

CHINOOK MUSIC SALES ANALYSIS AND REPORT



[This Photo](#) by Unknown Author is licensed under [CC BY-NC-ND](#)

The Chinook Music Sales Report provides an in-depth analysis of the store's digital media sales from 2007 to 2010. By leveraging SQL Server Management Studio for data extraction and Power BI for visualizations, this report examines critical metrics such as top-performing artists, customer purchasing behaviours, popular genres, sales trends, and customer lifetime value (CLV). The objective of this analysis is to generate actionable insights that can inform business strategies aimed at boosting sales, improving customer retention, and driving profitability.

Project Overview

In this capstone project, we conducted an analysis of the Chinook sample database, which is structured to represent a digital media store. Utilizing SQL through SQL Server Management Studio (SSMS) and Microsoft Power

BI for visualization, the objective was to extract valuable insights from the data and effectively communicate the findings through visual reports.

Objectives

Data Exploration: Gain a thorough understanding of the database structure and its contents.

Data Analysis: Conduct targeted analyses to extract meaningful insights from the dataset.

Visualization: Develop visual reports to effectively convey the findings.

Presentation: Compile and present the results in a detailed report or interactive dashboard.

Use Cases:

The project adopted the use cases below:

Top-Selling Artists: Identify artists with the highest sales and analyze their sales trends over time.

Customer Purchase Patterns: Segment customers based on purchase behavior and identify key characteristics of high-value customers.

Genre Popularity: Determine the most popular music genres and analyze the genre popularity changes over different periods.

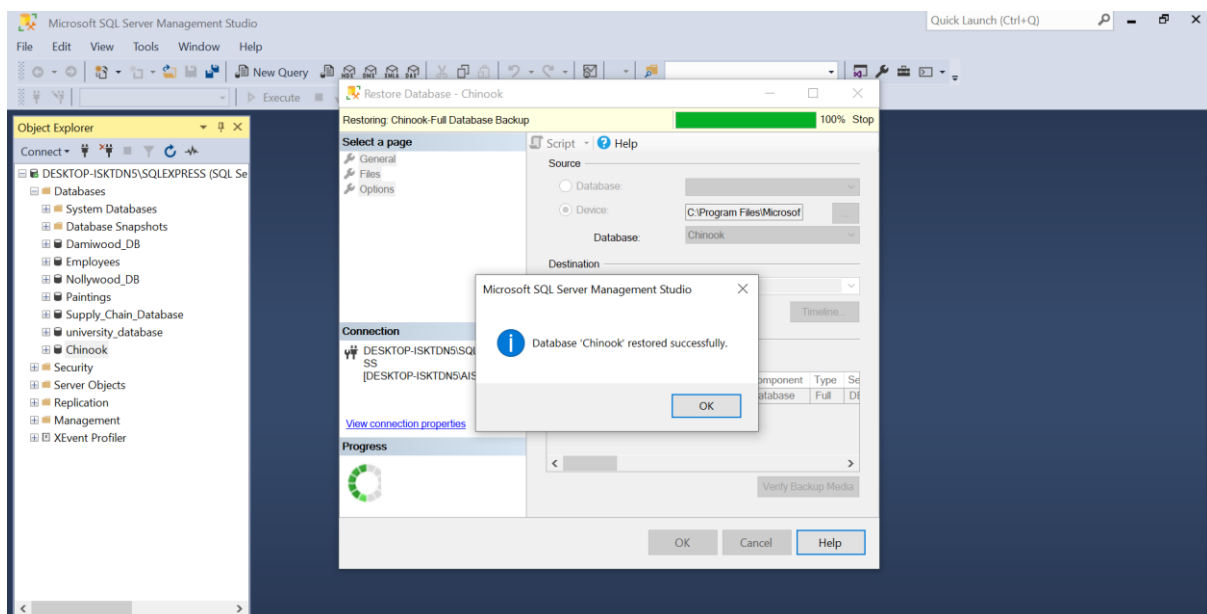
Sales Over Time: Analyze monthly and yearly sales trends, including seasonal effects.

Customer Lifetime Value (CLV): Calculate the lifetime value of customers based on their purchase history and provide recommendations for improving customer retention.

Data Overview

As previously mentioned, the Chinook database is a sample dataset modelled after a digital media store, covering sales data from 2007 to 2010. The dataset was obtained online from GitHub. It includes eleven (11) tables: Album, Artist, Customer, Employee, Genre, Invoice, InvoiceLine, MediaType, Playlist, PlaylistTrack, and Track.

The backup file (.bak) was sourced from Chris Woodruff ("Woody") and restored in SQL Server Management Studio for analysis.



Methodology

We conducted a thorough exploration of the dataset, which spanned 11 tables. This initial analysis provided us with insights into the dataset's structure and allowed us to identify key patterns and trends.

The tools utilized for this analysis included SQL Server Management Studio, Excel, and Power BI.

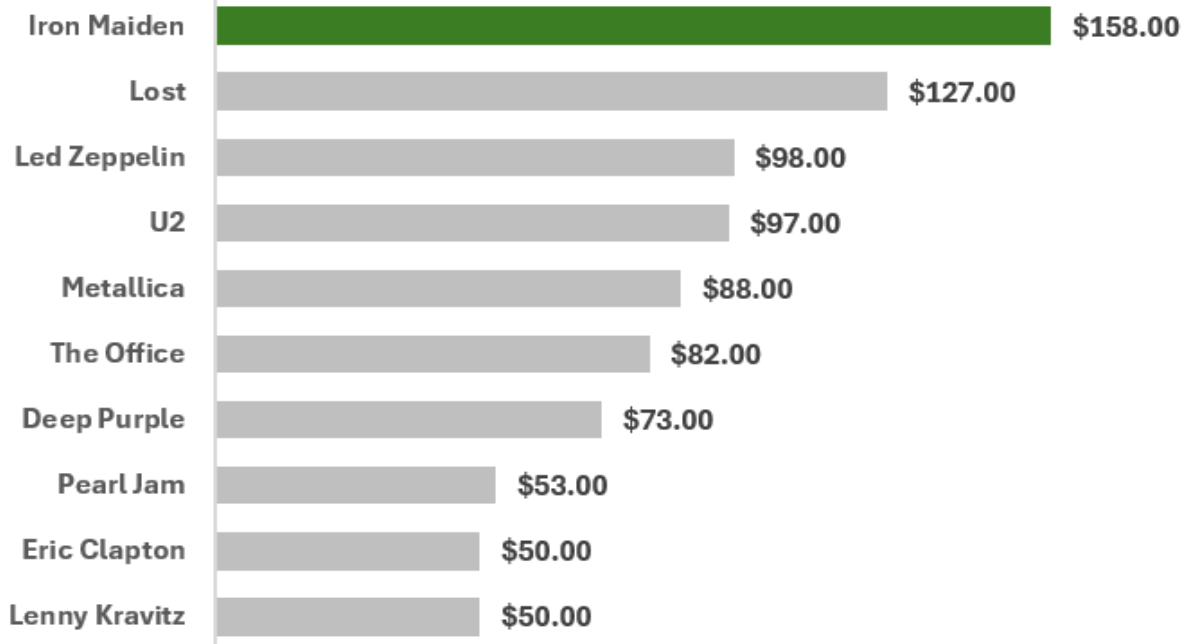
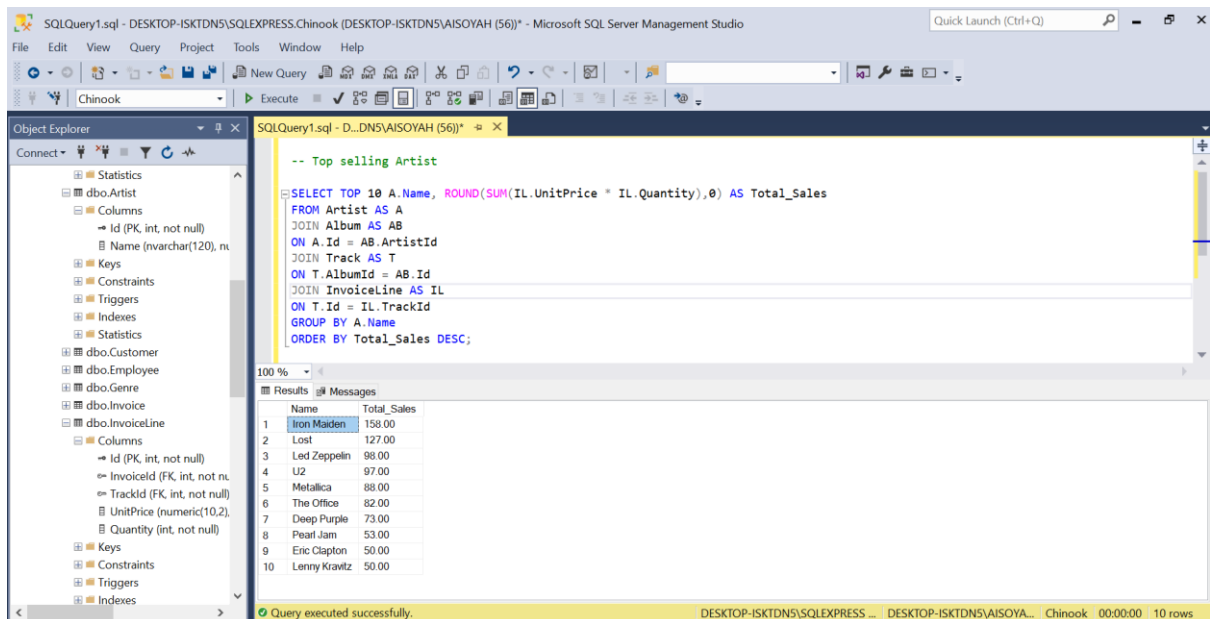
Key metrics examined were the total number of artists, the albums they produced, the revenue generated from store sales, as well as the countries, media types, and music genres involved.

Analysis

Best-Selling Artists: We determined which artists had the highest sales and examined the trends in their sales over time.

The following SQL query was used to identify the top 10 artists with the most album sales.

```
SELECT TOP 10 A.Name, ROUND(SUM(IL.UnitPrice * IL.Quantity),0) AS  
Total_Sales  
  
FROM Artist AS A  
  
JOIN Album AS AB  
  
ON A.Id = AB.ArtistId  
  
JOIN Track AS T  
  
ON T.AlbumId = AB.Id  
  
JOIN InvoiceLine AS IL  
  
ON T.Id = IL.TrackId  
  
GROUP BY A.Name  
  
ORDER BY Total_Sales DESC;
```



Top 10 Best Selling Artist

The chart above shows that the artist with the highest sales was Iron Maiden. The revenue generated from the sales of albums belonging to Iron Maiden was to the tune of \$158. This was followed by Lost with the sales amount of \$127, Led Zeppelin with \$98, and U2 with \$97.

To understand and gain insight into the sales trend over time, we used the query below:

```
SELECT

    DATEPART(MONTH, InvoiceDate) AS Month_Number,

    DATENAME(MONTH, InvoiceDate) AS Month_Name,

    SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales

FROM Album AS AB

JOIN TRACK AS T

ON AB.Id = T.AlbumId

JOIN InvoiceLine AS IL

ON T.Id = IL.TrackId

JOIN Invoice AS I

ON IL.InvoiceId=I.Id

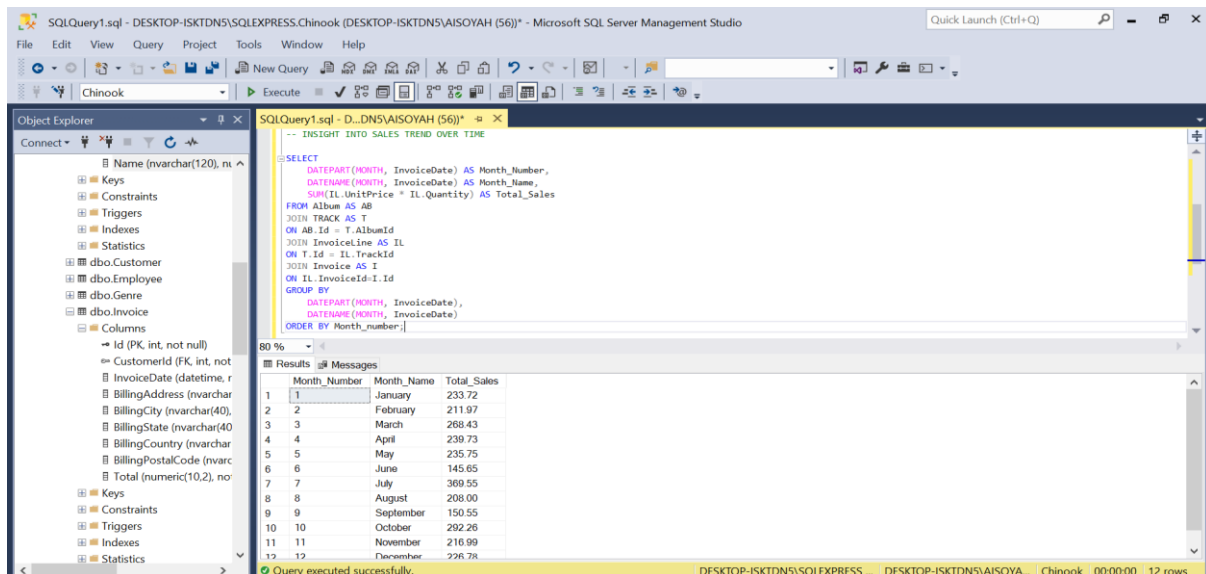
GROUP BY

    DATEPART(MONTH, InvoiceDate),

    DATENAME(MONTH, InvoiceDate)

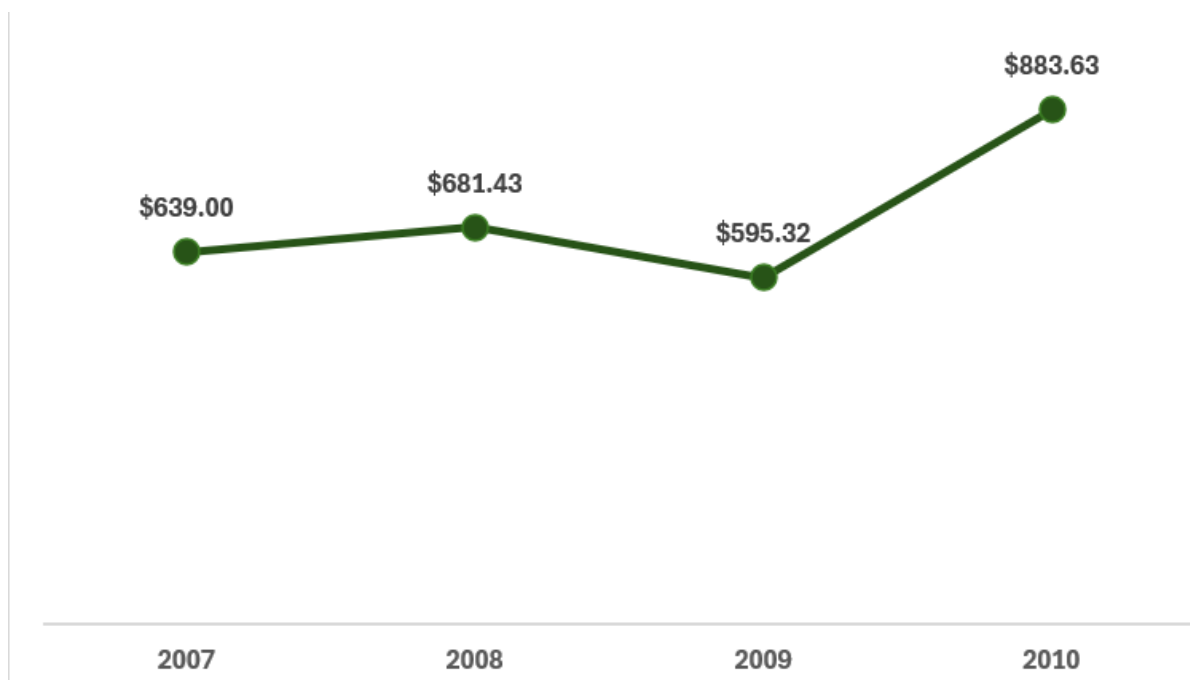
ORDER BY Month_number;
```

This result from the query was visualized using Microsoft Excel. The line chart shows the trend in sales across the years under review. July was the month with the highest sales of \$370, followed by October with \$292 in sales amount. The months of June and September were the months with the lowest sales with \$146 and \$151 respectively.



-- YEARLY SALES TREND

```
SELECT
    DATEPART(YEAR, InvoiceDate) AS Year_Number,
    SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales
FROM Album AS AB
JOIN TRACK AS T
ON AB.Id = T.AlbumId
JOIN InvoiceLine AS IL
ON T.Id = IL.TrackId
JOIN Invoice AS I
ON IL.InvoiceId=I.Id
GROUP BY
    DATEPART(YEAR, InvoiceDate)
ORDER BY Year_Number;
```



The chart shows that 2010 was the year with the most sales. However, 2009 recorded the lowest sales.

■ QUARTERLY SALES TREND

SELECT

CASE

WHEN DATEPART(QUARTER, InvoiceDate) = 1 THEN 'Q1'

WHEN DATEPART(QUARTER, InvoiceDate) = 2 THEN 'Q2'

WHEN DATEPART(QUARTER, InvoiceDate) = 3 THEN 'Q3'

ELSE 'Q4'

END AS Quarter_Name,

DATEPART(YEAR, InvoiceDate) AS Year,

SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales

FROM Album AS AB

JOIN TRACK AS T

ON AB.Id = T.AlbumId

JOIN InvoiceLine AS IL

ON T.Id = IL.TrackId

JOIN Invoice AS I

ON IL.InvoiceId=I.Id

GROUP BY DATEPART(QUARTER, InvoiceDate),

DATEPART(YEAR, InvoiceDate)

ORDER BY Quarter_Name, Year;

Quarter_Name	Year	Total_Sales
Q1	2007	\$188.20
Q1	2008	\$194.11
Q1	2009	\$105.97
Q1	2010	\$225.84
Q2	2007	\$165.45
Q2	2008	\$88.15
Q2	2009	\$138.71
Q2	2010	\$228.82
Q3	2007	\$152.53
Q3	2008	\$239.71
Q3	2009	\$145.63
Q3	2010	\$190.23
Q4	2007	\$132.82
Q4	2008	\$159.46
Q4	2009	\$205.01
Q4	2010	\$238.74

Sales by Quarter over the Years

Customer Purchase Patterns: We segmented customers based on their buying behaviour to uncover the key traits of high-value customers.

We used the RFM (Recency, Frequency, Monetary value) model for this analysis, which examines customer behaviour based on how recently they made a purchase, how often they purchase, and the amount spent.

Recency

To understand how recent a customer's purchase was, we grouped them by the year of their last purchase (Inactive, Less Recent, Recent, and Most Recent). Customers with their last purchase in 2007 were labelled as inactive, while those who made a purchase in 2010 were categorized as most recent.

```

SELECT
    CONCAT(C.firstName, ' ', C.LastName) AS Full_name,
    MAX(CAST(I.invoiceDate AS Date)) AS Last_Purchase,
    CASE
        WHEN YEAR (MAX(I.invoiceDate)) = '2010' THEN 'Most Recent'
        WHEN YEAR (MAX(I.invoiceDate)) = '2009' THEN 'Recent'
        WHEN YEAR (MAX(I.invoiceDate)) = '2008' THEN 'Less Recent'
        ELSE 'Inactive'
    END AS Recency
FROM Invoice AS I
JOIN Customer AS C
ON c.Id = i.CustomerId
GROUP BY CONCAT(C.firstName, ' ', C.LastName), I.InvoiceDate
ORDER BY Last_Purchase DESC;

```

Customers such as Eduardo Martins, Lucas Mancini, Jack Smith, Fynn Zimmerman, and Michelle Brooks were identified as the most recent purchasers. In contrast, John Gordon, Joao Fernandes, Hugh O'Reilly, Victor Stevens, and Luis Goncalves were categorized as inactive customers.

FREQUENCY

I categorized this based on the number of times a customer purchased within the year under review. To achieve this, we grouped the customers into three categories. ***Those who made 5 purchases or fewer were categorized as Less Frequent, and those between 6 and 10 were***

categorized as Frequent while those who made above 10 purchases were categorized as Most Frequent customers.

SELECT

CONCAT(C.firstName, ' ', C.LastName) AS Full_name,

COUNT(I.Id) Num_of_Purchase,

CASE

WHEN COUNT(I.Id) <= 5 THEN 'Less Frequent'

WHEN COUNT(I.Id) BETWEEN 5 AND 10 THEN 'Frequent'

ELSE 'Most Frequent'

END AS Frequency

FROM Invoice AS I

JOIN Customer AS C

ON C.Id = I.CustomerId

GROUP BY c.Id, c.FirstName, c.LastName

ORDER BY Num_of_Purchase DESC;

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The left pane displays the 'Object Explorer' with a tree view of the 'Chinook' database schema, including tables like Customer, Invoice, and Employee. The right pane shows a SQL query window with the following query:

```
SELECT
    CONCAT(C.firstName, ' ', C.LastName) AS Full_name,
    COUNT(I.Id) Num_of_Purchase,
    CASE
        WHEN COUNT(I.Id) <= 5 THEN 'Less Frequent'
        WHEN COUNT(I.Id) BETWEEN 5 AND 10 THEN 'Frequent'
        ELSE 'Most Frequent'
    END AS Frequency
FROM Invoice AS I
JOIN Customer AS C
ON C.Id = I.CustomerId
GROUP BY c.Id, c.FirstName, c.LastName
ORDER BY Num_of_Purchase DESC;
```

Below the query window, the 'Results' pane displays the output of the query as a table with 11 rows. The status bar at the bottom indicates 'Query executed successfully.' and '58 rows'.

	Full_name	Num_of_Purchase	Frequency
1	Leonie Köhler	15	Most Frequent
2	Eduardo Martins	14	Most Frequent
3	Fynn Zimmermann	13	Most Frequent
4	João Fernandes	12	Most Frequent
5	Robert Brown	11	Most Frequent
6	Dan Miller	11	Most Frequent
7	Terhi Härmäläinen	11	Most Frequent
8	Hugh O'Reilly	11	Most Frequent
9	Kathy Chase	10	Frequent
10	John Gordon	10	Frequent
11	Richard Cunningham	10	Frequent

Leonie Köhler had the highest number of purchases, with Eduardo Martin following closely behind, placing both in the 'Most Frequent' customer category. On the other hand, Madalena Sampaio, Tim Goyer, and Mark Taylor were among the 'Less Frequent' customers due to their lower number of purchases.

Monetary Value

Customers were classified into three categories based on their total sales amount. Those who spent \$45 or less were labelled as low-spending customers, while those with purchases between \$46 and \$90 were considered mid-spending customers. Customers who spent more than \$90 were placed in the high-spending category.

```
SELECT

CONCAT(c.firstName, ' ', c.LastName) AS Full_name,

COUNT(I.CustomerId) Num_of_Purchase,

SUM(I.Total) AS Total_Sum_Purchased,

CASE

    WHEN SUM(I.Total) <= 45 THEN 'Low-Value Customer'

    WHEN SUM(I.Total) BETWEEN 46 AND 90 THEN 'Mid-Valued Customer'

    ELSE 'High-Value Customer'

END AS Customer_Value

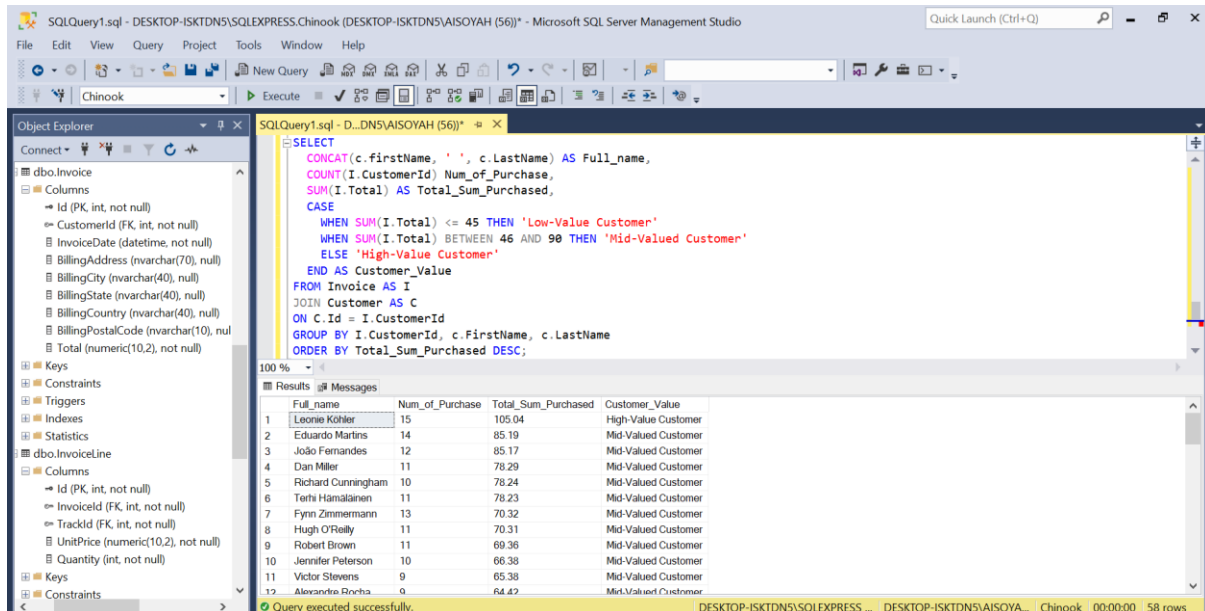
FROM Invoice AS I

JOIN Customer AS C

ON C.Id = I.CustomerId
```

GROUP BY I.CustomerId, c.FirstName, c.LastName

ORDER BY Total_Sum_Purchased DESC;



The screenshot shows the Microsoft SQL Server Management Studio interface. The left pane displays the 'Object Explorer' with the 'dbo.Invoice' table selected. The right pane shows the 'SQLQuery1.sql' file with the following query:

```
SELECT
    CONCAT(c.firstName, ' ', c.LastName) AS Full_name,
    COUNT(I.CustomerId) AS Num_of_Purchase,
    SUM(I.Total) AS Total_Sum_Purchased,
    CASE
        WHEN SUM(I.Total) <= 45 THEN 'Low-Value Customer'
        WHEN SUM(I.Total) BETWEEN 46 AND 90 THEN 'Mid-Valued Customer'
        ELSE 'High-Value Customer'
    END AS Customer_Value
FROM Invoice AS I
JOIN Customer AS C
ON C.Id = I.CustomerId
GROUP BY I.CustomerId, c.FirstName, c.LastName
ORDER BY Total_Sum_Purchased DESC;
```

The 'Results' pane shows the output of the query, which is a table with 4 columns: Full_name, Num_of_Purchase, Total_Sum_Purchased, and Customer_Value. The results are ordered by Total_Sum_Purchased in descending order.

	Full_name	Num_of_Purchase	Total_Sum_Purchased	Customer_Value
1	Leonie Köhler	15	105.04	High-Value Customer
2	Eduardo Martins	14	85.19	Mid-Valued Customer
3	João Fernandes	12	85.17	Mid-Valued Customer
4	Dan Miller	11	78.29	Mid-Valued Customer
5	Richard Cunningham	10	78.24	Mid-Valued Customer
6	Tertti Hämäläinen	11	78.23	Mid-Valued Customer
7	Fynn Zimmermann	13	70.32	Mid-Valued Customer
8	Hugh O'Reilly	11	70.31	Mid-Valued Customer
9	Robert Brown	11	69.36	Mid-Valued Customer
10	Jennifer Peterson	10	66.38	Mid-Valued Customer
11	Victor Stevens	9	65.38	Mid-Valued Customer
12	Alexandre Rocha	0	64.42	Mid-Valued Customer

The status bar at the bottom indicates 'Query executed successfully.' and '58 rows'.

It's not surprising that Leonie Köhler stands out as the only high-value customer, as he made the highest number of purchases, totaling \$105. The analysis also reveals that 57.6% of the customers fall into the mid-value category.

Top 5 Customers

SELECT TOP 5

CONCAT(c.firstName, ' ', c.LastName) AS Full_name,

COUNT(I.CustomerId) AS Num_of_Purchase,

SUM(I.Total) AS Total_Sum_Purchased,

ROUND(AVG(I.Total),2) AS Average_Purchase

FROM Invoice AS I

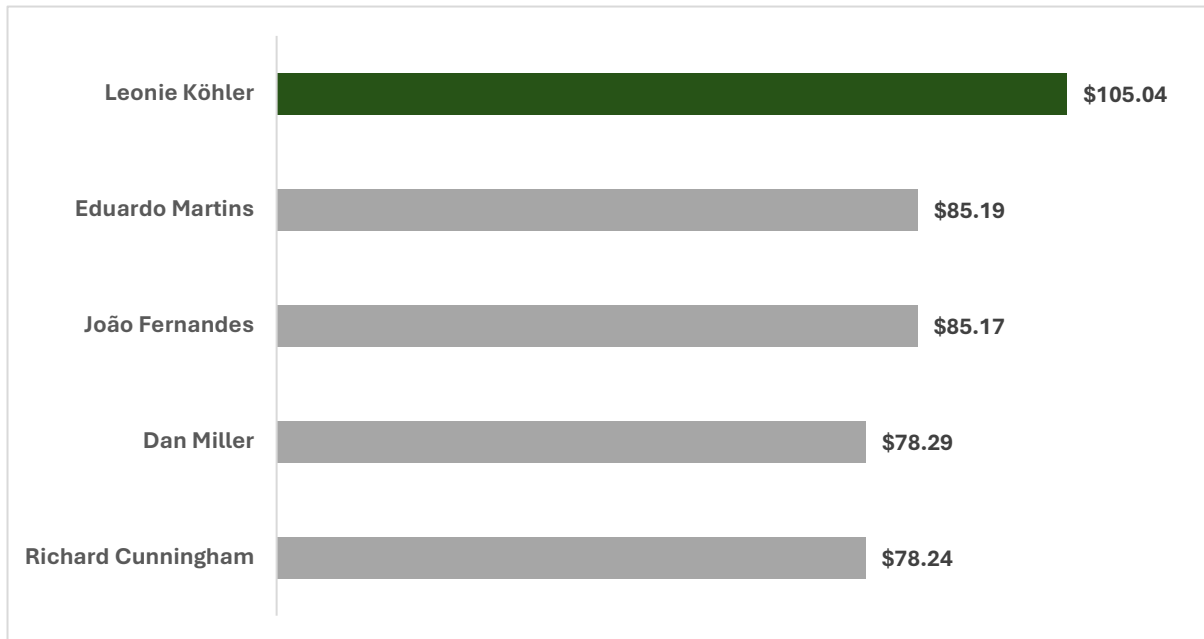
JOIN Customer AS C

ON C.Id = I.CustomerId

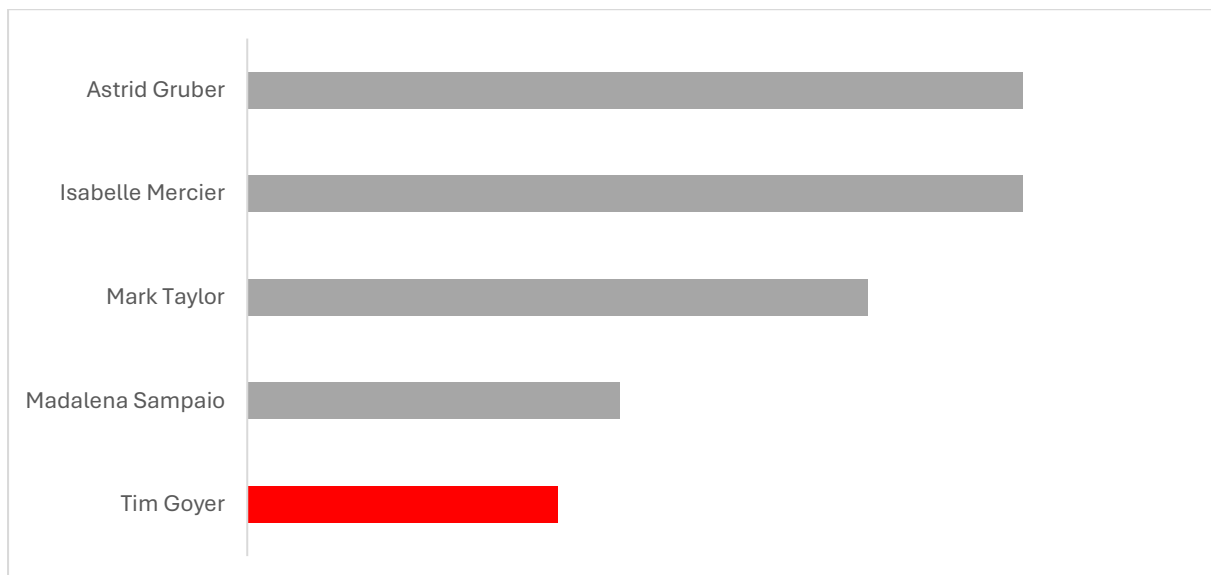
GROUP BY c.Id, c.FirstName, c.LastName

HAVING COUNT(c.Id) IN (10, 11, 12, 14, 15)

ORDER BY Total_Sum_Purchased DESC;



TOP 5 customers



Bottom 5 customers

Genre Popularity: Determine the most popular music genres and analyse the genre popularity changes over different periods.

-- FREQUENTLY SOLD GENRE AND AMOUNT GENERATED

SELECT

G.name AS Genre_name,

COUNT(IL.trackid) AS NumTrack_Sold,

ROUND(SUM(IL.UnitPrice * il.Quantity), 0) AS Total_Sales

FROM Genre As G

LEFT JOIN Track AS T

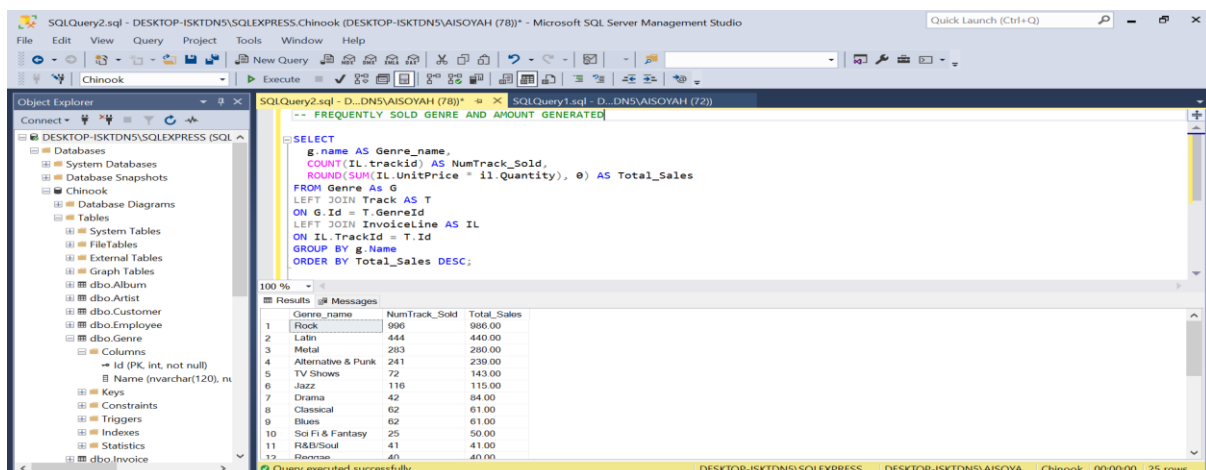
ON G.Id = T.GenreId

LEFT JOIN InvoiceLine AS IL

ON IL.TrackId = T.Id

GROUP BY g.Name

ORDER BY Total_Sales DESC;



The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL query:

```
-- FREQUENTLY SOLD GENRE AND AMOUNT GENERATED

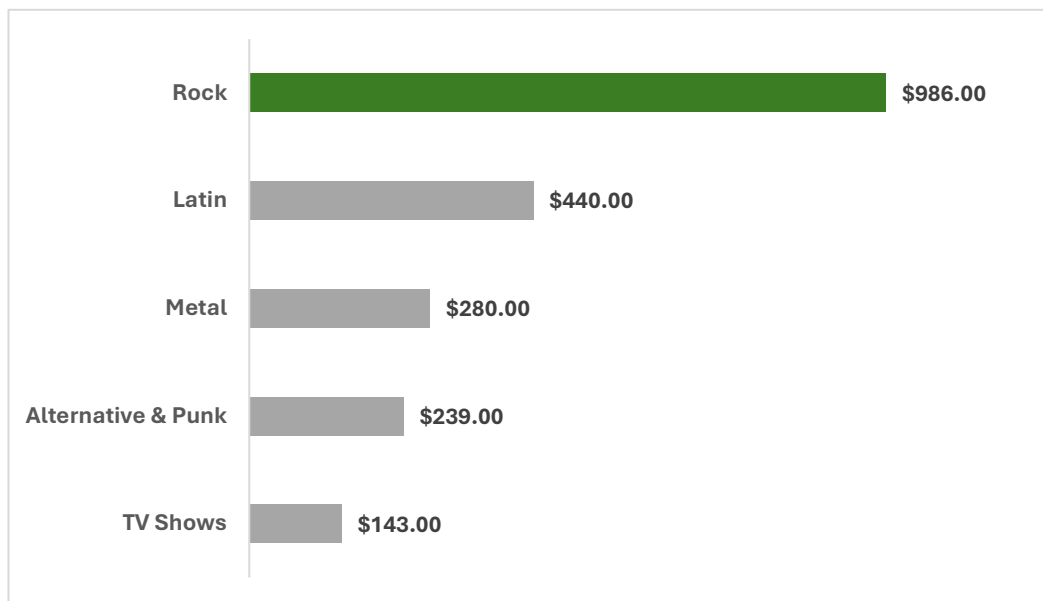
SELECT
    g.name AS Genre_name,
    COUNT(IL.trackid) AS NumTrack_Sold,
    ROUND(SUM(IL.UnitPrice * il.Quantity), 0) AS Total_Sales
FROM Genre As G
LEFT JOIN Track AS T
ON G.Id = T.GenreId
LEFT JOIN InvoiceLine AS IL
ON IL.TrackId = T.Id
GROUP BY g.Name
ORDER BY Total_Sales DESC;
```

The Results pane shows the following data:

	Genre_name	NumTrack_Sold	Total_Sales
1	Rock	996	986.00
2	Latin	444	440.00
3	Metal	263	290.00
4	Alternative & Punk	241	239.00
5	TV Shows	72	143.00
6	Jazz	116	115.00
7	Drama	42	64.00
8	Classical	62	61.00
9	Blues	62	61.00
10	Sci Fi & Fantasy	25	50.00
11	R&B/Soul	41	41.00
12	Electronic	40	40.00

The status bar at the bottom indicates: Query executed successfully. DESKTOP-ISKTDN5\SQLEXPRESS ... DESKTOP-ISKTDN5\VAISOYA... Chinook 00:00:00 25 rows

Sales from the rock music genre brought in \$986, making it the top-selling genre. Latin music came in second, generating \$444 in revenue. The opera genre did not have any sales during the review period. Rock and roll, with only \$8 in revenue, was the least popular genre and had the lowest sales among the 25 genres in the music store.



Sales Over Time: Examine sales patterns on a monthly and yearly basis, factoring in any seasonal trends.

Since we had previously created a query for a similar analysis, we only needed to adjust rather than writing new queries from scratch. These modifications enabled us to track the number of albums sold during each period — both yearly and monthly. Additionally, we included a query to identify which day of the week produced the highest revenue and album sales.

SELECT

DATEPART(YEAR, InvoiceDate) AS Month_Number,

SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales,

COUNT(I.Id) AS Number_of_Sales

FROM Album AS AL

JOIN Track AS T

ON T.AlbumId = AL.Id

JOIN InvoiceLine AS IL

ON T.Id = IL.TrackId

JOIN Invoice AS I

ON IL.InvoiceId = I.Id

GROUP BY DATEPART(YEAR, InvoiceDate)

ORDER BY Month_Number;

The screenshot displays the Microsoft SQL Server Management Studio interface. The 'Object Explorer' on the left shows the database structure, including tables like Album, Track, InvoiceLine, and Invoice. The central pane shows a SQL query that calculates total sales by year, grouped by the year extracted from the InvoiceDate. The 'Results' pane at the bottom shows the output of the query, which is a table with four rows representing the years 2007 through 2010, with columns for Month_Number, Total_Sales, and Number_of_Sales.

```
SELECT
    DATEPART(YEAR, InvoiceDate) AS Month_Number,
    SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales,
    COUNT(I.Id) AS Number_of_Sales
FROM Album AS AL
JOIN Track AS T
ON T.AlbumId = AL.Id
JOIN InvoiceLine AS IL
ON T.Id = IL.TrackId
JOIN Invoice AS I
ON IL.InvoiceId = I.Id
GROUP BY DATEPART(YEAR, InvoiceDate)
ORDER BY Month_Number;
```

Month_Number	Total_Sales	Number_of_Sales
2007	639.00	600
2008	681.43	657
2009	595.32	568
2010	883.63	837

```
SELECT

    DATEPART(MONTH, InvoiceDate) AS Month_Number,

    DATENAME(MONTH, invoiceDate) AS Month_Name,

    SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales,

    COUNT(I.Id) AS Number_of_Sales

FROM Album AS AL

JOIN Track AS T

ON T.AlbumId = AL.Id

JOIN InvoiceLine AS IL

ON T.Id = IL.TrackId

JOIN Invoice AS I

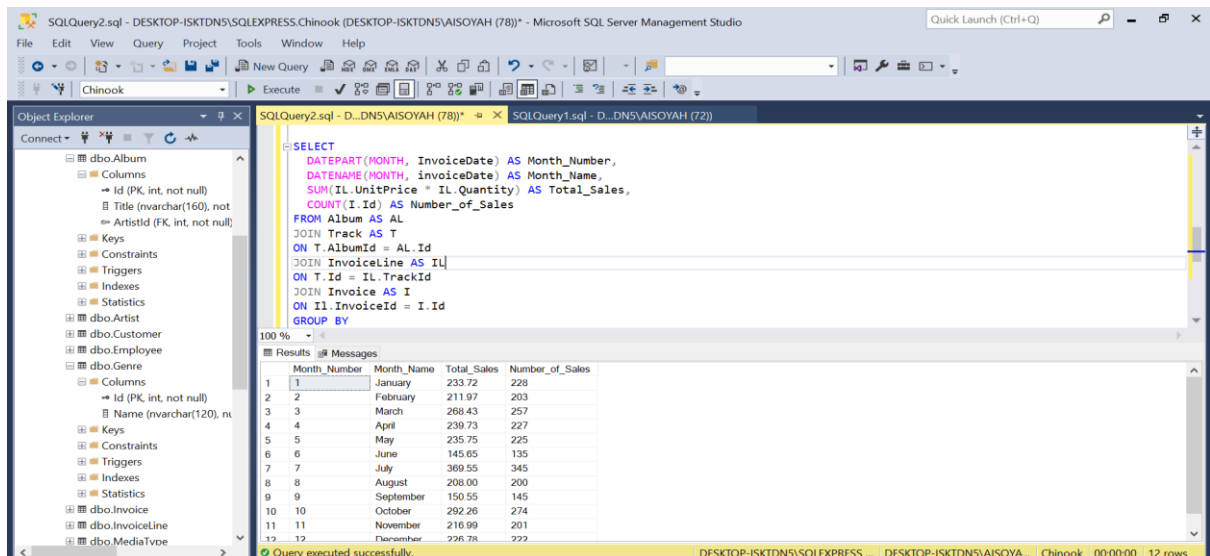
ON IL.InvoiceId = I.Id

GROUP BY

    DATENAME(MONTH, invoiceDate),

    DATEPART(MONTH, InvoiceDate)

ORDER BY Month_Number;
```



SELECT

DATEPART(WEEKDAY, InvoiceDate) AS Num_Weekdays,

DATENAME(WEEKDAY, InvoiceDate) AS Weekdays,

SUM(IL.UnitPrice * IL.Quantity) AS Total_Sales,

COUNT(I.Id) AS Number_of_Sales

FROM Album AS AL

JOIN Track AS T

ON T.AlbumId = AL.Id

JOIN InvoiceLine AS IL

ON T.Id = IL.TrackId

JOIN Invoice AS I

ON IL.InvoiceId = I.Id

GROUP BY

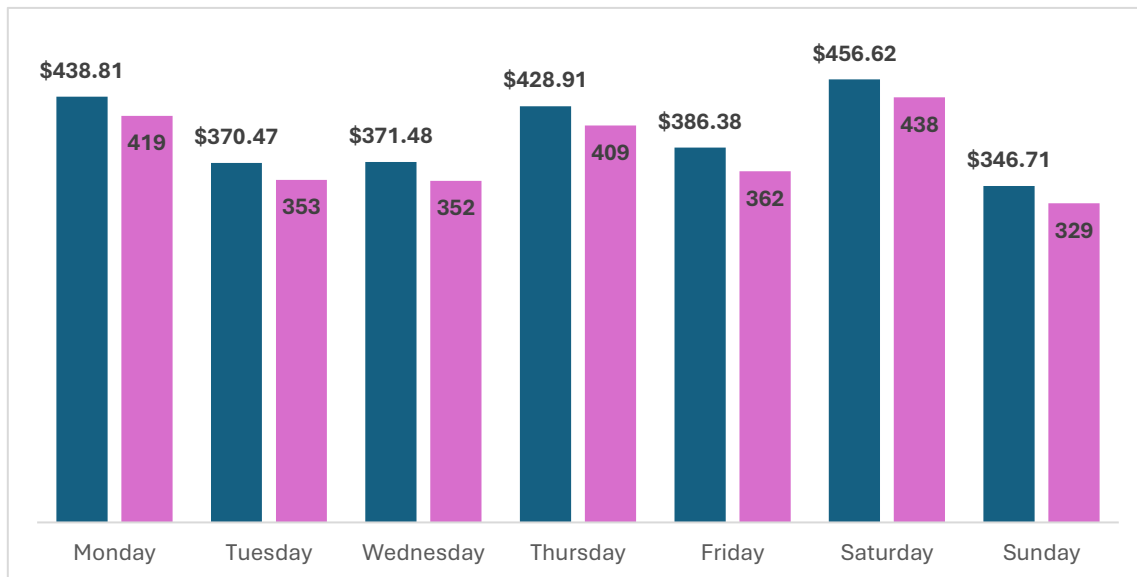
DATENAME(WEEKDAY, InvoiceDate),

DATEPART(WEEKDAY, InvoiceDate)

ORDER BY

DATENAME(WEEKDAY, InvoiceDate),

DATEPART(WEEKDAY, InvoiceDate)



The result shows that most sales were made on Saturdays, followed by Mondays and Thursdays.

Customer Lifetime Value (CLV): Calculate the lifetime value of customers based on their purchase history and provide recommendations for improving customer retention.

Customer Lifetime Value (CLV) is a metric that estimates the total revenue a business can expect to earn from a customer over the entire duration of their relationship. This is crucial for understanding customer profitability and making informed decisions about customer acquisition, retention strategies, and marketing investments.

WITH Customer_Lifetime_Value AS (

SELECT

C.CustomerId,

CONCAT(C.FirstName, ' ', C.LastName) AS Full_Name,

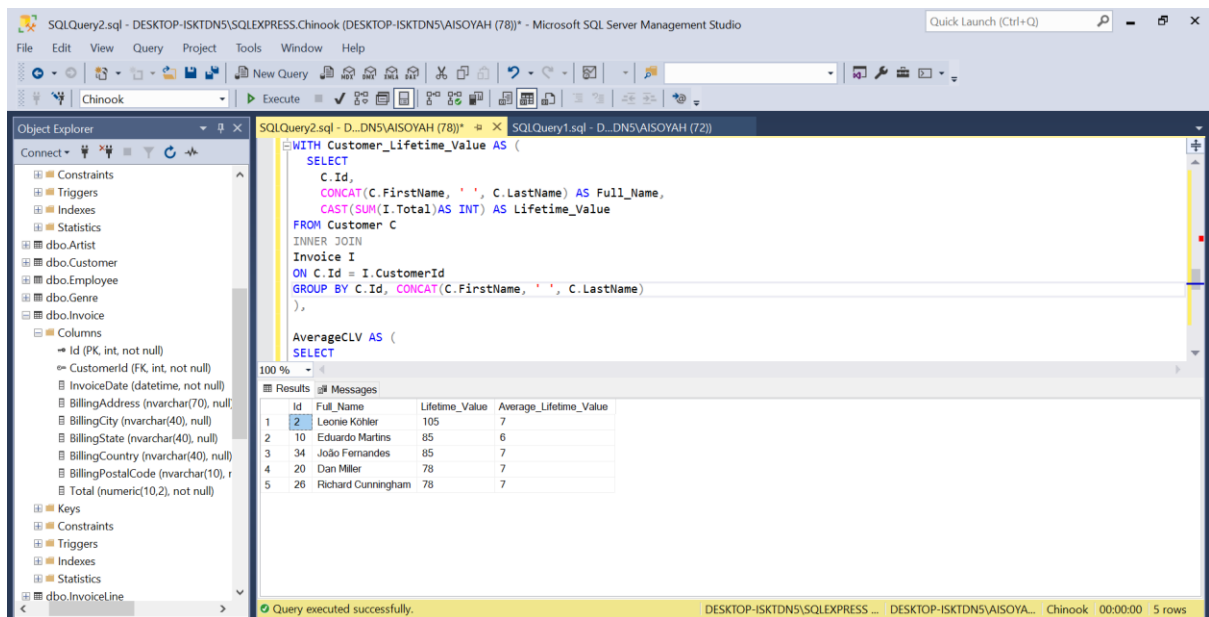
```
    CAST(SUM(I.Total)AS INT) AS Lifetime_Value
FROM Customer C
INNER JOIN
Invoice I
ON C.CustomerId = I.CustomerId
GROUP BY C.CustomerId, CONCAT(C.FirstName, ' ', C.LastName)
),
```

```
AverageCLV AS (
SELECT
    C.CustomerId,
    CAST (AVG(I.Total)AS INT) AS Average_Lifetime_Value
FROM Customer C
INNER JOIN
Invoice I ON C.CustomerId = I.CustomerId
GROUP BY
C.CustomerId
)
```

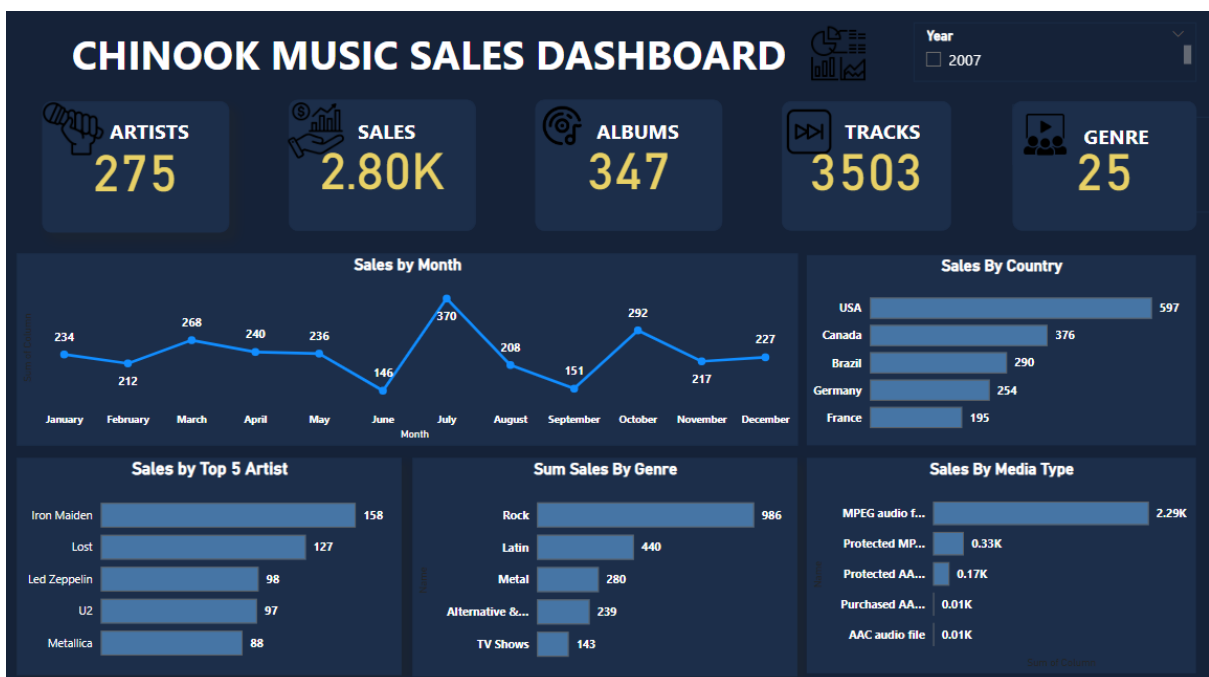
```
SELECT DISTINCT TOP 5
    C.CustomerId,
    C.Full_Name,
    C.Lifetime_Value,
    AC.Average_Lifetime_Value
FROM Customer_Lifetime_Value C
INNER JOIN
AverageCLV AC ON C.CustomerId = AC.CustomerId
```

ORDER BY

C.Lifetime_Value DESC;



DASHBOARD



Key Insights:

1. Top Customers:

- The top five customers expected to drive profits are Leonie Köhler, Eduardo Martins, João Fernandes, Dan Miller, and Richard Cunningham.

2. Top-Selling Artists:

- Iron Maiden was the highest-grossing artist, generating \$158 in revenue from album sales.
- Other significant contributors include Lost with \$127, Led Zeppelin with \$98, and U2 with \$97.

3. Customer Purchase Patterns:

- **Recency:** Eduardo Martins and Leonie Köhler are recent purchasers, showing strong customer engagement.
- **Frequency:** Leonie Köhler made the most purchases, with Eduardo Martins following closely.
- **Monetary Value:** Leonie Köhler is the top customer, contributing \$105 in sales, while 57.6% of customers fall within the mid-value category.

4. Genre Popularity:

- **Rock** led all genres with \$986 in sales, making it the most popular.

- **Latin music** came in second with \$444 in sales, while **Opera** had no sales during the period.
- **Rock and Roll** was the least popular, generating only \$8.

5. Sales Over Time:

- **2010** saw the highest yearly sales, while **2009** had the lowest.
- **July** had the peak monthly sales at \$370, followed by **October** with \$292.
- **Saturday** was the best day for sales and revenue, indicating potential for focused marketing efforts.

6. Customer Lifetime Value (CLV):

- Leonie Köhler, Eduardo Martins, and João Fernandes were identified as the top three customers in terms of profit potential, making them essential for the store's long-term success.

Recommendations:

1. Focus on High-Selling Artists:

- Increase promotions for top-selling artists like Iron Maiden and Led Zeppelin to capitalize on their popularity.

2. Enhance Customer Retention:

- Implement loyalty programs and personalized offers for high-frequency and high-value customers such as Leonie Köhler and Eduardo Martins.

3. Re-engage Inactive Customers:

- Target inactive customers (e.g., those who last purchased in 2007) with email campaigns or special offers to bring them back.

4. Boost Underperforming Genres:

- Explore promotions for genres like Rock and Roll, and consider seasonal offers to increase sales of genres such as Latin music during relevant cultural events.

5. Maximize Weekend Sales:

- Introduce new albums or run special promotions on Saturdays, as this day consistently generated the highest sales.

Conclusion:

The analysis of Chinook music store data reveals valuable insights into customer behaviour, genre popularity, and sales trends. By focusing on top-selling artists, improving customer retention, and optimizing marketing strategies based on these insights, the store can enhance its profitability and ensure sustained growth in a competitive market. Building relationships with high-value customers will be crucial for long-term success.