06b_timeseries_forecasting_gcloud_execution.md

Timeseries Forecasting using sessions in Serverless Spark through Google Cloud Shell

Following are the lab modules:

1. Understanding Data
2. Solution Architecture
3. Declaring Variables
4. Execution
5. Logging

1. Understanding Data

Data Files

The datasets used for this project are:

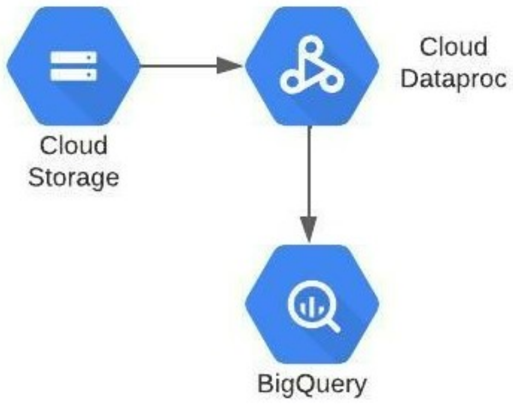
- train.csv: This file contains the date, store, item, sales data.

date - Date of the sale data. There are no holiday effects or store closures
store - Store ID
item - Item ID
sales - Number of items sold at a particular store on a particular date.

- test.csv:

id- Unique identifier
date - Date of the sale data. There are no holiday effects or store closures.
store - Store ID
item - Item ID

2. Solution Architecture



3. Declaring cloud shell variables

3.1 Set the PROJECT_ID in Cloud Shell

Open Cloud shell or navigate to shell.cloud.google.com
Run the below

```
gcloud config set project $PROJECT_ID
```

3.2 Verify the PROJECT_ID in Cloud Shell

Next, run the following command in cloud shell to ensure that the current project is set correctly:

```
gcloud config get-value project
```

3.3 Declare the variables

Based on the prereqs and checklist, declare the following variables in cloud shell by replacing with your values:

```
PROJECT_ID=$(gcloud config get-value project)       #current GCP project where we are building our use case
REGION=                                             #GCP region where all our resources will be created
SUBNET=                                             #subnet which has private google access enabled
BUCKET_CODE=                                        #GCP bucket where our code, data and model files will be stored
BUCKET_PHS=                                         #bucket where our application logs created in the history server will be stored
HISTORY_SERVER_NAME=                                #name of the history server which will store our application logs
BQ_DATASET_NAME=                                    #BigQuery dataset where all the tables will be stored
SESSION_NAME=                                       # Serverless Session name.
UMSA_NAME=                                          #user managed service account required for the PySpark job executions
SERVICE_ACCOUNT=$UMSA_NAME@$PROJECT_ID.iam.gserviceaccount.com
NAME=                                               #Your unique identifier
```

Note: For all the variables except 'NAME', please ensure to use the values provided by the admin team.

3.4 Update Cloud Shell SDK version

Run the below on cloud shell-

```
gcloud components update
```

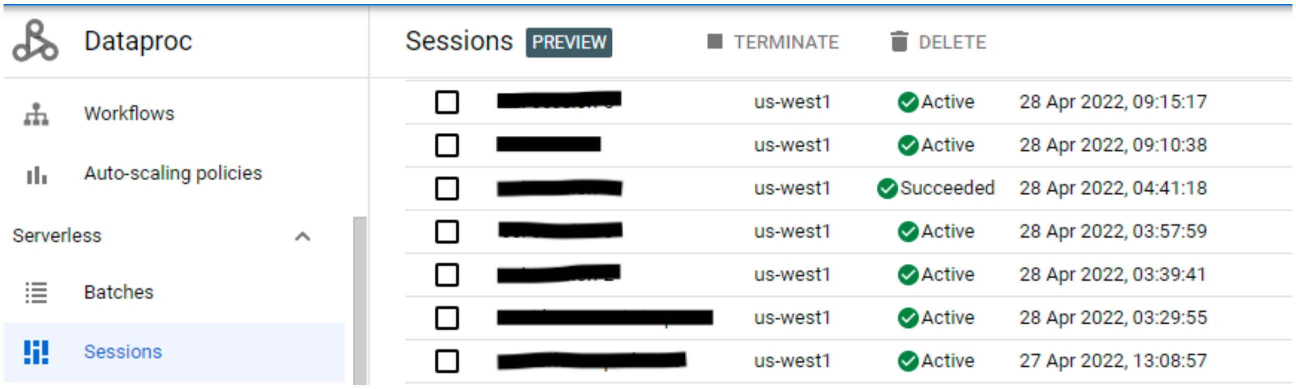
4. Execution

4.1. Run the Batch by creating sessions.

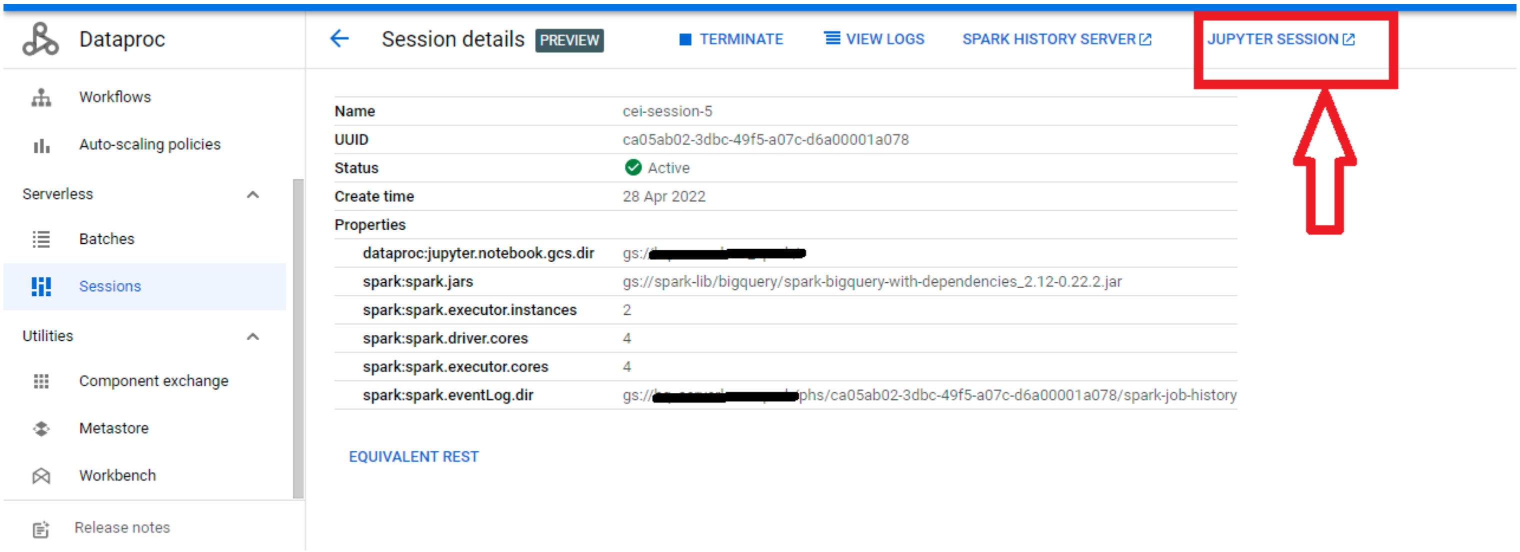
Run the below on cloud shell to create session. -

```
  gcloud beta dataproc sessions create spark $SESSION_NAME  \
--project=${PROJECT_ID} \
--location=${REGION} \
--property=spark.jars=gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar \
--history-server-cluster=projects/$PROJECT_ID/regions/$REGION/clusters/$HISTORY_SERVER_NAME \
--subnet=$SUBNET \
--property=dataproc:jupyter.notebook.gcs.dir=$BUCKET_CODE
```

- Once the serverless spark session has been created, open the session and click on the jupyter session.

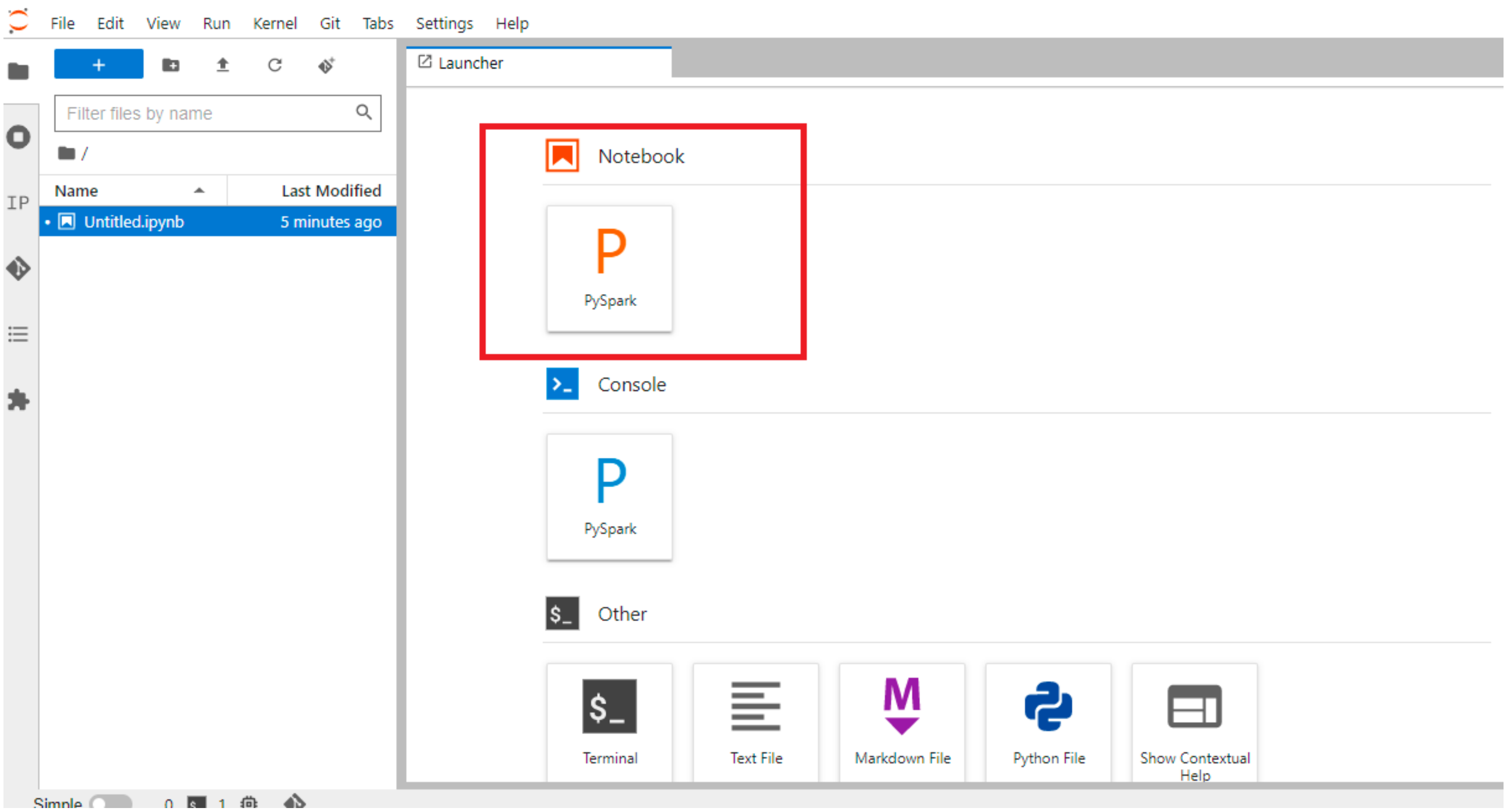


□	[redacted]	us-west1	✓ Active	28 Apr 2022, 09:15:17
□	[redacted]	us-west1	✓ Active	28 Apr 2022, 09:10:38
□	[redacted]	us-west1	✓ Succeeded	28 Apr 2022, 04:41:18
□	[redacted]	us-west1	✓ Active	28 Apr 2022, 03:57:59
□	[redacted]	us-west1	✓ Active	28 Apr 2022, 03:39:41
□	[redacted]	us-west1	✓ Active	28 Apr 2022, 03:29:55
□	[redacted]	us-west1	✓ Active	27 Apr 2022, 13:08:57

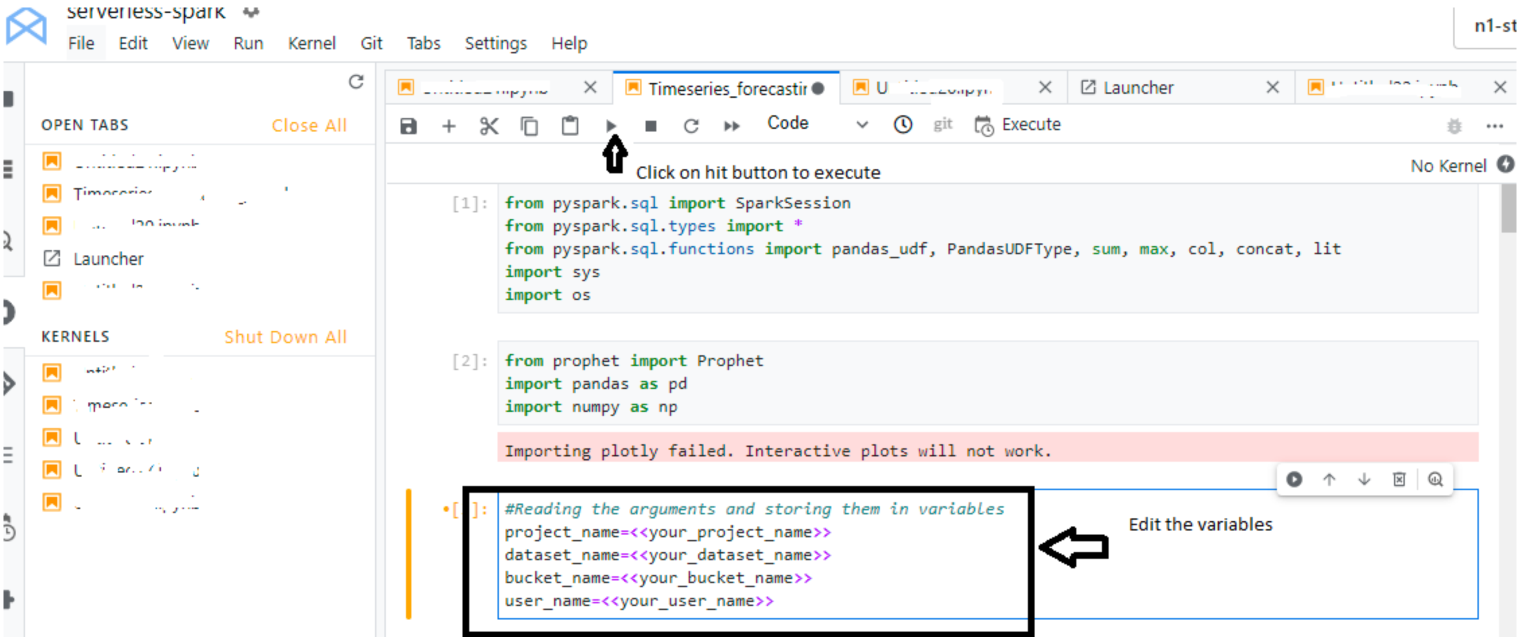


Name	cei-session-5
UUID	ca05ab02-3dbc-49f5-a07c-d6a00001a078
Status	✓ Active
Create time	28 Apr 2022
Properties	
dataproc:jupyter.notebook.gcs.dir	gs:/[redacted]
spark:spark.jars	gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar
spark:spark.executor.instances	2
spark:spark.driver.cores	4
spark:spark.executor.cores	4
spark:spark.eventLog.dir	gs://[redacted]phs/ca05ab02-3dbc-49f5-a07c-d6a00001a078/spark-job-history

- Select Pyspark Kernel for the execution.



- Copy the code from 00-scripts/timeseries_forecasting.py into the notebook created and edit the variables: project_name,dataset_name,bucket_name and name with your values and hit the **Execute** button to execute the code



Click on hit button to execute

```
[1]: from pyspark.sql import SparkSession
     from pyspark.sql.types import *
     from pyspark.sql.functions import pandas_udf, PandasUDFType, sum, max, col, concat, lit
     import sys
     import os
```

```
[2]: from prophet import Prophet
     import pandas as pd
     import numpy as np
```

```
Importing plotly failed. Interactive plots will not work.
```

```
[ ]: #Reading the arguments and storing them in variables
     project_name=<<your_project_name>>
     dataset_name=<<your_dataset_name>>
     bucket_name=<<your_bucket_name>>
     user_name=<<your_user_name>>
```

Edit the variables

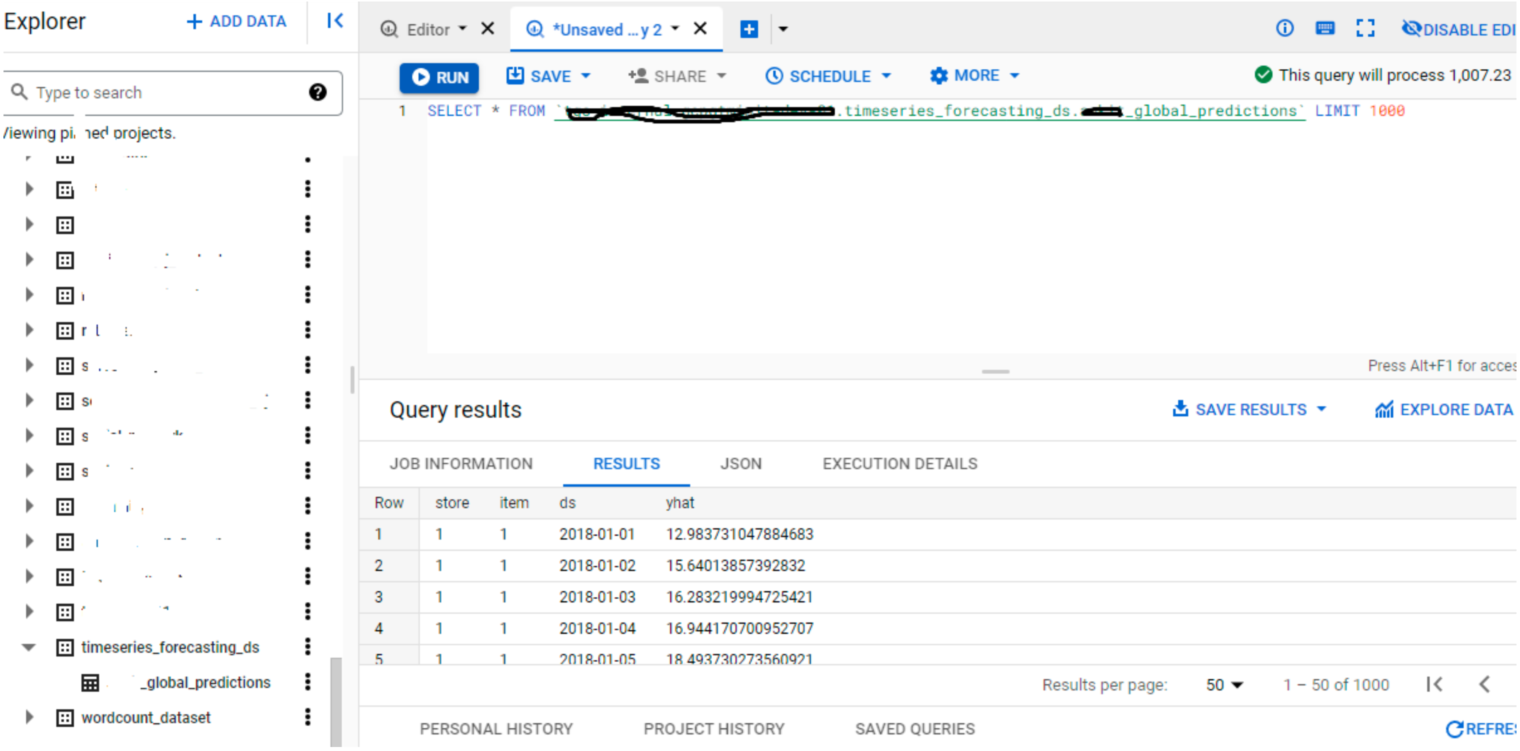
4.2. Check the output table in BigQuery

Navigate to BigQuery Console, and check the **timeseries_forecasting** dataset.
Once the code has successfully executed, four new tables '<your_name_here>_global_predictions' will be created :

To query the data to find the list of stocks with highest stringency Index, run the following query -

```
select * from `<GCP-PROJECT-NAME>.<BQ-DATASET-NAME>.<user_name>_global_predictions`
```

Note: Edit all occurrences of and to match the values of the variables PROJECT_ID,user_name and BQ_DATASET_NAME respectively



```
SELECT * FROM `[redacted].timeseries_forecasting_ds.[redacted]_global_predictions` LIMIT 1000
```

This query will process 1,007.23

Query results

Row	store	item	ds	yhat
1	1	1	2018-01-01	12.983731047884683
2	1	1	2018-01-02	15.64013857392832
3	1	1	2018-01-03	16.283219994725421
4	1	1	2018-01-04	16.944170700952707
5	1	1	2018-01-05	18.493730273560921

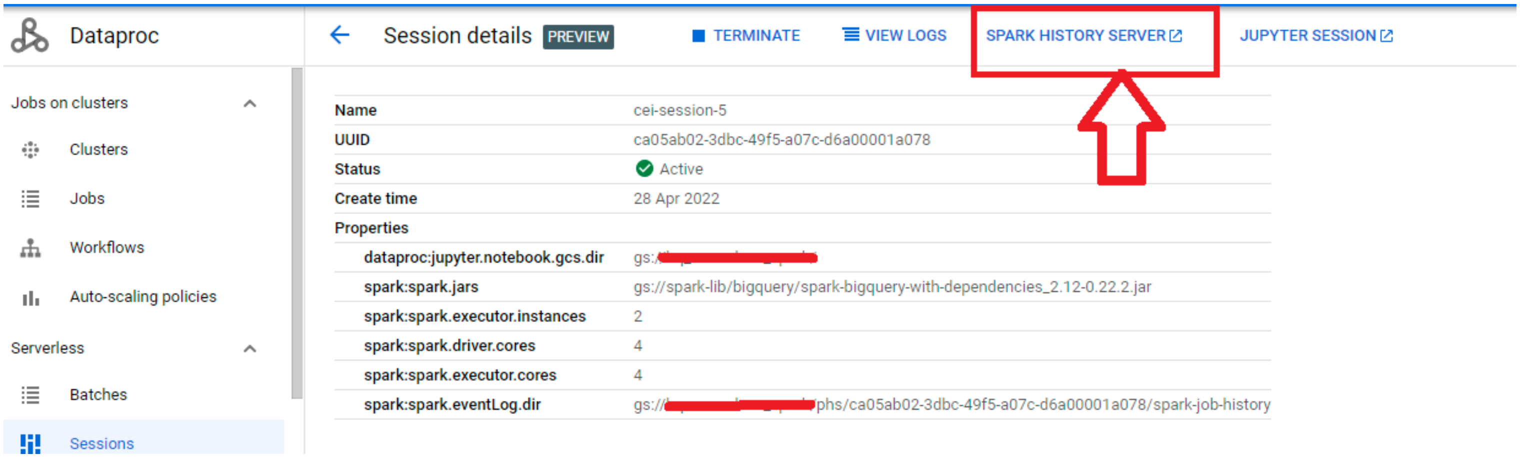
Results per page: 50 1 – 50 of 1000

5. Logging

5.1 Persistent History Server logs

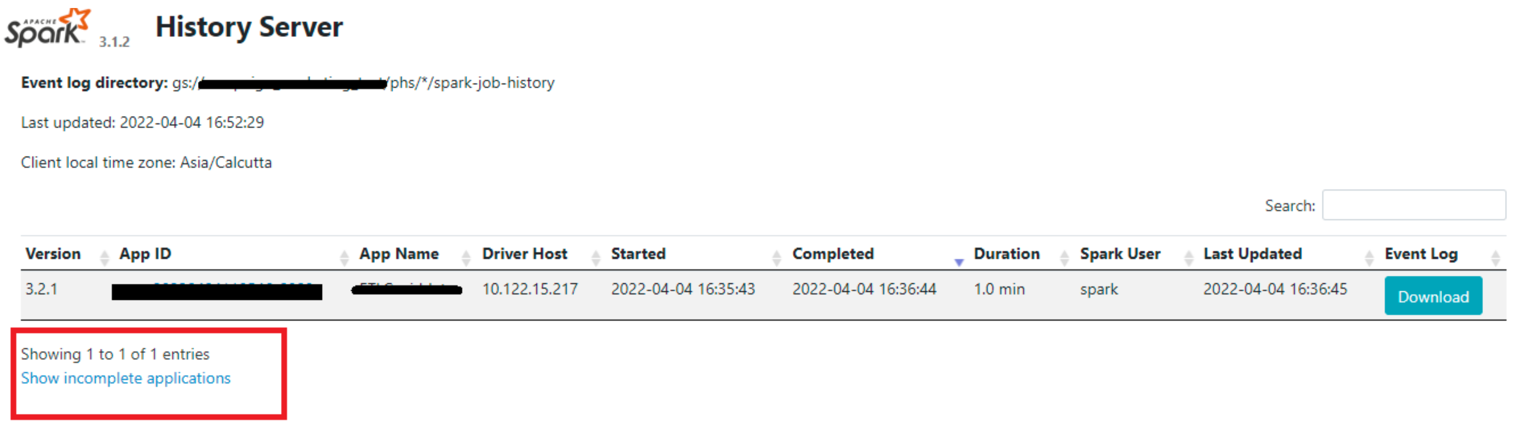
To view the Persistent History server logs, click the 'View History Server' button on the Sessions monitoring page and the logs will be shown as below:

As the session is still in active state , we will be able to find the logs in show incomplete applications.



Session details PREVIEW

Name	cei-session-5
UUID	ca05ab02-3dbc-49f5-a07c-d6a00001a078
Status	Active
Create time	28 Apr 2022
Properties	
dataproc:jupyter.notebook.gcs.dir	gs:/[redacted]
spark:spark.jars	gs://spark-lib/bigquery/spark-bigquery-with-dependencies_2.12-0.22.2.jar
spark:spark.executor.instances	2
spark:spark.driver.cores	4
spark:spark.executor.cores	4
spark:spark.eventLog.dir	gs://[redacted]phs/ca05ab02-3dbc-49f5-a07c-d6a00001a078/spark-job-history



3.1.2

History Server

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

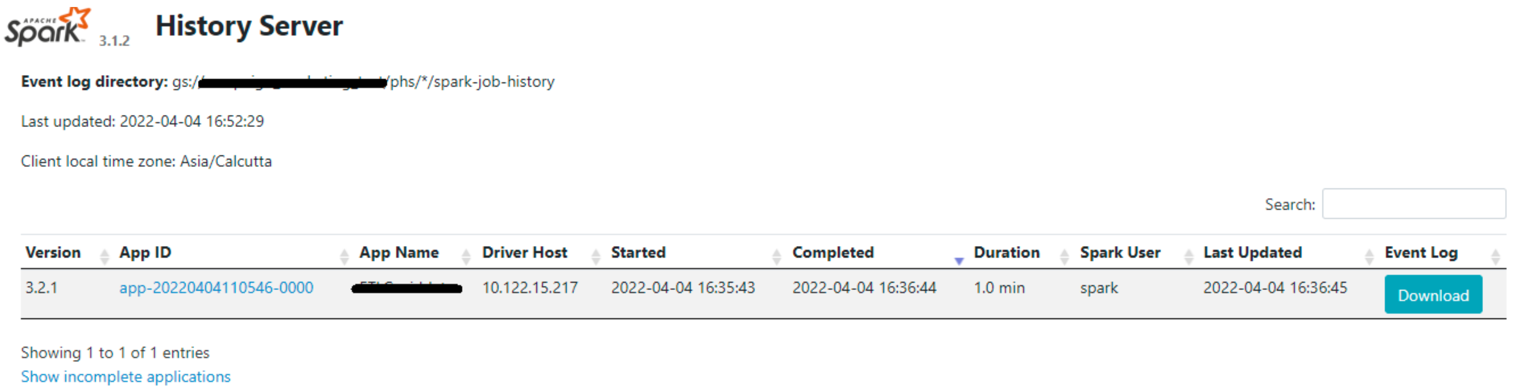
Last updated: 2022-04-04 16:52:29

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	[redacted]	[redacted]	10.122.15.217	2022-04-04 16:35:43	2022-04-04 16:36:44	1.0 min	spark	2022-04-04 16:36:45	Download

Showing 1 to 1 of 1 entries
Show incomplete applications



3.1.2

History Server

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

Last updated: 2022-04-04 16:52:29

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-20220404110546-0000	[redacted]	10.122.15.217	2022-04-04 16:35:43	2022-04-04 16:36:44	1.0 min	spark	2022-04-04 16:36:45	Download

Showing 1 to 1 of 1 entries
Show incomplete applications