

**Data Lake Insight**

# **Best Practice**

**Issue**            01  
**Date**             2025-01-10



**Copyright © Huawei Technologies Co., Ltd. 2025. All rights reserved.**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

## **Trademarks and Permissions**



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

## **Notice**

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

# Security Declaration

## Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process*. For details about this process, visit the following web page:

<https://www.huawei.com/en/psirt/vul-response-process>

For vulnerability information, enterprise customers can visit the following web page:

<https://securitybulletin.huawei.com/enterprise/en/security-advisory>

---

# Contents

---

<b>1 Overview.....</b>	<b>1</b>
<b>2 Analyzing Driving Behavior Data in IoV Scenarios Using DLI.....</b>	<b>3</b>
<b>3 Converting Data Format from CSV to Parquet.....</b>	<b>14</b>
<b>4 Analyzing E-Commerce BI Reports Using DLI.....</b>	<b>18</b>
<b>5 Analyzing Billing Consumption Data Using DLI.....</b>	<b>26</b>
<b>6 Analyzing Real-time E-Commerce Business Data Using DLI.....</b>	<b>31</b>
<b>7 Connecting BI Tools to DLI for Data Analysis.....</b>	<b>46</b>
7.1 Overview.....	46
7.2 Connecting DBeaver to DLI for Data Analysis and Query.....	46
7.3 Connecting DBT to DLI for Data Scheduling and Analysis.....	50
7.4 Connecting Yonghong BI to DLI for Data Query and Analysis.....	53
7.5 Connecting Power BI to DLI Using Kyuubi for Data Analysis and Query.....	58
7.6 Connecting FineBI to DLI Using Kyuubi for Data Analysis and Query.....	67
7.7 Connecting Superset to DLI Using Kyuubi for Data Analysis and Query.....	75
7.8 Connecting Tableau to DLI Using Kyuubi for Data Analysis and Query.....	84
7.9 Connecting Beeline to DLI Using Kyuubi for Data Analysis and Query.....	93

# 1 Overview

**Table 1-1** DLI best practices

Best Practice	Description
<a href="#">Analyzing Driving Behavior Data in IoV Scenarios Using DLI</a>	Use DLI to analyze driving behavior data in IoV scenarios.
<a href="#">Converting Data Format from CSV to Parquet</a>	Use DLI to convert CSV data to Parquet data.
<a href="#">Analyzing E-Commerce BI Reports Using DLI</a>	Use DLI to analyze e-commerce BI reports, based on real user, product, and review data (anonymized) from an online mall.
<a href="#">Analyzing Billing Consumption Data Using DLI</a>	Use DLI to analyze billing data and optimize costs, based on actual consumption data samples from DLI.
<a href="#">Analyzing Real-time E-Commerce Business Data Using DLI</a>	Use DLI Flink to analyze real-time e-commerce business data.
<a href="#">Connecting DBeaver to DLI for Data Analysis and Query</a>	Connect DBeaver to DLI and submit SQL queries.
<a href="#">Connecting DBT to DLI for Data Scheduling and Analysis</a>	Use DBT to submit DLI jobs.
<a href="#">Connecting Yonghong BI to DLI for Data Query and Analysis</a>	Connect Yonghong BI to DLI.
<a href="#">Connecting Power BI to DLI Using Kyuubi for Data Analysis and Query</a>	Connect Power BI to DLI using Kyuubi to access and analyze data in DLI.

Best Practice	Description
<a href="#">Connecting FineBI to DLI Using Kyuubi for Data Analysis and Query</a>	Connect FineBI to DLI using Kyuubi.
<a href="#">Connecting Superset to DLI Using Kyuubi for Data Analysis and Query</a>	Connect Superset to DLI using Kyuubi.
<a href="#">Connecting Tableau to DLI Using Kyuubi for Data Analysis and Query</a>	Connect Tableau to DLI using Kyuubi.
<a href="#">Connecting Beeline to DLI Using Kyuubi for Data Analysis and Query</a>	Connect Beeline to DLI using Kyuubi.

# 2 Analyzing Driving Behavior Data in IoV Scenarios Using DLI

## Scenario

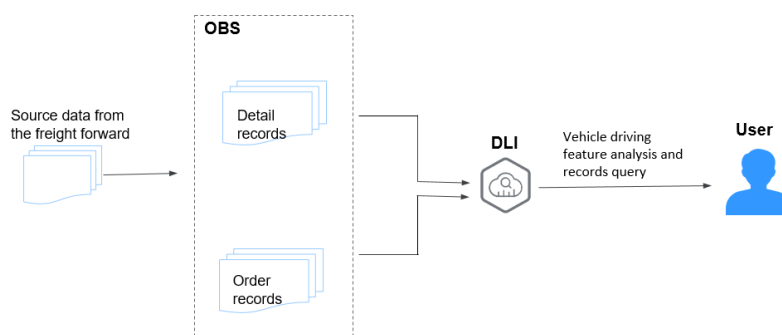
Cloud computing and big data provide companies with data analysis and mining capabilities required in the Internet of Vehicle (IoV) field, helping companies or department of motor vehicles manage and analyze vehicle and driving behavior data quickly and scientifically.

## Solution Architecture

DLI can query the records of vehicle driving features based on the detail records and freight order data regularly reported by the freight forwarder.

**Data Types** describes the data types used by DLI to record the data.

**Figure 2-1** Solution Overview



## Process

To use DLI to analyze driving behavior data, perform the following steps:

**Step 1: Uploading Data.** Upload the data to OBS.

**Step 2: Analyzing Data.** Use DLI to query the data.

## Example Code

Download the [data package](#) for sample data and detailed SQL statements.

## Solution Advantages

- Free of data migration: DLI can interconnect with multiple data sources. You only need to create SQL tables and map data sources.
- Easy to use: You can use standard SQL statements to compile metric analysis logic without paying attention to the complex distributed computing platform.
- Pay-per-use: Log analysis is scheduled periodically based on time-critical requirements. There is a long idle period between every two scheduling operations. DLI uses the pay-per-use billing mode, which effectively reduces your costs.

## Resource Planning and Costs

Table 2-1 Resource planning and costs

Resource	Description	Cost
OBS	You need to create an OBS bucket and upload data to OBS for data analysis using DLI.	<p>You will be charged for using the following OBS resources:</p> <ul style="list-style-type: none"><li>• <b>Storage Fee</b> for storing static website files in OBS.</li><li>• <b>Request Fee</b> for accessing static website files stored in OBS.</li><li>• <b>Traffic Fee</b> for using a custom domain name to access OBS over the public network.</li></ul> <p>The actual fee depends on the size of the stored file, the number of user access requests, and the traffic volume. Estimate the fee based on your service requirements.</p>
DLI	Before creating a SQL job, you need to purchase a queue. When using queue resources, you are billed based on the CUH of the queue.	<p>For example, if you purchase a pay-per-use queue, you will be billed based on the number of CUHs used by the queue.</p> <p>Usage is billed by the hour, with any usage less than one hour being rounded up to one hour. The number of hours is calculated on the hour. CUH pay-per-use billing = Unit price x Number of CUs x Number of hours.</p>



## Data Types

- Detail records

Detail records include the regularly reported location records and data of alarms triggered by abnormal driving behavior.

**Table 2-2** Detail records

Field	Data Type	Description
driverID	string	Driver ID
carNumber	string	License plate number
latitude	double	Latitude
longitude	double	Longitude
speed	int	Speed
direction	int	Direction
siteName	string	Site name
time	timestamp	Report time of the records
isRapidlySpeedup	int	Whether the vehicle rapidly speeds up. <b>1</b> indicates that the vehicle suddenly speeds up, and <b>0</b> indicates that the vehicle does not.
isRapidlySlowdown	int	Whether the vehicle suddenly slows down.
isNeutralSlide	int	Whether the vehicle is coasting.
isNeutralSlideFinished	int	Whether vehicle coasting has stopped.
neutralSlideTime	bigint	Time length of vehicle coasting.
isOverspeed	int	Whether the vehicle is speeding.
isOverspeedFinished	int	Whether the vehicle stops speeding.
overspeedTime	bigint	Duration of the vehicle speeding
isFatigueDriving	int	Whether fatigue driving occurs.

Field	Data Type	Description
isHthrottleStop	int	Whether the driver revs the engine in neutral.
isOilLeak	int	Abnormal oil consumption

- Order data  
Order data refers to the records of freight orders.

**Table 2-3** Order data

Field	Data Type	Description
orderNumber	string	Order ID
driverID	string	Driver ID
carNumber	string	License plate number
customerID	string	Customer ID
sourceCity	string	Departure
targetCity	string	Destination
expectArriveTime	timestamp	Expected delivery time
time	timestamp	Time when a record is generated.
action	string	Event type, including creating an order, dispatching goods, delivering packages, and signing orders.

## Step 1: Uploading Data

Upload the data to OBS for data analysis using DLI.

1. Download OBS Browser+. For details about the download address, see [Object Storage Service Tool Guide](#).
2. Install OBS Browser+. For details about the installation procedure, see [Object Storage Service Tool Guide](#).
3. Log in to OBS Browser+. OBS Browser+ supports two login modes: AK login (using access keys) or authorization code login. For details about the login procedure, see [Object Storage Service Tool Guide](#).
4. Upload data using the OBS browser+.

Start the OBS Browser+, click **Create Bucket** on the homepage. Select a region and enter a bucket name (for example, **DLI-demo**). After the bucket is created, return to the bucket list and click **DLI-demo**. OBS Browser+ supports

upload by dragging. You can drag one or more files or folders from a local path to the object list of a bucket or a parallel file system on OBS Browser+. You can even drag a file or folder directly to a specified folder on OBS Browser+.


Obtain the test data by downloading the [Best\\_Practice\\_01.zip](#) file and decompressing it. Perform the following operations:

- Detail records: Upload the **detail-records** folder in the **Data** directory to the root directory of the OBS bucket.
- Order data: Upload the **order-records** folder in the **Data** directory to the root directory of the OBS bucket.

## Step 2: Analyzing Data

Use DLI to query the data for analysis.




### 1. Creating a Database and a Table

- On the homepage of the management console, choose **Service List > Analytics > Data Lake Insight**.
- On the DLI console, click **SQL Editor**.
- In the left pane of the SQL Editor, select the **Databases** tab and click  to create the **demo** database.

**Figure 2-2** Creating a database

### Create Database

You can create 883 more databases. [Increase quota](#).

* Database Name	<input type="text" value="demo"/>		
* Enterprise Project	<input type="text" value="default"/>   <a href="#">Create Enterprise Project</a>		
Description	<input type="text" value=""/> 0/256		
Tags	<p>It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. <a href="#">View predefined tags</a> </p> <table><tr><td><input type="text" value="Tag key"/></td><td><input type="text" value="Tag value"/></td></tr></table> <p>You can add 10 more tags.</p>	<input type="text" value="Tag key"/>	<input type="text" value="Tag value"/>
<input type="text" value="Tag key"/>	<input type="text" value="Tag value"/>		
<input type="button" value="OK"/> <input type="button" value="Cancel"/>			

 NOTE

**Database Name** cannot be set to **default** because **default** is the built-in database.

- d. Choose the **demo** database, and enter the following SQL statement in the editing box:

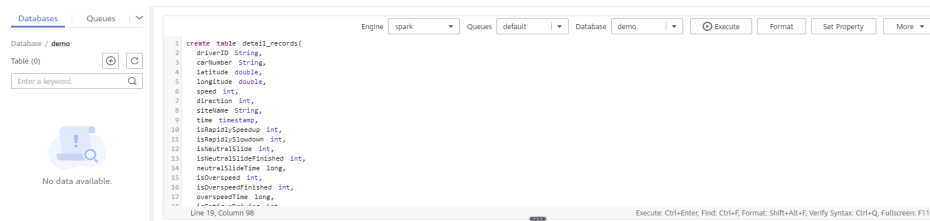
```
create table detail_records(  
  driverID String,  
  carNumber String,  
  latitude double,  
  longitude double,  
  speed int,  
  direction int,  
  siteName String,  
  time timestamp,  
  isRapidlySpeedup int,  
  isRapidlySlowdown int,  
  isNeutralSlide int,  
  isNeutralSlideFinished int,  
  neutralSlideTime long,  
  isOverspeed int,  
  isOverspeedFinished int,  
  overspeedTime long,  
  isFatigueDriving int,  
  isHthrottleStop int,  
  isOilLeak int) USING CSV OPTIONS (PATH 'obs://dli-demo/detail-records/');
```

 NOTE

Replace the file path in the preceding statement with the actual OBS path where the detail records are stored.

- e. Click **Execute** to create the **detail\_records** table. See [Figure 2-3](#).

**Figure 2-3** Creating the **detail\_records** table



- f. Run the following SQL statements to create the **event\_records** table in the **demo** database. The operation is similar to [1.d](#) and [1.e](#).

```
create table event_records(  
  driverID String,  
  carNumber String,  
  latitude double,  
  longitude double,  
  speed int,  
  direction int,  
  siteName String,  
  time timestamp,  
  isRapidlySpeedup int,  
  isRapidlySlowdown int,  
  isNeutralSlide int,  
  isNeutralSlideFinished int,  
  neutralSlideTime long,  
  isOverspeed int,  
  isOverspeedFinished int,  
  overspeedTime long,  
  isFatigueDriving int,  
  isHthrottleStop int,  
  isOilLeak int)
```

- g. Run the following SQL statements to extract the alarm and event data from the detail records and insert it into the **event\_records** table.

```
insert into table event_records
(select *
from detail_records
where isRapidlySpeedup > 0
OR isRapidlySlowdown > 0
OR isNeutralSlide > 0
OR isNeutralSlideFinished > 0
OR isOverspeed > 0
OR isOverspeedFinished > 0
OR isFatigueDriving > 0
OR isHthrottleStop > 0
OR isOilLeak > 0)
```

- h. Use another method to create the **order\_records** table.

On the left of the SQL job editor, click the **Databases** tab and click the demo database. Click the plus icon (+) on the right of **Table** to create a table, and set **Data Location** to **DLI**. Set the column types according to **Order data**.

**Figure 2-4** Creating the **order\_records** table

**Create Table**

You can create 51 more tables. [Increase quota.](#)

\* Name: order\_records

\* Data Location: DLI

Table Description: 0/100

* Column Name	* Type	Description	Operation
Normal   orderNumber	string		🗑️
Normal   driverID	string		🗑️
Normal   carNumber	string		🗑️
Normal   customerID	string		🗑️
Normal   sourceCity	string		🗑️
Normal   targetCity	string		🗑️
Normal   expectArriveTime	timestamp		🗑️
Normal   time	timestamp		🗑️
Normal   action	string		🗑️

Number of Columns: 9. If there are a large number of columns, you can use SQL statements to create a table or [import](#) column details from an Excel file.

**OK** Cancel

- i. Import the OBS data to the **order\_records** table. Choose **Data Management > Databases and Tables**. Click the demo database to go to the table management page. In the **Operation** column of the **order\_records** table, choose **More > Import**. Set **File Format** to **CSV**, the data storage path to **obs://DLI-demo/order-records/**, and retain default values for the rest parameters. Click **OK**.

#### NOTE

The default timestamp format is **yyyy-MM-dd HH:mm:ss**. To use other formats, select **Advanced Settings** and enter the desired timestamp format (not modified in this example).

Figure 2-5 Importing table data

**Import Data** ×

Database Name: demo

Table Name: order\_records

\* File Format: CSV Set the options in Advanced Settings as required.  
DLI supports the read of CSV data that is not compressed or compressed by gzip.

Queue: default

\* Path: obs://DLI-demo/order-records/ 📁

Advanced Settings:

OK Cancel

## 2. Querying Data

- a. Run the following SQL statements to query the alarm events of all drivers in a certain time period.

### NOTE

You can save the frequently-used query statements as a template by clicking **More > Save as Template** in the upper right corner of the editing window. The template is available for future use or can be modified in the SQL editor again.

Choose **Job Templates > SQL Templates** and click the **Custom Templates** tab. In the **Operation** column of the target template, click **Execute** to switch to the SQL editor. You can modify it as needed.

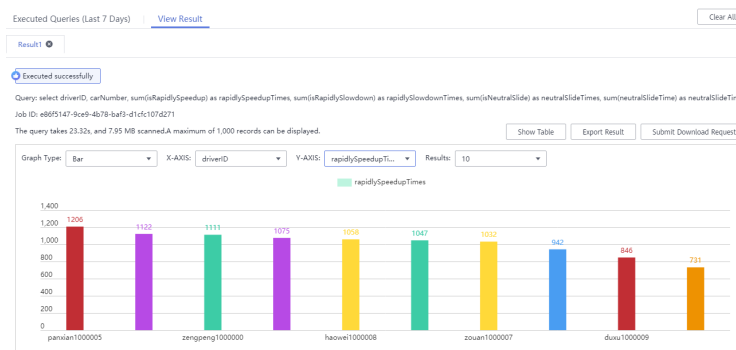
```
select
  driverID,
  carNumber,
  sum(isRapidlySpeedup) as rapidlySpeedupTimes,
  sum(isRapidlySlowdown) as rapidlySlowdownTimes,
  sum(isNeutralSlide) as neutralSlideTimes,
  sum(neutralSlideTime) as neutralSlideTimeTotal,
  sum(isOverspeed) as overspeedTimes,
  sum(overspeedTime) as overspeedTimeTotal,
  sum(isFatigueDriving) as fatigueDrivingTimes,
  sum(isHthrottleStop) as hthrottleStopTimes,
  sum(isOilLeak) as oilLeakTimes
from
  event_records
where
  time >= "2017-01-01 00:00:00"
  and time <= "2017-02-01 00:00:00"
group by
  driverID,
  carNumber
order by
  rapidlySpeedupTimes desc,
  rapidlySlowdownTimes desc,
  neutralSlideTimes desc,
  neutralSlideTimeTotal desc,
  overspeedTimes desc,
  overspeedTimeTotal desc,
  fatigueDrivingTimes desc,
  hthrottleStopTimes desc,
  oilLeakTimes desc
```

In the query result, click  to view graphical results.

- Set **Graph Type** to the bar chart.
- Set **X-AXIS** to **driverID**.
- Set **Y-AXIS** to **rapidlySpeedupTimes**.
- Set **Results** to **10**.

The command output is as follows:

**Figure 2-6** Rapid acceleration



- b. Run the following SQL statement to query the detailed record of a driver in a certain time period.

```
select
*
from
event_records
where
driverID = "panxian1000005"
and time >= "2017-01-01 00:00:00"
and time <= "2017-02-01 00:00:00"
```

In the query result, click  to view graphical results.

- Set **Graph Type** to the bar chart.
- Set **X-AXIS** to **driverID**.
- Set **Y-AXIS** to **speed**.
- Set **Results** to **10**.

The command output is as follows:

**Figure 2-7** Speeding record



c. Run the following SQL statement to query the order information.

```
select
*
from
order_records
where
orderNumber = "2017013013584419488"
order by
time desc
```

**Figure 2-8** Order information

The screenshot shows a DLI query result for order information. The query filters for orderNumber '2017013013584419488' and orders by time desc. The result is displayed as a table with the following data:

orderNumber	driverID	carNumber	customerID	sourceCity	targetCity	expectArriveTime	time	action
2017013013584419488	zouan1000007	66A58M83	zhujia151464313			Feb 1, 2017 1:58:35.000 GMT...	Jan 30, 20...	
2017013013584419488	zouan1000007	66A58M83	zhujia151464313			Feb 1, 2017 1:58:35.000 GMT...	Jan 30, 20...	

d. Run the following SQL statement to query a vehicle's driving feature according to the driver ID and time of departure.

```
select
driverID,
carNumber,
latitude,
longitude,
siteName,
time
from
detail_records
where
driverID = "panxian1000005"
and time > "2017-01-30 16:00:00"
and siteName IS NOT NULL
order by
time desc
```

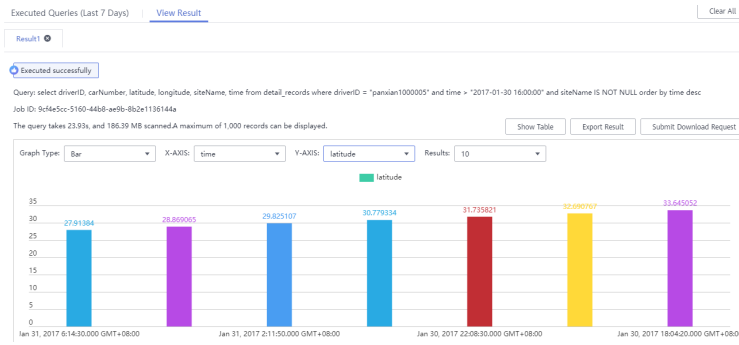
In the query result, click  to view graphical results.

- Set **Graph Type** to the bar chart.
- Set **X-AXIS** to **time**.
- Set **Y-AXIS** to **latitude**.
- Set **Results** to **10**.

The command output is as follows:



**Figure 2-9** Driving information



# 3 Converting Data Format from CSV to Parquet

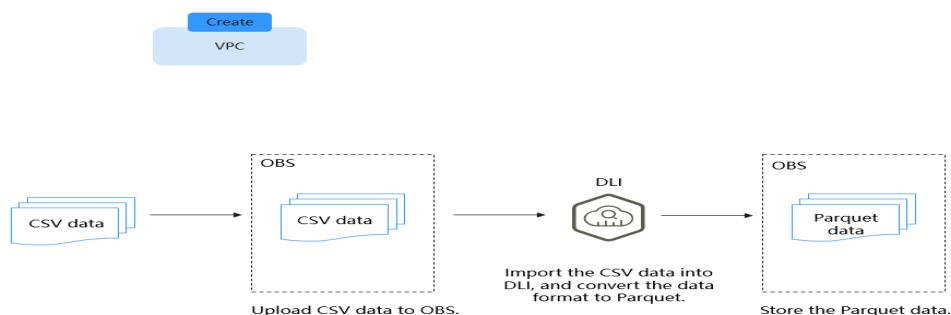
## Application Scenarios

Parquet is a columnar storage substrate created for simpler data analysis. This format can speed up queries by allowing only the required columns to be read and calculated. In addition, Parquet is built to support efficient compression schemes, which maximizes the storage efficiency on disks. Using DLI, you can easily convert data format from CSV to Parquet.

## Solution Overview

Upload CSV data to an OBS bucket, convert CSV data into Parquet data with DLI, and store the converted Parquet data to OBS.

Figure 3-1 Solution overview



## Process

To use DLI to convert CSV data into Parquet data, perform the following steps:

**Step 1: Creating and Uploading Data.** Upload data to your OBS bucket.

**Step 2: Using DLI to Convert CSV Data into Parquet Data.** Import CSV data to DLI and convert it into Parquet data.

## Solution Advantages

- **The query performance is improved.**

If you have text-based data files or tables in an HDFS and are using Spark SQL to query data, converting data format to Parquet can improve the query performance by about 30 times (or more in some cases), despite of the time consumed during the conversion.

- **Storage is saved.**

Parquet is built to support efficient compression schemes, which maximizes the storage efficiency on disks. With Parquet, the storage cost can be reduced by about 75%.

## Resource Planning and Costs

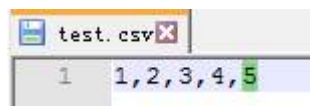
**Table 3-1** Resource planning and costs

Resource	Description	Cost
OBS	You need to create an OBS bucket and upload data to OBS for data analysis using DLI.	<p>You will be charged for using the following OBS resources:</p> <ul style="list-style-type: none"><li>• <b>Storage Fee</b> for storing static website files in OBS.</li><li>• <b>Request Fee</b> for accessing static website files stored in OBS.</li><li>• <b>Traffic Fee</b> for using a custom domain name to access OBS over the public network.</li></ul> <p>The actual fee depends on the size of the stored file, the number of user access requests, and the traffic volume. Estimate the fee based on your service requirements.</p>
DLI	Before creating a SQL job, you need to purchase a queue. When using queue resources, you are billed based on the CUH of the queue.	<p>For example, if you purchase a pay-per-use queue, you will be billed based on the number of CUHs used by the queue.</p> <p>Usage is billed by the hour, with any usage less than one hour being rounded up to one hour. The number of hours is calculated on the hour. CUH pay-per-use billing = Unit price x Number of CUs x Number of hours.</p>

## Step 1: Creating and Uploading Data

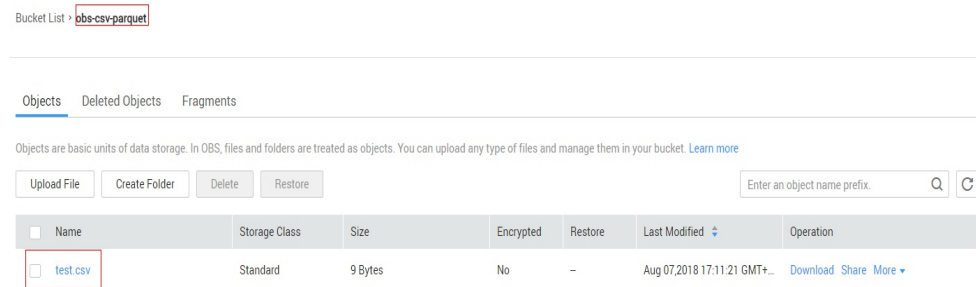
1. Create a CSV file. See **test.csv** in [Figure 3-2](#).

**Figure 3-2** Creating a **test.csv** file




- In the OBS management console, create a bucket, name it **obs-csv-parquet**, and upload the **test.csv** file to the bucket.

**Figure 3-3** Uploading CSV data to OBS



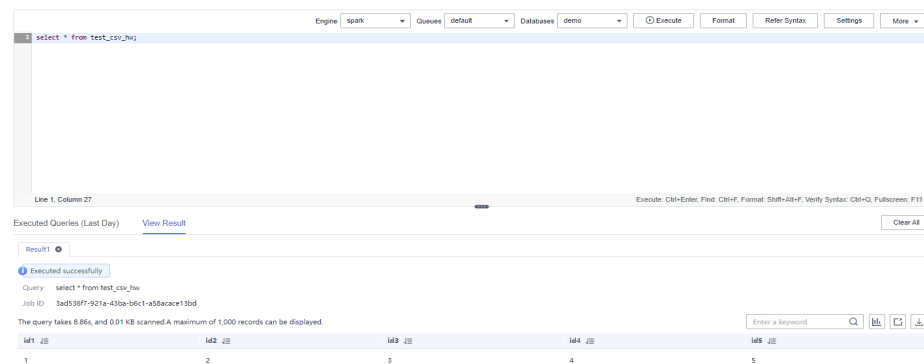
- Create a bucket and name it **obs-parquet-data** to store the converted parquet data.

## Step 2: Using DLI to Convert CSV Data into Parquet Data

- Go to the DLI console, click **SQL Editor** in the navigation pane.
- In the left pane of the SQL editor, click the **Databases** tab. Click , create a database, and name it **demo**.
- In the SQL editing window, set **Engine** to **spark**, **Queue** to **default**, and **Database** to **demo**. Execute the following statement to create table **test\_csv\_hw** to import the data in the **test.csv** file from OBS.
 

```
create table test_csv_hw(id1 int, id2 int, id3 int, id4 int, id5 int)
using csv
options(
path 'obs://obs-csv-parquet/test.csv'
)
```
- In the SQL editing window, query data in the **test\_csv\_hw** table.

**Figure 3-4** Querying data



- In the SQL job editing window, create a table to store the OBS data in Parquet format and name the table **test\_parquet\_hw**.
 

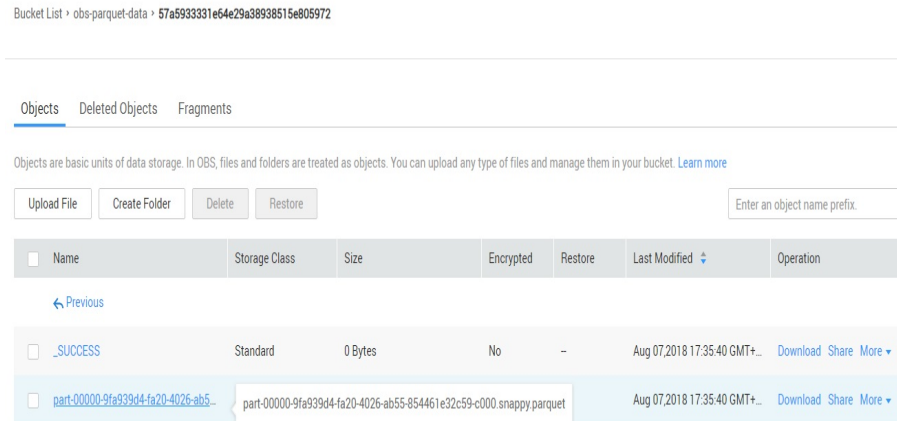
```
create table `test_parquet_hw` (`id1` INT, `id2` INT, `id3` INT, `id4` INT, `id5` INT)
using parquet
options (
path 'obs://obs-parquet-data/'
)
```

 **NOTE**

You do not need to specify a file because no Parquet file exists in this OBS bucket before the data is converted.

6. In the SQL editing window, execute the following statement to convert the CSV data to Parquet format and store the data in the specified OBS folder:  
`insert into test_parquet_hw select * from test_csv_hw`
7. Check the result. OBS automatically created a file for saving the result.

**Figure 3-5** Parquet data saved in a file in OBS



# 4 Analyzing E-Commerce BI Reports Using DLI

## Scenario

An e-commerce mall has accumulated hundreds of millions of loyal users and a massive amount of real data while maintaining rapid growth. How to use the BI tool to find business opportunities from historical data is a key issue in the precision marketing of big data applications. It is also the core technology required for intelligent upgrade of all e-commerce platforms.

This case is based on real user, product, and review data (anonymized) from an online mall. By using DLI to analyze various data features of users and products, it provides high-quality information for marketing decisions, advertisement recommendations, credit ratings, brand monitoring, and user behavior predictions.

## Process

To use DLI to analyze e-commerce data, perform the following steps:

**Step 1: Uploading Data.** Upload the data to OBS for data analysis using DLI.

**Step 2: Analyzing Data.** Use DLI to query the data for analysis.

## Data Types

To protect user privacy and data security, all sampled data is anonymized.

- User data

**Table 4-1** User data

Field	Data Type	Description	Value
user_id	int	User ID	Anonymized
age	int	Age group	-1 indicates that the user age is unknown.

Field	Data Type	Description	Value
gender	int	Gender	<ul style="list-style-type: none"> <li>0: Male</li> <li>1: Female</li> <li>2: Confidential</li> </ul>
rank	Int	User level	Sequenced list of user level. The higher the user level, the larger the number.
register_time	string	User registration date	Unit: day

- Product data

**Table 4-2** Product data

Field	Data Type	Description	Value
product_id	int	Product No.	Anonymized
a1	int	Attribute 1	Enumerated value. The value <b>-1</b> indicates unknown.
a2	int	Attribute 2	Enumerated value. The value <b>-1</b> indicates unknown.
a3	int	Attribute 3	Enumerated value. The value <b>-1</b> indicates unknown.
category	int	Category ID	Anonymized
brand	int	Brand ID	Anonymized

- Comment data

**Table 4-3** Comment data

Field	Data Type	Description	Value
deadline	string	End time	Unit: day
product_id	int	Product No.	Anonymized

Field	Data Type	Description	Value
comment_num	int	Segments of accumulated comment count	<ul style="list-style-type: none"> <li>● 0: No comment</li> <li>● 1: One comment</li> <li>● 2: 2 to 10 comments</li> <li>● 3: 11-50 comments</li> <li>● 4: More than 50 comments</li> </ul>
has_bad_comment	int	Whether there is negative feedback.	0: No; 1: Yes.
bad_comment_rate	float	Dissatisfaction rate	Proportion of the negative feedback.

- Action data

**Table 4-4** Action data

Field	Data Type	Description	Value
user_id	int	User ID	Anonymized
product_id	int	Product No.	Anonymized
time	string	Time of action	-
model_id	string	Module ID	Anonymized
type	string	<ul style="list-style-type: none"> <li>● Browse (refers to the offering details page)</li> <li>● Add to cart</li> <li>● Remove from cart</li> <li>● Place an order</li> <li>● Follow</li> <li>● Click</li> </ul>	-

## Step 1: Uploading Data


Upload the data to OBS for data analysis using DLI.

1. Download OBS Browser+. For details about the download address, see [Object Storage Service Tool Guide](#).
2. Install OBS Browser+. For details about the installation procedure, see [Object Storage Service Tool Guide](#).



3. Log in to OBS Browser+. OBS Browser+ supports two login modes: AK login (using access keys) or authorization code login. For details about the login procedure, see [Object Storage Service Tool Guide](#).
4. Upload data using the OBS Browser+.  
On the OBS Browser+ page, click **Create Bucket**. Select a region and enter a bucket name (for example, **DLI-demo**). After the bucket is created, return to the bucket list and click **DLI-demo**. OBS Browser+ supports upload by dragging. You can drag one or more files or folders from a local path to the object list of a bucket or a parallel file system on OBS Browser+. You can even drag a file or folder directly to a specified folder on OBS Browser+. Obtain the test data by downloading the [Best\\_Practice\\_04.zip](#) file, decompressing it, and uploading the **Data** folder to the root directory of the OBS bucket. The test data directory is as follows:
  - **data/JData\_User**: Data in the **user** table
  - **data/JData\_Product**: Data in the **product** table
  - **data/JData\_Product/JData\_Comment**: Data in the **comment** table
  - **data/JData\_Action**: Data the **action** table

## Step 2: Analyzing Data

1. Creating a Database and a Table
  - a. On the top menu bar of the portal page, choose **Products > Analytics > Data Lake Insight (DLI)**.
  - b. Create a demo database. On the DLI console, choose **Job Management > SQL Jobs**. Click the created job on the displayed page to go to the **SQL Editor** page.
  - c. In the left pane of the SQL Editor, select the **Databases** tab and click  to create the **demo** database. For details, see [Figure 4-1](#).

**Figure 4-1** Creating a database

### Create Database

You can create 883 more databases. [Increase quota.](#)

\* Database Name

\* Enterprise Project  [Create Enterprise Project](#)

Description

Tags

It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. [View predefined tags](#)

You can add 10 more tags.

**NOTE**

The **default** database is a built-in database. You cannot create a database named **default**.

- d. Choose the **demo** database, and enter the following SQL statement in the editing box:

```
create table user(
  user_id int,
  age int,
  gender int,
  rank int,
  register_time string
) USING csv OPTIONS (path "obs://DLI-demo/data/JData_User")
```

**NOTE**

The file path in the preceding SQL statement is the actual OBS path for storing data.

- e. Click **Execute** to create the user information table user.
- f. Create the **product**, **comment**, and **action** tables in the same way.

- Product data

```
create table product(
  product_id int,
  a1 int,
  a2 int,
  a3 int,
  category int,
  brand int
) USING csv OPTIONS (path "obs://DLI-demo/data/JData_Product")
```

- **Comment table**

```
create table comment(
  deadline string,
  product_id int,
  comment_num int,
  has_bad_comment int,
  bad_comment_rate float
) USING csv OPTIONS (path "obs://DLI-demo/data/JData_Comment")
```

- **Action table**

```
create table action(
  user_id int,
  product_id int,
  time string,
  model_id string,
  type string
) USING csv OPTIONS (path "obs://DLI-demo/data/JData_Action");
```

2. Querying Data

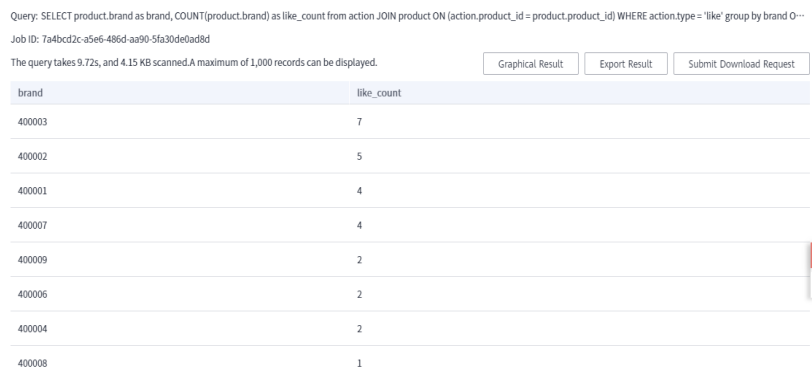
You can save common query statements as templates on the **Template Management** page for later use. For details, see [SQL Template Management](#) in *Data Lake Insight User Guide*.


- Top 10 products with the most likes
  - i. Run the following SQL statement to analyze the top 10 products with the most likes.

```
SELECT
  product.brand as brand,
  COUNT(product.brand) as like_count
from
  action
  JOIN product ON (action.product_id = product.product_id)
WHERE
  action.type = 'like'
group by
  brand
ORDER BY like_count desc
limit
  10
```

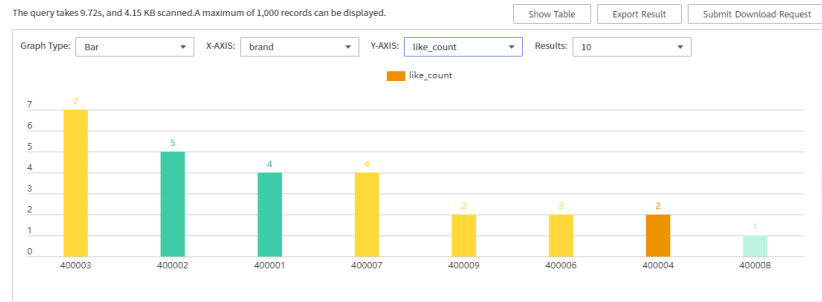
- ii. Click **Execute**. The execution results are displayed, as shown in [Figure 4-2](#).

**Figure 4-2** Querying results



- iii. Click  to view the result in a chart.

**Figure 4-3** Graphical results



- Top 10 worst-rated products
  - i. Run the following SQL statement to analyze the top 10 worst-rated products:

```
SELECT
DISTINCT product_id,
comment_num,
bad_comment_rate
from
comment
where
comment_num > 3
order by
bad_comment_rate desc
limit
10
```

- ii. Click **Execute**. The execution results are displayed, as shown in [Figure 4-4](#).

**Figure 4-4** Querying results


Executed successfully

Query: SELECT DISTINCT product\_id, comment\_num, bad\_comment\_rate from comment where comment\_num > 3 order by bad\_comment\_rate desc limit 10  
Job ID: 5b7457f5-42f2-4b54-b757-6afd70b0fb2a

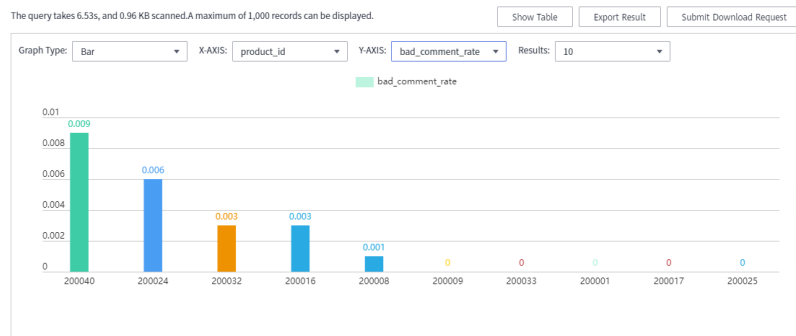
The query takes 6.53s, and 0.96 KB scanned. A maximum of 1,000 records can be displayed.

Graphical Result   Export Result   Submit Download Request

product_id	comment_num	bad_comment_rate
200040	4	0.009
200024	4	0.006
200032	4	0.003
200016	4	0.003
200008	4	0.001
200009	4	0
200033	4	0
200001	4	0
200017	4	0
200025	4	0

- iii. Click  to view the result in a chart.

**Figure 4-5** Graphical result



You can also analyze data for age distribution, gender ratio, offering evaluation, purchase number, and browsing statistics of users.

# 5 Analyzing Billing Consumption Data Using DLI

---

## Scenario

You can analyze DLI billing data (account information has been masked) on the big data analysis platform of DLI, find possible optimization, and figure out some measures to reduce costs for using DLI.

## Analysis Process

Perform the following steps to analyze billing data and reduce costs:

**Step 1: Obtaining Consumption Data.** Obtain billing data of an account.

**Step 2: Analyzing Billing Data and Reducing Costs.** Analyze the consumption data, find the resources or users with high expenditure, and provide optimization measures to reduce cost.

## Resources and Costs

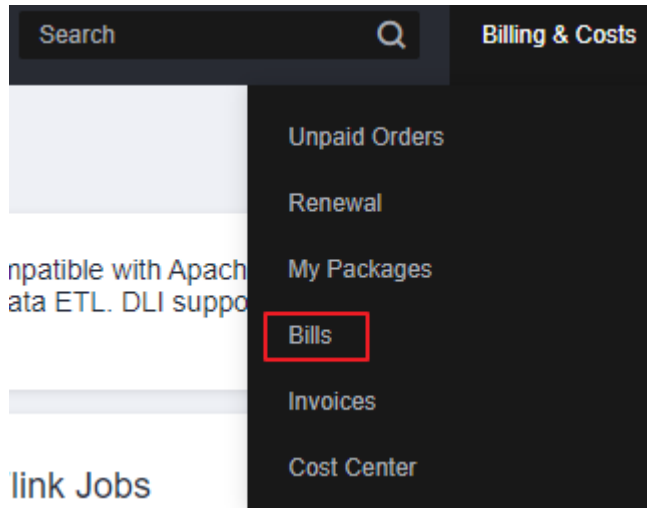
**Table 5-1** Resource planning and costs

Resource	Description	Cost
DLI	DLI is a big data analytics platform on Huawei Cloud. You are billed for using storage and compute resources. DLI supports three billing modes: yearly/monthly, package, and pay-per-use.	<p>You can run SQL jobs, Flink jobs, and Spark jobs on DLI.</p> <p>For SQL jobs, you are billed for both storage and compute resources. Compute resources can be billed based on a yearly/monthly basis or pay-per-use.</p> <ul style="list-style-type: none"> <li>• If you choose the yearly/monthly billing mode, fees are deducted based on the subscription period. This billing mode is recommended for its preferential price and exclusive compute resources within the subscription period.</li> <li>• In pay-per-use mode, fees are deducted by hour. You can choose either billing by CUH or by the amount of data scanned. Billing by CUH is recommended, for you can have exclusive resources and clear costing. In addition, you can purchase and use packages. <ul style="list-style-type: none"> <li>– Billing for CUH used = Number of CUs x Usage duration x Unit price. The unit of the usage duration is hour. If the duration is less than one hour, it is rounded to one hour.</li> <li>– Billing for the amount of data scanned = Amount of data scanned during SQL statement execution x Unit price. If a computing task times out or fails, no fee is charged for the task.</li> </ul> </li> <li>• For Flink and Spark jobs, you will be billed for compute resources only. The billing rules are same to those of SQL jobs.</li> </ul> <p>For details, see <a href="#">Price Calculator</a>.</p>

## Step 1: Obtaining Consumption Data

1. Obtain billing details.
  - a. Log in to the DLI console.
  - b. Click **Billing & Costs** on the upper right corner of the page. Choose **Bills**.

Figure 5-1 Bills



- c. On the **Dashboard** page of the **Billing Center**, click **Expenditure Details**. On the displayed page, set **Data Type** to **Usage Type** and **Data Period** to **Details**. Set time to the billing cycle you want.

In the title row of the displayed table, set **Service Type** to **Data Lake Insight (DLI)** and **Resource Type** to **DLI cuh**. Click **Export**. On the **Export** page, configure **Export Content** and **Period** as you need, and click **Export**. The **Export History** page is displayed.

Figure 5-2 DLI Bills

The image shows the 'Expenditure Details' page. At the top, there are filters for 'Data Type' (set to 'Usage Type', highlighted with a red box) and 'Data Period' (set to 'Details', highlighted with a red box). Below the filters is a table with columns: Billing, Enterp..., Account..., Service T..., Resource Type, Billing..., Expenditure Time, Order No./Transaction..., Bill Type, Transaction Time, Resource Na..., Resource Tag, Specifications, Region, AZ, and I. The table contains several rows of data for April 2022, showing resource usage and billing details.

- d. On the **Export History** page, wait until the file status changes to **Successful**. Click **Download**.

## Step 2: Analyzing Billing Data and Reducing Costs

1. Analyze billing details.
  - a. Upload the billing details downloaded in [Step 1: Obtaining Consumption Data](#) to the created OBS bucket.



- b. Create a table on DLI.
  - i. Log in to the DLI console. In the navigation pane, choose **SQL Editor**. Select **spark** for **Engine**, and select the queue and database. In this example, the default queue and database are used.
  - ii. The downloaded file contains information such as time and usage. Create a table on DLI based on these table headers. For details, see the following example.

```
CREATE TABLE `spending` (  
  account_period string,  
  EnterpriseProject string,  
  EnterpriseProjectID string,  
  accountID string,  
  product_type_code string,  
  product_type string,  
  product_code string,  
  product_name string,  
  product_id string,  
  mode string,  
  time1 string,  
  use_start string,  
  use_end string,  
  orderid string,  
  ordertime string,  
  resource_type string,  
  resource_id string,  
  resource_name string,  
  tag string,  
  skuid string,  
  `c22name` STRING,  
  `c23name` STRING,  
  `c24name` STRING,  
  `c25name` STRING,  
  `c26name` STRING,  
  `c27name` STRING,  
  `c28name` STRING,  
  `c29name` STRING,  
  size STRING,  
  `c31name` STRING,  
  `c32name` STRING,  
  `c33name` STRING,  
  `c34name` STRING,  
  `c35name` STRING,  
  `amount` STRING,  
  `c37name` STRING,  
  `c38name` STRING,  
  `c39name` STRING,  
  `c40name` STRING,  
  `c41name` STRING,  
  `c42name` STRING,  
  `c43name` STRING,  
  `c44name` STRING,  
  `c45name` STRING,  
  `c46name` STRING,  
  `c47name` STRING,  
  `c48name` STRING,  
  `c49name` STRING,  
  `c50name` STRING,  
  `c51name` STRING,  
  `c52name` STRING,  
  `c53name` STRING,  
  `c54name` STRING  
) USING csv options (  
  path 'obs://xxx/Spending(BYTransaction)_20200501_20200531.csv',  
  header true)
```

- c. Query **resource\_id** and **resource\_name** with the highest amount within the period.

The following statement shows the amount charged for using the SQL and Flink queues.

```
select resource_id, resource_name, sum(size)
as usage, sum(amount)
as sum_amount
from spending
group by resource_id, resource_name
order by sum_amount desc
```

**Figure 5-3** Query results

resource_id	resource_name	usage	sum_amount
d91d4616-b10c-471a-820d-e676e6c5f4b4	sql	5264	1842.3090000000000000
8163c227-89ca-48ac-aab5-38c5734ae425	flink	5264	1842.3090000000000000
90d0736b-f8ca-4bfb-b3a7-0e391ef700db	null	48	14.309000000000000000
d53a12ff-c0af-4ba1-bbct-858af463661c	dlitest	32	11.2
f82d5ef5-eb5f-4e0f-b9d6-9ca91e020009	test	16	5.6

- d. Run the following statements to analyze the usage periods of SQL and Flink resources:

```
select * from spending where resource_id = 'd91d4616-b10c-471a-820d-e676e6c5f4b4' order by
ordertime
```

The SQL queue was billed each hour from May 14 2020 17:00:00 GMT +08:00 to May 28, 2020 10:00:00 GMT+08:00.

Similarly, the Flink queue was continuously used from May 14, 2020 17:00:00 GMT+08:00 to May 28 2020 10:00:00 GMT+08:00.

2. Suggestion for reducing the cost

You can change the SQL and Flink queues to yearly/monthly queues for lower costs. If you are sure about the number of CUHs required for a job, you can purchase a package to reduce the cost.

DLI helps you to analyze billing details of your enterprise to quickly find the unreasonable expenses and control costs. You can also use DLI to reduce your cost on Huawei Cloud.

# 6 Analyzing Real-time E-Commerce Business Data Using DLI

## Scenario

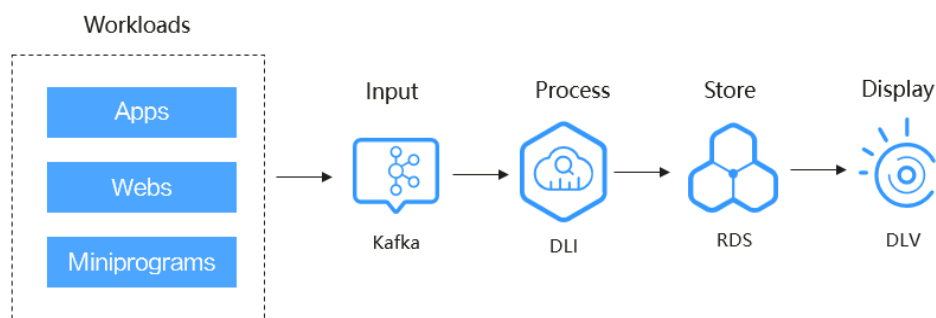
Online shopping is very popular for its convenience and flexibility. e-Commerce platform can be accessed via an array of methods, such as visiting the websites, using shopping apps, and accessing through mini-programs. A large volume of statistics data such as the real-time access volume, number of orders, and number of visitors needs to be collected and analyzed on each e-commerce platform every day. These data needs to be displayed in an intuitive way and updated in real time to help managers learn about data changes in a timely manner and adjust marketing policies accordingly. How can we efficiently and quickly collect statistics based on these metrics?

Assume the order information of each offering is written into Kafka in real time. The information includes the order ID, channel (websites or apps), order creation time, amount, actual payment amount after discount, payment time, user ID, username, and region ID. We need to collect statistics on such information based on metrics of each sales channel in real time, store the statistics in a database, and display the statistics on screens.

## Solution Overview

The following figure gives you an overview to user DLI Flink to analyze and process real-time e-commerce business data and sales data of all channels.

**Figure 6-1** Solution overview



## Process

To analyze real-time e-commerce data with DLI Flink, perform the following steps:

**Step 1: Creating Resources.** Create resources required for creating jobs belong to your account, including VPC, DMS, DLI, and RDS.

**Step 2: Obtaining the DMS Connection Address and Creating a Topic.** Obtain the connection address of the DMS Kafka instance and create a DMS topic.

**Step 3: Creating an RDS Database Table.** Obtain the private IP address of the RDS DB instance and log in to the instance to create an RDS database and MySQL table.

**Step 4: Creating an Enhanced Datasource Connection.** Create an enhanced datasource connection for the queue and test the connectivity between the queue and the RDS instance and the queue and the DMS instance, respectively.

**Step 5: Creating and Submitting a Flink Job.** Create a DLI Flink OpenSource SQL job and run it.

**Step 6: Querying the Result.** Query the Flink job results and display the results on a screen in DLV.

## Solution Advantages

- Cross-source analysis: You can perform association analysis on sales summary data of each channel stored in OBS. There is no need for data migration.
- SQL only: DLI has interconnected with multiple data sources. You can create tables using SQL statements to complete data source mapping.

## Resource Planning and Costs

**Table 6-1** Resource planning and costs

Resource	Description	Cost
OBS	You need to create an OBS bucket and upload data to OBS for data analysis using DLI.	<p>You will be charged for using the following OBS resources:</p> <ul style="list-style-type: none"><li>• <b>Storage Fee</b> for storing static website files in OBS.</li><li>• <b>Request Fee</b> for accessing static website files stored in OBS.</li><li>• <b>Traffic Fee</b> for using a custom domain name to access OBS over the public network.</li></ul> <p>The actual fee depends on the size of the stored file, the number of user access requests, and the traffic volume. Estimate the fee based on your service requirements.</p>

Resource	Description	Cost
DLI	Before creating a SQL job, you need to purchase a queue. When using queue resources, you are billed based on the CUH of the queue.	For example, if you purchase a pay-per-use queue, you will be billed based on the number of CUHs used by the queue. Usage is billed by the hour, with any usage less than one hour being rounded up to one hour. The number of hours is calculated on the hour. CUH pay-per-use billing = Unit price x Number of CUs x Number of hours.
VPC	You can customize subnets, security groups, network ACLs, and assign EIPs and bandwidths.	The VPC service is free of charge. EIPs are required if your resources need to access the Internet. EIP supports two billing modes: pay-per-use and yearly/monthly. For details, see <a href="#">VPC Billing</a> .
DMS Kafka	Kafka provides premium instances with computing, storage, and exclusive bandwidth resources.	Kafka supports two billing modes: pay-per-use and yearly/monthly. Billing items include Kafka instances and Kafka disk storage space. For details, see <a href="#">DMS for Kafka Billing</a> .
RDS MySQL	RDS for MySQL provides online cloud database services.	You are billed for RDS DB instances, database storage, and backup storage (optional). For details, see <a href="#">RDS Billing</a> .
DLV	DLV adapts to a wide range of on-premise and cloud data sources, and provides diverse visualized components for you to quickly customize your data screens.	If you use the DLV service, you will be charged for the purchased yearly/monthly DLV package.

## Example Data

- Order details wide table

Field	Data Type	Description
order_id	<i>string</i>	Order ID.

Field	Data Type	Description
order_channel	<i>string</i>	Order channel (websites or apps)
order_time	<i>string</i>	Time
pay_amount	<i>double</i>	Order amount
real_pay	<i>double</i>	Actual amount paid
pay_time	<i>string</i>	Payment time
user_id	<i>string</i>	User ID
user_name	<i>string</i>	Username
area_id	<i>string</i>	Region ID

- Result table: real-time statistics of the total sales amount in each channel

Field	Data Type	Description
begin_time	<i>varchar(32)</i>	Start time for collecting statistics on metrics
channel_code	<i>varchar(32)</i>	Channel code
channel_name	<i>varchar(32)</i>	Channel
cur_gmv	<i>double</i>	Gross merchandises value (GMV) of the day
cur_order_user_count	<i>bigint</i>	Number of users who settled the payment in the day
cur_order_count	<i>bigint</i>	Number of orders paid on the day
last_pay_time	<i>varchar(32)</i>	Latest settlement time
flink_current_time	<i>varchar(32)</i>	Flink data processing time

## Step 1: Creating Resources

Create VPC, DMS, RDS, DLI, and DLV resources listed in [Table 6-2](#).

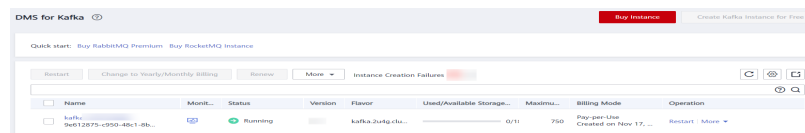
**Table 6-2** Cloud resources required

Resource	Description	Instructions
VPC	A VPC manages network resources on the cloud. <b>The network planning is described as follows:</b> <ul style="list-style-type: none"> <li>The VPCs specified for the Kafka and MySQL instances must be the same.</li> <li>The VPC network segment where the Kafka and MySQL instances belong cannot conflict with that of the DLI queue.</li> </ul>	<a href="#">Creating a VPC and Subnet</a>
DMS Kafka	In this example, the DMS for Kafka instance is the data source.	<a href="#">Getting Started with DMS for Kafka</a>
RDS MySQL	In this example, an RDS for MySQL instance provides the cloud database service.	<a href="#">Buying an RDS for MySQL DB Instance</a>
DLI	DLI provides real-time data analysis. Create a general-purpose queue that uses dedicated resources in yearly/monthly or pay-per-use billing mode. Otherwise, an enhanced network connection cannot be created.	<a href="#">Creating a Queue</a>
DLV	DLV displays the result data processed by the DLI queue in real time.	<a href="#">Creating Screens</a>

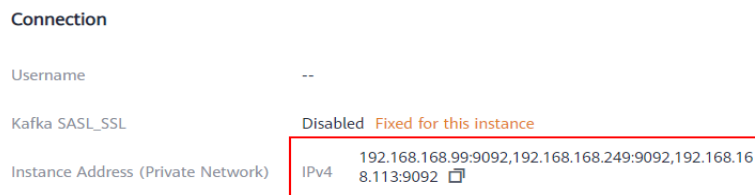
## Step 2: Obtaining the DMS Connection Address and Creating a Topic

1. Hover the mouse on the **Service List** icon and choose **Distributed Message Service** in **Application**. The DMS console is displayed. On the **DMS for Kafka** page, locate the Kafka instance you have created.

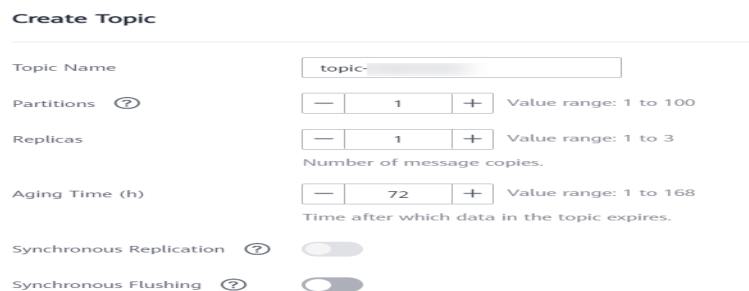
**Figure 6-2** Kafka instances



2. The instance details page is displayed. Obtain the **Instance Address (Private Network)** in the **Connection** pane.

**Figure 6-3** Connection address

3. Create a topic and name it **trade\_order\_detail\_info**.

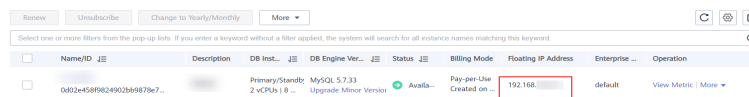
**Figure 6-4** Creating a topic

Configure the required topic parameters as follows:

- **Partitions:** Set it to **1**.
- **Replicas:** Set it to **1**.
- **Aging Time:** Set it to **72** hours.
- **Synchronous Flushing:** Disable this function.

### Step 3: Creating an RDS Database Table

1. Log in to the console, hover your mouse over the service list icon and choose **Relational Database Service** in **Databases**. The RDS console is displayed. On the **Instances** page, locate the created DB instance and view its floating IP address.

**Figure 6-5** Viewing the floating IP address

2. Click **More > Log In** in the **Operation** column. On the displayed page, enter the username and password for logging in to the instance and click **Test Connection**. After **Connection is successful** is displayed, click **Log In**.



**Figure 6-6** Logging in to an Instance

Instance Login Information

DB Instance Name: no\_delete      DB Engine Version: MySQL 5.7

\* Login Username:

\* Password:       Test Connection

Remember Password      Your password will be encrypted and stored securely.

Description:

Collect Metadata Periodically:       If not enabled, DAS can query the real-time structure information only from databases, which may affect the real-time performance of databases.

Show Executed SQL Statements:       If not enabled, the executed SQL statements cannot be viewed, and you need to input each SQL statement manually.

3. Click **Create Database**. In the displayed dialog box, enter database name **dli-demo**. Then, click **OK**.

**Figure 6-7** Creating a database

Create Database

\* Name:

Only user databases can be created

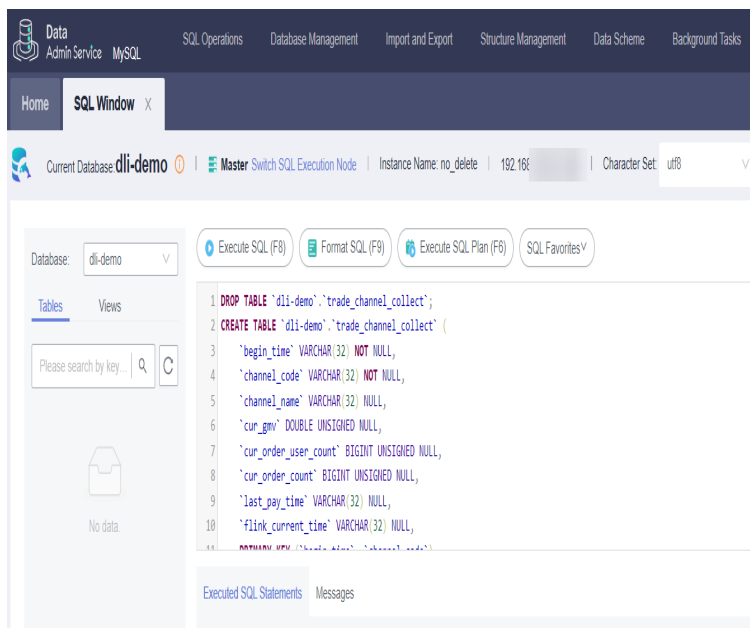
Character Set:

4. Choose **SQL Operation > SQL Query** and run the following SQL statement to create a MySQL table for test (**Example Data** describes the fields):

```
DROP TABLE `dli-demo`.`trade_channel_collect`;
CREATE TABLE `dli-demo`.`trade_channel_collect` (
  `begin_time` VARCHAR(32) NOT NULL,
  `channel_code` VARCHAR(32) NOT NULL,
  `channel_name` VARCHAR(32) NULL,
  `cur_gmv` DOUBLE UNSIGNED NULL,
  `cur_order_user_count` BIGINT UNSIGNED NULL,
  `cur_order_count` BIGINT UNSIGNED NULL,
  `last_pay_time` VARCHAR(32) NULL,
  `flink_current_time` VARCHAR(32) NULL,
  PRIMARY KEY (`begin_time`, `channel_code`)
) ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_general_ci
COMMENT = 'Real-time statistics on the total sales amount of each channel';
```

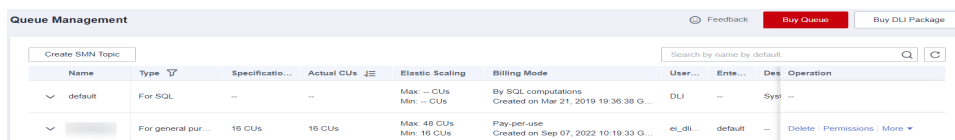
Figure 6-8 Creating a table



### Step 4: Creating an Enhanced Datasource Connection

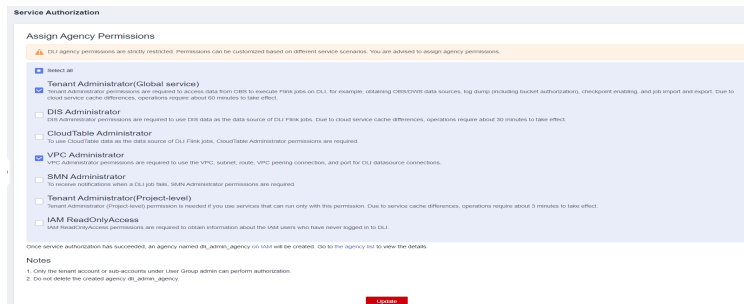
1. On the management console, hover the mouse on the service list icon and choose **Analytics > Data Lake Insight**. The DLI management console is displayed. Choose **Resources > Queue Management** to query the created DLI queue.

Figure 6-9 Queue list



2. In the navigation pane of the DLI management console, choose **Global Configuration > Service Authorization**. On the displayed page, select **VPC Administrator**, and click **Update** to grant the DLI user the permission to access VPC resources. The permission is used to create a VPC peering connection.

Figure 6-10 Updating agency permissions



3. Choose **Datasource Connections**. On the displayed **Enhanced** tab, click **Create**. Configure the following parameters, and click **OK**.

- **Connection Name:** Enter a name.
- **Resource Pool:** Select the general-purpose queue you have created.
- **VPC:** Select the VPC where the Kafka and MySQL instances are.
- **Subnet:** Select the subnet where the Kafka and MySQL instances are.

Figure 6-11 Creating an enhanced datasource

**Create Enhanced Connection** ✕

After you create the enhanced datasource connection, the system will automatically create a VPC peering connection and required routes.

★ Connection Name

Resource Pool

★ VPC

★ Subnet

Host Information

Tags   
 It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. [View predefined tags](#) ↻  
 To add a tag, enter a tag key and a tag value below.

The status of the created datasource connection is **Active** in the **Enhanced** tab.

Click the name of the datasource connection. On the details page, the connection status is **ACTIVE**.

Figure 6-12 Connection status

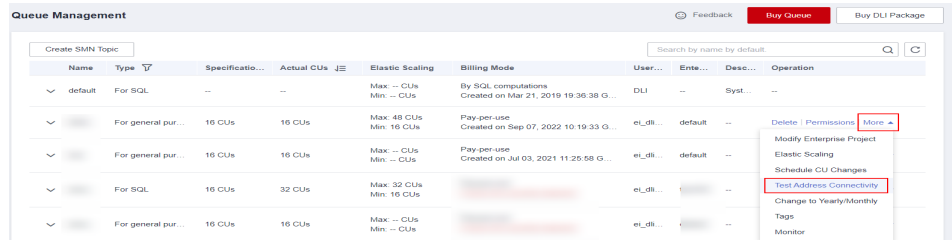
Datasource Connections		
Enhanced	Basic	Datasource Authentication
<input type="button" value="Create"/> The enhanced datasource connection supports queues created in		
Connection Name	Connection Status	
▼	<span style="color: green;">●</span> Active	

Figure 6-13 Details

VPC Peering ID	Resource Pool	Connection Status	Updated	Operation
▼ 5b6a1e43-ab62-4267-a3cc-191139aff0ee	auto_ep	<span style="color: green;">●</span> Active	Oct 13, 2022 19:48:11 GMT+08:00	Unbind Resource Pool

4. Test whether the queue can connect to RDS for MySQL and DMS for Kafka instances, respectively.
  - a. On the **Queue Management** page, locate the target queue. In the **Operation** column, click **More > Test Address Connectivity**.

**Figure 6-14** Testing address connectivity



- b. Enter the connection address of the DMS for Kafka instance and the private IP address of the RDS for MySQL instance to test the connectivity. If the test is successful, the DLI queue can connect to the Kafka and MySQL instances.

**Figure 6-15** Testing address connectivity

### Test Address Connectivity

Tests whether an address is reachable from a specified cluster. The address can be a domain name, an IP address, or a specified port.

\* Address

If the test fails, modify the security group rules of the VPC where the Kafka and MySQL instances are to allow DLI queue access on ports 9092 and 3306. You can obtain the network segment of the queue on its details page.

**Figure 6-16** VPC security group rules



## Step 5: Creating and Submitting a Flink Job

1. In the navigation pane on the left, choose **Job Management > Flink Jobs**. Click **Create Job**.
  - **Type:** Select **Flink OpenSource SQL**.
  - **Name:** Enter a name.

Figure 6-17 Creating a Flink Job

The screenshot shows a 'Create Job' dialog box with the following fields and options:

- Type:** Flink OpenSource SQL
- Name:** kafka-2-mysql
- Description:** Description
- Template Name:** --Select--
- Tags:** A section with a note: "It is recommended that you use TMS's predefined tag function to add the same tag to different cloud resources. View predefined tags". Below this is a text area for adding tags, and buttons for "Enter a tag key", "Enter a tag value", and "Add". A note at the bottom says "20 tags available for addition."

At the bottom of the dialog are two buttons: "OK" (highlighted in red) and "Cancel".

2. Click **OK**. The job editing page is displayed. The following is a simple SQL statement. You need to modify some parameter values based on the RDS and DMS instance information.

```

--*****_
-- Data source: trade_order_detail_info (order details wide table)
--*****_
create table trade_order_detail (
  order_id string, -- Order ID
  order_channel string, -- Channel
  order_time string, -- Order creation time
  pay_amount double, -- Order amount
  real_pay double, -- Actual payment amount
  pay_time string, -- Payment time
  user_id string, -- User ID
  user_name string, -- Username
  area_id string -- Region ID
) with (
  "connector.type" = "kafka",
  "connector.version" = "0.10",
  "connector.properties.bootstrap.servers" = "xxx:9092,xxx:9092,xxx:9092", -- Kafka connection
address
  "connector.properties.group.id" = "trade_order", -- Kafka groupID
  "connector.topic" = "trade_order_detail_info", -- Kafka topic
  "format.type" = "json",
  "connector.startup-mode" = "latest-offset"
);
--*****_
-- Result table: trade_channel_collect (real-time statistics on the total sales amount of each channel)
--*****_
create table trade_channel_collect(
  begin_time string, --Start time of statistics collection
  channel_code string, -- Channel ID
  channel_name string, -- Channel name
  cur_gmv double, -- GMV
  cur_order_user_count bigint, -- Number of payers
  cur_order_count bigint, -- Number of orders paid on the day
  last_pay_time string, -- Latest settlement time
  flink_current_time string,
  primary key (begin_time, channel_code) not enforced
) with (
  "connector.type" = "jdbc",
  "connector.url" = "jdbc:mysql://xxx:3306/xxx", -- MySQL connection address, in JDBC format
  "connector.table" = "xxx", -- MySQL table name
  "connector.driver" = "com.mysql.jdbc.Driver",
  'pwd_auth_name'= 'xxxx', -- Name of the datasource authentication of the password type created
on DLI. If datasource authentication is used, you do not need to set the username and password for
the job.

```

```
"connector.write.flush.max-rows" = "1000",
"connector.write.flush.interval" = "1s"
);
_*****_
-- Temporary intermediate table
_*****_
create view tmp_order_detail
as
select *
, case when t.order_channel not in ("webShop", "appShop", "miniAppShop") then "other"
  else t.order_channel end as channel_code -- Redefine channels. Only four enumeration values
are available: webShop, appShop, miniAppShop, and other.
, case when t.order_channel = "webShop" then _UTF16"Website"
  when t.order_channel = "appShop" then _UTF16"Shopping App"
  when t.order_channel = "miniAppShop" then _UTF16" Miniprogram"
  else _UTF16"Other" end as channel_name -- Channel name
from (
  select *
  , row_number() over(partition by order_id order by order_time desc ) as rn -- Remove duplicate
order data
  , concat(substr("2021-03-25 12:03:00", 1, 10), " 00:00:00") as begin_time
  , concat(substr("2021-03-25 12:03:00", 1, 10), " 23:59:59") as end_time
  from trade_order_detail
  where pay_time >= concat(substr("2021-03-25 12:03:00", 1, 10), " 00:00:00") --Obtain the data of
the current day. To accelerate running, 2021-03-25 12:03:00 is used to replace
cast(LOCALTIMESTAMP as string).
  and real_pay is not null
) t
where t.rn = 1;

-- Collect data statistics by channel.
insert into trade_channel_collect
select
  begin_time --Start time of statistics collection
  , channel_code
  , channel_name
  , cast(COALESCE(sum(real_pay), 0) as double) as cur_gmv -- GMV
  , count(distinct user_id) as cur_order_user_count -- Number of payers
  , count(1) as cur_order_count -- Number of orders paid on the day
  , max(pay_time) as last_pay_time -- Settlement time
  , cast(LOCALTIMESTAMP as string) as flink_current_time -- Current time of the flink task
from tmp_order_detail
where pay_time >= concat(substr("2021-03-25 12:03:00", 1, 10), " 00:00:00")
group by begin_time, channel_code, channel_name;
```

## NOTE

### Job logic

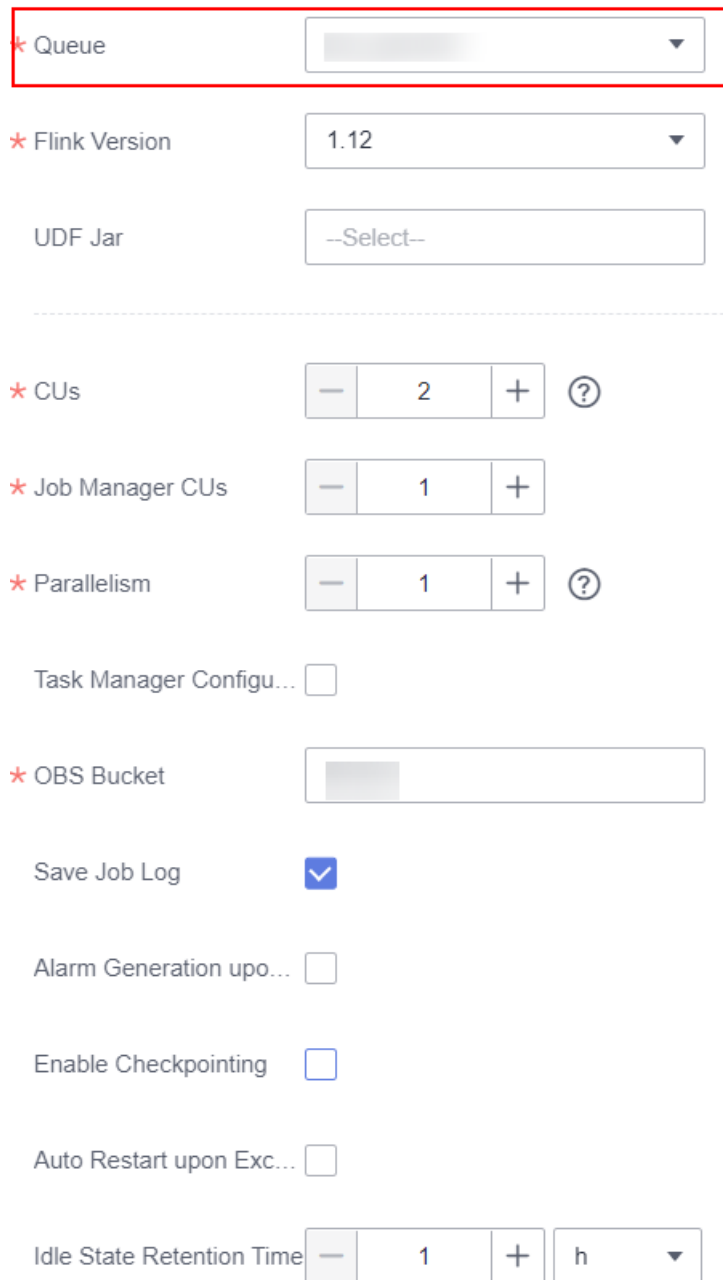
1. Create a Kafka source table to read consumption data from a specified Kafka topic.
2. Create a result table to write result data into MySQL through JDBC.
3. Implement the processing logic to collect statistics on each metric.

To simplify the final processing logic, create a view to preprocess the data.

1. Use over window condition and filters to remove duplicate data (the top N method is used). In addition, the built-in functions **concat** and **substr** are used to set 00:00:00 as the start time and 23:59:59 of the same day as the end time, and to collect statistics on orders paid later than 00:00:00 on the day. (To facilitate data simulation, replace **cast(LOCALTIMESTAMP as string)** with 2021-03-25 12:03:00.)
2. Based on the channels of the order data, the built-in condition function is used to set **channel\_code** and **channel\_name** to obtain the field information in the source table and the values of **begin\_time**, **end\_time**, **channel\_code**, and **channel\_name**.
4. Collect statistics on the required metrics, filter the data as required, and write the results to the result table.

3. Select the created general-purpose queue and submit the job.

**Figure 6-18** Flink OpenSource SQL Job



\* Queue

\* Flink Version

UDF Jar

---

\* CUs  ?

\* Job Manager CUs

\* Parallelism  ?

Task Manager Configu...

\* OBS Bucket

Save Job Log

Alarm Generation upo...

Enable Checkpointing

Auto Restart upon Exc...

Idle State Retention Time  h

4. Wait until the job status changes to **Running**. Click the job name to view the details.





- In the navigation pane on the left, choose **Job Management > Flink Jobs**, and click the job submitted in **3**. On the job details page, view the number of processed data records.

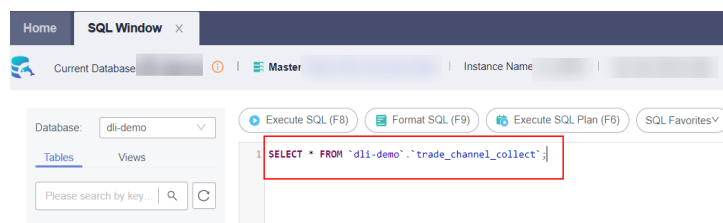
**Figure 6-21** Job details

Duration	Max C...	Task	Status	Back P...	Delay	Sent Records	Sent Bytes	Received Records	Received Bytes	Started	Ended
15d 14...	1	[Progress Bar]	Running	OK	--					Nov 15, 202...	--
15d 14...	1	[Progress Bar]	Running	OK	--					Nov 15, 202...	--
downk...	15d 14...	[Progress Bar]	Running	OK	53					Nov 15, 202...	--

## Step 6: Querying the Result

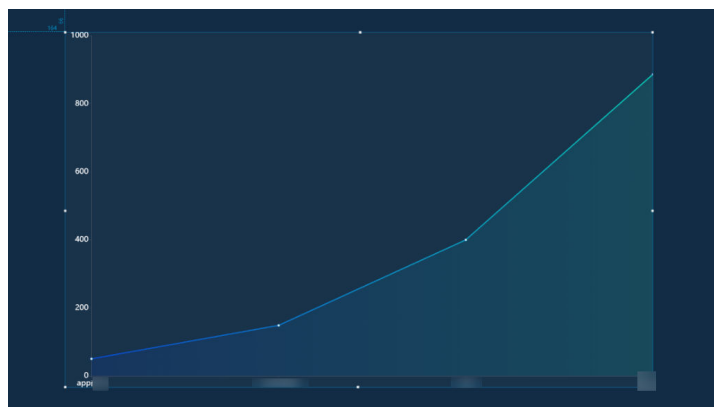
- Log in to the MySQL instance by referring to **2** and run the following SQL statement to query the result data processed by the Flink job:  
`SELECT * FROM `dli-demo`.`trade_channel_collect`;`

**Figure 6-22** Querying results



- Log in to the DLV console, configure a DLV screen, and run SQL statements to query data in the RDS for MySQL instance to display the data on the screen. For details, see [Editing Screens](#).

**Figure 6-23** DLV screen



# 7 Connecting BI Tools to DLI for Data Analysis

---

## 7.1 Overview

BI tools are powerful aids for data analysis, offering features such as data visualization, report generation, and dashboard creation.

By analyzing and processing data, DLI provides BI tools with standard, effective, and high-quality data for statistical analysis.

By connecting to DLI, BI tools can more flexibly access and analyze data, helping businesses make data-driven decisions quickly.

There are the following methods available to connect BI tools to DLI:

- DBeaver, DBT, and Yonghong BI can directly connect to DLI through the driver provided by DLI. This simplifies the configuration process, enabling users to directly use these powerful tools.
- Power BI, FineBI, Superset, Tableau, and Beeline can connect to DLI through Kyuubi. Kyuubi is a distributed SQL query engine that provides standard SQL APIs, making BI tools can interact with DLI through Kyuubi to query and analyze data.

### NOTE

The BI tools connect to DLI using DLI's SDK V2.

- Starting May 2024, new users can directly use DLI's SDK V2 without needing to whitelist it.
- For users who started using DLI before May 2024, to use this function, they must submit a service ticket to whitelist it.

## 7.2 Connecting DBeaver to DLI for Data Analysis and Query

DBeaver is a free, open source, and visual database management tool that you can use to manage various types of databases, including viewing the database

structure, executing SQL queries and scripts, and browsing and exporting data. This section describes how to connect DBeaver to DLI.

## Preparations

- **Toolkits**
  - **DLI JDBC driver:**  
Download the JDBC driver **huaweicloud-dli-jdbc-xxx-dependencies.jar** from the DLI management console.
  - **DBeaver client installation package:**  
The DBeaver official website provides client installation packages for different OSs. **Download** the required DBeaver client installation package and install it on the local PC. DBeaver 24.0.3 is recommended.
- **Connection information:**

**Table 7-1** Connection information

Item	Description	How to Obtain
DLI's AK/SK	AK/SK-based authentication refers to the use of an AK/SK pair to sign requests for identity authentication.	<a href="#">Obtaining an AK/SK</a>
DLI's endpoint address	Endpoint of a cloud service in a region.	<a href="#">Obtaining an Endpoint</a>
DLI's project ID	Project ID, which is used for resource isolation.	<a href="#">Obtaining a Project ID</a>
DLI's region information	DLI's region information	<a href="#">Regions and Endpoints</a>

## Step 1: Create a DLI JDBC Driver on DBeaver

1. In DBeaver, choose **Database > Driver Manager**.  
Use the driver class to load DLI's JDBC driver. Make sure to use the JAR file **huaweicloud-dli-jdbc-2.1.1-jar-with-dependencies.jar**.
2. Click **New**.
3. In the displayed dialog box, set driver parameters and click **OK**.  
Set driver parameters based on [Table 7-2](#).

**Table 7-2** Driver parameters

Parameter	Description
Driver Name	Enter a name that is easy to identify, for example, <b>GaussDB Driver</b> .
Driver Type	Set it to <b>Generic</b> .

Parameter	Description
Class Name	Class name
URL Template	Format of DLI's JDBC driver. For how to configure DLI's JDBC driver, see <a href="#">Format of DLI's JDBC driver</a> and <a href="#">Example configuration of DLI's JDBC driver</a> . jdbc:dli://<endPoint>/projectId?<key1>=<val1>;<key2>=<val2>...
Default Port	Port of the database to connect.
Default Database	Name of the database to connect.
Default User	Account name, which is <b>root</b> by default.

– **Format of DLI's JDBC driver**

jdbc:dli://<endPoint>/projectId?<key1>=<val1>;<key2>=<val2>...

? is followed by other configuration items, with each item listed in the "key=value" format and multiple items are separated by semicolons (;).

– **Example configuration of DLI's JDBC driver**

jdbc:dli://dli.ap-southeast-2.myhuaweicloud.com/0b33ea2a7e0010802fe4c009bb05076d?databasename=tpch;queueName=auto;accesskey=XXXX;secretkey=XXXXX;regionname=ap-southeast-2;enginetype=trino;catalog=lfcatalog

[Table 7-3](#) and [Table 7-4](#) describe the parameters.

**Table 7-3** Driver configuration parameters

Parameter	Description	How to Obtain
endPoint	Endpoint of a cloud service in a region.	<a href="#">Obtaining an AK/SK</a>
projectId	ID of the project where DLI resources are.	<a href="#">Obtaining an Endpoint</a>
<key1>=<val1>	? is followed by other configuration items, with each item listed in the "key=value" format and multiple items are separated by semicolons (;).	<a href="#">Table 7-4</a>

**Table 7-4** key=value parameter description

Parameter	Description	Mandatory	Example Value
queueName	DLI queue name.	Yes	dli_test

Parameter	Description	Mandatory	Example Value
databasename	Database name.	Yes	tpch
accesskey and secretkey	AK/SK that acts as the authentication key. Set them if <b>authenticationmode</b> is <b>aksk</b> .	Yes	accesskey=your-access-key secretkey=your-secret-key
regionname	Region name of DLI Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	-
charset	Encoding method used by JDBC. The default value is <b>UTF-8</b> .	No	-
enginetype	The options are <b>spark</b> and <b>trino</b> . Spark or Hetu queue (with <b>spark</b> as default).	No	trino
catalog	Metadata catalog name. It is mandatory when a LakeFormation catalog is used. In this case, it indicates the name of the LakeFormation catalog used.	No	lfcatalog

4. On the **Libraries** tab, click **Add File** and add **dli-jdbc-xxx-dependencies.jar** in **1**.
5. After the file is added, the driver class is empty. Click **Find Class** to set the driver class. The identified driver class must be the same as the class name specified on the **Settings** tab.
6. Click **OK** to complete the driver settings.

## Step 2: Test the Connection to the Database

1. In the menu bar of the DBeaver client, choose **Database > New Database Connection** and select the data driver created in **Step 1: Create a DLI JDBC Driver on DBeaver**.
2. Click **Finish** to connect to DLI. You can view information about the connected database in the **Database Navigator** view toolbar.
3. Query data on DLI through the new connection.

### Step 3: Compile SQL Queries on DBeaver

Once DBeaver is connected to DLI, you can compile SQL queries on DBeaver.

1. Select a database object in **Database Navigator** on the left and write SQL statements in the SQL editor in the middle.
2. Once a query is compiled, execute it by clicking the running button on the toolbar, which is usually represented by a green play icon.
3. After the query is executed, the result is displayed in the data grid below the SQL editor.

## 7.3 Connecting DBT to DLI for Data Scheduling and Analysis

Data Build Tool (DBT) is an open source data modeling and conversion tool that runs in Python environments. Connecting DBT to DLI can define and execute SQL transformations, supporting the entire data lifecycle management from integration to analysis. It is suitable for large-scale data analysis projects and complex data analysis scenarios.

This section describes how to connect DBT to DLI.

### Preparations

- **Environment requirements**  
Make sure that your system environment meets the following requirements:
  - Operating system: Windows or Linux
  - Ensure that Python is installed as DBT is Python-based.  
Python version: Python 3.8 or later. Python 3.8 is recommended.
- **Obtaining the dli-dbt driver package**  
Download the JDBC driver **huaweicloud-dli-jdbc-xxx-dependencies.jar** from the DLI management console.
- **Connection information:**

**Table 7-5** Connection information

Item	Description	How to Obtain
DLI's AK/SK	AK/SK-based authentication refers to the use of an AK/SK pair to sign requests for identity authentication.	<a href="#">Obtaining an AK/SK</a>
DLI's endpoint address	Endpoint of a cloud service in a region.	<a href="#">Obtaining an Endpoint</a>
DLI's project ID	Project ID, which is used for resource isolation.	<a href="#">Obtaining a Project ID</a>

Item	Description	How to Obtain
DLI's region information	DLI's region information	<a href="#">Regions and Endpoints</a>

## Step 1: Create a DBT Environment

### 1. Install dbt-core.

Install dbt-core of the recommended version.

```
pip install dbt-core==1.7.9
```

#### NOTE

pip is a package management tool for Python that is typically installed alongside Python.

If pip is not installed, install it using Python's built-in **ensurepip** module.

```
python -m ensurepip
```

### 2. Install dli-sdk-python.

Run the following installation command:

```
python setup.py install
```

### 3. Installing dli-dbt

Download the **dli-dbt** driver from the DLI management console.

Run the following installation command:

```
python setup.py install
```

Run the following command to check whether dbt is successfully installed:

```
dbt --version
```

## Step 2: Connect DBT to DLI

Configure the **profiles.yml** file to store information about the connection between DBT and DLI.

Find **.dbt** in the home directory of the server where DBT is installed and create or edit the **profiles.yml** file.

For example, in Windows, the path may be **C:\Users\Username\.dbt\profiles.yml**.

The file must contain the configuration of the connection between DBT and DLI. For example:

```
profiles:
- name: dbt_dli
  target: dev
  outputs:
  dev:
    type: dli
    region: your-region-name
    project_id: your-project_id
    access_id: your-ak
    secret_key: your-sk
    queue: your-queue-name
    database: your-dli-database
    schema: your-dli-schema
```

**Table 7-6** Parameters for connecting DBT to DLI

Parameter	Mandatory	Description	Example Value
type	Yes	Data source type. Set it to <b>dli</b> in this example.	dli
region	Yes	Region name.	ap-southeast-2
project_id	Yes	ID of the project where DLI resources are.	0b33ea2a7e0010802fe4c009bb05076d
access_id and secret_key	Yes	AK/SK that acts as the authentication key.	-
queue	Yes	DLI queue name.	dli_test
database	Yes	Data directory name, with <b>dli</b> as default. If LakeFormation metadata is used, enter the data directory name.	dli
schema	Yes	Name of the DLI database used to submit jobs.	tpch

### Step 3: Use DBT to Submit a Job to DLI

1. **Initialize a DBT project.**

Run the following command in an empty directory to initialize a DBT project:

```
dbt init
```

2. **Configure the dbt\_project.yml file.**

Create or edit the **dbt\_project.yml** file in the root directory of the project.

Configure the project by referring to [dbt\\_project.yml](#).

Ensure that the data source name defined in **profiles.yml** of the project has been set in the profile file in [Step 2: Connect DBT to DLI](#).

**Figure 7-1** profile file

```

dlitest:
  outputs:
    dev:
      type: dli
      region:
      project_id:
      access_id:
      secret_key:
      queue:
      database:
      schema:
      obs_endpoint:
  target: dev
  
```



Figure 7-2 profile configured in the dbt\_project.yml file

```

8
9 # This setting configures which "profile" dbt uses for this project.
10 profile: 'dlitest'
11
12 # These configurations specify where dbt should look for different types of files.
13 # The 'model-paths' config, for example, states that models in this project can be
14 # found in the "models/" directory. You probably won't need to change these!
15 model-paths: ["models"]
16 analysis-paths: ["analyses"]
17 test-paths: ["tests"]
18 seed-paths: ["seeds"]
19 macro-paths: ["macros"]
20 snapshot-paths: ["snapshots"]
21

```

3. **Verify the configuration.**

Run the following command to check whether the DBT configuration is correct:

```
dbt debug
```

4. **Run the job.**

Once the test is passed, run the following command to execute your data model:

```
dbt run
```

## 7.4 Connecting Yonghong BI to DLI for Data Query and Analysis

Yonghong BI is an enterprise-level data analysis tool that supports data visualization, report creation, data analysis, and decision-making. It helps businesses gain insights into their data and improve decision-making efficiency.

This section describes how to connect Yonghong BI to DLI.

### Preparations

- **Environment requirements:**
  - Yonghong BI has been installed.
- **DLI JDBC driver:**

Download the JDBC driver **huaweicloud-dli-jdbc-xxx-dependencies.jar** from the DLI management console.
- **Connection information:**

Table 7-7 Connection information

Item	Description	How to Obtain
DLI's AK/SK	AK/SK-based authentication refers to the use of an AK/SK pair to sign requests for identity authentication.	<a href="#">Obtaining an AK/SK</a>
DLI's endpoint address	Endpoint of a cloud service in a region.	<a href="#">Obtaining an Endpoint</a>

Item	Description	How to Obtain
DLI's project ID	Project ID, which is used for resource isolation.	<a href="#">Obtaining a Project ID</a>
DLI's region information	DLI's region information	<a href="#">Regions and Endpoints</a>

## Step 1: Creating a DLI Data Connection on Yonghong BI

1. Start Yonghong BI.
2. On the Yonghong BI page, click **Add Data Source**.
3. On the **New Connection Type** page, choose **GENERIC** as the data source type.
4. Configure the new connection.

**Driver:** Upload DLI's JDBC driver you have downloaded.

**URL:** Enter the URL of DLI JDBC. [Table 7-8](#) describes the URL format. [Table 7-9](#) describes attribute configuration items.

Specify a database.

### NOTE

- In **Schema**, you can optionally enter the name of the database to be accessed. If you enter the name, only tables in the database are displayed during data set creation. If you do not enter the name, tables in all databases are displayed during data set creation.
- Retain default values for other parameters. You do not need to select **Request Login**.

**Table 7-8** Database connection parameters

Parameter	Description
URL	<p>The URL format is as follows:</p> <pre><i>jdbc:dli://&lt;endPoint&gt;/&lt;projectId&gt;?&lt;key1&gt;=&lt;val1&gt;;&lt;key2&gt;=&lt;val2&gt;...</i></pre> <p><b>NOTE</b></p> <ul style="list-style-type: none"> <li>• <b>endpoint</b> indicates DLI's endpoint. Obtain it by referring to <a href="#">Regions and Endpoints</a>.</li> <li>• <b>projectId</b> indicates the project ID, which can be obtained from the <b>My Credentials</b> page of the public cloud platform.</li> <li>• ? is followed by other configuration items, with each item listed in the "key=value" format and multiple items are separated by semicolons (;). See <a href="#">Table 7-9</a>.</li> </ul>

**Table 7-9** Attribute configuration items

Attribute (key)	Mandatory	Default Value (value)	Description
enginetype	No	-	Type of the engine that executes jobs.
queuename	Yes	-	DLI queue name.
databasename	No	-	Default database to access. If not specified in the URL, you need to use <b>db.table</b> (for example, <b>select * from db.other.tabletest</b> ) to access tables in the database.
accesskey	Mandatory when authentication mode is ask.	-	Authentication mode that JDBC supports.
secretkey	Mandatory when authentication mode is ask.	-	Authentication mode that JDBC supports.
regionname	Mandatory when authentication mode is ask.	-	Obtain it by referring to <a href="#">Regions and Endpoints</a> .
servicename	Mandatory when authentication mode is ask.	-	Set it to <b>dli</b> as Yonghong BI connects to DLI.
catalog	No	dli	Type of metadata that the job reads.

5. On the tool bar of the displayed page, click **Test Connection**. Once the test is complete, click **Save**. Enter the data source name and save the data source.

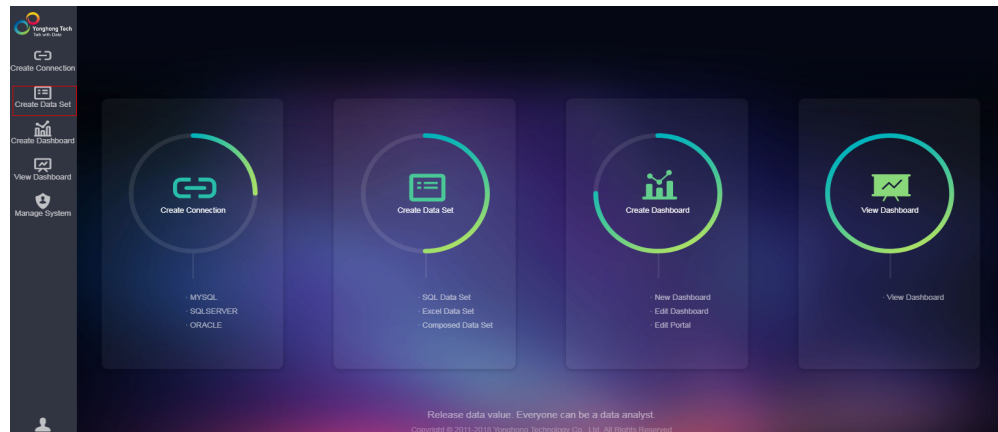
 **NOTE**

Currently, you can only save the data source to an existing folder as you are not allowed to save any data sources to the root directory.

## Step 2: Create a DLI Dataset on Yonghong BI

- Step 1** On the home page of the Yonghong SaaS production environment, click **Create Data Set** in the navigation pane on the left.

Figure 7-3 Creating a dataset



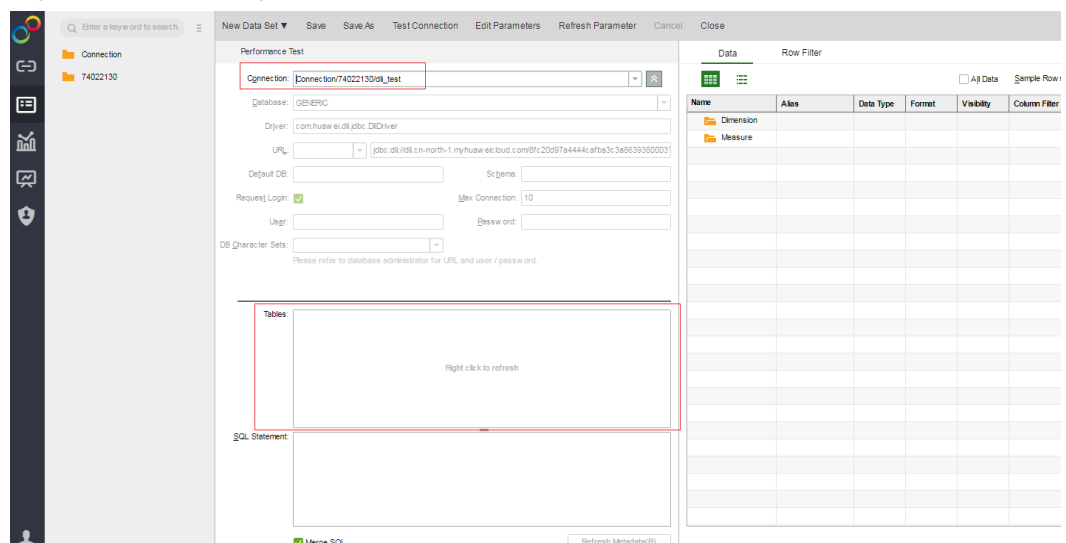
**Step 2** On the displayed page, click **SQL Data Set**.

Figure 7-4 Creating a SQL dataset



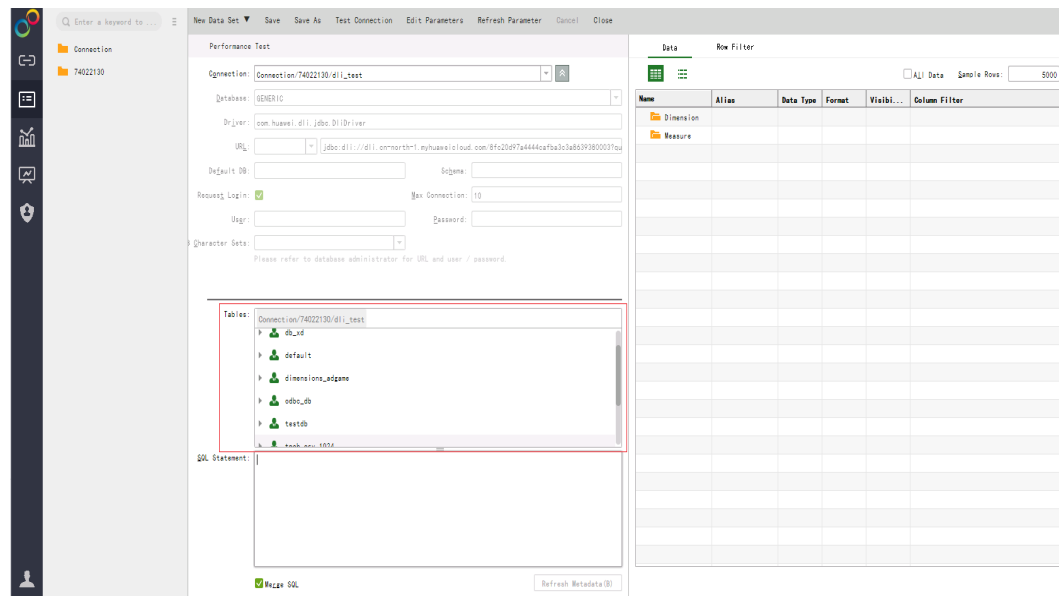
**Step 3** On the displayed page, select the added DLI data source from the **Connection** drop-down list box.


Figure 7-5 Selecting a data source



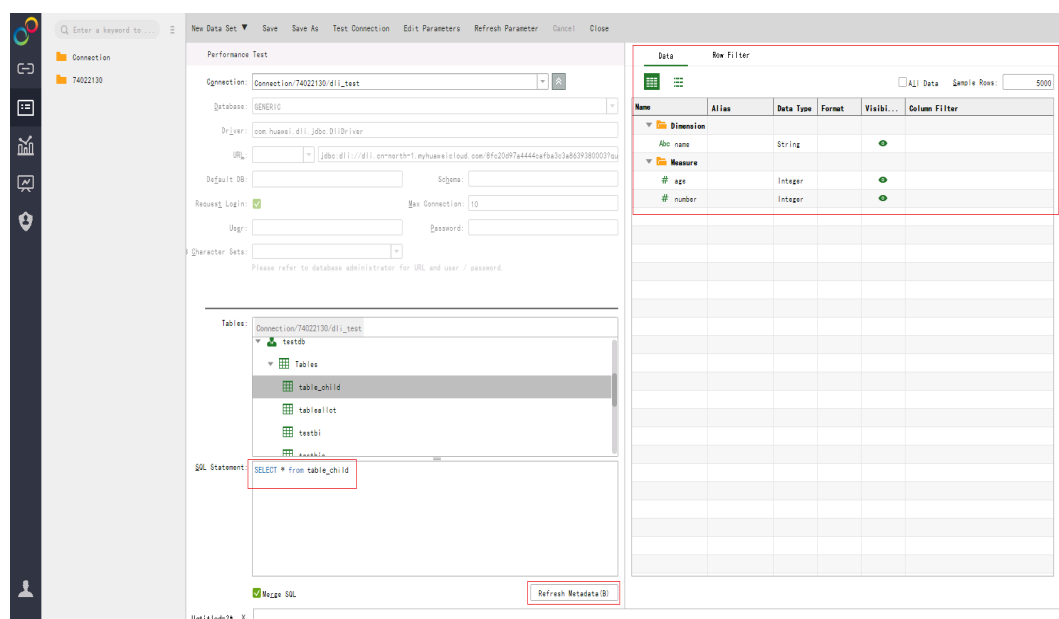
**Step 4** In the **Table** area on the left pane, right-click and choose **Update** to update tables. All databases and their tables are listed in the area. **Figure 7-6** shows the page displayed when **Table Structure** is not configured during connection creation.

**Figure 7-6** Updating tables



**Step 5** In the **SQL Statement** area on the left pane, enter the **select \* from table\_name** command to query tables. On the **Preview Data** page on the right pane, click . Metadata of the table, including fields and field types, is displayed.

**Figure 7-7** Querying the table




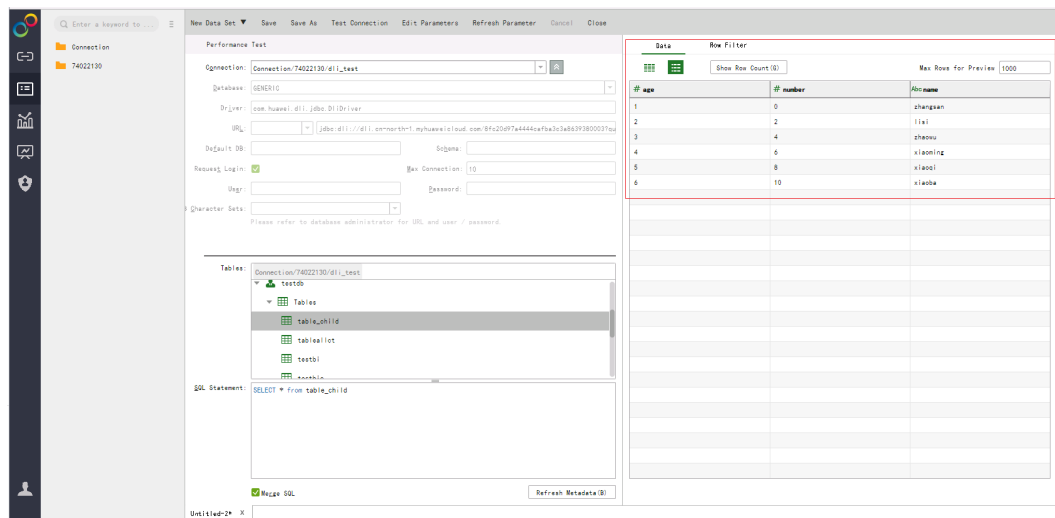
**Step 6** Click  in the right pane to query data details.

Figure 7-8 Querying data of the table



**Step 7** On the tool bar of the displayed page, click **Save**.

----End

Once Yonghong BI is connected to the DLI data source and a dataset is created, you can create BI charts in Yonghong BI as needed.

## 7.5 Connecting Power BI to DLI Using Kyuubi for Data Analysis and Query

Power BI offers features such as data integration, warehousing, reporting, and visualization. It can transform complex data into visual charts and dashboards that are easy to understand and interact with, helping businesses make data-driven decisions.

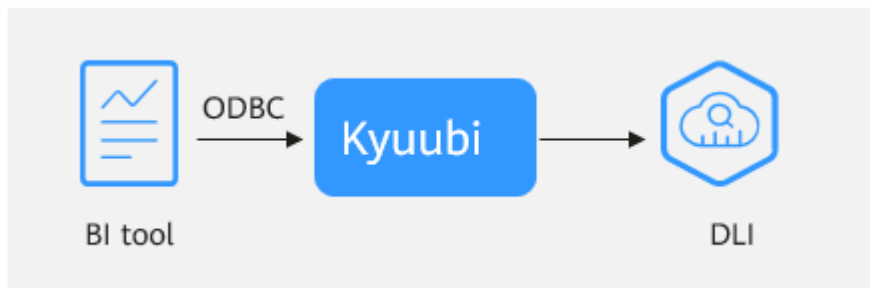
Kyuubi is a distributed SQL query engine that allows you to access and analyze data using standard SQL APIs.

Connect Power BI to Kyuubi and use Kyuubi to access DLI for data analysis and query. This simplifies the data access process, provides unified data management and analysis capabilities, and gains in-depth data insights.

This section describes how to connect Power BI to DLI using Kyuubi to access and analyze data in DLI.

## Procedure

Figure 7-9 Process



- **Step 1: Install and Connect Kyuubi to DLI:** Install and configure Kyuubi to ensure that Kyuubi can connect to DLI.
- **Step 2: Connect ODBC to Kyuubi:** Install the ODBC driver and connect the driver to the Kyuubi server.
- **Step 3: Connect Power BI to Kyuubi Using ODBC:** Create a data connection in the BI tool, use ODBC as the data source, and connect to Kyuubi through ODBC.

### Step 1: Install and Connect Kyuubi to DLI

To access Kyuubi from an external network, ensure that an EIP is bound to the ECS and ports **10009** and **3309** are enabled in the inbound rules of the security group.

#### Step 1 Install JDK.

Install JDK in your development environment before you can install and use Kyuubi.

Java SDKs require JDK 1.8 or later. To ensure compatibility with future versions, it is recommended to use version 1.8.

1. Download the JDK.  
Download the JDK 1.8 installation package from [Oracle official website](#) and install it.  
In this example, **jdk-8u261-linux-x64.tar.gz** is used.
2. Upload the JDK package to the corresponding directory of the Linux server and run the extraction command. In this example, the package is uploaded to **/usr/local**.  

```
sudo tar -xzf jdk-8u261-linux-x64.tar.gz -C /usr/local/
```
3. Configure environment variables.  
Add the following content to the **.bashrc** or **.profile** file:  

```
export JAVA_HOME=/usr/local/jdk-1.8.0_261
export PATH=$PATH:$JAVA_HOME/bin
```
4. Apply the environment variables.  

```
source ~/.bashrc
```
5. Run **java -version** to check whether the installation is successful. If the version number is displayed in the command output, the Java environment is successfully installed.

```
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
```

### Step 2 Install Kyuubi.

1. **Download the Kyuubi installation package.** Refer to **Deploying Kyuubi** to learn more how to install Kyuubi.
2. Extract the downloaded package.  

```
tar -xzf kyuubi-{version}-bin.tar.gz
```
3. (Optional) Configure environment variables.  
 Add the bin directory of Kyuubi to PATH environment variables to ensure that Kyuubi scripts can be called at any location.

### Step 3 Connect Kyuubi to DLI.

1. Add the DLI driver to the root directory of Kyuubi.  
 Place the driver in Kyuubi's root directory **/externals/engines/jdbc**.  
 Ensure that the user group and permissions of the plug-in are the same as those of other JAR files.
2. Modify the Kyuubi configuration file.  

```
cd $KYUUBI_HOME/confvi kyuubi-defaults.conf
```

**Table 7-10** describes the parameters.

**Table 7-10** Kyuubi parameters

Parameter	Description	Mandatory	Example Value
kyuubi.engine.type	JDBC service type. Set it to <b>dli</b> .	Yes	jdbc
kyuubi.engine.jdbc.type	Engine type. Set it to <b>dli</b> .	Yes	dli
kyuubi.engine.jdbc.driver.class	Name of the driver class used to connect JDBC. Set it to <b>com.huawei.dli.jdbc.DliDriver</b> .	Yes	com.huawei.dli.jdbc.DliDriver
kyuubi.engine.jdbc.connection.url	URL used to connect JDBC. Format: <b>jdbc:dli://{dliendpoint}/{projectId}</b> .	Yes	jdbc:dli://{dliendpoint} / {projectId}
kyuubi.engine.jdbc.session.initialize.sql	Initialization SQL statements executed when a JDBC session is established.	No	select 1  If <b>select 1</b> is displayed on the DLI management console, the initialization is successful.



Parameter	Description	Mandatory	Example Value
kyuubi.frontend.protocols	Frontend protocol supported by Kyuubi. Kyuubi supports various frontend protocols, allowing you to interact with Kyuubi through different APIs.	Yes	<ul style="list-style-type: none"> <li>- mysql</li> <li>- thrift_binary</li> </ul>
kyuubi.engine.dli.schema.show.name	<p>How the Kyuubi engine shows the schema name of data source APIs when you run <b>show schemas</b> or <b>show databases</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: The DLI name is included as the prefix when the schema name is shown.</li> <li>- <b>false</b>: The DLI name is not included when the schema name is shown.</li> </ul> <p>For example, if it is set to <b>true</b> and the DLI name is <b>hive</b>, the output is in the <b>hive.default</b> format when you run <b>show schemas</b>.</p> <p>If it is set to <b>false</b>, the output is in the <b>default</b> format.</p>	No	<ul style="list-style-type: none"> <li>- true</li> <li>- false</li> </ul>
kyuubi.engine.dli.jdbc.connection.region	Region name.	Yes	regionname=ap-southeast-2
kyuubi.engine.dli.jdbc.connection.queue	DLI queue name.	Yes	dli_test
kyuubi.engine.dli.jdbc.connection.database	Default database name used when the Kyuubi engine connects to the DLI data source through JDBC.	Yes	tpch
kyuubi.engine.dli.jdbc.connection.accesskey	AK that acts as the authentication key. Set it if <b>authenticationmode</b> is <b>ask</b> .	Yes	accesskey=your-access-key

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.jdbc.connection.secretkey	Region name. Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	secretkey=your-secret-key
kyuubi.engine.dli.jdbc.connection.project	ID of the project where DLI resources are.	Yes	0b33ea2a7e0010802fe4c009bb05076d
kyuubi.engine.dli.sql.limit.time.sec	Execution duration limit of a SQL query. The default value is <b>600</b> seconds.	No	300
kyuubi.engine.dli.result.line.number.limit	Maximum number of data records returned for a SQL query. By default, 100,000 data records are returned. Setting it to <b>-1</b> means that the number of returned data records is not limited.	Yes	50000
kyuubi.engine.dli.small.file.merge	Whether to enable automatic small file merging. The default value is <b>false</b> , indicating that the function is disabled. - <b>true</b> : Enable - <b>false</b> : Disable	Yes	true
kyuubi.engine.dli.bi.type	BI tool type. The options are <b>fine</b> , <b>grafana</b> , <b>superset</b> , <b>tableau</b> , <b>power</b> , <b>dbt</b> , and <b>yongHong</b> .	Yes	fine

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.boolean.type.to.int	<p>Whether DLI's Boolean data is returned as <b>1/0</b> or <b>true/false</b>.</p> <p>If the BI tool type is <b>grafana</b>, set this parameter to <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: returned as <b>1/0</b> (1: true; 0: false)</li> <li>- <b>false</b>: returned as <b>true/false</b>.</li> </ul> <p>The default value is <b>false</b>.</p>	No	false
kyuubi.engine.dli.set.conf.transform.to.annotation	<p>Allows you to set <b>set spark</b> in SQL statements.</p> <p>Set it to <b>true</b> for Power BI, FineBI, Superset, and DBT.</p>	No	true
kyuubi.engine.dli.set.conf.sql.suffix	<p>Allows you to set <b>set spark</b> at the end of SQL statements.</p> <p>Set it to <b>true</b> for Power BI and DBT.</p>	No	true
kyuubi.engine.dli.result.cache.enable	<p>Whether to enable database table data caching. Once enabled, database table data is automatically cached. The default value is <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: Enable</li> <li>- <b>false</b>: Disable</li> </ul>	No	true
kyuubi.engine.dli.cache.limit.line.number	<p>Maximum number of data records that can be cached.</p> <p>By default, 100,000 data records are cached.</p> <p>Setting it to <b>-1</b> means that the number of cached data records is not limited.</p>	No	1000
kyuubi.engine.dli.cache.time.sec	<p>Cache time.</p> <p>The default value is <b>1800</b> seconds.</p>	No	1800

Parameter	Description	Mandatory	Example Value
kyuubi.operation.incremental.collect	Kyuubi pre-loads select result data into the cache to accelerate data reading. You are advised to disable this function to avoid memory OOM issues when dealing with large volumes of data.	No	false Setting it to <b>false</b> indicates disabling preloading.
kyuubi.engine.jdbc.memory	Memory of the JDBC engine process The default value is <b>1g</b> . You are advised to set it to at least <b>5g</b> to increase memory for the JDBC engine process.	No	5g

3. Quickly start Kyuubi.

Go to the root directory **/bin** of the ECS and run the following command to start Kyuubi:

```
cd /bin
./kyuubi start restart
```

Once the start is successful, run SQL statements to check whether the connection between Kyuubi and DLI is normal.

**Step 4 (Optional) Configure the hosts file to speed up access to Kyuubi.**

To make access to Kyuubi faster, you are advised to add the Kyuubi host IP address to the **/etc/hosts** file on the host.

1. Run **ifconfig** to check the IP address of the host.

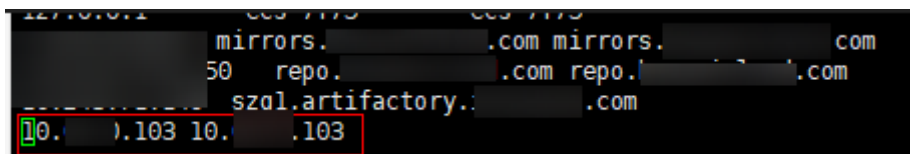
**Figure 7-10** Checking the host IP address

```
[root@ecs-7f75 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.63.0.103 netmask 255.255.255.0 broadcast 10.63.0.255
    inet6 fe80::f816:3eff:febd:3a50 prefixlen 64 scopeid 0x20<link>
    ether fa:16:3e:fd:3a:50 txqueuelen 1000 (Ethernet)
    RX packets 6654437 bytes 2845894229 (2.6 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4585886 bytes 1125818425 (1.0 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1502680 bytes 307935807 (293.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1502680 bytes 307935807 (293.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2. Add the IP address to the **/etc/hosts** file.

**Figure 7-11** Adding the host IP address to the /etc/hosts file



----End

## Step 2: Connect ODBC to Kyuubi

### Step 1 Install the ODBC driver.

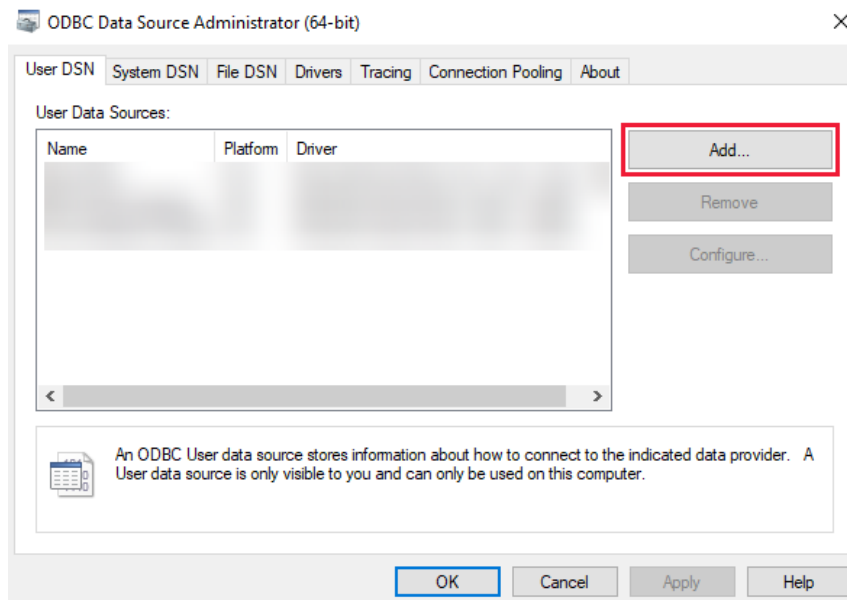
Install the ODBC driver on the local host based on the database type. In this example, the Hive database is used.

- [Cloudera Hive ODBC](#). Version 2.5.12 is recommended.
- [Microsoft Hive ODBC](#). Version 2.6.12.1012 is recommended.

### Step 2 Connect ODBC to Kyuubi.

1. In Windows, click **Control Panel**, double-click **Administrative Tools**, and double-click **Data Sources (ODBC)**.
2. Configure an ODBC data source.
  - a. Click **User DSN**.
  - b. Click **Add**.
  - c. Select **Hive ODBC Driver** and click **OK**.

**Figure 7-12** Creating an ODBC data source connection



3. In the dialog box that appears, enter information about the Kyuubi server.
  - **Database:** Enter the DLI database name.
  - **Host(s):** Enter the EIP of the Kyuubi server.
  - **Port:** listening port of Kyuubi. The Hive Thrift protocol is used. The default port is **10009**.

- **User Name** and **Password**: Configure the username and password of the Kyuubi server as required.

Set other advanced options and save the settings.

**Figure 7-13** ODBC data source connection parameters

The screenshot shows the ODBC data source configuration dialog. The 'Description' field contains 'Sample Microsoft Hive DSN'. The 'Host(s)' field is empty and highlighted with a red box. The 'Port' field contains '10009' and is also highlighted with a red box. The 'Database' field contains 'nbbtemp'. Under the 'Authentication' section, the 'Mechanism' dropdown is set to 'User Name' and is highlighted with a red box. The 'User Name' field is empty and highlighted with a red box. Other fields include 'Realm', 'Host FQDN' (containing '\_HOST'), 'Service Name' (containing 'hive'), and checkboxes for 'Canonicalize Principal FQDN' (checked) and 'Delegate Kerberos Credentials' (unchecked). There are buttons for 'Password Options...', 'Delegation UID', 'Thrift Transport' (set to 'SASL'), 'Proxy Options...', 'HTTP Options...', 'SSL Options...', 'Advanced Options...', and 'Logging Options...'. At the bottom, there is a version string 'v2.6.12.1012 (64 bit)' and 'Test', 'OK', and 'Cancel' buttons.

4. Click **Test** to check whether the data source connection is successful. If successful, click **OK** to save the connection.

----End

### Step 3: Connect Power BI to Kyuubi Using ODBC

1. Click and install Power BI. [Download the Power BI installation package.](#)
2. Start Power BI Desktop.
3. On the **Home** tab, click **Get data**.
4. Click **More...** to see other types of data sources available.
5. Select **ODBC** from the list as the data source type and click **Connect**.
6. In the **ODBC Driver Manager** window that appears, select the ODBC data source configured in [Step 2: Connect ODBC to Kyuubi](#) and click **OK**.

Power BI connects to Kyuubi using ODBC and allows you to preview and select tables and views in the database.

#### NOTE

When previewing database tables, select **limit**. Otherwise, all partitioned tables are scanned.

### Common Operations: Setting SQL Job Parameters

Select a configuration method based on the type of the installed ODBC driver.

- **Use the Cloudera Hive ODBC (v2.5.12) driver.**

You only need to add annotation parameters to the end of SQL statements.

```
-- @set Parameter
```

Example:

```
-- @set dli.sql.current.database=tpch
```

```
-- @set dli.sql.shuffle.partitions=100
```

- **Use the Microsoft Hive ODBC (v2.6.12.1012) driver.**

- a. Make sure that the following parameters are enabled in the **/conf/kyuubi-defaults.conf** file of Kyuubi:

```
kyuubi.engine.dli.set.conf.transform.to.annotation=true
```

```
kyuubi.engine.dli.set.conf.sql.suffix=true
```

- b. Add annotation parameters to the end of SQL statements.

```
set Parameter
```

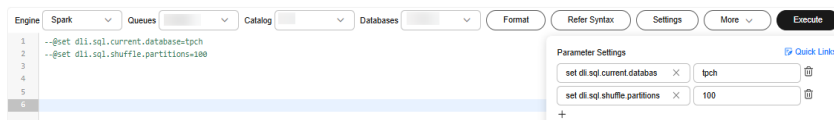
Example:

```
set dli.sql.current.database=tpch
```

```
set dli.sql.shuffle.partitions=100
```

The execution result in the SQL Editor of DLI is as follows:

**Figure 7-14** Viewing the configured parameters in the SQL Editor of DLI



## 7.6 Connecting FineBI to DLI Using Kyuubi for Data Analysis and Query

FineBI is an intelligent, visual tool that focuses on data analysis and visualization. It can connect to various data sources and transform intricate data into user-friendly charts and dashboards, allowing for swift and efficient data analysis.

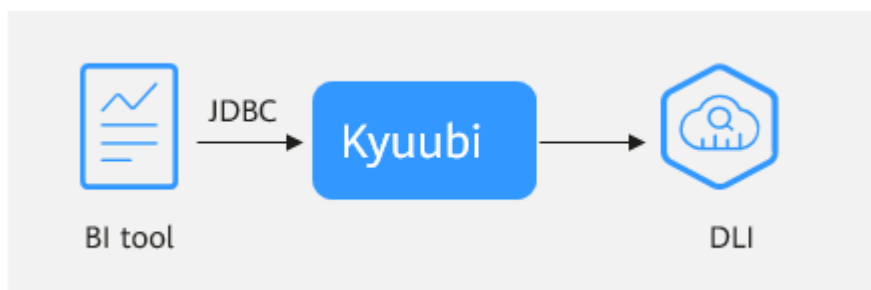
Kyuubi is a distributed SQL query engine that provides standard SQL APIs. This allows you to easily access and analyze data stored on big data platforms.

By connecting FineBI to Kyuubi, you can use Kyuubi to access DLI for data analysis and query. This integration simplifies data access and provides unified data management and analysis, enabling you to gain deeper insights into data.

This section describes how to connect FineBI to DLI using Kyuubi to access and analyze data in DLI.

## Procedure

Figure 7-15 Process



- **Step 1: Install and Connect Kyuubi to DLI:** Install and configure Kyuubi to ensure that Kyuubi can connect to DLI.
- **Step 2: Install FineBI and the Data Connection Driver:** Install FineBI and the data connection driver.
- **Step 3: Connect FineBI to Kyuubi:** Create a data connection in the BI tool and connect to Kyuubi using JDBC.

### Step 1: Install and Connect Kyuubi to DLI

To access Kyuubi from an external network, ensure that an EIP is bound to the ECS and ports **10009** and **3309** are enabled in the inbound rules of the security group.

#### Step 1 Install JDK.

Install JDK in your development environment before you can install and use Kyuubi.

Java SDKs require JDK 1.8 or later. To ensure compatibility with future versions, it is recommended to use version 1.8.

1. Download the JDK.  
Download the JDK 1.8 installation package from [Oracle official website](#) and install it.  
In this example, **jdk-8u261-linux-x64.tar.gz** is used.
2. Upload the JDK package to the corresponding directory of the Linux server and run the extraction command. In this example, the package is uploaded to **/usr/local**.  

```
sudo tar -xzf jdk-8u261-linux-x64.tar.gz -C /usr/local/
```
3. Configure environment variables.



Add the following content to the **.bashrc** or **.profile** file:

```
export JAVA_HOME=/usr/local/jdk-1.8.0_261
export PATH=$PATH:$JAVA_HOME/bin
```

4. Apply the environment variables.  
source ~/.bashrc
5. Run **java -version** to check whether the installation is successful. If the version number is displayed in the command output, the Java environment is successfully installed.  
java version "1.8.0\_261"  
Java(TM) SE Runtime Environment (build 1.8.0\_261-b12)  
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)

## Step 2 Install Kyuubi.

1. **Download the Kyuubi installation package.** Refer to [Deploying Kyuubi](#) to learn more how to install Kyuubi.
2. Extract the downloaded package.  
tar -xzf kyuubi-{version}-bin.tar.gz
3. (Optional) Configure environment variables.  
Add the bin directory of Kyuubi to PATH environment variables to ensure that Kyuubi scripts can be called at any location.

## Step 3 Connect Kyuubi to DLI.

1. Add the DLI driver to the root directory of Kyuubi.  
Place the driver in Kyuubi's root directory **/externals/engines/jdbc**.  
Ensure that the user group and permissions of the plug-in are the same as those of other JAR files.
2. Modify the Kyuubi configuration file.  
**cd \$KYUUBI\_HOME/confvi kyuubi-defaults.conf**  
**Table 7-11** describes the parameters.

**Table 7-11** Kyuubi parameters

Parameter	Description	Mandatory	Example Value
kyuubi.engine.type	JDBC service type. Set it to <b>dli</b> .	Yes	jdbc
kyuubi.engine.jdbc.type	Engine type. Set it to <b>dli</b> .	Yes	dli
kyuubi.engine.jdbc.driver.class	Name of the driver class used to connect JDBC. Set it to <b>com.huawei.dli.jdbc.DliDriver</b> .	Yes	com.huawei.dli.jdbc.DliDriver
kyuubi.engine.jdbc.connection.url	URL used to connect JDBC. Format: <b>jdbc:dli://{dliendpoint}/{projectId}</b> .	Yes	jdbc:dli://{dliendpoint} / {projectId}

Parameter	Description	Mandatory	Example Value
kyuubi.engine.jdbc.session.initialize.sql	Initialization SQL statements executed when a JDBC session is established.	No	select 1 If <b>select 1</b> is displayed on the DLI management console, the initialization is successful.
kyuubi.frontend.protocols	Frontend protocol supported by Kyuubi. Kyuubi supports various frontend protocols, allowing you to interact with Kyuubi through different APIs.	Yes	<ul style="list-style-type: none"> <li>- mysql</li> <li>- thrift_binary</li> </ul>
kyuubi.engine.dli.schema.show.name	<p>How the Kyuubi engine shows the schema name of data source APIs when you run <b>show schemas</b> or <b>show databases</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: The DLI name is included as the prefix when the schema name is shown.</li> <li>- <b>false</b>: The DLI name is not included when the schema name is shown.</li> </ul> <p>For example, if it is set to <b>true</b> and the DLI name is <b>hive</b>, the output is in the <b>hive.default</b> format when you run <b>show schemas</b>. If it is set to <b>false</b>, the output is in the <b>default</b> format.</p>	No	<ul style="list-style-type: none"> <li>- true</li> <li>- false</li> </ul>
kyuubi.engine.dli.jdbc.connection.region	Region name.	Yes	regionname=ap-southeast-2
kyuubi.engine.dli.jdbc.connection.queue	DLI queue name.	Yes	dli_test
kyuubi.engine.dli.jdbc.connection.database	Default database name used when the Kyuubi engine connects to the DLI data source through JDBC.	Yes	tpch

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.jdbc.connection.accesskey	AK that acts as the authentication key. Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	accesskey=your-access-key
kyuubi.engine.dli.jdbc.connection.secretkey	Region name. Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	secretkey=your-secret-key
kyuubi.engine.dli.jdbc.connection.project	ID of the project where DLI resources are.	Yes	0b33ea2a7e0010802fe4c009bb05076d
kyuubi.engine.dli.sql.limit.time.sec	Execution duration limit of a SQL query. The default value is <b>600</b> seconds.	No	300
kyuubi.engine.dli.result.line.number.limit	Maximum number of data records returned for a SQL query. By default, 100,000 data records are returned. Setting it to <b>-1</b> means that the number of returned data records is not limited.	Yes	50000
kyuubi.engine.dli.small.file.merge	Whether to enable automatic small file merging. The default value is <b>false</b> , indicating that the function is disabled. - <b>true</b> : Enable - <b>false</b> : Disable	Yes	true
kyuubi.engine.dli.bi.type	BI tool type. The options are <b>fine</b> , <b>grafana</b> , <b>superset</b> , <b>tableau</b> , <b>power</b> , <b>dbt</b> , and <b>yongHong</b> .	Yes	fine

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.boolean.type.to.int	<p>Whether DLI's Boolean data is returned as <b>1/0</b> or <b>true/false</b>.</p> <p>If the BI tool type is <b>grafana</b>, set this parameter to <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: returned as <b>1/0</b> (1: true; 0: false)</li> <li>- <b>false</b>: returned as <b>true/false</b>.</li> </ul> <p>The default value is <b>false</b>.</p>	No	false
kyuubi.engine.dli.set.conf.transform.to.annotation	<p>Allows you to set <b>set spark</b> in SQL statements.</p> <p>Set it to <b>true</b> for Power BI, FineBI, Superset, and DBT.</p>	No	true
kyuubi.engine.dli.set.conf.sql.suffix	<p>Allows you to set <b>set spark</b> at the end of SQL statements.</p> <p>Set it to <b>true</b> for Power BI and DBT.</p>	No	true
kyuubi.engine.dli.result.cache.enable	<p>Whether to enable database table data caching. Once enabled, database table data is automatically cached. The default value is <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: Enable</li> <li>- <b>false</b>: Disable</li> </ul>	No	true
kyuubi.engine.dli.cache.limit.line.number	<p>Maximum number of data records that can be cached.</p> <p>By default, 100,000 data records are cached.</p> <p>Setting it to <b>-1</b> means that the number of cached data records is not limited.</p>	No	1000
kyuubi.engine.dli.cache.time.sec	<p>Cache time.</p> <p>The default value is <b>1800</b> seconds.</p>	No	1800

Parameter	Description	Mandatory	Example Value
kyuubi.operation.incremental.collect	Kyuubi pre-loads select result data into the cache to accelerate data reading. You are advised to disable this function to avoid memory OOM issues when dealing with large volumes of data.	No	false Setting it to <b>false</b> indicates disabling preloading.
kyuubi.engine.jdbc.memory	Memory of the JDBC engine process  The default value is <b>1g</b> . You are advised to set it to at least <b>5g</b> to increase memory for the JDBC engine process.	No	5g

3. Quickly start Kyuubi.

Go to the root directory **/bin** of the ECS and run the following command to start Kyuubi:

```
cd /bin
./kyuubi start restart
```

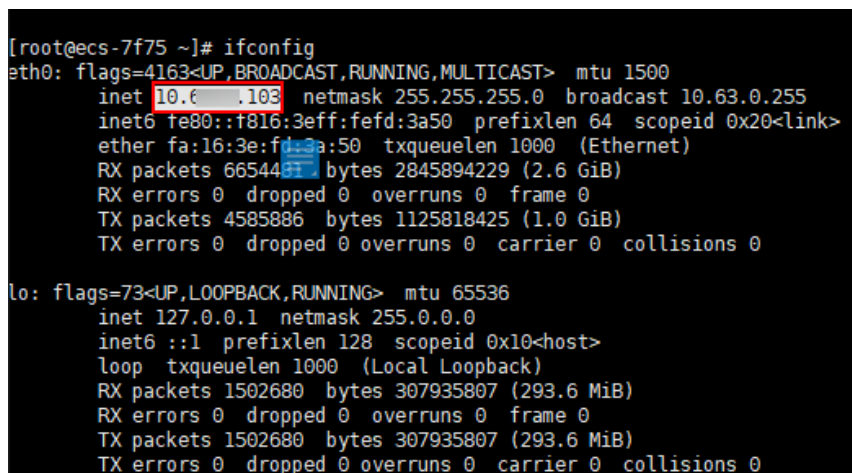
Once the start is successful, run SQL statements to check whether the connection between Kyuubi and DLI is normal.

**Step 4 (Optional) Configure the hosts file to speed up access to Kyuubi.**

To make access to Kyuubi faster, you are advised to add the Kyuubi host IP address to the **/etc/hosts** file on the host.

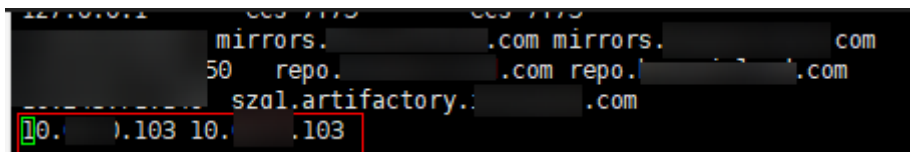
1. Run **ifconfig** to check the IP address of the host.

**Figure 7-16** Checking the host IP address



2. Add the IP address to the **/etc/hosts** file.

Figure 7-17 Adding the host IP address to the /etc/hosts file



----End

## Step 2: Install FineBI and the Data Connection Driver

### Step 1 Download and install FineBI.

1. Download the [FineBI installation package](#).
2. Find the downloaded FineBI installation program file.
3. Double-click the installation program.
4. Follow the installation wizard to perform installation operations, including accepting the license agreement, selecting the installation type (typical or custom), and setting the installation directory.

### Step 2 Integrate the JDBC driver with FineBI.

1. Download the data driver. Download the [Hive JDBC driver package](#). Version 2.6.23 is recommended.
2. Integrate the driver plug-in with FineBI.
  - a. Open FineBI.
  - b. Choose **Data Connection > Data Connection Management**.
  - c. Click **New Driver**. In the driver list, select the driver in [Step 2.1](#).

----End

## Step 3: Connect FineBI to Kyuubi

1. Open FineBI.
2. Choose **Data Connection > Data Connection Management**.
3. Click **Add Data Source**.
4. Select a database type in the data source wizard. In this example, **Hadoop Hive** is selected.
5. Configure database connection information.
  - **Data Connection Name**: Enter a data connection name.
  - **Driver**: Select the driver in [Step 2.1](#).
  - **Database Name**: Enter the DLI database name.
  - **Host**: Enter the IP address of the host where Kyuubi is installed.
  - **Port**: Enter the port used to access the Kyuubi host. The default port is **10009**.
  - **Authentication Method**: Select **Username and Password** for this example.
  - **Username**: username used to access the Kyuubi database.

- **Password:** password used to access the Kyuubi database.
- **Data Connection URL:**

Hive 0.11.0 or later:

```
jdbc:hive2://localhost:10009/databasename  
Example: jdbc:hive2://100.xx.xxx.243:10009/tpch
```

## Common Operations: Setting SQL Job Parameters

1. Make sure that the following parameters are enabled in the `/conf/kyuubi-defaults.conf` file of Kyuubi:

```
kyuubi.engine.dli.set.conf.transform.to.annotation=true
```

2. Add annotation parameters to the end of SQL statements.

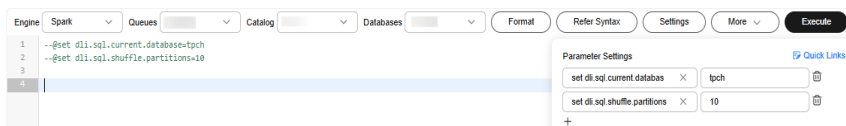
```
set Parameter
```

Example:

```
set dli.sql.current.database=tpch  
set dli.sql.shuffle.partitions=10
```

The execution result in the SQL Editor of DLI is as follows: The set parameters are changed to annotations and submitted to DLI for execution.

**Figure 7-18** Viewing the configured parameters in the SQL Editor of DLI



## 7.7 Connecting Superset to DLI Using Kyuubi for Data Analysis and Query

Superset is an open source platform for data exploration and visualization. It allows for fast and intuitive exploration of data, as well as the creation of rich data visualizations and interactive dashboards.

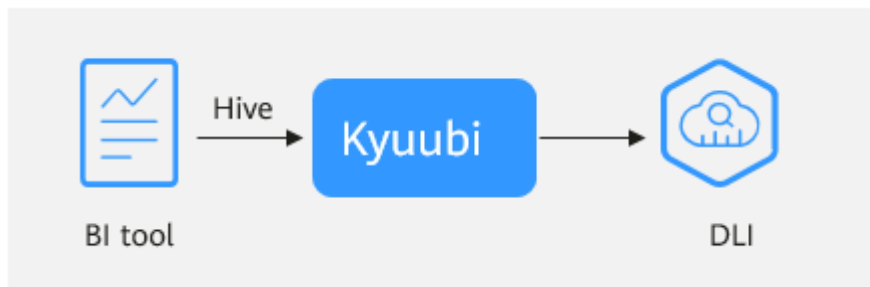
Kyuubi is a distributed SQL query engine that provides standard SQL APIs. This allows you to easily access and analyze data stored on big data platforms.

By connecting Superset to Kyuubi, you can use Kyuubi to access DLI for data analysis and query. This integration simplifies data access and provides unified data management and analysis, enabling you to gain deeper insights into data.

This section describes how to connect Superset to DLI using Kyuubi to access and analyze data in DLI.

## Procedure

Figure 7-19 Process



- **Step 1: Install and Connect Kyuubi to DLI:** Install and configure Kyuubi to ensure that Kyuubi can connect to DLI.
- **Step 2: Install Superset and the Data Connection Driver:** Install Superset and the data connection driver.
- **Step 3: Connect Superset to Kyuubi:** Create a data connection in the BI tool and connect to Kyuubi using JDBC.

### Step 1: Install and Connect Kyuubi to DLI

To access Kyuubi from an external network, ensure that an EIP is bound to the ECS and ports **10009** and **3309** are enabled in the inbound rules of the security group.

#### Step 1 Install JDK.

Install JDK in your development environment before you can install and use Kyuubi.

Java SDKs require JDK 1.8 or later. To ensure compatibility with future versions, it is recommended to use version 1.8.

1. Download the JDK.

Download the JDK 1.8 installation package from [Oracle official website](#) and install it.

In this example, **jdk-8u261-linux-x64.tar.gz** is used.

2. Upload the JDK package to the corresponding directory of the Linux server and run the extraction command. In this example, the package is uploaded to **/usr/local**.

```
sudo tar -xzf jdk-8u261-linux-x64.tar.gz -C /usr/local/
```

3. Configure environment variables.

Add the following content to the **.bashrc** or **.profile** file:

```
export JAVA_HOME=/usr/local/jdk-1.8.0_261
export PATH=$PATH:$JAVA_HOME/bin
```

4. Apply the environment variables.

```
source ~/.bashrc
```

5. Run **java -version** to check whether the installation is successful. If the version number is displayed in the command output, the Java environment is successfully installed.



```
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
```

### Step 2 Install Kyuubi.

1. **Download the Kyuubi installation package.** Refer to [Deploying Kyuubi](#) to learn more how to install Kyuubi.
2. Extract the downloaded package.  

```
tar -xzf kyuubi-{version}-bin.tar.gz
```
3. (Optional) Configure environment variables.  
 Add the bin directory of Kyuubi to PATH environment variables to ensure that Kyuubi scripts can be called at any location.

### Step 3 Connect Kyuubi to DLI.

1. Add the DLI driver to the root directory of Kyuubi.  
 Place the driver in Kyuubi's root directory **/externals/engines/jdbc**.  
 Ensure that the user group and permissions of the plug-in are the same as those of other JAR files.
2. Modify the Kyuubi configuration file.

```
cd $KYUUBI_HOME/confvi kyuubi-defaults.conf
```

[Table 7-12](#) describes the parameters.

**Table 7-12** Kyuubi parameters

Parameter	Description	Mandatory	Example Value
kyuubi.engine.type	JDBC service type. Set it to <b>dli</b> .	Yes	jdbc
kyuubi.engine.jdbc.type	Engine type. Set it to <b>dli</b> .	Yes	dli
kyuubi.engine.jdbc.driver.class	Name of the driver class used to connect JDBC. Set it to <b>com.huawei.dli.jdbc.DliDriver</b> .	Yes	com.huawei.dli.jdbc.DliDriver
kyuubi.engine.jdbc.connection.url	URL used to connect JDBC. Format: <b>jdbc:dli://{dliendpoint}/{projectId}</b> .	Yes	jdbc:dli://{dliendpoint} / {projectId}
kyuubi.engine.jdbc.session.initialize.sql	Initialization SQL statements executed when a JDBC session is established.	No	select 1 If <b>select 1</b> is displayed on the DLI management console, the initialization is successful.

Parameter	Description	Mandatory	Example Value
kyuubi.frontend.protocols	Frontend protocol supported by Kyuubi. Kyuubi supports various frontend protocols, allowing you to interact with Kyuubi through different APIs.	Yes	<ul style="list-style-type: none"> <li>- mysql</li> <li>- thrift_binary</li> </ul>
kyuubi.engine.dli.schema.show.name	<p>How the Kyuubi engine shows the schema name of data source APIs when you run <b>show schemas</b> or <b>show databases</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: The DLI name is included as the prefix when the schema name is shown.</li> <li>- <b>false</b>: The DLI name is not included when the schema name is shown.</li> </ul> <p>For example, if it is set to <b>true</b> and the DLI name is <b>hive</b>, the output is in the <b>hive.default</b> format when you run <b>show schemas</b>.</p> <p>If it is set to <b>false</b>, the output is in the <b>default</b> format.</p>	No	<ul style="list-style-type: none"> <li>- true</li> <li>- false</li> </ul>
kyuubi.engine.dli.jdbc.connection.region	Region name.	Yes	regionname=ap-southeast-2
kyuubi.engine.dli.jdbc.connection.queue	DLI queue name.	Yes	dli_test
kyuubi.engine.dli.jdbc.connection.database	Default database name used when the Kyuubi engine connects to the DLI data source through JDBC.	Yes	tpch
kyuubi.engine.dli.jdbc.connection.accesskey	<p>AK that acts as the authentication key.</p> <p>Set it if <b>authenticationmode</b> is <b>ask</b>.</p>	Yes	accesskey=your-access-key

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.jdbc.connection.secretkey	Region name. Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	secretkey=your-secret-key
kyuubi.engine.dli.jdbc.connection.project	ID of the project where DLI resources are.	Yes	0b33ea2a7e0010802fe4c009bb05076d
kyuubi.engine.dli.sql.limit.time.sec	Execution duration limit of a SQL query. The default value is <b>600</b> seconds.	No	300
kyuubi.engine.dli.result.line.number.limit	Maximum number of data records returned for a SQL query. By default, 100,000 data records are returned. Setting it to <b>-1</b> means that the number of returned data records is not limited.	Yes	50000
kyuubi.engine.dli.small.file.merge	Whether to enable automatic small file merging. The default value is <b>false</b> , indicating that the function is disabled. - <b>true</b> : Enable - <b>false</b> : Disable	Yes	true
kyuubi.engine.dli.bi.type	BI tool type. The options are <b>fine</b> , <b>grafana</b> , <b>superset</b> , <b>tableau</b> , <b>power</b> , <b>dbt</b> , and <b>yongHong</b> .	Yes	fine

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.boolean.type.to.int	<p>Whether DLI's Boolean data is returned as <b>1/0</b> or <b>true/false</b>.</p> <p>If the BI tool type is <b>grafana</b>, set this parameter to <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: returned as <b>1/0</b> (1: true; 0: false)</li> <li>- <b>false</b>: returned as <b>true/false</b>.</li> </ul> <p>The default value is <b>false</b>.</p>	No	false
kyuubi.engine.dli.set.conf.transform.to.annotation	<p>Allows you to set <b>set spark</b> in SQL statements.</p> <p>Set it to <b>true</b> for Power BI, FineBI, Superset, and DBT.</p>	No	true
kyuubi.engine.dli.set.conf.sql.suffix	<p>Allows you to set <b>set spark</b> at the end of SQL statements.</p> <p>Set it to <b>true</b> for Power BI and DBT.</p>	No	true
kyuubi.engine.dli.result.cache.enable	<p>Whether to enable database table data caching. Once enabled, database table data is automatically cached. The default value is <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: Enable</li> <li>- <b>false</b>: Disable</li> </ul>	No	true
kyuubi.engine.dli.cache.limit.line.number	<p>Maximum number of data records that can be cached.</p> <p>By default, 100,000 data records are cached.</p> <p>Setting it to <b>-1</b> means that the number of cached data records is not limited.</p>	No	1000
kyuubi.engine.dli.cache.time.sec	<p>Cache time.</p> <p>The default value is <b>1800</b> seconds.</p>	No	1800

Parameter	Description	Mandatory	Example Value
kyuubi.operation.incremental.collect	Kyuubi pre-loads select result data into the cache to accelerate data reading. You are advised to disable this function to avoid memory OOM issues when dealing with large volumes of data.	No	false Setting it to <b>false</b> indicates disabling preloading.
kyuubi.engine.jdbc.memory	Memory of the JDBC engine process  The default value is <b>1g</b> . You are advised to set it to at least <b>5g</b> to increase memory for the JDBC engine process.	No	5g

3. Quickly start Kyuubi.

Go to the root directory **/bin** of the ECS and run the following command to start Kyuubi:

```
cd /bin
./kyuubi start restart
```

Once the start is successful, run SQL statements to check whether the connection between Kyuubi and DLI is normal.

**Step 4 (Optional) Configure the hosts file to speed up access to Kyuubi.**

To make access to Kyuubi faster, you are advised to add the Kyuubi host IP address to the **/etc/hosts** file on the host.

1. Run **ifconfig** to check the IP address of the host.

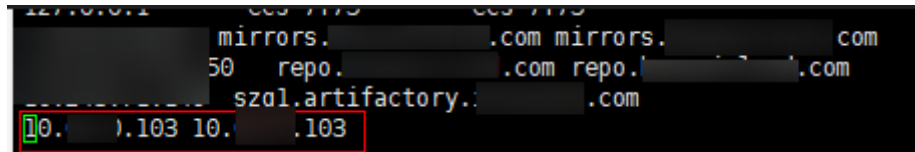
**Figure 7-20** Checking the host IP address

```
[root@ecs-7f75 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.63.0.103 netmask 255.255.255.0 broadcast 10.63.0.255
    inet6 fe80::f816:3eff:febd:3a50 prefixlen 64 scopeid 0x20<link>
    ether fa:16:3e:fd:3a:50 txqueuelen 1000 (Ethernet)
    RX packets 6654437 bytes 2845894229 (2.6 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4585886 bytes 1125818425 (1.0 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1502680 bytes 307935807 (293.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1502680 bytes 307935807 (293.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2. Add the IP address to the **/etc/hosts** file.

**Figure 7-21** Adding the host IP address to the /etc/hosts file



----End

## Step 2: Install Superset and the Data Connection Driver

**Step 1** Download and install Superset.

For details, see [Installing Superset](#).

The following uses how to **install Superset in Docker** as an example:

1. Install Docker.

Ensure that Docker is installed on the current host.

2. Pull the Superset image for Docker.

```
docker pull apache/superset
```

3. Start the Superset container.

```
docker run -p 8088:8088 apache/superset
```

Start the Superset container and map port 8088 of the container to port 8088 of the host machine.

4. Access Superset.

Access **http://localhost:8088** in a browser and log in using the default username and password (typically **admin/admin**).

**Step 2** [Download the Apache Hive driver package](#). PyHive 0.7.0 is recommended.

For details, see [Installing the Hive Driver](#).

----End

## Step 3: Connect Superset to Kyuubi

1. Open and log in to Superset.

2. Choose **Data > Databases**.

3. Click **Add Database**.

In the **Database** window that appears, select the driver installed in [Step 2: Install Superset and the Data Connection Driver](#).

4. Configure data connection information.

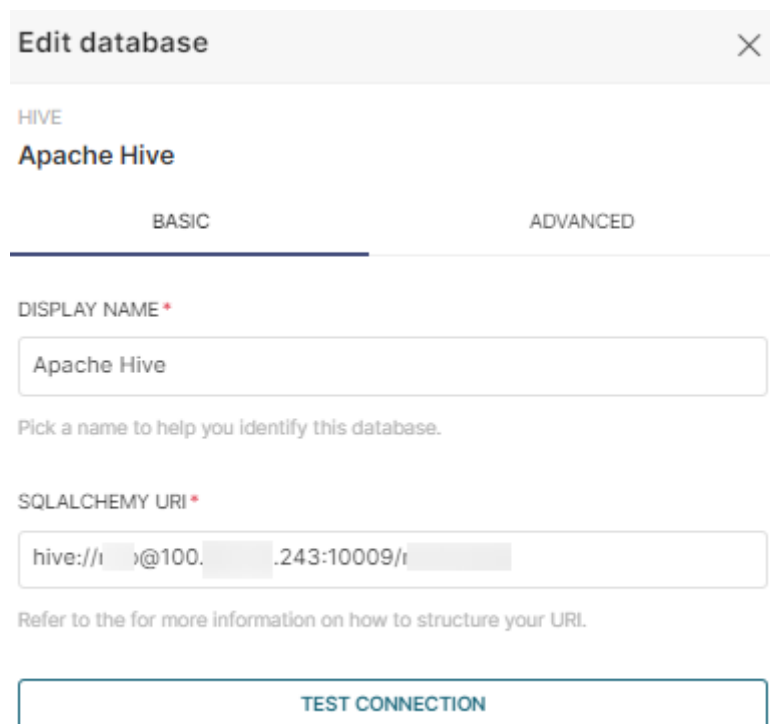
– **DISPLAY NAME:** Enter a data connection name.

– **SQL ALCHEMY URI:** Enter the URL of the configuration data connection.

Database type://username:password@host:port/database

Example: hive://username:password@100.xx.xxx.243:10009/tpch

**Figure 7-22** Configuring a Superset data connection



5. Click **TEST CONNECTION** to check whether the data source connection is successful. If successful, click **OK** to save the connection.

## Common Operations: Setting SQL Job Parameters

1. Make sure that the following parameters are enabled in the `/conf/kyuubi-defaults.conf` file of Kyuubi:

```
kyuubi.engine.dli.set.conf.transform.to.annotation=true
```

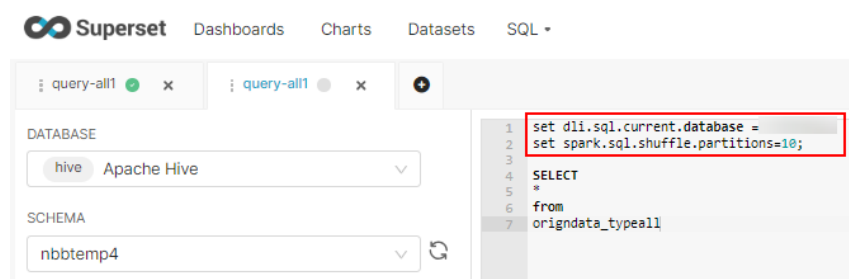
2. Add annotation parameters to the end of SQL statements.

*set Parameter*

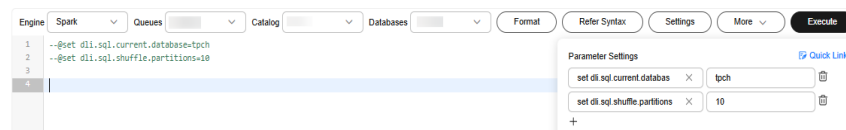
Example:

```
set dli.sql.current.database=tpch
set dli.sql.shuffle.partitions=10
```

**Figure 7-23** Example Superset parameter configuration



The execution result in the SQL Editor of DLI is as follows: The set parameters are changed to annotations and submitted to DLI for execution.

**Figure 7-24** Viewing the configured parameters in the SQL Editor of DLI

## 7.8 Connecting Tableau to DLI Using Kyuubi for Data Analysis and Query

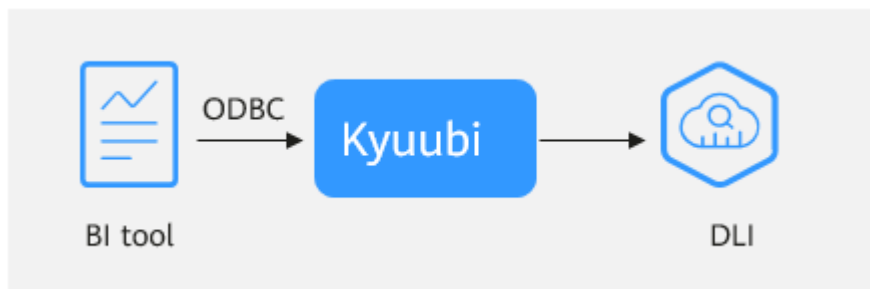
Tableau is a data analysis and visualization tool that allows you to connect to various data sources through a drag-and-drop interface. It enables the creation of interactive and shareable data visualizations, which can turn data into actionable insights.

Kyuubi is a distributed SQL query engine that provides standard SQL APIs. This allows you to easily access and analyze data stored on big data platforms.

By connecting Tableau to Kyuubi, you can use Kyuubi to access DLI for data analysis and query. This integration simplifies data access and provides unified data management and analysis, enabling you to gain deeper insights into data.

This section describes how to connect Tableau to DLI using Kyuubi to access and analyze data in DLI.

### Procedure

**Figure 7-25** Process

- **Step 1: Install and Connect Kyuubi to DLI:** Install and configure Kyuubi to ensure that Kyuubi can connect to DLI.
- **Step 2: Connect ODBC to Kyuubi:** Install Superset and the data connection driver.
- **Step 3: Connect Tableau to Kyuubi Using ODBC:** Create a data connection in the BI tool and connect to Kyuubi using JDBC.

### Step 1: Install and Connect Kyuubi to DLI

To access Kyuubi from an external network, ensure that an EIP is bound to the ECS and ports **10009** and **3309** are enabled in the inbound rules of the security group.

#### Step 1 Install JDK.



Install JDK in your development environment before you can install and use Kyuubi.

Java SDKs require JDK 1.8 or later. To ensure compatibility with future versions, it is recommended to use version 1.8.

1. Download the JDK.

Download the JDK 1.8 installation package from [Oracle official website](#) and install it.

In this example, **jdk-8u261-linux-x64.tar.gz** is used.

2. Upload the JDK package to the corresponding directory of the Linux server and run the extraction command. In this example, the package is uploaded to **/usr/local**.

```
sudo tar -xzf jdk-8u261-linux-x64.tar.gz -C /usr/local/
```

3. Configure environment variables.

Add the following content to the **.bashrc** or **.profile** file:

```
export JAVA_HOME=/usr/local/jdk-1.8.0_261
export PATH=$PATH:$JAVA_HOME/bin
```

4. Apply the environment variables.

```
source ~/.bashrc
```

5. Run **java -version** to check whether the installation is successful. If the version number is displayed in the command output, the Java environment is successfully installed.

```
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
```

## Step 2 Install Kyuubi.

1. [Download the Kyuubi installation package](#). Refer to [Deploying Kyuubi](#) to learn more how to install Kyuubi.

2. Extract the downloaded package.

```
tar -xzf kyuubi-{version}-bin.tar.gz
```

3. (Optional) Configure environment variables.

Add the bin directory of Kyuubi to PATH environment variables to ensure that Kyuubi scripts can be called at any location.

## Step 3 Connect Kyuubi to DLI.

1. Add the DLI driver to the root directory of Kyuubi.

Place the driver in Kyuubi's root directory **/externals/engines/jdbc**.

Ensure that the user group and permissions of the plug-in are the same as those of other JAR files.

2. Modify the Kyuubi configuration file.

```
cd $KYUUBI_HOME/confvi kyuubi-defaults.conf
```

[Table 7-13](#) describes the parameters.

**Table 7-13** Kyuubi parameters

Parameter	Description	Mandatory	Example Value
kyuubi.engine.type	JDBC service type. Set it to <b>dli</b> .	Yes	jdbc
kyuubi.engine.jdbc.type	Engine type. Set it to <b>dli</b> .	Yes	dli
kyuubi.engine.jdbc.driver.class	Name of the driver class used to connect JDBC. Set it to <b>com.huawei.dli.jdbc.DliDriver</b> .	Yes	com.huawei.dli.jdbc.DliDriver
kyuubi.engine.jdbc.connection.url	URL used to connect JDBC. Format: <b>jdbc:dli://<i>{dliendpoint}</i>/<i>{projectId}</i></b> .	Yes	jdbc:dli:// {dliendpoint} / {projectId}
kyuubi.engine.jdbc.session.initialize.sql	Initialization SQL statements executed when a JDBC session is established.	No	select 1 If <b>select 1</b> is displayed on the DLI management console, the initialization is successful.
kyuubi.frontend.protocols	Frontend protocol supported by Kyuubi. Kyuubi supports various frontend protocols, allowing you to interact with Kyuubi through different APIs.	Yes	- mysql - thrift_binary

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.schema.show.name	<p>How the Kyuubi engine shows the schema name of data source APIs when you run <b>show schemas</b> or <b>show databases</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: The DLI name is included as the prefix when the schema name is shown.</li> <li>- <b>false</b>: The DLI name is not included when the schema name is shown.</li> </ul> <p>For example, if it is set to <b>true</b> and the DLI name is <b>hive</b>, the output is in the <b>hive.default</b> format when you run <b>show schemas</b>.</p> <p>If it is set to <b>false</b>, the output is in the <b>default</b> format.</p>	No	<ul style="list-style-type: none"> <li>- true</li> <li>- false</li> </ul>
kyuubi.engine.dli.jdbc.connection.region	Region name.	Yes	regionname=ap-southeast-2
kyuubi.engine.dli.jdbc.connection.queue	DLI queue name.	Yes	dli_test
kyuubi.engine.dli.jdbc.connection.database	Default database name used when the Kyuubi engine connects to the DLI data source through JDBC.	Yes	tpch
kyuubi.engine.dli.jdbc.connection.accesskey	AK that acts as the authentication key. Set it if <b>authenticationmode</b> is <b>ask</b> .	Yes	accesskey=your-access-key
kyuubi.engine.dli.jdbc.connection.secretkey	Region name. Set it if <b>authenticationmode</b> is <b>ask</b> .	Yes	secretkey=your-secret-key
kyuubi.engine.dli.jdbc.connection.project	ID of the project where DLI resources are.	Yes	0b33ea2a7e0010802fe4c009bb05076d

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.sql.limit.time.sec	Execution duration limit of a SQL query. The default value is <b>600</b> seconds.	No	300
kyuubi.engine.dli.result.line.num.limit	Maximum number of data records returned for a SQL query. By default, 100,000 data records are returned. Setting it to <b>-1</b> means that the number of returned data records is not limited.	Yes	50000
kyuubi.engine.dli.small.file.merge	Whether to enable automatic small file merging. The default value is <b>false</b> , indicating that the function is disabled. - <b>true</b> : Enable - <b>false</b> : Disable	Yes	true
kyuubi.engine.dli.bi.type	BI tool type. The options are <b>fine</b> , <b>grafana</b> , <b>superset</b> , <b>tableau</b> , <b>power</b> , <b>dbt</b> , and <b>yongHong</b> .	Yes	fine
kyuubi.engine.dli.boolean.type.to.int	Whether DLI's Boolean data is returned as <b>1/0</b> or <b>true/false</b> . If the BI tool type is <b>grafana</b> , set this parameter to <b>true</b> . - <b>true</b> : returned as <b>1/0</b> (1: true; 0: false) - <b>false</b> : returned as <b>true/false</b> . The default value is <b>false</b> .	No	false
kyuubi.engine.dli.set.conf.transform.to.annotation	Allows you to set <b>set spark</b> in SQL statements. Set it to <b>true</b> for Power BI, FineBI, Superset, and DBT.	No	true

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.set.conf.sql.suffix	Allows you to set <b>set spark</b> at the end of SQL statements. Set it to <b>true</b> for Power BI and DBT.	No	true
kyuubi.engine.dli.result.cache.enable	Whether to enable database table data caching. Once enabled, database table data is automatically cached. The default value is <b>true</b> . - <b>true</b> : Enable - <b>false</b> : Disable	No	true
kyuubi.engine.dli.cache.limit.line.num	Maximum number of data records that can be cached. By default, 100,000 data records are cached. Setting it to <b>-1</b> means that the number of cached data records is not limited.	No	1000
kyuubi.engine.dli.cache.time.sec	Cache time. The default value is <b>1800</b> seconds.	No	1800
kyuubi.operation.incremental.collect	Kyuubi pre-loads select result data into the cache to accelerate data reading. You are advised to disable this function to avoid memory OOM issues when dealing with large volumes of data.	No	false Setting it to <b>false</b> indicates disabling preloading.
kyuubi.engine.jdbc.memory	Memory of the JDBC engine process The default value is <b>1g</b> . You are advised to set it to at least <b>5g</b> to increase memory for the JDBC engine process.	No	5g

3. Quickly start Kyuubi.

Go to the root directory **/bin** of the ECS and run the following command to start Kyuubi:

```
cd /bin  
./kyuubi start restart
```

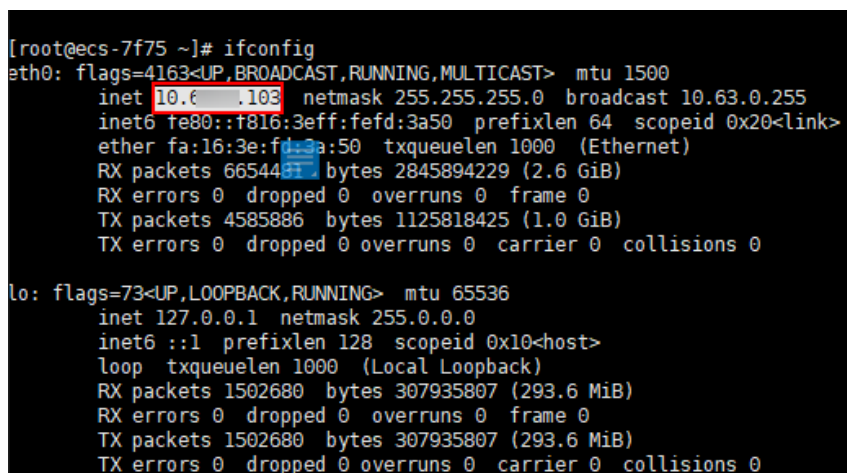
Once the start is successful, run SQL statements to check whether the connection between Kyuubi and DLI is normal.

#### Step 4 (Optional) Configure the hosts file to speed up access to Kyuubi.

To make access to Kyuubi faster, you are advised to add the Kyuubi host IP address to the `/etc/hosts` file on the host.

1. Run `ifconfig` to check the IP address of the host.

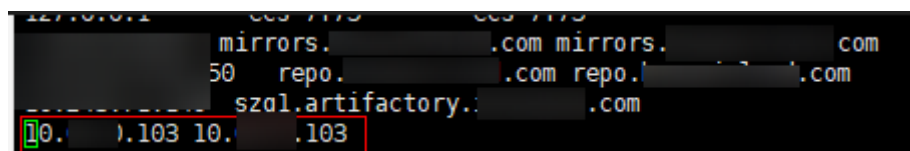
Figure 7-26 Checking the host IP address



```
[root@ecs-7f75 ~]# ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 10.63.0.103 netmask 255.255.255.0 broadcast 10.63.0.255  
    inet6 fe80::f816:3eff:febd:3a50 prefixlen 64 scopeid 0x20<link>  
    ether fa:16:3e:fd:3a:50 txqueuelen 1000 (Ethernet)  
    RX packets 6654431 bytes 2845894229 (2.6 GiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 4585886 bytes 1125818425 (1.0 GiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 1502680 bytes 307935807 (293.6 MiB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 1502680 bytes 307935807 (293.6 MiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2. Add the IP address to the `/etc/hosts` file.

Figure 7-27 Adding the host IP address to the `/etc/hosts` file



```
127.0.0.1 localhost  
... mirrors. ....com mirrors. ....com  
... 50 repo. ....com repo. ....com  
... szol.artifactory. ....com  
10.63.0.103 10.63.0.103
```

----End

## Step 2: Connect ODBC to Kyuubi

### Step 1 Install the ODBC driver.

Install the ODBC driver on the local host based on the database type. In this example, the Hive database is used.

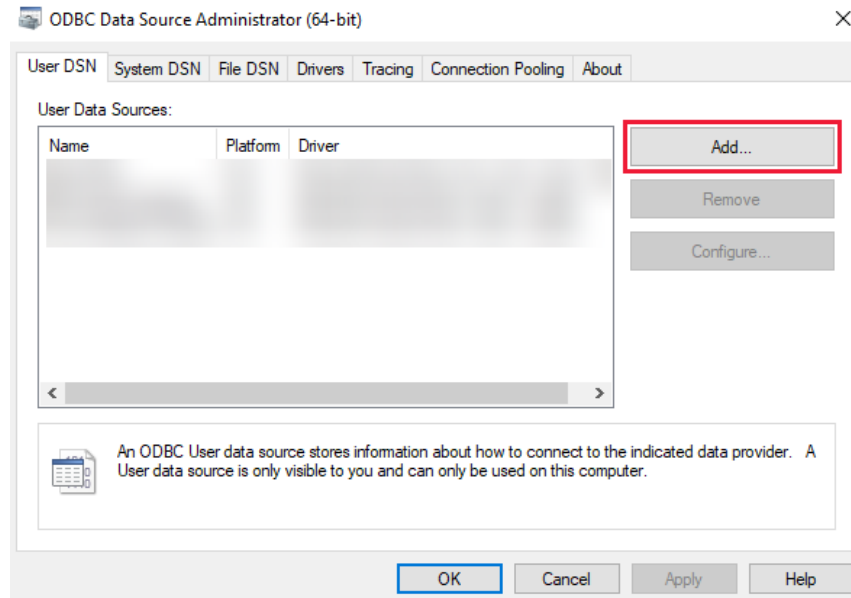
- [Cloudera Hive ODBC](#). Version 2.5.12 is recommended.
- [Microsoft Hive ODBC](#). Version 2.6.12.1012 is recommended.

### Step 2 Connect ODBC to Kyuubi.

1. In Windows, click **Control Panel**, double-click **Administrative Tools**, and double-click **Data Sources (ODBC)**.
2. Configure an ODBC data source.

- a. Click **User DSN**.
- b. Click **Add**.
- c. Select **Hive ODBC Driver** and click **OK**.

**Figure 7-28** Creating an ODBC data source connection



- 3. In the dialog box that appears, enter information about the Kyuubi server.
  - **Database Name:** Enter the DLI database name.
  - **Server Address:** Enter the EIP of the Kyuubi server.
  - **Port:** listening port of Kyuubi. The Hive Thrift protocol is used. The default port is **10009**.
  - **Username and Password:** Configure the username and password of the Kyuubi server as required.

Set other advanced options and save the configuration.

Figure 7-29 ODBC data source connection parameters

The screenshot shows the ODBC data source configuration dialog. The 'Description' field contains 'Sample Microsoft Hive DSN'. The 'Host(s)' field is empty and highlighted with a red box. The 'Port' field contains '10009' and is also highlighted with a red box. The 'Database' field contains 'nbbtemp'. Under the 'Authentication' section, the 'Mechanism' dropdown is set to 'User Name' and is highlighted with a red box. The 'User Name' field is empty and highlighted with a red box. Other fields include 'Realm', 'Host FQDN' (set to '\_HOST'), 'Service Name' (set to 'hive'), 'Canonicalize Principal FQDN' (checked), and 'Delegate Kerberos Credentials' (unchecked). There are also fields for 'Password', 'Delegation UID', 'Thrift Transport' (set to 'SASL'), and several options buttons like 'Proxy Options...', 'HTTP Options...', 'SSL Options...', 'Advanced Options...', and 'Logging Options...'. At the bottom, there are 'Test', 'OK', and 'Cancel' buttons, and the version 'v2.6.12.1012 (64 bit)' is displayed.

4. Click **Test** to check whether the data source connection is successful. If successful, click **OK** to save the connection.

----End

### Step 3: Connect Tableau to Kyuubi Using ODBC

1. Click and install Tableau. [Download the Tableau installation package.](#)



2. Open Tableau.
3. In the **Connect** pane of the **Start** page, select the type of data source you want to connect to. In this example, **Hive** is selected.
4. Configure data connection information.
  - **Connection: Hive.**
  - **Server:** IP address of the Kyuubi host.
  - **Port:** port used to connect to Kyuubi. The Hive Thrift protocol is used for interconnection. The default port is **10009**.
  - **Authentication:** Select **Username** for this example.
  - **Username:** Kyuubi username.
5. Click **Log In** to connect to Kyuubi.

## Common Operations: Setting SQL Job Parameters

Add annotation parameters to the end of SQL statements.

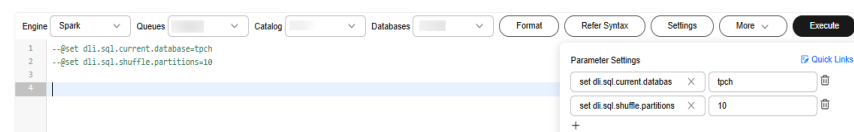
```
-- @set Parameter  
Example:  
-- @set dli.sql.current.database=tpch  
-- @set dli.sql.shuffle.partitions=10
```

Figure 7-30 Example Tableau parameter configuration

```
-- @set dli.sql.current.database=  
-- @set spark.sql.shuffle.partitions=  
  
select  
*  
from  
origndata_typeall
```

The execution result in the SQL Editor of DLI is as follows: The set parameters are changed to annotations and submitted to DLI for execution.

Figure 7-31 Viewing the configured parameters in the SQL Editor of DLI



## 7.9 Connecting Beeline to DLI Using Kyuubi for Data Analysis and Query

Beeline is a crucial tool for data analysts and engineers, especially in scenarios that involve large-scale data processing. With its SQL engine, Beeline enables you to execute data queries, analysis, and management tasks using SQL language.

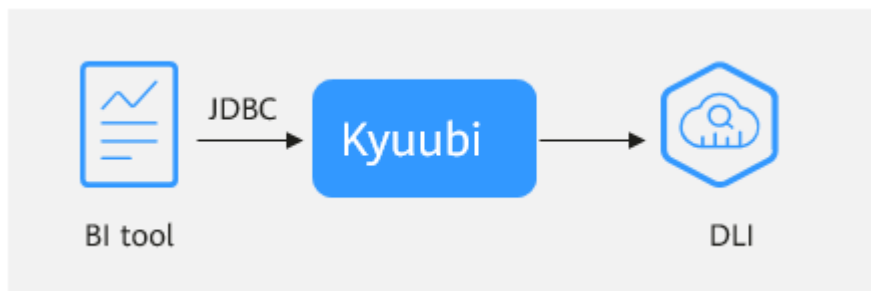
Kyuubi is a distributed SQL query engine that provides standard SQL APIs. This allows you to easily access and analyze data stored on big data platforms.

By connecting Beeline to Kyuubi, you can use Kyuubi to access DLI for data analysis and query. This integration simplifies data access and provides unified data management and analysis, enabling you to gain deeper insights into data.

This section describes how to connect Beeline to DLI using Kyuubi to access and analyze data in DLI.

## Procedure

Figure 7-32 Process



- **Step 1: Install and Connect Kyuubi to DLI:** Install and configure Kyuubi to ensure that Kyuubi can connect to DLI.
- **Step 2: Connect Beeline to Kyuubi:** Create a data connection in the BI tool and connect to Kyuubi using JDBC.

### Step 1: Install and Connect Kyuubi to DLI

To access Kyuubi from an external network, ensure that an EIP is bound to the ECS and ports **10009** and **3309** are enabled in the inbound rules of the security group.

#### Step 1 Install JDK.

Install JDK in your development environment before you can install and use Kyuubi.

Java SDKs require JDK 1.8 or later. To ensure compatibility with future versions, it is recommended to use version 1.8.

1. Download the JDK.  
Download the JDK 1.8 installation package from [Oracle official website](#) and install it.  
In this example, **jdk-8u261-linux-x64.tar.gz** is used.
2. Upload the JDK package to the corresponding directory of the Linux server and run the extraction command. In this example, the package is uploaded to **/usr/local**.  

```
sudo tar -xzf jdk-8u261-linux-x64.tar.gz -C /usr/local/
```
3. Configure environment variables.  
Add the following content to the **.bashrc** or **.profile** file:  

```
export JAVA_HOME=/usr/local/jdk-1.8.0_261
export PATH=$PATH:$JAVA_HOME/bin
```
4. Apply the environment variables.  

```
source ~/.bashrc
```

- Run **java -version** to check whether the installation is successful. If the version number is displayed in the command output, the Java environment is successfully installed.

```
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
```

### Step 2 Install Kyuubi.

- [Download the Kyuubi installation package](#). Refer to [Deploying Kyuubi](#) to learn more how to install Kyuubi.
- Extract the downloaded package.  

```
tar -xzf kyuubi-{version}-bin.tar.gz
```
- (Optional) Configure environment variables.  
 Add the bin directory of Kyuubi to PATH environment variables to ensure that Kyuubi scripts can be called at any location.

### Step 3 Connect Kyuubi to DLI.

- Add the DLI driver to the root directory of Kyuubi.  
 Place the driver in Kyuubi's root directory **/externals/engines/jdbc**.  
 Ensure that the user group and permissions of the plug-in are the same as those of other JAR files.
- Modify the Kyuubi configuration file.  

```
cd $KYUUBI_HOME/conf/vi kyuubi-defaults.conf
```

[Table 7-14](#) describes the parameters.

**Table 7-14** Kyuubi parameters

Parameter	Description	Mandatory	Example Value
kyuubi.engine.type	JDBC service type. Set it to <b>dli</b> .	Yes	jdbc
kyuubi.engine.jdbc.type	Engine type. Set it to <b>dli</b> .	Yes	dli
kyuubi.engine.jdbc.driver.class	Name of the driver class used to connect JDBC. Set it to <b>com.huawei.dli.jdbc.DliDriver</b> .	Yes	com.huawei.dli.jdbc.DliDriver
kyuubi.engine.jdbc.connection.url	URL used to connect JDBC. Format: <b>jdbc:dli:// {dliendpoint} / {projectId}</b> .	Yes	jdbc:dli://{dliendpoint} / {projectId}

Parameter	Description	Mandatory	Example Value
kyuubi.engine.jdbc.session.initialize.sql	Initialization SQL statements executed when a JDBC session is established.	No	select 1 If <b>select 1</b> is displayed on the DLI management console, the initialization is successful.
kyuubi.frontend.protocols	Frontend protocol supported by Kyuubi. Kyuubi supports various frontend protocols, allowing you to interact with Kyuubi through different APIs.	Yes	<ul style="list-style-type: none"> <li>- mysql</li> <li>- thrift_binary</li> </ul>
kyuubi.engine.dli.schema.show.name	<p>How the Kyuubi engine shows the schema name of data source APIs when you run <b>show schemas</b> or <b>show databases</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: The DLI name is included as the prefix when the schema name is shown.</li> <li>- <b>false</b>: The DLI name is not included when the schema name is shown.</li> </ul> <p>For example, if it is set to <b>true</b> and the DLI name is <b>hive</b>, the output is in the <b>hive.default</b> format when you run <b>show schemas</b>. If it is set to <b>false</b>, the output is in the <b>default</b> format.</p>	No	<ul style="list-style-type: none"> <li>- true</li> <li>- false</li> </ul>
kyuubi.engine.dli.jdbc.connection.region	Region name.	Yes	regionname=ap-southeast-2
kyuubi.engine.dli.jdbc.connection.queue	DLI queue name.	Yes	dli_test
kyuubi.engine.dli.jdbc.connection.database	Default database name used when the Kyuubi engine connects to the DLI data source through JDBC.	Yes	tpch

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.jdbc.connection.accesskey	AK that acts as the authentication key. Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	accesskey=your-access-key
kyuubi.engine.dli.jdbc.connection.secretkey	Region name. Set it if <b>authenticationmode</b> is <b>aksk</b> .	Yes	secretkey=your-secret-key
kyuubi.engine.dli.jdbc.connection.project	ID of the project where DLI resources are.	Yes	0b33ea2a7e0010802fe4c009bb05076d
kyuubi.engine.dli.sql.limit.time.sec	Execution duration limit of a SQL query. The default value is <b>600</b> seconds.	No	300
kyuubi.engine.dli.result.line.number.limit	Maximum number of data records returned for a SQL query. By default, 100,000 data records are returned. Setting it to <b>-1</b> means that the number of returned data records is not limited.	Yes	50000
kyuubi.engine.dli.small.file.merge	Whether to enable automatic small file merging. The default value is <b>false</b> , indicating that the function is disabled. - <b>true</b> : Enable - <b>false</b> : Disable	Yes	true
kyuubi.engine.dli.bi.type	BI tool type. The options are <b>fine</b> , <b>grafana</b> , <b>superset</b> , <b>tableau</b> , <b>power</b> , <b>dbt</b> , and <b>yongHong</b> .	Yes	fine

Parameter	Description	Mandatory	Example Value
kyuubi.engine.dli.boolean.type.to.int	<p>Whether DLI's Boolean data is returned as <b>1/0</b> or <b>true/false</b>.</p> <p>If the BI tool type is <b>grafana</b>, set this parameter to <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: returned as <b>1/0</b> (1: true; 0: false)</li> <li>- <b>false</b>: returned as <b>true/false</b>.</li> </ul> <p>The default value is <b>false</b>.</p>	No	false
kyuubi.engine.dli.set.conf.transform.to.annotation	<p>Allows you to set <b>set spark</b> in SQL statements.</p> <p>Set it to <b>true</b> for Power BI, FineBI, Superset, and DBT.</p>	No	true
kyuubi.engine.dli.set.conf.sql.suffix	<p>Allows you to set <b>set spark</b> at the end of SQL statements.</p> <p>Set it to <b>true</b> for Power BI and DBT.</p>	No	true
kyuubi.engine.dli.result.cache.enable	<p>Whether to enable database table data caching. Once enabled, database table data is automatically cached. The default value is <b>true</b>.</p> <ul style="list-style-type: none"> <li>- <b>true</b>: Enable</li> <li>- <b>false</b>: Disable</li> </ul>	No	true
kyuubi.engine.dli.cache.limit.line.number	<p>Maximum number of data records that can be cached.</p> <p>By default, 100,000 data records are cached.</p> <p>Setting it to <b>-1</b> means that the number of cached data records is not limited.</p>	No	1000
kyuubi.engine.dli.cache.time.sec	<p>Cache time.</p> <p>The default value is <b>1800</b> seconds.</p>	No	1800

Parameter	Description	Mandatory	Example Value
kyuubi.operation.incremental.collect	Kyuubi pre-loads select result data into the cache to accelerate data reading. You are advised to disable this function to avoid memory OOM issues when dealing with large volumes of data.	No	false Setting it to <b>false</b> indicates disabling preloading.
kyuubi.engine.jdbc.memory	Memory of the JDBC engine process  The default value is <b>1g</b> . You are advised to set it to at least <b>5g</b> to increase memory for the JDBC engine process.	No	5g

3. Quickly start Kyuubi.

Go to the root directory **/bin** of the ECS and run the following command to start Kyuubi:

```
cd /bin
./kyuubi start restart
```

Once the start is successful, run SQL statements to check whether the connection between Kyuubi and DLI is normal.

**Step 4 (Optional) Configure the hosts file to speed up access to Kyuubi.**

To make access to Kyuubi faster, you are advised to add the Kyuubi host IP address to the **/etc/hosts** file on the host.

1. Run **ifconfig** to check the IP address of the host.

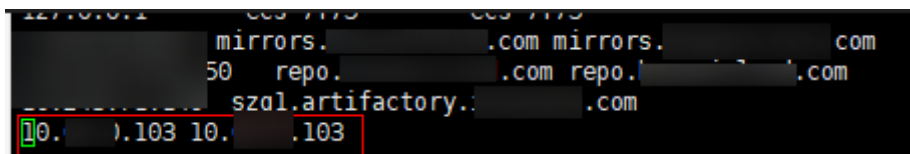
**Figure 7-33** Checking the host IP address

```
[root@ecs-7f75 ~]# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.63.0.103 netmask 255.255.255.0 broadcast 10.63.0.255
    inet6 fe80::f816:3eff:febd:3a50 prefixlen 64 scopeid 0x20<link>
    ether fa:16:3e:fd:3a:50 txqueuelen 1000 (Ethernet)
    RX packets 6654437 bytes 2845894229 (2.6 GiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4585886 bytes 1125818425 (1.0 GiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1502680 bytes 307935807 (293.6 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1502680 bytes 307935807 (293.6 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

2. Add the IP address to the **/etc/hosts** file.

**Figure 7-34** Adding the host IP address to the /etc/hosts file



----End

## Step 2: Connect Beeline to Kyuubi

Connect Kyuubi Beeline to Kyuubi Server.

```
kyuubi-beeline -n user1 -u "jdbc:hive2://<kyuubi-server-host>:<port>/"
```

- <kyuubi-server-host>: host name or IP address of Kyuubi Server.
- <port>: listening port of Kyuubi Server, which is **10009** by default.

Example configuration:

```
kyuubi-beeline -n user1 -u "jdbc:hive2://kyuubi-server-1:10009/"
```

## Common Operations: Setting SQL Job Parameters

1. Make sure that the following parameters are enabled in the `/conf/kyuubi-defaults.conf` file of Kyuubi:

```
kyuubi.engine.dli.set.conf.transform.to.annotation=true
```

2. Add annotation parameters to the end of SQL statements.

```
set Parameter
```

Example:

```
set dli.sql.current.database=tpch
set dli.sql.shuffle.partitions=10
```

The execution result in the SQL Editor of DLI is as follows: The set parameters are changed to annotations and submitted to DLI for execution.

**Figure 7-35** Viewing the configured parameters in the SQL Editor of DLI

