

# Hands-on 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	SALEPRICE
1	Cat	450.09
2	Dog	666.66
3	Parrot	50.00
4	Hamster	60.60
5	Goldfish	48.48

- 2.
- You will create a stored procedure routine named **RETRIEVE\_ALL**.
  - 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.
  - To create the stored procedure routine, copy the code below and paste it to the textarea of the **SQL** page. Click **Go**.

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
9. 9
10. 10
11. 11
12. 12

1. DELIMITER //
2.
```

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

Copied!


Run SQL query/queries on database Mysql\_learners: 

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

Clear

Format

Get auto-saved query

☐ Bind parameters 

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

Hide query box

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

```
CREATE PROCEDURE RETRIEVE_ALL() BEGIN SELECT * FROM PETALE; END
```

[\[Edit inline\]](#) [\[Ec](#)

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

Server: mysql:3306 » Database: HR » Table: EMPLOYEES

Run SQL query/queries on table HR

1 `SELECT * FROM `EMPLOYEES``

SELECT \* SELECT INSERT UPDATE DELETE Clear Format Get auto-saved query

☐ Bind parameters

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

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**.

1. 1

1. CALL RETRIEVE\_ALL;

Copied!

11 CALL RETRIEVE\_ALL;

Clear

Format

Get auto-saved query

☐ 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

[\[Edit inline\]](#) [\[ Edit \]](#) [\[ Create \]](#)

☐ Show all

Number of rows: 

25

 ▼

Filter rows: 

Search this table

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.

The screenshot shows the phpMyAdmin interface with the following elements:

- Left Sidebar:** A tree view showing the database structure. The 'mysql' database is selected, and the 'Procedures' folder is expanded. The procedure 'RETRIEVE\_ALL' is highlighted with a red box.
- Top Navigation:** A horizontal bar with tabs for 'Browse', 'Structure', 'SQL', 'Search', 'Insert', 'Export', 'Import', 'Privileges', 'Operations', and 'Triggers'. The 'SQL' tab is active.
- Main Content Area:**
  - A green message bar at the top states: 'Routine 'RETRIEVE\_ALL' has been modified.'
  - Below the message is the SQL definition for the procedure:

```
DROP PROCEDURE `RETRIEVE_ALL`; CREATE DEFINER=`root`@`%` PROCEDURE `RETRIEVE_ALL`() NOT DETERMINISTIC CONTAINS SQL SQL SECURITY DEFINER BEGIN SELECT * FROM PETSALE; END
```
  - Below the definition are links: '[Edit inline]', '[Edit]', and '[Create PHP code]'.
  - A section titled 'Run SQL query/queries on table mysql.PETSALE:' contains a text area with the query: 

```
1 SELECT * FROM `PETSALE` WHERE 1
```
  - Below the query text area are buttons: 'SELECT \*', 'SELECT', 'INSERT', 'UPDATE', 'DELETE', 'Clear', 'Format', and 'Get auto-saved query'.
  - Below the buttons is a checkbox labeled 'Bind parameters'.
  - At the bottom of the main area are options: 'Delimiter: ;', 'Show this query here again', 'Retain query box', 'Rollback when finished', and 'Enable foreign key checks' (checked). A 'Go' button is on the right.

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

The screenshot shows the phpMyAdmin interface with the 'RETRIEVE\_ALL' stored procedure selected. The 'Details' tab is active, displaying the following information:

- Routine name:** RETRIEVE\_ALL
- Type:** PROCEDURE
- Parameters:** (Empty table with columns: Direction, Name, Type, Length/Values, Options)
- Definition:**

```
1 BEGIN
2
3   SELECT * FROM PETSALE;
4
5
6 END
```
- Is deterministic:** ☐
- Adjust privileges:** ☒
- Definer:** `root`@`%`
- Security type:** DEFINER

At the bottom right of the 'Details' window, the 'Go' button is highlighted with a red box. Other buttons visible include 'Close', 'Operations', 'Triggers', 'Columns', and 'Checks'.

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**.

```
1. 1
2. 2
3. 3

1. DROP PROCEDURE RETRIEVE_ALL;
2.
3. CALL RETRIEVE_ALL;
```

Copied!

The screenshot shows a MySQL IDE interface with a menu bar at the top containing: Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, Events, Triggers, and Designer. The main query editor area contains the following SQL code:

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

Below the query editor are buttons for "Clear", "Format", and "Get auto-saved query". There is also a checkbox for "Bind parameters" with a help icon. At the bottom of the editor area, there is a row of options: "[ Delimiter ; ]", "Show this query here again", "Retain query box", "Rollback when finished", and "Enable foreign key checks" (which is checked).

A red error message box is displayed at the bottom of the IDE. It has the title "Error" and contains the following text:

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

2.
- You will create a stored procedure routine named **UPDATE\_SALEPRICE** with parameters **Animal\_ID** and **Animal\_Health**.

◦ 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**.

◦ This procedure routine will take animal ID and health conditon 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**.
1. 1

2. 2

3. 3

4. 4

5. 5

6. 6

7. 7

8. 8

9. 9

10. 10

11. 11

12. 12

13. 13

14. 14

15. 15

16. 16

17. 17

18. 18

19. 19

20. 20

21. 21

22. 22

23. 23

24. 24

25. 25

26. 26
1. DELIMITER @

2. CREATE PROCEDURE UPDATE\_SALEPRICE (
- about:blank
- 9/16

```
3.    IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5) )
4. BEGIN
5.
6.    IF Animal_Health = 'BAD' THEN
7.        UPDATE PETSale
8.        SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.25)
9.        WHERE ID = Animal_ID;
10.
11.   ELSEIF Animal_Health = 'WORSE' THEN
12.       UPDATE PETSale
13.       SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.5)
14.       WHERE ID = Animal_ID;
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.
```

Copied!

Server: mysql:5.6.27 Database: mysql\_learners

Structure SQL Search Query Export Import Operations Privileges Routines Events Triggers Designer

Run SQL query/queries on database Mysql\_learners: ?

```

15
16 ELSE
17     UPDATE PETALE
18     SET SALEPRICE = SALEPRICE
19     WHERE ID = Animal_ID;
20
21 END IF;
22
23 END @
24
25 DELIMITER ;
26

```

Clear Format Get auto-saved query

☐ Bind parameters ?

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

Hide query box

✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0214 seconds.)

```

CREATE PROCEDURE UPDATE_SALEPRICE ( IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5) ) BEGIN IF Animal_Health = 'BAD' THEN UPDATE PETALE SET SALEPRICE = SALEP
(SALEPRICE * 0.25) WHERE ID = Animal_ID; ELSEIF Animal_Health = 'WORSE' THEN UPDATE PETALE SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.5) WHERE ID = Animal_ID;
PETALE SET SALEPRICE = SALEPRICE WHERE ID = Animal_ID; END IF; END

```

[Edit inline] [Edit] [

- 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 PETALE table. open another **SQL** tab by clicking **Open in new Tab**

The screenshot shows the phpMyAdmin web interface. The browser address bar displays the URL: `lakshmih-8080.theiadocker-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/tbl_sql.php?db=HR&table=EMPLOYEES`. The phpMyAdmin interface includes a left sidebar with a database tree showing 'HR' and its tables: 'DEPARTMENTS', 'EMPLOYEES', 'JOBS', 'JOB\_HISTORY', and 'LOCATIONS'. The main panel is titled 'Run SQL query/queries on table HR' and shows the SQL tab with a query editor containing the text: `1 SELECT * FROM `EMPLOYEES``. A context menu is open over the query editor, with the option 'Open link in new tab' highlighted. Below the query editor are buttons for 'SELECT \*', 'SELECT', 'INSERT', 'UPDATE', 'DELETE', 'Clear', 'Format', and 'Get auto-saved query'. At the bottom, there are checkboxes for 'Bind parameters', 'Show this query here again', 'Retain query box', 'Rollback when finished', and 'Enable foreign key checks' (which is checked).

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.

```

1. 1
2. 2
3. 3
4. 4
5. 5

1. CALL RETRIEVE_ALL;
2.
3. CALL UPDATE_SALEPRICE(1, 'BAD');
4.
5. CALL RETRIEVE_ALL;

```

Copied!

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

CALL RETRIEVE\_ALL

☐ Show all | Number of rows: 25 ▾ | Filter rows:

+ 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 Data truncated for column 'SALEPRICE' at row 1

✓ Showing rows 0 - 4 (5 total, Query took 0.0015 seconds.)

CALL RETRIEVE\_ALL

☐ Show all | Number of rows: 25 ▾ | Filter 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	50.00	2018-06-04	2
4	Hamster	60.60	2018-06-11	6
5	Goldfish	48.48	2018-06-14	24

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.

```

1. 1
2. 2
3. 3
4. 4
5. 5

1. CALL RETRIEVE_ALL;
2.
3. CALL UPDATE_SALEPRICE(3, 'WORSE');
4.
5. CALL RETRIEVE_ALL;

```

Copied!

CALL RETRIEVE\_ALL

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

CALL RETRIEVE\_ALL

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

Options

	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

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

Query results operations

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

Options

D	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: 25 | Filter rows: Search this table

5. You can view the created stored procedure routine UPDATE\_SALEPRICE. Click on the **Routines** and view the procedure.

Structure
SQL
Search
Query
Export
Import
Operations
Privileges
Routines
Events
Triggers
Designer

**Routines**

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

☐ Check all With selected: Export Drop

**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**.

1. 1
2. 2

3. 3

1. DROP PROCEDURE UPDATE\_SALEPRICE;
- 2.
3. CALL UPDATE\_SALEPRICE;

Copied!

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

Clear

Format

Get auto-saved query

☐ Bind parameters ⓘ

[ Delimiter  ] ☐ Show this query here again ☐ Retain query box ☐ Rollback when finished ☒ 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](#)

## Changelog

Date	Version	Changed by	Change Description
2021-08-09	0.2	Sathya Priya	Updated HTML tags and SQL link
2021-11-01	0.1	Lakshmi Holla, Malika Singla	Initial Version

© IBM Corporation 2021. All rights reserved.