Cyclistic Data Analysis Divvy 2022 SQL and Power BI

Google Case study 1

Report By: Gayatri Ram Padile

Introduction

This is a case study from Google Data Analytics Professional Certificate Course. Analysis is based on data collected by Divvy and it's users. Divvy is Chicagoland’s bike share system across Chicago and Evanston.

This analysis was created to answer the business question: "How do annual members and casual riders use Cyclistic bikes differently?"

The project was divided into 6 steps.

Step 1 - Ask

The main problem I was trying to solve was to figure out how do annual members and casual riders use Cyclistic bikes differently.

Insights from this analysis can help to make data driven decisions to improve company’s strategy and marketing.

Lily Moreno was the key stakeholder - a director of marketing.

Step 2 - Prepare

I downloaded the data from the Divvy's website.

Data was separated on each month of the year. Data contains information about start and end trip time, bike stations, bike types and rider types.

I haven't noticed any issues with bias or credibility in this data. Data comes from the owner of bikes in association with Chicago City so I could say it is reliable.

There is a Data License Agreement between the Divvy Company and Chicago City. Some part of data is available for analysing it and it’s legal as far as I stick with the Agreement Rules.

Step 3 – Process

At the beginning I chose MS Excel to process the data because of build in functions. Data looked to be well collected and organized. But then I realized it would be better to Union all twelve months data. But merging all twelve csv files is too big data therefore MS Excel was not supporting that much data.

So, I decided to use Postgres SQL because source data were too big for MS Excel.

- The first step was to create twelve tables for twelve months.

```
--prepare phase

SELECT
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
FROM jan_01
```

- Then I Combine all twelve Months data to one table.

```
CREATE TABLE cyclic_bike_share as
SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM jan_01
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM feb_02
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM mar_03
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM arp_04
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM may_05
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
```

```sql
            member_casual

FROM june_06
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM july_07
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM aug_08
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM sep_09
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual

FROM oct_10
UNION ALL

SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual
            FROM nov_11

UNION ALL
SELECT
    ride_id,
            rideable_type,
            started_at,
            ended_at,
            start_station_name,
            end_station_name,
            start_lat,
            start_lng,
            end_lat,
            end_lng,
            member_casual
            FROM dec_12
```

- **Checking null values if any**

  ```
  SELECT * FROM cyclic_bike_share
  where ride_id IS NULL
  OR rideable_type IS NULL
  OR started_at IS NULL
  OR ended_at IS NULL
  OR start_station_name IS NULL
  OR end_station_name IS NULL
  OR start_lat IS NULL
  OR start_lng IS NULL
  OR end_lat IS NULL
  OR end_lng IS NULL
  OR member_casual IS NULL
  ```

- Updating Null values

```
   UPDATE cyclic_bikeshare
   SET end_station_name = 'not_mentioned'
   WHERE end_station_name IS NULL;

   UPDATE cyclic_bikeshare
   SET start_station_name = 'not_mentioned'
   WHERE start_station_name IS NULL;

   SELECT start_station_name
   FROM cyclic_bikeshare
   WHERE start_station_name = 'not_mentioned'

  UPDATE cyclic_bikeshare
  SET end_lat = '0'
 WHERE end_lat IS NULL;

UPDATE cyclic_bikeshare
SET end_lng = '0'
WHERE end_lng IS NULL;
```

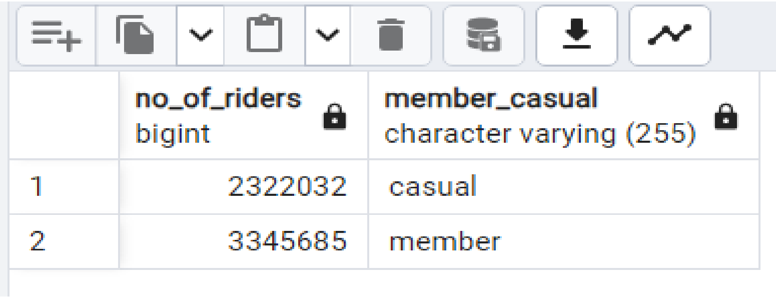
- Removing Duplicates If any

```
SELECT *,
COUNT(*) AS duplicate_values
FROM cyclic_bike_share
GROUP BY
ride_id,
rideable_type,
started_at,
ended_at,
start_station_name,
end_station_name,
start_lat,
start_lng,
end_lat,
end_lng,
member_casual
HAVING COUNT(*)>1
```

Step 4 – Analyze

- Count number of member and casual riders--

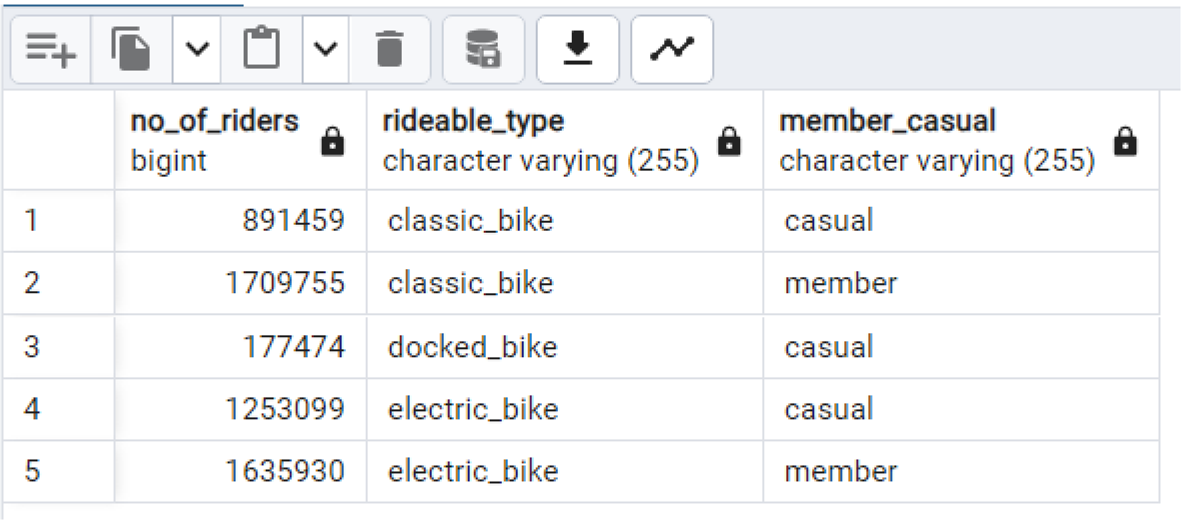
```
SELECT COUNT(ride_id) AS No_of_riders,
member_casual
FROM cyclic_bikeshare
GROUP BY member_casual;
```



	no_of_riders bigint	member_casual character varying (255)
1	2322032	casual
2	3345685	member

- To count number of riders by rideable type of member and causal riders.--

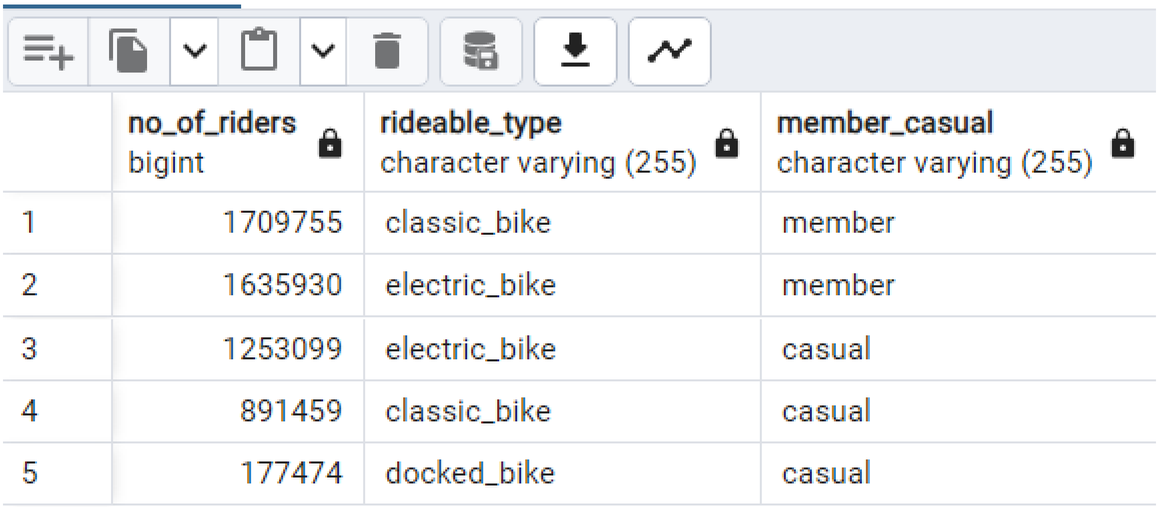
```
SELECT COUNT(ride_id) AS No_of_riders,
rideable_type,
member_casual
FROM cyclic_bikeshare
GROUP BY
rideable_type,
member_casual
ORDER BY
COUNT(ride_id) DESC
```



	no_of_riders bigint	rideable_type character varying (255)	member_casual character varying (255)
1	891459	classic_bike	casual
2	1709755	classic_bike	member
3	177474	docked_bike	casual
4	1253099	electric_bike	casual
5	1635930	electric_bike	member

- Mostly used start station by riders

```
SELECT COUNT(ride_id) AS No_of_riders,
start_station_name,
member_casual
FROM cyclic_bikeshare
GROUP BY
start_station_name,
member_casual
ORDER BY
COUNT(ride_id) DESC
limit 10
```



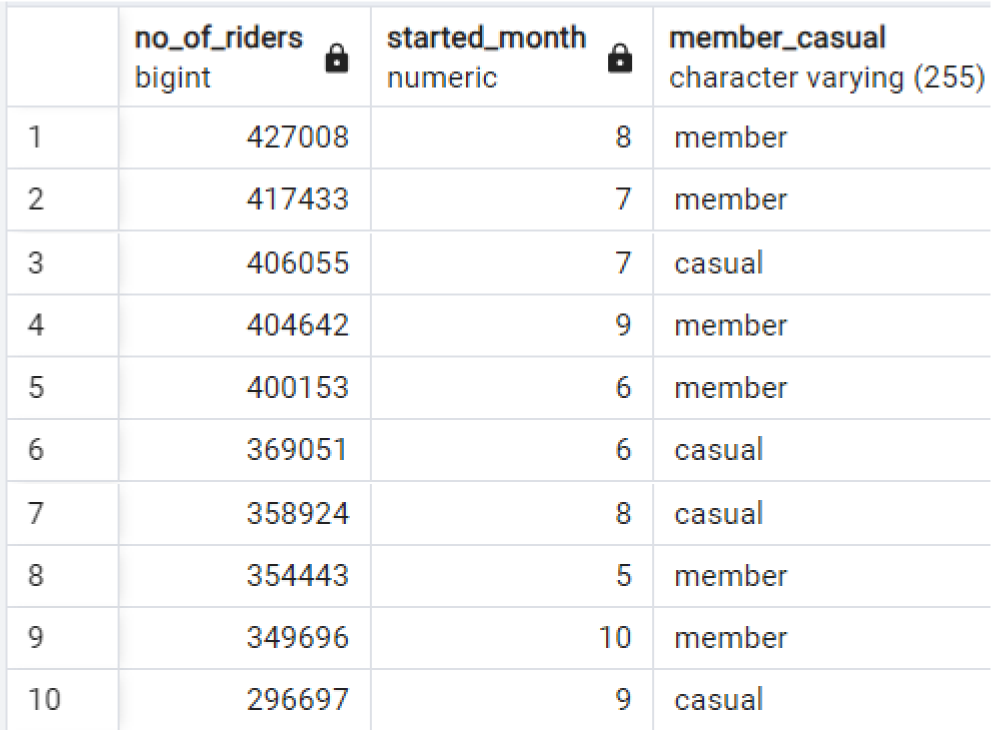
	no_of_riders bigint	rideable_type character varying (255)	member_casual character varying (255)
1	1709755	classic_bike	member
2	1635930	electric_bike	member
3	1253099	electric_bike	casual
4	891459	classic_bike	casual
5	177474	docked_bike	casual

- Mostly used end station by riders

```
SELECT COUNT(ride_id) AS No_of_riders,
end_station_name,
member_casual
FROM cyclic_bikeshare
GROUP BY
end_station_name,
member_casual
ORDER BY
COUNT(ride_id) DESC
limit 10
```

- **started time analysis of member and casual riders**

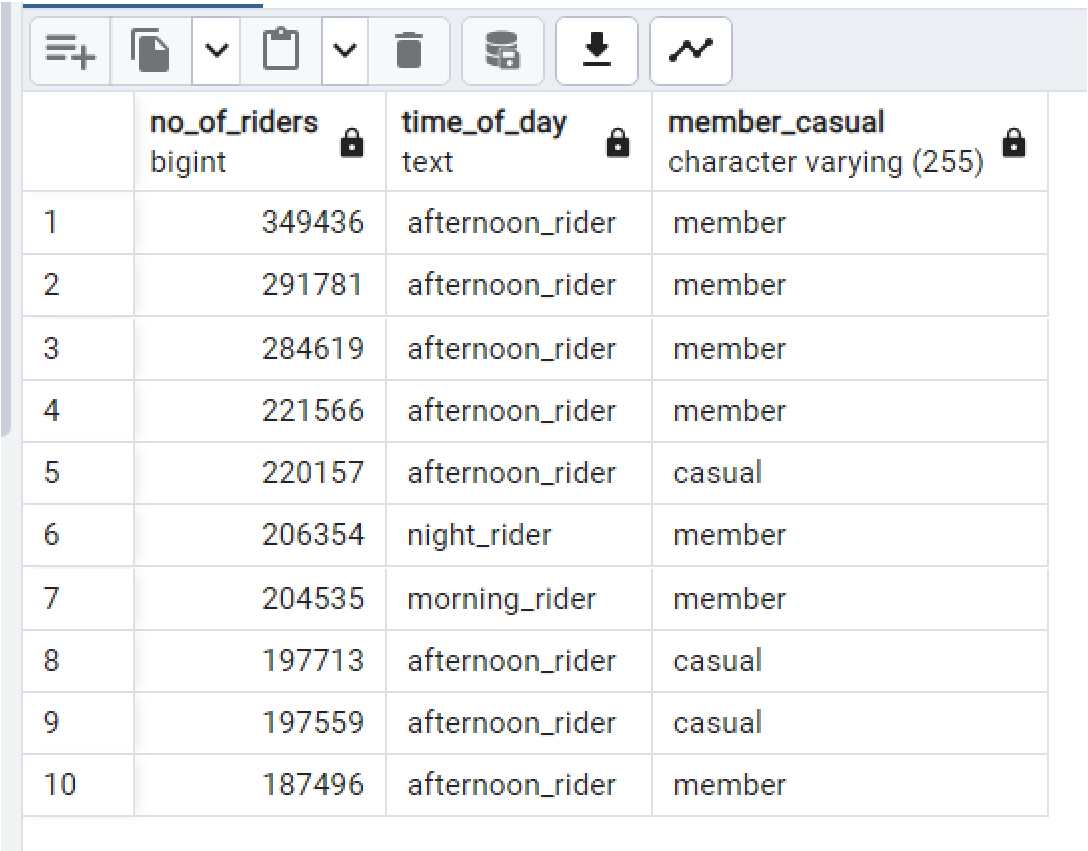
```
--Monthly analysis--
SELECT COUNT(ride_id) AS No_of_riders,
EXTRACT( MONTH FROM started_at) AS Started_month,
--EXTRACT( DAY FROM started_at) AS Started_Day,
--EXTRACT( HOUR FROM started_at) AS Started_Hour,
member_casual
FROM cyclic_bikeshare
GROUP BY
EXTRACT( MONTH FROM started_at),
member_casual
ORDER BY
COUNT(ride_id) DESC
```



	no_of_riders bigint	started_month numeric	member_casual character varying (255)
1	427008	8	member
2	417433	7	member
3	406055	7	casual
4	404642	9	member
5	400153	6	member
6	369051	6	casual
7	358924	8	casual
8	354443	5	member
9	349696	10	member
10	296697	9	casual

```
--Hourly analysis
SELECT
   COUNT(ride_id) AS No_of_riders,
   CASE
      WHEN EXTRACT (HOUR FROM started_at) >= '19' THEN 'night_rider'
      WHEN EXTRACT (HOUR FROM started_at) >= '12' THEN 'afternoon_rider'
      WHEN EXTRACT (HOUR FROM started_at) >= '05' THEN 'morning_rider'
      WHEN EXTRACT (HOUR FROM started_at) >= '00' THEN 'late_night_rider'
   END
    AS time_of_day,
          member_casual
FROM
   cyclic_bikeshare
GROUP BY
EXTRACT( HOUR FROM started_at),
member_casual
ORDER BY
COUNT(ride_id) DESC
limit 10
```

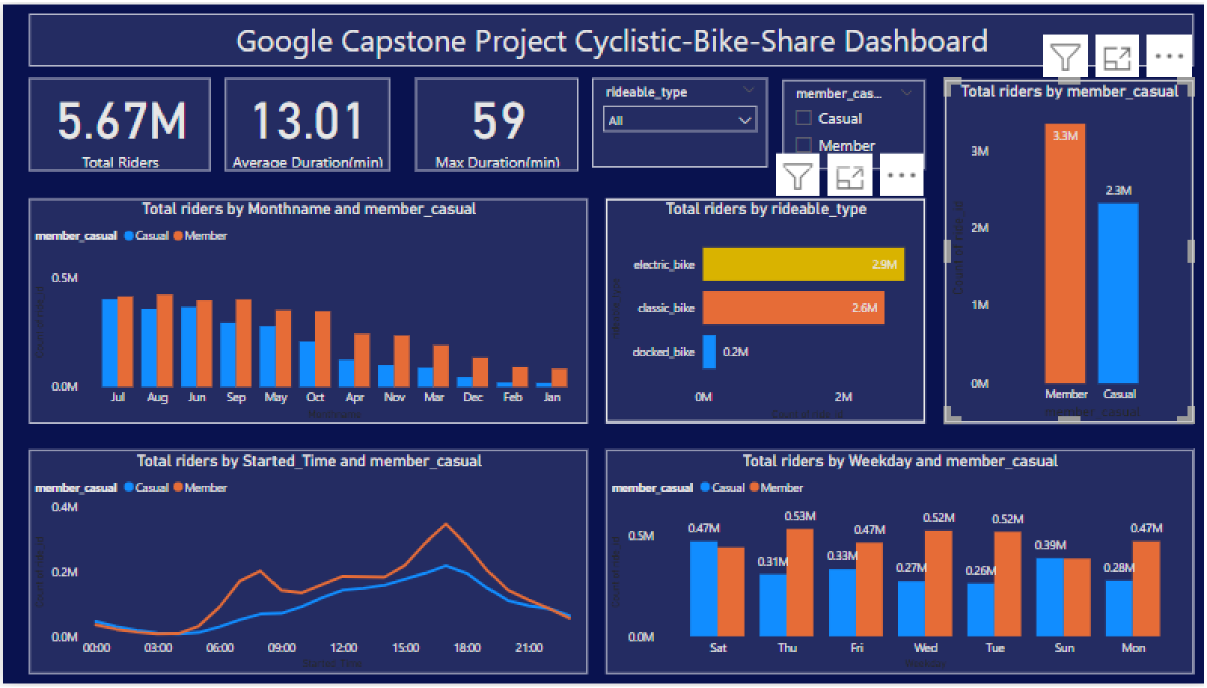
```
--Daily analysis
SELECT COUNT(ride_id) AS No_of_riders,
--EXTRACT( MONTH FROM started_at) AS Started_month,
--EXTRACT( DAY FROM started_at) AS Started_Day,
EXTRACT( HOUR FROM started_at) AS Started_Hour,
member_casual
FROM cyclic_bikeshare
GROUP BY
EXTRACT( HOUR FROM started_at),
member_casual
ORDER BY
COUNT(ride_id) DESC
```



	no_of_riders bigint	time_of_day text	member_casual character varying (255)
1	349436	afternoon_rider	member
2	291781	afternoon_rider	member
3	284619	afternoon_rider	member
4	221566	afternoon_rider	member
5	220157	afternoon_rider	casual
6	206354	night_rider	member
7	204535	morning_rider	member
8	197713	afternoon_rider	casual
9	197559	afternoon_rider	casual
10	187496	afternoon_rider	member

Step 5 - Share

I used Power BI for visualization.



After doing data visualization, I drew the following conclusions:

- There are much more member riders than casual riders in low season.
- In the Month of July, the ratio of riders is similar (0.41M).
- Share of casual riders in the whole riders increases from low season to high season and decreases from high season to low season.
- The most of casual and member riders uses bikes between 12 p.m. and 7 p.m.
- Share of casual riders in the whole riders is the biggest on Saturdays and Sundays and especially in the month of June and July.
- Average ride Duration of riders remains on the same level for the whole year (13:1 min).
- The maximum Average time of riders is minutes.

Step 6 - Act

The director of marketing believes the company’s future success depends on maximizing the number of annual memberships.

Therefore, I prepared top three recommendations based on my analysis.

Target the marketing campaign at casual riders who:

1. Use bikes on Saturdays and Sundays in the months from March to November. Also Thursdays in the months.
2. Use bikes between 12 p.m. and 7 p.m.
3. Average length of rides exceeds 30 minutes on Thursdays, Saturdays and Sundays in the months from March to November.

The company should prepare a promotional campaign for people riding these days/times of days/this long, present the benefits for new members of the annual subscription to encourage them to become members.