|  |
| --- |
| **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  );    **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'); |
| **Lab 1: Write SQL queries to retrieve specific columns (student\_name and age) from the students table.**  **ANS :-**  [SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) student\_name , age FROM students;    **Lab 2: Write SQL queries to retrieve all students whose age is greater than 10.**  **ANS :-**  SELECT age FROM students WHERE age>10; |
| **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');    **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)  ); |
| **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  );    **Lab 2: Use the CREATE command to create a database university\_db.**  **ANS :-**  [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-database.html) [DATABASE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-database.html) 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;    **Lab 2: Drop the course\_credits column from the courses table**  **ANS :-**  ALTER TABLE course DROP COLUMN course\_credits ; |
| **Lab 1: Drop the teachers table from the school\_db database.**  **ANS :-**  [DROP](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/drop-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/drop-table.html) 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);    **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';    **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;    **Lab 2: Sort the courses based on course\_duration in descending order using ORDER BY.**  **ANS :-**  SELECT \* FROM course ORDER BY course\_duration DESC;    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;    **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; |
| **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;    **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; |
| **Lab 1: Write a stored procedure to retrieve all employees from the employees table based on department.**  **ANS:-**  Delimeter $$  [CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html) [PROCEDURE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html) p\_1(e\_id int,e\_name varchar(60) , d\_name varchar(60))   BEGIN  [INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO employee [VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values)(e\_id , e\_name , d\_name);   END;  CALL p\_1(21 , 'rohanbhai' , 'sales');  CALL p\_1(22 , 'kishan','engineering');  CALL p\_1 (23 ,'vrusha','counsellar');    **Lab 2: Write a stored procedure that accepts course\_id as input and returns the course details.**  **ANS :-**  Delimeter $$  CREATE PROCEDURE t\_13(i int)  BEGIN  SELECT \* FROM course\_1 WHERE course\_id;  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;    **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; |
| **Lab 1: Create a trigger to automatically log changes to the employees table when a new employee is added.**  **ANS :-**  DELIMITER $$  CREATE TRIGGER t\_1 AFTER INSERT ON emp\_1 FOR EACH ROW  BEGIN  INSERT INTO dep\_1(id , name , records) VALUES(new.e\_id ,new.e\_name , 'record is succesfully inserted.');  END  INSERT INTO emp\_1 VALUES(101 , 'meet'),(102 , 'umang'),(103 , 'mehul');    **Lab 2: Create a trigger to update the last\_modified timestamp whenever an employee**  **record is updated.**  **ANS :-**  DELIMITER $$  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; |
|  |