

## 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 methods.

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();  
}
```

(Continuation in next book)



## The Prepared statement object creation

```
preparedStatement pstmt = null;  
try {
```

String SQL = "Update Employees SET age=? WHERE id=?";

```
psstmt = conn.prepareStatement(SQL);
```

catch (SQLException e) {

9  
Finally to

2

closing

Prepared statement ps/mnt = null;

try to

19. a

```
String SQL = "Update Employees SET age=? WHERE id=?";  
psmt = conn.prepareStatement(SQL);
```

```
psmt = conn.prepareStatement(SQL);
```

9. catch (SQLException e) {

finally 2

```
postmid.close();
```

Teacher's Signature \_\_\_\_\_



## Callable Statement Objects.

Just as a connection object creates the statements 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)
BEGIN
    SELECT first INTO EMP_FIRST
    FROM employees
    WHERE ID = EMP_ID;
```

### For MySQL

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