

**Lab 1: Create a new database named school\_db and a table called students with the following columns: student\_id, student\_name, age, class, and address.**

**Ans :-**

```
CREATE DATABASE school_db;
```

```
CREATE TABLE students(  
    student_id int,  
    student_name varchar(25),  
    age int,  
    class varchar(25),  
    address text  
);
```

student_id	student_name	age	class	address
------------	--------------	-----	-------	---------

**Lab 2: Insert five records into the students table and retrieve all records using the SELECT statement.**

**ANS :-**

insert into students VALUES

```
(1 , 'jaydip' , 19 , 'chemistry' , 'surat') ,  
(2 , 'meet' , 21 , 'physics' , 'ahemdabad'),  
(3 , 'rohan' , 20 , 'accounting' , 'gota road'),  
(4 , 'kishan' , 17 , 'CCC' , 'dholka'),  
(5 , 'nisha' , 18 , 'biology' , 'ahemdabad');
```

student_id	student_name	age	class	address
1	jaydip	19	chemistry	surat
2	meet	21	physics	ahemdabad
3	rohan	20	accounting	gota road
4	kishan	17	CCC	dholka
5	nisha	18	biology	ahemdabad

**Lab 1: Write SQL queries to retrieve specific columns (student\_name and age) from the students table.**

**ANS :-**

```
SELECT student_name , age FROM students;
```

student_name	age
jaydip	19
meet	21
rohan	20
kishan	17
nisha	18

**Lab 2: Write SQL queries to retrieve all students whose age is greater than 10.**

**ANS :-**

```
SELECT age FROM students WHERE age>10;
```

age
19
21
20
17
18

**Lab 1: Create a table teachers with the following columns: teacher\_id (Primary Key), teacher\_name (NOT NULL), subject (NOT NULL), and email (UNIQUE).**

**ANS :-**

```
CREATE TABLE teachers (  
    teacher_id int PRIMARY key,  
    teacher_name text NOT NULL,  
    subject varchar(40)NOT NULL,  
    email text UNIQUE  
);
```

INSERT into teachers VALUES

```
(1,'rakeshbhai','python','rakeshr123@gmail.com'),
(2,'brijeshbai','mathematics','brijeshb456@gmail.com'),
(3,'umang patel','java backend','umangp777@gmail.com'),
(4,'rinku mam','java script','rinkur99@gmail.com'),
(5,'khushi mam','data science','kkhushi87@gmail.com');
```

teacher_id	teacher_name	subject	email
1	rakeshbhai	python	rakeshr123@gmail.com
2	brijeshbai	mathematics	brijeshb456@gmail.com
3	umang patel	java backend	umangp777@gmail.com
4	rinku mam	java script	rinkur99@gmail.com
5	khushi mam	data science	kkhushi87@gmail.com

**Lab 2: Implement a FOREIGN KEY constraint to relate the teacher\_id from the teachers table with the students table.**

**ANS :-**

```
CREATE TABLE students1(
    student_id int PRIMARY KEY,
    student_name varchar(25),
    age int,
    teacher_id int ,
    FOREIGN KEY (teacher_id) REFERENCES teachers(teacher_id)
);
```

student_id	student_name	age	teacher_id
1	jaydip	19	3
2	meet	21	1
3	rohan	20	5
4	kishan	17	2
5	nisha	18	4

**Lab 1: Create a table courses with columns: course\_id, course\_name, and course\_credits. Set**

the `course_id` as the primary key.

**ANS :-**

```
CREATE TABLE course(  
    course_id int PRIMARY KEY,  
    course_name varchar(35),  
    course_credits int  
);
```

```
course_id  course_name  course_credits
```

**Lab 2: Use the CREATE command to create a database `university_db`.**

**ANS :-**

```
CREATE DATABASE university_db;
```

**Lab 1: Modify the courses table by adding a column `course_duration` using the ALTER command.**

**ANS :-**

```
ALTER TABLE course ADD COLUMN course_duration int;
```

```
course_id  course_name  course_credits  course_duration
```

**Lab 2: Drop the `course_credits` column from the courses table**

**ANS :-**

```
ALTER TABLE course DROP COLUMN course_credits ;
```

```
course_id  course_name  course_duration
```

**Lab 1: Drop the teachers table from the `school_db` database.**

**ANS :-**

```
DROP TABLE teachers;
```

**Lab 2: Drop the students table from the `school_db` database and verify that the table has been removed.**

**ANS :-**

```
DROP TABLE students;
```

```
DROP DATABASE school_db;
```

**Lab 1: Insert three records into the courses table using the INSERT command.**

**ANS :-**

```
INSERT INTO course VALUES  
(1,'java-full stack', 9),  
(2,'SE-Software Engineering', 10),  
(3,'data analysis', 12);
```

course_id	course_name	course_duration
1	java-full stack	9
2	SE-Software Engineering	10
3	data analysis	12

**Lab 2: Update the course duration of a specific course using the UPDATE command.**

**ANS :-**

```
UPDATE course SET course_duration=8 WHERE course_name='java-full stack';  
UPDATE course SET course_duration=11 WHERE course_name='SE-Software Engineering';  
UPDATE course SET course_duration=10 WHERE course_name='data analysis';
```

course_id	course_name	course_duration
1	java-full stack	8
2	SE-Software Engineering	11
3	data analysis	10

**Lab 3: Delete a course with a specific course\_id from the courses table using the DELETE command.**

**ANS :-**

```
DELETE FROM course WHERE course_id=1;  
DELETE FROM course WHERE course_id=2;  
DELETE FROM course WHERE course_id=3;
```

**Lab 1: Retrieve all courses from the courses table using the SELECT statement.**

**ANS :-**

SELECT \* FROM course;

course_id	course_name	course_duration
1	java-full stack	9
2	SE-Software Engineering	10
3	data analysis	12
4	python	2
5	data science	5

**Lab 2: Sort the courses based on course\_duration in descending order using ORDER BY.**

**ANS :-**

SELECT \* FROM course ORDER BY course\_duration DESC;

course_id	course_name	course_duration	▼ 1
3	data analysis	12	
2	SE-Software Engineering	10	
1	java-full stack	9	
5	data science	5	
4	python	2	

**Lab 3: Limit the results of the SELECT query to show only the top two courses using LIMIT.**

**ANS :-**SELECT \* FROM COURSE LIMIT>2;

**Lab 1: Create two tables: departments and employees. Perform an INNER JOIN to display employees along with their respective departments.**

**ANS :-**

SELECT employee.emp\_id , employee.emp\_name , department.dep\_name FROM employee INNER JOIN department ON employee.dep\_id = department.dep\_id;

emp_id	emp_name	dep_name
101	kishan	Engineering
102	rohan	Human Resources
103	mayur	Marketing
104	vishal	Marketing
105	suresh	Human Resources
106	ram	Engineering
107	jaydip	Human Resources

**Lab 2: Use a LEFT JOIN to show all departments, even those without employees.**

**ANS :-**

```
SELECT department.dep_name , employee.emp_name , department.dep_id FROM employee LEFT JOIN department ON department.dep_id=employee.dep_id;
```

dep_name	emp_name	dep_id
Engineering	kishan	2
Human Resources	rohan	1
Marketing	mayur	3
Marketing	vishal	3
Human Resources	suresh	1
Engineering	ram	2
Human Resources	jaydip	1

**Lab 1: Group employees by department and count the number of employees in each department using GROUP BY.**

**ANS :-**

```
SELECT d_name , COUNT(emp_name) AS total_emp FROM employee GROUP BY d_name;
```

d_name	total_emp
NULL	1
enginnering	1
human resorces	1
marketing	2
sales	2

**Lab 2: Use the AVG aggregate function to find the average salary of employees in each department.**

**ANS :-**

```
SELECT AVG(emp_salary) AS aevrage_salary FROM employee;
```

average\_salary  
29000.0000

**Lab 1: Write a stored procedure to retrieve all employees from the employees table based on department.**

**ANS:-**

Delimiter \$\$

```
CREATE PROCEDURE p_1(e_id int,e_name varchar(60) , d_name varchar(60))
BEGIN
INSERT INTO employee VALUES(e_id , e_name , d_name);
END;
```

```
CALL p_1(21 , 'rohanbhai' , 'sales');
CALL p_1(22 , 'kishan','engineering');
CALL p_1 (23 , 'vrusha','counsellar');
```

e_id	e_name	d_name
21	rohanbhai	sales
22	kishan	engineering
23	vrusha	counsellar

**Lab 2: Write a stored procedure that accepts course\_id as input and returns the course details.**

**ANS :-**

Delimiter \$\$

```
CREATE PROCEDURE t_13(i int)
BEGIN
SELECT * FROM course_1 WHERE course_id=i;
END;
```

**Lab 1: Create a view to show all employees along with their department names.**

**ANS :-**

```
CREATE VIEW v_1 AS SELECT emp_name , d_name FROM employee;
SELECT * FROM v_1;
```



emp_name	d_name
kishan	sales
rohan	enginnering
mayur	sales
vishal	marketing
suresh	human resorces
ram	marketing
jaydip	NULL

**Lab 2: Modify the view to exclude employees whose salaries are below \$50,000.**

**ANS :-**

```
CREATE VIEW v_14 AS SELECT emp_id ,emp_name , emp_salary ,d_name FROM employee
WHERE emp_salary<50000;
SELECT * FROM v_14;
```

emp_id	emp_name	emp_salary	d_name
102	rohan	30000	enginnering
104	vishal	25000	marketing
106	ram	23000	marketing
107	jaydip	25000	NULL

**Lab 1: Create a trigger to automatically log changes to the employees table when a new employee is added.**

**ANS :-**

DELIMITER \$\$

CREATE TRIGGER t\_1 AFTER INSERT ON emp\_1 FOR EACH ROW

BEGIN

INSERT INTO dep\_1(id , name , records) VALUES(new.e\_id ,new.e\_name , 'record is succesfully inserted.');

END

INSERT INTO emp\_1 VALUES(101 , 'meet'),(102 , 'umang'),(103 , 'mehul');

id	name	time_date	records
101	meet	2025-01-17 10:43:00	record is succesfully inserted.
102	umang	2025-01-17 10:43:00	record is succesfully inserted.
103	mehul	2025-01-17 10:43:00	record is succesfully inserted.

**Lab 2: Create a trigger to update the last\_modified timestamp whenever an employee record is updated.**

**ANS :-**

DELIMITER \$\$

CREATE TRIGGER t\_2 AFTER UPDATE ON emp\_1 FOR EACH ROW

BEGIN

INSERT INTO dep\_1(id , name , records) VALUES(new.e\_id ,new.e\_name , 'record is succesfully updated.');

END

UPDATE emp\_1 SET e\_name='krisha' WHERE e\_id=101;

id	name	time_date	records
101	meet	2025-01-17 10:43:00	record is succesfully inserted.
102	umang	2025-01-17 10:43:00	record is succesfully inserted.
103	mehul	2025-01-17 10:43:00	record is succesfully inserted.
101	krisha	2025-01-17 10:47:50	record is succesfully updated.