* Stored Procedures:

PL/SQL is a block-structured language that enables developers to combine the power of SQL with procedural statements.  
A stored procedure in PL/SQL is nothing but a series of declarative SQL statements which can be stored in the database catalogue. A procedure can be thought of as a function or a method.

Syntax: CREATE [OR REPLACE] PROCEDURE procedure\_name [(parameter\_name [IN | OUT | IN OUT] type [, ...])]

{IS | AS}

BEGIN

< SQL statements >

END procedure\_name;

Examples:

Example 1:

use lib;

DELIMITER &&

create procedure get\_User()

BEGIN

declare S int default 101;

case S

when 101 THEN select 'Elena';

when 102 then select 'Damon';

when 103 then select 'Stephen';

else select 'No such user';

begin

end

end case;

END&&

call get\_User();

Example 2:

use emp

DELIMITER &&

create procedure getemployee()

BEGIN

DECLARE

sal\_count INTEGER;

SELECT \* FROM employee WHERE salary> 45000;

SELECT COUNT(salary) AS sal\_count FROM employee;

END&&

call getemployee();

Example 3:

CREATE OR REPLACE PROCEDURE emp\_salary\_increase (emp\_id IN emptbl.empID%type, salary\_inc IN OUT emptbl.salary%type)

IS tmp\_sal number;

BEGIN

SELECT salary INTO tmp\_sal FROM emp\_tbl WHERE empID = emp\_id;

IF tmp\_sal between 10000 and 20000 THEN

salary\_inout := tmp\_sal \* 1.2;

ELSIF tmp\_sal between 20000 and 30000 THEN

salary\_inout := tmp\_sal \* 1.3;

ELSIF tmp\_sal > 30000 THEN

salary\_inout := tmp\_sal \* 1.4;

END IF;

END;

Call emp\_salary\_increase();

* Functions :

A stored function is a special kind stored program that returns a single value. Typically, you use stored functions to encapsulate common formulas or business rules that are reusable among SQL statements or stored programs.

Syntax:

DELIMITER $$

create function function nameI param1,param2…)

returns datatype

[not] DETERMINISTIC

Begin

---statements

End $$

DELIMITER;

Example:

use emp

DELIMITER $$

CREATE FUNCTION EmpLevel(sal DECIMAL(10,2))

RETURNS VARCHAR(20)

DETERMINISTIC

BEGIN

DECLARE EmpLevel VARCHAR(20);

IF sal > 50000 THEN

SET EmpLevel = 'PLATINUM';

ELSEIF (sal >= 50000 AND

sal <= 10000) THEN

SET EmpLevel = 'GOLD';

ELSEIF sal < 10000 THEN

SET EmpLevel = 'SILVER';

END IF;

-- return the customer level

RETURN (EmpLevel);

END$$

DELIMITER ;

Calling in SQL statement:

SELECT empname, EmpLevel(salary) FROM employee;

Example 2:

use emp

DELIMITER $$

CREATE FUNCTION Emp\_Occupation( deptid int)

RETURNS VARCHAR(20)

DETERMINISTIC

BEGIN

DECLARE emp\_occupation VARCHAR(20);

IF deptid > 300 THEN

SET emp\_occupation = 'Tester';

ELSEIF (deptid <= 200 AND

deptid >= 300) THEN

SET emp\_occupation = 'Programmer';

ELSEIF deptid < 100 THEN

SET emp\_occupation = 'Consultant';

END IF;

-- return the customer occupation

RETURN (emp\_occupation);

END$$

Calling in SQL statement:

SELECT empname, deptid, Emp\_Occupation(deptid) FROM employee;

Example 3:

DELIMITER $$

CREATE FUNCTION calcProfit(cost FLOAT, price FLOAT) RETURNS DECIMAL(9,2)

BEGIN

DECLARE profit DECIMAL(9,2);

SET profit = price-cost;

RETURN profit;

END$$

DELIMITER ;

SELECT \*, calcProfit(prod\_cost,prod\_price) AS profit FROM products;