Recommended Songs Query

July 25, 2019 3:27 PM

```
SELECT
    M4.Song_ID,
    Song.Song_Name,
    Song.Song_Length,
    Song.Genre,
    Song.BPM,
    Album.Album_Name,
    Artist.Artist_Name,
    Album.Date_Released
FROM
    (
    SELECT
        Song_ID
    FROM
        Favourites
    NATURAL JOIN(
        SELECT
            Client_ID
        FROM
            (
            SELECT
                Favourites.Client_ID,
                COUNT(Song_ID) AS Matches
            FROM
                Favourites
            WHERE
                Favourites.Client_ID <> ? AND Song_ID IN(
                SELECT
                    Song_ID
                FROM
                    Favourites
                WHERE
                    Favourites.Client_ID = ?
            )
        GROUP BY
            Favourites.Client_ID
        ) AS M
    WHERE
        M.Matches =(
        SELECT
            MAX(M2.Matches)
        FROM
            (
            SELECT
                Favourites.Client_ID,
                COUNT(Song_ID) AS Matches
            FROM
                Favourites
            WHERE
                Favourites.Client_ID <> ? AND Song_ID IN(
                SELECT
                    Song_ID
                FROM
                    Favourites
                WHERE
                    Favourites.Client_ID = ?
            )
        GROUP BY
            Favourites.Client_ID
        ) AS M2
    )
    ) AS M3
WHERE
    Song_ID NOT IN(
    SELECT
        Favourites.Song_ID
    FROM
        Favourites
    WHERE
        Favourites.Client_ID = ?
)
) AS M4,
Song,
Album,
Artist
WHERE
    M4.Song_ID = Song.Song_ID AND Song.Album_ID = Album.Album_ID AND Album.Artist_ID =
Artist.Artist_ID
```

Step 4: Finally, display these songs
Now we have the list of Song_ID we are recommending to the user. We now join this list with the Song, Album, and Artist tables and select the columns we want to display to give the user the necessary info they would want to see, consistent with our song view.

Step 1: Find how many matches the other users have with the current user
This is done by counting the songs per user that where the Client_ID is NOT the logged in User, and where the Song_ID is IN the current Users Favourites list

Step 2: Find the user with the most matches
This is done in the WHERE clause stating that Matches = the nested query giving the max matches

Step 3: Get the Song_ID list of the user with most matches minus the current user's favourite list
Since MINUS is not a supported function, we use NOT IN. We select the Song_ID s of the matched used that are NOT IN the list of Song_ID s of the current user's favourites list.

Join the Tables based on these matching columns