***SOAL***

# **Load data from CSV to Hive**

Given the file “**Manufacturers**.**csv**” and “**Regions**.**csv**”, you were asked to load the data from **Comma-Separated Values** (**CSV**) file to **Hive** for data integration.

# **Load data from MySQL to Hive**

Given the file “**create+insert.sql**” that consists of the data about **sales**, **cars**, and **stores**. You need to load the data to **MySQL** database, then **ingest** the data from **MySQL** database to **Hive** for data integration.

# **Query Analysis**

From the data in **Hive**, you need to gain some sales insight in **AmorE’s Dealership**, below are some statements you need to answer using **Hive** / **Impala** query:

* 1. Show **cars** which has been bought **more than 30** units **in 2018.**
  2. Show **top 10 region** which has the **most transactions** **in America.**
  3. Show **top 3 stores** which earned the **most gross profit** **in 2019.**
  4. Show **car manufacturers** that have **sell cars more than the average car manufacturer** for cars that are made **after 2010.**
  5. Show **store and total cars sold** for each store who did **more than 5 transactions** within year **2018-2019** and **determine how much investment** will be given based on the total car sold on the **year 2018-2019**:

|  |  |
| --- | --- |
| Total Car Sold | Investment |
| > 200 | 100.000.000 |
| 100 - 200 | 50.000.000 |
| 30 – 99 | 20.000.000 |
| 20 – 29 | 10.000.000 |
| < 20 | 0 |

***JAWABAN***

1. Proses pembuatan Databases

CREATE DATABASE AmoresDealership

CREATE EXTERNAL TABLE Manufacturers(

ManufacturerID INT,

ManufacturerName VARCHAR (255)

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

TBLPROPERTIES("skip.header.line.count"="1")

CREATE EXTERNAL TABLE Regions(

RegionID INT,

RegionName VARCHAR (255),

Country VARCHAR(255)

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

TBLPROPERTIES("skip.header.line.count"="1")

Proses memasukan data .CSV ke HIVE

(Bagian terminal)

(base) [cloudera@quickstart ~]$ hadoop fs -copyFromLocal Desktop/data

(base) [cloudera@quickstart ~]$ hadoop fs -ls

Found 3 items

drwxrwxrwx - cloudera cloudera 0 2020-10-19 09:43 RamenData

drwxr-xr-x - cloudera cloudera 0 2020-10-26 20:02 data

drwxr-xr-x - cloudera cloudera 0 2020-02-07 20:15 tmp

(base) [cloudera@quickstart ~]$ hadoop fs -chmod 777 /user/cloudera/data

(base) [cloudera@quickstart ~]$ hadoop fs -ls

Found 3 items

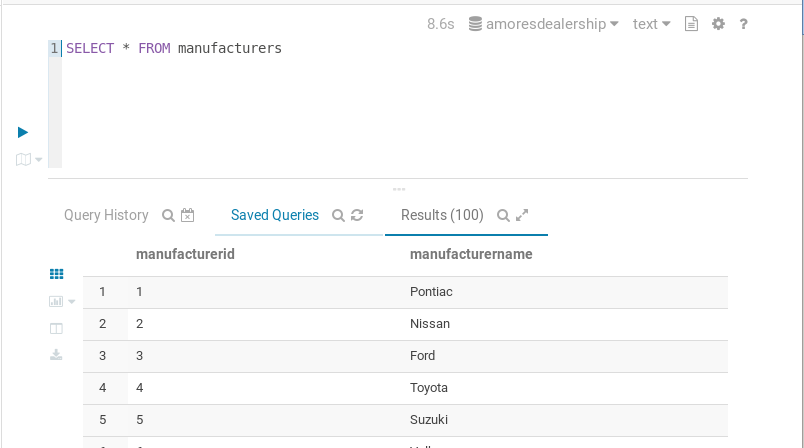
drwxrwxrwx - cloudera cloudera 0 2020-10-19 09:43 RamenData

drwxrwxrwx - cloudera cloudera 0 2020-10-26 20:02 data

drwxr-xr-x - cloudera cloudera 0 2020-02-07 20:15 tmp

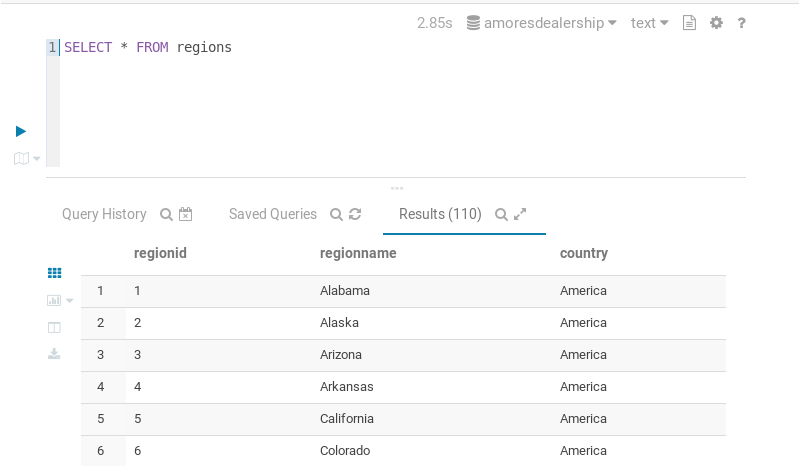
(Bagian Hive)

load data inpath '/user/cloudera/data/Manufacturers.csv' into table manufacturers 



load data inpath '/user/cloudera/data/Regions.csv' into table regions





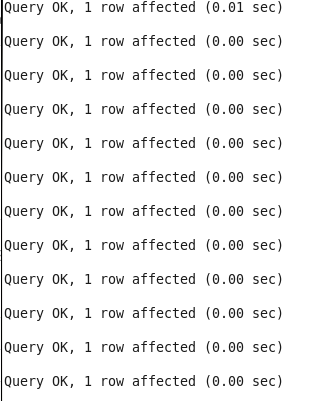
1. Proses pembuatan Databases di MySQL (Terminal)

(base) [cloudera@quickstart ~]$ mysql -u root -p

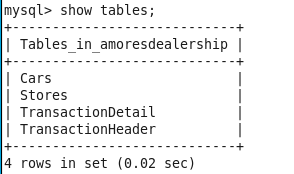
mysql> create database romannovelstore;

mysql> use romannovelstore;

mysql> source Desktop/data/create+insert.sql;

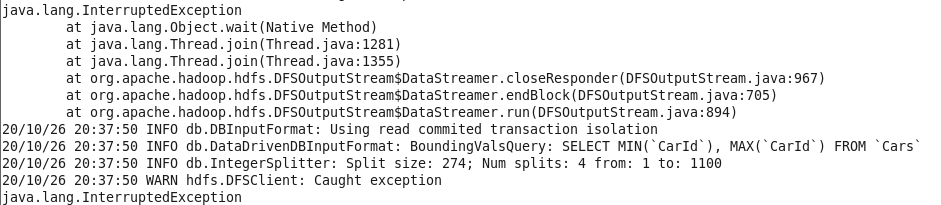


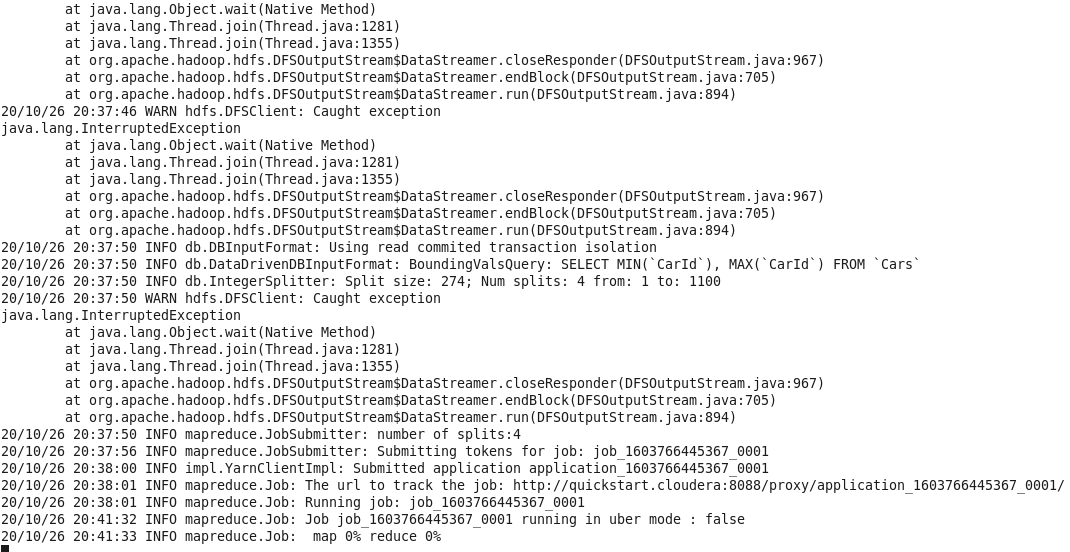
mysql> show tables;



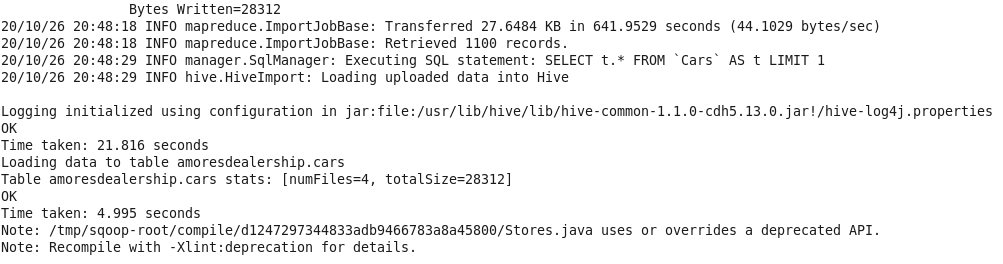
(base) [cloudera@quickstart ~]$ sudo sqoop import-all-tables --connect jdbc:mysql://quickstart:3306/amoresdealership --username=root -P --hive-import --hive-database=amoresdealership

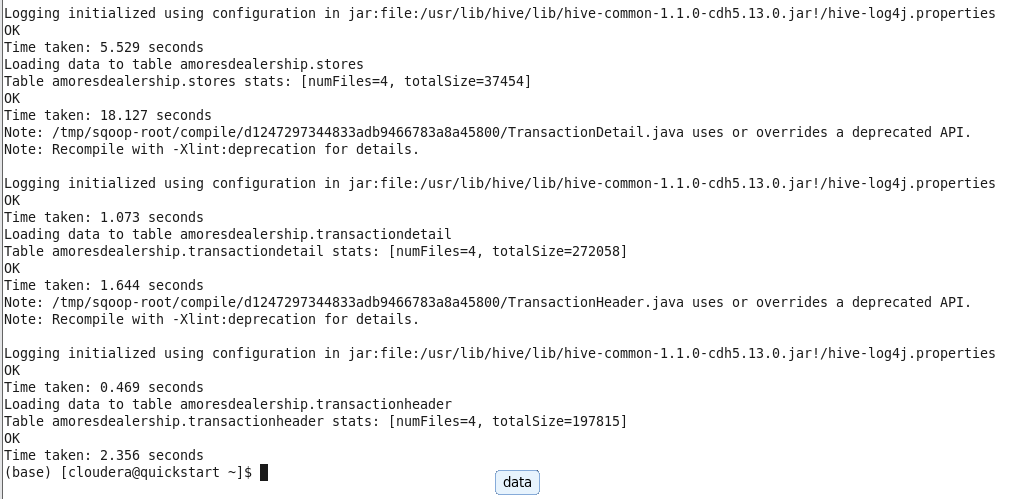




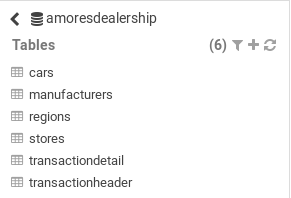








# setelah loading, jika sudah ok, kembali ke hive, refresh table, pilih Invalidate all metadata and rebuild index, Hasil nya seperti berikut ini:



1. Query :
2. Show **cars** which has been bought **more than 30** units **in 2018.**

SELECT carid,sum(quantity)FROM transactiondetail

JOIN transactionheader

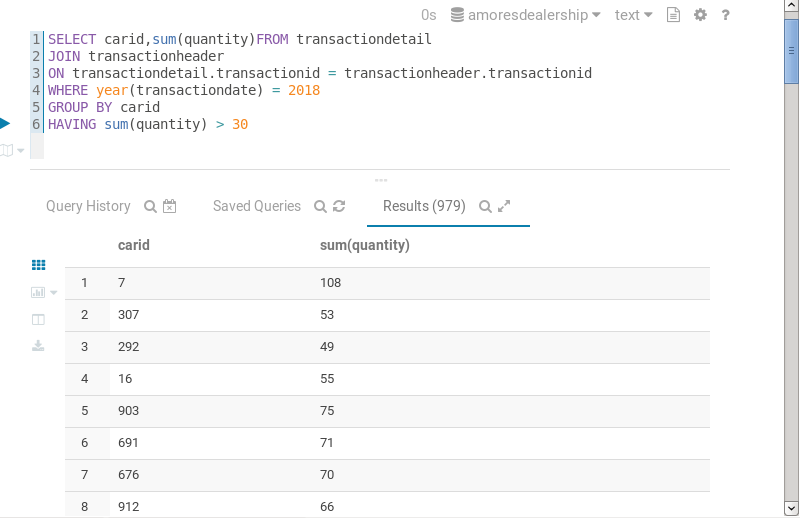
ON transactiondetail.transactionid = transactionheader.transactionid

WHERE year(transactiondate) = 2018

GROUP BY carid

HAVING sum(quantity) > 30

Proof :



1. Show **top 10 region** which has the **most transactions** **in America.**

SELECT regionname,sum(quantity) FROM regions

JOIN stores ON regions.regionid = stores.regionid

JOIN transactionheader ON stores.storeid = transactionheader.storeid

JOIN transactiondetail

ON transactionheader.transactionid = transactiondetail.transactionid

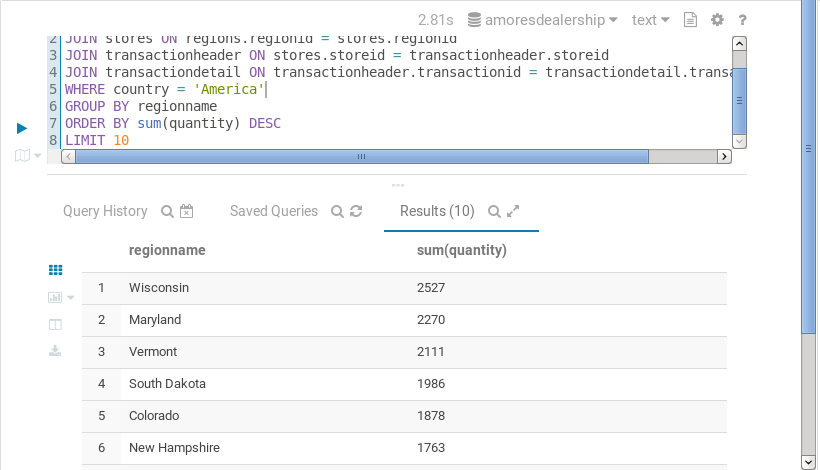
WHERE country = 'America'

GROUP BY regionname

ORDER BY sum(quantity) DESC

LIMIT 10

Proof:



1. Show **top 3 stores** which earned the **most gross profit** **in 2019.**

SELECT storename, sum(transactiondetail.quantity\*cars.carprice)FROM stores

JOIN transactionheader ON stores.storeid = transactionheader.storeid

JOIN transactiondetail ON transactiondetail.transactionid = transactiondetail.transactionid

JOIN cars ON cars.carid = transactiondetail.carid

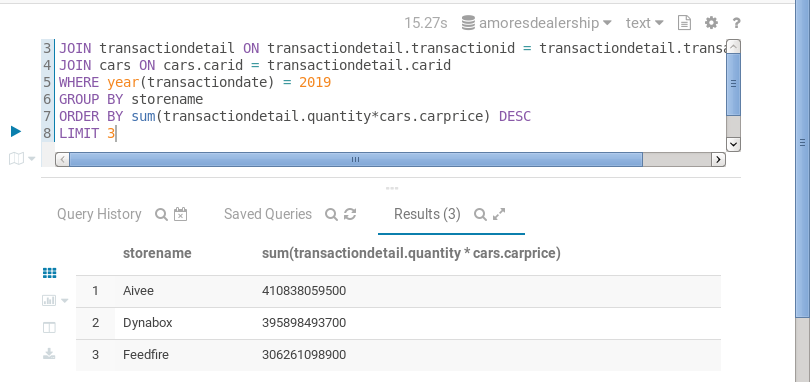
WHERE year(transactiondate) = 2019

GROUP BY storename

ORDER BY sum(transactiondetail.quantity\*cars.carprice) DESC

LIMIT 3

Proof:



1. Show **car manufacturers** that have **sell cars more than the average car manufacturer** for cars that are made **after 2010.**

SELECT m.manufacturername, SUM(td.quantity) as TotalSum

FROM manufacturers m

JOIN cars c ON m.manufacturerid = c.manufacturerid

JOIN transactiondetail td ON c.carid = td.carid

JOIN

(SELECT td.carid, SUM(td.quantity) as TotalSum

FROM transactiondetail td

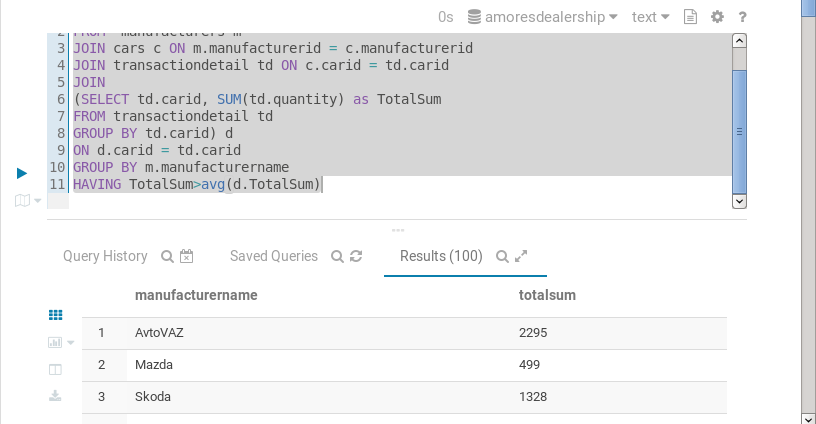
GROUP BY td.carid) d

ON d.carid = td.carid

GROUP BY m.manufacturername

HAVING TotalSum>avg(d.TotalSum)

Proof:



1. Show **store and total cars sold** for each store who did **more than 5 transactions** within year **2018-2019** and **determine how much investment** will be given based on the total car sold on the **year 2018-2019.**

|  |  |
| --- | --- |
| Total Car Sold | Investment |
| > 200 | 100.000.000 |
| 100 - 200 | 50.000.000 |
| 30 – 99 | 20.000.000 |
| 20 – 29 | 10.000.000 |
| < 20 | 0 |

SELECT storename,sum(quantity) as TotalBought,

(

CASE

WHEN sum(quantity)>200 THEN 100000000

WHEN sum(quantity)>100 THEN 50000000

WHEN sum(quantity)>30 THEN 20000000

WHEN sum(quantity)>20 THEN 10000000

ELSE 0

END

) AS Investment

FROM stores

JOIN transactionheader ON stores.storeid = transactionheader.storeid

AND (year(transactiondate)>2017 AND year(transactiondate)<2019)

JOIN transactiondetail ON transactionheader.transactionid = transactiondetail.transactionid

JOIN cars ON cars.carid = transactiondetail.carid

GROUP BY storename

HAVING count(transactiondetail.transactionid)>5

Proof:

