

Indian Institute of Information Technology Sri City

Database Management Systems LAB-07

Date: 12/10/2021

TOPIC :Procedures

Instructors: Dr. Odelu Varnaga, Dr. Rakesh, and Dr. Annushree

Stored procedures

Creating a procedure:

```
Delimiter //
Create procedure <procedure_name> (<parameters>)
Begin
    • <variables>
    • <queries>
End //
Delimiter ;

Call <procedure_name> ;
```

NOTE: Don't use keywords for creating procedures

Example:

1. mysql> delimiter aa
mysql> create procedure sar()
begin
select count(*) from borrower;
end aa
mysql> delimiter !
2. mysql> call sar() !
3. mysql> call sar !

Dropping a procedure:

```
DROP PROCEDURE [IF EXISTS] <stored_procedure_name> <delimiter_symbol>
```

Example

1. mysql> drop procedure sar!

To know the procedures already existing in database:

- SHOW PROCEDURE STATUS;
- mysql> SELECT
 routine_name
 FROM
 information_schema.routines
 WHERE
 routine_type = 'PROCEDURE'
 AND routine_schema = '<your_db_name>';

Variables declaration and assignment

Declaration:

DECLARE variable_name datatype(**size**) [**DEFAULT** default_value];

Assignment:

SET variable_name = **value**;

Example:

```
mysql> delimiter //
mysql> create procedure sar()
begin
  declare a,b int default 0;
  set a=10;
  select count(*)+a into b from borrower;
  select b;
end//
Delimiter ;
```

Parameters:

- In MySQL, a parameter has one of three modes: IN,OUT, or INOUT.

Defining a parameter:

[IN | OUT | INOUT] parameter_name datatype[(length)]

In above syntax,

1. First, specify the parameter mode, which can be IN , OUT or INOUT , depending on the purpose of the parameter in the stored procedure.
2. Second, specify the name of the parameter. The parameter name must follow the naming rules of the column name in MySQL.
3. Third, specify the data type and maximum length of the parameter.

In parameter Example:

```
mysql> delimiter //
mysql> create procedure sar( in name varchar(10) )
begin
select * from depositor where customer_name=name;
end//
mysql> call sar('Johnson');
```

Out Parameter Example:

```
mysql> delimiter //
mysql> create procedure temp(in balance int,out count int)
begin
select count(*) into count from account where account.balance >= balance;
end //
mysql> call temp(300,@a);
mysql> select @a;
```

IN OUT parameter Example:

```
mysql> delimiter //
mysql> create procedure setcounter(inout counter int,in increment int)
begin
set counter = counter + increment ;
end//
mysql> set @counter=1;
mysql> call setcounter(@counter,1);
mysql> call setcounter(@counter,10);
mysql> select @counter;
```

Conditional statements:

IF:

- The IF statement has three forms: simple IF-THEN statement, IF-THEN-ELSE statement, and IF-THEN-ELSEIF- ELSE statement.

```
IF condition THEN
  statements;
ELSEIF elseif-condition THEN
  elseif-statements;
...
ELSE
  else-statements;
END IF;
```

Example:

```
mysql> delimiter $$
mysql> create procedure customerlevel( in id varchar(10),out level varchar(20))
begin
  declare bal numeric(12,2) default 0;
  select balance into bal from account where account_number = id;
  if bal > 700 then
    set level='PLATINUM';
  elseif bal<=700 and bal>300 then
    set level='GOLD';
  else
    set level='SILVER';
  end if;
end $$
```

```
mysql> delimiter ;
mysql> call customerlevel('A-101',@a);
```

```
mysql> select @a;
```

CASE statement:

The CASE statement has two forms: simpleCASE and searched CASE statements.

simpleCASE syntax:

```
CASE case_value
  WHEN when_value1 THEN statements
  WHEN when_value2 THEN statements
  ...
  [ELSE else-statements]
END CASE;
```

simpleCASE Example:

```
mysql> create procedure account_check(in id varchar(10),out status varchar(10))
begin
declare temp varchar(10) default 'Absent';
case id
when 'A-101' then
set status='Present';
Select status;
when 'A-102' then
set status='Present';
when 'A-201' then
set status='Present';
when 'A-215' then
set status='Present';
when 'A-217' then
set status='Present';
when 'A-222' then
set status='Present';
when 'A-305' then
set status='Present';
else
Set status='Absent';
end case;
end $$
```

```
mysql> delimiter ;
mysql> call account_check('A-305',@a);
```

```
mysql> select @a;
```

searched CASE syntax:

```
CASE
  WHEN search_condition1 THEN statements
  WHEN search_condition1 THEN statements
  ...
  [ELSE else-statements]
END CASE;
```

searched CASE example:

```
mysql> create procedure customerlevel2( in id varchar(10),out level varchar(20))
begin
  declare bal numeric(12,2) default 0;
  select balance into bal from account where account_number = id;
  case
  when bal>700 then
  set level='PLATINUM';
  when bal<=700 and bal>300 then
  set level='GOLD';
  else
  set level='SILVER';
  end case;
  end $$
```

```
mysql> delimiter ;
mysql> call customerlevel2('A-101',@a);
```

```
mysql> select @a;
```

Loop:

- The LOOP statement allows you to execute one or more statements repeatedly.

Basic syntax of the LOOP statement:

```
[begin_label:] LOOP  
    statement_list  
END LOOP [end_label]
```

- Typically, you terminate the loop when a condition is satisfied by using the LEAVE statement. Repeat the loop using ITERATE

```
[label]: LOOP  
    ...  
    IF condition THEN  
        LEAVE [label];  
    END IF;  
    ...  
END LOOP;
```

Loop Example

```
DELIMITER $$  
CREATE PROCEDURE LoopDemo()  
BEGIN  
    DECLARE x INT;  
    DECLARE str VARCHAR(255);  
    SET x = 1;  
    SET str = "";  
loop_label: LOOP  
    IF x > 10 THEN  
        LEAVE loop_label;  
    END IF;  
    SET x = x + 1;  
    IF (x mod 2) THEN  
        ITERATE loop_label;  
    ELSE  
        SET str = CONCAT(str,x,',');  
    END IF;  
    END LOOP;  
    SELECT str;  
END$$
```

```
DELIMITER ;
```

While loop

```
[begin_label:] WHILE search_condition DO  
    statement_list  
END WHILE [end_label]
```

While example:

```
DELIMITER $$  
CREATE PROCEDURE dowhile(inout num int)  
BEGIN  
    WHILE num > 0 DO  
        SET num = num - 1;  
    END WHILE;  
END$$  
DELIMITER ;
```

References for functions:

<https://dev.mysql.com/doc/refman/8.0/en/numeric-functions.html>

<https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html>

<https://dev.mysql.com/doc/refman/8.0/en/string-functions.html>

Exercise:

1. Create a procedure to insert a tuple into any table of bank schema
2. Drop the created procedure
3. Create a procedure that takes any 2 numbers and returns their sum and multiplication.
4. Write a procedure with only one parameter such that it returns the square root of any given number
5. Write a Procedure that returns no.of characters in any given string
6. Write a procedure with only one parameter such that it displays the factorial of the given number
7. Write a procedure(that accepts 2 arguments,one argument has old name the other holds new name) to update name of the existing customer to a new name
8. Procedure that accepts customer id and displays whether he has loan or not
9. Display city of given customer and If he/she is not an existing customer create a new entry in customer table
10. Display account numbers of customers whose balance is above the given amount