Question 1 & 2:

With the highest fare or passenger_count by date, vendorid 103846 had the most trips, 58248

Passenger_count is 86485; Fare_amount is 776534.9

SQL Script:

SELECT Trip_count, Passenger_count

FROM TABLE

ORDER BY Passenger_count DESC;

SELECT Trip_count, Fare_amount

FROM TABLE

ORDER BY Fare_amount DESC;

In [79]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv('/Users/xiaoyan/Documents/uber homework/green_tripdata_2014-06.csv')
print(df.shape)
```

```
(1048573, 20)
```

In [80]:
```
df = df.loc[~((df['Trip_distance'] == 0))]
print(df.shape)
```

```
(1034586, 20)
```

In [81]:
```
df.dtypes
```

Out[81]:
```
VendorID                   int64
lpep_pickup_datetime      object
Lpep_dropoff_datetime     object
Store_and_fwd_flag        object
RateCodeID                 int64
Pickup_longitude         float64
Pickup_latitude          float64
Dropoff_longitude        float64
Dropoff_latitude         float64
Passenger_count            int64
Trip_distance            float64
Fare_amount              float64
Extra                    float64
MTA_tax                  float64
Tip_amount               float64
Tolls_amount             float64
Ehail_fee                float64
Total_amount             float64
Payment_type               int64
Trip_type                float64
dtype: object
```

In [82]:
```
df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
df.Lpep_dropoff_datetime = pd.to_datetime(df.Lpep_dropoff_datetime)
df["Trip_count"] = pd.NaT
df["Trip_count"] = 1
df.head(10)
```

Out[82]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	24.5	0.0	0.5	0.0	7.50	NaN	32.50	1	1.0
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	7.0	0.0	0.0	0.0	0.00	NaN	7.00	2	1.0
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	6.5	0.5	0.5	0.0	0.00	NaN	7.50	2	1.0
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	29.5	0.5	0.5	0.0	0.00	NaN	30.50	2	1.0
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	7.0	0.0	0.5	0.0	0.00	NaN	7.50	2	1.0
5	2	2014-06-01	2014-06-01 14:14:00	N	2	0.0	0.0	-73.782570	40.644104	1	...	52.0	0.0	0.5	0.0	5.33	NaN	57.83	2	1.0
6	2	2014-06-01	2014-06-01 23:03:00	N	1	0.0	0.0	-73.988029	40.755943	1	...	20.0	0.5	0.5	0.0	0.00	NaN	21.00	2	1.0
7	2	2014-06-01	2014-06-01 04:55:00	N	1	0.0	0.0	-73.924644	40.761440	1	...	4.5	0.5	0.5	0.0	0.00	NaN	5.50	2	1.0
8	2	2014-06-01	2014-06-01 13:12:00	N	1	0.0	0.0	-73.980057	40.775574	1	...	13.0	0.0	0.5	3.9	0.00	NaN	17.40	1	1.0
9	2	2014-06-01	2014-06-01 17:26:00	N	1	0.0	0.0	-73.956589	40.786682	1	...	7.0	0.0	0.5	2.1	0.00	NaN	9.60	1	1.0

10 rows × 21 columns

In [83]:
```
df.dtypes
```

Out[83]:
```
VendorID                          int64
lpep_pickup_datetime     datetime64[ns]
Lpep_dropoff_datetime    datetime64[ns]
Store_and_fwd_flag               object
RateCodeID                        int64
Pickup_longitude                float64
Pickup_latitude                 float64
Dropoff_longitude               float64
Dropoff_latitude                float64
Passenger_count                   int64
Trip_distance                   float64
Fare_amount                     float64
Extra                           float64
MTA_tax                         float64
Tip_amount                      float64
Tolls_amount                    float64
Ehail_fee                       float64
Total_amount                    float64
Payment_type                      int64
Trip_type                       float64
Trip_count                        int64
dtype: object
```

In [84]:
```
dgroup1 = df.groupby(pd.Grouper(key='lpep_pickup_datetime', freq='1D')).sum().sort_values(by='Passenger_count', ascending=False)
dgroup1.head(10)
```

Out[84]:

lpep_pickup_datetime	VendorID	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Trip_count
2014-06-07	103846	61932	-4.297526e+06	2.368649e+06	-4.301450e+06	2.370660e+06	86485	193231.64	776534.90	14715.00	28691.5	68292.09	8298.25	0.0	896552.34	92230	59049.0	58248
2014-06-14	103755	61575	-4.286293e+06	2.362491e+06	-4.290858e+06	2.364853e+06	85320	188027.30	767757.83	14388.50	28604.0	67164.37	8594.23	0.0	886532.75	92124	58846.0	58071
2014-06-21	98793	58637	-4.092964e+06	2.255820e+06	-4.097242e+06	2.258042e+06	81946	182122.00	727185.53	14276.02	27306.5	63934.74	6424.47	0.0	839137.45	87491	56155.0	55448
2014-06-13	92478	55476	-3.821681e+06	2.106727e+06	-3.824939e+06	2.108367e+06	74577	152922.79	653833.72	24531.00	25436.5	58405.38	6001.71	0.0	768227.95	82003	52585.0	51772
2014-06-08	87449	51976	-3.607677e+06	1.988463e+06	-3.610503e+06	1.989895e+06	72840	166931.06	647507.48	12781.00	24076.0	55822.04	7214.76	0.0	747435.64	77650	49572.0	48891
2014-06-15	87167	51366	-3.601553e+06	1.985142e+06	-3.604632e+06	1.986760e+06	72339	162044.57	639448.02	12719.50	24080.0	51775.91	5895.72	0.0	733931.30	78918	49340.0	48797
2014-06-06	88098	52916	-3.641735e+06	2.007605e+06	-3.646556e+06	2.010220e+06	72034	149721.50	637344.01	22095.50	24255.0	54649.93	7228.29	0.0	745592.03	79100	50165.0	49369
2014-06-20	87317	52216	-3.612092e+06	1.991175e+06	-3.615086e+06	1.992708e+06	71422	149885.89	628417.77	22078.02	24025.0	54527.81	6148.37	0.0	735216.04	78199	49678.0	48933
2014-06-01	85579	50856	-3.529189e+06	1.945299e+06	-3.532980e+06	1.947251e+06	71040	158625.63	643875.44	12297.59	23550.0	55447.80	6608.80	0.0	741793.35	76045	48501.0	47829
2014-06-22	83257	49623	-3.462095e+06	1.908138e+06	-3.465666e+06	1.910008e+06	69881	158594.51	617401.89	12251.00	23112.0	54667.08	5332.55	0.0	713781.32	73985	47516.0	46917

In [85]:
```
dgroup2 = df.groupby(pd.Grouper(key='lpep_pickup_datetime', freq='1D')).sum().sort_values(by='Fare_amount', ascending=False)
dgroup2.head(10)
```

Out[85]:

lpep_pickup_datetime	VendorID	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Trip_count
2014-06-07	103846	61932	-4.297526e+06	2.368649e+06	-4.301450e+06	2.370660e+06	86485	193231.64	776534.90	14715.00	28691.5	68292.09	8298.25	0.0	896552.34	92230	59049.0	58248
2014-06-14	103755	61575	-4.286293e+06	2.362491e+06	-4.290858e+06	2.364853e+06	85320	188027.30	767757.83	14388.50	28604.0	67164.37	8594.23	0.0	886532.75	92124	58846.0	58071
2014-06-21	98793	58637	-4.092964e+06	2.255820e+06	-4.097242e+06	2.258042e+06	81946	182122.00	727185.53	14276.02	27306.5	63934.74	6424.47	0.0	839137.45	87491	56155.0	55448
2014-06-13	92478	55476	-3.821681e+06	2.106727e+06	-3.824939e+06	2.108367e+06	74577	152922.79	653833.72	24531.00	25436.5	58405.38	6001.71	0.0	768227.95	82003	52585.0	51772
2014-06-08	87449	51976	-3.607677e+06	1.988463e+06	-3.610503e+06	1.989895e+06	72840	166931.06	647507.48	12781.00	24076.0	55822.04	7214.76	0.0	747435.64	77650	49572.0	48891
2014-06-01	85579	50856	-3.529189e+06	1.945299e+06	-3.532980e+06	1.947251e+06	71040	158625.63	643875.44	12297.59	23550.0	55447.80	6608.80	0.0	741793.35	76045	48501.0	47829
2014-06-15	87167	51366	-3.601553e+06	1.985142e+06	-3.604632e+06	1.986760e+06	72339	162044.57	639448.02	12719.50	24080.0	51775.91	5895.72	0.0	733931.30	78918	49340.0	48797
2014-06-06	88098	52916	-3.641735e+06	2.007605e+06	-3.646556e+06	2.010220e+06	72034	149721.50	637344.01	22095.50	24255.0	54649.93	7228.29	0.0	745592.03	79100	50165.0	49369
2014-06-20	87317	52216	-3.612092e+06	1.991175e+06	-3.615086e+06	1.992708e+06	71422	149885.89	628417.77	22078.02	24025.0	54527.81	6148.37	0.0	735216.04	78199	49678.0	48933
2014-06-22	83257	49623	-3.462095e+06	1.908138e+06	-3.465666e+06	1.910008e+06	69881	158594.51	617401.89	12251.00	23112.0	54667.08	5332.55	0.0	713781.32	73985	47516.0	46917

In [86]:
```
print(dgroup1["Passenger_count"].max())
print(dgroup2["Fare_amount"].max())
```

```
86485
776534.9
```

In [ ]:

Question3

In [87]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv('/Users/xiaoyan/Documents/uber homework/green_tripdata_2014-06.csv')

df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
df.Lpep_dropoff_datetime = pd.to_datetime(df.Lpep_dropoff_datetime)

df = df.loc[~((df['Trip_distance'] == 0))]
df['Tip_amount'] = df['Tip_amount'].fillna(0)
df.head(5)
```

Out[87]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	8.00	24.5	0.0	0.5	0.0	7.5	NaN	32.5	1
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	0.94	7.0	0.0	0.0	0.0	0.0	NaN	7.0	2
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	1.12	6.5	0.5	0.5	0.0	0.0	NaN	7.5	2
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	6.97	29.5	0.5	0.5	0.0	0.0	NaN	30.5	2
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	1.04	7.0	0.0	0.5	0.0	0.0	NaN	7.5	2

In [88]:
```
df['RateCodeID'] = (df.Fare_amount + df.Tip_amount + df.Tolls_amount + df.Total_amount) / df.Trip_distance
df.head(5)
```

Out[88]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type
0	2	2014-06-01	2014-06-01 07:40:00	N	8.062500	0.0	0.0	-73.865570	40.770863	1	8.00	24.5	0.0	0.5	0.0	7.5	NaN	32.5	1
1	2	2014-06-01	2014-06-01 15:25:00	N	14.893617	0.0	0.0	-73.928314	40.815762	1	0.94	7.0	0.0	0.0	0.0	0.0	NaN	7.0	2
2	2	2014-06-01	2014-06-01 02:06:00	N	12.500000	0.0	0.0	-73.929260	40.855412	1	1.12	6.5	0.5	0.5	0.0	0.0	NaN	7.5	2
3	2	2014-06-01	2014-06-01 23:33:00	N	8.608321	0.0	0.0	-73.997047	40.749828	1	6.97	29.5	0.5	0.5	0.0	0.0	NaN	30.5	2
4	2	2014-06-01	2014-06-01 13:05:00	N	13.942308	0.0	0.0	-73.948982	40.797092	1	1.04	7.0	0.0	0.5	0.0	0.0	NaN	7.5	2

In [ ]:

Question4

The average difference between the driven distance and the haversine distance of the trip is 8.441395808977928.

In [89]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv('/Users/xiaoyan/Documents/uber homework/green_tripdata_2014-06.csv')

df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
df.Lpep_dropoff_datetime = pd.to_datetime(df.Lpep_dropoff_datetime)

df = df.loc[~((df['Trip_distance'] == 0))]
df.head(5)
```

Out[89]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	8.00	24.5	0.0	0.5	0.0	7.5	NaN	32.5	1
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	0.94	7.0	0.0	0.0	0.0	0.0	NaN	7.0	2
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	1.12	6.5	0.5	0.5	0.0	0.0	NaN	7.5	2
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	6.97	29.5	0.5	0.5	0.0	0.0	NaN	30.5	2
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	1.04	7.0	0.0	0.5	0.0	0.0	NaN	7.5	2

In [90]:
```
df["Haversine_distance"] = pd.NaT
df["Distance_difference"] = pd.NaT
df
df.head(5)
```

Out[90]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Haversine_distance
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	NaN
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	NaN
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	NaN
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	NaN
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	NaN

5 rows × 22 columns

In [91]:
```
from numpy import cos, sin, arcsin, sqrt
from math import radians

import pandas as pd
import numpy as np
from datetime import datetime, timedelta

def haversine(row):
    lon1 = row['Pickup_longitude']
    lat1 = row['Pickup_latitude']
    lon2 = row['Dropoff_longitude']
    lat2 = row['Dropoff_latitude']
    lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2
    c = 2 * arcsin(sqrt(a))
    m = 3956 * c
    return m

df['Haversine_distance'] = df.apply(lambda row: haversine(row), axis=1)
```

In [92]:
```
df["Distance_difference"] = df["Haversine_distance"] - df["Trip_distance"]
df.head(5)
```

Out[92]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Haversine_distance
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	5375.238
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	5379.035
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	5379.590
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	5381.727
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	5379.858

5 rows × 22 columns

In [93]:
```
average_distance = df["Distance_difference"].mean()
print(average_distance)
```

```
8.441395808977928
```

In [ ]:

Question5

In [94]:
```
import pandas as pd
import numpy as np
import seaborn as sns
sns.set()
import matplotlib.pyplot as plt
%matplotlib inline
import datetime as dt

df = pd.read_csv('/Users/xiaoyan/Documents/uber homework/green_tripdata_2014-06.csv')
df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
df.Lpep_dropoff_datetime = pd.to_datetime(df.Lpep_dropoff_datetime)

df = df.loc[~((df['Trip_distance'] == 0))]
df['Tip_amount'] = df['Tip_amount'].fillna(0)

df.head(5)
```

Out[94]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	8.00	24.5	0.0	0.5	0.0	7.5	NaN	32.5	1
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	0.94	7.0	0.0	0.0	0.0	0.0	NaN	7.0	2
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	1.12	6.5	0.5	0.5	0.0	0.0	NaN	7.5	2
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	6.97	29.5	0.5	0.5	0.0	0.0	NaN	30.5	2
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	1.04	7.0	0.0	0.5	0.0	0.0	NaN	7.5	2

In [95]:
```
df["Hour"] = pd.NaT
df["Period"] = pd.NaT
df.head(5)
```

Out[95]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Hour	Period
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	NaT	NaT
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	NaT	NaT
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	NaT	NaT
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	NaT	NaT
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	NaT	NaT

5 rows × 22 columns

In [96]:
```
def period(x):
    if x in range(6,12):
        return "Morning"
    elif x in range(12,16):
        return "Afternoon"
    elif x in range(16,22):
        return "Evening"
    else:
        return "Late Night"
```

In [97]:
```
df["Hour"] = df["lpep_pickup_datetime"].dt.hour
df["Period"] = df["Hour"].apply(period)
df.head(5)
```

Out[97]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Hour	Period
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	0	Late Night
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	0	Late Night
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	0	Late Night
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	0	Late Night
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	0	Late Night

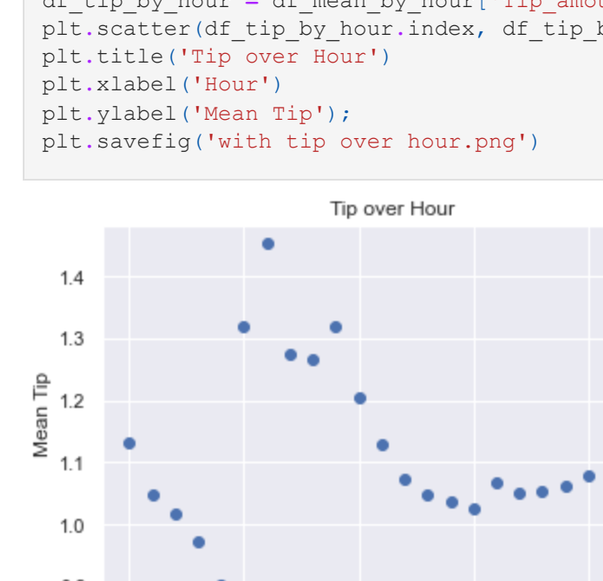
5 rows × 22 columns

In [98]:
```
dgroup1 = df.groupby('Hour')
df_mean_by_hour = dgroup1.mean()
df_mean_by_hour.head()
```

Out[98]:

Hour	VendorID	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type
0	1.789996	1.052438	-72.751303	40.090728	-73.671980	40.597184	1.479466	3.241512	12.766197	0.490094	0.493311	1.132805	0.080307	NaN	14.962714	1.569988	1.011915
1	1.788081	1.053073	-73.856770	40.699679	-73.888942	40.717682	1.472829	3.174336	12.293523	0.491798	0.493241	1.048159	0.072177	NaN	14.399301	1.589704	1.012347
2	1.782937	1.058402	-73.846275	40.694003	-73.893261	40.719677	1.473775	3.204460	12.257606	0.490160	0.492512	1.018588	0.065098	NaN	14.324144	1.598195	1.013568
3	1.772114	1.065110	-73.845633	40.697876	-73.872680	40.712592	1.501028	3.316927	12.467817	0.489189	0.491861	0.973130	0.083708	NaN	14.506527	1.626932	1.014880
4	1.748594	1.100340	-73.827693	40.697839	-73.881953	40.727390	1.472711	3.671389	13.491933	0.483709	0.487463	0.901526	0.166040	NaN	15.531161	1.690012	1.021639

In [99]:
```
df_tip_by_hour = df_mean_by_hour['Tip_amount']
plt.scatter(df_tip_by_hour.index, df_tip_by_hour )
plt.title('Tip over Hour')
plt.xlabel('Hour')
plt.ylabel('Mean Tip');
plt.savefig('with tip over hour.png')
```



Analysis

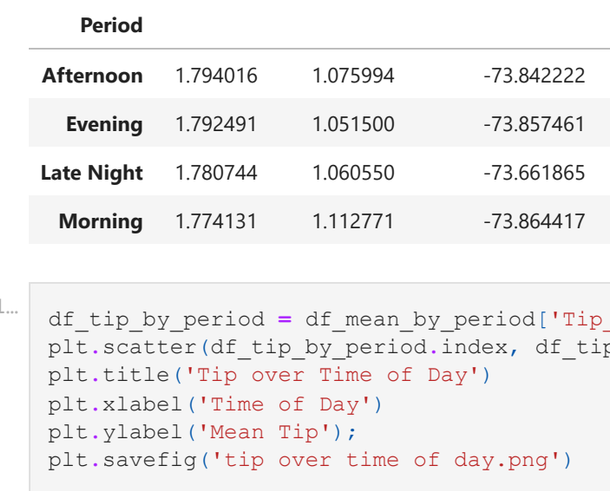
As you can see, with tipping over hour, approximately from 5:00am to 10:00am(In the morning), passengers tip most.

In [100]:
```
dgroup2 = df.groupby('Period')
df_mean_by_period = dgroup2.mean()
df_mean_by_period.head()
```

Out[100]:

Period	VendorID	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Hour
Afternoon	1.794016	1.075994	-73.842222	40.714302	-73.864745	40.723036	1.429526	3.101380	13.186733	0.012058	0.490724	1.044353	0.165856	NaN	14.899890	1.638235	1.016249	13.611736
Evening	1.792491	1.051500	-73.857461	40.710408	-73.882001	40.721676	1.463932	2.915657	12.380143	0.610283	0.493640	1.074414	0.101473	NaN	14.660325	1.595521	1.011238	18.546912
Late Night	1.780744	1.060550	-73.661865	40.594913	-73.848265	40.696643	1.479735	3.326575	12.818375	0.489554	0.492369	1.118257	0.093971	NaN	15.012860	1.584814	1.013710	9.807796
Morning	1.774131	1.112771	-73.864417	40.732837	-73.886764	40.739450	1.393931	3.261774	13.384389	0.006921	0.486172	1.253474	0.201131	NaN	15.337630	1.560452	1.025737	8.901915

In [101]:
```
df_tip_by_period = df_mean_by_period['Tip_amount']
plt.scatter(df_tip_by_period.index, df_tip_by_period)
plt.title('Tip over Time of Day')
plt.xlabel('Time of Day')
plt.ylabel('Mean Tip');
plt.savefig('tip over time of day.png')
```



Analysis

With tipping over time of day, from most to least, morning, late night, evening and afternoon. Based on the rule we split the time of day before, morning(6:00 am to 12:00pm), afternoon(from 12:00 pm to 16:00 pm), evening(from 16:00 pm to 22:00 pm), late night(from 22:00 pm to 6:00 am). We can see that Morning and Late Night(from 22:00pm to 12:00pm), passengers tip more.

In [ ]:

Question 6

In [102]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv('/Users/xiaoyan/Documents/uber homework/green_tripdata_2014-06.csv')

df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
df.Lpep_dropoff_datetime = pd.to_datetime(df.Lpep_dropoff_datetime)

df = df.loc[~((df['Trip_distance'] == 0))]
df.head(5)
```

Out[102]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	8.00	24.5	0.0	0.5	0.0	7.5	NaN	32.5	1
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	0.94	7.0	0.0	0.0	0.0	0.0	NaN	7.0	2
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	1.12	6.5	0.5	0.5	0.0	0.0	NaN	7.5	2
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	6.97	29.5	0.5	0.5	0.0	0.0	NaN	30.5	2
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	1.04	7.0	0.0	0.5	0.0	0.0	NaN	7.5	2

In [103]:
```
df["Haversine_distance"] = pd.NaT
df["Distance_difference"] = pd.NaT
df
df.head(5)
```

Out[103]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Haversine_distance
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	NaN
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	NaN
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	NaN
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	NaN
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	NaN

5 rows × 22 columns

In [104]:
```
from numpy import cos, sin, arcsin, sqrt
from math import radians

import pandas as pd
import numpy as np
from datetime import datetime, timedelta

def haversine(row):
    lon1 = row['Pickup_longitude']
    lat1 = row['Pickup_latitude']
    lon2 = row['Dropoff_longitude']
    lat2 = row['Dropoff_latitude']
    lon1, lat1, lon2, lat2 = map(radians, [lon1, lat1, lon2, lat2])
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = sin(dlat/2)**2 + cos(lat1) * cos(lat2) * sin(dlon/2)**2
    c = 2 * arcsin(sqrt(a))
    m = 3956 * c
    return m

df['Haversine_distance'] = df.apply(lambda row: haversine(row), axis=1)
```

In [105]:
```
df["Distance_difference"] = df["Haversine_distance"] - df["Trip_distance"]
df.head(5)
```

Out[105]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Haversine_distance
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	5375.238
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	5379.035
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	5379.590
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	5381.727
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	5379.858

5 rows × 22 columns

In [106]:
```
average_distance = df["Distance_difference"].mean()
print(average_distance)
```

```
8.441395808977928
```

In [107]:
```
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
predictors = ["Haversine_distance"]
X = df[predictors]
X_train, X_test, y_train, y_test = train_test_split(X, df.Trip_distance, test_size = 0.3, random_state = 5)
```

In [108]:
```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(724210, 1)
(310376, 1)
(724210,)
(310376,)
```

In [109]:
```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(X_train, y_train)

pred_test = lm.predict(X_test)
print(y_test)
print(pred_test)

r2 = r2_score(y_test, pred_test)
print('R^2: %.4f' % (r2))
```

```
180709    5.72
485614    1.61
114007    0.81
65357     1.40
459152    0.30
          ...
817885    7.19
399046    1.86
312061    0.71
87846     0.49
720793    1.41
Name: Trip_distance, Length: 310376, dtype: float64
[3.1271665  3.12634823 3.12619813 ... 3.12614296 3.1261442  3.12618223]
R^2: 0.0002
```

Analysis

When the driver picks up the passenger, the information they usually provide is the pick up latitude and longitude, the drop off latitude and longitude. However, use this kind of information to predict the length of trip is unconvenient, they are not in the same unit. It's difficult to predict. Based on the previous question 4, we can use the pick up latitude and longitude, the drop off latitude and longitude to calculate the haversine distance. I assume there's some correlation between the trip distance and haversine distance. Therefore, I use the linear regression to see whether we can use haversine distance to calculate trip distance or not. The R^2 score is 0.0002, obviously, that can not be explained fully, that means we can not use the haversine distance to calculate the trip distance correctly if we just look at the R^2 score. However, we need to consider other factors which might affect the R^2 score. When I calculate the Haversine distance, I use the Haversine formula found on google, I am not sure the accuracy of this formula. Because that's the only independent variable I use to predict the trip distance, if the formula is not totally correct which must cause the following problem(R^2 is not high enough). However, if we look at trip distance(the true test data) and the predicted trip distance(the predict test data), the distance difference is not too much. In my opinion, we can predict the approximate length of trip based on the given information(pick up latitude, longitude and the drop off latitude, longitude), but that's not the accurate length.

In [ ]:

Question7

In [110]:
```
import pandas as pd
import numpy as np
import seaborn as sns
sns.set()
import matplotlib.pyplot as plt
%matplotlib inline
import datetime as dt

df = pd.read_csv('/Users/xiaoyan/Documents/uber homework/green_tripdata_2014-06.csv')
df.lpep_pickup_datetime = pd.to_datetime(df.lpep_pickup_datetime)
df.Lpep_dropoff_datetime = pd.to_datetime(df.Lpep_dropoff_datetime)

df = df.loc[~((df['Trip_distance'] == 0))]
df['Tip_amount'] = df['Tip_amount'].fillna(0)

df.head(5)
```

Out[110]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	8.00	24.5	0.0	0.5	0.0	7.5	NaN	32.5	1
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	0.94	7.0	0.0	0.0	0.0	0.0	NaN	7.0	2
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	1.12	6.5	0.5	0.5	0.0	0.0	NaN	7.5	2
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	6.97	29.5	0.5	0.5	0.0	0.0	NaN	30.5	2
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	1.04	7.0	0.0	0.5	0.0	0.0	NaN	7.5	2

In [111]:
```
df["Hour"] = pd.NaT
df["Period"] = pd.NaT
df.head(5)

def period(x):
    if x in range(6,12):
        return "Morning"
    elif x in range(12,16):
        return "Afternoon"
    elif x in range(16,22):
        return "Evening"
    else:
        return "Late Night"
```

In [112]:
```
df["Hour"] = df["lpep_pickup_datetime"].dt.hour
df["Period"] = df["Hour"].apply(period)
df.head(5)
```

Out[112]:

	VendorID	lpep_pickup_datetime	Lpep_dropoff_datetime	Store_and_fwd_flag	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	...	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Hour	Period
0	2	2014-06-01	2014-06-01 07:40:00	N	1	0.0	0.0	-73.865570	40.770863	1	...	0.0	0.5	0.0	7.5	NaN	32.5	1	1.0	0	Late Night
1	2	2014-06-01	2014-06-01 15:25:00	N	5	0.0	0.0	-73.928314	40.815762	1	...	0.0	0.0	0.0	0.0	NaN	7.0	2	1.0	0	Late Night
2	2	2014-06-01	2014-06-01 02:06:00	N	1	0.0	0.0	-73.929260	40.855412	1	...	0.5	0.5	0.0	0.0	NaN	7.5	2	1.0	0	Late Night
3	2	2014-06-01	2014-06-01 23:33:00	N	1	0.0	0.0	-73.997047	40.749828	1	...	0.5	0.5	0.0	0.0	NaN	30.5	2	1.0	0	Late Night
4	2	2014-06-01	2014-06-01 13:05:00	N	1	0.0	0.0	-73.948982	40.797092	1	...	0.0	0.5	0.0	0.0	NaN	7.5	2	1.0	0	Late Night

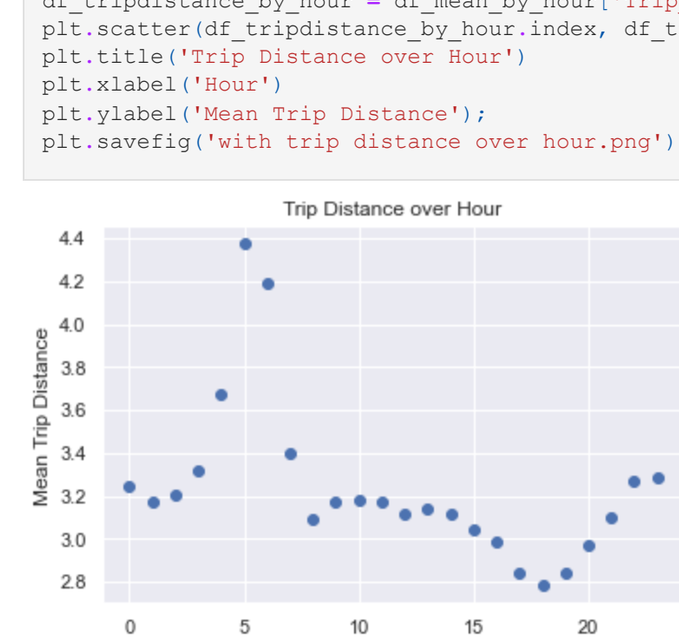
5 rows × 22 columns

In [113]:
```
dgroup1 = df.groupby('Hour')
df_mean_by_hour = dgroup1.mean()
df_mean_by_hour.head()
```

Out[113]:

Hour	VendorID	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type
0	1.789996	1.052438	-72.751303	40.090728	-73.671980	40.597184	1.479466	3.241512	12.766197	0.490094	0.493311	1.132805	0.080307	NaN	14.962714	1.569988	1.011915
1	1.788081	1.053073	-73.856770	40.699679	-73.888942	40.717682	1.472829	3.174336	12.293523	0.491798	0.493241	1.048159	0.072177	NaN	14.399301	1.589704	1.012347
2	1.782937	1.058402	-73.846275	40.694003	-73.893261	40.719677	1.473775	3.204460	12.257606	0.490160	0.492512	1.018588	0.065098	NaN	14.324144	1.598195	1.013568
3	1.772114	1.065110	-73.845633	40.697876	-73.872680	40.712592	1.501028	3.316927	12.467817	0.489189	0.491861	0.973130	0.083708	NaN	14.506527	1.626932	1.014880
4	1.748594	1.100340	-73.827693	40.697839	-73.881953	40.727390	1.472711	3.671389	13.491933	0.483709	0.487463	0.901526	0.166040	NaN	15.531161	1.690012	1.021639

In [114]:
```
df_tripdistance_by_hour = df_mean_by_hour['Trip_distance']
plt.scatter(df_tripdistance_by_hour.index, df_tripdistance_by_hour )
plt.title('Trip Distance over Hour')
plt.xlabel('Hour')
plt.ylabel('Mean Trip Distance');
plt.savefig('with trip distance over hour.png')
```



Analysis

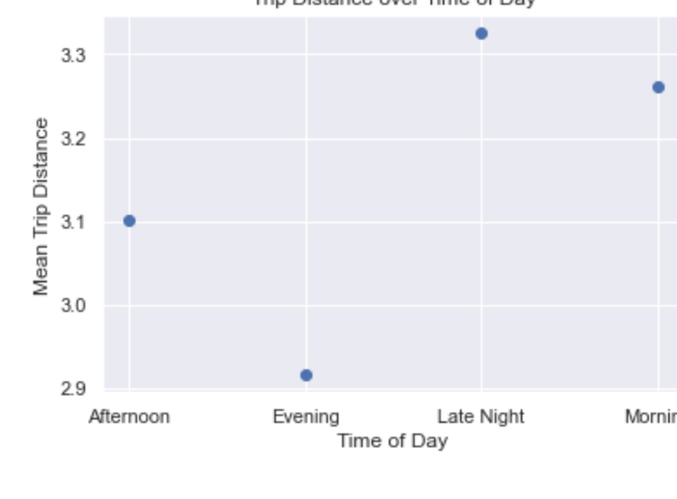
As we can see, from 5:00 am to 6:00 am(pick up time), trip distance is more than other time of day. If the passengers departure earlier, maybe the place they need to go is further than usual.

In [115]:
```
dgroup2 = df.groupby('Period')
df_mean_by_period = dgroup2.mean()
df_mean_by_period.head()
```

Out[115]:

Period	VendorID	RateCodeID	Pickup_longitude	Pickup_latitude	Dropoff_longitude	Dropoff_latitude	Passenger_count	Trip_distance	Fare_amount	Extra	MTA_tax	Tip_amount	Tolls_amount	Ehail_fee	Total_amount	Payment_type	Trip_type	Hour
Afternoon	1.794016	1.075994	-73.842222	40.714302	-73.864745	40.723036	1.429526	3.101380	13.186733	0.012058	0.490724	1.044353	0.165856	NaN	14.899890	1.638235	1.016249	13.611736
Evening	1.792491	1.051500	-73.857461	40.710408	-73.882001	40.721676	1.463932	2.915657	12.380143	0.610283	0.493640	1.074414	0.101473	NaN	14.660325	1.595521	1.011238	18.546912
Late Night	1.780744	1.060550	-73.661865	40.594913	-73.848265	40.696643	1.479735	3.326575	12.818375	0.489554	0.492369	1.118257	0.093971	NaN	15.012860	1.584814	1.013710	9.807796
Morning	1.774131	1.112771	-73.864417	40.732837	-73.886764	40.739450	1.393931	3.261774	13.384389	0.006921	0.486172	1.253474	0.201131	NaN	15.337630	1.560452	1.025737	8.901915

In [116]:
```
df_tripdistance_by_period = df_mean_by_period['Trip_distance']
plt.scatter(df_tripdistance_by_period.index, df_tripdistance_by_period)
plt.title('Trip Distance over Time of Day')
plt.xlabel('Time of Day')
plt.ylabel('Mean Trip Distance');
plt.savefig('trip distance over time of day.png')
```



Analysis

As we can see, in the late night, the trip distance is further than other time of day. According to the rule we define before, late night is from 22:00 pm to 6:00 am, trip distance is further than other time of day, which also corresponds to the result we analyze the trip distance over hour.