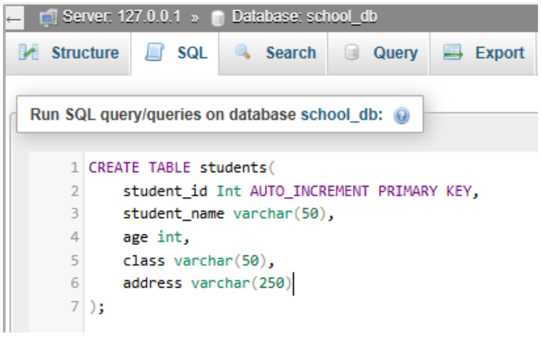
Module 4 – Introduction to DBMS

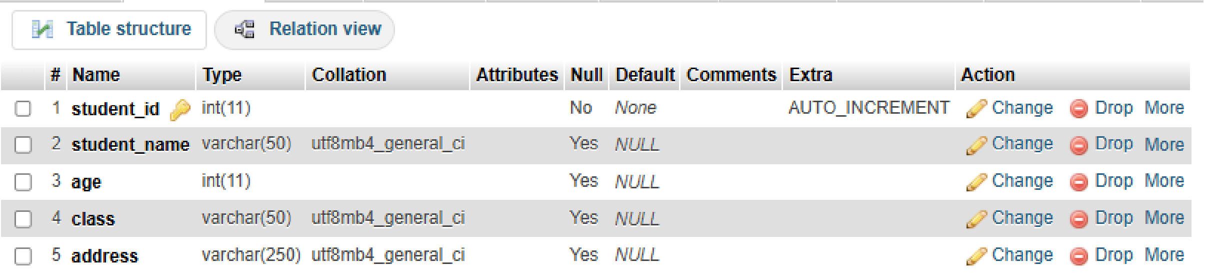
1-Introduction to SQL

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.



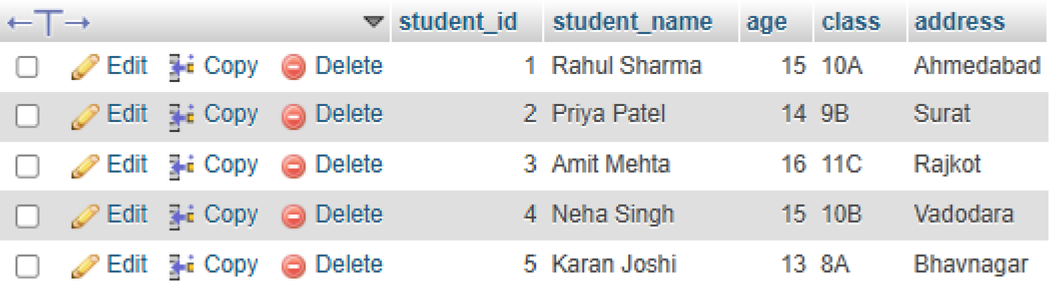
Run SQL query/queries on database school_db:

```
CREATE TABLE students(
    student_id Int AUTO_INCREMENT PRIMARY KEY,
    student_name varchar(50),
    age int,
    class varchar(50),
    address varchar(250)
);
```



	#	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(50)	utf8mb4_general_ci		Yes	NULL			Change Drop More
☐	3	age	int(11)			Yes	NULL			Change Drop More
☐	4	class	varchar(50)	utf8mb4_general_ci		Yes	NULL			Change Drop More
☐	5	address	varchar(250)	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.

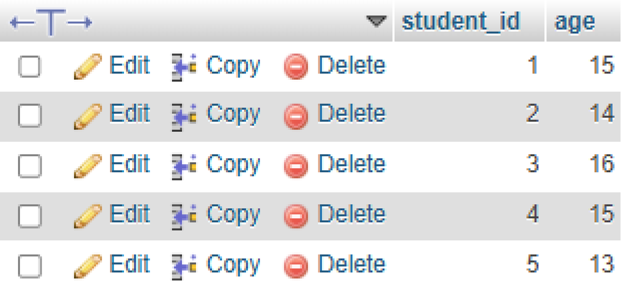


←T→				student_id	student_name	age	class	address
☐	Edit	Copy	Delete	1	Rahul Sharma	15	10A	Ahmedabad
☐	Edit	Copy	Delete	2	Priya Patel	14	9B	Surat
☐	Edit	Copy	Delete	3	Amit Mehta	16	11C	Rajkot
☐	Edit	Copy	Delete	4	Neha Singh	15	10B	Vadodara
☐	Edit	Copy	Delete	5	Karan Joshi	13	8A	Bhavnagar

2. SQL Syntax

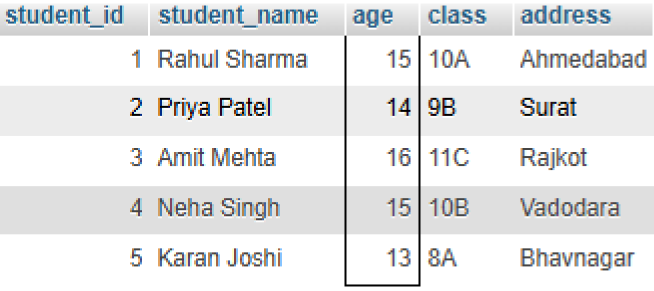
LAB EXERCISES:

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



←T→				student_id	age
☐	Edit	Copy	Delete	1	15
☐	Edit	Copy	Delete	2	14
☐	Edit	Copy	Delete	3	16
☐	Edit	Copy	Delete	4	15
☐	Edit	Copy	Delete	5	13

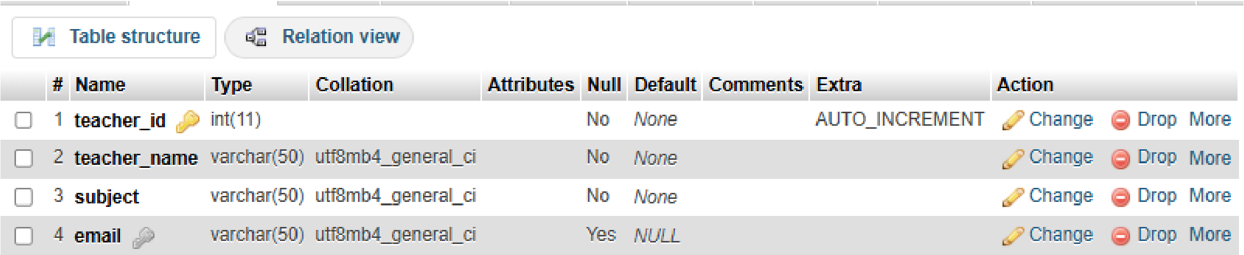
Lab 2: Write SQL queries to retrieve all students whose age is greater than 10.



student_id	student_name	age	class	address
1	Rahul Sharma	15	10A	Ahmedabad
2	Priya Patel	14	9B	Surat
3	Amit Mehta	16	11C	Rajkot
4	Neha Singh	15	10B	Vadodara
5	Karan Joshi	13	8A	Bhavnagar

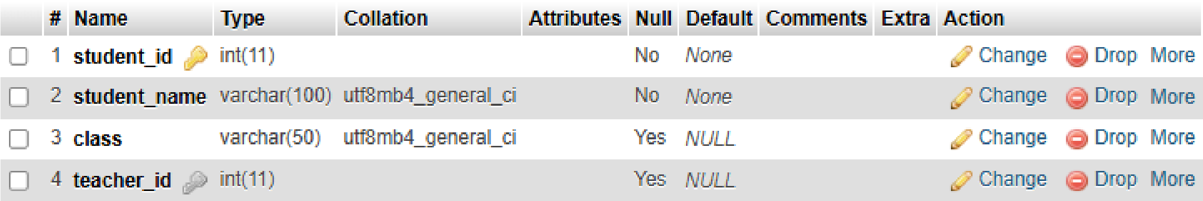
3. SQL Constraints

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



	#	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(50)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	subject	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	4	email	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More

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

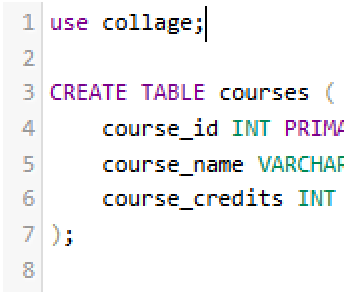


	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	student_id	int(11)			No	*None*			Change Drop More
☐	2	student_name	varchar(100)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	class	varchar(50)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More
☐	4	teacher_id	int(11)			Yes	*NULL*			Change Drop More

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

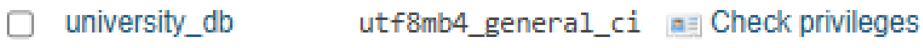
LAB EXERCISES:

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



```
use collage;

CREATE TABLE courses (
    course_id INT PRIMARY KEY,
    course_name VARCHAR(100),
    course_credits INT
);

```

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



□ university_db utf8mb4_general_ci Check privileges

5. ALTER Command

LAB EXERCISES:

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

Ans-

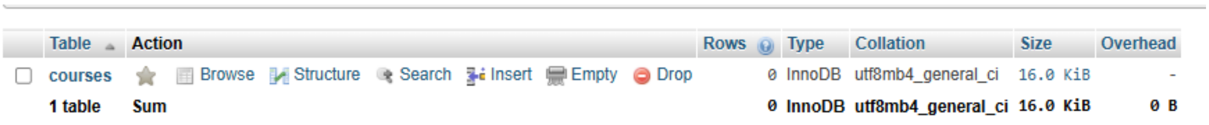
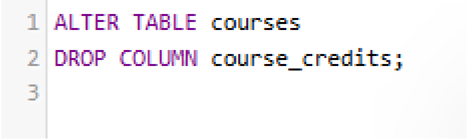


	Table	Action	Rows	Type	Collation	Size	Overhead
□	courses	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
	1 table	Sum	0	InnoDB	utf8mb4_general_ci	16.0 KiB	0 B

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

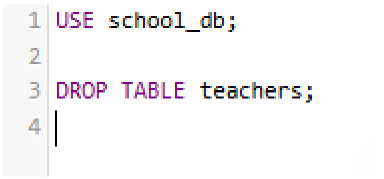


```
ALTER TABLE courses
DROP COLUMN course_credits;

```

6. DROP Command

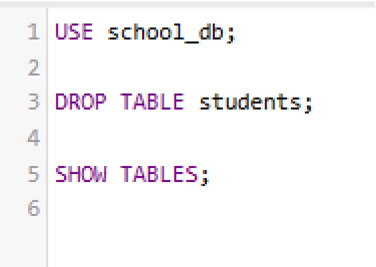
LAB EXERCISES:

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



```
USE school_db;

DROP TABLE teachers;

```

Lab 2: Drop the students table from the school_db database and verify that the table has been removed.

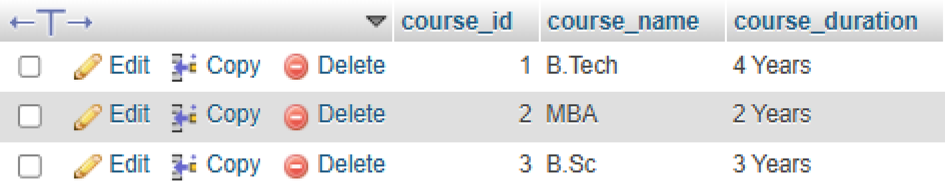


```
USE school_db;

DROP TABLE students;

SHOW TABLES;

```

7. Data Manipulation Language (DML)

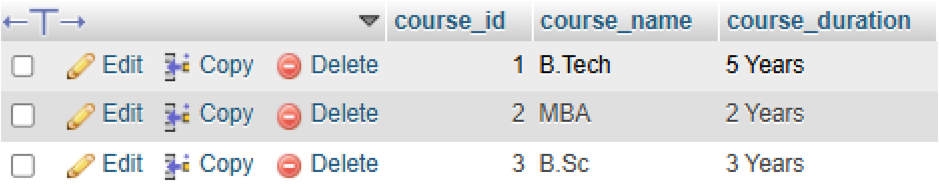
LAB EXERCISES:

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



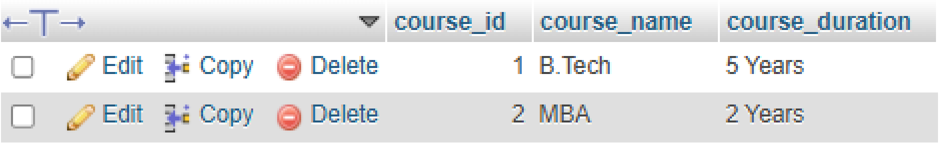
←T→	course_id	course_name	course_duration
□ Edit Copy Delete	1	B.Tech	4 Years
□ Edit Copy Delete	2	MBA	2 Years
□ Edit Copy Delete	3	B.Sc	3 Years

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



←T→	course_id	course_name	course_duration
□ Edit Copy Delete	1	B.Tech	5 Years
□ Edit Copy Delete	2	MBA	2 Years
□ Edit Copy Delete	3	B.Sc	3 Years

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

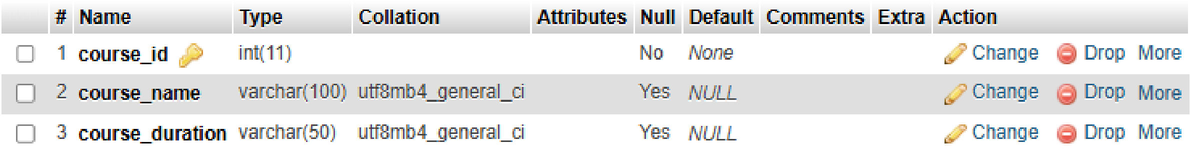


←T→	course_id	course_name	course_duration
□ Edit Copy Delete	1	B.Tech	5 Years
□ Edit Copy Delete	2	MBA	2 Years

8. Data Query Language (DQL)

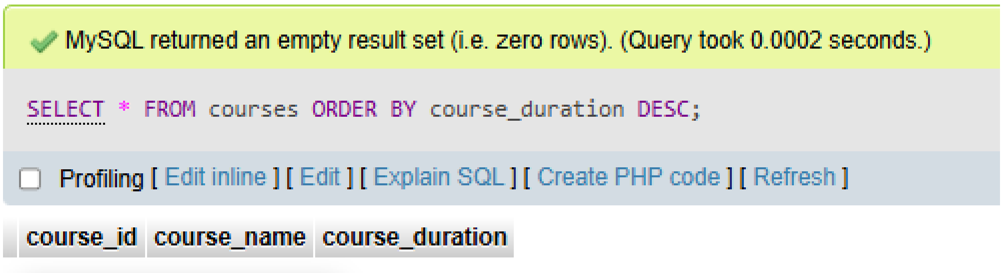
LAB EXERCISES:

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



	#	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(50)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More

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

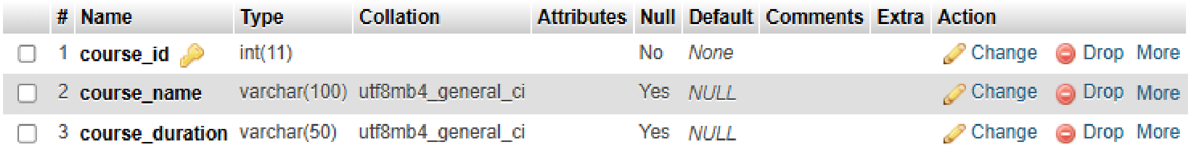


✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0002 seconds.)

```
SELECT * FROM courses ORDER BY course_duration DESC;
```

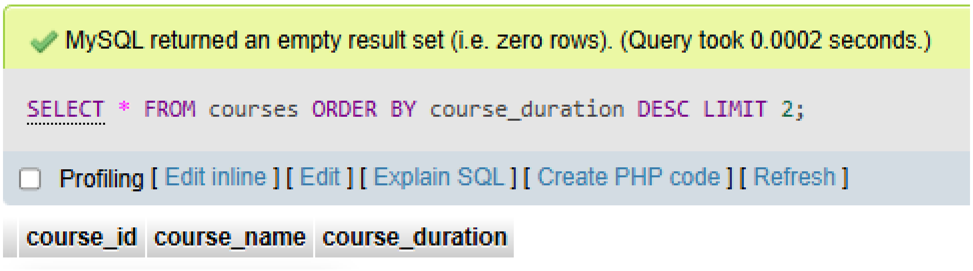
□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

course_id	course_name	course_duration



	#	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(50)	utf8mb4_general_ci		Yes	*NULL*			Change Drop More

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



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

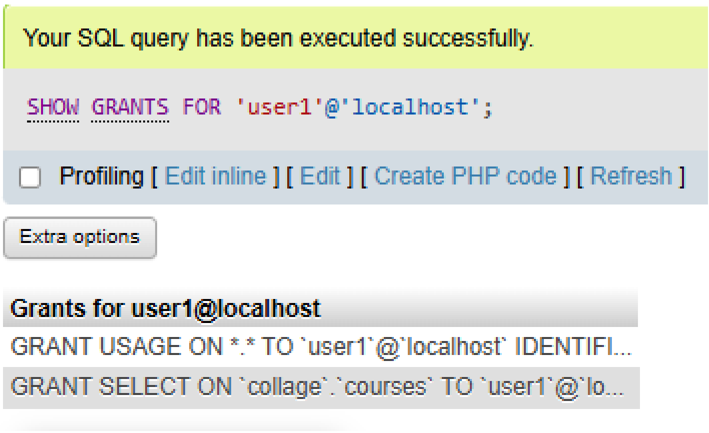
```
SELECT * FROM courses ORDER BY course_duration DESC LIMIT 2;
```

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

course_id	course_name	course_duration

9. Data Control Language (DCL)

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



Your SQL query has been executed successfully.

```
SHOW GRANTS FOR 'user1'@'localhost';
```

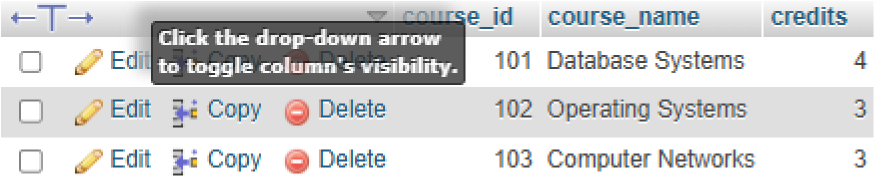
□ Profiling [Edit inline] [Edit] [Create PHP code] [Refresh]

Grants for user1@localhost
GRANT USAGE ON *.* TO \`user1\`@\`localhost\` IDENTIFI...
GRANT SELECT ON \`collage\`.\`courses\` TO \`user1\`@\`lo...

10. Transaction Control Language (TCL)

LAB EXERCISES:

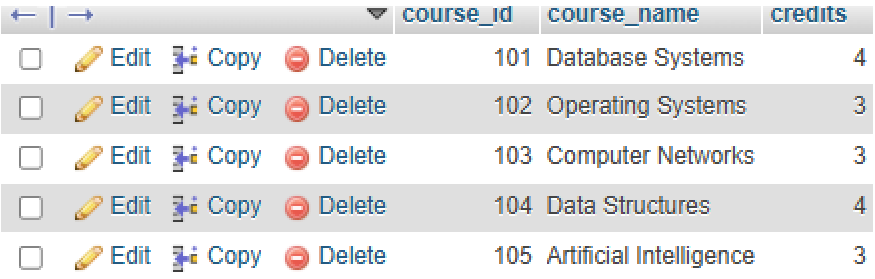
• Lab 1: Insert a few rows into the courses table and use COMMIT to save the changes.



Click the drop-down arrow to toggle column's visibility.

←T→				course_id	course_name	credits
□	Edit	Copy	Delete	101	Database Systems	4
□	Edit	Copy	Delete	102	Operating Systems	3
□	Edit	Copy	Delete	103	Computer Networks	3

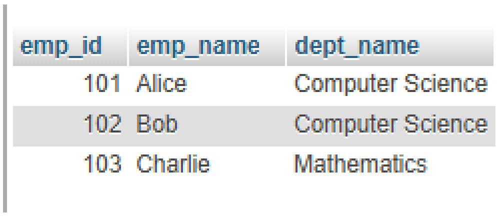
• Lab 2: Insert additional rows, then use ROLLBACK to undo the last insert operation.



← | →				course_id	course_name	credits
□	Edit	Copy	Delete	101	Database Systems	4
□	Edit	Copy	Delete	102	Operating Systems	3
□	Edit	Copy	Delete	103	Computer Networks	3
□	Edit	Copy	Delete	104	Data Structures	4
□	Edit	Copy	Delete	105	Artificial Intelligence	3

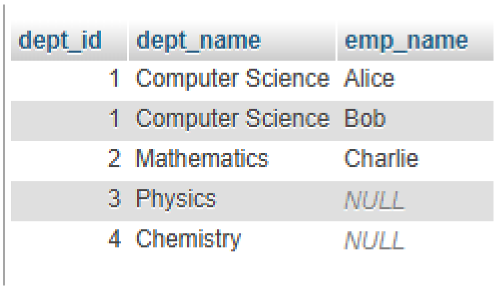
11. SQL Joins

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



emp_id	emp_name	dept_name
101	Alice	Computer Science
102	Bob	Computer Science
103	Charlie	Mathematics

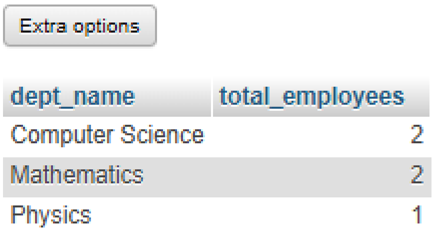
• Lab 2: Use a LEFT JOIN to show all departments, even those without employees.



dept_id	dept_name	emp_name
1	Computer Science	Alice
1	Computer Science	Bob
2	Mathematics	Charlie
3	Physics	*NULL*
4	Chemistry	*NULL*

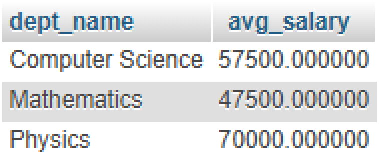
12. SQL Group By

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



dept_name	total_employees
Computer Science	2
Mathematics	2
Physics	1

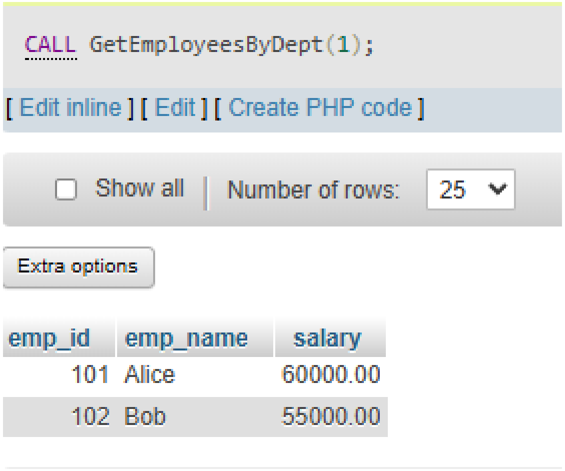
• Lab 2: Use the AVG aggregate function to find the average salary of employees in each department.



dept_name	avg_salary
Computer Science	57500.000000
Mathematics	47500.000000
Physics	70000.000000

13. SQL Stored Procedure

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

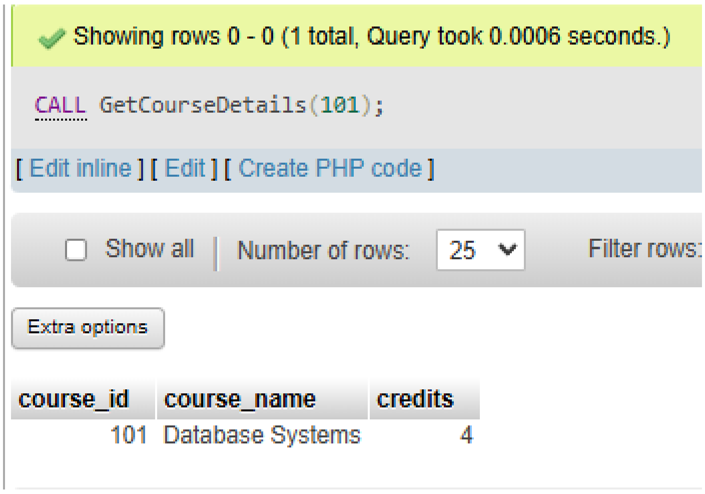


```
CALL GetEmployeesByDept(1);
```

□ Show all | Number of rows: 25

emp_id	emp_name	salary
101	Alice	60000.00
102	Bob	55000.00

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

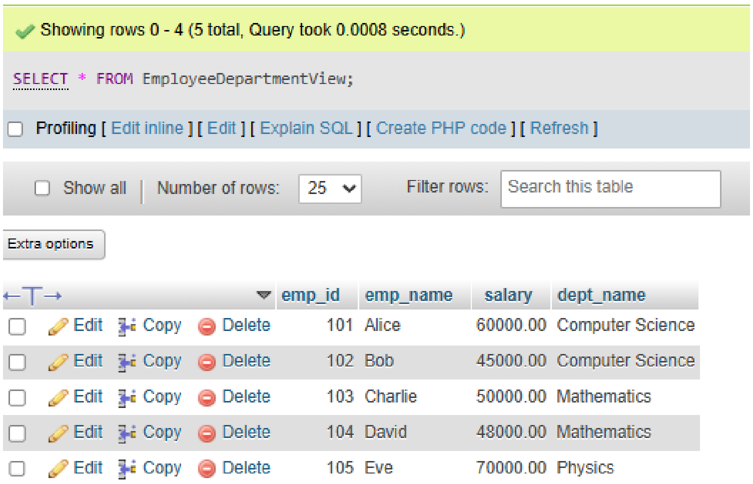

✓ Showing rows 0 - 0 (1 total, Query took 0.0006 seconds.)

```
CALL GetCourseDetails(101);
```

course_id	course_name	credits
101	Database Systems	4

14. SQL View

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

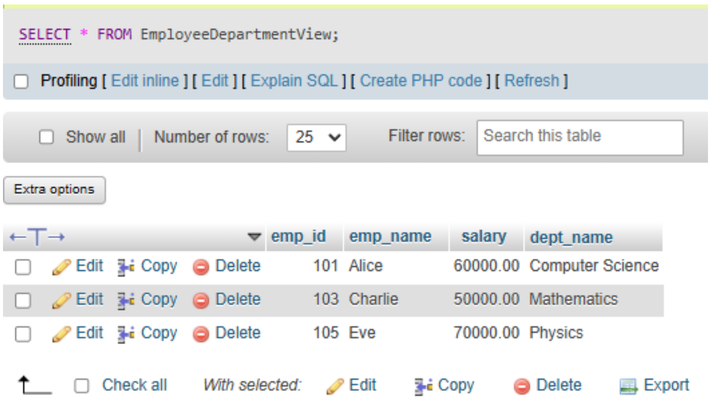

✓ Showing rows 0 - 4 (5 total, Query took 0.0008 seconds.)

```
SELECT * FROM EmployeeDepartmentView;
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all
Number of rows: 25
Filter rows: Search this table
Extra options
←T→

	emp_id	emp_name	salary	dept_name
□ Edit Copy Delete	101	Alice	60000.00	Computer Science
□ Edit Copy Delete	102	Bob	45000.00	Computer Science
□ Edit Copy Delete	103	Charlie	50000.00	Mathematics
□ Edit Copy Delete	104	David	48000.00	Mathematics
□ Edit Copy Delete	105	Eve	70000.00	Physics

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


```
SELECT * FROM EmployeeDepartmentView;
```

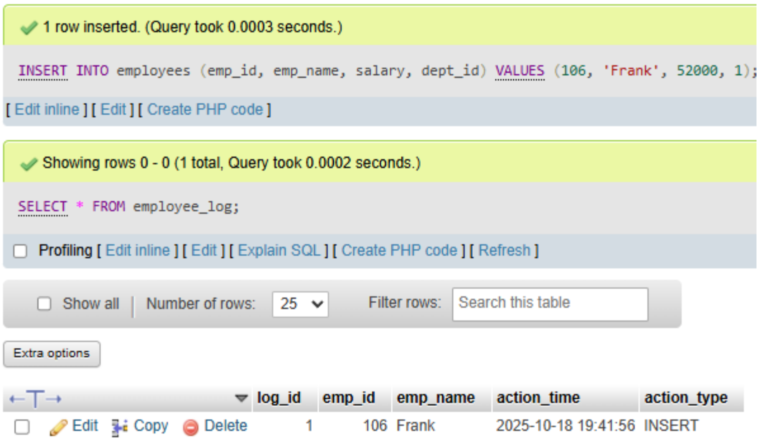
□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all
Number of rows: 25
Filter rows: Search this table
Extra options
←T→

	emp_id	emp_name	salary	dept_name
□ Edit Copy Delete	101	Alice	60000.00	Computer Science
□ Edit Copy Delete	103	Charlie	50000.00	Mathematics
□ Edit Copy Delete	105	Eve	70000.00	Physics

□ Check all
With selected:
Edit
Copy
Delete
Export

15. SQL Triggers

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



✓ 1 row inserted. (Query took 0.0003 seconds.)

```
INSERT INTO employees (emp_id, emp_name, salary, dept_id) VALUES (106, 'Frank', 52000, 1);
```

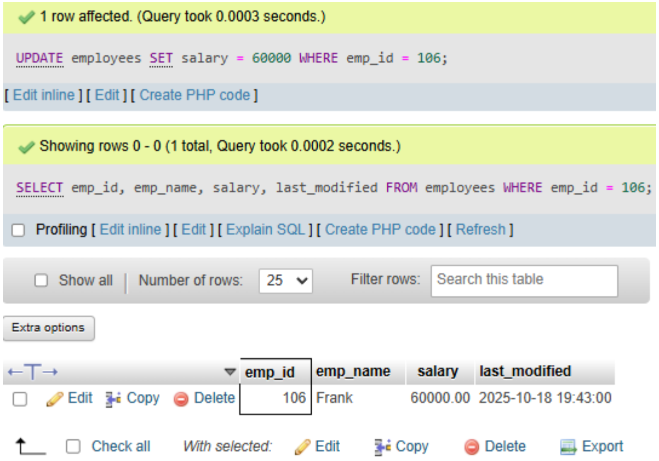
✓ Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)

```
SELECT * FROM employee_log;
```

log_id	emp_id	emp_name	action_time	action_type
1	106	Frank	2025-10-18 19:41:56	INSERT

[Edit inline] [Edit] [Create PHP code]
□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all | Number of rows: 25 Filter rows: Search this table
Extra options
□ Edit Copy Delete

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



✓ 1 row affected. (Query took 0.0003 seconds.)

```
UPDATE employees SET salary = 60000 WHERE emp_id = 106;
```

✓ Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)

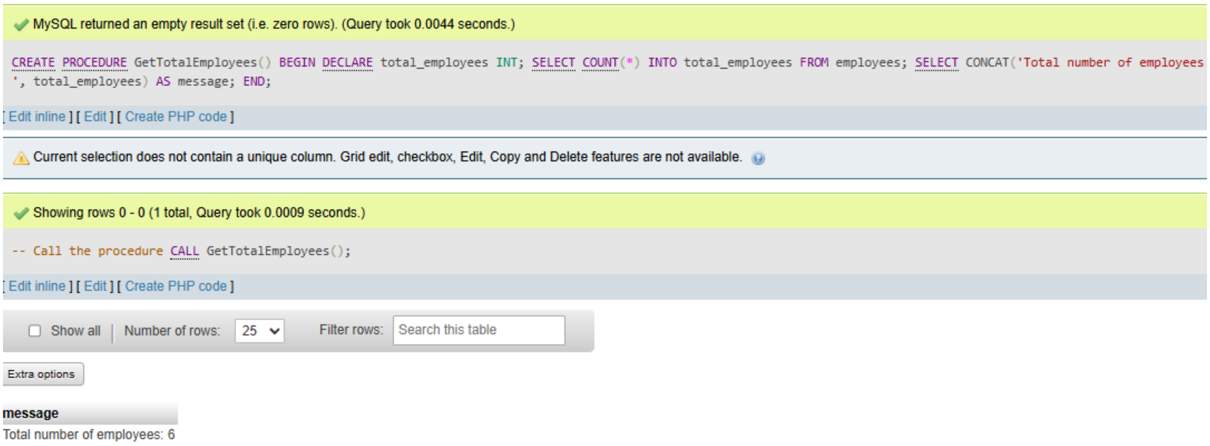
```
SELECT emp_id, emp_name, salary, last_modified FROM employees WHERE emp_id = 106;
```

emp_id	emp_name	salary	last_modified
106	Frank	60000.00	2025-10-18 19:43:00

[Edit inline] [Edit] [Create PHP code]
□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all | Number of rows: 25 Filter rows: Search this table
Extra options
□ Edit Copy Delete
□ Check all With selected: Edit Copy Delete Export

16. Introduction to PL/SQL

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



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0044 seconds.)

```
CREATE PROCEDURE GetTotalEmployees() BEGIN DECLARE total_employees INT; SELECT COUNT(*) INTO total_employees FROM employees; SELECT CONCAT('Total number of employees
', total_employees) AS message; END;
```

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

⚠ Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

✓ Showing rows 0 - 0 (1 total, Query took 0.0009 seconds.)

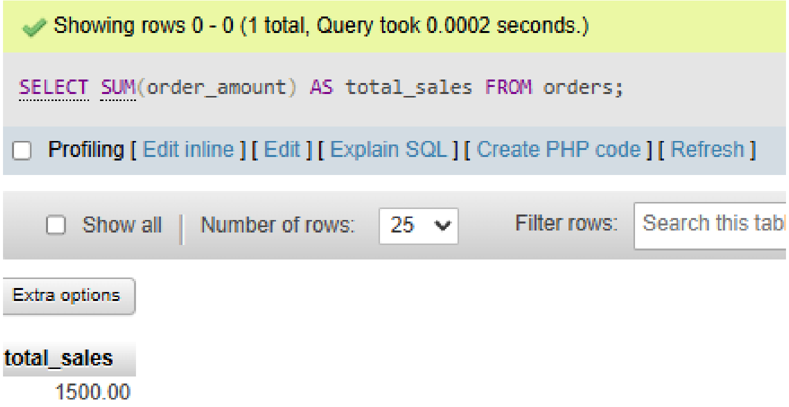
```
-- Call the procedure CALL GetTotalEmployees();
```

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

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

```
message
Total number of employees: 6
```

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



✓ Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)

```
SELECT SUM(order_amount) AS total_sales FROM orders;
```

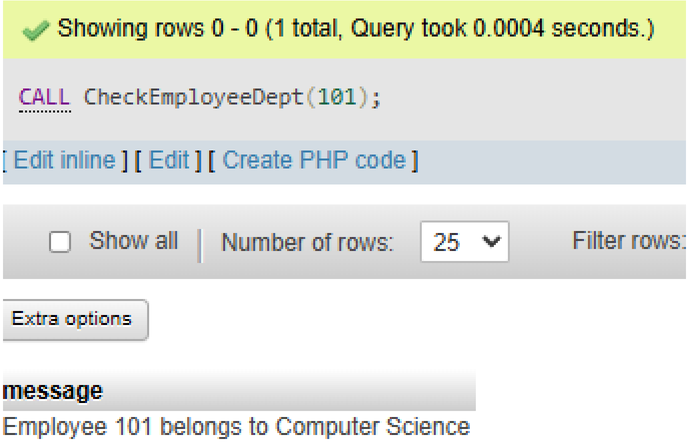
□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

□ Show all | Number of rows: 25
Filter rows: Search this tabl
Extra options

```
total_sales
     1500.00
```

17. PL/SQL Control Structures

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



✓ Showing rows 0 - 0 (1 total, Query took 0.0004 seconds.)

```
CALL CheckEmployeeDept(101);
```

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

```
message
Employee 101 belongs to Computer Science
```

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

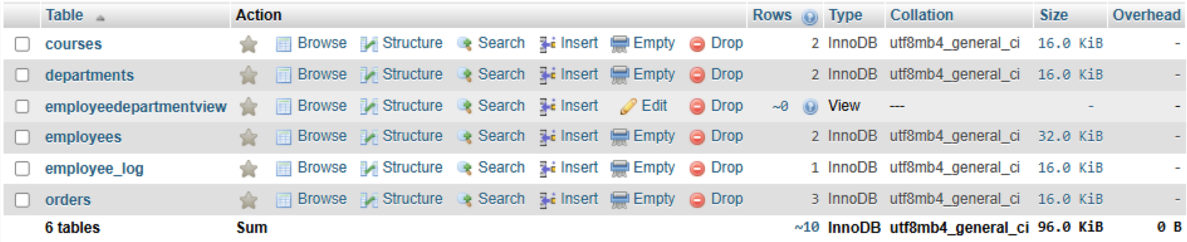


	Table	Action	Rows	Type	Collation	Size	Overhead
□	courses	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	departments	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	employeedepartmentview	Browse Structure Search Insert Edit Drop	~0	View	---	-	-
□	employees	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	32.0 KiB	-
□	employee_log	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	orders	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
	6 tables	Sum	~10	InnoDB	utf8mb4_general_ci	96.0 KiB	0 B

18. SQL Cursors

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

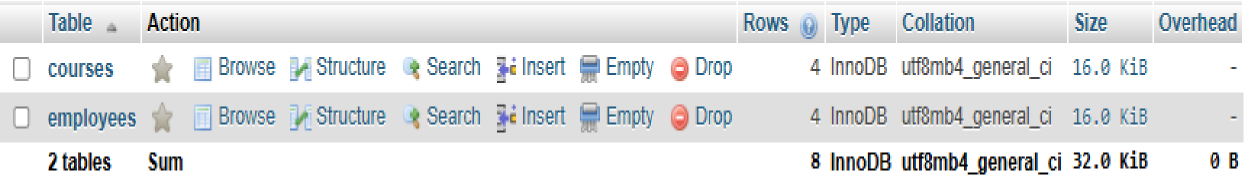
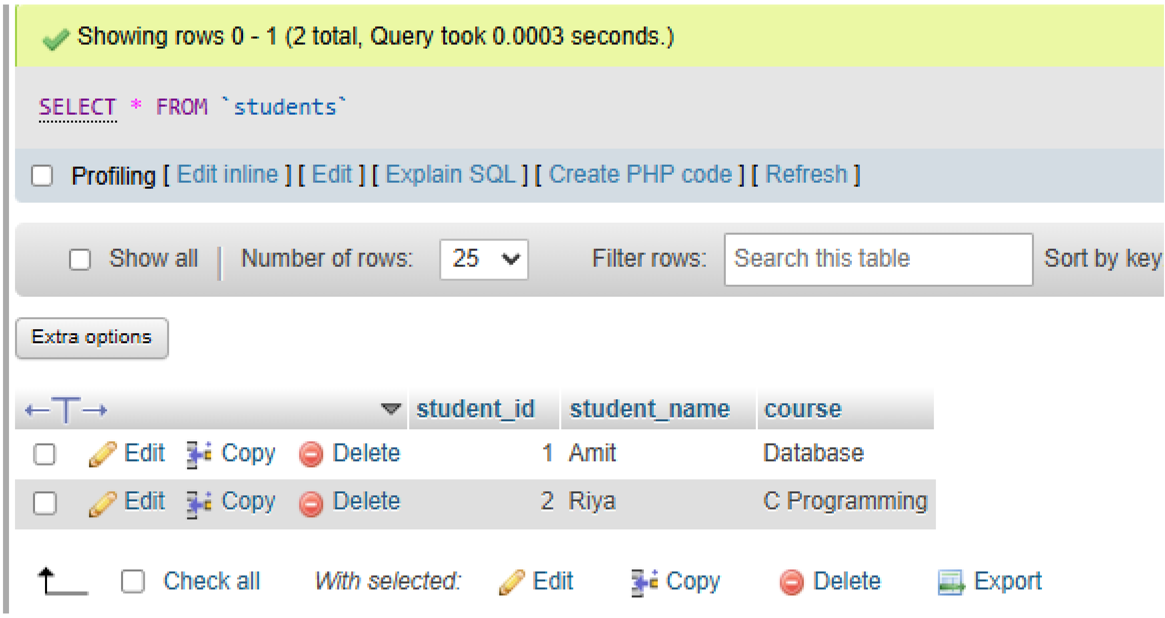


	Table	Action	Rows	Type	Collation	Size	Overhead
□	courses	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
□	employees	Browse Structure Search Insert Empty Drop	4	InnoDB	utf8mb4_general_ci	16.0 KiB	-
	2 tables	Sum	8	InnoDB	utf8mb4_general_ci	32.0 KiB	0 B

19. Rollback and Commit Savepoint

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



✓ Showing rows 0 - 1 (2 total, Query took 0.0003 seconds.)

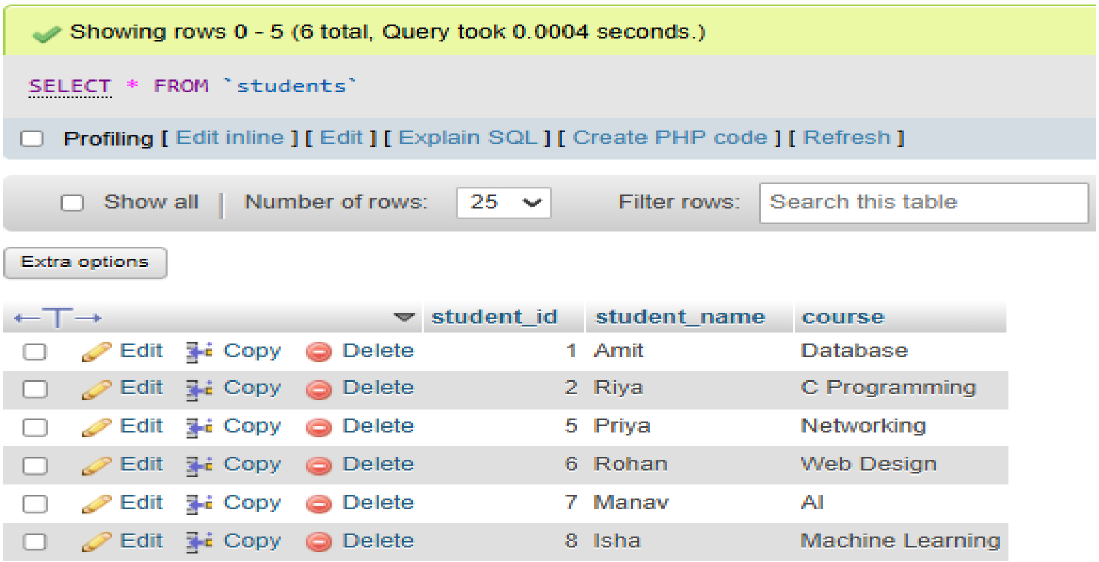
```
SELECT * FROM `students`
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
□ Show all | Number of rows: 25 Filter rows: Search this table Sort by key
Extra options

	student_id	student_name	course
□ Edit Copy Delete	1	Amit	Database
□ Edit Copy Delete	2	Riya	C Programming

□ Check all With selected: Edit Copy Delete Export

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



✓ Showing rows 0 - 5 (6 total, Query took 0.0004 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
Extra options

				student_id	student_name	course
□	Edit	Copy	Delete	1	Amit	Database
□	Edit	Copy	Delete	2	Riya	C Programming
□	Edit	Copy	Delete	5	Priya	Networking
□	Edit	Copy	Delete	6	Rohan	Web Design
□	Edit	Copy	Delete	7	Manav	AI
□	Edit	Copy	Delete	8	Isha	Machine Learning