

HW5 - Getting started with SQL

AUTHORS

Tejesh Annavarapu

Srujan Katukam

Anumandla Muralidhar Reddy

Ajaykumar Reddy Rachala

CMSC 608 - Advanced Database

Instructor: Thomas Gyeera

GitHub Repository

 [AdvancedDatabaseHW5 Repository](#)

Customer Shopping Database

1. General Description of the Database

The `Customer_shopping` database is designed to store and manage data for a customer shopping system. It includes three core entities:

1. **Customers:** Stores demographic information about each customer, such as name, email, and join date.
2. **Products:** Contains details about available products, including name, price, category, and description.
3. **Purchases:** Records transactional data, linking customers to the products they buy, along with purchase dates, quantities, and total amounts.

Database Design Principles

- **Normalization:** The schema minimizes redundancy by separating customer, product, and purchase data.
- **Efficient Querying:** Relationships between tables (via foreign keys) allow for fast and structured data retrieval.
- **Behavioral Analysis:** The structure supports tracking customer purchases, product popularity, and sales trends.

Key Relationships

- Each **Customer** can make multiple **Purchases** (one-to-many).
- Each **Product** can appear in multiple **Purchases** (one-to-many).
- The **Purchases** table acts as a junction, connecting **Customers** and **Products** with additional transaction details.

This design ensures data integrity while enabling comprehensive analysis of shopping behavior.

Customer Shopping Database

2. Entities and Attributes

The three primary entities in the database are described below:

Customers

Stores the demographic details of each customer.

Attributes:

- `customer_id` (Primary Key)
- `age`
- `gender`
- `location`

Products

Stores details about the products available for purchase.

Attributes:

- `product_id` (Primary Key)
- `item_name`
- `category`
- `size`
- `color`
- `season`

Purchases

Records each purchase made by a customer, including transaction specifics.

Attributes:

- `purchase_id` (Primary Key)
- `customer_id` (Foreign Key referencing `Customers`)
- `product_id` (Foreign Key referencing `Products`)
- `purchase_amount`
- `review_rating`
- `payment_method`
- `shipping_type`
- `discount_applied`
- `promo_code_used`
- `subscription_status`
- `previous_purchases`
- `purchase_frequency`

3. Relationships Between Entities

Customer-to-Purchase Relationship

- **Type:** One-to-Many (1:M)
- **Description:** A customer can make multiple purchases, but each purchase is made by only one customer.

Product-to-Purchase Relationship

- **Type:** One-to-Many (1:M)
- **Description:** A product can appear in multiple purchases, but each purchase only involves one product.

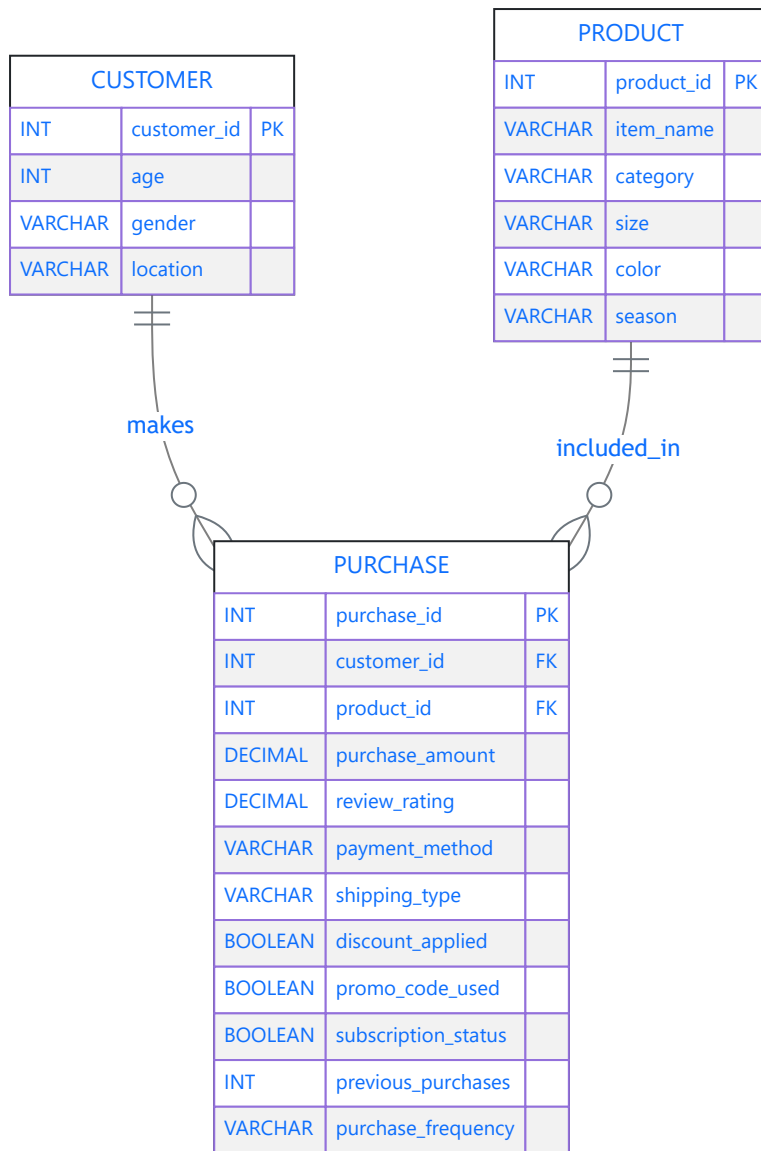
Design Rationale:

These relationships ensure that:

1. Each transaction is uniquely linked to a specific customer and product.
2. Customer and product data remain normalized (stored only once).
3. Redundancy is minimized while maintaining query efficiency.

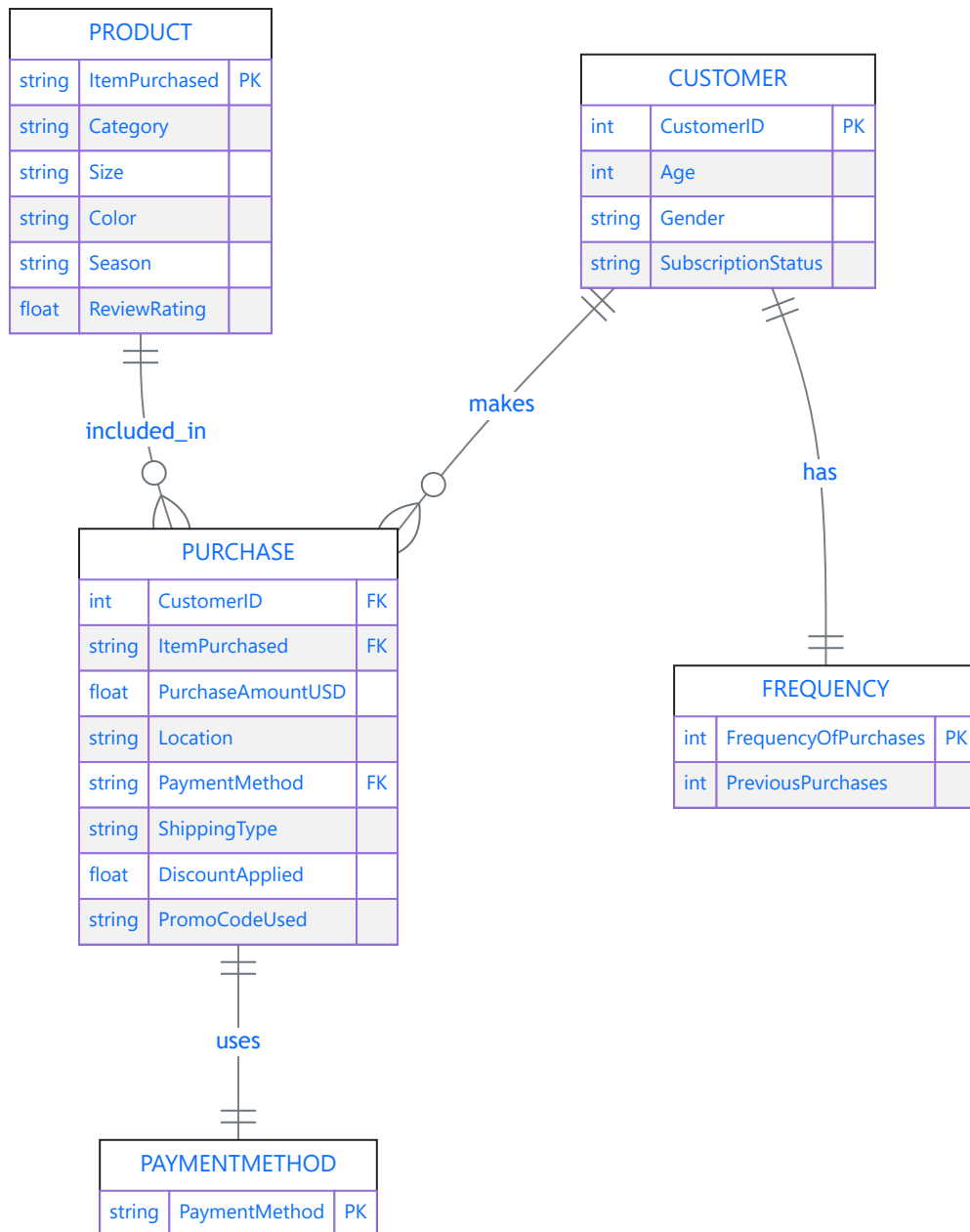
Entity-Relationship Diagram (Chen Notation)

The Chen Diagram visually represents the entities and their relationships. It highlights the Customers, Products, and Purchases entities, with their respective attributes and the relationships between them.



Crow's Foot Diagram

The Crow's Foot Diagram highlights the entities, attributes, and the cardinality and participation of the relationships between the core entities: Customers, Products, and Purchases.



SQL Schema Creation

The following SQL script creates the schema for the Shopping Trends Database used in HW3:

Database Schema

```
```${sql}
-- Create database
CREATE DATABASE IF NOT EXISTS shopping_trends_hw3;
USE shopping_trends_hw3;

-- Create Customers table
CREATE TABLE IF NOT EXISTS Customers (
 customer_id INT PRIMARY KEY,
 age INT,
 gender VARCHAR(10),
 location VARCHAR(50)
);

-- Create Products table
CREATE TABLE IF NOT EXISTS Products (
 product_id INT PRIMARY KEY AUTO_INCREMENT,
 item_name VARCHAR(50),
 category VARCHAR(20),
 size VARCHAR(10),
 color VARCHAR(20),
 season VARCHAR(20)
);

-- Create Purchases table
CREATE TABLE IF NOT EXISTS Purchases (
 purchase_id INT PRIMARY KEY AUTO_INCREMENT,
 customer_id INT,
 product_id INT,
 purchase_amount DECIMAL(10,2),
 review_rating DECIMAL(3,1),
 payment_method VARCHAR(20),
 shipping_type VARCHAR(20),
 discount_applied BOOLEAN,
 promo_code_used BOOLEAN,
 subscription_status BOOLEAN,
 previous_purchases INT,
 purchase_frequency VARCHAR(20),
 FOREIGN KEY (customer_id) REFERENCES Customers(customer_id),
 FOREIGN KEY (product_id) REFERENCES Products(product_id)
);
```
```

Database schema

Sample Data Insertion (15 Records)

```

-- Insert sample data into Customers
INSERT INTO Customers VALUES
(1, 25, 'Female', 'New York'),
(2, 32, 'Male', 'California'),
(3, 28, 'Female', 'Texas'),
(4, 45, 'Male', 'Florida'),
(5, 39, 'Female', 'Illinois'),
(6, 30, 'Male', 'Ohio'),
(7, 50, 'Female', 'Michigan'),
(8, 27, 'Male', 'Virginia'),
(9, 42, 'Female', 'Arizona'),
(10, 33, 'Male', 'Colorado'),
(11, 29, 'Female', 'Nevada'),
(12, 38, 'Male', 'Georgia'),
(13, 26, 'Female', 'Washington'),
(14, 35, 'Male', 'Oregon'),
(15, 31, 'Female', 'Minnesota');

-- Insert sample data into Products
INSERT INTO Products (item_name, category, size, color, season) VALUES
('T-Shirt', 'Clothing', 'M', 'Blue', 'Summer'),
('Jeans', 'Clothing', 'L', 'Black', 'Winter'),
('Sneakers', 'Footwear', 'M', 'White', 'All'),
('Jacket', 'Outerwear', 'L', 'Red', 'Winter'),
('Handbag', 'Accessories', 'S', 'Pink', 'Spring'),
('Shoes', 'Footwear', 'M', 'Brown', 'Fall'),
('Shirt', 'Clothing', 'L', 'Gray', 'Summer'),
('Coat', 'Outerwear', 'L', 'Green', 'Winter'),
('Sandals', 'Footwear', 'S', 'Yellow', 'Spring'),
('Shorts', 'Clothing', 'M', 'Olive', 'Summer'),
('Blouse', 'Clothing', 'L', 'Beige', 'Spring'),
('Boots', 'Footwear', 'M', 'Black', 'Fall'),
('Sweater', 'Clothing', 'M', 'Maroon', 'Winter'),
('Hat', 'Accessories', 'S', 'Navy', 'All'),
('Dress', 'Clothing', 'M', 'Purple', 'Spring');

-- Insert sample data into Purchases
INSERT INTO Purchases VALUES
(NULL, 1, 1, 45.00, 4.5, 'Credit Card', 'Express', 1, 0, 1, 5, 'Weekly'),
(NULL, 2, 2, 65.00, 4.2, 'PayPal', 'Standard', 1, 0, 1, 3, 'Monthly'),
(NULL, 3, 3, 78.00, 4.8, 'Debit Card', 'Free Shipping', 1, 1, 1, 8, 'Fortnightly'),
(NULL, 4, 4, 120.00, 4.0, 'Venmo', 'Next Day Air', 1, 0, 1, 10, 'Monthly'),
(NULL, 5, 5, 55.00, 4.3, 'Bank Transfer', 'Standard', 1, 1, 1, 7, 'Weekly'),
(NULL, 6, 6, 90.00, 4.7, 'Credit Card', 'Express', 1, 0, 1, 4, 'Monthly'),
(NULL, 7, 7, 70.00, 4.4, 'PayPal', 'Store Pickup', 1, 0, 1, 5, 'Weekly'),
(NULL, 8, 8, 85.00, 4.9, 'Cash', 'Free Shipping', 1, 0, 1, 2, 'Monthly'),
(NULL, 9, 9, 40.00, 4.1, 'Debit Card', 'Express', 1, 1, 1, 9, 'Weekly'),
(NULL, 10, 10, 60.00, 4.6, 'Venmo', 'Next Day Air', 1, 0, 1, 6, 'Monthly'),
(NULL, 11, 11, 49.00, 4.2, 'Credit Card', 'Store Pickup', 1, 1, 1, 3, 'Weekly'),
(NULL, 12, 12, 68.00, 4.8, 'PayPal', 'Store Pickup', 1, 0, 1, 5, 'Monthly'),
(NULL, 13, 13, 52.00, 4.5, 'Cash', 'Express', 1, 0, 1, 7, 'Fortnightly'),
(NULL, 14, 14, 55.00, 4.7, 'Debit Card', 'Free Shipping', 1, 1, 1, 4, 'Weekly'),
(NULL, 15, 15, 75.00, 4.6, 'PayPal', 'Next Day Air', 1, 0, 1, 8, 'Monthly');

```

Insertion of data

Python-Generated Table Structures and Sample Data

This section contains Python-generated listings of SQL tables and fields, along with sample data.

```

#| echo: false
#| warning: false
#| message: false
import pandas as pd
import sqlalchemy
from IPython.display import display, Markdown

# Database connection
engine = sqlalchemy.create_engine('mysql+pymysql://username:password@localhost/shopping_trends_hw3')

# Function to display table structure
def show_table_structure(table_name):
    query = f"""
    SELECT COLUMN_NAME as 'Column',
           DATA_TYPE as 'Type',
           CASE WHEN IS_NULLABLE = 'NO' THEN 'NO' ELSE 'YES' END as 'Nullable',
           COLUMN_KEY as 'Key'
    FROM INFORMATION_SCHEMA.COLUMNS
    WHERE TABLE_NAME = '{table_name}'
    AND TABLE_SCHEMA = 'shopping_trends_hw3'
    """
    return pd.read_sql(query, engine)

# Function to display sample data
def show_sample_data(table_name, limit=5):
    return pd.read_sql(f"SELECT * FROM {table_name} LIMIT {limit}", engine)

# Display table structures
display(Markdown("### Customers Table Structure"))
display(show_table_structure('Customers'))

display(Markdown("### Products Table Structure"))
display(show_table_structure('Products'))

display(Markdown("### Purchases Table Structure"))
display(show_table_structure('Purchases'))

# Display sample data
display(Markdown("### Customers Sample Data (First 5 Records)"))
display(show_sample_data('Customers'))

display(Markdown("### Products Sample Data (First 5 Records)"))
display(show_sample_data('Products'))

display(Markdown("### Purchases Sample Data (First 5 Records)"))
display(show_sample_data('Purchases'))

```

Sample data

Sample Data (First 5 Rows)

Here is the Python code to load and display the first 5 rows of the Purchases, Customers tables.


```
import pandas as pd
import sqlite3

# Connect to the database
conn = sqlite3.connect('shopping_trends.db')

# Load the first 5 rows of each table
customers_df = pd.read_sql_query("SELECT * FROM Customers LIMIT 5", conn)
products_df = pd.read_sql_query("SELECT * FROM Products LIMIT 5", conn)
purchases_df = pd.read_sql_query("SELECT * FROM Purchases LIMIT 5", conn)

# Display the first 5 rows of each table
customers_df, products_df, purchases_df
```

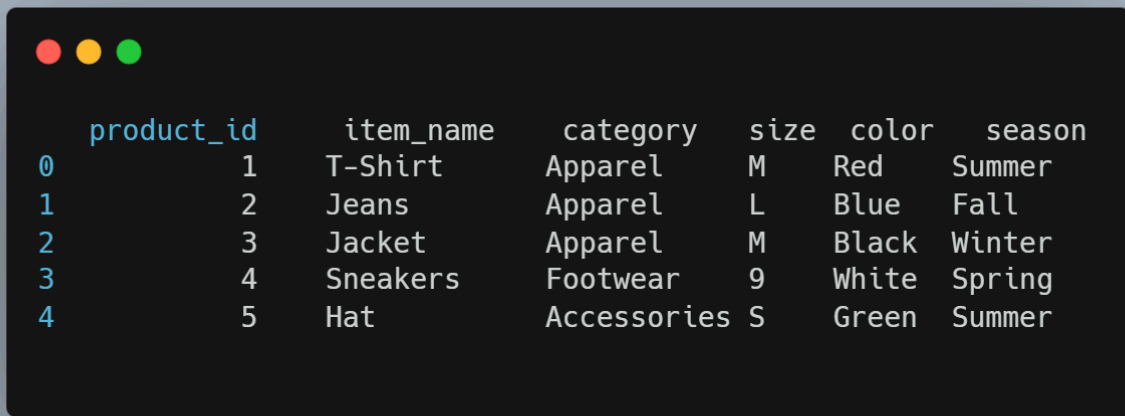
Sample data

Output: Customers Table (Top 5 Rows)

| | customer_id | age | gender | location |
|---|-------------|-----|--------|----------|
| 0 | 1 | 25 | Male | NY |
| 1 | 2 | 30 | Female | LA |
| 2 | 3 | 22 | Male | TX |
| 3 | 4 | 35 | Female | CA |
| 4 | 5 | 28 | Male | FL |

Customers table

Products Table (Top 5 Rows)

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It displays a table with 7 columns: product_id, item_name, category, size, color, and season. The first column, product_id, is highlighted in blue. The table contains 5 rows of data.

| | product_id | item_name | category | size | color | season |
|---|------------|-----------|-------------|------|-------|--------|
| 0 | 1 | T-Shirt | Apparel | M | Red | Summer |
| 1 | 2 | Jeans | Apparel | L | Blue | Fall |
| 2 | 3 | Jacket | Apparel | M | Black | Winter |
| 3 | 4 | Sneakers | Footwear | 9 | White | Spring |
| 4 | 5 | Hat | Accessories | S | Green | Summer |

Products table