Big Data Systems (S1-23_DSEOGZG522) Assignment

# Group Submission

- **GAGAN GUPTA (2022OG04044)**
- **VINAY SINGLA (2022OG04045)**

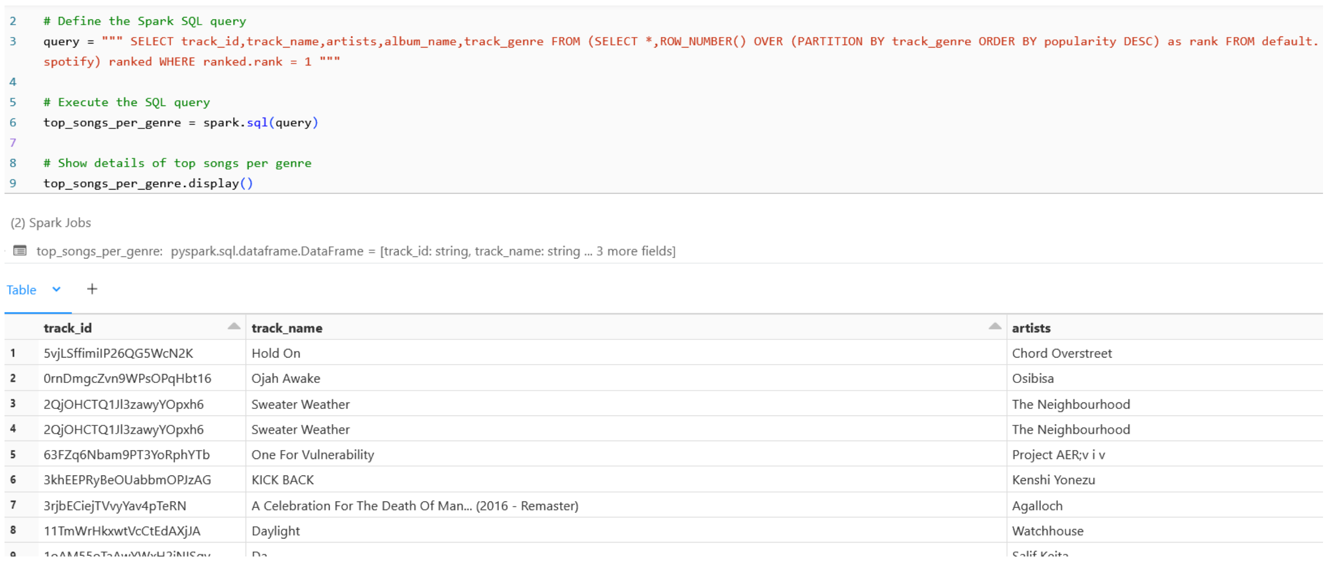
Part 3 - OLAP Analytics

❖ *GITHUB URL : Execution Video*

https://github.com/sinvin-dse/spotify/blob/main/BDS_Assignment_Part_3_OLAP_Analytics_GG_VS.mp4

❖ *Track details group by genre with the highest popularity*

Using PySPARK and SPARK SQL, filter tracks by grouping based on track genre and order by popularity in descending order.



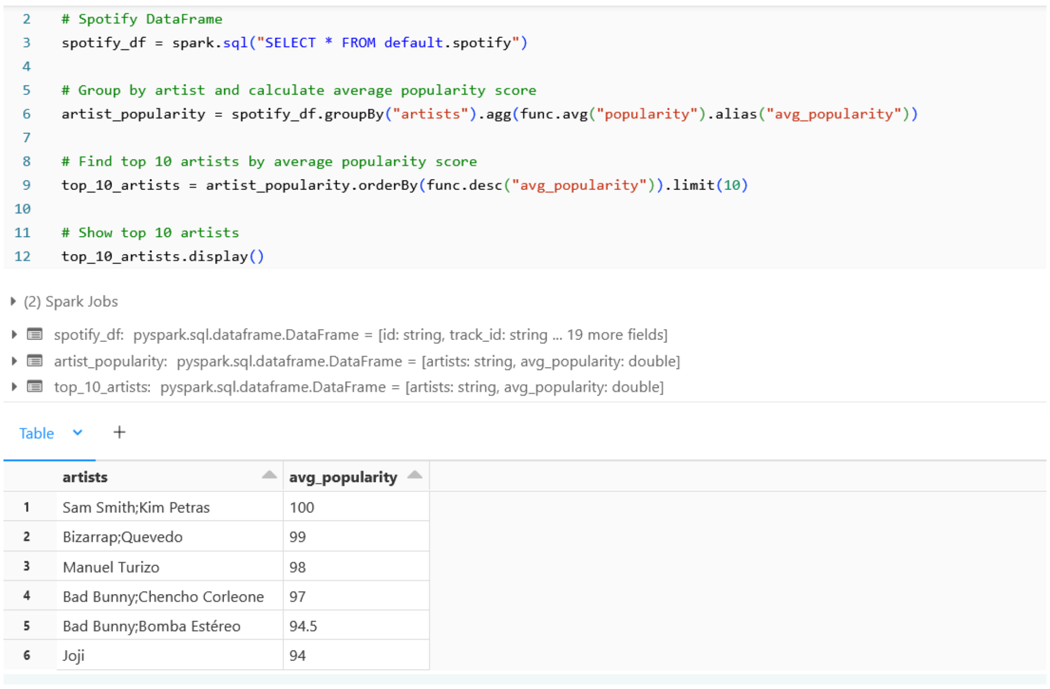
```
# Define the Spark SQL query
query = """ SELECT track_id,track_name,artists,album_name,track_genre FROM (SELECT *,ROW_NUMBER() OVER (PARTITION BY track_genre ORDER BY popularity DESC) as rank FROM default.
spotify) ranked WHERE ranked.rank = 1 """

# Execute the SQL query
top_songs_per_genre = spark.sql(query)

# Show details of top songs per genre
top_songs_per_genre.display()
```

```
(2) Spark Jobs
top_songs_per_genre:  pyspark.sql.dataframe.DataFrame = [track_id: string, track_name: string ... 3 more fields]
```

	track_id	track_name	artists
1	5vjLSffimiIP26QG5WcN2K	Hold On	Chord Overstreet
2	0rnDmgcZvn9WPsOPqHbt16	Ojah Awake	Osibisa
3	2QjOHCTQ1Jl3zawyYOpxh6	Sweater Weather	The Neighbourhood
4	2QjOHCTQ1Jl3zawyYOpxh6	Sweater Weather	The Neighbourhood
5	63FZq6Nbam9PT3YoRphYTb	One For Vulnerability	Project AER;v i v
6	3khEEPRyBeOUabbmOPJzAG	KICK BACK	Kenshi Yonezu
7	3rjbECiejTVvyYav4pTeRN	A Celebration For The Death Of Man... (2016 - Remaster)	Agalloch
8	11TmWrHkxwtVcCtEdAXjJA	Daylight	Watchhouse
9	1oAM55oTaAwYWxH2iNISqv	Da	Salif Keita

❖ *Top 10 Artists with the highest average popularity*

Filter the top 10 artists with the highest average popularity using PySpark SQL functions.



```python
# Spotify DataFrame
spotify_df = spark.sql("SELECT * FROM default.spotify")

# Group by artist and calculate average popularity score
artist_popularity = spotify_df.groupBy("artists").agg(func.avg("popularity").alias("avg_popularity"))

# Find top 10 artists by average popularity score
top_10_artists = artist_popularity.orderBy(func.desc("avg_popularity")).limit(10)

# Show top 10 artists
top_10_artists.display()
```

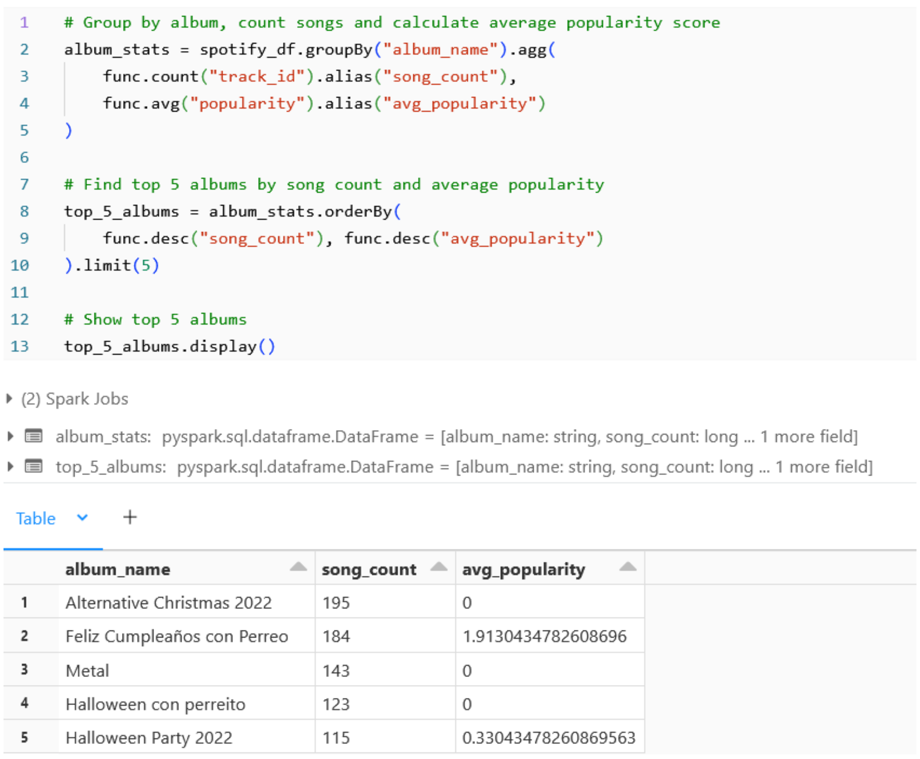
```
(2) Spark Jobs
spotify_df:  pyspark.sql.dataframe.DataFrame = [id: string, track_id: string ... 19 more fields]
artist_popularity:  pyspark.sql.dataframe.DataFrame = [artists: string, avg_popularity: double]
top_10_artists:  pyspark.sql.dataframe.DataFrame = [artists: string, avg_popularity: double]
```

Table

	artists	avg_popularity
1	Sam Smith;Kim Petras	100
2	Bizarrap;Quevedo	99
3	Manuel Turizo	98
4	Bad Bunny;Chencho Corleone	97
5	Bad Bunny;Bomba Estéreo	94.5
6	Joji	94

❖ *Top 5 Albums with most no of songs and having highest average probability score*

Filter Top 5 Albums with most no of songs and having highest average probability score using SQL functions Count & Average.



```python
# Group by album, count songs and calculate average popularity score
album_stats = spotify_df.groupBy("album_name").agg(
    func.count("track_id").alias("song_count"),
    func.avg("popularity").alias("avg_popularity")
)

# Find top 5 albums by song count and average popularity
top_5_albums = album_stats.orderBy(
    func.desc("song_count"), func.desc("avg_popularity")
).limit(5)

# Show top 5 albums
top_5_albums.display()
```

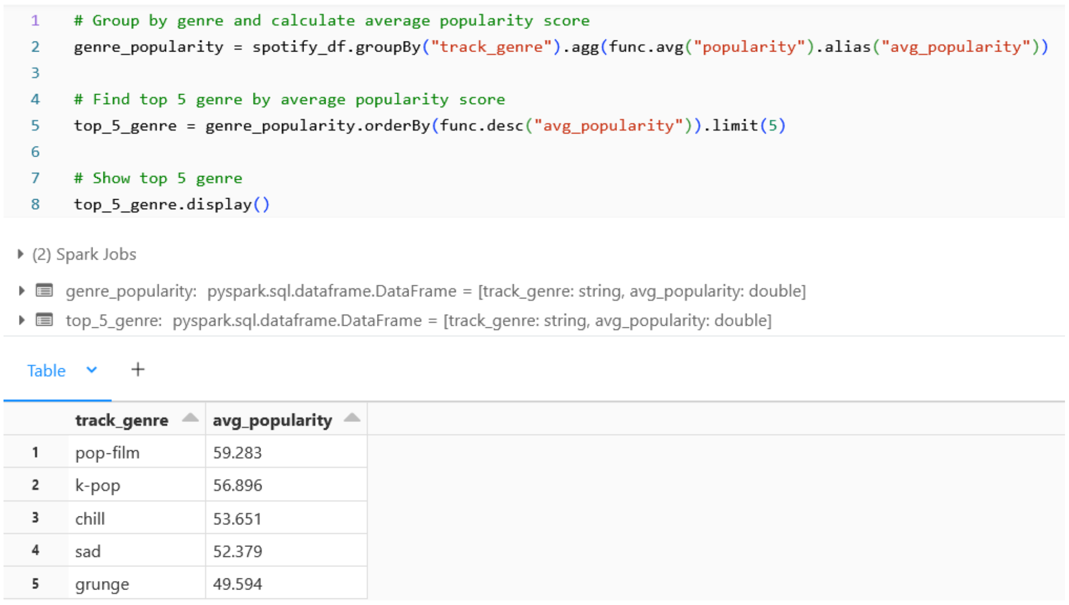
```
(2) Spark Jobs
album_stats:  pyspark.sql.dataframe.DataFrame = [album_name: string, song_count: long ... 1 more field]
top_5_albums:  pyspark.sql.dataframe.DataFrame = [album_name: string, song_count: long ... 1 more field]
```

Table

	album_name	song_count	avg_popularity
1	Alternative Christmas 2022	195	0
2	Feliz Cumpleaños con Perreo	184	1.9130434782608696
3	Metal	143	0
4	Halloween con perreito	123	0
5	Halloween Party 2022	115	0.33043478260869563

❖ *Top Genre with the highest average popularity*

This will filter the Top Genre with the highest average popularity using Average function.



```python
# Group by genre and calculate average popularity score
genre_popularity = spotify_df.groupBy("track_genre").agg(func.avg("popularity").alias("avg_popularity"))

# Find top 5 genre by average popularity score
top_5_genre = genre_popularity.orderBy(func.desc("avg_popularity")).limit(5)

# Show top 5 genre
top_5_genre.display()
```

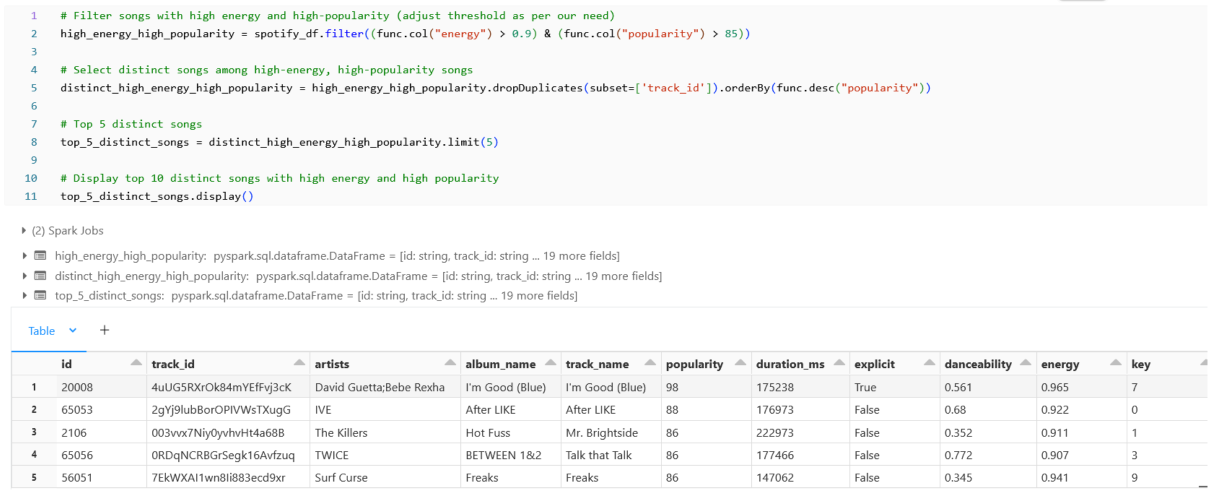
```
(2) Spark Jobs
genre_popularity:  pyspark.sql.dataframe.DataFrame = [track_genre: string, avg_popularity: double]
top_5_genre:  pyspark.sql.dataframe.DataFrame = [track_genre: string, avg_popularity: double]
```

Table

	track_genre	avg_popularity
1	pop-film	59.283
2	k-pop	56.896
3	chill	53.651
4	sad	52.379
5	grunge	49.594

❖ *Top 10 Tracks with the highest energy & popularity score*

This will filter the Top 10 tracks with the highest energy and popularity.



```python
# Filter songs with high energy and high-popularity (adjust threshold as per our need)
high_energy_high_popularity = spotify_df.filter((func.col("energy") > 0.9) & (func.col("popularity") > 85))

# Select distinct songs among high-energy, high-popularity songs
distinct_high_energy_high_popularity = high_energy_high_popularity.dropDuplicates(subset=['track_id']).orderBy(func.desc("popularity"))

# Top 5 distinct songs
top_5_distinct_songs = distinct_high_energy_high_popularity.limit(5)

# Display top 10 distinct songs with high energy and high popularity
top_5_distinct_songs.display()
```

```
(2) Spark Jobs
high_energy_high_popularity:  pyspark.sql.dataframe.DataFrame = [id: string, track_id: string ... 19 more fields]
distinct_high_energy_high_popularity:  pyspark.sql.dataframe.DataFrame = [id: string, track_id: string ... 19 more fields]
top_5_distinct_songs:  pyspark.sql.dataframe.DataFrame = [id: string, track_id: string ... 19 more fields]
```

Table

	id	track_id	artists	album_name	track_name	popularity	duration_ms	explicit	danceability	energy	key
1	20008	4uUG5RXrOk84mYEfFvj3cK	David Guetta;Bebe Rexha	I'm Good (Blue)	I'm Good (Blue)	98	175238	True	0.561	0.965	7
2	65053	2gYj9lubBorOPIVWsTXugG	IVE	After LIKE	After LIKE	88	176973	False	0.68	0.922	0
3	2106	003vvx7Niy0yvhvHt4a68B	The Killers	Hot Fuss	Mr. Brightside	86	222973	False	0.352	0.911	1
4	65056	0RDqNCRBGrSegk16Avfzuq	TWICE	BETWEEN 1&2	Talk that Talk	86	177466	False	0.772	0.907	3
5	56051	7EkWXAI1wn8Ii883ecd9xr	Surf Curse	Freaks	Freaks	86	147062	False	0.345	0.941	9