

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:

The screenshot shows the Microsoft Visual Studio Code interface with the following extensions installed:

- IBM Db2 for VS Code**: Provides support for developing IBM Db2 databases.
- SQLite Inspector**: Allows browsing and querying SQLite databases.
- DAB Init (Data API Builder)**: Creates Data API Builder configurations.
- GetBotAI Code Assistant**: An AI assistant for writing code.
- Hibernate Query Test**: Tests JPQL/HQL and native SQL queries.
- dbt**: The official dbt VS Code Extension.
- Rocket ACUCOBOL**: Extension for ACUCOBOL-GJ.
- Alva - AI Assistant**: Autocorrect, secure, test, and AI writing assistance.
- Claude 4, GPT-4**: Claude 4, GPT-5, DeepSeek, SixthAI AI writing assistance.

The current file being edited is `16.3.sql`, which contains the following code:

```
// Library Management System Schema
-- Table: Books
class Book {
    int id;
    String title;
    String author;
    String isbn;
    int publicationYear;
    bool isAvailable;
}
Book({
    required this.id,
    required this.title,
    required this.author,
    required this.isbn,
    required this.publicationYear,
    required this.isAvailable,
});
// Table: Members
class Member {
    int id;
    String name;
    String email;
    String phone;
    Datetime membershipDate;
}
Member({
    required this.id,
    required this.name,
    required this.email,
    required this.phone,
    required this.membershipDate,
});
```

```

File Edit Selection View ... ← → Search C:\> Users > user 16.3.sql 12.1.py 13.3.py 16.3.sql

C:\> Users > user 16.3.sql
68 // TABLE: Members
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 // Table: Loans
78 /*
79 CREATE TABLE Loans (
80     id INTEGER PRIMARY KEY AUTOINCREMENT,
81     book_id INTEGER NOT NULL,
82     member_id INTEGER NOT NULL,
83     loan_date DATE NOT NULL,
84     return_date DATE,
85     FOREIGN KEY (book_id) REFERENCES Books(id),
86     FOREIGN KEY (member_id) REFERENCES Members(id)
87 );
88 /*
89 /*
90 // Sample Data Inserts
91 // Books
92 INSERT INTO Books (id, title, author, isbn, publication_year, is_available) VALUES
93 (1, 'The Great Gatsby', 'F. Scott Fitzgerald', '9780743273565', 1925, 1),
94 (2, '1984', 'George Orwell', '9780451524935', 1949, 1),
95 (3, 'To Kill a Mockingbird', 'Harper Lee', '9780060935467', 1960, 0);
96 /*
97 // Members
98 INSERT INTO Members (id, name, email, phone, membership_date) VALUES
99 (1, 'Alice Johnson', 'alice.johnson@example.com', '555-1234', '2022-01-15'),
100 (2, 'Bob Smith', 'bob.smith@example.com', '555-5678', '2023-02-10'),
101 (3, 'Carol Lee', 'carol.lee@example.com', '555-9912', '2024-03-05');
102 /*
103 /*

```

Ln 58, Col 42 Spaces: 4 UTF-8 CRLF () MS SQL ENG IN 10:10 AM 10/22/2025

```

File Edit Selection View ... ← → Search C:\> Users > user 16.3.sql 12.1.py 13.3.py 16.3.sql

C:\> Users > user 16.3.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', '555-9912', '2024-03-05');
103 /*
104 // Loans
105 INSERT INTO Loans (id, book_id, member_id, loan_date, return_date) VALUES
106 (1, 1, 1, '2024-06-01', NULL),
107 (2, 2, 2, '2024-06-02', '2024-06-09'),
108 (3, 3, 3, '2024-06-03', NULL);
109 /*
110 task3:
111 -- List all books borrowed by a specific member (replace :member_id with the desired member's id)
112 SELECT Books.*
113 FROM Books
114 INNER JOIN Loans ON Books.id = loans.book_id
115 WHERE Loans.member_id = :member_id;
116 /*
117 task4:
118 -- Update a book's availability to FALSE (0) when borrowed
119 UPDATE Books
120 SET is_available = 0
121 WHERE id = :book_id;
122 -- Safely delete a member record (only if no active loans exist)
123 DELETE FROM Members
124 WHERE id = :member_id
125 AND NOT EXISTS (
126     SELECT 1 FROM Loans
127     WHERE member_id = :member_id
128     AND return_date IS NULL
129 )

```

Ln 58, Col 42 Spaces: 4 UTF-8 CRLF () MS SQL ENG IN 10:10 AM 10/22/2025

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.