**Cyclistic Data Analysis Project**

**A summary of the content:**

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

**Introduction:**

In this project I will analyze the Cyclistic’s data to get insights and answer questions using the six steps of the data analysis process (Ask, Prepare, Process, Analyze, Share, and Act) using R for processing and cleaning, SQL for summarizing and analyzing, Tableau for visualizing the data, and PowerPoint for presenting the insights to answer the business task.

1. **Ask:** A clear statement of the business task.
   * We need to get insights to answer: **How do annual members and casual riders use Cyclistic bikes differently?**
   * But first we need to know:
     1. What is the weekly average ride length for casual and members riders?
     2. Is there a correlation between the seasons in Chicago and the number of rides?
     3. Is there a correlation between the seasons in Chicago and ride length?
     4. What is the average number of rides on each day of the week?
     5. What is the average ride length on each day of the week?
     6. Which stations do casual riders often visit?
     7. What is the most used bike type for casual riders?
2. **Prepare:** A description of all data sources used.
   * I downloaded the previous 12 months data from this site <https://divvy-tripdata.s3.amazonaws.com/index.html> (from Jan-2023 till Dec-2023) after making sure that this source is reliable, original, comprehensive, current, and cited.
3. **Process:** Documentation of any cleaning or manipulation of data.
   * **Collecting the data:**
     1. First downloading the data (12 separate csv files as file for each month).
     2. Unzip the files.
     3. Create a folder on my desktop to house the files. Use proper file-naming conventions.
     4. Create subfolders for the .csv file so that I have a copy of the original data. Move the downloaded files to the proper subfolder.
     5. Open RStudio to merge all the 12 csv files together in one data frame as in the below code chunk: *(I am using R language as the data is around 5 million records and Excel won’t handle it properly)*

```{R}

library(tidyverse)

library(data.table)

df <- list.files(path = 'D:/Bike-share data/data/Bike-Share/', pattern = '\*.csv') %>%

map\_df(~fread(.))

head(df)

```

I used the head() function to see if the data collected properly.

* + - * Alternatively, we can use Python to collect the 12 csv files into one data frame as below:

import pandas as pd

import glob

# Get a list of all CSV files in a directory

csv\_files = glob.glob('D:/Bike-share data/data/Bike-Share/\*.csv')

# Create an empty dataframe to store the combined data

combined\_df = pd.DataFrame()

# Loop through each CSV file and append its contents to the combined dataframe

for csv\_file in csv\_files:

df = pd.read\_csv(csv\_file)

combined\_df = pd.concat([combined\_df, df])

print(combined\_df)

* **Manipulate the data for better understanding:**

1. Use the colnames() function to see the column names.

```{R}

colnames(df)

```

1. Create a new variable called ride\_length. Calculate the length of each ride by subtracting the column started\_at from the column ended\_at and divide it by 60 to convert to minutes, as in the below code chunk:

```{R}

df <- mutate(df, ride\_length=(ended\_at-started\_at)/60)

```

1. Create a column called day\_of\_week, and calculate the day of the week that each ride started using the wday function, noting that 1 = Sunday and 7 = Saturday.

```{R}

df <- mutate(df, day\_of\_week=wday(started\_at))

```

1. Converting the blank values to NA.

```{R}

df <- df %>% mutate\_at(c('ride\_id','rideable\_type','start\_station\_name','start\_station\_id',"end\_station\_name","end\_station\_id", "member\_casual"), ~na\_if(., ''))

```

1. Creating another dataframe (df\_cleaned) without the unwanted columns start\_lat, start\_lng, end\_lat, end\_lng, start\_station\_id, end\_station\_id because these columns won’t be needed for this analysis, and renaming rideable\_type to bike\_type and member\_casual to customer\_type.

```{R}

df\_cleaned <- df %>%

select("ride\_id", "rideable\_type", "started\_at", "ended\_at", "start\_station\_name", "end\_station\_name", "member\_casual", "ride\_length", "day\_of\_week") %>%

rename(bike\_type=rideable\_type, customer\_type=member\_casual)

```

* + Now we will export the data from R to MySQL database as it processes the data more quickly than RStudio.
    1. Install the RMySQL and DBI packages and load them.

```{R}

install.packages('RMySQL')

install.packages('DBI')

library(RMySQL)

library(DBI)

```

* + 1. Open MySQL and create an empty table on MySQL database with the same column names as following.

CREATE TABLE bike\_share (

ride\_id VARCHAR (45) NOT NULL,

bike\_type VARCHAR(45),

started\_at DATETIME,

ended\_at DATETIME,

start\_station\_name VARCHAR(45),

end\_station\_name VARCHAR(45),

customer\_type VARCHAR(45),

ride\_length FLOAT,

day\_of\_week INT,

PRIMARY KEY (ride\_id));

* + 1. On R run the below code to connect to the MySQL database that is hosted on the local machine.

```{R}

con <- dbConnect(MySQL(), user="user", password="pass",

dbname="database", host="localhost”)

```

* + 1. Then run the below code to view our tables that are in the database.

```{R}

dbListTables(con)

```

* + 1. Use the dbWriteTable() to export the R dataframe (df) to MySQL, then disconnect from the server.

dbWriteTable(con, name= "bike\_share", value= df, append= TRUE, temporary= FALSE, row.names = FALSE)

dbDisconnect(con)

* + 1. To make sure that the data is imported successfully to MySQL database we will run the below query.

SELECT

COUNT(\*)

FROM

bike\_share;

mysql> 5719877 row(s) returned.

The count result for all the records is 5,719,877 which is the same as in the R dataframe.

* Now it is time to run some calculations on the data to make sure it is clean and reliable.

1. Count the ride\_length records where equal or below 0 minutes and drop those records.

SELECT

COUNT(\*)

FROM

bike\_share

WHERE

ride\_length <= 0;

mysql> 528row(s) returned.

DELETE FROM bike\_share

WHERE

ride\_length <= 0;

mysql> 528row(s) affected.

1. There are records with the same station name in start\_station\_name and end\_station\_name, with ride\_length less than 1 minute, which means that the customer might have changed his mind and docked the bike back to the station.

So we need to delete those records.

SELECT

COUNT(\*)

FROM

bike\_share

WHERE

ride\_length < 1

AND start\_station\_name = end\_station\_name;

mysql> COUNT(\*)

83501

DELETE FROM bike\_share

WHERE

ride\_length < 1 AND start\_station\_name = end\_station\_name;

mysql> 83501 row(s) affected.

1. There are also records with null start\_station\_name and end\_station\_name and ride\_length less than 1 minute, so we will drop it.

SELECT

COUNT(\*)

FROM

bike\_share

WHERE

start\_station\_name IS NULL

AND end\_station\_name IS NULL

AND ride\_length < 1;

mysql> COUNT(\*)

41144

DELETE FROM bike\_share

WHERE

start\_station\_name IS NULL

AND end\_station\_name IS NULL

AND ride\_length < 1;

mysql> 41144 row(s) affected.

1. Remove (Temp) from start\_station\_name and end\_station\_name to make it more consistent.

UPDATE bike\_share

SET

end\_station\_name = REPLACE(end\_station\_name, ' (Temp)', ''),

start\_station\_name = REPLACE(start\_station\_name, ' (Temp)', '')

WHERE

end\_station\_name LIKE '% (Temp)'

OR start\_station\_name LIKE '% (Temp)';

mysql> 112695 row(s) affected Rows matched: 112695 Changed: 112695 Warnings: 0

1. Search for outliers in the ride\_length to avoid skewing our analysis.

We can find the outliers by calculating the interquartile range (IQR) as below:

SELECT ride\_length\_quartile, MAX(ride\_length) AS quartile\_break

FROM( SELECT ride\_id, ride\_length, NTILE(4) OVER (ORDER BY ride\_length) AS ride\_length\_quartile

FROM bike\_share) AS quartiles

WHERE ride\_length\_quartile IN (1, 3)

GROUP BY ride\_length\_quartile;

mysql>

ride\_length\_quartile quartile\_break

1 5.7

3 17.2167

IQR = Q3 – Q1 = 17.2167 – 5.7

IQR = 11.5167

High outlier ≥ Q3 + (1.5 x IQR)

Low outlier ≤ Q1 − (1.5 x IQR)

17.2167 + (1.5 x 11.5167) = 34.49175

5.7 - (1.5 x 11.5167) = -11.57505

DELETE FROM bike\_share

WHERE

ride\_length < - 11.57505

OR ride\_length > 34.49175;

mysql> 412675 row(s) affected.

1. Calculate the minimum and maximum ride\_length. to make sure there are no outliers.

SELECT

MIN(ride\_length) AS min\_ride\_length,

MAX(ride\_length) AS max\_ride\_length

FROM

bike\_share;

mysql> min\_ride\_length max\_ride\_length

0.15 34.4833

1. Add new column Season and season\_num.

ALTER TABLE bike\_share

ADD COLUMN Season VARCHAR(45)

ADD COLUMN season\_num INT;

UPDATE bike\_share

SET

season = CASE

WHEN month\_number IN (12 , 1, 2) THEN 'Winter'

WHEN month\_number IN (3 , 4, 5) THEN 'Spring'

WHEN month\_number IN (6 , 7, 8) THEN 'Summer'

WHEN month\_number IN (9 , 10, 11) THEN 'Fall'

END;

mysql> 5160876 row(s) affected Rows matched: 5160876 Changed: 5160876 Warnings: 0

UPDATE bike\_share

SET

season\_num = CASE

WHEN season = 'Winter' THEN 1

WHEN season = 'Spring' THEN 2

WHEN season = 'Summer' THEN 3

WHEN season = 'Fall' THEN 4

END;

mysql> 5160876 row(s) affected Rows matched: 5160876 Changed: 5160876 Warnings: 0

1. **Analyze:** A summary of the analysis.

* Now it is time for some exploration.
* **What is the weekly average ride length for casual and members riders?**
  + 1. Let’s first calculate the mode for each customer\_type.

CREATE TEMPORARY TABLE temp

SELECT customer\_type, day\_of\_week, COUNT(\*) AS days

FROM bike\_share

GROUP BY customer\_type, day\_of\_week;

SELECT customer\_type, day\_of\_week

FROM (SELECT day\_of\_week, COUNT(\*) AS count\_days

FROM bike\_share

GROUP BY customer\_type, day\_of\_week) t1,

(SELECT customer\_type, MAX(days) AS common\_day

FROM temp

GROUP BY customer\_type) t2

WHERE count\_days=common\_day;

mysql> 14 row(s) affected Records: 14 Duplicates: 0 Warnings: 0

mysql> customer\_type day\_of\_week

casual 7

member 5

Now we know that the causal customers usually go for a ride on Sundays, while the member customers on Thursdays.

* + 1. Calculate the average ride\_length and group by customer\_type.

SELECT

customer\_type,

AVG(ride\_length) AS avg\_ride\_length

FROM

bike\_share

GROUP BY customer\_type;

mysql> customer\_type avg\_ride\_length

casual 12.48

member 10.37

* + 1. Calculate the weekly average ride\_length per customer\_type.

But first we will add week\_number and month\_number as we will need them in further analysis.

ALTER TABLE bike\_share

ADD COLUMN week\_number INT,

ADD COLUMN month\_number INT;

UPDATE bike\_share

SET

week\_number = WEEK(started\_at),

month\_number = month(started\_at);

mysql> 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0

mysql> 5160876 row(s) affected Rows matched: 5160876 Changed: 5160876 Warnings: 0

SELECT

t2.customer\_type,

AVG(avg\_ride\_length) AS avg\_weekly\_ride\_length

FROM

(SELECT

customer\_type,

week\_number,

AVG(ride\_length) AS avg\_ride\_length

FROM

bike\_share

GROUP BY customer\_type, week\_number

ORDER BY week\_number) t2

GROUP BY t2.customer\_type

ORDER BY avg\_weekly\_ride\_length;

mysql> customer\_type avg\_weekly\_ride\_length

casual 11.625

member 10.046

The result shows that the **average annual and weekly ride length for Causal rides is higher than Member rides.**

* **Is there a correlation between the seasons in Chicago and the number of rides?**
  1. First let’s create a temporary table with the daily rides count and seasons number.

CREATE TEMPORARY TABLE daily\_rides\_count

SELECT

Season, season\_num, DATE(started\_at) days, COUNT(ride\_id) AS ride\_count

FROM

bike\_share

GROUP BY season, season\_num, days

ORDER BY days;

mysql> 365 row(s) affected Records: 365 Duplicates: 0 Warnings: 0

* 1. Now we can calculate the correlation between the seasons and the number of rides.

SELECT

(AVG(season\_num \* ride\_count) - AVG(season\_num) \* AVG(ride\_count)) /

(SQRT(AVG(season\_num \* season\_num) - AVG(season\_num) \* AVG(season\_num)) \*

SQRT(AVG(ride\_count \* ride\_count) - AVG(ride\_count) \* AVG(ride\_count))) AS correlation

FROM

daily\_rides\_count;

mysql> 0.588

**The result is closer to +1 which means there is a strong correlation between the seasons and the number of rides.**

* **Is there a correlation between the seasons in Chicago and ride length?**
  1. Let’s create a temporary table with the daily rides average and seasons number.

CREATE TEMPORARY TABLE avg\_daily\_rides

SELECT

Season, season\_num, DATE(started\_at) days, AVG(ride\_length) AS avg\_ride\_length

FROM

bike\_share

GROUP BY season, season\_num, days

ORDER BY days;

mysql> 365 row(s) affected Records: 365 Duplicates: 0 Warnings: 0

* 1. Now we can calculate the correlation between the seasons and the daily average ride length.

SELECT

(AVG(season\_num \* avg\_ride\_length) - AVG(season\_num) \* AVG(avg\_ride\_length)) /

(SQRT(AVG(season\_num \* season\_num) - AVG(season\_num) \* AVG(season\_num)) \*

SQRT(AVG(avg\_ride\_length \* avg\_ride\_length) - AVG(avg\_ride\_length) \* AVG(avg\_ride\_length))) AS correlation

FROM

avg\_daily\_rides;

mysql> 0.423

**The above score is moderate correlation which means there is a correlation between the seasons and the daily average ride length but not strong.**

* **What is the average number of rides on each day of the week?**
  1. Calculate the average number of rides.

SELECT

DATE\_FORMAT(daily, '%W') weekday,

t1.day\_of\_week,

ROUND(AVG(ride\_num), 2) avg\_ride\_num

FROM

bike\_share t1,

(SELECT

DATE(started\_at) daily, COUNT(ride\_id) ride\_num, day\_of\_week

FROM

bike\_share

GROUP BY daily , day\_of\_week

ORDER BY daily) t2

WHERE

t1.day\_of\_week = t2.day\_of\_week

GROUP BY weekday , t1.day\_of\_week

ORDER BY t1.day\_of\_week;

mysql>  **weekday day\_of\_week avg\_ride\_num**

Sunday 1 12147.92

Monday 2 12754.48

Tuesday 3 14564.71

Wednesday 4 14878.54

Thursday 5 15244.81

Friday 6 14673.48

Saturday 7 14750.06

* **What is the average ride length on each day of the week?**
  1. Calculate the average ride length.

SELECT

DATE\_FORMAT(started\_at, '%W') weekday,

day\_of\_week,

ROUND(AVG(ride\_length), 2) avg\_ride\_length

FROM

bike\_share

GROUP BY weekday, day\_of\_week

ORDER BY day\_of\_week;

mysql> **weekday day\_of\_week avg\_ride\_length**

Sunday 1 12.02

Monday 2 10.61

Tuesday 3 10.63

Wednesday 4 10.56

Thursday 5 10.68

Friday 6 10.99

Saturday 7 12.17

* **Which stations do casual riders often visit?**
  1. Calculate top 10 started from stations.

SELECT start\_station\_name, COUNT(ride\_id) rides\_count

FROM bike\_share

WHERE start\_station\_name IS NOT NULL AND customer\_type = 'casual'

GROUP BY start\_station\_name

ORDER BY rides\_count DESC

LIMIT 10;

mysql> **start\_station\_name rides\_count**

Streeter Dr & Grand Ave 30230

DuSable Lake Shore Dr & Monroe St 19468

DuSable Lake Shore Dr & North Blvd 15955

Michigan Ave & Oak St 14588

Shedd Aquarium 13154

Millennium Park 12641

Theater on the Lake 12298

Wells St & Concord Ln 11056

Dusable Harbor 10253

Clark St & Elm St 9715

* 1. Calculate top 10 visited stations.

SELECT end\_station\_name, COUNT(ride\_id) rides\_count

FROM bike\_share

WHERE end\_station\_name IS NOT NULL AND customer\_type = 'casual'

GROUP BY end\_station\_name

ORDER BY rides\_count DESC

LIMIT 10;

mysql> **end\_station\_name rides\_count**

Streeter Dr & Grand Ave 31244

DuSable Lake Shore Dr & North Blvd 17808

DuSable Lake Shore Dr & Monroe St 16690

Michigan Ave & Oak St 15147

Millennium Park 14747

Theater on the Lake 12254

Shedd Aquarium 11246

Wells St & Concord Ln 10954

Clark St & Lincoln Ave 9628

Clark St & Armitage Ave 9535

* + **What is the most used bike\_type for casual riders?**
    1. Group by the bike\_type with the count of rides.

SELECT bike\_type, COUNT(ride\_id) rides\_count

FROM bike\_share

WHERE customer\_type = 'casual'

GROUP BY bike\_type

ORDER BY rides\_count DESC;

mysql> **bike\_type rides\_count**

electric\_bike 983407

classic\_bike 702702

docked\_bike 45048

1. **Share:** Supporting visualizations and key findings.
   * **Now with the analysis we have made so far, we can create visuals using Tableau Public to help us to find the trends and seasonality.** 
     1. To move the data to Tableau Public we need to move it to RStudio and save it as .RData file.

```{R}

con <- dbConnect(MySQL(), user="user", password="pass",

dbname="database", host="localhost")

df\_cleaned <- dbReadTable(con, name= "bike\_share", row.names = FALSE)

dbDisconnect(con)

save(df\_cleaned, file = 'bike\_share.RData')

```

* + 1. Then open Tableau Public and chose **Statistical File**, then chose **bike\_share.RData**.
  + Create a Dashboard with the analysis we have made in the earlier steps to answer each question.
  + Link to the created Tableau Story:

<https://public.tableau.com/app/profile/ahmed.nasser.viz/viz/BikeShare_17087406050140/CyclisticStory>

1. **Act:** The top three recommendations based on my analysis.
   * **Conclusion:**
2. Casual riders usually go for a ride on Sundays, while member riders go on Thursdays.
3. We found that casual riders tend to ride longer distances on average than member riders. Thus, introducing different packages would likely attract casual riders to subscribe.
4. There is a strong correlation between the number of rides and the seasons, with a noticeable increase starting from the end of the spring till the beginning of the fall, meaning that riders prefer hot weather.
5. Ride numbers increase on weekdays, but ride durations increase on weekends, indicating that casual riders spend more time riding on weekends when they have free time to spare.
6. Upon conducting an online search for the most visited stations, it was found that they are all located in or near parks, meaning they take the rides to visit the parks.
7. Electric bikes are the most used by both types of riders. They are equipped with a thumb throttle and pedal assist, which makes it easier to travel further and tackle steep hills.
   * **Recommendations:**
     1. Launch a weekly and monthly package for riders who have a high average ride duration.
     2. Seasonal offer with a discount for new members that starts from the end of spring until summer.
     3. Create a special discount at weekends for members whose ride length is longer than 25 minutes.
     4. Design merchandise such as shirts and water bottles with the company logo in the top visited parks.
     5. Discount on parks tickets for members who arrive with Cyclistic bikes.
     6. Increase the number of electric bikes in the steep hills' areas.