**Assignment-1**

Paresh Patil

ID: 8771798

PROG8450: Big Data Integration and Storage

Big data Solutions Architecture

Prof. Saber Amini

1 Write a Hive query to create a table called used\_cars from data. Use a schema that is   
appropriate for the column headings

I have used **cars** as a table name.

Text

Description automatically generated

A picture containing text

Description automatically generated

2. Look at the date column of the table used\_cars. Why does the date column have all   
NULL values?

:- Because the date column values are not in proper format and seconds are in decimals.

3. Bonus: Create a table such that the date column is read correctly based on the format   
in the dataset.

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated with low confidence

4. Write Hive queries to see how many missing values you have in each attribute? Based   
on the results, document how many missing values in each column we have. Especially,   
mention those columns with more than 50% missing values.

A screenshot of a computer

Description automatically generated with medium confidence

The query is not working for some of the columns, Thus I did it in a basic way.

Text

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

* Skt\_year is 84.91%
* Fuel\_type is 52.00%
* Colur\_slug is 94.10%

5. Group the price column and count the number of unique prices. Do you notice if there   
is a single price that is repeating across the ads?

Text

Description automatically generated

:- I could see 1295.34 prices is repeating maximum time

6. Write a Hive query to create a new table called clean\_used\_cars from used\_cars with   
the following conditions:   
o Drop the columns with more than 50% missing values   
o The manufacture year between 2000 and 2017 including 2000 and 2017   
o Both maker and model exist in the row   
o The price range is from 3000 to 2000,000 (3000 ≤ price ≤ 2000,000)   
o Remove any price you singled out in Step 3 (ie a price that repeats too frequently for   
a random set of ads).

Text

Description automatically generated  
   
Text

Description automatically generated

7. Write a Hive query to find how many records remained clean\_used\_cars

Text

Description automatically generated

8. Write a Hive query to find the make and model for the cars with the top 10 highest average price Text

Description automatically generated

9. Write a Hive query to find the make and model for the cars with the top 10 lowest   
average price

Text

Description automatically generated  
10. Write a Hive query to recommend top five make and model for Economic segment   
customers (Top five manufacturers in the 3000 to 20,000 price range;3000≤price<20,000) -   
based on the top average price Text

Description automatically generated with low confidence

11. Write a Hive query to recommend top five make and model for Intermediate segment   
customers (Top five manufacturers in the 20,000 to 300,000 price range;   
3000≤price<20,000) - based on the top average price

Text

Description automatically generated  
12. Write a Hive query to recommend the top five make and model for the Luxury segment   
customers (Top five manufacturers in the 300,000 to 2000,000 price range;   
300,000≤price<2000,000) - based on the top average price

Text

Description automatically generated

13) top five high mileage used carsText

Description automatically generated

14) Top five high engine power cars

Text

Description automatically generated

15) Top five less car age and less price range lower than 4000$

Text

Description automatically generated with low confidence

QUERIES 1 to 15 in sequence;

CREATE EXTERNAL TABLE IF NOT EXISTS cars (

maker string,

model string,

mileage float,

manufacture\_year int,

engine\_displacement int,

engine\_power int,

body\_type string,

color\_slug string,

stk\_year int,

transmission string,

door\_count int,

seat\_count int,

fuel\_type string,

date\_created timestamp,

date\_last\_seen timestamp,

price\_eur float)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/BigData/Hive/';

CREATE EXTERNAL TABLE IF NOT EXISTS Temp\_usedcars (

maker string,

model string,

mileage float,

manufacture\_year int,

engine\_displacement int,

engine\_power int,

body\_type string,

color\_slug string,

stk\_year int,

transmission string,

door\_count int,

seat\_count int,

fuel\_type string,

date\_created string,

date\_last\_seen string,

price\_eur float)

ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/BigData/Hive/';

CREATE EXTERNAL TABLE IF NOT EXISTS USEDCARS (

maker string,

model string,

mileage float,

manufacture\_year int,

engine\_displacement int,

engine\_power int,

body\_type string,

color\_slug string,

stk\_year int,

transmission string,

door\_count int,

seat\_count int,

fuel\_type string,

date\_created timestamp,

date\_last\_seen timestamp,

price\_eur float)

INSERT INTO TABLE USEDCARS SELECT maker,

model,

mileage,

manufacture\_year,

engine\_displacement,

engine\_power,

body\_type,

color\_slug,

stk\_year,

transmission,

door\_count,

seat\_count,

fuel\_type,

CAST(from\_unixtime(unix\_timestamp(date\_created, 'yyyy-MM-dd HH:mm:ss')) AS timestamp),

CAST(from\_unixtime(unix\_timestamp(date\_last\_seen, 'yyyy-MM-dd HH:mm:ss')) AS timestamp),

price\_eur FROM Temp\_usedcars;

SELECT

COUNT(\*)-COUNT(maker ) As maker ,

COUNT(\*)-COUNT(model ) As model ,

((COUNT(\*)-COUNT(mileage ))/COUNT(\*))\*100 As mileage,

((COUNT(\*)-COUNT(manufacture\_year))/COUNT(\*))\*100 As manufacture\_year,

((COUNT(\*)-COUNT(engine\_displacement))/COUNT(\*))\*100 As engine\_displacement,

((COUNT(\*)-COUNT(engine\_power))/COUNT(\*))\*100 As engine\_power,

COUNT(\*)-COUNT(body\_type) As body\_type,

COUNT(\*)-COUNT(color\_slug) As color\_slug,

((COUNT(\*)-COUNT(stk\_year))/COUNT(\*))\*100 As stk\_year,

COUNT(\*)-COUNT(transmission ) As transmission,

((COUNT(\*)-COUNT(door\_count))/COUNT(\*))\*100 As door\_count,

((COUNT(\*)-COUNT(seat\_count))/COUNT(\*))\*100 As seat\_count,

COUNT(\*)-COUNT(fuel\_type) As fuel\_type,

COUNT(\*)-COUNT(date\_created) As date\_created,

COUNT(\*)-COUNT(date\_last\_seen) As date\_last\_seen,

COUNT(\*)-COUNT(price\_eur) As price\_eur

from USEDCARS

select COUNT(\*)\*100/(select COUNT(\*) from USEDCARS) from USEDCARS where maker= '' or maker is NULL;

select COUNT(\*)\*100/(select COUNT(\*) from USEDCARS) from USEDCARS where model = '' or model is NULL;

select COUNT(\*)\*100/(select COUNT(\*) from USEDCARS) from USEDCARS where body\_type= '' or body\_type is NULL;

select COUNT(\*)\*100/(select COUNT(\*) from USEDCARS) from USEDCARS where color\_slug= '' or color\_slug is NULL;

select COUNT(\*)\*100/(select COUNT(\*) from USEDCARS) from USEDCARS where transmission= '' or transmission is NULL;

select COUNT(\*)\*100/(select COUNT(\*) from USEDCARS) from USEDCARS where fuel\_type= '' or fuel\_type is NULL;

select price\_eur,count(price\_eur) AS COUNTPRICE from USEDCARS group by price\_eur order by COUNTPRICE DESC LIMIT 15

CREATE EXTERNAL TABLE IF NOT EXISTS clean\_used\_cars(

maker string,

model string,

mileage float,

manufacture\_year int,

engine\_displacement int,

engine\_power int,

body\_type string,

transmission string,

door\_count int,

seat\_count int,

date\_created timestamp,

date\_last\_seen timestamp,

price\_eur float);

INSERT INTO TABLE clean\_used\_cars SELECT maker,

model,

mileage,

manufacture\_year,

engine\_displacement,

engine\_power,

body\_type,

transmission,

door\_count,

seat\_count,

date\_created,

date\_last\_seen,

price\_eur FROM USEDCARS where manufacture\_year >= 2000 and manufacture\_year <= 2017 AND (maker IS NOT NULL and model != '') AND (model IS NOT NULL and maker !='') AND (price\_eur between 3000 and 2000000) AND (price\_eur != 1295.34)

select count(\*) from USEDCARS;

select maker,model, AVG(price\_eur) AS AVG\_PRICE\_EUR from clean\_used\_cars GROUP BY maker,model ORDER by

AVG\_PRICE\_EUR desc LIMIT 10;

select maker,model, AVG(price\_eur) AS AVG\_PRICE\_EUR from clean\_used\_cars GROUP BY maker,model ORDER by

AVG\_PRICE\_EUR asc LIMIT 10;

select maker,model, AVG(price\_eur) AS AVG\_PRICE\_EUR from clean\_used\_cars GROUP BY maker,model having AVG(price\_eur) >=3000 AND AVG(price\_eur) <20000 ORDER by AVG\_PRICE\_EUR desc LIMIT 5;

select maker,model, AVG(price\_eur) AS AVG\_PRICE\_EUR from clean\_used\_cars GROUP BY maker,model having AVG(price\_eur) <300000 AND AVG(price\_eur) >=20000 ORDER by AVG\_PRICE\_EUR desc LIMIT 5;

select maker,model, AVG(price\_eur) AS AVG\_PRICE\_EUR from clean\_used\_cars GROUP BY maker,model having AVG(price\_eur) >=300000 AND AVG(price\_eur) <2000000 ORDER by AVG\_PRICE\_EUR desc LIMIT 10;

lowest mileage

select maker,model,AVG(engine\_power) AS AVG\_enginepower, AVG(price\_eur) AS AVG\_PRICE\_EUR from clean\_used\_cars where engine\_power is not NULL GROUP BY maker,model,AVG\_enginepower, ORDER by AVG\_enginepower desc LIMIT 5 ;

HIgh engine power

select maker,model,AVG(engine\_power) as avgenginepower from clean\_used\_cars group by maker,model order by avgenginepower desc LIMIT 5 ;

less used cars with less price

SELECT maker,model,abs(extract(year from date\_created) - extract(year from current\_date())) as car\_age,price\_eur from clean\_used\_cars where price\_eur <=4000 order by car\_age LIMIT 5 ;