

## Cyclistic Data Cleaning and Exploration with SQL Queries

```
1  /*
2
3  /*
4  Cyclistic Bike Data Cleaning and Exploration in SQL Queries
5
6  Skills Used: Temporary Table, CTE, Aggregate Functions, Window Function, Timestamp Function, Conditional Function, Subquery
7
8  Platform: GCP BigQuery
9
10 */
11
```

### -- DATA CLEANING IN SQL QUERIES

#### -- Preview of the four tables

```
12
13
14
15
16
17
18 SELECT
19 *
20 FROM
21 `causal-bison-323215.Cyclistic.DivvyTrips2019Q1` LIMIT 20;
22
23 SELECT
24 *
25 FROM
26 `causal-bison-323215.Cyclistic.DivvyTrips2019Q2` LIMIT 20;
27
28 SELECT
29 *
30 FROM
31 `causal-bison-323215.Cyclistic.DivvyTrips2019Q3` LIMIT 20;
32
33 SELECT
34 *
35 FROM
36 `causal-bison-323215.Cyclistic.DivvyTrips2019Q4` LIMIT 20;
37
```

#### -- Change DivvyTrips2019Q2 column names to same as Q1, Q3 and Q4 using temp table

```
38
39
40 BEGIN
41 CREATE OR REPLACE TEMPORARY TABLE DivvyTrips2019Q2Temp
42 AS
43 SELECT
44     _01__Rental_Details_Rental_ID AS trip_id,
45     _01__Rental_Details_Local_Start_Time AS start_time,
46     _01__Rental_Details_Local_End_Time AS end_time,
47     _01__Rental_Details_Bike_ID AS bikeid,
48     _01__Rental_Details_Duration_In_Seconds_Uncapped AS tripduration,
49     _03__Rental_Start_Station_ID AS from_station_id,
50     _03__Rental_Start_Station_Name AS from_station_name,
51     _02__Rental_End_Station_ID AS to_station_id,
52     _02__Rental_End_Station_Name AS to_station_name,
53     User_Type AS usertype,
54     Member_Gender AS gender,
55     _05__Member_Details_Member_Birthday_Year AS birthyear
56
57 FROM
58 `causal-bison-323215.Cyclistic.DivvyTrips2019Q2`;
59 End;
60
```

#### -- Temp table DivvyTrips2019Q2Temp stored as causal-bison-323215.\_script2cb74db8e872be8c7189bb08b16c7d5c3fba71e6.DivvyTrips2019Q2Temp

```
61
62
63 SELECT
64 *
65 FROM
66 `causal-bison-323215._script2cb74db8e872be8c7189bb08b16c7d5c3fba71e6.DivvyTrips2019Q2Temp` LIMIT 10
67
68
```

#### -- Merge DivvyTrips2019Q, DivvyTrips2019Q2Temp, DivvyTrips2019Q3 and DivvyTrips2019Q4 into one table

```
69
70
71 BEGIN
72 CREATE OR REPLACE TEMP TABLE DivvyTrips2019
73 AS
74 SELECT
75 *
76 FROM
77 `causal-bison-323215.Cyclistic.DivvyTrips2019Q1`
78 UNION ALL
79 SELECT
80 *
81 FROM
82 `causal-bison-323215._script2cb74db8e872be8c7189bb08b16c7d5c3fba71e6.DivvyTrips2019Q2Temp`
83 UNION ALL
84 SELECT
```

```

85      *
86 FROM
87   `causal-bison-323215.Cyclistic.DivvyTrips2019Q3`
88 UNION ALL
89 SELECT
90      *
91 FROM
92   `causal-bison-323215.Cyclistic.DivvyTrips2019Q4`;
93 END;
94

```

```

95 -- Temp table DivvyTrips2019 stored `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
96
97 SELECT
98      *
99 FROM
100   `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
101

```

```

102 -- Alternatively, one could create a CTE to rename the columns in DivvyTrips2019Q2, union the CTE with the other
103 three tables and save the result into a new table, DivvyTrips2019V2
104

```

```

105 WITH NewDivvyTrips2019Q2 AS (SELECT
106   _01__Rental_Details_Rental_ID AS trip_id,
107   _01__Rental_Details_Local_Start_Time AS start_time,
108   _01__Rental_Details_Local_End_Time AS end_time,
109   _01__Rental_Details_Bike_ID AS bikeid,
110   _01__Rental_Details_Duration_In_Seconds_Uncapped AS tripduration,
111   _03__Rental_Start_Station_ID AS from_station_id,
112   _03__Rental_Start_Station_Name AS from_station_name,
113   _02__Rental_End_Station_ID AS to_station_id,
114   _02__Rental_End_Station_Name AS to_station_name,
115   User_Type AS usertype,
116   Member_Gender AS gender,
117   _05__Member_Details_Member_Birthday_Year AS birthyear
118 FROM
119   `causal-bison-323215.Cyclistic.DivvyTrips2019Q2`
120 )
121 SELECT
122      *
123 FROM
124   `causal-bison-323215.Cyclistic.DivvyTrips2019Q1`
125 UNION ALL
126 SELECT
127      *
128 FROM
129   NewDivvyTrips2019Q2
130 UNION ALL
131 SELECT
132      *
133 FROM
134   `causal-bison-323215.Cyclistic.DivvyTrips2019Q3`
135 UNION ALL
136 SELECT
137      *
138 FROM
139   `causal-bison-323215.Cyclistic.DivvyTrips2019Q4`;
140
141

```

```

142 -- Check if the primary key column, trip-id, has duplicate value
143

```

```

144 SELECT
145   COUNT(DISTINCT trip_id) AS unique_count_tripId,
146   COUNT(*) AS total_number_rows
147 FROM
148   `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
149

```

```

150 -- Check if columns contained nulls
151

```

```

152 SELECT
153   COUNTIF(trip_id IS NULL) AS trip_id,
154   COUNTIF(start_time IS NULL) AS start_time,
155   COUNTIF(end_time IS NULL) AS end_time,
156   COUNTIF(tripduration IS NULL) AS tripduration,
157   COUNTIF(bikeid IS NULL) AS bikeid,
158   COUNTIF(from_station_id IS NULL) AS from_station_id,
159   COUNTIF(from_station_name IS NULL) AS from_station_name,
160   COUNTIF(to_station_id IS NULL) AS to_station_id,
161   COUNTIF(to_station_name IS NULL) AS to_station_name,
162   COUNTIF(usertype IS NULL) AS usertype,
163   COUNTIF(gender IS NULL) AS gender,
164   COUNTIF(birthyear IS NULL) AS birthyear
165 FROM
166   `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;
167

```

```

168 -- Examine the birthyear column
169
170 SELECT
171     MIN(birthyear) AS min_birthyear,
172     MAX(birthyear) AS max_birthyear
173 FROM
174     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;
175
176
177 -- Update the gender column; replace nulls with 'Unknown'
178
179 UPDATE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
180 SET gender = 'Unknown'
181 WHERE gender IS NULL;
182
183 SELECT
184     *
185 FROM
186     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
187 WHERE
188     gender IS NULL;
189
190 -- Create and populate column ride_length
191
192 ALTER TABLE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
193 ADD COLUMN ride_length INTEGER;
194
195 UPDATE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
196 SET ride_length = TIMESTAMP_DIFF(end_time, start_time, SECOND)
197 WHERE trip_id IS NOT NULL;
198
199 SELECT
200     MIN(ride_length) AS min_ride_duration,
201     MAX(ride_length) AS max_ride_duration
202 FROM
203     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;
204
205 SELECT
206     COUNT(*) AS rows_with_negative_ride_length,
207 FROM
208     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
209 WHERE
210     ride_length <= 0;
211
212 -- Delete rows with negative ride_length value
213
214 DELETE
215 FROM
216     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
217 WHERE
218     ride_length <= 0;
219
220 SELECT
221     COUNT(*) AS rows_with_negative_ride_length,
222 FROM
223     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
224 WHERE
225     ride_length <= 0;
226
227 SELECT
228     MIN(ride_length) AS min_ride_duration,
229     MAX(ride_length) AS max_ride_duration
230 FROM
231     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;
232
233 -- Create and populate day_of_week column
234
235 ALTER TABLE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
236 ADD COLUMN day_of_week INTEGER;
237
238 UPDATE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
239 SET day_of_week = EXTRACT(DAYOFWEEK FROM start_time)
240 WHERE trip_id IS NOT NULL;
241
242 SELECT
243     *
244 FROM
245     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019` LIMIT 10;
246
247 SELECT
248     MIN(day_of_week) AS min_day_of_week,
249     MAX(day_of_week) AS max_day_of_week
250 FROM
251     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;

```

```

252 -- Create and populate day column
253
254 ALTER TABLE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
255 ADD COLUMN day STRING;
256
257 UPDATE `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
258 SET day = CASE day_of_week
259 WHEN 1 THEN 'Sunday'
260 WHEN 2 THEN 'Monday'
261 WHEN 3 THEN 'Tuesday'
262 WHEN 4 THEN 'Wednesday'
263 WHEN 5 THEN 'Thursday'
264 WHEN 6 THEN 'Friday'
265 WHEN 7 THEN 'Saturday'
266 ELSE NULL
267 END
268 WHERE trip_id IS NOT NULL;
269
270 SELECT
271 *
272 FROM
273 `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019` LIMIT 10;
274
275 SELECT
276 COUNTIF(day IS NULL) AS count
277 FROM
278 `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;
279
280

```

```

281 -----
282
283 -- EXPLORATORY DATA ANALYSIS WITH SQL QUERIES
284

```

```

285 -- Different types of usertype and their breakdown by count and gender
286
287 SELECT
288     usertype AS user,
289     gender AS gender,
290     COUNT(*) AS number
291 FROM
292     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
293 GROUP BY usertype, gender
294 ORDER BY usertype, gender;
295
296 -- Average, lowest and highest ride duration, number of trips and busiest day of 2019
297
298 SELECT
299     ROUND(AVG(ride_length), 2) AS average_ride_duration_sec,
300     MIN(ride_length) AS lowest_ride_duration_sec,
301     MAX(ride_length) AS highest_ride_duration_sec,
302     COUNT(trip_id) AS total_trip,
303     APPROX_TOP_COUNT(day, 1) AS busiest_day
304 FROM
305     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;
306

```

```

307 -- Average, lowest and highest ride duration, number of trips and busiest day for each type of user
308
309 SELECT
310     usertype AS user,
311     ROUND(AVG(ride_length), 2) AS average_ride_duration_sec,
312     MIN(ride_length) AS lowest_ride_duration_sec,
313     MAX(ride_length) AS highest_ride_duration_sec,
314     COUNT(trip_id) AS total_trip,
315     APPROX_TOP_COUNT(day, 1) AS busiest_day
316 FROM
317     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
318 GROUP BY usertype;
319

```

```

320 -- Proportion of rides above average ride duration for each type of user
321
322 SELECT
323     usertype AS user,
324     ROUND(AVG(ride_length), 2) AS avg_ride_duration_sec,
325     COUNTIF((ride_length - avg_ride_duration) > 0) AS ride_above_avg_duration,
326     COUNT(trip_id) AS total_trip,
327     ROUND(COUNTIF((ride_length - avg_ride_duration) > 0)/COUNT(trip_id)*100, 2) AS percent_ride_above_avg_duration,
328 FROM (
329     SELECT
330         usertype,
331         ride_length,
332         trip_id,
333         AVG(ride_length) OVER (
334             PARTITION BY usertype

```

```

335     ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING)
336     AS avg_ride_duration,
337     FROM
338     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019` )
339 GROUP BY usertype;
340
341 -- Average ride duration for users by the day of the week
342
343 SELECT
344     usertype AS user,
345     day AS day,
346     ROUND(AVG(ride_length), 2) AS avg_ride_duration,
347 FROM
348     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
349 GROUP BY usertype, day
350 ORDER BY usertype;
351
352 -- Number of rides for users by day of the week
353
354 SELECT
355     usertype AS user,
356     day AS day,
357     COUNT(trip_id) AS total_trip
358 FROM
359     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
360 GROUP BY usertype, day
361 ORDER BY usertype;
362
363 -- Average ride duration for users by the day of the week by quarter
364
365 SELECT
366     usertype AS user,
367     EXTRACT(QUARTER FROM start_time) AS quarter,
368     day AS day,
369     ROUND(AVG(ride_length), 2) AS avg_ride_duration
370 FROM
371     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
372 GROUP BY usertype, quarter, day
373 ORDER BY usertype, quarter;
374
375 -- Number of rides for users by day of the week by quarter
376
377 SELECT
378     usertype AS user,
379     EXTRACT(QUARTER FROM start_time) AS quarter,
380     day AS day,
381     COUNT(trip_id) AS total_trip
382 FROM
383     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`
384 GROUP BY usertype, quarter, day
385 ORDER BY usertype, quarter;
386
387 -- Save temporary table as BigQuery table for export in order to visualize with Tableau
388
389 SELECT
390     trip_id,
391     start_time,
392     end_time,
393     bikeid,
394     from_station_id,
395     from_station_name,
396     to_station_id,
397     to_station_name,
398     usertype,
399     ride_length,
400     day
401 FROM
402     `causal-bison-323215._script203775267b33d1a1aef3c416b8f33c8849058ea0.DivvyTrips2019`;

```