# Module 5 – Introduction to DBMS

(lab-ex)

-harsh chauhan

# Introduction to SQL

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 students(student_id int PRIMARY KEY AUTO_INCREMENT,student_name varchar(100), age int, class int, address varchar(200))

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0037 seconds.)

```
CREATE TABLE students(student_id int PRIMARY KEY AUTO_INCREMENT,student_name varchar(100), age int, class int, address varchar(200));
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

| | # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|---|
| ☐ | 1 | **student_id** | int(11) | | | No | *None* | | AUTO_INCREMENT | Change Drop More |
| ☐ | 2 | **student_name** | varchar(100) | utf8mb4_general_ci | | Yes | *NULL* | | | Change Drop More |
| ☐ | 3 | **age** | int(11) | | | Yes | *NULL* | | | Change Drop More |
| ☐ | 4 | **class** | int(11) | | | Yes | *NULL* | | | Change Drop More |
| ☐ | 5 | **address** | varchar(200) | utf8mb4_general_ci | | Yes | *NULL* | | | Change Drop More |

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

5 rows inserted.
Inserted row id: 5 (Query took 0.0046 seconds.)

```
INSERT into students(student_id,student_name,age,class,address) VALUES(null,'harsh',22,'3','rajkot'),(null,'anuradha',19,'1','gondal'),
(null,'vedisha',22,'3','morbi'),(null,'sayan',23,'2','bhavnagar'),(null,'nibhay',20,'2','godhara');
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

| | | | student_id | student_name | age | class | address |
|---|---|---|---|---|---|---|---|
| Edit | Copy | Delete | 1 | harsh | 22 | 3 | rajkot |
| Edit | Copy | Delete | 2 | anuradha | 19 | 1 | gondal |
| Edit | Copy | Delete | 3 | vedisha | 22 | 3 | morbi |
| Edit | Copy | Delete | 4 | sayan | 23 | 2 | bhavnagar |
| Edit | Copy | Delete | 5 | nibhay | 20 | 2 | godhara |

Check all *With selected:* Edit Copy Delete Export

# SQL Syntax

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

SELECT student_name,age FROM `students`

Showing rows 0 - 4 (5 total, Query took 0.0005 seconds.)

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

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| ←T→ | | | | student_name | age |
|---|---|---|---|---|---|
| Edit | Copy | Delete | | harsh | 22 |
| Edit | Copy | Delete | | anuradha | 19 |
| Edit | Copy | Delete | | vedisha | 22 |
| Edit | Copy | Delete | | sayan | 23 |
| Edit | Copy | Delete | | nibhay | 20 |

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

SELECT age FROM `students` WHERE age>10

Showing rows 0 - 4 (5 total, Query took 0.0008 seconds.)

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

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| ←T→ | | | age |
|---|---|---|---|
| Edit | Copy | Delete | 22 |
| Edit | Copy | Delete | 19 |
| Edit | Copy | Delete | 22 |
| Edit | Copy | Delete | 23 |
| Edit | Copy | Delete | 20 |

Check all *With selected:* Edit Copy Delete Export

# SQL Constraints

• 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 AUTO_INCREMENT,teacher_name varchar(100) NOT NULL,subject varchar(100) NOT NULL, email varchar(100) UNIQUE)

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0011 seconds.)

```
CREATE TABLE teachers(teacher_id int Primary Key AUTO_INCREMENT,teacher_name varchar(100) NOT NULL,subject varchar(100) NOT NULL, email varchar(100)
UNIQUE);
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | teacher_id | int(11) | | | No | None | | AUTO_INCREMENT | Change Drop More |
| 2 | teacher_name | varchar(100) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 3 | subject | varchar(100) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 4 | email | varchar(100) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

Add to central columns Remove from central columns

Print Propose table structure Track table Move columns Normalize

Add 1 column(s) after email Go

Indexes

| Action | Keyname | Type | Unique | Packed | Column | Cardinality | Collation | Null | Comment |
|---|---|---|---|---|---|---|---|---|---|
| Edit Rename Drop | PRIMARY | BTREE | Yes | No | teacher_id | 0 | A | No | |
| Edit Rename Drop | email | BTREE | Yes | No | email | 0 | A | Yes | |

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

ALTER TABLE students ADD teacher_id int;
ALTER TABLE students ADD FOREIGN KEY (teacher_id) REFERENCES teachers(teacher_id);

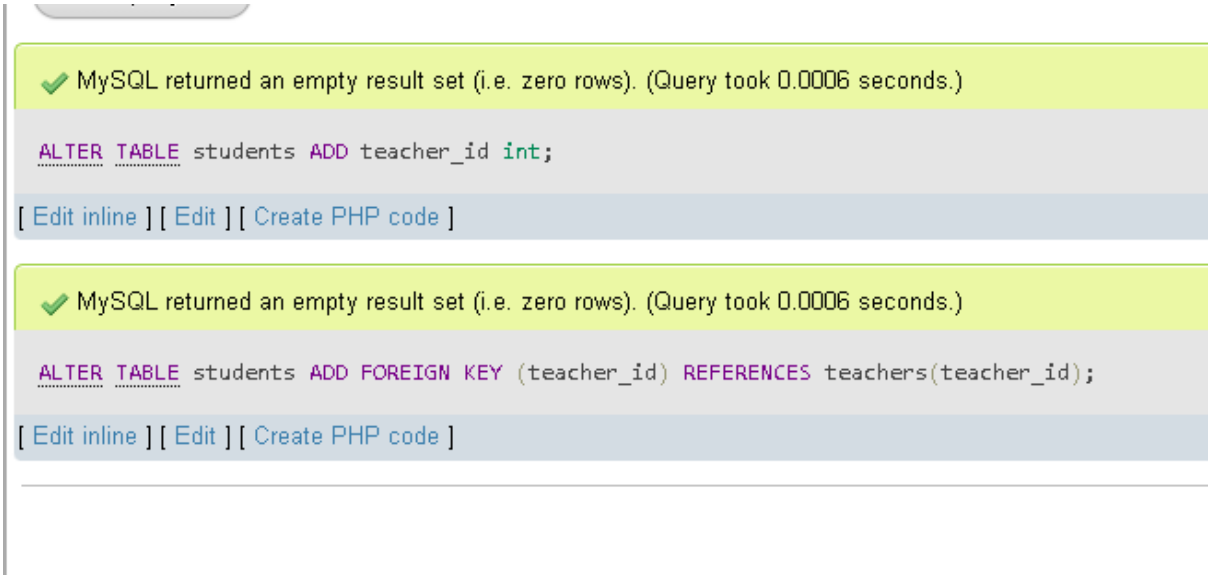


Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | **student_id** | int(11) | | | No | *None* | | AUTO_INCREMENT | Change Drop More |
| 2 | **student_name** | varchar(100) | utf8mb4_general_ci | | Yes | *NULL* | | | Change Drop More |
| 3 | **age** | int(11) | | | Yes | *NULL* | | | Change Drop More |
| 4 | **class** | int(11) | | | Yes | *NULL* | | | Change Drop More |
| 5 | **address** | varchar(200) | utf8mb4_general_ci | | Yes | *NULL* | | | Change Drop More |
| 6 | **teacher_id** | int(11) | | | Yes | *NULL* | | | Change Drop More |

Check all *With selected:* Browse Change Drop Primary Unique Index Spatial F

Add to central columns Remove from central columns

Print Propose table structure Track table Move columns Normalize

Add 1 column(s) after teacher_id Go

**Indexes**

| Action | Keyname | Type | Unique | Packed | Column | Cardinality | Collation | Null | Comment |
|---|---|---|---|---|---|---|---|---|---|
| Edit Rename Drop | **PRIMARY** | BTREE | Yes | No | student_id | 5 | A | No | |
| Edit Rename Drop | **teacher_id** | BTREE | No | No | teacher_id | 2 | A | Yes | |

Showing rows 0 - 4 (5 total, Query took 0.0003 seconds.)

```
SELECT * FROM `students`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

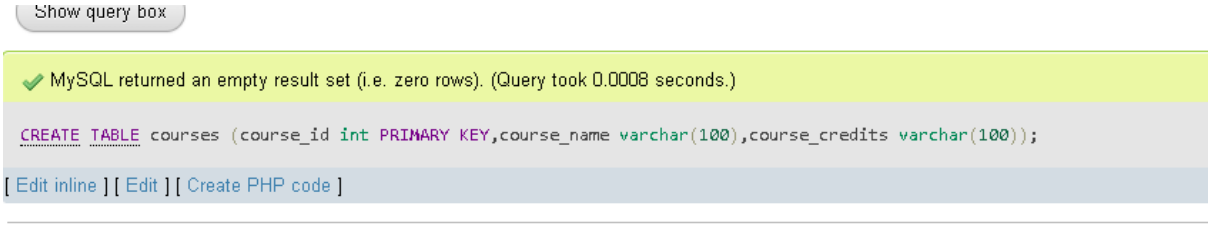
| | student_id | student_name | age | class | address | teacher_id |
|---|---|---|---|---|---|---|
| Edit Copy Delete | 1 | harsh | 22 | 3 | rajkot | *NULL* |
| Edit Copy Delete | 2 | anuradha | 19 | 1 | gondal | *NULL* |
| Edit Copy Delete | 3 | vedisha | 22 | 3 | morbi | *NULL* |
| Edit Copy Delete | 4 | sayan | 23 | 2 | bhavnagar | *NULL* |
| Edit Copy Delete | 5 | nibhay | 20 | 2 | godhara | *NULL* |

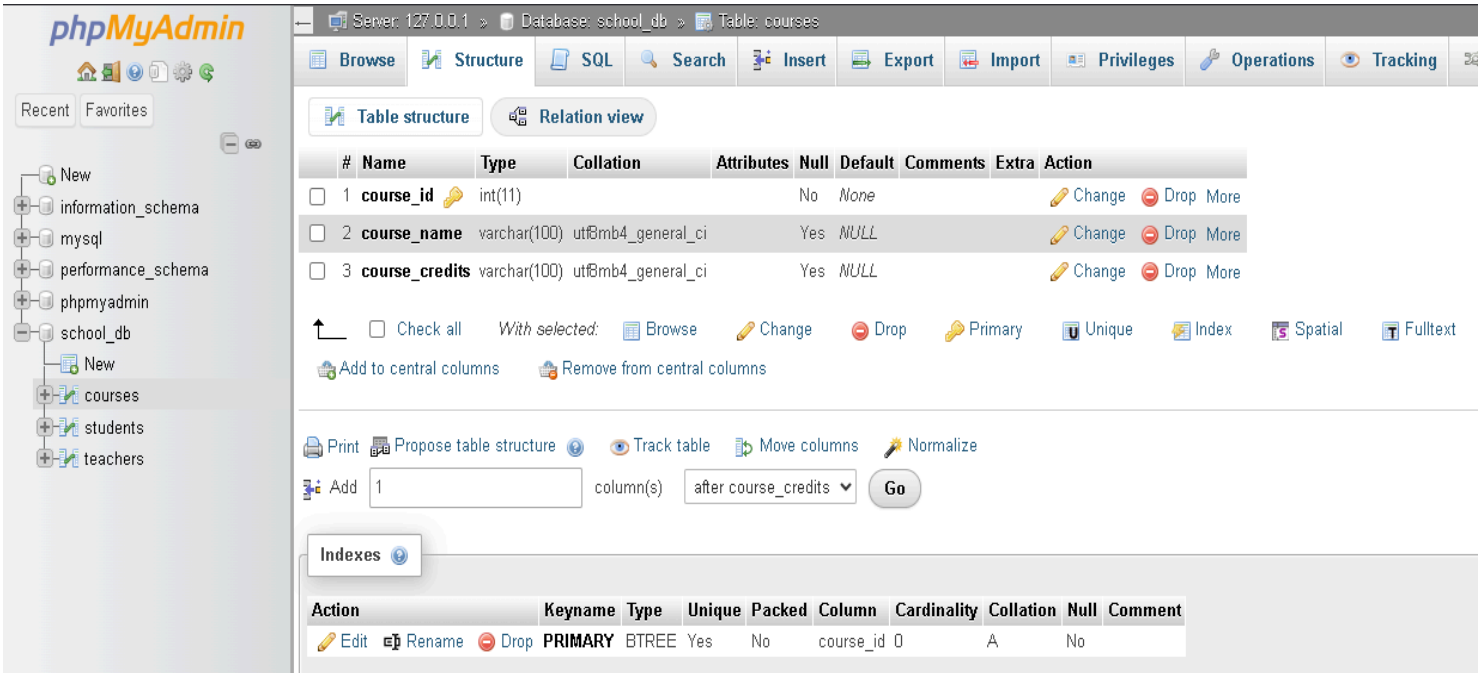
Check all *With selected:* Edit Copy Delete Export

# Main SQL Commands and Sub-commands (DDL)

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

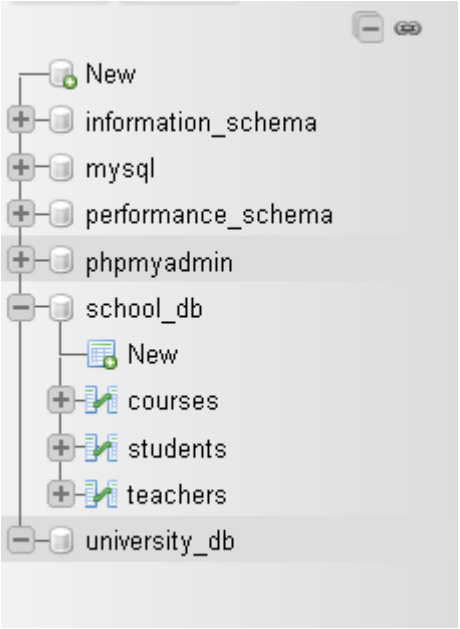
CREATE TABLE courses (course_id int PRIMARY KEY,course_name varchar(100),course_credits varchar(100))



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

CREATE DATABASE university_db



# ALTER Command

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

ALTER TABLE courses add course_duration varchar(100)

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0011 seconds.)

```
ALTER TABLE courses add course_duration varchar(100);
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | course_id | int(11) | | | No | None | | | Change Drop More |
| 2 | course_name | varchar(100) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 3 | course_credits | varchar(100) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 4 | course_duration | varchar(100) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

Add to central columns Remove from central columns

• Lab 2: Drop the course_credits column from the courses table.

ALTER TABLE courses DROP course_credits

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0013 seconds.)

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

[ Edit inline ] [ Edit ] [ Create PHP code ]

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Tr

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | course_id | int(11) | | | No | None | | | Change Drop More |
| 2 | course_name | varchar(100) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |
| 3 | course_duration | varchar(100) | utf8mb4_general_ci | | Yes | NULL | | | Change Drop More |

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

Add to central columns Remove from central columns

# DROP Command

- 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.

First we have to remove constraint key from tables
For that first we have to know what is name of key and for that go to information_schema database then go to KEY_COLUMN_USAGE
And find 'students' in table_name column and 'teacher_id' in column_name column.
This is where you can see the name of the constraint key name in the constraint_name column.

ALTER TABLE students DROP CONSTRAINT students_ibfk_1;

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0013 seconds.)

```
ALTER TABLE students DROP CONSTRAINT students_ibfk_1;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

drop TABLE teachers;
drop TABLE students

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)

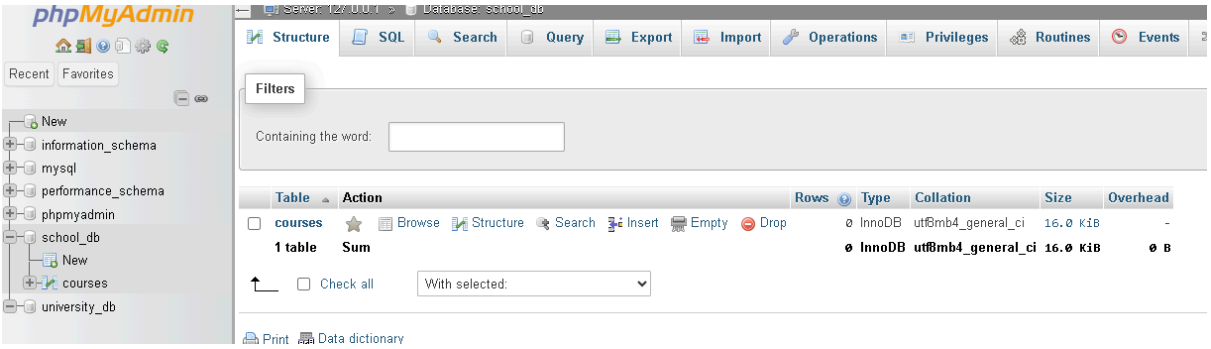
```
drop TABLE teachers;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0009 seconds.)

```
drop TABLE students;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]



# Data Manipulation Language (DML)

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

INSERT INTO courses (course_id,course_name,course_duration) VALUES (1001,'soft dev','8 months'),(1002,'web dev','6 months'),(1003,'flutter','10 months')

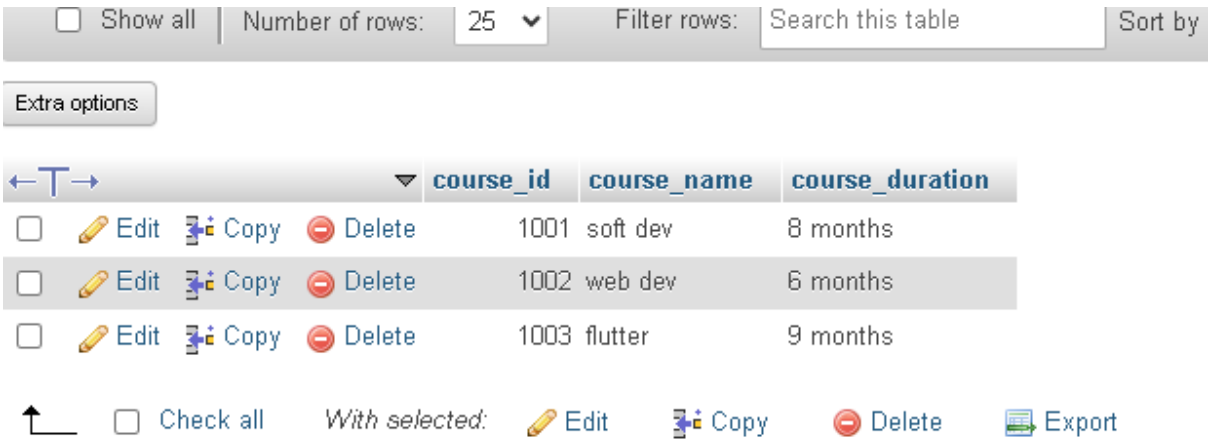
Extra options

| ←T→ | | | | course_id | course_name | course_duration |
|---|---|---|---|---|---|---|
| ☐ | Edit | Copy | Delete | 1001 | soft dev | 8 months |
| ☐ | Edit | Copy | Delete | 1002 | web dev | 6 months |
| ☐ | Edit | Copy | Delete | 1003 | flutter | 10 months |

Check all *With selected:* Edit Copy Delete Export

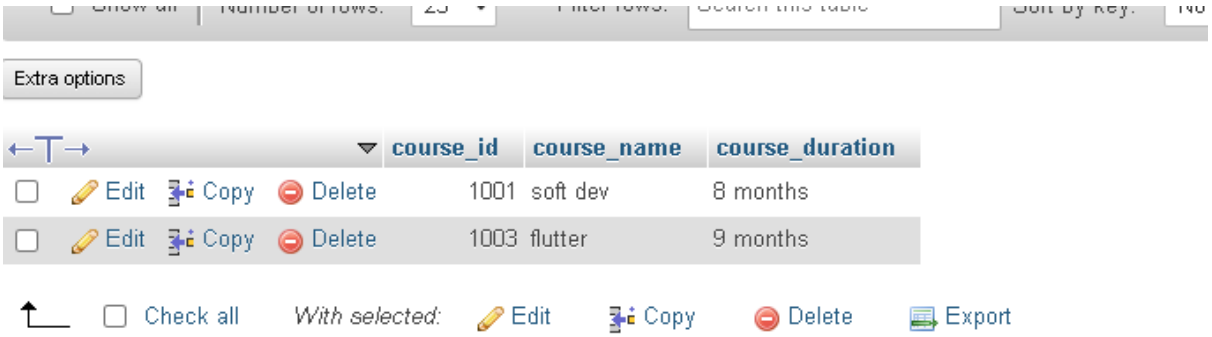
• Lab 2: Update the course duration of a specific course using the UPDATE command.

UPDATE courses SET course_duration = '9 months'
WHERE course_id = 1003;



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

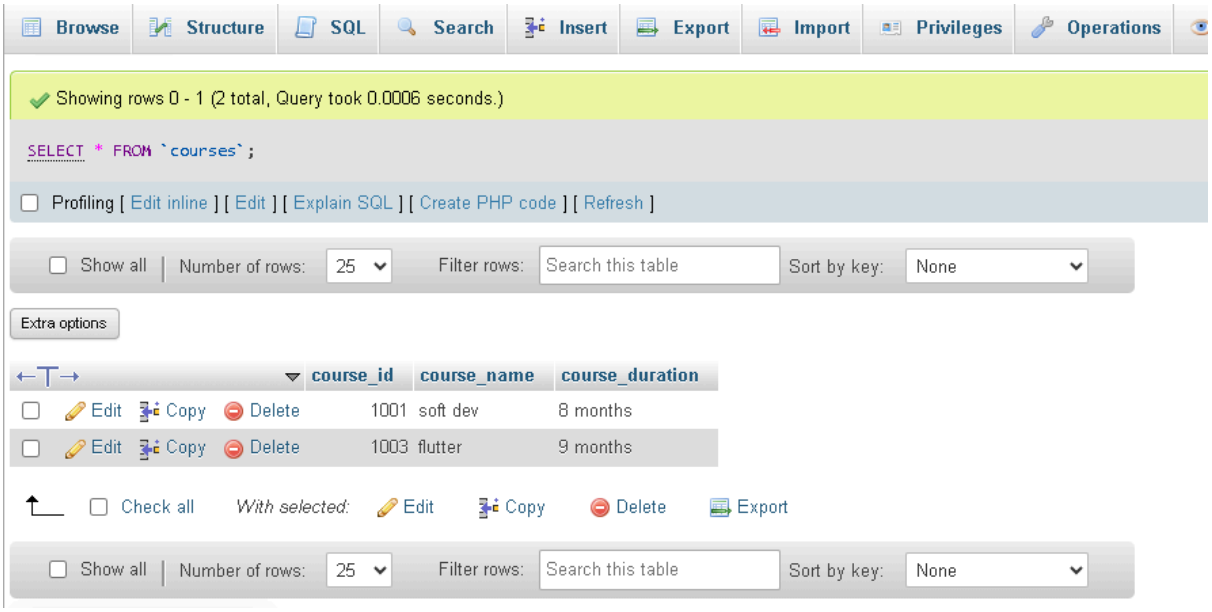
delete from courses where course_id=1002



# Data Query Language (DQL)

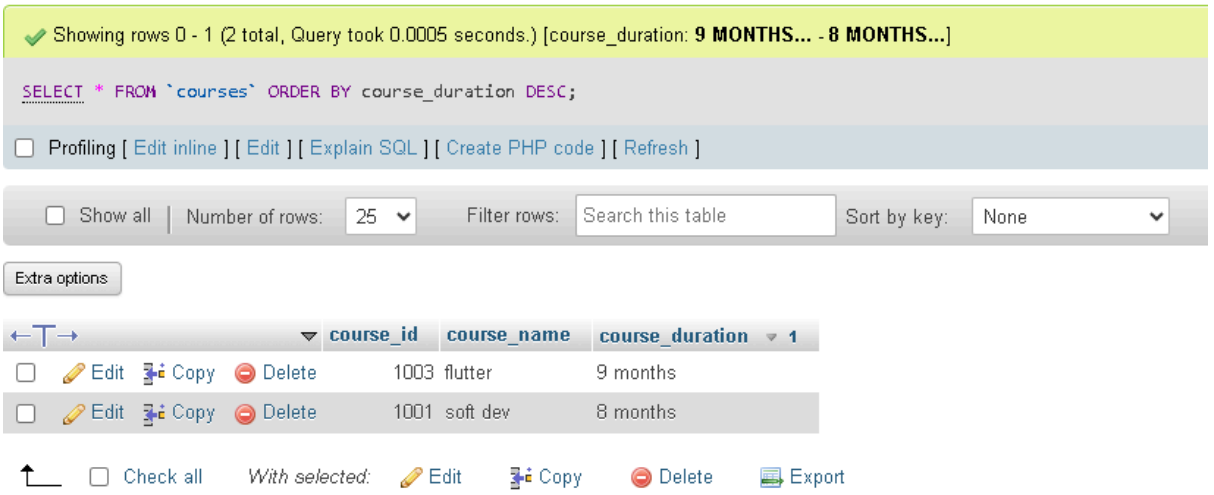
- Lab 1: Retrieve all courses from the courses table using the SELECT statement.

SELECT * FROM `courses`



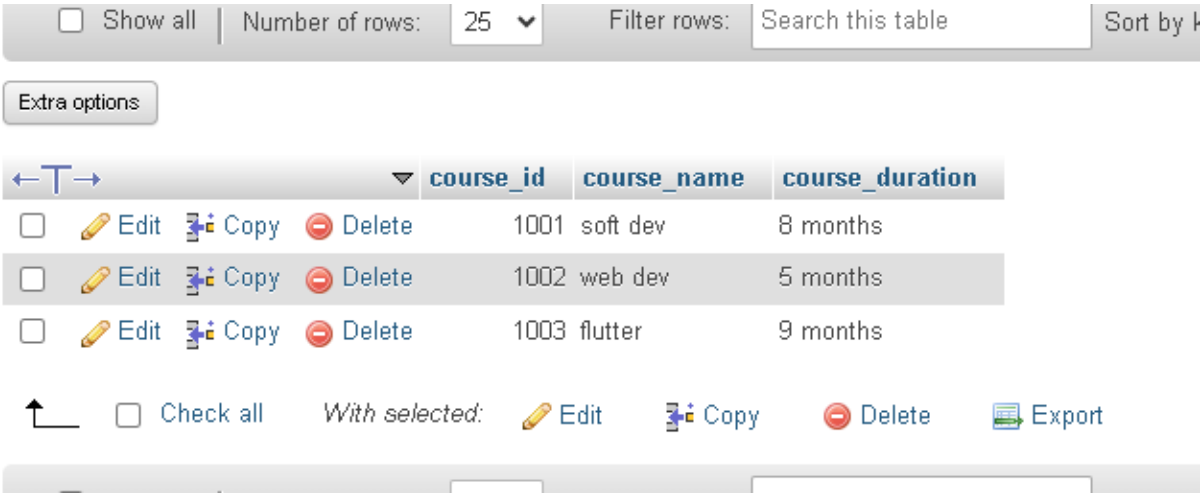
- Lab 2: Sort the courses based on course_duration in descending order using ORDER BY.

SELECT * FROM `courses` ORDER BY course_duration DESC

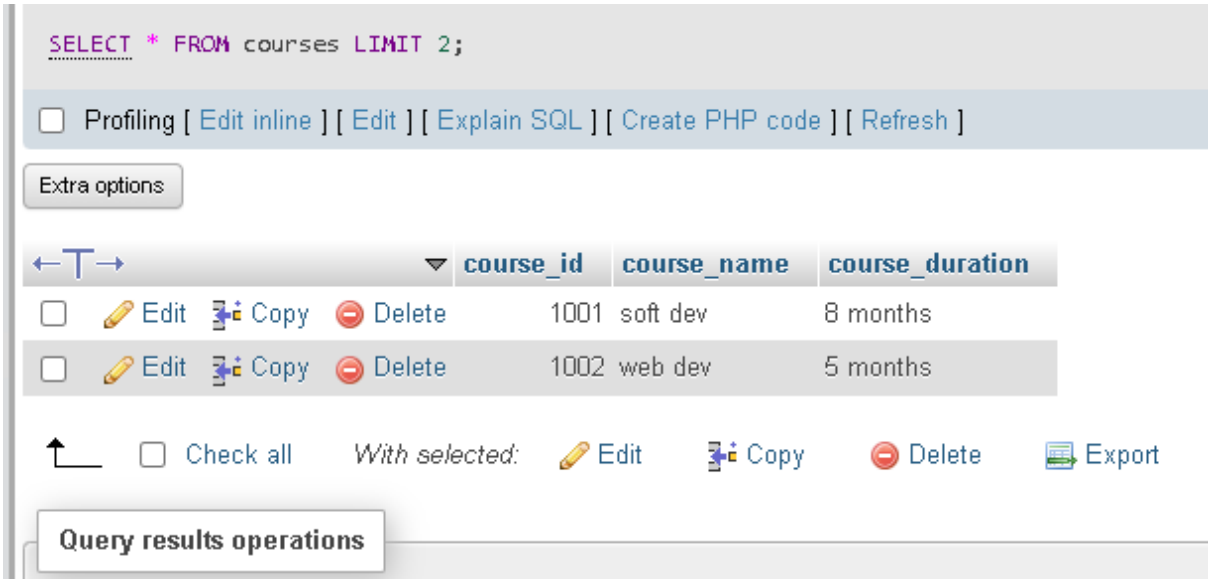


• Lab 3: Limit the results of the SELECT query to show only the top two courses using LIMIT.

Adding one entry just for more convenience




SELECT * FROM courses LIMIT 2

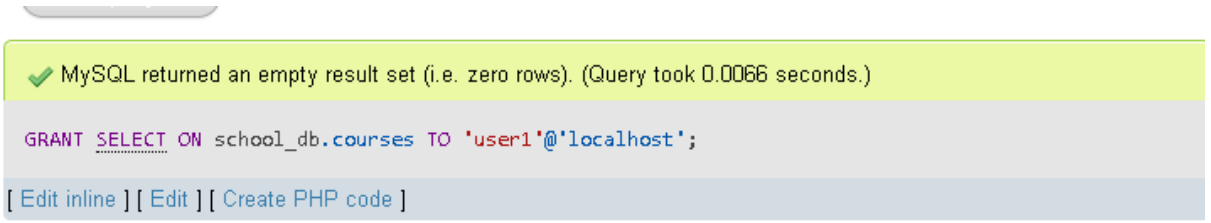


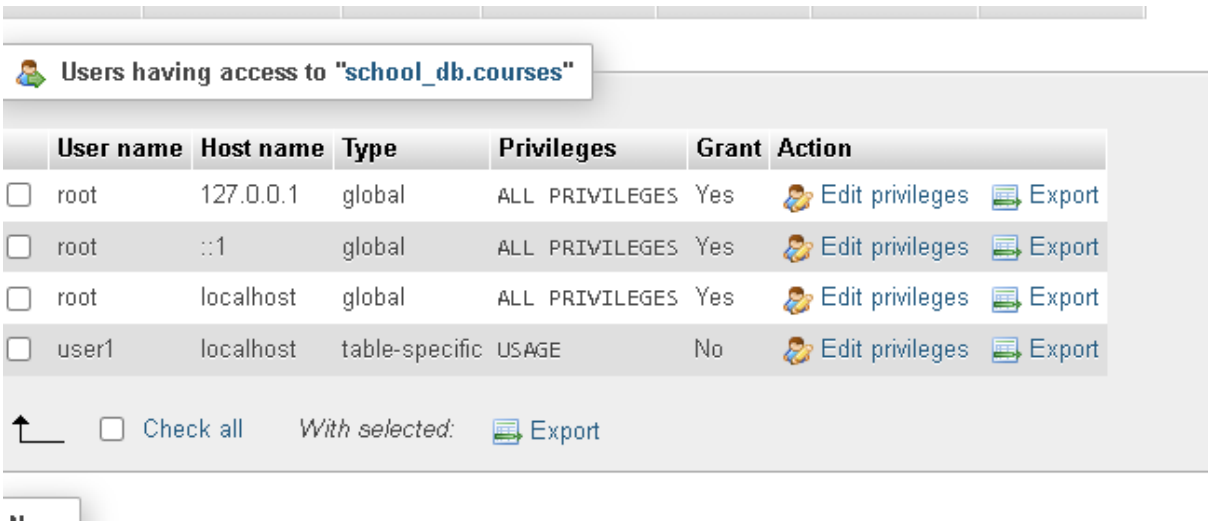
# Data Control Language (DCL)

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

CREATE USER 'user1'@'localhost' IDENTIFIED BY 'password1';
CREATE USER 'user2'@'localhost' IDENTIFIED BY 'password2';

GRANT SELECT ON school_db.courses TO 'user1'@'localhost';



Edit privileges: User account *'user1'@'localhost'* - Databases *school_db* - Table *courses*

**Table-specific privileges**

*Note: MySQL privilege names are expressed in English.*

| SELECT | INSERT | UPDATE | REFERENCES |
|---|---|---|---|
| course_id<br>course_name<br>course_duration | course_id<br>course_name<br>course_duration | course_id<br>course_name<br>course_duration | course_id<br>course_name<br>course_duration |
| Select all | Select all | Select all | Select all |
| *Or* ☐ None | *Or* ☐ None | *Or* ☐ None | *Or* ☐ None |

☐ DELETE
☐ CREATE
☐ DROP
☐ GRANT
☐ INDEX
☐ ALTER
☐ CREATE VIEW
☐ SHOW VIEW
☐ TRIGGER
☐ DELETE HISTORY

• Lab 2: Revoke the INSERT permission from user1 and give it to user2.

```
REVOKE INSERT ON school_db.courses FROM
    'user1'@'localhost';
GRANT INSERT ON school_db.courses TO
    'user2'@'localhost';
FLUSH PRIVILEGES;
```

Show query box

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0059 seconds.)

```
REVOKE INSERT ON school_db.courses FROM 'user1'@'localhost';
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0021 seconds.)

```
GRANT INSERT ON school_db.courses TO 'user2'@'localhost';
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0018 seconds.)

```
FLUSH PRIVILEGES;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Users having access to "school_db.courses"

| | User name | Host name | Type | Privileges | Grant | Action | |
|---|---|---|---|---|---|---|---|
| ☐ | root | 127.0.0.1 | global | ALL PRIVILEGES | Yes | Edit privileges | Export |
| ☐ | root | ::1 | global | ALL PRIVILEGES | Yes | Edit privileges | Export |
| ☐ | root | localhost | global | ALL PRIVILEGES | Yes | Edit privileges | Export |
| ☐ | user1 | localhost | table-specific | USAGE | No | Edit privileges | Export |
| ☐ | user2 | localhost | table-specific | USAGE | No | Edit privileges | Export |

Check all *With selected:* Export

New

Edit privileges: User account *'user2'@'localhost'* - Databases *school_db* - Table *courses*

Table-specific privileges

*Note: MySQL privilege names are expressed in English.*

| SELECT | INSERT | UPDATE | REFERENCES |
|---|---|---|---|
| course_id | course_id | course_id | course_id |
| course_name | course_name | course_name | course_name |
| course_duration | course_duration | course_duration | course_duration |
| Select all | Select all | Select all | Select all |
| Or ☐ None | Or ☐ None | Or ☐ None | Or ☐ None |

☐ DELETE
☐ CREATE
☐ DROP
☐ GRANT
☐ INDEX
☐ ALTER
☐ CREATE VIEW
☐ SHOW VIEW
☐ TRIGGER
☐ DELETE HISTORY

# Transaction Control Language (TCL)

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

START TRANSACTION;

INSERT INTO school_db.courses (course_id, course_name, course_duration) VALUES (1004, 'digi marketing','9 months'), (1005,'frontend','4 months');

COMMIT;

Show query box

✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
START TRANSACTION;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

✔ 2 rows inserted. (Query took 0.0035 seconds.)

```
INSERT INTO school_db.courses (course_id, course_name, course_duration) VALUES (1004, 'digi marketing','9 months'), (1005,'frontend','4 months');
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

✔ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0018 seconds.)

```
COMMIT;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

```
SELECT * FROM `courses`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| | course_id | course_name | course_duration |
|---|---|---|---|
| Edit Copy Delete | 1001 | soft dev | 8 months |
| Edit Copy Delete | 1002 | web dev | 5 months |
| Edit Copy Delete | 1003 | flutter | 9 months |
| Edit Copy Delete | 1004 | digi marketing | 9 months |
| Edit Copy Delete | 1005 | frontend | 4 months |

Check all *With selected:* Edit Copy Delete Export

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

START TRANSACTION;

INSERT INTO school_db.courses (course_id, course_name, course_duration) VALUES (1006, 'digi marketing','6 months'), (1007,'backend','12 months');

ROLLBACK;

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0002 seconds.)

```
START TRANSACTION;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

2 rows inserted. (Query took 0.0023 seconds.)

```
INSERT INTO school_db.courses (course_id, course_name, course_duration) VALUES (1006, 'digi marketing','6 months'), (1007,'backend','12 months');
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0014 seconds.)

```
ROLLBACK;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

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

START TRANSACTION;

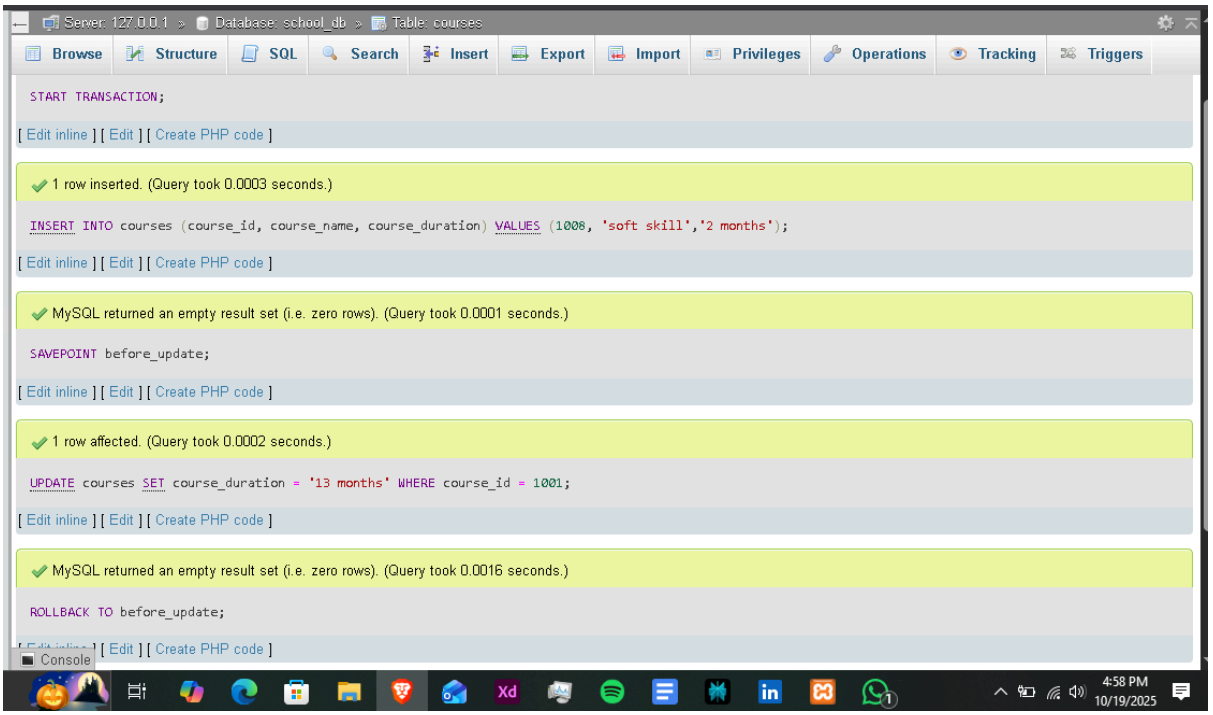
INSERT INTO courses (course_id, course_name, course_duration) VALUES (1008, 'soft skill','2 months');

SAVEPOINT before_update;

UPDATE courses SET course_duration = '13 months' WHERE course_id = 1001;

ROLLBACK TO before_update;

COMMIT;



```
SELECT * FROM `courses`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| ←T→ | | | | course_id | course_name | course_duration |
|---|---|---|---|---|---|---|
| | Edit | Copy | Delete | 1001 | soft dev | 8 months |
| | Edit | Copy | Delete | 1002 | web dev | 5 months |
| | Edit | Copy | Delete | 1003 | flutter | 9 months |
| | Edit | Copy | Delete | 1004 | digi marketing | 9 months |
| | Edit | Copy | Delete | 1005 | frontend | 4 months |
| | Edit | Copy | Delete | 1008 | soft skill | 2 months |

Check all *With selected:* Edit Copy Delete Export

# SQL Joins

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

```
SELECT * FROM `employee`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| | emp_id | emp_name | dept_id | salary |
|---|---|---|---|---|
| Edit Copy Delete | 1 | harsh | 4 | 80000 |
| Edit Copy Delete | 2 | luffy | 4 | 20000 |
| Edit Copy Delete | 3 | itachi | 1 | 120000 |
| Edit Copy Delete | 4 | sukuna | 1 | 30000 |
| Edit Copy Delete | 5 | law | 1 | 80000 |
| Edit Copy Delete | 6 | megumin | 2 | 10000 |
| Edit Copy Delete | 7 | maomao | 2 | 90000 |

Check all *With selected:* Edit Copy Delete Export

```
SELECT * FROM `departments`
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| | dept_id | dept_name |
|---|---|---|
| Edit Copy Delete | 1 | bata |
| Edit Copy Delete | 2 | jbl |
| Edit Copy Delete | 3 | souled |
| Edit Copy Delete | 4 | anya |

Check all *With selected:* Edit Copy Delete Export

ALTER TABLE employee ADD CONSTRAINT fk_dept FOREIGN KEY (dept_id) REFERENCES departments(dept_id);

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | emp_id | int(11) | | | No | None | | AUTO_INCREMENT | Change Drop More |
| 2 | emp_name | varchar(100) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 3 | dept_id | int(11) | | | No | None | | | Change Drop More |
| 4 | salary | int(11) | | | No | None | | | Change Drop More |

Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext

SELECT employee.emp_id,employee.emp_name,departments.dept_name FROM employee
INNER JOIN departments
ON employee.dept_id = departments.dept_id;

Showing rows 0 - 6 (7 total, Query took 0.0003 seconds.)

```
SELECT employee.emp_id,employee.emp_name,departments.dept_name FROM employee INNER JOIN departments ON employee.dept_id = departments.dept_id;
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| emp_id | emp_name | dept_name |
|---|---|---|
| 1 | harsh | anya |
| 2 | luffy | anya |
| 3 | itachi | bata |
| 4 | sukuna | bata |
| 5 | law | bata |
| 6 | megumin | jbl |
| 7 | maomao | jbl |

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

```
SELECT
    departments.dept_id,departments.dept_name,employee.emp_id,employee.emp_name FROM departments
LEFT JOIN employee
ON departments.dept_id = employee.dept_id;
```

```
SELECT departments.dept_id,departments.dept_name,employee.emp_id,employee.emp_name FROM departments LEFT JOIN employee ON departments.dept_id =
employee.dept_id;
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

Extra options

| dept_id | dept_name | emp_id | emp_name |
|---|---|---|---|
| 1 | bata | 3 | itachi |
| 1 | bata | 4 | sukuna |
| 1 | bata | 5 | law |
| 2 | jbl | 6 | megumin |
| 2 | jbl | 7 | maomao |
| 3 | souled | NULL | NULL |
| 4 | anya | 1 | harsh |
| 4 | anya | 2 | luffy |

# SQL Group By

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

SELECT dept_id,COUNT(emp_id) AS total_emp FROM employee GROUP BY dept_id

Showing rows 0 - 2 (3 total, Query took 0.0002 seconds.)

```
SELECT dept_id,COUNT(emp_id) AS total_emp FROM employee GROUP BY dept_id;
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

| dept_id | total_emp |
|---|---|
| 1 | 3 |
| 2 | 2 |
| 4 | 2 |

Show all | Number of rows: 25 Filter rows: Search this table

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

SELECT dept_id,AVG(salary) AS avg_salary FROM employee GROUP by dept_id

```
SELECT dept_id,AVG(salary) AS avg_salary FROM employee GROUP by dept_id;
```

Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

| | | | dept_id | avg_salary |
|---|---|---|---|---|
| Edit | Copy | Delete | 1 | 76666.6667 |
| Edit | Copy | Delete | 2 | 50000.0000 |
| Edit | Copy | Delete | 4 | 50000.0000 |

Check all *With selected:* Edit Copy Delete Export

# SQL Stored Procedure

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

```
DELIMITER $$

CREATE PROCEDURE GetEmps(IN dept INT)
BEGIN
    SELECT emp_id, emp_name,salary, dept_id
    FROM employee
    WHERE dept_id = dept;
END $$

DELIMITER ;
```

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0059 seconds.)

```
CREATE PROCEDURE GetEmps(IN dept INT) BEGIN SELECT emp_id, emp_name,salary, dept_id FROM employee WHERE dept_id = dept; END;
```

[ Edit inline ] [ Edit ] [ Create PHP code ]

Check all | Export | Drop

| | Name | Type | Returns | |
|---|---|---|---|---|
| ☐ | GetEmps | PROCEDURE | | Edit Execute Export Drop |

Execute routine `GetEmps`

Routine parameters

| Name | Type | Function | Value |
|---|---|---|---|
| dept | INT | | 1 |

Go Close

Your SQL query has been executed successfully.
3 rows affected by the last statement inside the procedure.

```
SET @p0='1'; CALL `GetEmps`(@p0);
```

Execution results of routine `GetEmps`

| emp_id | emp_name | salary | dept_id |
|---|---|---|---|
| 3 | itachi | 120000 | 1 |
| 4 | sukuna | 30000 | 1 |
| 5 | law | 80000 | 1 |

Routines

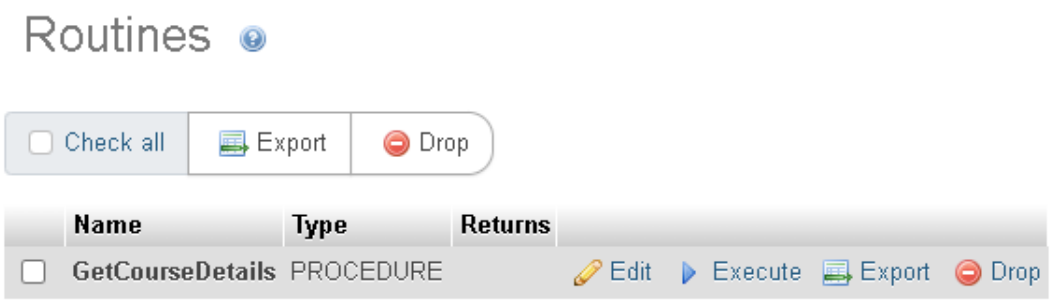
Check all | Export | Drop

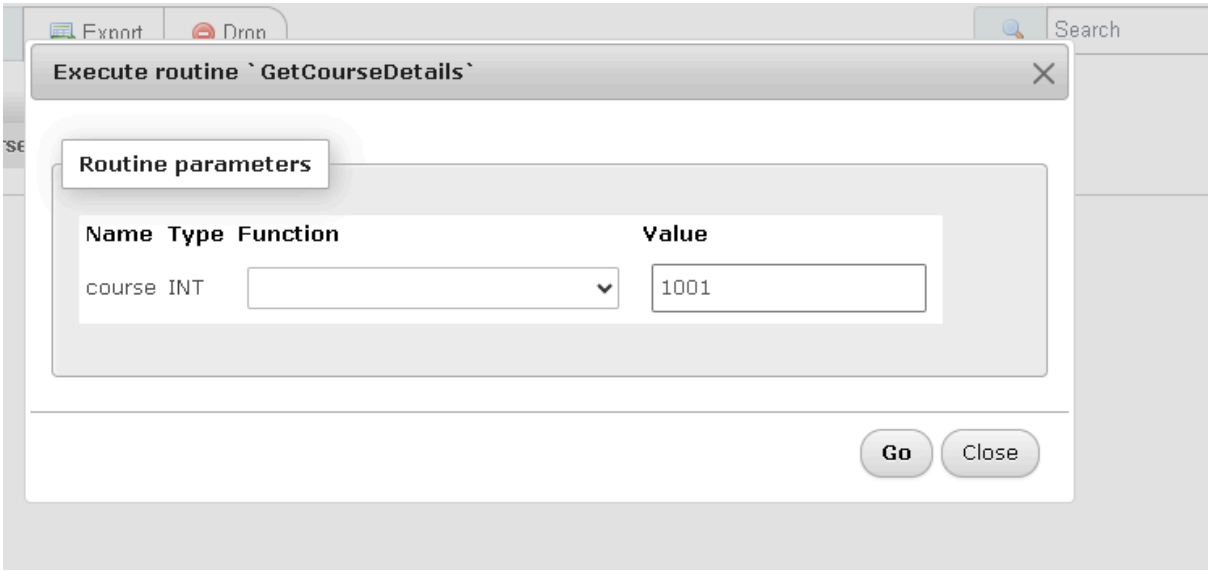
Search | Create new routine

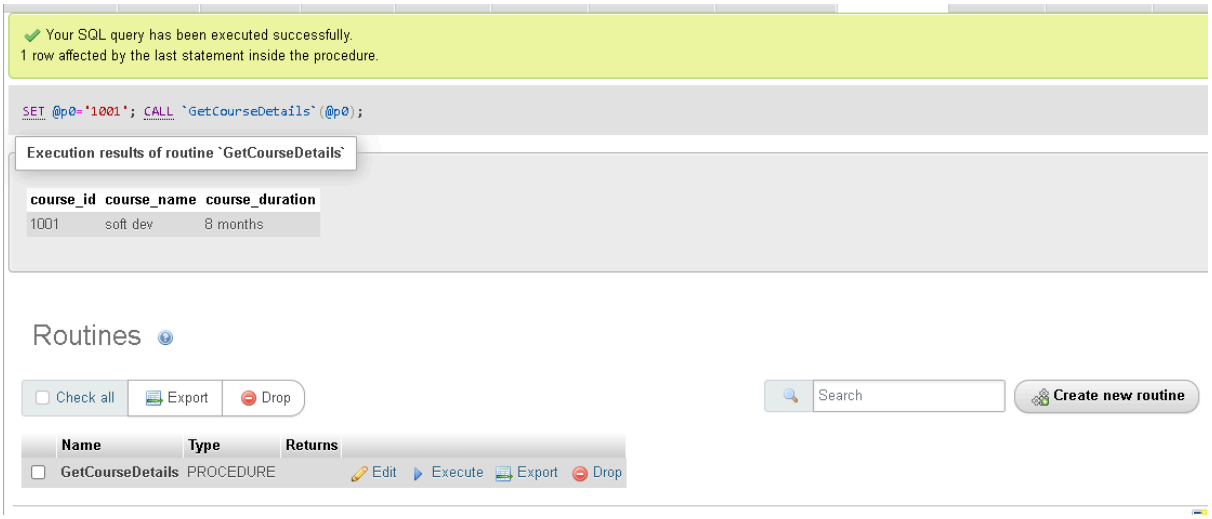
| | Name | Type | Returns | |
|---|---|---|---|---|
| ☐ | GetEmps | PROCEDURE | | Edit Execute Export Drop |

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

```
DELIMITER //

CREATE PROCEDURE GetCourseDetails(IN course INT)
BEGIN
    SELECT *
    FROM courses
    WHERE course_id = course;
END //

DELIMITER ;
```



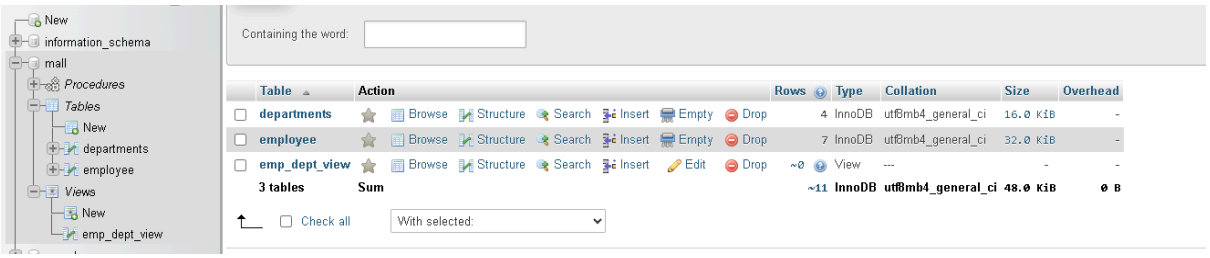

Export
Drop
Search
Execute routine `GetCourseDetails`
Routine parameters
Name Type Function
Value
course INT
1001
Go
Close

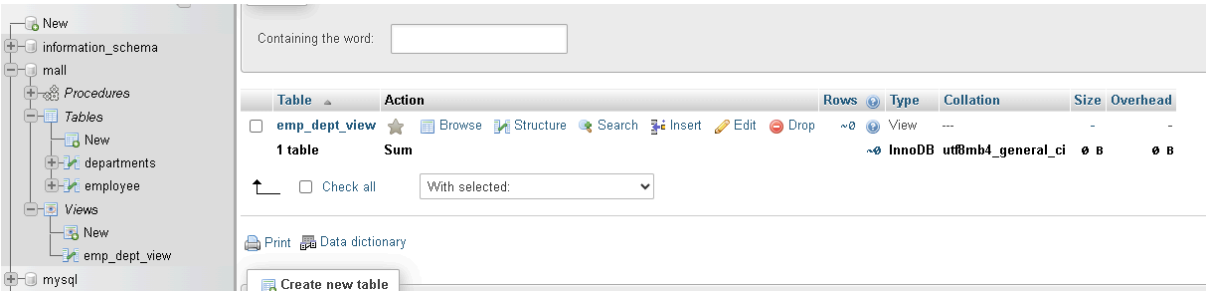

Your SQL query has been executed successfully.
1 row affected by the last statement inside the procedure.
SET @p0='1001'; CALL `GetCourseDetails`(@p0);
Execution results of routine `GetCourseDetails`
course_id course_name course_duration
1001 soft dev 8 months
Routines
Check all
Export
Drop
Search
Create new routine
Name Type Returns
GetCourseDetails PROCEDURE
Edit
Execute
Export
Drop

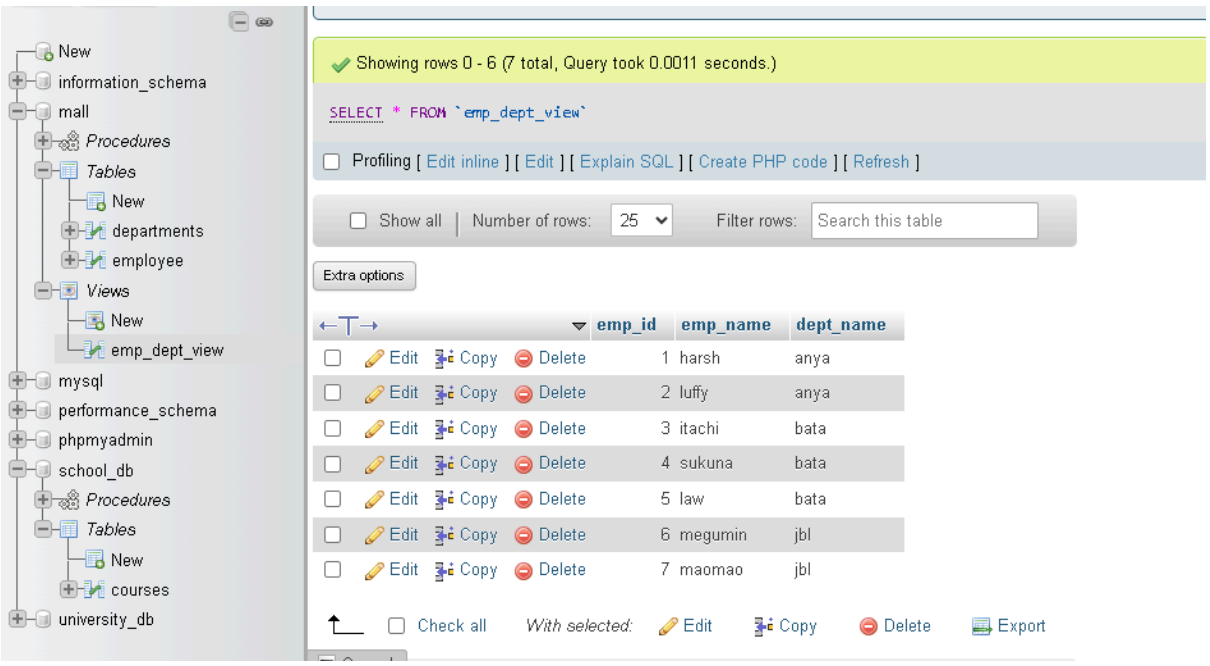
# SQL View

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

```
CREATE VIEW emp_dept_view AS
SELECT employee.emp_id, employee.emp_name,
    departments.dept_name
FROM employee
INNER JOIN departments
ON employee.dept_id = departments.dept_id;
```

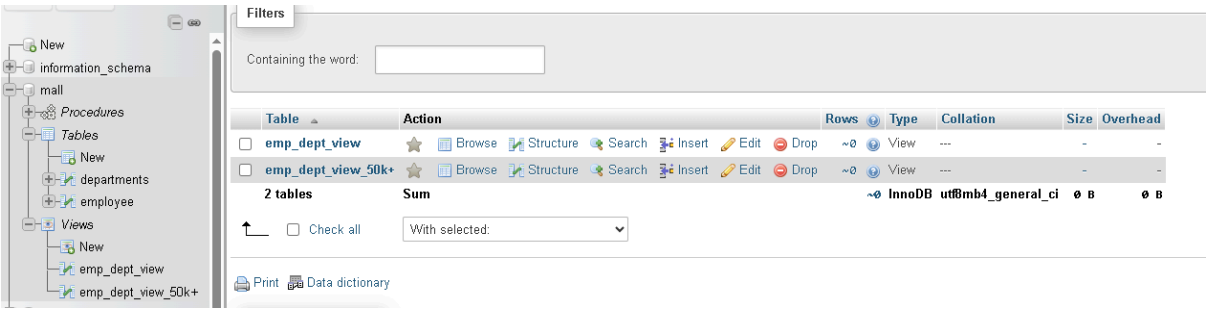


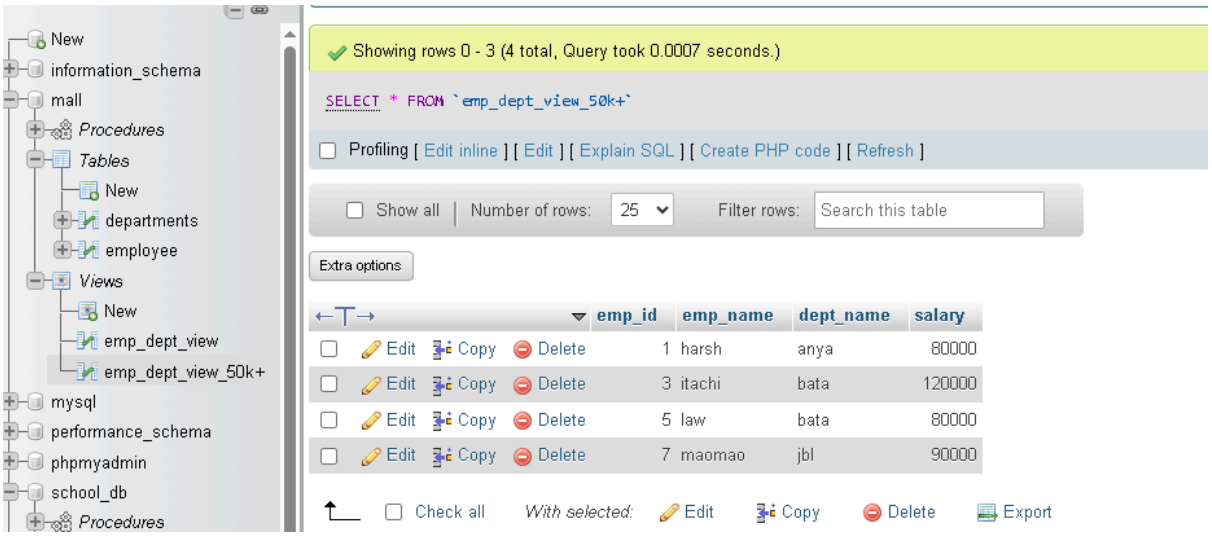




New
information_schema
mall
Procedures
Tables
New
departments
employee
Views
New
emp_dept_view
mysql
performance_schema
phpmyadmin
school_db
Procedures
Tables
New
courses
university_db
Showing rows 0 - 6 (7 total, Query took 0.0011 seconds.)
SELECT * FROM `emp_dept_view`
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
Show all | Number of rows: 25 Filter rows: Search this table
Extra options
emp_id emp_name dept_name
Edit Copy Delete 1 harsh anya
Edit Copy Delete 2 luffy anya
Edit Copy Delete 3 itachi bata
Edit Copy Delete 4 sukuna bata
Edit Copy Delete 5 law bata
Edit Copy Delete 6 megumin jbl
Edit Copy Delete 7 maomao jbl
Check all With selected: Edit Copy Delete Export

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

```
CREATE VIEW emp_dept_view_less_50k+ AS
SELECT employee.emp_id, employee.emp_name,
    departments.dept_name, employee.salary
FROM employee
INNER JOIN departments
ON employee.dept_id = departments.dept_id
WHERE employee.salary >= 50000;
```

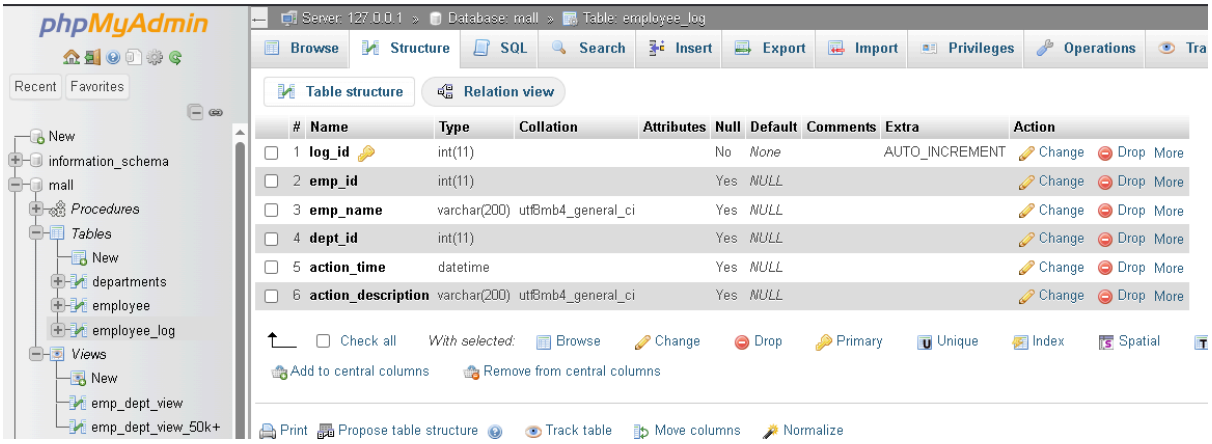


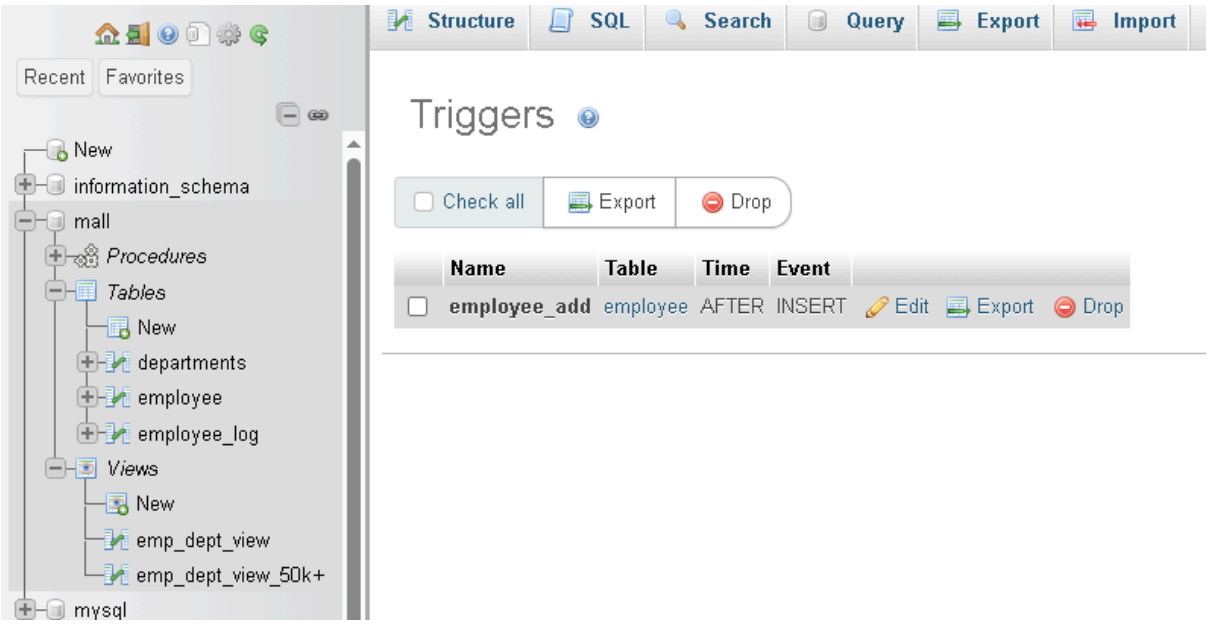
# SQL Triggers

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

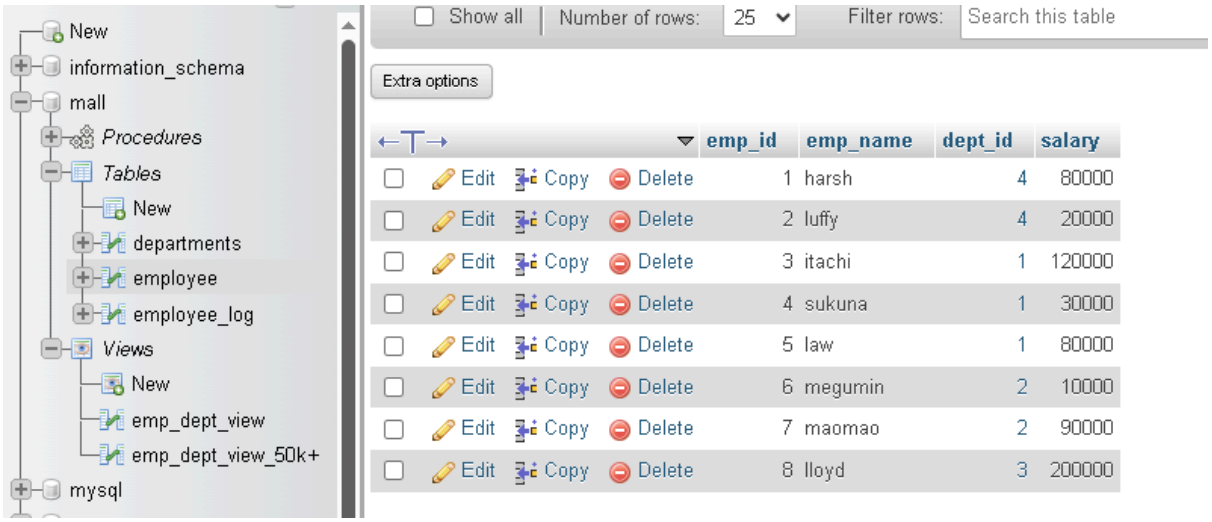
```
CREATE TABLE employee_log
(
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    emp_id INT, emp_name VARCHAR(200), dept_id INT,
    action_time DATETIME,action_description
    VARCHAR(200)
);
```



```
DELIMITER $$

CREATE TRIGGER employee_add
AFTER INSERT ON employee
FOR EACH ROW
BEGIN
    INSERT INTO employee_log (emp_id, emp_name,
    dept_id, action_time, action_description)
    VALUES (NEW.emp_id, NEW.emp_name,
    NEW.dept_id, NOW(), 'new employee added');
END $$

DELIMITER ;
```



New
information_schema
mall
Procedures
Tables
New
departments
employee
employee_log
Views
New
emp_dept_view
emp_dept_view_50k+
SELECT * FROM `employee_log`
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
Show all | Number of rows: 25 Filter rows: Search this table
Extra options
log_id emp_id emp_name dept_id action_time action_description
Edit Copy Delete 1 8 lloyd 3 2025-10-24 20:30:02 new employee added
Check all With selected: Edit Copy Delete Export
Show all | Number of rows: 25 Filter rows: Search this table

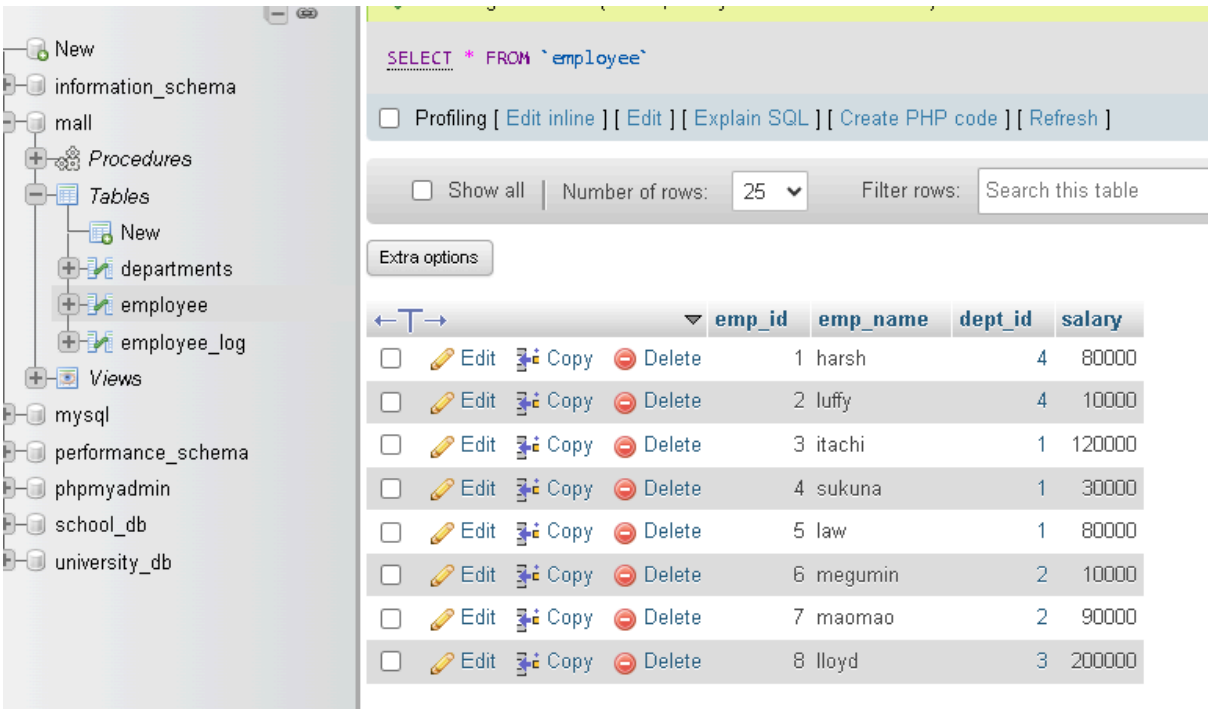

New
information_schema
mall
Procedures
Tables
New
departments
employee
employee_log
Views
New
emp_dept_view
emp_dept_view_50k+
mysql
Show all | Number of rows: 25 Filter rows: Search this table
Extra options
emp_id emp_name dept_id salary
Edit Copy Delete 1 harsh 4 80000
Edit Copy Delete 2 luffy 4 20000
Edit Copy Delete 3 itachi 1 120000
Edit Copy Delete 4 sukuna 1 30000
Edit Copy Delete 5 law 1 80000
Edit Copy Delete 6 megumin 2 10000
Edit Copy Delete 7 maomao 2 90000
Edit Copy Delete 8 lloyd 3 200000

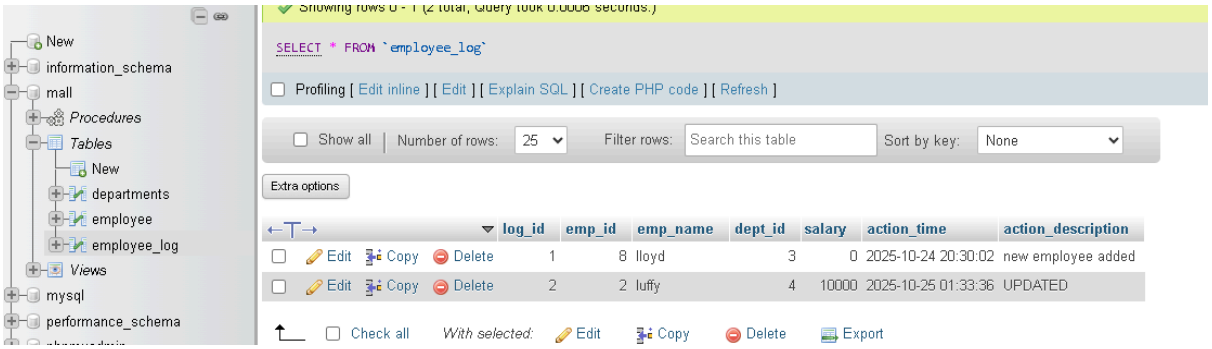
• Lab 2: Create a trigger to update the last_modified timestamp whenever an employee record is updated.

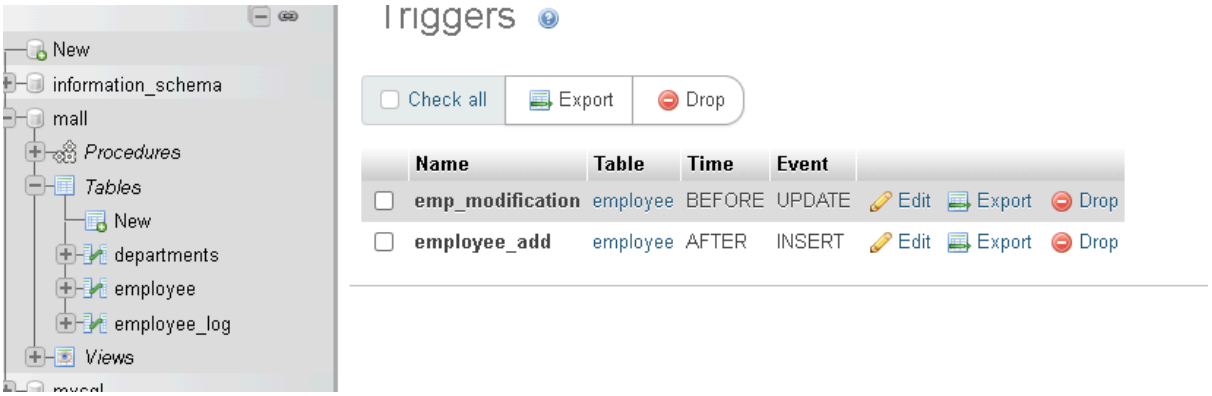
```
DELIMITER $$

    CREATE TRIGGER emp_modification
    BEFORE UPDATE ON employee
    FOR EACH ROW

    BEGIN
     INSERT INTO employee_log (emp_id, emp_name,
    dept_id, salary, action_time, action_description)
     VALUES (NEW.emp_id, NEW.emp_name,
    NEW.dept_id, NEW.salary, NOW(),'UPDATED');

    END$$

    DELIMITER ;
```

For checking

update employee set salary=10000 where emp_id=2


New
information_schema
mall
Procedures
Tables
New
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
SELECT * FROM `employee`
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
Show all | Number of rows: 25 Filter rows: Search this table
Extra options
emp_id emp_name dept_id salary
Edit Copy Delete 1 harsh 4 80000
Edit Copy Delete 2 luffy 4 10000
Edit Copy Delete 3 itachi 1 120000
Edit Copy Delete 4 sukuna 1 30000
Edit Copy Delete 5 law 1 80000
Edit Copy Delete 6 megumin 2 10000
Edit Copy Delete 7 maomao 2 90000
Edit Copy Delete 8 lloyd 3 200000

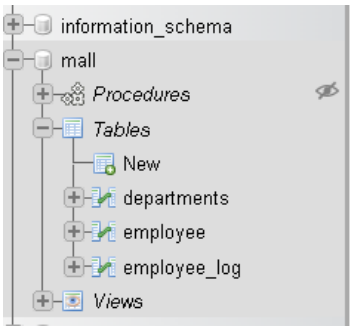

New
information_schema
mall
Procedures
Tables
New
departments
employee
employee_log
Views
mysql
performance_schema
SELECT * FROM `employee_log`
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options
log_id emp_id emp_name dept_id salary action_time action_description
Edit Copy Delete 1 8 lloyd 3 0 2025-10-24 20:30:02 new employee added
Edit Copy Delete 2 2 luffy 4 10000 2025-10-25 01:33:36 UPDATED
Check all With selected: Edit Copy Delete Export

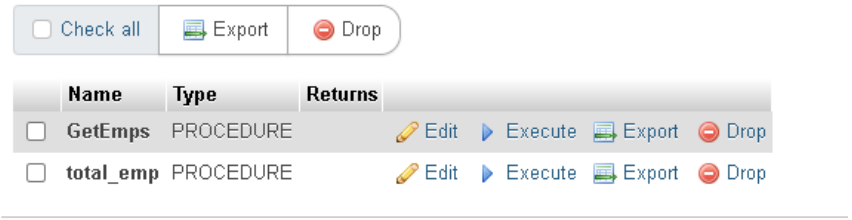

New
information_schema
mall
Procedures
Tables
New
departments
employee
employee_log
Views
Triggers
Check all Export Drop
Name Table Time Event
emp_modification employee BEFORE UPDATE Edit Export Drop
employee_add employee AFTER INSERT Edit Export Drop

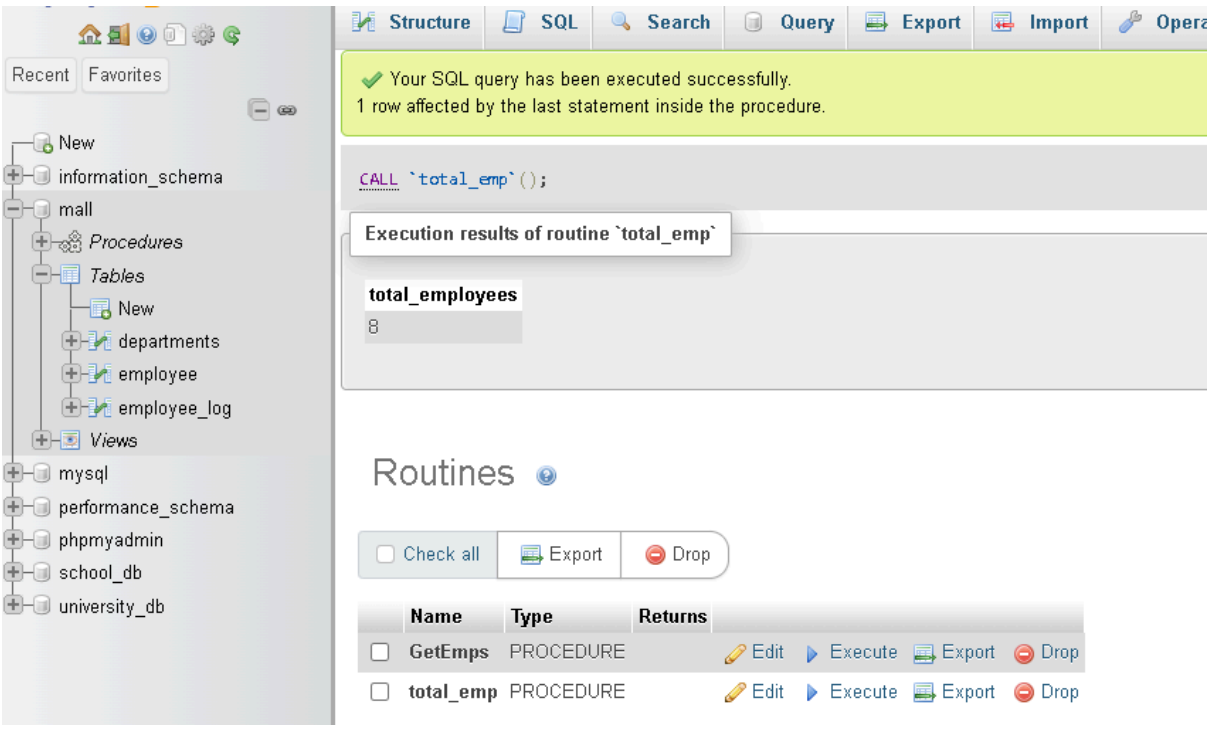
# Introduction to PL/SQL

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

```
DELIMITER $$

CREATE PROCEDURE total_emp()
BEGIN
  DECLARE total_employees INT;

  SELECT COUNT(emp_id) AS total_employees FROM
employee;


END$$

DELIMITER ;
```




Structure
SQL
Search
Query
Export
Import
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
Your SQL query has been executed successfully.
1 row affected by the last statement inside the procedure.
CALL `total_emp`();
Execution results of routine `total_emp`
total_employees
8
Routines
Check all
Export
Drop
Name
Type
Returns
GetEmps
PROCEDURE
Edit
Execute
Export
Drop
total_emp
PROCEDURE
Edit
Execute
Export
Drop

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

## Customer table structure

Server: 127.0.0.1 » Database: mall » Table: customer

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Triggers

Table structure | Relation view

| # | Name | Type | Collation | Attributes | Null | Default | Comments | Extra | Action |
|---|---|---|---|---|---|---|---|---|---|
| 1 | cust_id | int(11) | | | No | None | | AUTO_INCREMENT | Change Drop More |
| 2 | cust_name | varchar(200) | utf8mb4_general_ci | | No | None | | | Change Drop More |
| 3 | buying_time | datetime | | | No | None | | | Change Drop More |
| 4 | total_bill | int(11) | | | No | None | | | Change Drop More |

Check all | With selected: Browse | Change | Drop | Primary | Unique | Index | Spatial | Fulltext

Add to central columns | Remove from central columns

Showing rows 0 - 4 (5 total, Query took 0.0004 seconds.)

```
SELECT * FROM `customer`
```

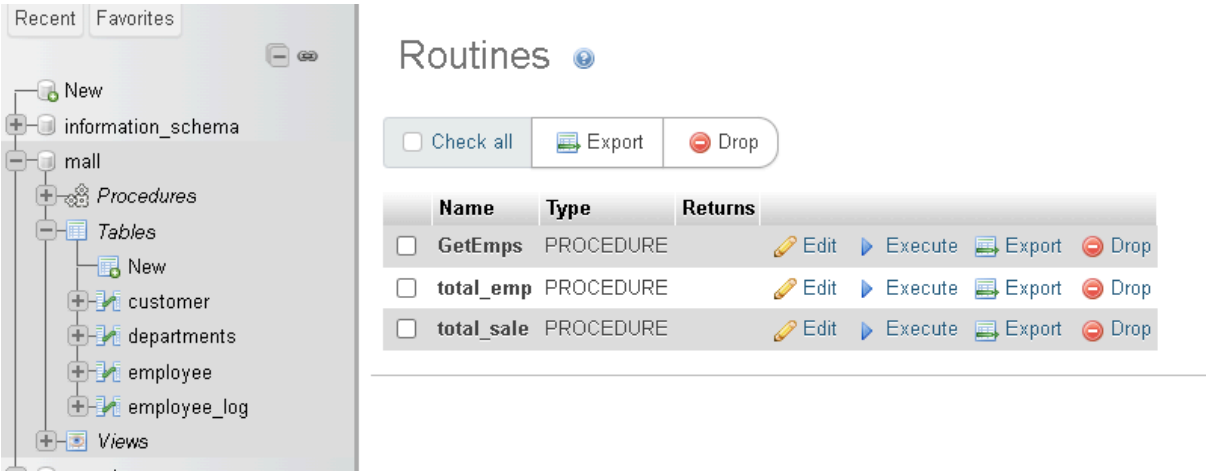
Profiling [ Edit inline ] [ Edit ] [ Explain SQL ] [ Create PHP code ] [ Refresh ]

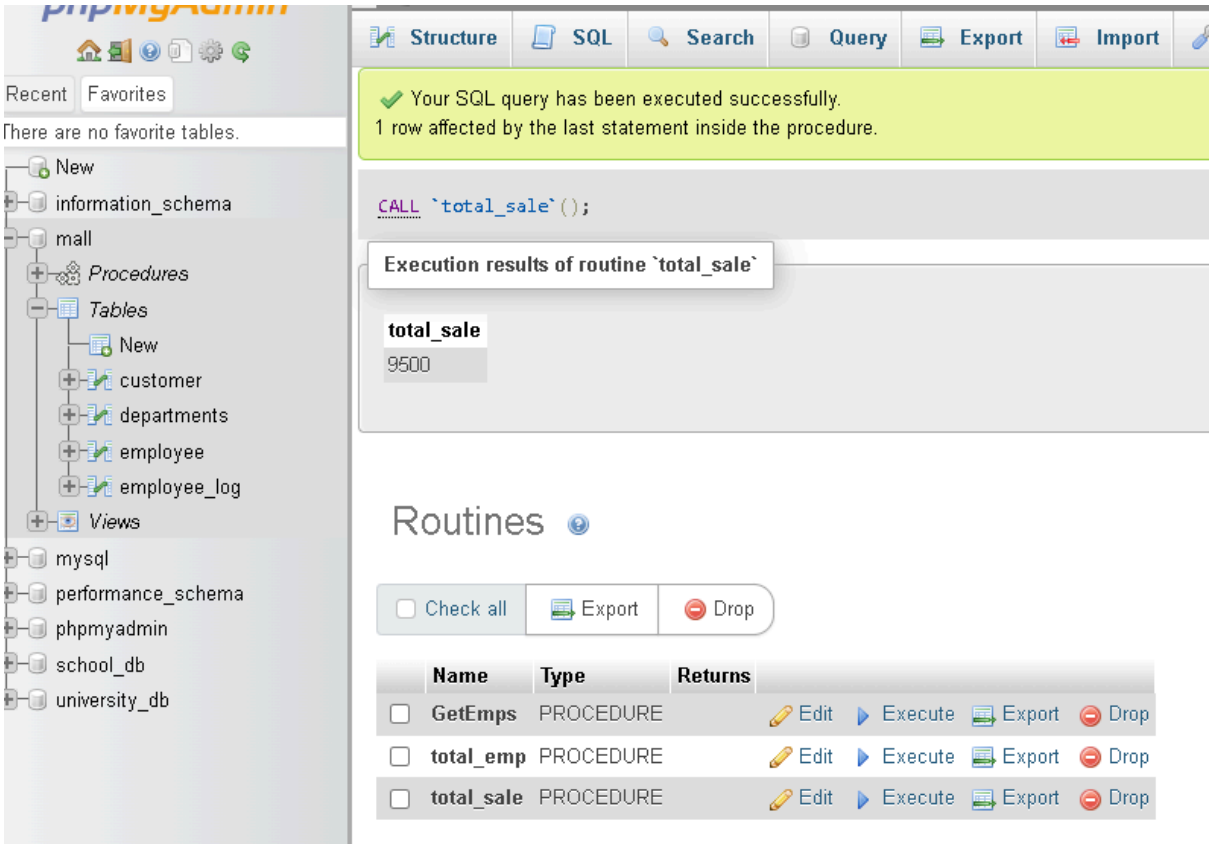
Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

| | cust_id | cust_name | buying_time | total_bill |
|---|---|---|---|---|
| Edit Copy Delete | 1 | ved | 2025-10-24 22:25:25 | 2000 |
| Edit Copy Delete | 2 | abhay | 2025-10-24 22:25:25 | 1500 |
| Edit Copy Delete | 3 | himu | 2025-10-24 22:25:25 | 3000 |
| Edit Copy Delete | 4 | anu | 2025-10-24 22:25:25 | 1000 |
| Edit Copy Delete | 5 | sweta | 2025-10-24 22:25:25 | 2000 |

```
DELIMITER $$

CREATE PROCEDURE total_sale()
BEGIN

  SELECT SUM(total_bill) AS total_sale FROM customer;
END$$

DELIMITER ;
```

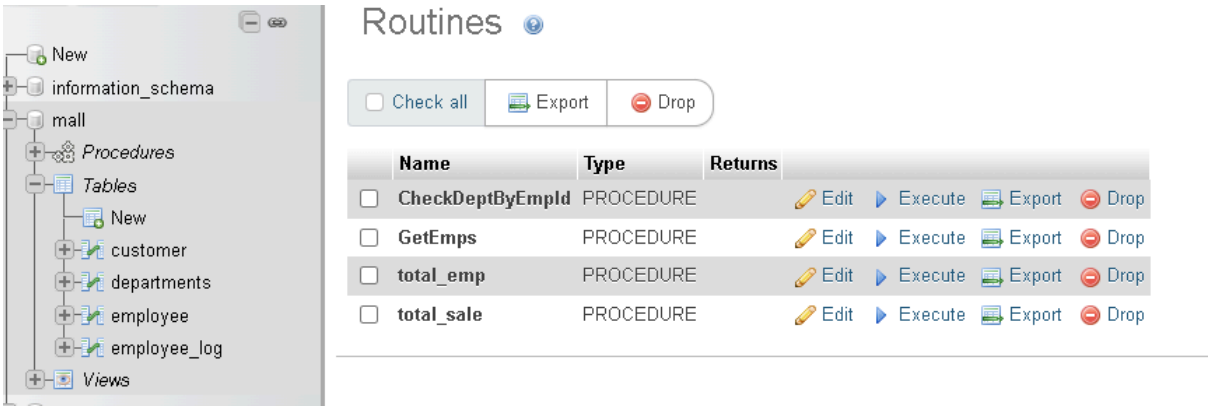

Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
Routines
Check all
Export
Drop
Name
Type
Returns
GetEmps PROCEDURE Edit Execute Export Drop
total_emp PROCEDURE Edit Execute Export Drop
total_sale PROCEDURE Edit Execute Export Drop

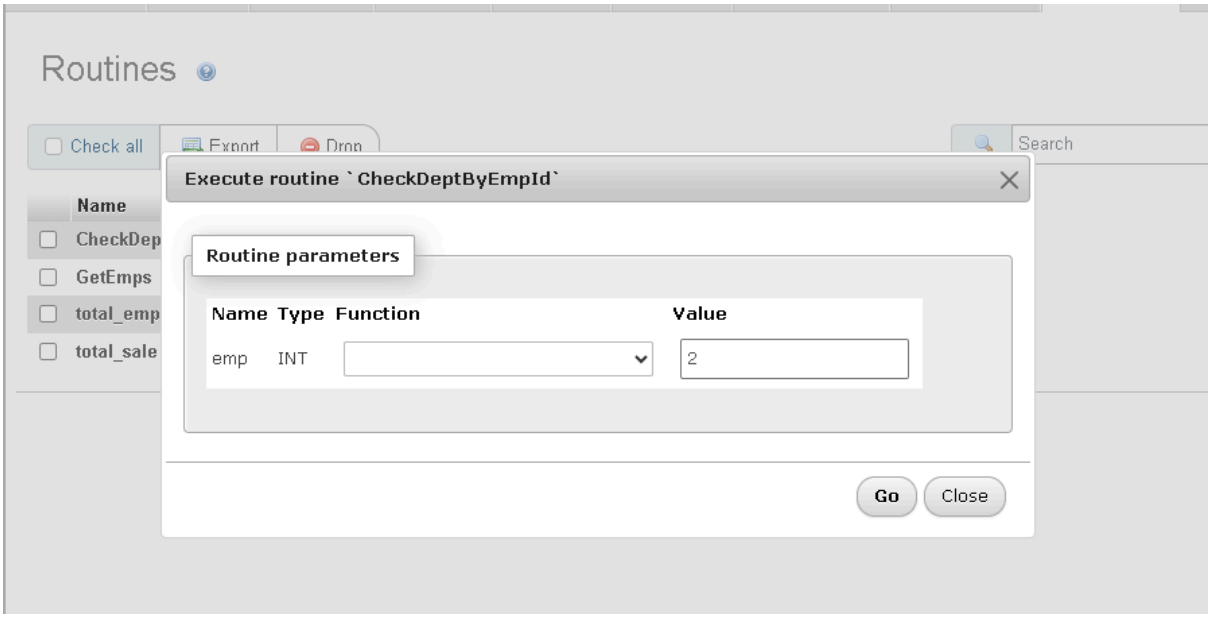

Structure
SQL
Search
Query
Export
Import
Recent
Favorites
There are no favorite tables.
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
Your SQL query has been executed successfully.
1 row affected by the last statement inside the procedure.
CALL `total_sale`();
Execution results of routine `total_sale`
total_sale
9500
Routines
Check all
Export
Drop
Name
Type
Returns
GetEmps PROCEDURE Edit Execute Export Drop
total_emp PROCEDURE Edit Execute Export Drop
total_sale PROCEDURE Edit Execute Export Drop

# PL/SQL Control Structures

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

```
DELIMITER $$

CREATE PROCEDURE CheckDeptByEmpId(IN emp INT)
BEGIN
  DECLARE temp_name VARCHAR(100);

  SELECT d.dept_name INTO temp_name
  FROM employee e
  JOIN departments d ON e.dept_id = d.dept_id
  WHERE e.emp_id = emp;

  IF temp_name IS NOT NULL THEN
    SELECT CONCAT('Employee is in ', temp_name) AS
Message;
  ELSE
    SELECT 'Employee id is invalid ' AS Message;
  END IF;

END $$

DELIMITER ;
```


New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
Routines
Check all
Export
Drop
Name
Type
Returns
CheckDeptByEmpId
PROCEDURE
Edit
Execute
Export
Drop
GetEmps
PROCEDURE
total_emp
PROCEDURE
total_sale
PROCEDURE


Routines
Check all
Search
Execute routine `CheckDeptByEmpId`
Routine parameters
Name
Type
Function
Value
emp
INT
2
Go
Close

Recent Favorites

- New
- information_schema
- mall
  - Procedures
  - Tables
    - New
    - customer
    - departments
    - employee
    - employee_log
  - Views
- mysql
- performance_schema
- phpmyadmin
- school_db
- university_db

Structure | SQL | Search | Query | Export | Import | Operati

Your SQL query has been executed successfully.
1 row affected by the last statement inside the procedure.

```
SET @p0='2'; CALL `CheckDeptByEmpId`(@p0);
```

Execution results of routine `CheckDeptByEmpId`

| Message |
|---|
| Employee is in anya |

## Routines

Check all | Export | Drop

| Name | Type | Returns | |
|---|---|---|---|
| CheckDeptByEmpId | PROCEDURE | | Edit Execute Export Drop |
| GetEmps | PROCEDURE | | Edit Execute Export Drop |
| total_emp | PROCEDURE | | Edit Execute Export Drop |
| total_sale | PROCEDURE | | Edit Execute Export Drop |

phpMyAdmin

Recent Favorites

- New
- information_schema
- mall
  - Procedures
  - Tables
    - New
    - customer
    - departments
    - employee
    - employee_log
  - Views
- mysql
- performance_schema
- phpmyadmin
- school_db
- university_db

Server: 127.0.0.1 » Database: mall

Structure | SQL | Search | Query | Export | Import | Operations

Your SQL query has been executed successfully.
1 row affected by the last statement inside the procedure.

```
SET @p0='5'; CALL `CheckDeptByEmpId`(@p0);
```

Execution results of routine `CheckDeptByEmpId`

| Message |
|---|
| Employee is in bata |

## Routines

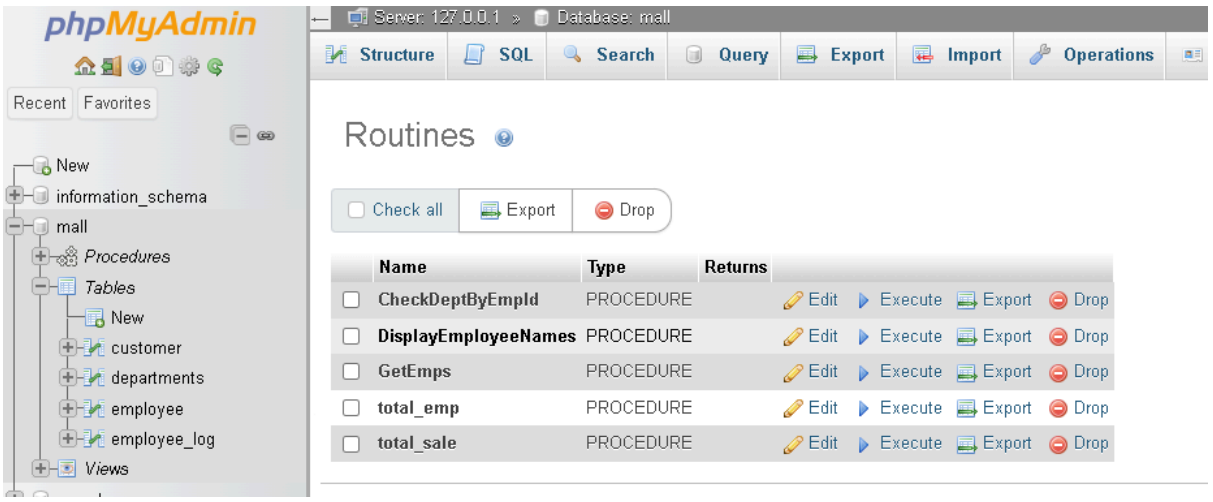
Check all | Export | Drop

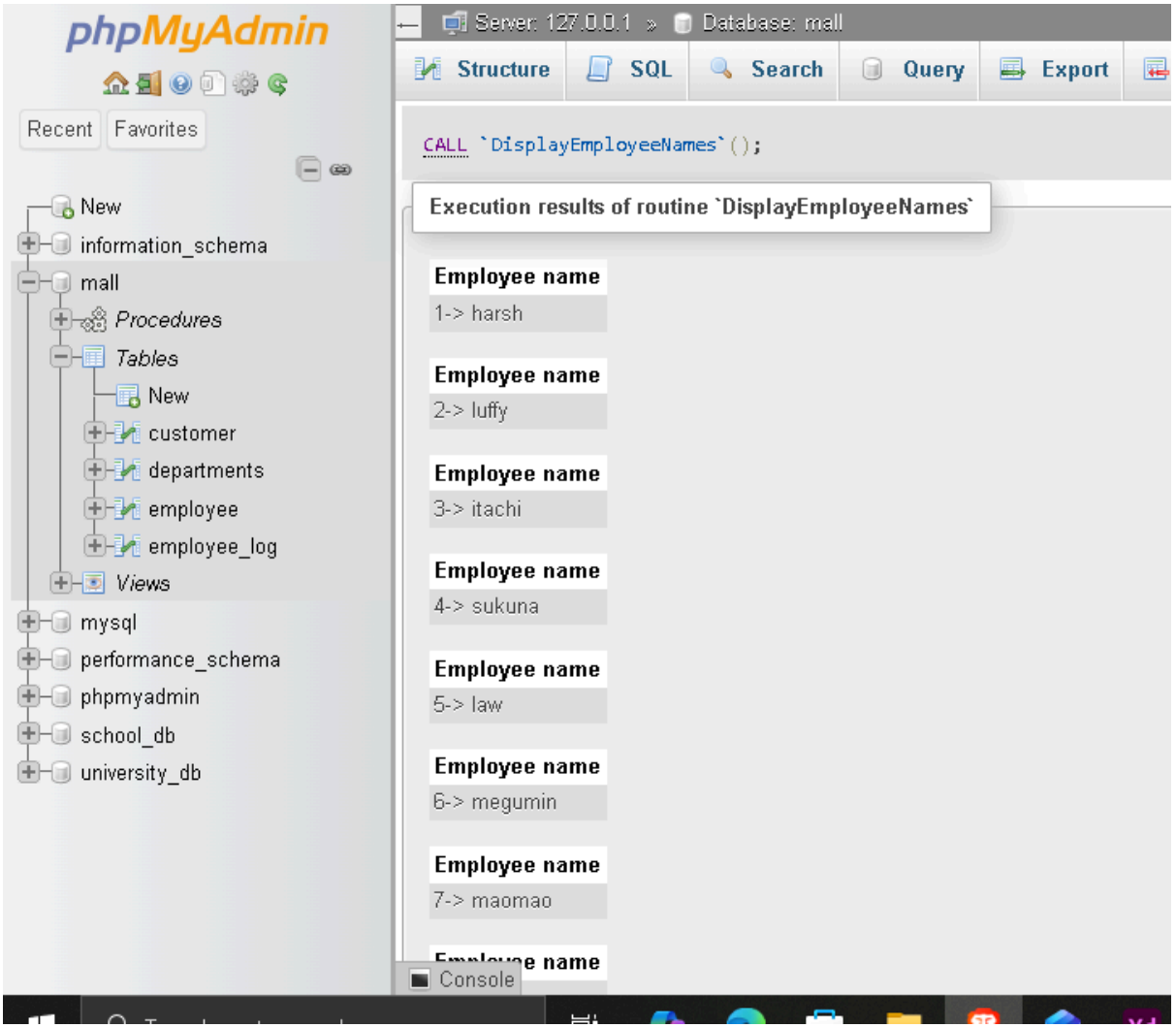
| Name | Type | Returns | |
|---|---|---|---|
| CheckDeptByEmpId | PROCEDURE | | Edit Execute Export Drop |
| GetEmps | PROCEDURE | | Edit Execute Export Drop |
| total_emp | PROCEDURE | | Edit Execute Export Drop |
| total_sale | PROCEDURE | | Edit Execute Export Drop |

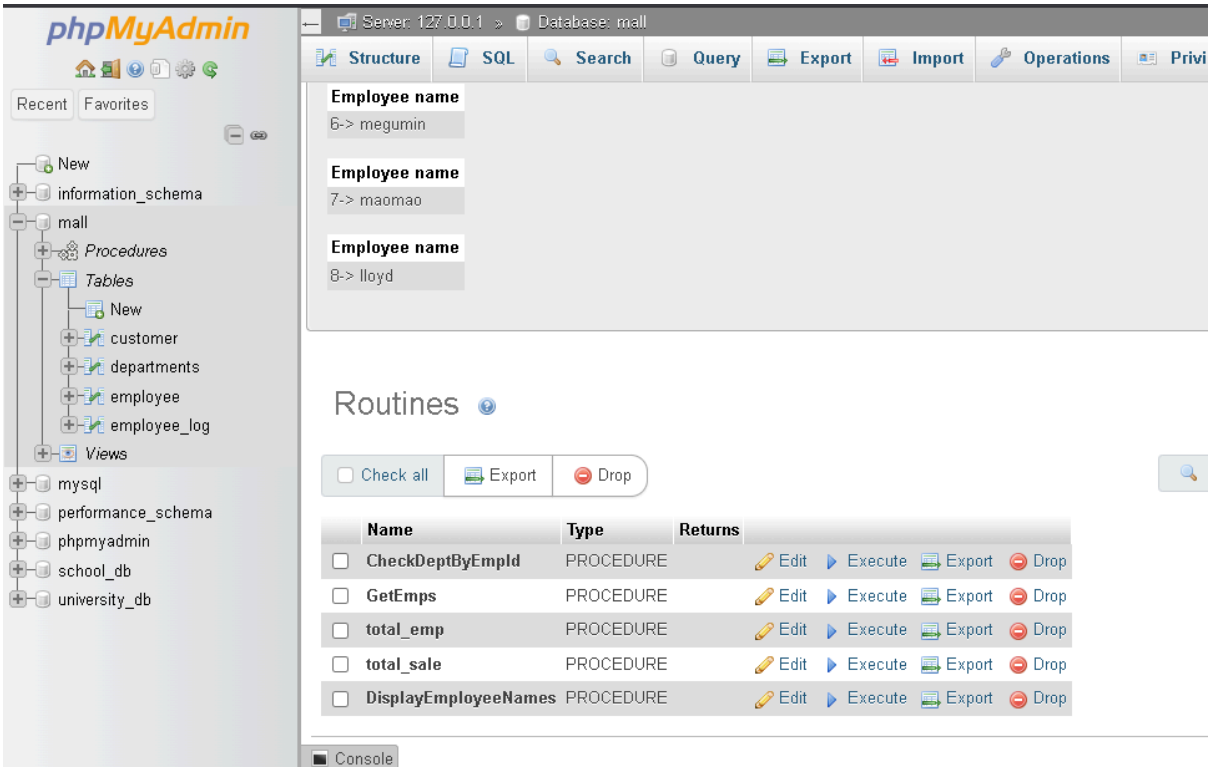
Console

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

```
DELIMITER $$

CREATE PROCEDURE DisplayEmployeeNames()
BEGIN
   DECLARE total INT DEFAULT 0;
   DECLARE counter INT DEFAULT 1;
   DECLARE empName VARCHAR(100);

   SELECT COUNT(*) INTO total FROM employee;

   WHILE counter <= total DO
      SELECT emp_name INTO empName
      FROM employee
      WHERE emp_id = counter;

      SELECT CONCAT(counter, '-> ', empName) AS
'Employee name';

      SET counter = counter + 1;
   END WHILE;
END $$

DELIMITER ;
```


phpMyAdmin
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
Server: 127.0.0.1 » Database: mall
Structure
SQL
Search
Query
Export
Import
Operations
Routines
Check all
Export
Drop
Name
Type
Returns
CheckDeptByEmpId PROCEDURE Edit Execute Export Drop
DisplayEmployeeNames PROCEDURE Edit Execute Export Drop
GetEmps PROCEDURE Edit Execute Export Drop
total_emp PROCEDURE Edit Execute Export Drop
total_sale PROCEDURE Edit Execute Export Drop


phpMyAdmin
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
Server: 127.0.0.1 » Database: mall
Structure
SQL
Search
Query
Export
CALL `DisplayEmployeeNames`();
Execution results of routine `DisplayEmployeeNames`
Employee name
1-> harsh
Employee name
2-> luffy
Employee name
3-> itachi
Employee name
4-> sukuna
Employee name
5-> law
Employee name
6-> megumin
Employee name
7-> maomao
Console

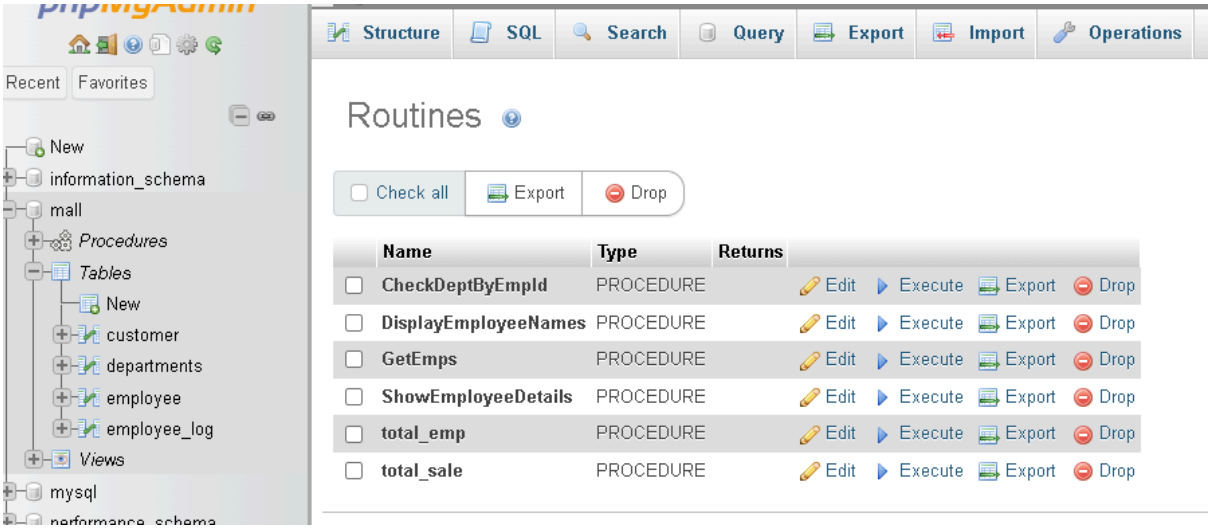

phpMyAdmin
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
Server: 127.0.0.1 » Database: mall
Structure
SQL
Search
Query
Export
Import
Operations
Employee name
6-> megumin
Employee name
7-> maomao
Employee name
8-> lloyd
Routines
Check all
Export
Drop
Name
Type
Returns
CheckDeptByEmpId PROCEDURE Edit Execute Export Drop
GetEmps PROCEDURE Edit Execute Export Drop
total_emp PROCEDURE Edit Execute Export Drop
total_sale PROCEDURE Edit Execute Export Drop
DisplayEmployeeNames PROCEDURE Edit Execute Export Drop
Console

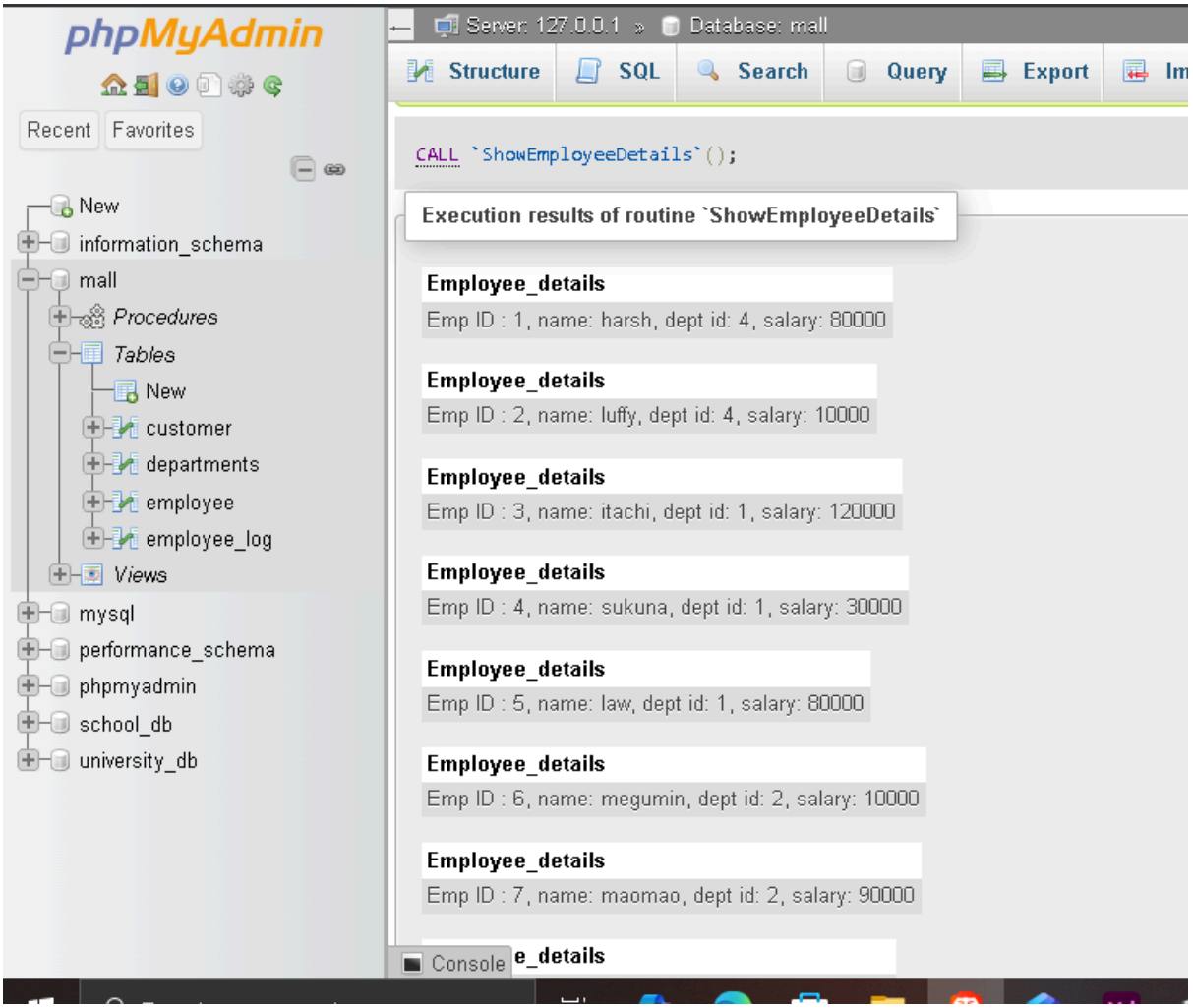
# SQL Cursors

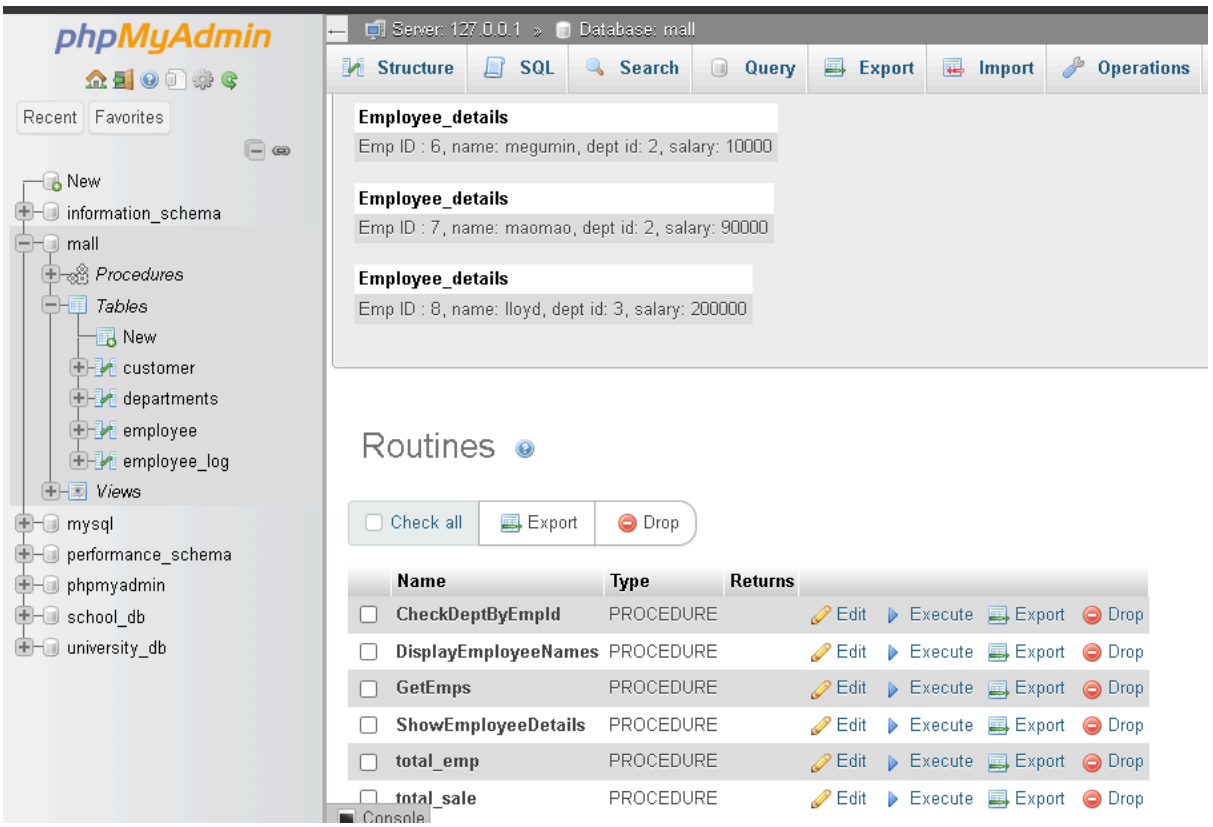
• Lab 1: Write a PL/SQL block using an explicit cursor to retrieve and display employee details.

```
DELIMITER $$

CREATE PROCEDURE ShowEmployeeDetails()
BEGIN
   DECLARE h_id INT;
   DECLARE h_name VARCHAR(100);
   DECLARE h_dept INT;
   DECLARE h_salary INT;
   DECLARE done INT DEFAULT 0;

   DECLARE emp_cursor CURSOR FOR
      SELECT emp_id, emp_name, dept_id, salary FROM
employee;

   DECLARE CONTINUE HANDLER FOR NOT FOUND
SET done = 1;

   OPEN emp_cursor;

   read_loop: LOOP
      FETCH emp_cursor INTO h_id, h_name, h_dept,
h_salary;
      IF done = 1 THEN
```

```
            LEAVE read_loop;
        END IF;

        SELECT CONCAT('Emp ID : ', h_id, ', name: ', h_name,
', dept id: ', h_dept, ', salary: ', h_salary) AS
Employee_details;
    END LOOP;

    CLOSE emp_cursor;
END $$

DELIMITER ;
```



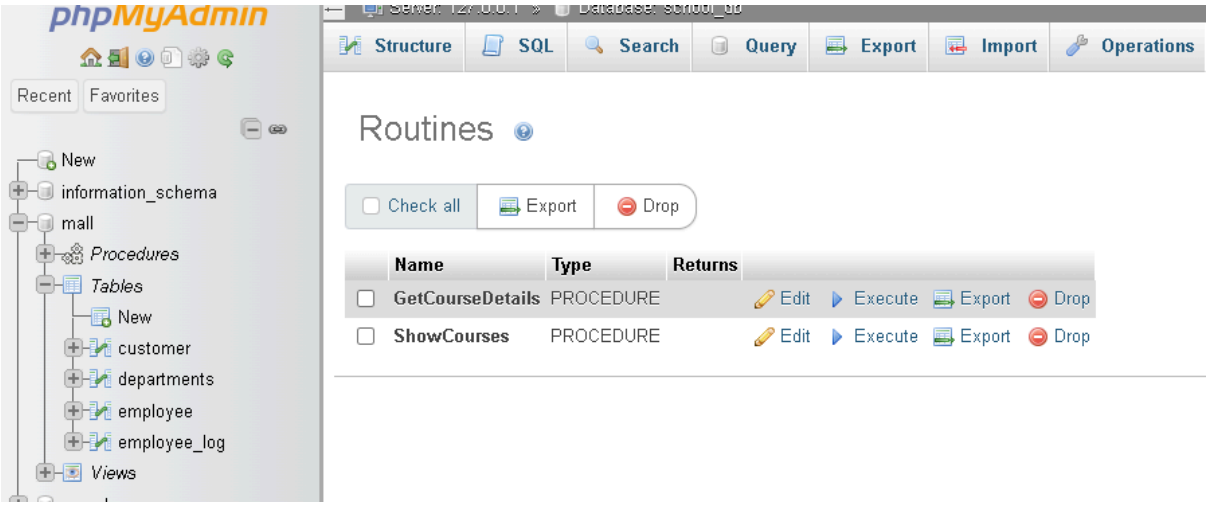

phpMyAdmin
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
Server: 127.0.0.1 » Database: mall
Structure
SQL
Search
Query
Export
CALL `ShowEmployeeDetails`();
Execution results of routine `ShowEmployeeDetails`
Employee_details
Emp ID : 1, name: harsh, dept id: 4, salary: 80000
Employee_details
Emp ID : 2, name: luffy, dept id: 4, salary: 10000
Employee_details
Emp ID : 3, name: itachi, dept id: 1, salary: 120000
Employee_details
Emp ID : 4, name: sukuna, dept id: 1, salary: 30000
Employee_details
Emp ID : 5, name: law, dept id: 1, salary: 80000
Employee_details
Emp ID : 6, name: megumin, dept id: 2, salary: 10000
Employee_details
Emp ID : 7, name: maomao, dept id: 2, salary: 90000
Console

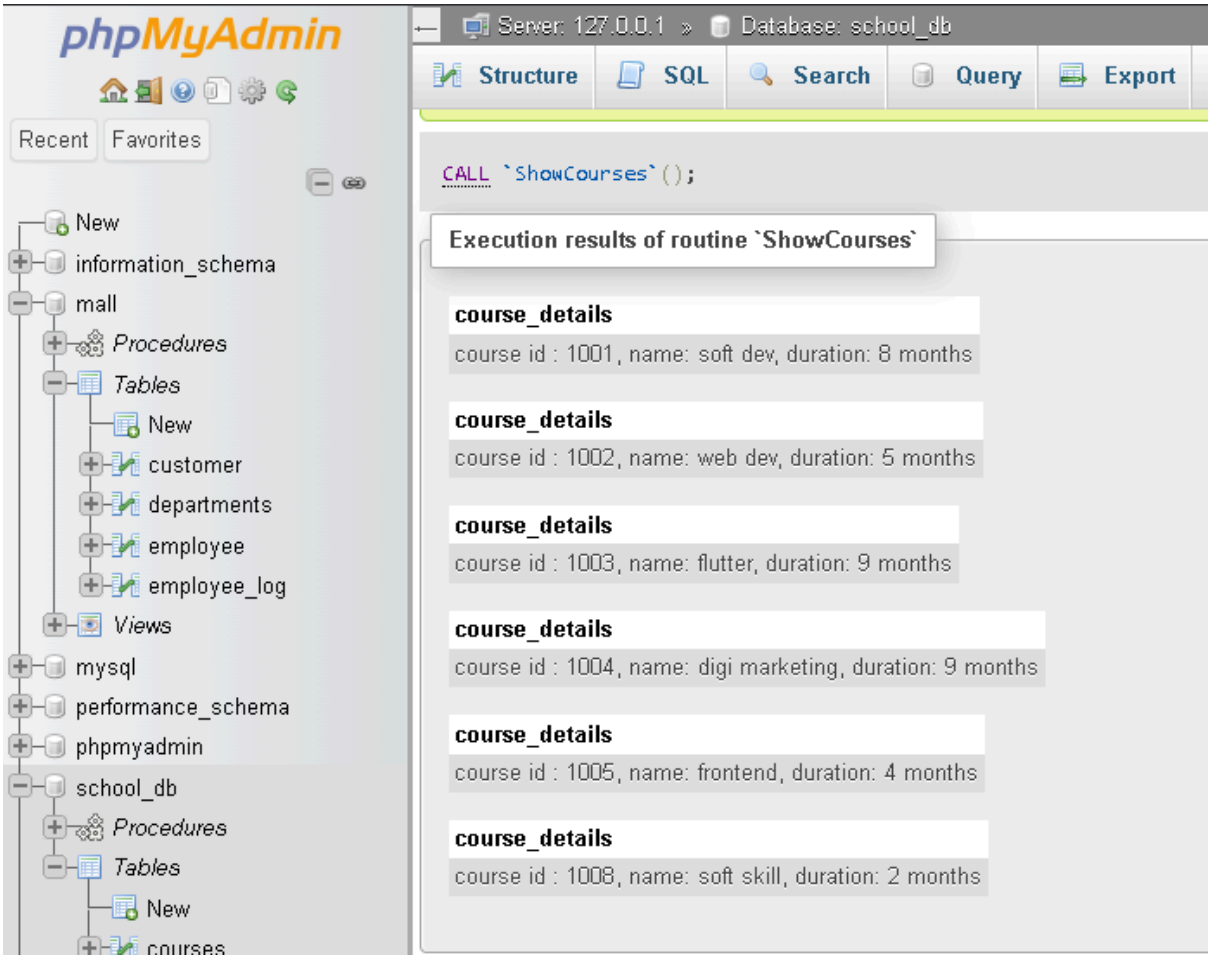

phpMyAdmin
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
university_db
Server: 127.0.0.1 » Database: mall
Structure
SQL
Search
Query
Export
Import
Operations
Employee_details
Emp ID : 6, name: megumin, dept id: 2, salary: 10000
Employee_details
Emp ID : 7, name: maomao, dept id: 2, salary: 90000
Employee_details
Emp ID : 8, name: lloyd, dept id: 3, salary: 200000
Routines
Check all
Export
Drop
Name
Type
Returns
CheckDeptByEmpId PROCEDURE Edit Execute Export Drop
DisplayEmployeeNames PROCEDURE Edit Execute Export Drop
GetEmps PROCEDURE Edit Execute Export Drop
ShowEmployeeDetails PROCEDURE Edit Execute Export Drop
total_emp PROCEDURE Edit Execute Export Drop
total_sale PROCEDURE Edit Execute Export Drop
Console

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

```
DELIMITER $$

CREATE PROCEDURE ShowCourses()
BEGIN
   DECLARE c_id INT;
   DECLARE c_name VARCHAR(100);
   DECLARE c_duration VARCHAR(50);
   DECLARE done INT DEFAULT 0;

   DECLARE course_cursor CURSOR FOR
      SELECT course_id, course_name, course_duration
FROM courses;

   DECLARE CONTINUE HANDLER FOR NOT FOUND
SET done = 1;

   OPEN course_cursor;

   read_loop: LOOP
      FETCH course_cursor INTO c_id, c_name, c_duration;
      IF done = 1 THEN
         LEAVE read_loop;
      END IF;
      SELECT CONCAT('course id : ', c_id, ', name: ',
c_name, ', duration: ', c_duration) AS course_details;
   END LOOP;
```

```
    CLOSE course_cursor;
END $$

DELIMITER ;
```




phpMyAdmin
Recent
Favorites
New
information_schema
mall
Procedures
Tables
New
customer
departments
employee
employee_log
Views
mysql
performance_schema
phpmyadmin
school_db
Procedures
Tables
New
courses
Server: 127.0.0.1 » Database: school_db
Structure
SQL
Search
Query
Export
CALL `ShowCourses`();
Execution results of routine `ShowCourses`
course_details
course id : 1001, name: soft dev, duration: 8 months
course_details
course id : 1002, name: web dev, duration: 5 months
course_details
course id : 1003, name: flutter, duration: 9 months
course_details
course id : 1004, name: digi marketing, duration: 9 months
course_details
course id : 1005, name: frontend, duration: 4 months
course_details
course id : 1008, name: soft skill, duration: 2 months

# Rollback and Commit Savepoint

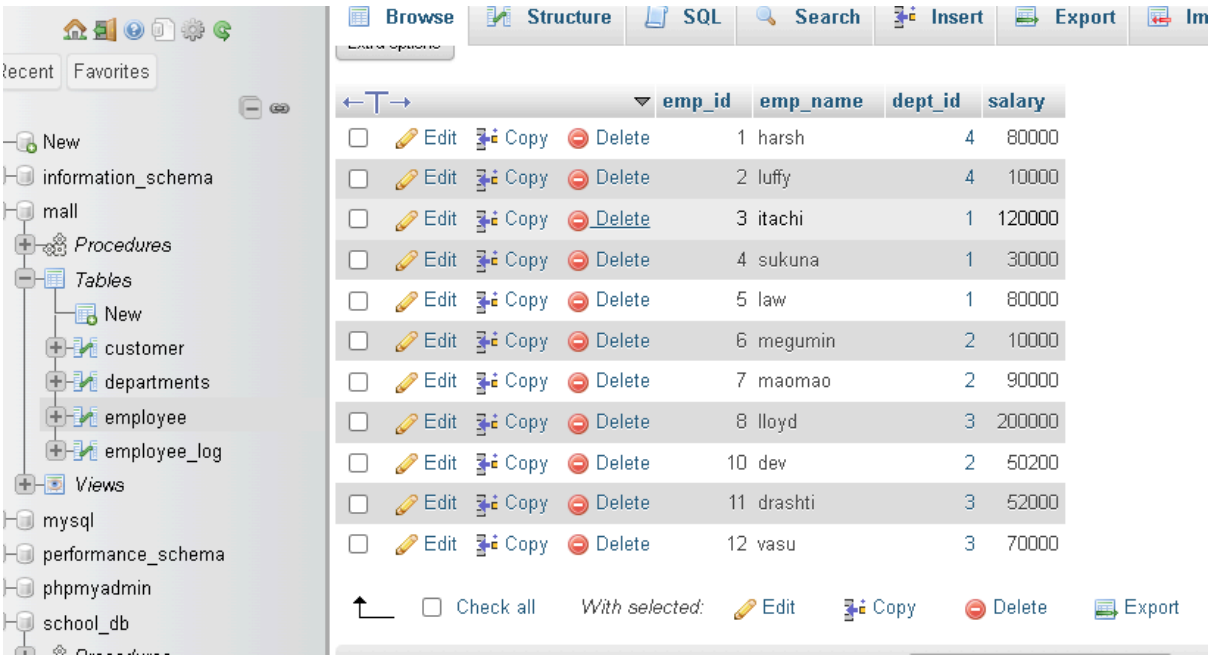
• Lab 1: Perform a transaction where you create a savepoint, insert records, then rollback to the savepoint.

START TRANSACTION;

INSERT INTO employee (emp_name, dept_id, salary)
VALUES ('vasu', 3, 70000);
SAVEPOINT after_first_insert;

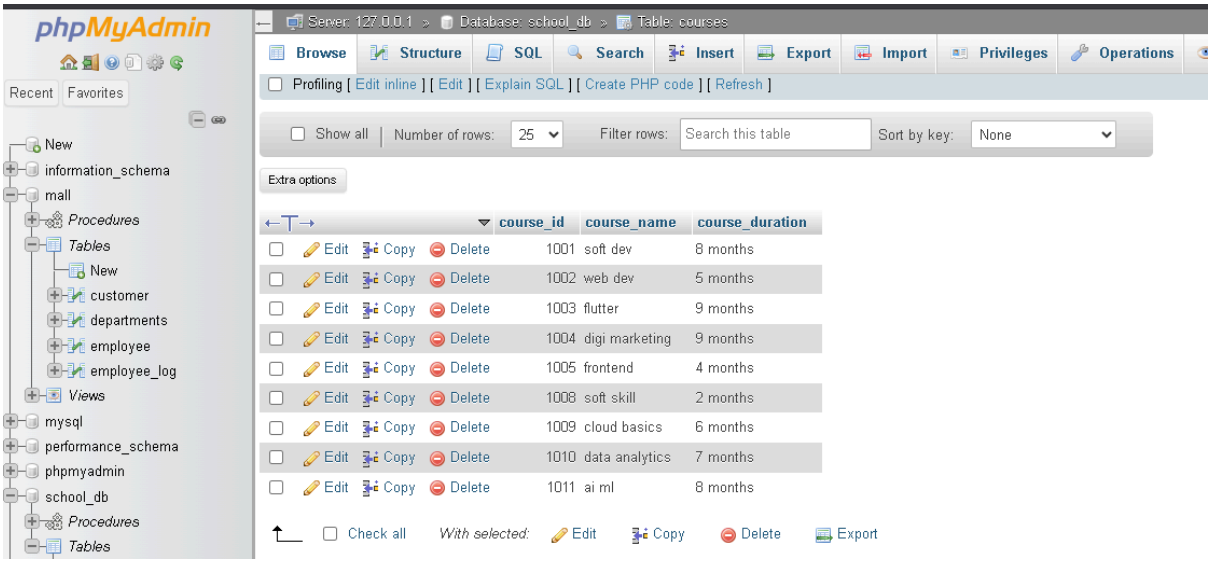
INSERT INTO employee (emp_name, dept_id, salary)
VALUES ('khwab', 1, 52000);
ROLLBACK TO after_first_insert;

COMMIT;



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

```
START TRANSACTION;

INSERT INTO courses (course_id, course_name,
course_duration)
VALUES (1009, 'cloud basics', '6 months');

SAVEPOINT after_first_course;

INSERT INTO courses (course_id, course_name,
course_duration)
VALUES (1010, 'data analytics', '7 months');

RELEASE SAVEPOINT after_first_course;
COMMIT;

START TRANSACTION;

INSERT INTO courses (course_id, course_name,
course_duration)
VALUES (1011, 'ai ml', '8 months');

SAVEPOINT after_new_course;

INSERT INTO courses (course_id, course_name,
course_duration)
VALUES (1012, 'seo', '5 months');
```

ROLLBACK TO after_new_course;

COMMIT;