Tunes & Trends: A SQL Dive into Music Sales

INTRODUCTION

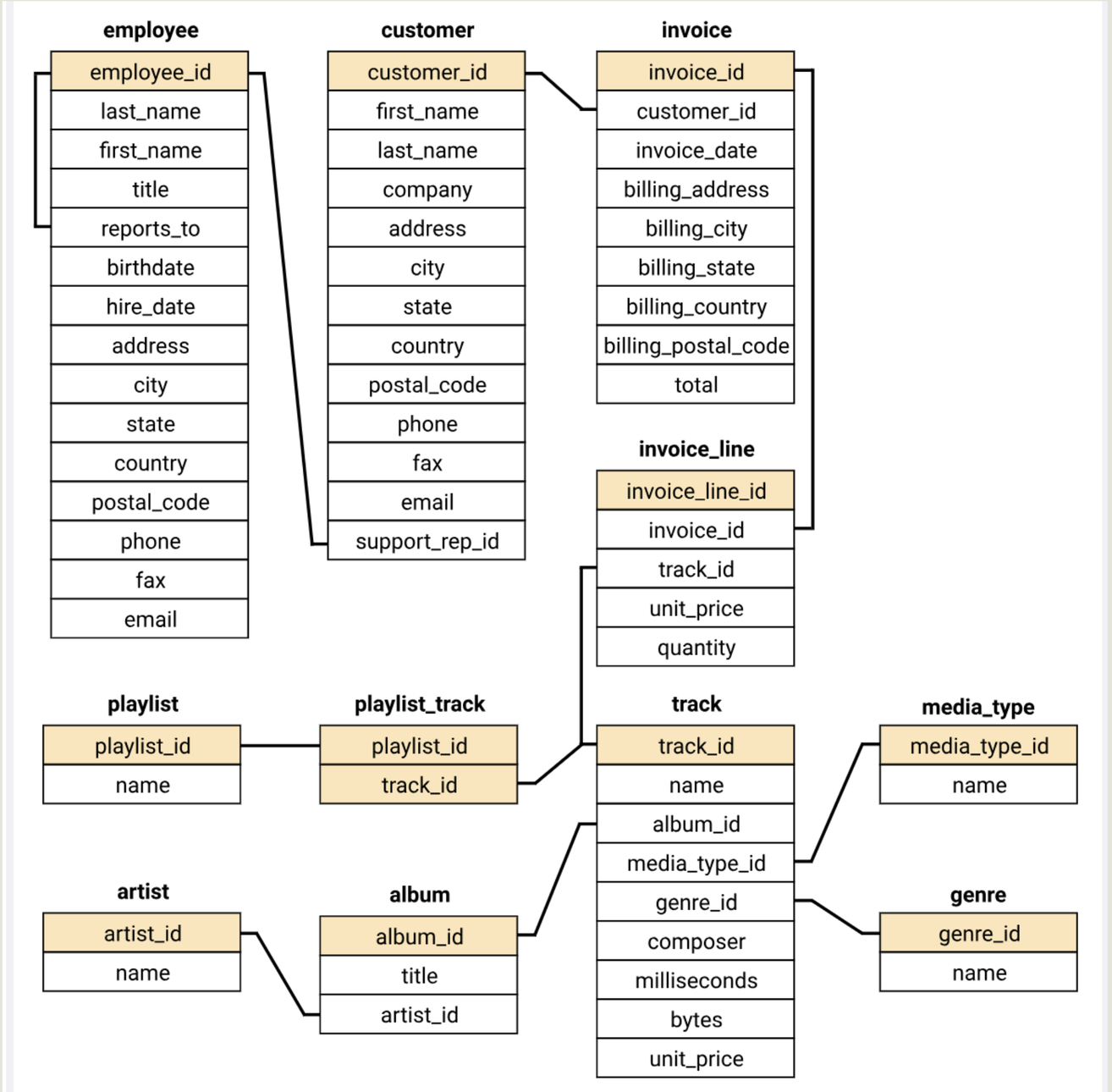
"Tunes & Trends: A SQL Dive into Music Sales" explores insights from a music store database by analyzing eleven tables: ALBUM, ARTIST, CUSTOMER, EMPLOYEE, GENRE, INVOICE, INVOICE_LINE, TRACK, PLAYLIST, PLAYLIST_TRACK, and MEDIA_TYPE.

The project answers key business questions grouped into three levels:

- **Easy:** Identifying top customers, best-performing cities, and invoice trends.
- **Moderate:** Analyzing rock music listeners, top rock artists, and longest songs.
- **Advanced:** Exploring customer spending by artist, popular genres by country, and top-spending customers per country.

This project showcases practical data analysis using SQL to uncover trends and provide actionable insights.

DATA MODELLING (SCHEMA)



Who is the senior most employee based on job title?

```sql
SELECT *
FROM EMPLOYEE
WHERE LEVELS IN (SELECT MAX(LEVELS) FROM EMPLOYEE);


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

Which countries have the most Invoices?

```sql
SELECT
    BILLING_COUNTRY, COUNT(INVOICE_ID) AS COUNT_OF_INVOICES
FROM
    INVOICE
GROUP BY BILLING_COUNTRY
ORDER BY COUNT_OF_INVOICES DESC;
```

What are top 3 values of total invoice

```
SELECT TOTAL
FROM INVOICE
ORDER BY TOTAL DESC
LIMIT 3;
```

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
SELECT
    BILLING_CITY, SUM(TOTAL) AS INVOICE_TOTAL
FROM
    INVOICE
GROUP BY BILLING_CITY
ORDER BY INVOICE_TOTAL DESC
LIMIT 1;
```

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

```sql
SELECT
    C.CUSTOMER_ID,
    C.FIRST_NAME,
    C.LAST_NAME,
    SUM(I.TOTAL) AS TOTAL
FROM
    CUSTOMER AS C
        JOIN
    INVOICE AS I ON C.CUSTOMER_ID = I.CUSTOMER_ID
GROUP BY C.CUSTOMER_ID , C.FIRST_NAME , C.LAST_NAME
ORDER BY TOTAL DESC
LIMIT 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 C.EMAIL, C.FIRST_NAME, C.LAST_NAME, G.NAME
FROM
    CUSTOMER AS C
        JOIN
    INVOICE AS I ON C.CUSTOMER_ID = I.CUSTOMER_ID
        JOIN
    INVOICE_LINE AS IL ON I.INVOICE_ID = IL.INVOICE_ID
        JOIN
    TRACK T ON IL.TRACK_ID = T.TRACK_ID
        JOIN
    GENRE AS G ON T.GENRE_ID = G.GENRE_ID
WHERE
    G.NAME IN ('ROCK')
ORDER BY C.EMAIL;
```

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
    A.ARTIST_ID,
    A.NAME AS ARTIST_NAME,
    COUNT(T.TRACK_ID) AS COUNT_TRACK
FROM
    ARTIST AS A
        JOIN
    ALBUM AS AL ON A.ARTIST_ID = AL.ARTIST_ID
        JOIN
    TRACK AS T ON AL.ALBUM_ID = T.ALBUM_ID
        JOIN
    GENRE AS G ON T.GENRE_ID = G.GENRE_ID
WHERE
    G.NAME IN ('ROCK')
GROUP BY A.ARTIST_ID , A.NAME
ORDER BY COUNT_TRACK DESC
LIMIT 10;
```

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 AS TRACK_LENGTH
FROM
    TRACK
WHERE
    MILLISECONDS > (SELECT
            AVG(MILLISECONDS) AS AVG_TRACK_LENGTH
        FROM
            TRACK)
ORDER BY TRACK_LENGTH DESC;
```

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

```sql
WITH BSA AS
(SELECT A.ARTIST_ID,A.NAME AS ARTIST_NAME,SUM(IL.UNIT_PRICE*IL.QUANTITY) AS TOTAL_SALES
FROM INVOICE_LINE IL
JOIN TRACK T ON IL.TRACK_ID=T.TRACK_ID
JOIN ALBUM AL  ON T.ALBUM_ID=AL.ALBUM_ID
JOIN ARTIST AS A ON AL.ARTIST_ID=A.ARTIST_ID
GROUP BY 1,2
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 TOTAL_SPEND
FROM INVOICE I
JOIN CUSTOMER C 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=T.TRACK_ID
JOIN ALBUM AL  ON T.ALBUM_ID=AL.ALBUM_ID
JOIN BSA  ON AL.ARTIST_ID=BSA.ARTIST_ID
GROUP BY 1,2,3,4
ORDER BY 5 DESC;
```

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
-- BY CTE
WITH BEST_SELLING_COUNTRY AS
(SELECT COUNT(I.INVOICE_ID) AS TOTAL_PURCHASE, C.COUNTRY,G.NAME,
ROW_NUMBER()OVER(PARTITION BY C.COUNTRY ORDER BY COUNT(I.INVOICE_ID) DESC) AS ROW_NO
FROM CUSTOMER C
JOIN INVOICE I ON C.CUSTOMER_ID=I.CUSTOMER_ID
JOIN INVOICE_LINE IL ON I.INVOICE_ID=IL.INVOICE_ID
JOIN TRACK T ON IL.TRACK_ID=T.TRACK_ID
JOIN GENRE G ON T.GENRE_ID=G.GENRE_ID
GROUP BY C.COUNTRY,G.NAME)
SELECT * FROM BEST_SELLING_COUNTRY
WHERE ROW_NO =1;
```

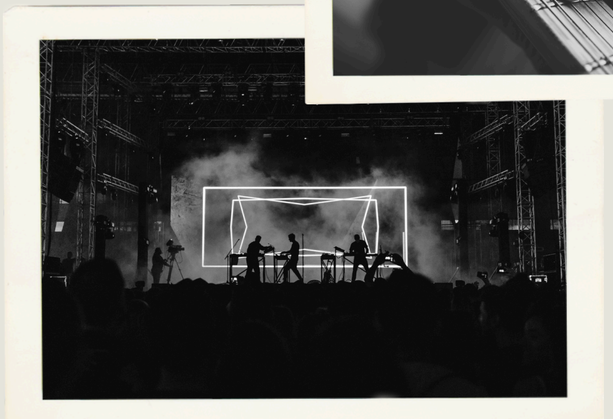
```sql
-- BY SUBQUERRY
SELECT * FROM
(SELECT COUNT(I.INVOICE_ID) AS TOTAL_PURCHASE, C.COUNTRY,G.NAME,
ROW_NUMBER()OVER(PARTITION BY C.COUNTRY ORDER BY COUNT(I.INVOICE_ID) DESC) AS ROW_NO
FROM CUSTOMER C
JOIN INVOICE I ON C.CUSTOMER_ID=I.CUSTOMER_ID
JOIN INVOICE_LINE IL ON I.INVOICE_ID=IL.INVOICE_ID
JOIN TRACK T ON IL.TRACK_ID=T.TRACK_ID
JOIN GENRE G ON T.GENRE_ID=G.GENRE_ID
GROUP BY C.COUNTRY,G.NAME) AS BEST_SELLING_COUNTRY
WHERE ROW_NO =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
-- BY CTE
WITH CUSTOMER_WITH_COUNTRY AS
(SELECT C.CUSTOMER_ID,C.FIRST_NAME,C.LAST_NAME,I.BILLING_COUNTRY,SUM(I.TOTAL) TOTAL_SPEND ,
ROW_NUMBER()OVER(PARTITION BY I.BILLING_COUNTRY ORDER BY SUM(I.TOTAL) DESC) AS ROW_NO
FROM CUSTOMER C
JOIN INVOICE I ON C.CUSTOMER_ID=I.CUSTOMER_ID
GROUP BY C.CUSTOMER_ID,C.FIRST_NAME,C.LAST_NAME,I.BILLING_COUNTRY)
SELECT * FROM CUSTOMER_WITH_COUNTRY
WHERE ROW_NO=1;
```

```sql
-- BY SUBQUERRY
SELECT * FROM
(SELECT C.CUSTOMER_ID,C.FIRST_NAME,C.LAST_NAME,I.BILLING_COUNTRY,SUM(I.TOTAL) TOTAL_SPEND ,
ROW_NUMBER()OVER(PARTITION BY I.BILLING_COUNTRY ORDER BY SUM(I.TOTAL) DESC) AS ROW_NO
FROM CUSTOMER C
JOIN INVOICE I ON C.CUSTOMER_ID=I.CUSTOMER_ID
GROUP BY C.CUSTOMER_ID,C.FIRST_NAME,C.LAST_NAME,I.BILLING_COUNTRY) AS CUSTOMER_WITH_COUNTRY
WHERE ROW_NO=1;
```

THANK YOU



- THANK YOU FOR VIEWING MY PROJECT!
- I'D LOVE TO HEAR YOUR THOUGHTS OR FEEDBACK—DROP A COMMENT OR CONNECT WITH ME

- **ANSHUL FUNDE**

(Aspiring Data Analyst | Passionate About Data-Driven Insights)

www.linkedin.com/in/anshul-funde-1386261a4

fundeanshul44@gmail.com