Principled Technologies®

TEST REPORT
JANUARY 2009

OLTP performance comparison: Solid-state drives vs. hard disk drives

Executive summary

Intel Corporation (Intel) commissioned Principled Technologies (PT) to compare performance and power for online transaction processing (OLTP) of two types of disk drives:

- Intel X25-E Extreme SATA solid-state drives (SSDs)
- standard 15K RPM SAS hard disk drives (HDDs)

We focused on a single usage scenario: that of a user needing additional server performance and wanting to know whether six SSDs installed in server drive slots would provide better performance and lower power consumption than a full external 24-disk enclosure of HDDs. This is a user who does *not* need the additional storage capacity of the HDDs.

We used the DVD Store Version 2 (DS2) test tool. DS2 is an open-source simulation of an online e-commerce DVD store. Its main throughput metric is orders per minute, or OPM. We also measured power consumption during the test and while the systems were idle.

KEY FINDINGS

- Six internal SSDs delivered up to 35 percent higher and better performance on our OLTP tests than a full 24-disk enclosure of 15K RPM SAS HDDs.
- The SSD configuration, including the server, used nearly 35 percent less power when active and approximately 42 percent less power when idle than the HDD configuration.
- The SSD configuration delivered higher performance without the need for an additional RAID controller or an additional external drive enclosure.

For the HDD tests, we tested a full shelf of 24 Seagate Savvio 15K SAS 73GB hard disk drives in a Newisys NDS-2240 enclosure attached to a server via an LSI Logic MegaRAID SAS 8888ELP RAID Controller. For the SSD tests, we tested six 32GB Intel X25-E Extreme SATA solid-state drives that we placed directly in the existing server drive slots, which connect to the built-in RAID controller. For both tests, we used a server with four six-core Intel Xeon X7460 processors at 2.66 GHz and 8 GB of RAM. We ran a single instance of DS2 and tested with a single 20GB database.

In these OLTP tests, six internal SSDs delivered higher and better performance and lower power consumption than a full 24-disk enclosure of 15K RPM SAS HDDs. As a result, they are the better solution when a user needs additional drive performance but not the additional storage of the HDDs. The internal SSD solution also saves the cost of additional RAID controllers, external drive enclosures, and rack space the HDDs require.

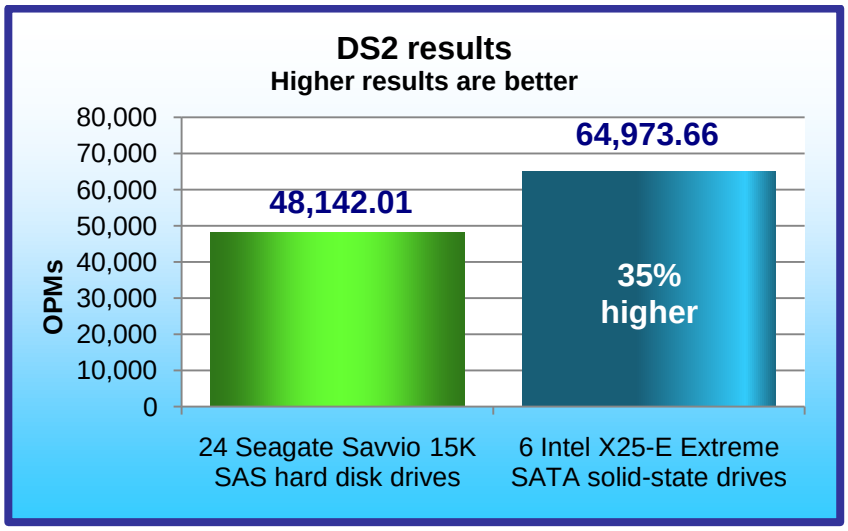

Figure 1: DS2 performance results in OPM for the two storage configurations. A higher OPM score is better.

Figure 1 shows the OPM results for the 24 HDDs and the six SSDs. These results are the average OPM performance during 200 seconds of steady activity and heavy load during a DS2 test run. We report the median OPM result of three DS2 test runs.

Six SSDs provided up to 35 percent better DS2 performance than a full enclosure of 24 HDDs: 64,973.7 OPM for the SSDs vs. 48,142.0 OPM for the HDDs.

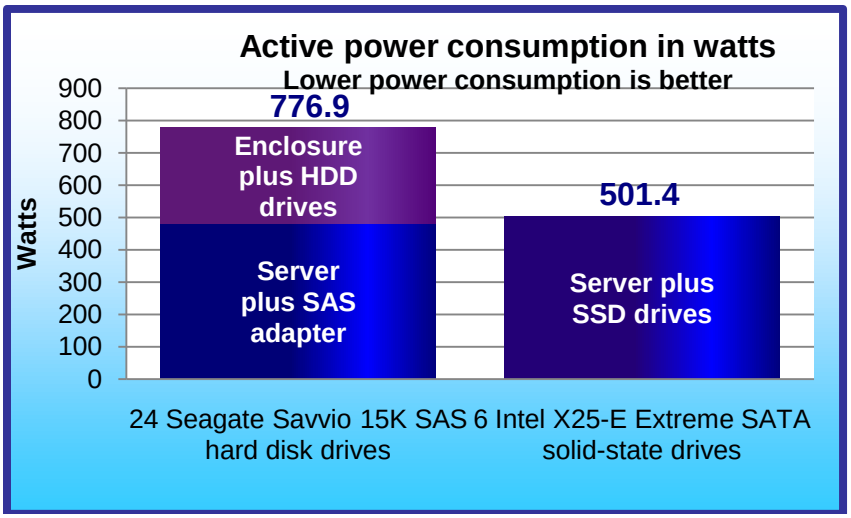

Figure 2: Active power consumption in watts for the two storage configurations. Lower active power consumption is better. The SSD configuration, including the server, used 35 percent less power when active.

We used power analyzers to log the power consumption (in watts) of the server and the enclosure (for the HDD tests) at one-second intervals during the tests. Figure 2 shows the average power consumption during the same periods and the same runs that produced the results in Figure 1.

During a period of server activity, the SSD configuration, including the server, used less power than the HDD configuration: 501.4 watts for the SSD configuration vs. 776.9 watts for the HDD configuration. In fact, the Server plus SSD drives in the SSD configuration consumed only slightly more power, an additional 20.3 watts, than only the Server plus SAS adapter in the HDD configuration.

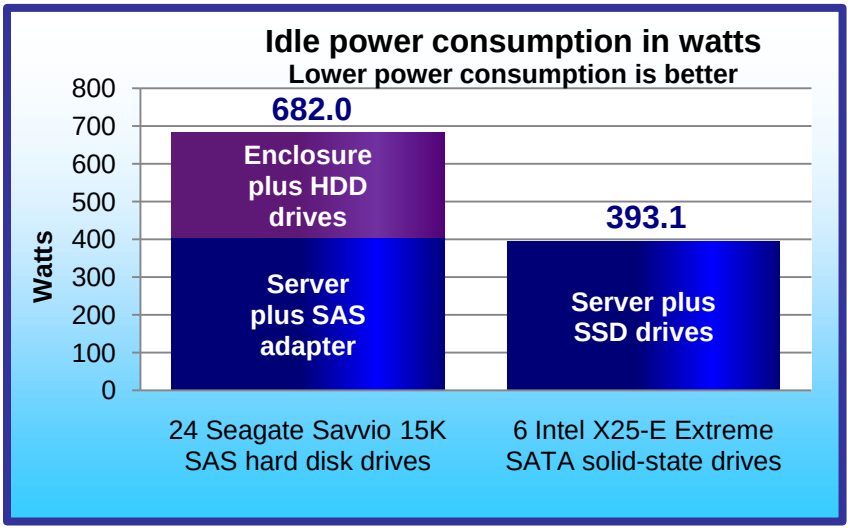

Figure 3: Idle power in watts consumption for the two storage configurations. Lower idle power consumption is better. The SSD configuration, including the server, used 42 percent less power when idle.

We also used power analyzers to log the power consumption (in watts) of the server and the enclosure (for the HDD tests) for two minutes while the server and drives were idle. Figure 3 presents those results.

During an idle period, the SSD configuration, including the server, used less power than the HDD configuration: 393.1 watts for the SSD configuration vs. 682.0 watts for the HDD configuration. In fact, the SSD configuration used 10.9 fewer watts than the Server plus SAS adapter in the HDD configuration.

We also measured processor utilization during the peak testing time, the period of steady activity and maximum I/O. The three SSDs drove the processors approximately 52 percent more than the 24 HDDs; 39.1 percent for the SSDs and 25.7 percent for the HDDs.

Workload

We conducted our testing using DVD Store Version 2, an open-source application with a back-end database component, a front-end Web application layer, and a driver layer that operates as the middle tier and actually executes the workload.

Because our goal was to isolate and test database server storage, we did not use the Web application layer. Instead, we ran the driver application directly via its command-line interface.

DS2 models an online DVD store. Virtual customers log in; browse movies by actor, title, or category; and purchase movies. The workload also creates new customers. Browsing movies involves select operations, some of which use full-text search and some of which do not. The purchase, login, and new customer procedures involve updates and inserts, as well as selects. The workload's main reporting metric is orders per minute, or OPM.

For more details about the DS2 tool, see http://www.delltechcenter.com/page/DVD+Store.

The server ran a single instance of DS2, which spawned 32 threads. This simulated a heavily loaded environment; the load-generating system ran with no think time, blasting requests as quickly as the server could handle them.

The DS2 driver application creates an orders-per-minute performance counter on the system. We created a data collector set to collect statistics once every second. We ran the test three times and report results from the run that produced the median of the three OPM results.

In addition, we monitored server performance statistics using the Reliability and Performance Monitor on the server. Our experiments showed that this function did not affect the performance of the test. We used these results to verify that neither the processor nor the memory was a bottleneck in the test.

To record the server's power consumption during testing, we used an Extech Instruments Power Analyzer/Datalogger. We used one Extech to measure the power draw of the server and a second Extech to measure the power draw of the drive array. We recorded power while servers were in idle and active states. Idle power is the average power consumption during two minutes while the server and drives were idle before DS2 runs. Active power is the average power consumption during 200 seconds of peak activity during the DS2 test run. We captured power consumption at one-second intervals. For the HDD results, we report the active and idle power consumption for both the storage array and server combined. We report active power consumption from the same runs that produced the median OPM values that we report.

Test results

Figure 4 provides test results for the two storage configurations.

	6 Intel X25-E Extreme SATA solid-state drives	24 Seagate Savvio 15K SAS hard disk drives
DS2 results for the three runs (higher is better)		
Run 1: OPM	64,973.7	48,172.0
Run 2: OPM	67,356.7	46,586.3
Run 3: OPM	64,203.0	48,142.0
Median OPM (higher is better)	64,973.7	48,142.0
Power measurements* from median run		
Idle power (lower is better)	393.1	682.0
Active power (lower is better)	501.4	776.9

Figure 4: Test results for the two storage configurations.

*Note: The power measurement includes the power usage of the server and the enclosure (for the HDD configuration).

Test methodology

For the HDD tests, we installed the 24 HDDs into a Newisys NDS-2240 enclosure, which we connected to a server via an LSI Logic MegaRAID SAS 8888ELP RAID Controller. For the SSD tests, we placed the six SSDs in the existing server drive slots. We configured the test drives as RAID 5 and created a database partition, which we used to hold database files and indices. Intel selected and provided the storage array, HDDs, and SSDs. PT provided the server. We ran DS2 tests, repeating the tests three times on each drive type.

	Intel X25-E Extreme SATA solid-state drives	Seagate Savvio 15K SAS hard disk drives
Vendor and model number	Intel SSDSA2SH032G1GN	Seagate ST973451SS
Number of drives in system	6	24
Size	32 GB	73 GB
RPM	N/A	15,000 RPM
Type	SATA 3.0 Gb/s	SAS 3.0 Gb/s
Controller	Integrated Intel RAID Controller SROMBSASFC	LSI Logic MegaRAID SAS 8888ELP RAID Controller
Controller driver	LSI 2.23.0.64 (07/01/2008)	LSI 2.23.0.64 (07/01/2008)

Figure 5: The drives we tested.

Figure 5 shows the configuration of the drives we tested.

Appendix A provides more detailed information on the storage configuration.

We used one server to host SQL Server and generate the DS2 workload to create demand on the storage mediums. The server ran with a 64-bit version of Windows Server 2003 x64, SQL Server 2008 Enterprise Edition x64, and .NET 2.0. Our server ran the DS2 driver application and executed a workload against the DS2 database. We attached the HDD storage array to the server via an LSI Logic MegaRAID SAS 8888ELP RAID Controller. We conducted all tests in a climate-controlled room.

Our server contained eight drives. We configured the first two internal server drives using RAID 0 for the operating system, SQL Server 2008 software, and the database log files. We used the remaining six internal drives in the SSD tests only.

The performance limits for this testing came from the storage; the CPU and other components had capacity to spare. To ensure that the database would not fit in cache, we made sure that the memory of the server was less than the size of our database. Figure 6 shows the configuration for the database server.

	Intel Xeon processor X7460-based server
Processors	Four six-core Intel Xeon X7460 processors at 2.66 GHz
Memory	8 GB of PC2-5300 FB-DDR2 memory
Internal disk	Two 73.4GB, 10,000RPM Seagate ST973401SS SAS drives
NICs	Intel Pro/1000 EB NIC and Intel 82575EB
OS	Microsoft Windows Server 2003 x64
Software	SQL Server 2008 Enterprise Edition x64 and DVD Store Version 2

Figure 6: Database server configuration.

Appendix B provides more detailed information on the test environment.

Setting up the storage disks

Before we ran the tests, we configured each of the storage disks as RAID 5 with disk cache enabled, ran Iometer on the SSDs to season them, and used Diskpart to align all drives on a 4KB boundary.

The rest of this section gives instructions for each of those steps.

Setting up the RAID (SSDs and HDDs)

1. Enter the MegaRAID BIOS Configuration Utility.
2. Select your adapter, and click Next.
3. Click Configuration Wizard.
4. Select New Configuration, and click Next.
5. At the This is a Destructive Operation! screen, click Yes.
6. Select Custom Configuration, and click Next.
7. Assign all of the drives in your array to your RAID, and click Accept DG.
8. Click Next.
9. Click Add to SPAN.
10. Click Next.
11. Set the RAID level to RAID 5, set Disk Cache to enabled, and change Select Size to the suggested RAID 5 size on the right.
12. Click Next.
13. Click Accept.
14. Click Yes.
15. Click Yes.
16. Click Home.
17. Click Exit.

Seasoning the drives (SSDs only)

Note: We preconditioned the drives so that our tests would deliver accurate sustained performance values. Without preconditioning, tests could deliver highly variable performance.

1. Plug in an SSD that you have securely erased or freshly performed a low-level format on.
2. Initialize the disk, and format it as NTFS.
3. With Iometer, run a one-second 128K sequential read test to the entire logical block addressing (LBA) drive space. This enables all LBAs to have some content so the SSD does not have an artificially high reserve space. Note: We used Iometer 2008-06-22-rc1, available from http://sourceforge.net/projects/iometer.
4. Delete the IOBW.tst file from the SSD drive.
5. With Iometer, run a 5,700-second 128K sequential read test (request size aligned on 4K sector boundaries) on 100 percent of the drive. This preconditions the drive.

Formatting the drive array with Diskpart (SSDs and HDDs)

1. Open a command prompt.
2. Type `cd c:\windows\system32`.
3. Type `diskpart.exe`.
4. Type `List Disk` to determine the name of your RAID array.
5. Type `Select Disk #` where Disk # is the name of your RAID array.
6. Type `Create partition primary align=4`.
7. Type `Assign Letter=E` to assign this new partition the letter E.
8. Type `Exit`.
9. In Windows, click Start, right-click My Computer, and select Manage.
10. Click Disk Management.
11. Right-click the partition, and select Format.
12. Name the partition according to what kind of drives you are using, and format the drives as NTFS.

Connecting the Extech Power Analyzer/Datalogger

To record each storage configuration's power consumption during testing, we used an Extech Instruments (www.extech.com) 380803 Power Analyzer/Datalogger. Because the server had two power supplies, we measured the power draw of the server by using a single Extech Power Analyzer with a splitter cable.
We used a second Extech Power Analyzer to measure the power draw of the drive array for the HDD tests. The enclosure also had dual power supplies, so we used a splitter cable to measure the power draw through a single meter.

We connected the Extech Power Analyzers via a RS-232 cable to a separate power monitoring system to record the power draw of the devices under test. We used the Power Analyzer's Data Acquisition Software (version 2.11) installed on the power monitoring system to capture all the recordings.

Installing and setting up SQL Server 2008 on the test server

1. Enter the SQL Server 2008 installation CD.
2. Click OK.
3. Accept the license agreement.
4. Click Install.
5. Click Exit.
6. Click Next.
7. Select I Agree.
8. Click Next.
9. Click Continue.
10. Click Finish.
11. Click OK.
12. Restart the Server.
13. Click My Computer.
14. Double-click the CD Drive.
15. Click Installation.
16. Click New SQL Server Stand-Alone Installation or Add Features to an Existing Installation.
17. Click OK.
18. Select Specify a Free Edition.
19. Click Next.
20. Accept the license terms.
21. Click Next.
22. Click Install.
23. Click Next.
24. Select the following options:
    - Database Engine Services
    - Full-Text Search
    - Client Tools Connectivity
    - Client Tools Backwards Compatibility
    - Management Tools Basic
    - Management Tools Complete
25. Click Next.
26. Click Next.
27. Click Next.
28. Change the SQL Server Agent Account Name to NT AUTHORITY\SYSTEM.
29. Change the SQL Server Database Account Name to NT AUTHORITY\SYSTEM.
30. Click Next.
31. Select Mixed Mode.
32. Enter a password for the administrator "sa" account.
33. Confirm your password.
34. Click Add Current User.
35. Click Next.

36. Click Next.
37. Click Next.
38. Click Install.
39. Click Next.
40. Click Close.
41. Right-click My Computer, and select Manage.
42. Click Local Users and Groups.
43. Click Users.
44. Right-click, and select New User.
45. Type `SQLFTUser` for the username.
46. Type `password` for the password, and confirm password.
47. Uncheck the User Must Change Password at Next Logon Checkbox.
48. Check the Password Never Expires box.
49. Click Create.
50. Repeat steps 44 through 49 for the username SQLLocalUser.
51. Run the SQL Configuration Manager.
52. Click SQL Server Services.
53. Right-click SQL Server, and select Properties.
54. Select Log On as This Account.
55. Type `SQLLocalUser` for the account name.
56. Type `password` for the password, and confirm password.
57. Click OK.
58. Click Yes.
59. Right-click SQL Full-text Filter, and select Properties.
60. Select the Service tab.
61. Change the Start Mode to automatic.
62. Click Apply.
63. Select Log On as This Account.
64. Type `SQLFTUser` for the account name.
65. Type `password` for the password, and confirm password.
66. Click Apply.
67. Click Start.
68. Click OK.

Installing and setting up DVD Store on the test server

1. Copy the ds2 folder containing the DVD Store executables to C:\ds2.
2. Double-click the folder.
3. Create a file named `DS2_1.bat` that contains the following DVD Store run command line:
   ```
   c:\ds2run\ds2sqlserverdriver --target=localhost --ramp_rate=10 --run_time=20
   --n_threads=32 --db_size_str=W --think_time=0 --database_name=DS2_1
   ```
4. Run DS2_1.bat for 10 seconds to create the DVD Store testing counters.
5. Click Start→Run.
6. Type `perfmon`, and press Enter.
7. Expand Performance Logs and Alerts.
8. Click Counter Logs.
9. Right-click the right pane, and select New Log Settings.
10. Type `DVD Store` as the name.
11. Click OK.
12. Click Add Counters.
13. Add the following counters: `Memory\Available Mbytes, Physical Disk(0 C:)\% Idle Time, Physical Disk(0 C:)\Disk Read Bytes/sec, Physical Disk(0 C:)\Disk Reads/sec, Physical Disk(0 C:)\Disk Write Bytes/sec, Physical Disk(0 C:)\Disk Writes/sec, Physical Disk(1 E:)\% Idle Time, Physical Disk(1 E:)\Disk Read Bytes/sec, Physical Disk(1 E:)\Disk Reads/sec, Physical Disk(1 E:)\Disk Write`

```
Bytes/sec, Physical Disk(1 E:)\Disk Writes/sec, Physical Disk(1 E:)\Disk
Transfers/sec, Processor(0-23)\% Processor Time, SQLServer:Buffer
Manager\Buffer cache hit ratio, SQLServer:Buffer Manager\Checkpoint
pages/sec, Test\MaxRT, Test\OPM
```

14. Click Close.
15. Change the Sample Interval to 1.
16. Select the Log Files tab.
17. Change the Log file type to Text File (Comma delimited).
18. Select the Schedule tab.
19. Change the Start log to start manually.
20. Click OK.
21. Click Yes.
22. Open My Computer.
23. Double-click Drive E.
24. Create a new folder named `SQLData`.
25. Right-click that folder, and select Properties.
26. Select the Security tab.
27. Click Add.
28. In the Object Names box, type `SQLLocalUser`.
29. Click OK.
30. For full access, check the box beside Allow.

Generating the datasets

We built the database schema using the scripts included in the DS2 distribution package, though we made a few minor modifications. The DS2 stress tool provides options to generate 10MB, 1GB, or 100GB datasets. To get the tool to generate the 20 GB of user data we used in this test, we had to make a few straightforward changes to the source code and to the DVD Store application's scripts. Note: We created our test data on a Linux system.

Editing the ds2_create_orders.c module

The module ds2_create_orders.c defines constants that define the maximum values for the customer ID and the product ID. The constants for the 20GB database size did not exist. We added the constants for this size.

On the command line for the ds2_create_orders.c module, we specified the size. The available options were S (small), M (medium), and L (large). We added the case W for the 20GB database. In the switch statement that sets the values for the variables max_cust_id and max_prod_id, we added cases that assigned them the proper values for the 20GB database size.

We recompiled the ds2_create_orders.c module on Linux, following the instructions in the header comments. We used the following command line: `gcc -o ds2_create_orders ds2_create_orders.c -lm`

Editing the ds2_create_cust.c module

We had to make the same changes to the ds2_create_cust.c module that we made to the ds2_create_orders.c module. On the command line for the ds2_create_cust.c module, we specified the size. The available options were S (small), M (medium), and L (large). We added the case W for the 20GB database. In the switch statement that sets the values for the variables max_cust_id and max_prod_id, we added cases that assigned them the proper values for the 20GB database size.

We recompiled the ds2_create_cust.c module on Linux, following the instructions in the header comments. We used the following command line: `gcc -o ds2_create_cust ds2_create_cust.c -lm`

Generating the data for the 20GB database

We used shell scripts to run all four of the executables that generate the data. The distribution did not include shell scripts for the 20GB size. We wrote shell scripts based on the ds2_create_cust_large.sh and ds2_create_orders_large.sh scripts. The ds2_create_prod and ds2_create_inv executables did not ship with associated shell scripts, so we created shell scripts using the instructions in the readme files. We ran the shell scripts in the following order to generate the data for the 20GB database:

- ds2_create_orders_20gb.sh
- ds2_create_inv_20gb.sh
- ds2_create_prod_20gb.sh
- ds2_create_cust_20gb.sh

We waited until the processes finished before we moved onto the next step.

Creating the database

We modified the database creation SQL Server scripts included in the DVD Store distribution package to build the database schema, which includes the file structure, tables, indices, stored procedures, triggers, and so on. We built a master copy of the 20GB database version for SQL Server 2008, and then used that master copy to restore our test database to the test server between each test run.

We followed these steps to create the database:

1. We created the database and file structure using database creation scripts provided with DS2. We made size modifications specific to our 20GB database and the appropriate changes to drive letters.
2. We created database tables, stored procedures, and objects.
3. We set the database recovery model to bulk-logged to prevent excess logging.
4. We loaded the generated data. For data loading, we used the import wizard in SQL Server Management Studio. Where necessary, we retained options from the original scripts, such as "Enable Identity Insert."
5. We created indices, full-text catalogs, primary keys, and foreign keys using the database-creation scripts.
6. We updated statistics on each table according to database-creation scripts, which sample 18 percent of the table data.
7. We created ds2user SQL Server login and user for testing.
8. We set the database recovery model back to full.

We made the following several changes in the build scripts:

- Because we varied the size of the datasets, we sized the files in our scripts to reflect the database size and the number of files per filegroup. We allowed for approximately 40 percent free space in our database files to ensure that filegrowth activity did not occur during the testing.
- We followed Microsoft's recommendation of having 0.25 to 1 file per filegroup per core and we used eight files per filegroup on our 24-core server.
- We did not use the DBCC PINTABLE command for the CATEGORIES and PRODUCTS tables, both because Microsoft recommends against this practice and because the commands do nothing in SQL Server 2008.
- In SQL 2008, we added the FORCESEEK query hint to the BROWSE_BY_ACTOR stored procedure, to force SQL Server 2008 to use an index seek, instead of an index scan, in its query execution plan. We made this change because our initial tests showed that SQL Server was using a highly inefficient index scan. Therefore, we created the SQL Server 2008 BROWSE_BY_ACTOR procedure as follows:

```
CREATE PROCEDURE BROWSE_BY_ACTOR
 (
 @batch_size_in  INT,
 @actor_in   VARCHAR(50)
 )

 AS

 SET ROWCOUNT @batch_size_in
 SELECT * FROM PRODUCTS
```

```
--added to force index seek
WITH (FORCESEEK)

WHERE CONTAINS(ACTOR, @actor_in)
SET ROWCOUNT 0
GO
```

- We created a SQL Server login called `ds2user` and a database user mapped to this login. We made each such user a member of the db_owner fixed database role.
- Using the DVD Store scripts as a reference, we created the full-text catalog and index on the PRODUCTS table manually in SQL Server Management Studio.

We then performed a full backup of the database. This backup allowed us to restore the server to a pristine state relatively quickly between tests.

Editing the DVD Store scripts

We had to make a few minor modifications to the DVD Store test scripts. We detail these modifications below.

Editing the ds2xdriver.cs module

To use the 20GB database we created earlier, we had to change the following constants:

- In the routine Controller(), we changed the string "sizes." We added the W option for the 20GB database size. DS2 uses the sizes string to interpret the db_size_str option.
- In the class Controller, we changed the arrays MAX_CUSTOMER and MAX_PRODUCT. To each, we added values specifying the bounds for the customer and product IDs. The Controller() routine uses these arrays.
- We added a command line parameter for the database name: `–database_name`

Editing the ds2sqlserverfns.cs module

We changed the connection string to increase the number of available connections, to not use the default administrator ("sa") account, and to include a parameter for the database name. We raised the available connections limit from the default of 100 to 200 to allow room for experimentation. We created a user account called `ds2User` and used that account.

The ds2connect routine in the ds2sqlserverfns.cs module defines sConnectionString. We used the following string; the changes we made appear in bold.

```
string sConnectionString = "User ID=ds2User;Initial Catalog="+dbname+";Max
Pool Size=200;Connection Timeout=120;Data Source=" + Controller.target;
```

Recompiling the ds2sqlserverdriver.exe executable

We recompiled the ds2xdriver.cs and ds2sqlserverfns.cs module on Windows by following the instructions in the header comments. Because the DS2 instructions were for compiling from the command line, we used the following steps:

1. We opened a command prompt.
2. We used the cd command to change to the directory containing our sources.
3. We ran the batch file C:\Program Files\Microsoft Visual Studio 9.0\Common7\Tools\vsvars32.bat. This set up the environment variables for us.
4. We executed the following command:
   csc /out:ds2sqlserverdriver.exe ds2xdriver.cs ds2sqlserverfns.cs /d:USE_WIN32_TIMER /d:GEN_PERF_CTRS

Creating a script to delete and recreate the DB2 database

We created the following script, DS2_Drop_and_Restore.sql, to delete and recreate the DB2 database between each run:

```
USE [master]
GO

IF EXISTS (SELECT name FROM sys.databases WHERE name = N'DS2_1')
DROP DATABASE [DS2_1]
GO
RESTORE DATABASE [DS2_1] FROM DISK = N'C:\Program Files\Microsoft SQL
Server\MSSQL10.MSSQLSERVER\MSSQL\Backup\DS2_BackupFile_20GB.bak' WITH FILE = 1,
MOVE N'primary' TO N'E:\SQLData\DS2_1.mdf', MOVE N'cust1' TO
N'E:\SQLData\DS2_1_1.ndf',
MOVE N'cust2' TO N'E:\SQLData\DS2_1_2.ndf', MOVE N'cust3' TO
N'E:\SQLData\DS2_1_3.ndf',
MOVE N'cust4' TO N'E:\SQLData\DS2_1_4.ndf', MOVE N'cust5' TO
N'E:\SQLData\DS2_1_5.ndf',
MOVE N'cust6' TO N'E:\SQLData\DS2_1_6.ndf', MOVE N'cust7' TO
N'E:\SQLData\DS2_1_7.ndf',
MOVE N'cust8' TO N'E:\SQLData\DS2_1_8.ndf', MOVE N'ind1' TO
N'E:\SQLData\DS2_1_9.ndf',
MOVE N'ind2' TO N'E:\SQLData\DS2_1_10.ndf', MOVE N'ind3' TO
N'E:\SQLData\DS2_1_11.ndf',
MOVE N'ind4' TO N'E:\SQLData\DS2_1_12.ndf', MOVE N'ind5' TO
N'E:\SQLData\DS2_1_13.ndf',
MOVE N'ind6' TO N'E:\SQLData\DS2_1_14.ndf', MOVE N'ind7' TO
N'E:\SQLData\DS2_1_15.ndf',
MOVE N'ind8' TO N'E:\SQLData\DS2_1_16.ndf', MOVE N'ds_misc1' TO
N'E:\SQLData\DS2_1_17.ndf',
MOVE N'ds_misc2' TO N'E:\SQLData\DS2_1_18.ndf', MOVE N'ds_misc3' TO
N'E:\SQLData\DS2_1_19.ndf',
MOVE N'ds_misc4' TO N'E:\SQLData\DS2_1_20.ndf', MOVE N'ds_misc5' TO
N'E:\SQLData\DS2_1_21.ndf',
MOVE N'ds_misc6' TO N'E:\SQLData\DS2_1_22.ndf', MOVE N'ds_misc7' TO
N'E:\SQLData\DS2_1_23.ndf',
MOVE N'ds_misc8' TO N'E:\SQLData\DS2_1_24.ndf', MOVE N'orders1' TO
N'E:\SQLData\DS2_1_25.ndf',
MOVE N'orders2' TO N'E:\SQLData\DS2_1_26.ndf', MOVE N'orders3' TO
N'E:\SQLData\DS2_1_27.ndf',
MOVE N'orders4' TO N'E:\SQLData\DS2_1_28.ndf', MOVE N'orders5' TO
N'E:\SQLData\DS2_1_29.ndf',
MOVE N'orders6' TO N'E:\SQLData\DS2_1_30.ndf', MOVE N'orders7' TO
N'E:\SQLData\DS2_1_31.ndf',
MOVE N'orders8' TO N'E:\SQLData\DS2_1_32.ndf', MOVE N'ds_log' TO N'C:\Program
Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\Data\DS2_1_33.ldf',

NOUNLOAD, STATS = 10
GO
USE [DS2_1]
GO

/****** Object: User [ds2user] Script Date: 11/03/2008 14:28:07 ******/
IF EXISTS (SELECT * FROM sys.database_principals WHERE name = N'ds2user')
DROP USER [ds2user]
GO
USE [master]
GO
```

```
CREATE LOGIN [ds2user] WITH PASSWORD=N'', DEFAULT_DATABASE=[master],
CHECK_EXPIRATION=OFF, CHECK_POLICY=OFF
GO

USE [DS2_1]
GO
CREATE USER [ds2user] FOR LOGIN [ds2user]
GO
USE [DS2_1]
GO
EXEC sp_addrolemember N'db_owner', N'ds2user'
GO
```

Running the DS2 test

Before starting each DS2 test, we deleted and recreated the DB2 database. We rebooted the server and allowed it to sit idle for at least eight minutes to ensure that it finished with all aspects of the boot process. We started the power analyzer and recorded power consumption at an idle state for two minutes. We then ran the DS2 tests three times and recorded power during the test runs. This section describes this procedure:

1. Run the "DS2_Drop_and _Restore.sql" script on the test server to restore the database to its original state.
2. Click Connect.
3. Press F5.
4. Restart the server once the database finishes restoring.
5. Wait eight minutes.
6. Start recording power on the power monitoring system.
7. Wait one minute and 50 seconds.
8. Begin the perfmon DVD Store Counter on the test server.
9. Wait 10 seconds.
10. Run DS2_1.bat on the test server to begin the DVD Store benchmark.
11. Stop perfmon on the test server once DVD Store completes its test.
12. Stop recording power one minute past the end of the test.
13. Repeat steps 1 through 12 two more times, for a total of three runs.

Reporting results

For our OPM and active power consumption metrics, we use the results of the period from 1,000 seconds to 1,200 seconds into the test. This is a period of steady activity and heavy load, and it suffers from neither ramp-up nor ramp-down effects.

We ran the test three times for each of the two configurations of drives. We followed these steps to identify the results for each drive configuration:

1. During the DS2 tests, data collector sets on the system collected the OPM statistics once every second. We averaged those values for the period from 1,000 seconds to 1,200 seconds into the test for each of the three test runs. We report the median of the three results as the OPM result for the configuration.

2. We used a power analyzer to collect power measurements once every second during the test runs. For the same run that produced the median OPM result for the configuration, we averaged the active power measurements for the period from 1,000 seconds to 1,200 seconds into the test. We report that average as the active power consumption for the configuration.

3. We started the power measurements two minutes before each run. For each of the three runs, we calculated the average of the power measurements for those two minutes. We report the median of the three results as the idle power consumption for the configuration.

Appendix A: Storage configuration

Figure 7 describes the storage disks.

	6 Intel X25-E Extreme SATA solid-state drives	24 Seagate Savvio 15K SAS hard disk drives
Storage connectivity	SATA	SAS
Storage model	Integrated Intel RAID Controller SROMBSASFC	Newisys NDS-2240
Number of storage controllers	1	1
HBA model and firmware	Integrated Intel RAID Controller SROMBSASFC 1.11.32-0307	MegaRAID SAS 8888ELP 1.20.32-0512
Number of HBAs/host	1	1
Total number of drives tested in solution	6	24

Figure 7: Primary storage hardware.

Figure 8 shows the storage drive configuration.

	6 Intel X25-E Extreme SATA solid-state drives	24 Seagate Savvio 15K SAS hard disk drives
Drive type, speed	SSD	SAS, 15K RPM
Firmware	8620	SM04
Raw capacity per drive	32 GB	73 GB
Number of physical drives in test	6	24
Total raw storage capacity	192 GB	1,752 GB
RAID level	RAID 5	RAID 5
Total formatted capacity	144 GB	144 GB

Figure 8: Primary storage drive configuration.

Appendix B: Test environment

We used one server, a SQL Server 2008 database server, to generate the workload and create demand on the storage. Figure 9 provides detailed configuration information for the test server.

	Intel Xeon processor X7460-based server
General processor setup	
Number of processor packages	4
Number of cores per processor package	6
Number of hardware threads per core	1
System power management policy	Always on
CPU	
Vendor	Intel
Name	Intel Xeon processor X7460
Stepping	1
Socket type	Socket P (478)
Core frequency	2.66 GHz
Front-side bus frequency	1,066 MHz
L1 cache	32 KB + 32 KB (per core)
L2 cache	3 x 3 MB (each 3 MB shared by 2 cores)
L3 cache	16 MB
Platform	
Vendor and model number	Intel Fox Cove
Motherboard model number	S7000FC4UR
Motherboard chipset	Intel ID3600
Motherboard revision number	01
BIOS name and version	Intel SFC4UR.868.01.00.0024.061320082253
BIOS settings	Default
Memory module(s)	
Vendor and model number	Kingston KVR667D2D4F5/2G
Type	PC2-5300 FB-DDR2
Speed	667 MHz
Speed in the system currently running @	667 MHz
Timing/Latency (tCL-tRCD-iRP-tRASmin)	5-5-5-15
RAM module size	2 GB
Number of RAM modules	4
Chip organization	Double-sided
Total system memory	8 GB
Hard drive	
Vendor and model number	Seagate ST973401SS
Number of disks in system	2
Size	146.8 GB
Buffer size	8 MB

	Intel Xeon processor X7460-based server
RPM	10,000
Type	SAS
Controller	Integrated Intel RAID Controller SROMBSASFC
Driver	LSI 2.23.0.64
Operating system	
Name	Windows Server 2003 Enterprise x64 Edition
Build number	3790
Service Pack	2
File system	NTFS
Kernel	ACPI Multiprocessor x64-based PC
Language	English
Microsoft DirectX version	9.0c
Graphics	
Vendor and model number	ATI ES1000
Chipset	ES1000
BIOS version	BK-ATI VER008.005.031.000
Type	Integrated
Memory size	32 MB
Resolution	1,024 x 768
Network card/subsystem	
Vendor and model number	Intel PRO/1000 EB
Type	Integrated
Driver version	Intel 9.12.30.0
Vendor and model number	Intel 82575EB
Type	PCI Express
Driver version	Intel 10.3.49.0
Optical drive	
Vendor and model number	Optiarc DVD-ROM DDU810A
USB ports	
Number	5
Type	USB 2.0
Power supplies	
Total number	2
Wattage of each	1,570W
Cooling fans	
Total number	8
Dimensions	4 x 80 mm + 4 x 120 mm
Voltage	12 V
Amps	4 x 1.76 A + 4 x 3.3 A

Figure 9: Detailed system configuration information for the test server.

About Principled Technologies

We provide industry-leading technology assessment and fact-based marketing services. We bring to every assignment extensive experience with and expertise in all aspects of technology testing and analysis, from researching new technologies, to developing new methodologies, to testing with existing and new tools. When the assessment is complete, we know how to present the results to a broad range of target audiences. We provide our clients with the materials they need, from market-focused data to use in their own collateral to custom sales aids, such as test reports, performance assessments, and white papers. Every document reflects the results of our trusted independent analysis.

We provide customized services that focus on our clients' individual requirements. Whether the technology involves hardware, software, Web sites, or services, we offer the experience, expertise, and tools to help you assess how it will fare against its competition, its performance, whether it's ready to go to market, and its quality and reliability.

Our founders, Mark L. Van Name and Bill Catchings, have worked together in technology assessment for over 20 years. As journalists, they published over a thousand articles on a wide array of technology subjects. They created and led the Ziff-Davis Benchmark Operation, which developed such industry-standard benchmarks as Ziff Davis Media's Winstone and WebBench. They founded and led eTesting Labs, and after the acquisition of that company by Lionbridge Technologies were the head and CTO of VeriTest.

Principled Technologies, Inc.
1007 Slater Road, Suite 250
Durham, NC 27703
www.principledtechnologies.com
info@principledtechnologies.com