

# Proyecto de Bases de datos para Spotify

Juan Sebastian Blanco Peña<sup>1</sup>, Samuel Stiben Suescun Hernandez<sup>2</sup>,  
Carlos Satiago Acosta Achury<sup>3</sup>

Faculta de ingenieria

Universidad UCentral

Posgrado de Analítica de datos

Curso de Bases de Datos

Bogotá, Colombia

{<sup>1</sup>jblancop, <sup>2</sup>ssuescunh,}<sup>3</sup>@ucentral.edu.co, <sup>3</sup>cacostaa1@ucentral.edu.co

November 25, 2024

## Contents

<b>1</b>	<b>Introducción (Max 250 Palabras) - (<i>Primera entrega</i>)</b>	<b>3</b>
<b>2</b>	<b>Características del proyecto de investigación que hace uso de Bases de Datos (Max 500 Palabras) - (<i>Primera entrega</i>)</b>	<b>3</b>
2.1	Título del proyecto de investigación (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	4
2.2	Objetivo general (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	4
2.2.1	Objetivos especificos (Max 100 Palabras) - ( <i>Primera entrega</i> )	4
2.3	Alcance (Max 200 Palabras) - ( <i>Primera entrega</i> ) . . . . .	5
2.4	Pregunta de investigación (Max 100 Palabras) - ( <i>Primera entrega</i> ) .	5
2.5	Hipotesis (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	5
<b>3</b>	<b>Reflexiones sobre el origen de datos e información (Max 400 Palabras) - (<i>Primera entrega</i>)</b>	<b>6</b>
3.1	¿Cual es el origen de los datos e información ? (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	6
3.2	¿Cuales son las consideraciones legales o eticas del uso de la información? (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	6
3.3	¿Cuales son los retos de la información y los datos que utilizara en la base de datos en terminos de la calidad y la consolidación? (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	7
3.4	¿Que espera de la utilización de un sistema de Bases de Datos para su proyecto? (Max 100 Palabras) - ( <i>Primera entrega</i> ) . . . . .	7

<b>4</b>	<b>Diseño del Modelo de Datos del SMBD (Sistema Manejador de Bases de Datos)(Primera entrega)</b>	<b>8</b>
4.1	Características del SMBD (Sistema Manejador de Bases de Datos) para el proyecto (Primera entrega) . . . . .	8
4.2	Diagrama modelo de datos (Primera entrega) . . . . .	8
4.3	Imágenes de la Base de Datos (Primera entrega) . . . . .	9
4.4	Código SQL - lenguaje de definición de datos (DDL) (Primera entrega) . . . . .	9
4.5	Código SQL - Manipulación de datos (DML) (Primera entrega) . .	11
4.6	Código SQL + Resultados: Vistas (Primera entrega) . . . . .	11
4.7	Código SQL + Resultados: Triggers (Primera entrega) . . . . .	12
4.8	Código SQL + Resultados: Funciones (Primera entrega) . . . . .	13
4.9	Código SQL + Resultados: procedimientos almacenados (Primera entrega) . . . . .	13
<b>5</b>	<b>Bases de Datos No-SQL (Segunda entrega)</b>	<b>14</b>
5.1	Diagrama Bases de Datos No-SQL (Segunda entrega) . . . . .	14
5.2	SMBD utilizado para la Base de Datos No-SQL (Segunda entrega)	15
<b>6</b>	<b>Aplicación de ETL (Extract, Transform, Load) y Bodega de Datos (Tercera entrega)</b>	<b>18</b>
6.1	Ejemplo de aplicación de ETL y Bodega de Datos (Tercera entrega)	18
6.2	Automatización de Datos (Tercera entrega) . . . . .	19
6.3	Integración de Datos (Tercera entrega) . . . . .	19
<b>7</b>	<b>Proximos pasos (Tercera entrega)</b>	<b>20</b>
<b>8</b>	<b>Lecciones aprendidas (Tercera entrega)</b>	<b>21</b>
<b>9</b>	<b>Bibliografía</b>	<b>22</b>

## 1 Introducción (Max 250 Palabras) - (*Primera entrega*)

El análisis de datos musicales ha cobrado gran relevancia en los últimos años, gracias al auge de plataformas de streaming como Spotify, que generan enormes volúmenes de información sobre pistas y hábitos de los usuarios. Este proyecto tiene como objetivo explorar y analizar un conjunto de datos de Spotify, enfocado en varias características clave de las pistas, como su duración, popularidad, bailabilidad, energía y otras propiedades acústicas.

La base de datos utilizada contiene información detallada de las pistas, incluyendo datos cuantitativos como la duración en milisegundos y el nivel de energía en un rango de 0.0 a 1.0, así como características categóricas como el género musical o si la pista contiene contenido explícito. Estas características permiten realizar análisis que pueden revelar patrones en los tipos de canciones más populares o identificar factores que influyen en su éxito.

El proyecto abordará preguntas como: ¿Qué factores están correlacionados con la popularidad de una canción? ¿Existen diferencias significativas entre géneros musicales en términos de bailabilidad, energía o valencia? Los resultados proporcionarán una visión profunda sobre cómo diversos atributos afectan la recepción y éxito de las pistas en la plataforma.

## 2 Características del proyecto de investigación que hace uso de Bases de Datos (Max 500 Palabras) - (*Primera entrega*)

Este proyecto se centra en el análisis de una base de datos de Spotify, que contiene información detallada sobre canciones lanzadas en distintos años, incluyendo varios atributos. La base de datos está estructurada en un formato tabular, donde cada registro representa una pista única, y las columnas corresponden a diferentes características de las pistas. La base cuenta con más de **[cantidad de registros]** y ofrece una amplia variedad de tipos de datos, desde enteros hasta valores flotantes y cadenas de texto.

Antes de proceder al análisis, será necesario realizar un proceso de limpieza y transformación de los datos. Se verificarán valores nulos o faltantes y se llevarán a cabo las imputaciones o eliminaciones pertinentes. Para asegurar la consistencia de los datos, se normalizarán algunas variables como la duración de las pistas y se estandarizarán las unidades de medida. Además, las características categóricas como el género y el indicador de contenido explícito serán codificadas en formato adecuado para el análisis posterior.

Se diseñarán consultas SQL que permitan extraer información de interés, como el promedio de popularidad de las canciones por género, o la distribución de la bailabilidad según el año de lanzamiento. Se realizarán agrupaciones de los datos según características como año o género, y se utilizarán filtros para

focalizar el análisis en subconjuntos específicos de la base de datos.

El análisis exploratorio inicial revelará posibles relaciones entre las variables. Se calcularán medidas estadísticas clave, como la correlación entre popularidad y variables como energía, valencia y duración. Si es pertinente, se aplicarán modelos predictivos simples, como regresión lineal, para determinar cómo ciertas características influyen en la popularidad de una pista.

Los resultados del análisis se presentarán mediante gráficos y visualizaciones que faciliten la comprensión de patrones emergentes. Entre estos, se espera obtener gráficos de barras y dispersión que permitan observar las relaciones entre características musicales y métricas como popularidad o bailabilidad.

Este proyecto busca utilizar las capacidades de la base de datos para identificar patrones en la música popular, entendiendo cómo diferentes atributos influyen en la percepción y el éxito de las canciones. Los hallazgos contribuirán a un mejor entendimiento de los factores detrás de las tendencias musicales actuales.

## **2.1 Título del proyecto de investigación (Max 100 Palabras) - *(Primera entrega)***

Exploración de la Influencia de Atributos Musicales en la Popularidad de Canciones en Spotify

## **2.2 Objetivo general (Max 100 Palabras) - *(Primera entrega)***

El objetivo de este proyecto es examinar los atributos de las pistas de música presentes en una base de datos de Spotify para determinar cuáles están asociados con su popularidad. A través de un análisis sistemático de diversas características musicales, como la duración, el género, y otras propiedades acústicas, se pretende identificar patrones que permitan entender mejor qué factores contribuyen al éxito de una canción en términos de reproducciones y aceptación en la plataforma.

### **2.2.1 Objetivos específicos (Max 100 Palabras) - *(Primera entrega)***

- Limpiar y procesar la base de datos de Spotify para asegurar la consistencia y calidad de los datos, incluyendo el tratamiento de valores faltantes y la normalización de variables.
- Realizar un análisis exploratorio para identificar las relaciones entre popularidad y características musicales como bailabilidad, energía, valencia y duración.
- Aplicar consultas SQL y técnicas estadísticas para comparar las características de las canciones según su género y año de lanzamiento.
- Visualizar los resultados mediante gráficos que faciliten la interpretación de los factores que influyen en el éxito de las canciones.

### **2.3 Alcance (Max 200 Palabras) - (*Primera entrega*)**

Este proyecto se centrará en el análisis de datos disponibles en una base de Spotify, enfocándose en identificar los factores que influyen en la popularidad de las canciones. El análisis abarcará desde la limpieza y transformación de los datos hasta la aplicación de consultas SQL y herramientas estadísticas básicas para obtener información relevante sobre las relaciones entre las diferentes variables.

El alcance incluye la exploración de patrones y tendencias en las pistas analizadas, además de la creación de visualizaciones que faciliten la comprensión de los resultados. Las limitaciones del proyecto incluyen el uso de los datos disponibles sin la incorporación de fuentes externas y la ausencia de modelos predictivos avanzados, ya que el enfoque principal será descriptivo.

Este análisis proporcionará una base para comprender mejor los elementos que influyen en el éxito musical, lo que puede ser útil para investigaciones futuras más detalladas.

### **2.4 Pregunta de investigación (Max 100 Palabras) - (*Primera entrega*)**

¿Cuáles son los factores clave que determinan la popularidad de una canción en Spotify? ¿Existen patrones o relaciones consistentes entre las características musicales y el éxito de una pista, considerando diferentes géneros y años de lanzamiento?

### **2.5 Hipotesis (Max 100 Palabras) - (*Primera entrega*)**

Las canciones con mayores valores de bailabilidad, energía y valencia tienden a ser más populares en Spotify. Además, se espera que ciertos géneros musicales presenten una correlación más fuerte con la popularidad, mientras que características como la duración o la presencia de contenido explícito tengan un impacto menor en el éxito de una pista.

### 3 Reflexiones sobre el origen de datos e información (Max 400 Palabras) - (*Primera entrega*)

El análisis de este proyecto se basa en datos obtenidos de la API de Spotify, que ofrece acceso a información detallada sobre canciones, artistas y álbumes. A través de funciones específicas, se extraen métricas relevantes que permiten una comprensión más profunda de las características musicales. Las definiciones de métricas como duración, popularidad, y atributos de audio (bailabilidad, energía, etc.) se establecen mediante un conjunto de criterios estandarizados, asegurando la consistencia y claridad en el análisis.

El proceso de recolección de datos implica una búsqueda inicial por artista, seguida de la obtención de información relacionada sobre álbumes y pistas. Cada canción es analizada para capturar sus características acústicas y métricas de popularidad, lo que permite evaluar cómo estos elementos influyen en el éxito de las canciones. La información recopilada no solo incluye datos cuantitativos, sino también contextuales, como géneros musicales y relaciones entre artistas.

Es importante reconocer que los datos extraídos dependen de la disponibilidad y accesibilidad de la API de Spotify. La calidad de los resultados también se ve afectada por la precisión de las definiciones y la implementación de las funciones que generan los DataFrames. Asimismo, el enfoque adoptado se limita a un conjunto específico de artistas, lo que podría influir en la generalización de los hallazgos. A pesar de estas limitaciones, el uso de una fuente de datos robusta como Spotify permite abordar preguntas de investigación sobre tendencias y patrones en la música contemporánea de manera efectiva.

#### 3.1 ¿Cual es el origen de los datos e información ? (Max 100 Palabras) - (*Primera entrega*)

Los datos utilizados en este proyecto provienen de la API de Spotify, que ofrece acceso a información detallada sobre canciones, álbumes y artistas. Mediante el uso de funciones de Python, se recopilieron métricas clave de pistas musicales, como duración, popularidad y características acústicas. Este enfoque permite obtener un conjunto de datos rico y variado, esencial para el análisis de los factores que influyen en la popularidad de las canciones.

#### 3.2 ¿Cuales son las consideraciones legales o eticas del uso de la información? (Max 100 Palabras) - (*Primera entrega*)

El uso de datos de la API de Spotify está sujeto a los términos y condiciones establecidos por la plataforma, que regulan la propiedad intelectual y la distribución de contenido. Es importante que como los datos se utilizaran exclusivamente con fines académicos y de investigación, respetando la privacidad de los artistas y oyentes. Además, se debe dar reconocimiento adecuado a la fuente de los datos en cualquier publicación o presentación resultante del análisis.

### **3.3 ¿Cuales son los retos de la información y los datos que utilizara en la base de datos en terminos de la calidad y la consolidación? (Max 100 Palabras) - (Primera entrega)**

Los principales retos incluyen la calidad y la integridad de los datos obtenidos de la API de Spotify. Pueden existir registros incompletos o inconsistentes, así como variaciones en la precisión de las métricas de popularidad y características acústicas. Además, la consolidación de datos provenientes de diferentes fuentes puede resultar complicada, requiriendo un cuidadoso proceso de validación y limpieza para asegurar que el análisis sea fiable y significativo.

### **3.4 ¿Que espera de la utilización de un sistema de Bases de Datos para su proyecto? (Max 100 Palabras) - (Primera entrega)**

Se espera que la utilización de un sistema de bases de datos permita una gestión más eficiente y organizada de la información extraída de la API de Spotify. Esto facilitará el análisis de grandes volúmenes de datos, permitiendo realizar consultas complejas y obtener insights relevantes de manera rápida y precisa. Además, un sistema de bases de datos garantizará la integridad y consistencia de los datos, lo que es crucial para la validez de los resultados obtenidos en el análisis de la popularidad de las canciones.

## 4 Diseño del Modelo de Datos del SMBD (Sistema Manejador de Bases de Datos) (Primera entrega)

A continuación, se presenta el esquema de la base de datos relacional que almacena información sobre artistas, álbumes, pistas, géneros y sus relaciones. Se incluyen también índices y triggers para mejorar el rendimiento y la integridad de los datos.

### 4.1 Características del SMBD (Sistema Manejador de Bases de Datos) para el proyecto (Primera entrega)

Para este proyecto, se está utilizando Oracle como Sistema Manejador de Bases de Datos (SMBD). Oracle ofrece varias características clave:

- **Soporte de SQL estándar:** Compatible con las operaciones ANSI SQL.
- **Triggers y Procedimientos Almacenados:** Para automatizar tareas y realizar operaciones complejas.
- **Índices:** Aumentan el rendimiento en búsquedas y consultas.
- **Control de transacciones:** Garantiza la consistencia y la integridad de los datos.
- **Soporte de vistas, funciones y procedimientos:** Para encapsular y simplificar la lógica de la aplicación.

### 4.2 Diagrama modelo de datos (Primera entrega)

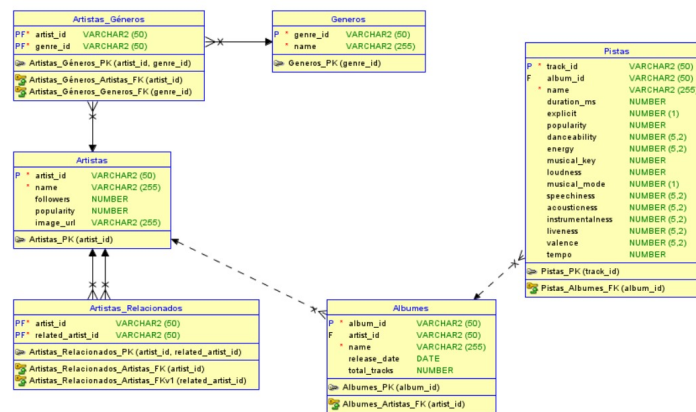


Figure 1: Diagrama representativo del esquema de base de datos



### 4.3 Imágenes de la Base de Datos (*Primera entrega*)

### 4.4 Código SQL - lenguaje de definición de datos (DDL) (*Primera entrega*)

```
1 CREATE TABLE Artistas (  
2     artist_id VARCHAR2(50) PRIMARY KEY,  
3     name VARCHAR2(255) NOT NULL,  
4     followers NUMBER,  
5     popularity NUMBER CHECK (popularity BETWEEN 0 AND 100),  
6     image_url VARCHAR2(255)  
7 );
```

Listing 1: Definición de la tabla de Artistas

```
1 CREATE TABLE Albumes (  
2     album_id VARCHAR2(50) PRIMARY KEY,  
3     artist_id VARCHAR2(50),  
4     name VARCHAR2(255) NOT NULL,  
5     release_date DATE,  
6     total_tracks NUMBER,  
7     FOREIGN KEY (artist_id) REFERENCES Artistas(artist_id)  
8     ON DELETE CASCADE  
9 );
```

Listing 2: Definición de la tabla de Álbumes

```
1 CREATE TABLE Pistas (  
2     track_id VARCHAR2(50) PRIMARY KEY,  
3     album_id VARCHAR2(50),  
4     name VARCHAR2(255) NOT NULL,  
5     duration_ms NUMBER,  
6     explicit NUMBER(1) CHECK (explicit IN (0, 1)),  
7     popularity NUMBER CHECK (popularity BETWEEN 0 AND 100),  
8     danceability NUMBER(5, 2) CHECK  
9     (danceability BETWEEN 0.0 AND 1.0),  
10    energy NUMBER(5, 2) CHECK (energy BETWEEN 0.0 AND 1.0),  
11    musical_key NUMBER,  
12    loudness NUMBER,  
13    musical_mode NUMBER(1) CHECK (musical_mode IN (0, 1)),  
14    speechiness NUMBER(5, 2)  
15    CHECK (speechiness BETWEEN 0.0 AND 1.0),  
16    acousticness NUMBER(5, 2)  
17    CHECK (acousticness BETWEEN 0.0 AND 1.0),  
18    instrumentalness NUMBER(5, 2)  
19    CHECK (instrumentalness BETWEEN 0.0 AND 1.0),  
20    liveness NUMBER(5, 2) CHECK (liveness BETWEEN 0.0 AND 1.0),  
21    valence NUMBER(5, 2) CHECK (valence BETWEEN 0.0 AND 1.0),  
22    tempo NUMBER,  
23    FOREIGN KEY (album_id) REFERENCES Albumes(album_id)  
24    ON DELETE CASCADE  
25 );
```

Listing 3: Definición de la tabla de Pistas

```

1 CREATE TABLE Generos (
2     genre_id VARCHAR2(50) PRIMARY KEY,
3     name VARCHAR2(255) NOT NULL
4 );

```

Listing 4: Definición de la tabla de Géneros

```

1 CREATE TABLE Artistas_Gneros (
2     artist_id VARCHAR2(50),
3     genre_id VARCHAR2(50),
4     PRIMARY KEY (artist_id, genre_id),
5     FOREIGN KEY (artist_id) REFERENCES Artistas(artist_id)
6         ON DELETE CASCADE,
7     FOREIGN KEY (genre_id) REFERENCES Generos(genre_id)
8         ON DELETE CASCADE
9 );

```

Listing 5: Definición de la tabla de relación Artistas-Géneros

```

1 CREATE TABLE Artistas_Relacionados (
2     artist_id VARCHAR2(50),
3     related_artist_id VARCHAR2(50),
4     PRIMARY KEY (artist_id, related_artist_id),
5     FOREIGN KEY (artist_id) REFERENCES Artistas(artist_id)
6         ON DELETE CASCADE,
7     FOREIGN KEY (related_artist_id) REFERENCES Artistas(artist_id)
8         ON DELETE CASCADE
9 );

```

Listing 6: Definición de la tabla de Artistas Relacionados

```

1 CREATE INDEX idx_popularity_artist ON Artistas (popularity);
2 CREATE INDEX idx_popularity_track ON Pistas (popularity);
3 CREATE INDEX idx_genero_name ON Generos (name);
4 CREATE INDEX idx_artistas_generos ON Artistas_Gneros (genre_id);
5 CREATE INDEX idx_artistas_relacionados ON
6     Artistas_Relacionados (related_artist_id);

```

Listing 7: Definición de índices para mejorar el rendimiento

El código anterior crea un esquema de base de datos para almacenar información sobre artistas, álbumes, pistas y géneros musicales. A continuación se describen brevemente sus elementos clave:

**Artistas:** Tabla que almacena información sobre los artistas, incluyendo su popularidad y número de seguidores.

**Álbumes:** Almacena los detalles de los álbumes de los artistas, como su nombre, fecha de lanzamiento y número de pistas.

**Pistas:** Contiene información sobre las canciones, incluyendo características musicales como duración, energía, y popularidad.

**Generos:** Define los géneros musicales asociados a los artistas.

**Relaciones entre tablas:** Las tablas Artistas\_Géneros y Artistas\_Relacionados manejan las relaciones muchos a muchos entre artistas y géneros, y entre artistas

entre sí, respectivamente.

**Índices:** Se crearon índices para mejorar el rendimiento en las consultas más frecuentes (popularidad de artistas, pistas y nombres de géneros).

**Triggers:** garantizan que, al eliminar un artista, también se eliminen las relaciones asociadas con géneros y otros artistas.

Este esquema busca optimizar el análisis y la manipulación de los datos musicales en el proyecto de investigación.

## 4.5 Código SQL - Manipulación de datos (DML) (*Primera entrega*)

```
1 INSERT INTO Artistas (artist_id, name, followers, popularity,
2   image_url)
3 VALUES ('A001', 'Taylor_Swift', 80000000, 95,
4   'https://image.taylorswift.com');
```

Listing 8: Inserción de un nuevo artista en la base de datos

### Actualización de Pista

```
1 UPDATE Pistas
2 SET popularity = 90
3 WHERE track_id = 'TR001';
```

Listing 9: Actualización de la popularidad de una pista

### Eliminación de Artista

```
1 DELETE FROM Artistas
2 WHERE artist_id = 'A001';
```

Listing 10: Eliminación de un artista de la base de datos

## 4.6 Código SQL + Resultados: Vistas (*Primera entrega*)

Ejemplos de vistas para consultas más complejas

Vista 1: Información de Álbumes y Artistas

```
1 CREATE VIEW Vista_Albumes_Artistas AS
2 SELECT
3     A.name AS artista,
4     AL.name AS album,
5     AL.release_date,
6     AL.total_tracks,
7     AVG(P.popularity) AS popularidad_promedio
8 FROM
9     Artistas A
10    JOIN Albumes AL ON A.artist_id = AL.artist_id
11    JOIN Pistas P ON AL.album_id = P.album_id
12 GROUP BY
13     A.name, AL.name, AL.release_date, AL.total_tracks;
```

Listing 11: Vista para información de álbumes y artistas

## Vista 2: Pistas Explícitas y Detalles del Álbum

```
1 CREATE VIEW Vista_Pistas_Explicitas AS
2 SELECT
3     A.name AS artista,
4     AL.name AS album,
5     P.name AS pista,
6     P.popularity,
7     P.explicit
8 FROM
9     Artistas A
10    JOIN Albumes AL ON A.artist_id = AL.artist_id
11    JOIN Pistas P ON AL.album_id = P.album_id
12 WHERE
13     P.explicit = 1;
```

Listing 12: Vista para pistas explícitas y detalles del álbum

## Vista 3: Artistas y Géneros Asociados

```
1 CREATE VIEW Vista_Artistas_Generos AS
2 SELECT
3     A.name AS artista,
4     G.name AS genero
5 FROM
6     Artistas A
7    JOIN Artistas_Generos AG ON A.artist_id = AG.artist_id
8    JOIN Generos G ON AG.genre_id = G.genre_id;
```

Listing 13: Vista para artistas y géneros asociados

## 4.7 Código SQL + Resultados: Triggers (*Primera entrega*)

```
1 CREATE OR REPLACE TRIGGER eliminar_relacion_artista_genero
2 AFTER DELETE ON Artistas
3 FOR EACH ROW
4 BEGIN
5     DELETE FROM Artistas_Generos
6     WHERE artist_id = :OLD.artist_id;
7 END;
```

Listing 14: Trigger para eliminar relaciones de género

### Trigger para eliminar relaciones entre artistas al eliminar un artista

```
1 CREATE OR REPLACE TRIGGER eliminar_relacion_artistas
2 AFTER DELETE ON Artistas
3 FOR EACH ROW
4 BEGIN
5     DELETE FROM Artistas_Relacionados
6     WHERE artist_id = :OLD.artist_id OR related_artist_id = :OLD.artist_id;
7 END;
```

Listing 15: Trigger para eliminar relaciones entre artistas

## 4.8 Código SQL + Resultados: Funciones (*Primera entrega*)

```
1 CREATE OR REPLACE FUNCTION CalcularEdadAlbum(album_fecha DATE)
2 RETURN NUMBER IS edad NUMBER;
3 BEGIN
4     edad := FLOOR(MONTHS_BETWEEN(SYSDATE, album_fecha) / 12);
5     RETURN edad;
6 END;
```

Listing 16: Función para calcular la edad de un álbum

Ejemplo de uso

```
1 SELECT name, CalcularEdadAlbum(release_date) AS edad
2 FROM Alumes;
```

Listing 17: Consulta para calcular la edad de un álbum

## 4.9 Código SQL + Resultados: procedimientos almacenados (*Primera entrega*)

```
1 CREATE OR REPLACE PROCEDURE ActualizarPopularidadArtista
2 (p_artist_id IN VARCHAR2) IS
3     v_promedio_popularidad NUMBER;
4 BEGIN
5     SELECT AVG(P.popularity)
6     INTO v_promedio_popularidad
7     FROM Alumes AL
8     JOIN Pistas P ON AL.album_id = P.album_id
9     WHERE AL.artist_id = p_artist_id;
10
11     UPDATE Artistas
12     SET popularity = v_promedio_popularidad
13     WHERE artist_id = p_artist_id;
14 END;
```

Listing 18: Procedimiento para actualizar popularidad

Ejemplo de uso

```
1 BEGIN
2     ActualizarPopularidadArtista('A001');
3 END;
```

Listing 19: Llamada al procedimiento

## 5 Bases de Datos No-SQL (*Segunda entrega*)

### 5.1 Diagrama Bases de Datos No-SQL (*Segunda entrega*)

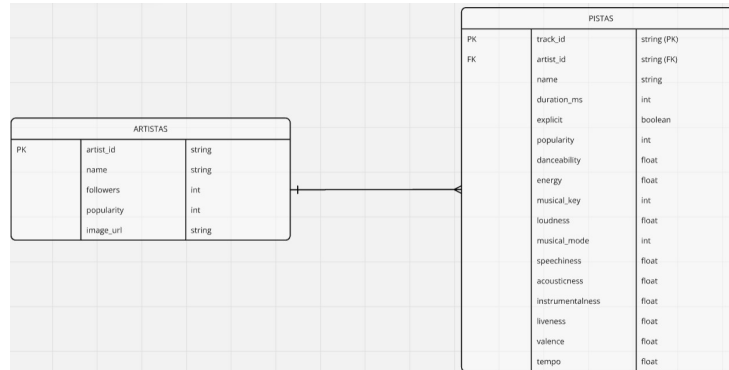


Figure 2: Diagrama base de datos No-SQL (Meta-Modelo Lógico)

El diagrama representa la estructura de un **Meta-Modelo Lógico** para una base de datos No-SQL diseñada para gestionar información de artistas musicales y sus canciones.

La colección **Artistas** contiene documentos que almacenan información como:

- Nombre del artista.
- Número de seguidores.
- Popularidad.
- URL de la imagen del artista.

Cada documento incluye un identificador único denominado **artist\_id**, que sirve para distinguir a cada artista dentro de la colección.

La colección **Pistas** almacena información sobre las canciones asociadas a los artistas, como:

- Nombre de la canción.
- Duración en milisegundos.
- Popularidad.
- Atributos musicales (por ejemplo: energía, acústica, tempo, entre otros).

Para establecer una relación entre las canciones y sus respectivos artistas, cada documento en la colección **Pistas** incluye un campo **artist\_id**, que coincide con el identificador correspondiente en la colección **Artistas**.

Este diseño refleja la flexibilidad y escalabilidad de un esquema No-SQL, permitiendo almacenar y relacionar datos jerárquicos sin las restricciones de las claves primarias o foráneas propias de los modelos relacionales.

## 5.2 SMBD utilizado para la Base de Datos No-SQL (*Segunda entrega*)

Para esta entrega, se utilizó **MongoDB** como Sistema Manejador de Bases de Datos No-SQL por su flexibilidad y capacidad de manejar grandes volúmenes de datos no estructurados. Sus características clave incluyen:

- Modelo basado en documentos que facilita la representación de datos en formato similar a JSON.
- Escalabilidad horizontal para gestionar datos de forma eficiente incluso con crecimiento continuo.
- Índices avanzados para realizar búsquedas rápidas.
- Alta disponibilidad gracias a los conjuntos de réplicas.
- Facilidad de integración con lenguajes modernos como Python para una manipulación ágil de los datos.

### Diseño del Sistema

El sistema se enfocó en almacenar información de dos artistas musicales: **J Balvin** y **Shakira**, junto con sus respectivas pistas musicales.

### Estructura de las colecciones

- **Observatorio\_Artists:** Contiene información detallada de los artistas, como su ID único, nombre, cantidad de seguidores, nivel de popularidad y enlace a su imagen.

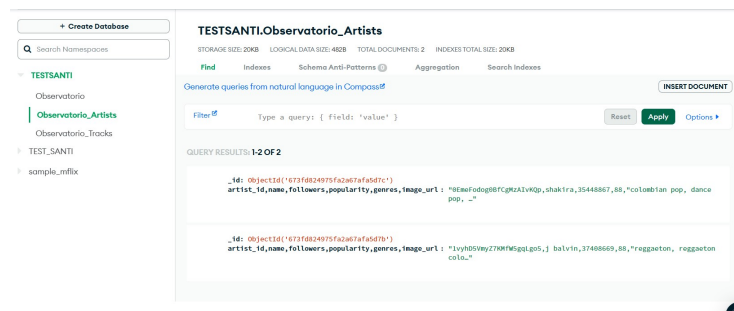


Figure 3: Observatorio\_Artists Mongoddb atlas

- **Observatorio\_Tracks:** Incluye datos relacionados con las pistas musicales de cada artista, como el ID único, duración, popularidad, y métricas técnicas como *danceability*, *energy* y *tempo*.

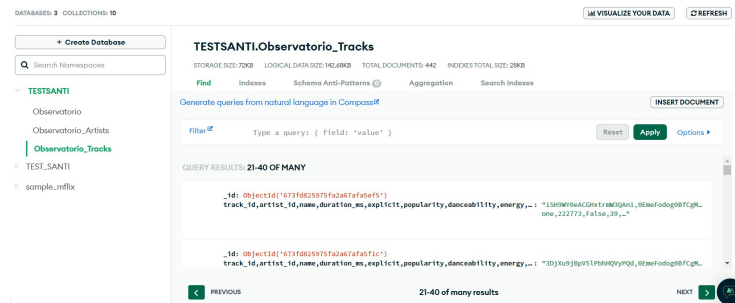


Figure 4: Observatorio\_Tracks Mongoddb atlas

## Relación entre las colecciones

Ambas colecciones están conectadas a través del campo `artist_id`, lo que permite identificar y vincular cada pista con su respectivo artista.

## Implementación Técnica

Se procesaron dos archivos CSV: uno con información sobre los artistas y otro con los datos de sus pistas musicales. Estos datos fueron cargados a las colecciones de MongoDB utilizando herramientas como **pandas** para el manejo de los archivos y **pymongo** para la conexión e inserción en MongoDB Atlas.

MongoDB permitió transformar los datos en documentos BSON y organizarlos en colecciones que reflejan las relaciones entre los artistas y sus pistas.

## Resultados

El sistema logró almacenar y relacionar información específica sobre **J Balvin** y **Shakira**, como sus métricas de popularidad y características de sus canciones. Las colecciones contienen:

- **Artistas:** Dos registros únicos con sus atributos relevantes.
- **Pistas musicales:** Una lista detallada de las canciones de cada artista, con sus respectivas métricas técnicas y datos de popularidad.

Esto permitió consultas ágiles para analizar la información y obtener estadísticas clave, como las canciones más populares o las métricas de energía y ritmo asociadas a sus pistas.

## Ventajas del Sistema No-SQL

- **Flexibilidad:** La capacidad de MongoDB para manejar datos heterogéneos permitió integrar métricas musicales y datos de artistas sin restricciones de esquemas.



- **Escalabilidad:** Su diseño permite incorporar más artistas o pistas sin impactar el rendimiento.
- **Rapidez en consultas:** El uso de índices optimizó la recuperación de datos de artistas y pistas específicos.
- **Integración ágil:** MongoDB se integró de forma efectiva con Python, permitiendo un desarrollo rápido y simplificado.
- **Alta disponibilidad:** Los conjuntos de réplicas garantizaron la accesibilidad continua a los datos.

## Conclusión base No-Sql

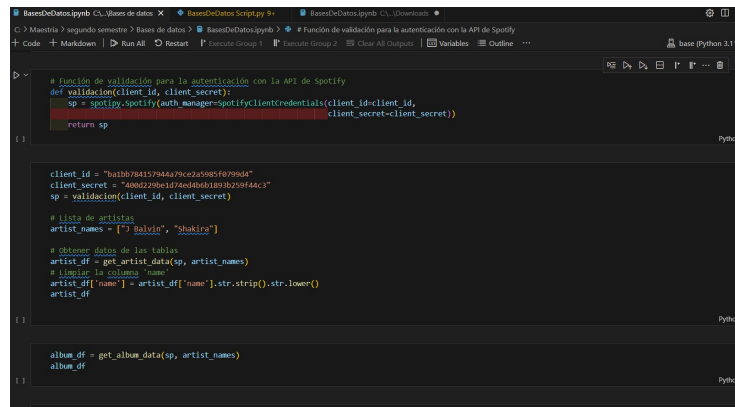
El uso de MongoDB para almacenar y analizar datos de **J Balvin** y **Shakira** permitió gestionar de forma eficiente tanto información básica como métricas avanzadas de sus pistas musicales. La flexibilidad y escalabilidad de este sistema lo convierten en una herramienta ideal para proyectos que requieren manejar datos dinámicos, facilitando el análisis y la toma de decisiones en el ámbito musical.

## 6 Aplicación de ETL (Extract, Transform, Load) y Bodega de Datos (*Tercera entrega*)

### 6.1 Ejemplo de aplicación de ETL y Bodega de Datos (*Tercera entrega*)

En este proyecto, el proceso ETL fue implementado mediante un script en Python que permitió extraer, transformar y cargar los datos en las plataformas seleccionadas: **MongoDB Atlas** y **Oracle Database**. La extracción de los datos se llevó a cabo conectándonos a la API de Spotify mediante el uso de la biblioteca **spotipy**. A través de esta herramienta, se autenticó el acceso y se realizaron solicitudes para obtener información detallada sobre artistas, canciones, géneros y álbumes. Los datos obtenidos fueron almacenados temporalmente en estructuras de datos de Python como listas y diccionarios, lo que facilitó su manipulación y procesamiento posterior.

La fase de transformación consistió en limpiar y estructurar los datos extraídos para garantizar consistencia y calidad. Se eliminaron registros duplicados, se completaron datos faltantes y se realizaron conversiones de formato, como la normalización de nombres de artistas y géneros y la transformación de fechas a formatos estándar. Además, se establecieron relaciones entre entidades como artistas y sus álbumes o canciones, creando un modelo que se ajustara tanto al esquema no relacional de MongoDB como al modelo relacional de Oracle.



```
# función de validación para la autenticación con la API de Spotify
def validacion(client_id, client_secret):
    sp = spotipy.Spotify(auth_manager=SpotifyClientCredentials(client_id=client_id,
                                                              client_secret=client_secret))
    return sp

client_id = "ba1bb70411704da79ce2a100f0709d4"
client_secret = "4000220be1d74ed4db0b1893b20f4ac3"
sp = validacion(client_id, client_secret)

# lista de artistas
artist_names = ["Halvin", "Shakira"]

# Obtener datos de las tablas
artist_df = get_artist_data(sp, artist_names)
# limpiar la columna "name"
artist_df["name"] = artist_df["name"].str.strip().str.lower()

album_df = get_album_data(sp, artist_names)
```

Figure 5: código utilizado ETL SQL API Spotify

Finalmente, en la fase de carga, los datos transformados fueron almacenados en **MongoDB Atlas** y en **Oracle Database**. En MongoDB, los datos se organizaron en colecciones específicas utilizando la biblioteca **pymongo**, lo que permitió manejar documentos JSON de forma eficiente. Para Oracle, los datos se cargaron en tablas previamente diseñadas, donde se definieron claves primarias y foráneas para mantener la integridad referencial. Este proceso se realizó mediante la biblioteca **cx\_Oracle**, que permitió establecer una conexión estable y gestionar tanto la creación como la inserción de registros.

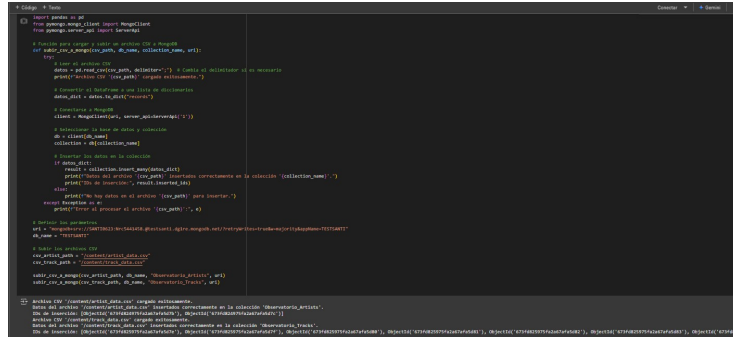


Figure 6: Carga de datos a observatorios MongoDB atlas

## 6.2 Automatización de Datos (*Tercera entrega*)

El proceso completo fue automatizado utilizando un archivo ejecutable **.bat**, diseñado para iniciar el script ETL directamente desde la línea de comandos en sistemas Windows. El archivo **.bat** contiene instrucciones claras para localizar y ejecutar el script de Python, asegurando que el entorno y las dependencias estén correctamente configurados. Para garantizar una ejecución periódica y sin intervención manual, se utilizó el programador de tareas de Windows, configurado para activar el archivo en intervalos específicos, como diariamente o semanalmente. Adicionalmente, se incorporó un sistema de generación de logs en el script, que permite registrar mensajes de estado y errores durante la ejecución, facilitando así el monitoreo y la depuración en caso de fallos.

## 6.3 Integración de Datos (*Tercera entrega*)

La integración de datos se logró mediante la combinación de dos plataformas complementarias. Por un lado, **MongoDB Atlas** se utilizó para almacenar datos semiestructurados, como documentos JSON, proporcionando flexibilidad y velocidad en consultas orientadas a la recuperación de información sobre artistas, géneros y álbumes. Por otro lado, **Oracle Database** fue utilizada para datos estructurados, lo que permitió realizar análisis avanzados y generar reportes basados en consultas SQL complejas que aprovechan las relaciones entre entidades. Este enfoque dual garantiza que los datos estén disponibles para diferentes tipos de análisis, maximizando su utilidad en aplicaciones prácticas.

## 7 Proximos pasos (*Tercera entrega*)

En cuanto a los próximos pasos, creo que sería ideal aplicar lo aprendido en proyectos personales que permitan experimentar más allá del aula. Esto nos dará práctica con las herramientas y conceptos vistos, además de ayudarnos a identificar problemas prácticos que podrían surgir en proyectos reales. Todo lo aprendido puede complementarse perfectamente con las áreas de análisis avanzado y manejo de grandes volúmenes de datos, conectando las herramientas y enfoques que estamos explorando, como en el proximo semestre con Inteligencia de Negocios y Big Data. De cara al proyecto de profundización o tesis, será clave definir qué motor de base de datos será el más adecuado para trabajar con los datos que se recolecten, ya sea con sistemas robustos como Oracle, SQL Server o incluso bases de datos no relacionales, dependiendo del tipo y volumen de la información. Lo importante no será solo almacenar datos, sino también procesarlos y analizarlos de manera eficiente para responder a las preguntas de investigación planteadas.

## 8 Lecciones aprendidas *(Tercera entrega)*

Sobre las lecciones aprendidas, aunque ya tenía experiencia previa con motores SQL, volver a los conceptos básicos ayudó a reforzar el entendimiento y a conectar la teoría con la práctica laboral. Muchas veces, en el día a día, nos enfocamos tanto en la ejecución que olvidamos la importancia de las bases. Este repaso permitió comprender mejor no solo cómo funcionan las bases de datos, sino también cómo optimizarlas. Por otro lado, explorar distintas herramientas y motores, tanto relacionales como no relacionales, mostró que no existe una única solución para todos los problemas, sino que cada uno tiene sus propias fortalezas dependiendo del caso. Además, las actualizaciones tecnológicas vistas en clase nos dejaron claro que este campo está en constante evolución, y mantenerse al día será esencial para aprovechar al máximo las oportunidades que estas herramientas ofrecen.

## 9 Bibliografía

Chodorow, K. (2013). *MongoDB: The definitive guide* (2nd ed.). O'Reilly Media.

Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of database systems* (7th ed.). Pearson.

McKinney, W. (2017). *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython* (2nd ed.). O'Reilly Media.

Spotify for Developers. (n.d.). *Spotify Web API documentation*. Retrieved November 2024, from <https://developer.spotify.com/documentation/web-api/>

Oracle. (n.d.). *Oracle database documentation*. Retrieved November 2024, from <https://docs.oracle.com/en/database/>

Allen, G. R. (2015). *Learning SQL* (2nd ed.). O'Reilly Media.