SQL project 1

for basic operations. In this project, we will create a database for a fictional bookstore, where we can manage books, authors, and sales. We'll use SQL commands to perform various operations like creating tables, inserting data, querying data, updating information, and deleting records.

Project Overview:

1. **Create a database**: We'll start by creating a database called Bookstore.
2. **Create tables**: We'll create tables to store information about books, authors, and sales.
3. **Insert data**: We'll insert some example data into these tables.
4. **Run queries**: We'll run some basic queries to retrieve data from the tables.
5. **Update and delete**: We'll also demonstrate how to update and delete data from the tables.

Step-by-Step SQL Project

1. Create a Database

First, we need to create a database where all our tables and data will be stored.

```
CREATE DATABASE Bookstore;
```

Explanation : Think of a database like a folder. This folder will hold different pieces of information (tables) related to your bookstore.

2. Create Tables

Now, we will create three tables:

- Books: To store details about each book.
- Authors: To store details about authors.
- Sales: To store information about each book sale.

```
-- Create the 'Books' table
CREATE TABLE Books (
    book_id INT PRIMARY KEY,
    title VARCHAR(100),
    author_id INT,
    price DECIMAL(10, 2),
    genre VARCHAR(50)
);
```

-- Create the 'Authors' table

```
CREATE TABLE Authors (
   author_id INT PRIMARY KEY,
   name VARCHAR(100),
   birth_year INT
);
```

-- **Create the 'Sales' table**

```
CREATE TABLE Sales (
   sale_id INT PRIMARY KEY,
   book_id INT,
   sale_date DATE,
   quantity INT,
   total_amount DECIMAL(10, 2)
);
```

Explanation for the three tables created:

- **Books table**: Stores the book's book_id, title, author_id (who wrote the book), price (cost of the book), and genre (type of book, like fiction or non-fiction).
- **Authors table**: Stores the author_id, name, and birth_year of the author.
- **Sales table**: Stores information about book sales, like sale_id, book_id (which book was sold), sale_date, quantity (how many books were sold), and total_amount (the total money made from the sale).

3. Insert Data

Let's insert some example data into the Books, Authors, and Sales tables.

-- Inserting data into Authors table

```
INSERT INTO Authors (author_id, name, birth_year)
VALUES (1, 'J.K. Rowling', 1965),
    (2, 'George Orwell', 1903),
    (3, 'Jane Austen', 1775);
```

-- **Inserting data into Books table**

```
INSERT INTO Books (book_id, title, author_id, price, genre)
```

VALUES (1, 'Harry Potter and the Sorcerer\'s Stone', 1, 19.99, 'Fantasy'),

(2, '1984', 2, 14.99, 'Dystopian'),

(3, 'Pride and Prejudice', 3, 9.99, 'Romance');

-- Inserting data into Sales table

INSERT INTO Sales (sale_id, book_id, sale_date, quantity, total_amount)

VALUES (1, 1, '2025-04-01', 3, 59.97),

(2, 2, '2025-04-02', 1, 14.99),

(3, 3, '2025-04-03', 2, 19.98);

Explanation for the insertions

- **Authors table**: We are adding three authors with their names and birth years.
- **Books table**: We are adding three books, each with a title, the author's ID (linking to the Authors table), price, and genre.
- **Sales table**: We are adding sales records, showing how many books were sold on a certain date, and the total amount made from the sale.

4. Run Queries

Now that we have data, we can run some queries to retrieve information.

Example 1: List all books and their prices.

SELECT title, price FROM Books;

Explanation: This will give us a list of all books along with their prices.

Example 2: Find all books written by "J.K. Rowling".

SELECT title FROM Books

WHERE author_id = 1;

Explanation: This will give us all the books written by the author with author_id 1, which is J.K. Rowling.

Example 3: Show the total sales amount for each book.

SELECT **Books.**title, SUM(**Sales.**total_amount) AS total_sales

FROM Books

JOIN Sales ON **Books**.book_id = **Sales**.book_id

GROUP BY **Books**.title;

Explanation: This query joins the Books and Sales tables, sums up the sales for each book, and groups the result by book title.

Class work :Return a joined table in bookstore book_title and authors_birthyear.

5. Update Data

Let’s update the price of a book.

UPDATE Books

SET price = 24.99

WHERE book_id = 1;

Explanation: This updates the price of the book with book_id 1 (Harry Potter and the Sorcerer's Stone) to $24.99.

6. Delete Data

Let’s delete a book from the database.

DELETE FROM Books

WHERE book_id = 3;

Explanation: This deletes the book with book_id 3 (Pride and Prejudice) from the Books table.