Evidence-based Best Practices for JD Edwards EnterpriseOne

Using Oracle 11gR2
Real Application Testing

Dallas Willett & Jeremiah Wilton
Technical Leads
Blue Gecko, Inc.
Evidence-based Best Practices for JD Edwards EnterpriseOne

• Measurable results with Real Application Testing
• WTS does JD Edwards EnterpriseOne well
• Blue Gecko does Oracle well
• Together we set out to demonstrate how to use concrete data from real-life production workloads to tune EnterpriseOne on Oracle
Why Tune JD Edwards EnterpriseOne on Oracle?

• Tuning leads to less time spent waiting for the database
• And Less time spent waiting for the database leads to:
  • Better user experience
  • Improved employee productivity
  • Allows for company growth with existing IT resources
  • Lower total cost of ownership
Why Real Application Testing?

• With Real Application Testing, Oracle gives us the ability to empirically determine the best way to tune the database.

• Empirically derived values are better than:
  • Guessing
  • Estimates
  • Rules of thumb
What will you get from this presentation?

• Technical understanding of the technologies we used in this experiment

• A idea of the results you can expect from your partnership with WTS and Blue Gecko.
Why do this experiment?

- JD Edwards is now an Oracle product
- Dearth of advice for tuning JD Edwards on Oracle RDBMS
- WTS excels in JD Edwards on Oracle
  - App-specific database modifications for tuning
  - High availability, scalability
- Blue Gecko excels in Oracle tuning, support and troubleshooting
  - Service time reduction
  - Problem root cause identification
Ivan Oss

- WTS Application Engineering Manager
- 12 Years managing development operations at JDEdwards
- JDE's 3rd Oracle DBA
- Contributing author and technical editor for several bestselling books on Oracle Databases

Dallas Willett

- Blue Gecko
- Oracle DBA for 14 years
- Amazon.com’s second DBA 1997-2006

Jeremiah Wilton

- Blue Gecko
- Oracle DBA for 17 years
- Amazon.com’s first DBA 1997-2004
- Member, Oak Table
Ivan Oss

- WTS Application Engineering Manager
- 12 Years managing development operations at JDEdwards
- JDE's 3rd Oracle DBA
- Contributing author and technical editor for several bestselling books on Oracle Databases

Dallas Willett

- Blue Gecko
- Oracle DBA for 14 years
- Amazon.com’s second DBA 1997-2006

Jeremiah Wilton

- Blue Gecko
- Oracle DBA for 17 years
- Amazon.com’s first DBA 1997-2004
- Member, Oak Table
WTS Corporate Overview

• Largest provider of hosted applications, managed disaster recovery, and infrastructure services for JD Edwards applications
• Co-founded with JD Edwards in 1997
• Over 70 hosted and disaster recovery customers
• Support 10,000+ users across 17 time zones
• Visit WTS at Booth #615 or www.wts.com
Ivan Oss

- WTS Application Engineering Manager
- 12 Years managing development operations at JDEdwards
- JDE's 3rd Oracle DBA
- Contributing author and technical editor for several bestselling books on Oracle Databases

Dallas Willett

- Blue Gecko
- Oracle DBA for 14 years
- Amazon.com’s second DBA 1997-2006

Jeremiah Wilton

- Blue Gecko
- Oracle DBA for 17 years
- Amazon.com’s first DBA 1997-2004
- Member, Oak Table
Blue Gecko, Inc

- Remote DBA Services
  - Oracle Database
  - Oracle E-Business Suite
  - MySQL
  - Microsoft SQL Server
- 24x7 Support and Proactive monitoring
- Highly competent
  - Many ex-Amazon DBAs
Our Experiment...

• Purpose:
  • Tune JD Edwards EnterpriseOne on Oracle

• Method:
  • Use Oracle Real Application Testing, which will allow us to gather measurable results
What are we measuring?

• DB time used by a real-life application
  • Not buffer gets
  • Not CPU used
  • Not disk reads
  • Not number of … anything

• Just DB time by the application
The Components of our Experiment...

- Windows Server 2003 R2 64-bit (5.2)
- JD Edwards EnterpriseOne
- Oracle Database 11gR2 64-bit (11.2.0.2)
  - Oracle Real Application Testing
    - Database Replay
    - SQL Performance Analyzer (SPA)
  - Oracle Diagnostic and Tuning Packs
    - SQL Tuning Advisor
    - Automatic Workload Repository (AWR)
    - Automatic Database Diagnostic Monitor (ADDM)
The Components of our Experiment...

- Windows Server 2003 R2 64-bit (5.2)
- JD Edwards EnterpriseOne
- Oracle Database 11gR2 64-bit (11.2.0.2)
  - Oracle Real Application Testing
    - Database Replay
    - SQL Performance Analyzer (SPA)
  - Oracle Diagnostic and Tuning Packs
    - SQL Tuning Advisor
    - Automatic Workload Repository (AWR)
    - Automatic Database Diagnostic Monitor (ADDM)
The Components of our Experiment...

• Windows Server 2003 R2 64-bit (5.2)
• JD Edwards EnterpriseOne
• Oracle Database 11gR2 64-bit (11.2.0.2)
  • Oracle Real Application Testing
    • Database Replay
    • SQL Performance Analyzer (SPA)
  • Oracle Diagnostic and Tuning Packs
    • SQL Tuning Advisor
    • Automatic Workload Repository (AWR)
    • Automatic Database Diagnostic Monitor (ADDM)
The Components of our Experiment...

- Windows Server 2003 R2 64-bit (5.2)
- JD Edwards EnterpriseOne
- Oracle Database 11gR2 64-bit (11.2.0.2)
  - Oracle Real Application Testing
    - Database Replay
    - SQL Performance Analyzer (SPA)
  - Oracle Diagnostic and Tuning Packs
    - SQL Tuning Advisor
    - Automatic Workload Repository (AWR)
    - Automatic Database Diagnostic Monitor (ADDM)
Real Application Testing

• Please do not call it RAT (per Oracle)
• Added-cost feature set for guaranteeing stable changes

• Two mainly unrelated components
  • Database Replay (DB Replay)
    • Captures and replays a faithful production workload
    • Detects performance differences across changes
    • This is the component we used for this experiment
  • SQL Performance Analyzer (SPA)
    • Benchmarks selected sets of SQL statements across changes
    • Detects regressions in individual SQL statements
Real Application Testing

- Please do not call it RAT (per Oracle)
- Added-cost feature set for guaranteeing stable changes
- Two mainly unrelated components
  - Database Replay (DB Replay)
    - Captures and replays a faithful production workload
    - Enables the detection performance differences across changes
    - This is the component we used for this experiment
  - SQL Performance Analyzer (SPA)
    - Benchmarks selected sets of SQL statements across changes
    - Detects regressions in individual SQL statements
How DB Replay works

Step 1: Capture Production Workload
Step 2: Setup Clone Database
Step 3: Prepare Cloned Database
Step 4: Baseline Test Run
Step 5: Subsequent Test Runs
How DB Replay works

Step 1: Capture Production Workload

- Capture a workload to files
  - Running production system
  - Typical (or peak) workload period
  - Oracle 9.2.0.8, 10.2.0.3-5, 11g
How DB Replay works

Step 2: **Setup Clone Database**

- Clone production database using RMAN
- Recover until the time/SCN at which the workload capture began
How DB Replay works

Step 3: Prepare Cloned Database

- Upgrade the clone to 11g
  - If production wasn’t 11g already
- Create a Flashback Database Restore Point
- Preprocess the captured workload files
How DB Replay works

Step 4: **Baseline Test Run**

- Replay the workload against the clone database
- First run used as a baseline
- Capture performance data.
  - Take AWR Snapshots during the test
  - Analyze performance and start tuning
How DB Replay works

Step 5: Subsequent Test Runs

- Implement tuning changes
- Flashback Database
  - Or re-clone to the workload start time
- Replay the workload
- Compare AWR statistics between runs
How SQL Tuning Advisor works

Step 1: Take AWR Snapshots
Step 2: Run SQL Tuning Advisor
Step 3: Tune Database
How SQL Tuning Advisor works

Step 1: Take AWR Snapshots

- Take AWR snapshots while DB Replay is running
  - Pick a begin time and end time for your snapshots
  - Write code such that the snapshots are taken at the same time during each replay
How SQL Tuning Advisor works

Step 2: Run SQL Tuning Advisor

• Create a SQL Set from your two AWR Snapshots
  • Order SQL Statements by elapsed time
  • Specify the number of SQL statements to analyze (Top-N)
• Create and execute a Tuning Task over this SQL Set
How SQL Tuning Advisor works

Step 3: **Tune Database**

- Run a report on your tuning task
- SQL Tuning Advisor recommends
  - New SQL profiles
  - Index changes
Our Setup...

- Collected production workload on Oracle 10gR2
- Cloned the production database to similar hardware
- Upgraded the clone to 11gR2
Our Tests...

• We ran the workload multiple times with Database Replay to produce three test results:

  (Test A) Baseline E1 Database

• Some additional tuning/enhancements from what you would get from the JD Edwards Deployment Tool
  • Appropriately sized SGA and other Oracle initialization parameters
  • ASM for datafiles
  • Relevant OS and Database patching
Our Tests...

(Test B) With WTS database customizations for E1

• Exactly like production
• Has additional Indexing and SQL Profiles not found in the baseline E1 database.

(Test C) With WTS database customizations for E1 plus Blue Gecko advanced tuning

• Even more Indexing and SQL Profiles
• Multiple tuning passes with DB Replay
## Our Tests...

<table>
<thead>
<tr>
<th>Database Replays</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Baseline E1 DB (Test A)</td>
<td>Removed WTS Customizations</td>
</tr>
<tr>
<td>2) WTS Customizations</td>
<td>Same as Production</td>
</tr>
<tr>
<td></td>
<td>(Test B)</td>
</tr>
<tr>
<td>3) Blue Gecko Run</td>
<td>Tuned Top-100 SQLs from Run2</td>
</tr>
<tr>
<td>4) Blue Gecko Run</td>
<td>Tuned Top-100 SQLs from Run3</td>
</tr>
<tr>
<td>5) Blue Gecko Run (Test 3)</td>
<td>Tuned Top-100 SQLs from Run4</td>
</tr>
</tbody>
</table>
The Results...

(Test A) Baseline database created by E1
DB Time of 214 minutes

(Test B) With WTS database customizations for E1
DB Time of 151 minutes
(42% performance increase)

(Test C) With WTS database customizations for E1
plus Blue Gecko advanced tuning
DB Time of 65 minutes
(229% performance increase)
Our Process...

The DB Replay runs we executed:

Run #1: Baseline E1 DB
  (Test A)
Run #2: WTS database customizations for E1
  (Test B)
Run #3: Added SQL Profiles and Indexes from Run #2
Run #4: Added SQL Profiles and Indexes from Run #3
Run #5: Added SQL Profiles and Indexes from Run #4
  (Test C)
The Results...

<table>
<thead>
<tr>
<th>Database Replays</th>
<th>New SQL Profiles</th>
<th>New Indexes</th>
<th>DB Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) Baseline E1 DB (Test A)</td>
<td>0</td>
<td>0</td>
<td>214 min</td>
</tr>
<tr>
<td>2) WTS Customizations (Test B)</td>
<td>13</td>
<td>8</td>
<td>151 min</td>
</tr>
<tr>
<td>3) Blue Gecko Run</td>
<td>41</td>
<td>31</td>
<td>78 min</td>
</tr>
<tr>
<td>4) Blue Gecko Run</td>
<td>57</td>
<td>43</td>
<td>72 min</td>
</tr>
<tr>
<td>5) Blue Gecko Run (Test 3)</td>
<td>65</td>
<td>46</td>
<td>65 min</td>
</tr>
</tbody>
</table>
Technical Details

- Capture Script
- Replay Script
- SQL Tuning Script
- More Detailed AWR Results
Technical Details

- Replay Script
- SQL Tuning Script
- More Detailed AWR Results
Capture Script

Capture production workload

• PL/SQL or use Enterprise Manager to kick this off

SQL> create directory CAPTURE_DIR as 'f:\capture_dir';

SQL> exec dbms_workload_capture.start_capture(
   name => :capture_name,
   dir => 'CAPTURE_DIR',
   duration => 3*60*60);

SQL> exec dbms_workload_capture.finish_capture;
RMAN Clone

Setup Clone Database

```
RMAN> connect target sys/password
RMAN> connect auxiliary /
RMAN> duplicate target database to AUXSID;
```

- Clone will be specific to your environment

```
SQL> exec dbms_workload_replay.process_capture(
    capture_dir => 'CAPTURE_DIR');
```
Replay Script

Step 1: Flashback the Database
Step 2: Load Tuning Changes
Step 3: Prepare Replay
Step 4: Start Replay
Step 5: Monitor Replay

- The SQL scripts we used in this experiment and presentation are available at:
  - http://www.bluegecko.net/oracle/real-application-testing
Replay Script

Step 1: Flashback the Database

```sql
SQL> @save_sql_profiles.sql
SQL> shutdown immediate
SQL> startup mount
SQL> flashback database
to restore point pre_capture;
SQL> alter database open resetlogs;
```

- We used dbms_datapump commands to save the current SQL Profiles between tests.
- Flashback Database saved us a lot of time
Replay Script

Step 2: Load Tuning Changes

```sql
SQL> create directory workload_dir
       as 'f:\export_dir';
SQL> @load_sql_profiles.sql
SQL> @schema_changes.sql
```

- We used `dbms_datapump` to reload the SQL Profiles from previous test back into the database.
- We added new indexes suggestions from the SQL Tuning Advisor to our `schema_changes.sql` script.
Replay Script

Step 3: Prepare Replay

SQL> exec dbms_workload_replay.initialize_replay
       (replay_name=>:replay_nm,
        replay_dir=>'WORKLOAD_DIR');
SQL> @delete_old_replay_info.sql
SQL> @remap_connections.sql

• Removed old replay information
• Remap connections from production DB to clone
Replay Script

Step 3: Prepare Replay (continued…)

• Remove any delay between SQL statements

  SQL> exec dbms_workload_replay.prepare_replay
       (think_time_scale=>0);

• The default for THINK_TIME_SCALE 100 (percent)
  • Used to compare the production database to your cloned database

• We set THINK_TIME_SCALE to zero.
  • We’re not comparing to production
  • We’re doing comparisons between runs on our cloned database
Replay Script

Step 4: Start Replay

```
SQL> !nohup ./run_wrc.bash
SQL> exec dbms_workload_replay.start_replay;
```

• We kicked off 25 Workload Replay Clients (wrc)
• We ran these wrc processes on another machine
Replay Script

Step 5: Monitor Replay

SQL> @replay_minder.proc
SQL> exec dbms_scheduler.create_job
    (job_name=>'replay_minder_job',
     job_type=>'STORED_PROCEDURE',
     job_action=>'replay_minder',
     enabled=>true);

- Stored Procedure checks the progress of the replay periodically
  - Takes initial and final AWR Snapshots
  - Then terminates the replay
SQL Tuning Script

Step 1: Create a SQL Set
Step 2: Populate the SQL Set
Step 3: Create and Execute a Tuning Task
Step 4: Examine Tuning Report

• The SQL scripts we used in this experiment and presentation are available at:
  • http://www.bluegecko.net/oracle/real-application-testing
SQL Tuning Script

Step 1: Create a SQL Set

```
SQL> exec dbms_sqltune.create_sqlset
     (sqlset_name=>:sqlset_nm);
```

- To populate this new SQL Set, we used the AWR Snapshots that we took during DB Replay
SQL Tuning Script

Step 2: Populate the SQL Set

```sql
SQL> OPEN v_cursor FOR SELECT VALUE(P) FROM table
     (dbms_sqltune.select_workload_repository(
       begin_snap=>v_begin_snap,
       end_snap=>v_end_snap,
       basic_filter=>
         'parsing_schema_name <> ''SYS''',
       ranking_measure1=>'elapsed_time',
       result_limit => &v_result_limit,
       attribute_list=>''ALL'')) P;
```
SQL Tuning Script

Step 2: Populate the SQL Set (continued…)

```
SQL> exec dbms_sqltune.load_sqlset(
      sqlset_name => :sqlset_nm,
      populate_cursor => v_cursor,
      load_option => 'MERGE',
      update_option => 'ACCUMULATE');
```
SQL Tuning Script

Step 3: Create and Execute a Tuning Task

```sql
SQL> exec :tuning_task :=
    dbms_sqltune.create_tuning_task(
        sqlset_name => :sqlset_nm,
        time_limit => &v_dur_mins*60);

SQL> exec dbms_sqltune.execute_tuning_task
    (:tuning_task);
```
SQL Tuning Script

Step 4: Examine Tuning Report

```sql
SQL> select dbms_sqltune.report_tuning_task (:tuning_task) from dual;
SQL> select dbms_sqltune.script_tuning_task (:tuning_task) from dual;
```
## SQL Tuning Advisor Output

- Query of DBA_SQLSET_STATEMENTS

<table>
<thead>
<tr>
<th>SQL_ID</th>
<th>ELAPSED_TIME</th>
<th>EXECUTIONS</th>
<th>OPTIMIZER_COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>343px2czg00br</td>
<td>406061746</td>
<td>2</td>
<td>24151</td>
</tr>
<tr>
<td>0bbd27chkkw0s</td>
<td>350630523</td>
<td>29</td>
<td>225</td>
</tr>
<tr>
<td>3tyyq6gm3zs0j</td>
<td>315514346</td>
<td>1</td>
<td>1903</td>
</tr>
<tr>
<td>1dumbdy32gvdx</td>
<td>279378002</td>
<td>32</td>
<td>1420</td>
</tr>
<tr>
<td>4mp577zznhjju</td>
<td>172512711</td>
<td>7</td>
<td>90634</td>
</tr>
</tbody>
</table>
Global SQL Tuning Result Statistics

Number of SQLs Analyzed : 5
Number of SQLs in the Report : 4
Number of SQLs with Findings : 4
Number of SQLs with SQL profiles recommended : 3
Number of SQLs with Index Findings : 3
Number of SQLs with Timeouts : 1
# SQL Tuning Advisor Output

## SQLs with Findings Ordered by Maximum (Profile/Index)

<table>
<thead>
<tr>
<th>object ID</th>
<th>SQL ID</th>
<th>profile (benefit)</th>
<th>index (benefit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>343px2czg00br</td>
<td>99.32%</td>
<td>99.95%</td>
</tr>
<tr>
<td>4</td>
<td>3tyyq6gm3zs0j</td>
<td>51.78%</td>
<td>98.16%</td>
</tr>
<tr>
<td>3</td>
<td>0bbd27chkw0s</td>
<td></td>
<td>97.33%</td>
</tr>
<tr>
<td>6</td>
<td>4mp577zznhjju</td>
<td></td>
<td>93.55%</td>
</tr>
</tbody>
</table>
SQL Tuning Advisor Output

• Example SQL Profile Recommendation

Recommendation (estimated benefit: 99.32%)

Consider accepting the recommended SQL profile.

execute dbms_sqltune.accept_sql_profile(
    task_name => 'TASK_367582', object_id => 2,
    task_owner => 'SYS', replace => TRUE);
### SQL Tuning Advisor Output

- **Example SQL Profile Recommendation**

**DETAILS SECTION**

<table>
<thead>
<tr>
<th>Original Plan</th>
<th>SQL Profile</th>
<th>% Improved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed Time: 15.51271</td>
<td>.028304</td>
<td>99.81 %</td>
</tr>
<tr>
<td>CPU Time: .59375</td>
<td>.003125</td>
<td>99.47 %</td>
</tr>
<tr>
<td>User I/O Time: 15.19699</td>
<td>.026501</td>
<td>99.82 %</td>
</tr>
</tbody>
</table>
SQL Tuning Advisor Output

• Example Index Recommendation

  Recommendation (estimated benefit: 99.95%)
  ---------------------------------------------------------------------
  - Consider running the Access Advisor to improve the physical schema design or creating the recommended index.

  create index IDX$$_59BD60018 on F470371("SZDOCO","SZLITM");
SQL Tuning Advisor Output

- We decided not to accept any parallel query SQL profiles

Recommendation (estimated benefit: 93.06%)

------------------------------------------
- Consider accepting the recommended SQL profile to use parallel execution for this statement.

execute dbms_sqltune.accept_sql_profile(
  task_name => 'TASK_367574', object_id => 38,
  task_owner => 'SYS', replace => TRUE,
  profile_type => DBMS_SQLTUNE.PX_PROFILE);
The Results...

(Test A) Baseline database created by E1
DB Time of 214 minutes

(Test B) With WTS database customizations for E1
DB Time of 151 minutes
(42% performance increase)

(Test C) With WTS database customizations for E1
plus Blue Gecko advanced tuning
DB Time of 65 minutes
(229% performance increase)
### The results continued...

**Additional AWR Data:**

<table>
<thead>
<tr>
<th>Test</th>
<th>Test Time</th>
<th>SQL Profiles</th>
<th>Indexes</th>
</tr>
</thead>
<tbody>
<tr>
<td>A  Baseline E1</td>
<td>214 min</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B  WTS Customizations</td>
<td>151 min</td>
<td>13</td>
<td>8</td>
</tr>
<tr>
<td>C  Blue Gecko Advanced Tuning</td>
<td>65 min</td>
<td>65</td>
<td>46</td>
</tr>
</tbody>
</table>
The results continued...

Additional AWR Data:

<table>
<thead>
<tr>
<th>Test</th>
<th>DB Time</th>
<th>CPU Time</th>
<th>IO Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A  Baseline E1</td>
<td>214 min</td>
<td>65 min</td>
<td>147 min</td>
</tr>
<tr>
<td>B  WTS Customizations</td>
<td>151 min</td>
<td>101 min</td>
<td>48 min</td>
</tr>
<tr>
<td>C  Blue Gecko Advanced Tuning</td>
<td>65 min</td>
<td>38 min</td>
<td>24 min</td>
</tr>
</tbody>
</table>
The results continued...

Additional AWR Data:

<table>
<thead>
<tr>
<th>Test</th>
<th>I/O Reads</th>
<th>I/O Writes</th>
</tr>
</thead>
<tbody>
<tr>
<td>A  Baseline E1</td>
<td>161 GB</td>
<td>9 GB</td>
</tr>
<tr>
<td></td>
<td>590 req/sec</td>
<td>120 req/sec</td>
</tr>
<tr>
<td>B  WTS Customizations</td>
<td>116 GB</td>
<td>9 GB</td>
</tr>
<tr>
<td></td>
<td>520 req/sec</td>
<td>140 req/sec</td>
</tr>
<tr>
<td>C  Blue Gecko Advanced Tuning</td>
<td>54 GB</td>
<td>6 GB</td>
</tr>
<tr>
<td></td>
<td>135 req/sec</td>
<td>150 req/sec</td>
</tr>
</tbody>
</table>
The results continued...

Additional AWR Data:

<table>
<thead>
<tr>
<th>Test</th>
<th>DB Time</th>
<th>Clock Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Baseline E1</td>
<td>209 min</td>
<td>113 min</td>
</tr>
<tr>
<td>B WTS Customizations</td>
<td>163 min</td>
<td>104 min</td>
</tr>
<tr>
<td>C Blue Gecko</td>
<td>65 min</td>
<td>87 min</td>
</tr>
<tr>
<td></td>
<td>Advanced Tuning</td>
<td></td>
</tr>
</tbody>
</table>
The results continued...

Additional AWR Data:

<table>
<thead>
<tr>
<th>Test</th>
<th>Buffer Cache Hit Ratio</th>
<th>Buffer Gets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A  Baseline E1</td>
<td>98%</td>
<td>208 million</td>
</tr>
<tr>
<td>B  WTS Customizations</td>
<td>96%</td>
<td>82 million</td>
</tr>
<tr>
<td>C  Blue Gecko Advanced Tuning</td>
<td>99%</td>
<td>70 million</td>
</tr>
</tbody>
</table>
The results continued...

Buffer Pool Advisory (Read Factor versus Pool Size)
Evolve with WTS and BlueGecko

• Are you using industry best practices for JDE management?
• How big is your support community?
• How many master level Oracle DBAs do you have on staff?
• How much time do you have for R&D projects?
Evidence-based Best Practices for JD Edwards EnterpriseOne

Using Oracle 11gR2
Real Application Testing

• The SQL scripts we used in this experiment and presentation are available at:
  http://www.bluegecko.net/oracle/real-application-testing

• Visit WTS at Booth #615 or www.wts.com
Did you register through Quest?
You are now a member!
Continue the education ALL YEAR:

Regional Education Conferences:
• Northeast: July 20-21
• West: Aug. 16-17
• Midwest: Nov. 7-8

Online Learning Series:
• Business Intelligence: May 11-27
• HCM: Aug. 9-25
• Financials: Sept. 8-27
• Upgrades: Oct. 13-27
• Year-End: Nov. 3-15

Plus 200+ OnDemand Online Sessions
Save by using the member rate!

www.questdirect.org