**🎵 Digital Music Store SQL Project**

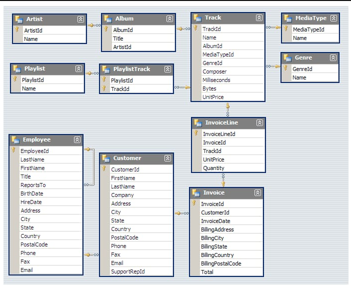
**Tools**: PostgreSQL, pgAdmin, SQL

**🎯 Project Goal**

Designed and implemented a relational database to simulate a digital music store environment. Imported multiple CSV files into PostgreSQL via pg Admin to populate core tables such as Artist, Album, Track, Playlist, Invoice, and Customer.

* Defined and enforced foreign key relationships to normalize data and maintain referential integrity.
* Developed and executed SQL queries to extract meaningful insights around customer behavior, sales performance, and genre popularity.
* Demonstrated full data life cycle capabilities — from ingestion and modeling to advanced data analysis and insights.

**SCHEMA OF DATA SET**:

****

**📊 Query Analysis Levels**

To show progression and skill growth, I categorized SQL queries into three levels of complexity:

* Beginner-Level Queries :

1. Who is the senior most employee based on job title?

2. Which countries have the most Invoices?

3. What are top 3 values of total invoice?

4. Which city has the best customers? We would like to throw a promotional Music Festival in the city we made the most money. Write a query that returns one city that has the highest sum of invoice totals. Return both the city name & sum of all invoice totals.

5. Who is the best customer? The customer who has spent the most money will be declared the best customer. Write a query that returns the person who has spent the most money.

* Intermediate-Level Queries:

1. Write query to return the email, first name, last name, & Genre of all Rock Music. listeners. Return your list ordered alphabetically by email starting with A.

2. Let's invite the artists who have written the most rock music in our data set. Write a query that returns the Artist name and total track count of the top 10 rock bands.

* Advanced-Level Queries:

1. Find how much amount spent by each customer on artists? Write a query to return customer name, artist name and total spent.

2. We want to find out the most popular music Genre for each country. We determine the most popular genre as the genre with the highest amount of purchases. Write a query that returns each country along with the top Genre. For countries where the maximum number of purchases is shared return all Genres.

**SQL Queries:**

* Beginner-Level Queries :

1. Who is the senior most employee based on job title?

SELECT \*   
 FROM Employee   
 ORDER BY levels DESC   
 LIMIT 1;

Output:  


2. Which countries have the most Invoices?

select count(\*) as c,billing\_country

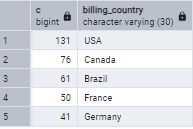
from invoice

group by billing\_country

order by c desc

limit 5;

Output:



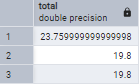
3. What are top 3 values of total invoice?

select total from invoice

order by total desc

limit 3;

Output:



4. Which city has the best customers? We would like to throw a promotional Music Festival in the city we made the most money. Write a query that returns one city that has the highest sum of invoice totals. Return both the city name & sum of all invoice totals.

select sum(total) as invoice\_total,billing\_city

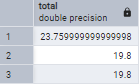
from invoice

group by billing\_city

order by invoice\_total desc

limit 3;

Output:



5. Who is the best customer? The customer who has spent the most money will be declared the best customer. Write a query that returns the person who has spent the most money.

select customer.customer\_id,customer.first\_name,customer.last\_name,sum(invoice.total) as total

from customer

join invoice on customer.customer\_id = invoice.customer\_id

group by customer.customer\_id

order by total desc

limit 1;

Output:



* Intermediate-Level Queries:

1. Write query to return the email, first name, last name, & Genre of all Rock Music. listeners. Return your list ordered alphabetically by email starting with A.

select distinct email,first\_name,last\_name

from customer

join invoice on customer.customer\_id = invoice.customer\_id

join invoice\_line on invoice.invoice\_id = invoice\_line.invoice\_id

where track\_id in(

select track\_id from track

join genre on track.genre\_id=genre.genre\_id

where genre.name like 'A%'

)

order by email

limit 5;

Output:



2. Let's invite the artists who have written the most rock music in our data set. Write a query that returns the Artist name and total track count of the top 10 rock bands.

select artist.artist\_id,artist.name,count(track.track\_id) as no\_of\_songs

from track

join album on album.album\_id = track.album\_id

join artist on artist.artist\_id=album.artist\_id

join genre on genre.genre\_id =track.genre\_id

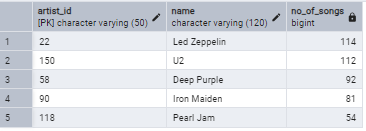
where genre.name ilike 'rock'

group by artist.artist\_id

order by no\_of\_songs desc

limit 5;

Output:



3. Return all the track names that have a song length longer than the average song length. Return the Name and Milliseconds for each track. Order by the song length with the longest songs listed first.

select name,milliseconds

from track

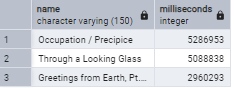
where milliseconds > (

select avg(milliseconds) as avg\_track\_length from track)

order by milliseconds desc

Limit 3;

Output:



* Advanced-Level Queries:

1. Find how much amount spent by each customer on artists? Write a query to return customer name, artist name and total spent.

WITH best\_selling\_artist AS (

SELECT

ar.artist\_id,

ar.name AS artist\_name,

SUM(il.unit\_price \* il.quantity) AS total\_sales

FROM invoice\_line il

JOIN track t ON t.track\_id = il.track\_id

JOIN album al ON al.album\_id = t.album\_id

JOIN artist ar ON ar.artist\_id = al.artist\_id

GROUP BY ar.artist\_id, ar.name

ORDER BY total\_sales DESC

LIMIT 1

)

SELECT

c.customer\_id,

c.first\_name,

c.last\_name,

bsa.artist\_name,

SUM(il.unit\_price \* il.quantity) AS amount\_spent

FROM invoice i

JOIN customer c ON c.customer\_id = i.customer\_id

JOIN invoice\_line il ON il.invoice\_id = i.invoice\_id

JOIN track t ON t.track\_id = il.track\_id

JOIN album alb ON alb.album\_id = t.album\_id

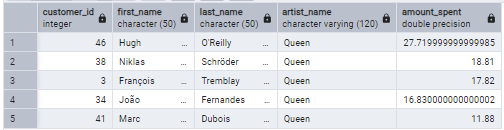
JOIN best\_selling\_artist bsa ON bsa.artist\_id = alb.artist\_id -- Filter only best-selling artist

GROUP BY c.customer\_id, c.first\_name, c.last\_name, bsa.artist\_name

ORDER BY amount\_spent DESC

limit 5;

Output:



2. We want to find out the most popular music Genre for each country. We determine the most popular genre as the genre with the highest amount of purchases. Write a query that returns each country along with the top Genre. For countries where the maximum number of purchases is shared return all Genres.

with popular\_genre as(

selectcount(invoice\_line.quantity)as purchase,customer.country,genre.name,genre.genre\_id,

row\_number()over(partition by customer.country order by count(invoice\_line.quantity)desc) as Rowno

from invoice\_line

join invoice on invoice.invoice\_id=invoice\_line.invoice\_id

join customer on customer.customer\_id=invoice.customer\_id

join track on track.track\_id = invoice\_line.track\_id

join genre on genre.genre\_id=track.genre\_id

group by 2,3,4

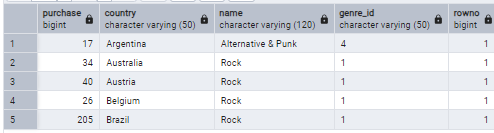
order by 2 asc,1 desc

)

select \* from popular\_genre where Rowno<=1

limit 5;

Output:



This project enhanced my SQL skills through real-world data analysis.  
I extracted key insights on customer spending, sales trends, and genre popularity.  
It demonstrates my ability to manage data from import to insight generation.