Proceso en BigQuery

- **Conectar/importar datos.**

Se ha creado un proyecto en BigQuery, se agregó un dataset al cual se le han agregado las tablas, track_in_spotify, track_in_competition y track_technical_info.

Se hizo una corrección manual en el encabezado del nombre del artista en “track in spotify”, contiene caracteres no identificados (paréntesis).

- **Identificar y manejar valores nulos.**

Se ha identificado nulos a través de comandos SQL en:
track_technical_info; key_null: 95
track_in_competition; in_shazam_charts: 50

```
--Query para identificar nulos
SELECT
  COUNTIF(track_id IS NULL) AS track_id_null,
  COUNTIF(track_name IS NULL) AS track_name_null,
  COUNTIF(artists_name IS NULL) AS artists_name_null,
  COUNTIF(artist_count IS NULL) AS artist_count_null,
  COUNTIF(released_year IS NULL) AS released_year_null,
  COUNTIF(released_month IS NULL) AS released_month_null,
  COUNTIF(released_day IS NULL) AS released_day_null,
  COUNTIF(in_spotify_playlists IS NULL) AS in_spotify_playlists_null,
  COUNTIF(in_spotify_charts IS NULL) AS in_spotify_charts_null,
  COUNTIF(streams IS NULL) AS streams_null,
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
;

SELECT
  COUNTIF(track_id IS NULL) AS track_id_null,
  COUNTIF(in_apple_playlists IS NULL) AS in_apple_playlists_null,
  COUNTIF(in_apple_charts IS NULL) AS in_apple_charts_null,
  COUNTIF(in_deezer_playlists IS NULL) AS in_deezer_playlists_null,
  COUNTIF(in_deezer_charts IS NULL) AS in_deezer_charts_null,
  COUNTIF(in_shazam_charts IS NULL) AS in_shazam_charts_null,
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_competition`
;

SELECT
  COUNTIF(track_id IS NULL) AS track_id_null,
  COUNTIF(bpm IS NULL) AS bpm_null,
```

```
  COUNTIF(key IS NULL) AS key_null,
  COUNTIF(mode IS NULL) AS mode_null,
  COUNTIF(`danceability_%` IS NULL) AS danceability_null,
  COUNTIF(`valence_%` IS NULL) AS valence_null,
  COUNTIF(`energy_%` IS NULL) AS energy_null,
  COUNTIF(`acousticness_%` IS NULL) AS acousticness_null,
  COUNTIF(`instrumentalness_%` IS NULL) AS instrumentalness_null,
  COUNTIF(`liveness_%` IS NULL) AS liveness_null,
  COUNTIF(`speechiness_%` IS NULL) AS speechiness_null,
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_technical_info`
```

- **Identificar y manejar valores duplicados.**

Se han identificado duplicados a través de comandos SQL, en:
track _in_ spotify, a través de track_name y artists_name, con 4 casos, es decir, 8 elementos.

```
-- Query para identificar duplicados
SELECT
track_name, artists_name,
COUNT(*) AS cantidad
FROM `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
GROUP BY track_name, artists_name
HAVING COUNT(*) > 1

--Query para seleccionar los duplicados
WITH duplicados AS (
  SELECT
    track_name,
    artists_name,
    COUNT(*) AS duplicado
  FROM
    `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
  GROUP BY
    track_name,
    artists_name
  HAVING
    COUNT(*) > 1
)

SELECT
  original.*
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify` AS original
JOIN
```

```
  duplicados
ON
  original.track_name = duplicados.track_name
  AND original.artists_name = duplicados.artists_name
ORDER BY
  original.track_name,
  original.artists_name
```

Se identificó que cada una de las canciones duplicadas, se decidió, quitar una versión de cada canción. Las canciones eliminadas, cuentan con el siguiente track id: 7173596, 3814670, 8173823, 1119309. A la par se buscó duplicados en las otras dos tablas a partir del track_id, no se encontró coincidencias.

- **Identificar y manejar datos fuera del alcance del análisis.**

Se han manejado datos fuera del alcance a través de comandos SQL. Se identificaron los campos key y mode en “track technical info”, como campos con información técnica no relevante para la validación de hipótesis. Aunado a ello el campo key cuenta con una gran cantidad de nulos. Se utilizó la siguiente query:

```
--Query para eliminar campos a partir de EXCEPT
SELECT
*
EXCEPT (key,mode)
FROM `proyecto2-hipotesis-spotify.dataset_spotify.track_technical_info`
```

- **Identificar y manejar datos discrepantes en variables categóricas**

Se ha identificado y manejado datos discrepantes en variables categóricas a través de comandos SQL. Se encontró en “track name” y “artists name” de la tabla “track in spotify”, el uso de símbolos, los cuales se reemplazaron a través de la función REGEXP_REPLACE. Se utilizó el siguiente query:

```
--Query para quitar símbolos
SELECT
  REGEXP_REPLACE(track_name, r'[^a-zA-Z0-9]', ' ') AS track_name_limpo,
  REGEXP_REPLACE(artists_name, r'[^a-zA-Z0-9 ]', ' ') AS artist_name_limpio,
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
```

- **Identificar y manejar datos discrepantes en variables numérica**

Se ha identificado y manejado datos discrepantes en variables numéricas a través de comandos SQL. Se calculó el mínimo, máximo y promedio de cada tabla. Sin embargo, para el campo streams en “track in spotify", se identificó que los valores son de tipo STRING aunque sean números, además se encontró un valor como una línea de texto grande, que impedía el cálculo del promedio y el máximo. Se utilizó SAFE_CAST para cambiar el tipo STRING por INTEGER, así como evitar algún error.

```sql
--Query mínimo, máximo y promedio
SELECT
  MAX(in_apple_playlists) AS max_apple_playlis,
  MIN(in_apple_playlists) AS min_apple_playlis,
  AVG(in_apple_playlists) AS avg_apple_playlist,
  MAX(in_apple_charts) AS max_apple_charts,
  MIN(in_apple_charts) AS min_apple_charts,
  AVG(in_apple_charts) AS avg_apple_charts,
  MAX(in_deezer_playlists) AS max_in_deezer_playlist,
  MIN(in_deezer_playlists) AS min_in_deezer_playlist,
  AVG (in_deezer_playlists) AS avg_in_deezer_playlist,
  MAX(in_deezer_charts) AS max_in_deezer_charts,
  MIN(in_deezer_charts) AS min_deezer_charts,
  AVG(in_deezer_charts) AS avg_deezer_charts,
  MAX(in_shazam_charts) AS max_shazam_charts,
  MIN(in_shazam_charts) AS min_shazam_charts,
  AVG (in_shazam_charts) AS avg_shazam_charts
  FROM
 `proyecto2-hipotesis-spotify.dataset_spotify.track_in_competition`
;
SELECT
  MAX(in_spotify_playlists) AS max_in_spotify_playlists,
  MIN(in_spotify_playlists) AS min_in_spotify_playlists,
  AVG(in_spotify_playlists) AS avg_in_spotify_playlists,
  MAX(in_spotify_charts) AS max_spotify_charts,
  MIN(in_spotify_charts) AS min_spotify_charts,
  AVG(in_spotify_charts) AS avg_spotify_charts,
  MAX(SAFE_CAST (streams AS INT64)) AS max_streams,
  MIN(streams) AS min_streams,
  AVG(SAFE_CAST (streams AS INT64)) AS avg_streams,
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`;
```

- **Comprobar y cambiar tipo de dato**

Se ha cambiado el tipo de dato a través de comandos SQL. Se utilizó la función SAFE_CAST, en el campo de streams de la tabla “track in spotify”, para cambiar los valores de STRING a INTEGER. Además se usó NOT LIKE para omitir un dato que contiene una línea de texto. Se utilizó la siguiente query:

```sql
--Query cambiar datos
WITH cleaned_data AS (
  SELECT
    SAFE_CAST(streams AS INT64) AS streams_limpio,
```

```
    *
  FROM
    `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
)

SELECT
  track_id, streams_limpio
FROM
  cleaned_data
WHERE
  streams NOT LIKE "%BPM%"
```

- **Crear nuevas variables**

Se ha creado una nueva variable a través de comandos SQL. Para la tabla track in spotify, se creó la variable de “fecha de lanzamiento” a partir de concatenar las fechas de año, mes y día a partir de la función DATE y SAFE_CAST. Se hizo la siguiente query:

```
--Query para fecha
SELECT
  DATE( SAFE_CAST(released_year AS INT64), SAFE_CAST(released_month AS INT64),
SAFE_CAST(released_day AS INT64) ) AS release_date
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
```

Además se creó la variable “total playlist” para determinar la participación total en playlists. Esta query se incorporó en un LEFT JOIN posterior a partir de la suma de:
in_spotify_playlists + in_apple_playlists + in_deezer_playlists AS total_playlist

- **Unir tablas**

Se ha utilizado un LEFT JOIN para crear un view “dataset_spotify_02”, con la información de las tres tablas, sin embargo, como paso previo se decidió limpiar cada una de las tablas y crear un view limpio. A continuación se explica el proceso.

View “track in competition limpio”
Este view se creó a partir de toda la selección, se considera que todos los campos y registros son fundamentales para el estudio, además de que no se encontraron duplicados, y los nulos solo corresponden a 50 registros en in_shazam_chart.

```
--Query view track in competition limpio
SELECT
  *
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_competition`
```

View “track in spotify limpio”
En este view, se limpió los símbolos de los campos track name y artist name con REGEXP_REPLACE, se cambió el tipo de dato STRING a INTEGER con SAFE_CAST, además, se agregó la fecha de lanzamiento con el formato año, mes, día con DATE. Por último, se eliminó el registro de línea de texto antes identificado en streams y una versión de cada canción repetida con NOT LIKE. Se utilizó las siguiente query:

```
--Query view track in spotify limpio
SELECT
  *,
  REGEXP_REPLACE(track_name, r'[^a-zA-Z0-9 ]', ' ') AS track_name_limpio,
  REGEXP_REPLACE(artists_name, r'[^a-zA-Z0-9 ]', ' ') AS artist_name_limpio,
  SAFE_CAST(streams AS INT64) AS streams_limpio,
  DATE( SAFE_CAST(released_year AS INT64), SAFE_CAST(released_month AS INT64),
SAFE_CAST(released_day AS INT64) ) AS release_date
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify`
WHERE
  streams NOT LIKE '%BPM%'
  AND track_id != '7173596'
  AND track_id != '3814670'
  AND track_id != '8173823'
  AND track_id != '1119309'
```

View “track technical info limpio”
En este view, se decidió quitar los campos key y mode con un EXCEPT, debido a la cantidad de nulos y al ser información no relevante para la validación de hipótesis. El query quedó de la siguiente manera:

```
--Query view track in technical info limpio
SELECT
  * EXCEPT (KEY,
    mode)
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.track_technical_info`
```

Por último, se creó un LEFT JOIN con las tres vistas limpias que se llamó “dataset_spotify_02”, a continuación se muestra el query del LEFT JOIN. Se pidió en esta consulta ordenar los campos de manera específica.

```
--Query view dataset_spotify_02
SELECT
TS.track_id,
```

```
TS.track_name_limpio,
TS.artist_name_limpio,
TS.artist_count,
TS.released_year,
TS.released_month,
TS.released_day,
TS.release_date,
TS.in_spotify_playlists,
TS.in_spotify_charts,
TS.streams_limpio,
TC.in_apple_playlists,
TC.in_apple_charts,
TC.in_deezer_playlists,
TC.in_deezer_charts,
TC.in_shazam_charts,
TI.bpm,
TI.`danceability_%`,
TI.`valence_%`,
TI.`energy_%`,
TI.`acousticness_%`,
TI.`instrumentalness_%`,
TI.`liveness_%`,
TI.`speechiness_%`,
TS.in_spotify_playlists + TC.in_apple_playlists + TC.in_deezer_playlists AS
total_playlist
 FROM `proyecto2-hipotesis-spotify.dataset_spotify.track_in_spotify_limpio` TS
 left join
`proyecto2-hipotesis-spotify.dataset_spotify.track_in_competition_limpio` TC
 on TS.track_id = TC.track_id
 left join
`proyecto2-hipotesis-spotify.dataset_spotify.track_technical_info_limpio` TI
 on TS.track_id = TI.track_id
```

Cabe mencionar que posteriormente a “dataset_spotify_02”, se le hicieron modificaciones por lo que la view del consolidado será el “data set spotify 03”. En pasos subyacentes se explica cómo se llegó al resultado.

- **Construir tablas auxiliares**

Se ha creado una tabla temporal con WITH, canciones por artista solista.

```
WITH canciones_por_artista AS (
  SELECT
    artist_name_limpio,
    COUNT(track_id) AS total_canciones,
```

```
    SUM(streams_limpio) AS total_streams
  FROM
    `proyecto2-hipotesis-spotify.dataset_spotify.dataset_spotify_03`
  WHERE
    artist_count = 1
  GROUP BY
    artist_name_limpio
)

SELECT
  *
FROM
```

- **Calcular cuartiles, deciles o percentiles**

Se han calculado los cuartiles para las variables de, streams, bpm, y los porcentajes de bailabilidad (danceability), valencia (valence), energía (energy), acústica (acousticness), instrumentalidad (instrumentalness), vivacidad (liveness), habla (speechiness). Se clasificó como “alto” al cuartil 4, y como “bajo” a todo lo demás. Se utilizó la siguiente query:

```
--Query view dataset_spotify_03 cuartiles
WITH
  Quartiles AS (
  SELECT
    streams_limpio,
    bpm,
    `danceability_%`,
    `valence_%`,
    `energy_%`,
    `acousticness_%`,
    `instrumentalness_%`,
    `liveness_%`,
    NTILE(4) OVER (ORDER BY streams_limpio) AS quartile_streams,
    NTILE(4) OVER (ORDER BY bpm) AS quartile_bpm,
    NTILE(4) OVER (ORDER BY `danceability_%`) AS quartile_danceability,
    NTILE(4) OVER (ORDER BY `valence_%`) AS quartile_valence,
    NTILE(4) OVER (ORDER BY `energy_%`) AS quartile_energy,
    NTILE(4) OVER (ORDER BY `acousticness_%`) AS quartile_acousticness,
    NTILE(4) OVER (ORDER BY `instrumentalness_%`) AS quartile_instrumentalness,
    NTILE(4) OVER (ORDER BY `liveness_%`) AS quartile_liveness,
    NTILE(4) OVER (ORDER BY `speechiness_%`) AS quartile_speechiness,
  FROM
    `proyecto2-hipotesis-spotify.dataset_spotify.dataset_spotify_02` )
SELECT
  a.*,
```

```
IF
  (Quartiles.quartile_streams = 4, "alto", "bajo") AS cat_streams,
IF
  (Quartiles.quartile_bpm = 4, "alto", "bajo") AS cat_bpm,
IF
  (Quartiles.quartile_danceability = 4, "alto", "bajo") AS cat_danceability,
IF
  (Quartiles.quartile_valence = 4, "alto", "bajo") AS cat_valence,
IF
  (Quartiles.quartile_energy = 4, "alto", "bajo") AS cat_energy,
IF
  (Quartiles.quartile_acousticness = 4, "alto", "bajo") AS cat_acousticness,
IF
  (Quartiles.quartile_instrumentalness = 4, "alto", "bajo") AS
cat_instrumentalness,
IF
  (Quartiles.quartile_liveness = 4, "alto", "bajo") AS cat_liveness,
IF
  (Quartiles.quartile_speechiness = 4, "alto", "bajo") AS cat_speechiness,
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.dataset_spotify_02` a
LEFT JOIN
  Quartiles
ON
  a.streams_limpio = Quartiles.streams_limpio
  AND a.`danceability_%` = Quartiles.`danceability_%`
```

- **Calcular correlación entre variables**

Se ha calculado la correlación entre dos variables, de acuerdo a las hipótesis presentadas. A continuación se muestra la query y por último el resultado de correlación y su interpretación.

```
SELECT
  --Hipotesis1
  CORR(streams_limpio,bpm) AS correlation_streams_bpm,
  --Hipotesis2
  CORR(in_spotify_charts,in_deezer_charts) AS correlation_spotifyc_deezer,
  CORR(in_spotify_charts,in_apple_charts) AS correlation_spotifyc_apple,
  CORR(in_spotify_charts,in_shazam_charts) AS correlation_spotifyc_shazam,
  --Hipotesis 3
  CORR(streams_limpio, total_playlist) AS correlation_streams_playlist,
  --Hipotesis 5
  CORR(streams_limpio, `danceability_%`) AS correlation_streams_danceability,
  CORR(streams_limpio, `valence_%`) AS correlation_streams_valence,
```

```sql
  CORR(streams_limpio, `energy_%`) AS correlation_streams_energy,
  CORR(streams_limpio, `acousticness_%`) AS correlation_streams_acousticness,
  CORR(streams_limpio, `instrumentalness_%`) AS
correlation_streams_instrumentalness,
  CORR(streams_limpio, `liveness_%`) AS correlation_streams_liveness,
;
  --Hipotesis 4
SELECT
  CORR(total_streams, total_canciones) AS correlation_tstreams_tcanciones
FROM
  `proyecto2-hipotesis-spotify.dataset_spotify.canciones_solistas_streams`
```