‘Cyclistic’ Bike-Share Analysis: How Does A Bike-Share Navigate Speedy Success?

For this case study, I assume the role of a junior data analyst working in the marketing analyst team at *Cyclistic*, a bike-share company in Chicago. The director of marketing believes that the company’s future success depends on maximizing the number of annual memberships. Therefore, my team is tasked to design a new marketing strategy to convert casual riders into annual members. To do that, we need to better understand (1) how annual members and casual riders differ, (2) why casual riders would buy a membership, and (3) how digital media could affect their marketing tactics. The director of marketing specifically assigned me to solve the first problem.

To accomplish my task, I will follow the steps of the data analysis process: **Ask**, **Prepare**, **Process**, **Analyze**, **Share**, and **Act**.

Tools: Spreadsheet (Microsoft Excel), SQL (BigQuery), Tableau Public
Documentation: Visit my **GitHub account.**
Visualization: Go to my **Tableau account.**

Step 1: Ask

Context

In 2016, Cyclistic launched a successful bike-share offering. Since then, the program has grown to a fleet of 5,824 bicycles that are geotracked and locked into a network of 692 stations across Chicago.

Cyclistic offers three pricing plans: single-ride passes, full-day passes, and annual memberships. Customers who purchase the first two are called casual riders, and the rest are considered annual members.

Task

Analyze how casual riders and annual members use Cyclistic bikes differently. Specifically, determine the following:

1. Proportion of **casual riders** and **annual members**
2. Number of rides by **bike type**
3. Frequency of rides by **month** (season)
4. Frequency of rides by **day** of the week (weekday vs weekend)
5. Frequency of rides by **hour** within a day
6. Average **ride duration** by **day** of the week (weekday vs weekend)

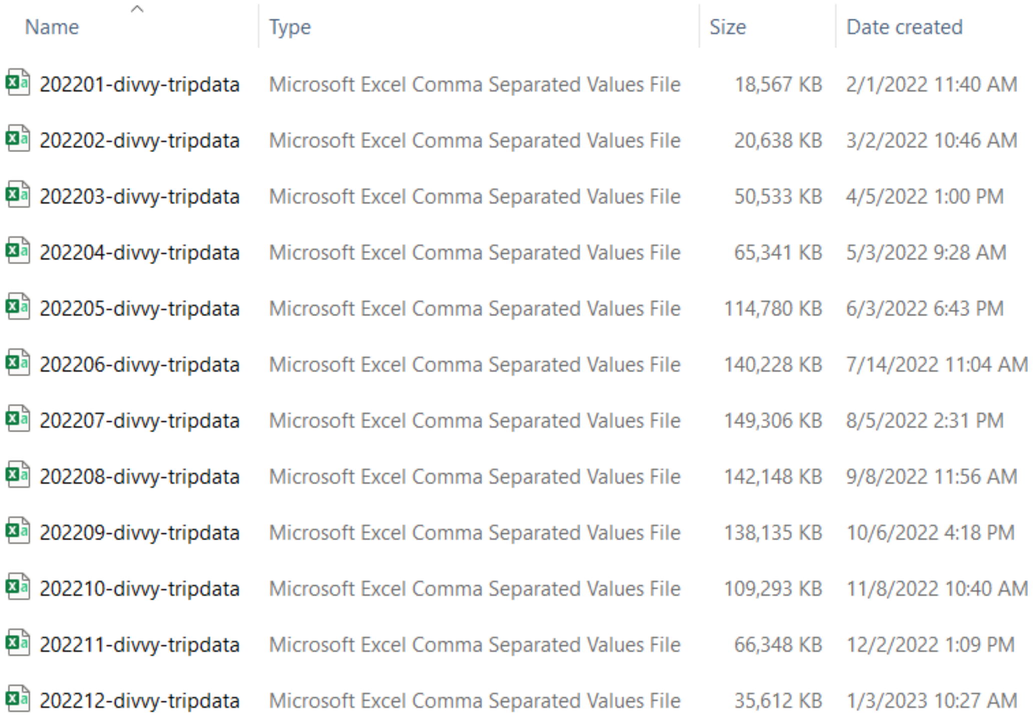
Step 2: Prepare

Access the data

To accomplish my task, I will use Cyclistic's historical trip data for the year 2022. The dataset has been made available by Motivate International Inc. under this license.

Collect the data

The dataset is composed of CSV files containing monthly data. I downloaded 12 files (1 GB in total) corresponding to the months of January-December 2022.



Name	Type	Size	Date created
202201-divvy-tripdata	Microsoft Excel Comma Separated Values File	18,567 KB	2/1/2022 11:40 AM
202202-divvy-tripdata	Microsoft Excel Comma Separated Values File	20,638 KB	3/2/2022 10:46 AM
202203-divvy-tripdata	Microsoft Excel Comma Separated Values File	50,533 KB	4/5/2022 1:00 PM
202204-divvy-tripdata	Microsoft Excel Comma Separated Values File	65,341 KB	5/3/2022 9:28 AM
202205-divvy-tripdata	Microsoft Excel Comma Separated Values File	114,780 KB	6/3/2022 6:43 PM
202206-divvy-tripdata	Microsoft Excel Comma Separated Values File	140,228 KB	7/14/2022 11:04 AM
202207-divvy-tripdata	Microsoft Excel Comma Separated Values File	149,306 KB	8/5/2022 2:31 PM
202208-divvy-tripdata	Microsoft Excel Comma Separated Values File	142,148 KB	9/8/2022 11:56 AM
202209-divvy-tripdata	Microsoft Excel Comma Separated Values File	138,135 KB	10/6/2022 4:18 PM
202210-divvy-tripdata	Microsoft Excel Comma Separated Values File	109,293 KB	11/8/2022 10:40 AM
202211-divvy-tripdata	Microsoft Excel Comma Separated Values File	66,348 KB	12/2/2022 1:09 PM
202212-divvy-tripdata	Microsoft Excel Comma Separated Values File	35,612 KB	1/3/2023 10:27 AM

Examine the data

Upon inspection, each file has the following attributes or headers:

Header	Description
`ride_id`	unique id of the ride
`rideable_type`	type of bike ridden
`started_at`	date and time the ride started
`ended_at`	date and time the ride ended
`start_station_name`	name of the ride’s starting station
`start_station_id`	id of the ride’s starting station
`end_station_name`	name of the ride’s ending station
`end_station_id`	id of the ride’s ending station
`start_lat`	latitude coordinate of the ride’s starting station
`start_lng`	longitude coordinate of the ride’s starting station
`end_lat`	latitude coordinate of the ride’s ending station
`end_lng`	longitude coordinate of the ride’s ending station
`member_casual`	membership type: **`member`** for annual membership riders or **`casual`** for casual riders

The files have the following number of observations or rows:

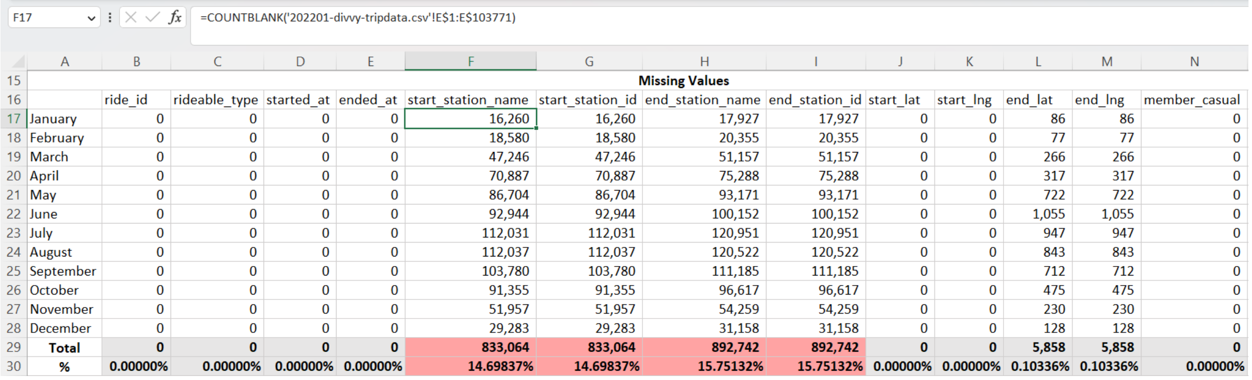
Table		Rows
1	January	103,771
2	February	115,610
3	March	284,043
4	April	371,250
5	May	634,859
6	June	769,205
7	July	823,489
8	August	785,933
9	September	701,340
10	October	558,686
11	November	337,736
12	December	181,807
Total		**5,667,729**

Step 3: Process

Clean the data with spreadsheet

I initially cleaned each file in Microsoft Excel to reduce its size before uploading it to BigQuery (limited to 100 MB).

1. I checked for incomplete values in all columns using the COUNTBLANK function. I found a significant number of missing values in `start_station_name`, `start_station_id`, `end_station_name`, and `end_station_id`, making them unreliable for data analysis. Thus, I decided to delete them from each table.

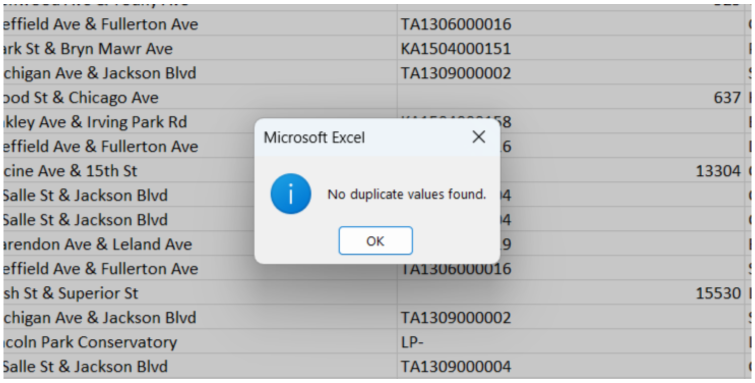


F17 =COUNTBLANK('202201-divvy-tripdata.csv'!E$1:E$103771)

	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_id	end_station_name	end_station_id	start_lat	start_lng	end_lat	end_lng	member_casual
January	0	0	0	0	16,260	16,260	17,927	17,927	0	0	86	86	0
February	0	0	0	0	18,580	18,580	20,355	20,355	0	0	77	77	0
March	0	0	0	0	47,246	47,246	51,157	51,157	0	0	266	266	0
April	0	0	0	0	70,887	70,887	75,288	75,288	0	0	317	317	0
May	0	0	0	0	86,704	86,704	93,171	93,171	0	0	722	722	0
June	0	0	0	0	92,944	92,944	100,152	100,152	0	0	1,055	1,055	0
July	0	0	0	0	112,031	112,031	120,951	120,951	0	0	947	947	0
August	0	0	0	0	112,037	112,037	120,522	120,522	0	0	843	843	0
September	0	0	0	0	103,780	103,780	111,185	111,185	0	0	712	712	0
October	0	0	0	0	91,355	91,355	96,617	96,617	0	0	475	475	0
November	0	0	0	0	51,957	51,957	54,259	54,259	0	0	230	230	0
December	0	0	0	0	29,283	29,283	31,158	31,158	0	0	128	128	0
Total	**0**	**0**	**0**	**0**	**833,064**	**833,064**	**892,742**	**892,742**	**0**	**0**	**5,858**	**5,858**	**0**
%	**0.00000%**	**0.00000%**	**0.00000%**	**0.00000%**	**14.69837%**	**14.69837%**	**15.75132%**	**15.75132%**	**0.00000%**	**0.00000%**	**0.10336%**	**0.10336%**	**0.00000%**

2. I checked for irrelevant values in the remaining columns and noticed that `start_lat`, `start_lng`, `end_lat`, and `end_lng` are unlikely to be useful in addressing my specific tasks. Hence, I decided to also delete them.

3. I checked for duplicate values in `ride_id` using the Remove Duplicates feature. No duplicate values were found in each file.

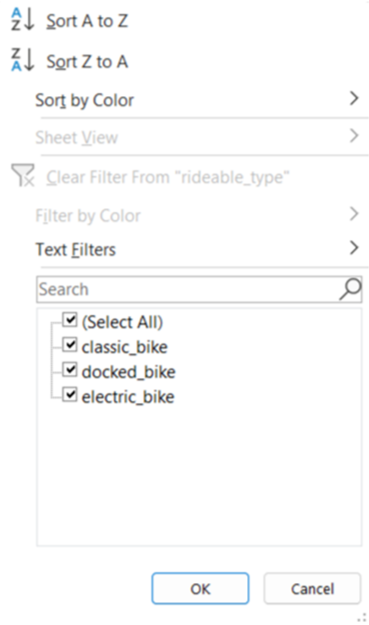


effield Ave & Fullerton Ave	TA1306000016	
ark St & Bryn Mawr Ave	KA1504000151	
chigan Ave & Jackson Blvd	TA1309000002	
ood St & Chicago Ave		637
kley Ave & Irving Park Rd	58	
effield Ave & Fullerton Ave	6	
cine Ave & 15th St		13304
Salle St & Jackson Blvd	4	
Salle St & Jackson Blvd	4	
arendon Ave & Leland Ave	9	
effield Ave & Fullerton Ave	TA1306000016	
sh St & Superior St		15530
chigan Ave & Jackson Blvd	TA1309000002	
coln Park Conservatory	LP-	
Salle St & Jackson Blvd	TA1309000004	

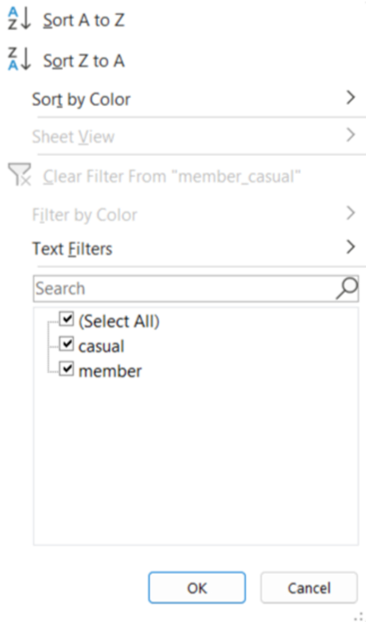
Microsoft Excel

No duplicate values found.

4. I checked for inaccurate values in rideable_type and member_casual. No inaccurate values were found in each file: bike type only includes classic_bike, docked_bike, and electric_bike, and membership type only includes casual and member.



☑ (Select All)
☑ classic_bike
☑ docked_bike
☑ electric_bike



☑ (Select All)
☑ casual
☑ member

5. Since I intend to compare the ride duration of casual riders and annual members, I inserted a column named ride_length computed as ended_at minus started_at. Then, I converted the column into TIME (HH:MM:SS) format.

6. I checked for outliers in ride_length using the Custom Filter feature. I retrieved values less than 0:01:00 (one minute) or greater than 24:00:00 (24 hours) and deleted the rows containing those values.

Table		Deleted Rows	Percentage
1	January	10,326	9.95%
2	February	15,924	13.77%
3	March	32,304	11.37%
4	April	45,822	12.34%
5	May	78,630	12.39%
6	June	98,490	12.80%
7	July	112,308	13.64%
8	August	110,058	14.00%
9	September	97,254	13.87%
10	October	81,456	14.58%
11	November	47,562	14.08%
12	December	29,478	16.21%
Total		**647,304**	**13.40%**

7. Finally, I sorted the table according to `started_at` in ascending order and saved it.

Transform the data with SQL

In BigQuery, I created a dataset named **cyclistic_2022** and uploaded the 12 cleaned tables. Then, I executed the following procedures:

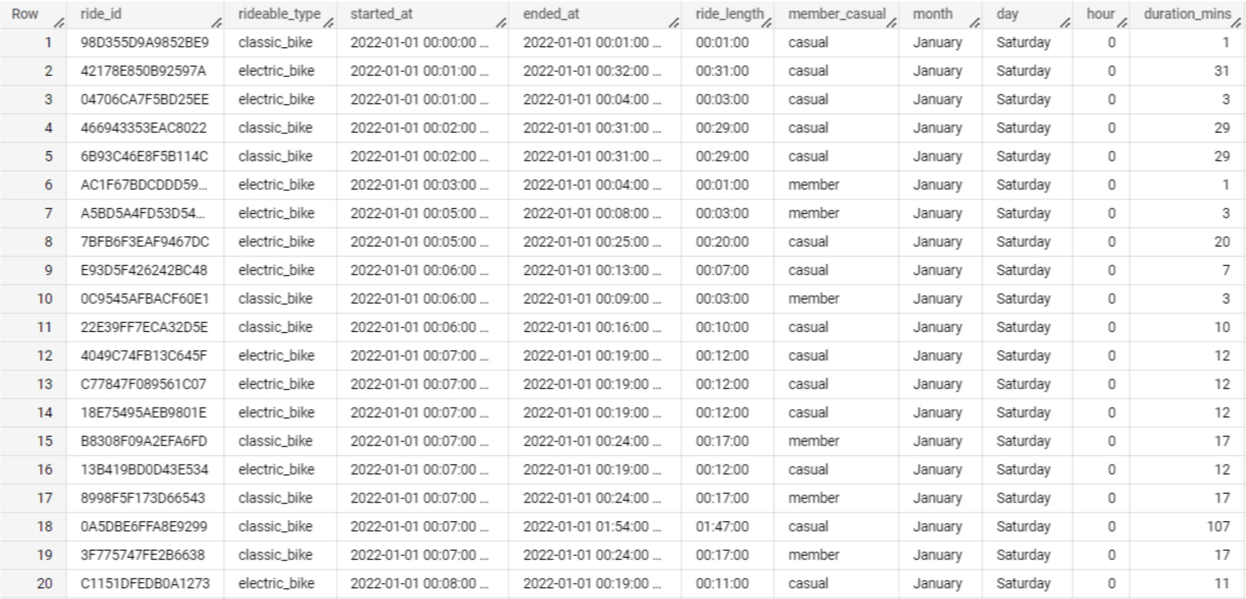
1. I created a table named **all_months** to combine the rows from the 12 tables using the UNION ALL command.

```sql
CREATE TABLE awesome-tempo-374012.cyclistic_2022.all_months AS
  SELECT *
  FROM (
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.January`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.February`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.March`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.April`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.May`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.June`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.July`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.August`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.September`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.October`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.November`
    UNION ALL
    SELECT *
    FROM `awesome-tempo-374012.cyclistic_2022.December`
  );
```

2. From that table, I created a new table named **main_table** to add other columns necessary to accomplish my tasks. Since I intend to compare the frequency of rides by month, day, and hour, I inserted columns for such attributes using the CASE statement and the EXTRACT function. Since I also want to get the average ride duration and realized ride_length (TIME format) cannot be used for the AVG function, I also added a new column named duration_mins (INTEGER format) computed as the minute interval between ended_at and started_at.

```
CREATE TABLE awesome-tempo-374012.cyclistic_2022.main_table AS
  SELECT
    ride_id,
    rideable_type,
    started_at,
    ended_at,
    ride_length,
    member_casual,
    CASE
      WHEN EXTRACT(MONTH FROM started_at) = 1 THEN "January"
      WHEN EXTRACT(MONTH FROM started_at) = 2 THEN "February"
      WHEN EXTRACT(MONTH FROM started_at) = 3 THEN "March"
      WHEN EXTRACT(MONTH FROM started_at) = 4 THEN "April"
      WHEN EXTRACT(MONTH FROM started_at) = 5 THEN "May"
      WHEN EXTRACT(MONTH FROM started_at) = 6 THEN "June"
      WHEN EXTRACT(MONTH FROM started_at) = 7 THEN "July"
      WHEN EXTRACT(MONTH FROM started_at) = 8 THEN "August"
      WHEN EXTRACT(MONTH FROM started_at) = 9 THEN "September"
      WHEN EXTRACT(MONTH FROM started_at) = 10 THEN "October"
      WHEN EXTRACT(MONTH FROM started_at) = 11 THEN "November"
      ELSE "December"
    END AS month,
    CASE
      WHEN EXTRACT(DAYOFWEEK FROM started_at) = 1 THEN "Sunday"
      WHEN EXTRACT(DAYOFWEEK FROM started_at) = 2 THEN "Monday"
      WHEN EXTRACT(DAYOFWEEK FROM started_at) = 3 THEN "Tuesday"
      WHEN EXTRACT(DAYOFWEEK FROM started_at) = 4 THEN "Wednesday"
      WHEN EXTRACT(DAYOFWEEK FROM started_at) = 5 THEN "Thursday"
      WHEN EXTRACT(DAYOFWEEK FROM started_at) = 6 THEN "Friday"
      ELSE "Saturday"
    END AS day,
    EXTRACT(HOUR FROM started_at) AS hour,
    DATE_DIFF(ended_at, started_at, minute) AS duration_mins,
  FROM `awesome-tempo-374012.cyclistic_2022.all_months`
  ORDER BY started_at ASC;
```

Here’s a preview of the latest table:



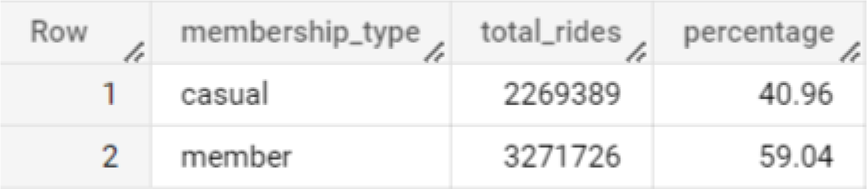
Row	ride_id	rideable_type	started_at	ended_at	ride_length	member_casual	month	day	hour	duration_mins
1	98D355D9A9852BE9	classic_bike	2022-01-01 00:00:00 ...	2022-01-01 00:01:00 ...	00:01:00	casual	January	Saturday	0	1
2	42178E850B92597A	electric_bike	2022-01-01 00:01:00 ...	2022-01-01 00:32:00 ...	00:31:00	casual	January	Saturday	0	31
3	04706CA7F5BD25EE	electric_bike	2022-01-01 00:01:00 ...	2022-01-01 00:04:00 ...	00:03:00	casual	January	Saturday	0	3
4	466943353EAC8022	classic_bike	2022-01-01 00:02:00 ...	2022-01-01 00:31:00 ...	00:29:00	casual	January	Saturday	0	29
5	6B93C46E8F5B114C	classic_bike	2022-01-01 00:02:00 ...	2022-01-01 00:31:00 ...	00:29:00	casual	January	Saturday	0	29
6	AC1F67BDCDDD59...	electric_bike	2022-01-01 00:03:00 ...	2022-01-01 00:04:00 ...	00:01:00	member	January	Saturday	0	1
7	A5BD5A4FD53D54...	electric_bike	2022-01-01 00:05:00 ...	2022-01-01 00:08:00 ...	00:03:00	member	January	Saturday	0	3
8	7BFB6F3EAF9467DC	electric_bike	2022-01-01 00:05:00 ...	2022-01-01 00:25:00 ...	00:20:00	casual	January	Saturday	0	20
9	E93D5F426242BC48	electric_bike	2022-01-01 00:06:00 ...	2022-01-01 00:13:00 ...	00:07:00	casual	January	Saturday	0	7
10	0C9545AFBACF60E1	classic_bike	2022-01-01 00:06:00 ...	2022-01-01 00:09:00 ...	00:03:00	member	January	Saturday	0	3
11	22E39FF7ECA32D5E	classic_bike	2022-01-01 00:06:00 ...	2022-01-01 00:16:00 ...	00:10:00	casual	January	Saturday	0	10
12	4049C74FB13C645F	electric_bike	2022-01-01 00:07:00 ...	2022-01-01 00:19:00 ...	00:12:00	casual	January	Saturday	0	12
13	C77847F089561C07	electric_bike	2022-01-01 00:07:00 ...	2022-01-01 00:19:00 ...	00:12:00	casual	January	Saturday	0	12
14	18E75495AEB9801E	electric_bike	2022-01-01 00:07:00 ...	2022-01-01 00:19:00 ...	00:12:00	casual	January	Saturday	0	12
15	B8308F09A2EFA6FD	classic_bike	2022-01-01 00:07:00 ...	2022-01-01 00:24:00 ...	00:17:00	member	January	Saturday	0	17
16	13B419BD0D43E534	electric_bike	2022-01-01 00:07:00 ...	2022-01-01 00:19:00 ...	00:12:00	casual	January	Saturday	0	12
17	8998F5F173D66543	classic_bike	2022-01-01 00:07:00 ...	2022-01-01 00:24:00 ...	00:17:00	member	January	Saturday	0	17
18	0A5DBE6FFA8E9299	classic_bike	2022-01-01 00:07:00 ...	2022-01-01 01:54:00 ...	01:47:00	casual	January	Saturday	0	107
19	3F775747FE2B6638	classic_bike	2022-01-01 00:07:00 ...	2022-01-01 00:24:00 ...	00:17:00	member	January	Saturday	0	17
20	C1151DFEDB0A1273	electric_bike	2022-01-01 00:08:00 ...	2022-01-01 00:19:00 ...	00:11:00	casual	January	Saturday	0	11

Step 4: Analyze

Once again, I used BigQuery to analyze the data and extract the values I need to accomplish my tasks.

Proportion of casual riders and annual members

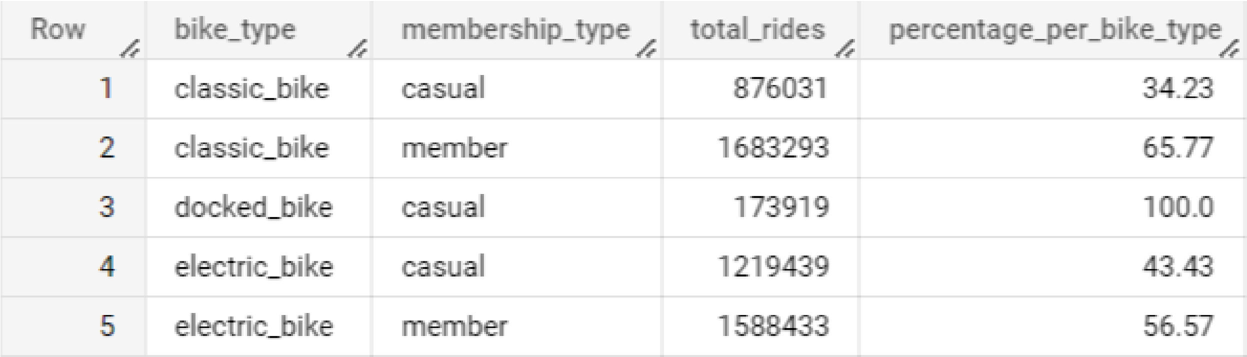
```
SELECT
  member_casual as membership_type,
  COUNT(member_casual) AS total_rides,
  ROUND(COUNT(member_casual) / SUM(COUNT(member_casual)) OVER() * 100, 2) AS percentage
FROM
  `awesome-tempo-374012.cyclistic_2022.main_table`
GROUP BY
  member_casual
ORDER BY
 member_casual;
```



Row	membership_type	total_rides	percentage
1	casual	2269389	40.96
2	member	3271726	59.04

Number of rides by bike type

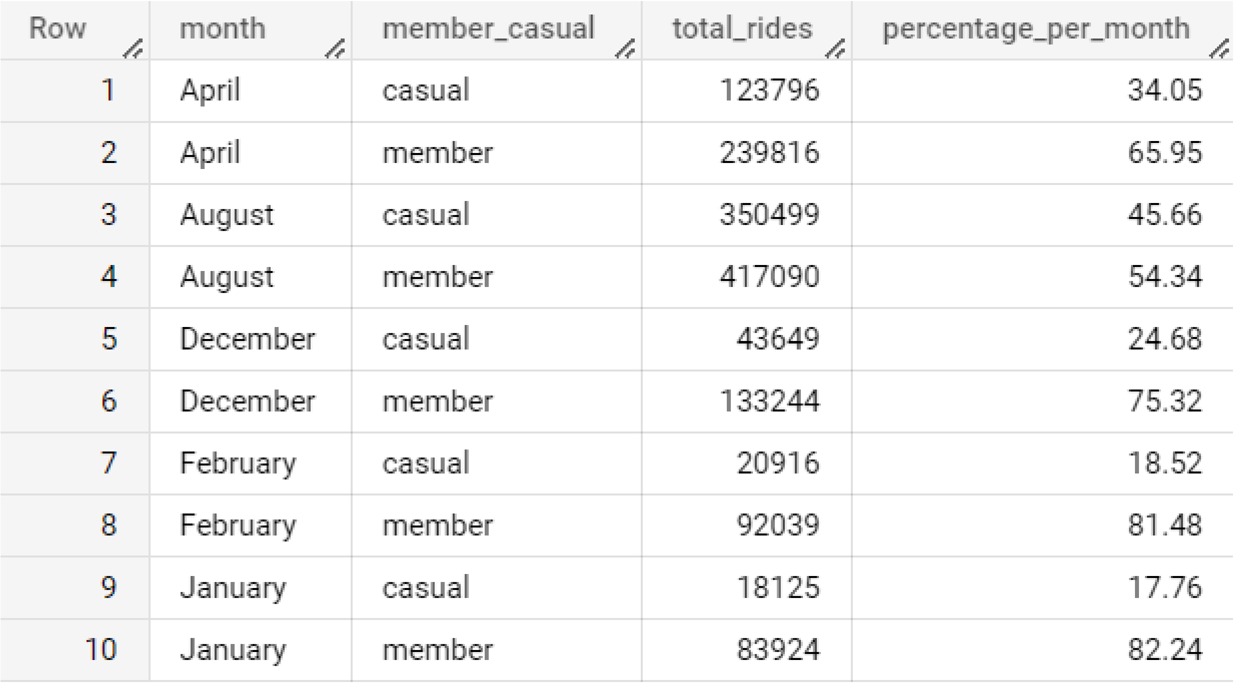
```sql
SELECT
  rideable_type AS bike_type,
  member_casual AS membership_type,
  COUNT(rideable_type) AS total_rides,
  ROUND(COUNT(rideable_type) / SUM(COUNT(rideable_type)) OVER(PARTITION BY rideable_type) * 100, 2) AS percentage_per_bike_type
FROM
  `awesome-tempo-374012.cyclistic_2022.main_table`
GROUP BY
  rideable_type, member_casual
ORDER BY
  rideable_type, member_casual;
```



Row	bike_type	membership_type	total_rides	percentage_per_bike_type
1	classic_bike	casual	876031	34.23
2	classic_bike	member	1683293	65.77
3	docked_bike	casual	173919	100.0
4	electric_bike	casual	1219439	43.43
5	electric_bike	member	1588433	56.57

Frequency of rides by month (season)

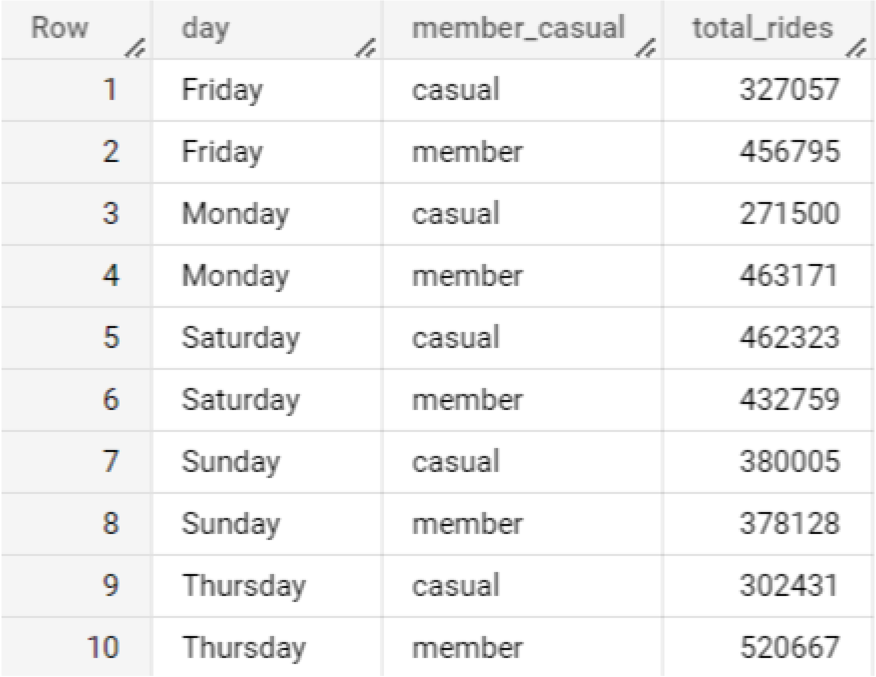
```
SELECT
  month,
  member_casual,
  COUNT(month) AS total_rides,
  ROUND(COUNT(month) / SUM(COUNT(month)) OVER(PARTITION BY month) * 100, 2) AS percent
age_per_month
FROM
  `awesome-tempo-374012.cyclistic_2022.main_table`
GROUP BY
  month, member_casual
ORDER BY
  month, member_casual;
```



Row	month	member_casual	total_rides	percentage_per_month
1	April	casual	123796	34.05
2	April	member	239816	65.95
3	August	casual	350499	45.66
4	August	member	417090	54.34
5	December	casual	43649	24.68
6	December	member	133244	75.32
7	February	casual	20916	18.52
8	February	member	92039	81.48
9	January	casual	18125	17.76
10	January	member	83924	82.24

Frequency of rides by day (weekday vs weekend)

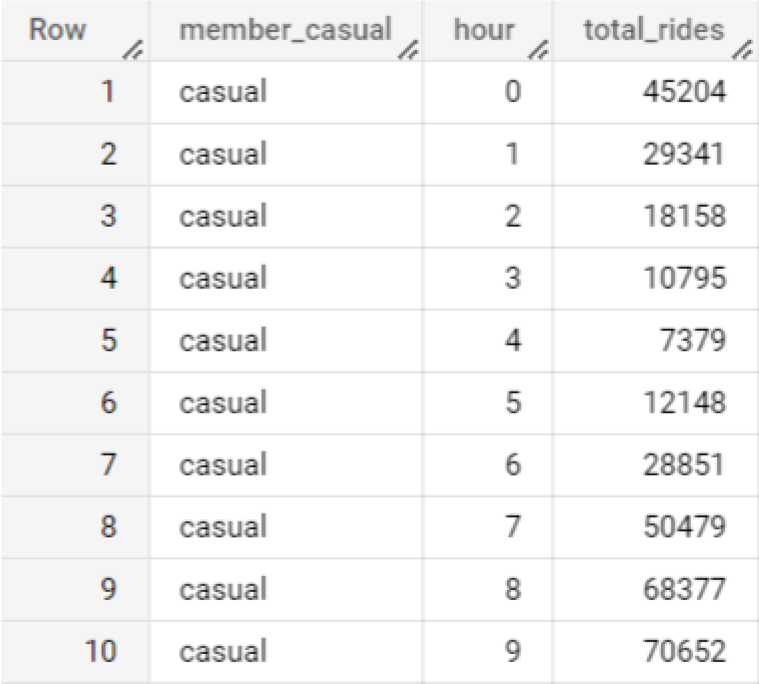
```
SELECT
  day,
  member_casual,
  COUNT(day) AS total_rides
FROM
  `awesome-tempo-374012.cyclistic_2022.main_table`
GROUP BY
  day, member_casual
ORDER BY
  day, member_casual;
```



Row	day	member_casual	total_rides
1	Friday	casual	327057
2	Friday	member	456795
3	Monday	casual	271500
4	Monday	member	463171
5	Saturday	casual	462323
6	Saturday	member	432759
7	Sunday	casual	380005
8	Sunday	member	378128
9	Thursday	casual	302431
10	Thursday	member	520667

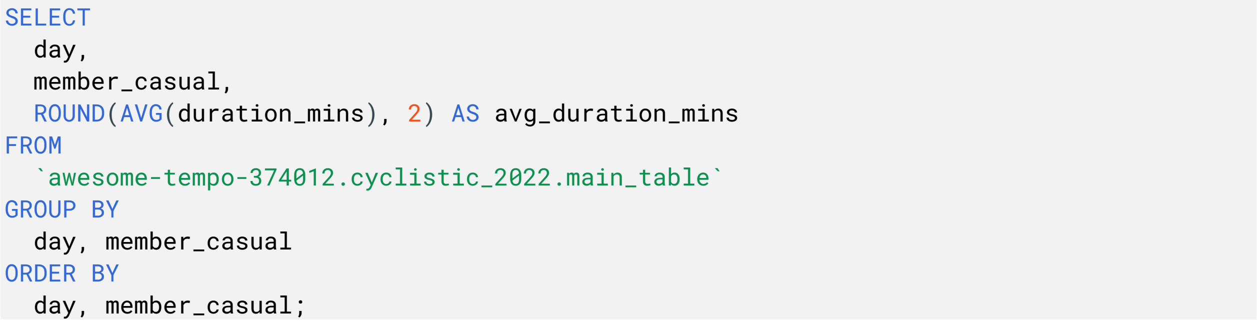
Frequency of rides by hour

```
SELECT
  member_casual,
  hour,
  COUNT(hour) AS total_rides
FROM
  `awesome-tempo-374012.cyclistic_2022.main_table`
GROUP BY
  member_casual, hour
ORDER BY
  member_casual, hour;
```

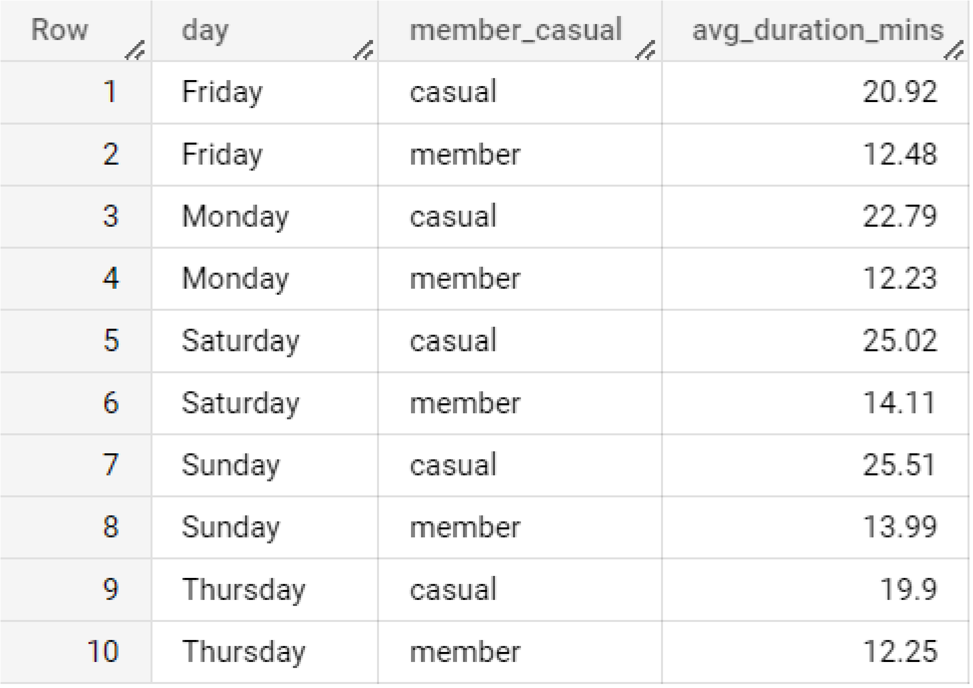


Row	member_casual	hour	total_rides
1	casual	0	45204
2	casual	1	29341
3	casual	2	18158
4	casual	3	10795
5	casual	4	7379
6	casual	5	12148
7	casual	6	28851
8	casual	7	50479
9	casual	8	68377
10	casual	9	70652

Average ride duration by day (weekday vs weekend)



```
SELECT
  day,
  member_casual,
  ROUND(AVG(duration_mins), 2) AS avg_duration_mins
FROM
  `awesome-tempo-374012.cyclistic_2022.main_table`
GROUP BY
  day, member_casual
ORDER BY
  day, member_casual;
```



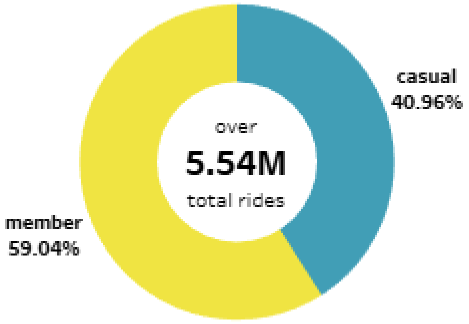
Row	day	member_casual	avg_duration_mins
1	Friday	casual	20.92
2	Friday	member	12.48
3	Monday	casual	22.79
4	Monday	member	12.23
5	Saturday	casual	25.02
6	Saturday	member	14.11
7	Sunday	casual	25.51
8	Sunday	member	13.99
9	Thursday	casual	19.9
10	Thursday	member	12.25

Step 5: Share

I used Tableau Public to visualize the results of my analysis. Visit my **Tableau account** to check the data viz for this project.

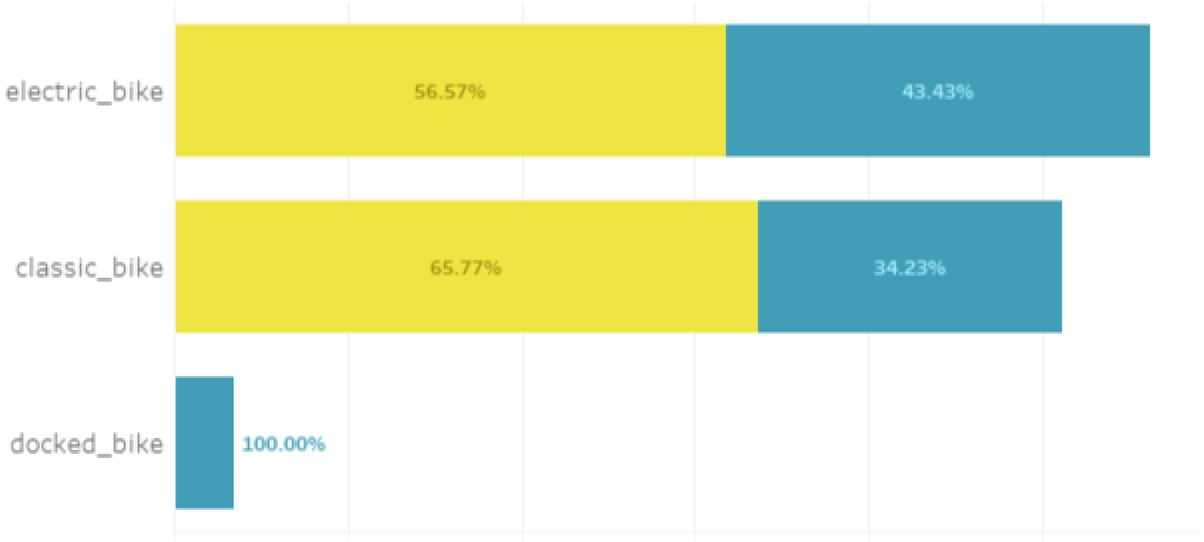
Proportion of casual riders and annual members

A majority of the 5.54 million total rides taken in 2022 were by annual members (59.04%).



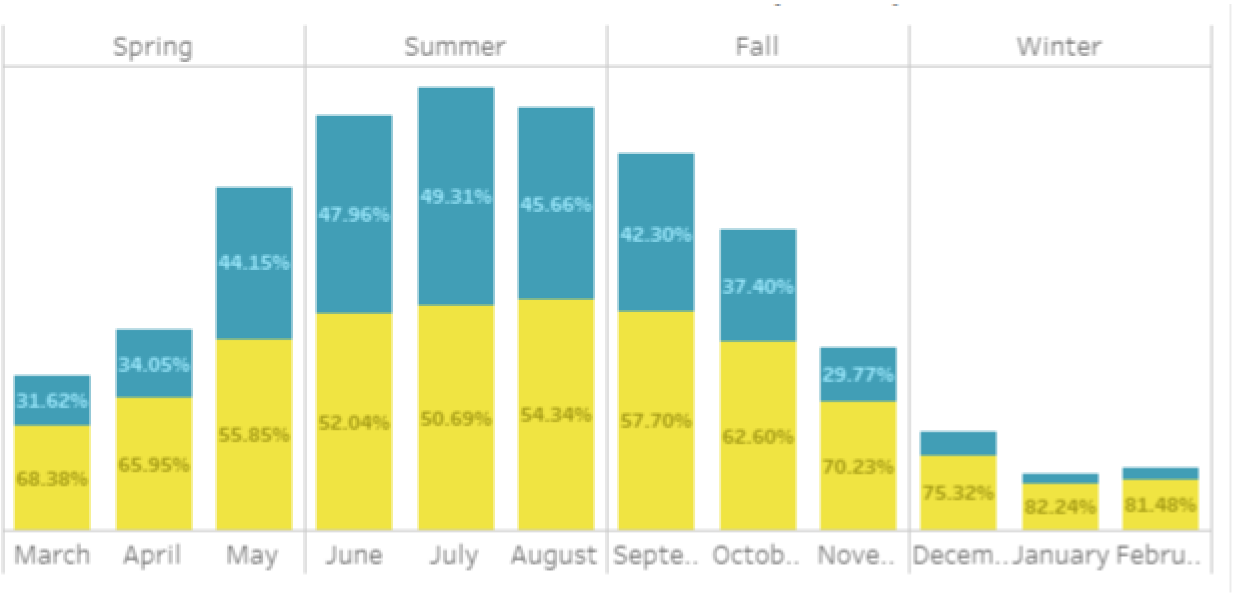
Number of rides by bike type

Classic bikes are preferred by annual members, while electric bikes are preferred by casual riders.



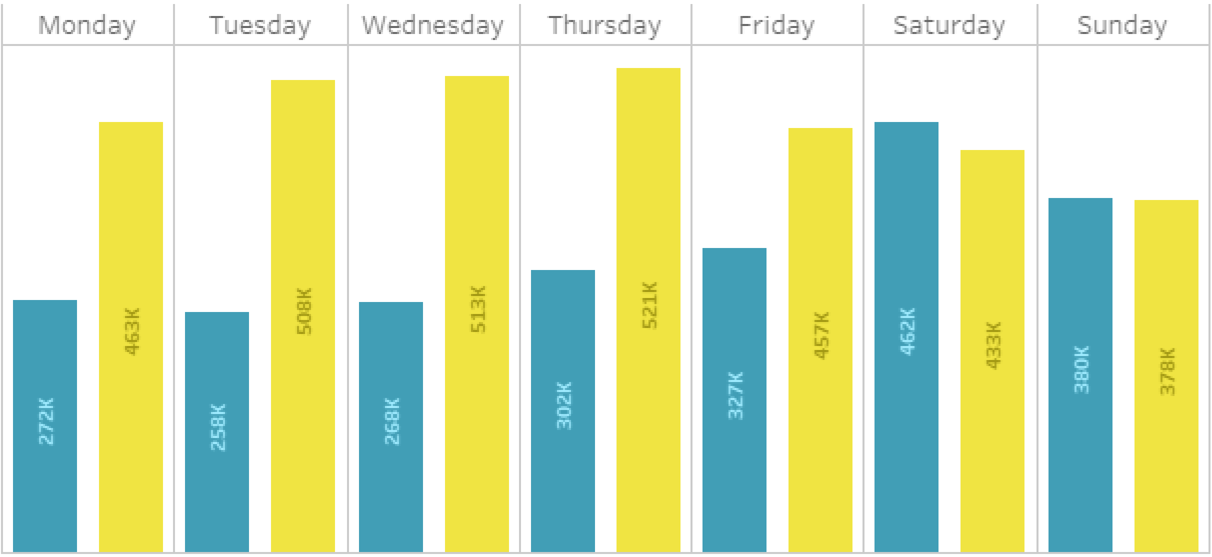
Frequency of rides by month (season)

Cyclistic bikes are rented the most by both annual members and casual riders during the summer months (June to August), and the least during the winter months (December to January).



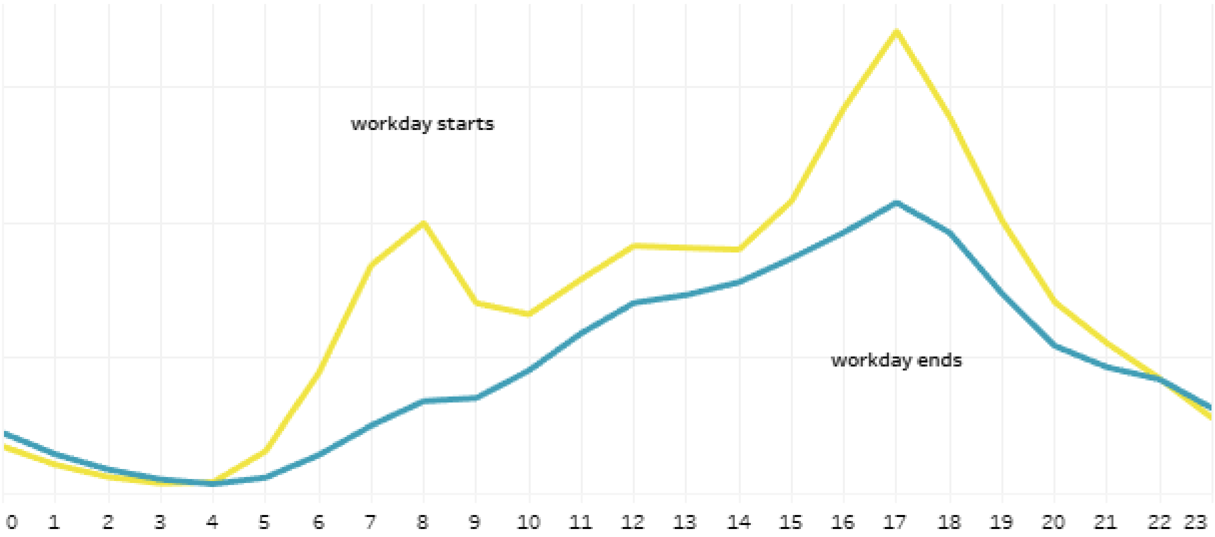
Frequency of rides by day (weekday vs weekend)

Annual members tend to rent bikes more during weekdays, while casual riders rent bikes more during weekends.



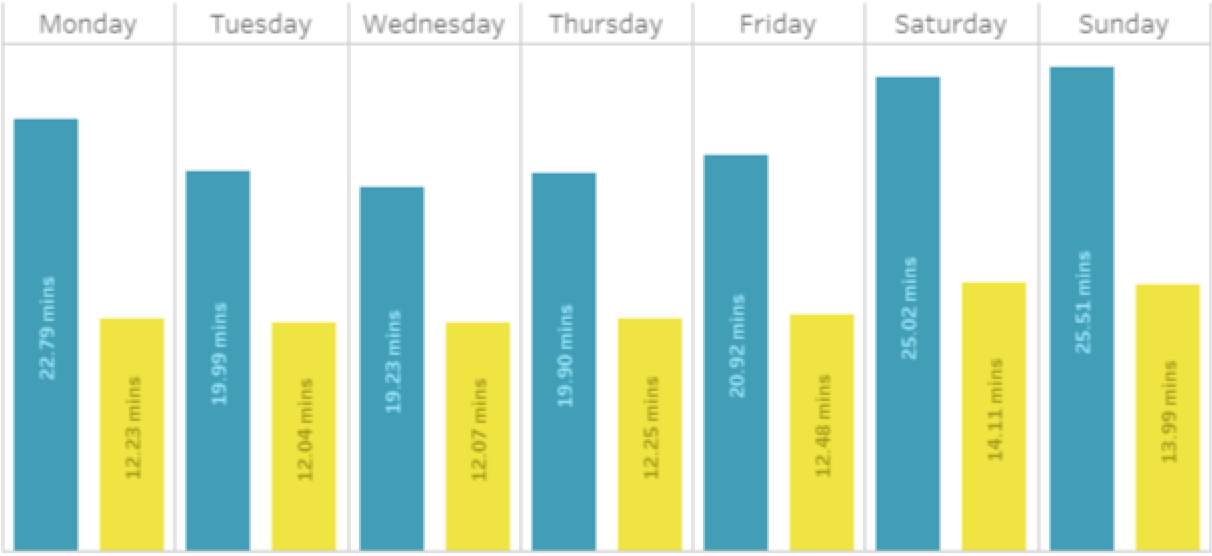
Frequency of rides by hour

Annual members typically rent bikes during rush hours (6-10 AM and 3-7 PM), while casual riders gradually rent bikes from 6 AM until 5 PM.



Average ride duration by day (weekday vs weekend)

On average, casual riders rent bikes for longer periods than annual members, particularly on weekends.



Step 6: Act

Based on these findings, I would recommend the following strategies to help Cyclistic convert more casual riders into annual members:

Expand the electric bike fleet

As casual riders tend to prefer electric bikes, Cyclistic could consider expanding its electric bike fleet to attract more casual riders. This could include offering discounts or promotions for new members who choose the electric bike option.

Promote summer riding

Since both annual members and casual riders rent bikes the most during the summer, Cyclistic could develop a summer promotion campaign that offers discounts or other incentives to encourage casual riders to become annual members during this peak season.

Offer flexible pricing plans

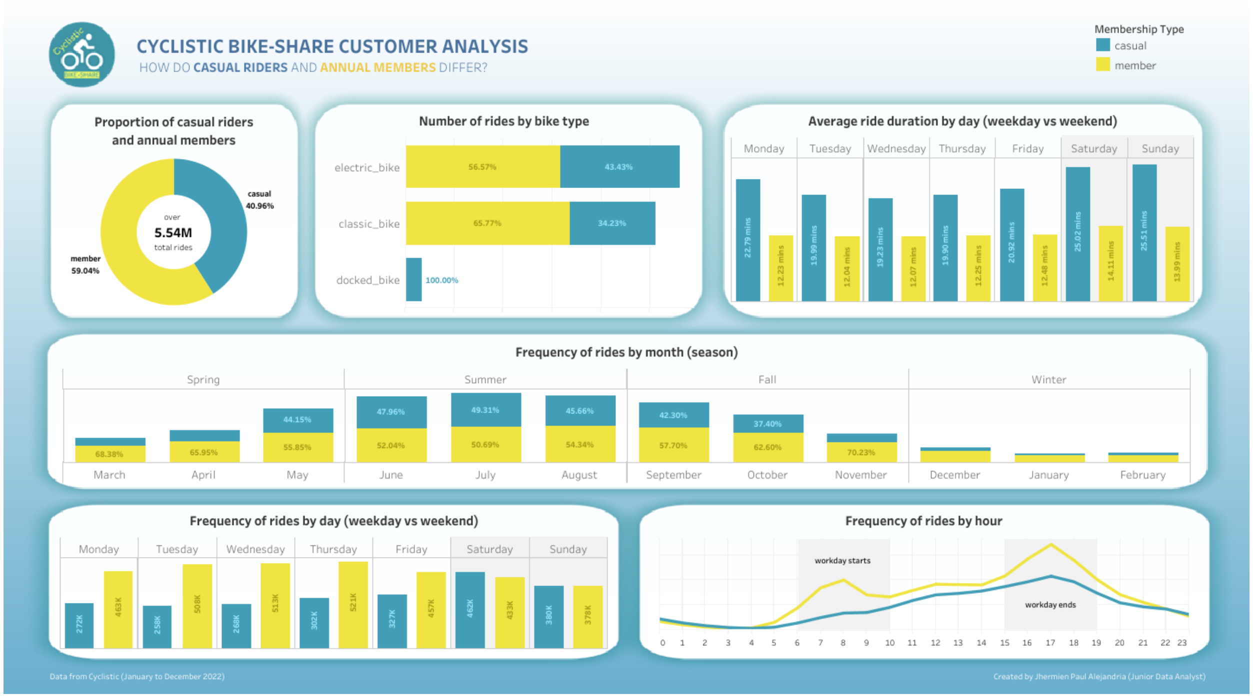
Since casual riders tend to rent bikes more during weekends, Cyclistic could consider offering flexible pricing plans that allow weekend-only usage at a discounted rate. This could appeal to casual riders who may not want to commit to a full annual membership.

Host events during peak hours

Cyclistic could host events during peak hour periods that cater to the needs of casual riders and promote the benefits of becoming an annual member. This could include commuter-focused events, such as group rides or bike safety classes, that help riders feel more comfortable using bikes for commuting purposes.

Offer bike storage solutions

Since casual riders tend to rent bikes for longer periods, Cyclistic could consider offering bike storage solutions at select locations to make it easier for casual riders to store their bikes safely and conveniently during longer rides or breaks.



*This project was completed as part of the requirements for the **Google Data Analytics Professional Certificate** program.*