1. Write a spark code for executing the Hash example provided in slide 13 on Hashing from Lab 1 Presentation, on the public file*: 'gs://bdl-bucket-1/lab4_dataset.csv'.* You would have to find the number of user clicks between 0-6, 6-12, 12-18, and 18-24, as was discussed in the first class.

a. Submit the python file with your code.

b. Also, provide the text file containing your output.

Answer: The following screenshots contain the python code to implement the given task:

```
import pyspark
import sys

def return_group(x):
    sen = x.split(",")          #Takes the entire line in the form of date,time,userid and splits using ','
    t = sen[1]                   #extracts the value of time in the line
    hour = int(str(t).split(':')[0])     #Time is in HH:mm format so splitting the hours and minutes attributes
    minute = int(str(t).split(':')[1])
    hour = hour + (minute/60)        #Converting from HH:mm format to HH (hour) format only
    if hour>0 and hour<=6:
      return '0-6'
    if hour>6 and hour<=12:
      return '6-12'
    if hour>12 and hour<=18:             #returning the group based on condition
      return '12-18'
    if (hour>18 and hour<=24) or hour ==0 :
      return '18-24'

if len(sys.argv) != 3:
  raise Exception("Exactly 2 arguments are required: <inputUri> <outputUri>")       #Expection that raises when input arguments are not in the form
                                                                                        #__,input,output

inputURL=sys.argv[1]    #reading the input path
outputURL=sys.argv[2]   #reading the output path

sc = pyspark.SparkContext()
df = sc.textFile(inputURL)    #reading the input file as text
header = df.first()          # getting the header of the input file
df  = df.filter(lambda line: line != header)  #it is important to delete the header as it contains labels but not data

groups = df.map(return_group)     #apply map operation
output = groups.map(lambda element: (element, 1)).reduceByKey(lambda c1, c2: c1 + c2)  #reduce operation
```

Steps followed:

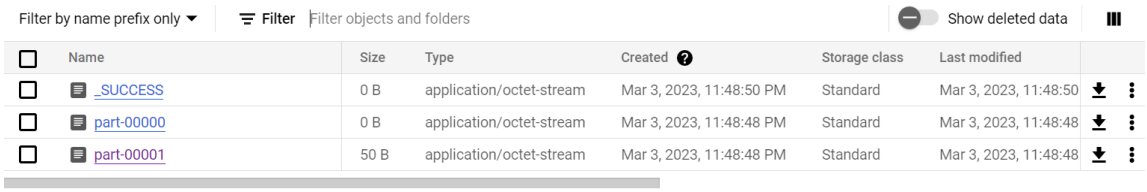
1. Enable Dataproc API's in the GCP platform
2. Create a bucket named assignment4-me19b190 to store the input and output files
3. Create a DataProc cluster using the following command

```
me19b190@cloudshell:~ (me19b190-project1)$ gcloud dataproc clusters create me19b190clusterass4     --project=me19b190-project1     --region="us-central1"     --
single-node
Waiting on operation [projects/me19b190-project1/regions/us-central1/operations/f9f1e79c-c730-32a0-9383-9d5083bbdd13].
Waiting for cluster creation operation...working.
WARNING: No image specified. Using the default image version. It is recommended to select a specific image version in production, as the default image version m
ay change at any time.
WARNING: Failed to validate permissions required for default service account: '775468331472-compute@developer.gserviceaccount.com'. Cluster creation could still
 be successful if required permissions have been granted to the respective service accounts as mentioned in the document https://cloud.google.com/dataproc/docs/
concepts/configuring-clusters/service-accounts#dataproc_service_accounts_2. If a cross project service account has been provided please make sure to follow the
instructions at https://cloud.google.com/iam/docs/impersonating-service-accounts#attaching-different-project as not configuring properly could cause cluster cre
ation failures during later stages.
Waiting for cluster creation operation...working...
```

4. Write the python code to perform the required task and save as .py file.
5. Then submit the job to the Dataproc cluster by specifying the path of input file and output file location and the python file containing the required code. The following is the command to submit the job.

```
me19b190@cloudshell:~ (me19b190-project1)$ gcloud dataproc jobs submit pyspark me19b190-groupusers-assignemnt4.py     --cluster=me19b190clusterass4     --region
="us-central1"     -- gs://assignment-4-me19b190/input/ gs://assignment-4-me19b190/output/
Job [a1304fd30b5b480ba6fa3f283b4c2215] submitted.
Waiting for job output...
23/03/03 18:18:20 INFO org.apache.spark.SparkEnv: Registering MapOutputTracker
23/03/03 18:18:20 INFO org.apache.spark.SparkEnv: Registering BlockManagerMaster
23/03/03 18:18:20 INFO org.apache.spark.SparkEnv: Registering BlockManagerMasterHeartbeat
23/03/03 18:18:20 INFO org.apache.spark.SparkEnv: Registering OutputCommitCoordinator
23/03/03 18:18:20 INFO org.sparkproject.jetty.util.log: Logging initialized @4666ms to org.sparkproject.jetty.util.log.Slf4jLog
23/03/03 18:18:20 INFO org.sparkproject.jetty.server.Server: jetty-9.4.40.v20210413; built: 2021-04-13T20:42:42.668Z; git: b881a572662e1943a14ae12e7e1207989f218
b74; jvm 1.8.0_362-b09
23/03/03 18:18:20 INFO org.sparkproject.jetty.server.Server: Started @4791ms
23/03/03 18:18:20 INFO org.sparkproject.jetty.server.AbstractConnector: Started ServerConnector@4c8eb4c5{HTTP/1.1, (http/1.1)}{0.0.0.0:38759}
23/03/03 18:18:21 INFO org.apache.hadoop.yarn.client.RMProxy: Connecting to ResourceManager at me19b190clusterass4-m/10.128.0.17:8032
23/03/03 18:18:22 INFO org.apache.hadoop.yarn.client.AHSProxy: Connecting to Application History server at me19b190clusterass4-m/10.128.0.17:10200
23/03/03 18:18:23 INFO org.apache.hadoop.conf.Configuration: resource-types.xml not found
```

6. Go to the output directory. We will find output text files containing the required output.



Output file:

The output text file, the python file have been attached with the submission.

2. Provide a brief description of the functionality of the following services:

a. HDFS

b. Hive

c. Pig

d. Yarn

a. HDFS:

HDFS (Hadoop Distributed file system) is an Apache software foundation that manages data. It is a distributed file system designed to run on commodity hardware. The main advantage of HDFS is that it is fault tolerant. It is a master slave architecture. HDFS can create multiple replicas for a particular file. Data can be replicated for data integrity. Data is stored on multiple servers so that lost data can be retrieved even in case of a local system or server failure. It enables stream data processing and data sets that usually operate on Hadoop are of huge volumes. The master node in the HDFS is called the NameNode(which distributes tasks) and slave nodes(that do the actual computation) are called datanodes.

b. Hive:

Hive is extensively used for data analysis. They offer three functionalities namely, Data Summarization, data analysis on distributed file and data query. Hive provides high query language(HQL) that supports DML and user defined functions. Hive compiler converts these queries to map reduce jobs. Hive has numerous in built mathematical/numerical functions, collection function, string functions and date functions to perform several operations. Hive also supports User defined functions.

c. Pig:

Pig is platform that is used to process large datasets. Pig represents Big Data as flows. It provides a scripting language called Pig Latin which is used for data analysis. Pig uses a query approach like SQL and Pig Latin is SQL like language. It provides many built in operators and supports nested data types like tuples, bags and maps. It supports own user defined functions. It allows a split in the pipeline. The major advantage of Pig is that is handles both structured and unstructured data.

d. Yarn:

Hadoop YARN (Yet Another Resource Negotiator) Architecture is the reference architecture for resource management for Hadoop framework components. The four major components of Resource Manager, Node Manager, Containers, and Application Master. Resource manager manages application management and job scheduling for batch process. The element known as the node manager controls how tasks are distributed across each data node in the cluster. Containers are the hardware parts of the Node, such as the CPU and RAM, that are controlled by YARN. The Hadoop cluster's Application Master is responsible for administering and monitoring the application lifecycle. YARN overcomes the scalability issue of Map Reduce. The salient features of YARN are scalability, Compatibility, Cluster utilization and Multi tenancy.