```sql
#SELECT para visualizar minha tabela - Spotify
SELECT
  *
FROM
  `projeto--2.Hipoteses.Spotify`
LIMIT
  1000



  #COUNT para contar quantas variáveis são nulas na coluna streams da tabela Spotify
SELECT
  COUNT(*)
FROM
  `projeto--2.Hipoteses.Spotify`
WHERE
  `streams` IS NULL;



  #COUNT AS REPEAT para contar AS repetidas com base NO track_name e streams - spotify
SELECT
  track_name,
  streams,
  COUNT(*) AS REPEAT
FROM
  `projeto--2.Hipoteses.Spotify`
GROUP BY
  track_name,
  streams
HAVING
  COUNT(*) >1



  # Cláusula EXCEPT para remover coluna e criar a tabela technical_info_final -
technical_info.
SELECT
*EXCEPT
  (key)
FROM
```

```
  `projeto--2.Hipoteses.Technical_info`;



  # REGEXP para substituir string - spotify
SELECT
  track_name,artist_s__name,
  REGEXP_REPLACE(track_name, '[^a-zA-Z0-9]', ' ') AS track_name_limpo,
  REGEXP_REPLACE(artist_s__name, '[^a-zA-Z0-9]', ' ') AS artist_s__name_limpo
FROM
  `projeto--2.Hipoteses.Spotify`;



  #LOWER para deixar string minúscula
SELECT
  LOWER(track_name) AS track_name_minusculo,
  LOWER(artist_s__name) AS artist_s__name_minusculo
FROM
  `projeto--2.Hipoteses.Spotify`;



  # UPPER para deixar AS string maiúsculas
SELECT
  UPPER(track_name) AS track_name_maiusculo,
  UPPER(artist_s__name) AS artist_s__name_maiusculo
FROM
 `projeto--2.Hipoteses.Spotify`;



  # MIN. MAX, AVG para calcular máximo, mínimo e média - spotify
SELECT
  MAX(streams),
  MIN(streams),
  AVG(streams)
FROM
```

```
  `projeto--2.Hipoteses.Spotify`;


   # MIN. MAX, AVG para calcular máximo, mínimo e média. - competition
SELECT
  MAX(in_apple_playlists) AS max_in_apple_playlists,
  MIN(in_apple_playlists) AS min_in_apple_playlists,
  AVG(in_apple_playlists) AS med_in_apple_playlists,
  MAX(in_apple_charts) AS max_in_apple_charts,
  MIN(in_apple_charts) AS min_in_apple_charts,
  AVG(in_apple_charts) AS med_in_apple_charts,
  MAX(in_deezer_playlists) AS max_in_deezer_playlists,
  MIN(in_deezer_playlists) AS min_in_deezer_playlists,
  AVG(in_deezer_playlists) AS med_in_deezer_playlists,
  MAX(in_deezer_charts) AS max_in_deezer_charts,
  MIN(in_deezer_charts) AS min_in_deezer_charts,
  AVG(in_deezer_charts) AS med_in_deezer_charts,
  MAX(in_shazam_charts) AS max_in_shazam_charts,
  MIN(in_shazam_charts) AS min_in_shazam_charts,
  AVG(in_shazam_charts) AS med_in_shazam_charts
FROM
  `projeto--2.Hipoteses.Competition`;


  # MIN. MAX, AVG para calcular máximo, mínimo e média. - technical_info
SELECT
  MAX(bpm) AS max_bpm,
  MIN(bpm) AS min_bpm,
  AVG(bpm) AS med_bpm,
  MAX(danceability__) AS max_danceability,
  MIN(danceability__) AS min_danceability,
  AVG(danceability__) AS med_danceability,
  MAX(valence__) AS max_valence,
  MIN(valence__) AS min_valence,
  AVG(valence__) AS med_valence,
  MAX(energy__) AS max_energy,
  MIN(energy__) AS min_energy,
  AVG(energy__) AS med_energy,
```

```sql
  MAX(acousticness__) AS max_acousticness,
  MIN(acousticness__) AS min_acousticness,
  AVG(acousticness__) AS med_acousticness,
  MAX(instrumentalness__) AS max_instrumentalness,
  MIN(instrumentalness__) AS min_instrumentalness,
  AVG(instrumentalness__) AS med_instrumentalness,
  MAX(liveness__) AS max_liveness,
  MIN(liveness__) AS min_liveness,
  AVG(liveness__) AS med_liveness,
  MAX(speechiness__) AS max_speechiness,
  MIN(speechiness__) AS min_speechiness,
  AVG(speechiness__) AS med_speechiness
FROM `projeto--2.Hipoteses.Technical_info`;
```

```sql
  # SAFE_CAST modifica os dados de string para integer - spotify
SELECT
  SAFE_CAST (streams AS int64) AS streams_limpo,
  SAFE_CAST (track_id AS int64) AS track_id_limpo
FROM
  `projeto--2.Hipoteses.Spotify`;
```

```sql
   # Usei o SAFE_CAST para transformar os dados em INTEGER e depois calcular o valor
MIN. MAX, AVG para calcular máximo, mínimo e média - spotify
WITH MIN_MAX_MED_spotify AS (
SELECT
SAFE_CAST (streams AS int64) AS streams_limpo,
SAFE_CAST (track_id AS int64) AS track_id_limpo,
FROM
  `projeto--2.Hipoteses.Spotify`
  )
  SELECT
  MAX(streams_limpo) AS max_streams_limpo,
  MIN(streams_limpo) AS min_streams_limpo,
  AVG(streams_limpo) AS med_streams_limpo,
  MAX(track_id_limpo) AS max_track_id_limpo,
  MIN(track_id_limpo) AS min_track_id_limpo,
```

```
  AVG(track_id_limpo) AS med_track_id_limpo
FROM
  MIN_MAX_MED_spotify;




  #CONCAT para concatenar três colunas e criar uma nova com ano/mês/dia. - spotify
SELECT
  DATE(CONCAT(released_year,'-',released_month,'-',released_day)) AS released_date,
FROM
  `projeto--2.Hipoteses.Spotify`
GROUP BY
  released_month,
  released_day,
  released_year;



  #DATE para criar uma coluna nova com ano/mês/dia e SUM para somar a quantidade de
playlist por dia. - spotify
SELECT
  DATE(released_year, released_month,released_day) AS released_date,
  SUM(in_spotify_playlists) AS n_playlists
FROM
  `projeto--2.Hipoteses.Spotify`
GROUP BY
  released_date;



  # COUNT realizar a soma DO numero de musicas por artistas. Spotify
SELECT
  artist_s__name,
  COUNT(*) AS n_musicas
FROM
  `projeto--2.Hipoteses.Spotify`
GROUP BY
  artist_s__name;
```

```sql
 # JOIN para unir as tres tabelas spotify, competition e technical_info e criar a
Tab_Gravadora.
  SELECT
    *
  FROM
    `projeto--2.Hipoteses.Spotify` AS spotify
JOIN
    `projeto--2.Hipoteses.Technical_info` AS technical_info
  ON
    spotify.track_id = technical_info.track_id
JOIN
    `projeto--2.Hipoteses.Competition` AS competition
  ON
    spotify.track_id = competition.track_id;



#Limpeza dos dados da Tab_Gravadora e criacao da tabela Tab_Gravadora1.
WITH Gravadora AS (
  SELECT
    *,
    DATE(released_year, released_month, released_day) AS released_date,
    CASE
      WHEN track_id = '0:00' THEN '1001427'
      ELSE track_id
    END AS track_id_limpo,
    UPPER(mode) AS mode_maiusculo,
    UPPER(REGEXP_REPLACE(track_name, '[^a-zA-Z0-9 ]', '')) AS track_name_limpo,
    UPPER(REGEXP_REPLACE(artist_s__name, '[^a-zA-Z0-9 ]', '')) AS artist_name_limpo,
    SAFE_CAST(streams AS INT64) AS streams_limpo,
    (in_spotify_playlists + in_apple_playlists + in_deezer_playlists) AS
total_playlists,
    (IFNULL(in_apple_charts, 0) + IFNULL(in_deezer_charts, 0) +
IFNULL(in_shazam_charts, 0) + IFNULL(in_spotify_charts, 0)) / 4.0 AS
media_total_charts
  FROM
    `projeto--2.Hipoteses.Tab_Gravadora`
  WHERE
    in_shazam_charts IS NOT NULL
```

```
)
SELECT *
FROM Gravadora;


#Alterar o valor nulo da coluna stream_limpo para a media na Tab_Gravadora1 e criacao
da tabela Tab_Gravadora2.
SELECT *,
      CASE
      WHEN streams_limpo IS NULL THEN 514137424
      ELSE streams_limpo
    END AS streams_corrigido
FROM `projeto--2.Hipoteses.Tab_Gravadora1`


#WITH para criar tabela temporária para calcular o numero de musica de cada artista da
Tab_Gravadora2 e criacao da tabela Tab_Gravadora3.
WITH n_musicas AS (
SELECT
  artist_name_limpo,
  COUNT(*) AS n_musicas
FROM `projeto--2.Hipoteses.Tab_Gravadora2`
GROUP BY
 artist_name_limpo
)
SELECT *
FROM `projeto--2.Hipoteses.Tab_Gravadora2` AS Gravadora
RIGHT JOIN n_musicas
ON Gravadora.artist_name_limpo = n_musicas.artist_name_limpo;


#EXCEPT para limpeza da colunas repetidas da Tab_Gravadora3 e criacao da tabela
Tab_Gravadora_Final.
SELECT * EXCEPT (key, track_id_1,track_id_2, track_id, track_name, artist_s__name,
streams, mode, streams_limpo, artist_name_limpo_1)
FROM `projeto--2.Hipoteses.Tab_Gravadora3`;



#Quartil e Segmentação
CREATE OR REPLACE TABLE `projeto--2.Hipoteses.Tab_Gravadora_Final` AS
```

```sql
WITH Quartil AS (
  SELECT
    streams_corrigido, bpm, danceability__, valence__, energy__, acousticness__,
instrumentalness__, liveness__, speechiness__,
    NTILE(4) OVER (ORDER BY streams_corrigido) AS quartil_streams,
    NTILE(4) OVER (ORDER BY bpm) AS quartil_bpm,
    NTILE(4) OVER (ORDER BY danceability__) AS quartil_danceability,
    NTILE(4) OVER (ORDER BY valence__) AS quartil_valence,
    NTILE(4) OVER (ORDER BY energy__) AS quartil_energy,
    NTILE(4) OVER (ORDER BY acousticness__) AS quartil_acousticness,
    NTILE(4) OVER (ORDER BY instrumentalness__) AS quartil_instrumentalness,
    NTILE(4) OVER (ORDER BY liveness__) AS quartil_liveness,
    NTILE(4) OVER (ORDER BY speechiness__) AS quartil_speechiness
  FROM `projeto--2.Hipoteses.Tab_Gravadora_Final` AS gravadora
)
SELECT
  gravadora.*,
  Quartil.quartil_streams,
  Quartil.quartil_bpm,
  Quartil.quartil_danceability,
  Quartil.quartil_valence,
  Quartil.quartil_energy,
  Quartil.quartil_acousticness,
  Quartil.quartil_instrumentalness,
  Quartil.quartil_liveness,
  Quartil.quartil_speechiness,
  CASE
    WHEN quartil_streams = 1 OR quartil_streams = 2 THEN 'Baixo'
    WHEN quartil_streams = 3 OR quartil_streams = 4 THEN 'Alto'
  END AS segmentacao_streams,
  CASE
    WHEN quartil_bpm = 1 OR quartil_bpm = 2 THEN 'Baixo'
    WHEN quartil_bpm = 3 OR quartil_bpm = 4 THEN 'Alto'
  END AS segmentacao_bpm,
  CASE
    WHEN quartil_danceability = 1 OR quartil_danceability = 2 THEN 'Baixo'
    WHEN quartil_danceability = 3 OR quartil_danceability = 4 THEN 'Alto'
  END AS segmentacao_danceability,
  CASE
    WHEN quartil_valence = 1 OR quartil_valence = 2 THEN 'Baixo'
    WHEN quartil_valence = 3 OR quartil_valence = 4 THEN 'Alto'
```

```sql
  END AS segmentacao_valence,
  CASE
    WHEN quartil_energy = 1 OR quartil_energy = 2 THEN 'Baixo'
    WHEN quartil_energy = 3 OR quartil_energy = 4 THEN 'Alto'
  END AS segmentacao_energy,
  CASE
    WHEN quartil_acousticness = 1 OR quartil_acousticness = 2 THEN 'Baixo'
    WHEN quartil_acousticness = 3 OR quartil_acousticness = 4 THEN 'Alto'
  END AS segmentacao_acousticness,
  CASE
    WHEN quartil_instrumentalness = 1 OR quartil_instrumentalness = 2 THEN 'Baixo'
    WHEN quartil_instrumentalness = 3 OR quartil_instrumentalness = 4 THEN 'Alto'
  END AS segmentacao_instrumentalness,
  CASE
    WHEN quartil_liveness = 1 OR quartil_liveness = 2 THEN 'Baixo'
    WHEN quartil_liveness = 3 OR quartil_liveness = 4 THEN 'Alto'
  END AS segmentacao_liveness,
  CASE
    WHEN quartil_speechiness = 1 OR quartil_speechiness = 2 THEN 'Baixo'
    WHEN quartil_speechiness = 3 OR quartil_speechiness = 4 THEN 'Alto'
  END AS segmentacao_speechiness
FROM
  `projeto--2.Hipoteses.Tab_Gravadora_Final` AS gravadora
LEFT JOIN
  Quartil ON gravadora.streams_corrigido = Quartil.streams_corrigido
  AND gravadora.bpm = Quartil.bpm
  AND gravadora.danceability__ = Quartil.danceability__
  AND gravadora.valence__ = Quartil.valence__
  AND gravadora.energy__ = Quartil.energy__
  AND gravadora.acousticness__ = Quartil.acousticness__
  AND gravadora.instrumentalness__ = Quartil.instrumentalness__
  AND gravadora.liveness__ = Quartil.liveness__
  AND gravadora.speechiness__ = Quartil.speechiness__;



#Correlação Hipótese 1, 2, 3 e 5.
SELECT
CORR (streams_corrigido, bpm) AS corre_streams_bpm,
CORR (in_spotify_charts, in_deezer_charts) AS corre_charts_DeeSpot,
CORR (streams_corrigido, total_playlists) AS corre_streams_playlists,
```

```sql
CORR (streams_corrigido, danceability__) AS corre_streams_dancea,
CORR (streams_corrigido, valence__) AS corre_streams_valen,
CORR (streams_corrigido, energy__) AS corre_streams_energ,
CORR (streams_corrigido, acousticness__) AS corre_streams_acoust,
CORR (streams_corrigido, instrumentalness__) AS corre_streams_instru,
CORR (streams_corrigido, liveness__) AS corre_streams_liven,
CORR (streams_corrigido, speechiness__) AS corre_streams_speec
FROM `projeto--2.Hipoteses.Tab_Gravadora_Final`;


# verificar a quantidade de músicas - criei a tabela TAB_Gravadora_N_Musica
  SELECT
    artist_name_limpo,
    COUNT(DISTINCT track_name_limpo) AS total_musicas,
    SUM(streams_corrigido) AS total_streams
  FROM
    `projeto--2.Hipoteses.Tab_Gravadora_Final`
  GROUP BY
    artist_name_limpo


#Correlação Hipótese 4
SELECT
CORR (total_musicas, total_streams) AS corre_streams_total_musicas,
FROM `projeto--2.Hipoteses.Tab_Gravadora_N_Musicas`;
```