**‘**

**UNIVERSITY DATABASE MANAGEMENT SYSTEM**

**Topic of the Project**

**Description of the project:**

The main objective of UNIVERSITY MANAGEMENT SYSTEM is to automate all functionalities of the university. Using this system you can manage all University management work like Academic activities, Research projects,

Faculty management and hostel management. Using this University management system, you can view or update data and information about students and faculty easily. This system helps in managing the activity

like student admission, hostel joining, Department information.

The UNIVERSITY MANAGEMENT SYSTEM can be used to store student information like Name, Address, DOB, and Student\_Id, Course etc. User can create report regarding any student any time using this system.

Using this system you can register new student and their course details. You can also add new faculty in the system and can check details of the faculty easily. Student can also check course

detail online from this system.

Using this system you can manage all information of all aspects of a University, its students, faculties, Departments, Research Project and other hostel details.

College management system provides the easiest way to manage all functionalities of a college. This system facilitates colleges to maintain the functionality related to college employees and

their students.

User can also retrieve following data from these tables -

Student Table- This table consists of details about the various details about the student like name , Student\_Id, phone no. , address and age.

Constraint: Student id will be unique for every student.

Department Table- This table consists of details about the various details about the department like Name, Department\_id , Faculty\_id and Student\_id.

Constraint- Department\_id is unique for each department.

Course Table - This table consists of details about the various details about the course like Course\_Id, Name, Credit point, Faculty\_Id, Student\_Id and department.

Constraint- Course\_Id is unique for each department.

Research Project Table - This table consists of details about the faculties who are associated or working on particular project in research domain.

Constraint- Project\_Id is unique for each Student.

Hostel Details - This table consists of details about the various details about the Hostel like Name, Date of joining and Department.

Funding Organization Table - This table consists of the details about the funds given the particular project.

ER Model is the graphical approach to the database design which describes how the database is related to each other.

UNIVERSITY MANAGEMENT SYSTEM:->

The benefits of University management System :-

1) They can easily retrieve all information related to student and faculty.

2) User has all the Collective records of students of all the branches.

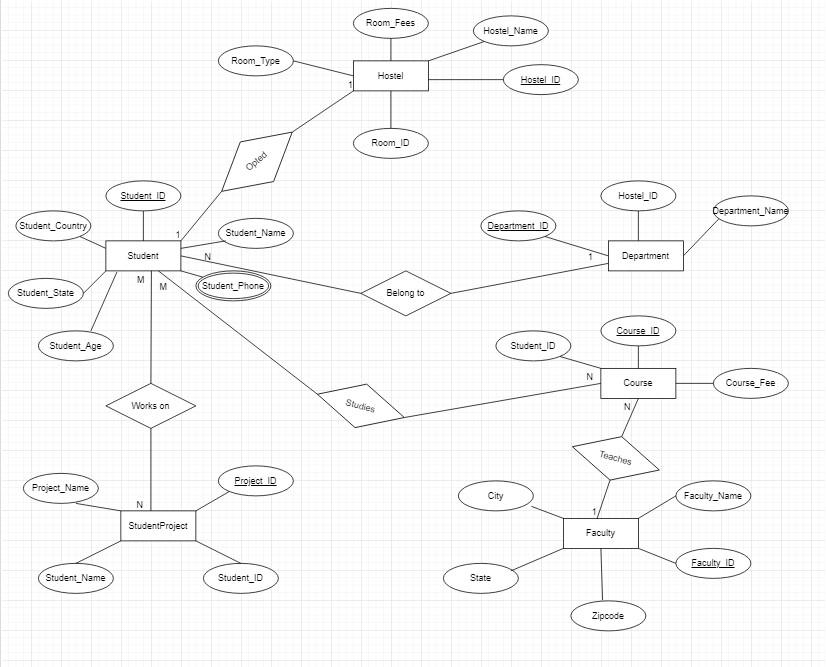
3) User can check all the records of faculties of all departments anytime.

4) This system gives easy approach to find the detail information for any student/faculty.

5) This system is beneficial for both students and faculty as they can get all previous or current information when they need.

Also it can help to maintain the hostel reports of university students in proper way.

**ER Diagram**

****

**Database**

**COMMANDS:**

CREATE TABLE Student(

Student\_Id VARCHAR(10),

Student\_Name VARCHAR(50),

Student\_Phone VARCHAR(50),

Student\_Age INT,

Student\_State VARCHAR(20),

Student\_Country VARCHAR(50)

);

INSERT INTO Student VALUES('S38', 'Sanya', '7206195759 , 7419165944', 18, 'Haryana', 'India');

INSERT INTO Student VALUES('S39', 'Sanya Midha', '7206194757', 18, 'Assam', 'India');

INSERT INTO Student VALUES('S40', 'Sanya Sharma', '7206576823', 19, 'Delhi', 'India');

INSERT INTO Student VALUES('S44', 'Sanyam Mahajan', '7206195757', 17, 'Bihar', 'India');

INSERT INTO Student VALUES('S60', 'Shivakshi', '7206195768', 19, 'Punjab', 'India');

INSERT INTO Student VALUES('S50', 'Sarthak Narang', '7206189045', 20, 'Rajasthan', 'India');

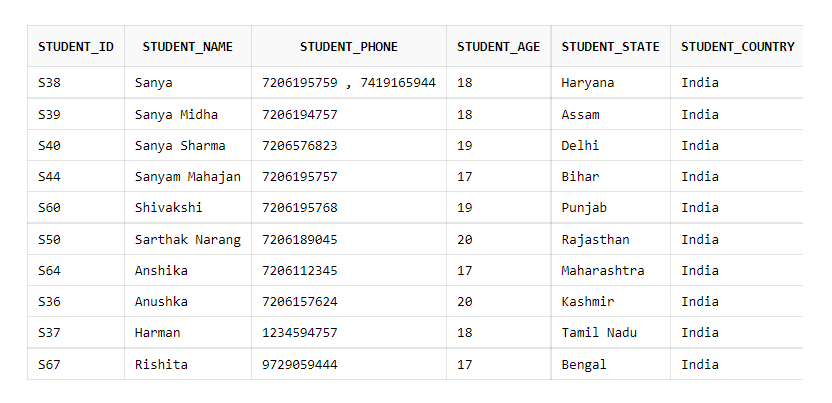
INSERT INTO Student VALUES('S64', 'Anshika', '7206112345', 17, 'Maharashtra', 'India');

INSERT INTO Student VALUES('S36', 'Anushka', '7206157624', 20, 'Kashmir', 'India');

INSERT INTO Student VALUES('S37', 'Harman', '1234594757', 18, 'Tamil Nadu', 'India');

INSERT INTO Student VALUES('S67', 'Rishita', '9729059444', 17, 'Bengal', 'India');

SELECT \* FROM Student;

****

**COMMANDS:**

CREATE TABLE studentproject(

Project\_Id VARCHAR(5),

Student\_Id VARCHAR(5),

Student\_Name VARCHAR(20),

Project\_Name VARCHAR(20)

);

INSERT INTO studentproject VALUES('P09', 'S38', 'Sanya', 'Cloud Development');

INSERT INTO studentproject VALUES('P07', 'S39', 'Sanya Midha', 'Web Development');

INSERT INTO studentproject VALUES('P08', 'S40', 'Sanya Sharma', 'Geo Location');

INSERT INTO studentproject VALUES('P06', 'S44', 'Sanyam Mahajan', 'IOT Devices');

INSERT INTO studentproject VALUES('P05', 'S60', 'Shivakshi', 'AI Fundamentals');

INSERT INTO studentproject VALUES('P04', 'S50', 'Sarthak Narang', 'Fullstack Development');

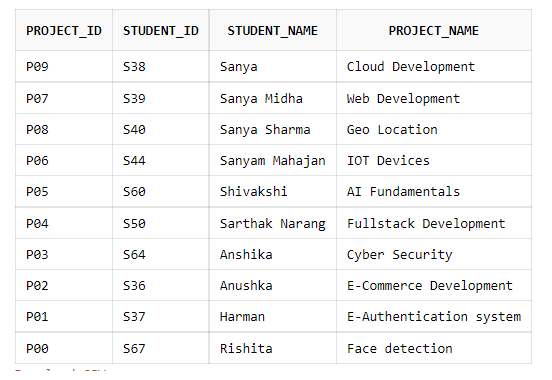
INSERT INTO studentproject VALUES('P03', 'S64', 'Anshika', 'Cyber Security');

INSERT INTO studentproject VALUES('P02', 'S36', 'Anushka', 'E-Commerce Development');

INSERT INTO studentproject VALUES('P01', 'S37', 'Harman', 'E-Authentication system');

INSERT INTO studentproject VALUES('P00', 'S67', 'Rishita', 'Face detection');

SELECT \* FROM studentproject;



**COMMANDS:**

CREATE TABLE Course(

course\_id INT,

stud\_id VARCHAR(

course\_fee INT

);

INSERT INTO Course VALUES(100,'S38',82000);

INSERT INTO Course VALUES(101,'S39',56000);

INSERT INTO Course VALUES(102,'S36',87000);

INSERT INTO Course VALUES(101,'S44',56000);

INSERT INTO Course VALUES(104,'S60',90000);

INSERT INTO Course VALUES(103,'S67',66000);

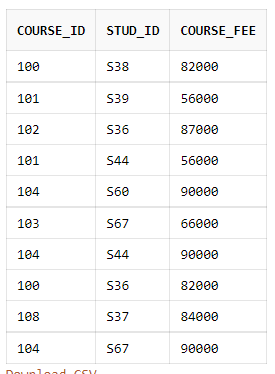
INSERT INTO Course VALUES(104,'S44',90000);

INSERT INTO Course VALUES(100,'S36',82000);

INSERT INTO Course VALUES(108,'S37',84000);

INSERT INTO Course VALUES(104,'S67',90000);

SELECT \* FROM Course;



**COMMANDS:**

CREATE TABLE Hostel(

Hostel\_name VARCHAR(20),

Hostel\_id VARCHAR(20),

Room\_id VARCHAR(20),

Room\_type VARCHAR(50),

Room\_fees INT

);

INSERT INTO Hostel VALUES('Pie','H1', 'H1-A2', 'Twosharing',60000);

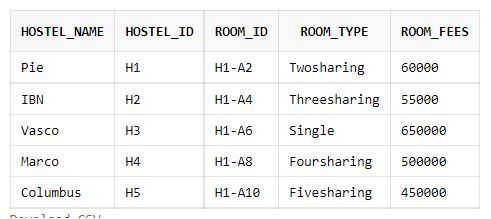
INSERT INTO Hostel VALUES('IBN','H2', 'H1-A4', 'Threesharing',55000);

INSERT INTO Hostel VALUES('Vasco','H3', 'H1-A6', 'Single',650000);

INSERT INTO Hostel VALUES('Marco','H4', 'H1-A8', 'Foursharing',500000);

INSERT INTO Hostel VALUES('Columbus','H5', 'H1-A10', 'Fivesharing',450000);

SELECT \* FROM Hostel;



**COMMANDS:**

CREATE TABLE Department(

Department\_Id INT,

Department\_Name VARCHAR(20),

Student\_Id VARCHAR(5)

);

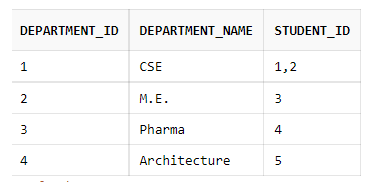
INSERT INTO Department VALUES(001, 'CSE', '1,2');

INSERT INTO Department VALUES(002, 'M.E.', '3');

INSERT INTO Department VALUES(003, 'Pharma', '4');

INSERT INTO Department VALUES(004, 'Architecture', '5');

SELECT \* FROM Department;



**COMMAND:**

CREATE TABLE Faculty(

Faculty\_name VARCHAR(30),

Faculty\_id VARCHAR(20),

zipcode INT,

state VARCHAR(50)

);

INSERT INTO Faculty VALUES('Reena','F1', 132103, 'Assam');

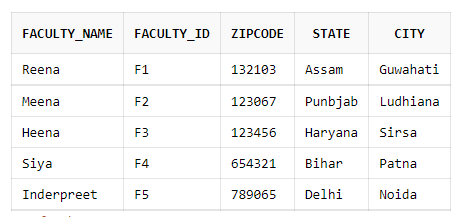
INSERT INTO Faculty VALUES('Meena','F2', 123067, 'Punbjab');

INSERT INTO Faculty VALUES('Heena','F3', 123456, 'Haryana');

INSERT INTO Faculty VALUES('Siya','F4', 654321, 'Bihar');

INSERT INTO Faculty VALUES('Inderpreet','F5', 789065, 'Delhi');

SELECT \* FROM Faculty;



**Normalization**

**Normalization of Student Table:**

**Functional Dependencies:**

1. Student\_Id -> Student\_Name
2. Student\_Id -> Student\_State
3. Student\_State -> Student\_Country
4. Student\_Id-> Student\_Age

**Candidate Key:**

{Student\_Id}

* As the Student table has multi-valued attribute Student\_phone, So it violates the rule of 1nf. So we will convert that multivalued attribute to an atomic-valued attribute.
* As the table is already in 1nf and t no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table. So the table is in 2nf.
* As the table is in 2nf and

Student\_Id -> Student\_State and Student\_State -> Student\_Country are true.

So Student\_Country is transitively dependent on Student\_Id. It violates

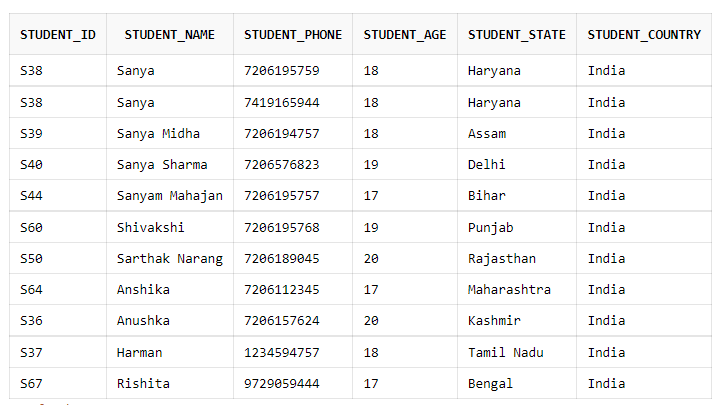
the rules of 3nf.

To convert it in 3nf, we will decompose the relation

Student(Student\_Id, Student\_Name, Student\_Phone, Student\_Age, Student\_State, Student\_country) as:

Student1 (Student\_Id, Student\_Name, Student\_Phone, Student\_State, Student\_Age)

Student2 (Student\_State, Student\_Country)

****

STUDENT 1

STUDENT 2

**NORMALIZATION OF COURSE TABE:**

Course\_Fee cannot alone decide the value of Course\_Id or Student\_Id,  
Course\_Fee together with Student\_Id cannot decide the value of Course\_Id;  
Course\_Fee together with Course\_Id cannot decide the value of Student\_Id;  
Hence,  
Course\_Fee would be a non-prime attribute, as it does not belong to the candidate key.

**CANDIDATE KEY:**

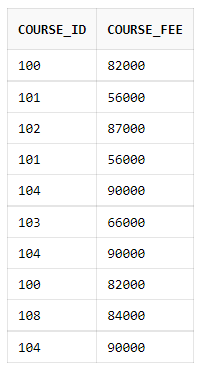
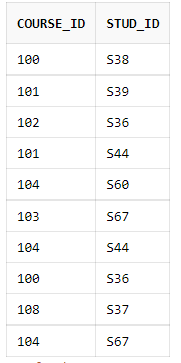
{Student\_Id, Course\_Id} ;

* As there is no multivalued attribute in the course table, So the table is in 1NF.
* But, Course\_Id -> Course\_Fee, i.e., Course\_Fee is dependent on Course\_Id, which is a proper subset of the candidate key. Non-prime attribute Couse\_Fee is dependent on a proper subset of the candidate key, which is a partial dependency and so this relation is not in 2NF.

To convert the above table to 2NF,we need to decompose the table into two tables such as :  
Course 1: Student\_Id, Course\_Id  
Course 2: Course\_Id, Course\_Fee

Course 1

Course 2



**NORMALIZATION OF FACULTY TABLE**:

**CANDIDATE KEY:**

{Faculty\_Id}

* As there is not any multivalued attribute in faculty table, therefore it is in 1NF.
* As the table is already in 1NF and no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table. So the table is in 2NF.
* Faculty\_State, Faculty\_City -> Faculty\_Zip

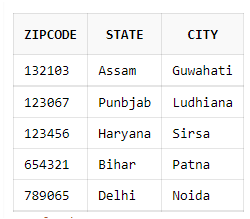
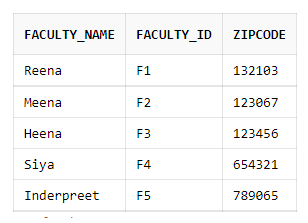
Faculty\_Zip-> Faculty\_Id

So the non-prime attributes (Faculty\_State, Faculty\_City) transitively depend on super key (Faculty\_Id), therefore it violates the rule of 3NF.

To convert the above table to 3NF,we need to decompose the table into two tables such as :  
Faculty1: Faculty\_Id, Faculty\_name, Faculty\_Zip  
Faculty2: Faculty\_Zip, Faculty\_State, Faculty\_City

Faculty1

Faculty2



NORMALIZATION OF PROJECT TABLE:  
CANDIDATE KEY:  
{Student\_Id, Project\_Id}

* As the table does not has any multivalued attribute, therefore the table is in 1NF.
* AS

Student\_Id -> Student\_Name

Project\_Id -> Project\_Name

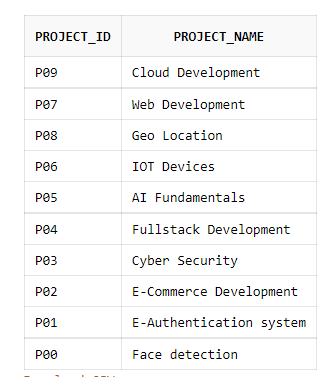
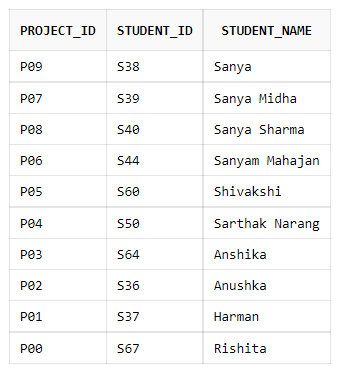
Which make the relation partial dependent

Therefore, the project table violates the rule of 2NF.

To convert the above table to 2NF,we need to decompose the table into two tables such as :  
Project1: Student\_Id, Project\_Id, Student\_Name  
Project2: Project\_Id, Project\_Name

Project1

Project2



NORMALIZATION OF HOSTEL TABLE:

CANDIDATE KEY:

{Hostel\_Id}

* As there is no multivalued attribute in the table, So the table is in 1NF.
* As the table is in 1NF and no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table. So the table is in 2NF.
* Hostle\_Id -> Room\_Id

Room\_Id -> Room\_type

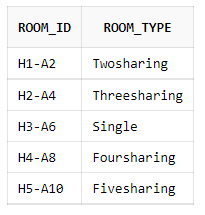
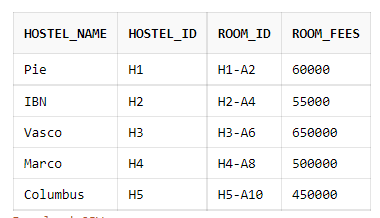
Therefore, Hostle\_Id -> Room\_Type

As there exist a transitive functional dependency, therefore it violates the rule of 3NF.

To convert the above table to 3NF,we need to decompose the table into two tables such as :  
Hostel1: Hostel\_Id, Room\_Id, Room\_Fees  
Hostel2: Room\_Id, Room\_Type

Hostel1

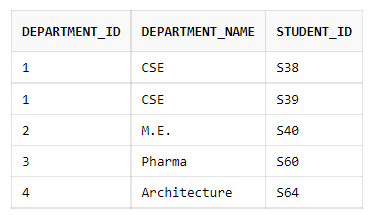
Hostel2



NORMALIZATION IN DEPARTMENT TABLE:

* AS there exist a multivalued attribute name Student\_Id, So we need to convert it into atomic valued attribute.

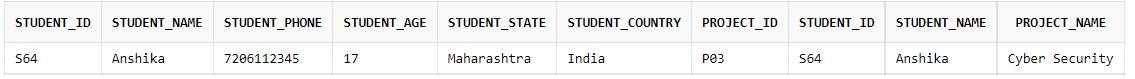
Department1



**Design 8-10 Queries**

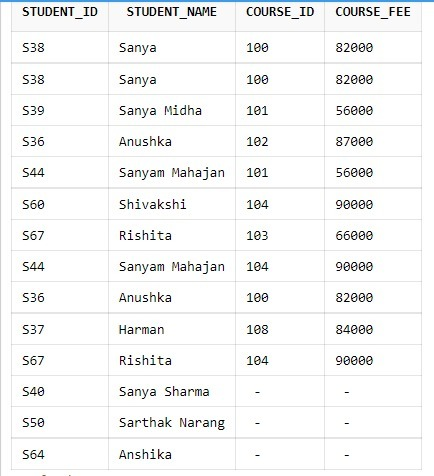
Query 1: Display all the details of those students whose project name is cyber Security.

SELECT \* FROM Student inner join Studentproject on Student.Student\_Id =Studentproject.Student\_Id WHERE Project\_Name = 'Cyber Security';



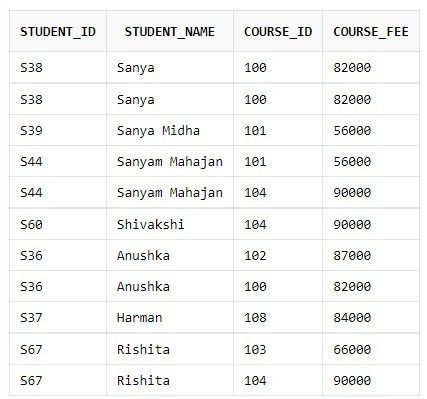
Query 2: Display Student id , Name, Course Id and course fee irrespective of the fact that their course fee is present or not in the course table.

SELECT Student\_Id, Student\_Name, course\_id, course\_fee from Student left outer join Course on Student.Student\_Id = Course.stud\_id;



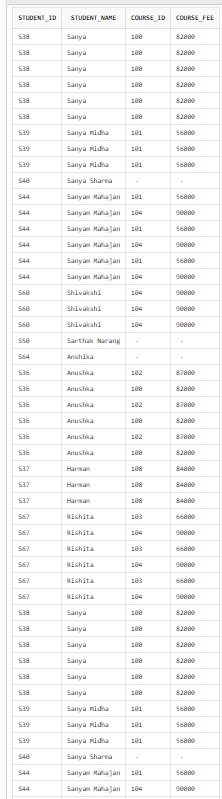
Query3: Display student\_id, name , course\_id and course\_fee irrespective of the fact that their names are present on or not in the ‘student\_table’.

SELECT Student\_Id, Student\_Name, course\_id, course\_fee from Student right outer join Course on Student.Student\_Id = Course.stud\_id;



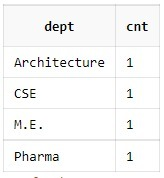
Query4: Display the student\_id, name, course id, course fee of student irrespective of the fact that their respective data is available in the respective tables

SELECT Student\_Id, Student\_Name, course\_id, course\_fee from Student full outer join Course on Student.Student\_Id = Course.stud\_id;



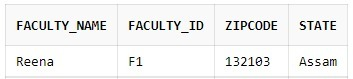
Query5: Display all the Department names as ‘dept’ followed by the count of students as ‘int’ in that particular dept.

SELECT Department\_Name as "dept", count(Department\_Id) as "cnt" from Department group by Department\_Name;



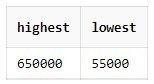
Query6: Display all the faculty details whose state is Assam

SELECT \* FROM Faculty where state = 'Assam';



Query7: Display the highest room fee as ‘highest’ and the lowest in hostel table

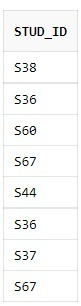
SELECT max(Room\_fees) as "highest" , min(Room\_fees) as "lowest" from Hostel;



.

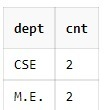
Query8: Display stud\_id of students whose course\_fee is greater than 60,000.

SELECT stud\_id from Course where course\_fee > 60000



Query 9: Display the department name as ‘dept’ followed by the count of students of the ‘CSE’ and ‘ME’ department as ‘int’

Select Department\_Name as "dept", count(Department\_Id) as "cnt" from Department group by Department\_Name having Department\_Name = 'CSE' or Department\_Name = 'M.E.';



Query10: Display Projectname of all the students whose project id is P09

SELECT Project\_Name FROM studentproject where Project\_Id='P09';

