E-Commerce Company Case Study

By. Parag Fartode

paragfartode34@gmail.com

Overview:

This project focuses on analysing the extensive datasets of an e-commerce company to extract valuable insights that drive business strategies across various departments. As a data analyst, your role is to leverage SQL and data analysis techniques to inform decision-making in areas such as customer insights, product performance, sales optimization, and inventory management. The insights derived from this analysis will support the company in enhancing customer satisfaction, optimizing operations, and boosting overall sales performance.

Business Context:

My work will have a direct impact on the following business verticals:

- **Customer Insights:** Understanding the customer base to tailor marketing strategies that improve customer engagement and loyalty.
- **Product Analysis:** Evaluating product performance to make informed decisions about stock levels, pricing strategies, and promotional activities.
- **Sales Optimization:** Analysing sales data to identify trends, opportunities, and areas for improvement in the sales process.
- **Inventory Management:** Managing stock levels to ensure product availability while minimizing excess inventory and associated costs.

Dataset Details:

The datasets used in this case study include:

- **Customers Dataset:** Contains information on customer IDs, names, and locations.
- **Products Dataset:** Includes product IDs, names, categories, and prices.
- **Orders Dataset:** Captures details such as order IDs, order dates, customer IDs, and total order amounts.
- **OrderDetails Dataset:** Provides information on order IDs, product IDs, quantities, and prices per unit.

Introduction:

In the modern e-commerce landscape, data-driven decision-making is crucial for maintaining a competitive edge. This project explores how data analysis can be used to optimize various aspects of an e-commerce business, from understanding customer behaviour to managing inventory efficiently. By utilizing SQL queries to analyse the company's datasets, we can uncover patterns, trends, and insights that will inform strategic decisions across different business functions.

Scope:

- **Customer Segmentation:** Analysing customer data to identify distinct segments based on behaviour, location, and purchasing patterns.
- **Product Performance Analysis:** Evaluating product sales and profitability to determine which products should be prioritized in marketing efforts and inventory management.
- **Sales Trend Analysis:** Identifying trends in sales data over time to predict future sales performance and inform promotional strategies.
- **Inventory Optimization:** Ensuring that stock levels are aligned with customer demand, reducing the risk of stockouts or excess inventory.

Working:

This project involves executing SQL queries to extract, manipulate, and analyze data from the company's databases. The analysis will be performed on different datasets, with each query designed to answer specific business questions. The results of these analyses will be compiled into reports that provide actionable insights for various stakeholders within the company.

Purpose:

The primary purpose of this project is to demonstrate how data analysis can be used to improve business outcomes in an e-commerce setting. By analysing key datasets, we can provide the company with the information needed to make data-driven decisions that enhance customer satisfaction, optimize product offerings, and improve overall operational efficiency.

Goals of the Proposed System:

1. **Data-Driven Decision Making:** Enable the company to make informed decisions based on accurate and timely data.
2. **Customer-Centric Strategies:** Develop marketing strategies that are tailored to the needs and preferences of different customer segments.
3. **Optimized Inventory Management:** Maintain optimal stock levels to meet customer demand without incurring unnecessary costs.
4. **Improved Sales Performance:** Identify opportunities to increase sales through trend analysis and targeted promotions.
5. **Enhanced Operational Efficiency:** Streamline processes related to product management, sales, and inventory control.

Technical Feasibility:

This project is technically feasible, given the availability of the required datasets and the capability to execute complex SQL queries. The backend of the system is supported by a robust SQL database, which allows for efficient data retrieval and manipulation. The system is designed to be scalable, accommodating the growth of the company's data and analytical needs over time.

Advantages:

1. **Improved Customer Understanding:** By analysing customer data, the company can better understand its customers and tailor its offerings to meet their needs.
2. **Optimized Product Offerings:** Product analysis enables the company to focus on high-performing products and phase out underperforming ones.
3. **Increased Sales:** Sales trend analysis helps the company identify opportunities to boost sales through targeted promotions and pricing strategies.
4. **Efficient Inventory Management:** Proper inventory management ensures that the company can meet customer demand without overstocking, reducing costs and improving cash flow.
5. **Enhanced Reporting:** The system provides detailed reports that give stakeholders a clear view of the company's performance across different areas.

Summary:

This project demonstrates the importance of data analysis in driving business success in the e-commerce industry. By analysing key datasets, we can provide the company with the insights needed to make informed decisions that enhance customer satisfaction, optimize product offerings, and improve overall operational efficiency. The implementation of this system will enable the company to maintain a competitive edge in the rapidly evolving e-commerce landscape.

Tables and Attributes:

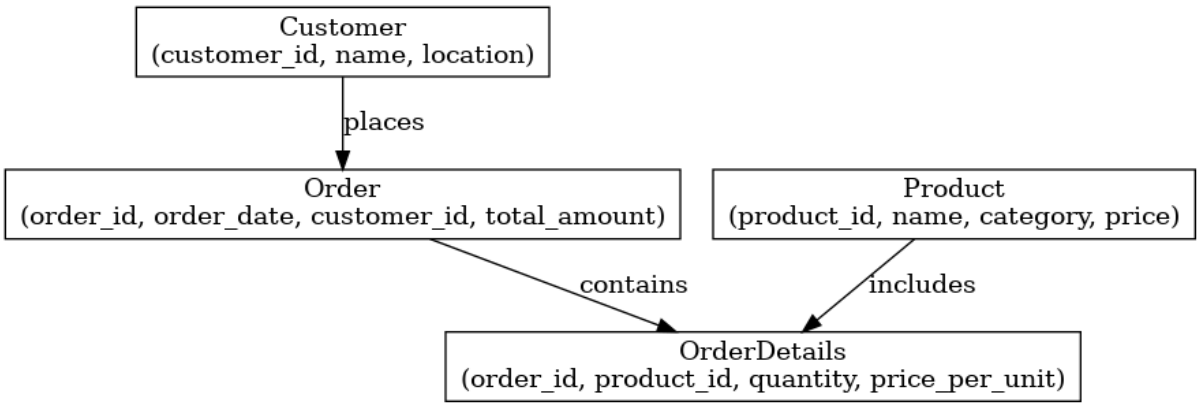
1. **Customers**:
    - customer_id (Primary Key)
    - name
    - location
2. **Products**:
    - product_id (Primary Key)
    - name
    - category
    - price
3. **Orders**:
    - order_id (Primary Key)
    - order_date
    - customer_id (Foreign Key referencing Customers.customer_id)
    - total_amount
4. **OrderDetails**:
    - order_id (Composite Primary Key, Foreign Key referencing Orders.order_id)
    - product_id (Composite Primary Key, Foreign Key referencing Products.product_id)
    - quantity
    - price_per_unit

Relationships:

- **Customers → Orders**: 1-to-many (One customer can place many orders)
- **Orders → OrderDetails**: 1-to-many (One order can contain multiple order details)
- **Products → OrderDetails**: 1-to-many (One product can appear in multiple order details).

ER Diagram Description:

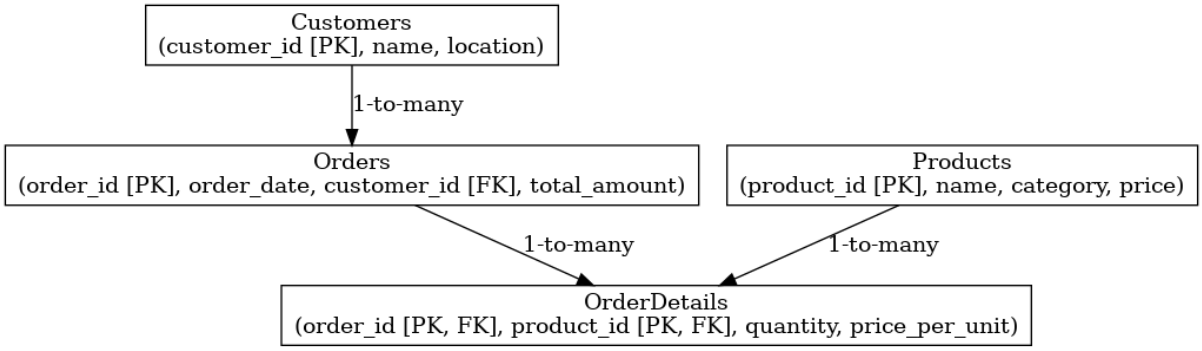
- The **Customer** entity is connected to the **Order** entity with a 1-to-many relationship.
- The **Order** entity is connected to the **OrderDetails** entity with a 1-to-many relationship.
- The **Product** entity is connected to the **OrderDetails** entity with a 1-to-many relationship.



Relational Schema:

Relationships:

- **Customers → Orders**: 1-to-many (One customer can place many orders)
- **Orders → OrderDetails**: 1-to-many (One order can contain multiple order details)
- **Products → OrderDetails**: 1-to-many (One product can appear in multiple order details)



SQL CODE IMPLEMENTATION:

---------- 1. Customer Insights ----------

---------- 1.1. Top Locations by Total Spend----------

```
SELECT c.location, SUM(o.total_amount) AS total_spend
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.location
ORDER BY total_spend DESC;
```

-- Chennai has most amount spend 3890000/-

----------1. 2. Most Frequent Customers ----------

```
SELECT c.customer_id, c.name, COUNT(o.order_id) AS total_orders
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name
ORDER BY total_orders DESC;
```

-- Divya Upadhyay has most orders with custumer_id 57 has 8 orders.

---------- 1.3. Customer Purchase Patterns by Product Category----------

```
SELECT c.customer_id, c.name, p.category, COUNT(od.product_id) AS
total_purchases
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
JOIN OrderDetails od ON o.order_id = od.order_id
JOIN Products p ON od.product_id = p.product_id
GROUP BY c.customer_id, c.name, p.category
ORDER BY total_purchases DESC;
```

-- Romil Bora purchase 13 times

---------- 1.4. Average Spending Per Customer ----------

```
SELECT c.customer_id, c.name, AVG(o.total_amount) AS avg_spending
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name
ORDER BY avg_spending DESC;
```

-- Sahil Chaudhary avg_spending is most 321000.0000/-

---------- 1.5. Customer Retention Analysis ----------

```
SELECT c.customer_id, c.name, COUNT(o.order_id) AS order_count
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name
HAVING order_count > 1
ORDER BY order_count DESC
LIMIT 5;
```

-- Divya Upadhyay, Romil Bora, Saksham Buch, Indran's Lanka, Ayesha Wali are genuine customers.

---------- 2. Product Analysis ----------

----------2.1. Top-Selling Products----------

```
SELECT p.product_id, p.name, SUM(od.quantity) AS total_sold
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name
ORDER BY total_sold DESC;
```

-- Digital SLR Camera sold most and Least Smartphone 6"

----------2.2 Revenue by Product Category----------

```
SELECT p.category, SUM(od.quantity * od.price_per_unit) AS total_revenue
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.category
ORDER BY total_revenue DESC;
```

-- Electronics items has most revenue 12758000/-

---------- 3. Price Performance Analysis ----------

```
SELECT p.product_id, p.name, p.price, SUM(od.quantity) AS total_sold,
SUM(od.quantity * od.price_per_unit) AS total_revenue
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name, p.price
ORDER BY total_revenue DESC;
```

-- Laptop 15"Pro has most revenue 7560000/-

---------- 4. Least Selling Products----------

```
SELECT p.product_id, p.name, SUM(od.quantity) AS total_sold
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name
ORDER BY total_sold ASC
LIMIT 3;
```

-- Smartphone 6” , Smartwatch Fitness Tracker and Laptop 15" Pro Sold less 108,119,126 respectively .

---------- 5. Product Category Sales Trends ----------

```
SELECT p.category, MONTH(o.order_date) AS month, SUM(od.quantity) AS
total_sold
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
JOIN Orders o ON od.order_id = o.order_id
GROUP BY p.category, month
ORDER BY month ASC, total_sold DESC;
```

-- In September (9) sales of Electronics are high, In July (7) sales of Wearable Tech are high, In December (12) Sales is high of Photography.

---------- 3. Sales Optimization ----------
---------- 3.1. Monthly Sales Performance----------

```
SELECT MONTH(o.order_date) AS month, SUM(o.total_amount) AS total_sales
FROM Orders o
GROUP BY month
ORDER BY total_sales DESC;
```

-- September (9) month sale is highest 2927000/-

---------- 3.2. Highest Revenue Orders----------

```
SELECT o.order_id, o.order_date, SUM(od.quantity * od.price_per_unit) AS
order_revenue
FROM Orders o
JOIN OrderDetails od ON o.order_id = od.order_id
GROUP BY o.order_id, o.order_date
ORDER BY order_revenue DESC
LIMIT 10;
```

-- Single day sale or order_id 188 in highest on 2023-07-01

---------- 3.3. Sales Trends Over Time----------

```
SELECT YEAR(o.order_date) AS year, MONTH(o.order_date) AS month,
SUM(o.total_amount) AS total_sales
FROM Orders o
GROUP BY year, month
ORDER BY year ASC, month ASC;
```

-- in February, march sales downs.

---------- 3.4. Average Order Value----------

```
SELECT AVG (total_amount) AS avg_order_value
FROM Orders;
```

-- avg_order_value 98915.00.

---------- 3.5. Sales by Product Category----------

```
SELECT p.category, SUM(od.quantity * od.price_per_unit) AS total_sales
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.category
ORDER BY total_sales DESC;
```

-- Electronics 12758000/-, Photography 6040000/-, Wearable Tech 985000/-

---------- 4. Inventory Management ----------
---------- 4.1. Total Stock Sold by Product----------

```
SELECT p.product_id, p.name, SUM(od.quantity) AS total_sold
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name
ORDER BY total_sold DESC;
```

-- Digital SLR Camera sold most 151.

----------4.2. Low Stock Products (assuming 130) ----------

```
SELECT p.product_id, p.name, SUM(od.quantity) AS total_sold
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name
HAVING total_sold < 130
ORDER BY total_sold ASC;
```

-- Smartphone 6”, Smartwatch Fitness Tracker, Laptop 15"Pro are our least in stocks

---------- 4.3. Inventory Turnover Rate (based on sales) ----------

```
SELECT p.product_id, p.name, SUM(od.quantity*od.price_per_unit) AS
total_turnover
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name
ORDER BY total_turnover DESC;
```

--Laptop 15 Pro has maximum turnover rate of 7560000/-

----------4. 4. Inventory Usage Over Time----------

```
SELECT p.product_id, p.name, MONTH(o.order_date) AS month, SUM(od.quantity)
AS total_used
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
JOIN Orders o ON od.order_id = o.order_id
GROUP BY p.product_id, p.name, month
ORDER BY month ASC, total_used DESC;
```

---------- 4.5. High Volume Products----------

```
SELECT p.product_id, p.name, SUM(od.price_per_unit) AS total_volume
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name
HAVING total_volume > 400000
ORDER BY total_volume DESC;
```

-- Laptop 15" Pro, Digital SLR Cameramen-Book Reader, Smartphone 6", Bluetooth Headphones, Portable Bluetooth Speaker are High Volume products.

---------- Customer Insights: High vs. Low Spending Customers ----------

```
SELECT c.customer_id, c.name,
    CASE
        WHEN SUM(o.total_amount) > 150000 THEN 'High Spender'
        ELSE 'Low Spender'
    END AS spending_category
FROM Customers c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name;
```

------ Product Analysis: Classify Products as High, Medium, or Low Priced -------

```
SELECT p.product_id, p.name, p.price,
    CASE
        WHEN p.price > 20000 THEN 'High Priced'
        WHEN p.price BETWEEN 10000 AND 20000 THEN 'Medium Priced'
        ELSE 'Low Priced'
    END AS price_category
FROM Products p;
```

---------- Sales Optimization Order Size Classification----------

```
SELECT o.order_id, o.order_date, o.total_amount,
    CASE
        WHEN o.total_amount > 50000 THEN 'Large Order'
        WHEN o.total_amount BETWEEN 30000 AND 50000 THEN 'Medium Order'
        ELSE 'Small Order'
    END AS order_size
FROM Orders o;
```

---------- Inventory Management: Sales Volume Category by Product ----------

```
SELECT p.product_id, p.name, SUM(od.quantity) AS total_sold,
    CASE
        WHEN SUM(od.quantity) > 140 THEN 'High Volume'
        WHEN SUM(od.quantity) BETWEEN 120 AND 140 THEN 'Medium Volume'
        ELSE 'Low Volume'
    END AS sales_volume_category
FROM Products p
JOIN OrderDetails od ON p.product_id = od.product_id
GROUP BY p.product_id, p.name;
```

---------- Customer Insights: Geographic Customer Analysis----------

```
 SELECT c.customer_id, c.name, c.location,
      CASE
         WHEN c.location in ('Delhi','jaipur','lucknow') THEN 'North Region'
         WHEN c.location in  ('Ahmedabad','Mumbai','Pune') THEN 'West Region'
         WHEN c.location in ('Chennai','Hyderabad','Lucknow') THEN 'South Region'
         ELSE 'East Region'
      END AS region
FROM Customers c;
```

---------- Region Wise Sales ----------

```
SELECT
      CASE
         WHEN c.location IN ('Delhi', 'Jaipur', 'Lucknow') THEN 'North Region'
         WHEN c.location IN ('Ahmedabad', 'Mumbai', 'Pune') THEN 'West Region'
         WHEN c.location IN ('Chennai', 'Hyderabad', 'Bangalore') THEN 'South
Region'
         WHEN c.location = 'Kolkata' THEN 'East Region'
      END AS region,
      SUM(o.total_amount) AS region_sales
FROM Customers c
LEFT JOIN orders o
ON c.customer_id = o.customer_id
GROUP BY CASE
         WHEN c.location IN ('Delhi', 'Jaipur', 'Lucknow') THEN 'North Region'
         WHEN c.location IN ('Ahmedabad', 'Mumbai', 'Pune') THEN 'West Region'
         WHEN c.location IN ('Chennai', 'Hyderabad', 'Bangalore') THEN 'South
Region'
         WHEN c.location = 'Kolkata' THEN 'East Region'
        END
ORDER BY region_sales DESC;
-- South Region Has maximum sales 7378000/-
```

==========

==========