

VERITAS Database Edition 3.5 for Oracle

**PERFORMANCE BRIEF – OLTP COMPARISON
FOR SOLARIS**

SEPTEMBER 25, 2002

INTRODUCTION

This document describes the performance of VERITAS Database Edition 3.5 *for Oracle* in a 64-bit Solaris 9 operating environment as measured by an Online Transaction Processing (OLTP) workload. The purpose of this brief is to illustrate the impact of different I/O and memory configurations to the database performance. The benchmark used for this performance comparison was derived from the commonly known TPC-C benchmark that comprises a mixture of read-only and update intensive transactions that simulate a warehouse supplier environment. (Details on this benchmark can be obtained from the Transaction Processing Council's web page at <http://www.tpc.org>.)

The Database Edition 3.5 comprises the following components:

VERITAS Volume Manager™ (VxVM®) 3.5, patch 1

VERITAS File System™ (VxFS®) 3.5 (including Quick I/O, Cached Quick I/O, and Storage Checkpoints)

VERITAS Extension for Oracle Disk Manager™ 3.5

TEST CONFIGURATION

HARDWARE TEST ENVIRONMENT

The OLTP benchmark tests were conducted on a Sun Microsystems Ultra Enterprise 10000 domain with 13 processors and 10 GB of memory. The UE 10000 system was attached with six 10-bay CLARiiON DAE (Disk Array Enclosure) JBOD racks via six Sun™ StorEdge™ Sbus FC-100 Host Adapters. Each CLARiiON DAE contained 10 18-GB Seagate drives (10,000 RPM).

SOFTWARE TEST ENVIRONMENT

The following software releases were used in the tests:

VERITAS Database Edition 3.5 for Oracle

Oracle 9iR2 (release 9.2.0.1, 64-bit)

Solaris 9 (release 4/02, 64-bit)

The file systems under test were configured over one 58-way striped volume of 200 GB built on 58 disks on three controllers. All file systems were built on this striped volume. The Oracle redo logs were created on a single drive in one of the CLARiiON racks. For the raw I/O configuration, all the Oracle files except redo logs were striped 58-way on the same set of drives to ensure equal drive usage. VERITAS Volume Manager was used to create the striped volumes for all configurations.

The size of the database used for the test is 130 GB with a total of 81 Oracle datafiles, including redo logs, indexes, rollback segments, and temporary and user tablespaces. The size of the database is that of a fully scaled TPC-C database with a scale factor of 1,000 warehouses.

BENCHMARK TESTS

The benchmark tests were conducted with 1 to 8GB of Oracle buffer cache and in eight I/O configurations:

- RAW – uses the VERITAS Volume Manager raw volumes directly,
- QIO – uses the Quick I/O feature of VERITAS File System,
- CQIO – uses the Cached Quick I/O feature of VERITAS File System,
- ODM – uses the Oracle Disk Manager I/O feature of VERITAS File System¹,
- DIO – uses the VERITAS File System direct I/O mode²,
- BIO – uses the default VERITAS File System buffered I/O mode,

The Oracle block size used for all these tests was 2K. During the test, Oracle statistics, Volume Manager statistics, Quick I/O statistics, and ODM I/O statistics were gathered in addition to the benchmark throughput numbers. The system parameters and the Oracle parameters used in the benchmark tests are provided in Appendix A and B.

¹ Oracle Disk Manager is a disk management interface that enhances file management and disk I/O throughput in a database environment.

The ODM Application Programming Interface (API) is defined by Oracle and first introduced in Oracle9i. VERITAS Extension for Oracle Disk Manager provides a dynamically-load library and a kernel driver to support the ODM API in Oracle9i.

² VxFS DIO mode uses the mount option “-o convosync=direct.”

RESULTS AND ANALYSIS

The primary performance metric used in this brief is a throughput metric that measures the number of transactions completed per minute (TPM). The transaction mix in this OLTP benchmark represents the processing of an order as it is entered, paid for, checked, and delivered, following the model of a complete business activity. The TPM metric is, therefore, considered a measure of business throughput. Table 1 lists the database throughput of the benchmark tests driven by 50 batch users in different I/O configurations and various Oracle buffer cache sizes.

DATABASE THROUGHPUT RELATIVE TO RAW I/O

The relative plot of database throughput compared to Raw I/O is shown in Figure 1.

Table 1 - Database throughput of the benchmark tests.

Database Throughput in Transactions per Minute (TPM)								
I/O Configuration	Size of Oracle Buffer Cache							
	1GB	2GB	3GB	4GB	5GB	6GB	7GB	8GB
Raw I/O	8,283	10,424	11,825	12,777	13,434	13,880	14,181	14,405
Quick I/O	8,229	10,398	11,784	12,742	13,413	13,884	14,204	14,339
Cached Quick I/O	11,109	12,292	12,789	13,131	13,003	13,309	13,480	13,370
Oracle Disk Manager I/O	8,028	10,019	11,201	12,163	12,841	13,223	13,572	13,726
VxFS direct I/O	4,081	4,805	5,418	5,840	5,838	5,833	6,054	6,476
VxFS buffered I/O	2,223	2,518	2,447	2,431	2,909	4,338	5,038	4,487

Figure 1 - Relative plot of database throughput compared to Raw I/O.

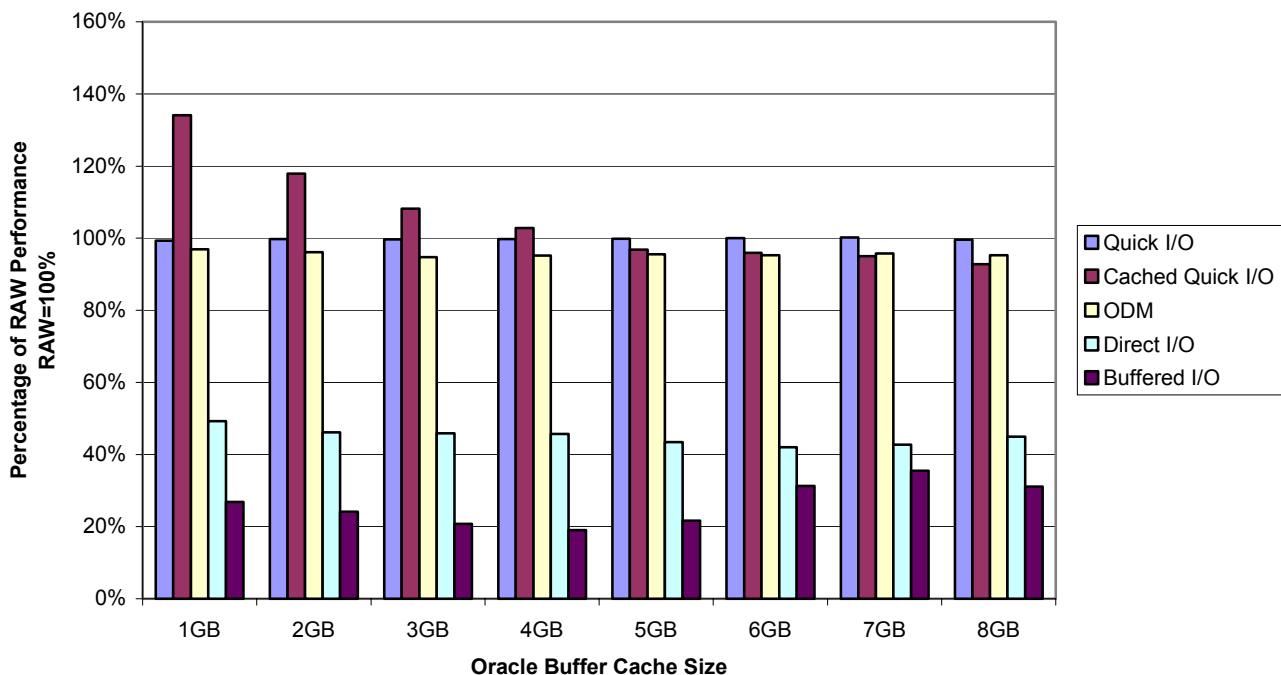
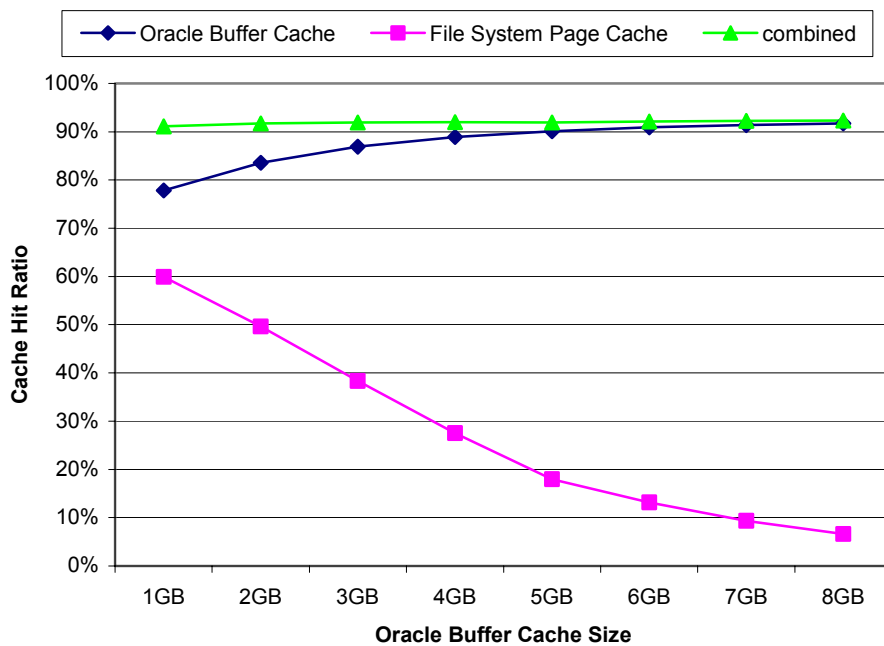


Figure 1 shows that the database throughput with Quick I/O matches closely with that of the Raw I/O. This reaffirms the excellence of Quick I/O for achieving raw I/O-equivalent performance while still providing file system manageability to Oracle databases.

USE VERITAS QUICK I/O OR ODM FOR 64-BIT ORACLE ENVIRONMENTS

Figure 1 also shows that Cached Quick I/O can only outperform Quick I/O when Oracle buffer cache is 4GB or less in this 10GB memory database server. Cached Quick I/O uses operating system memory, external to Oracle SGA, as a second level cache to buffer Oracle databases. The effectiveness of the second level cache is thus determined by the cache hit ratio of the file system page cache. Figure 2 shows the cache hit ratio of these two caches individually and all together (marked as *combined*) with different Oracle buffer cache sizes. It is interesting to observe that moving memory between these two caches only marginally affects the *combined* cache hit ratio. However, Table 1 shows that the database throughput of Cached Quick I/O keep improving as the size of Oracle buffer cache increases. Therefore, we recommend Quick I/O or ODM I/O over Cached Quick I/O in 64-bit Oracle environments because Oracle buffer cache is more efficient for buffering Oracle databases than the file system page cache.

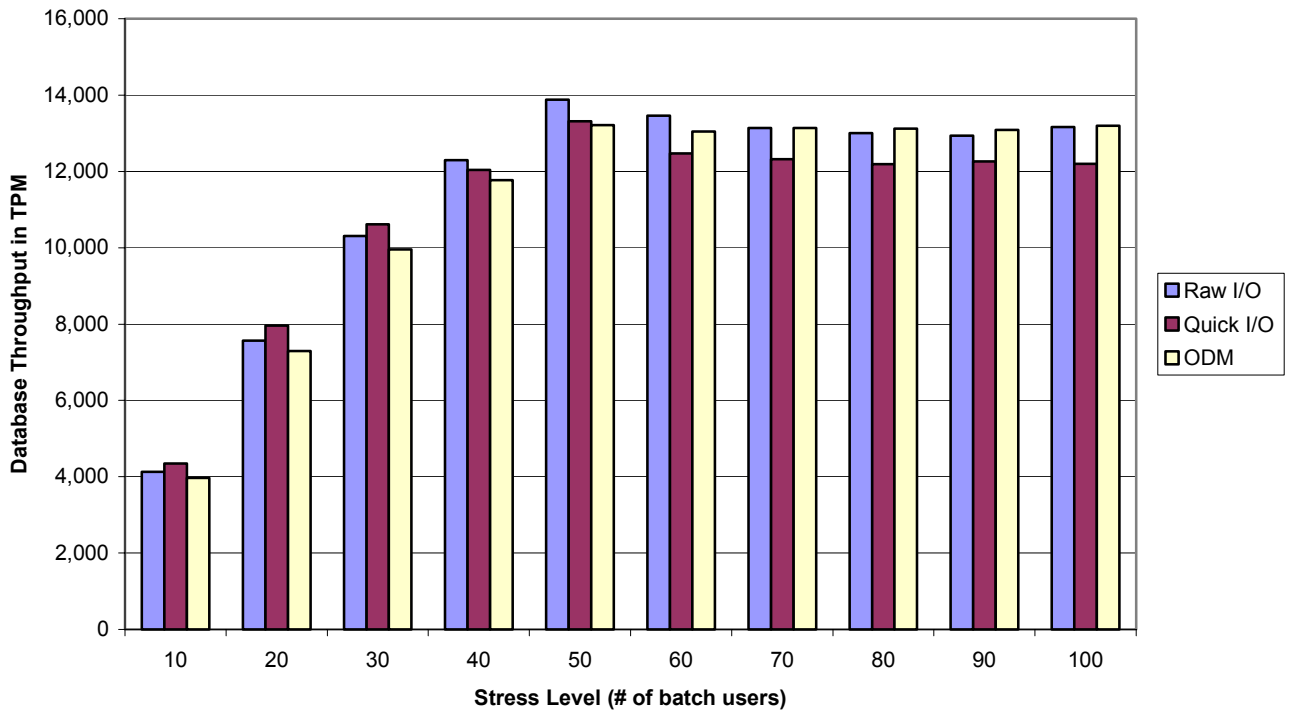
Figure 2 shows the cache effectiveness of double buffering in Cached Quick I/O.



ODM PERFORMANCE SCALES WITH INCREASING NUMBER OF USERS

Oracle Disk Manger I/O, although only available to Oracle9i, also achieves database throughput close to Raw I/O as shown in Figure 1.³ The superiority of ODM becomes more obvious when the database load increases. Figure 3 shows that ODM is the only I/O configuration that can keep up with the Raw I/O at higher stress levels.

Figure 3 - Database throughput measured at different stress levels.



³ The small performance difference between Quick I/O and ODM I/O is caused by an Oracle 9iR2 bug. Due to this bug, Oracle issues too many unnecessary ODM calls for polling I/O completion. This bug has been identified as cross platform and will be fixed in the next patch release of Oracle 9iR2 - 9.2.0.2 .

SUMMARY

The OLTP benchmark used in this study is commonly used to evaluate database performance of specific hardware and software configurations. By normalizing the system configuration and varying the file system I/O configuration, it was possible to study the impact of various storage layouts on database performance with this benchmark.

The OLTP performance measurements illustrate that the Quick I/O and Oracle Disk Manager features enable the VERITAS Database Edition *for Oracle* to have equal performance compared to raw partition configurations. As the previous studies reported, this performance superiority remains the same no matter which Oracle release (32-bit Oracle or 64-bit Oracle) or which Solaris 8/9 flavor (32-bit or 64-bit) was used.⁴

When 32-bit Oracle is used, we can only allocate up to 4GB of operating system memory to Oracle SGA. For large memory systems, VERITAS Cached Quick I/O is able to utilize the memory beyond the 4GB Oracle SGA limit as a second level to Oracle databases. The second level cache improves Oracle read performance, when data blocks are not cached in the Oracle buffer cache, but in the file system page cache. With 64-bit Oracle, the benefit of the second level cache diminishes quickly as more memory is moved to the Oracle SGA.

⁴ See Performance Brief of VERITAS Database Edition 2.1.1, 2.2, and 3.0 *for Oracle*.

APPENDIX A

The following `/etc/system` file was used to configure Solaris 9 for the benchmark tests.

```
*ident "@(#)system 1.18 97/06/27 SMI" /* SVR4 1.5 */
*
* SYSTEM SPECIFICATION FILE
*
* moddir:
*
* Set the search path for modules. This has a format similar to the
* csh path variable. If the module isn't found in the first directory
* it tries the second and so on. The default is /kernel /usr/kernel
*
* Example:
*     moddir: /kernel /usr/kernel /other/modules
*
* root device and root filesystem configuration:
*
* The following may be used to override the defaults provided by
* the boot program:
*
* rootfs:          Set the filesystem type of the root.
*
* rootdev:        Set the root device. This should be a fully
*                 expanded physical pathname. The default is the
*                 physical pathname of the device where the boot
*                 program resides. The physical pathname is
*                 highly platform and configuration dependent.
*
* Example:
*     rootfs:ufs
*     rootdev:/sbus@1,f8000000/esp@0,800000/sd@3,0:a
*
* (Swap device configuration should be specified in /etc/vfstab.)
*
* exclude:
*
* Modules appearing in the moddir path which are NOT to be loaded,
* even if referenced. Note that `exclude' accepts either a module name,
* or a filename which includes the directory.
*
* Examples:
*     exclude: win
*     exclude: sys/shmsys
*
* forceload:
*
* Cause these modules to be loaded at boot time, (just before mounting
* the root filesystem) rather than at first reference. Note that
* forceload expects a filename which includes the directory. Also
* note that loading a module does not necessarily imply that it will
* be installed.
```

```

*
*   Example:
*       forceload: drv/foo

* set:
*
*   Set an integer variable in the kernel or a module to a new value.
*   This facility should be used with caution.  See system(4).
*
*   Examples:
*
*   To set variables in 'unix':
*
*       set nautopush=32
*       set maxusers=40
*
*   To set a variable named 'debug' in the module named 'test_module'
*
*       set test_module:debug = 0x13

*set physmem=0x80000

*
*   Increase file descriptor limit
*
set rlim_fd_cur=1024
set pt_cnt=1024

*
*   Oracle Shared Memory and Operation Parameters
*
*   for 32-bit Solaris
*
*set shmsys:shminfo_shmmax=4294967295
*
*   for 64-bit Solaris
*
set shmsys:shminfo_shmmax= 10000000000
set shmsys:shminfo_shmmin=1
set shmsys:shminfo_shmmni=100
set shmsys:shminfo_shmseg=10
set semsys:seminfo_semmni=100
set semsys:seminfo_semmsl=100
set semsys:seminfo_semmns=420
set semsys:seminfo_semopm=100
set semsys:seminfo_semvmx=32767

* vxvm_START (do not remove)
forceload: drv/vxdmp
forceload: drv/vxio
forceload: drv/vxspec
* vxvm_END (do not remove)

* vxfs_START -- do not remove the following lines:
* VxFS requires a stack size greater than the default 8K.
* The following value allows the kernel stack size to be

```

```
* increased to 24K for Solaris 7, 8 and 9, and 16K for Solaris 2.6.  
set lwp_default_stksize=0x6000  
* vxfs_END
```

```
* vxfs_START -- do not remove the following lines:  
* VxFS requires a stack size greater than the default 8K.  
* The following value allows the kernel stack size to be  
* increased to 24K for Solaris 7, 8 and 9, and 16K for Solaris 2.6.  
set rpcmod:svc_default_stksize=0x6000  
* vxfs_END
```

Appendix B

The following *init.ora* file was used in the benchmark tests. The `db_block_buffers` parameter was changed for different Oracle buffer sizes.

```
control_files                = ?/dbs/tpcc_disks/control_001

parallel_max_servers        = 30
recovery_parallelism        = 20
compatible                  = 8.1.5.0.0
db_name                     = tpcc
db_files                    = 200
db_file_multiblock_read_count = 32
dml_locks                   = 500
hash_join_enabled          = FALSE
log_archive_start           = FALSE
log_checkpoint_interval     = 100000000
log_checkpoints_to_alert    = TRUE
log_buffer                  = 1048576
max_rollback_segments       = 200
open_cursors                = 200
processes                   = 200
sessions                    = 600
transactions                 = 400
#distributed_transactions   = 0
transactions_per_rollback_segment = 1
shared_pool_size            = 64M
buffer_pool_recycle         = 8000
cursor_space_for_time       = TRUE
timed_statistics            = TRUE

rollback_segments           = (t_0_1, t_0_2, t_0_3, t_0_4, t_0_5, t_0_6, t_0_7, t_0_8,
t_0_9, t_0_10, t_0_11, t_0_12, t_0_13, t_0_14, t_0_15, t_0_16, t_0_17, t_0_18, t_0_19,
t_0_20, t_0_21, t_0_22, t_0_23, t_0_24, t_0_25, t_0_26, t_0_27, t_0_28, t_0_29, t_0_30,
t_0_31, t_0_32, t_0_33, t_0_34, t_0_35, t_0_36, t_0_37, t_0_38, t_0_39, t_0_40, t_0_41,
t_0_42, t_0_43, t_0_44, t_0_45, t_0_46, t_0_47, t_0_48, t_0_49, t_0_50, t_0_51, t_0_52,
t_0_53, t_0_54, t_0_55, t_0_56, t_0_57, t_0_58, t_0_59, t_0_60, t_0_61, t_0_62, t_0_63,
t_0_64, t_0_65, t_0_66, t_0_67, t_0_68, t_0_69, t_0_70, t_0_71, t_0_72, t_0_73, t_0_74,
t_0_75, t_0_76, t_0_77, t_0_78, t_0_79, t_0_80, t_0_81, t_0_82, t_0_83, t_0_84, t_0_85,
t_0_86, t_0_87, t_0_88, t_0_89, t_0_90, t_0_91, t_0_92, t_0_93, t_0_94, t_0_95, t_0_96,
t_0_97, t_0_98, t_0_99, t_0_100, t_0_101, t_0_102, t_0_103, t_0_104, t_0_105, t_0_106,
t_0_107, t_0_108, t_0_109, t_0_110, t_0_111, t_0_112, t_0_113, t_0_114, t_0_115, t_0_116,
t_0_117, t_0_118, t_0_119, t_0_120, t_0_121, t_0_122, t_0_123, t_0_124, t_0_125, t_0_126,
t_0_127, t_0_128, t_0_129, t_0_130, t_0_131, t_0_132, t_0_133, t_0_134, t_0_135, t_0_136,
t_0_137, t_0_138, t_0_139, t_0_140, t_0_141, t_0_142, t_0_143, t_0_144, t_0_145, t_0_146,
t_0_147, t_0_148, t_0_149, t_0_150, t_0_151, t_0_152, t_0_153, t_0_154, t_0_155, t_0_156,
t_0_157, t_0_158, t_0_159, t_0_160, t_0_161, t_0_162, t_0_163, t_0_164, t_0_165, t_0_166,
t_0_167, t_0_168, t_0_169, t_0_170, t_0_171, t_0_172, t_0_173, t_0_174, t_0_175, t_0_176,
t_0_177, t_0_178, t_0_179, t_0_180, t_0_181, t_0_182, t_0_183, t_0_184, t_0_185, t_0_186,
t_0_187, t_0_188, t_0_189, t_0_190, t_0_191, t_0_192, t_0_193, t_0_194, t_0_195, t_0_196,
t_0_197, t_0_198, t_0_199, t_0_200)
```