HammerDB Oracle Trace Replay

This guide gives you an introduction to using HammerDB with Oracle Trace Files that can be converted and replayed against an Oracle database. Within HammerDB this trace file replay functionality is only available with Oracle as the other supported databases do not provide an interface to generating a detailed trace of workloads.

Generating Oracle Trace Files

To begin converting Oracle trace file workloads with HammerDB the first step is to load an Oracle trace file under the File:Open menu option. Before doing this therefore an Oracle trace file must be generated using Oracle Event 10046. (There are numerous methods to generate Oracle trace files of which only event 10046, level 4 to capture bind variables is covered here).

As an example the simplest method to start and stop the trace of a session interactively is as follows:

```
PDB1@ORCL> connect / as sysdba
Connected.
CDB$ROOT@ORCL> alter session set events '10046 trace name context forever, level
4';

Session altered.

CDB$ROOT@ORCL> select sysdate from dual;

SYSDATE
---------
01-SEP-14

CDB$ROOT@ORCL> alter session set events '10046 trace name context off';

Session altered.
```

For more advanced use the creation of a logon trigger is recommended. This trigger can then be enabled or disabled to capture the trace information for a particular use. The example uses the user TPCC created for an Oracle HammerDB OLTP test.

```
CDB$ROOT@ORCL> create or replace trigger logon_trigger
  2  after logon on database
  3  begin
  4  if (user = 'TPCC') then
  5  execute immediate
```

```
  6  'alter session set events ''10046 trace name context forever, level 4''';
  7  end if;
  8  end;
  9  /

Trigger created.

CDB$ROOT@ORCL> connect tpcc/tpcc@pdb1
Connected.
PDB1@ORCL> select sysdate from dual;

SYSDATE
---------
01-SEP-14
```

The trigger must be created as SYS with SYSDBA privileges, if created by system the trigger will create successfully but fail on the user login.

This event will produce a trace file in the diagnostic area specified for the database server and not on the client system, for example:

```
[oracle@MERLIN trace]$ pwd
/u01/app/oracle/diag/rdbms/orcl/orcl/trace
```

By default the file will be identifiable by ora_SPID.trc, however there are also methods that can be used to set the name of the trace file. Note that in a Shared Server environment (previously MTS) one users' session may be distributed across numerous trace files as the user processes share multiple server processes. Therefore in this environment it is necessary to reassemble the trace file data before converting with the Oracle utility 'trcsess'.

An example extract from a raw trace file is shown below:

```
[oracle@MERLIN trace]$ more orcl_ora_3831.trc
Trace file /u01/app/oracle/diag/rdbms/orcl/orcl/trace/orcl_ora_3831.trc
Oracle Database 12c Enterprise Edition Release 12.1.0.1.0 - 64bit Production
With the Partitioning, OLAP, Advanced Analytics and Real Application Testing opt
Ions
=====================
PARSING IN CURSOR #139906403096752 len=24 dep=0 uid=0 oct=3 lid=0 tim=366990772
hv=2343063137 ad='d5d3fc90' sqlid='7h35uxf5uhmm1'
select sysdate from dual
END OF STMT
PARSE #139906403096752:c=2000,e=3953,p=0,cr=0,cu=0,mis=1,r=0,dep=0,og=1,plh=1388
734953,tim=366990768
EXEC #139906403096752:c=0,e=28,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=1,plh=1388734953
,tim=366990837
FETCH #139906403096752:c=0,e=20,p=0,cr=0,cu=0,mis=0,r=1,dep=0,og=1,plh=138873495
3,tim=366990901
STAT #139906403096752 id=1 cnt=1 pid=0 pos=1 obj=0 op='FAST DUAL  (cr=0 pr=0 pw=
0 time=9 us cost=2 size=0 card=1)'
FETCH #139906403096752:c=0,e=5,p=0,cr=0,cu=0,mis=0,r=0,dep=0,og=0,plh=1388734953
,tim=366991629
```

For more information on the trace file format the document Note:39817.1 Subject “Interpreting Raw SQL_TRACE output “ available from My Oracle Support as shown in Figure1 is a useful reference, however this knowledge is not essential as HammerDB can convert this raw format into a form that can be replayed.

Figure 1 My Oracle Support

Converting Oracle Trace Files

It is important to note that to convert Oracle trace files you must have selected Oracle from the treeview. If another database is selected the button to convert Oracle trace files is disabled as shown in Figure 2.

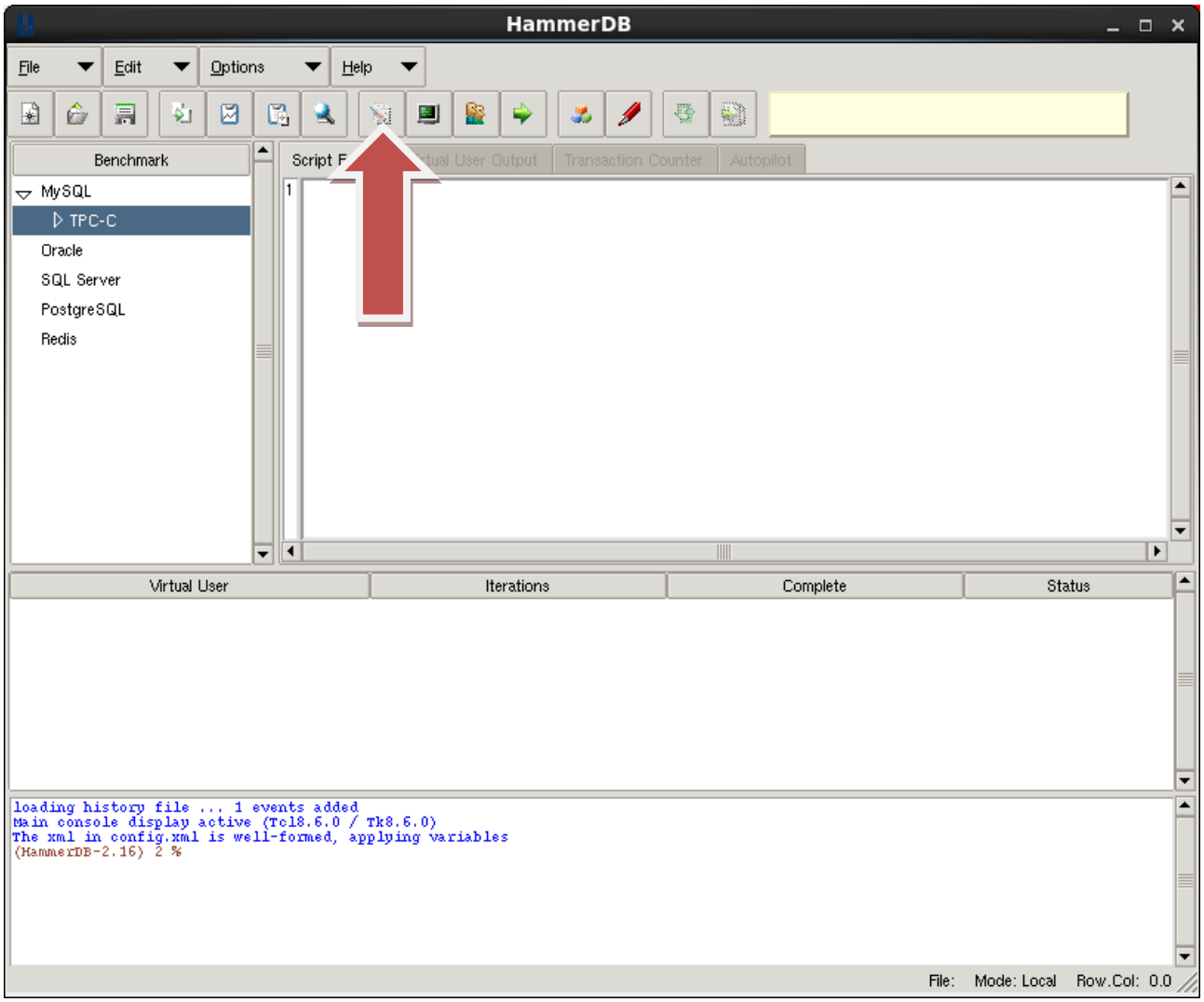

Figure 2 Trace Conversion Disabled

To enable tracing from the Benchmark Options select Oracle as shown in Figure 3.

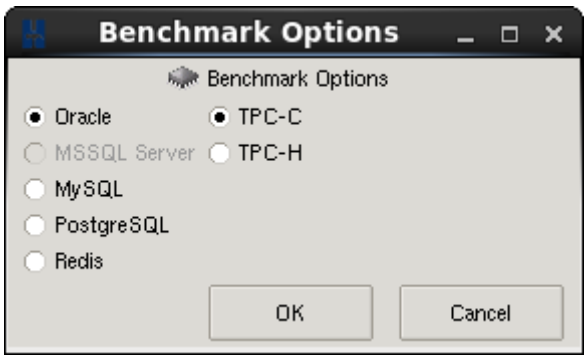

Figure 3 Benchmark Options

Once selected Oracle will move to the top of the treeview and the trace conversion button is enabled, ready for a trace file to be loaded as shown in Figure 4.

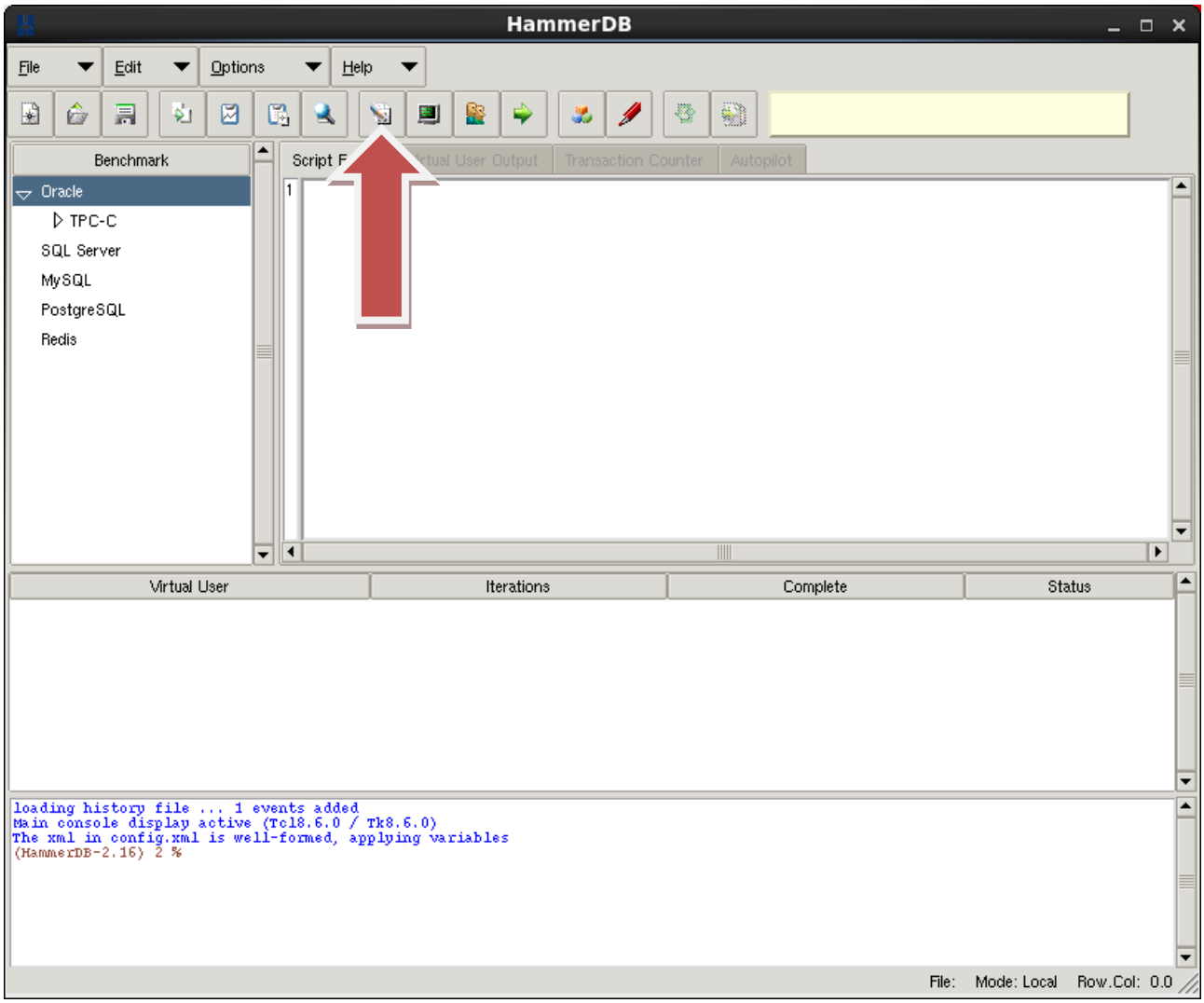

Figure 4 Trace Conversion Enabled

Copy the trace file to the client machine or location where HammerDB is running and use the File:Open menu option or the “Open an existing file button” to display the Open File dialogue as shown in Figure 5.

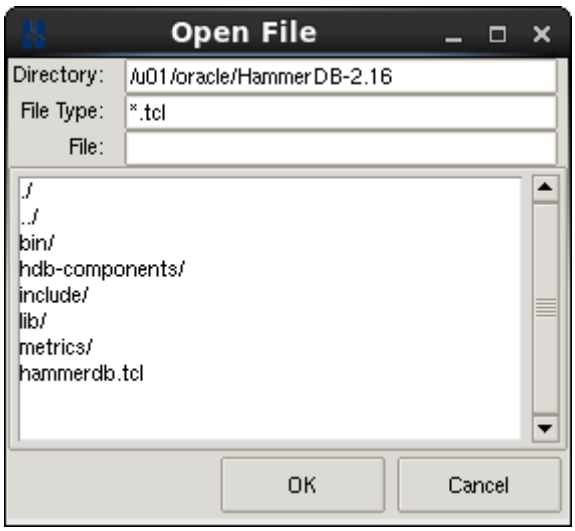

Figure 5 Open File Dialogue

The Open File dialog allows the specifying of the Directory and a filter for the file type, by default this is *.tcl. Change the file extension to 'trc' and change directory to the location of your files, select the trace file you previously generated as shown in Figure 6.

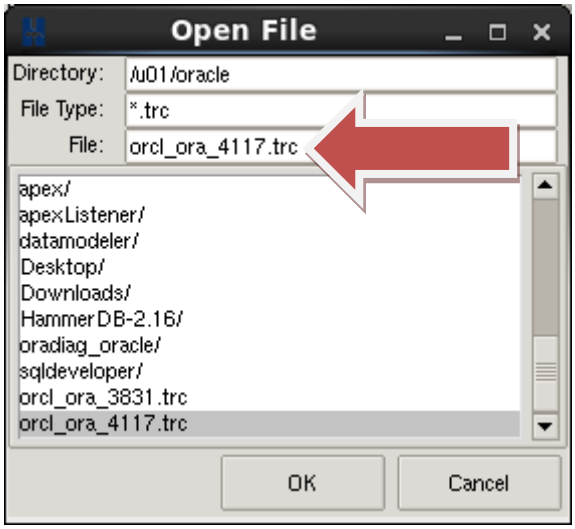

Figure 6 Select trace File

The raw trace file will be loaded into the Script Editor window enabling you Click the "Convert Oracle Trace file to Oratcl" button as shown in Figure 7.

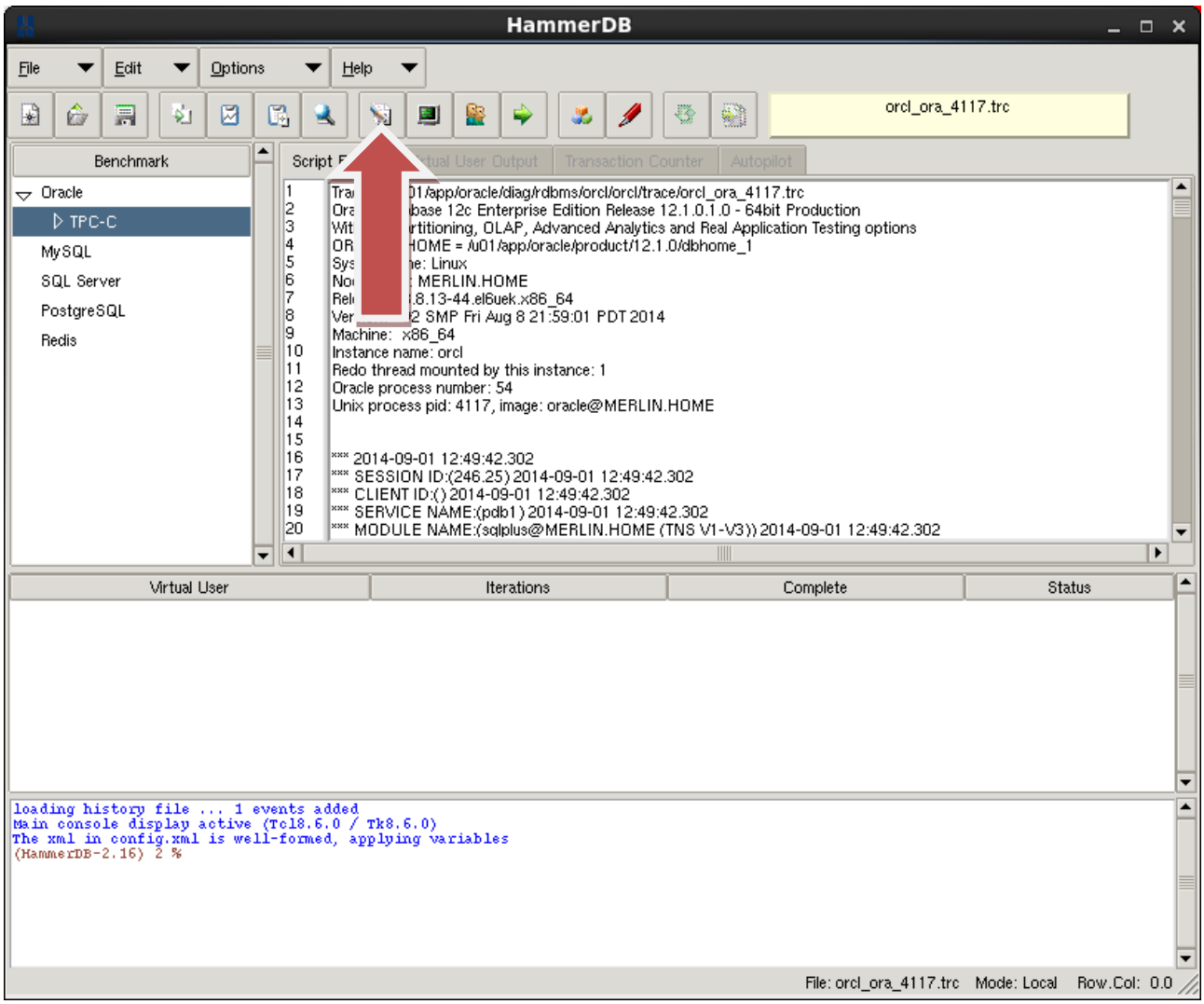

Figure 7 Raw trace file loaded for conversion.

The Trace file will be converted into a form that can be replayed by HammerDB as shown in Figure 8.

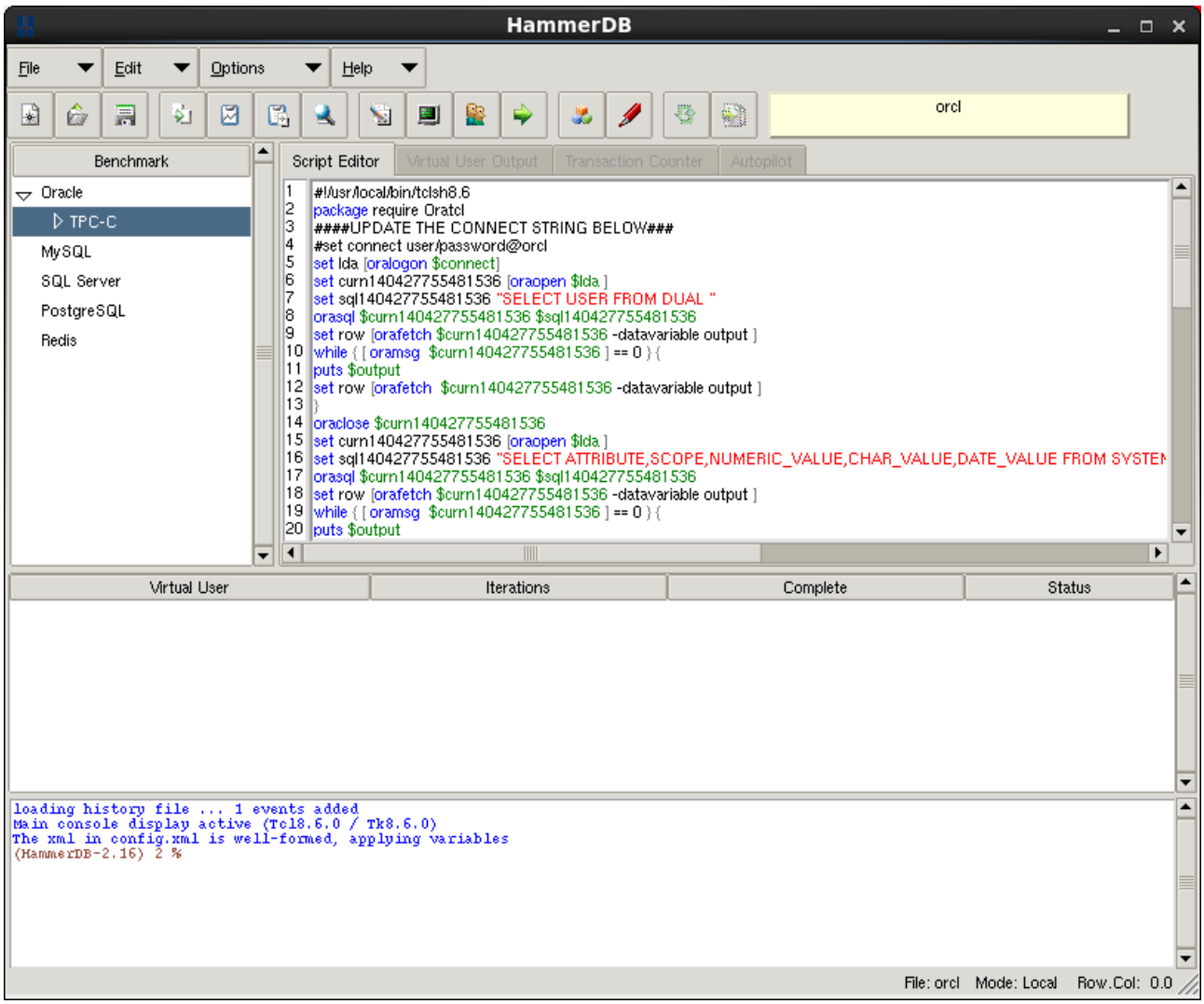

Figure 8 Converted Trace File

Note that the trace file never records a users' authentication details. Therefore the connect string must always be modified manually after conversion. Remove the comment before the "set connect" line on line 4 and enter the correct username and password. The SID will be set by default to the SID that the trace file was taken from and therefore if using a pluggable database then the correct container must also be set to the correct identifier as shown in Figure 9.

Figure 9 Modified Connect String

Once the connect string is updated the generated script is ready for running against the database and as shown in Figure 10 contains the original statements that were traced.

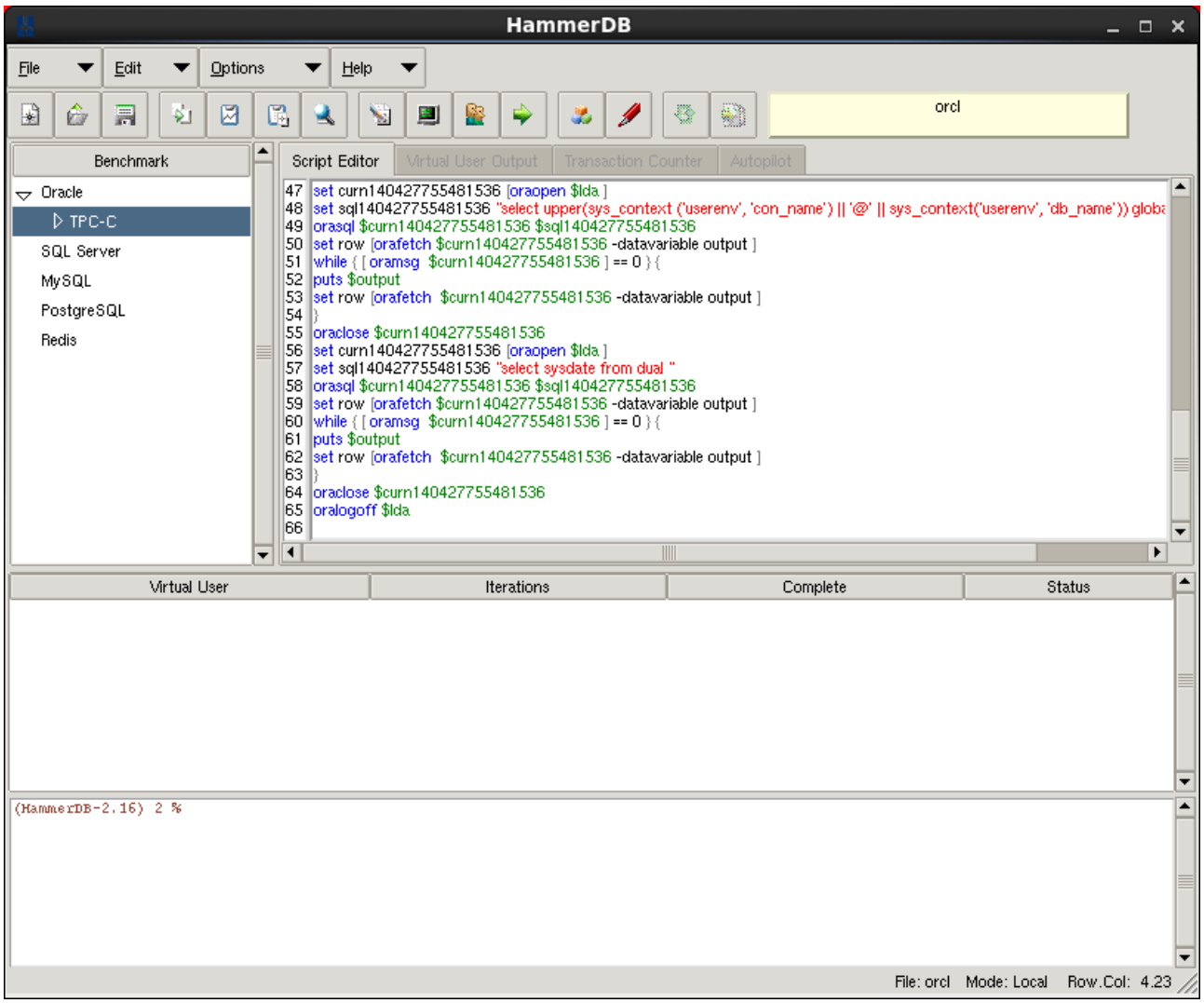

Figure 10 Traced Statements

The save menu option or “Save current file” button can be used to save the generated script for reloading at a later point in time.

Replaying Oracle Trace Files

Next to the “Convert” button there is a “Test current code” button. Click on this to test the code in an individual Virtual User environment. Once tested the window can be closed manually or by clicking the same button now containing a stop image. Figure 11 shows the output of replaying the script above converted from the trace file generated by the logon trigger.

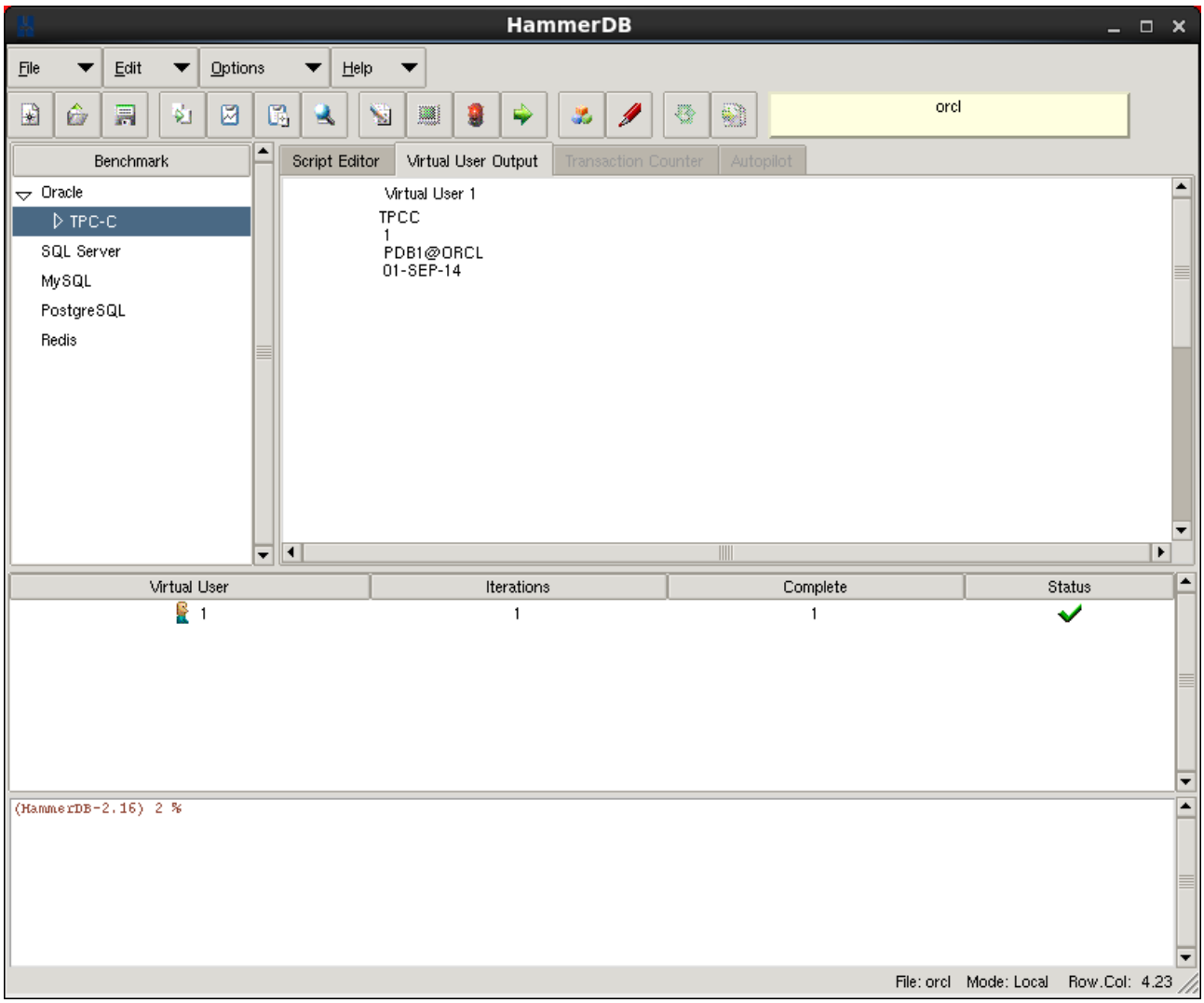

Figure 11 Replayed Script

Once the script has been tested and is ready for running select the Virtual Users:Vuser Options menu option or from the treeview. Enter the desired number of Virtual Users and click OK.

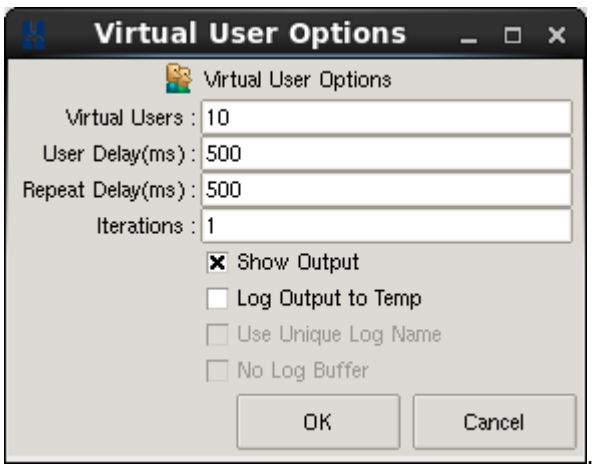

Figure 12 Virtual User Options

Once the options have been selected click on the “Create Virtual Users” button or select the same option from the treeview. This will create the selected number of Virtual Users that can then be run with the Run Virtual Users option as shown in Figure 13.

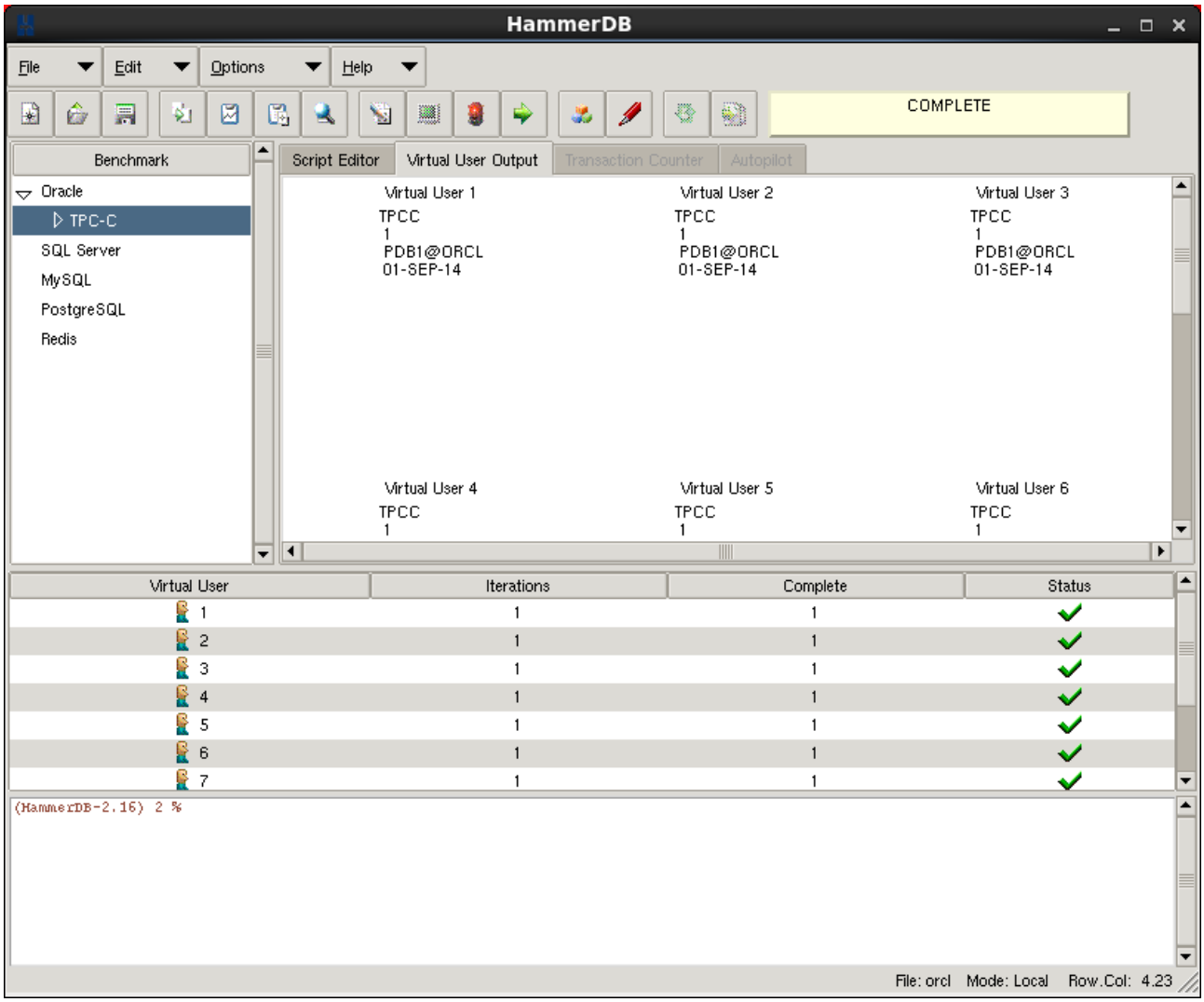

Figure 13 Run Virtual Users

The tracefile has now been replayed with multiple Virtual Users illustrating the basic building blocks for creating a bespoke Oracle workload from a captured and converted trace file. As a further examples using the same logon trigger for the TPCC user the HammerDB OLTP workload has now been loaded and for example purposes the number of transactions set to 100 as shown in Figure 14.

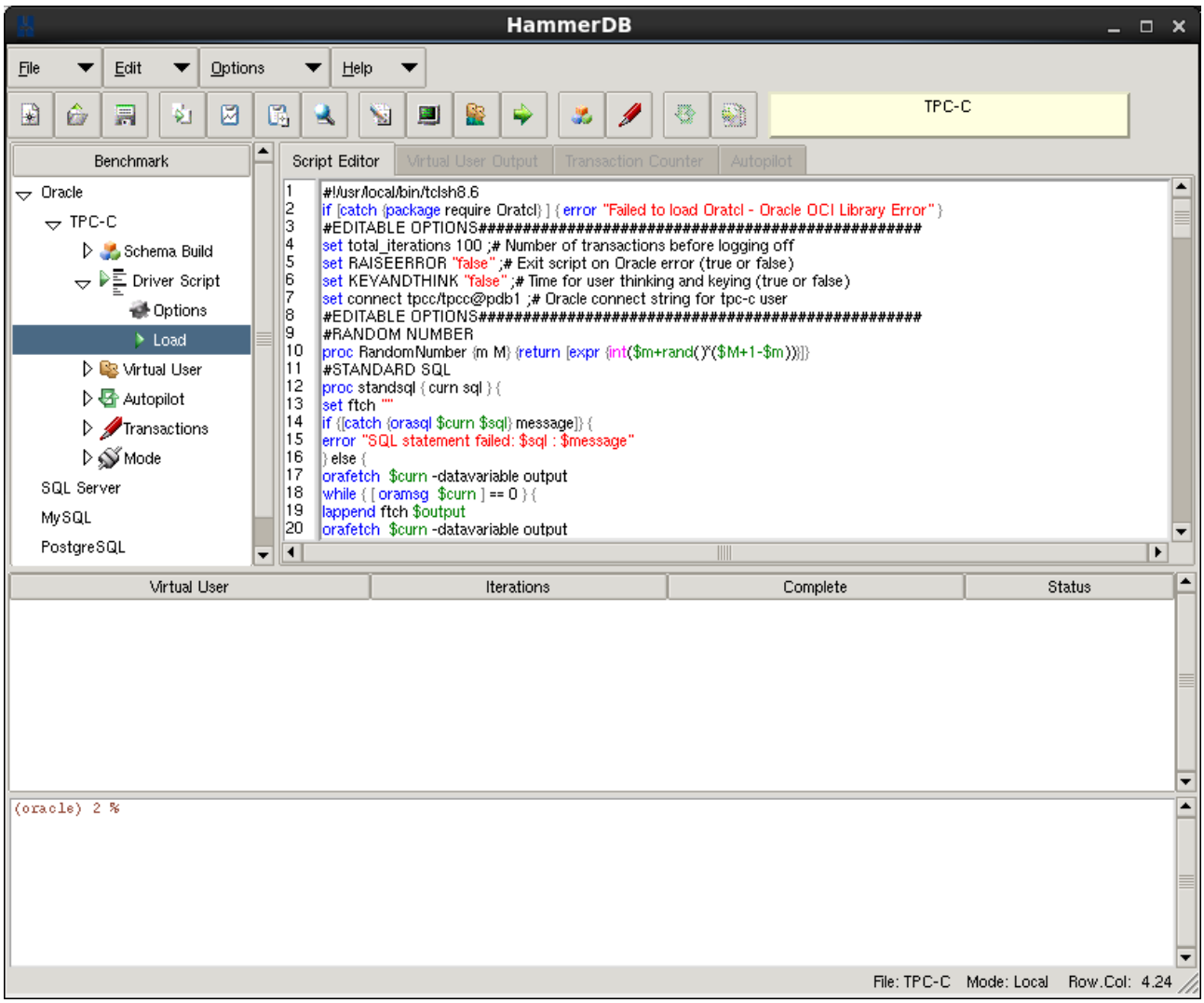

Figure 14 TPC-C Workload Loaded

Running the workload as normal generates a tracefile that can be loaded. Converting the File as previously shown and updating the connect string means that the workload from the tracefile is now in a format that can be replayed as shown in Figure 15.

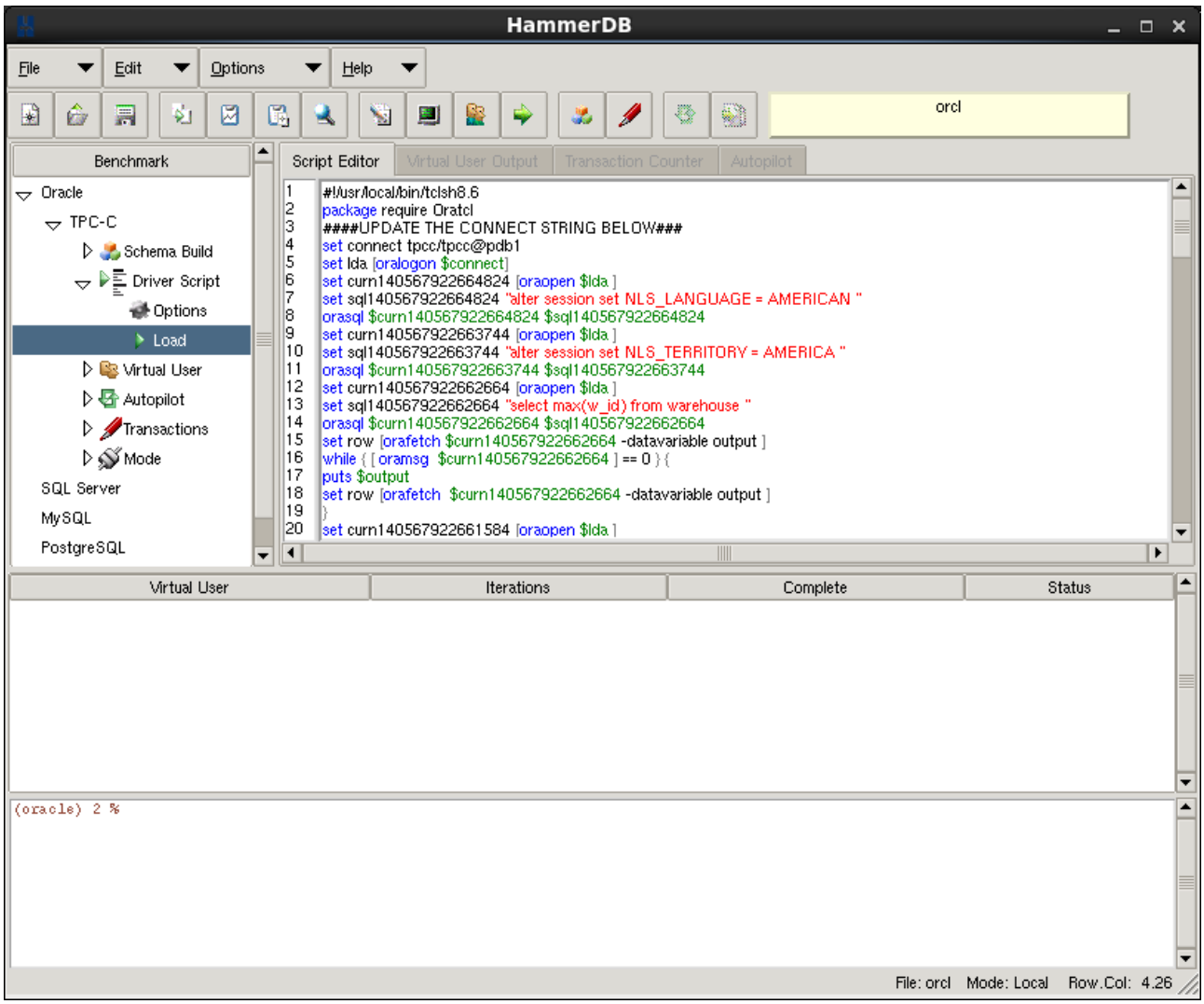

Figure 15 Converted tracefile

Note that as shown in Figure 16 HammerDB will correctly format, extract and insert bind variables for workload replay. Once complete click on “Destroy Virtual Users” to return back to an original starting state.

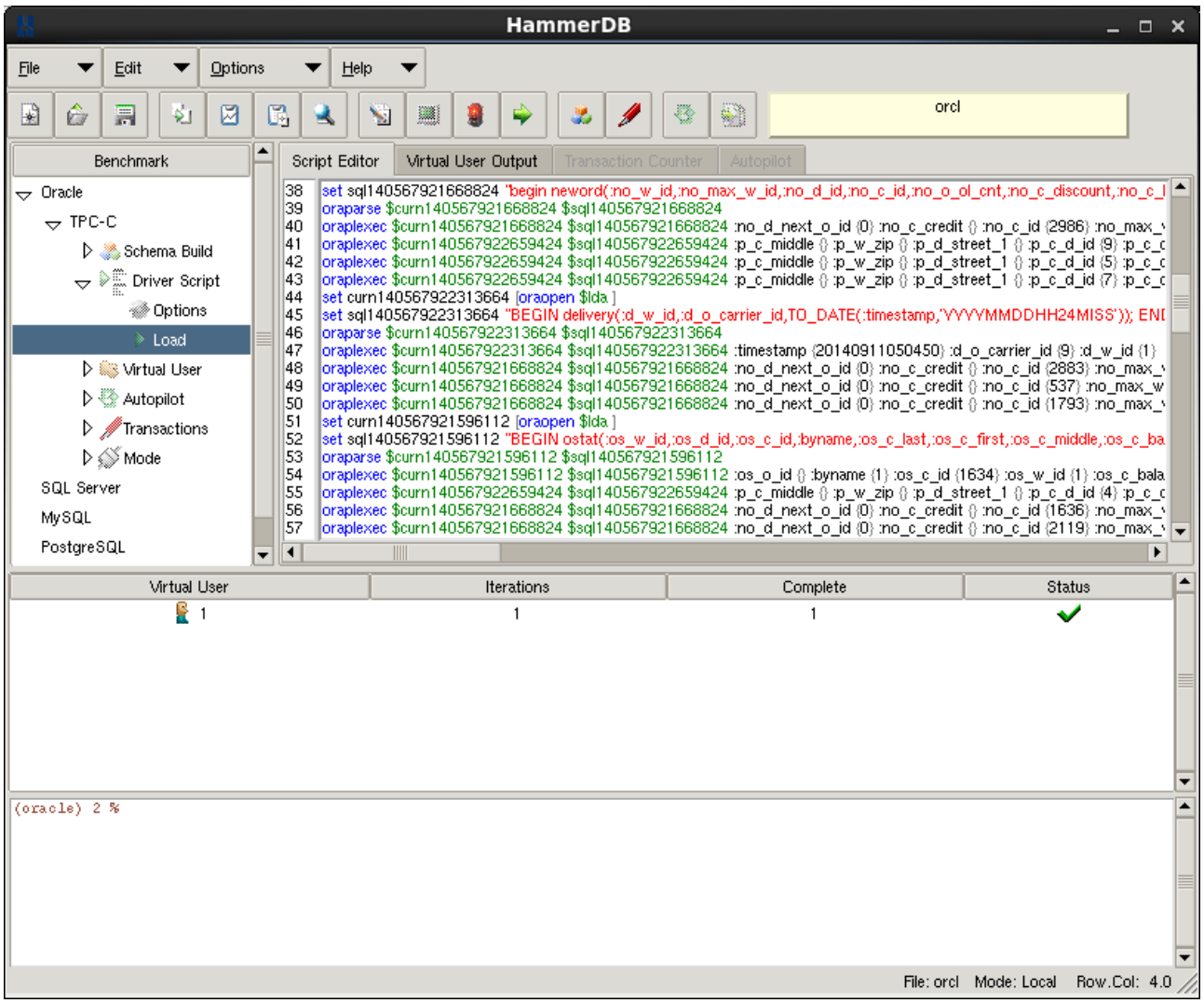

Figure 16 Replayed tracefile

Capturing Errors from Trace File Workloads

It must be remembered that although the previous workloads shown are suitable for directly replaying, with some applications replaying data from a previous transaction may result in constraint violations or updates of data that no longer exists that results in errors during replay. One of the most common error messages received is is the standard PL/SQL failure reported by Oratcl oraplexec: SQL execution failed. To get the actual Oracle error message underlying this error you can use the TCL "catch" command to capture the error and print it out inline. For example if you wanted to see the error in the neword procedure of the TPC-C script change the oraplexec line to look like the following :

```
if {[ catch {oraplexec $curn1 $sql5 :no_w_id $no_w_id :no_max_w_id $w_id_input
:no_d_id $no_d_id :no_c_id $no_c_id :no_o_ol_cnt $ol_cnt :no_c_discount {NULL}
:no_c_last {NULL} :no_c_credit {NULL} :no_d_tax {NULL} :no_w_tax {NULL}
:no_d_next_o_id {0} :timestamp $date} message]} {
puts $message
puts [ oramsg $curn1 all ]
}
```

So you are adding the statements as below to the oraplexec statement

```
if {[ catch { ... } message] } {
puts $message
puts [ oramsg $curn1 all ]
}
```

Note that the cursor variable $curn1 used in oramsg is the same variable used in the previous oraplexec.

You can then run the script by pressing the test current code button (or running one user thread with an output window)

The $message variable will contain and print out (using the "puts" command) any TCL error and the [oramsg $curn1 all] command will then print out the oracle error from the cursor $curn1.

An example is if the procedure neword has not yet been created:

This then gives the full output from Oracle as shown here instead of just the standard message :

```
new order
{6550 {ORA-06550: line 1, column 7:
PLS-00201: identifier 'NEWORD' must be declared
ORA-06550: line 1, column 7:
PL/SQL: Statement ignored} 0 0 4 8}
```

Support and Questions

For help use the HammerDB Sourceforge forum available at the HammerDB sourceforge project.