**2. DATA INGESTION**

**Part 2**

* 1. **1. Create the necessary structure in a MySQL database using the steps mentioned below: a. Create a new database in MySQL with the name *midproject***
  2. **Solution:**

**mysql -u edu\_labuser -pedureka -h dbserver.edu.cloudlab.com**

**uselabuser\_database**

**show tables; -- Permission denied inorder to create databases**

* + - 1. **b. Create a table in this database with the name *census\_adult*to store the input dataset**

**Solution:**

**CREATE TABLE IF NOT EXISTS census\_adult\_1042728 (**

**age int**

**,workclass VARCHAR(50)**

**,fnlwgt int**

**,education VARCHAR(20)**

**,education\_num int**

**,marital\_status VARCHAR(20)**

**,occupation VARCHAR(50)**

**,relationship VARCHAR(50)**

**,race VARCHAR(50)**

**,sex VARCHAR(20)**

**,capital\_gain int**

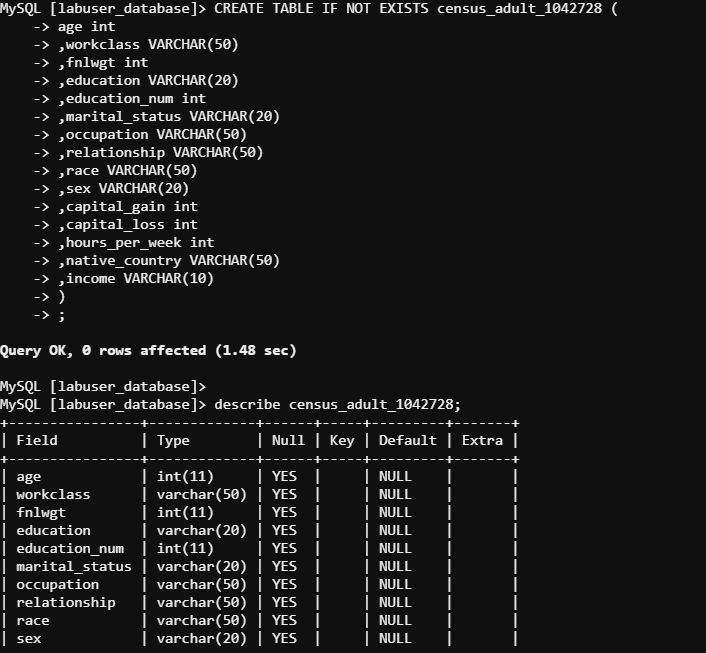
**,capital\_loss int**

**,hours\_per\_week int**

**,native\_country VARCHAR(50)**

**,income VARCHAR(10)**

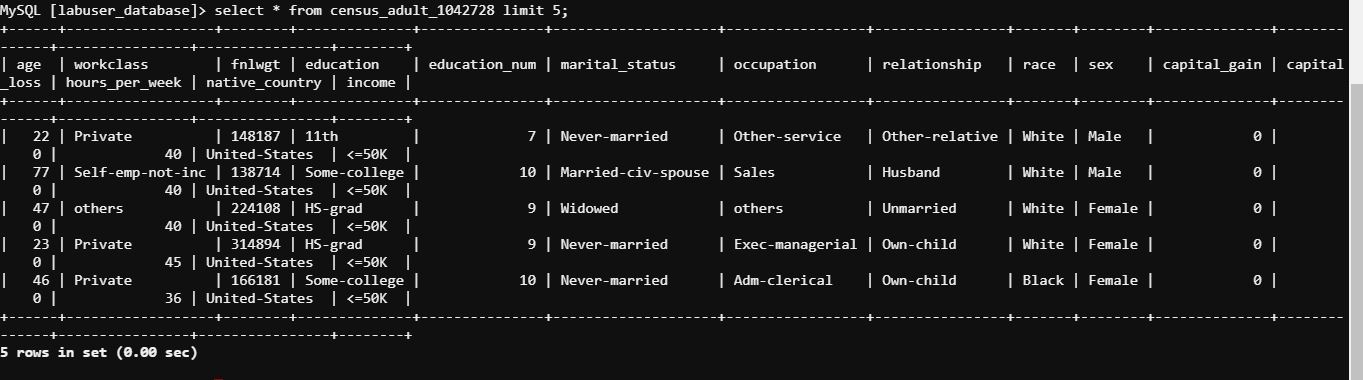
**)**



**c. Load the dataset into the table**

**LOAD DATA LOCAL INFILE '/mnt/home/edureka\_702869/Mid\_Term\_Project-2/censusdata.csv' INTO TABLE census\_adult\_1042728 FIELDS TERMINATED BY ',' IGNORE 1 LINES;**

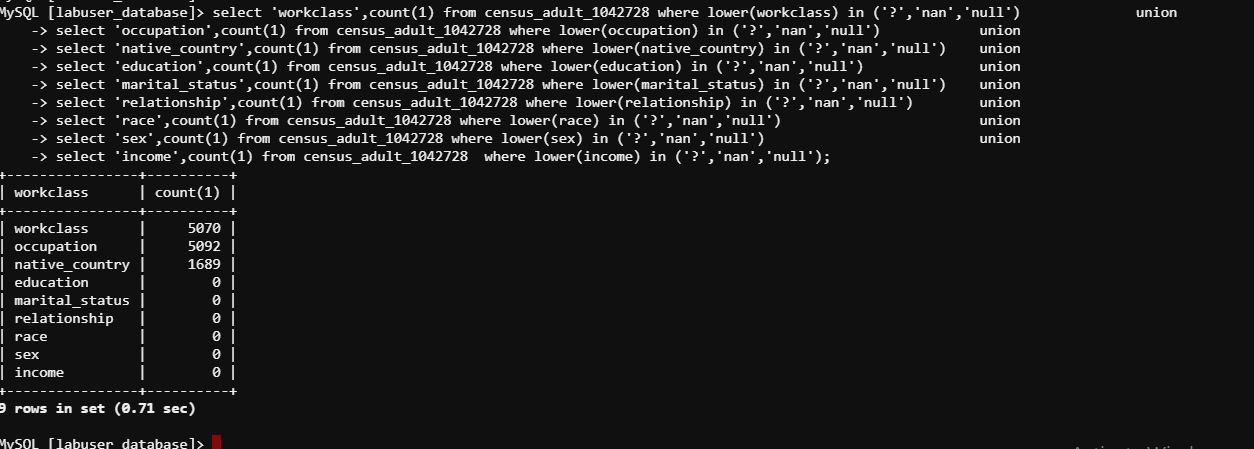
**d. Verify whether data is loaded properly**



**e. Verify the table for unwanted data such as ‘?’,’Nan’ and ‘Null’**

* 1. **select 'workclass',count(1) from census\_adult\_1042728 where lower(workclass) in ('?','nan','null') union**
  2. **select 'occupation',count(1) from census\_adult\_1042728 where lower(occupation) in ('?','nan','null') union**
  3. **select 'native\_country',count(1) from census\_adult\_1042728 where lower(native\_country) in ('?','nan','null') union**
  4. **select 'education',count(1) from census\_adult\_1042728 where lower(education) in ('?','nan','null') union**
  5. **select 'marital\_status',count(1) from census\_adult\_1042728 where lower(marital\_status) in ('?','nan','null') union**
  6. **select 'relationship',count(1) from census\_adult\_1042728 where lower(relationship) in ('?','nan','null') union**
  7. **select 'race',count(1) from census\_adult\_1042728 where lower(race) in ('?','nan','null') union**
  8. **select 'sex',count(1) from census\_adult\_1042728 where lower(sex) in ('?','nan','null') union**

**select 'income',count(1) from census\_adult\_1042728 where lower(income) in ('?','nan','null');**

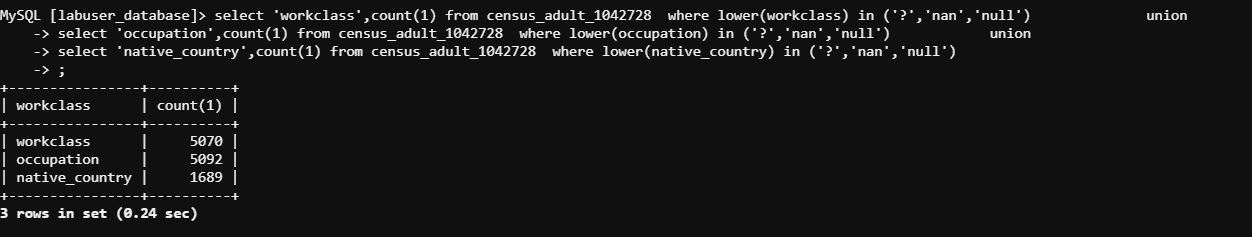
****

**f. Get the counts for the columns which contain unwanted data**

**select 'workclass',count(1) from census\_adult\_1042728 where lower(workclass) in ('?','nan','null') union**

**select 'occupation',count(1) from census\_adult\_1042728 where lower(occupation) in ('?','nan','null') union**

**select 'native\_country',count(1) from census\_adult\_1042728 where lower(native\_country) in ('?','nan','null')**



**g. Clean the data by replacing the unwanted data with *others***

**UPDATE census\_adult\_1042728 SET workclass = 'others' WHERE lower(workclass) in ('?','nan','null');**

**UPDATE census\_adult\_1042728 SET occupation = 'others' WHERE lower(occupation) in ('?','nan','null');**

**UPDATE census\_adult\_1042728 SET native\_country = 'others' WHERE lower(native\_country) in ('?','nan','null');**

**2. Import the above data from MySQL into a Hive table using Sqoop**

**sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --username edu\_labuser --password edureka --table census\_adult\_1042728 --hive-import --create-hive-table --hive-table census\_adult\_hive1\_edureka\_1042728 --fields-terminated-by ',' -m 1 --target-dir /user/edureka\_1042728/project2/pro\_hive\_1**

**3. Connect to PySpark using web console to access the created Hive table. Perform the following queries and note the time taken for execution in each of the queries.**

**from pyspark.context import SparkContext**

**from pyspark.sql import HiveContext**

**sqlContext = HiveContext(sc)**

**census\_adult\_hive = sqlContext.table("census\_adult\_hive1\_edureka\_1042728 ")**

**census\_adult\_hive.registerTempTable("census\_adult\_hive\_temp")**

**from datetime import datetime**

**3.a. Query the table to get the number of adults based on income and gender**

**before\_time = datetime.now()**

**sqlContext.sql("select income,sex,count(1) from census\_adult\_hive\_temp group by income,sex").show()**

**after\_time = datetime.now()**

**delta = after\_time - before\_time**

**time\_in\_seconds = delta.seconds + delta.microseconds/1E6**

**print("number of adults based on income and gender Time Taken: ",time\_in\_seconds)**



**3.b. Query the table to get the number of adults based on income and workclass**

**before\_time = datetime.now()**

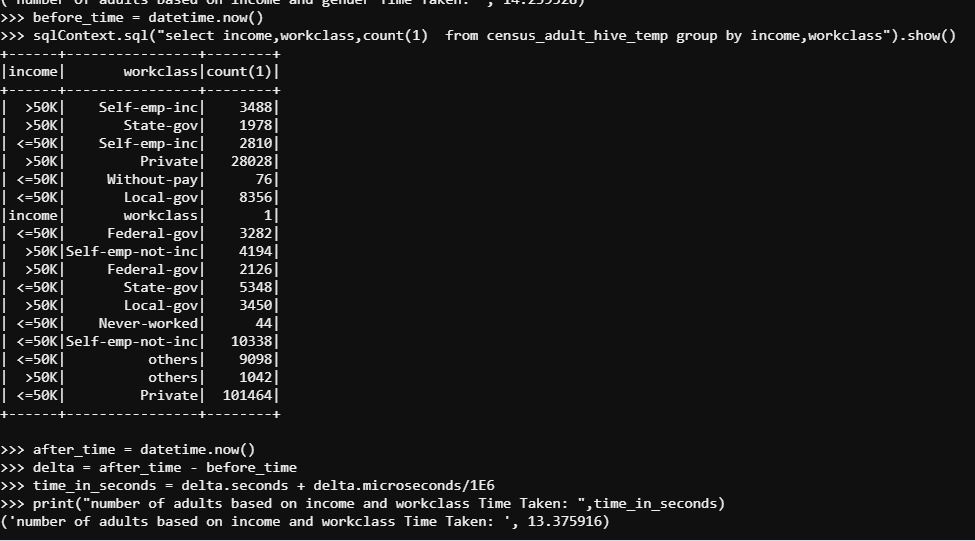
**sqlContext.sql("select income,workclass,count(1) from census\_adult\_hive\_temp group by income,workclass").show()**

**after\_time = datetime.now()**

**delta = after\_time - before\_time**

**time\_in\_seconds = delta.seconds + delta.microseconds/1E6**

**print("number of adults based on income and workclass Time Taken: ",time\_in\_seconds)**



**4. Access the following two tables created as part of Problem 1 (HDFS and Hive) and perform the steps as mentioned below: a. Access Hive External Table with partition**

**Solution:**

1. **Query the table to get the number of adults based on income and gender**

**censusdata\_partitioned\_external = sqlContext.table("work\_class\_ext")**

**censusdata\_partitioned\_external.registerTempTable("censusdata\_partitioned\_external\_temp")**

**before\_time = datetime.now()**

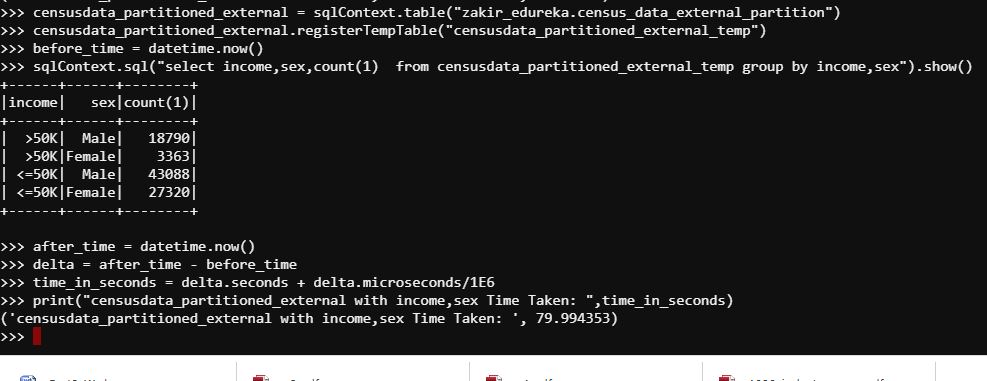
**sqlContext.sql("select income,sex,count(1) from censusdata\_partitioned\_external\_temp group by income,sex").show()**

**after\_time = datetime.now()**

**delta = after\_time - before\_time**

**time\_in\_seconds = delta.seconds + delta.microseconds/1E6**

**print("censusdata\_partitioned\_external with income,sex Time Taken: ",time\_in\_seconds)**

****

1. **Query the table to get the number of adults based on income and workclass**

**before\_time = datetime.now()**

**sqlContext.sql("select income,workclass,count(1) from censusdata\_partitioned\_external\_temp group by income,workclass").show()**

**after\_time = datetime.now()**

**delta = after\_time - before\_time**

**time\_in\_seconds = delta.seconds + delta.microseconds/1E6**

**print("censusdata\_partitioned\_external with income,workclass Time Taken: ",time\_in\_seconds)**



**b. Access Hive Internal Table with Partition**

**i. Query the table to get the number of adults based on income and gender**

**censusdata\_partitioned = sqlContext.table("workclass1")**

**censusdata\_partitioned.registerTempTable("censusdata\_partitioned\_temp")**

**before\_time = datetime.now()**

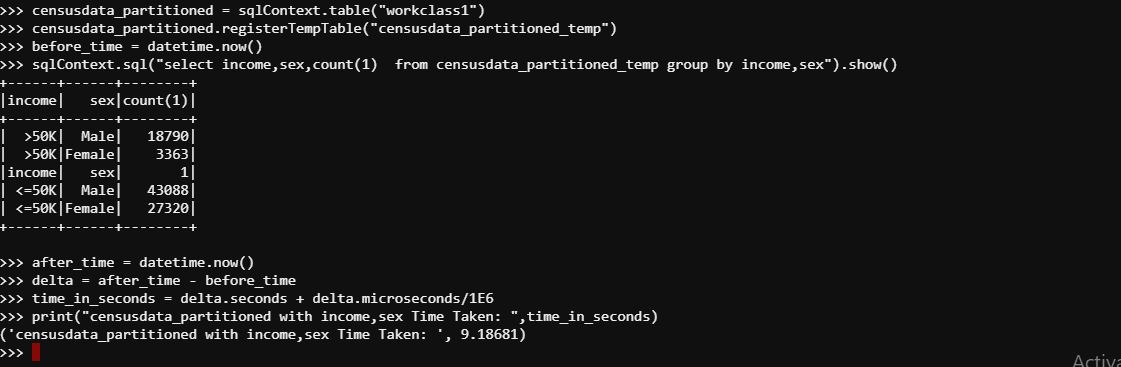
**sqlContext.sql("select income,sex,count(1) from censusdata\_partitioned\_temp group by income,sex").show()**

**after\_time = datetime.now()**

**delta = after\_time - before\_time**

**time\_in\_seconds = delta.seconds + delta.microseconds/1E6**

**print("censusdata\_partitioned with income,sex Time Taken: ",time\_in\_seconds)**



**ii. Query the table to get the number of adults based on income and workclass**

**before\_time = datetime.now()**

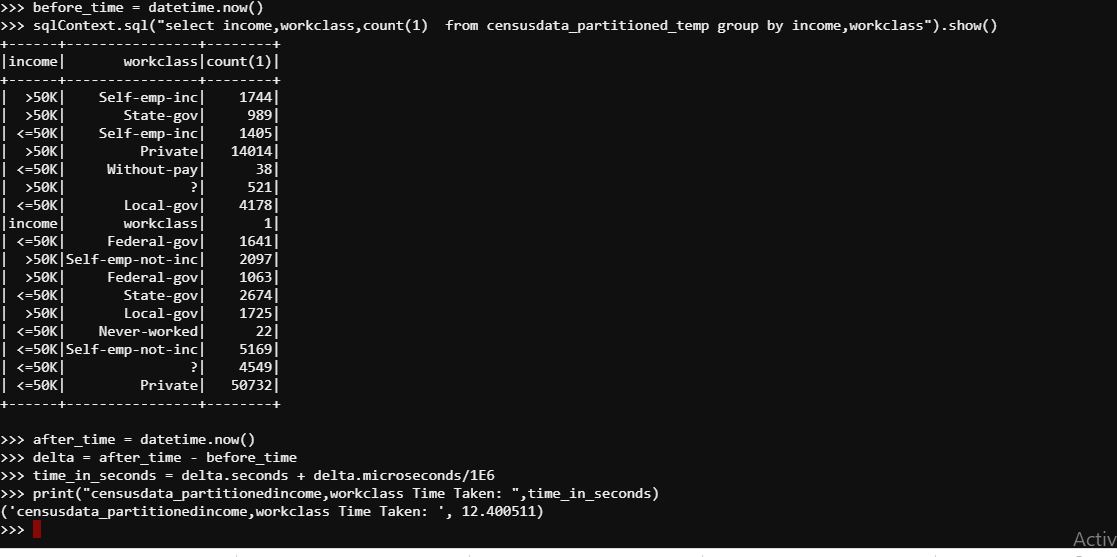
**sqlContext.sql("select income,workclass,count(1) from censusdata\_partitioned\_temp group by income,workclass").show()**

**after\_time = datetime.now()**

**delta = after\_time - before\_time**

**time\_in\_seconds = delta.seconds + delta.microseconds/1E6**

**print("censusdata\_partitionedincome,workclass Time Taken: ",time\_in\_seconds)**



**5. Comment on the time taken for executing these commands using Spark as compared to the time taken for execution in Hive (Problem Statement 1).**

**Solution:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Spark** | | | **Hive** | | |
| Table | 4 Row | 6 Row | Table | 4 Row | 6 Row |
| **census\_adult\_hive\_edureka\_702869** | 8.9 | 13.7 | **censusdata** | 24.25 | 21.77 |
| **census\_data\_external\_partition** | 19.8 | 17.1 | **census\_data\_partition** | 34.54 | 26.28 |
| **census\_data\_partition** | 9.1 | 12.4 | **Census\_data\_external\_partition** | 19.17 | 18.11 |