MySQL Stored Procedures By Mayur Purushvani

CREATE PROCEDURE :

```
DELIMITER $$

CREATE PROCEDURE GetCustomers()
BEGIN
        SELECT
                customerName,
                city,
                state,
                postalCode,
                country
        FROM
                customers
        ORDER BY customerName;
END$$
DELIMITER ;
```

- ➔ CALL GetCustomers();

ADVANTAGES :

- ➔ Reduce network traffic
- ➔ Make database more secure

DISADVANTAGES :

- ➔ Resource Usage
- ➔ Maintenances
- ➔ Troubleshooting

Delimiter :

- ➔ The delimiter_character may consist of a single character or multiple characters e.g., // or $$. However, you should avoid using the backslash (\) because this is the escape character in MySQL.
- ➔ For example, this statement changes the delimiter to //

DROP PROCEDURE :

- ➔ DROP PROCEDURE ‘[PROCEDURE_NAME’]

<u>MySQL stored procedure parameters :</u>

<u>IN :</u>

- ➔ IN is the default mode. When you define an IN parameter in a stored procedure, the calling program has to pass an argument to the stored procedure. In addition, the value of an IN parameter is protected.

- ➔
```
DELIMITER //

CREATE PROCEDURE GetEmployee( IN name VARCHAR(20))

BEGIN

 SELECT * FROM EMPLOYEE WHERE SALARY>30000

END //

DELIMITER ;
```

<u>OUT :</u>

- ➔ The value of an OUT parameter can be changed inside the stored procedure and its new value is passed back to the calling program.

- ➔
```
DELIMITER //

CREATE PROCEDURE GetEmployee( IN name VARCHAR(20), OUT phone
VARCHAR(20))

BEGIN

SELECT * FROM EMPLOYEE WHERE SALARY>30000

END //

DELIMITER ;
```

<u>INOUT :</u>

- ➔ An INOUT parameter is a combination of IN and OUT parameters. It means that the calling program may pass the argument, and the stored procedure can modify the INOUT parameter, and pass the new value back to the calling program.

- ➔ DELIMITER //

  CREATE PROCEDURE GetEmployee(INOUT name VARCHAR(20) IN salary VARCHAR(20))

  BEGIN

  SELECT * FROM EMPLOYEE WHERE SALARY>30000

  END //

  DELIMITER ;

<u>Declare Variable :</u>

- ➔ DECLARE totalsize VARCHAR(20) DEFAULT 0;
- ➔ SET totalsize =10;

<u>LISTING STORED PROCEDURE :</u>

- ➔ SHOW PROCEDURE STATUS;
- ➔ SHOW PROCEDURE STATUS WHERE DB='EMPLOYEE';

<u>IF STATEMENT :</u>

- ➔ IF salary >3000

      SET salary_level = 'Good;

- ➔ Else

      SET salary_level = 'Average';

- ➔ ENF IF;

CASE Statement :

```
CASE country
        WHEN ‘India’ THEN
                SET shipping = “2 days”;
        WHEN ‘Canada’ THEN
                SET shipping = “1 days”;
        ELSE
                SET shipping = “3 days”;
END CASE;
```

MySQL LOOP :

```
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;
```

MySQL WHILE LOOP :

```
WHILE counter <= day DO
     CALL InsertCalendar(dt);
     SET counter = counter + 1;
     SET dt = DATE_ADD(dt,INTERVAL 1 day);
END WHILE;
```

MySQL REPEAT LOOP :

```
        REPEAT
                SET result = CONCAT(result,counter,',');
                SET counter = counter + 1;
        UNTIL counter >= 10
        END REPEAT;
```

MySQL LEAVE : [To Exit from stored procedure]

```
        CREATE PROCEDURE sp_name()
                sp: BEGIN
          IF condition THEN
             LEAVE sp;
          END IF;
        END$$
```

MySQL Cursor : [Read-only, Non-scrollable, Asensitive]

```
BEGIN
        DECLARE finished INT DEFAULT 0;
        DECLARE email_list varchar(500) default “”;
        DECLARE email varchar(30)default “”;
        DECLARE user_data CURSOR FOR SELECT email from employee_list limit 5;
        DECLARE CONTINUE HANDLER FOR NOT FOUND SET finished = 1;
        OPEN user_data;
        get_user_email : loop
        FETCH cursor_name INTO email;
        IF finished = 1 THEN
                LEAVE get_user_email;
        END IF;
        SET email_list = CONCAT(email_list,”, “,email);
        END LOOP get_user_email;
        CLOSE user_data;
        SELECT email_list;
END
```