GROCERY DATABASE MODELLING

Group 3

PROJECT OVERVIEW

The project is about designing a database model for a Grocery store to store, retrieve and perform data analysis to enhance decision-making, and improve customer satisfaction by efficiently managing various data types relevant to the store's operations.

We created different entities and how they would relate to each other in a database.

The key Entities include:

Products: Information about each product, including product ID, name, category, price, quantity in stock, supplier details, and expiration dates.

Suppliers: Details of all suppliers, such as supplier ID, name, contact information, product categories supplied, and delivery schedules.

Customers: Customer profiles containing customer ID, name, contact information, purchase history, and loyalty program details.

Employees: Employee records including employee ID, name, contact information, role, work schedule, and performance reviews.

Order: Includes order ID, order date, delivery date total amount, and customer ID, payment.

Department: Includes department ID, name, and aisle layout.

PRIMARY KEY AND FOREIGN KEY

- Uniqueness: Each customer and order can be uniquely identified, preventing duplicates, and ensuring data integrity.
- Relationship Management: These IDs help establish clear relationships between different data entities, like linking a specific customer to their orders and transactions.
- Efficiency: Unique identifiers make it easier and faster to query and retrieve specific records, enhancing database performance.
- Consistency: Using unique IDs maintains consistency across various database operations, such as updates, deletions, and mergers.
- Data Tracking: Unique identifiers facilitate accurate tracking and analysis of customer behaviors and order histories, supporting better decision-making and personalized experiences.

RELATIONSHIP BETWEEN ENTITES

- The relationship between the customer and customer address tables is characterized as a one-to-many relationship, indicating that each customer may place orders to different addresses, with no nulls permitted.

- The relationship between the customer table and the orders table is also one-to-many, demonstrating that a customer can have multiple orders, while several orders can be associated with a single customer.

- The Customer to order is a one to one relationship. They dependent on each other. This means order entity can not exist without the Customer entities.

- We established a many-to-many relationship between the orders and products tables, as multiple products can correspond to numerous orders and vice versa.

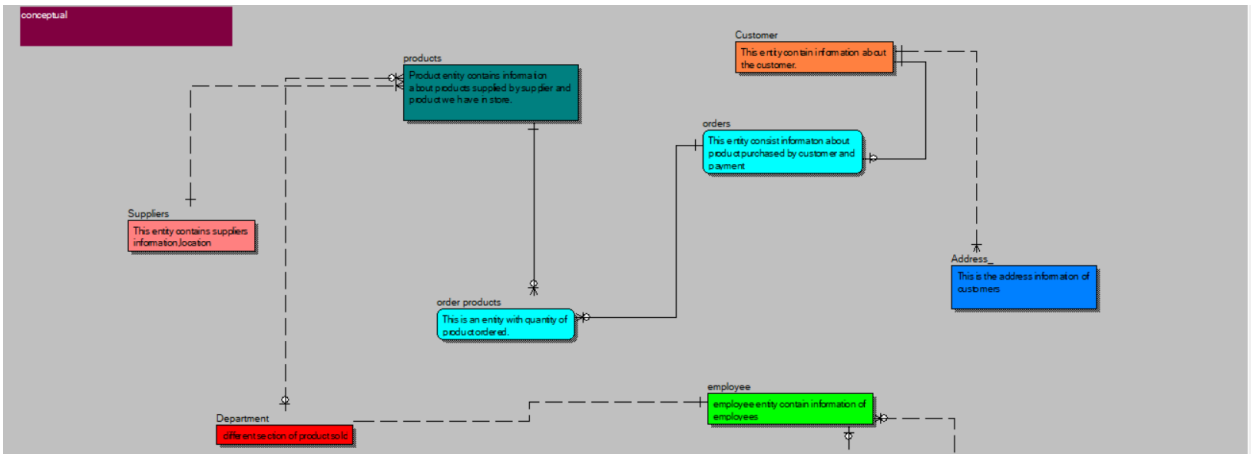
- The relationship between the supplier and product is a one-to-many relationship as this state that a supplier can supply many products.

- Department to Employee entity is a one-to-one relationship stating that each employee must be in one department.

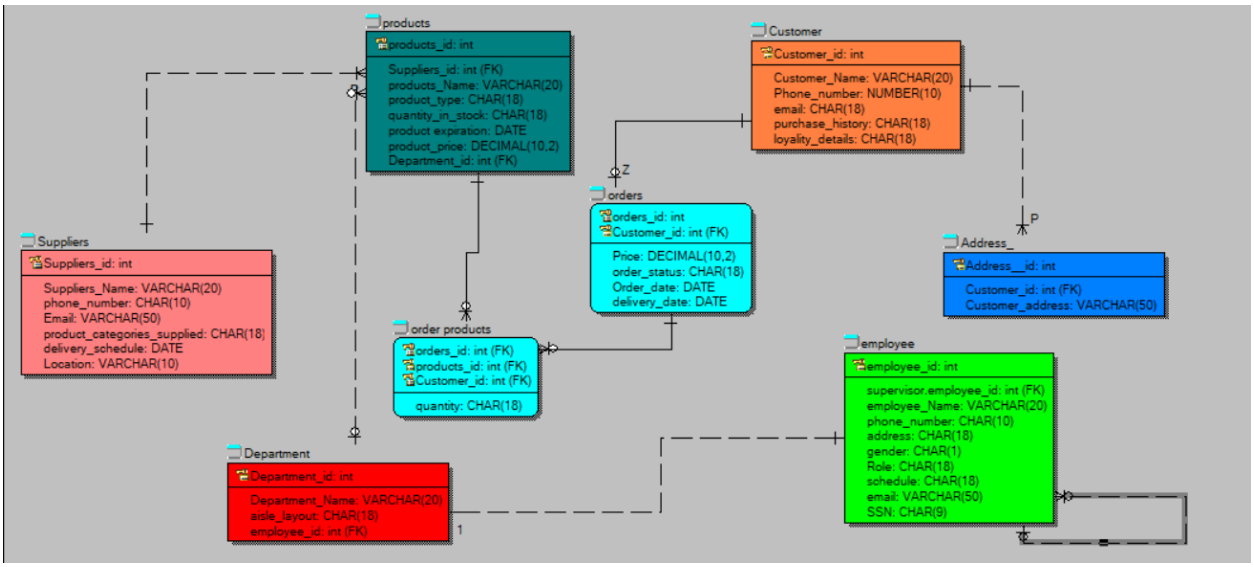
- Department to Product Relationships is a one to many relationships, which means products can be in different categories in the department.

This is a Vitual representation of our models.

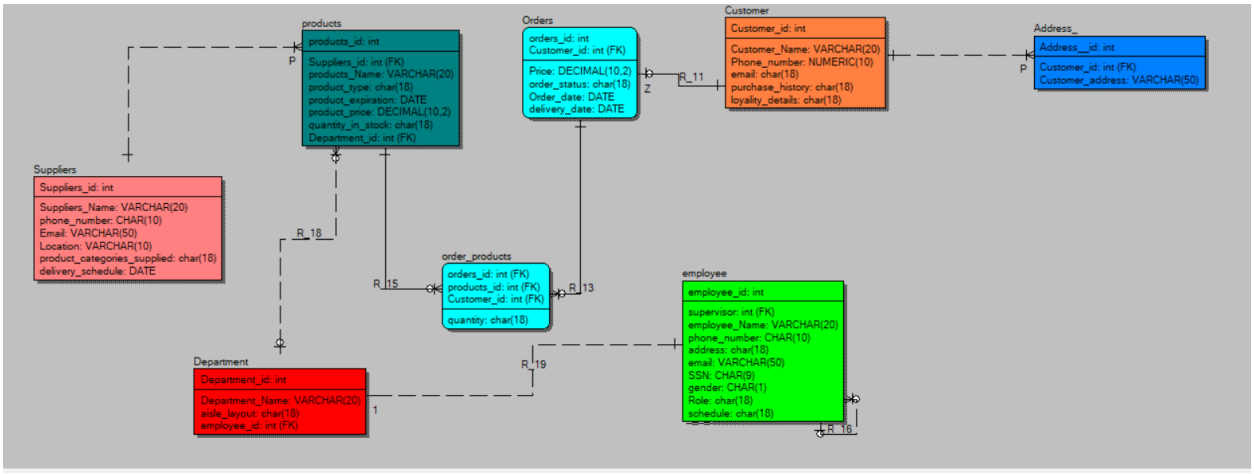
Conceptual Phase: Where we created and defined the entities.



Logical Phase: where the attribute for the entities was described and the relationship, cardinality was defined.



Physical Phase: Where the final deployment of all the relationship done in logical is also shown.



DATABASE IMPLEMENTATION USING **FORWARD ENGINEER** TOOL

This tool generates a SQL script which creates the database with its tables, columns, and constraints from its ERD, and runs this script to have the database ready in MySQL Server.

- This is a part from the code that was generated using **Forward Engineer** tool:

SQLQuery1.sql - L...VB7U9O\scorp (64))*

```
CREATE TABLE Customer
(
    Customer_id          int NOT NULL,
    Customer_Name        VARCHAR(20) NULL,
    Phone_number         NUMERIC(10) NULL,
    email                char(18) NULL,
    purchase_history     char(18) NULL,
    loyality_details     char(18) NULL,
    PRIMARY KEY (Customer_id)
);



CREATE TABLE Orders
(
    orders_id            int NOT NULL,
    Price                DCCIMAL(10,2) NULL,
    order_status         char(18) NULL,
    Order_date           DATE NOT NULL,
    delivery_date        DATE NOT NULL,
    Customer_id          int NOT NULL,
    PRIMARY KEY (orders_id,Customer_id),
    CONSTRAINT R_11 FOREIGN KEY (Customer_id) REFERENCES Customer (Customer_id)
);
```

100 %

Messages

```
Commands completed successfully.

Completion time: 2025-02-27T17:58:25.6670965-05:00
```

After executing the SQL script, the database will be ready to use on the server.