**Cyclistic: A Bike Rental Service Analysis**

* **Ask**

Analyze and differentiate between behaviors of casual and member riders to find trends and insights. This information will be utilized to create a marketing campaign which will aim at converting Casual riders to Member riders as the stakeholders have noted member riders are more profitable than casual riders.

**Insights:**

1. How do annual members and casual riders use Cyclistic bikes differently?
2. Why would casual riders buy Cyclistic annual memberships?
3. How can Cyclistic use digital media to influence casual riders to become members?

* **Prepare**

The dataset is divided into 12 files each for a month of the year 2021. They all contain 13 columns and over **100,000** rows of data. A simple peek through conditional formatting/=CountIf() shows us there are many empty values in each file. This is accounted for in the cleaning process.

* **Process**

The first step is to combine the individual monthly datasets into 1 collective dataset. For all the processing and analyzation process, we’ll use BigQuery and SQL queries.

\*DML statements have been compensated for through the creation of new tables as BigQuery doesn’t allow DML statements in Sandbox mode.

\*The datasets for the months August-October were divided into 2 halves to make the csv files size less than 100mb as BigQuery doesn’t allow local files larger than 100mb to be uploaded.

Create table cyclistic-analysis-381219.Dataset.bike\_data as (

Select \* from (

Select \* from cyclistic-analysis-381219.Dataset.Jan\_21

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Feb\_21

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.March\_21

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.April\_21

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.May\_21

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.June\_21\_a

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.June\_21\_b

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.July\_21\_a

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.July\_21\_b

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.August\_21\_a

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.August\_21\_b

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Sep\_21\_a

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Sep\_21\_b

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Oct\_21\_a

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Oct\_21\_b

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Nov\_21

UNION ALL

Select \* from cyclistic-analysis-381219.Dataset.Dec\_21 ) )

The resulting table has **5,570,960** rows.

**Cleaning:**

Through the following query, we find that **1038673** rows contain NULL values for start\_station\_name, start\_station\_id, end\_station\_name, or end\_station\_id columns.

with table\_data as (

Select \* from cyclistic-analysis-381219.Dataset.bike\_data )

SELECT count(\*) FROM table\_data where start\_station\_name is NULL or start\_station\_id is NULL or end\_station\_name is NULL or end\_station\_id is NULL

We remove these rows by creating a new table as follows:

create table cyclistic-analysis-381219.Dataset.bike\_data\_v2 as (

SELECT \* FROM cyclistic-analysis-381219.Dataset.bike\_data where start\_station\_name is not NULL and start\_station\_id is not NULL and end\_station\_name is not NULL and end\_station\_id is not NULL)

These null containing rows constitute **18.64%** of the total data. So, this cleaning step costs us a significant portion of (although unusable) data.

Now we are left with **4,532,287** rows of data.

Ride ID is the primary key in this dataset and if we check for any possible duplicates for it, we find none.

with table\_data as (

Select \* from cyclistic-analysis-381219.Dataset.bike\_data\_v2)

SELECT DISTINCT count(\*) FROM table\_data

Gives **4,532,287** which is equal to total number of rows hence no duplicates.

We check for other inconsistencies like no 4th value in rideable\_types column, no 3rd value in member\_casual column etc. and find none.

In the final step of cleaning, we extract time from started\_at and ended\_at columns to make it easy to calculate the total time of each ride.

create table cyclistic-analysis-381219.Dataset.bike\_data\_v3 as (

SELECT \*, Extract(time from started\_at) as start\_time, Extract(time from ended\_at) as end\_time FROM cyclistic-analysis-381219.Dataset.bike\_data\_v2 )

* **Analyze:**

We have **2,517,975** Members and **2,014,312** Casual riders in our dataset.

SELECT member\_casual, count(\*) FROM `cyclistic-analysis-381219.Dataset.bike\_data\_v6` group by member\_casual

The first step in our analysis is to calculate the total time taken by each ride. We use the following query to perform this step:

create table cyclistic-analysis-381219.Dataset.bike\_data\_v4 as (

Select \*, DATE\_DIFF(ended\_at, started\_at, minute) AS ride\_time from cyclistic-analysis-381219.Dataset.bike\_data\_v3)

A little bit of cleaning is performed on this newly created column by removing all those entries where the ride time is less than 0 minutes or greater than 24 hours.

create table cyclistic-analysis-381219.Dataset.bike\_data\_v5 as (

select \* from cyclistic-analysis-381219.Dataset.bike\_data\_v4 where ride\_time >= 0 and ride\_time <= 1440)

Now we are left with **4,530,947** rows of data.

Now we create two additional columns, one for month and one for day of each ride through the following query:

create table cyclistic-analysis-381219.Dataset.bike\_data\_v6 as (

Select \*, case

When extract(DAYOFWEEK from started\_at) = 1 Then 'Sun'

When extract(DAYOFWEEK from started\_at) = 2 Then 'Mon'

When extract(DAYOFWEEK from started\_at) = 3 Then 'Tue'

When extract(DAYOFWEEK from started\_at) = 4 Then 'Wed'

When extract(DAYOFWEEK from started\_at) = 5 Then 'Thu'

When extract(DAYOFWEEK from started\_at) = 6 Then 'Fri'

When extract(DAYOFWEEK from started\_at) = 7 Then 'Sat'

End as day, case

When extract(Month from started\_at) = 1 Then 'Jan'

When extract(Month from started\_at) = 2 Then 'Feb'

When extract(Month from started\_at) = 3 Then 'Mar'

When extract(Month from started\_at) = 4 Then 'Apr'

When extract(Month from started\_at) = 5 Then 'May'

When extract(Month from started\_at) = 6 Then 'Jun'

When extract(Month from started\_at) = 7 Then 'Jul'

When extract(Month from started\_at) = 8 Then 'Aug'

When extract(Month from started\_at) = 9 Then 'Sep'

When extract(Month from started\_at) = 10 Then 'Oct'

When extract(Month from started\_at) = 11 Then 'Nov'

When extract(Month from started\_at) = 12 Then 'Dec'

End as month

from cyclistic-analysis-381219.Dataset.bike\_data\_v5 )

Average length of rides for each group:

|  |  |
| --- | --- |
| **Casual** | 27.93 minutes |
| **Member** | 12.93 minutes |

with table\_data as (

Select \* from cyclistic-analysis-381219.Dataset.bike\_data\_v6)

Select avg(ride\_time) from table\_data where member\_casual = 'member'

Most/Least popular days for each group:

|  |  |  |
| --- | --- | --- |
| **Type** | **Most Popular Day** | **Least Popular Day** |
| Casual | Saturday | Tuesday |
| Member | Wednesday | Sunday |

with table\_data as (

Select \* from cyclistic-analysis-381219.Dataset.bike\_data\_v6)

Select count(\*) as rides, day from table\_data where member\_casual = 'casual' group by day order by rides DESC

Most/Least popular months for each group:

|  |  |  |
| --- | --- | --- |
| **Type** | **Most Popular Month** | **Least Popular Month** |
| Casual | September | February |
| Member | September | February |

Select count(\*) as rides, month from table\_data where member\_casual = 'casual' group by month order by rides DESC

Most/least popular bikes for each group:

|  |  |  |
| --- | --- | --- |
| **Type** | **Most Popular Bike** | **Least Popular Bike** |
| Casual | Classic | Docked |
| Member | Classic | Docked |

Select count(\*) as rides, rideable\_type from table\_data where member\_casual = 'member' group by rideable\_type order by rides DESC

\*There is only a **single** case where a member used a Docked bike. They only use Classic or Electric.

Most/least popular hours for rides for each group:

|  |  |  |
| --- | --- | --- |
| **Type** | **Most popular hours** | **Least popular hours** |
| Casual | 17, 18, 16 (5 PM, 6 PM, 4 PM) | 3 AM, 5 AM, 4 AM |
| Member | 17, 18, 16 (5 PM, 6 PM, 4 PM) | 2 AM, 4 AM, 3 AM |

with table\_data as (

SELECT member\_casual, Extract(hour from started\_at) as starting\_hour, count(\*) as total\_rides,

from cyclistic-analysis-381219.Dataset.bike\_data\_v6

group by member\_casual, starting\_hour

)

Select \* from (Select \*, row\_number() over (partition by member\_casual order by total\_rides desc) AS ride\_rank from table\_data) where ride\_rank <= 3

This shows us that the riders are most active at evening hours especially around 5 PM and least active during early morning hours around 4 AM.

* **Share:**

No analysis is complete without a proper dashboard/visualization of the facts and figures found during the process. For visualization of this project, we downloaded the final dataset from BigQuery and loaded it into Tablaeu.

The visualization can be found here:

https://public.tableau.com/views/CyclisticBikeRentalServiceAnalysisCapstoneProjectforGoogleDataAnalyticsCourse/CyclisticDashboard

* **Act:**
* To convert occasional riders into regular members, the marketing campaign should target the summer season and specifically the hours between 4pm and 6pm when ridership tends to increase.
* It's also important to focus on weekends when casual riders are most active. One way to entice occasional riders to become members is to offer a summer membership package.
* Another effective strategy is to remind casual riders that bikes can be picked up from one station and returned to any other station, at any time, which could encourage them to upgrade to an annual membership instead of buying day passes.
* Since regular members are more profitable for the company, it's recommended to incentivize them to spread the word about the benefits of membership. For instance, offering a slightly discounted membership fee to members who successfully recruit a casual rider into becoming a member.

**END**