Metrocar Funnel Analysis Report

October 25, 2023

Contents:

By,

Deepthi Binu

dipkrishna@gmail.com

Introduction:

The purpose of this project is to conduct a comprehensive funnel analysis for Metrocar, a ride-sharing app, to identify areas for improvement and optimization in the user journey. Our goal is to gain insights into user behavior at each stage of the funnel, enabling us to enhance the user experience, boost conversion rates, and explore the potential for surge pricing.
The Metrocar's business data from January 2021 to April 2022 are used for the analysis.

Software Utilized:
This project leverages two key software platforms to conduct a thorough analysis and visualization of the data:

- **SQL (Structured Query Language)**
- **Tableau**

By harnessing the power of SQL and Tableau,the aim is to provide actionable recommendations based on a data-driven analysis of Metrocar's customer funnel.

User Funnel Analysis

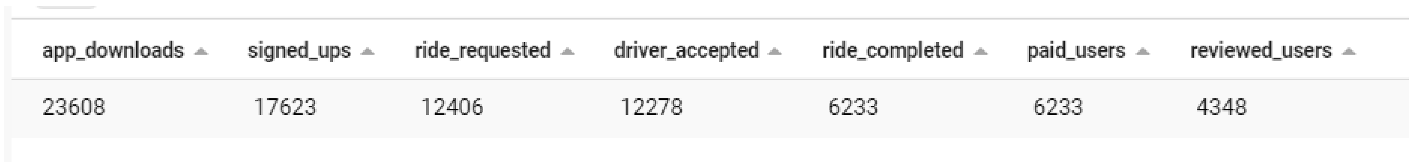
The User Funnel Analysis is a systematic approach to understand the journey of users through the Metrocar app. This process helps in identifying the key stages where users interact with the app and evaluating their conversion rates at each stage. The goal is to gain insights into user behavior, pinpoint areas of improvement, and provide recommendations for enhancing the user experience.

Step 1: Data Gathering
The analysis begins with the collection of relevant data from various sources.
Data sources include app_downloads, signups, ride_requests, transactions, and reviews tables.

Step 2: Data Integration
Data from different sources are integrated using SQL Queries, establishing connections between app downloads, signups, ride requests, transactions, and reviews. This integration allows for a comprehensive view of the user journey segmented by platform and Age-range.

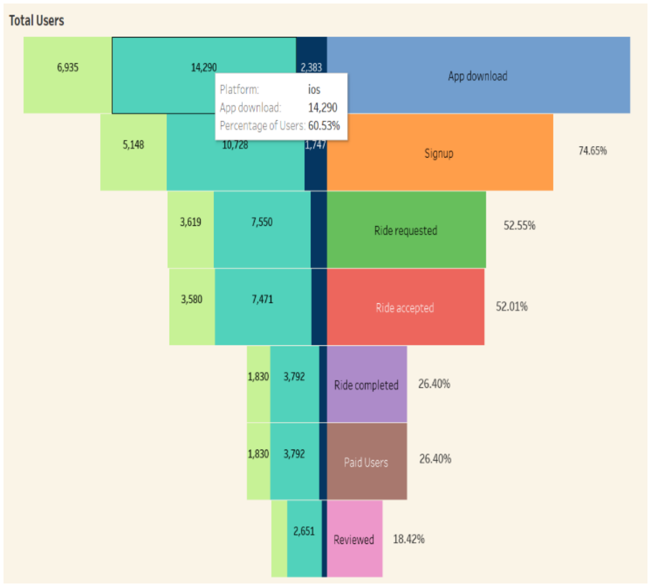
Step 3: Funnel Construction
A funnel is constructed to visualize the sequential stages of user interaction.
The funnel typically consists of the following stages:

i. App Download - User downloads Metrocar app from the app store.
ii. Signup- User creates an account with personal details.
iii. Request Ride- User enters pickup, destination, and ride details.
iv. Driver Acceptance- Nearby driver accepts the ride request.
v. Ride Completed- User gets in the car and travels to their destination.
vi. Payment - Payment is automatically processed through the app.
vii. Review - User rates the driver and leaves feedback.



app_downloads	signed_ups	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
23608	17623	12406	12278	6233	6233	4348

Step 4: SQL Queries, Tables and Tableau Integration
SQL queries are used to extract, clean, and transform the data for analysis.
SQL queries include:
Data aggregation and filtering
Calculation of conversion rates
Grouping of users by age range and platform
The results of SQL queries are then integrated into Tableau for visualization.



Step 5: Conversion Rate Calculation

The conversion rate is calculated for each stage by dividing the number of users who successfully progress to the next stage by the total number of users in the previous stage.
Conversion rates provide insights into the effectiveness of the app at different stages.

Funnel Usage percentage difference from top to bottom:

app_downloads ▲	signed_ups ▲	ride_requested ▲	driver_accepted ▲	ride_completed ▲	paid_users ▲	reviewed_users ▲
23608	74.65	70.40	98.97	50.24	50.24	24.67

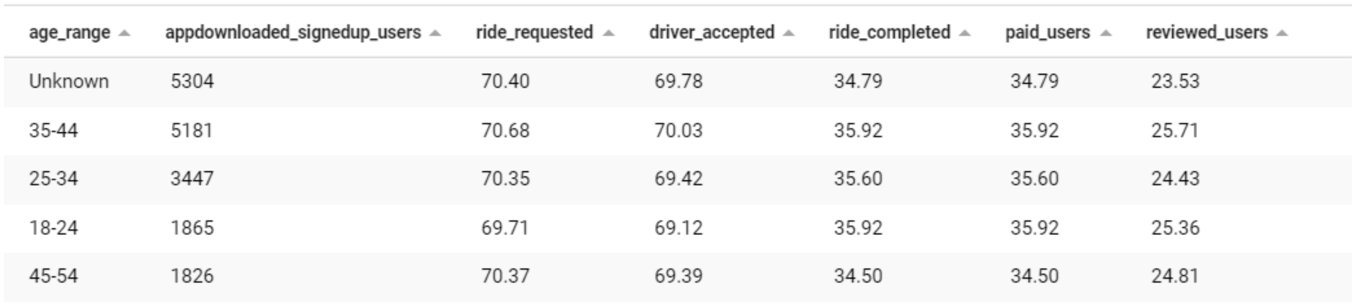
Funnel step-to-step percentage difference:

app_downloads ▲	signed_ups ▲	ride_requested ▲	driver_accepted ▲	ride_completed ▲	paid_users ▲	reviewed_users ▲
23608	74.65	52.55	52.01	26.40	26.40	18.42

Funnel Conversion rate by platform

platform ▲	app_downloads ▲	signed_ups ▲	ride_requested ▲	driver_accepted ▲	ride_completed ▲	paid_users ▲	reviewed_users ▲
ios	14290	75.07	52.83	52.28	26.54	26.54	18.55
android	6935	74.23	52.18	51.62	26.39	26.39	18.36
web	2383	73.31	51.91	51.49	25.64	25.64	17.79

Group Users by Age-range and analyzed the funnel:



age_range	appdownloaded_signedup_users	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
Unknown	5304	70.40	69.78	34.79	34.79	23.53
35-44	5181	70.68	70.03	35.92	35.92	25.71
25-34	3447	70.35	69.42	35.60	35.60	24.43
18-24	1865	69.71	69.12	35.92	35.92	25.36
45-54	1826	70.37	69.39	34.50	34.50	24.81

Step 6: Key Insights from User Funnel Analysis:

- App Downloads - 23,608:

A substantial number of users have downloaded the Metrocar app, indicating a healthy initial user base.

- Signed-Ups - 17,623:

The number of users who signed up after downloading the app is slightly lower than the total app downloads.
A potential gap exists in converting app downloads into sign-ups.

- Ride Requests - 12,406:

A significant drop-off occurs between sign-ups and ride requests, with fewer users proceeding to request rides.
This stage could be a potential area for improvement to enhance user engagement.

- Driver Accepted - 12,278:

The majority of ride requests result in driver acceptance, suggesting a strong conversion rate at this stage.

- Ride Completed - 6,233:

The ride completion stage exhibits a notable decline in user engagement.
Addressing drop-offs in this stage is crucial to maximize the app's value for users.

- Paid Users - 6,233:

The number of users who make payments aligns with the ride completion stage, indicating successful monetization of the service.

- Reviewed Users - 4,348:

A significant portion of users provide reviews after completing rides.
Encouraging user reviews and feedback is essential for improving user satisfaction and the app's reputation.

Platform-Based Analysis:

iOS users consistently exhibit higher conversion rates across all funnel stages compared to Android and Web users. For instance, the percentage of signed-up users from app downloads is 75.07% for iOS, while Android and Web have slightly lower rates at 74.23% and 73.31%, respectively.
The lower conversion rates on Android and Web platforms indicate areas with room for improvement, particularly in the sign-up and ride request stages.

Age Group-Based Analysis:

The age group-based funnel analysis provides valuable insights into user behavior. The age group "35-44" shows the highest conversion rates at nearly all stages of the funnel. This age group has the highest percentage of users progressing from app downloads to ride completion, making them a significant target audience.
The age group "18-24" exhibits a strong conversion rate, with nearly 70% of users progressing from app downloads to ride completion.
The age group "Unknown" presents higher attrition rates, suggesting a need for improved engagement strategies.

Overall Funnel Analysis:

There are notable drop-offs between the app download, sign-up, and ride request stages, signifying challenges in user engagement and retention.
The conversion rates for ride completion (6233 users) and paid users (6233 users) are particularly significant, indicating a strong correlation between ride completion and the generation of revenue.
The "reviewed users" metric, with 4348 users, demonstrates that a substantial portion of users provide feedback, which can be leveraged to enhance the overall user experience and retention.

These insights reveal the strengths and weaknesses of the user funnel. While the app has successfully attracted a large user base, optimizing the transition from app downloads to sign-ups and addressing drop-offs in the ride completion stage are areas with room for improvement. These key insights provide a foundation for data-driven decision-making and actionable recommendations to enhance user journeys, address drop-off hotspots, and optimize the funnel for Metrocar's ride application.

Step 7: Recommendations for Funnel Improvement:

To address the identified weaknesses and capitalize on opportunities for user funnel optimization, Metrocar should consider the following strategies:

- Optimize the Onboarding Process:
  Streamline the sign-up process to bridge the gap between app downloads and sign-ups.
  Ensure that the registration process is user-friendly and efficient, encouraging more users to complete their profiles.

- Enhance User Engagement:
  Implement strategies to boost user engagement between the sign-up and ride request stages.
  Utilize in-app notifications, personalized offers, or referral programs to incentivize users to request rides.

- Minimize Drop-offs During Ride Completion:
  Investigate the reasons behind the drop-offs in the ride completion stage and address any usability or technical issues.
  Enhance the user experience during rides to increase user satisfaction and completion rates.
- Encourage Reviews and Feedback:

Actively solicit user reviews and feedback after rides, both positive and constructive.
Utilize user feedback to make continuous improvements to the app's features and services.

- Segment and Target Users:
  Leverage user data to segment users based on their behaviors and preferences.
  Create targeted marketing campaigns and promotions to re-engage users at different stages of the funnel.

- Monitor Funnel Performance:
  Continuously track the conversion rates at each funnel stage.
  Implement analytics tools to gain real-time insights into user behavior and identify drop-off hotspots.

- A/B Testing and Experimentation:
  Conduct A/B testing to assess the impact of changes in app features and user experience.
  Experiment with different strategies to improve user retention and funnel progression.

- Retargeting and Communication:
  Implement retargeting campaigns for users who dropped off at various stages.
  Maintain open lines of communication with users through notifications and relevant updates.

- Data-Driven Decision-Making:
  Invest in robust data analytics to gain a deeper understanding of user behavior.
  Make data-driven decisions to guide improvements and prioritize features that enhance the user experience.

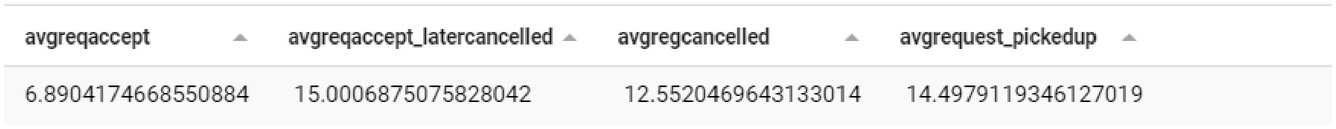
These recommendations are designed to enhance user funnel progression and overall user experience, resulting in increased user satisfaction and app success for Metrocar.

Time-based Analysis

The Time Analysis Report focuses on key time-related insights extracted from the available data, spanning various aspects of Metrocar's operations. It includes the aggregated data related to the average request accept time, monthly app download trends, monthly ride request trends and daily request time distribution by hour using SQL..

Average Request Time Metrics

Average Request Accept Time: The average time taken for drivers to accept ride requests is approximately 15.00 minutes.
Average Request to Cancel Time: The average duration before ride requests are canceled by users is around 12.55 minutes.
Average Pickup Time: The time it takes for drivers to reach the pickup location averages about 14.50 minutes.



avgreqaccept	avgreqaccept_latercancelled	avgregcancelled	avgrequest_pickedup
6.8904174668550884	15.0006875075828042	12.5520469643133014	14.4979119346127019

Monthly App Download Chart

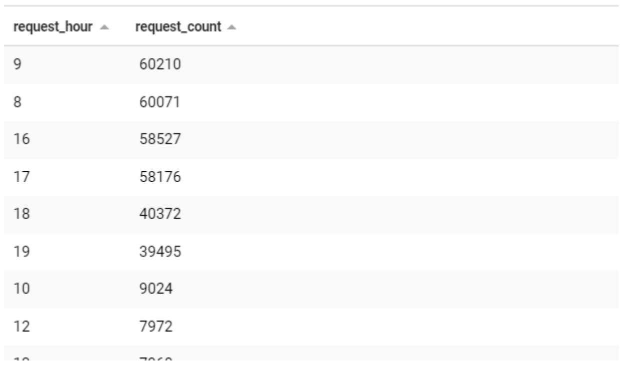
In January 2021, there were 2,045 app downloads.
June 2021 recorded 2,031 app downloads.
Monthly download counts range from 1,931 to 2,045, with slight variations throughout the year.

Monthly Ride Request Count:

December 2021 recorded the highest number of ride requests, with 47,782 requests.
November 2021 closely followed with 43,455 ride requests.
Requests remained consistently high throughout the last quarter of 2021.

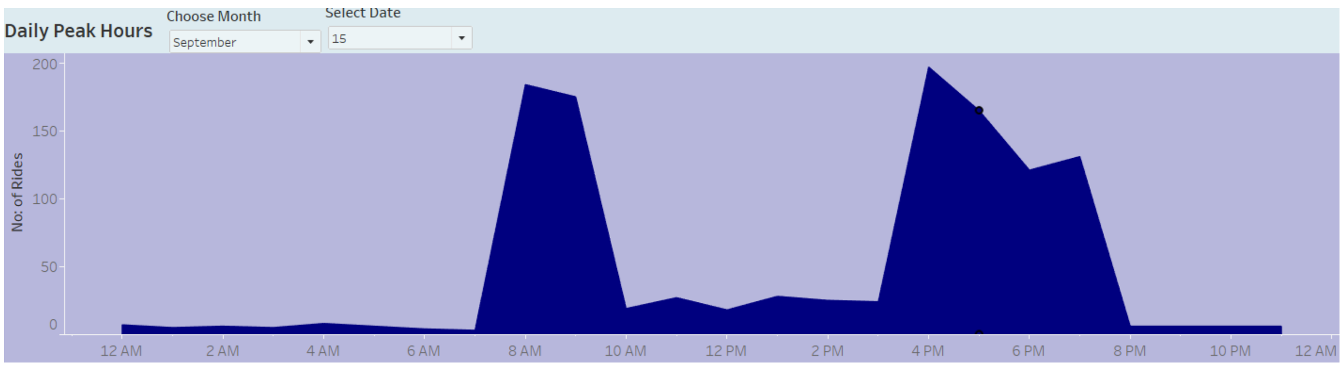
Avg: Daily Request Time Distribution by Hour

The busiest hours for ride requests are between 8 AM and 9 AM, with over 60,000 requests in each hour.
Afternoon hours from 4 PM to 6 PM also witness high request volumes.
Overnight hours between 12 AM and 3 AM experience lower request counts, with around 1,500 to 1,600 requests each hour.



request_hour	request_count
9	60210
8	60071
16	58527
17	58176
18	40372
19	39495
10	9024
12	7972
[illegible]	[illegible]

The results of SQL queries are then integrated into Tableau for visualization.



Key Insights of Time Analysis:

- Ride requests are concentrated during the morning and late afternoon hours, indicating peak demand times.
- App downloads have been relatively stable throughout the year.
- The average request accept time and average pickup time are within reasonable limits.
- The average request to cancel time is relatively shorter, suggesting prompt user decisions regarding request cancellations.
- The data indicates a strong positive trend in ride request counts from the start of the data set until the end of 2021, with some fluctuations.
- 2021 saw a steady increase in ride request counts from January to December, reaching its peak in December.
- The beginning of 2022 experienced a slight decrease in requests.
- Prior to 2021, ride request counts were significantly lower.

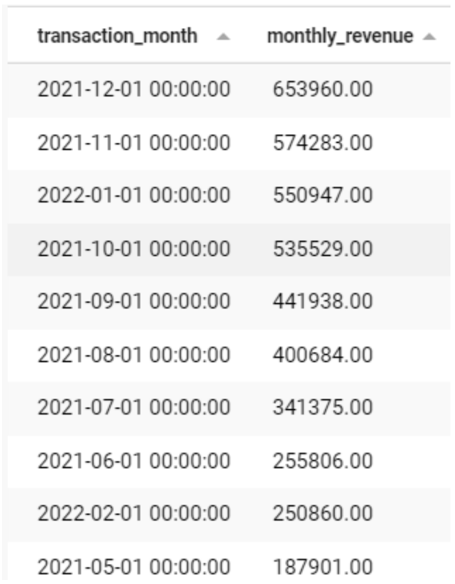
Recommendations related to Time Analysis:

- Capitalize on peak demand times by considering surge pricing strategies.
- Focus on maintaining the consistent number of monthly app downloads.
- Metrocar should continue to analyze and monitor ride request patterns.
- Strategies to manage peak demand times effectively, such as surge pricing, can be considered.
- Focusing on retaining customers and attracting new ones will be essential to maintaining growth in ride request counts.
- The company may explore the reasons for fluctuations in ride requests in early 2022 and work on strategies to regain growth
- Keep monitoring and optimizing the response time between request acceptances and pickups.
- Investigate and address any factors leading to shorter request to cancel times to enhance user experience.

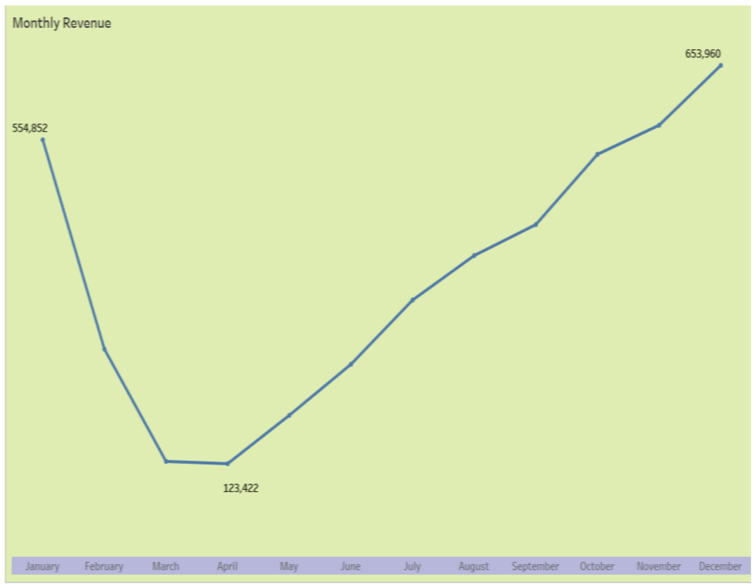
This Time Analysis Report serves as a valuable resource for understanding the temporal aspects of Metrocar's operations and provides insights to inform strategic decisions and improvements.

Revenue-Based Analysis Report

The Revenue Based Analysis Report focuses on key revenue-related insights extracted from the available data, spanning various aspects of Metrocar's operations. It includes the aggregated data related to the monthly revenue trends, platform-based revenue distribution, age group contributions, and the average revenue generated per user using SQL. The results of SQL queries are then integrated into Tableau for visualization.



transaction_month	monthly_revenue
2021-12-01 00:00:00	653960.00
2021-11-01 00:00:00	574283.00
2022-01-01 00:00:00	550947.00
2021-10-01 00:00:00	535529.00
2021-09-01 00:00:00	441938.00
2021-08-01 00:00:00	400684.00
2021-07-01 00:00:00	341375.00
2021-06-01 00:00:00	255806.00
2022-02-01 00:00:00	250860.00
2021-05-01 00:00:00	187901.00



Total Monthly Revenue: The combined revenue from all sources amounts to $4,472,182.

Key Insights on revenue-based analysis:

Monthly Revenue Trends:
The report highlights the monthly revenue trends, showcasing a positive growth trajectory. In December 2021, Metrocar achieved the highest monthly revenue at $653,960, indicating increased user engagement and transactions.

Platform-Based Revenue:
iOS users significantly contribute to the total revenue, with a total of $2,721,961. Android and Web platforms follow closely with revenues of $1,307,676 and $442,545, respectively. Understanding the platform dynamics is crucial for optimizing revenue sources.

Age Group Contributions: The report dissects the revenue by age groups, with the '35-44' age group generating the highest revenue at $1,336,910. The 'Unknown' age group follows closely with $1,316,723 in revenue, while '25-34' users contribute $882,904. Analyzing age group-specific revenue allows Metrocar to tailor marketing and engagement strategies.

Average Revenue Per User: Calculated at approximately $718 per user, understanding the average revenue per user enables Metrocar to gauge the lifetime value of a customer. Strategies to increase user spending or enhance customer retention can be developed based on this metric.

Recommendations on revenue-based analysis:

Platform Focus: Given the significant contribution of iOS users to the revenue, Metrocar should focus on retaining and engaging iOS users. Implement strategies such as exclusive features or loyalty programs to enhance their experience.

Age Group Strategies: Develop targeted marketing campaigns and promotions to engage the '35-44' age group, which represents a substantial revenue source. Personalized offers and incentives can drive revenue growth.

User Experience Enhancement: Continuously invest in improving the user experience. A satisfied user is more likely to engage with the platform and generate revenue. Focus on ease of use, speed, and efficiency.

In summary, this report provides a comprehensive view of Metrocar's revenue performance and offers actionable insights for optimizing revenue sources, enhancing user engagement, and ensuring long-term revenue growth.

Summary of Key Insights

Key User Funnel Analysis:

- The number of users who signed up after downloading the app is slightly lower than the total app downloads. A potential gap exists in converting app downloads into sign-ups.
- A significant drop-off occurs between sign-ups and ride requests, with fewer users proceeding to request rides. This stage could be a potential area for improvement to enhance user engagement.
- The ride completion stage exhibits a notable decline in user engagement. Addressing drop-offs in this stage is crucial to maximize the app's value for users.
- A significant portion of users provide reviews after completing rides. Encouraging user reviews and feedback is essential for improving user satisfaction and the app's reputation.

Platform-Based Analysis:

- iOS users consistently exhibit higher conversion rates across all funnel stages compared to Android and Web users, indicating a need for improvement, particularly in the sign-up and ride request stages.

Age Group-Based Analysis:

- The age group "35-44" shows the highest conversion rates at nearly all stages of the funnel, making them a significant target audience.
- The age group "18-24" exhibits a strong conversion rate, with nearly 70% of users progressing from app downloads to ride completion.
- The age group "Unknown" presents higher attrition rates, suggesting a need for improved engagement strategies.

Time Analysis:

- Ride requests are concentrated during the morning and late afternoon hours, indicating peak demand times.
- Monthly app downloads have been relatively stable throughout the year.
- The average request accept time and average pickup time are within reasonable limits.
- The average request to cancel time is relatively shorter, suggesting prompt user decisions.
- Ride request counts exhibit a positive trend until the end of 2021, with fluctuations.
- December 2021 experienced the highest ride request counts.
- The beginning of 2022 saw a slight decrease in requests.

Revenue-Based Analysis:

- Monthly revenue trends indicate positive growth, with December 2021 showing the highest monthly revenue.
- iOS users contribute significantly to the total revenue, followed by Android and Web platforms.
- Age group "35-44" generates the highest revenue, while "Unknown" and "18-24" also contribute substantially.
- The average revenue per user is approximately $718.

Summary of Key Recommendations:

- Optimize the onboarding process to bridge the gap between app downloads and sign-ups.
- Enhance user engagement strategies between sign-up and ride request stages.
- Investigate and address drop-offs in the ride completion stage.
- Encourage user reviews and feedback to improve user satisfaction.
- Segment and target users based on their behaviors and preferences.
- Monitor funnel performance and implement analytics tools for insights.
- Conduct A/B testing and experimentation for user retention.
- Implement retargeting campaigns and maintain communication with users.
- Invest in robust data analytics to make data-driven decisions.
- Capitalize on peak demand times using surge pricing.
- Focus on retaining customers and attracting new ones.
- Continuously monitor and optimize response times between request acceptances and pickups.
- Explore reasons for fluctuations in ride requests and work on strategies to regain growth.
- Focus on retaining and engaging iOS users who contribute significantly to revenue.
- Develop targeted marketing campaigns and promotions for the "35-44" age group.
- Continuously invest in improving the user experience to increase user spending and retention.

Conclusion:

In conclusion, this comprehensive analysis of Metrocar's operations has unveiled valuable insights that are pivotal for the continued growth and success of the service. The examination of user funnels, platform dynamics, age group behaviors, time-based patterns, and revenue structures has provided a 360-degree view of Metrocar's strengths and areas that require improvement.

It is evident that Metrocar has successfully attracted a substantial user base, with a significant number of app downloads and user sign-ups. However, the analysis has pinpointed several critical drop-off points in the user funnel, such as the transition from sign-up to ride requests and the decline in user engagement during the ride completion stage. Addressing these challenges offers an opportunity to enhance user retention and satisfaction.

Platform-based analysis has underscored the need to focus on retaining and engaging iOS users, who consistently exhibit higher conversion rates. The age group analysis further refines our targeting strategies, emphasizing the importance of the "35-44" age group and highlighting opportunities for improved engagement with other segments.

Time analysis provides a valuable understanding of peak demand times and the stability of app downloads, while revenue-based insights have shown positive growth trends and the significance of iOS users in contributing to revenue.

Our recommendations encompass a spectrum of strategies, from optimizing the onboarding process to implementing targeted marketing campaigns and focusing on user experience enhancements. These data-driven recommendations serve as actionable steps for Metrocar to enhance user funnel progression, retain and engage users, and drive revenue growth.

As Metrocar continues to evolve and expand its services, it is essential to make informed decisions and prioritize features and strategies that enhance the overall user experience. By doing so, Metrocar is well-positioned to not only retain its existing user base but also attract new users and establish itself as a prominent player in the ride-sharing industry.

This report serves as a valuable guide for Metrocar's future endeavors, offering a roadmap to unlock its full potential, optimize operations, and provide users with an exceptional and reliable service.

Limitations:

- Data Quality: The accuracy and completeness of the data are crucial, and any data issues can affect the analysis.
- Data Source: The analysis is based on data up to January 2022 and may not reflect recent changes.
- Causation vs. Correlation: Correlations identified do not imply causation.
- User Behavior Complexity: User behavior is influenced by various unmeasured factors.
- Competitive Landscape: Competitive actions can impact user behavior.
- Privacy and Ethics: Handling user data ethically and within privacy regulations is essential.
- Implementation Challenges: Some recommendations may require significant resources.
- External Factors: Economic conditions and external events can affect results.
- User Feedback Bias: User feedback may not represent all user experiences.
- Timeframe: User behavior evolves over time, requiring ongoing analysis and adaptation.

Recommended Future Steps:

- Continuous Monitoring: Regularly monitor key metrics and KPIs to stay updated on user behavior trends.
- A/B Testing: Implement A/B testing for app features and user experience enhancements to assess their impact.
- Refinement of Onboarding: Continually refine the onboarding process to improve the transition from app downloads to sign-ups.
- Enhanced User Engagement: Develop and launch strategies for boosting user engagement between sign-up and ride request stages.
- Drop-Off Mitigation: Investigate and address the causes of drop-offs during the ride completion stage.
- Feedback Utilization: Actively encourage user reviews and feedback to make ongoing improvements to the app.
- User Segmentation: Leverage user data to create targeted marketing campaigns for different user segments.
- Surge Pricing Strategies: Implement surge pricing strategies to capitalize on peak demand times and increase revenue.
- Retention Strategies: Develop and implement user retention strategies to maintain growth in ride request counts.
- Enhance User Experience: Invest in optimizing response times, addressing factors leading to shorter request to cancel times, and providing a seamless user experience.

These future steps aim to address the identified weaknesses, capitalize on opportunities, and ensure continuous improvement in Metrocar's operations and user experience.

Link to the Visualization in Tableau

https://public.tableau.com/app/profile/deepthi.binu/viz/CustomerFunnelAnalysis3/MetroCarFunnelAnalysis

Link to the PowerPoint presentation video

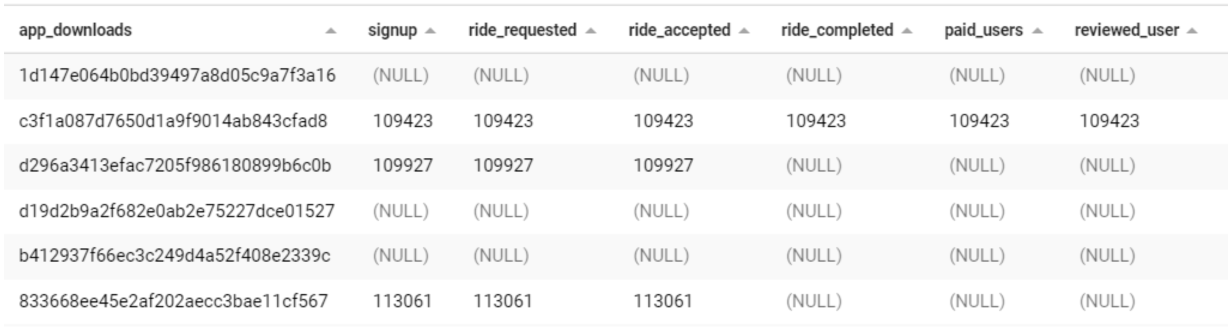
https://www.youtube.com/watch?v=cW_BYNUwapI

Appendix

The SQL queries used for this analysis are attached below.

User_id Aggregation for Funnel analysis

Sample Output:



app_downloads	signup	ride_requested	ride_accepted	ride_completed	paid_users	reviewed_user
1d147e064b0bd39497a8d05c9a7f3a16	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)
c3f1a087d7650d1a9f9014ab843cfad8	109423	109423	109423	109423	109423	109423
d296a3413efac7205f986180899b6c0b	109927	109927	109927	(NULL)	(NULL)	(NULL)
d19d2b9a2f682e0ab2e75227dce01527	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)
b412937f66ec3c249d4a52f408e2339c	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)
833668ee45e2af202aecc3bae11cf567	113061	113061	113061	(NULL)	(NULL)	(NULL)

Query used:

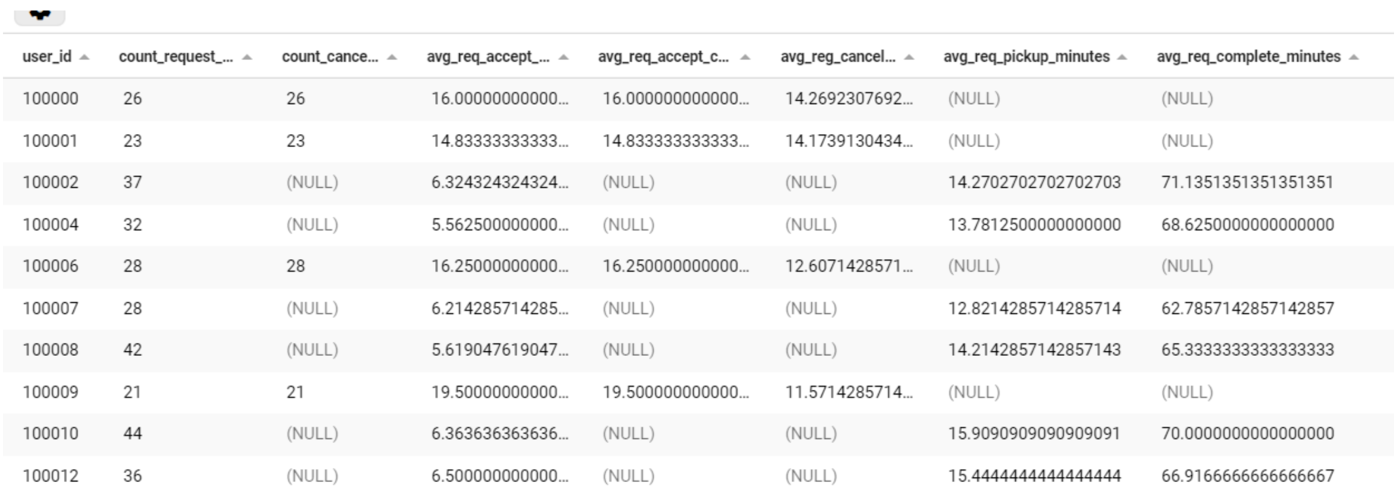
```
WITH user_accept AS
(SELECT DISTINCT rr.user_id AS user_id FROM ride_requests rr WHERE accept_ts IS NOT NULL),
  user_complete AS
  (SELECT DISTINCT rr.user_id AS user_id FROM ride_requests rr WHERE dropoff_ts IS NOT NULL AND cancel_ts IS NULL),
  user_paid AS
   (SELECT DISTINCT rr.user_id AS user_id FROM ride_requests rr LEFT JOIN transactions t ON rr.ride_id = t.ride_id WHERE
t.transaction_id IS NOT NULL),
  user_reviewd AS
  (SELECT DISTINCT user_id FROM reviews)

SELECT
DISTINCT ad.app_download_key AS app_downloads,
  s.user_id AS signup,
  rr.user_id AS ride_requested,
  user_accept.user_id AS ride_accepted,
  user_complete.user_id AS ride_completed,
  user_paid.user_id AS paid_users,
  user_reviewd.user_id AS reviewed_user
  FROM
  app_downloads ad
  LEFT JOIN
  signups s
  ON ad.app_download_key = s.session_id
  LEFT JOIN
    ride_requests rr
  ON s.user_id = rr.user_id
  LEFT JOIN transactions tr
  ON rr.ride_id = tr.ride_id
  LEFT JOIN reviews rev
  ON s.user_id = rev.user_id
```

```
    LEFT JOIN user_accept
    ON s.user_id = user_accept.user_id
    LEFT JOIN user_complete
    ON s.user_id = user_complete.user_id
    LEFT JOIN user_paid
    ON s.user_id = user_paid.user_id
    LEFT JOIN user_reviewd
    ON s.user_id = user_reviewd.user_id;
```

Time Analysis: sql code:

Sample Output:



user_id	count_request_...	count_cance...	avg_req_accept_...	avg_req_accept_c...	avg_reg_cancel...	avg_req_pickup_minutes	avg_req_complete_minutes
100000	26	26	16.00000000000...	16.000000000000...	14.2692307692...	(NULL)	(NULL)
100001	23	23	14.83333333333...	14.833333333333...	14.1739130434...	(NULL)	(NULL)
100002	37	(NULL)	6.324324324324...	(NULL)	(NULL)	14.2702702702702703	71.1351351351351351
100004	32	(NULL)	5.562500000000...	(NULL)	(NULL)	13.7812500000000000	68.6250000000000000
100006	28	28	16.25000000000...	16.250000000000...	12.6071428571...	(NULL)	(NULL)
100007	28	(NULL)	6.214285714285...	(NULL)	(NULL)	12.8214285714285714	62.7857142857142857
100008	42	(NULL)	5.619047619047...	(NULL)	(NULL)	14.2142857142857143	65.3333333333333333
100009	21	21	19.50000000000...	19.500000000000...	11.5714285714...	(NULL)	(NULL)
100010	44	(NULL)	6.363636363636...	(NULL)	(NULL)	15.9090909090909091	70.0000000000000000
100012	36	(NULL)	6.500000000000...	(NULL)	(NULL)	15.4444444444444444	66.9166666666666667

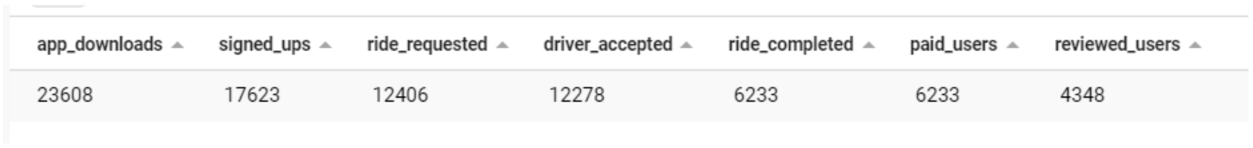
Query:

```
WITH ValidRequests AS (
    SELECT USER_ID,
           COUNT(*) AS Count_Request_Not_Null
    FROM ride_requests
    WHERE request_ts IS NOT NULL
    GROUP BY USER_ID
),
ValidCancellations AS (
    SELECT USER_ID,
           COUNT(*) AS Count_Cancel_Not_Null
    FROM ride_requests
    WHERE cancel_ts IS NOT NULL
    GROUP BY USER_ID
),
AvgReqAccept AS (
    SELECT USER_ID,
           AVG(EXTRACT(EPOCH FROM (accept_ts - request_ts)) / 60) AS Avg_Req_Accept_Minutes
    FROM ride_requests
    WHERE accept_ts IS NOT NULL
    GROUP BY USER_ID
),
AvgReqAccept2Cancel AS (
  SELECT USER_ID,
      AVG(EXTRACT(EPOCH FROM (accept_ts - request_ts)) / 60) AS Avg_Req_Accept_Cancel_Minutes
  FROM ride_requests
  WHERE accept_ts IS NOT NULL AND cancel_ts IS NOT NULL
  GROUP BY USER_ID
),
AvgRegCancel AS (
  SELECT USER_ID,
      AVG(EXTRACT(EPOCH FROM (cancel_ts - request_ts)) / 60) AS Avg_Reg_Cancel_Minutes
  FROM ride_requests
  WHERE cancel_ts IS NOT NULL
  GROUP BY USER_ID
),
AvgReqPickup AS (
  SELECT USER_ID,
      AVG(EXTRACT(EPOCH FROM (pickup_ts - request_ts)) / 60) AS Avg_Req_pickup_Minutes
  FROM ride_requests
  WHERE pickup_ts IS NOT NULL
  GROUP BY USER_ID
),
AvgReqComplete AS (
  SELECT USER_ID,
      AVG(EXTRACT(EPOCH FROM (dropoff_ts - request_ts)) / 60) AS Avg_Req_Complete_Minutes
  FROM ride_requests
  WHERE accept_ts IS NOT NULL
  GROUP BY USER_ID)

SELECT
  rr.USER_ID,
  VR.Count_Request_Not_Null,
  VC.Count_Cancel_Not_Null,
  ARA.Avg_Req_Accept_Minutes,
  ARAC.Avg_Req_Accept_Cancel_Minutes,
  ARC.Avg_Reg_Cancel_Minutes,
  ARP.Avg_Req_pickup_Minutes,
  ARCOM.Avg_Req_Complete_Minutes
FROM ride_requests rr
LEFT JOIN ValidRequests VR ON rr.USER_ID = VR.USER_ID
LEFT JOIN ValidCancellations VC ON rr.USER_ID = VC.USER_ID
LEFT JOIN AvgReqAccept ARA ON rr.USER_ID = ARA.USER_ID
LEFT JOIN AvgReqAccept2Cancel ARAC ON rr.USER_ID = ARAC.USER_ID
LEFT JOIN AvgRegCancel ARC ON rr.USER_ID = ARC.USER_ID
LEFT JOIN AvgReqPickup ARP ON rr.USER_ID = ARP.USER_ID
LEFT JOIN AvgReqComplete ARCOM ON rr.USER_ID = ARCOM.USER_ID
GROUP BY 1,2,3,4,5,6,7,8;
```

Unique Users in Each Stage:

Output:



app_downloads	signed_ups	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
23608	17623	12406	12278	6233	6233	4348

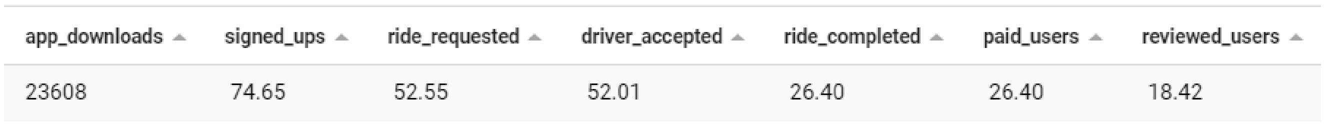
Query Used:

```
/* User Count funnel*/
WITH Funnel AS (
   SELECT
      ad.app_download_key AS app_download_key,
      s.user_id AS signup_user_id,
      rr.user_id AS request_user_id,
      rr.accept_ts AS ride_accepted_ts,
      rr.dropoff_ts AS ride_completed_ts,
                              tr.ride_id AS transcation_ride_id,
                              rev.user_id AS reviewed
   FROM app_downloads ad
   LEFT JOIN signups s ON ad.app_download_key = s.session_id
   LEFT JOIN ride_requests rr ON s.user_id = rr.user_id
          LEFT JOIN transactions tr ON rr.ride_id = tr.ride_id
          LEFT JOIN reviews rev ON s.user_id = rev.user_id
)
SELECT
   COUNT(DISTINCT app_download_key) AS app_downloads,
   COUNT(DISTINCT signup_user_id) AS signed_ups,
   COUNT(DISTINCT request_user_id) AS ride_requested,
   COUNT(DISTINCT CASE WHEN ride_accepted_ts IS NOT NULL THEN request_user_id END) AS driver_accepted,
   COUNT(DISTINCT CASE WHEN ride_completed_ts IS NOT NULL THEN request_user_id END)  AS ride_completed,
   COUNT(DISTINCT CASE WHEN transcation_ride_id IS NOT NULL THEN signup_user_id END)  AS Paid_users,
   COUNT(DISTINCT CASE WHEN reviewed IS NOT NULL THEN signup_user_id END) AS Reviewed_users

FROM Funnel;
```

FUNNEL Usage percentage difference from top to bottom:

Output:



app_downloads	signed_ups	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
23608	74.65	52.55	52.01	26.40	26.40	18.42

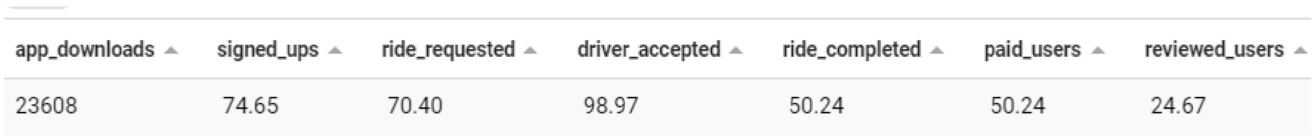
Query used:

```
WITH Funnel AS (
   SELECT
      ad.app_download_key AS app_download_key,
      s.user_id AS signup_user_id,
      rr.user_id AS request_user_id,
      rr.accept_ts AS ride_accepted_ts,
      rr.dropoff_ts AS ride_completed_ts,
                              tr.ride_id AS transcation_ride_id,
                              rev.user_id AS reviewed
   FROM app_downloads ad
   LEFT JOIN signups s ON ad.app_download_key = s.session_id
   LEFT JOIN ride_requests rr ON s.user_id = rr.user_id
        LEFT JOIN transactions tr ON rr.ride_id = tr.ride_id
        LEFT JOIN reviews rev ON s.user_id = rev.user_id
)
SELECT
   COUNT(DISTINCT app_download_key) AS app_downloads,
   ROUND(COUNT(DISTINCT signup_user_id) * 100.0 / COUNT(DISTINCT app_download_key),2) AS signed_ups,
   ROUND(COUNT(DISTINCT request_user_id) * 100.0 / COUNT(DISTINCT app_download_key),2) AS ride_requested,
   ROUND(COUNT(DISTINCT CASE WHEN ride_accepted_ts IS NOT NULL THEN request_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key),2) AS driver_accepted,
   ROUND(COUNT(DISTINCT CASE WHEN ride_completed_ts IS NOT NULL THEN request_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key),2) AS ride_completed,
   ROUND(COUNT(DISTINCT CASE WHEN transcation_ride_id IS NOT NULL THEN signup_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key),2) AS Paid_users,
   ROUND(COUNT(DISTINCT CASE WHEN reviewed IS NOT NULL THEN signup_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key),2) AS Reviewed_users

FROM Funnel;
```

FUNNEL STEP-TO-STEP percentage difference:

Output:



app_downloads	signed_ups	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
23608	74.65	70.40	98.97	50.24	50.24	24.67

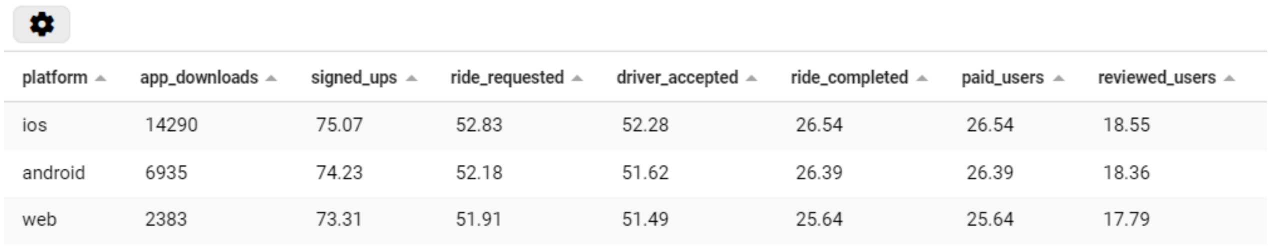
Query:

```
WITH Funnel AS (
    SELECT
        ad.app_download_key AS app_download_key,
        s.user_id AS signup_user_id,
        rr.user_id AS request_user_id,
        rr.accept_ts AS ride_accepted_ts,
        rr.dropoff_ts AS ride_completed_ts,
                                tr.ride_id AS transcation_ride_id,
                                rev.user_id AS reviewed
    FROM app_downloads ad
    LEFT JOIN signups s ON ad.app_download_key = s.session_id
    LEFT JOIN ride_requests rr ON s.user_id = rr.user_id
         LEFT JOIN transactions tr ON rr.ride_id = tr.ride_id
         LEFT JOIN reviews rev ON s.user_id = rev.user_id
)
```

```
SELECT
   COUNT(DISTINCT app_download_key) AS app_downloads,
   ROUND(COUNT(DISTINCT signup_user_id) * 100.0 / COUNT(DISTINCT app_download_key),2) AS signed_ups,
   ROUND(COUNT(DISTINCT request_user_id) * 100.0 / COUNT(DISTINCT signup_user_id),2) AS ride_requested,
   ROUND(COUNT(DISTINCT CASE WHEN ride_accepted_ts IS NOT NULL THEN request_user_id END) * 100.0 / COUNT(DISTINCT
request_user_id),2) AS driver_accepted,
   ROUND(COUNT(DISTINCT CASE WHEN ride_completed_ts IS NOT NULL THEN request_user_id END) * 100.0 /
COUNT(DISTINCT request_user_id),2) AS ride_completed,
   ROUND(COUNT(DISTINCT CASE WHEN transcation_ride_id IS NOT NULL THEN signup_user_id END) * 100.0 /
COUNT(DISTINCT request_user_id),2) AS Paid_users,
   ROUND(COUNT(DISTINCT CASE WHEN reviewed IS NOT NULL THEN signup_user_id END) * 100.0 / COUNT(DISTINCT
signup_user_id),2) AS Reviewed_users
FROM Funnel;
```

Funnel Conversion rate by platform

Output:



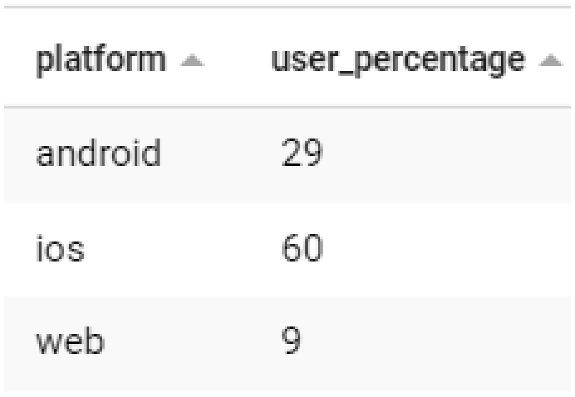
platform	app_downloads	signed_ups	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
ios	14290	75.07	52.83	52.28	26.54	26.54	18.55
android	6935	74.23	52.18	51.62	26.39	26.39	18.36
web	2383	73.31	51.91	51.49	25.64	25.64	17.79

Query Used:

```
/* Funnel Conversion rate by Platform */
WITH Funnel AS (
   SELECT
      ad.platform AS platform,
      ad.app_download_key AS app_download_key,
      s.user_id AS signup_user_id,
      rr.user_id AS request_user_id,
      rr.accept_ts AS ride_accepted_ts,
      rr.dropoff_ts AS ride_completed_ts,
      tr.ride_id AS transaction_ride_id,
      rev.user_id AS reviewed
   FROM app_downloads ad
   LEFT JOIN signups s ON ad.app_download_key = s.session_id
   LEFT JOIN ride_requests rr ON s.user_id = rr.user_id
   LEFT JOIN transactions tr ON rr.ride_id = tr.ride_id
   LEFT JOIN reviews rev ON s.user_id = rev.user_id
)
SELECT
   platform,
   COUNT(DISTINCT app_download_key) AS app_downloads,
   ROUND(COUNT(DISTINCT signup_user_id) * 100.0 / COUNT(DISTINCT app_download_key), 2) AS signed_ups,
   ROUND(COUNT(DISTINCT request_user_id) * 100.0 / COUNT(DISTINCT app_download_key), 2) AS ride_requested,
   ROUND(COUNT(DISTINCT CASE WHEN ride_accepted_ts IS NOT NULL THEN request_user_id END) * 100.0 / COUNT(DISTINCT
app_download_key), 2) AS driver_accepted,
   ROUND(COUNT(DISTINCT CASE WHEN ride_completed_ts IS NOT NULL THEN request_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key), 2) AS ride_completed,
    ROUND(COUNT(DISTINCT CASE WHEN transaction_ride_id IS NOT NULL THEN signup_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key), 2) AS paid_users,
    ROUND(COUNT(DISTINCT CASE WHEN reviewed IS NOT NULL THEN signup_user_id END) * 100.0 / COUNT(DISTINCT
app_download_key), 2) AS reviewed_users
FROM Funnel
GROUP BY platform
ORDER BY 2 DESC;
```

User Analysis by Different Platforms

Output:



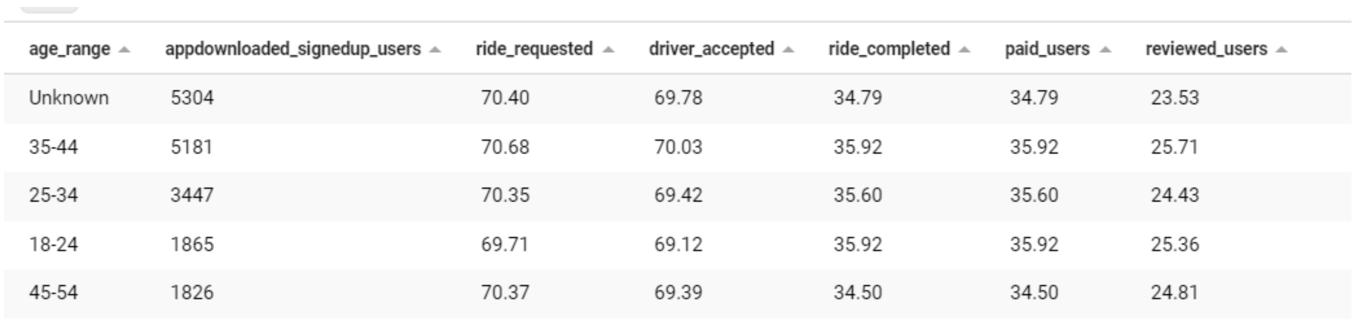
platform	user_percentage
android	29
ios	60
web	9

Query used:

```
SELECT ap.platform, (COUNT(DISTINCT s.user_id)*100/(SELECT COUNT(DISTINCT user_id) FROM signups)) AS user_percentage
FROM app_downloads ap
LEFT JOIN signups s
ON ap.app_download_key = s.session_id
GROUP BY 1;
```

Group users by age range and analyze their progression through the funnel

Output:



age_range	appdownloaded_signedup_users	ride_requested	driver_accepted	ride_completed	paid_users	reviewed_users
Unknown	5304	70.40	69.78	34.79	34.79	23.53
35-44	5181	70.68	70.03	35.92	35.92	25.71
25-34	3447	70.35	69.42	35.60	35.60	24.43
18-24	1865	69.71	69.12	35.92	35.92	25.36
45-54	1826	70.37	69.39	34.50	34.50	24.81

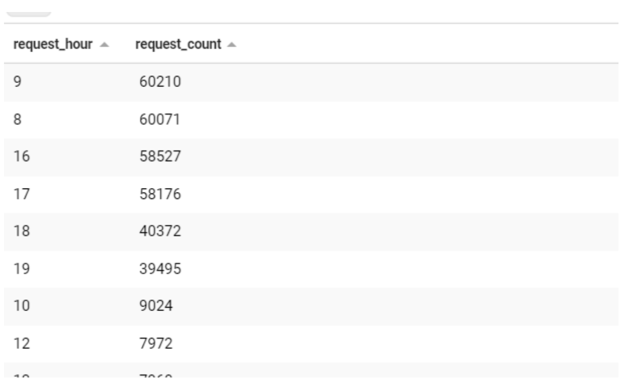
Query Used:

```
/* users by age range and analyze their progression through the funnel.*/
WITH Funnel AS (
    SELECT
        s.age_range,
        ad.app_download_key,
        s.user_id AS signup_user_id,
        rr.user_id AS request_user_id,
        rr.accept_ts AS ride_accepted_ts,
        rr.dropoff_ts AS ride_completed_ts,
        t.ride_id AS transaction_ride_id,
        rev.user_id AS reviewed
    FROM app_downloads ad
    LEFT JOIN signups s ON ad.app_download_key = s.session_id
    LEFT JOIN ride_requests rr ON s.user_id = rr.user_id
    LEFT JOIN transactions t ON rr.ride_id = t.ride_id
    LEFT JOIN reviews rev ON s.user_id = rev.user_id

)
SELECT
    age_range,
    COUNT(DISTINCT app_download_key) AS Appdownloaded_Signedup_users,
    ROUND(COUNT(DISTINCT request_user_id) * 100.0 / COUNT(DISTINCT app_download_key), 2) AS ride_requested,
    ROUND(COUNT(DISTINCT CASE WHEN ride_accepted_ts IS NOT NULL THEN request_user_id END) * 100.0 / COUNT(DISTINCT
app_download_key), 2) AS driver_accepted,
    ROUND(COUNT(DISTINCT CASE WHEN ride_completed_ts IS NOT NULL THEN request_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key), 2) AS ride_completed,
    ROUND(COUNT(DISTINCT CASE WHEN transaction_ride_id IS NOT NULL THEN signup_user_id END) * 100.0 /
COUNT(DISTINCT app_download_key), 2) AS paid_users,
    ROUND(COUNT(DISTINCT CASE WHEN reviewed IS NOT NULL THEN signup_user_id END) * 100.0 / COUNT(DISTINCT
app_download_key), 2) AS reviewed_users
FROM Funnel
WHERE age_range IS NOT NULL
GROUP BY age_range
ORDER BY 2 DESC;
```

Assess ride request patterns and determine peak demand times.

Sample Output:



request_hour	request_count
9	60210
8	60071
16	58527
17	58176
18	40372
19	39495
10	9024
12	7972

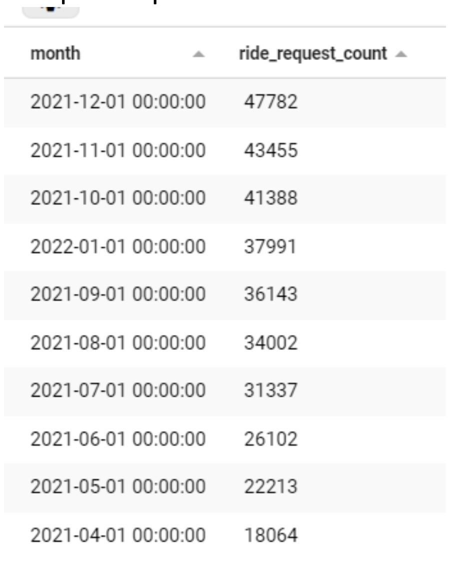
Query Used:

```
WITH RequestTimestamps AS (
   SELECT
      EXTRACT(HOUR FROM request_ts) AS request_hour,
      COUNT(*) AS request_count
   FROM ride_requests
   WHERE request_ts IS NOT NULL
   GROUP BY EXTRACT(HOUR FROM request_ts)
   ORDER BY EXTRACT(HOUR FROM request_ts)
)

SELECT
   request_hour,
   request_count
FROM RequestTimestamps
ORDER BY 2 DESC;
```

Analyse monthly ride request patterns:

Sample output:



month	ride_request_count
2021-12-01 00:00:00	47782
2021-11-01 00:00:00	43455
2021-10-01 00:00:00	41388
2022-01-01 00:00:00	37991
2021-09-01 00:00:00	36143
2021-08-01 00:00:00	34002
2021-07-01 00:00:00	31337
2021-06-01 00:00:00	26102
2021-05-01 00:00:00	22213
2021-04-01 00:00:00	18064

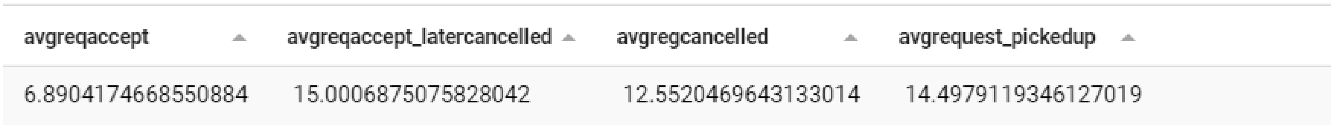
Query Used:

```
SELECT DATE_TRUNC('month', request_ts) AS month, COUNT(*) AS Ride_request_count
FROM ride_requests
```

```
GROUP BY month
ORDER BY 2 DESC;
```

Analyse the average Request to cancel timing:

Output:



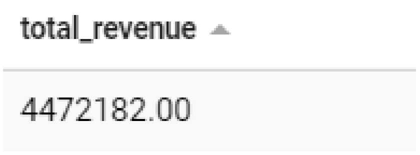
avgreqaccept	avgreqaccept_latercancelled	avgregcancelled	avgrequest_pickedup
6.8904174668550884	15.0006875075828042	12.5520469643133014	14.4979119346127019

Query Used:

```
SELECT  (SELECT AVG(EXTRACT(EPOCH FROM (accept_ts - request_ts)) / 60) AS Avg_Req_Accept_Minutes
   FROM ride_requests
   WHERE accept_ts IS NOT NULL)AS AvgReqAccept,
   (SELECT AVG(EXTRACT(EPOCH FROM (accept_ts - request_ts)) / 60) AS Avg_Req_Accept_Cancel_Minutes
   FROM ride_requests
   WHERE accept_ts IS NOT NULL AND cancel_ts IS NOT NULL) AS AvgReqAccept_LaterCancelled,
(SELECT AVG(EXTRACT(EPOCH FROM (cancel_ts - request_ts)) / 60) AS Avg_Reg_Cancel_Minutes
   FROM ride_requests
   WHERE cancel_ts IS NOT NULL) AS AvgRegCancelled,
(SELECT AVG(EXTRACT(EPOCH FROM (pickup_ts - request_ts)) / 60) AS Avg_Req_pickup_Minutes
   FROM ride_requests
   WHERE pickup_ts IS NOT NULL) AS AvgRequest_Pickedup
 FROM
   app_downloads ad
   LEFT JOIN
   signups s
   ON ad.app_download_key = s.session_id
   LEFT JOIN
      ride_requests rr
   ON s.user_id = rr.user_id
   LEFT JOIN transactions tr
   ON rr.ride_id = tr.ride_id
   LEFT JOIN reviews rev
   ON s.user_id = rev.user_id
   LIMIT 1
   ;
```

Total Revenue:

Output:



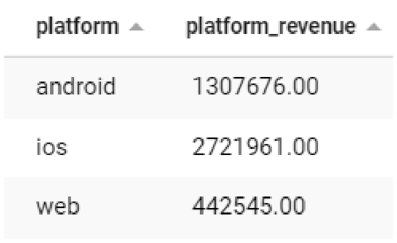
total_revenue
4472182.00

Query used:

```
SELECT SUM(purchase_amount_usd) AS total_revenue
FROM transactions;
```

Revenue: Platform wise

Output:



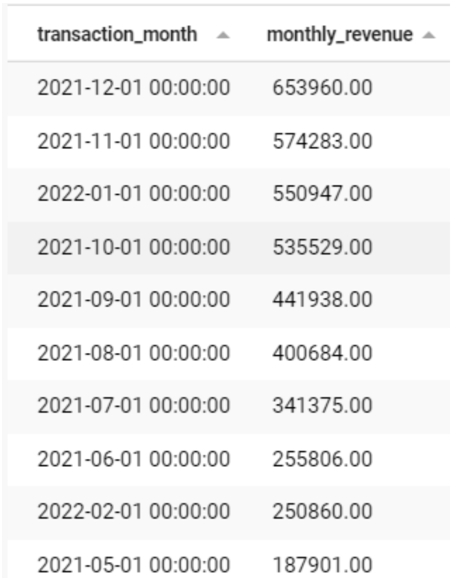
platform	platform_revenue
android	1307676.00
ios	2721961.00
web	442545.00

Query Used:

```
SELECT platform, ROUND(CAST(SUM(purchase_amount_usd) AS INTEGER),2) AS platform_revenue
FROM app_downloads ad
JOIN signups s ON ad.app_download_key = s.session_id
JOIN ride_requests rr ON s.user_id = rr.user_id
JOIN transactions tr ON rr.ride_id = tr.ride_id
GROUP BY platform;
```

Monthly Revenue:

Sample Output:



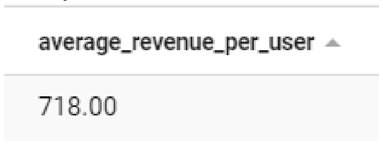
transaction_month	monthly_revenue
2021-12-01 00:00:00	653960.00
2021-11-01 00:00:00	574283.00
2022-01-01 00:00:00	550947.00
2021-10-01 00:00:00	535529.00
2021-09-01 00:00:00	441938.00
2021-08-01 00:00:00	400684.00
2021-07-01 00:00:00	341375.00
2021-06-01 00:00:00	255806.00
2022-02-01 00:00:00	250860.00
2021-05-01 00:00:00	187901.00

Query Used:

```
SELECT DATE_TRUNC('month', transaction_ts) AS transaction_month, SUM(purchase_amount_usd) AS
monthly_revenue
FROM transactions
GROUP BY transaction_month
ORDER BY transaction_month;
```

Average Revenue per user:

Output:



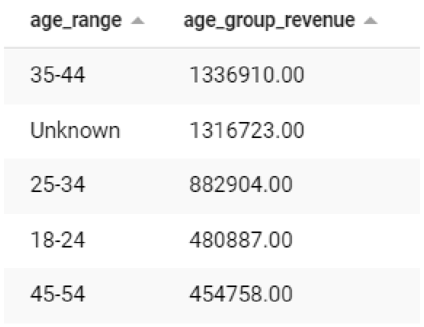
average_revenue_per_user
718.00

Query used:

```
SELECT ROUND(CAST(AVG(total_revenue)AS INTEGER),2) AS average_revenue_per_user
FROM (
  SELECT rr.user_id, SUM(t.purchase_amount_usd) AS total_revenue
  FROM transactions t
       LEFT JOIN ride_requests rr
       ON t.ride_id = rr.ride_id
  GROUP BY user_id ) AS user_revenue;
```

Age Group Revenue:

Output :



age_range	age_group_revenue
35-44	1336910.00
Unknown	1316723.00
25-34	882904.00
18-24	480887.00
45-54	454758.00

Query Used:

```
SELECT s.age_range, ROUND(CAST(SUM(tr.purchase_amount_usd)AS INTEGER),2) AS age_group_revenue
FROM  transactions tr
LEFT JOIN ride_requests rr
ON tr.ride_id = rr.ride_id
LEFT JOIN signups s
ON s.user_id = rr.user_id
GROUP BY 1
ORDER BY 2 DESC;
```

Thanks

Deepthi Binu