CIS5200 Term Project Tutorial

Authors: Samyuktha Muralidharan;Sanjana Boddireddy ;Shilpa Konde Deshmukh;Verusca Aimie Capuno

Instructor: Jongwook Woo

Date: 05/17/2020

Lab Tutorial

Samyuktha Muralidharan (smurali2@calstatela.edu)

Sanjana Boddireddy (sboddir@calstatela.edu)

Shilpa Konde Deshmukh(skonded@calstatela.edu)

Verusca Aimie Capuno (vcapuno@calstatela.edu)

05/17/2020

Airbnb Data Analysis using Hive

Objective

Airbnb is an online marketplace that connects people who want to rent out their homes with people who are looking for accommodations in that locale. Travelers can often book an Airbnb for less than the cost of a hotel room and provided with more amenities. It has become a strong competitor against the hotel industry.

The project aims at performing data analysis and sheds insights on Airbnb Data using Hive and presenting visualizations using Tableau and Excel 3D Maps.

We analyze the data on the below factors,

- → Determining the number of Airbnb listings worldwide and in the various States of USA.
- → Seasonality in Demand for listings and various property types in New York City and LA County over years.
- → Distribution and the number of listings in NYC and LA County.
- → Revenue of various neighborhoods in NYC and LA County.
- → Determining the average price of listings and variation of price with property types in several neighborhoods of NYC.
- → Supply and Demand for various bedroom/accommodation configurations in LA County.
- → Analyzing what it takes to become a Super host.

Introduction

This analysis about Airbnb data gives Airbnb an idea about how they can help their hosts improve their bookings and that in turn increases the value of Airbnb. We are using Hive for the analysis and presenting the visualization in Tableau.

In this lab you will learn to,

- → Load data from local Desktop to Linux shell.
- → Download and upload files from/to HDFS.
- → Create tables in Hive and use HiveQL to query the data.
- → Visualize the data in Tableau and Excel 3D Maps.

Platform Specifications

- → Hadoop Cluster version: Hadoop 2.7.1.2.4.2.0-258
- → Cluster number of nodes: 3
- → CPU speed: 2.20 GHz
- → HDFS capacity: 1003.6 GB
- → Memory: 241.821 GB
- → Hive version: Apache Hive Version 1.2.1000.2.4.2.0-258

Prerequisites

- → Microsoft Excel 2010,2013 or 2016 installed.
- → Excel 3-D Maps enabled.
- → Tableau Desktop 2020.1 installed for data visualization.

Downloading Dataset files

1.We need to download the Airbnb Ratings, New York & Los Angeles Listings, Airbnb Reviews files from the Kaggle website. Below is the URL of the Kaggle website where the 4 files are located,

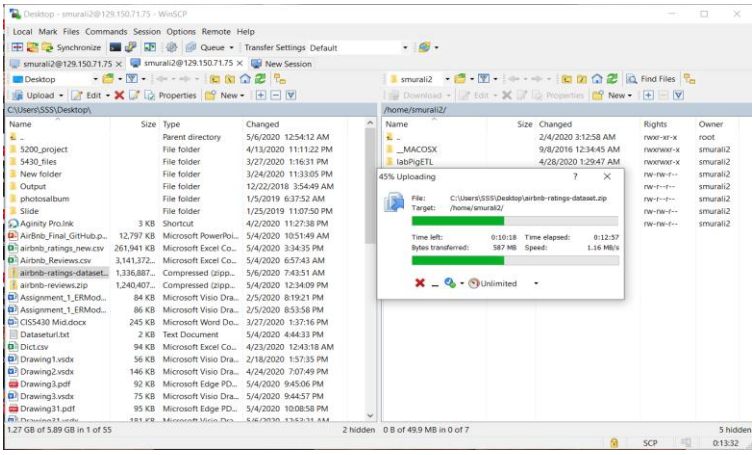
https://www.kaggle.com/samyukthamurali/airbnb-ratings-dataset

Click on the Download button and a zip file **airbnb-ratings-dataset.zip** is downloaded into your local system.

2.We need to upload this zip file to remotely located Oracle BDCE Hadoop cluster. We are using WinSCP to transfer the zip file to the remote node of Oracle BDCE.

Connect to the host using smurali2@129.150.71.75 with your username and password.

Note: You must give your own username and also change the ip address if you have a different one.



3.In the shell terminal type in the *ssh* command to connect to the Hadoop cluster.

We need to remotely access Oracle BDCE Hadoop cluster using **Putty** or **Terminal** (in Windows, LINUX or MAC) using the following *ssh* command with your username and ip address.

Samyukthas-MacBook-Pro:~ samyukthamuralidharan$ ssh smurali2@129.150.71.75

```
Samyukthas-MacBook-Pro:~ samyukthamuralidharan$ ssh smurali2@129.150.71.75
-- WARNING -- This system is for the use of authorized users only. Individuals
using this computer system without authority or in excess of their authority
are subject to having all their activities on this system monitored and
recorded by system personnel. Anyone using this system expressly consents to
such monitoring and is advised that if such monitoring reveals possible
evidence of criminal activity system personnel may provide the evidence of such
monitoring to law enforcement officials.

smurali2@129.150.71.75's password:
Last login: Wed May  6 07:16:42 2020 from oc-129-150-71-75.compute.oraclecloud.com
-bash-4.1$
```

Note: Do not forget to change to your username and the IP address if you have a different one.

This output shows that you are successfully connected.

4.We need to check if the airbnb-ratings-dataset.zip file is transferred to the remote machine using *ls* command

-bash-4.2$ ls

airbnb-ratings-dataset.zip

5.Unzip this file using the unzip command,

unzip airbnb-ratings-dataset.zip

```
-bash-4.2$ unzip airbnb-ratings-dataset.zip
Archive:  airbnb-ratings-dataset.zip
  inflating: LA_Listings.csv
  inflating: NY_Listings.csv
  inflating: airbnb_ratings_new.csv
  inflating: airbnb-reviews.csv
```

6.Check if the files are present in the current directory using the *ls* command,

-bash-4.2$ ls

airbnb_ratings_new.csv

NY_Listings.csv

LA_Listings.csv

airbnb-reviews.csv

7.We need to download the following files using *wget* shell command in the remote node of Oracle BDCE.

New York Review Dates file:

-bash-4.2$ wget -O NY_Review_Dates.csv http://data.insideairbnb.com/united-states/ny/new-york-city/2020-03-13/visualisations/reviews.csv

Los Angeles Review Dates file:

-bash-4.2$ wget -O LA_Review_Dates.csv http://data.insideairbnb.com/united-states/ca/los-angeles/2020-03-13/visualisations/reviews.csv

List the current directory to check if the files are downloaded successfully using the command *ls*.

-bash-4.1$ ls

NY_Review_Dates.csv

LA_Review_Dates.csv

Using *ls –al* command,

```
-bash-4.2$ ls -al
-rw-r--r--.  1 smurali2 smurali2 268227239 May  7 06:47 airbnb_ratings_new.csv
-rw-rw-r--  1 smurali2 smurali2 3216764625 May 7 06:44 airbnb-reviews.csv
-rw-r--r--.  1 smurali2 smurali2  22394381 May  7 07:04 LA_Listings.csv
-rw-rw-r--.  1 smurali2 smurali2  26685254 Mar 16 03:31 LA_Review_Dates.csv
-rw-r--r--.  1 smurali2 smurali2  28041975 May  7 07:05 NY_Listings.csv
-rw-rw-r--.  1 smurali2 smurali2  24760723 Mar 15 19:54 NY_Review_Dates.csv
```

8.Once the files are available, we need to create a directory in HDFS called 'airbnb_dataset'.

hdfs dfs -mkdir airbnb_dataset

Create a sub-directory under airbnb_dataset as 'airbnb_ratings'

hdfs dfs -mkdir airbnb_dataset/airbnb_ratings

Put the airbnb_ratings_new.csv file from the home directory to airbnb_dataset/airbnb_ratings directory.

-bash-4.1$ hdfs dfs -put airbnb_ratings_new.csv airbnb_dataset/airbnb_ratings

To check if the file is uploaded successfully,use the below command,

-bash-4.1$ hdfs dfs -ls airbnb_dataset/airbnb_ratings

Found 1 items

-rw-r--r--   2 smurali2 hdfs 268227239 2020-05-07 07:03
airbnb_dataset/airbnb_ratings/airbnb_ratings_new.csv

9.Create a sub-directory named 'airbnb_reviews' under airbnb_dataset

hdfs dfs -mkdir airbnb_dataset/airbnb_reviews

Put the airbnb-reviews.csv file from the home directory to airbnb_dataset/airbnb_reviews directory.

-bash-4.1$ hdfs dfs -put airbnb-reviews.csv airbnb_dataset/airbnb_reviews

To check if files are uploaded successfully use the below command.

-bash-4.1 hdfs dfs –ls airbnb_dataset/airbnb_reviews

Found 1 items

-rw-r--r-- 2 smurali2 hdfs 3216764625 2020-05-07 06:44 airbnb_dataset/airbnb_reviews/airbnb-reviews.csv

10.Create a sub-directory named 'LA_Listings' under airbnb_dataset

-bash-4.2$ hdfs dfs -mkdir airbnb_dataset/LA_Listings

Put the LA_Listings.csv file into the LA_Listings directory using the below command,

-bash-4.2$ hdfs dfs -put LA_Listings.csv airbnb_dataset/LA_Listings

Check if the file is uploaded successfully using the below command,

-bash-4.2$ hdfs dfs -ls airbnb_dataset/LA_Listings

Found 1 items

-rw-r--r-- 2 smurali2 hdfs 22394381 2020-05-07 07:07 airbnb_dataset/LA_Listings/LA_Listings.csv

11.Similarly follow the above 3 steps to create folders for the remaining csv files and upload the files to their respective folders.

-bash-4.2$ hdfs dfs -mkdir airbnb_dataset/**NY_Listings**

-bash-4.2$ hdfs dfs -put **NY_Listings.csv** airbnb_dataset/NY_Listings

-bash-4.2$ hdfs dfs -ls airbnb_dataset/NY_Listings

Found 1 items

-rw-r--r-- 2 smurali2 hdfs 28041975 2020-05-07 07:08 airbnb_dataset/NY_Listings/NY_Listings.csv

-bash-4.2$ hdfs dfs -mkdir airbnb_dataset/**LA_Review_Dates**

-bash-4.2$ hdfs dfs -put **LA_Review_Dates.csv** airbnb_dataset/LA_Review_Dates

-bash-4.2$ hdfs dfs -ls airbnb_dataset/LA_Review_Dates

Found 1 items

-rw-r--r-- 2 smurali2 hdfs 26685254 2020-05-07 07:16 airbnb_dataset/LA_Review_Dates/LA_Review_Dates.csv

-bash-4.2$ hdfs dfs -mkdir airbnb_dataset/**NY_Review_Dates**

-bash-4.2$ hdfs dfs -put **NY_Review_Dates.csv** airbnb_dataset/NY_Review_Dates

-bash-4.2$ hdfs dfs -ls airbnb_dataset/NY_Review_Dates

Found 1 items

-rw-r--r-- 2 smurali2 hdfs 24760723 2020-05-07 07:17 airbnb_dataset/NY_Review_Dates/NY_Review_Dates.csv

12.**Run the following HDFS command to make the beeline command work for the new folder 'airbnb_dataset'.**

-bash-4.1$ hdfs dfs -chmod -R o+w airbnb_dataset

-bash-4.2$ hdfs dfs -ls

Found 5 items

...

drwxr-xrwx - smurali2 hdfs 0 2020-05-07 07:17 airbnb_dataset

Creating Hive Tables to Query Airbnb Data

1.Open another terminal and connect it to Oracle BDCE cluster using *ssh*. Then you need to connect to Beeline Command Line Interface as follows,

-bash-4.2$ beeline

beeline> !connect jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1-bdcsce-2:2181,bigdai1-bdcsce-3:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2?tez.queue.name=interactive bdcsce_admin

```
-bash-4.1$ beeline
WARNING: Use "yarn jar" to launch YARN applications.
Beeline version 1.2.1000.2.4.2.0-258 by Apache Hive
beeline> !connect jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1-bdcsce-2:2181,bigdai1-bdcsce-3:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiv
eserver2?tez.queue.name=interactive bdcsce_admin
Connecting to jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1-bdcsce-2:2181,bigdai1-bdcsce-3:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveser
ver2?tez.queue.name=interactive
Enter password for jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1-bdcsce-2:2181,bigdai1-bdcsce-3:2181/;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hi
veserver2?tez.queue.name=interactive:
Connected to: Apache Hive (version 1.2.1000.2.4.2.0-258)
Driver: Hive JDBC (version 1.2.1000.2.4.2.0-258)
Transaction isolation: TRANSACTION_REPEATABLE_READ
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1>
```

Note: Hit **Enter key** when you are asked for password.

2.Now you must create your own database with your username to separate your tables from other users' tables. For example, the user should run the following command,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> CREATE DATABASE IF NOT EXISTS smurali2;

No rows affected (0.276 seconds)

Use the below command to check if the database is created or not.

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW DATABASES;

```
| sboddir        |
| sedulak        |
| skashif        |
| skonded        |
| skundu         |
| smareed        |
| smurali2       |
| sshinde        |
| sshinde6       |
```

3.Now you should use the database you created for the analysis using the following command,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> use smurali2;

No rows affected (0.295 seconds)

Note: Don't forget to change to your username instead of smurali2.

The following Hive statement creates an external table for **Airbnb_Ratings**. It is advisable to create external tables as they preserve data in the original file format, while allowing Hive to perform queries against the data within the file.

Table 1:

In Beeline CLI you need to copy and paste the following HiveQL code to create an external table 'Airbnb_Ratings'.

```
CREATE EXTERNAL TABLE Airbnb_Ratings(Listing_ID BigInt,Name String,Host_ID BigInt,Host_Name String,Host_Response_Rate String,Superhost Boolean,Host_Listings_Count Int,Street String,City String,Neighbourhood String,State String,Country String,Latitude Decimal(10,8),Longitude Decimal(11,8),Property_Type String,Room_Type String,Accommodates Int,Bathrooms Int,Bedrooms Int,Amenities String,Price Int,Min_Nights Int,Max_Nights Int,Availability_365 Int,Date_Last_Scraped String,No_Of_Reviews Int,Last_Review_Date String,Review_Scores_Rating Int,Review_Scores_Accuracy Int,Review_Scores_Cleanliness Int,Review_Scores_Checkin Int,Review_Scores_Communication Int,Review_Scores_Location Int,Review_Scores_Value Int,Reviews_Per_Month Int)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/smurali2/airbnb_dataset/airbnb_ratings'
TBLPROPERTIES('skip.header.line.count'='1');
```

After executing this query check if Airbnb_Ratings table is created successfully by using the below query.

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW TABLES;

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW TABLES;
+--------------------------+--+
|         tab_name         |
+--------------------------+--+
| airbnb_ratings           |
```

Now we can query the contents of 'Airbnb_Ratings' table,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1>SELECT * FROM Airbnb_Ratings LIMIT 2;

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM airbnb_ratings LIMIT 2;
+------------------------------------------------------------------------------------------------------------------------------+
| airbnb_ratings.listing_id  |         airbnb_ratings.name          | airbnb_ratings.host_id  | airbnb_ratings.host_name  | airbnb_ratings.host_response_rate  | airbnb_ratings.su
perhost  | airbnb_ratings.host_listings_count  |            airbnb_ratings.street               | airbnb_ratings.city  | airbnb_ratings.neighbourhood  | airbnb_ratings.state
| airbnb_ratings.country  | airbnb_ratings.latitude  | airbnb_ratings.longitude  | airbnb_ratings.property_type  | airbnb_ratings.room_type  | airbnb_ratings.accommodates  | air
bnb_ratings.bathrooms  | airbnb_ratings.bedrooms  |                                                                                          airbnb_ratings.amenities
                                                | airbnb_ratings.price  | airbnb_ratings.min_nights  | airbnb_ratings.max_nights  | airbnb_ratings.availability
_365  | airbnb_ratings.date_last_scraped  | airbnb_ratings.no_of_reviews  | airbnb_ratings.last_review_date  | airbnb_ratings.review_scores_rating  | airbnb_ratings.review_score
s_accuracy  | airbnb_ratings.review_scores_cleanliness  | airbnb_ratings.review_scores_checkin  | airbnb_ratings.review_scores_communication  | airbnb_ratings.review_scores_loca
tion  | airbnb_ratings.review_scores_value  | airbnb_ratings.reviews_per_month  |
+------------------------------------------------------------------------------------------------------------------------------+
| 13588243                  | Designer Urban Living - 2BR 1200sq  | 4386046                 | Chris                     | 90%                                | false
        | 3                              | Lau Li Street| 0| Hong Kong Island 0| Hong Kong  | 0                     | Wan Chai                     | Hong Kong Island
| Hong Kong               | 22.28544814              | 114.1917323               | Apartment                     | Entire home/apt           | 4                            | 1
                       | 2                       | TV;Internet;Wireless Internet;Air Conditioning;Kitchen;Elevator in Building;Heating;Washer;Dryer;Smoke Detector;Essentials;24
-Hour Check-in;Hangers;Hair Dryer;Iron;Laptop Friendly Workspace  | NULL                  | 3                          | 1125                       | 358
      | 8/7/16                            | 0                             |                                  | NULL                                 | NULL
            | NULL                                      | NULL                                  | NULL                                         | NULL
      | NULL                                | NULL                           |
| 5534229                   | A 2 Passi da San Pietro             | 28697142                | Veronica                  | 100%                               | false
        | 5                              | 00165| Rm 00165| Italy                | 165                   | XIII Aurelia                 | Rm
| Italy                   | 41.89587833              | 12.4544301                | Bed & Breakfast               | Private room              | 3                            | 1
                       | 1                       | TV;Internet;Wireless Internet;Air conditioning;Smoking allowed;Pets allowed;Heating;First aid kit;Fire extinguisher;Essential
s;Shampoo                                                         | 74                    | 1                          | 1125                       | 365
      | 5/8/17                            | 2                             | 8/29/15                          | 90                                   | 9
            | 10                                        | 8                                     | 8                                            | 9
      | 9                                   | 0                              |
```

We can also see the structure of the table using the below query,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> DESCRIBE Airbnb_Ratings;

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> DESCRIBE Airbnb_Ratings;
+------------------------------+----------------+----------+--+
|           col_name           |   data_type    | comment  |
+------------------------------+----------------+----------+--+
| listing_id                   | bigint         |          |
| name                         | string         |          |
| host_id                      | bigint         |          |
| host_name                    | string         |          |
| host_response_rate           | string         |          |
| superhost                    | boolean        |          |
| host_listings_count          | int            |          |
| street                       | string         |          |
| city                         | string         |          |
| neighbourhood                | string         |          |
| state                        | string         |          |
| country                      | string         |          |
| latitude                     | decimal(10,8)  |          |
| longitude                    | decimal(11,8)  |          |
| property_type                | string         |          |
| room_type                    | string         |          |
| accommodates                 | int            |          |
| bathrooms                    | int            |          |
| bedrooms                     | int            |          |
| amenities                    | string         |          |
| price                        | int            |          |
| min_nights                   | int            |          |
| max_nights                   | int            |          |
| availability_365             | int            |          |
| date_last_scraped            | string         |          |
| no_of_reviews                | int            |          |
| last_review_date             | string         |          |
| review_scores_rating         | int            |          |
| review_scores_accuracy       | int            |          |
| review_scores_cleanliness    | int            |          |
| review_scores_checkin        | int            |          |
| review_scores_communication  | int            |          |
| review_scores_location       | int            |          |
| review_scores_value          | int            |          |
| reviews_per_month            | int            |          |
+------------------------------+----------------+----------+--+
35 rows selected (0.219 seconds)
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1>
```

Table 2:

In Beeline CLI you need to copy and paste the following HiveQL code to create an external table Bnb_Rev.

```
CREATE EXTERNAL TABLE Bnb_Rev(Listing_ID BigInt,ID BigInt,Date_Of_Review Date,Reviewer_ID
BigInt,Reviewer_Name String,Comments String)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY '\u0059'
STORED AS TEXTFILE LOCATION '/user/smurali2/airbnb_dataset/airbnb_reviews'
TBLPROPERTIES ('skip.header.line.count'='1');
```

We are creating a table with the same schema as the 'Bnb_Rev' table called Airbnb_Reviews and populating it with records from the table 'Bnb_Rev' with Listing ID not null.

CREATE TABLE Airbnb_Reviews like Bnb_Rev;

INSERT OVERWRITE TABLE Airbnb_Reviews SELECT * FROM Bnb_Rev WHERE Listing_ID IS NOT NULL;

We need to check if the table is successfully created using the below query,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW TABLES;

```
+-----------------+
|    tab_name     |
+-----------------+
| airbnb_ratings  |
| airbnb_reviews  |
```

Now we can query the contents of the Airbnb_Reviews table,
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM Airbnb_Reviews LIMIT 2;

```
+--------------------------+-------------------+-------------------------------+---------------------------+-----------------------------
+-----------------------------------------------------+
| airbnb_reviews.listing_id  | airbnb_reviews.id  | airbnb_reviews.date_of_review  | airbnb_reviews.reviewer_id  | airbnb_reviews.reviewer_name
|              airbnb_reviews.comments                |
+--------------------------+-------------------+-------------------------------+---------------------------+-----------------------------
+-----------------------------------------------------+
| 488835                   | 104522929         | 2016-09-27                    | 66003975                  | Carole
| First time using Airbnb and couldn't be happier! Nuriel is a wonderful host, welcoming, warm, friendly and always happy to help. The room was p
erfect, just as described, and in a great location. I would thoroughly recommend Nuriels, and I will definitely be back! |
| 549036                   | 37751194          | 2015-07-10                    | 33501858                  | Gabi
| Muy bien ubicado. Las fotos y la descrpción corresponden con lo que uno se encuentra en el lugar. |
+--------------------------+-------------------+-------------------------------+---------------------------+-----------------------------
+-----------------------------------------------------+
```

We can see the structure of the Airbnb_Reviews table using the below query,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> DESCRIBE Airbnb_Reviews;

```
+-----------------+-----------+----------+
|    col_name     | data_type | comment  |
+-----------------+-----------+----------+
| listing_id      | bigint    |          |
| id              | bigint    |          |
| date_of_review  | date      |          |
| reviewer_id     | bigint    |          |
| reviewer_name   | string    |          |
| comments        | string    |          |
+-----------------+-----------+----------+
6 rows selected (0.384 seconds)
```

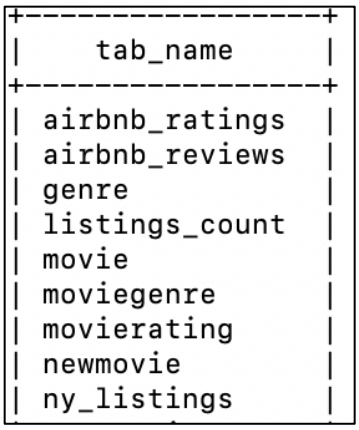
Table 3:

Create an external table 'NY_Listings' by copy pasting the following Hive QL code in Beeline CLI,

```
CREATE EXTERNAL TABLE NY_Listings(Listing_ID BigInt,Name String,Host_ID BigInt,Host_Name
String,Host_Response_Rate String,Superhost Boolean,Host_Listings_Count Int,Street String,City
String,Neighbourhood String,State String,Country String,Latitude Decimal(10,8),Longitude
Decimal(11,8),Property_Type String,Room_Type String,Accommodates Int,Bathrooms Int,Bedrooms
Int,Amenities String,Price Int,Min_Nights Int,Max_Nights Int,Availability_365 Int,Date_Last_Scraped
String,No_Of_Reviews Int,Last_Review_Date String,Review_Scores_Rating Int,Review_Scores_Accuracy
Int,Review_Scores_Cleanliness Int,Review_Scores_Checkin Int,Review_Scores_Communication
Int,Review_Scores_Location Int,Review_Scores_Value Int,Reviews_Per_Month Int)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/smurali2/airbnb_dataset/NY_Listings'
TBLPROPERTIES('skip.header.line.count'='1');
```

→ **This table consists of the details of all the listings in New York City.**

We need to check if the table is successfully created using the below query,
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW TABLES;



tab_name
airbnb_ratings
airbnb_reviews
genre
listings_count
movie
moviegenre
movierating
newmovie
ny_listings

We can query the contents of NY_Listings table using the below query,

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Listings LIMIT 3;

ny_listings.listing_id	ny_listings.name	ny_listings.host_id	ny_listings.host_name	ny_listings.host_response_rate	ny_listings.superhost	ny_listings.host_listings_count	ny_listings.street	ny_listings.city	ny_listings.neighbourhood	ny_listings.state	ny_listings.country	ny_listings.latitude	ny_listings.longitude	ny_listings.property_type	ny_listings.room_type	ny_listings.accommodates	ny_listings.bathrooms	ny_listings.bedrooms	ny_listings.amenities	ny_listings.price	ny_listings.min_nights	ny_listings.max_nights	ny_listings.availability_365	ny_listings.date_last_scraped	ny_listings.no_of_reviews	ny_listings.last_review_date	ny_listings.review_scores_rating	ny_listings.review_scores_accuracy	ny_listings.review_scores_cleanliness	ny_listings.review_scores_checkin	ny_listings.review_scores_communication	ny_listings.review_scores_location	ny_listings.review_scores_value	ny_listings.reviews_per_month
2515	Sunny Private Room	16286162	Pat	1	false	4	Bronx| NY| United States	Bronx	Allerton	NY	United States	40.86688872	-73.85775618	House	Private room	1	1	1	["Cable TV","Internet","Wireless Internet","Kitchen","Free parking on premises","Buzzer/wireless intercom","Heating","Smoke detector","Carbon monoxide detector","First aid kit","Safety card","Fire extinguisher","Essentials","Shampoo","Lock on bedroom door","Hangers","Hair dryer","Iron","Laptop friendly workspace","translation missing: en.hosting_amenity_49","translation missing: en.hosting_amenity_50","Private living room","Private entrance","Hot water","Bed linens","Extra pillows and blankets","Microwave","Refrigerator","Dishes and silverware","Cooking basics","Stove","Garden or backyard","Luggage dropoff allowed","Long term stays allowed","Cleaning before checkout"]	43	2	30	276	NULL	66	NULL	96	10	9	10	10	9	10	1
2539	Comfy bedroom minutes to Manhattan	44260966	Alicia	1	false	1	Bronx| NY| United States	Bronx	Soundview	NY	United States	40.82939197	-73.86513684	House	Private room	1	1	1	["TV","Internet","Wireless Internet","Air conditioning","Kitchen","Heating","Smoke detector","Carbon monoxide detector","First aid kit","Safety card","Fire extinguisher","Essentials","Shampoo","Lock on bedroom door","Hangers","Hair dryer","Iron","Laptop friendly workspace","translation missing: en.hosting_amenity_49","translation missing: en.hosting_amenity_50"]	28	2	31	345	NULL	38	NULL	89	10	9	9	10	9	9	1
2595	Spacious RM in Private Home; Backyard/Near Metro	105394139	Ciprian	1	false	16	Bronx| NY| United States	Bronx	Fordham	NY	United States	40.86913892	-73.89509569	House	Private room	4	3	2	["Internet","Wireless Internet","Air conditioning","Kitchen","Heating","Smoke detector","Carbon monoxide detector","First aid kit","Essentials","Lock on bedroom door","Hangers","Hair dryer","Iron","Self Check-In","Keypad"]	80	3	31	355	NULL	18	NULL	90	9	9	10	9	9	3

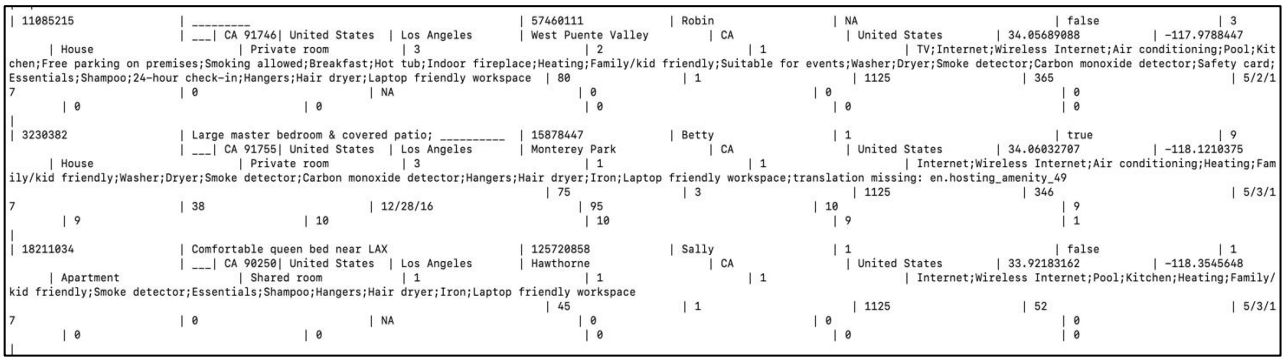
Table 4:

Copy and paste the following Hive QL code in Beeline CLI to create an External table 'LA_Listings',

```
CREATE EXTERNAL TABLE LA_Listings(Listing_ID BigInt,Name String,Host_ID BigInt,Host_Name
String,Host_Response_Rate String,Superhost Boolean,Host_Listings_Count Int,Street String,City
String,Neighbourhood String,State String,Country String,Latitude Decimal(10,8),Longitude
Decimal(11,8),Property_Type String,Room_Type String,Accommodates Int,Bathrooms Int,Bedrooms
Int,Amenities String,Price Int,Min_Nights Int,Max_Nights Int,Availability_365 Int,Date_Last_Scraped
String,No_Of_Reviews Int,Last_Review_Date String,Review_Scores_Rating Int,Review_Scores_Accuracy
Int,Review_Scores_Cleanliness Int,Review_Scores_Checkin Int,Review_Scores_Communication
Int,Review_Scores_Location Int,Review_Scores_Value Int,Reviews_Per_Month Int)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/smurali2/airbnb_dataset/LA_Listings'
TBLPROPERTIES('skip.header.line.count'='1');
```

→ **This table consists of the details of all the listings in Los Angeles County.**

Check if the table is created successfully using the 'SHOW TABLES' command and check the contents of the table using 'SELECT' statement like we did in the previous steps.



Tables 5 and 6:

We are creating 2 external tables **'NY_Review_Dates'** and **'LA_Review_Dates'** to store the Review Dates for the cities New York and Los Angeles respectively. This table has until the most recent review dates.

Using the below two queries to create external tables 'NY_Review_Dates' and 'LA_Review_Dates' respectively,

```
CREATE EXTERNAL TABLE NY_Review_Dates(Listing_ID BigInt,Review_Date Date)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION
'/user/smurali2/airbnb_dataset/NY_Review_Dates'
TBLPROPERTIES('skip.header.line.count'='1');

CREATE EXTERNAL TABLE LA_Review_Dates(Listing_ID BigInt,Review_Date Date)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION
'/user/smurali2/airbnb_dataset/LA_Review_Dates'
TBLPROPERTIES('skip.header.line.count'='1');
```

Check if the tables are created using the SHOW TABLES command,

```
+-------------------+
|     tab_name      |
+-------------------+
| airbnb_ratings    |
| airbnb_reviews    |
| genre             |
| la_listings       |
| la_review_dates   |
| listings_count    |
| movie             |
| moviegenre        |
| movierating       |
| newmovie          |
| ny_listings       |
| ny_review_dates   |
```

Check the contents of the tables using the SELECT command,

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Review_Dates LIMIT 5;
+-----------------------------+-----------------------------+--+
| ny_review_dates.listing_id  | ny_review_dates.review_date  |
+-----------------------------+-----------------------------+--+
| 2060                        | 2008-09-22                  |
| 2595                        | 2009-11-21                  |
| 2595                        | 2009-12-05                  |
| 2595                        | 2009-12-10                  |
| 2595                        | 2010-04-09                  |
+-----------------------------+-----------------------------+--+
5 rows selected (0.236 seconds)
```

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Review_Dates LIMIT 5;
+-----------------------------+-----------------------------+--+
| la_review_dates.listing_id  | la_review_dates.review_date  |
+-----------------------------+-----------------------------+--+
| 109                         | 2011-08-15                  |
| 109                         | 2016-05-15                  |
| 344                         | 2016-06-14                  |
| 344                         | 2016-12-11                  |
| 344                         | 2016-12-30                  |
+-----------------------------+-----------------------------+--+
5 rows selected (0.086 seconds)
```

Analysis 1

1.1 Number of Airbnb Listings in various Countries.

- **We are creating a table 'Listings_Count_World' and adding records to this table using the below CTAS query,**

```
CREATE TABLE Listings_Count_World ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/user/smurali2/Listings_Count_World' AS SELECT COUNT(Listing_ID) AS No_Of_Listings,Country FROM Airbnb_Ratings GROUP BY Country ORDER BY No_Of_listings DESC;
```

- **We can check if the table is created successfully,**

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW TABLES;

```
+----------------------+
|       tab_name       |
+----------------------+
| airbnb_ratings       |
| airbnb_reviews       |
| genre                |
| la_listings          |
| la_review_dates      |
| listings_count_world |
```

- **We can check the contents of the table,**

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM Listings_Count_World;

```
+-------------------------------------+-------------------------------+
| listings_count_world.no_of_listings | listings_count_world.country  |
+-------------------------------------+-------------------------------+
| 143954                              | United States                 |
| 64407                               | United Kingdom                |
| 56562                               | France                        |
| 50697                               | Spain                         |
| 40693                               | Australia                     |
| 33972                               | Canada                        |
| 33145                               | Italy                         |
| 20576                               | Germany                       |
| 20545                               | Denmark                       |
| 18547                               | Netherlands                   |
| 9201                                | Austria                       |
| 7419                                | Belgium                       |
| 6729                                | Ireland                       |
| 6423                                | Hong Kong                     |
| 5127                                | Greece                        |
| 2381                                | Switzerland                   |
| 51                                  | China                         |
| 2                                   |                               |
| 2                                   | Vatican City                  |
| 2                                   | Mexico                        |
| 1                                   | Uruguay                       |
| 1                                   | Cuba                          |
| 1                                   | Vanuatu                       |
+-------------------------------------+-------------------------------+
23 rows selected (0.431 seconds)
```

1.2 Number of Airbnb Listings in various States in United States.

- **We are creating a table 'Listings_Count_US' and adding records to this table using the below CTAS query,**

```
CREATE TABLE Listings_Count_US ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' LOCATION '/user/smurali2/Listings_Count_US' AS SELECT COUNT(Listing_ID) AS No_Of_Listings,State FROM airbnb_ratings WHERE Country='United States' GROUP BY State HAVING No_Of_Listings>100 ORDER BY No_Of_Listings Desc;
```

- **We can check if the table is created successfully,**

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SHOW TABLES;

```
+----------------------+
|       tab_name       |
+----------------------+
| airbnb_ratings       |
| airbnb_reviews       |
| genre                |
| la_listings          |
| la_review_dates      |
| listings_count_us    |
| listings_count_world |
```

- **We can check the contents of the table,**

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM Listings_Count_US;

```
+-----------------------------------+--------------------------+--+
| listings_count_us.no_of_listings  | listings_count_us.state  |
+-----------------------------------+--------------------------+--+
| 49311                             | CA                       |
| 44308                             | NY                       |
| 9662                              | TX                       |
| 7756                              | DC                       |
| 5332                              | TN                       |
| 5307                              | LA                       |
| 5204                              | IL                       |
| 4870                              | MA                       |
| 3917                              | CO                       |
| 3817                              | WA                       |
| 3549                              | OR                       |
| 864                               | NC                       |
+-----------------------------------+--------------------------+--+
12 rows selected (0.078 seconds)
```

- **After creating the tables, we need to download the files into our local system to visualize it in Excel 3D Maps, (a)**

Run the following HDFS shell commands at Beeline CLI to list the files present in **'Listings_Count_World'** and **'Listings_Count_US'** Directories at **'/user/smurali2'** path. A file named '000000_0' is present.

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> dfs -ls /user/smurali2/Listings_Count_World;

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> dfs -ls /user/smurali2/Listings_Count_World;
+-----------------------------------------------------------------------------------------------------------------------------+--+
|                                                  DFS Output                                                                 |
+-----------------------------------------------------------------------------------------------------------------------------+--+
| Found 1 items                                                                                                               |
| -rwxr-xrwx   2 bdcsce_admin hdfs        300 2020-05-08 06:45 /user/smurali2/Listings_Count_World/000000_0                   |
+-----------------------------------------------------------------------------------------------------------------------------+--+
2 rows selected (0.018 seconds)
```

0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> dfs -ls /user/smurali2/Listings_Count_US;

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> dfs -ls /user/smurali2/Listings_Count_US;
+-----------------------------------------------------------------------------------------------------------------------------+--+
|                                                  DFS Output                                                                 |
+-----------------------------------------------------------------------------------------------------------------------------+--+
| Found 1 items                                                                                                               |
| -rwxr-xrwx   2 bdcsce_admin hdfs         97 2020-05-08 07:00 /user/smurali2/Listings_Count_US/000000_0                      |
+-----------------------------------------------------------------------------------------------------------------------------+--+
2 rows selected (0.014 seconds)
```

- **Open another terminal with the Shell CLI and execute the following command to download the HDFS file 000000_0 to your Oracle BDCE master node, as files named 'listings_count_world.csv' and 'listings_count_US.csv' respectively (b)**

Note: Do not forget to use your username instead of 'smurali2'.

-bash-4.2$ hdfs dfs -get /user/smurali2/Listings_Count_World/000000_0 listings_count_world.csv

-bash-4.2$ pwd

/home/smurali2

-bash-4.2$ ls

```
-bash-4.2$ ls
Airbnb_Listings             Airbnb_Reviews.csv  LA_Listings.csv
airbnb-ratings-dataset.zip  dictionary.tsv      LA_Review_Dates.csv
airbnb_ratings_new.csv      genre.java          listings_count_world.csv
```

-bash-4.2$ hdfs dfs -get /user/smurali2/Listings_Count_US/000000_0 listings_count_US.csv

-bash-4.2$ pwd

/home/smurali2

-bash-4.2$ ls

```
-bash-4.2$ ls
Airbnb_Listings             dictionary.tsv      listings_count_US.csv
airbnb-ratings-dataset.zip  genre.java          listings_count_world.csv
```

- **We need to download the files into our local system using SCP (Using Git Bash,Minty,Linux/Mac terminals) (c)**

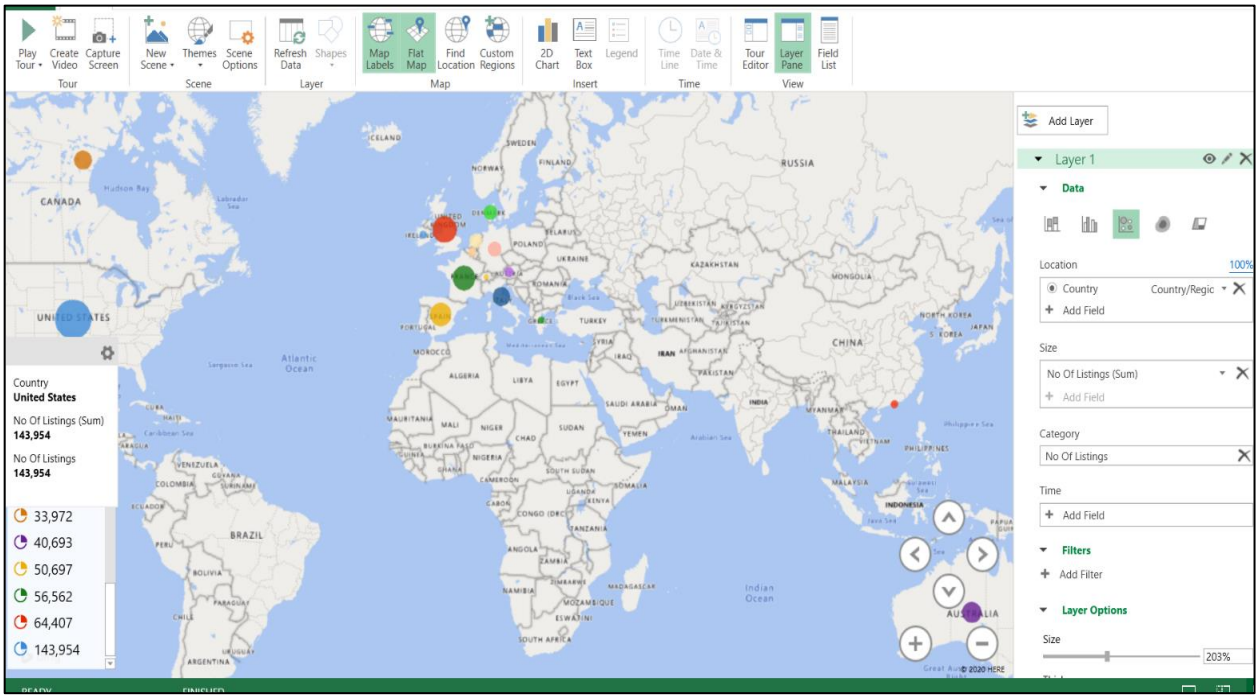
```
Samyukthas-MacBook-Pro:~ samyukthamuralidharan$ scp
smurali2@129.150.71.75:~/listings_count_world.csv .
smurali2@129.150.71.75's password:
listings_count_world.csv                                    100%  300     3.5KB/s
00:00

Samyukthas-MacBook-Pro:~ samyukthamuralidharan$ scp
smurali2@129.150.71.75:~/listings_count_US.csv .
smurali2@129.150.71.75's password:
listings_count_US.csv                                       100%   97     1.2KB/s   +00:00
```

Note: Use your username instead of 'smurali2'.

- **Once the files are downloaded into our local system, we need to visualize it in Excel 3-D Maps.**

a. Open the file 'listings_count_world.csv' in Excel application.
b. Insert headers for the two columns as 'No Of Listings' and 'Country' respectively.
c. Save the file in *.xlsx format.
d. Go to the Insert tab in Excel and select 3-D Maps.
e. Under Location field choose 'Country', under Height choose 'No Of Listings(Sum)' and in Category field choose 'No Of Listings'.
f. Change the visualization to 'Bubble' under Data.
g. We can see the number of listings in each country when you hover over the respective bubble as seen below (Or in the box at the bottom left corner).

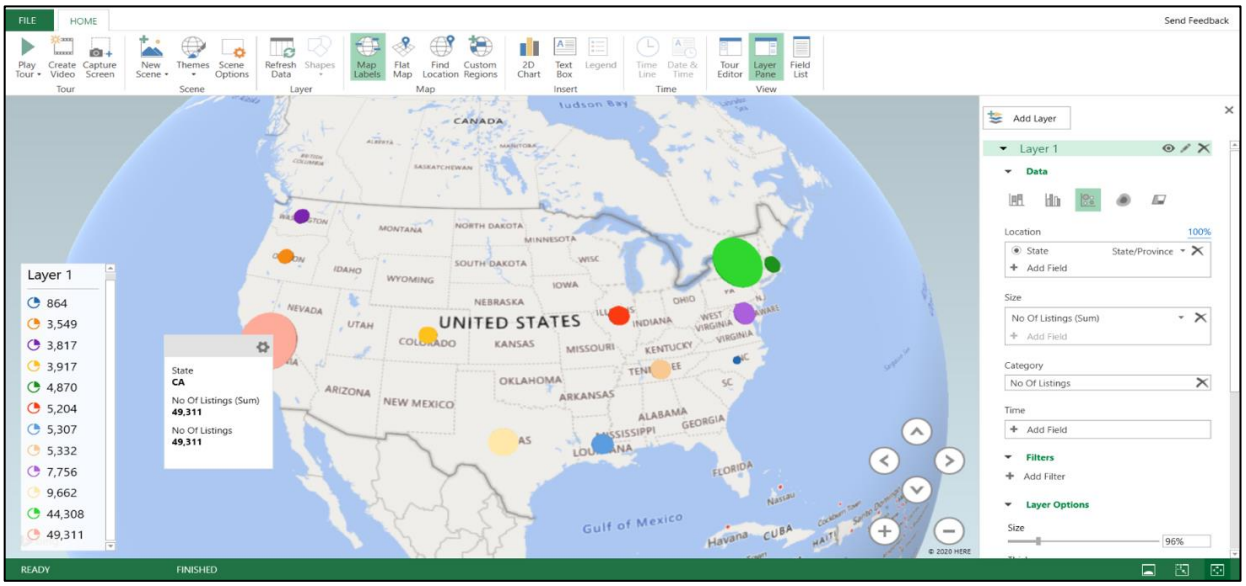


→ **We can see that United States has the greatest number of Airbnb Listings(143,954) followed by United Kingdom (64,407).**

Repeat the above steps for the file 'listings_count_US.csv'.

Note: Insert the headers for the two columns as 'No Of Listings' and 'State' respectively and in Excel 3D-Maps choose **State** under Location field.

The below picture shows the number of listings in various US states.



→ **We see that California and New York States have the greatest number of Airbnb Listings in the US.**

We are considering Los Angeles and New York Cities for the rest of the analysis.

Analysis 2:

Seasonality in Demand:

In this part we are determining the number of bookings in NYC and LA over years.

Note: Due to unavailability of Airbnb booking data we use the Review Dates as a proxy for Booking Dates to determine the demand for Airbnb rentals. Airbnb claims that 60% of their guests review their stay and a review is provided by guests within 2 weeks of their stay.

New York City:

- **We are creating a table 'NY_Dates' to store the Listing_ID and its corresponding review dates for New York City using the CTAS query below,**

```
CREATE TABLE NY_Dates ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/NY_Dates' AS SELECT DISTINCT NY.Listing_ID,NR.Review_Date FROM
NY_Listings NY,NY_Review_Dates NR WHERE NY.Listing_ID=NR.Listing_ID
UNION DISTINCT
SELECT DISTINCT NY.Listing_ID,ar.Date_Of_Review FROM airbnb_reviews ar,NY_Listings NY WHERE
ar.Listing_ID=NY.Listing_ID ORDER BY Review_Date;
```

- **We are creating a table to store the dates for listings and its latitude, longitude and borough.**

```
CREATE TABLE NY_Bookings ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/NY_Bookings' AS SELECT nyl.Listing_ID,nyl.Latitude,nyl.Longitude,
nyl.City,nyl.Neighbourhood,nyd.Review_Date FROM NY_Listings nyl,NY_Dates nyd WHERE
nyl.Listing_ID=nyd.Listing_ID ORDER BY City;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the tables using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Bookings LIMIT 5;
+-------------------------+-----------------------+------------------------+--------------------+-----------------------------+---------------------------+--+
| ny_bookings.listing_id  | ny_bookings.latitude  | ny_bookings.longitude  | ny_bookings.city   | ny_bookings.neighbourhood   | ny_bookings.review_date   |
+-------------------------+-----------------------+------------------------+--------------------+-----------------------------+---------------------------+--+
| 30340171                | 40.87684              | -73.86389              | Bronx              | Williamsbridge              | 2020-01-01                |
| 18590                   | 40.84212398           | -73.78519698           | Bronx              | City Island                 | 2019-06-08                |
| 116551                  | 40.85926899           | -73.90142454           | Bronx              | Fordham                     | 2020-02-24                |
| 89621                   | 40.90389506           | -73.84151296           | Bronx              | Wakefield                   | 2019-11-02                |
| 17747                   | 40.85197955           | -73.78930423           | Bronx              | City Island                 | 2019-11-02                |
+-------------------------+-----------------------+------------------------+--------------------+-----------------------------+---------------------------+--+
5 rows selected (0.077 seconds)
```

- **After creating the table ,we need to download the file into our local system to visualize it in Tableau,**

Note: The steps to download the file to your local system is the same as the steps mentioned in Analysis 1 (**Steps (a),(b), (c) on pages 17,18 and 19**). These three steps must be followed in all the further analysis.

Do not forget to change the name of the directory to the current one '**NY_Bookings**' and download the file with the name **'ny_bookings.csv' into your local system.**

- **Once the file is downloaded into our local system, we need to visualize the data in Tableau,**

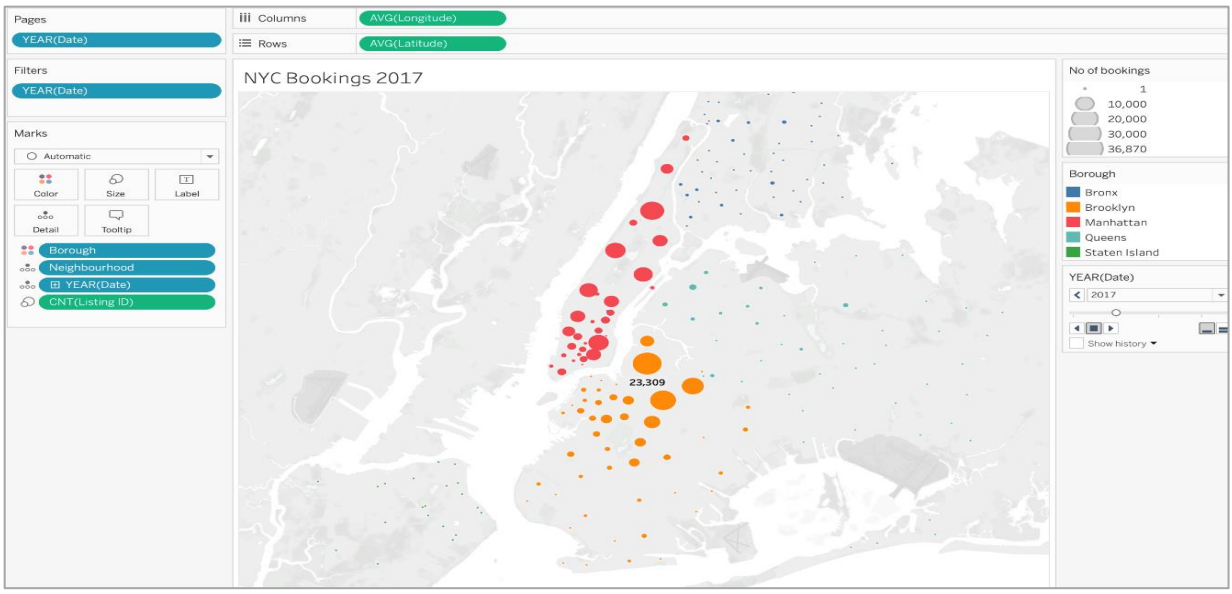
1) Open Tableau Desktop application.
2) Choose File-> Open-> ny_bookings.csv
3) We need to rename the fields.
4) Go to Sheet 1.

- **Visualizing data in Tableau for 'ny_bookings.csv' file,**

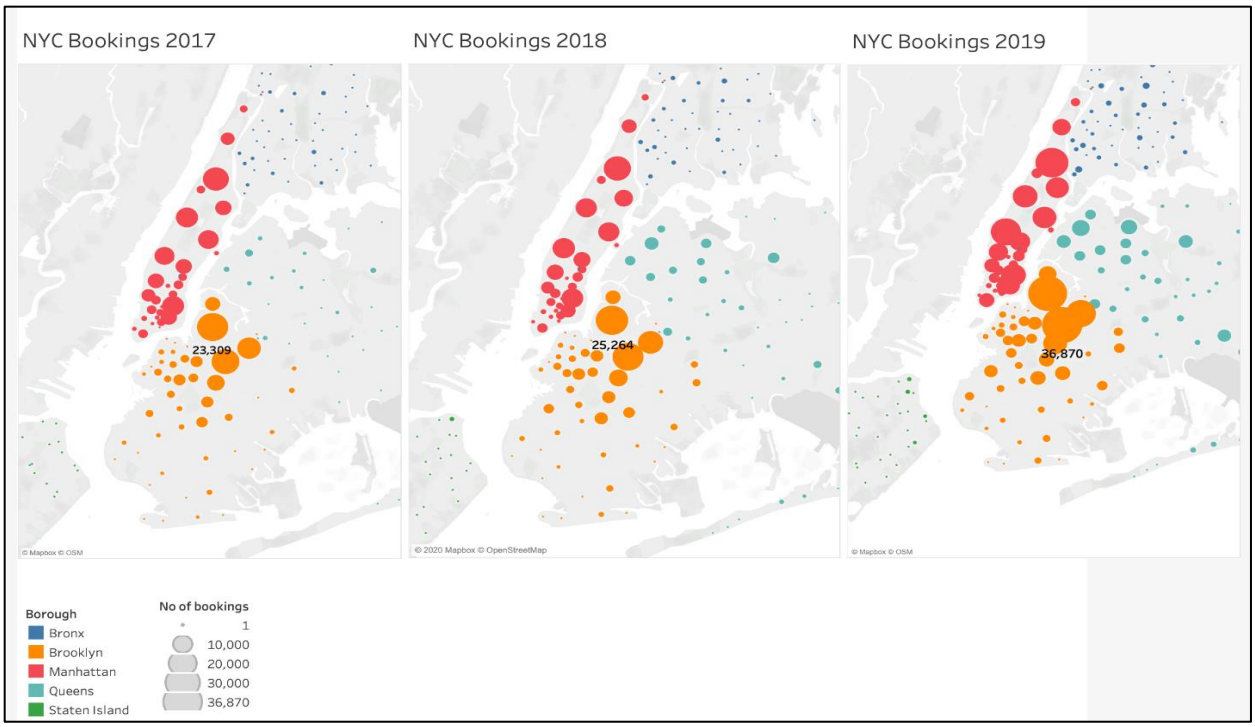
a. Right click on Latitude in Measures and change the Geographic role to Latitude. Do the same for Longitude.

a. Drag the Latitude to Rows and Longitude to Columns.
b. Symbol maps is generated.
c. Drag Borough, Neighbourhood, Date and Listing ID to **Marks** field.
d. In the drop down of the Listing ID field in Marks, choose COUNT.
e. In the left side(…) of the Borough field in Marks, choose color.In this way we can differentiate each borough by color.
f. In the left side(…) of the CNT(Listing ID) field in Marks, choose size, so that the size of the circles represent the number of bookings.
g. Filter the Date to choose YEAR and select the years you need to view the data for. (I have chosen 2017,2018,2019).
h. Drag the YEAR field to Pages, so that you can navigate between different years in the same sheet.

The sample worksheet for the year 2017 with the filters is shown below,



I have combined all three years into a single dashboard as shown below,

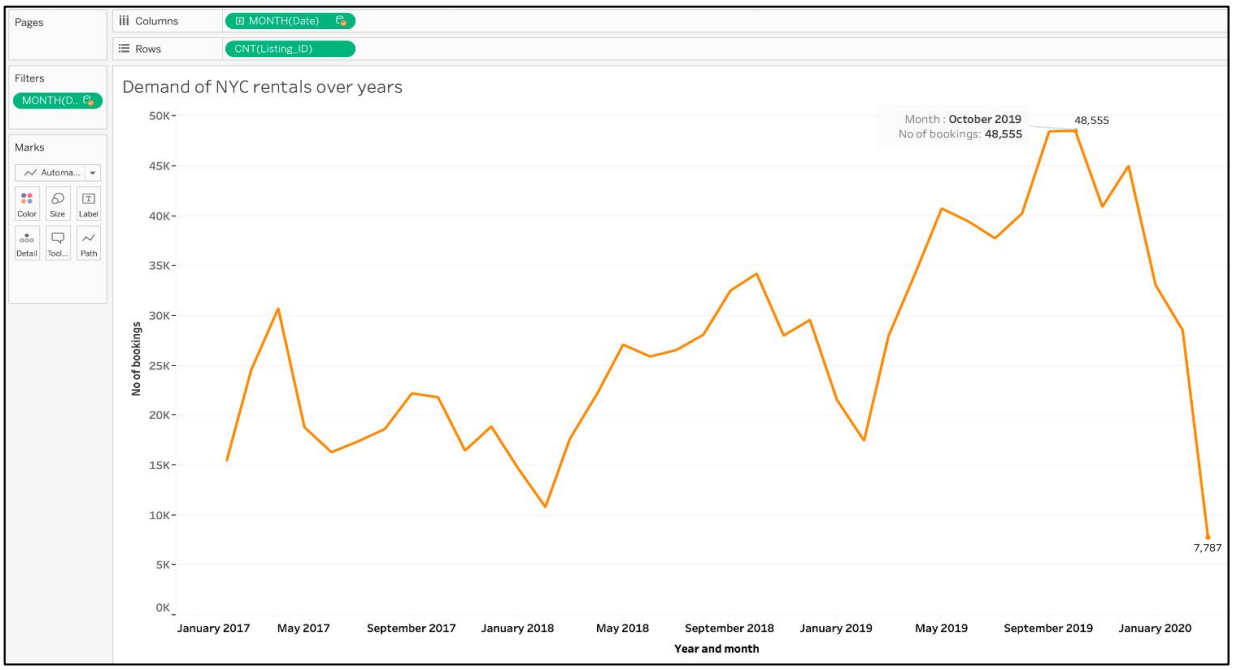


→ **We see that there is slight increase in the bookings from 2017 to 2018 and the number of bookings has greatly increased in 2019 for NYC.**

- **We can see the variation in the number of bookings over years and the total number of bookings using a lines(continuous) graph,**

a. Drag the Listing ID to Rows and Date to Columns.
b. Choose Month from Date and filter the years as needed.

We can see the below graph,



→ **We see that the number of bookings has gradually increased from 2017 and the maximum number of bookings have taken place in the year 2019 in the months of September and October.**

Los Angeles County:

- **We are creating a table 'LA_Dates' to store the Listing_ID and its corresponding review dates for Los Angeles County using the CTAS query below,**

```
CREATE TABLE LA_Dates ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/LA_Dates' AS
SELECT DISTINCT LA.Listing_ID,LR.Review_Date FROM LA_Listings LA,LA_Review_Dates LR WHERE
LA.Listing_ID=LR.Listing_ID
UNION DISTINCT
SELECT DISTINCT LA.Listing_ID,ar.Date_Of_Review FROM airbnb_reviews ar,LA_Listings LA WHERE
ar.Listing_ID=LA.Listing_ID ORDER BY Review_Date;
```

- **We are creating a table 'LA_Bookings' to store the dates for listings and its latitude, longitude and neighborhood.**

```
CREATE TABLE LA_Bookings ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/LA_Bookings' AS SELECT
lal.Listing_ID,lal.Latitude,lal.Longitude,lal.Neighbourhood,ld.Review_Date FROM LA_Listings
lal,LA_Dates ld WHERE lal.Listing_ID=ld.Listing_ID ORDER BY Neighbourhood;
```

- **We can check if the tables are created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Bookings LIMIT 5;
+-------------------------+-----------------------+------------------------+-----------------------------+---------------------------+--+
| la_bookings.listing_id  | la_bookings.latitude  | la_bookings.longitude  | la_bookings.neighbourhood   | la_bookings.review_date   |
+-------------------------+-----------------------+------------------------+-----------------------------+---------------------------+--+
| 14078522                | 34.51012966           | -118.2124069           | Acton                       | 2019-06-15                |
| 14078522                | 34.51012966           | -118.2124069           | Acton                       | 2019-07-28                |
| 14078522                | 34.51012966           | -118.2124069           | Acton                       | 2019-05-11                |
| 29471370                | 34.45475              | -118.17875             | Acton                       | 2019-12-07                |
| 14078522                | 34.51012966           | -118.2124069           | Acton                       | 2018-07-05                |
+-------------------------+-----------------------+------------------------+-----------------------------+---------------------------+--+
5 rows selected (0.216 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Steps (a),(b), (c) on pages 17,18 and 19.**Do not forget to change the name of the directory to the current one '**LA_Bookings**' and download the file with the name **'la_bookings.csv'.**

- **Uploading the file to Tableau,**

Note: The steps to upload the csv file to Tableau is the same for all analysis. We need to follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC). Don't forget to choose the current file 'la_bookings.csv'.

- **Visualizing data in Tableau for 'la_bookings.csv' file,**

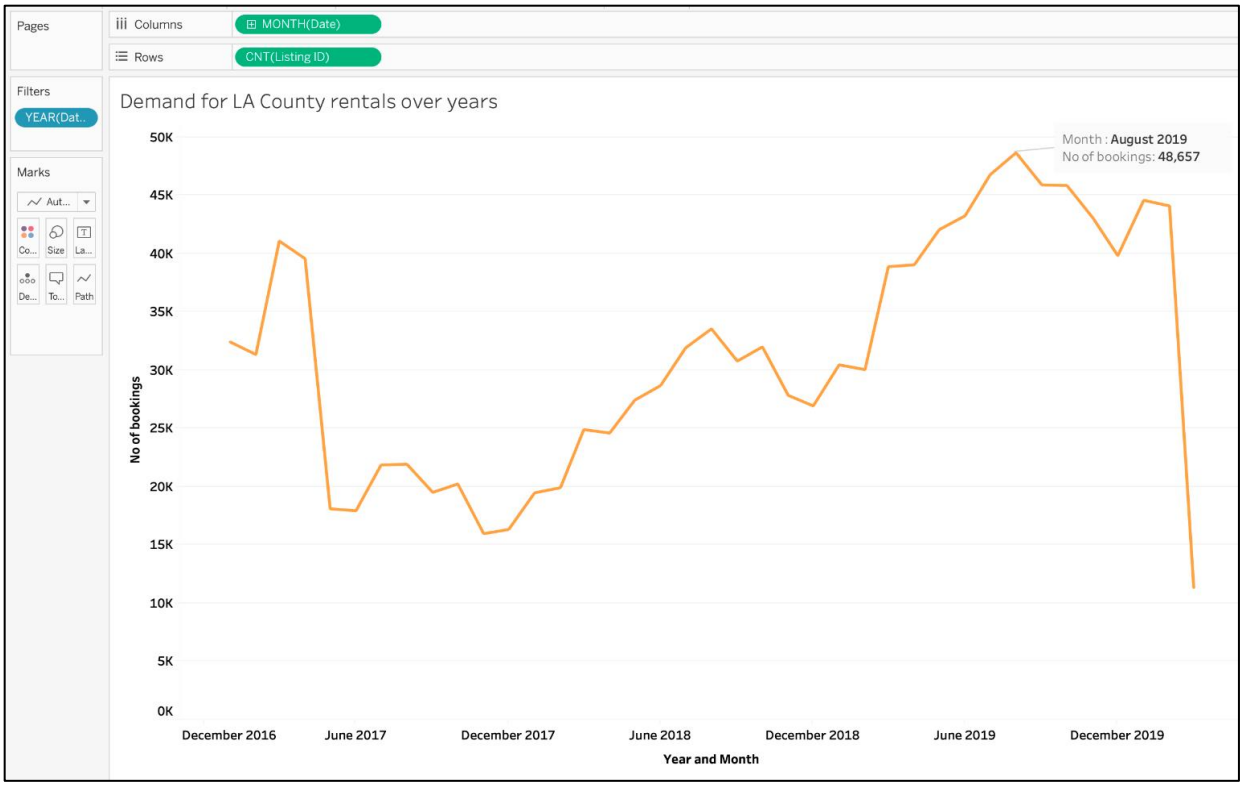
Repeat the same steps as we did for 'ny_bookings.csv' in the previous section.

Dashboard containing all the three years' bookings,

- **We can see the variation in the number of bookings over years and the total number of bookings using a lines(continuous) graph.**

Do the same steps as we did in the previous section (For NYC)

You can see the below chart,



→ **We see that there are a greater number of bookings in LA County in the years 2017 and 2019.**

Analysis 3:

In this part we are analyzing the Demand for various property types across years in NYC and LA County.

New York City:

We are choosing Apartment type for analysis as there are many listings and bookings for Apartment types in NYC.

- **We are creating a table 'NY_Apt' to store the details of the listings that are of Apartment type and its corresponding booking dates.**

```
CREATE TABLE NY_Apt ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/NY_Apt' AS SELECT nyl.Listing_ID,nyl.Latitude,nyl.Longitude,
nyl.City,nyl.Neighbourhood,nyl.Property_Type,nyd.Review_Date FROM NY_Listings nyl,NY_Dates
nyd WHERE nyl.Listing_ID=nyd.Listing_ID AND nyl.Property_Type='Apartment' ORDER BY City;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM Ny_Apt LIMIT 5;
+--------------------+-------------------+---------------------+---------------+-------------------------+-------------------------+---------------------+--+
| ny_apt.listing_id  | ny_apt.latitude   | ny_apt.longitude    | ny_apt.city   | ny_apt.neighbourhood    | ny_apt.property_type    | ny_apt.review_date  |
+--------------------+-------------------+---------------------+---------------+-------------------------+-------------------------+---------------------+--+
| 182649             | 40.83030074       | -73.92825959        | Bronx         | Highbridge              | Apartment               | 2019-12-14          |
| 115678             | 40.85959069       | -73.8700668         | Bronx         | Allerton                | Apartment               | 2020-02-18          |
| 33807187           | 40.82758          | -73.88741           | Bronx         | Longwood                | Apartment               | 2019-04-21          |
| 135393             | 40.81634093       | -73.92651884        | Bronx         | Mott Haven              | Apartment               | 2018-08-11          |
| 141154             | 40.82416563       | -73.90156246        | Bronx         | Longwood                | Apartment               | 2019-09-30          |
+--------------------+-------------------+---------------------+---------------+-------------------------+-------------------------+---------------------+--+
5 rows selected (0.295 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one 'NY_Apt' and download the file with the name **'ny_apt.csv'.**

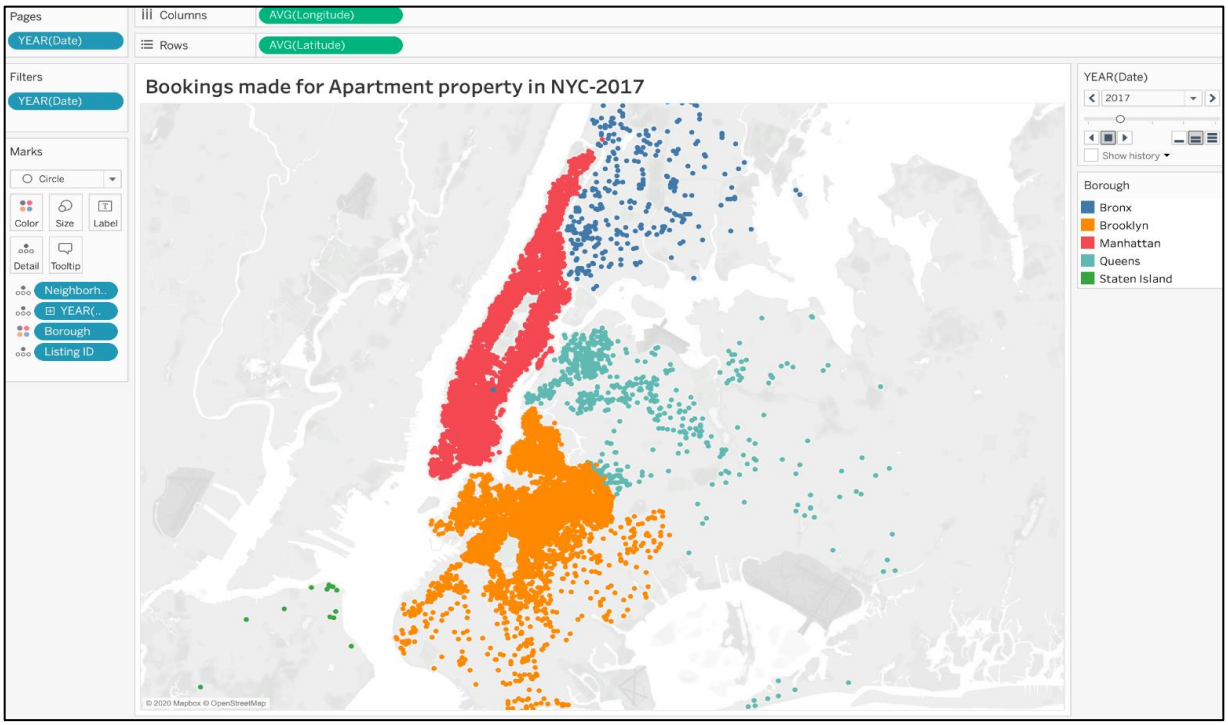
- **Uploading the file to Tableau,**

Note: **Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'ny_apt.csv'.**

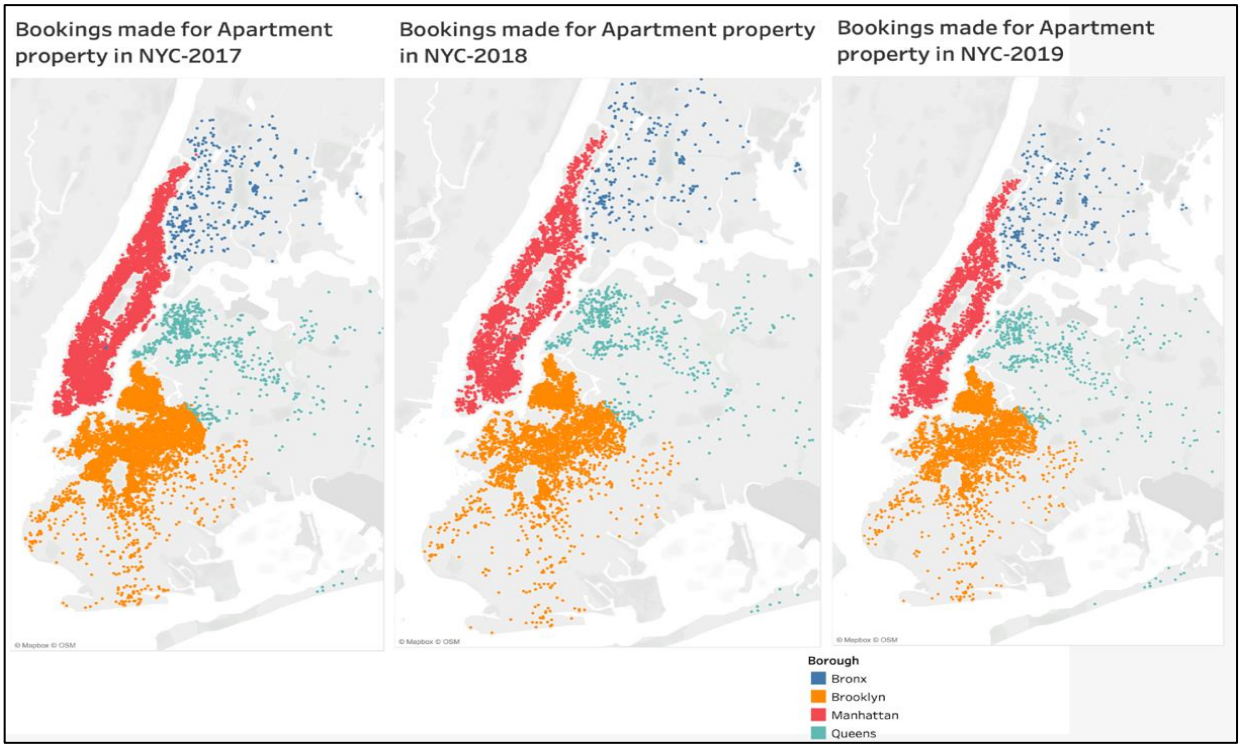
- **Visualizing data in Tableau for ny_apt.csv file,**

a. Right click on Latitude in Measures and change the Geographic role to Latitude.Do the same for Longitude.
b. Drag the Latitude to rows and Longitude to Columns.
c. Symbol maps is generated.
d. Drag Borough, Neighborhood, Date and Listing ID to **Marks** field.
e. In the drop down of the Listing ID field in Marks,change it to Dimension.
f. In the left side(…) of the Borough field in Marks, choose color.In this way we can differentiate each borough by color.
g. Filter the Date to choose YEAR and choose the years you need to view data for. (I have chosen 2017,2018,2019).
h. Drag the YEAR field to Pages, so that you can navigate between different years in the same sheet.

The sample worksheet for the year 2017 with the filters is shown below,



Dashboard containing all the three years' bookings,



→ **We see that Apartment bookings in NYC are high in 2017 and they have gradually reduced from 2017.**

In this part we are determining the exact number of bookings for all types of properties in NYC over years.

- **We are creating a table 'NY_Properties' to store the Booking dates of Listings of all the property types in NYC.**

```
CREATE TABLE NY_Properties ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/NY_Properties' AS SELECT COUNT(Listing_ID) AS No_Of_Listings,Property_Type,YEAR(review_date) AS year,Date_Format(review_date,'MMM') AS Month FROM (SELECT nyl.Listing_ID,nyl.Property_Type,nyd.review_date FROM NY_Listings nyl,NY_Dates nyd where nyl.Listing_ID=nyd.Listing_ID ORDER BY Listing_ID) a group by YEAR(review_date),Date_Format(review_date,'MMM'),Property_Type ORDER BY Property_Type;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Properties LIMIT 5;
+-------------------------------+-------------------------------+---------------------+-----------------------+--+
| ny_properties.no_of_listings  | ny_properties.property_type   | ny_properties.year  | ny_properties.month   |
+-------------------------------+-------------------------------+---------------------+-----------------------+--+
| 8712                          | Apartment                     | 2015                | Jun                   |
| 698                           | Apartment                     | 2012                | Jul                   |
| 2072                          | Apartment                     | 2013                | May                   |
| 2047                          | Apartment                     | 2014                | Mar                   |
| 8564                          | Apartment                     | 2015                | Jul                   |
+-------------------------------+-------------------------------+---------------------+-----------------------+--+
5 rows selected (0.214 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one 'NY_Properties' and download the file with the name **'ny_properties.csv'.**

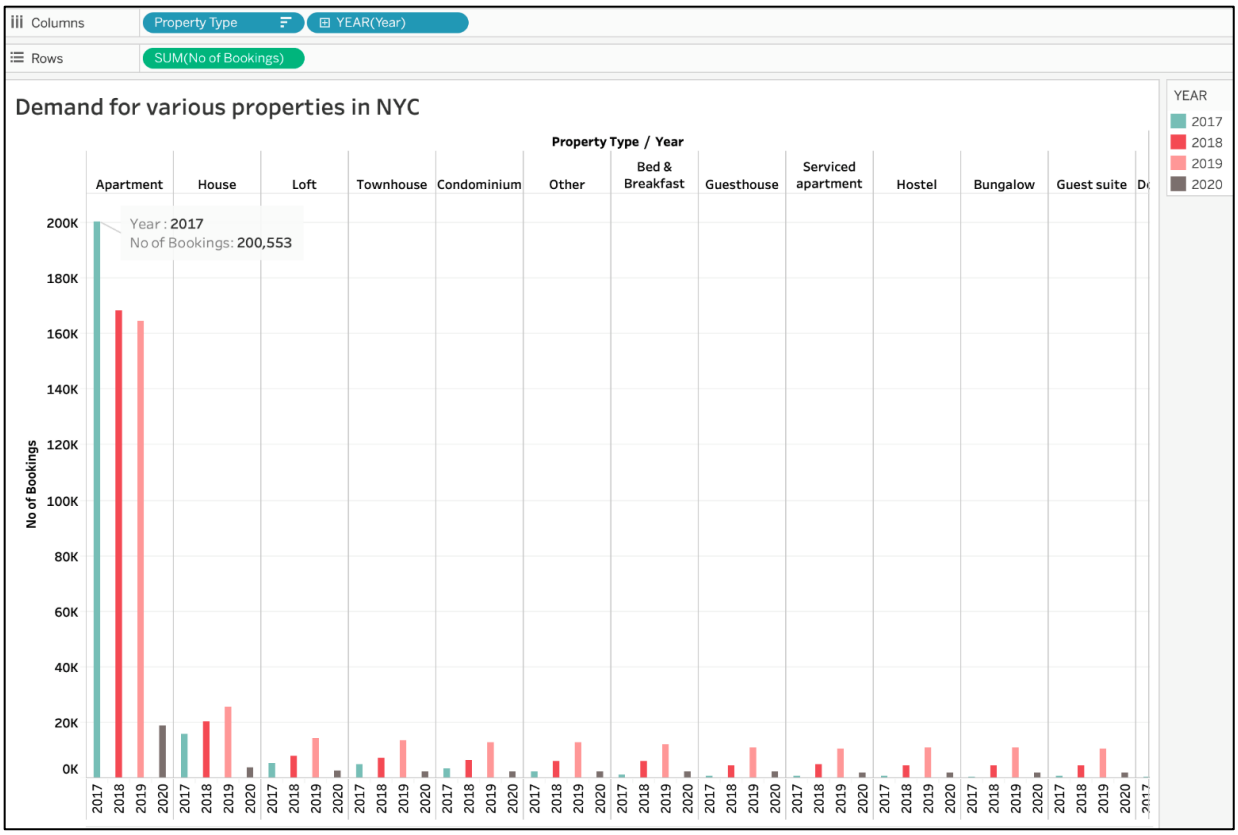
- **Uploading the file to Tableau,**

Note: **Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'ny_properties.csv'.**

- **Visualizing data in Tableau for 'ny_properties.csv' file,**

a. Drag the Property Type and Year Dimensions to Columns.
b. Change the Year to Date data type from the drop down.
c. Drag the No of Booking measure to Rows.
d. Choose Side-by-side bars chart.

We can see the below chart,



→ **We see that the bookings for Apartment are maximum followed by House.**

→ **The number of bookings for Apartment have decreased from 2017 to 2019 while the bookings for House have gradually increased from 2017.**

Los Angeles County:

- **We are creating a table 'LA_House' to store the details of the listings that are oh House type and its corresponding booking dates.**

```
CREATE TABLE LA_House ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/LA_House' AS SELECT la.Listing_ID,la.Latitude,la.Longitude,la.Neighbourhood,lad.Review_Date FROM LA_Listings la,LA_Dates lad WHERE la.Listing_ID=lad.Listing_ID AND la.Property_Type='House' ORDER BY Neighbourhood;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_House LIMIT 5;
+----------------------+---------------------+---------------------+--------------------------+----------------------+--+
| la_house.listing_id  | la_house.latitude   | la_house.longitude  | la_house.neighbourhood   | la_house.review_date |
+----------------------+---------------------+---------------------+--------------------------+----------------------+--+
| 14078522             | 34.51012966         | -118.2124069        | Acton                    | 2018-05-27           |
| 14078522             | 34.51012966         | -118.2124069        | Acton                    | 2020-01-11           |
| 14078522             | 34.51012966         | -118.2124069        | Acton                    | 2019-12-30           |
| 14078522             | 34.51012966         | -118.2124069        | Acton                    | 2020-02-19           |
| 14078522             | 34.51012966         | -118.2124069        | Acton                    | 2018-04-09           |
+----------------------+---------------------+---------------------+--------------------------+----------------------+--+
5 rows selected (0.214 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.**Do not forget to change the name of the directory to the current one 'LA_House' and download the file with the name **'la_house.csv'.**

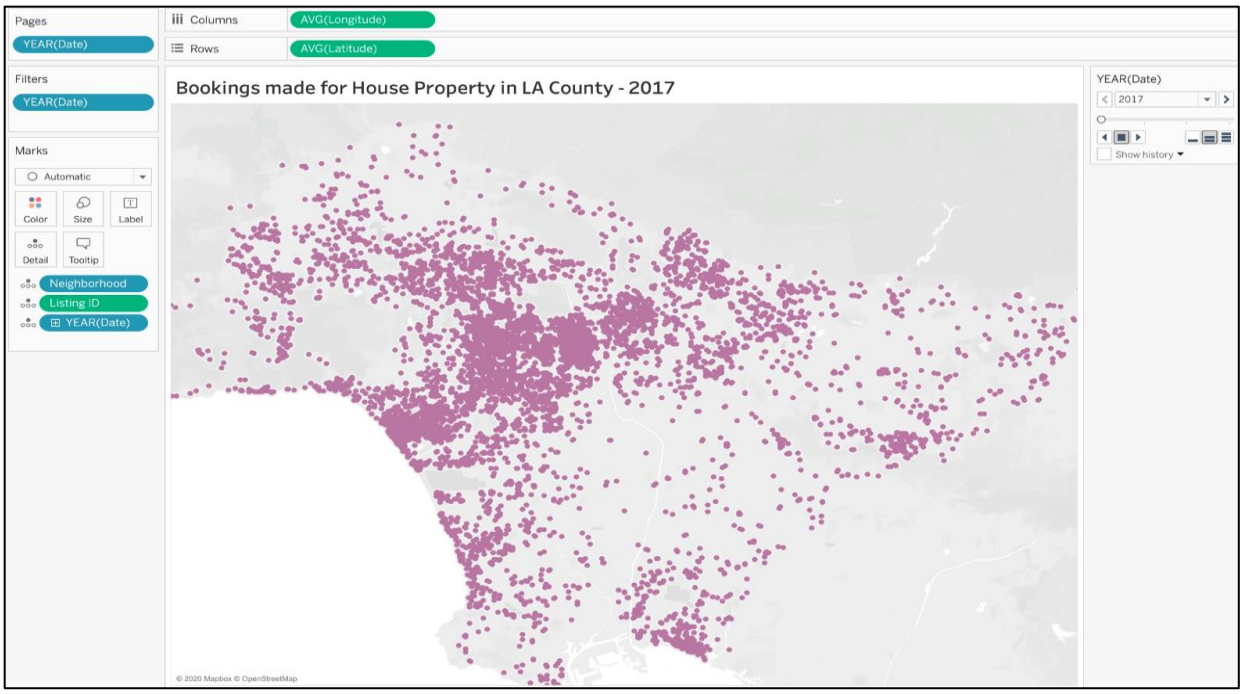
- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC). Don't forget to choose the current file 'la_house.csv'.

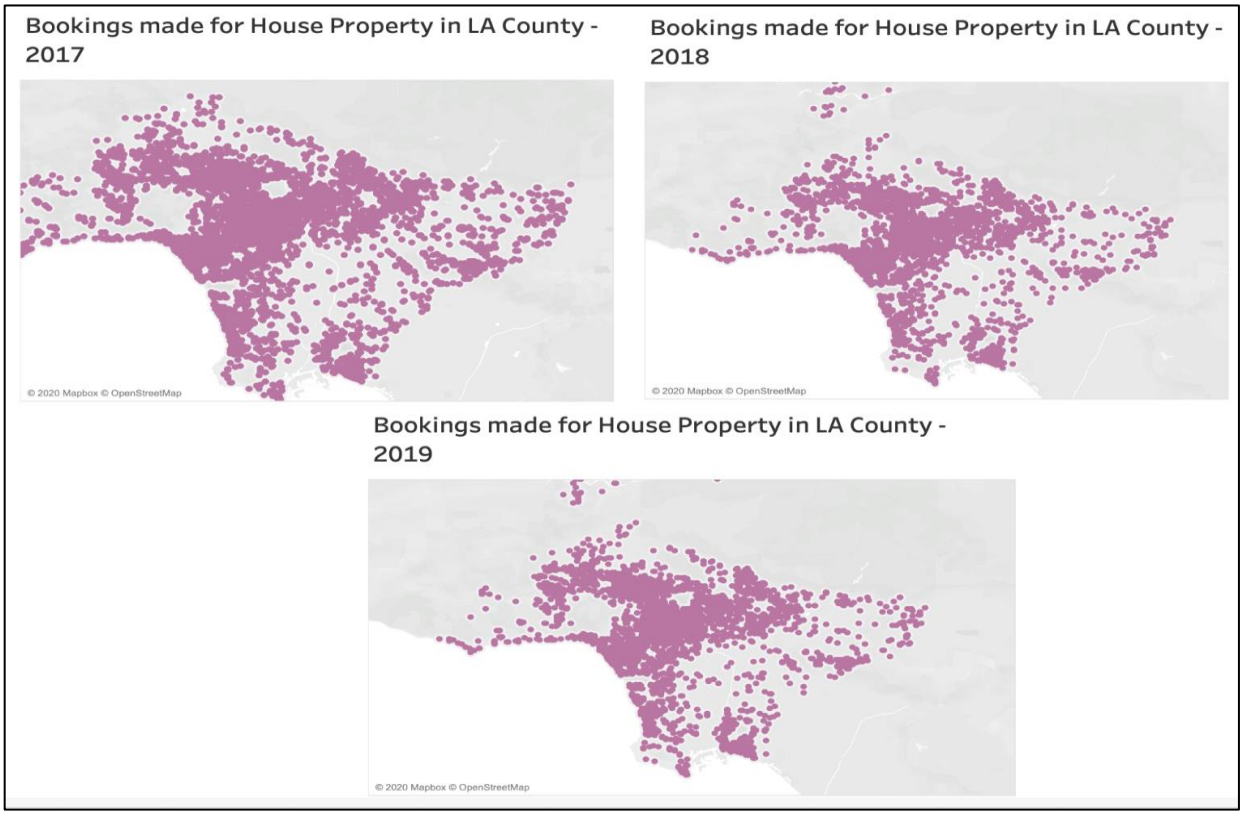
- **Visualizing data in Tableau for la_house.csv file,**

a. Right click on Latitude in Measures and change the Geographic role to Latitude.Do the same for Longitude.
b. Drag the Latitude to rows and Longitude to Columns.
c. Symbol maps is generated.
d. Drag Neighborhood,Date and Listing ID to **Marks** field.
e. In the drop down of the Listing ID field in Marks, change it to Dimension.
f. In the left side(…) of the Borough field in Marks, choose color.In this way we can differentiate each borough by color.
g. Filter the Date to choose YEAR and choose the years you need to view data for. (I have chosen 2017,2018,2019).
h. Drag the YEAR field to Pages, so that you can navigate between different years in the same sheet.

The sample worksheet for the year 2017 with the filters is shown below,



Dashboard containing all the three years' bookings,



→ **We see that the bookings are high in 2017 and they have largely reduced in 2018 and then slightly increased in 2019.**

In this part we are determining the exact number of bookings for all types of properties over years in LA County.

- **We are creating a table 'LA_Properties' to store the Booking dates of all property types in LA.**

```
CREATE TABLE LA_Properties ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/LA_Properties' AS
SELECT COUNT(Listing_ID) As No_Of_Listings,Property_Type,Review_Date FROM
(SELECT ll.Listing_ID,ll.Property_Type,ld.review_date FROM LA_Listings ll,LA_Dates ld where
ll.Listing_ID=ld.Listing_ID ORDER BY Listing_ID) a group by Review_Date,Property_Type ORDER BY
Property_Type;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Properties LIMIT 5;
+-------------------------------+-------------------------------+-------------------------------+--+
| la_properties.no_of_listings  | la_properties.property_type   | la_properties.review_date     |
+-------------------------------+-------------------------------+-------------------------------+--+
| 185                           | Apartment                     | 2015-12-24                    |
| 235                           | Apartment                     | 2018-03-04                    |
| 138                           | Apartment                     | 2019-06-26                    |
| 117                           | Apartment                     | 2020-01-21                    |
| 4                             | Apartment                     | 2012-10-13                    |
+-------------------------------+-------------------------------+-------------------------------+--+
5 rows selected (0.075 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: Follow steps (a),(b), (c) on pages 17,18 and 19. Do not forget to change the name of the directory to the current one 'LA_Properties' and download the file with the name **'la_properties.csv'.**

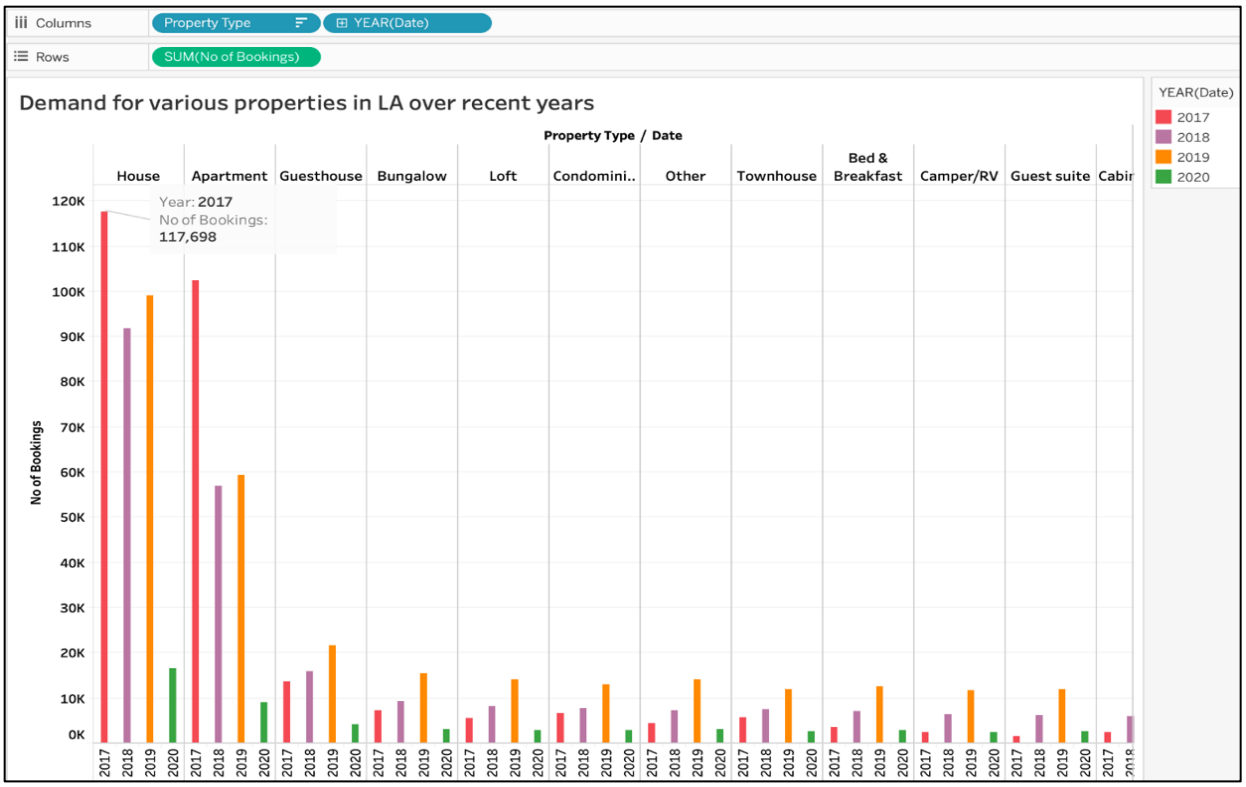
- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'la_properties.csv'.

- **Visualizing data in Tableau for 'la_properties.csv' file,**

a. Drag the Property Type and Date Dimensions to Column.
b. Drag the No of Bookings measure to Rows.
c. Filter the Year to 2017,2018,2019,2020.
d. Choose Side-by-side bars chart.

We can see the below chart,



→ **We see that the maximum number of bookings is for House followed by Apartments in 2017.The bookings for House have reduced in 2018 and 2019.**

Analysis 4:

We are analyzing the distribution of listings and the number of listings in various neighborhoods of NYC and LA.

New York City:

- **We are creating a table 'NY_Distribution' for the number of listings in each borough and neighborhood.**

```
CREATE TABLE NY_Distribution ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/NY_Distribution' AS SELECT Count(Listing_ID) AS Total_Listings,City,Neighbourhood FROM ny_listings GROUP BY City,Neighbourhood ORDER BY City;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Distribution LIMIT 5;
+-----------------------------------+-----------------------+-----------------------------------+--+
| ny_distribution.total_listings    | ny_distribution.city  | ny_distribution.neighbourhood     |
+-----------------------------------+-----------------------+-----------------------------------+--+
| 30                                | Bronx                 | Belmont                           |
| 29                                | Bronx                 | Bronxdale                         |
| 11                                | Bronx                 | Baychester                        |
| 5                                 | Bronx                 | Castle Hill                       |
| 23                                | Bronx                 | City Island                       |
+-----------------------------------+-----------------------+-----------------------------------+--+
5 rows selected (0.285 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**NY_Distribution**' and download the file with the name **'ny_distribution.csv'.**

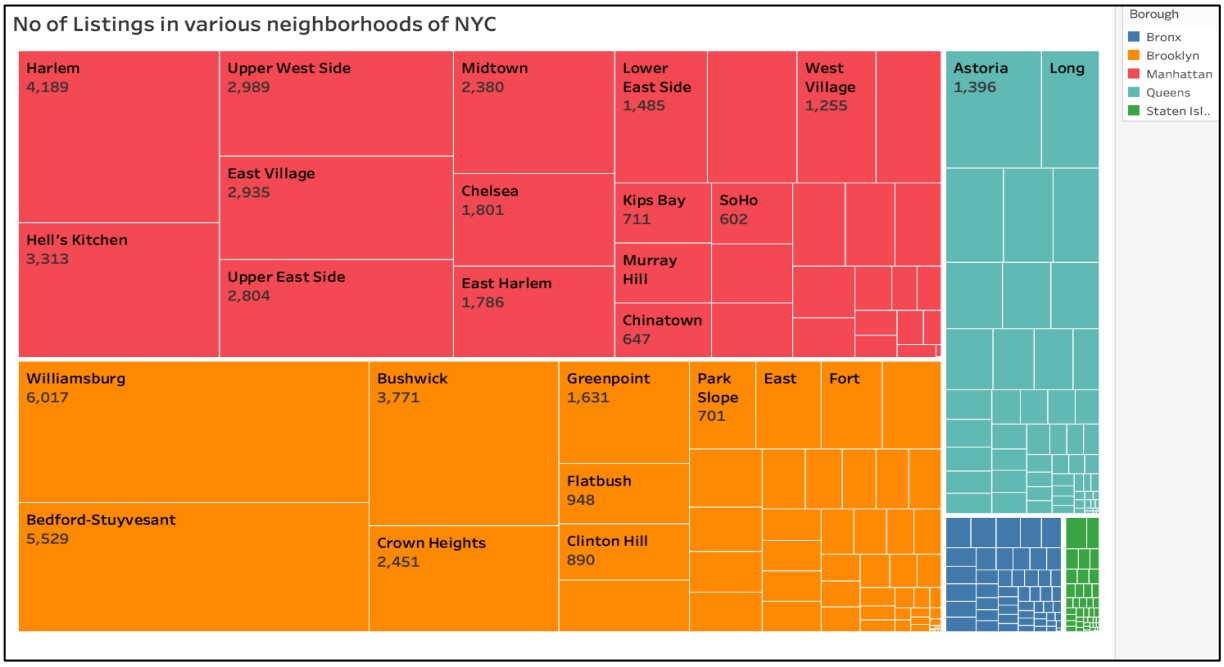
- **Uploading the file to Tableau,**

Note: **Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'ny_distribution.csv'.**

- **Visualizing data in Tableau for ny_distribution.csv file,**

a. Drag Borough and Neighborhood to Rows.
b. Drag No Of Listings to Columns.
c. Choose **Tree Maps** chart.

We can see the below chart,



→ **We see that Manhattan and Brooklyn boroughs have the highest number of listings. Harlem, Bedford-Stuyvesant and Williamsburg are the neighborhoods with more listings.**

Los Angeles:

- **We are creating a table 'LA_Distribution' for the number of listings in each neighborhood in LA County.**

```
CREATE TABLE LA_Distribution ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/LA_Distribution' AS SELECT Count(Listing_ID) AS Total_Listings,Neighbourhood FROM LA_listings GROUP BY Neighbourhood;
```

- **We can check if the table is created using SHOW TABLES command and check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Distribution LIMIT 5;
+--------------------------------+--------------------------------+--+
| la_distribution.total_listings | la_distribution.neighbourhood  |
+--------------------------------+--------------------------------+--+
| 1                              |                                |
| 7                              | Acton                          |
| 80                             | Adams-Normandie                |
| 73                             | Agoura Hills                   |
| 20                             | Agua Dulce                     |
+--------------------------------+--------------------------------+--+
5 rows selected (0.072 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: Follow steps (a),(b), (c) on pages 17,18 and 19. Do not forget to change the name of the directory to the current one '**LA_Distribution**' and download the file with the name **'la_distribution.csv'.**

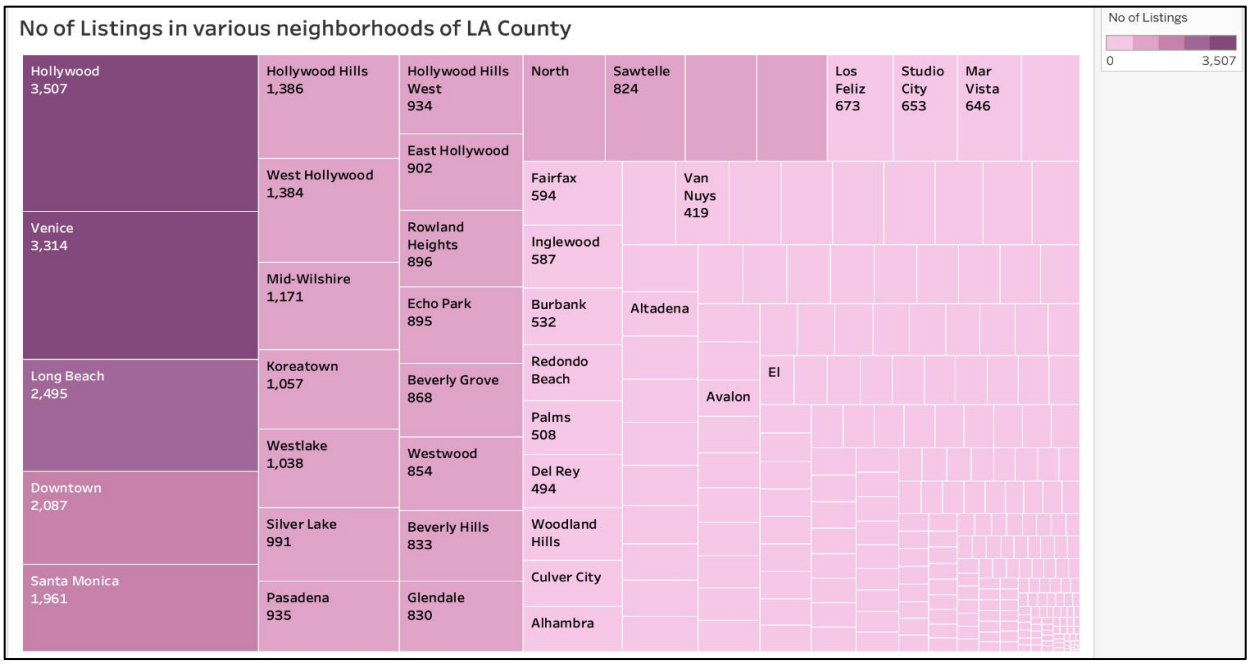
- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'la_distribution.csv'.

- **Visualizing data in Tableau for 'la_distribution.csv' file,**

a. Drag Neighborhood to Rows.
b. Drag No Of Listings to Columns.
c. Choose **Tree Maps** chart.

We can see the below chart,



→ **We see that Hollywood, Venice and Long beach are the neighborhoods with maximum number of listings in LA County.**

Analysis 5:

In this part we are determining the Revenue of various neighborhoods and which months fetch more revenue in New York City and Los Angeles County.

New York City:

- **We are creating a table 'NY_Revenue' to calculate and store the revenue of various neighborhoods in NYC.**

```
CREATE TABLE NY_Revenue ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/NY_Revenue' AS SELECT SUM(ny.min_nights * ny.Price * nyd.No_Of_Bookings) AS Revenue,nyd.Month,ny.City,ny.Neighbourhood FROM ny_listings ny,(SELECT Listing_ID,COUNT(*) AS No_Of_Bookings,date_format(review_date,'MMM') AS Month FROM ny_dates GROUP BY Listing_ID,date_format(review_date,'MMM')) nyd WHERE ny.Listing_ID=nyd.Listing_ID GROUP BY ny.City,ny.Neighbourhood,nyd.Month ORDER BY ny.City;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Revenue LIMIT 5;
+----------------------+---------------------+--------------------+-----------------------------+--+
| ny_revenue.revenue   | ny_revenue.month    | ny_revenue.city    | ny_revenue.neighbourhood    |
+----------------------+---------------------+--------------------+-----------------------------+--+
| 49833                | Aug                 | Bronx              | Allerton                    |
| 45610                | Dec                 | Bronx              | Allerton                    |
| 31581                | Feb                 | Bronx              | Allerton                    |
| 42311                | Jan                 | Bronx              | Allerton                    |
| 44279                | Jul                 | Bronx              | Allerton                    |
+----------------------+---------------------+--------------------+-----------------------------+--+
5 rows selected (0.088 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**NY_Revenue**' and download the file with the name **'ny_revenue.csv'.**

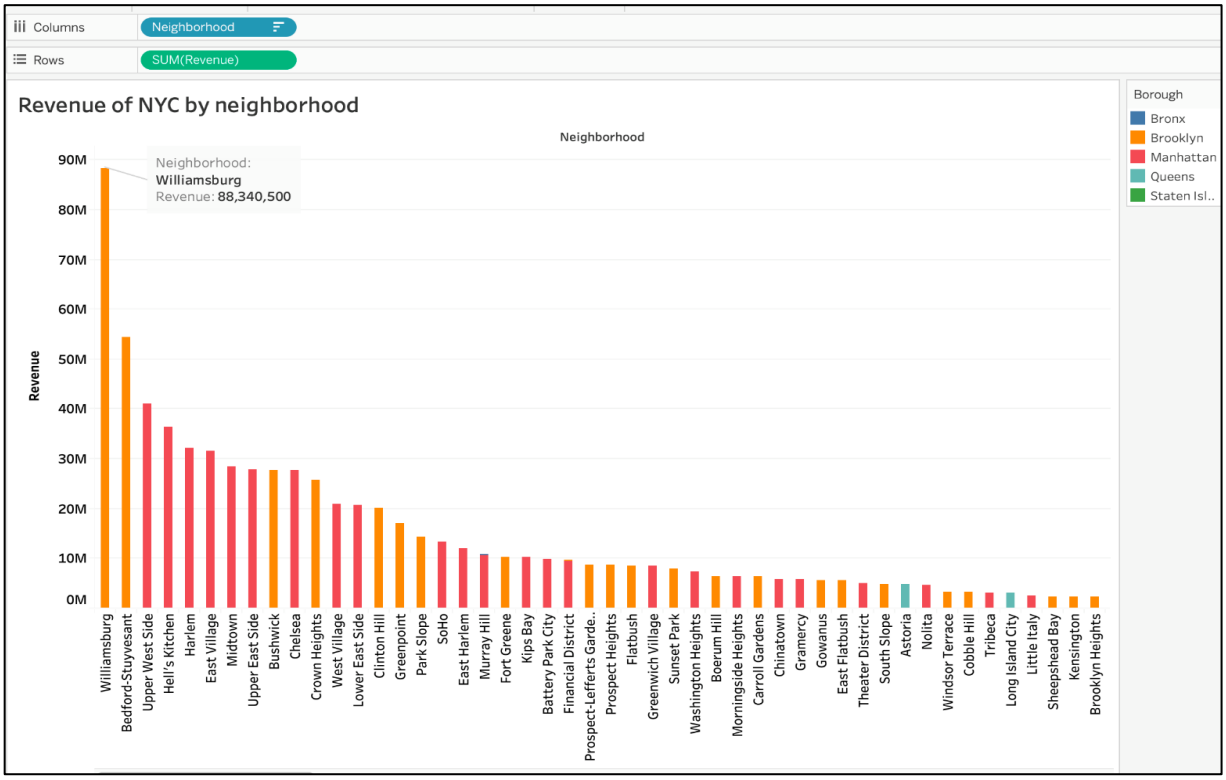
- **Uploading the file to Tableau,**

Note: **Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'ny_revenue.csv'.**

- **Visualizing data in Tableau for 'ny_revenue.csv' file,**

a. Drag Borough and Neighborhood to Columns
b. Drag Revenue to Rows.
c. Choose **Stacked Bars** chart.
d. Sort Neighborhood by Revenue field Descending.

You can see the following chart,



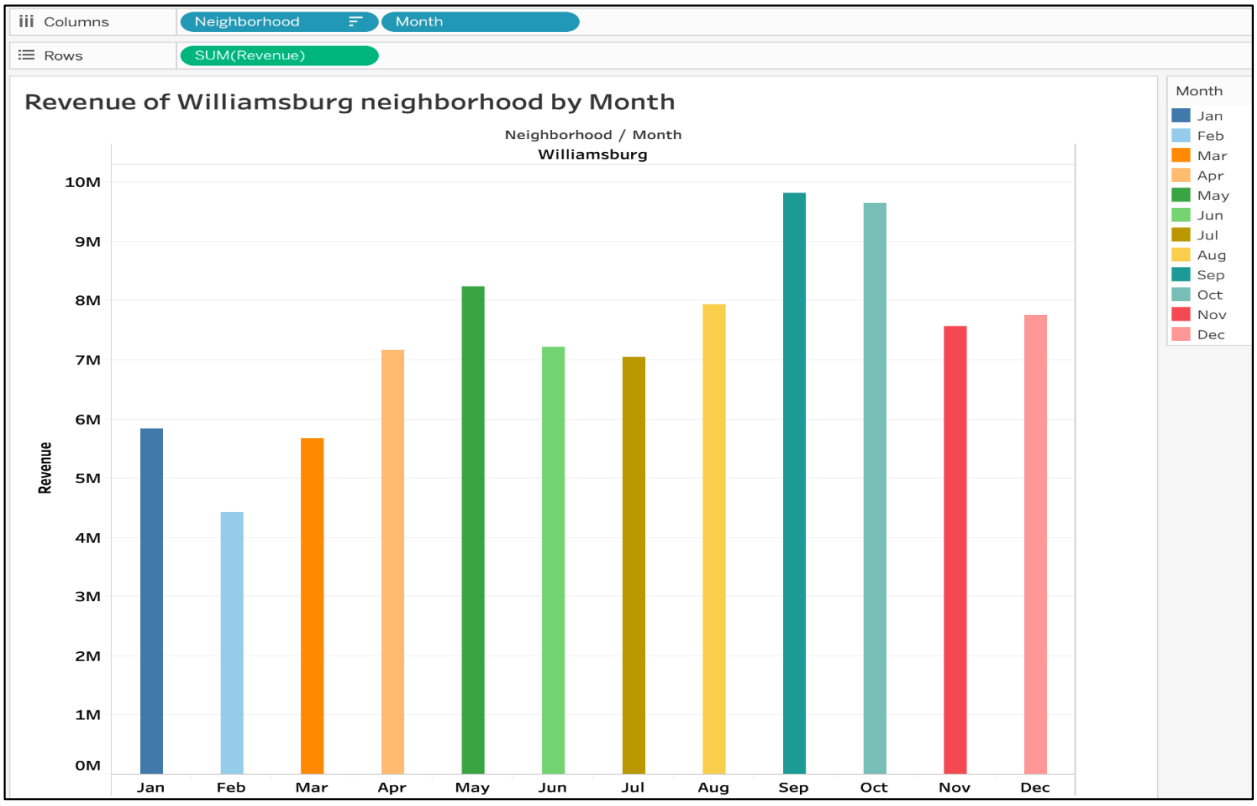
→ **We see that the revenue is maximum in Brooklyn and Manhattan boroughs. Furthermore, the neighborhoods Williamsburg and Bedford-Stuyvesant in Brooklyn and Upper-West-side in Manhattan provide the maximum revenue.**

- **We analyze the Revenue of all the months for Williamsburg neighborhood in Brooklyn.**

In Tableau,

a. Add Month dimension to Columns.

b. Filter the borough to choose Brooklyn and Neighborhood as Williamsburg. We can see the following chart,



→ **We see that the Revenue is maximum during the months of May, August, September and October. These are the best periods for the host to maximize revenue and the months before May are the best times for maintenance work.**

Los Angeles:

- **Create a table 'LA_Revenue' to calculate and store the revenue of various neighborhoods in LA.**

```
CREATE TABLE LA_Revenue ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/LA_Revenue' AS SELECT SUM(la.min_nights*la.Price*lad.No_Of_Bookings)
AS Revenue,lad.Month,la.Neighbourhood FROM la_listings la,(SELECT Listing_ID,COUNT(*) AS
No_Of_Bookings,date_format(review_date,'MMM') AS Month FROM LA_Dates GROUP BY
Listing_ID,date_format(review_date,'MMM')) lad WHERE la.Listing_ID=lad.Listing_ID GROUP BY
la.Neighbourhood,lad.Month ORDER BY la.Neighbourhood;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Revenue LIMIT 5;
+---------------------+--------------------+---------------------------+--+
| la_revenue.revenue  | la_revenue.month   | la_revenue.neighbourhood  |
+---------------------+--------------------+---------------------------+--+
| 603                 | Dec                | Acton                     |
| 916                 | Jun                | Acton                     |
| 913                 | Mar                | Acton                     |
| 918                 | Aug                | Acton                     |
| 470                 | Jan                | Acton                     |
+---------------------+--------------------+---------------------------+--+
5 rows selected (0.083 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**LA_Revenue**' and download the file with the name **'la_revenue.csv'.**

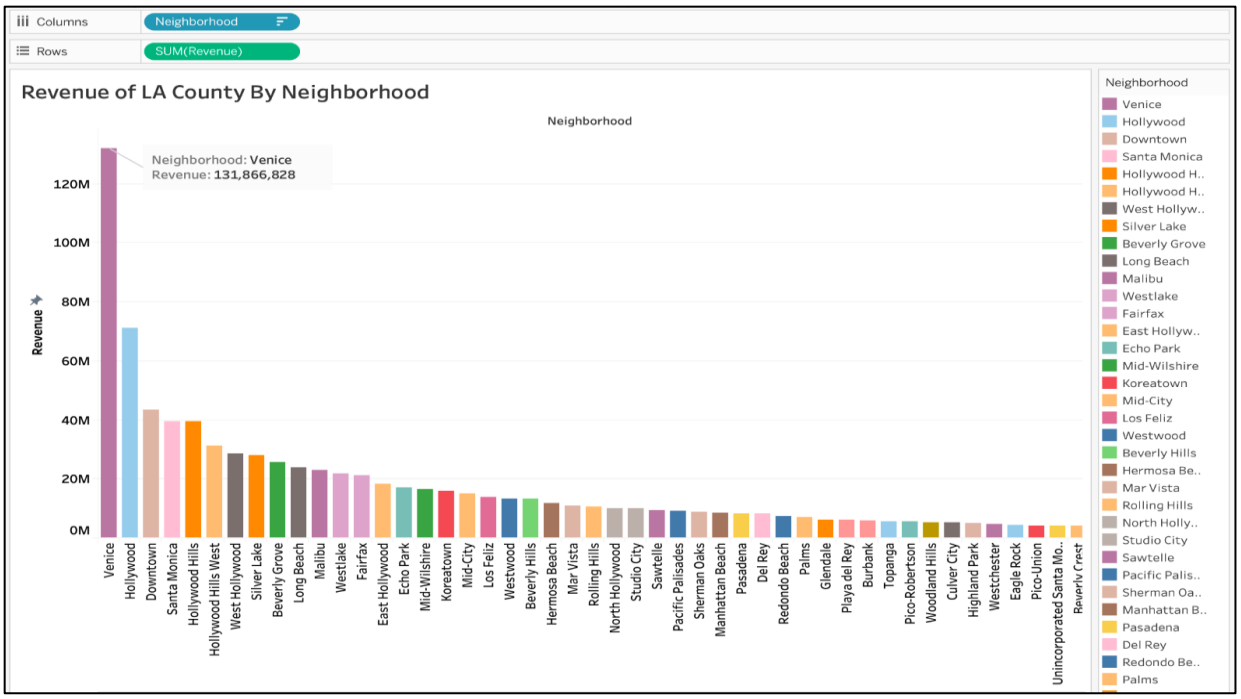
- **Uploading the file to Tableau,**

Note: **Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'la_revenue.csv'.**

- **Visualization of 'la_revenue.csv' in Tableau,**

a. Drag Neighborhood to Columns.
b. Drag Revenue to Rows.
c. Choose **Stacked Bars** chart.
d. Sort Neighborhood by Revenue field Descending.

You can see the below chart,



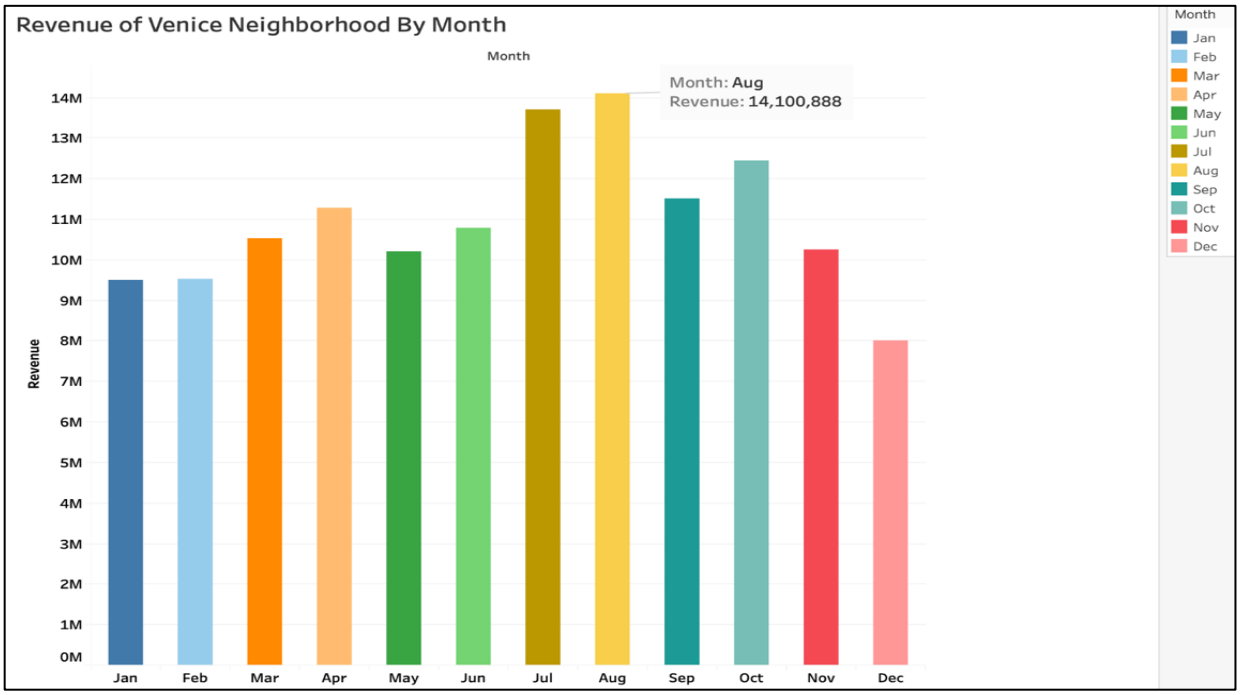
→ **We see that the neighborhood Venice has the highest revenue followed by Hollywood in LA.**

We analyze the Revenue of all months for Venice neighborhood.

- **In Tableau,**

a. Add Month dimension to Columns.

b. Filter the Neighborhood as Venice. We can see the following chart,



→ **We can see that the Revenue is maximum during the months of July, August and October. These are the best periods for the host to maximize revenue and the months before May are the best times for maintenance work.**

Analysis 6:

We are determining the average price of each neighborhood and the most expensive neighborhoods in NYC.

- **We are creating a table 'NY_Price' to store the price of the listings and its location.**

```
CREATE TABLE NY_Price ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/NY_Price' AS Select Price,Review_Scores_location AS
Location_Score,City,Neighbourhood,Latitude,Longitude FROM NY_Listings ORDER BY Neighbourhood;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Price LIMIT 5;
+-----------------+--------------------------+----------------+--------------------------+---------------------+----------------------+--+
| ny_price.price  | ny_price.location_score  | ny_price.city  | ny_price.neighbourhood   | ny_price.latitude   | ny_price.longitude   |
+-----------------+--------------------------+----------------+--------------------------+---------------------+----------------------+--+
| 50              | 0                        | Bronx          | Allerton                 | 40.86285            | -73.86669            |
| 142             | 0                        | Bronx          | Allerton                 | 40.85978            | -73.8619             |
| 110             | 9                        | Bronx          | Allerton                 | 40.86868176         | -73.85482829         |
| 142             | 0                        | Bronx          | Allerton                 | 40.86013            | -73.86339            |
| 142             | 0                        | Bronx          | Allerton                 | 40.86014            | -73.86301            |
+-----------------+--------------------------+----------------+--------------------------+---------------------+----------------------+--+
5 rows selected (0.14 seconds)
```

- **After creating the table ,we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**NY_Price**' and download the file with the name **'ny_price.csv'.**

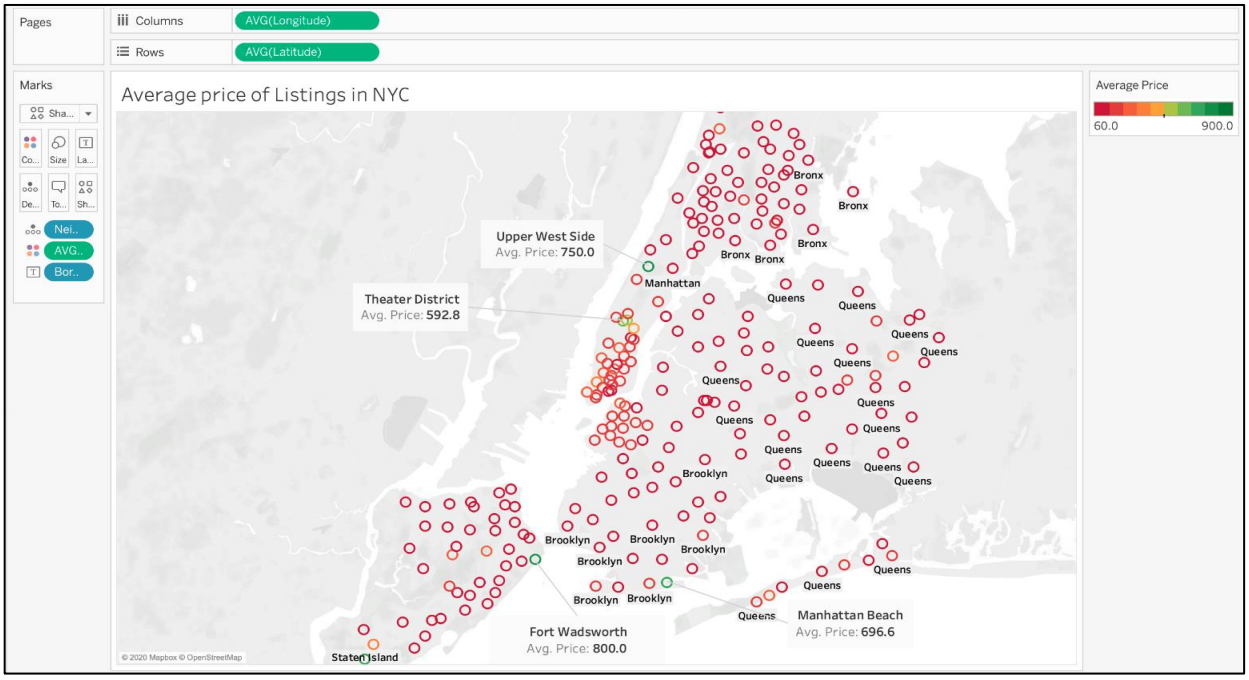
- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'ny_price.csv'.

- **Visualizing data in Tableau for 'ny_price.csv' file,**

a. Right click on Latitude in measures and change the Geographic role to Latitude. Do the same for Longitude.
b. Drag the Latitude to Rows and Longitude to Columns.
c. Symbol maps is generated.
d. Drag Borough, Neighborhood and Price to **Marks** field.
e. In the drop down of the Price field in Marks, choose Average.
f. In the left side(…) of the Borough field in Marks, choose Label. In this way we can differentiate each borough by its name.
g. In the left side(…) of the AVG(Price) field in Marks, choose color, so that each price range is of a different color.

We can see the below chart,



→ **We see that the Average price of Listings in neighborhoods of Manhattan and Brooklyn are higher.**

Analysis 7:

In this part we are determining the average price of each property type in various neighborhoods of NYC.

- **We are creating a table 'NY_Price_Type' to store the average price of each property type in various neighborhoods of NYC.**

```
CREATE TABLE NY_Price_Type ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/NY_Price_Type' AS SELECT COUNT(Listing_ID) AS
No_Of_Listings,City,Neighbourhood,Property_Type,AVG(Price) AS Average_Price FROM NY_Listings
GROUP BY City,Neighbourhood,Property_Type ORDER BY Property_Type;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_Price_Type LIMIT 5;
+-----------------------------+---------------------+-------------------------------+-------------------------------+-------------------------------+--+
| ny_price_type.no_of_listings | ny_price_type.city | ny_price_type.neighbourhood | ny_price_type.property_type | ny_price_type.average_price |
+-----------------------------+---------------------+-------------------------------+-------------------------------+-------------------------------+--+
| 2                           | Staten Island       | Concord                       | Apartment                     | 135.0                         |
| 6                           | Bronx               | Melrose                       | Apartment                     | 69.16666666666667             |
| 6                           | Bronx               | Hunts Point                   | Apartment                     | 64.16666666666667             |
| 7                           | Bronx               | Marble Hill                   | Apartment                     | 79.57142857142857             |
| 1                           | Staten Island       | Dongan Hills                  | Apartment                     | 120.0                         |
+-----------------------------+---------------------+-------------------------------+-------------------------------+-------------------------------+--+
5 rows selected (0.102 seconds)
```

- **After creating the table we need to download the file into our local system to visualize it in Tableau,**

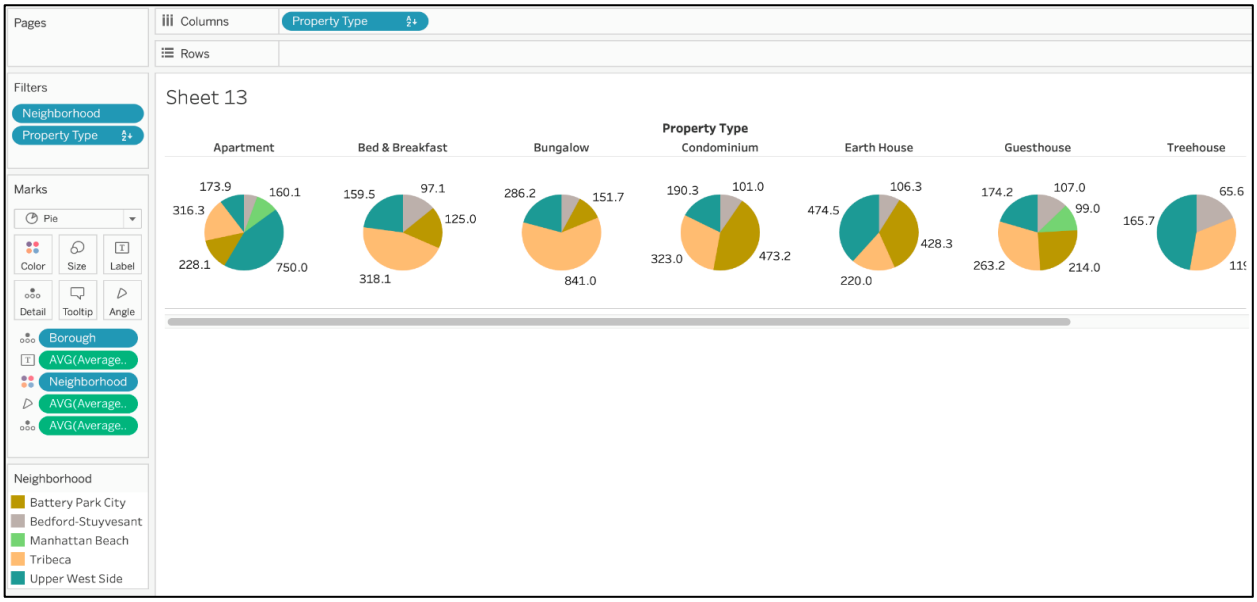
Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**NY_Price_Type**' and download the file with the name **'ny_price_type.csv'.**

- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC).Don't forget to choose the current file 'ny_price_type.csv'.

- **Visualizing the data in Tableau for 'ny_price_type.csv' file,**

a. Drag the Property Type to Columns.
b. Drag Borough, Neighbourhood,Average Price to Marks field
c. In the drop down of the Price field in Marks, choose Average.
d. Choose **Pie Charts.**
e. In the left side(…) of the Neighborhood field in Marks,choose Color.In this way we can differentiate the neighbourhoods by its color.
f. In the left side(…) of the AVG(Price) field in Marks,choose Angle,so that each price has a different angle in the pie chart
g. We are filtering 5 neighbourhoods (Seen in the chart).



→ **We see that Apartment and Treehouse have higher average price in Upper West Side neighborhood. Bungalow and Bed & Breakfast have higher average price in Tribeca.**

We are determining the number of listings for various prices in NYC.

- **We are creating a table 'NY_PriceListing' to store the No.Of.Listings for each price in NYC.**

```
CREATE TABLE NY_PriceListing ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/NY_PriceListing' AS SELECT COUNT(Listing_ID) AS No_Of_Listings,City,Price FROM NY_Listings GROUP BY City,Price ORDER BY Price;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM NY_PriceListing LIMIT 8;
+--------------------------------+-----------------------+-----------------------+--+
| ny_pricelisting.no_of_listings | ny_pricelisting.city  | ny_pricelisting.price |
+--------------------------------+-----------------------+-----------------------+--+
| 74                             | Brooklyn              | 0                     |
| 15                             | Queens                | 0                     |
| 6                              | Bronx                 | 0                     |
| 202                            | Manhattan             | 0                     |
| 3                              | Staten Island         | 0                     |
| 1                              | Manhattan             | 5                     |
| 13                             | Manhattan             | 10                    |
| 8                              | Queens                | 10                    |
+--------------------------------+-----------------------+-----------------------+--+
8 rows selected (0.074 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**NY_PriceListing**' and download the file with the name **'ny_pricelisting.csv'.**

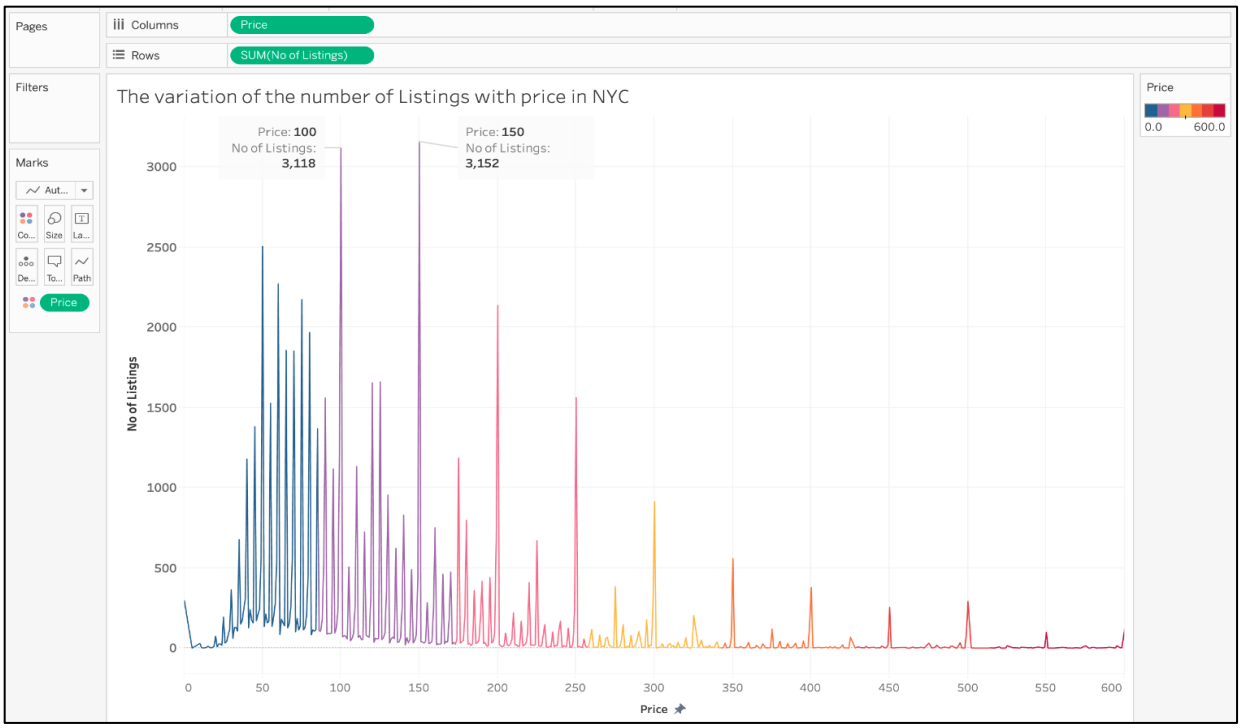
- **Uploading the file to Tableau,**

Note: **Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC). Don't forget to choose the current file 'ny_pricelisting.csv'.**

- **Visualizing the data in Tableau for 'ny_pricelisting.csv' file,**

a. Drag the Price to Columns and No of Listings to Rows.
b. Click the dropdown of Price and change it to Dimension.
c. Drag Price to Marks field and choose Color legend.

You can see the following chart,



→ **We see that there are a greater number of listings in the price range of ($50-$200).**

Analysis 8:

Los Angeles:

Supply and Demand – Bedroom configuration

Here we are determining the ratio of the number of bookings (demand) to the number of listings (supply) of different bedroom configurations.

- **We are creating a table 'LA_Bedrooms' to store the ratio of No.of.bookings to No.Of.Listings for each bedroom configuration in LA.**

```
CREATE TABLE LA_Bedrooms ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/LA_Bedrooms' AS SELECT Bedrooms,COUNT(distinct la.Listing_ID) AS No_Of_Listings,COUNT(Review_Date) AS NO_Of_Bookings,(COUNT(Review_Date)/COUNT(distinct la.Listing_ID)) AS Ratio FROM LA_Listings la,LA_Dates lad WHERE la.Listing_ID=lad.Listing_ID AND Bedrooms IS NOT NULL Group By Bedrooms;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Bedrooms;
+----------------------+----------------------------+-----------------------------+----------------------+--+
| la_bedrooms.bedrooms | la_bedrooms.no_of_listings | la_bedrooms.no_of_bookings  |  la_bedrooms.ratio   |
+----------------------+----------------------------+-----------------------------+----------------------+--+
| 1                    | 17937                      | 872082                      | 48.619167084796786   |
| 2                    | 4030                       | 168840                      | 41.89578163771712    |
| 3                    | 1726                       | 58351                       | 33.80706836616454    |
| 4                    | 602                        | 19561                       | 32.493355481727576   |
| 5                    | 186                        | 4877                        | 26.22043010752688    |
| 6                    | 52                         | 1840                        | 35.38461538461539    |
| 7                    | 23                         | 623                         | 27.08695652173913    |
| 8                    | 7                          | 230                         | 32.857142857142854   |
| 9                    | 4                          | 13                          | 3.25                 |
| 10                   | 3                          | 27                          | 9.0                  |
+----------------------+----------------------------+-----------------------------+----------------------+--+
10 rows selected (0.224 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**LA_Bedrooms**' and download the file with the name **'la_bedrooms.csv'.**

- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC). Don't forget to choose the current file 'la_bedrooms.csv'.

- **Visualizing the data in Tableau for 'la_bedrooms.csv' file,**

a. Change Bedrooms Measure to Dimension by right clicking on Bedrooms.
b. Drag Bedrooms to Columns and Ratio to Rows.
c. Choose **Circle Views** chart.
d. You can see the below chart,

→ **We see that 1 and 2 bedroom Listing configurations are the most sought after in LA.**

Supply and Demand - Guest group configuration

We are determining the common group size of visitors in LA.

- **We are creating a table 'LA_Accommodates' to store the ratio of 'No.of.bookings' to 'No.Of.Listings' for each accommodation configuration in LA.**

```
CREATE TABLE LA_Accommodates ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/LA_Accommodates' AS SELECT Accommodates,COUNT(distinct la.Listing_ID) AS No_Of_Listings,COUNT(Review_Date) AS NO_Of_Bookings,(COUNT(Review_Date)/COUNT(distinct la.Listing_ID)) AS Ratio FROM LA_Listings la,LA_Dates lad WHERE la.Listing_ID=lad.Listing_ID AND Accommodates IS NOT NULL Group By Accommodates;
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Accommodates;
+-------------------------------+-----------------------------------+-------------------------------+-----------------------+--
| la_accommodates.accommodates  | la_accommodates.no_of_listings    | la_accommodates.no_of_bookings | la_accommodates.ratio |
+-------------------------------+-----------------------------------+-------------------------------+-----------------------+--
| 1                             | 2412                              | 50868                         | 21.08955223880597     |
| 2                             | 9951                              | 518406                        | 52.09586976183298     |
| 3                             | 2586                              | 129816                        | 50.19953596287703     |
| 4                             | 4144                              | 195311                        | 47.131032818532816    |
| 5                             | 1351                              | 57962                         | 42.90303478904515     |
| 6                             | 2011                              | 91314                         | 45.40726006961711     |
| 7                             | 396                               | 16047                         | 40.52272727272727     |
| 8                             | 842                               | 33205                         | 39.43586698337292     |
| 9                             | 110                               | 4199                          | 38.17272727272727     |
| 10                            | 338                               | 13493                         | 39.92011834319526     |
| 11                            | 41                                | 1447                          | 35.292682926829265    |
| 12                            | 118                               | 3549                          | 30.076271186440678    |
| 13                            | 14                                | 602                           | 43.0                  |
| 14                            | 63                                | 2739                          | 43.476190476190474    |
| 15                            | 23                                | 709                           | 30.82608695652174     |
| 16                            | 170                               | 6777                          | 39.86470588235294     |
+-------------------------------+-----------------------------------+-------------------------------+-----------------------+--
16 rows selected (0.084 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one '**LA_Accommodates**' and download the file with the name **'la_accommodates.csv'.**

- **Uploading the file to Tableau,**

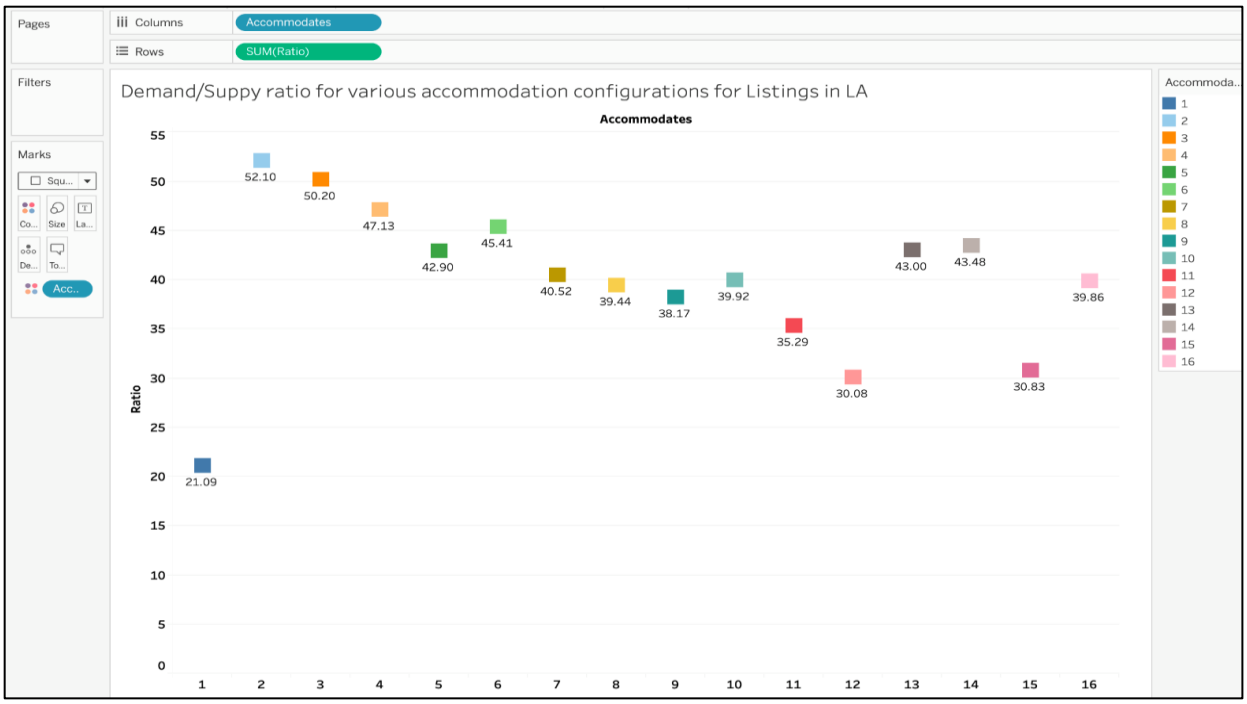
Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC). Don't forget to choose the current file 'la_accommodates.csv'.

- **Visualizing the data in Tableau for 'la_accommodates.csv' file,**

a. Drag Accommodates to Columns and Ratio to Rows.

b. Choose Square shaped views.

You can see the below chart,



→ **We see that the listings that accommodate 2-3 people have a higher Demand/Supply ratio followed by listings that accommodate 4 people.**

Analysis 9:

What it takes to become a Superhost?

Airbnb awards the title of "Superhost" to a small fraction of its dependable hosts. The superhost gets more business in the form of higher bookings, the customer gets improved service and Airbnb gets happy satisfied customers.
In the first part we are finding the percentage of superhosts in both NYC and LA County.

- **We are creating two tables Superhost_LA and Superhost_NY,**

```
CREATE TABLE Superhost_LA ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/Superhost_LA' AS SELECT Listing_ID,Host_Response_Rate,Review_Scores_Rating,Superhost FROM LA_Listings WHERE Host_Response_Rate != 'NA';
```

```
CREATE TABLE Superhost_NY ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE LOCATION '/user/smurali2/Superhost_NY' AS SELECT Listing_ID,Host_Response_Rate,Review_Scores_Rating,Superhost FROM NY_Listings WHERE Host_Response_Rate != 'NA';
```

- **We can check if the table is created using SHOW TABLES command and also check the contents of the table using SELECT command.**

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM Superhost_LA LIMIT 5;
+--------------------------+-----------------------------------+-----------------------------------+--------------------------+--+
| superhost_la.listing_id  | superhost_la.host_response_rate   | superhost_la.review_scores_rating | superhost_la.superhost   |
+--------------------------+-----------------------------------+-----------------------------------+--------------------------+--+
| 3230382                  | 1                                 | 95                                | true                     |
| 18211034                 | 1                                 | 0                                 | false                    |
| 12209039                 | 1                                 | 0                                 | false                    |
| 10339306                 | 0                                 | 100                               | false                    |
| 16114973                 | 1                                 | 100                               | false                    |
+--------------------------+-----------------------------------+-----------------------------------+--------------------------+--+
5 rows selected (0.08 seconds)
```

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM Superhost_NY LIMIT 5;
+--------------------------+-----------------------------------+-----------------------------------+--------------------------+--+
| superhost_ny.listing_id  | superhost_ny.host_response_rate   | superhost_ny.review_scores_rating | superhost_ny.superhost   |
+--------------------------+-----------------------------------+-----------------------------------+--------------------------+--+
| 2515                     | 1                                 | 96                                | false                    |
| 2539                     | 1                                 | 89                                | false                    |
| 2595                     | 1                                 | 90                                | false                    |
| 3330                     | 0.7                               | 85                                | false                    |
| 3647                     | 1                                 | 95                                | false                    |
+--------------------------+-----------------------------------+-----------------------------------+--------------------------+--+
5 rows selected (0.085 seconds)
```

- **After creating the table, we need to download the file into our local system to visualize it in Tableau,**

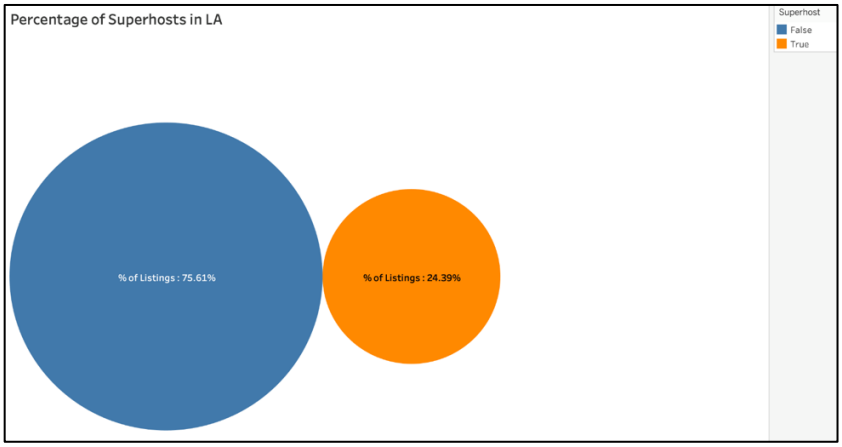
Note: **Follow steps (a),(b), (c) on pages 17,18 and 19.** Do not forget to change the name of the directory to the current one 'Superhost_LA' and download the file with the name **'superhost_la.csv'.**

- **Uploading the file to Tableau,**

Note: Follow the steps (1,2,3,4) on Page 21 of Analysis 2 (NYC). Don't forget to choose the current file 'superhost_la.csv'.

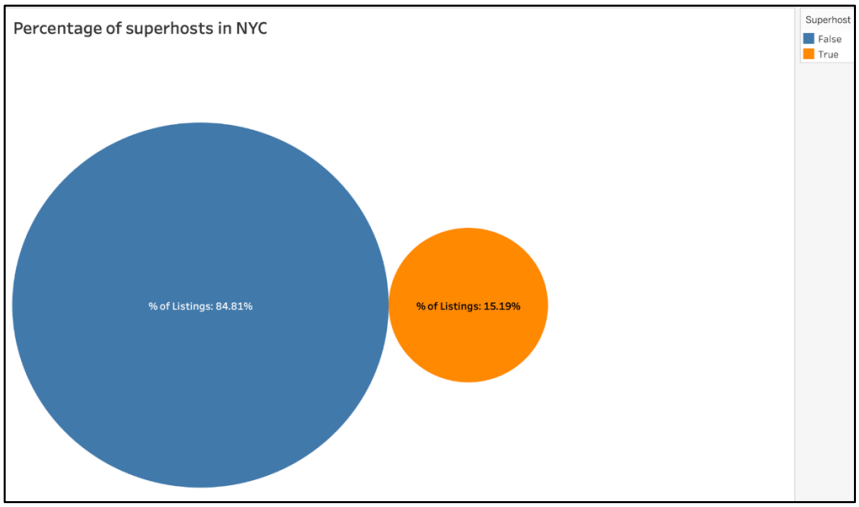
- **Visualizing the data in Tableau for 'superhost_la.csv' file,**

a. Drag Superhost to Columns and Listing ID to Rows.
b. Select Count of Listing ID from drop down.
c. Drag Superhost to Marks field and choose Color to differentiate values True and False for the Superhosts.
d. Drag Count(Listing ID) to Marks field and choose Percent of Total(Under Quick Table Calculation) in the dropdown. This gives the percentage of superhosts and ordinary hosts out of total listings.
e. Choose Packed bubbles chart.
f. You can see the below chart,



Follow the above process for superhost_ny.csv file.

You can see the below chart,



→ **We see that the percentage of Superhosts is high in LA County when compared to NYC. So let us proceed with LA County Listings for the next analysis.**

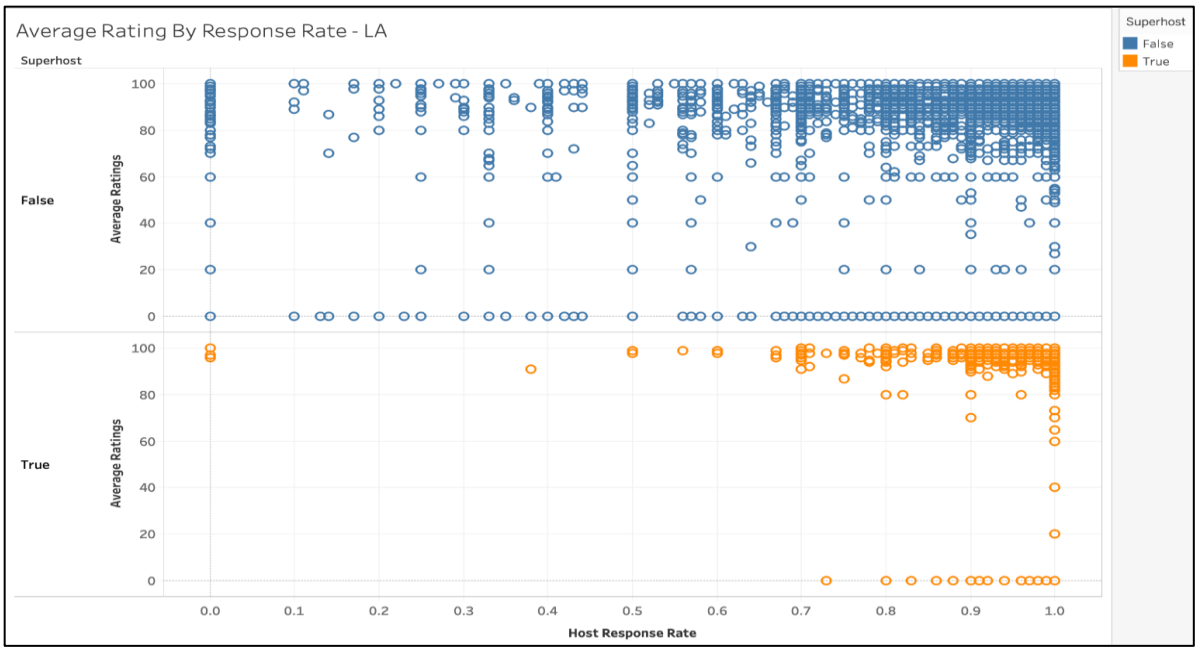
Airbnb's site has a set of requirements that must be fulfilled in order to become a Superhost. Maintaining a review rate above 50%, a response rate above 90% and many other factors.

In this part we are plotting Average Ratings vs Host_Response_Rate for the Listings in LA.

- **Visualization in Tableau,**

a. Choose superhost_la as the data source.
b. Drag Host Response Rate to Columns and change it to dimension.
c. Drag Average Ratings and Superhost to Rows and change Average Ratings to Dimension.
d. Drag Superhost to Marks field and choose Color to differentiate Superhosts and Ordinary hosts.

You can see the below chart,



Our findings, while mostly in line with the Airbnb's criteria, also show some outliers. While most super-hosts are in the high-rating: high-response-rate region, we can also see a few super hosts with response rates less than 75% (which violates the 90%+ criteria set by Airbnb). This is a very small part of the hosts.

In terms of Ratings, there are many hosts rated above 80%. With that being said, most Airbnb hosts lie in the high-rating: high-response region, but only a small fraction get to be super hosts. So we can conclude that becoming a Super host takes a lot more than high ratings & response rates.

We also look into the reviews received by Superhosts in LA

- We create a table LA_Host for Listings with maximum Host Response Rate and Ratings and their corresponding reviews.

```
CREATE TABLE LA_Host ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' STORED AS TEXTFILE
LOCATION '/user/smurali2/LA_Host' AS SELECT
SLA.Listing_ID,SLA.Host_Response_Rate,SLA.Review_Scores_Rating,SLA.Superhost,regexp_replace(ar.co
mments,',','') AS Comments FROM Superhost_LA SLA,airbnb_reviews ar WHERE
SLA.Listing_ID=ar.Listing_ID AND (Review_Scores_Rating>90 AND Host_Response_Rate = 1) AND
Superhost=True ORDER BY SLA.Listing_ID;
```

```
0: jdbc:hive2://bigdai1-bdcsce-1:2181,bigdai1> SELECT * FROM LA_Host LIMIT 5;
+---------------------+---------------------------+-------------------------------+---------------------+---------------------------------------------------------------------------------------------------------+
| la_host.listing_id  | la_host.host_response_rate  | la_host.review_scores_rating  | la_host.superhost  |
                                                                                                                                   la_host.comments
                                                                   |
+---------------------+---------------------------+-------------------------------+---------------------+---------------------------------------------------------------------------------------------------------+
| 5728               | 1                          | 94                            | true               | Great location very quiet and low key - which allowed me to be busy with both work and
getting in the ocean.  Lovely people - though didn't see folks much by my choice needing a retreat in sunny LA.
                                                                   |
| 5728               | 1                          | 94                            | true               | Sanni's place is very cozy clean and so inspiring. Great location and quiet and friendl
y neighborhood too. You will love it!
                                                                   |
| 5728               | 1                          | 94                            | true               | Sanni and family made me feel right at home in their fun backyard oasis. The room was c
lean quiet and very charming. The instructions were clear and easy to follow. Minutes walk or a $5 uber ride to Venice the space had everything I would ever need for a California visit. Highly r
ecommended!
                                                                   |
| 5728               | 1                          | 94                            | true               | The Circle had a pretty groovy vibe. What a great start to my journey to Hawaii the nex
t day! It was even better waking up to the view of a beautiful waning moon. Sanni offered us a bagel and some fruit for breakfast like we were family. Super sweet folks and beautiful energy amon
gst the whole family. Her fashions are to die for so bring a few extra bucks!!! Easy to get to (Super shuttle from the airport for $19) and a block away from public transportation that'll take y
ou down to the beach ($1.50). All in all great experience.  |
| 5728               | 1                          | 94                            | true               | We had a super quiet and relaxing time in The Circle Treehouse but it's definitely a pl
ace to stay for a certain sort of person. My girlfriend and I are pretty laid back and adaptable so we were able to deal with the heat and rustic-ness of the treehouse pretty easily. But it's de
finitely an un-insulated treehouse. Totally worth the money but definitely be ready for a treehouse experience. (Also: bring an extra pillow!)
```

- We are generating n-grams from the reviews to find the positive factors for the Super hosts using the below query,

select explode(ngrams(sentences(lower(comments)),4,40)) from la_host;

We can see the below output,

```
+------------------------------------------------------------------------+--+
|                                  col                                   |
+------------------------------------------------------------------------+--+
| {"ngram":["was","a","great","host"],"estfrequency":6513.0}             |
| {"ngram":["definitely","stay","here","again"],"estfrequency":4907.0}   |
| {"ngram":["we","had","a","great"],"estfrequency":4581.0}               |
| {"ngram":["had","a","great","time"],"estfrequency":3747.0}             |
| {"ngram":["would","definitely","stay","here"],"estfrequency":3706.0}   |
| {"ngram":["had","everything","we","needed"],"estfrequency":3672.0}     |
| {"ngram":["in","a","great","location"],"estfrequency":3522.0}          |
| {"ngram":["was","very","clean","and"],"estfrequency":3439.0}           |
| {"ngram":["great","place","to","stay"],"estfrequency":3340.0}          |
| {"ngram":["had","a","great","stay"],"estfrequency":3183.0}             |
| {"ngram":["is","a","great","host"],"estfrequency":3161.0}              |
| {"ngram":["i","would","definitely","stay"],"estfrequency":2951.0}      |
| {"ngram":["a","great","host","and"],"estfrequency":2620.0}             |
| {"ngram":["i","would","highly","recommend"],"estfrequency":2409.0}     |
| {"ngram":["a","great","place","to"],"estfrequency":2385.0}             |
| {"ngram":["i","would","definitely","recommend"],"estfrequency":2270.0} |
| {"ngram":["i","had","a","great"],"estfrequency":2182.0}                |
| {"ngram":["this","was","my","first"],"estfrequency":2175.0}            |
| {"ngram":["we","had","a","wonderful"],"estfrequency":2157.0}           |
| {"ngram":["home","away","from","home"],"estfrequency":2116.0}          |
| {"ngram":["place","to","stay","in"],"estfrequency":2050.0}             |
| {"ngram":["easy","to","communicate","with"],"estfrequency":2039.0}     |
| {"ngram":["would","stay","here","again"],"estfrequency":1997.0}        |
| {"ngram":["to","the","beach","and"],"estfrequency":1969.0}             |
| {"ngram":["would","definitely","stay","again"],"estfrequency":1950.0}  |
| {"ngram":["close","to","the","beach"],"estfrequency":1877.0}           |
| {"ngram":["it","was","a","great"],"estfrequency":1843.0}               |
| {"ngram":["to","stay","here","again"],"estfrequency":1835.0}           |
| {"ngram":["thank","you","so","much"],"estfrequency":1737.0}            |
| {"ngram":["had","a","wonderful","stay"],"estfrequency":1712.0}         |
| {"ngram":["my","husband","and","i"],"estfrequency":1657.0}             |
| {"ngram":["we","had","everything","we"],"estfrequency":1648.0}         |
| {"ngram":["one","of","the","best"],"estfrequency":1615.0}              |
| {"ngram":["was","a","wonderful","host"],"estfrequency":1609.0}         |
| {"ngram":["we","were","able","to"],"estfrequency":1607.0}              |
| {"ngram":["the","location","was","perfect"],"estfrequency":1596.0}     |
| {"ngram":["we","would","definitely","stay"],"estfrequency":1591.0}     |
| {"ngram":["was","exactly","as","described"],"estfrequency":1589.0}     |
| {"ngram":["the","location","is","great"],"estfrequency":1569.0}        |
| {"ngram":["definitely","stay","there","again"],"estfrequency":1559.0}  |
+------------------------------------------------------------------------+--+
40 rows selected (10.835 seconds)
```

We see that LA Superhosts have many positive reviews with respect to host, cleanliness, location, communication and genuineness of the property. These are some of the factors required to become a Super Host.

References:

1. **GitHub Link:** https://github.com/Samyuktha-M/5200-project
2. **Dataset URL:** https://www.kaggle.com/samyukthamurali/airbnb-ratings-dataset
3. **Dataset Source URL:**
   - http://insideairbnb.com/get-the-data.html
   - https://public.opendatasoft.com/explore/dataset/airbnb-ratings/table/?disjunctive.city&disjunctive.neighbourhood_cleansed&sort=number_of_reviews
   - https://public.opendatasoft.com/explore/dataset/airbnb-reviews/table/
4. **References URL:**
   - http://www.columbia.edu/~sg3637/airbnb_final_analysis.html
   - https://towardsdatascience.com/airbnb-rental-listings-dataset-mining-f972ed08ddec
   - https://nycdatascience.com/blog/student-works/analysis-and-machine-learning-modeling-of-new-york-city-airbnb-data/