Assignment

Course Title: Database Management System
Course Code: CIS222 and CIS222L

Submitted to:

Dr. ZZZZZZZZZZZ
Associate Professor

Department of Computing And Information System
Daffodil International University

Submitted By:

Name: Nahin Fardosh
ID: 193-16-480
Section: A
Department of Computing And Information System
Daffodil International University

Date of Submission: 05/04/2025

Theory Part

Task 1 -Q1

The **relational model** was chosen for the DIU student database because of its **simplicity, flexibility, and powerful querying capabilities**. Here's why:

- ➔ **Data Integrity & Accuracy**:
  Relational databases support constraints like **primary keys**, **foreign keys**, and **unique constraints** that maintain data accuracy and relationships between tables .

- ➔ **Structured Organization**:
  Entities such as students, teachers, and courses can be **clearly structured** into tables with defined attributes and relationships, making the data model easy to understand and manage.

- ➔ **Efficient Querying & Reporting**:
  The **SQL language** allows powerful data retrieval using `JOIN`, `GROUP BY`, `HAVING`, etc., which is essential for a university system where analytics and reports are frequently generated.

Task 1 -Q2

Mr. Fahim might face several issues while working with the relational model:

- ➔ **Normalization Complexity**:
  Ensuring data is normalized (removing redundancy) without losing important relationships can be challenging and may require multiple iterations of design.

- ➔ **Handling Many-to-Many Relationships**:
  Some relationships (students enrolling in many courses, teachers teaching many subjects) require **junction tables**. These can become complex to manage.

➔ **Data Redundancy & Anomalies**:
Without proper normalization and constraints, there may be issues like **update, delete, and insert anomalies**.

➔ **Scalability & Performance**:
As the student database grows, queries on large tables with multiple joins might **slow down performance** if not optimized properly with indexes and best practices.

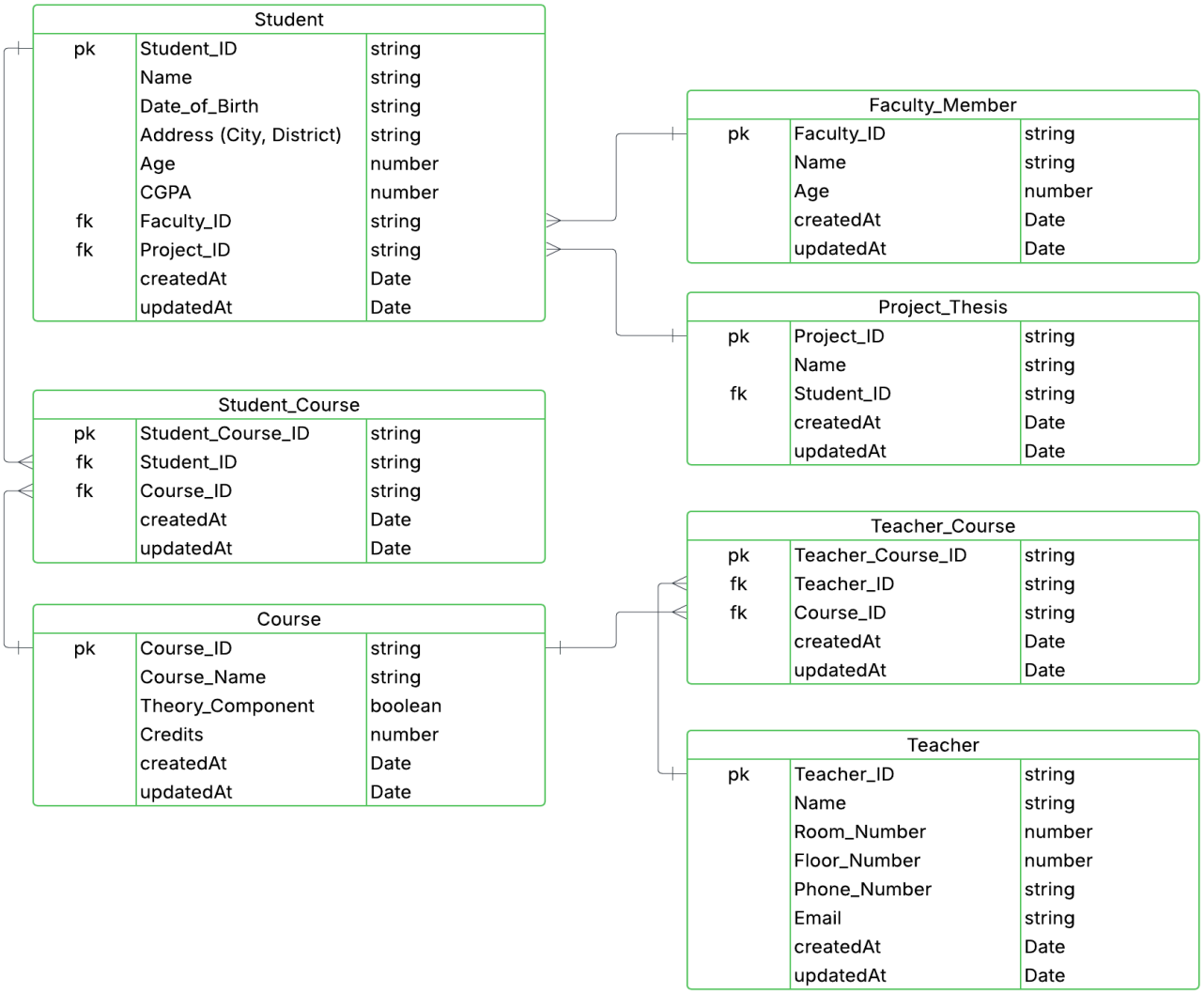
Task 1 -Q3

Mr. Mehedi most likely described a **Weak Entity** in the blood donation system.

➔ A **Weak Entity** is one that **cannot exist without being associated** with another entity (known as the identifying or owner entity).

➔ In a blood donation system, an example could be the **Donation Record**, which cannot exist without a **Donor** entity. The donation details (like date, location, blood type) rely on the existence of the donor.

Task 2 -Q4



Task 2 -Q5

Entity Name	Attribute Name	Data Type	Description	Key Type
faculty_member	Faculty_ID	VARCHAR(50)	Unique ID of the faculty	Primary Key
	Name	VARCHAR(100)	Full name of faculty member	
	Age	INT	Age of the	

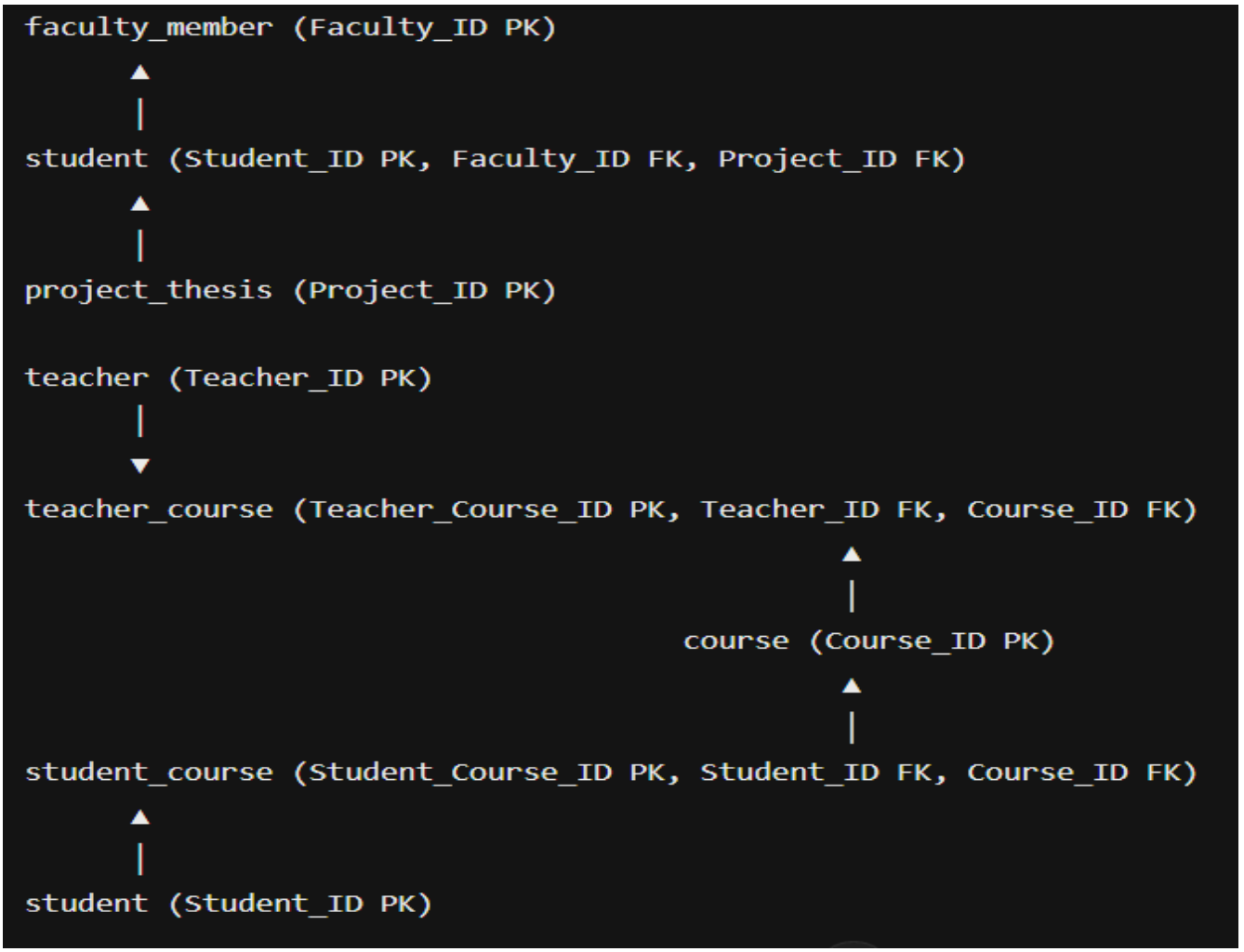
			faculty member	
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	
course	Course_ID	VARCHAR(50)	Unique ID for the course	Primary Key
	Course_Name	VARCHAR(255)	Name of the course	
	Theory_Component	BOOLEAN	Indicates if the course has theory	
	Credits	INT	Course credit value	
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	
teacher	Teacher_ID	VARCHAR(50)	Unique ID of the teacher	Primary Key
	Name	VARCHAR(50)	Full name of the teacher	
	Room_Number	INT	Room number assigned	
	Floor_Number	INT	Floor_number	
	Phone_Number	VARCHAR(20)	Contact number	
	Email	VARCHAR(100)	Email address	
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	
project_thesis	Project_ID	VARCHAR(50)	Unique project ID	Primary Key

	Name	VARCHAR(50)	Name/title of the project	
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	
student	Student_ID	VARCHAR(50)	Unique student ID	Primary Key
	Name	VARCHAR(100)	Student full name	
	Date_of_Birth	DATE	Date of birth	
	Address	VARCHAR(255)	Student's address	
	Age	INT	Student's age	
	CGPA	DECIMAL(3,2)	Student's CGPA	
	Faculty_ID	VARCHAR(50)	Linked faculty ID	Foreign Key
	Project_ID	VARCHAR(50)	Linked project ID	Foreign Key
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	
teacher_course	Teacher_Course_ID	VARCHAR(50)	Unique ID	Primary Key
	Teacher_ID	VARCHAR(50)	Teacher assigned	Foreign Key
	Course_ID	VARCHAR(50)	Course assigned	Foreign Key
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	
student_course	Student_Cours	VARCHAR(50)	Unique ID	Primary

	e_ID			Key
	Student_ID	VARCHAR(50)	Student enrolled	Foreign Key
	Course_ID	VARCHAR(50)	Enrolled course	Foreign Key
	createdAt	TIMESTAMP	Creation timestamp	
	updatedAt	TIMESTAMP	Last update timestamp	

Task 2 -Q6

Below is a textual representation of the relational model (ER-style view):



Task 2 -Q7

```
SELECT s.Student_ID, s.Name, COUNT(sc.Course_ID) AS Course_Count
FROM student s
JOIN student_course sc ON s.Student_ID = sc.Student_ID
GROUP BY s.Student_ID, s.Name
HAVING COUNT(sc.Course_ID) > 2;
```

```
SELECT * FROM teacher
WHERE Email LIKE '%diu%'
AND Phone_Number LIKE '017%';
```

```
SELECT * FROM student
WHERE Address = 'Dhaka' AND CGPA > 3.65;
```

```
SELECT * FROM student
WHERE Address NOT IN ('Dhaka', 'Chittagong');
```

```
SELECT s.Student_ID, s.Name, sc.Course_ID
FROM student s
JOIN student_course sc ON s.Student_ID = sc.Student_ID
WHERE sc.Course_ID IN ('CIS222', 'CIS223', 'CIS224');
```

```
SELECT * FROM project_thesis
WHERE Name LIKE '%Library%' OR Name LIKE '%AI%';
```

```
SELECT * FROM student
WHERE Age BETWEEN 20 AND 24
AND CGPA > 3.71;
```

```
SELECT c.Course_ID, c.Course_Name
FROM course c
LEFT JOIN student_course sc ON c.Course_ID = sc.Course_ID
WHERE sc.Student_ID IS NULL;
```

```
SELECT c.Course_ID, c.Course_Name, COUNT(sc.Student_ID) AS
Student_Count
FROM course c
JOIN student_course sc ON c.Course_ID = sc.Course_ID
GROUP BY c.Course_ID, c.Course_Name
HAVING COUNT(sc.Student_ID) > 1;
```

```
SELECT * FROM student
WHERE Faculty_ID = '710002880';
```

Lab Part

Task 1 -QL1

```
--1 Faculty_Member (Independent)
CREATE TABLE faculty_member (
    Faculty_ID VARCHAR(50) PRIMARY KEY,
    Name VARCHAR(100),
    Age INT,
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

```
--2 Course (Independent)
CREATE TABLE course (
    Course_ID VARCHAR(50) PRIMARY KEY,
    Course_Name VARCHAR(255),
    Theory_Component BOOLEAN,
    Credits INT,
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

```
--3 Teacher (Independent)
CREATE TABLE teacher (
    Teacher_ID VARCHAR(50) PRIMARY KEY,
    Name VARCHAR(100),
    Room_Number INT,
    Floor_Number INT,
    Phone_Number VARCHAR(20),
    Email VARCHAR(100),
```

```
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

```
--4 Project_Thesis (Independent, before Student)
CREATE TABLE project_thesis (
    Project_ID VARCHAR(50) PRIMARY KEY,
    Name VARCHAR(255),
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
);
```

```
--5 Student (Depends on Faculty_Member and Project_Thesis)
CREATE TABLE student (
    Student_ID VARCHAR(50) PRIMARY KEY,
    Name VARCHAR(100),
    Date_of_Birth DATE,
    Address VARCHAR(255),
    Age INT,
    CGPA DECIMAL(3,2),
    Faculty_ID VARCHAR(50),
    Project_ID VARCHAR(50),
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    FOREIGN KEY (Faculty_ID) REFERENCES faculty_member(Faculty_ID),
    FOREIGN KEY (Project_ID) REFERENCES project_thesis(Project_ID)
);
```

```
--6 Teacher_Course (Depends on Teacher and Course)
```

```
CREATE TABLE teacher_course (
    Teacher_Course_ID VARCHAR(50) PRIMARY KEY,
    Teacher_ID VARCHAR(50),
    Course_ID VARCHAR(50),
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    FOREIGN KEY (Teacher_ID) REFERENCES teacher(Teacher_ID),
    FOREIGN KEY (Course_ID) REFERENCES course(Course_ID)
);
```

```
--7 Student_Course (Depends on Student and Course)
CREATE TABLE student_course (
    Student_Course_ID VARCHAR(50) PRIMARY KEY,
    Student_ID VARCHAR(50),
    Course_ID VARCHAR(50),
    createdAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    FOREIGN KEY (Student_ID) REFERENCES student(Student_ID),
    FOREIGN KEY (Course_ID) REFERENCES course(Course_ID)
);
```

Task 1 -QL2

```
SELECT s.Student_ID, s.Name, COUNT(sc.Course_ID) AS Course_Count
FROM student s
JOIN student_course sc ON s.Student_ID = sc.Student_ID
```

```
GROUP BY s.Student_ID, s.Name
HAVING COUNT(sc.Course_ID) > 2;
```

Output:

Student_ID	Name	Course_Count
193-16-461	Alice Johnson	4
193-16-467	Frank Wilson	3
193-16-480	Hannah Martin	4

Show all | Number of rows: 25 Filter rows: Search this table

```
SELECT * FROM teacher
WHERE Email LIKE '%diu%'
AND Phone_Number LIKE '017%';
```

Output:

	Teacher_ID	Name	Room_Number	Floor_Number	Phone_Number	Email	createdAt	updatedAt
Edit Copy Delete	T001	Prof. Adams	101	1	01728170790	adams@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:29:16
Edit Copy Delete	T002	Prof. Baker	102	1	01728170791	baker@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:29:39
Edit Copy Delete	T003	Prof. Carter	103	1	01728170792	carter@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:29:46
Edit Copy Delete	T004	Prof. Davis	201	2	01728170793	davis@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:29:58
Edit Copy Delete	T005	Prof. Evans	202	2	01728170794	evans@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:30:11
Edit Copy Delete	T006	Prof. Foster	203	2	01728170795	foster@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:30:19
Edit Copy Delete	T007	Prof. Green	301	3	01728170796	green@diu.edu.bd	2025-04-04 22:37:40	2025-04-04 23:30:26

Check all With selected: Edit Copy Delete Export

```
SELECT * FROM student
WHERE Address = 'Dhaka' AND CGPA > 3.65;
```

Output:

	Student_ID	Name	Date_of_Birth	Address	Age	CGPA	Faculty_ID	Project_ID	createdAt	updatedAt
Edit Copy Delete	193-16-461	Alice Johnson	2000-03-15	Dhaka	24	3.80	710002880	P001	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-480	Hannah Martin	2000-06-18	Dhaka	24	3.90	710002887	P008	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-483	Jessica Green	2001-04-08	Dhaka	22	3.70	710002889	P010	2025-04-04 22:54:03	2025-04-04 22:54:03

Check all *With selected:* Edit Copy Delete Export

```
SELECT * FROM student
WHERE Address NOT IN ('Dhaka', 'Chittagong');
```

Output:

	Student_ID	Name	Date_of_Birth	Address	Age	CGPA	Faculty_ID	Project_ID	createdAt	updatedAt
Edit Copy Delete	193-16-463	Charlie Brown	2001-01-10	Khulna	22	3.90	710002882	P003	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-464	David White	1998-11-05	Rajshahi	21	3.50	710002883	P004	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-465	Emily Harris	2000-05-30	Sylhet	20	3.70	710002884	P005	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-467	Frank Wilson	1999-09-12	Rangpur	19	3.60	710002885	P006	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-468	Grace Thomas	2001-02-25	Comilla	25	3.80	710002886	P007	2025-04-04 22:54:03	2025-04-04 22:54:03

Check all *With selected:* Edit Copy Delete Export

```
SELECT s.Student_ID, s.Name, sc.Course_ID
FROM student s
JOIN student_course sc ON s.Student_ID = sc.Student_ID
WHERE sc.Course_ID IN ('CIS222', 'CIS223', 'CIS224');
```

Output:

Student_ID	Name	Course_ID
193-16-461	Alice Johnson	CIS222
193-16-462	Bob Smith	CIS223
193-16-463	Charlie Brown	CIS224
193-16-461	Alice Johnson	CIS223
193-16-461	Alice Johnson	CIS224
193-16-480	Hannah Martin	CIS222
193-16-480	Hannah Martin	CIS223
193-16-467	Frank Wilson	CIS222
193-16-467	Frank Wilson	CIS224

```
SELECT * FROM project_thesis
WHERE Name LIKE '%Library%' OR Name LIKE '%AI%';
```

Output:

	Project_ID	Name	createdAt	updatedAt
Edit Copy Delete	P001	AI Chatbot for Education	2025-04-04 22:39:26	2025-04-04 22:39:26
Edit Copy Delete	P002	Blockchain-based Voting System	2025-04-04 22:39:26	2025-04-04 22:39:26
Edit Copy Delete	P005	AI-Powered Healthcare Assistant	2025-04-04 22:39:26	2025-04-04 22:39:26

Check all *With selected:* Edit Copy Delete Export

```
SELECT * FROM student
WHERE Age BETWEEN 20 AND 24
AND CGPA > 3.71;
```

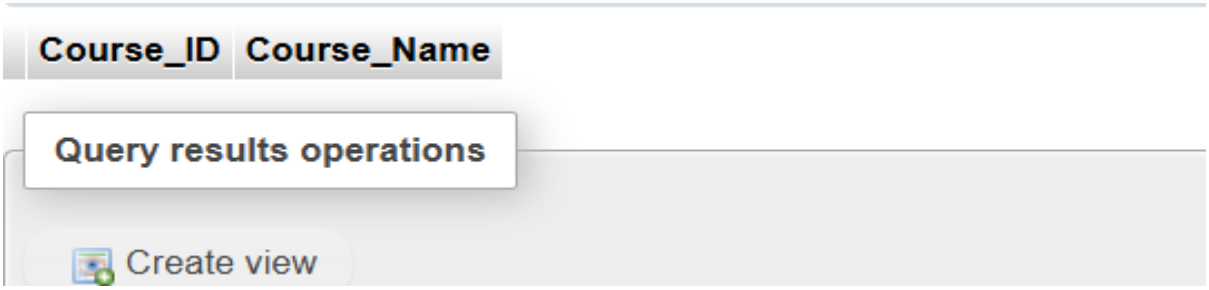
Output:

	Student_ID	Name	Date_of_Birth	Address	Age	CGPA	Faculty_ID	Project_ID	createdAt	updatedAt
Edit Copy Delete	193-16-461	Alice Johnson	2000-03-15	Dhaka	24	3.80	710002880	P001	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-463	Charlie Brown	2001-01-10	Khulna	22	3.90	710002882	P003	2025-04-04 22:54:03	2025-04-04 22:54:03
Edit Copy Delete	193-16-480	Hannah Martin	2000-06-18	Dhaka	24	3.90	710002887	P008	2025-04-04 22:54:03	2025-04-04 22:54:03

Check all *With selected:* Edit Copy Delete Export

```
SELECT c.Course_ID, c.Course_Name
FROM course c
LEFT JOIN student_course sc ON c.Course_ID = sc.Course_ID
WHERE sc.Student_ID IS NULL;
```

Output :




```
SELECT c.Course_ID, c.Course_Name, COUNT(sc.Student_ID) AS
Student_Count
FROM course c
JOIN student_course sc ON c.Course_ID = sc.Course_ID
GROUP BY c.Course_ID, c.Course_Name
HAVING COUNT(sc.Student_ID) > 1;
```

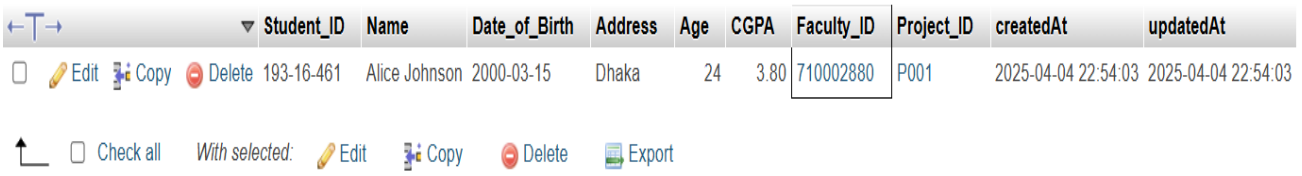
Output:

Course_ID	Course_Name	Student_Count
CIS222	Database Systems	3
CIS223	Computer Networks	3
CIS224	Operating Systems	3
CIS225	Artificial Intelligence	3
CIS226	Software Engineering	3

Show all | Number of rows: 25 Filter rows: Search this table

```
SELECT * FROM student
WHERE Faculty_ID = '710002880';
```

Output:



	Student_ID	Name	Date_of_Birth	Address	Age	CGPA	Faculty_ID	Project_ID	createdAt	updatedAt
Edit Copy Delete	193-16-461	Alice Johnson	2000-03-15	Dhaka	24	3.80	710002880	P001	2025-04-04 22:54:03	2025-04-04 22:54:03

Check all With selected: Edit Copy Delete Export

Task 2 -QL2

Video Link :