProjectId(1911L);
        request.setNodeName(nodeName);
        request.setProjectEnv("PROD");
        ListNodesResponse acsResponse = client.getAcsResponse(request);
        List<ListNodesResponse.Data.NodesItem> nodesItemList = acsResponse.getData().getNod
es();
        return nodesItemList.get(0).getNodeId();
    }
    public static void RunCycleDagNodes(Long nodeId) throws Exception {
        RunCycleDagNodesRequest request = new RunCycleDagNodesRequest();
        request.setIncludeNodeIds(nodeId.toString());
        request.setName("rerun_job");
        request.setParallelism(false);
        request.setProjectEnv("PROD");
        request.setRootNodeId(nodeId);
        request.setStartBizDate("2021-09-29 00:00:00");
        request.setEndBizDate("2021-09-29 00:00:00");
        request.setProjectEnv("PROD");
        RunCycleDagNodesResponse acsResponse = client.getAcsResponse(request);
    }
  /*
  Replace the values of the stepType and datasource parameters in the following code snippe
t with polardb:
  {
    "type": "job",
    "version": "2.0",
    "steps": [
        {
            "stepType": "mysql",
            "parameter": {
                "envType": 0,
```

```
                "datasource": "mysql_from_polardb",
                "column": [
                    "id",
                    "name",
                    "create_time",
                    "create_user"
                ],
                "tableComment": "Test",
                "connection": [
                    {
                        "selectedDatabase": "polardb_db1",
                        "datasource": "mysql_from_polardb",
                        "table": [
                            "lcl_test_demo"
                        ]
                    }
                ],
                "where": "",
                "splitPk": "id",
                "encoding": "UTF-8"
            },
            "name": "Reader",
            "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
                "partition": "pt=${bizdate}",
                "truncate": true,
                "datasource": "odps_first",
                "envType": 0,
                "column": [
                    "id",
                    "name",
                    "create_time",
                    "create_user"
                ],
                "emptyAsNull": false,
                "tableComment": "Test",
                "table": "lcl_test_demo"
            },
            "name": "Writer",
            "category": "writer"
        }
    ],
    "setting": {
        "errorLimit": {
            "record": ""
        },
        "locale": "zh_CN",
        "speed": {
            "throttle": false,
            "concurrent": 2
        }
```

```
    },
    "order": {
        "hops": [
            {
                "from": "Reader",
                "to": "Writer"
            }
        ]
    }
}
```

The following code snippet shows the replacement results:

```
  {
    "type": "job",
    "version": "2.0",
    "steps": [
        {
            "stepType": "polardb",
            "parameter": {
                "envType": 0,
                "datasource": "polardb",
                "column": [
                    "id",
                    "name",
                    "create_time",
                    "create_user"
                ],
                "tableComment": "Test",
                "connection": [
                    {
                        "selectedDatabase": "polardb_db1",
                        "datasource": "polardb",
                        "table": [
                            "lcl_test_demo"
                        ]
                    }
                ],
                "where": "",
                "splitPk": "id",
                "encoding": "UTF-8"
            },
            "name": "Reader",
            "category": "reader"
        },
        {
            "stepType": "odps",
            "parameter": {
                "partition": "pt=${bizdate}",
                "truncate": true,
                "datasource": "odps_first",
                "envType": 0,
                "column": [
                    "id",
                    "name",
                    "create_time",
                    "create_user"
```

```
                ],
                "emptyAsNull": false,
                "tableComment": "Test",
                "table": "lcl_test_demo"
            },
            "name": "Writer",
            "category": "writer"
        }
    ],
    "setting": {
        "errorLimit": {
            "record": ""
        },
        "locale": "zh_CN",
        "speed": {
            "throttle": false,
            "concurrent": 2
        }
    },
    "order": {
        "hops": [
            {
                "from": "Reader",
                "to": "Writer"
            }
        ]
    }
}
  */
    public static String modifyContent(String content, String newStepType, String newDataso
urce){
        JSONObject jsonObject = JSON.parseObject(content);
        JSONArray steps = jsonObject.getJSONArray("steps");
        if (steps != null) {
            for (int i = 0; i < steps.size(); ++i) {
                JSONObject step = steps.getJSONObject(i);
                if (step != null && step.getString("category") != null && "reader".equals(s
tep.getString("category"))) {
                    if (step.getString("stepType") != null && "mysql".equals(step.getString
("stepType"))) {
                        step.put("stepType", newStepType);
                        JSONObject parameter = step.getJSONObject("parameter");
                        if (parameter  != null) {
                            parameter.put("datasource", newDatasource);
                            JSONArray connections = parameter.getJSONArray("connection");
                            if (connections != null) {
                                for (int j = 0; j < connections.size(); ++j) {
                                    JSONObject connection = connections.getJSONObject(j);
                                    if (connection != null) {
                                        connection.put("datasource", newDatasource);
                                    }
                                }
                            }
                        }
```

```
                    }
                }
            }
        }
        return  jsonObject.toJSONString();
    }
    static IAcsClient client;
    public static void main(String[] args) throws Exception {
        String akId = "XXX";
        String akSecret = "XXX";
        String regionId = "cn-chengdu";
        IClientProfile profile = DefaultProfile.getProfile(regionId, akId, akSecret);
        DefaultProfile.addEndpoint(regionId, "dataworks-public", "dataworks." + regionId +
".aliyuncs.com");
        client = new DefaultAcsClient(profile);
        String folderPath = "Business Flow/Test Workflow/Data Integration/";
        String filename = "mysql_to_odps";
        ListFilesResponse.Data.File file = ListFiles(folderPath, filename);
        Long fileId = file.getFileId();
        System.out.println(file.getFileId());
        String content = GetFiles(fileId);
        String contentModified = modifyContent(content, "polardb", "polardb_datasource");
        //  "polardb" indicates the type of the new data source. The original MySQL data so
urce is replaced by a PolarDB data source.
        //
        UpdateDISyncTask(file.getFileId(), contentModified);
        Long deployId = submitFile(fileId);
        getDeployment(deployId);
        Thread.sleep(10000);
        getDeployment(deployId);
        deployId = deploy(fileId);
        getDeployment(deployId);
        Thread.sleep(10000);
        getDeployment(deployId);
        Long nodeId = listNode(filename);
        RunCycleDagNodes(nodeId);
    }
}
```

2.Data development

2.1. Best practice to configure scheduling dependencies

When you configure scheduling dependencies, ancestor and descendant nodes are associated by output names. This topic describes how to manage the input and output on which a node depends for scheduling.

Configure the node input

You can configure the node input in one of the following ways:

- Use the automatic parsing feature to parse node dependencies from the code.
- Enter the output name of the parent node to manually configure node dependencies.

> **Note** When you specify a parent node, you enter the output name of the parent node. If the name of the parent node is different from the output name of the parent node, you must enter the output name.

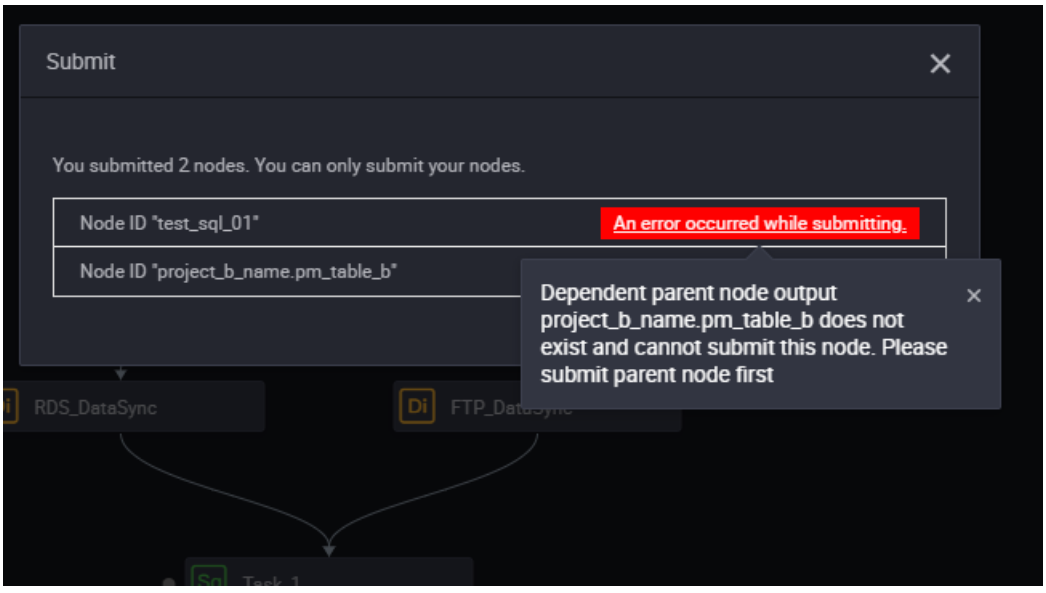
If you use the automatic parsing feature to configure the node input, the dependencies that are automatically parsed may be invalid. To determine whether a parsed dependency is valid, find the parsed parent node and check whether a value is displayed in the **Parent Node ID** column.

A dependency is a logical relationship between two nodes. You can configure valid dependencies only for nodes that actually exist.

Invalid dependencies

A dependency is invalid in the following two cases:

- The parent node does not exist.
- The output name of the parent node does not exist.



Usually, the dependency is invalid because the output name of the parsed parent node does not exist. Assume that the project_b_name.pm_table_b table has no output, or the output of the table is incorrectly configured. The dependency is invalid.

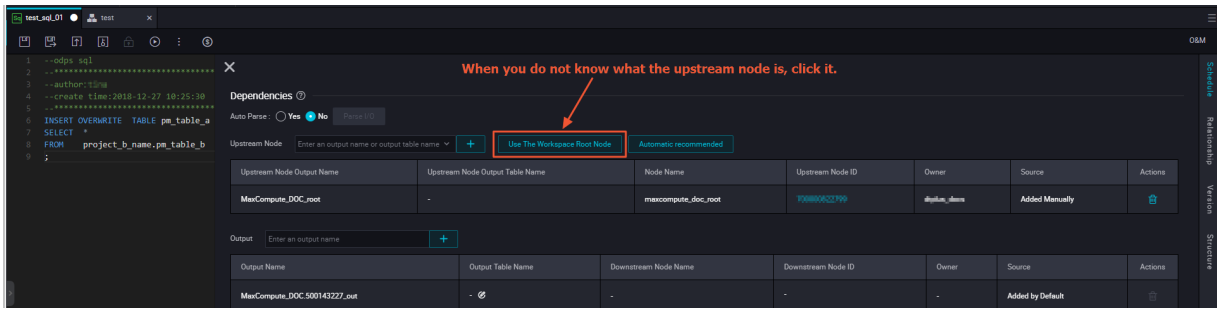
To resolve this issue, perform the following steps:

- Check whether the table has an output.
- If the table has an output, add the output to the Parent Nodes section for the node that depends on the output.

> **Note** Note: When you specify a parent node, you enter the output name of the parent node. If the name of the parent node is different from the output name of the parent node, you must enter the output name.
>
> Assume that the output name of Node A is A1, and Node B depends on Node A. In this case, enter A1 in the search box, and then click the plus sign (+) next to the search box.

Configure a parent node

If your table has no parent node, you can click **Use Root Node**.



Configure the node output

You can use the same name for the name, the output name, and the output table name of a node to efficiently configure the node output.

- You can know a specific table on which the node performs operations.
- You can know the impact that is caused when the node fails to be run.
- Assume that you use the automatic parsing feature to configure the node output. If the name, output name, and output table name of the node are the same, you can improve the precision of automatic parsing.

Automatic parsing

Auto Parse: Node dependencies are automatically parsed from the code.

The principle of auto parsing: Only table names can be obtained from the code. The automatic parsing feature is used to parse output nodes based on the table names.

Assume that the following code is written for a node:

```
INSERT OVERWRITE TABLE pm_table_a SELECT * FROM project_b_name.pm_table_b ;
```

DataWorks automatically parses the current node. The input of the current node is the `pm_table_b` table in the `project_b_name` workspace. The output of the current node is the `pm_table_a` table. The output name of the parent node is `project_b_name.pm_table_b`. The output name of the current node is `project_name.pm_table_a`. In this example, Workspace test_pm_01 is used.

- If you do not want to parse node dependencies from the code, set the Auto Parse parameter to **No**.
- The code may contain many temporary tables whose names start with **t_**. Temporary tables are not involved in the parsing of a scheduling dependency. You can specify the prefix of temporary table names on the Workspace Settings tab.
- If a table in the code of a node is both an output table and a referenced table on which another table depends, the table is parsed only as an output table.
- If a table in the code of a node is used as an output table or a referenced table multiple times, only one scheduling dependency is parsed.

> **Note** By default, a table whose name starts with **t_** is recognized as a temporary table. Temporary tables are not involved in the parsing of a scheduling dependency. If a table whose name starts with **t_** is not a temporary table, contact the workspace administrator to modify related settings on the **Workspace Settings** tab.

Delete table input and output

Static tables are often used for data analytics. If you import data from local files to static tables, these tables do not have outputs.

When you configure a dependency, you must prevent the static table from being parsed as the input of a node. If the name of a static table does not start with **t_**, it is not recognized as a temporary table. In this case, delete the table input.

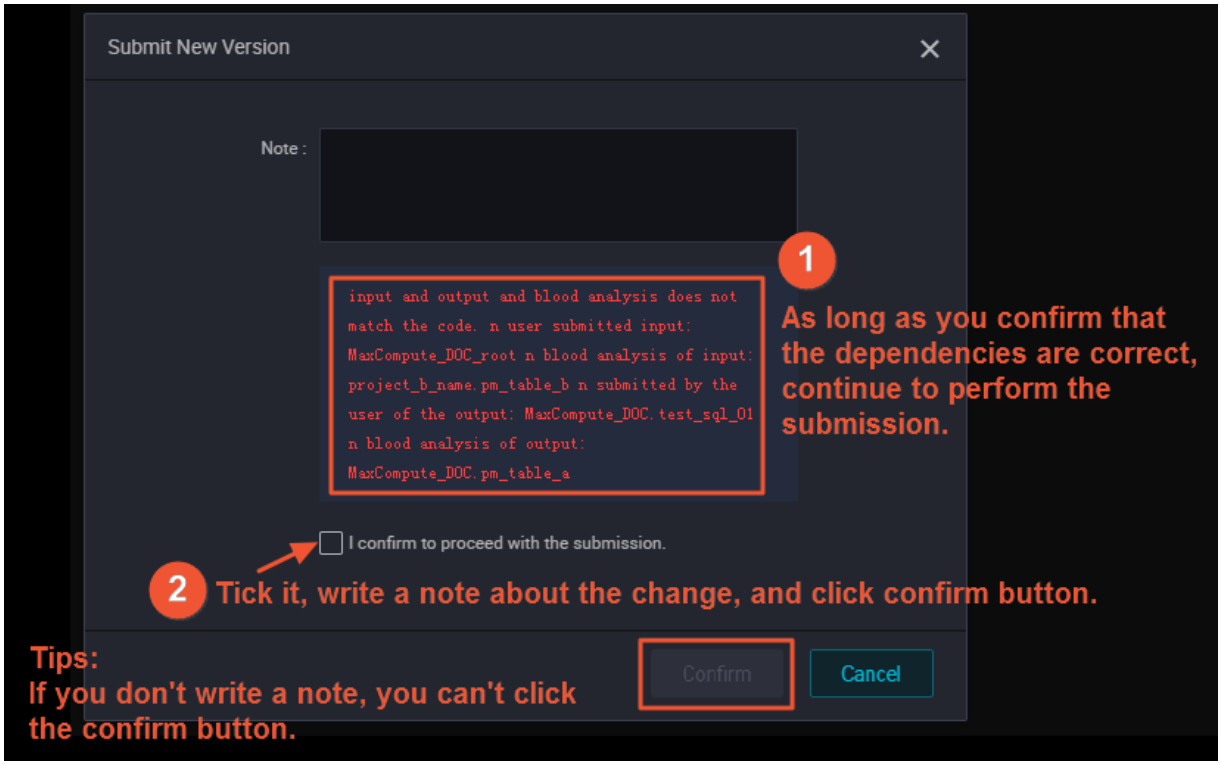
Right-click the table name in the code and select **Delete Input**.

If you upgrade the DataWorks service from V1.0 to V2.0, the default output name of the migrated node is in the format of `Workspace name. Node name`.

Usage notes

After you configure dependencies and commit the node, a dialog box appears. If the dependencies you specified are different from those obtained based on lineage analysis, the dialog box contains a check box that asks you to confirm whether to commit the node.

Make sure that the dependencies are correct before you select the check box. If you are not sure about the dependencies, reconfigure the dependencies based on the preceding steps.



If you are sure that the dependencies are correct, select **I confirm to proceed with the commission** and click **OK**.

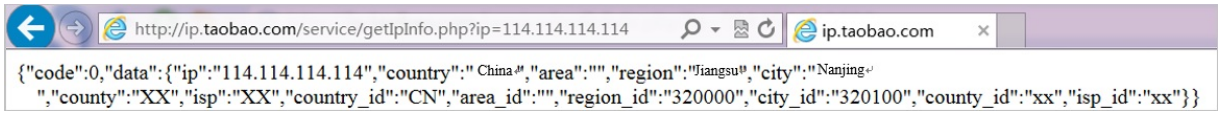
2.2. Use MaxCompute to query geolocations of IP addresses

This topic describes how to use MaxCompute to query geolocations of IP addresses. To query geolocations of IP addresses, you can download an IP address geolocation library, upload the library to MaxCompute, create a user-defined function (UDF), and then execute an SQL statement.

Prerequisites

Context

You can send an HTTP request to call the API operation provided by the IP address geolocation library of Taobao to query the geolocation of an IP address. The API operation returns the geolocation information in a string, as shown in the following figure.



However, HTTP requests are not allowed in MaxCompute. You can query geolocations of IP addresses in MaxCompute by using one of the following methods:

- Execute SQL statements to download IP address data from the IP address geolocation library of Taobao to a local computer. Then, send HTTP requests to query the data.

> **Note** This method is inefficient. In addition, the query frequency must be less than 10 queries per second (QPS). Otherwise, query requests are rejected by the IP address geolocation library of Taobao.

- Download the IP address geolocation library to a local computer. Then, query the geolocation information from the local library.

  > **Note** This method is inefficient and is not suitable when the data needs to be uploaded and analyzed in a data warehouse service.

- Maintain an IP address geolocation library and upload it to MaxCompute regularly. Then, query the geolocation information in MaxCompute.

  > **Note** This method is efficient. However, you must maintain the IP address geolocation library regularly.

Download an IP address geolocation library

1. Obtain an IP address geolocation library. In this example, the sample IP address geolocation library ipdata.txt.utf8 is used. This IP address geolocation library is an incomplete library in UTF-8 format.
2. Download the ipdata.txt.utf8 file. The following figure shows the data in the file.

   ```
   0,16777215,"0.0.0.0","0.255.255","","Intranet IP","Intranet IP","Intranet IP"
   16777216,16777471,"1.0.0.0","1.0.0.255","Australia","","","",""
   16777472,16778239,"1.0.1.0","1.0.3.255","China","Fujian","Fuzhou","Telecom"
   ```

   The following content describes the data in the sample IP address geolocation library.

   - The format of the data is UTF-8.
   - The first two strings in a data record are the start IP address and end IP address of an IP address range, in the format of decimal integer literal. The third and fourth strings are equivalent to the first two strings, but are expressed in dotted decimal notation. The decimal integer literal format makes it easy to check whether an IP address is within a specific IP address range.

   > **Note** You can also use your own IP address geolocation library.

Upload the IP address library

1. Execute the following statement on the MaxCompute client to create the ipresource table that is used to store IP address geolocation data:

```
DROP TABLE IF EXISTS ipresource ;
CREATE TABLE IF NOT EXISTS ipresource
(
    start_ip BIGINT
    ,end_ip BIGINT
    ,start_ip_arg string
    ,end_ip_arg string
    ,country STRING
    ,area STRING
    ,city STRING
    ,county STRING
    ,isp STRING
);
```

2. Run the following Tunnel command to upload data in the ipdata.txt.utf8 file to the ipresource table:

```
odps@ workshop_demo>tunnel upload D:/ipdata.txt.utf8 ipresource;
```

In the preceding command, *D:/ipdata.txt.utf8* is the local path of the ipdata.txt.utf8 file. For more information about the command, see Tunnel commands.

You can execute the following statement to check whether the data in the file is uploaded:

```
--Count the number of the data records in the ipresource table.
select count(*) from ipresource;
```

3. Execute the following SQL statement to obtain the first 10 data records in the ipresource table:

```
select * from ipresource limit 10;
```

The following figure shows the result of the preceding SQL statement.

```
Job Queueing...
+------------+------------+--------------+--------------+---------+------+---------+--------+------+
| start_ip   | end_ip     | start_ip_arg | end_ip_arg   | country | area | city    | county | isp  |
+------------+------------+--------------+--------------+---------+------+---------+--------+------+
| 3395369026 | 3395369026 | "202.97.56.66" | "202.97.56.66" | "China" | "Hunan"     | "Changsha" | ""  | "Telecom" |
| 3395369027 | 3395369028 | "202.97.56.67" | "202.97.56.68" | "China" | "Heilongjiang" | ""      | ""  | "Telecom" |
| 3395369029 | 3395369029 | "202.97.56.69" | "202.97.56.69" | "China" | "Anhui"     | "Hefei"    | ""  | "Telecom" |
| 3395369030 | 3395369030 | "202.97.56.70" | "202.97.56.70" | "China" | "Hunan"     | "Changsha" | ""  | "Telecom" |
| 3395369031 | 3395369033 | "202.97.56.71" | "202.97.56.73" | "China" | "Heilongjiang" | ""      | ""  | "Telecom" |
| 3395369034 | 3395369034 | "202.97.56.74" | "202.97.56.74" | "China" | "Hunan"     | "Changsha" | ""  | "Telecom" |
| 3395369035 | 3395369036 | "202.97.56.75" | "202.97.56.76" | "China" | "Heilongjiang" | ""      | ""  | "Telecom" |
| 3395369037 | 3395369037 | "202.97.56.77" | "202.97.56.77" | "China" | "Jiangsu"   | "Nanjing"  | ""  | "Telecom" |
| 3395369038 | 3395369038 | "202.97.56.78" | "202.97.56.78" | "China" | "Hunan"     | "Changsha" | ""  | "Telecom" |
| 3395369039 | 3395369040 | "202.97.56.79" | "202.97.56.80" | "China" | "Heilongjiang" | ""      | ""  | "Telecom" |
+------------+------------+--------------+--------------+---------+------+---------+--------+------+
```

Create a UDF

1.
2. Create a Python resource.
   i.
   ii.

iii. Enter the following code in the code editor:

```
from odps.udf import annotate
@annotate("string->bigint")
class ipint(object):
    def evaluate(self, ip):
        try:
            return reduce(lambda x, y: (x << 8) + y, map(int, ip.split('.')))
        except:
            return 0
```

iv.

3. Create a function.

i. Right-click the workflow that you created and choose **Create > MaxCompute > Function**.

ii. In the **Create Function** dialog box, set the **Function Name** parameter and click **Commit**.

> **Note** If multiple MaxCompute compute engines are bound to the current workspace, you must select one from the **Engine Instance MaxCompute** drop-down list.

iii. On the configuration tab of the function, set the parameters as required.

Parameter	Description
Function Type	The type of the function. Valid values: **Mathematical Function**, **Aggregate Function**, **String Function**, **Date Function**, **Analytic Function**, and **Other**.
Engine Instance MaxCompute	The MaxCompute engine that is bound to the current workspace. By default, you cannot change the engine.
Function Name	The name of the function, that is, the name used to reference the function in SQL. The function name must be globally unique and cannot be changed after the function is created.
Owner	The owner of the function. This parameter is automatically set.

Parameter	Description
Class Name	Required. The name of the class that implements the function. **Note** If the resource type is Python, enter the class name in the Python resource name.Class name format. Do not include the .py extension in the resource name.
Resources	Required. The list of resources. You can search for existing resources in the current workspace in fuzzy match mode. Separate multiple resources with commas (,).
Description	The description of the function.
Expression Syntax	The syntax of the function, for example, `test`.
Parameter Description	The description of supported input and output parameters.
Return Value	Optional. The value to return. Example: 1.
Example	Optional. The example of the function.

4. Click the icon in the toolbar.
5. Commit the function.
   i. Click the icon in the toolbar to commit the function.
   ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.
   iii. Click **OK**.

Execute an SQL statement to query the geolocation of an IP address

1.
2.
3. On the configuration tab of the ODPS SQL node, enter the following statement:

```
select * from ipresource
WHERE ipint('1.2.24.2') >= start_ip
AND ipint('1.2.24.2') <= end_ip
```

4.
5.

2.3. Use a PyODPS node to reference a third-party package

This topic describes how to use a PyODPS node in DataWorks to reference a third-party package. You can reference a common Python script or a third-party open source package.

Reference a common Python script

1.
2. Create a Python resource.
   i. On the **DataStudio** page, move the pointer over the icon and choose **MaxCompute > Resource > Python**.

      Alternatively, you can click the name of the desired workflow in the **Business Flow** section, right-click **MaxCompute**, and then choose **Create > Resource > Python**.
   ii. In the **Create Resource** dialog box, set the **Resource Name** and **Location** parameters. In this example, the Resource Name parameter is set to pyodps_packagetest.py.

      > **Notice** The resource name can contain letters, digits, periods (.), underscores (_), and hyphens (-) and must end with .py.

   iii. Click **Create**.
   iv. On the configuration tab of the newly created Python resource, enter the common Python script that you want to reference. In this example, the following script is used:

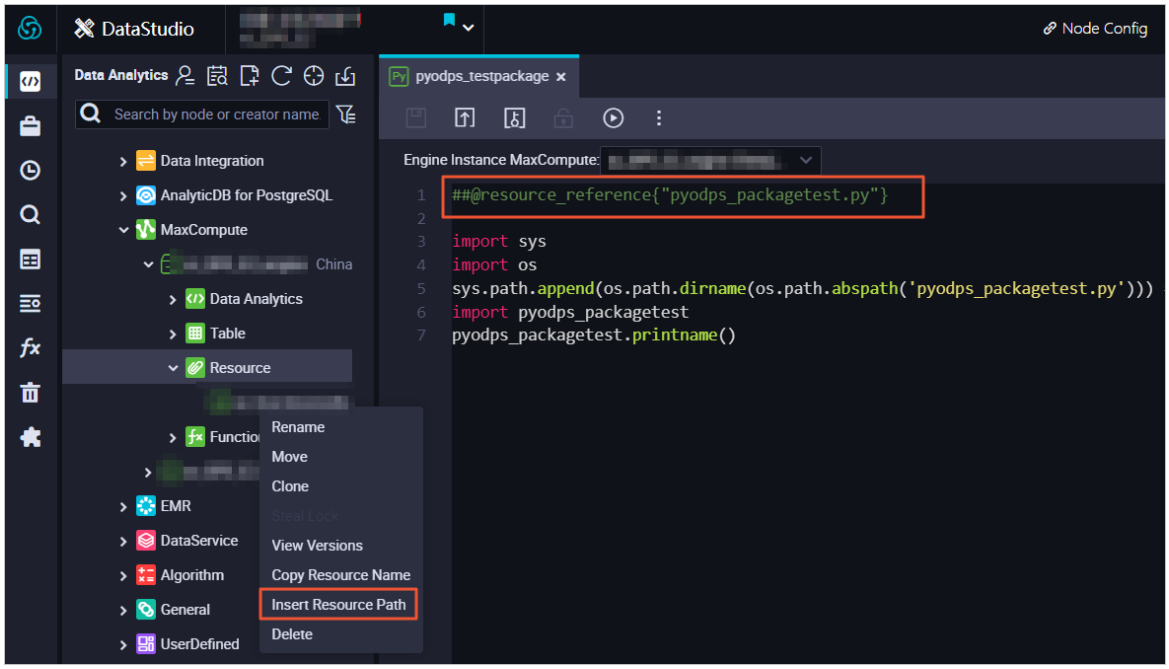
```
# import os
# print os.getcwd()
# print os.path.abspath('.')
# print os.path.abspath('..')
# print os.path.abspath(os.curdir)
def printname():
    print 'test2'
print 123
```

   v. Click the icon in the top toolbar.
3. Create a PyODPS 2 node.
   i. In the **Business Flow** section, find the workflow in which you want to create a PyODPS 2 node, right-click **MaxCompute**, and then choose **Create > PyODPS 2**.
   ii. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters. In this example, the Node Name parameter is set to pyodps_testpackage.

      > **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

   iii. Click **Commit**.
4. Open the configuration tab of the newly created PyODPS 2 node. Then, right-click the name of the Python resource in the Resource folder of your workflow and select **Insert Resource Path**.

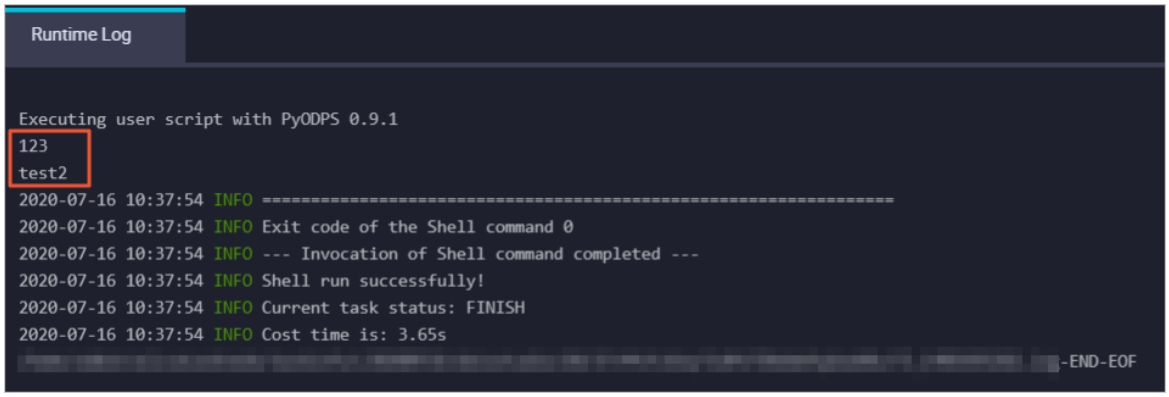
   After the resource is referenced, the `##@resource_reference{"pyodps_packagetest.py"}` statement is automatically written in the code editor of the PyODPS 2 node.



5. Enter the code that is used to reference the common Python script in the code editor of the PyODPS 2 node. In this example, the following code is used:

```
##@resource_reference{"pyodps_packagetest.py"} # This statement is required to referenc
e the created Python resource.
import sys
import os
sys.path.append(os.path.dirname(os.path.abspath('pyodps_packagetest.py'))) # Import the
resource to the workspace.
import pyodps_packagetest # Reference the resource. You must delete the .py suffix in t
he resource name.
pyodps_packagetest.printname() # Call the method.
```

6. Click the icon in the top toolbar and view the results on the **Runtime Log** tab in the lower part of the configuration tab.



Reference a third-party open source package

Before you reference a third-party open source package, you must use pip to install the package and make sure that the following requirements are met:

- An exclusive resource group for scheduling is available. For more information, see Create an exclusive

resource group for scheduling.

- The third-party open source package is installed in O&M Assistant of the exclusive resource group for scheduling. For more information, see O&M Assistant. PyODPS nodes include PyODPS 2 nodes and PyODPS 3 nodes.
  - If you want to use a PyODPS 2 node to reference the third-party open source package, run the following command to install the package:

    ```
    pip install <Package that you want to reference> -i https://pypi.tuna.tsinghua.edu.cn/simple
    ```

    If you are prompted to upgrade pip after you run the preceding command, run the following command:

    ```
    pip install --upgrade pip -i https://pypi.tuna.tsinghua.edu.cn/simple
    ```

  - If you want to use a PyODPS 3 node to reference the third-party open source package, run the following command to install the package:

    ```
    /home/tops/bin/pip3 install <Package that you want to reference> -i https://pypi.tuna.tsinghua.edu.cn/simple
    ```

    After the package is installed, run the `import` command to import the package. For example, use O&M Assistant to run the `pip3 -install oss2` command to install the package oss2. Then, run the `import oss2` command in the PyODPS 3 node to import and reference oss2.

    If you are prompted to upgrade pip after you run the preceding commands, run the following command:

    ```
    /home/tops/bin/pip3 install --upgrade pip -i https://pypi.tuna.tsinghua.edu.cn/simple
    ```

    If the following error occurs when you use the PyODPS 3 node, submit a ticket to apply for permissions.

    ```
    "/home/admin/usertools/tools/cmd-0.sh:Line 3: /home/tops/bin/python3: The file or directory does not exist."
    ```

2.4. Run nodes at a specific time by using branch nodes

This topic describes how to run nodes at a specific time by using branch nodes.

Background information

You cannot schedule a node to run on the last day of each month by using a cron expression. After branch nodes are supported, you can create a branch node and use the switch-case model to schedule a node to run on the last day of each month.

Branch nodes and other control nodes

On the **DataStudio** page, you can view the control nodes that are supported by DataWorks of the current edition, including assignment nodes, branch nodes, and MERGE nodes.

Features of each control node:

- **Assignment node**: You can pass the output of an assignment node to its child nodes. For more information, see Configure an assignment node.

  An assignment node reuses the node context feature for configuring dependencies. In addition to constants and variables, an assignment node supports custom output in the node context. For more information, see Configure context-based parameters. DataWorks captures or displays the query result of an assignment node and uses the result as the value of the outputs parameter in the node context for child nodes to reference.

- **Branch node**: A branch node determines which child nodes are run.

  A branch node reuses the feature of DataWorks for configuring input and output in dependencies. For more information, see Configure same-cycle scheduling dependencies.

  For common nodes, the output is only a globally unique string. To configure a node as the child node of a common node, you can specify the globally unique string of the common node as the input of the child node.

  However, for a branch node, you can select the output that is associated with a condition as the output of the branch node when you configure child nodes. When nodes become child nodes of the branch node, the nodes are also associated with the condition:

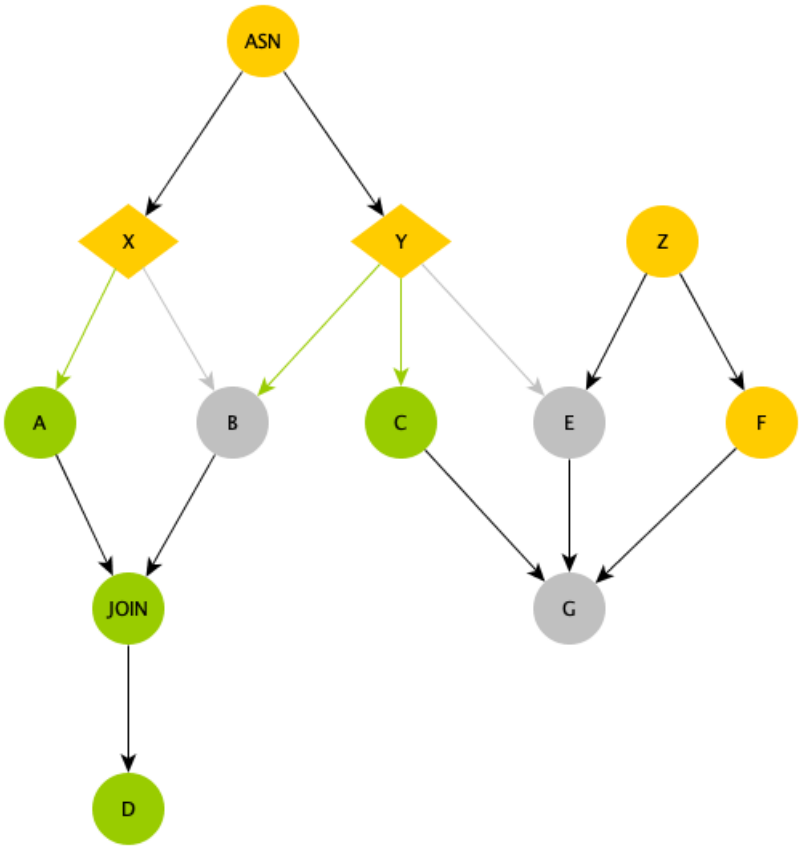
  - If the condition is met, the child nodes that depend on the output associated with the condition are run.
  - Other child nodes that do not depend on the output associated with the condition are dry run.

- **MERGE node**: A MERGE node is scheduled to run no matter whether its upstream nodes are run.

  On a branch that is not selected by a branch node, DataWorks sets instances of all nodes on the branch as dry-run instances. An instance is dry run when one of its upstream instances is a dry-run instance.

  DataWorks allows you to use a MERGE node to prevent the dry-run property from being passed downstream forever. A MERGE node is set as successful no matter how many upstream nodes are dry run, and the descendant nodes are not dry run. For more information, see Configure a merge node.

The following figure shows the logical relationships in a dependency tree that contains branch nodes.



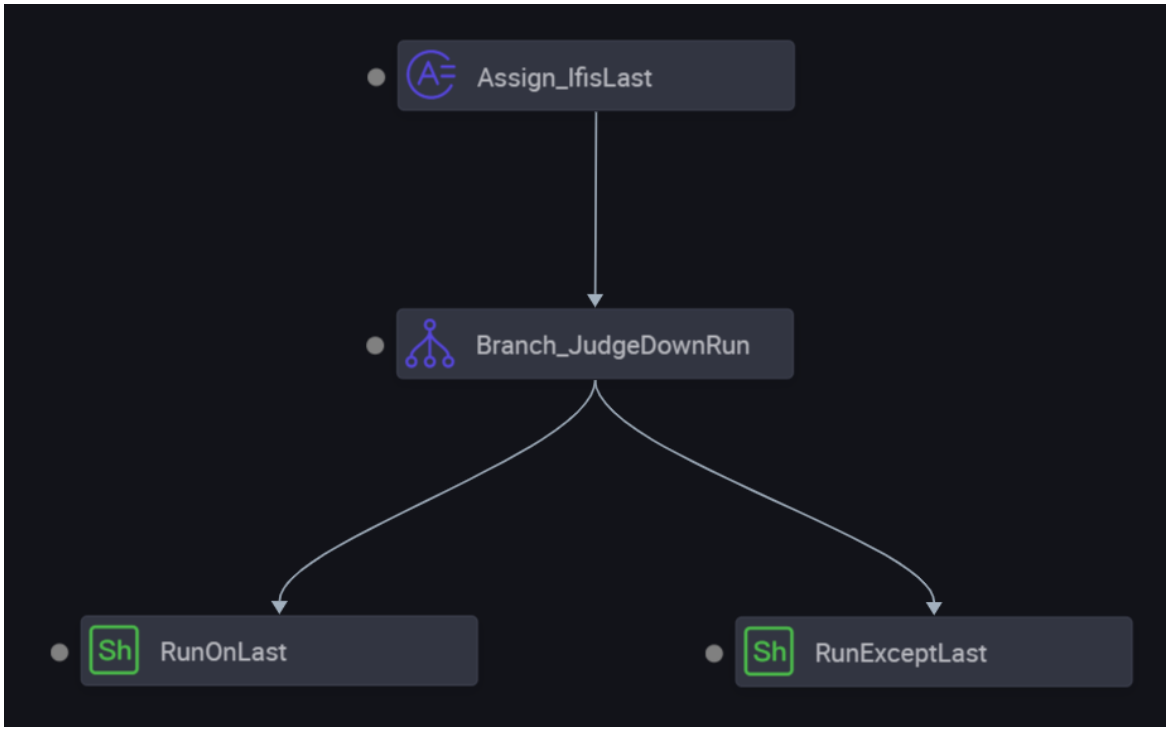
- ASN: an assignment node, which is used for computing in complex scenarios and allows you to select conditions for branch nodes.
- X and Y: the branch nodes, which are the child nodes of the assignment node ASN and select branches based on the output of the assignment node. Node X selects the left branch, and Node Y selects the left two branches, as marked by the green lines in the following figure.
  - Nodes A and C are run because they are selected by Nodes X and Y.
  - Node B is selected by Node Y. However, Node B is dry run because it is not selected by Node X.
  - Node E is not selected by Node Y. Therefore, Node E is also dry run even though it has a common node Z as a parent node.
  - Node G is also dry run because its parent node E is dry run even if Nodes C and F are run.
  - When will the dry-run property not be passed downstream?

    The JOIN node is a MERGE node. Its special feature is to prevent the dry-run property from being passed. Node D is a child node of the JOIN node. Therefore, Node D does not inherit the dry-run property from Node B and can be run.

You can use branch nodes together with other control nodes to schedule a node to run only on the last day of each month.

Use a branch node

1. Define node dependencies.



i. The root node, which is an assignment node, determines whether the current day is the last day of the month based on the scheduled time parameter SKYNET_CYCTIME. If the current day is the last day, 1 is returned. Otherwise, 0 is returned. The output is captured by DataWorks and passed to the child node.

ii. The branch node defines branches based on the output of the assignment node.

iii. Two Shell nodes are associated with the branch node to run different branch logic.

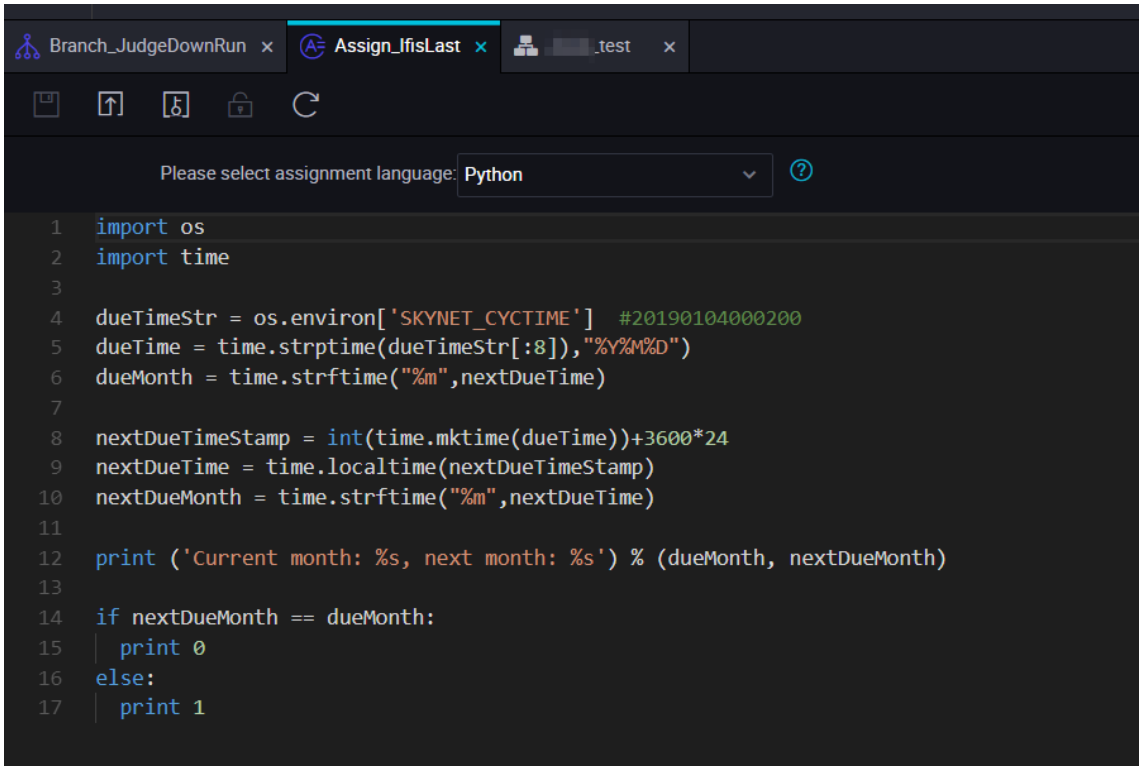
2. Define the assignment node.

   A created assignment node has a built-in outputs parameter. An assignment node can be written in the SQL, SHELL, or Python language.

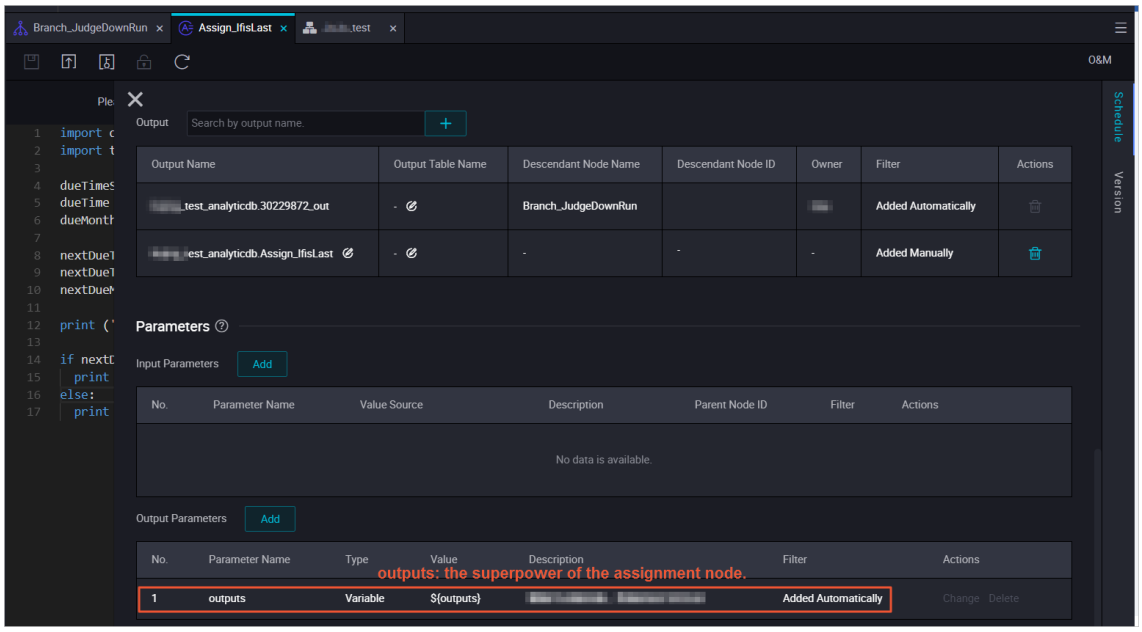
   - For the SQL type, DataWorks captures the last SELECT statement as the value of the outputs parameter.
   - For the SHELL and Python types, DataWorks captures the standard output in the last line as the value of the outputs parameter.

   In this topic, the code of the assignment node is written in Python. The following figures show the scheduling properties and code of the assignment node.

   - Code



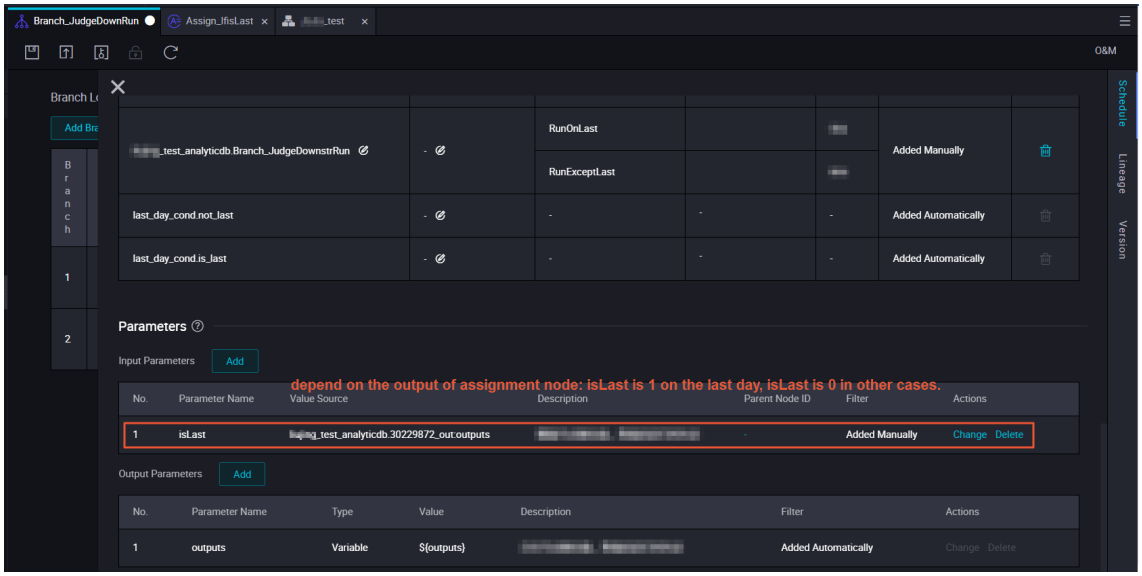
- Configuration of scheduling properties



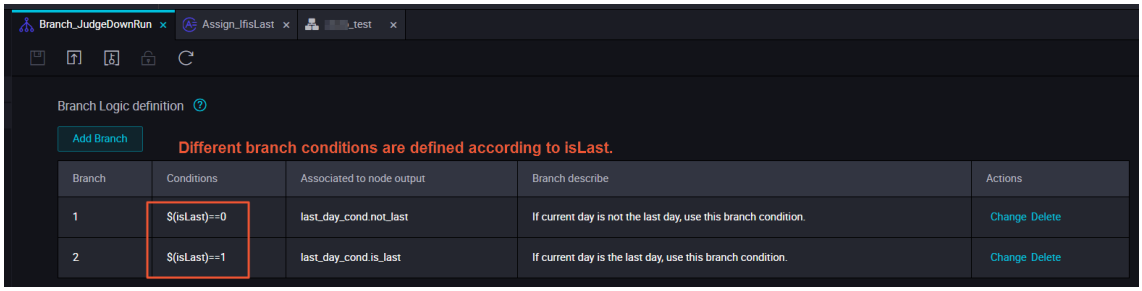
3. Define branches.

   You can define conditions for branch nodes by using Python expressions. Each condition is associated with an output. If a condition is met, the child node that depends on the output associated with the condition is run, and other child nodes are dry run.

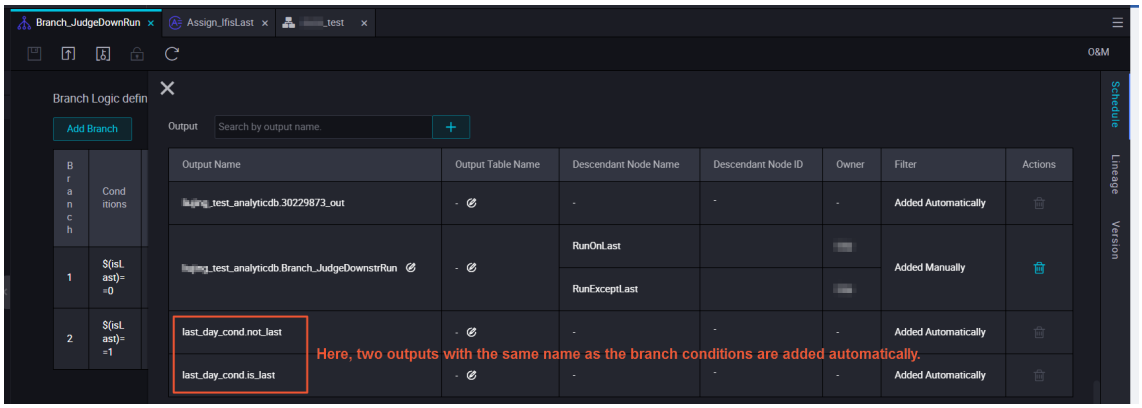
   - Configuration of scheduling properties



- Configuration of branches



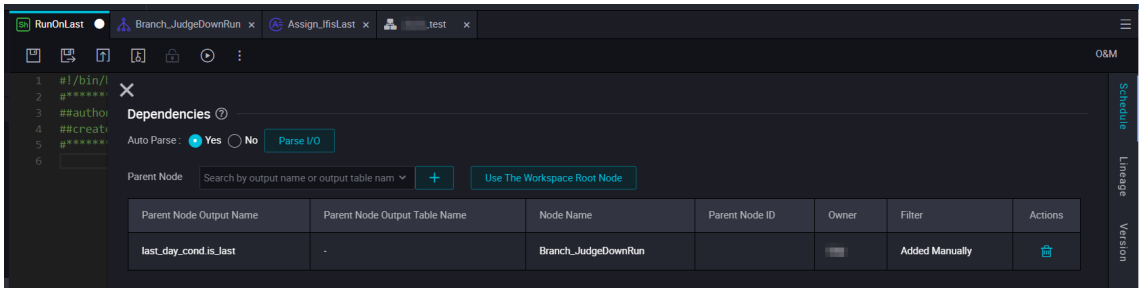
- Outputs that are associated with the conditions and generated in the scheduling configuration



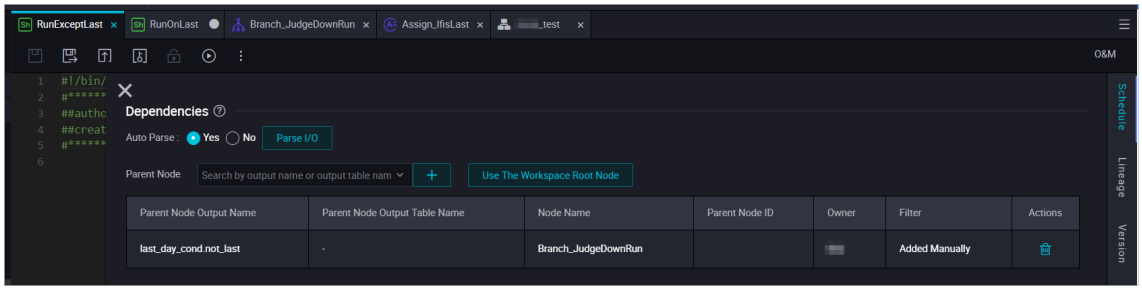
4. Associate task execution nodes with different branches.

   The branch node has three outputs. You can select one output as the input. Select an output with caution because the outputs are associated with conditions.

   - Dependency for the node that is to be run on the last day of each month



- Dependency for the node that is to be run on other days of each month

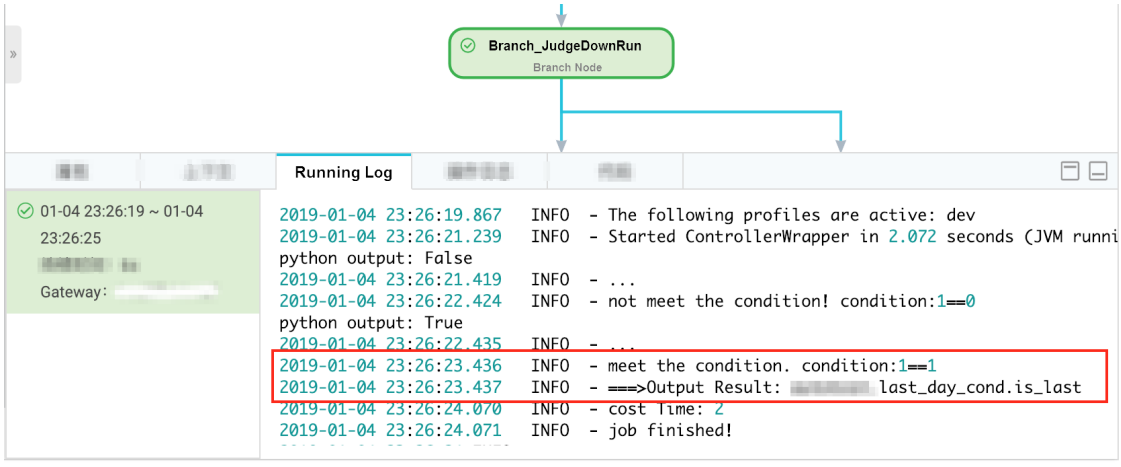


5. Verify the result.

   After you perform the preceding steps, you can commit and deploy the nodes. After you deploy the nodes, you can generate retroactive data to test the running effect. Set the data timestamps to December 30, 2018 and December 31, 2018. This means that the scheduled time is December 31, 2018 and January 1, 2019. The first batch of retroactive data triggers the logic for the last day. The second batch of retroactive data triggers the logic for days except the last day. The two have the following differences:

   The data timestamp is December 30, 2018. In other words, the scheduled time is December 31, 2018.

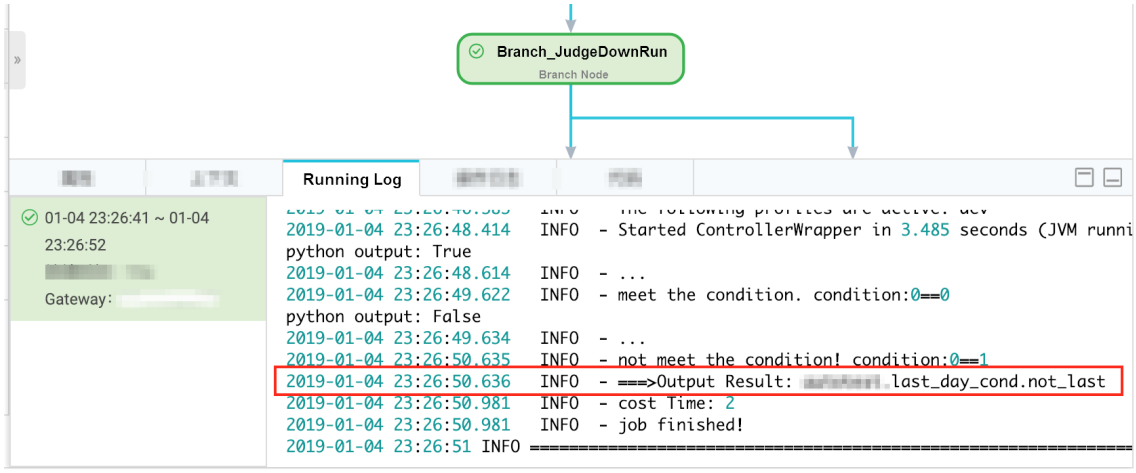
   - Branch selection result of the branch node



   - The node to be run on the last day is run.
   - The node to be run on days except the last day is dry run.

   The data timestamp is December 31, 2018. In other words, the scheduled time is January 1, 2019.

   - Branch selection result of the branch node



- The node to be run on the last day is dry run.
- The node to be run on days except the last day is run.

Summary

Take note of the following instructions when you use branch nodes:

- DataWorks captures the last SELECT statement or the last line of the standard output stream of an assignment node as the output of the assignment node. The output is referenced by child nodes.
- Each output of a branch node is associated with a condition. When you associate a child node with a branch node, you must understand the condition that is associated with each output before the selection.
- If a branch is not selected, the child node on the branch is dry run. The dry-run property is always passed downstream until a MERGE node is found.

2.5. Connect DataV to DataWorks DataService Studio

This topic describes how to connect DataV to DataWorks DataService Studio. You can connect Hologres to DataWorks to create APIs in DataService Studio and call the APIs in DataV. Then, DataV presents the analysis results of MaxCompute data.

Background

MaxCompute is a fast and fully managed computing platform for large-scale data warehousing. It can process terabytes, petabytes, or even exabytes of data. As a wide variety of data collection methods emerge and huge amounts of data is accumulated, traditional software cannot meet data processing requirements. MaxCompute is used to store and compute masses of structured data. MaxCompute provides stable support for all offline business of data analytics for Alibaba Group over the years.

In the past, if you want to present the analysis results of large amounts of data in DataV, you need to create a workflow to import data from your system to a MySQL database. This process involves complex operations and requires high costs. DataWorks provides the Data Integration, DataStudio, and DataService Studio services for you to develop data. You can integrate these services with MaxCompute to build data warehouses for your enterprise with ease.

DataWorks DataService Studio allows you to use the codeless user interface (UI) to create APIs based on data tables without the need to write code. Then, you can call the APIs in DataV to present data analysis results on dashboards. This way, you can develop a data warehouse and present data in an efficient way.

Prerequisites

A data source is prepared. DataV is activated.

Create a connection

DataService Studio supports connections to various data sources:

- Relational databases: ApsaraDB RDS, Distributed Relational Database Service (DRDS), MySQL, PostgreSQL, Oracle, and SQL Server
- Analytic databases: AnalyticDB
- NoSQL databases: Tablestore and MongoDB

1. Log on to the DataWorks console. In the left-side navigation pane, click Workspaces. On the Workspaces page, find the workspace that you want to manage and click **DataService Studio** in the Actions column.
2. In the **Service Development** pane, move the pointer over the icon and select **New Data Source**. The **Data Source** page appears.
3. Click **Add data source** in the upper-right corner.
4. In the **Add data source** dialog box, click **Hologres** in the Big Data Storage section.

   This example describes how to create a connection to Hologres. This allows you to query data in MaxCompute in real time.
5. In the **Add Hologres data source** dialog box, set the parameters as required.

Parameter	Description
Data source type	In this example, a connection to Hologres is used. Only **Alibaba Cloud instance mode** is supported.
Data Source Name	The name of the connection. The name can contain letters, digits, and underscores (_), and must start with a letter.
Data source description	The description of the connection. The description can be up to 80 characters in length.
Environment	The environment in which the connection is used. Valid values: **Development** and **Production**. **Note** This parameter is displayed only when the workspace is in standard mode.
Instance ID	The ID of the Hologres instance from which you want to synchronize data. You can obtain the instance ID from the Hologres console.

Parameter	Description
Database Name	The name of the database in the Hologres instance.
AccessKey ID	You can obtain the AccessKey ID from the Security Management page.
AccessKey Secret	You can obtain the AccessKey secret from the Security Management page.

6. Click **Test connectivity**.
7. After the connection passes the connectivity test, click **Complete**.

Create an API

After you create a connection, go to the **DataService Studio** page. This example describes how to create an API by using the codeless UI.

1. Click the More icon in the upper-left corner and choose **All Products > DataService Studio**. The **DataService Studio** page appears.
2. In the Service Development pane, move the pointer over the icon and choose **New API > Generate API**.
3. In the **Generate API** dialog box, set the parameters as required. In this example, set the API mode parameter to Wizard Mode.

Parameter	Description
API Name	The name of the API. The name must be 4 to 50 characters in length, and can contain letters, digits, and underscores (_). It must start with a letter.
API Group	The API group to which the API belongs. An API group is a collection of APIs for a specific feature or scenario. An API group is the API management unit of API Gateway. Each API group corresponds to a specific API product in the Alibaba Cloud API Marketplace. To create an API group, move the pointer over the **Create** icon in the Service Development pane and select **New API Group**.
API Path	The path in which the API is stored, such as */user*.
Protocol	The protocol used by the API. Valid values: HTTP and HTTPS.
Request Method	The request method used by the API. Valid values: GET and POST.
Response Content Type	The return type of the API. The value is fixed to JSON.
Description	The description of the API.

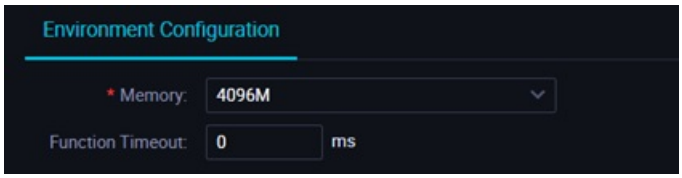
4. Click **OK**. The configuration tab of the API appears.

Set the API parameters

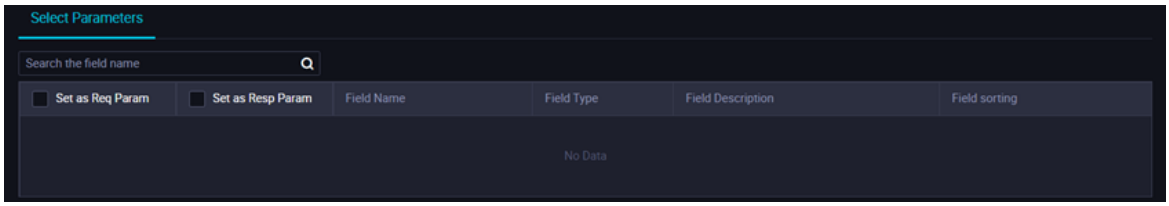
1. In the **Select Table** section, select Hologres from the **Datasource Type** drop-down list and set the **Datasource Name** and **Table Name** parameters as required.

   > **Note**
   > - Before you set API parameters, you must configure a connection in Data Integration. You can enter a table name in the Table Name field to search for the desired table.
   > - After you create the API, the table configuration tab automatically appears for you to select a table for the API.

2. In the **Environment Configuration** section, set the **Memory** and **Function Timeout** parameters as required.

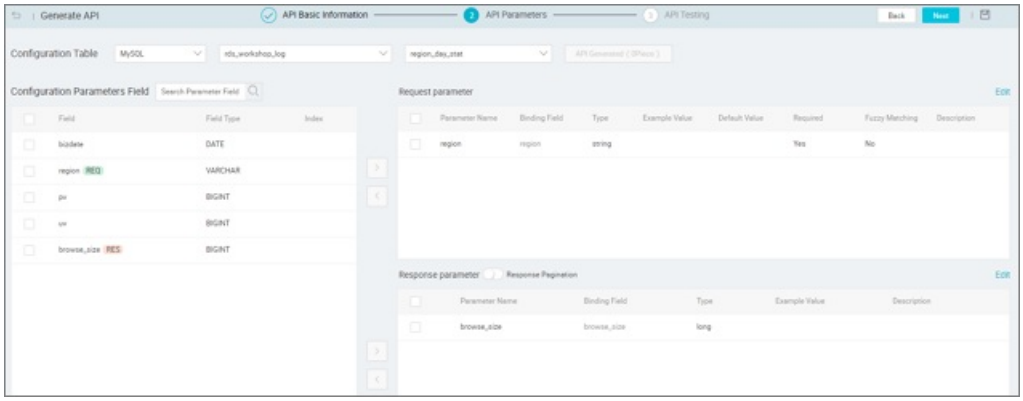
   

3. After you select a table in the Select Table section, all fields in the table appear in the **Select Parameters** section. Select the fields that need to be set as request parameters and the fields that need to be set as response parameters. Add them to the request parameter list and the response parameter list.

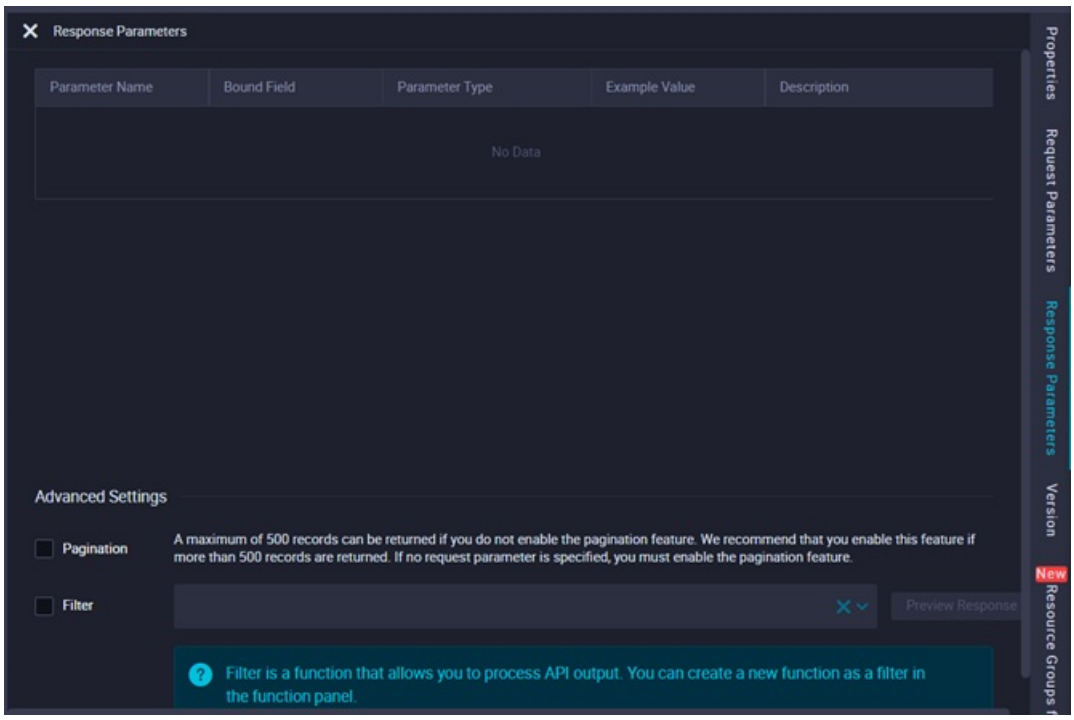
4. Edit request parameters.

   In the right-side navigation pane, click the **Request Parameters** tab. In the Request Parameters panel, set the **Parameter Name**, **Parameter Type**, **Operator**, **Required**, **Example Value**, **Default Value**, and **Description** parameters.

5. Edit response parameters.

In the right-side navigation pane, click the **Response Parameters** tab. In the Response Parameters panel, set the **Parameter Name**, **Parameter Type**, **Example Value**, and **Description** parameters. You can also select **Pagination** and **Filter** in the Advanced Settings section.

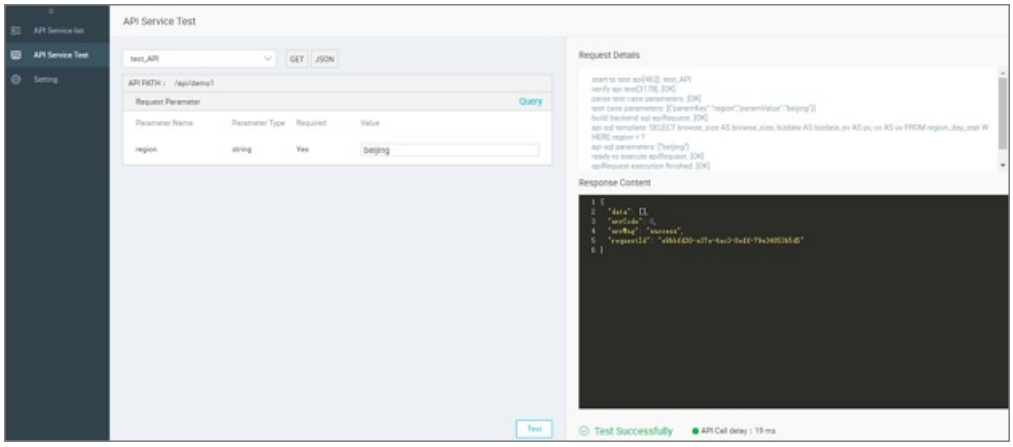


Select Pagination based on the actual requirements.

- If you do not select **Pagination**, the API returns a maximum of 2,000 records by default.
- If the API may return more than 2,000 records, we recommend that you select **Pagination**.

Test the API

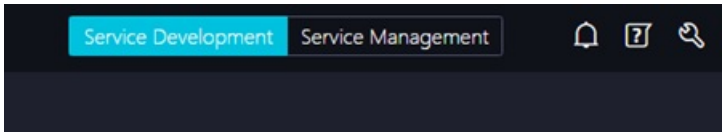
After you set the API parameters and save the settings, click **Test** in the upper-right corner. The Test APIs dialog box appears.



Set the parameters and click **Test** to send an API request. The request and response details appear on the right. If the API fails the test, check the error message, modify the API settings, and then test the API again.

Publish the API

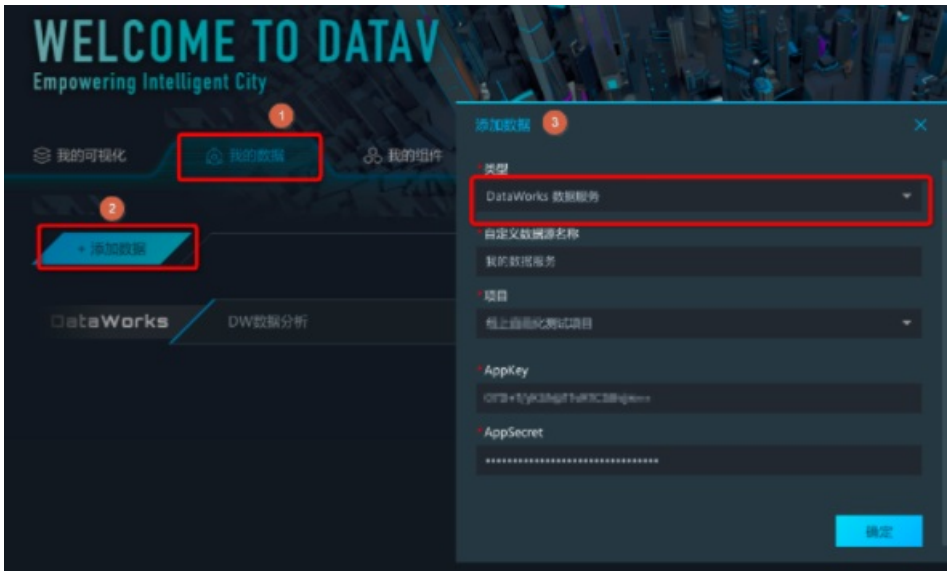
1. After the API passes the test, return to the **DataService Studio** page.
2. Click **Publish** in the upper-right corner. The API is published.
3. After the API is published, click **Service Management** in the upper-right corner to view the API details.



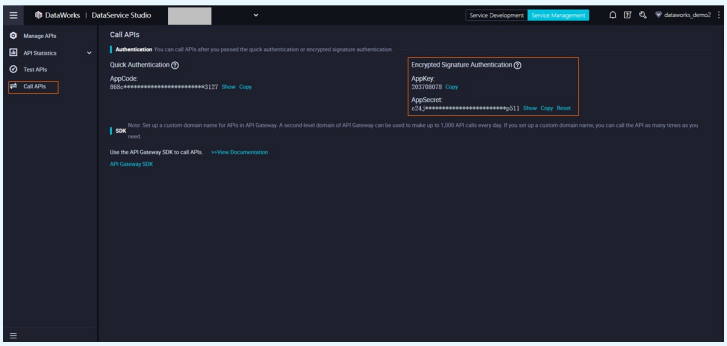
If you want to call the API, click **Service Management**. On the page that appears, click **Call APIs** in the left-side navigation pane. DataService Studio supports quick authentication based on the AppCode and encrypted signature authentication based on the AppKey and AppSecret. The following sections describe how to call DataService Studio APIs in DataV.

Add DataService Studio as a data source

1. Log on to the DataV console.
2. In the top navigation bar, click **Data Sources** and then **Add Source**.
3. In the **Add Data Source** dialog box, set the parameters as required.



Parameter	Description
Type	The type of the data source.
Name	The name of the data source. Enter a custom name.
Project	The DataWorks workspace from which the required data comes.

Parameter	Description
AppKey and AppSecret	The AppKey and AppSecret of the account that has the permissions to access a DataWorks workspace. **Note** To obtain the AppKey and AppSecret, log on to the DataWorks console, go to the DataService Studio page, and then click **Service Management** in the upper-right corner. On the page that appears, click **Call APIs** in the left-side navigation pane. 

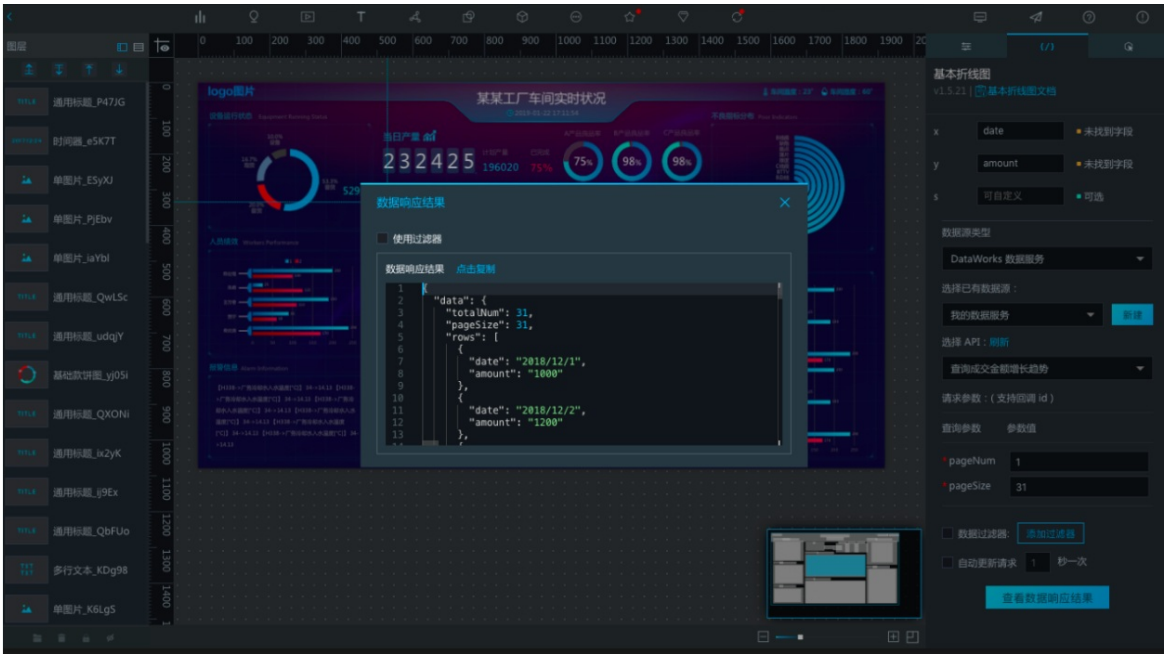
Call an API of DataService Studio in DataV

1. Log on to the DataV console. In the top navigation bar, click **Projects**. On the page that appears, click **Create Project**.
2. On the page that appears, select a template and click **Create Project**. In this example, select the **Smart Factory** template.

Widgets in the template contain static data. This example describes how to call the API to query the growth trend of the transaction amount and present the query results in the Basic Line Chart widget.

3. Select the Basic Line Chart widget and go to the data panel. In the data panel, set the **Data Source Type** parameter to **DataWorks**.
4. Select the added data source and created API and set the query parameters. In this example, set the **Page Size** parameter to 31 to query data of one month.

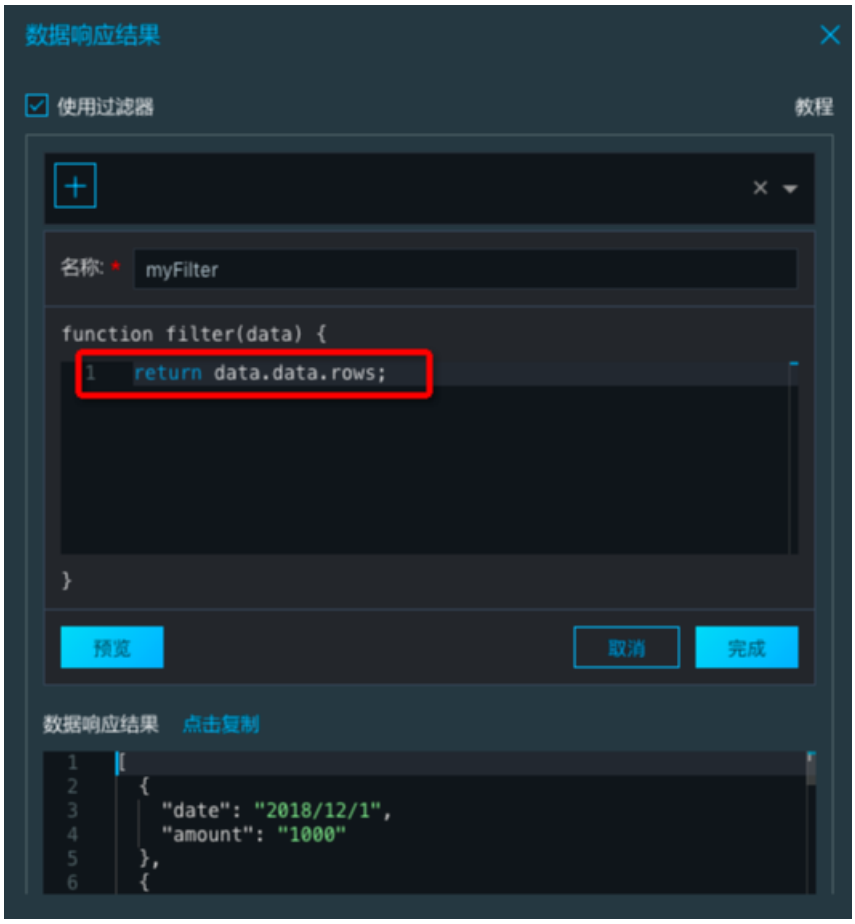
5. Click **Preview Data Response** to view the query results of the API.
6. Set the x field to **date** and the y field to **amount**.



The preceding figure shows that the x and y fields match no data. This is because DataV has requirements for the data format and cannot identify fields with a deep structure. You need to add a data filter to filter out unnecessary fields.

7. Select **Data Filter** and click the **plus (+)** icon on the right to add a data filter. You can also enter JavaScript code in the code editor to filter and process the data analysis results. The data parameter of the filter specifies the JSON objects returned by the API.

In this example, enter the code `return data.data.rows;` to filter out fields except the rows array returned by the API. Click Preview to preview the filtered data in the lower part, and click **Complete**.



After you add a data filter, DataV can match data based on the specified fields.

The date format returned by the API is different from the default date format of the widget. Therefore, DataV does not correctly display the line chart. You must set the date format for the X axis of the line chart before DataV can correctly display the line chart.

8. Go to the configuration panel. Click **Axis Label** under **X Axis**. Set the data type to date and the data format to a value in the YYYY/MM/DD format, such as 2016/01/01. Then, DataV correctly displays the line chart.

After the preceding steps are performed, DataV can call the API created in DataService Studio based on a MaxCompute table to obtain and display the data analysis results. The following figure shows the data display effect.

Usage notes

After you connect DataV to DataWorks DataService Studio by adding DataService Studio as a data source of DataV, you can directly call an API of DataService Studio in DataV. In this case, you do not need to enter the API URL. You do not need to enter the AppKey and AppSecret for calling each API. In addition, you can set API parameters by using a table. This is convenient, secure, and reliable.

DataService Studio allows you to directly present the data analysis results processed by MaxCompute in DataV. This seamlessly links data analytics and data presentation.

When you use DataService Studio and DataV together, take note of the following items:

- APIs created on the codeless UI in DataWorks DataService Studio can only query data in a single table based on simple criteria. APIs created in the code editor allow you to use SQL statements or functions to query data in multiple tables based on complex criteria. You can create APIs by using the codeless UI or code editor based on your needs.
- If you want an API to query data within milliseconds, we recommend that you use a relational database, a NoSQL database, or an AnalyticDB database as the data source.
- DataV widgets only support arrays, but DataService Studio APIs return complete JSON strings that contain error codes. Therefore, you must use filters to process results returned by the APIs. You can add filters in DataV or when you configure APIs in DataService Studio.

  If pagination is disabled for an API, add a filter to filter out fields except the data array returned by the API. If pagination is enabled for an API, add a filter to filter out fields except the data.rows array returned by the API.
- If you want to present a data analysis result in multiple types of charts, such as in both a line chart and a column chart, DataV treats the data of each type as an object and distinguishes the types by field. In this case, you need to use a filter for format conversion.

2.6. Use a PyODPS node to send emails

This topic describes how to use a PyODPS node that runs on an exclusive resource group to send emails.

Context

Different from a Python script, a PyODPS node in DataWorks can interact with MaxCompute for data analytics and processing. DataWorks cannot automatically send emails as scheduled. You can create a PyODPS node and run it on an exclusive resource group to read data from MaxCompute and then send emails.

> **Note** TCP port 25 is the default email service port. For security purposes, port 25 is blocked on Elastic Compute Service (ECS) instances. Therefore, you cannot use port 25 on exclusive resource groups. We recommend that you use port 465 to send emails.

If you run a PyODPS node on an exclusive resource group to send emails, the users of the exclusive resource group cannot log on to the ECS instances in the group. As a result, the users cannot install other third-party Python modules to implement additional features.

Procedure

1. Add an exclusive resource group.
   i. Log on to the DataWorks console.
   ii. In the left-side navigation pane, click **Resource Groups**.

iii. On the **Resource Groups** page, click Create Resource Group for Scheduling on the **Exclusive Resource Groups** tab.

iv. In the **Create a dedicated resource group** panel, set the parameters as required. For more information, see Create and use an exclusive resource group for scheduling.

> **Note** Add an exclusive resource group that is in the same region as the DataWorks workspace.

v. Click **OK**.

2. Associate the exclusive resource group with the desired workspace.

i. On the Exclusive Resource Groups tab, find the desired exclusive resource group and click **Change Workspace** in the Actions column.

ii. In the **Modify home workspace** dialog box, find the workspace with which you want to associate the exclusive resource group.

iii. Click **Bind** in the Actions column.

3. Go to the **DataStudio** page.

i. Click **Workspaces** in the left-side navigation pane.

ii. Select the region where the desired workspace resides in the upper-left corner.

iii. In the workspace list, find the desired workspace and click **Data Analytics** in the Actions column. The DataStudio page appears.

4. Create a PyODPS 2 node.

i. On the DataStudio page, move the pointer over the icon and choose **MaxCompute > PyODPS 2**.

You can also click the desired workflow, right-click **MaxCompute**, and then choose **Create > PyODPS 2**.

ii. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

> **Note** The node name must be 1 to 128 characters in length and can contain letters, digits, underscores (_), and periods (.).

iii. Click **Commit**.

iv. On the configuration tab of the PyODPS 2 node, enter the following code to send emails by using Simple Mail Transfer Protocol (SMTP):

```
import smtplib
from email.mime.text import MIMEText
from odps import ODPS
mail_host = '<yourHost>' // The address of the email server.
mail_username = '<yourUserName>' // The username that is used to log on to the mail
box of the sender.
mail_password = '<yourPassWord>'  // The password that is used to log on to the mai
lbox of the sender.
mail_sender = '<senderAddress>' // The email address of the sender.
mail_receivers = ['<receiverAddress>']  // The email address of the recipient.
mail_content=""        // The email content to be sent.
o=ODPS('access_key','access_secretkey','default_project_name',endpoint='maxcompute_
service_endpoint')
with o.execute_sql('query_sql').open_reader() as reader:
           for record in reader:
                   mail_content+=str(record['column_name'])+' '+record['column_name
']+'\n'
message = MIMEText(mail_content,'plain','utf-8')
message['Subject'] = 'mail test'
message['From'] = mail_sender
message['To'] = mail_receivers[0]
try:
           smtpObj = smtplib.SMTP_SSL(mail_host+':465')
           smtpObj.login(mail_username,mail_password)
           smtpObj.sendmail(
               mail_sender,mail_receivers,message.as_string())
           smtpObj.quit()
           print('mail send success')
except smtplib.SMTPException as e:
           print('mail send error',e)
```

Alternatively, you can enter the following code to send emails:

```
import smtplib
from email.mime.text import MIMEText
from odps import ODPS
mail_host = 'smtp.office365.com'  // The address of the email server.
mail_username = 'xxxx' // The username that is used to log on to the mailbox of the
sender.
mail_password = 'xxx'  // The password that is used to log on to the mailbox of the
sender.
mail_sender = 'xxx' // The email address of the sender.
mail_receivers = ['xxx']  // The email address of the recipient.
mail_content=""        // The email content to be sent.
o=ODPS('access_key','access_secretkey','default_project_name',endpoint='maxcompute_
service_endpoint')
with o.execute_sql('query_sql').open_reader() as reader:
    for record in reader:
                mail_content+=str(record['column_name'])+' '+record['column_name']+
'\n'
message = MIMEText(mail_content,'plain','utf-8')
message['Subject'] = 'mail test'
message['From'] = mail_sender
message['To'] = mail_receivers[0]
try:
           smtpObj = smtplib.SMTP()
           smtpObj.connect(mail_host,587)
           smtpObj.ehlo()
           smtpObj.starttls()
           smtpObj.login(mail_username,mail_password)
           smtpObj.sendmail(
               mail_sender,mail_receivers,message.as_string())
           smtpObj.quit()
           print('mail send success')
except smtplib.SMTPException as e:
          print('mail send error',e)
```

> **Note** When you use the PyODPS 2 node to send mails, the PyODPS 2 node stores the data it reads in a temporary file first and sends the data by email. The number of data records in the email that you want to send is unlimited.

v. Click the icon in the top toolbar.

5. Commit the node.

> **Notice** Before you commit the node, you must click the **Properties** tab in the right-side navigation pane and set the **Rerun** and **Parent Nodes** parameters.

i. Click the icon in the top toolbar.

ii. In the **Commit Node** dialog box, enter your comments in the **Change description** field.

iii. Click **OK**.

In a workspace in standard mode, you must click **Deploy** in the upper-right corner after you commit the node. For more information, see Deploy nodes.

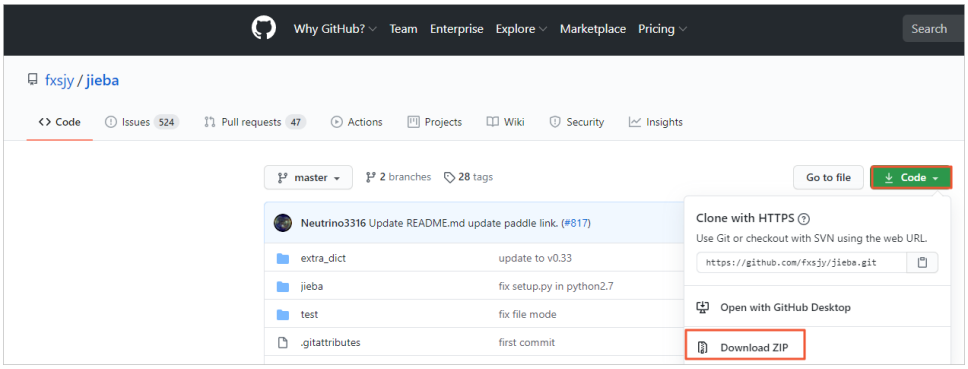
6. Change the resource group that is used to run the PyODPS 2 node.
   i. On the configuration tab of the PyODPS 2 node, click **Operation Center** in the upper-right corner.
   ii. In the left-side navigation pane of the Operation Center page, choose **Cycle Task Maintenance > Cycle Task**.
   iii. On the page that appears, click the rightwards arrow in the middle to show the node list.
   iv. Find the desired node and choose **More > Modify Scheduling Resource Group** in the Actions column.
   v. In the **Modify Scheduling Resource Group** dialog box, select the desired resource group from the **New Resource Group** drop-down list.
   vi. Click **OK**.
7. Test the PyODPS 2 node. For more information, see View auto triggered nodes.

2.7. Use a PyODPS node to segment Chinese text based on Jieba

This topic describes how to use a PyODPS node in DataWorks to segment Chinese text based on the open source segmentation tool Jieba, and write the segmented words and phrases to a new table. This topic also describes how to use closure functions to segment Chinese text based on a custom dictionary.

Prerequisites

- A DataWorks workspace is created. In this example, a workspace in **basic mode** is used. The workspace is associated with multiple MaxCompute compute engines. For more information, see Create a workspace.
- The open source Jieba package is downloaded from GitHub.



Context

PyODPS nodes integrate MaxCompute SDK for Python. You can directly edit Python code and use MaxCompute SDK for Python in PyODPS nodes of DataWorks. For more information about PyODPS nodes, see Create a PyODPS 2 node.

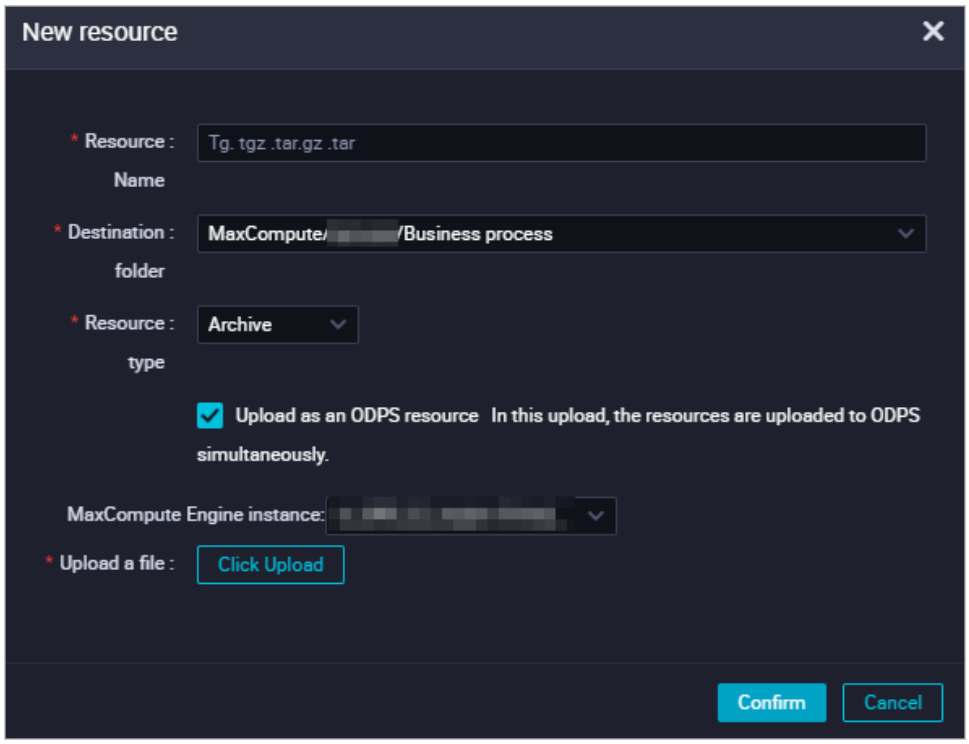
> **Notice** Sample code in this topic is for reference only. We recommend that you do not use the code in your production environment.

Procedure

1. Create a workflow.
   i. Log on to the DataWorks console.
   ii. In the left-side navigation pane, click **Workspaces**.
   iii. Select the region where the required workspace resides, find the workspace, and then click **Data Analytics** in the Actions column.
   iv. Move the pointer over the icon and click **Workflow**.
   v. In the **Create Workflow** dialog box, set the **Workflow Name** and **Description** parameters.

   > **Notice** The Workflow Name parameter must be 1 to 128 characters in length, and can contain letters, digits, underscores (_), and periods (.).

   vi. Click **Create**.
2. Upload the jieba-master.zip package.
   i. Click the created workflow, right-click **Resource** under **MaxCompute**, and then choose **Create > Archive**.
   ii. In the **Create Resource** dialog box, set the parameters.



Parameter	Description

Parameter	Description
Resource Name	The name of the resource. It can be different from the name of the uploaded file. However, the resource name must comply with the following conventions: ■ The name can contain only letters, digits, periods (.), underscores (_), and hyphens (-). ■ If Archive is selected from the Resource Type drop-down list, the extension of the resource name must be the same as that of the file name. The extension can be .zip, .tgz, .tar.gz, or .tar.
Location	The folder for storing the resource. The default value is the path of the current folder. You can modify the path based on your business requirements.
Resource Type	Select **Archive** from the Resource Type drop-down list. **Note** If the resource package has been uploaded to the MaxCompute client, clear **Upload to MaxCompute**. Otherwise, an error occurs during the upload process.
Engine Instance	Select the compute engine where the resource resides from the drop-down list. If only one instance is bound to your workspace, this parameter is not displayed.
Upload	Click **Upload**, select the downloaded file jieb-master.zip from the on-premises machine, and then click **Open**.

iii. In the **Create Resource** dialog box, click **Create**.

iv. Click the icon in the toolbar. In the **Commit Node** dialog box, set the **Change description** parameter.

v. Click **OK**.

3. Create a table for storing test data.

i. Click the created workflow, right-click **Table** under **MaxCompute**, and then select **Create Table**.

ii. In the **Create Table** dialog box, set **Table Name** to an appropriate value, such as jieba_test, and select **MaxCompute** from the Please select an Engine type drop-down list.

iii. Click **Create**.

iv. Click **DDL Statement** and enter the following DDL statement for creating a table.

The table in this example contains two columns. You can segment text in one column during data development.

```
CREATE TABLE jieba_test (
    `chinese` string,
    `content` string
);
```

v. Click **Generate Table Schema**.

vi. In the **Confirm** message, click **OK**.

vii. In the **General** section, set the **Display Name** parameter for the table.

viii. Click **Commit to Production Environment**.

ix. In the **Commit to Production Environment** dialog box, select **I am aware of the risk and confirm the commissions** and click **OK**.

4. Use the same method to create a table for storing the test result.

In this example, only text in the chinese column of the test data is segmented. Therefore, the result table contains only one column. The following DDL statement is used to create a table for storing the test result:

```
CREATE TABLE jieba_result (
    `chinese` string
) ;
```

5. Download the test data for text segmentation.

6. Upload test data.

i. Click the icon on the **DataStudio** page.

ii. In the **Data Import Wizard** dialog box, enter the name of the test table jieba_test to which data needs to be imported, select the table, and then click **Next**.

iii. Click **Browse**, upload the *jieba_test.csv* file from the on-premises machine, and then click **Next**.

iv. Select **By Name** and click **Import Data**.

7. Create a PyODPS 2 node.

i. Click the required workflow, right-click **Data Analytics** under **MaxCompute**, and then choose **Create > PyODPS 2**.

ii. In the **Create Node** dialog box, set **Node Name** to an appropriate value, such as word_split, and set **Location**.

> **Note** The node name must be 1 to 128 characters in length, and can contain letters, digits, underscores (_), and periods (.).

iii. Click **Commit**.

iv. On the configuration tab of the node, select the **MaxCompute compute engine** and enter the following PyODPS code:

```
def test(input_var):
    import jieba
    import sys
    reload(sys)
    sys.setdefaultencoding('utf-8')
    result=jieba.cut(input_var, cut_all=False)
    return "/ ".join(result)
hints = {
    'odps.isolation.session.enable': True
}
libraries =['jieba-master.zip']  # Reference the jieba-master.zip package.
iris = o.get_table('jieba_test').to_df()  # Reference data in the jieba_test table.

example = iris.chinese.map(test).execute(hints=hints, libraries=libraries)
print(example)  # Display the text segmentation result, which is of the MAP type.
abci=list(example)   # Convert the text segmentation result to the LIST type.
i = 0
for i in range(i,len(abci)):
    pq=str(abci[i])
    o.write_table('jieba_result',[pq])  # Write the data records to the jieba_resul
t table one by one.
    i+=1
else:
    print("done")
```

v. Click the icon in the toolbar to save the code.

vi. Click the icon in the toolbar. In the **Arguments** dialog box, select a resource group from the **Resource Group** drop-down list.

vii. Click **OK** to test the PyODPS 2 node.

viii. View the running result of the Jieba segmentation program on the **Runtime Log** tab in the lower part of the page.

8. Create and run an ODPS SQL node.

i. Click the required workflow, right-click **Data Analytics** under **MaxCompute**, and then choose **Create > ODPS SQL**.

ii. In the **Create Node** dialog box, set the **Node Name** and **Location** parameters.

> **Note** The node name must be 1 to 128 characters in length, and can contain letters, digits, underscores (_), and periods (.).

iii. Click **Commit**.

iv. On the configuration tab of the node, enter the SQL statement `select * from jieba_result ;` .

v. Click the icon in the toolbar to save the SELECT statement.

vi. Click the icon in the toolbar. In the **Arguments** dialog box, select a resource group from the **Resource Group** drop-down list.

vii. Click **OK** to execute the SELECT statement.

viii. In the **Expense Estimate** dialog box, check the estimated cost and click **Run**.

ix. View the running result on the **Runtime Log** tab in the lower part of the page.

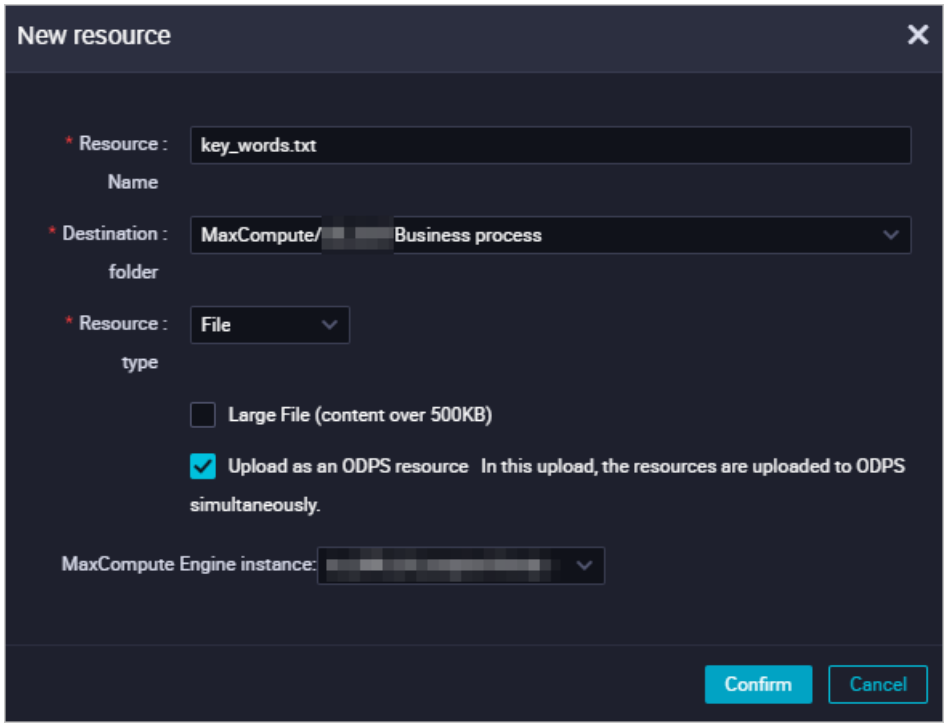
9. If the dictionary of the Jieba tool does not meet your requirements, use a custom dictionary.

   You can use a PyODPS user-defined function (UDF) to read resources uploaded to MaxCompute. The resources can be tables or files. In this case, you must write the UDF as a closure function or callable class. If you need to reference complex UDFs, you can create a MaxCompute function in DataWorks. For more information, see Create a MaxCompute function.

   In this topic, a closure function is used to reference the custom dictionary file key_words.txt that is uploaded to MaxCompute.

   i. Click the required workflow, right-click **Resource** under **MaxCompute**, and then choose **Create > File**.

ii. In the **Create Resource** dialog box, set the parameters.



Parameter	Description
Resource Name	The name of the resource. It can contain letters, digits, periods (.), underscores (_), and hyphens (-).
Location	The folder for storing the resource. The default value is the path of the current folder. You can modify the path based on your business requirements.
Resource Type	Select **File** from the Resource Type drop-down list. Make sure that **Upload to MaxCompute** is selected when you create the resource file used in this topic.
Engine Instance	Select the compute engine where the resource resides from the Engine Instance drop-down list.

iii. Click **Create**.

iv. On the configuration tab of the resource, select the **MaxCompute compute engine** and enter the content of the custom dictionary.

Dictionary format:

- Each word occupies one line.
- Each line contains the following parts: word, frequency, and parts of speech. The frequency and part of speech are optional. Separate every two parts with a space. The order of the three parts cannot be adjusted.

If you upload a dictionary file from the on-premises machine to DataWorks, the file must be encoded in UTF-8.

v. Click the icon in the toolbar to commit the resource.

vi. Create a **PyODPS 2** node and enter the following code:

```
def test(resources):
    import jieba
    import sys
    reload(sys)
    sys.setdefaultencoding('utf-8')
    fileobj = resources[0]
    def h(input_var):# Use the nested h() function to load the dictionary and segme
nt text.
        import jieba
        jieba.load_userdict(fileobj)
        result=jieba.cut(input_var, cut_all=False)
        return "/ ".join(result)
    return h
hints = {
    'odps.isolation.session.enable': True
}
libraries =['jieba-master.zip']  # Reference the jieba-master.zip package.
iris = o.get_table('jieba_test').to_df()  # Reference data in the jieba_test table.

file_object = o.get_resource('key_words.txt') # Use the get_resource() function to
reference the MaxCompute resource.
example = iris.chinese.map(test, resources=[file_object]).execute(hints=hints, libr
aries=libraries) # Call the map function to transfer the resources parameter.
print(example)  # Display the text segmentation result, which is of the MAP type.
abci=list(example)   # Convert the text segmentation result to the List type.
for i in range(i,len(abci)):
    pq=str(abci[i])
    o.write_table('jieba_result',[pq])  # Write the data records to the jieba_resul
t table one by one.
    i+=1
else:
    print("done")
```

vii. Run the code and compare the results before and after the custom dictionary is referenced.

2.8. Build a data warehouse for an enterprise based on AnalyticDB for MySQL

This topic describes how to build a data warehouse for an enterprise based on AnalyticDB for MySQL, and use the data warehouse to perform O&M and manage metadata.

Before you perform operations in this topic, create a workspace. For more information, see Create a workspace.

Configure an AnalyticDB for MySQL connection

1. Log on to the DataWorks console. In the left-side navigation pane, click Workspaces. On the Workspaces page, find the target workspace and click **Data Integration** in the Actions column.
2. On the **Data Integration** page, click **Connection** in the left-side navigation pane.

3. On the **Data Source** page that appears, click **Add a Connection** in the upper-right corner.
4. In the **Add Connection** dialog box that appears, click **AnalyticDB for MySQL2.0** or AnalyticDB for MySQL3.0 in the Relational Database section.
5. In the dialog box that appears, set the required parameters.

   > **Note**
   > - Only nodes that run on exclusive resource groups for scheduling can access AnalyticDB for MySQL data stores. For more information about how to create and use an exclusive resource group, see Create and use an exclusive resource group for scheduling.
   > - When you create an AnalyticDB for MySQL 2.0 connection, you must enter the AccessKey ID and AccessKey secret of the current Alibaba Cloud account or RAM user for identity authentication.
   > - When you create an AnalyticDB for MySQL 3.0 connection, you must enter the username and password used to connect to the AnalyticDB for MySQL 3.0 database for identity authentication. After you create an AnalyticDB for MySQL 3.0 database, you must log on to Cloud Native Data Warehouse Console and create the username and password for connecting to the database first.

6. Click **Test Connection**.
7. After the connection passes the connectivity test, click **Complete**.

Add information about your exclusive resource group for scheduling to the whitelist of the AnalyticDB for MySQL 3.0 database

An AnalyticDB for MySQL 3.0 database uses a whitelist to control client access. It allows access requests from a client only when the client information is included in the whitelist.

1. Obtain information about the exclusive resource group for scheduling.

   To guarantee that an exclusive resource group for scheduling can properly access an AnalyticDB for MySQL 3.0 database, you must add information, such as the Elastic Network Interface (ENI) IP address, about the exclusive resource group to the whitelist of the AnalyticDB for MySQL 3.0 database. For more information, see Add the exclusive resource group for scheduling to the whitelist of the data source to be accessed. Skip the operations in this section if you use an AnalyticDB for MySQL 2.0 database.

2. Add information about the exclusive resource group for scheduling to the whitelist of the AnalyticDB for MySQL 3.0 database.
   i. Log on to Cloud Native Data Warehouse Console. In the left-side navigation pane, click **Clusters**. On the page that appears, find the target cluster on the V3.0 Clusters tab and click the cluster ID. The details page of the cluster appears. Then, click **Data Security**.
   ii. On the page that appears, click **Create Whitelist** in the upper-right corner. In the dialog box that appears, add information about the exclusive resource group to the whitelist.

Create a workflow

1. In the DataWorks console, click the DataWorks icon in the upper-left corner and choose **All Products > DataStudio**.
2. On the Data Analytics tab, right-click Business Flow and select Create Workflow. In the **Create Workflow** dialog box that appears, set **Workflow Name** and **Description**.

3. Click **Create**.

Create a batch sync node

1. Click the created workflow, right-click **Data Integration**, and then choose **Create > Batch Synchronization**.
2. In the **Create Node** dialog box that appears, set **Node Name** and click **Commit**. A batch sync node is created.
3. On the configuration tab of the batch sync node, set the parameters under **Source** and **Target** in the Connections section.
4. Configure the mappings between the fields in the source and destination tables in the Mappings section.

   Fields in the source table on the left have a one-to-one mapping with fields in the destination table on the right. You can click **Add** to add a field, or move the pointer over a field and click the **Delete** icon to delete the field.
5. Configure channel control policies.

   Configure the maximum transmission rate and dirty data check rules.

Parameter	Description
Expected Maximum Concurrency	The maximum number of concurrent threads to read data from or write data to data stores within the batch sync node. You can configure the concurrency for the node on the codeless user interface (UI).
Bandwidth Throttling	Specifies whether to enable bandwidth throttling. You can enable bandwidth throttling and set a maximum transmission rate to avoid heavy read workload of the source. We recommend that you enable bandwidth throttling and set the maximum transmission rate to a proper value.
Dirty Data Records Allowed	The maximum number of dirty data records allowed.

6. Click the **Properties** tab in the right-side navigation pane to configure recurrence for the batch sync node.
7. After the configuration is completed, click the **Save** and **Run** icons in sequence.

Create a data analytics node

1. Click the target workflow, right-click **UserDefined**, and then choose **Create > AnalyticDB for MySQL**.
2. In the **Create Node** dialog box that appears, set **Node Name** and click **Commit**.
3. On the configuration tab that appears, select a connection and enter SQL statements based on the syntax supported by AnalyticDB for MySQL. You can enter Data Manipulation Language (DML) or Data Definition Language (DDL) statements in the SQL code editor.
4. Click the **Properties** tab in the right-side navigation pane to configure recurrence for the node.
5. After the configuration is completed, click the **Save** icon to save the node settings. Then, click the **Run** icon to run the SQL statements you entered.

Perform O&M

After you commit and deploy a created node, click the DataWorks icon in the upper-left corner and choose **All Products > Operation Center**. You can perform O&M on the page that appears. For more information, see Operation Center.

Manage metadata

Click the DataWorks icon in the upper-left corner and choose **All Products > DataMap**. You can manage metadata on the page that appears. For more information, see Data Map.

3.Data security

3.1. Grant access to a specific UDF to a specified user

This topic describes how to grant access to a specific user-defined function (UDF) only to a specified user. This best practice relates to data security as it involves data encryption and decryption algorithms.

Prerequisites

The MaxCompute client is installed. For more information, see MaxCompute client.

Context

Typically, you can use one of the following methods to control users' access permissions:

- Create a package to include all objects in a workspace that are required by another target workspace, and authorize use of the package in the target workspace.

  This method applies to scenarios where you want to authorize shared access to resources across workspaces. However, after you grant access to the package to a user with the developer role, the user has full permissions on all objects in the package. This may incur uncontrollable risks. For more information, see Package-based resource sharing across projects.

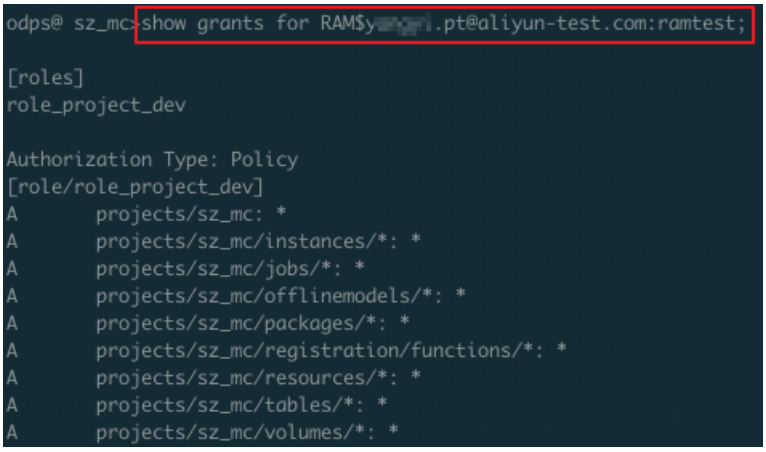
  - The following figure shows the permissions that the developer role has on a DataWorks workspace.

    ```
    odps@ sz_mc>desc role role_project_dev;

    Authorization Type: Policy
    A       projects/sz_mc: *
    A       projects/sz_mc/instances/*: *
    A       projects/sz_mc/jobs/*: *
    A       projects/sz_mc/offlinemodels/*: *
    A       projects/sz_mc/packages/*: *
    A       projects/sz_mc/registration/functions/*: *
    A       projects/sz_mc/resources/*: *
    A       projects/sz_mc/tables/*: *
    A       projects/sz_mc/volumes/*:
    ```

    As shown in the preceding figure, the developer role has full permissions on all packages, functions, resources, and tables in the workspace by default. This does not meet permission management requirements.

- The following figure shows the permissions that a Resource Access Management (RAM) user has on a DataWorks workspace after the RAM user is assigned the developer role.



  In view of the above, you cannot precisely grant access to a specific UDF to a specified user by using package-based authorization or by assigning the default role of DataWorks to the user. For example, if you assign the developer role to the RAM user `RAM$xxxxx.pt@example.com:ramtest`, the RAM user has full permissions on all objects in the current workspace. For more information, see Authorize users.

- Create a role in the DataWorks console for permission control.

  Log on to the DataWorks console. In the left-side navigation pane, click Workspaces. On the page that appears, find the target workspace and click Data Analytics in the Actions column. On the DataStudio page that appears, click the **Workspace Manage** icon in the upper-right corner. On the page that appears, click **MaxCompute Management** in the left-side navigation pane and then **Custom User Roles**. On the Custom User Roles page that appears, click Create Role to create a role for permission control. This method, however, can only grant permissions on a table or workspace, but not on a specific UDF.

- Use a role policy and a project policy to grant access to a specific UDF only to a specified user.

  Role and project policies allow you to grant a specific permission on a specific resource to a specified user.

  > **Note** For security purposes, we recommend that you apply role and project policies in a test workspace if you are a beginner of DataWorks.

To sum up, you can use a role policy and a project policy to grant access to a specific UDF only to a specified user.

- To forbid users from accessing a specific resource in a workspace, follow these steps: Assign the developer role to the users and configure a role policy to deny the users' requests for accessing the resource on the MaxCompute client.
- To permit one of these forbidden users to access the resource, configure a project policy to allow the user's requests for accessing the resource on the MaxCompute client.

Procedure

1. Create a role that is by default denied access to the UDF named getregion.
   i. On the MaxCompute client, run the following command to create the role denyudfrole:

```
create role denyudfrole;
```

ii. Create a role policy file with the following content:

```
{
"Version": "1", "Statement"
[{
"Effect":"Deny",
"Action":["odps:Read","odps:List"],
"Resource":"acs:odps:*:projects/sz_mc/resources/getaddr.jar"
},
{
"Effect":"Deny",
"Action":["odps:Read","odps:List"],
"Resource":"acs:odps:*:projects/sz_mc/registration/functions/getregion"
}
 ] }
```

iii. On the MaxCompute client, set the storage path for the role policy file.

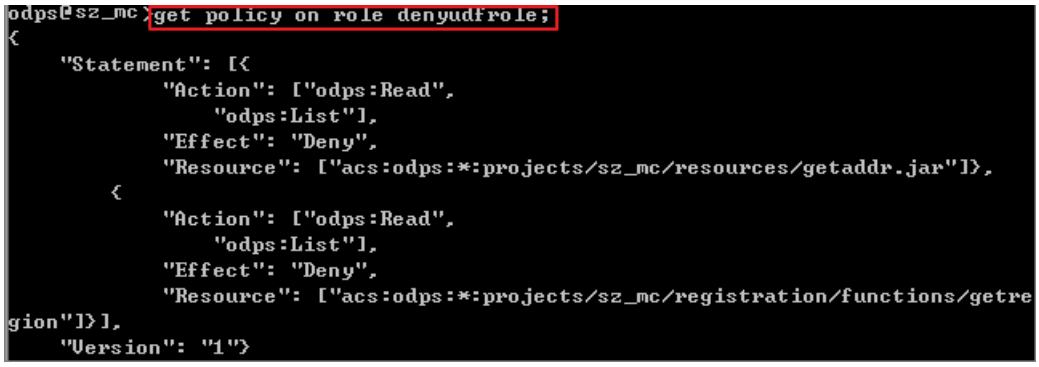
Run the following command:

```
put policy /Users/yangyi/Desktop/role_policy.json on role denyudfrole;
```

iv. On the MaxCompute client, run the following command to check the role policy:

```
get policy on role denyudfrole;
```

The following figure shows the command output.

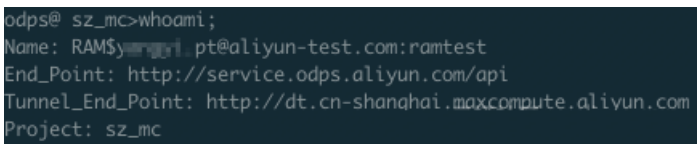


v. On the MaxCompute client, run the following command to assign the denyudfrole role to a RAM user with the developer role:

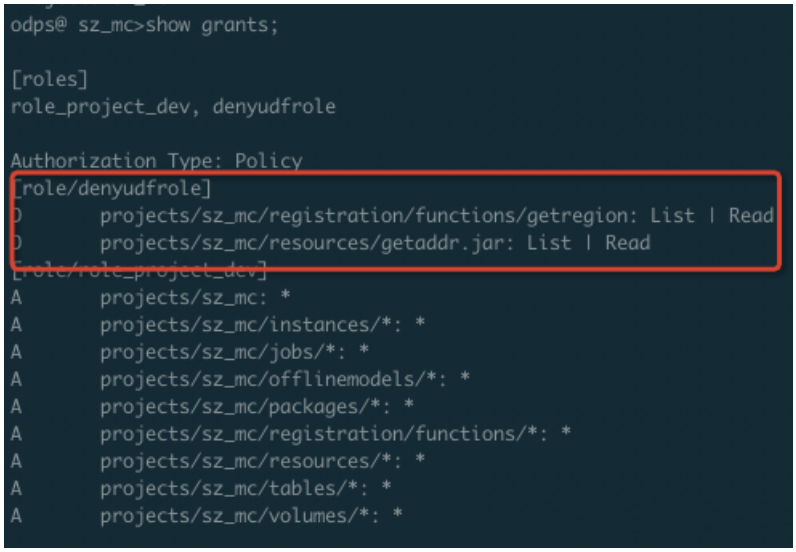
```
grant denyudfrole to RAM$xxxx.pt@example.com:ramtest;
```

2. Verify that the denyudfrole role is created.

i. Log on to the MaxCompute client as the RAM user to which the denyudfrole role is assigned. Then, run the `whoami;` command to check the current logon user.

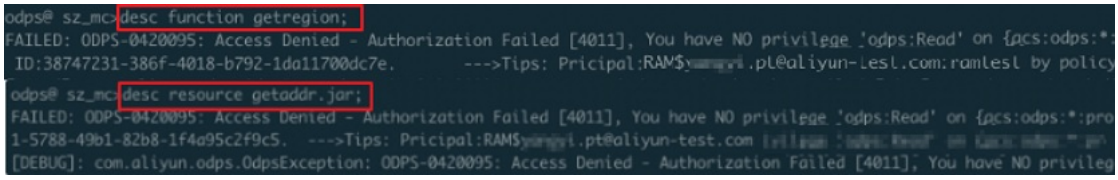


ii. Run the `show grants;` command to check the permissions of the current logon user.



The command output indicates that the RAM user has two roles: role_project_dev and denyudfrole. role_project_dev is the default developer role in a DataWorks workspace.

iii. Check the permissions of the RAM user on the UDF getregion and its dependent package.



The command output indicates that the RAM user with the developer role of a DataWorks workspace does not have the read permission on the UDF getregion. To grant access to the UDF getregion only to the RAM user, you need to further configure a project policy.

3. Configure a project policy.

i. Create a project policy file with the following content:

```
{
"Version": "1", "Statement":
[{
"Effect":"Allow",
"Principal":"RAM$yangyi.pt@example.com:yangyitest",
"Action":["odps:Read","odps:List","odps:Select"],
"Resource":"acs:odps:*:projects/sz_mc/resources/getaddr.jar"
},
{
"Effect":"Allow",
 "Principal":"RAM$xxxx.pt@example.com:yangyitest",
"Action":["odps:Read","odps:List","odps:Select"],
"Resource":"acs:odps:*:projects/sz_mc/registration/functions/getregion"
}] }
```

ii. On the MaxCompute client, set the storage path for the project policy file.

Run the following command:

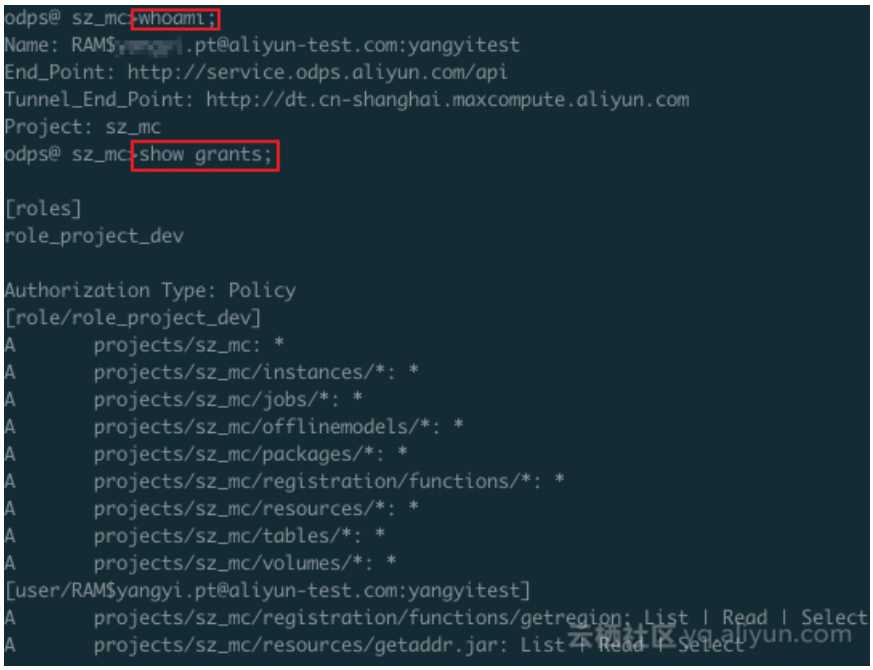
```
put policy /Users/yangyi/Desktop/project_policy.json;
```

iii. On the MaxCompute client, run the following command to check the project policy:

```
get policy;
```

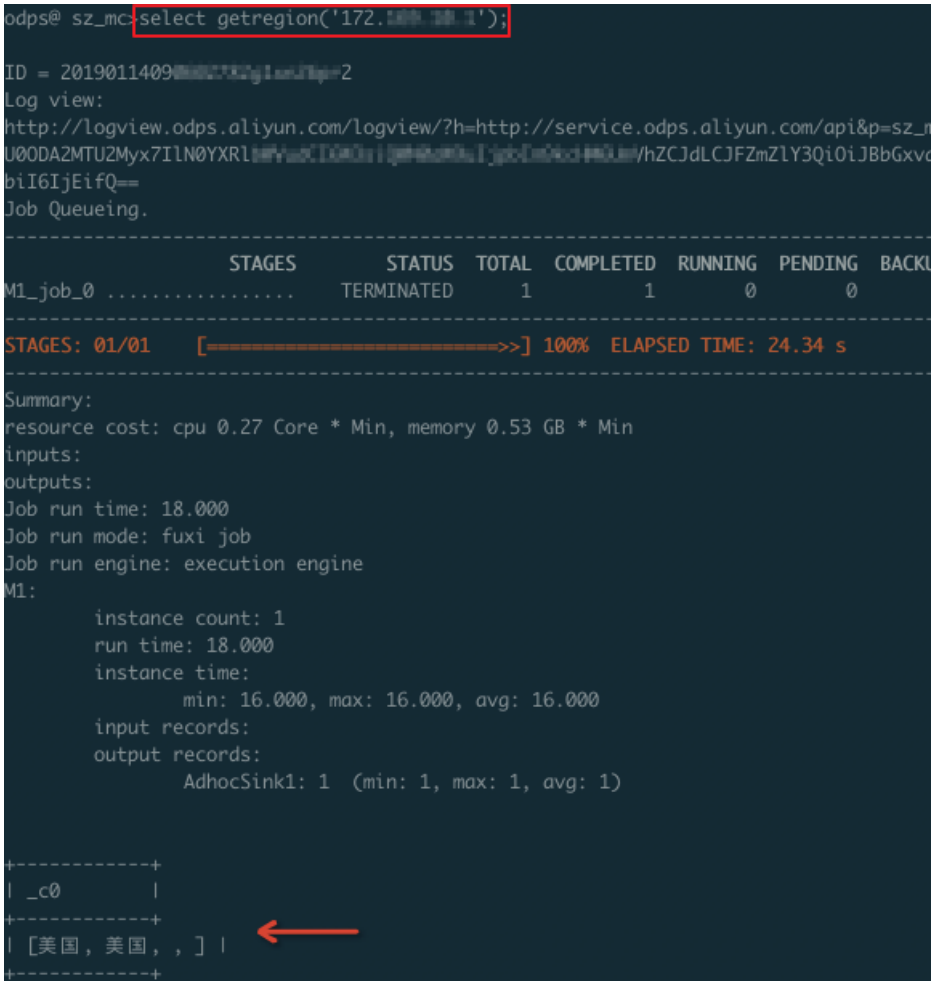
The following figure shows the command output.

iv. Run the `whoami;` command to check the current logon user. Then, run the `show grants;` command to check the permissions of the user.



v. Run an SQL node and check whether only the specified RAM user can access the specific UDF and its dependent package.

- The following command output indicates that the specified RAM user can access the specific UDF.



- The following command output indicates that the specified RAM user can access the dependent package of the UDF.

3.2. Allow a RAM user to log on to DataWorks only from a specific IP address

This topic describes how to allow a RAM user to log on to DataWorks only from a specific local IP address.

Prerequisites

A RAM user is created and granted the required permission. For more information, see Prepare a RAM

user. The **AliyunDataWorksFullAccess** permission is the system default permission and cannot be modified. You must create a custom permission policy.

Create a custom policy

1. Log on to the RAM console by using your Alibaba Cloud account.
2. In the left-side navigation pane, choose **Permissions > Policies**.
3. Click **Create Policy**.
4. On the **Create Custom Policy** page, set the **Policy Name** parameter to **dataworksIPlimit1** and the **Configuration Mode** parameter to **Script** to configure a custom policy.

   The following information is the complete content of the custom permission. The acs:SourceIP parameter specifies the IP address that is allowed to access DataWorks. You can specify multiple IP addresses. For more information about the parameters of the custom permission, see Policy elements.

```
{
    "Version": "1",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "dataworks:*"
            ],
            "Resource": [
                "acs:dataworks:*:*:*"
            ],
            "Condition": {
                "NotIpAddress": {
                    "acs:SourceIp": [
                        "10.0.0.0",
                        "192.168.0.0"
                    ]
                }
            }
        }
    ]
}
```

5. Click **OK**.

Attach the custom policy to the RAM user

1. In the left-side navigation pane, click **Users** under **Identities**.
2. In the **User Logon Name/Display Name** column, click the username of the RAM user.
3. Click the **Permissions** tab, and click **Add Permissions**. Then, the **Principal** field is automatically set.
4. Click the **Custom Policy** tab.
5. In the **Authorization Policy Name** column on the left, click the permission policy that you want to attach to the RAM user.

> **Note** In the box on the right, you can click the cross sign (×) next to a policy to delete the policy.

6. Click **OK**.
7. Click **Complete**.

4.Data analysis

4.1. Intelligently recommend items on e-commerce websites

Based on its cutting-edge big data and artificial intelligence (AI) technologies and years of experience in the e-commerce industry, Alibaba provides Artificial Intelligence Recommendation for developers. Artificial Intelligence Recommendation is a personalized recommendation service that can help increase the customer purchase rate and order conversion rate.

Overview

This topic describes how to use Artificial Intelligence Recommendation to build an intelligent system for recommending items on e-commerce websites. The system uses the following services of Alibaba Cloud: Log Service for collecting logs, ApsaraDB for RDS as the back-end data service, MaxCompute as a data warehouse, and DataWorks for synchronizing and processing data.

Recommendation information on e-commerce websites includes product properties such as logistics information and sales information. Such information appears on homepages or as news feeds and can promote transactions and increase the customer purchase rate and order conversion rate.

Scenarios

- Displays users' preferred and personalized contents in a waterfall flow view.
- Displays recommendations related to a product on the product details page.
- Displays focus pictures and popular recommendations on website homepages.
- Displays news, live videos, and social events.

Process

The process of implementing intelligent recommendation on e-commerce websites is as follows:

1. Log Service collects user behavior logs.
2. Log Service uploads user behavior logs to MaxCompute.
3. Log Service uploads product and user data to MaxCompute.
4. MaxCompute pushes data to Artificial Intelligent Recommendation, which then uses AI algorithms to generate recommendations based on the data.

Highlights

- Artificial Intelligent Recommendation is a mature solution developed by Alibaba based on its years of experience. It is suitable for various industries and scenarios and can meet diversified business needs of different users.
- DataWorks in this solution frees you from concerns about stream processing, algorithms, O&M, and monitoring.
- Data is isolated between users and sensitive information is encrypted, which guarantee information security.
- Advanced algorithm technologies, multi-mode integration, efficient cold start, real-time policy adjustment, and model training are used, freeing you from manual operations.

- Multiple services are seamlessly integrated, which enables data synchronization within hours.

Details

For more information, see Best practice of intelligently recommending items on e-commerce websites.

4.2. Analyze offline data in the Internet and e-commerce industries

You can use MaxCompute, ApsaraDB RDS for MySQL, and DataWorks that Alibaba Cloud provides to analyze offline data in the Internet and e-commerce industries. Then you can use DataV to present business metrics based on the analysis results.

Overview

This topic describes how to use big data analytics to analyze the sales data of e-commerce websites and present the generated statistics on a DataV dashboard. Such statistics include the sales metrics, customer metrics, sales rankings, and order distribution. DataV dynamically displays sales data on dashboards and allows you to query data as required, greatly improving data readability.

Scenarios

- Displays data of e-commerce websites on kanbans.
- Displays the analysis results of the business trends in and outside China.
- Monitors data risks in the Internet and financial industries.

Process

The process of analyzing offline data in the Internet and e-commerce industries is as follows:

1. ApsaraDB for RDS synchronizes user orders and other data to MaxCompute.
2. MaxCompute transmits the raw data to DataWorks. DataWorks processes the raw data and exposes an API for accessing the processed data.
3. DataV requests the processed data from DataWorks through the API and displays the data on a dashboard.

Highlights

- Large-scale storage: Ultra-large storage capacity is available to store exabytes of data.
- High performance: This solution achieves more efficient and stable performance.
- Low cost: This solution costs less than using a user-created database.
- High security: Tenant data is isolated between workspaces and all computing nodes are run in sandboxes.
- Visualized editing: You can drag and drop items on the graphical editing page to visualize big data in a professional way.

Details

For more information, see Best practice of analyzing offline data in the Internet and e-commerce industries.

4.3. Use MaxCompute to analyze data and Quick BI to present the analysis results

You can use MaxCompute, ApsaraDB RDS for MySQL, and DataWorks that Alibaba Cloud provides to analyze data, and use Quick BI to present the analysis results.

Overview

This topic describes how to use MaxCompute and DataWorks to perform extract, transform, load (ETL) operations on business and log data and synchronize the data to AnalyticDB for MySQL, and then use Quick BI to present the data analysis results.

Scenarios

- Performs Business Intelligence (BI) analysis for websites, apps, and mini programs in the Internet, e-commerce, and gaming industries.
- Performs BI analysis for various websites.

Process

The process of using MaxCompute to analyze data and Quick BI to present the analysis results is as follows:

1. Data Integration synchronizes business and log data to MaxCompute.
2. MaxCompute and DataWorks perform ETL operations on the data.
3. MaxCompute and DataWorks synchronize the data analysis results to AnalyticDB for MySQL.
4. Quick BI obtains the analysis results from AnalyticDB for MySQL and displays the results.

Highlights

- With the core capability of quick and real-time data analysis provided by AnalyticDB for MySQL and Quick BI, this solution attracts users to synchronize business and log data to Alibaba Cloud Log Service and AnalyticDB.
- Log Service used in this solution can be seamlessly integrated with various services, such as Realtime Compute, Elasticsearch, AnalyticDB for MySQL, E-MapReduce, and DataV, enhancing user experience.
- This solution leverages powerful data processing and analysis capabilities of MaxCompute and AnalyticDB for MySQL, which makes it easier to build big data platforms and compute huge amounts of data. This solution also effectively reduces costs while guaranteeing data security for enterprises.
- This solution is seamlessly integrated with third-party open-source services. In this way, these third-party services can transmit logs to Log Service without the need to access user applications.

Details

For more information, see Best practice of using MaxCompute to analyze data and Quick BI to present the analysis results.