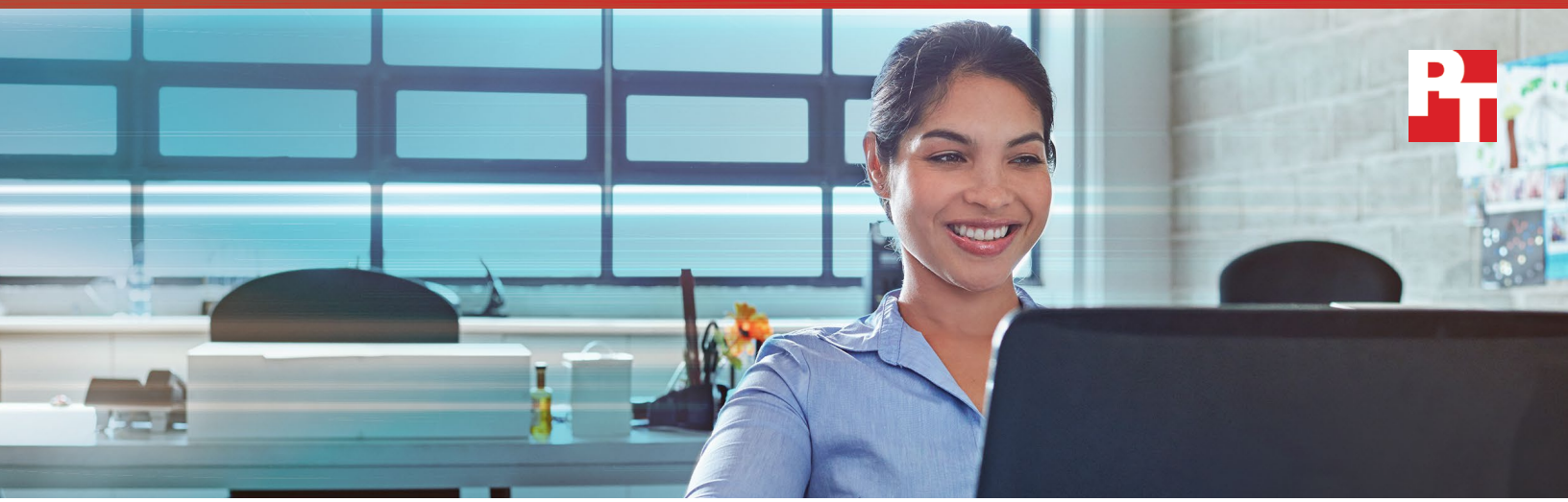


Get rid of database workload silos

The all-flash Dell EMC SC5020 storage array handled transactional database workloads and data mart imports better than an HPE solution without sacrificing performance

Your business may need to keep track of dozens of different initiatives—but that doesn't mean you need dozens of separate storage solutions to get the job done. To reduce complexity, your business may consider storage solutions that can take care of multiple jobs at once without sacrificing performance. For example, if you operate a brick-and-mortar store and an online store, you should be able to retrieve customer data from both sources without compromising transactional database performance. The all-flash Dell EMC™ SC5020 storage array aims to be just such a solution.

In the Principled Technologies datacenter, we tested the all-flash Dell EMC SC5020 array and the HPE™ Nimble Storage® AF5000 All Flash Array to see how well they performed while handling two workloads at once: processing orders from a transactional application, and loading terabytes of data into an empty data mart. The Dell EMC SC5020 delivered stronger transactional database performance, whereas the HPE array processed orders more slowly and took longer to load data.

With the all-flash Dell EMC array, your company could attend to more customer orders each minute and save time while simultaneously importing data.

The all-flash Dell EMC SC5020 storage array

Process up to 3.4x the orders per minute
during a data mart import, at a fraction of the latency*

Load to a data mart up to 31% faster
during an online transaction processing workload, at a fraction of the latency*

*Compared to a solution based on the HPE Nimble Storage AF5000 All Flash Array

How we tested

In the Principled Technologies datacenter, we compared all-flash storage solutions from Dell EMC and HPE in a test that combined two storage-intensive workloads: processing orders in a transactional database and importing data into a data mart.

The following table details the components of each solution.

	Dell EMC solution	HPE solution
Storage array	1 x Dell EMC SC5020	1 x HPE Nimble Storage AF5000 All Flash Array
Transactional database servers	3 x Dell EMC PowerEdge™ R740xd	3 x HPE ProLiant DL380 Gen10
Data mart import server	1 x Dell EMC PowerEdge R930	1 x Dell EMC PowerEdge R930

Transactional database workload

We clustered together three servers to serve as hosts for our transactional database test. The three servers ran 20 VMs total. We used a benchmarking tool called DVD Store 2 (DS2) to generate the database workload.[1] DS2 simulates an online video marketplace, mimicking the way thousands of users might shop in a real-life scenario. The tool records the number of tasks each solution fulfills during the course of the workload, and outputs the overall average in terms of operations per minute, or OPM.

What's a data mart?

Your enterprise business collects data from many different departments. Whether for sales, marketing, or research and development, you will often want to bundle data from disparate sources and load it into a single location for analysis and reporting. The data mart is a convenient place to store different departments' information for processing and analysis.

Data mart import workload

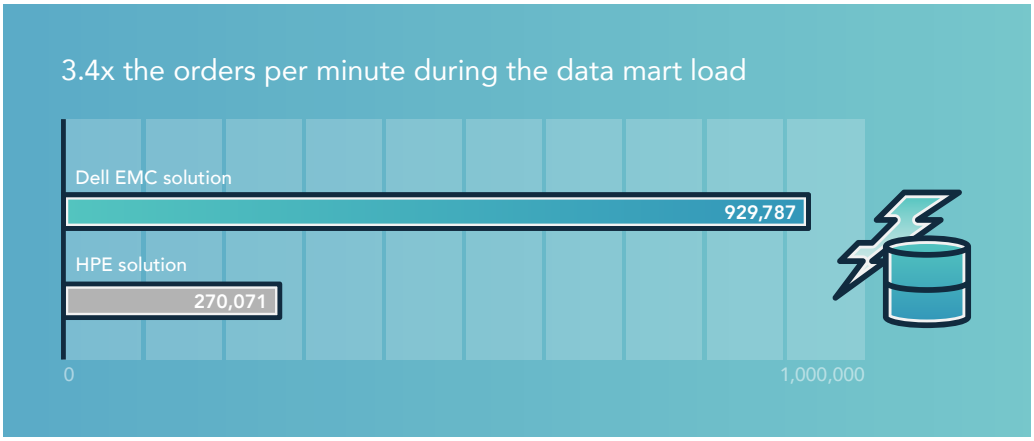
We used one server as a dedicated data mart import host, and used the storage array to host files for an empty Microsoft® SQL Server® database. We outfitted the server with enough virtual compute, RAM, and physical storage resources to prevent the server from becoming a bottleneck. We used a script to copy raw data from three PCIe SSDs into the storage array's database. Once the storage imported all data into the database, we determined how long the process took by comparing timestamps from the start and end of the test run.

Combining the tests

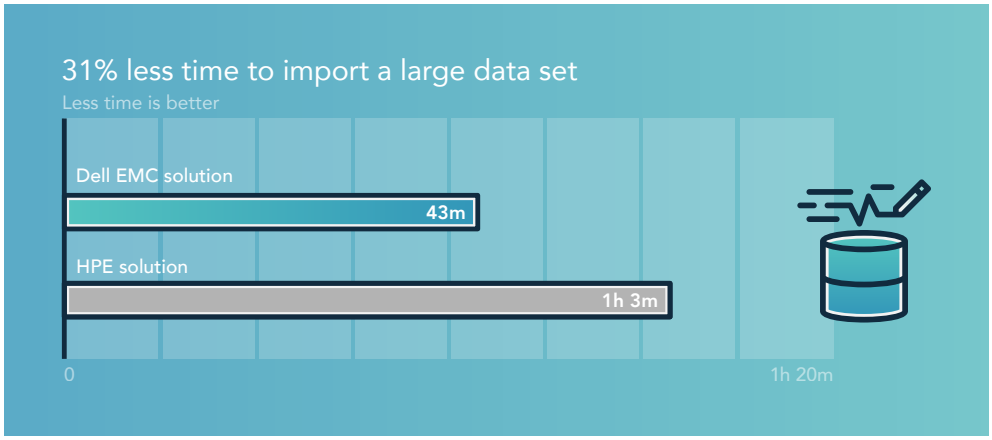
We began the combined test by letting DVD Store 2 ramp up to a stable OPM rate. Next, we launched the data mart script to begin the import process. Even though DS2 began to run before the data mart workload did, we include DS2 results from only the time the two workloads ran simultaneously.

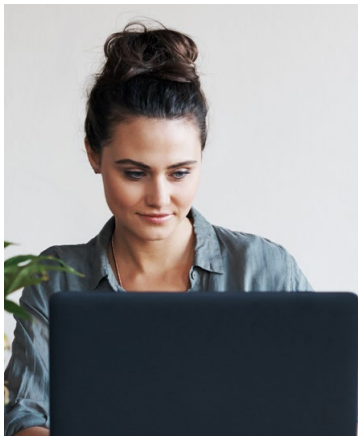
Handle demanding transactional database work while loading to a data mart without losing performance

The task of transferring files to a single data mart typically requires its own dedicated storage. But with newer technologies, your business could get rid of this restrictive silo and combine multiple workloads into a single solution. To that end, our test results suggest the all-flash Dell EMC SC5020 could support multiple storage-intensive workloads better than a solution built on the HPE Nimble Storage AF5000 All Flash Array.



While loading to a data mart, the all-flash Dell EMC solution fulfilled over 929K orders per minute on average, while the HPE solution only managed 270K—less than a third of the Dell EMC solution's capability. While juggling transactional database work, it took the Dell EMC solution just 43 minutes to load 3 TB of information into a data mart. That same task took the HPE solution over an hour—46 percent longer.





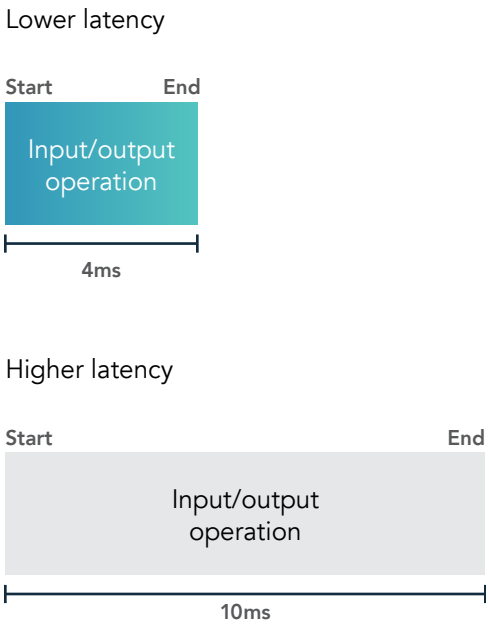
Lower latency contributes to stronger performance

In our tests, an all-flash Dell EMC storage solution processed more OPM and finished a data mart import faster than the HPE solution. The Dell EMC solution also performed both workloads at lower latency than the HPE solution—up to 83 percent. But what is latency, and how does it affect your organization?

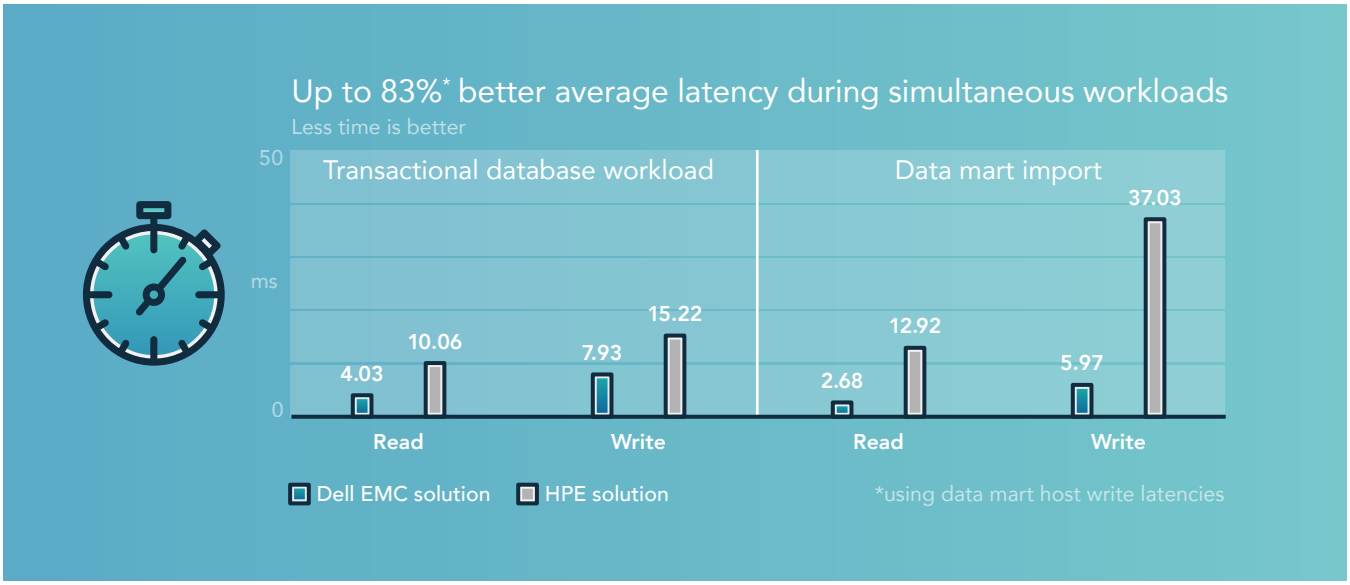
In this paper, latency represents the amount of time it takes the storage array to act on a request after receiving it from the server.

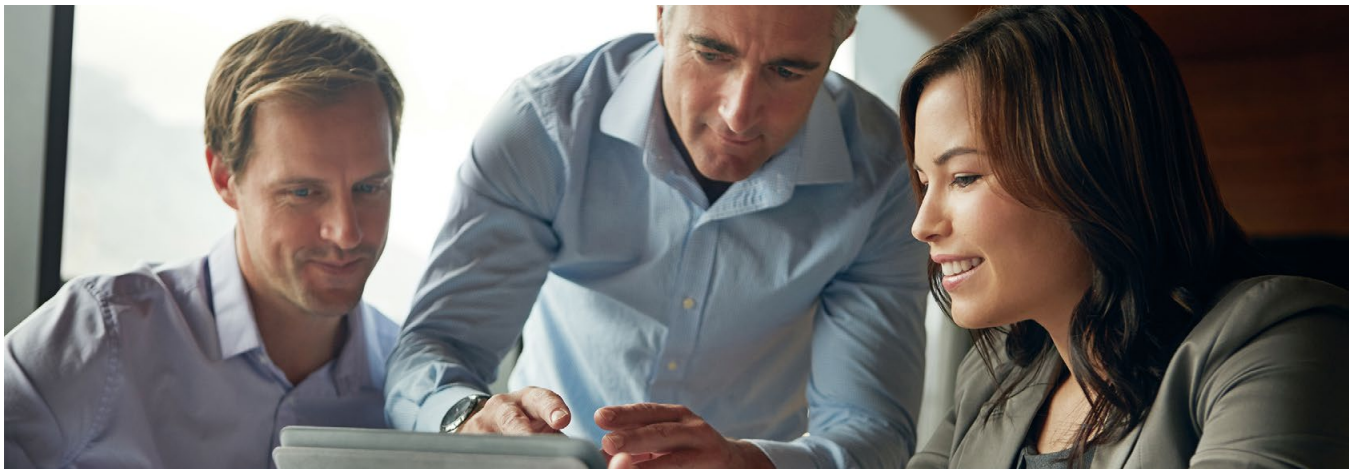
The server could request a **write** operation ("Put this data into the database"), or a **read** operation ("Find this data from the database"). Low latency means your storage begins to handle these operations quickly after receiving a request from the server.

In a real-world situation, the storage array handles many requests at the same time. When your storage is quick to respond, it can process more of these requests during peak times.



Below is a graph comparing latency across the two solutions for each workload during the simultaneous test. The Dell EMC solution functioned at up to 83 percent better average latency than the HPE solution. The better latency likely contributed to the Dell EMC solution's higher OPM count and to its speed at importing to a data mart.





Conclusion

The single-purpose approach to datacenter storage solutions may have been useful in the past, but with newer technologies, your business could benefit from the reduced complexity afforded by running multiple workloads on a single solution. In our datacenter, a solution based on the all-flash Dell EMC SC5020 storage array proved capable of supporting transactional database workloads and data mart imports with better performance than a solution based on the HPE Nimble Storage AF5000 All Flash Array. The all-flash Dell EMC solution processed more orders per minute and completed data mart imports faster—capabilities that could prove valuable to organizations seeking to free their business from a siloed datacenter.

For more information about Dell EMC SC Series, visit DellEMC.com/SCSeries.

1 "Dell DVD Store Database Test Suite," accessed 5 December 2017, http://linux.dell.com/dvdstore/

On November 8, 2017, we finalized the hardware and software configurations we tested. Updates for current and recently released hardware and software appear often, so unavoidably these configurations may not represent the latest versions available when this report appears. For older systems, we chose configurations representative of typical purchases of those systems. We concluded hands-on testing on December 1, 2017.

Appendix A: System configuration information

Servers under test

Server configuration information	Dell EMC PowerEdge R740xd	HPE ProLiant DL380 Gen10
BIOS name and version	Dell 1.1.7	HPE U30 v1.02
Non-default BIOS settings	None	None
Operating system name and version/build number	VMware® ESXi™ 6.5.0 build 6765664	VMware ESXi 6.5.0 build 6765664
Date of last OS updates/patches applied	11/14/17	11/14/17
Power management policy	Default	Default
Processor		
Number of processors	2	2
Vendor and model	Intel® Xeon® Platinum 8164	Intel Xeon Platinum 8164
Core count (per processor)	26	26
Core frequency (GHz)	2.00	2.00
Stepping	H0	H0
Memory module(s)		
Total memory in system (GB)	192	192
Number of memory modules	12	12
Vendor and model	Hynix HMA82GR7AFR8N-VK	HPE SmartMemory 840756-091
Size (GB)	16	16
Type	PC4-21300	PC4-21300
Speed (MHz)	2,666	2,666
Speed running in the server (MHz)	2,666	2,666
Storage controller		
Vendor and model	PERC H730P Mini (Embedded)	HPE Smart Array P408i-a SR Gen10
Cache size (GB)	N/A	2
Firmware version	25.5.3.0005	1.04

Server configuration information	Dell EMC PowerEdge R740xd	HPE ProLiant DL380 Gen10
Local storage		
Number of drives	2	2
Drive vendor and model	Seagate® ST300MP0026	HPE EH000300JWCPK
Drive size (GB)	300	300
Drive information (speed, interface)	15k, 12Gb SAS, HDD	15k, 12Gb SAS, HDD
Network adapter		
Vendor and model	QLogic® 577xx/578xx 10 Gb Ethernet BCM57800	HPE Ethernet 1Gb 4-port 331i Adapter
Number and type of ports	2 x 10GbE, 2 x 1GbE	4 x 1GbE
Fibre Channel HBA		
Vendor and model	Emulex LightPulse® LPe31002-M6-D 2-Port 16Gb Fibre	HPE SN1100Q 16Gb 2P FC HBA
Number and type of ports	2 x 16Gb Fibre Channel	2 x 16Gb Fibre Channel
Power supplies		
Vendor and model	DELL 05RHVVA00	HPE 865408-B21
Number of power supplies	2	2
Wattage of each (W)	750	500

Storage configuration information

Storage configuration information	Dell EMC SC5020 Storage Array	HPE Nimble AF5000 All Flash array
OS version	Dell Storage 7.2.11.4	Nimble OS 5.0.1.100-526912-opt
Number of storage controllers	2	2
Number and type of ports per controller	4 x 16Gb Fibre Channel	4 x 16Gb Fibre Channel
Number of storage shelves	1	1
Number of drives per shelf	30	24
Drive vendor and model number	Toshiba® PX05SRB096Y	Samsung® MZ-7LM480N
Drive size (GB)	960	480
Drive information (speed, interface, type)	N/A, 12Gb SAS, SSD	N/A, 6Gb SAS, SSD

Appendix B: How we tested

Configuring the Brocade G620 FC switches

We configured the two switches we used for each testbed with the Brocade Web Tools.

1. Log into the switch's web tools using Internet Explorer.
2. Verify the connections to the storage controllers and hosts are online and showing a healthy status.
3. Right-click each port connected to the storage controllers, click rename, and rename Port Name to identify what it is connected to.
4. Verify all eight storage controller ports and all eight host ports are renamed.
5. On the top menu, click Configure, then Zone Admin.
6. In the new window, navigate to the Zone tab, click the arrow next to New Zone, select New Peer Zone, enter a name for the first zone, and click OK.
7. Select every port connected to the hosts, and add them to Peer Members.
8. Once all the Peer Members have been added, click Clone, name the clone, and click OK.
9. Repeat clone procedure twice more for a total of four Peer zones.
10. From the Name dropdown menu, select the first Peer Zone.
11. Select the first port connected to each of the storage controllers, and add them to Principle Members.
12. Repeat step 10 and 11 for each subsequent Peer zone, using the second, third, and fourth ports for their respective zone.
13. Click the Zone Config tab above the zone dropdown menu.
14. Click New Zone Config, enter a name for the configuration, and click OK.
15. Expand Peer Zones, click each Peer zone, and add it to the right Zone Config Members.
16. Click Save Config, and click Yes.
17. Once the commit process is finished, click Enable Config, select the newly saved zone config, click OK, and click Yes.

Installing VMware ESXi 6.5 on the host servers

We completed this installation process on all eight hosts for the HPE and Dell EMC solutions.

1. Attach the installation media.
2. Boot the server.
3. At the VMware Installer screen, press Enter.
4. At the EULA screen, to Accept and Continue, press F11.
5. Under Storage Devices, select the appropriate storage device, and press Enter.
6. For the keyboard layout, select US, and press Enter.
7. Enter the root password twice, and press Enter.
8. To start installation, press F11.
9. After the server reboots, press F2, and enter root credentials.
10. Select Configure Management Network, and press Enter.
11. Select the appropriate network adapter, and select OK.
12. Select IPv4 settings, and enter the desired IP address, subnet mask, and gateway for the server.
13. Select OK, and restart the management network.
14. Repeat steps 1 through 13 on the remaining host servers.

Deploying the VMware vCenter Server® 6.5 appliance

We deployed a VMware vCenter® instance, hosted on a separate management server.

1. Attach the installation media.
2. Navigate to the appropriate folder, and install the vcsa installation client.
3. Open the link to vcsa-setup.html found in the main folder.
4. Click Install.
5. Check the box to accept the EULA, and click Next.
6. Enter the FQDN or IP of the target server, root, the password, and click Next.
7. Click Yes.
8. Enter a name for the appliance, and an OS password. Confirm the OS password, and click Next.

9. Select Install vCenter Server with embedded Platform Services Controller, and click Next.
10. Select Create a new SSO domain.
11. Enter a vCenter SSO password, and confirm the password.
12. Enter an SSO Domain name, an SSO Site name, and click Next.
13. Select the appliance size, and click Next.
14. Select a datastore, check to enable Thin Disk Mode, and click Next.
15. Select Use an embedded database (PostgreSQL), and click Next.
16. Enter a network address, system name, subnet mask, gateway, and DNS server(s).
17. Configure time synch by entering NTP server(s), enable SSH, and click Next.
18. Click Finish.

Creating a cluster and adding the hosts to VMware vCenter

We completed the following steps on each vCenter deployment to add the eight corresponding hosts to each VMware cluster.

1. Once logged into the vCenter, navigate to Hosts and Clusters.
2. Select the primary site management vCenter.
3. Right-click the vCenter object, and select New Datacenter...
4. Enter a name for the new datacenter, and click OK.
5. Right-click the new datacenter, and click New Cluster...
6. Enter `vSAN` as the name for the new cluster.
7. Click OK.
8. Once the cluster appears, right-click the cluster, and click Add Host.
9. Enter the IP address for the first server, and click Next.
10. Enter the root credentials for the server, and click Next.
11. To accept the server's certificate, click Yes.
12. Review the server details, and click Next.
13. Assign the desired license, and click Next.
14. Disable Lockdown mode, and click Next.
15. Click Finish.
16. Repeat steps 10 through 15 for the remaining servers.

Creating volumes on the Dell EMC SC5020 storage array

Dell EMC configured the storage. From there, we used the vSphere plugin to deploy volumes as follows.

1. Log into the vSphere Web Client.
2. Right-click the Dell EMC cluster, hover over All Dell Storage Actions, and select Add Multiple Datastores…
3. Click Next.
4. Select Create New Dell Volume, and click Next.
5. Enter a base name for the OS volume, enter `91` for the volume size, and click Next.
6. Click Next.
7. Click Next.
8. Click Next.
9. Select VMFS-6 for the File System Version, and click Next.
10. Select an inventory location, and click Next.
11. Enter `20` for the number of volumes to create (1 per VM), and click Next.
12. Verify that all the volumes are configured correctly, and click Finish.
13. Repeat for the other volumes, using the following quantities and sizes:
    - 40 x 48GB database volumes (two per VM)
    - 20 x 83GB logging volumes (one per VM)
    - 8 x 484GB data mart database volumes
    - 1 x 250GB data mart OS volume
    - 1 x 364GB data mart logging volume

Creating volumes on the Nimble Storage AF5000 Array

HPE configured the storage. From there, we used the vSphere plugin to deploy volumes as follows.

1. Log into the vSphere Web Client.
2. Right-click the HPE datacenter, hover over Nimble Storage Actions, and select Create Datastore.
3. Select the correct Nimble Group, and click Next.
4. Enter a name for the OS volume, select the HPE cluster, and click Next.
5. Enter `91` for Datastore Size, ensure Enable Deduplication is unchecked, and click Next.
6. For Protection, select None, and click Next.
7. Click Next.
8. Click Next.
9. Verify that the volume is configured correctly, and click Finish.
10. Repeat until all volumes are created as detailed below.
    - 20 x 91GB OS volumes (one per VM)
    - 40 x 48GB database volumes (two per VM)
    - 20 x 83GB logging volumes (one per VM)
    - 8 x 484GB data mart database volumes
    - 1 x 250GB data mart OS volume
    - 1 x 364GB data mart logging volume

Creating the first database application VM in ESXi

1. In VMware vCenter, navigate to Virtual Machines.
2. To create a new VM, click the icon.
3. Leave Create a new virtual machine selected, and click Next.
4. Enter a name for the virtual machine, and click Next.
5. Place the VM on the desired host with available CPUs, and click Next.
6. Select the appropriate datastore to host the VM, and click Next.
7. In the Customize Hardware section, use the following settings:
    - Set the vCPU count to 8.
    - Set the Memory to 16GB and check the Reserve all guest memory (All locked) checkbox.
    - Add 1 x 70GB VMDK for OS and set to thick provision eager zeroed. Note: You will create the additional VMDKs post-clone.
    - Create three additional VMware Paravirtual SCSI controllers.
    - Attach the OS ISO to the CD/DVD drive.
8. Click Next.
9. Click Finish.

Installing Microsoft Windows Server 2016 Datacenter Edition on the first VM

1. Boot the VM to the installation media.
2. Press any key when prompted to boot from DVD.
3. When the installation screen appears, leave language, time/currency format, and input method as default, and click Next.
4. Click Install now.
5. When the installation prompts you, enter the product key.
6. Check I accept the license terms, and click Next.
7. Click Custom: Install Windows only (advanced).
8. Select Windows Server 2016 Datacenter Edition (Desktop Experience), and click Next.
9. Select Drive 0 Unallocated Space, and click Next, at which point Windows begins automatically, and restarts automatically after completing.
10. When the Settings page appears, fill in the Password and Reenter Password fields with the same password.
11. Log in with the password you set up previously.

Installing VMware Tools on the first VM

1. Right-click the VM in vCenter, and click Install VMware Tools to mount the appropriate image to the VM's virtual CD-ROM drive.
2. Ensure the VM is powered on, and log in as an administrator.
3. Navigate to the virtual CD-ROM drive in the VM, and double-click setup.exe to begin the wizard.
4. Follow the wizard, and select the Typical installation option.
5. When the VMware Tools installation has completed, restart the VM.

Installing Microsoft SQL Server 2017 on the first VM

1. Prior to installing, add the .NET Framework 3.5 feature to the server.
2. Attach the installation media ISO for SQL Server 2017 to the VM.
3. Click Run SETUP.EXE. If Autoplay does not begin the installation, navigate to the SQL Server 2017 DVD, and double-click it.
4. In the left pane, click Installation.
5. Click New SQL Server stand-alone installation or add features to an existing installation.
6. Specify Evaluation as the edition you are installing, and click Next.
7. To accept the license terms, click the checkbox, and click Next.
8. Click Use Microsoft Update to check for updates, and click Next.
9. At the Feature Selection screen, select Database Engine Services, Full-Text and Semantic Extractions for Search, Client Tools Connectivity, and Client Tools Backwards Compatibility.
10. Click Next.
11. At the Instance configuration screen, leave the default selection of default instance, and click Next.
12. At the Server Configuration screen, accept defaults, and click Next.
13. At the Database Engine Configuration screen, select the authentication method you prefer. For our testing purposes, we selected Mixed Mode.
14. Enter and confirm a password for the system administrator account.
15. Click Add Current user. This may take several seconds.
16. Click Next.
17. At the Ready to Install screen, click Install.
18. Close the installation window.
19. In the SQL Server Installation Center, click Install SQL Server Management Tools.
20. Click Download SQL Server Management Studio.
21. Click Run.
22. When the Microsoft SQL Server Management Studio screen appears, click Install.
23. When the installation completes, click Close.
24. Close the installation window.
25. Shut down the VM, and create clones for the remaining number of VMs onto corresponding OS LUNs on the storage, as well as one additional base image to be transferred to the data mart server.

Creating the additional VMDKs in ESXi

After cloning the VMs, we created VMDKs in each of the datastores for database data and log files to correspond to each VM as follows:

- 2 x 45GB VMDKs for database files, with each VMDK assigned to its own Paravirtual SCSI controller
- 1 x 80GB VMDK for database logs, assigned to the default SCSI controller to be shared with the OS VMDK

We set all VMDKs to thick provision eager zeroed.

Configuring the bulk load data mart

We used HammerDB to generate TPCH-compliant source data at scale factor 3000, for a total of 3.31 TB of raw data. The generated data exists in the form of pipe-delimited text files, which we placed on NVMe PCIe SSDs for fast reads. We split the six largest tables into 64 separate files for parallel loading. Each chunk had its own table in SQL Server, for a total of 6 x 64 one-to-one streams. We used batch scripting and SQLCMD to start 64 simultaneous SQL scripts. Each script contained BULK INSERT statements to load the corresponding chunk for each table. For example, the 17th SQL script loaded ORDERS_17.txt into table ORDERS_17, and upon finishing, began loading LINEITEM_17.txt into table LINEITEM_17, and so on through each table.

The latencies we reported for this test come from the following ESXTOP counters:

- Disk Adapter Average Driver MilliSec/Read
- Disk Adapter Average Driver MilliSec/Write

We reported the read and write latency of the data mart import server.

Generating the data

1. Download HammerDB v2.21, and run hammerdb.bat.
2. Click Options→Benchmark.
3. Select the radio buttons for MSSQL Server and TPC-H, and click OK.
4. In the left pane, expand SQL Server→TPC-H→Datagen, and double-click Options.
5. Select the radio button for 3000, enter a target location for the TPC-H-like data, and select 32 for the number of virtual users. Click OK.
6. Double-click Generate, and click Yes.

Creating the target database

We used the following SQL script to create the target database (we replaced some lines with an ellipses for clarity):

```
IF EXISTS (SELECT name FROM master.dbo.sysdatabases WHERE name = 'tpch3000')
    DROP DATABASE tpch3000
GO

CREATE DATABASE tpch3000
ON PRIMARY
(   NAME        = tpch3000_root,
    FILENAME    = 'F:\tpch\tpch_root.mdf',
    SIZE        = 100MB,
    FILEGROWTH  = 100MB),


FILEGROUP   DATA_FG_MISC
(   NAME  = tpch3000_data_ms,
    FILENAME    = 'F:\tpch\tpch_data_ms.mdf',
    SIZE        = 500MB,
    FILEGROWTH  = 100MB),

FILEGROUP DATA_FG_01 (NAME = tpch3000_data_01, FILENAME = 'F:\tpch\tpch_data_01.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_02 (NAME = tpch3000_data_02, FILENAME = 'G:\tpch\tpch_data_02.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_03 (NAME = tpch3000_data_03, FILENAME = 'H:\tpch\tpch_data_03.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_04 (NAME = tpch3000_data_04, FILENAME = 'I:\tpch\tpch_data_04.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_05 (NAME = tpch3000_data_05, FILENAME = 'J:\tpch\tpch_data_05.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_06 (NAME = tpch3000_data_06, FILENAME = 'K:\tpch\tpch_data_06.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_07 (NAME = tpch3000_data_07, FILENAME = 'L:\tpch\tpch_data_07.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_08 (NAME = tpch3000_data_08, FILENAME = 'M:\tpch\tpch_data_08.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_09 (NAME = tpch3000_data_09, FILENAME = 'F:\tpch\tpch_data_09.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_10 (NAME = tpch3000_data_10, FILENAME = 'G:\tpch\tpch_data_10.mdf', SIZE = 56320MB,
```

```
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_11 (NAME = tpch3000_data_11, FILENAME = 'H:\tpch\tpch_data_11.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_12 (NAME = tpch3000_data_12, FILENAME = 'I:\tpch\tpch_data_12.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_13 (NAME = tpch3000_data_13, FILENAME = 'J:\tpch\tpch_data_13.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_14 (NAME = tpch3000_data_14, FILENAME = 'K:\tpch\tpch_data_14.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_15 (NAME = tpch3000_data_15, FILENAME = 'L:\tpch\tpch_data_15.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_16 (NAME = tpch3000_data_16, FILENAME = 'M:\tpch\tpch_data_16.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_17 (NAME = tpch3000_data_17, FILENAME = 'F:\tpch\tpch_data_17.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_18 (NAME = tpch3000_data_18, FILENAME = 'G:\tpch\tpch_data_18.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_19 (NAME = tpch3000_data_19, FILENAME = 'H:\tpch\tpch_data_19.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_20 (NAME = tpch3000_data_20, FILENAME = 'I:\tpch\tpch_data_20.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_21 (NAME = tpch3000_data_21, FILENAME = 'J:\tpch\tpch_data_21.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_22 (NAME = tpch3000_data_22, FILENAME = 'K:\tpch\tpch_data_22.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_23 (NAME = tpch3000_data_23, FILENAME = 'L:\tpch\tpch_data_23.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_24 (NAME = tpch3000_data_24, FILENAME = 'M:\tpch\tpch_data_24.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_25 (NAME = tpch3000_data_25, FILENAME = 'F:\tpch\tpch_data_25.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_26 (NAME = tpch3000_data_26, FILENAME = 'G:\tpch\tpch_data_26.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_27 (NAME = tpch3000_data_27, FILENAME = 'H:\tpch\tpch_data_27.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_28 (NAME = tpch3000_data_28, FILENAME = 'I:\tpch\tpch_data_28.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_29 (NAME = tpch3000_data_29, FILENAME = 'J:\tpch\tpch_data_29.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_30 (NAME = tpch3000_data_30, FILENAME = 'K:\tpch\tpch_data_30.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_31 (NAME = tpch3000_data_31, FILENAME = 'L:\tpch\tpch_data_31.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_32 (NAME = tpch3000_data_32, FILENAME = 'M:\tpch\tpch_data_32.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_33 (NAME = tpch3000_data_33, FILENAME = 'F:\tpch\tpch_data_33.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_34 (NAME = tpch3000_data_34, FILENAME = 'G:\tpch\tpch_data_34.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_35 (NAME = tpch3000_data_35, FILENAME = 'H:\tpch\tpch_data_35.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_36 (NAME = tpch3000_data_36, FILENAME = 'I:\tpch\tpch_data_36.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_37 (NAME = tpch3000_data_37, FILENAME = 'J:\tpch\tpch_data_37.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_38 (NAME = tpch3000_data_38, FILENAME = 'K:\tpch\tpch_data_38.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_39 (NAME = tpch3000_data_39, FILENAME = 'L:\tpch\tpch_data_39.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_40 (NAME = tpch3000_data_40, FILENAME = 'M:\tpch\tpch_data_40.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_41 (NAME = tpch3000_data_41, FILENAME = 'F:\tpch\tpch_data_41.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_42 (NAME = tpch3000_data_42, FILENAME = 'G:\tpch\tpch_data_42.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_43 (NAME = tpch3000_data_43, FILENAME = 'H:\tpch\tpch_data_43.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_44 (NAME = tpch3000_data_44, FILENAME = 'I:\tpch\tpch_data_44.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
```

```
FILEGROUP DATA_FG_45 (NAME = tpch3000_data_45, FILENAME = 'J:\tpch\tpch_data_45.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_46 (NAME = tpch3000_data_46, FILENAME = 'K:\tpch\tpch_data_46.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_47 (NAME = tpch3000_data_47, FILENAME = 'L:\tpch\tpch_data_47.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_48 (NAME = tpch3000_data_48, FILENAME = 'M:\tpch\tpch_data_48.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_49 (NAME = tpch3000_data_49, FILENAME = 'F:\tpch\tpch_data_49.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_50 (NAME = tpch3000_data_50, FILENAME = 'G:\tpch\tpch_data_50.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_51 (NAME = tpch3000_data_51, FILENAME = 'H:\tpch\tpch_data_51.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_52 (NAME = tpch3000_data_52, FILENAME = 'I:\tpch\tpch_data_52.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_53 (NAME = tpch3000_data_53, FILENAME = 'J:\tpch\tpch_data_53.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_54 (NAME = tpch3000_data_54, FILENAME = 'K:\tpch\tpch_data_54.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_55 (NAME = tpch3000_data_55, FILENAME = 'L:\tpch\tpch_data_55.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_56 (NAME = tpch3000_data_56, FILENAME = 'M:\tpch\tpch_data_56.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_57 (NAME = tpch3000_data_57, FILENAME = 'F:\tpch\tpch_data_57.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_58 (NAME = tpch3000_data_58, FILENAME = 'G:\tpch\tpch_data_58.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_59 (NAME = tpch3000_data_59, FILENAME = 'H:\tpch\tpch_data_59.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_60 (NAME = tpch3000_data_60, FILENAME = 'I:\tpch\tpch_data_60.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_61 (NAME = tpch3000_data_61, FILENAME = 'J:\tpch\tpch_data_61.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_62 (NAME = tpch3000_data_62, FILENAME = 'K:\tpch\tpch_data_62.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_63 (NAME = tpch3000_data_63, FILENAME = 'L:\tpch\tpch_data_63.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB),
FILEGROUP DATA_FG_64 (NAME = tpch3000_data_64, FILENAME = 'M:\tpch\tpch_data_64.mdf', SIZE = 56320MB,
FILEGROWTH  = 100MB)
LOG ON
(   NAME        = tpch3000_log,
    FILENAME    = 'N:\LOG\tpch3000\tpch3000_log.ldf',
    SIZE        = 360000MB,
    FILEGROWTH  = 100MB)

GO

/*set db options*/
ALTER DATABASE tpch3000 SET RECOVERY SIMPLE
ALTER DATABASE tpch3000 SET AUTO_CREATE_STATISTICS OFF ALTER DATABASE tpch3000 SET AUTO_UPDATE_STATISTICS
OFF ALTER DATABASE tpch3000 SET PAGE_VERIFY NONE
USE tpch3000 GO

create table CUSTOMER_1 ([c_custkey] [bigint] NOT NULL,[c_mktsegment] [char](10) NULL,[c_nationkey] [int]
NULL,[c_name] [varchar](25) NULL,[c_address] [varchar](40) NULL,[c_phone] [char](15) NULL,[c_ acctbal]
[money] NULL,[c_comment] [varchar](118) NULL) on DATA_FG_01
create table CUSTOMER_2 ([c_custkey] [bigint] NOT NULL,[c_mktsegment] [char](10) NULL,[c_nationkey] [int]
NULL,[c_name] [varchar](25) NULL,[c_address] [varchar](40) NULL,[c_phone] [char](15) NULL,[c_ acctbal]
[money] NULL,[c_comment] [varchar](118) NULL) on DATA_FG_02
…
create table CUSTOMER_64 ([c_custkey] [bigint] NOT NULL,[c_mktsegment] [char](10) NULL,[c_nationkey]
[int] NULL,[c_name] [varchar](25) NULL,[c_address] [varchar](40) NULL,[c_phone] [char](15) NULL,[c_
acctbal] [money] NULL,[c_comment] [varchar](118) NULL) on DATA_FG_64

create table LINEITEM_1 ([l_shipdate] [date] NULL,[l_orderkey] [bigint] NOT NULL,[l_discount] [money] NOT
NULL,[l_extendedprice] [money] NOT NULL,[l_suppkey] [int] NOT NULL,[l_quantity] [bigint] NOT NULL,[l_
```

```
returnflag] [char](1) NULL,[l_partkey] [bigint] NOT NULL,[l_linestatus] [char](1) NULL,[l_ tax] [money]
NOT NULL,[l_commitdate] [date] NULL,[l_receiptdate] [date] NULL,[l_shipmode] [char]
(10) NULL,[l_linenumber] [bigint] NOT NULL,[l_shipinstruct] [char](25) NULL,[l_comment] [varchar](44)
NULL) on DATA_FG_01
create table LINEITEM_2 ([l_shipdate] [date] NULL,[l_orderkey] [bigint] NOT NULL,[l_discount] [money] NOT
NULL,[l_extendedprice] [money] NOT NULL,[l_suppkey] [int] NOT NULL,[l_quantity] [bigint] NOT NULL,[l_
returnflag] [char](1) NULL,[l_partkey] [bigint] NOT NULL,[l_linestatus] [char](1) NULL,[l_ tax] [money]
NOT NULL,[l_commitdate] [date] NULL,[l_receiptdate] [date] NULL,[l_shipmode] [char]
(10) NULL,[l_linenumber] [bigint] NOT NULL,[l_shipinstruct] [char](25) NULL,[l_comment] [varchar](44)
NULL) on DATA_FG_02
…
create table LINEITEM_64 ([l_shipdate] [date] NULL,[l_orderkey] [bigint] NOT NULL,[l_discount] [money]
NOT NULL,[l_extendedprice] [money] NOT NULL,[l_suppkey] [int] NOT NULL,[l_quantity] [bigint] NOT NULL,[l_
returnflag] [char](1) NULL,[l_partkey] [bigint] NOT NULL,[l_linestatus] [char](1) NULL,[l_ tax] [money]
NOT NULL,[l_commitdate] [date] NULL,[l_receiptdate] [date] NULL,[l_shipmode] [char]
(10) NULL,[l_linenumber] [bigint] NOT NULL,[l_shipinstruct] [char](25) NULL,[l_comment] [varchar](44)
NULL) on DATA_FG_64

create table ORDERS_1 ([o_orderdate] [date] NULL,[o_orderkey] [bigint] NOT NULL,[o_custkey] [bigint]
NOT NULL,[o_orderpriority] [char](15) NULL,[o_shippriority] [int] NULL,[o_clerk] [char](15) NULL,[o_
orderstatus] [char](1) NULL,[o_totalprice] [money] NULL,[o_comment] [varchar](79) NULL) on DATA_FG_01

create table ORDERS_2 ([o_orderdate] [date] NULL,[o_orderkey] [bigint] NOT NULL,[o_custkey] [bigint]
NOT NULL,[o_orderpriority] [char](15) NULL,[o_shippriority] [int] NULL,[o_clerk] [char](15) NULL,[o_
orderstatus] [char](1) NULL,[o_totalprice] [money] NULL,[o_comment] [varchar](79) NULL) on DATA_FG_02
…
create table ORDERS_64 ([o_orderdate] [date] NULL,[o_orderkey] [bigint] NOT NULL,[o_custkey] [bigint]
NOT NULL,[o_orderpriority] [char](15) NULL,[o_shippriority] [int] NULL,[o_clerk] [char](15) NULL,[o_
orderstatus] [char](1) NULL,[o_totalprice] [money] NULL,[o_comment] [varchar](79) NULL) on DATA_FG_64

create table PART_1 ([p_partkey] [bigint] NOT NULL,[p_type] [varchar](25) NULL,[p_size] [int] NULL,[p_
brand] [char](10) NULL,[p_name] [varchar](55) NULL,[p_container] [char](10) NULL,[p_mfgr] [char](25)
NULL,[p_retailprice] [money] NULL,[p_comment] [varchar](23) NULL) on DATA_FG_01  create table PART_2 ([p_
partkey] [bigint] NOT NULL,[p_type] [varchar](25) NULL,[p_size] [int] NULL,[p_brand] [char](10) NULL,[p_
name] [varchar](55) NULL,[p_container] [char](10) NULL,[p_mfgr] [char](25) NULL,[p_retailprice] [money]
NULL,[p_comment] [varchar](23) NULL) on DATA_FG_02
…
create table PART_64 ([p_partkey] [bigint] NOT NULL,[p_type] [varchar](25) NULL,[p_size] [int] NULL,[p_
brand] [char](10) NULL,[p_name] [varchar](55) NULL,[p_container] [char](10) NULL,[p_mfgr] [char](25)
NULL,[p_retailprice] [money] NULL,[p_comment] [varchar](23) NULL) on DATA_FG_64

create table PARTSUPP_1 ([ps_partkey] [bigint] NOT NULL,[ps_suppkey] [int] NOT NULL,[ps_supplycost]
[money] NOT NULL,[ps_availqty] [int] NULL,[ps_comment] [varchar](199) NULL) on DATA_FG_01
create table PARTSUPP_2 ([ps_partkey] [bigint] NOT NULL,[ps_suppkey] [int] NOT NULL,[ps_supplycost]
[money] NOT NULL,[ps_availqty] [int] NULL,[ps_comment] [varchar](199) NULL) on DATA_FG_02
…
create table PARTSUPP_64 ([ps_partkey] [bigint] NOT NULL,[ps_suppkey] [int] NOT NULL,[ps_supplycost]
[money] NOT NULL,[ps_availqty] [int] NULL,[ps_comment] [varchar](199) NULL) on DATA_FG_64

create table SUPPLIER_1 ([s_suppkey] [int] NOT NULL,[s_nationkey] [int] NULL,[s_comment] [varchar]
(102) NULL,[s_name] [char](25) NULL,[s_address] [varchar](40) NULL,[s_phone] [char](15) NULL,[s_ acctbal]
[money] NULL) on DATA_FG_01
create table SUPPLIER_2 ([s_suppkey] [int] NOT NULL,[s_nationkey] [int] NULL,[s_comment] [varchar]
(102) NULL,[s_name] [char](25) NULL,[s_address] [varchar](40) NULL,[s_phone] [char](15) NULL,[s_ acctbal]
[money] NULL) on DATA_FG_02
…
create table SUPPLIER_64 ([s_suppkey] [int] NOT NULL,[s_nationkey] [int] NULL,[s_comment] [varchar]
(102) NULL,[s_name] [char](25) NULL,[s_address] [varchar](40) NULL,[s_phone] [char](15) NULL,[s_ acctbal]
[money] NULL) on DATA_FG_64
```

Inserting the data into Microsoft SQL Server
We used 64 individual SQL scripts to create a BULK INSERT process on each filegroup. The first script is shown here as an example:

```
bulk insert tpch3000..CUSTOMER_1 from 'O:\CUSTOMER_1.tbl' with
(TABLOCK,DATAFILETYPE='char',CODEPAGE='raw',FieldTerminator='|',BATCHSIZE=14062500)
bulk insert tpch3000..LINEITEM_1 from 'O:\LINEITEM_1.tbl' with
(TABLOCK,DATAFILETYPE='char',CODEPAGE='raw',FieldTerminator='|',BATCHSIZE=562500000)
bulk insert tpch3000..ORDERS_1 from 'O:\ORDERS_1.tbl' with
```

```
(TABLOCK,DATAFILETYPE='char',CODEPAGE='raw',FieldTerminator='|',BATCHSIZE=140625000)
bulk insert tpch3000..PART_1 from 'O:\PART_1.tbl' with
(TABLOCK,DATAFILETYPE='char',CODEPAGE='raw',FieldTerminator='|',BATCHSIZE=18750000)
bulk insert tpch3000..PARTSUPP_1 from 'O:\PARTSUPP_1.tbl' with
(TABLOCK,DATAFILETYPE='char',CODEPAGE='raw',FieldTerminator='|',BATCHSIZE=75000000)
bulk insert tpch3000..SUPPLIER_1 from 'O:\SUPPLIER_1.tbl' with
(TABLOCK,DATAFILETYPE='char',CODEPAGE='raw',FieldTerminator='|',BATCHSIZE=937500)
```

Starting the SQL BULK INSERT scripts

We used Windows CMD and SQLCMD to start the 64 BULK INSERT scripts with CPU affinity:

```
START /NODE 0 /AFFINITY 1 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_1.sql
START /NODE 0 /AFFINITY 2 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_2.sql
START /NODE 0 /AFFINITY 4 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_3.sql
START /NODE 0 /AFFINITY 8 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_4.sql
START /NODE 0 /AFFINITY 16 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_5.sql
START /NODE 0 /AFFINITY 32 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_6.sql
START /NODE 0 /AFFINITY 64 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_7.sql
START /NODE 0 /AFFINITY 128 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_8.sql
START /NODE 0 /AFFINITY 256 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_9.sql
START /NODE 0 /AFFINITY 512 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_10.sql
START /NODE 0 /AFFINITY 1024 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_11.sql
START /NODE 0 /AFFINITY 2048 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_12.sql
START /NODE 0 /AFFINITY 4096 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_13.sql
START /NODE 0 /AFFINITY 8192 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_14.sql
START /NODE 0 /AFFINITY 16384 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_15.sql
START /NODE 0 /AFFINITY 32768 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_16.sql
START /NODE 0 /AFFINITY 65536 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_17.sql
START /NODE 0 /AFFINITY 131072 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_18.sql
START /NODE 0 /AFFINITY 262144 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_19.sql
START /NODE 0 /AFFINITY 524288 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_20.sql
START /NODE 0 /AFFINITY 1048576 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_21.sql
START /NODE 0 /AFFINITY 2097152 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_22.sql
START /NODE 0 /AFFINITY 4194304 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_23.sql
START /NODE 0 /AFFINITY 8388608 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_24.sql
START /NODE 0 /AFFINITY 16777216 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_25.sql
START /NODE 0 /AFFINITY 33554432 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_26.sql
START /NODE 0 /AFFINITY 67108864 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_27.sql
START /NODE 0 /AFFINITY 134217728 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_28.sql
START /NODE 0 /AFFINITY 268435456 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_29.sql
START /NODE 0 /AFFINITY 536870912 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_30.sql
START /NODE 0 /AFFINITY 1073741824 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_31.sql
START /NODE 0 /AFFINITY 2147483648 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_32.sql
START /NODE 1 /AFFINITY 1 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_33.sql
START /NODE 1 /AFFINITY 2 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_34.sql
START /NODE 1 /AFFINITY 4 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_35.sql
START /NODE 1 /AFFINITY 8 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_36.sql
START /NODE 1 /AFFINITY 16 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_37.sql
START /NODE 1 /AFFINITY 32 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_38.sql
START /NODE 1 /AFFINITY 64 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_39.sql
START /NODE 1 /AFFINITY 128 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_40.sql
START /NODE 1 /AFFINITY 256 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_41.sql
START /NODE 1 /AFFINITY 512 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_42.sql
START /NODE 1 /AFFINITY 1024 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_43.sql
START /NODE 1 /AFFINITY 2048 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_44.sql
START /NODE 1 /AFFINITY 4096 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_45.sql
START /NODE 1 /AFFINITY 8192 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_46.sql
START /NODE 1 /AFFINITY 16384 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_47.sql
START /NODE 1 /AFFINITY 32768 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_48.sql
START /NODE 1 /AFFINITY 65536 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_49.sql
START /NODE 1 /AFFINITY 131072 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_50.sql
START /NODE 1 /AFFINITY 262144 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_51.sql
START /NODE 1 /AFFINITY 524288 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_52.sql
START /NODE 1 /AFFINITY 1048576 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_53.sql
START /NODE 1 /AFFINITY 2097152 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_54.sql
START /NODE 1 /AFFINITY 4194304 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_55.sql
START /NODE 1 /AFFINITY 8388608 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_56.sql
```

```
START /NODE 1 /AFFINITY 16777216 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_57.sql
START /NODE 1 /AFFINITY 33554432 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_58.sql
START /NODE 1 /AFFINITY 67108864 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_59.sql
START /NODE 1 /AFFINITY 134217728 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_60.sql
START /NODE 1 /AFFINITY 268435456 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_61.sql
START /NODE 1 /AFFINITY 536870912 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_62.sql
START /NODE 1 /AFFINITY 1073741824 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_63.sql
START /NODE 1 /AFFINITY 2147483648 SQLCMD -S localhost -d tpch3000 -U sa -P Password1 -i C:\scripts\gen_64.sql
```

Configuring DVD Store 2

Data generation overview

We generated the data using the Install.pl script included with DVD Store version 2.1 (DS2), providing the parameters for our 40GB database size and the Microsoft SQL Server 2016 platform. We ran the Install.pl script on a utility system running Linux®. The Install.pl script also generated the database schema.

After processing the data generation, we transferred the data files and schema creation files to a Windows-based system running SQL Server 2014. We built the 40GB database in SQL Server 2014, and then performed a full backup, storing the backup file on the C: drive for quick access. We used that backup file to restore the database to the SQL Server 2016-based VMs between test runs.

The only modification we made to the schema creation scripts were the specified file sizes for our database. We explicitly set the file sizes higher than necessary to ensure that no file-growth activity would affect the outputs of the test. Besides this file size modification, the database schema was created and loaded according to the DVD Store documentation. Specifically, we followed the steps below:

1. We generated the data and created the database and file structure using database creation scripts in the DS2 download. We made size modifications specific to our 40GB database and the appropriate changes to drive letters.
2. We transferred the files from our Linux data generation system to a Windows system running SQL Server.
3. We created database tables, stored procedures, and objects using the provided DVD Store scripts.
4. We set the database recovery model to bulk-logged to prevent excess logging.
5. We loaded the data we generated into the database. 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.
6. We created indices, full-text catalogs, primary keys, and foreign keys using the database-creation scripts.
7. We updated statistics on each table according to database-creation scripts, which sample 18 percent of the table data.
8. On the SQL Server instance, we created a ds2user SQL Server login using the following Transact-SQL (T-SQL) script:

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

9. We set the database recovery model back to full.
10. We created the necessary full text index using SQL Server Management Studio.
11. We created a database user and mapped this user to the SQL Server login.
12. We then performed a full backup of the database. This backup allowed us to restore the databases to a pristine state relatively quickly between tests.

Logical name	Filegroup	Initial size (MB)
Database files		
primary	PRIMARY	4
cust1	DS_CUST_FG	6,000
cust2	DS_CUST_FG	6,000
cust3	DS_CUST_FG	6,000
cust4	DS_CUST_FG	6,000
ind1	DS_IND_FG	3,205
ind2	DS_IND_FG	3,205
ind3	DS_IND_FG	3,205

Logical name	Filegroup	Initial size (MB)
ind4	DS_IND_FG	3,205
ds_misc	DS_MISC_FG	200
orders1	DS_ORDERS	3,000
orders2	DS_ORDERS	3,000
orders3	DS_ORDERS	3,000
orders4	DS_ORDERS	3,000
Log files		
ds_log	Not applicable	24,781

Configuring the database workload client

For our testing, we used a virtual client for the Microsoft SQL Server client. To create this client, we installed Windows Server 2012, assigned a static IP address, and installed .NET 3.5.

Running the DVD Store tests

We created a series of batch files, SQL scripts, and shell scripts to automate the complete test cycle. DVD Store outputs an orders-per-minute metric, which is a running average calculated through the test. In this report, we report the last OPM reported by each client/target pair. We ran the test three times and reported the median run from each solution. Each complete test cycle consisted of the following general steps:

1. Clean up prior outputs from the target system and the client driver system.
2. Drop the databases from the target.
3. Restore the databases on the target.
4. Shut down the target.
5. Reboot the host and client system.
6. Wait for a ping response from the server under test and the client system.
7. Let the test server idle for 10 minutes.
8. Start the DVD Store driver on the client.

We used the following DVD Store parameters for testing:

```
ds2sqlserverdriver.exe --target=<target_IP> --ramp_rate=10 --run_time=180 --n_threads=32 --db_ size=40GB
--think_time=0 --detailed_view=Y --warmup_time=15 --report_rate=1 --csv_output=<drive path>
```

The latencies we reported for this test come from the following ESXTOP counters:

- Disk Adapter Average Driver MilliSec/Read
- Disk Adapter Average Driver MilliSec/Write

We reported the average of the read and write latencies for the three transactional database workload servers.

This project was commissioned by Dell EMC.