

## Entrega Final Proyecto Spotify

Anguie Garcia, Juan Sebastián Sanabria, Juan Sebastián Estupiñán

### 1. Descripción general de cada dataset.

Los dataset corresponden a artistas y canciones, provenientes de la plataforma de Spotify. El diccionario de datos de los dataset se expone a continuación.

Para el dataset de artistas:

Nombre	Tipo de dato	Rango	Concepto
id	object		Id del artista
followers	int64	0 a 7.890023e+07	Cantidad de seguidores
genres	object		Géneros musicales del artista
name	object		Nombre del artista
popularity	int64	0 a 1.000000e+02	Popularidad del artista

Para el dataset de canciones:

Nombre	Tipo de dato	Rango	Concepto
id	Objeto		El ID de Spotify para la pista.
name	object		Nombre de la canción.
popularity	float64	0 a 100	Popularidad de la canción.
duration_ms	int64	[3.344000e+03 a 5.621218e+06]	La duración de la pista en milisegundos.
explicit	int64	-1 a 1	
artists	object		Nombre artistas
Id_artists	object		Id de los artistas
Release_date	object		Fecha de lanzamiento de la canción
danceability	float64	0 a 0.991	La capacidad de baile describe qué tan adecuada es una pista para bailar (0,0 es menos bailable y 1,0 es más bailable)
energy	float64	0 a 1	Representa una medida perceptiva de intensidad y actividad.
key	int64	0 a 11	Indica la clave de la composición de la canción
loudness	float63	-60 a 0	El volumen general de una canción en decibelios (dB). Los valores de sonoridad se promedian en toda la pista y son útiles para comparar la sonoridad relativa de las pistas.

mode	int64	0 o 1	Indica la modalidad (mayor o menor) de una pista, el tipo de escala de la que se deriva su contenido melódico. Mayor está representado por 1 y menor es 0.
speechiness	float64	0 a 1	Es un indicador que detecta la presencia de palabras habladas en la canción, una canción en promedio tiene 0.33 de palabra y una de rap 0.66
acousticness	float64	0 a 1	Es una medida que se encarga de indicar si una canción es acústica o no
Instrumentalness	float 64	0 a 1	muestra si la canción es instrumental o no. Entre más cercano el valor a 1 es más probable que la canción no tenga voces.
Liveness	float64	0 a 1	muestra si la canción fue grabada con público. Entre más cercano el valor a 1 es más probable que la canción haya sido grabada con voces.
Valence	float64	0 a 1	Representa que tan positiva es la canción. Entre más cercano el valor a 1 la canción es más positiva.
Tempo	float64	0 a 246.381000	Dice cuantos beats por minuto (BPM) tiene la canción.
Time_signature	integer	0 a 5	Clasifica las canciones de acuerdo al compás en la que cada una de ellas se escribió.

## 2. Análisis de calidad de datos y procesos de limpieza implementados.

Como se puede observar en las tablas, los rangos de las variables coinciden con el soporte de Spotify.

Para el dataset de canciones:

- Dimensión de conformidad para fecha 'release date': Se realiza una prueba con la variable 'release date' para validar que todos datos estén conformes con la estructura año-mes-día, sin embargo, el 23% de estos datos no estaban completos, por lo que se elige tomar el dato del año únicamente para la solución propuesta, como una nueva columna y se expresa la variable como tipo de dato datetime con el argumento errors='coerce' para que cualquier valor que no se pueda convertir a una fecha se reemplace con un valor nulo (NaN) y garantizando la menor pérdida de información posible.
- Se valida que no existan duplicados en el ID del dataset.
- Se realiza un fillna(0) y dropna debido a la presencia de valores nulos en algunas variables, esto con el fin de garantizar que todos los datos existan para el análisis y el modelo.

## 3. Análisis exploratorio de datos.

### ■ Análisis univariados

Para el dataset de artistas:

- La popularidad de los artistas presenta asimetría positiva, lo que indica que la mayoría se concentra en los puntos más bajos de estas puntuaciones, por lo que es más probable que los artistas tengan puntuaciones bajas, de hecho, se observa una gran cantidad de artistas con puntuación cero.
- Como se observó en el diagrama de cajas y bigotes para la variable 'followers', la cantidad de seguidores por artista varía demasiado, los atípicos abundan en esta variable, lo que puede ser positivo en el momento de realizar clusters o grupos.

Para el dataset de canciones:

- Con el plot de barra de frecuencias por año, se puede apreciar que la cantidad de canciones 'sonadas' ha aumentado de forma significativa y no siempre supera el año inmediatamente anterior. Respecto al dataset, el 2020 es el año con más canciones.
- La duración de las canciones en milisegundos tiene una media de  $2.148930e+05$  ms, equivalente a 3.58 minutos aproximadamente, sin embargo, tal como se puede observar en el histograma y el boxlot generados, esta variable posee una gran cantidad de atípicos, lo que puede significar que dentro de las canciones pueden existir mixes de canciones que alcanzan una duración de hasta una hora.
- Se realizó un barplot, boxplot o histograma según la variable, con el fin de poder identificar el comportamiento de las variables, saber cómo está distribuida, si facilita realizar grupos de canciones de acuerdo a la variable. Además, vemos que las variables que son booleanas facilitan la segmentación y recomendación, las que tienen alto skewness también ayudan a crear segmentos.
- La frecuencia de popularidad de las canciones indica que son más las canciones que no son populares que las que sí, de hecho, resulta que las más populares resultan ser las atípicas, esto obedece al gran número de canciones que se suben en Spotify y no son famosas.

#### ■ Análisis bivariados

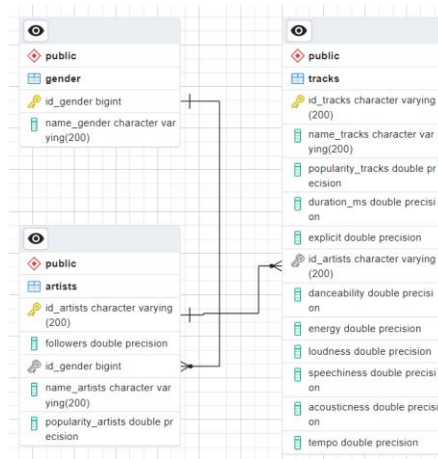
Para el dataset de artistas:

- Con la correlación por el método 'spearman' se encuentra una correlación significativa entre la popularidad del artista y la cantidad de seguidores que tiene.

Para el dataset de canciones:

- La gráfica de correlación generada indica una correlación positiva y 'grande' de aproximadamente 0.76 entre el loudness y energy, lo que parece tener sentido, además de una correlación significativa negativa de  $-0.72$  entre energy y acousticness. Adicionalmente se puede observar una correlación positiva de 0.53 entre las variables valence y danceability y una correlación negativa de  $-0.52$  entre las variables acousticness y loudness.
- En la gráfica 'Los dos artistas más sonados por año (Desde el 2000)' se puede ver que los artistas difieren por año, es decir, no hay un artista que sea el más sonado por más de un año, además, como se observa en la barra del año 2021, Justin Bieber es el artista con más número de veces que aparece.
- Con la gráfica 'canciones con mayor puntuación por año' se puede inferir que desde el año 2000, el top de las cinco canciones por año con mayor puntuación, en su mayoría corresponde a colaboraciones entre artistas, también conocido como 'featuring'.

#### 4. Modelo de datos relacional de la base de datos PostgreSQL.



En PostgreSQL se crean tres tablas, la tabla **artists** contiene el identificador de los artistas llamado *id\_artists*, el identificador de los géneros llamado *id\_gender* que se obtiene a partir de la toma de todos los géneros de la columna géneros del dataframe inicial y tomando el primer género de la lista. Estos datos se guardan en la tabla **gender**, compuesta por el identificador del género *id\_gender* y el nombre del género *name\_gender*. Adicionalmente la tabla artists contiene el nombre de los artistas *name\_artists* y su popularidad *popularity\_artists*. En la tabla **tracks** se asocian los datos del artista, y datos como el nombre de las canciones *name\_tracks*, su popularidad *popularity\_tracks* y otras variables como *duration\_ms*, *explicit*, *danceability*, *energy*, *loudness*, *speechiness*, *acousticness* y *tempo*, descritas en el inciso 1 de este informe. Claramente se observan las llaves primarias de la tabla gender: *id\_gender*, de la tabla artists : *id\_artists* con llave foránea en *id\_gender* y en la tabla tracks con llave foránea en *id\_tracks*.

## 5. Esquema de la bodega de datos en Bigquery

Al igual que en PostgreSQL, para Bigquery se las mismas tres tablas, artists, genders y tracks. Las tres tablas cuentan con las mismas características descritas previamente, es decir cuenta con los mismos campos y las mismas llaves primarias. A continuación, se puede observar los esquemas de las tres tablas en Bigquery.

**Esquema actual**

ADD POLICY TAG

Filtro Ingresar el nombre o el valor de la propiedad

Nombre del campo	Tipo	Modo	Interfaz	Valor predeterminado	Etiquetas de política	Descripción
id	STRING	NULLABLE		Valor predeterminado		Descripción: ID de los artistas
followers	FLOAT	NULLABLE		Valor predeterminado		Descripción: Cantidad de seguidores
genre_id	STRING	NULLABLE		Valor predeterminado		Descripción: ID de los géneros
name_art	STRING	NULLABLE		Valor predeterminado		Descripción: Nombre del artista
popularity	INTEGER	NULLABLE		Valor predeterminado		Descripción: Popularidad de la canción

Figura. Esquema de la tabla de artistas en Bigquery.

**Esquema actual**

ADD POLICY TAG

Filtro Ingresar el nombre o el valor de la propiedad

Nombre del campo	Tipo	Modo	Interfaz	Valor predeterminado	Etiquetas de política	Descripción
genre_id	INTEGER	NULLABLE		Valor predeterminado		Descripción: ID de los géneros
genre	STRING	NULLABLE		Valor predeterminado		Descripción: Nombre del género

Figura. Esquema de la tabla de géneros en Bigquery.

**Esquema actual**

ADD POLICY TAG

Filtro Ingresar el nombre o el valor de la propiedad

<input type="checkbox"/>	Nombre del campo	Tipo	Modo	Interfaz	Valor predeterminado	Etiquetas de políticas	Descripción
<input type="checkbox"/>	id	STRING	NULLABLE		Valor predeterminado		Descripción: Id de la canción
<input type="checkbox"/>	name_track	STRING	NULLABLE		Valor predeterminado		Descripción: Nombre de la canción
<input type="checkbox"/>	id_artist	STRING	NULLABLE		Valor predeterminado		Descripción: Id del artista
<input type="checkbox"/>	popularity	FLOAT	NULLABLE		Valor predeterminado		Descripción: Popularidad de la canción
<input type="checkbox"/>	duration_s	FLOAT	NULLABLE		Valor predeterminado		Descripción: Duración de la canción en segundos
<input type="checkbox"/>	explicit	INTEGER	NULLABLE		Valor predeterminado		Descripción: Si la canción tiene letra explícita
<input type="checkbox"/>	danceability	FLOAT	NULLABLE		Valor predeterminado		Descripción: Qué tan adecuada es una canción p
<input type="checkbox"/>	energy	FLOAT	NULLABLE		Valor predeterminado		Descripción: Representa una medida percceptiva
<input type="checkbox"/>	loudness	FLOAT	NULLABLE		Valor predeterminado		Descripción: El volumen general de una canción
<input type="checkbox"/>	speechiness	FLOAT	NULLABLE		Valor predeterminado		Descripción: Detecta la presencia de palabras ha
<input type="checkbox"/>	acousticness	FLOAT	NULLABLE		Valor predeterminado		Descripción: Muestra si la canción es acústica
<input type="checkbox"/>	tempo	FLOAT	NULLABLE		Valor predeterminado		Descripción: El tempo global estimado de una ca

Figura. Esquema de la tabla de canciones en Bigquery.

## 6. Arquitectura ETL

### Extracción

En primera medida, las tablas se extraen en Python usando la biblioteca pandas con la función `read_csv()` para leer el archivo CSV en un objeto `DataFrame`. De esta forma se leen los archivos *artists\_mode.csv* y *tracks\_mode.csv*, obteniendo dos `DataFrames` listos para su transformación.

### Transformación

Para la tabla artistas se realizan las siguientes transformaciones.

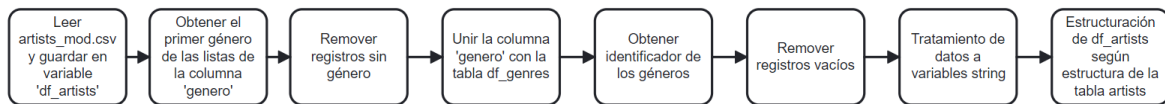


Figura. Transformación de datos de artista `df_artists`.

Para la tabla de canciones:

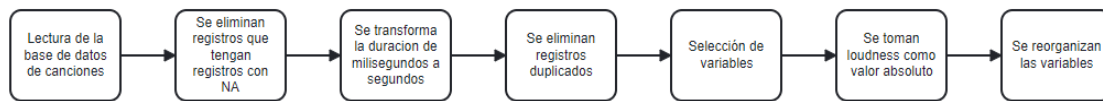


Figura. Transformación de datos de artista `df_tracks`.

### Carga

- **PostgreSQL:** A continuación, se exponen los pasos realizados para la carga de las tablas en la base de datos relacional.

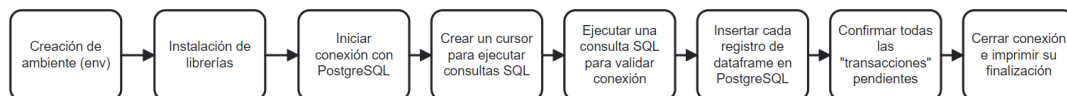


Figura. Esquema de pasos para carga de tablas en PostgreSQL.

- **BigQuery:** A continuación, se exponen los pasos realizados para la carga de las tablas en la base de datos relacional.

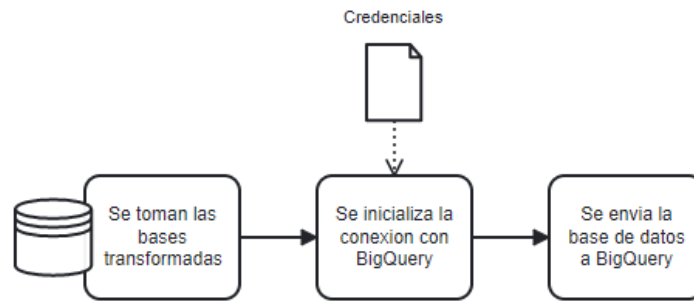


Figura. Esquema de pasos para carga de tablas en BigQuery.

## 7. Arquitectura general de la solución

De acuerdo al diagrama de bloques que se muestra en la siguiente figura, se pueden ver los diferentes componentes principales, procesos y resultados del proyecto. En primer lugar, están las dos bases de datos iniciales; una de artistas y otra de canciones. Luego de esto, se hizo un ETL con estas dos bases de datos originales dando como resultado 3 nuevas bases de datos con datos transformados y limpios. Estas tres bases de datos nuevas fueron el principal insumo para crear el dashboard, el modelo y un API.

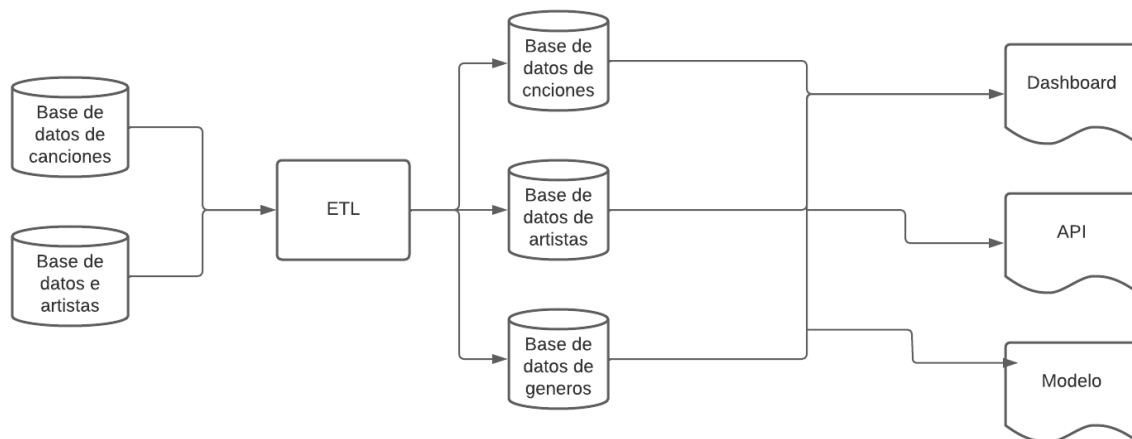


Figura. Diagrama de Bloques.

## 8. Descripción del sistema de recomendación

Las variables que usan utilizando para calcular la similitud entre canciones son "popularity", "duration\_ms", "explicit", "danceability", "energy", "loudness", "speechiness", "acousticness" y "tempo".

Para el modelo se usa la librería Annoy de Python, esta librería utiliza una técnica conocida como "árboles de búsqueda aproximada" para crear índices de búsqueda de alta velocidad en grandes conjuntos de datos.

Los hiperparámetros de Annoy son:

- **n\_trees**: número de árboles que se van a construir en el índice. Aumentar este valor aumenta la precisión, pero también aumenta el tiempo de construcción y el espacio en memoria, en este caso 1000.
- **metric**: métrica de distancia utilizada para calcular la distancia entre vectores. En este caso la métrica de distancia es la Euclidiana.
- **n\_jobs**: número de trabajos en paralelo para ejecutar durante la construcción del índice. El valor es 1, por lo que significa que se utilizarán todos recursos del computador.

A continuación, se exponen unos casos de prueba que muestran su funcionamiento.

Como se puede observar en la siguiente imagen se hizo una prueba con “Callaita” de Bad Bunny y se puede observar las primeras 13 canciones que el modelo recomienda con base al ingreso del nombre de esta canción.



```
tracks_df[['name', 'artists']].loc[neighbors]
```

	name	artists
91959	Callaita	['Bad Bunny', 'Tainy']
238448	Nada De Nada	['Marco di Mauro']
83858	Junto Al Amanecer	['J Alvarez']
114083	Monster	['Imagine Dragons']
455169	Harder Dan Ik Hebben Kan	['BLØF']
69880	River Deep, Mountain High	['Céline Dion']
583094	Hallelujah	['Molly Sandén']
404485	What About Now	['Daughtry']
187154	Knock On Wood	['Ami Stewart']
299969	Wooden Jesus	['Temple Of The Dog']
336175	ฝัน	['Lipta']
560969	ヒビカセ	['Reol']
329705	Querer Tu Amor	['Fama']

Figura. Prueba de funcionamiento

## 9. Descripción del Dashboard

Los indicadores y las gráficas implementadas de *géneros preferidos por artistas y canciones*, *canciones más populares en Spotify*, *Popularidad según baileabilidad* y *Popularidad según duración en ms*, son importantes para el negocio son importantes para el negocio de Spotify por varias razones:

- **Identificación de tendencias**: Al monitorear los géneros preferidos por los usuarios y las canciones más populares, Spotify puede identificar tendencias emergentes en la música y utilizar esta información para mejorar la experiencia del usuario al proporcionar contenido relevante y actualizado.

- **Personalización de la experiencia del usuario:** Al conocer los géneros y canciones preferidas de los usuarios, Spotify puede personalizar la experiencia del usuario recomendando contenido relevante basado en sus preferencias de música.
- **Generación de ingresos:** La popularidad de ciertas canciones y artistas también puede ser utilizada para aumentar los ingresos de Spotify. Por ejemplo, Spotify puede promocionar canciones populares a través de listas de reproducción destacadas y anuncios publicitarios, lo que puede generar más reproducciones y, por lo tanto, mayores ingresos para la plataforma, además de disminuir la ratio de usuarios que terminan su suscripción.

En resumen, los indicadores y las gráficas implementadas de géneros preferidos por artistas y canciones más populares en Spotify son importantes porque permiten a la plataforma mejorar la experiencia del usuario y generar mayores ingresos al proporcionar contenido relevante y actualizado basado en las preferencias de los usuarios. Para acceder de clic [Dashboard Spotify](#).

## 10. API POST

Se construyo una APP para que el usuario pueda recibir una recomendación de 20 canciones a partir de colocar el nombre y un artista de la canción, al enviar el artista y el nombre de la canción que le gustó, la aplicación lanzará el modelo de acuerdo con estos datos y devolverá una lista de 20 canciones que se recomienda que pueden llegar a ser de su agrado

## ANEXOS

Para la carga de los datos en PostgreSQL se tienen en cuenta la descarga e instalación del programa, el inicio del servidor, la creación de la base de datos y del usuario y la asignación de permisos como se expone en el siguiente esquema.

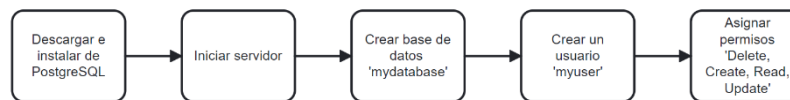


Figura. Pasos para creación de base de datos local.

Se crean las tablas SQL, teniendo en cuenta el modelo de datos relacional, para ello se nombran las variables y se declara el tipo de datos al que pertenece, además de las llaves primarias y llaves foráneas según corresponda.

```

Query  Query History
1  CREATE TABLE gender (
2    id_gender bigint PRIMARY KEY,
3    name_gender varchar(200)
4  );
5
6  CREATE TABLE Artists (
7    id_artists varchar(200) PRIMARY KEY,
8    followers float,
9    id_gender bigint,
10   FOREIGN KEY (id_gender) REFERENCES gender(id_gender),
11   name_artists varchar(200),
12   popularity_artists float
13 );
14
15 CREATE TABLE Tracks (
16   ID_tracks varchar(500) PRIMARY KEY,
17   name_tracks varchar(500),
18   popularity_tracks float,
19   duration_ms float,
20   explicit float,
21   id_artists varchar(200),
22   FOREIGN KEY (id_artists) REFERENCES Artists(id_artists),
23   danceability float,
24   energy float,
25   loudness float,
26   speechiness float,
27   acousticness float,
28   tempo float
29 );
  
```

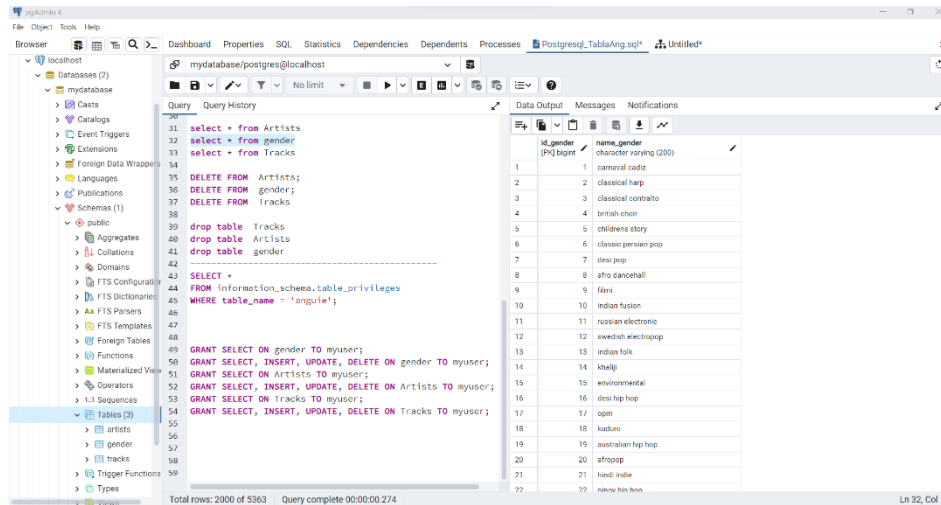


Figura. Creación de tablas en PostgreSQL.

La asignación de los permisos se realiza con los siguientes comandos.

```
GRANT SELECT ON gender TO myuser;  
GRANT SELECT, INSERT, UPDATE, DELETE ON gender TO myuser;  
GRANT SELECT ON Artists TO myuser;  
GRANT SELECT, INSERT, UPDATE, DELETE ON Artists TO myuser;  
GRANT SELECT ON Tracks TO myuser;  
GRANT SELECT, INSERT, UPDATE, DELETE ON Tracks TO myuser;
```

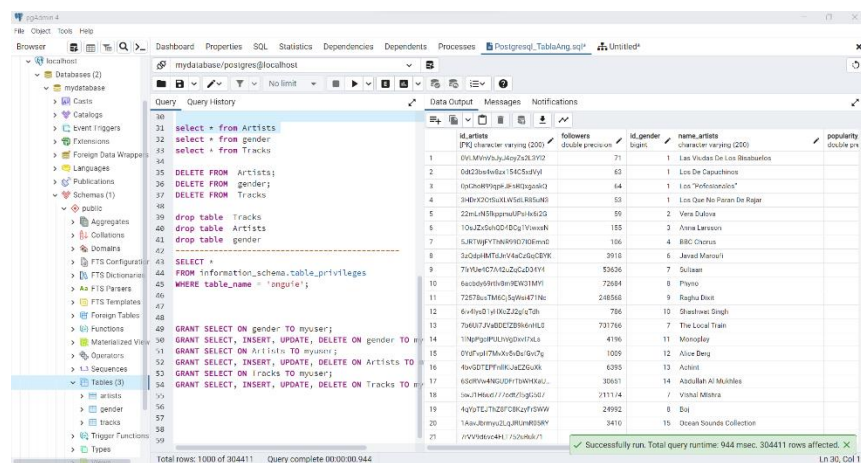
En la siguiente figura se observa la tabla **gender** con los registros cargados desde Python. Los registros corresponden a la tabla de géneros, contiene la columna `id_gender` y `name_gender`, que equivale al identificador y el concepto del género. En total se subieron 5363 géneros distintos.



id_gender	name_gender
1	canon
2	classical
3	classical crossover
4	british choir
5	children's story
6	classical persian pop
7	classical pop
8	afro dancehall
9	film
10	indian fusion
11	russian electronic
12	swedish electro
13	indian folk
14	khazip
15	environmental
16	classical hip hop
17	open
18	kuadru
19	australian hip hop
20	afropop
21	hindi indie
22	new hip hop

Figura. Registros de tabla **gender** en PostgreSQL.

En la siguiente figura se observa la tabla **Artists** con los registros cargados desde Python. Los registros corresponden a la tabla de artistas, contienen las columnas `id_artists`, `followers`, `id_gender`, `name_artists`, `popularity`. En total se cargó la información de 304411 artistas.



id_artists	followers	id_gender	name_artists	popularity
1	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
2	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
3	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
4	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
5	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
6	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
7	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
8	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
9	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
10	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
11	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
12	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
13	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
14	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
15	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
16	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
17	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
18	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
19	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
20	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000
21	0.0000000000000000	1	Los Videntes De Los Angeles	0.0000000000000000

Figura. Registros de tabla **Artists** en PostgreSQL.

En la siguiente figura se observa la tabla **Tracks** con los registros cargados desde Python. Los registros corresponden a la tabla de canciones, contienen las variables normalizadas de id,

name\_tracks, id\_artists, popularity, duration\_s, explicit, danceability, energy, loudness, speechiness, acousticness, tempo. En total se cargó la información de 267018 canciones.

The screenshot shows the PostgreSQL pgAdmin interface. The left sidebar displays the database structure, with 'Tracks' selected under 'Tables (3)'. The main window shows the SQL query editor with the following query:

```

select * from Artists;
select * from Gender;
select * from Tracks;
DELETE from Artists;
DELETE from Gender;
DELETE from Tracks;
drop table Tracks;
drop table Artists;
drop table Gender;
SELECT *
FROM information_schema.table_privileges
WHERE table_name = 'Tracks';

```

The right pane displays the query results in a table with the following columns: id, name\_tracks, popularity, tracks, duration, and cost. The results show 10 rows of data, including tracks like 'Daphnia 1.6 - Rongipara Anapaka', 'Vijaya Ganesa - Rameswar calio', and 'Lark of the Evening'.

The bottom status bar indicates: 'Successfully ran. Total query runtime: 3 secs 295 msec. 2670 rows affected. 10 rows selected.'

Figura. Registros de tabla **Tracks** en PostgreSQL.