**Proyecto SQL: Plataforma de Streaming Musical**

**1. Primeros pasos: Creación y preparación de la base de datos**

Para comenzar el proyecto seguimos los pasos iniciales para definir y preparar la base de datos:

1. **Diseño del modelo lógico (ERD):**
   * Identificamos las entidades necesarias para simular una plataforma de streaming de música.
   * Creamos tablas principales: usuarios, planes, suscripciones, artistas, albumes, canciones, playlists, historial\_reproduccion, likes.
   * Relacionamos las tablas respetando cardinalidades (1:N, N:M).
2. **Implementación en MySQL Workbench:**
   * Creamos la base de datos music\_streaming.
   * Definimos cada tabla con sus **claves primarias (PK)** y **claves foráneas (FK)**.
   * Definimos restricciones como NOT NULL, UNIQUE, DEFAULT y tipos de datos adecuados (VARCHAR, INT, DATE, DECIMAL).
3. **Población de datos:**
   * Insertamos al menos 15 registros coherentes en cada tabla.
   * Ejemplo: artistas con varios álbumes, usuarios con distintas playlists, canciones con reproducciones, etc.
4. **Revisión del esquema:**
   * Exportamos el **ERD** desde Workbench para visualizar todas las relaciones.
   * Ejecutamos consultas simples (SELECT \* FROM tabla) para comprobar la correcta inserción de datos.

Con esto, la base de datos queda lista para comenzar con las consultas SQL.

**2. Realización de las consultas SQL**

Se ejecutaron un total de **50 queries**, que abarcan JOINS, subconsultas, funciones de agregación y filtros. Cada consulta responde a una pregunta de negocio real.

A continuación, se listan las 50 consultas junto con su explicación breve:

**Query 1: Listar todas las canciones con el nombre de su álbum y artista.**

**SQL:**

SELECT c.titulo, al.titulo AS album, a.nombre AS artista

FROM canciones c

JOIN albumes al ON c.album\_id = al.album\_id

JOIN artistas a ON al.artista\_id = a.artista\_id;

**Explicación:** Une canciones con su álbum y artista, mostrando información completa de cada canción.

**Query 2: Mostrar usuarios y el plan al que están suscritos.**

**SQL:**

SELECT u.nombre, p.nombre\_plan

FROM usuarios u

JOIN suscripciones s ON u.usuario\_id = s.usuario\_id

JOIN planes p ON s.plan\_id = p.plan\_id

WHERE s.estado = 'activa';

**Explicación:** JOINS para relacionar usuarios con suscripciones y planes activos.

**Query 3: Ver playlists de cada usuario junto con cuántas canciones contiene.**

**SQL:**

SELECT u.nombre, p.nombre AS playlist, COUNT(pc.cancion\_id) AS total\_canciones

FROM playlists p

JOIN usuarios u ON p.usuario\_id = u.usuario\_id

LEFT JOIN playlist\_cancion pc ON p.playlist\_id = pc.playlist\_id

GROUP BY p.playlist\_id, u.nombre;

**Explicación:** Cuenta canciones dentro de cada playlist por usuario.

**Ejemplo de consultas avanzadas (de las más relevantes)**

* **Query 7: Canciones con más reproducciones que el promedio global.**  
  Identifica canciones “hits” con un subquery que calcula el promedio.
* **Query 28: Ranking de artistas por número de oyentes únicos.**  
  Combina múltiples tablas (artistas, albumes, canciones, historial\_reproduccion) para generar un ranking de popularidad.
* **Query 29: Plan con mayor ingreso generado (precio \* suscriptores).**  
  Calcula ingresos por plan con SUM() y muestra el más rentable.
* **Query 43: Canción más reproducida por cada usuario.**  
  Usa funciones de agregación y ranking (MAX()) para obtener el top individual de cada usuario.

**3. Extras y recomendaciones**

Además del trabajo con queries, se realizaron pasos adicionales para enriquecer el proyecto:

1. **Visualización del ERD:**
   * El diagrama entidad-relación fue exportado como imagen .png y como modelo .mwb para documentación.
2. **Repositorio GitHub:**
   * Se subieron tres archivos principales:
     + create\_streaming\_musical.sql → creación de tablas e inserción de datos.
     + queries\_streaming\_musical.sql → las 50 consultas numeradas y comentadas.
     + ERD\_music\_streaming.png → diagrama de relaciones.
3. **Presentación final:**
   * Incluye explicación de objetivos, ERD y ejemplos de queries destacadas.
   * Se enfatizó la importancia de SQL en **decisiones de negocio**, como:
     + Ranking de artistas más escuchados.
     + Detección de usuarios más activos.
     + Planes que generan mayores ingresos.
4. **Aprendizajes principales:**
   * Cómo diseñar una base de datos relacional coherente.
   * El uso de consultas SQL para obtener insights reales.
   * Trabajo en equipo y documentación profesional en proyectos de datos.

**Conclusión**

El proyecto de “Plataforma de Streaming Musical” permitió simular un sistema similar a Spotify/Apple Music, donde se gestionan usuarios, suscripciones, artistas, álbumes, canciones y playlists.

Gracias a las 50 consultas realizadas:

* Aprendimos a manejar desde **JOINS básicos** hasta **subqueries complejas**.
* Demostramos cómo SQL puede generar **información valiosa para la gestión de una empresa tecnológica**.
* Documentamos todo el proceso en un repositorio GitHub con scripts, ERD y resultados, mostrando un trabajo profesional y completo.