Cyclistic Capstone Project

Data Analyst: Sebastián Cantergiani

Client: Cyclistic

Date: February 2023

Introduction:

The Cyclistic case study is a capstone project for Google Data Analytics Professional Course that will go through each step of the data analysis process. Which are; ask, prepare, process, analyze, share and act. More details of the case study can be found here.

Deliverables:

1. A clear statement of the business task.
2. A description of all data sources used.
3. Documentation of any cleaning or manipulation of data.
4. A summary of the analysis.
5. Supporting visualizations and key findings.
6. Top three recommendations based on the analysis.

Tools Used:

- Excel - check file integrity.
- SQL - for data preparation and processing.
- Power BI - for further analysis and data visualizations.
- PowerPoint - for data visualization presentations.
- Github- for store codes and changelogs into notebooks.

Resources:

- Link to the presentation can be found here.
- Further detail on the dashboard can be seen here.

ASK

Purpose:

Cyclistics wants to maximize the number of annual memberships by converting casual riders into annual members.

Key Stakeholders:

- Director of marketing - Lily Moreno.
- Cyclistic marketing analytics team.
- Cyclistic executive team.

Business Task:

1. Examine how annual members and casual riders use Cyclistic bikes differently in the last 12 months.
2. Why would casual riders buy Cyclistic annual memberships?
3. How can Cyclistic use digital media to influence casual riders to become members?

Scope of Work and Limitations:

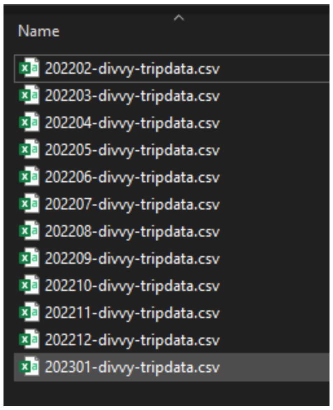
For the purpose of this case study we will only focus on Identifying differences between Cyclistic casual and members riders using bicycles in the past 12 months.

PREPARE

Data source:

First hand data coming from Cyclistic cloud storage. This study used a monthly trip dataset from January 2022 to December 2022. More detail can be found here.

The files were downloaded as CSV and stored locally in a folder using the file convention “YYYYMM divvy-tripdata.CSV”.



```
Name
202202-divvy-tripdata.csv
202203-divvy-tripdata.csv
202204-divvy-tripdata.csv
202205-divvy-tripdata.csv
202206-divvy-tripdata.csv
202207-divvy-tripdata.csv
202208-divvy-tripdata.csv
202209-divvy-tripdata.csv
202210-divvy-tripdata.csv
202211-divvy-tripdata.csv
202212-divvy-tripdata.csv
202301-divvy-tripdata.csv
```

Privacy:

Data-privacy issues prohibit using riders’ personally identifiable information. More detail can be found here.

Transformation:

1. The files were inspected in Excel in order to check the data integrity, for consistency, accuracy and completeness. No duplicates were found and all files were consistent in their headings showing as follows:
    a. ride_id
    b. rideable_type
    c. started_at
    d. ended_at
    e. start_station_name
    f. start_station_id
    g. end_station_name
    h. end_station_id
    i. start_lat
    j. start_lng
    k. end_lat
    l. end_lng
    m. member_casual

    However after applying filters to the data, the completeness in start and end stations may be compromised due to missed information. This can lead to sample bias and will be further investigated in the process phase.

2. Once inspected in Excel, each file was uploaded to a new BigQuery database named "cyclistic_data".

3. Added new columns:
    a. ride_length : calculated minutes between ended_at and started_at with “DATE_DIFF” function.
    b. day_of_week: extract the number of the week for the started_at date with “EXTRACT” function.

4. Merge the datasets addressing duplicates with the DISTINCT and UNION DISTINCT.

5. Saved the query in a new view called "2022_tripdata" containing the following code:

```
-- Merge all tables and add columns ride_length and day_of_week --

SELECT
 DISTINCT *, date_diff(ended_at,started_at,MINUTE) AS ride_length,
EXTRACT(DAYOFWEEK FROM started_at) AS  day_of_week
FROM
(SELECT * FROM `cyclistic_data.202201-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202202-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202203-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202204-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202205-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202206-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202207-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202208-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202209-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202210-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202211-divvy-tripdata`
  UNION DISTINCT SELECT * FROM `cyclistic_data.202212-divvy-tripdata`)

-- Note: we have 5436715 rows --
```

PROCESS

Data validation:

1. Check Duplicates:

```
SELECT
  DISTINCT COUNT(ride_id)
FROM
  `cyclistic_data.2022_tripdata`
```

Row count did not change, that means there are no duplicates.

2. Check if there is NULL values:

```
SELECT
  COUNT(*)
FROM
  `cyclistic_data.2022_tripdata`
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 start_station_id IS NULL
  OR end_station_name IS NULL
  OR end_station_id 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
  OR ride_length IS NULL
  OR day_of_week IS NULL)
```

1067355 rows with at least one NULL cell.

3. Check for anomalies in the added columns ride_length and day_of_week:

```sql
SELECT
  MIN(ride_length),MAX(ride_length),
FROM
  `cyclistic_data.2022_tripdata`
```

```sql
SELECT
  DISTINCT(day_of_week),
FROM
  `cyclistic_data.2022_tripdata`
```

No anomalies found.

Data cleaning:

SQL

1. Removed rows with NULL cells:

```
SELECT
  *
FROM
  `cyclistic_data.2022_tripdata`
WHERE NOT (
  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 start_station_id IS NULL
  OR end_station_name IS NULL
  OR end_station_id 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
  OR ride_length IS NULL
  OR day_of_week IS NULL)

-- Note: rows went from 5436715 to 4369360. This represent over 20% of our
data --
```

A considerable amount of NULL values are coming from station_name and station_id, which represent around 20% of the size of the initial dataset. Proceeding with this step will imply a risk of missing important data for the analysis. That being said, we proceed to replace the NULL values instead to avoid sample bias.

In an ideal situation we would ask Cyclistic if this could mean that users also pick and drop bicycles on non-station points, or if this is related to a software issue.

2. Replaced NULL values, filtered ride_length dates below 0, trimmed all strings to ensure consistency and sorted the data:

```
SELECT
  TRIM(ride_id) AS ride_id,
  TRIM(rideable_type) AS rideable_type,
  started_at,
  ended_at,
  ride_length,
  day_of_week,
  TRIM(COALESCE(start_station_name,"N/A")) AS start_station_name, --
Replace NULL values.
  TRIM(COALESCE(start_station_id,"N/A")) AS start_station_id,
  TRIM(COALESCE(end_station_name, "N/A")) AS end_station_name,
  TRIM(COALESCE(end_station_id, "N/A")) AS end_station_id,
  start_lat,
  start_lng,
  end_lat,
  end_lng,
  TRIM(member_casual) AS member_casual
FROM
  `cyclistic_data.2022_tripdata`
WHERE
  ride_length > 0
ORDER BY started_at ASC

-- This will be our cleaned SQL table for further analysis.
-- Row count at this point is 5328012
```

This will be our cleaned SQL table for further analysis. Row count at this point is 5328012.

3. Saved the the cleaned data as a view called ‘202202_tripdata_cleaned’

ANALYZE

SQL

1. Calculated mean and max of ride_length. Also inspect the mode of day_of_week:

AVG( ), MAX ()

```
SELECT
  AVG(ride_length) AS avg_ride_length, MAX(ride_length) AS max_ride_length
FROM
  `cyclistic_data.202202_tripdata_cleaned`
WHERE
  ride_length > 0

  -- Results:
  --AVG: 19.2058585
  --MAX: 41387
```

Ride_id count, avg_ride_length, max_ride_length by segment

```
SELECT
  member_casual,COUNT(*) AS num_of_rides,AVG(ride_length) AS
avg_ride_length, MAX(ride_length) AS max_ride_length
FROM
  `cyclistic_data.202202_tripdata_cleaned`
WHERE
  ride_length > 0
GROUP BY
  member_casual
--Results: member_casual | num_of_rides | avg_ride_length |max_ride_length
--         member        |  3143445     | 12.5638         | 1559
--         casual        |  2184567     | 28.7632         | 41387
```

Members rotate faster, although casual riders have more than double ride durations on average.

Day_mode:

```
SELECT
  APPROX_TOP_COUNT(day_of_week, 7) AS day_mode -- 7 represent the number
of values to bring the mode.
FROM
  `cyclistic_data.202202_tripdata_cleaned`
WHERE
  ride_length > 0

  -- Results:
  --Day Nº | Count
  --   7   | 861084
  --   5   | 792110
  --   4   | 755137
  --   6   | 754496
  --   3   | 734095
  --   1   | 728420
  --   2   | 702670
```

Saturday is the most frequent day for bike riding.

Day_mode by type of user:

```
SELECT
  member_casual, APPROX_TOP_COUNT(day_of_week, 1) AS day_mode
FROM
  `cyclistic_data.202202_tripdata_cleaned`
WHERE
  ride_length > 0
GROUP BY
  member_casual

--Results: member_casual | day_mode | count_day_mode
--             casual    |   7      | 446447
--             member    |   5      | 501335

Week numbers Sunday = 1 and Saturday 7.
```

Rotation for casual riders is higher on Saturdays over any other day meanwhile members rotate more on thursdays.

Avg_ride by day_of_week

```
SELECT
   day_of_week, avg(ride_length) AS avg_ride_length
FROM
  `cyclistic_data.202202_tripdata_cleaned`
GROUP BY day_of_week
ORDER BY avg(ride_length) DESC

-- Results: day_of_week | avg_ride_length |
--                  1   |  23.9135086     |
--                  7   |  23.4546722     |
--                  6   |  18.7637588     |
--                  2   |  18.1948055     |
--                  5   |  16.8689866     |
--                  3   |  16.4938298     |
--                  4   |  16.2901142     |

Week numbers Sunday = 1 and Saturday 7.
```

On average rides are longer Saturdays and Sundays for both casual and member riders.

Rideable_type by type of user:

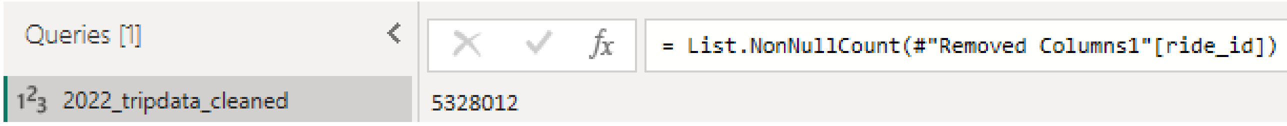
```
SELECT
   member_casual, rideable_type, count(rideable_type) AS
count_rideable_type
FROM
  `cyclistic_data.202202_tripdata_cleaned`
GROUP BY member_casual, rideable_type
ORDER BY count(rideable_type) DESC

-- Results: member_casual | rideable_type | count_rideable_type
--          member        | classic_bike  | 1685694
--          member        | electric_bike | 1457751
--          casual        | electric_bike | 1129727
--          casual        | classic_bike  | 879193
--          casual        | docked_bike   | 175647
```

From the analysis we can observe that members use more classic bikes while casual riders prefer electric bikes. On the other hand docked bikes have only been used by non member users in the last 12 months.

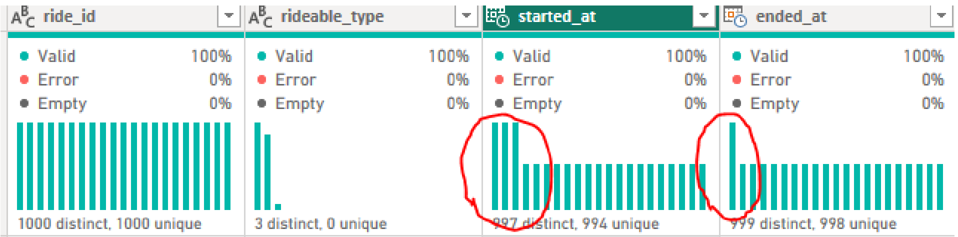
Power BI Recheck

1. The dataset cleaned previously in SQL is loaded into Power Query inside Power BI for further inspection. First we proceed to check if the row counts match with our original file by clicking ride_id > transform > statistics > count values.

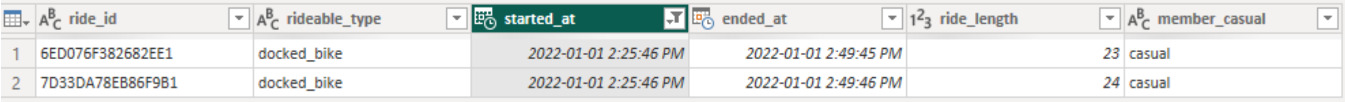


```
= List.NonNullCount(#"Removed Columns1"[ride_id])
5328012
```

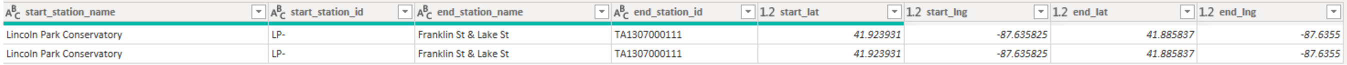
2. Checking the column distributions we see a possible duplicate, so we investigate further to check it.



ride_id		rideable_type		started_at		ended_at	
● Valid	100%	● Valid	100%	● Valid	100%	● Valid	100%
● Error	0%	● Error	0%	● Error	0%	● Error	0%
● Empty	0%	● Empty	0%	● Empty	0%	● Empty	0%
1000 distinct, 1000 unique		3 distinct, 0 unique		997 distinct, 994 unique		999 distinct, 998 unique	



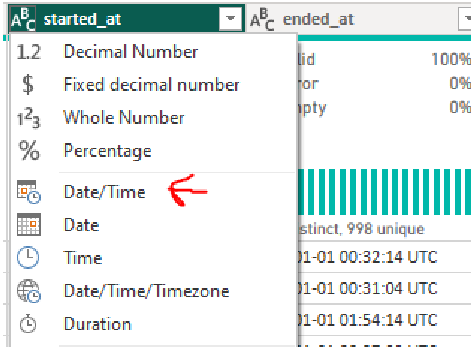
	ride_id	rideable_type	started_at	ended_at	ride_length	member_casual
1	6ED076F382682EE1	docked_bike	2022-01-01 2:25:46 PM	2022-01-01 2:49:45 PM	23	casual
2	7D33DA78EB86F9B1	docked_bike	2022-01-01 2:25:46 PM	2022-01-01 2:49:46 PM	24	casual



start_station_name	start_station_id	end_station_name	end_station_id	start_lat	start_lng	end_lat	end_lng
Lincoln Park Conservatory	LP-	Franklin St & Lake St	TA1307000111	41.923931	-87.635825	41.885837	-87.6355
Lincoln Park Conservatory	LP-	Franklin St & Lake St	TA1307000111	41.923931	-87.635825	41.885837	-87.6355

In an ideal situation we would have asked the data engineer of Cyclistic to check why this could have happened. For the sake of this case study we assume it's possible to exist a datetime with the same coordinate for the starting and ending point.

3. Data transformation must be done in columns started_at and ended_at to be recognized as date format.
    a. Extract > Text before delimiter
    b. Change type > Date / Time



4. Modified the day_of_week column to start on monday instead of sunday:

```
day_of_week = WEEKDAY('2022_tripdata_cleaned'[started_at],2)
```

SHARE

1. Dashboard:

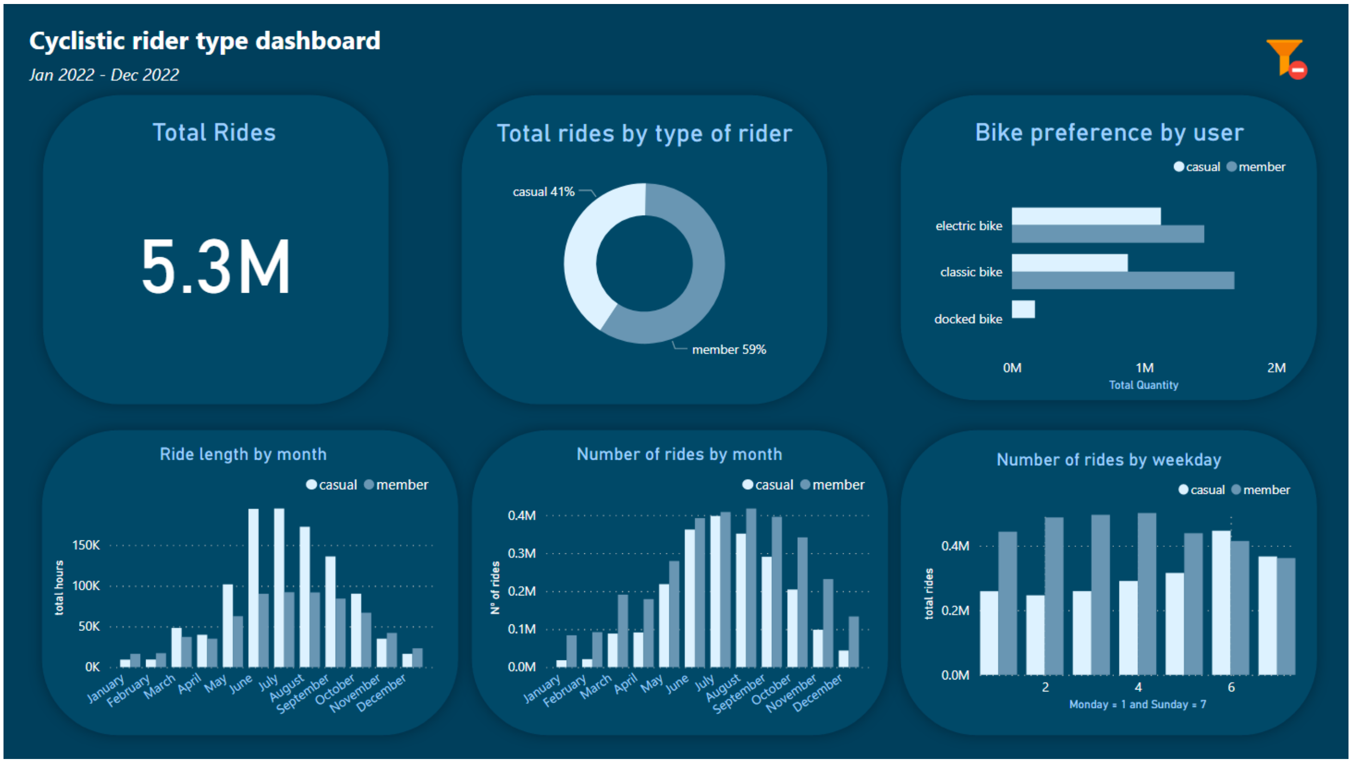
For the purpose of this project we will be making a dashboard in Power BI and a presentation for our stakeholders. That being said we created the following type of visualizations:

- Label - Shows total rides of 2022
- Donough chart - Shows participation of total rides segmented by user type.
- Bar chart - Shows bike preference by user.
- Column chart
    - Ride length by month - Indicates the total ride time in hours for each month by type of client.
    - Number of rides by month - Indicates how many times the service was used by type of client.
    - Number of rides by weekday - Shows the weekly rotation for Cyclistic bike rent services by user type.

2. Meaning and refining:

In order to ensure meaning and refining data visualization

a. A title and a subtitle was added to add context to the analysis and indicate the scope of work.
b. ‘Y- axis’ start point set to 0 for accurate proportions.
c. Carefully choose a color palette, shape and size in order to contrast and draw the most important findings.
d. Sync filters to better show findings.
e. Added a button to easily remove all filters.



3. Presentation:

Later, the graphs that show key finding were exported to create the presentation found <insert link>.

a. Added an introduction slide with the name of the analysis and the year of study.

b. Added context by introducing the objectives.

c. Segmented the key findings and ordered them to show broad findings first then details.

d. Added small entry effects in order to maintain focus on key elements.

e. Added a small description of findings and a more detailed description in the speaker notes.

f. Inserted conclusions and appendices slides for more details.

ACT

1. Conclusions:

a. Casual riders tend to ride longer and have extended sessions in the summer season.
b. On average non-member clients ride longer than members.
c. Weekends are preferred by casual riders.
d. Electric bikes are picked more often by casual riders.

2. Recommendations:

From the analysis it can be inferred that casual riders differ in many ways from member riders. That being said the top three recommendations are:

a. Seek for weekend member incentives such as; discounts, free passes, and alliances.
b. Free or discounted electric bike rides since casual riders prefer them.
c. Adjust the business goal in order to create season passes and maximize summer and fall clients.

The marketing analytics team should focus on these insights related to the business task, and find the way to drive the correct marketing strategy in order to maximize the memberships.

3. Expand findings:

a. The findings could be expanded searching for a correlation between tourist increase in summer season.
b. Age and sex of users could have been a good option for an even more targeted marketing strategy.