**Assignment No: 03**

**University Database Management System**

**By**

**Sachin Gunjal**

**Introduction**

The **University Database Management System (UDBMS)** is an SQL-based project that aims to create a structured and relational database to efficiently manage university-related data.

The system includes multiple interconnected entities such as colleges, departments, professors, and students. By implementing appropriate constraints like **Primary Key**, **Foreign Key**, and **NOT NULL**, this project ensures data accuracy, uniqueness, and consistency across tables. It serves as a foundational project for understanding relational databases and SQL operations in real-world education systems.

**ER Diagram**

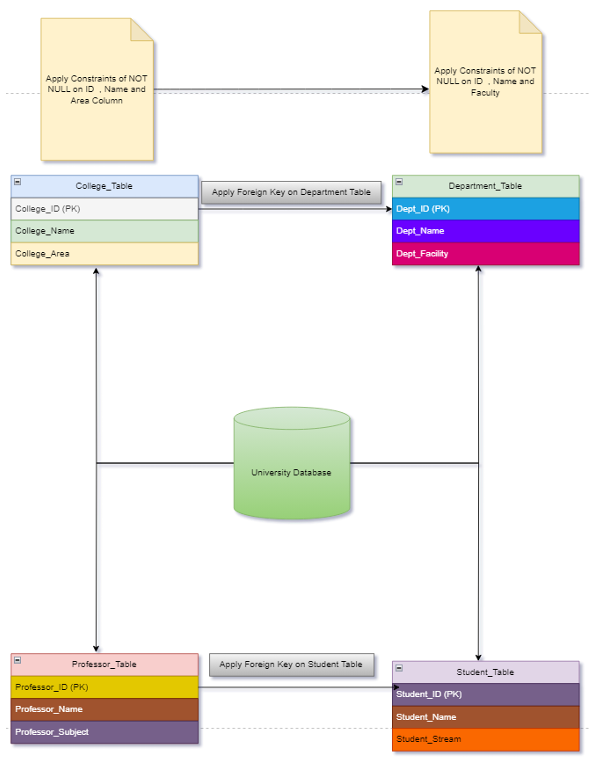
**ER Diagram Explanation**

Refer to the ER diagram provided in the image:

* **College\_Table**
  + College\_ID (PK)
  + College\_Name (NOT NULL)
  + College\_Area (NOT NULL)
* **Department\_Table**
  + Dept\_ID (PK)
  + Dept\_Name (NOT NULL)
  + Dept\_Facility (NOT NULL)
  + **Foreign Key** linked to College\_ID (not shown in the table but mentioned to be added)
* **Professor\_Table**
  + Professor\_ID (PK)
  + Professor\_Name
  + Professor\_Subject
  + (Could link to Department\_ID optionally for normalization)
* **Student\_Table**
  + Student\_ID (PK)
  + Student\_Name
  + Student\_Stream
  + **Foreign Key** linked to Department\_ID (implied by arrow from department)

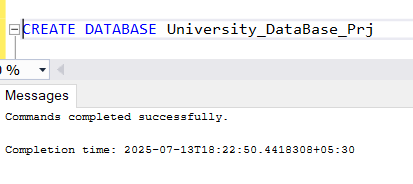
**Key Constraints Applied:**

* Primary keys on ID columns ensure uniqueness.
* NOT NULL on important fields like names and facility info.
* Foreign Keys link dependent tables (e.g., Students → Departments, Departments → Colleges).

****

1. **CREATE DATABASE**

We have created university database for this project to add tables to maintain data integrity and organizing and structuring data in a specific way to enhance its readability and usability.

****

1. **CREATE AND INSERT DATA INTO TABLES**
2. **College\_Table**

* College\_ID(PK)
* College\_Name
* College\_Area

CREATE TABLE College\_Table(

College\_ID INT PRIMARY KEY NOT NULL,

College\_Name VARCHAR(100) NOT NULL,

College\_Area VARCHAR(100) NOT NULL

);

INSERT INTO College\_Table (College\_ID, College\_Name, College\_Area) VALUES

(1, 'Sunrise Institute', 'Pune'),

(2, 'Global Tech University', 'Mumbai'),

(3, 'Skyline College', 'Delhi'),

(4, 'Greenfield University', 'Bangalore'),

(5, 'Northern Heights College', 'Chennai'),

(6, 'Central State University', 'Hyderabad'),

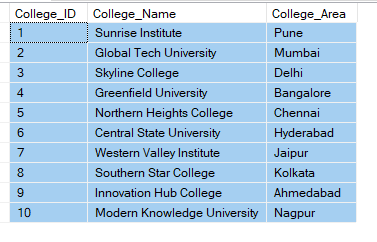
(7, 'Western Valley Institute', 'Jaipur'),

(8, 'Southern Star College', 'Kolkata'),

(9, 'Innovation Hub College', 'Ahmedabad'),

(10, 'Modern Knowledge University', 'Nagpur');

**SELECT \* FROM College\_Table**

****

1. **Department\_Table**

* Dept\_ID(PK)
* Dept\_Name
* Dept\_Facility

**Foreign key on Department key from College\_table**

CREATE TABLE Department\_Table(

Dept\_ID INT PRIMARY KEY NOT NULL,

Dept\_Name VARCHAR(100) NOT NULL,

Dept\_Facility VARCHAR(100) NOT NULL,

College\_ID INT,

FOREIGN KEY (College\_ID) REFERENCES College\_Table(College\_ID)

);

INSERT INTO Department\_Table (Dept\_ID, Dept\_Name, Dept\_Facility, College\_ID) VALUES

(101, 'Computer Science', 'AI Lab', 1),

(102, 'Mechanical', 'Robotics Lab', 2),

(103, 'Electronics', 'Circuit Design Lab', 3),

(104, 'Civil Engineering', 'Structure Lab', 4),

(105, 'Electrical', 'Power Systems Lab', 5),

(106, 'Information Tech', 'Coding Hub', 6),

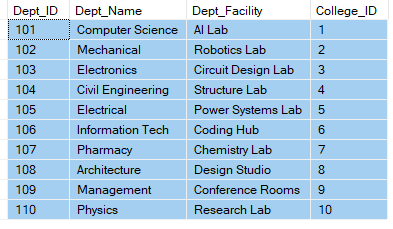
(107, 'Pharmacy', 'Chemistry Lab', 7),

(108, 'Architecture', 'Design Studio', 8),

(109, 'Management', 'Conference Rooms', 9),

(110, 'Physics', 'Research Lab', 10);

**SELECT \* FROM Department\_Table**

****

1. **Professor\_Table**

* Professor\_ID(PK)
* Professor\_Name
* Professor\_Subject

CREATE TABLE Professor\_Table(

Professor\_ID INT PRIMARY KEY NOT NULL,

Professor\_Name VARCHAR(100),

Professor\_Subject VARCHAR(100)

);

INSERT INTO Professor\_Table (Professor\_ID, Professor\_Name, Professor\_Subject) VALUES

(201, 'Dr. Neha Sharma', 'Data Structures'),

(202, 'Dr. Ravi Patil', 'Thermodynamics'),

(203, 'Dr. Ayesha Khan', 'Digital Circuits'),

(204, 'Dr. Karan Mehta', 'Concrete Tech'),

(205, 'Dr. Meena Desai', 'Electrical Machines'),

(206, 'Dr. Sandeep Roy', 'Database Systems'),

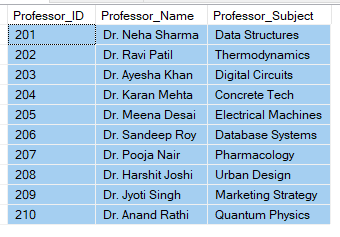
(207, 'Dr. Pooja Nair', 'Pharmacology'),

(208, 'Dr. Harshit Joshi', 'Urban Design'),

(209, 'Dr. Jyoti Singh', 'Marketing Strategy'),

(210, 'Dr. Anand Rathi', 'Quantum Physics');

**SELECT \* FROM Professor\_Table**

****

1. **Student\_Table**

* Student\_ID(PK)
* Student\_Name
* Student\_Stream

**Foreign Key on Student\_Table from Professor\_Table**

CREATE TABLE Student\_Table(

Student\_ID INT PRIMARY KEY NOT NULL,

Student\_Name VARCHAR(100),

Student\_Stream VARCHAR(100),

Professor\_ID INT,

FOREIGN KEY (Professor\_ID) REFERENCES Professor\_Table(Professor\_ID)

);

INSERT INTO Student\_Table (Student\_ID, Student\_Name, Student\_Stream, Professor\_ID) VALUES

(301, 'Amit Rawat', 'B.Tech CS', 201),

(302, 'Sneha More', 'B.Tech ME', 202),

(303, 'Ravi Thakur', 'B.Tech EC', 203),

(304, 'Komal Jadhav', 'B.Tech CE', 204),

(305, 'Farhan Shaikh', 'B.Tech EE', 205),

(306, 'Anjali Reddy', 'B.Tech IT', 206),

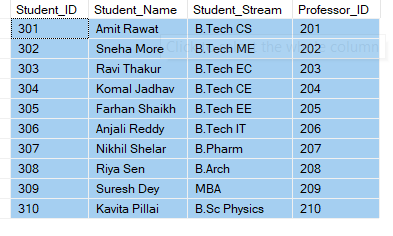
(307, 'Nikhil Shelar', 'B.Pharm', 207),

(308, 'Riya Sen', 'B.Arch', 208),

(309, 'Suresh Dey', 'MBA', 209),

(310, 'Kavita Pillai', 'B.Sc Physics', 210);

SELECT \* FROM STUDENT\_TABLE

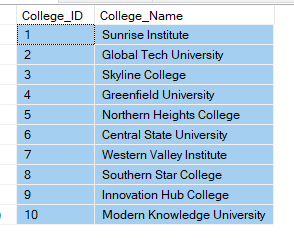
****

We created a relational structure where each student is associated directly with a professor. This establishes a one-to-many relationship, where one professor can be linked to multiple students. By connecting students to professors instead of departments, we focused on the mentorship or academic supervision aspect, which simplifies the relationship and helps track which professor is responsible for which student. This setup ensures clear academic mapping within the database.

1. **Task**

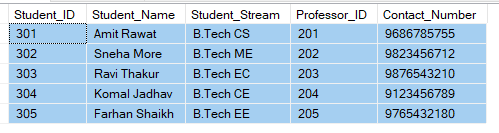
**1.Give the information of College\_ID and College\_name from College\_Table**

SELECT College\_ID, College\_Name FROM College\_Table



**2. Show Top 5 rows from Student table.**

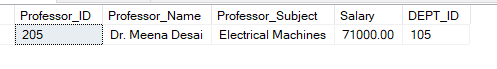
SELECT TOP 5 \* FROM STUDENT\_TABLE



**3. What is the name of professor whose ID is 5**

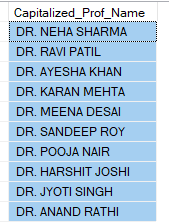
SELECT \* FROM Professor\_Table

WHERE Professor\_ID = 205



1. **Convert the name of the Professor into Upper case**

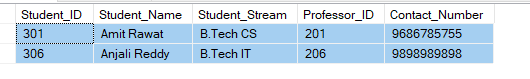
SELECT UPPER(Professor\_Name) as Capitalized\_Prof\_Name FROM Professor\_Table



1. **Show me the names of those students whose name is start with a**

SELECT \* FROM STUDENT\_TABLE

WHERE Student\_Name LIKE 'A%'

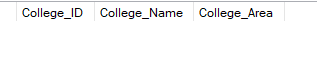


**6. Give the name of those colleges whose end with a**

SELECT \* FROM College\_Table

WHERE College\_Name LIKE '%a'

**NO College name start with A**



1. **Add one Salary Column in Professor\_Table**

ALTER TABLE Professor\_Table

ADD Salary DECIMAL(10,2);

UPDATE Professor\_Table SET SALARY = 75000 WHERE Professor\_ID = 201;

UPDATE Professor\_Table SET SALARY = 72000 WHERE Professor\_ID = 202;

UPDATE Professor\_Table SET SALARY = 70000 WHERE Professor\_ID = 203;

UPDATE Professor\_Table SET SALARY = 73000 WHERE Professor\_ID = 204;

UPDATE Professor\_Table SET SALARY = 71000 WHERE Professor\_ID = 205;

UPDATE Professor\_Table SET SALARY = 76000 WHERE Professor\_ID = 206;

UPDATE Professor\_Table SET SALARY = 69000 WHERE Professor\_ID = 207;

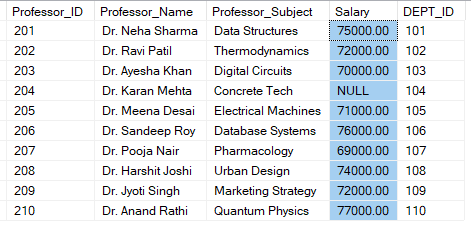
UPDATE Professor\_Table SET SALARY = 74000 WHERE Professor\_ID = 208;

UPDATE Professor\_Table SET SALARY = 72000 WHERE Professor\_ID = 209;

UPDATE Professor\_Table SET SALARY = 77000 WHERE Professor\_ID = 210;

SELECT \* FROM Professor\_Table

Added NULL values in it next queries



**8. Add one Contact Column in Student\_table**

ALTER TABLE STUDENT\_TABLE

ADD Contact\_Number INT

ALTER TABLE STUDENT\_TABLE

ALTER COLUMN CONTACT\_NUMBER BIGINT

UPDATE Student\_Table SET Contact\_Number = 9686785755 WHERE Student\_ID = 301;

UPDATE Student\_Table SET Contact\_Number = 9823456712 WHERE Student\_ID = 302;

UPDATE Student\_Table SET Contact\_Number = 9876543210 WHERE Student\_ID = 303;

UPDATE Student\_Table SET Contact\_Number = 9123456789 WHERE Student\_ID = 304;

UPDATE Student\_Table SET Contact\_Number = 9765432180 WHERE Student\_ID = 305;

UPDATE Student\_Table SET Contact\_Number = 9898989898 WHERE Student\_ID = 306;

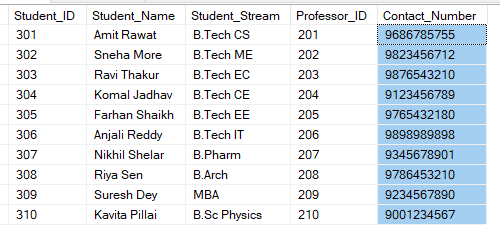
UPDATE Student\_Table SET Contact\_Number = 9345678901 WHERE Student\_ID = 307;

UPDATE Student\_Table SET Contact\_Number = 9786453210 WHERE Student\_ID = 308;

UPDATE Student\_Table SET Contact\_Number = 9234567890 WHERE Student\_ID = 309;

UPDATE Student\_Table SET Contact\_Number = 9001234567 WHERE Student\_ID = 310;

SELECT \* FROM Student\_Table



**9. Find the total Salary of Professor**

SELECT \* FROM Professor\_Table

SELECT SUM(SALARY) AS Total\_Professor\_Salary

FROM Professor\_Table



**10. Change datatype of any one column of any one Table**

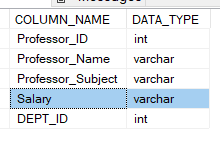
ALTER TABLE PROFESSOR\_TABLE

ALTER COLUMN SALARY VARCHAR(50);

SELECT COLUMN\_NAME, DATA\_TYPE

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = 'Professor\_Table';

****

**4.Task 3**

**1. Show first 5 records from Students table and Professor table Combine**

SELECT TOP 5 ST.Student\_ID, ST.Student\_Name, ST.Student\_Stream, ST.CONTACT\_NUMBER,

PT.Professor\_ID, PT.Professor\_Name, PT.Professor\_Subject, PT.SALARY

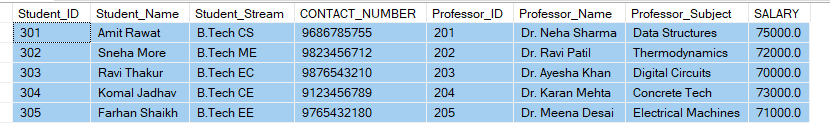
FROM Student\_Table ST

INNER JOIN

Professor\_Table PT

ON

ST.Professor\_ID = PT.Professor\_ID



**2. Apply Inner join on all 4 tables together(Syntax is mandatory)**

**-CREATED Column in Professor\_table**

ALTER TABLE PROFESSOR\_TABLE

ADD DEPT\_ID INT

**-Converted column in foreign kry**

ALTER TABLE Professor\_Table

ADD CONSTRAINT FK\_Professor\_Department

FOREIGN KEY (Dept\_ID)

REFERENCES Department\_Table(Dept\_ID);

**-Added values in column**

UPDATE Professor\_Table SET Dept\_ID = 101 WHERE Professor\_ID = 201;

UPDATE Professor\_Table SET Dept\_ID = 102 WHERE Professor\_ID = 202;

UPDATE Professor\_Table SET Dept\_ID = 103 WHERE Professor\_ID = 203;

UPDATE Professor\_Table SET Dept\_ID = 104 WHERE Professor\_ID = 204;

UPDATE Professor\_Table SET Dept\_ID = 105 WHERE Professor\_ID = 205;

UPDATE Professor\_Table SET Dept\_ID = 106 WHERE Professor\_ID = 206;

UPDATE Professor\_Table SET Dept\_ID = 107 WHERE Professor\_ID = 207;

UPDATE Professor\_Table SET Dept\_ID = 108 WHERE Professor\_ID = 208;

UPDATE Professor\_Table SET Dept\_ID = 109 WHERE Professor\_ID = 209;

UPDATE Professor\_Table SET Dept\_ID = 110 WHERE Professor\_ID = 210;

SELECT CT.College\_Name, DT.Dept\_Name, ST.Student\_Name, PT.Professor\_Name

FROM College\_Table CT

INNER JOIN

Department\_Table DT

ON CT.College\_ID = DT.College\_ID

INNER JOIN

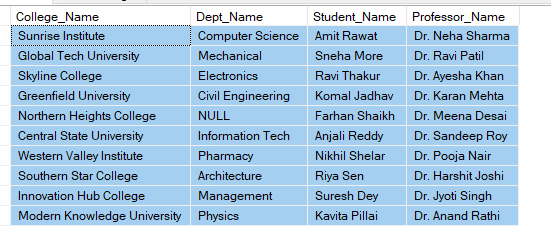
Professor\_Table PT

ON DT.Dept\_ID = PT.Dept\_ID

INNER JOIN

Student\_Table ST

ON ST.Professor\_ID = PT.Professor\_ID;



**3. Show Some null values from Department table and Professor table.**

**Added some NULL values inside columns**

UPDATE Department\_Table

SET Dept\_Name = 'NULL'

WHERE Dept\_ID = 105;

UPDATE Professor\_Table

SET Salary = 'NULL'

WHERE Professor\_ID = 204;

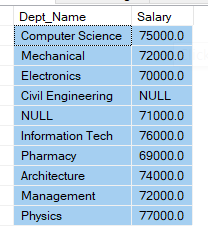
SELECT DT.Dept\_Name, PT.Salary FROM DEPARTMENT\_TABLE DT

LEFT JOIN

Professor\_Table PT

ON

DT.Dept\_ID = PT.DEPT\_ID



**4. Create a View from College Table and give those records whose college name starts with C**

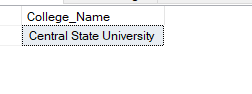
CREATE VIEW CollegeNamesWith\_C

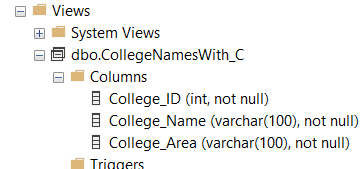
AS

SELECT \* FROM College\_Table

WHERE College\_Name LIKE 'C%'

SELECT College\_Name FROM CollegeNamesWith\_C





**5. Create Stored Procedure of Professor table whatever customer ID will be given by user it should show whole records of it.**

CREATE PROCEDURE Find\_Details

@Cust\_id INT

AS

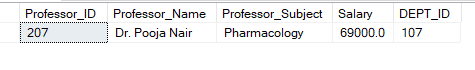
BEGIN

SELECT \* FROM Professor\_Table

WHERE Professor\_ID = @CUST\_ID;

END

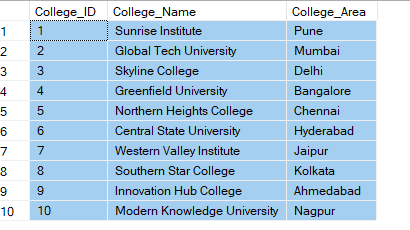
EXEC Find\_Details 207



**6. Rename the College\_Table to College\_Tables\_Data .**

EXEC sp\_rename 'COLLEGE\_TABLE', 'COLLEGE\_TABLES\_DATA';

SELECT \* FROM College\_Tables\_data

****

**5. Conclusion**

The University DBMS project successfully demonstrated how to design and manage a structured relational database using SQL. By creating and linking four core tables—College, Department, Professor, and Student—we built a system that models real-world academic data relationships.

Through this project, we implemented foreign keys, handled null values, performed joins, used stored procedures, and modified schemas using **ALTER**, **UPDATE**, and **CAST** commands. We also encountered and resolved common SQL errors like datatype mismatches and arithmetic overflows, which strengthened understanding of data integrity and error handling.

In conclusion, this project provided hands-on experience in building a scalable and normalized database structure, enhanced query-writing skills, and developed a practical understanding of how relational databases operate in educational institutions or similar domains. It lays a solid foundation for working on larger-scale data analytics, reporting, or software applications.