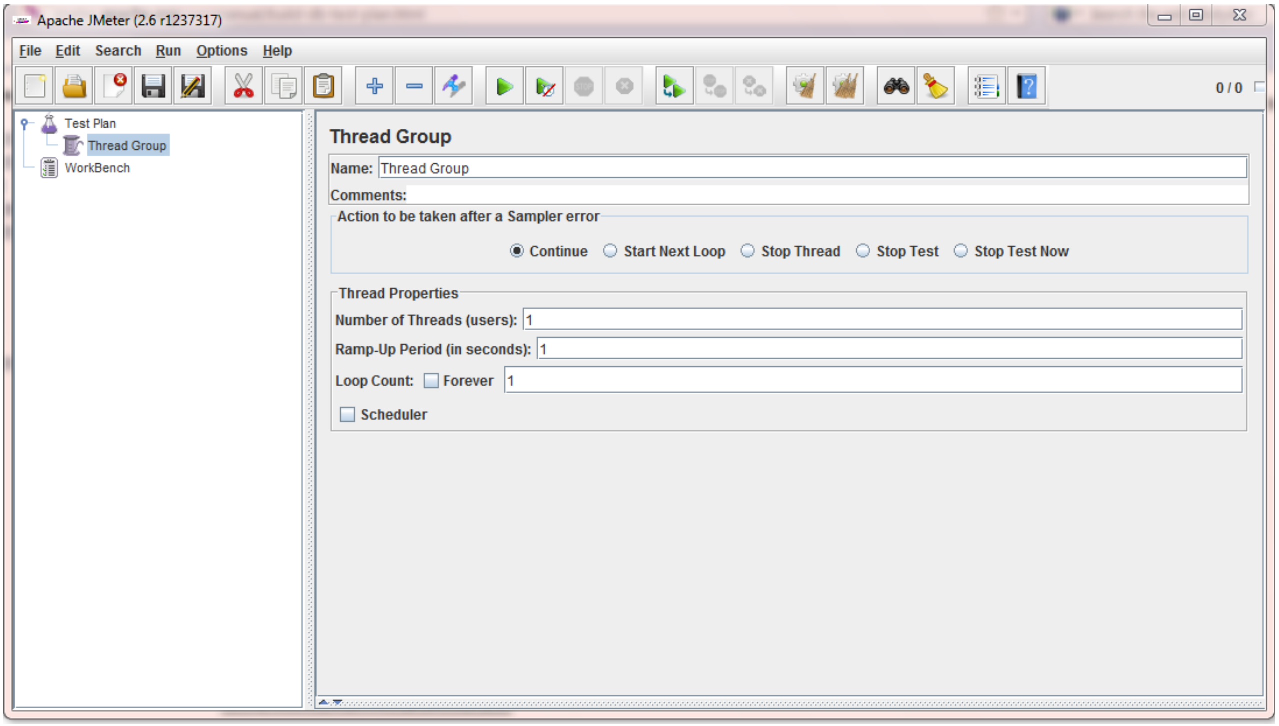
Building a Database Test Plan

In this lesson, you will learn how to create a basic Test Plan to test a database server. You will create ten users that send five SQL requests to the database server. Also, you will tell the users to run their tests three times. So, the total number of requests is (10 users) x (2 requests) x (repeat 3 times) = 60 JDBC requests.

1. Adding Users

The first step you want to do with every JMeter Test Plan is to add a Thread Group element. The Thread Group tells JMeter the number of users you want to simulate, how often the users should send requests, and the how many requests they should send.

Go ahead and add the ThreadGroup element by first selecting the Test Plan, clicking your right mouse button to get the Add menu, and then select Add → ThreadGroup.



Thread Group

Name: Thread Group

Comments:

Action to be taken after a Sampler error

● Continue ○ Start Next Loop ○ Stop Thread ○ Stop Test ○ Stop Test Now

Thread Properties

Number of Threads (users): 1

Ramp-Up Period (in seconds): 1

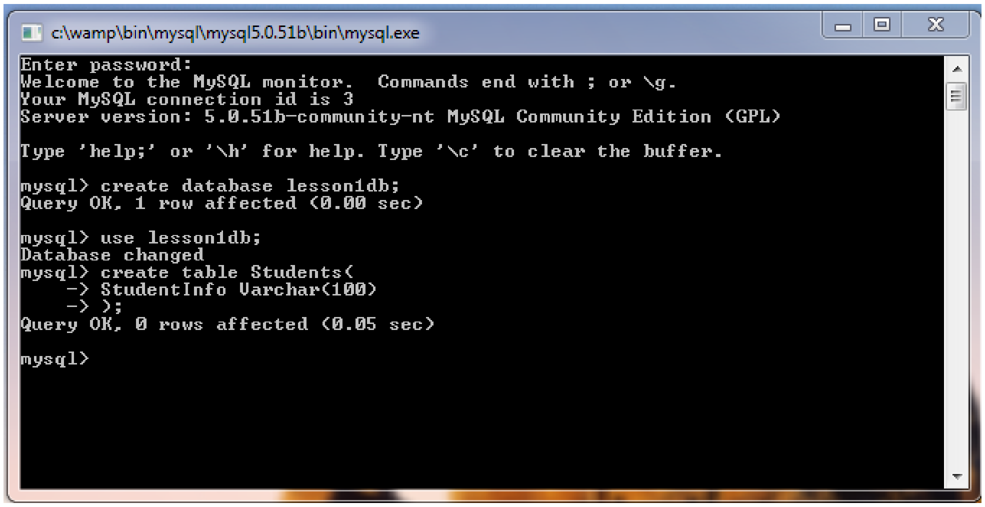
Loop Count: □ Forever 1

□ Scheduler

Figure1.1.Thread Group with Default Values

Providing a more descriptive name for the Thread Group as 'JDBC Users'.

You will need a valid MySQL database, database table, and user-level access to that table. In the example shown here, the database is 'lesson1db' and the table name is 'Students'



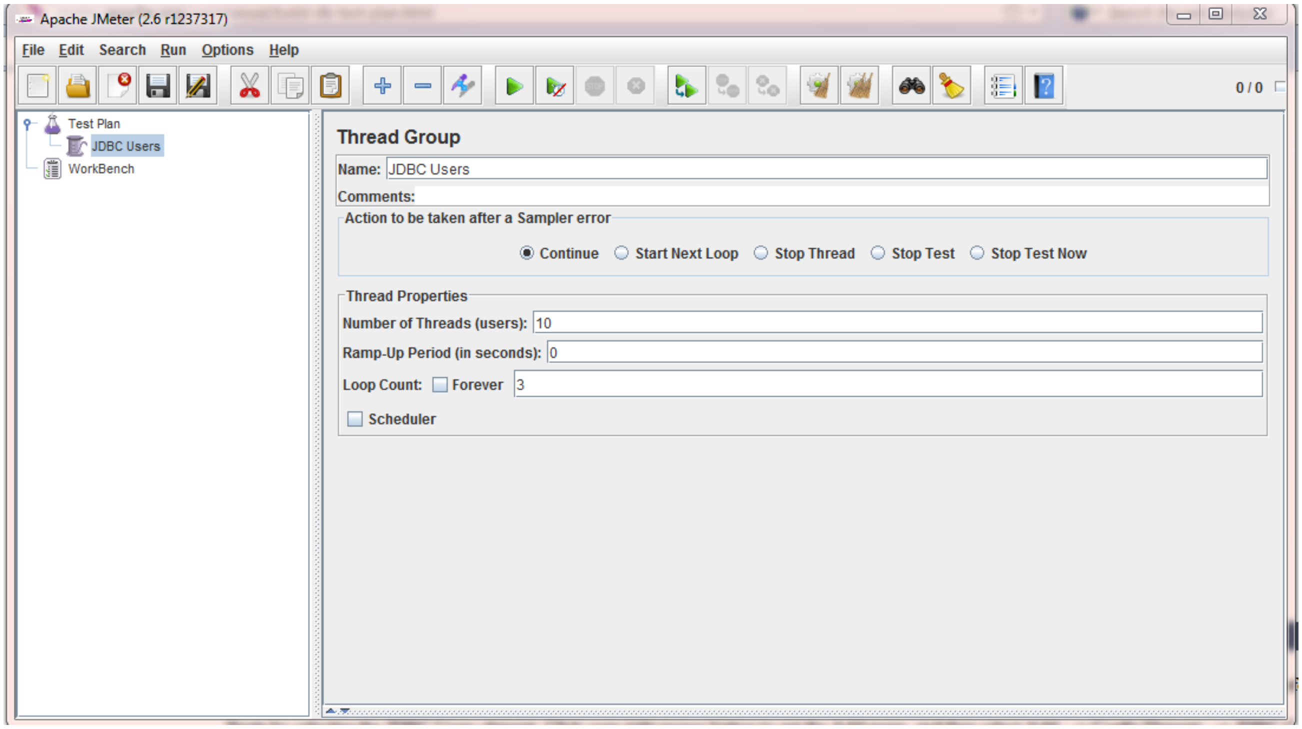
c:\wamp\bin\mysql\mysql5.0.51b\bin\mysql.exe

```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.0.51b-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> create database lesson1db;
Query OK, 1 row affected (0.00 sec)

mysql> use lesson1db;
Database changed
mysql> create table Students(
    -> StudentInfo Varchar(100)
    -> );
Query OK, 0 rows affected (0.05 sec)

mysql>
```

Figure1.2.MySql Command

Next, increase the number of users to 10.

In the next field, the Ramp-Up Period, leave the the default value of 0 seconds. This property tells JMeter how long to delay between starting each user. For example, if you enter a Ramp-Up Period of 5 seconds, JMeter will finish starting all of your users by the end of the 5 seconds. So, if we have 5 users and a 5 second Ramp-Up Period, then the delay between starting users would be 1 second (5 users / 5 seconds = 1 user per second). If you set the value to 0, then JMeter will immediately start all of your users.


Apache JMeter (2.6 r1237317)
File Edit Search Run Options Help
0 / 0
Test Plan
JDBC Users
WorkBench

Thread Group

Name: JDBC Users

Comments:

Action to be taken after a Sampler error

● Continue ○ Start Next Loop ○ Stop Thread ○ Stop Test ○ Stop Test Now

Thread Properties

Number of Threads (users): 10

Ramp-Up Period (in seconds): 0

Loop Count: □ Forever 3

□ Scheduler

Figure1.3.JDBC Users Thread Group

2. Adding JDBC Requests

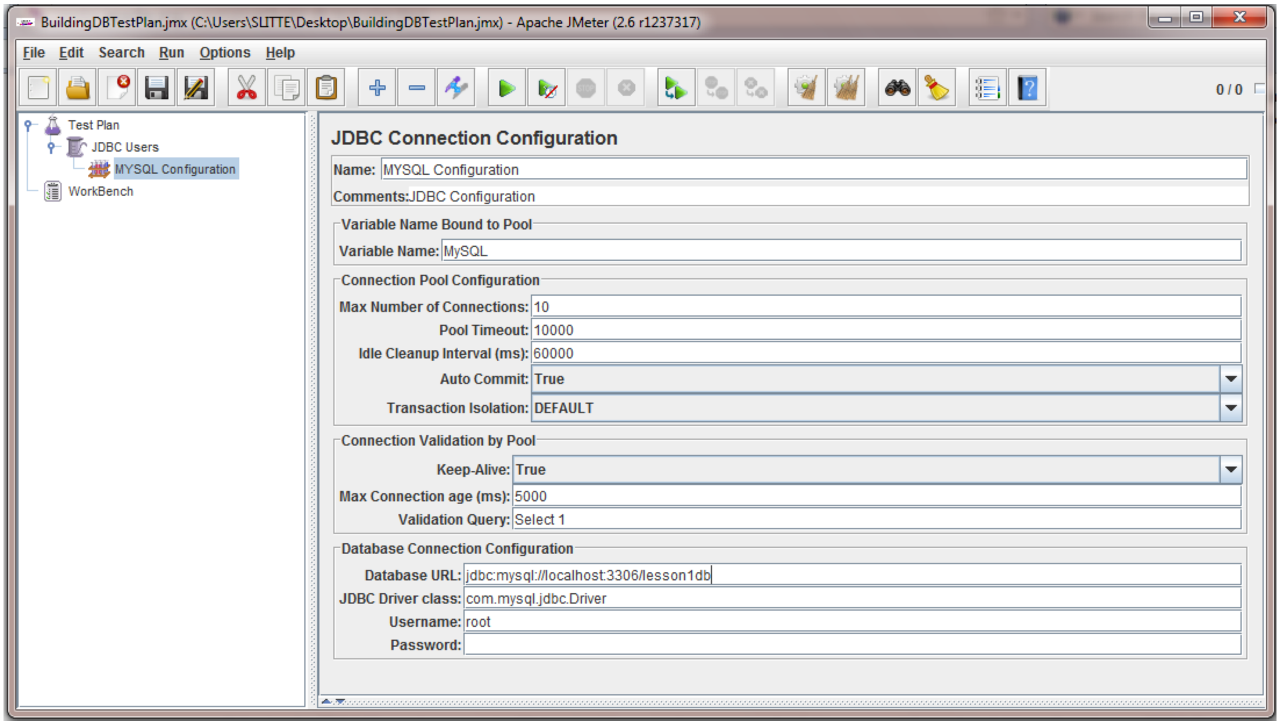
Begin by selecting the JDBC Users element. Click your right mouse button to get the Add menu, and then select Add → Config Element → JDBC Connection Configuration.

This example uses the MySQL database driver. Download MySQL Connector jar file from this link http://www.mysql.com/products/connector/
Extract the zip file and copy .jar file to the JMeter lib directory.

Set up the following fields:

- Variable name bound to pool. This needs to uniquely identify the configuration. It is used by the JDBC Sampler to identify the configuration to be used.
- Database URL: jdbc:mysql://localhost:3306/lesson1db
- JDBC Driver class: com.mysql.jdbc.Driver
- Username: "your MySQL username"
- Password: "your MySQL password"

The other fields on the screen can be left as the defaults. JMeter creates a database connection pool with the configuration settings as specified in the Control Panel. The pool is referred to in JDBC Requests in the 'Variable Name' field. Several different JDBC Configuration elements can be used, but they must have unique names. Every JDBC Request must refer to a JDBC Configuration pool. More than one JDBC Request can refer to the same pool.


BuildingDBTestPlan.jmx (C:\Users\SLITTE\Desktop\BuildingDBTestPlan.jmx) - Apache JMeter (2.6 r1237317)
File Edit Search Run Options Help
0 / 0
Test Plan
JDBC Users
MYSQL Configuration
WorkBench

JDBC Connection Configuration

Name: MYSQL Configuration

Comments:JDBC Configuration

Variable Name Bound to Pool

Variable Name: MySQL

Connection Pool Configuration

Max Number of Connections: 10

Pool Timeout: 10000

Idle Cleanup Interval (ms): 60000

Auto Commit: True

Transaction Isolation: DEFAULT

Connection Validation by Pool

Keep-Alive: True

Max Connection age (ms): 5000

Validation Query: Select 1

Database Connection Configuration

Database URL: jdbc:mysql://localhost:3306/lesson1db

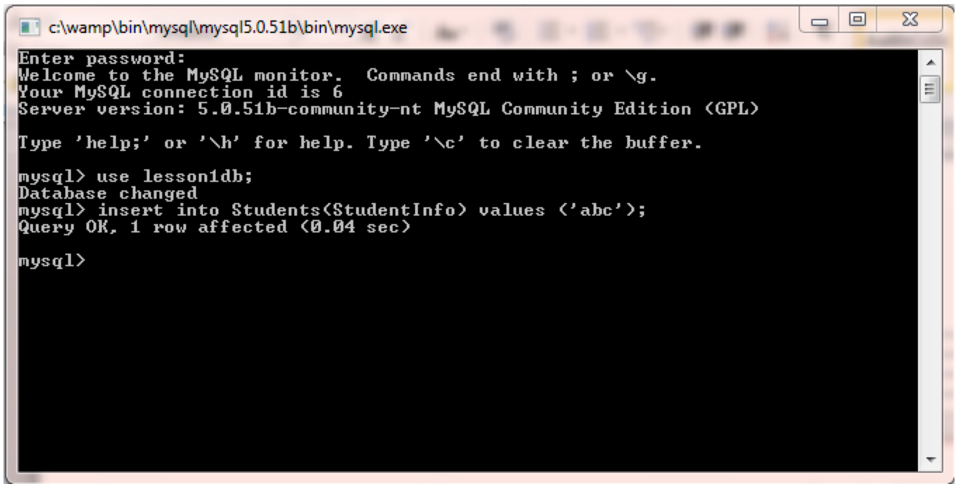
JDBC Driver class: com.mysql.jdbc.Driver

Username: root

Password:

Figure1.4.JDBC Configuration

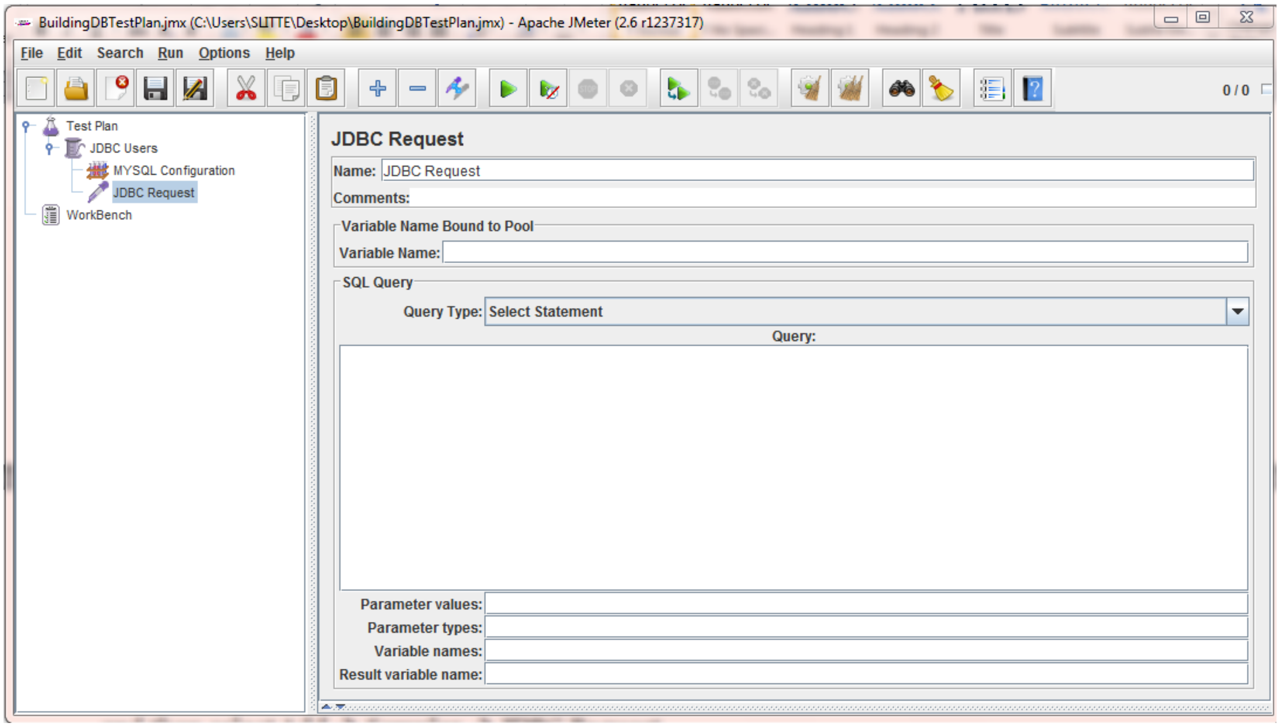
You need to insert some values for your MySQL database called 'lesson1db'.



```
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.0.51b-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use lesson1db;
Database changed
mysql> insert into Students(StudentInfo) values ('abc');
Query OK, 1 row affected (0.04 sec)

mysql>
```

Figure1.4.Insert data to the MySQL database

Selecting the JDBC Users element again. Click your right mouse button to get the Add menu, and then select Add → Sampler → JDBC Request



BuildingDBTestPlan.jmx (C:\Users\SLITTE\Desktop\BuildingDBTestPlan.jmx) - Apache JMeter (2.6 r1237317)

Test Plan
JDBC Users
MYSQL Configuration
JDBC Request
WorkBench

JDBC Request

Name: JDBC Request

Comments:

Variable Name Bound to Pool

Variable Name:

SQL Query

Query Type: Select Statement

Query:

Parameter values:

Parameter types:

Variable names:

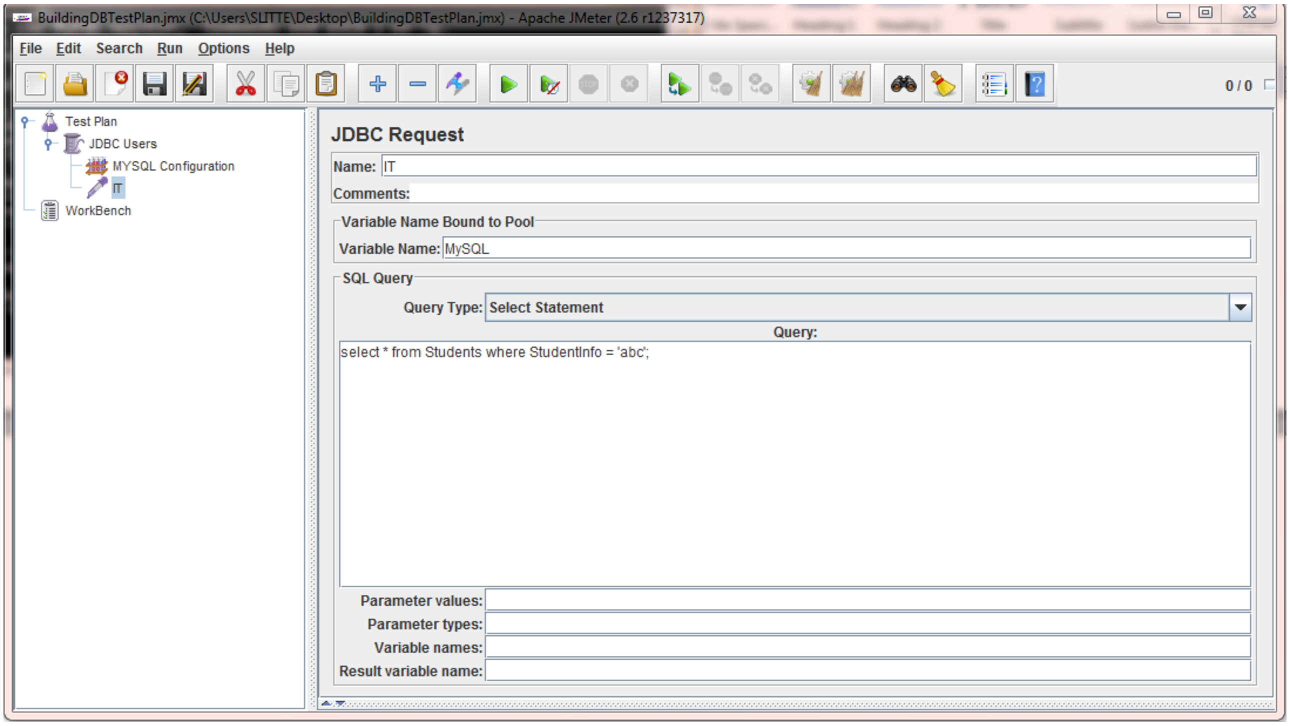
Result variable name:

Figure1.5 JDBC Request

In our Test Plan, we will make two JDBC requests. The first one is for IT Student, and the second is SE Student.

- Change the Name to "IT".
- Enter the Pool Name: MySQL (same as in the configuration element)
- Enter the SQL Query String field

Select * from Students where StudentInfo=’abc’;


BuildingDBTestPlan.jmx (C:\Users\SLITTE\Desktop\BuildingDBTestPlan.jmx) - Apache JMeter (2.6 r1237317)
File
Edit
Search
Run
Options
Help
0 / 0
Test Plan
JDBC Users
MYSQL Configuration
IT
WorkBench

JDBC Request

Name: IT

Comments:

Variable Name Bound to Pool

Variable Name: MySQL

SQL Query

Query Type: Select Statement

Query:

```
select * from Students where StudentInfo = 'abc';
```

Parameter values:

Parameter types:

Variable names:

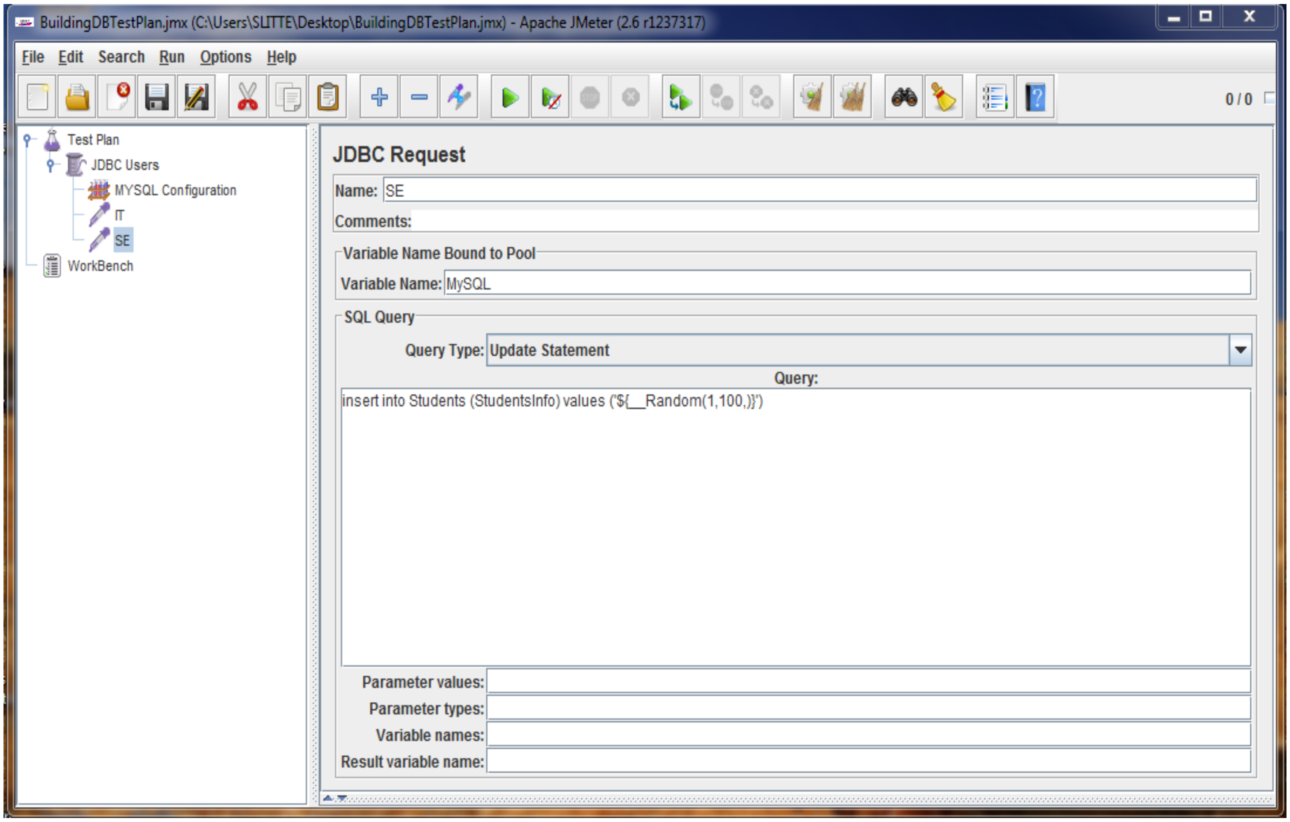
Result variable name:

Figure1.6.JDBC Request for IT Student

Next, add the second JDBC Request and edit the following properties:

- Change the Name to "SE".
- Enter the SQL Query String field.

Insert into Students (StudentInfo) values (‘${__Random (1, 100,)}’)


BuildingDBTestPlan.jmx (C:\Users\SLITTE\Desktop\BuildingDBTestPlan.jmx) - Apache JMeter (2.6 r1237317)
File Edit Search Run Options Help
0 / 0
Test Plan
JDBC Users
MYSQL Configuration
IT
SE
WorkBench
JDBC Request

Name: SE

Comments:

Variable Name Bound to Pool

Variable Name: MySQL

SQL Query

Query Type: Update Statement

Query:

```
insert into Students (StudentsInfo) values ('${__Random(1,100,)}')
```

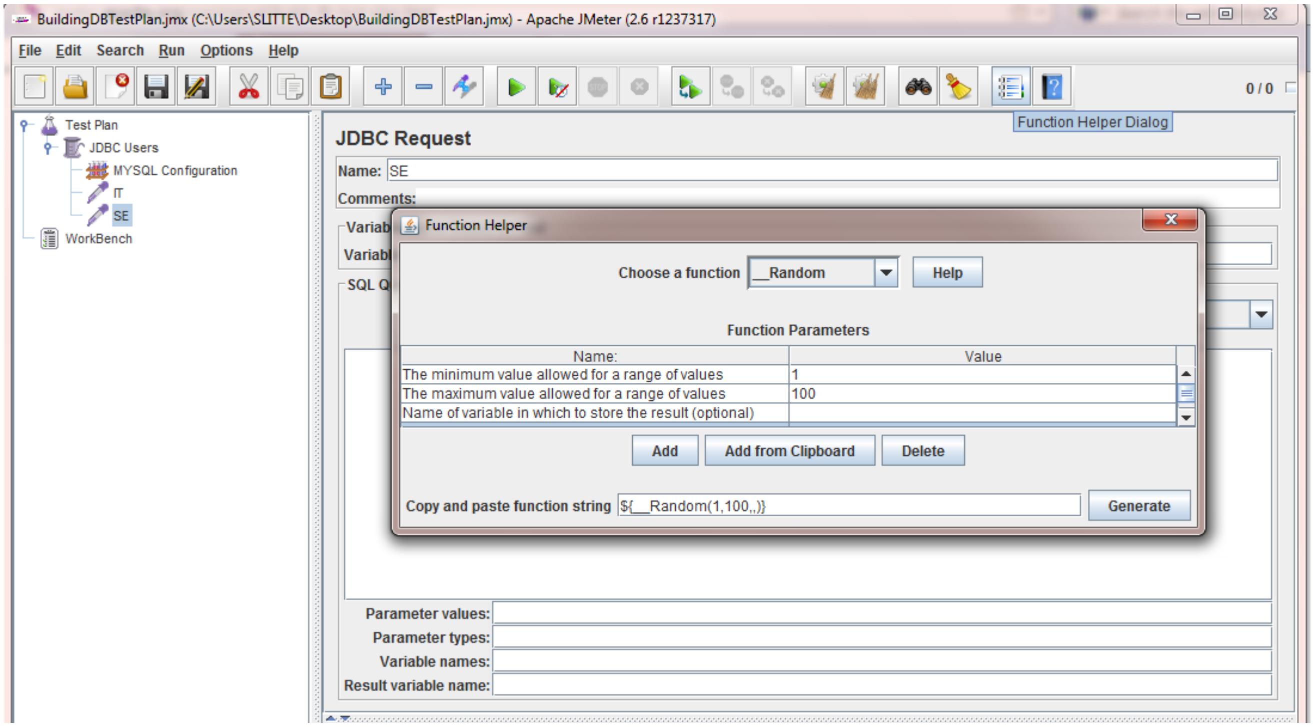
Parameter values:

Parameter types:

Variable names:

Result variable name:

To get the random value will use the Function Helper Dialog box.



Name: SE

Comments:

Choose a function: __Random

Name:	Value
The minimum value allowed for a range of values	1
The maximum value allowed for a range of values	100
Name of variable in which to store the result (optional)	

Copy and paste function string `${__Random(1,100,,)}`

Parameter values:

Parameter types:

Variable names:

Result variable name:

Figure1.8.JDBC Request for SE Student

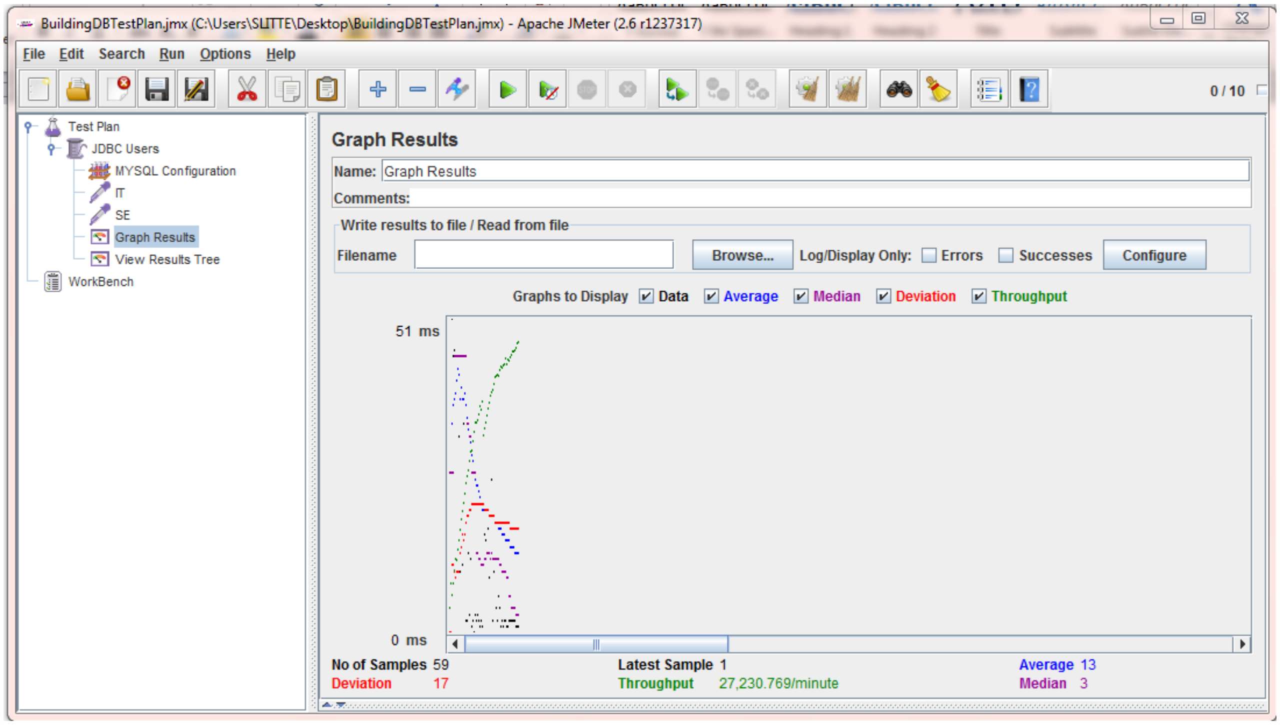
3. Adding a Listener to View/Store the test Results

The final element you need to add to your Test Plan is a Listener. This element is responsible for storing all of the results of your JDBC requests in a file and presenting a visual model of the data.

Select the JDBC Users element and add a Graph Results listener (Add → Listener → Graph Results).

Select the JDBC Users element and add a View Results Tree (Add → Listener → View Results Tree).

Before execute the JMeter test plan you need to log on to your MySQL server.



Graph Results

Name: Graph Results

Comments:

Write results to file / Read from file

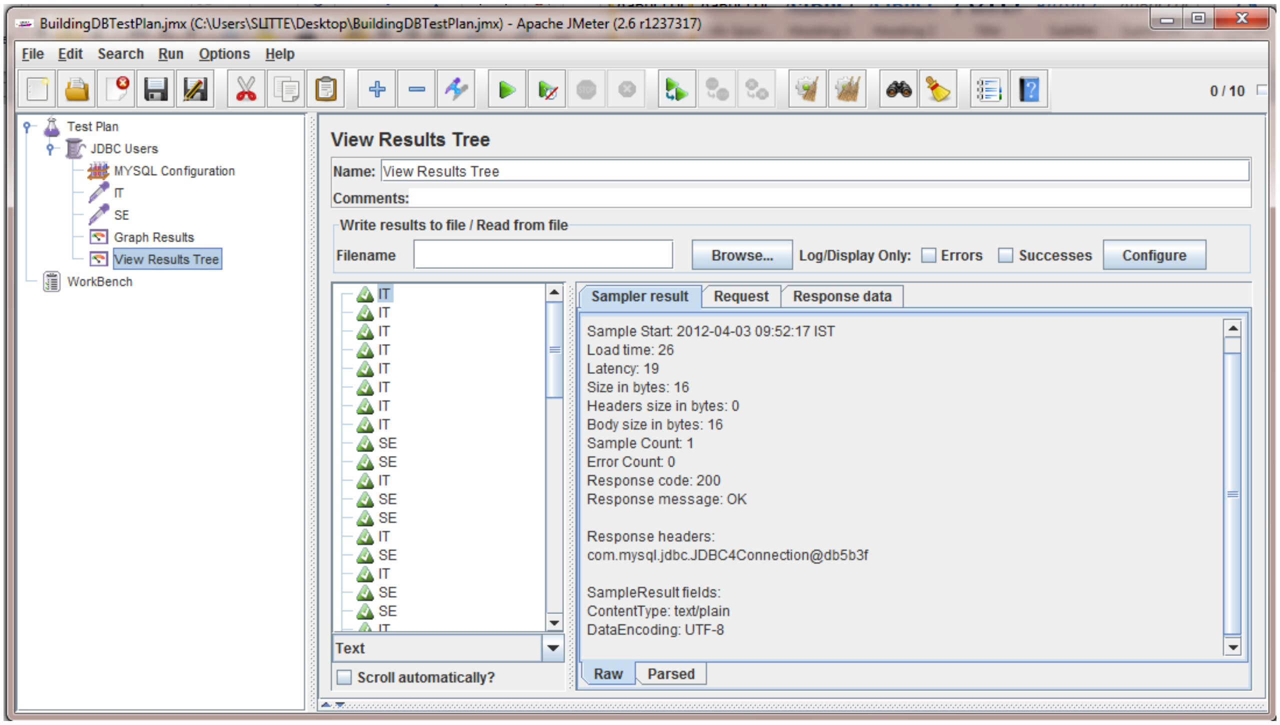
Filename

Log/Display Only: □ Errors □ Successes

Graphs to Display ☑ Data ☑ Average ☑ Median ☑ Deviation ☑ Throughput

No of Samples 59
Deviation 17
Latest Sample 1
Throughput 27,230.769/minute
Average 13
Median 3

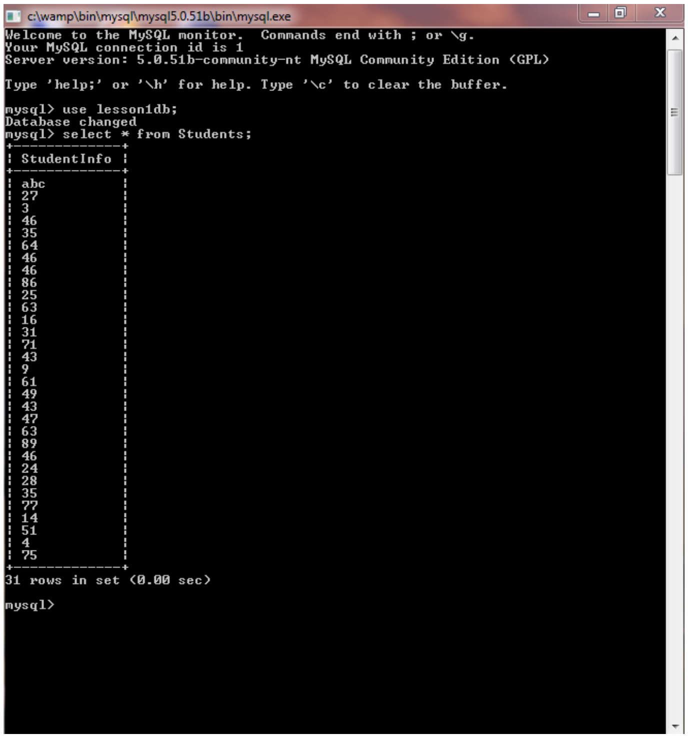
Figure1.9.Graph Results Listener



```
Sample Start: 2012-04-03 09:52:17 IST
Load time: 26
Latency: 19
Size in bytes: 16
Headers size in bytes: 0
Body size in bytes: 16
Sample Count: 1
Error Count: 0
Response code: 200
Response message: OK

Response headers:
com.mysql.jdbc.JDBC4Connection@db5b3f

SampleResult fields:
ContentType: text/plain
DataEncoding: UTF-8
```

Figure1.9.View Results Tree

You can test your data relevant to "Students" table form MySQL command line.



```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.0.51b-community-nt MySQL Community Edition (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use lesson1db;
Database changed
mysql> select * from Students;
+-------------+
| StudentInfo |
+-------------+
| abc         |
| 27          |
| 3           |
| 46          |
| 35          |
| 64          |
| 46          |
| 46          |
| 86          |
| 25          |
| 63          |
| 16          |
| 31          |
| 71          |
| 43          |
| 9           |
| 61          |
| 49          |
| 43          |
| 47          |
| 63          |
| 89          |
| 46          |
| 24          |
| 28          |
| 35          |
| 77          |
| 14          |
| 51          |
| 4           |
| 75          |
+-------------+
31 rows in set (0.00 sec)

mysql>
```

Figure1.10.View Results in Command line