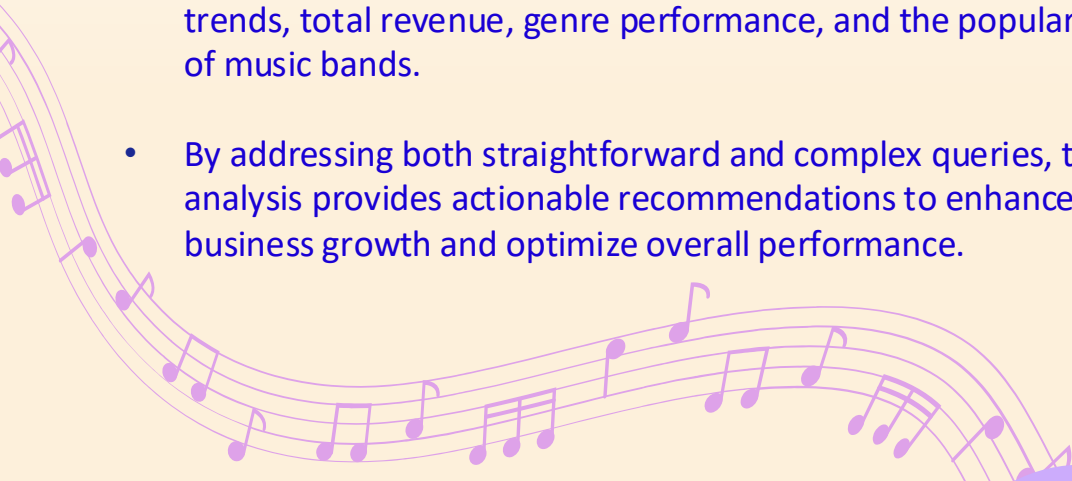


# Music Store Analysis

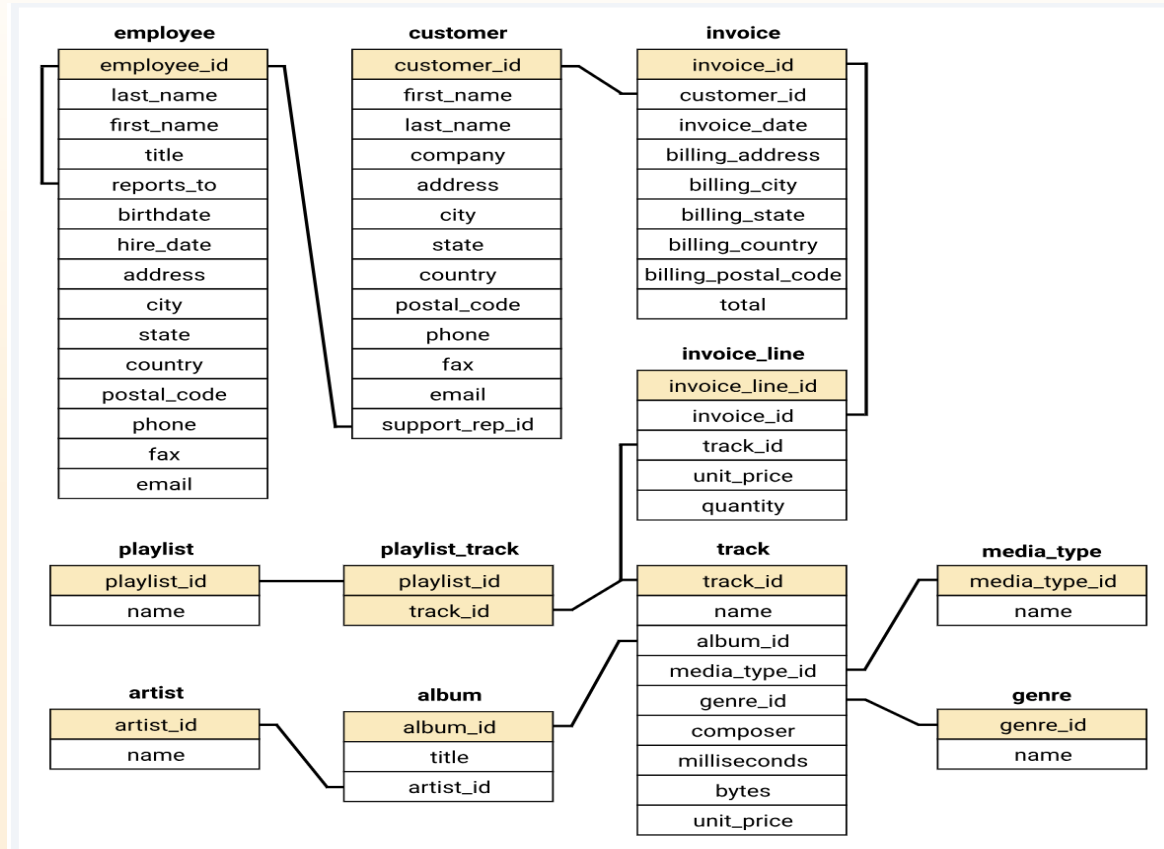


# Objective

- This project aims to analyse a digital music store database using SQL, with a focus on delivering valuable insights to stakeholders for informed decision-making.
- Through comprehensive SQL queries, the project explores various aspects, including geographic expansion, purchasing trends, total revenue, genre performance, and the popularity of music bands.
- By addressing both straightforward and complex queries, the analysis provides actionable recommendations to enhance business growth and optimize overall performance.



# Database Schema



# Query Complexity Levels

01

## Easy

Select  
Group By  
Order By  
Limit  
Asc/Desc

02

## Moderate

Joins  
Order By  
Group By  
Limits

03

## Advanced

CTE (Common Table  
Expressions)  
Recursion





1

Easy



**Q1.** Who is the senior most employee based on job title?

## Query

```
/* Q1: Who is the senior most employee based on job title? */  
SELECT title, last_name, first_name FROM employee  
ORDER BY levels DESC  
LIMIT 1 --this will only show one row from the top--
```

## Output

	title character varying (50) 🔒	last_name character (50) 🔒	first_name character (50) 🔒
1	Senior General Manager	Madan	Mohan

## Query

```
/* Q2: Which countries have the most Invoices? */  
SELECT COUNT(*) AS c, billing_country FROM invoice  
GROUP BY billing_country  
ORDER BY c DESC
```

## Output

	c bigint	billing_country character varying (30)
1	131	USA
2	76	Canada
3	61	Brazil
4	50	France
5	41	Germany
6	30	Czech Republic
7	29	Portugal
8	28	United Kingdom
9	21	India
10	13	Chile

Total rows: 24 of 24    Query complete 00



**Q2.** Which  
countries have the  
most Invoices?


Q3. What are top 3  
values of total  
invoice



### Query

```
SELECT total FROM invoice
ORDER BY total DESC
limit 3
```

### Output

	total double precision 
1	23.759999999999998
2	19.8
3	19.8



**Q4.** 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

## Query

```
SELECT billing_city, SUM(total) AS InvoiceTotal FROM invoice
GROUP BY billing_city
ORDER BY InvoiceTotal DESC
```

## Output

	billing_city character varying (30)	invoicetotal double precision
1	Prague	273.24000000000007
2	Mountain View	169.29
3	London	166.32
4	Berlin	158.4
5	Paris	151.47
6	São Paulo	129.69
7	Dublin	114.83999999999997
8	Delhi	111.86999999999999
9	São José dos Campos	108.89999999999998
10	Brasília	106.91999999999999

## Query

```
SELECT customer.customer_id, first_name, last_name, SUM(total) AS total_spending
FROM customer
JOIN invoice ON customer.customer_id = invoice.customer_id
GROUP BY customer.customer_id
ORDER BY total_spending DESC
LIMIT 1;
```

## Output

	customer_id [PK] integer	first_name character (50)	last_name character (50)	total_spending double precision
1	5	R	Madhav	144.54000000000002



**Q5.** 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

2

**Moderate**



**Q1.** 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



## Query

```
SELECT DISTINCT email AS Email, first_name AS FirstName, last_name AS LastName, genre.name AS Name
FROM customer
JOIN invoice ON invoice.customer_id = customer.customer_id
JOIN invoice_line ON invoice_line.invoice_id = invoice.invoice_id
JOIN track ON track.track_id = invoice_line.track_id
JOIN genre ON genre.genre_id = track.genre_id
WHERE genre.name LIKE 'Rock'
ORDER BY email;
```

## Output

	email character varying (50)	firstname character (50)	lastname character (50)	name character varying (120)
1	aaronmitchell@yahoo.ca	Aaron	Mitchell	Rock
2	alero@uol.com.br	Alexandre	Rocha	Rock
3	astrid.gruber@apple.at	Astrid	Gruber	Rock
4	bjorn.hansen@yahoo.no	Bjørn	Hansen	Rock
5	camille.bernard@yahoo.fr	Camille	Bernard	Rock
6	daan.peeters@apple.be	Daan	Peeters	Rock
7	diego.gutierrez@yahoo.ar	Diego	Gutiérrez	Rock
8	dmiller@comcast.com	Dan	Miller	Rock
9	dominiquelefebvre@gmail.c...	Dominique	Lefebvre	Rock
10	edfrancis@yahoo.ca	Edward	Francis	Rock

Total rows: 59 of 59    Query complete 00:00:00.237    Ln 59, Col 16

## Query

```
SELECT artist.artist_id, artist.name, COUNT(artist.artist_id) AS number_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 LIKE 'Rock'
GROUP BY artist.artist_id
ORDER BY number_of_songs DESC
LIMIT 10;
```

## Output

	artist_id [PK] character varying (50) 	name character varying (120) 	number_of_songs bigint 
1	22	Led Zeppelin	114
2	150	U2	112
3	58	Deep Purple	92
4	90	Iron Maiden	81
5	118	Pearl Jam	54
6	152	Van Halen	52
7	51	Queen	45
8	142	The Rolling Stones	41
9	76	Creedence Clearwater Revival	40
10	52	Kiss	35

✧ **Q2.** Let's invite the artists who have written the most rock music in our dataset. Write a query that returns the Artist name and total track count of the top 10 rock bands

**Q3.** 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

## Query

```
SELECT name, milliseconds
FROM track
WHERE milliseconds > (
    SELECT AVG(milliseconds) AS avg_track_length
    FROM track )
ORDER BY milliseconds DESC;
```

## Output

	name character varying (150)	milliseconds integer
1	Occupation / Precipice	5286953
2	Through a Looking Glass	5088838
3	Greetings from Earth, Pt. 1	2960293
4	The Man With Nine Lives	2956998
5	Battlestar Galactica, Pt. 2	2956081
6	Battlestar Galactica, Pt. 1	2952702
7	Murder On the Rising Star	2935894
8	Battlestar Galactica, Pt. 3	2927802
9	Take the Celestra	2927677
10	Fire In Space	2926593
Total rows: 494 of 494    Query complete 00:00:00.153    Ln 80, Col 28		

3

Advanced



**Q1.** Find how much amount spent by each customer on artists? Write a query to return customer name, artist name and total spent.



## Query

```
WITH best_selling_artist AS ( --temporary table with cte--
    SELECT artist.artist_id AS artist_id, artist.name AS artist_name, SUM(invoice_line.unit_price*invoice_line.quantity) AS total_sales
    FROM invoice_line
    JOIN track ON track.track_id = invoice_line.track_id
    JOIN album ON album.album_id = track.album_id
    JOIN artist ON artist.artist_id = album.artist_id
    GROUP BY 1 --col no--
    ORDER BY 3 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
GROUP BY 1,2,3,4
ORDER BY 5 DESC;
```

## Output

	customer_id integer	first_name character (50)	last_name character (50)	artist_name character varying (120)	amount_spent double precision
1	46	Hugh	O'Reilly	Queen	27.719999999999985
2	38	Niklas	Schröder	Queen	18.81
3	3	François	Tremblay	Queen	17.82
4	34	João	Fernandes	Queen	16.830000000000002
5	53	Phil	Hughes	Queen	11.88
6	41	Marc	Dubois	Queen	11.88
7	47	Lucas	Mancini	Queen	10.89
8	33	Ellie	Sullivan	Queen	10.89

Total rows: 43 of 43    Query complete 00:00:00.128    Ln 110, Col 17



## Query

```
WITH popular_genre AS
(
    SELECT COUNT(invoice_line.quantity) AS purchases, 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
```

## Output

	purchases bigint	country character varying (50)	name character varying (120)	genre_id character varying (50)	rowno bigint
1	17	Argentina	Alternative & Punk	4	1
2	34	Australia	Rock	1	1
3	40	Austria	Rock	1	1
4	26	Belgium	Rock	1	1
5	205	Brazil	Rock	1	1
6	333	Canada	Rock	1	1
7	61	Chile	Rock	1	1
8	143	Czech Republic	Rock	1	1

Total rows: 24 of 24    Query complete 00:00:00.136    Ln 132, Col 45



**Q2.** 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

**Q3.** Write a query that determines the customer that has spent the most on music for each country. Write a query that returns the country along with the top customer and how much they spent. For countries where the top amount spent is shared, provide all customers who spent this amount



## Query

```
WITH RECURSIVE
  customer_with_country AS (
    SELECT customer.customer_id, first_name, last_name, billing_country, SUM(total) AS total_spending
    FROM invoice
    JOIN customer ON customer.customer_id = invoice.customer_id
    GROUP BY 1,2,3,4
    ORDER BY 2,3 DESC),

  country_max_spending AS(
    SELECT billing_country, MAX(total_spending) AS max_spending
    FROM customer_with_country
    GROUP BY billing_country)

SELECT cc.billing_country, cc.total_spending, cc.first_name, cc.last_name, cc.customer_id
FROM customer_with_country cc
JOIN country_max_spending ms
ON cc.billing_country = ms.billing_country
WHERE cc.total_spending = ms.max_spending
ORDER BY 1;
```

## Output

	billing_country character varying (30)	total_spending double precision	first_name character (50)	last_name character (50)	customer_id integer
1	Argentina	39.6	Diego	Gutiérrez	56
2	Australia	81.18	Mark	Taylor	55
3	Austria	69.3	Astrid	Gruber	7
4	Belgium	60.38999999999999	Daan	Peeters	8
5	Brazil	108.89999999999998	Luis	Gonçalves	1
6	Canada	99.99	François	Tremblay	3
7	Chile	97.02000000000001	Luis	Rojas	57
8	Czech Republic	144.54000000000002	R	Madhav	5
Total rows: 24 of 24    Query complete 00:00:00.132    Ln 198, Col 12					

**Thank  
You!**

