STRUCTURED QUERY LANGUAGE (SQL) PROJECT

BY OKOLI KOSISOCHUKWU ANASTESIA

Table of Contents

1. Introduction

1.1 Background

1.2 Objectives

1.3 Scope

2. Step 1: Database Setup & Data Import

3. Step 2: Data Cleaning

4. Step 3: Exploratory Data Analysis (EDA)

5. Step 4: Business Analysis (Project Questions i–x)

6. Step 5: Reporting & Insights

7. Appendix: Full One-Click Project Script

1.Introduction

1.1 Background

In the fiercely competitive retail sector, companies depend on precise data analysis to assess performance, predict trends, and comprehend customer behavior. A multitude of data is produced by sales transactions, including revenue, product categories, volumes, and consumer demographics. However, before it can be turned into useful insights, this raw data frequently needs to be cleaned and organized.

By using **SQL Server**, businesses can create a centralized sales database that not only stores data securely but also enables robust querying for business intelligence. This project demonstrates the complete workflow — from database creation and data cleaning to analysis and reporting.

1.2 Objective

The main objectives of this project are:

- **Database Setup** – Create a retail sales database in SQL Server and import raw transactional data.
- **Data Cleaning** – Identify and handle missing or invalid values to ensure data integrity.
- **Exploratory Data Analysis (EDA)** – Use SQL queries to uncover initial trends and distributions.
- **Business Analysis** – Answer specific business questions related to sales performance, customer behavior, and product categories.
- **Comprehensive Reporting** – Document the entire process and findings in a structured format suitable for professional portfolio use.

1.3 Scope

This project covers the end-to-end lifecycle of a retail sales analytics task:

- Designing the sales database schema.
- Importing CSV data into SQL Server using the **Import Flat File**.
- Performing data cleaning operations to remove null values and ensure consistency.
- Executing SQL queries to answer ten business-focused questions such as:
  - Which product categories generate the highest revenue?
  - Who are the top 5 customers based on spending?
  - What is the sales distribution across different shifts (morning, afternoon, evening)?
  - Which month in each year is the best-performing?
- Preparing a professional report of all steps, queries, and insights.

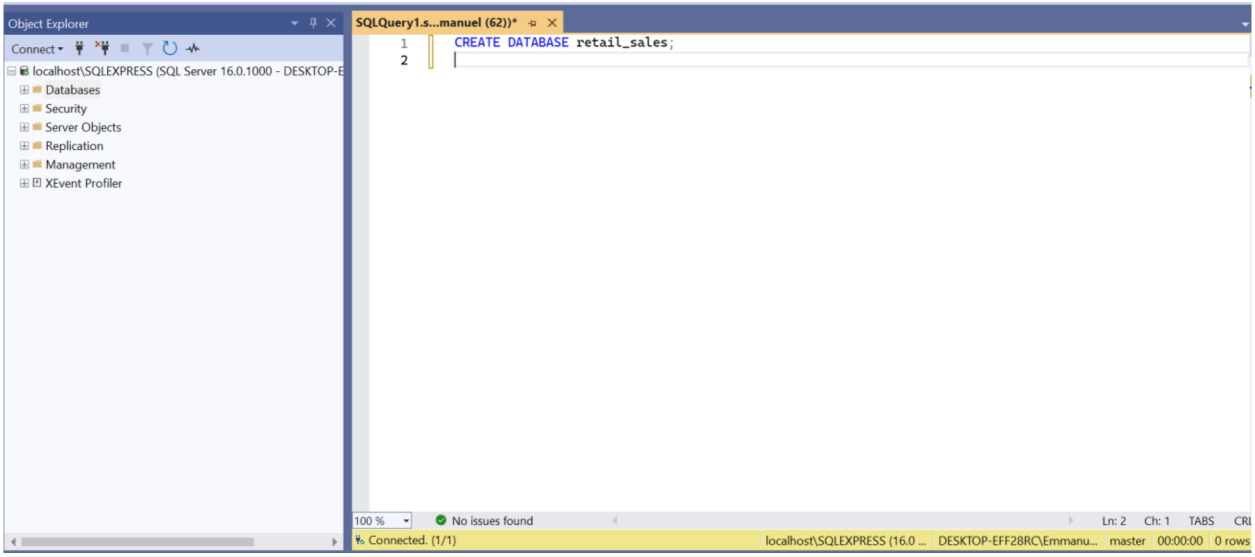
By the end of this project, the dataset evolves from **raw CSV files** to a **cleaned, structured, and analyzed sales database**, showcasing practical SQL skills applicable to real-world retail analytics. Below is the step-by-step process of data setup and import.

2 Step 1: Database Setup & Data Import

1.a Creating the Database

The first step in this project is to set up the retail sales database where all the transactional data will be stored. Using SQL Server Management Studio (SSMS), we begin by creating a dedicated database using syntax named retail_sales.

```
-- Create database

CREATE DATABASE retail_sales;
```



```
CREATE DATABASE retail_sales;
```

No issues found

Connected. (1/1)

Object Explorer
Connect
localhost\SQLEXPRESS (SQL Server 16.0.1000 - DESKTOP-E
Databases
Security
Server Objects
Replication
Management
XEvent Profiler
SQLQuery1.s...manuel (62))*
100 %
Ln: 2
Ch: 1
TABS
CRL
localhost\SQLEXPRESS (16.0 ...
DESKTOP-EFF28RC\Emmanu...
master
00:00:00
0 rows

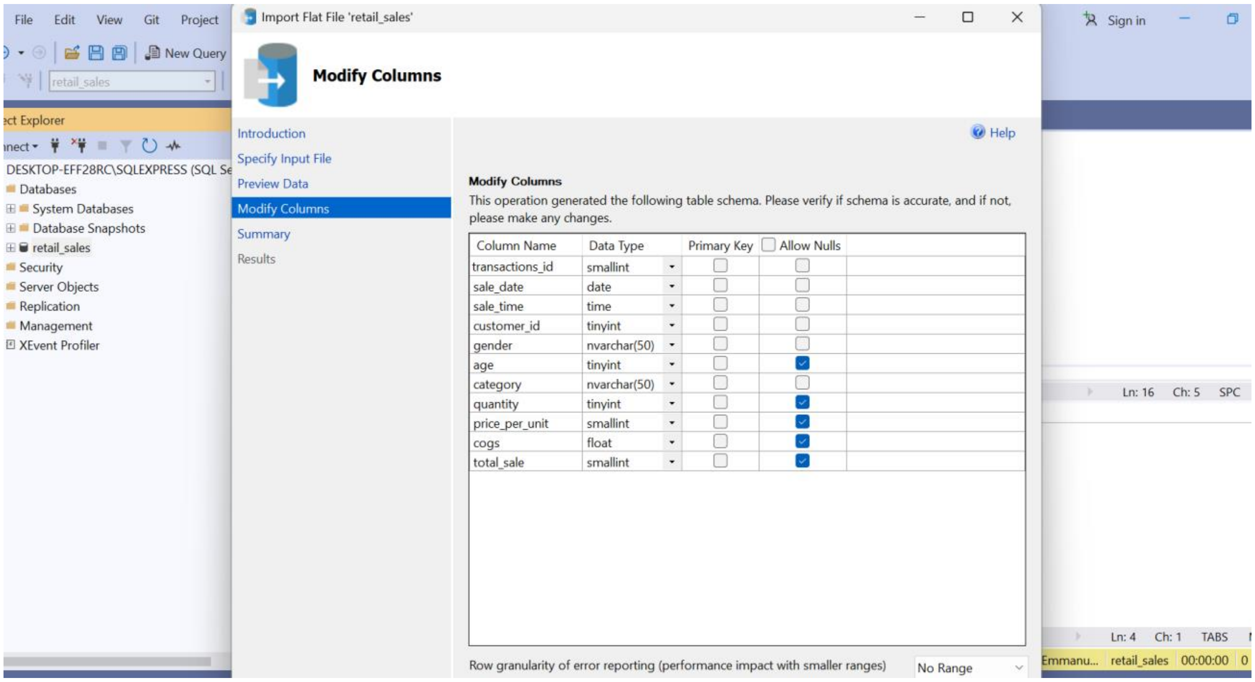
Fig i

In Fig i, this ensures that all subsequent operations — table creation, data import, and analysis are organized within a single structured database environment.

1.b Importing the Data

The dataset provided is in CSV format. In SQL Server Management Studio, the Import Flat File is used to bring the data. The process is as follows:

- Right-click the database (retail_sales) in Object Explorer.
- Select Tasks > Import Flat File.
- Browse and select the CSV file (e.g., *SQL - retail_sales TB.csv*).
- Preview the data and ensure columns are mapped correctly to the schema.
- Complete the wizard to import the data.



Modify Columns

Modify Columns

This operation generated the following table schema. Please verify if schema is accurate, and if not, please make any changes.

Column Name	Data Type	Primary Key	□ Allow Nulls
transactions_id	smallint	□	□
sale_date	date	□	□
sale_time	time	□	□
customer_id	tinyint	□	□
gender	nvarchar(50)	□	□
age	tinyint	□	☑
category	nvarchar(50)	□	□
quantity	tinyint	□	☑
price_per_unit	smallint	□	☑
cogs	float	□	☑
total_sale	smallint	□	☑

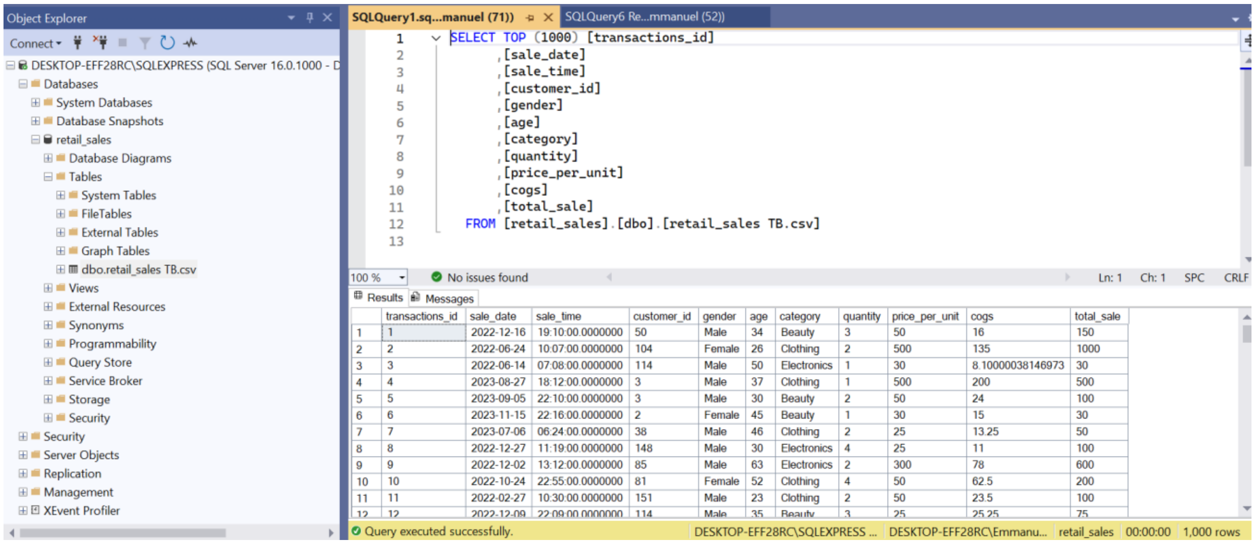
Row granularity of error reporting (performance impact with smaller ranges) No Range

Fig ii

In Fig ii, transaction_id was checked as the primary key, no allow null. While allow null was checked for other columns.

1.c Verifying Data Import

Once the data is imported, it is essential to verify that the records have been loaded successfully. This is done by querying the first few rows of the table.



```sql
SELECT TOP (1000) [transactions_id]
      ,[sale_date]
      ,[sale_time]
      ,[customer_id]
      ,[gender]
      ,[age]
      ,[category]
      ,[quantity]
      ,[price_per_unit]
      ,[cogs]
      ,[total_sale]
  FROM [retail_sales].[dbo].[retail_sales TB.csv]
```

No issues found

	transactions_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale
1	1	2022-12-16	19:10:00.0000000	50	Male	34	Beauty	3	50	16	150
2	2	2022-06-24	10:07:00.0000000	104	Female	26	Clothing	2	500	135	1000
3	3	2022-06-14	07:08:00.0000000	114	Male	50	Electronics	1	30	8.10000038146973	30
4	4	2023-08-27	18:12:00.0000000	3	Male	37	Clothing	1	500	200	500
5	5	2023-09-05	22:10:00.0000000	3	Male	30	Beauty	2	50	24	100
6	6	2023-11-15	22:16:00.0000000	2	Female	45	Beauty	1	30	15	30
7	7	2023-07-06	06:24:00.0000000	38	Male	46	Clothing	2	25	13.25	50
8	8	2022-12-27	11:19:00.0000000	148	Male	30	Electronics	4	25	11	100
9	9	2022-12-02	13:12:00.0000000	85	Male	63	Electronics	2	300	78	600
10	10	2022-10-24	22:55:00.0000000	81	Female	52	Clothing	4	50	62.5	200
11	11	2022-02-27	10:30:00.0000000	151	Male	23	Clothing	2	50	23.5	100
12	12	2022-12-09	22:09:00.0000000	114	Male	35	Beauty	3	25	25.25	75

Query executed successfully.

Fig iii

At the retail_sales TB.csv table, right click, Select Top 1000 rows.

Fig iii shows a representation showing the dataset. At this point, the database is fully set up and populated with raw sales data. The next step will involve Data Cleaning (Step 2), where we will identify and remove null or invalid values to prepare the dataset for analysis.

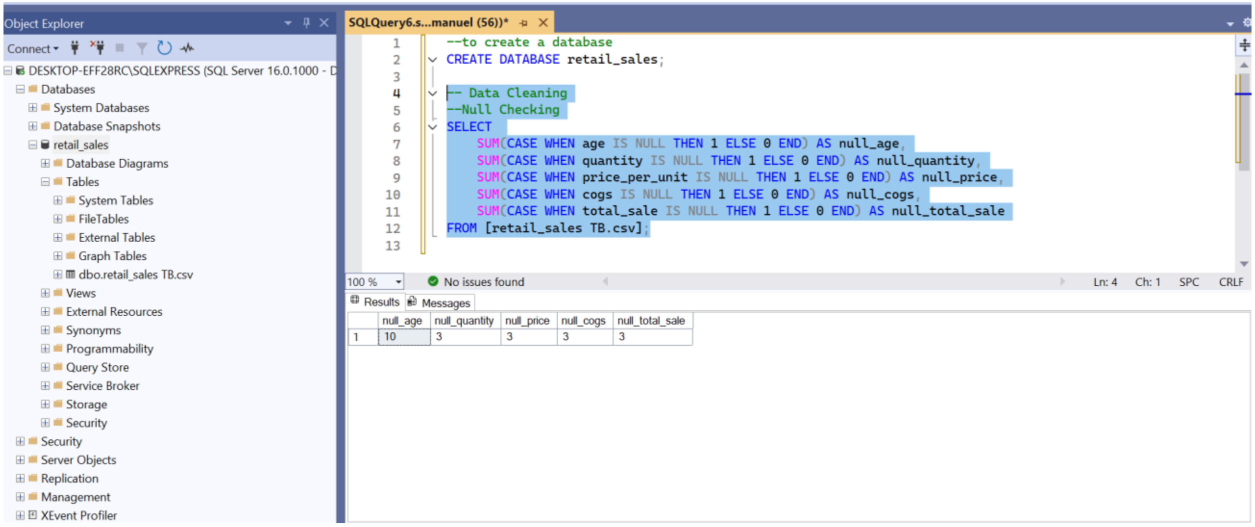
3 Step 2: Data Cleaning

Once the database was successfully created and populated with the raw retail sales data, the next step involved cleaning the dataset to ensure accuracy and reliability of insights.

2.a Identifying Null Values

To check for missing or null values across critical columns such as age, quantity, price_per_unit, cogs, and total_sale, the following SQL query was executed:

```
SELECT
    SUM (CASE WHEN age IS NULL THEN 1 ELSE 0 END) AS null_age,
    SUM (CASE WHEN quantity IS NULL THEN 1 ELSE 0 END) AS null_quantity,
    SUM (CASE WHEN price_per_unit IS NULL THEN 1 ELSE 0 END) AS null_price,
    SUM (CASE WHEN cogs IS NULL THEN 1 ELSE 0 END) AS null_cogs,
    SUM (CASE WHEN total_sale IS NULL THEN 1 ELSE 0 END) AS null_total_sale
FROM retail_sales TB.csv;
```



```
--to create a database
CREATE DATABASE retail_sales;

-- Data Cleaning
--Null Checking
SELECT
    SUM(CASE WHEN age IS NULL THEN 1 ELSE 0 END) AS null_age,
    SUM(CASE WHEN quantity IS NULL THEN 1 ELSE 0 END) AS null_quantity,
    SUM(CASE WHEN price_per_unit IS NULL THEN 1 ELSE 0 END) AS null_price,
    SUM(CASE WHEN cogs IS NULL THEN 1 ELSE 0 END) AS null_cogs,
    SUM(CASE WHEN total_sale IS NULL THEN 1 ELSE 0 END) AS null_total_sale
FROM [retail_sales TB.csv];
```

	null_age	null_quantity	null_price	null_cogs	null_total_sale
1	10	3	3	3	3

Fig iv

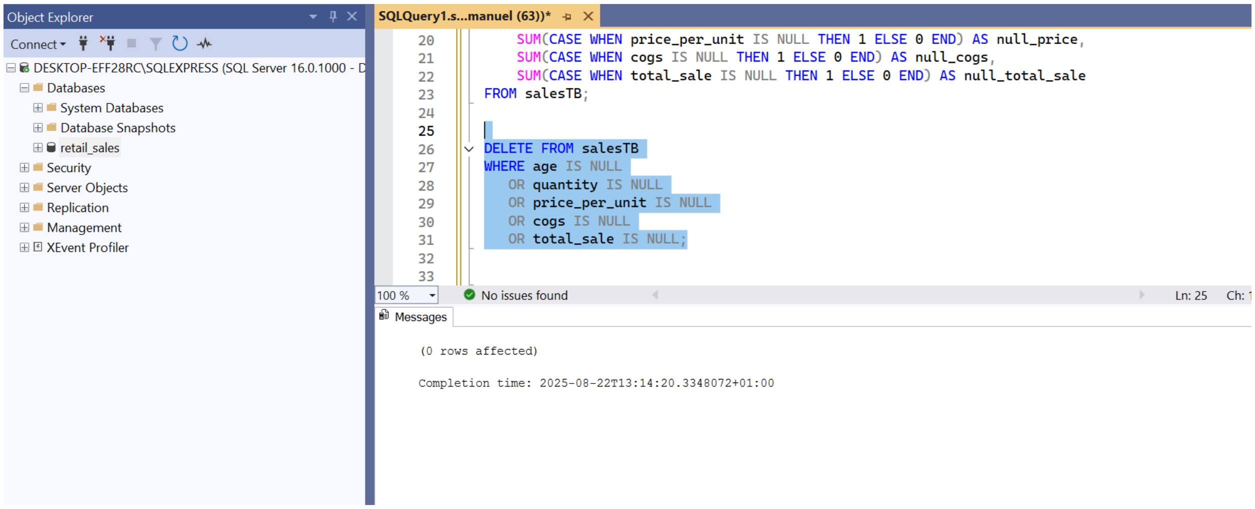
In Fig iv, this query provided a quick summary of the number of null values in each column.

2.b Removing Records with Missing Values

Since null values can distort sales totals and averages, rows containing them were removed using:

```
DELETE FROM [retail_sales TB.csv];
```

```
WHERE age IS NULL
  OR quantity IS NULL
  OR price_per_unit IS NULL
  OR cogs IS NULL
  OR total_sale IS NULL;
```



```
    SUM(CASE WHEN price_per_unit IS NULL THEN 1 ELSE 0 END) AS null_price,
    SUM(CASE WHEN cogs IS NULL THEN 1 ELSE 0 END) AS null_cogs,
    SUM(CASE WHEN total_sale IS NULL THEN 1 ELSE 0 END) AS null_total_sale
FROM salesTB;


DELETE FROM salesTB
WHERE age IS NULL
   OR quantity IS NULL
   OR price_per_unit IS NULL
   OR cogs IS NULL
   OR total_sale IS NULL;
```

```
(0 rows affected)

Completion time: 2025-08-22T13:14:20.3348072+01:00
```

Fig v

In Fig v, the result returned **0**, confirming that the dataset was now clean and ready for analysis.

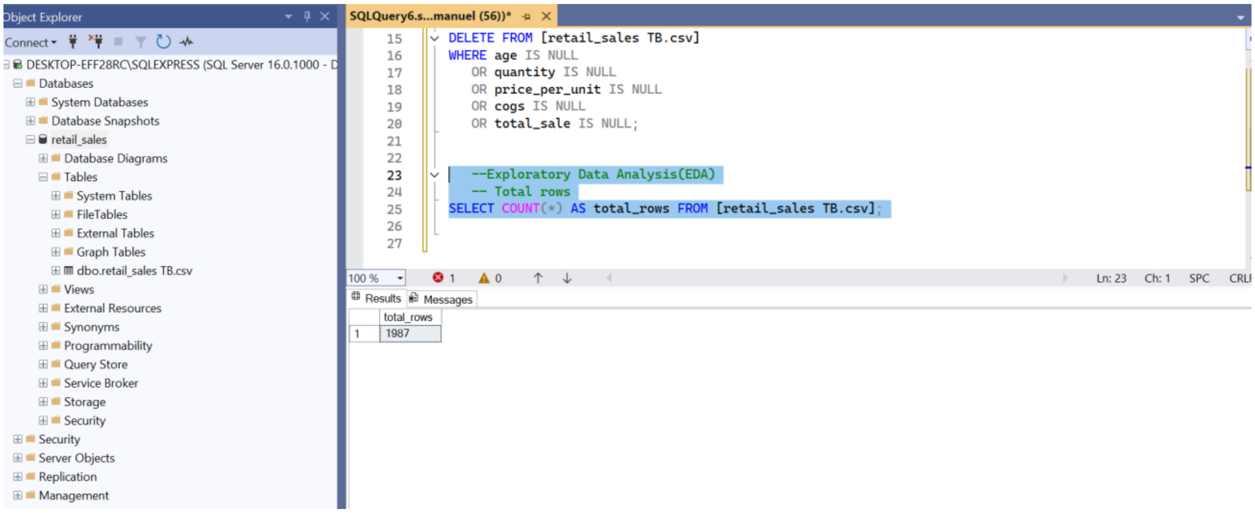
4 Step 3: Exploratory Data Analysis (EDA)

With the cleaned dataset in place, the next step was to perform **exploratory data analysis (EDA)**. The goal of this phase was to better understand the structure of the sales data, identify patterns, and prepare for deeper business analysis.

3.a Checking Total Records

The first step was to confirm the number of transactions available after cleaning:

```
SELECT COUNT (*) AS total_transactions

FROM [retail_sales TB.csv];
```



```
DELETE FROM [retail_sales TB.csv]
WHERE age IS NULL
   OR quantity IS NULL
   OR price_per_unit IS NULL
   OR cogs IS NULL
   OR total_sale IS NULL;


   --Exploratory Data Analysis(EDA)
   -- Total rows
SELECT COUNT(*) AS total_rows FROM [retail_sales TB.csv];
```

	total_rows
1	1987

Fig vi

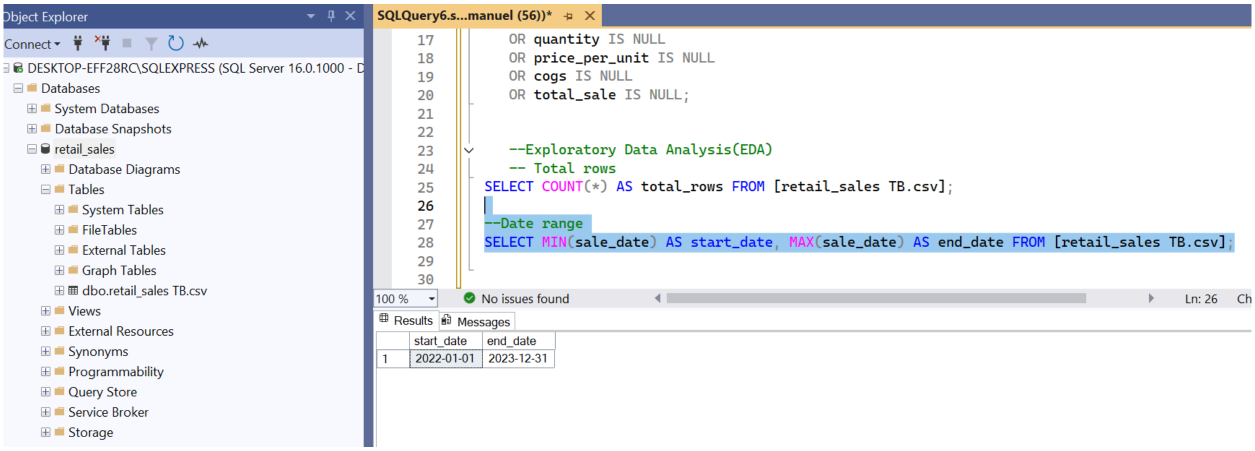
In Fig vi, this query returned the total count of rows in the dataset, representing all valid transactions.

3.b Date Range of Sales

To understand the time period covered by the dataset, the following query was run:

```
SELECT

   MIN (sale_date) AS start_date,

   MAX (sale_date) AS end_date
```

FROM [retail_sales TB.csv];



```
    OR quantity IS NULL
    OR price_per_unit IS NULL
    OR cogs IS NULL
    OR total_sale IS NULL;


    --Exploratory Data Analysis(EDA)
    -- Total rows
SELECT COUNT(*) AS total_rows FROM [retail_sales TB.csv];

--Date range
SELECT MIN(sale_date) AS start_date, MAX(sale_date) AS end_date FROM [retail_sales TB.csv];
```

	start_date	end_date
1	2022-01-01	2023-12-31

Fig vii

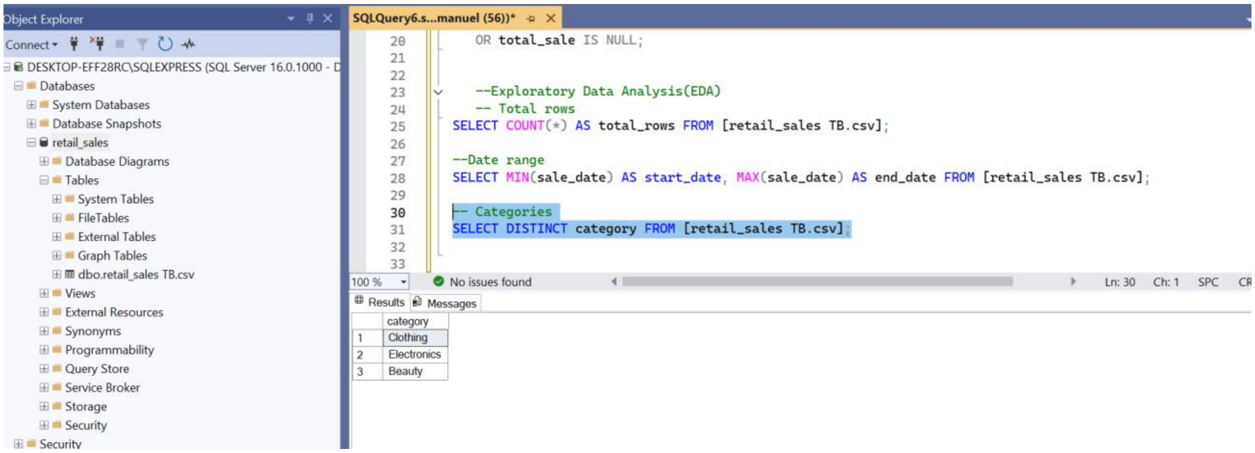
In Fig vii, this revealed the earliest and latest sales dates captured in the database.

3.c Distinct Product Categories

To examine the variety of products sold, distinct categories were identified:

SELECT DISTINCT category

FROM [retail_sales TB.csv];



```
    OR total_sale IS NULL;


    --Exploratory Data Analysis(EDA)
    -- Total rows
SELECT COUNT(*) AS total_rows FROM [retail_sales TB.csv];

--Date range
SELECT MIN(sale_date) AS start_date, MAX(sale_date) AS end_date FROM [retail_sales TB.csv];

-- Categories
SELECT DISTINCT category FROM [retail_sales TB.csv];
```

	category
1	Clothing
2	Electronics
3	Beauty

Fig viii

In Fig viii, this provided a list of all product categories available in the dataset (e.g., Clothing, Beauty, Electronics, etc.).

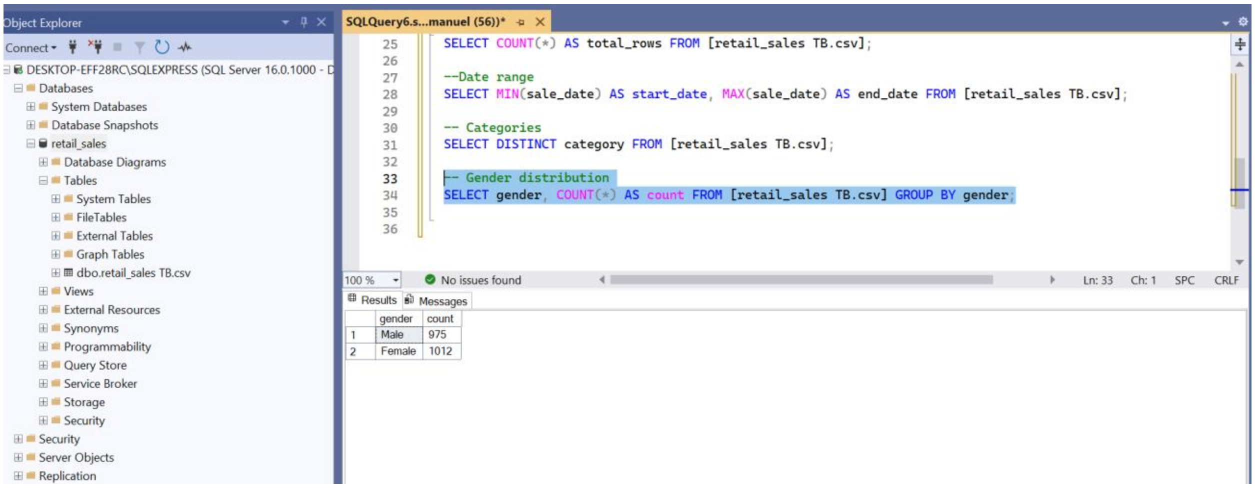
3.d Gender Distribution of Customers

Customer demographics were checked by analyzing gender distribution:

SELECT gender, COUNT (*) AS transactions

FROM [retail_sales TB.csv];

GROUP BY gender;



```
SELECT COUNT(*) AS total_rows FROM [retail_sales TB.csv];

--Date range
SELECT MIN(sale_date) AS start_date, MAX(sale_date) AS end_date FROM [retail_sales TB.csv];

-- Categories
SELECT DISTINCT category FROM [retail_sales TB.csv];

-- Gender distribution
SELECT gender, COUNT(*) AS count FROM [retail_sales TB.csv] GROUP BY gender;
```

	gender	count
1	Male	975
2	Female	1012

Fig ix

In Fig ix, this showed how many transactions were made by male vs. female customers.

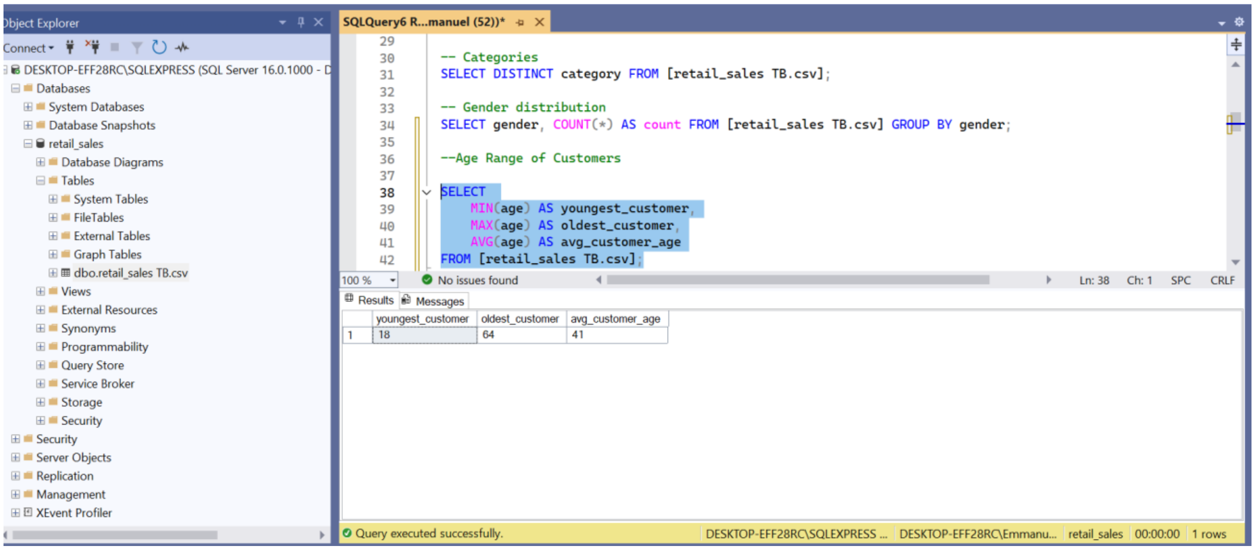
3.e Age Range of Customers

The age profile of customers was explored using:

SELECT

MIN (age) AS youngest_customer,

```
    MAX (age) AS oldest_customer,
    AVG (age) AS avg_customer_age
FROM [retail_sales TB.csv];
```



```
-- Categories
SELECT DISTINCT category FROM [retail_sales TB.csv];

-- Gender distribution
SELECT gender, COUNT(*) AS count FROM [retail_sales TB.csv] GROUP BY gender;

--Age Range of Customers

SELECT
    MIN(age) AS youngest_customer,
    MAX(age) AS oldest_customer,
    AVG(age) AS avg_customer_age
FROM [retail_sales TB.csv];
```

No issues found

	youngest_customer	oldest_customer	avg_customer_age
1	18	64	41

Query executed successfully. DESKTOP-EFF28RC\SQLEXPRESS ... DESKTOP-EFF28RC\Emmanu... retail_sales 00:00:00 1 rows

Fig x

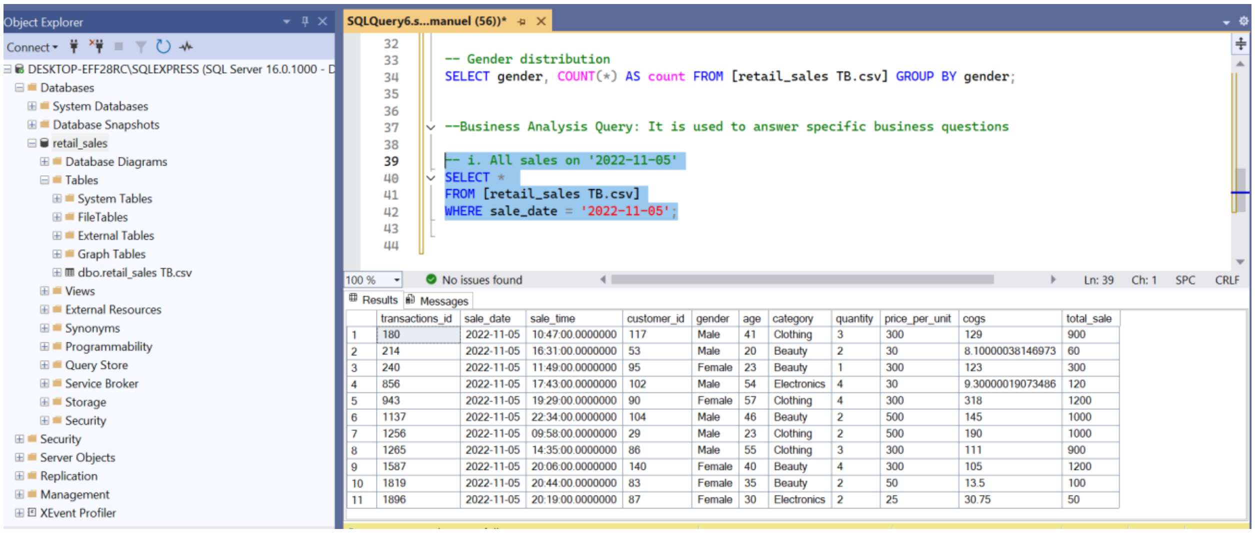
In Fig x, this gave insights into the customer base in terms of age.

5 Step 4: Business Analysis (Project Questions)

In this phase, SQL was used to answer specific business questions based on the cleaned retail sales dataset.

4.a Retrieve all columns for sales made on *2022-11-05*

```
SELECT *
FROM [retail_sales TB.csv]
WHERE sale_date = '2022-11-05';
```



```
-- Gender distribution
SELECT gender, COUNT(*) AS count FROM [retail_sales TB.csv] GROUP BY gender;


--Business Analysis Query: It is used to answer specific business questions

-- i. All sales on '2022-11-05'
SELECT *
FROM [retail_sales TB.csv]
WHERE sale_date = '2022-11-05';
```

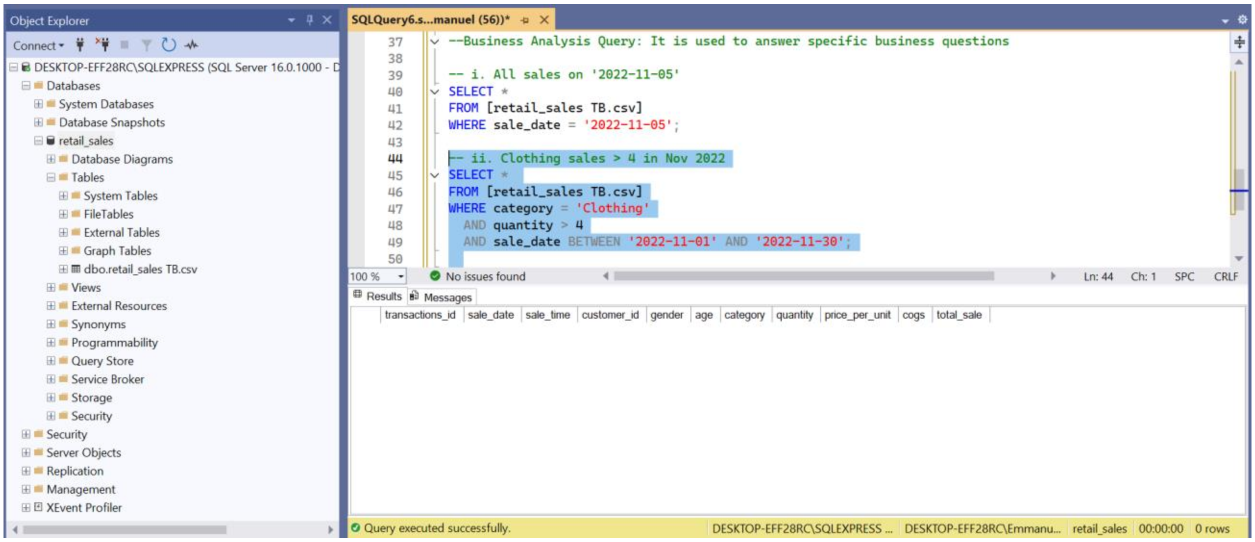
	transactions_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale
1	180	2022-11-05	10:47:00.0000000	117	Male	41	Clothing	3	300	129	900
2	214	2022-11-05	16:31:00.0000000	53	Male	20	Beauty	2	30	8.10000038146973	60
3	240	2022-11-05	11:49:00.0000000	95	Female	23	Beauty	1	300	123	300
4	856	2022-11-05	17:43:00.0000000	102	Male	54	Electronics	4	30	9.30000019073486	120
5	943	2022-11-05	19:29:00.0000000	90	Female	57	Clothing	4	300	318	1200
6	1137	2022-11-05	22:34:00.0000000	104	Male	46	Beauty	2	500	145	1000
7	1256	2022-11-05	09:58:00.0000000	29	Male	23	Clothing	2	500	190	1000
8	1265	2022-11-05	14:35:00.0000000	86	Male	55	Clothing	3	300	111	900
9	1587	2022-11-05	20:06:00.0000000	140	Female	40	Beauty	4	300	105	1200
10	1819	2022-11-05	20:44:00.0000000	83	Female	35	Beauty	2	50	13.5	100
11	1896	2022-11-05	20:19:00.0000000	87	Female	30	Electronics	2	25	30.75	50

Fig xi

In Fig xi, this returns all transactions recorded on November 5, 2022.

4.b Retrieve transactions where the category is *Clothing* and quantity sold is more than 4 in November 2022

```
SELECT *

FROM [retail_sales TB.csv]

WHERE category = 'Clothing'

  AND quantity > 4

  AND sale_date BETWEEN '2022-11-01' AND '2022-11-30';
```



```
--Business Analysis Query: It is used to answer specific business questions

-- i. All sales on '2022-11-05'
SELECT *
FROM [retail_sales TB.csv]
WHERE sale_date = '2022-11-05';

-- ii. Clothing sales > 4 in Nov 2022
SELECT *
FROM [retail_sales TB.csv]
WHERE category = 'Clothing'
  AND quantity > 4
  AND sale_date BETWEEN '2022-11-01' AND '2022-11-30';
```

No issues found

	transactions_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale

Query executed successfully. DESKTOP-EFF28RC\SQLEXPRESS ... DESKTOP-EFF28RC\Emmanu... retail_sales 00:00:00 0 rows

Fig xii

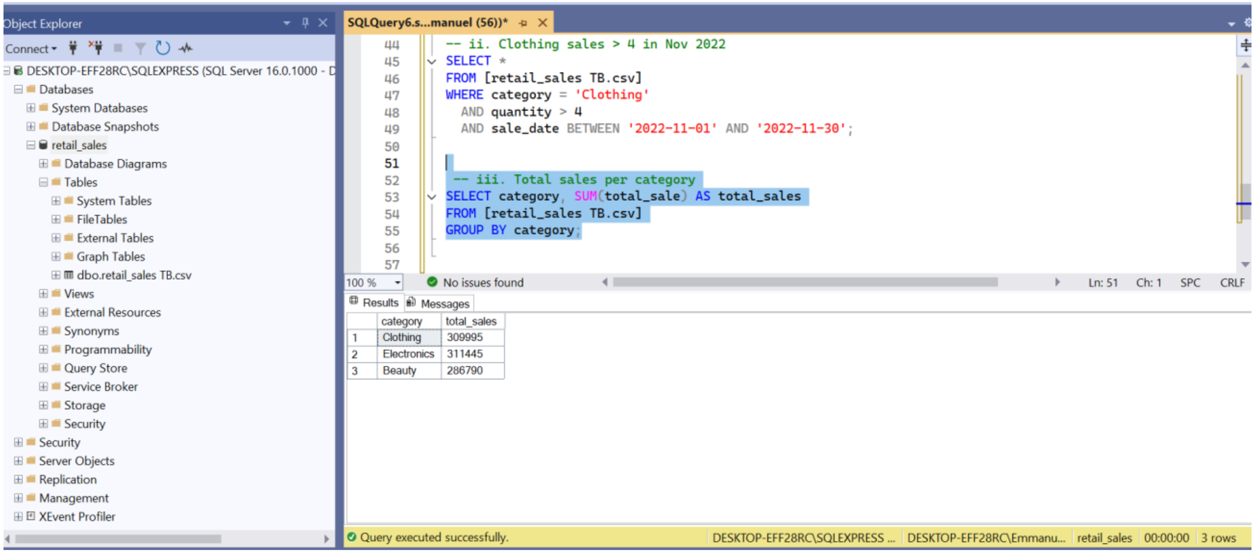
In Fig xii, it filters transactions for the Clothing category with bulk orders in Nov-2022.

4.c Calculate the total sales for each category

SELECT category, SUM (total_sale) AS total_sales

FROM [retail_sales TB.csv]

GROUP BY category;



```sql
-- ii. Clothing sales > 4 in Nov 2022
SELECT *
FROM [retail_sales TB.csv]
WHERE category = 'Clothing'
  AND quantity > 4
  AND sale_date BETWEEN '2022-11-01' AND '2022-11-30';


 -- iii. Total sales per category
SELECT category, SUM(total_sale) AS total_sales
FROM [retail_sales TB.csv]
GROUP BY category;
```

No issues found

	category	total_sales
1	Clothing	309995
2	Electronics	311445
3	Beauty	286790

Query executed successfully. DESKTOP-EFF28RC\SQLEXPRESS ... DESKTOP-EFF28RC\Emmanu... retail_sales 00:00:00 3 rows

Fig xiii

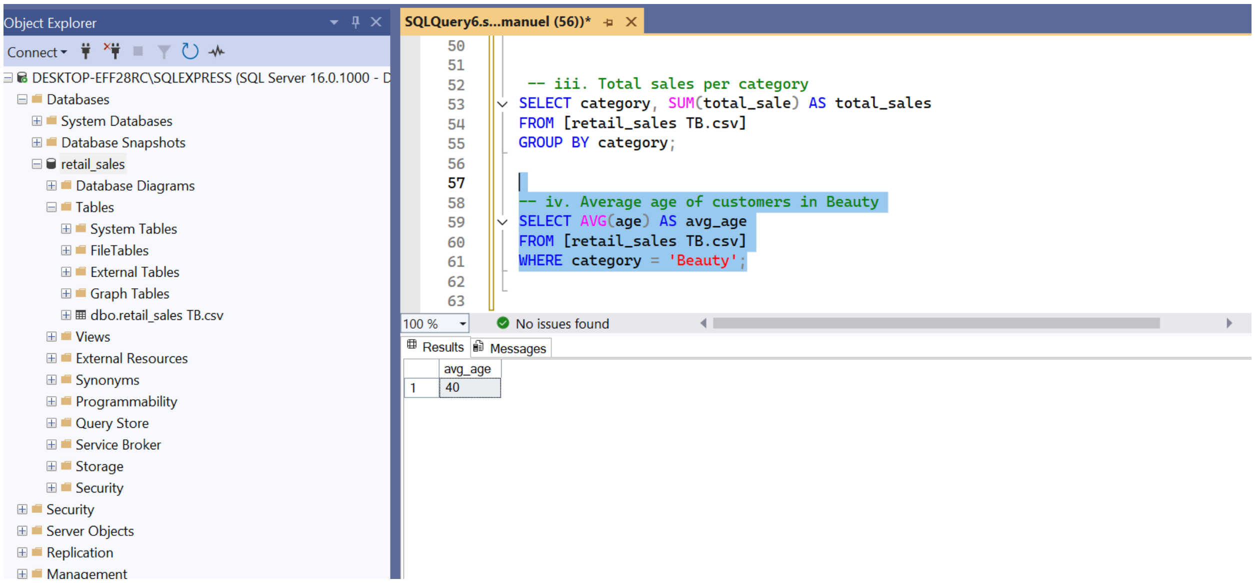
In Fig xiii, it provides sales contribution of each category.

4.d Find the average age of customers who purchased items from the *Beauty* category

SELECT AVG (age) AS avg_age_beauty_customers

FROM [retail_sales TB.csv]

WHERE category = 'Beauty';



```
 -- iii. Total sales per category
SELECT category, SUM(total_sale) AS total_sales
FROM [retail_sales TB.csv]
GROUP BY category;


-- iv. Average age of customers in Beauty
SELECT AVG(age) AS avg_age
FROM [retail_sales TB.csv]
WHERE category = 'Beauty';
```

No issues found

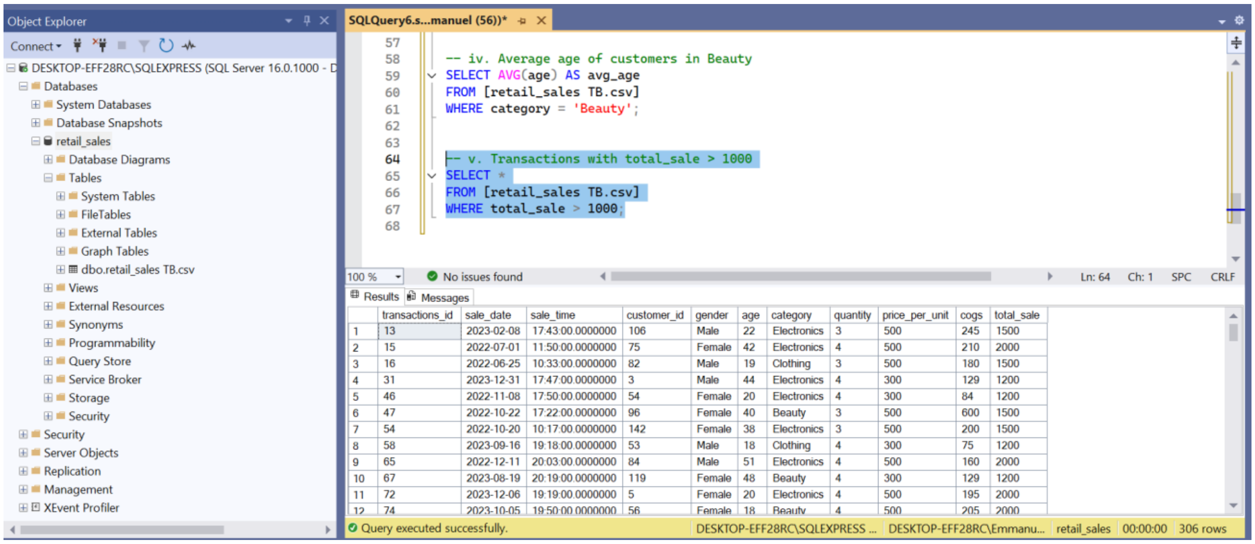
	avg_age
1	40

Fig xiv

In Fig xiv, it shows demographic insights for Beauty category buyers.

4.e Find all transactions where total_sale is greater than 1000

```
SELECT *

FROM [retail_sales TB.csv]

WHERE total_sale > 1000;
```



```sql

-- iv. Average age of customers in Beauty
SELECT AVG(age) AS avg_age
FROM [retail_sales TB.csv]
WHERE category = 'Beauty';


-- v. Transactions with total_sale > 1000
SELECT *
FROM [retail_sales TB.csv]
WHERE total_sale > 1000;

```

	transactions_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale
1	13	2023-02-08	17:43:00.0000000	106	Male	22	Electronics	3	500	245	1500
2	15	2022-07-01	11:50:00.0000000	75	Female	42	Electronics	4	500	210	2000
3	16	2022-06-25	10:33:00.0000000	82	Male	19	Clothing	3	500	180	1500
4	31	2023-12-31	17:47:00.0000000	3	Male	44	Electronics	4	300	129	1200
5	46	2022-11-08	17:50:00.0000000	54	Female	20	Electronics	4	300	84	1200
6	47	2022-10-22	17:22:00.0000000	96	Female	40	Beauty	3	500	600	1500
7	54	2022-10-20	10:17:00.0000000	142	Female	38	Electronics	3	500	200	1500
8	58	2023-09-16	19:18:00.0000000	53	Male	18	Clothing	4	300	75	1200
9	65	2022-12-11	20:03:00.0000000	84	Male	51	Electronics	4	500	160	2000
10	67	2023-08-19	20:19:00.0000000	119	Female	48	Beauty	4	300	129	1200
11	72	2023-12-06	19:19:00.0000000	5	Female	20	Electronics	4	500	195	2000
12	74	2023-10-05	19:50:00.0000000	56	Female	18	Beauty	4	500	205	2000

Query executed successfully. | DESKTOP-EFF28RC\SQLEXPRESS ... | DESKTOP-EFF28RC\Emmanu... | retail_sales | 00:00:00 | 306 rows

Fig xv

In Fig xv, it identifies high-value transactions.

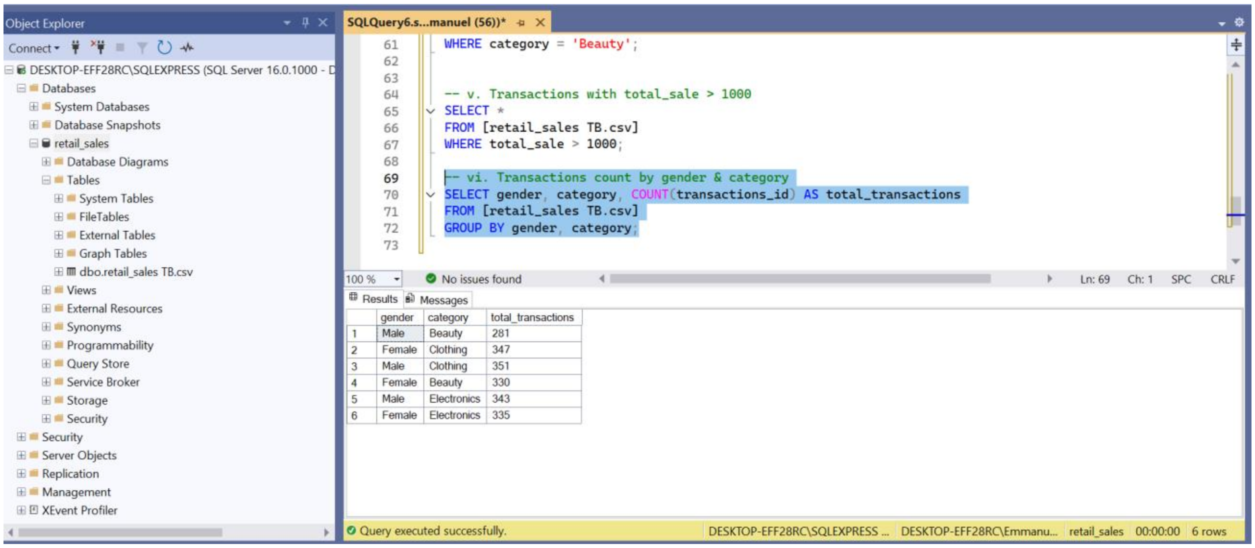
4.f Find the total number of transactions made by each gender in each category

SELECT gender, category, COUNT (transaction_id) AS total_transactions

FROM [retail_sales TB.csv]

GROUP BY gender, category

ORDER BY category, gender;



```sql
WHERE category = 'Beauty';


-- v. Transactions with total_sale > 1000
SELECT *
FROM [retail_sales TB.csv]
WHERE total_sale > 1000;

-- vi. Transactions count by gender & category
SELECT gender, category, COUNT(transactions_id) AS total_transactions
FROM [retail_sales TB.csv]
GROUP BY gender, category;
```

No issues found

	gender	category	total_transactions
1	Male	Beauty	281
2	Female	Clothing	347
3	Male	Clothing	351
4	Female	Beauty	330
5	Male	Electronics	343
6	Female	Electronics	335

Query executed successfully. DESKTOP-EFF28RC\SQLEXPRESS ... DESKTOP-EFF28RC\Emmanu... retail_sales 00:00:00 6 rows

Fig xvi

In Fig xvi, it breaks down transactions by gender within each category.

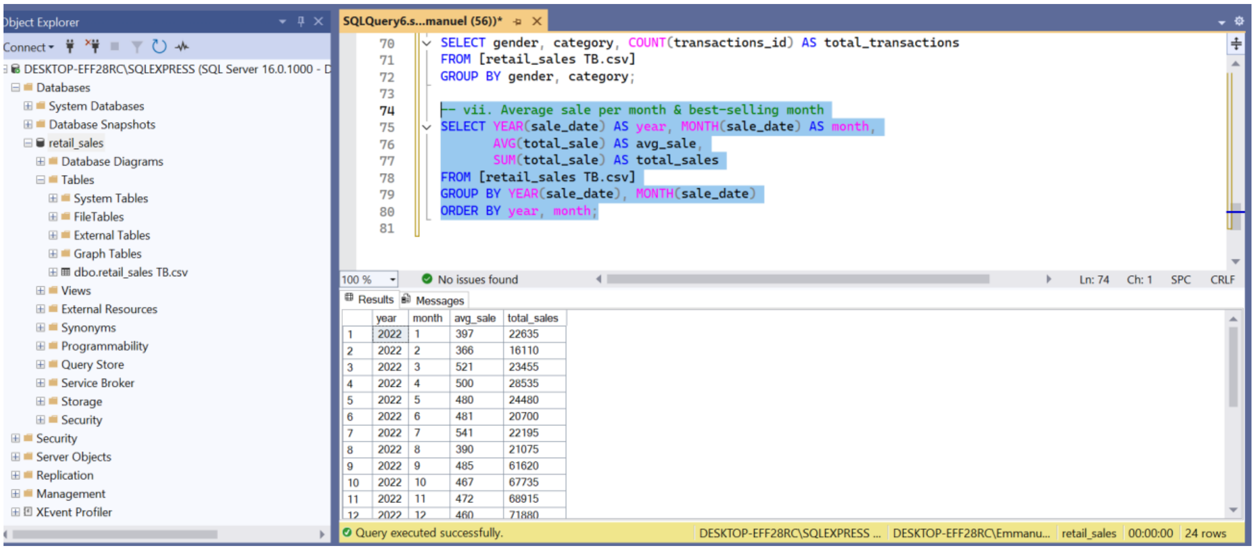
4.g Calculate the average sale for each month and find the best-selling month in each year

```
-- Monthly average sales

SELECT

    YEAR (sale_date) AS year,

    MONTH (sale_date) AS month,

    AVG (total_sale) AS avg_monthly_sale

FROM [retail_sales TB.csv]

GROUP BY YEAR (sale_date), MONTH (sale_date)

ORDER BY year, month;
```

4.h Best-selling month per year

```
SELECT year, month, avg_monthly_sale
```

```
FROM (
    SELECT
        YEAR (sale_date) AS year,
        MONTH (sale_date) AS month,
        AVG (total_sale) AS avg_monthly_sale,
        RANK () OVER (PARTITION BY YEAR (sale_date) ORDER BY AVG (total_sale)
DESC) AS rnk
    FROM [retail_sales TB.csv]
    GROUP BY YEAR (sale_date), MONTH (sale_date)
) ranked
WHERE rnk = 1;
```



```
SELECT gender, category, COUNT(transactions_id) AS total_transactions
FROM [retail_sales TB.csv]
GROUP BY gender, category;

-- vii. Average sale per month & best-selling month
SELECT YEAR(sale_date) AS year, MONTH(sale_date) AS month,
       AVG(total_sale) AS avg_sale,
       SUM(total_sale) AS total_sales
FROM [retail_sales TB.csv]
GROUP BY YEAR(sale_date), MONTH(sale_date)
ORDER BY year, month;
```

	year	month	avg_sale	total_sales
1	2022	1	397	22635
2	2022	2	366	16110
3	2022	3	521	23455
4	2022	4	500	28535
5	2022	5	480	24480
6	2022	6	481	20700
7	2022	7	541	22195
8	2022	8	390	21075
9	2022	9	485	61620
10	2022	10	467	67735
11	2022	11	472	68915
12	2022	12	460	71880

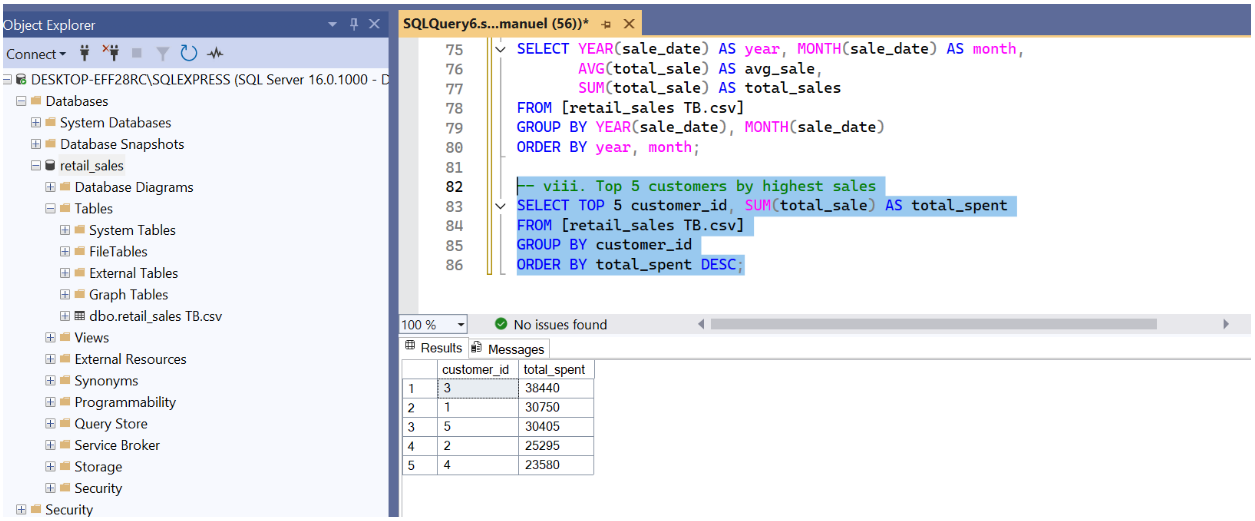
```
Query executed successfully.    DESKTOP-EFF28RC\SQLEXPRESS ...    DESKTOP-EFF28RC\Emmanu...    retail_sales    00:00:00    24 rows
```

Fig xvii

In Fig xvii, first query shows monthly trends, second highlights the best month each year.

4.i Find the top 5 customers based on highest total sales

```
SELECT TOP 5 customer_id, SUM (total_sale) AS total_sales

FROM [retail_sales TB.csv]

GROUP BY customer_id

ORDER BY total_sales DESC;
```



```
SELECT YEAR(sale_date) AS year, MONTH(sale_date) AS month,
       AVG(total_sale) AS avg_sale,
       SUM(total_sale) AS total_sales
FROM [retail_sales TB.csv]
GROUP BY YEAR(sale_date), MONTH(sale_date)
ORDER BY year, month;

-- viii. Top 5 customers by highest sales
SELECT TOP 5 customer_id, SUM(total_sale) AS total_spent
FROM [retail_sales TB.csv]
GROUP BY customer_id
ORDER BY total_spent DESC;
```

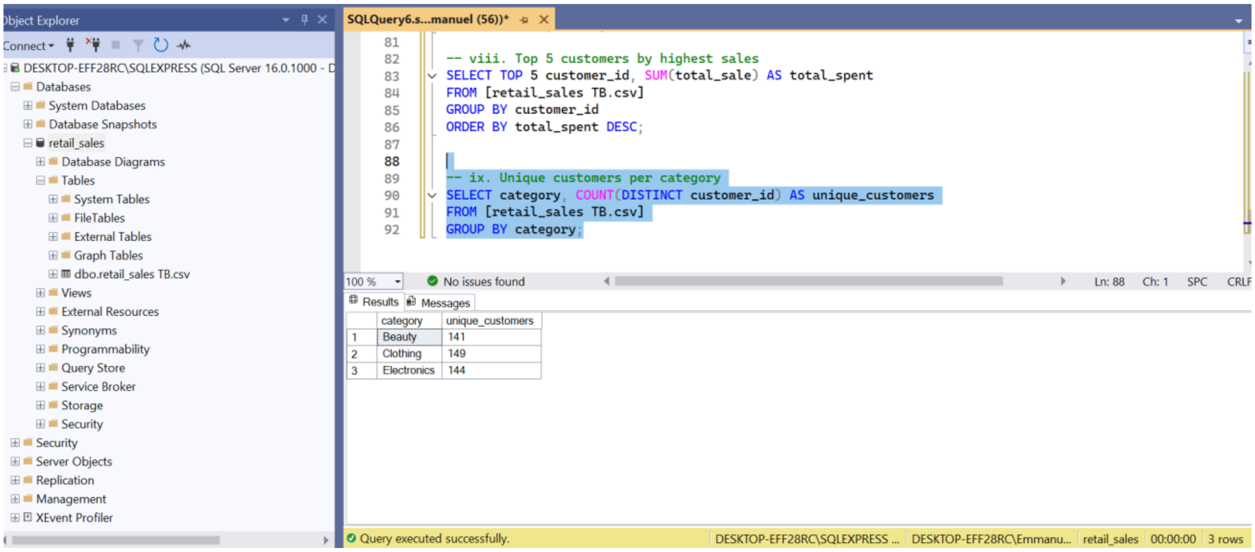
```
   customer_id  total_spent
1  3            38440
2  1            30750
3  5            30405
4  2            25295
5  4            23580
```

Fig xviii

In Fig xviii, it Identifies the most valuable customers.

4.j Find the number of unique customers who purchased items from each category

```
SELECT category, COUNT (DISTINCT customer_id) AS unique_customers

FROM [retail_sales TB.csv]

GROUP BY category;
```



```sql

-- viii. Top 5 customers by highest sales
SELECT TOP 5 customer_id, SUM(total_sale) AS total_spent
FROM [retail_sales TB.csv]
GROUP BY customer_id
ORDER BY total_spent DESC;


-- ix. Unique customers per category
SELECT category, COUNT(DISTINCT customer_id) AS unique_customers
FROM [retail_sales TB.csv]
GROUP BY category;
```

	category	unique_customers
1	Beauty	141
2	Clothing	149
3	Electronics	144

Query executed successfully.

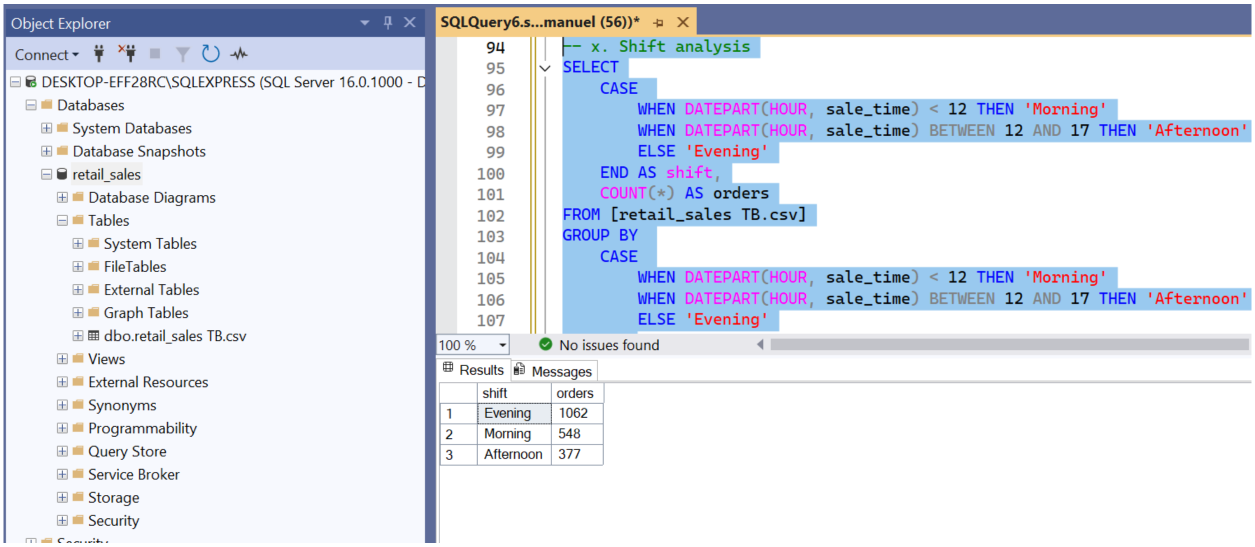
Fig xix

In Fig xix, it measures customer reach per product category.

4.k Create shifts and count number of orders per shift

```sql
SELECT
    CASE
        WHEN sale_time < '12:00:00' THEN 'Morning'
        WHEN sale_time BETWEEN '12:00:00' AND '17:00:00' THEN 'Afternoon'
        ELSE 'Evening'
    END AS shift,
    COUNT (transaction_id) AS total_orders
FROM [retail_sales TB.csv]
GROUP BY
    CASE
```

```
        WHEN sale_time < '12:00:00' THEN 'Morning'

        WHEN sale_time BETWEEN '12:00:00' AND '17:00:00' THEN 'Afternoon'

        ELSE 'Evening'

    END

ORDER BY total_orders DESC;
```



```
-- x. Shift analysis
SELECT
    CASE
        WHEN DATEPART(HOUR, sale_time) < 12 THEN 'Morning'
        WHEN DATEPART(HOUR, sale_time) BETWEEN 12 AND 17 THEN 'Afternoon'
        ELSE 'Evening'
    END AS shift,
    COUNT(*) AS orders
FROM [retail_sales TB.csv]
GROUP BY
    CASE
        WHEN DATEPART(HOUR, sale_time) < 12 THEN 'Morning'
        WHEN DATEPART(HOUR, sale_time) BETWEEN 12 AND 17 THEN 'Afternoon'
        ELSE 'Evening'
```

No issues found

	shift	orders
1	Evening	1062
2	Morning	548
3	Afternoon	377

Fig xx

In Fig xx, it categorizes transactions into shifts and shows order volume per time of day.

6 Step 5: Reporting & Insights

The study yields information that directs operations, marketing, and retailing. Examples and practical suggestions are provided below;

- **Category Performance:** Priority funding for marketing and inventories should go to the categories with the highest overall sales.

- **Customer Demographics:** If younger customers are drawn to beauty, focus your campaigns on platforms that appeal to them.
- **High-Value Orders:** The majority of transactions over 1000 fall into particular categories; take into account protection plans and premium packages.
- **Seasonality:** Peak months, which are typically November through December, call for more staff, inventory, and promotions.
- **Top Customers:** Incorporate targeted incentives and loyalty benefits to increase lifetime value and retention.
- **Shift Patterns:** Adjusting staffing and in-store promotions for the store window is advised due to afternoon traffic peaks.

Explore the full project in the link below

https://drive.google.com/file/d/1BG0bSIaOiXNTV4fhXBEuFXRPExaRDaMV/view?usp=drive_link