Import libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sqlalchemy as sa
import plotly.graph_objects as go
import plotly.express as px
```

Start coding or generate with AI.

Connect to database

```
metrocar_url = "postgresql://xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
```

```
engine = sa.create_engine(metrocar_url)
connection = engine.connect().execution_options(isolation_level="AUTOCOMMIT")
```

Start coding or generate with AI.

Import all tables and read as Pandas dataframe

```
# All tables were imported and read with panda as data frame
transactions = pd.read_sql("SELECT * FROM transactions",connection)
signups = pd.read_sql("SELECT * FROM signups",connection)
ride_requests = pd.read_sql("SELECT * FROM ride_requests",connection)
reviews = pd.read_sql("SELECT * FROM reviews",connection)
app_downloads = pd.read_sql("SELECT * FROM app_downloads",connection)
```

```
# Check data
reviews.head()
```

	review_id	ride_id	user_id	driver_id	rating	review
0	50000	3000002	112008	101175	1	Horrible service. The driver was reckless and ...
1	50001	3000004	101504	106914	5	Metrocar's customer service is top-notch. I ha...
2	50002	3000005	116115	113371	5	Metrocar never disappoints. Whether it's a sho...
3	50003	3000006	102429	107336	5	Metrocar never disappoints. Whether it's a sho...

Start coding or generate with AI.

Reviews

```
reviews['rating_given'] = reviews['rating'].notna()
reviews_funnel = reviews.loc[:, ['rating_given']].sum()
#reviews_funnel
reviews.head()
```

	review_id	ride_id	user_id	driver_id	rating	review	rating_given
0	50000	3000002	112008	101175	1	Horrible service. The driver was reckless and ...	True
1	50001	3000004	101504	106914	5	Metrocar's customer service is top-notch. I ha...	True
2	50002	3000005	116115	113371	5	Metrocar never disappoints. Whether it's a sho...	True
3	50003	3000006	102429	107336	5	Metrocar never disappoints. Whether it's a sho...	True

Start coding or generate with AI.

```
reviews_by_user = reviews.groupby('user_id')[['rating_given']].any()
reviews_by_user.head()
```

user_id	rating_given
100002	True
100004	True
100007	True
100008	True

Start coding or generate with AI.

Transactions

```
# Create a new column 'payment_approved' using boolean masking
transactions['payment_approved'] = transactions['charge_status'] == 'Approved'

transactions.head()
```

	transaction_id	ride_id	purchase_amount_usd	charge_status	transaction_ts	payment_approved
0	10000000	3000000	13.55	Approved	2021-03-28 19:11:00	True
1	10000001	3000001	27.77	Approved	2021-11-10 16:59:00	True
2	10000002	3000002	21.84	Approved	2021-09-08 21:03:00	True
3	10000003	3000004	26.86	Approved	2021-05-28 09:21:00	True

Start coding or generate with AI.

```
# Combining reviews and transactions
transactions_reviews = pd.merge(transactions,
                reviews,
                how = 'left',
                on = 'ride_id')

transactions_reviews[['payment_approved', 'rating_given']] = transactions_reviews[['payment_approved', 'rating_given']].fill
transactions_reviews.head()
```

	transaction_id	ride_id	purchase_amount_usd	charge_status	transaction_ts	payment_approved	review_id	user_id	driv
0	10000000	3000000	13.55	Approved	2021-03-28 19:11:00	True	NaN	NaN	
1	10000001	3000001	27.77	Approved	2021-11-10 16:59:00	True	NaN	NaN	
2	10000002	3000002	21.84	Approved	2021-09-08 21:03:00	True	50000.0	112008.0	10
3	10000003	3000004	26.86	Approved	2021-05-28 09:21:00	True	50001.0	101504.0	10
4	10000004	3000005	21.72	Approved	2021-12-01 16:59:00	True	50002.0	116115.0	11

Start coding or generate with AI.

```
# Creating payment funnel
payment_funnel = transactions_reviews.loc[:,['payment_approved', 'rating_given']].sum()
payment_funnel
```

	0
payment_approved	212628
rating_given	156211

Start coding or generate with AI.

Rides (requests, acceptance, completion)

```
ride_requests["ride_requested"] = ride_requests['request_ts'].notna()
ride_requests["ride_accepted"] = ride_requests['accept_ts'].notna()
ride_requests["ride_completed"] = ride_requests['dropoff_ts'].notna()

ride_funnel = ride_requests.loc[:,['ride_requested','ride_accepted','ride_completed']].sum()
ride_funnel
```

	0
ride_requested	385477
ride_accepted	248379
ride_completed	223652

Start coding or generate with AI.

```
# Combining transactions_reviews and ride_requests
ride_transactions_reviews = pd.merge(ride_requests,
                transactions_reviews,
                how = 'left',
                on = 'ride_id')
ride_transactions_reviews[['ride_requested','ride_accepted','ride_completed','payment_approved', 'rating_given']] = ride_tra
ride_transactions_reviews.head()
```

	ride_id	user_id_x	driver_id_x	request_ts	accept_ts	pickup_location	dropoff_location	pickup_ts	dropoff_ts	cancel
0	3359548	112173	113668.0	2021-11-09 08:45:00	2021-11-09 08:51:00	40.67070005 -74.08670696	40.82356701 -73.85050356	2021-11-09 09:05:00	2021-11-09 09:21:00	
1	3359549	117478	114207.0	2022-03-07 09:03:00	2022-03-07 09:11:00	40.68788763 -74.05141857	40.77103753 -73.86478068	2022-03-07 09:17:00	2022-03-07 10:34:00	
2	3359550	116402	NaN	2022-01-23 08:26:00	NaT	40.79087235 -73.82275652	40.79635637 -74.11758388	NaT	NaT	2022-0 08:4
3	3359551	102510	NaN	2021-04-02 18:34:00	NaT	40.78786348 -74.06419292	40.8219556 -73.92284677	NaT	NaT	2021-0 18:3
4	3359552	113434	NaN	2021-12-01 16:11:00	NaT	40.82568853 -73.99340635	40.86128377 -73.83059678	NaT	NaT	2021-1 16:2

5 rows × 24 columns

Start coding or generate with AI.

Creating Full Ride Funnel

```
ride_funnel = ride_transactions_reviews.loc[:,['ride_requested','ride_accepted','ride_completed', 'payment_approved','rating
ride_funnel
```

	0
ride_requested	385477
ride_accepted	248379
ride_completed	223652
payment_approved	212628
rating_given	156211

Start coding or generate with AI.

Double-click (or enter) to edit

```
# Combining reviews and ride_requests, user level
ride_requests_reviews = pd.merge(ride_requests,
                reviews_by_user,
                how = 'left',
                on = 'user_id')
ride_requests_reviews.head()
```

	ride_id	user_id	driver_id	request_ts	accept_ts	pickup_location	dropoff_location	pickup_ts	dropoff_ts	cancel_ts
0	3359548	112173	113668.0	2021-11-09 08:45:00	2021-11-09 08:51:00	40.67070005 -74.08670696	40.82356701 -73.85050356	2021-11-09 09:05:00	2021-11-09 09:21:00	NaT
1	3359549	117478	114207.0	2022-03-07 09:03:00	2022-03-07 09:11:00	40.68788763 -74.05141857	40.77103753 -73.86478068	2022-03-07 09:17:00	2022-03-07 10:34:00	NaT
2	3359550	116402	NaN	2022-01-23 08:26:00	NaT	40.79087235 -73.82275652	40.79635637 -74.11758388	NaT	NaT	2022-01-23 08:42:00
3	3359551	102510	NaN	2021-04-02 18:34:00	NaT	40.78786348 -74.06419292	40.8219556 -73.92284677	NaT	NaT	2021-04-02 18:37:00
4	3359552	113434	NaN	2021-12-01 16:11:00	NaT	40.82568853 -73.99340635	40.86128377 -73.83059678	NaT	NaT	2021-12-01 16:29:00

Start coding or generate with AI.

```
ride_requests_reviews[['ride_requested','ride_accepted','ride_completed', 'rating_given']] = ride_requests_reviews[['ride_re
ride_requests_reviews.head()
```

	ride_id	user_id	driver_id	request_ts	accept_ts	pickup_location	dropoff_location	pickup_ts	dropoff_ts	cancel_ts
0	3359548	112173	113668.0	2021-11-09 08:45:00	2021-11-09 08:51:00	40.67070005 -74.08670696	40.82356701 -73.85050356	2021-11-09 09:05:00	2021-11-09 09:21:00	NaT
1	3359549	117478	114207.0	2022-03-07 09:03:00	2022-03-07 09:11:00	40.68788763 -74.05141857	40.77103753 -73.86478068	2022-03-07 09:17:00	2022-03-07 10:34:00	NaT
2	3359550	116402	NaN	2022-01-23 08:26:00	NaT	40.79087235 -73.82275652	40.79635637 -74.11758388	NaT	NaT	2022-01-23 08:42:00
3	3359551	102510	NaN	2021-04-02 18:34:00	NaT	40.78786348 -74.06419292	40.8219556 -73.92284677	NaT	NaT	2021-04-02 18:37:00
4	3359552	113434	NaN	2021-12-01 16:11:00	NaT	40.82568853 -73.99340635	40.86128377 -73.83059678	NaT	NaT	2021-12-01 16:29:00

Start coding or generate with AI.

```
# Rides by user
rides_by_user = ride_requests_reviews.groupby("user_id")[['ride_requested','ride_accepted','ride_completed', 'rating_given']
rides_by_user.head()
```

user_id	ride_requested	ride_accepted	ride_completed	rating_given
100000	True	True	False	False
100001	True	True	False	False
100002	True	True	True	True
100004	True	True	True	True

Start coding or generate with AI.

```
# Creating rides by user funnel
user_funnel = rides_by_user.loc[:,['ride_requested','ride_accepted','ride_completed', 'rating_given']].sum()
user_funnel
```

	0
ride_requested	12406
ride_accepted	12278
ride_completed	6233
rating_given	4348

Start coding or generate with AI.

Signups

```
signups["user_signup"] = signups['signup_ts'].notna()
```

```
# combining rides_by_user and signups
signups_rides_reviews = pd.merge(signups, rides_by_user,
                how = 'left',
                on = 'user_id')
signups_rides_reviews[['user_signup','ride_requested','ride_accepted','ride_completed', 'rating_given']] = signups_rides_rev
signups_rides_reviews.head()
```

	user_id	session_id	signup_ts	age_range	user_signup	ride_requested	ride_accepted	ride_complete
0	100001	58bec37ab818df39219ee36c124a1de9	2021-01-01 19:14:44	25-34	True	True	True	Fals
1	100002	c320ac72fe5e8cbfcf58458c36213ba5	2021-01-01 11:49:58	35-44	True	True	True	Tru
2	100003	ea11fb90284aa1c06933805f43c3e87a	2021-01-01 18:34:14	25-34	True	False	False	Fals

Start coding or generate with AI.

```
# Creating Signup Funnel
signup_funnel = signups_rides_reviews.loc[:,['user_signup','ride_requested','ride_accepted','ride_completed', 'rating_given'
signup_funnel
```

	0
user_signup	17623
ride_requested	12406
ride_accepted	12278
ride_completed	6233
rating_given	4348

Start coding or generate with AI.

App Downloads

```
app_downloads["user_downloaded"] = app_downloads['download_ts'].notna()
app_downloads
```

	app_download_key	platform	download_ts	user_downloaded
0	06f49bcc6895f888eba41043f95348ba	android	2021-05-13 13:12:06	True
1	60d79d5ac63159a5dffc13e42d87e070	android	2021-01-17 17:40:24	True
2	a3e52e50d379c3da808c4d8864f0d996	android	2021-01-11 04:02:52	True
3	2f7551cdd9a0a658350394e51bc74de3	android	2021-07-22 21:00:34	True
4	81adc238826a8dce8a706c083abc095e	web	2021-04-24 01:43:54	True
...	...	...	...	...
23603	2c1cb16bcc24ba46f8f90ef16ae70c4f	ios	2021-11-15 05:26:25	True
23604	2ad17e668a2622de46395ae228843ad8	ios	2021-08-02 17:01:05	True
23605	9a590716fc8ef31a11da8234a582dfed	ios	2021-06-12 00:41:13	True
23606	7e422317ed36f5424ebf6d8b728dd84c	ios	2021-04-18 00:39:40	True
23607	c68c776cf294c432991f3115bb622318	ios	2021-06-24 02:52:06	True

Start coding or generate with AI.

```
signups_rides_reviews.head()
```

	user_id	session_id	signup_ts	age_range	user_signup	ride_requested	ride_accepted	ride_complete
0	100001	58bec37ab818df39219ee36c124a1de9	2021-01-01 19:14:44	25-34	True	True	True	Fals
1	100002	c320ac72fe5e8cbfcf58458c36213ba5	2021-01-01 11:49:58	35-44	True	True	True	Tru
2	100003	ea11fb90284aa1c06933805f43c3e87a	2021-01-01 18:34:14	25-34	True	False	False	Fals

Start coding or generate with AI.

```
# combining signups_rides_reviews and downloads
downloads_to_reviews = pd.merge(app_downloads, signups_rides_reviews,
                how = 'left',
                left_on="app_download_key",
                 right_on="session_id")
```

Start coding or generate with AI.

```
downloads_to_reviews[['user_downloaded', 'user_signup','ride_requested','ride_accepted','ride_completed', 'rating_given']] =
downloads_to_reviews['age_range'] = downloads_to_reviews['age_range'].fillna('Unknown')
```

Start coding or generate with AI.

```
# Age Group
customer_funnel_by_age = downloads_to_reviews.groupby('age_range')[['user_downloaded', 'user_signup','ride_requested','ride_
customer_funnel_by_age
```

age_range	18-24	25-34	35-44	45-54	Unknown
user_downloaded	1865	3447	5181	1826	11289
user_signup	1865	3447	5181	1826	5304
ride_requested	1300	2425	3662	1285	3734
ride_accepted	1289	2393	3628	1267	3701
ride_completed	670	1227	1861	630	1845

Start coding or generate with AI.

```
# Platform
customer_funnel_by_platform = downloads_to_reviews.groupby('platform')[['user_downloaded', 'user_signup','ride_requested','r
customer_funnel_by_platform
```

platform	android	ios	web
user_downloaded	6935	14290	2383
user_signup	5148	10728	1747
ride_requested	3619	7550	1237
ride_accepted	3580	7471	1227
ride_completed	1830	3792	611

Start coding or generate with AI.

```
# Creating Full Customer Funnel
customer_funnel = downloads_to_reviews.loc[:,['user_downloaded', 'user_signup','ride_requested','ride_accepted','ride_comple
customer_funnel
```

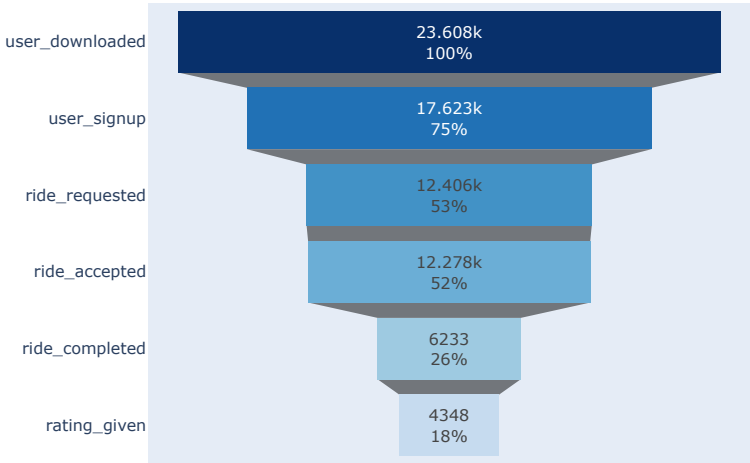
	0
user_downloaded	23608
user_signup	17623
ride_requested	12406
ride_accepted	12278
ride_completed	6233
rating_given	4348

Start coding or generate with AI.

Plotting Customer Funnel

```
total = customer_funnel[0]  # The first step is the total
percentages = (customer_funnel / total * 100).round(2)
colors = ['#08306B', '#2171B5', '#4292C6', '#6BAED6', '#9ECAE1', '#C6DBEF']

# Creating the funnel figure
fig = go.Figure(go.Funnel(
    y=customer_funnel.index,  # The names of the steps
    x=customer_funnel.values,  # The values for each step
    textinfo="value+percent initial",  # Display values and percentages
    marker=dict(color=colors)  # Apply gradient colors
))

fig.show()
```

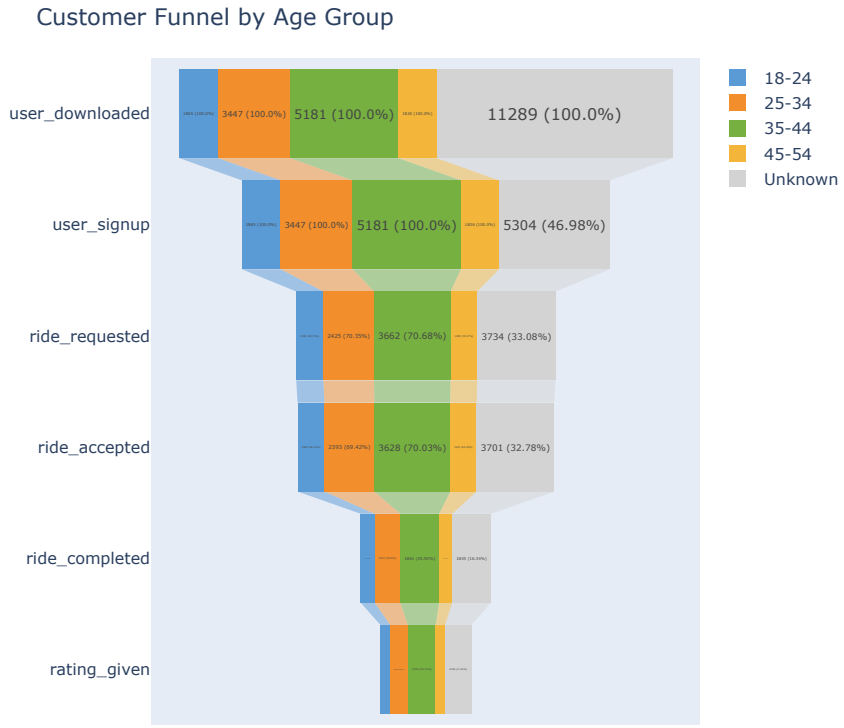
```
<ipython-input-28-040286b6c99e>:1: FutureWarning: Series.__getitem__ treating keys as positions is deprecated. In a futu
  total = customer_funnel[0]  # The first step is the total
```



Start coding or generate with AI.

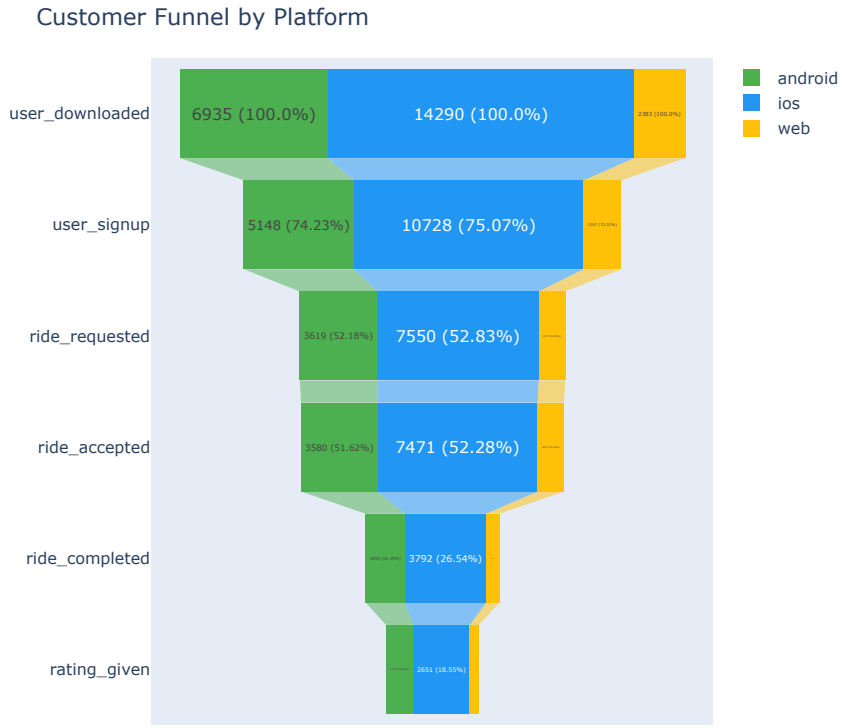
Plotting Customer By Age Group Funnel

```
import pandas as pd
import plotly.graph_objects as go

# Sample data setup
# Assuming customer_funnel_by_age is already defined
# customer_funnel_by_age = downloads_to_reviews.groupby('age_range')[
#     ['user_downloaded', 'user_signup', 'ride_requested', 'ride_accepted', 'ride_completed', 'rating_given']
# ].sum().T

# Calculate the percentage for each step within each age group
percentages = (customer_funnel_by_age / customer_funnel_by_age.loc['user_downloaded']) * 100

# Create custom text for each cell in the dataframe
custom_text = customer_funnel_by_age.astype(str) + ' (' + percentages.round(2).astype(str) + '%)'

# Define a new series of hash colors
colors = ["#5B9BD5", "#F28E2B", "#76B041", "#F3B63A", "#D3D3D3"]

# Create a figure for each age group
fig = go.Figure()

for i, age_group in enumerate(customer_funnel_by_age.columns):
    fig.add_trace(go.Funnel(
        name=age_group,
        y=customer_funnel_by_age.index,  # Funnel steps
        x=customer_funnel_by_age[age_group].values,
        text=custom_text[age_group].values,
        textinfo="text",  # Display only the custom text
        textposition="inside",
        marker=dict(color=colors[i % len(colors)]),  # Use color from the defined palette
        hoverinfo="text"  # Shows only the custom text on hover
    ))

# Update layout to separate age groups
fig.update_layout(
    title='Customer Funnel by Age Group',
    funnelmode='stack',
    margin=dict(l=50, r=50, t=50, b=50),
    height=600,
)

fig.show()
```



Start coding or generate with AI.

```
# customer_funnel_by_platform = downloads_to_reviews.groupby('platform')[
#     ['user_downloaded', 'user_signup', 'ride_requested', 'ride_accepted', 'ride_completed', 'rating_given']
# ].sum().T

# Calculate the percentage for each step within each platform
percentages = (customer_funnel_by_platform / customer_funnel_by_platform.loc['user_downloaded']) * 100

# Create custom text for each cell in the dataframe
custom_text = customer_funnel_by_platform.astype(str) + ' (' + percentages.round(2).astype(str) + '%)'

# Define a series of hash colors for green, blue, and tan
colors = ['#4CAF50',  # Green (a medium green)
          '#2196F3',  # Blue (a medium blue)
          '#FFC107']  # Tan (a medium tan or amber)


# Create a figure for each platform
fig = go.Figure()

for i, platform in enumerate(customer_funnel_by_platform.columns):
    fig.add_trace(go.Funnel(
        name=platform,
        y=customer_funnel_by_platform.index,  # Funnel steps
        x=customer_funnel_by_platform[platform].values,
        text=custom_text[platform].values,
        textinfo="text",  # Display only the custom text
        textposition="inside",
        marker=dict(color=colors[i % len(colors)]),  # Use color from the defined palette
        hoverinfo="text"  # Shows only the custom text on hover
    ))

# Update layout to separate platforms
fig.update_layout(
    title='Customer Funnel by Platform',
    funnelmode='stack',
    margin=dict(l=50, r=50, t=50, b=50),
    height=600,
)

fig.show()
```

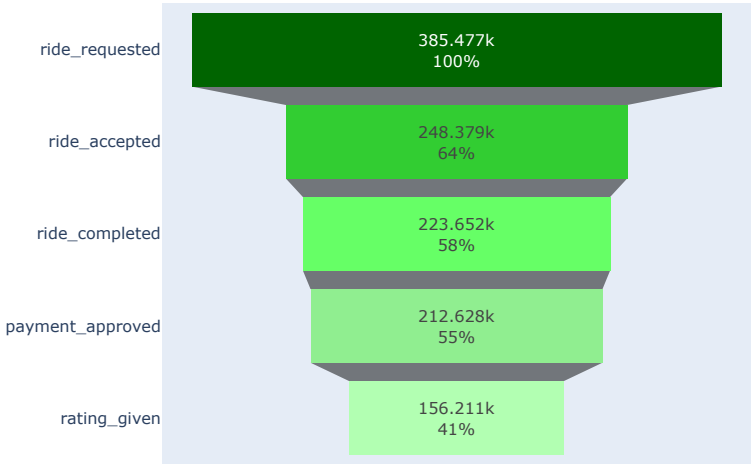


Start coding or generate with AI.

Plotting Rides Funnel

```
total = ride_funnel[0]  # The first step is the total
percentages = (ride_funnel / total * 100).round(2)
colors = ['#006400', '#32CD32', '#66FF66', '#90EE90', '#B2FFB2']

# Creating the funnel figure
fig = go.Figure(go.Funnel(
    y=ride_funnel.index,  # The names of the steps
    x=ride_funnel.values,  # The values for each step
    textinfo="value+percent initial",  # Display values and percentages
    marker=dict(color=colors)  # Apply gradient colors
))

fig.show()
```

```
<ipython-input-31-e880ca9345cf>:1: FutureWarning:
Series.__getitem__ treating keys as positions is deprecated. In a future version, integer keys will always be treated as
```



Start coding or generate with AI.

Mekonnen

Start coding or generate with AI.

```
#sign_down['age_range'] = sign_down['age_range'].fillna('Unknown')
#sign_down.head()
```

Start coding or generate with AI.

```
ride_sign = ride_requests.merge(signups, how='left', on='user_id')
ride_sign_trans = ride_sign.merge(transactions, how='left', on='ride_id')
four_tables = ride_sign_trans.merge(reviews, how='left', on='ride_id')
```

Start coding or generate with AI.

```
sign_down = app_downloads.join(signups, how='left')
ride_transact = ride_requests.merge(transactions, how='left', on='ride_id')
ride_trans_rev = ride_transact.merge(reviews, how='left', on='ride_id')
all_tables = ride_trans_rev.join(sign_down, how='outer')
```

Start coding or generate with AI.

```
# three_tables = sign_down.merge(ride_requests, how='right', on='user_id', suffixes=('_load', '_ride'))
# four_tables = three_tables.merge(reviews, how='left', on='ride_id')
```

```
ride_sign['has_request'] = ride_sign['request_ts'].notnull()
ride_sign['has_accept'] = ride_sign['accept_ts'].notnull()
ride_sign['has_dropoff'] = ride_sign['dropoff_ts'].notnull()
ride_sign_trans['has_paid'] = ride_sign_trans['charge_status'] == 'Approved'
four_tables['has_review'] = four_tables['review'].notnull()
```

Start coding or generate with AI.

```
#all_tables.info()
```

```
age_ride = {
    'ride_request': ride_sign[ride_sign['has_request']].groupby('age_range')['ride_id'].count(),
```

```
    'ride_accepted': ride_sign[ride_sign['has_accept']].groupby('age_range')['ride_id'].count(),
    'ride_completed': ride_sign[ride_sign['has_dropoff']].groupby('age_range')['ride_id'].count(),
    'ride_paid': ride_sign_trans[ride_sign_trans['charge_status'] == 'Approved'].groupby('age_range')['ride_id'].count(),
    'reviewed': four_tables[four_tables['review'].notnull()].groupby('age_range')['ride_id'].count()
}
age_ride_df = pd.DataFrame(age_ride)
age_ride_df.columns = ['ride_requested','ride_accepted','ride_completed','ride_paid', 'reviewed']


#four_tables[four_tables['review'].notnull()].groupby('age_range')['ride_id'].count()


age_ride_df = pd.DataFrame(age_ride)
age_ride_df
```

	ride_request	ride_accepted	ride_completed	ride_paid	reviewed
age_range					
18-24	40620	26607	24046	22922	16982
25-34	75236	48879	44121	41900	30295
35-44	114209	74130	66853	63521	47881
45-54	39683	25236	22675	21529	16287

Start coding or generate with AI.

```
age_rides = age_ride_df.copy()


for value in age_rides.columns[:]:
    age_rides[f'{value}_rate'] = round(age_rides[value] / age_rides['ride_request'] * 100, 1)
    print(age_rides)
```

```
            ride_request  ride_accepted  ride_completed  ride_paid  reviewed  \
age_range
18-24              40620          26607           24046      22922     16982
25-34              75236          48879           44121      41900     30295
35-44             114209          74130           66853      63521     47881
45-54              39683          25236           22675      21529     16287
Unknown           115729          73527           65957      62756     44766

            ride_request_rate
age_range
18-24                   100.0
25-34                   100.0
35-44                   100.0
45-54                   100.0
Unknown                 100.0
            ride_request  ride_accepted  ride_completed  ride_paid  reviewed  \
age_range
18-24              40620          26607           24046      22922     16982
25-34              75236          48879           44121      41900     30295
35-44             114209          74130           66853      63521     47881
45-54              39683          25236           22675      21529     16287
Unknown           115729          73527           65957      62756     44766

            ride_request_rate  ride_accepted_rate
age_range
18-24                   100.0                65.5
25-34                   100.0                65.0
35-44                   100.0                64.9
45-54                   100.0                63.6
Unknown                 100.0                63.5
            ride_request  ride_accepted  ride_completed  ride_paid  reviewed  \
age_range
18-24              40620          26607           24046      22922     16982
25-34              75236          48879           44121      41900     30295
35-44             114209          74130           66853      63521     47881
45-54              39683          25236           22675      21529     16287
Unknown           115729          73527           65957      62756     44766

            ride_request_rate  ride_accepted_rate  ride_completed_rate
age_range
18-24                   100.0                65.5                 59.2
25-34                   100.0                65.0                 58.6
35-44                   100.0                64.9                 58.5
45-54                   100.0                63.6                 57.1
Unknown                 100.0                63.5                 57.0
            ride_request  ride_accepted  ride_completed  ride_paid  reviewed  \
age_range
18-24              40620          26607           24046      22922     16982
25-34              75236          48879           44121      41900     30295
35-44             114209          74130           66853      63521     47881
```

```
45-54            39683          25236           22675      21529     16287
Unknown         115729          73527           65957      62756     44766

           ride_request_rate  ride_accepted_rate  ride_completed_rate  \
age_range
18-24                  100.0                65.5                 59.2
25-34                  100.0                65.0                 58.6
35-44                  100.0                64.9                 58.5
```

Start coding or generate with AI.

```
age_rides
```

	ride_request	ride_accepted	ride_completed	ride_paid	reviewed	ride_request_rate	ride_accepted_rate	ride_c
age_range								
18-24	40620	26607	24046	22922	16982	100.0	65.5	
25-34	75236	48879	44121	41900	30295	100.0	65.0	
35-44	114209	74130	66853	63521	47881	100.0	64.9	
45-54	39683	25236	22675	21529	16287	100.0	63.6	
Unknown	115729	73527	65957	62756	44766	100.0	63.5	

Start coding or generate with AI.

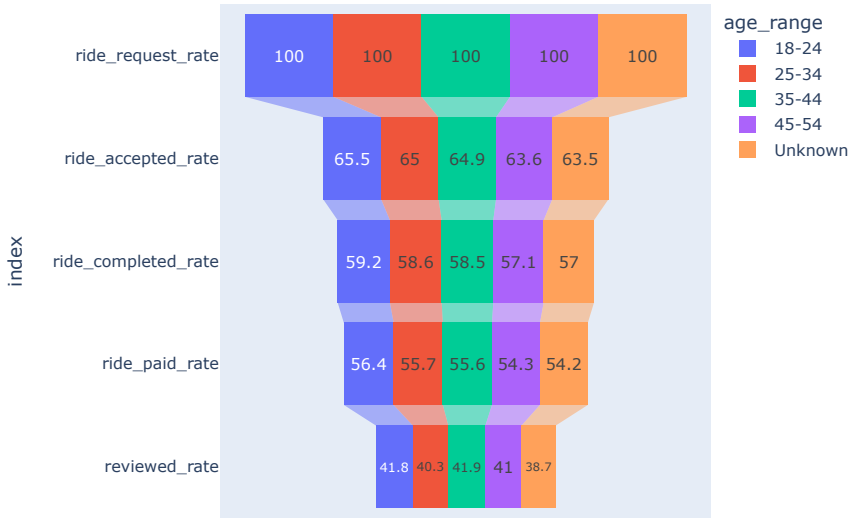
```
cols_to_drop = age_rides.columns[:5]
age_rides_selected = age_rides.drop(columns=cols_to_drop)
```

```
age_rides_selected = age_rides[age_rides.columns[5:10]]
age_rides_selected
```

	ride_request_rate	ride_accepted_rate	ride_completed_rate	ride_paid_rate	reviewed_rate
age_range					
18-24	100.0	65.5	59.2	56.4	41.8
25-34	100.0	65.0	58.6	55.7	40.3
35-44	100.0	64.9	58.5	55.6	41.9
45-54	100.0	63.6	57.1	54.3	41.0

Start coding or generate with AI.

```
fig = px.funnel(age_rides_selected.T)
fig.show()
```



Start coding or generate with AI.

```
ride_sign_down = ride_sign.join(app_downloads, how='outer')
ride_sign_trans_down = ride_sign_trans.join(app_downloads, how='outer')
four_tables_down = four_tables.join(app_downloads, how='outer')
```

```
ride_sign_down['has_request'] = ride_sign_down['request_ts'].notnull()
ride_sign_down['has_accept'] = ride_sign_down['accept_ts'].notnull()
ride_sign_down['has_dropoff'] = ride_sign_down['dropoff_ts'].notnull()
ride_sign_trans_down['has_paid'] = ride_sign_trans_down['charge_status'] == 'Approved'
four_tables_down['has_review'] = four_tables_down['review'].notnull()
```

Start coding or generate with AI.

```
platform_ride = {
    'ride_request': ride_sign_down[ride_sign_down['has_request']].groupby('platform')['ride_id'].count(),
    'ride_accepted': ride_sign_down[ride_sign_down['has_accept']].groupby('platform')['ride_id'].count(),
    'ride_completed': ride_sign_down[ride_sign_down['has_dropoff']].groupby('platform')['ride_id'].count(),
    'ride_paid': ride_sign_trans_down[ride_sign_trans_down['charge_status'] == 'Approved'].groupby('platform')['ride_id'].cc
    'reviewed': four_tables_down[four_tables_down['review'].notnull()].groupby('platform')['ride_id'].count()
}
platform_ride_df = pd.DataFrame(platform_ride)
platform_ride_df.columns = ['ride_requested','ride_accepted','ride_completed','ride_paid', 'reviewed']
```

```
platform_ride_df
```

platform	ride_requested	ride_accepted	ride_completed	ride_paid	reviewed
android	6935	4429	4015	3824	2813
ios	14290	9300	8389	7958	5851

```
for value in platform_ride_df.columns[:]:
    platform_ride_df[f'{value}_rate'] = round(platform_ride_df[value] / platform_ride_df['ride_requested'] * 100, 1)
    print(platform_ride_df)
```

```
ios                     100.0
```

```
android                 100.0                63.9
ios                     100.0                65.1
web                     100.0                64.5
          ride_requested  ride_accepted  ride_completed  ride_paid  reviewed  \
platform
android             6935           4429            4015       3824      2813
ios                14290           9300            8389       7958      5851
web                 2383           1536            1380       1321       952

          ride_requested_rate  ride_accepted_rate  ride_completed_rate
platform
android                 100.0                63.9                 57.9
ios                     100.0                65.1                 58.7
web                     100.0                64.5                 57.9
          ride_requested  ride_accepted  ride_completed  ride_paid  reviewed  \
platform
android             6935           4429            4015       3824      2813
ios                14290           9300            8389       7958      5851
web                 2383           1536            1380       1321       952

          ride_requested_rate  ride_accepted_rate  ride_completed_rate  \
platform
android                 100.0                63.9                 57.9
ios                     100.0                65.1                 58.7
web                     100.0                64.5                 57.9

          ride_paid_rate
platform
android             55.1
ios                 55.7
web                 55.4
          ride_requested  ride_accepted  ride_completed  ride_paid  reviewed  \
platform
android             6935           4429            4015       3824      2813
ios                14290           9300            8389       7958      5851
web                 2383           1536            1380       1321       952

          ride_requested_rate  ride_accepted_rate  ride_completed_rate  \
platform
android                 100.0                63.9                 57.9
ios                     100.0                65.1                 58.7
web                     100.0                64.5                 57.9

          ride_paid_rate  reviewed_rate
platform
android             55.1           40.6
ios                 55.7           40.9
web                 55.4           39.9
```

```
platform_ride_df = pd.DataFrame(platform_ride_df)
platform_ride_df
```

platform	ride_requested	ride_accepted	ride_completed	ride_paid	reviewed	ride_requested_rate	ride_accepted_rate	rid
android	6935	4429	4015	3824	2813	100.0	63.9	
ios	14290	9300	8389	7958	5851	100.0	65.1	
web	2383	1536	1380	1321	952	100.0	64.5	

```
cols_to_drop = platform_ride_df.columns[:5]
platform_rides_selected = platform_ride_df.drop(columns=cols_to_drop)
platform_rides_selected
```

platform	ride_requested_rate	ride_accepted_rate	ride_completed_rate	ride_paid_rate	reviewed_rate
android	100.0	63.9	57.9	55.1	40.6
ios	100.0	65.1	58.7	55.7	40.9

Start coding or generate with AI.

```
fig = px.funnel(platform_rides_selected.T)
fig.show()
```

Start coding or generate with AI.

```
ride_requests['hour1'] = ride_requests['request_ts'].dt.hour
```

```
ride_requests['hour2'] = ride_requests['accept_ts'].dt.hour
ride_requests['hour3'] = ride_requests['cancel_ts'].dt.hour
ride_requests['minutes'] = ride_requests['cancel_ts'].dt.minute
ride_requests['req_minutes'] = ride_requests['request_ts'].dt.minute.astype(float)
```

```
ride_demand_per_hour = ride_requests.groupby('hour1')['ride_id'].count()
ride_supply_per_hour = ride_requests.groupby('hour1')['driver_id'].nunique()
```

```
ride_demand_per_hour = pd.DataFrame(ride_demand_per_hour)
ride_supply_per_hour = pd.DataFrame(ride_supply_per_hour)
```

```
demand_supply = ride_supply_per_hour.merge(ride_demand_per_hour, how='left', left_on='hour1', right_on='hour1')
```

```
demand_supply.columns = ['Ride supply', 'Ride demand']
demand_supply
```

hour1	Ride supply	Ride demand
0	969	1554
1	1022	1593
2	1000	1627
3	964	1543
4	995	1576
5	1031	1633
6	971	1548
7	1023	1618
8	15641	60071
9	15664	60210
10	4898	9024
11	4419	7928
12	4443	7972
13	4487	7960
14	4428	7934
15	4428	7957
16	15537	58527
17	15464	58176
18	13524	40372
19	13493	39495
20	1384	2254
21	1054	1701
22	995	1624

```
average_bill = transactions['purchase_amount_usd'].mean()
average_bill
```

```
19.996162878042668
```

```
demand_supply['range'] = demand_supply['Ride demand'] - demand_supply['Ride supply']
demand_supply
total_gap = demand_supply['range'].sum()
total_gap * average_bill
```

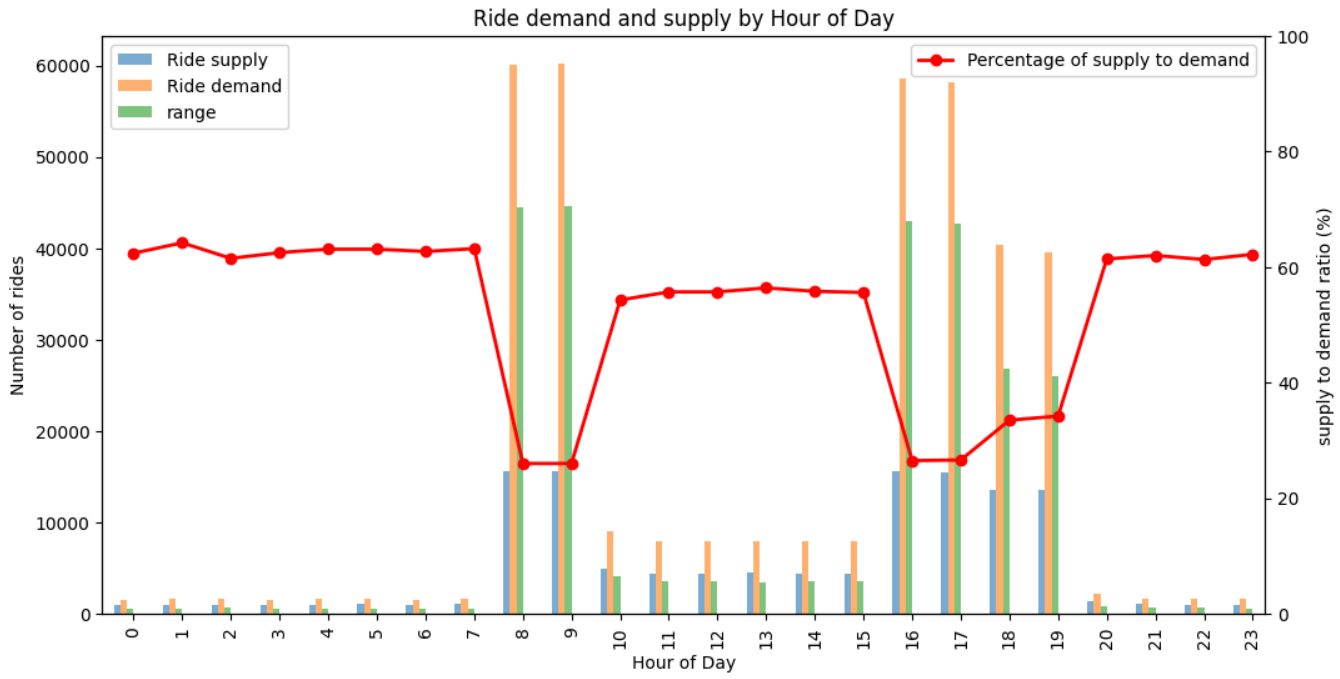
```
5132215.164278431
```

```
supply_demand_ratio = round(demand_supply['Ride supply'] / demand_supply['Ride demand'] * 100, 1)
supply_demand_ratio
```

hour1	0
0	62.4
1	64.2
2	61.5
3	62.5
4	63.1
5	63.1
6	62.7
7	63.2
8	26.0
9	26.0
10	54.3
11	55.7
12	55.7
13	56.4
14	55.8
15	55.6
16	26.5
17	26.6
18	33.5
19	34.2
20	61.4
21	62.0
22	61.3
23	62.2

Start coding or generate with AI.

```
# Display the quantitative results
# print("Ride demand vs supply by Hour:")
# print(table_data)

# Plot ride demand and supply by hour with percentage
fig, ax1 = plt.subplots(figsize=(12, 6))

# Bar plot for absolute number of ride demand and supply
demand_supply.plot(kind='bar', ax=ax1, alpha=0.6, position=1, width=0.4, label='Number of demand and supply')
ax1.set_xlabel('Hour of Day')
ax1.set_ylabel('Number of rides')
ax1.set_title('Ride demand and supply by Hour of Day')
ax1.legend(loc='upper left')

# Secondary axis for percentage of supply to demand
ax2 = ax1.twinx()
supply_demand_ratio.plot(kind='line', ax=ax2, color='r', marker='o', linewidth=2, label='Percentage of supply to demand')
ax2.set_ylabel('supply to demand ratio (%)')
ax2.set_ylim(0, 100)  # Ensure the y-axis for percentage is wide enough to display all values
ax2.legend(loc='upper right')

plt.show()
```




```
rides_cancelled = ride_requests[ride_requests['cancel_ts'].notnull()]
rides_cancelled_19_21h = rides_cancelled[(rides_cancelled['hour3'] >= 19) & (rides_cancelled['hour3'] <= 21)]
rides_requested_19_21h = ride_requests[(ride_requests['hour1'] >= 19) & (ride_requests['hour3'] <= 21)]
rides_requested_19_21h.head(2)
```

	ride_id	user_id	driver_id	request_ts	accept_ts	pickup_location	dropoff_location	pickup_ts	dropoff_ts	cancel_ts
46	3359594	102130	NaN	2021-03-25 19:36:00	NaT	40.75485595 -74.05608702	40.82173597 -74.10065525	NaT	NaT	2021-03-25 19:43:00
81	3359630	108481	NaN	2021-08-08 19:47:00	NaT	40.74517305 -73.81581231	40.79755461 -74.07798425	NaT	NaT	2021-08-08 19:54:00

```
cols_to_drop = rides_cancelled_19_21h.columns[5:9]
rides_cancelled_19_21h_selected = rides_cancelled_19_21h.drop(columns=cols_to_drop)
cols_to_drop1 = rides_requested_19_21h.columns[5:9]
rides_requested_19_21h_selected = rides_requested_19_21h.drop(columns=cols_to_drop1)
```

Start coding or generate with AI.

```
rides_cancelled_19_21h_selected['starth'] = rides_cancelled_19_21h_selected['hour3'] - 19
rides_cancelled_19_21h_selected['startmin'] = rides_cancelled_19_21h_selected['starth'] * 60
rides_cancelled_19_21h_selected['totalmin'] = rides_cancelled_19_21h_selected['startmin'] + rides_cancelled_19_21h_selected[
h19_h21_cancelled_rides = rides_cancelled_19_21h_selected[rides_cancelled_19_21h_selected['totalmin'] <= 120.0]
```

Start coding or generate with AI.

```
h19_h21_cancelled_rides.head(2)
```

	ride_id	user_id	driver_id	request_ts	accept_ts	cancel_ts	ride_requested	ride_accepted	ride_completed	hour1	ho
46	3359594	102130	NaN	2021-03-25 19:36:00	NaT	2021-03-25 19:43:00	True	False	False	19	I
81	3359630	108481	NaN	2021-08-08 19:47:00	NaT	2021-08-08 19:54:00	True	False	False	19	I

Start coding or generate with AI.

```
rides_requested_19_21h_selected.head(2)
```

	ride_id	user_id	driver_id	request_ts	accept_ts	cancel_ts	ride_requested	ride_accepted	ride_completed	hour1	ho
46	3359594	102130	NaN	2021-03-25 19:36:00	NaT	2021-03-25 19:43:00	True	False	False	19	I
81	3359630	108481	NaN	2021-08-08 19:47:00	NaT	2021-08-08 19:54:00	True	False	False	19	I

Start coding or generate with AI.

```
rides_requested_19_21h['starth'] = rides_requested_19_21h['hour1'] - 19
rides_requested_19_21h['startmin'] = rides_requested_19_21h['starth'] * 60
rides_requested_19_21h = pd.DataFrame(rides_requested_19_21h)
```

```
<ipython-input-70-53a1666fdbad>:1: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-

<ipython-input-70-53a1666fdbad>:2: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-
```

Start coding or generate with AI.

```
rides_requested_19_21h.head(2)
```

	ride_id	user_id	driver_id	request_ts	accept_ts	pickup_location	dropoff_location	pickup_ts	dropoff_ts	cancel_ts
46	3359594	102130	NaN	2021-03-25 19:36:00	NaT	40.75485595 -74.05608702	40.82173597 -74.10065525	NaT	NaT	2021-03-25 19:43:00
81	3359630	108481	NaN	2021-08-08 19:47:00	NaT	40.74517305 -73.81581231	40.79755461 -74.07798425	NaT	NaT	2021-08-08 19:54:00

Start coding or generate with AI.

Start coding or generate with AI.

```
rides_requested_19_21h['rid_totalmin'] = rides_requested_19_21h['startmin'] + rides_requested_19_21h['req_minutes']
h19_h21_requested_rides = rides_requested_19_21h[rides_requested_19_21h['rid_totalmin'] <= 120.0]
```

```
h19_h21_cancelled_rides.head(2)
```

	ride_id	user_id	driver_id	request_ts	accept_ts	cancel_ts	ride_requested	ride_accepted	ride_completed	hour1	ho
46	3359594	102130	NaN	2021-03-25 19:36:00	NaT	2021-03-25 19:43:00	True	False	False	19	I
81	3359630	108481	NaN	2021-08-08 19:47:00	NaT	2021-08-08 19:54:00	True	False	False	19	I

```
h19_h21_requested_rides.head(2)
```

	ride_id	user_id	driver_id	request_ts	accept_ts	pickup_location	dropoff_location	pickup_ts	dropoff_ts	cancel_ts
46	3359594	102130	NaN	2021-03-25 19:36:00	NaT	40.75485595 -74.05608702	40.82173597 -74.10065525	NaT	NaT	2021-03-25 19:43:00
81	3359630	108481	NaN	2021-08-08 19:47:00	NaT	40.74517305 -73.81581231	40.79755461 -74.07798425	NaT	NaT	2021-08-08 19:54:00

2 rows × 21 columns

Start coding or generate with AI.

```
rides_selected = h19_h21_cancelled_rides[['ride_id','totalmin']]
ride_req_selected = h19_h21_requested_rides[['ride_id', 'rid_totalmin']]
ride_req_selected_min = h19_h21_requested_rides.groupby('rid_totalmin')['ride_id'].count()
ride_req_selected_min.columns = ['count_ride_req']
ride_req_selected_min = pd.DataFrame(ride_req_selected_min)
```

```
ride_req_selected_min
```

rid_totalmin	ride_id
0.0	285
1.0	250
2.0	256
3.0	283
4.0	254
...	...
116.0	12
117.0	9
118.0	6
119.0	11
120.0	15

```
rides_selected_min = rides_selected.groupby('totalmin')['ride_id'].count()
rides_selected_min = pd.DataFrame(rides_selected_min)
```

```
rides_selected_min.columns =['ride_count']
```

```
request_vs_cancel = ride_req_selected_min.merge(rides_selected_min, left_on='rid_totalmin', right_on='totalmin')
request_vs_cancel.columns = ['ride_requested', 'ride_cancelled']
```

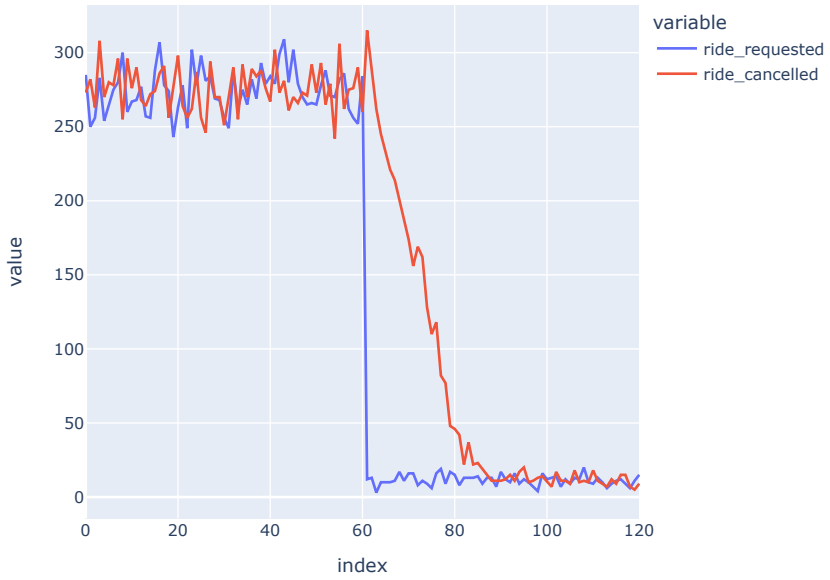
```
request_vs_cancel
```

	ride_requested	ride_cancelled
0	285	273
1	250	282
2	256	263
3	283	308
4	254	270
...	...	...
116	12	15
117	9	15
118	6	7
119	11	5
120	15	9

```
rides_selected_min.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 121 entries, 0.0 to 120.0
Data columns (total 1 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   ride_count  121 non-null    int64
dtypes: int64(1)
memory usage: 1.9 KB
```

```
fig = px.line(request_vs_cancel)
fig.show()
```



Start coding or generate with AI.

Start coding or generate with AI.

AnnaMatviichuk

Distribution of Ride Requests Throughout the Day

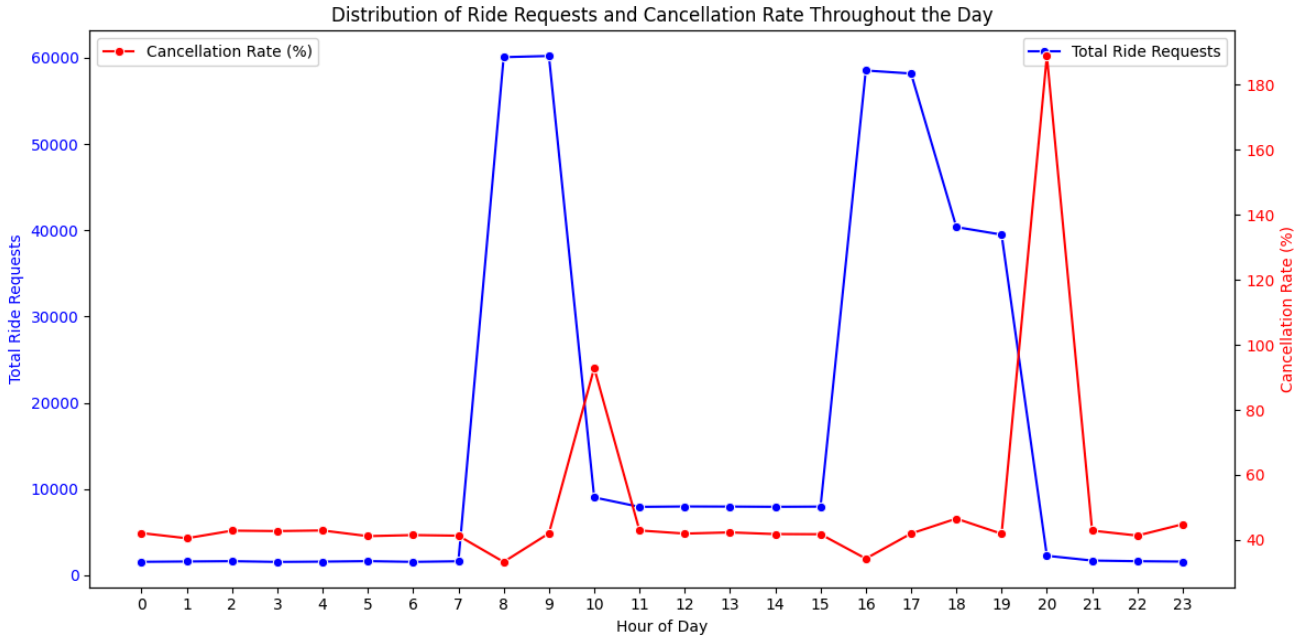
Start coding or generate with AI.

Distribution of Ride Requests and Cancellation Rate Throughout the Day

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sqlalchemy import create_engine

# Connection details
connection_string = "postgresql://Test:bQNxVzJL4g6u@ep-noisy-flower-846766-pooler.us-east-2.aws.neon.tech/Metrocar"
engine = create_engine(connection_string)

# Load ride_requests data
ride_requests = pd.read_sql_query("SELECT ride_id, request_ts, cancel_ts FROM ride_requests", engine)

# Ensure that request_ts and cancel_ts are in datetime format
ride_requests['request_ts'] = pd.to_datetime(ride_requests['request_ts'])
ride_requests['cancel_ts'] = pd.to_datetime(ride_requests['cancel_ts'])

# Extract hour of day from request_ts
ride_requests['request_hour'] = ride_requests['request_ts'].dt.hour

# Extract hour of day from cancel_ts
ride_requests['cancel_hour'] = ride_requests['cancel_ts'].dt.hour

# Group by request_hour and count the number of ride requests
hourly_ride_requests = ride_requests.groupby('request_hour').agg(total_ride_requests=('ride_id', 'count')).reset_index()

# Group by cancel_hour and count the number of cancellations
hourly_cancellations = ride_requests[ride_requests['cancel_ts'].notnull()].groupby('cancel_hour').agg(total_cancellations=('

# Merge the two dataframes to have both requests and cancellations
hourly_data = pd.merge(hourly_ride_requests, hourly_cancellations, left_on='request_hour', right_on='cancel_hour', how='left
#hourly_data['total_cancellations'].fillna(0, inplace=True)  # Fill NaNs with 0

# Calculate the cancellation rate
```

```
hourly_data['cancellation_rate'] = (hourly_data['total_cancellations'] / hourly_data['total_ride_requests']) * 100

# Sort by request_hour
hourly_data_sorted = hourly_data.sort_values(by='request_hour')

# Display quantitative results
print("Hourly Data with Cancellation Rates (%):")
print(hourly_data_sorted[['request_hour', 'total_ride_requests', 'total_cancellations', 'cancellation_rate']])

# Plotting
fig, ax1 = plt.subplots(figsize=(12, 6))

# Plot total ride requests
sns.lineplot(x='request_hour', y='total_ride_requests', data=hourly_data_sorted, marker='o', color='blue', ax=ax1, label='To
ax1.set_xlabel('Hour of Day')
ax1.set_ylabel('Total Ride Requests', color='blue')
ax1.tick_params(axis='y', labelcolor='blue')

# Create a second y-axis to plot the cancellation rate
ax2 = ax1.twinx()
sns.lineplot(x='request_hour', y='cancellation_rate', data=hourly_data_sorted, marker='o', color='red', ax=ax2, label='Cance
ax2.set_ylabel('Cancellation Rate (%)', color='red')
ax2.tick_params(axis='y', labelcolor='red')

plt.title('Distribution of Ride Requests and Cancellation Rate Throughout the Day')
plt.xticks(range(24))  # Ensure all hours are shown on x-axis
plt.grid(False)

# Show plot
plt.tight_layout()
plt.show()
```

```
Hourly Data with Cancellation Rates (%):
    request_hour  total_ride_requests  total_cancellations  cancellation_rate
0              0                 1554                  654          42.084942
1              1                 1593                  645          40.489642
2              2                 1627                  697          42.839582
3              3                 1543                  659          42.709008
4              4                 1576                  676          42.893401
5              5                 1633                  672          41.151255
6              6                 1548                  642          41.472868
7              7                 1618                  668          41.285538
8              8                60071                19953          33.215695
9              9                60210                25295          42.011294
10            10                 9024                 8382          92.885638
11            11                 7928                 3401          42.898587
12            12                 7972                 3343          41.934270
13            13                 7960                 3367          42.298995
14            14                 7934                 3315          41.782203
15            15                 7957                 3321          41.736835
16            16                58527                20036          34.233772
17            17                58176                24405          41.950289
18            18                40372                18789          46.539681
19            19                39495                16539          41.876187
20            20                 2254                 4258         188.908607
21            21                 1701                  729          42.857143
22            22                 1624                  671          41.317734
23            23                 1580                  708          44.810127
```



Start coding or generate with AI.

Start coding or generate with AI.

Exploratory data analysis with Python

```
# Import all the needed libraries
import pandas as pd
import sqlalchemy as sa
!pip install psycopg2-binary
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import numpy as np
import plotly.graph_objects as go
```

```
Requirement already satisfied: psycopg2-binary in /usr/local/lib/python3.10/dist-packages (2.9.9)
```

```
# Database connection URL
engine = sa.create_engine('postgresql://Test:bQNxVzJL4g6u@ep-noisy-flower-846766-pooler.us-east-2.aws.neon.tech/Metrocar')
connection = engine.connect().execution_options(isolation_level="AUTOCOMMIT")
```

```
inspector = sa.inspect(engine)
inspector.get_table_names()
```

```
['transactions', 'signups', 'ride_requests', 'reviews', 'app_downloads']
```

```
# All tables were imported and read with panda as data frame
app_downloads = pd.read_sql("SELECT * FROM app_downloads",connection)
signups = pd.read_sql("SELECT * FROM signups",connection)
ride_requests = pd.read_sql("SELECT * FROM ride_requests",connection)
transactions = pd.read_sql("SELECT * FROM transactions",connection)
reviews = pd.read_sql("SELECT * FROM reviews",connection)
```

```
#Check for Duplications
print("Duplicate rows in app_downloads:", app_downloads.duplicated().sum())
print("Duplicate rows in signups:", signups.duplicated().sum())
print("Duplicate rows in ride_requests:", ride_requests.duplicated().sum())
print("Duplicate rows in transactions:", transactions.duplicated().sum())
print("Duplicate rows in reviews:", reviews.duplicated().sum())
```

```
Duplicate rows in app_downloads: 0
Duplicate rows in signups: 0
Duplicate rows in ride_requests: 0
Duplicate rows in transactions: 0
Duplicate rows in reviews: 0
```

```
# Check for NULL values
print("NULL values in app_downloads:\n", app_downloads.isna().sum())
print("NULL values in signups:\n", signups.isna().sum())
print("NULL values in ride_requests:\n", ride_requests.isna().sum())
print("NULL values in transactions:\n", transactions.isna().sum())
print("NULL values in reviews:\n", reviews.isna().sum())
```

```
NULL values in app_downloads:
 app_download_key    0
platform            0
download_ts         0
dtype: int64
NULL values in signups:
 user_id       0
session_id    0
signup_ts     0
age_range     0
dtype: int64
NULL values in ride_requests:
 ride_id                  0
user_id                   0
driver_id            137098
request_ts                0
accept_ts            137098
pickup_location           0
dropoff_location          0
pickup_ts            161825
dropoff_ts           161825
cancel_ts            223652
dtype: int64
NULL values in transactions:
 transaction_id         0
ride_id                0
purchase_amount_usd    0
charge_status          0
transaction_ts         0
dtype: int64
NULL values in reviews:
 review_id    0
ride_id      0
user_id      0
driver_id    0
rating       0
review       0
dtype: int64
```

```
# Check for 'Unknown' values
print("Unknown values in app_downloads:\n", (app_downloads == 'Unknown').sum())
print("Unknown values in signups:\n", (signups == 'Unknown').sum())
print("Unknown values in ride_requests:\n", (ride_requests == 'Unknown').sum())
print("Unknown values in transactions:\n", (transactions == 'Unknown').sum())
print("Unknown values in reviews:\n", (reviews == 'Unknown').sum())
```

```
Unknown values in app_downloads:
 app_download_key    0
platform            0
download_ts         0
dtype: int64
Unknown values in signups:
```

```
 user_id          0
session_id       0
signup_ts        0
age_range     5304
dtype: int64
Unknown values in ride_requests:
 ride_id             0
user_id             0
driver_id           0
request_ts          0
accept_ts           0
pickup_location     0
dropoff_location    0
pickup_ts           0
dropoff_ts          0
cancel_ts           0
dtype: int64
Unknown values in transactions:
 transaction_id         0
ride_id                0
purchase_amount_usd    0
charge_status          0
transaction_ts         0
dtype: int64
Unknown values in reviews:
 review_id    0
ride_id      0
user_id      0
driver_id    0
rating       0
review       0
dtype: int64
```

Start coding or generate with AI.

PYTHON QUIZ

Question 1: How many times was the app downloaded?

```
# Count the total number of app downloads
total_downloads = app_downloads['app_download_key'].nunique()
print(f'Total number of app downloads: {total_downloads}')
```

```
Total number of app downloads: 23608
```

Question 2: How many users signed up on the app?

```
# Count the total number of users who signed up
total_signups = signups['user_id'].nunique()
print(f'Total number of users who signed up: {total_signups}')
```

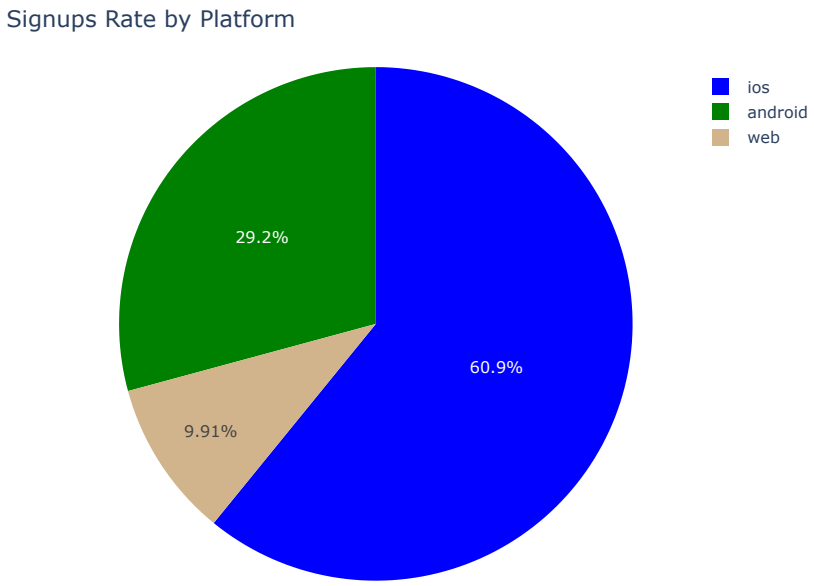
```
Total number of users who signed up: 17623
```

2 Addition: Metrocar currently supports 3 different platforms: ios, android, and web. To recommend where to focus our marketing budget for the upcoming year, what insights can we make based on the platform?

```
# left join between app_downloads and signups
merged_df = pd.merge(app_downloads, signups, left_on='app_download_key', right_on='session_id', how='left')

# Group by platform and aggregate
result_df = merged_df.groupby('platform').agg(
    total_downloads=('app_download_key', 'count'),
    total_signups=('user_id', 'count')
).reset_index()


# Display
result_df['signup_rate'] = result_df['total_signups'] / result_df['total_downloads']
result_df
```

	platform	total_downloads	total_signups	signup_rate
0	android	6935	5148	0.742322
1	ios	14290	10728	0.750735

```
#Using plotly express to create a pie chart to show the rate of signups by platform
df = px.data.tips()
fig = px.pie(result_df, names="platform", values="total_signups", color_discrete_sequence=["blue", "green", "tan"])
fig.update_layout(title_text="Signups Rate by Platform")
fig.show()
```



Question 3: How many rides were requested through the app?

```
# Count the total number of rides requested
total_rides_requested = ride_requests['ride_id'].nunique()
print(f'Total number of rides requested: {total_rides_requested}')
```

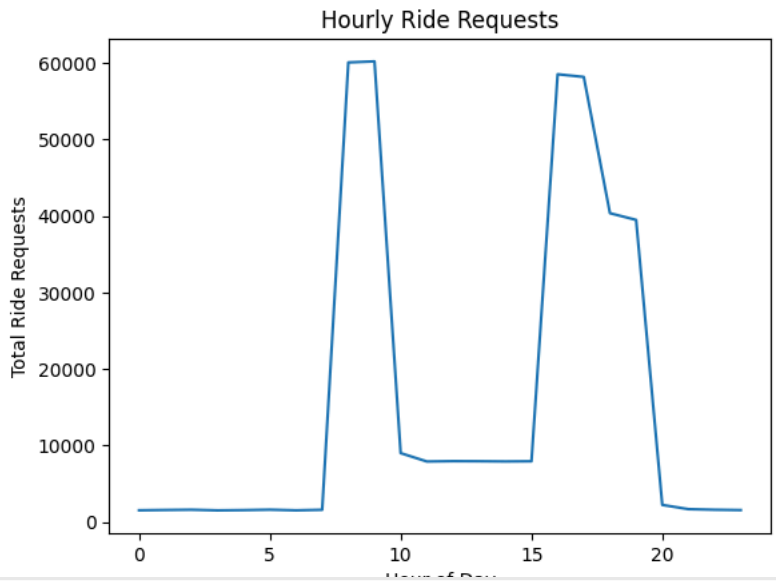
```
Total number of rides requested: 385477
```

Question 3 Addition: If we want to adopt a price-surging strategy, in such way the distribution of ride requests look like throughout the day

```
# Ensure that request_ts is in datetime format
ride_requests['request_ts'] = pd.to_datetime(ride_requests['request_ts'])

# Extract hour of day from request_ts
ride_requests['hour_of_day'] = ride_requests['request_ts'].dt.hour

# Group by hour_of_day and count the number of ride requests
hourly_ride_requests = (ride_requests
    .groupby('hour_of_day')
    .agg(total_ride_requests=('ride_id', 'count'))
    .reset_index()
)

# Sort by hour_of_day
hourly_ride_requests_sorted = hourly_ride_requests.sort_values(by='hour_of_day')

# Display
hourly_ride_requests_sorted
```

	hour_of_day	total_ride_requests
0	0	1554
1	1	1593
2	2	1627
3	3	1543
4	4	1576
5	5	1633
6	6	1548
7	7	1618
8	8	60071
9	9	60210
10	10	9024
11	11	7928
12	12	7972
13	13	7960
14	14	7934
15	15	7957
16	16	58527
17	17	58176
18	18	40372
19	19	39495
20	20	2254
21	21	1701
22	22	1624

```
sns.lineplot(data=hourly_ride_requests_sorted, x='hour_of_day', y='total_ride_requests')
plt.title('Hourly Ride Requests')
plt.xlabel('Hour of Day')
plt.ylabel('Total Ride Requests')
plt.show()
```



Question 4:How many rides were requested and completed through the app?

```
# rides requested through the app
completed_rides = ride_requests['dropoff_ts'].value_counts().sum()
print(f'The total number of completed rides are: {completed_rides}')
users_completed_ride = ride_requests[ride_requests['dropoff_ts'].notnull()]
```

```
# rides completed through the app
num_users_completed = users_completed_ride['user_id'].nunique()
print(f'The number of users who completed a ride are: {num_users_completed}')
```

```
The total number of completed rides are: 223652
The number of users who completed a ride are: 6233
```

Question 5: How many rides were requested and how many unique users requested a ride?

```
rides_requested = ride_requests['request_ts'].value_counts().sum()
print(f'The total number of users that requested a ride are: {rides_requested}')
number_of_unique_users = ride_requests['user_id'].nunique()
print(f'The number of unique users who requested a ride are: {number_of_unique_users}')
```

```
The total number of users that requested a ride are: 385477
The number of unique users who requested a ride are: 12406
```

Question 6: What is the average time of a ride from pick up to drop off?

```
#6 What is the average time of a ride from pick up to drop off?
ride_requests['time_elapsed'] = ride_requests['dropoff_ts'] - ride_requests['pickup_ts']
ride_requests['time_elapsed'].mean()
```

```
Timedelta('0 days 00:52:36.738772736')
```

Question 7: how many rides were accepted by a driver

```
#how many rides were accepted by a driver
ride_accepted_by_driver = {'The number of rides accepted by the driver are':ride_requests['accept_ts'].count()}
print(ride_accepted_by_driver)

unique_user_rides_accepted = {'The number of unique_user_rides_accepted are': ride_requests[ride_requests['accept_ts'].notnu
print(unique_user_rides_accepted)

ride_accept_rate_by_driver = {'The rate of request acceptance by the driver is':ride_requests['accept_ts'].count() / ride_re
print(ride_accept_rate_by_driver)
```

```
{'The number of rides accepted by the driver are': 248379}
{'The number of unique_user_rides_accepted are': 12278}
{'The rate of request acceptance by the driver is': 0.6443419451744202}
```

Question 8: For how many rides did we successfully collect payments and how much was collected.?

```
# 8A. For how many rides did we successfully collect payments and how much was collected.

rides_transaction = ride_requests.merge(transactions, how='left', on ='ride_id')

# The total number of transaction can be calculated by counting trnsaction Id or related columns.
transactions_approved = rides_transaction[rides_transaction['charge_status'] == 'Approved']
num_successful_transaction = {'The number of successful transaction is ':transactions_approved['transaction_id'].count()}
print(num_successful_transaction)

# The total amount of money collected from this transaction can be calculated by the summing the cost of each transaction.
total_money_collected = {'The total amount of money collected in $ is':transactions_approved['purchase_amount_usd'].sum()}
print(total_money_collected)

# 8B. what is the transaction completion rate of users whose request has been accepted and completed the ride.
transaction_completion = {'The transaction completion rate is':transactions_approved['transaction_id'].count() /  rides_trar
print(transaction_completion)

# 8C. what is the pickup rate after being accepted by the driver?
pickup_rate = {'The pickup completion rate is':rides_transaction['pickup_ts'].count() / rides_transaction['accept_ts'].count
print(pickup_rate)

# Unique_users_transactions_approved
#transactions_approved['charge_status'].unique()
#total_users_paid = {'Unique_users_transactions_approved is':transactions_approved['user_id'].nunique()}
#total_users_paid

Unique_users_transactions_approved = {'Unique_users_transactions_approved is':transactions_approved['user_id'].nunique()}
Unique_users_transactions_approved
```

```
{'The number of successful transaction is ': 212628}
{'The total amount of money collected in $ is': 4251667.609999999}
{'The transaction completion rate is': 0.9507091374099047}
{'The pickup completion rate is': 0.9004464950740602}
```

```
{'Unique_users_transactions_approved is': 6233}
```

Question 9: How many ride requests happened on each platform?

```
# Step 1: Merge app_downloads with signups
merged_df1 = pd.merge(app_downloads, signups, how='left', left_on='app_download_key', right_on='session_id')

# Step 2: Merge the result with ride_requests
merged_df2 = pd.merge(merged_df1, ride_requests, how='left', on='user_id')

# Step 3: Group by platform and count distinct ride_ids
result = merged_df2.groupby('platform')['ride_id'].nunique().reset_index()

# Renaming the columns to match the desired output
result.columns = ['platform', 'ride_requests']

print(result)
```

```
   platform  ride_requests
0  android         112317
1      ios         234693
2      web          38467
```

Question 10: What is the drop-off from users signing up to users requesting a ride?

```
# Step 1: Calculate the total number of distinct user_id in signups
total_signups = signups['user_id'].nunique()

# Step 2: Calculate the number of distinct user_id in ride_requests
ride_request_users = ride_requests['user_id'].nunique()

# Step 3: Compute the percentage of signups that did not make a ride request
signup_to_request_per = (1 - (ride_request_users / total_signups)) * 100

# Convert to a DataFrame to match the SQL result format
result = pd.DataFrame({'signup_to_request_per': [signup_to_request_per]})
```

Number of unique user reviews

```
# Number of reviews from unique user
unique_reviews = pd.merge(reviews, ride_requests, on='ride_id')
reviews_of_unique_users = unique_reviews ['user_id_x'].nunique()
print(f'The number of reviews from unique users are: {reviews_of_unique_users}')
```

```
The number of reviews from unique users are: 4348
```

Reviews from users

```
# Reviews
#reviews = reviews['review_id'].nunique()
#print(f'The number of unique reviews are: {reviews_of_unique_users}')

reviews_of_unique_users = reviews['review_id'].nunique()
print(f'The number of unique reviews are: {reviews_of_unique_users}')
```

```
The number of unique reviews are: 156211
```

Start coding or generate with AI.

Start coding or generate with AI.

Double-click (or enter) to edit

SQL QUIZZES

Quiz 1: Explore the Metrocar Data with SQL

```
import pandas as pd
import sqlalchemy as sa
```

```
engine = sa.create_engine("postgresql://Test:bQNxVzJL4g6u@ep-noisy-flower-846766-pooler.us-east-2.aws.neon.tech/Metrocar")
connection = engine.connect().execution_options(isolation_level="AUTOCOMMIT")
```

Question 1: How many times was the app downloaded?

```
query ="""
SELECT COUNT(DISTINCT app_download_key) AS times_downloaded
FROM app_downloads
;"""

pd.read_sql(sa.text(query),connection)
```

times_downloaded

Question 2: How many users signed up on the app?

```
query ="""
SELECT COUNT(DISTINCT user_id)
FROM signups
;"""

pd.read_sql(sa.text(query),connection)
```

count

Question 3: How many rides were requested through the app?

```
query ="""
SELECT COUNT(DISTINCT ride_id) as rides_requested
FROM ride_requests
;"""

pd.read_sql(sa.text(query),connection)
```

rides_requested

Question 4: How many rides were requested and completed through the app?

```
query="""
SELECT
  COUNT(DISTINCT ride_id) AS rides_requested,
  COUNT(DISTINCT CASE WHEN cancel_ts IS NULL THEN ride_id END) AS rides_completed
FROM ride_requests
;"""

pd.read_sql(sa.text(query),connection)
```

rides_requested	rides_completed

Question 5: How many rides were requested and how many unique users requested a ride?

```
query="""
SELECT
  COUNT(DISTINCT ride_id) AS rides_requested,
  COUNT(DISTINCT user_id) AS users_requested
FROM ride_requests
;"""

pd.read_sql(sa.text(query),connection)
```

Question 6: What is the average time of a ride from pick up to drop off?

```
query="""
SELECT AVG(dropoff_ts - pickup_ts) AS pickup_dropoff_time_avg
FROM ride_requests
;"""

pd.read_sql(sa.text(query),connection)
```

	pickup_dropoff_time_avg

Question 7: How many rides were accepted by a driver?

```
query="""
SELECT COUNT(DISTINCT ride_id) AS rides_accepted
FROM ride_requests
WHERE accept_ts is not null
;"""

pd.read_sql(sa.text(query),connection)
```