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.

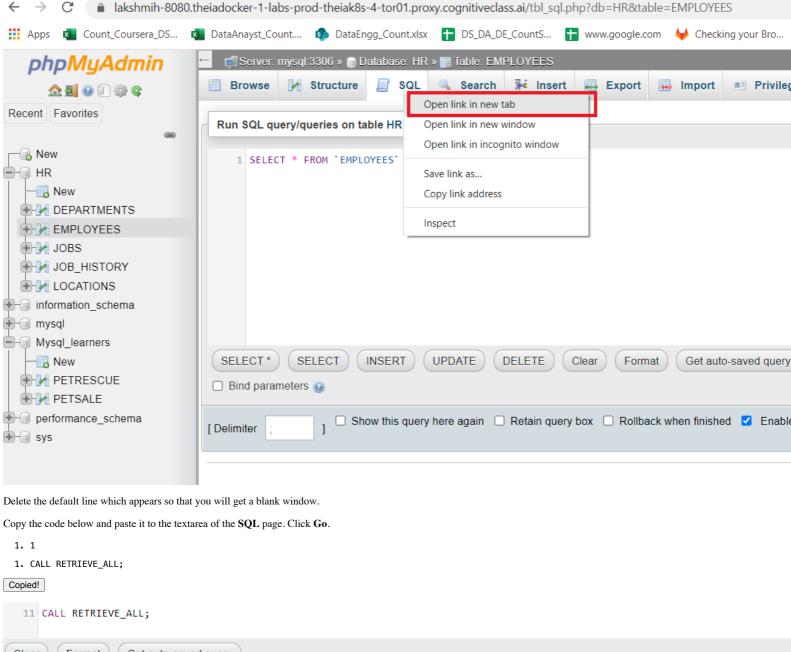
• To create the stored procedure routine, copy the code below and paste it to the textarea of the SQL page. Click Go.

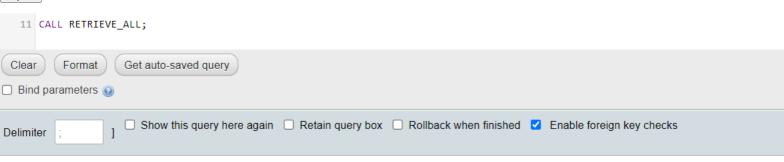
```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
 1. DELIMITER //
2.
3. CREATE PROCEDURE RETRIEVE_ALL()
4.
5. BEGIN
 6. SELECT * FROM PETSALE;
7. END //
  8. DELIMITER ;
Copied!
       Run SQL query/queries on database Mysql_learners: (a)
           1 DELIMITER //
           2
           3 CREATE PROCEDURE 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 <a>②</a>

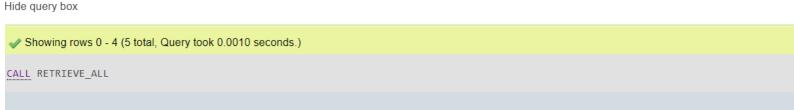
    Show this query here again
    Retain query box
    Rollback when finished
    Enable foreign key checks

     [ Delimiter
      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

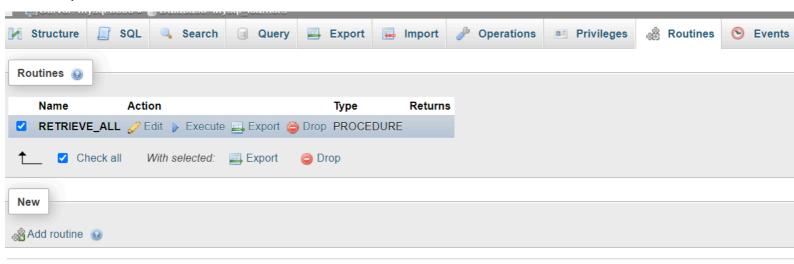






	☐ Show al	Number of	rows:	25 🗸	Filter rov	WS:	Search this table	
Options								
)	ANIMAL	SALEPRICE	SALED	ATE	QUANTITY			
1	Cat	450.09	2018-0	5-29	9			
2	Dog	666.66	2018-0	6-01	3			
3	Parrot	50.00	2018-0	6-04	2			
4	Hamster	60.60	2018-0	6-11	6			
5	Goldfish	48.48	2018-0	6-14	24			

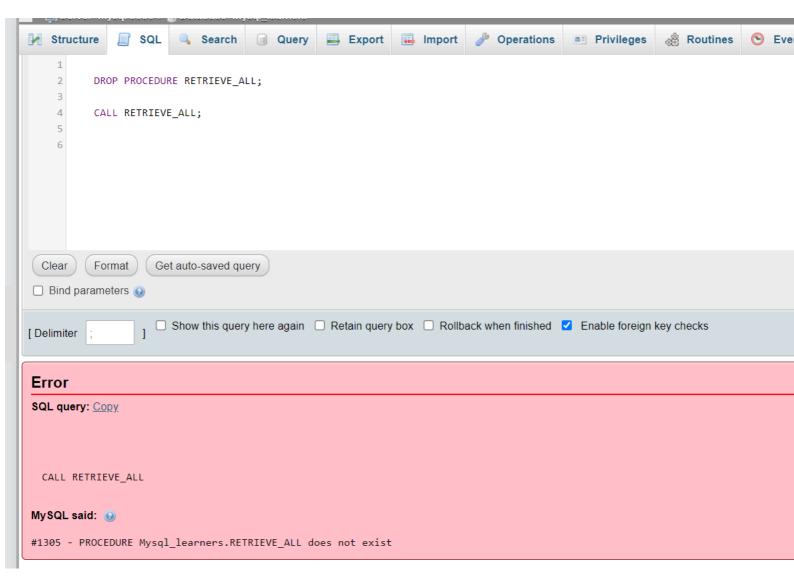
3. You can view the created stored procedure routine RETRIEVE_ALL. On the left panel, expand the mysql option. Click on **Procedures** then click on the **RETRIEVE_ALL** and view the procedure.



- 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.
- 1. 1 2. 2 3. 3

- DROP PROCEDURE RETRIEVE_ALL;
 GALL RETRIEVE_ALL;

Copied!



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 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 that:
 - o 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
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
1. DELIMITER @
2. CREATE PROCEDURE UPDATE_SALEPRICE (IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5))
3. BEGIN
4. IF Animal_Health = 'BAD' THEN
5. UPDATE PETSALE
6. SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.25)
7. WHERE ID = Animal_ID;
8. ELSEIF Animal_Health = 'WORSE' THEN
9. UPDATE PETSALE
```

```
11.
12.
            WHERE ID = Animal_ID;
        FLSF
 13.
            UPDATE PETSALE
            SET SALEPRICE = SALEPRICE
 14.
 15.
            WHERE ID = Animal_ID;
 16.
        END IF;
 17. END @
 18.
 19. DELIMITER ;
Copied!
     Structure

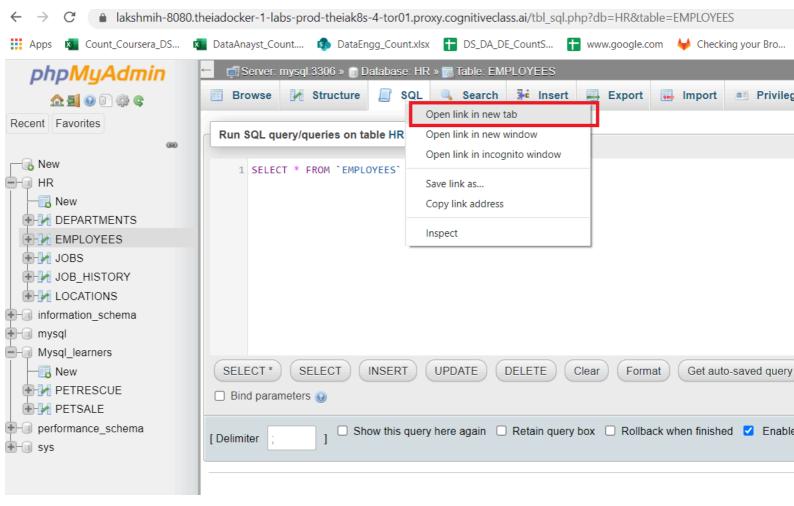
☐ SQL

                                         Query
                                                      Export
                                                                   - Import
                                                                                 Operations
                                                                                                  Privileges
                                                                                                                  Routines
                                                                                                                                  Events
                                Search
  Run SQL query/queries on database Mysql_learners: (a)
     15
     16
            ELSE
     17
                 UPDATE PETSALE
                 SET SALEPRICE = SALEPRICE
     18
     19
                 WHERE ID = Animal_ID;
     20
     21
            END IF;
     22
     23 END @
     24
     25 DELIMITER;
     26
   Clear
            Format
                       Get auto-saved query
  ☐ Bind parameters <a> ○</a>
                        □ Show this query here again □ Retain query box □ Rollback when finished ✓ Enable foreign key checks
 [ Delimiter
 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
  (SALEPRICE * 0.25) WHERE ID = Animal_ID; ELSEIF Animal_Health = 'WORSE' THEN UPDATE PETSALE SET SALEPRICE = SALEPRICE - (SALEPRIC
  PETSALE SET SALEPRICE = SALEPRICE WHERE ID = Animal_ID; END IF; END
```

SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.5)

10.

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.

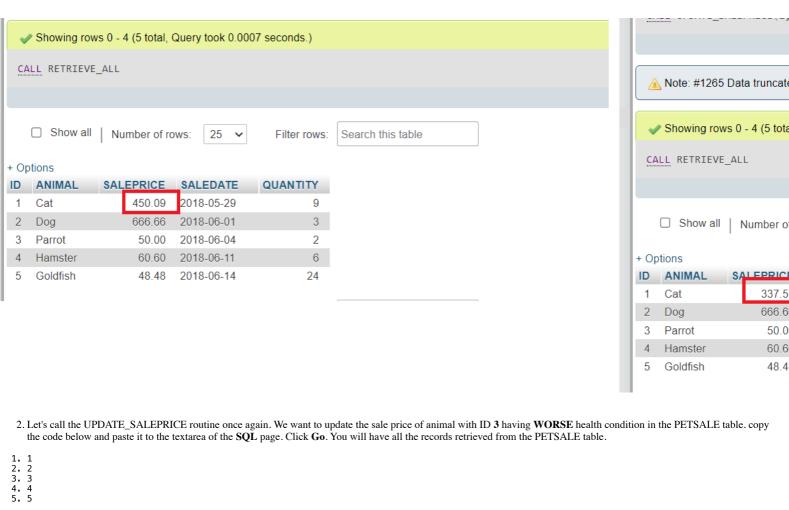
Copy the code below and paste it to the textarea of the \mathbf{SQL} page. Click \mathbf{Go} .

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

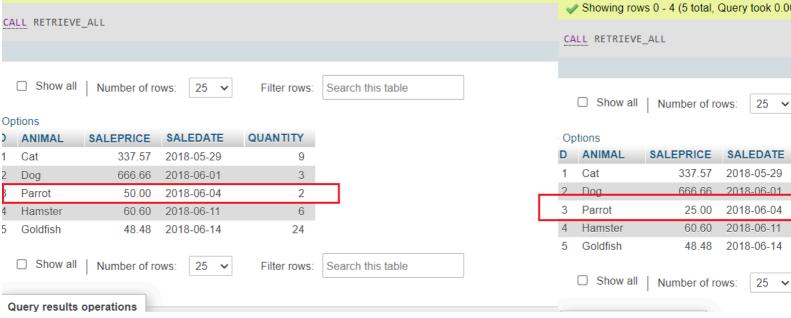
```
1. 1
2. 2
3. 3
4. 4
5. 5

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

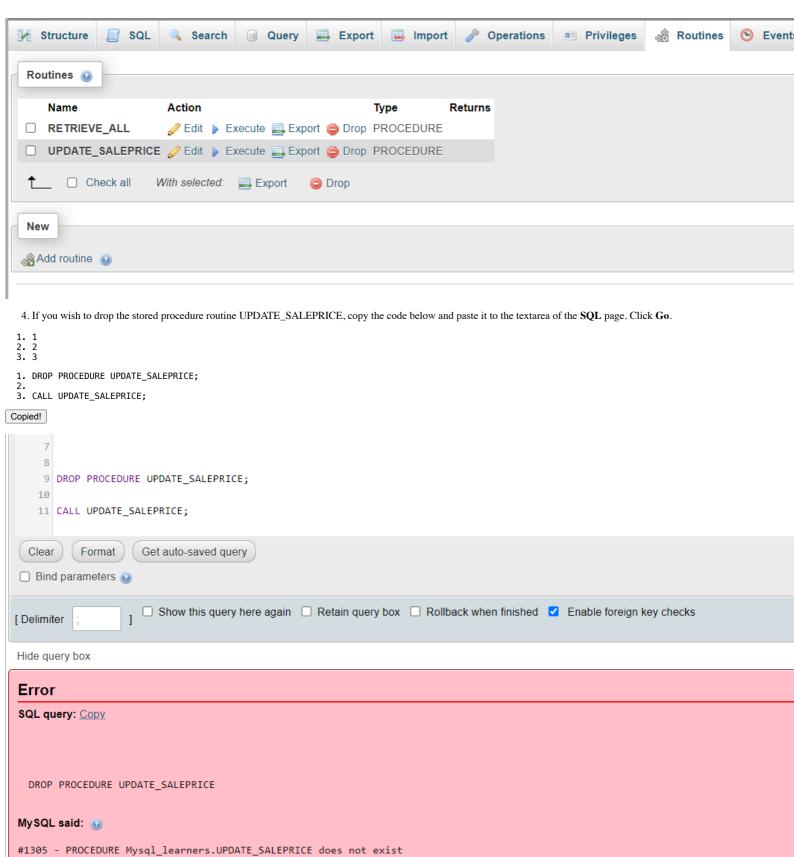
Copied!
```







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



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 Exectue a stored procedure
- Drop a stored procedure once its utility is over

Author(s)

Lakshmi Holla

Malika Singla

Abhishek Gagneja

Changelog

Date	Version	Changed by	Change Description
2023-10-31	0.4	Mercedes Schneider	QA Edits
2023-10-16	0.3	Abhishek Gagneja	Updated the instructions
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 2023. All rights reserved.