

### Hands-on Lab: Stored Procedures

Estimated time needed: 10 minutes

In this lab, you will create and execute stored procedures on IBM Db2 using SQL. A stored procedure is a set of SQL statements that are stored and executed on the database server. So instead of sending multiple SQL statements from the client to the server, you encapsulate them in a stored procedure on the server and send one statement from the client to execute them. Also, stored procedures can be useful if you have an SQL query that you write over and over again. You can save it as a stored procedure, and then just call it to execute it. In stored procedures, you can also pass parameters so that a stored procedure can act based on the passed parameter values.

### Software Used in this Lab

In this lab, you will use an IBM Db2 Database. Db2 is a Relational Database Management System (RDBMS) from IBM, designed to store, analyze and retrieve data efficiently.

To complete this lab you will utilize a Db2 database service on IBM Cloud. If you did not already complete this lab task earlier in this module, you will not yet have access to Db2 on IBM Cloud, and you will need to follow the lab below first:

• Hands-on Lab: Sign up for IBM Cloud, Create Db2 service instance and Get started with the Db2 console

### 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 Db2. 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 Db2 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

Please go through the lab below to learn how to upload and run a script on Db2 console (for this case, you need don't need to know anything else other than how to upload and run a script):

• Hands-on Lab: Create tables using SQL scripts and Load data into tables

### **Objectives**

After completing this lab, you will be able to:

- · Create stored procedures
- · Execute stored procedures

about:blank 1/15

### **Instructions**

When you approach the exercises in this lab, follow the instructions to run the queries on Db2:

- Go to the Resource List of IBM Cloud by logging in where you can find the Db2 service instance that you created in a previous lab under Services section. Click on the Db2-xx service. Next, open the Db2 Console by clicking on Open Console button. Click on the 3-bar menu icon in the top left corner and go to the Run SQL page. The Run SQL tool enables you to run SQL statements.
  - o If needed, follow Hands-on Lab: Sign up for IBM Cloud, Create Db2 service instance and Get started with the Db2 console

#### Exercise 1

In this exercise, you will create and execute a stored procedure to read data from 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.

1. 1 2. 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.

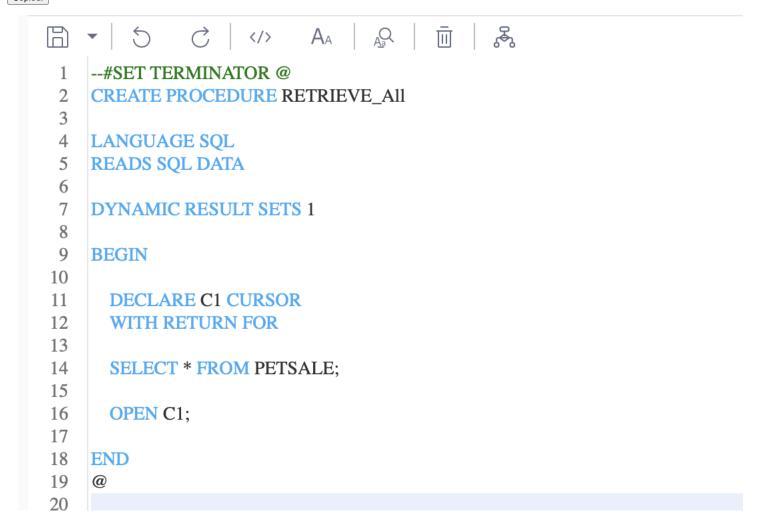
• To create the stored procedure routine, copy the code below and paste it to the textbox of the Run SQL page. Click Run all.

```
4. 4
 5. 5
6. 6
7. 7
10. 10
11. 11
12. 12
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
19. 19
 1.
         --#SET TERMINATOR @
        CREATE PROCEDURE RETRIEVE ALL
 2.
                                               -- Name of this stored procedure routine
 3.
 4.
        LANGUAGE SQL
                                               -- Language used in this routine
        READS SQL DATA
                                               -- This routine will only read data from the table
 6.
7.
        DYNAMIC RESULT SETS 1
                                               -- Maximum possible number of result-sets to be returned to the caller query
 8.
 9.
        BEGIN
10.
             DECLARE C1 CURSOR
                                               -- CURSOR C1 will handle the result-set by retrieving records row by row from the table
```

about:blank 2/15

```
12.
            WITH RETURN FOR
                                            -- This routine will return retrieved records as a result-set to the caller query
13.
            SELECT * FROM PETSALE;
                                            -- Query to retrieve all the records from the table
14.
15.
            OPEN C1;
                                            -- Keeping the CURSOR C1 open so that result-set can be returned to the caller query
16.
17.
18.
        END
19.
                                             -- Routine termination character
```

Copied!



3. To call the RETRIEVE ALL routine, copy the code below in a new blank script and paste it to the textbox of the Run SQL page. Click Run all. You will have all the records retrieved from the PETSALE table.

```
    CALL RETRIEVE_ALL;

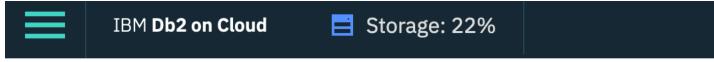
                              -- Caller query
Copied!
```

about:blank 3/15



4. You can view the created stored procedure routine RETRIEVE\_ALL. Click on the 3-bar menu icon in the top left corner and click **EXPLORE > APPLICATION OBJECTS > Stored Procedures**. Find the procedure routine RETRIEVE\_ALL from Procedures by clicking **Select All**. Click on the procedure routine **RETRIEVE\_ALL**.

about:blank 4/15



# STORED PROCEDURES

Filter by schema name or procedure name **Schemas** New implicit schema Select All ✓ AUDIT 2 procedures ✓ ZJH17769 2 procedures ✓ DB2INST1 1 procedure ✓ ERRORSCHEMA 0 procedure ✓ SQL74605 *0 procedure* ✓ ST\_INFORMTN\_SCHEMA 0 procedure

about:blank 5/15

5. If you wish to drop the stored procedure routine RETRIEVE\_ALL, copy the code below and paste it to the textbox of the Run SQL page. Click Run all.

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



### 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.

about:blank 6/15

ID 📤	ANIMAL	SALEPRI
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 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 textbox of the Run SQL page. Click Run all.

```
1. 1
 2. 2
3. 3
 8.8
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
27. 27
28. 28
         --#SET TERMINATOR @
 1.
         CREATE PROCEDURE UPDATE SALEPRICE (
 2.
 3.
             IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5) )
                                                                            -- ( { IN/OUT type } { parameter-name } { data-type }, ... )
 4.
         LANGUAGE SQL
                                                                             -- Language used in this routine
 6.
7.
8.
         MODIFIES SQL DATA
                                                                             -- This routine will only write/modify data in the table
 9.
10.
             IF Animal_Health = 'BAD' THEN
                                                                            -- Start of conditional statement
                  UPDATE PETSALE
                  SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.25)
12.
13.
                  WHERE ID = Animal ID;
14.
15.
             ELSEIF Animal_Health = 'WORSE' THEN
16.
17.
                  SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.5)
18.
                  WHERE ID = Animal_ID;
19.
             ELSE
20.
```

about:blank 7/15

```
21. UPDATE PETSALE
22. SET SALEPRICE = SALEPRICE
23. WHERE ID = Animal_ID;
24.
25. END IF; -- End of conditional statement
26.
27. END
28. @ -- Routine termination character
```

```
\overrightarrow{C} \langle \cdot \rangle A_A A_A
     --#SET TERMINATOR @
 1
 2
    CREATE PROCEDURE UPDATE_SALEPRICE (
      IN Animal_ID INTEGER, IN Animal_Health VARCHAR(5))
 3
 4
 5
    LANGUAGE SQL
 6
    MODIFIES SQL DATA
 7
 8
    BEGIN
 9
10
      IF Animal Health = 'BAD' THEN
         UPDATE PETSALE
11
12
         SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.25)
13
         WHERE ID = Animal ID;
14
15
      ELSEIF Animal_Health = 'WORSE' THEN
16
         UPDATE PETSALE
17
         SET SALEPRICE = SALEPRICE - (SALEPRICE * 0.5)
18
         WHERE ID = Animal ID;
19
20
      ELSE
21
         UPDATE PETSALE
         SET SALEPRICE = SALEPRICE
22
23
         WHERE ID = Animal ID;
24
25
      END IF:
26
27
    END
28
     @
29
```

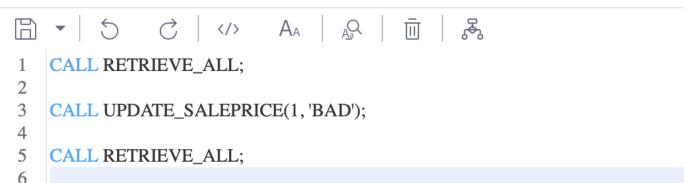
3. Let's call the UPDATE\_SALEPRICE routine. We want to update the sale price of animal with ID 1 having BAD health condition in the PETSALE table. Copy the code below in a **new blank script** and paste it to the textbox of the **Run SQL** page. Click **Run all**. You will have all the records retrieved from the PETSALE table.

```
1. 1
2. 2
3. 3
4. 4
5. 5
1. CALL RETRIEVE_ALL;
```

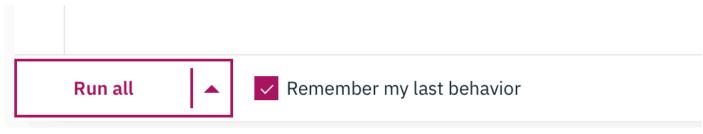
about:blank 8/15

```
3. CALL UPDATE_SALEPRICE(1, 'BAD'); -- Caller query
4.
5. CALL RETRIEVE_ALL;
Copied!
```

about:blank 9/15



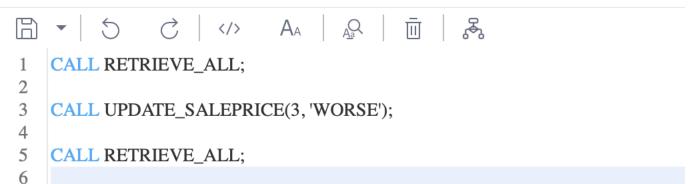
about:blank 10/15



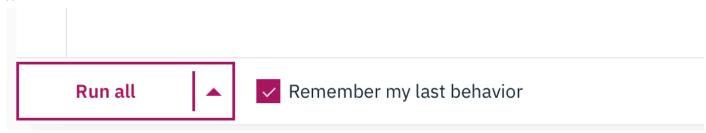
4. Let's call the UPDATE\_SALEPRICE routine once again. We want to update the sale price of animal with ID 3 having WORSE health condition in the PETSALE table. Copy the code below and paste it to the textbox of the Run SQL page. Click Run all. You will have all the records retrieved from the PETSALE table.

```
1. 1
2. 2
3. 3
4. 4
5. 5
1. CALL RETRIEVE_ALL;
2.
3. CALL UPDATE_SALEPRICE(3, 'WORSE'); -- Caller query
4.
5. CALL RETRIEVE_ALL;
Copied!
```

about:blank 11/15

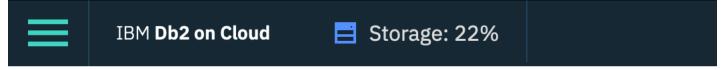


about:blank 12/15



5. You can view the created stored procedure routine UPDATE\_SALEPRICE. Click on the 3-bar menu icon in the top left corner and click EXPLORE > APPLICATION OBJECTS > Stored Procedures. Find the procedure routine UPDATE\_SALEPRICE from Procedures by clicking Select All. Click on the procedure routine UPDATE\_SALEPRICE.

about:blank 13/15



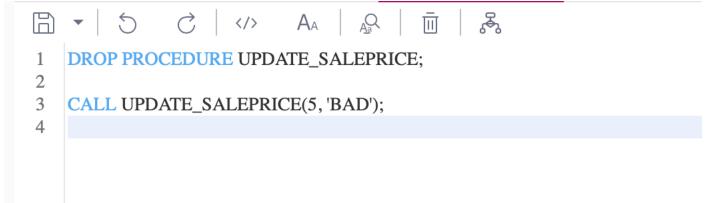
# STORED PROCEDURES

Filter by schema name or procedure name	Q
Schemas  ✓ Select All   → New implicit sch	nema
✓ AUDIT 2 procedures	
✓ ZJH17769 2 procedures	
✓ DB2INST1 1 procedure	
✓ ERRORSCHEMA 0 procedure	
SQL74605 <i>O procedure</i>	
ST_INFORMTN_SCHEMA 0 procedure	

6. If you wish to drop the stored procedure routine UPDATE\_SALEPRICE, copy the code below and paste it to the textbox of the Run SQL page. Click Run all.

about:blank 14/15

<sup>1. 1
1.</sup> DROP PROCEDURE UPDATE\_SALEPRICE;
Copied!



Congratulations! You have completed this lab, and you are ready for the next topic.

### Author(s)

• Sandip Saha Joy

## **Other Contributor(s)**

-

# Changelog

Date	Version	Changed by	<b>Change Description</b>
2023-05-10	1.2	Eric Hao & Vladislav Boyko	Updated Page Frames
2020-12-25	1.1	Steve Ryan	ID Reviewed
2020-12-14	1.0	Sandip Saha Joy	Created initial version

© IBM Corporation 2023. All rights reserved.

about:blank 15/15