DATA ANALYSIS ON SPOTIFY TRACKS FROM JAN 2022 TO MAY 2023

SQL project to analyse and answer interesting questions related to tracks and artists. The dataset used in this project is provided by Spotify and collected from Kaggle. It contains approx. 1.47 Lakh records from January 2022 to May 2023.

BUSINESS PROBLEMS

Q1. Top 10 Favourite Songs by Daily Ranking.

Q2. Top 10 Favourite Artists by Daily Ranking.

Q3. Top 10 Songs which can be recommended to Fitness Freaks.

Q4. Top 5 Emerging Artists of 2022 to May 2023.

Q5. Recommend Top 10 Song for the Weekend.

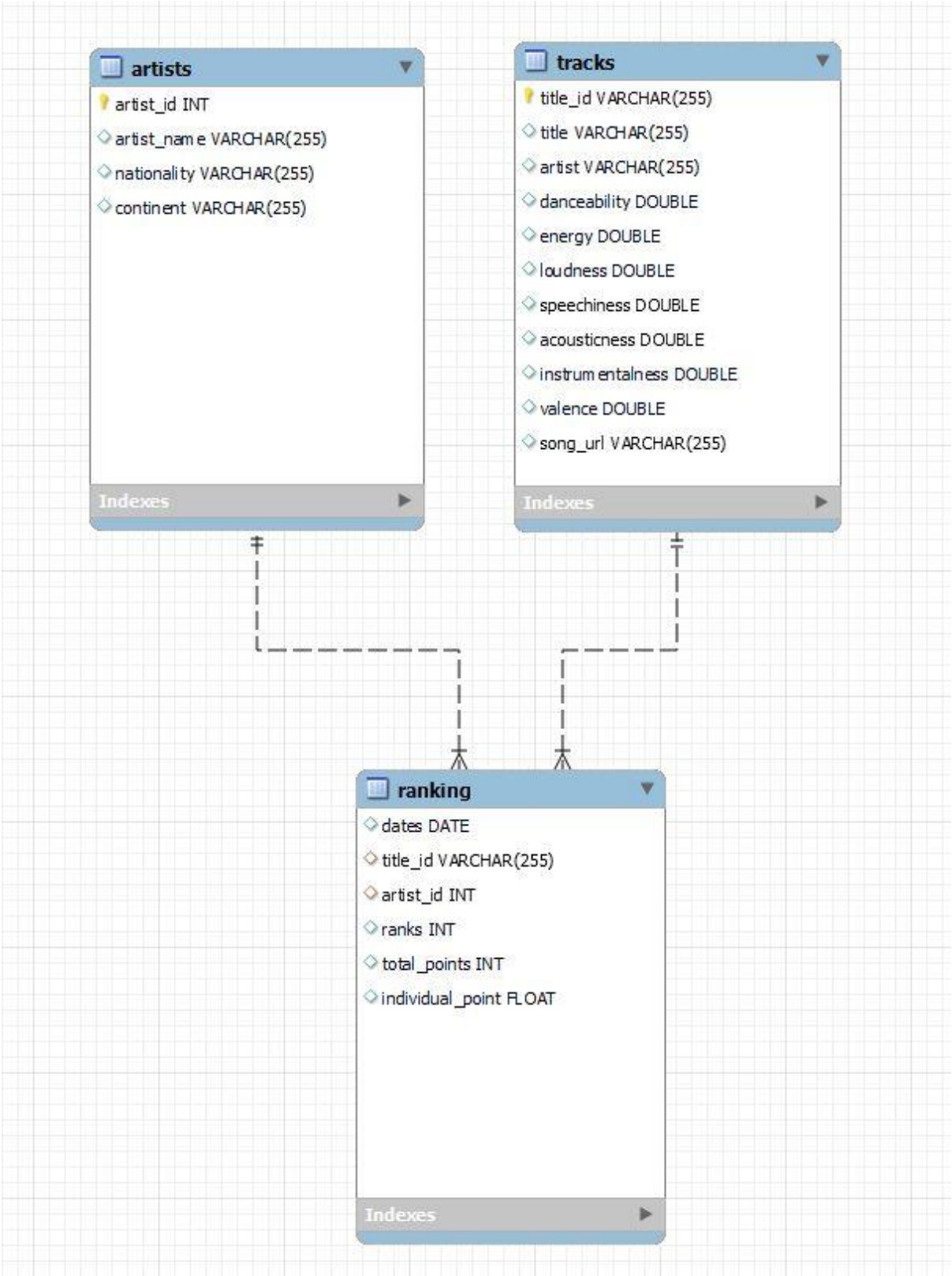
Q6. Weekly trending songs for the whole period.

Q7. Artist Name and 'Count of days' when they have two or more than two songs in the top 200 songs list.

Q8. Check if there is any association between the number of artists per song and the popularity of the song.

Q9. Number of artists in each continent and respective country.

SPOTIFY DATABASE



NORMALIZED DATA IN TABLES

CREATING A TABLE FOR TRACKS

```
CREATE TABLE tracks (
    title_id VARCHAR(255) PRIMARY KEY,
    title VARCHAR(255),
    artist VARCHAR(255),
    danceability FLOAT(25),
    energy FLOAT(25),
    loudness FLOAT(25),
    speechiness FLOAT(25),
    acousticness FLOAT(25),
    instrumentalness FLOAT(25),
    valence FLOAT(25),
    song_url VARCHAR(255)
);
```

CREATING A TABLE FOR ARTISTS

```
CREATE TABLE artists(
artist_id int PRIMARY KEY,
artist_name VARCHAR(255),
nationality VARCHAR(255),
continent VARCHAR(255)
);
```

CREATING A TABLE FOR RANK OF TRACKS

```
CREATE TABLE ranking(
 dates DATE,
 title_id VARCHAR(255),
 artist_id INT,
 ranks INT,
 total_points INT,
 individual_point FLOAT,
 FOREIGN key (title_id) REFERENCES tracks(title_id),
 FOREIGN key (artist_id) REFERENCES artists(artist_id)
);
```

Q1. Top 10 Favourite Songs by Daily Ranking.

```
WITH cte
AS (
	SELECT t.title
		,COUNT(DISTINCT r.dates) AS song_of_the_day
	FROM ranking r
	INNER JOIN tracks t ON r.title_id = t.title_id
	WHERE r.ranks = 1
	GROUP BY 1
	ORDER BY 2 DESC LIMIT 10
	)
SELECT title as Song_Name
FROM cte;
```

Q2. Top 10 Favourite Artists by Daily Ranking.

```
WITH cte
AS (
	SELECT a.artist_name
		,count(DISTINCT r.dates) AS song_of_the_day
	FROM ranking r
	INNER JOIN artists a ON a.artist_id = r.artist_id
	WHERE r.ranks = 1
	GROUP BY 1
	ORDER BY 2 DESC LIMIT 10
	)
SELECT artist_name as Artist_Name
FROM cte;
```

Q3. Top 10 Songs which can be recommended to Fitness Freaks.

```
SELECT t.title
	,count(DISTINCT r.dates) '#intop200'
	,min(r.ranks) as min_rank_over_the_period
FROM ranking r
INNER JOIN tracks t ON r.title_id = t.title_id
WHERE energy > 0.8
	AND loudness > - 500
GROUP BY 1
ORDER BY 2 DESC,3 ASC
limit 10;
```

Q4. Top 5 Emerging Artists of 2022 to May 2023.

```
CREATE TEMPORARY TABLE monthly_artists_points
SELECT DATE_FORMAT(r.dates, '%Y-%m') AS yearmonth
	,r.artist_id
	,SUM(r.individual_point) AS points
FROM ranking r
GROUP BY 1
	,2
ORDER BY 1
	,3 DESC;

SELECT sub.yearmonth
	,sub.artist_id
```

```
	,a.artist_name
FROM (
	SELECT yearmonth
		,artist_id
		,points
		,rank() OVER (
			PARTITION BY yearmonth ORDER BY points DESC
			) AS rnk
	FROM monthly_artists_points
	) sub
INNER JOIN artists a ON sub.artist_id = a.artist_id
WHERE rnk = 1;

DROP TABLE IF EXISTS monthly_artists_points;
```

Q5. Recommend Top 10 Song for the Weekend.

```
SELECT t.title
FROM (
	SELECT title_id
		,count(title_id) AS counts
		,sum(total_points) AS all_points
	FROM (
		SELECT CASE
				WHEN date_format(dates, '%W') IN (
					'Saturday'
					,'Sunday'
					)
				THEN "Yes"
```

```
                ELSE "No"
                END AS weekend
        ,title_id
        ,total_points
    FROM ranking
    WHERE CASE
            WHEN date_format(dates, '%W') IN (
                'Saturday'
                ,'Sunday'
                )
            THEN "Yes"
            ELSE "No"
            END = "Yes"
    ) sub
GROUP BY 1
ORDER BY 3 DESC limit 10
) sub2
INNER JOIN tracks t ON sub2.title_id = t.title_id;
```

Q6. Weekly trending songs for the whole period.

```
-- select max(dates) from ranking;
-- SELECT COUNT(DISTINCT WEEK(dates)) FROM ranking where dates > '2022-01-01' and  dates < '2022-01-09';
-- SELECT COUNT(DISTINCT WEEK(dates)) FROM ranking where dates > '2023-05-27' and  dates <= '2023-05-29';
SELECT start_of_week
    ,t.title
FROM (
```

```
SELECT *
FROM (
	SELECT *
		,rank() OVER (
			PARTITION BY yearweek ORDER BY total_points DESC
			) AS rnk
	FROM (
		SELECT yearweek
			,title_id
			,min(dates) start_of_week
			,avg(all_points) AS total_points
		FROM (
			SELECT *
				,sum(total_points) OVER (
					PARTITION BY yearweek
					,title_id ORDER BY yearweek
					) AS all_points
			FROM (
				SELECT *
					,CONCAT (
						year(dates)
						,'-'
						,CASE
							WHEN week(dates) < 10
								THEN CONCAT (
									'0'

,week(dates)
```

```
                                                                    )
                                                                ELSE week(dates)
                                                                END
                                                            ) AS yearweek
                                                    FROM (
                                                        SELECT DISTINCT dates
                                                            ,title_id
                                                            ,ranks
                                                            ,total_points
                                                        FROM ranking
                                                        WHERE dates > '2022-01-01'
                                                            AND dates <= '2023-05-27'
                                                        ORDER BY 1
                                                            ,3
                                                        ) sub
                                                    ) sub2
                                                ) sub3
                                            GROUP BY yearweek
                                                ,title_id
                                            ORDER BY 1
                                                ,2
                                                ,4 DESC
                                            ) sub4
                                        ) sub5
                                    WHERE rnk = 1
                                    ) sub6
                            INNER JOIN tracks t ON sub6.title_id = t.title_id
                            ORDER BY 1;
```

Q7. Artist Name and 'Count of days' when they have two or more than two songs in the top 200 songs list.

```
SELECT *
FROM ranking limit 10;


SELECT a.artist_name
	,count(sub.dates) AS days
FROM (
	SELECT dates
		,artist_id
		,count(title_id) AS cnt
	FROM ranking
	GROUP BY 1
		,2
	ORDER BY 1
		,2
	) sub
INNER JOIN artists a ON sub.artist_id = a.artist_id
WHERE sub.cnt >= 2
GROUP BY 1
ORDER BY 2 DESC;


-- select * from tracks order by artist limit 100;
-- select * from artists limit 10;
SELECT *
FROM tracks limit 10;


SELECT *
```

```
FROM ranking
ORDER BY dates
	,ranks limit 400;
```

Q8. Check if there is any association between the number of artists per song and the popularity of the song.

```
SELECT sum(CASE
			WHEN cnt = 1
				THEN 1
			ELSE 0
			END) AS '#of_artists1'
	,sum(CASE
			WHEN cnt = 2
				THEN 1
			ELSE 0
			END) AS '#of_artists2'
	,sum(CASE
			WHEN cnt = 3
				THEN 1
			ELSE 0
			END) AS '#of_artists3'
	,sum(CASE
			WHEN cnt = 4
				THEN 1
			ELSE 0
			END) AS '#of_artists4'
	,sum(CASE
			WHEN cnt = 5
```

```
                    THEN 1
                ELSE 0
                END) AS '#of_artists5'
        ,sum(CASE
                WHEN cnt = 6
                    THEN 1
                ELSE 0
                END) AS '#of_artists6'
FROM (
        SELECT dates
            ,title_id
            ,count(artist_id) AS cnt
        FROM ranking
        WHERE ranks <= 10 -- 5% of
        GROUP BY 1
            ,2
        HAVING count(artist_id)
        ORDER BY 1
        ) sub;
```

Q9. Number of artists in each continent and respective country.

```
SELECT continent
        ,nationality
        ,count(artist_id) AS no_of_artists
FROM artists
WHERE continent <> 'Unknown'
GROUP BY 1
        ,2
```

```
ORDER BY 1 ASC
        ,2 ASC;
```

10. The number of songs belonging to each continent and country produced by individual singers.

```
SELECT continent
        ,nationality
        ,count(t.title_id) AS no_of_songs
FROM tracks t
LEFT JOIN ranking r ON t.title_id = r.title_id
LEFT JOIN artists a ON r.artist_id = a.artist_id
WHERE a.continent IS NOT NULL
        AND a.nationality IS NOT NULL
        AND a.continent <> 'Unknown'
        AND a.nationality <> 'Unknown'
GROUP BY 1
        ,2
ORDER BY 1
        ,3 DESC;
```