MySQL – Bài thực hành số 3Chương trình hàm, thủ tục

GHI NHỚ

**CREATE SP**

DELIMITER //

CREATE PROCEDURE sp\_name

BEGIN

SELECT Clause;

END //

DELIMITER ;

**CALL** sp\_name()

**DROP SP**

DROP {PROCEDURE | FUNCTION} [IF EXISTS] sp\_name

**ALTER SP**

ALTER PROCEDURE proc\_name [characteristic ...]

**Declaring variables**

DECLARE variable\_name datatype(size) DEFAULT default\_value;

**Assigning variables**

DECLARE total\_count INT DEFAULT 0 ;

SET total\_count = 10;

DECLARE total\_products INT DEFAULT 0;

SELECT COUNT(\*) INTO total\_products

FROM products;

**In MySQL, a parameter has one of three modes: IN, OUT and INOUT.**

IN: the default mode. IN indicates that a parameter can be passed into stored procedures but any modification inside stored procedure does not change parameter.

OUT: indicates that stored procedure can change this parameter and pass back to the calling program.

INOUT: obviously this mode is combined of IN and OUT mode; you can pass parameter into stored procedure and get it back with the new value from calling program.

**MySQL supports cursor in stored procedures, functions and triggers.**

DECLARE cursor\_name CURSOR FOR SELECT\_statement

OPEN cursor\_name;

FETCH cursor\_name INTO variable list;

CLOSE cursor\_name;

One of the most important point when working with cursor is you should use a NOT FOUND handler to avoid raising a fatal “no data to fetch” condition.

**Handler**

DECLARE type HANDLER FOR condition1, condition2, condition3, ... statement;

***condition\_name***

SQLWARNING: Matches any SQLSTATE that begins with 01.

NOT FOUND: Matches any SQLSTATE that begins with 02.

SQLEXCEPTION:

***handler\_type:***

CONTINUE

EXIT

UNDO

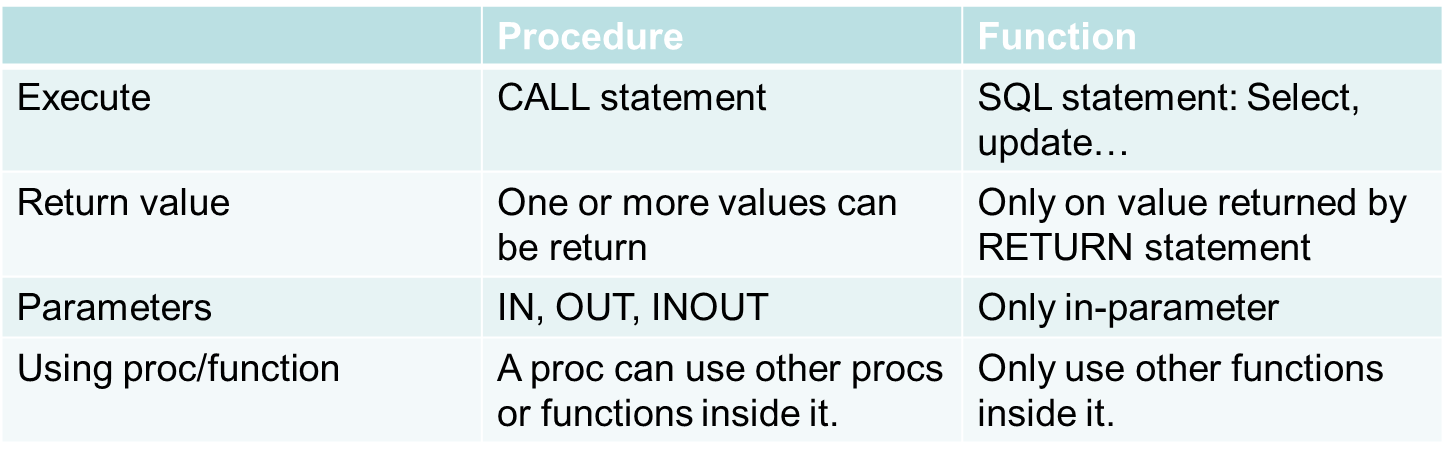
**CREATE FUNCTION**

CREATE FUNCTION name ([parameterlist])

RETURNS datatype [options]

sqlcode

**DROP FUNCTION** [IF EXISTS] name

****

**2- Hàm (Function)**

Giống như thủ tục, hàm ( function) là nhóm các lệnh PL/SQL thực hiện chức năng nào đó. Khác với thủ tục, các hàm sẽ trả về một giá trị ngay tại lời gọi của nó.

Hàm cũng có thể được lưu giữ ngay trên database dưới dạng Store procedure.

Cú pháp tạo Hàm.

-- function\_name: Tên hàm

-- argument: Tên tham số

-- mode: IN hoặc không cần viết.

-- datatype: Kiểu dữ liệu của tham số

CREATE FUNCTION <function\_name>

(

[

[mode1] argument1 datatype1,

[mode2] argument2 datatype2,

...

]

)

RETURNS datatype

BEGIN

-- Khai báo biến sử dụng

-- Code nội dung của hàm

-- Trả về giá trị của hàm.

END;

-- Một hàm có tham số:

CREATE FUNCTION Sum\_Ab(a Integer, b Integer)

RETURNS Integer

Begin

return a + b;

End;

-- Một hàm không tham số:

CREATE FUNCTION Get\_Current\_Datetime()

RETURNS Date

Begin

return current\_date();

End;

**Hủy Function**

-- Hủy Function

DROP FUNCTION <function\_name>;

-- Ví dụ:

DROP FUNCTION My\_Function;

-- Ví dụ:

DROP FUNCTION IF EXISTS `My\_Function`;

**Gọi hàm**

-- Khi gọi hàm phải khai báo một biến trả về

-- Khai báo một biến c.

Declare c Integer;

-- Gọi hàm.

Set c = Sum\_Ab(10, 100);

**Ví dụ tạo một hàm.**

**Đây là một ví dụ tạo một hàm đầu tiên của bạn với MySQL:**

1. **Tạo một hàm (Function)**
2. **Biên dịch hàm này**
3. **Chạy hàm**

-- Huỷ function My\_Sum nếu nó đã tồn tại.

-- (Để cho phép tạo lại)

DROP function if Exists `My\_Sum`;

-- Khi lập trình hàm/thủ tục bạn cần sử dụng dấu chấm phẩy để

-- ngăn cách các câu lệnh khác nhau.

-- Sử dụng DELIMITER $$ để cho phép sử dụng dấu chấm phẩy.

DELIMITER $$

Create Function My\_Sum(P\_a Float, p\_B Float) returns Float

Begin

-- Khai báo một biến Float

Declare v\_C Float;

-- Sét giá trị cho biến v\_C

Set V\_C = p\_A + p\_B;

-- Giá trị trả về của hàm.

return v\_C;

End;

<https://o7planning.org/vi/10325/lap-trinh-database-voi-mysql#a265360>

**Ví dụ trong Slide**

DELIMITER $$

CREATE PROCEDURE GetOfficeByCountry(IN countryName VARCHAR(255))

BEGIN

SELECT city, phone

FROM offices

WHERE country = countryName;

END $$

DELIMITER ;

CALL GetOfficeByCountry('USA')

DELIMITER $$

CREATE PROCEDURE CountOrderByStatus(

IN orderStatus VARCHAR(25), OUT total INT)

BEGIN

SELECT count(orderNumber) INTO total

FROM orders

WHERE status = orderStatus;

END$$

DELIMITER ;

So to get number of shipped orders, we just perform following statements

CALL CountOrderByStatus('Shipped',@total);

To get number of orders in process we do the same as above

CALL CountOrderByStatus('in process',@total);

SELECT @total AS total\_in\_process;

DELIMITER $$

CREATE PROCEDURE Capitalize(INOUT str VARCHAR(1024))

BEGIN

DECLARE i INT DEFAULT 1;

DECLARE myc, pc CHAR(1);

DECLARE outstr VARCHAR(1000) DEFAULT str;

WHILE i <= CHAR\_LENGTH(str) DO

SET myc = SUBSTRING(str, i, 1);

SET pc = CASE WHEN i = 1 THEN ' '

ELSE SUBSTRING(str, i - 1, 1)

END;

IF pc IN (' ', '&', '''', '\_', '?', ';', ':', '!', ',', '-', '/', '(', '.') THEN

SET outstr = INSERT(outstr, i, 1, UPPER(myc));

END IF;

SET i = i + 1;

END WHILE;

SET str = outstr;

END$$

DELIMITER ;

Using Capitalize stored procedure:

SET @str = 'mysql stored procedure tutorial';

CALL Capitalize(@str);

SELECT @str;

DELIMITER $$

CREATE PROCEDURE LoopProc()

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 ;

// Cursor

CREATE PROCEDURE curdemo()

BEGIN

DECLARE done INT DEFAULT FALSE;

DECLARE a CHAR(16);

DECLARE b, c INT;

DECLARE cur1 CURSOR FOR SELECT id,data

FROM test.t1;

DECLARE cur2 CURSOR FOR SELECT i FROM test.t2;

DECLARE CONTINUE HANDLER FOR NOT FOUND

SET done = TRUE;

OPEN cur1;

OPEN cur2;

read\_loop: LOOP

FETCH cur1 INTO a, b;

FETCH cur2 INTO c;

IF done THEN

LEAVE read\_loop;

END IF;

IF b < c THEN

INSERT INTO test.t3 VALUES (a,b);

ELSE

INSERT INTO test.t3 VALUES (a,c);

END IF;

END LOOP;

CLOSE cur1;

CLOSE cur2;

END;

// Handler

CREATE FUNCTION order\_of\_customer(customer\_id INT) RETURNS INT

READS SQL DATA

BEGIN

DECLARE Order\_id INT;

DECLARE EXIT HANDLER FOR NOT FOUND RETURN NULL;

SELECT OrderNumber INTO Order\_id

FROM Order

WHERE CustomerNumber = customer\_id;

RETURN Order\_id;

**BÀI TẬP**1. Tạo một stored procedure tên là displayFilmInfo nhận category\_id và language\_id  
như là các tham số:  
• Nếu category\_id và language\_id được chỉ rõ, trả lại thông tin các film có  
category\_id và language bằng các giá trị truyền vào.  
• Nếu 0 được truyền như một tham số cho language\_id thì trả lại thông tin  
các film có category\_id là tham số truyền vào.  
• Nếu 0 được truyền như một tham số cho category\_id thì trả lại các film có  
language\_id là tham số truyền vào.

2. Viết hàm tính tổng số film được thuê của của hàng trong tháng, năm. Với tham số  
đầu vào là mã của hàng, tháng, năm. Sau đó sử dụng hàm trong lệnh SELECT đưa ra các cửa hàng và tổng số film được thuê tại cửa hàng trong tháng 2/2006.

3. Viết stored procedure thực hiện công việc sau:  
Giảm p% giá thuê cho n film có số lượng thuê ít nhất trong tháng, năm lấy ra từ date(Trong đó p, n, date là tham số đầu vào).  
*Gợi ý: Thay vì cách sử dụng lệnh cập nhật theo lô. Làm các bước sau: Thực hiện lệnh  
SELECT lấy ra n film cần tìm, sau đó sử dụng con trỏ duyệt các* film\_id *từ lệnh SELECT trên và thực hiện cập nhật film đó*