```
import pandas as pd
import numpy as np
```

```
df = pd.read_csv("2023_Taxi_5M.csv")
```

```
/tmp/ipython-input-729068485.py:1: DtypeWarning: Columns (4,10,16) have mixed ty
  df = pd.read_csv("2023_Taxi_5M.csv")
```

```
df.head()
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	trip
0	2	01/01/2023 12:32:10 AM	01/01/2023 12:40:36 AM	1.0	
1	2	01/01/2023 12:39:42 AM	01/01/2023 12:50:36 AM	1.0	
2	2	01/01/2023 12:09:29 AM	01/01/2023 12:29:23 AM	2.0	
3	2	01/01/2023 12:45:11 AM	01/01/2023 01:07:39 AM	1.0	
4	1	01/01/2023 12:51:45 AM	01/01/2023 12:58:18 AM	1.0	

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4278823 entries, 0 to 4278822
Data columns (total 19 columns):
 #   Column                 Dtype  
---  ------                 -----  
 0   VendorID               int64  
 1   tpep_pickup_datetime   object 
 2   tpep_dropoff_datetime  object 
 3   passenger_count        float64
 4   trip_distance          object 
 5   RatecodeID             float64
 6   store_and_fwd_flag     object 
 7   PULocationID           int64  
 8   DOLocationID           int64  
 9   payment_type           float64
 10  fare_amount            object 
 11  extra                  float64
 12  mta_tax                float64
 13  tip_amount             float64
 14  tolls_amount           float64
 15  improvement_surcharge  float64
 16  total_amount           object 
 17  congestion_surcharge   float64
 18  airport_fee            float64
dtypes: float64(10), int64(3), object(6)
memory usage: 620.3+ MB
```

```
df.isna().sum()
```

	0
VendorID	0
tpep_pickup_datetime	0
tpep_dropoff_datetime	0
passenger_count	130107
trip_distance	0
RatecodeID	130107
store_and_fwd_flag	130107
PULocationID	0
DOLocationID	0
payment_type	1
fare_amount	1
extra	1
mta_tax	1
tip_amount	1
tolls_amount	1
improvement_surcharge	1
total_amount	1
congestion_surcharge	130108
airport_fee	130108

dtype: int64

```
df['airport_fee'] = df['airport_fee'].fillna(0)
df['congestion_surcharge'] = df['congestion_surcharge'].fillna(0)
```

```
print(df['passenger_count'].mean())
```

```
1.367321118148362
```

```
df['passenger_count'] = df['passenger_count'].fillna(1)

df = df.drop(columns=['store_and_fwd_flag'])
df['RatecodeID'] = df['RatecodeID'].fillna(99)
```

```
df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'])
df['tpep_dropoff_datetime'] = pd.to_datetime(df['tpep_dropoff_datetime'])
```

```
/tmp/ipython-input-1208249868.py:1: UserWarning: Could not infer format, so eac
  df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'])
/tmp/ipython-input-1208249868.py:2: UserWarning: Could not infer format, so eac
  df['tpep_dropoff_datetime'] = pd.to_datetime(df['tpep_dropoff_datetime'])
```

```
df.select_dtypes(include=['number']).agg([ 'min', 'max','mean']).T
```

	min	max	mean
VendorID	1.00	6.00	1.738111
passenger_count	0.00	9.00	1.356152
RatecodeID	1.00	99.00	4.580434
PULocationID	1.00	265.00	165.163665
DOLocationID	1.00	265.00	163.940594
payment_type	0.00	4.00	1.188986
extra	-7.50	96.38	1.568548
mta_tax	-0.50	5.75	0.486007
tip_amount	-81.00	480.10	3.518550
tolls_amount	-49.85	196.99	0.588766
improvement_surcharge	-1.00	1.00	0.980018
congestion_surcharge	-2.50	2.75	2.195470
airport_fee	-1.75	1.75	0.135036

```
df['trip_distance'] = df['trip_distance'].str.replace(',', '', regex=False)

df['trip_distance'] = df['trip_distance'].astype(float)
```

```
float_cols = [
    'passenger_count',
    'trip_distance',
    'RatecodeID',
    'fare_amount',
    'extra',
    'mta_tax',
    'tip_amount',
    'tolls_amount',
    'improvement_surcharge',
    'total_amount',
    'congestion_surcharge',
    'airport_fee'
]

for col in float_cols:
    if col in df.columns:
        df[col] = (
            df[col]
            .astype(str)
            .str.replace(',', '', regex=False)
        )
        df[col] = pd.to_numeric(df[col], errors='coerce')
```

```
df.select_dtypes(include=['number']).agg([ 'min', 'max','mean']).T
```

	min	max	mean
VendorID	1.00	6.00	1.738111
passenger_count	0.00	9.00	1.356152
trip_distance	0.00	335004.33	5.948710
RatecodeID	1.00	99.00	4.580434
PULocationID	1.00	265.00	165.163665
DOLocationID	1.00	265.00	163.940594
payment_type	0.00	4.00	1.188986
fare_amount	-900.00	386983.63	19.598446
extra	-7.50	96.38	1.568548
mta_tax	-0.50	5.75	0.486007
tip_amount	-81.00	480.10	3.518550
tolls_amount	-49.85	196.99	0.588766
improvement_surcharge	-1.00	1.00	0.980018
total_amount	-901.00	386987.63	28.538945
congestion_surcharge	-2.50	2.75	2.195470
airport_fee	-1.75	1.75	0.135036

```
df = df[df['trip_distance'] > 0.5]
df = df[df['trip_distance'] < 100]

df = df[~df.select_dtypes('number').lt(0).any(axis=1)]
```

```
df.select_dtypes(include=['number']).agg([ 'min', 'max','mean']).T
```

	min	max	mean
VendorID	1.00	6.00	1.747906
passenger_count	0.00	9.00	1.337836
trip_distance	0.51	99.76	3.686328
RatecodeID	1.00	99.00	7.651442
PULocationID	1.00	265.00	164.975514
DOLocationID	1.00	265.00	163.595329
payment_type	0.00	4.00	1.115288
fare_amount	0.00	1025.00	20.360364
extra	0.00	14.25	1.610381
mta_tax	0.00	5.75	0.496961
tip_amount	0.00	280.00	3.681055
tolls_amount	0.00	94.75	0.620149
improvement_surcharge	0.00	1.00	0.999241
total_amount	0.00	1026.50	29.641226
congestion_surcharge	0.00	2.75	2.179459
airport_fee	0.00	1.75	0.136827

```
df = df[df['total_amount'] > 3.7]

df.select_dtypes(include=['number']).agg([ 'min', 'max','mean']).T
```

	min	max	mean
VendorID	1.00	6.00	1.747953
passenger_count	0.00	9.00	1.337851
trip_distance	0.51	99.76	3.686093
RatecodeID	1.00	99.00	7.649950
PULocationID	1.00	265.00	164.976648
DOLocationID	1.00	265.00	163.595775
payment_type	0.00	4.00	1.115181
fare_amount	0.00	1025.00	20.362166
extra	0.00	14.25	1.610515
mta_tax	0.00	5.75	0.496997
tip_amount	0.00	280.00	3.681384
tolls_amount	0.00	94.75	0.620204
improvement_surcharge	0.00	1.00	0.999271
total_amount	3.98	1026.50	29.643776
congestion_surcharge	0.00	2.75	2.179649
airport_fee	0.00	1.75	0.136829

```
df = df[df['passenger_count'] >= 1]
```

```
df.select_dtypes(include=['number']).agg([ 'min', 'max','mean','median','std']
```

	min	max	mean	median	std
VendorID	1.00	6.00	1.759152	2.00	0.441703
passenger_count	1.00	9.00	1.357890	1.00	0.856223
trip_distance	0.51	99.76	3.695883	1.94	4.585362
RatecodeID	1.00	99.00	7.748972	1.00	24.706680
PULocationID	1.00	265.00	164.944502	162.00	64.111443
DOLocationID	1.00	265.00	163.578450	162.00	69.923377
payment_type	0.00	4.00	1.113693	1.00	0.514395
fare_amount	0.00	1025.00	20.398935	14.20	17.795989
extra	0.00	14.25	1.583503	1.00	1.840527
mta_tax	0.00	5.75	0.496983	0.50	0.039211
tip_amount	0.00	280.00	3.688199	3.00	4.006492
tolls_amount	0.00	87.50	0.623581	0.00	2.208256
improvement_surcharge	0.00	1.00	0.999264	1.00	0.022974
total_amount	3.98	1026.50	29.691756	21.84	22.455239
congestion_surcharge	0.00	2.75	2.176726	2.50	0.838856
airport_fee	0.00	1.75	0.137468	0.00	0.457917

```
df['pickup_hour'] = df['tpep_pickup_datetime'].dt.hour
df['pickup_day'] = df['tpep_pickup_datetime'].dt.day
df['pickup_weekend'] = df['tpep_pickup_datetime'].dt.weekday >= 4 # Friday=4,
df['pickup_month'] = df['tpep_pickup_datetime'].dt.month
df['pickup_year'] = df['tpep_pickup_datetime'].dt.year
df['dropoff_hour'] = df['tpep_dropoff_datetime'].dt.hour
df['dropoff_day'] = df['tpep_dropoff_datetime'].dt.day
df['dropoff_weekend'] = df['tpep_dropoff_datetime'].dt.weekday >= 4 # Friday=4
df['dropoff_month'] = df['tpep_dropoff_datetime'].dt.month
df['dropoff_year'] = df['tpep_dropoff_datetime'].dt.year

df.head()
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
131072	2	2023-01-12 09:58:04	2023-01-12 10:13:08	1.0
131073	2	2023-01-12 09:07:47	2023-01-12 09:36:28	1.0
131074	2	2023-01-12 09:39:29	2023-01-12 09:58:25	1.0
131075	2	2023-01-12 09:14:18	2023-01-12 09:29:54	1.0
131076	2	2023-01-12 09:41:56	2023-01-12 09:59:12	1.0

5 rows × 28 columns

```
df['trip_duration_min'] = (
    (df['tpep_dropoff_datetime'] - df['tpep_pickup_datetime'])
    .dt.total_seconds() / 60
)

df = df[df['trip_duration_min'] > 0]
```

```
df['fare_per_mile'] = df['fare_amount'] / df['trip_distance']
```

```
df['tip_percent'] = df['tip_amount'] / df['fare_amount']
```

```
df.groupby('pickup_weekend')['total_amount'].mean()
```

	total_amount
pickup_weekend	
False	29.972848
True	29.212195

dtype: float64

```
df.groupby('pickup_weekend')['total_amount'].median()
```

	total_amount
pickup_weekend	
False	22.2
True	21.3

dtype: float64

```
df['calculated_total'] = (
    df['fare_amount'] +
    df['extra'] +
    df['mta_tax'] +
    df['tip_amount'] +
    df['tolls_amount'] +
    df['improvement_surcharge'] +
    df['congestion_surcharge'] +
    df['airport_fee']
)

df['total_diff'] = df['total_amount'] - df['calculated_total']
```

```
df['total_diff'].abs().describe().round(2)
```

	total_diff
count	1609793.00
mean	0.70
std	1.15
min	0.00
25%	0.00
50%	0.00
75%	2.50
max	4.50

dtype: float64

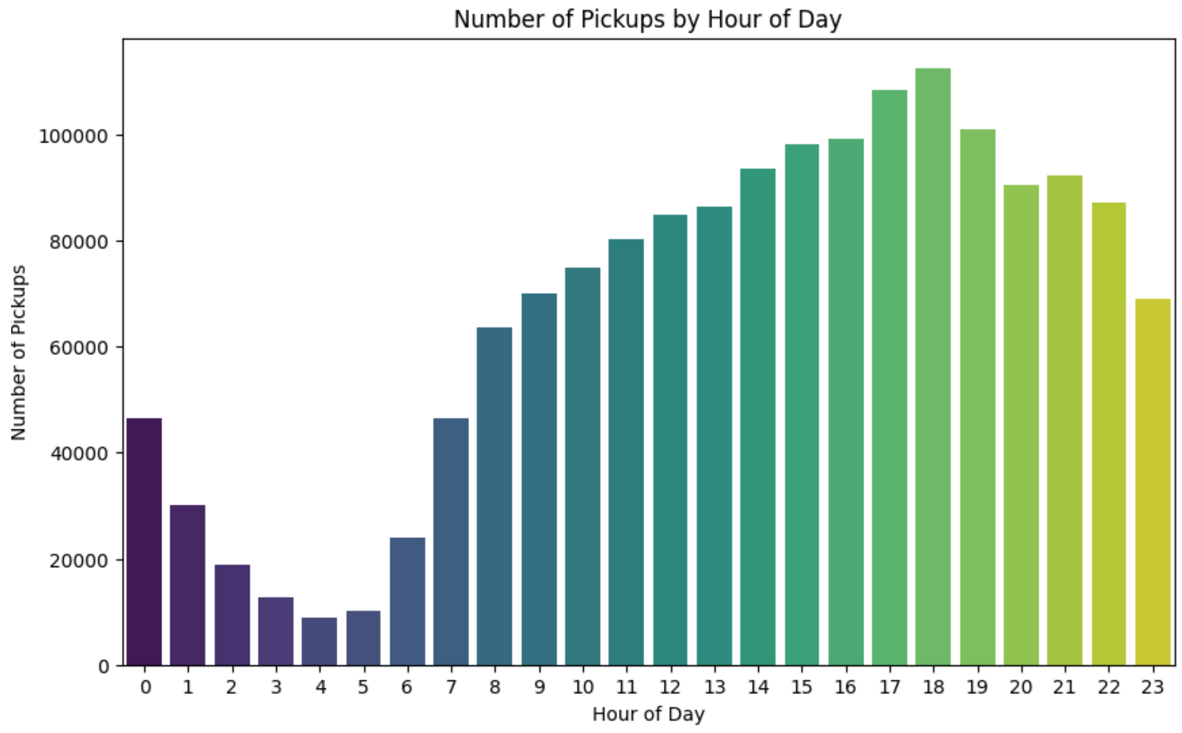
```
df.head()
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
131072	2	2023-01-12 09:58:04	2023-01-12 10:13:08	1.0
131073	2	2023-01-12 09:07:47	2023-01-12 09:36:28	1.0
131074	2	2023-01-12 09:39:29	2023-01-12 09:58:25	1.0
131075	2	2023-01-12 09:14:18	2023-01-12 09:29:54	1.0
131076	2	2023-01-12 09:41:56	2023-01-12 09:59:12	1.0

5 rows × 33 columns

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 6))
sns.countplot(x='pickup_hour', data=df, palette='viridis')
plt.title('Number of Pickups by Hour of Day')
plt.xlabel('Hour of Day')
plt.ylabel('Number of Pickups')
plt.xticks(range(0, 24))
plt.show()
```

```
tmp/ipython-input-1569673394.py:5: FutureWarning:

assing `palette` without assigning `hue` is deprecated and will be removed in v

 sns.countplot(x='pickup_hour', data=df, palette='viridis')
```



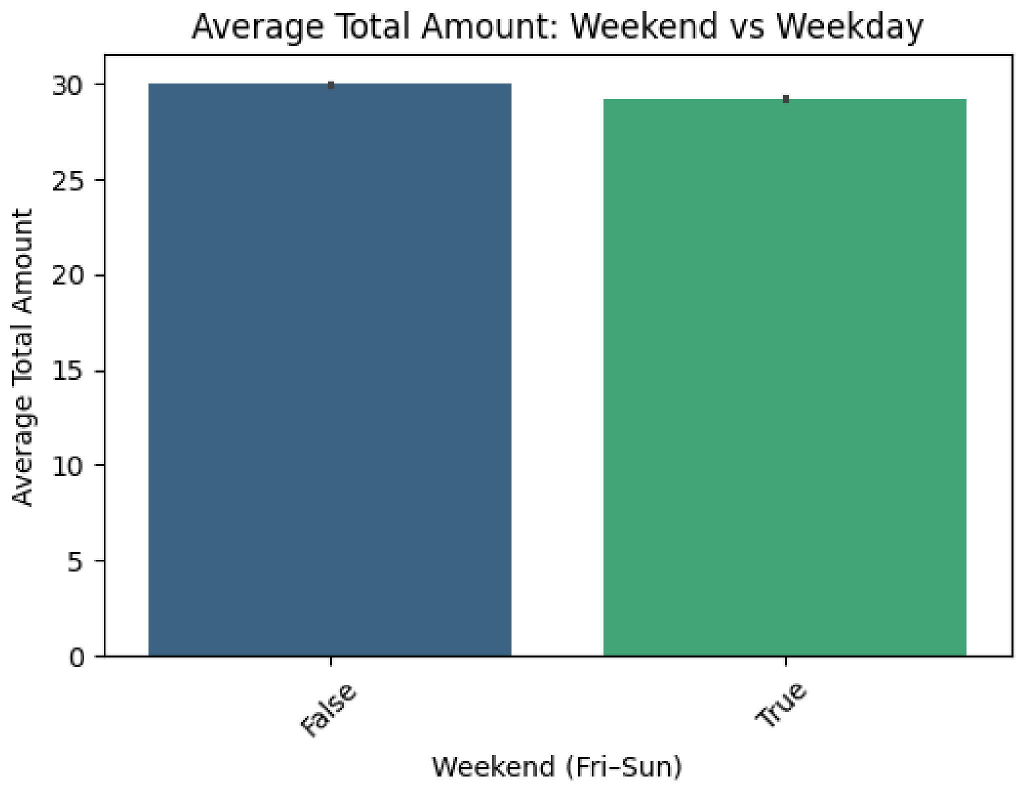
```
df.groupby('pickup_weekend')['total_amount'].mean().round(2)
```

	total_amount
pickup_weekend	
False	29.97
True	29.21

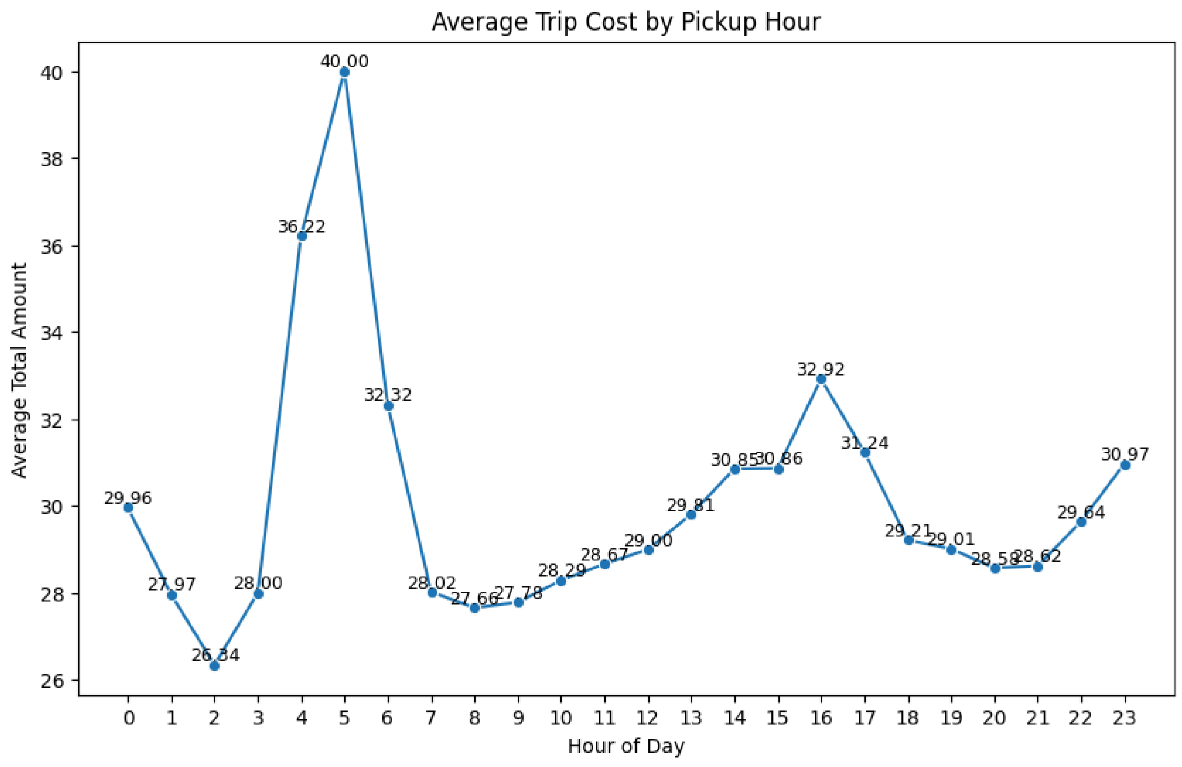
dtype: float64

```
plt.figure(figsize=(6, 4))
sns.barplot(x='pickup_weekend', y='total_amount', data=df, palette='viridis')
plt.xticks(rotation=45)
plt.title('Average Total Amount: Weekend vs Weekday')
plt.xlabel('Weekend (Fri-Sun)')
plt.ylabel('Average Total Amount')
plt.show()
```

```
/tmp/ipython-input-1988818216.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v

  sns.barplot(x='pickup_weekend', y='total_amount', data=df, palette='viridis')
```

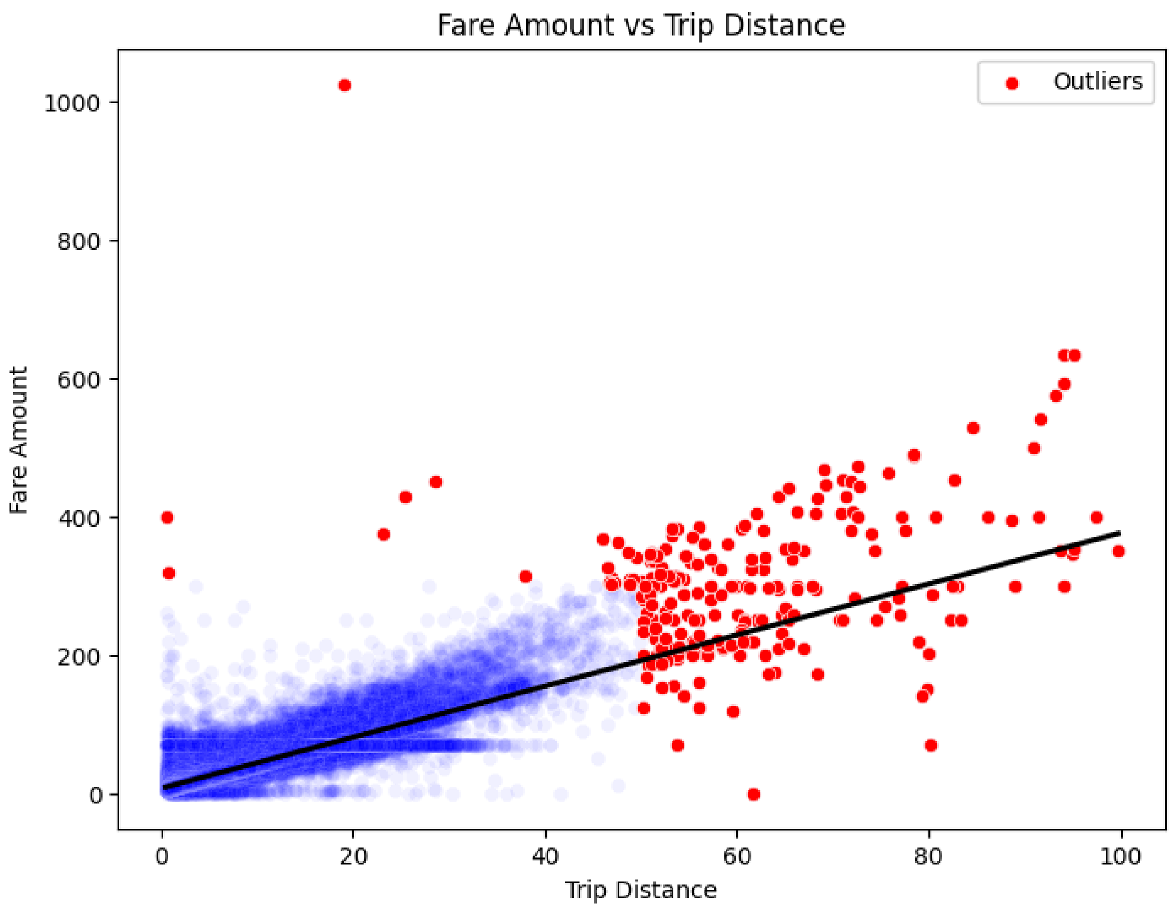


```
hourly_avg = (
    df.groupby('pickup_hour')['total_amount']
    .mean()
    .reset_index()
)

plt.figure(figsize=(10, 6))
sns.lineplot(
    x='pickup_hour',
    y='total_amount',
    data=hourly_avg,
    marker='o'
)

for _, row in hourly_avg.iterrows():
    plt.text(
        row['pickup_hour'],
        row['total_amount'],
        f"{row['total_amount']:.2f}",
        ha='center',
        va='bottom',
        fontsize=9
    )

plt.title('Average Trip Cost by Pickup Hour')
plt.xlabel('Hour of Day')
plt.ylabel('Average Total Amount')
plt.xticks(range(0, 24))
plt.show()
```



```
outliers = df[
    (df['trip_distance'] > 50) |
    (df['fare_amount'] > 300)
]

plt.figure(figsize=(8, 6))

# נקודות רגילות
sns.scatterplot(
    x='trip_distance',
    y='fare_amount',
    data=df,
    alpha=0.05,
    color='blue'
)

# חריגים
sns.scatterplot(
    x='trip_distance',
    y='fare_amount',
    data=outliers,
    color='red',
    label='Outliers'
)

# קו רגרסיה
sns.regplot(
    x='trip_distance',
    y='fare_amount',
    data=df,
    scatter=False,
    color='black'
)

plt.title('Fare Amount vs Trip Distance')
plt.xlabel('Trip Distance')
plt.ylabel('Fare Amount')
plt.legend()
plt.show()
```

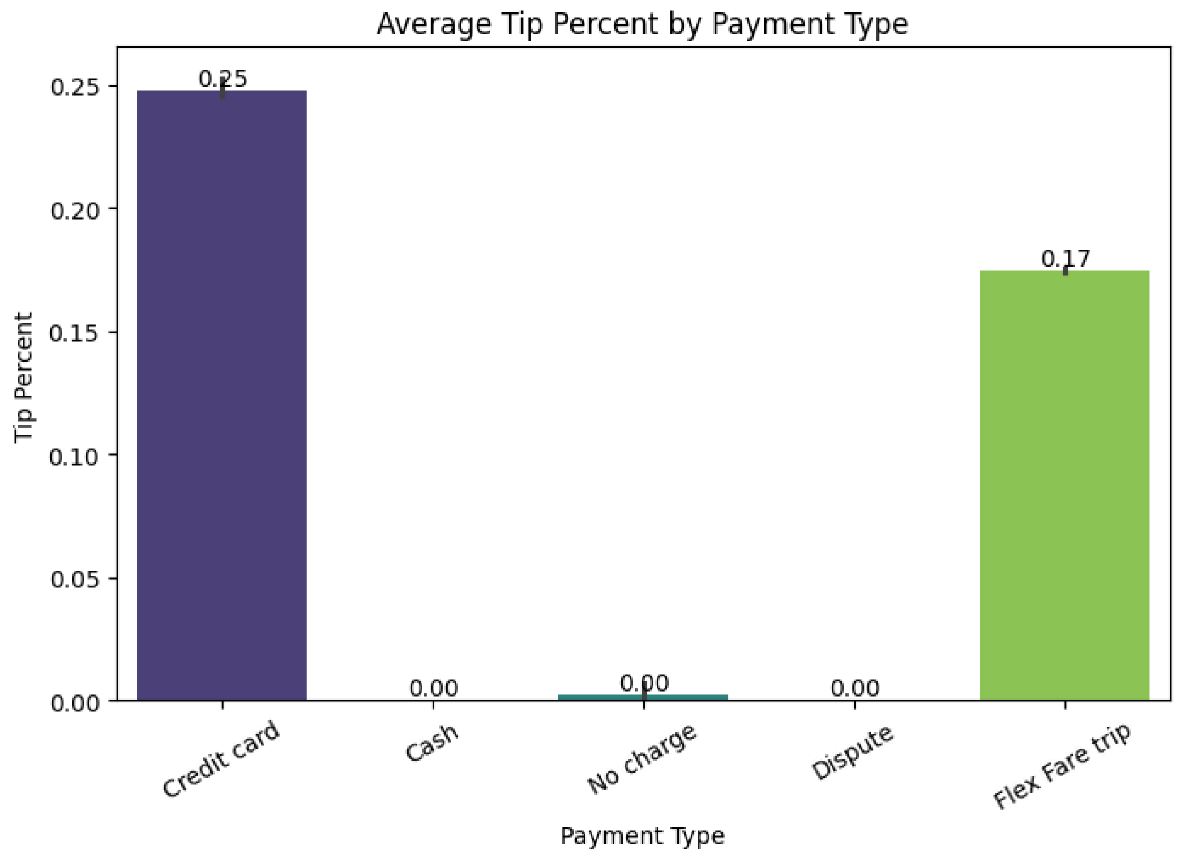
```
/usr/local/lib/python3.12/dist-packages/IPython/core/pylabtools.py:151: UserWar
  fig.canvas.print_figure(bytes_io, **kw)
```



```
payment_map = {
    0: 'Flex Fare trip',
    1: 'Credit card',
    2: 'Cash',
    3: 'No charge',
    4: 'Dispute',
    5: 'Unknown',
    6: 'Voided trip'
}

df['payment_type_label'] = df['payment_type'].map(payment_map)
```

```
plt.figure(figsize=(8, 5))

ax = sns.barplot(
    x='payment_type_label',
    y='tip_percent',
    data=df,
    palette='viridis'
)

# הוספת הערכים מעל העמודות
for p in ax.patches:
    value = p.get_height()
    ax.annotate(
        f"{value:.2f}",
        (p.get_x() + p.get_width() / 2., value),
        ha='center',
        va='bottom',
        fontsize=10
    )

plt.title('Average Tip Percent by Payment Type')
plt.xlabel('Payment Type')
plt.ylabel('Tip Percent')
plt.xticks(rotation=30)
plt.show()
```

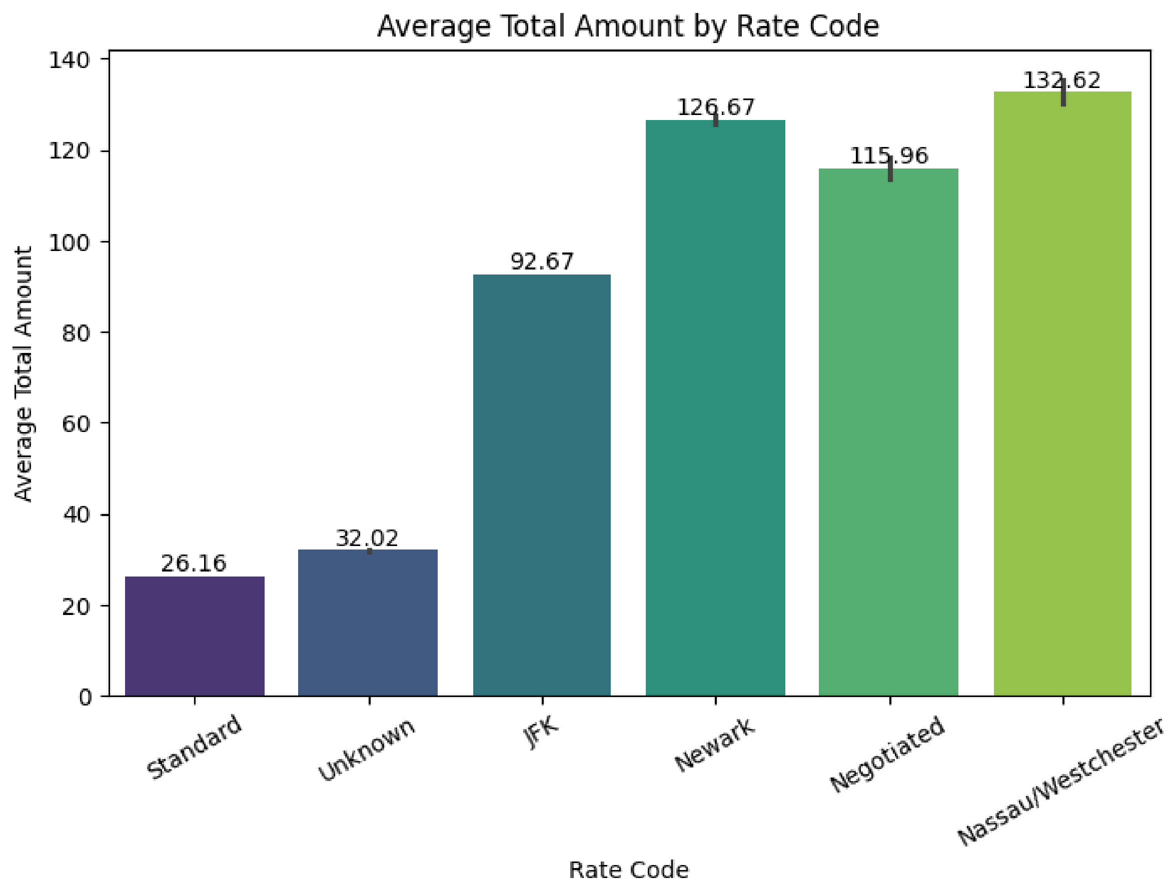
```
/tmp/ipython-input-1626809601.py:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v

  ax = sns.barplot(
```



```
ratecode_map = {
    1: 'Standard',
    2: 'JFK',
    3: 'Newark',
    4: 'Nassau/Westchester',
    5: 'Negotiated',
    6: 'Group ride',
    99: 'Unknown'
}

df['Ratecode_label'] = df['RatecodeID'].map(ratecode_map)


plt.figure(figsize=(8, 5))

ax = sns.barplot(
    x='Ratecode_label',
    y='total_amount',
    data=df,
    palette='viridis'
)

# הוספת הערכים מעל העמודות
```

```
for p in ax.patches:
    value = p.get_height()
    ax.annotate(
        f"{value:.2f}",
        (p.get_x() + p.get_width() / 2., value),
        ha='center',
        va='bottom',
        fontsize=10
    )

plt.title('Average Total Amount by Rate Code')
plt.xlabel('Rate Code')
plt.ylabel('Average Total Amount')
plt.xticks(rotation=30)
plt.show()
```

```
/tmp/ipython-input-2133174923.py:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v

  ax = sns.barplot(
```



```
df.groupby('Ratecode_label')['trip_distance'].mean().round(2)
```

Ratecode_label	trip_distance
JFK	18.25
Nassau/Westchester	20.46
Negotiated	15.50
Newark	18.06
Standard	2.89
Unknown	4.84

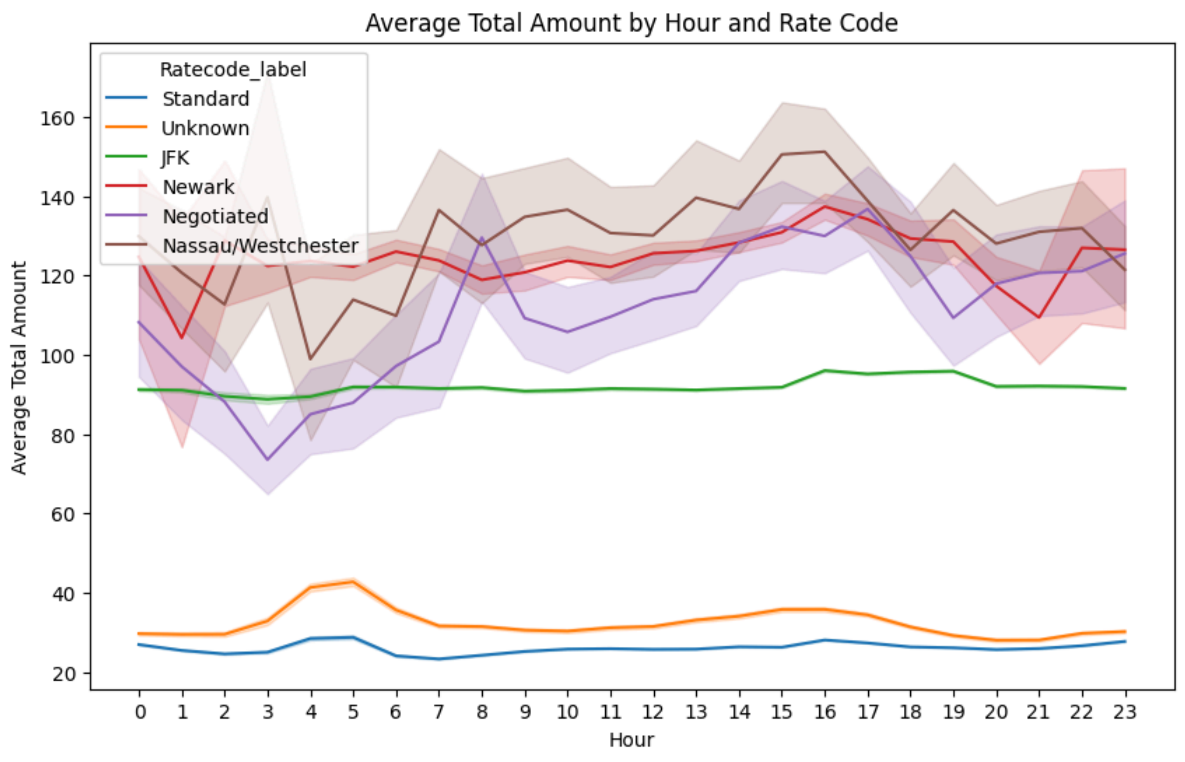
dtype: float64

```
df.groupby('Ratecode_label')['trip_duration_min'].mean().round(2)
```

Ratecode_label	trip_duration_min
JFK	52.79
Nassau/Westchester	42.49
Negotiated	40.80
Newark	44.31
Standard	16.41
Unknown	21.94

dtype: float64

```
plt.figure(figsize=(10, 6))
sns.lineplot(
    x='pickup_hour',
    y='total_amount',
    hue='Ratecode_label',
    data=df,
    estimator='mean'
)
plt.title('Average Total Amount by Hour and Rate Code')
plt.xlabel('Hour')
plt.ylabel('Average Total Amount')
plt.xticks(range(0, 24))
plt.show()
```



```
df.groupby(['Ratecode_label', 'pickup_weekend'])['total_amount'].mean().round(
```

		total_amount
Ratecode_label	pickup_weekend	
JFK	False	93.06
	True	92.05
Nassau/Westchester	False	131.29
	True	134.93
Negotiated	False	118.54
	True	112.37
Newark	False	127.22
	True	125.88
Standard	False	26.61
	True	25.37
Unknown	False	32.12
	True	31.89

dtype: float64

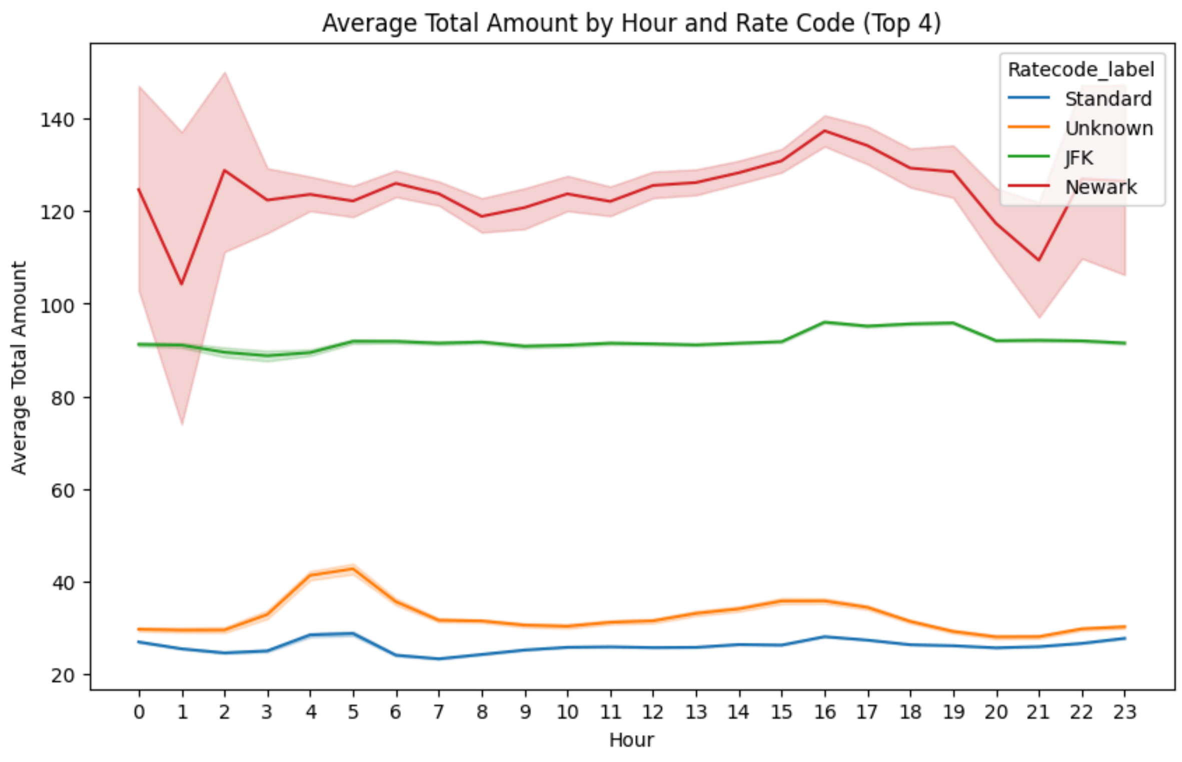
```
df[df['Ratecode_label'] == 'Unknown'].shape[0] / df.shape[0]
```

```
0.06818764897101677
```

```
df.groupby('Ratecode_label')[['trip_distance','trip_duration_min','total_amoun
```

	trip_distance	trip_duration_min	total_amount
Ratecode_label			
JFK	18.25	52.79	92.67
Nassau/Westchester	20.46	42.49	132.62
Negotiated	15.50	40.80	115.96
Newark	18.06	44.31	126.67
Standard	2.89	16.41	26.16
Unknown	4.84	21.94	32.02

```
top_ratecodes = df['Ratecode_label'].value_counts().head(4).index  # 4 נפוצים
df_top = df[df['Ratecode_label'].isin(top_ratecodes)]

plt.figure(figsize=(10,6))
sns.lineplot(x='pickup_hour', y='total_amount', hue='Ratecode_label', data=df_tc
plt.title('Average Total Amount by Hour and Rate Code (Top 4)')
plt.xlabel('Hour')
plt.ylabel('Average Total Amount')
plt.xticks(range(0,24))
plt.show()
```



```
df['total_diff_abs'] = df['total_diff'].abs()
df['total_diff_abs'].describe().round(2)
```

	total_diff_abs
count	1609793.00
mean	0.70
std	1.15
min	0.00
25%	0.00
50%	0.00
75%	2.50
max	4.50

dtype: float64

```
{
    "diff<=0.01": (df['total_diff_abs'] <= 0.01).mean(),
    "diff<=1.00": (df['total_diff_abs'] <= 1.00).mean(),
    "diff<=2.50": (df['total_diff_abs'] <= 2.50).mean(),
}
```

```
{'diff<=0.01': np.float64(0.7225450725652305),
 'diff<=1.00': np.float64(0.7225463149609919),
 'diff<=2.50': np.float64(0.9622821070783635)}
```

```
df['has_tip'] = df['tip_amount'] > 0
```

```
df['has_tip'].mean().round(3) * 100 # %
```

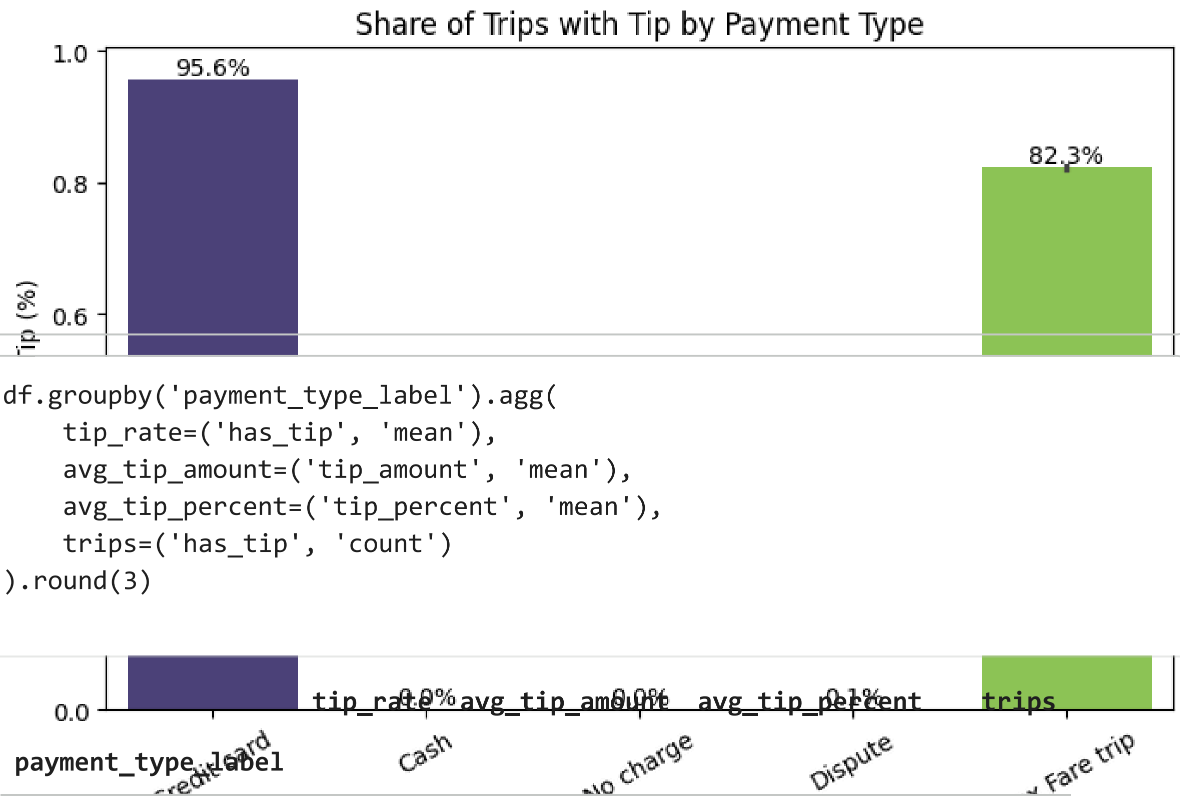
```
np.float64(79.2)
```

```
df.groupby('payment_type_label')['has_tip'].agg(
    trips='count',
    tip_rate='mean'
).round(3)
```

payment_type_label	trips	tip_rate
Cash	250384	0.000
Credit card	1244981	0.956
Dispute	8639	0.001
Flex Fare trip	101554	0.823
No charge	4235	0.000

```
plt.figure(figsize=(8,5))
ax = sns.barplot(
    x='payment_type_label',
    y='has_tip',
    data=df,
    palette='viridis'
)

for p in ax.patches:
    v = p.get_height()
    ax.annotate(
        f"{v*100:.1f}%",
        (p.get_x() + p.get_width()/2, v),
        ha='center',
        va='bottom',
        fontsize=10
    )

plt.title('Share of Trips with Tip by Payment Type')
plt.xlabel('Payment Type')
plt.ylabel('Trips with Tip (%)')
plt.xticks(rotation=30)
plt.show()
```

```
/tmp/ipython-input-1545293828.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v

  ax = sns.barplot(
```



```
df.groupby('payment_type_label').agg(
    tip_rate=('has_tip', 'mean'),
    avg_tip_amount=('tip_amount', 'mean'),
    avg_tip_percent=('tip_percent', 'mean'),
    trips=('has_tip', 'count')
).round(3)
```

	tip_rate	avg_tip_amount	avg_tip_percent	trips
payment_type_label				
Cash	0.000	0.000	0.000	250384
Credit card	0.956	4.462	inf	1244981
Dispute	0.001	0.003	0.000	8639
Flex Fare trip	0.823	3.770	inf	101554
No charge	0.000	0.024	0.003	4235

```
df.groupby('pickup_hour')['has_tip'].mean().round(3)
```

	has_tip
pickup_hour	
0	0.789
1	0.782
2	0.771
3	0.749
4	0.704
5	0.698
6	0.748
7	0.806
8	0.818
9	0.790
10	0.768
11	0.764
12	0.766
13	0.762
14	0.765
15	0.772
16	0.778
17	0.802
18	0.816
19	0.818
20	0.822
21	0.827
22	0.822
23	0.807

dtype: float64

```
df.groupby('pickup_weekend')['has_tip'].mean().round(3)
```

	has_tip
pickup_weekend	
False	0.800
True	0.778

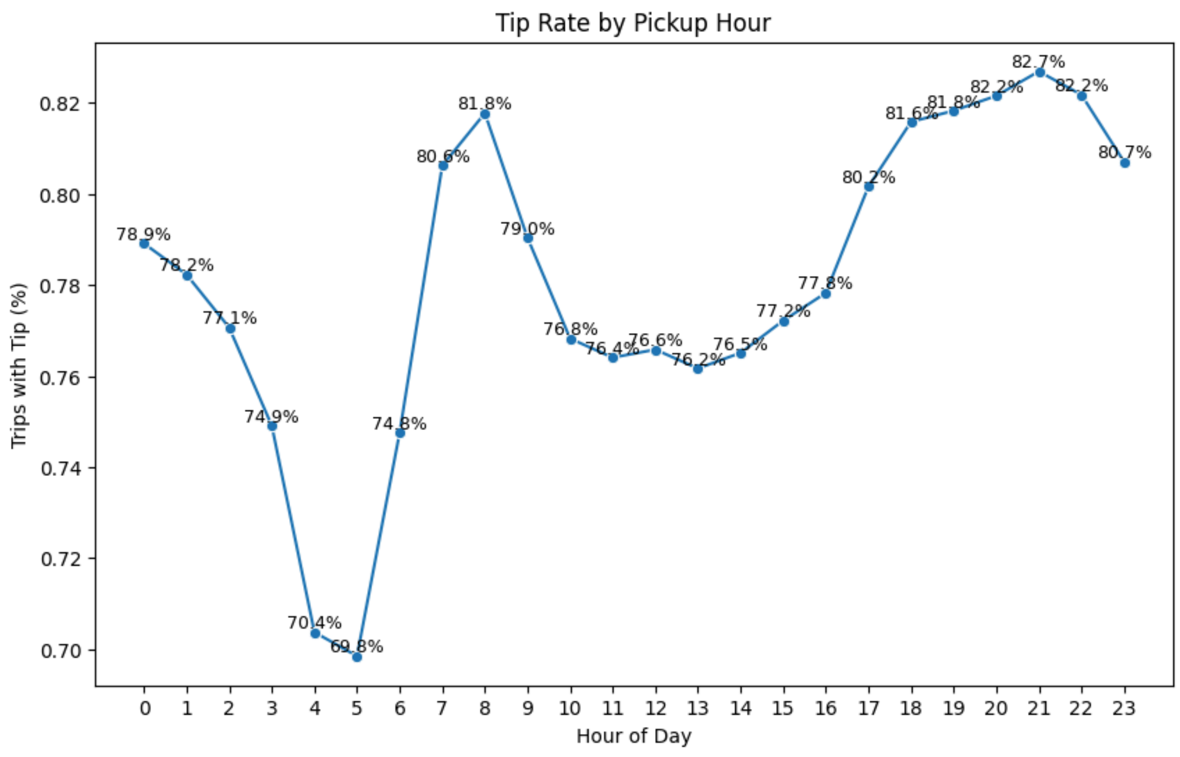
dtype: float64

```
df.groupby('Ratecode_label')['has_tip'].mean().round(3)
```

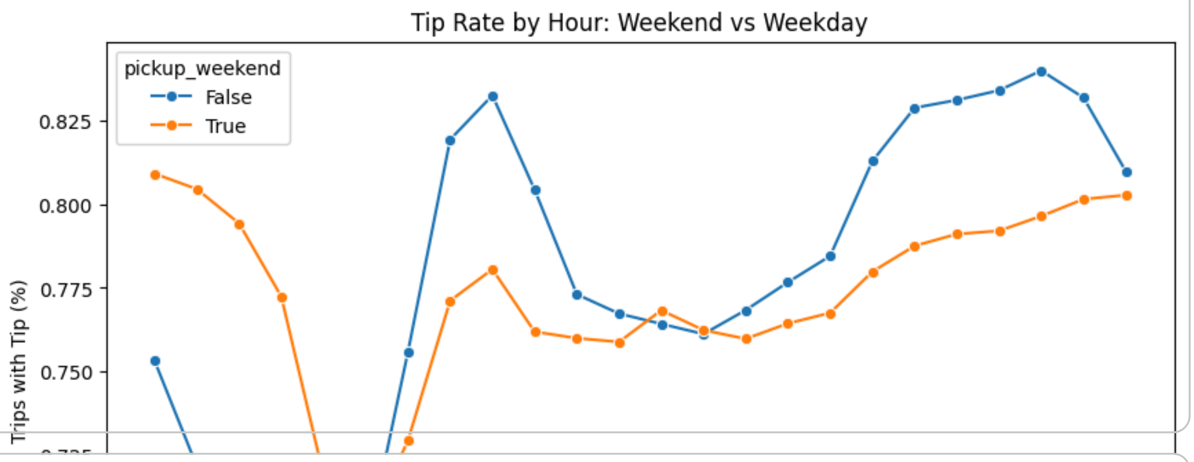
	has_tip
Ratecode_label	
JFK	0.761
Nassau/Westchester	0.614
Negotiated	0.663
Newark	0.753
Standard	0.796
Unknown	0.762

dtype: float64

```
plt.figure(figsize=(10,6))
sns.lineplot(x='pickup_hour', y='has_tip', data=tip_by_hour, marker='o')

for _, row in tip_by_hour.iterrows():
    plt.text(
        row['pickup_hour'],
        row['has_tip'],
        f"{row['has_tip']*100:.1f}%",
        ha='center',
        va='bottom',
        fontsize=9
    )

plt.title('Tip Rate by Pickup Hour')
plt.xlabel('Hour of Day')
plt.ylabel('Trips with Tip (%)')
plt.xticks(range(0,24))
plt.show()
```



```
tip_hour_weekend = (
    df.groupby(['pickup_hour','pickup_weekend'])['has_tip']
    .mean()
    .reset_index()
)

plt.figure(figsize=(10,6))
sns.lineplot(
    x='pickup_hour',
    y='has_tip',
    hue='pickup_weekend',
    data=tip_hour_weekend,
    marker='o'
)

plt.title('Tip Rate by Hour: Weekend vs Weekday')
plt.xlabel('Hour of Day')
plt.ylabel('Trips with Tip (%)')
plt.xticks(range(0,24))
plt.show()
```



```
driver_hourly = (
    df.groupby('pickup_hour')
    .agg(
        trips=('total_amount','count'),
        avg_fare=('total_amount','mean'),
        tip_rate=('has_tip','mean')
    )
    .round(3)
    .reset_index()
)

driver_hourly
```

	pickup_hour	trips	avg_fare	tip_rate
0	0	46517	29.960	0.789
1	1	30063	27.966	0.782
2	2	18875	26.344	0.771
3	3	12702	28.000	0.749
4	4	8985	36.222	0.704
5	5	10092	40.000	0.698
6	6	24055	32.324	0.748
7	7	46423	28.023	0.806
8	8	63742	27.660	0.818
9	9	69999	27.783	0.790
10	10	74901	28.291	0.768
11	11	80214	28.666	0.764
12	12	84896	28.999	0.766
13	13	86444	29.805	0.762
14	14	93597	30.854	0.765
15	15	98230	30.864	0.772
16	16	99100	32.921	0.778
17	17	108494	31.240	0.802
18	18	112565	29.214	0.816
19	19	101032	29.013	0.818
20	20	90538	28.576	0.822
21	21	92235	28.620	0.827
22	22	87121	29.640	0.822
23	23	68973	30.967	0.807

```
driver_weekend = (
    df.groupby(['pickup_hour','pickup_weekend'])
    .agg(
        trips=('total_amount','count'),
        avg_fare=('total_amount','mean'),
        tip_rate=('has_tip','mean')
    )
    .round(3)
    .reset_index()
)

driver_weekend
```

```
df['fare_per_mile'] = df['fare_amount'] / df['trip_distance']

suspicious_short_expensive = df[
    (df['trip_distance'] < 1) &
    (df['fare_amount'] > df['fare_amount'].quantile(0.99))
]

suspicious_short_expensive[['trip_distance','fare_amount','fare_per_mile']].de
```

	trip_distance	fare_amount	fare_per_mile
count	51.00	51.00	51.00
mean	0.70	117.51	175.11
std	0.12	67.08	112.60
min	0.51	78.00	96.67
25%	0.60	84.15	111.92
50%	0.70	87.30	131.62
75%	0.80	110.00	174.07
max	0.96	400.00	666.67

```
threshold = df['fare_per_mile'].quantile(0.995)

suspicious_fpm = df[df['fare_per_mile'] > threshold]
```

```python
suspicious_fpm[['trip_distance','fare_amount','fare_per_mile']]
```

	trip_distance	fare_amount	fare_per_mile
131785	0.60	15.5	25.833333
131873	4.00	82.0	20.500000
132002	0.88	17.0	19.318182
132537	0.79	15.6	19.746835
133493	0.81	17.0	20.987654
...	...	...	...
4256995	1.60	70.0	43.750000
4257067	0.75	17.7	23.600000
4257077	0.54	10.7	19.814815
4257192	0.66	12.8	19.393939
4258617	0.73	65.0	89.041096

8048 rows × 3 columns

```python
suspicious_fast = df[
    (df['trip_duration_min'] < 5) &
    (df['total_amount'] > df['total_amount'].quantile(0.99))
]

suspicious_fast[['trip_duration_min','total_amount']].describe().round(2)
```

	trip_duration_min	total_amount
count	71.00	71.00
mean	1.50	152.83
std	1.44	68.87
min	0.03	104.33
25%	0.38	107.33
50%	0.80	121.10
75%	2.41	181.62
max	4.75	481.20

1	0	True	29881	28.178	0.809
					0.722
3	1	True	21998	26.363	0.805
					0.688
5	2	True	14666	25.241	0.794
					0.667
7	3	True	9891	26.684	0.772
					0.675
9	4	True	5896	32.415	0.719
					0.696
11	5	True	4072	41.932	0.702
					0.756
13	6	True	7557	35.914	0.729
14	7	False	33767	27.009	0.819
20	10	False	47297	28.493	0.773
21	10	True	27604	27.946	0.760
22	11	False	[illegible]	29.200	0.767
23	11	True	30827	27.810	0.759
24	12	False	[illegible]	29.421	0.764
25	12	True	33098	28.337	0.768
26	13	False	[illegible]	30.241	0.761
27	13	True	34218	29.141	0.762
28	14	False	[illegible]	31.240	0.768
29	14	True	35600	30.224	0.760

```python
from sklearn.linear_model import LinearRegression

X = df[['trip_distance']]
y = df['fare_amount']
```

```
model = LinearRegression()
model.fit(X, y)

df['fare_pred'] = model.predict(X)
df['fare_residual'] = df['fare_amount'] - df['fare_pred']

# חריגים חיוביים קיצוניים
suspicious_residuals = df[df['fare_residual'] > df['fare_residual'].quantile(0
suspicious_residuals
```

```
39    19    True   32402   28.151   0.791
40    20    False  63671   28.314   0.834
42    21    False  64654   28.287   0.840
44    22    False  58218   29.433   0.832
46    23    False  41675   31.244   0.810
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
131873	1	2023-01-12 10:45:48	2023-01-12 10:46:03	1.0
132088	2	2023-01-12 11:44:10	2023-01-12 12:22:20	1.0
132240	2	2023-01-12 11:37:09	2023-01-12 12:12:50	2.0
132509	2	2023-01-12 11:32:36	2023-01-12 12:17:04	1.0
132546	2	2023-01-12 11:36:32	2023-01-12 11:59:56	1.0
...	...	...	...	...
4258817	2	2023-11-09 23:48:33	2023-11-10 01:15:19	1.0
4259354	2	2023-11-09 23:52:51	2023-11-10 00:27:32	1.0
4259357	2	2023-11-09 23:23:11	2023-11-09 23:55:39	1.0
4259442	2	2023-11-10 00:29:42	2023-11-10 01:11:39	1.0
4259783	2	2023-11-10 00:17:23	2023-11-10 00:54:16	1.0

8049 rows × 39 columns

```
suspicious_tips = df[
    (df['tip_percent'] > 0.6) &
    (df['total_amount'] < 20)
]

suspicious_tips
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
131555	2	2023-01-12 10:35:53	2023-01-12 10:41:02	2.0
133487	1	2023-01-12 13:34:53	2023-01-12 13:41:24	2.0
133731	1	2023-01-12 13:57:45	2023-01-12 14:06:08	1.0
134574	1	2023-01-12 14:14:14	2023-01-12 14:21:26	1.0
4257273	2	2023-11-09 21:35:36	2023-11-09 21:37:44	1.0
4259047	2	2023-11-09 23:48:36	2023-11-09 23:53:56	1.0

```
df[
    (df['pickup_hour'].between(3,5)) &
    (df['fare_per_mile'] > df['fare_per_mile'].quantile(0.99))
]
```

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
158774	2	2023-01-14 04:33:29	2023-01-14 04:40:09	2.0
299529	1	2023-01-25 05:10:50	2023-01-25 05:19:38	1.0
299535	1	2023-01-25 05:19:37	2023-01-25 05:27:06	1.0
313368	2	2023-01-26 05:43:57	2023-01-26 05:44:13	1.0
378428	2	2023-01-31 04:59:43	2023-01-31 05:02:19	1.0
...	...	...	...	...
4122368	1	2023-10-29 04:02:18	2023-10-29 04:09:14	1.0
4122384	2	2023-10-29 04:19:31	2023-10-29 04:30:45	1.0
4124611	2	2023-11-01 03:08:57	2023-11-01 03:14:00	1.0
4124760	1	2023-11-01 04:01:03	2023-11-01 04:29:48	1.0
4243767	1	2023-11-09 05:22:57	2023-11-09 05:48:53	1.0

143 rows × 39 columns