

# Hands-on Lab: Stored Procedures



**Estimated time needed:** 20 minutes

Stored Procedures in SQL are a type of database object that allow you to encapsulate a series of SQL statements into a single routine. They are stored in the database data dictionary and can be invoked from an application program or from the database command interface. Stored procedures can accept input parameters and return multiple values of output parameters. They can also include control-of-flow constructs such as loops and conditional statements. Stored procedures offer several benefits including improved performance, higher productivity, ease of use, and increased scalability. They also provide a mechanism for enforcing business rules and data integrity in the database system.

## Objectives

After completing this lab, you will be able to:

- Create stored procedures
- Execute stored procedures

## 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.

For this lab, you need to create a database PETS in the phpMyAdmin interface. Download the PETSALe-CREATE-v2.sql script below, upload it to console under the PETS database. Upon execution, 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](#)

## Stored Procedure: 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. 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**.

```
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 ;
```

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 PETSALe; END
```

2. 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`
```

Buttons: SELECT \*, SELECT, INSERT, UPDATE, DELETE, Clear, Format, Get auto

☐ Bind parameters

[ Delimiter ; ] ☐ Show this query here again ☐ Retain query box ☐ Rollback when finish

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;
```

Buttons: 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
```

☐ 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

3. You can view the created stored procedure routine RETRIEVE\_ALL. On the left panel, expand the **PETS** database option and click on **Procedures** to view the procedure.

Current server: phpMyAdmin demo - M

Recent Favorites

Type to filter these, Enter t

performance schema

PETS

Procedures

Tables

New

PETSALE

## Routines

☐ Check all Export Drop

Name	Type	Returns
<input type="checkbox"/> RETRIEVE_ALL	PROCEDURE	<a href="#">Edit</a> <a href="#">Execute</a> <a href="#">Export</a> <a href="#">Drop</a>

[phpMyAdmin Demo Server](#): Git information missing!

4. 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;
```

Structure SQL Search Query Export Import Operations Privileges Routines

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

Clear Format Get auto-saved query

☐ Bind parameters

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

### Error

SQL query: [Copy](#)

```
CALL RETRIEVE_ALL
```

MySQL said:

#1305 - PROCEDURE Mysql\_learners.RETRIEVE\_ALL does not exist

## Stored Procedure: Exercise 2

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

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 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 that:
  - 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 a MySQL IDE interface with a menu bar (Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines) and a toolbar. The main window is titled "Run SQL query/queries on database Mysql\_learners:". The SQL editor contains the following code:

```
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
```

Below the editor are buttons for "Clear", "Format", and "Get auto-saved query". There is a checkbox for "Bind parameters" which is unchecked. At the bottom, there is a "Delimit" section with a dropdown menu showing ";" and a checkbox for "Show this query here again" which is unchecked. Other checkboxes include "Retain query box" (unchecked), "Rollback when finished" (unchecked), and "Enable foreign key checks" (checked).

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'
(SALEPRICE * 0.25) WHERE ID = Animal_ID; ELSEIF Animal_Health = 'WORSE' THEN UPDATE PETSale SET SALEPRICE = SALEPRICE
PETSale SET SALEPRICE = SALEPRICE WHERE ID = Animal_ID; END IF; END
```

1. 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**

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;
```

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

2. 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;
```

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

Showing rows 0 - 4 (5 total)

CALL RETRIEVE\_ALL

☐ Show all

Number of rows: 25

Filter rows: Search this table

Options

☐ Show all

Number of rows: 25

Filter rows: Search this table

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

StructureSQLSearchQueryExportImportOperationsPrivilegesRoutines

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

4. 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;
```

```
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

## Conclusion

Congratulations! You have completed this lab on creating stored procedures in MySQL.

You are now able to:

- Write a stored procedure as per requirement
- Call or Execute a stored procedure
- Drop a stored procedure once its utility is over

## Author(s)

[Lakshmi Holla](#)

[Malika Singla](#)

[Abhishek Gagneja](#)

© IBM Corporation 2023. All rights reserved.