

VERITAS Database Edition™ 2.1

for Oracle®

**Performance Brief –
OLTP Comparison on Solaris 7**

N O V E M B E R 1 9 9 9


VERITAS

BUSINESS WITHOUT INTERRUPTION™



Introduction

This document describes the performance of VERITAS® Database Edition™ 2.1 *for Oracle*® on the 32-bit and 64-bit Solaris7 platforms 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. 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>.)

This release of the Database Edition supports Solaris 2.5.1, 2.6 and both 32-bit and 64-bit Solaris 7. The Database Edition 2.1 comprises the following components:

- VERITAS File System™ (VxFS®) 3.3.2 with VERITAS Quick I/O™ 3.3.2 (including Cached Quick I/O)
- VERITAS Volume Manager™ (VxVM®) 3.0.2
- VERITAS Volume Manager Storage Administrator™ 3.0.3
- VERITAS NetBackup Block-Level Incremental Backup Extension 3.2 *for Oracle*

Test Configuration

Tests were run on a Sun Microsystems Ultra Enterprise 10000 system with 16 processors and 6GB of memory. A system board with additional 4GB memory was later added to the system for additional runs. The UE 10000 system was attached with four Sun D1000 racks via 8 Sun Sbus F/W/D SCSI-2 controllers. Each D1000 rack had 12 8-GB drives and each controller had 6 drives attached to it.

The following software releases were used in testing:

- VERITAS Database Edition 2.1 *for Oracle*
- Oracle8i (Release 8.1.5, both 32-bit and 64-bit)
- Solaris 7 (both 32-bit and 64-bit)

The file systems under test were configured over one 36-way striped volume of 60 GB built on 36 disks on 6 controllers. All file systems were built on this striped volume. The Oracle redo logs were created on a single drive on a separate controller that also serviced the disk containing the Oracle executables and benchmark scripts. For the raw I/O configuration, all the Oracle files were striped 36-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 was 50 GB with a total of 59 Oracle datafiles, including redo logs, indexes, rollback segments, and temporary and user tablespaces. The size of the database was that of a fully scaled TPC-C database with a scale factor of 200 warehouses.

Tests were run on five database configurations, each with a different Oracle buffer cache size as shown. The Oracle block size used for all these tests was 2K. During the test, `sar` data (system activity reporter data), Oracle statistics, and Quick I/O statistics (using `qiostat`, where appropriate) were gathered in addition to the throughput numbers.

The system parameters and the Oracle parameters used in the benchmark are provided in Appendix A and Appendix B.

The following I/O configurations were tested:

- Raw I/O—uses the VxVM raw volumes directly
- VxFS Quick I/O—uses Quick I/O feature of VxFS
- VxFS Cached Quick I/O—uses Cached Quick I/O feature of VxFS
- VxFS buffered I/O—uses the default VxFS buffered I/O mode
- VxFS direct I/O—uses VxFS direct I/O mode
- UFS buffered I/O—uses the default UFS buffered I/O mode
- UFS direct I/O—uses the optional UFS direct I/O mode

Overview and Analysis of Results

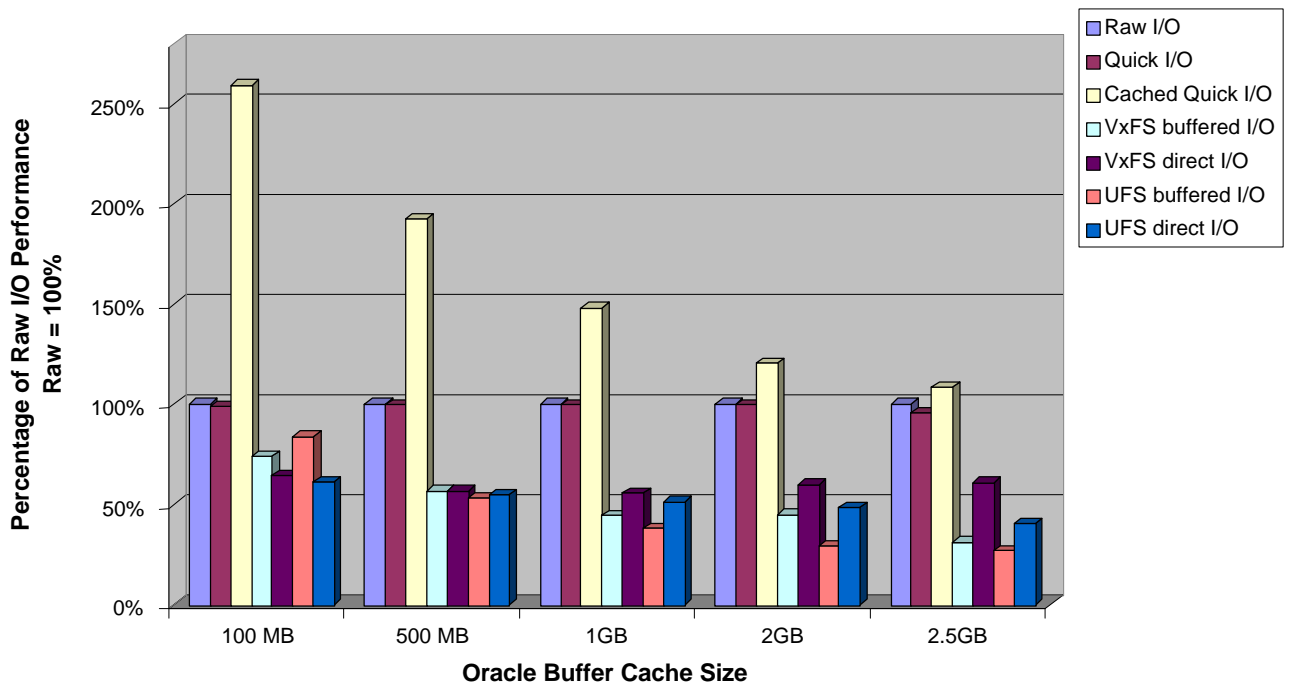
The results of the tests are presented in the tables and graphs below. The performance metric used for this comparison is the throughput rate measured by the number of transactions completed per minute (TPM). The transaction mix in this benchmark represents the processing of an order as it is entered, paid for, checked, and delivered and follows the model of a complete business activity. Therefore, the TPM metric is considered a measure of business throughput and is shown for each of the tested I/O configurations.

Table 1

Database Throughput in Transactions per minutes (TPM)					
I/O Configuration	Size of Oracle Buffers (SGA)				
	100MB	500MB	1GB	2GB	2.5GB
Raw I/O ¹	2218	4118	5683	7512	7978
Quick I/O ¹	2186	4107	5653	7498	7622
Cached Quick I/O ¹	5750	7914	8412	9031	8643
VxFS buffered I/O ¹	1646	2342	2515	3361	2487
VxFS direct I/O ¹	1429	2341	3163	4512	4855
UFS buffered I/O ²	1861	2201	2171	2195	2128
UFS direct I/O ²	1360	2250	2911	3622	3243

The relative plot of the above I/O configurations is illustrated in Figure 1 for five different Oracle SGA sizes running on Oracle 8.1.5 (32-bit) on Solaris 7 (32-bit).

Figure 1



¹ Uses asynchronous I/O (disk_asynch_io = TRUE in init.ora.) Only raw I/O and quick I/O support kernelized asynchronous I/O (KAIO) on Solaris.

² Uses synchronous I/O (disk_asynch_io = FALSE in init.ora)

The performance numbers obtained with Quick I/O track that of the raw I/O configuration in every tested configuration. This similarity is due to the performance characteristics of Quick I/O mimicking those of raw I/O.

The results also show that Cached Quick I/O delivers a large gain in performance as compared to raw I/O or Quick I/O when the file system has a large page cache. The gain is due to Cached Quick I/O allowing the large memory of the operating system (external to the Oracle SGA) to be used as a second level cache. The gain diminished as the Oracle buffer size became larger (and the operating system page cache became smaller).

32-bit OS versus 64-bit OS (32-bit Oracle)

Solaris 7 supports two flavors: 32-bit or 64-bit. The 32-bit Oracle 8.1.5 can run on both 32-bit and 64-bit Solaris 7 platforms. The following table shows the OLTP throughput when 32-bit Oracle was used with 32-bit Solaris and 64-bit Solaris under different Oracle buffer sizes. The tests were run with asynchronous I/O (`disk_asynch_io=TRUE`).

Table 2

32-bit Solaris vs. 64-bit Solaris (32-bit Oracle)						
	32-bit Solaris 7			64-bit Solaris 7		
	100MB	1GB	2.5GB	100MB	1GB	2.5GB
Raw I/O	2218	5683	7978	2209	5693	7816
Quick I/O	2186	5653	7622	2185	5644	7621
Cached Quick I/O	5750	8412	8643	5756	8429	8512

The results show very little performance difference for all I/O configurations between 32-bit Solaris 7 and 64-bit Solaris 7. In either case, Quick I/O achieves virtually the same level of performance as raw I/O, while Cached Quick I/O shows significant performance gains.

32-bit Oracle versus 64-bit Oracle (64-bit Solaris)

Oracle8i, 64-bit is a different release from Oracle8i, 32-bit. The 64-bit Oracle release supports SGA sizes bigger than 4GB (a 32-bit limitation); it therefore has some different performance characteristics on the Solaris 7 platform.

The following table shows the performance difference between 32-bit Oracle and 64-bit Oracle running with 64-bit Solaris 7. The results show that the 64-bit Oracle performance was about the same as the 32-bit Oracle performance when the Oracle buffer size was less than 2.5GB. The results also show that a bigger Oracle buffer size can be used with 64-bit Oracle to further improve the performance, which could not be achieved with 32-bit Oracle.

Table 3

32-bit Oracle vs. 64-bit Oracle (64-bit Solaris)							
	32-bit Oracle (6GB)			64-bit Oracle (10GB)			
	100MB	1GB	2.5GB	1GB	2.5GB	5GB	8GB
Raw I/O	2209	5693	7816	5528	7876	9259	9193
QIO	2185	5644	7621	5570	7817	9127	9080
CQIO	5756	8429	8512	9244	10115	10408	9039

Figure 2

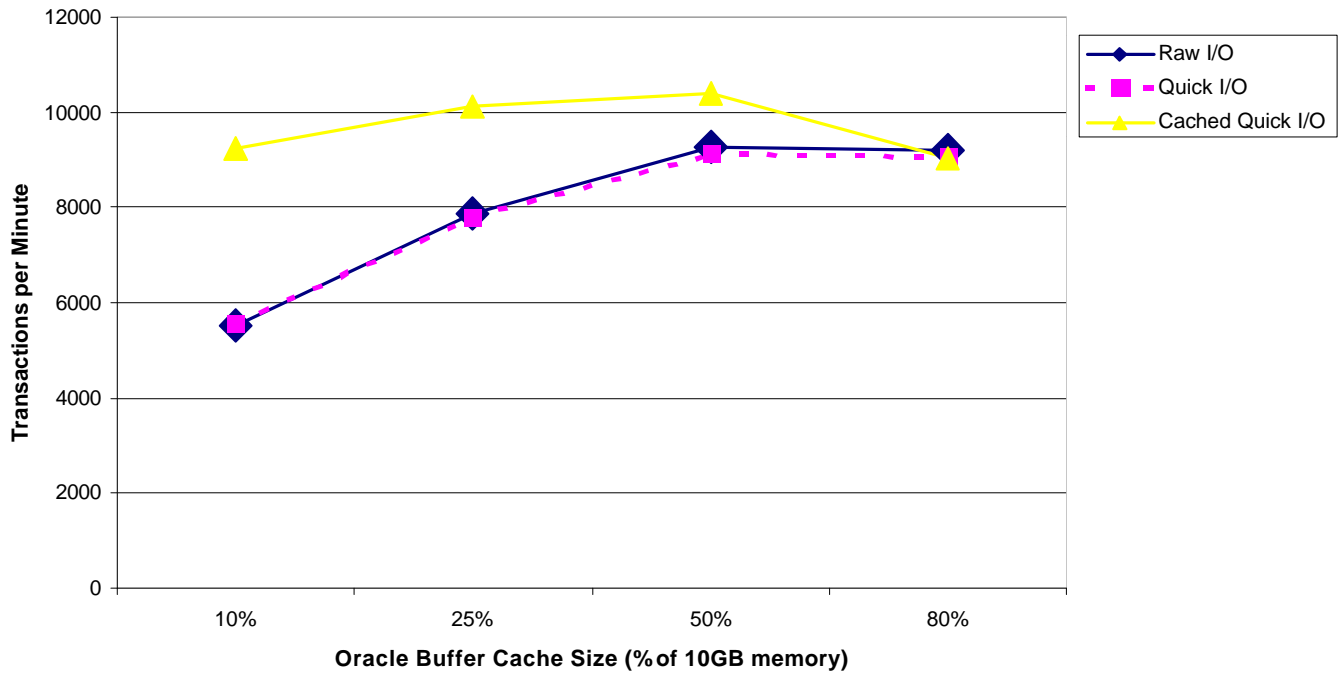


Figure 2 shows how OLTP performance was affected by the percentage of system memory used by Oracle buffers. As expected, increasing the Oracle buffer size improved the OLTP throughput for both the Quick I/O and raw I/O configuration. Cached Quick I/O still helped when some of the database blocks can be cached in the operating system. However, when the Oracle SGA size became closer to the total memory size, data caching in the operating system may actually slow down the OLTP performance. Another observation from the plots is that the best OLTP performance may be achieved by Cached Quick I/O, if the ratio between Oracle SGA and the operating system cache can be properly tuned.

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 I/O configuration, it was possible to study the impact of various storage layouts on database performance.

The OLTP performance measurements illustrate that the Quick I/O feature enables the VERITAS Database Edition *for Oracle* to have equal performance compared to raw partition configurations. This Quick I/O behavior remains the same no matter which Oracle release (32-bit Oracle or 64-bit Oracle) or which Solaris 7 flavor (32-bit or 64-bit) was used.

When 32-bit Oracle is used, the largest SGA size that can be allocated by Oracle is about 4GB. When a system has plenty memory, the memory beyond the 4GB Oracle SGA limit can be used to provide a second level cache when Cached Quick I/O is used. The second level cache may improve Oracle performance, when data blocks that could not be cached in Oracle can be cached in the operating system cache. In the 32-bit Solaris 7 environment Cached Quick I/O achieved up to a 159% performance gain over that experienced with raw partitions. The benefit of the second level cache has diminished somewhat when 64-bit Oracle is available, which allows for more than 4GB memory to be allocated by Oracle for its SGA. However, Cached Quick I/O may still be able to help the database performance. Careful tuning on the memory usage may be necessary to achieve an optimal performance.

Appendix A

The following /etc/system file was used to configure Solaris 7 for the tests:

```
*ident      "@(#)system 1.15  92/11/14 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
```

```
* SUNWecsr start
```

```
* The following system parameters are inserted here in order to
* override the default values used by workstations.
```

```
* These values have been tuned for better performance on Enterprise
* server systems.
```

```
* DO NOT EDIT!
```

```
set tune_t_gpgslo=250
set tune_t_minarmem=100
set tune_t_minasmem=250
```

```
set pln:pln_enable_detach_suspend=1
set soc:soc_enable_detach_suspend=1
set ssd:ssd_enable_detach_suspend=1
* SUNWecsr end
```

```
**
```

```
** Added for tpc-C testing
```

```
**
```

```
set msgsys:msginfo_msgmap=200
set msgsys:msginfo_msgmni=100
set msgsys:msginfo_msgtql=80
set msgsys:msginfo_msgseg=2048
set shmsys:shminfo_shmmax=0xffffffff
set shmsys:shminfo_shmseg=200
set semsys:seminfo_semmap=100
set semsys:seminfo_semmni=1000
set semsys:seminfo_semmns=4000
set semsys:seminfo_semmnu=800
set semsys:seminfo_semmsl=512
*set semsys:seminfo_semume=600
```

```
**
```

```
** END of tpc-C Stuff  
**
```

```
* vxvm_START (do not remove)  
forceload: drv/atf  
forceload: drv/pln  
forceload: drv/ses  
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 values allow the kernel stack size  
* for all threads to be increased to 16K.  
*
```

```
set lwp_default_stksize=0x4000  
set rpcmod:svc_run_stksize=0x4000  
* vxfs_END
```

Appendix B

The following *init.ora* file was used during the tests. The `db_block_buffers` parameter was changed for different Oracle buffer sizes. When the tests were run with synchronous I/O, `disk_asynch_io` was set to `FALSE` and the `dbwr_io_slaves` parameter was set to 20.

```
control_files                = (/tpcc_disks/cntrl1)
db_block_size                = 2048

# change the db_block_buffers for different Oracle buffer sizes
db_block_buffers            = 512000          # 1GB

# the following two lines are commented out during asynchronous I/O tests
# disk_synch_io              = FALSE
# dbwr_io_slaves            = 20

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     = 1000000000
log_checkpoints_to_alert    = TRUE
log_buffer                  = 1048576
gc_releasable_locks        = 0
max_rollback_segments      = 220
open_cursors                = 200
processes                   = 200
sessions                    = 600
transactions                 = 400
distributed_transactions    = 0
transactions_per_rollback_segment = 1
rollback_segments          =
(t1,t2,t3,t4,t5,t6,t7,t8,t9,t10,t11,t12,t13,t14,t15,t16,t17,t18,t19,t20,t21,t
22,t23,t24,t25,t26,t27,t28,t29,t30,t31,t32,t33,t34,t35,t36,t37,t38,t39,t40,t4
1,t42,t43,t44,t45,t46,t47,t48,t49,t50)
shared_pool_size            = 30000000
cursor_space_for_time       = TRUE
audit_trail                 = FALSE
```



*Corporate Headquarters
1600 Plymouth Street
Mountain View, CA 94043*

North American Sales Headquarters

*400 International Parkway
Heathrow, FL 32746
800-327-2232 or 407-531-7501
407-531-7730 Fax*

Global Locations

*United Kingdom
0800-614-961 or
44-(0)870-2431000
44-(0)870-2431001 Fax*

*France
33-1-41-91-96-37
33-1-41-91-96-38 Fax*

*Germany
49-(0)69-9509-6188
49-(0)69-9509-6264 Fax*

*South Africa
27-11-448-2080
27-11-448-1980 Fax*

*Australia
1-800-BACKUP
61-(0)2-8904-9833 Fax*

*Hong Kong
852-2507-2233
852-2598-7788 Fax*

*Japan
81-3-5532-8217
81-3-5532-0887 Fax*

*Malaysia
603-715-9297
603-715-9291 Fax*

*Singapore
65-488-7596
65-488-7525 Fax*

*China
8610-62638358
8610-62638359 Fax*

Electronic communication

*E-Mail:
sales@veritas.com*

*World Wide Web:
<http://www.veritas.com>*