How I did it?

Initially, I reviewed the data on Excel, but I opted to use SQL on BigQuery to analyse it due to the substantial number of rows, which was approximately 400,000. To organise the files, I renamed them according to the year and month format (e.g., 2023-01, 2023-02), and subsequently merged them into a single table named "Cyclistic" using the UNION operator.

```
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 `casestudy1-cyclistic-380506.DataSet.2023-01`
UNION ALL
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 `casestudy1-cyclistic-380506.DataSet.2023-02`
```

After executing this query, I selected the option to save the results and created a table in BigQuery with the name "Cyclistic" to store the data.

I then began cleaning the data.
Firstly, I checked for the duplicate rows in the data set.

```
SELECT ride_id, COUNT(*) as count
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic`
GROUP BY ride_id
HAVING COUNT(*) > 1
```

I found that 1 ride_id is repeated. I checked that ride_id and found these are not duplicate entries, they are different rides taken that have been given the same ride id for whatever reason. I decided to leave them in as they will not affect the analysis.

Row	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_id	end_station_n	end_station_id	start_lat
1	7.41E+21	electric_bike	2023-02-08 1...	2023-02-08 ...	*null*	*null*	Damen A...	KA17018054	4
2	7.41E+21	classic_bike	2023-01-20 1...	2023-01-20 ...	Franklin St & A...	TA1309000...	Clinton S...	13021	41.879434

After this, I will add 2 new columns "ride_length_minutes" and "day_of_week" and include only valid columns.The "ride_length_minutes" column calculates the length of each ride in minutes.The "day_of_week" column indicates the day of the week the ride took place, with 1 representing Sunday and 7 representing Saturday.

```
SELECT ride_id, rideable_type,
        started_at,
        ended_at,
        ROUND(TIMESTAMP_DIFF(ended_at, started_at, second)/60, 1) AS ride_length_minutes,
        EXTRACT(DAYOFWEEK FROM started_at) AS day_of_week,
        start_station_name,
        start_station_id,
        end_station_name,
        end_station_id,
        member_casual
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic`
```

Then saved this result as a new table titled "Cyclistic-N".

Let's Analyse the data now

Ride Length

Finding the average, minimum and maximum ride lengths for all rides.

```
SELECT AVG(ride_length_minutes) AS avg_ride_len,
        MIN(ride_length_minutes) AS min_ride_len,
        MAX(ride_length_minutes) AS max_ride_len
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
```

Row	avg_ride_len	min_ride_len	max_ride_len
1	13.26867701827485	-3.3	33603.7

It is observed that we have a minimum ride length as negative, which doesn't make sense. I will filter out the negative ride length as that is invalid.

```
SELECT AVG(ride_length_minutes) AS avg_ride_len,
        MIN(ride_length_minutes) AS min_ride_len,
        MAX(ride_length_minutes) AS max_ride_len
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE ride_length_minutes > 0
```

Row	avg_ride_len	min_ride_len	max_ride_len
1	13.2824654...	0.1	33603.7

Now, Let's calculate average, minimum and maximum ride lengths for casual riders and member riders.

```
SELECT member_casual,
        AVG(ride_length_minutes) AS avg_ride_len,
        MIN(ride_length_minutes) AS min_ride_len,
        MAX(ride_length_minutes) AS max_ride_len
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE ride_length_minutes > 0
GROUP BY member_casual
```

Row	avg_ride_len	min_ride_len	max_ride_len
1	23.0742199...	0.1	33603.7

Finding out the number of rides taken by members and casuals on each day of the week.

```
SELECT day_of_week,
        COUNT(ride_id) AS rides_by_member
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE member_casual= 'memberl'
        AND ride_length_minutes > 0
GROUP BY day_of_week
ORDER BY day_of_week
```

```
SELECT day_of_week,
        COUNT(ride_id) AS rides_by_casual
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE member_casual= 'casual'
        AND ride_length_minutes > 0
GROUP BY day_of_week
ORDER BY day_of_week
```

Let’s calculate the average ride length for each day of the week, for both members and causal riders.

```
SELECT day_of_week,
	AVG(ride_length_minutes) AS member_avg_ride_length
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE member_casual= 'member'
	AND ride_length_minutes > 0

GROUP BY day_of_week
ORDER BY day_of_week
```

```
SELECT day_of_week,
	AVG(ride_length_minutes) AS casual_avg_ride_length
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE member_casual= 'casual'
	AND ride_length_minutes > 0

GROUP BY day_of_week
ORDER BY day_of_week
```

Member VS Casual

First calculate: Of all the rides taken, how many were by members and how many were by casual riders?

```
Select COUNT(*) as rides_taken
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE ride_length_minutes > 0
Group by member_casual
```

Checking to see if there are differences in the type of vehicles chosen

```
SELECT rideable_type,
	COUNT(*) AS rides
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE ride_length_minutes > 0
GROUP BY rideable_type
```

Let's find out which stations are most frequented by riders.

```
SELECT start_station_name,
	COUNT(*) AS total_rides
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE
	ride_length_minutes > 0
	AND start_station_name != 'null'
GROUP BY start_station_name
ORDER BY COUNT(*) DESC LIMIT 10
```

```
SELECT end_station_name,
	COUNT(*) AS total_rides
FROM `casestudy1-cyclistic-380506.DataSet.Cyclistic-N`
WHERE
	ride_length_minutes > 0
	AND end_station_name != 'null'
GROUP BY end_station_name
ORDER BY COUNT(*) DESC LIMIT 10
```