

Lab: Stored Procedures in MySQL using phpMyAdmin



Estimated time needed: 20 minutes

In this lab, you will learn how to create tables and load data in the MySQL database service using the phpMyAdmin graphical user interface (GUI) tool.

Software Used in this Lab

In this lab, you will use [MySQL](#). MySQL is a Relational Database Management System (RDBMS) designed to efficiently store, manipulate, and retrieve data.



To complete this lab you will utilize MySQL relational database service available as part of IBM Skills Network Labs (SN Labs) Cloud IDE. SN Labs is a virtual lab environment used in this course.

Database Used in this Lab

`Mysql_learners` database has been used in this lab.

Data Used in this Lab

The data used in this lab is internal data. You will be working on the `PETSALE` table.

ID ▲	ANIMAL	SALEPRICE
1	Cat	450.09
2	Dog	666.66
3	Parrot	50.00
4	Hamster	60.60
5	Goldfish	48.48

This lab requires you to have the PETSALE table populated with sample data on mysql phpadmin interface. You might have created and populated a PETSALE table in a previous lab. But for this lab, it is recommended you download the `PETSALE-CREATE-v2.sql` script below, upload it to phpadmin console and run it. The script will create a new PETSALE table dropping any previous PETSALE table if exists, and will populate it with the required sample data.

- [PETSALE-CREATE-v2.sql](#)

Objectives

After completing this lab, you will be able to:

- Create stored procedures
- Execute stored procedures

Exercise 1

In this exercise, you will create and execute a stored procedure to read data from a table on mysql phpadmin using SQL.

1. Make sure you have created and populated the **PETSALE** table following the steps in the “**Data Used in this Lab**” section of this lab.

ID ▲	ANIMAL	SALEPRI
1	Cat	450.09
2	Dog	666.66
3	Parrot	50.00
4	Hamster	60.60
5	Goldfish	48.48

2.
 - o You will create a stored procedure routine named **RETRIEVE_ALL**.
 - o This **RETRIEVE_ALL** routine will contain an SQL query to retrieve all the records from the PETSALE table, so you don't need to write the same query over and over again. You just call the stored procedure routine to execute the query everytime.
 - o To create the stored procedure routine, copy the code below and paste it to the textarea of the **SQL** page. Click **Go**.

```
DELIMITER //
CREATE PROCEDURE RETRIEVE_ALL()
BEGIN
  SELECT * FROM PETSALE;

END //
DELIMITER ;
```

Run SQL query/queries on database Mysql_learners:

```
1 DELIMITER //
2
3 CREATE PROCEDURE RETRIEVE_ALL()
4
5 BEGIN
6
7     SELECT * FROM PETSALE;
8
9
10 END //
11
12 DELIMITER ;
```

 Bind parameters 

[Delimiter :] Show this query here again Retain query box Rollback when finished Enable foreign key checks

 MySQL returned an empty result set (i.e. zero rows). (Query took 0.0064 seconds.)

CREATE PROCEDURE RETRIEVE_ALL() BEGIN SELECT * FROM PETSALE; END

3. To call the RETRIEVE_ALL routine, open another **SQL** tab by clicking **Open in new Tab**

The screenshot shows the phpMyAdmin interface. On the left, there's a tree view of databases and tables. The 'EMPLOYEES' table under the 'HR' database is selected. The main panel shows a SQL query: 'SELECT * FROM `EMPLOYEES`'. A context menu is open at the top right of the SQL area, with the option 'Open link in new tab' highlighted.

Delete the default line which appears so that you will get a blank window.

copy the code below and paste it to the textarea of the **SQL** page. Click **Go**.

```
CALL RETRIEVE_ALL;
```

11 CALL RETRIEVE_ALL;

Bind parameters

Delimiter : [] Show this query here again Retain query box Rollback when finished Enable foreign key checks

[Hide query box](#)

Showing rows 0 - 4 (5 total, Query took 0.0010 seconds.)

CALL RETRIEVE_ALL

Show all | Number of rows: 25 Filter rows:

Options

	ANIMAL	SALEPRICE	SALEDATE	QUANTITY
1	Cat	450.09	2018-05-29	9
2	Dog	666.66	2018-06-01	3
3	Parrot	50.00	2018-06-04	2
4	Hamster	60.60	2018-06-11	6
5	Goldfish	48.48	2018-06-14	24

4. You can view the created stored procedure routine RETRIEVE_ALL. On the left panel, expand the mysql option. Click on **Procedures** then click on the **RETRIEVE_ALL** and view the procedure.

phpMyAdmin

Server: mysql:3306 » Database: mysql » Table: PETSALE

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#) [Privileges](#) [Operations](#)

Routine 'RETRIEVE_ALL' has been modified.

```
DROP PROCEDURE `RETRIEVE_ALL`; CREATE DEFINER='root'@`%` PROCEDURE `RETRIEVE_ALL`() NOT DETERMINISTIC CONTAINS SQL SQL SECURITY DEFINER
BEGIN
    SELECT * FROM `PETSALE` WHERE 1;
END
```

Run SQL query/queries on table mysql.PETSALE:

ID	ANIMAL	SALEPRICE	SALEDATE	QUANTITY
1	Cat	450.09	2018-05-29	9
2	Dog	666.66	2018-06-01	3
3	Parrot	50.00	2018-06-04	2
4	Hamster	60.60	2018-06-11	6
5	Goldfish	48.48	2018-06-14	24

Bind parameters

Delimiter : [] Show this query here again Retain query box Rollback when finished Enable foreign key checks

After clicking on the Procedure **Retrieval_All**, you can view the procedure definition and execute it by clicking on **GO**.

The screenshot shows the phpMyAdmin interface for creating a stored procedure named 'RETRIEVE_ALL'. The 'Details' tab is selected. The 'Routine name' is set to 'RETRIEVE_ALL' and the 'Type' is 'PROCEDURE'. The 'Definition' section contains the following SQL code:

```
1 BEGIN
2
3   SELECT * FROM PETSALE;
4
5
6 END
```

Below the code, the 'Is deterministic' checkbox is unchecked. The 'Adjust privileges' checkbox is checked. The 'Definer' field is set to 'root'@'%'. The 'Security type' is 'DEFINER'. The 'Go' button at the bottom right is highlighted with a red box.

5. If you wish to drop the stored procedure routine RETRIEVE_ALL, copy the code below and paste it to the textarea of the SQL page. Click Go.

```
DROP PROCEDURE RETRIEVE_ALL;
CALL RETRIEVE_ALL;
```

The screenshot shows the MySQL Workbench interface. The top navigation bar includes tabs for Structure, SQL, Search, Query, Export, Import, Operations, Privileges, and Routines. The SQL tab is active. Below the tabs is a code editor window containing the following SQL script:

```

1 DROP PROCEDURE RETRIEVE_ALL;
2
3 CALL RETRIEVE_ALL;
4
5
6

```

Below the code editor are several buttons: Clear, Format, Get auto-saved query, Bind parameters (with a help icon), and checkboxes for Delimiter, Show this query here again, Retain query box, Rollback when finished, and Enable foreign key checks. The Delimiter field contains a semicolon. The "Enable foreign key checks" checkbox is checked.

A red box highlights the "Error" section below the code editor. It displays the following message:

Error

SQL query: [Copy](#)

CALL RETRIEVE_ALL

MySQL said: [?](#)

#1305 - PROCEDURE Mysql_learners.RETRIEVE_ALL does not exist

Exercise 2

In this exercise, you will create and execute a stored procedure to write/modify data in a table on Db2 using SQL.

1. Make sure you have created and populated the **PETSALE** table following the steps in the “Data Used in this Lab” section of this lab.

ID ▲	ANIMAL	SALEPRI
1	Cat	450.09
2	Dog	666.66
3	Parrot	50.00
4	Hamster	60.60
5	Goldfish	48.48

- 2.
- o You will create a stored procedure routine named **UPDATE_SALEPRICE** with parameters **Animal_ID** and **Animal_Health**.
 - o This **UPDATE_SALEPRICE** routine will contain SQL queries to update the sale price of the animals in the PETSALE table depending on their health conditions, **BAD** or **WORSE**.
 - o This procedure routine will take animal ID and health condition as parameters which will be used to update the sale price of animal in the PETSALE table by an amount depending on their health condition. Suppose -
 - For animal with ID XX having BAD health condition, the sale price will be reduced further by 25%.
 - For animal with ID YY having WORSE health condition, the sale price will be reduced further by 50%.
 - For animal with ID ZZ having other health condition, the sale price won't change.

- To create the stored procedure routine, copy the code below and paste it to the textarea of the **SQL** page. Click **Go**.

```
DELIMITER @
CREATE PROCEDURE UPDATE_SALEPRICE (
    IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5) )
BEGIN
    IF Animal_Health = 'BAD' THEN
        UPDATE PETSALE
        SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.25)
        WHERE ID = Animal_ID;

    ELSEIF Animal_Health = 'WORSE' THEN
        UPDATE PETSALE
        SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.5)
        WHERE ID = Animal_ID;

    ELSE
        UPDATE PETSALE
        SET SALEPRICE = SALEPRICE
        WHERE ID = Animal_ID;
    END IF;

END @
DELIMITER ;
```

The screenshot shows the MySQL Workbench interface with the following details:

- Top Bar:** Shows the connection name "SERVER: mysql:3306" and the database "Database: mysql_learners".
- Toolbar:** Includes tabs for Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, and Help.
- Query Editor:** A large text area containing the following MySQL code for a stored procedure:


```

15
16    ELSE
17        UPDATE PETSALE
18        SET SALEPRICE = SALEPRICE
19        WHERE ID = Animal_ID;
20
21    END IF;
22
23 END @
24
25 DELIMITER ;
26
      
```
- Buttons:** Below the editor are buttons for Clear, Format, Get auto-saved query, Bind parameters (with a help icon), and checkboxes for Show this query here again, Retain query box, Rollback when finished, and Enable foreign key checks.
- Feedback:** A green message bar at the bottom left indicates: "MySQL returned an empty result set (i.e. zero rows). (Query took 0.0214 seconds.)"
- Code Preview:** The bottom section shows the full CREATE PROCEDURE statement:


```

CREATE PROCEDURE UPDATE_SALEPRICE ( IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5) ) BEGIN IF Animal_Health = 'BAD' THEN
(SALEPRICE * 0.25) WHERE ID = Animal_ID; ELSEIF Animal_Health = 'WORSE' THEN UPDATE PETSALE SET SALEPRICE = SALEPRICE -
PETSAL SET SALEPRICE = SALEPRICE WHERE ID = Animal_ID; END IF; END
      
```

- Let's call the UPDATE_SALEPRICE routine. We want to update the sale price of animal with ID 1 having **BAD** health condition in the PETSALE table. open another **SQL** tab by clicking **Open in new Tab**

The screenshot shows the phpMyAdmin interface. On the left, a tree view lists databases: New, HR, information_schema, mysql, Mysql_learners, performance_schema, and sys. Under the HR database, tables DEPARTMENTS, EMPLOYEES, JOBS, JOB_HISTORY, and LOCATIONS are visible. The EMPLOYEES table is selected. The main panel shows the SQL query 'SELECT * FROM `EMPLOYEES`'. A context menu is open over this query, with the option 'Open link in new tab' highlighted.

Delete the default line which appears so that you will get a blank window.

copy the code below and paste it to the textarea of the **SQL** page. Click **Go**.

Note if you have dropped RETREIVE_ALL procedure rerun the creation script of that procedure before executing these lines.

```
CALL RETRIEVE_ALL;
CALL UPDATE_SALEPRICE(1, 'BAD');
CALL RETRIEVE_ALL;
```

Showing rows 0 - 4 (5 total, Query took 0.0007 seconds.)

CALL RETRIEVE_ALL

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

+ Options

ID	ANIMAL	SALEPRICE	SALEDATE	QUANTITY
1	Cat	450.09	2018-05-29	9
2	Dog	666.66	2018-06-01	3
3	Parrot	50.00	2018-06-04	2
4	Hamster	60.60	2018-06-11	6
5	Goldfish	48.48	2018-06-14	24

Note: #1265 D

Show all |

+ Options

ID	ANIMAL	SALEPRICE	SALEDATE	QUANTITY
1	Cat			
2	Dog			
3	Parrot			
4	Hamster			
5	Goldfish			

4. Let's call the UPDATE_SALEPRICE routine once again. We want to update the sale price of animal with ID 3 having **WORSE** health condition in the PETSALE table. copy the code below and paste it to the textarea of the SQL page. Click **Go**. You will have all the records retrieved from the PETSALE table.

```
CALL RETRIEVE_ALL;
CALL UPDATE_SALEPRICE(3, 'WORSE');
CALL RETRIEVE_ALL;
```

Showing rows 0 - 4 (5 total, Que

CALL RETRIEVE_ALL

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

Options

ID	ANIMAL	SALEPRICE	SALEDATE	QUANTITY
1	Cat	337.57	2018-05-29	9
2	Dog	666.66	2018-06-01	3
3	Parrot	50.00	2018-06-04	2
4	Hamster	60.60	2018-06-11	6
5	Goldfish	48.48	2018-06-14	24

CALL RETRIEVE_ALL

Show all | Number of rows:

Options

ID	ANIMAL	SALEPRICE	SALEDATE	QUANTITY
1	Cat	337.57	2018-05-29	9
2	Dog	666.66	2018-06-01	3
3	Parrot	25.00	2018-06-04	2
4	Hamster	60.60	2018-06-11	6
5	Goldfish	48.48	2018-06-14	24

Show all | Number of rows:

Query results operations

5. You can view the created stored procedure routine UPDATE_SALEPRICE. Click on the **Routines** and view the procedure.

The screenshot shows the MySQL Workbench interface with the 'Routines' tab selected. There are two stored procedures listed:

Name	Action	Type	Returns
<input type="checkbox"/> RETRIEVE_ALL	Edit Execute Export Drop	PROCEDURE	
<input type="checkbox"/> UPDATE_SALEPRICE	Edit Execute Export Drop	PROCEDURE	

Below the table, there are buttons for 'Check all' and 'With selected:' followed by 'Export' and 'Drop' buttons.

New

Add routine

6. If you wish to drop the stored procedure routine UPDATE_SALEPRICE, copy the code below and paste it to the textarea of the **SQL** page. Click **Go**.

```
DROP PROCEDURE UPDATE_SALEPRICE;
CALL UPDATE_SALEPRICE;
```

The SQL page contains the following code:

```
7
8
9 DROP PROCEDURE UPDATE_SALEPRICE;
10
11 CALL UPDATE_SALEPRICE;
```

Below the code are several buttons: Clear, Format, Get auto-saved query, Bind parameters, and a Delimiter input field set to semicolon. At the bottom are checkboxes for Show this query here again, Retain query box, Rollback when finished, and Enable foreign key checks.

[Hide query box](#)

Error

SQL query: [Copy](#)

```
DROP PROCEDURE UPDATE_SALEPRICE
```

MySQL said: [?](#)

```
#1305 - PROCEDURE Mysql_learners.UPDATE_SALEPRICE does not exist
```

Congratulations! You have completed this lab on creating stored procedures in MySQL, and are ready for the next topic.

Author(s)

[Lakshmi Holla](#)

[Malika Singla](#)

© IBM Corporation. All rights reserved.