AHMED YOUSEFF

TAXI TRIP DATA ANALYSIS

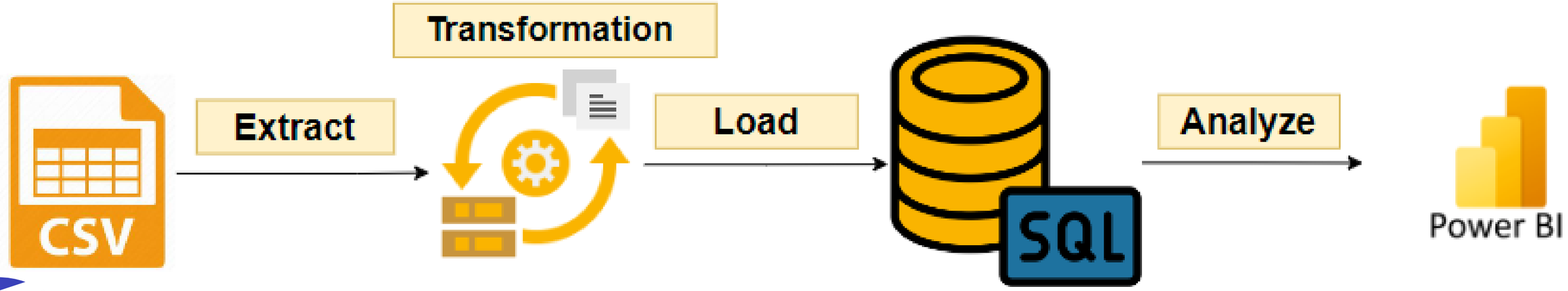
https://github.com/ahmedyouseff-19/Taxi-Trip

ABOUT DATASET

VendorID	A code indicating the TPEP 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.
PULocationID	TLC Taxi Zone in which the taximeter was engaged
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged
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**
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.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	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.
Total_amount	The total amount charged to passengers. Does not include cash tips.
Congestion_Surcharge	Total amount collected in trip for NYS congestion surcharge.
Airport_fee	$1.25 for pick up only at LaGuardia and John F. Kennedy Airports

DATA PIPELINE



EXTRACT DATA

In [ ]:
```
data=pd.read_csv("data/uber_data.csv",sep=',')
```

In [ ]:
```
data['tpep_pickup_datetime'] = pd.to_datetime(data['tpep_pickup_datetime'])
data['tpep_dropoff_datetime'] = pd.to_datetime(data['tpep_dropoff_datetime'])
```

In [ ]:
```
data = data.drop_duplicates().reset_index(drop=True)
data['trip_id'] = data.index
```

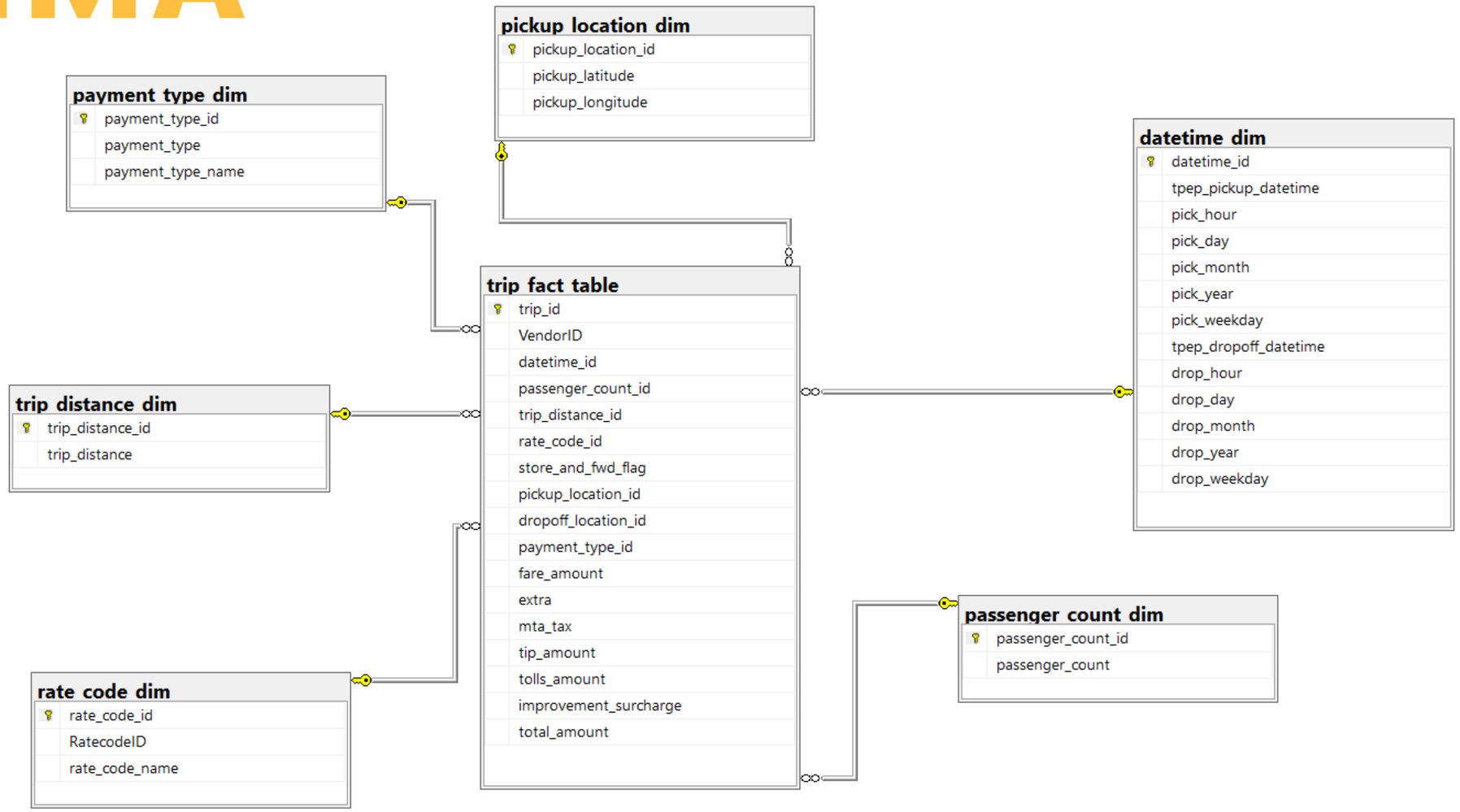
In [69]:
```
data.head()
```

Out[69]:

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip_distance	pickup_longitude	pickup_latitude	F
0	1	2016-01-03	2016-01-03 00:07:00	1	2.50	-73.976746	40.765152	
1	1	2016-01-03	2016-01-03 00:11:00	1	2.90	-73.983482	40.767925	
2	2	2016-01-03	2016-01-03 00:31:00	2	19.98	-73.782021	40.644810	
3	2	2016-01-03	2016-01-03 00:00:00	3	10.78	-73.863419	40.769814	
4	2	2016-01-03	2016-01-03 00:00:00	5	30.43	-73.971741	40.792183	

DATA WAREHOUSE

SCHEMA



TRANSFORM DATA

Create datetime_dim Table

```
datetime_dim = data[['tpep_pickup_datetime','tpep_dropoff_datetime']].reset_index(drop=True)
datetime_dim['tpep_pickup_datetime'] = datetime_dim['tpep_pickup_datetime']
datetime_dim['pick_hour'] = datetime_dim['tpep_pickup_datetime'].dt.hour
datetime_dim['pick_day'] = datetime_dim['tpep_pickup_datetime'].dt.day
datetime_dim['pick_month'] = datetime_dim['tpep_pickup_datetime'].dt.month
datetime_dim['pick_year'] = datetime_dim['tpep_pickup_datetime'].dt.year
datetime_dim['pick_weekday'] = datetime_dim['tpep_pickup_datetime'].dt.weekday
###################
datetime_dim['tpep_dropoff_datetime'] = datetime_dim['tpep_dropoff_datetime']
datetime_dim['drop_hour'] = datetime_dim['tpep_dropoff_datetime'].dt.hour
datetime_dim['drop_day'] = datetime_dim['tpep_dropoff_datetime'].dt.day
datetime_dim['drop_month'] = datetime_dim['tpep_dropoff_datetime'].dt.month
datetime_dim['drop_year'] = datetime_dim['tpep_dropoff_datetime'].dt.year
datetime_dim['drop_weekday'] = datetime_dim['tpep_dropoff_datetime'].dt.weekday
###################
datetime_dim['datetime_id'] = datetime_dim.index
datetime_dim = datetime_dim[['datetime_id', 'tpep_pickup_datetime', 'pick_hour', 'pick_day', 'pick_month', 'pick_year', 'pick_week
                             'tpep_dropoff_datetime', 'drop_hour', 'drop_day', 'drop_month', 'drop_year', 'drop_weekday']]
```

Create passenger_count_dim Table

```
passenger_count_dim = data[['passenger_count']].reset_index(drop=True)
passenger_count_dim['passenger_count_id'] = passenger_count_dim.index
passenger_count_dim = passenger_count_dim[['passenger_count_id','passenger_count']]
```

Create trip_distance_dim Table

```
trip_distance_dim = data[['trip_distance']].reset_index(drop=True)
trip_distance_dim['trip_distance_id'] = trip_distance_dim.index
trip_distance_dim = trip_distance_dim[['trip_distance_id','trip_distance']]
```

Create rate_code_dim Table

```
rate_code_type = {
    1:"Standard rate",
    2:"JFK",
    3:"Newark",
    4:"Nassau or Westchester",
    5:"Negotiated fare",
    6:"Group ride"
}

rate_code_dim = data[['RatecodeID']].reset_index(drop=True)
rate_code_dim['rate_code_id'] = rate_code_dim.index
rate_code_dim['rate_code_name'] = rate_code_dim['RatecodeID'].map(rate_code_type)
rate_code_dim = rate_code_dim[['rate_code_id','RatecodeID','rate_code_name']]
```

TRANSFORM DATA

Create pickup_location_dim Table

```
pickup_location_dim = data[['pickup_longitude', 'pickup_latitude']].reset_index(drop=True)
pickup_location_dim['pickup_location_id'] = pickup_location_dim.index
pickup_location_dim = pickup_location_dim[['pickup_location_id','pickup_latitude','pickup_longitude']]
```

Create dropoff_location_dim Table

```
dropoff_location_dim = data[['dropoff_longitude', 'dropoff_latitude']].reset_index(drop=True)
dropoff_location_dim['dropoff_location_id'] = dropoff_location_dim.index
dropoff_location_dim = dropoff_location_dim[['dropoff_location_id','dropoff_latitude','dropoff
```

Create payment_type_dim Table

```
payment_type_name = {
    1:"Credit card",
    2:"Cash",
    3:"No charge",
    4:"Dispute",
    5:"Unknown",
    6:"Voided trip"
}
payment_type_dim = data[['payment_type']].reset_index(drop=True)
payment_type_dim['payment_type_id'] = payment_type_dim.index
payment_type_dim['payment_type_name'] = payment_type_dim['payment_type'].map(payment_type_name)
payment_type_dim = payment_type_dim[['payment_type_id','payment_type','payment_type_name']]
```

Create trip_fact_table Table

```
trip_fact_table = data.merge(passenger_count_dim, left_on='trip_id', right_on='passenger_count_id') \
                      .merge(trip_distance_dim, left_on='trip_id', right_on='trip_distance_id') \
                      .merge(rate_code_dim, left_on='trip_id', right_on='rate_code_id') \
                      .merge(pickup_location_dim, left_on='trip_id', right_on='pickup_location_id') \
                      .merge(dropoff_location_dim, left_on='trip_id', right_on='dropoff_location_id')\
                      .merge(datetime_dim, left_on='trip_id', right_on='datetime_id') \
                      .merge(payment_type_dim, left_on='trip_id', right_on='payment_type_id') \
                      [['trip_id','VendorID', 'datetime_id', 'passenger_count_id',
                        'trip_distance_id', 'rate_code_id', 'store_and_fwd_flag', 'pickup_location_id', 'dropoff_location_id',
                        'payment_type_id', 'fare_amount', 'extra', 'mta_tax', 'tip_amount', 'tolls_amount',
                        'improvement_surcharge', 'total_amount']]
```

CONNECT TO DATABASE

In [ ]:

```
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=AHMEDYOUSEFF;'
                      'Database=Taxi_TripDWH;'
                      'Trusted_Connection=yes;')

cursor = conn.cursor()
```

LOAD DATA INTO DWH

```
for row in datetime_dim.itertuples():
    cursor.execute('''
                INSERT INTO datetime_dim (datetime_id, tpep_pickup_datetime,
                   pick_hour,pick_day,pick_month,pick_year,pick_weekday,
                   tpep_dropoff_datetime,drop_hour,drop_day,drop_month,
                   drop_year,drop_weekday)
                VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?)
                ''',
                row.datetime_id,
                row.tpep_pickup_datetime,
                row.pick_hour,
                row.pick_day,
                row.pick_month,
                row.pick_year,
                row.pick_weekday,
                row.tpep_dropoff_datetime,
                row.drop_hour,
                row.drop_day,
                row.drop_month,
                row.drop_year,
                row.drop_weekday,
                )
conn.commit()
```

```
for row in dropoff_location_dim.itertuples():
    cursor.execute('''
                INSERT INTO dropoff_location_dim (dropoff_location_id, dropoff_latitude,
                   dropoff_longitude)
                VALUES (?,?,?)
                ''',
                row.dropoff_location_id,
                row.dropoff_latitude,
                row.dropoff_longitude,
                )
conn.commit()
```

```
for row in passenger_count_dim.itertuples():
    cursor.execute('''
                INSERT INTO passenger_count_dim (passenger_count_id, passenger_count)
                VALUES (?,?)
                ''',
                row.passenger_count_id,
                row.passenger_count,
                )
conn.commit()
```

In [57]:
```
for row in payment_type_dim.itertuples():
    cursor.execute('''
                INSERT INTO payment_type_dim (payment_type_id, payment_type,
                   payment_type_name)
                VALUES (?,?,?)
                ''',
                row.payment_type_id,
                row.payment_type,
                row.payment_type_name,
                )
conn.commit()
```

In [59]:
```
for row in pickup_location_dim.itertuples():
    cursor.execute('''
                INSERT INTO pickup_location_dim (pickup_location_id, pickup_latitude,
                   pickup_longitude)
                VALUES (?,?,?)
                ''',
                row.pickup_location_id,
                row.pickup_latitude,
                row.pickup_longitude,
                )
conn.commit()
```

In [61]:
```
for row in rate_code_dim.itertuples():
    cursor.execute('''
                INSERT INTO rate_code_dim (rate_code_id, RatecodeID,
                   rate_code_name)
                VALUES (?,?,?)
                ''',
                row.rate_code_id,
                row.RatecodeID,
                row.rate_code_name,
                )
conn.commit()
```

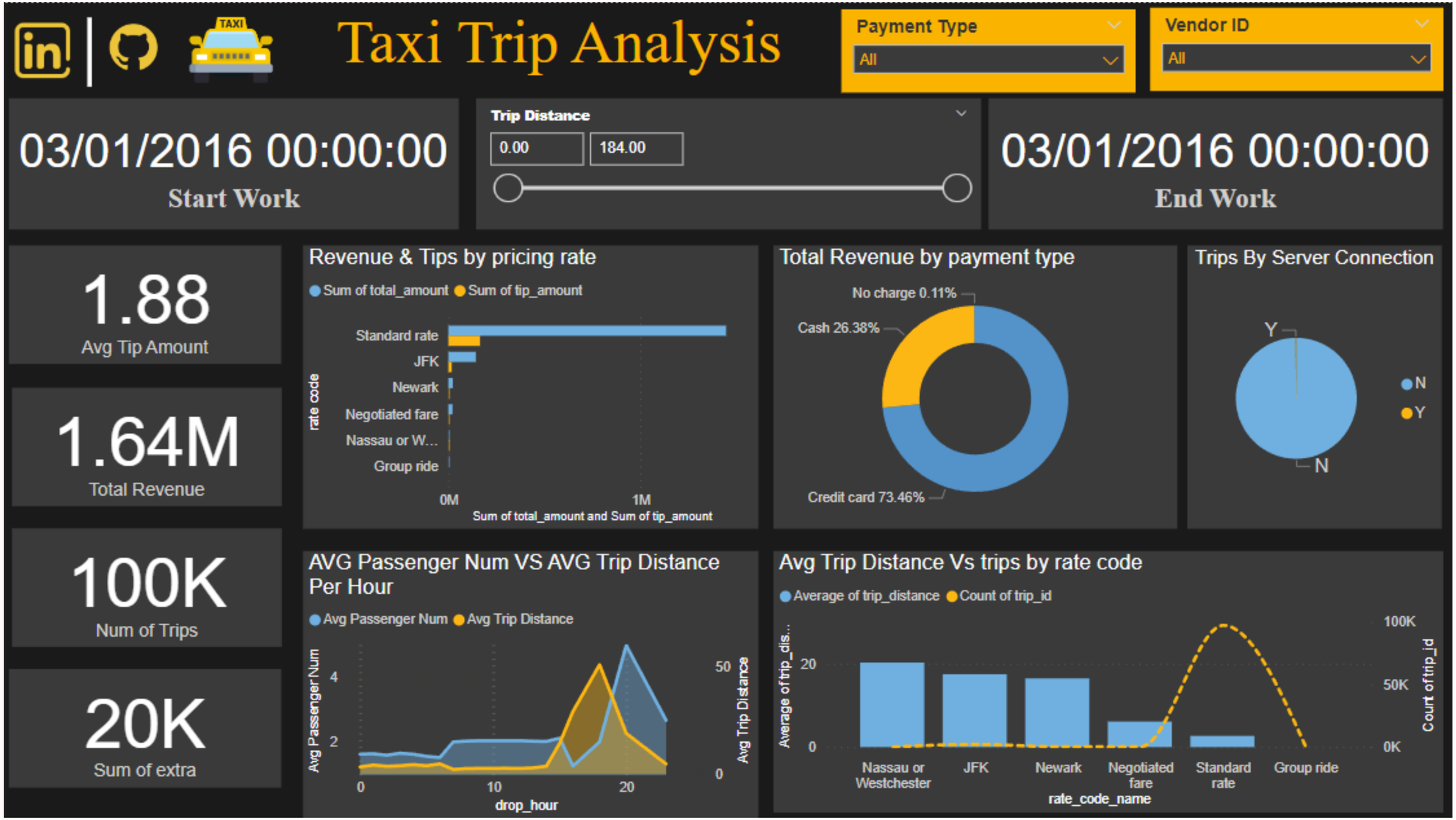
In [64]:
```
for row in trip_distance_dim.itertuples():
    cursor.execute('''
                INSERT INTO trip_distance_dim (trip_distance_id, trip_distance)
                VALUES (?,?)
                ''',
                row.trip_distance_id,
                row.trip_distance,
                )
conn.commit()
```

In [67]:
```
for row in trip_fact_table.itertuples():
    cursor.execute('''
                INSERT INTO trip_fact_table (trip_id,VendorID,datetime_id,passenger_count_id,trip_distance_id,rate_code_id,
                   dropoff_location_id,payment_type_id,fare_amount,extra,mta_tax,tip_amount,tolls_amount,improvement_surcha
                VALUES (?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)
                ''',
                row.trip_id,
                row.VendorID,
                row.datetime_id,
                row.passenger_count_id,
                row.trip_distance_id,
                row.rate_code_id,
                row.store_and_fwd_flag,
                row.pickup_location_id,
                row.dropoff_location_id,
                row.payment_type_id,
                row.fare_amount,
                row.extra,
                row.mta_tax,
                row.tip_amount,
                row.tolls_amount,
                row.improvement_surcharge,
                row.total_amount,
                )
conn.commit()
```

KPIS

1. HOW DOES AVG PASSENGER COUNT RELATE TO HOURLY TRIP DISTANCE?
2. WHAT IS THE START AND END DATE OF THE CAR'S DAILY WORK?
3. NUMBER OF TRIPS EACH DAY
4. WHAT IS THE TOTAL REVENUE FOR ALL TRIPS?
5. NUMBER OF TRIPS FOR EACH PAYMENT METHOD
6. RELATION BETWEEN REVENUE AND TIPS GRATUITIES FOR EACH PRICE RATE
7. RELATION BETWEEN NUMBER OF TRIPS AND SERVER CONNECTION
8. AMOUNT OF EXTRA FOR EACH TRIP
9. RELATION BETWEEN REVENUE AND PAYMENT TYPE

REBORT


Taxi Trip Analysis
Payment Type
All
Vendor ID
All
03/01/2016 00:00:00
Start Work
Trip Distance
0.00
184.00
03/01/2016 00:00:00
End Work
1.88
Avg Tip Amount
1.64M
Total Revenue
100K
Num of Trips
20K
Sum of extra
Revenue & Tips by pricing rate
Sum of total_amount
Sum of tip_amount
Standard rate
JFK
Newark
Negotiated fare
Nassau or W...
Group ride
rate code
0M
1M
Sum of total_amount and Sum of tip_amount
Total Revenue by payment type
No charge 0.11%
Cash 26.38%
Credit card 73.46%
Trips By Server Connection
Y
N
N
Y
AVG Passenger Num VS AVG Trip Distance Per Hour
Avg Passenger Num
Avg Trip Distance
Avg Passenger Num
Avg Trip Distance
drop_hour
Avg Trip Distance Vs trips by rate code
Average of trip_distance
Count of trip_id
Average of trip_dis...
Count of trip_id
Nassau or Westchester
JFK
Newark
Negotiated fare
Standard rate
Group ride
rate_code_name