Oracle Performance Technical Notes

Last reviewed October 28, 2007

Purpose:

To provide a list of general guidelines for optimization of an Oracle database server when this server is used with the Ultimus BPM Suite. These suggestions are based upon in-house testing done within Ultimus Labs. These guidelines are in no way a complete list of all the configuration options, settings and improvements that can be applied to Oracle Database Server. For detailed information about any of the guidelines in this document, or about further optimization, please refer to your Oracle Database Server documentation and other resources.

Note: The information in this document is based on observations from testing in the Ultimus Labs. It is published purely for information. Ultimus is not responsible for any damages caused by the use of the information in this document.

Summary:

There are several key areas of performance tuning that will be discussed.

- DISK I/O
- SHARED SERVER
- MEMORY
- TABLE CACHEING.
- ANALYZE/QUERY EXECUTION

A certified on-site Oracle DBA should evaluate these configuration guidelines. Impact on other applications should be considered and performance improvements, while observed by Ultimus in the Ultimus Labs, are not guaranteed. Ultimus is not responsible for any damages caused by implementing these guidelines.

Our tests have shown significant improvement in performance of the entire Ultimus BPM Suite when the Oracle Server is optimized. These recommendations are to be used as a general guide only. Each individual configuration will differ according to the resources available on the Oracle Server machine.

Guidelines:

Disk I/O

DISK STRIPING

When data is not available in memory, it must be read from the data files located on the hard drive. As the number of reads to the physical drive increase and multiple processes try to access the same disk, contention occurs. Balancing the Oracle db files across multiple drives can decrease this contention. In turn, performance will increase.

One way to accomplish this is by using disk striping. Disk striping will balance the reads and writes to multiple drives therefore increasing performance. This can be setup using the Disk Administrator provided by NT4.0 and Win2000. Consult the Microsoft Help for this configuration.

Once the striped disks are configured, create the database on the striped drive. Place all of the database files on this drive except for the redo logs. These can be left on a separate drive.

There are alternate ways to setup Oracle on multiple drives without using striped drives. However, these methods will not be discussed. There are many resources available for performing this type of setup.

Shared Server

Your Oracle database can be created using either a “Dedicated Server Mode” or a “Shared Server Mode”. Select the Shared Server Mode. This will allow pre-spawned connections to be waiting database calls, thus decreasing the total connection time applications take when connecting to Oracle via ODBC. The Oracle multi-threaded server feature will be enabled.

After this is checked and several setup screens later, the option to select Connection Pooling will appear. Select “ENABLE”.

Memory

The second area to tune will be the memory. The memory settings can be found in the init.ora or can be tuned at the time the database is created. If your database has already been created use the following query to return the current init.ora parameters.

```
SELECT          name, value
FROM            v$parameter
Where   name in ('db_block_buffers', 'db_block_size',
                'shared_pool_size', 'sort_area_size');
```

DB_BLOCK_SIZE

The best setting for DB_BLOCK_SIZE is operating system dependant. The values can range from 2k to 32k for any OS. For NT or Win2000 create this using a 4K Block size. This setting must be done during the creation of the database.

DB_BLOCK_BUFFERS

This is the most important parameter in the init.ora. This setting X DB_BLOCK_SIZE should be 25-50% of available memory, but is limited by number of users and amount of memory on the server. If the system contains several hundred users +, this percentage might have to be lowered until additional memory can be added to the server. Setting this too high will consume the available system resources. But setting too low will cause performance problems. The following script will determine if the data block buffers setting is high enough. The buffer hit ratio should be 95% or higher. However, increasing the ratio from 95%- 99% can yield performance gains of over 400%.

```
SELECT (1 – (sum (decode (name, 'physical reads',
        Value, 0)) / (sum(decode(name, db block gets',
        Value, 0)) + sum(decode(name, 'consistent gets',
        Value, 0))))) * 100 "Hit Ratio"
FROM v$sysstat;
```

If this setting is too low increasing the DB_BLOCK_BUFFERS will increase the percentage. Additional memory may need to be added to allow the hit ratio to remain at the recommended level while not consuming all of the memory on the system.

SHARED_POOL_SIZE

This is the memory allocated for the library and data dictionary cache. Determine the dictionary hit ratio by executing the following query. The hit ratio should be greater than 95%. If this setting is too low, increase the SHARED_POOL_SIZE parameter in the init.ora.

```
SELECT (1 – (sum (getmisses) / sum(gets)))
        * 100 “Hit Ratio”
FROM v$rowcache;
```

Determine the library cache hit ratio by executing the following query. This hit ration should also be greater than 95%. Increasing the SHARED_POOL_SIZE will increase the hit ratio.

```
SELECT sum(pins) / (sum(pins) + sum(reloads)) *
        100 “Hit Ratio”
FROM v$librarycache;
```

SORT_AREA_SIZE

This is the memory allocated for sorting. Memory for sorting is contained in the systems main memory outside of Oracle. This is setting is on a per-user basis and is returned in bytes. Setting this too high can consume the available system resources. Setting this to low can cause disk swapping. Run the following query to determine the number of disk sorts versus the number of memory sorts.

```
SELECT a.value “Disk Sorts”, b.value “Memory Sorts”,
        Round ((100*b.value)/decode ((a.value+b.value), 0,1, (a.value + b.value)),2)
        “Pct Memory Sorts”
FROM v$sysstat a, v$sysstat b
WHERE a.name = ‘sorts (disk)’ AND b.name =‘sorts (memory)’;
```

Cache Tables

It is also beneficial to cache small, frequently used tables. Some of the suggested ones are:

- **INCIDENT**
- **MAPS**
- **DEPARTMENTS**
- **GROUPS**
- **JOBS**

These tables can be set to cache using DBA Studio. To perform this operation, click on SCHEMA from the desired database in DBA Studio. Select TABLE, the correct SCHEMA, then the tables listed above (one at a time). Click on the Options tab. Check the "Place frequently accessed data to the top of the buffer cache.(CACHE)". This can also be accomplished by running the following query:

```
ALTER TABLE "table name"
CACHE;
```

Analyze/Query Execution

Once Ultimus has been loaded and processes have been running, run the following query to analyze the schema. Make sure that the 'USERNAME is in UPPERCASE.

```
Execute dbms_utility.analyze_schema('USERNAME', 'COMPUTE');
```

Followed by:

```
Alter system flush shared_pool;
```

This will give the Oracle cost based optimizer statistics to generate better execution plans. Running this on a regular basis will allow Oracle to update the execution path to achieve the best possible performance.