# Music Store EDA using Microsoft SQL Server Management Studio

Welcome to the exploratory data analysis (EDA) project for the music store company's database. In this project, we embark on a journey to delve deep into the intricacies of the data housed within the company's database, uncovering insights, patterns, and trends that will empower informed decision-making and strategic planning.

**Objective:**

The primary objective of this EDA project is to gain a comprehensive understanding of the music store company's database. Through rigorous analysis, we aim to extract valuable insights that can drive business growth, optimize operational efficiency, and enhance customer satisfaction.

**Scope:**

The scope of this EDA project encompasses various facets of the music store company's database, including but not limited to:

**Data Exploration**:

We will explore the structure and contents of the database, identifying key tables, attributes, and relationships.

2. Descriptive Statistics: Through the application of statistical measures and visualization techniques, we will gain insights into the distribution, central tendency, and dispersion of the data.

3. Pattern Recognition: By analyzing patterns and trends within the data, we aim to uncover valuable insights that can inform strategic decision-making processes.

4. Anomaly Detection: We will employ anomaly detection techniques to identify and investigate any irregularities or outliers present in the data.

5. Data Quality Assessment: Assessing the quality of the data is crucial for ensuring its reliability and validity. We will perform data quality checks to identify and address any inconsistencies, missing values, or inaccuracies.

**Approach:**

Our approach to conducting this EDA project will be systematic and thorough. We will follow a structured methodology encompassing the following steps:

1. Data Collection: We will gather relevant data from the music store company's database, ensuring completeness and accuracy.

2. Data Preprocessing: Prior to analysis, we will preprocess the data to address any missing values, outliers, or inconsistencies.

3. Exploratory Analysis: Utilizing a combination of statistical analysis and data visualization techniques, we will explore the data from various perspectives, uncovering meaningful insights.

4. Insight Generation: Through in-depth analysis and interpretation of the findings, we will generate actionable insights that can drive business decisions and strategies.

5. Reporting: Finally, we will document our findings, insights, and recommendations in a comprehensive report, providing stakeholders with a clear understanding of the outcomes of the EDA project.

# SQL Queries

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

**Query:**

```
SELECT TOP 1 title, last_name, first_name
FROM employee
ORDER BY levels DESC
```
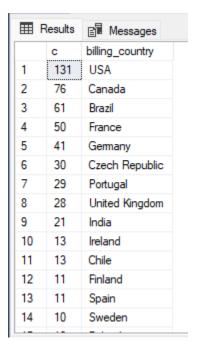
**Output:**

| | title | last_name | first_name |
|---|---|---|---|
| 1 | Senior General Manager | Madan | Mohan |

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

**Query:**

```
SELECT COUNT(*) AS c, billing_country
FROM invoice
GROUP BY billing_country
ORDER BY c DESC
```

**Output:**

| | c | billing_country |
|---|---|---|
| 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 | Ireland |
| 11 | 13 | Chile |
| 12 | 11 | Finland |
| 13 | 11 | Spain |
| 14 | 10 | Sweden |

**Q3:** What are top 3 values of total invoice?

**Query:**

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

**Output:**

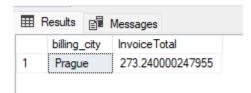| | total |
|---|---|
| 1 | 23.7600002288818 |
| 2 | 19.7999992370605 |
| 3 | 19.7999992370605 |
| 4 | 19.7999992370605 |
| 5 | 19.7999992370605 |
| 6 | 18.8099994659424 |
| 7 | 17.8199996948242 |
| 8 | 17.8199996948242 |
| 9 | 17.8199996948242 |
| 10 | 17.8199996948242 |
| 11 | 17.8199996948242 |
| 12 | 17.8199996948242 |
| 13 | 17.8199996948242 |
| 14 | 16.8299999237061 |

**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 TOP 1 billing_city,SUM(total) AS InvoiceTotal
FROM invoice
GROUP BY billing_city
ORDER BY InvoiceTotal DESC
```

**Output:**

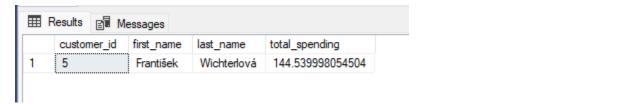| | billing_city | InvoiceTotal |
|---|---|---|
| 1 | Prague | 273.240000247955 |

**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.

**Query:**

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

**Output:**

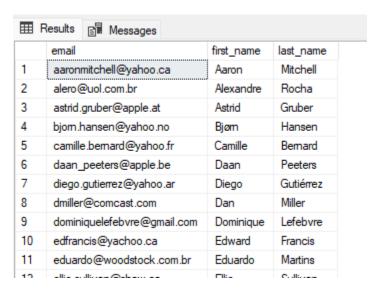| | customer_id | first_name | last_name | total_spending |
|---|---|---|---|---|
| 1 | 5 | František | Wichterlová | 144.539998054504 |

**Q6:** 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:**

```sql
SELECT DISTINCT c.email, c.first_name, c.last_name
FROM customer c
JOIN invoice i ON c.customer_id = i.customer_id
JOIN invoice_line il ON i.invoice_id = il.invoice_id
WHERE il.track_id IN (
SELECT t.track_id FROM track t
JOIN genre g ON t.genre_id = g.genre_id
WHERE g.name LIKE 'Rock'
)
ORDER BY c.email;
```

**Output:**

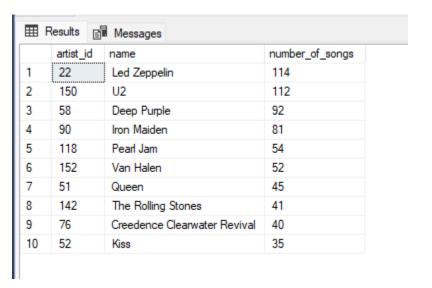| | email | first_name | last_name |
|---|---|---|---|
| 1 | aaronmitchell@yahoo.ca | Aaron | Mitchell |
| 2 | alero@uol.com.br | Alexandre | Rocha |
| 3 | astrid.gruber@apple.at | Astrid | Gruber |
| 4 | bjorn.hansen@yahoo.no | Bjørn | Hansen |
| 5 | camille.bernard@yahoo.fr | Camille | Bernard |
| 6 | daan_peeters@apple.be | Daan | Peeters |
| 7 | diego.gutierrez@yahoo.ar | Diego | Gutiérrez |
| 8 | dmiller@comcast.com | Dan | Miller |
| 9 | dominiquelefebvre@gmail.com | Dominique | Lefebvre |
| 10 | edfrancis@yachoo.ca | Edward | Francis |
| 11 | eduardo@woodstock.com.br | Eduardo | Martins |
| 12 | | | |

**Q7:** 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.

**Query:**

```sql
SELECT TOP 10 a.artist_id, a.name, COUNT(t.track_id) AS number_of_songs
FROM track t
JOIN album al ON al.album_id = t.album_id
JOIN artist a ON a.artist_id = al.artist_id
JOIN genre g ON g.genre_id = t.genre_id
WHERE g.name LIKE 'Rock'
GROUP BY a.artist_id, a.name
ORDER BY number_of_songs DESC;
```
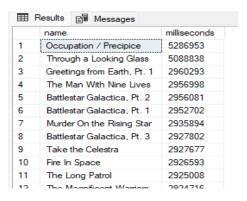
**Output:**

| | artist_id | name | number_of_songs |
|---|---|---|---|
| 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 |

**Q8:** 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:**

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

| | name | milliseconds |
|---|---|---|
| 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 Warrior | 2924716 |

**Output:**

**Q9**: 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 returns all Genres.

**Query:**

```sql
WITH popular_genre AS
(
SELECT
 COUNT(il.quantity) AS purchases,
 c.country,
 g.name AS genre_name,
 g.genre_id,
 ROW_NUMBER() OVER(PARTITION BY c.country ORDER BY COUNT(il.quantity) DESC) AS RowNo
 FROM
 invoice_line il
JOIN invoice i ON i.invoice_id = il.invoice_id
JOIN customer c ON c.customer_id = i.customer_id
JOIN track t ON t.track_id = il.track_id
JOIN genre g ON g.genre_id = t.genre_id
 GROUP BY
 c.country,
 g.name,
 g.genre_id
)
SELECT
country,
genre_name,
genre_id,
purchases
FROM
popular_genre
WHERE
RowNo = 1;
```

**Output:**

| | country | genre_name | genre_id | purchases |
|---|---|---|---|---|
| 1 | Argentina | Alternative & Punk | 4 | 17 |
| 2 | Australia | Rock | 1 | 34 |
| 3 | Austria | Rock | 1 | 40 |
| 4 | Belgium | Rock | 1 | 26 |
| 5 | Brazil | Rock | 1 | 205 |
| 6 | Canada | Rock | 1 | 333 |
| 7 | Chile | Rock | 1 | 61 |
| 8 | Czech Republic | Rock | 1 | 143 |
| 9 | Denmark | Rock | 1 | 24 |
| 10 | Finland | Rock | 1 | 46 |
| 11 | France | Rock | 1 | 211 |
| 12 | Germany | Rock | 1 | 194 |

**Q10**: 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:**

```sql
WITH popular_genre AS
(
SELECT
  COUNT(il.quantity) AS purchases,
  c.country,
  g.name AS genre_name,
  g.genre_id,
  ROW_NUMBER() OVER(PARTITION BY c.country ORDER BY COUNT(il.quantity) DESC) AS RowNo
FROM
  invoice_line il
JOIN invoice i ON i.invoice_id = il.invoice_id
JOIN customer c ON c.customer_id = i.customer_id
JOIN track t ON t.track_id = il.track_id
JOIN genre g ON g.genre_id = t.genre_id
GROUP BY
  c.country,
  g.name,
  g.genre_id
)
SELECT
country,
genre_name,
genre_id,
purchases
FROM
popular_genre
WHERE
RowNo = 1;
```

**Output:**

| | country | genre_name | genre_id | purchases |
|---|---|---|---|---|
| 1 | Argentina | Alternative & Punk | 4 | 17 |
| 2 | Australia | Rock | 1 | 34 |
| 3 | Austria | Rock | 1 | 40 |
| 4 | Belgium | Rock | 1 | 26 |
| 5 | Brazil | Rock | 1 | 205 |
| 6 | Canada | Rock | 1 | 333 |
| 7 | Chile | Rock | 1 | 61 |
| 8 | Czech ... | Rock | 1 | 143 |
| 9 | Denmark | Rock | 1 | 24 |
| 10 | Finland | Rock | 1 | 46 |
| 11 | France | Rock | 1 | 211 |
| 12 | Germany | Rock | 1 | 194 |
| 13 | Hungary | Rock | 1 | 44 |

## Conclusion:

As we embark on this EDA project, we anticipate uncovering valuable insights that will not only enhance our understanding of the music store company's database but also contribute to its continued success and growth. Through collaboration, diligence, and a commitment to excellence, we aim to deliver actionable recommendations that will drive tangible business outcomes.

Report made by

**Saurabh Kulkarni**

(Certified in SQL, Power BI, MS Excel)

Thank you.