Figure 1: datamock

4315 Project

Skyler Layne, 212166906

This is an informal document on my software datamock, for more detailed documentation please view the code it is freely available (MIT) here https://www.github.com/skylerto/datamock

For more complicated examples, you can view the API http://skylerlayne.me/datamock/

Datamock

Datamock is a database mocking extension for JPF.

It attempts to model a real database in memory. It works well for databases which are created, read, updated, and destroyed from the same application. This is because it's implemented entirely with data structures.

This is done because of some issues such as relational algebra when faced with implementing native peers which reach into the JVM and execute native methods on the database.

There's more discussion on this written in a separate paper, please see the resources section for more information.

Usage

Please ensure you have a working copy of JPF and a MySQL server running on your machine.

For a simple example, unzip the code, enter the code directory. In this directory there is another called scripts:

Please execute the scripts from the project root, e.g. `./scripts/run-jpf`.

```
scripts
- run-java
- run-jpf
- run-spec
- setup

0 directories, 4 files
```

The most important of these to run first is the setup, if you are planning to run the code through mysql database.

run-java

This will compile and execute the code using java with MySQL jdbc adaptor, available in lib/

run-jpf

This script will compile the code for JPF and run it through jpf with the project.jpf file.

run-spec

This script will compile the code for java and MySQL then execute the, however small, test suite.

setup

This script can be used to setup the database on your computer, runs the setup.sql file.

Example Code/Run on your own

Create the following `project.jpf` file

```
target=App
classpath=bin/;bin/model-classes/;
```

A small example can be found enclosed in the below as well as in the zip that came with this software.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

/**
 * A simple example of the usage of DataMock.
 *
 * @author Skyler Layne (c) All Rights Reserved.
 *
 */
public class App {

  /**
   * Main.
   *
   * @param args
   *           - sys args.
   */
  public static void main(String[] args) {

    Connection con = null;
    Statement stmt;

    String database = "mission";
    String host = "localhost";
    String username = "root";
    String password = "";

    try {
      Class.forName("com.mysql.jdbc.Driver").newInstance();

      con = DriverManager.getConnection("jdbc:mysql://" + host + "/" + database +
          "?useSSL=false",
          username, password);

      stmt = con.createStatement();

      /* Create table a table. */
      stmt.execute("drop table if exists students");
      stmt.execute("create table students (name VARCHAR(20), id INT, PRIMARY KEY (id))");

      /* Retrieve Data the data. */
      ResultSet rs = stmt.executeQuery("select * from students");
      while (rs.next()) {
```

```
      System.out.println(rs.getString("name") + ", " + rs.getInt("id"));
    }

    /* Insert Data Into the Database. */
    System.out.println("Do some Inserting...");
    stmt.executeUpdate("insert into students (name, id) values ('Skyler Layne',
        212166906)");
    stmt.executeUpdate("insert into students (name, id) values ('Jane Smith',
        212166908)");
    stmt.executeUpdate("insert into students (name, id) values ('John Doe',
        212166907)");

    /* Retrieve Data the data. */
    rs = stmt.executeQuery("select * from students");
    while (rs.next()) {
      System.out.println(rs.getString("name") + ", " + rs.getInt("id"));
    }

    /* Remove from database. */
    System.out.println("Do a Delete...");
    stmt.executeUpdate("delete from students where name='Jane Smith'");

    /* Retrieve Data the data. */
    rs = stmt.executeQuery("select * from students");
    while (rs.next()) {
      System.out.println(rs.getString("name") + ", " + rs.getInt("id"));
    }

    /* Drop table. */
    System.out.println("Do a Table Drop...");
    stmt.executeUpdate("drop table students");

    /* Retrieve Data the data, should throw an SQLException because we've deleted the
        table. */
    rs = stmt.executeQuery("select * from students");
    while (rs.next()) {
      System.out.println(rs.getString("name") + ", " + rs.getInt("id"));
    }

    /* Shut it down. */
    rs.close();
    stmt.close();
    con.close();

  } catch (InstantiationException | IllegalAccessException | ClassNotFoundException
      | SQLException e) {
    e.printStackTrace();
  }
 }
}
```

References

[1] *Model Checking Database Applications* by Milos Gligoric and Rupak Majumdar.