MIS 6346.503 - Big Data

Group 10: Project Report

Project Members:
Satbir Singh Dhanjal
Mihika Tushar Gupte
Rudra Abhishek

Contents

PROJECT DESCRIPTION:

New York City is one of the most populous cities in the United States, and it has millions of taxi trips taken every month. Our main aim is to explore and analyze the effect of introducing the public transportation means of yellow taxis. We will be studying the impact these means have brought to the overall urban movement in NYC. We will begin by exploring the datasets at our disposal spanning the different NYC boroughs.

Our project aims to understand the current patterns outlined in the data, whether in terms of geographic location, frequency of trips, trip durations, fare amounts, and to process the data to develop insights.

The analysis will benefit decision-makers to properly manage their resources and help taxi companies to maximize their utilization by diverting the cabs into the locations during specific times and indirectly help taxi drivers to make more profits. It would also help passengers get insights on the trip details based on zones, trip fares which would help them better manage their upcoming trips.

DATASET:

The datasets used in the project include the trip records of the biggest group of taxi operators in NYC, the New York Yellow Taxi Trip Data.

The dataset is of size: **7.26 GB and has 84,399,019 records**

To be able to make concrete observations about the taxi usage patterns, we have limited the analysis of the taxi service datasets to a timeframe of a year.

Description of the Dataset used:

- The yellow taxi trip dataset consists of fields capturing pick-up and drop-off dates/times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts.
- The dataset consists of two tables which would be joined to create a database.
- The two tables along with the columns that they consist of are stated below:

Taxi Zones:

Columns	**Description**
LocationID	Unique identifier for each zone
Borough	a town or district which is an administrative unit.
Zone	List of sectors a borough is divided into
Service_zone	Zones to identify the types of taxi service provided

Trip Data:

Columns	Description
VendorID	A code indicating the Taxicab & Livery Passenger Enhancement Programs (TPEP/LPEP) provider that provided the record. ---- 1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
passenger_count	The number of passengers in the vehicle. (This is a driver-entered value)
trip_distance	The elapsed trip distance in miles reported by the taximeter.
RatecodeID	The final rate code in effect at the end of the trip. *RateCodeID: The final rate code in effect at the end of the trip. ---- 1= Standard rate ---- 2=JFK ---- 3=Newark ---- 4=Nassau or Westchester ---- 5=Negotiated fare ---- 6=Group ride
store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server. ---- Y= store and forward trip ---- N= not a store and forward trip
PULocationID	NYC Taxi and Limousine Commission (TLC) Taxi Zone in which the taximeter was engaged
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged.
payment_type	A numeric code signifying how the passenger paid for the trip. ---- 1= Credit card ---- 2= Cash ---- 3= No charge ---- 4= Dispute ---- 5= Unknown ---- 6= Voided trip
fare_amount	The time-and-distance fare calculated by the meter.
extra	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
mta_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
tip_amount	This field is automatically populated for credit card tips. Cash tips are not included.
tolls_amount	Total amount of all tolls paid in trip.

improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
total_amount	The total amount charged to passengers. Does not include cash tips.
congestion_surcharge	Increase in fair price caused by various parameters.

SCREENSHOTS OF DATA SOURCES:

Figure 1: Taxi zone csv

	LocationID	Borough	Zone	service_zone
2	1	EWR	Newark Airport	EWR
3	2	Queens	Jamaica Bay	Boro Zone
4	3	Bronx	Allerton/Pelham Gardens	Boro Zone
5	4	Manhattan	Alphabet City	Yellow Zone
6	5	Staten Island	Arden Heights	Boro Zone
7	6	Staten Island	Arrochar/Fort Wadsworth	Boro Zone
8	7	Queens	Astoria	Boro Zone
9	8	Queens	Astoria Park	Boro Zone
10	9	Queens	Auburndale	Boro Zone
11	10	Queens	Baisley Park	Boro Zone
12	11	Brooklyn	Bath Beach	Boro Zone
13	12	Manhattan	Battery Park	Yellow Zone
14	13	Manhattan	Battery Park City	Yellow Zone
15	14	Brooklyn	Bay Ridge	Boro Zone
16	15	Queens	Bay Terrace/Fort Totten	Boro Zone
17	16	Queens	Bayside	Boro Zone
18	17	Brooklyn	Bedford	Boro Zone
19	18	Bronx	Bedford Park	Boro Zone
20	19	Queens	Bellerose	Boro Zone
21	20	Bronx	Belmont	Boro Zone
22	21	Brooklyn	Bensonhurst East	Boro Zone
23	22	Brooklyn	Bensonhurst West	Boro Zone
24	23	Staten Island	Bloomfield/Emerson Hill	Boro Zone
25	24	Manhattan	Bloomingdale	Yellow Zone
26	25	Brooklyn	Boerum Hill	Boro Zone

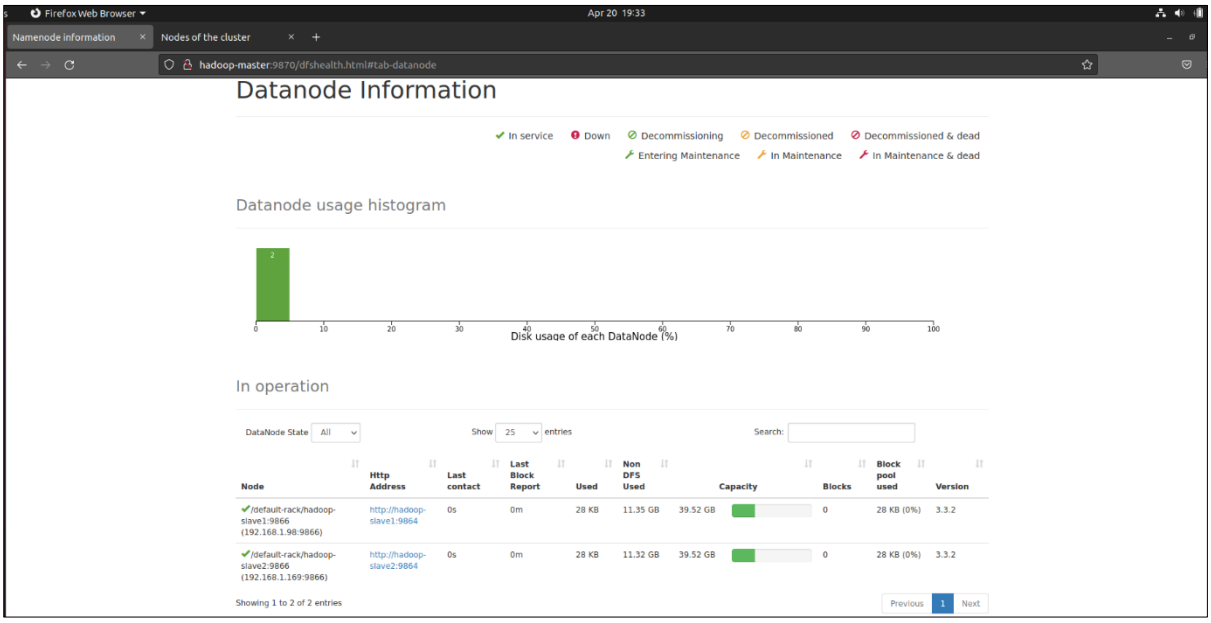
Figure 2: Yellow Taxi Trip Data csv

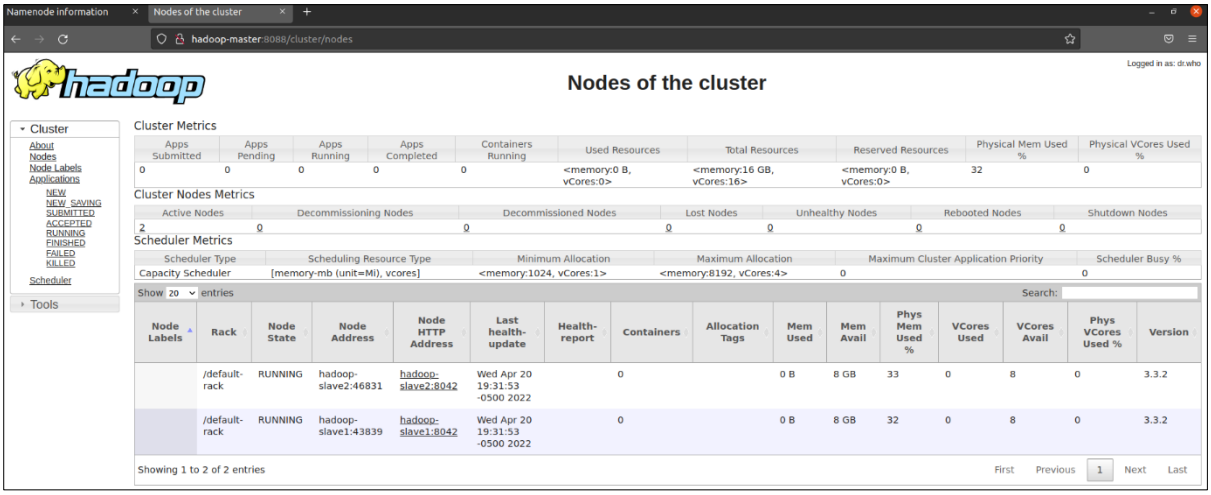
	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	RatecodeID	store_and_fwd_flag	PULocationID	DOLocationID	payment_type	fare_amount	extra	mta_tax	tip_amount	tolls_amount	improvement_surcharge	total_amount	congestion_surcharge
2	1	4/1/2019 0:04	4/1/2019 0:06	1	0.5	1	N	239	239	1	4	3	0.5	1	0	0.3	8.8	2.5
3	1	4/1/2019 0:22	4/1/2019 0:25	1	0.7	1	N	230	100	2	4.5	3	0.5	0	0	0.3	8.3	2.5
4	1	4/1/2019 0:39	4/1/2019 1:19	1	10.9	1	N	68	127	1	36	3	0.5	7.95	0	0.3	47.75	2.5
5	1	4/1/2019 0:35	4/1/2019 0:37	1	0.2	1	N	68	68	2	3.5	3	0.5	0	0	0.3	7.3	2.5
6	1	4/1/2019 0:44	4/1/2019 0:57	1	4.8	1	N	50	42	1	15.5	3	0.5	3.85	0	0.3	23.15	2.5
7	1	4/1/2019 0:29	4/1/2019 0:38	1	1.7	1	N	95	196	2	8.5	0.5	0.5	0	0	0.3	9.8	0
8	1	4/1/2019 0:06	4/1/2019 0:08	1	0	1	N	211	211	3	3	3	0.5	0	0	0.3	6.8	2.5
9	1	4/1/2019 0:52	4/1/2019 0:55	1	0.2	1	N	237	162	1	4	3	0.5	0	0	0.3	7.8	2.5
10	2	4/1/2019 0:52	4/1/2019 1:11	1	4.15	1	N	148	37	2	16.5	0.5	0.5	0	0	0.3	20.3	2.5
11	1	4/1/2019 0:02	4/1/2019 0:03	1	0	5	N	265	265	2	0.01	0	0	0	0	0.3	0.31	0
12	1	4/1/2019 0:03	4/1/2019 0:03	1	0	5	N	265	265	1	200	0	0	40.05	0	0.3	240.35	0
13	1	4/1/2019 0:13	4/1/2019 0:20	1	1.3	1	N	237	142	2	6.5	3	0.5	0	0	0.3	10.3	2.5
14	4	4/1/2019 0:25	4/1/2019 0:55	1	10.06	1	N	249	69	2	32.5	0.5	0.5	0	0	0.3	36.3	2.5
15	2	4/1/2019 0:14	4/1/2019 0:42	1	18.5	4	N	132	265	1	65.5	0.5	0.5	13.36	0	0.3	80.16	0
16	1	4/1/2019 0:13	4/1/2019 0:22	1	3.3	1	N	107	263	2	11	3	0.5	0	0	0.3	14.8	2.5
17	1	4/1/2019 0:24	4/1/2019 0:31	1	0.9	1	N	107	114	1	6	3	0.5	1.5	0	0.3	11.3	2.5
18	1	4/1/2019 0:43	4/1/2019 0:50	1	1.5	1	N	233	140	1	7.5	3	0.5	2.26	0	0.3	13.56	2.5
19	2	4/1/2019 0:10	4/1/2019 0:21	2	1.89	1	N	114	224	1	9.5	0.5	0.5	2.66	0	0.3	15.96	2.5
20	2	4/1/2019 0:38	4/1/2019 0:49	2	3.39	1	N	237	42	2	11.5	0.5	0.5	0	0	0.3	15.3	2.5
21	1	4/1/2019 0:03	4/1/2019 0:10	1	0.8	1	N	249	158	1	6	3	0.5	1.95	0	0.3	11.75	2.5
22	1	4/1/2019 0:22	4/1/2019 0:41	1	5.3	1	N	186	202	2	18	3	0.5	0	0	0.3	21.8	2.5
23	1	4/1/2019 0:07	4/1/2019 0:13	1	1.5	1	N	25	181	1	7	0.5	0.5	1	0	0.3	9.3	0
24	1	4/1/2019 0:58	4/1/2019 1:02	1	0.9	1	N	48	170	2	5	3	0.5	0	0	0.3	8.8	2.5
25	1	4/1/2019 0:22	4/1/2019 0:43	1	4	1	N	148	225	1	17.5	3	0.5	4.25	0	0.3	25.55	2.5
26	1	4/1/2019 0:59	4/1/2019 1:11	1	3.2	1	N	140	74	2	12	3	0.5	0	0	0.3	15.8	2.5
27	2	4/1/2019 0:41	4/1/2019 0:52	2	4.08	1	N	254	259	1	14	0.5	0.5	0	0	0.3	15.3	0
28	1	4/1/2019 0:51	4/1/2019 0:53	1	0.3	1	N	186	68	1	3.5	3	0.5	3	0	0.3	10.3	2.5
29	1	4/1/2019 0:11	4/1/2019 0:21	1	2.5	1	N	48	239	1	10.5	3	0.5	2.85	0	0.3	17.15	2.5
30	1	4/1/2019 0:24	4/1/2019 0:34	1	2.4	1	N	239	41	1	10	3	0.5	2.75	0	0.3	16.55	2.5
31	1	4/1/2019 0:03	4/1/2019 0:16	1	6.3	1	N	132	216	2	19	0.5	0.5	0	0	0.3	20.3	0
32	2	4/1/2019 0:21	4/1/2019 0:23	1	0.41	1	N	151	24	2	3.5	0.5	0.5	0	0	0.3	4.8	0
33	2	4/1/2019 0:39	4/1/2019 1:07	5	19.41	2	N	132	107	1	52	0	0.5	11.06	0	0.3	66.36	2.5
34	1	4/1/2019 0:11	4/1/2019 0:18	3	2.4	1	N	234	229	2	8.5	3	0.5	0	0	0.3	12.3	2.5
35	1	4/1/2019 0:38	4/1/2019 0:41	1	0.9	1	N	239	236	1	5	3	0.5	1	0	0.3	9.8	2.5
36	1	4/1/2019 0:02	4/1/2019 0:12	1	1.6	1	N	129	82	2	8.5	0.5	0.5	0	0	0.3	9.8	0
37	1	4/1/2019 0:22	4/1/2019 0:34	1	3.1	1	N	239	42	2	11.5	3	0.5	0	0	0.3	15.3	2.5
38	2	4/1/2019 0:07	4/1/2019 0:31	1	16.83	2	N	132	170	1	52	0	0.5	12.21	5.76	0.3	73.27	2.5

BIG-DATA INFRASTRUCTURE SETUP:

For the project we setup a **Multi Node Cluster**. Our Multi Node Cluster in Hadoop contains two Data Nodes in a distributed Hadoop environment (namely hadoop-slave1 and hadoop-slave2)
This setup was used by us as the dataset that we chose had a total of 84,399,019 records.

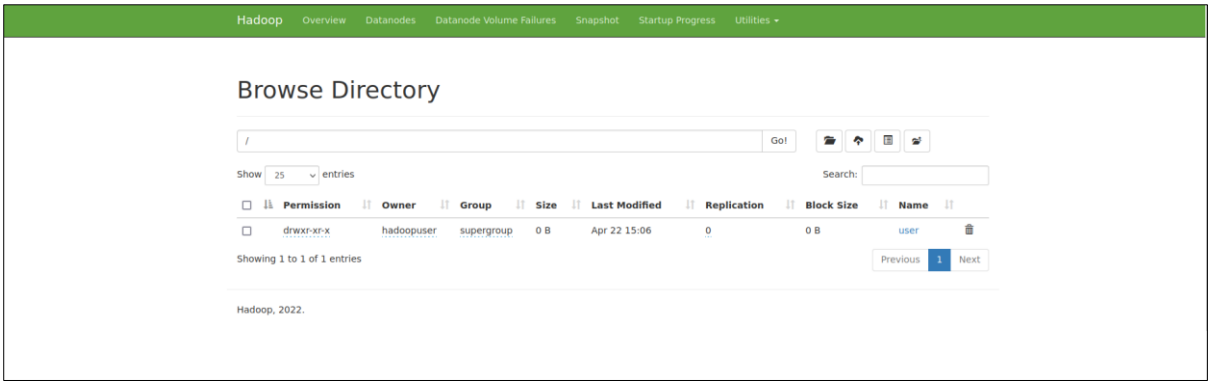
Below screenshots represent the status of the data nodes:

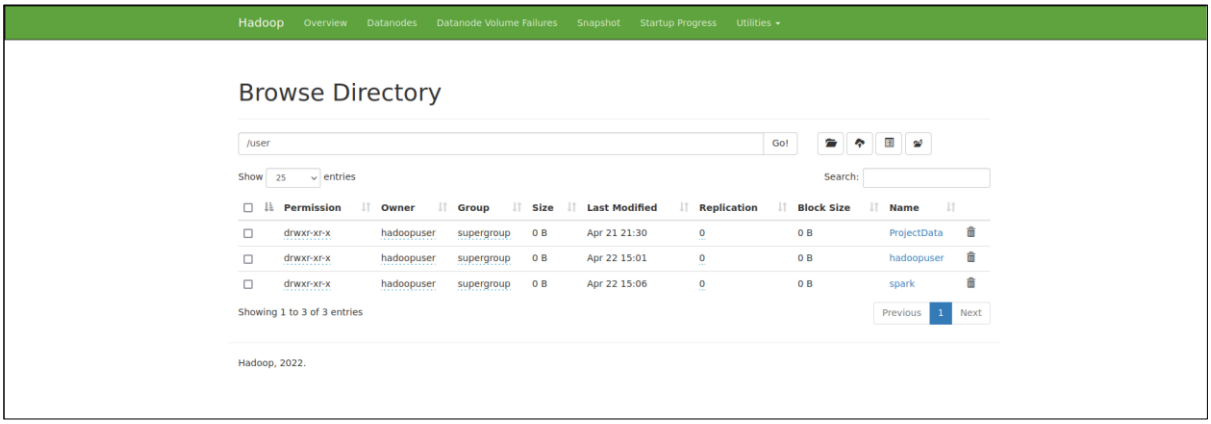




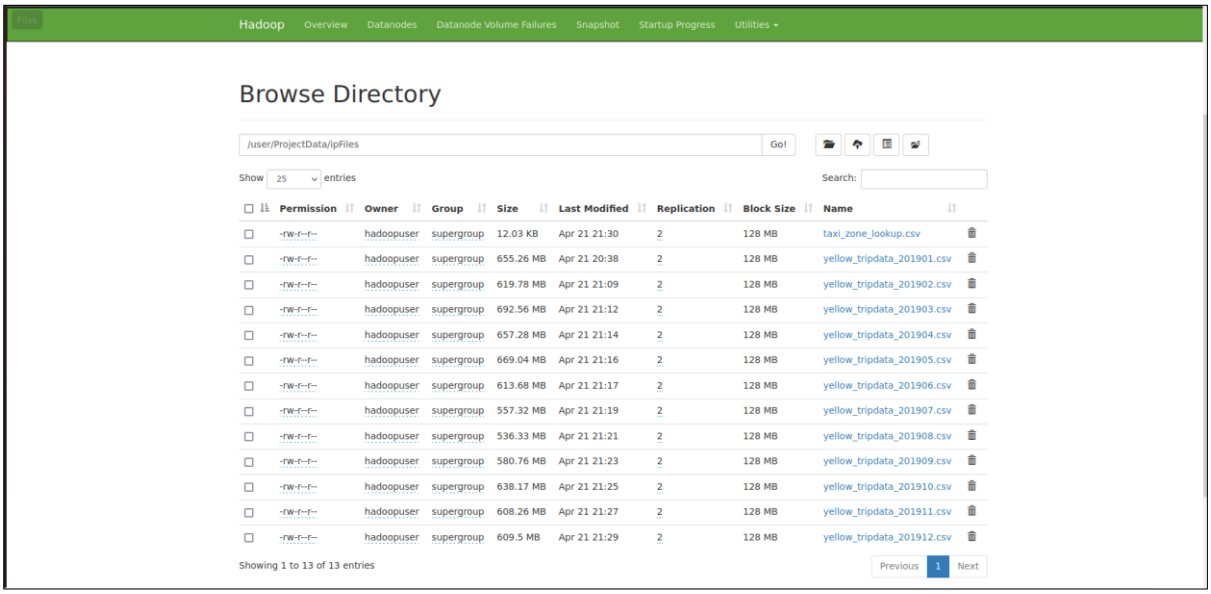
- After setting-up the Hadoop infrastructure, we copied all the data from the csv's that we had at our disposal to HDFS

--Creation of Hadoop Directories:





--Loading the csv files into the file path



- We setup the Hive infrastructure and queried to generate business insights

1. How many Total Trips were taken per Month?

```
hive> select month(y.tpep_pickup_datetime) as month, count(y.trip_distance) as number_of_trips from newyorktaxi.yellow_tripdata y group by mon
th(y.tpep_pickup_datetime);
```

month	number_of_trips
1	7668135
2	7019277
3	7832348
4	7433160
5	7565372
6	6940776
7	6310349
8	6073055
9	6567604
10	7214095
11	6877809
12	6896721

-Based on the number of trips that happen each month the taxi company must decide of increasing the number of taxis and drivers.

2. Tips given to drivers as per zones:

```
hive> select count(y.tip_amount) as number_of_tips, zone from newyorktaxi.yellow_tripdata y, newyorktaxi.zone_lookup z where y.pulocationid=z.
locationid group by z.zone;
```

number_of_tips	zone
141487	"Alphabet City"
3333	"Claremont/Bathgate"
1699566	"Garment District"
2774	"Howard Beach"
3001	"Laurelton"
2214434	"Midtown North"
1926439	"Midtown South"
19110	"Mott Haven/Port Morris"
6772	"Soundview/Castle Hill"
4112	"Starrett City"
3274283	"Upper East Side North"
3621709	"Upper East Side South"
118262	"Astoria"
1713	"College Point"
177	"Crotona Park"
2054	"Glendale"
1699	"Hollis"
3813	"Mount Hope"
48409	"Williamsburg (South Side)"
7774	"Bay Ridge"
555	"Forest Park/Highland Park"
11521	"Jamaica"
5731	"Kew Gardens"
3948	"Kingsbridge Heights"
5758	"Ocean Hill"

--- The zones in the Upper East Side (South and North) are the ones that had drivers that received more tips.

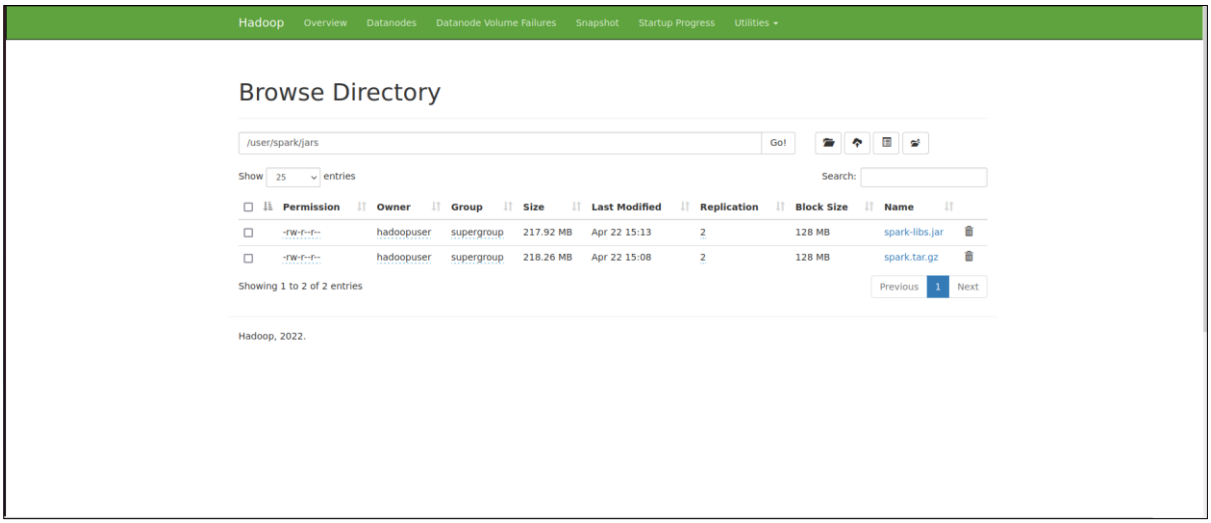
3. Maximum Tips given to drivers as per zones:

```
hive> select max(y.tip_amount) as Highest_tip_amount, zone from newyorktaxi.yellow_tripdata y, newyorktaxi.zone_lookup z where y.pulocationid=
z.locationid group by z.zone;
highest_tip_amount      zone
9.99    "Alphabet City"
9.84    "Claremont/Bathgate"
9.99    "Garment District"
9.94    "Howard Beach"
8.57    "Laurelton"
9.99    "Midtown North"
9.99    "Midtown South"
9.99    "Mott Haven/Port Morris"
9.73    "Soundview/Castle Hill"
9.00    "Starrett City"
9.99    "Upper East Side North"
9.99    "Upper East Side South"
9.99    "Astoria"
9.68    "College Point"
8.08    "Crotona Park"
9.35    "Glendale"
9.05    "Hollis"
9.95    "Mount Hope"
9.99    "Williamsburg (South Side)"
9.98    "Bay Ridge"
9.95    "Forest Park/Highland Park"
9.99    "Jamaica"
9.98    "Kew Gardens"
9.65    "Kingsbridge Heights"
9.95    "Ocean Hill"
9.80    "Ocean Parkway South"
9.96    "Ozone Park"
5.06    "West Brighton"
9.99    "Yorkville East"
9.99    "Yorkville West"
```

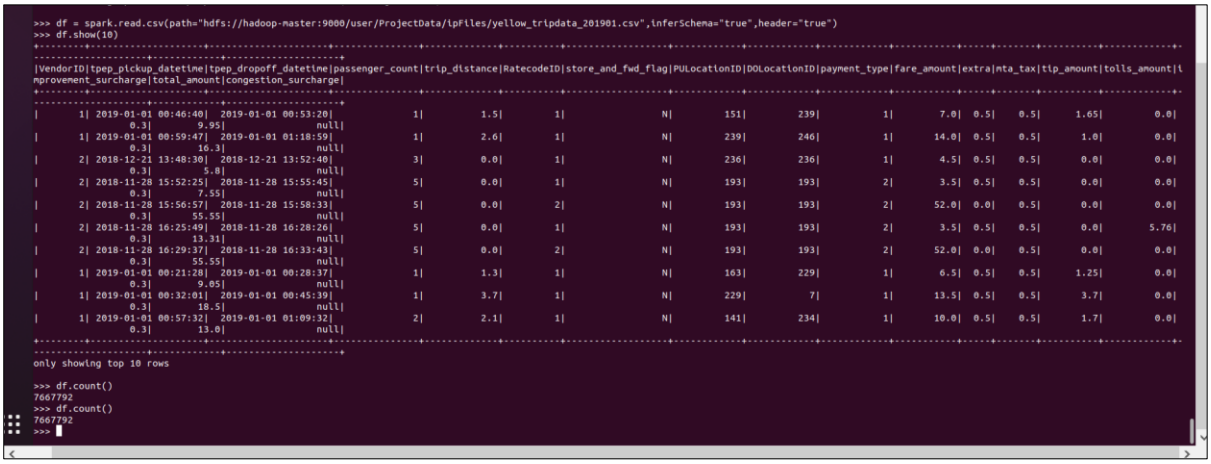
--We observed that the zone West Brighton was an only zone where the passengers tend to give lesser tips as compared to all the other zones.

- Now we went ahead and setup the Spark infrastructure

--Spark Directories:



-- Reading a single file for 2019 and displaying the count for the same (January 2019)

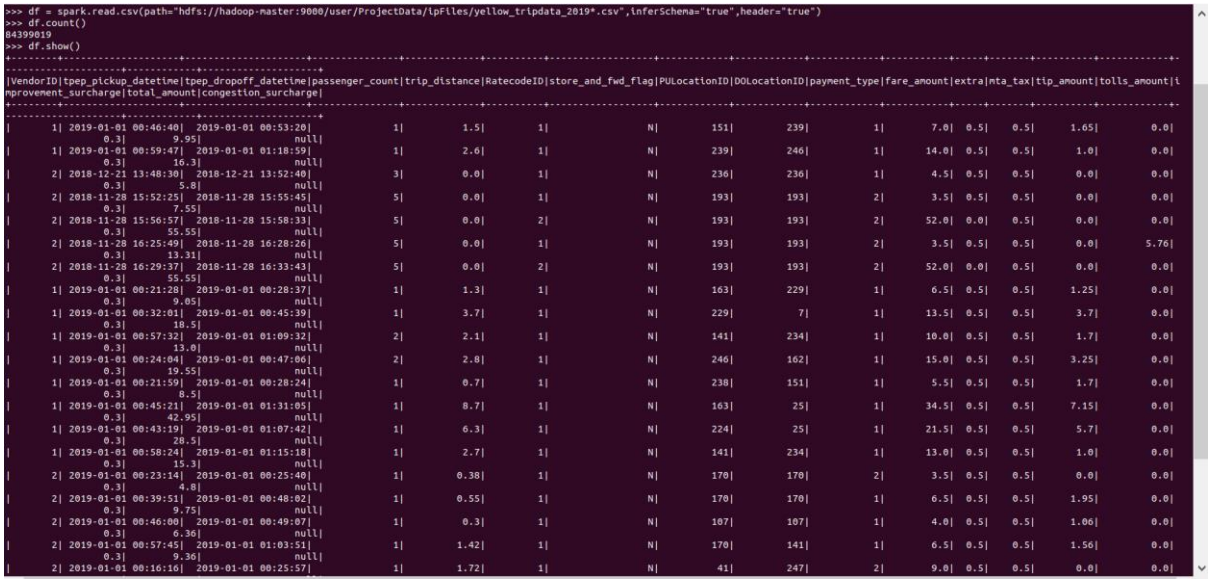


```
>>> df = spark.read.csv(path="hdfs://hadoop-master:9000/user/ProjectData/ipFiles/yellow_tripdata_201901.csv",inferSchema="true",header="true")
>>> df.show(10)
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----+-------+----------+-----------+--------------------+------------+
|VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|passenger_count|trip_distance|RatecodeID|store_and_fwd_flag|PULocationID|DOLocationID|payment_type|fare_amount|extra|mta_tax|tip_amount|tolls_amount|improvement_surcharge|total_amount|congestion_surcharge|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----+-------+----------+-----------+--------------------+------------+
|       1| 2019-01-01 00:46:40| 2019-01-01 00:53:20|              1|          1.5|         1|                 N|         151|         239|           1|        7.0|  0.5|    0.5|      1.65|         0.0|                 0.3|        9.95|                null|
|       1| 2019-01-01 00:59:47| 2019-01-01 01:18:59|              1|          2.6|         1|                 N|         239|         246|           1|       14.0|  0.5|    0.5|       1.0|         0.0|                 0.3|        16.3|                null|
|       2| 2018-12-21 13:48:30| 2018-12-21 13:52:40|              3|          0.0|         1|                 N|         236|         236|           1|        4.5|  0.5|    0.5|       0.0|         0.0|                 0.3|         5.8|                null|
|       2| 2018-11-28 15:52:25| 2018-11-28 15:55:45|              5|          0.0|         1|                 N|         193|         193|           2|        3.5|  0.5|    0.5|       0.0|         0.0|                 0.3|        7.55|                null|
|       2| 2018-11-28 15:56:57| 2018-11-28 15:58:33|              5|          0.0|         2|                 N|         193|         193|           2|       52.0|  0.0|    0.5|       0.0|         0.0|                 0.3|       55.55|                null|
|       2| 2018-11-28 16:25:49| 2018-11-28 16:28:26|              5|          0.0|         1|                 N|         193|         193|           2|        3.5|  0.5|    0.5|       0.0|        5.76|                 0.3|       13.31|                null|
|       2| 2018-11-28 16:29:37| 2018-11-28 16:33:43|              5|          0.0|         2|                 N|         193|         193|           2|       52.0|  0.0|    0.5|       0.0|         0.0|                 0.3|       55.55|                null|
|       1| 2019-01-01 00:21:28| 2019-01-01 00:28:37|              1|          1.3|         1|                 N|         163|         229|           1|        6.5|  0.5|    0.5|      1.25|         0.0|                 0.3|        9.05|                null|
|       1| 2019-01-01 00:32:01| 2019-01-01 00:45:39|              1|          3.7|         1|                 N|         229|           7|           1|       13.5|  0.5|    0.5|       3.7|         0.0|                 0.3|        18.5|                null|
|       1| 2019-01-01 00:57:32| 2019-01-01 01:09:32|              2|          2.1|         1|                 N|         141|         234|           1|       10.0|  0.5|    0.5|       1.7|         0.0|                 0.3|        13.0|                null|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----+-------+----------+-----------+--------------------+------------+
only showing top 10 rows

>>> df.count()
7667792
>>> df.count()
7667792
>>>
```

-- Count for the entire dataset

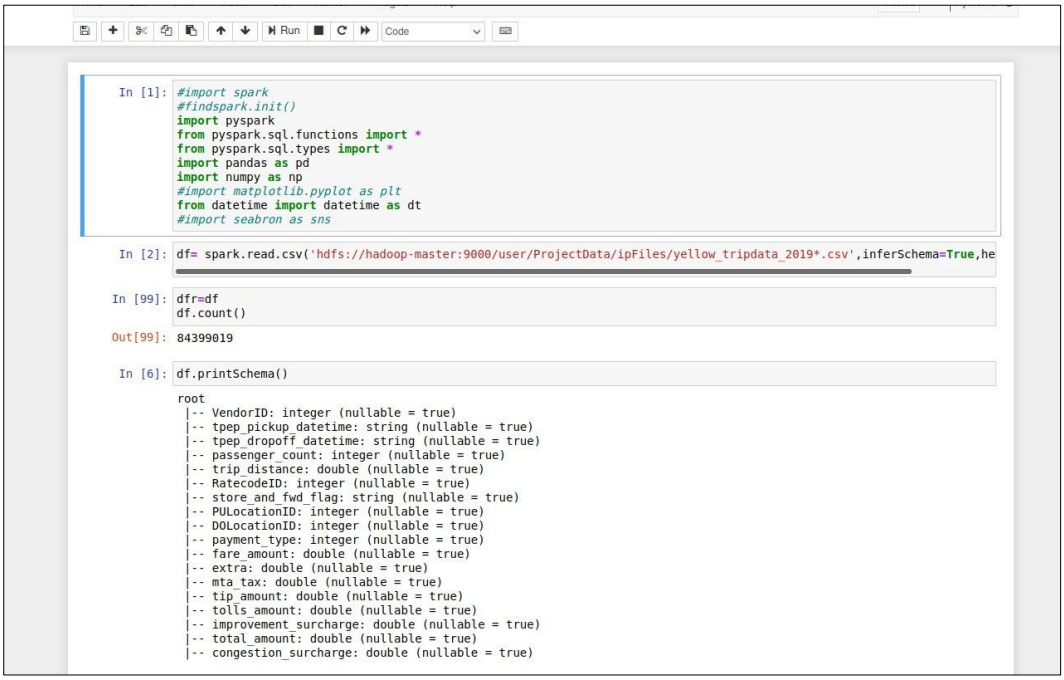
```
>>> df = spark.read.csv(path="hdfs://hadoop-master:9000/user/ProjectData/ipFiles/yellow_tripdata_2019*.csv",inferSchema="true",header="true")
>>> df.count()
84399019
>>>
```

--Displaying the records from the Dataset:



```
>>> df = spark.read.csv(path="hdfs://hadoop-master:9000/user/ProjectData/ipFiles/yellow_tripdata_2019*.csv",inferSchema="true",header="true")
>>> df.count()
84399019
>>> df.show()
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----+-------+----------+-----------+--------------------+------------+
|VendorID|tpep_pickup_datetime|tpep_dropoff_datetime|passenger_count|trip_distance|RatecodeID|store_and_fwd_flag|PULocationID|DOLocationID|payment_type|fare_amount|extra|mta_tax|tip_amount|tolls_amount|improvement_surcharge|total_amount|congestion_surcharge|
+--------+--------------------+--------------------+---------------+-------------+----------+------------------+------------+------------+------------+-----+-------+----------+-----------+--------------------+------------+
|       1| 2019-01-01 00:46:40| 2019-01-01 00:53:20|              1|          1.5|         1|                 N|         151|         239|           1|        7.0|  0.5|    0.5|      1.65|         0.0|                 0.3|        9.95|                null|
|       1| 2019-01-01 00:59:47| 2019-01-01 01:18:59|              1|          2.6|         1|                 N|         239|         246|           1|       14.0|  0.5|    0.5|       1.0|         0.0|                 0.3|        16.3|                null|
|       2| 2018-12-21 13:48:30| 2018-12-21 13:52:40|              3|          0.0|         1|                 N|         236|         236|           1|        4.5|  0.5|    0.5|       0.0|         0.0|                 0.3|         5.8|                null|
|       2| 2018-11-28 15:52:25| 2018-11-28 15:55:45|              5|          0.0|         1|                 N|         193|         193|           2|        3.5|  0.5|    0.5|       0.0|         0.0|                 0.3|        7.55|                null|
|       2| 2018-11-28 15:56:57| 2018-11-28 15:58:33|              5|          0.0|         2|                 N|         193|         193|           2|       52.0|  0.0|    0.5|       0.0|         0.0|                 0.3|       55.55|                null|
|       2| 2018-11-28 16:25:49| 2018-11-28 16:28:26|              5|          0.0|         1|                 N|         193|         193|           2|        3.5|  0.5|    0.5|       0.0|        5.76|                 0.3|       13.31|                null|
|       2| 2018-11-28 16:29:37| 2018-11-28 16:33:43|              5|          0.0|         2|                 N|         193|         193|           2|       52.0|  0.0|    0.5|       0.0|         0.0|                 0.3|       55.55|                null|
|       1| 2019-01-01 00:21:28| 2019-01-01 00:28:37|              1|          1.3|         1|                 N|         163|         229|           1|        6.5|  0.5|    0.5|      1.25|         0.0|                 0.3|        9.05|                null|
|       1| 2019-01-01 00:32:01| 2019-01-01 00:45:39|              1|          3.7|         1|                 N|         229|           7|           1|       13.5|  0.5|    0.5|       3.7|         0.0|                 0.3|        18.5|                null|
|       1| 2019-01-01 00:57:32| 2019-01-01 01:09:32|              2|          2.1|         1|                 N|         141|         234|           1|       10.0|  0.5|    0.5|       1.7|         0.0|                 0.3|        13.0|                null|
|       1| 2019-01-01 00:24:04| 2019-01-01 00:47:06|              2|          2.8|         1|                 N|         246|         162|           1|       15.0|  0.5|    0.5|      3.25|         0.0|                 0.3|       19.55|                null|
|       1| 2019-01-01 00:21:59| 2019-01-01 00:28:24|              1|          0.7|         1|                 N|         238|         151|           1|        5.5|  0.5|    0.5|       1.7|         0.0|                 0.3|         8.5|                null|
|       1| 2019-01-01 00:45:21| 2019-01-01 01:31:05|              1|          8.7|         1|                 N|         163|          25|           1|       34.5|  0.5|    0.5|      7.15|         0.0|                 0.3|       42.95|                null|
|       1| 2019-01-01 00:43:19| 2019-01-01 01:07:42|              1|          6.3|         1|                 N|         224|          25|           1|       21.5|  0.5|    0.5|       5.7|         0.0|                 0.3|       28.5|                null|
|       1| 2019-01-01 00:58:24| 2019-01-01 01:15:18|              1|          2.7|         1|                 N|         141|         234|           1|       13.0|  0.5|    0.5|       1.0|         0.0|                 0.3|       15.3|                null|
|       2| 2019-01-01 00:23:14| 2019-01-01 00:25:40|              1|         0.38|         1|                 N|         170|         170|           2|        3.5|  0.5|    0.5|       0.0|         0.0|                 0.3|        4.8|                null|
|       2| 2019-01-01 00:39:51| 2019-01-01 00:48:02|              1|         0.55|         1|                 N|         170|         170|           1|        6.5|  0.5|    0.5|      1.95|         0.0|                 0.3|       9.75|                null|
|       2| 2019-01-01 00:46:00| 2019-01-01 00:49:07|              1|          0.3|         1|                 N|         107|         107|           1|        4.0|  0.5|    0.5|      1.06|         0.0|                 0.3|       6.36|                null|
|       2| 2019-01-01 00:57:45| 2019-01-01 01:03:51|              1|         1.42|         1|                 N|         170|         141|           1|        6.5|  0.5|    0.5|      1.56|         0.0|                 0.3|       9.36|                null|
|       2| 2019-01-01 00:16:16| 2019-01-01 00:25:57|              1|         1.72|         1|                 N|          41|         247|           2|        9.0|  0.5|    0.5|       0.0|         0.0|                 0.3|
```

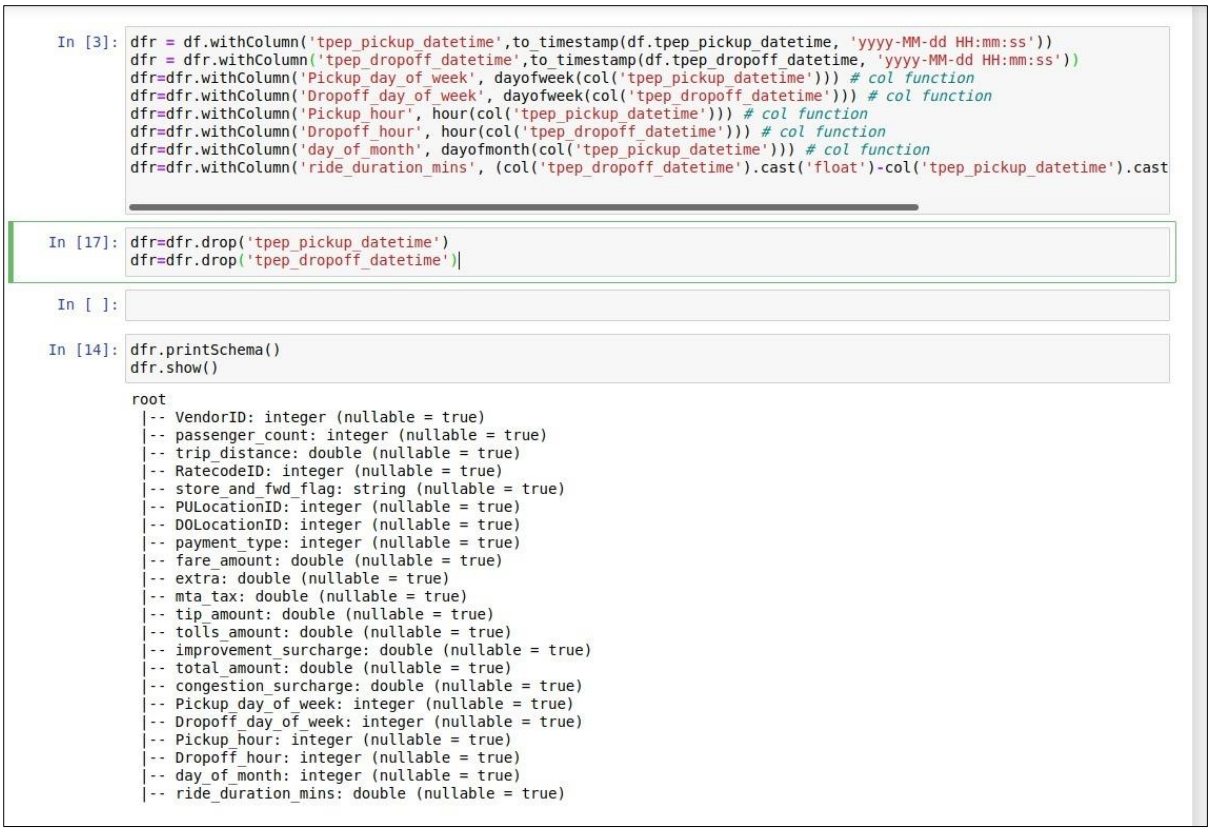
We went ahead and created a PySpark connection to analyze the data in detail:



```
In [1]:
#import spark
#findspark.init()
import pyspark
from pyspark.sql.functions import *
from pyspark.sql.types import *
import pandas as pd
import numpy as np
#import matplotlib.pyplot as plt
from datetime import datetime as dt
#import seabron as sns

In [2]:
df= spark.read.csv('hdfs://hadoop-master:9000/user/ProjectData/ipFiles/yellow_tripdata_2019*.csv',inferSchema=True,he

In [99]:
dfr=df
df.count()

Out[99]:
84399019

In [6]:
df.printSchema()

root
 |-- VendorID: integer (nullable = true)
 |-- tpep_pickup_datetime: string (nullable = true)
 |-- tpep_dropoff_datetime: string (nullable = true)
 |-- passenger_count: integer (nullable = true)
 |-- trip_distance: double (nullable = true)
 |-- RatecodeID: integer (nullable = true)
 |-- store_and_fwd_flag: string (nullable = true)
 |-- PULocationID: integer (nullable = true)
 |-- DOLocationID: integer (nullable = true)
 |-- payment_type: integer (nullable = true)
 |-- fare_amount: double (nullable = true)
 |-- extra: double (nullable = true)
 |-- mta_tax: double (nullable = true)
 |-- tip_amount: double (nullable = true)
 |-- tolls_amount: double (nullable = true)
 |-- improvement_surcharge: double (nullable = true)
 |-- total_amount: double (nullable = true)
 |-- congestion_surcharge: double (nullable = true)
```

The files that were loaded were then cleaned to limit the data only for a year

The csv files contained the date fields as string datatype, these fields were then converted into datetime to do further analysis on them:



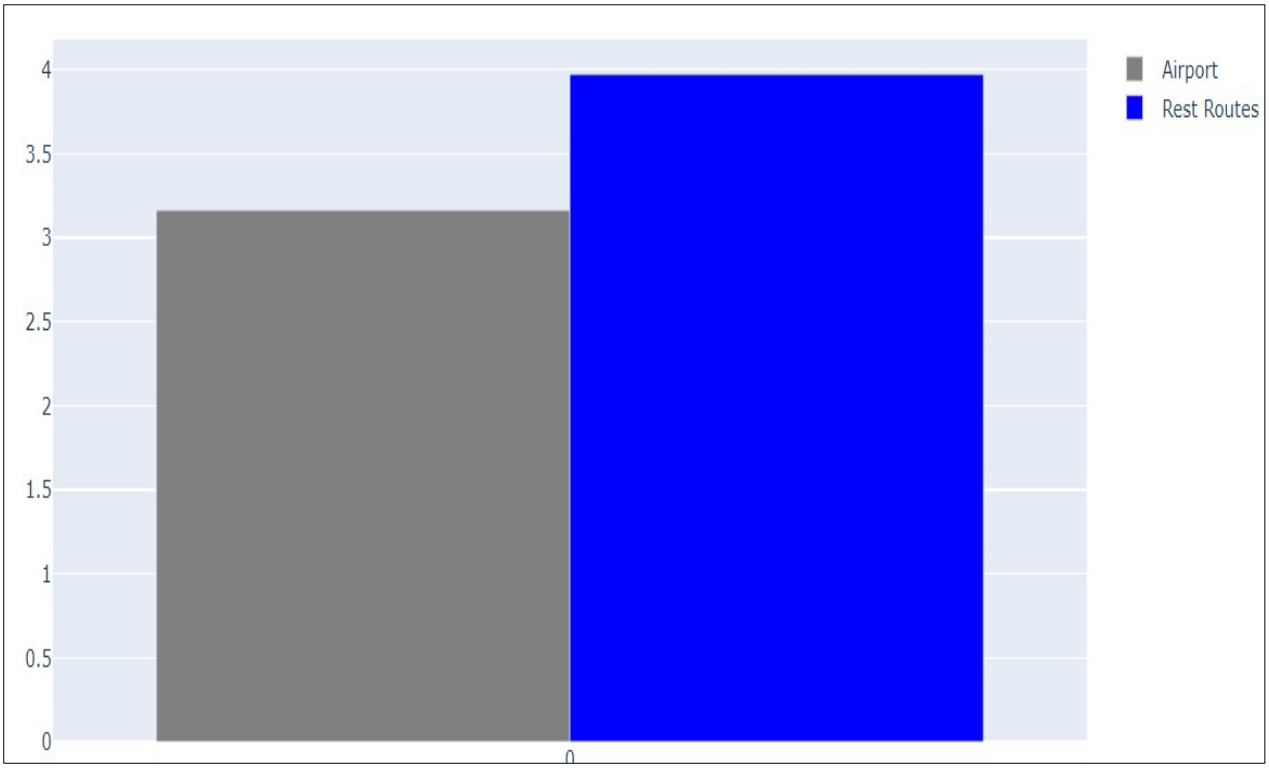
```
In [3]:
dfr = df.withColumn('tpep_pickup_datetime',to_timestamp(df.tpep_pickup_datetime, 'yyyy-MM-dd HH:mm:ss'))
dfr = dfr.withColumn('tpep_dropoff_datetime',to_timestamp(df.tpep_dropoff_datetime, 'yyyy-MM-dd HH:mm:ss'))
dfr=dfr.withColumn('Pickup_day_of_week', dayofweek(col('tpep_pickup_datetime'))) # col function
dfr=dfr.withColumn('Dropoff_day_of_week', dayofweek(col('tpep_dropoff_datetime'))) # col function
dfr=dfr.withColumn('Pickup_hour', hour(col('tpep_pickup_datetime'))) # col function
dfr=dfr.withColumn('Dropoff_hour', hour(col('tpep_dropoff_datetime'))) # col function
dfr=dfr.withColumn('day_of_month', dayofmonth(col('tpep_pickup_datetime'))) # col function
dfr=dfr.withColumn('ride_duration_mins', (col('tpep_dropoff_datetime').cast('float')-col('tpep_pickup_datetime').cast

In [17]:
dfr=dfr.drop('tpep_pickup_datetime')
dfr=dfr.drop('tpep_dropoff_datetime')

In [ ]:

In [14]:
dfr.printSchema()
dfr.show()

root
 |-- VendorID: integer (nullable = true)
 |-- passenger_count: integer (nullable = true)
 |-- trip_distance: double (nullable = true)
 |-- RatecodeID: integer (nullable = true)
 |-- store_and_fwd_flag: string (nullable = true)
 |-- PULocationID: integer (nullable = true)
 |-- DOLocationID: integer (nullable = true)
 |-- payment_type: integer (nullable = true)
 |-- fare_amount: double (nullable = true)
 |-- extra: double (nullable = true)
 |-- mta_tax: double (nullable = true)
 |-- tip_amount: double (nullable = true)
 |-- tolls_amount: double (nullable = true)
 |-- improvement_surcharge: double (nullable = true)
 |-- total_amount: double (nullable = true)
 |-- congestion_surcharge: double (nullable = true)
 |-- Pickup_day_of_week: integer (nullable = true)
 |-- Dropoff_day_of_week: integer (nullable = true)
 |-- Pickup_hour: integer (nullable = true)
 |-- Dropoff_hour: integer (nullable = true)
 |-- day_of_month: integer (nullable = true)
 |-- ride_duration_mins: double (nullable = true)
```

--Displaying 5 records from the dataset after performing the above queries:

VendorID	passenger_count	trip_distance	RatecodeID	store_and_fwd_flag	PULocationID	DOLocationID	payment_type	fare_amount	extra	mta_tax	tip_amount	tolls_amount	improvement_surcharge	total_amount	congestion_surcharge	Pickup_day_of_week	Dropoff_day_of_week	Pickup_hour	Dropoff_hour	Day_of_month	Ride-duration_minutes
1	1	0.5	1	N	239	239	1	4.0	3.0	0.5	1.0	0.0	0.3	8.8	2.5	2	2	0	0	1	0.0
1	1	0.7	1	N	230	100	2	4.5	3.0	0.5	0.0	0.0	0.3	8.3	2.5	2	2	0	0	1	0.0
1	1	10.9	1	N	68	127	1	36.0	3.0	0.5	7.95	0.0	0.3	47.75	2.5	2	2	0	1	1	0.0
1	1	0.2	1	N	68	68	2	3.5	3.0	0.5	0.0	0.0	0.3	7.3	2.5	2	2	0	0	1	0.0
1	1	4.8	1	N	50	42	1	15.5	3.0	0.5	3.85	0.0	0.3	23.15	2.5	2	2	0	0	1	0.0

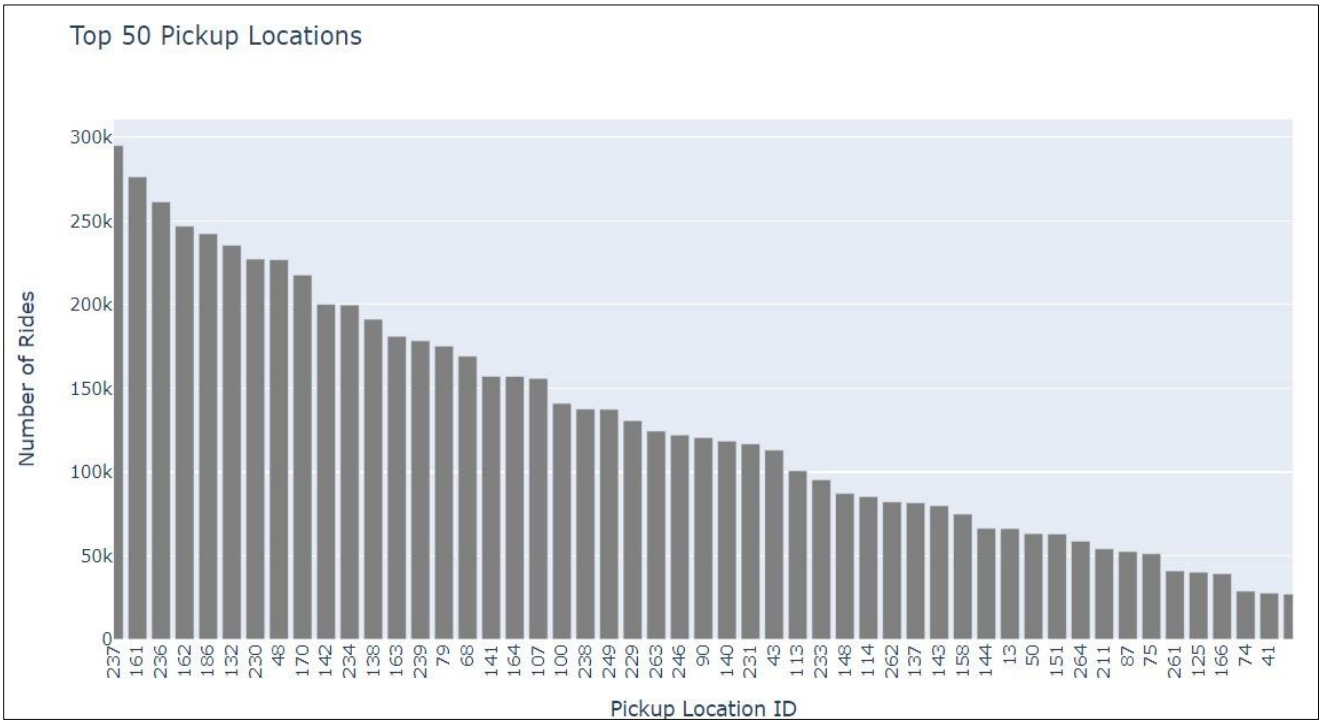
ANALYSIS AND INSIGHTS:

1. **Do airport (JFK & Newark) routes result in higher average fares?**



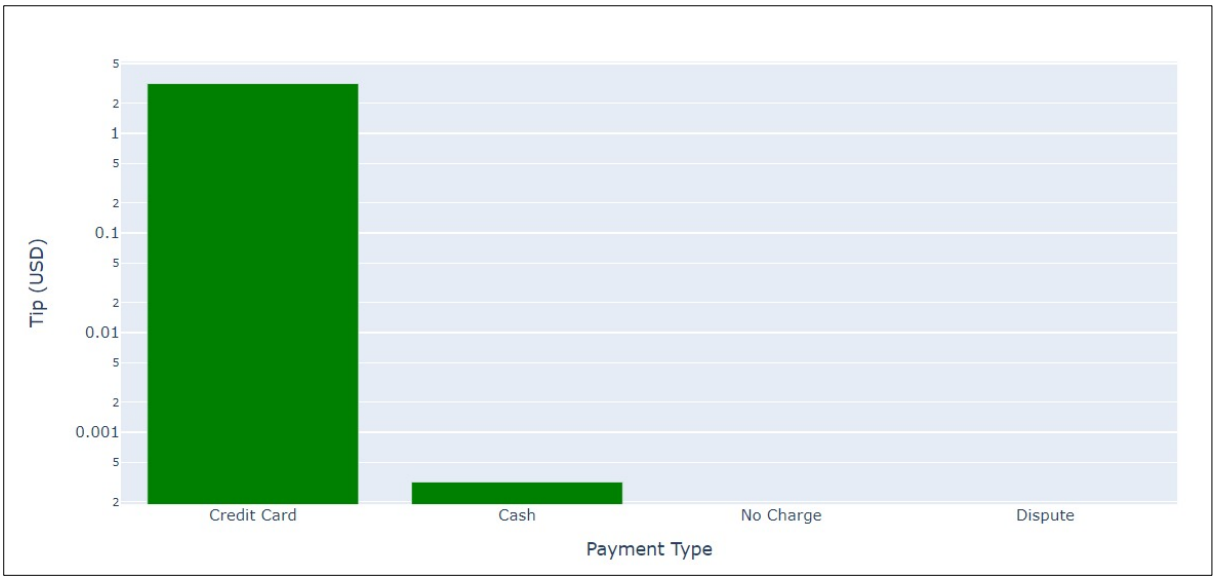
We observed that the inner routes (represented by red bar) were the ones that resulted in higher average as compared to the airport routes

2. **Where are the most pick-ups and drop-offs happening?**



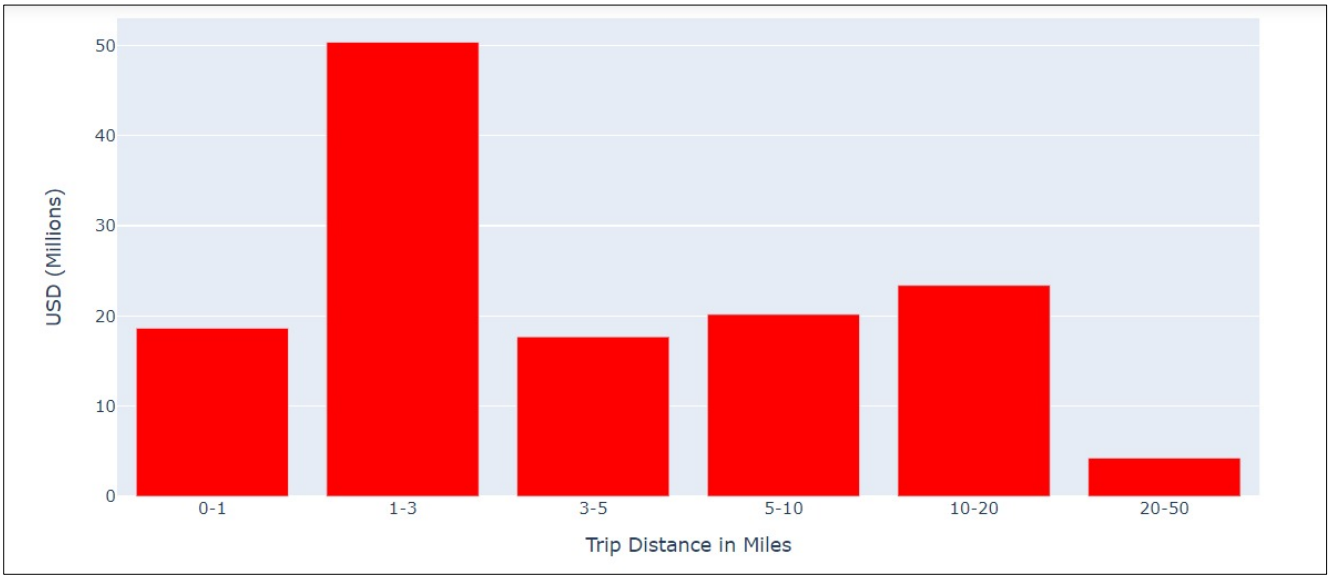
Majority of the pick-ups and drop-offs were found out in Manhattan. We could infer from this that Manhattan being a tourist attraction the majority rides were taken around it in 2019.

3. **Avg Tip by Payment Type**



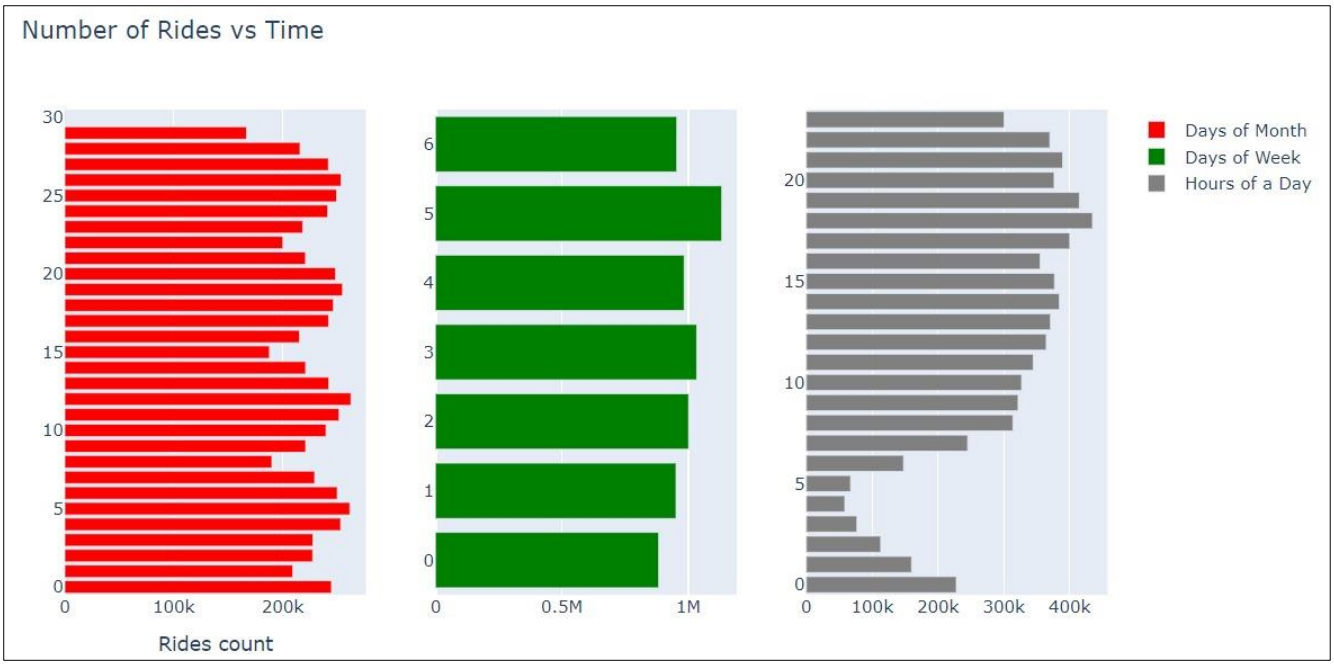
Majority of the tips received by drivers were through a Credit card payment

4. **Finding out the fare (in USD) by Trip Distance:**



There was more fare charged for shorter distance as compared to longer distance

5. **Number of rides by time**



INTERESTING FINDINGS:

Below is a list of interesting finding in our dataset:

- The trips taken to the airport were found to be lesser per distance driven!
- It was found out that people who use card as a payment option were the only ones to tip while those who used cash did not. Or most drivers just don't record cash tips.