**A Curiosity Shop ERD: Refining Database Architecture**

Kelsey Thompson

Colorado State University Global

MIS407: Database Concepts

Dr. Mario Missakian

May 11, 2024

**A Curiosity Shop ERD: Refining Database Architecture**

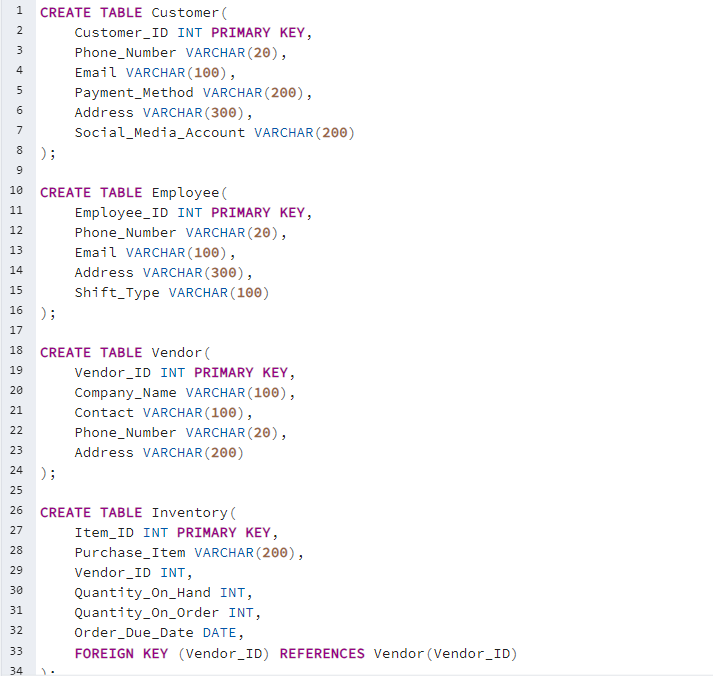
For this assignment, I will be tasked with expanding and trimming down the database applications for the business, Curiosity Shop. Expanding on the current sales records, we will streamline the inventory and customer and employee data management tables, while maintaining the existing data on customers, employees, vendors, sales, and items.

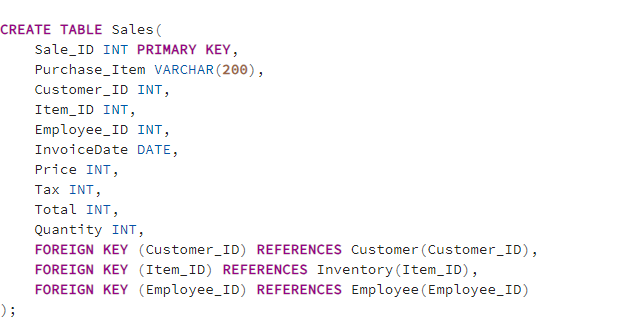
In this project, I will create entities for an Entity-Relationship Diagram (ERD) model based on the Sales and Purchased Items tables, extending them to include the Inventory and Sales and Employee data requirements. I utilized ProgreSQL for this portion.

Creating an ERD involves specifying identifiers and attributes for all entities, while justifying decisions regarding minimum and maximum cardinalities is essential for effective data modeling. I will demonstrate this skill throughout the project.

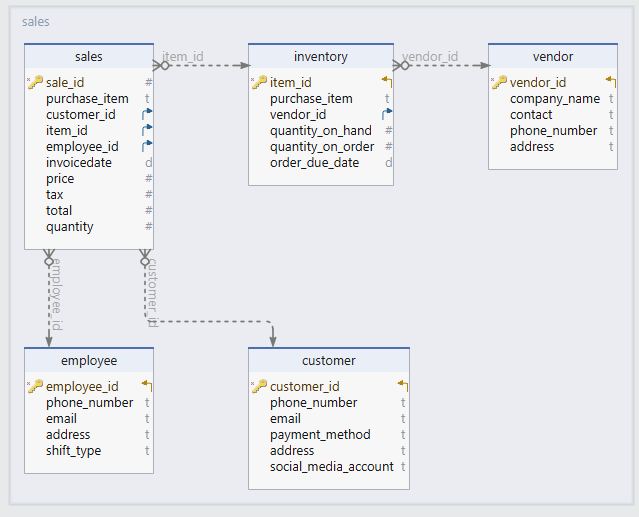
For example, there are one-to-many cardinalities and many-to-many cardinalities in the database. The employee and sales or vendor and inventory stand one-to-many. For many-to-many cardinality, the customers and sales tables adhere.

1. I compartmentalized the tables and developed code tailored to the distinct requirements and data utilization of the sales (customer table and Sales table), HR (employee), and Purchase team (vendors table), as well as management (all tables except inventory table exclusively):





1. Representing, is the corresponding ERD:



1. I would like to discuss the relationships and my driving decisions while I was creating the ERD:

I created an Inventory table that holds a unique item\_ID for all various products that the store carries. The entities for an ERD model were established based on the following tables: Sales, Purchased Items, Inventory, Vendors, and Employee. The identifiers are the ID's (ID's of Sales, Purchased Items, Inventory, Vendors, and Employee) in each table. The attributes consist of each individual column per table for all the entities. Additionally, I created the Employee, Customer, and Vendor tables from scratch. Modifications were made to the Sales and Inventory tables based on the columns within the Sales table and the Purchased Items table provided by the shop. Unnecessary attributes were removed from the original entities as the three new entities were added. Specifically, I removed the LastName, FirstName, and Phone columns from the Sales table (keeping these columns in the customer table alone) and changed InvoiceItem to Purchase\_Item for clarity. I also added the Item\_ID, Quantity, and Customer\_ID columns to the Sales table.

Compartmentalizing for the sales and Purchase teams, HR, and Management, I divided the tables to suit those individual needs and data usage of the marketing team (Customer table and Sales table), HR (employee), and Purchase team (vendors table), and management (all tables but the inventory table exclusively).

Although the managers expressed a desire to have the Customer and Employee tables redesigned using subtypes, I advise against this to maintain clarity and distinction. Keeping these tables separate will prevent the integration of these differentiating tables. If I were to create a parent table, we would need two identifiers for each row, one for Customer and one for Employee. However, most individuals are either one or the other and not both, making this approach less practical. Overall, I find the revised database to satisfy the needs and functionality of the business.