

San Jose State University

Department of Applied Science

One Washington Square, 95112



DATA 225: Sec-22 Database Systems for Analytics

Course Instructor: Ron Mak

Report

An Application for Interactive Query and Exploration of an Superstore Dataset

Team DATA PIRATES

Akshat Patel	Ashritha Kumari B	Dhrumil Shah	Maulik Jyani
016800121	016798860	016781934	016804086

TABLE OF CONTENTS

1. INTRODUCTION.....	01
2. OBJECTIVE.....	02
3. DATA SOURCE.....	02
4. OPERATIONAL DATABASE DESIGN.....	03
A. ER DIAGRAM.....	04
B. RELATIONAL SCHEMA.....	04
C. EER DIAGRAM.....	05
D. ER RELATIONSHIPS.....	05
E. ENTITIES AND ITS ATTRIBUTES.....	06
5. ETL TO OPERATIONAL DATABASE DESIGN.....	10
6. ANALYTICAL DATABASE DESIGN.....	11
A. STAR SCHEMA.....	13
7. ETL TO ANALYTICAL DATABASE DESIGN.....	13
8. USE CASES.....	15
A. OPERATIONAL DATABASE.....	15
B. ANALYTICAL DATABASE.....	19
9. FUTURE SCOPE	22
10. REFERENCES.....	22

INTRODUCTION

This project is centered around the development of a comprehensive and efficient database system for a superstore. The primary objective is to create a robust platform that enables customers to compare prices across different websites associated with the superstore. By leveraging this database system, customers gain access to real-time pricing information, allowing them to make well-informed purchasing decisions and find the best available deals.

The database structure is designed to store essential data such as customer demographics, product information, order details, shipment types, and pricing details from multiple websites. This ensures that the system is equipped with the necessary information to facilitate seamless price comparison.

The overarching goal is to enhance customer satisfaction by providing a user-friendly platform that promotes transparency in online shopping. Customers can easily search for products and view price comparisons across various websites, empowering them to identify the most favorable prices. By streamlining the process of comparing prices, the system simplifies the shopping experience and saves customers valuable time and effort.

Furthermore, the database system enables the analysis of pricing data to identify trends and optimize pricing strategies. This analysis provides valuable insights for decision-makers, allowing them to make informed decisions that maximize overall profitability. High-level and low-level dashboards are implemented to present comprehensive sales data, including overall sales figures, sales per product category, and monthly sales trends. These dashboards serve as powerful tools for decision-makers to monitor and analyze the performance of the superstore, enabling them to make data-driven decisions that drive growth and success.

In summary, this superstore database project aims to create a robust database system that empowers customers with the ability to compare prices, make informed purchasing decisions, and find the best deals available. It prioritizes customer satisfaction, transparency, and profitability, while providing decision-makers with valuable insights through comprehensive dashboards.

OBJECTIVE

The primary aim of this project is to develop a database system for a superstore with a focus on facilitating price comparison across multiple websites. The specific objectives of the project include:

1. Establishing a robust and efficient database structure capable of storing customer demographics, product information, order details, shipment types, and pricing data from various websites associated with the superstore.
2. Implementing a mechanism to retrieve real-time pricing information for specific products from different websites and enable a comparison of prices.
3. Empowering customers to make informed purchasing decisions by providing them with access to the best-priced products available across multiple websites.
4. Enhancing customer satisfaction by promoting transparency and enabling customers to find the best deals in a convenient and efficient manner.
5. Improving the overall shopping experience for customers by simplifying the process of comparing prices across multiple websites, thereby saving time and effort.
6. Analyzing pricing data to identify trends, optimize pricing strategies, and enhance the superstore's competitiveness in the market.
7. Developing high-level and low-level dashboards to assist decision-makers in improving overall profit and analyzing sales performance in each product category.

By accomplishing these objectives, the project aims to provide customers with a valuable tool to find the most favorable prices while ensuring that the superstore remains competitive in the online marketplace.

DATA SOURCE

The dataset utilized in this project was sourced from the Kaggle website [link here](#), from which , we have taken the important tables namely Product, Customer, Orders, Category, Ship Modes etc.

Rest of the data, we have generated it using [Mockaroo](#).

Created a project ,designed schemas, generated, and loaded the datasets in MOCKAROO.

Performed some ETL using Microsoft Excel, Python and finally loaded them into the database.

OPERATIONAL DATABASE DESIGN

The application is mainly divided into two parts, Operational and Analytical.

This Operation Part provides a user interface that enables customers to view products available in the superstore. Customers can search for products by name, making it easier to find specific items of interest. Additionally, they can filter products based on price range, website, and category, allowing for a more personalized shopping experience.

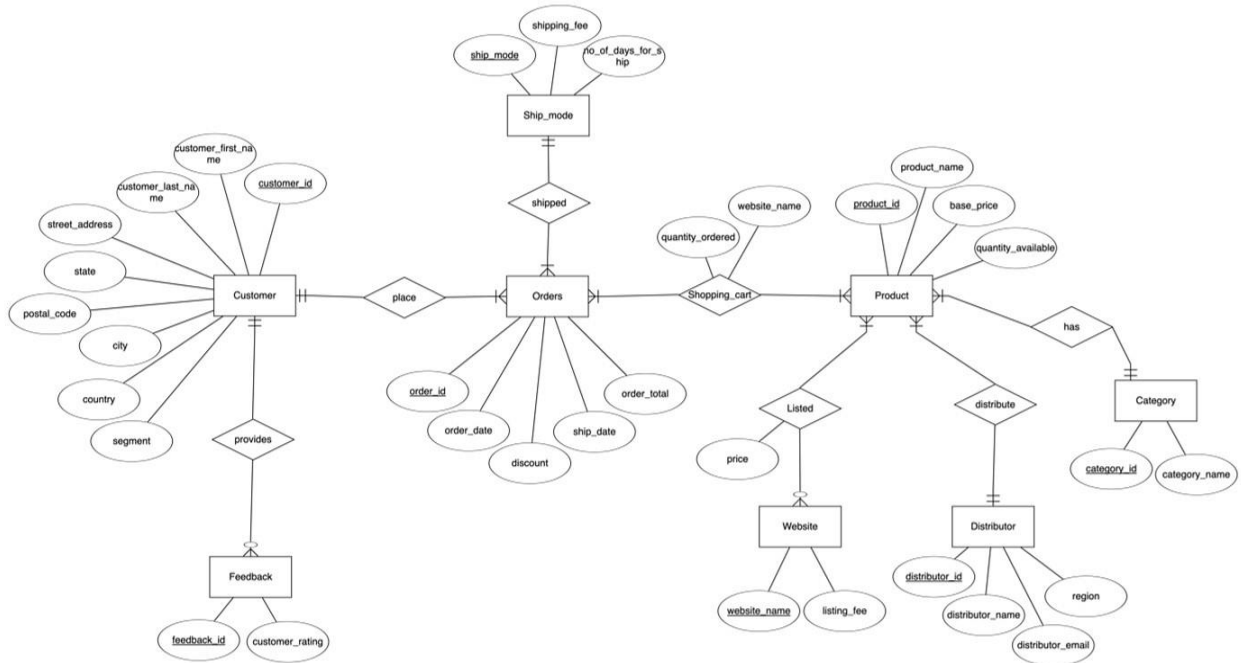
Customers have the functionality to add products to their cart, facilitating the selection and organization of desired items. Furthermore, they can remove products from the cart if they change their mind or no longer wish to purchase a particular item.

To proceed with the purchase, customers who are not registered users can easily register and create an account. Registered users can conveniently sign in to access their order history and track their purchases. This feature ensures a smooth and personalized shopping experience for customers.

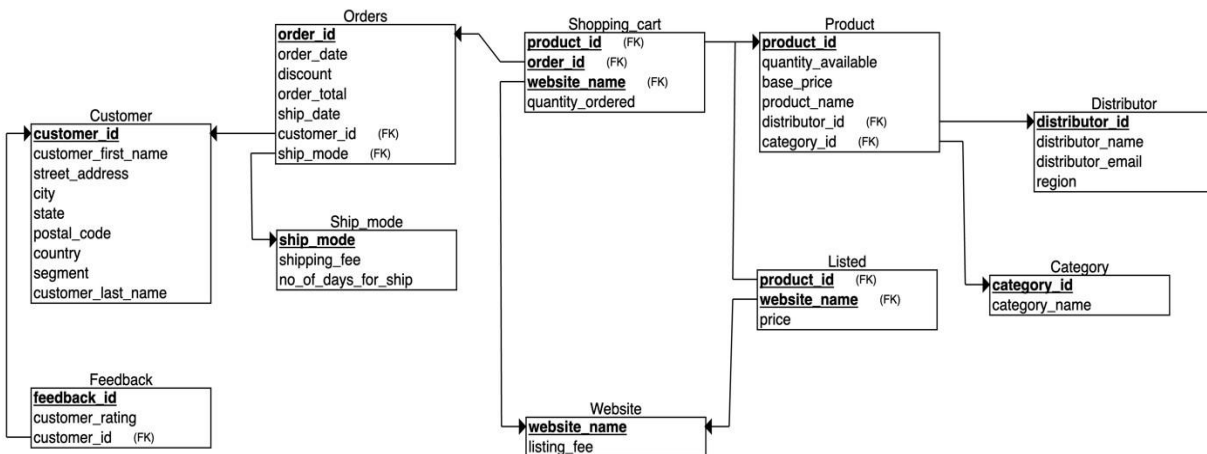
By incorporating these features, the project aims to provide customers with a user-friendly and efficient platform for exploring products, managing their shopping cart, and making purchases. The seamless registration and sign-in process enhances convenience and facilitates a more personalized shopping journey for customers.

Analytical part covers Analyzing the data and producing graphs. The relationship between each entity can be described by ER diagram which is a relationship diagram that explains 1:1 (ONE - ONE), 1:M (ONE - MANY), M: M (MANY - MANY)

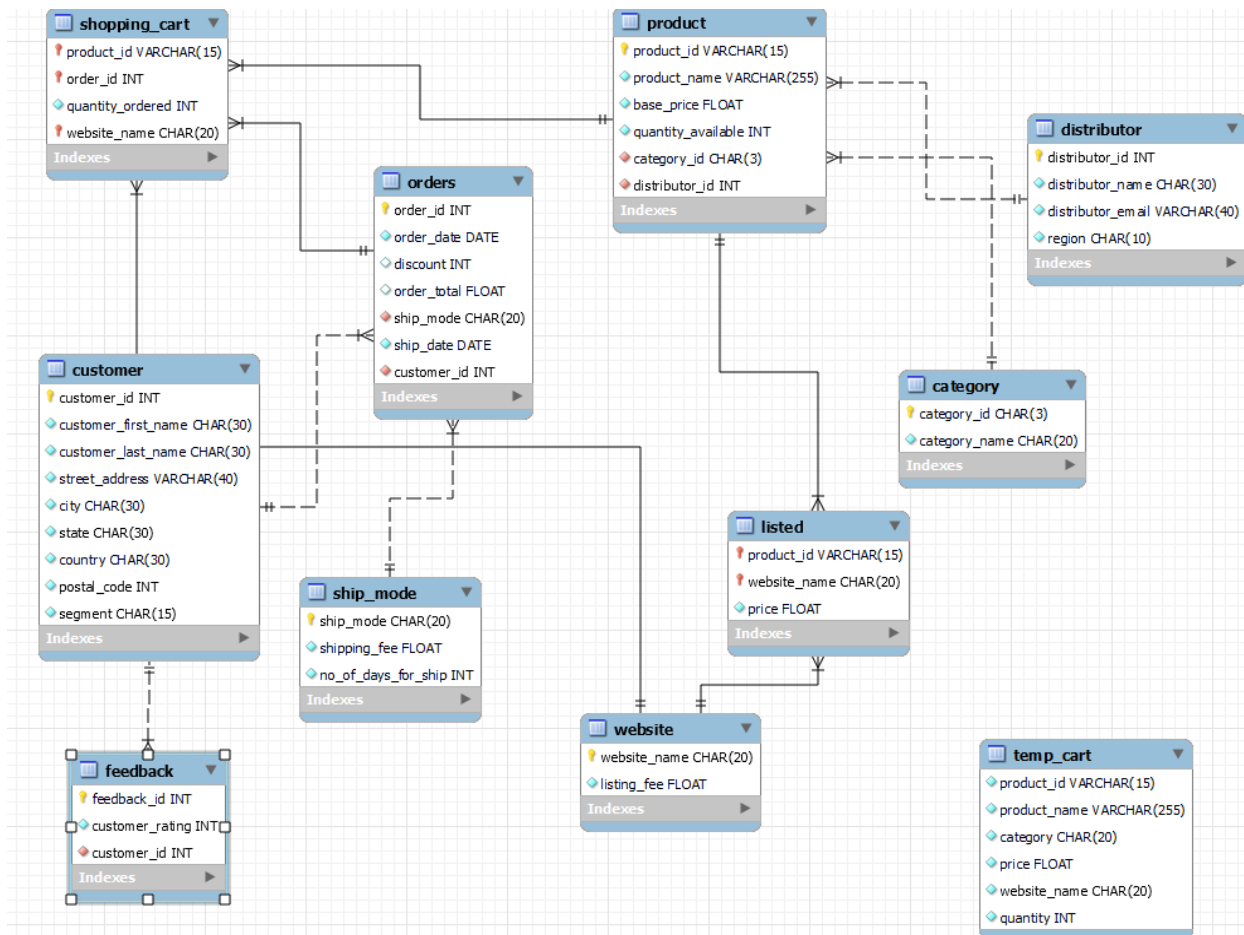
ER diagram



Relational Schema



EER diagram



ER Relationships

In the context of the superstore database, there are several entity relationships that exist among the tables. The relationships can be described as follows:

A. One-to-Many (1:N) Relationship:

- **Category to Product:** One category can have multiple products, but a product belongs to only one category.
- **Distributor to Product:** One distributor can supply multiple products, but a product is supplied by only one distributor.
- **Ship_Mode to Order:** One shipping mode can be associated with multiple orders, but an order is associated with only one shipping mode.
- **Customer to Order:** One customer can place multiple orders, but an order is placed by only one customer.

- **Product to Shopping_Cart:** One product can be added to multiple shopping carts, but a shopping cart contains only one product.
- **Website to Listed:** One website can have multiple listings, but a listing belongs to only one website.
- **Product to Temp_Cart:** One product can be added to multiple temporary carts, but a temporary cart contains only one product.

B. One-to-One (1:1) Relationship:

- **Category to Listed:** Each category can have multiple listings, but a listing belongs to only one category.

C. Many-to-Many (N:M) Relationship:

- **Product to Order:** A product can be present in multiple orders, and an order can contain multiple products. This relationship is typically realized through an intermediate table that represents the "line items" or details of each product within an order.

The Entities and their Attributes are mentioned below:

1. Category

- category_id:** This column serves as the primary key of the "category" table, uniquely identifying each category entry.
- category_name:** This column stores the name of the category, such as "Furniture," "Office supplies," or "Technology".

2. Customer

- customer_id:** This column serves as the primary key of the "customer" table, uniquely identifying each customer entry.
- customer_first_name:** This column stores the first name of the customer.
- customer_last_name:** This column stores the last name or surname of the customer.
- street_address:** This column stores the street address of the customer's residence or business location.
- city:** This column stores the name of the city where the customer resides or the business is located.
- state:** This column stores the name of the state or province where the customer's city is located.
- country:** This column stores the name of the country where the customer resides or the business is located.
- postal_code:** This column stores the postal or ZIP code associated with the customer's address.

- i) **segment:** This column stores the segment or category to which the customer belongs, such as "Home," "Consumer," "Corporate," or "individual."

3. Distributor

- a) **distributor_id:** This column serves as the primary key of the "distributor" table, uniquely identifying each distributor entry.
- b) **distributor_name:** This column stores the name of the distributor.
- c) **distributor_email:** This column stores the email address of the distributor.
- d) **region:** This column stores the region or geographical area associated with the distributor.

4. Feedback

- a) **feedback_id:** This column serves as the primary key of the "feedback" table, uniquely identifying each feedback entry.
- b) **customer_rating:** This column stores the rating given by the customer in their feedback. The rating can be on a scale, such as 1-5 stars.
- c) **customer_id:** This column stores the customer ID associated with the feedback.

5. Listed

- a) **product_id:** This column represents the product identifier and serves as a foreign key in the "listed" table.
- b) **website_name:** This column stores the name of the website where the product is listed. Examples of website names could include "Amazon", "Costco", "Ikea", "Target", "Walmart".
- c) **price:** This column stores the price of the product on the listed website.

6. Order

- a) **order_id:** This column serves as the primary key of the "order" table, uniquely identifying each order entry.
- b) **order_date:** This column stores the date when the order was placed.
- c) **discount:** This column stores the discount amount applied to the order, if any.
- d) **order_total:** This column stores the total price of the order. It represents the overall cost of the items purchased by the customer, including taxes, shipping fees, and any applicable discounts.
- e) **ship_mode:** This column stores the shipping mode or method chosen for delivering the order. It represents the selected shipping option, such as "first class shipping," "second class shipping," "same day shipping," or "standard class shipping."
- f) **ship_date:** This column stores the date when the order was shipped or is scheduled to be shipped.
- g) **customer_id:** This column stores the customer ID associated with the order.

7. Product

- a) **product_id**: This column serves as the primary key of the "product" table, uniquely identifying each product entry.
- b) **product_name**: This column stores the name of the product, such as "Books," "Chair," or "Desk Lamp."
- c) **base_price**: This column stores the base price of the product.
- d) **quantity_available**: This column stores the quantity of the product available in the store's inventory.
- e) **category_id**: This column represents the category identifier associated with the product. It serves as a foreign key.
- f) **distributor_id**: This column represents the distributor identifier associated with the product. It serves as a foreign key.

8. Ship_mode

- a) **ship_mode**: This column stores the name of the shipping mode or method, such as "first class shipping," "second class shipping," "same day shipping," or "standard class shipping." The ship_mode column allows customers to choose their preferred shipping option during the checkout process.
- b) **shipping_fee**: This column stores the cost or fee associated with the selected shipping mode. It represents the additional charge that customers need to pay for the chosen shipping option.
- c) **no_of_days_for_ship**: This column stores the estimated number of days required for the order to be shipped and delivered to the customer.

9. Shopping_cart

- a) **product_id**: This column represents the product identifier and serves as a foreign key in the "shopping_cart" table.
- b) **order_id**: This column serves as a foreign key referencing the order to which the items in the shopping cart belong.
- c) **quantity_ordered**: This column stores the quantity of a particular product that the customer has ordered.
- d) **website_name**: This column stores the name of the website where the customer added the product to the shopping cart.

10. Website

- a) **website_name**: This column stores the name of the website or online platform, such as "Amazon," "Costco," "IKEA," "Target," "Walmart," or any other website participating in the superstore's distribution network.
- b) **listing_fee**: This column stores the fee charged to the superstore for listing its products on the website.

11. Temp_cart (This table has been created only for our logic purpose)

- a) **product_id**: This column represents the identifier of the product added to the shopping cart.
- b) **product_name**: This column stores the name of the product added to the shopping cart.
- c) **category**: This column stores the category of the product added to the shopping cart, such as "Technology," "Furniture," or "Office Supplies."
- d) **price**: This column stores the price of the product added to the shopping cart. It represents the cost of the product per number of unit(of same kind) purchased.
- e) **website_name**: This column stores the name of the website or online platform from which the product was added to the shopping cart.
- f) **quantity_ordered**: This column stores the quantity of a particular product that the customer has ordered.

The "temp_cart" table serves as a temporary repository for storing data during the customer's shopping session. It holds information about the products selected by the customer, including their identification, names, categories, prices, associated websites, and the quantities ordered. This temporary storage allows the superstore to keep track of the customer's selections and facilitates the checkout process. Once the customer completes the purchase, the data from the temp_cart table is typically deleted, as it is no longer needed.

ETL to OPERATIONAL DATABASE

Customer Table Creation

```
1 sql = ( """
2     CREATE TABLE customer
3     (
4         customer_id INT NOT NULL AUTO_INCREMENT,
5         customer_first_name CHAR(30) NOT NULL,
6         customer_last_name CHAR(30) NOT NULL,
7         street_address VARCHAR(40) NOT NULL,
8         city CHAR(30) NOT NULL,
9         state CHAR(30) NOT NULL,
10        country CHAR(30) NOT NULL,
11        postal_code INT NOT NULL,
12        segment CHAR(15) NOT NULL,
13        CONSTRAINT customer_pk PRIMARY KEY (customer_id)
14    );
15    """
16 )
17
18 make_table('customer', sql, cursor_db);
19 cursor_db.execute('ALTER TABLE customer AUTO_INCREMENT = 100000;')
```

Extracting the data from the CSV file and loading the data into a dataframe

```
1 def read_insert_direct(path, query, table):
2     first = True
3     with open(path, newline='') as csv_file:
4         data = csv.reader(csv_file, delimiter=',', quotechar='"')
5
6         for row in data:
7             if not first:
8                 cursor_db.execute(query, row)
9                 first = False
10    conn_db.commit()
11
12    display_table(table, conn_db, rows = 1)
```

Data Insertion in Customer Table

```
7 sql_cu = ( """
8     INSERT INTO customer ( customer_first_name, customer_last_name, street_address, city, state, country,
9         postal_code, segment)
10    VALUES (%s, %s, %s, %s, %s, %s, %s, %s)
11    """
12 )
13
1 read_insert_direct('Dataset/customer.csv', sql_cu, 'customer')
```

Product Table Creation

```
1 sql = ( """
2     CREATE TABLE product
3     (
4         product_id VARCHAR(15) NOT NULL,
5         product_name VARCHAR(255) NOT NULL,
6         base_price FLOAT NOT NULL,
7         quantity_available INT NOT NULL,
8         category_id CHAR(3) NOT NULL,
9         distributor_id INT NOT NULL,
10        CONSTRAINT product_pk PRIMARY KEY (product_id),
11        CONSTRAINT product_fk1 FOREIGN KEY (distributor_id) REFERENCES distributor(distributor_id),
12        CONSTRAINT product_fk2 FOREIGN KEY (category_id) REFERENCES category(category_id)
13    );
14    """
15 )
16
17 make_table('product', sql, cursor_db)
```

Data Insertion in Product Table

```
43
44 sql_p = ( """
45     INSERT INTO product
46     VALUES (%s, %s, %s, %s, %s, %s)
47     """
48 )
49
```

```
1 read_insert_direct('Dataset/product.csv', sql_p, 'product')
```

Orders Table Creation

```
1 sql = ( """
2     CREATE TABLE orders
3     (
4         order_id INT NOT NULL AUTO_INCREMENT,
5         order_date DATE NOT NULL,
6         discount INT DEFAULT 0,
7         order_total FLOAT DEFAULT 0,
8         ship_mode CHAR(20) NOT NULL,
9         ship_date DATE NOT NULL,
10        customer_id INT NOT NULL,
11        CONSTRAINT orders_pk PRIMARY KEY (order_id),
12        CONSTRAINT orders_fk1 FOREIGN KEY (customer_id) REFERENCES customer(customer_id),
13        CONSTRAINT orders_fk2 FOREIGN KEY (ship_mode) REFERENCES ship_mode(ship_mode)
14    );
15    """
16 )
17
18 make_table('orders', sql, cursor_db);
19 cursor_db.execute('ALTER TABLE orders AUTO_INCREMENT = 10000001;')
```

Data Insertion in Orders Table

```
14 sql_o = ( """
15     INSERT INTO orders (order_date, discount, ship_mode, ship_date, customer_id)
16     VALUES (%s, %s, %s, %s, %s)
17     """
18 )
19
1 read_insert_orders('Dataset/orders.csv', sql_o, 'orders')
```

Note*

To see Other Table creation and insertion of data you can refer **Superstore_db.ipynb** File in **Load_Database** Folder or **operational_database_load.pdf** File in **Other Material** Folder.

All the csv Files Loaded are in the Load_Database Folder -> Dataset Folder

ANALYTICAL DATABASE DESIGN

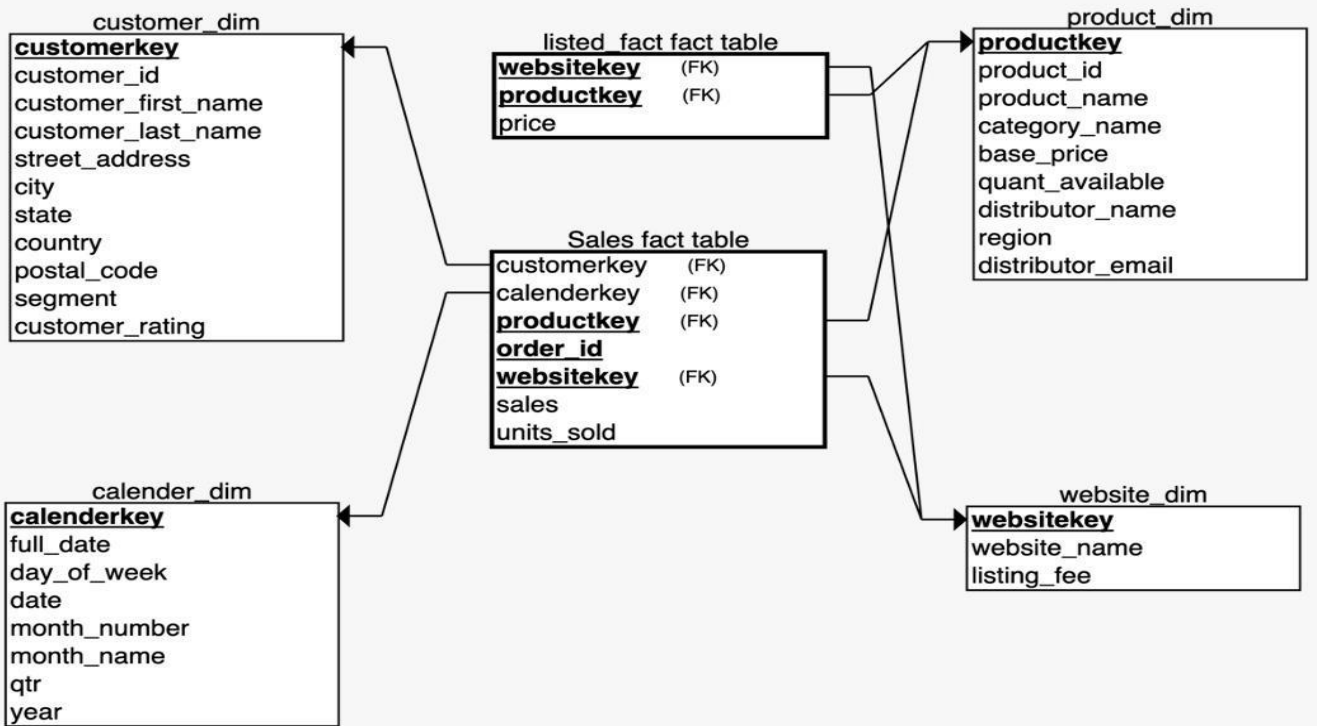
In Analytical Database we have designed 4 Dimension tables as below:

- 1. calendar_dim:** This table contains information about calendar dates and related attributes, it provides date-related information for analyzing and grouping sales data by various time dimensions such as day, month, quarter, and year.
- 2. customer_dim:** This table holds customer details such as names, addresses, segments, and ratings, it allows for analyzing customer-related metrics and segmenting customers based on attributes like location, segment, and rating.
- 3. website_dim:** This table stores information about websites, including their names and listing fees, it enables analysis of website performance and cost considerations by providing data on listing fees and associated sales.
- 4. product_dim:** This table contains information about products, including their names, prices, availability, categories, and distributor details, it allows for analyzing product-related metrics, such as sales by category, distributor performance, and product availability.

In Analytical Database we have designed 2 Fact tables as below:

- 1. listed_fact:** This table represents the pricing information for products listed on websites, it facilitates tracking and analyzing the pricing of products across different websites, enabling price comparison and monitoring changes over time.
- 2. sales_fact:** This table captures sales transactions and associated metrics, including order IDs, sales amounts, and units sold, it serves as the central table for analyzing sales data, enabling various types of analysis such as sales by customer, product, website, and time.

Star Schema



ETL to ANALYTICAL DATABASE

Creation of Customer Dimension

```

In [5]: sql = ( """
        CREATE TABLE customer_dim
        (
        customerkey INT NOT NULL AUTO_INCREMENT,
        customer_id VARCHAR(10) NOT NULL,
        customer_first_name CHAR(30) NOT NULL,
        customer_last_name CHAR(30) NOT NULL,
        street_address CHAR(40) NOT NULL,
        city CHAR(30) NOT NULL,
        state CHAR(30) NOT NULL,
        country CHAR(30) NOT NULL,
        postal_code INT NOT NULL,
        segment CHAR(15) NOT NULL,
        customer_rating INT,
        PRIMARY KEY (customerkey)
        );
        """
    )

    make_table('customer_dim', sql, cursor_wh)
  
```

Insertion into Customer Dimension

```
In [6]: sql = ( """
        INSERT INTO datapirates_wh.customer_dim(customer_id, customer_first_name, customer_last_name, street_address,city,
        state, country, postal_code, segment, customer_rating)
        SELECT customer.customer_id, customer_first_name, customer_last_name, street_address, city, state, country, postal_code,
        segment, AVG(customer_rating)
        FROM customer LEFT JOIN feedback ON customer.customer_id = feedback.customer_id
        GROUP BY customer.customer_id
        """
    )

    cursor_db.execute(sql)
    conn_db.commit()

    display_table('customer_dim', conn_wh, rows = 1)
```

Creation of Sales Fact Table

```
In [15]: sql = ( """
        CREATE TABLE sales_fact
        (
            customerkey INT NOT NULL,
            calendarkey INT NOT NULL,
            websitekey INT NOT NULL,
            productkey INT NOT NULL,
            order_id INT NOT NULL,
            sales FLOAT,
            units_sold INT,
            PRIMARY KEY (productkey, order_id),
            FOREIGN KEY (customerkey) REFERENCES customer_dim(customerkey),
            FOREIGN KEY (calendarkey) REFERENCES calendar_dim(calendarkey),
            FOREIGN KEY (websitekey) REFERENCES website_dim(websitekey),
            FOREIGN KEY (productkey) REFERENCES product_dim(productkey)
        );
        """
    )

    make_table('sales_fact', sql, cursor_wh)
```

Insertion into Sales Fact Table

```
In [3]: sql = ( """
        INSERT INTO datapirates_wh.sales_fact(customerkey, calendarkey, websitekey, productkey, order_id, sales,
        units_sold)
        SELECT datapirates_wh.customer_dim.customerkey, datapirates_wh.calendar_dim.calendarkey,
        datapirates_wh.website_dim.websitekey, datapirates_wh.product_dim.productkey, orders.order_id,
        SUM(orders.order_total) AS sales, SUM(shopping_cart.quantity_ordered)
        FROM orders, shopping_cart, datapirates_wh.customer_dim, datapirates_wh.calendar_dim,
        datapirates_wh.website_dim, datapirates_wh.product_dim
        WHERE orders.order_id = shopping_cart.order_id
        AND orders.order_date = datapirates_wh.calendar_dim.full_date
        AND orders.customer_id = datapirates_wh.customer_dim.customer_id
        AND shopping_cart.product_id = datapirates_wh.product_dim.product_id
        AND shopping_cart.website_name = datapirates_wh.website_dim.website_name
        """
    )

    cursor_db.execute(sql)
    conn_db.commit()

    display_table('sales_fact', conn_wh, rows = 1)
```


USE CASES

Operational Database

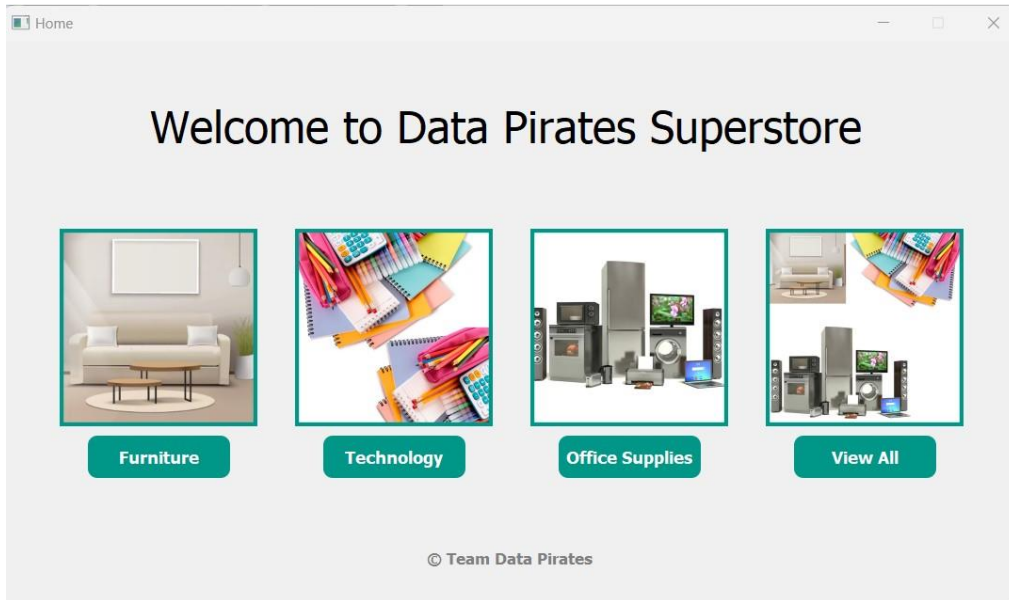


Figure 1: Welcome Page

Figure 1, depicts the initial page of the operational database for a superstore. Customers have the option to search for products either by name or category.

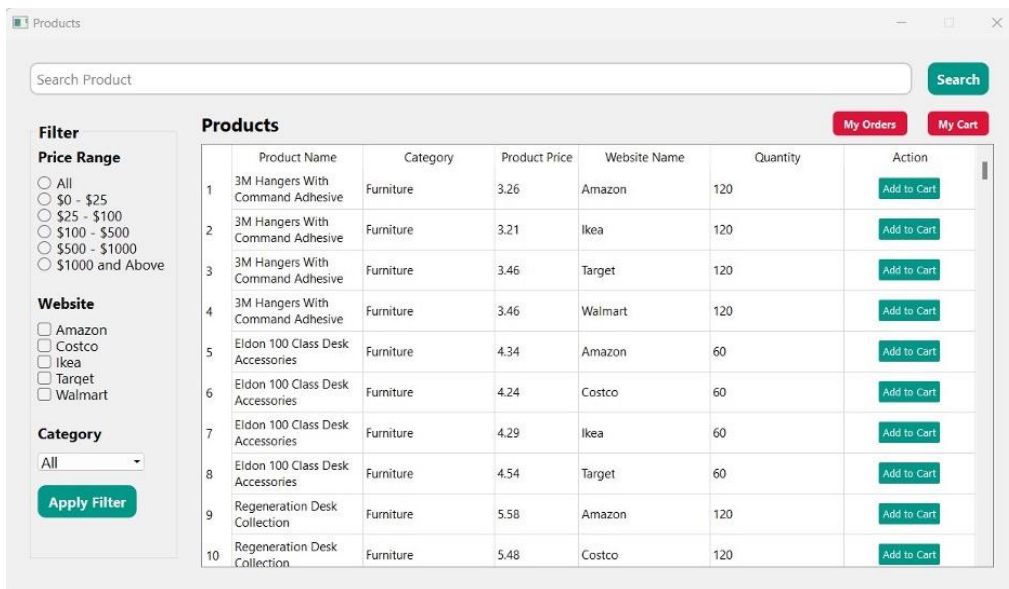


Figure 2: Product List

In Figure 2, a comprehensive list of available products is displayed based on the selection made on the Welcome Page.

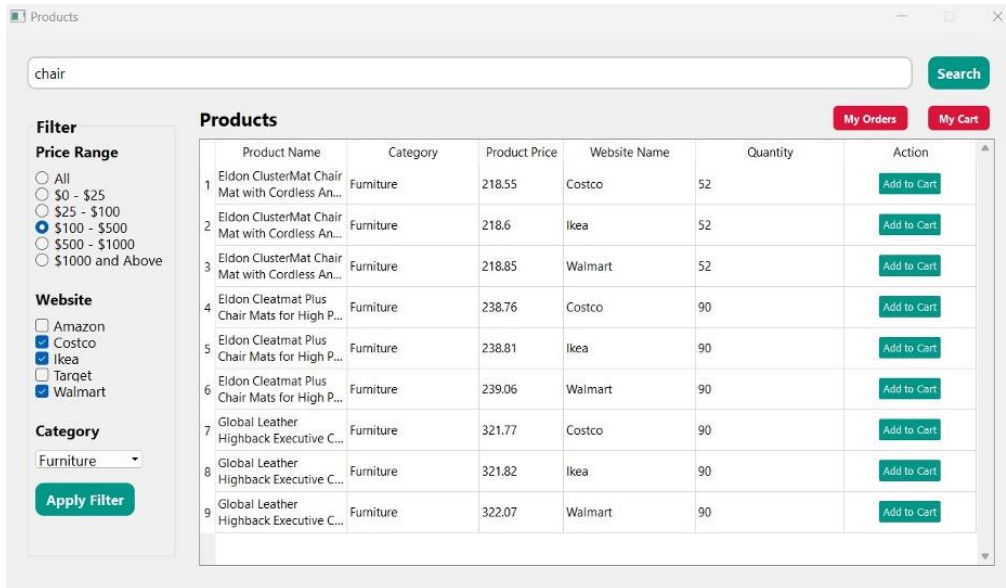


Figure 3: Filtered Product Search

Figure 3, showcases the search results obtained after applying various filters. Specifically, it demonstrates a search for chairs priced between \$100 and \$500 from websites such as 'Costco', 'Ikea', and 'Walmart' within the 'Furniture' category.

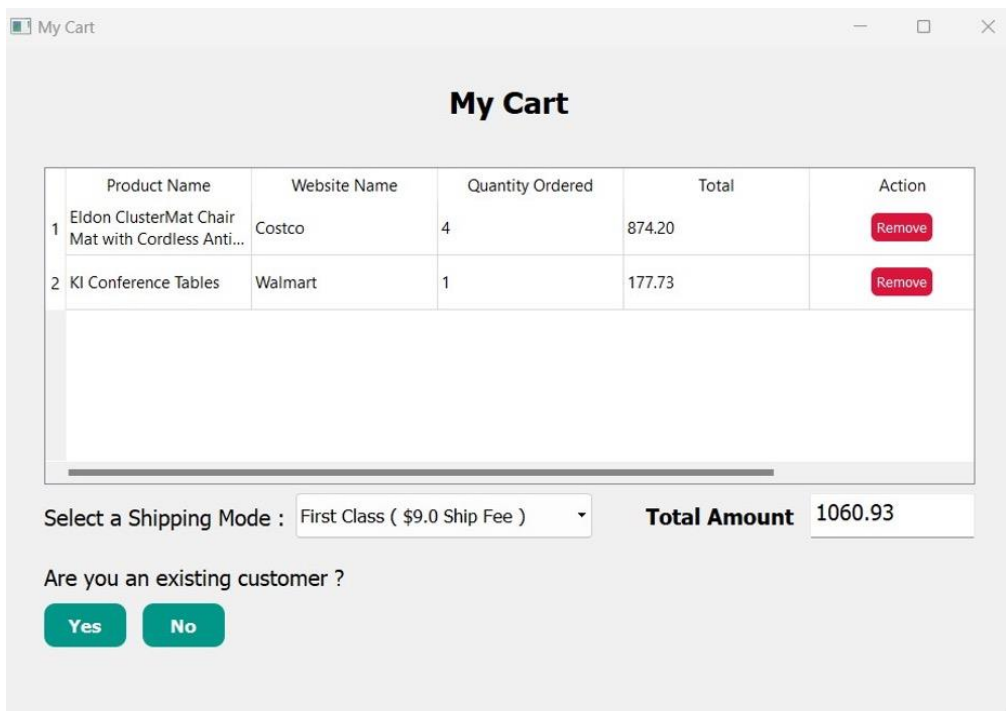


Figure 4: Shopping Cart

Figure 4, illustrates the customer's shopping cart, which contains all the selected products. Customers have the option to remove items if needed. The default shipping option is 'Standard Class' with a \$5.0 shipping fee. Customers can choose alternative shipping methods. The total cost, including product prices

and shipping fees, is displayed. To finalize an order, customers must indicate whether they are existing customers. If so, they need to validate their Customer ID. Otherwise, they need to fill out the Customer Registration Form.

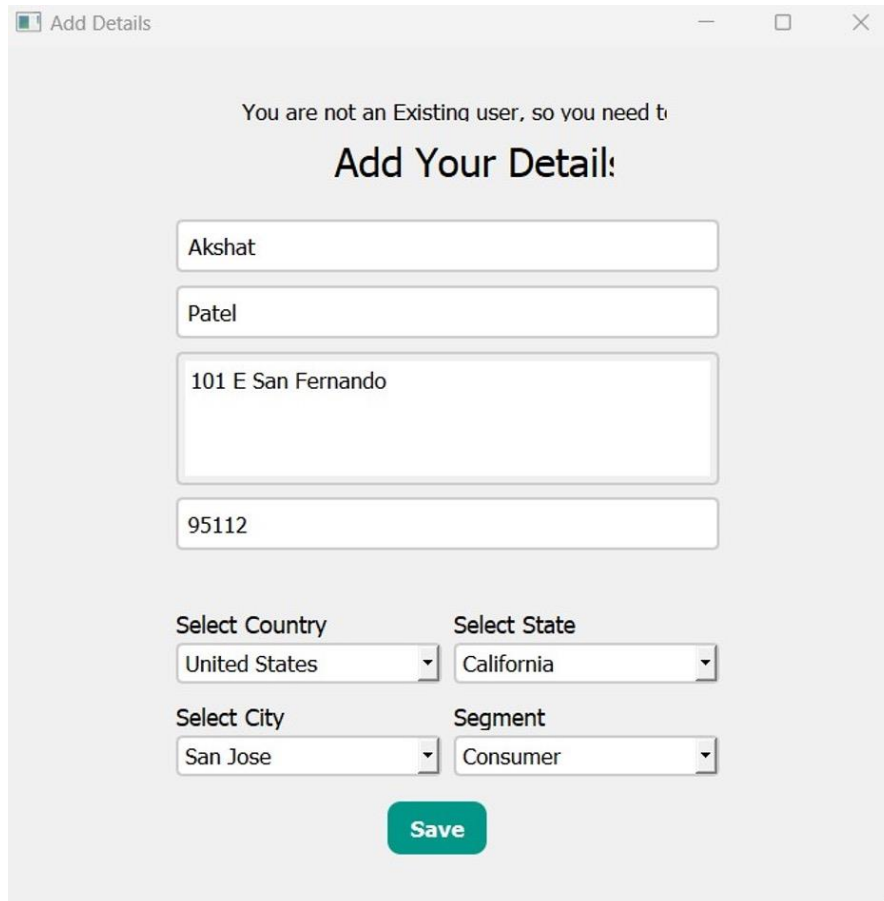
A screenshot of a web application window titled "Add Details". The window has a light gray background and standard window controls (minimize, maximize, close) in the top right corner. The main heading inside the window is "Add Your Details". Below the heading, there are four text input fields stacked vertically. The first field contains "Akshat", the second contains "Patel", the third contains "101 E San Fernando", and the fourth contains "95112". Below these fields are four dropdown menus arranged in two rows. The first row has "Select Country" (with "United States" selected) and "Select State" (with "California" selected). The second row has "Select City" (with "San Jose" selected) and "Segment" (with "Consumer" selected). At the bottom center of the form is a green "Save" button.

Figure 5: New Customer Registration Form

Figure 5, presents a registration form designed for new customers, collecting their demographic information.

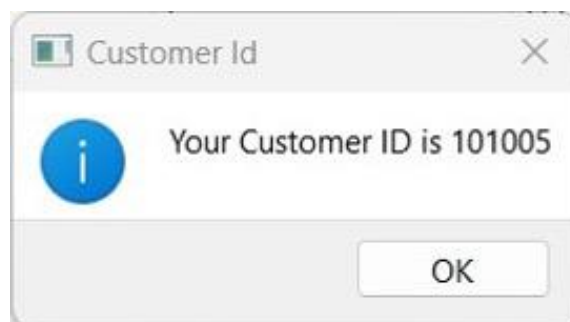


Figure 6: New Customer ID Generated

Once a new customer completes the registration form, they are assigned a new customer ID, as depicted in **Figure 6**.

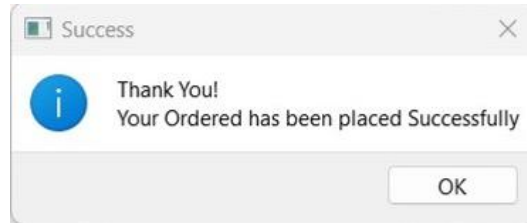


Figure 7: Successful Order Placement

Figure 7, showcases the order status, indicating that an order has been successfully placed.

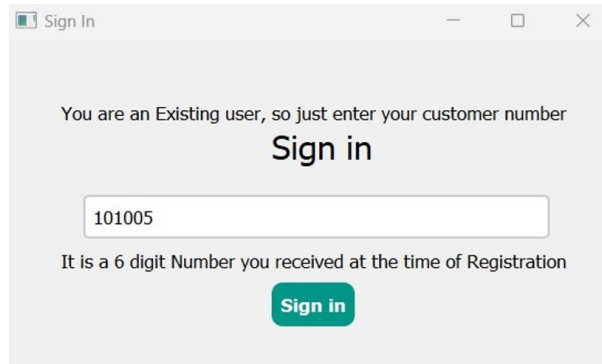


Figure 8: Customer ID Validation

For existing customers to place an order or access their order history, they need to verify their identity by entering their customer ID, as shown in **Figure 8**.

Your Orders						
Customer Id 101005						
	Ordered Date	Product Name	Quantity Ordered	Website	Ship Mode	Ship Date
1	2023-05-22	Kl Conference Tables	1	Walmart	First Class	2023-05-23
2	2023-05-22	Eldon ClusterMat Chair Mat with Cordless ...	4	Costco	First Class	2023-05-23

Figure 9: Order History

Figure 9, provides a comprehensive view of a customer's order history based on the provided customer ID in **Figure 8**.

Analytical Database

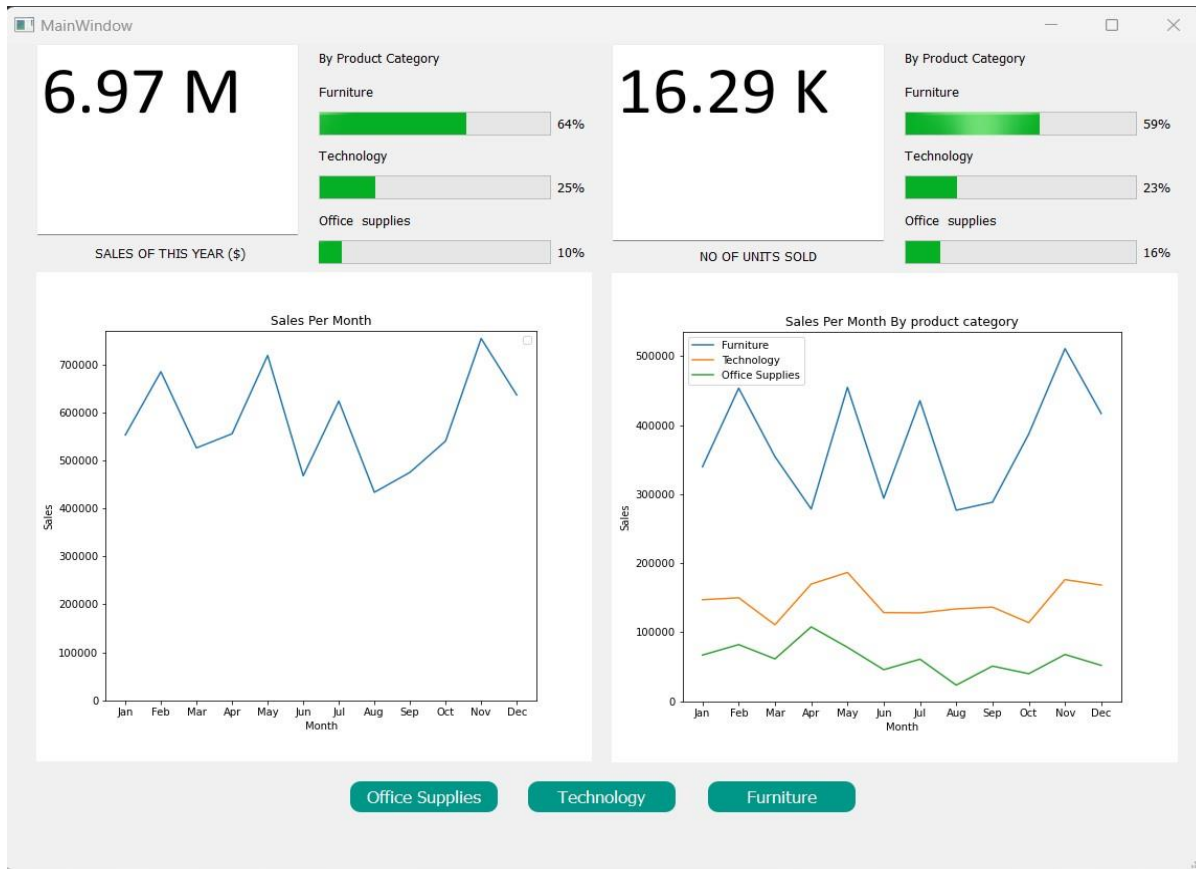


Figure 10: Overall Sales Dashboard

To enhance profitability and facilitate decision-making, **Figure 10** presents the sales dashboard of the analytical database for the superstore.

This dashboard provides an overview of total sales and units sold, as well as trends in overall monthly sales and sales by product category through line graphs, having different product categories namely Furniture, Technology and Office Supplies.

In the bottom it displays 2 line graphs of the sales per month of combined all categories and distinct product categories.

It is useful in helping the superstore agency to identify which product category sells most in which month.

Not only, does the project provide customers with a platform where they can easily compare prices and find the best deals available all-together. But also, it provides a platform for the superstore agency where upon clicking any of the 3 buttons (Furniture, Office Supplies, Technology) on the home page of analytical dashboard it would display in depth details of previous customers ordering that particular category for a particular state.

It displays the,

1. Top products sold (Pie chart)
2. Month-wise sale (Bar chart)
3. Sale count from different websites (Line graph)
4. Order wise sales from those cities of a particular selected state. (Bar graph)

These graphs can be useful in helping the superstore agency to identify :

1. Which product sells the most
2. Monthly sales
3. Which websites sells a particular category the most
4. Cites where the sales of a particular state is the most.

Based on these factors the agency can effectively enhance their business and gain financial profits.



Figure 11: Office Supplies Sales Dashboard

Figure 11, showcases the sales dashboard specifically focused on office supply products. It includes sales information per city, top-selling products, monthly sales, and sales from different websites, with the ability to filter by customer state. Here we have selected California State. All the graphs and charts are in respective of California state.



Figure 12: Technology Sales Dashboard

Figure 12, represents the sales dashboard specifically tailored for technology products. It provides insights into sales per city, top-selling products, monthly sales, and sales from various websites, with the option to filter by customer state. Here we have selected Alabama State. All the graphs and charts are in respective of Alabama state.



Figure 13: Furniture Sales Dashboard

Figure 13, demonstrates the sales dashboard dedicated to furniture products. It offers information on sales per city, top-selling products, monthly sales, and sales from different websites, with the ability to filter by customer state. Here we have selected Florida State. All the graphs and charts are in respective of Florida state.

FUTURE SCOPE

The future scope of the project includes integrating secure payment options, implementing a wishlist feature, expanding filtering options, providing personalized recommendations, incorporating social media integration, developing a mobile application, enhancing inventory management, improving analytics and reporting capabilities, offering multi-language support, exploring external platform integrations, and ensuring internationalization support. These enhancements aim to provide a seamless and personalized shopping experience, expand the reach of the superstore, and optimize business operations.

REFERENCES

1. Alam M, Mohd Noor. Superstore retailing A comprehensive literature review from consumer perspective. 2019;7:163–187.
<https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0279262#>
2. kaggle – <https://www.kaggle.com/datasets/rohitsahoo/sales-forecasting>
3. MySQL - <https://www.mysql.com/>
4. Qt - <https://www.qt.io/>
5. Python - <https://docs.python.org/3/tutorial/index.html>
6. Google – <https://www.google.com/>
7. Stackoverflow - <https://stackoverflow.com/>

NOTES **

To Check ETL in Details you can check **operational_database_load.pdf** file for operational database, **analytical_database_load.pdf** file for analytical database in **Other Material** Folder. You can also see Presentation and GUI screenshots in the screenshots Folder in Other Materials.

To execute an Application,

In Analytical GUI Run **MAIN.ipynb** file and in Operational GUI Run **MainApp.ipynb**.