ASSIGNMENT 3: PARALLEL PROCESSING

HARSHIKA AKULA (2075727)

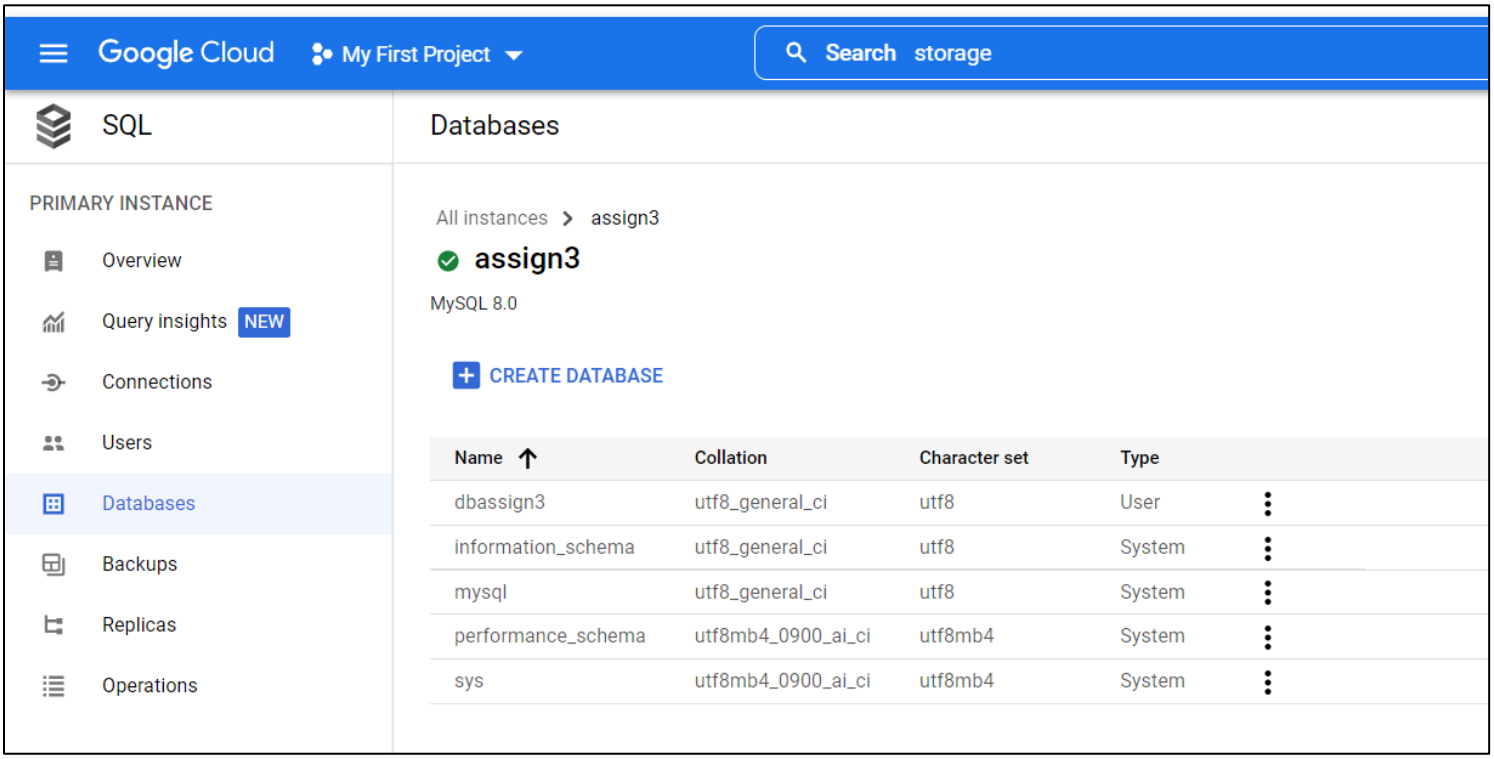
In this report, we built a pyspark application in jupyter notebook to analyze the data and executed in EMR cluster. The dataset has been imported and taken from google cloud platform. Below are the steps for the process.

1. Uploading the dataset from local system to GOOGLE CLOUD

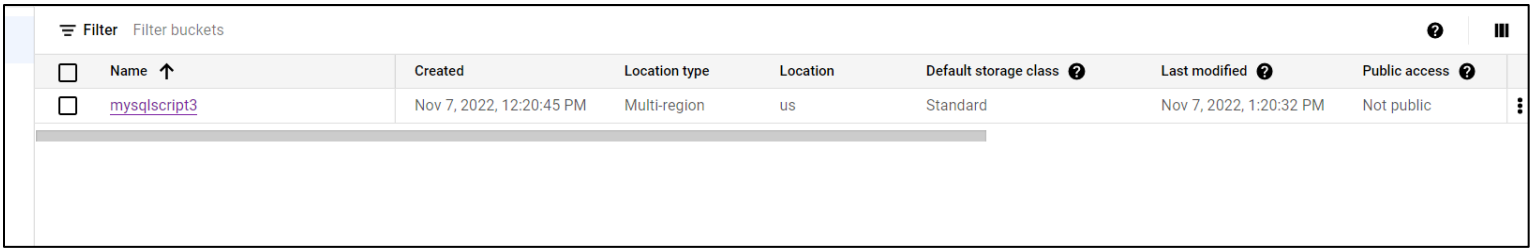
i. In this step, we are creating an MySQL instance named "assign3" by giving specific details like Rootpassword, region and database version. We should also specify the ip address of jupyter notebook that we are using inorder to connect to this sql instance. I have uploaded mysql file from local system to cloud storage into a bucket named "**mysqlscript3**"

ii. Under **assign3** sql instance, I have created a database named "**dbassign3**" into which we are importing sql file that we uploaded in the **mysqlscript3** bucket.

Below is the screenshot for **assign3** sql instance with **dbassign3** database

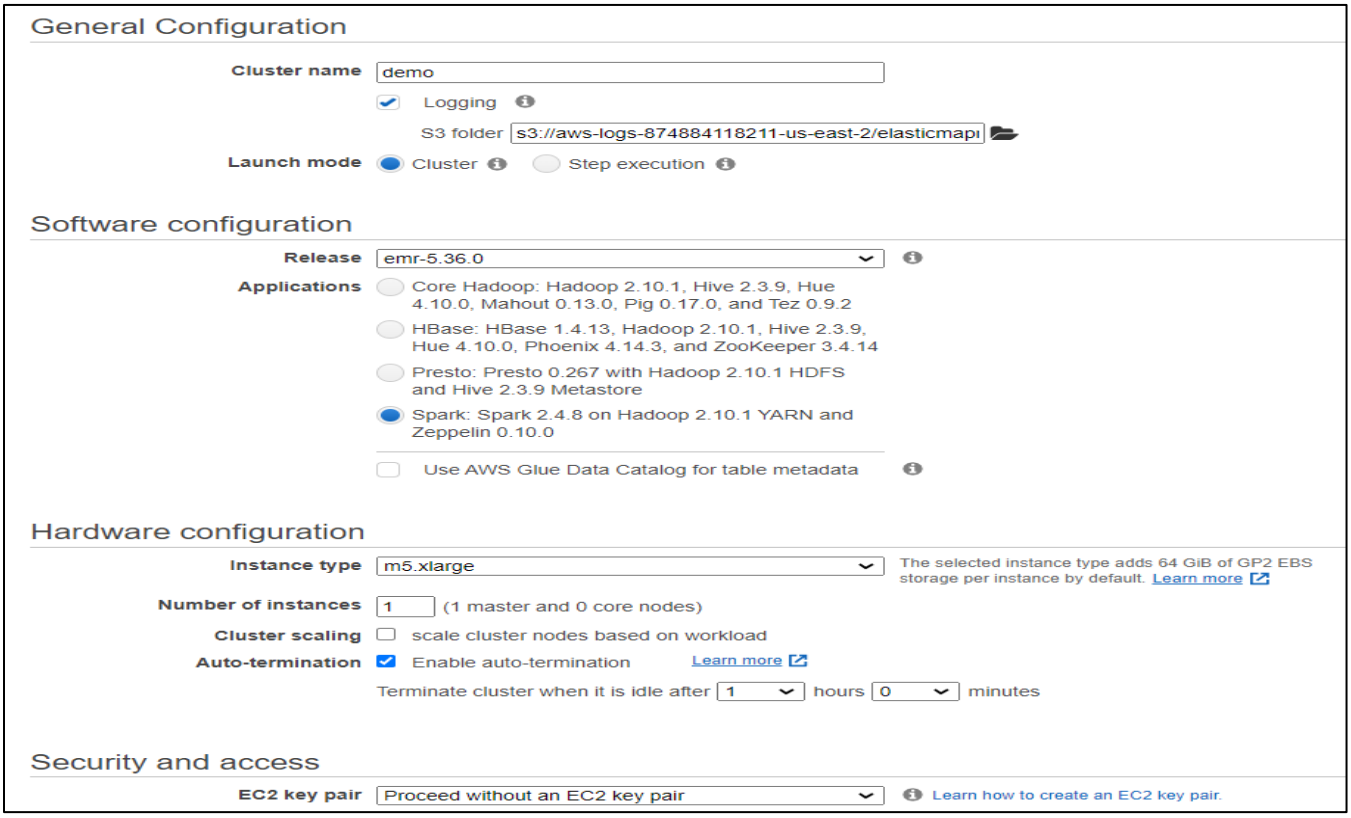


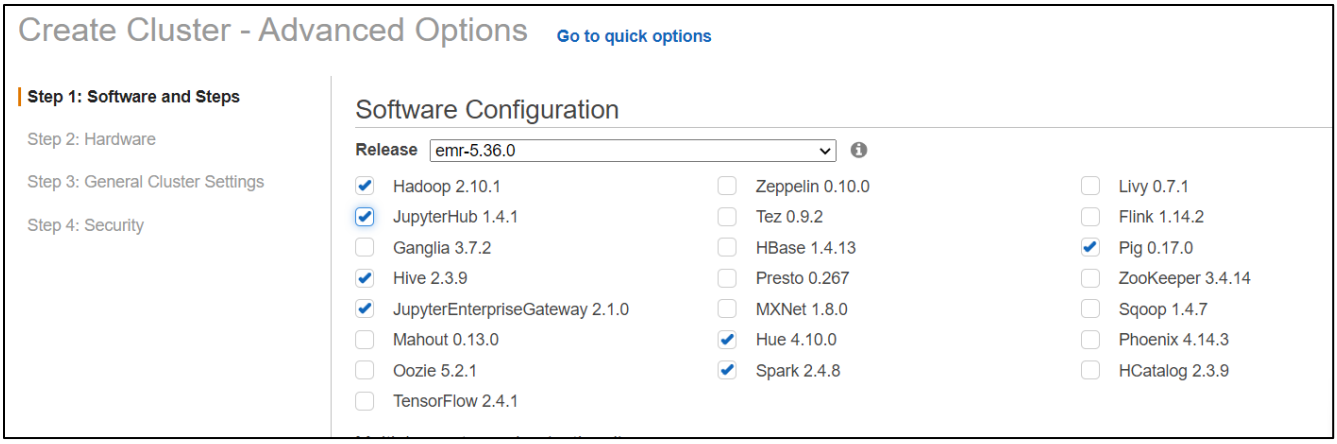
Below is the screenshot for bucket creation named "**mysqlscript3**"



2. Creation an EMR cluster in AWS

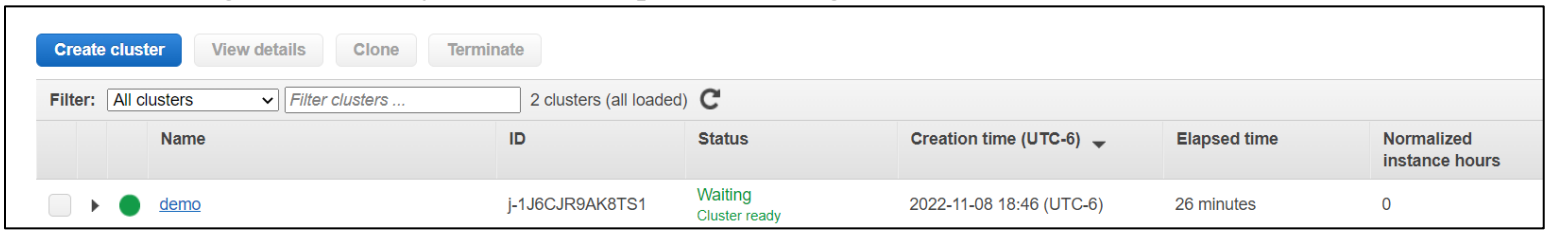
i. In this step,we log in to aws console and go to EMR service. We create a cluster named "**demo**" and specific details like emr version(emr-5.36.0) , Applications(Spark), No EC2 key-pair.Under advanced options, we also make changes in software configuration to run spark application in jupyter notebook.



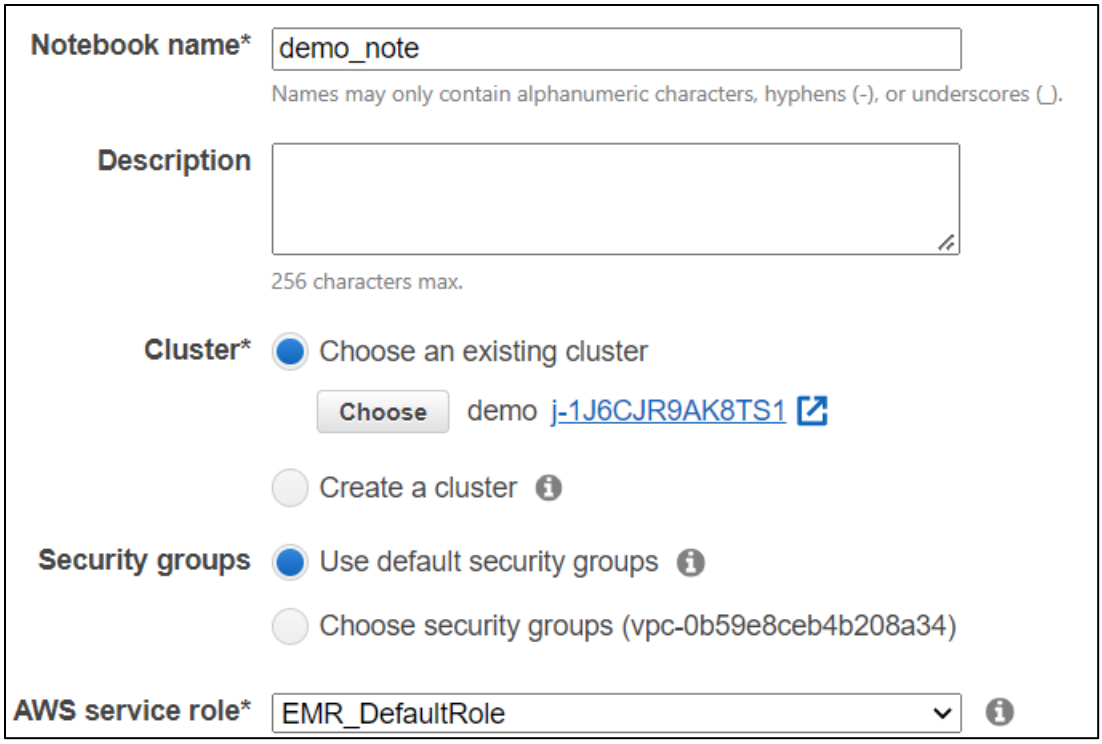


ii. Under IAM Console,We also need to attach two policies for EMR_DefaultRole, those policies are "**AmazonElasticMapReduceEditorsRole**" and "**AmazonS3FullAccess**".

iii. After clicking on create button,we can monitor the status of cluster creation.If the status says "waiting cluster ready", cluster is up and running.



iv. After creation of cluster, we create a notebook named "**demo_note**". since we have an active cluster "**demo**" running, we choose the existing cluster to attach to the notebook.choose the default service IAM role and click on create notebook.If the status of the notebook says ready then we can launch the Jupyter.



3. Connecting the GCP cloud SQL in Jupyter using Pyspark

After launching the jupyter notebook, we select the pyspark kernel amongst various jupyter kernels specified. We need to connect to database of GCP instance by installing "sql-connector-python" package and importing sql.connector. Then we use connect function in which we specify the username, password, public ip address of **assign3** instance and database name("**dbassign3**). Then we connect with the cursor and write the query.We also instantiate the spark using SparkSession and create object for both spark context and spark session.After that we retrieve all the tables with proper columns and data from gcp instance and convert it to spark dataframe and start ETL operations on those dataframes according to the problem statement given.