```
WITH JoinedData AS (
                                                                             SELECT
                                                                  ,ch.rental_id
                                      ,IFNULL(ch.duration_ms, 0) AS duration_ms
                                              ,IFNULL(ch.bike_id, 0) AS bike_id
                      ,IFNULL(ch.end_date, TIMESTAMP('2021-01-01')) AS end_date
                                ,IFNULL(ch.end_station_id, 0) AS end_station_id
                            ,IFNULL(cs_end.name, 'Unknown') AS end_station_name
                  ,IFNULL(ch.start_date, TIMESTAMP('2021-01-01')) AS start_date
                            ,IFNULL(ch.start_station_id, 0) AS start_station_id
                        ,IFNULL(cs_start.name, 'Unknown') AS start_station_name
                              ,IFNULL(cs_start.longitude, 0) AS start_longitude
                                ,IFNULL(cs_start.latitude, 0) AS start_latitude
                                  ,IFNULL(cs_end.longitude, 0) AS end_longitude
                                     IFNULL(cs_end.latitude, 0) AS end_latitude
                    FROM `data-analysis-389112.Project_Google.cycle_hire_new` AS ch
     LEFT JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS cs_start
                                           ON ch.start_station_id = cs_start.id
       LEFT JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS cs_end
                                               ON ch.end_station_id = cs_end.id
                                                                              WHERE
                   ch.start_date >= TIMESTAMP('2021-01-01') AND ch.start_date <
                                                                ('TIMESTAMP('2021-04-01
           AND ch.end_date >= TIMESTAMP('2021-01-01') AND ch.end_date <
                                                        ('TIMESTAMP('2021-04-01
                                                                             ,(

                                                               ) CleanedData AS
                                                                     SELECT
                                                                     ,*
TIMESTAMP_DIFF(end_date, start_date, SECOND) AS rental_duration_seconds
                                                            FROM JoinedData
                                                                              (

                                                                       * SELECT
                                                              ;FROM CleanedData
```

```
WITH q1 AS (
    SELECT *
    FROM `data-analysis-389112.Project_Google.cycle_hire_new`
    WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date <
TIMESTAMP('2021-04-01')
    AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')
)

SELECT
    start_stations.name AS start_station_name,


    AVG(duration) / 60 AS avg_rental_duration_minutes
FROM q1
JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS start_stations
    ON q1.start_station_id = start_stations.id
GROUP BY
    start_station_name

ORDER BY
    Start_station_name desc
```

```
select start_station_name,count (start_station_id) most_returned
`from `data-analysis-389112.Project_Google.cycle_hire_new
group by start_station_name
order by most_returned desc
```

```
WITH q1 AS
( SELECT * FROM `data-analysis-389112.Project_Google.cycle_hire_new` WHERE start_date
>= TIMESTAMP ('2021-01-01') AND start_date < TIMESTAMP ('2021-04-01') and end_date >=
TIMESTAMP ('2021-01-01') AND end_date <TIMESTAMP ('2021-04-01')
)
select format_timestamp('%A', start_date) AS day_of_week,
COUNT (*) as frequency
FROM `data-analysis-389112.Project_Google.cycle_hire_new`
 GROUP BY day_of_week
 ORDER BY frequency DESC
 select extract (DAYOFWEEK from start_date) as day_of_week, AVG (TIMESTAMP_DIFF
(end_date,start_date,second)) / 60 as avg_ride_duration_minutes
  from `data-analysis-389112.Project_Google.cycle_hire_new`
  group by day_of_week
  order by avg_ride_duration_minutes desc
```

```
  select extract (month from start_date ) as month, avg (timestamp_diff
(end_date,start_date,second)) / 60 as avg_ride_duration_minutes
  from `data-analysis-389112.Project_Google.cycle_hire_new`
   group by month
```

```
   select duration/60 as duration, COUNT (*) AS rental_count,
   FROM `data-analysis-389112.Project_Google.cycle_hire_new`
   WHERE start_date >= TIMESTAMP ('2021-01-01') AND start_date < TIMESTAMP
('2021-04-01')
   GROUP BY duration
   ORDER BY rental_count DESC
```

```
SELECT
                                                                ,start_station_name
                              ,FORMAT_TIMESTAMP('%I %p', start_date) AS rental_hour
                                                           COUNT(*) AS rental_count
                                                                                   FROM
                               `data-analysis-389112.Project_Google.cycle_hire_new`
                                                                                  WHERE
                                              ('start_date >= TIMESTAMP('2021-01-01
                                           ('AND start_date < TIMESTAMP('2021-04-01
                                                                               GROUP BY
                                                    start_station_name, rental_hour
```

```
ORDER BY
;Rental_hour
```

```
WITH q1 AS (

    SELECT *

    FROM `data-analysis-389112.Project_Google.cycle_hire_new`

    WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date <
TIMESTAMP('2021-04-01')


 AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')

)


SELECT start_station_id, start_station_name, COUNT(start_station_id) AS
start_station_count, bikes_count, docks_count, nbEmptyDocks

FROM q1

JOIN `data-analysis-389112.Project_Google.cycle_stations_pro`

ON id = start_station_id AND id = end_station_id
```

```sql
WHERE id IN (14, 191, 307, 303, 194, 154)

GROUP BY start_station_id, start_station_name, docks_count, bikes_count, nbEmptyDocks

ORDER BY start_station_count DESC
```

```sql
SELECT
    EXTRACT(MONTH FROM start_date) AS ride_month,
    COUNT(*) AS ride_count
FROM
    `data-analysis-389112.Project_Google.cycle_hire_new`
WHERE
    start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2022-01-01')
GROUP BY
    ride_month
ORDER BY
    ride_month;
```

[Author name]

[Course title]

תוכן

Chapter 1: Facing the Business Topic:

- Objective:
    - As an analyst for Zen City, the goal is to devise strategies that enhance the usefulness of Zen City, expand its user base, and ultimately increase bike rentals for the forthcoming quarter.

· Challenges Identified:

○ Station Popularity Discrepancy: While stations in the Hyde Park area are significantly popular, other stations like Waterloo and King's Cross are underutilized.

○ Riding Patterns: There's an observed trend where the average riding time increases as summer approaches.

○ Station Inconsistencies: Despite its popularity, the Hyde Park area has several stations with minimal activity. Conversely, Belgravia station, which has scant bikes, shows high rental rates.

· Recommendations:

○ Collaborate with Local Businesses: Establish partnerships with businesses in the Hyde Park vicinity. Through the Zen City app, introduce promotions and competitive events. Offer rewards and incentives to riders, making the bike rental experience more enticing.

○ Optimize Station Locations: Reorganize stations within Hyde Park to ensure better access for riders. Consider reducing the number of bikes at certain stations while redirecting some to the Belgravia station to cater to its high demand.

○ Infrastructure and Partnership: Investigate the infrastructure around Waterloo and King's Cross. Given that these are major railway hubs with substantial foot traffic, and considering the non-ideal biking infrastructure, it's pivotal to engage with local authorities and railway management. The aim is to develop a collaborative plan that makes these areas more bike-friendly.

○ Strategic Advertising: Amplify advertising efforts during morning rush hours. By placing ads on billboards and screens in strategic locations, Zen City can position itself as the primary commuting option for those rushing to work. This not only promotes Zen City but also ingrains the idea of biking as a swift and efficient mode of transport.

- In conclusion, by addressing these challenges with the provided recommendations, Zen City can anticipate a notable increase in bike rentals, expanding its user base, and further establishing itself as a key player in eco-friendly urban transportation.

Chapter 2: Data Exploration

overview of the datasets:

- cycle_hire_new:

  - rental_id: Unique identifier for each rental transaction.
  - duration: Duration of the rental in seconds.
  - duration_ms: Duration of the rental in milliseconds.
  - bike_id: Identifier for each bike.
  - bike_model: Model of the bike (some entries are missing).
  - end_date: Date and time when the bike was returned.

○ end_station_id: Identifier for the station where the bike was returned.

○ end_station_name: Name of the station where the bike was returned.

○ start_date: Date and time when the bike was rented.

○ start_station_id: Identifier for the station where the bike was rented from.

○ start_station_name: Name of the station where the bike was rented from.

○ end_station_logical_terminal, start_station_logical_terminal, end_station_priority_id: These columns have missing entries and would need more context or further inspection.

· cycle_stations_pro:

○ id: Unique identifier for each station.

○ installed: Indicates if the station is installed.

○ latitude, longitude: Coordinates of the station.

○ locked: Indicates if the station is locked.

○ name: Name of the station.

○ bikes_count: Number of bikes currently at the station.

○ docks_count: Total number of docks at the station.

○ nbEmptyDocks: Number of empty docks at the station.

○ temporary: Indicates if the station is temporary.

○ terminal_name: Terminal name of the station.

○ install_date: Installation date of the station.

○ removal_date: Removal date of the station (many entries are missing, implying that the stations are still operational).

· Temporal Patterns & Seasonal Trends:

Analyze the frequency of bike rentals over time to identify any patterns. We'll look for trends on a daily, weekly, and monthly basis.

We'll load the data and visualize the frequency of bike rentals over time -

1. We group the data by date and count the rentals to mimic the SQL `GROUP BY` and `COUNT` functions.

2. Identify outliers in rental durations using descriptive statistics.

Starting with the temporal patterns:

The data has been grouped by date, and we have counted the number of rentals for each day, similar to our previous visual analysis.

Next, we'll focus on identifying outliers or anomalies in the rental durations. Using descriptive statistics, we'll examine the distribution of rental durations and identify any unusual values.

1. Compute basic statistics like mean, median, and standard deviation for the `duration' column.

2. Define outliers as durations that fall outside the range of [mean - 3*standard deviation, mean + 3*standard deviation].

Based on our analysis, there are 9 records in the `cycle_hire_new` dataset that can be classified as outliers in terms of rental duration. These outliers are defined as durations that fall outside three standard deviations from the mean.

Here's a sample of some of the outlier records:

- Rental ID : 106334367 has a duration of 13,560 minutes.

- Rental ID : 104894254 has a duration of 7,440 minutes.

- Rental ID : 104849257 has a duration of 9,060 minutes.

- ... and so on.

Handling these outliers will depend on the business context. They might represent genuine long-term rentals or could be errors or anomalies in the data collection process. It would be beneficial to consult with the business team or check the system's data collection logic.

With this, we have mimicked the SQL approach using pandas for:

1. Temporal patterns (grouping by date and counting rentals).

2. Identifying outliers in rental durations.

- Outliers & Anomalies:

  ○ We'll identify any unusual data points in terms of rental duration, distances, or any other relevant metric. Once identified, we'll discuss potential strategies to handle them.

For the topic of Outliers and Anomalies, we previously looked into:

Rental durations to identify outliers.

The overall distribution of rental durations.

Let's craft SQL queries for these:

Rental Durations & Outliers:

To identify outliers in rental durations using the Interquartile Range (IQR) method, we can follow these steps:

a. Calculate Q1, Q3, and IQR.

b. Define outliers as durations outside the range [Q1 - 1.5IQR, Q3 + 1.5IQR].

Zen’s City Project:

Before we start working on the project we want to understand a few things about our data.
The first thing is to understand what kind of data we can get from each table,
In this case we have 2 tables with each table containing different information about our riding stations,

The first one is called “cycle_hire_new”, which include the following columns:

Rental_id

Duration

Duration_ms

Bike_id

Bike_model

Start_date

Start_station_id

Start_station_name

End_date

End_station_id

End_station_name

End_station_logical_terminal

Start_station_logical_terminal

End_station_priority_id

Which basically tells us about every rental there was , the duration of each rental in ms and seconds, the bike id, which model it is, where does it start and where does it end as well.
The second one is called cycle_station_pro which includes the following columns:

Id

Installed

Latitude

Longitude

Locked

Name

Bikes_count

Docks_count

nbEmptyDocks

Temporary

Terminal_name

Install_date

Removal_date

So basically what the second table shows is a more detailed view on every bike station, it includes the id of the station, when it was installed, It also answers where is the Latitude and Longitude of the station, it its locked or not, how many docks are in the station, how many bikes there is to the station, If its Temporary or not, when did it got installed and if and when it was removed.

So first of all we wanted to understand how we can connect this two tables by the primary key so we looked and looked and find out that the id in cycle_station_pro is connected to 2 columns in cycle_hire_new the first one is start_station_id and the second connection is to end_station_id, with this knowledge we know how to connect both tables and we understood that we can try and explore it in the best possible way,

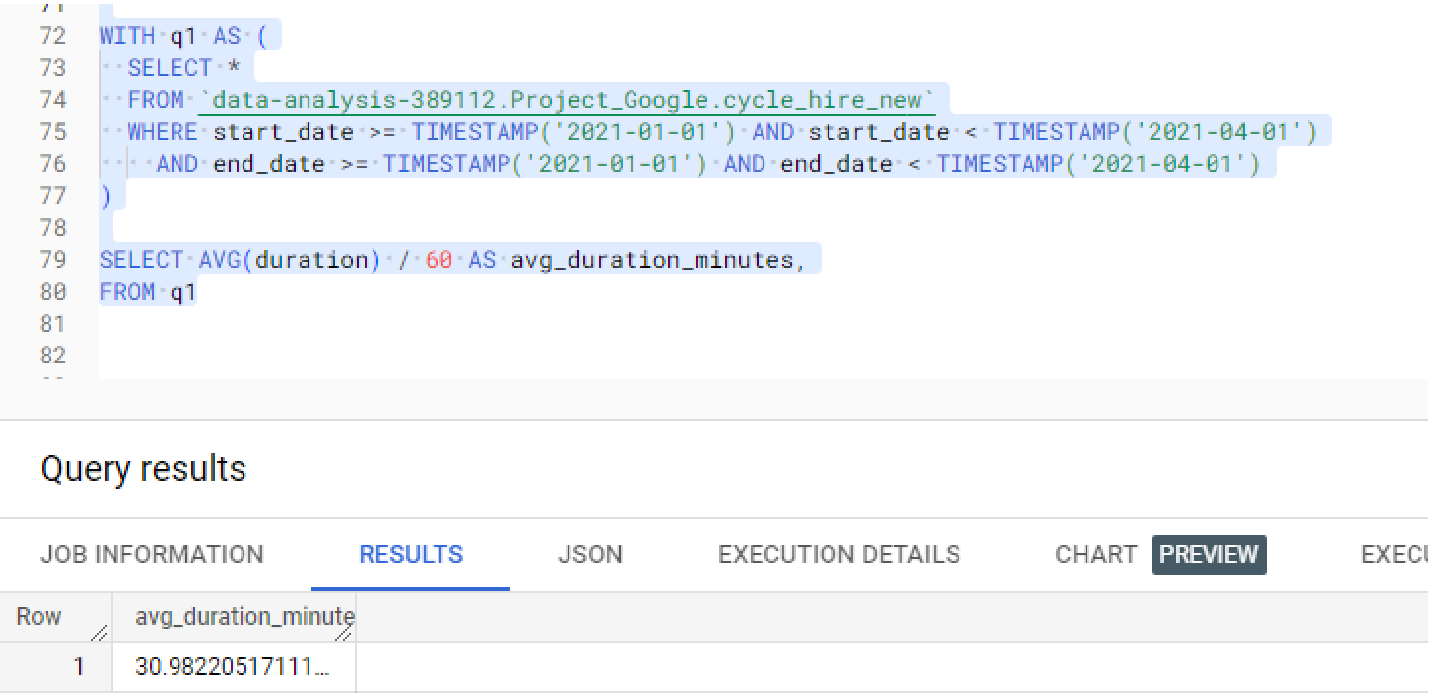
So we first started off by joining the data and cleaning with our code and try and make it the best looking way by taking off nulls and things that are irrelevant:

```
WITH
JoinedData AS (

                                                  SELECT
```

```
                                                             ,ch.rental_id

                                 ,IFNULL(ch.duration_ms, 0) AS duration_ms

                                         ,IFNULL(ch.bike_id, 0) AS bike_id

                 ,IFNULL(ch.end_date, TIMESTAMP('2021-01-01')) AS end_date

                           ,IFNULL(ch.end_station_id, 0) AS end_station_id

                       ,IFNULL(cs_end.name, 'Unknown') AS end_station_name

             ,IFNULL(ch.start_date, TIMESTAMP('2021-01-01')) AS start_date

                       ,IFNULL(ch.start_station_id, 0) AS start_station_id

                   ,IFNULL(cs_start.name, 'Unknown') AS start_station_name

                         ,IFNULL(cs_start.longitude, 0) AS start_longitude

                           ,IFNULL(cs_start.latitude, 0) AS start_latitude

                             ,IFNULL(cs_end.longitude, 0) AS end_longitude

                                IFNULL(cs_end.latitude, 0) AS end_latitude

               FROM `data-analysis-389112.Project_Google.cycle_hire_new` AS ch

LEFT JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS cs_start

                                      ON ch.start_station_id = cs_start.id

  LEFT JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS cs_end

                                          ON ch.end_station_id = cs_end.id

                                                                         WHERE

              ch.start_date >= TIMESTAMP('2021-01-01') AND ch.start_date <
                                                           ('TIMESTAMP('2021-04-01

              AND ch.end_date >= TIMESTAMP('2021-01-01') AND ch.end_date <
                                                           ('TIMESTAMP('2021-04-01

                                                                                 ,(
) CleanedData AS
SELECT
,*
TIMESTAMP_DIFF(end_date, start_date, SECOND) AS rental_duration_seconds
FROM JoinedData
(


* SELECT
;FROM CleanedData
```

After that we know we are on to a great start and we started to ask ourselves some questions, and we started building some questions that can be answered with queries.

The first question was how much time do people rent the bike for in avg:



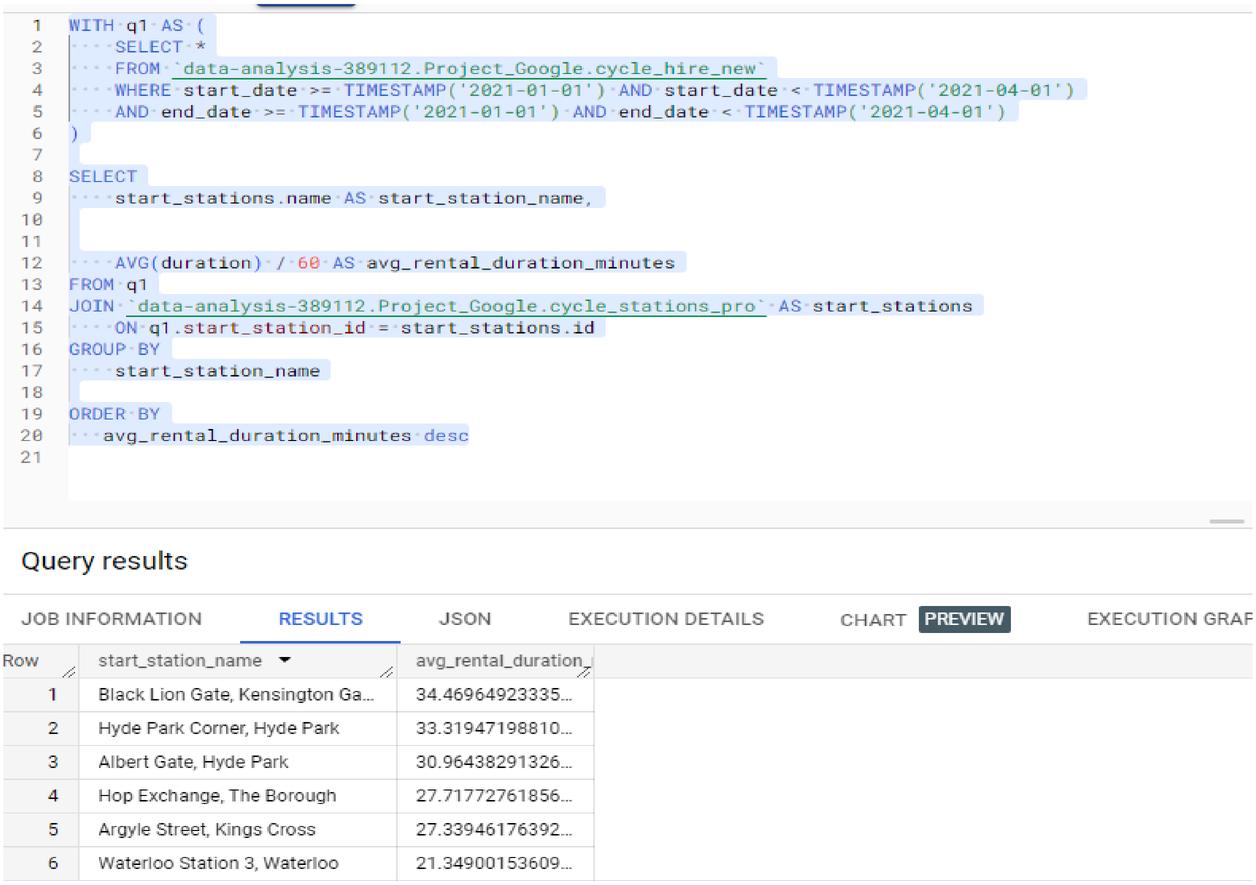
```
WITH q1 AS (
  SELECT *
  FROM `data-analysis-389112.Project_Google.cycle_hire_new`
  WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2021-04-01')
    AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')
)

SELECT AVG(duration) / 60 AS avg_duration_minutes,
FROM q1
```

Query results

Row	avg_duration_minute
1	30.98220517111...

But we didn't stop there, we wanted to know we wanted to know even further about the avg duration time for each station, so we upgraded our query

```
WITH q1 AS (
    SELECT *
    FROM `data-analysis-389112.Project_Google.cycle_hire_new`
    WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date <
TIMESTAMP('2021-04-01')
    AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')
)

SELECT
    start_stations.name AS start_station_name,


    AVG(duration) / 60 AS avg_rental_duration_minutes
FROM q1
JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS start_stations
    ON q1.start_station_id = start_stations.id
GROUP BY
    start_station_name

ORDER BY
    Start_station_name desc
```



```
WITH q1 AS (
   SELECT *
   FROM `data-analysis-389112.Project_Google.cycle_hire_new`
   WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2021-04-01')
   AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')
)

SELECT
   start_stations.name AS start_station_name,


   AVG(duration) / 60 AS avg_rental_duration_minutes
FROM q1
JOIN `data-analysis-389112.Project_Google.cycle_stations_pro` AS start_stations
   ON q1.start_station_id = start_stations.id
GROUP BY
   start_station_name

ORDER BY
   avg_rental_duration_minutes desc

```

Query results

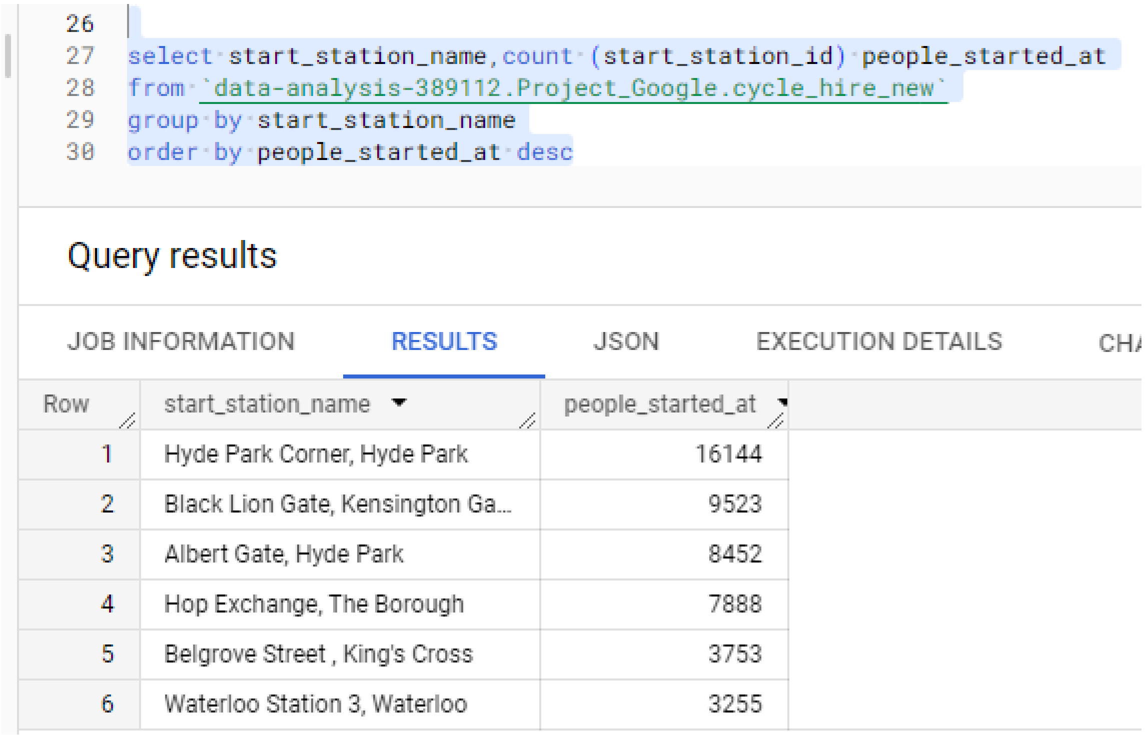
Row	start_station_name	avg_rental_duration_
1	Black Lion Gate, Kensington Ga...	34.46964923335...
2	Hyde Park Corner, Hyde Park	33.31947198810...
3	Albert Gate, Hyde Park	30.96438291326...
4	Hop Exchange, The Borough	27.71772761856...
5	Argyle Street, Kings Cross	27.33946176392...
6	Waterloo Station 3, Waterloo	21.34900153609...

After looking at the differences between every station, we started to think there are huge differences between the stations, as you can see in the image we only have 6 starting stations, 2 of them are above the avg rental duration, with the best one scoring an avg of 34.46 minutes and in the other hand the worst one has an avg of 21.34 minutes which is 38%! Less then the best rentaling station.

After realizing things look different for each station based on the avg rentling duration we wanted to know how much people actually rent a bike from each station and then combining both factors to realising where is our best stations

So we did the following code:

```
select start_station_name,count (start_station_id) most_returned
`from `data-analysis-389112.Project_Google.cycle_hire_new
group by start_station_name
order by people_started_at desc
```



```
select start_station_name,count (start_station_id) people_started_at
from `data-analysis-389112.Project_Google.cycle_hire_new`
group by start_station_name
order by people_started_at desc
```

Query results

Row	start_station_name	people_started_at
1	Hyde Park Corner, Hyde Park	16144
2	Black Lion Gate, Kensington Ga...	9523
3	Albert Gate, Hyde Park	8452
4	Hop Exchange, The Borough	7888
5	Belgrove Street , King's Cross	3753
6	Waterloo Station 3, Waterloo	3255

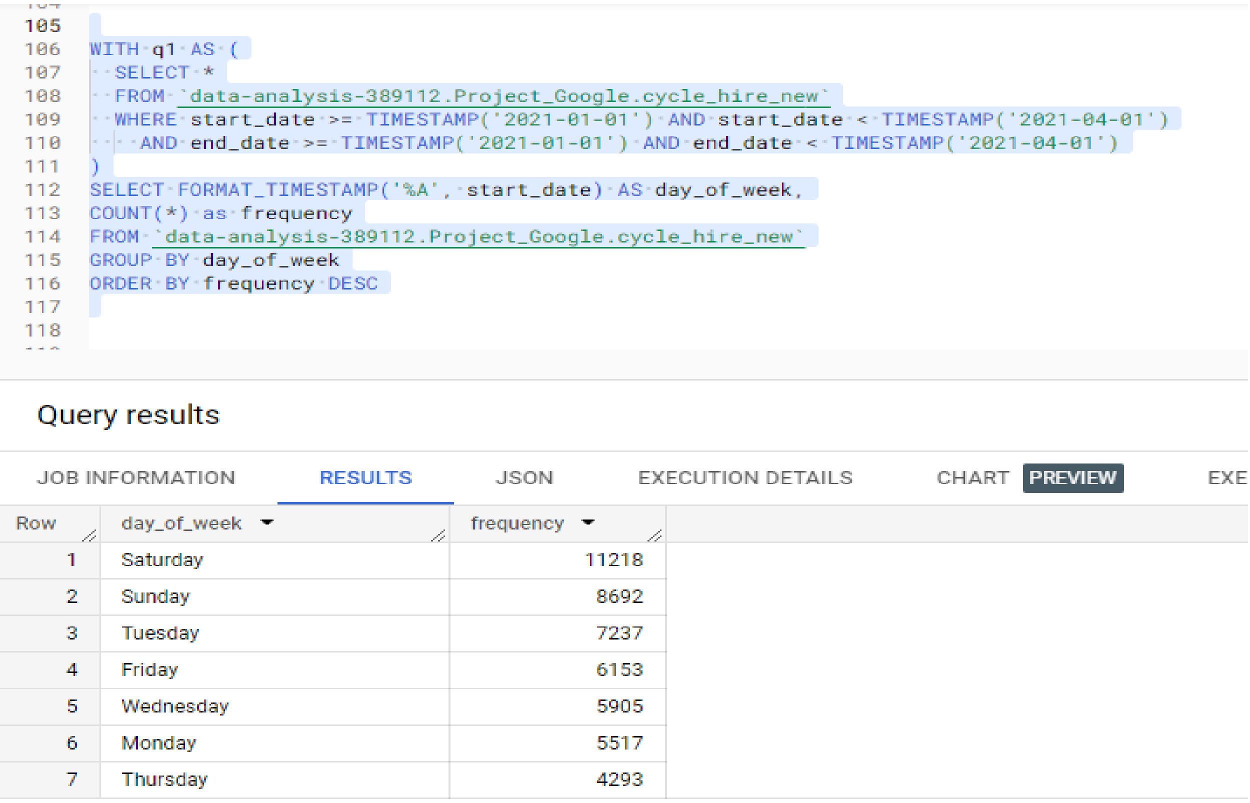
Its a simple code, that tells us how much people rented a bike from each of the 6 stations.

As you can see Black lion Gate and Hyde park are still leading strong in the top, while waterloo and Kings cross are way at the bottom,

After understanding which starting station had a better Q we also wanted to know if there was a specific day which people took the bike

So we did the following query:

```
WITH q1 AS
( SELECT * FROM `data-analysis-389112.Project_Google.cycle_hire_new` WHERE start_date
>= TIMESTAMP ('2021-01-01') AND start_date < TIMESTAMP ('2021-04-01') and end_date >=
TIMESTAMP ('2021-01-01') AND end_date <TIMESTAMP ('2021-04-01')
)
select format_timestamp('%A', start_date) AS day_of_week,
COUNT (*) as frequency
FROM `data-analysis-389112.Project_Google.cycle_hire_new`
 GROUP BY day_of_week
 ORDER BY frequency DESC
```



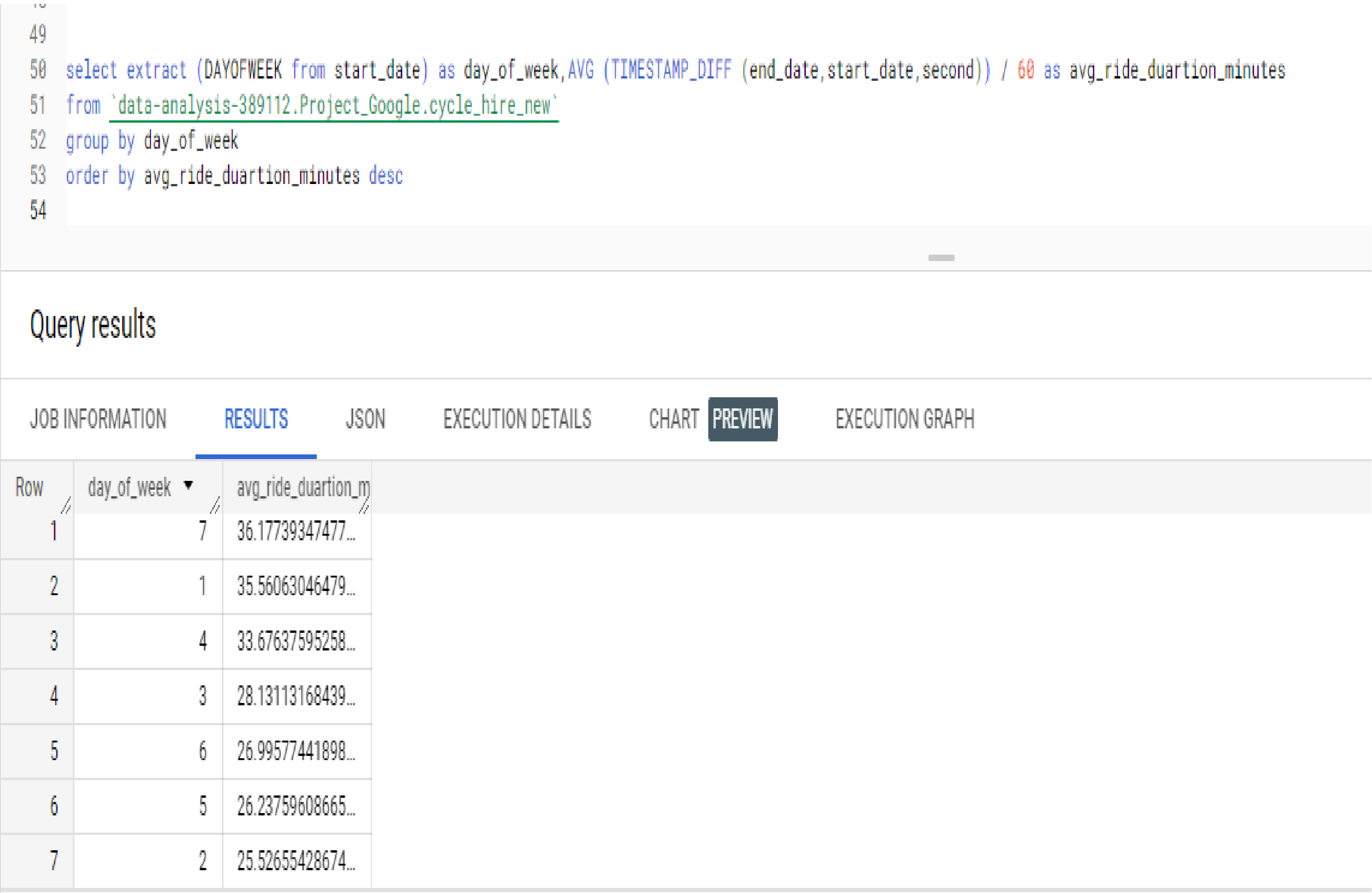
```
WITH q1 AS (
  SELECT *
  FROM `data-analysis-389112.Project_Google.cycle_hire_new`
  WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2021-04-01')
    AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')
)
SELECT FORMAT_TIMESTAMP('%A', start_date) AS day_of_week,
COUNT(*) as frequency
FROM `data-analysis-389112.Project_Google.cycle_hire_new`
GROUP BY day_of_week
ORDER BY frequency DESC
```

Query results

Row	day_of_week	frequency
1	Saturday	11218
2	Sunday	8692
3	Tuesday	7237
4	Friday	6153
5	Wednesday	5905
6	Monday	5517
7	Thursday	4293

As we could see Saturday was the best day for our stations best on frequency but we also wanted to know the avg time people rented our bikes for each day with this query:

```
 select extract (DAYOFWEEK from start_date) as day_of_week, AVG (TIMESTAMP_DIFF
(end_date,start_date,second)) / 60 as avg_ride_duration_minutes
  from `data-analysis-389112.Project_Google.cycle_hire_new`
  group by day_of_week
  order by avg_ride_duration_minutes desc
```

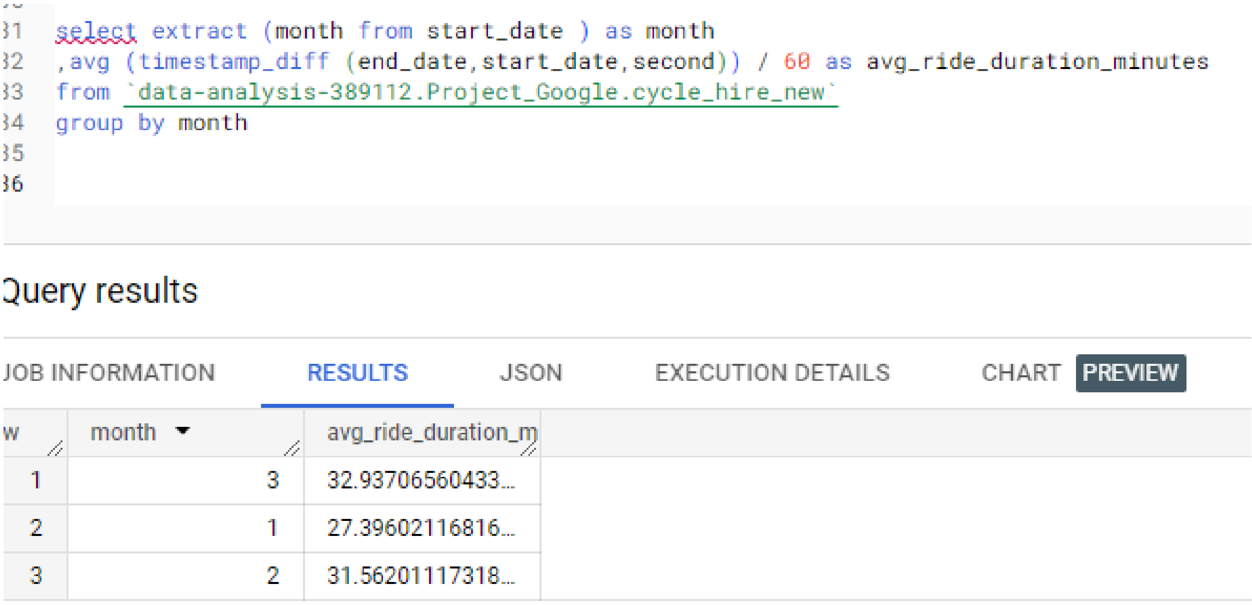


```
select extract (DAYOFWEEK from start_date) as day_of_week,AVG (TIMESTAMP_DIFF (end_date,start_date,second)) / 60 as avg_ride_duartion_minutes
from `data-analysis-389112.Project_Google.cycle_hire_new`
group by day_of_week
order by avg_ride_duartion_minutes desc
```

Row	day_of_week ▾	avg_ride_duartion_m
1	7	36.17739347477...
2	1	35.56063046479...
3	4	33.67637595258...
4	3	28.13113168439...
5	6	26.99577441898...
6	5	26.23759608665...
7	2	25.52655428674...

And just like that we were positive that people rent way more in the weekend, But we didnt stopped there there was so much more to explore, and so many questions to ask ourselves, we started by dividing the Q to 3 single months to see which month was the strongest.

```
  select extract (month from start_date ) as month, avg (timestamp_diff
(end_date,start_date,second)) / 60 as avg_ride_duration_minutes
  from `data-analysis-389112.Project_Google.cycle_hire_new`
   group by month
```



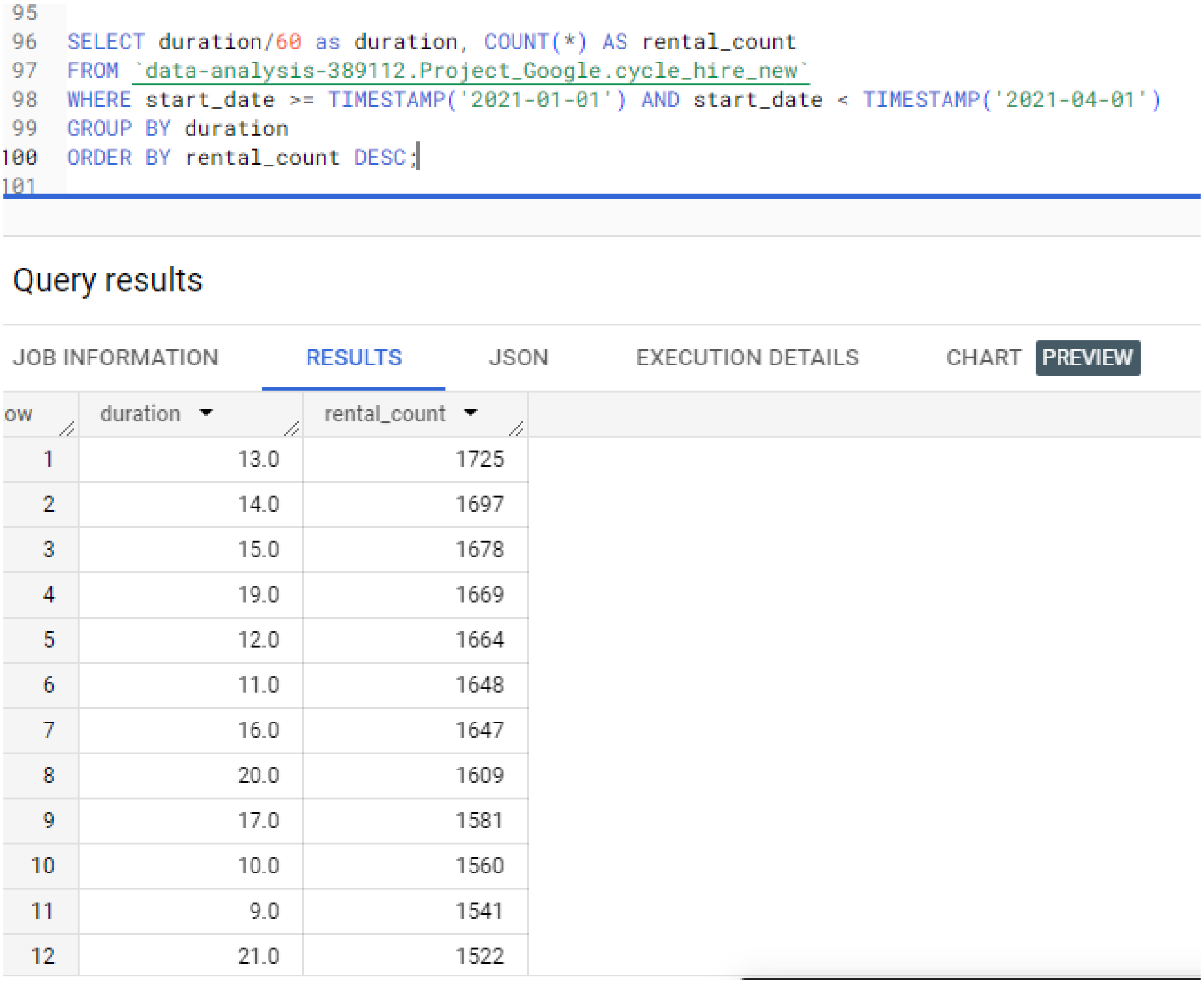
```
select extract (month from start_date ) as month
,avg (timestamp_diff (end_date,start_date,second)) / 60 as avg_ride_duration_minutes
from `data-analysis-389112.Project_Google.cycle_hire_new`
group by month
```

w	month	avg_ride_duration_m
1	3	32.93706560433...
2	1	27.39602116816...
3	2	31.56201117318...

As you can see the month of march was the strongest month with a big 5.54% upgrade since january, and we started investigating why it could be, based on weather and major events.

We also did a query to see all of the total times people took our bikes for rental to had a better understanding the data and the avg, and look at anomalies as well

```
   select duration/60 as duration, COUNT (*) AS rental_count,
   FROM `data-analysis-389112.Project_Google.cycle_hire_new`
   WHERE start_date >= TIMESTAMP ('2021-01-01') AND start_date < TIMESTAMP
('2021-04-01')
   GROUP BY duration
   ORDER BY rental_count DESC
```



```
SELECT duration/60 as duration, COUNT(*) AS rental_count
FROM `data-analysis-389112.Project_Google.cycle_hire_new`
WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2021-04-01')
GROUP BY duration
ORDER BY rental_count DESC;
```

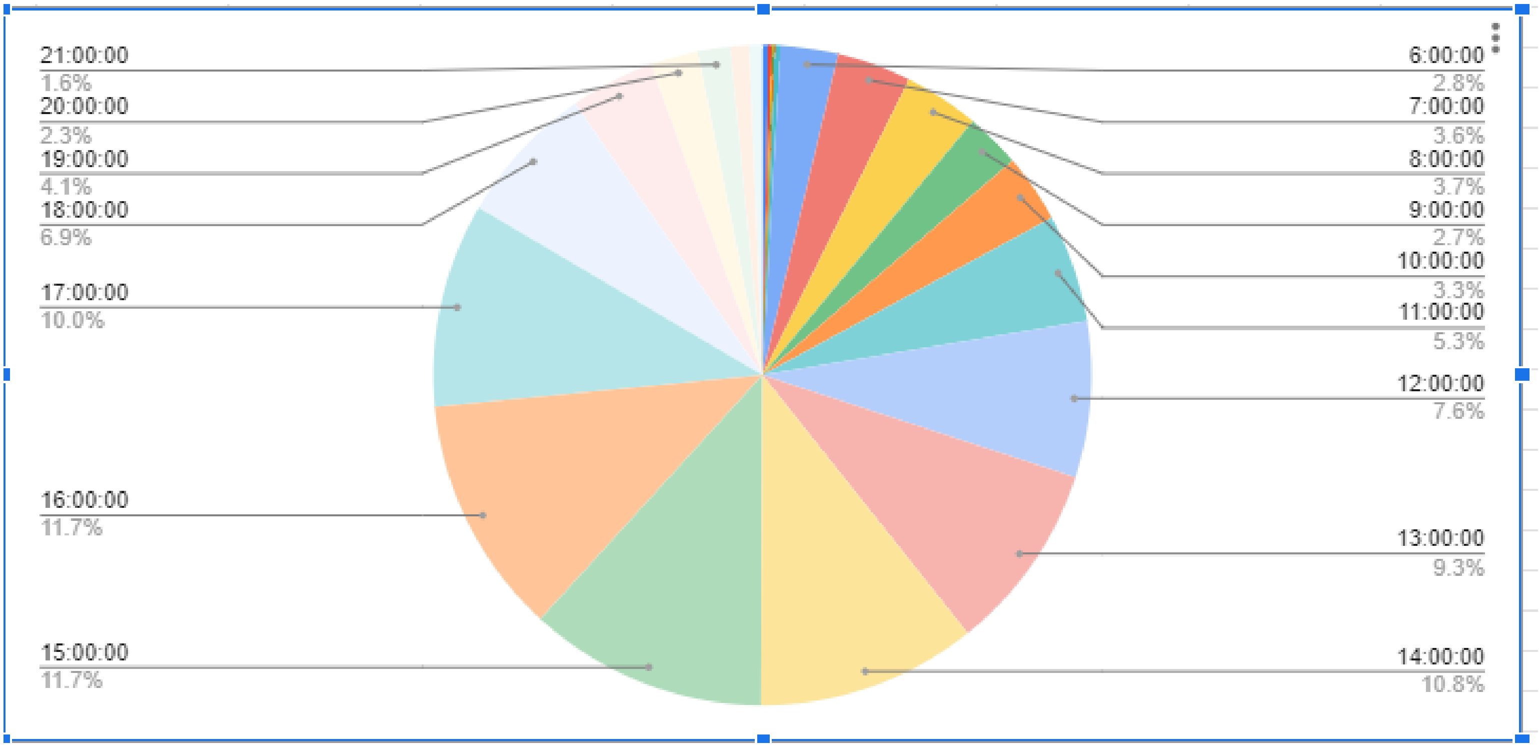
ow	duration	rental_count
1	13.0	1725
2	14.0	1697
3	15.0	1678
4	19.0	1669
5	12.0	1664
6	11.0	1648
7	16.0	1647
8	20.0	1609
9	17.0	1581
10	10.0	1560
11	9.0	1541
12	21.0	1522

We also wanted to know which time of the day we have the most rentals with this query : `SELECT`

```
                                                          ,start_station_name
```

```
,FORMAT_TIMESTAMP('%I %p', start_date) AS rental_hour
COUNT(*) AS rental_count
FROM
`data-analysis-389112.Project_Google.cycle_hire_new`
WHERE
('start_date >= TIMESTAMP('2021-01-01
('AND start_date < TIMESTAMP('2021-04-01
GROUP BY
start_station_name, rental_hour
ORDER BY
;Rental_hour
```

And after doing this query we took it to an excel sheet and had a pie chart showing in the best possible way all of the times and how much people based on the daily rented from us.



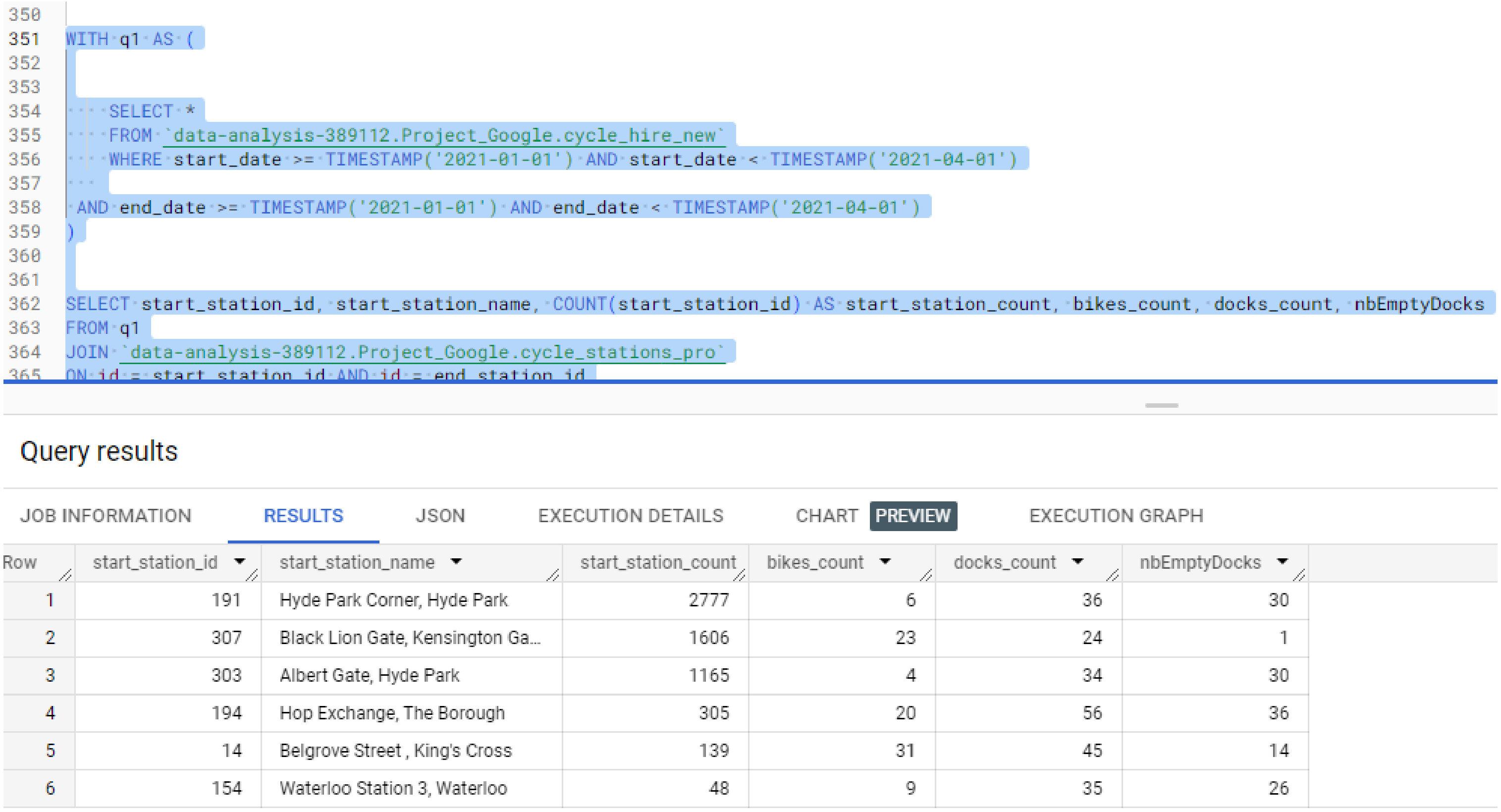
As we could see the traffic and interest in our bikes went up in around 12:00 and came to its peek at 15:00 and then went down at around 18.

After gathering all the information we needed there was the last thing we wanted to see, and thats if all of the stations are utilized correctly, and by that i mean for us to check if the number of docks count is higher or lower that it needed to be.

So we did the following query:

```
WITH q1 AS (



    SELECT *

    FROM `data-analysis-389112.Project_Google.cycle_hire_new`

    WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2021-04-01')


 AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')

)



SELECT start_station_id, start_station_name, COUNT(start_station_id) AS start_station_count, bikes_count, docks_count, nbEmptyDocks

FROM q1
```

```
JOIN `data-analysis-389112.Project_Google.cycle_stations_pro`

ON id = start_station_id AND id = end_station_id

WHERE id IN (14, 191, 307, 303, 194, 154)

GROUP BY start_station_id, start_station_name, docks_count, bikes_count, nbEmptyDocks

ORDER BY start_station_count DESC
```



```

WITH q1 AS (


    SELECT *
    FROM `data-analysis-389112.Project_Google.cycle_hire_new`
    WHERE start_date >= TIMESTAMP('2021-01-01') AND start_date < TIMESTAMP('2021-04-01')
    
 AND end_date >= TIMESTAMP('2021-01-01') AND end_date < TIMESTAMP('2021-04-01')
)


SELECT start_station_id, start_station_name, COUNT(start_station_id) AS start_station_count, bikes_count, docks_count, nbEmptyDocks
FROM q1
JOIN `data-analysis-389112.Project_Google.cycle_stations_pro`
ON id = start_station_id AND id = end_station_id
```

Row	start_station_id	start_station_name	start_station_count	bikes_count	docks_count	nbEmptyDocks
1	191	Hyde Park Corner, Hyde Park	2777	6	36	30
2	307	Black Lion Gate, Kensington Ga...	1606	23	24	1
3	303	Albert Gate, Hyde Park	1165	4	34	30
4	194	Hop Exchange, The Borough	305	20	56	36
5	14	Belgrove Street , King's Cross	139	31	45	14
6	154	Waterloo Station 3, Waterloo	48	9	35	26

And the results are as expected there are so many docks in Waterloo,king cross and The borough, which need to be condensed, If you take a look based on the docks count in black lion gate and look at its performance its remarkable, we understood now that people love all of the nearby hyde park stations, and we need to start utilizing them in a better way and try to make the most of our bikes, especially in our strongest hours.