CYCLISTI CASE STUDY

WITH

----------------------------> Combine 12 monthly dataset to 1 yearly dataset <-------------------------

year\_data AS

(

SELECT \*

FROM Cyclistic\_TripData\_202006

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202007

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202008

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202009

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202010

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202011

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202012

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202101

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202102

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202103

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202104

UNION ALL

SELECT \*

FROM Cyclistic\_TripData\_202105

),

---------------------------> Clear empty cells from the dataset <----------------------------------------

null\_cleaned AS

(

SELECT \*

FROM year\_data WHERE start\_station\_name NOT LIKE '%NULL%'

AND end\_station\_name NOT LIKE '%NULL%'

AND start\_lat NOT LIKE '%NULL%'

AND start\_lng NOT LIKE '%NULL%'

AND end\_lat NOT LIKE '%NULL%'

AND end\_lng NOT LIKE '%NULL%'

),

---------------------------> Aggregated rider length as Minutes, and assign respective day of the week <---------------------------

aggre\_data AS (

SELECT \*,

DATEDIFF(MINUTE,started\_at, ended\_at) as TotalMinutes,

CASE

WHEN day\_of\_week = 1 THEN 'Sunday'

WHEN day\_of\_week = 2 THEN 'Monday'

WHEN day\_of\_week = 3 THEN 'Tuesday'

WHEN day\_of\_week = 4 THEN 'Wednesday'

WHEN day\_of\_week = 5 THEN 'Thursday'

WHEN day\_of\_week = 6 THEN 'Friday'

ELSE

'Saturday'

END

AS Day\_Week

FROM null\_cleaned

),

---------------------------> Remove Ride\_id with characters more than 16 <-------------------------------------------------------------

clean\_ride\_id\_data AS

(

SELECT \*

FROM aggre\_data

WHERE LEN(ride\_id) = 16 AND TotalMinutes >= 1

),

---------------------------> TRIM station name to ensure no extra space, and to replace (\*), (Temp), filter out row with(LBS-WH-TEST) in start station name <------------------

cstart\_station\_name\_data AS

(

SELECT ride\_id,

TRIM(REPLACE

(REPLACE

(start\_station\_name, '(\*)',''),

'(TEMP)','')) AS start\_station\_name\_clean

FROM clean\_ride\_id\_data

WHERE start\_station\_name NOT LIKE '%(LBS-WH-TEST)%'

),

---------------------------> TRIM station name to ensure no extra space, and to replace (\*), (Temp), filter out row with(LBS-WH-TEST) in end station name <------------------

cend\_station\_name\_data AS

(

SELECT ride\_id,

TRIM(REPLACE

(REPLACE

(end\_station\_name, '(\*)',''),

'(TEMP)','')) AS end\_station\_name\_clean

FROM clean\_ride\_id\_data

WHERE end\_station\_name NOT LIKE '%(LBS-WH-TEST)%'

),

---------------------------> JOIN both cleaned station column ON ride\_id <---------------------------------------------------------

station\_name AS

(

SELECT ss.ride\_id, ss.start\_station\_name\_clean, es.end\_station\_name\_clean

FROM cstart\_station\_name\_data ss

JOIN cend\_station\_name\_data es

ON ss.ride\_id = es.ride\_id

),

---------------------------> JOIN clean station columns to the main dataset ON ride\_id <---------------------------------------------------------

final\_table AS

(

Select sn.ride\_id, crid.rideable\_type, crid.member\_casual, crid.Day\_Week,

CAST(crid.started\_at AS date) AS Date\_ofYear, crid.ended\_at, crid.TotalMinutes,

sn.start\_station\_name\_clean, sn.end\_station\_name\_clean,

crid.start\_lat, crid.start\_lng,

crid.end\_lat, crid.end\_lng

FROM clean\_ride\_id\_data crid

JOIN station\_name sn

ON crid.ride\_id = sn.ride\_id

),

-------------------------------------------------------------------------------------------------------------------------------------------------------

--------------------------------------------------------->> DATA EXPLORATION & ANALYSIS <<-------------------------------------------------------------

---------------------------> 1. Find out total numbers of member or casual riders DEPARTING from respective stations <---------------------------------

casual\_depart\_station AS

(

SELECT COUNT(member\_casual) AS Casual, start\_station\_name\_clean

FROM final\_table

WHERE member\_casual = 'casual'

GROUP BY start\_station\_name\_clean

),

member\_depart\_station AS

(

SELECT COUNT(member\_casual) AS Member, start\_station\_name\_clean

FROM final\_table

WHERE member\_casual = 'member'

GROUP BY start\_station\_name\_clean

),

---------------------------> Join member & casual riders ON departing bike stations <----------------------------------------------------------------

depart\_station AS

(

SELECT cds.start\_station\_name\_clean, cds.Casual, mds.Member

FROM casual\_depart\_station cds

JOIN member\_depart\_station mds

ON cds.start\_station\_name\_clean = mds.start\_station\_name\_clean

),

---------------------------> GROUP departing station name with distinct Latitude and Altitude <------------------------------------------------------

depart\_latlng AS

(

SELECT DISTINCT start\_station\_name\_clean, ROUND(AVG(start\_lat),4) AS dep\_lat, Round(AVG(start\_lng),4) AS dep\_lng

FROM final\_table

GROUP BY start\_station\_name\_clean

),

---------------------------> Join location coordinate data with ridership count <---------------------------------------------------------------------

---------------------------- (Export to excel & import to tableau for geo-visualisation)--------------------------------------------------------------

locationviz\_depart AS

(

SELECT dl.start\_station\_name\_clean, ds.Casual, ds.Member, dl.dep\_lat, dl.dep\_lng

FROM depart\_station ds

JOIN depart\_latlng dl

ON ds.start\_station\_name\_clean = dl.start\_station\_name\_clean

),

---------------------------> 2. Find out total numbers of member or casual riders ARRIVING for respective stations <----------------------------------

casual\_arrive\_station AS

(

SELECT COUNT(member\_casual) AS Casual, end\_station\_name\_clean

FROM final\_table

WHERE member\_casual = 'casual'

GROUP BY end\_station\_name\_clean

),

member\_arrive\_station AS

(

SELECT COUNT(member\_casual) AS Member, end\_station\_name\_clean

FROM final\_table

WHERE member\_casual = 'member'

GROUP BY end\_station\_name\_clean

),

---------------------------> Join member & casual riders ON arriving bike stations <------------------------------------------------------------------

arrive\_station AS

(

SELECT cas.end\_station\_name\_clean, cas.Casual, mas.Member

FROM casual\_arrive\_station cas

JOIN member\_arrive\_station mas

ON cas.end\_station\_name\_clean = mas.end\_station\_name\_clean

),

---------------------------> GROUP arriving station name with distinct Latitude and Altitude <------------------------------------------------------

arrive\_latlng AS

(

SELECT DISTINCT end\_station\_name\_clean, ROUND(AVG(end\_lat),4) AS arr\_lat, Round(AVG(end\_lng),4) AS arr\_lng

FROM final\_table

GROUP BY end\_station\_name\_clean

),

---------------------------> Join location coordinate data with ridership count <---------------------------------------------------------------------

---------------------------- (Export to excel & import to tableau for geo-visualisation)--------------------------------------------------------------

locationviz\_arrive AS

(

SELECT al.end\_station\_name\_clean, ast.Casual, ast.Member, al.arr\_lat, al.arr\_lng

FROM arrive\_station ast

JOIN arrive\_latlng al

ON ast.end\_station\_name\_clean = al.end\_station\_name\_clean

),

-----------> 3. To find out trips taken by casual & members respectively group by days <---------------------------------

day\_trips\_casual AS

(

SELECT count(member\_casual) AS casual, Date\_ofYear

FROM final\_table

WHERE member\_casual = 'casual'

GROUP BY Date\_ofYear

),

day\_trips\_member AS

(

SELECT count(member\_casual) AS member, Date\_ofYear

FROM final\_table

WHERE member\_casual = 'member'

GROUP BY Date\_ofYear

),

day\_trip\_year AS

(

SELECT dtc.casual, dtm.member, dtm.Date\_ofYear

FROM day\_trips\_casual dtc

JOIN day\_trips\_member dtm

ON dtc.Date\_ofYear = dtm.Date\_ofYear

),

---------------------------------> 3. To find out trips taken by casual & members respectively group by the DAY OF THE WEEK <-------------------------------------------

Day\_Week\_trips\_casual AS

(

SELECT count(member\_casual) AS casual, Day\_Week

FROM final\_table

WHERE member\_casual = 'casual'

GROUP BY Day\_Week

),

Day\_Week\_trips\_member AS

(

SELECT count(member\_casual) AS member, Day\_Week

FROM final\_table

WHERE member\_casual = 'member'

GROUP BY Day\_Week

),

Day\_Week\_trips AS

(

SELECT dwtc.Day\_Week, dwtc.casual, dwtm.member

FROM Day\_Week\_trips\_casual dwtc

JOIN Day\_Week\_trips\_member dwtm

ON dwtc.Day\_Week = dwtm.Day\_Week

),

------------------------------> 4. To find the AVERAGE ride time for Casual & Member Riders <----------------------------------------------------------

data\_totalmin\_casual AS

(

SELECT AVG(TotalMinutes) AS AVG\_ride\_casual

FROM final\_table

WHERE member\_casual = 'casual'

),

data\_totalmin\_member AS

(

SELECT AVG(TotalMinutes) AS AVG\_ride\_member

FROM final\_table

WHERE member\_casual = 'member'

),

----------------------------> 5. To find the Overall Rider Count for Casual & Member riders <-----------------------------------------------------------

totalride\_casual AS

(

SELECT count(member\_casual) AS ridership\_casual

FROM final\_table

WHERE member\_casual = 'casual'

),

totalride\_member AS

(

SELECT count(member\_casual) AS ridership\_member

FROM final\_table

WHERE member\_casual = 'member'

)

SELECT \*

FROM totalride\_member