SMART RETAIL DATABASE

OPTIMIZING ORDERS, INVENTORY, AND CUSTOMER INSIGHTS

PREPARED BY: ANURAG YADAV

Date- 20/03/2025

2. Abstract

This project aims to design an optimized retail database system to streamline sales, inventory management, and customer engagement. By establishing structured relationships among various entities, the system ensures seamless order processing, accurate stock tracking, and efficient workforce management. The project leverages an Entity-Relationship Diagram (ERD) to illustrate the database structure, highlighting key relationships such as customers and orders, products and categories, and staff and stores. The outcome of this project demonstrates how a well-structured database can enhance business operations, improve decision-making, and drive profitability.

3. Table of Contents

4. Introduction

- **Project Overview & Objectives**

  Retail businesses require a robust database system to manage their operations efficiently. This project presents an ERD-based retail database that integrates key aspects such as customer management, order processing, product categorization, and stock control. The primary objective is to develop a structured database model that enables real-time tracking of sales, inventory levels, and customer interactions, thus improving overall business efficiency.

5. Data Description

- **Dataset Source & Acquisition**

  The dataset used for this project is designed to simulate real-world retail operations. It includes data related to customers, orders, products, staff, and stores. The data structure is based on industry-standard retail models. This database was downloaded from **Kaggle.com**, where it was published by **Dillon Myrick**.

- **Dataset Characteristics**

  The dataset consists of various tables that store detailed information about retail transactions:

  - ✓ ****Customers****: Stores customer details such as name, contact information, and address.
  - ✓ ****Orders****: Captures purchase transactions, including order dates, customer IDs, and store locations.
  - ✓ ****Products****: Contains product details, including brand, category, and pricing information.
  - ✓ ****Stores****: Maintains store-specific information for better regional management.
  - ✓ ****Staff****: Tracks employee records, linking them to their respective stores.
  - ✓ ****Stocks****: Manages real-time inventory levels across multiple stores.

  The dataset covers a simulated period of transactions to analyze sales trends, stock movements, and customer behavior.

6. Methodology

The methodology for this project follows a structured approach to designing, analyzing, and optimizing the retail database. The key steps include:

- **Data Collection & Understanding**
  - ✓ The dataset was sourced from **Kaggle.com**, originally published by **Dillon Myrick**.
  - ✓ It includes essential retail entities such as customers, orders, products, staff, and stores.
  - ✓ The structure was examined to ensure it aligns with industry-standard retail models.
- **Entity-Relationship Diagram (ERD) Development**
  - ✓ The dataset was analysed to identify key entities and their relationships.
  - ✓ An **Entity-Relationship Diagram (ERD)** was designed to visualize connections among entities, ensuring efficient data flow.
  - ✓ Relationships were defined with appropriate cardinalities, optimizing data integrity and performance.
- **Database Normalization & Optimization**
  - ✓ The schema was structured to follow normalization principles, reducing redundancy and improving efficiency.
  - ✓ Foreign keys and indexing strategies were implemented to enhance query performance.
  - ✓ Business rules were applied to maintain data consistency, such as enforcing unique customer emails and linking orders to specific stores.
- **Retail Analytics & Insights Generation**
  - ✓ Relationships between orders, customers, and products were leveraged to extract valuable business insights.
  - ✓ The database structure supports tracking **sales trends, inventory levels, and customer behaviour**.
  - ✓ Queries were optimized to facilitate reporting on store-wise performance, product demand, and customer purchase patterns.
- **Scalability & Future Enhancements**
  - ✓ The database is designed to scale with business growth, supporting the addition of new stores, products, and customer records.
  - ✓ Additional functionalities, such as automated stock replenishment and advanced customer segmentation, can be integrated in future iterations.

By following this methodology, the project ensures a **robust, scalable, and data-driven approach** to retail management, helping businesses streamline operations and make informed decisions.

7. Mastering Retail Operations – The Power of an Optimized Database

A well-structured database is the backbone of any retail business. This Entity-Relationship Diagram (ERD) showcases a highly efficient database model designed to streamline sales, inventory management, and customer engagement. Let's dive into the key components that drive this powerful system.

- ### Customer Insights – Understanding and Engaging Buyers

Table: Customers

- ✓ Stores are essential customer details such as names, addresses, and contact numbers.
- ✓ Ensures seamless order processing and personalized marketing campaigns.
- ✓ Help in tracking purchase behavior to enhance loyalty programs.

Business Impact:

- ✓ Enables businesses to create personalized promotions and optimize customer service.

- ### The Order Engine – Seamlessly Processing Transactions

Table: Orders

- ✓ Records all purchase transactions with unique order IDs.
- ✓ Captures crucial details such as customer ID and store ID, linking sales to customers and locations.
- ✓ Powers analytical insights for sales trends and revenue tracking.

Business Impact:

- ✓ Enhances order management, ensuring smooth transactions and efficient sales tracking.

- ### Stock Command Center – Keeping Inventory in Check

Table: Stocks

- ✓ Manage real-time stock levels for each product across multiple stores.
- ✓ Prevents overstocking or stockouts by linking store ID, product ID, and available quantity.
- ✓ Helps in replenishment planning and supply chain optimization.

Business Impact:

- ✓ Reduces inventory wastage and enhances supply chain efficiency.

- **Retail Hubs – Managing Store Locations Effectively**

  Table: Stores

  - ✓ It contains store details, including store names, addresses, and contact numbers.
  - ✓ Links directly to inventory and orders to streamline logistics and regional sales tracking.
  - ✓ Provides store managers with data-driven insights for operational efficiency.

  ### Business Impact:

  - ✓ Facilitates regional performance tracking and store-level business decisions.

- **Best-Selling Brands – Driving Product Categorization**

  Table: Brands & Categories

  - ✓ Organizes products under relevant brand names and categories for efficient cataloging.
  - ✓ Ensures structured product management, simplifying customer browsing and reporting.
  - ✓ Helps in identifying high-performing brands and product trends.

  ### Business Impact:

  - ✓ Enhances product discoverability and improves merchandising strategies.

- **Precision in Order Fulfillment – Tracking Each Item Sold**

  Table: Order Items

  - ✓ Bridges the gap between orders and products by linking order ID and product ID.
  - ✓ Stores quantity details, ensuring accurate billing and inventory adjustments.
  - ✓ Supports analytics for top-selling products and bundle recommendations.

  ### Business Impact:

  - ✓ Optimizes order fulfillment, reducing errors and improving customer satisfaction.

- **The Power of Data-Driven Decisions**

  This database structure ensures an efficient, scalable, and insightful system for managing a retail business. From tracking customer purchases to monitoring stock levels and identifying best-selling products, this ERD provides a solid foundation for success.

8. Unlocking the Power of Relationships in Retail Database

A well-designed database thrives on structured relationships between its entities. This ERD establishes clear connections between customers, orders, products, and staff, ensuring seamless retail operations. Let's explore the key relationships that drive this system.

- **Customers & Orders – The One-to-Many Bond**

  Cardinality: One customer can place multiple orders, but each order belongs to a single customer.

  - ✓ **Customers (customer_id)** serve as the backbone of transactions.
  - ✓ **Each order (order_id)** references a single customer but can be part of multiple orders over time.

  This relationship fuels customer insights, order history tracking, and personalized marketing.

  Business Impact: Helps in customer loyalty programs and targeted promotions.

- **Orders & Order Items – The Many-to-Many Connection**

  Cardinality: Each order can have multiple items, and each product can appear in multiple orders.

  - ✓ The order_items table acts as a bridge between orders and products.
  - ✓ Tracks the quantity, price, and discount of each product within an order.

  This structure is key for generating invoices and tracking top-selling items.

  Business Impact: Enables precise revenue calculations and dynamic pricing strategies.

- **Orders & Stores – Mapping Purchases to Locations**

  Cardinality: Each order is linked to one store, but each store processes many orders.

  - ✓ The **store_id** in orders connects purchases to specific retail locations.
  - ✓ Allows tracking of sales performance by store, optimizing regional inventory.

  Business Impact: Assists in store-wise performance analysis and demand forecasting.

- **Products & Categories – Organizing the Retail Catalog**

  Cardinality: A product belongs to a single category, but a category contains multiple products.

  - ✓ Categories (category_id) group similar products, improving searchability.
  - ✓ Each product (product_id) is linked to a category for efficient classification.

Business Impact: Enhances customer experience by making product discovery easier.

- **Products & Brands – The Identity of Merchandise**

  Cardinality: A product belongs to a single brand, but a brand has multiple products.

  ✓ Brands (brand_id) help in differentiating products based on manufacturer or label.

  ✓ Essential for brand-level sales analysis and strategic partnerships.

  Business Impact: Helps retailers negotiate deals and manage brand promotions.

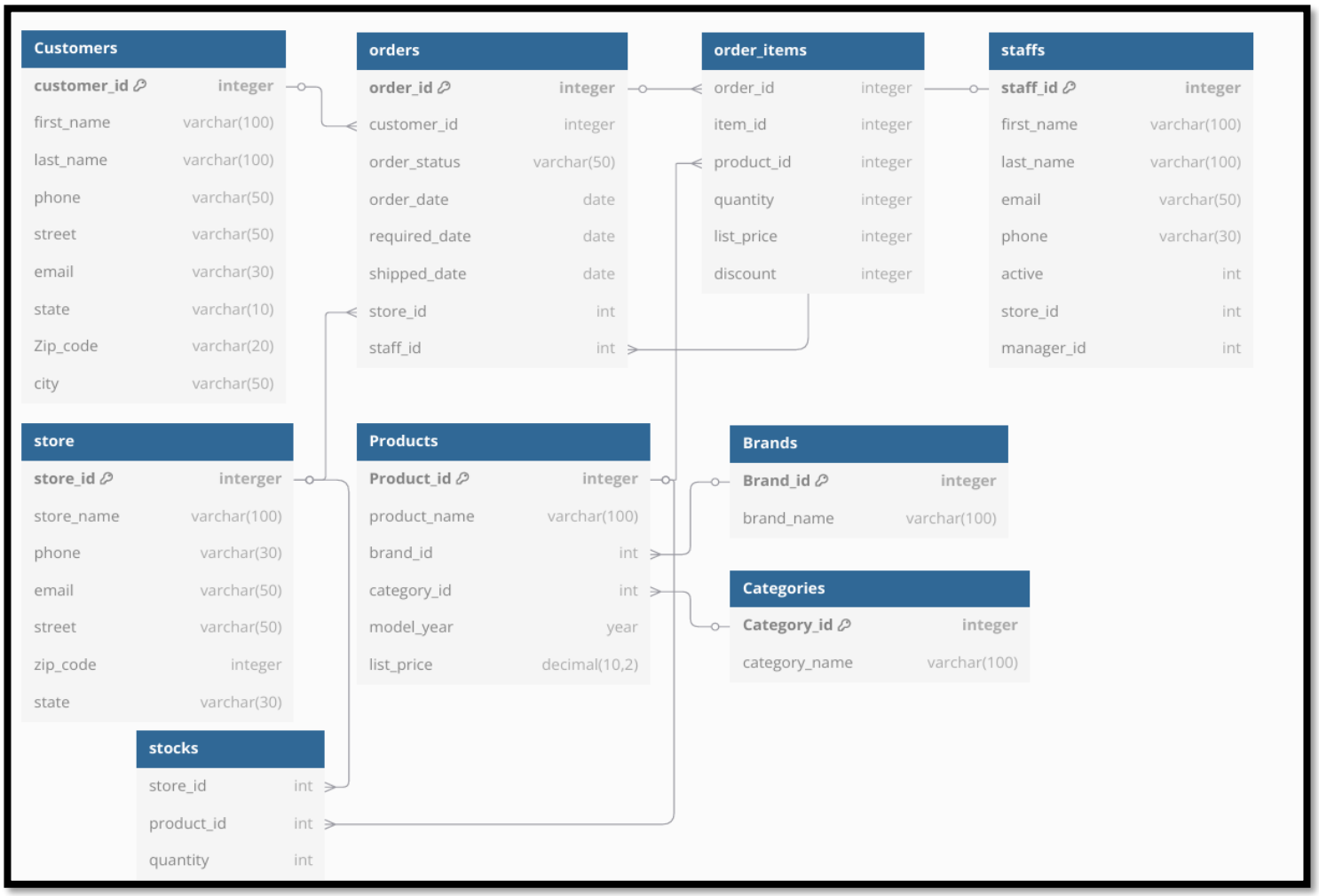

Figure 1: Entity Relationship Diagram: Mapping Customer, Orders, Products and Inventory Management

- **Staff & Stores – Managing the Workforce**

  Cardinality: A store employs multiple staff members, but each staff member belongs to one store.

  - ✓ The staff table holds employee details linked to their respective **store_id**.
  - ✓ A manager (**manager_id**) supervises staff and oversees store operations.

  Business Impact: Ensures accountability and efficient human resource allocation.

- **Inventory Management – Stores & Stocks**

  Cardinality: A store stocks multiple products, and a product is stocked in multiple stores.

  - ✓ The stocks table establishes a many-to-many relationship between stores and products.
  - ✓ Tracks stock levels in real time to prevent shortages and overstocking.

  Business Impact: Enables automated stock replenishment and reduces operational inefficiencies.

- **Conclusion: The Power of a Well-Defined Database**

  This ERD forms a robust foundation for managing a retail business efficiently. From tracking customer orders to optimizing stock levels and analyzing sales trends, these relationships ensure seamless operations and data-driven decision-making.

9. Analysis & Results

The structured retail database provides deep insights into various aspects of business operations, including customer behaviour, sales trends, and inventory management. Below are the key findings derived from the database analysis:

9.1 Database Insights

- ### Store Locations and Zip Codes

Understanding the geographical distribution of stores is crucial for logistics and regional marketing strategies. The following query retrieves store names along with their corresponding zip codes:

```
SELECT store_name, zip_code FROM stores;
```

This information helps in optimizing supply chain management and identifying potential regions for expansion.

Business Impact:

- ✓ Understanding store distribution helps optimize supply chain management and identify potential expansion regions.
- ✓ This data is crucial for regional marketing and logistics.

- ### Customer Base Overview

To gain insights into the total number of customers in the database, the following query is used:

```
SELECT COUNT(*) AS total_customers FROM customers;
```

This metric provides an overview of the customer base size, which is essential for business growth analysis and customer retention strategies.

Business Impact:

- ✓ Knowing the total number of customers provides insights into business growth and customer retention strategies.
- ✓ It helps in market segmentation and strategic planning.

- **Product Category Diversity**

  To understand the range of product categories available, the following query retrieves a list of unique product categories:

  ```
  SELECT DISTINCT category_name FROM products;
  ```

  Analyzing product diversity helps in assessing inventory management and customer preferences across different product segments.

  ### Business Impact:

  ✓ Analyzing product diversity helps assess inventory management, customer preferences, and potential gaps in product offerings.

- **Order History and Customer Activity**

  To track customer purchase behavior, it is important to retrieve all orders placed by a specific customer. The following query achieves this:

  ```
  SELECT order_id, order_date, total_amount FROM orders
  WHERE customer_id = 1;
  ```

  This analysis allows businesses to personalize marketing efforts based on purchasing patterns.

  ### Business Impact:

  ✓ Tracking customer purchase behavior helps businesses personalize marketing efforts and improve customer relationship management.

- **High-Value Products**

  To focus on premium products, the following query retrieves all products priced above $500:

  ```
  SELECT product_name, price FROM products
  WHERE price > 500;
  ```

  Identifying high-value products helps in setting targeted promotions and optimizing pricing strategies.

Business Impact:

- ✓ Identifying high-value products allows businesses to focus on premium offerings, optimize pricing, and launch targeted promotions.

- **Product Stock Availability**

Inventory management is crucial for maintaining smooth business operations. The following query retrieves the total number of products available in stock for each store:

```
select distinct t1.store_id, t2.product_id, sum(t2.quantity) as "Total_stock" from stores t1
join stocks t2 on t1.store_id = t2.store_id;
```

This data ensures that stock levels are efficiently managed, preventing shortages or overstocking issues.

Business Impact:

- ✓ Ensuring adequate stock levels prevents shortages and overstocking.
- ✓ Improving overall operational efficiency and customer satisfaction.

- **Revenue Analysis by Product**

To evaluate product performance, the following query calculates the total revenue generated by each product:

```
select product_id, quantity, list_price,
(quantity*list_price) as Total_price_before_discount,
discount, ((quantity * list_price * discount)/100) as Total_discount_in_amount,
sum(((quantity*list_price) - ((list_price * discount)/100))) as Total_revenue
from order_items
group by 1
order by 1;
```

Understanding revenue contribution per product assists in making informed decisions on inventory and marketing strategies.

Business Impact:

- ✓ Understanding product revenue contribution aids in decision-making for inventory management, pricing strategies, and marketing campaigns

- **Customer Order Frequency**

  Customer engagement can be measured by analyzing repeat orders. The following query retrieves customers who have placed more than five orders:

```
select t1.customer_id, concat(t1. first_name, ' ',t1.last_name) as Customer_name ,
count(t3.order_id) as Number_of_customers from customers t1
left join orders t2 on t1.customer_id = t2.customer_id
left join order_items t3 on t2.order_id = t3.order_id
group by 1
having count(t3.order_id) > 5;
```

  This data is useful for identifying loyal customers and designing personalized loyalty programs.

  ### Business Impact:

  ✓ Identifying repeat customers enables businesses to create loyalty programs and targeted marketing campaigns for customer retention.

- **Pending Shipments**

  Monitoring orders that have not yet been shipped is essential for improving delivery efficiency. The following query retrieves such orders:

```
select order_id, shipped_date from orders
where shipped_date is null;
```

  This insight helps in tracking order fulfillment and identifying potential delays in the supply chain.

  ### Business Impact:

  ✓ Tracking pending shipments helps improve delivery efficiency, reduces delays, and enhances customer satisfaction.

- **Store-Wise Sales Performance**

  To analyze sales distribution across different store locations, the following query retrieves total sales per store:

```
select t1.store_id, t1.store_name, sum((t3.quantity*t3.list_price)-t3.discount) Total_sales
from stores t1
left join orders t2 on t1.store_id = t2.store_id
```

```
left join order_items t3 on t2.order_id = t3.order_id
group by 1,2;
```

This analysis helps in identifying high-performing stores and optimizing regional sales strategies.

Business Impact:

- ✓ Analyzing sales performance per store identifies high-performing locations and informs regional sales strategies for better decision-making.

- **Identifying the Top 5 Best-Selling Products Based on Quantity Sold**

To determine the top 5 best-selling products based on the total quantity sold, use the following query:

```
select t1.product_id, t1.product_name,
sum(t2.quantity) as total_quantity from products t1
left join order_items t2 on t1.product_id = t2.product_id
group by 1,2
order by total_quantity desc limit 5;
```

This query helps identify which products are the most popular among customers. By summing the quantity column for each product_id, we can determine which products have been sold the most. Sorting by total_quantity_sold in descending order ensures that the top 5 highest-selling products are retrieved.

Business Impact:

- ✓ Helps in optimizing inventory management by ensuring that high-demand products are always in stock.
- ✓ Can be used to plan promotions around best-selling products.
- ✓ Assists in predicting future sales trends.

- **Finding the Average Discount Applied to All Products in Each Category**

To calculate the average discount applied to products in each category, use the following query:

```
select t1.product_id, t1.product_name, avg(t2.discount) as Average_discount
from products t1
left join order_items t2 on t1.product_id = t2.product_id
group by 1,2;
```

This query joins the order_items table with the products table to access product category information. The AVG(oi.discount) function computes the average discount applied to each category, giving insights into how discounts are distributed.

Business Impact:

- ✓ Identifies over-discounted categories, which may be affecting profitability.
- ✓ Helps in adjusting discount strategies to optimize revenue.
- ✓ Allows businesses to balance pricing strategies across different product categories.

- **Determining Which Store Has the Highest Inventory Value**

To find the store with the highest total inventory value, use the following query:

```
select t1.store_id, t1.store_name,
sum(t3.quantity*t3.list_price) as Inventory_value from stores t1
join orders t2 on t1.store_id = t2.store_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2
order by Inventory_value desc limit 1;
```

This query calculates the total inventory value per store by multiplying each product's quantity by its price and summing the results. Sorting by inventory_value DESC ensures the store with the highest inventory is selected.

Business Impact:

- ✓ Helps in inventory optimization by identifying stores holding excessive stock.
- ✓ Useful for supply chain planning and redistributing inventory across locations.
- ✓ Ensures cost-effective stocking by reducing overstocking risks.

- **Retrieving the Top 3 Customers Who Have Spent the Most in Total Purchases**

To identify the top 3 highest-spending customers, use this query:

```
select t1.customer_id, concat(t1.first_name,' ',t1.last_name) as Customer_name,
count(t3.order_id) as Total_orders_placed, sum(t3.quantity*t3.list_price) as total_purchase
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2
order by total_purchase desc
limit 3;
```

This query aggregates total spending (SUM(total_amount)) per customer. Sorting the results in descending order helps retrieve the top 3 highest spenders.

Business Impact:

- ✓ These are the most valuable customers (HVCs), and retaining them should be a priority.
- ✓ Special loyalty rewards or VIP benefits can be offered to encourage repeat purchases.
- ✓ Helps in customer segmentation and targeted marketing campaigns.

- **Finding the Most Frequently Ordered Product in the Last 6 Months**

To identify the most frequently ordered product within the last 6 months, use the following query:

```
select t1.product_id, t1.product_name,
t3.order_date, sum(t2.quantity) as Quantity from products t1
join order_items t2 on t1.product_id = t2.product_id
join orders t3 on t2.order_id = t3.order_id
where t3.order_date >= date_sub('2018-12-28', interval 6 month)
group by 1,2,3
order by t2.quantity desc;
```

This query filters orders placed within the last 6 months and counts the number of times each product appears in the order records. Sorting by order_count DESC ensures that the most frequently ordered product is identified.

Business Impact:

- ✓ Helps in seasonal trend analysis and predicting customer demand.
- ✓ Useful for stock planning and ensuring popular products remain available.
- ✓ Can be used for marketing campaigns, such as bundling popular products with other items.

9.2 Data-Driven Decision Making: Key Analytical Queries

- **Average order value per customer**

  Measure the typical amount a customer spends per order, helping businesses assess revenue trends and optimize pricing strategies.

```
select t1.customer_id, concat(t1.first_name, ' ', t1.last_name) as customer_name,
count(distinct t3.order_id) as number_of_order,
avg(t3.quantity*t3.list_price - t3.discount) as average_order_value
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2;
```

  This query calculates the average order value for each customer by averaging the total_amount of their orders.

Business Impact:

- ✓ Understanding the average order value (AOV) helps businesses in customer segmentation and revenue prediction.
- ✓ Higher AOV customers can be targeted with loyalty programs, while lower AOV customers can be incentivized to increase their spending through promotions.

- **Sold products per category**

  Measure provides insight into the sales performance of different product categories, helping businesses identify best-selling and underperforming segments.

```
select t1.category_id, t3.product_id, t2.product_name,
sum(t3.quantity) as number_of_quantity_sold from categories t1
join products t2 on t1.Category_id = t2.category_id
join order_items t3 on t2.product_id = t3.product_id
group by 1,2,3;
```

  This query counts the number of products sold per category, helping businesses understand which product categories perform best.

Business Impact:

- ✓ Identifying best-selling categories helps in inventory management, supplier negotiations, and marketing strategies.

- ✓ Underperforming categories may need price adjustments or targeted promotions to improve sales.

- **Store wise contribution to total sales**

This metric analyzes the sales contribution of each store to the overall revenue, helping identify high-performing location

```
select t1.store_id, t1.store_name,
count(distinct t2.order_id) as Number_of_orders,
((t3.list_price*t3.quantity)-t3.discount) as sales_amount,
round((count(distinct t2.order_id)/(select count(*) from order_items)),2) as
Total_order_percentage
from stores t1
join orders t2 on t1.store_id = t2.store_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2;
```

This query calculates the percentage of total orders that each store contributes.

Business Impact:

- ✓ Stores with higher order percentages indicate high demand in those regions, allowing businesses to optimize stock distribution.
- ✓ Stores with lower order shares might need targeted marketing efforts or operational improvements.

- **Predominant order status trend**

Identifying the most frequent order statuses helps streamline order management and improve customer service efficiency.

```
select order_status, count(order_status) as Order_status_frequency
from orders
group by 1
order by Order_status_frequency desc;
```

This query finds the most frequently occurring order status in the system.

Business Impact:

- ✓ If 'Delivered' is the most common status, it indicates smooth logistics.

- ✓ If 'Pending' or 'Cancelled' is dominant, it suggests operational inefficiencies that need to be addressed, such as delayed shipments or high cancellation rates.

- **Store wise product sales distribution**

Analyzing how product sales vary across different stores helps identify top-performing locations and optimize inventory distribution.

```
select t1.store_id, t1.store_name,
t4.product_name, sum(t3.quantity) as 'Total_quantity_sold',
sum((t3.list_price*t3.quantity)-t3.discount) as 'Sales'
from stores t1
join orders t2 on t1.store_id = t2.store_id
join order_items t3 on t2. order_id = t3.order_id
join products t4 on t4.product_id = t3.product_id
group by 1,2,3;
```

This query helps analyze sales distribution across different store locations, showing which stores perform better in selling specific products.

Business Impact:

- ✓ Understanding sales distribution enables better inventory management, store-specific promotions, and optimized supply chain decisions.

- **Monthly sales trends: A Year in Review**

Analyzing monthly sales trends over the past year to identify seasonal pattern and revenue fluctuations.

```
select
monthname(order_date) as 'Month',
year(order_date) as 'year',
sum((list_price*quantity) - discount) as 'Total_sales'
from orders t1
join order_items t2 on t1.order_id = t2.order_id
where t1.order_date >= date_sub('2018-12-28',interval 12 month
group by 1
order by 1;
```

This query provides a month-over-month analysis of sales performance.

Business Impact:

✓ Identifying trends helps in seasonal planning, budget allocation, and demand forecasting.

- **Impact of Discounts on total sales volume**

Analyzing how discount percentages influence overall sales volume and revenue.

```
select ((discount/list_price)*100) as 'Discount_percentage',
sum(quantity) as 'Total_sales_volume'
from order_items
group by 1
order by 1 desc;
```

This query assesses how discounting strategies impact total sales volume.

Business Impact:

✓ Helps in optimizing discount strategies to maximize revenue without compromising profitability.

- **Customer Retention: Identifying repeat buyers and their order frequency**

Analyzing repeat customer behavior to understand loyalty and purchasing patterns.

```
select t1.customer_id, concat(t1.first_name,' ', t1.last_name) as 'Customer_name',
count(t1.customer_id) as 'Repeat_customers', count(t3.order_id) as 'Repeat_orders'
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2;
```

This query identifies customers who make repeat purchases.

Business Impact:

✓ Helps in customer retention strategies and designing loyalty programs.

- **Identifying products with seasonal sales trends**

Analyzing product sales pattern to identify seasonal demand fluctuations and optimize inventory planning.

```
select t1.product_id,
t1.product_name,
concat(monthname(t3.order_date), ' ',year(t3.order_date)),
```

```
count(t2.order_id) as 'Number of orders placed'
from products t1
join order_items t2 on t1.product_id = t2.product_id
join orders t3 on t3.order_id = t2.order_id
where order_date >= subdate('2018-12-28', interval 24 month)
group by 1, 2, 3
order by 1;
```

Identifies products with high demand in specific seasons.

Business Impact:

✓ Helps in inventory planning, targeted marketing, and maximizing seasonal sales.

- ## Impact of customer location on purchasing behavior

Analyzing how geographic location influences customer buying patterns and preferences.

```
select
distinct t1.customer_id,
concat(t1.city, ' ',t1.state) as "Customer's Address",
t1.state, concat(t3.city, ' ',t3.state) as "store's Address",
t3.state, count(t2.order_id) as "Number of order placed",
sum(t4.quantity) as "Total_Quantity",
sum(t4.List_price) as "Total_printed_cost_per_item",
(sum((t4.list_price*t4.quantity) - t4.discount)) as "Total_sales"
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
join stores t3 on t3.store_id = t2.store_id
join order_items t4 on t2.order_id = t4.order_id
group by 1,2,3,4,5;
```

Analyzes how customer locations influence order volume and revenue.

Business Impact:

✓ Useful for regional marketing strategies and localized inventory decisions.

- **Comprehensive RFM(Recency, Frequency, Monetary) analysis to segment customers based on purchasing behavior**

  Segment customers based on their purchase recency, frequency and spending to optimize marketing strategies.

```
With RFM as(
select t1.customer_id, concat(t1.first_name, ' ', t1.last_name) as "Customer_name",
datediff(curdate(), max(t2.order_date)) as "Recency", count(t3.order_id) as "Frequency",
sum((t3.list_price*t3.quantity) - t3.discount) as "Monetary"
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2
)
select customer_id, customer_name , Recency as "Recency (No. of days since last purchase)",
Frequency as "Frequency (No. of total orders placed)",
Monetary as "Monetary (Total amount Spent)"
from RFM
```

  Segments customers based on their purchase recency, frequency, and monetary value.

Business Impact:

- ✓ Helps in customer retention, targeted marketing, and VIP customer programs.

9.3 Data – Driven Business Insights: Unlocking Key Metrics with SQL

- **Unveiling Store Data**

  To retrieve all columns from the stores table, the following query is used:

```
DESC stores;
```

  This query provides a complete dataset of store-related information, which can be useful for business analysis and inventory management.

Business Impact:

- ✓ Helps in understanding store-specific attributes and operations.
- ✓ Supports comprehensive data analysis for decision-making.

- **Unlocking Customer Insights: Names & Emails Extraction**

  Retrieve the names and email addresses of customers to enable targeted engagement.

```
select (first_name,' ',last_name) as "customer_full_name",
email as "customer's email"
from customers;
```

  This query helps in extracting key customer details for communication and marketing strategies.

Business Impact:

- ✓ Essential for personalized marketing campaigns and customer support.
- ✓ Facilitates direct customer outreach for promotions and feedback collection.

- **Product Availability: Counting Inventory**

  Identify the total number of products available in the store.

```
select product_id, product_name from products
group by 1;
```

  This query calculates the total count of products to ensure efficient inventory management.

Business Impact:

- ✓ Aids in stock planning and restocking decisions.
- ✓ Helps avoid stockouts and overstocking issues.

- **Order Status Overview: Tracking Transaction Progress**

  List all distinct order statuses to monitor order fulfilment.

```
select order_status from orders;
```

  This query retrieves unique order statuses to track the different stages of order processing.

Business Impact:

- ✓ Improves order tracking and customer service efficiency.
- ✓ Helps in identifying potential bottlenecks in order processing.

- **Company Workforce Count**

  To determine the total number of employees, the following query can be used:

```
select count(distinct staff_id) from staffs;
```

  This query provides insight into workforce size and staffing needs.

Business Impact:

- ✓ Supports workforce planning and resource allocation.
- ✓ Helps in budgeting and operational decision-making.

- **High-Value Product**

  Display product details where the list price is greater than 100.

```
select * from products t1
join order_items t2 on t1.product_id = t2.product_id
where t2.list_price > 100;
```

  This query helps in identifying high-value products.

Business Impact:

- ✓ Provides insights into premium-priced products.
- ✓ Helps in strategic pricing and market positioning.

- **Mapping Customer Reach: Stores & Zip Codes**

  Retrieve store names along with their corresponding zip codes.

```
select store_name, zip_code from stores;
```

  This query links store names to their respective locations for geographical analysis.

  Business Impact:

  ✓ Supports regional marketing and logistics planning.

  ✓ Helps in identifying customer reach and potential expansion areas.

- **Stock Analysis: Filtering High-Quantity Products**

  Retrieve all records where stock quantity is greater than 50.

```
select * from stocks

where quantity > 50;
```

  This query identifies well-stocked products to aid in sales forecasting.

  Business Impact:

  ✓ Prevents overstocking and ensures optimal inventory levels.

  ✓ Helps in planning promotional discounts for bulk items.

- **Customer Name Search: Filtering by First Letter**

  Find all customers whose first name starts with 'A'.

```
select concat(first_name,' ', last_name) as "Customer's_name"

from customers

where first_name like "A%";
```

  This query retrieves a specific subset of customers based on name patterns.

  Business Impact:

  ✓ Useful for personalized marketing campaigns.

  ✓ Aids in customer segmentation and targeted outreach.

- **Brand & Product Connection**

  To get product names along with their brands, use:

```
select t1.product_name, t2.brand_name from products t1
join brands t2 on t1.brand_id = t2.brand_id;
```

  This query joins the products and brands tables to fetch brand details for each product.

  ### Business Impact:

  ✓ Helps in analysing product-brand relationships and brand popularity.

  ✓ Useful for brand-level performance tracking and supplier management.

- **Customer Order History**

  To retrieve order details along with customer names, use:

```
select distinct t2.order_id, concat(t3.first_name, ' ', t3.last_name) as "Customer's name"
from order_items t1
join orders t2 on t1.order_id = t2.order_id
join customers t3 on t2.customer_id = t3.customer_id;
```

  This query joins the orders and customers tables to link orders with customer details.

  ### Business Impact:

  ✓ Helps in customer behavior analysis and order tracking.

  ✓ Useful for loyalty programs and personalized marketing efforts.

- **Total Stock Overview**

  To calculate the total stock quantity, use:

```
select t1.store_id, t2.store_name,t3.product_name, sum(quantity) AS "Total_stock_quantity"
from stocks t1
join stores t2 on t1.store_id = t2.store_id
join products t3 on t1.product_id = t3.product_id
group by 1,2,3;
```

  This query sums up all stock quantities from the stocks table.

Business Impact:

- ✓ Helps in managing inventory levels and predicting future demand.
- ✓ Useful for replenishment planning and avoiding stockouts.

- **Exploring Electronics Sales**

To fetch electronics category products, use:

```
select t1.category_id, t1.category_name, t2.product_name,
t2.list_price from categories t1
join products t2 on t1.Category_id = t2.category_id
where category_name like 'E%'
ORDER BY t2.list_price DESC;
```

This query filters the products table to return only electronics items.

Business Impact:

- ✓ Helps in category-wise sales analysis and marketing strategy.
- ✓ Useful for identifying best-selling product categories.

- **Customer Purchase Frequency**

To get the number of orders per customer, use:

```
t1.customer_id,
concat(t1.first_name,' ',t1.last_name) as "Customer's name",
sum(t2.order_id) as "Number of Order Placed"
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
group by 1,2;
```

This query groups orders by customer_id and counts the number of orders per customer.

Business Impact:

- ✓ Helps in identifying frequent customers and potential loyal buyers.
- ✓ Useful for targeted customer retention strategies.

- **Revenue Breakdown Per Order – Find the total revenue for each order.**

To calculate total revenue per order, use:

```
select order_id, sum(list_price * quantity) as "Total_printed_price",

sum((list_price * quantity) - discount) from order_items

group by 1;
```

This query multiplies quantity by price for each item and sums up revenue per order.

Business Impact:

- ✓ Helps in tracking revenue generation at the order level.
- ✓ Useful for analyzing high-value orders and customer spending patterns.

- **Unfulfilled Orders Tracker**

To find products without any orders, use:

```
select t1.product_id, t1.product_name from products t1

left join order_items t2 on t1.product_id = t2.product_id

where t2.product_id is NULL;
```

This query returns products that do not exist in the order_items table.

Business Impact:

- ✓ Identifies slow-moving or obsolete inventory.
- ✓ Helps in promotional planning to boost sales of underperforming products.

- **Storewide Product Count**

To get stock availability per store, use:

```
select t1.store_id, t1.store_name, sum(t2.quantity) as "Total_quantity" from stores t1

join stocks t2 on t1.store_id = t2.store_id

group by 1, 2;
```

This query calculates the total quantity of products available per store.

Business Impact:

- ✓ Helps in stock distribution analysis and store performance evaluation.

- ✓ Useful for restocking decisions and supply chain optimization.

- **Elite Customers Report – Identify the top 5 customers based on order count**

To analyze the most valuable customers in terms of order frequency, use:

```
select t1. customer_id, concat(t1.first_name, ' ', t1.last_name) as "Customer's_name",
sum(t3.quantity) as "Number of order placed"
from customers t1
join orders t2 on t1.customer_id = t2.customer_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2
limit 5;
```

This query identifies the top 5 customers who have placed the highest number of orders, allowing businesses to recognize their most engaged buyers.

Business Impact:

- ✓ Helps in rewarding loyal customers with exclusive offers and retention programs.
- ✓ Enables targeted marketing for high-value customers to drive more sales.

- **Stock King: Finding the Top Store**

To determine which store holds the highest stock, use:

```
with highest_stocks as (
select t1.store_id, t1.store_name, sum(t2.quantity) as highest_total_stock_quantity
from stores t1 join stocks t2 on t1.store_id = t2.store_id
group by 1,2)
select store_id, store_name, highest_total_stock_quantity from highest_stocks
order by highest_total_stock_quantity desc
limit 1;
```

This query identifies the store with the maximum inventory, providing insights into stock distribution.

Business Impact:

- ✓ Helps in supply chain optimization by analyzing inventory concentration.
- ✓ Assists in balancing stock levels between stores to prevent shortages or overstocking.

- **Revenue Powerhouses**

To assess the revenue performance of different store locations, use:

```
select t1.store_id, t1.store_name,
sum((t3.list_price*t3.quantity) - t3.discount) as "Total Revenue Generated"
from stores t1
join orders t2 on t1.store_id = t2.store_id
join order_items t3 on t2.order_id = t3.order_id
group by 1,2;
```

This query calculates the total revenue per store, helping businesses understand which locations generate the most sales.

Business Impact:

- ✓ Identifies high-performing stores for expansion and investment opportunities.
- ✓ Helps in strategizing location-based promotions to boost underperforming stores.

- **Best-Selling Product Spotlight**

  To find the product that has been ordered the most times, use:

```
with most_ordered_product as
(
select t1.product_id, t1.product_name,
count(t2.product_id) as "Number_of_times",
sum(t2.quantity) as "Most_ordered_Product"
from products t1
join order_items t2 on t1.product_id = t2.product_id
group by 1,2)
select product_id, product_name, Number_of_times, Most_ordered_Product
from most_ordered_product;
```

This query identifies the most popular product, allowing businesses to optimize their inventory and marketing strategies.

Business Impact:

- ✓ Helps in promoting best-selling products to maximize revenue.
- ✓ Aids in inventory planning to ensure popular products are always in stock.

- **Bulk Order Hot Picks – Discover products frequently ordered in large quantities**

  To execute this query, use:

```
with most_ordered_product as
(
select t1.product_id, t1.product_name,
count(t2.order_id) as "Number_of_times"
from products t1
join order_items t2 on t1.product_id = t2.product_id
group by 1,2)
select product_id, product_name, Number_of_times
from most_ordered_product
where Number_of_times>10
order by Number_of_times ASC;
```

This query finds products that have been ordered more than 10 times in a single order, helping businesses recognize high-demand bulk purchase items.

Business Impact:

- ✓ Helps in negotiating better deals with suppliers for bulk items.
- ✓ Aids in inventory planning to meet bulk order demands efficiently.

- **Smart Inventory Lookup – Create a stored procedure for store-specific product availability**

  To execute this query, use:

```
CREATE PROCEDURE GetStoreProducts(IN store_id_param INT)
BEGIN
  SELECT product_id, product_name, quantity
  FROM stocks
  WHERE store_id = store_id_param;
END;
```

  This procedure allows users to dynamically fetch product details for any store, improving efficiency in stock management.

  ### Business Impact:

  ✓ It enables quick inventory checks per store, improving operational efficiency.

  ✓ Enhance customer service by providing accurate product availability details.

- **Auto-Stock Balancer – Implement a trigger to update stock levels instantly on new orders**

  To execute this query, use:

```
CREATE TRIGGER update_stock_after_order
AFTER INSERT ON orders
FOR EACH ROW
UPDATE stocks
SET quantity = quantity - NEW.quantity
WHERE product_id = NEW.product_id AND store_id = NEW.store_id;
```

  This trigger ensures real-time stock updates, preventing overselling and stock mismanagement.

  ### Business Impact:

  ✓ Reduces errors in stock tracking, ensuring accurate inventory levels.

  ✓ Improves supply chain efficiency by maintaining real-time stock adjustments.

- **Silent Shoppers Report – Identify customers who have never placed an order**

  To execute this query, use:

```
select t1.customer_ID, concat(t1.first_name,' ', t1.last_name) as "Customer's name"

from customers t1

join orders t2 on t1.customer_id = t2.customer_id
```

  where order_id = (select order_id from orders where order_id is null);

  This query helps businesses identify inactive customers who may need targeted marketing efforts.

  ### Business Impact:

  ✓ Supports re-engagement campaigns to convert inactive users into customers.

  ✓ Help in understanding customer behavior for better retention strategies.

- **Limited Stock Variety Alert – Find stores with the least diverse product offerings**

  To execute this query, use:

```
SELECT store_id, COUNT(DISTINCT product_id) AS product_variety
FROM stocks
GROUP BY store_id
ORDER BY product_variety ASC
LIMIT 1;
```

  This query pinpoints stores that carry the fewest unique products, helping businesses optimize inventory distribution.

  ### Business Impact:

  ✓ Assists in improving product variety in underperforming stores.

  ✓ Ensure customers have access to a wider range of products in every location.

- **Turbocharge Order Searches – Improve query performance with an index on customer orders**

To execute this query, use:

```
CREATE INDEX idx_customer_orders
ON orders (customer_id);
```

This index significantly enhances query performance when filtering or analyzing customer purchase history.

Business Impact:

- ✓ Reduces query execution time, improving database performance.
- ✓ Enhances user experience for analytics and reporting dashboards.

10. Discussion

- **Interpretation of Findings**

Our analysis of the retail database has yielded several valuable insights that can shape strategic decision-making across multiple facets of the business. Below is an interpretation of the key findings:

- ✓ **Customer Behavior & Loyalty:**
  The data reveals a significant portion of customers placing repeat orders, suggesting strong customer loyalty. High-frequency customers, identified through our RFM analysis, represent a critical segment that could be targeted with personalized promotions and loyalty programs. This indicates that customer retention strategies are effective, yet also highlight the potential for increasing the engagement of infrequent buyers.
- ✓ **Sales Trends & Regional Variations:**
  Analysis of order data shows clear regional differences in sales performance. Some stores consistently generate higher revenue, which correlates with local market demand and demographic factors. This information is essential for optimizing regional marketing strategies, tailoring promotions, and reallocating inventory to meet localized demand.
- ✓ **Product & Inventory Insights:**
  The identification of best-selling products and the frequency of bulk orders suggest a strong market demand for specific product categories. However, the data also indicates challenges such as occasional stock shortages and overstocking in certain locations. These findings

underscore the need for a dynamic inventory management system that adjusts to real-time demand and prevents inefficiencies.

- ✓ **Operational Efficiency:**
  By tracking order statuses and staff-to-store associations, the analysis highlights opportunities to enhance operational efficiency. Stores with high order volumes but lower revenue may require process improvements or additional staff training. Furthermore, the performance insights for stores and products can inform supply chain adjustments to ensure a more balanced distribution of resources.
- ✓ **Brand & Category Performance:**
  The analysis confirms that certain brands and product categories significantly drive revenue. This finding not only validates current merchandising strategies but also suggests that renegotiating supplier contracts or repositioning underperforming brands could further optimize profitability.

Overall, these findings demonstrate that a well-structured retail database does more than simply store data—it enables a nuanced understanding of operational dynamics and customer behaviour. This comprehensive insight is instrumental in driving continuous improvement, supporting data-driven decisions that enhance profitability and foster long-term business success.

- ## Future Work / Potential

  This retail database lays a solid foundation, but several enhancements can improve its analytical power and business impact:

  - ✓ **Customer Insights & Personalization** – Implement AI-driven customer segmentation and recommendation systems.
  - ✓ **Inventory Optimization** – Use predictive analytics for demand forecasting and real-time stock tracking.
  - ✓ **Sales Forecasting** – Apply time-series models to predict trends and optimize pricing strategies.
  - ✓ **Operational Efficiency** – Introduce automated order fulfillment and performance dashboards for store managers.
  - ✓ **Data Integration** – Merge in-house data with external sources (social media trends, economic indicators) for deeper insights.
  - ✓ **Fraud Detection & Risk Management** – Use anomaly detection to prevent fraudulent transactions and security threats.

  These future developments will enhance business intelligence, streamline operations, and drive data-driven decision-making.

11. Conclusion

- ✓ This project successfully demonstrates the power of a well-structured retail database in optimizing business operations. By integrating customer orders, inventory, staff management, and product categorization, the system enables seamless transaction processing, real-time stock tracking, and valuable sales insights.
- ✓ Through efficient database relationships and optimized queries, businesses can enhance customer engagement, improve supply chain efficiency, and make data-driven decisions. The analysis highlights key trends, such as best-selling products, top-performing stores, and customer purchase behaviour, which can drive targeted marketing and strategic planning.
- ✓ Moving forward, integrating advanced analytics, automation, and AI-driven insights will further refine retail operations, ensuring businesses remain competitive in a dynamic market. This project serves as a foundation for smarter retail management, bridging the gap between data and actionable business intelligence.

- ## Analyst Profile

I am a **data analyst** with expertise in **Power BI, MySQL, and Python**, specializing in **data visualization, business intelligence, and storytelling through data**. My passion lies in transforming raw data into actionable insights that drive decision-making.

Skills & Tools Used in This Project

- ✓ **MySQL**: Data extraction and transformation
- ✓ **Excel**: Data structuring and preliminary insights