***Documentación de limpieza y manipulación de datos***

Los datos son de una fuente primaria: <https://divvy-tripdata.s3.amazonaws.com/index.html>

Usamos un tamaño de la muestra para cada mes con la fórmula para calcular el mismo.

\*Debido a la cantidad de datos y no tener cuenta paga en BigQuery se procede de la siguiente forma:

- Cargar los 12 meses en Excel a través de BigQuery.

-Eliminación de columnas que no vamos a utilizar para el análisis.

- Catar cuantos datos precisamos para el muestreo aleatorio.

- Eliminación de filas vacías.

- Realizar un muestreo aleatorio para cada mes.

- Fusionar los muestreos aleatorios de cada mes en una sola hoja.

- Realizar un muestreo aleatorio final a través de la fórmula.

Al tener disponibilidad de tiempo abarcaremos un tamaño de muestra mayor:

* 99% nivel de confianza / 1% margen de error.
* Tamaño de la muestra final:
* Se crearon dos nuevas columnas: “ride\_length”: calcula la diferencia entre “ended\_at” y “started\_at” .

“day\_of\_week”: determina que día es a partir de la columna “started\_at”, usando la función “DIASEMANA()

* Se eliminaron datos duplicados.
* Se guarda en formato csv para seguir en BigQuery.

***En BigQuery***

Se importo el archivo CSV a BigQuery dentro del dataset “proyecto” con el nombre “random\_sample”.

1. ***Detección de valores duplicados.***

Para verificar si existen registros duplicados en la tabla, se utilizó la cláusula DISTINCT:

SELECT DISTINCT \* FROM proyecto.sample;

1. ***Verificación de formato fecha***

Se comprobó que los campos de fecha y hora (started\_at y ended\_at) estuvieran correctamente almacenados en el formato adecuado (TIMESTAMP en este caso) .

1. ***Verificación de la validez de las fechas de viaje***

Se comprobó que no existieran registros con fechas futuras en los campos started\_at o ended\_at:

SELECT COUNT(\*)

FROM proyecto.sample

WHERE started\_at > CURRENT\_TIMESTAMP()

OR ended\_at > CURRENT\_TIMESTAMP();

El resultado fue 0, lo que indica que no hay registros con fechas posteriores a la actual.

1. ***Análisis de valores nulos***

Se calculó el porcentaje de valores nulls para re-confirmar que hayan sido eliminados y evaluar si es necesaria su eliminación.

SELECT

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

FROM proyecto.sample;

El resultado mostró el porcentaje de registros afectados, lo que ayudó a determinar si era viable eliminarlos.

1. ***Eliminación de registros con múltiples valores nulos***

BigQuery no permite la operación DELETE de manera directa en tablas estándar a menos que se tenga una cuenta paga. Como alternativa, se creó una nueva tabla sin los registros con valores nulos en las DOS columnas mencionadas:

CREATE OR REPLACE TABLE proyecto.sample

SELECT \*

FROM proyecto.sample

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

Esto garantiza que los datos sean más completos y confiables para el análisis.

1. ***Validación de categorías de usuarios***

Para asegurarse de que la columna member\_casual solo contenga dos tipos de usuarios (member y casual), se ejecutó la siguiente consulta:

SELECT member\_casual, COUNT(\*)

FROM proyecto.sample

GROUP BY member\_casual;

El resultado confirmó que no hay valores inesperados en esta columna.

1. Se creo una tabla con el muestreo aleatorio ().

CREATE OR REPLACE TABLE proyecto.sample\_limited AS

SELECT \*

FROM proyecto.sample

LIMIT ;

1. **Análisis descriptivo.**
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. Creación de las consultas para graficár:***

***\* Consulta para el número de viajes y promediode viaje según estación del año.***

* /\* 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;

Guardamos esa consulta como archivo .csv

* ***Consulta para confirmar las 10 estaciones más concurridas:***
* /\* 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;
* ***Consulta para el numero de viajes por tipo de bicicleta:***
* /\* 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;

Guardamos esa consulta como archivo .csv

* ***Consulta para las 3 horas más demandantes, cantidad de viajes según el día y la estación:***
* /\* 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;

Guardamos esa consulta como archivo .csv

***En R***

Creación de 16 gráficas con los archivos .csv:

- 4 para la cantidad de viajes para miembros y casuales para cada día de la semana según la estación del año.

- 4 para el promedio de duración del viaje para miembros y casuales para cada día de la semana según la estación del año.

- 4 gráficas para uso de bicicleta por día de la semana según estación.

- 4 gráficas para el uso de cada tipo de bicicleta según estación.

***Presentación en PowerPoint***