

Build and Execute MySQL, PostgreSQL, and SQLServer to Data Catalog Connectors

GSP814



Google Cloud Self-Paced Labs

Overview

[Data Catalog](#) is a fully managed and scalable metadata management service that empowers organizations to quickly discover, understand, and manage all their data. It offers a simple and easy-to-use search interface for data discovery, a flexible and powerful cataloging system for capturing both technical and business metadata, and a strong security and compliance foundation with Cloud Data Loss Prevention (DLP) and Cloud Identity and Access Management (IAM) integrations.

Using Data Catalog

There are two main ways you interact with Data Catalog:

- Searching for data assets that you have access to.
- Tagging assets with metadata.

What you will learn

In this lab, you will learn how to:

- Enable the Data Catalog API so that you can use this service in your Google Cloud project.
- Execute SQLServer to Data Catalog connector.
- Execute PostgreSQL to Data Catalog connector.
- Execute MySQL to Data Catalog connector.

Prerequisites

Very Important: Before starting this lab, log out of your personal or corporate gmail account, or run this lab in Incognito. This prevents sign-in confusion while the lab is running.

Setup and requirements

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

What you need

To complete this lab, you need:

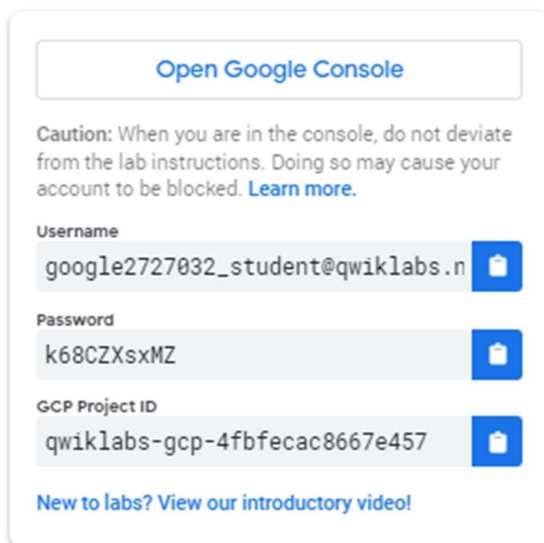
- Access to a standard internet browser (Chrome browser recommended).
- Time to complete the lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab.

Note: If you are using a Pixelbook, open an Incognito window to run this lab.

How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



The screenshot shows a sign-in panel with the following elements:

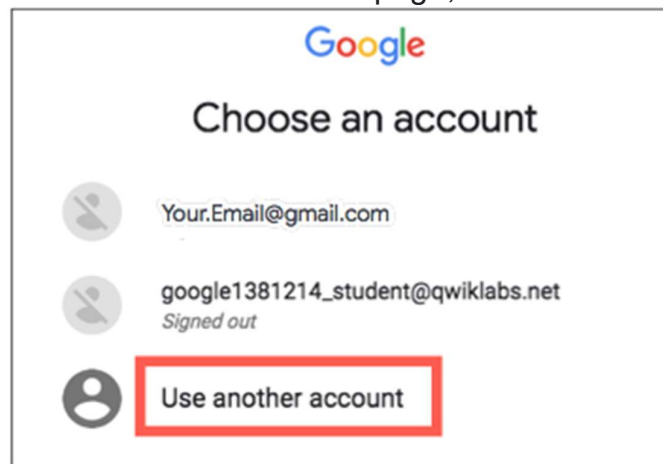
- A button at the top labeled "Open Google Console".
- A caution message: "Caution: When you are in the console, do not deviate from the lab instructions. Doing so may cause your account to be blocked. [Learn more.](#)"
- Three input fields, each with a copy icon to its right:
 - Username:** google2727032_student@qwiklabs.n
 - Password:** k68CZXsxMZ
 - GCP Project ID:** qwiklabs-gcp-4fbfecac8667e457
- A link at the bottom: "New to labs? View our introductory video!"

2. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



Tip: Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another**



Account.

3. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

Important: You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

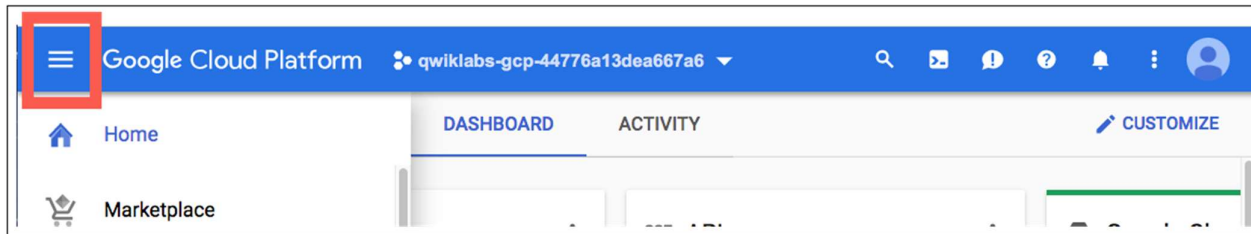
4. Click through the subsequent pages:

- Accept the terms and conditions.
- Do not add recovery options or two-factor authentication (because this is a temporary account).
- Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-

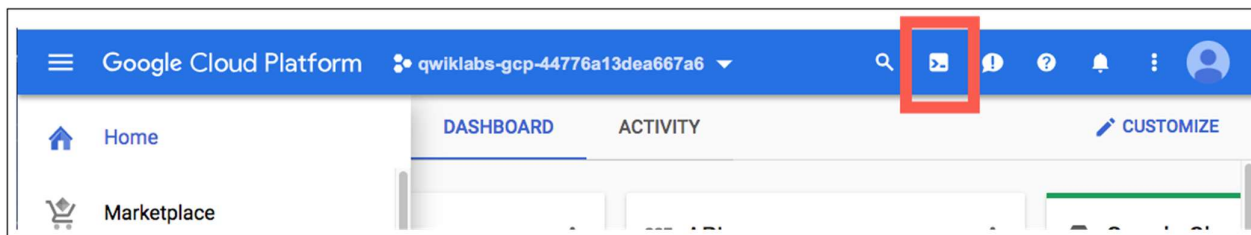
left.



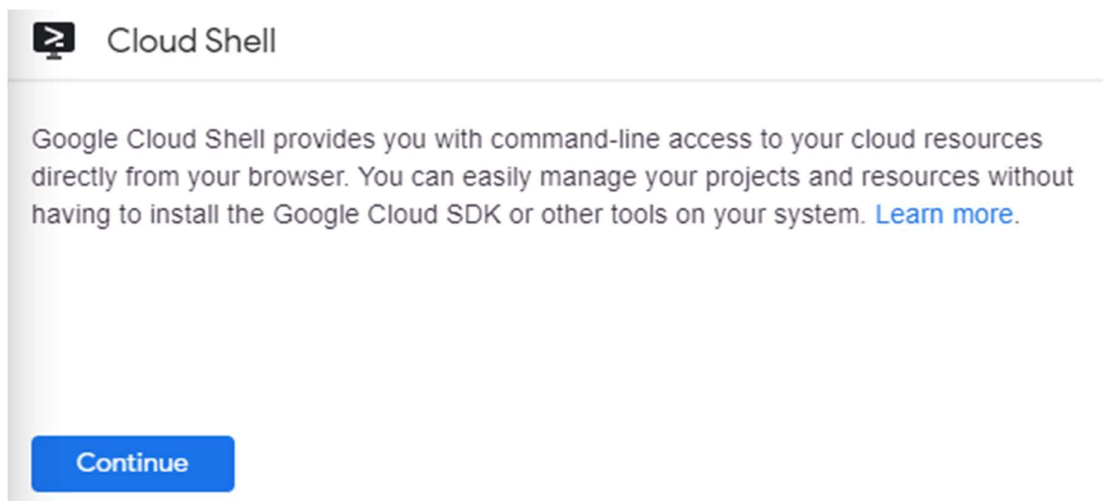
Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

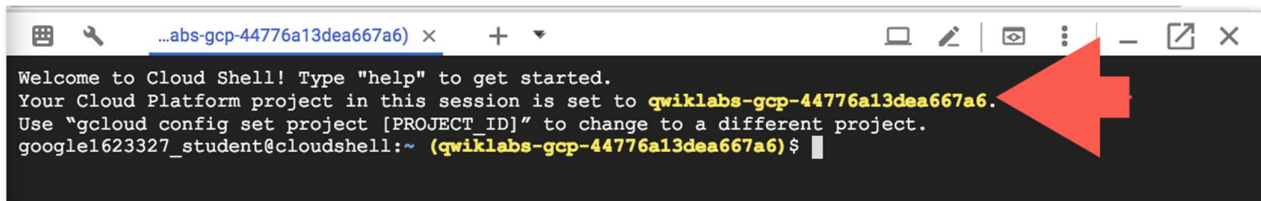
In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



Click **Continue**.



It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT_ID*. For example:



```
...abs-gcp-44776a13dea667a6) x + ▾
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to qwiklabs-gcp-44776a13dea667a6.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
google1623327_student@cloudshell:~ (qwiklabs-gcp-44776a13dea667a6) $
```

`gcloud` is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

You can list the active account name with this command:

```
gcloud auth list
```

(Output)

```
Credentialed accounts:
- <myaccount>@<mydomain>.com (active)
```

(Example output)

```
Credentialed accounts:
- google1623327_student@qwiklabs.net
```

You can list the project ID with this command:

```
gcloud config list project
```

(Output)

```
[core]
project = <project ID>
```

(Example output)

```
[core]
project = qwiklabs-gcp-44776a13dea667a6
```

For full documentation of `gcloud` see the [gcloud command-line tool overview](#).

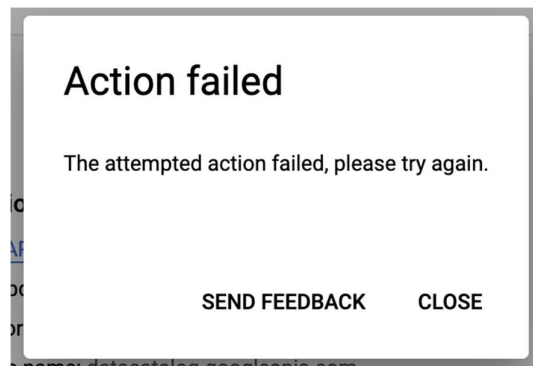
Enable the Data Catalog API

Open the **Navigation menu** and select **APIs and Services > Library**.

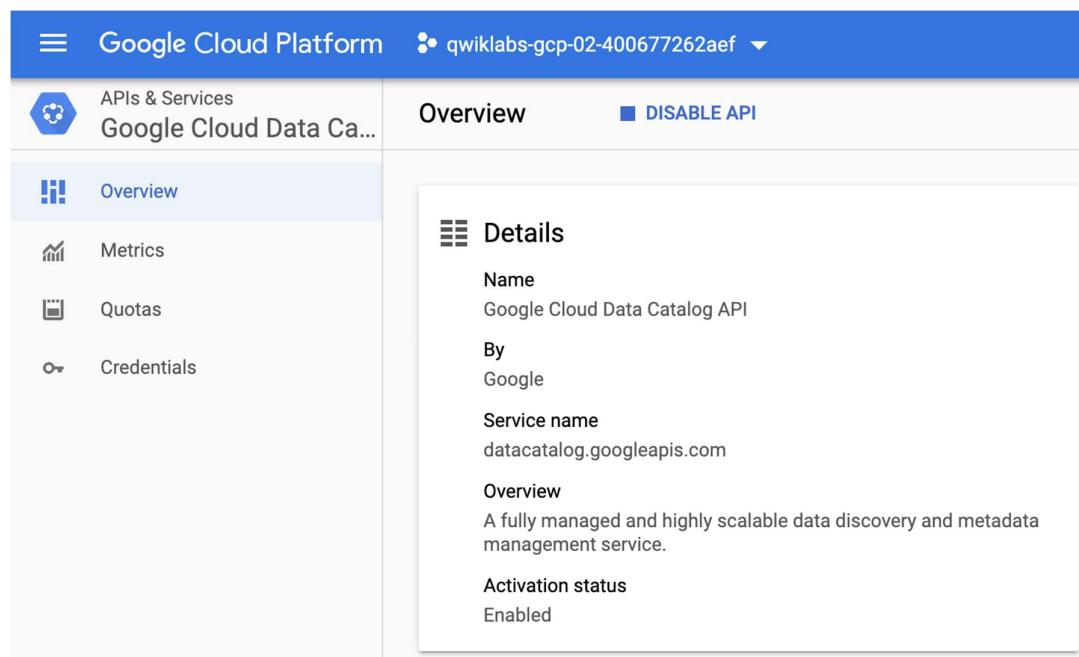
In the search bar, enter in "Data Catalog" and select the Google Cloud Data Catalog API.

Then click **Enable**.

If you run into the following error after trying to enable the Data Catalog API:



Click **Close** and **refresh** your browser tab. Then click **Enable** again. The Data Catalog API should be successfully enabled:



SQLServer to Data Catalog

Start by setting up your environment. Run the following command to set your Project ID, replacing `<YOUR_PROJECT_ID>` with the Project ID found in the connection details panel:

```
gcloud config set project <YOUR_PROJECT_ID>
```

Next set it as an environment variable:

```
export PROJECT_ID=$(gcloud config get-value project)
```

Create the SQLServer Database

You can use the open source scripts at [Cloud SQL SQLServer Tooling](https://github.com/mesmacosta/cloudsql-sqlserver-tooling) to create and populate your SQLServer instance.

In your Cloud Shell session, run the following command to clone the required repository:

```
git clone https://github.com/mesmacosta/cloudsql-sqlserver-tooling
```

Now change your current working directory to the cloned repo directory:

```
cd cloudsql-sqlserver-tooling
```

Now run the `init-db.sh` script.

```
source init-db.sh
```

This will create your SQLServer instance and populate it with random schema.

This will take around five minutes to complete. You can move on when you receive the following output:

```
CREATE TABLE factory_warehouse15797.employees53b82dc5 ( school180581 REAL, reason91250  
DATETIME, randomdata32431 BINARY, phone_number52754 REAL, person66471 REAL,  
credit_card75527 DATETIME )  
  
COMPLETED
```

Set Up the Service Account

Run the following command to create a Service Account:

```
gcloud iam service-accounts create sqlserver2dc-credentials \  
--display-name "Service Account for SQLServer to Data Catalog connector" \  
--project $PROJECT_ID
```


Now create and download the Service Account Key.

```
gcloud iam service-accounts keys create "sqlserver2dc-credentials.json" \
--iam-account "sqlserver2dc-credentials@$PROJECT_ID.iam.gserviceaccount.com"
```

Add the Data Catalog admin role to the Service Account:

```
gcloud projects add-iam-policy-binding $PROJECT_ID \
--member "serviceAccount:sqlserver2dc-credentials@$PROJECT_ID.iam.gserviceaccount.com" \
--quiet \
--project $PROJECT_ID \
--role "roles/datacatalog.admin"
```

Execute SQLServer to Data Catalog connector.

You can build the SQLServer connector yourself by going to [this GitHub repository](#). To facilitate its usage, we are going to use a docker image. The environment variables were loaded by the `init-db.sh` script.

Run the following command to execute the connector:

```
docker run --rm --tty -v \
"$PWD":/data mesmacosta/sqlserver2datacatalog:stable \
--datacatalog-project-id=$PROJECT_ID \
--datacatalog-location-id=us-central1 \
--sqlserver-host=$public_ip_address \
--sqlserver-user=$username \
--sqlserver-pass=$password \
--sqlserver-database=$database
```


Soon after you should receive the following output:

```
=====End sqlserver-to-datacatalog=====
```

Search for the SQLServer Entries in Data Catalog

After the script finishes, open the navigation menu and select **Data Catalog** from the list of services.

In the search bar, enter **sqlserver**. You should see the following Tag Templates and Entry Group appear in the results:

 Data Catalog

[← Search](#) [+ CREATE](#)

Sort by
Relevance

Systems

Data types

☐ Include public datasets

Now click on the `sqlserver` Entry Group. Your console should resemble the following:

[← Entry group details](#) [EDIT ENTRY GROUP](#)

sqlserver

ENTRIES

DETAILS

[+ CREATE](#) [EDIT](#) [DELETE](#)

<input type="checkbox"/>	Name
<input type="checkbox"/>	factory_warehouse18982
<input type="checkbox"/>	companies97f27849
<input type="checkbox"/>	personal_info6dd06e2a
<input type="checkbox"/>	personsd75f7c3e
<input type="checkbox"/>	storecd34f3a2
<input type="checkbox"/>	stored7dc05c5
<input type="checkbox"/>	factory_warehouse82240
<input type="checkbox"/>	home08b8338b
<input type="checkbox"/>	school_infobecb6e60
<input type="checkbox"/>	school_infod050b0d9

This is the real value of an Entry Group—you can see all entries that belong to `sqlserver` using the UI.

Click on one of the `warehouse` entries. Look at the Custom entry details and tags:

Custom entry details and Tags:

Custom entry details

Display name	factory_warehouse3455	Location	us-central1
Type	schema	Resource URL	://35.239.240.172/factory_warehouse3455
System	sqlserver	Description	
Creation time			
Last modification time			
Expiration time	Never		

Tags (1)

Sqlserver Schema - Metadata

Display name	Value	Type
Number of tables	5	DOUBLE

This is the real value the connector adds — it allows you to have the metadata searchable in Data Catalog.

Cleaning up

To delete the created resources, run the following command to delete the SQLServer metadata:

```
./cleanup-db.sh
```

Now execute the cleaner container:

```
docker run --rm --tty -v \
"$PWD":/data mesmacosta/sqlserver-datacatalog-cleaner:stable \
--datacatalog-project-ids=$PROJECT_ID \
--rdbms-type=sqlserver \
--table-container-type=schema
```

Now run the following command to delete the SQLServer database:

```
./delete-db.sh
```

From the **Navigation menu** click **Data Catalog**. Search for **sqlserver**. You will no longer see the SQLServer Tag Templates in the results:

Data Catalog Search **+ CREATE**

sqlserver

Sort by: Relevance Systems Data types ☐ Include public datasets

Name	Description	Type	System	Project	Last modified
No rows to display					

Ensure you see the following output in Cloud Shell before you move on:

```
Cloud SQL Instance deleted
COMPLETED
```

You will now learn how to do the same thing with a PostgreSQL instance.

PostgreSQL to Data Catalog

Create the PostgreSQL Database

You can use the open source scripts at [Cloud SQL PostgreSQL Tooling](https://github.com/mesmacosta/cloudsql-postgresql-tooling) to create and populate your PostgreSQL instance.

Run the following command in Cloud Shell to return to your home directory:

```
cd
```

Run the following command to clone the Github repository:

```
git clone https://github.com/mesmacosta/cloudsql-postgresql-tooling
```

Now change your current working directory to the cloned repo directory:

```
cd cloudsql-postgresql-tooling
```

Now execute the `init-db.sh` script:

```
source init-db.sh
```

This will create your PostgreSQL instance and populate it with random schema. Soon after you should receive the following output:

```
CREATE TABLE factory_warehouse69945.home17e97c57 ( house57588 DATE, paragraph64180  
SMALLINT, ip_address61569 JSONB, date_time44962 REAL, food19478 JSONB, state8925  
VARCHAR(25), cpf75444 REAL, date_time96090 SMALLINT, reason7955 CHAR(5),  
phone_number96292 INT, size97593 DATE, date_time609 CHAR(5), location70431 DATE )  
COMPLETED
```

Set Up the Service Account

Create a Service Account.

```
gcloud iam service-accounts create postgresql2dc-credentials \  
--display-name "Service Account for PostgreSQL to Data Catalog connector" \  
--project $PROJECT_ID
```

Next create and download the Service Account Key.

```
gcloud iam service-accounts keys create "postgresql2dc-credentials.json" \  
--iam-account "postgresql2dc-credentials@$PROJECT_ID.iam.gserviceaccount.com"
```

Next add Data Catalog admin role to the Service Account.

```
gcloud projects add-iam-policy-binding $PROJECT_ID \  
--member "serviceAccount:postgresql2dc-credentials@$PROJECT_ID.iam.gserviceaccount.com" \  
\
```

```
--quiet \  
--project $PROJECT_ID \  
--role "roles/datacatalog.admin"
```

Execute PostgreSQL to Data Catalog connector

You can build the PostgreSQL connector yourself by going to [this GitHub repository](#). To facilitate its usage, we are going to use a docker image. The environment variables were loaded by the `init-db.sh` script.

Execute the connector:

```
docker run --rm --tty -v \  
"$PWD":/data mesmacosta/postgresql2datacatalog:stable \  
--datacatalog-project-id=$PROJECT_ID \  
--datacatalog-location-id=us-central1 \  
--postgresql-host=$public_ip_address \  
--postgresql-user=$username \  
--postgresql-pass=$password \  
--postgresql-database=$database
```

Soon after you should receive the following output:

```
=====End postgresql-to-datacatalog=====
```

Check the results of the script

Ensure that you are in the Data Catalog home page.

In the search bar, enter **PostgreSQL**. You should see the following Tag Templates and Entry Group appear in the results:

Data Catalog

Search

CREATE

PostgreSQL

Sort by

Relevance

Systems

Data types

☐

Include public datasets

Name	Description	Type	System	Project	Last modified
Postgresql Table - Metadata	-	Tag template	Data Catalog	qwiklabs-gcp-02-5ed5ce1fe757	Feb 1, 2021
Postgresql Schema - Metadata	-	Tag template	Data Catalog	qwiklabs-gcp-02-5ed5ce1fe757	Feb 1, 2021
postgresql	-	Entry group	Data Catalog	qwiklabs-gcp-02-5ed5ce1fe757	
personal_infoeaa1b52d	-	Table	postgresql	-	
home8d2351aa	-	Table	postgresql	-	
home5d67a108	-	Table	postgresql	-	
home52cae506	-	Table	postgresql	-	
employeeesc51aabe2	-	Table	postgresql	-	
factory_warehouse57631	-	schema	postgresql	qwiklabs-gcp-02-5ed5ce1fe757	

Now click on the `postgresql` Entry Group. Your console should resemble the following:

[←](#) Entry group details [EDIT ENTRY GROUP](#)

postgresql

[ENTRIES](#) [DETAILS](#)

[+ CREATE](#) [EDIT](#) [DELETE](#)

<input type="checkbox"/>	Name
<input type="checkbox"/>	company_warehouse20525
<input type="checkbox"/>	companies8e9c683a
<input type="checkbox"/>	companiesa854f044
<input type="checkbox"/>	personal_info1a1af47c
<input type="checkbox"/>	personal_infoeba36bc3
<input type="checkbox"/>	store737c146e
<input type="checkbox"/>	company_warehouse76538
<input type="checkbox"/>	home8abf9baa
<input type="checkbox"/>	personal_info16f42d0e
<input type="checkbox"/>	personal_info62b97ce8

This is the real value of an Entry Group — you can see all entries that belong to `postgresql` using the UI.

Click on one of the `warehouse` entries. Look at the Custom entry details and tags:

Custom entry details and Tags:

Data Catalog

[←](#) company_warehouse36062 [+ ATTACH TAGS](#)

qwiklabs-gcp-02-5ed5ce1fe757 > us-central1 > postgresql

Filter tags and schema

Custom entry details

Tags

Custom entry details

Display name	company_warehouse36062	Location	us-central1
Type	schema	Resource URL	//35.194.61.27/company_warehouse36062
System	postgresql	Description	
Creation time			
Last modification time			
Expiration time	Never		

Tags (1)

Postgresql Schema - Metadata

Display name	Value	Type
Number of tables	5	DOUBLE

This is the real value the connector adds—it allows you to have the metadata searchable in Data Catalog.

Cleaning up

To delete the created resources, run the following command to delete the PostgreSQL metadata:

```
./cleanup-db.sh
```

Now execute the cleaner container:

```
docker run --rm --tty -v \
"$PWD":/data mesmacosta/postgresql-datacatalog-cleaner:stable \
--datacatalog-project-ids=$PROJECT_ID \
--rdbms-type=postgresql \
--table-container-type=schema
```

Finally, delete the PostgreSQL database:

```
./delete-db.sh
```

Now, from the **Navigation menu** click on **Data Catalog**. Search for **PostgreSQL**. You will no longer see the PostgreSQL Tag Templates in the results:



Ensure you see the following output in Cloud Shell before you move on:

```
Cloud SQL Instance deleted
COMPLETED
```

You will now learn how to do the same thing with a MySQL instance.

MySQL to Data Catalog

Create the MySQL Database

You can use the open source scripts at [Cloud SQL MySQL Tooling](https://github.com/mesmacosta/cloudsql-mysql-tooling) to create and populate your MySQL instance.

Run the following command in Cloud Shell to return to your home directory:

```
cd
```

Run the following command to clone the GitHub repository:

```
git clone https://github.com/mesmacosta/cloudsql-mysql-tooling
```

Now change your current working directory to the cloned repo directory:

```
cd cloudsql-mysql-tooling
```

Next execute the `init-db.sh` script:

```
source init-db.sh
```

This will create your MySQL instance and populate it with random schema. After a few minutes, you should receive the following output:

```
CREATE TABLE factory_warehouse14342.persons88a5ebc4 ( address9634 TEXT, cpf12934 FLOAT, food88799 BOOL, food4761 LONGTEXT, credit card44049 FLOAT, city8417 TINYINT, name76076 DATETIME, address19458 TIME, reason49953 DATETIME )  
COMPLETED
```

Set Up the Service Account

Run the following to create a Service Account:

```
gcloud iam service-accounts create mysql2dc-credentials \  
--display-name "Service Account for MySQL to Data Catalog connector" \  
--project $PROJECT_ID
```

Next, create and download the Service Account Key:

```
gcloud iam service-accounts keys create "mysql2dc-credentials.json" \  
--iam-account "mysql2dc-credentials@$PROJECT_ID.iam.gserviceaccount.com"
```

Next add Data Catalog admin role to the Service Account:

```
gcloud projects add-iam-policy-binding $PROJECT_ID \  
--member "serviceAccount:mysql2dc-credentials@$PROJECT_ID.iam.gserviceaccount.com" \  
--quiet \  
--project $PROJECT_ID \  

```



```
--role "roles/datacatalog.admin"
```

Execute MySQL to Data Catalog connector

You can build the MySQL connector yourself by going to [this GitHub repository](#). To facilitate its usage, this lab uses a docker image. The environment variables were loaded by the `init-db.sh` script.

Execute the connector:

```
docker run --rm --tty -v \  
"$PWD":/data mesmacosta/mysql2datacatalog:stable \  
--datacatalog-project-id=$PROJECT_ID \  
--datacatalog-location-id=us-central1 \  
--mysql-host=$public_ip_address \  
--mysql-user=$username \  
--mysql-pass=$password \  
--mysql-database=$database
```




Soon after you should receive the following output:

```
=====End mysql-to-datacatalog=====
```

Check the results of the script

Ensure that you are in the Data Catalog home page.

In the search bar, type **MySQL**. You should see the following Tag Templates and Entry Group appear in the results:

 Data Catalog  Search 

Sort by
Relevance

Systems

Data types

☐ Include public datasets

Name	Description	Type	System	Project	Last modified
MySQL Table - Metadata	-	Tag template	Data Catalog	qwiklabs-gcp-00-bdb724eb1316	May 27, 2020
MySQL Database - Metadata	-	Tag template	Data Catalog	qwiklabs-gcp-00-bdb724eb1316	May 27, 2020
mysql	-	Entry Group	Data Catalog	qwiklabs-gcp-00-bdb724eb1316	

Now click on the `mysql` Entry Group. Your console should resemble the following:

mysql

ENTRIES

DETAILS


 CREATE  EDIT  DELETE

<input type="checkbox"/>	Name
<input type="checkbox"/>	company_warehouse23772
<input type="checkbox"/>	companies31286a71
<input type="checkbox"/>	personal_info478e9f63
<input type="checkbox"/>	personal_info6f9999e2
<input type="checkbox"/>	personsefe79844
<input type="checkbox"/>	store58e791bb
<input type="checkbox"/>	company_warehouse34632
<input type="checkbox"/>	companiesfcd07fb0
<input type="checkbox"/>	homeab4e7189
<input type="checkbox"/>	personal_info41b8fb12

This is the real value of an Entry Group — you can see all entries that belong to MySQL using the UI.

Click on one of the `warehouse` entries. Look at the Custom entry details and tags:

Custom entry details and Tags:

 Data Catalog

[company_warehouse3065](#) [+ ATTACH TAGS](#)

qwiklabs-gcp-04-e4ece18e59b6 > us-central1 > mysql

Filter tags and schema

Custom entry details

Tags

Custom entry details

Display name	company_warehouse3065	Location	us-central1
Type	database	Resource URL	//35.188.10.123//company_warehouse3065
System	mysql	Description	
Creation time			
Last modification time			
Expiration time	Never		

Tags (1)

Mysql Database - Metadata

Display name	Value	Type
Number of tables	5	DOUBLE

This is the real value the connector adds — it allows you to have the metadata searchable in Data Catalog.

Cleaning up

To delete the created resources, run the following command to delete the MySQL metadata:

```
./cleanup-db.sh
```


Now execute the cleaner container:

```
docker run --rm --tty -v \
"$PWD":/data mesmacosta/mysql-datacatalog-cleaner:stable \
--datacatalog-project-ids=$PROJECT_ID \
--rdbms-type=mysql \
--table-container-type=database
```

Finally, delete the PostgreSQL database:

```
./delete-db.sh
```

From the **Navigation menu** click **Data Catalog**. Search for **MySQL**. You will no longer see the MySQL Tag Templates in the results:

 Data Catalog

[←](#) Search [+ CREATE](#) ▼

Sort by
Relevance ▼

Systems ▼

Data types ▼

☐ Include public datasets

Ensure you see the following output in Cloud Shell before you move on:

```
Cloud SQL Instance deleted
COMPLETED
```

Congratulations!

Great job! You received hands-on practice with Data Catalog connectors.

In this lab, you learned how to:

- Enable the Data Catalog API.
- Create a dataset.
- Copy a public New York Taxi table to your dataset.
- Create a tag template and attach the tag to your table.



Finish Your Quest

This self-paced lab is part of the Qwiklabs [BigQuery for Data Warehousing](#), [BigQuery for Marketing Analysts](#), and [Data Catalog Fundamentals](#) Quests. A Quest is a series of related labs that form a learning path. Completing a Quest earns you a badge to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in a Quest and get immediate completion credit if you've taken this lab. See other available [Qwiklabs Quests](#).

Next Steps / Learn More

- [Read the Data Catalog Overview](#)
- [Learn How to search with Data Catalog](#)
- [Browse the Overview of APIs and Client Libraries](#)

End your lab

When you have completed your lab, click **End Lab**. Qwiklabs removes the resources you've used and cleans the account for you.

You will be given an opportunity to rate the lab experience. Select the applicable number of stars, type a comment, and then click **Submit**.

The number of stars indicates the following:

- 1 star = Very dissatisfied
- 2 stars = Dissatisfied
- 3 stars = Neutral
- 4 stars = Satisfied
- 5 stars = Very satisfied

You can close the dialog box if you don't want to provide feedback.

For feedback, suggestions, or corrections, please use the **Support** tab.

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. [Our classes](#) include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. [Certifications](#) help you validate and prove your skill and expertise in Google Cloud technologies.

Manual Last Updated February 02, 2021

Lab Last Tested February 02, 2021

Copyright 2021 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.