Session 1

Statement, PreparedStatement y Connections Pool

Aims and learning outcomes

The objectives of this exercise are:

- To review key JDBC concepts.
- To compare the use of Statement and PreparedStatement.
- To establish connections with different DBMS, such as Oracle and HSQLdb.
- To utilize a Connection Pool and compare performance times.

Getting started

- Download the material from the Virtual Campus:
  - An HSQLDB database.
  - A script to create a similar database in Oracle.
  - JDBC drivers to connect to both DBMS.
  - A driver to handle connection pools (C3P0).
- Create a workspace directory and boot the Eclipse IDE.
- Start the HSQLdb server (run data/startup script).
- Connect to the Oracle server. Run the script to create CarWorkshop database in Oracle.

Connection information

- **Oracle**
  - URL = "jdbc:oracle:thin:@156.35.94.98:1521:DESA19"; //(ports 1521 or 5850)
  - USER = your UO identifier
  - PASS = your password
- **HSQLDB**
  - URL = "jdbc:hsqldb:hsql://localhost";
  - USER = "sa";
  - PASS = "";

Exercises

Create projects to solve exercises. Do not forget to configure build path to link libraries (drivers and connection pool manager as needed).

To delete tables from last year (if it was necessary) use:

```
SELECT 'DROP TABLE '||table_name||' CASCADE CONSTRAINTS;' FROM user_tables
```

Then execute the result of that command to remove all the tables.

Exercise 1: Statement vs. PreparedStatement time comparison

Write a java program to compare performance between statement and preparedStatement. For this purpose, your program must connect to the database and run a loop to iterate one thousand times executing a query every time, using a Statement object. Finally, you must release the resources. Then, repeat the code with PreparedStatement.

This program must measure how long it takes to execute a block of code. To get a significant difference, a complex query must be written, specifically one that changes in each iteration of the loop. For example, update table TINVOICES increasing amount by one, when the number of the invoice matches the value of the control variable of the loop.

In case of Statement, write the SQL statement by joining strings, for example. In the case of PreparedStatement, use placeholders ("?").

Use System.currentTimeMillis() to take time; it returns date/time from 1970/01/01 in milliseconds:

1. Run the program connecting to HSQLDB first.
2. Then, run again connecting to Oracle this time.
3. As this second execution will be a bit disappointing, mainly due to the network delay, implement a third version using batch processing (link).

In each execution, compare both times. Which takes longer? Is this the expected output?

You can see a very interesting discussion about what has been worked on in this session here.

Exercise 2: Check the cost of opening and close connections

A database connection is a large and complex object that handles all communication between an application and the underlying database. As such, database connections are time consuming to establish and consume a significant amount of memory on both, the client application and the database server.

In this exercise, you must write a program to get a better idea of the cost of connecting to a database. To do this, we will proceed in two different ways, measuring times and comparing them:

1. Write a loop to iterate one hundred times. Each iteration should open a connection to the database, do a query and close he connection.

2. Repeat the exercise but now, open and close connection outside the loop.

Exercise 3: Connections pool

For this exercise, use c3p0 connections pool whose drivers can be found in lib folder downloaded from Virtual Campus or you can download them directly from here.
The idea is to create a pool of connections with the following properties:

- Max number of connections (maxPoolSize): 30
- Minimum number of connections(minPoolSize): 3
- Initial pool size(initialPoolSize): 3
-

To check connection pool advantages, you must write a program as in the previous exercise.

Write a loop to iterate one thousand times. In each iteration, the program must connect to database in the usual way, do an operation and close connection. Later, you must repeat the code but using a connection from the pool.

The operation can be counting the number of vehicles in TVEHICLES.

```
for (int i=0; i<1000; i++){
    doConnect();
    doSomething();
    doDisconnect();
}
```

As an example of c3p0 using, a piece of code is shown below; it is available in c3p0 website (http://www.mchange.com/projects/c3p0/#using_combopooleddatasource). This example comes together with the driver itself.
The code sets a Pooled DataSource.

```
DataSource ds_unpooled =
DataSources.unpooledDataSource("jdbc:postgresql://localhost/testdb",
                                                    "swaldman",
                                                "test-password");
Map overrides = new HashMap();
overrides.put("minPoolSize", 3);
overrides.put("maxPoolSize", 50);
overrides.put("initialPoolSize", 3);
ds_pooled = DataSources.pooledDataSource(ds_unpooled, overrides );
// it is already set
// Now, get a connection
con = ds_pooled.getConnection();
```

1. Run the exercise with both Oracle and HSQLdb. Compare how much time it takes with and without a pool of connections
2. Finally, write a new version of this same exercise but split the task into 10 different threads so each one executes the loop 10 times. Run the exercise with both Oracle and HSQLdb. Compare how much time it takes with and without a pool of connections.