**DATA ENGINEERING CAPSTONE PROJECT**

**Business objective**

For big corporation employee’s data from the 1980s, to design data model with all the tables to hold data, import the CSVs into a SQL database, transfer SQL database to HDFS/Hive, and perform analysis using Hive/Impala/Spark/SparkML using the data and create data and ML pipelines. Required to create end to end data pipeline and analyzing the data.

**Contents**

1. Data used and Description
2. Technology stack used
3. ER diagram (data model)
4. Create database & tables in MySQL server as per the above ER Diagram
5. Create Sqoop job to transfer the data from MySQL to HDFS (Data required to store in Parque/Avro/Json format)
6. Create database in Hive as per the above ER Diagram and load the data into Hive tables
7. Work on Exploratory data analysis as per the analysis requirement using Hive/Impala and Spark SQL (expecting to get the data from hive tables).
   1. EDA outputs in hive/impala
   2. EDA outputs in SPARK
8. ML Model: - Classification Model
   1. logistic Regression
   2. Random Forest Classifier
9. Create entire data pipeline and ML pipe line
10. Challenges
11. Way ahead or conclusion

**Data used and Description**

Given tables with their attributes are -

1. Employees(employees.csv)

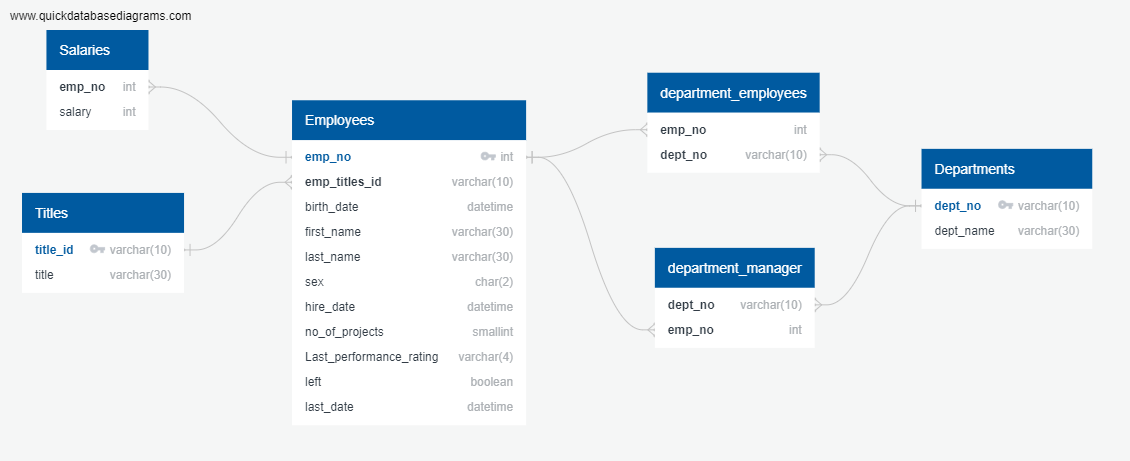
* emp\_no – Employee Id – Integer – Not Null
* emp\_titles\_id – designation id – Not Null
* birth\_date – Date of Birth – Date Time – Not Null
* first\_name – First Name – Character – Not Null
* last\_name – Last Name – Character – Not Null
* sex – Gender – Character – Not Null
* hire\_date – Employee Hire date –Date Time -Not Null
* no\_of\_projects – Number of projects worked on – Integer – Not Null
* Last\_performance\_rating – Last year performance rating – Character – Not Null
* left – Employee left the organization – Boolean – Not Null
* Last\_date - Last date of employment (Exit Date) – Date Time

1. Titles(titles.csv)
   * title\_id – Unique id of type of employee (designation id) – Character – Not Null
   * title – Designation – Character – Not Null
2. Salary(salaries.csv)
   * emp\_no – Employee id – Integer – Not Null
   * Salary – Employee’s Salary – Integer – Not Null
3. Departments(departments.csv)
   * dept\_no - Unique id for each department – character – Not Null
   * dept\_name – Department Name – Character – Not Null
4. Department Managers (dept\_manager.csv)
   * dept\_no - Unique id for each department – character – Not Null
   * emp\_no – Employee number (head of the department ) – Integer – Not Null
5. Department Employees(dept\_emp.csv)
   * emp\_no – Employee id – Integer – Not Null
   * dept\_no - Unique id for each department – character – Not Null

**Technology stack used**

* MySQL (to create database)
* Linux Commands
* Sqoop (Transfer data from MySQL Server to HDFS/Hive)
* HDFS (to store the data)
* Hive (to create database)
* Hive & Impala (to perform the EDA)
* SparkSQL (to perform the EDA)
* SparkML (to perform model building

**ER Diagram (data model)**



**Create database & tables in MySQL server as per the above ER Diagram**

MySQL codes:

1. Login to mysql from shell

mysql -u anabig114225 -pBigdata123

show databases;

use databasename;

2. a) Create tables in mysql manually

CREATE TABLE employees(

emp\_no int not null,

emp\_titles\_id varchar(10) not null,

birth\_date varchar(20) not null,

first\_name varchar(30) not null,

last\_name varchar(30) not null,

sex char(2) not null,

hire\_date varchar(20) not null,

no\_of\_projects smallint not null,

Last\_performance\_rating varchar(4) not null,

left\_company boolean not null,

last\_date varchar(20),

PRIMARY KEY(emp\_no),

CONSTRAINT FK\_title\_id FOREIGN KEY (emp\_titles\_id) REFERENCES titles(title\_id));

CREATE TABLE titles(

title\_id varchar(10) not null,

title varchar(30) not null,

PRIMARY KEY(title\_id),

);

CREATE TABLE salaries(

emp\_no int not null,

salary int not null,

CONSTRAINT FK\_emp\_no FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no) );

CREATE TABLE departments(

dept\_no varchar(10) not null,

dept\_name varchar(30) not null,

PRIMARY KEY(dept\_no) );

CREATE TABLE department\_manager(

dept\_no varchar(10) not null,

emp\_no int not null,

CONSTRAINT FK\_dept\_no FOREIGN KEY (dept\_no) REFERENCES departments(dept\_no),

CONSTRAINT FK\_emp\_no1 FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no) );

CREATE TABLE department\_employees(

emp\_no int not null,

dept\_no varchar(10) not null,

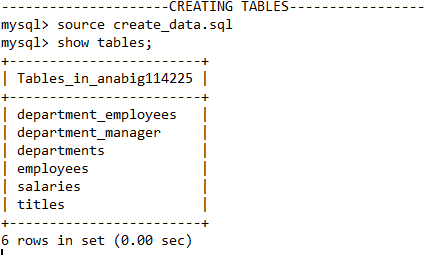
CONSTRAINT FK\_dept\_no2 FOREIGN KEY (emp\_no) REFERENCES employees(emp\_no),

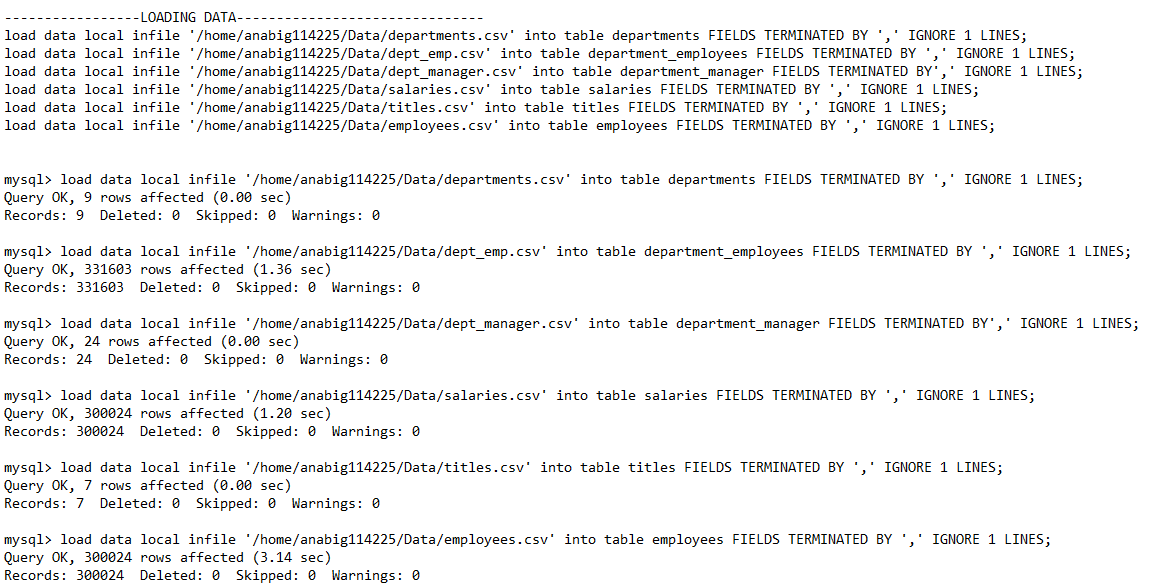
CONSTRAINT FK\_emp\_no2 FOREIGN KEY (dept\_no) REFERENCES departments(dept\_no) );

2.b) . OR create table using the .sql file where all the above create commands written

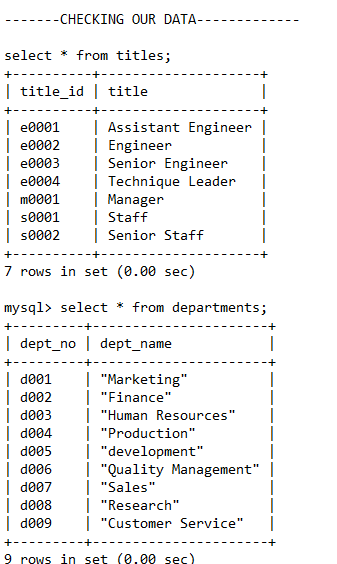
upload create\_tabeles.sql to ftp (https://npbdh.cloudloka.com/ftp

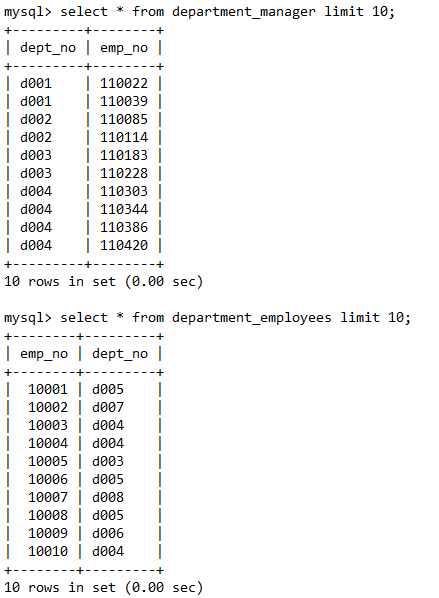
run the below command to create tables under

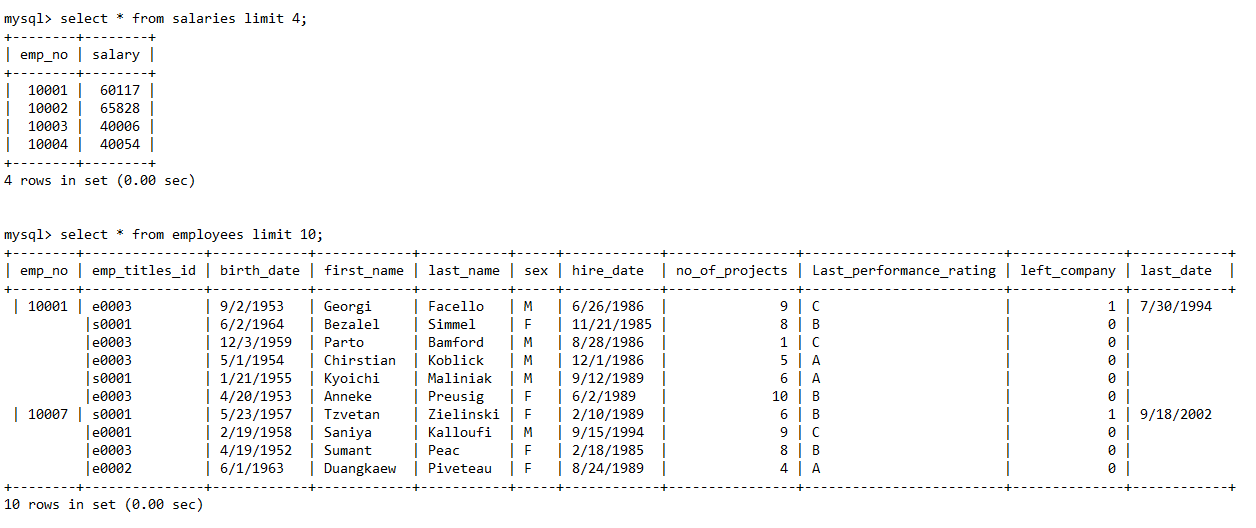
****

3. Loading data  ****

4. Verifing whether our data is properly inserted

****

****

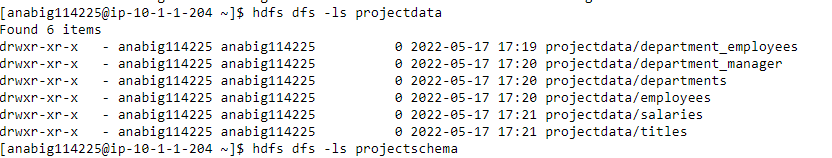
****

**Create Sqoop job to transfer the data from MySQL to HDFS (Data required to store in Parque/Avro/Json format)**

In shell

**Importing the data using sqoop :-** saving in avro format at new directory as projectdata

sqoop import-all-tables --connect jdbc:mysql://ip-10-1-1-204.ap-south-1.compute.internal:3306/anabig114225 --username anabig114225 --password Bigdata123 --compression-codec=snappy --as-avrodatafile --warehouse-dir=/user/anabig114225/projectdata --m 1 --driver com.mysql.jdbc.Driver



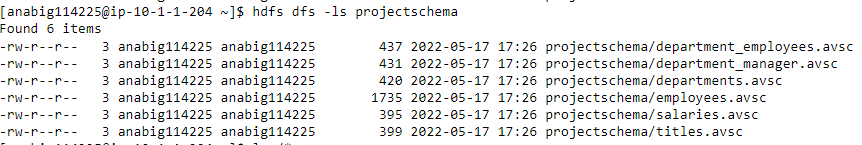
**Locating the schema :-** schema is saved as .avsc format



creating a new directory as projectschema in hdfs where the schema will be saved

hdfs dfs -mkdir projectschema

hdfs dfs -copyFromLocal ~/\*.avsc projectschema

****

Now data has been successfully transferred to HDFS.

**Create database in Hive as per the above ER Diagram and load the data into Hive tables**

create database project\_de;

use project\_de;

CREATE EXTERNAL TABLE employees STORED AS AVRO

LOCATION '/user/anabig114225/projectdata/employees'

TBLPROPERTIES ('avro.schema.url'='/user/anabig114225/projectschema/employees.avsc');

CREATE EXTERNAL TABLE titles STORED AS AVRO

LOCATION '/user/anabig114225/projectdata/titles'

TBLPROPERTIES ('avro.schema.url'='/user/anabig114225/projectschema/titles.avsc');

CREATE EXTERNAL TABLE salaries STORED AS AVRO

LOCATION '/user/anabig114225/projectdata/salaries'

TBLPROPERTIES ('avro.schema.url'='/user/anabig114225/projectschema/salaries.avsc');

CREATE EXTERNAL TABLE departments STORED AS AVRO

LOCATION '/user/anabig114225/projectdata/departments'

TBLPROPERTIES ('avro.schema.url'='/user/anabig114225/projectschema/departments.avsc');

CREATE EXTERNAL TABLE department\_manager STORED AS AVRO

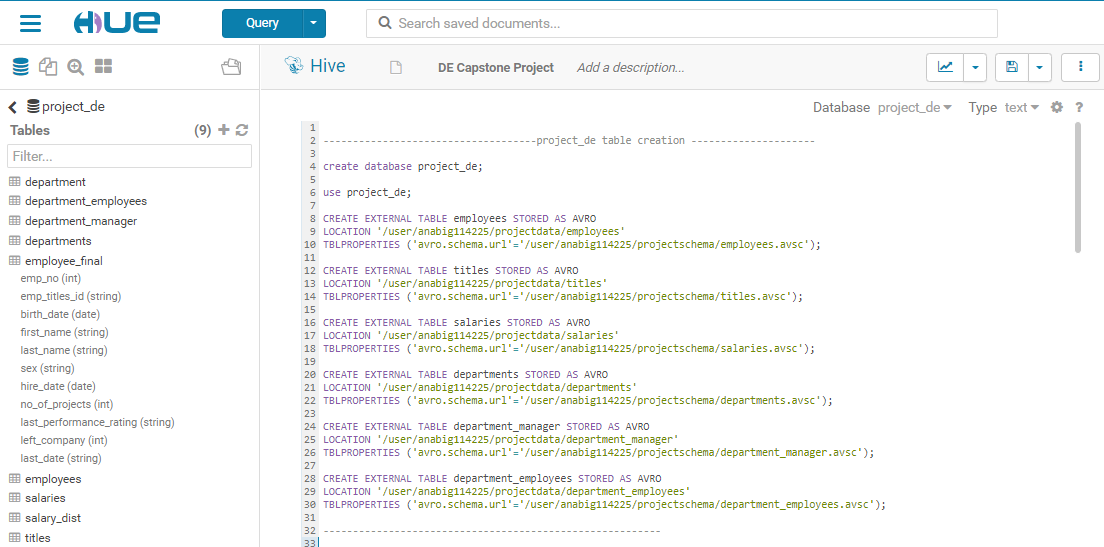
LOCATION '/user/anabig114225/projectdata/department\_manager'

TBLPROPERTIES ('avro.schema.url'='/user/anabig114225/projectschema/department\_manager.avsc');

CREATE EXTERNAL TABLE department\_employees STORED AS AVRO

LOCATION '/user/anabig114225/projectdata/department\_employees'

TBLPROPERTIES ('avro.schema.url'='/user/anabig114225/projectschema/department\_employees.avsc');

****

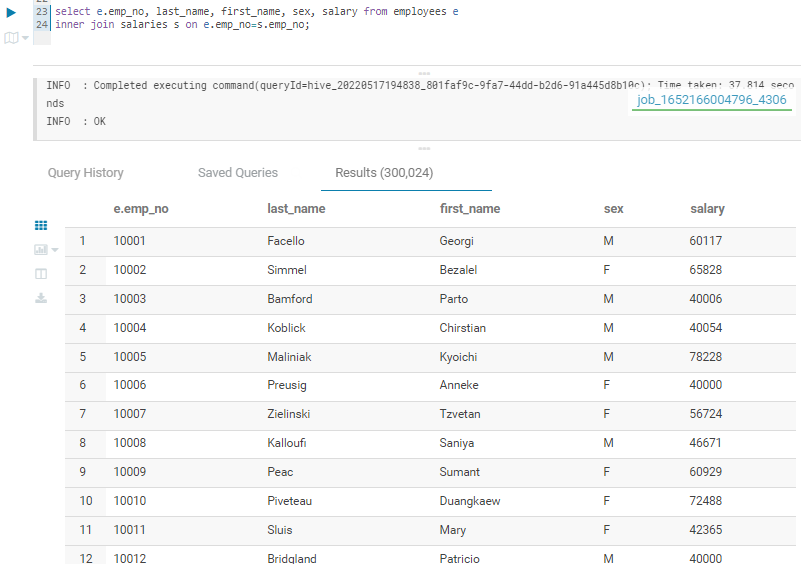
**Work on Exploratory data analysis as per the analysis requirement using Hive/Impala and Spark SQL (expecting to get the data from hive tables).**

**EDA outputs in hive**

**1.** A list showing employee number, last name, first name, sex, and salary for each employee.

**select e.emp\_no, last\_name, first\_name, sex, salary from employees e**

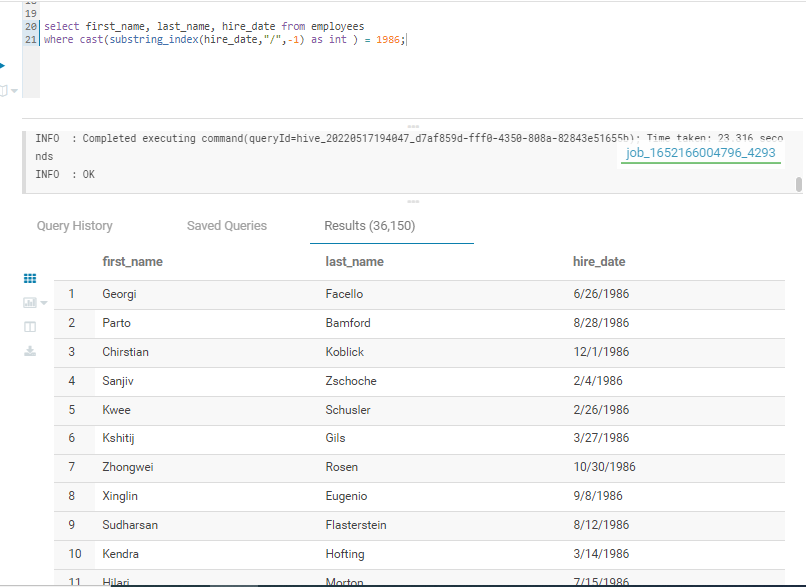
**inner join salaries s on e.emp\_no=s.emp\_no;**



**2.** A list showing first name, last name, and hire date for employees who were hired in 1986.

**select first\_name, last\_name, hire\_date from employees**

**where cast(substring\_index(hire\_date,"/",-1) as int ) = 1986;**

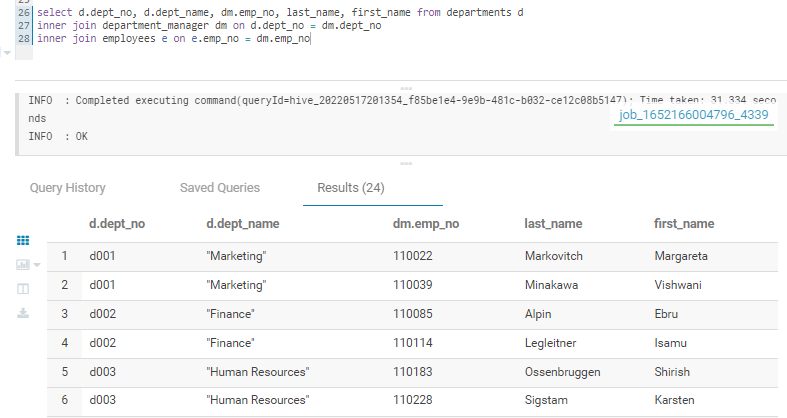


3. A list showing the manager of each department with the following information: department number, department name, the manager's employee number, last name, first name.

**select d.dept\_no, d.dept\_name, dm.emp\_no, last\_name, first\_name from departments d**

**inner join department\_manager dm on d.dept\_no = dm.dept\_no**

**inner join employees e on e.emp\_no = dm.emp\_no**

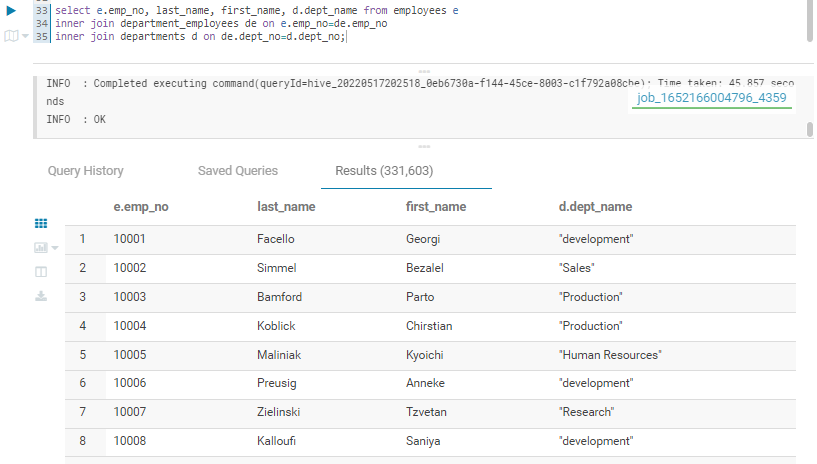


4. A list showing the department of each employee with the following information: employee number, last name, first name, and department name.

**select e.emp\_no, last\_name, first\_name, d.dept\_name from employees e**

**inner join department\_employees de on e.emp\_no=de.emp\_no**

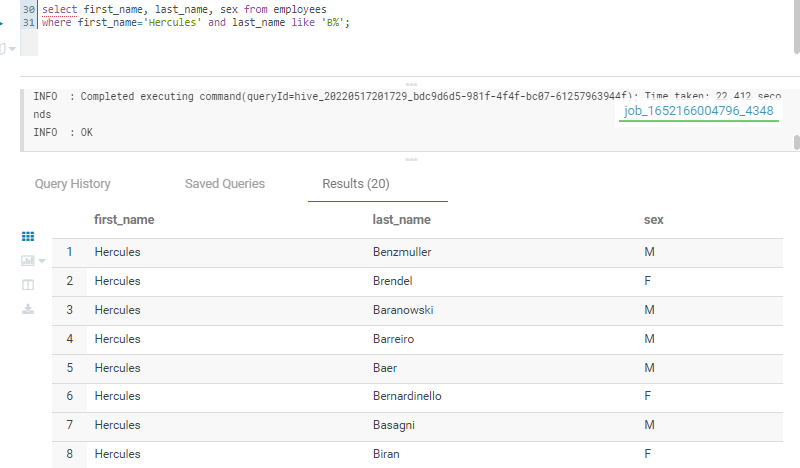
**inner join departments d on de.dept\_no=d.dept\_no;**

****

5. A list showing first name, last name, and sex for employees whose first name is "Hercules" and last names begin with "B.“

**select first\_name, last\_name, sex from employees**

**where first\_name='Hercules' and last\_name like 'B%';**



6. A list showing all employees in the Sales department, including their employee number, last name, first name, and department name.

**create table department as**

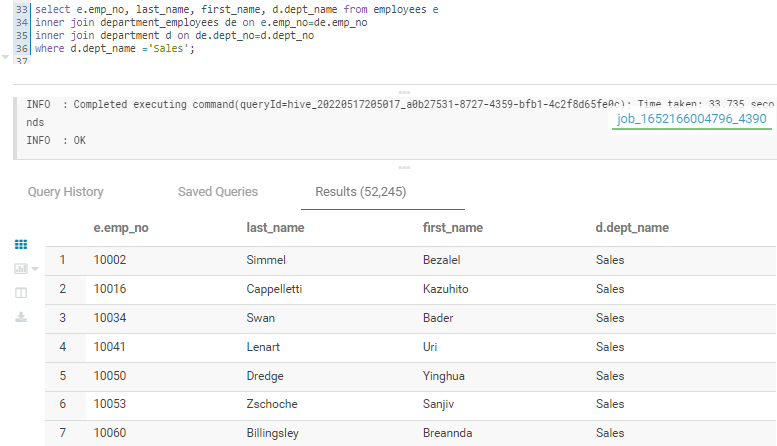
**select dept\_no, substr(dept\_name, 2, length(dept\_name)-2) as dept\_name from departments;**

**select e.emp\_no, last\_name, first\_name, d.dept\_name from employees e**

**inner join department\_employees de on e.emp\_no=de.emp\_no**

**inner join department d on de.dept\_no=d.dept\_no**

**where d.dept\_name ='Sales';**

****

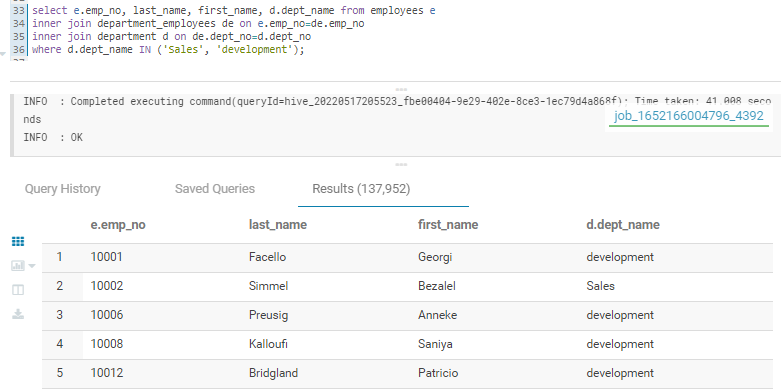
**7.** A list showing all employees in the Sales and Development departments, including their employee number, last name, first name, and department name.

**select e.emp\_no, last\_name, first\_name, d.dept\_name from employees e**

**inner join department\_employees de on e.emp\_no=de.emp\_no**

**inner join department d on de.dept\_no=d.dept\_no**

**where d.dept\_name IN ('Sales', 'development');**

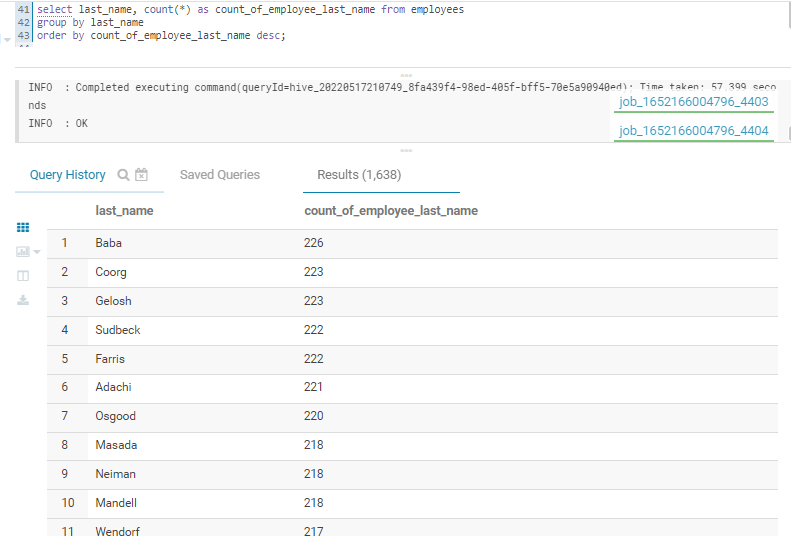
****

**8.** A list showing the frequency count of employee last names, in descending order. ( i.e., how many employees share each last name

**select last\_name, count(\*) as count\_of\_employee\_last\_name from employees**

**group by last\_name**

**order by count\_of\_employee\_last\_name desc;**



9. Histogram to show the salary distribution among the employees

**select**

**cast(hist.x as int) as bin\_center,**

**cast(hist.y as bigint) as bin\_height**

**from**

**(select**

**histogram\_numeric(salary, 20) as A\_hist**

**from**

**salaries) t**

**lateral view explode(A\_hist) exploded\_table as hist;**

****

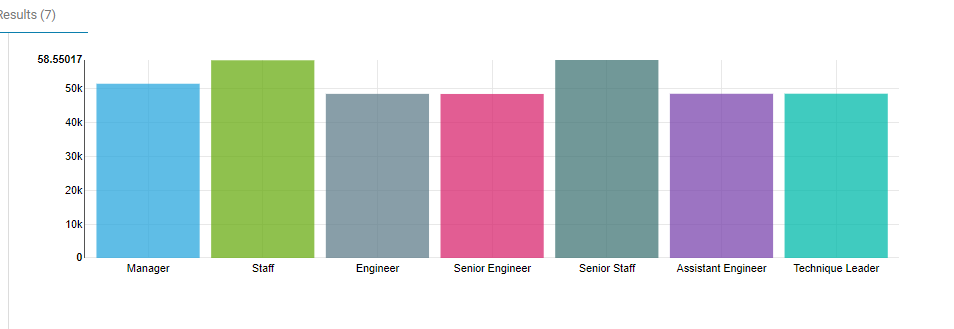
**10.** Bar graph to show the Average salary per title (designation)

**select t.title, avg(s.salary) as avg\_salary from titles t**

**inner join employees e on t.title\_id = e.emp\_titles\_id**

**inner join salaries s on e.emp\_no = s.emp\_no**

**group by t.title;**

****

**11.** Calculate employee tenure & show the tenure distribution among the employees

**12. a)**  Count the number of employee’s left and not left the company.

**select left\_company, count(\*) from employees**

**group by left\_company;**

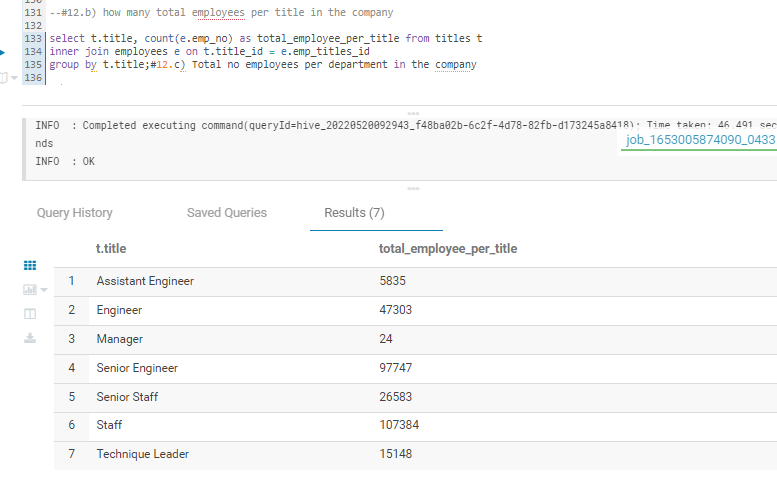
****

1. b) how many total employees per title in the company

**select t.title, count(e.emp\_no) as total\_employee\_per\_title from titles t**

**inner join employees e on t.title\_id = e.emp\_titles\_id**

**group by t.title;**



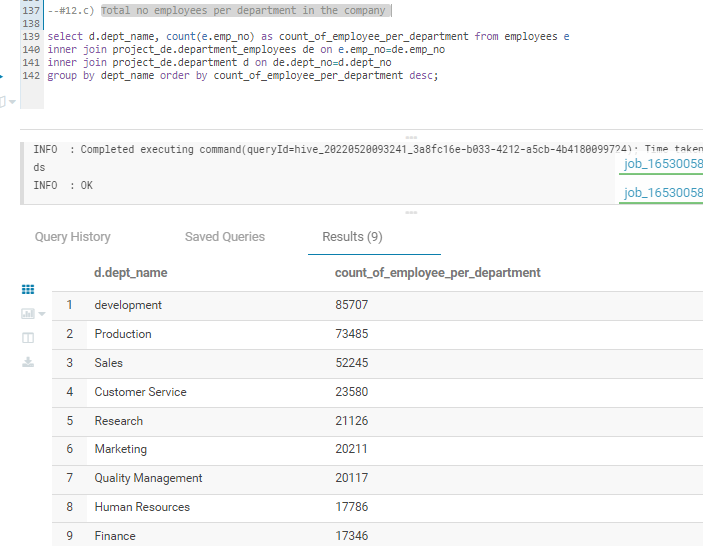
**12.c)** Total no employees per department in the company

**select d.dept\_name, count(e.emp\_no) as count\_of\_employee\_per\_department from employees e**

**inner join project\_de.department\_employees de on e.emp\_no=de.emp\_no**

**inner join project\_de.department d on de.dept\_no=d.dept\_no**

**group by dept\_name order by count\_of\_employee\_per\_department desc;**

****

**12.d**) top 3 department where employees are leaving the company

**select d.dept\_name, count(e.emp\_no) as total\_no\_of\_employees\_left from project\_de.employees e**

**inner join project\_de.department\_employees de on e.emp\_no=de.emp\_no**

**inner join project\_de.department d on de.dept\_no=d.dept\_no**

**where left\_company = "true" group by dept\_name order by total\_no\_of\_employees\_left desc;**

**12. e)**  Create bins of Salary to show the frequency of number of employees in each salary group.

**Create table table salary\_dist**

**select**

**case**

**when s.salary >= 40000 and s.salary < 50000 then '40-50k'**

**when s.salary >= 50000 and s.salary < 60000 then '50 -60k'**

**when s.salary >= 60000 and s.salary < 70000 then '60 -70k'**

**when s.salary >= 70000 and s.salary < 80000 then '70 -80k'**

**when s.salary >= 80000 and s.salary < 90000 then '80 -90k'**

**when s.salary >= 90000 and s.salary < 100000 then '90 -100k'**

**when s.salary >= 100000 then '100k+'**

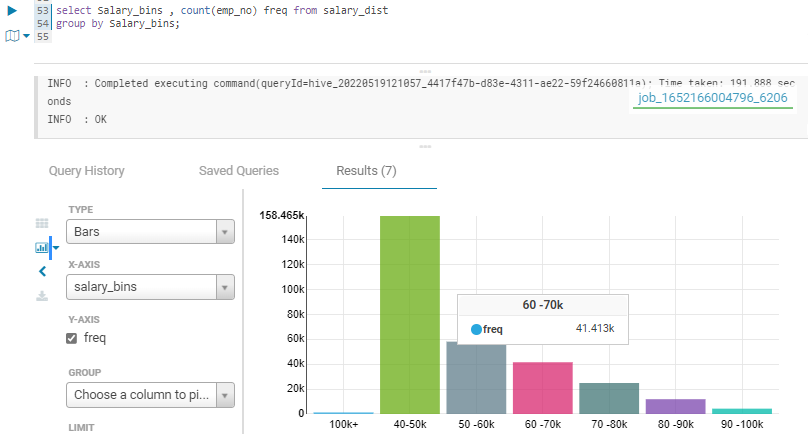
**end as Salary\_bins, e.emp\_no**

**from employees e**

**inner join salaries s on e.emp\_no = s.emp\_no;**

**select Salary\_bins , count(emp\_no) freq from salary\_dist**

**group by Salary\_bins;**



12.f) list of emp\_name, title, dept\_name, salary for each employee

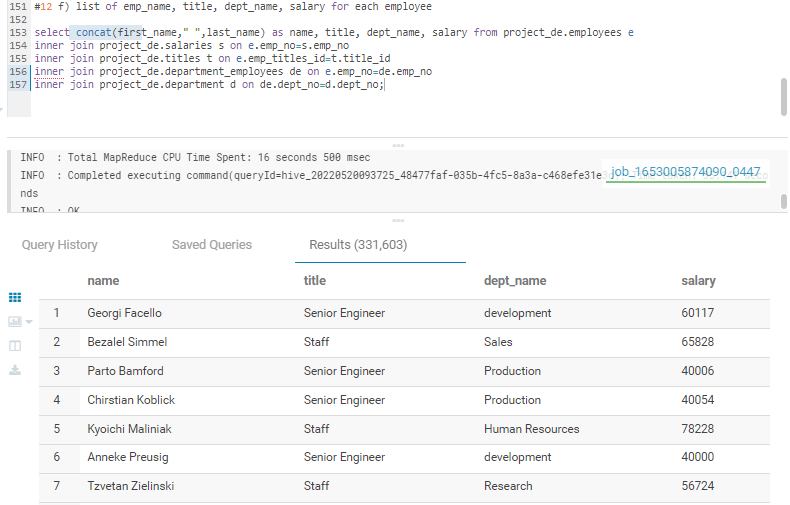
**select concat(first\_name," ",last\_name) as name, title, dept\_name, salary from project\_de.employees e**

**inner join project\_de.salaries s on e.emp\_no=s.emp\_no**

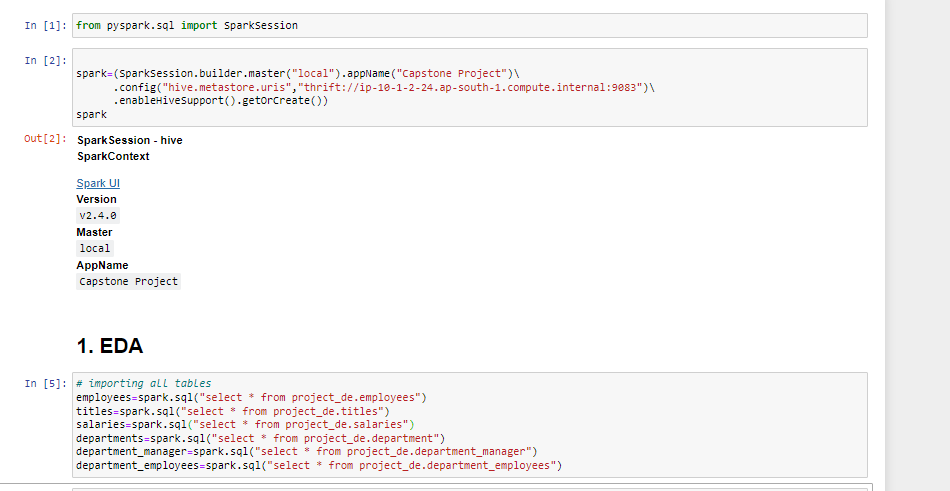
**inner join project\_de.titles t on e.emp\_titles\_id=t.title\_id**

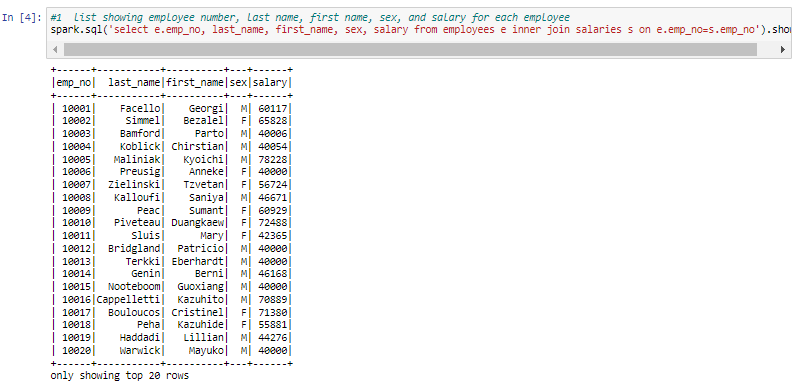
**inner join project\_de.department\_employees de on e.emp\_no=de.emp\_no**

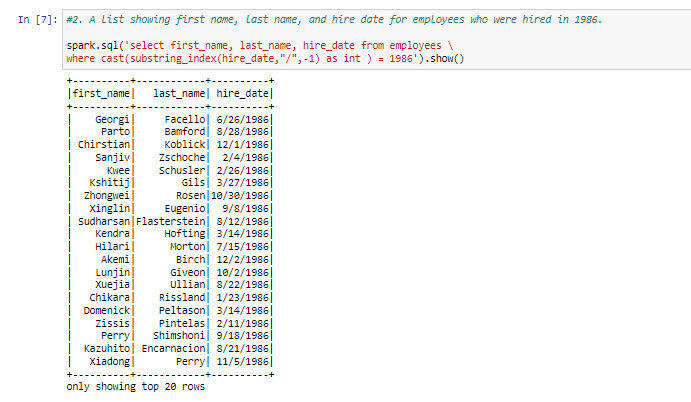
**inner join project\_de.department d on de.dept\_no=d.dept\_no;**

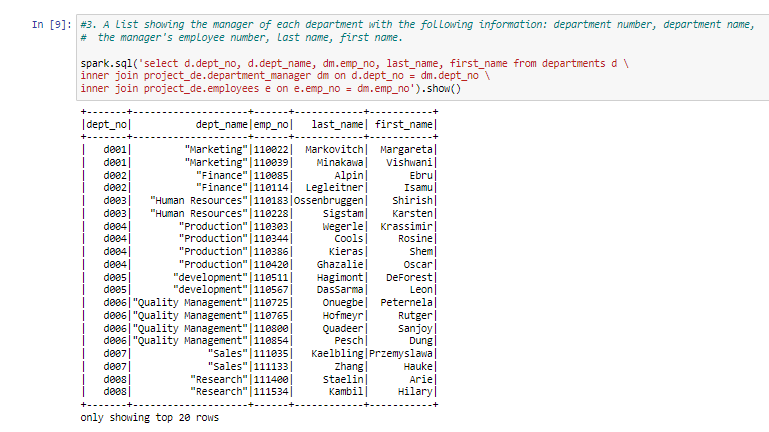
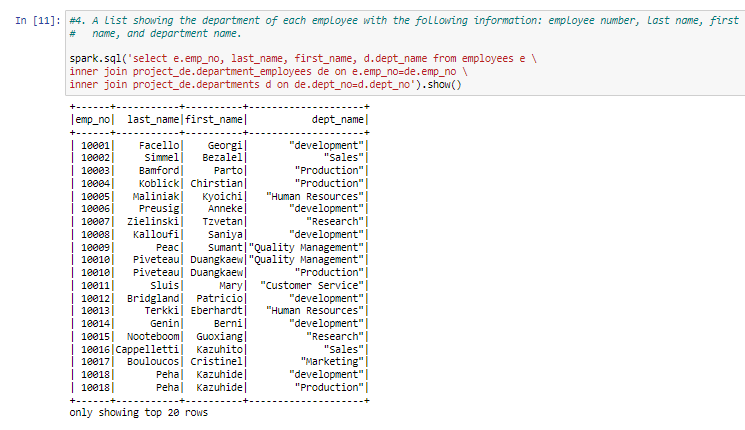
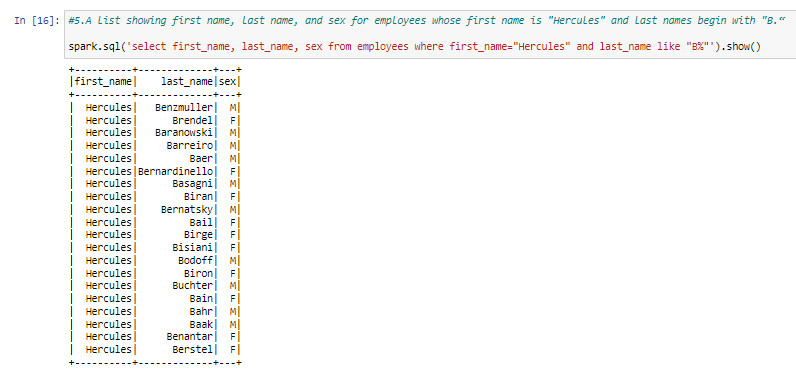
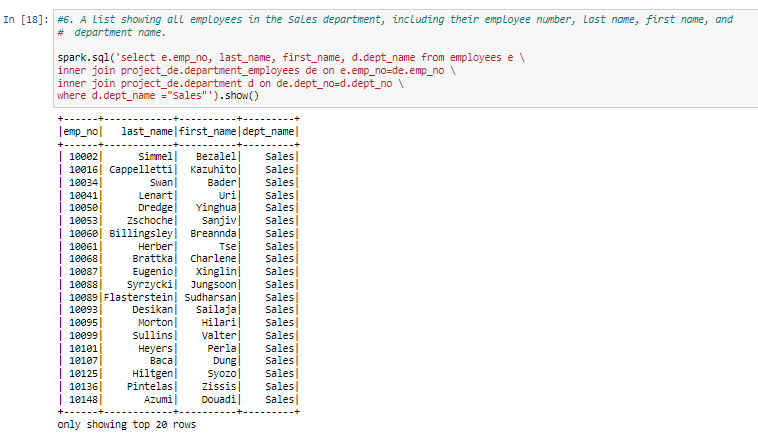
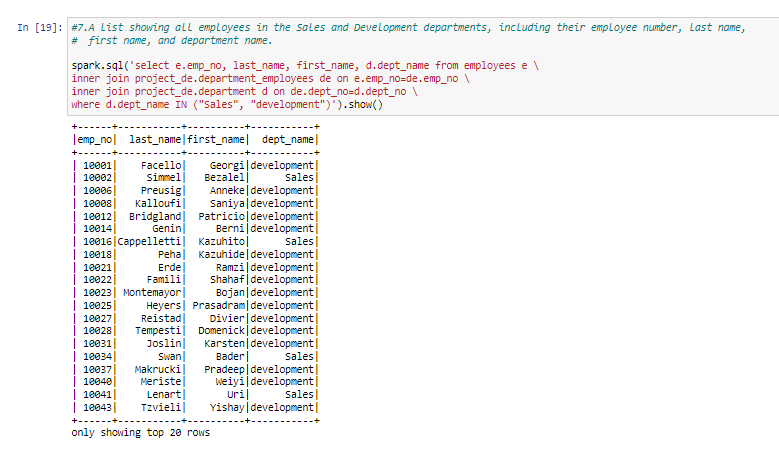
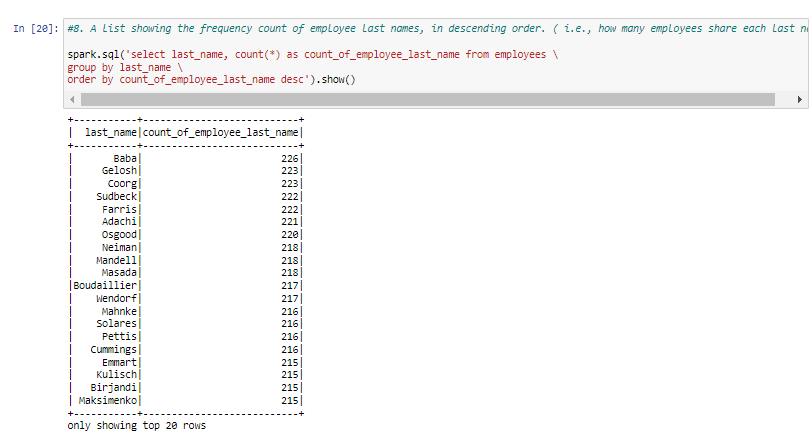
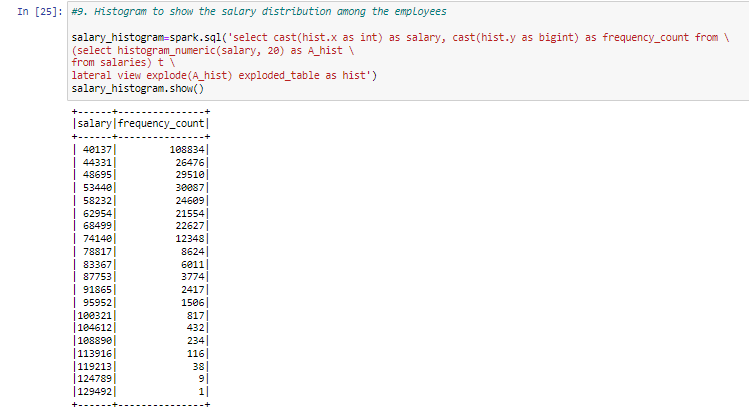
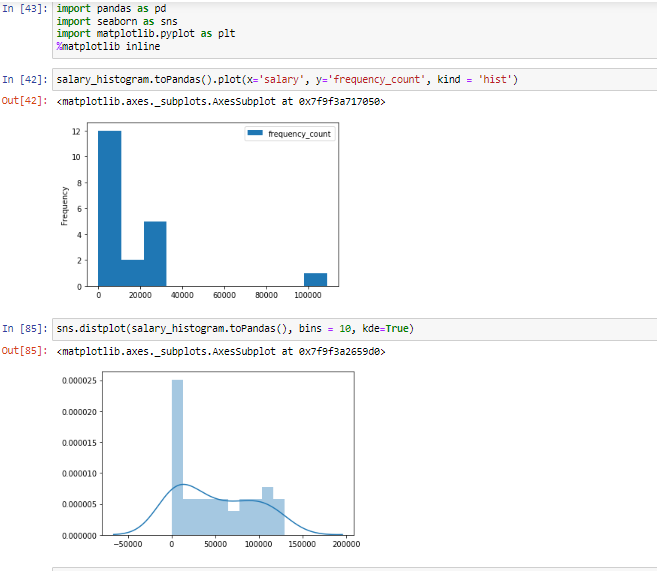


**EDA outputs in SPARK**

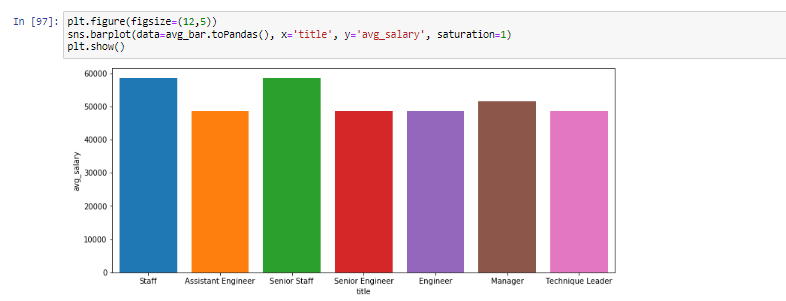
****

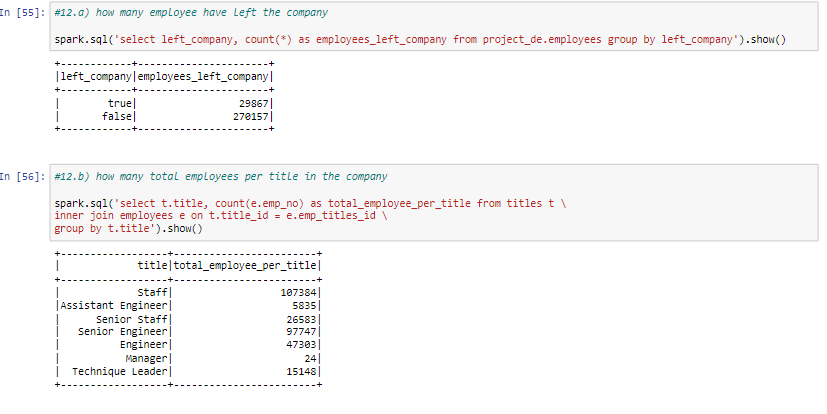
****

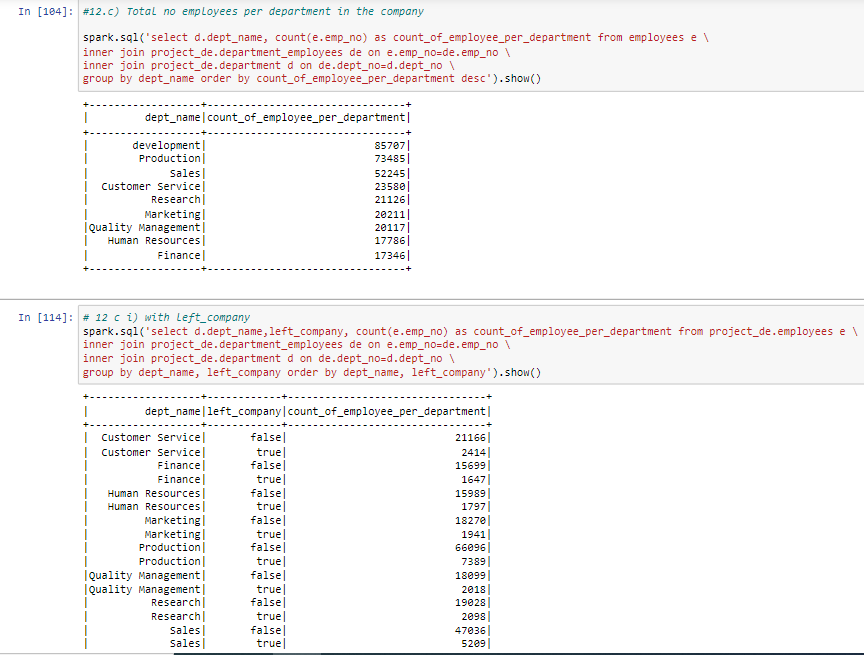
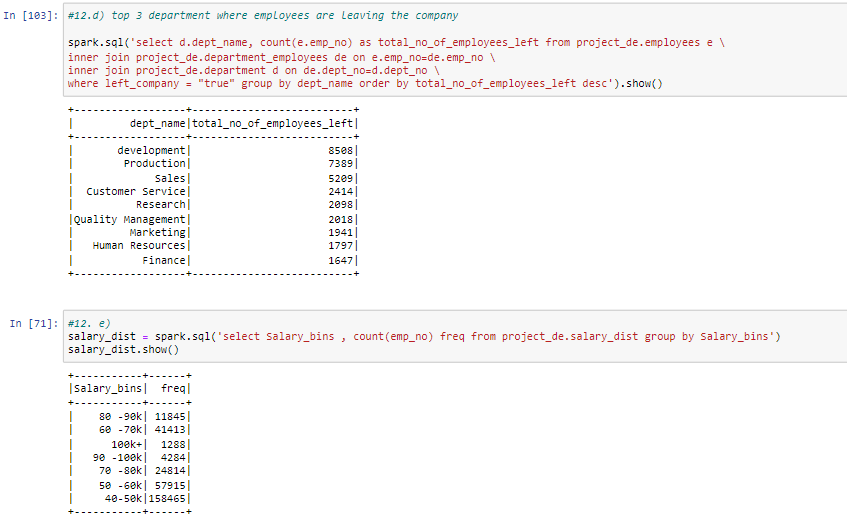
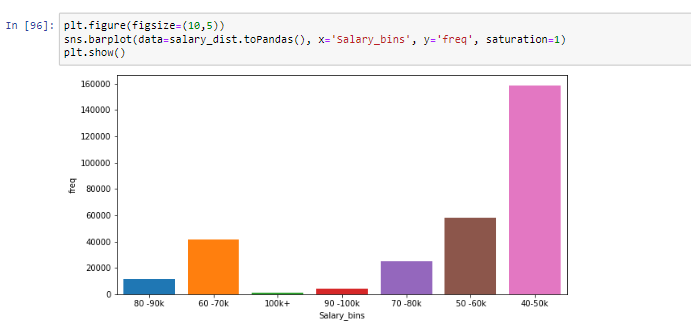
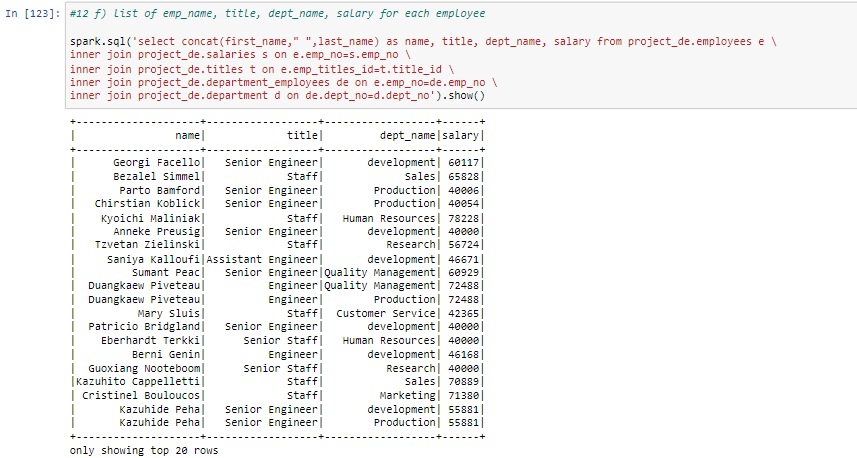
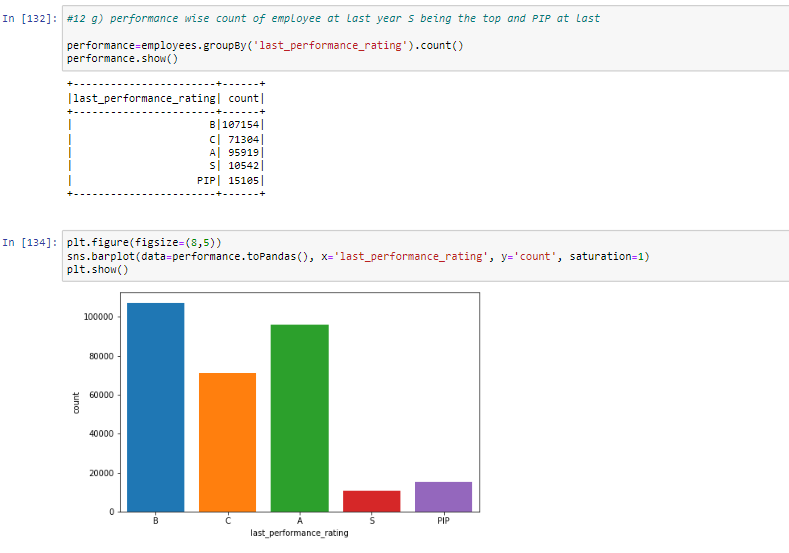
****

**** ****      

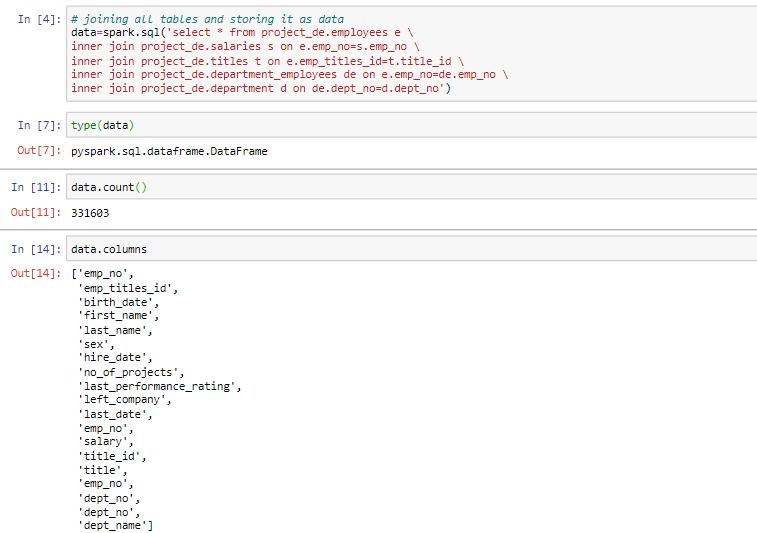
****

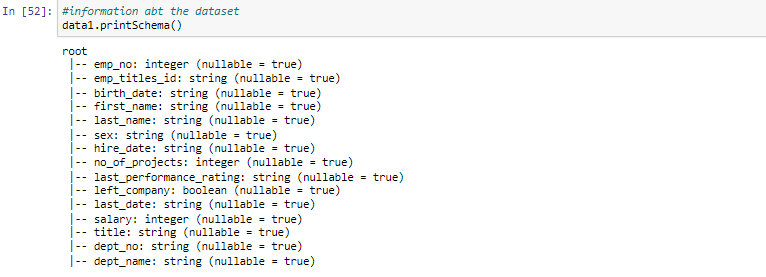
****

****

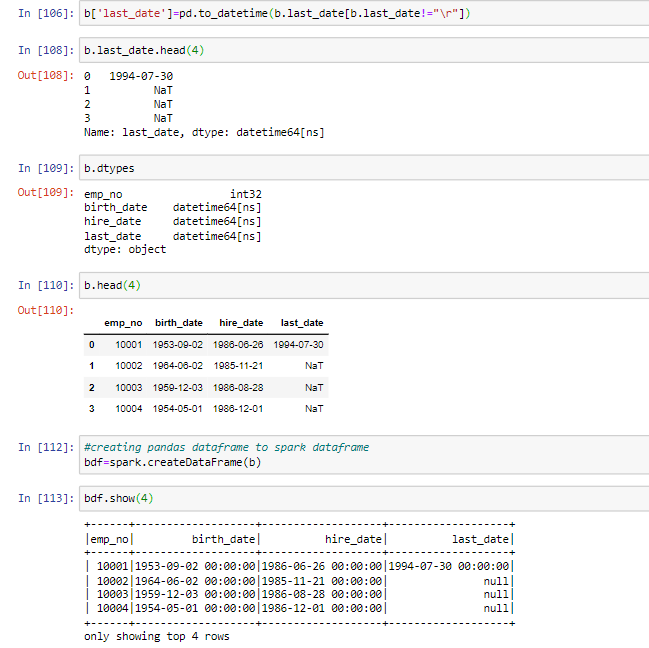
**** ****   

**Build ML Model : - Classification Model**

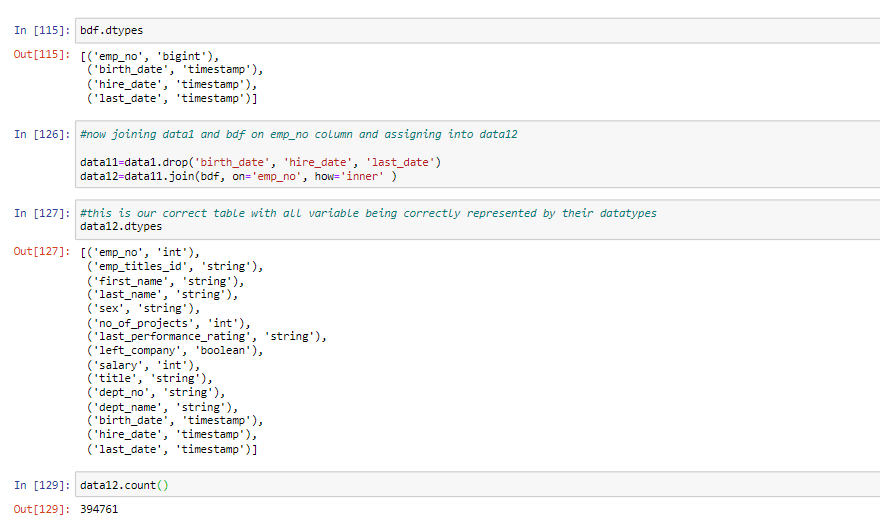
****

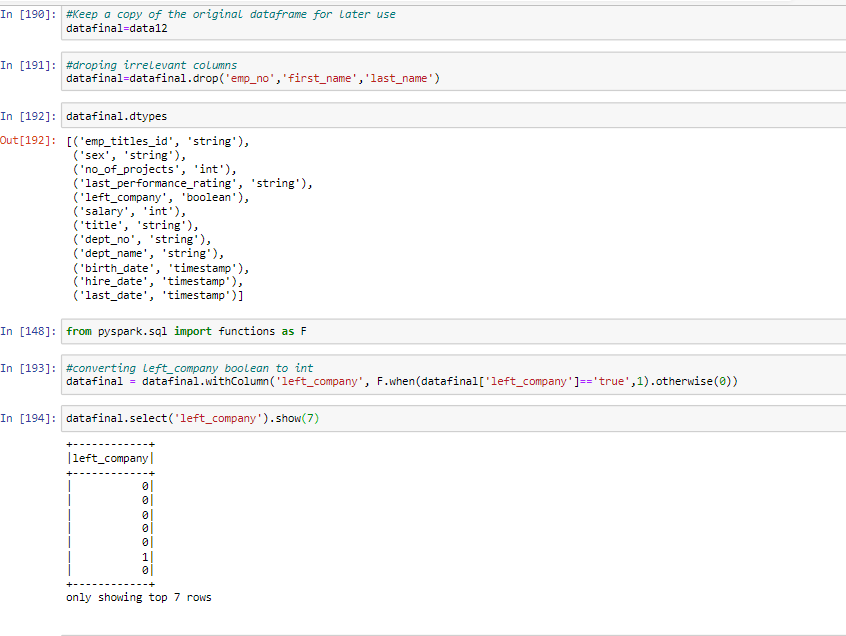
****

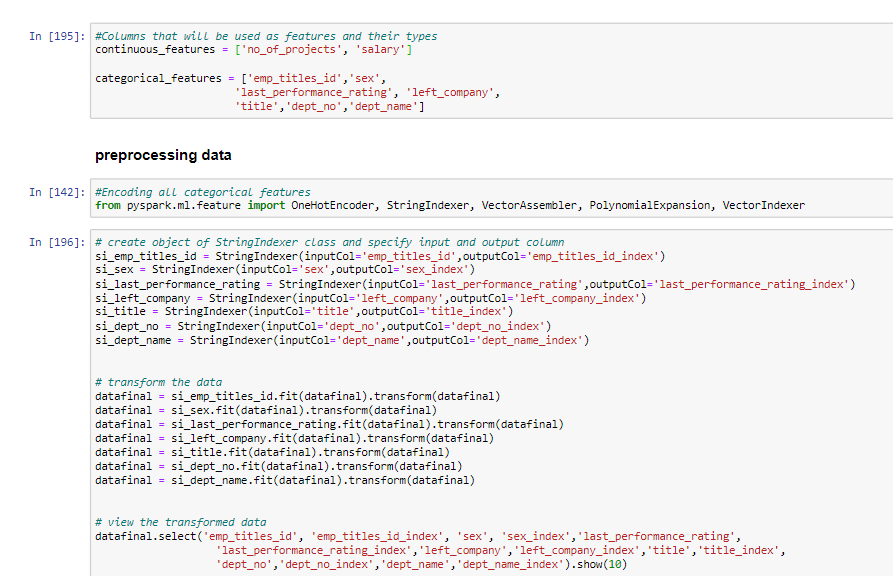
****

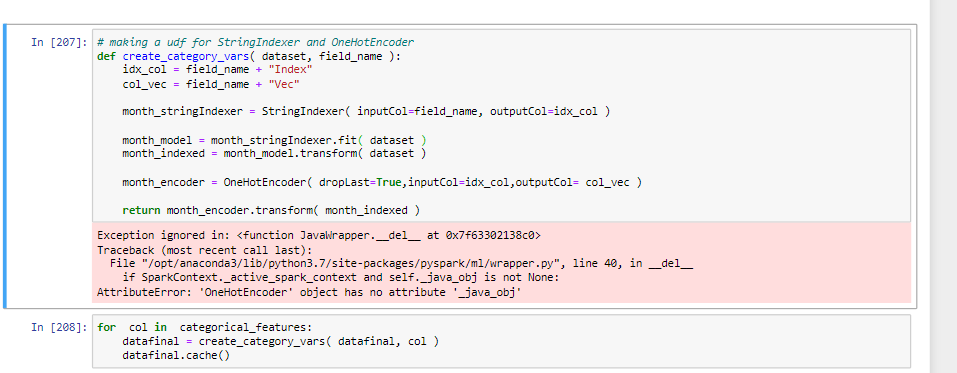


**Logistic Regression**



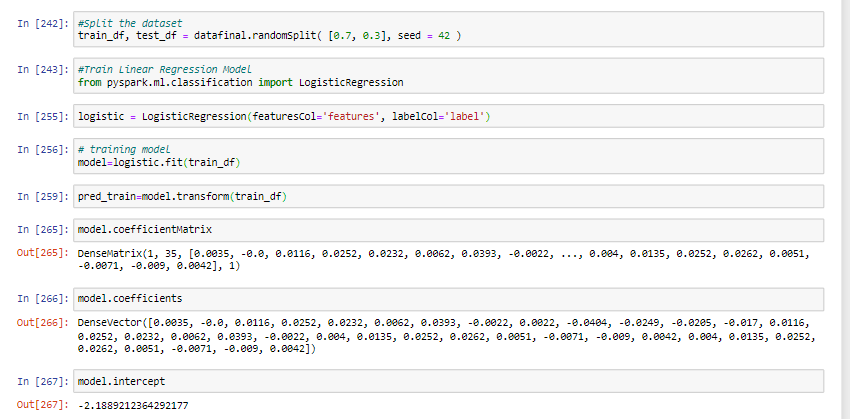


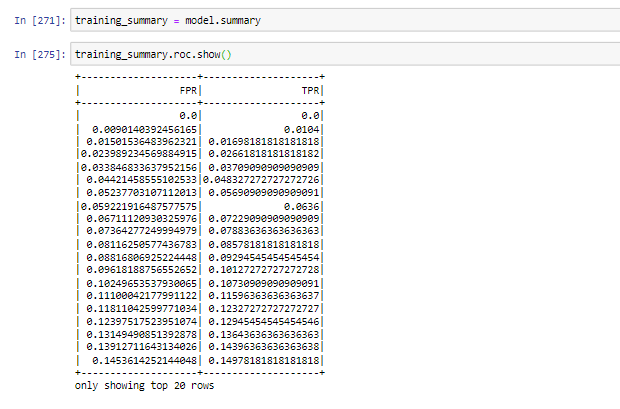


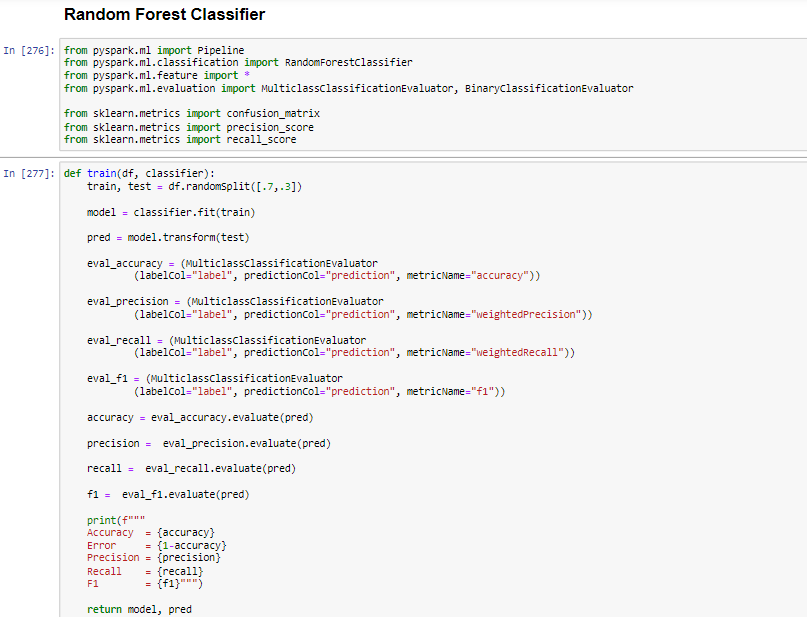


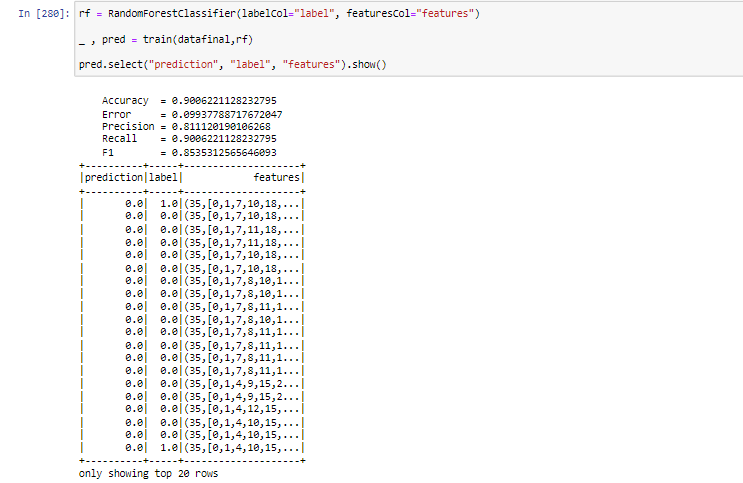












**Create entire data pipeline and ML pipe line**

**Create .sql file with commands of create database and table with queries.**

File - > create\_database\_table\_pipeline\_sql.sql



**Create .sh file as mysql\_sqoop\_pipeline\_sh.sh and execute it.**

Commands are to be saved in mysql\_sqoop\_pipeline\_sh.sh file are

hdfs dfs -rm -r projectdata

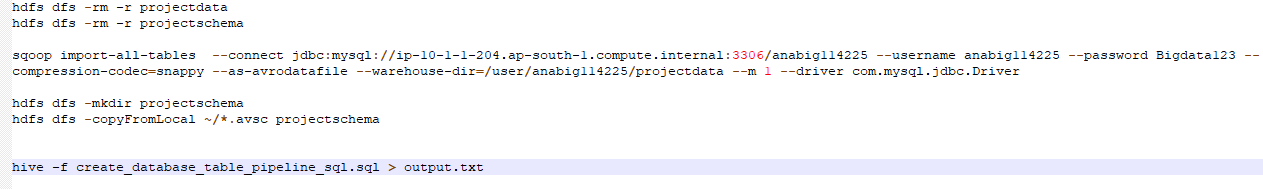
hdfs dfs -rm -r projectschema

sqoop import-all-tables --connect jdbc:mysql://ip-10-1-1-204.ap-south-1.compute.internal:3306/anabig114225 --username anabig114225 --password Bigdata123 --compression-codec=snappy --as-avrodatafile --warehouse-dir=/user/anabig114225/projectdata --m 1 --driver com.mysql.jdbc.Driver

hdfs dfs -mkdir projectschema

hdfs dfs -copyFromLocal ~/\*.avsc projectschema

hive -f create\_database\_table\_pipeline\_sql.sql > output.txt



**Create entire ML pipeline**



**Challenges**

* Creating the data tables as per their correct data type and then importing it to MySQL
* Deciding in which format to import the tables from MySQL into HDFS
* Finding. avsc schema files of table and then saving them on hdfs into new directory
* Creating table in hive using the format as specified earlier while importing the tables,

With mentioning data location and schema location.

* Taking hive tables to spark
* Conveting datatypes of variables
* Building sparkML with different techniques
* Finding wayouts to create data pipeline and ML pipeline

**Way ahead or Conclusion**

On whole it’s way very good learning project assimilating all interconnecting all tools mysql, sqoop, hdfs, hive, spark, sparkML for transferring the data tables, schemas and doing analysis on them. All of the tools being integrated into this one single project of data engineering.