06b_customer_churn_console_execution.md

# Customer Churn using Serverless Spark through Google Cloud Console

**Goal** - Data Preparation and Model Training for Detecting Customer Churn.

Following are the lab modules:

## 1. Understanding Data

The dataset used for this project is customer churn data and customer test data..

The dataset contains the following features:

- Churn - Binary field which represents customers who left/were retained within the last month
- Services that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies
- Customer account information – how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges
- Demographic info about customers – gender, age range, and if they have partners and dependents

**Note:** The following features refer to these same-host connections.

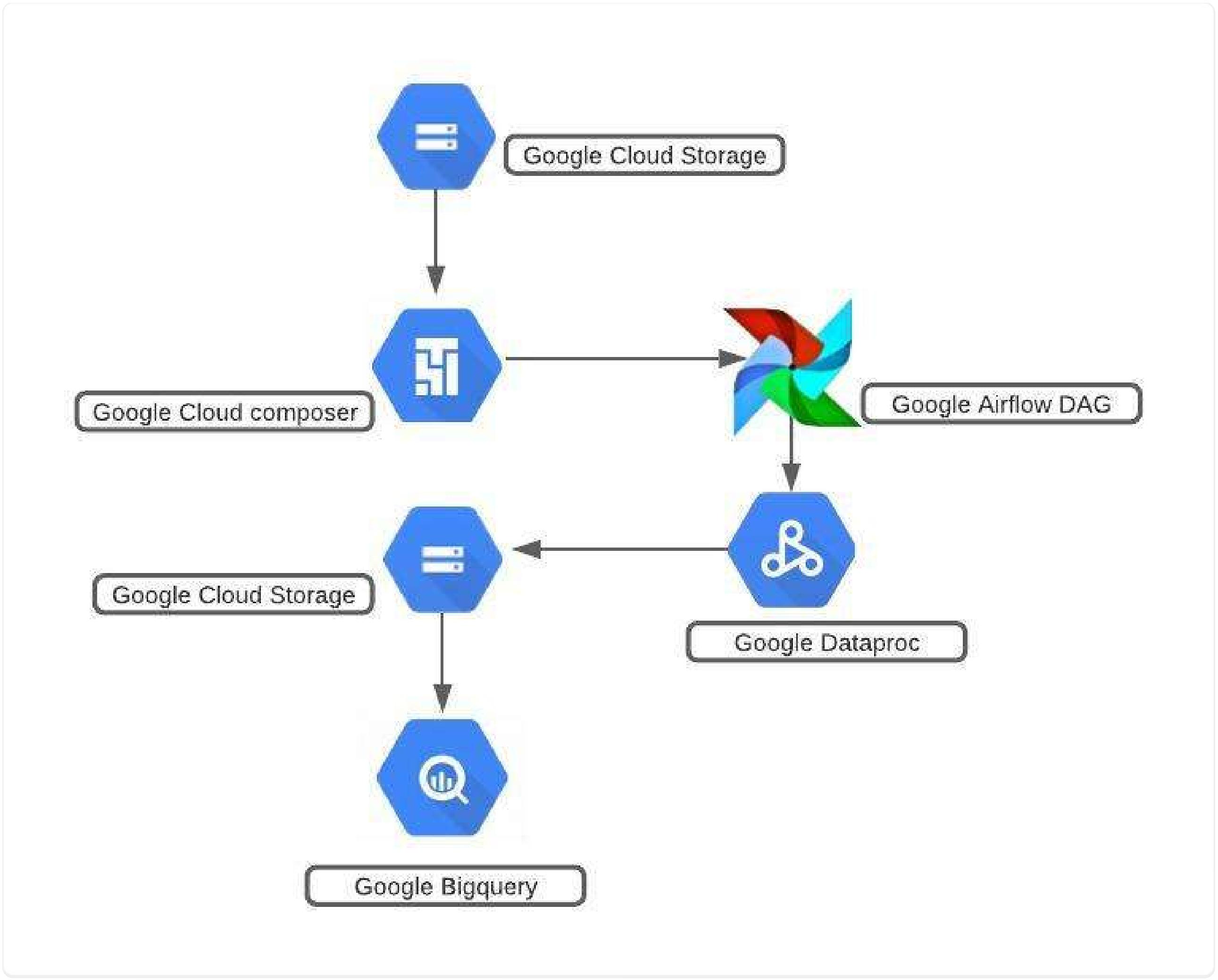
- serror_rate

- rerror_rate
- same_srv_rate
- diff_srv_rate
- srv_count

**Note:** The following features refer to these same-service connections.

- srv_serror_rate
- srv_rerror_rate
- srv_diff_host_rate

## 2. Solution Architecture



### Model Pipeline

The model pipeline involves the following steps:

- Data cleanup and preparation
- Building and training a Machine Learning Model (Random Forest Classifier) before saving it into a GCS bucket
- Using the model built in above step to evaluate test data

## 3. Parameters required for the lab

Keep the following details handy for configuring the serverless batch jobs:

```
PROJECT_ID=                                                 #current GCP project where we ar
REGION=                                                     #GCP region where all our resour
BQ_DATASET_NAME=                                            #BigQuery dataset where all the
BUCKET_CODE=                                                #GCP bucket where our code, data
HISTORY_SERVER_NAME=spark-phs                              #name of the history server which
VPC_NAME=                                                  # Primary VPC containing the subn
SUBNET=                                                     #subnet which has private google
UMSA=serverless-spark                                       #name of the user managed servic
SERVICE_ACCOUNT=$UMSA@$PROJECT_ID.iam.gserviceaccount.com
NAME=                                                        #Your unique identifier
```

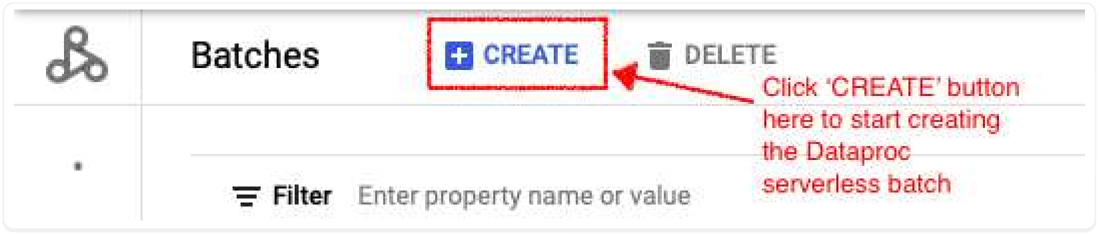
**Note:** The values for all the above parameters will be provided by the admin team.

## 4. Data Preparation

Based on EDA, the data preparation script has been created. Among the 21 columns, relevant features have been selected and stored in BQ for the next step of model training.

### 4.1. Create a new batch

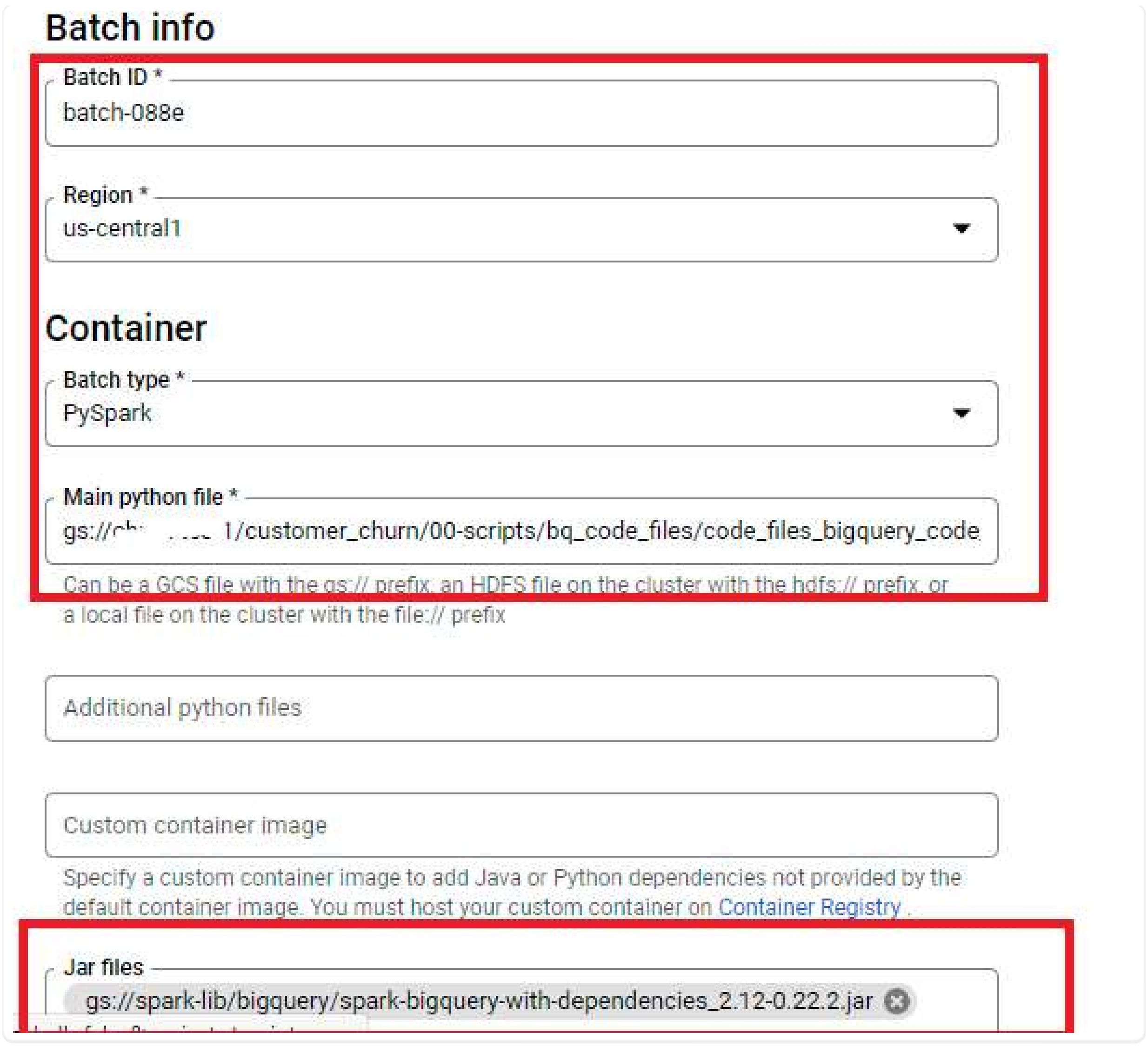
Navigate to Dataproc > Serverless > Batches and click on **+CREATE**


Batches
CREATE
DELETE
Click 'CREATE' button here to start creating the Dataproc serverless batch
Filter
Enter property name or value

### 4.2. Provide the details for the batch

Next, fill in the following values in the batch creation window as shown in the images below:

- **Batch ID** - A unique identifier for your batch
- **Region** - The region name provided by the Admin team
- **Batch Type** - PySpark
- **Main Python File** - gs://<your_code_bucket_name>/customer_churn/00-scripts/customer_churn_data_prep.py
- **JAR Files** - gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar
- **Arguments** -
  Four Arguments needs to be provided.
  - <your_project_id>
  - <your_dataset_name>
  - <your_code_bucket_name>
  - <your_name> **Note:** Press RETURN after each argument
- **Service Account** - <UMSA_NAME>@<PROJECT_ID>.iam.gserviceaccount.com
- **Network Configuration** - select the network and subnetwork with Private Google Access Enabled Run PySpark Serverless Batch for Data Preparation
- **History Server Cluster** - <your_phs_cluster_name>



## Batch info

Batch ID *
batch-088e

Region *
us-central1

## Container

Batch type *
PySpark

Main python file *
gs://[illegible]1/customer_churn/00-scripts/bq_code_files/code_files_bigquery_code

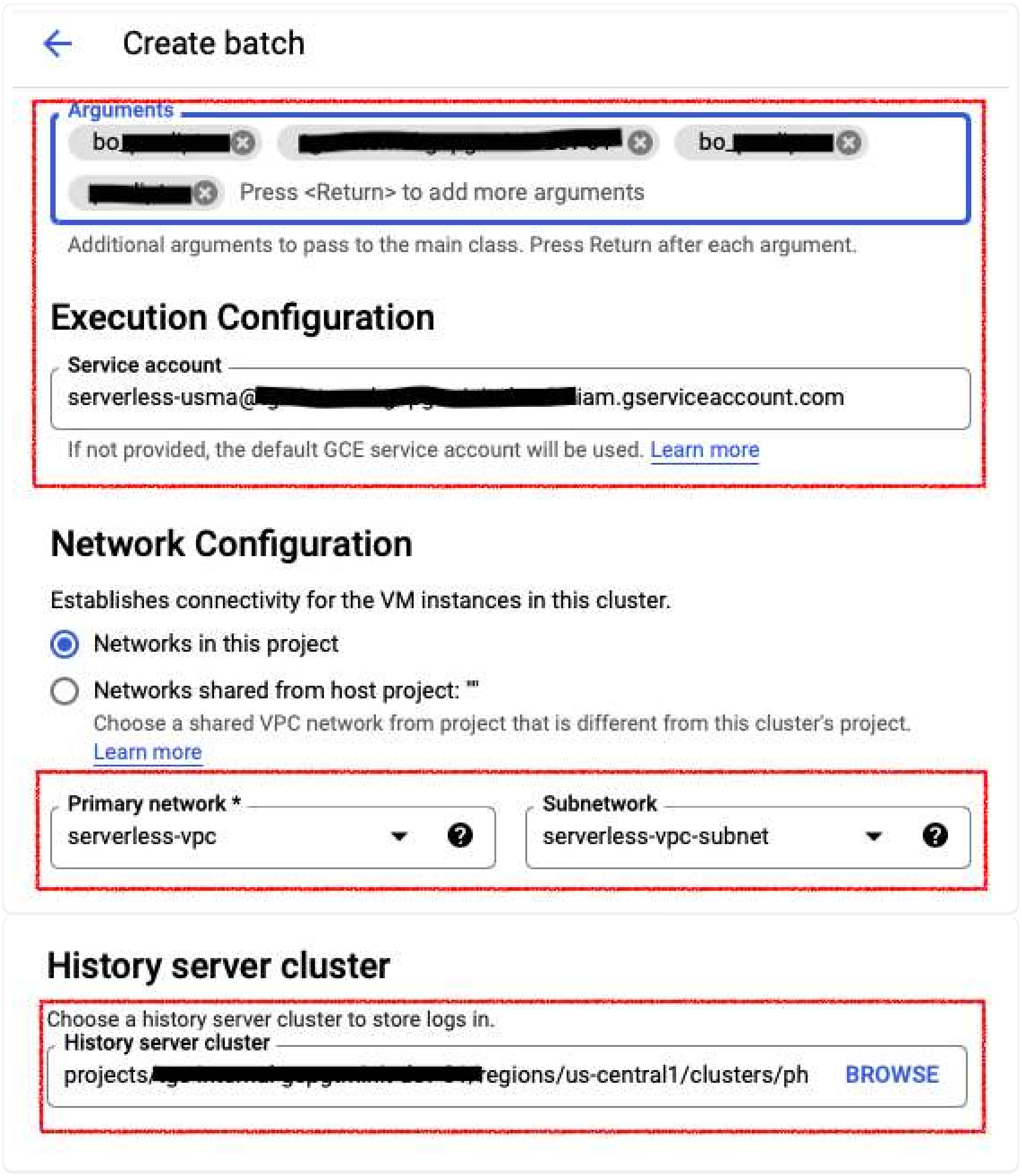
Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix

Additional python files

Custom container image

Specify a custom container image to add Java or Python dependencies not provided by the default container image. You must host your custom container on Container Registry .

Jar files
gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar


Create batch

Arguments
bo[redacted] [redacted] bo[redacted]
[redacted] Press <Return> to add more arguments
Additional arguments to pass to the main class. Press Return after each argument.

Execution Configuration

Service account
serverless-usma@[redacted]iam.gserviceaccount.com
If not provided, the default GCE service account will be used. Learn more

Network Configuration

Establishes connectivity for the VM instances in this cluster.
[✓ Networks in this project]
Networks shared from host project: ""
Choose a shared VPC network from project that is different from this cluster's project.
Learn more

Primary network *: serverless-vpc
Subnetwork: serverless-vpc-subnet

History server cluster

Choose a history server cluster to store logs in.
History server cluster
projects/[redacted]regions/us-central1/clusters/ph BROWSE

### 4.3. Submit the Serverless batch

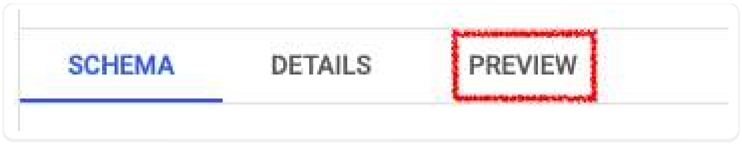
Once all the details are in, you can submit the batch. As the batch starts, you can see the execution details and logs on the console.

### 4.4. Check the output table in BQ

Navigate to BigQuery Console, and check the **customer_churn_lab** dataset.
Once the data preparation batch is completed, new tables
'<your_name_here>_training_data' and '<your_name_here>_test_data' will be created.

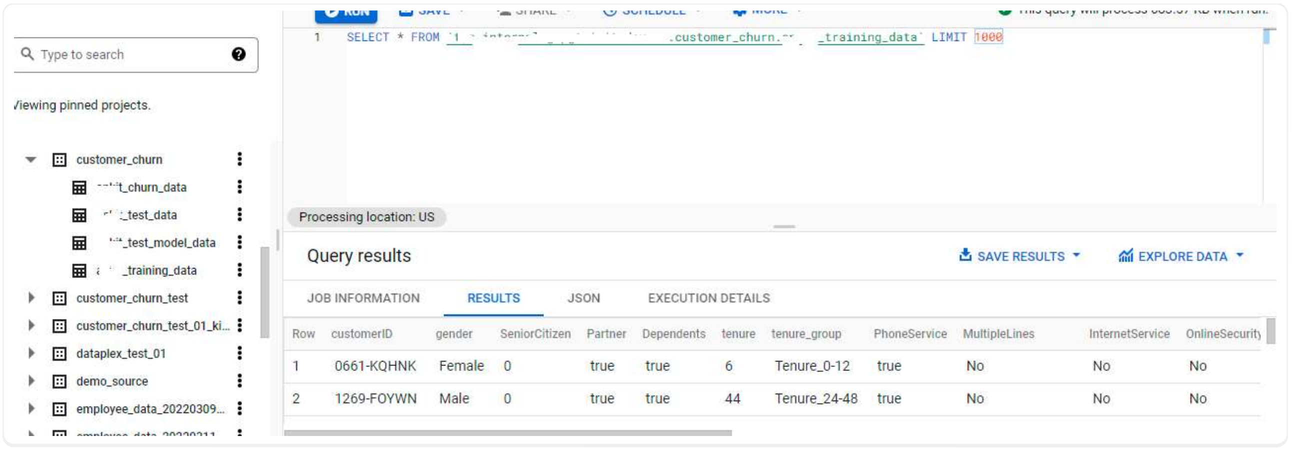
To view the data in these tables -

- Select the table from BigQuery Explorer by navigating 'project_id' > 'dataset' > 'table_name'
- Click on the **Preview** button to see the data in the table



**Note:** If the **Preview** button is not visible, run the below queries to view the data. However, these queries will be charged for the full table scan.

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_training_data` LIMIT
```



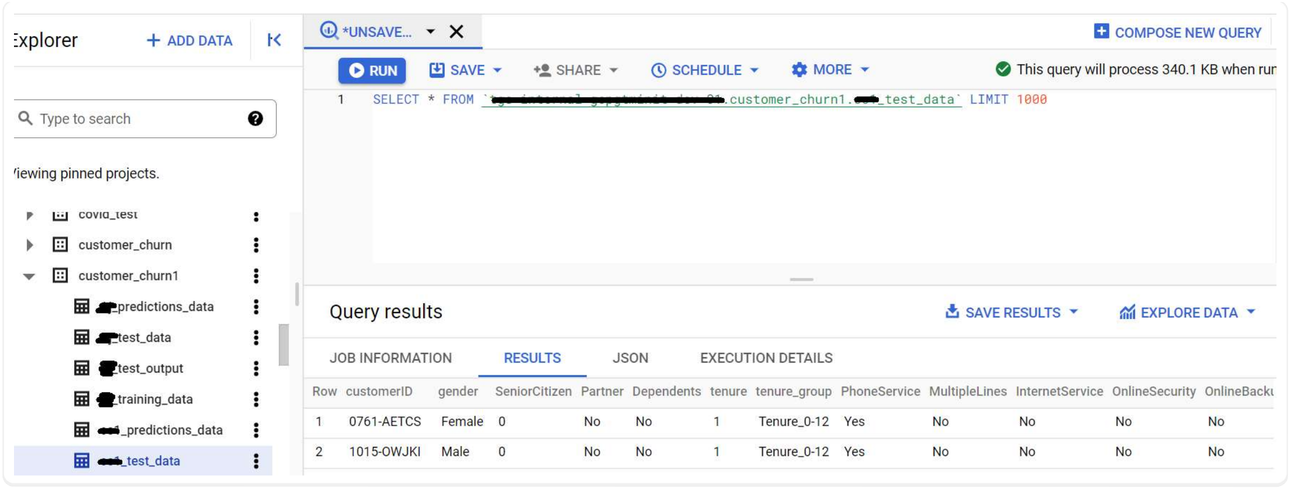
```
SELECT * FROM `[redacted].customer_churn.[redacted]_training_data` LIMIT 1000
```

Query results

| Row | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | tenure_group | PhoneService | MultipleLines | InternetService | OnlineSecurity |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0661-KQHNK | Female | 0 | true | true | 6 | Tenure_0-12 | true | No | No | No |
| 2 | 1269-FOYWN | Male | 0 | true | true | 44 | Tenure_24-48 | true | No | No | No |

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_test_data` LIMIT 10
```

<your_name_here>: _test_data table



```
SELECT * FROM `[redacted].customer_churn1.[redacted]_test_data` LIMIT 1000
```

Query results

| Row | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | tenure_group | PhoneService | MultipleLines | InternetService | OnlineSecurity | OnlineBacku |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0761-AETCS | Female | 0 | No | No | 1 | Tenure_0-12 | Yes | No | No | No | No |
| 2 | 1015-OWJKI | Male | 0 | No | No | 1 | Tenure_0-12 | Yes | No | No | No | No |

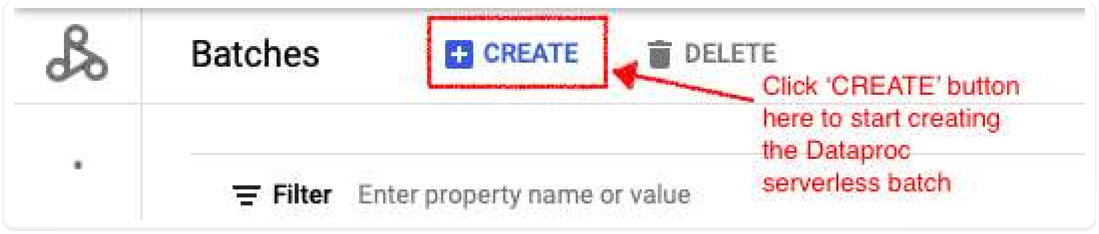
**Note:** Edit all occurrences of <project_name> and <dataset_name> to match the values of the variables PROJECT_ID, and BQ_DATASET_NAME respectively

## 5. Model Training and Testing

Repeat the same steps as above to submit another batch for model training.

### 5.1. Create a new batch

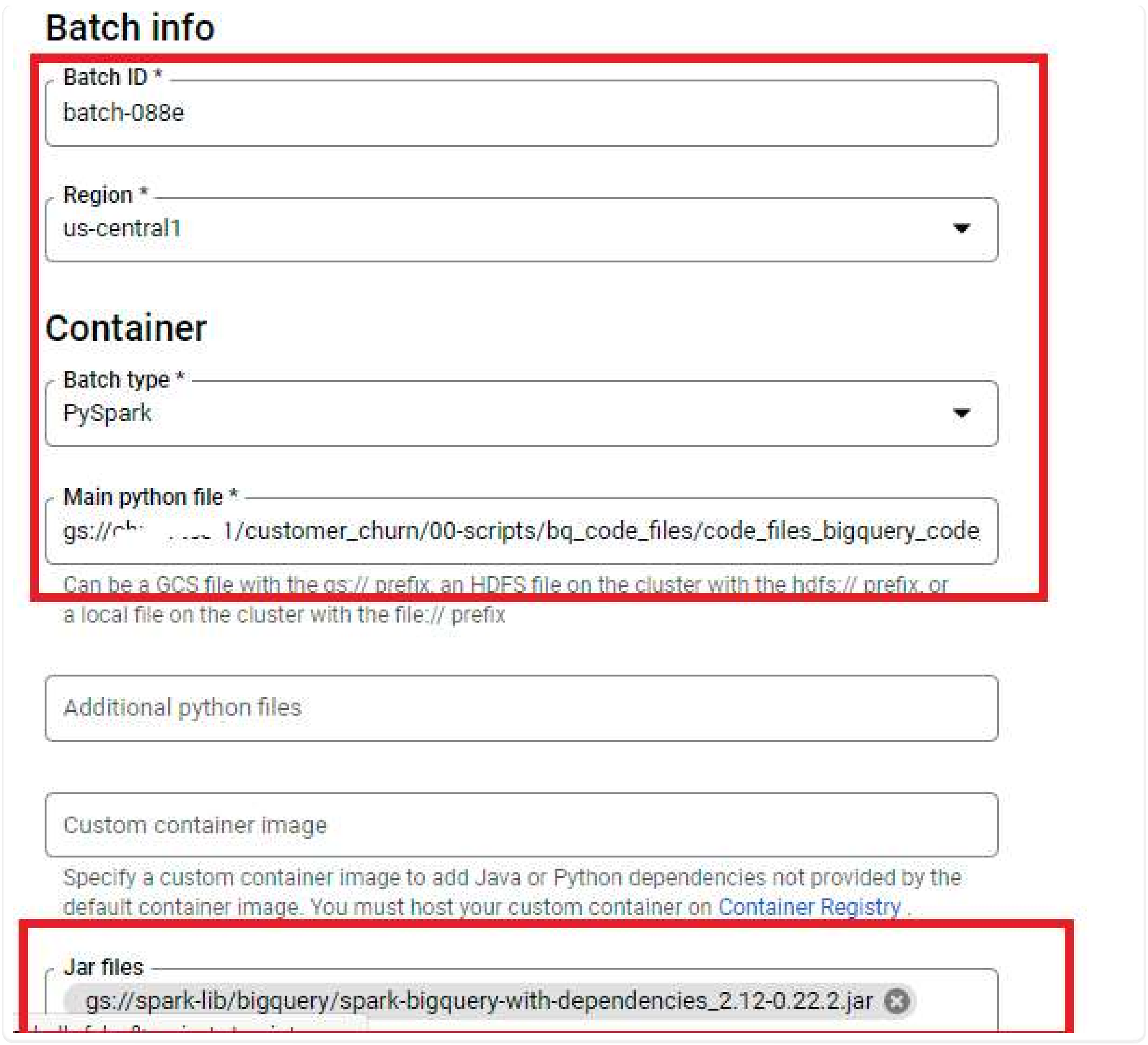
Navigate to Dataproc > Serverless > Batches and click on **+CREATE**


Batches
CREATE
DELETE
Click 'CREATE' button here to start creating the Dataproc serverless batch
Filter
Enter property name or value

### 5.2. Provide the details for the batch

Next, fill in the following values in the batch creation window as shown in the images below:

- **Batch ID** - A unique identifier for your batch
- **Region** - The region name provided by the Admin team
- **Batch Type** - PySpark
- **Main Python File** - gs://<your_code_bucket_name>/customer_churn/00-scripts/customer_churn_model_building.py
- **JAR Files** - gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar
- **Arguments** -
  Four Arguments needs to be provided.
  - <your_project_id>
  - <your_dataset_name>
  - <your_code_bucket_name>
  - <your_name> **Note:** Press RETURN after each argument
- **Service Account** - <UMSA_NAME>@<PROJECT_ID>.iam.gserviceaccount.com
- **Network Configuration** - select the network and subnetwork with Private Google Access Enabled Run PySpark Serverless Batch for Data Preparation
- **History Server Cluster** - <your_phs_cluster_name>


Batch info

Batch ID *
batch-088e

Region *
us-central1

Container

Batch type *
PySpark

Main python file *
gs://[redacted]1/customer_churn/00-scripts/bq_code_files/code_files_bigquery_code

Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix

Additional python files

Custom container image

Specify a custom container image to add Java or Python dependencies not provided by the default container image. You must host your custom container on Container Registry .

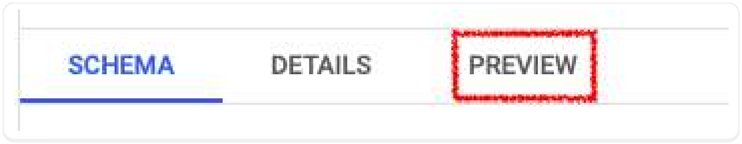
Jar files
gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar

## 5.3. Query the model_test results BQ table

Navigate to BigQuery Console, and check the **customer_churn_lab** dataset.
Once the modeling batch is completed, a new table '<your_name_here>_predictions_data' will be created.

To view the data in this table -

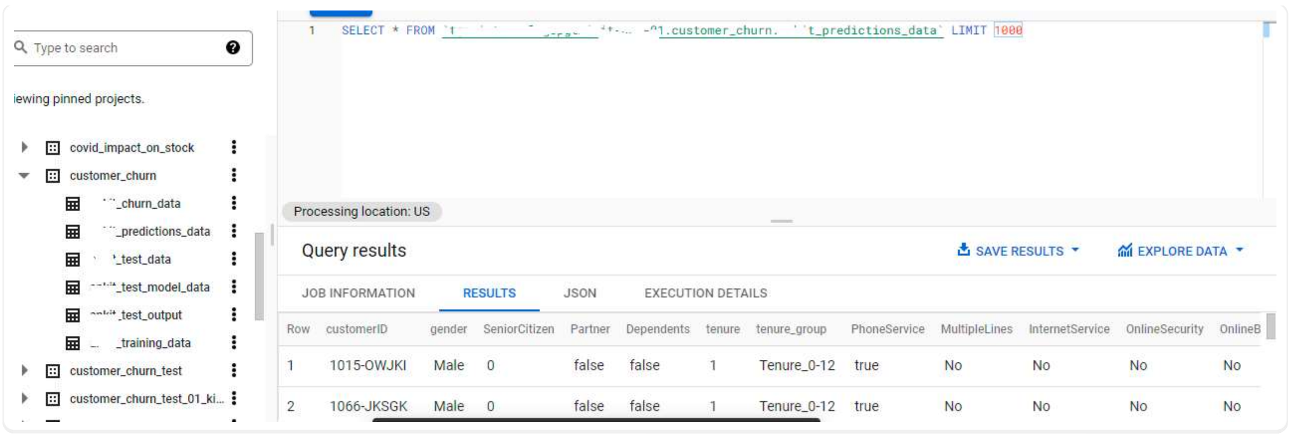
- Select the table from BigQuery Explorer by navigating 'project_id' > 'dataset' > 'table_name'
- Click on the **Preview** button to see the data in the table



**Note:** If the **Preview** button is not visible, run the below queries to view the data. However, these queries will be charged for the full table scan.

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_predictions_data` LI
```

**Note:** Edit all occurrences of <project_name> and <dataset_name> to match the values of the variables PROJECT_ID, and BQ_DATASET_NAME respectively



```
SELECT * FROM `[illegible].customer_churn.[illegible]_predictions_data` LIMIT 1000
```

Processing location: US

Query results

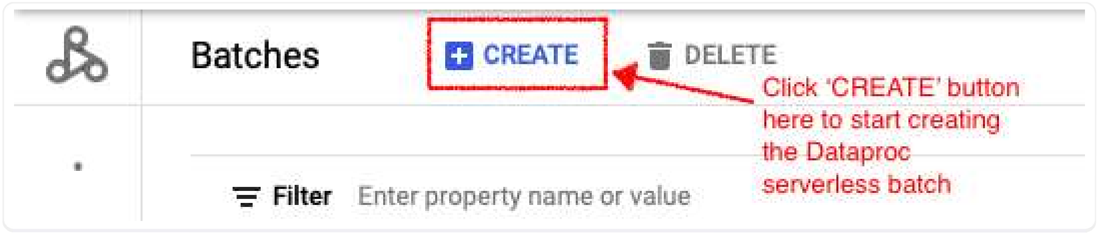
| Row | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | tenure_group | PhoneService | MultipleLines | InternetService | OnlineSecurity | OnlineB |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1015-OWJKI | Male | 0 | false | false | 1 | Tenure_0-12 | true | No | No | No | No |
| 2 | 1066-JKSGK | Male | 0 | false | false | 1 | Tenure_0-12 | true | No | No | No | No |

## 6. Model Evaluation

Repeat the same steps as above to submit another batch for model training.

### 6.1. Create a new batch

Navigate to Dataproc > Serverless > Batches and click on **+CREATE**

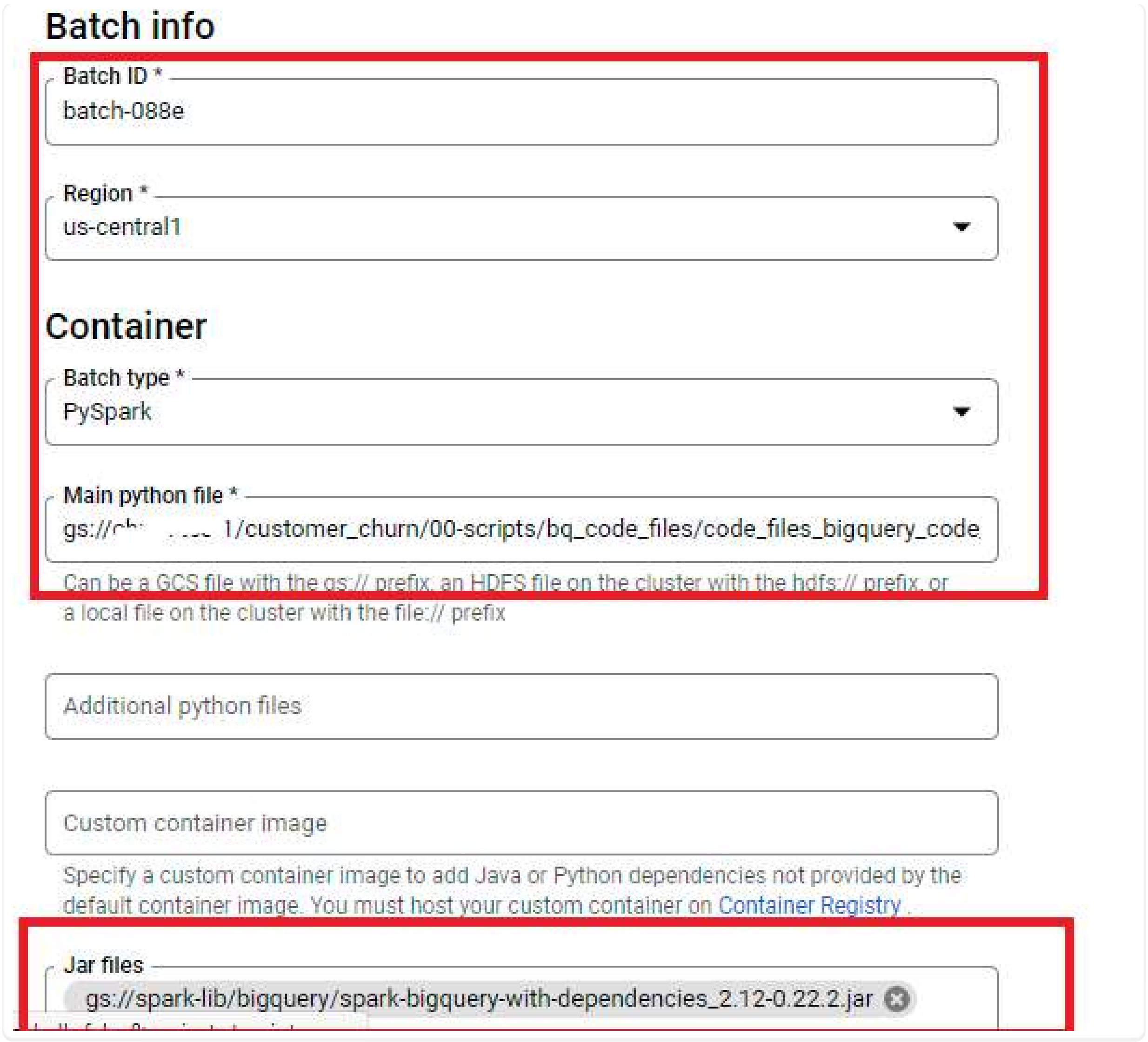

Batches
CREATE
DELETE
Click 'CREATE' button here to start creating the Dataproc serverless batch
Filter
Enter property name or value

### 6.2. Provide the details for the batch

Next, fill in the following values in the batch creation window as shown in the images below:

- **Batch ID** - A unique identifier for your batch
- **Region** - The region name provided by the Admin team

- **Batch Type** - PySpark
- **Main Python File** - gs://<your_code_bucket_name>/customer_churn/00-scripts/customer_churn_model_testing.py
- **JAR Files** - gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar
- **Arguments** -
  Four Arguments needs to be provided.
  - <your_project_id>
  - <your_dataset_name>
  - <your_code_bucket_name>
  - <your_name> **Note:** Press RETURN after each argument
- **Service Account** - <UMSA_NAME>@<PROJECT_ID>.iam.gserviceaccount.com
- **Network Configuration** - select the network and subnetwork with Private Google Access Enabled Run PySpark Serverless Batch for Data Preparation
- **History Server Cluster** - <your_phs_cluster_name>



Batch info

Batch ID *
batch-088e

Region *
us-central1

Container

Batch type *
PySpark

Main python file *
gs://ch[illegible]1/customer_churn/00-scripts/bq_code_files/code_files_bigquery_code

Can be a GCS file with the gs:// prefix, an HDFS file on the cluster with the hdfs:// prefix, or a local file on the cluster with the file:// prefix

Additional python files

Custom container image

Specify a custom container image to add Java or Python dependencies not provided by the default container image. You must host your custom container on Container Registry .

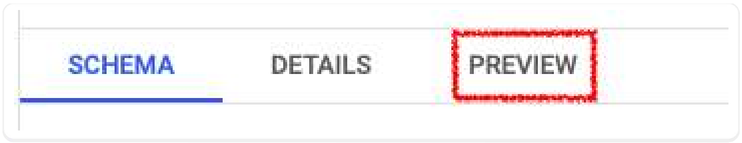
Jar files
gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar

### 6.3. Query the model_test results BQ table

Navigate to BigQuery Console, and check the **customer_churn_lab** dataset.
Once the model_testing batch is completed, a new table '<your_name_here>_test_output' will be created.

To view the data in this table -

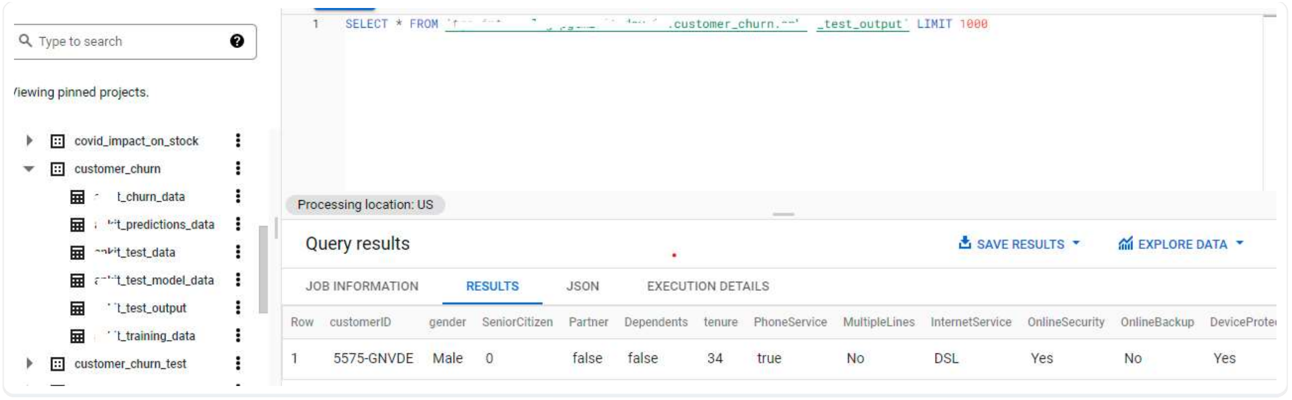
- Select the table from BigQuery Explorer by navigating 'project_id' > 'dataset' > 'table_name'
- Click on the **Preview** button to see the data in the table



**Note:** If the **Preview** button is not visible, run the below queries to view the data. However, these queries will be charged for the full table scan.

```
SELECT * FROM `<project_name>.<dataset_name>.<your_name_here>_test_output` LIMIT 1
```

**Note:** Edit all occurrences of <project_name> and <dataset_name> to match the values of the variables PROJECT_ID, and BQ_DATASET_NAME respectively



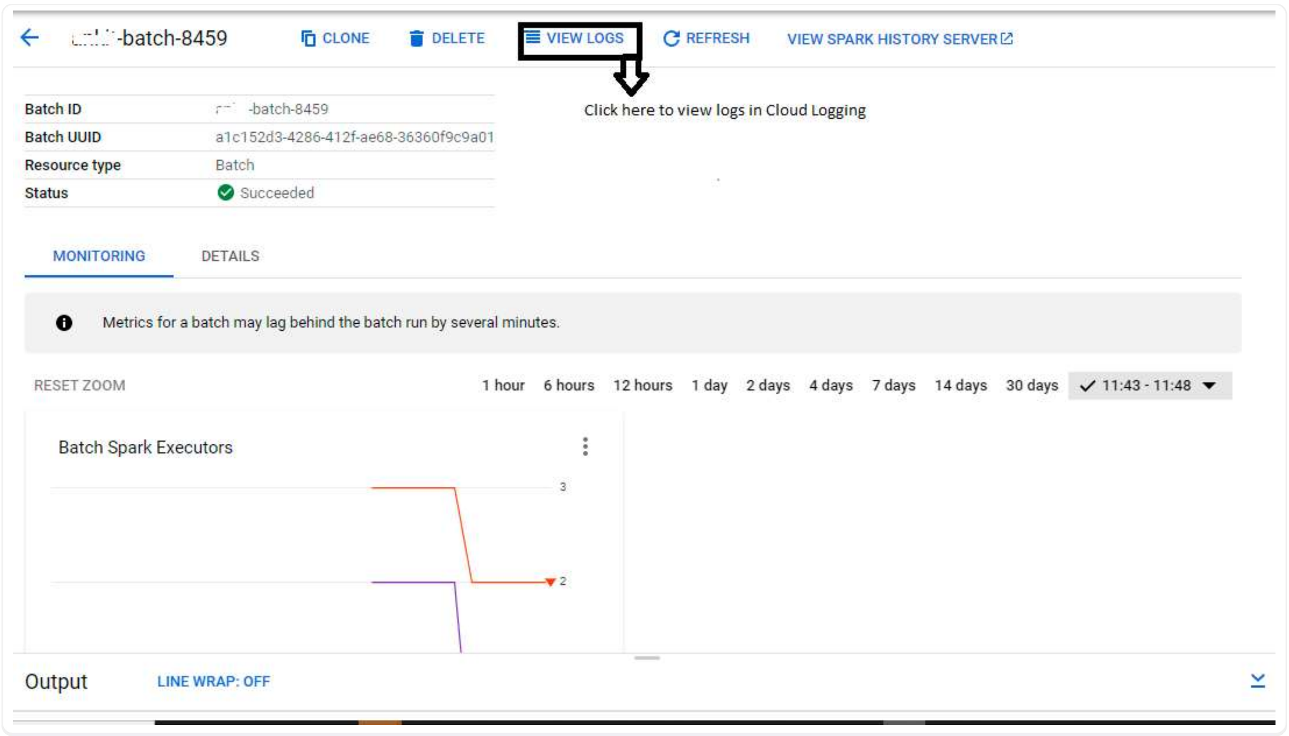
```
SELECT * FROM `[illegible].customer_churn.[illegible]_test_output` LIMIT 1000
```

| Row | customerID | gender | SeniorCitizen | Partner | Dependents | tenure | PhoneService | MultipleLines | InternetService | OnlineSecurity | OnlineBackup | DeviceProte |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 5575-GNVDE | Male | 0 | false | false | 34 | true | No | DSL | Yes | No | Yes |

# 7. Logging

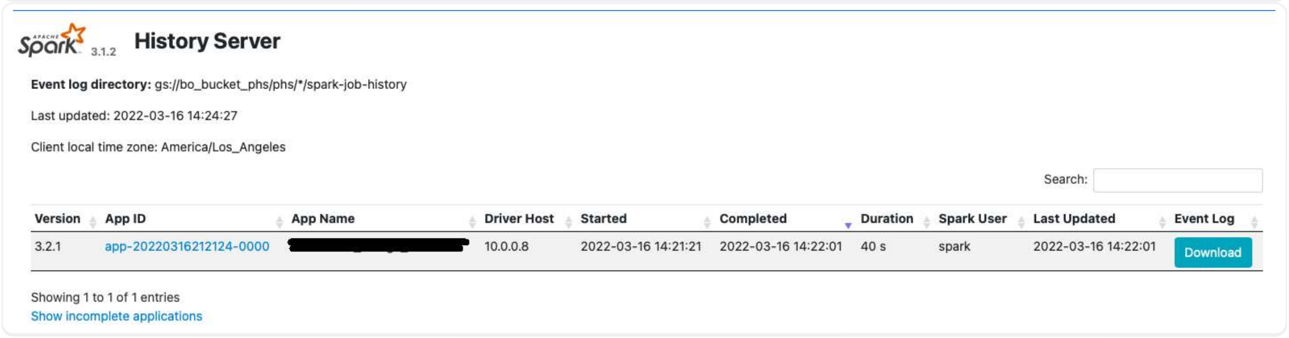
### 7.1 Serverless Batch logs

Logs associated with the application can be found in the logging console under **Dataproc > Serverless > Batches > <batch_name>**.
You can also click on “View Logs” button on the Dataproc batches monitoring page to get to the logging page for the specific Spark job.


[redacted]-batch-8459
CLONE
DELETE
VIEW LOGS
REFRESH
VIEW SPARK HISTORY SERVER
Click here to view logs in Cloud Logging

| | |
|---|---|
| Batch ID | [redacted]-batch-8459 |
| Batch UUID | a1c152d3-4286-412f-ae68-36360f9c9a01 |
| Resource type | Batch |
| Status | ✓ Succeeded |

Metrics for a batch may lag behind the batch run by several minutes.



**Event log directory:** gs://bo_bucket_phs/phs/*/spark-job-history

Last updated: 2022-03-16 14:24:27

Client local time zone: America/Los_Angeles

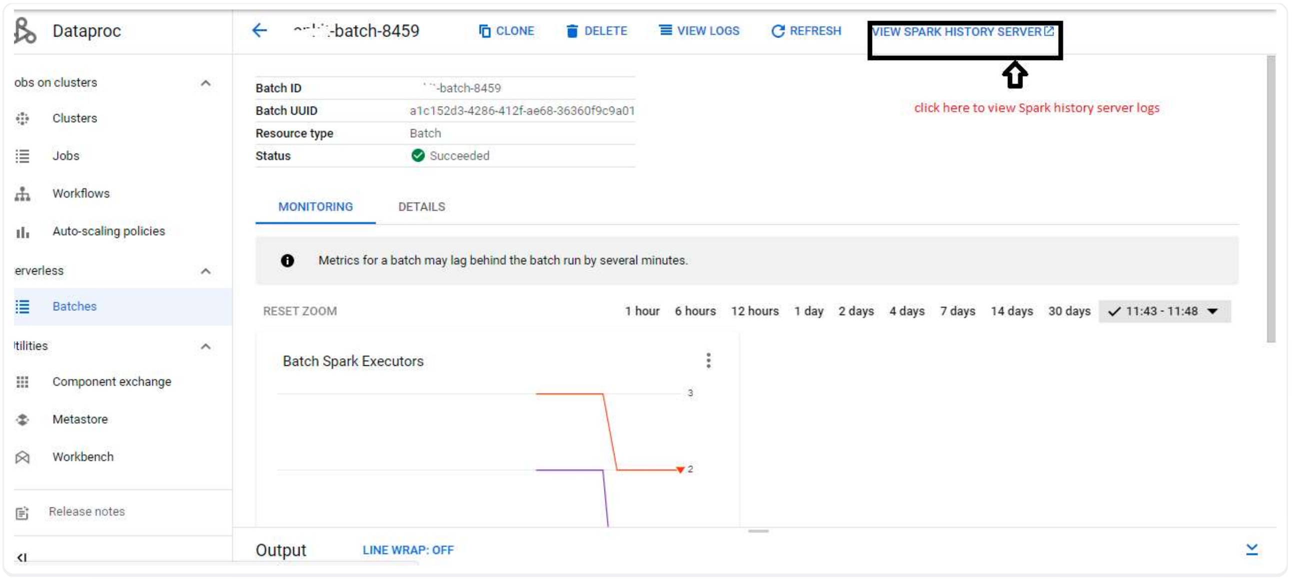
Search:

| Version | App ID | App Name | Driver Host | Started | Completed | Duration | Spark User | Last Updated | Event Log |
|---|---|---|---|---|---|---|---|---|---|
| 3.2.1 | app-20220316212124-0000 | [redacted] | 10.0.0.8 | 2022-03-16 14:21:21 | 2022-03-16 14:22:01 | 40 s | spark | 2022-03-16 14:22:01 | Download |

Showing 1 to 1 of 1 entries
Show incomplete applications

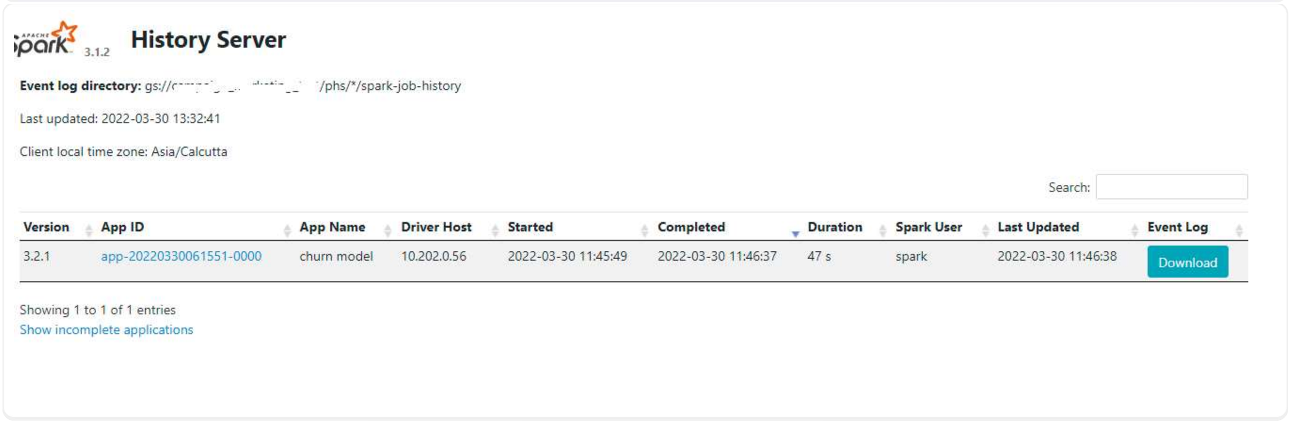
## 7.2 Persistent History Server logs

To view the Persistent History server logs, click the 'View History Server' button on the Dataproc batches monitoring page and the logs will be shown as below:



| | |
|---|---|
| Batch ID | -batch-8459 |
| Batch UUID | a1c152d3-4286-412f-ae68-36360f9c9a01 |
| Resource type | Batch |
| Status | Succeeded |

Metrics for a batch may lag behind the batch run by several minutes.



**Event log directory:** gs://[illegible]/phs/*/spark-job-history

Last updated: 2022-03-30 13:32:41

Client local time zone: Asia/Calcutta

Search:

| Version | App ID | App Name | Driver Host | Started | Completed | Duration | Spark User | Last Updated | Event Log |
|---|---|---|---|---|---|---|---|---|---|
| 3.2.1 | app-20220330061551-0000 | churn model | 10.202.0.56 | 2022-03-30 11:45:49 | 2022-03-30 11:46:37 | 47 s | spark | 2022-03-30 11:46:38 | Download |

Showing 1 to 1 of 1 entries

Show incomplete applications