

Assignment 16.3

NAME:J.ABHIRAM

2403A51342

BATCH:06

Prompts:

Task Description #1 – Schema Generation

Ask AI to design a schema for an Online Course Management System (Tables: Courses, Students, Enrollments).

Task Description #2 – Insert Data

Ask AI to generate INSERT INTO statements for the above schema (at least 3 sample records per table).

Task Description #3 – Basic Queries

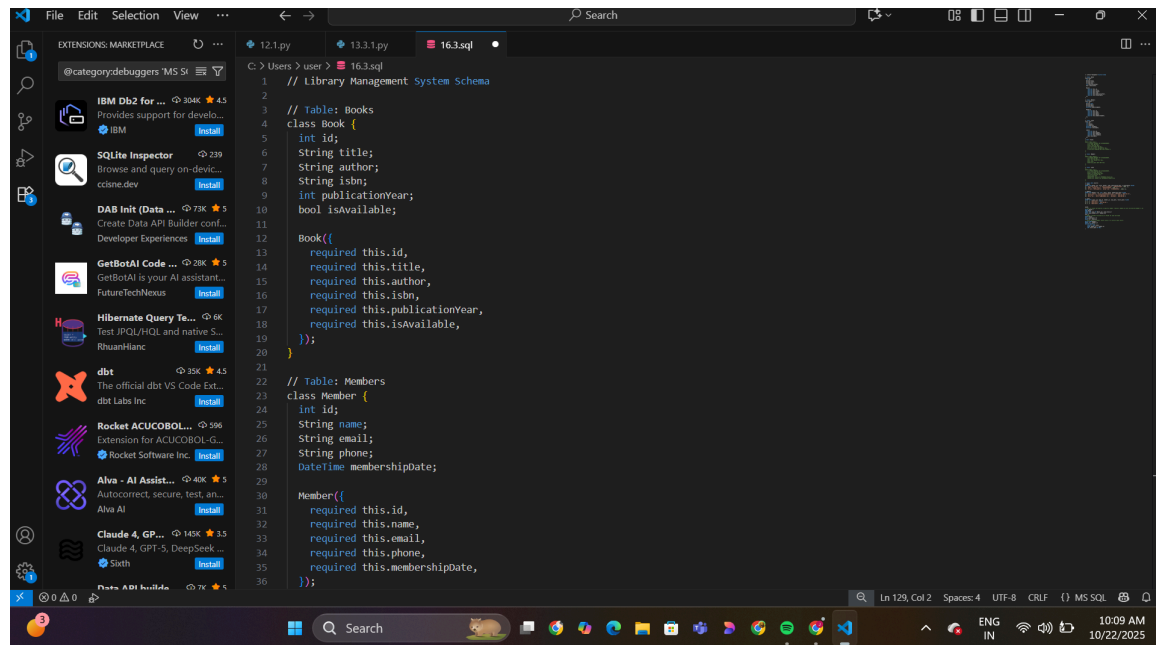
Use AI to create SQL queries to list all courses a particular student is enrolled in.

Task Description #4 – Update and Delete Queries

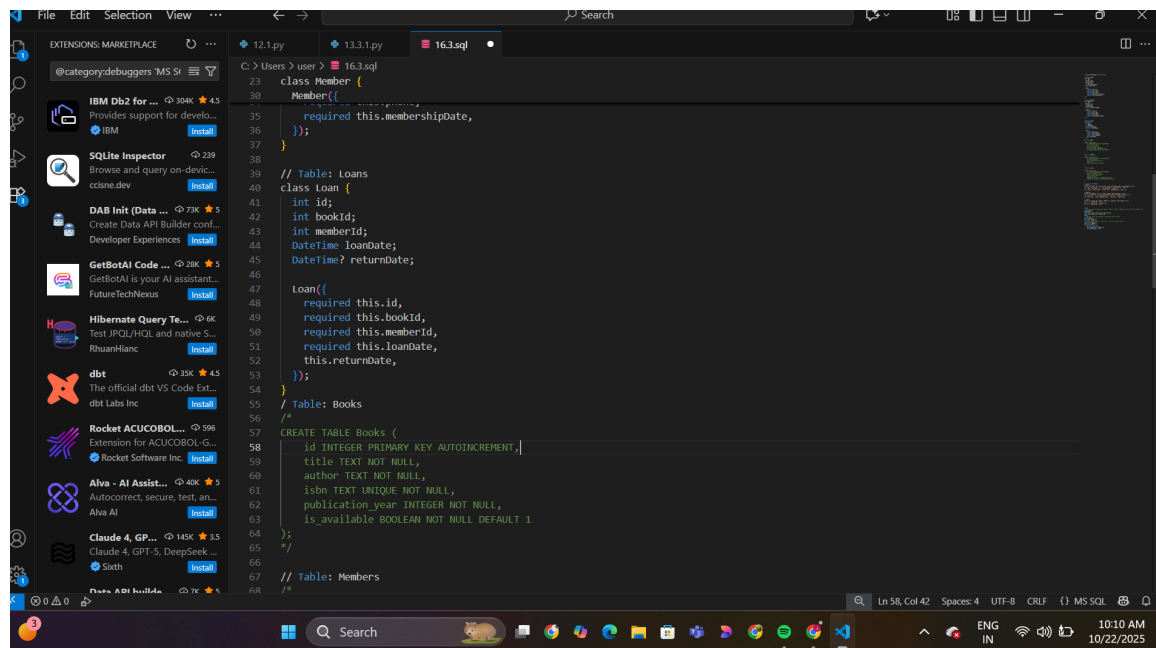
Use AI to generate queries for:

- Updating a course's availability to 'Closed' when seats are full.
- Deleting an enrollment record safely when a student withdraws.

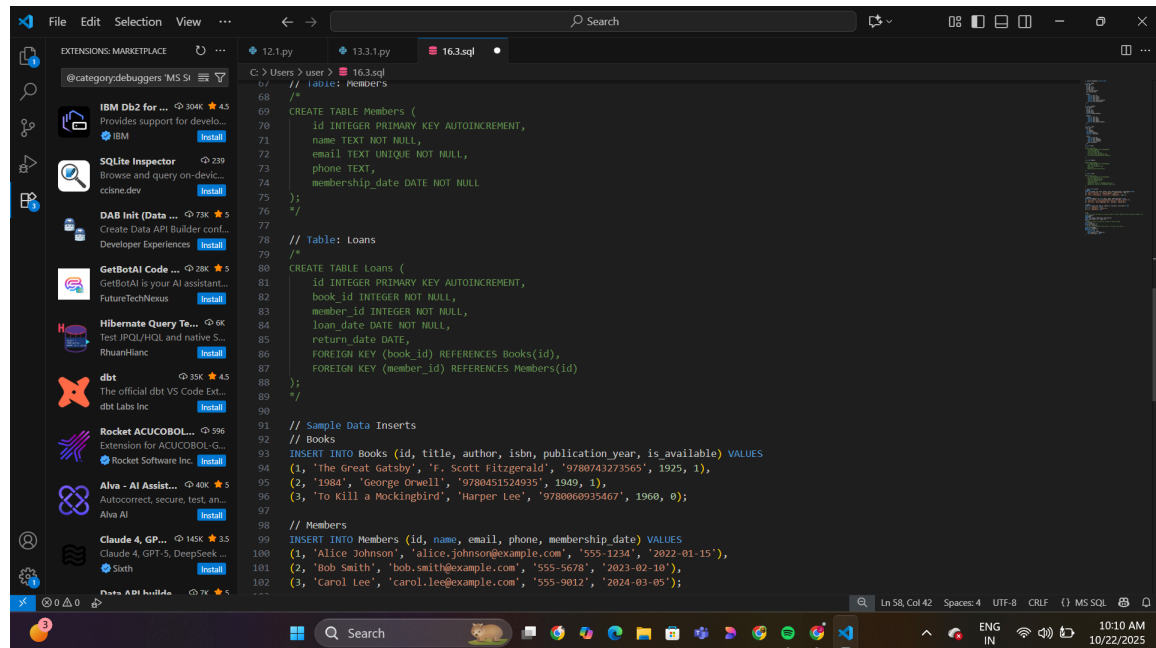
Code:



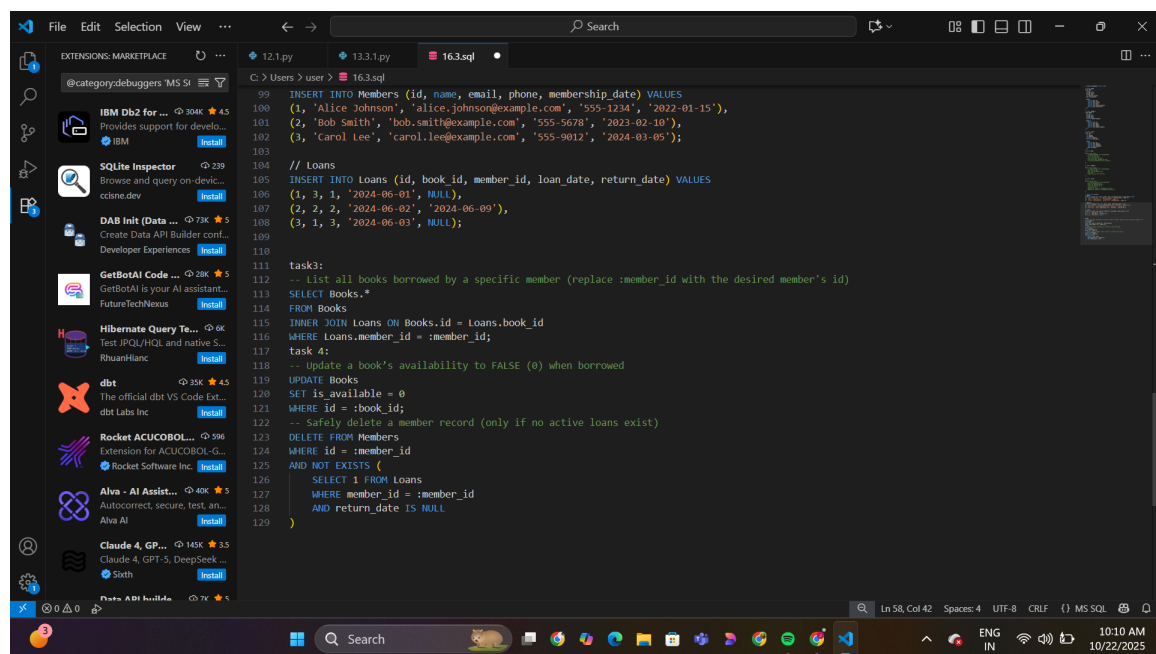
```
1 // Library Management System Schema
2
3 // Table: Books
4 class Book {
5     int id;
6     String title;
7     String author;
8     String isbn;
9     int publicationYear;
10    bool isAvailable;
11
12    Book(
13        required this.id,
14        required this.title,
15        required this.author,
16        required this.isbn,
17        required this.publicationYear,
18        required this.isAvailable,
19    );
20
21 // Table: Members
22 class Member {
23     int id;
24     String name;
25     String email;
26     String phone;
27     DateTime membershipDate;
28
29    Member(
30        required this.id,
31        required this.name,
32        required this.email,
33        required this.phone,
34        required this.membershipDate,
35    );
36 }
```



```
23 class Member {
24     int id;
25     String name;
26     String email;
27     String phone;
28     DateTime membershipDate;
29
30    Member(
31        required this.id,
32        required this.name,
33        required this.email,
34        required this.phone,
35        required this.membershipDate,
36    );
37 }
38
39 // Table: Loans
40 class Loan {
41     int id;
42     int bookId;
43     int memberId;
44     DateTime loanDate;
45     DateTime? returnDate;
46
47    Loan(
48        required this.id,
49        required this.bookId,
50        required this.memberId,
51        required this.loanDate,
52        this.returnDate,
53    );
54 }
55
56 // Table: Books
57 /*
58 CREATE TABLE Books (
59     id INTEGER PRIMARY KEY AUTOINCREMENT,
60     title TEXT NOT NULL,
61     author TEXT NOT NULL,
62     isbn TEXT UNIQUE NOT NULL,
63     publication_year INTEGER NOT NULL,
64     is_available BOOLEAN NOT NULL DEFAULT 1
65 );
66 */
67
68 // Table: Members
69 /*
```



```
163.sql
C:\Users\user> 163.sql
67 // Table: Members
68 /*
69 CREATE TABLE Members (
70     id INTEGER PRIMARY KEY AUTOINCREMENT,
71     name TEXT NOT NULL,
72     email TEXT UNIQUE NOT NULL,
73     phone TEXT,
74     membership_date DATE NOT NULL
75 );
76 */
77
78 // Table: Loans
79 /*
80 CREATE TABLE Loans (
81     id INTEGER PRIMARY KEY AUTOINCREMENT,
82     book_id INTEGER NOT NULL,
83     member_id INTEGER NOT NULL,
84     loan_date DATE NOT NULL,
85     return_date DATE,
86     FOREIGN KEY (book_id) REFERENCES Books(id),
87     FOREIGN KEY (member_id) REFERENCES Members(id)
88 );
89 */
90
91 // Sample Data Inserts
92 // Books
93 INSERT INTO Books (id, title, author, isbn, publication_year, is_available) VALUES
94 (1, 'The Great Gatsby', 'F. Scott Fitzgerald', '9780743273565', 1925, 1),
95 (2, '1984', 'George Orwell', '9780451524935', 1949, 1),
96 (3, 'To Kill a Mockingbird', 'Harper Lee', '9780060935467', 1960, 0);
97
98 // Members
99 INSERT INTO Members (id, name, email, phone, membership_date) VALUES
100 (1, 'Alice Johnson', 'alice.johnson@example.com', '555-1234', '2022-01-15'),
101 (2, 'Bob Smith', 'bob.smith@example.com', '555-5678', '2023-02-10'),
102 (3, 'Carol Lee', 'carol.lee@example.com', '555-9012', '2024-03-05');
```



```
163.sql
99 INSERT INTO Members (id, name, email, phone, membership_date) VALUES
100 (1, 'Alice Johnson', 'alice.johnson@example.com', '555-1234', '2022-01-15'),
101 (2, 'Bob Smith', 'bob.smith@example.com', '555-5678', '2023-02-10'),
102 (3, 'Carol Lee', 'carol.lee@example.com', '555-9012', '2024-03-05');
103
104 // Loans
105 INSERT INTO Loans (id, book_id, member_id, loan_date, return_date) VALUES
106 (1, 3, 1, '2024-06-01', NULL),
107 (2, 2, 2, '2024-06-02', '2024-06-09'),
108 (3, 1, 3, '2024-06-03', NULL);
109
110
111 task3:
112 -- List all books borrowed by a specific member (replace :member_id with the desired member's id)
113 SELECT Books.*
114 FROM Books
115 INNER JOIN Loans ON Books.id = Loans.book_id
116 WHERE Loans.member_id = :member_id;
117
118 task 4:
119 -- Update a book's availability to FALSE (0) when borrowed
120 UPDATE Books
121 SET is_available = 0
122 WHERE id = :book_id;
123
124 -- Safely delete a member record (only if no active loans exist)
125 DELETE FROM Members
126 WHERE id = :member_id
127 AND NOT EXISTS (
128     SELECT 1 FROM Loans
129     WHERE member_id = :member_id
130     AND return_date IS NULL
131 );
```

Observations

1. The schema design using AI reduces manual effort and ensures structural consistency.
2. AI-generated INSERT queries help quickly populate test data for validation.
3. Using AI for query formulation ensures correct syntax and improves learning of SQL patterns.

4. The update and delete queries highlight how AI assists in automating database maintenance tasks.