

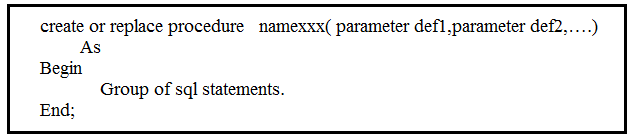
The project contains the 1 or more modules.

Module contains the 1 or more Applications.

**1. Stored Procedure:-** In database programming, if any group of sql statements execution is required repeatedly then we define those sql statements as group. We call that group repeatedly based on our requirement. The stored procedures are **procedure or function.**

1.1.Hence procedure is group of statements that performs a business logic. This procedure is always stored in database permanently for future purpose and hence name stored procedure. Usually stored procedure is created by DBA not by Java developer.

Syntax:-

 Where procedure can have 3 types of parameters.

1. IN:- It provides value to procedure from outside the procedure.

2.OUT:- Procedure supplies values to outside environment by out type parameter.

3.INOUT:- It used to provide input and to collect output.

The parameter definition syntax:

Parametername [IN/Out/INOUT] datatype;

Ex:-

SQL> create or replace procedure abc(a IN number,b IN number,c OUT number) as

2 begin

3 c:=a+b;

4 end;

5 /

Procedure created.

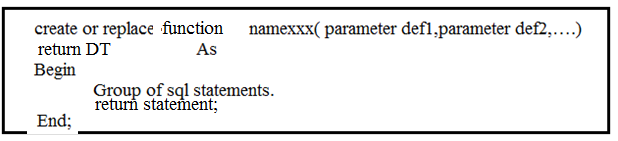
Note :- If procedure is created with compile time errors, then to know compilation errors,use following command.

SQL>show errors.

It will displays list of errors.

1.2. Hence Function is group of statements that performs a business logic. This function is always stored in database permanently for future purpose and hence name stored procedure. Usually stored procedure is created by DBA not by Java developer.

Syntax:-

 Where function can have 3 types of parameters.

1. IN:- It provides value to function from outside the function.

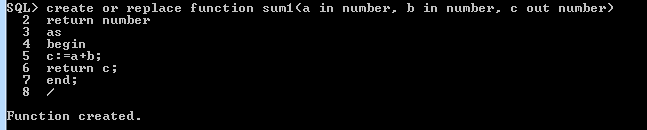
2.OUT:- function supplies values to outside environment by out type parameter.

3.INOUT:- It used to provide input and to collect output.

The parameter definition syntax:

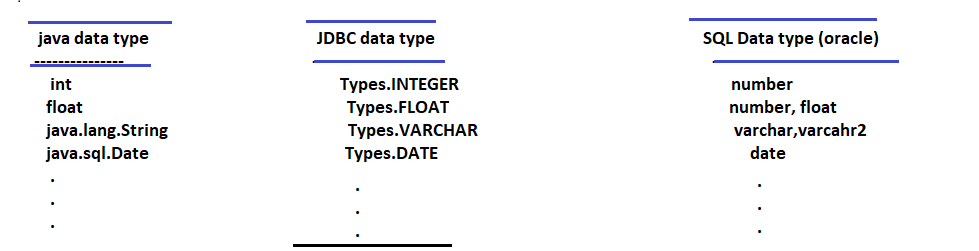
Parametername [IN/Out/INOUT] datatype;

Example:-



**Note:- Function must return a value But procedure does not return value.**

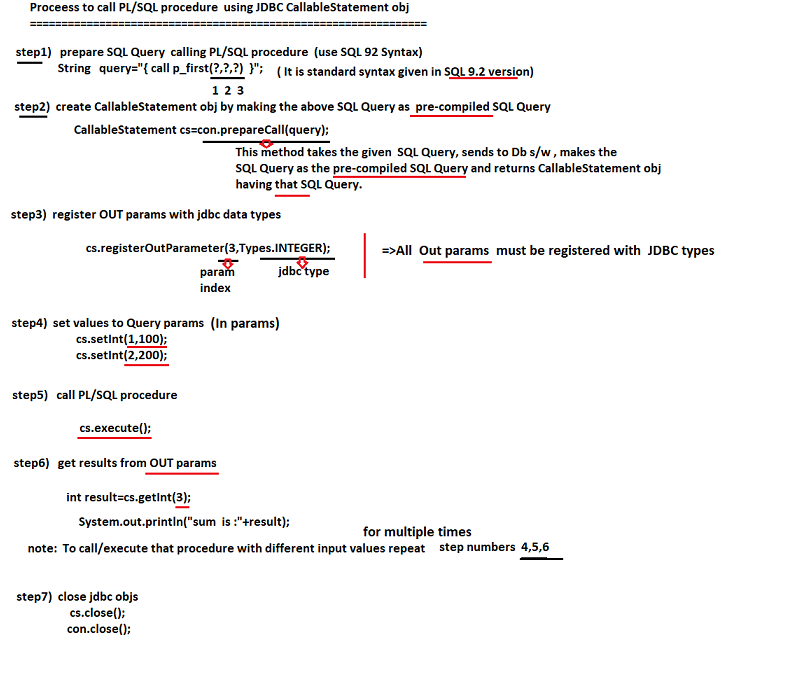
**2. JDBC Data Types:-** These are bridge data types between java DTs and under lying DB s/w Data types. These Data types are useful to guide JDBC driver s/w to convert java notation data to SQL notation data and vice versa. All JDBC Data types are integer constants in java.sql.Types class.



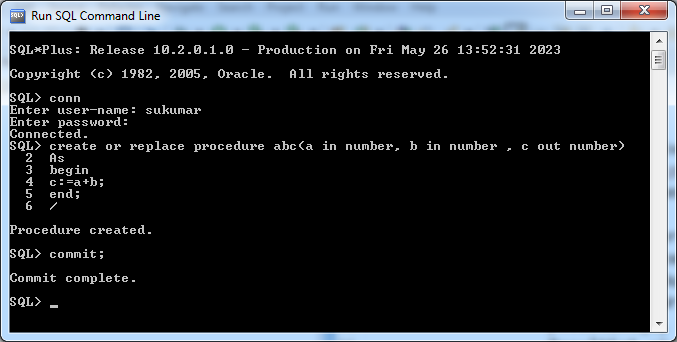


3.Callable Statement:- The SQL queries of procedure/function are precompiled sql queries resides forever in DB s/w. This Precompiled query is stored in Callable Statement object. We can execute the pre-compiled queries multiple-times from same applications, different applications using jdbc callable statement .

3.1. Callable Statement for invoking the procedure:



Example:



**package** com.raos.jdbc;

**import** java.io.IOException;

**import** java.sql.CallableStatement;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.SQLException;

**import** java.sql.Types;

**import** java.util.Scanner;

**public** **class** Sample {

**public** **static** String *query*="{call abc(?,?,?)}";

**public** **static** **void** main(String[] args) **throws** IOException{

**int** a,b,c;

**try**(Connection con=DriverManager.*getConnection*("jdbc:oracle:thin:@localhost:1521:xe","sukumar","sukumar");

CallableStatement ps=con.prepareCall(*query*);

Scanner s=**new** Scanner(System.***in***))

{

System.***out***.print("Enter the FirstNumber:");

a=s.nextInt();

System.***out***.print("Enter the second Number:");

b=s.nextInt();

ps.setInt(1, a);

ps.setInt(2, b);

ps.registerOutParameter(3, Types.***INTEGER***);

ps.execute();

System.***out***.println("Sum:"+ps.getInt(3));

}

**catch**(SQLException s) {

System.***out***.println(s.getMessage());

}

}//main

}//class

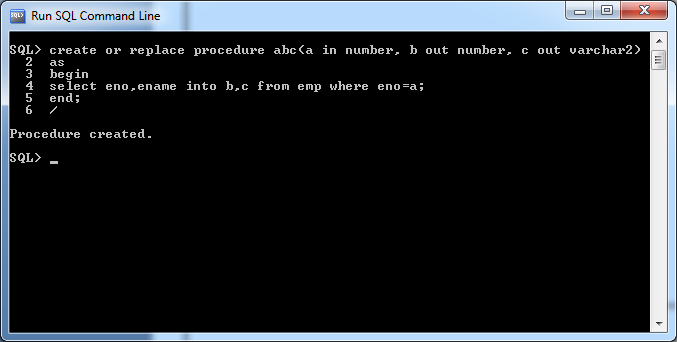
Output:-

Enter the FirstNumber:1

Enter the second Number:2

Sum:3

Example:2 Write JDBC application to retrieve specific employee record.



**package** com.raos.jdbc;

**import** java.sql.CallableStatement;

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.sql.Types;

**import** java.util.Scanner;

**public** **class** Sample {

**public** **static** String *qur*="{call abc(?,?,?)}";

**public** **static** **void** main(String[] args) {

**try**(Scanner s=**new** Scanner(System.***in***);

Connection con=DriverManager.*getConnection*("jdbc:oracle:thin:@localhost:1521:xe","sukumar","sukumar");

CallableStatement c=con.prepareCall(*qur*)

){

**int** eno=0;

**if**(s!=**null**) {

System.***out***.print("Enter eno:");

eno=s.nextInt();

}

**if**(c!=**null**) {

c.setInt(1, eno);

c.registerOutParameter(2, Types.***INTEGER***);

c.registerOutParameter(3, Types.***VARCHAR***);

c.execute();

System.***out***.println("Eno:"+c.getInt(2)+" "+"Ename:"+c.getString(3));

}

}

**catch**(SQLException e) {

**if**(e.getErrorCode()==1403) {

System.***out***.println("Invalid Employee Number");

}

}

}//main

}//class

Run-1:

Enter eno:1

Eno:1 Ename:suku

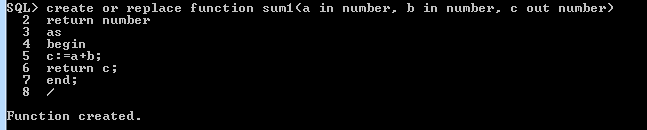
Run-2:

Enter eno:7

Invalid Employee Number.

3.2. Callable Statement for invoking Function:-

Example:-



import java.sql.CallableStatement;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Types;

import java.util.Scanner;

import oracle.jdbc.OracleTypes;

public class Sample {

public static String qur="{?=call sum1(?,?,?)}";

public static void main(String[] args) {

try(Scanner s=new Scanner(System.in);

Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","sukumar","sukumar");

CallableStatement c=con.prepareCall(qur)

){

int a,b;

System.out.print("Enter first Number:");

a=s.nextInt();

System.out.print("Enter the Second Number:");

b=s.nextInt();

if(c!=null) {

c.registerOutParameter(1, Types.INTEGER);

c.registerOutParameter(4, Types.INTEGER);

c.setInt(2, a);

c.setInt(3,b);

c.execute();

System.out.println("Sum:"+c.getInt(1));

}

}

catch(SQLException e) {

if(e.getErrorCode()==1403) {

System.out.println("Invalid Account Number");

}

}

}//main

}//class

Output:

----------

Enter first Number:10

Enter the Second Number:20

Sum:30