🎵 Music Industry Data Analysis Using PostgreSQL

This project focuses on analyzing music industry data using PostgreSQL to uncover insights about customer behavior, music genre trends, and revenue performance. The main objective was to understand which genres, artists, and cities drive the highest revenue and to identify key business opportunities for marketing and event planning.

Sachin Kush Navagekar
🎯 **Data Analyst | Turning Data into Insights**

01

02

03

🟦 Project Overview

This project focuses on analyzing music industry data using PostgreSQL to uncover insights about customer behavior, music genre trends, and revenue performance. The main objective was to understand which genres, artists, and cities drive the highest revenue and to identify key business opportunities for marketing and event planning.

🧩 Tools & Technologies

- Database: PostgreSQL
- Techniques: SQL Joins, CTEs (Common Table Expressions), Aggregate Functions, Subqueries, and Data Filtering
- Visualization: Power BI / Tableau (optional for dashboards)
- Dataset: Music Store / Music Streaming Sales Data

🔍 Business Objectives

1. Identify the most popular music genres based on listener data.
2. Determine the top customers and cities generating the highest revenue.
3. Analyze customer preferences and purchase patterns.
4. Provide data-driven insights for marketing, sales, and event strategy (e.g., promotional music festivals).

who is the senior most employee based on job title?

```
select * from employee


SELECT
    *
FROM
    EMPLOYEE
ORDER BY
    LEVELS DESC
LIMIT
    1;
```

	employee_id [PK] character varying (50)	last_name character (50)	first_name character (50)	title character varying (50)	reports_to character varying (30)	levels character varying (10)	birthdate timestamp without time zone	hire_date timestamp without time zone	add cha
1	9	Madan ...	Mohan ...	Senior General Manager	[null]	L7	1961-01-26 00:00:00	2016-01-14 00:00:00	100

which countries have the most invoices?

```sql
SELECT
    COUNT(*) AS MOST,
    BILLING_COUNTRY
FROM
    INVOICE I
    JOIN CUSTOMER C ON C.CUSTOMER_ID = I.CUSTOMER_ID
GROUP BY
    BILLING_COUNTRY
ORDER BY
    MOST DESC
    limit 10;
```

	most 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

what are top 3 values of total invoice

```sql
SELECT
    INVOICE_ID,
    SUM(TOTAL) AS TOTAL
FROM
    INVOICE
GROUP BY
    INVOICE_ID
ORDER BY
    TOTAL DESC
LIMIT
    3;
```

	invoice_id [PK] integer	total double precision
1	183	23.759999999999998
2	526	19.8
3	92	19.8

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

```sql
WITH
    TOTAL AS (
        SELECT
            C.CUSTOMER_ID AS CUST_ID,
            FIRST_NAME AS F_NAME,
            LAST_NAME AS L_NAME,
            BILLING_CITY AS CITY,
            SUM(TOTAL) AS INVOICE_TOTAL
        FROM
            INVOICE I
            JOIN CUSTOMER C ON C.CUSTOMER_ID = I.CUSTOMER_ID
        GROUP BY
            BILLING_CITY,
            F_NAME,
            L_NAME,
            C.CUSTOMER_ID
        ORDER BY
            INVOICE_TOTAL DESC
        LIMIT
            1
    )
SELECT
    CUST_ID,
    F_NAME,
    L_NAME,
    CITY,
    INVOICE_TOTAL
FROM
    TOTAL;
```

	cust_id integer	f_name character (50)	l_name character (50)	city character varying (30)	invoice_total double precision
1	5	R ...	Madhav ...	Prague	144.54000000000002

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.

```sql
SELECT DISTINCT
    EMAIL,
    FIRST_NAME,
    LAST_NAME
FROM
    CUSTOMER C
    JOIN INVOICE I ON I.CUSTOMER_ID = C.CUSTOMER_ID
    JOIN INVOICE_LINE IL ON I.INVOICE_ID = IL.INVOICE_ID
    JOIN TRACK T ON IL.TRACK_ID = IL.TRACK_ID
    JOIN GENRE G ON T.GENRE_ID = G.GENRE_ID
WHERE
    G.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	dominiquelefebvre@gmail.c...	Dominique	Lefebvre
10	edfrancis@yachoo.ca	Edward	Francis
11	eduardo@woodstock.com.br	Eduardo	Martins
12	ellie.sullivan@shaw.ca	Ellie	Sullivan
13	emma_jones@hotmail.com	Emma	Jones
14	enrique_munoz@yahoo.es	Enrique	Muñoz
15	fernadaramos4@uol.com.br	Fernanda	Ramos
16	fharris@google.com	Frank	Harris
17	fralston@gmail.com	Frank	Ralston
18	ftremblay@gmail.com	François	Tremblay
19	fzimmermann@yahoo.de	Fynn	Zimmermann
20	hannah.schneider@yahoo.de	Hannah	Schneider
21	hholy@gmail.com	Helena	Holý
22	hleacock@gmail.com	Heather	Leacock

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.

```sql
SELECT
    COUNT(A.ARTIST_ID) AS COUNT_A,
    A.NAME
FROM
    ARTIST A
    JOIN ALBUM AM ON AM.ARTIST_ID = A.ARTIST_ID
    JOIN TRACK T ON T.ALBUM_ID = AM.ALBUM_ID
    JOIN GENRE G ON G.GENRE_ID = T.GENRE_ID
GROUP BY
    A.NAME,
    G.NAME
HAVING
    G.NAME = 'Rock'
ORDER BY
    COUNT_A DESC
LIMIT
    10;
```

	count_a bigint	name character varying (120)
1	114	Led Zeppelin
2	112	U2
3	92	Deep Purple
4	81	Iron Maiden
5	54	Pearl Jam
6	52	Van Halen
7	45	Queen
8	41	The Rolling Stones
9	40	Creedence Clearwater Revival
10	35	Kiss

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.

```sql
SELECT
    NAME,
    MILLISECONDS
FROM
    TRACK
WHERE
    MILLISECONDS > (
        SELECT
            AVG(MILLISECONDS) AS AVG_S
        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
11	The Long Patrol	2925008
12	The Magnificent Warriors	2924716
13	The Living Legend, Pt. 1	2924507
14	The Gun On Ice Planet Zero, Pt. 2	2924341
15	The Hand of God	2924007
16	Experiment In Terra	2923548
17	War of the Gods, Pt. 2	2923381
18	The Living Legend, Pt. 2	2923298
19	War of the Gods, Pt. 1	2922630
20	Lost Planet of the Gods, Pt. 1	2922547
21	Baltar's Escape	2922088
22	The Lost Warrior	2920045

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

```sql
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 (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
9	20	Dan	Miller	Queen	3.96
10	5	R	Madhav	Queen	3.96
11	23	John	Gordon	Queen	2.9699999999999998
12	54	Steve	Murray	Queen	2.9699999999999998
13	31	Martha	Silk	Queen	2.9699999999999998
14	16	Frank	Harris	Queen	1.98
15	17	Jack	Smith	Queen	1.98
16	24	Frank	Ralston	Queen	1.98
17	30	Edward	Francis	Queen	1.98
18	35	Madalena	Sampaio	Queen	1.98
19	36	Hannah	Schneider	Queen	1.98
20	11	Alexandre	Rocha	Queen	1.98
21	8	Daan	Peeters	Queen	1.98
22	42	Wyatt	Girard	Queen	1.98
23	44	Terhi	Hämäläinen	Queen	1.98

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
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
```

	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
9	24	Denmark	Rock	1	1
10	46	Finland	Rock	1	1
11	211	France	Rock	1	1
12	194	Germany	Rock	1	1
13	44	Hungary	Rock	1	1
14	102	India	Rock	1	1
15	72	Ireland	Rock	1	1
16	35	Italy	Rock	1	1
17	33	Netherlands	Rock	1	1
18	40	Norway	Rock	1	1
19	40	Poland	Rock	1	1
20	108	Portugal	Rock	1	1
21	46	Spain	Rock	1	1
22	60	Sweden	Rock	1	1
23	166	United Kingdom	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 customers who spent this amount.

```sql
WITH RECURSIVE
    customter_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 customter_with_country
        GROUP BY billing_country)

SELECT cc.billing_country, cc.total_spending, cc.first_name, cc.last_name, cc.customer_id
FROM customter_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;
```

	billing_country character varying (30)	total_spending numeric	first_name character (50)	last_name character (50)	customer_id integer
1	Argentina	39.60	Diego	Gutiérrez	56
2	Australia	81.18	Mark	Taylor	55
3	Austria	69.30	Astrid	Gruber	7
4	Belgium	60.39	Daan	Peeters	8
5	Brazil	108.90	Luís	Gonçalves	1
6	Canada	99.99	François	Tremblay	3
7	Chile	97.02	Luis	Rojas	57
8	Czech Republic	144.54	R	Madhav	5
9	Denmark	37.62	Kara	Nielsen	9
10	Finland	79.20	Terhi	Hämäläinen	44
11	France	99.99	Wyatt	Girard	42
12	Germany	94.05	Fynn	Zimmermann	37
13	Hungary	78.21	Ladislav	Kovács	45
14	India	111.87	Manoj	Pareek	58
15	Ireland	114.84	Hugh	O'Reilly	46
16	Italy	50.49	Lucas	Mancini	47
17	Netherlands	65.34	Johannes	Van der Berg	48
18	Norway	72.27	Bjørn	Hansen	4
19	Poland	76.23	Stanisław	Wójcik	49
20	Portugal	102.96	João	Fernandes	34
21	Spain	98.01	Enrique	Muñoz	50
22	Sweden	75.24	Joakim	Johansson	51

Thank You

FOR YOUR ATTENTION

01

02

03

“Special thanks to everyone who inspired this project. Analyzing music data with PostgreSQL reminded me that the best insights often come from curiosity and creativity.”