Module 8 - Java
Working with Databases

Advanced Java Certification Training

Akram M'Tir

1. 1. Write a programme to get the student's name for a given student's ID from the test.student table.
   a. Use stored procedure (script provided in scripts/get_name.sql).

Database, Table

```
MariaDB [test]> describe STUDENT;
+-------+--------------+------+-----+---------+----------------+
| Field | Type         | Null | Key | Default | Extra          |
+-------+--------------+------+-----+---------+----------------+
| ID    | int(11)      | NO   | PRI | NULL    | auto_increment |
| NAME  | varchar(255) | YES  |     | NULL    |                |
| MARKS | int(11)      | YES  |     | NULL    |                |
+-------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

MariaDB [test]> select * from STUDENT;
+----+-------+-------+
| ID | NAME  | MARKS |
+----+-------+-------+
|  1 | Tom   |    95 |
|  2 | John  |    75 |
|  3 | Alice |    90 |
|  4 | Bob   |    98 |
```

Procedure Creation and query

```
MariaDB [test]> DROP PROCEDURE IF EXISTS get_name;
Query OK, 0 rows affected (0.00 sec)

MariaDB [test]> delimiter //
MariaDB [test]> CREATE PROCEDURE `get_name` (IN id INT, OUT name VARCHAR(20))
    ->     BEGIN
    ->       SELECT STUDENT.NAME INTO name FROM STUDENT WHERE STUDENT.ID=id;
    ->     END//
Query OK, 0 rows affected (0.00 sec)

MariaDB [test]> delimiter ;
MariaDB [test]>
MariaDB [test]> call get_name(3 , @n);
Query OK, 1 row affected (0.00 sec)

MariaDB [test]>
MariaDB [test]>
MariaDB [test]> select @n;
+-------+
| @n    |
+-------+
| Alice |
+-------+
1 row in set (0.00 sec)
```

Java Code

OutParamDemo.java | TestConnectionMySQL.java | dbconn.properties | log4j.properties | Crea

```java
package module8.proc;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.sql.CallableStatement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;
import org.apache.log4j.Logger;

public class OutParamDemo {

    private static final Logger log4j = Logger.getLogger(OutParamDemo.class.getName());
    private static final String DB_CONN_FILE = "resources/dbconn.properties";

    public static void main(String[] args) {
        Properties p = new Properties();

        try {
            p.load(new FileInputStream(new File(DB_CONN_FILE)));

        } catch (IOException e) {
            log4j.error("Error reading " + DB_CONN_FILE, e);
            System.exit(-1);
        }

        log4j.debug("Connecting to database.");

        try (Connection conn = DriverManager.getConnection(p.getProperty("url"), p);) {
            log4j.debug("Connecting succeeded.");

            String sqlStat = "CALL get_name(?, ?)";
            int ID = 2;
            try (CallableStatement proc = conn.prepareCall(sqlStat);) {
                proc.setInt(1, ID);
                proc.execute();

                String str = null;
                str = proc.getString(2);

                log4j.debug("ID : " + ID + " --> Name  " + str + " .");
            }
        } catch (SQLException e) {
            log4j.error("Query failed.", e);
        } finally {
            log4j.debug("Connection closed.");
        }
    }
}
```

Console Log4j output

Problems | Console | Javadoc | Declaration | Terminal | Debug

```
<terminated> OutParamDemo [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/j
2018-05-25 02:56:17 main       DEBUG module8.proc.OutParamDemo:34 - Connecting to database.
2018-05-25 02:56:18 main       DEBUG module8.proc.OutParamDemo:37 - Connecting succeeded.
2018-05-25 02:56:18 main       DEBUG module8.proc.OutParamDemo:50 - ID : 2 --> Name  John .
2018-05-25 02:56:18 main       DEBUG module8.proc.OutParamDemo:57 - Connection closed.
```

2. Write a programme that does a wildcard search of student's name from the test.student table .

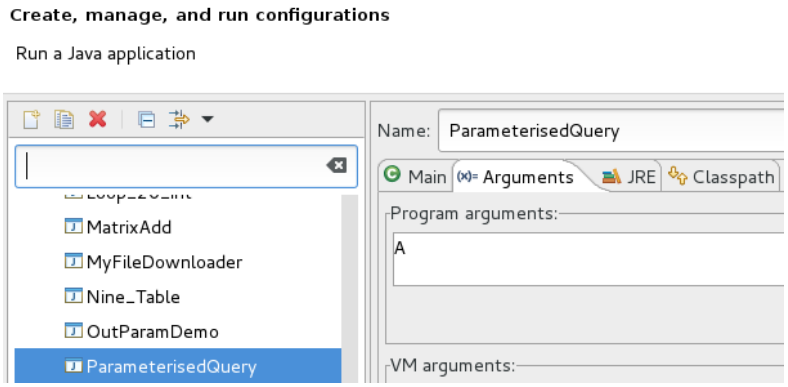
a. The search criteria should be parameterized.
b. Display the matched student's id and name.
c. Handle the appropriate errors.

Database, Table

```
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

MariaDB [test]>
MariaDB [test]>
MariaDB [test]> select * from STUDENT;
+----+-------+-------+
| ID | NAME  | MARKS |
+----+-------+-------+
|  1 | Tom   |    95 |
|  2 | John  |    75 |
|  3 | Alice |    90 |
|  4 | Bob   |    98 |
|  5 | Anna  |    96 |
|  6 | Alex  |    77 |
|  7 | Aaron |    82 |
|  8 | Amy   |    93 |
+----+-------+-------+
8 rows in set (0.00 sec)
```

Input Arguments used for the search



Console Log4j Output

```
Problems  Console  Javadoc  Declaration  Terminal  Debug
<terminated> ParameterisedQuery [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/ja
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:38 - Connecting to database.
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:41 - Connecting succeeded.
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:50 - 3 Alice 90
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:50 - 5 Anna 96
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:50 - 6 Alex 77
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:50 - 7 Aaron 82
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:50 - 8 Amy 93
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:52 - Fetched 5 rows.
2018-05-25 03:25:05 main       DEBUG module8.query.ReadingTableDemo:58 - Connection closed.
```

Java Code

ParameterisedQuery.java | OutParamDemo.java | TestConnectionMySQL.java

```java
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.Properties;

import org.apache.log4j.Logger;

public class ParameterisedQuery {

    private static final Logger log4j = Logger.getLogger(ReadingTableDemo.class.getName());
    private static final String DB_CONN_FILE = "resources/dbconn.properties";

    public static void main(String[] args) {

        if (args.length != 1) {
            System.out.println("Usage: java ParameterisedQuery One_Character");
            System.exit(-1);
        }
        String paramSearch = args[0];

        Properties p = new Properties();

        try {
            p.load(new FileInputStream(new File(DB_CONN_FILE)));

        } catch (IOException e) {
            log4j.error("Error reading " + DB_CONN_FILE, e);
            System.exit(-1);
        }

        log4j.debug("Connecting to database.");

        try (Connection conn = DriverManager.getConnection(p.getProperty("url"), p);) {
            log4j.debug("Connecting succeeded.");

            String sqlStat = "select * from STUDENT WHERE NAME like '" + paramSearch + "%'";
            Statement stmt = conn.createStatement();

            try (ResultSet set = stmt.executeQuery(sqlStat);) {
                int counter = 0;
                while (set.next()) {
                    counter++;
                    log4j.debug(set.getString("ID") + " " + set.getString("NAME") + " " + set.getString("MARKS"));
                }
                log4j.debug("Fetched " + counter + " rows.");
            }

        } catch (SQLException e) {
            log4j.error("Query failed.", e);
        } finally {
            log4j.debug("Connection closed.");
        }

    }

}
```

3. Write a programme to batch process insert and update statements to the test.student table.

a. Handle the appropriate errors.

b. Use save-points to define boundaries of transactions.

Batch Insert

```
MariaDB [test]> select * from STUDENT;
+----+-------+-------+
| ID | NAME  | MARKS |
+----+-------+-------+
|  1 | Tom   |    95 |
|  2 | John  |    75 |
|  3 | Alice |    90 |
|  4 | Bob   |    98 |
|  5 | Anna  |    96 |
|  6 | Alex  |    77 |
|  7 | Aaron |    82 |
|  8 | Amy   |    93 |
+----+-------+-------+
8 rows in set (0.00 sec)

MariaDB [test]> select * from STUDENT;
+----+--------+-------+
| ID | NAME   | MARKS |
+----+--------+-------+
|  1 | Tom    |    95 |
|  2 | John   |    75 |
|  3 | Alice  |    90 |
|  4 | Bob    |    98 |
|  5 | Anna   |    96 |
|  6 | Alex   |    77 |
|  7 | Aaron  |    82 |
|  8 | Amy    |    93 |
|  9 | Bella  |    65 |
| 10 | Bailly |    75 |
+----+--------+-------+
10 rows in set (0.00 sec)
```

```
Problems  Console  Javadoc  Declaration  Terminal  Debug
<terminated> FixRecords [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 25, 2018, 4:02:02 AM)
2018-05-25 04:02:03 main       DEBUG module8.batch.FixRecords:34 - Connecting to database.
2018-05-25 04:02:03 main       DEBUG module8.batch.FixRecords:37 - Connecting succeeded.
2018-05-25 04:02:03 main       DEBUG module8.batch.FixRecords:56 - Executing batch.
2018-05-25 04:02:03 main       DEBUG module8.batch.FixRecords:60 - Inserted  2 record(s).
2018-05-25 04:02:03 main       DEBUG module8.batch.FixRecords:63 - Commited transaction.
2018-05-25 04:02:03 main       DEBUG module8.batch.FixRecords:73 - Connection closed.
```

Java Code

```java
import java.sql.Savepoint;
import java.util.Properties;

import org.apache.log4j.Logger;

public class FixRecords {

    private static final Logger log4j = Logger.getLogger(FixRecords.class.getName());
    private static final String DB_CONN_FILE = "resources/dbconn.properties";

    public static void main(String[] args) {
        Properties p = new Properties();

        try {
            p.load(new FileInputStream(new File(DB_CONN_FILE)));

        } catch (IOException e) {
            log4j.error("Error reading " + DB_CONN_FILE, e);
            System.exit(-1);
        }

        log4j.debug("Connecting to database.");

        try (Connection conn = DriverManager.getConnection(p.getProperty("url"), p);) {
            log4j.debug("Connecting succeeded.");

            Savepoint marker = null;
            conn.setAutoCommit(false);

            String sqlStat = "INSERT INTO STUDENT (NAME, MARKS) VALUE(?, ?)"; // PK AutoIncrement used

            try (PreparedStatement stmt = conn.prepareStatement(sqlStat);) {
                // Start transaction
                marker = conn.setSavepoint("start_trans");
                stmt.setString(1, "Bella");
                stmt.setInt(2, 65);
                stmt.addBatch();

                stmt.setString(1, "Bailly");
                stmt.setInt(2, 75);
                stmt.addBatch();

                log4j.debug("Executing batch.");

                int[] r = stmt.executeBatch();

                log4j.debug("Inserted  " + r.length + " record(s).");

                conn.commit();
                log4j.debug("Commited transaction.");
                // Ending transaction
                conn.releaseSavepoint(marker);
                conn.setAutoCommit(true);
            }

        } catch (SQLException e) {
            log4j.error("Batch processing failed.", e);
        } finally {
            log4j.debug("Connection closed.");
```

Update records Batch

```
File  Edit  View  Search  Terminal  Help
MariaDB [test]> select * from STUDENT;
+----+--------+-------+
| ID | NAME   | MARKS |
+----+--------+-------+
|  1 | Tom    |    95 |
|  2 | John   |    75 |
|  3 | Alice  |    90 |
|  4 | Bob    |    98 |
|  5 | Anna   |    96 |
|  6 | Alex   |    77 |
|  7 | Aaron  |    82 |
|  8 | Amy    |    93 |
|  9 | Bella  |    65 |
| 10 | Bailly |    75 |
+----+--------+-------+
10 rows in set (0.00 sec)

MariaDB [test]> select * from STUDENT;
+----+--------+-------+
| ID | NAME   | MARKS |
+----+--------+-------+
|  1 | Tom    |    95 |
|  2 | John   |    75 |
|  3 | Alice  |    90 |
|  4 | Bob    |    98 |
|  5 | Anna   |    96 |
|  6 | Alex   |    77 |
|  7 | Aaron  |    82 |
|  8 | Amy    |    93 |
|  9 | Bella  |    33 |
| 10 | Bailly |    66 |
+----+--------+-------+
10 rows in set (0.00 sec)
```

console Log4j output

```
Problems  Console  Javadoc  Declaration  Terminal  Debug
<terminated> FixRecords [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 25, 2018, 4:14:43 AM)
2018-05-25 04:14:44 main      DEBUG module8.batch.FixRecords:31 - Connecting to database.
2018-05-25 04:14:44 main      DEBUG module8.batch.FixRecords:34 - Connecting succeeded.
2018-05-25 04:14:44 main      DEBUG module8.batch.FixRecords:53 - Executing batch.
2018-05-25 04:14:44 main      DEBUG module8.batch.FixRecords:57 - Inserted/updated  2 record(s).
2018-05-25 04:14:44 main      DEBUG module8.batch.FixRecords:60 - Commited transaction.
2018-05-25 04:14:44 main      DEBUG module8.batch.FixRecords:69 - Connection closed.
```

```java
    private static final Logger log4j = Logger.getLogger(FixRecords.class.getName());
    private static final String DB_CONN_FILE = "resources/dbconn.properties";

    public static void main(String[] args) {
        Properties p = new Properties();

        try {
            p.load(new FileInputStream(new File(DB_CONN_FILE)));

        } catch (IOException e) {
            log4j.error("Error reading " + DB_CONN_FILE, e);
            System.exit(-1);
        }

        log4j.debug("Connecting to database.");

        try (Connection conn = DriverManager.getConnection(p.getProperty("url"), p);) {
            log4j.debug("Connecting succeeded.");

            Savepoint marker = null;
            conn.setAutoCommit(false);

            //String sqlStat = "INSERT INTO STUDENT (NAME, MARKS) VALUE(?, ?)"; // PK AutoIncrement us
            String sqlStat = "UPDATE STUDENT SET MARKS=?  WHERE ID=?";

            try (PreparedStatement stmt = conn.prepareStatement(sqlStat);) {
                // Start transaction
                marker = conn.setSavepoint("start_trans");
                stmt.setInt(1, 66);
                stmt.setInt(2, 10);
                stmt.addBatch();

                stmt.setInt(1, 33);
                stmt.setInt(2, 9);
                stmt.addBatch();

                log4j.debug("Executing batch.");

                int[] r = stmt.executeBatch();

                log4j.debug("Inserted/updated  " + r.length + " record(s).");

                conn.commit();
                log4j.debug("Commited transaction.");
                // Ending transaction
                conn.releaseSavepoint(marker);
                conn.setAutoCommit(true);
            }

        } catch (SQLException e) {
            log4j.error("Batch processing failed.", e);
        } finally {
            log4j.debug("Connection closed.");
        }

    }

}
```