



Sanam Teri Kasam
(Original Motion Picture Soundtrack)
Himesh Reshammiya,
Sameer Anjaan, Subrat...



Aashiqui 2
Mohammed Rafi, Alka Yagnik,
Udit Narayan, Alka Yagnik



Yeh Jawaani Hai Deewani
Sonu Nigam, Alka Yagnik,
Pritam



Glory
Yo Yo Honey Singh



Who (Remixes)
JMN MUSE



Mismatched: Season 3
(Soundtrack from the TV Series)
Mismatched - Cast

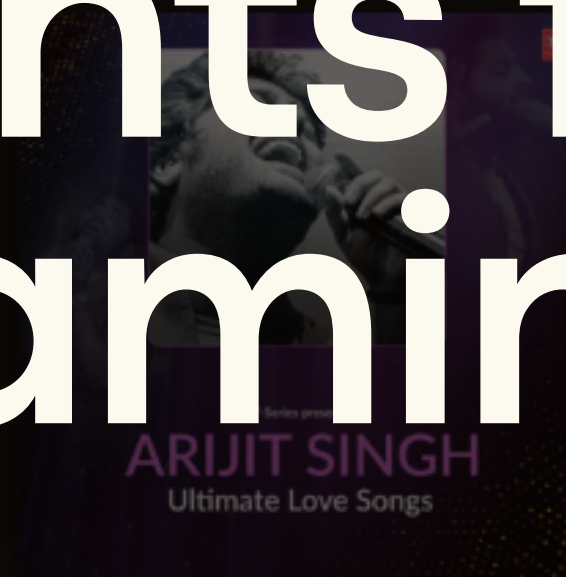
SQL-Based Data Insights for a Music Streaming App



Sicario
Shubh



Young G.O.A.T.
Cheema Y, Gur Sidhu



Ultimate Love Songs -
Arijit Singh
Arijit Singh



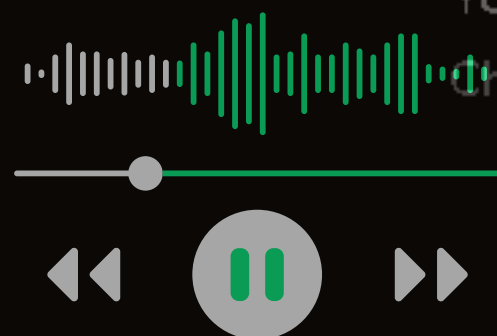
Sanam Teri Kasam
(Original Motion Picture Soundtrack)
Himesh Reshammiya



Making Memories
Karan Aujla, Ikky



Hurry Up Tomorrow
The Weeknd



Introduction

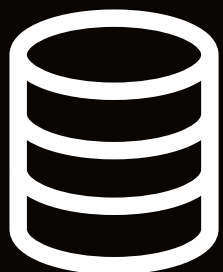
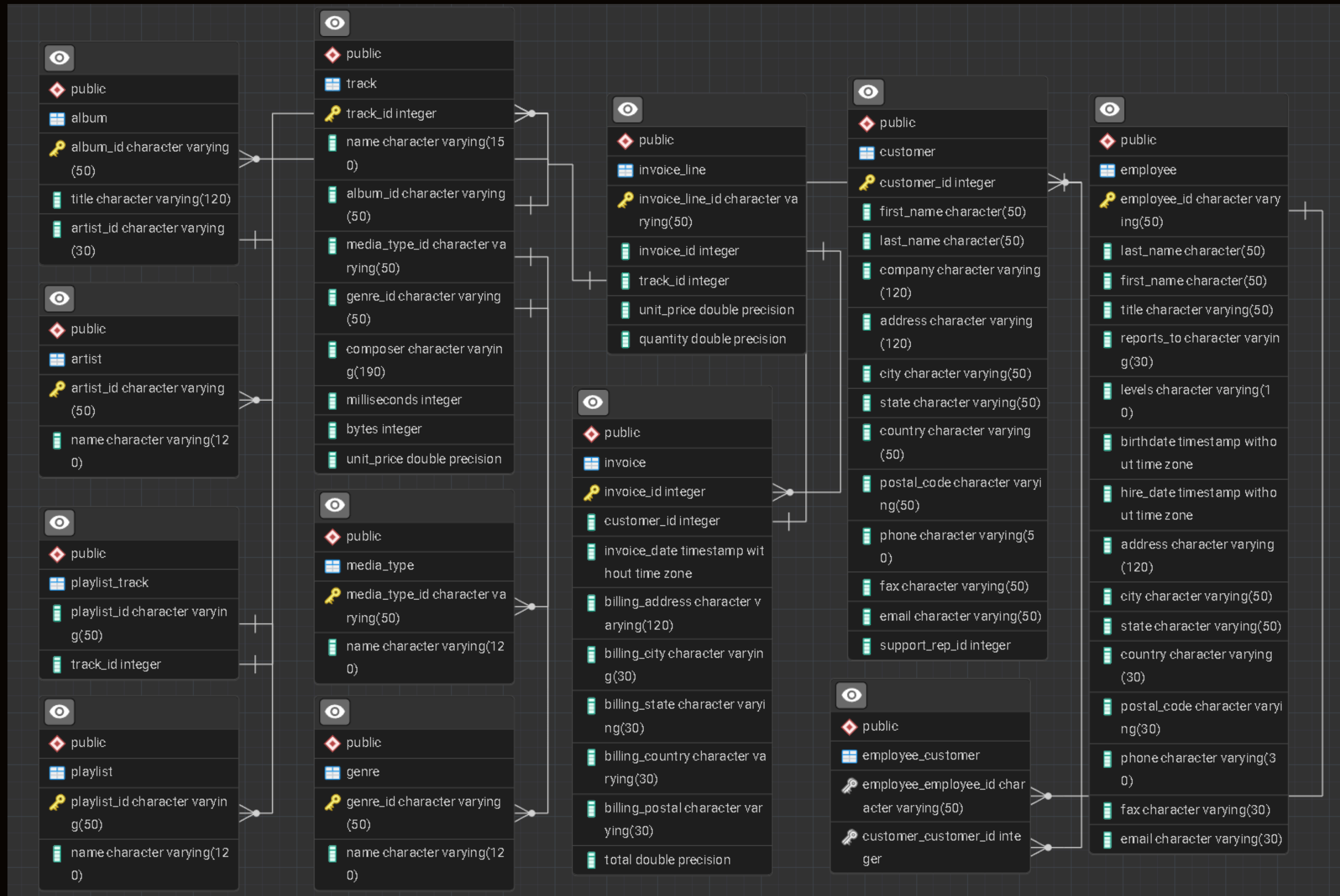
Hello, my name is Firdous Rahmani, and this project explores the power of SQL in analyzing a music streaming service's database. In today's digital music industry, data-driven insights are essential for understanding customer behavior, optimizing sales strategies, and enhancing user experience. This project utilizes SQL to analyze a music streaming service's database, extracting valuable insights into key business metrics.

Through structured queries, this analysis uncovers:

- Top customers and revenue-generating cities
- Most popular music genres by country
- Best-selling artists and their top listeners
- Invoice trends and customer spending behavior



Database Schema



Which countries have the most invoices?

```
select  count(*) as total , billing_country
from invoice
group by billing_country
order by total desc
```

| | total 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 |
| 11 | 13 | Ireland |
| 12 | 11 | Spain |
| 13 | 11 | Finland |



Which city has the best customers? We would like to throw a promotional music festival in the city where we made the most money. Write a query that returns the one city that has the highest sum of invoice totals. Return both 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 1;
```

| | invoice_total double precision | billing_city character varying (30) |
|---|-----------------------------------|--|
| 1 | 273.2400000000000007 | Prague |



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
```

| | customer_id [PK] integer | first_name character (50) | last_name character (50) | total double precision |
|---|-----------------------------|------------------------------|-----------------------------|---------------------------|
| 1 | 5 | R... | Madhav... | 144.540000000000002 |



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 = 'Rock')

order by email;
```

| | email character varying (50) | first_name character (50) | last_name character (50) |
|---|---------------------------------|------------------------------|-----------------------------|
| 1 | aaronmitchell@yahoo.ca | Aaron | Mitchell |
| 2 | alero@uol.com.br | Alexandre | Rocha |
| 3 | astrid.gruber@apple.at | Astrid | Gruber |
| 4 | bjorn.hansen@yahoo.no | Bjørn | Hansen |
| 5 | camille.bernard@yahoo.fr | Camille | Bernard |
| 6 | daan_peeters@apple.be | Daan | Peeters |
| 7 | diego.gutierrez@yahoo.ar | Diego | Gutiérrez |
| 8 | dmiller@comcast.com | Dan | Miller |
| 9 | dominique.lefebvre@gmail.com | Dominique | Le febvre |



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(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 = 'Rock'
group by artist.artist_id
order by number_of_songs DESC
limit 10;
```

| | 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 |



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 average_track_length from track)
order by milliseconds DESC;
```

| | 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 |



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

```
WITH best_selling_artist AS (
    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
    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;
```

| | customer_id integer | first_name character (50) | last_name character (50) | artist_name character varying | 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 |
| 9 | 29 | Don | Miller | Queen | 9.96 |



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 Genre

```
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
```

| | purchases bigint | country character varying | name character varying (120) | genre_id character | 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 |
| 9 | 24 | Denmark | Rock | 1 | 1 |
| 10 | 46 | Finland | Rock | 1 | 1 |
| 11 | 211 | France | Rock | 1 | 1 |

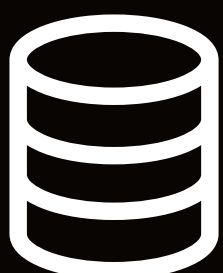


Write a query that determines the customer that has spent the most on music for each country. Write a query that returns the country along with the top customer and how much they spent. For countries where the top amount spent is shared , provide all cusyomers who spent this amount.

```
With Customer_with_country As (
    SELECT customer.customer_id , first_name, last_name, billing_country, SUM(total) AS total_spending,
    ROW_NUMBER() OVER(PARTITION BY billing_country ORDER BY SUM(total)DESC) AS RowNo
    FROM invoice
    JOIN customer on customer.customer_id = invoice.customer_id
    GROUP BY 1,2,3,4
    order by 4 ASC , 5 DESC)






select * from Customer_with_country where RowNo <= 1
```

| | customer_id integer | first_name character (50) | last_name character (50) | billing_country character varying | total_spending double precision | rowno bigint |
|----|------------------------|------------------------------|-----------------------------|--------------------------------------|------------------------------------|-----------------|
| 1 | 56 | Diego | Gutiérrez | Argentina | 39.6 | 1 |
| 2 | 55 | Mark | Taylor | Australia | 81.18 | 1 |
| 3 | 7 | Astrid | Gruber | Austria | 69.3 | 1 |
| 4 | 8 | Daan | Peeters | Belgium | 60.38999999999999 | 1 |
| 5 | 1 | Luís | Gonçalves | Brazil | 108.89999999999998 | 1 |
| 6 | 3 | François | Tremblay | Canada | 99.99 | 1 |
| 7 | 57 | Luis | Rojas | Chile | 97.020000000000001 | 1 |
| 8 | 5 | R | Madhav | Czech Republic | 144.540000000000002 | 1 |
| 9 | 9 | Kara | Nielsen | Denmark | 37.61999999999999 | 1 |
| 10 | 44 | Terhi | Hämäläinen | Finland | 79.2 | 1 |
| 11 | 42 | Wyatt | Girard | France | 99.99 | 1 |
| 12 | 37 | Eynn | Zimmermann | Germany | 94.050000000000001 | 1 |

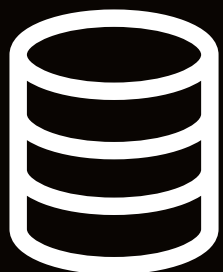


Conclusion

Key Takeaways from the Insights :

-  **Top-Spending Customers:** Identified the most valuable customers in different regions.
-  **Best Revenue-Generating City:** Determined the city that contributes the most sales.
-  **Most Popular Music Genre per Country:** Found the dominant genre based on purchase data.
-  **Top Rock Artists:** Recognized the bands with the highest number of tracks.
-  **Best-Selling Artists:** Discovered which artists generate the most revenue.

These insights provide actionable business intelligence, allowing music platforms to tailor their marketing efforts, target the right customers, and enhance overall profitability. This project highlights the power of SQL in transforming raw data into strategic decisions for the music industry.





crush
Alaina Castillo



Drop 7
Little Simz



Perfect (Exceeder)
David Guetta, Mason,
Princess Superstar



Thrill Again
ZHU, UPSAHL



HELLO (from The
Tiger's Apprentice)
ATARASHII GAKKO!



Please
NYK

Thank You!



problems
Reikko



Take Me
Dreane



BORDER ROOM
noui



Sweet Hurting
GANGGA, Valentina Ploy



Forever Was Never
Enough
Peder Elias



Head Up
Cheat Codes, Birdy

LinkedIn

Github