Case Study

Database Design & development

E-commerce Platform

MANUMON SCARIA

https://github.com/scariams

Table of content:

INTRODUCTION

E-commerce has transformed business operations and consumer shopping habits, making it a vital component of the modern digital economy. Industry leaders like Amazon have set new standards by offering vast product selections, personalized shopping experiences, and efficient logistics. However, maintaining high customer satisfaction remains a critical factor in sustaining success in this highly competitive market.

This case study explores the significance of customer satisfaction in e-commerce, with a specific focus on Amazon, by leveraging SQL-based data analysis. The research aims to evaluate customer satisfaction in online retail by examining key factors such as product ratings, customer reviews, return rates, and service response times.

Through structured data analysis, this study involves creating SQL tables and executing queries to identify meaningful patterns. By utilizing SQL-based data retrieval and analysis, it seeks to uncover the common factors influencing customer satisfaction and provide actionable insights for enhancing e-commerce operations.

Mission

To provide a seamless, user-friendly e-commerce platform connecting buyers with a wide range of products globally

Objectives

The objective of this case study is to understand how e-commerce platforms, like Amazon, affect customer satisfaction. By looking at real data, the study aims to find out what makes customers happy or dissatisfied when shopping online. This helps businesses improve their services and better meet customer needs.

The case study also focuses on gaining practical experience in creating and managing databases using SQL queries. Students will learn how to build databases to store and organize e-commerce data, such as customer orders, reviews, and product information. They will work with SQL commands to create tables, insert data, and retrieve useful information.

Overall, the case study helps participants understand both the importance of customer satisfaction in e-commerce and the technical skills needed to manage and analyse data using SQL. This combination of theory and hands-on practice prepares students for data-driven decision-making in online businesses

Database design & table

The database and tables listed below show the structure of the key tables, including the data types of the fields and the allocated space for each data type

Table 1: SELLERS

It represents a database table schema for a "Seller" entity. The SELLER ID is the unique identifier with an allotted length of 20. The VARCHAR data type is used to store strings or variable-length text.

SELLER ID	INTEGER(20)
Name	VARCHAR(100)
Email	VARCHAR(100)
Phone number	VARCHAR(15)
Address	VARCHAR(255)

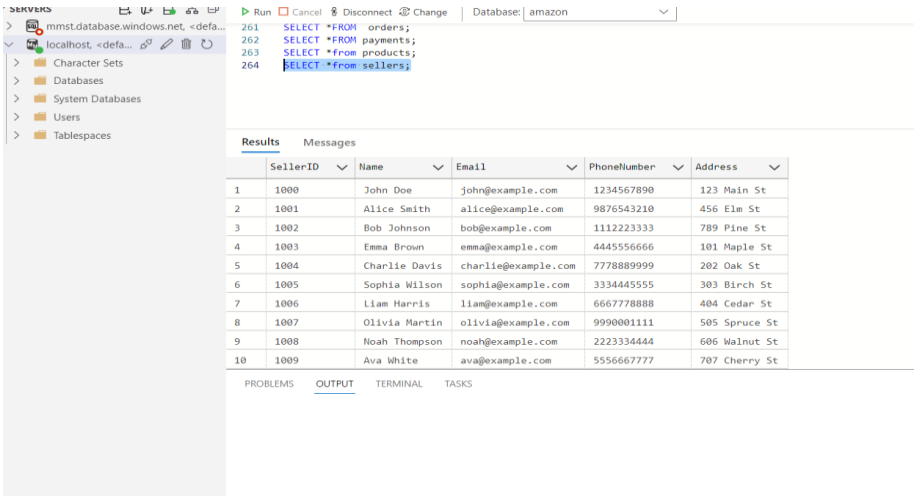


Table 2: PRODUCTS

In this product table, the unique key is **Product ID**, while **Seller ID** and **Order ID** are foreign keys. **Category ID** is an attribute

PRODUCTS :–

PRODUCT ID	INTEGER(20)
Seller id	INTEGER(20)
Order id	INT(20)
Category id	INT(20
Product name	VARCHAR(100
Price	DECIMAL
Rating	DECIMAL
Brands	VARCHAR(50)

```
262  SELECT *FROM payments;
263  SELECT *from products;
264  SELECT *from sellers;
```

Results Messages

	ProductID	SellerID	OrderID	CategoryID	ProductName	Price	Rating	Brand
1	3000	1000	4000	101	Laptop	1200.00	4.50	Dell
2	3001	1001	4001	102	Smartphone	800.00	4.20	Samsung
3	3002	1002	4002	103	Headphones	150.00	4.70	Sony
4	3003	1003	4003	104	Smartwatch	250.00	4.30	Apple
5	3004	1004	4004	105	Tablet	600.00	4.10	Lenovo
6	3005	1005	4005	106	Camera	1000.00	4.60	Canon
7	3006	1006	4006	107	Gaming Console	500.00	4.80	Nintendo
8	3007	1007	4007	108	TV	900.00	4.40	LG
9	3008	1008	4008	109	Monitor	300.00	4.50	HP
10	3009	1009	4009	110	Keyboard	50.00	4.00	Logitech

PROBLEMS OUTPUT TERMINAL TASKS

Table 3: customers

The **Customer** table represents customer data. **Customer ID** is the primary key, stored as an integer with a length of up to 20 characters. The **VARCHAR** data type is used to store variable-length text values.

CUSTOMERS:-

Customer id	INTEGER(20
Name	VARCHAR(100
Email	VARCHAR(100)
Phone Number	VARCHAR(15)
Address	VARCHAR(255

```
263    SELECT *from products;
264    SELECT *from sellers;
265    SELECT * FROM Customers;
```

Results Messages

	CustomerID	Name	Email	PhoneNumber	Address
1	2000	Michael Scott	michael@example.com	1112223333	1 Office St
2	2001	Dwight Schrute	dwight@example.com	4445556666	2 Beet Farm
3	2002	Jim Halpert	jim@example.com	7778889999	3 Scranton St
4	2003	Pam Beesly	pam@example.com	3334445555	4 Art St
5	2004	Stanley Hudson	stanley@example.com	6667778888	5 Crossword St
6	2005	Kevin Malone	kevin@example.com	9990001111	6 Chili St
7	2006	Angela Martin	angela@example.com	2223334444	7 Cat St
8	2007	Oscar Martinez	oscar@example.com	5556667777	8 Accounting St
9	2008	Toby Flenderson	toby@example.com	8889990000	9 HR St
10	2009	Ryan Howard	ryan@example.com	1231231234	10 Temp St

PROBLEMS OUTPUT TERMINAL TASKS

Table 4: orders

The "orders" table has an **order_id** as the primary key, with **customer id** and **review_id** as foreign keys linking to the respective customer and review tables. The **amount** is stored as a **decimal** data type to handle numeric values with precision, allowing for up to two decimal places for accurate monetary representation.

ORDERS :-

Order id	INTEGER(20)
Customer id	INTEGER(20)
Shopping Address	VARCHAR(255)
Amount	DECIMAL
Order Date	DATETIME
Review id	INT(20)

```
260    SELECT * FROM Customers;
261    SELECT *FROM orders;
262    SELECT *FROM payments;
263    SELECT *from products;
264    SELECT *from sellers;
265    SELECT * FROM Customers;
```

Results Messages

	OrderID	CustomerID	ShippingAddress	Amount	OrderDate	ReviewsID
1	4000	2000	123 Main St	1200.00	2025-01-01 10:00:00	7000
2	4001	2001	456 Elm St	800.00	2025-01-02 11:00:00	7001
3	4002	2002	789 Pine St	150.00	2025-01-03 12:00:00	7002
4	4003	2003	101 Maple St	250.00	2025-01-04 13:00:00	7003
5	4004	2004	202 Oak St	600.00	2025-01-05 14:00:00	7004
6	4005	2005	303 Birch St	1000.00	2025-01-06 15:00:00	7005
7	4006	2006	404 Cedar St	500.00	2025-01-07 16:00:00	7006
8	4007	2007	505 Spruce St	900.00	2025-01-08 17:00:00	7007
9	4008	2008	606 Walnut St	300.00	2025-01-09 18:00:00	7008
10	4009	2009	707 Cherry St	50.00	2025-01-10 19:00:00	7009

PROBLEMS OUTPUT TERMINAL TASKS

Table 5: shipping

In the "shipping" table, the **shipping_id** is the primary key, while the **order_id** serves as a foreign key, linking the shipping information to the corresponding order in the "orders" table.

SHIPPING :-

Shipping id	INTEGER(20)
Order id	INT(20)
Tracking Number	VARCHAR(50)
Delivary Date	DATETIME
Shipping Date	DATETIME
Shipping Address	VARCHAR(255)

```
266  SELECT *FROM shipping
```

Results Messages

	ShippingID	OrderID	TrackingNumber	DeliveryDate	ShippingDate	ShippingAddress
1	6000	4000	TRK123456	2025-01-05 10:00:00	2025-01-02 09:00:00	123 Main St
2	6001	4001	TRK234567	2025-01-06 11:00:00	2025-01-03 10:00:00	456 Elm St
3	6002	4002	TRK345678	2025-01-07 12:00:00	2025-01-04 11:00:00	789 Pine St
4	6003	4003	TRK456789	2025-01-08 13:00:00	2025-01-05 12:00:00	101 Maple St
5	6004	4004	TRK567890	2025-01-09 14:00:00	2025-01-06 13:00:00	202 Oak St
6	6005	4005	TRK678901	2025-01-10 15:00:00	2025-01-07 14:00:00	303 Birch St
7	6006	4006	TRK789012	2025-01-11 16:00:00	2025-01-08 15:00:00	404 Cedar St
8	6007	4007	TRK890123	2025-01-12 17:00:00	2025-01-09 16:00:00	505 Spruce St
9	6008	4008	TRK901234	2025-01-13 18:00:00	2025-01-10 17:00:00	606 Walnut St
10	6009	4009	TRK012345	2025-01-14 19:00:00	2025-01-11 18:00:00	707 Cherry St

PROBLEMS OUTPUT TERMINAL TASKS

Table 6: payments

In the "payment" table, the **payment_id** serves as both the unique key and primary key, while the **order_id** is a foreign key that links the payment to the corresponding order in the "orders" table

PAYMENTS :-

Payment id	INTEGER(20)
Order id	INT(20)
Payment Method	VARCHAR(50)

```
261    SELECT *FROM  orders;
262    SELECT *FROM payments;
263    SELECT *from products;
264    SELECT *from sellers;
265    SELECT * FROM Customers;
```

Results Messages

	PaymentID	OrderID	Amount	PaymentMethod	PaymentDate
1	5000	4000	1200.00	Credit Card	2025-01-01 10:30:00
2	5001	4001	800.00	PayPal	2025-01-02 11:30:00
3	5002	4002	150.00	Debit Card	2025-01-03 12:30:00
4	5003	4003	250.00	Bank Transfer	2025-01-04 13:30:00
5	5004	4004	600.00	Credit Card	2025-01-05 14:30:00
6	5005	4005	1000.00	PayPal	2025-01-06 15:30:00
7	5006	4006	500.00	Debit Card	2025-01-07 16:30:00
8	5007	4007	900.00	Bank Transfer	2025-01-08 17:30:00
9	5008	4008	300.00	Credit Card	2025-01-09 18:30:00
10	5009	4009	50.00	PayPal	2025-01-10 19:30:00

PROBLEMS OUTPUT TERMINAL TASKS

Table 7: reviews

The review table has a review ID as the primary key, and the rating of the product is represented as a decimal data type

REVIEWS :-

Review id	INTEGER(20)
Rating	DECIMAL
Review Text	TEXT

```
265    SELECT * FROM Customers;
266    SELECT *FROM shipping
267    SELECT *FROM reviews
```

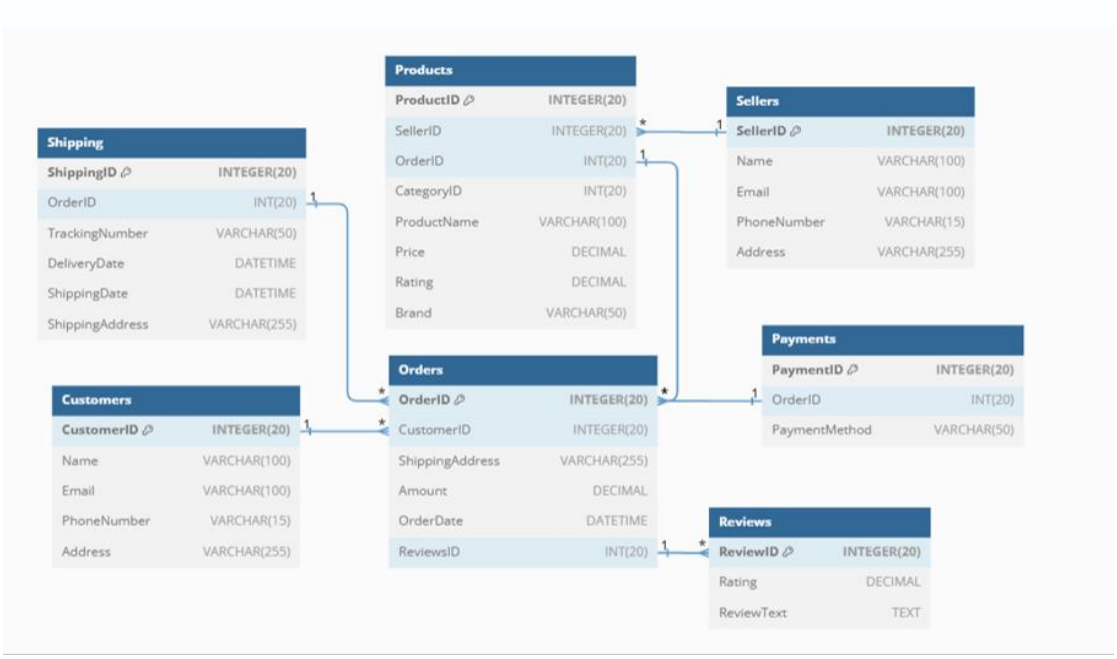
Results Messages

	ReviewID	Rating	ReviewText
1	7000	4.50	Great product, fast shipping!
2	7001	4.20	Good value for money.
3	7002	4.70	Excellent quality, highly recommended!
4	7003	4.30	Works as expected, no issues.
5	7004	4.10	Decent product, but could be better.
6	7005	4.60	Very happy with the purchase.
7	7006	4.80	Fantastic! Best purchase ever.
8	7007	4.40	Solid build quality and performance.
9	7008	4.50	Met my expectations, would buy again.
10	7009	4.00	Average, but does the job.

PROBLEMS OUTPUT TERMINAL TASKS

Entity-Relationship Diagram (ERD)

The entity-relationship diagram describes the relationships between each table and how they are connected. For instance, it illustrates whether the relationship is one-to-one, one-to-many, or many-to-many, along with the primary keys and foreign keys involved



1, Customers ↔ Orders (One-to-Many)

- A customer can place multiple orders.
- Each order belongs to a single customer (**CustomerID** in the **Orders** table).

2, Orders ↔ Products (Many-to-Many)

- A single order can contain multiple products.
- A product can be part of multiple orders.

3, Orders ↔ Shipping (One-to-One)

- Each order has one shipping record.
- Each shipping record corresponds to a single order (**OrderID** in the **Shipping** table).

4, Orders ↔ Payments (One-to-One)

- Each order is associated with a single payment.
- Each payment corresponds to a single order (**OrderID** in the **Payments** table).

5, Orders ↔ Reviews (One-to-One)

- An order can have one review.
- Each review belongs to a single order (**ReviewID** in the **Orders** table

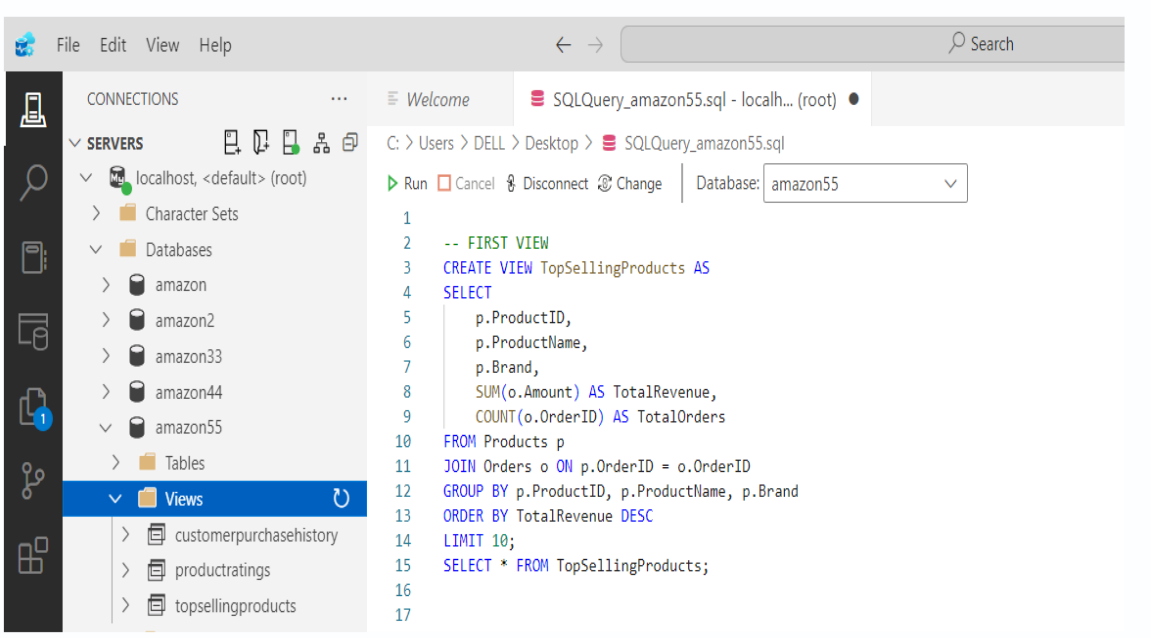
6, Products ↔ Sellers (Many-to-One)

- A seller can sell multiple products.
- Each product belongs to a single seller (**SellerID** in the **Products** table).

Views

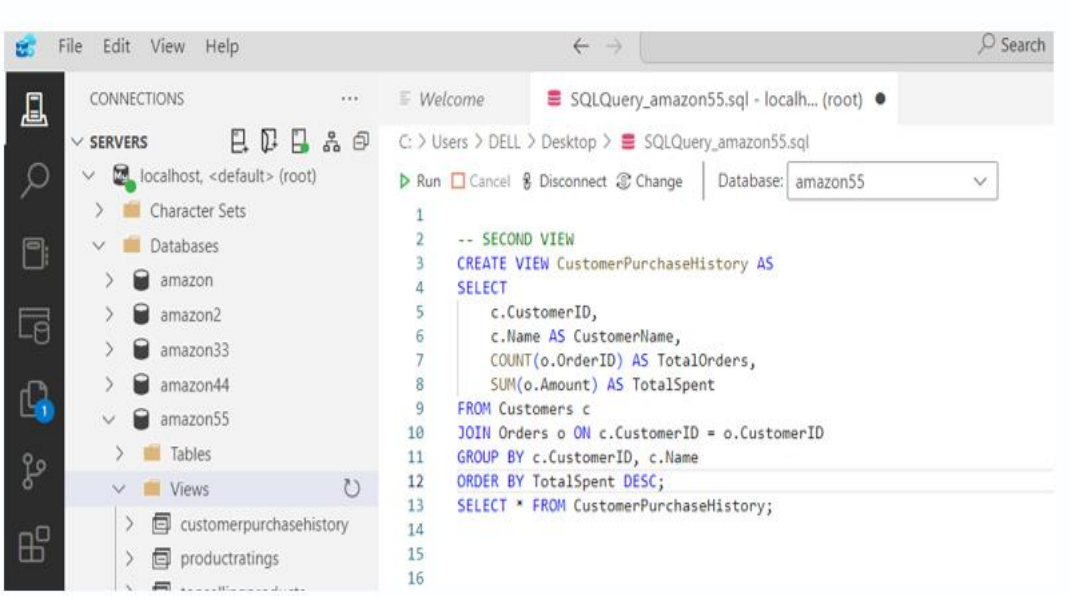
A view is a virtual table based on the result of a query. In this case study, the view identifies the best-selling products by total revenue and order count, helping sellers understand which products generate the most sales on an e-commerce platform

View 1: Top-Selling Products by Revenue



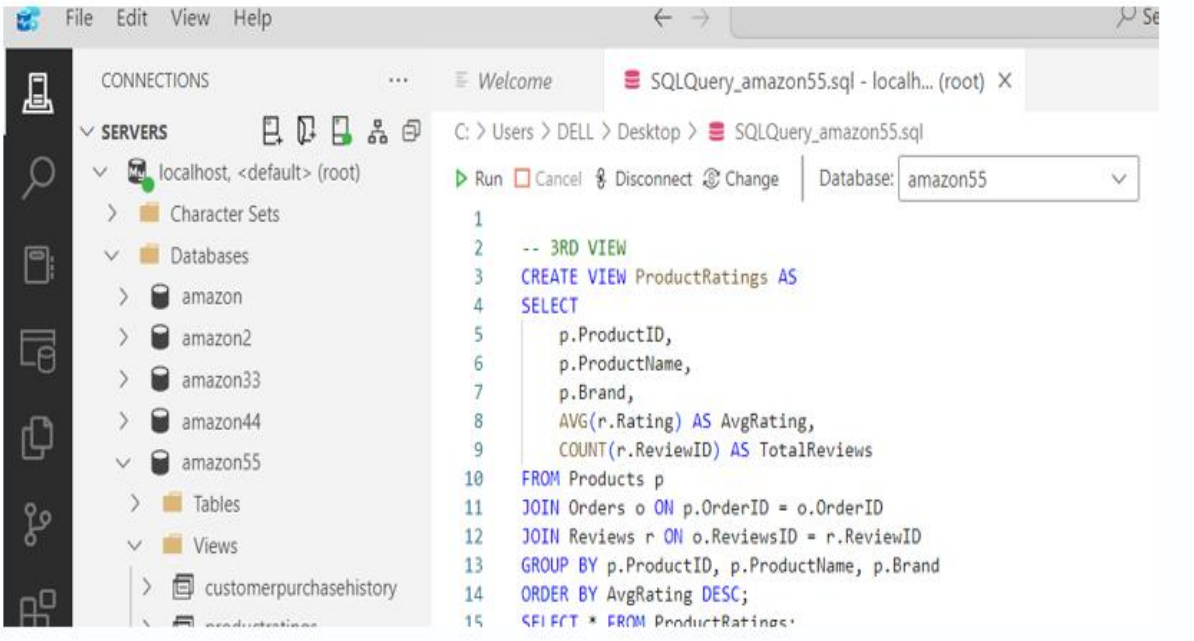
The view helps identify the best-selling products and allows us to understand which products sell more and which sell less

View 2: Customer Purchase History



The customer purchase view is used to identify the purchasing history of customers. This helps us understand customer insights and enables us to create a targeted market strategy based on the data

View 3: Product Ratings



The benefit of this view is that it helps us identify customer satisfaction with the product and aids in inventory management

Conclusion

A database plays a pivotal role in the success of e-commerce companies by supporting various key functions that drive operations. It is instrumental in efficient inventory management, allowing e-commerce companies like Amazon to track products, stock levels, and demand patterns in real-time. This ensures that products are readily available for customers while reducing the chances of overstocking or stockouts.

The database also boosts operational efficiency by streamlining processes related to order fulfilment, payment processing, and customer service. With a centralized system, e-commerce companies can quickly retrieve and update information, ensuring smooth day-to-day operations across their vast networks of warehouses, distribution centres, and delivery systems.

Furthermore, the database significantly enhances customer satisfaction. By collecting and analyzing real-time data on products, orders, and customer preferences, e-commerce companies can personalize their offerings and tailor the shopping experience to individual needs. The ability to make data-driven decisions enables companies to adjust marketing, pricing, and promotions based on customer behaviours and trends.

Additionally, the database optimizes the supply chain by providing insights into inventory levels, order status, and shipment tracking. This leads to faster deliveries and more accurate forecasting. Ultimately, a well-designed database is integral to a company’s success. It supports a seamless and personalized shopping experience by delivering the right products at the right time, ensuring that customers receive high-quality service and satisfaction.

With the power of real-time data, e-commerce companies are not only able to optimize internal operations but also improve customer engagement and loyalty, maintaining their position as leading players in the market. The company's mission to provide an efficient, customer-centric platform relies heavily on the effective use of its database, making it a cornerstone of its business strategy.