Spotify Music Dataset Analysis using SQL

Overview:

This project involves analyzing a spotify dataset having various attributes about albums, tracks, streams, views, likes, artists and other components using SQL. It covers the end-to-end process of normalizing a denormalized dataset, performing SQL queries of varying complexity (easy, medium, and advanced), and optimizing query performance. The primary goal of the project is to practice advanced SQL skills and generate valuable insights from the dataset.

Project Steps

1. **Data Exploration**
   Before diving into SQL, it's important to understand the dataset thoroughly. It is said 80% is data exploration & understanding and 20% is data analysis. The dataset contains attributes such as:
   - Artist
   - Track
   - Album
   - Album Type
   - Duration
   - Platform
   - Various metrics such as danceability, liveness, energy, Loudness and more.

2. **Data cleaning**
   During data exploration, I found some inconsistency in the dataset which results in dropping rows for a more consistent and result-driven dataset.

3. **Querying the Data**
   After the data is cleaned, various SQL queries can be written to explore and analyze the data. Queries are categorized into easy, medium and advanced levels to help progressively develop SQL proficiency.

   Easy Queries
   - Simple data retrieval, filtering and basic aggregations.

   Medium Queries
   - More complex queries involving grouping, aggregation functions,

   Advanced Queries
   - Nested subqueries, window functions, CTEs, and performance optimization

15 Practice Questions

Easy Level

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

```sql
-- Q1. Retrieve the names of all tracks that have more than 1 billion streams.
Select Track, Stream
From project..spotify_data
Where Stream >= 1000000000
Order BY Stream DESC;
```

00 % Results Messages

	Track	Stream
1	Blinding Lights	3386520288
2	Shape of You	3362005201
3	Someone You Loved	2634013335
4	rockstar (feat. 21 Savage)	2594926619
5	Sunflower - Spider-Man: Into the Spider-Verse	2538329799
6	Sunflower - Spider-Man: Into the Spider-Verse	2538329799

2. List all albums along with their respective artists.

```sql
-- Q2. List all albums along with their respective artists.
Select Distinct Album, Artist
from project..spotify_data;
```

sults Messages

Album	Artist
- TRAGEDY +	$NOT
!Volare! The Very Best of the Gipsy Kings	Gipsy Kings
"Awaken, My Love!"	Childish Gambino
"Heroes" (2017 Remaster)	David Bowie
"Let Go" Dj Pack	Dina Rae
"Let's Rock"	The Black Keys

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

```sql
-- Q3. Get the total number of comments for tracks where licensed = TRUE.
select Track, sum(Comments) as total_comment
From project..spotify_data
Where Licensed = 1
Group by Track;
```

Results | Messages

Track	total_comment
!ly (feat. Coez)	1628
#1 - Colby O'Donis Remix	12359
#41	1248
#NAME?	70172
#thatPOWER	301321
#Ysya2020 Vol. 5 - Silbando	521

4. Find all tracks that belong to the album type `single`.
5. Count the total number of tracks by each artist.

```sql
-- Q5. Count the total number of tracks by each artist.
select Artist, COUNT(Track) as total_tracks
from project..spotify_data
Group by Artist
Order by total_tracks;
```

Results | Messages

Artist	total_tracks
back number	1
Bootie Brown	1
Stars Music Chile	1
Mrs. GREEN APPLE	2
Vaundy	2
Jimin	3

Medium Level

1. Calculate the average danceability of tracks in each album.

```sql
-- Q1. Calculate the average danceability of tracks in each album.
Select Album,
            Track,
            Avg(Danceability) as Avg_Danceabiltiy
From project..spotify_data
Group By Album,Track
Order by Avg_Danceabiltiy DESC;
```

Results | Messages

Album	Track	Avg_Danceabiltiy
FOR CERTAIN (Deluxe)	CAN'T TOUCH THIS	0.975
Funky Friday	Funky Friday	0.975
Shock Value	Give It To Me	0.975
Blood Shore Season 2	Psycho Pass	0.973
The House Is Burning [homies begged]	RIP Young (feat. Project Pat & Juicy J) - Remix	0.971
N9NA	Like I Ain't	0.97

2. Find the top 5 tracks with the highest energy values.

```sql
-- Q2. Find the top 5 tracks with the highest energy values.
    Select Top 5 Track,
        Max(Energy) as Energy_level
    From project..spotify_data
    Group by Track
    Order by Energy_level DESC;
```

Results | Messages

Track	Energy_level
Gentle Piano Melodies	1
Rain and Thunderstorm, Pt. 3	1
Rain and Thunderstorm, Pt. 33	1
Rain and Thunderstorm, Pt. 4	1
Rain and Thunderstorm, Pt. 6	1

3. List all tracks along with their views and likes where `official_video = TRUE`.
4. For each album, calculate the total views of all associated tracks.

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

```
-- Q5.Retrieve the track names that have been streamed on Spotify more than YouTube.

Select *
From (
Select Track,
        -- most_playedon,
        coalesce(sum(Case when most_playedon = 'Youtube' Then Stream End),0) as youtube_stream,
        Coalesce(sum(Case when most_playedon = 'Spotify' Then Stream End),0) as spotify_stream
From project..spotify_data
Group by Track) As track_name
Where spotify_stream > youtube_stream
        And youtube_stream <>0
```

esults | Messages

Track	youtube_stream	spotify_stream
(Don't Fear) The Reaper	21305089	441277003
21 Hungarian Dances, WoO 1: Hungarian Dance No....	39575743	79151486
23	52297195	62639183
3 A.M.	31174412	607872704
A las Nueve	53863032	107560421
abc	5288836	350213604

Advanced Level

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

```
-- Q1. Find the top 3 most-viewed tracks for each artist using window functions.
With most_view_tranks As(
Select *,
        Dense_Rank() over(Partition By Artist Order By total_views DESC) as view_rank
From (
Select Artist, Track, sum(Views) as total_views
From project..spotify_data
Group By Artist, Track
-- Order by Artist, total_views DESC
) As track_view
)
Select *
     --Artist, Track, total_views
From most_view_tranks
Where view_rank <=3;
```

esults | Messages

Artist	Track	total_views	view_rank
$NOT	Tell Em	41100657	1
$NOT	Like Me (feat. iann dior)	15803517	2
$NOT	Mean	13563870	3
$uicideboy$	Paris	175156959	1
$uicideboy$	For the Last Time	91771038	2
$uicideboy$	Kill Yourself (Part III)	31674988	3

2. Write a query to find tracks where the liveness score is above the average.
3. Use a `WITH` clause to calculate the difference between the highest and lowest energy values for tracks in each album.

```sql
-- Q3. Use a WITH clause to calculate the difference between the highest and lowest energy values for tracks in each album.

With energy_difference As(
Select Album,
        MAX(Energy) Max_Energy,
        MIN(Energy) As Min_Energy
From project..spotify_data
Group by Album
-- Order by Album
)
Select *,
    (Max_Energy - Min_Energy) as energy_diff
From energy_difference
Order By energy_diff DESC;
```

Messages

Album	Max_Energy	Min_Energy	energy_diff
White Noise	0.908	0.00125	0.90675
Spotify Singles - Holiday	0.904	0.068	0.836
Spotify Singles	0.894	0.0708	0.8232
UNDERTALE Soundtrack	0.978	0.162	0.816
Making Mirrors	0.9	0.0891	0.8109
Everytime We Touch (Premium Edition)	0.978	0.173	0.805

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

```sql
-- Q4. Find the tracks where the enegy-to-liveness ratio is greater than 1.2
Select *
From (
Select Track,
        Energy,
        Liveness,
        Round((Energy / Liveness),2) As Energy_Liveness_Ratio
From project..spotify_data
) As ratio
Where Energy_Liveness_Ratio > 1.2
Order By Energy_Liveness_Ratio;
```

Results Messages

Track	Energy	Liveness	Energy_Liveness_Ratio
Doruklara Sevdaland?m	0.522	0.431	1.21
Lust for a Vampyr	0.503	0.417	1.21
Vai Errar de Novo - Ao Vivo	0.856	0.71	1.21
Separate Ways (Worlds Apart)	0.892	0.74	1.21
Hurts Like Hell	0.129	0.107	1.21
Mas Macarena (feat. Los Del Rio)	0.897	0.741	1.21

5.

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

```
-- Q4. Calculate the cumulative sum of likes for track ordered by the number of views, using window function

Select *,
        SUM(Likes) over(Order By Views) as cumulative_likes
From (
    Select Track, Likes, Views
    From project..spotify_data
        ) As a
Where Likes> 0;
```

Results | Messages

Track	Likes	Views	cumulative_likes
When a Blind Man Cries - B-Side; 2012 Remaster	1	31	1
ResistirÃ©	9	49	10
Let Go - Acapella	1	56	11
Fallen Angel - Remastered	5	61	16
Thozhiya (From "Kadhalil Vizhundhen")	2	71	18
Winterreise, Op. 89, D. 911 (arr. T. Zimmermann ...	1	90	19