JSP Assignment

Statement

Use this for general-purpose - access to your database. Useful when you are using static SQL statements at runtime. The statement interface cannot accept parameters.

Prepared Statement

Use this when you plan to use the SQL statement many times. The prepared statement interface accepts input parameters at runtime.

Callable Statement

Use this when you want to access the database stored procedure. The callable statement interface can also accept runtime input parameters.

JSP - Assignment

The Statement objects

Creation

```
Statement stmt = null;
try {
    stmt = conn.createStatement();
    :
} catch (SQLException e) {
    ....
}
finally {
    :
}
```

There are 3 execution method.

boolean execute (String SQL) :- Returns a boolean value of true if a ResultSet object can be retrieved.

int executeUpdate (String SQL) :- Returns the number of rows affected by the execution of the SQL statement.

ResultSet executeQuery :- Returns a ResultSet object.

closing :-

You can close by using close() method.

```
Statement stmt = null;
try {
    stmt = conn.createStatement();
    :
} catch (SQLException e) {
    :
} finally {
    stmt.close();
}
```

The Prepared statement objet
creation

```
prepared statent pstmt = null;
try {
    string SQL = "Update Employees SET age=? WHERE id=?";
    pstmt = conn.preparestatemt(SQL);

}

catch (SQL Exception e) {
    :
}

finally {
    ...
}

closing
Prepared statemt pstmt = null;
try {
    string SQL = "Update Employees SET age=? WHERE id=?";
    pstmt = conn.prepare Statemt(SQL);
    :
} catch (SQLException e) {
    :
}
finally {
    pstmt.close();
}
```

Callable Statement objects.

Just as a connection object creates the statement and prepared statement objects, it also creates the callable statement object, which would be used to create a call to a database stored procedure.

creation

~~create~~ CREATE OR REPLACE PROCEDURE getEmpName
(EMP_ID IN NUMBER, EMP_FIRST OUT VARCHAR) AS
BEGIN
   SELECT first INTO EMP_FIRST
   FROM employees
   WHERE ID = EMP_ID;

For MySQL

DELIMITER $$
DROP PROCEDURE IF EXISTS `EMP`.`getEmpName` $$
CRETE PROCEDURE `EMP`.`getEmpName`
(IN EMP_ID INT, OUT EMP_FIRST VARCHAR(255))
BEGIN
   SELECT first INTO EMP_FIRST
   FROM employees
   WHERE ID = EMP_ID;
END $$
DELIMITER ;

There are 3 types of parameters exists
IN, OUT, INOUT.

Snipet

```
Callable Statement cstmt= null;
try {
    String SQL = "{ call getEmp Name (?,?)}";
    cstmt = conn.prepareCall (SQL);
    ....
} catch (SQLExeftion e) {
    ...
}
finally {
    ...
}
```

Closing Callable Statement

We can close callable statement by just
calling close() method.

```
Callable Statement cstmt = null;
try {
    String SQL = "{ call getEmpName(?,?)}";
    cstmt = conn.prepareCall(SQL);
    ...
}
```