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 Procedures. The Callable Statement interface can also accept runtime input parameters.

JSP - Assignment

The Statmat objects

Creation

```
Statemat Stmt = null;
try{
  Stmt = Conn.creatiStatmet();
  :
}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 executiUpdate(String SQL) :- Returns the number of rows affected by the execution of the SQL Statment.

ResultSet executi Query :- Returns a ResultSet object.

closing :- You can close by using close() method.

```
Statemet Stmt = null;
try{
  Stmt = conn.creatiStatement();
  :
}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 = ?";
    Pstmt = conn.prepareStatement (SQL);
    :
}
catch(SQL Exception e){
    :
}
finally{
    :
}
```

closing

```
Prepared Statement pstmt = null;
try{
    String SQL = "Update Employees SET age = ? WHERE id = ?
    pstmt = conn.prepareStatement(SQL);
    :
}catch(SQLException e){
    :
}
finally{
    pstmt.close();
}
```

Teacher's Signature

Callable Statement objects.

Just as a Connection object creates the Statement and PreparedStatement objects, it also creates the callable statement object, which would be used to execute a call to a database stored procedure.

creation

```
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 $$
DROP PROCEDURE IF EXISTS `EMP`.`getEmpName`$$
CREATE PROCEDURE `EMP`.`getEmpName`
  (IN EMP_ID INT, OUT EMP_FIRST VARCHAR(25))
  BEGIN
    SELECT first INTO EMP_FIRST
    FROM Employees
    WHERE ID = EMP_ID;
END$$
DELIMITER ;
```

Teacher's Signature

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

Snipet

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

Closing Callable Statement

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

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

Teacher's Signature

```
catch (SQLException e) {
    ...
}
finally {
    cstmt.close();
}
```