

Replicated Databases

Source Code:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
import java.util.Properties;
import java.util.ArrayList;
import java.util.concurrent.Semaphore;

public class JdbcMTSample extends Thread
{
    // Set default number of threads to 10
    private static int NUM_OF_THREADS = 10;

    int m_myId;

    static    int c_nextId = 1;
    static Connection s_conn = null;
    static Semaphore semaphore = new Semaphore(1);

    synchronized static int getNextId()
    {
        return c_nextId++;
    }

    public static void main (String args [])
    {
        try
        {
            // Load the JDBC driver //
            DriverManager.registerDriver
                (new oracle.jdbc.driver.OracleDriver());
            Class.forName("oracle.jdbc.OracleDriver");
            // If NoOfThreads is specified, then read it
            if (args.length > 1) {
                System.out.println("Error: Invalid Syntax. ");
            }
        }
    }
}
```

```

        System.out.println("java JdbcMTSample [NoOfThreads]");
        System.exit(0);
    }
    else if (args.length == 1)

        NUM_OF_THREADS = Integer.parseInt (args[0]);

    // Create the threads
    Thread[] threadList = new Thread[NUM_OF_THREADS];

    // spawn threads
    for (int i = 0; i < NUM_OF_THREADS; i++)
    {
        threadList[i] = new JdbcMTSample();
        threadList[i].start();
    }

    // wait for all threads to end
    for (int i = 0; i < NUM_OF_THREADS; i++)
    {
        threadList[i].join();
    }

}
catch (Exception e)
{
    e.printStackTrace();
}
}

public JdbcMTSample()
{
    super();
    // Assign an ID to the thread
    m_myId = getNextId();
}

public void run()
{
    Connection conn1 = null;
    Connection conn2 = null;

    ResultSet    rs    = null;

```

```
ResultSet rs0 = null;
```

```
Statement stmt = null;
```

```
Statement stmt0 = null;
```

```
ArrayList<String> lastitem = new ArrayList<String>();
```

```
try
```

```
{
```

```
    semaphore.acquire(); // providing mutual exclusion
```

```
    // Get the connection
```

```
String dbURL1 = "jdbc:oracle:thin:system/oracle@192.168.56.101:1521:orclcdb";
```

```
String dbURL2 = "jdbc:oracle:thin:fabedin/04118552@oracle.scs.ryerson.ca:1521:orcl";
```

```
    conn1 = DriverManager.getConnection(dbURL1);
```

```
conn2 = DriverManager.getConnection(dbURL2);
```

```
    if (conn1 != null) {
```

```
        System.out.println("Connected with connection #1");
```

```
        conn1.setAutoCommit(false);
```

```
    }
```

```
    else{
```

```
        System.out.println("Connection Failed with connection #1");
```

```
        //transaction = false;
```

```
    }
```

```
if (conn2 != null) {
```

```
    System.out.println("Connected with connection #2");
```

```
    conn1.setAutoCommit(false);
```

```
    }
```

```
    else{
```

```
        System.out.println("Connection Failed with connection #2");
```

```
        //transaction = false;
```

```
    }
```

```
    //String nameToInsert = "test" + m_myId % 5 ; // generates not unique names
```

```
String nameToInsert = "test12";
```

```
String nameToDelete = "test2";
```

```
    // You need to create a table called TESTJ with one string attributes call Name by  
SqlDeveloper in Site1
```

```

// check starts from here
    stmt = conn1.createStatement ();
    stmt0 = conn2.createStatement ();

    // Execute the Query
    rs = stmt.executeQuery ("SELECT * FROM TESTJ");
    rs0 = stmt0.executeQuery ("SELECT * FROM TESTJ");
    // Loop through the results
    while (rs.next()){
        System.out.println("Thread " + m_myId +
            " Name : " + rs.getString("Name"));
        lastitem.add(rs.getString("Name"));
    }

    while (rs0.next()){
        System.out.println("Thread " + m_myId +
            " Name : " + rs0.getString("Name"));
        lastitem.add(rs0.getString("Name"));
    }

    // Close all the resources
    rs.close();
    rs0.close();
    stmt.close();
    stmt0.close();
// check ends here

    // Create a Statement

    if (!lastitem.contains(nameToInsert))
    {
        String insert = "INSERT into TESTJ VALUES('"+ nameToInsert + "')";

        try (Statement stmt1 = conn1.createStatement()) {
            stmt1.executeQuery(insert);
        } catch (SQLException e) {
            System.out.println(e.getErrorCode());
        }
        System.out.println("SITE1 Transaction Completed");
        conn1.commit();

    try (Statement stmt3 = conn2.createStatement()) {
        stmt3.executeQuery(insert);
    }

```

```

    } catch (SQLException e) {
        System.out.println(e.getErrorCode());
    }
    System.out.println("SITE2 Transaction Completed");
    conn2.commit();
}

if (lastitem.contains(nameToDelete))
{
    String delete = "DELETE FROM TESTJ WHERE Name = " + "" + nameToDelete + "";

    try (Statement stmt2 = conn1.createStatement()) {
        stmt2.executeQuery(delete);
    } catch (SQLException e) {
        System.out.println(e.getErrorCode());
    }
    System.out.println("SITE1 Transaction Completed");
    conn1.commit();

    try (Statement stmt4 = conn2.createStatement()) {
        stmt4.executeQuery(delete);
    } catch (SQLException e) {
        System.out.println(e.getErrorCode());
    }
    System.out.println("SITE2 Transaction Completed");
    conn2.commit();

}

if (conn1 != null)
    conn1.close();

if (conn2 != null)
    conn2.close();

    System.out.println("Thread " + m_myId + " is finished. ");
}

catch (Exception e)
{
    System.out.println("Thread " + m_myId + " got Exception: " + e);
    e.printStackTrace();
    return;
}

```

```

    }
    semaphore.release();
}

}

```

Screenshots:

Terminal:

```

Rohits-MacBook-Pro:Assignment1 nidumukkala$ javac -cp ojdbc6.jar: JdbcMTSample.java
Rohits-MacBook-Pro:Assignment1 nidumukkala$ java -cp ojdbc6.jar: JdbcMTSample
Connected with connection #1
Connected with connection #2
Thread 1 Name : test1
Thread 1 Name : test0
Thread 1 Name : test3
Thread 1 Name : test4
Thread 1 Name : test1
Thread 1 Name : test0
Thread 1 Name : test3
Thread 1 Name : test4
SITE1 Transaction Completed
SITE2 Transaction Completed
Thread 1 is finished.
Connected with connection #1
Connected with connection #2
Thread 2 Name : test1
Thread 2 Name : test0
Thread 2 Name : test3
Thread 2 Name : test4
Thread 2 Name : test12
Thread 2 Name : test1
Thread 2 Name : test0
Thread 2 Name : test3
Thread 2 Name : test4
Thread 2 Name : test12
Thread 2 is finished.

```

```

Thread 8 Name : test0
Thread 8 Name : test3
Thread 8 Name : test4
Thread 8 Name : test12
Thread 8 is finished.
Connected with connection #1
Connected with connection #2
Thread 9 Name : test1
Thread 9 Name : test0
Thread 9 Name : test3
Thread 9 Name : test4
Thread 9 Name : test12
Thread 9 Name : test1
Thread 9 Name : test0
Thread 9 Name : test3
Thread 9 Name : test4
Thread 9 Name : test12
Thread 9 is finished.
Connected with connection #1
Connected with connection #2
Thread 10 Name : test1
Thread 10 Name : test0
Thread 10 Name : test3
Thread 10 Name : test4
Thread 10 Name : test12
Thread 10 Name : test1
Thread 10 Name : test0
Thread 10 Name : test3
Thread 10 Name : test4
Thread 10 Name : test12
Thread 10 is finished.
Rohits-MacBook-Pro:Assignment1 nidumukkala$

```

SQLDeveloper:

TESTJ table on Oracle VM

The screenshot shows the Oracle VM interface. On the left, the 'Connections' pane lists various database objects, including 'TESTJ'. The main window displays the 'TESTJ' table structure with a single column named 'NAME'. Below the structure, the data is listed in a table format.

	NAME
1	test1
2	test0
3	test3
4	test4
5	test12

TESTJ table on Ryerson Database:

The screenshot shows the Ryerson Database interface. On the left, the 'Connections' pane lists various database objects, including 'TESTJ'. The main window displays the 'TESTJ' table structure with a single column named 'NAME'. Below the structure, the data is listed in a table format.

	NAME
1	test1
2	test0
3	test3
4	test4
5	test12