**Data Cleaning and Handling Documentation**

The data comes from a primary source: <https://divvy-tripdata.s3.amazonaws.com/index.html>.  
A sample size was used for each month, calculated using the corresponding formula.

*Due to the large amount of data and not having a paid BigQuery account, the following approach was used:*

* Load the 12 months into Excel via BigQuery.
* Remove the columns that will not be used for analysis.
* Calculate how many data points are needed for the random sampling.
* Remove empty rows.
* Perform random sampling for each month.
* Merge the random samples from each month into a single sheet.
* Perform a final random sample using the formula.

When time is available, a larger sample size will be covered:

* 99% confidence level / 1% margin of error.
* Final sample size.
* Two new columns were created:
  + “ride\_length”: calculates the difference between “ended\_at” and “started\_at”.
  + “day\_of\_week”: determines the day of the week based on the “started\_at” column using the “WEEKDAY()” function.
* Duplicate data was removed.
* The file was saved in CSV format to continue in BigQuery.

**In BigQuery**

The CSV file was imported into BigQuery within the “project” dataset under the name “random\_sample”.

1. **Detection of duplicate values:** To check for duplicate records in the table, the DISTINCT clause was used:

SELECT DISTINCT \* FROM proyecto.sample;

1. **Date Format Verification**  
   It was verified that the date and time fields (started\_at and ended\_at) were correctly stored in the appropriate format (TIMESTAMP in this case).
2. **Verification of Trip Date Validity**  
   It was verified that no records had future dates in the started\_at or ended\_at fields:

SELECT COUNT(\*)

FROM proyecto.sample

WHERE started\_at > CURRENT\_TIMESTAMP()

OR ended\_at > CURRENT\_TIMESTAMP();

The result was 0, indicating there are no records with future dates.

1. **Null Value Analysis**  
   The percentage of null values was calculated to reconfirm that they had been removed and assess whether further removal was necessary.

SELECT

(COUNTIF(end\_station\_name IS NULL AND end\_station\_name IS NULL ) / COUNT(\*)) \* 100 AS null\_percentage

FROM proyecto.sample;

The result showed the percentage of affected records, helping determine if it was viable to remove them.

1. **Elimination of Records with Multiple Null Values**  
   BigQuery does not allow the DELETE operation directly on standard tables unless you have a paid account. As an alternative, a new table was created without records containing null values in BOTH mentioned columns:

CREATE OR REPLACE TABLE proyecto.sample

SELECT \*

FROM proyecto.sample

WHERE NOT (end\_station\_name IS NULL AND start\_station\_name IS NULL );

This ensures the data is more complete and reliable for analysis.

1. **User Category Validation**  
   To ensure that the member\_casual column only contains two types of users (member and casual), the following query was executed:

SELECT member\_casual, COUNT(\*)

FROM proyecto.sample

GROUP BY member\_casual;

The result confirmed there were no unexpected values in this column.

1. **Random Sampling Table Creation**  
   A table was created with the random sample:

CREATE OR REPLACE TABLE proyecto.sample\_limited AS

SELECT \*

FROM proyecto.sample

LIMIT <sample\_size>;

1. **Descriptive Analysis**
2. /\* Descriptive analytics \*/
3. SELECT
4. day\_of\_week,
5. COUNT(\*) AS total\_trips,
6. MIN(TIME\_DIFF(ride\_length, TIME '00:00:00', SECOND)) / 60 AS min\_ride\_length\_minutes,
7. MAX(TIME\_DIFF(ride\_length, TIME '00:00:00', SECOND)) / 60 AS max\_ride\_length\_minutes,
8. AVG(TIME\_DIFF(ride\_length, TIME '00:00:00', SECOND)) / 60 AS avg\_ride\_length\_minutes,
9. STDDEV(TIME\_DIFF(ride\_length, TIME '00:00:00', SECOND)) / 60 AS stddev\_ride\_length\_minutes
10. FROM
11. project.random\_sample
12. GROUP BY
13. day\_of\_week
14. ORDER BY
15. day\_of\_week;
16. /\* Moda day\_of\_week \*/
17. SELECT
18. day\_of\_week, COUNT(\*) AS moda
19. FROM
20. project.random\_sample
21. GROUP BY day\_of\_week
22. ORDER BY moda DESC
23. LIMIT 1;

9. **Creation of Queries for Graphing:**

Query for the number of trips and average trip duration by season.

* /\* For graphic averages \*/
* SELECT
* day\_of\_week,
* CASE
* WHEN EXTRACT(MONTH FROM started\_at) IN (12, 01, 02) THEN 'Winter'
* WHEN EXTRACT(MONTH FROM started\_at) IN (03, 04, 05) THEN 'Spring'
* WHEN EXTRACT(MONTH FROM started\_at) IN (06, 07, 08) THEN 'Summer'
* WHEN EXTRACT(MONTH FROM started\_at) IN (09, 10, 11) THEN 'Autum'
* END AS season,
* member\_casual,
* COUNT(\*) AS total\_trips,
* AVG(TIME\_DIFF(ride\_length, TIME '00:00:00', SECOND)) / 60 AS avg\_ride\_length\_minutes
* FROM
* `project.random\_sample`
* GROUP BY
* day\_of\_week,
* season,
* member\_casual
* ORDER BY
* season,
* day\_of\_week,
* member\_casual;

Save that query as a .csv file

10. Query to confirm the 10 busiest stations

* /\* The most 10 used stations for casuals (start\_station and end \_stations) \*/
* SELECT
* start\_station\_name AS start\_station,
* end\_station\_name AS end\_station,
* member\_casual,
* COUNT(\*) AS total\_trips\_start,
* COUNT(\*) AS total\_trips\_end
* FROM
* project.random\_sample
* WHERE member\_casual= 'casual'
* GROUP BY
* start\_station\_name,
* member\_casual,
* end\_station\_name
* ORDER BY
* member\_casual,
* total\_trips\_start DESC,
* total\_trips\_end DESC
* LIMIT 10;

11. Query for the number of trips by type of bicycle:

* /\* Numbers of trips on bicycle type \*/
* SELECT
* rideable\_type,
* day\_of\_week,
* Seasons,
* COUNT(\*) AS total\_bike\_trips
* FROM
* project.random\_sample
* GROUP BY
* rideable\_type,
* day\_of\_week,
* Seasons
* ORDER BY
* Seasons,
* day\_of\_week,
* total\_bike\_trips DESC;

save that query as a .csv file

12. Query for the 3 most demanding hours, number of trips according to the day and season.

* /\* Number of trips per hour, day and season \*/
* WITH TripCounts AS (
* SELECT
* EXTRACT(HOUR FROM started\_at) AS hour\_of\_day,
* FORMAT\_DATE('%A', DATE(started\_at)) AS day\_of\_week,
* CASE
* WHEN EXTRACT(MONTH FROM started\_at) IN (12, 1, 2) THEN 'winter'
* WHEN EXTRACT(MONTH FROM started\_at) IN (3, 4, 5) THEN 'spring'
* WHEN EXTRACT(MONTH FROM started\_at) IN (6, 7, 8) THEN 'summer'
* WHEN EXTRACT(MONTH FROM started\_at) IN (9, 10, 11) THEN 'autumn'
* END AS season,
* COUNT(\*) AS total\_trips
* FROM
* `project.random\_sample`
* GROUP BY
* hour\_of\_day, day\_of\_week, season
* ),
* RankedTrips AS (
* SELECT
* hour\_of\_day,
* day\_of\_week,
* season,
* total\_trips,
* ROW\_NUMBER() OVER (PARTITION BY season, day\_of\_week ORDER BY total\_trips DESC) AS row\_num
* FROM
* TripCounts
* )
* SELECT
* hour\_of\_day,
* day\_of\_week,
* season,
* total\_trips
* FROM
* RankedTrips
* WHERE
* row\_num <= 3
* ORDER BY
* season, day\_of\_week, total\_trips DESC, hour\_of\_day;

save that query as a .csv file

**In R**

Creation of 16 graphs with the .csv files:

- 4 graphs for the number of trips for members and casual commuters for each day of the week by season.

- 4 graphs for the average trip duration for members and casual commuters for each day of the week by season.

- 4 graphs for bicycle use by day of the week by season.

- 4 graphs for the use of each type of bicycle by season.

**-PowerPoint presentation**