

Bookstore Database Project

Overview

This repository contains a simple **relational database design and implementation** for a bookstore system. The project demonstrates how books, customers, and purchases are structured and related using an Entity Relationship Diagram (ERD) and SQL scripts.

The goal of this project is to showcase database modeling, table creation, relationships using primary and foreign keys, and basic data insertion using SQL.

Repository Contents

- **ERD (Entity Relationship Diagram)**

A visual representation of the database structure showing entities, attributes, and relationships between tables.

- **SQL Script**

Contains:

- Table creation statements
 - Primary and foreign key constraints
 - Data insertion statements
 - Table alterations (adding new columns)
 - Sample queries
-

Database Structure

The database consists of **three core tables**:

1. BOOKS

Stores information about books available in the bookstore.

Key Columns: - ISBN (Primary Key) - TITLE - AUTHOR - GENRE - YEAR_OF_PUBLICATION - EDITION

Each book is uniquely identified by its ISBN.

2. CUSTOMERS

Stores information about customers who purchase books.

Key Columns: - `CUST_ID` (Primary Key) - `CUST_NAME` - `EMAIL` - `HOME_ADDRESS` - `PHONE_NO` - `GENDER`

Each customer has a unique customer ID.

3. PURCHASES

Stores transaction records for books purchased by customers.

Key Columns: - `PURC_ID` (Primary Key) - `ISBN` (Foreign Key → BOOKS.ISBN) - `CUST_ID` (Foreign Key → CUSTOMERS.CUST_ID) - `UNIT_PRICE` - `QUANTITY` - `PURC_DATE`

This table creates a relationship between **BOOKS** and **CUSTOMERS**, capturing purchase history.

Relationships (as shown in the ERD)

- One **Book** can appear in many **Purchases**
- One **Customer** can make many **Purchases**
- Each **Purchase** references exactly one **Book** and one **Customer**

These relationships are enforced using **foreign key constraints**.

Features Demonstrated

- Relational database design
 - Primary and foreign key usage
 - Table alteration (`ALTER TABLE`)
 - Data insertion with `INSERT INTO`
 - Basic querying using `SELECT`
 - Realistic sample data for testing
-

How to Use

1. Open the SQL file in your preferred SQL environment (MySQL, SQL Server, PostgreSQL with minor syntax adjustments).
2. Run the script sequentially to:
3. Create tables

4. Insert sample data
 5. Apply table modifications
 6. Use the provided `SELECT` statements or write your own queries to explore the data.
-

Possible Improvements

- Add constraints such as `NOT NULL` and `UNIQUE` where applicable
 - Normalize phone numbers and addresses
 - Add indexes for faster querying
 - Create views and stored procedures
 - Implement triggers for inventory or audit logs
-

Author

Akinmola Daniel

Database Design & SQL Practice Project

License

This project is for learning and demonstration purposes.