# SQL DATA ANALYST PROJECT - SPOTIFY

CREATE TABLE spotify\_tb(Artist VARCHAR (255),

Track VARCHAR(255),

Album VARCHAR (255),

Album\_type VARCHAR (15),

Danceability FLOAT,

Energy FLOAT,

Loudness FLOAT,

Speechiness FLOAT,

Acousticness FLOAT,

Instrumentalness FLOAT,

Liveness FLOAT,

Valence FLOAT,

Tempo FLOAT,

Duration\_min FLOAT,

Title VARCHAR (200),

Channel VARCHAR (60),

Views FLOAT,

Likes BIGINT,

Comments BIGINT,

Licensed BOOLEAN,

official\_video BOOLEAN,

Stream BIGINT,

EnergyLiveness BIGINT,

most\_playedon VARCHAR (10)

)

ALTER TABLE spotify\_tb

ALTER COLUMN EnergyLiveness TYPE FLOAT USING EnergyLiveness::FLOAT;

ALTER TABLE spotify\_tb

ALTER COLUMN Likes TYPE FLOAT USING Likes::FLOAT;

SELECT \* FROM spotify\_tb;

-- EDA

-- Total no. of rows

SELECT COUNT(\*) FROM spotify\_tb

-- Total no. of artists

SELECT COUNT(DISTINCT artist) FROM spotify\_tb

-- Total no. of albums

SELECT COUNT(DISTINCT album) FROM spotify\_tb

-- Different types of album

SELECT DISTINCT album\_type FROM spotify\_tb

-- Maximum duration of a song

SELECT MAX(duration\_min) FROM spotify\_tb

-- Minimum duration of a song

SELECT MIN(duration\_min) FROM spotify\_tb

SELECT \* FROM spotify\_tb

WHERE duration\_min = 0;

DELETE FROM spotify\_tb

WHERE duration\_min = 0;

SELECT \* FROM spotify\_tb

WHERE duration\_min = 0;

-- Different types of channel

SELECT DISTINCT channel

FROM spotify\_tb;

-----------------------------------------------------------------------------------------------------------------------

-- 1.Retrieve the names of all tracks that have more than 1 billion streams.

SELECT \* FROM spotify\_tb

WHERE stream > 1000000000;

-- 2.List all albums along with their respective artists.

SELECT DISTINCT album, artist

FROM spotify\_tb;

--3. Get the total number of comments for tracks where licensed = TRUE

SELECT SUM(comments) AS total\_comments FROM spotify\_tb

WHERE licensed = 'true'

-- 4.Find all tracks that belong to the album type single.

SELECT \* FROM spotify\_tb

WHERE album\_type = 'single'

-- 5.Count the total number of tracks by each artist.

SELECT artist, COUNT(track) AS total\_tracks FROM spotify\_tb

GROUP BY 1

ORDER BY 2 DESC;

--------------------------------------------------------------------------------------------------------------------------

-- 6.Calculate the average danceability of tracks in each album.

SELECT album, AVG(danceability) AS avg\_danceability FROM spotify\_tb

GROUP BY 1

ORDER BY 2 DESC;

-- 7.Find the top 5 tracks with the highest energy values.

SELECT track, AVG(energy) FROM spotify\_tb

Group by 1

ORDER BY 2

LIMIT 5

-- 8. List all tracks along with their views and likes where official\_video = TRUE.

SELECT track, SUM(views) AS total\_views, SUM(likes) AS total\_likes FROM spotify\_tb

WHERE official\_video ='true'

GROUP BY 1

ORDER BY 2 DESC;

-- 9.For each album, calculate the total views of all associated tracks.

SELECT \* FROM spotify\_tb

SELECT album, track,

SUM(views) AS total\_views

FROM spotify\_tb

GROUP BY 1,2

ORDER BY 3 DESC;

-- 10.Retrieve the track names that have been streamed on Spotify more than YouTube.

SELECT \* FROM (SELECT track,

COALESCE (SUM(CASE WHEN most\_playedon = 'Youtube' THEN stream END), 0) AS streamed\_on\_youtube,

COALESCE (SUM(CASE WHEN most\_playedon = 'Spotify' THEN stream END), 0) AS streamed\_on\_spotify

FROM spotify\_tb

GROUP BY 1) AS t1

WHERE streamed\_on\_spotify > streamed\_on\_youtube

AND streamed\_on\_youtube <> 0

--------------------------------------------------------------------------------------------------------------------------

-- 11. Find the top 3 most-viewed tracks for each artist using window functions.

with ranking\_artist

AS

(SELECT artist, track, SUM(views) AS total\_views,

DENSE\_RANK() OVER(PARTITION BY artist ORDER BY SUM(views) DESC) as rank

FROM spotify\_tb

GROUP BY 1, 2

order by 1,3 desc)

SELECT \* FROM ranking\_artist

WHERE rank <= 3

-- 12. Write a query to find tracks where the liveness score is above the average.

SELECT track, artist, album FROM spotify\_tb

WHERE liveness > (SELECT AVG(liveness) FROM spotify\_tb)

-- 13. Use a WITH clause to calculate the difference between the highest and lowest energy values for tracks in each album.

WITH cte

AS

(SELECT

album,

MAX(energy) as highest\_energy,

MIN(energy) as lowest\_energy

FROM spotify\_tb

GROUP BY 1

)

SELECT

album,

highest\_energy - lowest\_energery as energy\_diff

FROM cte

ORDER BY 2 DESC

--14. Find tracks where the energy-to-liveness ratio is greater than 1.2.

SELECT track FROM spotify\_tb

WHERE energy/liveness > 1.2

-- 15. Calculate the cumulative sum of likes for tracks ordered by the number of views, using window functions.

SELECT

track,

views,

likes,

SUM(likes) OVER (ORDER BY views DESC) AS cumulative\_likes

FROM spotify\_tb