College Administration Management System

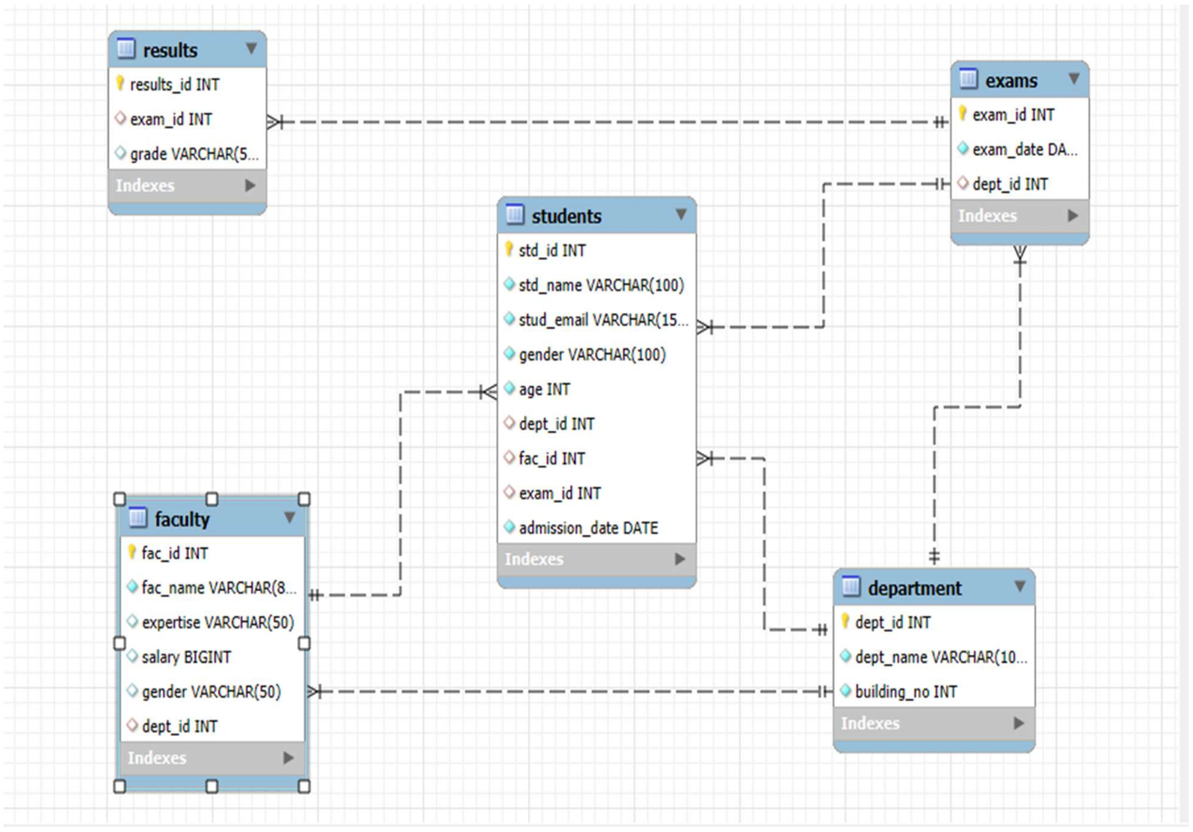
Name : VAISHNAV NAIR

INTRODUCTION

The College Administration Management System (CAMS) is an SQL-based project developed to simplify and modernize the management of academic and administrative activities within a college. Managing student records, faculty details, departmental structures, examinations, and results can become highly complex when handled manually or through fragmented tools such as spreadsheets. CAMS addresses this challenge by implementing a relational database model that integrates all these components into a single, organized system.

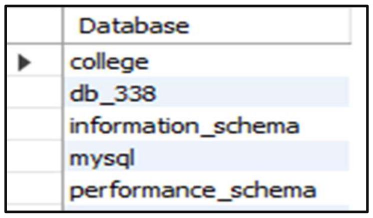
The project leverages the capabilities of a Relational Database Management System (RDBMS) to ensure data consistency, integrity, and accessibility. Through well-defined relationships among entities such as students, faculty, departments, exams, and results, the system eliminates redundancy and provides a reliable framework for data storage and retrieval. Students' details, including their personal information, departmental association, faculty mentors, and exam participation, are securely maintained, while faculty records capture expertise, roles, and departmental assignments. Similarly, the system manages exam scheduling and links results directly to student performance, ensuring transparency and accuracy.

From a technical perspective, the project includes the creation of relational tables, views, and queries that streamline data management. The project leverages SQL to perform data manipulation and query operations such as filtering, sorting, aggregating, and combining data to gain insights. It also employs advanced SQL concepts like subqueries, joins, and window functions to enhance the functionality of the database.


results
results_id INT
exam_id INT
grade VARCHAR(5...
Indexes
exams
exam_id INT
exam_date DA...
dept_id INT
Indexes
students
std_id INT
std_name VARCHAR(100)
stud_email VARCHAR(15...
gender VARCHAR(100)
age INT
dept_id INT
fac_id INT
exam_id INT
admission_date DATE
Indexes
faculty
fac_id INT
fac_name VARCHAR(8...
expertise VARCHAR(50)
salary BIGINT
gender VARCHAR(50)
dept_id INT
Indexes
department
dept_id INT
dept_name VARCHAR(10...
building_no INT
Indexes

Databases :

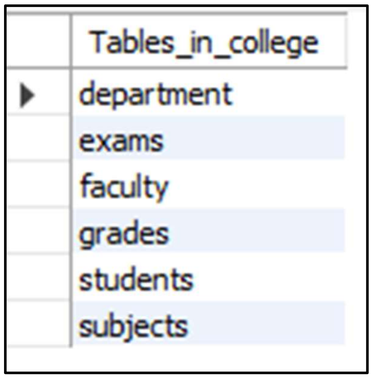
```
CREATE DATABASE college;
USE college;
SHOW DATABASES;
```



```
Database
college
db_338
information_schema
mysql
performance_schema
```

Tables in College Database :

```
SHOW TABLES;
```



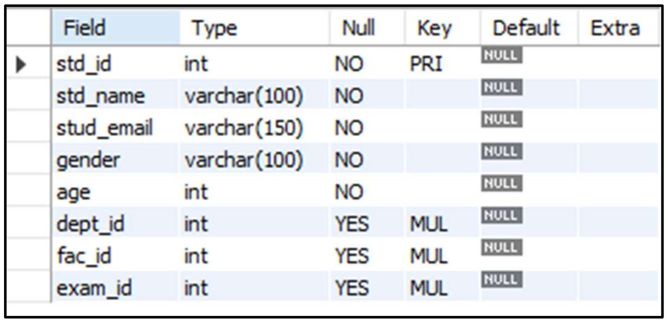
```
Tables_in_college
department
exams
faculty
grades
students
subjects
```

1.DATA DEFINITION LANGUAGE (DDL):

1. Creating Tables:

A) Students

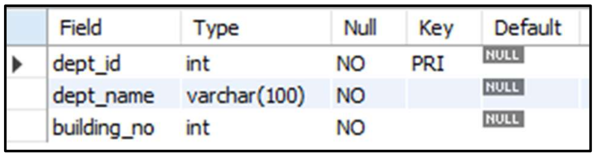
```
CREATE TABLE Students(std_id INT PRIMARY KEY, std_name
 VARCHAR(100) NOT NULL, stud_email VARCHAR(150) NOT
 NULL, gender VARCHAR(100) NOT NULL, age INT NOT
 NULL CHECK(age>18), dept_id INT, fac_id INT, exam_id INT,
 FOREIGN KEY (dept_id) REFERENCES
 Department(dept_id),FOREIGN KEY (exam_id) REFERENCES
 Exams(exam_id), FOREIGN KEY (fac_id) REFERENCES
 Faculty(fac_id));
DESC students;
```



Field	Type	Null	Key	Default	Extra
std_id	int	NO	PRI	NULL	
std_name	varchar(100)	NO		NULL	
stud_email	varchar(150)	NO		NULL	
gender	varchar(100)	NO		NULL	
age	int	NO		NULL	
dept_id	int	YES	MUL	NULL	
fac_id	int	YES	MUL	NULL	
exam_id	int	YES	MUL	NULL	

B) Department

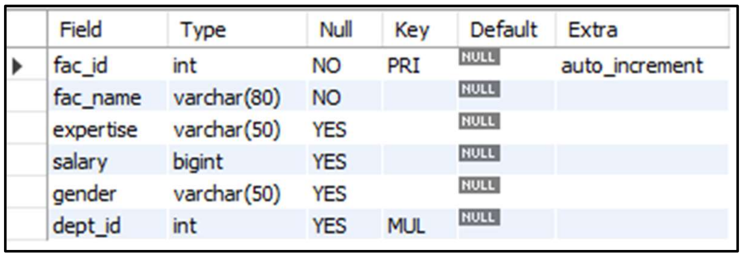
```
CREATE TABLE Department(dept_id INT PRIMARY KEY, dept_name VARCHAR(100)
NOT NULL, building_no INT NOT NULL);
DESC Department;
```



Field	Type	Null	Key	Default
dept_id	int	NO	PRI	NULL
dept_name	varchar(100)	NO		NULL
building_no	int	NO		NULL

C) Faculty

```
CREATE TABLE Faculty(fac_id INT AUTO_INCREMENT PRIMARY KEY, fac_name
VARCHAR(80) NOT NULL, expertise VARCHAR(50), salary BIGINT ,gender VARCHAR(50),
dept_id INT, FOREIGN KEY (dept_id) REFERENCES Department(dept_id));
DESC Faculty;
```

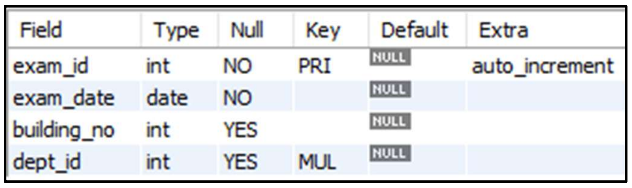


Field	Type	Null	Key	Default	Extra
fac_id	int	NO	PRI	NULL	auto_increment
fac_name	varchar(80)	NO		NULL	
expertise	varchar(50)	YES		NULL	
salary	bigint	YES		NULL	
gender	varchar(50)	YES		NULL	
dept_id	int	YES	MUL	NULL	

D)Exams

CREATE TABLE Exams(exam_id INT AUTO_INCREMENT PRIMARY KEY, exam_date DATE NOT NULL, building_no INT, dept_id INT, FOREIGN KEY (dept_id) REFERENCES Department(dept_id));

DESC Exams;

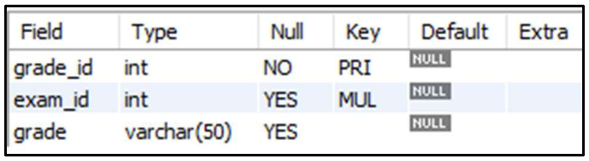


Field	Type	Null	Key	Default	Extra
exam_id	int	NO	PRI	NULL	auto_increment
exam_date	date	NO		NULL	
building_no	int	YES		NULL	
dept_id	int	YES	MUL	NULL	

E) Grades

CREATE TABLE Grades(grade_id INT PRIMARY KEY, exam_id INT, grade VARCHAR(50), FOREIGN KEY (exam_id) REFERENCES Exams(exam_id));

DESC Grades;

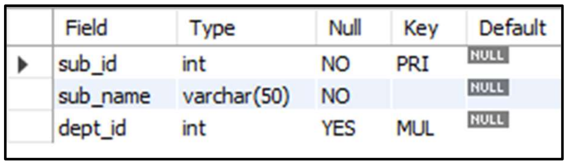


Field	Type	Null	Key	Default	Extra
grade_id	int	NO	PRI	NULL	
exam_id	int	YES	MUL	NULL	
grade	varchar(50)	YES		NULL	

F) Subjects

CREATE TABLE Subjects(sub_id INT PRIMARY KEY, sub_name VARCHAR(50) NOT NULL, dept_id INT, FOREIGN KEY (dept_id) REFERENCES Department(dept_id));

DESC subjects;

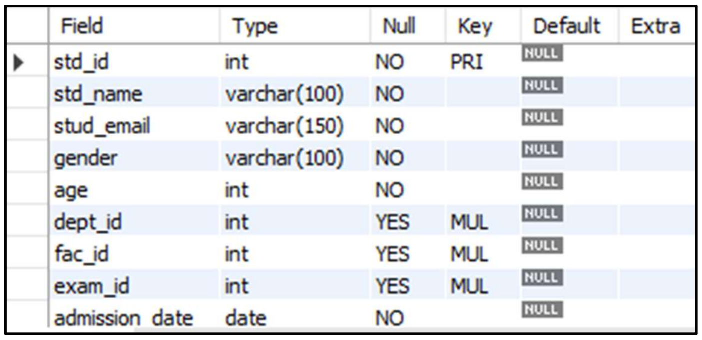


Field	Type	Null	Key	Default
sub_id	int	NO	PRI	NULL
sub_name	varchar(50)	NO		NULL
dept_id	int	YES	MUL	NULL

2.Alter table :

- Alter Table : Add column

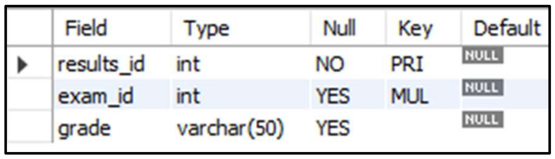
ALTER TABLE Students ADD COLUMN admission_date DATE NOT NULL;



Field	Type	Null	Key	Default	Extra
std_id	int	NO	PRI	NULL	
std_name	varchar(100)	NO		NULL	
stud_email	varchar(150)	NO		NULL	
gender	varchar(100)	NO		NULL	
age	int	NO		NULL	
dept_id	int	YES	MUL	NULL	
fac_id	int	YES	MUL	NULL	
exam_id	int	YES	MUL	NULL	
admission_date	date	NO		NULL	

- Alter Table : Change Column

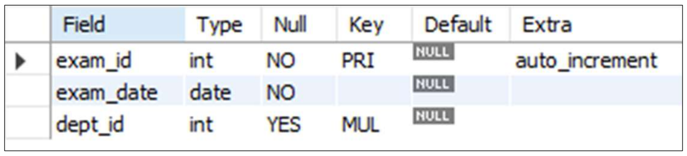
ALTER TABLE Grades CHANGE grade_id results_id INT;



Field	Type	Null	Key	Default
results_id	int	NO	PRI	NULL
exam_id	int	YES	MUL	NULL
grade	varchar(50)	YES		NULL

- Alter Table : Drop column

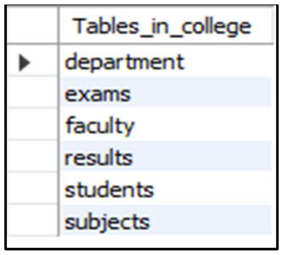
ALTER TABLE Exams DROP COLUMN building_no;



Field	Type	Null	Key	Default	Extra
exam_id	int	NO	PRI	NULL	auto_increment
exam_date	date	NO		NULL	
dept_id	int	YES	MUL	NULL	

- Alter Table : Rename table

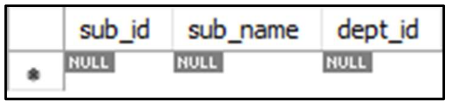
ALTER TABLE Grades RENAME Results;



Tables_in_college
department
exams
faculty
results
students
subjects

3.Truncate table :

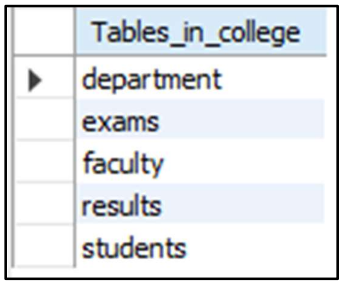
```
TRUNCATE Subjects;
```



	sub_id	sub_name	dept_id
*	NULL	NULL	NULL

4.Drop Table :

```
DROP TABLE Subjects;
```

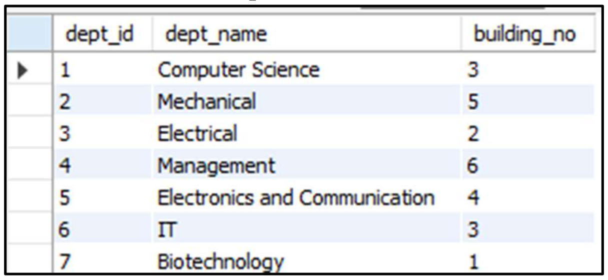


	Tables_in_college
▶	department
	exams
	faculty
	results
	students

2.DATA MANIPULATION LANGUAGE (DML):

1. Insert into table :

```
INSERT INTO Department VALUES (5, "Electronics and Communication", 4);
SELECT * FROM Department;
```

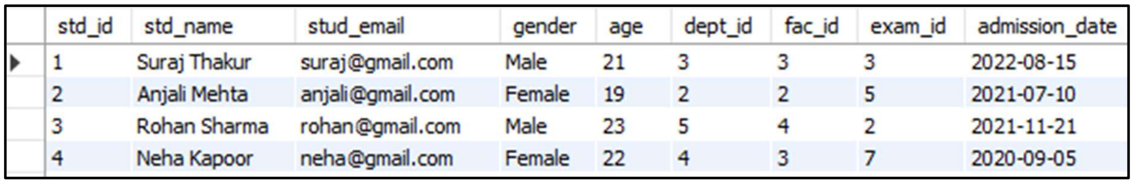


dept_id	dept_name	building_no
1	Computer Science	3
2	Mechanical	5
3	Electrical	2
4	Management	6
5	Electronics and Communication	4
6	IT	3
7	Biotechnology	1

2. Update into Table :

Q. Update student age and admission date.

```
UPDATE Students SET age=23,
admission_date='2021-11-21' WHERE
std_id=3;
```

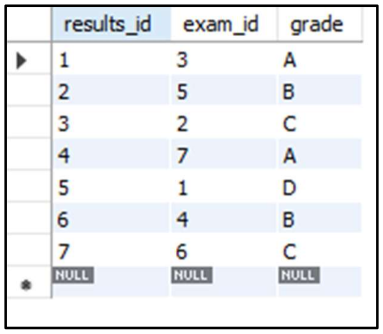


std_id	std_name	stud_email	gender	age	dept_id	fac_id	exam_id	admission_date
1	Suraj Thakur	suraj@gmail.com	Male	21	3	3	3	2022-08-15
2	Anjali Mehta	anjali@gmail.com	Female	19	2	2	5	2021-07-10
3	Rohan Sharma	rohan@gmail.com	Male	23	5	4	2	2021-11-21
4	Neha Kapoor	neha@gmail.com	Female	22	4	3	7	2020-09-05

3. Delete from table :

Q.Delete record having results_id 8.

```
DELETE FROM results WHERE results_id=8;
```



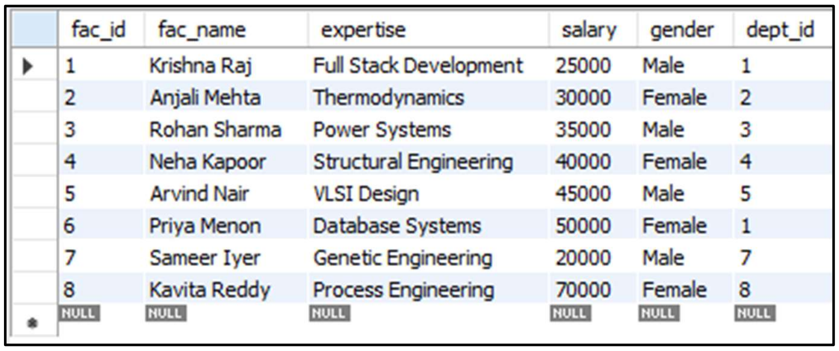
results_id	exam_id	grade
1	3	A
2	5	B
3	2	C
4	7	A
5	1	D
6	4	B
7	6	C
NULL	NULL	NULL

3.DATA QUERY LANGUAGE (DQL) :

1.Select Query:

a) Select Query for entire data.

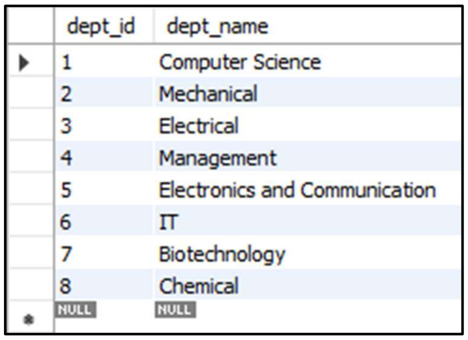
SELECT * FROM Faculty;



fac_id	fac_name	expertise	salary	gender	dept_id
1	Krishna Raj	Full Stack Development	25000	Male	1
2	Anjali Mehta	Thermodynamics	30000	Female	2
3	Rohan Sharma	Power Systems	35000	Male	3
4	Neha Kapoor	Structural Engineering	40000	Female	4
5	Arvind Nair	VLSI Design	45000	Male	5
6	Priya Menon	Database Systems	50000	Female	1
7	Sameer Iyer	Genetic Engineering	20000	Male	7
8	Kavita Reddy	Process Engineering	70000	Female	8
NULL	NULL	NULL	NULL	NULL	NULL

b) Select specific data from table.

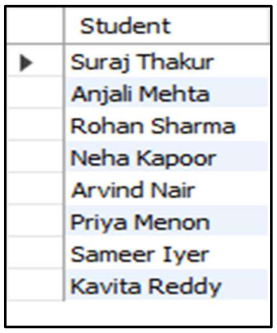
SELECT dept_id,dept_name FROM Department;



dept_id	dept_name
1	Computer Science
2	Mechanical
3	Electrical
4	Management
5	Electronics and Communication
6	IT
7	Biotechnology
8	Chemical
NULL	NULL

c)Select query with an alias as Column name.

SELECT std_name AS Student FROM Students;

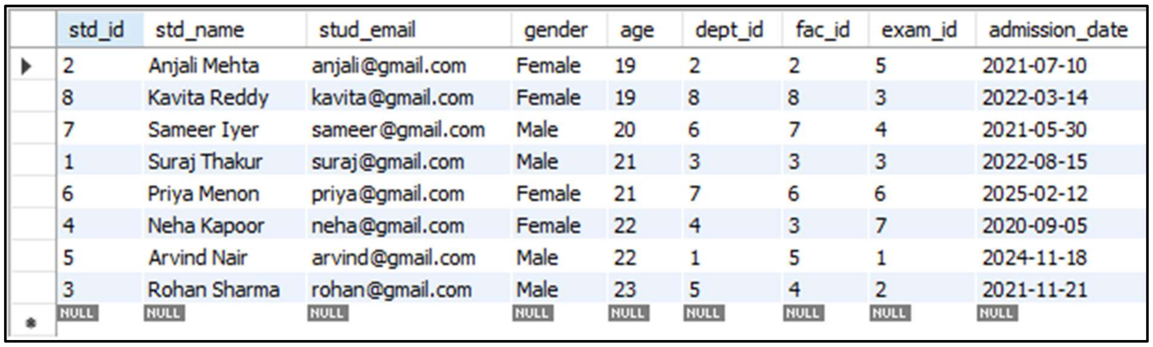


Student
Suraj Thakur
Anjali Mehta
Rohan Sharma
Neha Kapoor
Arvind Nair
Priya Menon
Sameer Iyer
Kavita Reddy

2.ORDER BY

a)List of students in ascending order by age.

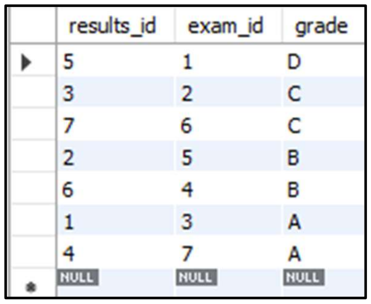
SELECT * FROM Students ORDER BY age;



std_id	std_name	stud_email	gender	age	dept_id	fac_id	exam_id	admission_date
2	Anjali Mehta	anjali@gmail.com	Female	19	2	2	5	2021-07-10
8	Kavita Reddy	kavita@gmail.com	Female	19	8	8	3	2022-03-14
7	Sameer Iyer	sameer@gmail.com	Male	20	6	7	4	2021-05-30
1	Suraj Thakur	suraj@gmail.com	Male	21	3	3	3	2022-08-15
6	Priya Menon	priya@gmail.com	Female	21	7	6	6	2025-02-12
4	Neha Kapoor	neha@gmail.com	Female	22	4	3	7	2020-09-05
5	Arvind Nair	arvind@gmail.com	Male	22	1	5	1	2024-11-18
3	Rohan Sharma	rohan@gmail.com	Male	23	5	4	2	2021-11-21
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

b)List of results in descending order by grade.

SELECT * FROM Results ORDER BY Grade DESC;

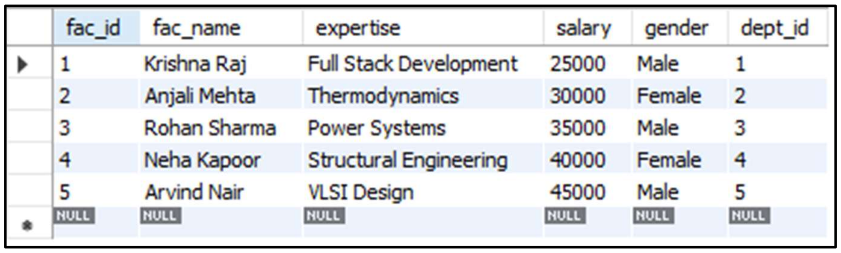


results_id	exam_id	grade
5	1	D
3	2	C
7	6	C
2	5	B
6	4	B
1	3	A
4	7	A
NULL	NULL	NULL

3.LIMIT QUERY

Display 5 faculty members.

SELECT * FROM Faculty LIMIT 5;

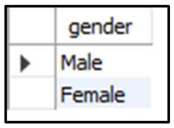


fac_id	fac_name	expertise	salary	gender	dept_id
1	Krishna Raj	Full Stack Development	25000	Male	1
2	Anjali Mehta	Thermodynamics	30000	Female	2
3	Rohan Sharma	Power Systems	35000	Male	3
4	Neha Kapoor	Structural Engineering	40000	Female	4
5	Arvind Nair	VLSI Design	45000	Male	5
NULL	NULL	NULL	NULL	NULL	NULL

4.DISTINCT QUERY

Display Unique Gender from Faculty.

SELECT DISTINCT gender FROM Faculty;



gender
Male
Female

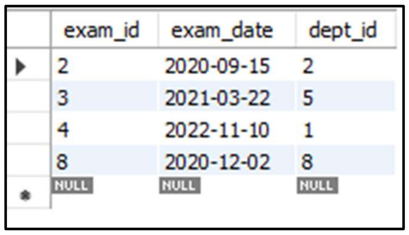
5.WHERE BY CLAUSE:

1)With Comparison Operator

Find exams conducted on before February 02, 2023.

SELECT * FROM Exams WHERE

exam_date < '2023-02-02';



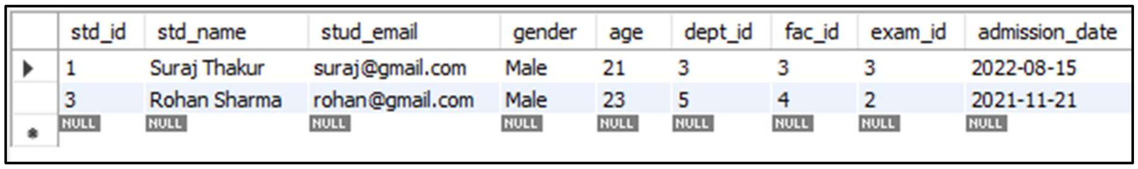
	exam_id	exam_date	dept_id
▶	2	2020-09-15	2
	3	2021-03-22	5
	4	2022-11-10	1
	8	2020-12-02	8
*	NULL	NULL	NULL

2)Logical Operator

- Using AND/OR Operator

Find students who whose age is less than 23 and have dept_id=3 or dept_id=5.

SELECT * FROM Students WHERE age < 24 AND dept_id = 3 OR dept_id=5;

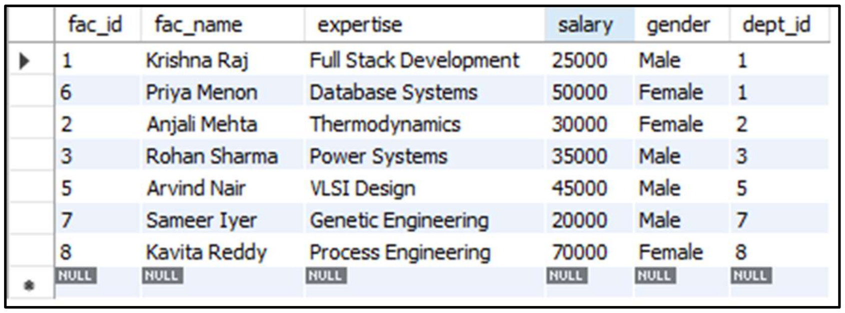


	std_id	std_name	stud_email	gender	age	dept_id	fac_id	exam_id	admission_date
▶	1	Suraj Thakur	suraj@gmail.com	Male	21	3	3	3	2022-08-15
	3	Rohan Sharma	rohan@gmail.com	Male	23	5	4	2	2021-11-21
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- Using NOT Operator

Find faculty members who do not belong to department 4.

SELECT * FROM Faculty WHERE dept_id != 4;

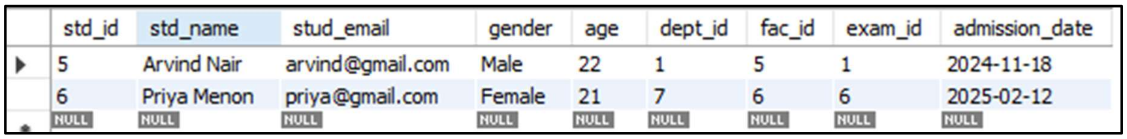


	fac_id	fac_name	expertise	salary	gender	dept_id
▶	1	Krishna Raj	Full Stack Development	25000	Male	1
	6	Priya Menon	Database Systems	50000	Female	1
	2	Anjali Mehta	Thermodynamics	30000	Female	2
	3	Rohan Sharma	Power Systems	35000	Male	3
	5	Arvind Nair	VLSI Design	45000	Male	5
	7	Sameer Iyer	Genetic Engineering	20000	Male	7
	8	Kavita Reddy	Process Engineering	70000	Female	8
*	NULL	NULL	NULL	NULL	NULL	NULL

□ Using BETWEEN operator

Find students with admission date between 1st Jan 2024 and 31st Dec 2025.

```
SELECT * FROM Students WHERE admission_date BETWEEN
'2024-01-01' AND '2025-12-31';
```

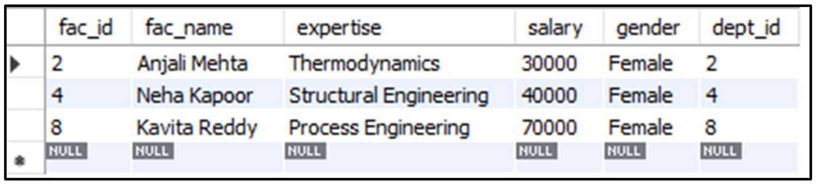


std_id	std_name	stud_email	gender	age	dept_id	fac_id	exam_id	admission_date
5	Arvind Nair	arvind@gmail.com	Male	22	1	5	1	2024-11-18
6	Priya Menon	priya@gmail.com	Female	21	7	6	6	2025-02-12
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

□ Using IN operator

Find faculty members whose dept_id belong in either 2,4,6 or 8.

```
SELECT * FROM Faculty WHERE dept_id IN(2,4,6,8);
```

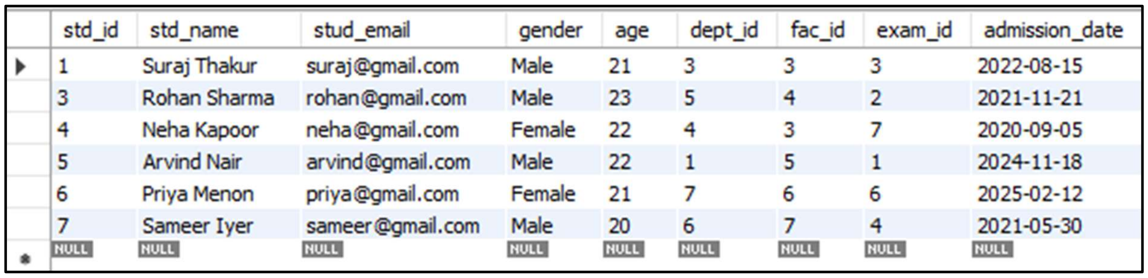


fac_id	fac_name	expertise	salary	gender	dept_id
2	Anjali Mehta	Thermodynamics	30000	Female	2
4	Neha Kapoor	Structural Engineering	40000	Female	4
8	Kavita Reddy	Process Engineering	70000	Female	8
NULL	NULL	NULL	NULL	NULL	NULL

□ Using ANY operator

Find student details for someone whose age is higher than a student whose age is 19.

```
SELECT * FROM Students WHERE age > ANY(SELECT age FROM Students
WHERE age = 19);
```

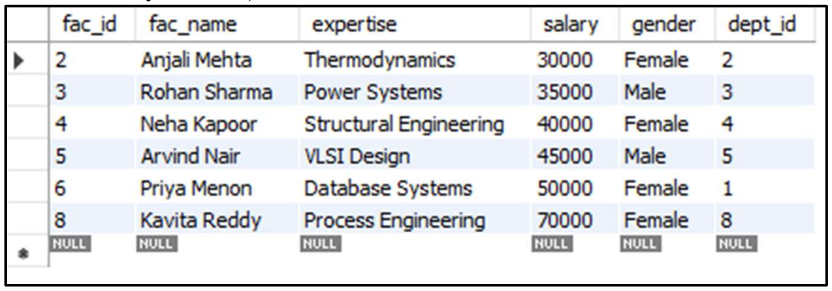


std_id	std_name	stud_email	gender	age	dept_id	fac_id	exam_id	admission_date
1	Suraj Thakur	suraj@gmail.com	Male	21	3	3	3	2022-08-15
3	Rohan Sharma	rohan@gmail.com	Male	23	5	4	2	2021-11-21
4	Neha Kapoor	neha@gmail.com	Female	22	4	3	7	2020-09-05
5	Arvind Nair	arvind@gmail.com	Male	22	1	5	1	2024-11-18
6	Priya Menon	priya@gmail.com	Female	21	7	6	6	2025-02-12
7	Sameer Iyer	sameer@gmail.com	Male	20	6	7	4	2021-05-30
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- Using ALL operator

Find faculty details whose salary is greater than faculty with salary of 30000.

SELECT * FROM Faculty WHERE salary > ALL(SELECT salary FROM Faculty WHERE salary < 30000);



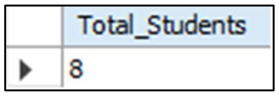
fac_id	fac_name	expertise	salary	gender	dept_id
2	Anjali Mehta	Thermodynamics	30000	Female	2
3	Rohan Sharma	Power Systems	35000	Male	3
4	Neha Kapoor	Structural Engineering	40000	Female	4
5	Arvind Nair	VLSI Design	45000	Male	5
6	Priya Menon	Database Systems	50000	Female	1
8	Kavita Reddy	Process Engineering	70000	Female	8
NULL	NULL	NULL	NULL	NULL	NULL

6.AGGREGATE FUNCTIONS:

- Count Function:

Find the total number of students in colege.

SELECT COUNT(*) AS Total_Students FROM Students;

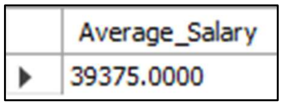


Total_Students
8

- Average Function with round function:

Find the average salary of all faculty members.

SELECT AVG(salary) AS Average_Salary FROM Faculty;



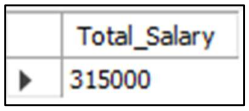
Average_Salary
39375.0000

- Sum Function :

Display total Salary Paid to All Faculty.

SELECT SUM(salary) AS Total_Salary

FROM Faculty;

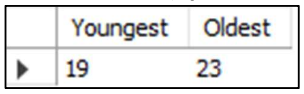


Total_Salary
315000

☐ Max, Min Function:

Find the youngest and oldest student in the college.

SELECT MIN(age) AS Youngest, MAX(age) AS Oldest FROM Students;



Youngest	Oldest
19	23

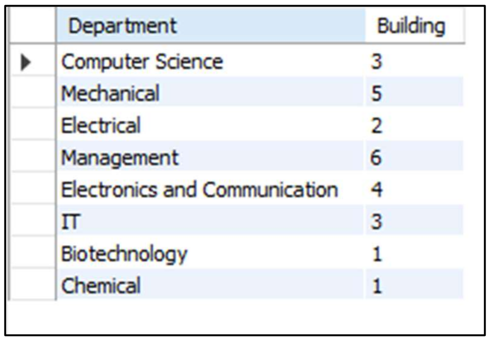
7.GROUP BY :

To display building for each department.

SELECT dept_name AS Department,

building_no AS Building FROM Department

GROUP BY dept_id;



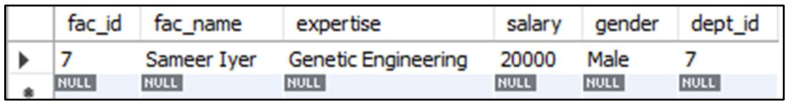
Department	Building
Computer Science	3
Mechanical	5
Electrical	2
Management	6
Electronics and Communication	4
IT	3
Biotechnology	1
Chemical	1

8.LIKE OPERATOR :

Find Faculty members whose name starts with s second lastletter is e.

SELECT * FROM Faculty WHERE

fac_name LIKE 'S%e_';



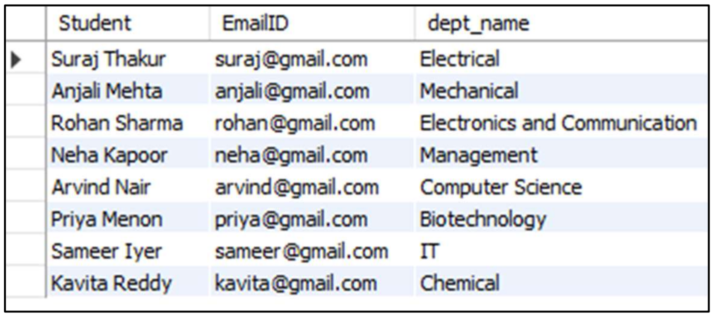
fac_id	fac_name	expertise	salary	gender	dept_id
7	Sameer Iyer	Genetic Engineering	20000	Male	7
NULL	NULL	NULL	NULL	NULL	NULL

9.JOINS :

- Inner Join:

To fetch all student names, their email id along with the department they belong to .

SELECT s.std_name AS Student, s.stud_email AS EmailID, d.dept_name FROM Students AS s INNER JOIN Department AS d ON s.dept_id = d.dept_id;

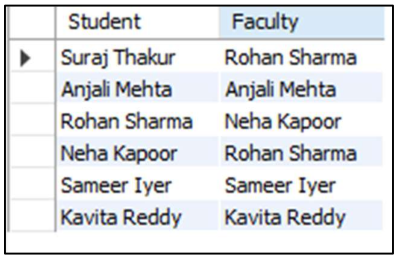


Student	EmailID	dept_name
Suraj Thakur	suraj@gmail.com	Electrical
Anjali Mehta	anjali@gmail.com	Mechanical
Rohan Sharma	rohan@gmail.com	Electronics and Communication
Neha Kapoor	neha@gmail.com	Management
Arvind Nair	arvind@gmail.com	Computer Science
Priya Menon	priya@gmail.com	Biotechnology
Sameer Iyer	sameer@gmail.com	IT
Kavita Reddy	kavita@gmail.com	Chemical

- Left Join:

Showing the students with the faculty assigned to them for students who enrolled before 2024.

SELECT s.std_name AS Student, f.fac_name AS Faculty FROM Students AS s LEFT JOIN Faculty AS f ON s.fac_id = f.fac_id WHERE s.admission_date<'2023-12-31';



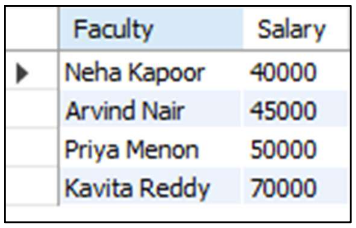
Student	Faculty
Suraj Thakur	Rohan Sharma
Anjali Mehta	Anjali Mehta
Rohan Sharma	Neha Kapoor
Neha Kapoor	Rohan Sharma
Sameer Iyer	Sameer Iyer
Kavita Reddy	Kavita Reddy

10.SUBQUERIES:

1.Single row Subqueries:

Find Faculty name and salary for those who earn more than the Average of salary.

SELECT fac_name as Faculty, Salary FROM Faculty WHERE
salary > (SELECT AVG(salary) FROM Faculty);

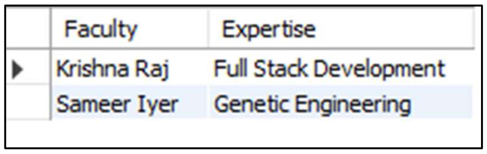


Faculty	Salary
Neha Kapoor	40000
Arvind Nair	45000
Priya Menon	50000
Kavita Reddy	70000

2.Multiple row subquery:

List of faculty names with their expertise in Full Stack Development or Genetic Engineering.

SELECT fac_name AS Faculty, Expertise FROM Faculty WHERE expertise
IN(SELECT expertise FROM FACULTY WHERE expertise="Full Stack Development"
OR expertise="Genetic Engineering");

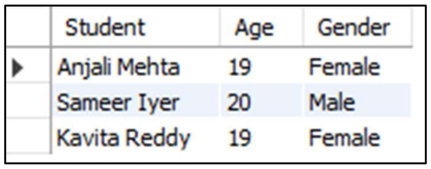


Faculty	Expertise
Krishna Raj	Full Stack Development
Sameer Iyer	Genetic Engineering

3.Multiple column subquery :

Display youngest student along with their gender.

SELECT std_name AS Student, Age, Gender FROM Students WHERE (age,gender) IN (SELECT MIN(age),gender FROM Students GROUP BY Gender);

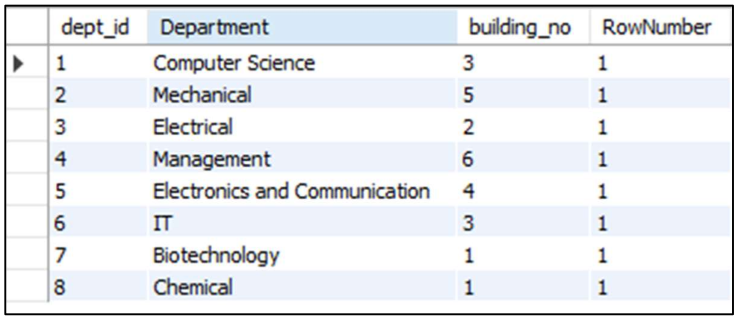


Student	Age	Gender
Anjali Mehta	19	Female
Sameer Iyer	20	Male
Kavita Reddy	19	Female

11. WINDOW FUNCTIONS

- Row_Number():

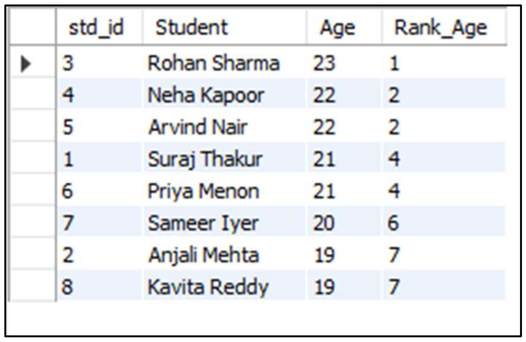
SELECT dept_id, dept_name AS Department, building_no, ROW_NUMBER() OVER(PARTITION BY dept_id ORDER BY dept_id) AS RowNumber FROM Department;



dept_id	Department	building_no	RowNumber
1	Computer Science	3	1
2	Mechanical	5	1
3	Electrical	2	1
4	Management	6	1
5	Electronics and Communication	4	1
6	IT	3	1
7	Biotechnology	1	1
8	Chemical	1	1

- Rank():

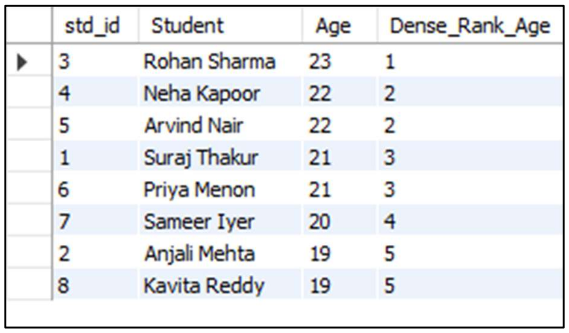
SELECT std_id, std_name AS Student, Age, RANK() OVER(ORDER BY Age DESC) AS Rank_Age FROM Students;



std_id	Student	Age	Rank_Age
3	Rohan Sharma	23	1
4	Neha Kapoor	22	2
5	Arvind Nair	22	2
1	Suraj Thakur	21	4
6	Priya Menon	21	4
7	Sameer Iyer	20	6
2	Anjali Mehta	19	7
8	Kavita Reddy	19	7

- Dense_Rank():

SELECT std_id, std_name AS Student, Age, DENSE_RANK() OVER(ORDER BY Age DESC) AS Dense_Rank_Age FROM Students;



std_id	Student	Age	Dense_Rank_Age
3	Rohan Sharma	23	1
4	Neha Kapoor	22	2
5	Arvind Nair	22	2
1	Suraj Thakur	21	3
6	Priya Menon	21	3
7	Sameer Iyer	20	4
2	Anjali Mehta	19	5
8	Kavita Reddy	19	5

---END---