**Introduction to SQL**

LAB EXERCISES:

**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.**

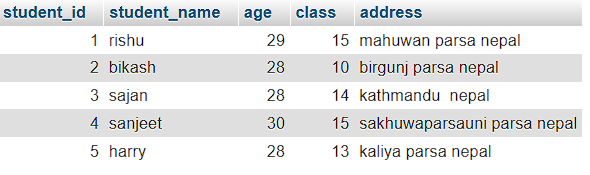
**Lab 2: Insert five records into the students table and retrieve all records using the SELECT statement.**

Command:

CREATE DATABASE school\_db;

CREATE TABLE students(student\_id int, student\_name varchar(30), age int , class int , address text);

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

****

**2. SQL Syntax**

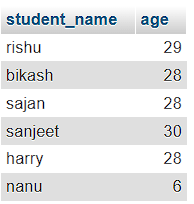
LAB EXERCISES:

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

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

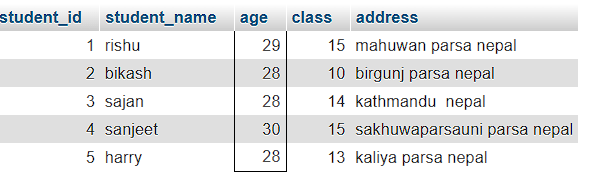
Command:

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



Command:

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html)  \*FROM `student` WHERE age>10;



**3. SQL Constraints**

LAB EXERCISES:

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

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

Command:

CREATE TABLE teachers

(

teacher\_id int (Primary Key),

teacher\_name varchar (NOT NULL),

subject varchar (NOT NULL),

email text (UNIQUE)

),



Command:

CREATE TABLE students(

student\_id int PRIMARY KEY,

student\_name varchar(30),

age int , class int ,

address text,

teacher\_id int,

FOREIGN KEY(teacher\_id) REFERENCES students(teacher\_id)

);



**4. Main SQL Commands and Sub-commands (DDL)**

LAB EXERCISES:

**• Lab 1: Create a table courses with columns: course\_id, course\_name, and course\_credits. Set the course\_id as the primary key.**

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

**Command:**

CREATE DATABASE university\_db;

CREATE TABLE courses(

course\_id int PRIMARY KEY,

course\_name varchar(30),

course\_credits int

);



**5. ALTER Command**

LAB EXERCISES:

**• Lab 1: Modify the courses table by adding a column course\_duration using the ALTER command.**

**• Lab 2: Drop the course\_credits column from the courses table.**

**Command:**

ALTER TABLE courses add course\_duration int;

****

**Command:**

ALTER TABLE courses DROP COLUMN course\_credits;

****

**6. DROP Command**

LAB EXERCISES:

**• Lab 1: Drop the teachers table from the school\_db database.**

**• Lab 2: Drop the students table from the school\_db database and verify that the table has been removed.**

**Command:**

DROP TABLE students;

DROP TABLE TEACHERS;

**7. Data Manipulation Language (DML)**

LAB EXERCISES:

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

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

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

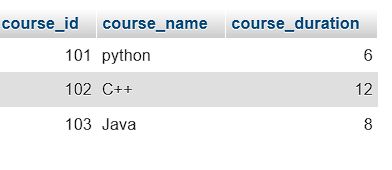
**Command:**

INSERT INTO courses VALUES(

101,'python','6 month'),

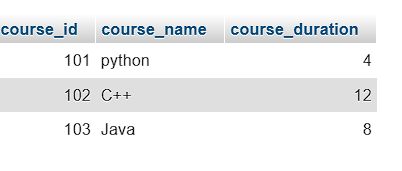
(102,'C++','12 month'),

(103,'Java','8 month');



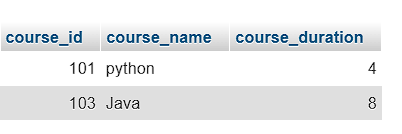
**Command:**

UPDATE courses SET course\_duration=4 WHERE course\_name=’python’;



**Command:**

DELETE FROM courses WHERE course\_id=102;



**8. Data Query Language (DQL)**

LAB EXERCISES:

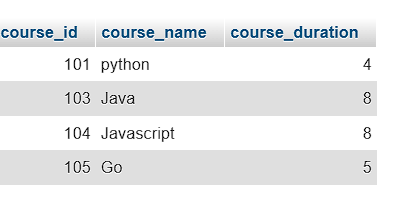
**• Lab 1: Retrieve all courses from the courses table using the SELECT statement.**

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

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

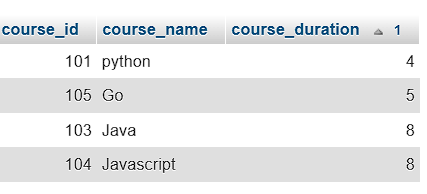
**Command:**

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

****

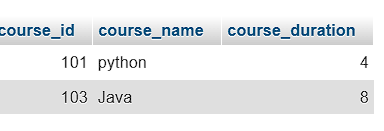
**Command:**

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

****

**Command:**

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

****

**11. SQL Joins**

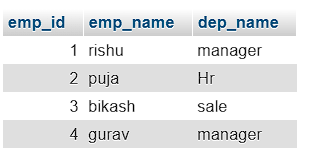
LAB EXERCISES:

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

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

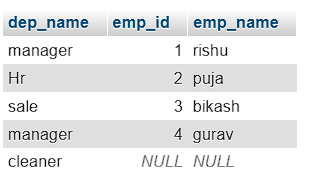
**Command:**

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

****

**Command:**

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

****

**12. SQL Group By**

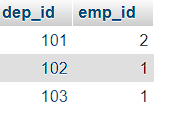
LAB EXERCISES:

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

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

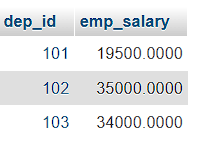
**Command:**

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) dep\_id, [COUNT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html%23function_count)(\*) AS emp\_id FROM employees GROUP by dep\_id;

****

**Command:**

[SELECT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/select.html) dep\_id, [AVG](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/aggregate-functions.html%23function_avg)(emp\_salary) AS emp\_salary FROM employees GROUP by dep\_id;

****

**13. SQL Stored Procedure**

LAB EXERCISES:

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

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

**Command:**

DELIMITER $$

CREATE PROCEDURE GetEmployeesByDepartment(IN dep\_id INT)

BEGIN

SELECT

emp\_id,

emp\_name,

emp\_gander,

emp\_salary,

dep\_id

FROM

employees

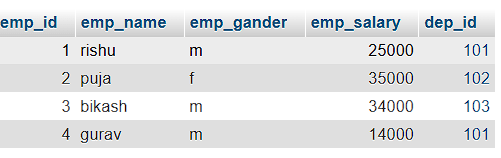
WHERE

dep\_id = dept\_id;

END $$

DELIMITER ;

CALL GetEmployeesByDepartment(2)

****

**Command:**

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

BEGIN

SELECT

course\_id,

course\_name,

course\_duration

FROM

courses

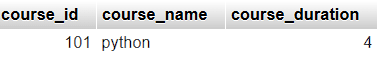
WHERE

course\_id = input\_course\_id;

END $$

DELIMITER ;

[CALL](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/call.html) GetCourseDetails(101);



**14. SQL View**

LAB EXERCISES:

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

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

**Command:**

CREATE VIEW EmployeeDepartmentView AS

SELECT

e.emp\_id,

e.emp\_name,

e.emp\_gander,

e.emp\_salary,

d.dep\_id,

d.dep\_name

FROM

employees e

JOIN

departments d

ON

e.dep\_id = d.dep\_id;

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



**Command:**

CREATE OR REPLACE VIEW EmployeeDepartmentView AS

SELECT

e.emp\_id,

e.emp\_name,

e.emp\_gander,

e.emp\_salary,

d.dep\_id,

d.dep\_name

FROM

employees e

JOIN

departments d

ON

e.dep\_id = d.dep\_id

WHERE

e.emp\_salary >= 34000;

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

****

**15. SQL Triggers**

LAB EXERCISES:

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

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

**Command:**

[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) employee\_changes\_log ( log\_id INT AUTO\_INCREMENT PRIMARY KEY, employee\_id INT [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL, change\_type VARCHAR(50) [NOT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/logical-operators.html%23operator_not) NULL, change\_date TIMESTAMP [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), details TEXT );

DELIMITER $$

CREATE TRIGGER after\_employee\_insert

AFTER INSERT ON employees

FOR EACH ROW

BEGIN

INSERT INTO employee\_changes\_log (

employee\_id,

change\_type,

details

)

VALUES (

NEW.emp\_id,

'INSERT',

CONCAT('New employee added: ', NEW.emp\_name, ' ',

', Department ID: ', NEW.dep\_id,

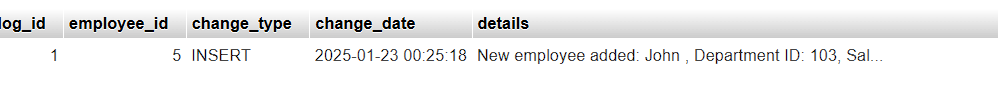
', Salary: $', NEW.emp\_salary)

);

END $$

DELIMITER ;

INSERT INTO employees (emp\_id, emp\_name, dep\_id, emp\_salary) VALUES (4, 'John', 101, 60000);

****

[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 TIMESTAMP [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 = CURRENT\_TIMESTAMP;

END $$

DELIMITER ;

UPDATE employees

SET emp\_salary = 65000

WHERE emp\_id = 5;

