Q.1. 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.

```
CREATE TABLE student (
   student_id int,
   student_name text,
   age int,
   class varchar(2),
   address text
   );
```

Ans.

student_id	student_name	age	class	address

Q.1. Lab 2: Insert five records into the students table and retrieve all records using the SELECT statement

```
INSERT into student VALUES(1,'kishan',18,'a','ghatlodiya'),
 (2,'meru',20,'b','chandlodiya'),
 (3,'rohan',21,'b','kalupur'),
 (4,'kirtan',22,'b','ratanpur'),
 (5,'bhargavi',24,'a','chandlodiya');
```

Ans.

student_id	student_name	age	class	address
1	kishan	18	a	ghatlodiya
2	meru	20	b	chandlodiya
3	rohan	21	b	kalupur
4	kirtan	22	b	ratanpur
5	bhargavi	24	a	chandlodiya

Q.2. Lab 1: Write SQL queries to retrieve specific columns (student_name and age) from the students table.

```
SELECT student_name,age FROM student;
```

Ans.

student_name	age
kishan	18
meru	20
rohan	21
kirtan	22
bhargavi	24

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

```
SELECT * FROM student WHERE age>=10;
```

Ans.

student_id	student_name	age	class	address
1	kishan	18	a	ghatlodiya
2	meru	20	b	chandlodiya
3	rohan	21	b	kalupur
4	kirtan	22	b	ratanpur
5	bhargavi	24	a	chandlodiya

Q.3. Lab 1: Create a table teachers with the following columns: teacher_id (Primary Key), teacher_name (NOT NULL), subject (NOT NULL), and email (UNIQUE).

```
CREATE TABLE teachers( teacher_id int PRIMARY KEY, teacher_name text not null, subjec
t text NOT NULL, email text UNIQUE );
INSERT INTO teachers VALUES(1,'kinal','English','kinal@123gmail.com'),
(2,'sanju','s.s','sanju@123gmail.com'),
(3,'hiral','gujrati','hiral@123gmail.com'),
(4,'kishan','hindi','kishan@123gmail.com'),
(5,'rohan','sanskrit','rohan@123gmail.com');
```

Ans.

teacher_id	teacher_name	subject	email
1	kinal	English	kinal@123gmail.com
2	sanju	s.s	sanju@123gmail.com
3	hiral	gujrati	hiral@123gmail.com
4	kishan	hindi	kishan@123gmail.com
5	rohan	sanskrit	rohan@123gmail.com

Q.3. Lab 2: Implement a FOREIGN KEY constraint to relate the teacher_id from the teachers table with the students table.

```
CREATE TABLE student1( student_id int PRIMARY KEY, student_name text, student
_subject text, teacher_id int, FOREIGN KEY (teacher_id) REFERENCES teachers(
teacher_id));
```

```
INSERT INTO student1
VALUES(11,'ramu','English',1),(12,'ronny','gujrati',2),(13,'rocky','s.s',3),(14,'rahul','hindi',4),
(15,'raj','sanscrit',5);
```

Ans.

student_id	student_name	student_subject	teacher_id
11	ramu	English	1
12	ronny	gujrati	2
13	rocky	s.s	3
14	rahul	hindi	4
15	raj	sanscrit	5

Q.4. Lab 1: Create a table courses with columns: course_id, course_name, and course_credits. Set the course_id as the primary key.

```
CREATE TABLE courses(
 course_id int primary key,
 course_name text,
course_credits int
   );
```

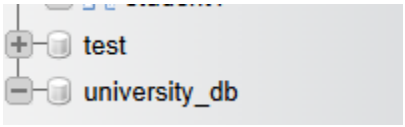
Ans.

course_id	course_name	course_credits

Q.4. Lab 2: Use the CREATE command to create a database university_db.

```
CREATE DATABASE university_db;
```

Ans.



Q.5. Lab 1: Modify the courses table by adding a column course_duration using the ALTER command.

```
ALTER TABLE courses ADD course_duration int;
```

Ans.

course_id	course_name	course_credits	course_duration

Q.5. Lab 2: Drop the course_credits column from the courses table.

```
ALTER TABLE courses DROP COLUMN course_credits;
```

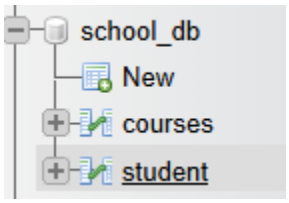
Ans.

course_id	course_name	course_duration

Q.6 Lab 1: Drop the teachers table from the school_db database.

```
DROP TABLE teachers;
```

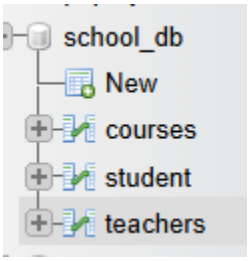
Ans.



Q.6 : Drop the students table from the school_db database and verify that the table has been removed.

DROP TABLE student1;

Ans.



Q.7 Lab 1: Insert three records into the courses table using the INSERT command.

INSERT into courses VALUES(1,'java',2),(2,'python',1),(3,'c',2),(4,'c++',1);

Ans.

course_id	course_name	course_duration
1	java	2
2	python	1
3	c	2
4	c++	1

Q.7 Lab 2: Update the course duration of a specific course using the UPDATE command.

UPDATE courses set course_duration=3 WHERE course_name='python';

Ans.

course_id	course_name	course_duration
1	java	2
2	python	3
3	c	2
4	c++	1

Q.7 Lab 3: Delete a course with a specific course_id from the courses table using the DELETE command.

DELETE FROM courses WHERE course_id=3;

Ans.

course_id	course_name	course_duration
1	java	2
2	python	3
4	c++	1

Q.8 Lab 1: Retrieve all courses from the courses table using the SELECT statement.

*Select * from courses;*

Ans.

course_id	course_name	course_duration
1	java	2
2	python	3
4	c++	1

Q.8 Lab 2: Sort the courses based on course_duration in descending order using ORDER BY.

*SELECT * FROM courses ORDER BY course_duration;*

Ans.

course_id	course_name	course_duration ▲ 1
4	c++	1
1	java	2
2	python	3

Q.8 Lab 3: Limit the results of the SELECT query to show only the top two courses using LIMIT.

*SELECT * FROM courses limit 2;*

Ans.

course_id	course_name	course_duration
1	java	2
2	python	3

Q.9

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

select emp2.Emp_Name,department.dep_name,department.dep_id from emp2 INNER JOIN department on emp2.Emp_No=department.Emp_No;

Emp_Name	dep_name	dep_id
kishan	perchese	11
krish	selse	12
krisha	perchese	13
chaya	legal	14
meru	selse	15
hiral	selse	16
lila	legal	17
rohan	perchese	18

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

select department.dep_name,emp2.Emp_Name,department.dep_id from department LEFT OUTER JOIN emp2 on department.Emp_No=emp2.Emp_No;

Ans.

dep_name	Emp_Name	dep_id
perchese	kishan	11
selse	krish	12
perchese	krisha	13
legal	chaya	14
selse	meru	15
selse	hiral	16
legal	lila	17
perchese	rohan	18

Q.12

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

SELECT dep_name,COUNT(dep_name)FROM department GROUP by dep_name;

Ans.

dep_name	COUNT(dep_name)
legal	2
perchese	3
selse	3

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

SELECT AVG(Salary)Salary FROM emp2;

Ans.

Salary
38250.0000

Q.13

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

```
DELIMITER $$

CREATE PROCEDURE Getdepartment( d text)
BEGIN
  SELECT Emp_No,Emp_Name,Salary
  FROM emp
  WHERE emp_dept = d;
END
call getdepartment('fainace');
ans.
```

Emp_No	Emp_Name	Salary
1	kishan	20000
4	chaya	25000
8	rohan	55000

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

```
DELIMITER $$
CREATE PROCEDURE coursedata5(i int)
BEGIN
SELECT rollno,sname,course FROM student WHERE rollno = i;
END;
Call coursedata5(3);
Ans.
```

rollno	sname	course
3	meru	java

Q.14

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

```
CREATE VIEW emp_v1 as SELECT Emp_Name,emp_dept FROM emp;
SELECT * FROM emp_v1;
Ans.
```

Emp_Name	emp_dept
kishan	fainace
krish	perchese
krisha	legal
chaya	fainace
meru	perchese
hiral	perchese
lila	legal
rohan	fainace

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

CREATE VIEW emp_v3 as SELECT Emp_No,emp_name,Salary FROM emp WHERE Salary>50000;
*SELECT * FROM emp_v3;*
Ans.

Emp_No	emp_name	Salary
3	krisha	67000
8	rohan	55000

Q.15

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

```
create TRIGGER insered_data AFTER INSERT on employee for EACH ROW BEGIN insert into e
mployee1(emp_id,emp_cname, action_performed) VALUES(new.emp_id, new.emp_name, 'Record
inserted'); end;
```

ans.

emp_id	emp_cname	date_time	action_performed
1	rajeshbhai	2025-01-20 13:26:31	Record inserted
2	mehulnhai	2025-01-20 13:26:31	Record inserted
3	rajubhai	2025-01-20 13:26:31	Record inserted

```
INSERT into employee VALUES(1,'rajeshbhai'),(2,'mehulnhai'),(3,'rajubhai');
```

Ans.

emp_id	emp_name
1	rajeshbhai
2	mehulnhai
3	rajubhai

Lab **2: Create a trigger to update the last_modified timestamp whenever an employee record is**

updated.

```
DELIMITER $$

CREATE TRIGGER update_1 AFTER UPDATE ON employee FOR EACH ROW

BEGIN

INSERT INTO employee1(emp_id,emp_cname,action_performed)
VALUES(new.emp_id,new.emp_name,"Record Updated!");

END
UPDATE employee set emp_name='kishan' WHERE emp_id=3;
```

Ans.

emp_id	emp_cname	date_time	action_performed
1	rajeshbhai	2025-01-20 13:26:31	Record inserted
2	mehulnhai	2025-01-20 13:26:31	Record inserted
3	rajubhai	2025-01-20 13:26:31	Record inserted
3	kishan	2025-01-20 13:38:37	Record Updated!