CIS 5200 Term Project Tutorial

Authors: Ragi Dave, An Mach, Ankita Hasmukhbhai Savaliya, Bhumika Suvagia

Instructor: Dr. Jongwook Woo

Date: 05/21/2023

Lab Tutorial

Ragi Dave (rdave4@calstatela.edu)

An Mach (amach3@calstatela.edu)

Ankita Hasmukhbhai Savaliya (asavali2@calstatela.edu)

Bhumika Suvagia (bsuvagi@calstatela.edu)

05/21/2023

Flight Prices Analysis using Hive

Objectives

In this hands-on lab, you will learn how to:

- Upload the data into HDFS using -put command.
- Create Hive tables and perform data engineering in Hive using HiveQL
- Query data using HiveQL
- Secure copy tables created with HiveQL from Hadoop to local computers
- Create visualizations with Excel, Tableau, and Power BI

Platform Spec

hdfs version

```
-bash-4.2$ hdfs version
Hadoop 3.1.2
Source code repository ssh://git@bitbucket.oci.oraclecorp.com:7999/bdcs/apache_bigtop.git -r 4100eb8d8581c4328601079ff5af522f95e9977f
Compiled by root on 2023-02-27T08:26Z
Compiled with protoc 2.5.0
From source with checksum b367ca15864aef16725a3035859c9ece
This command was run using /usr/odh/1.1.5/hadoop/hadoop-common-3.1.2.jar
```

lscpu

```
-bash-4.2$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                8
On-line CPU(s) list:   0-7
Thread(s) per core:    2
Core(s) per socket:    4
Socket(s):             1
NUMA node(s):          1
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel(R) Xeon(R) Platinum 8167M CPU @ 2.00GHz
Stepping:              4
CPU MHz:               1995.312
BogoMIPS:              3990.62
Virtualization:        VT-x
Hypervisor vendor:     KVM
Virtualization type:   full
L1d cache:             32K
L1i cache:             32K
L2 cache:              4096K
L3 cache:              16384K
NUMA node0 CPU(s):     0-7
```

hdfs dfsadmin -report

```
-bash-4.2$ hdfs dfsadmin -report
Configured Capacity: 419520548352 (390.71 GB)
Present Capacity: 417859894557 (389.16 GB)
DFS Remaining: 42330897693 (39.42 GB)
DFS Used: 375528996864 (349.74 GB)
DFS Used%: 89.87%
Replicated Blocks:
        Under replicated blocks: 0
        Blocks with corrupt replicas: 0
        Missing blocks: 0
        Missing blocks (with replication factor 1): 0
        Low redundancy blocks with highest priority to recover: 0
        Pending deletion blocks: 0
Erasure Coded Block Groups:
        Low redundancy block groups: 0
        Block groups with corrupt internal blocks: 0
        Missing block groups: 0
        Low redundancy blocks with highest priority to recover: 0
        Pending deletion blocks: 0

-------------------------------------------------
report: Access denied for user amach3. Superuser privilege is required
```

- Cluster Version: Hadoop 3.1.2
- CPU Speed: 1995.312 MHz
- Number of CPU cores: 8 cores * 5 nodes = 40 cores
- Number of nodes: 5 (2 Master and 3 Worker)
- Total Memory Size: 390.71 GB

Dataset Details

- DATASET NAME: Flight Prices
- DATASET URL:https://www.kaggle.com/datasets/dilwong/flightprices

https://www.kaggle.com/datasets/mike90/airport-codes

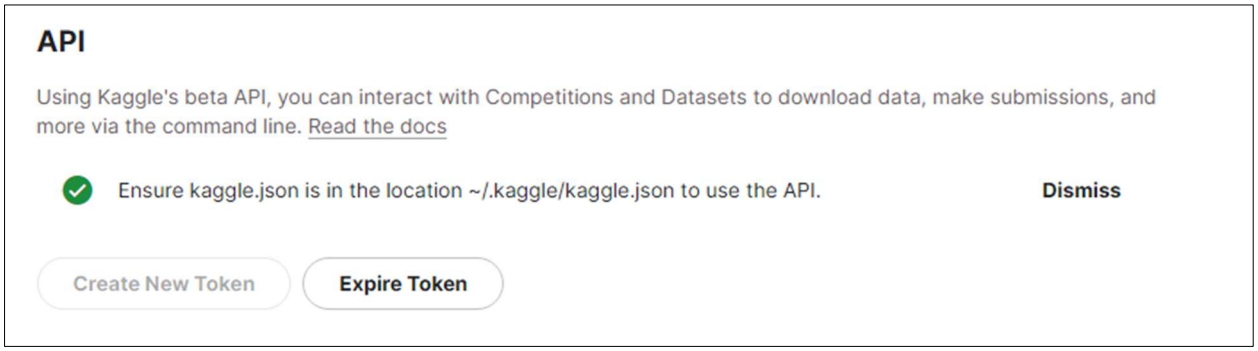
- TOTAL SIZE: 31.09 GB
- COUNTRIES CONSIDERED:USA
- NUMBER OF FILES: 2
- FILE FORMAT: CSV
- GITHUB LINK: https://github.com/amach3/Flight-Price-Analysis.git

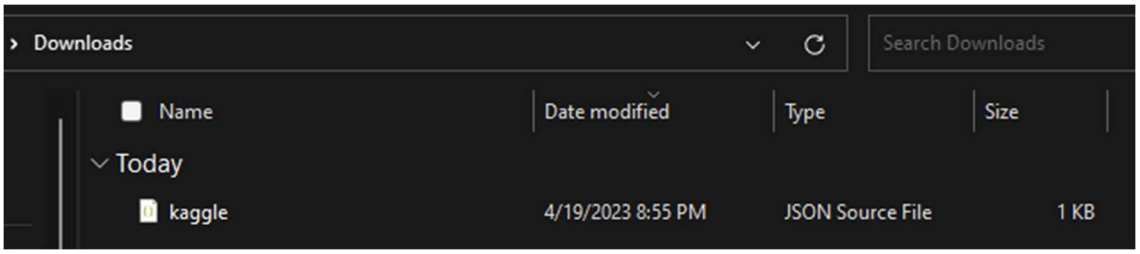
Step 1: Download Kaggle dataset in Google Colab

Instruction was found at https://www.kaggle.com/general/156610.

Note: We realized the steps in this section are not necessary in this project. We could have downloaded the zip file directly from Kaggle then unzip the file on a local computer. You can skip to Step 2 section.

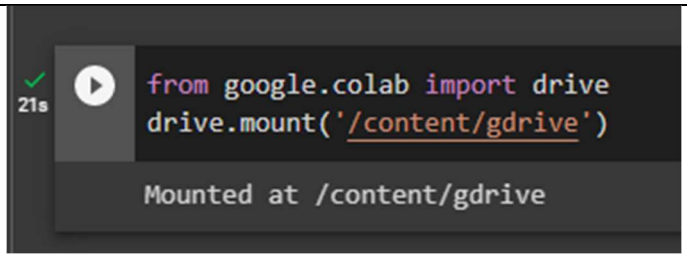
1. Go to your kaggle account, Scroll to API section and Click Expire API Token to remove previous tokens
2. Click on Create New API Token - It will download kaggle.json file on your machine.





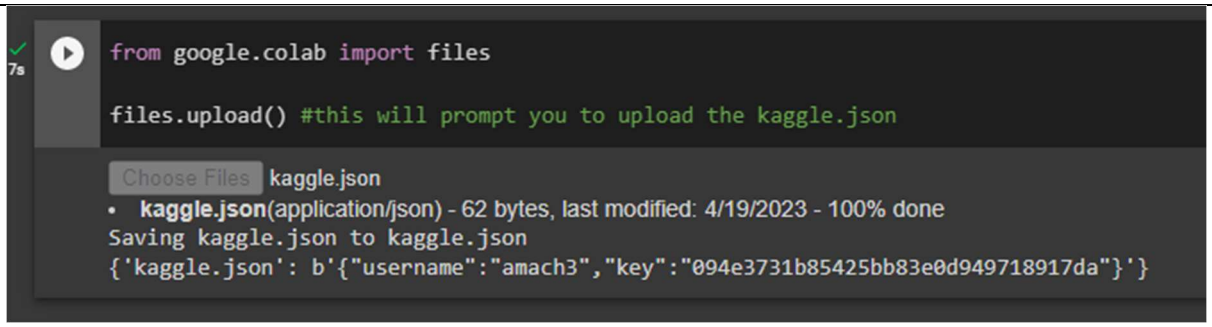
3.Go to your Google Colab project file and run the following commands to mount your Google Drive files Following code make mount your google drive

```
from google.colab import drive
drive.mount('/content/gdrive')
```



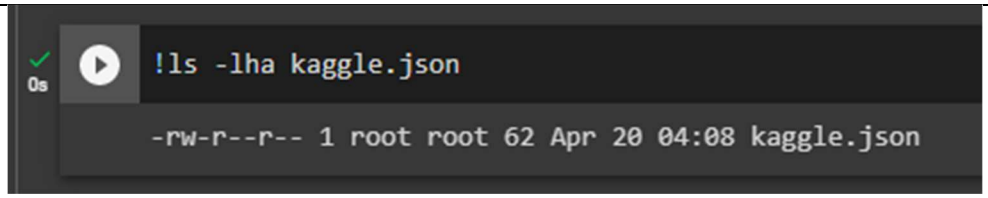
• Now upload the kaggle.json file

```
from google.colab import files
files.upload() #this will prompt you to upload the kaggle.json
```



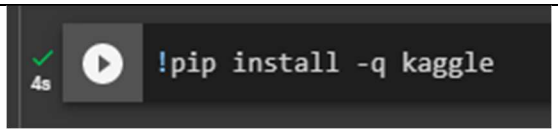
• make sure kaggle.json file is present

```
!ls -lha kaggle.json
```



• Install kaggle API client

```
!pip install -q kaggle
```

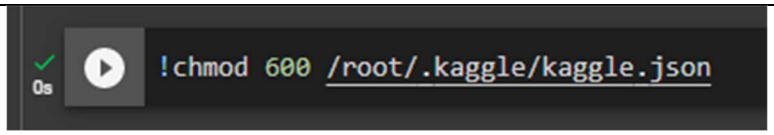


• kaggle API client expects the file to be in ~/.kaggle
• so move it there

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
```

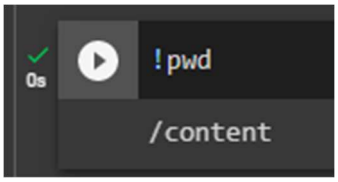
• we need to set permissions

```
!chmod 600 /root/.kaggle/kaggle.json
```



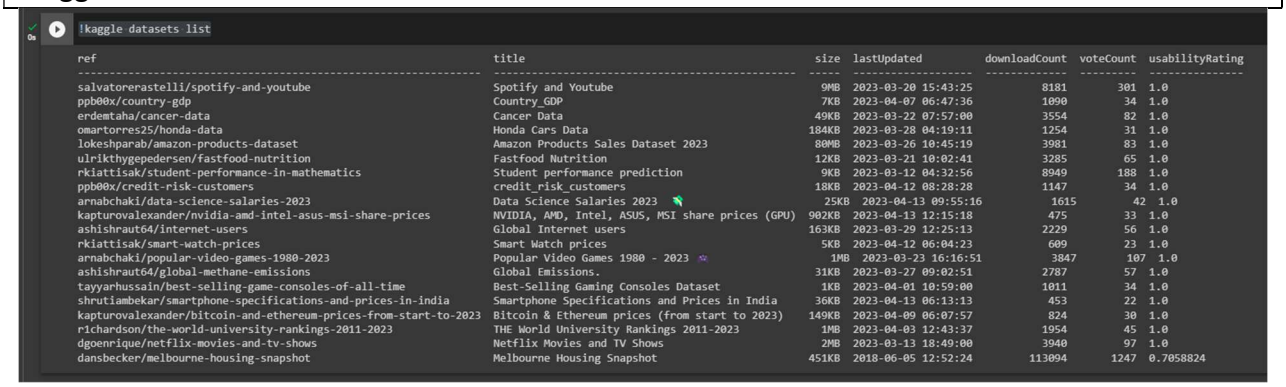
• check your directory before downloading the datasets

```
!pwd
```



• list all available datasets

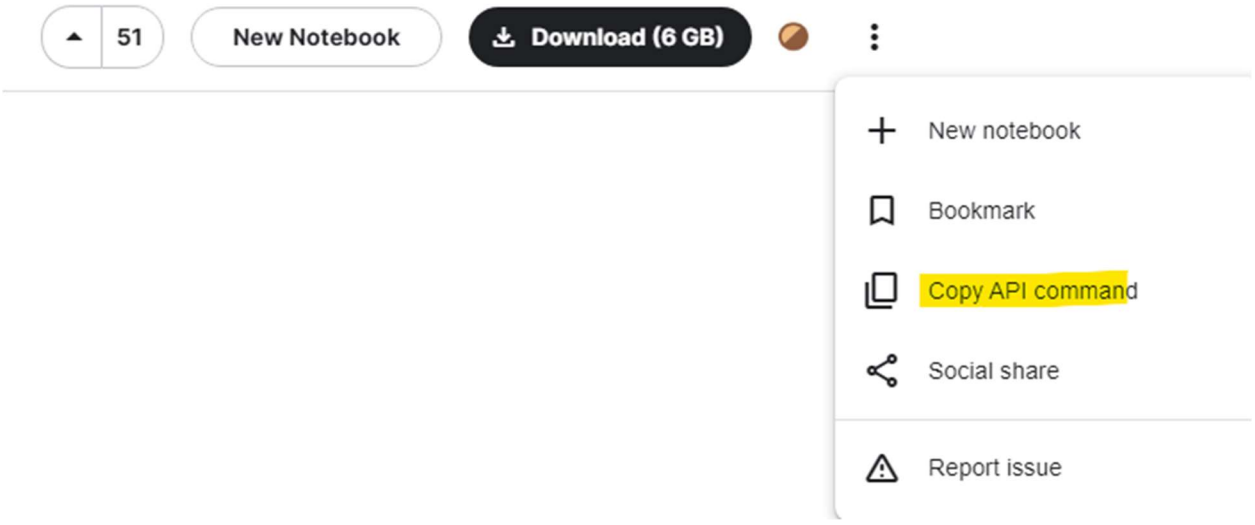
```
!kaggle datasets list
```



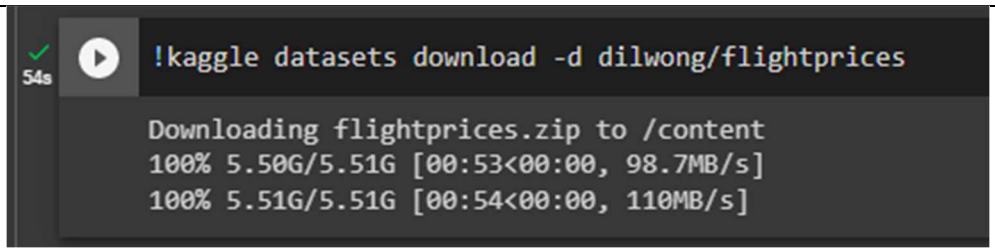
```
!kaggle datasets list

ref                                                             title                                                size  lastUpdated          downloadCount  voteCount  usabilityRating
--------------------------------------------------------------  ---------------------------------------------------  ----  -------------------  -------------  ---------  ---------------
salvatorerastelli/spotify-and-youtube                           Spotify and Youtube                                   9MB  2023-03-20 15:43:25           8181        301  1.0
ppb00x/country-gdp                                              Country_GDP                                           7KB  2023-04-07 06:47:36           1090         34  1.0
erdemtaha/cancer-data                                           Cancer Data                                          49KB  2023-03-22 07:57:00           3554         82  1.0
omartorres25/honda-data                                         Honda Cars Data                                     184KB  2023-03-28 04:19:11           1254         31  1.0
lokeshparab/amazon-products-dataset                             Amazon Products Sales Dataset 2023                   80MB  2023-03-26 10:45:19           3981         83  1.0
ulrikthygepedersen/fastfood-nutrition                           Fastfood Nutrition                                   12KB  2023-03-21 10:02:41           3285         65  1.0
rkiattisak/student-performance-in-mathematics                   Student performance prediction                        9KB  2023-03-12 04:32:56           8949        188  1.0
ppb00x/credit-risk-customers                                    credit_risk_customers                                18KB  2023-04-12 08:28:28           1147         34  1.0
arnabchaki/data-science-salaries-2023                           Data Science Salaries 2023                           25KB  2023-04-13 09:55:16           1615         42  1.0
kapturovalexander/nvidia-amd-intel-asus-msi-share-prices         NVIDIA, AMD, Intel, ASUS, MSI share prices (GPU)    902KB  2023-04-13 12:15:18            475         33  1.0
ashishraut64/internet-users                                     Global Internet users                               163KB  2023-03-29 12:25:13           2229         56  1.0
rkiattisak/smart-watch-prices                                   Smart Watch prices                                    5KB  2023-04-12 06:04:23            609         23  1.0
arnabchaki/popular-video-games-1980-2023                        Popular Video Games 1980 - 2023                       1MB  2023-03-23 16:16:51           3847        107  1.0
ashishraut64/global-methane-emissions                           Global Emissions.                                    31KB  2023-03-27 09:02:51           2787         57  1.0
tayyarhussain/best-selling-game-consoles-of-all-time            Best-Selling Gaming Consoles Dataset                  1KB  2023-04-01 10:59:00           1011         34  1.0
shrutiambekar/smartphone-specifications-and-prices-in-india     Smartphone Specifications and Prices in India        36KB  2023-04-13 06:13:13            453         22  1.0
kapturovalexander/bitcoin-and-ethereum-prices-from-start-to-2023 Bitcoin & Ethereum prices (from start to 2023)     149KB  2023-04-09 06:07:57            824         30  1.0
r1chardson/the-world-university-rankings-2011-2023              THE World University Rankings 2011-2023               1MB  2023-04-03 12:43:37           1954         45  1.0
dgoenrique/netflix-movies-and-tv-shows                          Netflix Movies and TV Shows                           2MB  2023-03-13 18:49:00           3940         97  1.0
dansbecker/melbourne-housing-snapshot                           Melbourne Housing Snapshot                          451KB  2018-06-05 12:52:24         113094       1247  0.7058824
```

• download the required dataset from kaggle by copying the API command from the dataset page (https://www.kaggle.com/datasets/dilwong/flightprices?select=itineraries.csv)

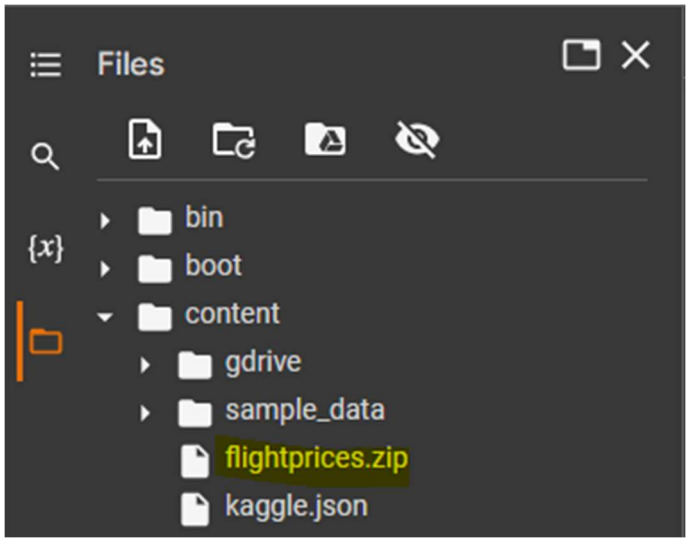



```
!kaggle datasets download -d dilwong/flightprices
```

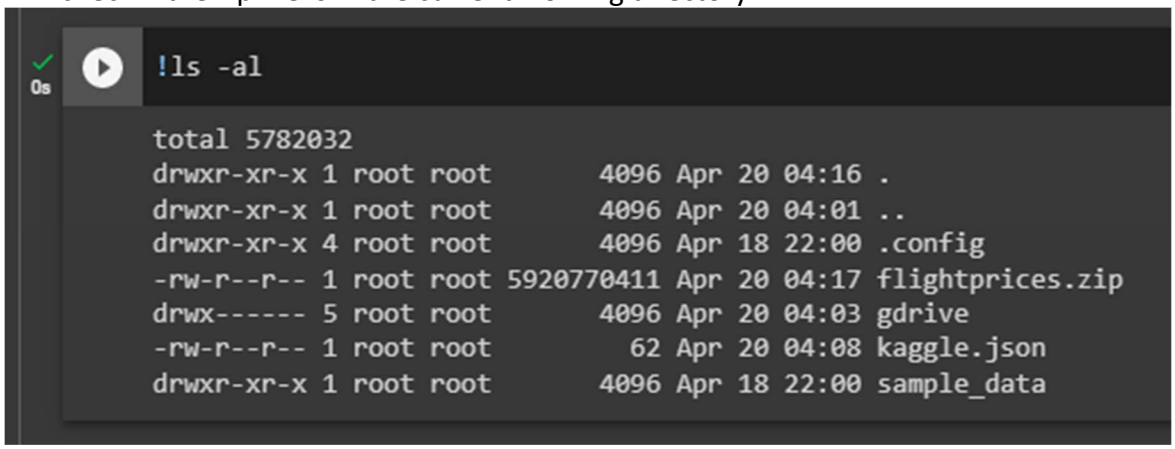


```
!kaggle datasets download -d dilwong/flightprices

Downloading flightprices.zip to /content
100% 5.50G/5.51G [00:53<00:00, 98.7MB/s]
100% 5.51G/5.51G [00:54<00:00, 110MB/s]
```

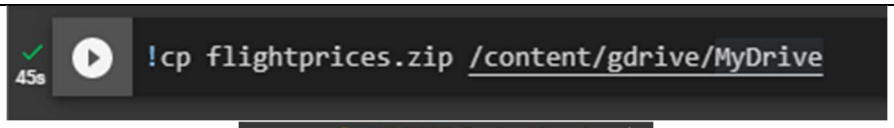


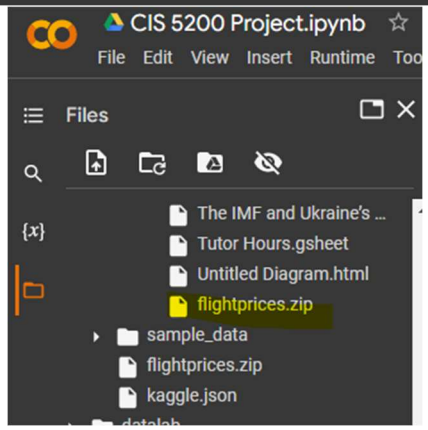
- check if the zip file is in the current working directory



- Copy the zip file to Google Drive

```
!cp flightprices.zip /content/gdrive/MyDrive
```

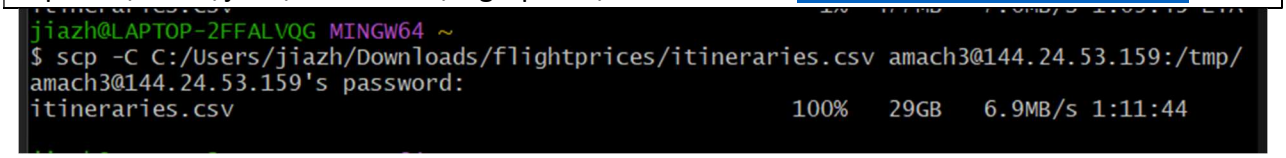




Step 2: Secure copy csv files from local computer to Linux serve

- Download the flightprices.zip file from Google Drive then unzip it in Download folder, then secure copy it to tmp folder on Linux server.

```
scp -C C:/Users/jiazh/Downloads/flightprices/itineraries.csv amach3@144.24.53.159:/tmp
```



- Download the airport.csv file from Kaggle and unzip it in Download folder. Then secure copy it to linux.

```
scp C:/Users/jiazh/Downloads/airportcode/airports.csv
amach3@144.24.53.159:/home/amach3
```

Step 3: Copy files from Linux server to HDFS

- Make FlightData directory and put itineraries.csv file here

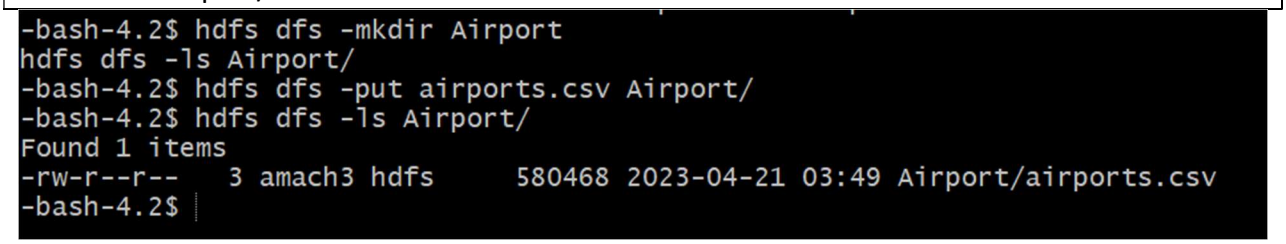
```
hdfs dfs -mkdir FlightData
hdfs dfs -put /tmp/itineraries.csv /user/amach3/FlightData/
hdfs dfs -ls FlightData/
```

- Make the FlightData directory public

```
hdfs dfs -chmod -R go+rx /user/amach3/FlightData
```

- Make Airport directory and put Airport.csv file here

```
hdfs dfs -mkdir Airport
hdfs dfs -put airports.csv Airport/
hdfs dfs -ls Airport/
```



- make the Airport directory public

```
hdfs dfs -chmod -R go+rx /user/amach3/Airport
```

Step 4: Create tables in Hive

1. Airport table

```
Beeline
Use amach3;
DROP TABLE IF EXISTS Airport;
CREATE EXTERNAL TABLE IF NOT EXISTS Airport
(Name string, City string, Country string, IATA string, ICAO string,
Latitude double, Longitude double)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/amach3/Airport/'
TBLPROPERTIES ('skip.header.line.count'='1');

SELECT * FROM Airport LIMIT 10;
```

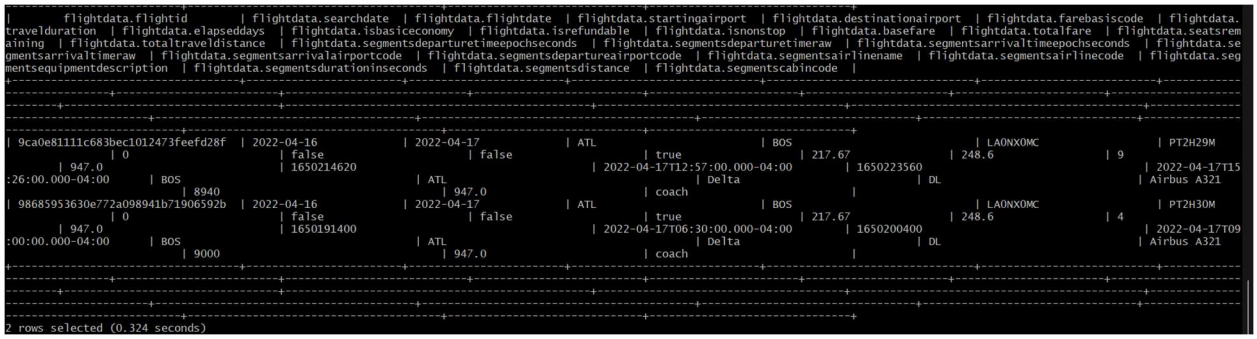
airport.name	airport.city	airport.country	airport.iata	airport.icao	airport.latitude	airport.longitude
Goroka Airport	Goroka	Papua New Guinea	GKA	AYGA	-6.081689835	145.3919983
Madang Airport	Madang	Papua New Guinea	MAG	AYMD	-5.207079887	145.7890015
Mount Hagen Kagamuga Airport	Mount Hagen	Papua New Guinea	HGU	AYMH	-5.826789856	144.2960052
Nadzab Airport	Nadzab	Papua New Guinea	LAE	AYNZ	-6.569803	146.725977
Port Moresby Jacksons International Airport	Port Moresby	Papua New Guinea	POM	AYPY	-9.443380356	147.2200012
Wewak International Airport	Wewak	Papua New Guinea	WWK	AYWK	-3.583830118	143.6690063
Narsarsuaq Airport	Narssarssuaq	Greenland	UAK	BGBW	61.16049957	-45.42599869
Godthaab / Nuuk Airport	Godthaab	Greenland	GOH	BGGH	64.19090271	-51.67810059
Kangerlussuaq Airport	Sondrestrom	Greenland	SFJ	BGSF	67.0122219	-50.71160316
Thule Air Base	Thule	Greenland	THU	BGTL	76.53119659	-68.70320129

2. FlightData table

```
DROP TABLE IF EXISTS FlightData;

CREATE EXTERNAL TABLE IF NOT EXISTS FlightData
(FlightID string, SearchDate date, FlightDate date, StartingAirport string,
DestinationAirport string, FareBasisCode string, TravelDuration string,
ElapsedDays int, IsBasicEconomy boolean, IsRefundable boolean,
IsNonStop boolean, BaseFare double, TotalFare double, SeatsRemaining int,
TotalTravelDistance double, SegmentsDepartureTimeEpochSeconds string,
SegmentsDepartureTimeRaw string, SegmentsArrivalTimeEpochSeconds string,
SegmentsArrivalTimeRaw string, SegmentsArrivalAirportCode string,
SegmentsDepartureAirportCode string, SegmentsAirlineName string,
SegmentsAirlineCode string, SegmentsEquipmentDescription string,
SegmentsDurationInSeconds string, SegmentsDistance double,
SegmentsCabinCode string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/amach3/FlightData/'
TBLPROPERTIES ('skip.header.line.count'='1');

SELECT * FROM FlightData LIMIT 2;
```



```
| flightdata.flightid | flightdata.searchdate | flightdata.flightdate | flightdata.startingairport | flightdata.destinationairport | flightdata.farebasiscode | flightdata.travelduration | flightdata.elapseddays | flightdata.isbasiceconomy | flightdata.isrefundable | flightdata.isnonstop | flightdata.basefare | flightdata.totalfare | flightdata.seatsremaining | flightdata.totaltraveldistance | flightdata.segmentsdeparturetimeepochseconds | flightdata.segmentsdeparturetimeraw | flightdata.segmentsarrivaltimeepochseconds | flightdata.segmentsarrivaltimeraw | flightdata.segmentsarrivalairportcode | flightdata.segmentsdepartureairportcode | flightdata.segmentsairlinename | flightdata.segmentsairlinecode | flightdata.segmentsequipmentdescription | flightdata.segmentsdurationinseconds | flightdata.segmentsdistance | flightdata.segmentscabincode |
| 9ca0e81111c683bec1012473feefd28f | 2022-04-16 | 2022-04-17 | ATL | BOS | LA0NX0MC | PT2H29M | 0 | false | false | true | 217.67 | 248.6 | 9 | 947.0 | 1650214620 | 2022-04-17T12:57:00.000-04:00 | 1650223560 | 2022-04-17T15:26:00.000-04:00 | BOS | ATL | Delta | DL | Airbus A321 | 8940 | 947.0 | coach |
| 98685953630e772a098941b71906592b | 2022-04-16 | 2022-04-17 | ATL | BOS | LA0NX0MC | PT2H30M | 0 | false | false | true | 217.67 | 248.6 | 4 | 947.0 | 1650191400 | 2022-04-17T06:30:00.000-04:00 | 1650200400 | 2022-04-17T09:00:00.000-04:00 | BOS | ATL | Delta | DL | Airbus A321 | 9000 | 947.0 | coach |
2 rows selected (0.324 seconds)
```

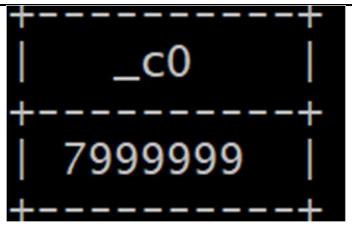
Since the dataset is too large, Dr. Woo advised us to reduce our data size to 2 – 3 GB by sampling our data.

```
SELECT COUNT(*) FROM FlightData; --82138753 rows
--10% is about 8,000,000 rows

CREATE EXTERNAL TABLE IF NOT EXISTS FlightData2
(FlightID string, SearchDate date, FlightDate date, StartingAirport string,
DestinationAirport string, FareBasisCode string, TravelDuration string,
ElapsedDays int, IsBasicEconomy boolean, IsRefundable boolean,
IsNonStop boolean, BaseFare double, TotalFare double, SeatsRemaining int,
TotalTravelDistance int, SegmentsDepartureTimeEpochSeconds string,
SegmentsDepartureTimeRaw string, SegmentsArrivalTimeEpochSeconds string,
SegmentsArrivalTimeRaw string, SegmentsArrivalAirportCode string,
SegmentsDepartureAirportCode string, SegmentsAirlineName string,
SegmentsAirlineCode string, SegmentsEquipmentDescription string,
SegmentsDurationInSeconds string, SegmentsDistance int,
SegmentsCabinCode string)
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
STORED AS TEXTFILE LOCATION '/user/amach3/FlightData2/'
TBLPROPERTIES ('skip.header.line.count'='1');

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData
WHERE rand() <= 0.1
distribute by rand()
SORT BY rand()
LIMIT 8000000;

SELECT COUNT(*) FROM FlightData2;
```

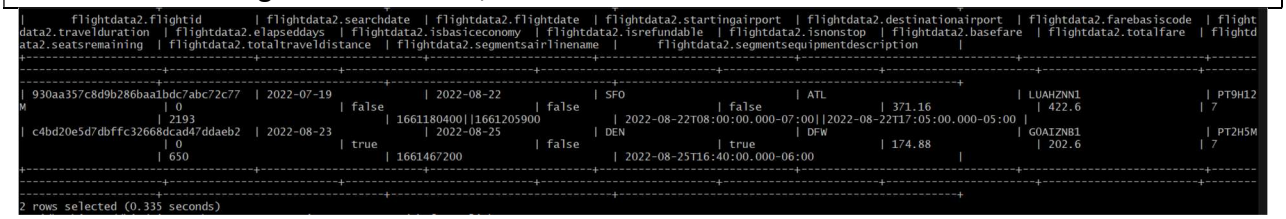


```
+----------+
|   _c0    |
+----------+
| 7999999  |
+----------+
```

Step 5: Data Engineering

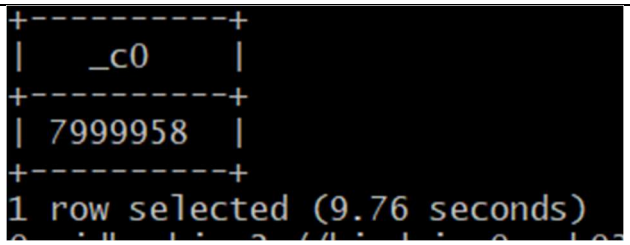
1. Drop unusable columns

```
ALTER TABLE FlightData2 REPLACE COLUMNS (FlightID string, SearchDate date, FlightDate date,
StartingAirport string,
DestinationAirport string, FareBasisCode string, TravelDuration string,
ElapsedDays int, IsBasicEconomy boolean, IsRefundable boolean,
IsNonStop boolean, BaseFare double, TotalFare double, SeatsRemaining int,
TotalTravelDistance int, SegmentsAirlineName string, SegmentsEquipmentDescription string);
SELECT * FROM FlightData2 LIMIT 2;
```



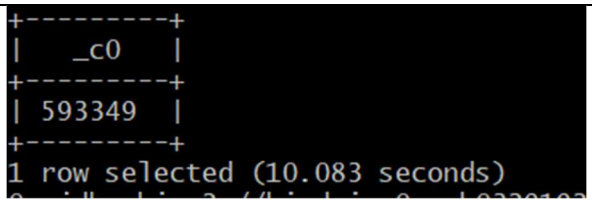
2. Remove any duplicate rows from FlightData2 table

```
INSERT OVERWRITE TABLE  FlightData2
SELECT DISTINCT * FROM FlightData2;
-- Check number of rows after removing duplicates
SELECT COUNT(*) FROM FlightData2;
```

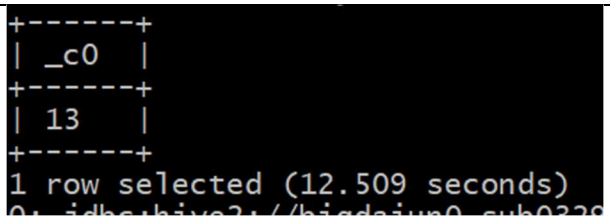


3. Check null values

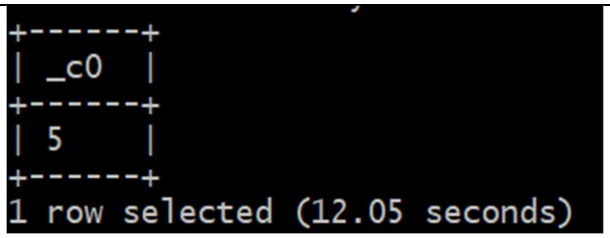
```
SELECT COUNT(*) FROM FlightData2 where TotalTravelDistance IS NULL;
```



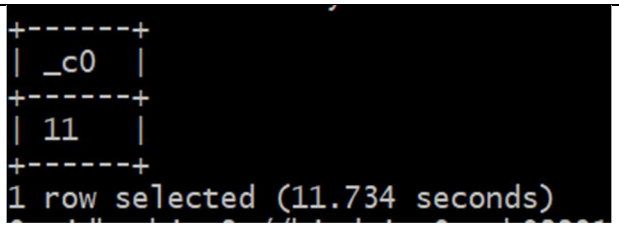

```
SELECT count(*) from FlightData2 where FlightDate IS NULL;
```



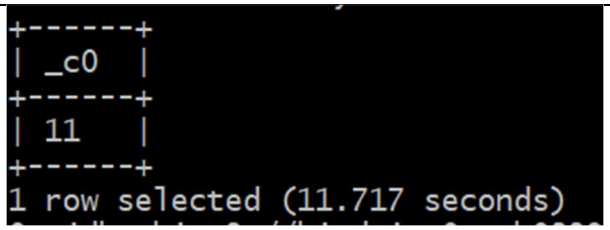
SELECT count(*) from FlightData2 where StartingAirport IS NULL;



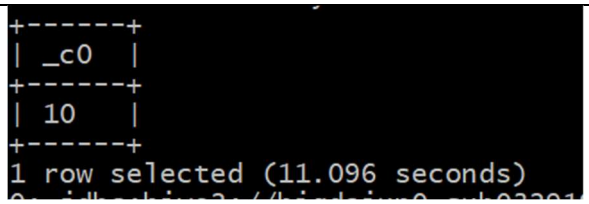
SELECT count(*) from FlightData2 where DestinationAirport IS NULL;



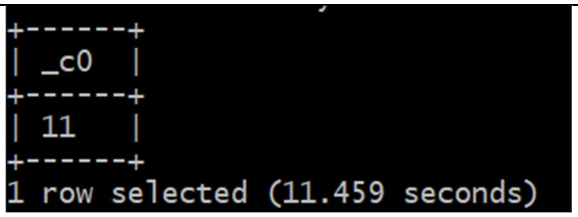
SELECT count(*) from FlightData2 where TravelDuration IS NULL;



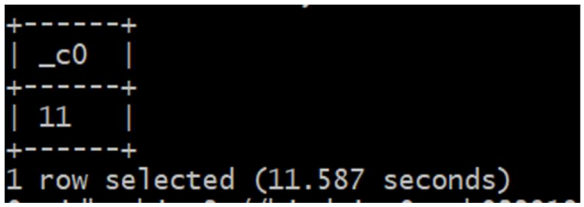
SELECT count(*) from FlightData2 where IsBasicEconomy IS NULL;



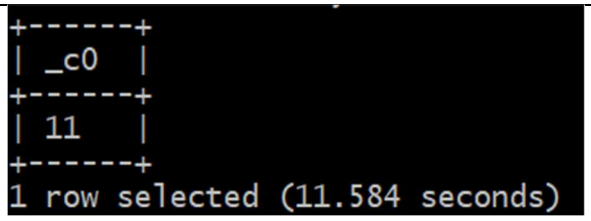
SELECT count(*) from FlightData2 where IsNonStop IS NULL;



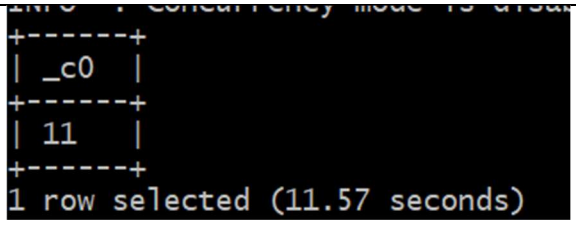
SELECT count(*) from FlightData2 where BaseFare IS NULL;



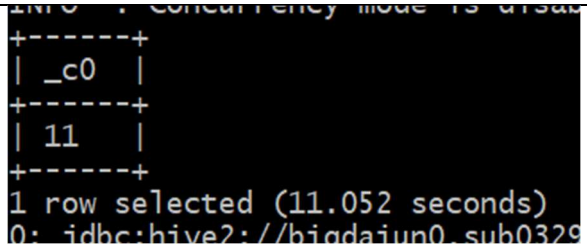

```
SELECT count(*) from FlightData2 where TotalFare IS NULL;
```



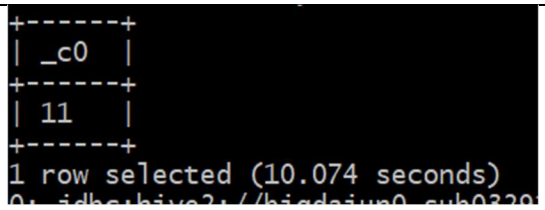

```
SELECT count(*) from FlightData2 where SeatsRemaining IS NULL;
```




```
SELECT count(*) from FlightData2 where SegmentsAirlineName IS NULL;
```




```
SELECT count(*) from FlightData2 where SegmentsEquipmentDescription IS NULL;
```



Since our dataset is huge (almost 8 million rows of data), we decided to remove rows with null values detected above.

```
INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE TotalTravelDistance IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE FlightDate IS NOT NULL;
```

```
INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE StartingAirport IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE DestinationAirport IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE TravelDuration IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE IsBasicEconomy IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE IsNonStop IS NOT NULL;
INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE BaseFare IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE TotalFare IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE SeatsRemaining IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE SegmentsAirlineName IS NOT NULL;

INSERT OVERWRITE TABLE FlightData2
SELECT * FROM FlightData2 WHERE SegmentsEquipmentDescription IS NOT NULL;

SELECT COUNT(*) from FlightData2;
```

```
+----------+
|   _c0    |
+----------+
| 7406035  |
+----------+
1 row selected (12.393 seconds)
```

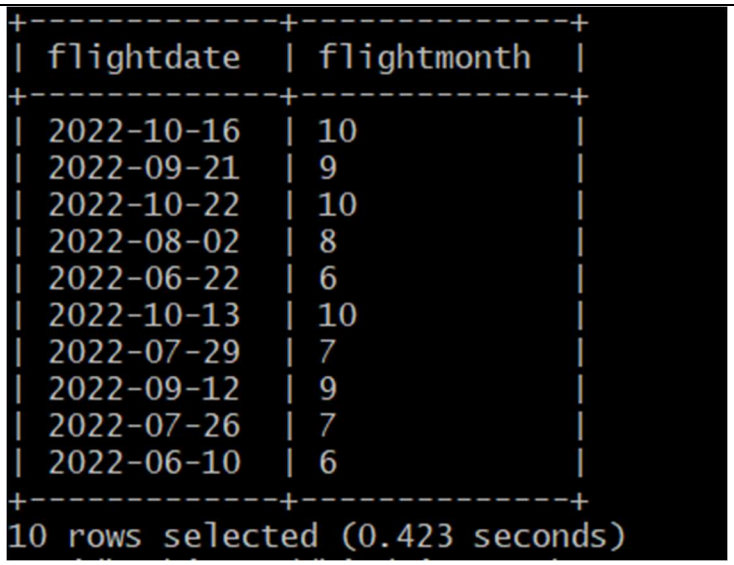
4. Add FlightMonth column by getting the month from FlightDate column

```
ALTER TABLE FlightData2 ADD COLUMNS (FlightMonth int);

INSERT OVERWRITE TABLE FlightData2
SELECT FlightID, SearchDate, FlightDate, StartingAirport,
```

```
DestinationAirport, FareBasisCode string, TravelDuration,
ElapsedDays, IsBasicEconomy, IsRefundable,
IsNonStop, BaseFare, TotalFare, SeatsRemaining,
TotalTravelDistance, SegmentsAirlineName , SegmentsEquipmentDescription,
MONTH(FlightDate) AS FlightMonth FROM FlightData2;

SELECT FlightDate, FlightMonth FROM FlightData2 LIMIT 10;
```



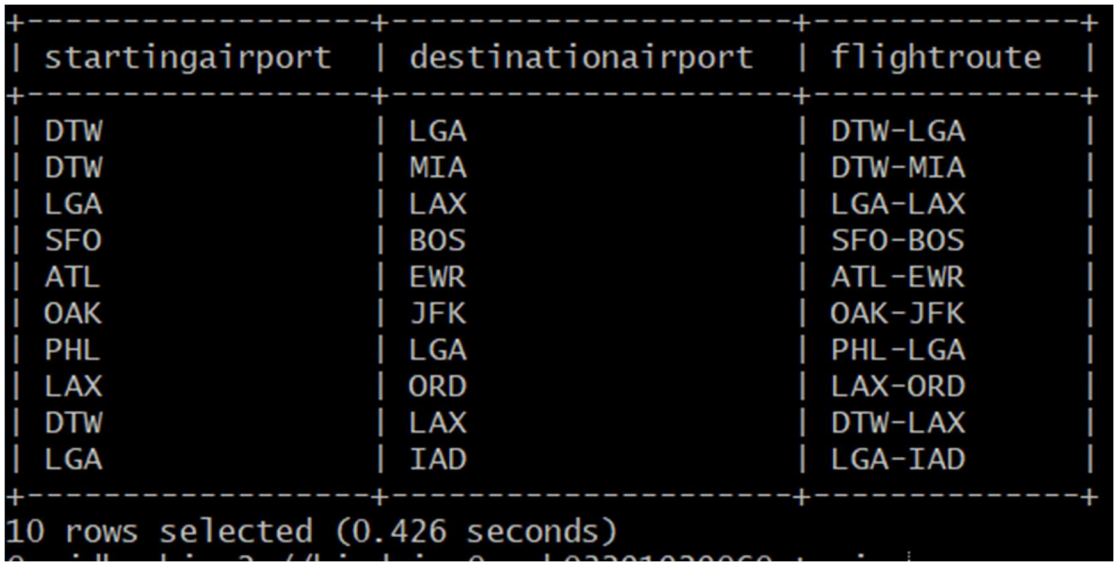
```
+-------------+--------------+
| flightdate  | flightmonth  |
+-------------+--------------+
| 2022-10-16  | 10           |
| 2022-09-21  | 9            |
| 2022-10-22  | 10           |
| 2022-08-02  | 8            |
| 2022-06-22  | 6            |
| 2022-10-13  | 10           |
| 2022-07-29  | 7            |
| 2022-09-12  | 9            |
| 2022-07-26  | 7            |
| 2022-06-10  | 6            |
+-------------+--------------+
10 rows selected (0.423 seconds)
```

5. Add a FlightRoute column that concatenate startingAirport with destinationAirport columns

```
ALTER TABLE FlightData2 ADD COLUMNS (FlightRoute string);

INSERT OVERWRITE TABLE FlightData2
SELECT FlightID, SearchDate, FlightDate, StartingAirport,
DestinationAirport, FareBasisCode string, TravelDuration,
ElapsedDays, IsBasicEconomy, IsRefundable,
IsNonStop, BaseFare, TotalFare, SeatsRemaining,
TotalTravelDistance, SegmentsAirlineName , SegmentsEquipmentDescription, FlightMonth,
CONCAT(StartingAirport, '-', DestinationAirport) AS FlightRoute
FROM FlightData2;

SELECT StartingAirport, DestinationAirport, FlightRoute FROM FlightData2 limit 10;
```



startingairport	destinationairport	flightroute
DTW	LGA	DTW-LGA
DTW	MIA	DTW-MIA
LGA	LAX	LGA-LAX
SFO	BOS	SFO-BOS
ATL	EWR	ATL-EWR
OAK	JFK	OAK-JFK
PHL	LGA	PHL-LGA
LAX	ORD	LAX-ORD
DTW	LAX	DTW-LAX
LGA	IAD	LGA-IAD

10 rows selected (0.426 seconds)

Step 6: Write analysis queries in Hive

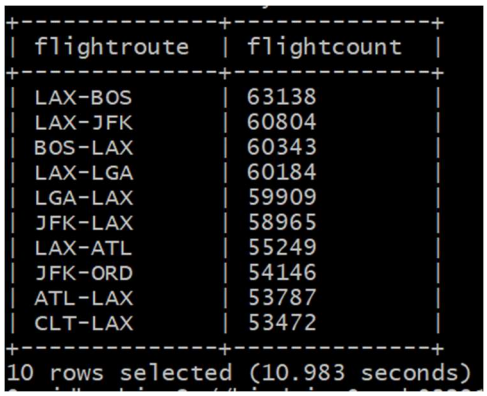
Analysis 1: Top 10 most popular flight routes

The analysis is conducted by following these steps:

- Select the flightRoute column from the FlightData2 table.
- Count the FlightID and assign the alias "FlightCount" to the count column.
- Group the FlightRoute column to group the flight data by unique flight routes.
- Sort the flightCount in descending order.
- Limit the results to the top 10 flight routes.
- Insert the resulting data into the "FlightRoute" subdirectory of "FlightData2".

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/FlightRoute/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT FlightRoute, COUNT(FlightID) AS FlightCount
FROM FlightData2
GROUP BY FlightRoute
ORDER BY FlightCount DESC
LIMIT 10;
```

By following the above steps, we got the top 10 most popular flight routes in the Unites States between April to November 2022 and its count same as below.



flightroute	flightcount
LAX-BOS	63138
LAX-JFK	60804
BOS-LAX	60343
LAX-LGA	60184
LGA-LAX	59909
JFK-LAX	58965
LAX-ATL	55249
JFK-ORD	54146
ATL-LAX	53787
CLT-LAX	53472

10 rows selected (10.983 seconds)

Analysis 2: Least 10 popular flight routes

The analysis is conducted by following these steps:

- Select the flightRoute column from the FlightData2 table.
- Count the FlightID and assign the alias "FlightCount" to the count column.
- Group the FlightRoute column to group the flight data by unique flight routes.
- Sort the flightCount in ascending order.
- Limit the results to the top 10 flight routes.
- Insert the resulting data into the " LeastPopularRoute" subdirectory of "FlightData2".

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/LeastPopularRoute/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT FlightRoute, COUNT(FlightID) AS FlightCount
FROM FlightData2
GROUP BY FlightRoute
ORDER BY FlightCount ASC
LIMIT 10;
```

By following the above steps, we got the least 10 popular flight routes in the United States same as below.

flightroute	flightcount
EWR-LGA	3
IAD-DTW	10188
DTW-IAD	11073
CLT-IAD	12181
JFK-OAK	12810
MIA-OAK	13446
IAD-ATL	13652
ATL-IAD	13794
IAD-CLT	13844
EWR-OAK	14008

Analysis 3: Top 15 Most Expensive Routes:

Below mentioned query is for calculate and write the average flight price for top 15 flight routes in the FlightData2 table to a directory in HDFS.

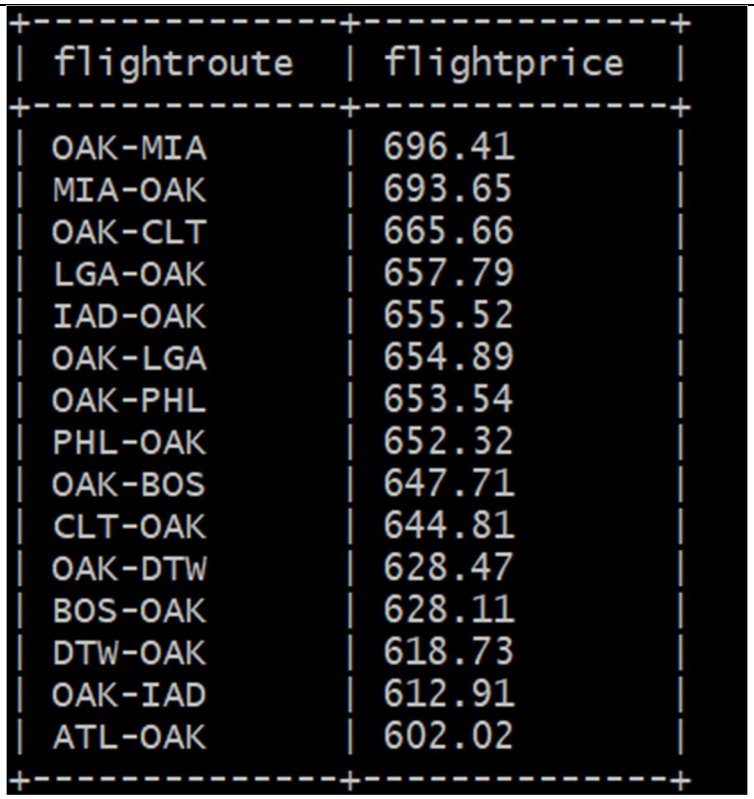
This is the SELECT statement that calculates the average flight price for each flight route in the FlightData2 table.

The ROUND() function is used to round the average price to two decimal places.

The results are the group by flight route sorted in descending order by flight price,limited to top 15 results.

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/FlightPrice/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT FlightRoute, ROUND(AVG(TotalFare),2) AS FlightPrice
FROM FlightData2
```

```
GROUP BY FlightRoute
ORDER BY FlightPrice DESC LIMIT 15;
```



flightroute	flightprice
OAK-MIA	696.41
MIA-OAK	693.65
OAK-CLT	665.66
LGA-OAK	657.79
IAD-OAK	655.52
OAK-LGA	654.89
OAK-PHL	653.54
PHL-OAK	652.32
OAK-BOS	647.71
CLT-OAK	644.81
OAK-DTW	628.47
BOS-OAK	628.11
DTW-OAK	618.73
OAK-IAD	612.91
ATL-OAK	602.02

Analysis 4: Average Airfare and Average Travel Distance over a Month in 2022

To carry out the mentioned analysis, the following steps were performed:

- Calculate the average of the TotalFare and Travel Distance data for each month using the AVG() function.
- Round the calculated averages to two decimal places using the ROUND() function.
- Take the resulting data, including the month, average TotalFare, and average Travel Distance.
- Insert the resulting data into subdirectory "PriceDistanceOverTime" of FlightData2.

By following these steps, the "PriceDistanceOverTime" contains information about the month, average TotalFare, and average Travel Distance for the peroid between March to December 2022.

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/PriceDistanceOverTime/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT FlightMonth, ROUND(AVG(TotalFare),2) AS AverageAirfare,
ROUND(AVG(TotalTravelDistance),2) AS AverageTravelDistance
FROM FlightData2
GROUP BY FlightMonth;
```

Execute the following command to confirm :

```
SELECT FlightMonth, ROUND(AVG(TotalFare),2) AS AverageAirfare,
ROUND(AVG(TotalTravelDistance),2) AS AverageTravelDistance
```

```
FROM FlightData2
GROUP BY FlightMonth;
```

```
+-------------+-----------------+-----------------------+
| flightmonth | averageairfare  | averagetraveldistance |
+-------------+-----------------+-----------------------+
| 8           | 338.14          | 1606.34               |
| 4           | 358.01          | 1593.09               |
| 6           | 410.76          | 1560.08               |
| 7           | 391.91          | 1565.84               |
| 10          | 313.39          | 1657.82               |
| 11          | 276.23          | 1671.29               |
| 5           | 371.85          | 1571.6                |
| 9           | 309.98          | 1659.03               |
+-------------+-----------------+-----------------------+
8 rows selected (13.431 seconds)
```

Analysis 5: Average Airfare by Days per Month

The following steps were performed:

- Calculate the average of the TotalFare and Travel Distance data for each month using the AVG() function.
- These averages were then rounded to two decimal places using the ROUND() function.
- Additionally, the FlightDate column was utilized to determine the weeks and days of the flights.
- The resulting data consisted of the FlightDate, average TotalFare, and average TravelDistance.
- Insert the resulting data into subdirectory "Days" of FlightData2.

As a result, the "Days" subdirectory now contains information about the month, average TotalFare, and average Travel Distance for the period spanning from April to November 2022.

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/Days/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT FlightDate, ROUND(AVG(TotalFare),2) AS AverageAirfare,
ROUND(AVG(TotalTravelDistance),2) AS AverageTravelDistance
FROM FlightData2
GROUP BY FlightDate;
```

Execute the following command to confirm :

```
SELECT FlightDate, ROUND(AVG(TotalFare),2) AS AverageAirfare,
ROUND(AVG(TotalTravelDistance),2) AS AverageTravelDistance
FROM FlightData2
GROUP BY FlightDate;
```

flightdate	averageairfare	averagetraveldistance
2022-04-18	405.31	1625.24
2022-04-19	332.93	1616.81
2022-04-23	414.35	1587.06
2022-04-25	384.87	1550.95
2022-05-03	275.04	1593.98
2022-05-04	285.02	1568.19
2022-05-05	332.62	1569.91
2022-05-08	381.07	1547.24
2022-05-09	346.42	1566.21
2022-05-11	310.03	1569.18
2022-05-13	387.34	1551.53
2022-05-15	491.2	1565.12
2022-05-16	399.3	1563.66
2022-05-17	328.78	1579.96
2022-05-19	403.81	1583.46
2022-05-20	405.33	1575.39
2022-05-23	392.06	1566.3
2022-05-26	425.86	1544.28
2022-05-31	386.29	1562.8
2022-06-02	357.81	1554.55
2022-06-03	372.51	1537.79
2022-06-07	339.85	1530.92
2022-06-08	363.65	1531.91
2022-06-10	432.37	1555.07
2022-06-11	422.79	1598.98
2022-06-13	423.21	1550.7
2022-06-14	369.01	1560.69
2022-06-15	383.21	1553.51
2022-06-16	417.63	1565.1
2022-06-18	430.12	1616.63
2022-06-19	471.9	1566.58
2022-06-20	448.72	1564.32
2022-06-21	376.95	1555.83
2022-06-22	379.27	1555.73
2022-06-23	428.93	1560.93
2022-06-27	423.33	1562.79
2022-06-30	448.49	1564.57
2022-07-02	434.04	1619.51
2022-07-05	371.22	1553.83
2022-07-07	359.36	1568.99
2022-07-09	414.03	1615.15
2022-07-11	380.18	1549.49
2022-07-12	337.36	1559.44
2022-07-13	352.34	1561.49
2022-07-15	405.93	1541.28
2022-07-16	414.58	1617.04
2022-07-17	462.39	1549.7
2022-07-18	393.78	1551.7
2022-07-19	345.4	1560.09
2022-07-20	357.45	1565.57
2022-07-27	351.56	1565.88
2022-07-31	452.76	1551.54
2022-08-01	382.42	1573.03
2022-08-05	378.79	1563.74
2022-08-08	370.88	1589.69
2022-08-11	354.6	1588.95
2022-08-12	366.51	1585.15
2022-08-14	424.57	1588.72
2022-08-15	357.75	1599.47
2022-08-16	307.17	1614.89
2022-08-19	347.36	1600.17
2022-08-20	335.7	1624.11
2022-08-22	332.48	1608.58
2022-08-23	281.21	1620.12
2022-08-25	318.45	1627.54
2022-08-26	325.61	1620.7
2022-08-28	364.08	1612.98
2022-08-29	304.33	1646.02
2022-09-03	299.13	1687.2
2022-09-06	306.82	1629.51
2022-09-10	282.62	1712.09
2022-09-11	359.25	1646.5
2022-09-14	270.72	1662.92
2022-09-15	325.98	1654.78
2022-09-16	330.94	1644.07
2022-09-17	293.66	1695.51
2022-09-19	332.64	1631.61
2022-09-20	265.81	1675.1
2022-09-21	275.35	1647.43
2022-09-22	339.76	1638.75
2022-09-23	352.5	1661.57
2022-09-24	299.78	1711.3
2022-09-25	372.51	1658.7
2022-09-30	328.84	1687.31
2022-10-02	370.03	1653.83
2022-10-04	255.41	1687.5
2022-10-05	277.13	1649.24
2022-10-06	346.66	1608.97
2022-10-07	377.1	1638.09
2022-10-09	388.5	1563.83
2022-10-10	384.64	1611.82
2022-10-14	349.47	1654.3
2022-10-17	333.02	1664.12
2022-10-19	265.21	1657.87
2022-10-23	406.8	1587.11
2022-10-24	328.01	1644.34
2022-10-26	240.3	1684.8

2022-09-20	265.81	1675.1
2022-09-21	275.35	1647.43
2022-09-22	339.76	1638.75
2022-09-23	352.5	1661.57
2022-09-24	299.78	1711.3
2022-09-25	372.51	1658.7
2022-09-30	328.84	1687.31
2022-10-02	370.03	1653.83
2022-10-04	255.41	1687.5
2022-10-05	277.13	1649.24
2022-10-06	346.66	1608.97
2022-10-07	377.1	1638.09
2022-10-09	388.5	1563.83
2022-10-10	384.64	1611.82
2022-10-14	349.47	1654.3
2022-10-17	333.02	1664.12
2022-10-19	265.21	1657.87
2022-10-23	406.8	1587.11
2022-10-24	328.01	1644.34
2022-10-26	240.3	1684.8
2022-10-28	303.13	1693.61
2022-10-31	284.61	1713.07
2022-11-02	237.33	1687.56

flightdate	averageairfare	averagetraveldistance
2022-11-04	310.05	1678.25
2022-11-07	299.04	1664.56
2022-11-09	234.42	1659.34
2022-11-11	303.26	1671.68
2022-11-12	252.19	1704.3
2022-11-13	375.34	1602.74
2022-11-14	293.36	1683.27
2022-11-15	221.0	1688.78
2022-11-16	238.53	1669.19
2022-11-17	317.29	1627.99
2022-11-18	383.39	1584.47

2022-04-17	441.42	1590.33
2022-04-20	334.24	1623.18
2022-04-21	371.01	1590.75
2022-04-22	397.79	1589.72
2022-04-24	529.18	1548.26
2022-04-26	307.5	1570.86
2022-04-27	310.35	1597.23
2022-04-28	351.84	1589.7
2022-04-29	359.5	1614.8
2022-04-30	315.71	1620.13
2022-05-01	438.12	1557.95
2022-05-02	330.11	1575.82
2022-05-06	344.1	1583.98
2022-05-07	303.09	1623.11
2022-05-10	284.19	1582.25
2022-05-12	393.4	1568.91
2022-05-14	363.99	1603.39
2022-05-18	347.65	1574.03
2022-05-21	360.99	1623.54
2022-05-22	481.55	1563.97
2022-05-24	328.91	1584.66
2022-05-25	361.98	1575.65
2022-05-27	430.1	1527.99
2022-05-28	359.29	1618.46
2022-05-29	354.01	1555.83
2022-05-30	445.83	1513.01
2022-06-01	335.61	1536.92
2022-06-04	372.49	1602.29
2022-06-05	462.42	1552.4
2022-06-06	392.12	1553.38
2022-06-09	414.45	1551.94
2022-06-12	495.96	1544.53
2022-06-17	438.53	1568.79
2022-06-24	445.28	1544.42
2022-06-25	450.1	1615.16
2022-06-26	501.63	1574.61
2022-06-28	367.78	1548.63
2022-06-29	381.78	1543.21
2022-07-01	474.28	1543.92
2022-07-03	388.26	1595.31
2022-07-04	347.4	1560.78
2022-07-06	349.58	1568.87
2022-07-08	370.61	1547.22
2022-07-10	457.5	1564.33
2022-07-14	393.23	1545.68
2022-07-21	392.44	1545.79
2022-07-22	403.64	1550.32
2022-07-23	414.75	1608.22
2022-07-24	455.79	1545.47
2022-07-25	394.71	1556.81
2022-07-26	345.31	1566.9
2022-07-28	384.01	1572.02
2022-07-29	395.46	1560.28
2022-07-30	414.44	1614.21
2022-08-02	333.48	1564.9
2022-08-03	338.49	1578.58
2022-08-04	367.32	1570.37
2022-08-06	393.83	1637.52
2022-08-07	436.91	1587.35
2022-08-09	320.14	1591.17
2022-08-10	326.79	1597.2
2022-08-13	382.28	1642.96
2022-08-17	306.45	1603.07
2022-08-18	340.1	1605.49
2022-08-21	392.51	1592.84
2022-08-24	284.52	1620.53
2022-08-27	303.28	1653.58
2022-08-30	262.26	1631.85
2022-08-31	286.26	1623.9
2022-09-01	339.19	1599.34
2022-09-02	350.92	1619.45
2022-09-04	310.29	1651.38
2022-09-05	378.45	1581.82
2022-09-07	266.29	1654.32
2022-09-08	284.31	1653.4
2022-09-09	301.3	1669.59
2022-09-12	311.82	1654.18
2022-09-13	258.39	1675.94
2022-09-18	400.1	1619.42
2022-09-26	311.27	1670.9
2022-09-27	262.93	1686.87
2022-09-28	270.66	1687.45
2022-09-29	311.22	1693.68
2022-10-01	295.0	1716.03
2022-10-03	307.09	1682.41
2022-10-08	312.09	1672.65
2022-10-11	293.29	1644.59
2022-10-12	286.33	1645.56

flightdate	averageairfare	averagetraveldistance
2022-10-13	338.19	1652.72
2022-10-15	301.09	1673.53
2022-10-16	416.74	1590.91
2022-10-18	250.79	1666.16
2022-10-20	327.96	1645.34
2022-10-21	332.22	1658.22
2022-10-22	282.01	1689.31
2022-10-25	237.56	1668.23
2022-10-27	286.88	1669.04

2022-05-06	344.1	1583.98
2022-05-07	303.09	1623.11
2022-05-10	284.19	1582.25
2022-05-12	393.4	1568.91
2022-05-14	363.99	1603.39
2022-05-18	347.65	1574.03
2022-05-21	360.99	1623.54
2022-05-22	481.55	1563.97
2022-05-24	328.91	1584.66
2022-05-25	361.98	1575.65
2022-05-27	430.1	1527.99
2022-05-28	359.29	1618.46
2022-05-29	354.01	1555.83
2022-05-30	445.83	1513.01
2022-06-01	335.61	1536.92
2022-06-04	372.49	1602.29
2022-06-05	462.42	1552.4
2022-06-06	392.12	1553.38
2022-06-09	414.45	1551.94
2022-06-12	495.96	1544.53
2022-06-17	438.53	1568.79
2022-06-24	445.28	1544.42
2022-06-25	450.1	1615.16
2022-06-26	501.63	1574.61
2022-06-28	367.78	1548.63
2022-06-29	381.78	1543.21
2022-07-01	474.28	1543.92
2022-07-03	388.26	1595.31
2022-07-04	347.4	1560.78
2022-07-06	349.58	1568.87
2022-07-08	370.61	1547.22
2022-07-10	457.5	1564.33
2022-07-14	393.23	1545.68
2022-07-21	392.44	1545.79
2022-07-22	403.64	1550.32
2022-07-23	414.75	1608.22
2022-07-24	455.79	1545.47
2022-07-25	394.71	1556.81
2022-07-26	345.31	1566.9
2022-07-28	384.01	1572.02
2022-07-29	395.46	1560.28
2022-07-30	414.44	1614.21
2022-08-02	333.48	1564.9
2022-08-03	338.49	1578.58
2022-08-04	367.32	1570.37
2022-08-06	393.83	1637.52
2022-08-07	436.91	1587.35
2022-08-09	320.14	1591.17
2022-08-10	326.79	1597.2
2022-08-13	382.28	1642.96
2022-08-17	306.45	1603.07
2022-08-18	340.1	1605.49
2022-08-21	392.51	1592.84
2022-08-24	284.52	1620.53
2022-08-27	303.28	1653.58
2022-08-30	262.26	1631.85
2022-08-31	286.26	1623.9
2022-09-01	339.19	1599.34
2022-09-02	350.92	1619.45
2022-09-04	310.29	1651.38
2022-09-05	378.45	1581.82
2022-09-07	266.29	1654.32
2022-09-08	284.31	1653.4
2022-09-09	301.3	1669.59
2022-09-12	311.82	1654.18
2022-09-13	258.39	1675.94
2022-09-18	400.1	1619.42
2022-09-26	311.27	1670.9
2022-09-27	262.93	1686.87
2022-09-28	270.66	1687.45
2022-09-29	311.22	1693.68
2022-10-01	295.0	1716.03
2022-10-03	307.09	1682.41
2022-10-08	312.09	1672.65
2022-10-11	293.29	1644.59
2022-10-12	286.33	1645.56

flightdate	averageairfare	averagetraveldistance
2022-10-13	338.19	1652.72
2022-10-15	301.09	1673.53
2022-10-16	416.74	1590.91
2022-10-18	250.79	1666.16
2022-10-20	327.96	1645.34
2022-10-21	332.22	1658.22
2022-10-22	282.01	1689.31
2022-10-25	237.56	1668.23
2022-10-27	286.88	1669.04
2022-10-29	243.05	1718.24
2022-10-30	365.93	1684.06
2022-11-01	237.12	1690.23
2022-11-03	292.59	1680.41
2022-11-05	250.78	1712.76
2022-11-06	356.48	1633.73
2022-11-08	222.0	1666.73
2022-11-10	308.91	1640.23
2022-11-19	377.91	1614.22

218 rows selected (17.859 seconds)

Analysis 6: Most popular destinations over a month in 2022

For this analysis, we joined two tables 1) FlightData2 and 2) Airport.

- The FlightData2 table was used to obtain information such as the Destination airport, FlightDate, and FlightID. The FlightDate column was formatted using the date_format() function to represent the month as 'MM-01-yyyy' format. This formatting was done to calculate the total number of flights per month, ensuring that the month portion remains consistent with a value of 01 rather than representing different dates within the month.
- The Airport table was utilized to retrieve additional data, specifically the Latitude and Longitude values associated with each Destination airport.
- To combine the information from both tables, a JOIN operation was performed based on the common field DestinationAirport in the FlightData2 table and the IATA column in the Airport table. These columns contain a 3-character airport code that facilitates the matching of records.
- The resulting data contains the following fields: DestinationAirport, formatted_date, Latitude, Longitude, and FlightCount.
- Insert the resulting data into subdirectory "Destination" of FlightData2.

As a result, the " Destination" subdirectory now contains information about DestinationAirport, formatted_date, Latitude, Longitude, and FlightCount from April to November 2022.

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/Destination/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT DestinationAirport, date_format(FlightDate,'MM-01-yyyy') AS formatted_date,Latitude,
Longitude, COUNT(FlightID) AS FlightCount
FROM FlightData2 F
JOIN Airport A
ON F.DestinationAirport = A.IATA
GROUP BY DestinationAirport,date_format(FlightDate,'MM-01-yyyy'), Latitude, Longitude;
```

To see the result:

```
SELECT DestinationAirport, date_format(FlightDate,'MM-01-yyyy') AS formatted_date,Latitude,
Longitude, COUNT(FlightID) AS FlightCount
FROM FlightData2 F
JOIN Airport A
ON F.DestinationAirport = A.IATA
GROUP BY DestinationAirport,date_format(FlightDate,'MM-01-yyyy'), Latitude, Longitude;
```

destinationairport	formatted_date	latitude	longitude	flightcount
ATL	07-01-2022	33.6367	-84.428101	71991
ATL	09-01-2022	33.6367	-84.428101	95253
BOS	05-01-2022	42.36429977	-71.00520325	47187
BOS	11-01-2022	42.36429977	-71.00520325	17078
DEN	10-01-2022	39.86169815	-104.6729965	53910
DFW	06-01-2022	32.896801	-97.038002	67548
DFW	11-01-2022	32.896801	-97.038002	15644
DTW	10-01-2022	42.21239853	-83.35340118	49283
EWR	10-01-2022	40.69250107	-74.16870117	45257
JFK	11-01-2022	40.63980103	-73.77890015	13539
ORD	09-01-2022	41.9786	-87.9048	110728
PHL	07-01-2022	39.87189865	-75.2410965	66857
DEN	09-01-2022	39.86169815	-104.6729965	88198
DTW	07-01-2022	42.21239853	-83.35340118	62527
DTW	08-01-2022	42.21239853	-83.35340118	78988
IAD	06-01-2022	38.94449997	-77.45580292	50838
LAX	06-01-2022	33.94250107	-118.4079971	99473
LAX	09-01-2022	33.94250107	-118.4079971	145866
LGA	10-01-2022	40.77719879	-73.87259674	72834
OAK	06-01-2022	37.721298	-122.221001	37366
ORD	07-01-2022	41.9786	-87.9048	80350
SFO	06-01-2022	37.61899948	-122.375	76918
DEN	05-01-2022	39.86169815	-104.6729965	40034
DFW	08-01-2022	32.896801	-97.038002	112516
DTW	09-01-2022	42.21239853	-83.35340118	82245
LGA	07-01-2022	40.77719879	-73.87259674	96853
OAK	04-01-2022	37.721298	-122.221001	2882
OAK	11-01-2022	37.721298	-122.221001	11642
PHL	08-01-2022	39.87189865	-75.2410965	91432
PHL	11-01-2022	39.87189865	-75.2410965	12593
SFO	10-01-2022	37.61899948	-122.375	70440
ATL	08-01-2022	33.6367	-84.428101	93638
BOS	06-01-2022	42.36429977	-71.00520325	76186
DEN	11-01-2022	39.86169815	-104.6729965	12728
ORD	08-01-2022	41.9786	-87.9048	109741
ATL	10-01-2022	33.6367	-84.428101	60705
BOS	08-01-2022	42.36429977	-71.00520325	116800
BOS	09-01-2022	42.36429977	-71.00520325	114389
BOS	10-01-2022	42.36429977	-71.00520325	75810
CLT	07-01-2022	35.2140007	-80.94309998	75434
CLT	10-01-2022	35.2140007	-80.94309998	72829
CLT	11-01-2022	35.2140007	-80.94309998	15711
DEN	08-01-2022	39.86169815	-104.6729965	94902
DFW	07-01-2022	32.896801	-97.038002	77572
DFW	10-01-2022	32.896801	-97.038002	72630
DTW	11-01-2022	42.21239853	-83.35340118	11516
IAD	04-01-2022	38.94449997	-77.45580292	3518
JFK	08-01-2022	40.63980103	-73.77890015	93881
JFK	10-01-2022	40.63980103	-73.77890015	59116
LAX	04-01-2022	33.94250107	-118.4079971	8211
LAX	05-01-2022	33.94250107	-118.4079971	61966
MIA	04-01-2022	25.79319954	-80.29060364	6634
MIA	07-01-2022	25.79319954	-80.29060364	73082
OAK	10-01-2022	37.721298	-122.221001	47293
CLT	05-01-2022	35.2140007	-80.94309998	47144
CLT	09-01-2022	35.2140007	-80.94309998	109579
DEN	07-01-2022	39.86169815	-104.6729965	70902
DFW	05-01-2022	32.896801	-97.038002	44603
JFK	04-01-2022	40.63980103	-73.77890015	4335
LAX	10-01-2022	33.94250107	-118.4079971	90062
LAX	11-01-2022	33.94250107	-118.4079971	20264
LGA	04-01-2022	40.77719879	-73.87259674	6452
MIA	06-01-2022	25.79319954	-80.29060364	68290
MIA	11-01-2022	25.79319954	-80.29060364	13816
ORD	04-01-2022	41.9786	-87.9048	5759
ORD	05-01-2022	41.9786	-87.9048	45344
SFO	05-01-2022	37.61899948	-122.375	48895
SFO	09-01-2022	37.61899948	-122.375	112273
ATL	05-01-2022	33.6367	-84.428101	44436
ATL	06-01-2022	33.6367	-84.428101	65124
EWR	06-01-2022	40.69250107	-74.16870117	54056
EWR	08-01-2022	40.69250107	-74.16870117	73737
EWR	09-01-2022	40.69250107	-74.16870117	70677
IAD	05-01-2022	38.94449997	-77.45580292	28576
IAD	07-01-2022	38.94449997	-77.45580292	55505

IAD	07-01-2022	38.94449997	-77.45580292	55505
LAX	07-01-2022	33.94250107	-118.4079971	114778
MIA	09-01-2022	25.79319954	-80.29060364	95679
PHL	04-01-2022	39.87189865	-75.2410965	4156
SFO	07-01-2022	37.61899948	-122.375	87888
SFO	08-01-2022	37.61899948	-122.375	116333
CLT	08-01-2022	35.2140007	-80.94309998	104851
DEN	06-01-2022	39.86169815	-104.6729965	61438
DTW	05-01-2022	42.21239853	-83.35340118	34351
MIA	05-01-2022	25.79319954	-80.29060364	47015
MIA	08-01-2022	25.79319954	-80.29060364	93514
OAK	07-01-2022	37.721298	-122.221001	43490
ORD	10-01-2022	41.9786	-87.9048	69216
ORD	11-01-2022	41.9786	-87.9048	15720
PHL	06-01-2022	39.87189865	-75.2410965	60302
CLT	04-01-2022	35.2140007	-80.94309998	5563
EWR	05-01-2022	40.69250107	-74.16870117	34299
IAD	10-01-2022	38.94449997	-77.45580292	41800
JFK	06-01-2022	40.63980103	-73.77890015	57334
MIA	10-01-2022	25.79319954	-80.29060364	60762
OAK	05-01-2022	37.721298	-122.221001	22825
PHL	05-01-2022	39.87189865	-75.2410965	38376
BOS	04-01-2022	42.36429977	-71.00520325	5113
CLT	06-01-2022	35.2140007	-80.94309998	69230
DEN	04-01-2022	39.86169815	-104.6729965	5843
DFW	09-01-2022	32.896801	-97.038002	114076

destinationairport	formatted_date	latitude	longitude	flightcount
DTW	04-01-2022	42.21239853	-83.35340118	4610
EWR	04-01-2022	40.69250107	-74.16870117	4143
EWR	11-01-2022	40.69250107	-74.16870117	10482
IAD	08-01-2022	38.94449997	-77.45580292	66424
IAD	09-01-2022	38.94449997	-77.45580292	61443
LAX	08-01-2022	33.94250107	-118.4079971	154699
LGA	05-01-2022	40.77719879	-73.87259674	53223
LGA	09-01-2022	40.77719879	-73.87259674	117876
ORD	06-01-2022	41.9786	-87.9048	68904
PHL	09-01-2022	39.87189865	-75.2410965	92493
SFO	11-01-2022	37.61899948	-122.375	16120
ATL	04-01-2022	33.6367	-84.428101	5584
ATL	11-01-2022	33.6367	-84.428101	13183
EWR	07-01-2022	40.69250107	-74.16870117	58883
JFK	05-01-2022	40.63980103	-73.77890015	36718
JFK	09-01-2022	40.63980103	-73.77890015	91169
LGA	08-01-2022	40.77719879	-73.87259674	123334
LGA	11-01-2022	40.77719879	-73.87259674	15952
OAK	09-01-2022	37.721298	-122.221001	76588
SFO	04-01-2022	37.61899948	-122.375	5931
BOS	07-01-2022	42.36429977	-71.00520325	89571
DFW	04-01-2022	32.896801	-97.038002	6283
DTW	06-01-2022	42.21239853	-83.35340118	54456
IAD	11-01-2022	38.94449997	-77.45580292	9780
JFK	07-01-2022	40.63980103	-73.77890015	68896
LGA	06-01-2022	40.77719879	-73.87259674	83205
OAK	08-01-2022	37.721298	-122.221001	69076
PHL	10-01-2022	39.87189865	-75.2410965	60666

128 rows selected (42.332 seconds)

Analysis 7: Distribution of Basic Economy tickets

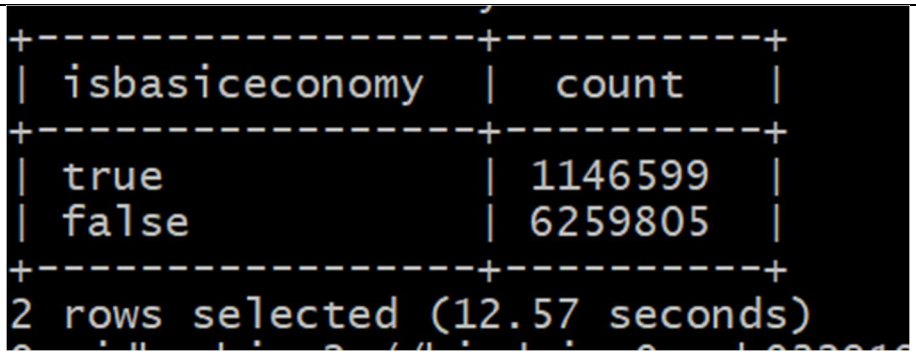
The following steps performed:

- In FlightData2,isBasicEconomy field contains Boolean value such as ‘Y’ and ‘N’. Bu using COUNT() function, it will calculate the total number of purchased tickets and saved as a count.
- The query groups the data by the isBasicEconomy column, which separates the records based on whether they are classified as basic economy or not.
- The resulting data contains isBasicEconomy and count.

- Insert the resulting data into subdirectory "BasicEconomy" of FlightData2.

As a result, the "BasicEconomy " subdirectory now contains information about isBasicEconomy and count.

```
INSERT OVERWRITE DIRECTORY '/user/amach3/FlightData2/BasicEconomy/'
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
SELECT isBasicEconomy, COUNT(*) as count
FROM FlightData2
GROUP BY isBasicEconomy;
```

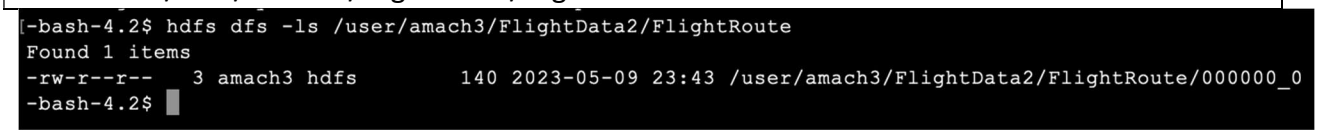


Step 7: Export query results to HDFS, secure copy text files to local computer

1. Top 10 most popular flight routes (FlightRoute.txt)

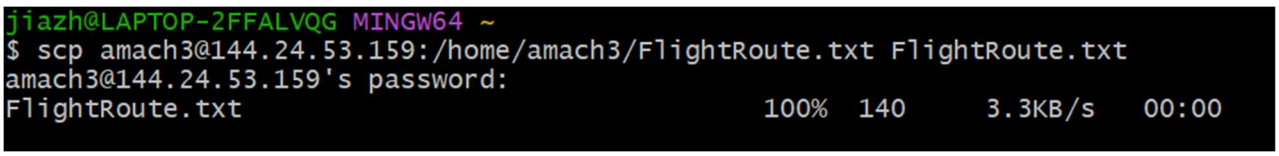
Open another terminal with git bash, minty, or putty, which is to connect the HADOOP CLUSTER to download the output file 000000_0 at the HDFS path "/user/amach3/FlightData2/FlightRoute":

```
hdfs dfs -ls /user/amach3/FlightData2/FlightRoute
```



Download the file to the local file systems:

```
hdfs dfs -get /user/amach3/FlightData2/FlightRoute/000000_0 FlightRoute.txt
scp amach3@144.24.53.159:/home/amach3/FlightRoute.txt FlightRoute.txt
```



2. Least 10 popular flight routes (LeastPopularRoute.txt)

Download the output file 000000_0 at the HDFS path "/user/amach3/FlightData2/LeastPopularRoute"

```
hdfs dfs -ls /user/amach3/FlightData2/LeastPopularRoute
```

```
-bash-4.2$ hdfs dfs -ls /user/amach3/FlightData2/LeastPopularRoute
Found 1 items
-rw-r--r--   3 amach3 hdfs        136 2023-05-02 02:06 /user/amach3/FlightData2/LeastPopularRoute/000000_0
-bash-4.2$
```

Download the file to the local file systems:

```
hdfs dfs -get /user/amach3/FlightData2/LeastPopularRoute/000000_0 LeastPopularRoute.txt
scp amach3@144.24.53.159:/home/amach3/LeastPopularRoute.txt LeastPopularRoute.txt
```

```
jiazh@LAPTOP-2FFALVQG MINGW64 ~
$ scp amach3@144.24.53.159:/home/amach3/LeastPopularRoute.txt LeastPopularRoute.txt
amach3@144.24.53.159's password:
LeastPopularRoute.txt                                         100%  136     2.7KB/s   00:00
```

3. Top 15 most expensive routes (FlightPrice.txt)

Download the output file "000000_0" to "FlightPrice.txt" using the following hdfs command:

```
hdfs dfs -get /user/amach3/FlightData2/FlightPrice/000000_0 FlightPrice.txt
scp amach3@144.24.53.159:/home/amach3/FlightPrice.txt FlightPrice.txt
```

```
jiazh@LAPTOP-2FFALVQG MINGW64 ~
$ scp amach3@144.24.53.159:/home/amach3/FlightPrice.txt FlightPrice.txt
amach3@144.24.53.159's password:
FlightPrice.txt                                100%   75     1.8KB/s   00:00
```

4. Airfare and Travel Distance over time (PriceDistanceOverTime.txt)

- This command lists all the files inside the subdirectory "PriceDistanceOverTime" in HDFS.

```
hdfs dfs -ls /user/amach3/FlightData2/PriceDistanceOverTime
```

```
-bash-4.2$ hdfs dfs -ls /user/amach3/FlightData2/PriceDistanceOverTime
Found 4 items
-rw-r--r--   3 amach3 hdfs         17 2023-05-09 23:46 /user/amach3/FlightData2/PriceDistanceOverTime/000000_0
-rw-r--r--   3 amach3 hdfs         59 2023-05-09 23:46 /user/amach3/FlightData2/PriceDistanceOverTime/000001_0
-rw-r--r--   3 amach3 hdfs         36 2023-05-09 23:46 /user/amach3/FlightData2/PriceDistanceOverTime/000002_0
-rw-r--r--   3 amach3 hdfs         34 2023-05-09 23:46 /user/amach3/FlightData2/PriceDistanceOverTime/000003_0
```

- Download the output files using -get command to the local file systems and rename them as PriceDistanceOverTime.txt.
- For these steps, we need to concatenate the files into the local Linux file system for downloading in step.

```
hdfs dfs -get /user/amach3/FlightData2/PriceDistanceOverTime/000000_0
hdfs dfs -get /user/amach3/FlightData2/PriceDistanceOverTime/000001_0
hdfs dfs -get /user/amach3/FlightData2/PriceDistanceOverTime/000002_0
hdfs dfs -get /user/amach3/FlightData2/PriceDistanceOverTime/000003_0

cat 000000_0 000002_0 000003_0 000001_0 > PriceDistanceOverTime.txt
```

- Open another terminal with git bash in order to import the output file using your lab computer (or your PC/Laptop) - you have to download the file to your lab computer (or your PC/Laptop). For example, your output file at the oracle cloud server is located at /home/amach3/PriceDistanceOverTime.txt. And, remotely copied to the file "PriceDistanceOverTime.txt".
- You will be prompted for your credentials. Provide your password.

```
scp amach3@144.24.53.159:/home/amach3/ PriceDistanceOverTime.txt
PriceDistanceOverTime.txt
```

```
jiazh@LAPTOP-2FFALVQG MINGW64 ~
$ scp amach3@144.24.53.159:/home/amach3/PriceDistanceOverTime.txt PriceDistanceOve
rTime.txt
amach3@144.24.53.159's password:
PriceDistanceOverTime.txt                     100%   17     0.3KB/s   00:00
```

5. Average Airfare by Days per Month (Days.txt)

- This command lists all the files inside the subdirectory "Days" in HDFS.

```
hdfs dfs -ls /user/amach3/FlightData2/Days/
```

```
-bash-4.2$ hdfs dfs -ls /user/amach3/FlightData2/Days/
Found 2 items
-rw-r--r--   3 amach3 hdfs       2858 2023-05-13 18:44 /user/amach3/FlightData2/Days/000000_0
-rw-r--r--   3 amach3 hdfs       2743 2023-05-13 18:44 /user/amach3/FlightData2/Days/000001_0
```

- Download the output files to the local file systems and rename them as Days.txt.
- For these steps, we need to concatenate the csv files into the local Linux file system for downloading in step.

```
hdfs dfs -get /user/amach3/FlightData2/Days/000000_0
hdfs dfs -get /user/amach3/FlightData2/Days/000001_0

cat 000000_0 000001_0 > Days.txt
```

```
-bash-4.2$ hdfs dfs -get /user/amach3/FlightData2/Days/000000_0
cat 000000_0 000001_0 > Days.txt
-bash-4.2$ hdfs dfs -get /user/amach3/FlightData2/Days/000001_0
-bash-4.2$ cat 000000_0 000001_0 > Days.txt
```

- Open another terminal with git bash in order to import the output file using your lab computer (or your PC/Laptop) - you have to download the file to your lab computer (or your PC/Laptop). For example, your output file at the oracle cloud server is located at /home/amach3/ Days.txt. And, remotely copied to the file “Days.txt”.
- You will be prompted for your credentials. Provide your password.

```
scp amach3@144.24.53.159:/home/amach3/Days.txt Days.txt
```

```
jiazh@LAPTOP-2FFALVQG MINGW64 ~
$ scp amach3@144.24.53.159:/home/amach3/Days.txt Days.txt
amach3@144.24.53.159's password:
Days.txt                                      100% 5601   116.5KB/s   00:00
```

6. Most popular destinations over a month in 2022 (Destination.txt)

- This command lists all the files inside the subdirectory "Destination" in HDFS.

```
hdfs dfs -ls /user/amach3/FlightData2/Destination/
```

```
-bash-4.2$ hdfs dfs -ls /user/amach3/FlightData2/Destination
Found 12 items
-rw-r--r--   3 amach3 hdfs        524 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000000_0
-rw-r--r--   3 amach3 hdfs        446 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000001_0
-rw-r--r--   3 amach3 hdfs        398 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000002_0
-rw-r--r--   3 amach3 hdfs        171 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000003_0
-rw-r--r--   3 amach3 hdfs        853 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000004_0
-rw-r--r--   3 amach3 hdfs        613 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000005_0
-rw-r--r--   3 amach3 hdfs        532 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000006_0
-rw-r--r--   3 amach3 hdfs        394 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000007_0
-rw-r--r--   3 amach3 hdfs        316 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000008_0
-rw-r--r--   3 amach3 hdfs        671 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000009_0
-rw-r--r--   3 amach3 hdfs        394 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000010_0
-rw-r--r--   3 amach3 hdfs        358 2023-05-05 03:16 /user/amach3/FlightData2/Destination/000011_0
-bash-4.2$
```

- Download the output files to the local file systems and rename them as Destination.txt.
- For these steps, we need to concatenate the csv files into the local Linux file system for downloading in step.

```
hdfs dfs -get /user/amach3/FlightData2/Destination/000000_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000001_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000002_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000003_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000004_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000005_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000006_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000007_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000008_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000009_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000010_0
hdfs dfs -get /user/amach3/FlightData2/Destination/000011_0

cat 000000_0 000002_0 000003_0 000001_0 000004_0 000005_0 000006_0 000007_0
000008_0 000009_0 000010_0 000011_0 > Destination.txt
```

```
-bash-4.2$ ls
000000_0  000002_0  000004_0  000006_0  000008_0  000010_0  Destination.txt
000001_0  000003_0  000005_0  000007_0  000009_0  000011_0
-bash-4.2$
```

- Open another terminal with git bash in order to import the output file using your lab computer (or your PC/Laptop) - you have to download the file to your lab computer (or your PC/Laptop). For example, your output file at the oracle cloud server is located at /home/amach3/ Destination.txt. And, remotely copied to the file “Destination.txt”.
- You will be prompted for your credentials. Provide your password.

```
scp amach3@144.24.53.159:/home/amach3/Destination.txt Destination.txt
```

```
jiazh@LAPTOP-2FFALVQG MINGW64 ~
$ scp amach3@144.24.53.159:/home/amach3/Destination.txt
Destination.txt
amach3@144.24.53.159's password:
Destination.txt       100% 5670   114.2KB/s   00:00
```

7. Distribution of basic economy tickets (BasicEconomy.txt)

- This command lists all the files inside the subdirectory "BasicEconomy" in HDFS.

```
hdfs dfs -ls /user/amach3/FlightData2/BasicEconomy/
```

```
-bash-4.2$ hdfs dfs -ls /user/amach3/FlightData2/BasicEconomy/

Found 2 items
-rw-r--r--   3 amach3 hdfs         13 2023-05-09 23:50 /user/amach3/FlightData2/BasicEconomy/000000_0
-rw-r--r--   3 amach3 hdfs         21 2023-05-09 23:50 /user/amach3/FlightData2/BasicEconomy/000001_0
-bash-4.2$
```

- Download the output files to the local file systems and rename them as BasicEconomy.txt.
- For these steps, we need to concatenate the csv files into the local Linux file system for downloading in step.

```
hdfs dfs -get /user/amach3/FlightData2/BasicEconomy/000000_0
hdfs dfs -get /user/amach3/FlightData2/BasicEconomy/000001_0
cat 000000_0 000001_0 > BasicEconomy.txt
```

- Open another terminal with git bash in order to import the output file using your lab computer (or your PC/Laptop) - you have to download the file to your lab computer (or your PC/Laptop). For example, your output file at the oracle cloud server is located at /home/amach3/ BasicEconomy.txt. And, remotely copied to the file “BasicEconomy.txt”.
- You will be prompted for your credentials. Provide your password.

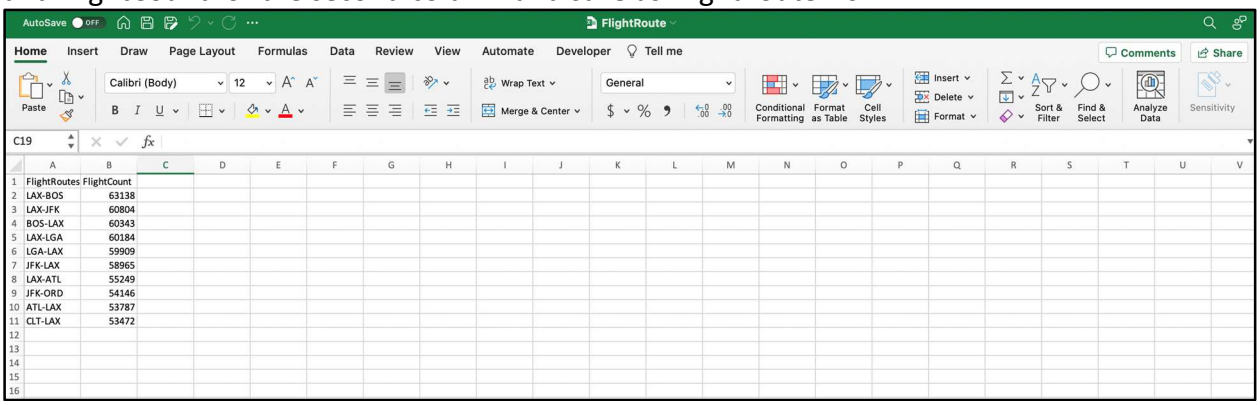
```
scp amach3@144.24.53.159:/home/amach3/BasicEconomy.txt BasicEconomy.txt
```

```
jiazh@LAPTOP-2FFALVQG MINGW64 ~
$ scp amach3@144.24.53.159:/home/amach3/BasicEconomy.txt BasicEconomy.txt
amach3@144.24.53.159's password:
BasicEconomy.txt                              100%   13     0.3KB/s   00:00
```

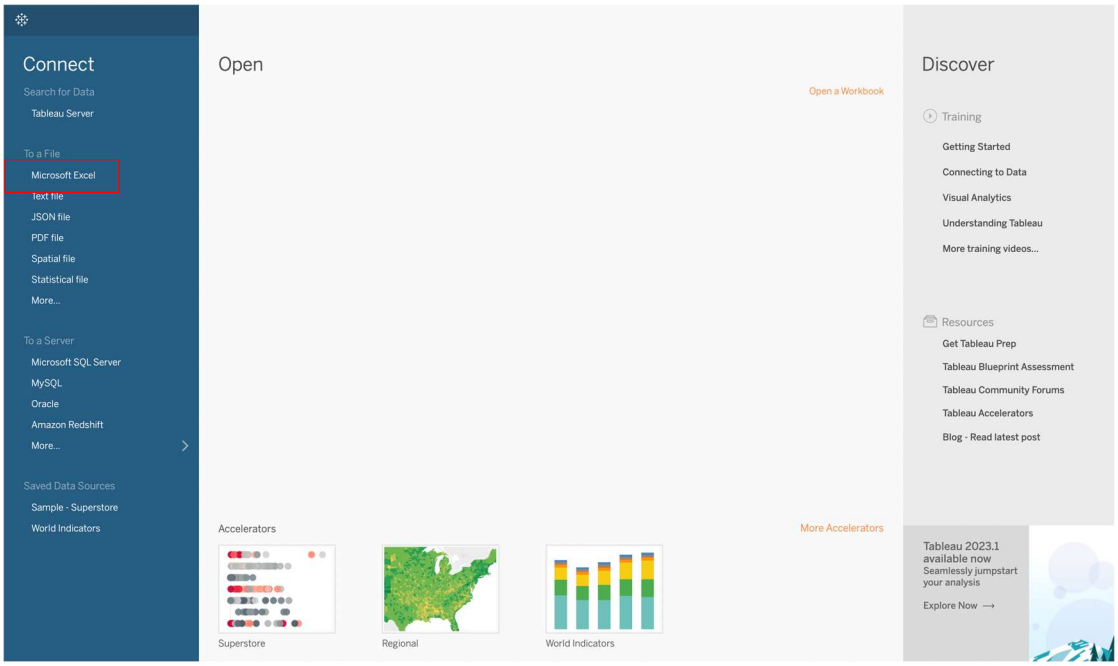
Step 8: Create visualizations.

1. Top 10 most popular flight routes (Bar chart)

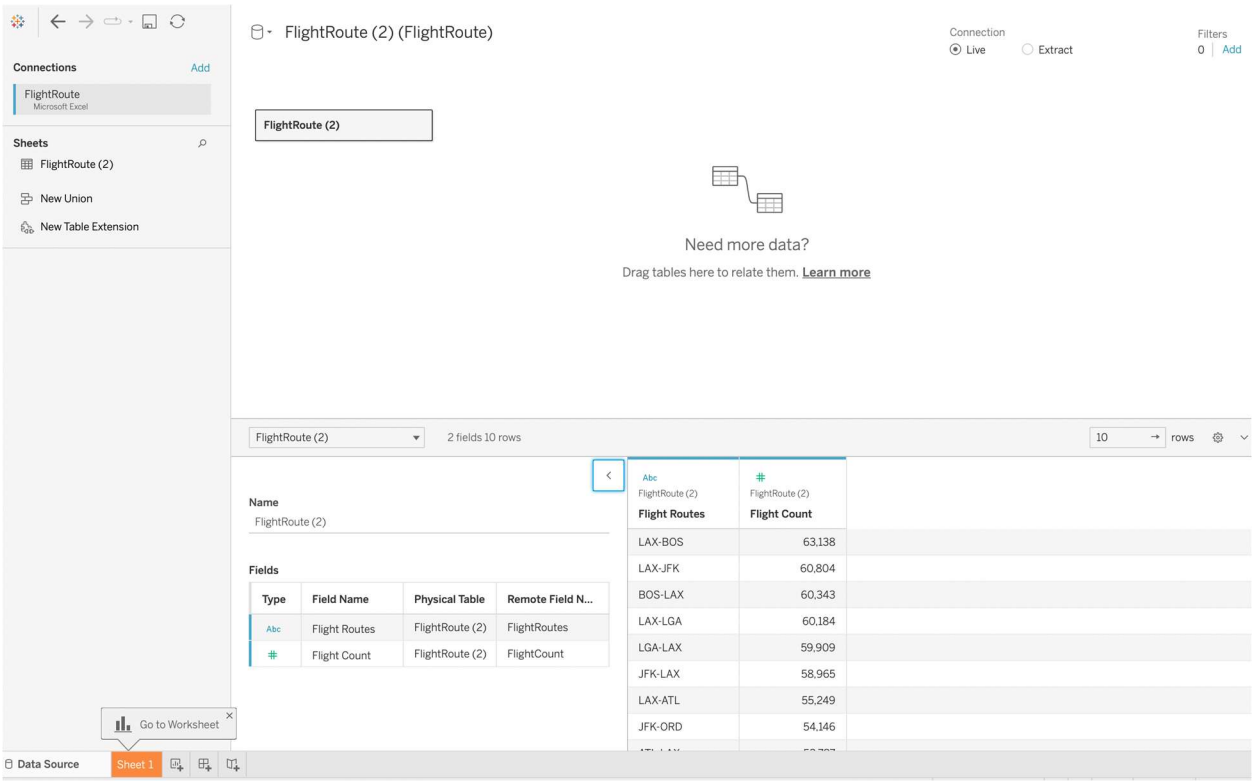
Open the FlightRoute.txt in the Microsoft excel select the **delimited → Next → select delimiters to tab and comma → Next → Finish**. Then insert the column name FlightRoutes in first column and FlightCount for the second column and save as FlightRoute.xlsx



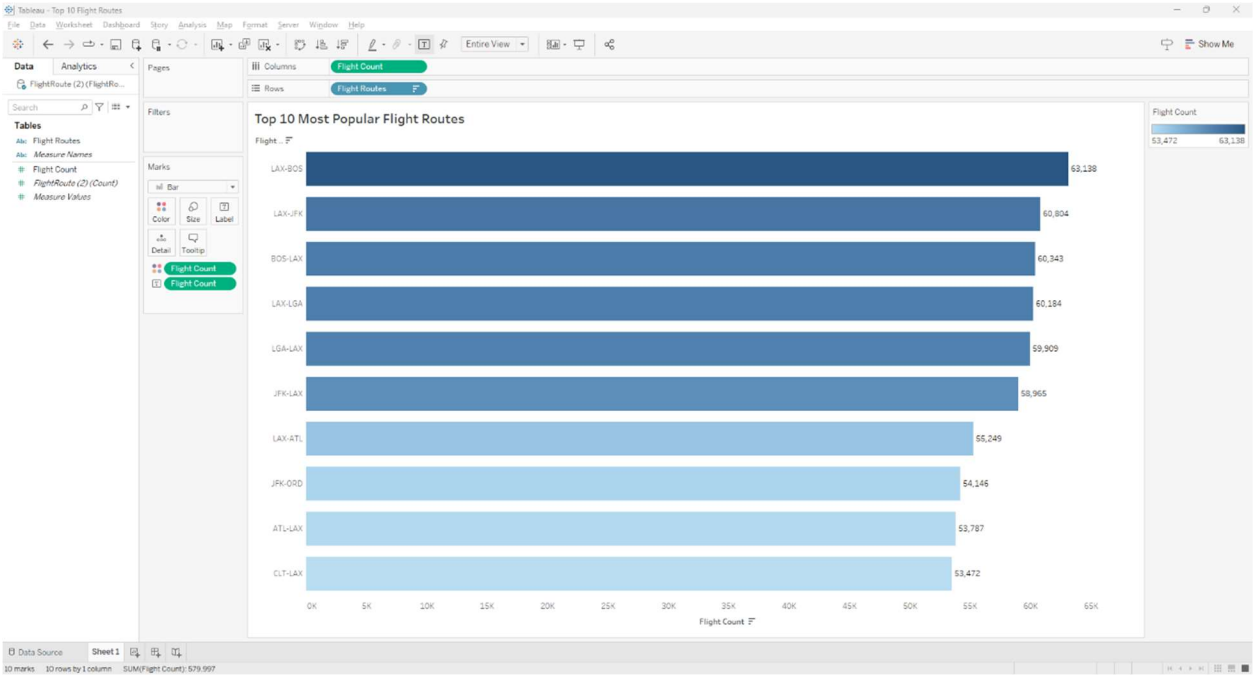
Open your Tableau to connect to your server. You need to select Microsoft Excel File to open the file FlightRoute.xlsx file.



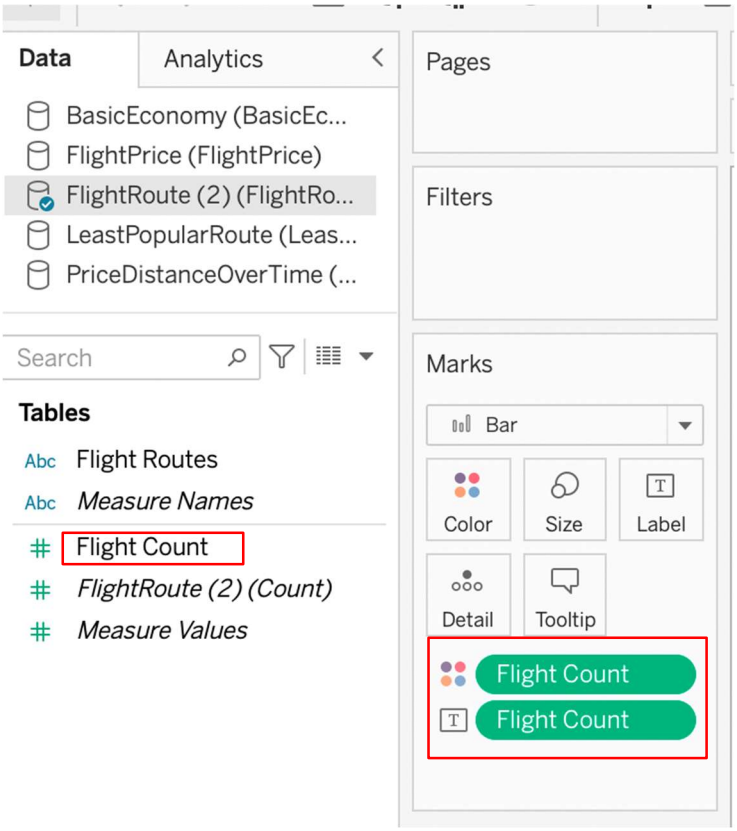
You will see the following data at **Data Source** tab.



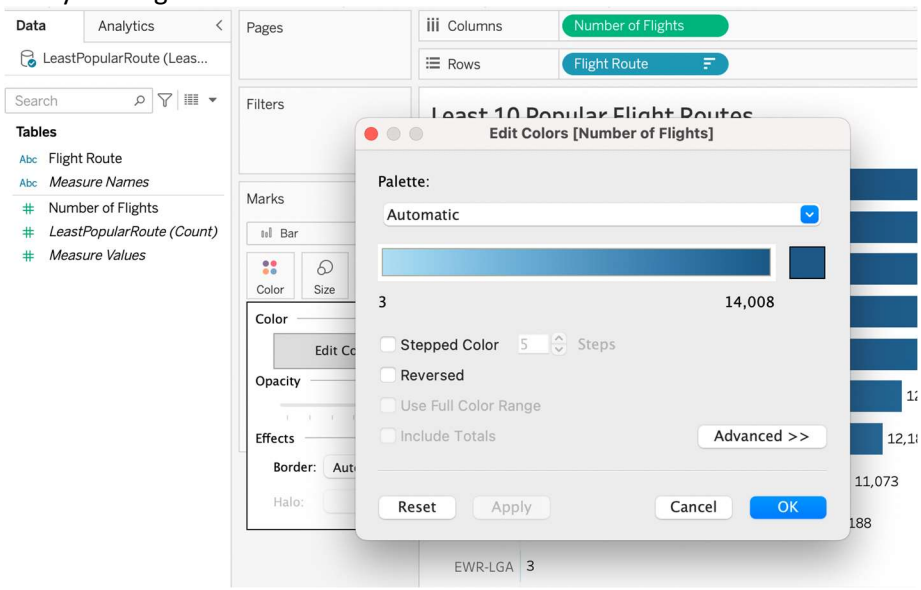
Select **Sheet 1** next to Data Source, which will present the following frame. Then drag the FlightCount to **Columns**, and FlightRoutes to **Rows**, which shows Top 10 most popular flight routes in the USA.

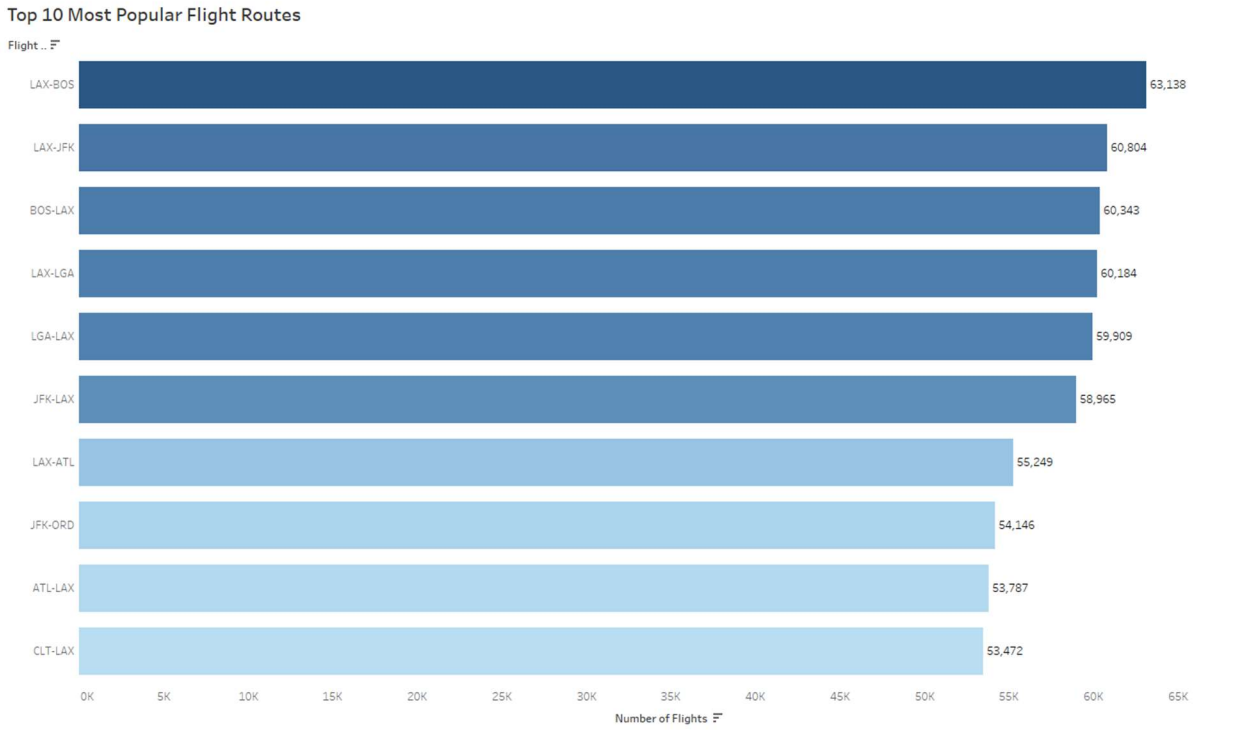


Drop the FlightCount to **Color and Label.**



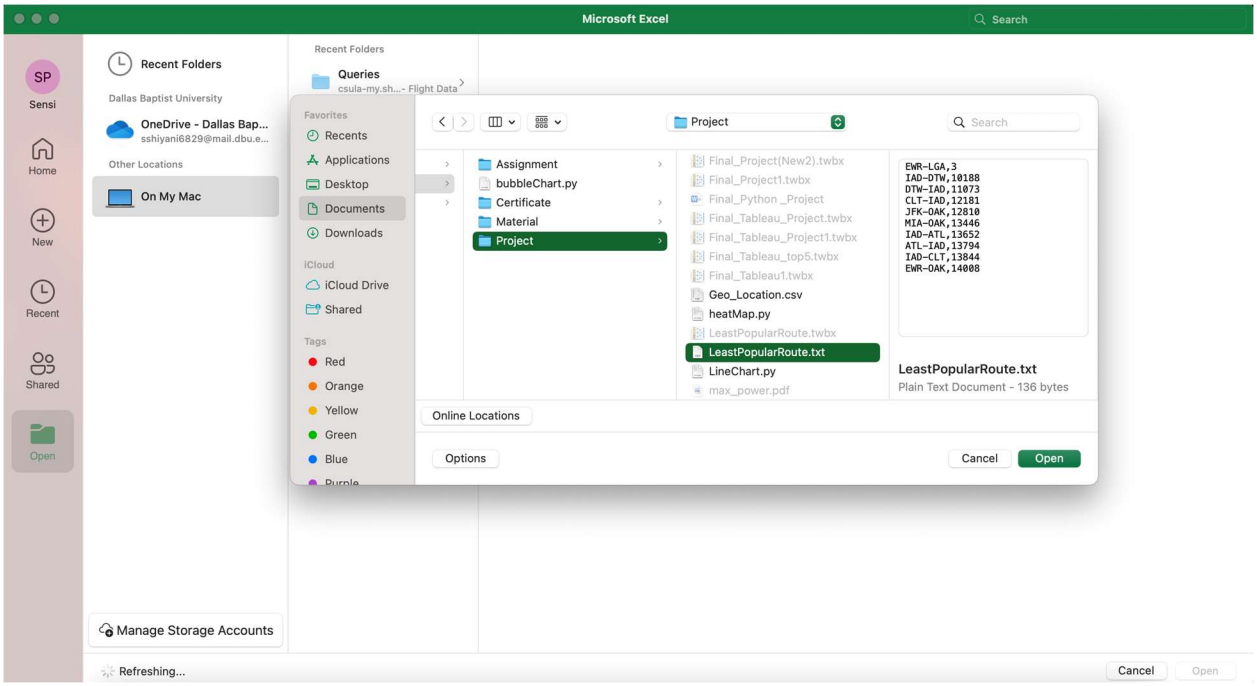
Change the color by clicking on **Color** same as below.

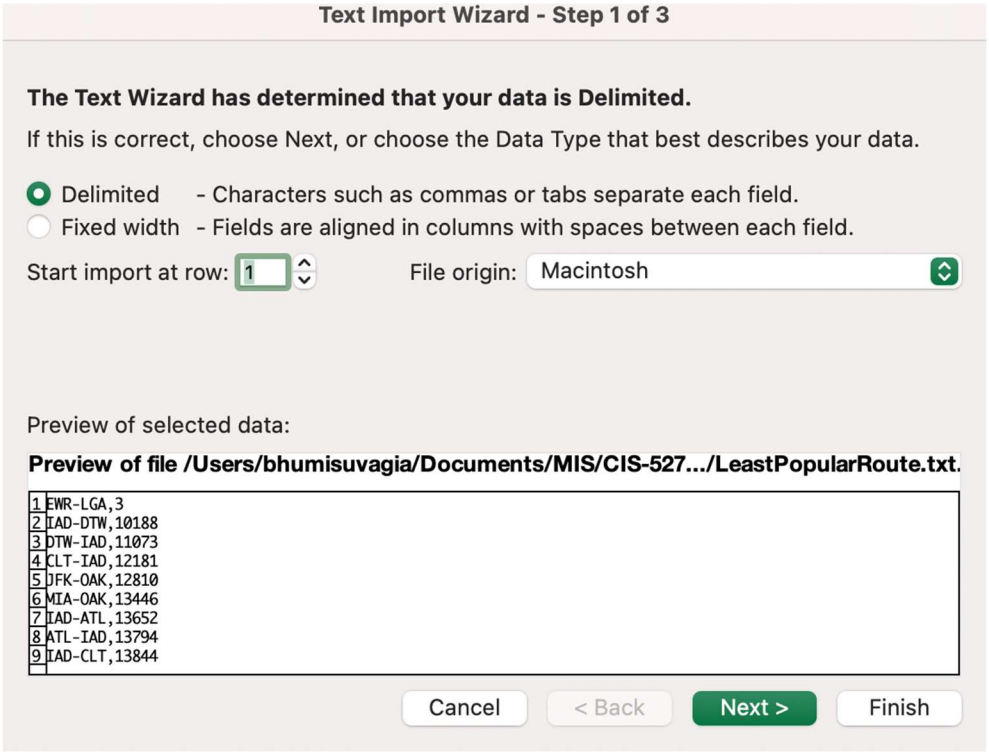


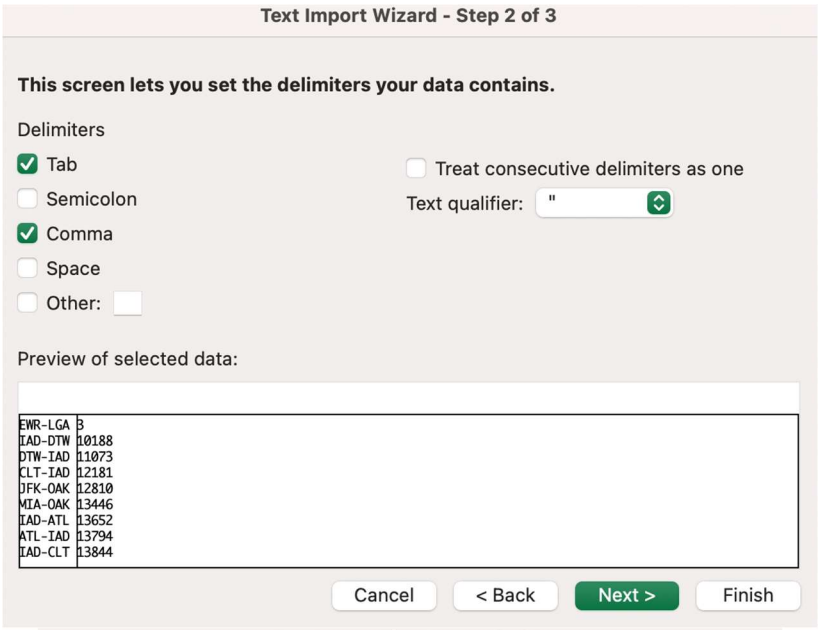


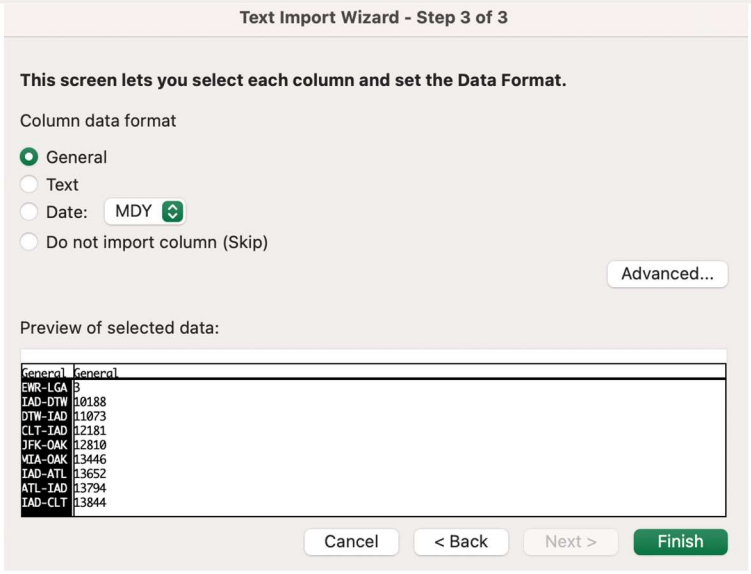
2. Least 10 popular flight routes (bar chart)

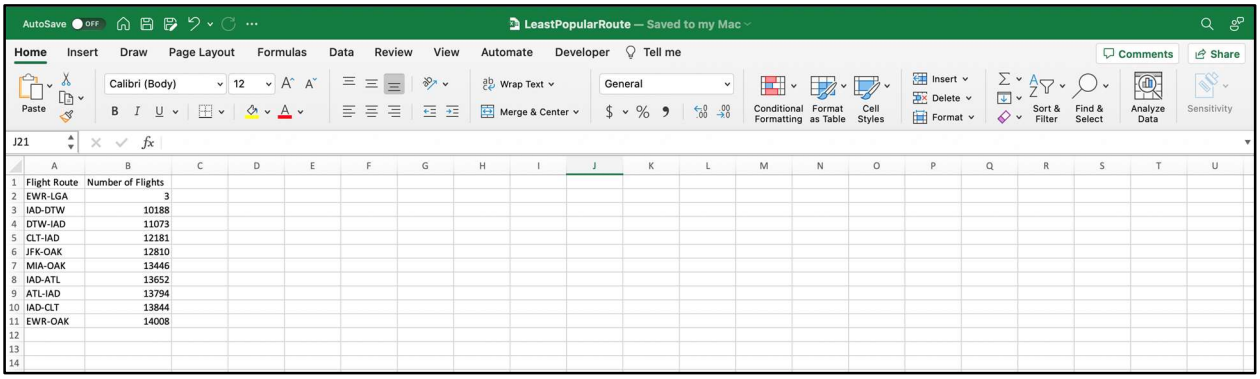
Open the LeastPopularRoute.txt in the Microsoft excel select the **delimited → Next → select delimiters to tab and comma → Next → Finish**. Then insert the column name Flight Route in first column and Number of Flights for the second column and save as LeastPopularRoute.xlsx


Microsoft Excel
Search
Recent Folders
Queries
csula-my.sh...- Flight Data
Dallas Baptist University
OneDrive - Dallas Bap...
sshiyani6829@mail.dbu.e...
Other Locations
On My Mac
Manage Storage Accounts
Refreshing...
Cancel
Open
Favorites
Recents
Applications
Desktop
Documents
Downloads
iCloud
iCloud Drive
Shared
Tags
Red
Orange
Yellow
Green
Blue
Project
Assignment
bubbleChart.py
Certificate
Material
Project
Final_Project(New2).twbx
Final_Project1.twbx
Final_Python _Project
Final_Tableau_Project.twbx
Final_Tableau_Project1.twbx
Final_Tableau_top5.twbx
Final_Tableau1.twbx
Geo_Location.csv
heatMap.py
LeastPopularRoute.twbx
LeastPopularRoute.txt
LineChart.py
max_power.pdf
EWR-LGA,3
IAD-DTW,10188
DTW-IAD,11073
CLT-IAD,12181
JFK-OAK,12810
MIA-OAK,13446
IAD-ATL,13652
ATL-IAD,13794
IAD-CLT,13844
EWR-OAK,14008
LeastPopularRoute.txt
Plain Text Document - 136 bytes
Online Locations
Options
Cancel
Open


Text Import Wizard - Step 1 of 3
The Text Wizard has determined that your data is Delimited.
If this is correct, choose Next, or choose the Data Type that best describes your data.
Delimited - Characters such as commas or tabs separate each field.
Fixed width - Fields are aligned in columns with spaces between each field.
Start import at row: 1
File origin: Macintosh
Preview of selected data:
Preview of file /Users/bhumisuvagia/Documents/MIS/CIS-527.../LeastPopularRoute.txt.
1 EWR-LGA,3
2 IAD-DTW,10188
3 DTW-IAD,11073
4 CLT-IAD,12181
5 JFK-OAK,12810
6 MIA-OAK,13446
7 IAD-ATL,13652
8 ATL-IAD,13794
9 IAD-CLT,13844
Cancel
< Back
Next >
Finish

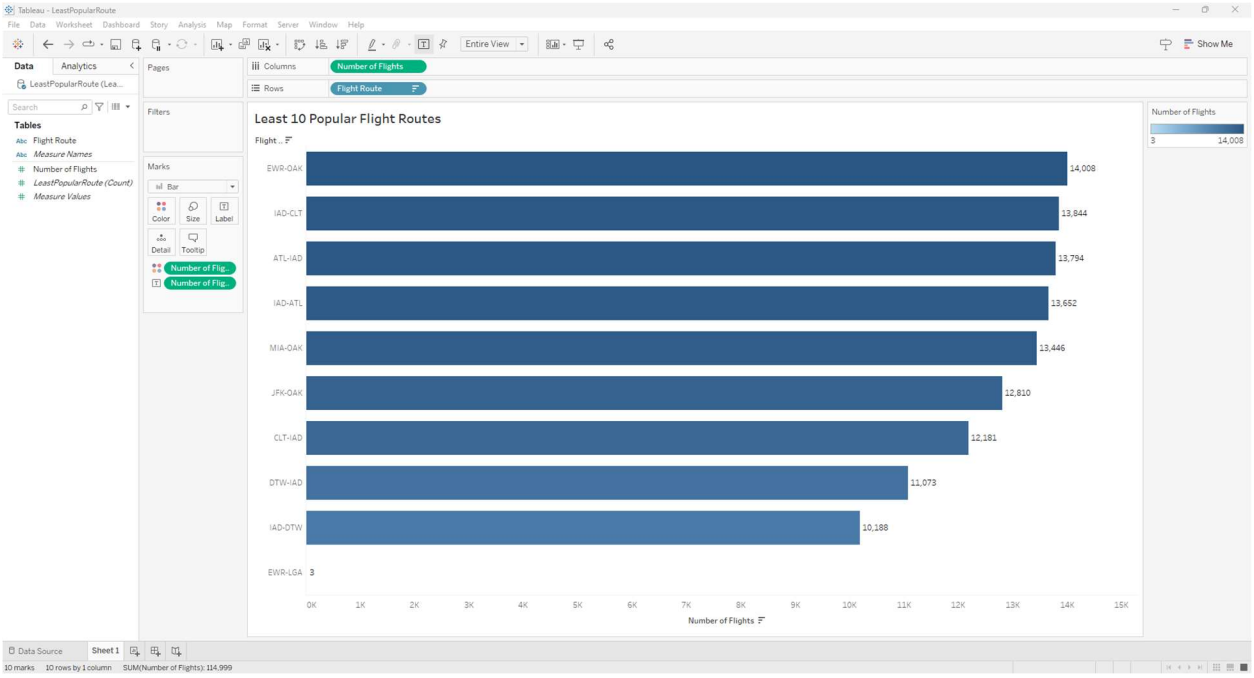




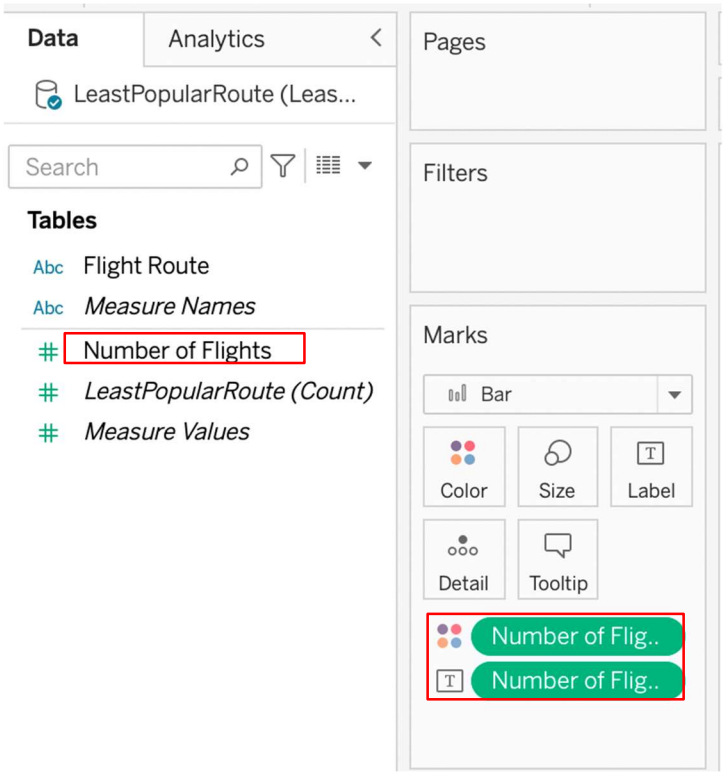


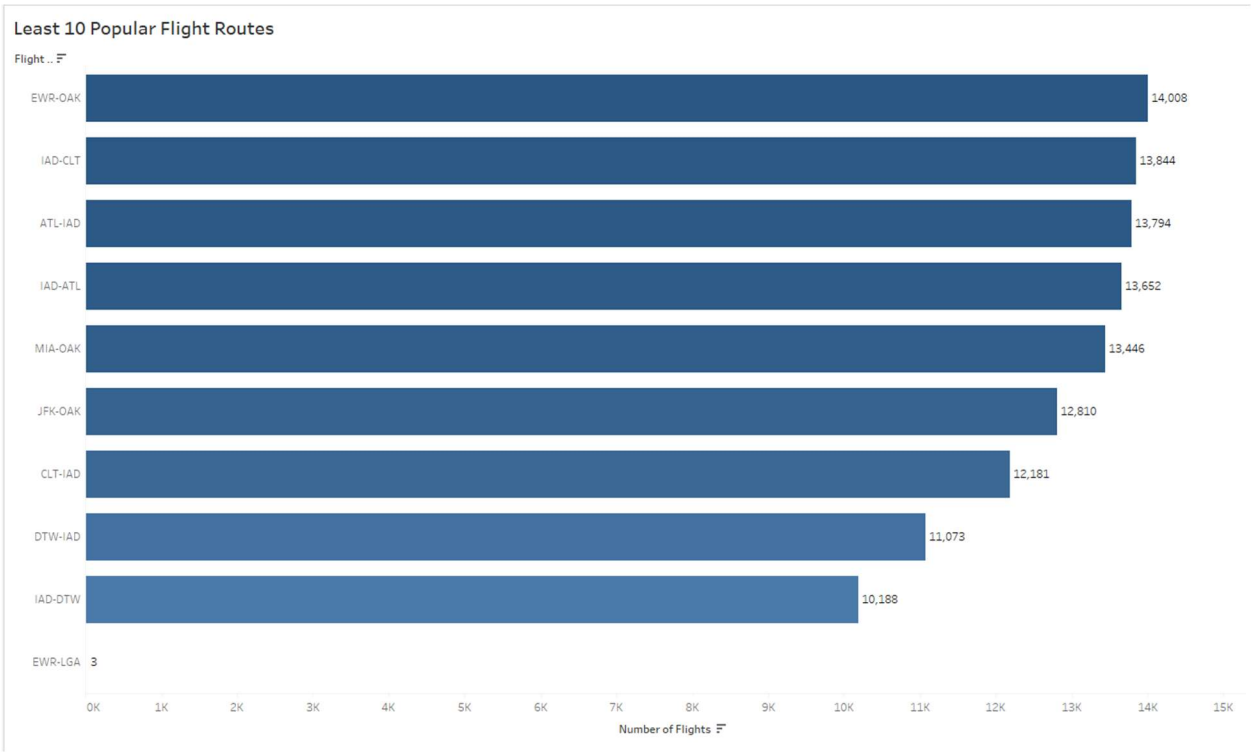
Open your Tableau to connect your server. You need to select Microsoft Excel File to open the file LeastPopularRoute.xlsx file same as we did for the top 10 most popular flight routes.

Select **Sheet 1** next to Data Source, which will present the following frame. Then drag the Number of Flights to **Columns**, and Flight Route to **Rows**, which shows Least 10 most popular flight routes in the USA.



Drop the Number of Flights to **Color** and **Label,** and set the **Standard to Entire View**



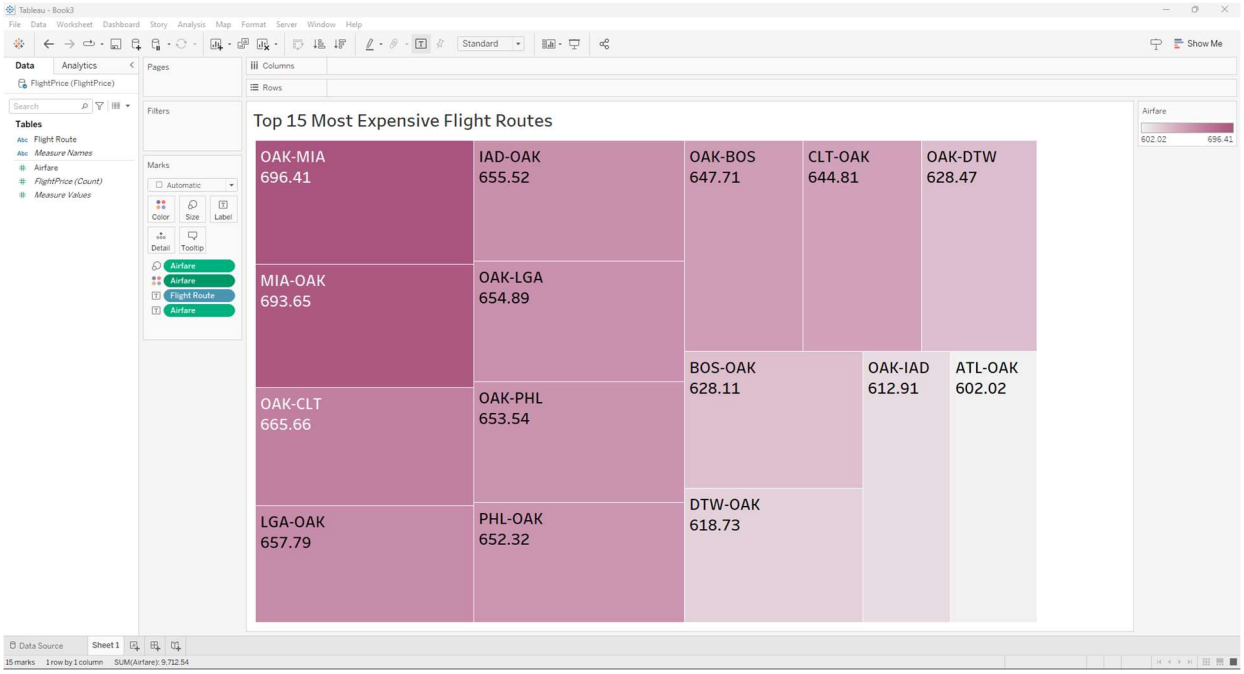


3. Top 15 most expensive routes (Heat map)

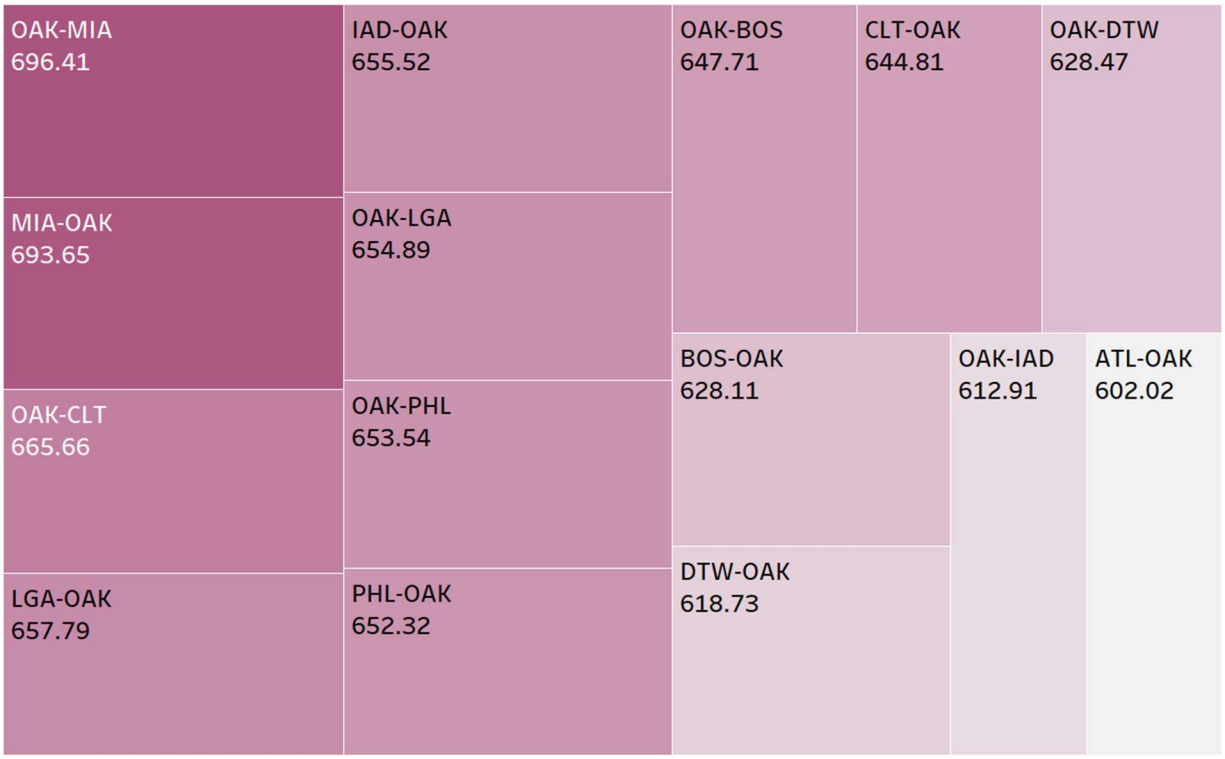
To show this Analysis we have chosen to create a heat map in Tableau.

First step, we decide to show Airfare in each rectangle, we choose Airfare to **color** and then other two measure are in **Text** and in **Size**. The reason to add those two measures in text is to show flight routes and flight fares clearly in the given rectangle

We have chosen the color shade dark pink to highlight the most expensive flight route.

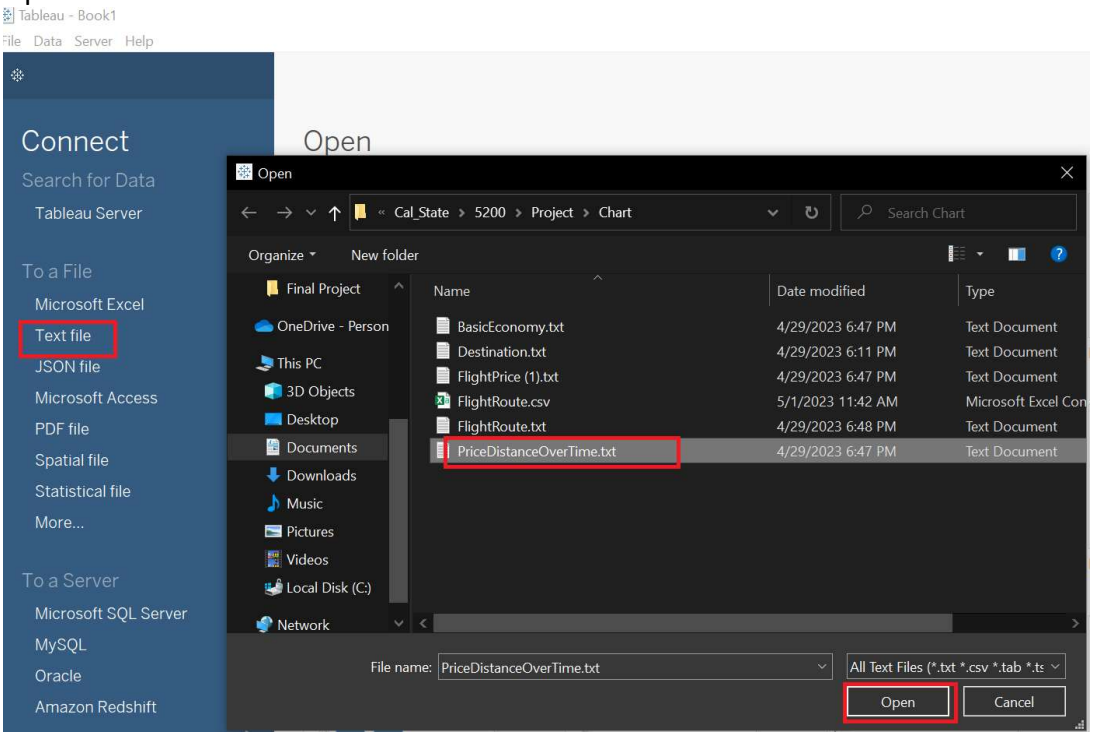


Top 15 Most Expensive Flight Routes



4. Airfare and Travel Distance over time (dual axis chart in Tableau)

Open → Text File:



Rename F1→ Flight Month, F2→ Average Airfare, and F3 → Average Travel Distance:

PriceDistanceOverTime.txt 3 fields 8 rows

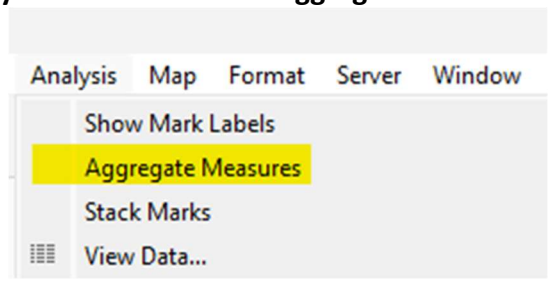
Name
PriceDistanceOverTime.txt

Fields

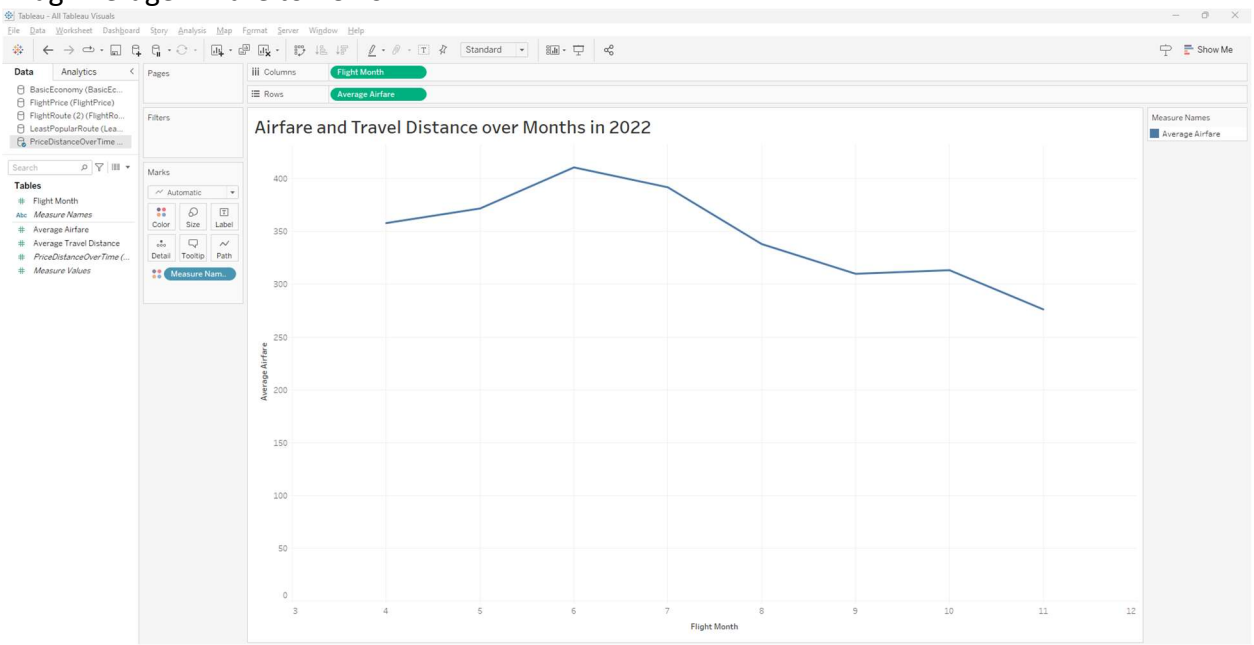
Type	Field Name	Physical Table	Remote F...
#	Flight ...	PriceDistanceOverTime.txt	F1
#	Avera...	PriceDistanceOverTime.txt	F2
#	Avera...	PriceDistanceOverTime.txt	F3

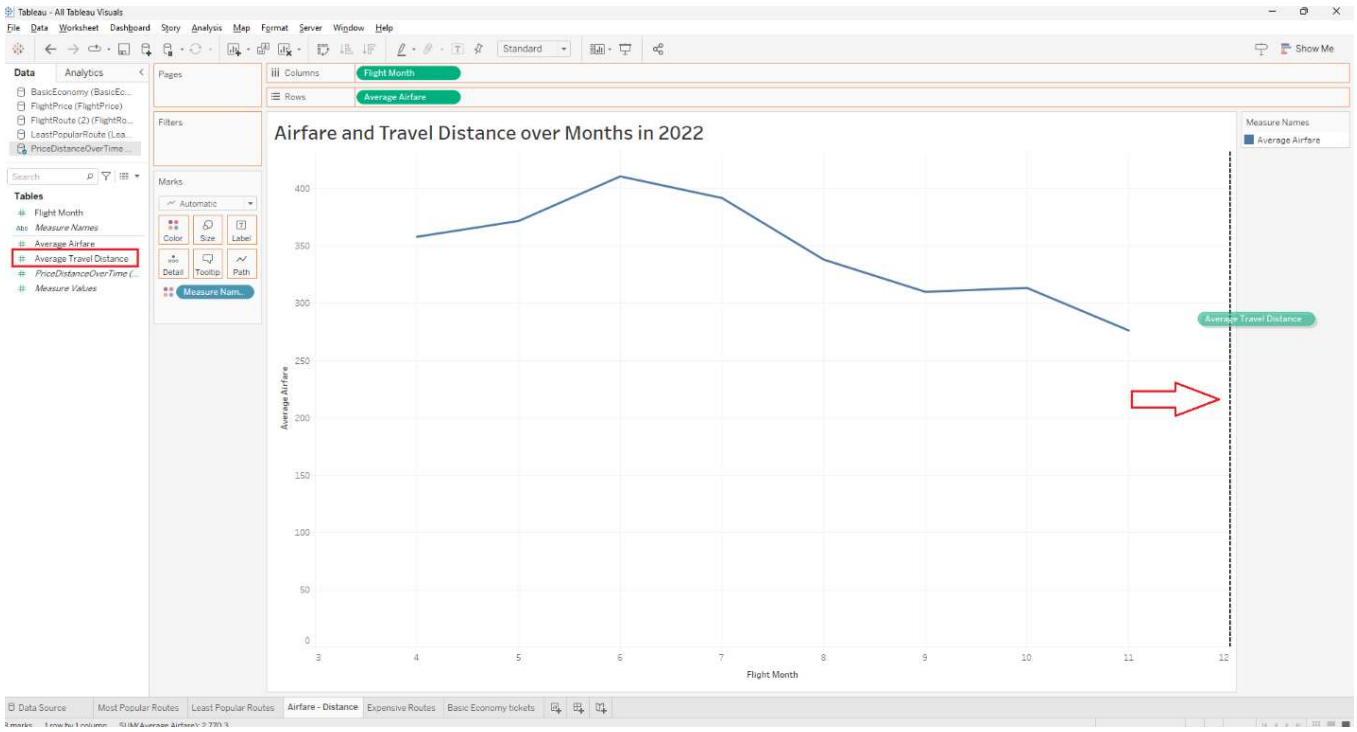
PriceDistanceOverTime.txt Flight Month	PriceDistanceOverTime.txt Average Airfare	PriceDistanceOverTime.txt Average Travel Distance
8	338.140	1,606.340
10	313.390	1,657.820
11	276.230	1,671.290
5	371.850	1,571.600
9	309.980	1,659.030
4	358.010	1,593.090
6	410.760	1,560.080
7	391.900	1,565.840

Because the measures have been aggregated using HiveQL, we do not need to aggregate data within Tableau. Go to **Analysis** tab and uncheck **Aggregate Measures**

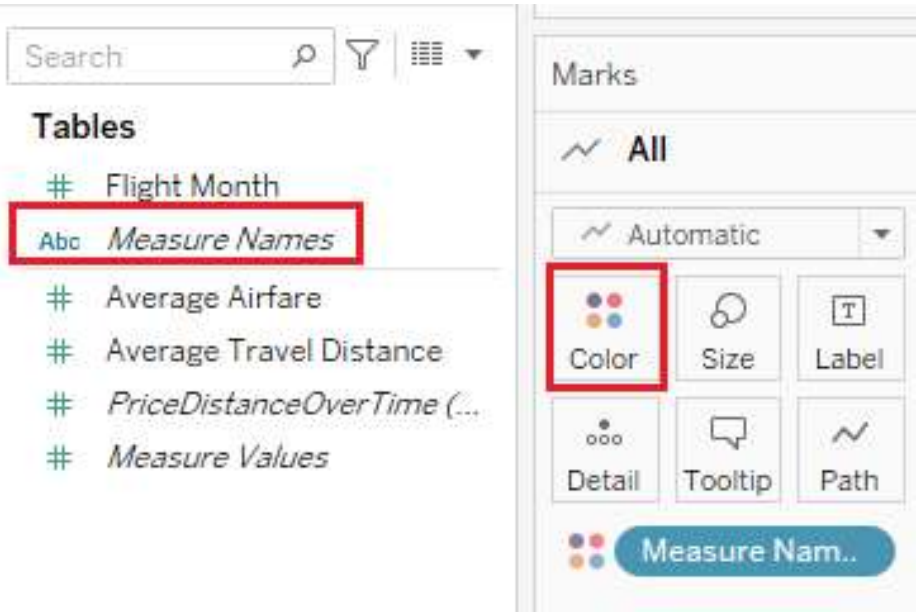


Drag Flight Month to **Columns**
Drag Average Airfare to **Rows**

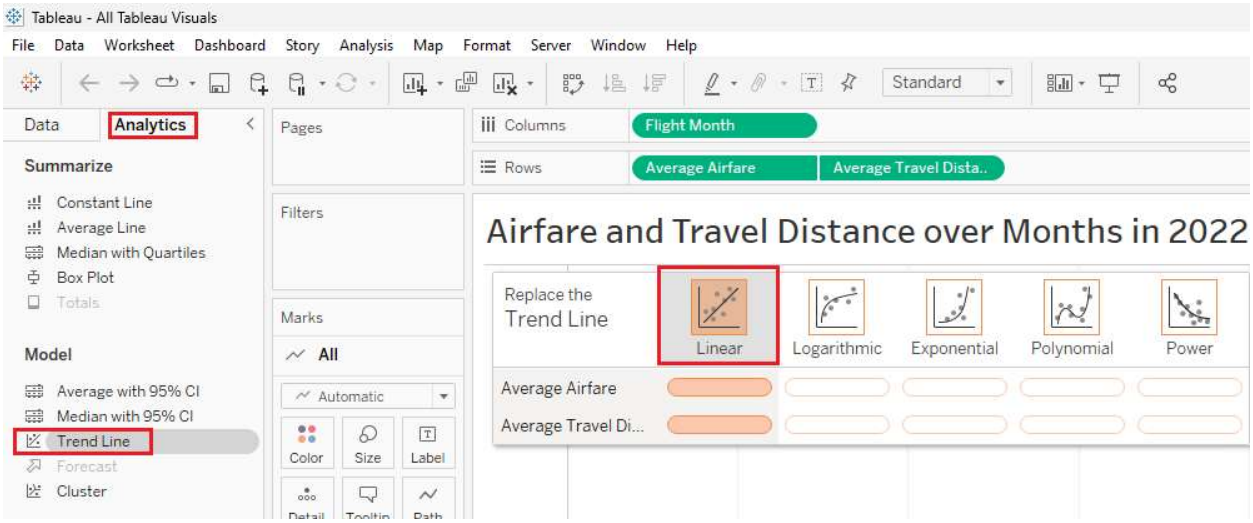


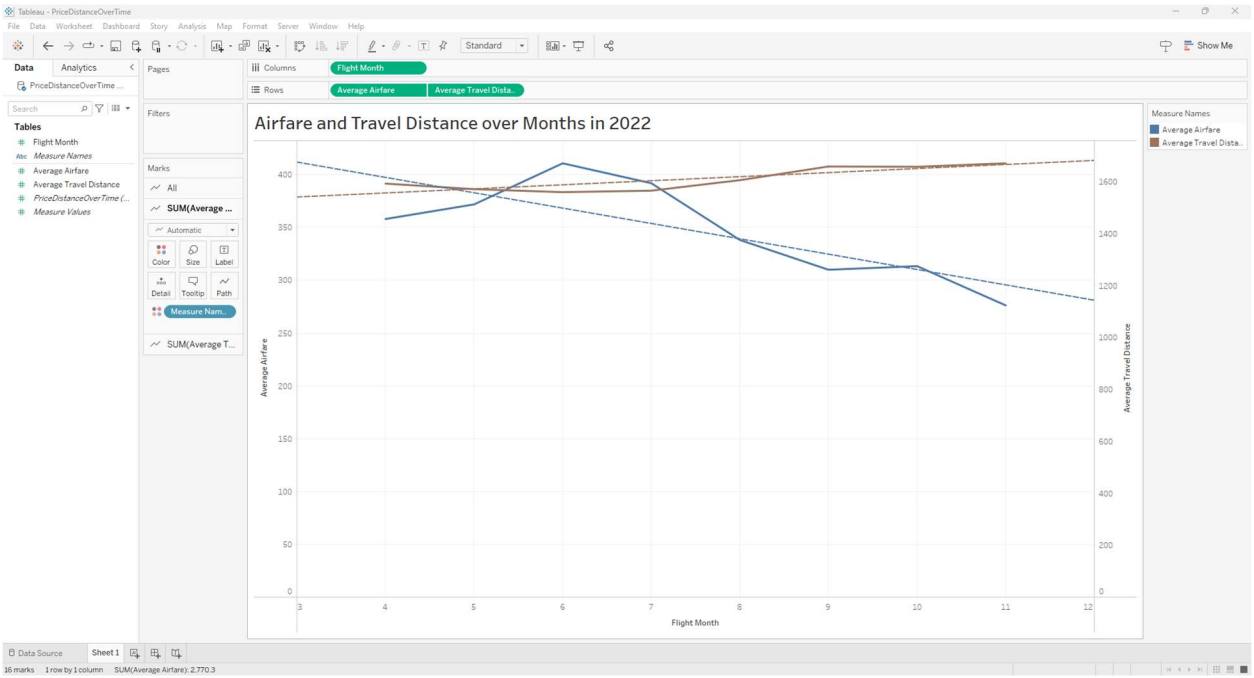


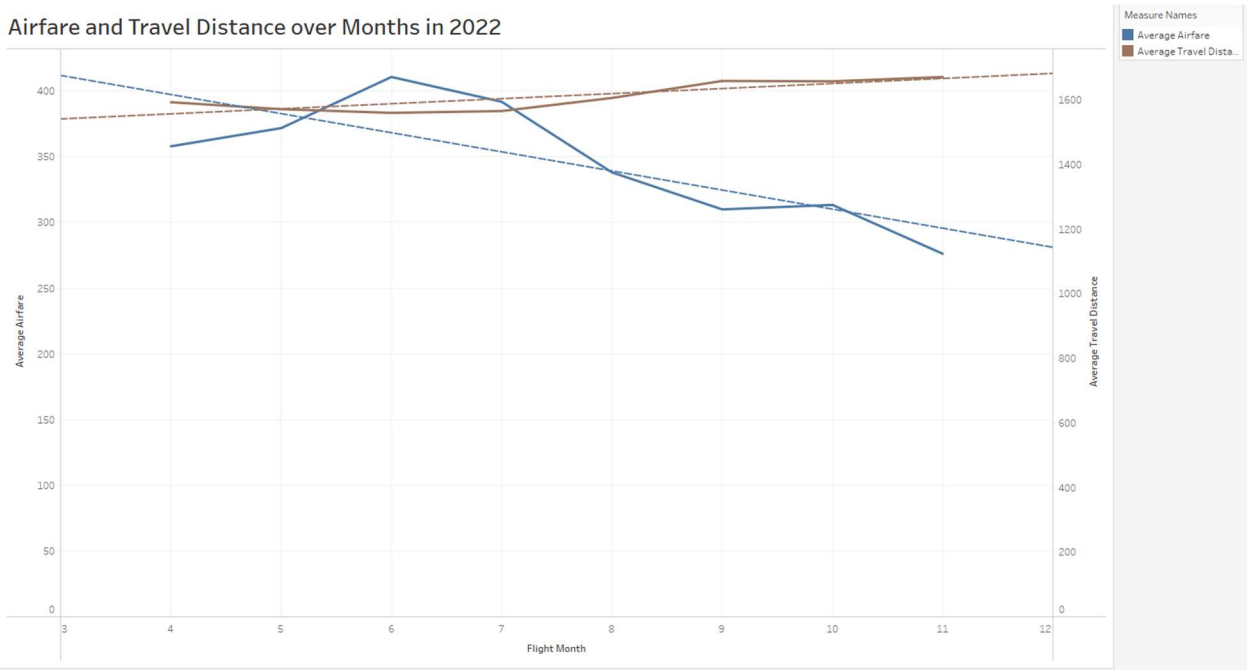
Drag Measure Names to **Color**



Click on **Analytics** tab, drag **Trend Line** to **Linear** type to add the dotted trend line to line chart.

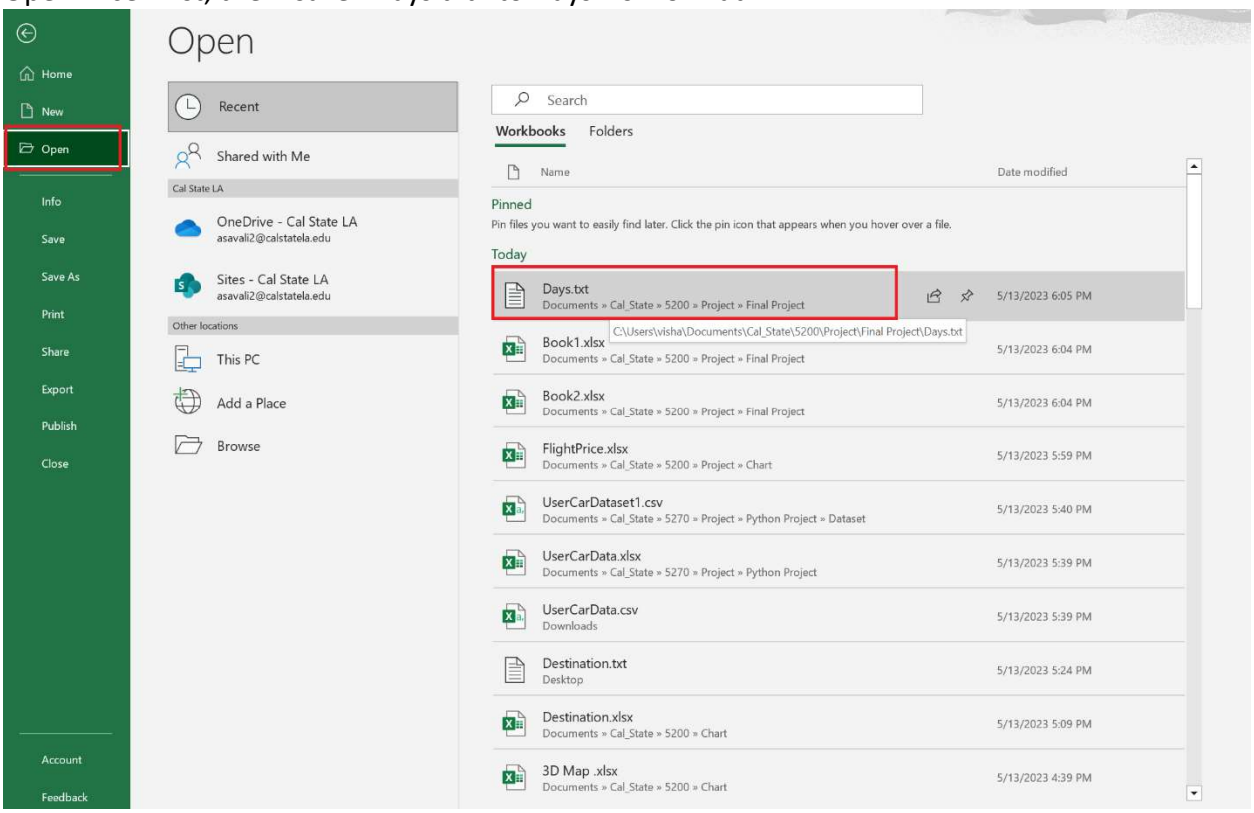


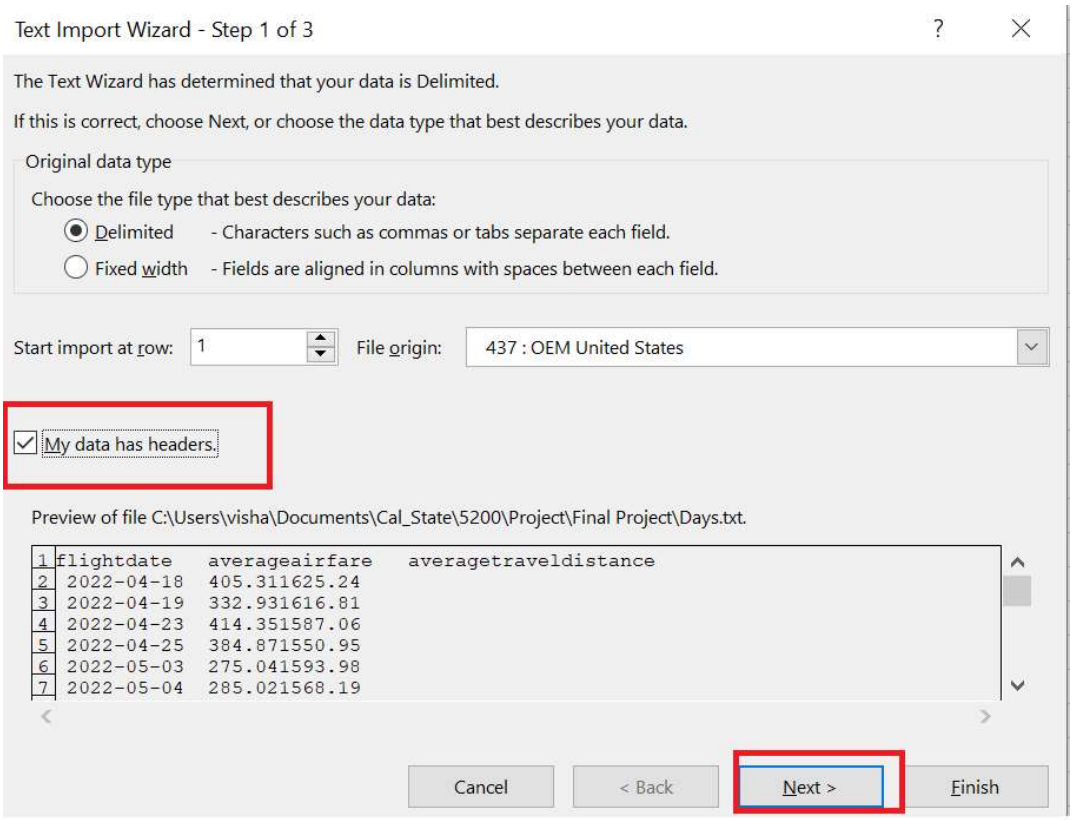


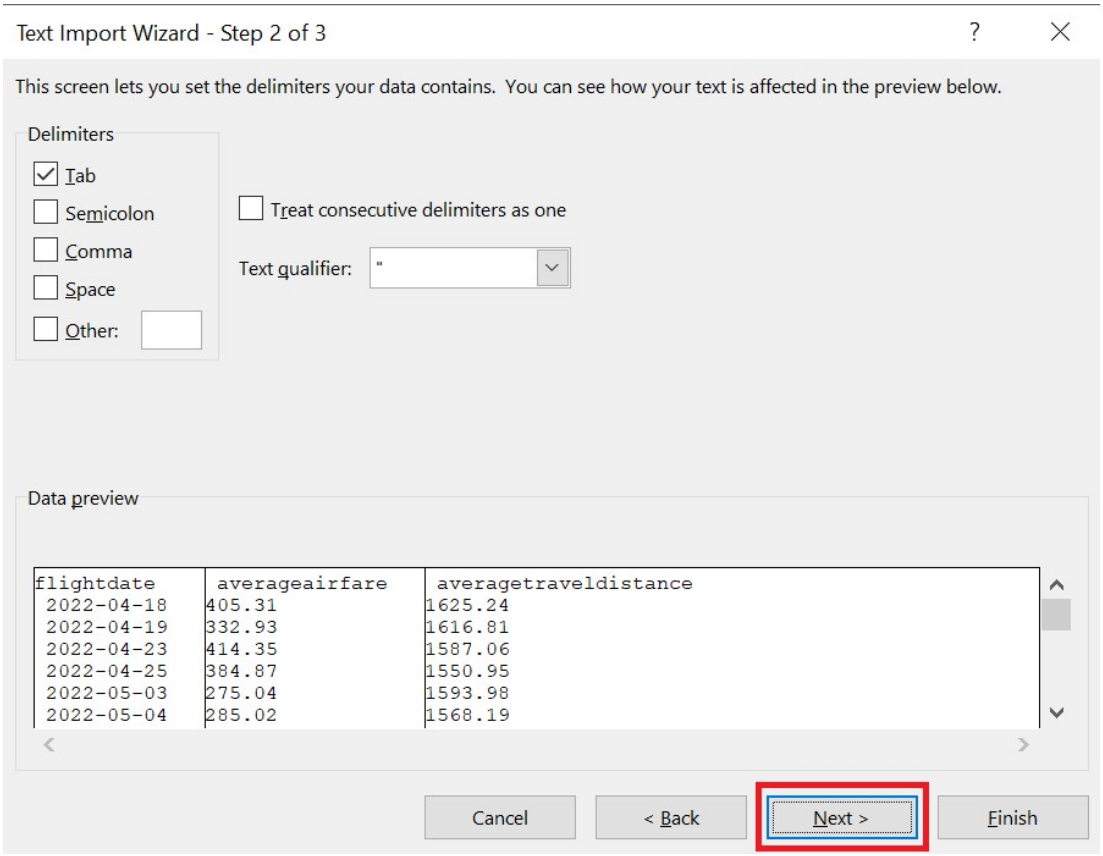


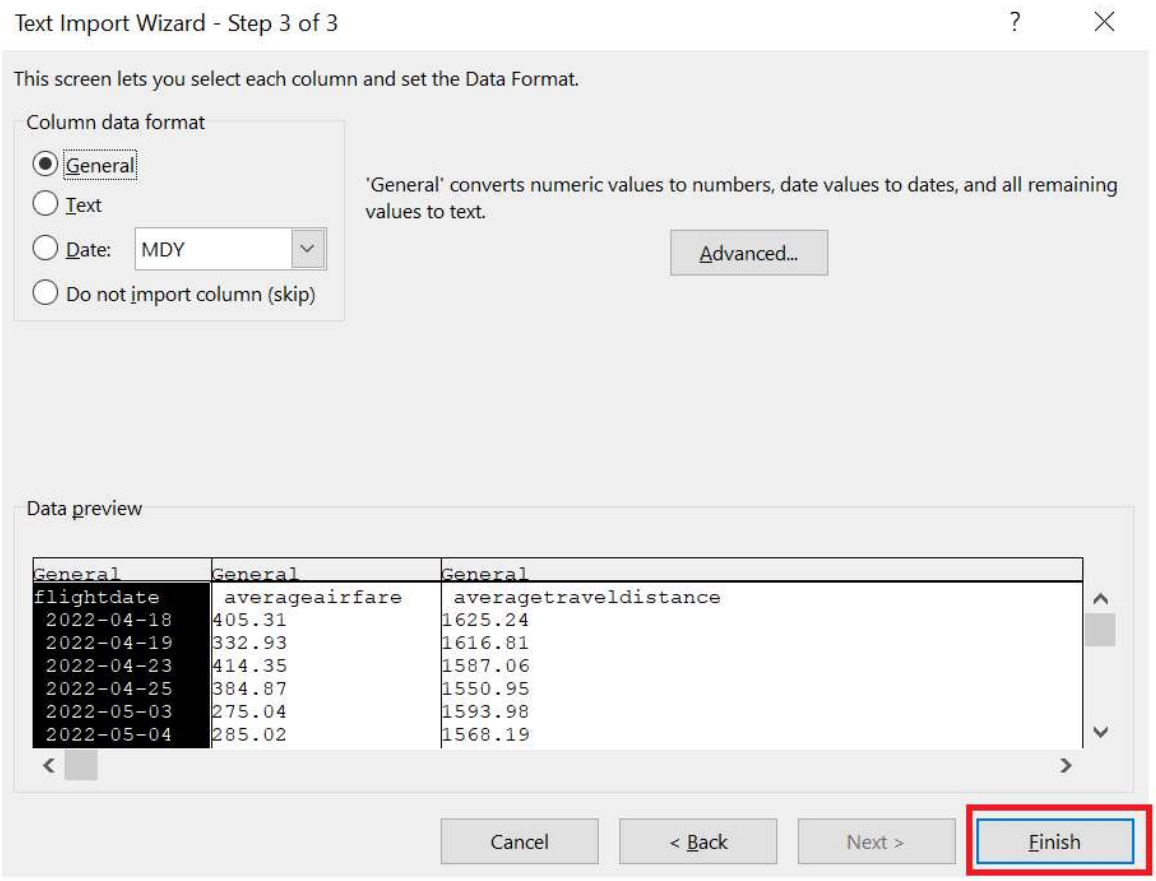
5. Average Airfare by Days per Month (bar chart in Power BI)

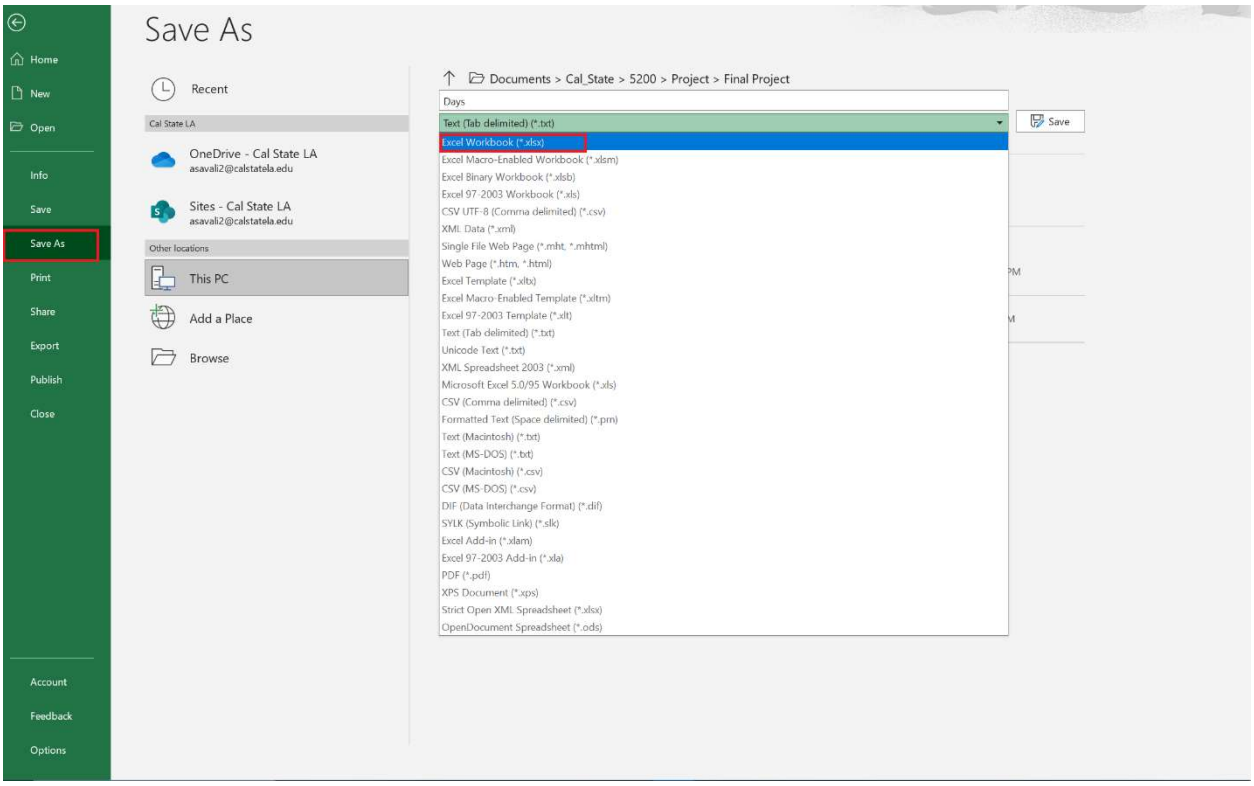
Open Excel first, then save "Days.txt" to Days.xlsx format.

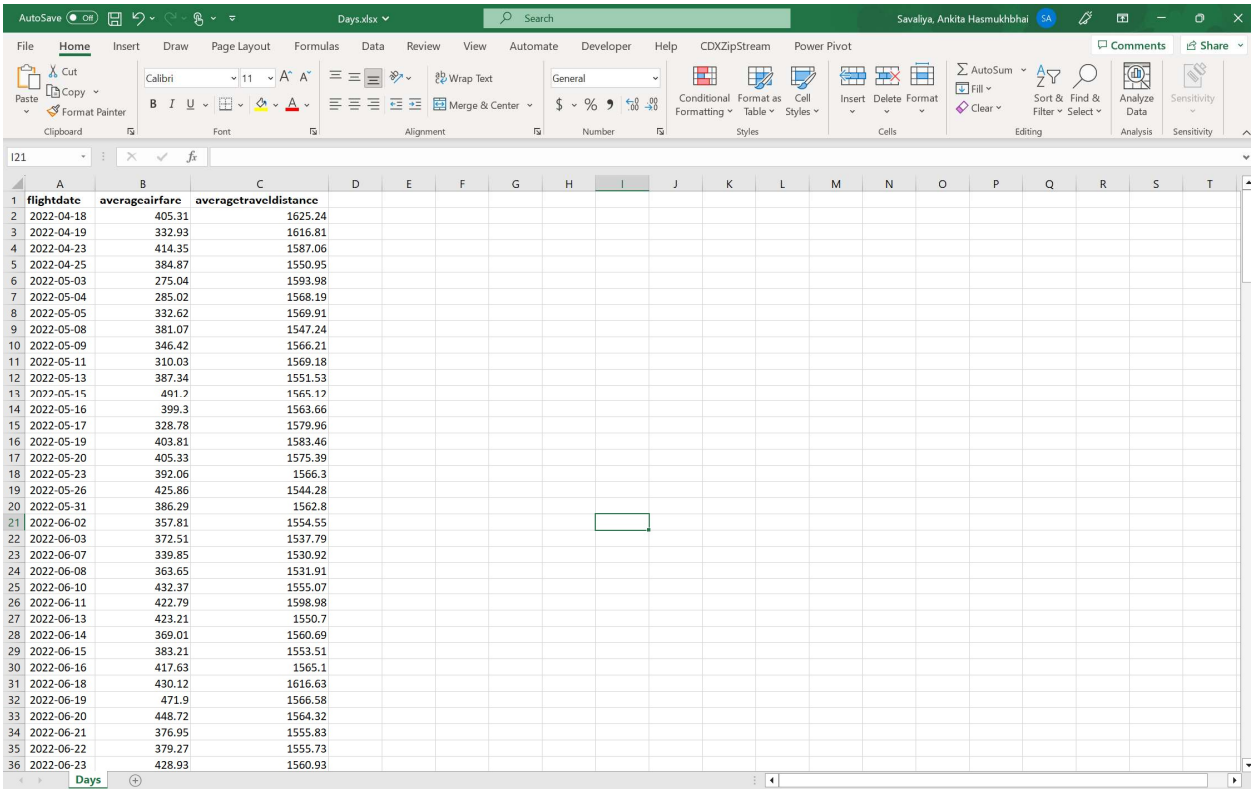



Text Import Wizard - Step 1 of 3
The Text Wizard has determined that your data is Delimited.
If this is correct, choose Next, or choose the data type that best describes your data.
Original data type
Choose the file type that best describes your data:
Delimited - Characters such as commas or tabs separate each field.
Fixed width - Fields are aligned in columns with spaces between each field.
Start import at row: 1
File origin: 437 : OEM United States
My data has headers.
Preview of file C:\Users\visha\Documents\Cal_State\5200\Project\Final Project\Days.txt.
1 flightdate averageairfare averagetraveldistance
2 2022-04-18 405.311625.24
3 2022-04-19 332.931616.81
4 2022-04-23 414.351587.06
5 2022-04-25 384.871550.95
6 2022-05-03 275.041593.98
7 2022-05-04 285.021568.19
Cancel
< Back
Next >
Finish


Text Import Wizard - Step 2 of 3
This screen lets you set the delimiters your data contains. You can see how your text is affected in the preview below.
Delimiters
Tab
Semicolon
Comma
Space
Other:
Treat consecutive delimiters as one
Text qualifier: "
Data preview
flightdate averageairfare averagetraveldistance
2022-04-18 405.31 1625.24
2022-04-19 332.93 1616.81
2022-04-23 414.35 1587.06
2022-04-25 384.87 1550.95
2022-05-03 275.04 1593.98
2022-05-04 285.02 1568.19
Cancel
< Back
Next >
Finish

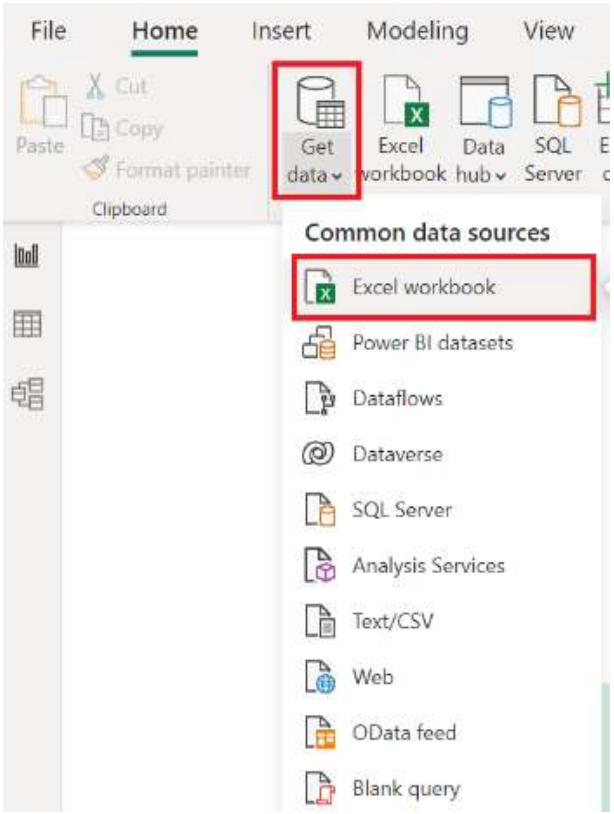

Text Import Wizard - Step 3 of 3
This screen lets you select each column and set the Data Format.
Column data format
General
Text
Date: MDY
Do not import column (skip)
'General' converts numeric values to numbers, date values to dates, and all remaining values to text.
Advanced...
Data preview
General General General
flightdate averageairfare averagetraveldistance
2022-04-18 405.31 1625.24
2022-04-19 332.93 1616.81
2022-04-23 414.35 1587.06
2022-04-25 384.87 1550.95
2022-05-03 275.04 1593.98
2022-05-04 285.02 1568.19
Cancel
< Back
Next >
Finish


Save As
Home
New
Open
Info
Save
Save As
Print
Share
Export
Publish
Close
Account
Feedback
Options
Recent
Cal State LA
OneDrive - Cal State LA
asavali2@calstatela.edu
Sites - Cal State LA
asavali2@calstatela.edu
Other locations
This PC
Add a Place
Browse
Documents > Cal_State > 5200 > Project > Final Project
Days
Text (Tab delimited) (*.txt)
Save
Excel Workbook (*.xlsx)
Excel Macro-Enabled Workbook (*.xlsm)
Excel Binary Workbook (*.xlsb)
Excel 97-2003 Workbook (*.xls)
CSV UTF-8 (Comma delimited) (*.csv)
XML Data (*.xml)
Single File Web Page (*.mht, *.mhtml)
Web Page (*.htm, *.html)
Excel Template (*.xltx)
Excel Macro-Enabled Template (*.xltm)
Excel 97-2003 Template (*.xlt)
Text (Tab delimited) (*.txt)
Unicode Text (*.txt)
XML Spreadsheet 2003 (*.xml)
Microsoft Excel 5.0/95 Workbook (*.xls)
CSV (Comma delimited) (*.csv)
Formatted Text (Space delimited) (*.prn)
Text (Macintosh) (*.txt)
Text (MS-DOS) (*.txt)
CSV (Macintosh) (*.csv)
CSV (MS-DOS) (*.csv)
DIF (Data Interchange Format) (*.dif)
SYLK (Symbolic Link) (*.slk)
Excel Add-in (*.xlam)
Excel 97-2003 Add-in (*.xla)
PDF (*.pdf)
XPS Document (*.xps)
Strict Open XML Spreadsheet (*.xlsx)
OpenDocument Spreadsheet (*.ods)

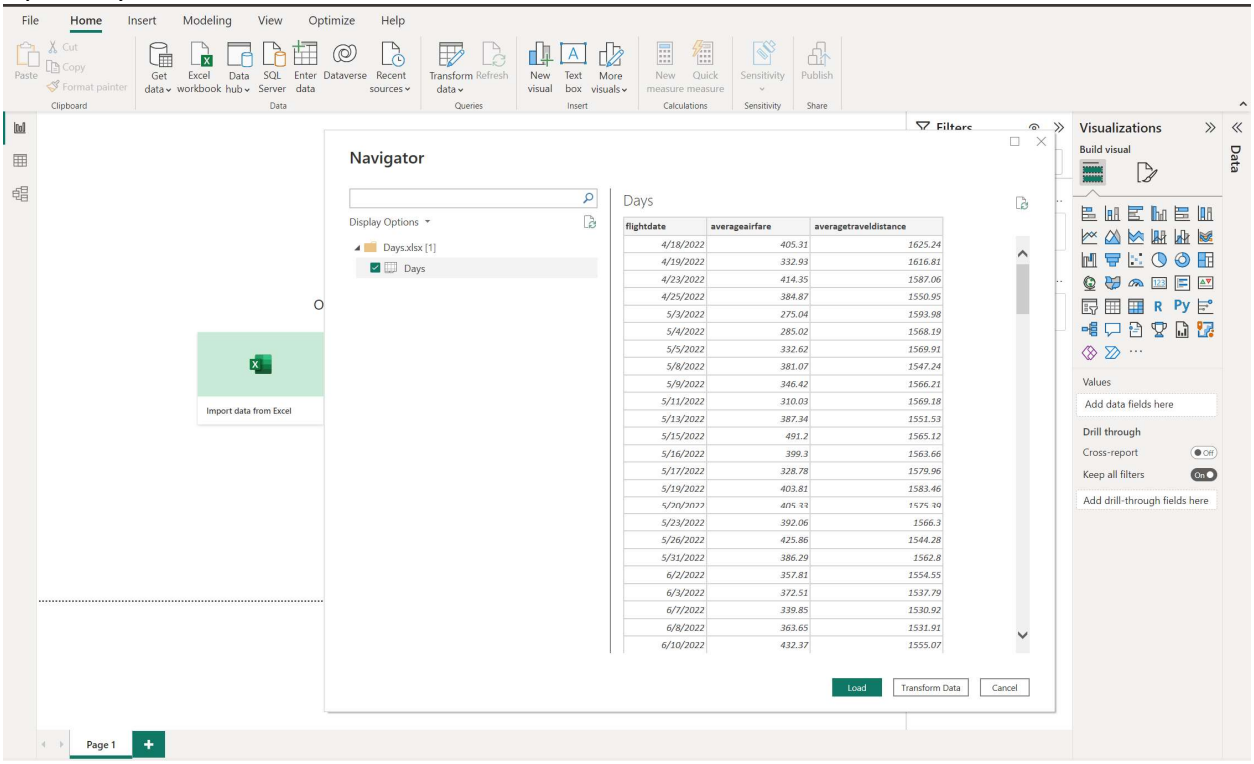


	flightdate	averageairfare	averagetraveldistance
2	2022-04-18	405.31	1625.24
3	2022-04-19	332.93	1616.81
4	2022-04-23	414.35	1587.06
5	2022-04-25	384.87	1550.95
6	2022-05-03	275.04	1593.98
7	2022-05-04	285.02	1568.19
8	2022-05-05	332.62	1569.91
9	2022-05-08	381.07	1547.24
10	2022-05-09	346.42	1566.21
11	2022-05-11	310.03	1569.18
12	2022-05-13	387.34	1551.53
13	2022-05-15	491.2	1565.12
14	2022-05-16	399.3	1563.66
15	2022-05-17	328.78	1579.96
16	2022-05-19	403.81	1583.46
17	2022-05-20	405.33	1575.39
18	2022-05-23	392.06	1566.3
19	2022-05-26	425.86	1544.28
20	2022-05-31	386.29	1562.8
21	2022-06-02	357.81	1554.55
22	2022-06-03	372.51	1537.79
23	2022-06-07	339.85	1530.92
24	2022-06-08	363.65	1531.91
25	2022-06-10	432.37	1555.07
26	2022-06-11	422.79	1598.98
27	2022-06-13	423.21	1550.7
28	2022-06-14	369.01	1560.69
29	2022-06-15	383.21	1553.51
30	2022-06-16	417.63	1565.1
31	2022-06-18	430.12	1616.63
32	2022-06-19	471.9	1566.58
33	2022-06-20	448.72	1564.32
34	2022-06-21	376.95	1555.83
35	2022-06-22	379.27	1555.73
36	2022-06-23	428.93	1560.93

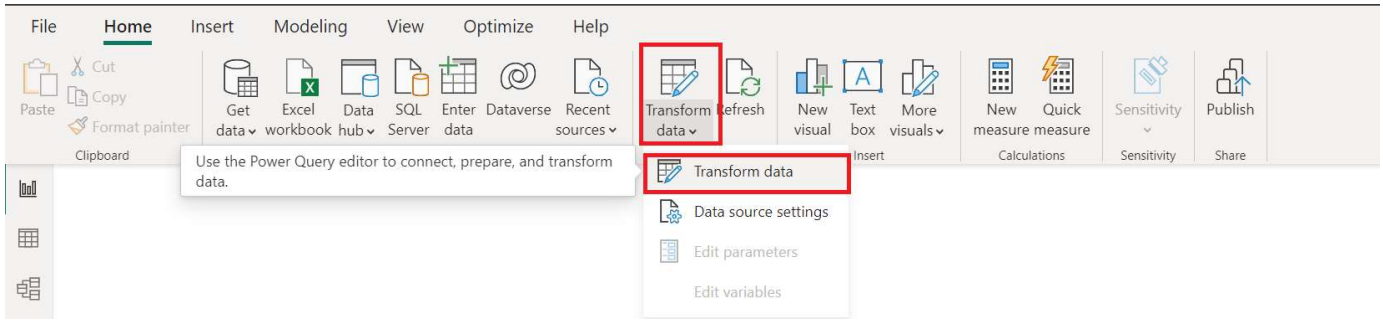
Now Open Power BI Desktop:



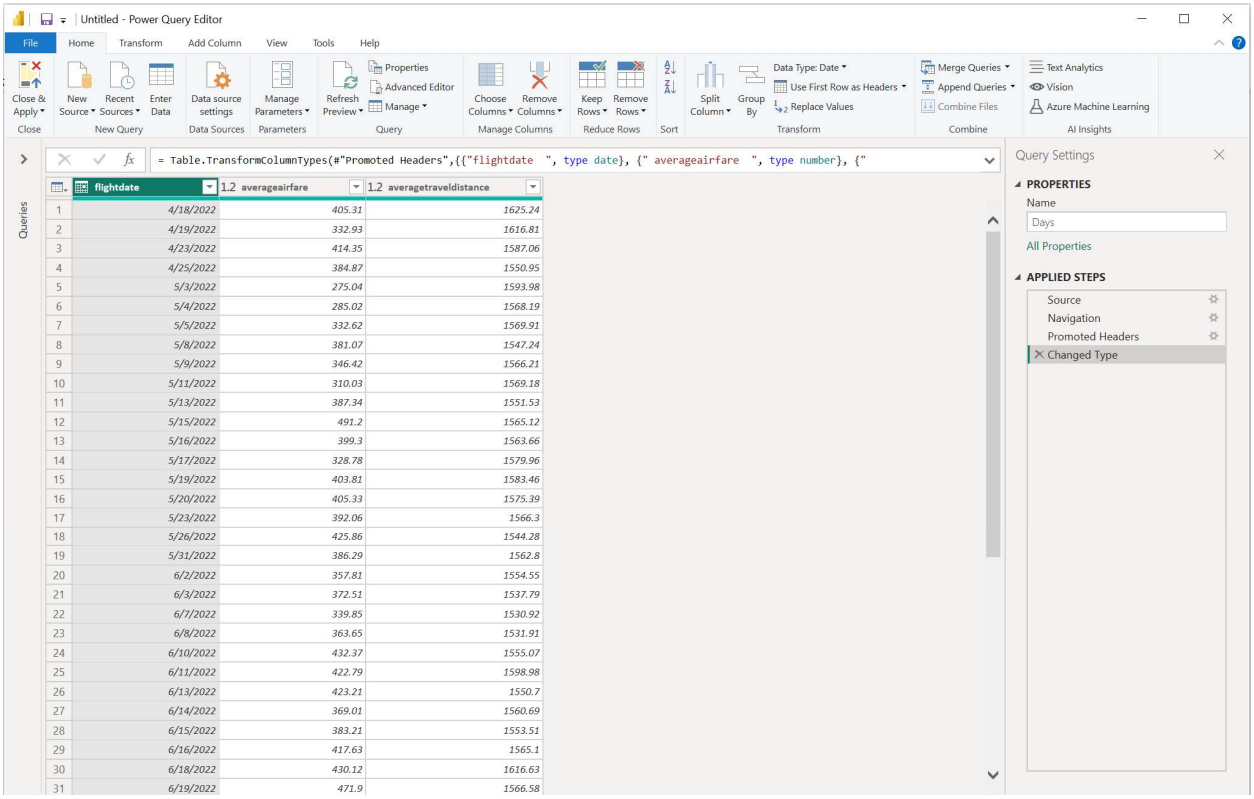
Open Days.xlsx:



Select **Transform data → Transform data**



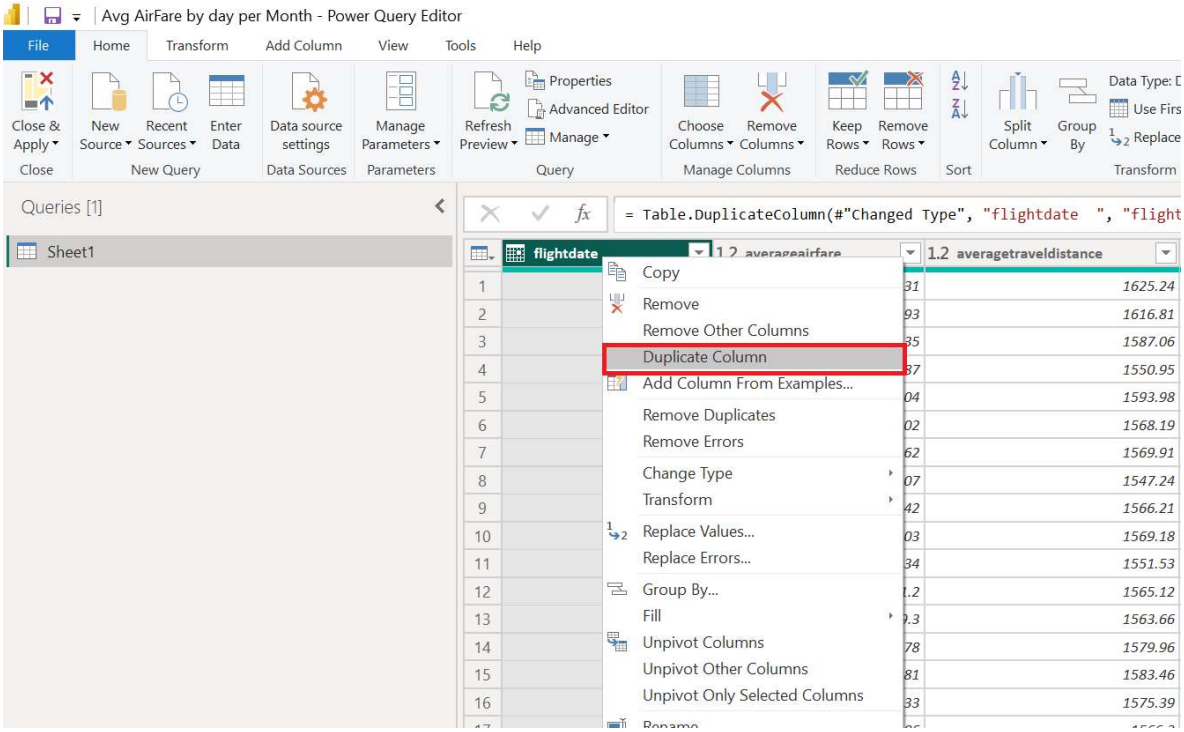
It shows:



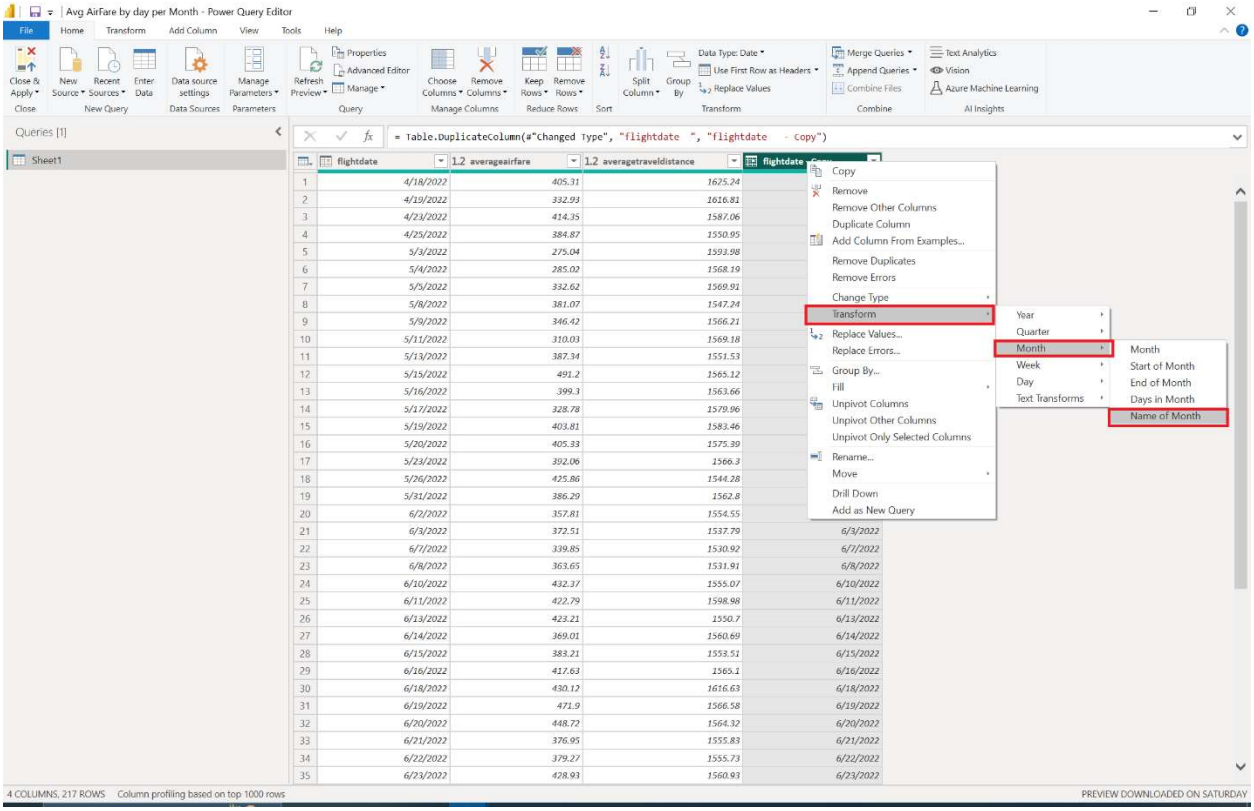
	flightdate	averageairfare	averagetraveldistance
1	4/18/2022	405.31	1625.24
2	4/19/2022	332.93	1616.81
3	4/23/2022	414.35	1587.06
4	4/25/2022	384.87	1550.95
5	5/3/2022	275.04	1593.98
6	5/4/2022	285.02	1568.19
7	5/5/2022	332.62	1569.91
8	5/8/2022	381.07	1547.24
9	5/9/2022	346.42	1566.21
10	5/11/2022	310.03	1569.18
11	5/13/2022	387.34	1551.53
12	5/15/2022	491.2	1565.12
13	5/16/2022	399.3	1563.66
14	5/17/2022	328.78	1579.96
15	5/19/2022	403.81	1583.46
16	5/20/2022	405.33	1575.39
17	5/23/2022	392.06	1566.3
18	5/26/2022	425.86	1544.28
19	5/31/2022	386.29	1562.8
20	6/2/2022	357.81	1554.55
21	6/3/2022	372.51	1537.79
22	6/7/2022	339.85	1530.92
23	6/8/2022	363.65	1531.91
24	6/10/2022	432.37	1555.07
25	6/11/2022	422.79	1598.98
26	6/13/2022	423.21	1550.7
27	6/14/2022	369.01	1560.69
28	6/15/2022	383.21	1553.51
29	6/16/2022	417.63	1565.1
30	6/18/2022	430.12	1616.63
31	6/19/2022	471.9	1566.58

In order to get the Month, Year, Weekday Name, Day of the Week Number(0-7) and Month Number(1-12):

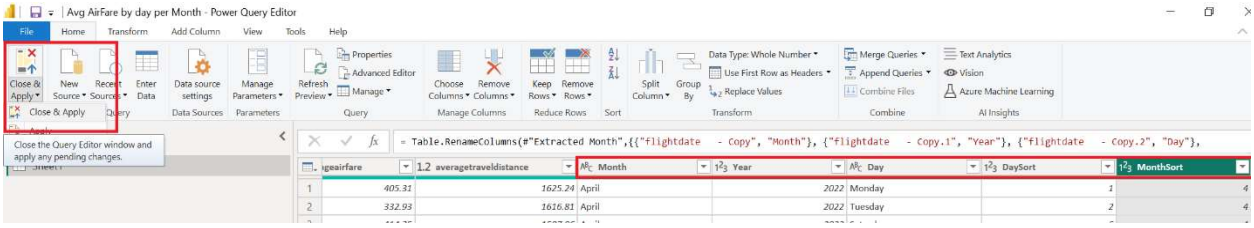
Step 1: Duplicate flightdate column by right clicking on the column



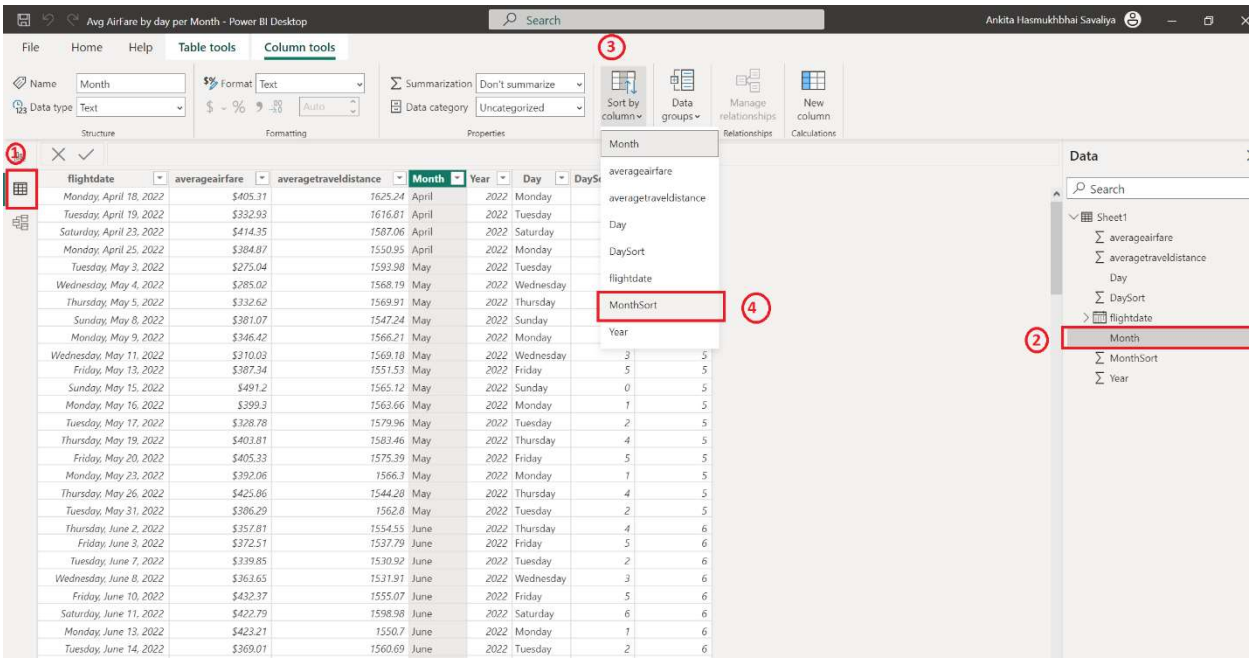
Step 2: Right click on Duplicated column, Transform → Month → Name of Month
Step 3: Follow step 1 and 2 for the Year, Weekday Name, Day of the Week Number(0-7) and Month Number(1-12)



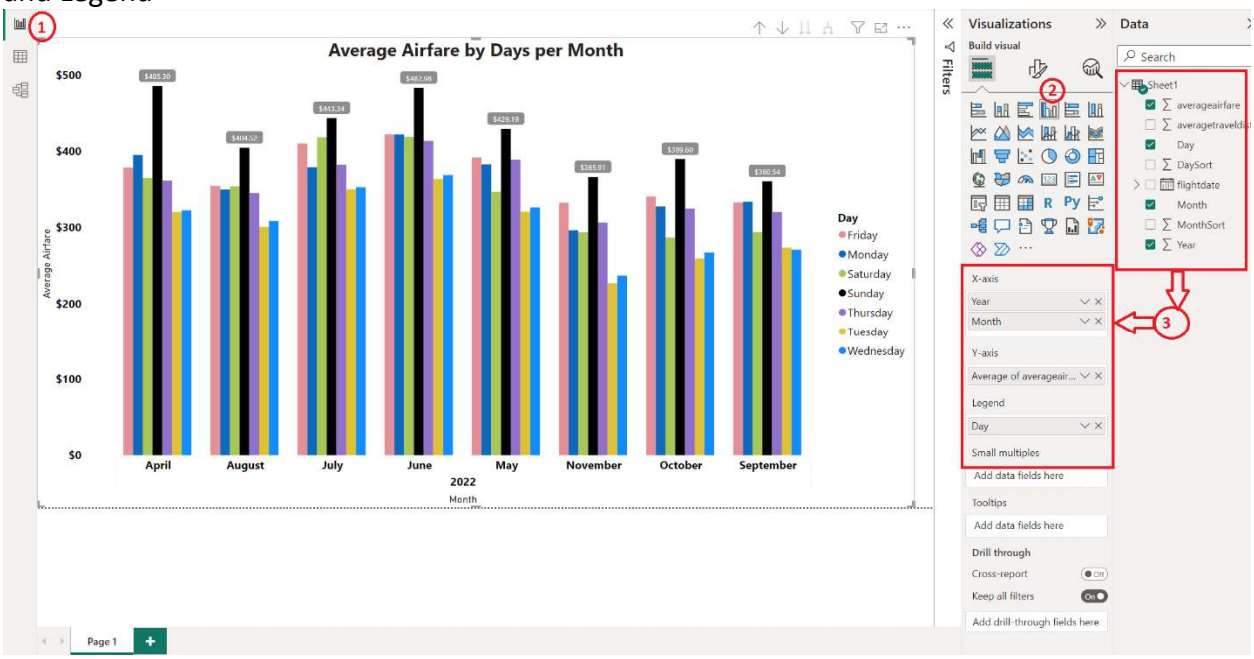
Step 4: Double Click on the All Duplicated Columns and Give appropriate Names
Step 5: Close & Apply of Transform Data Tab



Step 6: Data View→Select Month Column→Select Sort By Column→Select MonthSort
Step 7: Select Day Column→Select Sort By Column→Select DaySort

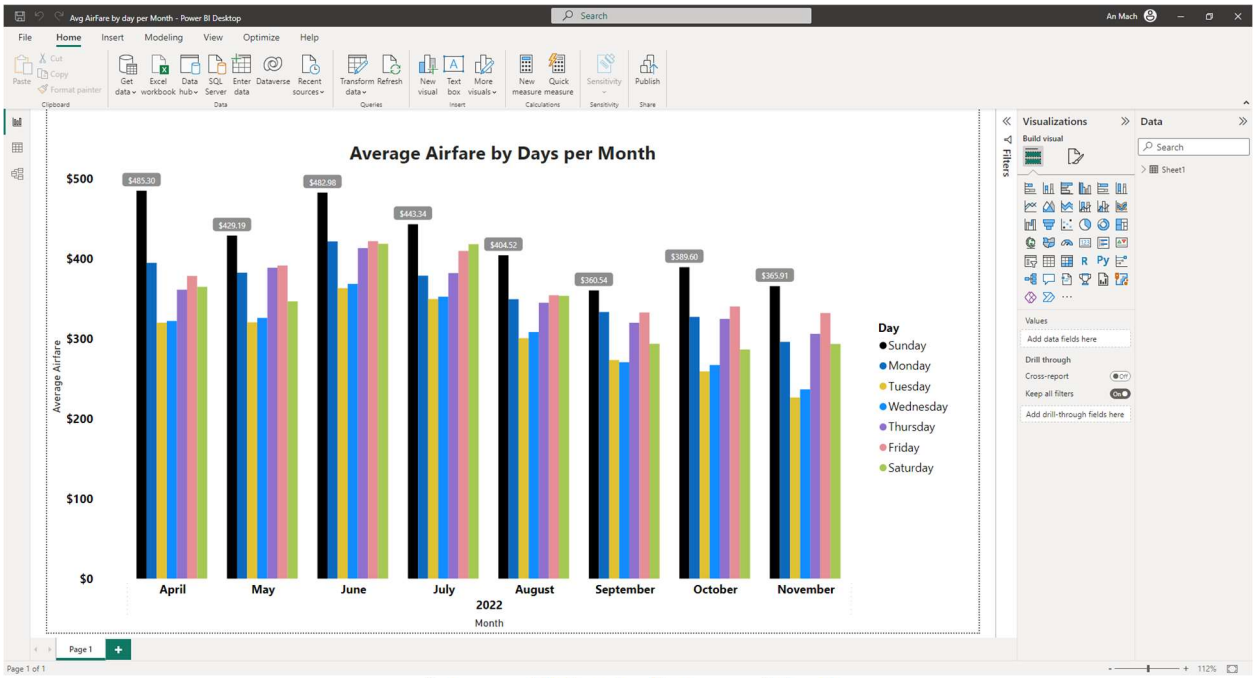


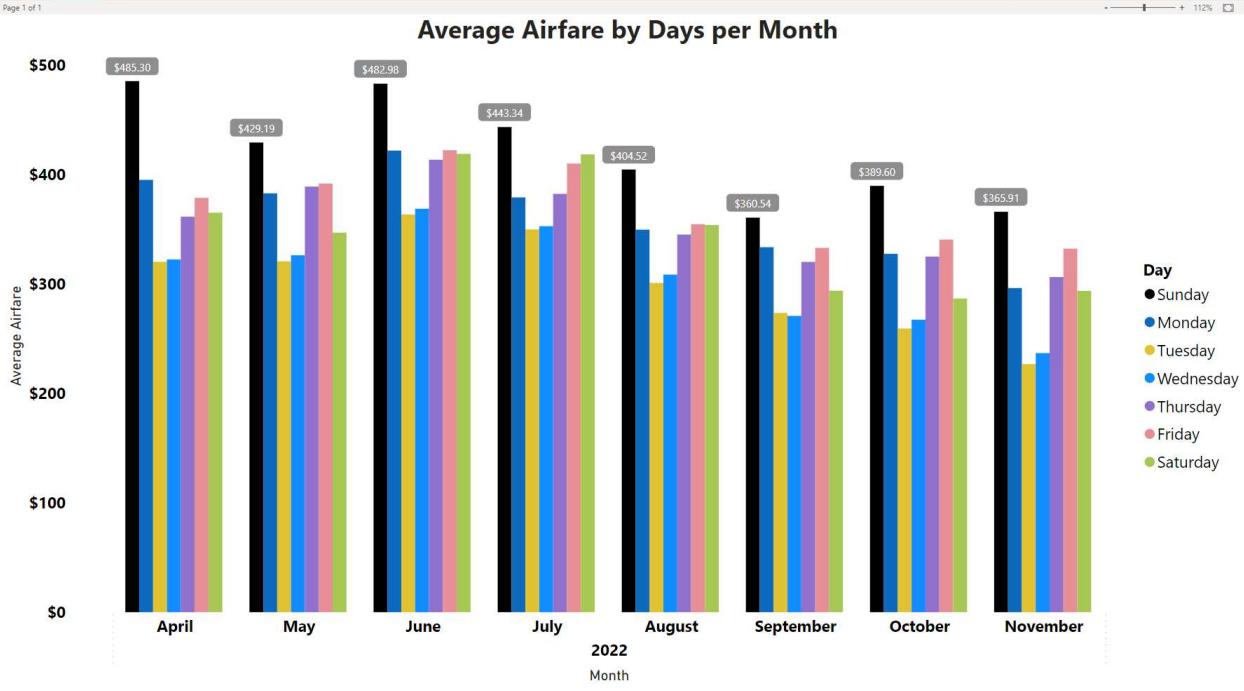
Step 8: Go to Report View→Select Clustered Column Chart→ Drag and Drop Columns to Axis and Legend



Step 9: Use format Tab to give titles to axis and color coding.

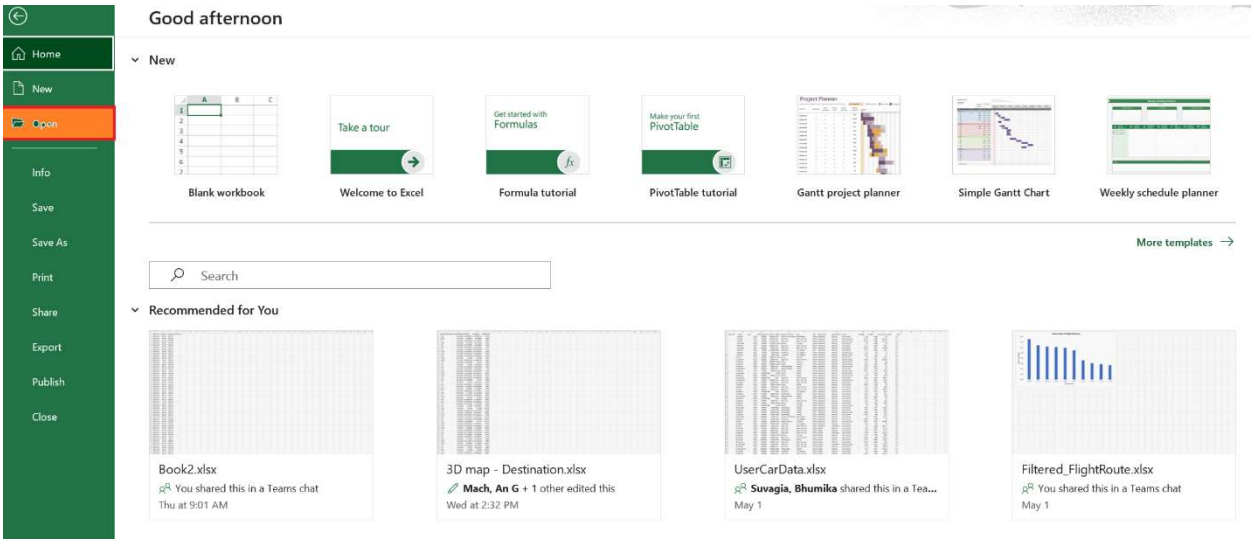
Final Result:



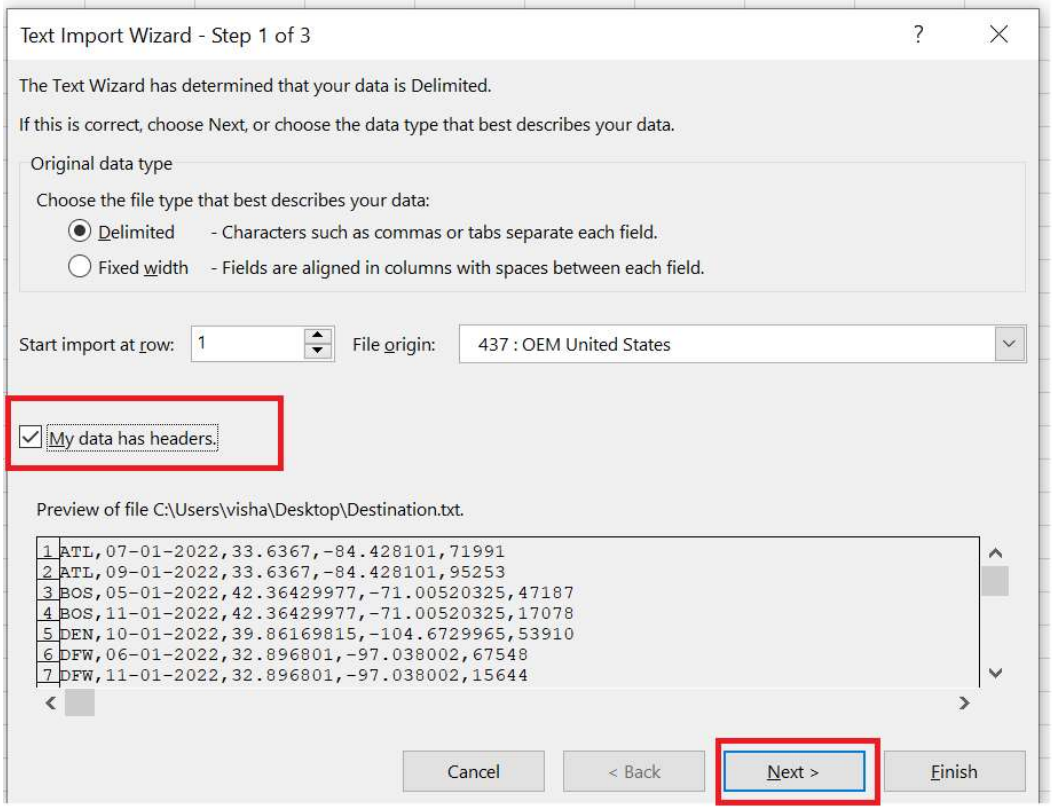


6. Which are the most popular destinations over the months?

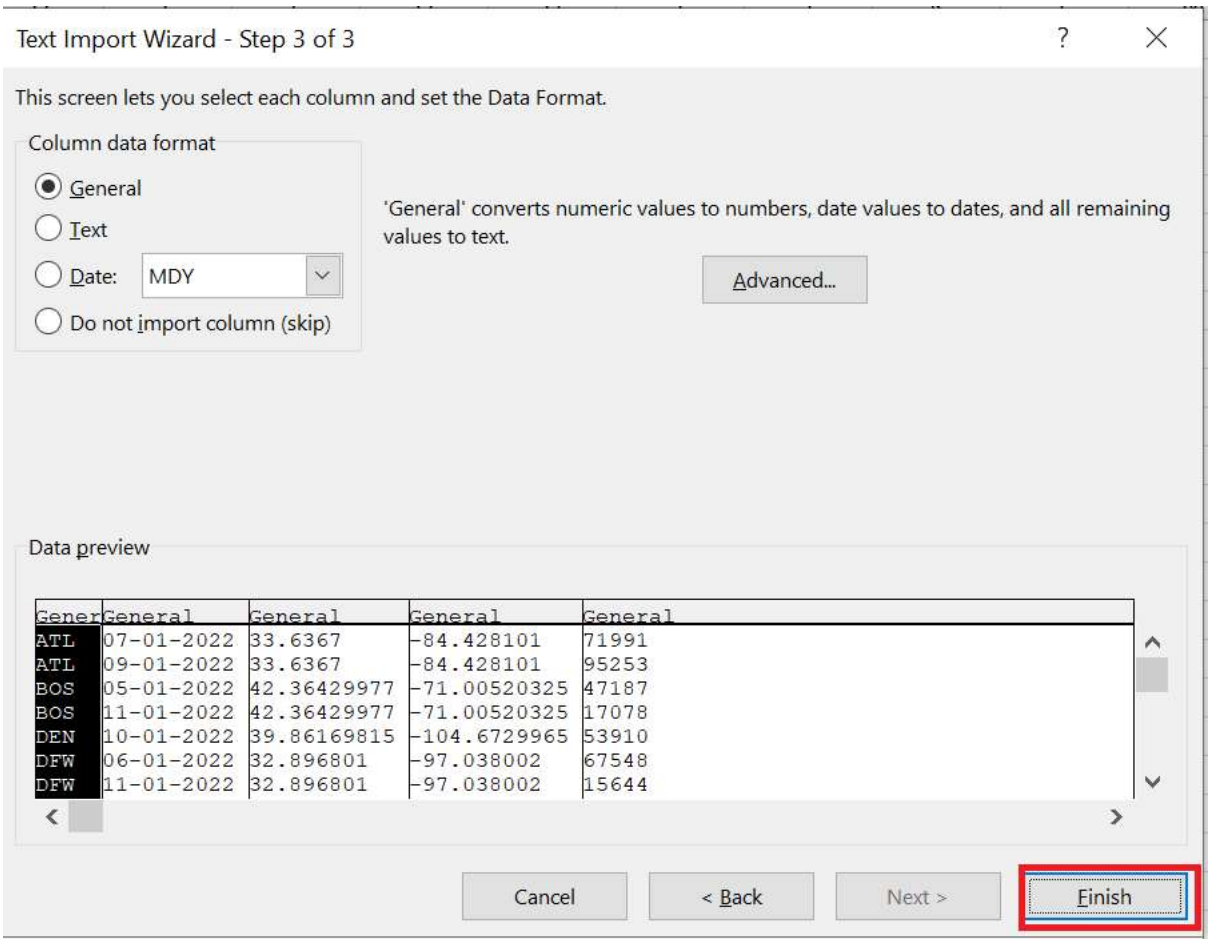
Open “Destination.txt” at Excel. Open Excel first, then open the data file from Excel in order to read the data as multiple records in multiple rows.



Select the file Destination.txt, and select **My data has headers**. Then, click **Next**:



Select **Finish**:

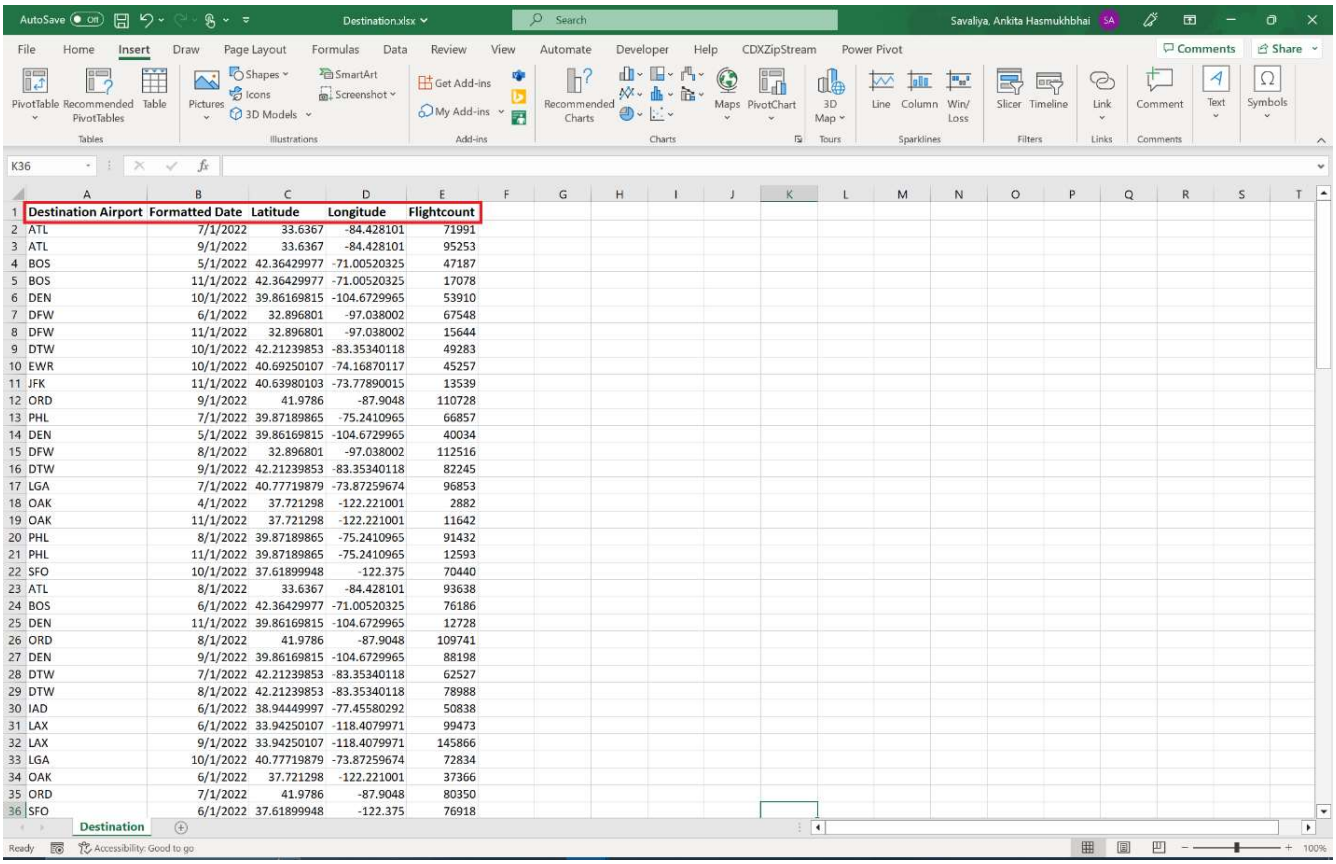


Insert the header to each column:

	Destination Airport	Formatted Date	Latitude	Longitude	Flightcount
2	ATL	7/1/2022	33.6367	-84.428101	71991
3	ATL	9/1/2022	33.6367	-84.428101	95253
4	BOS	5/1/2022	42.36429977	-71.00520325	47187
5	BOS	11/1/2022	42.36429977	-71.00520325	17078
6	DEN	10/1/2022	39.86169815	-104.6729965	53910
7	DFW	6/1/2022	32.896801	-97.038002	67548
8	DFW	11/1/2022	32.896801	-97.038002	15644
9	DTW	10/1/2022	42.21239853	-83.35340118	49283
10	EWR	10/1/2022	40.69250107	-74.16870117	45257
11	JFK	11/1/2022	40.63980103	-73.77890015	13539
12	ORD	9/1/2022	41.9786	-87.9048	110728
13	PHL	7/1/2022	39.87189865	-75.2410965	66857
14	DEN	5/1/2022	39.86169815	-104.6729965	40034
15	DFW	8/1/2022	32.896801	-97.038002	112516
16	DTW	9/1/2022	42.21239853	-83.35340118	82245

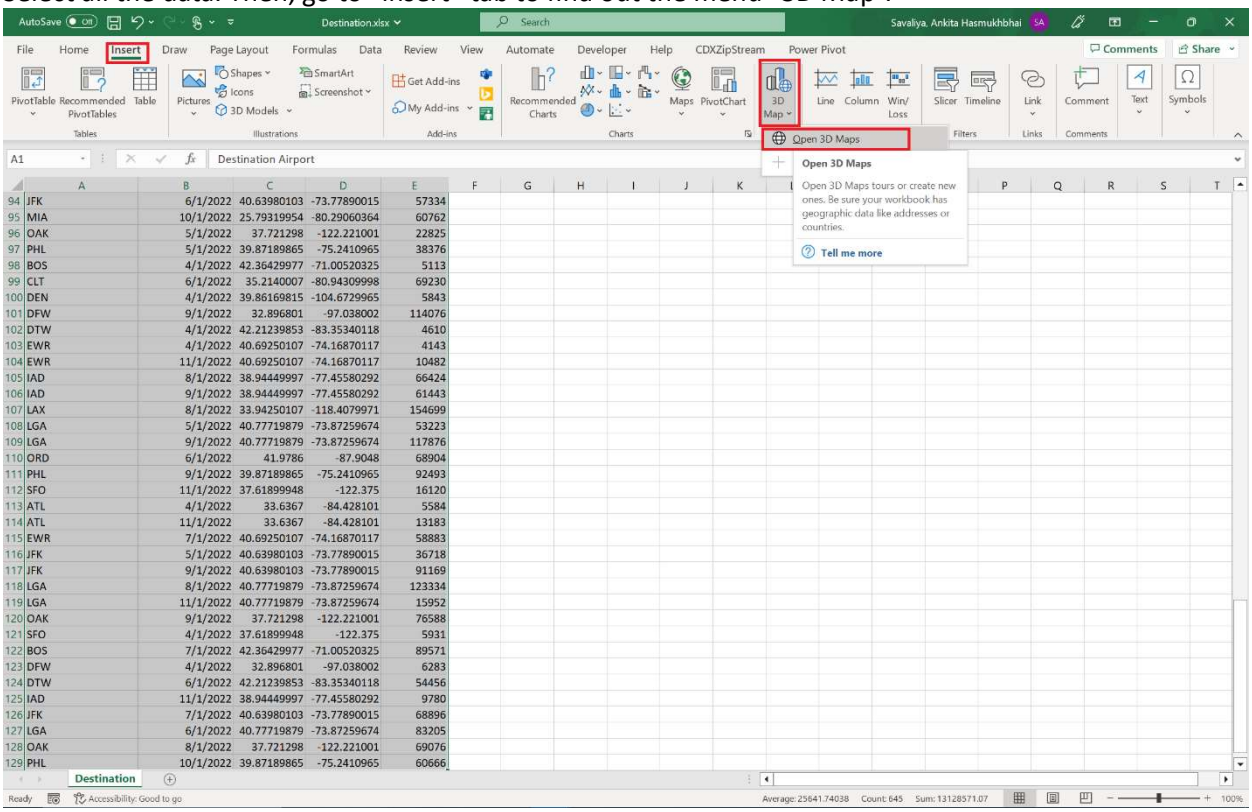
Save as xlsx:

Final Results:

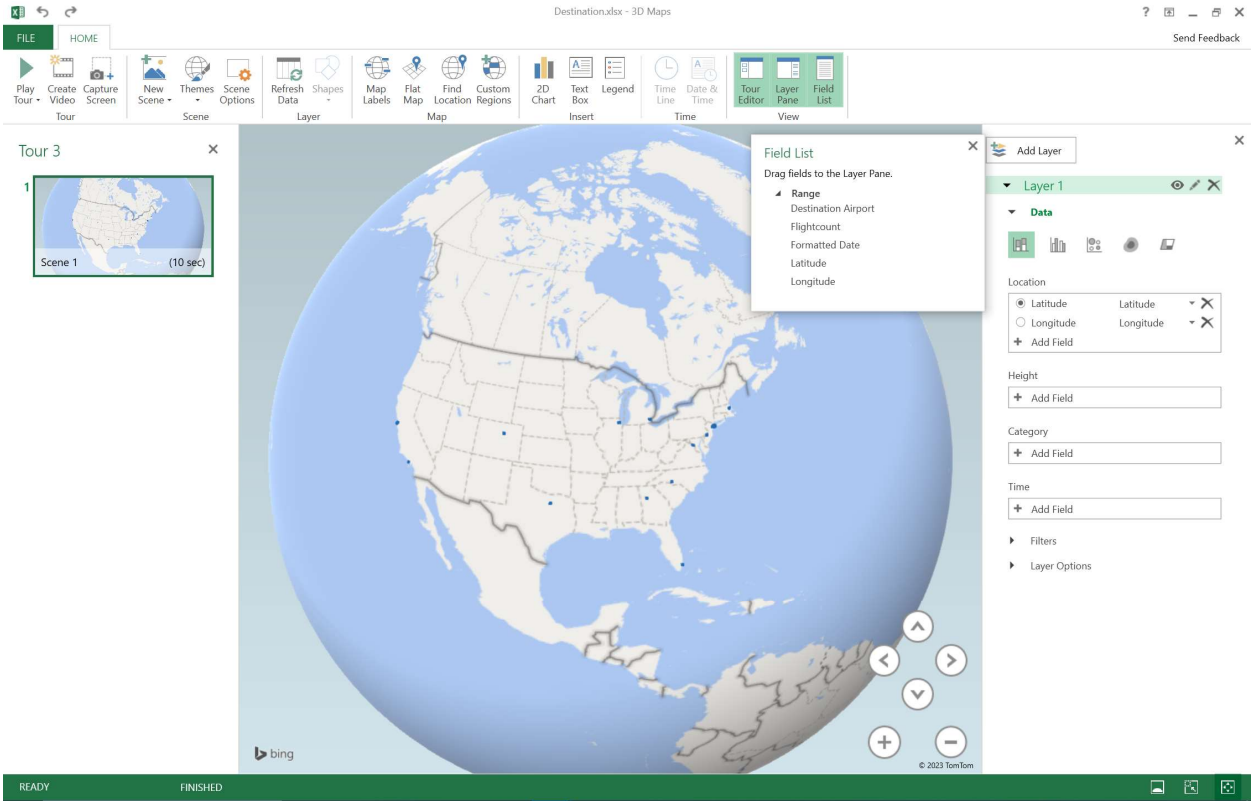


Destination Airport	Formatted Date	Latitude	Longitude	Flightcount
ATL	7/1/2022	33.6367	-84.428101	71991
ATL	9/1/2022	33.6367	-84.428101	95253
BOS	5/1/2022	42.36429977	-71.00520325	47187
BOS	11/1/2022	42.36429977	-71.00520325	17078
DEN	10/1/2022	39.86169815	-104.6729965	53910
DFW	6/1/2022	32.896801	-97.038002	67548
DFW	11/1/2022	32.896801	-97.038002	15644
DTW	10/1/2022	42.21239853	-83.35340118	49283
EWR	10/1/2022	40.69250107	-74.16870117	45257
JFK	11/1/2022	40.63980103	-73.77890015	13539
ORD	9/1/2022	41.9786	-87.9048	110728
PHL	7/1/2022	39.87189865	-75.2410965	66857
DEN	5/1/2022	39.86169815	-104.6729965	40034
DFW	8/1/2022	32.896801	-97.038002	112516
DTW	9/1/2022	42.21239853	-83.35340118	82245
LGA	7/1/2022	40.77719879	-73.87259674	96853
OAK	4/1/2022	37.721298	-122.221001	2882
OAK	11/1/2022	37.721298	-122.221001	11642
PHL	8/1/2022	39.87189865	-75.2410965	91432
PHL	11/1/2022	39.87189865	-75.2410965	12593
SFO	10/1/2022	37.61899948	-122.375	70440
ATL	8/1/2022	33.6367	-84.428101	93638
BOS	6/1/2022	42.36429977	-71.00520325	76186
DEN	11/1/2022	39.86169815	-104.6729965	12728
ORD	8/1/2022	41.9786	-87.9048	109741
DEN	9/1/2022	39.86169815	-104.6729965	88198
DTW	7/1/2022	42.21239853	-83.35340118	62527
DTW	8/1/2022	42.21239853	-83.35340118	78988
IAD	6/1/2022	38.94449997	-77.45580292	50838
LAX	6/1/2022	33.94250107	-118.4079971	99473
LAX	9/1/2022	33.94250107	-118.4079971	145866
LGA	10/1/2022	40.77719879	-73.87259674	72834
OAK	6/1/2022	37.721298	-122.221001	37366
ORD	7/1/2022	41.9786	-87.9048	80350
SFO	6/1/2022	37.61899948	-122.375	76918

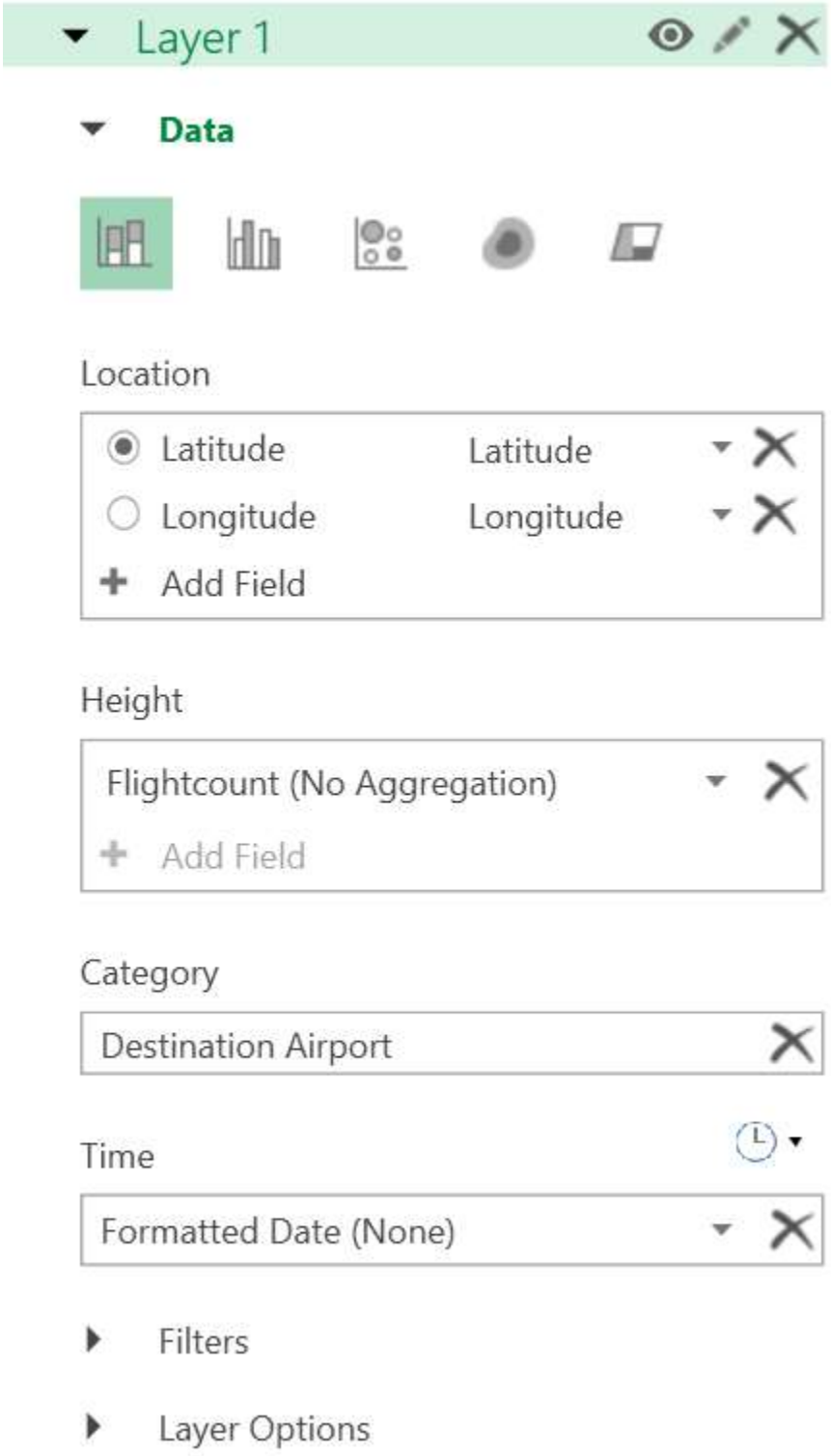
Select all the data. Then, go to “insert” tab to find out the menu “3D Map”.



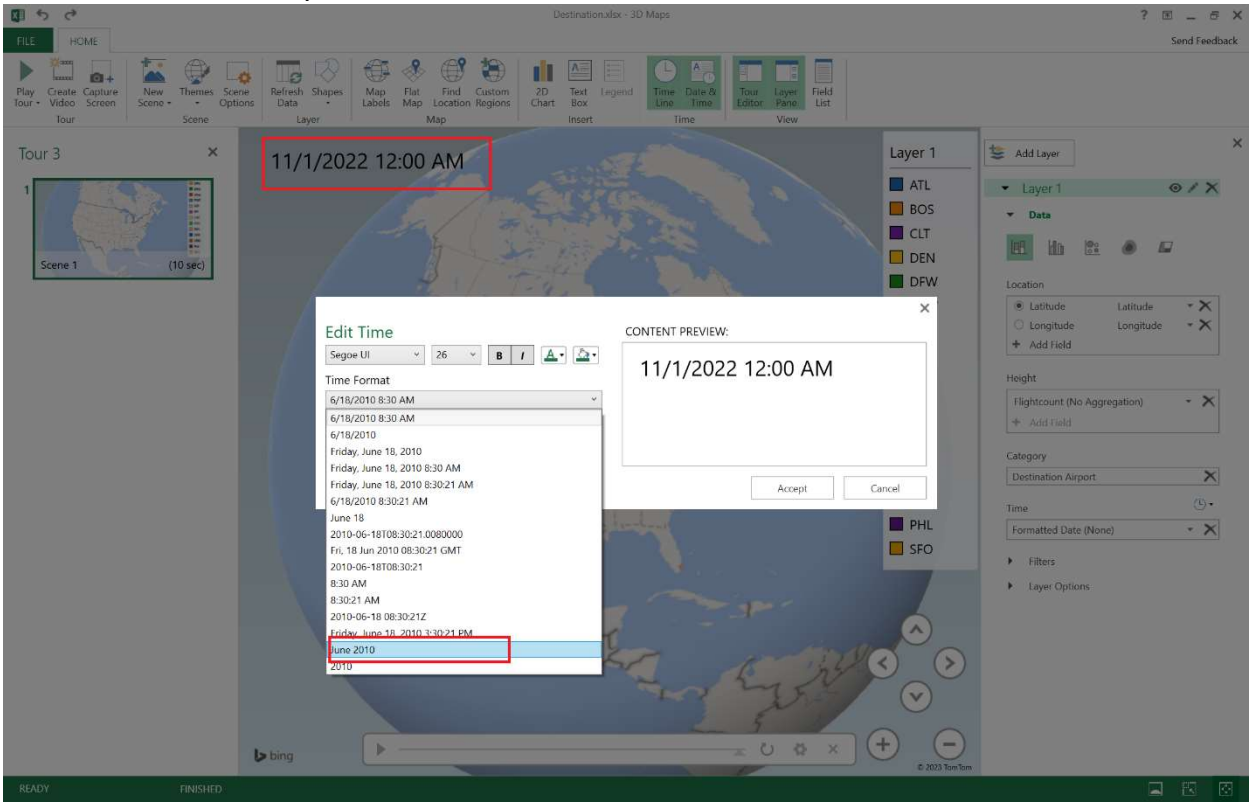
Screen looks like below:

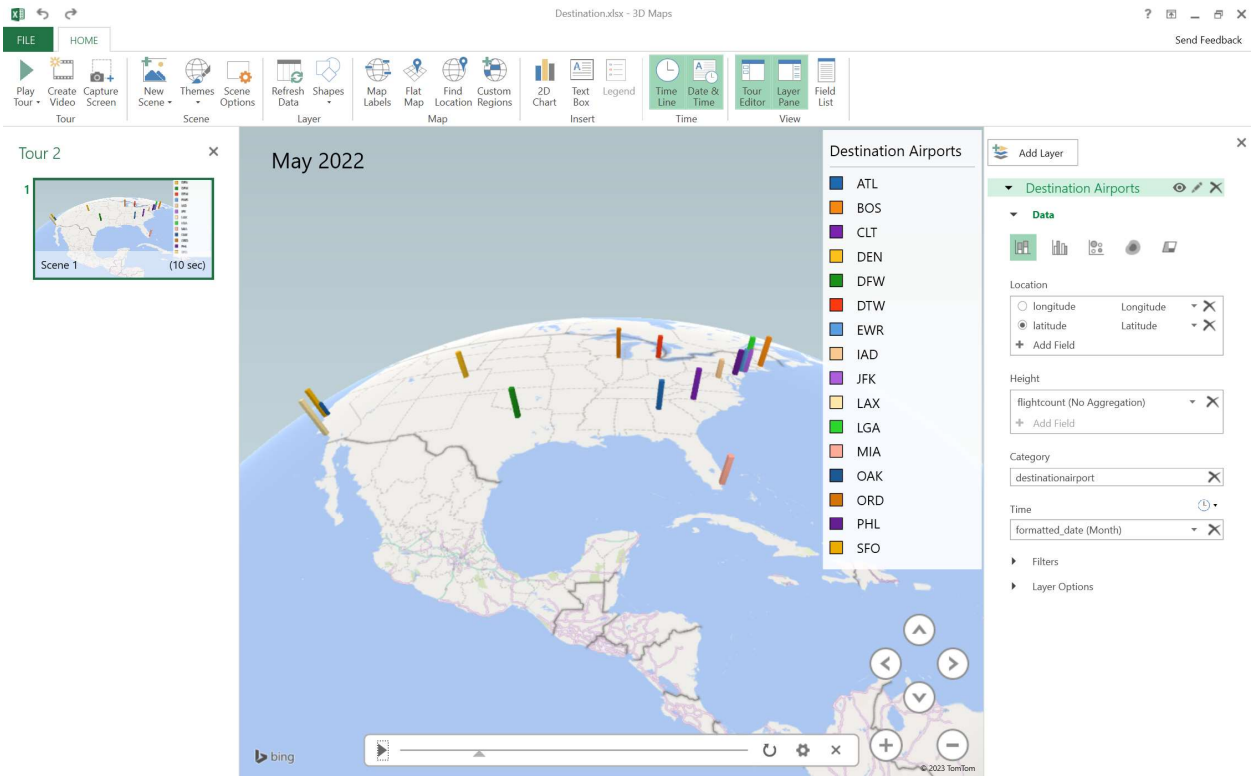


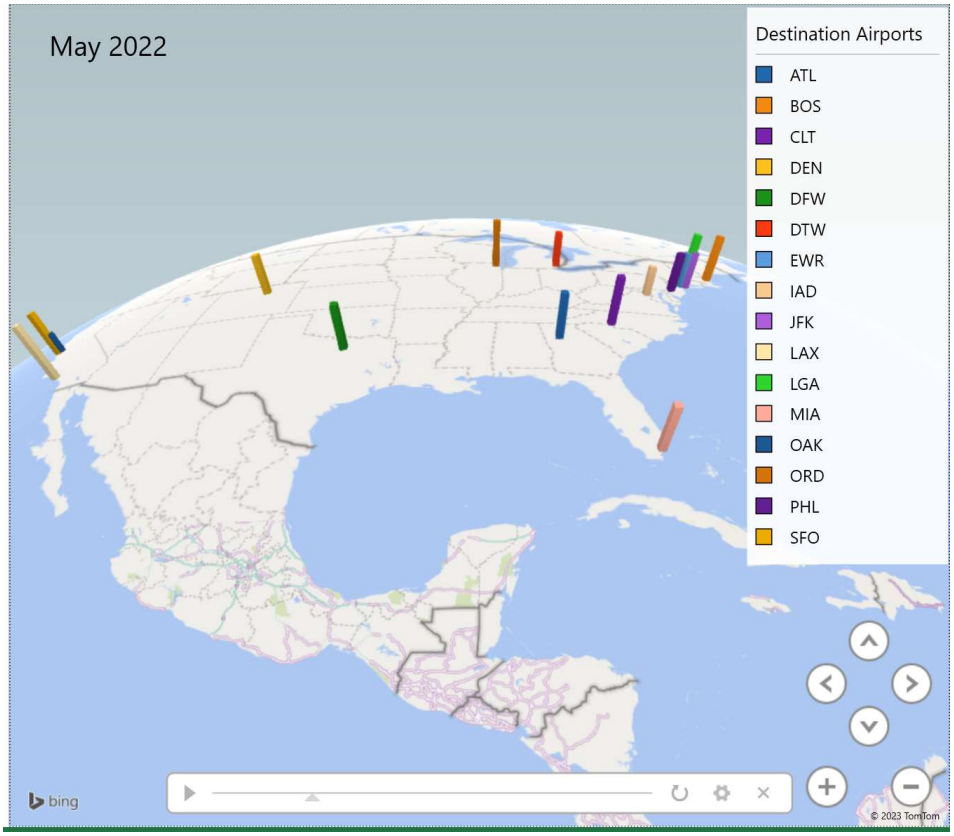
Select Location → Latitude, Height → Flightcount (No Aggregation), Category → Destination Airport, and Time → Formatted_Date (None).
Flightcount is set to "no aggregation" as the total number of tickets purchased (FlightID) has already been aggregated using HiveQL.



In the timeline, month year format has chosen.

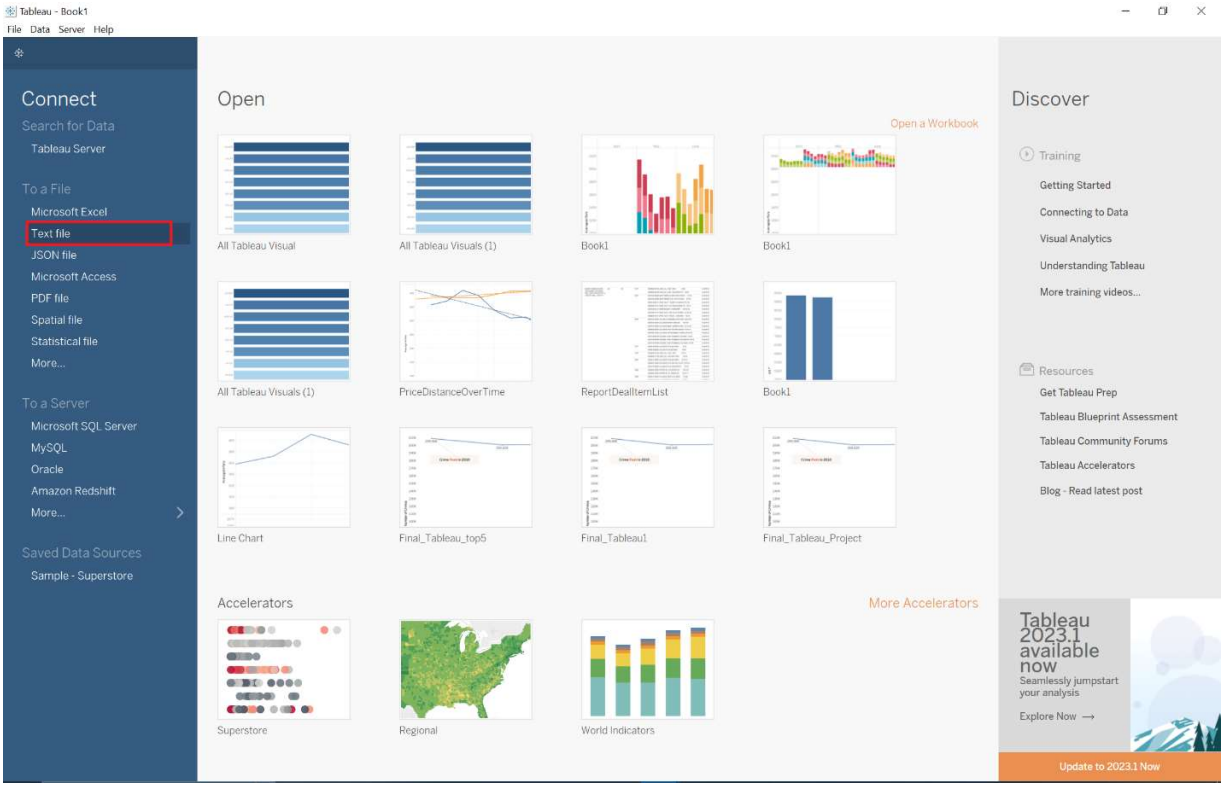




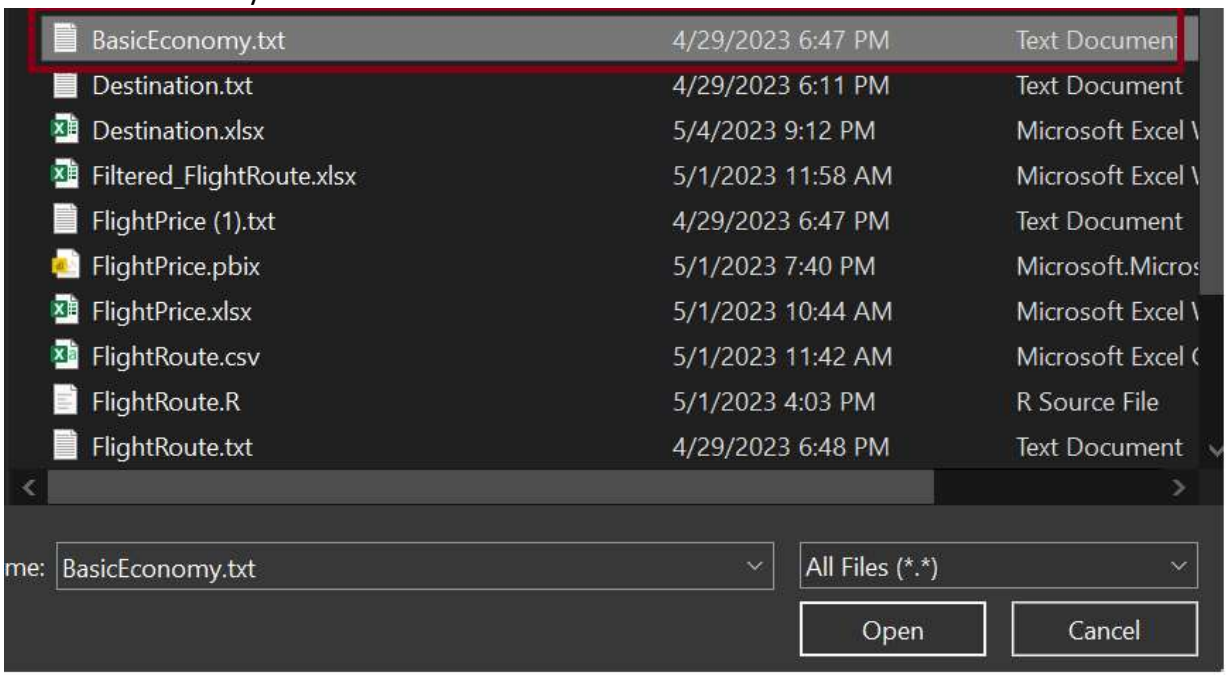


7. Distribution of basic economy tickets (pie chart in Tableau)

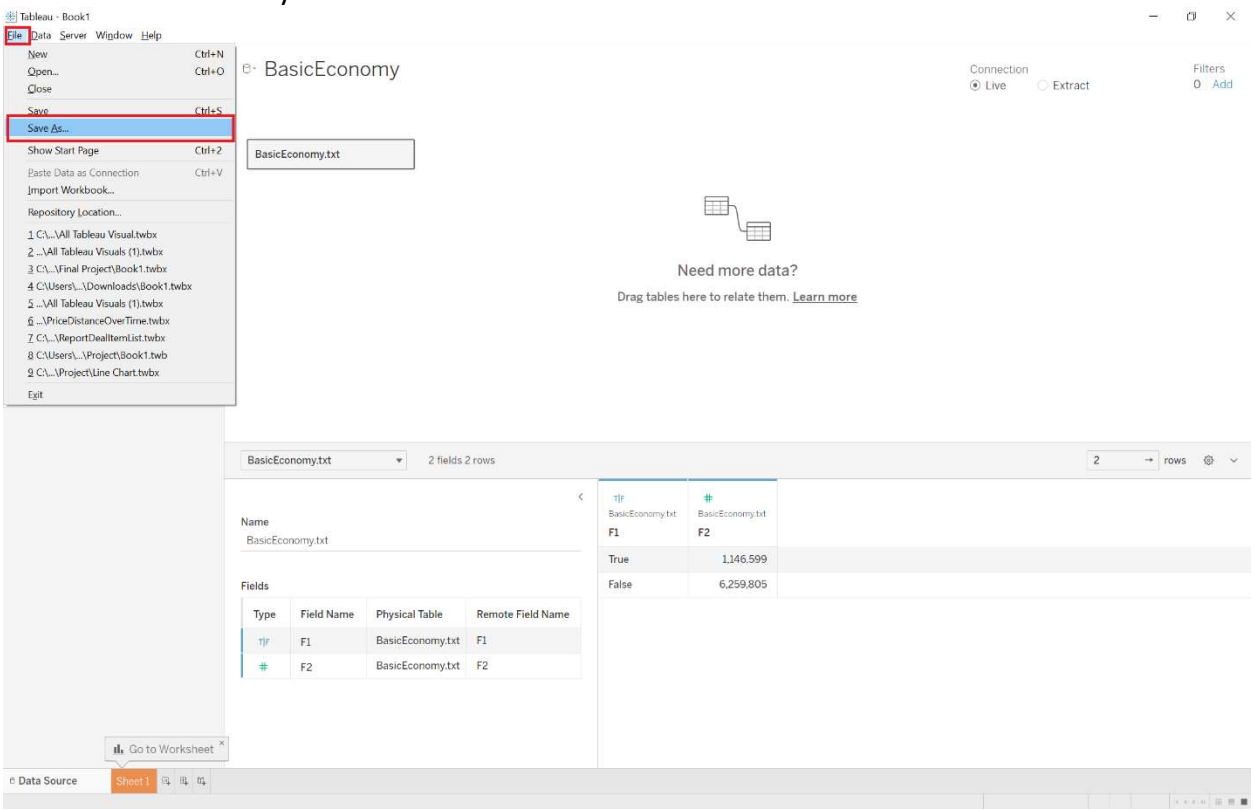
Open Tableau, select **Text file**:



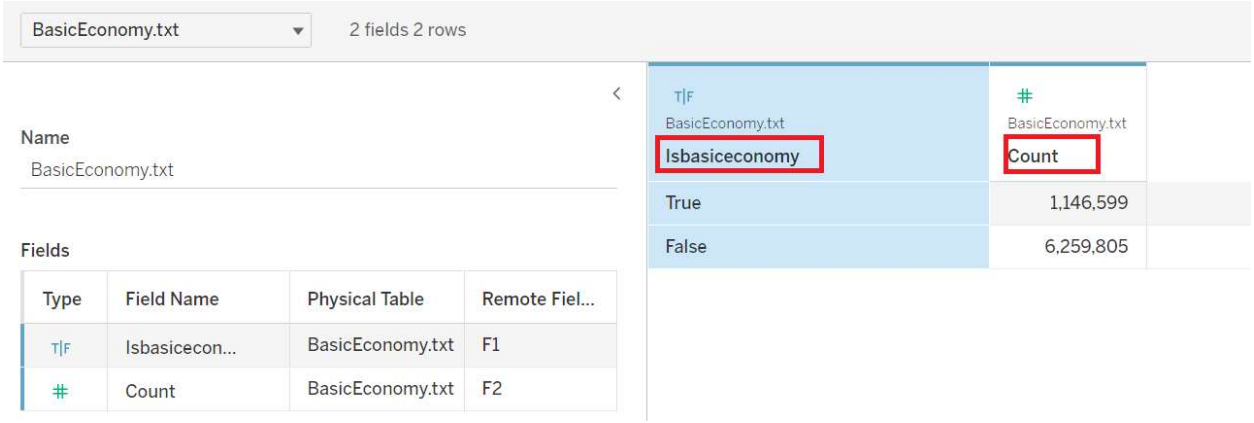
Select BasicEconomy.txt:



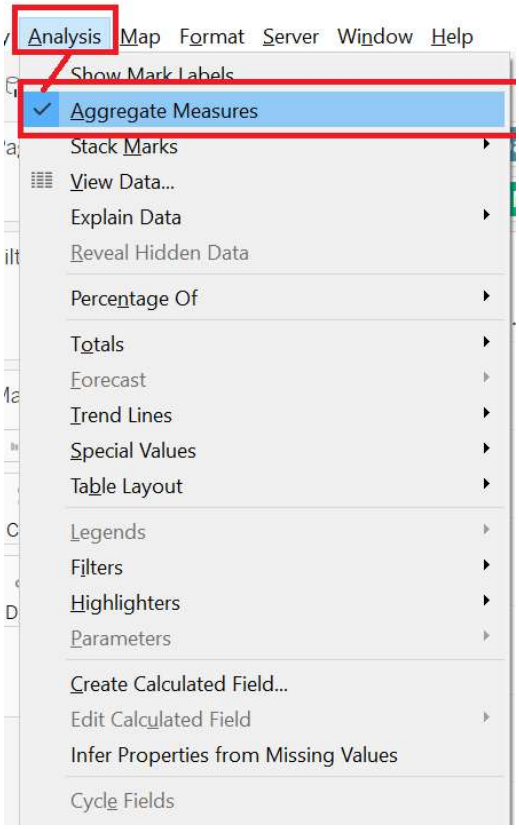
Save As BasicEconomy.twbx :

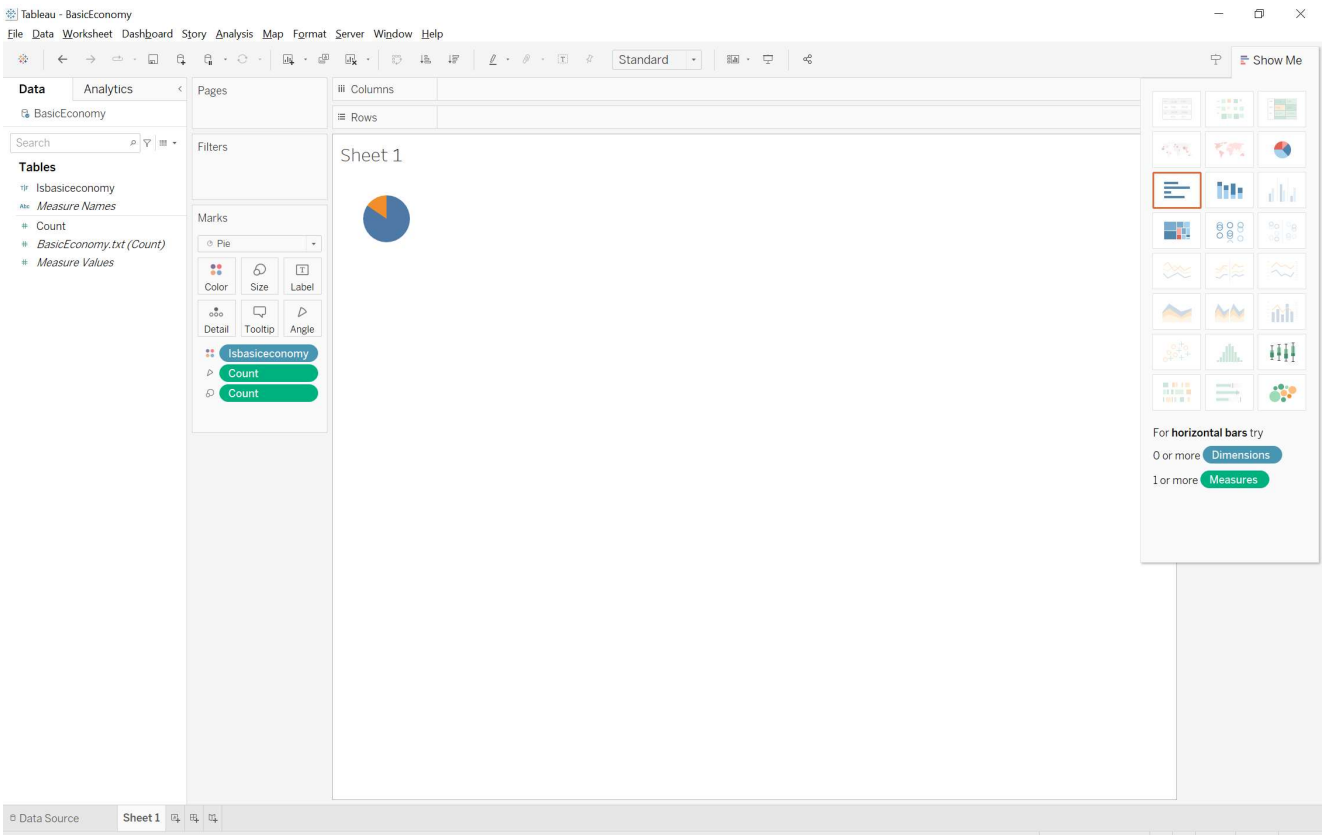


Rename F1 to Isbasiceconomy, and F2 to Count:

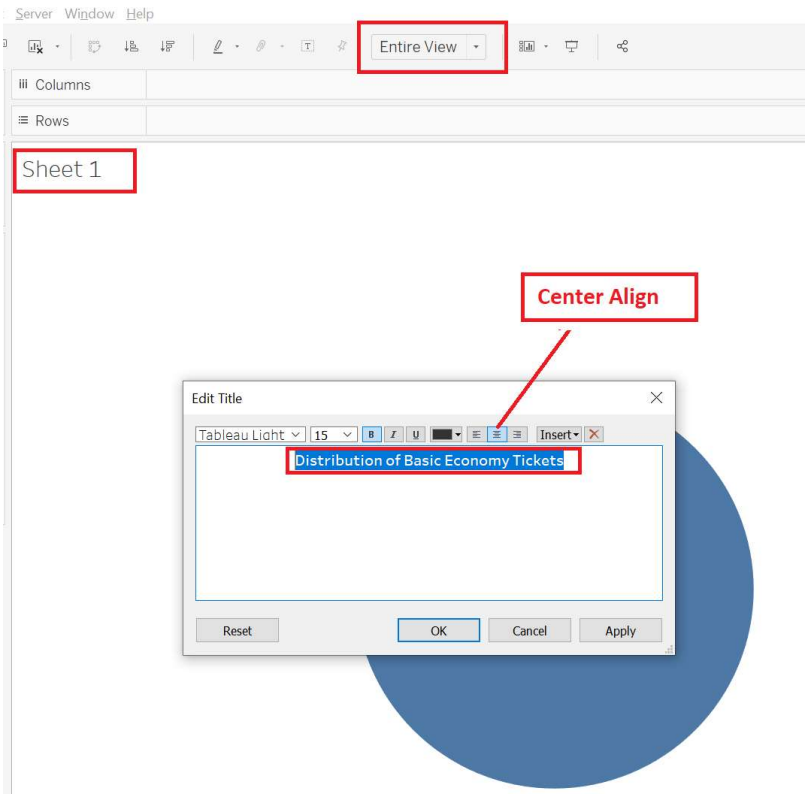


Go to sheet1:
Drag Isbasiceconomy to Columns
Drag Count to Rows
Uncheck Aggregrated Measures because it has already been aggregated using HiveQL.

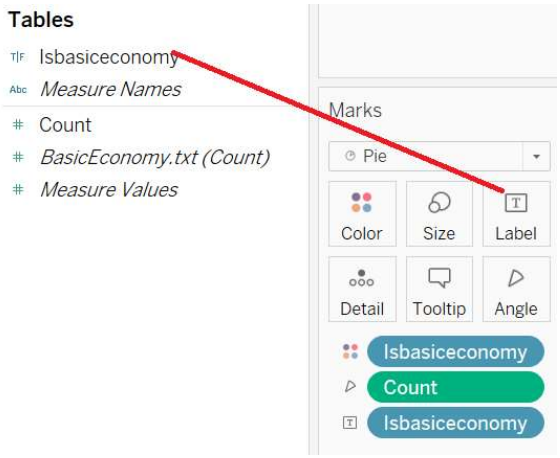




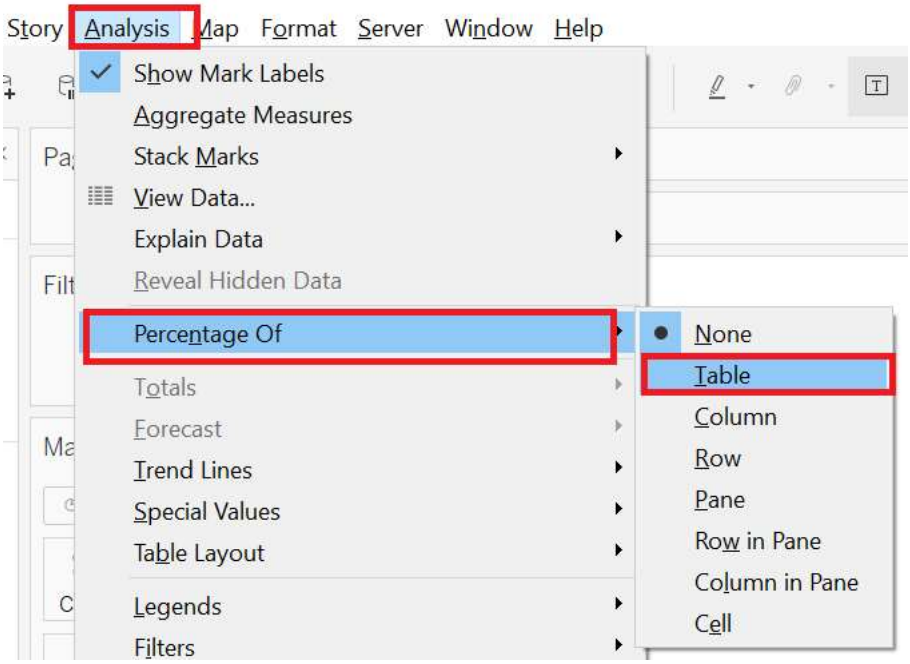
Change to Entire View. Change title sheet 1 to “Distribution of Basic Economy Tickets”:



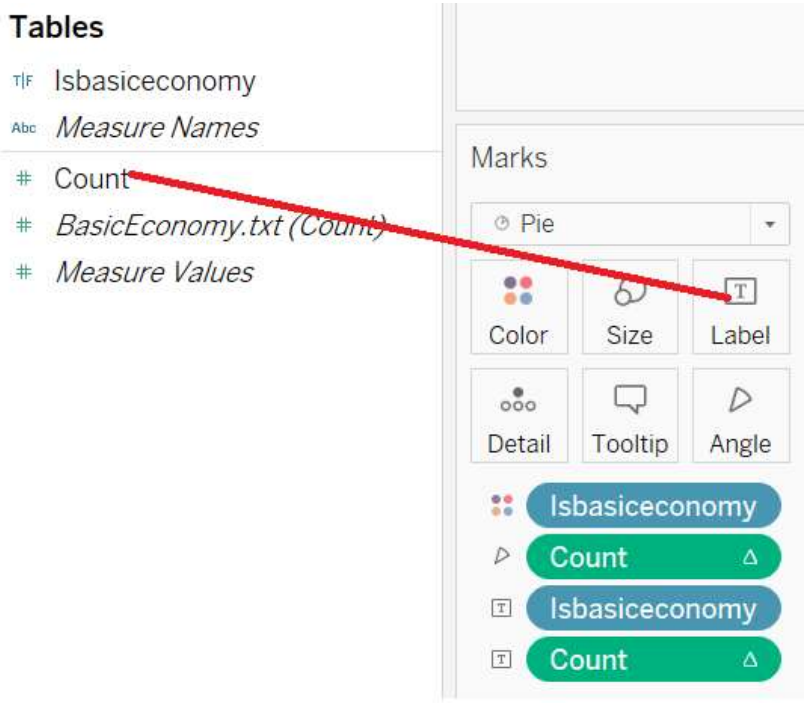
Drage Isbasiceconomy to Label:

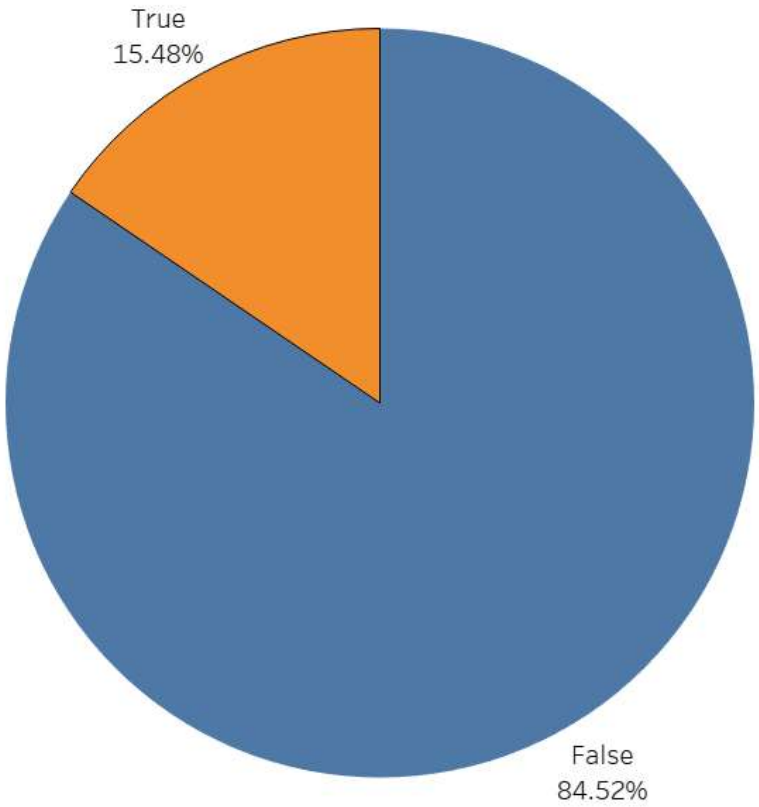


Go to Analysis → Percentage Of → Table:

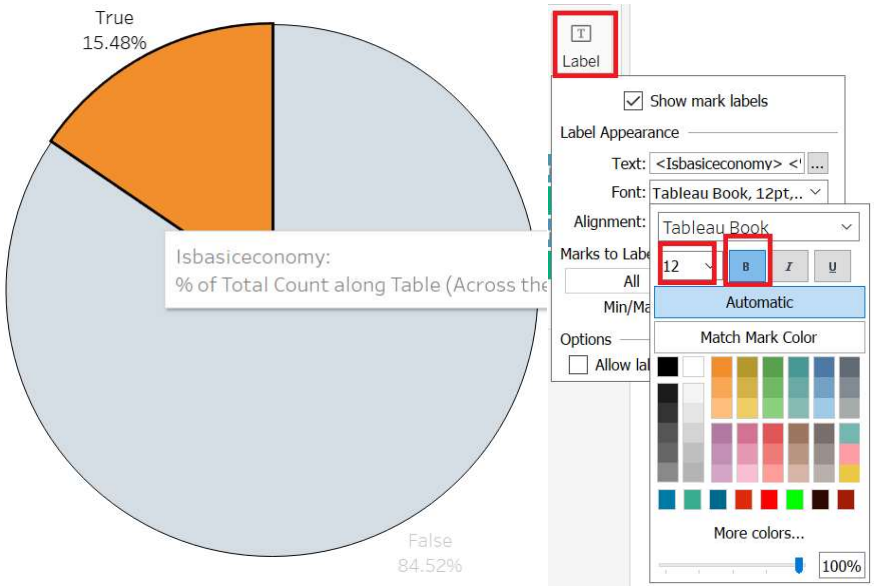


To display percentage: Drag Count to Label:

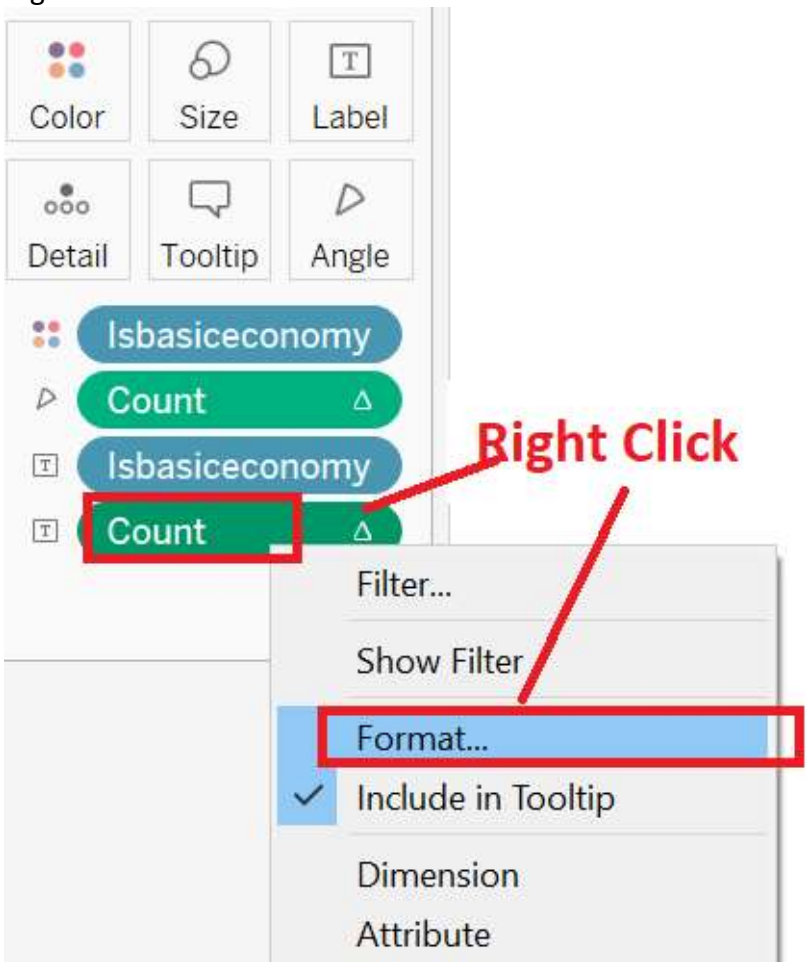


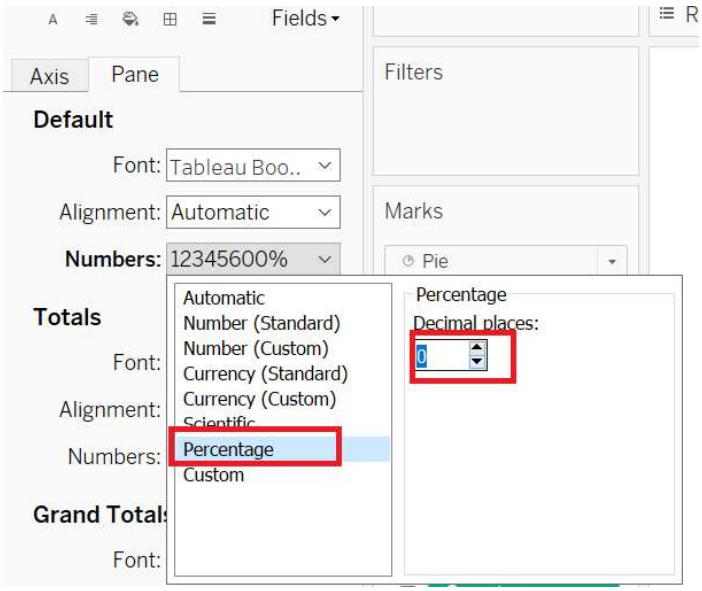


Select and Drag True and percentage inside the pie. Bold the label value:



To display only integer value:




Fields
Axis
Pane
Default
Font: Tableau Boo..
Alignment: Automatic
Numbers: 12345600%
Totals
Font:
Alignment:
Numbers:
Grand Totals
Font:
Filters
Marks
Pie
Automatic
Number (Standard)
Number (Custom)
Currency (Standard)
Currency (Custom)
Scientific
Percentage
Custom
Percentage
Decimal places:
0

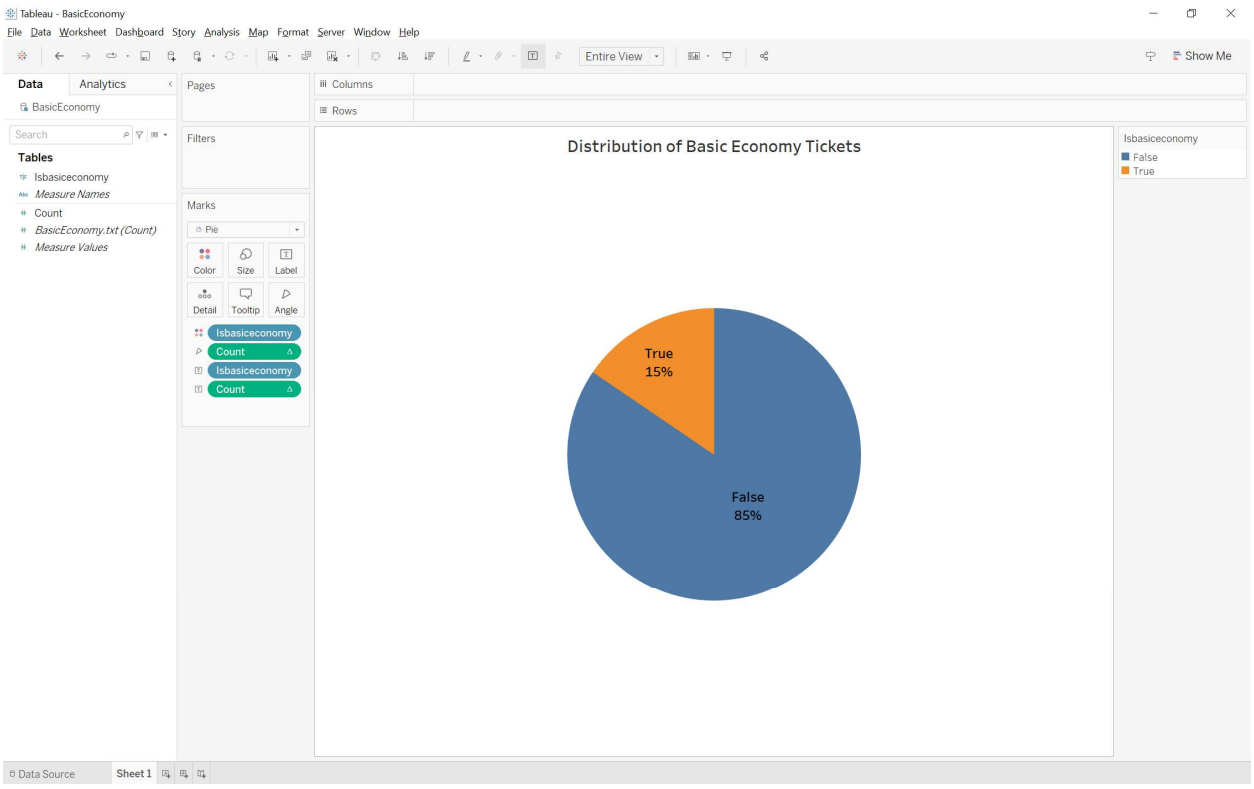
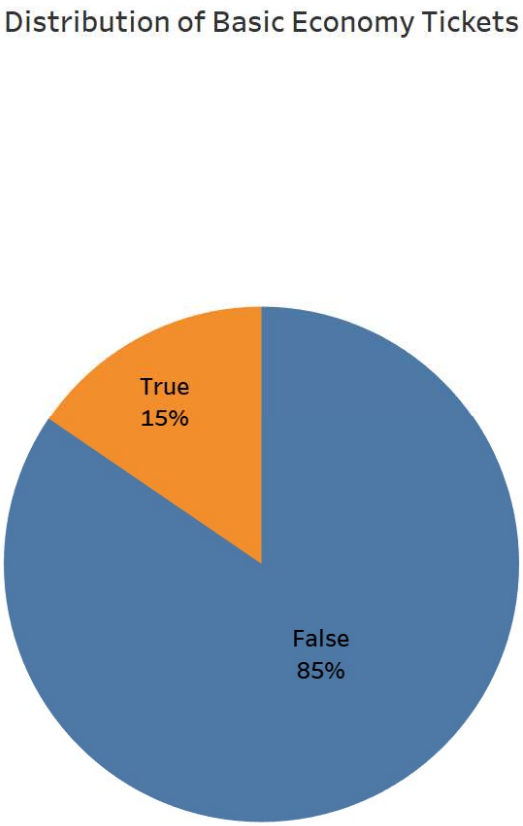


Tableau - BasicEconomy
File Data Worksheet Dashboard Story Analysis Map Format Server Window Help
Entire View
Show Me
Data
Analytics
BasicEconomy
Search
Tables
Isbasiceconomy
Measure Names
Count
BasicEconomy.txt (Count)
Measure Values
Pages
Filters
Marks
Pie
Color
Size
Label
Detail
Tooltip
Angle
Isbasiceconomy
Count
Isbasiceconomy
Count
Columns
Rows
Distribution of Basic Economy Tickets
True
15%
False
85%
Isbasiceconomy
False
True
Data Source
Sheet 1



Conclusion

In this tutorial, we learned how Hadoop Cluster can be used to analysis flight prices using Apache Hive. We went through a flow to understand how the raw data is first uploaded to HDFS, and then loaded to Hive tables for performing queries. Finally, we learned how to import the results of Hive queries and to create visualizations using tableau, Power BI, and 3D Map in Excel.

References

1. URL of Data Source: https://www.kaggle.com/datasets/dilwong/flightprices
https://www.kaggle.com/datasets/mike90/airport-codes
2. GitHub: https://github.com/amach3/Flight-Price-Analysis.git