**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.**

Syntax :

CREATE DATABASE school\_db;

CREATE TABLE students (

id INT,

name VARCHAR(50),

age INT,

calss INT,

address VARCHAR(100)

);

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

INSERT INTO students VALUE (1, 'Rahul', 18, 12, 'Surat'), (2, 'Raj', 19, 11, 'Mumbai') ,

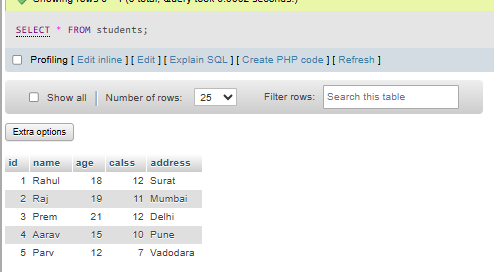
(3, 'Prem', 21, 12, 'Delhi'),

(4, 'Aarav', 15, 10, 'Pune'),

(5, 'Parv', 12, 07, 'Vadodara');

**retrieve all records using the SELECT statement.**

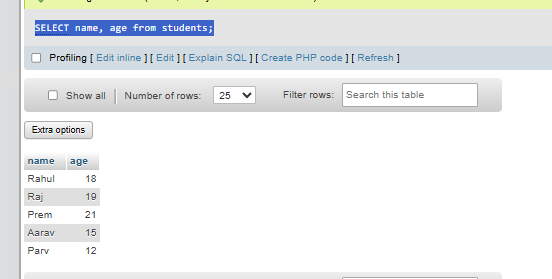
SELECT \* FROM students;



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

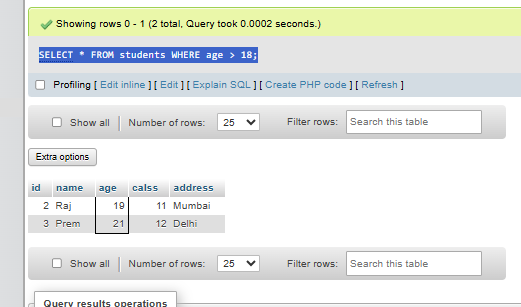
**students table.**

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) name, age from students;



Write SQL queries to retrieve all students whose age is greater than 18.

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM students WHERE age > 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).

[CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) teachers ( id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(100), subject VARCHAR(100), email VARCHAR(255) );[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO teachers (id, name, subject, email)

[VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values)

(101, 'Ram', 'sci', 'ram@gmail.com'),

(102, 'Shyam','maths','shyam@gmail.com'),

(103, 'Prakash','eng', 'prakash@gmail.com')

(104, 'Rahul','Guj', 'rahul@gmail.com')

(103, 'Rohit','Sanskrit', 'rohit@gmail.com')

**Lab : 2 Implement a FOREIGN KEY constraint to relate the teacher\_id from the**

**teachers table with the students table.**

CREATE TABLE teachers (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100),

subject VARCHAR(100),

email VARCHAR(255)

);

CREATE TABLE students (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100),

age INT,

class INT,

address VARCHAR(255),

teacher\_id INT,

FOREIGN KEY (teacher\_id) REFERENCES teachers(id)

);

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

**course\_credits. Set the course\_id as the primary key.**

[CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) courses ( course\_id INT PRIMARY KEY, course\_name VARCHAR(100) [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL, course\_credits INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL );

**Lab : 2 Use the CREATE command to create a database university\_db.**

**CREATE DATABASE university\_db;**

ALTER TABLE courses

ADD COLUMN course\_duration VARCHAR(50);

Drop the course\_credits column from the courses table.

[ALTER](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/alter-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/alter-table.html) courses DROP COLUMN course\_credits;

Drop the teachers table from the school

DROP TABLE teachers;

Drop the students table from the school\_db database and verify that the table has been removed.

DROP TABLE students;

SHOW TABLES;

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

[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO courses (course\_id, course\_name, course\_duration) [VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values) (1, 'Fullstack', '12 weeks'), (2, 'Backend', '10 weeks'), (3, 'Data science', '14 weeks'

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

[UPDATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/update.html) courses [SET](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/set.html) course\_duration = '15 weeks' WHERE course\_id = 2;

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

[DELETE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/delete.html) FROM courses WHERE course\_id = 3;

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

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM courses ;

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

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM courses ORDER BY course\_duration DESC;

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

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) \* FROM courses LIMIT 2;

**Lab 1: Create two new users user1 and user2 and grant user1 permission to SELECT from the courses table.**

CREATE [USER](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/information-functions.html%23function_user) 'user1'@'localhost' IDENTIFIED BY 'password1';

CREATE [USER](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/information-functions.html%23function_user) 'user2'@'localhost' IDENTIFIED BY 'password2';

GRANT [SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) ON school1.courses TO 'user1'@'localhost';

FLUSH PRIVILEGES;

**Revoke the INSERT permission from user1 and give it to user2.**

REVOKE [INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) ON school1.courses FROM 'user1'@'localhost';

GRANT [INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) ON school1.courses TO 'user2'@'localhost';

FLUSH PRIVILEGES;

**Lab 1: Insert a few rows into the courses table and use COMMIT to save the changes.**

START TRANSACTION;

[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO courses (course\_id, course\_name, course\_duration)

[VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values) (

101, 'Biology', '10 weeks'),

(102, 'Computer Science', '15 weeks'),

(103, 'History', '12 weeks');

COMMIT;

**Lab 2: Insert additional rows, then use ROLLBACK to undo the last insert operation.**

START TRANSACTION;

[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO courses (course\_id, course\_name, course\_duration)

[VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values) (104, 'Philosophy', '8 weeks'),

(105, 'Economics', '11 weeks');

ROLLBACK;

**Lab 3: Create a SAVEPOINT before updating the courses table, and use it to roll back specific changes.**

START TRANSACTION;

[UPDATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/update.html) courses [SET](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/set.html) course\_duration = '16 weeks' WHERE course\_id = 101;

SAVEPOINT before\_second\_update;

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

[CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) departments

( department\_id INT PRIMARY KEY, department\_name VARCHAR(100) [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL );

[CREATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/create-table.html) employees ( employee\_id INT PRIMARY KEY, employee\_name VARCHAR(100) [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL, department\_id INT, FOREIGN KEY (department\_id) REFERENCES departments(department\_id) );

[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO departments

(department\_id, department\_name)

[VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values)

(1, 'HR'), (2, 'Engineering'), (3, 'Sales');

[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO employees

(employee\_id, employee\_name, department\_id)

[VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values) (101, 'Ram', 2), (102, 'Shyam', 1), (103, 'Rahul', 2);

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) e.employee\_name, d.department\_name FROM employees e INNER JOIN departments d ON e.department\_id = d.department\_id;

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

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) d.department\_name, e.employee\_name FROM departments d [LEFT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/string-functions.html%23function_left) JOIN employees e ON d.department\_id = e.department\_id;

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

GROUP BY.

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) d.department\_name, [COUNT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html%23function_count)(\*) AS employee\_count FROM employees e JOIN departments d ON e.department\_id = d.department\_id GROUP BY d.department\_name;

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

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) d.department\_name, [AVG](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html%23function_avg)(e.salary) AS average\_salary FROM employees e JOIN departments d ON e.department\_id = d.department\_id GROUP BY d.department\_name;

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

DELIMITER $$

CREATE PROCEDURE GetEmployeesByDepartment(IN dept\_name VARCHAR(100))

BEGIN

SELECT \*

FROM employees

WHERE department = dept\_name;

END $$

DELIMITER ;

=====================================================================================

CALL GetEmployeesByDepartment('Sales');

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

DELIMITER $$

CREATE PROCEDURE GetCourseDetails(IN input\_course\_id INT)

BEGIN

SELECT \*

FROM courses

WHERE course\_id = input\_course\_id;

END $$

DELIMITER ;

CALL GetCourseDetails(101);

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

CREATE VIEW employee\_department\_view AS

SELECT

e.emp\_id,

e.emp\_name,

d.department\_name

FROM

employees e

JOIN

departments d ON e.department\_id = d.department\_id;

SELECT \* FROM employee\_department\_view;

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

CREATE VIEW employee\_department\_view AS

SELECT

e.emp\_id,

e.emp\_name,

d.department\_name,

e.salary

FROM

employees e

JOIN

departments d ON e.department\_id = d.department\_id

WHERE

e.salary >= 50000;

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

CREATE TABLE employee\_log (

log\_id INT AUTO\_INCREMENT PRIMARY KEY,

emp\_id INT,

emp\_name VARCHAR(100),

action VARCHAR(50),

log\_time DATETIME DEFAULT CURRENT\_TIMESTAMP

);

===============================================================================

DELIMITER $$

CREATE TRIGGER after\_employee\_insert

AFTER INSERT ON employees

FOR EACH ROW

BEGIN

INSERT INTO employee\_log (emp\_id, emp\_name, action)

VALUES (NEW.emp\_id, NEW.emp\_name, 'INSERT');

END $$

DELIMITER ;

==================================================================================

INSERT INTO employees (emp\_id, emp\_name, department\_id, salary)

VALUES (1001, 'Jane Doe', 2, 60000);

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

[ALTER](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/alter-table.html) [TABLE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/alter-table.html) employees ADD COLUMN last\_modified DATETIME [DEFAULT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_default) [CURRENT\_TIMESTAMP](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html%23function_current_timestamp) ON [UPDATE](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/update.html) [CURRENT\_TIMESTAMP](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html%23function_current_timestamp);

DELIMITER $$

CREATE TRIGGER before\_employee\_update

BEFORE UPDATE ON employees

FOR EACH ROW

BEGIN

SET NEW.last\_modified = NOW();

END $$

DELIMITER ;

**Lab 1: Write a PL/SQL block to print the total number of employees from the employees table.**

DELIMITER $$

CREATE PROCEDURE GetTotalEmployees()

BEGIN

SELECT COUNT(\*) AS total\_employees FROM employees;

END $$

DELIMITER ;

=================================================================================\

CALL GetTotalEmployees();

**Lab 2: Create a PL/SQL block that calculates the total sales from an orders table.**

DELIMITER $$

CREATE PROCEDURE GetTotalSales()

BEGIN

SELECT SUM(sales\_amount) AS total\_sales FROM orders;

END $$

DELIMITER ;

=================================================================================\

SELECT SUM(sales\_amount) AS total\_sales FROM orders;

**Lab 1: Write a PL/SQL block using an IF-THEN condition to check the department of an employee.**

DELIMITER $$

CREATE PROCEDURE get\_dept\_id()

BEGIN

DECLARE v\_dept\_id INT;

-- For example, assign a department ID

SET v\_dept\_id = 10;

-- Use it in some logic

SELECT \* FROM employees WHERE department\_id = v\_dept\_id;

END$$

DELIMITER ;

CALL get\_dept\_id();

** Lab 2: Use a FOR LOOP to iterate through employee records and display their names.**

DECLARE

BEGIN

FOR emp\_record IN (SELECT name FROM employees) LOOP

DBMS\_OUTPUT.PUT\_LINE(emp\_record.name);

END LOOP;

END;

/

**Lab 1: Write a PL/SQL block using an explicit cursor to retrieve and display employee details.**

**DECLARE**

CURSOR emp\_cursor IS

SELECT employee\_id, name, department FROM employees;

v\_employee\_id employees.employee\_id%TYPE;

v\_name employees.name%TYPE;

v\_department employees.department%TYPE;

BEGIN

OPEN emp\_cursor; -- Open the cursor

LOOP

FETCH emp\_cursor INTO v\_employee\_id, v\_name, v\_department;

EXIT WHEN emp\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('ID: ' || v\_employee\_id || ', Name: ' || v\_name || ', Department: ' || v\_department);

END LOOP;

CLOSE emp\_cursor; -- Close the cursor

END;

/

**Lab 2: Create a cursor to retrieve all courses and display them one by one.**

DECLARE

CURSOR course\_cursor IS

SELECT course\_id, course\_name FROM courses;

v\_course\_id courses.course\_id%TYPE;

v\_course\_name courses.course\_name%TYPE;

BEGIN

OPEN course\_cursor;

LOOP

FETCH course\_cursor INTO v\_course\_id, v\_course\_name;

EXIT WHEN course\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Course ID: ' || v\_course\_id || ', Course Name: ' || v\_course\_name);

END LOOP;

CLOSE course\_cursor;

END;

**Lab 1: Perform a transaction where you create a savepoint, insert records, then rollback to the savepoint.**

BEGIN

SAVEPOINT before\_inserts;

INSERT INTO employees (employee\_id, name, department) VALUES (201, 'John Doe', 'Finance');

INSERT INTO employees (employee\_id, name, department) VALUES (202, 'Jane Smith', 'Marketing');

ROLLBACK TO SAVEPOINT before\_inserts;

COMMIT;

END;

** Lab 2: Commit part of a transaction after using a savepoint and then rollback the remaining changes.**

BEGIN

INSERT INTO employees (employee\_id, name, department) VALUES (301, 'Alice Brown', 'Sales');

SAVEPOINT after\_first\_insert;

INSERT INTO employees (employee\_id, name, department) VALUES (302, 'Bob Green', 'IT');

INSERT INTO employees (employee\_id, name, department) VALUES (303, 'Charlie White', 'HR');

COMMIT;

INSERT INTO employees (employee\_id, name, department) VALUES (304, 'Diana King', 'Operations');

ROLLBACK;

END;