Cloud Computing And Big Data Project Report

Prof. Anwar Hossain

Group 16

Reported By:

Ahmed Elsayed Salama (21aes20)

Fatma Eldesouky (21feae)

Mahmoud Adel Khorshed (21mamm2)

Zeyad Tarek Mohamed (21ztem)

July 2022

Introduction And Objective

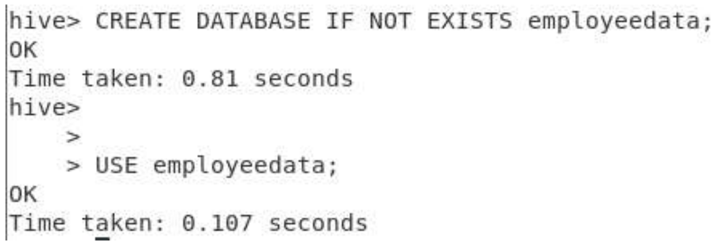
The Apache Hive ™ data warehouse software facilitates reading, writing, and managing large datasets residing in distributed storage using SQL. Structure can be projected onto data already in storage.

Apache Spark is a lightning-fast cluster computing designed for fast computation. It was built on top of Hadoop MapReduce, and it extends the MapReduce model to efficiently use more types of computations which includes Interactive Queries and Stream Processing. This is a brief tutorial that explains the basics of Spark Core programming.

Our objective here we want to create a database have three tables the **(employee-data-hive, department-data-hive, employee_ptn)** and we have the csv data **(Chicago employee)**, so we want to do some data analysis on these tables to have get useful insights from their data using hive commands or spark commands.

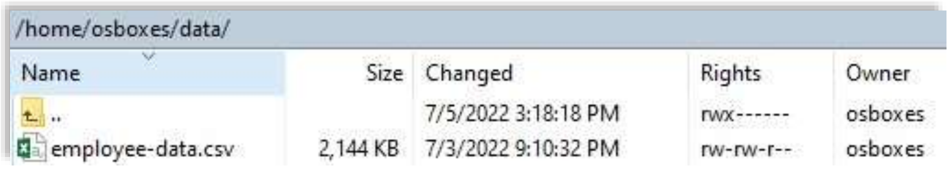
Before starting to do our project, we need to setup and do some important steps to make doing our tasks possible.

1. First, we create a database as the namespace of our tables that we want to create and use it.



```
hive> CREATE DATABASE IF NOT EXISTS employeedata;
OK
Time taken: 0.81 seconds
hive>
    >
    > USE employeedata;
OK
Time taken: 0.107 seconds
```

2. We moved the **employee-data.csv** file from **local machine** to local folder in VM OS through WinSCP software.



/home/osboxes/data/

Name	Size	Changed	Rights	Owner
..		7/5/2022 3:18:18 PM	rwx------	osboxes
employee-data.csv	2,144 KB	7/3/2022 9:10:32 PM	rw-rw-r--	osboxes

3. Then We copied this csv file into a hdfs folder because we will need later in the last task.

```
[osboxes@quickstart-bigdata ~]$ hdfs dfs -put /home/osboxes/project/employee-data.csv /user/osboxes/inputdata/
```

4. We changed some settings in hive to enable us using **DML** operations and we target to use **UPDATE** clause in **task 4**.

   Add these properties with these values in tag format in etc/gedit/hive-site.xml file or set them directly in hive shell as commands.

   A. **hive.support.concurrency – true**
   B. **hive.enforce.buckting – true**
   C. **hive.exec.dynamic.partition.mode – nonstrict**

D. **hive.txn.manager – org.apache.hadoop.hive.ql.lockmgr.DbTxnManager**
E. **hive.compactor.initiator.on – true**
F. **hive.compactor.worker.threads – 1**

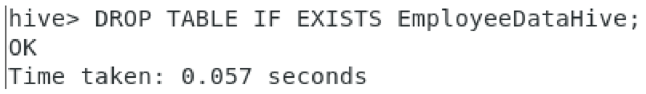
Note: to preform hive **CRUD** using **ACID** operations, your **hive** version must be **0.14** or above the table format must be **ORC** file format with **TBLPROPERTEIS('transactional' = 'true')** , also the table on which you want to perform the update operation must be **CLUSTERD BY** with some **buckets**.

Task 1

Goal: Create a Hive table named employee-data-hive based on the given dataset.

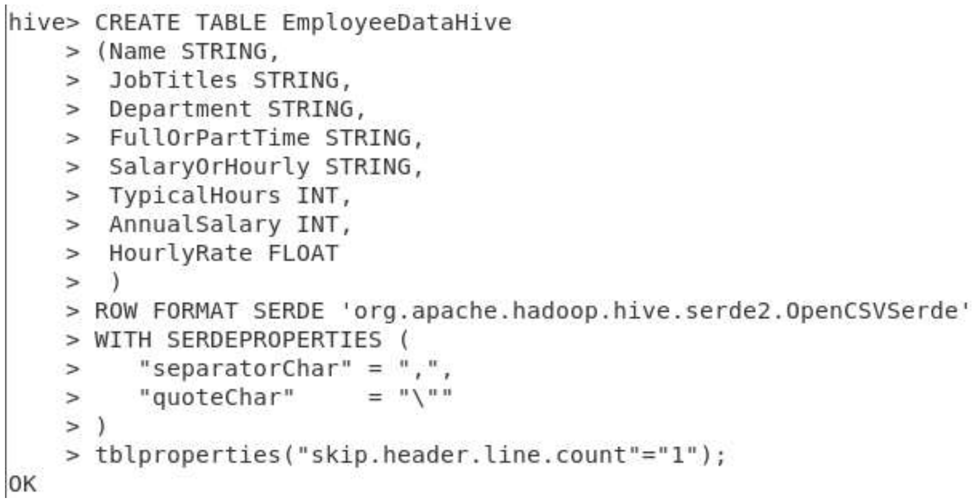
Solution

1. we used (drop table) command to drop any other tables that had the same name to prevent overwriting on it.



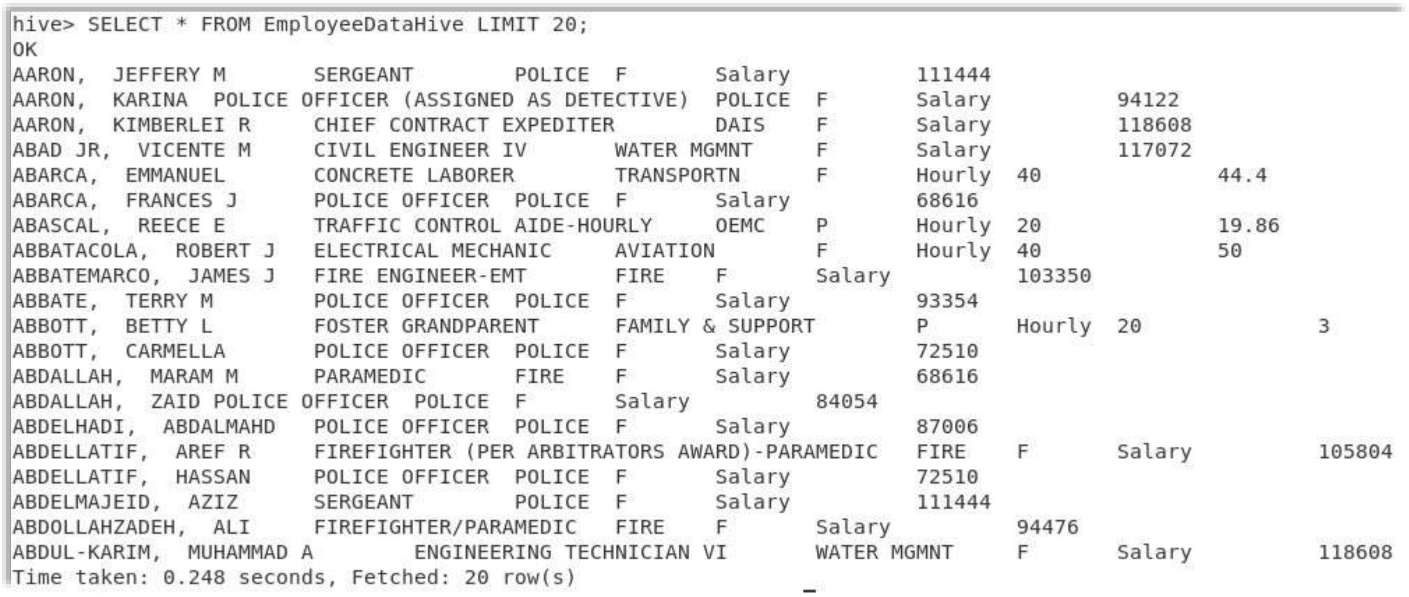
```
hive> DROP TABLE IF EXISTS EmployeeDataHive;
OK
Time taken: 0.057 seconds
```

2. Now we created (**EmployeeDataHive**) table to store employee's data.
3. The (**employee-data.CSV**) file is separated by comma but there is a **Name** column separate between first and last name by comma, also we found in row **24401** in **JobTitles** column there is value separated by comma too, so it will be conflicted when loading data into a table and separate it by comma.
4. To solve this issue, we used **OpenCSVSedre** library.
5. The **OpenCSVSerde** library has the following characteristics for string data: Uses double quotes (") as the default quote character, and allows you to specify separator, quote, and escape characters
6. Then we used **tblproperties** library to skip header in first row.



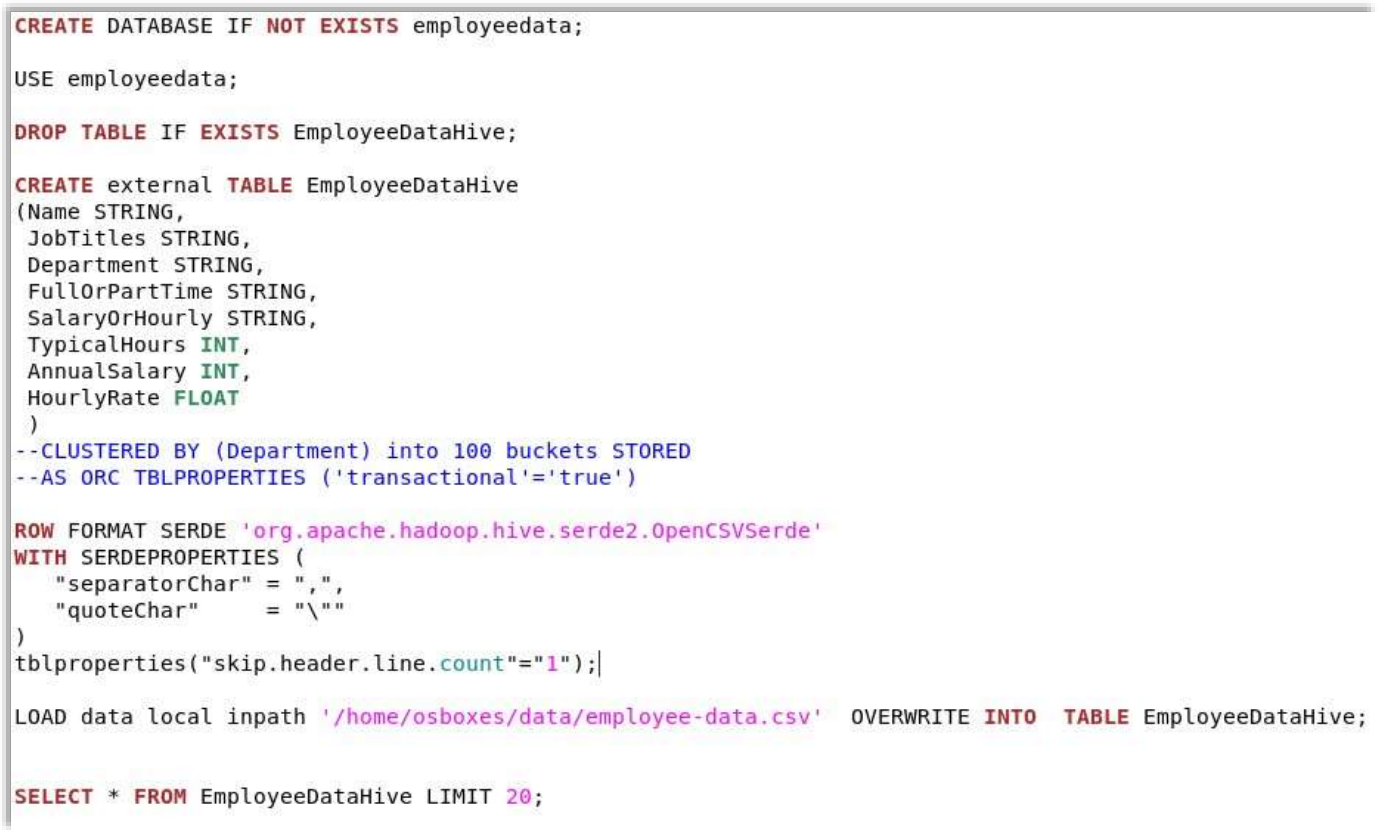
```
hive> CREATE TABLE EmployeeDataHive
    > (Name STRING,
    >  JobTitles STRING,
    >  Department STRING,
    >  FullOrPartTime STRING,
    >  SalaryOrHourly STRING,
    >  TypicalHours INT,
    >  AnnualSalary INT,
    >  HourlyRate FLOAT
    >  )
    > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
    > WITH SERDEPROPERTIES (
    >    "separatorChar" = ",",
    >    "quoteChar"     = "\""
    > )
    > tblproperties("skip.header.line.count"="1");
OK
```

7. Finally, we did an information retrieval query after we loaded our csv data into the table (**EmployeeDataHive**), the number of records is **32928 (header doesn't included)**.



```
hive> SELECT * FROM EmployeeDataHive LIMIT 20;
OK
AARON,  JEFFERY M       SERGEANT        POLICE  F       Salary          111444
AARON,  KARINA  POLICE OFFICER (ASSIGNED AS DETECTIVE)  POLICE  F       Salary          94122
AARON,  KIMBERLEI R     CHIEF CONTRACT EXPEDITER        DAIS    F       Salary          118608
ABAD JR,  VICENTE M     CIVIL ENGINEER IV       WATER MGMNT     F       Salary          117072
ABARCA,  EMMANUEL       CONCRETE LABORER        TRANSPORTN      F       Hourly  40              44.4
ABARCA,  FRANCES J      POLICE OFFICER  POLICE  F       Salary          68616
ABASCAL,  REECE E       TRAFFIC CONTROL AIDE-HOURLY     OEMC    P       Hourly  20              19.86
ABBATACOLA,  ROBERT J   ELECTRICAL MECHANIC     AVIATION        F       Hourly  40              50
ABBATEMARCO,  JAMES J   FIRE ENGINEER-EMT       FIRE    F       Salary          103350
ABBATE,  TERRY M        POLICE OFFICER  POLICE  F       Salary          93354
ABBOTT,  BETTY L        FOSTER GRANDPARENT      FAMILY & SUPPORT        P       Hourly  20              3
ABBOTT,  CARMELLA       POLICE OFFICER  POLICE  F       Salary          72510
ABDALLAH,  MARAM M      PARAMEDIC       FIRE    F       Salary          68616
ABDALLAH,  ZAID POLICE OFFICER  POLICE  F       Salary          84054
ABDELHADI,  ABDALMAHD   POLICE OFFICER  POLICE  F       Salary          87006
ABDELLATIF,  AREF R     FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC   FIRE    F       Salary          105804
ABDELLATIF,  HASSAN     POLICE OFFICER  POLICE  F       Salary          72510
ABDELMAJEID,  AZIZ      SERGEANT        POLICE  F       Salary          111444
ABDOLLAHZADEH,  ALI     FIREFIGHTER/PARAMEDIC   FIRE    F       Salary          94476
ABDUL-KARIM,  MUHAMMAD A        ENGINEERING TECHNICIAN VI       WATER MGMNT     F       Salary          118608
Time taken: 0.248 seconds, Fetched: 20 row(s)
```

The Final script for task 1



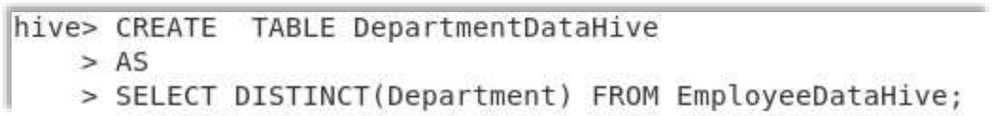
```
CREATE DATABASE IF NOT EXISTS employeedata;

USE employeedata;

DROP TABLE IF EXISTS EmployeeDataHive;

CREATE external TABLE EmployeeDataHive
(Name STRING,
 JobTitles STRING,
 Department STRING,
 FullOrPartTime STRING,
 SalaryOrHourly STRING,
 TypicalHours INT,
 AnnualSalary INT,
 HourlyRate FLOAT
 )
--CLUSTERED BY (Department) into 100 buckets STORED
--AS ORC TBLPROPERTIES ('transactional'='true')

ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES (
   "separatorChar" = ",",
   "quoteChar"     = "\""
)
tblproperties("skip.header.line.count"="1");

LOAD data local inpath '/home/osboxes/data/employee-data.csv'  OVERWRITE INTO  TABLE EmployeeDataHive;


SELECT * FROM EmployeeDataHive LIMIT 20;
```

Task 2

Goal: Create a department-data-hive table by selecting unique department names from the employee-data-hive and adding a column named DeptID in the new department-data-hive table and put unique values in the DeptID column.

Solution

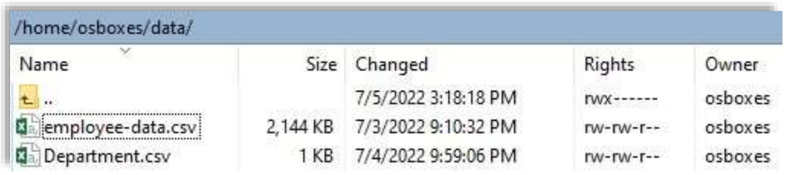
1. We first get unique department name from EmployeeDataHive table and show the unique department names.



```
hive> CREATE  TABLE DepartmentDataHive
    > AS
    > SELECT DISTINCT(Department) FROM EmployeeDataHive;
```

Note: the output here is only the unique departments names without and key or referred number column.

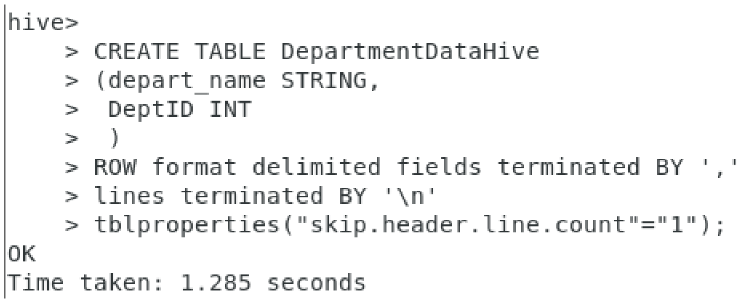
2. Then we took the result of the SELECT query manually and put it into an **EXCEL** sheet, then we generate an auto numeric incremental column started from 1 to the last department to refer to each department by a unique number this column name "**deptID**" by that the format of that sheet became (**depart_name**, **DeptID**), we saved that file as **Department.csv** and export that file into a csv file and move it to **VMware OS** to create a new table from that csv file.



/home/osboxes/data/

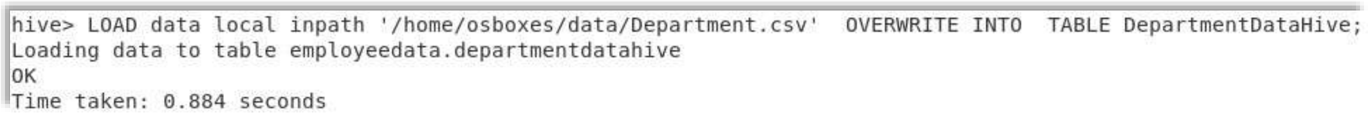
Name	Size	Changed	Rights	Owner
..		7/5/2022 3:18:18 PM	rwx------	osboxes
employee-data.csv	2,144 KB	7/3/2022 9:10:32 PM	rw-rw-r--	osboxes
Department.csv	1 KB	7/4/2022 9:59:06 PM	rw-rw-r--	osboxes

3. We created new table (**DepartmentDataHive**) from the new imported csv file to store the new data of Department into it.



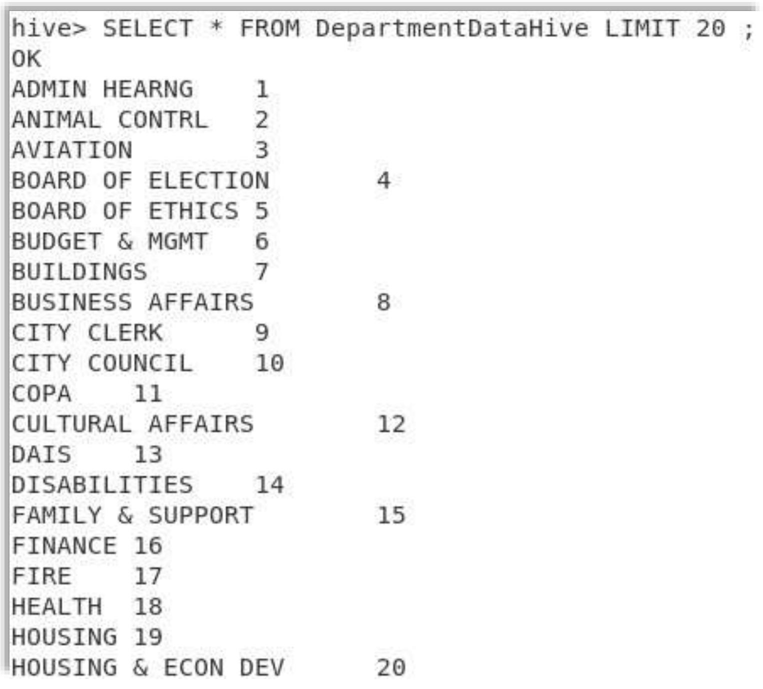
```
hive>
    > CREATE TABLE DepartmentDataHive
    > (depart_name STRING,
    >  DeptID INT
    >  )
    > ROW format delimited fields terminated BY ','
    > lines terminated BY '\n'
    > tblproperties("skip.header.line.count"="1");
OK
Time taken: 1.285 seconds
```

4. Load the Department.csv data file from local path to the table(**DepartmentDataHive**)



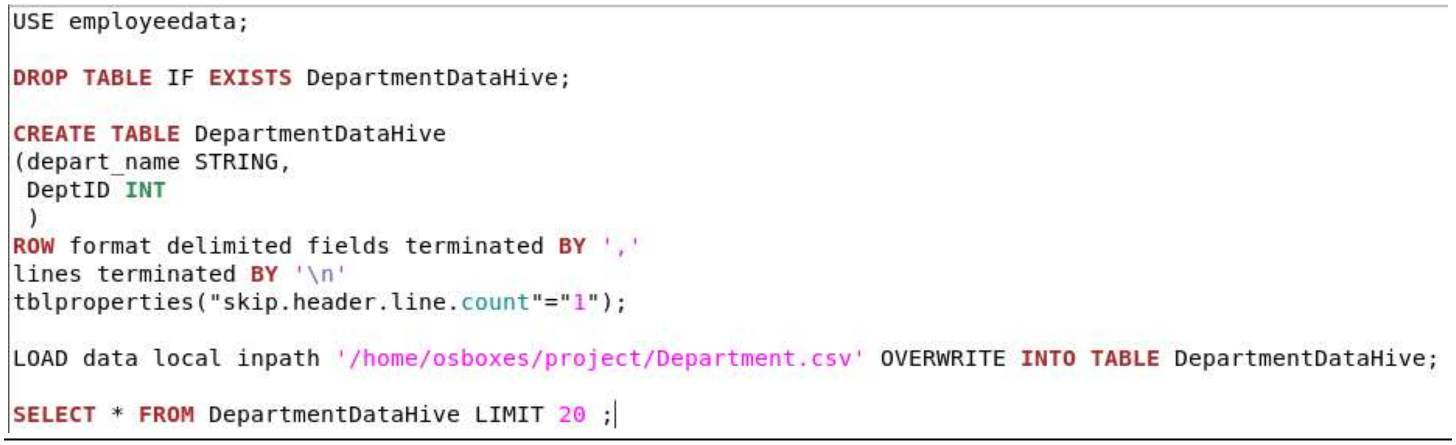
```
hive> LOAD data local inpath '/home/osboxes/data/Department.csv'  OVERWRITE INTO  TABLE DepartmentDataHive;
Loading data to table employeedata.departmentdatahive
OK
Time taken: 0.884 seconds
```

5. Then we showed our data after loaded into table (**DepartmentDataHive**), the number of departments is **36**.



```
hive> SELECT * FROM DepartmentDataHive LIMIT 20 ;
OK
ADMIN HEARNG    1
ANIMAL CONTRL   2
AVIATION        3
BOARD OF ELECTION       4
BOARD OF ETHICS 5
BUDGET & MGMT   6
BUILDINGS       7
BUSINESS AFFAIRS        8
CITY CLERK      9
CITY COUNCIL    10
COPA    11
CULTURAL AFFAIRS        12
DAIS    13
DISABILITIES    14
FAMILY & SUPPORT        15
FINANCE 16
FIRE    17
HEALTH  18
HOUSING 19
HOUSING & ECON DEV      20
```

The Final script for task 2

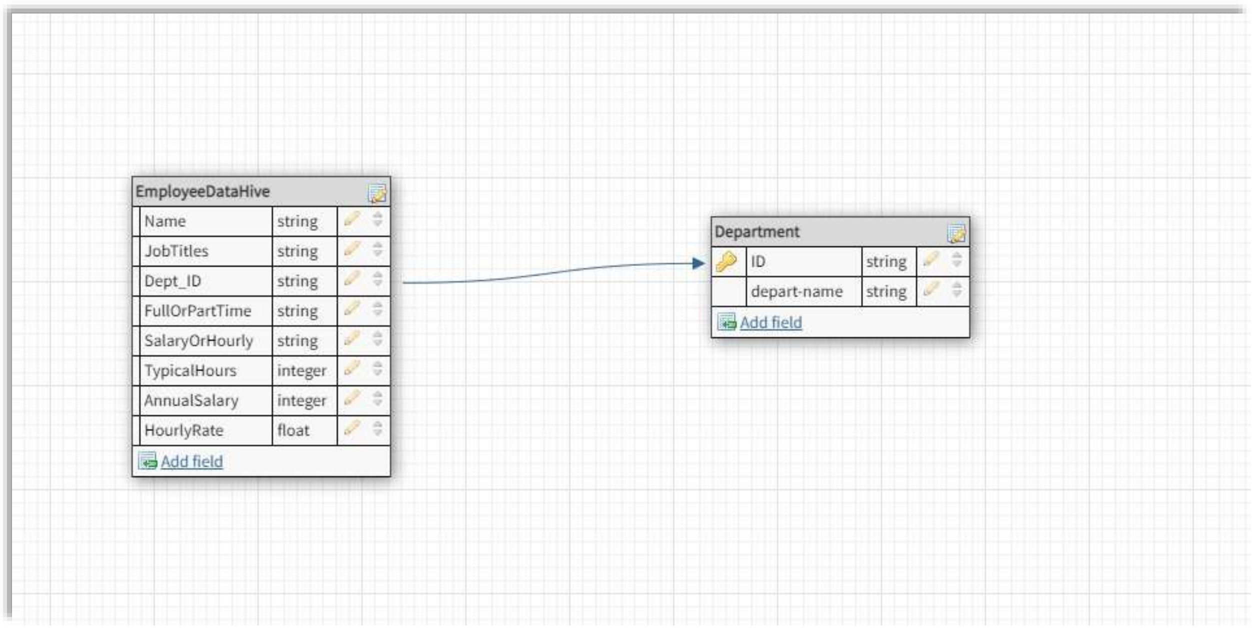


```
USE employeedata;

DROP TABLE IF EXISTS DepartmentDataHive;

CREATE TABLE DepartmentDataHive
(depart_name STRING,
 DeptID INT
 )
ROW format delimited fields terminated BY ','
lines terminated BY '\n'
tblproperties("skip.header.line.count"="1");

LOAD data local inpath '/home/osboxes/project/Department.csv' OVERWRITE INTO TABLE DepartmentDataHive;

SELECT * FROM DepartmentDataHive LIMIT 20 ;
```

Task 3

Goal 3.1 Update the employee-data-hive table by replacing the department field data with the deptID values as created in the department-data-hive table.

In this task, we need to create relation as represented in the below graph between **EmployeeDataHive** table and **DepartmentDataHive** to know the **ID** of each department name in **EmployeeDataHive** and to get that relationship with will use **JOIN CLAUSE**.

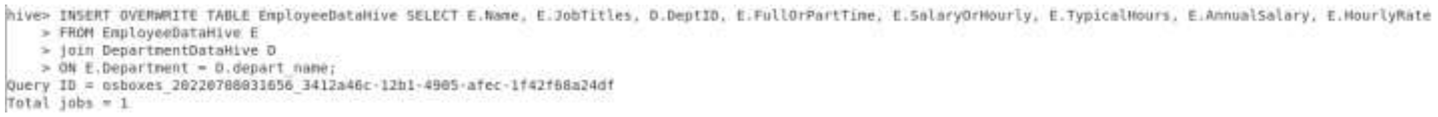


In this task we have 2 solutions:-

1. Using **INSERT OVERWRITE with JOIN clause**.
2. Using **CTAS** with **JOIN** clause.

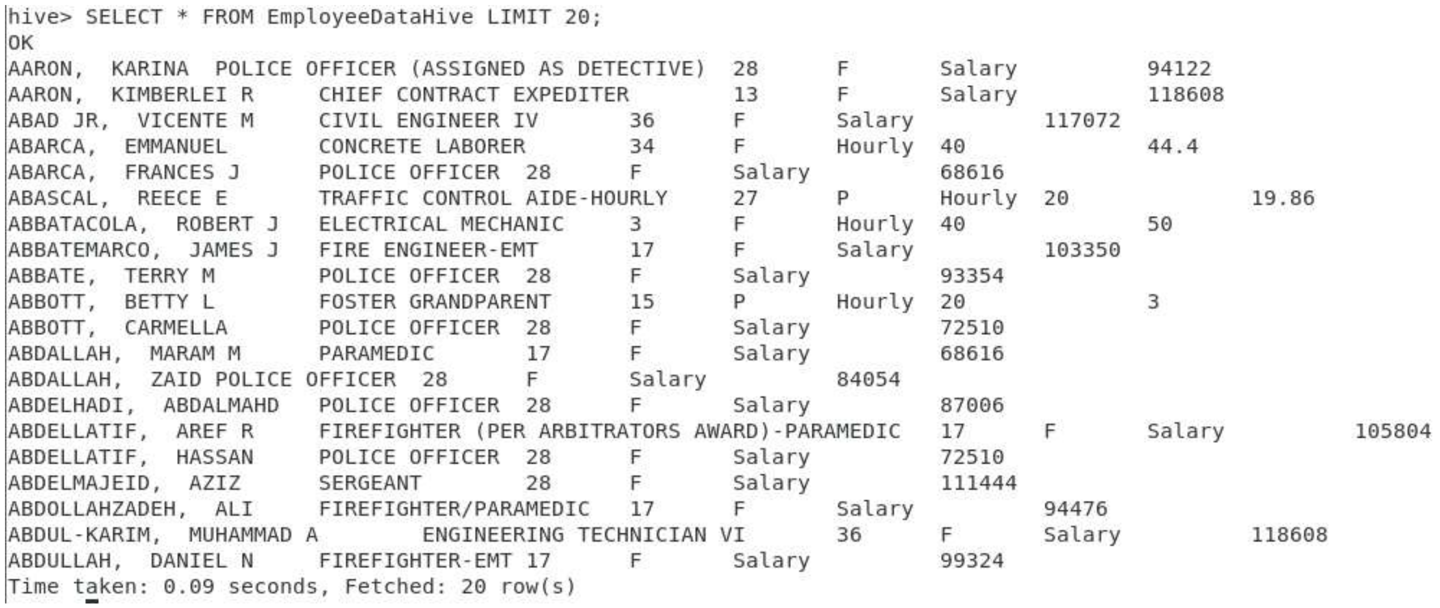
First Solution

1- Overwrite the data with **INSERT OVERWRTIE** clause.
2- Select the data that we want to replace it the old columns and it's same as its, but we'll replace only department column with DeptID column which is existing in **DepartmentDataHive** table.
3- We want to join **EmployeeDataHive** table and **DepartmentDataHive** table to find a relationship between these tables and that's relationship is based on **department** column in **EmployeeDataHive** table and **depart_name** column in **DepartmentDataHive** table, by that we will be able to know what's the **DeptID** for each department name in **EmployeeDataHive** table.



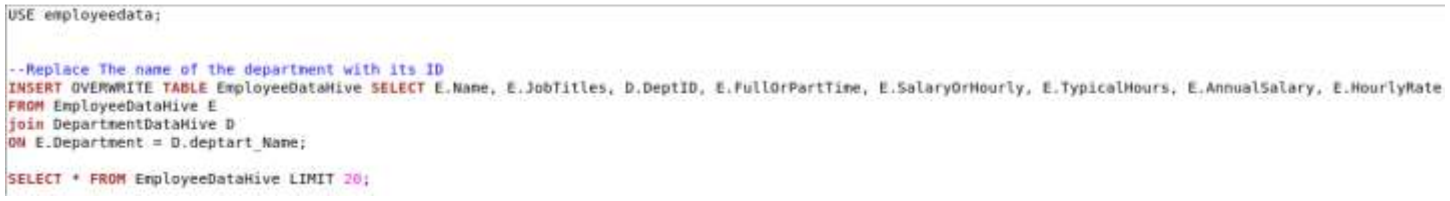
```
hive> INSERT OVERWRITE TABLE EmployeeDataHive SELECT E.Name, E.JobTitles, D.DeptID, E.FullOrPartTime, E.SalaryOrHourly, E.TypicalHours, E.AnnualSalary, E.HourlyRate
    > FROM EmployeeDataHive E
    > join DepartmentDataHive D
    > ON E.Department = D.depart_name;
Query ID = osboxes_20220708031656_3412a46c-12b1-4905-afec-1f42f68a24df
Total jobs = 1
```

4- Verify that the department name is relaced with their IDs correctly.



```
hive> SELECT * FROM EmployeeDataHive LIMIT 20;
OK
AARON,  KARINA  POLICE OFFICER (ASSIGNED AS DETECTIVE)  28      F       Salary          94122
AARON,  KIMBERLEI R     CHIEF CONTRACT EXPEDITER        13      F       Salary          118608
ABAD JR,  VICENTE M     CIVIL ENGINEER IV       36      F       Salary          117072
ABARCA,  EMMANUEL       CONCRETE LABORER        34      F       Hourly  40              44.4
ABARCA,  FRANCES J      POLICE OFFICER  28      F       Salary          68616
ABASCAL,  REECE E       TRAFFIC CONTROL AIDE-HOURLY     27      P       Hourly  20              19.86
ABBATACOLA,  ROBERT J   ELECTRICAL MECHANIC     3       F       Hourly  40              50
ABBATEMARCO,  JAMES J   FIRE ENGINEER-EMT       17      F       Salary          103350
ABBATE,  TERRY M        POLICE OFFICER  28      F       Salary          93354
ABBOTT,  BETTY L        FOSTER GRANDPARENT      15      P       Hourly  20              3
ABBOTT,  CARMELLA       POLICE OFFICER  28      F       Salary          72510
ABDALLAH,  MARAM M      PARAMEDIC       17      F       Salary          68616
ABDALLAH,  ZAID POLICE OFFICER  28      F       Salary          84054
ABDELHADI,  ABDALMAHD   POLICE OFFICER  28      F       Salary          87006
ABDELLATIF,  AREF R     FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC   17      F       Salary          105804
ABDELLATIF,  HASSAN     POLICE OFFICER  28      F       Salary          72510
ABDELMAJEID,  AZIZ      SERGEANT        28      F       Salary          111444
ABDOLLAHZADEH,  ALI     FIREFIGHTER/PARAMEDIC   17      F       Salary          94476
ABDUL-KARIM,  MUHAMMAD A        ENGINEERING TECHNICIAN VI       36      F       Salary          118608
ABDULLAH,  DANIEL N     FIREFIGHTER-EMT 17      F       Salary          99324
Time taken: 0.09 seconds, Fetched: 20 row(s)
```

The Final script for task 3.1.1

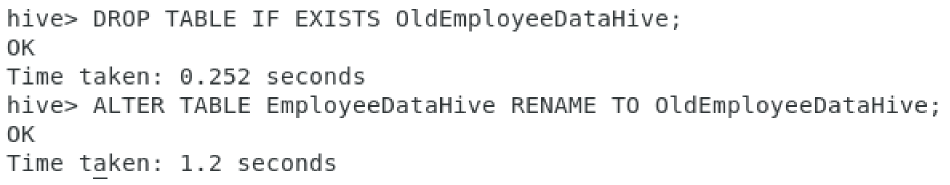


```
USE employeedata;


--Replace The name of the department with its ID
INSERT OVERWRITE TABLE EmployeeDataHive SELECT E.Name, E.JobTitles, D.DeptID, E.FullOrPartTime, E.SalaryOrHourly, E.TypicalHours, E.AnnualSalary, E.HourlyRate
FROM EmployeeDataHive E
join DepartmentDataHive D
ON E.Department = D.deptart_Name;

SELECT * FROM EmployeeDataHive LIMIT 20;
```

==

Second Solution

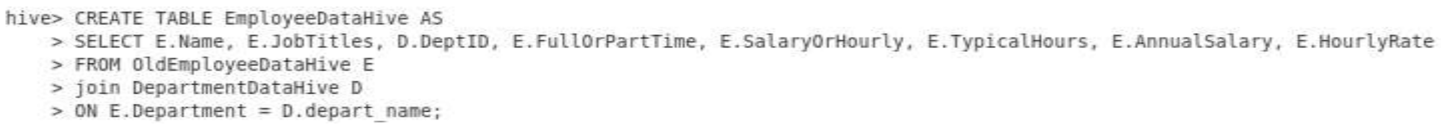
1- We **renamed** the **EmployeeDataHive** table to **OldEmployeeDataHive** to use the **EmployeeDataHive** table to store the new update.



```
hive> DROP TABLE IF EXISTS OldEmployeeDataHive;
OK
Time taken: 0.252 seconds
hive> ALTER TABLE EmployeeDataHive RENAME TO OldEmployeeDataHive;
OK
Time taken: 1.2 seconds
```

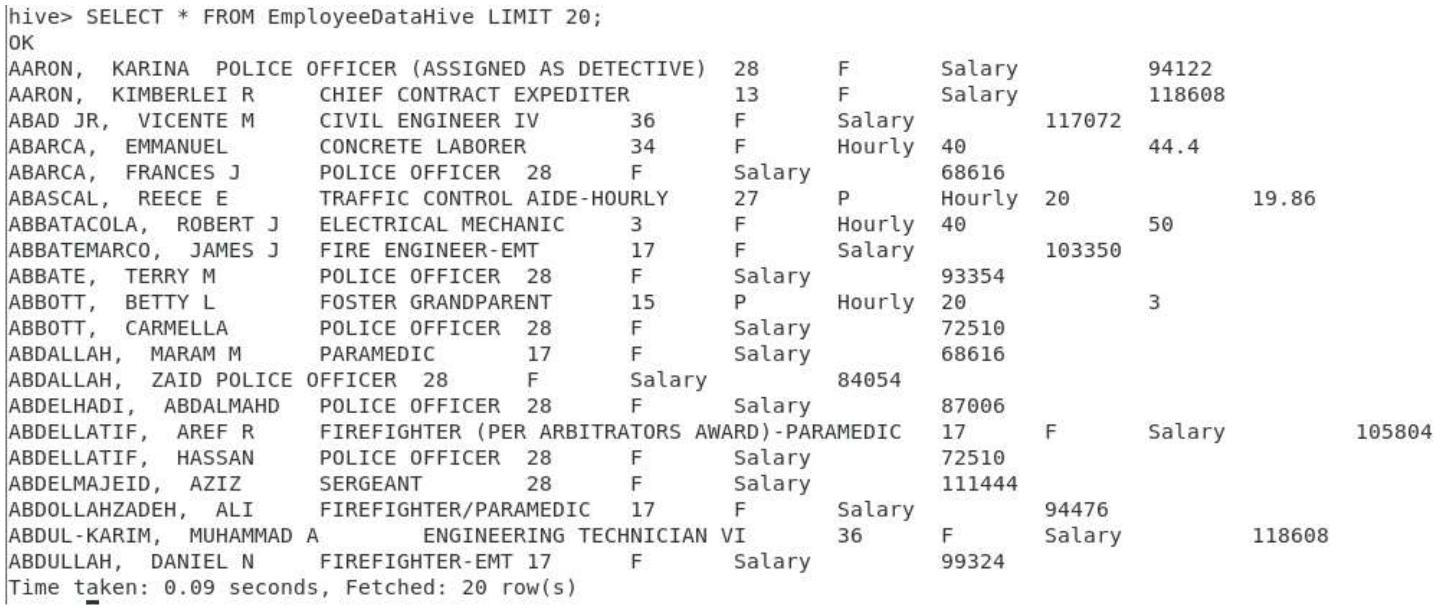
Note: Drop operation here just to not give us an error if **OldEmployeeDataHive** if it's already existing.

2- Then we used **CREATE AS SELECT** to create the new empty **EmployeeDataHive** table and we added the required columns and replaced the **department** column by joining the two tables (**OldEmployeeDataHive, DepartmentDataHive**) based on Department name, then select specific columns from the new table (remove department name from **EmployeeDataHive** **table** and replace it **DeptID** instead).



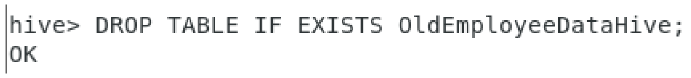
```
hive> CREATE TABLE EmployeeDataHive AS
    > SELECT E.Name, E.JobTitles, D.DeptID, E.FullOrPartTime, E.SalaryOrHourly, E.TypicalHours, E.AnnualSalary, E.HourlyRate
    > FROM OldEmployeeDataHive E
    > join DepartmentDataHive D
    > ON E.Department = D.depart_name;
```

3- Then we showed our data after updating the department name into **DeptID** from the table (**EmployeeDataHive**).



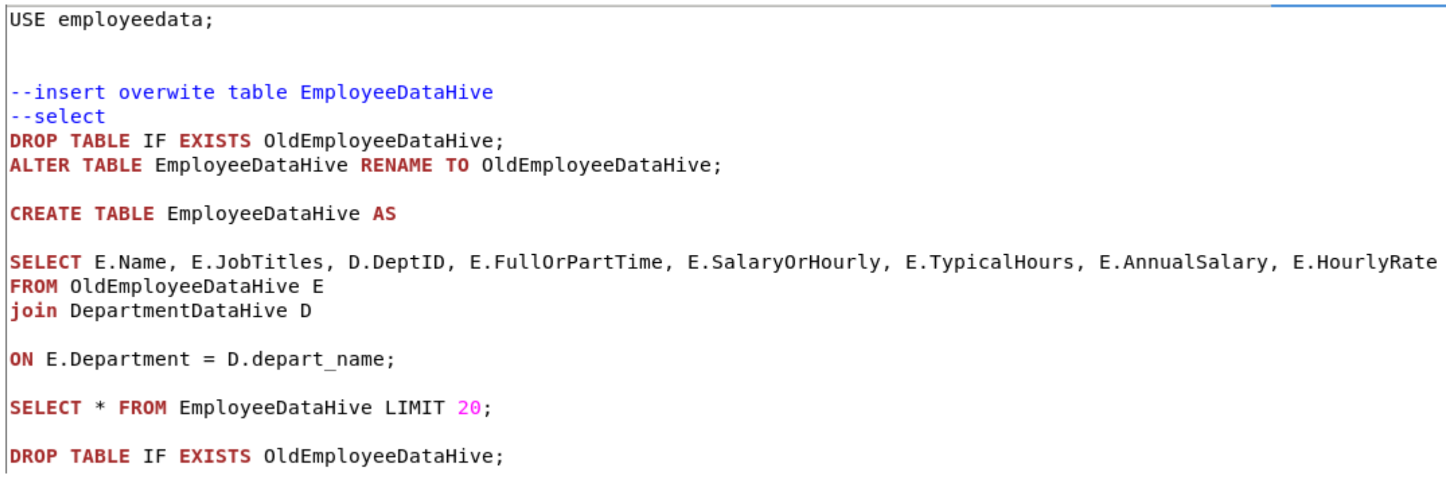
```
hive> SELECT * FROM EmployeeDataHive LIMIT 20;
OK
AARON,  KARINA  POLICE OFFICER (ASSIGNED AS DETECTIVE)  28      F       Salary          94122
AARON,  KIMBERLEI R     CHIEF CONTRACT EXPEDITER        13      F       Salary          118608
ABAD JR,  VICENTE M     CIVIL ENGINEER IV       36      F       Salary          117072
ABARCA,  EMMANUEL       CONCRETE LABORER        34      F       Hourly  40              44.4
ABARCA,  FRANCES J      POLICE OFFICER  28      F       Salary          68616
ABASCAL,  REECE E       TRAFFIC CONTROL AIDE-HOURLY     27      P       Hourly  20              19.86
ABBATACOLA,  ROBERT J   ELECTRICAL MECHANIC     3       F       Hourly  40              50
ABBATEMARCO,  JAMES J   FIRE ENGINEER-EMT       17      F       Salary          103350
ABBATE,  TERRY M        POLICE OFFICER  28      F       Salary          93354
ABBOTT,  BETTY L        FOSTER GRANDPARENT      15      P       Hourly  20              3
ABBOTT,  CARMELLA       POLICE OFFICER  28      F       Salary          72510
ABDALLAH,  MARAM M      PARAMEDIC       17      F       Salary          68616
ABDALLAH,  ZAID POLICE OFFICER  28      F       Salary          84054
ABDELHADI,  ABDALMAHD   POLICE OFFICER  28      F       Salary          87006
ABDELLATIF,  AREF R     FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC   17      F       Salary          105804
ABDELLATIF,  HASSAN     POLICE OFFICER  28      F       Salary          72510
ABDELMAJEID,  AZIZ      SERGEANT        28      F       Salary          111444
ABDOLLAHZADEH,  ALI     FIREFIGHTER/PARAMEDIC   17      F       Salary          94476
ABDUL-KARIM,  MUHAMMAD A        ENGINEERING TECHNICIAN VI       36      F       Salary          118608
ABDULLAH,  DANIEL N     FIREFIGHTER-EMT 17      F       Salary          99324
Time taken: 0.09 seconds, Fetched: 20 row(s)
```

4- Drop **OldEmployeeDataHive** because we don’t need it or its data anymore.



```
hive> DROP TABLE IF EXISTS OldEmployeeDataHive;
OK
```

The Final script for task 3.1.2



```
USE employeedata;


--insert overwite table EmployeeDataHive
--select
DROP TABLE IF EXISTS OldEmployeeDataHive;
ALTER TABLE EmployeeDataHive RENAME TO OldEmployeeDataHive;

CREATE TABLE EmployeeDataHive AS

SELECT E.Name, E.JobTitles, D.DeptID, E.FullOrPartTime, E.SalaryOrHourly, E.TypicalHours, E.AnnualSalary, E.HourlyRate
FROM OldEmployeeDataHive E
join DepartmentDataHive D

ON E.Department = D.depart_name;

SELECT * FROM EmployeeDataHive LIMIT 20;

DROP TABLE IF EXISTS OldEmployeeDataHive;
```

Task 3.2

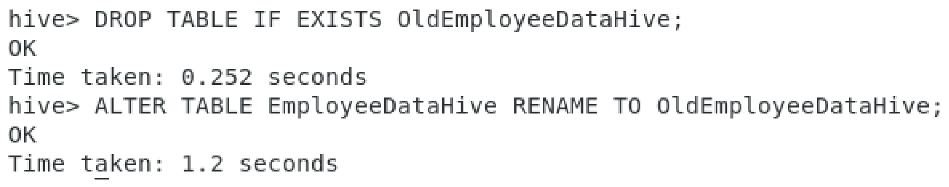
Goal: Also update the employee-data-hive table 'annual salary' field based on the 'Typical Hours' * 'Hourly Rate' * 52 if the annual salary field is empty.

this task we have 2 solutions:-

1. Using **UPDATE clause.**
2. Using **CASE WHEN THEN** clause.

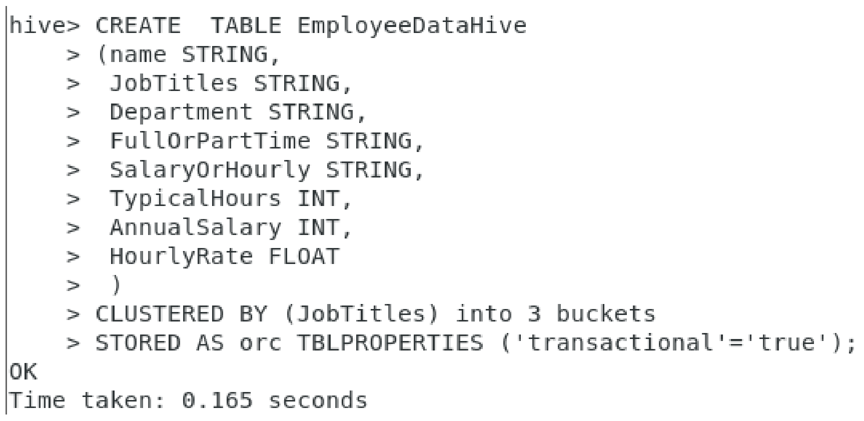
First Solution

1- Since ACID operations only is performed on **ORC** tables, so I renamed the **EmployeeDataHive** table to **OldEmployeeDataHive** to create a new empty ORC table.



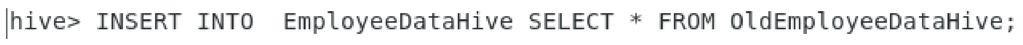
```
hive> DROP TABLE IF EXISTS OldEmployeeDataHive;
OK
Time taken: 0.252 seconds
hive> ALTER TABLE EmployeeDataHive RENAME TO OldEmployeeDataHive;
OK
Time taken: 1.2 seconds
```

2- Create an **ORC** table called **EmployeeDataHive** **CULSTERD BY JobtTitles** column and bucketed into **3** buckets (**3 reducers will be used to perform an ACID operations**).



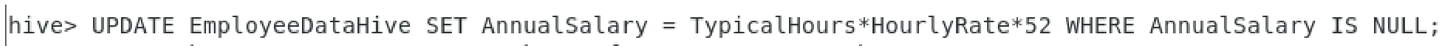
```
hive> CREATE  TABLE EmployeeDataHive
    > (name STRING,
    >  JobTitles STRING,
    >  Department STRING,
    >  FullOrPartTime STRING,
    >  SalaryOrHourly STRING,
    >  TypicalHours INT,
    >  AnnualSalary INT,
    >  HourlyRate FLOAT
    >  )
    > CLUSTERED BY (JobTitles) into 3 buckets
    > STORED AS orc TBLPROPERTIES ('transactional'='true');
OK
Time taken: 0.165 seconds
```

3- **INSERT** data into an ORC (**EmployeeDataHive**) table from **OldEmployeeDataHive.**



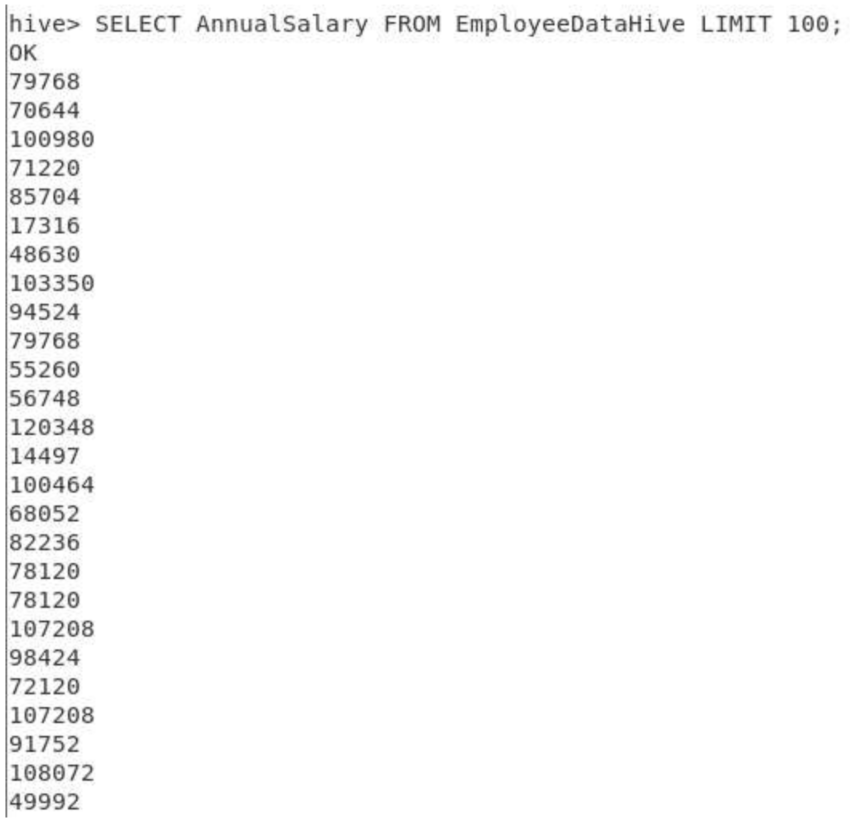
```
hive> INSERT INTO  EmployeeDataHive SELECT * FROM OldEmployeeDataHive;
```

4- Update **Annualsalary** column if its record is **NULL** by multiplying the value of **TypicalHours** by **HourlyRate** * 52 and it's not **NULL** keep it as it is.



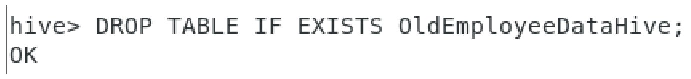
```
hive> UPDATE EmployeeDataHive SET AnnualSalary = TypicalHours*HourlyRate*52 WHERE AnnualSalary IS NULL;
```

5- Verify that **Annualsalary** is updated in correct way.



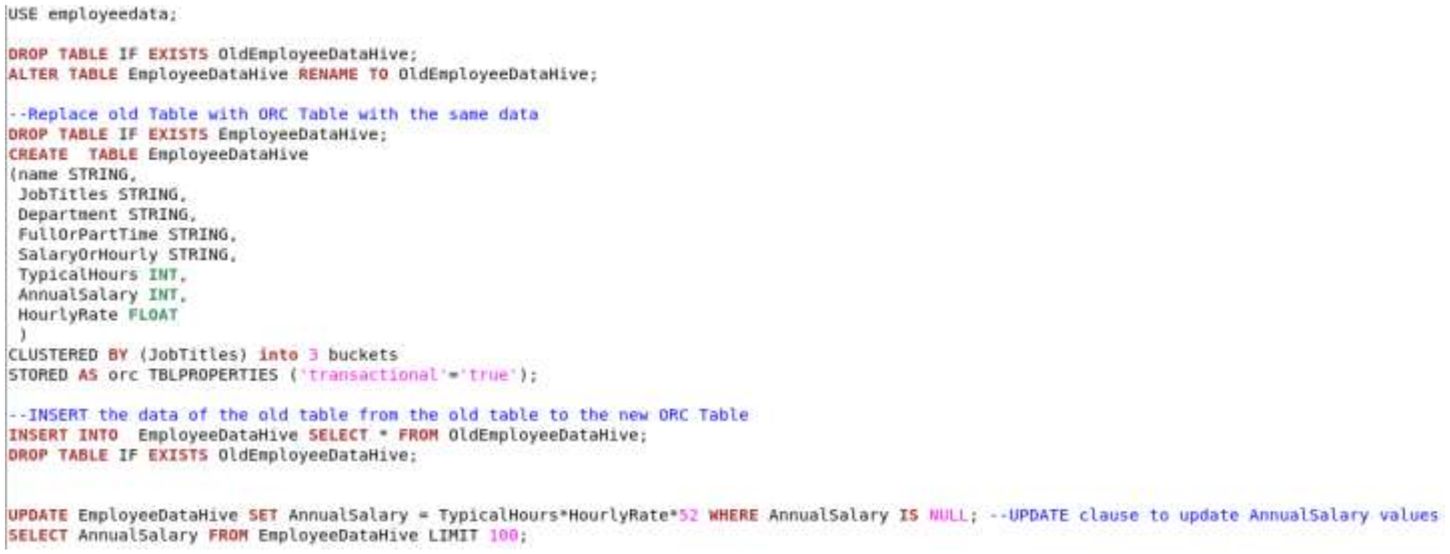
```
hive> SELECT AnnualSalary FROM EmployeeDataHive LIMIT 100;
OK
79768
70644
100980
71220
85704
17316
48630
103350
94524
79768
55260
56748
120348
14497
100464
68052
82236
78120
78120
107208
98424
72120
107208
91752
108072
49992
```

6- Drop **OldEmployeeDataHive** because we don’t need it or its data anymore.



```
hive> DROP TABLE IF EXISTS OldEmployeeDataHive;
OK
```

The Final script for task 3.2.1

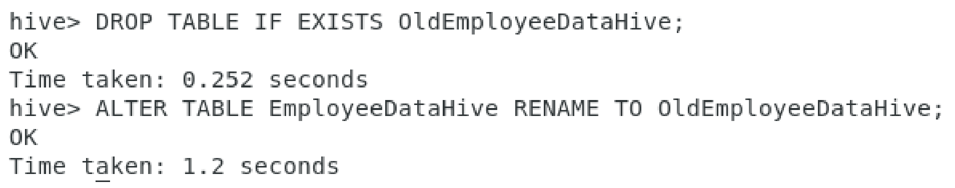


```
USE employeedata;

DROP TABLE IF EXISTS OldEmployeeDataHive;
ALTER TABLE EmployeeDataHive RENAME TO OldEmployeeDataHive;

--Replace old Table with ORC Table with the same data
DROP TABLE IF EXISTS EmployeeDataHive;
CREATE  TABLE EmployeeDataHive
(name STRING,
 JobTitles STRING,
 Department STRING,
 FullOrPartTime STRING,
 SalaryOrHourly STRING,
 TypicalHours INT,
 AnnualSalary INT,
 HourlyRate FLOAT
 )
CLUSTERED BY (JobTitles) into 3 buckets
STORED AS orc TBLPROPERTIES ('transactional'='true');

--INSERT the data of the old table from the old table to the new ORC Table
INSERT INTO  EmployeeDataHive SELECT * FROM OldEmployeeDataHive;
DROP TABLE IF EXISTS OldEmployeeDataHive;


UPDATE EmployeeDataHive SET AnnualSalary = TypicalHours*HourlyRate*52 WHERE AnnualSalary IS NULL; --UPDATE clause to update AnnualSalary values
SELECT AnnualSalary FROM EmployeeDataHive LIMIT 100;
```

==

Second Solution

1- We **renamed** the **EmployeeDataHive** table to **OldEmployeeDataHive** to use the **EmployeeDataHive** table to store the new update.

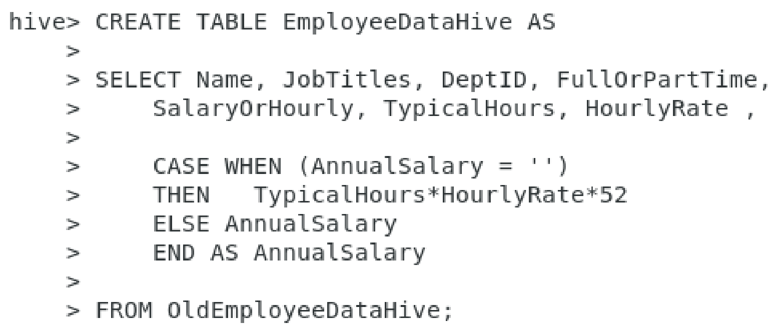


```
hive> DROP TABLE IF EXISTS OldEmployeeDataHive;
OK
Time taken: 0.252 seconds
hive> ALTER TABLE EmployeeDataHive RENAME TO OldEmployeeDataHive;
OK
Time taken: 1.2 seconds
```

Note: Drop operation here just to not give us an error if **OldEmployeeDataHive** if it's already existing.

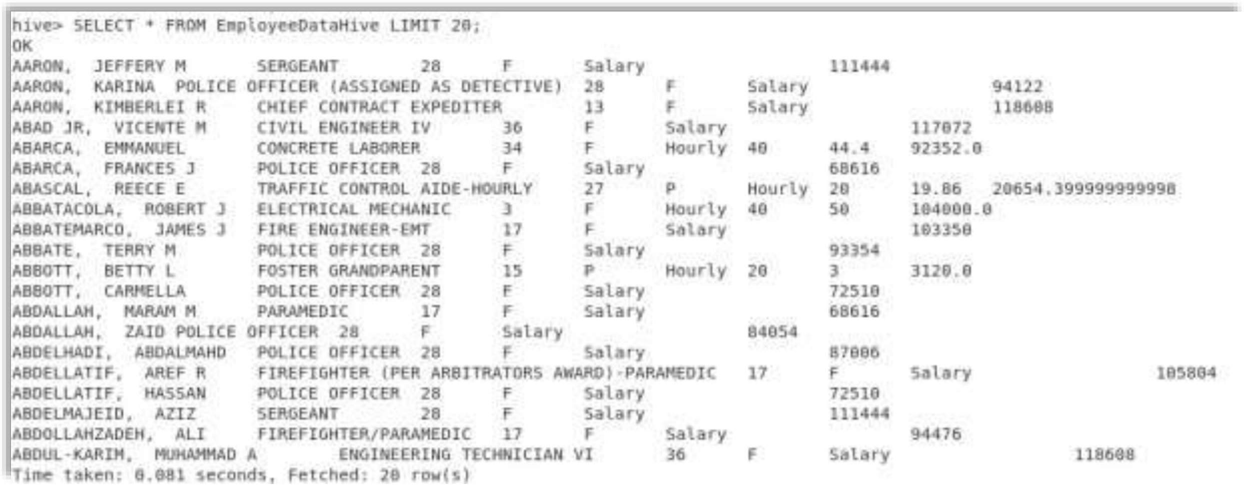
2- we used **(CREATE TABLE AS SELECT)** to update **Annualsalary** from table **EmployeeDataHive** based on the **TypicalHours** * **HourlyRate** * 52 if the annual salary field is **NULL**.

3- We used **CASE WHEN THEN** clause to check the condition of **Annualsalary** **NULL** or not If **NULL** it will apply the following calculation, else it will be the same value.



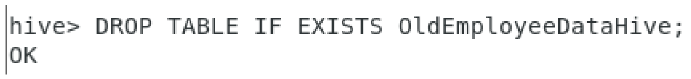
```
hive> CREATE TABLE EmployeeDataHive AS
    >
    > SELECT Name, JobTitles, DeptID, FullOrPartTime,
    >     SalaryOrHourly, TypicalHours, HourlyRate ,
    >
    >     CASE WHEN (AnnualSalary = '')
    >     THEN   TypicalHours*HourlyRate*52
    >     ELSE AnnualSalary
    >     END AS AnnualSalary
    >
    > FROM OldEmployeeDataHive;
```

4- Then we showed our data after updating the **Annualsalary** values.



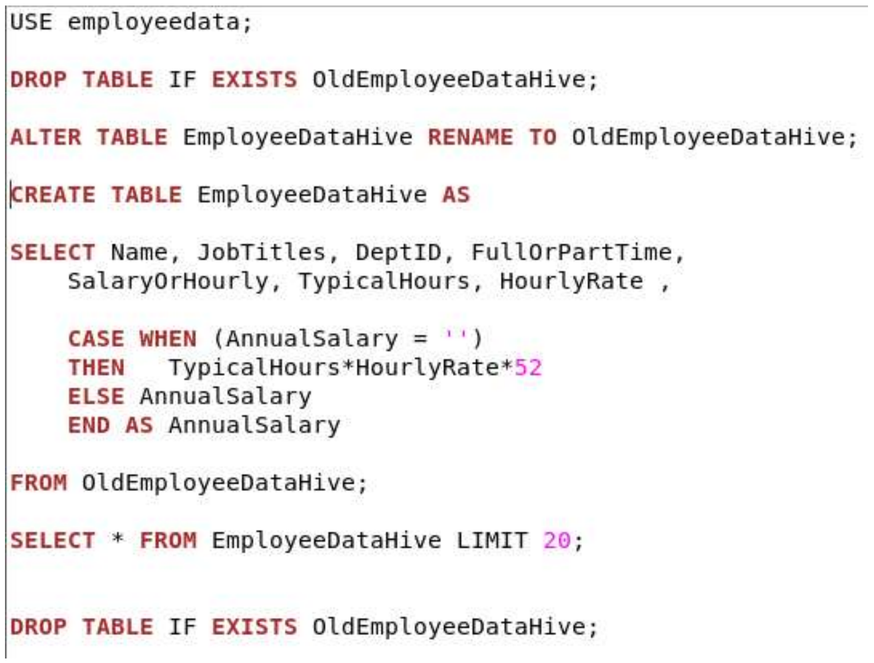
```
hive> SELECT * FROM EmployeeDataHive LIMIT 20;
OK
AARON,  JEFFERY M       SERGEANT        28      F       Salary                  111444
AARON,  KARINA  POLICE OFFICER (ASSIGNED AS DETECTIVE)  28      F       Salary                  94122
AARON,  KIMBERLEI R     CHIEF CONTRACT EXPEDITER        13      F       Salary                  118608
ABAD JR,  VICENTE M     CIVIL ENGINEER IV       36      F       Salary                  117072
ABARCA,  EMMANUEL       CONCRETE LABORER        34      F       Hourly  40      44.4    92352.0
ABARCA,  FRANCES J      POLICE OFFICER  28      F       Salary                  68616
ABASCAL,  REECE E       TRAFFIC CONTROL AIDE-HOURLY     27      P       Hourly  20      19.86   20654.399999999998
ABBATACOLA,  ROBERT J   ELECTRICAL MECHANIC     3       F       Hourly  40      50      104000.0
ABBATEMARCO,  JAMES J   FIRE ENGINEER-EMT       17      F       Salary                  103350
ABBATE,  TERRY M        POLICE OFFICER  28      F       Salary                  93354
ABBOTT,  BETTY L        FOSTER GRANDPARENT      15      P       Hourly  20      3       3120.0
ABBOTT,  CARMELLA       POLICE OFFICER  28      F       Salary                  72510
ABDALLAH,  MARAM M      PARAMEDIC       17      F       Salary                  68616
ABDALLAH,  ZAID POLICE OFFICER  28      F       Salary                  84054
ABDELHADI,  ABDALMAHD   POLICE OFFICER  28      F       Salary                  87006
ABDELLATIF,  AREF R     FIREFIGHTER (PER ARBITRATORS AWARD)-PARAMEDIC   17      F       Salary                  105804
ABDELLATIF,  HASSAN     POLICE OFFICER  28      F       Salary                  72510
ABDELMAJEID,  AZIZ      SERGEANT        28      F       Salary                  111444
ABDOLLAHZADEH,  ALI     FIREFIGHTER/PARAMEDIC   17      F       Salary                  94476
ABDUL-KARIM,  MUHAMMAD A        ENGINEERING TECHNICIAN VI       36      F       Salary                  118608
Time taken: 0.081 seconds, Fetched: 20 row(s)
```

6- Drop **OldEmployeeDataHive** because we don't need it or its data anymore.



```
hive> DROP TABLE IF EXISTS OldEmployeeDataHive;
OK
```

The Final script for task 3.2.2



```
USE employeedata;

DROP TABLE IF EXISTS OldEmployeeDataHive;

ALTER TABLE EmployeeDataHive RENAME TO OldEmployeeDataHive;

CREATE TABLE EmployeeDataHive AS

SELECT Name, JobTitles, DeptID, FullOrPartTime,
    SalaryOrHourly, TypicalHours, HourlyRate ,

    CASE WHEN (AnnualSalary = '')
    THEN   TypicalHours*HourlyRate*52
    ELSE AnnualSalary
    END AS AnnualSalary

FROM OldEmployeeDataHive;

SELECT * FROM EmployeeDataHive LIMIT 20;


DROP TABLE IF EXISTS OldEmployeeDataHive;
```

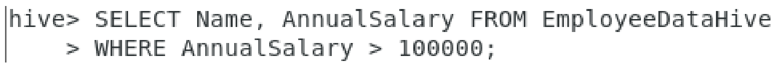
Task 4

Task 4.1

Goal: Display all employees list with salary more than $100,000 based on employee-data-hive table.

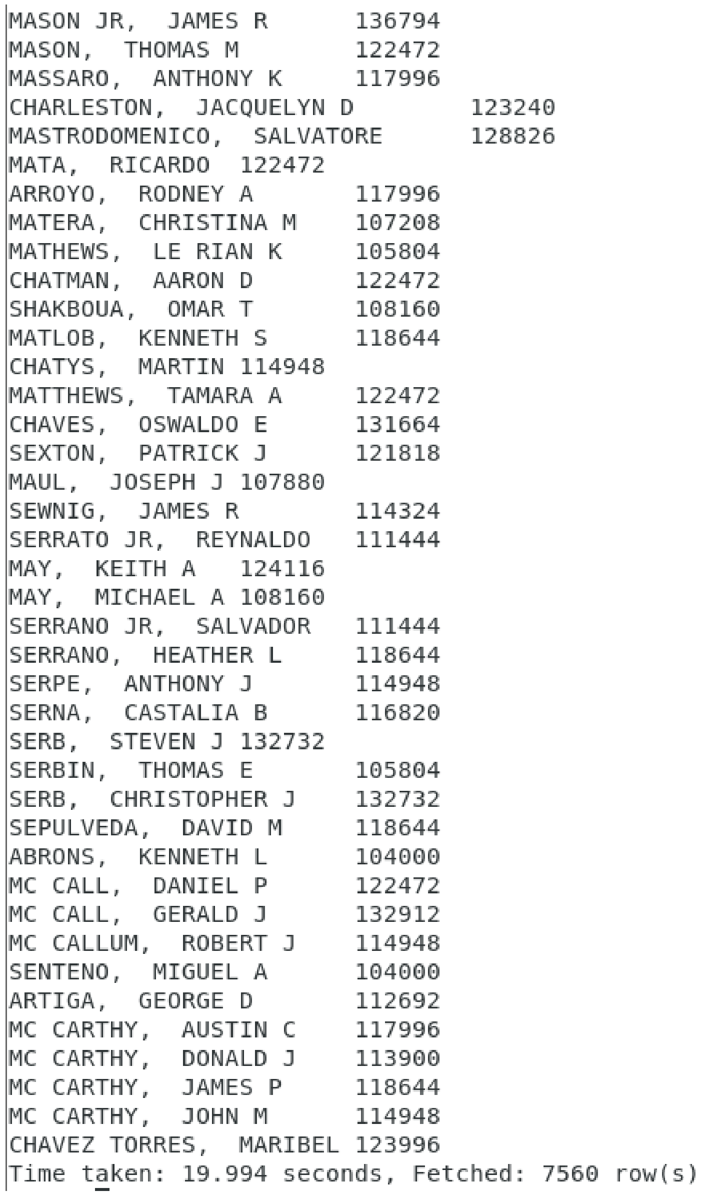
Solution

1- We selected the name of employees and their annual salaries who have **Annualsalary** greater than 100000 by **SELECT** command and **WHERE** clause to set a condition.



```
hive> SELECT Name, AnnualSalary FROM EmployeeDataHive
    > WHERE AnnualSalary > 100000;
```

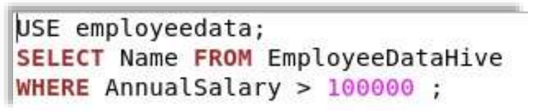
2- This is the sample of result because the data is large



```
MASON JR,  JAMES R      136794
MASON,  THOMAS M        122472
MASSARO,  ANTHONY K     117996
CHARLESTON,  JACQUELYN D        123240
MASTRODOMENICO,  SALVATORE      128826
MATA,  RICARDO  122472
ARROYO,  RODNEY A       117996
MATERA,  CHRISTINA M    107208
MATHEWS,  LE RIAN K     105804
CHATMAN,  AARON D       122472
SHAKBOUA,  OMAR T       108160
MATLOB,  KENNETH S      118644
CHATYS,  MARTIN 114948
MATTHEWS,  TAMARA A     122472
CHAVES,  OSWALDO E      131664
SEXTON,  PATRICK J      121818
MAUL,  JOSEPH J 107880
SEWNIG,  JAMES R        114324
SERRATO JR,  REYNALDO   111444
MAY,  KEITH A   124116
MAY,  MICHAEL A 108160
SERRANO JR,  SALVADOR   111444
SERRANO,  HEATHER L     118644
SERPE,  ANTHONY J       114948
SERNA,  CASTALIA B      116820
SERB,  STEVEN J 132732
SERBIN,  THOMAS E       105804
SERB,  CHRISTOPHER J    132732
SEPULVEDA,  DAVID M     118644
ABRONS,  KENNETH L      104000
MC CALL,  DANIEL P      122472
MC CALL,  GERALD J      132912
MC CALLUM,  ROBERT J    114948
SENTENO,  MIGUEL A      104000
ARTIGA,  GEORGE D       112692
MC CARTHY,  AUSTIN C    117996
MC CARTHY,  DONALD J    113900
MC CARTHY,  JAMES P     118644
MC CARTHY,  JOHN M      114948
CHAVEZ TORRES,  MARIBEL 123996
Time taken: 19.994 seconds, Fetched: 7560 row(s)
```

We got **7561** employees who have **Annualsalary** more than **100000,** and we selected **Annualsalary** to ensure that **WHERE** clause is worked correctly.

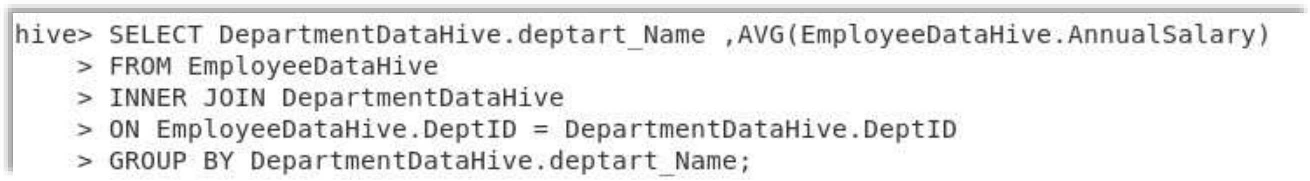
The Final script for task 4.1


```
USE employeedata;
SELECT Name FROM EmployeeDataHive
WHERE AnnualSalary > 100000 ;
```

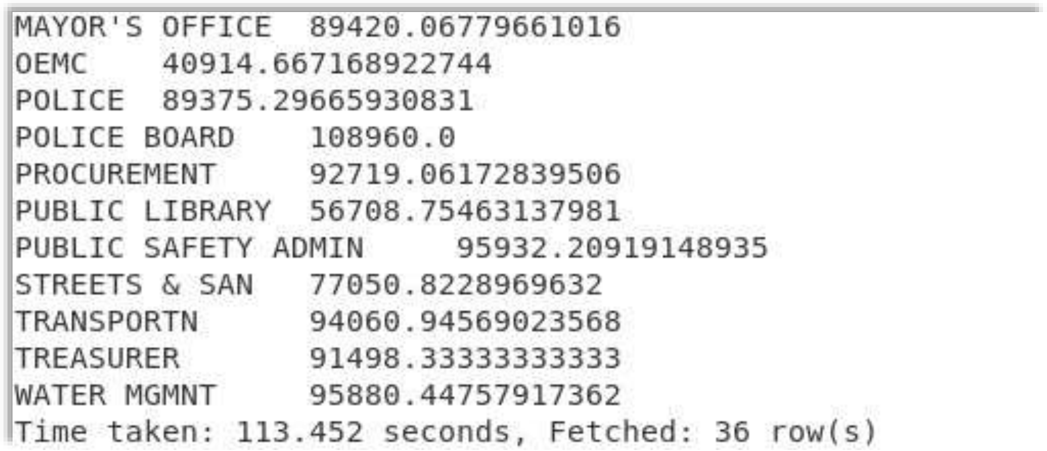
Task 4.2

Goal: join the employee-data-hive and department-data-hive table to show the average salary of employees by department name.

1- To get the **average salary** of employees by name of department, we will join two tables by INNER **JOIN** command then group by department name and we finally select that column (**depart_name and average salary**)

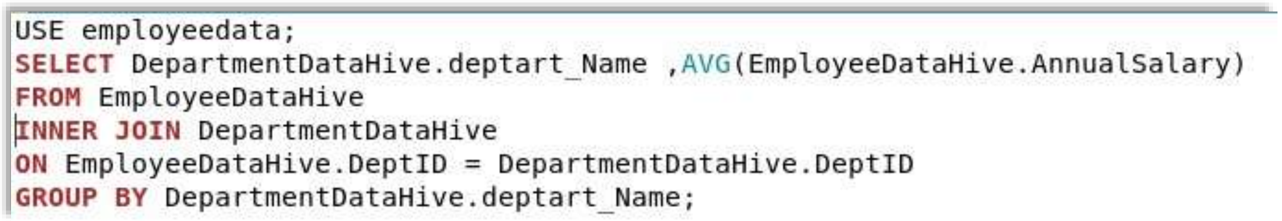

```
hive> SELECT DepartmentDataHive.deptart_Name ,AVG(EmployeeDataHive.AnnualSalary)
    > FROM EmployeeDataHive
    > INNER JOIN DepartmentDataHive
    > ON EmployeeDataHive.DeptID = DepartmentDataHive.DeptID
    > GROUP BY DepartmentDataHive.deptart_Name;
```

2- This sample of result


```
MAYOR'S OFFICE  89420.06779661016
OEMC    40914.667168922744
POLICE  89375.29665930831
POLICE BOARD    108960.0
PROCUREMENT     92719.06172839506
PUBLIC LIBRARY  56708.75463137981
PUBLIC SAFETY ADMIN     95932.20919148935
STREETS & SAN   77050.8228969632
TRANSPORTN      94060.94569023568
TREASURER       91498.33333333333
WATER MGMNT     95880.44757917362
Time taken: 113.452 seconds, Fetched: 36 row(s)
```

We got **36 departments** with average salaries.

The Final script for task 4.2

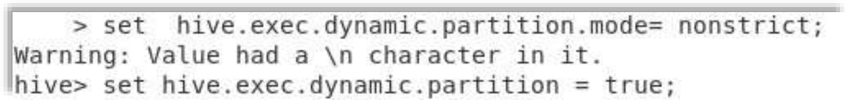


```
USE employeedata;
SELECT DepartmentDataHive.deptart_Name ,AVG(EmployeeDataHive.AnnualSalary)
FROM EmployeeDataHive
INNER JOIN DepartmentDataHive
ON EmployeeDataHive.DeptID = DepartmentDataHive.DeptID
GROUP BY DepartmentDataHive.deptart_Name;
```

Task 5

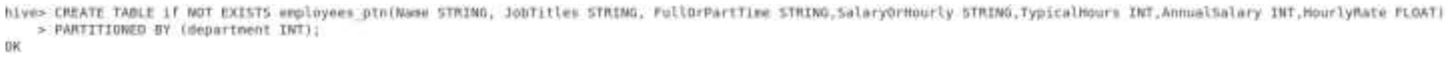
Goal: Create **5** partitions in **employees_ptn** table to store **5 departments** in the appropriate partition. Display the partition structure.

1- Edit dynamic partitioning in hive.



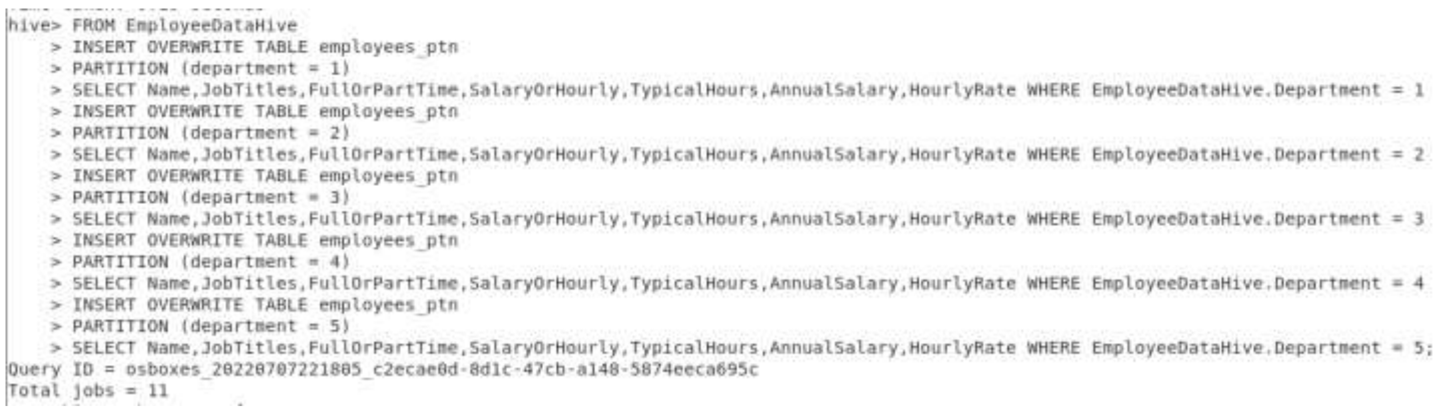
```
    > set  hive.exec.dynamic.partition.mode= nonstrict;
Warning: Value had a \n character in it.
hive> set hive.exec.dynamic.partition = true;
```

2- We created table **(employees_ptn)** to store **5** appropriate partitions and we partitioned by **department** column as string.



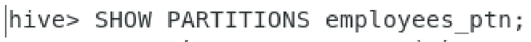
```
hive> CREATE TABLE if NOT EXISTS employees_ptn(Name STRING, JobTitles STRING, FullOrPartTime STRING,SalaryOrHourly STRING,TypicalHours INT,AnnualSalary INT,HourlyRate FLOAT)
    > PARTITIONED BY (department INT);
OK
```

3- Now we'll insert the appropriate data to these partitions from **EmployeeDataHive,** we will have **5** **WHERE** clauses each **WHERE** clause will extract the required data from **EmployeeDataHive** to each **partition**.

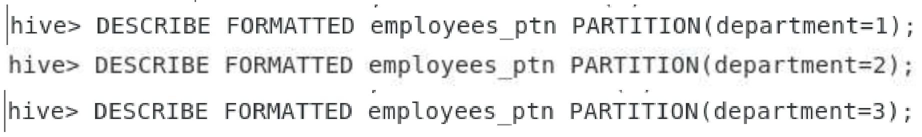


```
hive> FROM EmployeeDataHive
    > INSERT OVERWRITE TABLE employees_ptn
    > PARTITION (department = 1)
    > SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 1
    > INSERT OVERWRITE TABLE employees_ptn
    > PARTITION (department = 2)
    > SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 2
    > INSERT OVERWRITE TABLE employees_ptn
    > PARTITION (department = 3)
    > SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 3
    > INSERT OVERWRITE TABLE employees_ptn
    > PARTITION (department = 4)
    > SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 4
    > INSERT OVERWRITE TABLE employees_ptn
    > PARTITION (department = 5)
    > SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 5;
Query ID = osboxes_20220707221805_c2ecae0d-8d1c-47cb-a148-5874eeca695c
Total jobs = 11
```

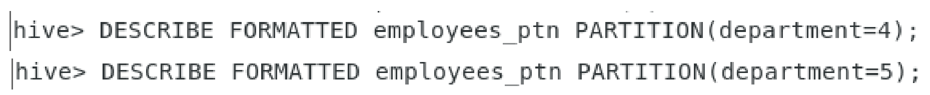
4- Display every partition structure by using DESCRIBE FORMATTED employees_ptn PARTITION(department= partition_number);



```
hive> SHOW PARTITIONS employees_ptn;
```



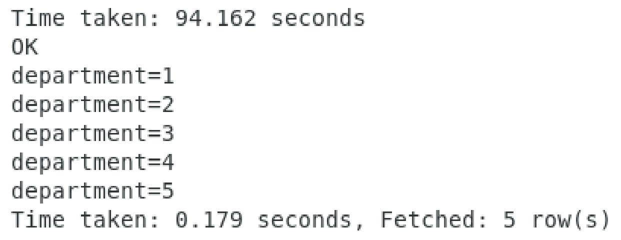
```
hive> DESCRIBE FORMATTED employees_ptn PARTITION(department=1);
hive> DESCRIBE FORMATTED employees_ptn PARTITION(department=2);
hive> DESCRIBE FORMATTED employees_ptn PARTITION(department=3);
```



```
hive> DESCRIBE FORMATTED employees_ptn PARTITION(department=4);
hive> DESCRIBE FORMATTED employees_ptn PARTITION(department=5);
```

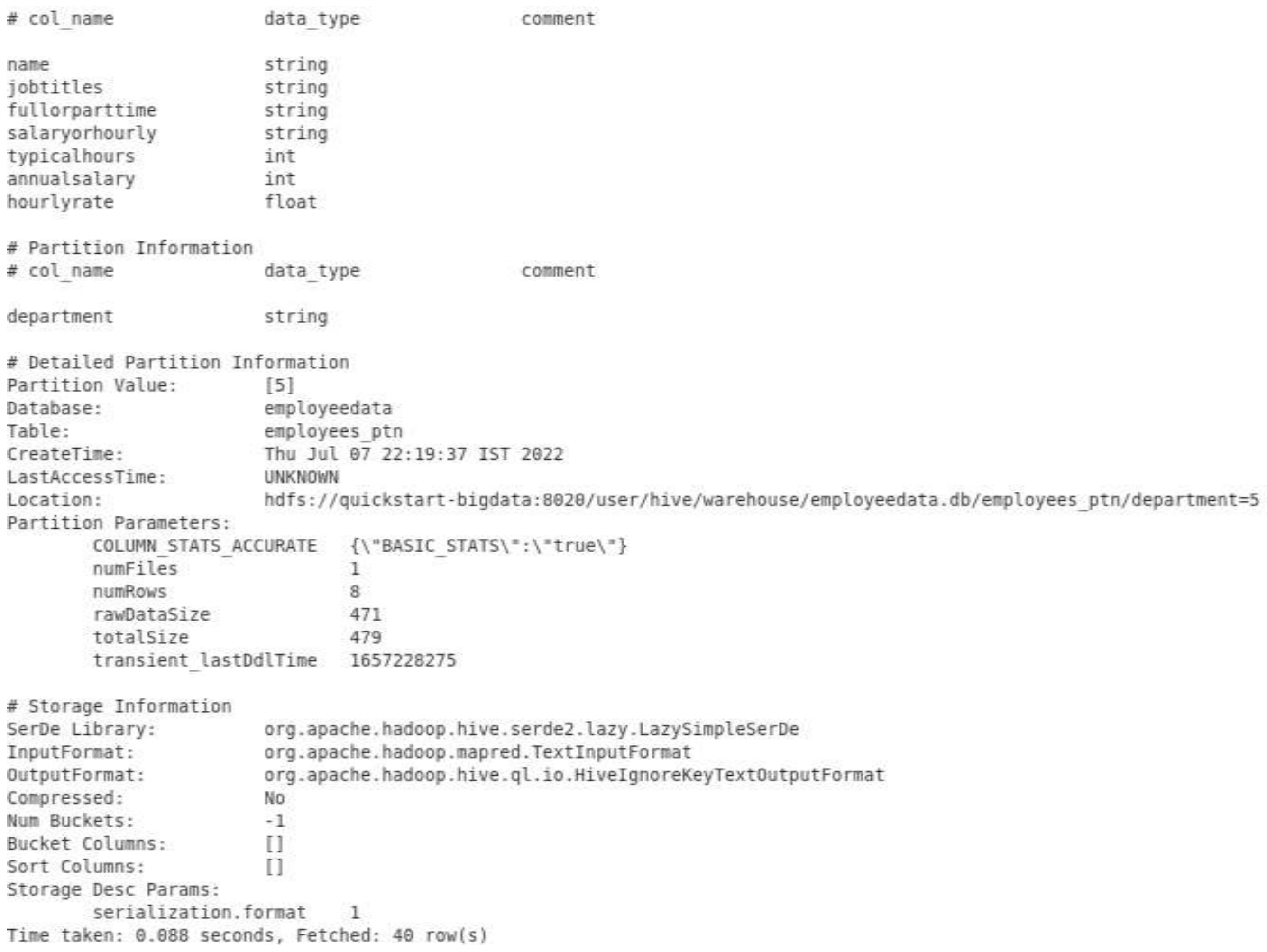
These are results samples.

All partitions in the table.



```
Time taken: 94.162 seconds
OK
department=1
department=2
department=3
department=4
department=5
Time taken: 0.179 seconds, Fetched: 5 row(s)
```

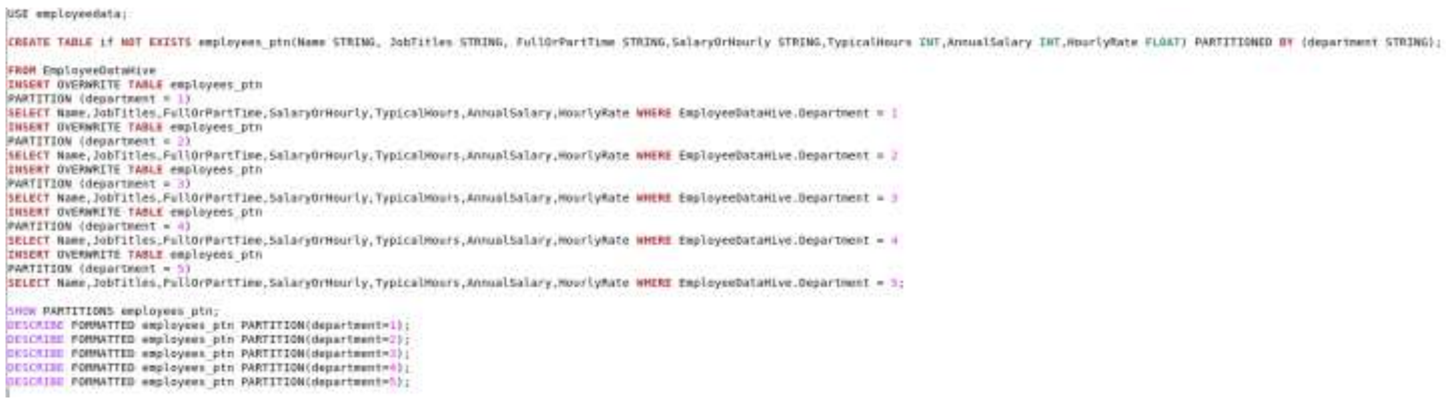
Partition department = 1 sturcture.



```
# col_name              data_type               comment

name                    string
jobtitles               string
fullorparttime          string
salaryorhourly          string
typicalhours            int
annualsalary            int
hourlyrate              float

# Partition Information
# col_name              data_type               comment

department              string

# Detailed Partition Information
Partition Value:        [5]
Database:               employeedata
Table:                  employees_ptn
CreateTime:             Thu Jul 07 22:19:37 IST 2022
LastAccessTime:         UNKNOWN
Location:               hdfs://quickstart-bigdata:8020/user/hive/warehouse/employeedata.db/employees_ptn/department=5
Partition Parameters:
        COLUMN_STATS_ACCURATE   {\"BASIC_STATS\":\"true\"}
        numFiles                1
        numRows                 8
        rawDataSize             471
        totalSize               479
        transient_lastDdlTime   1657228275

# Storage Information
SerDe Library:          org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe
InputFormat:            org.apache.hadoop.mapred.TextInputFormat
OutputFormat:           org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat
Compressed:             No
Num Buckets:            -1
Bucket Columns:         []
Sort Columns:           []
Storage Desc Params:
        serialization.format    1
Time taken: 0.088 seconds, Fetched: 40 row(s)
```

The above query will print another 4 results like above images but for other partitions.

The Final script for task 5



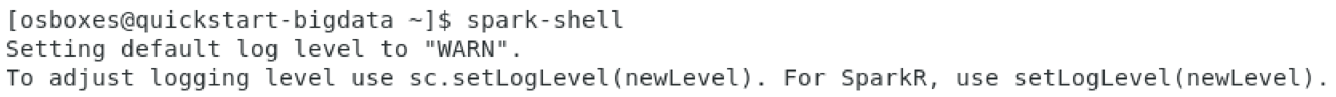
```
USE employeedata;

CREATE TABLE if NOT EXISTS employees_ptn(Name STRING, JobTitles STRING, FullOrPartTime STRING,SalaryOrHourly STRING,TypicalHours INT,AnnualSalary INT,HourlyRate FLOAT) PARTITIONED BY (department STRING);

FROM EmployeeDataHive
INSERT OVERWRITE TABLE employees_ptn
PARTITION (department = 1)
SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 1
INSERT OVERWRITE TABLE employees_ptn
PARTITION (department = 2)
SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 2
INSERT OVERWRITE TABLE employees_ptn
PARTITION (department = 3)
SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 3
INSERT OVERWRITE TABLE employees_ptn
PARTITION (department = 4)
SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 4
INSERT OVERWRITE TABLE employees_ptn
PARTITION (department = 5)
SELECT Name,JobTitles,FullOrPartTime,SalaryOrHourly,TypicalHours,AnnualSalary,HourlyRate WHERE EmployeeDataHive.Department = 5;

SHOW PARTITIONS employees_ptn;
DESCRIBE FORMATTED employees_ptn PARTITION(department=1);
DESCRIBE FORMATTED employees_ptn PARTITION(department=2);
DESCRIBE FORMATTED employees_ptn PARTITION(department=3);
DESCRIBE FORMATTED employees_ptn PARTITION(department=4);
DESCRIBE FORMATTED employees_ptn PARTITION(department=5);
```

Task 6

Goal: Create spark DataFrame based on the given dataset. Identify # of records in the DataFrame and show top 10 records.

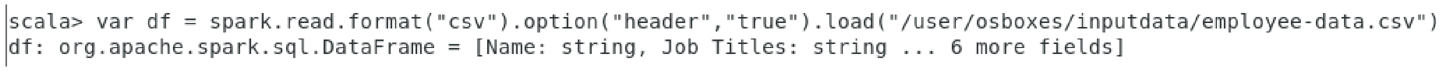
Solution

1- Open spark-shell to start write commands.



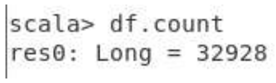
```
[osboxes@quickstart-bigdata ~]$ spark-shell
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
```

2- Create required **DataFrame** with header argument true to consider the first row of the data as header and load the data from csv file which is located into hdfs.



```
scala> var df = spark.read.format("csv").option("header","true").load("/user/osboxes/inputdata/employee-data.csv")
df: org.apache.spark.sql.DataFrame = [Name: string, Job Titles: string ... 6 more fields]
```

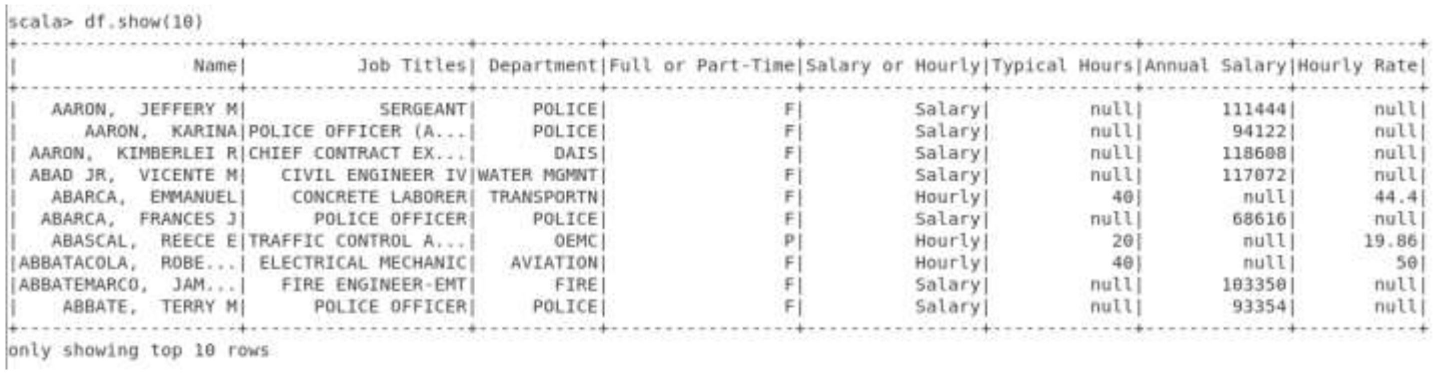
3- Start counting the number of rows of the csv file.



```
scala> df.count
res0: Long = 32928
```

We got **32928 employees** (header doesn't included).

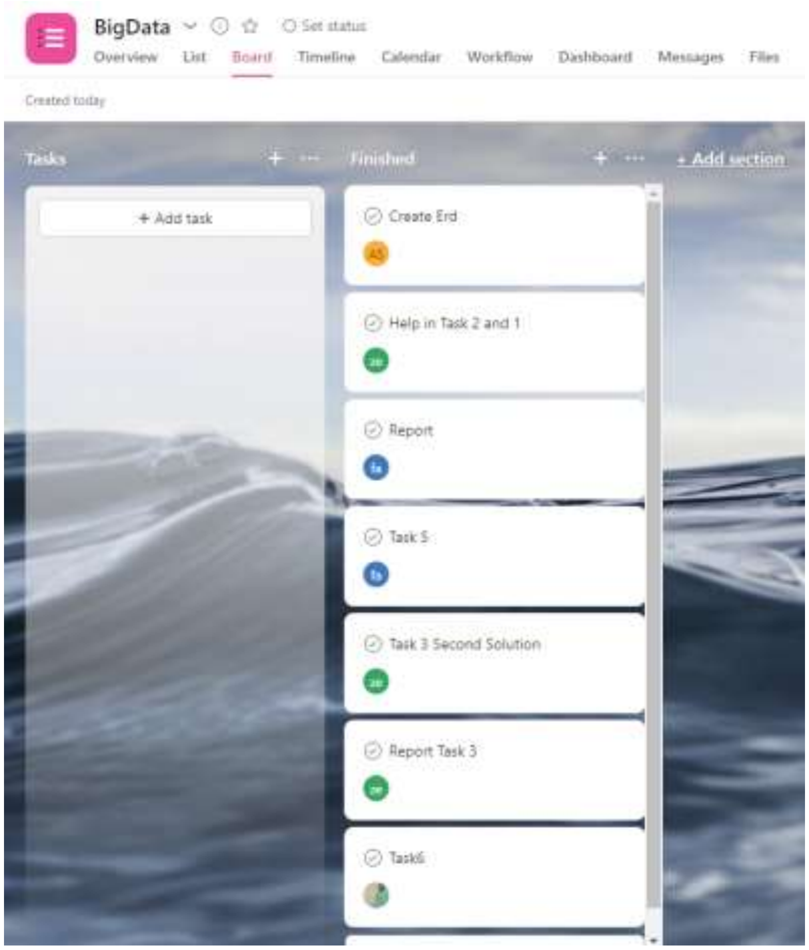
4- We displayed the top 10 records.



```
scala> df.show(10)
+--------------------+--------------------+------------+-----------------+----------------+-------------+-------------+-----------+
|                Name|          Job Titles|  Department|Full or Part-Time|Salary or Hourly|Typical Hours|Annual Salary|Hourly Rate|
+--------------------+--------------------+------------+-----------------+----------------+-------------+-------------+-----------+
|   AARON,  JEFFERY M|            SERGEANT|      POLICE|                F|          Salary|         null|       111444|       null|
|      AARON,  KARINA|POLICE OFFICER (A...|      POLICE|                F|          Salary|         null|        94122|       null|
| AARON,  KIMBERLEI R|CHIEF CONTRACT EX...|        DAIS|                F|          Salary|         null|       118608|       null|
| ABAD JR,  VICENTE M| CIVIL ENGINEER IV|WATER MGMNT|                F|          Salary|         null|       117072|       null|
|   ABARCA,  EMMANUEL|    CONCRETE LABORER| TRANSPORTN|                F|          Hourly|           40|         null|       44.4|
|  ABARCA,  FRANCES J|      POLICE OFFICER|      POLICE|                F|          Salary|         null|        68616|       null|
|   ABASCAL,  REECE E|TRAFFIC CONTROL A...|        OEMC|                P|          Hourly|           20|         null|      19.86|
|ABBATACOLA,  ROBE...| ELECTRICAL MECHANIC|    AVIATION|                F|          Hourly|           40|         null|         50|
|ABBATEMARCO,  JAM...|   FIRE ENGINEER-EMT|        FIRE|                F|          Salary|         null|       103350|       null|
|    ABBATE,  TERRY M|      POLICE OFFICER|      POLICE|                F|          Salary|         null|        93354|       null|
+--------------------+--------------------+------------+-----------------+----------------+-------------+-------------+-----------+
only showing top 10 rows
```

Workload

Every member in the team share in the project in equal efforts, managed by **Asana software** to deliver and handling their works.



1 - Ahmed Elsayed Salama

- Create the **ERD** Diagram of the project.
- Modify in **EXCEL** sheet.
- **Reported** and **reviewed** the project.
- Worked on **Task2 (Part2), Task 3, Task4 (Part2)**, **Task6**.

2 - Fatma Eldesouky

- She exerts more effort in the **report**.
- Worked on **Task 5 and Task 4** part 1.
- Was responsible for getting resources for our task along with **Mahmoud**.

3 - Mahmoud Khorshed

- He **shared** in all the project in each part.
- **Worked** on all tasks**.**
- **Was responsible for getting resources for our task along with Fatma.**

4 - Zeyad Tarek Mohamed

- He worked on **Task 3** the first solution.
- He worked on **Task 1** by finding smart solution, **Task6**, **Task2.**
- **Reported** and **reviewed** the project**.**

References

1- https://hive.apache.org/

2- https://www.tutorialspoint.com/apache_spark/index.htm

3- https://app.dbdesigner.net/designer/schema/0-project2_erd.

4- https://stackoverflow.com/questions/39032279/hive-update-with-subquery

5- https://stackoverflow.com/questions/18432925/handling-null-values-in-hive

6- https://riptutorial.com/hive/example/15796/orc#:~:text=The%20Optimized%20Row%20Columnar%20

7- https://sparkbyexamples.com/apache-hive/hive-enable-and-use-acid-transactions/

8- https://www.revisitclass.com/hadoop/case-statement-in-hive-with-examples/