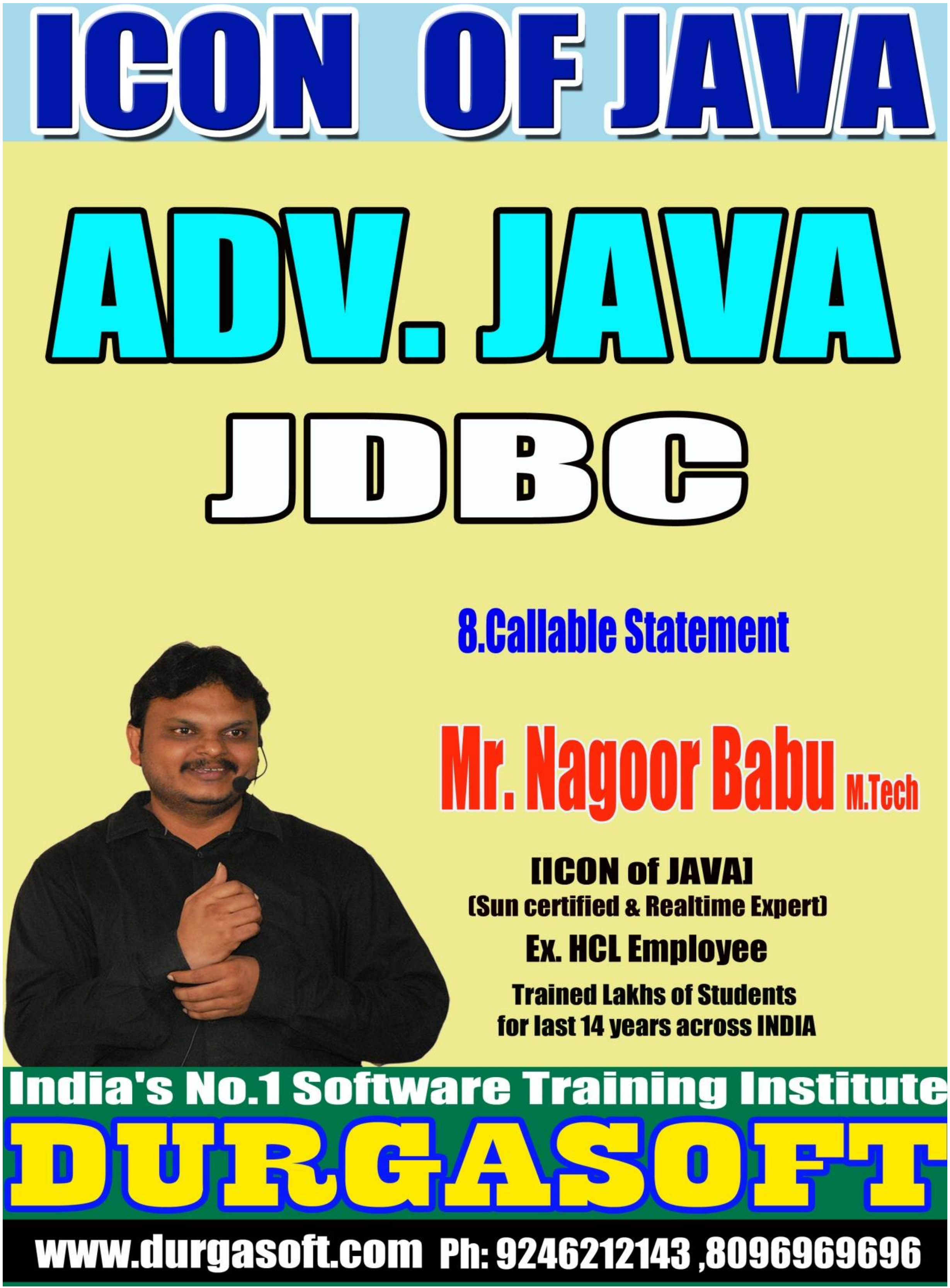


ICON OF JAVA

ADV. JAVA

JDBC

8.Callable Statement

Mr. Nagoor Babu M.Tech

[ICON of JAVA]
(Sun certified & Realtime Expert)
Ex. HCL Employee
Trained Lakhs of Students
for last 14 years across INDIA

India's No.1 Software Training Institute

DURGASOFT

www.durgasoft.com Ph: 9246212143 ,8096969696

Callable Statement

1)CallableStatement with procedure

2)CallableStatement with function

3)CallableStatement with CURSOR Type Procedure

4)CallableStatement with CURSOR type function

Stored Procedures And Functions:

What is the difference between Stored procedures and functions?
Ans: Stored procedure is a block of instructions defined at database to represent a particular action. Stored procedures will not use return statement to return a value.

Syntax: create or replace procedure procedure_name([param-list])
as

------------ Global declarations

BEGIN

------------ Database logic

END procedure_name;
/ (press enter to save and compile the procedure)

Stored function is a block of instructions defined at database to represent a particular action.Stored functions will use return statement to return a value.

Syntax: create or replace function function_name([param-list]) return data_type
as

```
------------ Global declarations
------------
BEGIN
------------
------------ Database logic
------------
return value;
END function_name;
/ (press enter to save and compile the function)
```

www.durgasoftonlinetraining.com
Online Training
Pre Recorded Video
Classes Training
Corporate Training
Ph: +91-8885252627, 7207212427
+91-7207212428
USA Ph : 4433326786
E-mail : durgasoftonlinetraining@gmail.com

In Jdbc applications, to access stored procedures and functions defined at database from Java application then we have to use CallableStatement object.

To represent CallableStatement object Jdbc API has provided the interface in the form of java.sql. CallableStatement.

If we want to use CallableStatement object in Jdbc applications then we have to use the following steps.

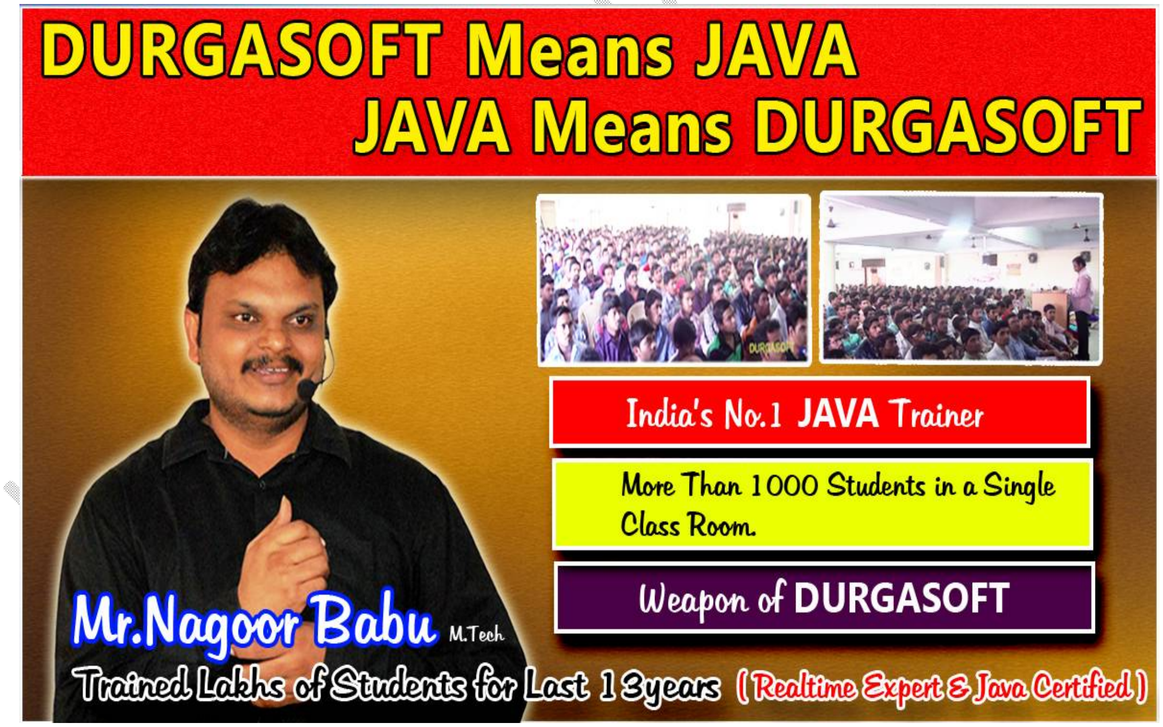
Step 1: Get CallableStatement object.

To create CallableStatement object in Jdbc applications we have to use the following method from Connection.

public CallableStatement prepareCall(String pro_cal)throws SQLException

Ex: CallableStatement cst=con.prepareCall("{call getSal(?,?)}");

When JVM encounters the above instruction JVM will pick up procedure call and send to Database Engine, where Database Engine will parse the procedure call and prepare a query plan with the positional parameters, as a result CallableStatement object will be created at Java application.


DURGASOFT Means JAVA
JAVA Means DURGASOFT
India's No.1 JAVA Trainer
More Than 1000 Students in a Single Class Room.
Weapon of DURGASOFT
Mr.Nagoor Babu M.Tech
Trained Lakhs of Students for Last 13years (Realtime Expert & Java Certified)

Note: In case of stored procedures and functions we are able to pass the parameters in the following 3 ways.

1. IN Type Parameter:

This parameter will get the value from procedure call or function call and make available to the procedure body or function body.

Syntax: var_name IN data_type

2. OUT Type Parameter:

This parameter will get the value from procedure body or function body and send that value to the respective procedure call or function call.

Syntax: var_name OUT data_type

3. INOUT Type Parameter:

This parameter is acting as both IN type and OUT type parameters.

Syntax: var_name INOUT data_type

Step 2: If we have IN type parameters in CallableStatement object then set values to IN type parameters.

To set values to IN type parameters we have to use the following method.

public void setXxx(int param_position, xxx value)

Where xxx may be byte, short, int and so on.

Ex: cst.setInt (1, 111);

Step 3: If we have OUT type parameter in CallableStatement object then we have to register OUT type parameter with a particular datatype.

To register OUT type parameter we will use the following method.

public void registerOutParameter(int param_position, int data_type)

Where data_type may be the constants from Types class like BYTE, SHORT, INTEGER, FLOAT and so on.

Ex: cst.registerOutParameter(2, Types.FLOAT);

Step 4: Make Database Engine to pick up the values from Query plan and to execute the respective procedure or function.

To achieve this we have to use The following method.

public void execute()throws SQLException

Ex: cst.execute();

Step 5: Get the values from OUT type parameters available in CallableStatement object.

After executing the respective procedure or function the respective values will be stored in OUT type parameters in CallableStatement object from stored procedure or functions. Toaccess the OUT type parameter values we have to use the following method.

public xxx getXxx(int param_position)

Where xxx may be byte, short, int and so on.

EX: float sal=cst.getFloat(2);

Ex:- execution of procedures

```
create or replace procedure getSal(id IN number, sal OUT number)
as
BEGIN
select esal into sal from emp where eno=id;
END getSal;
/
```

JdbcApp35:

```
import java.sql.*;
public class JdbcApp35{
public static void main(String[] args) throws Exception{
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection
con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","manager");
CallableStatement cst=con.prepareCall("{call getSal(?,?)}");
cst.setInt(1,101);
cst.registerOutParameter(2, Types.FLOAT);
cst.execute();
System.out.println("Salary........."+cst.getFloat(2));
con.close();
}
}
```

Ex:- execution of functions

```
create or replace function getAvg(id1 IN number, id2 IN number) return number
as
sal1 number;
sal2 number;
BEGIN
select esal into sal1 from emp where eno=id1;
select esal into sal2 from emp where eno=id2;
return (sal1+sal2)/2;
END getAvg;
/
```

JdbcApp36:

```
import java.sql.*;
```

```
public class JdbcApp36{
public static void main(String[] args) throws Exception{
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection
con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","system","manager");
CallableStatement cst=con.prepareCall("{?=call getAvg(?,?)}");
cst.setInt(2,888);
cst.setInt(3,6666);
cst.registerOutParameter(1, Types.FLOAT);
cst.execute();
System.out.println("Average Salary........"+cst.getFloat(1));
con.close();
}
}
```

JdbcApp37:

```
/*
create or replace procedure getEmps(sal IN number, emps OUT SYS_REFCURSOR)
 AS
 BEGIN
     open emps for
             select * from emp1 where esal<sal;
 END getEmps;
 /
 */
package com.durgasoft;

import java.io.FileInputStream;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.util.Properties;

import oracle.jdbc.internal.OracleTypes;

    public class JdbcApp37 {

    public static void main(String[] args)throws Exception {
            FileInputStream fis=new FileInputStream("db.properties");
            Properties p=new Properties();
            p.load(fis);
            String driver_class=p.getProperty("driver_class");
            String driver_url=p.getProperty("driver_url");
            Class.forName(driver_class);
```

```
                Connection con=DriverManager.getConnection(driver_url,p);
                CallableStatement cst=con.prepareCall("{call getEmps(?,?)}");
                cst.setFloat(1, 10000);
                cst.registerOutParameter(2, OracleTypes.CURSOR);
                cst.execute();
                Object obj=cst.getObject(2);
                ResultSet rs=(ResultSet)obj;
                System.out.println("ENO\tENAME\tESAL\tEADDR");
                System.out.println("----------------------------");
                while(rs.next()){

        System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getFloat(3)+"\t"+rs.getString(4));
                }
                con.close();
        }

}
```

db.properties

```
driver_class=oracle.jdbc.OracleDriver
driver_url=jdbc:oracle:thin:@localhost:1521:xe
user=system
password=durga
```

JdbcApp38:

```
/*
create or replace function getEmployees(no1 IN number,no2 IN number) return SYS_REFCURSOR
 AS
employees SYS_REFCURSOR;
 BEGIN
open employees for
      select * from emp1 where eno>=no1 and eno<=no2;
return employees;
 END getEmployees;
 /
 */
package com.durgasoft;

import java.io.FileInputStream;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.util.Properties;

import oracle.jdbc.internal.OracleTypes;

public class JdbcApp38 {

      public static void main(String[] args)throws Exception {
              FileInputStream fis=new FileInputStream("db.properties");
              Properties p=new Properties();
              p.load(fis);
              String driver_class=p.getProperty("driver_class");
              String driver_url=p.getProperty("driver_url");
```

```
                Class.forName(driver_class);
                Connection con=DriverManager.getConnection(driver_url, p);
                CallableStatement cst=con.prepareCall("{?=call getEmployees(?,?)}");
                cst.setInt(2, 111);
                cst.setInt(3, 555);
                cst.registerOutParameter(1, OracleTypes.CURSOR);
                cst.execute();
                ResultSet rs=(ResultSet)cst.getObject(1);
                System.out.println("ENO\tENAME\tESAL\tEADDR");
                System.out.println("------------------------");
                while(rs.next()){

        System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getFloat(3)+"\t"+rs.getString(4));
                }
                con.close();
        }
}
```

db.properties

```
driver_class=oracle.jdbc.OracleDriver
driver_url=jdbc:oracle:thin:@localhost:1521:xe
user=system
password=durga
```

www.durgasoftonlinetraining.com
Online Training
Pre Recorded Video
Classes Training
Corporate Training
Ph: +91-8885252627, 7207212427
+91-7207212428
USA Ph : 4433326786
E-mail : durgasoftonlinetraining@gmail.com