***PROJECT CODE***

***CHECKPOINT 1***

**Load the data into HDFS, Hive Managed table, Hive External table and Spark DataFrame.**

*HDFS:*

**//creating a directory in HDFS**

hdfs dfs -mkdir AadharDataSet

**//Loading a file from local directory to HDFS directory.**

hdfs dfs -put aadhar.csv AadharDataSet

**//Displaying the data present in a file stored in HDFS directory.**

hdfs dfs -cat AadharDataSet/aadhar.csv

*INTERNAL AND EXTERNAL DATABASE using Hive*

**//Creating a Hive Database**

create database if not exists Aadhar;

**//Opening a hive database for use.**

use Aadhar;

**//Creating a Managed DataBase using Hive**

create table if not exists Aadhar\_Managed(Registrar String,Enrollment\_Agency String,State String,District String,Sub\_District String,Pincode String,Gender String,Age int,Aadhar\_Generated int,Enrollment\_Rejected int,Residents\_Providing\_Email int,Residents\_Providing\_Mobile\_Number int) row format delimited fields terminated by ',' stored as orcfile TBLPROPERTIES('skip.header.line.count'='1');

**//Loading the data into Managed Database**

Load data inpath "/user/cloudera/AadharDataSet/aadhar.csv" into table Aadhar\_Managed;

**//Inserted the result of the query in the text File.**

insert overwrite local directory '/home/cloudera/AadharManaged' row format delimited fields terminated by "," stored as textfile select \* from Aadhar\_Managed limit 25;

**//Creating an External Database using Hive**

create external table if not exists Aadhar\_External(Registrar String,Enrollment\_Agency String,State String,District String,Sub\_District String,Pincode String,Gender String,Age int,Aadhar\_Generated int,Enrollment\_Rejected int,Residents\_Providing\_Email int,Residents\_Providing\_Mobile\_Number int) row format delimited fields terminated by ',' stored as textfile location "/user/cloudera/AadharDatSet/aadhar.csv" TBLPROPERTIES('skip.header.line.count'='1');

**//Inserted the result of the query in the textFile.**

insert overwrite local directory '/home/cloudera/AadharExternal' row format delimited fields terminated by "," stored as textfile select \* from Aadhar\_External limit 25;

*SPARK DATA FRAME*

**//Creating a RDD of a dataset**

val RDD=sc.textFile("/user/cloudera/AadharDataSet/aadhar.csv")

**//Extracting the first row from the dataset**

val firstRDD=RDD.first()

**//Remove the header row from the dataset**

val filteredRDD=RDD.filter(x=>x!=firstRDD)

**//Removing commas from the file**

val aadharRDD=filteredRDD.map(x=>(x.split(",")(0),x.split(",")(1),x.split(",")(2),x.split(",")(3),x.split(",")(4),x.split(",")(5),x.split(",")(6),x.split(",")(7).toInt,x.split(",")(8).toInt,x.split(",")(9).toInt,x.split(",")(10).toInt,x.split(",")(11).toInt))

**//Storing the RDD into Data Frame.**

val AadharDF=aadharRDD.toDF("Registrar","Enrollment\_Agency","State","District","Sub\_District","PinCode","Gender","Age","Aadhar\_Generated","Enrollment\_Rejected","Residents\_Providing\_Emails","Residents\_Providing\_Mobile\_Number")

**//Displaying the Results**

AadharDF.show(25)

***CHECKPOINT 2***

**2. Describe the schema.**

scala> AadharDF.printSchema

**//Converting Data Fields into Table.**

AadharDF.registerTempTable("Aadhar");

**3. Find the count and names of registrars in the table.**

val query=sqlContext.sql("select registrar,count(registrar) as Count from Aadhar group by registrar")

**4. Find the number of states, districts in each state and sub-districts in each district.**

val query=sqlContext.sql("select count(state) as COUNT\_OF\_STATE from Aadhar")

val query=sqlContext.sql("select state,count(district) as Count\_Of\_District from Aadhar group by state")

val query=sqlContext.sql("select district,count(sub\_district) as Count\_Of\_Sub\_District from Aadhar group by district")

**5. Find the number of males and females in each state from the table.**

val query=sqlContext.sql("select state,gender,count(gender) as Count from Aadhar group by state,gender order by state,gender")

**6. Find out the names of private agencies for each state.**

val query=sqlContext.sql("select state,enrollment\_agency,count(enrollment\_agency) as Count from Aadhar group by state,enrollment\_agency order by state,enrollment\_agency")

***CHECKPOINT 3***

**7. Find top 3 states generating most number of Aadhaar cards.**

val query=sqlContext.sql("select State,sum(aadhar\_generated) as Sum from Aadhar group by state order by sum(aadhar\_generated) desc limit 3")

**8. Find top 3 private agencies generating the most number of Aadhar cards.**

val query=sqlContext.sql("select Enrollment\_Agency,sum(aadhar\_generated) as Sum from Aadhar group by Enrollment\_Agency order by sum(aadhar\_generated) desc limit 3")

**9. Find the number of residents providing email, mobile number. (Hint: consider non-zero values.)**

val query=sqlContext.sql("select sum(Residents\_Providing\_Emails) as Sum\_of\_Residents\_Providing\_Emails,sum(Residents\_Providing\_Mobile\_Number) as Sum\_of\_Residents\_Providing\_Mobile\_Number from Aadhar")

**10. Find top 3 districts where enrolment numbers are maximum.**

val query=sqlContext.sql("select District,sum(aadhar\_generated + enrollment\_rejected) as Enrollment\_Number from Aadhar group by District order by sum(aadhar\_generated + enrollment\_rejected) desc limit 3")

***CHECKPOINT 4***

**11. Find the no. of Aadhaar cards generated in each state.**

val query=sqlContext.sql("select State, sum(aadhar\_generated) as Sum\_of\_Aadhar\_Generated from Aadhar group by State order by State")

**12. Create a data frame using the file and provide its summary.**

AadharDF.printSchema

**13. Write a command to see the correlation between “age” and “mobile\_number”? (Hint: Consider the percentage of people who have provided the mobile number out of the total applicants)**

val query=sqlContext.sql("select corr(age,residents\_providing\_mobile\_number) as Correlation from aadhar")

**14. Find the number of unique pincodes in the data.**

val query=sqlContext.sql("select distinct(pincode) as Unique\_Pincode from aadhar")

**15. Find the number of Aadhaar registrations rejected in Uttar Pradesh and Maharashtra.**

val query=sqlContext.sql("select State,Sum(enrollment\_rejected) as Registeration\_Rejected from aadhar where State='Uttar Pradesh' or State='Maharashtra' group by State")

***CHECKPOINT 5***

**16. The top 3 states where the percentage of Aadhaar cards being generated for males is the highest.**

val query=sqlContext.sql("select state,round((sum(aadhar\_generated)/sum(aadhar\_generated+rejected))\*100,2) Percentage\_of\_aadhar from aadhar where gender like 'M' group by state order by Percentage\_of\_aadhar desc limit 3");

**17. In each of these 3 states, identify the top 3 districts where the percentage of Aadhaar cards being rejected for females is the highest.**

val query=sqlContext.sql("select state,district,round((sum(rejected)/sum(aadhar\_generated+rejected))\*100,2) Percentage\_of\_rejected from aadhar where gender like 'F' and state like 'Andaman and Nicobar Islands' or state like 'Lakshadweep' or state like 'Others' group by state,district order by Percentage\_of\_rejected desc");

**18. The top 3 states where the percentage of Aadhaar cards being generated for females is the highest.**

val query=sqlContext.sql("select state,round((sum(aadhar\_generated)/sum(aadhar\_generated+rejected))\*100,2) Percentage\_of\_aadhar from aadhar where gender like 'F' group by state order by Percentage\_of\_aadhar desc limit 3");

**19. In each of these 3 states, identify the top 3 districts where the percentage of Aadhaar cards being rejected for males is the highest.**

val query=sqlContext.sql("select state,district,round((sum(rejected)/sum(aadhar\_generated+rejected))\*100,2) Percentage\_of\_rejected from aadhar where gender like 'M' and state like 'Dadra and Nagar Haveli' or state like 'Sikkim' or state like 'Others' group by state,district order by Percentage\_of\_rejected desc");

**20. The summary of the acceptance percentage of all the Aadhaar cards applications by bucketing the age group into 10 buckets.**

create table aadhar\_bucket(registrar string,private\_agency string,state string,district string,sub\_district string,pincode string,gender string, age int,aadhar\_generated int,rejected int,email\_id int,moblie\_number int) clustered by (age) into 10 buckets row format delimited fields terminated by ',' stored as textfile TBLPROPERTIES('serialization.null.format'='','skip.header.line.count'='1');