```sql
SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_03

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_04

UNION DISTINCT

SELECT *
FROM  my-data-project-8075.Cyclistic_trip_data.trip_data_2021_05

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_06

UNION DISTINCT

SELECT*
FROM  my-data-project-8075.Cyclistic_trip_data.trip_data_2021_07

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_08

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_09

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_10

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_11

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2021_12

UNION DISTINCT

SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.trip_data_2022_01

UNION DISTINCT

SELECT *
FROM  my-data-project-8075.Cyclistic_trip_data.trip_data_2022_02
```

To combined all previous 12 months of Cyclistic trip data from March 2021 to February 2022 using UNION DISTINCT to exclude duplicates. Total rows in the combined table are 5667986.

Then, I moved on to renaming the rideable_type, started_at, ended_at, and member_casual columns into new column names as shown below. Since BigQuery does not support combining ALTER() TABLE() with RENAME() COLUMN(), I had to rename and arrange the columns manually. After obtaining the results, the old combined table, "combined_trip_data" was deleted and the new table was saved in BigQuery using the same name.

```
SELECT ride_id,
rideable_type AS bike_type,
started_at AS start_date_time,
ended_at AS end_date_time,
start_station_name,
start_station_id,
end_station_name,
end_station_id,
start_lat,
start_lng,
end_lat,
end_lng,
member_casual AS rider_type

 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
```

Checking for row that have start_date_time same as end_date_time and start_date_time more than end_date_time

1) start_date_time = end_date_time

```
SELECT *
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
 WHERE start_date_time = end_date_time
```

```
There are 49562 row that have start_date_time equal to end_date_time. Afterwards I deleted
these rows using the following query:
```

```
DELETE
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
WHERE start_date_time = end_date_time
```

This statement removed 49,562 rows from combined_trip_data. To check for rows where star_date_time = end_date_time, I ran this query,

```
SELECT *
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
WHERE start_date_time = end_date_time
```

There is no data to display.

Moving on….

2) Checking for row with start_date_time > end_date_time

```
SELECT *
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
 WHERE start_date_time > end_date_time
```

There are 89 rows where the start_date_time is more than end_date_time. I assume this must be an error in data entry so rather than deleting these 89 rows, I decided to swap values between columns using UPDATE() ,SET() and WHERE () clauses for only affected rows.

```
UPDATE`my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
SET start_date_time = end_date_time, end_date_time = start_date_time
WHERE start_date_time > end_date_time
```

This statement modified 89 rows in combined_trip_data.

To confirm if any rows have start_date_time is more than end date time, I ran this query

```
SELECT*
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
WHERE start_date_time > end_date_time
```

There is no data to display

Checking if any rows have null start_date_time and end_date_time

```
SELECT *
FROM my-data-project-8075.Cyclistic_trip_data.combined_trip_data
WHERE start_date_time IS NULL OR end_date_time IS NULL
```

There is no data to display

```
SELECT *
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
 WHERE ride_id IS NULL OR bike_type IS NULL OR rider_type IS NULL
```

There is no data to display

```
SELECT DISTINCT start_station_name
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
```

Searched found Base - 2132 W Hubbard Warehouse which is the Cyclistic warehouse.

```sql
SELECT distinct end_station_name FROM `my-data-project-
8075.Cyclistic_trip_data.combined_trip_data`
```

Searched found Divvy Cassette Repair Mobile Station

```sql
SELECT *
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
WHERE start_station_name = end_station_name AND rider_type = "casual"
```

```sql
SELECT *
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
WHERE start_station_name = end_station_name AND rider_type = "member"
```

To explore further, I have to create ride_length and day_of_week columns (be sure to ORDER BY start_date_time)

```sql
SELECT *,
EXTRACT(DATE FROM start_date_time) AS start_date,
EXTRACT(DAYOFWEEK FROM start_date_time) AS day_of_week,
EXTRACT(MONTH FROM start_date_time) AS mnth_of_year,
EXTRACT(YEAR FROM start_date_time) AS year
(end_date_time - start_date_time) AS ride_length
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data`
 ORDER BY start_date_time
```

```sql
SELECT *,
EXTRACT(HOUR FROM ride_length) AS hour,
EXTRACT(MINUTE FROM ride_length) AS minute,
EXTRACT(SECOND FROM ride_length) AS second,

 FROM  my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver5
```

```sql
DELETE
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver5`
 WHERE start_station_name = 'Base - 2132 W Hubbard Warehouse'
```

```
Removed 356 rows where start_station is Cyclistic warehouse.
```

```
DELETE
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver5`
WHERE start_station_name = 'DIVVY CASSETTE REPAIR MOBILE STATION' OR end_station_name ='DIV
VY CASSETTE REPAIR MOBILE STATION'
```

Removed 8 rows with Divvy repair stations for both start and end stations

```
SELECT *
FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver6`
WHERE ride_id IS NULL OR
bike_type IS NULL OR start_date_time IS NULL OR end_date_time 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 rider_type IS NULL OR
day_of_week IS NULL OR mnth_of_year IS NULL OR ride_length IS NULL OR year IS NULL OR start
_date IS NULL OR hour IS NULL OR minute IS NULL OR second IS NULL
```

Searched found 1020325 rows containing null values especially for end_station_name and end_station_id columns. Since there is no available information to fill these null values, these rows need to be deleted. **The bike-sharing system has issues with difficulty for bikes to fully dock at stations rendering them to be stolen easily. This issue may also explain why significant number of rows have very ride lengths beyond 24 hours.**

Next, I removed rows containing 24 or more ride hours. Bikes that are not docked after 24 hours are considered stolen which will incur penalty charges for users.

```
DELETE FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver6`
WHERE hour >= 24
```

```
SELECT*,
 TIME(hour,minute,second)
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver6`
```

For data granularity, I created three columns using values from gh

```
SELECT *,

(hour + (minute/60) + (second/3600)) AS tduration_hour,      #tduration means trip duration
((hour*60) + minute + (second/60)) AS tduration_minute,
((hour*3600) + (minute*60) + second) AS tduration_second,



  FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver8`
```

```
SELECT *
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver9`
 WHERE tduration_minute <1
```

```
DELETE
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver9`
 WHERE tduration_minute <1
```

This statement removed 3,580 rows from combined_trip_data_ver9.

Total rows for cleaned_trip_data table is 4592561

Made new table with the following query:

```
SELECT ride_id,
bike_type,
start_date_time,
end_date_time,
start_station_name,
start_station_id,
end_station_name,
end_station_id,
start_lat,
start_lng,
end_lat,
end_lng,
rider_type,
day_of_week,
mnth_of_year,
year,
ride_date,
EXTRACT(HOUR FROM start_date_time) AS ride_hour,
EXTRACT(MINUTE FROM start_date_time) AS ride_minute,
tduration_hour AS ride_length_hour,
tduration_minute AS ride_length_minute,
tduration_second AS ride_length_second,
ride_length
 FROM `my-data-project-8075.Cyclistic_trip_data.combined_trip_data_ver9`

ORDER BY start_date_time
```

After creating the table, I decided to add a new column: **week_of_year**.

```
SELECT *,
EXTRACT(WEEK FROM start_date_time) AS week_of_year

 FROM `my-data-project-8075.Cyclistic_trip_data.cleaned_cyclistic_data`
 ORDER BY start_date_time
```

The final table for analysis has been created named **cleaned_cyclistic_trip_data.**
With that done, lets start the analysis…