Route: GET /query/songs1

Request Path: /query/songs1

Example Request: GET http://localhost:8080/query/songs1?x=2022-01-01&y=2022-12-31

Description:

This route handles a GET request to retrieve distinct song names from Ranks table

of MySQL database based on user’s specified date range.

Request Parameters: None

Query Parameters:

x: Date parameter representing the starting date of the range (format: YYYY-MM-DD).

y: Date parameter representing the ending date of the range (format: YYYY-MM-DD).

Response Parameters:

Content Type: JSON

Fields:

song: (Array) List of distinct song names retrieved from the database.

Example Response:["Song A", "Song B", "Song C"]

Route: GET /query/songs2

Request Path: /query/songs2

Example Request: GET http://localhost:8080/query/songs2?x=2022-01-01&y=2022-12-31

Description:

This route handles a GET request to retrieve distinct song names and corresponding artists from the Ranks table and Points table of a MySQL database based on the user’s specified date range.

Request Parameters: None

Query Parameters:

x: Date parameter representing the starting date of the range (format: YYYY-MM-DD).

y: Date parameter representing the ending date of the range (format: YYYY-MM-DD).

Response:

Content Type: JSON

Fields:

songs: (String)List of distinct song names retrieved from the database.

artists: (String)List of corresponding artists for the retrieved songs.

Example Response:

[

{

"songs": "Song A",

"artists": "Artist A"

},

{

"songs": "Song B",

"artists": "Artist B" } ]

Route: GET /query/popular-spotify-song

Request Path: /query/popular-spotify-song

Example Request: GET http://localhost:8080/query/popular-spotify-song

Description:

This route handles a GET request to retrieve the most popular song from the Points table of a MySQL database on Spotify's charts. The popularity is determined by the total points accumulated by each song. This route executes a MySQL query to find the song with the highest total points on Spotify's charts.

Request Parameters: None

Query Parameters: None

Response Parameters:

Content Type: JSON

Fields:

songs: (String)The title of the most popular song.

artists: (String)The artist of the most popular song.

totalPoints: (Number)The total points accumulated by the most popular song on Spotify.

Example Response:

{ "songs": "Song A", "artists": "Artist A", "totalPoints": 1000 }

Route: GET /query/biggest-difference

Request Path: /query/biggest-difference

Example Request: GET http://localhost:8080/query/biggest-difference

Description:

This route executes a MySQL query to find the biggest difference in chart rankings between the peak of a song on Spotify and Billboard. It calculates the absolute difference in ranks for each song between the Ranks and Points tables, and returns the song with the largest difference.

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

songs: (String)The title of the song with the biggest difference in rankings.

difference: (Number)The absolute difference in chart rankings between the peak of the song on Spotify and Billboard.

Example Response:

[ { "songs": "Song A", "difference": 5 },

{ "songs": "Song B", "difference": 8 },

{ "songs": "Song C", "difference": 3 }

]

Route: GET /query/most-danceable-nationalities

Request Path: /query/most-danceable-nationalities

Example Request: GET http://localhost:8080 /query/most-danceable-nationalities

Description:

This route executes a MySQL query to determine which nationalities have the most danceable songs. It calculates the average danceability score for songs by artists from each nationality, and returns the nationalities with the highest average danceability scores.

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

Nationality: (String) The nationality of the artists.

Danceability: (Number)The average danceability score of songs by artists from each nationality.

Example Response:

[

{ "Nationality": "American", "Danceability": 0.75 },

{ "Nationality": "British", "Danceability": 0.80 },

{ "Nationality": "Spanish", "Danceability": 0.70 }

]

Route: GET /query/nationalities-with-most-weeks

Request Path: /query/nationalities-with-most-weeks

Example Request: GET http://localhost:8080 /query/nationalities-with-most-weeks

Description:

This route executes a MySQL query to determine the nationalities with the most weeks on the Billboard Hot 100, considering only nationalities with a song that reached number 1 on Spotify's charts. It calculates the total number of weeks on the Hot 100 for each nationality, based on the number of songs by artists from each nationality, and returns the nationalities, artists, and the total number of weeks.

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

Nationality: (String) The nationality of the artists.

Artist: (String)The name of the artist.

Num: (Number)The total number of weeks on the Billboard Hot 100 for songs by artists from each nationality.

Example Response:

[

{ "Nationality": "American", "Artist": "Artist A", "Num": 50 },

{ "Nationality": "British", "Artist": "Artist B", "Num": 45 },

{ "Nationality": "Canadian", "Artist": "Artist C", "Num": 40 }

]

Route: GET /query/biggest-change-in-top10

Request Path: /query/ biggest-change-in-top10

Example Request: GET http://localhost:8080 /query/ biggest-change-in-top10

Description:

This route executes a MySQL query to determine the weeks with the biggest absolute change in the top 10 positions on the Billboard Hot 100 chart. It calculates the sum of the absolute differences between the last week's rank and the current week's rank for songs that were in the top 10 positions, and returns the date and the total change for each week.

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

Date: (String) The date of the week.

Change: (Number) The total absolute change in the top 10 positions for that week.

Example Response:

[

{ "Date": "2022-01-01", "Change": 50 },

{ "Date": "2022-01-08", "Change": 45 },

{ "Date": "2022-01-15", "Change": 40 }

]

Route: GET /query/most-danceable-dates

Request Path: /query/ most-danceable-dates

Example Request: GET http://localhost:8080 /query/ most-danceable-dates

Description:

This route executes a MySQL query to determine the dates with the most danceable songs according to both Spotify and Billboard charts. It calculates the average danceability score for songs on each date from both sources, combines them, and returns the dates with the highest combined average danceability scores

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

Date: (String) The date with the most danceable songs.

AverageDanceability: (Number) The combined average danceability score for songs on that date from both Spotify and Billboard charts.

Example Response:

[ { "Date": "2022-01-01", "AverageDanceability": 0.75 },

{ "Date": "2022-01-08", "AverageDanceability": 0.80 },

{ "Date": "2022-01-15", "AverageDanceability": 0.70 } ]

Route: GET /query/ above-average-danceability

Request Path: /query/ above-average-danceability

Example Request: GET http://localhost:8080 /query/ above-average-danceability

Description:

This route executes a MySQL query to determine the number of songs with above-average danceability among the 50 longest songs charting in the top 10 for both Billboard and Spotify. It calculates the average danceability score for all songs, and then checks if each song's danceability score is above or below the average. Finally, it returns the count of songs falling into each category.

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

Dance: (String)Indicates whether the song's danceability is above or below average.

Count: (Number)The count of songs falling into each category.

Example Response:

[ { "Dance": "Above Average", "Count": 25 },

{ "Dance": "Below Average", "Count": 25 } ]

Route: GET /query/ loudness-analysis

Request Path: /query/ loudness-analysis

Example Request: GET http://localhost:8080 /query/ loudness-analysis

Description:

This route executes a MySQL query to determine how many songs out of the 100 loudest songs have made it to the top 10 in both Billboard and Spotify charts. It calculates the counts of songs that made it to the top 10 in Billboard and Spotify separately and returns the results in a specified JSON format.

Request Parameters: None

Query Parameters: None

Response parameters:

Content Type: JSON

Fields:

Top10Billboard: (Number) The count of songs that made it to the top 10 in Billboard charts.

Top10Spotify: (Number)The count of songs that made it to the top 10 in Spotify charts.

Example Response:

{ "Top10Billboard": 40, "Top10Spotify": 35 }