# [C:\Users\jwoo5\AppData\Local\Temp\templateTermTutorial.html](http://www.calstatela.edu/centers/hipic) CIS4580 Term Project Tutorial https://avatars2.githubusercontent.com/u/4156894?v=3&s=100

California State University, Los Angeles

#### Authors: Andrew Rosalino; Andrew Tran; Edgar Manukian; Thomas Lew; Brian Sangalang

#### Instructor: [Jongwook Woo](https://www.linkedin.com/in/jongwook-woo-7081a85)

**Used Car Data Analysis from CarGurus using Hive, Pig, and PowerBI**

**Objectives**

* Upload data and Analyze via wget
* Create a table and test data via Hive
* Clean and test data via Pig
* Store and Export data into Hadoop then Local Machine
* Visualizations and Analysis using PowerBI

**Platform Specifications**

* Cluster Version: 2.7.1
* Cluster Number of Nodes: 3
* Memory Size: 180 GB
* OCPUs: 12

Step 1: Upload and Analyze Used Car Data in Hive

**1.** Open a shell terminal – git bash, minty, putty etc- and run the ssh command to connect to the Oracle Cloud.

ssh **atran80**@129.150.64.74

**2.** In your shell terminal, you have to download the following file “archive.zip”.

$ wget <https://cis4560.s3.us-east-2.amazonaws.com/archive.zip>

**Alternatively, the dataset is also hosted on Kaggle:** <https://www.kaggle.com/ananaymital/us-used-cars-dataset> **(2GB Compressed, 8GB Uncompressed)**

**3.** Unzip “archive.zip”

$ unzip archive.zip

**Archive: archive.zip**

**inflating: used\_cars\_data.csv**

**4.** Create car folder of your HDFS:

$ hdfs dfs -mkdir car

**5.** Upload used\_cars\_data.csv file to car folder of HDFS:

$ hdfs dfs -put used\_cars\_data.csv car/

$ hdfs dfs -ls car/

**Found 1 Items**

**-rw-r--r-- 2 atran80 hdfs 9980208148 2020-11-27 23:47 car/used\_cars\_data.csv**

**6**. Run the following HDFS command to make your *beeline* command works.

hdfs dfs -chmod -R o+w car/

Make sure if it has a permission for others:

-bash-4.1$ hdfs dfs -ls car/used\_cars\_data.csv

**-rw-r--rw- 2 atran80 hdfs 9980208148 2020-11-27 23:47 car/used\_cars\_data.csv**

**7**. Open an another terminal to run **beeline** and run “**!connect….”**.

beeline

!connect jdbc:hive2://bigdai-nov-bdcsce-1:2181,bigdai-nov-bdcsce-2:2181,bigdai-nov-bdcsce-3:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2?tez.queue.name=interactive bdcsce\_admin

**8**. **Call** your database of Hive as follows:

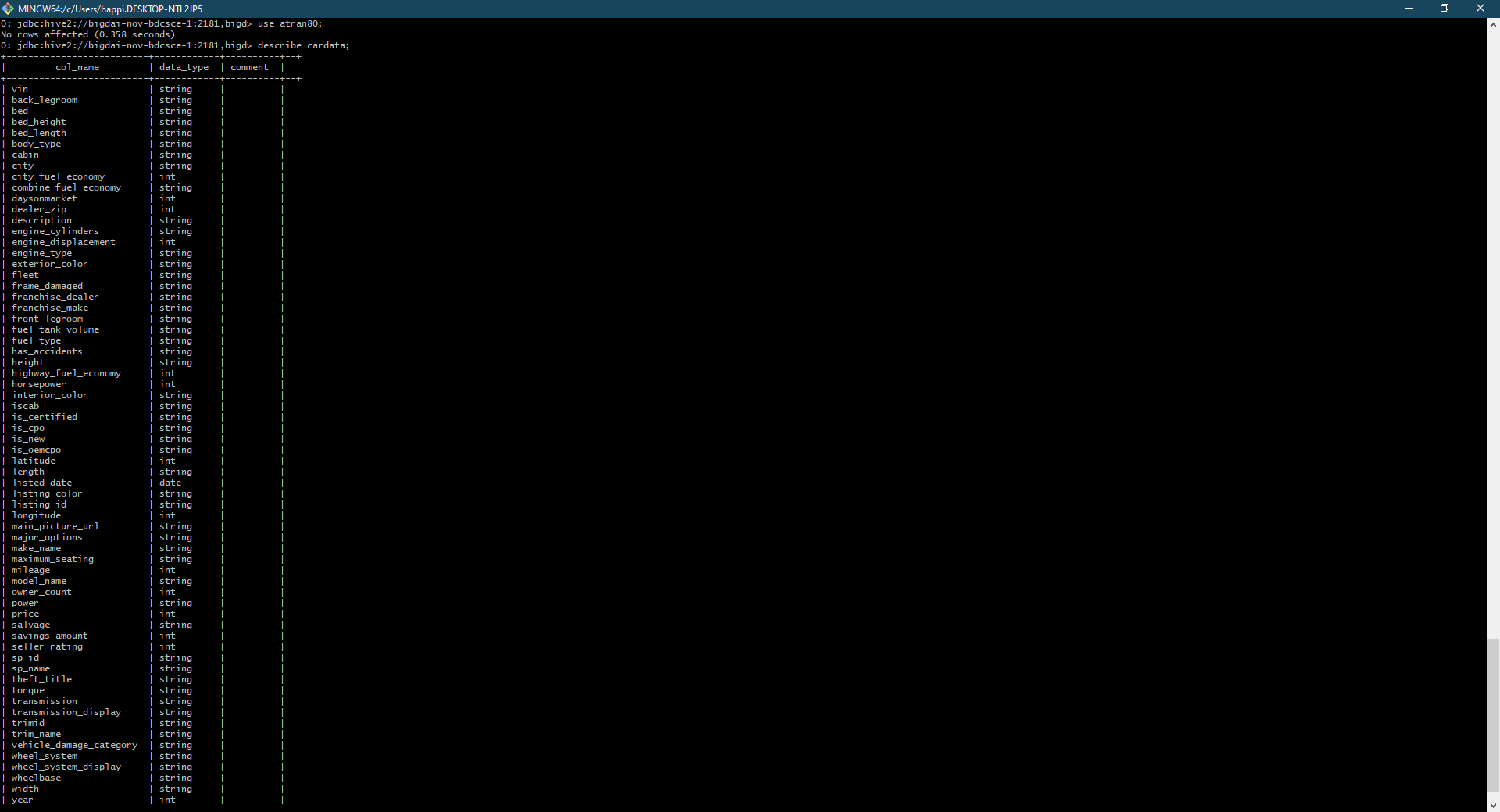
0: jdbc:hive2://bigdai-nov-bdcsce-1:2181,bigd> use **atran80**;

**9**. Then, you need to create **cardata** table using the following Hive command – Alternatively, you can open Hive environment at the shell terminal to run it. See the option “'skip.header.line.count' = '2'”, which is to skip the header line of CSV file that you don’t need.

|  |
| --- |
| DROP TABLE IF EXISTS cardata;  --create table cardata on space-delimited website log data  CREATE EXTERNAL TABLE IF NOT EXISTS cardata(vin string, back\_legroom string, bed string, bed\_height string, bed\_length string,  body\_type string, cabin string, city string, city\_fuel\_economy int, combine\_fuel\_economy string, daysonmarket int, dealer\_zip int,  description string, engine\_cylinders string, engine\_displacement int, engine\_type string, exterior\_color string, fleet string,  frame\_damaged string, franchise\_dealer string, franchise\_make string, front\_legroom string, fuel\_tank\_volume string, fuel\_type string,  has\_accidents string, height string, highway\_fuel\_economy int, horsepower int, interior\_color string, isCab string, is\_certified string,  is\_cpo string, is\_new string, is\_oemcpo string, latitude int, length string, listed\_date date, listing\_color string, listing\_id string,  longitude int, main\_picture\_url string, major\_options string, make\_name string, maximum\_seating string, mileage int, model\_name string,  owner\_count int, power string, price int, salvage string, savings\_amount int, seller\_rating int, sp\_id string, sp\_name string,  theft\_title string, torque string, transmission string, transmission\_display string, trimId string, trim\_name string,  vehicle\_damage\_category string, wheel\_system string, wheel\_system\_display string, wheelbase string, width string, year int )  ROW FORMAT DELIMITED FIELDS TERMINATED BY ' '  STORED AS TEXTFILE LOCATION '/user/atran80/car/'  TBLPROPERTIES ('skip.header.line.count'='2'); |

**10.** Use describe to see if table creation was successful:

Describe cardata;



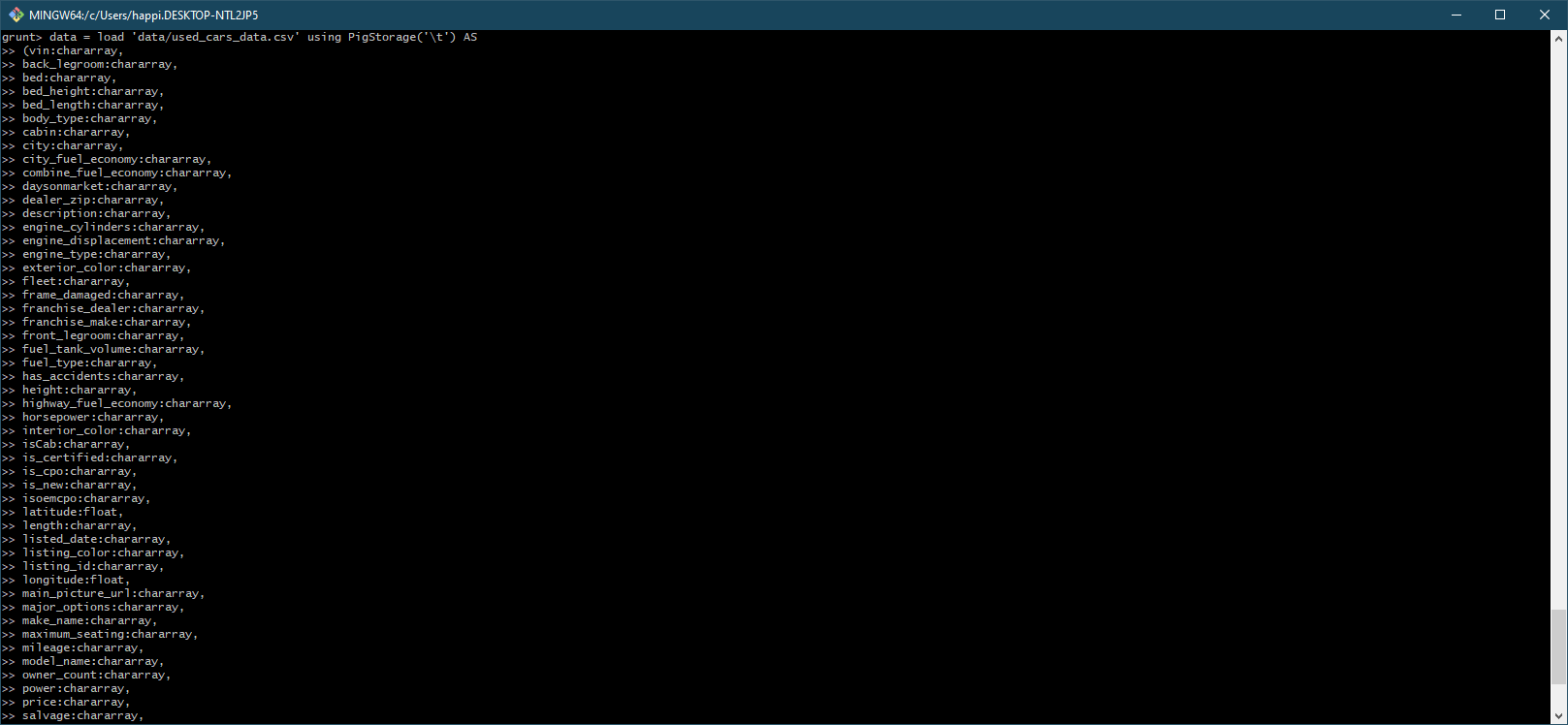
**11.** Run SQL queries to test the dataset:

Select \* from caradata limit 10;

Select make\_name, model\_name, price from cardata order by price desc limit 10;

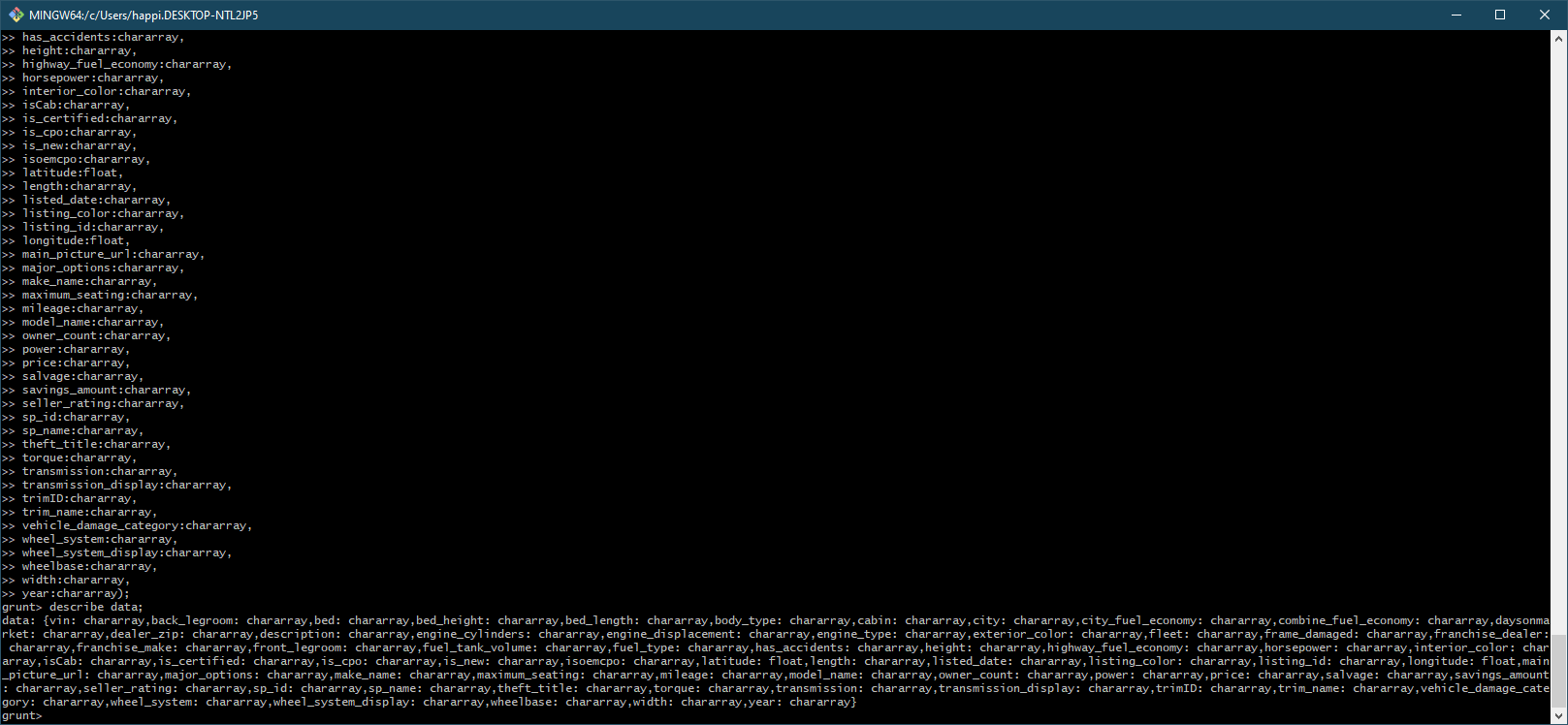
Step 2: Use Pig to Store and Clean the data

**1.** Enter “pig” into the git terminal and load our data and define the columns (The code is too long to be included here but can be copy and pasted from our Github account):



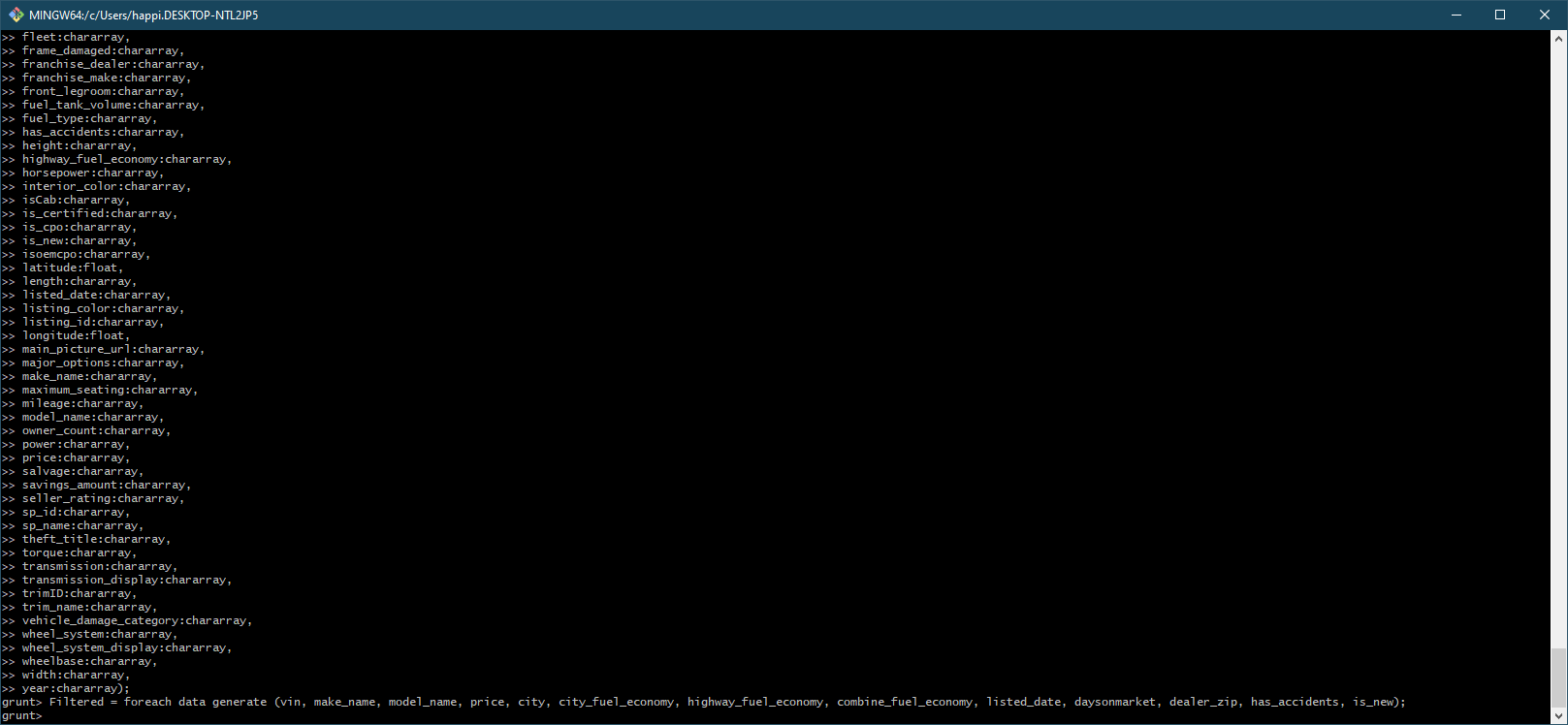
**2.** Describe the data to see if it was successful.

Describe cardata;



**3.** Filter the data so that only relevant columns are shown.

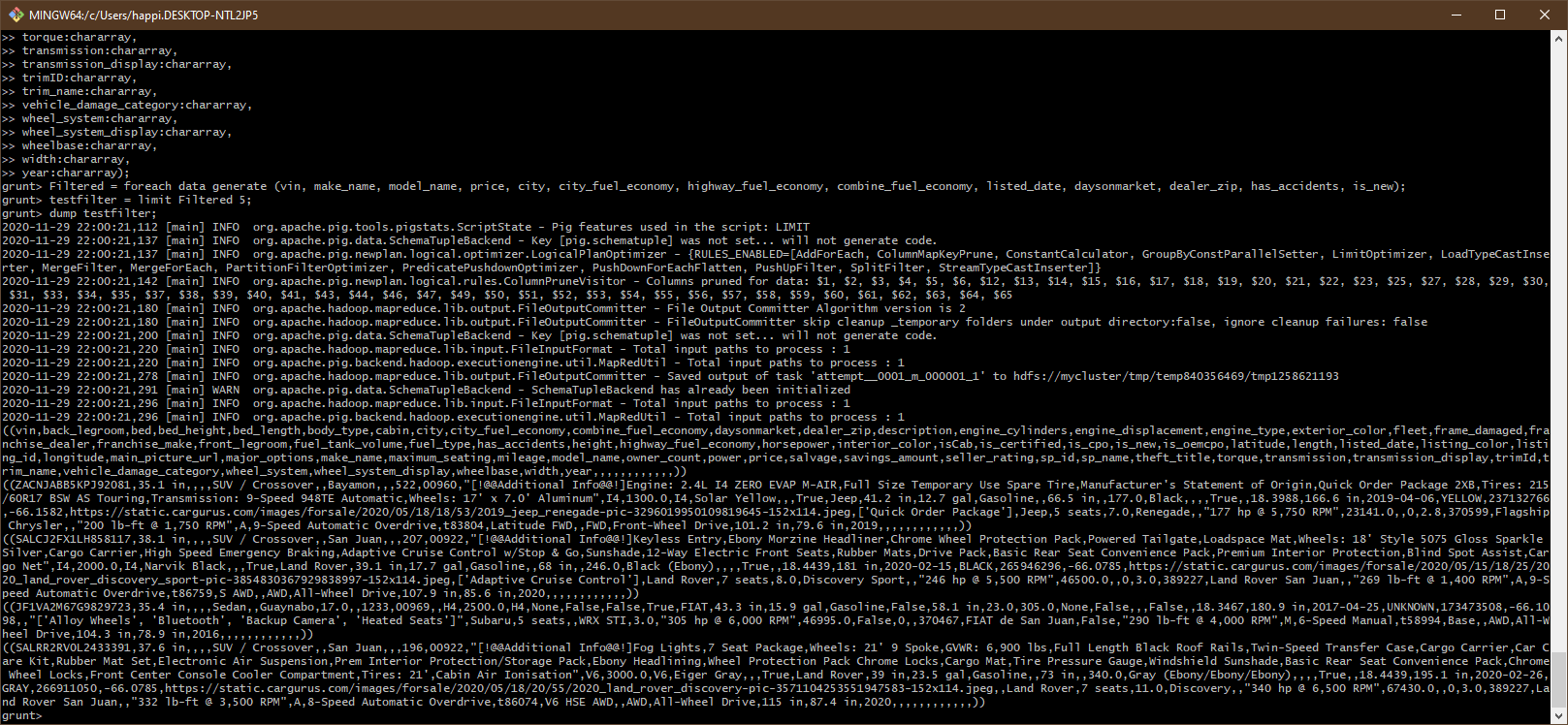
Filtered = foreach data generate (vin, make\_name, model\_name, price, city, city\_fuel\_economy, highway\_fuel\_economy, combine\_fuel\_economy, listed\_date, daysonmarket, dealer\_zip, has\_accidents, is\_new);



**4.** Test the data using a DUMP command with a limit on it

testfilter = limit Filtered 5;

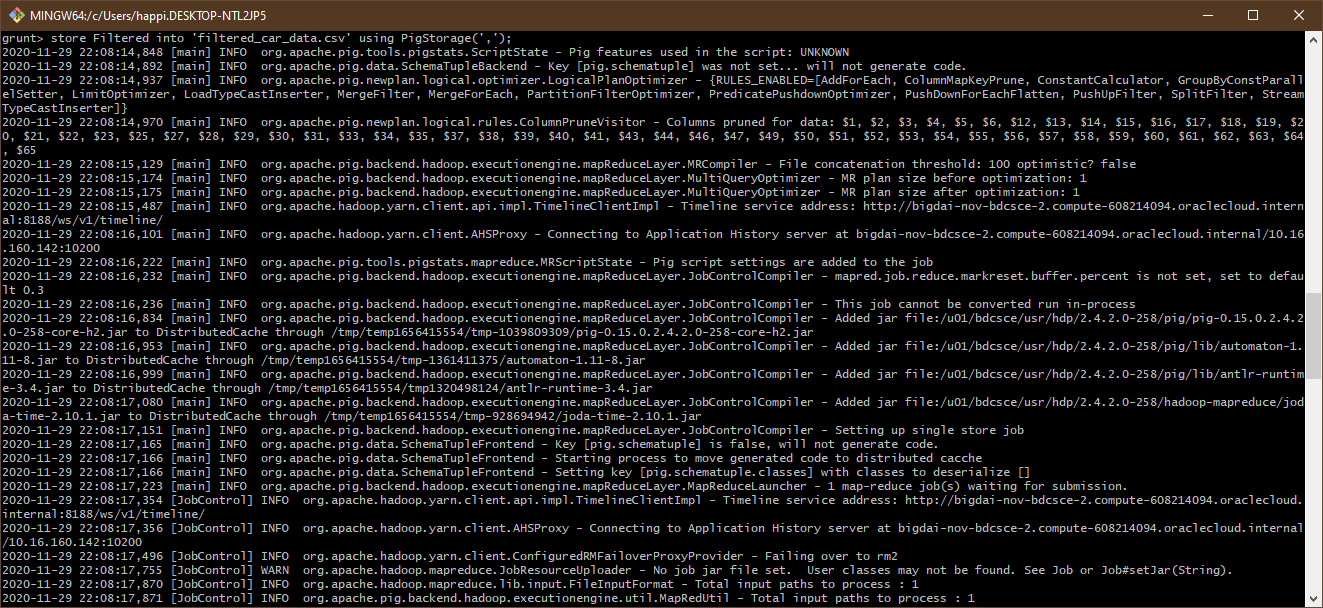
DUMP testfilter;



The data does not look very pretty but we can still store it and use PowerBI to analyze it.

**5.** Store Filtered into ‘filtered\_car\_data.csv’ using PigStorage(‘,’);

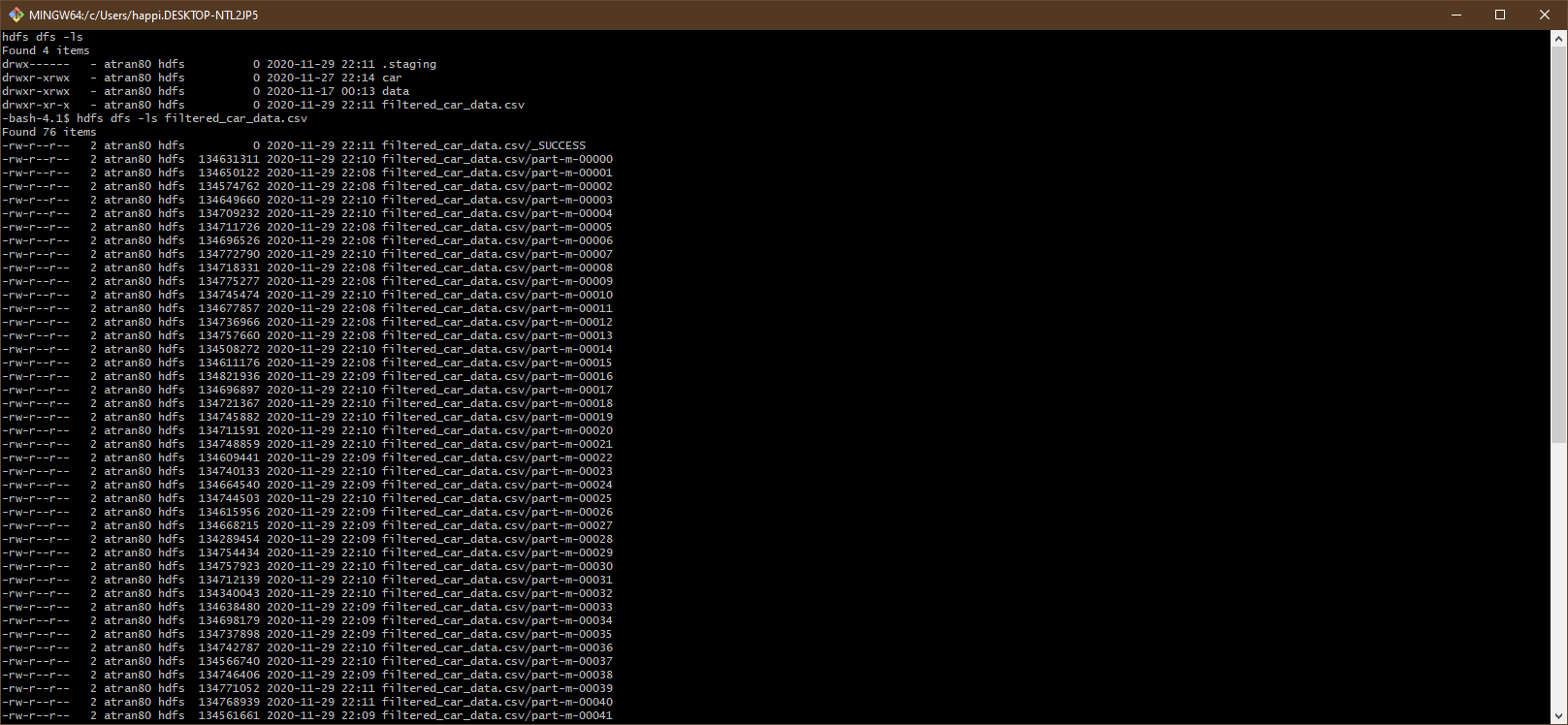
store Filtered into ‘filtered\_car\_data’ using PigStorage(‘,’);



**6.** Check if the file is present and has data in it in Hadoop.

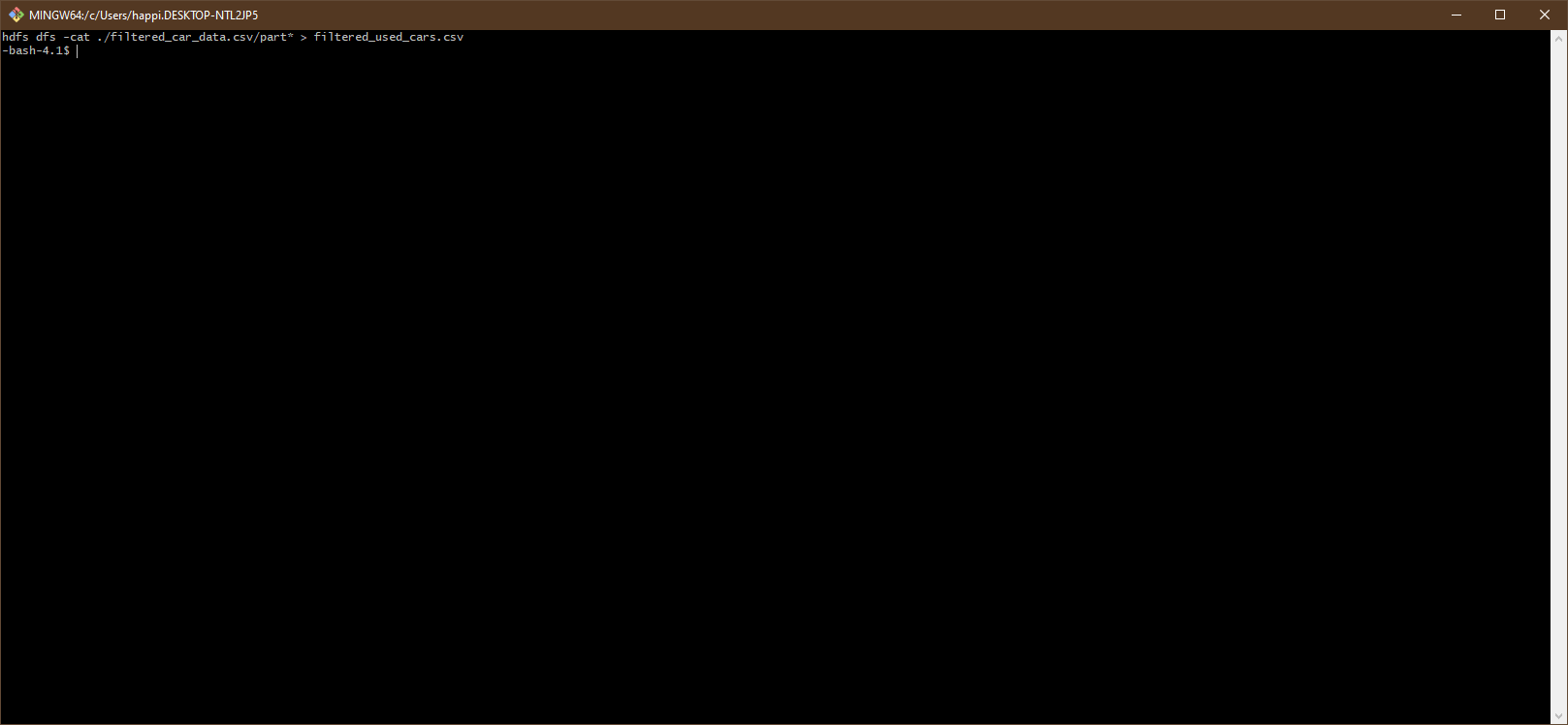
hdfs dfs -ls

hdfs dfs -ls filtered\_car\_data.csv



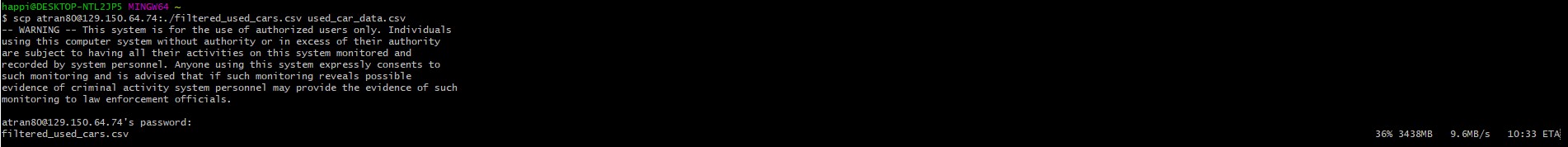
**7.** Convert the Pig results into a CSV.

hdfs dfs -cat ./filtered\_car\_data.csv/part\* > filtered\_used\_cars.csv

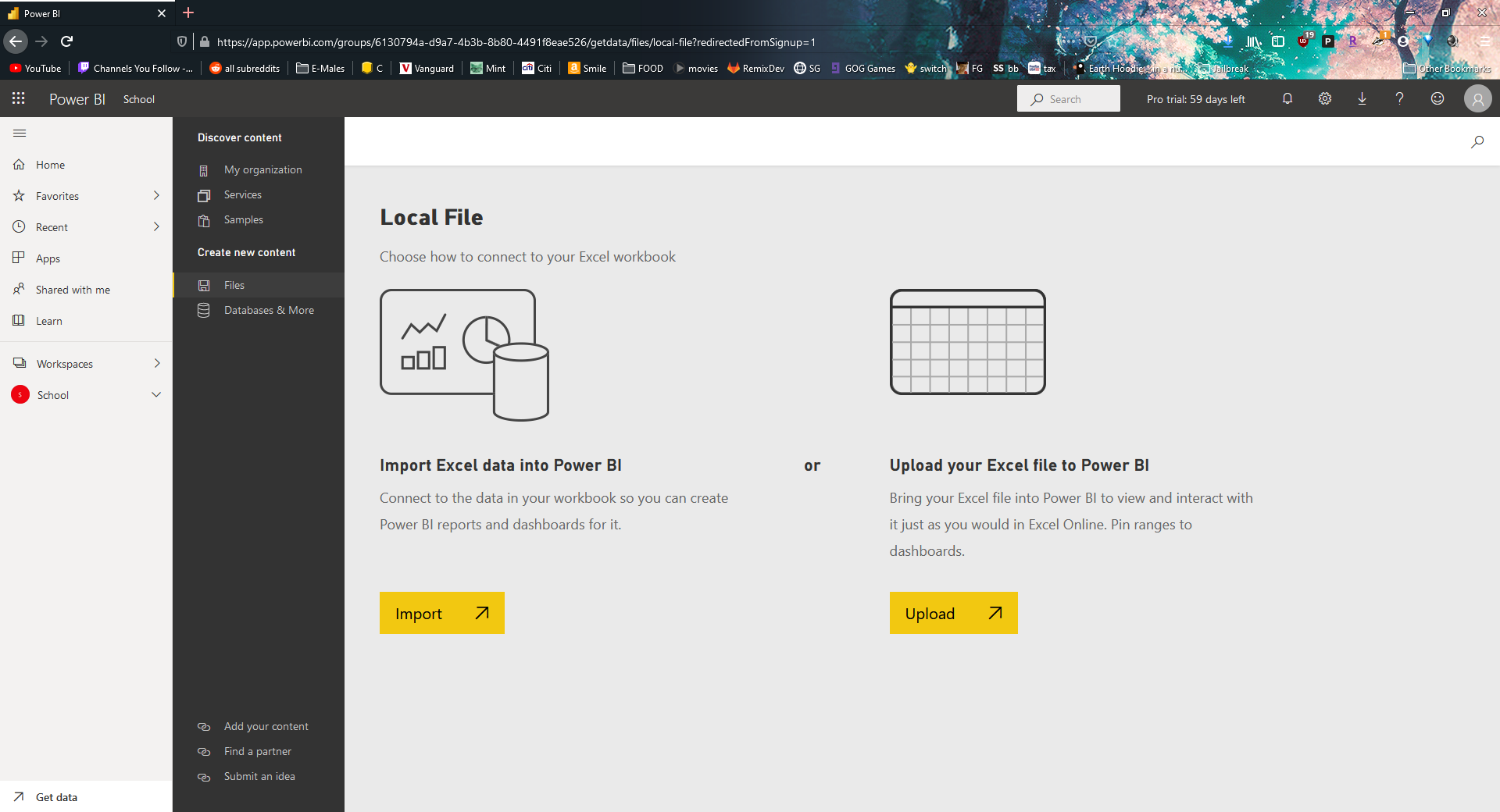


**8.** Download the newly created and filtered csv using SCP

scp atran80@129.150.64.74:./filtered\_used\_cars.csv used\_car\_data.csv



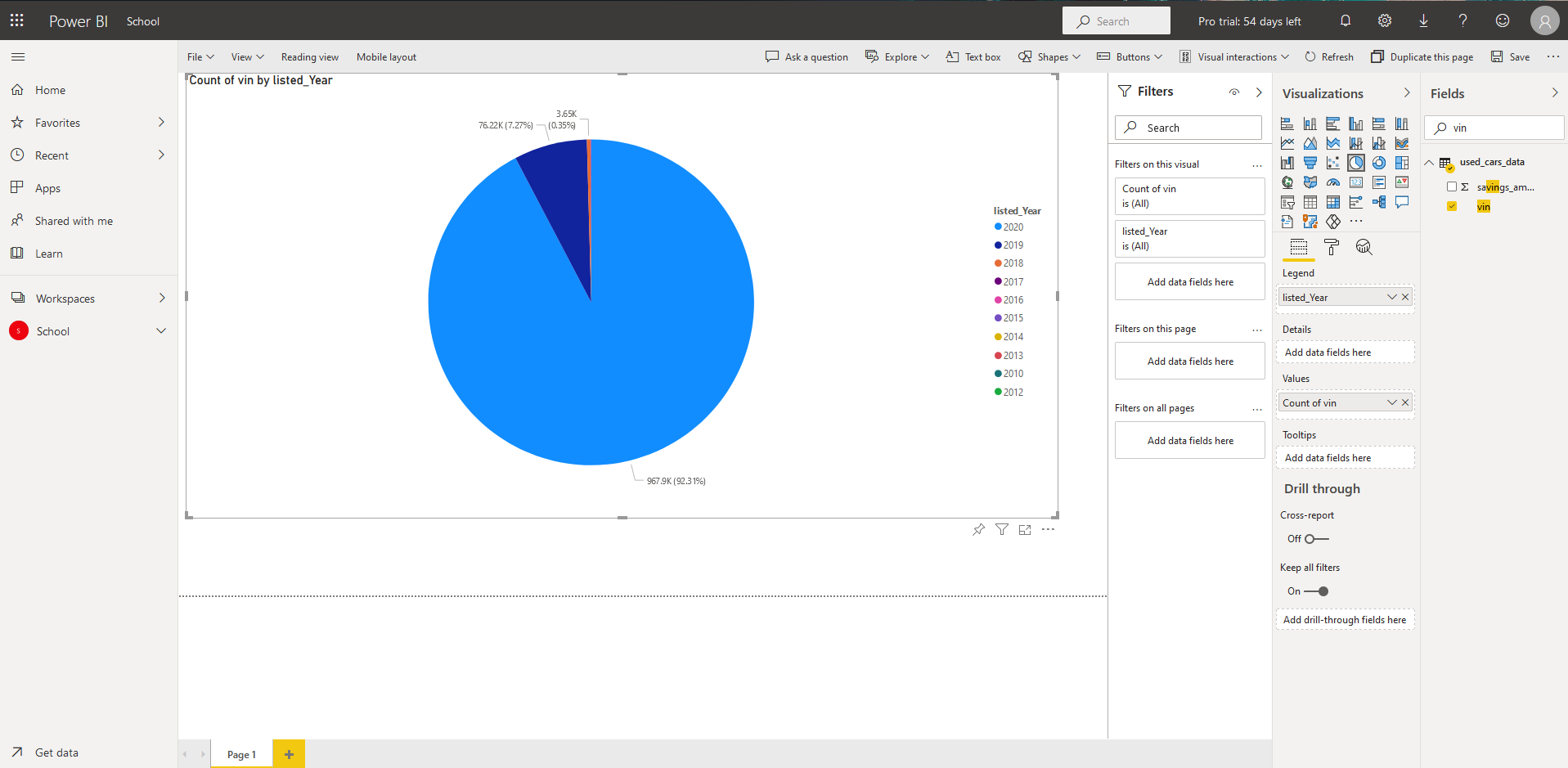
Step 3: Visualization using PowerBI

**1.** Sign into PowerBI and upload your new dataset

**2.** Create visualizations

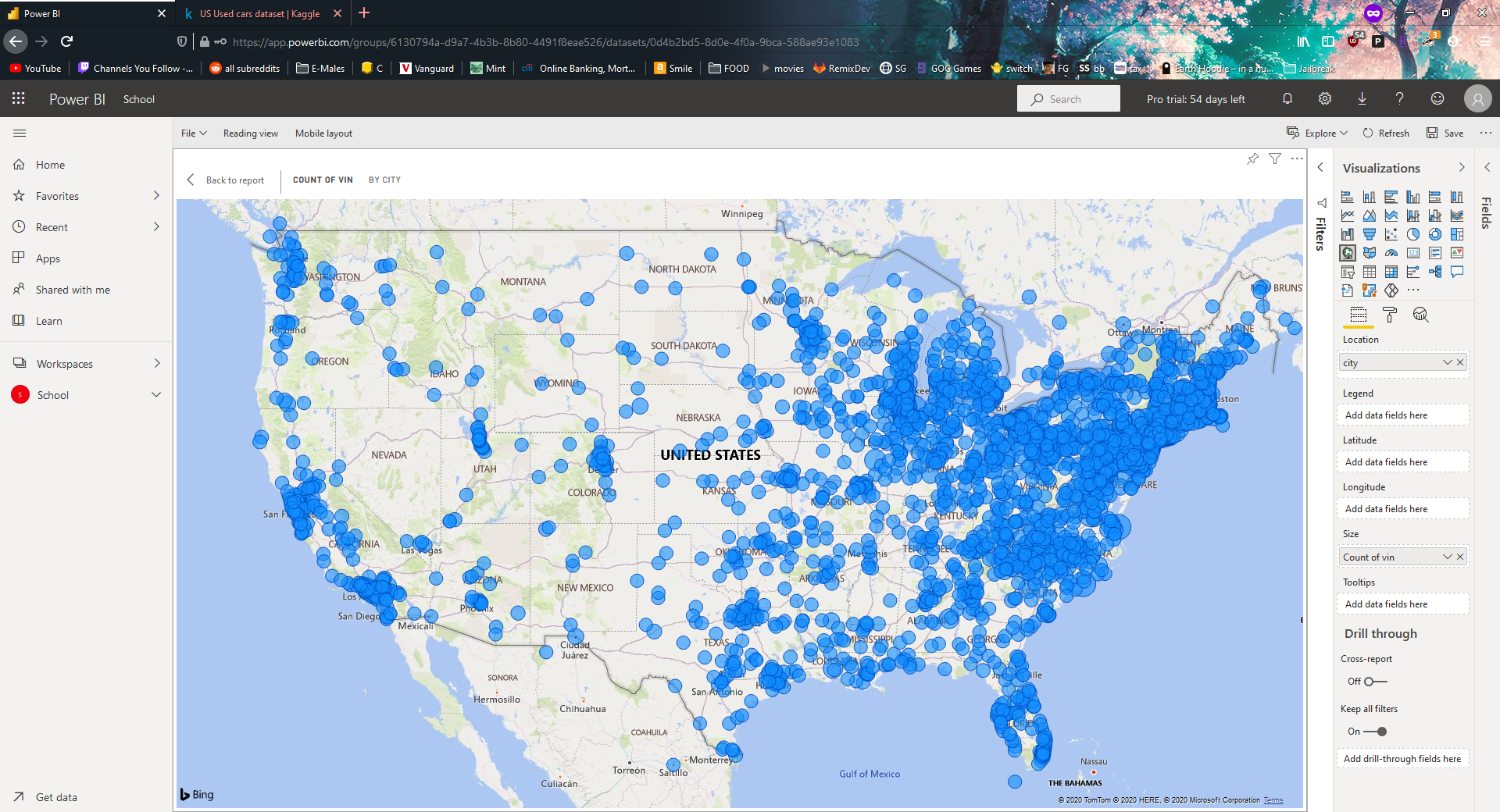
Pie Chart showing amount of listed vehicles by year

Select Pie chart icon -> Drag Listed year to legend and vin to values -> change vin to count of vin



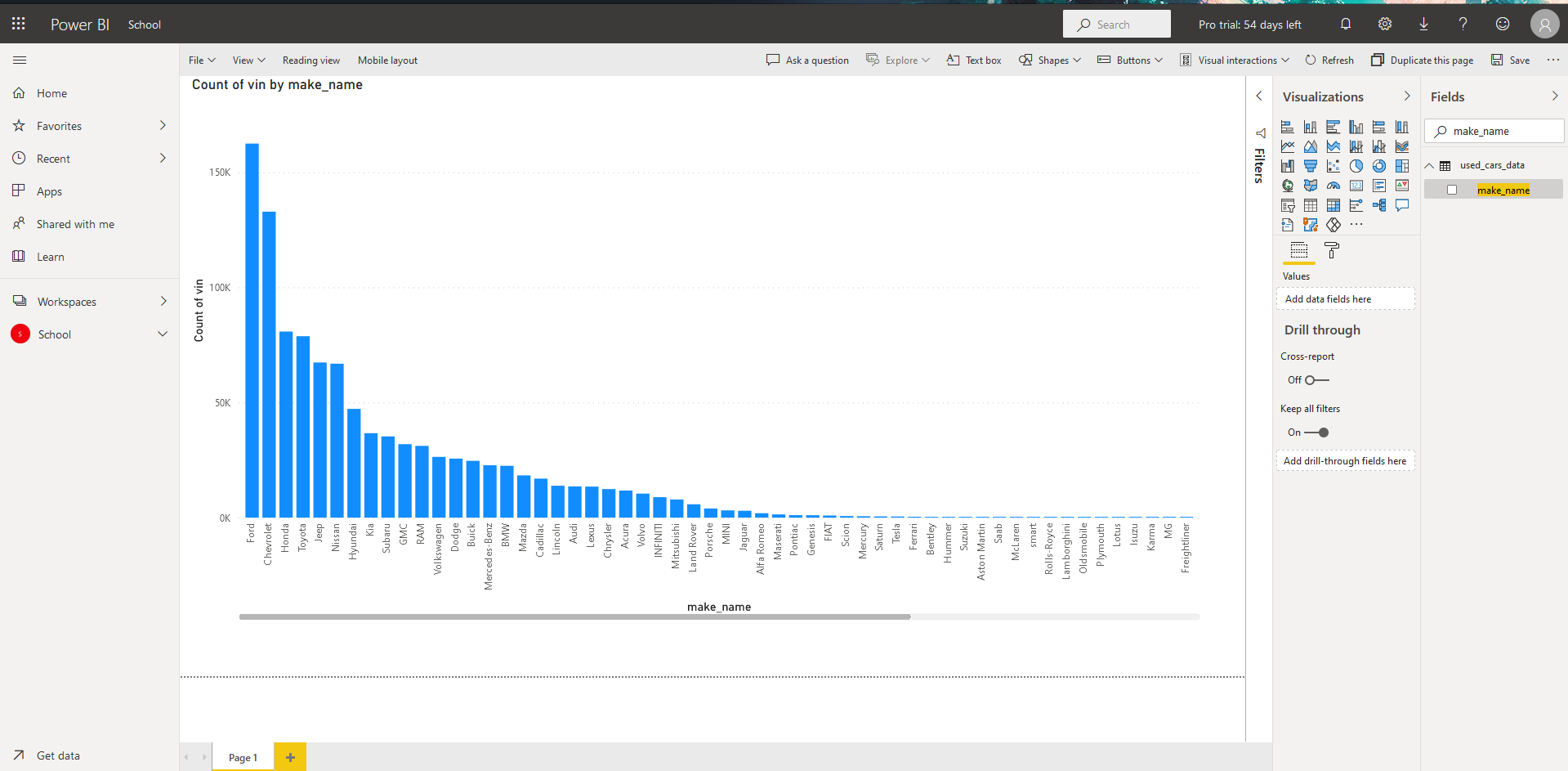
Map detailing number of listed cars in each United States city.

Select Map icon -> Drag city to location -> drag vin to size -> change vin to count of vin



Most popular used car make names.

Drag make name and vin onto the canvas -> change vin to count of vin



References

* 1. CarGurus used cars dataset: <https://www.kaggle.com/ananaymital/us-used-cars-dataset>
  2. Github: <https://github.com/aylmaokai/CIS4560>
  3. AWS Link: <https://cis4560.s3.us-east-2.amazonaws.com/archive.zip>