

ASSIGNMENT 5 - SPOTIFY DATA ANALYSIS

Introduction

Spotify is a digital music streaming service that provides users access to over 82 million songs, podcasts and audio books. The app was developed by Daniel Ek and Martin Lorenzton in 2006. This app has become a family name over the years and boasts over 457 million subscribers as of 2022, rivaling SoundCloud and Apple Music.

Spotify measures the popularity of its' artists based on their monthly listeners and number of streams they receive on songs produced. These streams are then multiplied by (0.003) and paid to artists as "Royalties", it is a modernized system of monetizing digital sales from traditional album sales (100 streams = 1 album). Ed Sheeran was Spotify's most streamed artist in 2019, however, the rank placements change rapidly depending on album releases, EP's, mixtapes and so forth!

Spotify is a perfect dataset to measure the popularity of songs against various music elements, across a large set of songs throughout the decades. This analysis can be used to demonstrate how peoples music tastes have been translated throughout the past two decades!

I will be creating an exploratory analysis by creating data visualizations and conducting statistical analyses to investigate the relationship between the use of non-traditional musical elements and the popularity of Spotify hits from 2000 to 2019.

Track Metadata

Column	Description
Track_Name	Song title
Artist_Name	Song Artist
Artist_Genre	Song Genre Category
Year	Song Billboard chart entry year

Audio Numerical Quantitive Data

Column	Description
Loudness	How loud a song is (db)
Duration_MS	How long the song is (seconds)
Tempo	How fast a song is (bpm)

Audio Qualitative Data

Column	Description
Energy	How energetic the song is
Dance_Ability	How easy it is to dance to
Valence	How positive the mood of the song is
Acousticness	How acoustic sounding the song is
Speechiness	How much of a song is spoken word
Track_Popularity	How popular a song is (as of time of data collection)

Table Structure:

Create a table named PLAYLIST with the following structure:

```
CREATE OR REPLACE TABLE PLAYLIST
(
  PLAYLIST_URL VARCHAR(100),
  YEAR_NO INT,
  TRACK_ID VARCHAR(50),
  TRACK_NAME VARCHAR(100),
  TRACK_POPULARITY INT,
  ALBUM VARCHAR(100),
  ARTIST_ID VARCHAR(30) ,
  ARTIST_NAME VARCHAR(50),
  ARTIST_GENRES VARCHAR(200),
  ARTIST_POPULARITY INT,
  DANCE_ABILITY DECIMAL(5,3),
  ENERGY DECIMAL(6,4),
  KEY_ID TINYINT,
  LOUDNESS DECIMAL(6,4),
  MODE_BIT TINYINT,
  SPEECHINESS DECIMAL(6,4),
  ACOUSTICNESS DECIMAL(10,8),
  INSTRUMENTALNESS DECIMAL(15,10),
  LIVENESS DECIMAL(6,4),
  VALENCE DECIMAL(6,4),
  TEMPO DECIMAL(7,4),
  DURATION_MS INT,
  TIME_SIGNATURE TINYINT,
  PRIMARY KEY (TRACK_ID, ARTIST_ID)
);
```

Task :

1. Check the entire dataset

SELECT * FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST;

36 -- 1. Check the entire dataset
37
38 SELECT * FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST;
39

Results

Chart

	PLAYLIST_URL	YEAR_NO	TRACK_ID	TRACK_NAME	TRACK_POPULARITY	ALBUM	...	ARTIST_ID
1	https://open.spotify.com/pl	2,000	3AJwUDP919kvQ9QcozQPXg	Yellow	91	Parachutes		4gzpq5DPGxSnKTe4SA8HAU
2	https://open.spotify.com/pl	2,000	2m1hi0nfMR9vdGC8UcrnwU	All The Small Things	84	Enema Of The State		6FBDaR13swtiWwGhX1WQsP
3	https://open.spotify.com/pl	2,000	3y4LxiYMgDI4RethdzpmNe	Breathe	69	Breathe		25NQNrVT2YbSW80ILRWJa
4	https://open.spotify.com/pl	2,000	60a0Rd6prkxjPbaKzXjfq	In the End	88	Hybrid Theory (Bonus Edition)		6XyY86QOPPrYVGvF9ch6wz
5	https://open.spotify.com/pl	2,000	62bOmKYxYg7dhrC6gH9vFn	Bye Bye Bye	74	No Strings Attached		6Ff53KvcvAj5U7Z1vojB5o

2. Number of songs on Spotify for each artist

SELECT ARTIST_NAME,
COUNT(TRACK_ID) AS TOT_SONGS
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
GROUP BY 1
ORDER BY 2 DESC;

40 -- 2. Number of songs on Spotify for each artist
41
42 SELECT ARTIST_NAME,
43 COUNT(TRACK_ID) AS TOT_SONGS
44 FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
45 GROUP BY 1
46 ORDER BY 2 DESC;
47

Results

Chart

	ARTIST_NAME	TOT_SONGS
1	Drake	32
2	Taylor Swift	31
3	Rihanna	27
4	Ariana Grande	22
5	BeyoncÃ©	22
6	Justin Bieber	21

3. Top 10 songs based on popularity

```
SELECT TRACK_NAME AS SONGS
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
ORDER BY TRACK_POPULARITY DESC
LIMIT 10;
```

```
48 -- 3. Top 10 songs based on popularity
49
50 SELECT TRACK_NAME AS SONGS
51 FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
52 ORDER BY TRACK_POPULARITY DESC
53 LIMIT 10;
54
```

Results Chart

	SONGS
1	Cruel Summer
2	august
3	I'm Good (Blue)
4	Anti-Hero
5	Starboy
6	I Ain't Worried
7	Blinding Lights
8	Calm Down (with Selena Gomez)
9	Sweater Weather
10	Yellow

4. Total number of songs on spotify based on year

```
SELECT YEAR_NO AS YEAR,
COUNT(TRACK_ID) AS TOT_SONGS
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
GROUP BY 1
ORDER BY 1;
```

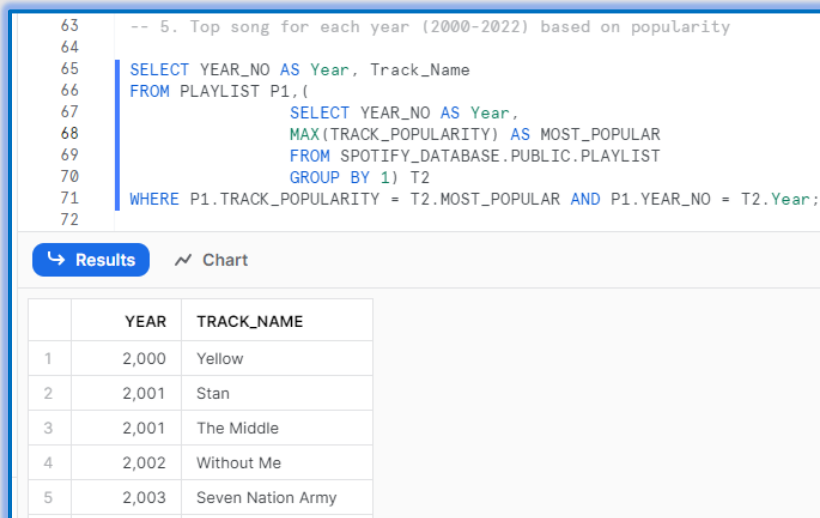
```
55 -- 4. Total number of songs on spotify based on year
56
57 SELECT YEAR_NO AS YEAR,
58 COUNT(TRACK_ID) AS TOT_SONGS
59 FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
60 GROUP BY 1
61 ORDER BY 1;
62
```

Results Chart

	YEAR	TOT_SONGS
1	2,000	100
2	2,001	100
3	2,002	100
4	2,003	100
5	2,004	100

5. Top song for each year (2000-2022) based on popularity

```
SELECT YEAR_NO AS Year, Track_Name
FROM PLAYLIST P1,(
    SELECT YEAR_NO AS Year,
    MAX(TRACK_POPULARITY) AS MOST_POPULAR
    FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
    GROUP BY 1) T2
WHERE P1.TRACK_POPULARITY = T2.MOST_POPULAR AND P1.YEAR_NO = T2.Year;
```



The screenshot shows a SQL query editor with a query to find the top song for each year from 2000 to 2022 based on popularity. The query is as follows:

```
-- 5. Top song for each year (2000-2022) based on popularity
SELECT YEAR_NO AS Year, Track_Name
FROM PLAYLIST P1,(
    SELECT YEAR_NO AS Year,
    MAX(TRACK_POPULARITY) AS MOST_POPULAR
    FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
    GROUP BY 1) T2
WHERE P1.TRACK_POPULARITY = T2.MOST_POPULAR AND P1.YEAR_NO = T2.Year;
```

Below the query editor, there is a 'Results' tab showing the following data:

	YEAR	TRACK_NAME
1	2,000	Yellow
2	2,001	Stan
3	2,001	The Middle
4	2,002	Without Me
5	2,003	Seven Nation Army

6. Analysis based on Tempo :

tempo > 121.08 -> 'Above Average Tempo'

tempo = 121.08 -> 'Average Tempo'

tempo < 121.08 -> 'Below Average Tempo'

Note:

I have created a View here so that I can use this view to answer other queries related to this analysis.

```
CREATE OR REPLACE VIEW PLAYLIST_TEMPO_ANALYSIS_VIEW AS
SELECT TRACK_NAME, ENERGY, TEMPO,
CASE
    WHEN TEMPO > 121.08 THEN 'Above Average Tempo'
    WHEN TEMPO = 121.08 THEN 'Average Tempo'
    WHEN TEMPO < 121.08 THEN 'Below Average Tempo'
END AS TEMPO_CATEGORY
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE TEMPO IS NOT NULL;
```

```
SELECT * FROM Playlist_Tempo_Analysis_View ;
```

```

74 -- 6. Analysis based on Tempo :
75 -- tempo > 121.08 -> 'Above Average Tempo'
76 -- tempo = 121.08 -> 'Average Tempo'
77 -- tempo < 121.08 -> 'Below Average Tempo'
78
79 CREATE OR REPLACE VIEW PLAYLIST_TEMPO_ANALYSIS_VIEW AS
80 SELECT TRACK_NAME, ENERGY, TEMPO,
81 CASE
82     WHEN TEMPO > 121.08 THEN 'Above Average Tempo'
83     WHEN TEMPO = 121.08 THEN 'Average Tempo'
84     WHEN TEMPO < 121.08 THEN 'Below Average Tempo'
85 END AS TEMPO_CATEGORY
86 FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
87 WHERE TEMPO IS NOT NULL;
88
89 | SELECT * FROM Playlist_Tempo_Analysis_View ;
90

```

Results Chart

	TRACK_NAME	ALBUM	ARTIST_NAME	ARTIST_GENRES	...	TEMPO	TEMPO_CATEGORY
1	Yellow	Parachutes	Coldplay	['permanent wave', 'pop']		173.3720	Above Average Tempo
2	All The Small Things	Enema Of The State	blink-182	['alternative metal', 'modern rock', 'pop punk', 'punk', 'rock', 'social pop punk']		148.7260	Above Average Tempo
3	Breathe	Breathe	Faith Hill	['contemporary country', 'country', 'country dawn', 'country road']		136.8590	Above Average Tempo
4	In the End	Hybrid Theory (Bonus Edition)	Linkin Park	['alternative metal', 'nu metal', 'post-grunge', 'rap metal', 'rock']		105.1430	Below Average Tempo
5	Bye Bye Bye	No Strings Attached	*NSYNC	['boy band', 'dance pop', 'pop']		172.6380	Above Average Tempo

7. Songs with Highest Tempo

```

SELECT TRACK_NAME, TEMPO, TEMPO_CATEGORY
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST_TEMPO_ANALYSIS_VIEW
ORDER BY 2 DESC
LIMIT 1;

```

```

92 -- 7. Songs with Highest Tempo
93
94 | SELECT TRACK_NAME, TEMPO, TEMPO_CATEGORY
95 | FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST_TEMPO_ANALYSIS_VIEW
96 | ORDER BY 2 DESC
97 | LIMIT 1;
98

```

Results Chart

	TRACK_NAME	TEMPO	TEMPO_CATEGORY
1	Buttons	210.8570	Above Average Tempo

8. Number of Songs for different Tempo Range : track_name, energy

Modern_Music -> tempo BETWEEN 60.00 AND 100.00

Classical_Music -> tempo BETWEEN 100.001 AND 120.00

Dance_Music -> tempo BETWEEN 120.001 AND 150.01

HighTempo_Music -> tempo > 150.01

Note:

I have created a View here so that I can use this view to answer other queries related to this analysis.

```
CREATE OR REPLACE VIEW PLAYLIST_TEMPO_ANALYSIS_VIEW_2 AS
SELECT TRACK_NAME, ENERGY, TEMPO,
CASE
    WHEN TEMPO BETWEEN 60.00 AND 100.00 THEN 'Modern_Music'
    WHEN TEMPO BETWEEN 100.001 AND 120.00 THEN 'Classical_Music'
    WHEN TEMPO BETWEEN 120.001 AND 150.01 THEN 'Dance_Music'
    WHEN TEMPO > 150.01 THEN 'HighTempo_Music'
END AS Music_Type
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE TEMPO IS NOT NULL;
```

```
SELECT * FROM PLAYLIST_TEMPO_ANALYSIS_VIEW_2;
```

```
99  -- 8. Number of Songs for different Tempo Range : track_name, energy
100 /*
101     Modern_Music -> tempo BETWEEN 60.00 AND 100.00
102     Classical_Music -> tempo BETWEEN 100.001 AND 120.00
103     Dance_Music -> tempo BETWEEN 120.001 AND 150.01
104     HighTempo_Music -> tempo > 150.01 */
105
106 CREATE OR REPLACE VIEW PLAYLIST_TEMPO_ANALYSIS_VIEW_2 AS
107 SELECT TRACK_NAME, ENERGY, TEMPO,
108 CASE
109     WHEN TEMPO BETWEEN 60.00 AND 100.00 THEN 'Modern_Music'
110     WHEN TEMPO BETWEEN 100.001 AND 120.00 THEN 'Classical_Music'
111     WHEN TEMPO BETWEEN 120.001 AND 150.01 THEN 'Dance_Music'
112     WHEN TEMPO > 150.01 THEN 'HighTempo_Music'
113 END AS Music_Type
114 FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
115 WHERE TEMPO IS NOT NULL;
116 SELECT * FROM PLAYLIST_TEMPO_ANALYSIS_VIEW_2;
```

Results Chart

	TRACK_NAME	ENERGY	TEMPO	MUSIC_TYPE
1	Yellow	0.6610	173.3720	HighTempo_Music
2	All The Small Things	0.8970	148.7260	Dance_Music
3	Breathe	0.4960	136.8590	Dance_Music
4	In the End	0.8640	105.1430	Classical_Music
5	Bye Bye Bye	0.9260	172.6380	HighTempo_Music

8.1 Number of songs of different TEMPO range

```
SELECT Music_Type, COUNT(TRACK_NAME) AS Tot_Songs
FROM PLAYLIST_TEMPO_ANALYSIS_VIEW_2
GROUP BY 1
ORDER BY 2 DESC;
```

```
118 -- 8.1 Number of songs of different TEMPO range
119
120 SELECT Music_Type, COUNT(TRACK_NAME) AS Tot_Songs
121 FROM PLAYLIST_TEMPO_ANALYSIS_VIEW_2
122 GROUP BY 1
123 ORDER BY 2 DESC;
124
```

Results Chart

	MUSIC_TYPE	TOT_SONGS
1	Dance_Music	824
2	Modern_Music	636
3	Classical_Music	514
4	HighTempo_Music	325

9. Energy Analysis : TOP 10 track_name, danceability, track_popularity

energy > 0.64 -> 'Above Average Energy'

energy = 0.64 -> 'Average Energy'

energy < 0.64 -> 'Below Average Energy'

energy BETWEEN 0.1 AND 0.3 -> 'Calm Music'

energy BETWEEN 0.3 AND 0.6 -> 'Moderate Music'

Energy > 0.6 -> 'Energetic Music'

Note:

- In this question, there is given 2 types of range on Energy so I have divided this question in 2 parts.
- I have created a View here so that I can use this view to answer other queries related to this analysis.

9.1 Energy Analysis : TOP 10 track_name, danceability, track_popularity

energy > 0.64 -> 'Above Average Energy'

energy = 0.64 -> 'Average Energy'

energy < 0.64 -> 'Below Average Energy'

```
CREATE OR REPLACE VIEW PLAYLIST_ENERGY_ANALYSIS_VIEW AS
SELECT TRACK_NAME, DANCE_ABILITY, TRACK_POPULARITY, ENERGY,
CASE
  WHEN ENERGY > 0.64 THEN 'Above Average Energy'
  WHEN ENERGY = 0.64 THEN 'Average Energy'
  WHEN ENERGY < 0.64 THEN 'Below Average Energy'
END AS Energy_Type
```



```
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE ENERGY IS NOT NULL;
```

```
SELECT * FROM PLAYLIST_ENERGY_ANALYSIS_VIEW;
```

```
126  /*
127  9.1 Energy Analysis : TOP 10 track_name, danceability, track_popularity
128      energy > 0.64 -> 'Above Average Energy'
129      energy = 0.64 -> 'Average Energy'
130      energy < 0.64 -> 'Below Average Energy'
131  */
132  CREATE OR REPLACE VIEW PLAYLIST_ENERGY_ANALYSIS_VIEW AS
133  SELECT TRACK_NAME, DANCE_ABILITY, TRACK_POPULARITY, ENERGY,
134  CASE
135      WHEN ENERGY > 0.64 THEN 'Above Average Energy'
136      WHEN ENERGY = 0.64 THEN 'Average Energy'
137      WHEN ENERGY < 0.64 THEN 'Below Average Energy'
138  END AS Energy_Type
139  FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
140  WHERE ENERGY IS NOT NULL;
141
142  SELECT * FROM PLAYLIST_ENERGY_ANALYSIS_VIEW;
143
```

Results Chart

	TRACK_NAME	DANCE_ABILITY	TRACK_POPULARITY	ENERGY	ENERGY_TYPE	...
1	Yellow	0.429	91	0.6610	Above Average Energy	
2	All The Small Things	0.434	84	0.8970	Above Average Energy	
3	Breathe	0.529	69	0.4960	Below Average Energy	
4	In the End	0.556	88	0.8640	Above Average Energy	
5	Bye Bye Bye	0.610	74	0.9260	Above Average Energy	

9.2 Energy Analysis : TOP 10 track_name, danceability, track_popularity

Energy BETWEEN 0.1 AND 0.3 -> 'Calm Music'
Energy BETWEEN 0.3 AND 0.6 -> 'Moderate Music'
Energy >0.6 -> 'Energetic Music'

```
CREATE OR REPLACE VIEW PLAYLIST_ENERGY_ANALYSIS_VIEW_2 AS
SELECT TRACK_NAME, DANCE_ABILITY, TRACK_POPULARITY, ENERGY,
CASE
    WHEN ENERGY BETWEEN 0.1 AND 0.3 THEN 'Calm Music'
    WHEN ENERGY BETWEEN 0.3 AND 0.6 THEN 'Moderate Music'
    WHEN ENERGY > 0.6 THEN 'Energetic Music'
    ELSE 'Others'
END AS Music_Type
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE ENERGY IS NOT NULL;
```

```
SELECT * FROM PLAYLIST_ENERGY_ANALYSIS_VIEW_2;
```

```

145  /*
146  9.2 Energy Analysis : TOP 10 track_name, danceability, track_popularity
147      energy BETWEEN 0.1 AND 0.3 -> 'Calm Music'
148      energy BETWEEN 0.3 AND 0.6 -> 'Moderate Music'
149      Energy >0.6 -> 'Energetic Music'
150  */
151
152  CREATE OR REPLACE VIEW PLAYLIST_ENERGY_ANALYSIS_VIEW_2 AS
153  SELECT TRACK_NAME, DANCE_ABILITY, TRACK_POPULARITY, ENERGY,
154  CASE
155      WHEN ENERGY BETWEEN 0.1 AND 0.3 THEN 'Calm Music'
156      WHEN ENERGY BETWEEN 0.3 AND 0.6 THEN 'Moderate Music'
157      WHEN ENERGY > 0.6 THEN 'Energetic Music'
158      ELSE 'Others'
159  END AS Music_Type
160  FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
161  WHERE ENERGY IS NOT NULL;
162
163  | SELECT * FROM PLAYLIST_ENERGY_ANALYSIS_VIEW_2;
164

```

Results Chart

	TRACK_NAME	DANCE_ABILITY	TRACK_POPULARITY	ENERGY	MUSIC_TYPE	...
1	Yellow	0.429	91	0.6610	Energetic Music	
2	All The Small Things	0.434	84	0.8970	Energetic Music	
3	Breathe	0.529	69	0.4960	Moderate Music	
4	In the End	0.556	88	0.8640	Energetic Music	
5	Bye Bye Bye	0.610	74	0.9260	Energetic Music	

10. Number of Songs for different energy ranges(above)

FOR 9.1

```

SELECT ENERGY_TYPE, COUNT(TRACK_NAME) AS TOT_SONGS
FROM PLAYLIST_ENERGY_ANALYSIS_VIEW
GROUP BY 1
ORDER BY 2 DESC;

```

```

166  /*
167  10. Number of Songs for different energy ranges(above)
168  */
169
170  SELECT ENERGY_TYPE, COUNT(TRACK_NAME) AS TOT_SONGS
171  FROM PLAYLIST_ENERGY_ANALYSIS_VIEW
172  GROUP BY 1
173  ORDER BY 2 DESC;

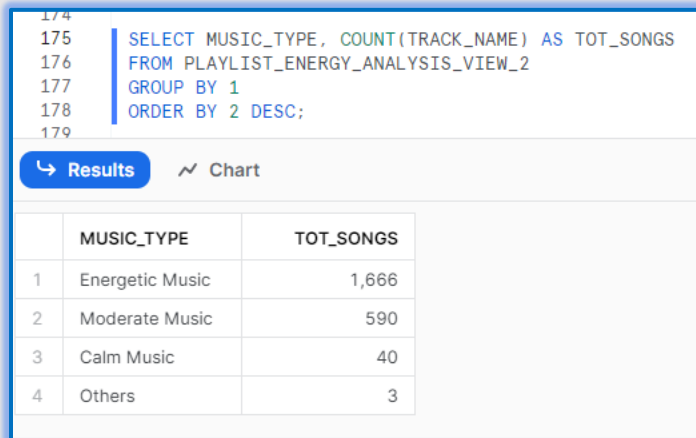
```

Results Chart

	ENERGY_TYPE	TOT_SONGS
1	Above Average Energy	1,502
2	Below Average Energy	796
3	Average Energy	1

FOR 9.2

```
SELECT MUSIC_TYPE, COUNT(TRACK_NAME) AS TOT_SONGS
FROM PLAYLIST_ENERGY_ANALYSIS_VIEW_2
GROUP BY 1
ORDER BY 2 DESC;
```



The screenshot shows a SQL query editor with a query window and a results window. The query window contains the following SQL code:

```
174
175 SELECT MUSIC_TYPE, COUNT(TRACK_NAME) AS TOT_SONGS
176 FROM PLAYLIST_ENERGY_ANALYSIS_VIEW_2
177 GROUP BY 1
178 ORDER BY 2 DESC;
179
```

The results window shows a table with the following data:

	MUSIC_TYPE	TOT_SONGS
1	Energetic Music	1,666
2	Moderate Music	590
3	Calm Music	40
4	Others	3

11. Danceability Analysis : Top 20 track_name, danceability

danceability BETWEEN 0.69 AND 0.79 -> 'Low Danceability'

(danceability BETWEEN 0.49 AND 0.68) OR (danceability BETWEEN 0.79 AND 0.89) ->

'Moderate Danceability'

(danceability BETWEEN 0.39 AND 0.49) OR (danceability BETWEEN 0.89 AND 0.99) ->

'High Danceability'

danceability < 0.39 OR danceability > 0.99 -> 'Cant Dance on this one'

Note:

I have created a View here so that I can use this view to answer other queries related to this analysis.

```
CREATE OR REPLACE VIEW PLAYLIST_DANCE_ABILITY_ANALYSIS_VIEW AS
```

```
SELECT TRACK_NAME, DANCE_ABILITY,
```

```
CASE
```

```
  WHEN DANCE_ABILITY BETWEEN 0.69 AND 0.79 THEN 'Low Danceability'
```

```
  WHEN (DANCE_ABILITY BETWEEN 0.49 AND 0.68) OR (DANCE_ABILITY BETWEEN 0.79 AND 0.89) THEN 'Moderate Danceability'
```

```
  WHEN (DANCE_ABILITY BETWEEN 0.39 AND 0.49) OR (DANCE_ABILITY BETWEEN 0.89 AND 0.99) THEN 'High Danceability'
```

```
  WHEN DANCE_ABILITY < 0.39 OR DANCE_ABILITY > 0.99 THEN 'Cant Dance on this one'
```

```
  ELSE 'Others'
```

```
END AS Dance_Ability_Category
```

```
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
```

```
WHERE DANCE_ABILITY IS NOT NULL;
```

SELECT * FROM PLAYLIST_DANCE_ABILITY_ANALYSIS_VIEW;

```
180  /*
181  11. Danceability Analysis : Top 20 track_name, danceability
182      danceability BETWEEN 0.69 AND 0.79 -> 'Low Danceability'
183      (danceability BETWEEN 0.49 AND 0.68) OR (danceability BETWEEN 0.79 AND 0.89) -> 'Moderate Danceability'
184      (danceability BETWEEN 0.39 AND 0.49) OR (danceability BETWEEN 0.89 AND 0.99) -> 'High Danceability'
185      danceability < 0.39 OR danceability > 0.99 -> 'Cant Dance on this one'
186  */
187
188  CREATE OR REPLACE VIEW PLAYLIST_DANCE_ABILITY_ANALYSIS_VIEW AS
189  SELECT TRACK_NAME, DANCE_ABILITY,
190  CASE
191      WHEN DANCE_ABILITY BETWEEN 0.69 AND 0.79 THEN 'Low Danceability'
192      WHEN (DANCE_ABILITY BETWEEN 0.49 AND 0.68) OR (DANCE_ABILITY BETWEEN 0.79 AND 0.89) THEN 'Moderate Danceability'
193      WHEN (DANCE_ABILITY BETWEEN 0.39 AND 0.49) OR (DANCE_ABILITY BETWEEN 0.89 AND 0.99) THEN 'High Danceability'
194      WHEN DANCE_ABILITY < 0.39 OR DANCE_ABILITY > 0.99 THEN 'Cant Dance on this one'
195      ELSE 'Others'
196  END AS Dance_Ability_Category
197  FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
198  WHERE DANCE_ABILITY IS NOT NULL;
199
200  SELECT * FROM PLAYLIST_DANCE_ABILITY_ANALYSIS_VIEW;
201
```

Results Chart

	TRACK_NAME	DANCE_ABILITY	DANCE_ABILITY_CATEGORY
1	Yellow	0.429	High Danceability
2	All The Small Things	0.434	High Danceability
3	Breathe	0.529	Moderate Danceability
4	In the End	0.556	Moderate Danceability
5	Bye Bye Bye	0.610	Moderate Danceability

12. Number of Songs for different danceability ranges(above)

SELECT DANCE_ABILITY_CATEGORY, COUNT(TRACK_NAME) AS TOT_SONGS
FROM PLAYLIST_DANCE_ABILITY_ANALYSIS_VIEW
GROUP BY 1
ORDER BY 2 DESC;

```
203  /*
204  12. Number of Songs for different danceability ranges(above)
205  */
206
207  SELECT DANCE_ABILITY_CATEGORY, COUNT(TRACK_NAME) AS TOT_SONGS
208  FROM PLAYLIST_DANCE_ABILITY_ANALYSIS_VIEW
209  GROUP BY 1
210  ORDER BY 2 DESC;
211
```

Results Chart

	DANCE_ABILITY_CATEGORY	TOT_SONGS
1	Moderate Danceability	1,267
2	Low Danceability	582
3	High Danceability	289
4	Cant Dance on this one	91
5	Others	70

13. Loudness Analysis : Top 20 track_name, loudness,
loudness BETWEEN -23.00 AND -15.00 ->'Low Loudness'
loudness BETWEEN -14.99 AND -6.00 -> 'Below Average Loudness'
loudness BETWEEN -5.99 AND -2.90 -> 'Above Average Loudness'
loudness BETWEEN -2.89 AND -1.00 -> 'Peak Loudness'

Note:

I have created a View here so that I can use this view to answer other queries related to this analysis.

```
CREATE OR REPLACE VIEW PLAYLIST_LOUDNESS_ANALYSIS_VIEW AS
SELECT TRACK_NAME, LOUDNESS,
CASE
    WHEN LOUDNESS BETWEEN -23.00 AND -15.00 THEN 'Low Loudness'
    WHEN LOUDNESS BETWEEN -14.99 AND -6.00 THEN 'Below Average Loudness'
    WHEN LOUDNESS BETWEEN -5.99 AND -2.90 THEN 'Above Average Loudness'
    WHEN LOUDNESS BETWEEN -2.89 AND -1.00 THEN 'Peak Loudness'
    ELSE 'Others'
END AS Loudness_Analysis
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE LOUDNESS IS NOT NULL;

SELECT * FROM PLAYLIST_LOUDNESS_ANALYSIS_VIEW;
```

```

213      /*
214      13. Loudness Analysis : Top 20 track_name, loudness,
215          loudness BETWEEN -23.00 AND -15.00 ->'Low Loudness'
216          loudness BETWEEN -14.99 AND -6.00 -> 'Below Average Loudness'
217          loudness BETWEEN -5.99 AND -2.90 -> 'Above Average Loudness'
218          loudness BETWEEN -2.89 AND -1.00 -> 'Peak Loudness'
219      */
220
221      CREATE OR REPLACE VIEW PLAYLIST_LOUDNESS_ANALYSIS_VIEW AS
222      SELECT TRACK_NAME, LOUDNESS,
223      CASE
224          WHEN LOUDNESS BETWEEN -23.00 AND -15.00 THEN 'Low Loudness'
225          WHEN LOUDNESS BETWEEN -14.99 AND -6.00 THEN 'Below Average Loudness'
226          WHEN LOUDNESS BETWEEN -5.99 AND -2.90 THEN 'Above Average Loudness'
227          WHEN LOUDNESS BETWEEN -2.89 AND -1.00 THEN 'Peak Loudness'
228          ELSE 'Others'
229      END AS Loudness_Analysis
230      FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
231      WHERE LOUDNESS IS NOT NULL;
232
233      SELECT * FROM PLAYLIST_LOUDNESS_ANALYSIS_VIEW;
234


```

Results
Chart

	TRACK_NAME	LOUDNESS	LOUDNESS_ANALYSIS
1	Yellow	-7.2270	Below Average Loudness
2	All The Small Things	-4.9180	Above Average Loudness
3	Breathe	-9.0070	Below Average Loudness
4	In the End	-5.8700	Above Average Loudness
5	Bye Bye Bye	-4.8430	Above Average Loudness

14. Number of Songs for different loudness ranges(above)

```
SELECT LOUDNESS_ANALYSIS, COUNT(TRACK_NAME) AS TOT_SONGS
FROM PLAYLIST_LOUDNESS_ANALYSIS_VIEW
GROUP BY 1
ORDER BY 2 DESC;
```



The screenshot shows a SQL query editor with a query window and a results window. The query window contains the following SQL code:

```
236  /*
237  14. Number of Songs for different Loudness ranges(above)
238  */
239
240  SELECT LOUDNESS_ANALYSIS, COUNT(TRACK_NAME) AS TOT_SONGS
241  FROM PLAYLIST_LOUDNESS_ANALYSIS_VIEW
242  GROUP BY 1
243  ORDER BY 2 DESC;
244
```

The results window shows a table with the following data:

	LOUDNESS_ANALYSIS	TOT_SONGS
1	Above Average Loudness	1,300
2	Below Average Loudness	907
3	Peak Loudness	84
4	Low Loudness	6
5	Others	2

15. Valence Analysis : Top 20 track_name, valence, track_popularity, valence > 0.535 -> Above Avg Valence valence = 0.535 -> Avg Valence valence < 0.535 -> Below Average'

Note:

I have created a View here so that I can use this view to answer other queries related to this analysis.

```
CREATE OR REPLACE VIEW PLAYLIST_VALENCE_ANALYSIS_VIEW AS
SELECT TRACK_NAME, TRACK_POPULARITY, VALENCE,
CASE
  WHEN valence > 0.535 THEN 'Above Avg Valence'
  WHEN valence = 0.535 THEN 'Avg Valence'
  WHEN valence < 0.535 THEN 'Below Avg Valence'
END AS Valence_Analysis
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE VALENCE IS NOT NULL;

SELECT * FROM PLAYLIST_VALENCE_ANALYSIS_VIEW;
```

```

246  /*
247  15. Valence Analysis : Top 20 track_name, valence, track_popularity,
248      valence > 0.535 -> Above Avg Valence
249      valence = 0.535 -> Avg Valence
250      valence < 0.535 -> Below Average'
251  */
252
253  CREATE OR REPLACE VIEW PLAYLIST_VALENCE_ANALYSIS_VIEW AS
254  SELECT TRACK_NAME, TRACK_POPULARITY, VALENCE,
255  CASE
256      WHEN valence > 0.535 THEN 'Above Avg Valence'
257      WHEN valence = 0.535 THEN 'Avg Valence'
258      WHEN valence < 0.535 THEN 'Below Avg Valence'
259  END AS Valence_Analysis
260  FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
261  WHERE VALENCE IS NOT NULL;
262
263  | SELECT * FROM PLAYLIST_VALENCE_ANALYSIS_VIEW;
264

```

Results

Chart

	TRACK_NAME	TRACK_POPULARITY	VALENCE	VALENCE_ANALYSIS
1	Yellow	91	0.2850	Below Avg Valence
2	All The Small Things	84	0.6840	Above Avg Valence
3	Breathe	69	0.2780	Below Avg Valence
4	In the End	88	0.4000	Below Avg Valence
5	Bye Bye Bye	74	0.8610	Above Avg Valence

16. Number of Songs for different valence ranges(above)

```

SELECT VALENCE_ANALYSIS, COUNT(TRACK_NAME) AS TOT_SONGS
FROM PLAYLIST_VALENCE_ANALYSIS_VIEW
GROUP BY 1
ORDER BY 2 DESC;

```

```

265  /*
266  16. Number of Songs for different valence ranges(above)
267  */
268
269  | SELECT VALENCE_ANALYSIS, COUNT(TRACK_NAME) AS TOT_SONGS
270  FROM PLAYLIST_VALENCE_ANALYSIS_VIEW
271  GROUP BY 1
272  ORDER BY 2 DESC;
273

```

Results

Chart

	VALENCE_ANALYSIS	TOT_SONGS
1	Above Avg Valence	1,166
2	Below Avg Valence	1,130
3	Avg Valence	3

17. Speechiness Analysis : Top 20 track_name, speechiness, tempo,
speechiness > 0.081-> Above Avg Speechiness
speechiness = 0.081-> Avg Speechiness
speechiness < 0.081-> Below Speechiness

```
SELECT TRACK_NAME,TEMPO, SPEECHINESS,
CASE
    WHEN SPEECHINESS > 0.081 THEN 'Above Avg Speechiness'
    WHEN SPEECHINESS = 0.081 THEN 'Avg Speechiness'
    WHEN SPEECHINESS < 0.081 THEN 'Below Avg Speechiness'
END AS Speechiness_Analysis
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE SPEECHINESS IS NOT NULL
LIMIT 20;
```

```
275  /*
276  17. Speechiness Analysis : Top 20 track_name, speechiness, tempo,
277      speechiness > 0.081-> Above Avg Speechiness
278      speechiness = 0.081-> Avg Speechiness
279      speechiness < 0.081-> Below Speechiness
280  */
281
282  SELECT TRACK_NAME,TEMPO, SPEECHINESS,
283  CASE
284      WHEN SPEECHINESS > 0.081 THEN 'Above Avg Speechiness'
285      WHEN SPEECHINESS = 0.081 THEN 'Avg Speechiness'
286      WHEN SPEECHINESS < 0.081 THEN 'Below Avg Speechiness'
287  END AS Speechiness_Analysis
288  FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
289  WHERE SPEECHINESS IS NOT NULL
290  LIMIT 20;
```

[Results](#) [Chart](#)

	TRACK_NAME	TEMPO	SPEECHINESS	SPEECHINESS_ANALYSIS
1	Yellow	173.3720	0.0281	Below Avg Speechiness
2	All The Small Things	148.7260	0.0488	Below Avg Speechiness
3	Breathe	136.8590	0.0290	Below Avg Speechiness
4	In the End	105.1430	0.0584	Below Avg Speechiness
5	Bye Bye Bye	172.6380	0.0479	Below Avg Speechiness
6	Thong Song	121.5490	0.0654	Below Avg Speechiness
7	The Real Slim Shady	104.5040	0.0572	Below Avg Speechiness
8	Back In Black	102.0220	0.0226	Below Avg Speechiness

18. Acoustic Analysis : DISTINCT TOP 25 track_name, album, artist_name, acousticness
(acousticness BETWEEN 0 AND 0.40000 -> 'Not Acoustic'
(acousticness BETWEEN 0.40001 AND 0.80000) ->'Acoustic'
(acousticness BETWEEN 0.80001 AND 1) ->'Highly Acoustic'

```
SELECT DISTINCT TRACK_NAME, ALBUM, ARTIST_NAME, ACOUSTICNESS,
CASE
  WHEN ACOUSTICNESS BETWEEN 0 AND 0.40000 THEN 'Not Acoustic'
  WHEN ACOUSTICNESS BETWEEN 0.40001 AND 0.80000 THEN 'Acoustic'
  WHEN ACOUSTICNESS BETWEEN 0.80001 AND 1 THEN 'Highly Acoustic'
  ELSE 'Others'
END AS Acousticness_Analysis
FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
WHERE ACOUSTICNESS IS NOT NULL
LIMIT 25;
```

```
293  /*
294  18. Acoustic Analysis : DISTINCT TOP 25 track_name, album, artist_name, acousticness
295      (acousticness BETWEEN 0 AND 0.40000 -> 'Not Acoustic'
296      (acousticness BETWEEN 0.40001 AND 0.80000) ->'Acoustic'
297      (acousticness BETWEEN 0.80001 AND 1) ->'Highly Acoustic'
298  */
299
300  SELECT DISTINCT TRACK_NAME, ALBUM, ARTIST_NAME, ACOUSTICNESS,
301  CASE
302      WHEN ACOUSTICNESS BETWEEN 0 AND 0.40000 THEN 'Not Acoustic'
303      WHEN ACOUSTICNESS BETWEEN 0.40001 AND 0.80000 THEN 'Acoustic'
304      WHEN ACOUSTICNESS BETWEEN 0.80001 AND 1 THEN 'Highly Acoustic'
305      ELSE 'Others'
306  END AS Acousticness_Analysis
307  FROM SPOTIFY_DATABASE.PUBLIC.PLAYLIST
308  WHERE ACOUSTICNESS IS NOT NULL
309  LIMIT 25;
```

Results

Chart

	TRACK_NAME	ALBUM	ARTIST_NAME	ACOUSTICNESS	ACOUSTICNESS_ANALYSIS
1	Yellow	Parachutes	Coldplay	0.00239000	Not Acoustic
2	All The Small Things	Enema Of The State	blink-182	0.01030000	Not Acoustic
3	Breathe	Breathe	Faith Hill	0.17300000	Not Acoustic
4	In the End	Hybrid Theory (Bonus Edition)	Linkin Park	0.00958000	Not Acoustic
5	Bye Bye Bye	No Strings Attached	*NSYNC	0.03100000	Not Acoustic

***** THANK YOU *****