End to End Music Sales Analysis

Project Overview

The **End to End Music Sales Analysis** project aims to provide a comprehensive analysis of music sales data using a combination of modern data engineering and analytics tools. This project leverages various technologies to extract, transform, and load (ETL) data, enabling insightful visualizations and analyses.

Tools Used

- **Snowflake**: A cloud-based data warehouse that stores the raw music sales data.
- **PySpark**: An open-source data processing framework used for ETL operations to clean, transform, and prepare the data for analysis.
- **PostgreSQL**: A relational database that serves as the data warehouse where transformed data is stored for reporting and analytics.
- **Power BI**: A powerful business analytics tool used for creating interactive visualizations and dashboards based on the processed data.
- **Google Gemini**: An AI-powered tool that assists in generating insights and summaries from the analyzed data.
- **Pandas and Seaborn**: Python libraries utilized for data manipulation and visualization, providing advanced statistical graphics for the analysis.

Project Workflow

1. **Data Extraction from Snowflake**:
    - The raw music sales data is extracted from the Snowflake database using SQL queries to retrieve relevant tables such as `EMPLOYEE`, `CUSTOMER`, `INVOICE`, and `TRACK`.
2. **Data Transformation with PySpark**:
    - PySpark is used to load the data into DataFrames, where various transformations are applied to calculate metrics such as total sales by employee, popular music by country, and customer lifetime value.
    - Temporary views are created for each transformation to facilitate the processing and organization of data.
3. **Loading Data into PostgreSQL**:
    - The transformed data is written to PostgreSQL tables, allowing for efficient querying and reporting.

4. **Data Analysis and Visualization**:
    - Power BI is employed to visualize the metrics calculated from the music sales data.
    - Google Gemini enhances the analysis by providing insights and recommendations based on the processed data.
    - Pandas and Seaborn are used for further analysis and advanced visualizations, allowing deeper exploration of the data.
5. **Insights and Reporting**:
    - The final analysis provides insights into sales performance, customer behavior, and trends in music consumption, helping stakeholders make informed business decisions.

Analysis Topics

1. Total Sales by Employee

- This analysis calculates the total sales attributed to each employee, allowing for performance evaluation. It helps identify top-performing employees and those who may need additional support or training.

2. Popular Music by Country

- This metric analyzes which tracks are most frequently purchased in different countries. Understanding regional preferences can guide marketing efforts and inventory management.

3. Popular Genre by Country

- This analysis reveals the popularity of music genres across various countries, providing insights into cultural differences in music preferences. It can inform artists and record labels about which genres to promote in specific regions.

4. Total Sales by Country

- This metric aggregates total sales data by country, allowing stakeholders to identify key markets and potential growth areas. It helps in understanding global sales performance and resource allocation.

5. Customer Lifetime Value

- Customer Lifetime Value (CLV) measures the total revenue a customer is expected to generate during their relationship with the business. This analysis helps in understanding customer retention and the effectiveness of marketing strategies.

6. Most Sold Album

- This metric identifies which albums have sold the most copies. Understanding trends in album sales can inform future production and marketing strategies for artists and labels.

7. Top Employees by Number of Supported Customers

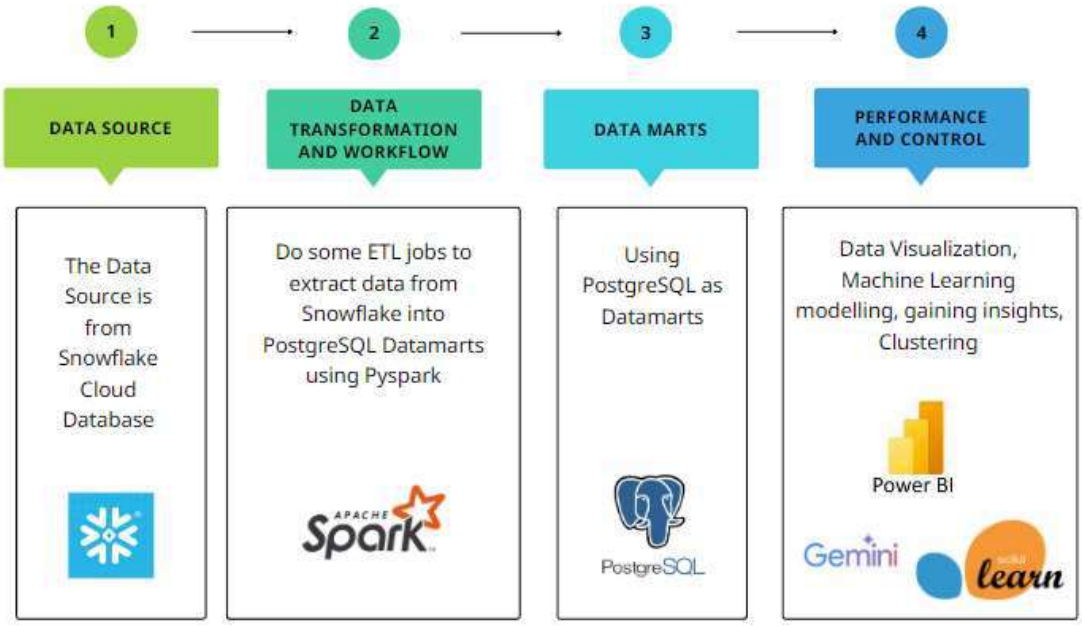
- This analysis determines which employees support the highest number of customers. It provides insights into employee engagement and customer service effectiveness, allowing for better allocation of customer support resources.

Conclusion

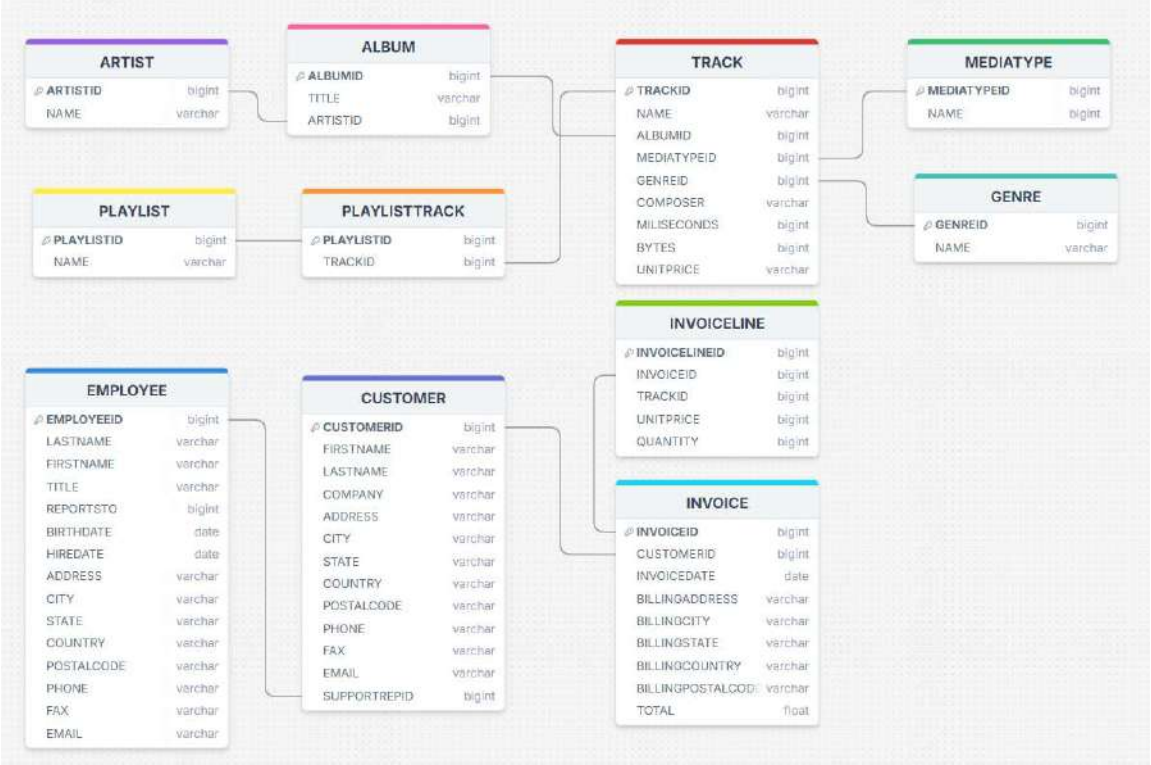
The **End to End Music Sales Analysis** project showcases a robust framework for handling and analyzing large datasets in the music industry. By integrating Snowflake, PySpark, PostgreSQL, Power BI, Google Gemini, and Python libraries, this project delivers valuable insights that can drive strategic decisions and enhance sales performance.

Tools that will be used

Project Workflow



ERD Diagram



Data Transformation with PySpark

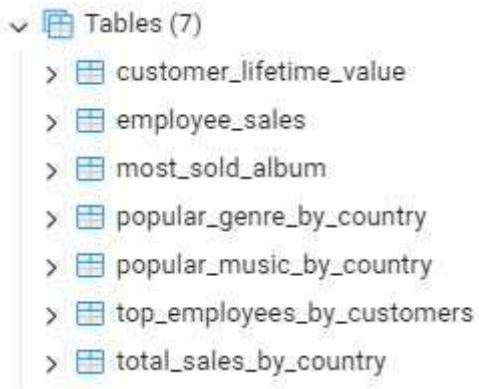
In [ ]:
```
from pyspark.sql.utils import AnalysisException
from pyspark.sql import SparkSession
```

```
# Initialize SparkSession with Snowflake and PostgreSQL JDBC drivers
spark = SparkSession.builder \
    .appName("SnowflakeToDataMart") \
    .config("spark.jars", "C:\\Users\\Michael\\Pyspark\\snowflake-jdbc-3.13.30.jar,
    .getOrCreate()

# Snowflake connection options
sfOptions = {
    "sfURL": "https://lx25930.europe-west3.gcp.snowflakecomputing.com",
    "sfAccount": "lx25930.europe-west3.gcp",
    "sfUser": "MAGICDASH",
    "sfPassword": "********************",
    "sfDatabase": "MUSIC_DATABASE",
    "sfSchema": "PUBLIC",
    "sfWarehouse": "COMPUTE_WH",
    "sfRole": "ACCOUNTADMIN"
}

# PostgreSQL connection options
jdbc_driver_path = "C:\\Users\\Michael\\Pyspark\\postgresql-42.7.3.jar"
jdbc_url = "jdbc:postgresql://localhost:5432/project4"
connection_properties = {
    "user": "postgres",
    "password": "****************",
    "driver": "org.postgresql.Driver"
}

# Snowflake source name for Spark
SNOWFLAKE_SOURCE_NAME = "net.snowflake.spark.snowflake"

# Load data from Snowflake
employee_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).option(
customer_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).option(
invoice_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).option("
invoiceline_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).opti
track_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).option("db
genre_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).option("db
album_df = spark.read.format(SNOWFLAKE_SOURCE_NAME).options(**sfOptions).option("db

# Create temporary views for each DataFrame
employee_df.createOrReplaceTempView("EMPLOYEE")
customer_df.createOrReplaceTempView("CUSTOMER")
invoice_df.createOrReplaceTempView("INVOICE")
invoiceline_df.createOrReplaceTempView("INVOICELINE")
track_df.createOrReplaceTempView("TRACK")
genre_df.createOrReplaceTempView("GENRE")
album_df.createOrReplaceTempView("ALBUM")

# SQL queries using the temporary views

# 1. Total sales by employee
sales_by_employee_df = spark.sql("""
SELECT e.EMPLOYEEID, e.FIRSTNAME, e.LASTNAME, SUM(i.TOTAL) AS TOTAL_SALES
FROM EMPLOYEE e
JOIN CUSTOMER c ON e.EMPLOYEEID = c.SUPPORTREPID
```

```
JOIN INVOICE i ON c.CUSTOMERID = i.CUSTOMERID
GROUP BY e.EMPLOYEEID, e.FIRSTNAME, e.LASTNAME
ORDER BY TOTAL_SALES DESC
""")

# 2. Popular music by country
popular_music_by_country_df = spark.sql("""
SELECT t.NAME AS TRACK_NAME, c.COUNTRY, COUNT(il.TRACKID) AS PLAY_COUNT
FROM TRACK t
JOIN INVOICELINE il ON t.TRACKID = il.TRACKID
JOIN INVOICE i ON il.INVOICEID = i.INVOICEID
JOIN CUSTOMER c ON i.CUSTOMERID = c.CUSTOMERID
GROUP BY t.NAME, c.COUNTRY
ORDER BY c.COUNTRY, PLAY_COUNT DESC
""")

# 3. Popular genre by country
popular_genre_by_country_df = spark.sql("""
SELECT g.NAME AS GENRE_NAME, c.COUNTRY, COUNT(il.TRACKID) AS PLAY_COUNT
FROM GENRE g
JOIN TRACK t ON g.GENREID = t.GENREID
JOIN INVOICELINE il ON t.TRACKID = il.TRACKID
JOIN INVOICE i ON il.INVOICEID = i.INVOICEID
JOIN CUSTOMER c ON i.CUSTOMERID = c.CUSTOMERID
GROUP BY g.NAME, c.COUNTRY
ORDER BY c.COUNTRY, PLAY_COUNT DESC
""")

# 4. Total sales by country
total_sales_by_country_df = spark.sql("""
SELECT c.COUNTRY, SUM(i.TOTAL) AS TOTAL_SALES
FROM CUSTOMER c
JOIN INVOICE i ON c.CUSTOMERID = i.CUSTOMERID
GROUP BY c.COUNTRY
ORDER BY TOTAL_SALES DESC
""")

# 5. Customer lifetime value
customer_lifetime_value_df = spark.sql("""
SELECT c.CUSTOMERID, c.FIRSTNAME, c.LASTNAME, SUM(i.TOTAL) AS LIFETIME_VALUE
FROM CUSTOMER c
JOIN INVOICE i ON c.CUSTOMERID = i.CUSTOMERID
GROUP BY c.CUSTOMERID, c.FIRSTNAME, c.LASTNAME
ORDER BY LIFETIME_VALUE DESC
""")

# 6. Most sold album
most_sold_album_df = spark.sql("""
SELECT al.TITLE AS ALBUM_NAME, COUNT(il.TRACKID) AS SOLD_TRACKS
FROM ALBUM al
JOIN TRACK t ON al.ALBUMID = t.ALBUMID
JOIN INVOICELINE il ON t.TRACKID = il.TRACKID
GROUP BY al.TITLE
ORDER BY SOLD_TRACKS DESC
""")
```

```
# 7. Top employees by number of supported customers
top_employees_df = spark.sql("""
SELECT e.EMPLOYEEID, e.FIRSTNAME, e.LASTNAME, COUNT(c.CUSTOMERID) AS SUPPORTED_CUST
FROM EMPLOYEE e
JOIN CUSTOMER c ON e.EMPLOYEEID = c.SUPPORTREPID
GROUP BY e.EMPLOYEEID, e.FIRSTNAME, e.LASTNAME
ORDER BY SUPPORTED_CUSTOMERS DESC
""")

# Load data into PostgreSQL
sales_by_employee_df.write.jdbc(url=jdbc_url, table="employee_sales", mode="overwri
popular_music_by_country_df.write.jdbc(url=jdbc_url, table="popular_music_by_countr
popular_genre_by_country_df.write.jdbc(url=jdbc_url, table="popular_genre_by_countr
total_sales_by_country_df.write.jdbc(url=jdbc_url, table="total_sales_by_country",
customer_lifetime_value_df.write.jdbc(url=jdbc_url, table="customer_lifetime_value"
most_sold_album_df.write.jdbc(url=jdbc_url, table="most_sold_album", mode="overwrit
top_employees_df.write.jdbc(url=jdbc_url, table="top_employees_by_customers", mode=
```

Processed Tables on PostgreSQL Database



Connect to PostgreSQL Datamart with Psycopg

In [1]:
```
import pandas as pd
import psycopg2
from psycopg2 import OperationalError

try:
    # Attempt to establish a connection to PostgreSQL
    connection = psycopg2.connect(
        host="localhost",
        dbname="project4",
        user="postgres",
        password="permataputihg101",
        port="5432"
    )
    print("Connection Success")
```

```
except OperationalError as e:
    print("Connection Failed")
    print(f"Error details: {e}")
```

```
Connection Success
```

In [2]:
```
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
sns.set_theme(color_codes=True)
```

In [3]:
```
import textwrap
import google.generativeai as genai
from IPython.display import display
from IPython.display import Markdown
```

Customer Lifetime Value

In [4]:
```
query = "SELECT * FROM public.customer_lifetime_value"
df = pd.read_sql_query(query, connection)
df.head()
```

```
C:\Users\Michael\AppData\Local\Temp\ipykernel_16724\1267695055.py:2: UserWarning: pa
ndas only supports SQLAlchemy connectable (engine/connection) or database string URI
or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider u
sing SQLAlchemy.
  df = pd.read_sql_query(query, connection)
```

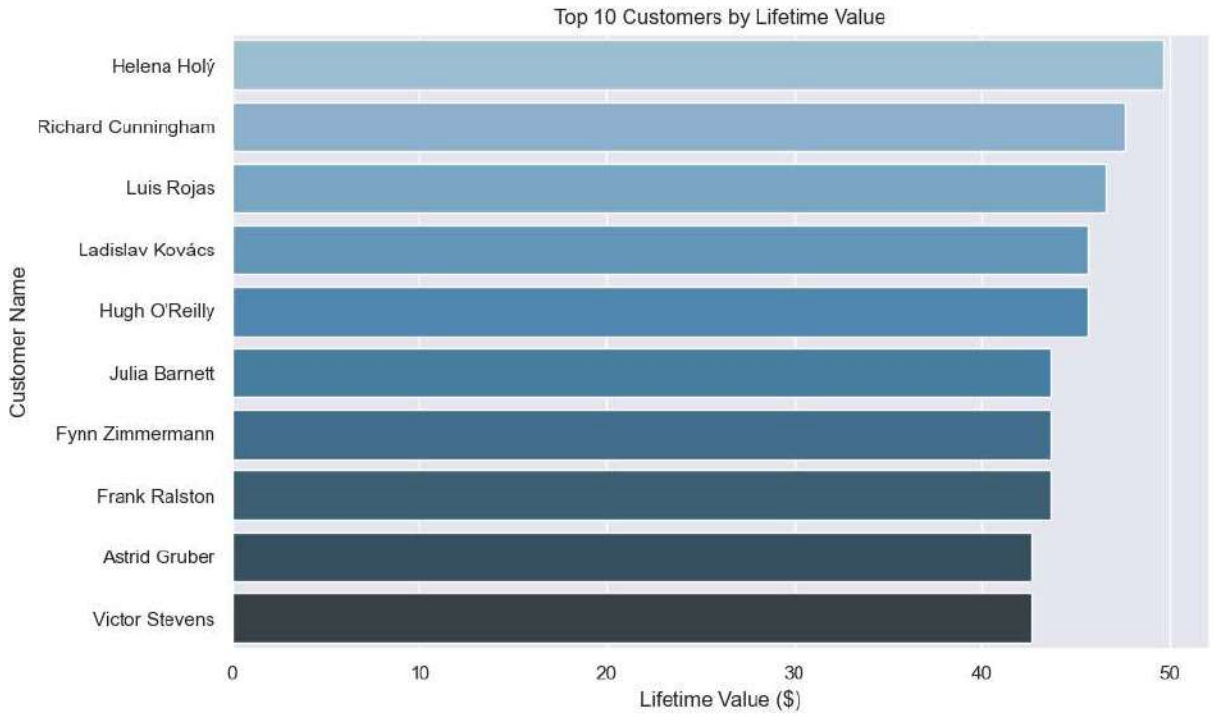
Out[4]:

	CUSTOMERID	FIRSTNAME	LASTNAME	LIFETIME_VALUE
0	6.0	Helena	Holý	49.62
1	26.0	Richard	Cunningham	47.62
2	57.0	Luis	Rojas	46.62
3	45.0	Ladislav	Kovács	45.62
4	46.0	Hugh	O'Reilly	45.62

Showing Top 10 Customers by Customer Lifetime Value

In [5]:
```
# Combine First and Last names into a single column for visualization
df['CUSTOMER_NAME'] = df['FIRSTNAME'] + ' ' + df['LASTNAME']

# Select the top 10 customers based on Lifetime Value
top_10_customers = df.nlargest(10, 'LIFETIME_VALUE')

# Set up the matplotlib figure
plt.figure(figsize=(10, 6))
```

```
# Plot horizontal bar chart
sns.barplot(x='LIFETIME_VALUE', y='CUSTOMER_NAME', data=top_10_customers, palette='

# Add labels and title
plt.xlabel('Lifetime Value ($)')
plt.ylabel('Customer Name')
plt.title('Top 10 Customers by Lifetime Value')

# Show plot
plt.tight_layout()
plt.savefig('clv.png', dpi=300)
plt.show()
```



In [7]:

```
def to_markdown(text):
  text = text.replace('•', '  *')
  return Markdown(textwrap.indent(text, '> ', predicate=lambda _: True))

genai.configure(api_key="*********************************")

import PIL.Image

img = PIL.Image.open("clv.png")
model = genai.GenerativeModel('gemini-1.5-flash-latest')
response = model.generate_content(img)

response = model.generate_content(["Explain it by points in simple and clear terms.
response.resolve()
formatted_text = to_markdown(response.text)
display(formatted_text)
```

> The graph shows the top 10 customers by lifetime value.
>
> **Key Findings:**
>
> - **Helena Holý** has the highest lifetime value, followed by **Richard Cunningham** and **Luis Rojas**.
> - The top 10 customers have a lifetime value ranging from **42** to **49.**
> - The difference in lifetime value between the top 3 customers and the rest is significant.
>
> **Actionable Insights:**
>
> - **Focus on retaining and nurturing the top 10 customers.** They are your most valuable assets and contribute significantly to your business.
> - **Analyze the factors that contribute to the high lifetime value of the top customers.** This could be their purchase history, frequency, average order value, or engagement with your brand.
> - **Develop targeted marketing campaigns and loyalty programs to further engage the top customers.** Offer them exclusive deals, personalized recommendations, and VIP experiences.
> - **Identify the common characteristics of the top customers.** This will help you attract and retain similar customers in the future.
> - **Investigate the reasons for the lower lifetime value of other customers.** This could reveal opportunities for improvement and help you increase the lifetime value of all your customers.

Employee Sales

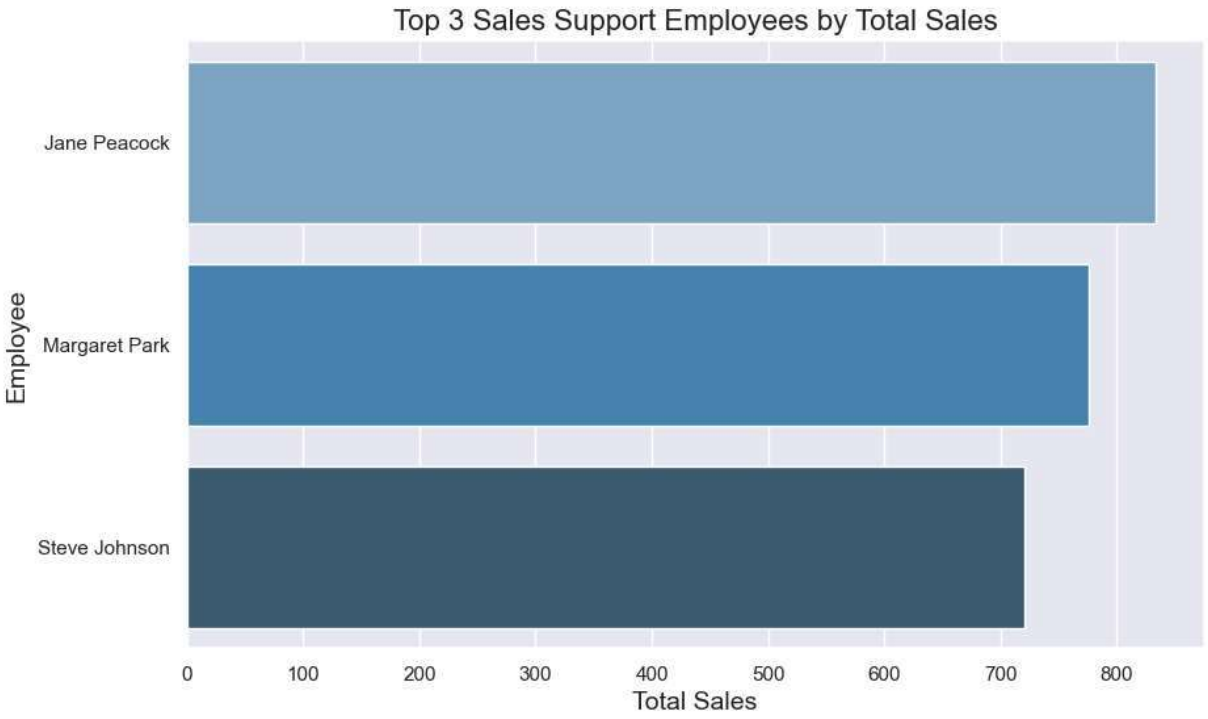
In [8]:
```
query = "SELECT * FROM public.employee_sales"
df = pd.read_sql_query(query, connection)
df.head()
```

```
C:\Users\Michael\AppData\Local\Temp\ipykernel_16724\674774404.py:2: UserWarning: pan
das only supports SQLAlchemy connectable (engine/connection) or database string URI
or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider u
sing SQLAlchemy.
  df = pd.read_sql_query(query, connection)
```

Out[8]:

	EMPLOYEEID	FIRSTNAME	LASTNAME	TOTAL_SALES
0	3.0	Jane	Peacock	833.04
1	4.0	Margaret	Park	775.40
2	5.0	Steve	Johnson	720.16

In [12]:
```
# Combine first and last names
df['FULL_NAME'] = df['FIRSTNAME'] + " " + df['LASTNAME']

# Sort by total sales and get the top 5
top_3_employees = df.nlargest(5, 'TOTAL_SALES')

# Create horizontal bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x='TOTAL_SALES', y='FULL_NAME', data=top_3_employees, palette='Blues_d'
plt.title('Top 3 Sales Support Employees by Total Sales', fontsize=16)
plt.xlabel('Total Sales', fontsize=14)
plt.ylabel('Employee', fontsize=14)
plt.savefig('e_sales.png', dpi=300)
plt.show()
```



In [13]:
```
def to_markdown(text):
  text = text.replace('•', '  *')
  return Markdown(textwrap.indent(text, '> ', predicate=lambda _: True))

genai.configure(api_key="**********************************")

import PIL.Image

img = PIL.Image.open("e_sales.png")
model = genai.GenerativeModel('gemini-1.5-flash-latest')
response = model.generate_content(img)

response = model.generate_content(["Explain it by points in simple and clear terms.
response.resolve()
formatted_text = to_markdown(response.text)
display(formatted_text)
```

> The bar chart shows the top 3 sales support employees by total sales. Here are the key findings and actionable insights:
>
> **Key Findings:**
>
> - **Jane Peacock is the top sales support employee with the highest total sales.** This indicates she is a high performer and a valuable asset to the company.
> - **Margaret Park is the second highest performer.** She is also a strong contributor to sales.
> - **Steve Johnson has the lowest total sales among the top 3.**
>
> **Actionable Insights:**
>
> - **Recognize and reward Jane Peacock and Margaret Park for their outstanding performance.** This could include bonuses, promotions, or other forms of recognition.
> - **Provide additional training and support to Steve Johnson to help him improve his sales performance.** This could include mentoring, coaching, or access to new resources.
> - **Analyze the factors that contribute to Jane Peacock's and Margaret Park's success.** This could help identify best practices that can be shared with other employees.
> - **Investigate the reasons behind Steve Johnson's lower sales performance.** This could include factors such as lack of experience, lack of training, or lack of motivation.
> - **Track sales performance over time to see how these employees are performing.** This will help identify trends and make necessary adjustments to strategies.

Most Sold Albums

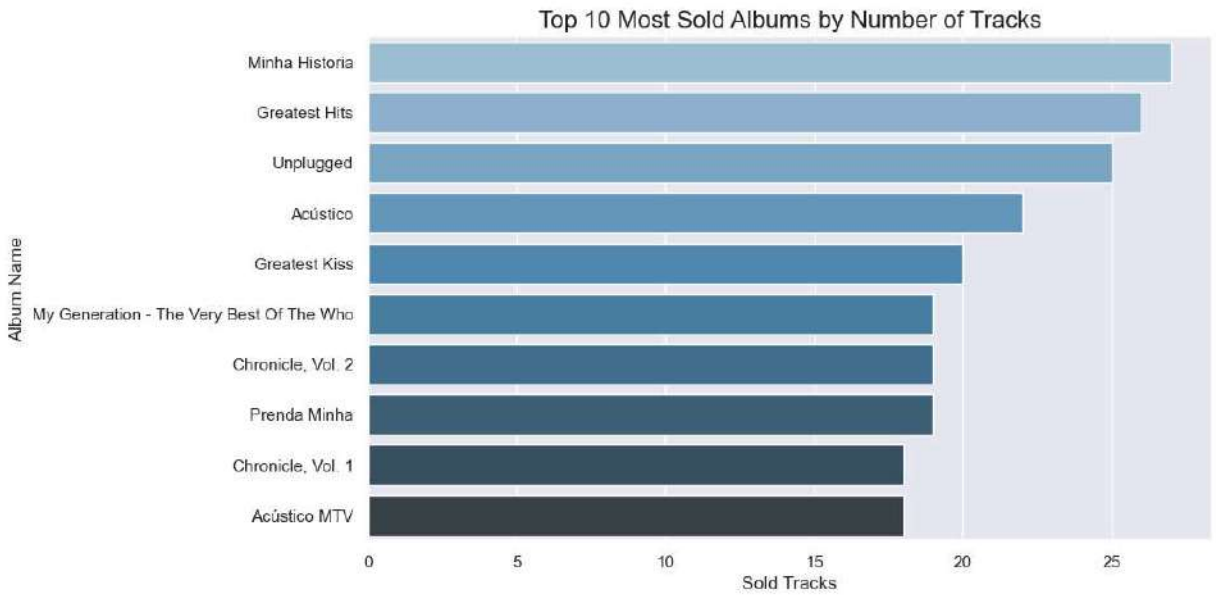
In [15]:
```
query = "SELECT * FROM public.most_sold_album"
df = pd.read_sql_query(query, connection)
df.head()
```

```
C:\Users\Michael\AppData\Local\Temp\ipykernel_16724\3001549988.py:2: UserWarning: pa
ndas only supports SQLAlchemy connectable (engine/connection) or database string URI
or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider u
sing SQLAlchemy.
  df = pd.read_sql_query(query, connection)
```

Out[15]:

	ALBUM_NAME	SOLD_TRACKS
0	Minha Historia	27
1	Greatest Hits	26
2	Unplugged	25
3	Acústico	22
4	Greatest Kiss	20

In [16]:
```
# Sort the data and select the top 10 albums
top_albums = df.nlargest(10, 'SOLD_TRACKS')

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='SOLD_TRACKS', y='ALBUM_NAME', data=top_albums, palette='Blues_d')

# Add labels and title
plt.title('Top 10 Most Sold Albums by Number of Tracks', fontsize=16)
plt.xlabel('Sold Tracks', fontsize=12)
plt.ylabel('Album Name', fontsize=12)
plt.savefig('album_sales.png', dpi=300)
# Display the plot
plt.show()
```



In [17]:
```
def to_markdown(text):
  text = text.replace('•', '  *')
  return Markdown(textwrap.indent(text, '> ', predicate=lambda _: True))

genai.configure(api_key="******************************")

import PIL.Image

img = PIL.Image.open("album_sales.png")
model = genai.GenerativeModel('gemini-1.5-flash-latest')
```

```
response = model.generate_content(img)

response = model.generate_content(["Explain it by points in simple and clear terms.
response.resolve()
formatted_text = to_markdown(response.text)
display(formatted_text)
```

Top 10 Most Sold Albums by Number of Tracks

Here are the key findings and actionable insights based on the chart:

- **Album Length Matters:** The top 10 most sold albums all have a significant number of tracks, ranging from 17 to 27. This indicates that consumers value quantity when it comes to music purchases.
- **Compilations Are Popular:** Many of the top-selling albums are compilations like "Greatest Hits" and "Best Of The Who," suggesting that consumers often prefer to buy a collection of popular songs rather than individual albums.
- **Variety is Key:** The presence of various albums across different genres, including acoustic, unplugged, and "greatest hits" compilations, indicates that a diverse catalog is important for success.
- **Focus on Track Count:** The data suggests that focusing on creating albums with a larger number of tracks could lead to increased sales. This could include longer albums with more original content or curated compilations of popular hits.
- **Capitalize on Nostalgia:** "Greatest Hits" and "Best Of" albums appeal to a sense of nostalgia. Building on this trend with curated compilations could be a successful strategy for new artists.
- **Consider Acoustic Variations:** The inclusion of acoustic versions of popular albums suggests that offering alternative takes on existing material can appeal to a broader audience.

Actionable Insights:

- **For New Artists:** Focus on creating albums with a good number of tracks, offering a variety of styles and potentially including a curated compilation of popular songs.
- **For Established Artists:** Consider releasing "Greatest Hits" or "Best Of" albums to capitalize on nostalgia and attract a broader audience.
- **For All Artists:** Explore the potential of acoustic variations of existing albums to reach a wider market.

Popular Genres by Country

In [22]:
```
query = "SELECT * FROM public.popular_genre_by_country"
df = pd.read_sql_query(query, connection)
df.head(5)
```

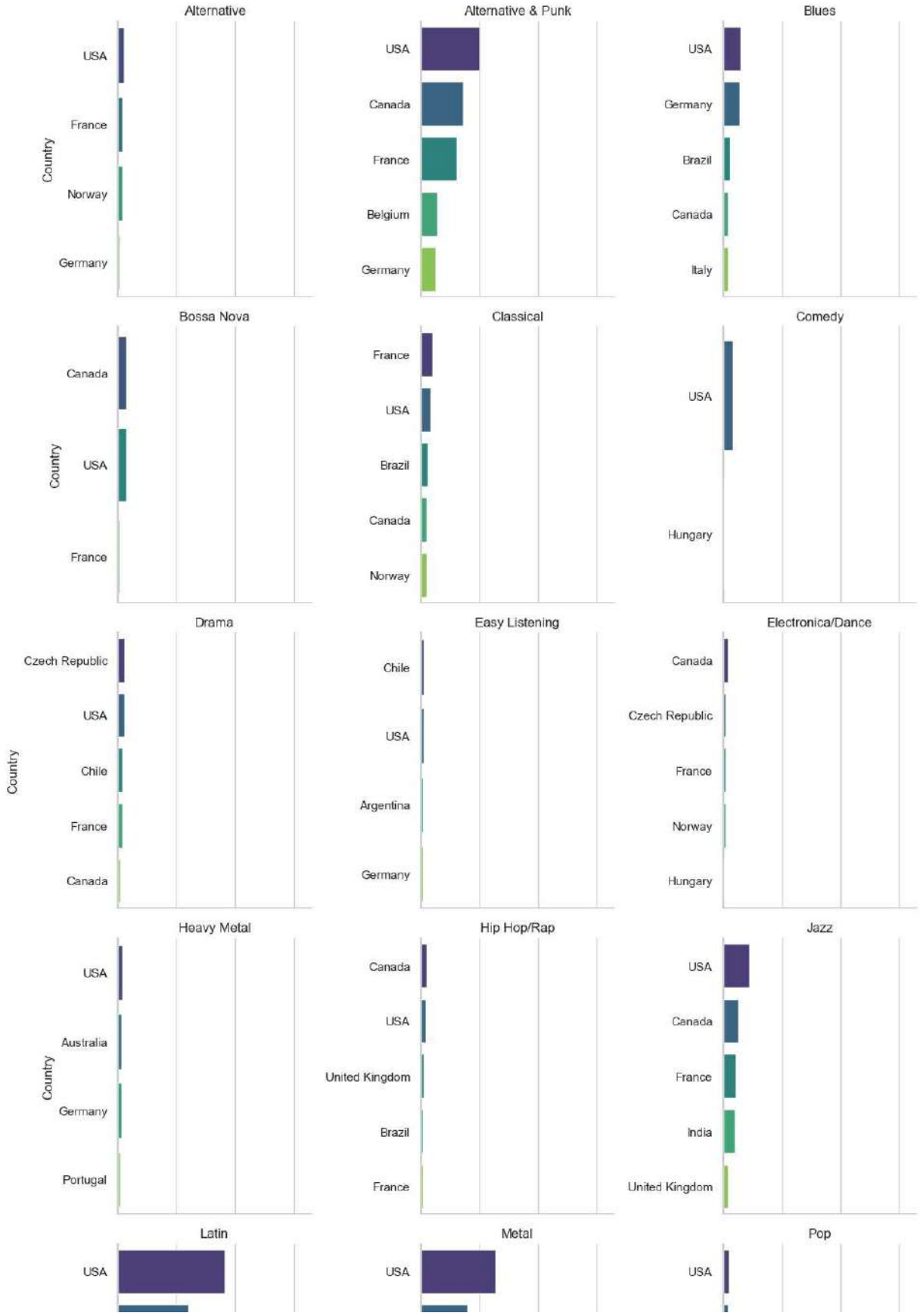
```
C:\Users\Michael\AppData\Local\Temp\ipykernel_16724\532899833.py:2: UserWarning: pan
das only supports SQLAlchemy connectable (engine/connection) or database string URI
or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider u
sing SQLAlchemy.
  df = pd.read_sql_query(query, connection)
```

Out[22]:

	GENRE_NAME	COUNTRY	PLAY_COUNT
0	Rock	Argentina	9
1	Alternative & Punk	Argentina	9
2	Latin	Argentina	8
3	Metal	Argentina	7
4	Jazz	Argentina	2

In [23]:
```
# Group by GENRE_NAME and COUNTRY, then sum the PLAY_COUNT
genre_country_counts = df.groupby(['GENRE_NAME', 'COUNTRY'])['PLAY_COUNT'].sum().re

# Get the top 5 countries for each genre
top_genres = genre_country_counts.sort_values(['GENRE_NAME', 'PLAY_COUNT'], ascendi
top_genres['Rank'] = top_genres.groupby('GENRE_NAME')['PLAY_COUNT'].rank(method='fi
top_genres_top5 = top_genres[top_genres['Rank'] <= 5]

# Set the aesthetic style of the plots
sns.set(style="whitegrid")

# Create a FacetGrid for each genre
g = sns.FacetGrid(top_genres_top5, col="GENRE_NAME", col_wrap=3, height=4, sharey=F

# Map the bar plot to the FacetGrid
g.map_dataframe(sns.barplot, x='PLAY_COUNT', y='COUNTRY', palette='viridis')

# Add titles and adjust aesthetics
g.set_titles(col_template="{col_name}")
g.set_axis_labels("Play Count", "Country")
g.fig.suptitle('Top 5 Countries by Genre (Play Count)', fontsize=16, y=1.05)

# Save the plot as a PNG file
plt.savefig('popular_genre.png', dpi=300)

# Show the plot
plt.show()
```

Top 5 Countries by Genre (Play Count)



In [24]:
```
def to_markdown(text):
  text = text.replace('•', '  *')
  return Markdown(textwrap.indent(text, '> ', predicate=lambda _: True))

genai.configure(api_key="*********************************")

import PIL.Image

img = PIL.Image.open("popular_genre.png")
```

```
model = genai.GenerativeModel('gemini-1.5-flash-latest')
response = model.generate_content(img)

response = model.generate_content(["Explain it by points in simple and clear terms.
response.resolve()
formatted_text = to_markdown(response.text)
display(formatted_text)
```

Genre-wise Top 5 Countries and Actionable Insights

Here's a breakdown of the top 5 countries for each genre, along with some actionable insights:

Genre: Alternative

- **Top 5:** USA, Canada, France, Norway, Germany
- **Insights:** The USA dominates the alternative genre. Focus on promoting artists from the USA and consider expanding to Canada for wider reach.

Genre: Alternative & Punk

- **Top 5:** USA, Canada, France, Belgium, Germany
- **Insights:** Similar to Alternative, the USA is prominent. Canada and France are strong contenders, suggesting potential for collaborations or partnerships.

Genre: Bossa Nova

- **Top 5:** Canada, USA, France, Brazil, Norway
- **Insights:** Canada leads in Bossa Nova, indicating a potential niche market. USA and France remain important, while Brazil's presence suggests a connection to Latin music fans.

Genre: Blues

- **Top 5:** USA, Germany, Brazil, Canada, Italy
- **Insights:** The USA is a powerhouse for blues. Germany, Brazil, and Canada represent strong potential, particularly for collaborations with US artists.

Genre: Classical

- **Top 5:** USA, France, Brazil, Canada, Norway
- **Insights:** USA holds the top spot, with France a significant player. Exploring collaborations with Brazilian musicians could tap into a new market.

Genre: Comedy

- **Top 5:** USA, Brazil, Canada, Norway, Hungary
- **Insights:** USA dominates comedy. Brazil, Canada, and Norway offer avenues for expansion, particularly in co-productions.

Genre: Drama

- **Top 5:** Czech Republic, USA, Chile, France, Canada

- **Insights:** Czech Republic leads the way, suggesting a focus on European markets. USA remains strong, with opportunities for co-productions or international adaptations.

Genre: Easy Listening

- **Top 5:** Chile, USA, Argentina, Germany, Denmark
- **Insights:** Chile is a surprise leader, indicating a niche market. USA and Argentina provide potential, with Germany and Denmark representing potential for European expansion.

Genre: Electronic/Dance

- **Top 5:** Canada, Czech Republic, France, Norway, Hungary
- **Insights:** Canada is leading, suggesting a strong focus on electronic music. Czech Republic and France offer possibilities for expansion, while Norway and Hungary may indicate specific niche markets.

Genre: Heavy Metal

- **Top 5:** USA, Australia, Germany, Portugal, France
- **Insights:** The USA leads the heavy metal scene. Australia and Germany are strong markets, while Portugal and France offer potential for niche outreach.

Genre: Hip Hop/Rap

- **Top 5:** USA, Canada, France, India, United Kingdom
- **Insights:** The USA is the dominant force, with Canada and France offering potential for international collaborations. India's presence suggests a growing hip hop scene there.

Genre: Jazz

- **Top 5:** USA, Canada, France, India, United Kingdom
- **Insights:** The USA is the top jazz market, with Canada and France presenting strong potential. India and the UK represent emerging markets for jazz.

Genre: Latin

- **Top 5:** USA, Canada, Brazil, United Kingdom, France
- **Insights:** The USA is the leading country for Latin music. Canada, Brazil, and the UK represent potential for collaborations or expansion.

Genre: Metal

- **Top 5:** USA, Canada, Brazil, United Kingdom, France
- **Insights:** The USA holds the top spot, with Canada and Brazil as key players. The UK and France offer potential for niche market exploration.

Genre: Pop

- **Top 5:** USA, Canada, Germany, France, United Kingdom
- **Insights:** The USA is the dominant force, with Canada and Germany providing strong opportunities. France and the UK represent potentially lucrative markets.

Genre: R&B/Soul

- **Top 5:** USA, Canada, Brazil, Hungary, Austria
- **Insights:** The USA is a top contender, with Canada and Brazil being prominent markets. Hungary and Austria represent niche markets for R&B/Soul.

Genre: Reggae

- **Top 5:** USA, Canada, Brazil, Czech Republic, Denmark
- **Insights:** The USA holds the top spot, with Canada and Brazil offering potential for expansion. Czech Republic and Denmark represent smaller markets for reggae.

Genre: Rock

- **Top 5:** USA, Canada, Brazil, France, Germany
- **Insights:** The USA is the leading country for rock music. Canada and Brazil are major players, while France and Germany offer possibilities for outreach.

Genre: Rock And Roll

- **Top 5:** USA, Canada, Finland, Netherlands, Brazil
- **Insights:** The USA is a top rock and roll market, with Canada following close behind. Finland, Netherlands, and Brazil represent niche markets for the genre.

Genre: Sci Fi & Fantasy

- **Top 5:** USA, Australia, Finland, Netherlands, Brazil
- **Insights:** The USA is the dominant force, with Australia offering significant potential. Finland, Netherlands, and Brazil represent smaller markets for sci-fi and fantasy.

Genre: Science Fiction

- **Top 5:** Germany, Brazil, France, Chile, Czech Republic
- **Insights:** Germany is leading the way, suggesting a strong demand for science fiction. Brazil and France offer potential for collaborations, with Chile and Czech Republic representing smaller markets.

Genre: Soundtrack

- **Top 5:** France, Germany, Brazil, USA, Argentina
- **Insights:** France leads the way, indicating a strong market for soundtracks. Germany, Brazil, and the USA offer significant potential, while Argentina represents a niche market.

Genre: TV Shows

- **Top 5:** USA, Czech Republic, Ireland, Austria, Hungary
- **Insights:** The USA is the dominant force, with Czech Republic, Ireland, Austria, and Hungary representing smaller markets for TV show soundtracks.

Genre: World

- **Top 5:** USA, Canada, Brazil, Portugal, Norway
- **Insights:** The USA holds the top spot, with Canada and Brazil offering potential for expansion. Portugal and Norway represent smaller markets for world music.

Total Sales by Country

In [25]:
```
query = "SELECT * FROM public.total_sales_by_country"
df = pd.read_sql_query(query, connection)
df.head(5)
```

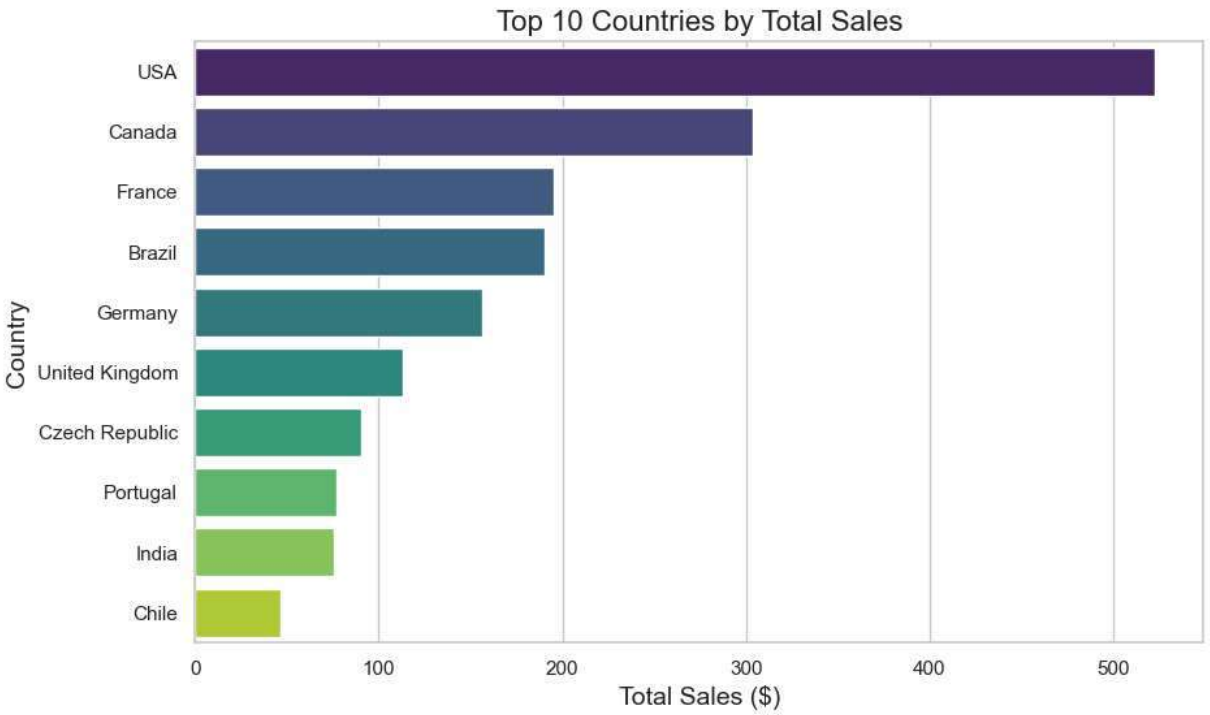
```
C:\Users\Michael\AppData\Local\Temp\ipykernel_16724\1511912243.py:2: UserWarning: pa
ndas only supports SQLAlchemy connectable (engine/connection) or database string URI
or sqlite3 DBAPI2 connection. Other DBAPI2 objects are not tested. Please consider u
sing SQLAlchemy.
  df = pd.read_sql_query(query, connection)
```

Out[25]:

	COUNTRY	TOTAL_SALES
0	USA	523.06
1	Canada	303.96
2	France	195.10
3	Brazil	190.10
4	Germany	156.48

In [26]:
```
# Sort the DataFrame by TOTAL_SALES and get the top 10 countries (in this case, onl
top_sales = df.sort_values(by='TOTAL_SALES', ascending=False).head(10)

# Set the aesthetic style of the plots
sns.set(style="whitegrid")
```

```
# Create a horizontal bar plot
plt.figure(figsize=(10, 6))
sns.barplot(x='TOTAL_SALES', y='COUNTRY', data=top_sales, palette='viridis')

# Add title and labels
plt.title('Top 10 Countries by Total Sales', fontsize=16)
plt.xlabel('Total Sales ($)', fontsize=14)
plt.ylabel('Country', fontsize=14)

# Save the plot as a PNG file
plt.savefig('top_countries_sales.png', dpi=300)

# Show the plot
plt.show()
```



In [27]:

```
def to_markdown(text):
  text = text.replace('•', '  *')
  return Markdown(textwrap.indent(text, '> ', predicate=lambda _: True))

genai.configure(api_key="*******************************")

import PIL.Image

img = PIL.Image.open("top_countries_sales.png")
model = genai.GenerativeModel('gemini-1.5-flash-latest')
response = model.generate_content(img)

response = model.generate_content(["Explain it by points in simple and clear terms.
response.resolve()
formatted_text = to_markdown(response.text)
display(formatted_text)
```

Top 10 Countries by Total Sales: Key Findings and Actionable Insights

Key Findings:

- **USA dominates sales:** The USA has significantly higher sales than any other country, indicating a strong market presence and potential for further growth.
- **Canada follows closely:** Canada represents a substantial market with potential for expansion, given its proximity to the USA and similar consumer demographics.
- **European markets show promise:** France, Brazil, and Germany demonstrate strong sales, suggesting that expanding in Europe could be a successful strategy.
- **Emerging markets are smaller but growing:** Countries like Czech Republic, Portugal, India, and Chile have lower sales but show potential for future growth as their economies develop.

Actionable Insights:

- **Focus on USA & Canada:** Prioritize these markets with targeted marketing campaigns and product offerings tailored to local preferences.
- **Explore European expansion:** Research opportunities in France, Brazil, and Germany to capitalize on existing demand.
- **Invest in emerging markets:** Monitor and potentially invest in markets like Czech Republic, Portugal, India, and Chile to tap into future growth potential.
- **Customize product offerings:** Tailor product features and marketing messaging to resonate with the specific needs and preferences of each country.
- **Localize marketing:** Utilize local language, cultural references, and marketing channels to reach the target audience effectively.

Overall:

The data suggests that focusing on the USA and Canada while simultaneously exploring expansion opportunities in Europe and emerging markets could lead to significant growth and success.