



The science behind the report:

Cut memory costs while adding capacity with Intel Xeon 6 and VMware Cloud Foundation 9.1

This document describes what we tested, how we tested, and what we found. To learn how these facts translate into real-world benefits, read the report [Cut memory costs while adding capacity with Intel Xeon 6 and VMware Cloud Foundation 9.1](#).

We concluded our hands-on testing on June 21, 2026. During testing, we determined the appropriate hardware and software configurations and applied updates as they became available. The results in this report reflect configurations that we finalized on June 18, 2026 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

Our results

To learn more about how we have calculated the wins in this report, go to <https://facts.pt/calculating-and-highlighting-wins>. Unless we state otherwise, we have followed the rules and principles we outline in that document.

Table 1: Memory-related hardware specifications and costs. We collected costs from Dell.com on June 8, 2026. Source: PT.

	512 GB of DRAM configuration	Memory-tiered configuration	1 TB of DRAM configuration
Memory			
1 TB of DRAM configuration	512	512	1,024
DIMM size (DDR5 6,400 MT/s) (GB)	32	32	64
Quantity	16	16	16
Unit cost	\$3,609.76	\$3,609.76	\$6,629.83
Memory cost subtotal	\$57,756.16	\$57,756.16	\$106,077.28

	512 GB of DRAM configuration	Memory-tiered configuration	1 TB of DRAM configuration
Storage — data (NVMe E3.S Gen5)			
Chassis	E3.S 8-slot NVMe Direct	E3.S 8-slot NVMe Direct	E3.S 8-slot NVMe Direct
Chassis delta	-\$5.72	-\$5.72	-\$5.72
Drive 1 — vSAN cache tier	3.2 TB MU NVMe E3.S Gen5	3.2 TB MU NVMe E3.S Gen5	3.2 TB MU NVMe E3.S Gen5
Drive 2 — vSAN capacity tier	3.2 TB MU NVMe E3.S Gen5	3.2 TB MU NVMe E3.S Gen5	3.84 TB RI NVMe E3.S Gen5
Drive 3 — vSAN capacity tier	3.84 TB RI NVMe E3.S Gen5	3.84 TB RI NVMe E3.S Gen5	3.84 TB RI NVMe E3.S Gen5
Drive 4 — memory tiering	N/A	3.84 TB RI NVMe E3.S Gen5	N/A
Quantity	3	4	3
3.2TB drive cost	\$8,389.09	\$8,389.09	\$8,389.09
3.84TB drive cost	\$10,783.48	\$10,783.48	\$10,783.48
Storage cost subtotal	\$27,561.66	\$38,345.14	\$29,956.05
Memory-related hardware costs total	\$57,756.16	\$68,539.64	\$106,077.28

Table 2: Transactional database results of our testing.

	512 GB of DRAM configuration	Memory-tiered configuration	1 TB of DRAM configuration
Databse VMs	6	12	12
Total NOPM	1,719,646	2,595,550	3,054,071
Percentage win	N/A	50%*	15%**
CPU utilization (%)	51	85	87

*vs. 512GB DRAM config

**vs. Memory-tiered config

Table 3: Server specifications and costs. We collected costs from Dell.com on June 8, 2026. Source: PT.

	512 GB of DRAM configuration	Memory-tiered configuration	1 TB of DRAM configuration
Total server acquisition cost			
Server			
Dell™ PowerEdge™ R770 base cost	\$41,999.02	\$41,999.02	\$41,999.02
Processor			
Intel® Xeon® 6745P primary (delta over base 6507P) cost	\$4,389.34	\$4,389.34	\$4,389.34
Intel Xeon 6745P secondary (additional socket) cost	\$4,389.34	\$4,389.34	\$4,389.34

	512 GB of DRAM configuration	Memory-tiered configuration	1 TB of DRAM configuration
Memory			
Total capacity (GB)	512	512	1,024
DIMM size (DDR5 6,400 MT/s) (GB)	32	32	64
Quantity	16	16	16
Unit cost	\$3,609.76	\$3,609.76	\$6,629.83
Memory cost subtotal	\$57,756.16	\$57,756.16	\$106,077.28
Storage - boot			
BOSS-N1 + 2× M.2 480 GB RAID 1 cost	\$7,846.14	\$7,846.14	\$7,846.14
Storage — data (NVMe E3.S Gen5)			
Chassis	E3.S 8-slot NVMe Direct	E3.S 8-slot NVMe Direct	E3.S 8-slot NVMe Direct
Chassis delta	-\$5.72	-\$5.72	-\$5.72
Drive 1 — vSAN cache tier	3.2 TB MU NVMe E3.S Gen5	3.2 TB MU NVMe E3.S Gen5	3.2 TB MU NVMe E3.S Gen5
Drive 2 — vSAN capacity tier	3.2 TB MU NVMe E3.S Gen5	3.2 TB MU NVMe E3.S Gen5	3.84 TB RI NVMe E3.S Gen5
Drive 3 — vSAN capacity tier	3.84 TB RI NVMe E3.S Gen5	3.84 TB RI NVMe E3.S Gen5	3.84 TB RI NVMe E3.S Gen5
Drive 4 — memory tiering	N/A	3.84 TB RI NVMe E3.S Gen5	N/A
Quantity	3	4	3
3.2TB drive cost	\$8,389.09	\$8,389.09	\$8,389.09
3.84TB drive cost	\$10,783.48	\$10,783.48	\$10,783.48
Storage cost subtotal	\$27,561.66	\$38,345.14	\$29,956.05
Networking			
OCP NIC delta (ConnectX-6 Dx 100GbE)	\$825.32	\$825.32	\$825.32
Power Supply			
Dual fully redundant 1500W MM Titanium	\$639.50	\$639.50	\$639.50
Estimated total server acquisition cost	\$145,400.76	\$156,189.96	\$196,121.99
Total NOPM/estimated total server acquisition dollar	11.8	16.62	15.57
Percentage win for the memory-tiered configuration		6%	

System configuration information

Table 4: Detailed information on the systems we tested.

System configuration information	Dell PowerEdge R770 (512 GB of memory)	Dell PowerEdge R770 (1 TB of memory)	Dell PowerEdge R770 (memory tiering)
BIOS name and version	Dell 1.8.5	Dell 1.8.5	Dell 1.8.5
Non-default BIOS settings	LOM Enabled	LOM Enabled	LOM Enabled
Operating system name and version/build number	VMware® ESXi 9.1.0.0.25370933	VMware ESXi 9.1.0.0.25370933	VMware ESXi 9.1.0.0.25370933
Date of last OS updates/patches applied	05/22/2026	05/22/2026	05/22/2026
System profile setting	Performance	Performance	Performance
Processor			
Number of processors	2	2	2
Vendor and model	Intel® Xeon® 6745P	Intel Xeon 6745P	Intel Xeon 6745P
Core count (per processor)	32	32	32
Core frequency (GHz)	3.60	3.60	3.60
Stepping	1	1	1
Memory module(s)			
Total memory in system (GB)	512	1,024	512
Number of memory modules	16	16	16
Vendor and model	Micron MTC20F1045S1RC64BDY	Samsung M321R8GA0PB2-CCPEC	Hynix HMC888AHBRA471N
Size (GB)	32	64	32
Type	DDR5	DDR5	DDR5
Speed (MT/s)	6,400	6,400	6,400
Speed running in the server (MT/s)	6,400	6,400	6,400
Storage controller			
Vendor and model	Dell BOSS-N1 DC-MHS	Dell BOSS-N1 DC-MHS	Dell BOSS-N1 DC-MHS
Cache size	N/A	N/A	N/A
Firmware version	2.2.13.2038	2.2.13.2038	2.2.13.2038
Local storage			
Number of drives	2	2	2
Drive vendor and model	Dell EC NVMe® ISE 7450 RI M.2	Dell EC NVMe ISE 7450 RI M.2	Dell EC NVMe ISE 7450 RI M.2
Drive size (GB)	480	480	480
Drive information (speed, interface, type)	M.2, NVMe	M.2, NVMe	M.2, NVMe

System configuration information	Dell PowerEdge R770 (512 GB of memory)	Dell PowerEdge R770 (1 TB of memory)	Dell PowerEdge R770 (memory tiering)
Memory tiering storage			
Number of drives	N/A	N/A	1
Drive vendor and model	N/A	N/A	Dell NVMe PM1743 RI E3.S
Drive size (TB)	N/A	N/A	3.84
Drive information (speed, interface, type)	N/A	N/A	PCIe® Gen5, NVMe 2.0
vSAN ESA shared storage			
Number of drives	12		
Drive vendor and model	4x Dell NVMe PM1743 RI E3.S 8x Dell NVMe PM1745 MU E3.S		
Drive size	4x 3.84 TB 8x 3.2 TB		
Drive information (speed, interface, type)	PCIe Gen5, NVMe 2.0		
Network adapter			
Vendor and model	NVIDIA® ConnectX®-6 Dx 2x 100G QSFP56	NVIDIA ConnectX-6 Dx 2x 100G QSFP56	NVIDIA ConnectX-6 Dx 2x 100G QSFP56
Number and type of ports	2x 100Gbps	2x 100Gbps	2x 100Gbps
Firmware version	22.44.10.36	22.44.10.36	22.44.10.36

How we tested

Our goal was to show that by using memory tiering in VMware vSphere 9.1 instead of increasing physical memory, organizations can improve online transaction processing (OLTP) database performance. To demonstrate the cost and performance advantages of memory tiering, we compared three server memory configurations, starting with a baseline 512GB memory configuration and no tiering. Our memory tiering configuration had the same memory capacity but with an additional dedicated NVMe drive for memory tiering. The third configuration had 1 TB of memory.

We used three Intel Xeon 6745P processor-powered Dell PowerEdge R770 servers from a VMware vSAN ESA cluster for testing, applying a different memory configuration to each. The servers ran Microsoft Windows Server 2025 VMs with Microsoft SQL Server 2025 using an 800-warehouse TPROC-C database from HammerDB 5.0, our testing tool. Our client VMs ran Ubuntu® 24.04 on a dedicated infrastructure server. Each server under test had a dedicated 100Gbps network connection for vSAN and another dedicated 100Gbps connection for HammerDB traffic.

We configured the SQL Server VMs with 4 vCPUs and 80 GB of RAM. Each configuration used 93 percent of its memory. After configuring our baseline SQL Server and client VMs, we cloned them out to each server under test. We enabled a 512GB memory tier on the additional 3.84 TB NVMe drive in the server for the memory tiering configuration. After configuring memory tiering, we ran a HammerDB TPROC-C workload three times on each configuration, reporting the median result.

Configuring the VMware vSphere 9.1 environment

Installing VMware vSphere 9.1

1. Boot to the VMware vSphere 9.1 installation media.
2. To continue, press Enter.
3. To accept the license agreement, press F11.
4. Select the installation location.
5. Select a language, and create the root password.
6. To install, press F11.

Installing VMware vCenter 9.1

1. Double-click the vCenter Server Appliance ISO to mount it.
2. Browse to the vcsa-ui-installer → win32 folder, and run installer.exe.
3. On the launch menu, choose Install.
4. On the Introduction tab, click Next.
5. Accept the license agreement, and click Next.
6. Enter the name of the esxi host on which to deploy vCenter, a username, and password, and click Next.
7. Click Accept.
8. Give the vCenter VM a name and root password, and click Next.
9. Select the deployment size, and click Next.
10. Select the datastore, and click Next.
11. Select DHCP for the IP assignment, and click Next.
12. Click Finish.
13. After Stage 1 is complete, click Continue to move to Stage 2.
14. Click Next.
15. Configure Time synchronization and SSH access, and click Next.
16. Assign an SSO domain name and password, and click Next.
17. Uncheck the box to join CEIP, and click Next.
18. Click Finish.
19. Click OK.

Enabling memory tiering on one of the vSphere hosts

1. Place the host into maintenance mode.
2. SSH to the host in maintenance mode.
3. List the NVMe devices on the host.

```
esxcli storage core adapter device list
```

4. Take note of the device name at path /vmfs/devices/disks/<nvme device name>
5. Create a memory tier partition on the device:

```
esxcli system tierdevice create -d /vmfs/devices/disks/<nvme device name>
```

6. Verify that the partition exists:

```
esxcli system tierdevice list
```

7. In the vSphere UI, click the host in maintenance mode.
8. Click Configure → Advanced System Settings.
9. Click the Key filter, and type VMKernel.Boot.memoryTiering
10. Click Edit, and change the value to True.
11. Click OK.
12. Reboot the host.
13. When the host has rebooted, exit out of maintenance mode.
14. Verify the host is using memory tiering by selecting the host and looking in the Memory section on the Summary tab.

Creating a Windows VM

1. Log into the vCenter.
2. Right-click the cluster → New Virtual Machine....
3. Highlight Create a new virtual machine, and click Next.
4. Give the VM a name, and click Next.
5. Select the cluster for the compute resource, and click Next.
6. Select the vSAN datastore, and click Next.
7. Set the VM compatibility to ESXi 9.0 and later, and click Next.
8. Set the VM Guest OS version to Microsoft Windows Server 2025 (64-bit), and click Next.
9. Customize the following VM hardware:
 - CPU: 4
 - Memory: 80 GB
 - New Hard Disk: 100 GB
10. Click Next.
11. Click Finish.

Creating an Ubuntu client VM

1. Log into the vCenter.
2. Right-click the cluster → New Virtual Machine....
3. Highlight Create a new virtual machine, and click Next.
4. Give the VM a name, and click Next.
5. Select the cluster for the compute resource, and click Next.
6. Select the vSAN datastore, and click Next.
7. Set the VM compatibility to ESXi 9.0 and later, and click Next.
8. Set the VM Guest OS version to Ubuntu Linux (64-bit), and click Next.
9. Customize the following VM hardware:
 - CPU: 16
 - Memory: 24 GB
 - New Hard Disk: 100 GB
10. Click Next.
11. Click Finish.

Configuring the test environment

Installing Windows Server 2025 on a VM

1. Attach the Windows Server 2025 ISO (boot volume for OCP) to the VM.
2. Open the VM console, and start the VM.
3. When prompted to boot from DVD, press any key.
4. On the installation screen, leave language, time/currency format, and input method as default, and click Next.
5. Click Install now.
6. When prompted, enter the product key.
7. Select Windows Server 2025 Datacenter Edition (Desktop Experience), and click Next.
8. Accept the license terms, and click Next.
9. Choose Custom: Install Windows only (advanced).
10. Select Drive 0 Unallocated Space, and to begin installation, click Next.
11. After installation completes and the system restarts, set a password on the Settings page, and log in.
12. Install VMware Tools in the VM.
13. From Server Manager, disable Windows Firewall.
14. Enable SSH.
15. Run Windows Updates to ensure the system is up to date.

After installing Windows, we added eight 20 GB disks for database data, one 100 GB disk for database logs, one 100 GB disk for database backups, one 50 GB disk for TempDB data, and one 25 GB disk for TempLogs on the SQL VM. We initialized and formatted the disks with a 64KB allocation unit size in Disk Management before installing SQL Server. We also set the visual effects in the performance options to adjust for best performance, and the power plan to High performance. Finally, we created a perfmon user-defined data collector set to collect performance metrics for each VM during the run.

Installing SQL Server on Windows Server 2025 VM

1. Attach the SQL Server 2025 installation media ISO to the VM.
2. Double-click the Setup application.
3. In the left pane, click Installation.
4. Click New SQL Server stand-alone installation or add features to an existing installation.
5. On the Edition window:
 - Keep Specify a Free Edition checked.
 - Select Evaluation, and click Next.
6. Accept the license terms, and click Next.
7. Click Use Microsoft Update to check for updates, and click Next.
8. On the Install Rules page, click Next.
9. On the Azure Extension for SQL Server page, deselect the extension, and click Next.
10. On the Feature Selection screen, check:
 - Database Engine Services
 - Full-Text and Semantic Extractions for Search
11. Click Next.
12. Leave the default instance selected, and click Next.
13. Leave the default Service Accounts, and click Next.
14. On the Server Configuration tab:
 - Choose Mixed Mode.
 - Enter and confirm a password for the sa account.
 - Click Add Current User to specify SQL Server administrators.
 - On the TempDB tab:
 - Set TempDB location to the TempDB volume.
 - Set TempLog location to the TempLog volume.
 - Click Next.
15. On the Error and usage reporting screen, click Next.
16. On the Installation Configuration Rules screen, verify no failures or warnings, and click Next.

17. On the Ready to Install screen, click Install.
18. Close the installation window when complete.
19. Download and install SQL Server Management Studio (SSMS).
20. Open SSMS, and run the following SQL queries to update SQL Server for HammerDB:

```
exec sp_configure 'show advanced options', '1'
reconfigure with override
exec sp_configure 'recovery interval','32767'
exec sp_configure 'max degree of parallelism','1'
exec sp_configure 'lightweight pooling','1'
exec sp_configure 'priority boost','1'
exec sp_configure 'max worker threads','3000'
exec sp_configure 'default trace enabled','0'
exec sp_configure 'network packet size','8192'
exec sp_configure 'remote query timeout', 0
go
reconfigure with override
```

21. Create a database for HammerDB TPROC-C:

```
CREATE DATABASE [tpcc]
CONTAINMENT = NONE
ON PRIMARY
( NAME = N'tpcc01', FILENAME = N'E:\data\tpcc01.mdf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc02', FILENAME = N'F:\data\tpcc02.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc03', FILENAME = N'G:\data\tpcc03.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc04', FILENAME = N'H:\data\tpcc04.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc05', FILENAME = N'I:\data\tpcc05.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc06', FILENAME = N'J:\data\tpcc06.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc07', FILENAME = N'K:\data\tpcc07.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB ),
( NAME = N'tpcc08', FILENAME = N'L:\data\tpcc08.ndf' , SIZE = 8192KB , FILEGROWTH = 65536KB )
LOG ON
( NAME = N'tpcc_log', FILENAME = N'M:\logs\tpcc_log.ldf' , SIZE = 8192KB , FILEGROWTH = 65536KB )
WITH LEDGER = OFF
GO
ALTER DATABASE [tpcc] SET COMPATIBILITY_LEVEL = 170
GO
ALTER DATABASE [tpcc] SET ANSI_NULL_DEFAULT OFF
GO
ALTER DATABASE [tpcc] SET ANSI_NULLS OFF
GO
ALTER DATABASE [tpcc] SET ANSI_PADDING OFF
GO
ALTER DATABASE [tpcc] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [tpcc] SET ARITHABORT OFF
GO
ALTER DATABASE [tpcc] SET AUTO_CLOSE OFF
GO
ALTER DATABASE [tpcc] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [tpcc] SET AUTO_CREATE_STATISTICS ON(INCREMENTAL = OFF)
GO
ALTER DATABASE [tpcc] SET AUTO_UPDATE_STATISTICS ON
GO
ALTER DATABASE [tpcc] SET CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [tpcc] SET CURSOR_DEFAULT GLOBAL
GO
ALTER DATABASE [tpcc] SET CONCAT_NULL_YIELDS_NULL OFF
GO
ALTER DATABASE [tpcc] SET NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [tpcc] SET QUOTED_IDENTIFIER OFF
GO
ALTER DATABASE [tpcc] SET RECURSIVE_TRIGGERS OFF
GO
ALTER DATABASE [tpcc] SET DISABLE_BROKER
GO
```

```

ALTER DATABASE [tpcc] SET AUTO_UPDATE_STATISTICS_ASYNC OFF
GO
ALTER DATABASE [tpcc] SET DATE_CORRELATION_OPTIMIZATION OFF
GO
ALTER DATABASE [tpcc] SET PARAMETERIZATION SIMPLE
GO
ALTER DATABASE [tpcc] SET READ_COMMITTED_SNAPSHOT OFF
GO
ALTER DATABASE [tpcc] SET READ_WRITE
GO
ALTER DATABASE [tpcc] SET RECOVERY FULL
GO
ALTER DATABASE [tpcc] SET MULTI_USER
GO
ALTER DATABASE [tpcc] SET PAGE_VERIFY CHECKSUM
GO
ALTER DATABASE [tpcc] SET TARGET_RECOVERY_TIME = 60 SECONDS
GO
ALTER DATABASE [tpcc] SET DELAYED_DURABILITY = DISABLED
GO
USE [tpcc]
GO
IF NOT EXISTS (SELECT name FROM sys.filegroups WHERE is_default=1 AND name = N'PRIMARY') ALTER
DATABASE [tpcc] MODIFY FILEGROUP [PRIMARY] DEFAULT
GO

```

Installing Ubuntu 24.04 on the client VM

1. Boot the VM to the Ubuntu Server 24.04 LTS installation media.
2. When prompted, select Install Ubuntu.
3. Select the desired language, and click Done.
4. Choose a keyboard layout, and click Done.
5. At the Network Connections screen, click Done.
6. At the Configure Proxy screen, click Done.
7. At the Configure Ubuntu Archive Mirror screen, click Done.
8. Select Use an entire disk, and click Done.
9. Click Continue.
10. Enter user account details, and click Done.
11. Enable OpenSSH Server install, and click Done.
12. At the installation summary screen, click Done.
13. When the installation finishes, unmount the installation media, and reboot the VM.

Configuring the client VM OS

1. Log into the client VM.
 2. Disable the firewall.
3. Add your user to the sudoers file by editing /etc/sudoers and adding the following line to the end of the file:

```
sudo ufw disable
```

```
<Username> ALL=(ALL:ALL) NOPASSWD: ALL
```

4. Generate an SSH key:

```
ssh-keygen -t rsa -b 4096
```

5. Add the SSH key to ~/.ssh/authorized_keys.
6. Update the VM:

```
sudo apt update  
sudo apt upgrade
```

7. Reboot the VM:

```
sudo reboot
```

8. Install Open VM tools:

```
sudo apt install open-vm-tools
```

9. Add the Microsoft Ubuntu repository:

```
sudo curl -sSL -O https://packages.microsoft.com/config/ubuntu/24.04/packages-microsoft-prod.deb  
sudo dpkg -i packages-microsoft-prod.deb
```

10. Update the VM again:

```
sudo apt update
```

11. Install additional tools:

```
sudo apt install wget curl vim tar zip unzip nmon sysstat tcsh mssql-tools18
```

12. Add the SQL Server tools to the console path:

```
echo 'export PATH="$PATH:/opt/mssql-tools18/bin"' >> ~/.bash_profile  
echo 'export PATH="$PATH:/opt/mssql-tools18/bin"' >> ~/.bashrc
```

Installing HammerDB 5.0

1. Download the HammerDB version 5.0 for Ubuntu 24.04:

```
wget https://github.com/TPC-Council/HammerDB/releases/download/v5.0/HammerDB-5.0-Prod-Lin-UBU24.tar.gz
```

2. Unpack the tar file:

```
tar -xzf HammerDB-5.0-Prod-Lin-UBU24.tar.gz
```

3. Reboot the VM:

```
sudo reboot
```

Generating an 800-warehouse TPROC-C database

1. Log into the client VM.
2. Navigate to the HammerDB directory:

```
cd ~/HammerDB-5.0
```

3. Run the HammerDB CLI:

```
./hammerdbcli
```

4. Set the appropriate DB build options:

```
dbset db mssqls
diset connection mssqls_server [IP Address or FQDN]
diset connection mssqls_tcp true
diset connection mssqls_authentication sql
diset connection mssqls_pass [Password]
diset tpcc mssqls_count_ware 800
diset tpcc mssqls_num_vu 16
diset tpcc mssqls_dbase tpcc
diset tpcc mssqls_use_bcp false
diset tpcc mssqls_raiseerror true
loadscript
```

5. Build the schema:

```
buildschema
```

After the schema build finished, we created a backup in SQL Server Management studio with the backup_db.sql script in the Scripts we used for testing section. After taking a backup of the database, we cloned out the SQL VMs and client VMs to each server before running the tests.

Running the test

Before running the tests, we cloned out one additional client VM to serve as a controller VM from which we ran our test scripts. Note: For the scripts in our Scripts we used for testing section to work properly, you must set up passwordless SSH on all servers and VMs in the testbed. We also edited the for loop logic in the run.sh and restore.sh scripts to match the number of VMs we ran for each test. We recommend making two sets of copies of the run.sh and restore.sh scripts: one set for the 6 VMs of the 512 GB of memory configuration and the other set for the 12 VMs of each of the other configurations. Then copy them to run.sh and restore.sh respectively before each run. We completed the following steps to execute the run script:

1. Log into the controller VM, and navigate to the scripts directory.
2. Run the test:

```
nohup ./run.sh 2>&1 &
```

3. After the test finishes and the restores complete, reboot the VMs.
4. Complete steps 1 through 3 two more times for a total of three runs, and report the median run.

Scripts we used for testing

backup_db.sql

```
BACKUP DATABASE [tpcc] TO DISK = N'Z:\backup\tpcc.bak' WITH NOFORMAT, NOINIT, NAME = N'tpcc-Full Database Backup', SKIP, NOREWIND, NOUNLOAD, COMPRESSION, STATS = 10
GO
```

drop-and-restore.sql

```
EXEC msdb.dbo.sp_delete_database_backuphistory @database_name = N'tpcc'
GO
use [master];
GO
USE [master]
GO
ALTER DATABASE [tpcc] SET SINGLE_USER WITH ROLLBACK IMMEDIATE
GO

GO
USE [master]
GO
DROP DATABASE [tpcc]
GO

USE [master]
RESTORE DATABASE [tpcc] FROM DISK = N'Z:\backup\tpcc.bak' WITH FILE = 1, NOUNLOAD, STATS = 5
GO
```

restore.ps1

```
Invoke-Sqlcmd -ServerInstance localhost -Database master -QueryTimeout 0 -InputFile "C:\pt-sql-hdb\drop-and-restore.sql"
```

restore.sh

```
#!/bin/bash

for i in {01..06};
do
    ssh Administrator@gnr-mssql-${i} "C:\pt-sql-hdb\restore.ps1" &
done
```

tpcc.tcl

```
#!/bin/tclsh

dbset db mssqls
diset connection mssqls_server [IP Address or FQDN]
diset connection mssqls_tcp true
diset connection mssqls_authentication sql
diset connection mssqls_pass [Password]
diset tpcc mssqls_count_ware 800
diset tpcc mssqls_num_vu 64
diset tpcc mssqls_dbase tpcc
```

```

diset tpcc mssqls_rampup 15
diset tpcc mssqls_duration 20
diset tpcc mssqls_raiseerror true
loadscript

vuset vu 64
vuset logtotemp 1
vuset showoutput 1
vuset unique 0

tcset refreshrate 1
tcset logtotemp 1
tcset unique 0
tcset timestamps 1

tcstart
vucreate
vurun
vudestroy
after 2500
tcstop

```

run_hdb.sh

```

#!/bin/bash

cd /home/pt/HammerDB-5.0/
./hammerdbcli auto tpcc.tcl

```

run.sh

```

#!/bin/bash

TIMESTAMP=$(date '+%Y%m%d_%H%M%S')
HDB_SCRIPT=tpcc.tcl
RESULTS_DIR=/home/pt/results
HDB_DIR=/home/pt/HammerDB-5.0
SCRIPT_DIR=/home/pt/scripts
HDB_RUN=${HDB_DIR}/${HDB_SCRIPT}
HDB_LOG=/tmp/hammerdb.log

#HammerDB run parameters
WAREHOUSE_COUNT=800
RAMPUP=15 # minutes
DURATION=20 # minutes
VIRTUAL_USERS=64
TOTAL_ITERATIONS=1000000000 #Run length
WARMUP=$((RAMPUP*60))
RUNTIME=$((DURATION*60))
RUN_LENGTH=$((WARMUP+RUNTIME)+60))

for i in {01..06};
do
    ssh gnr-hdb-${i} "sed -i 's/mssqls_server.*/mssqls_server 192.168.10.2${i}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/mssqls_linux_server.*/mssqls_linux_server 192.168.10.2${i}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/_total_iterations.*/_total_iterations ${TOTAL_ITERATIONS}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/_count_ware.*/_count_ware ${WAREHOUSE_COUNT}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/_allwarehouse.*/_allwarehouse false/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/_rampup.*/_rampup ${RAMPUP}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/_duration.*/_duration ${DURATION}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/_num_vu.*/_num_vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/vuset vu .*/vuset vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
    ssh gnr-hdb-${i} "sed -i 's/vuset logtotemp .*/vuset logtotemp 1/' ${HDB_RUN}"
done

```

```

#Make results folder for run and copy hammerdb config files
mkdir -p ${RESULTS_DIR}/${TIMESTAMP}/client_configs
RUN_DIR=${RESULTS_DIR}/${TIMESTAMP}
CLIENT_CFG_DIR=${RESULTS_DIR}/${TIMESTAMP}/client_configs
mkdir -p ${CLIENT_CFG_DIR}/hammerdb
cp ${HDB_RUN} ${CLIENT_CFG_DIR}/hammerdb/gnr-hdb-${i}.run.tcl

for i in {01..06};
do
    scp gnr-hdb-${i}:${HDB_RUN} ${CLIENT_CFG_DIR}/hammerdb/gnr-hdb-${i}.tpcc.tcl
done

#Start esxtop on SUT nodes
ssh root@gnr-sut-01 "rm -f /vmfs/volumes/6a0f7c2c-366bfd61-f4a8-d404e6efc5b0/esxtop/gnr-
sut-01-esxtop.csv"
ssh root@gnr-sut-02 "rm -f /vmfs/volumes/6a0f7c9e-8ac16e5d-9beb-d404e6efad80/esxtop/gnr-
sut-02-esxtop.csv"
ssh root@gnr-sut-03 "rm -f /vmfs/volumes/6a0f7ca4-e7b37d95-cacc-d404e6ede810/esxtop/gnr-
sut-03-esxtop.csv"
ssh root@gnr-sut-04 "rm -f /vmfs/volumes/6a0f7caa-7d0e63db-5dd7-6805cab727f0/esxtop/gnr-
sut-04-esxtop.csv"

ssh root@gnr-sut-01 "esxtop -a -b -d 5 > /vmfs/volumes/6a0f7c2c-366bfd61-f4a8-d404e6efc5b0/esxtop/gnr-
sut-01-esxtop.csv" &
ssh root@gnr-sut-02 "esxtop -a -b -d 5 > /vmfs/volumes/6a0f7c9e-8ac16e5d-9beb-d404e6efad80/esxtop/gnr-
sut-02-esxtop.csv" &
ssh root@gnr-sut-03 "esxtop -a -b -d 5 > /vmfs/volumes/6a0f7ca4-e7b37d95-cacc-d404e6ede810/esxtop/gnr-
sut-03-esxtop.csv" &
ssh root@gnr-sut-04 "esxtop -a -b -d 5 > /vmfs/volumes/6a0f7caa-7d0e63db-5dd7-6805cab727f0/esxtop/gnr-
sut-04-esxtop.csv" &

#Start perfmon on MSSQL VMs
for i in {01..06};
do
    ssh Administrator@gnr-mssql-${i} "rm -Force C:\PerfLogs\Admin\hammerdb\DataCollector01.blg"
    ssh Administrator@gnr-mssql-${i} "rm -Force C:\PerfLogs\Admin\hammerdb\perfmon.html"
done

for i in {01..06};
do
    ssh Administrator@gnr-mssql-${i} logman start hammerdb
done

#Run Test
echo -e "\nRunning test for $((RAMPUP+DURATION)) minutes!"

for i in {01..06};
do
    ssh gnr-hdb-${i} "rm -f ${HDB_LOG}"
done

sleep 5

for i in {01..06};
do
    ssh gnr-hdb-${i} ${SCRIPT_DIR}/run_hdb.sh &
done

sleep ${RUN_LENGTH}

#Stop esxtop on SUT nodes and perfmon on MSSQL VMs
mkdir -p ${RUN_DIR}/esxtop
mkdir -p ${RUN_DIR}/perfmon
ESXTOP_DIR=${RUN_DIR}/esxtop
PERFMON_DIR=${RUN_DIR}/perfmon

for i in {01..04};
do
    ssh root@gnr-sut-${i} pkill esxtop
done

```

```

for i in {01..06};
do
    ssh Administrator@gnr-mssql-${i} logman stop hammerdb
done

#Copy esxtop files
scp root@gnr-sut-01:/vmfs/volumes/6a0f7c2c-366bfd61-f4a8-d404e6efc5b0/esxtop/gnr-sut-01-esxtop.
csv ${ESXTOP_DIR}
scp root@gnr-sut-02:/vmfs/volumes/6a0f7c9e-8ac16e5d-9beb-d404e6efad80/esxtop/gnr-sut-02-esxtop.
csv ${ESXTOP_DIR}
scp root@gnr-sut-03:/vmfs/volumes/6a0f7ca4-e7b37d95-cacc-d404e6ede810/esxtop/gnr-sut-03-esxtop.
csv ${ESXTOP_DIR}
scp root@gnr-sut-04:/vmfs/volumes/6a0f7caa-7d0e63db-5dd7-6805cab727f0/esxtop/gnr-sut-04-esxtop.
csv ${ESXTOP_DIR}

#Create html reports from the perfmon files and copy both
for i in {01..06};
do
    ssh Administrator@gnr-mssql-${i} "tracertpt.exe C:\PerfLogs\Admin\hammerdb\DataCollector01.blg
-report C:\PerfLogs\Admin\hammerdb\perfmon.html -f html"
done

for i in {01..06};
do
    scp Administrator@gnr-mssql-${i}:C:/PerfLogs/Admin/hammerdb/DataCollector01.blg ${PERFMON_DIR}/
gnr-mssql-${i}-perfmon.blg
    scp Administrator@gnr-mssql-${i}:C:/PerfLogs/Admin/hammerdb/perfmon.html ${PERFMON_DIR}/gnr-mssql-
${i}-perfmon.html
done

#Copy HammerDB results
for i in {01..06};
do
    scp gnr-hdb-${i}:${HDB_LOG} ${RUN_DIR}/gnr-hdb-${i}-hammerdb.log
done

sleep 60

#Restore database after the run finishes
./restore.sh

#Parse results and output run summary
for i in {01..06};
do
    cat ${RUN_DIR}/gnr-hdb-${i}-hammerdb.log | grep TPM | awk '{ print $10 }' >> ${RUN_DIR}/TPM.txt
    cat ${RUN_DIR}/gnr-hdb-${i}-hammerdb.log | grep NOPM | awk '{ print $7 }' >> ${RUN_DIR}/NOPM.txt
done

touch ${RUN_DIR}/run_summary.txt
SUMMARY=${RUN_DIR}/run_summary.txt

echo "HammerDB run $(date)" >> ${SUMMARY}
echo "" >> ${SUMMARY}
echo "" >> ${SUMMARY}

echo "Number of virtual users: ${VIRTUAL_USERS}" >> ${SUMMARY}
echo "Rampup (min): ${RAMPUP}" >> ${SUMMARY}
echo "Duration (min): ${DURATION}" >> ${SUMMARY}
echo "" >> ${SUMMARY}

TPM_TOTAL=0
NOPM_TOTAL=0

for i in $(cat ${RUN_DIR}/TPM.txt);
do
    TPM_TOTAL=$(( ${TPM_TOTAL} + ${i} ))
done

for i in $(cat ${RUN_DIR}/NOPM.txt);
do

```

```

        NOPM_TOTAL=$(( ${NOPM_TOTAL}+${i} ))
done

for i in {01..06};
do
    echo "gnr-hdb-${i} TPM: $(cat ${RUN_DIR}/gnr-hdb-${i}-hammerdb.log | grep TPM | awk '{ print $10 }') " >> ${SUMMARY}
done

echo "" >> ${SUMMARY}
echo "TPM total: ${TPM_TOTAL}" >> ${SUMMARY}
echo "" >> ${SUMMARY}

for i in {01..06};
do
    echo "gnr-hdb-${i} NOPM: $(cat ${RUN_DIR}/gnr-hdb-${i}-hammerdb.log | grep NOPM | awk '{ print $7 }') " >> ${SUMMARY}
done

echo "" >> ${SUMMARY}
echo "NOPM total: ${NOPM_TOTAL}" >> ${SUMMARY}

cp ./nohup.out ${RUN_DIR}
rm -f ./nohup.out

```

This project was commissioned by Intel.

[Read the report ►](#)

Primary contributors

-  **Tech:** Aaron W.
-  **Writing:** Nathan P.
-  **Design:** Cassandra O.
-  **PM:** Trent W.

How we created this report

A PT team, which includes the contributors we've listed and others, created this report and performed the technical work behind it. We used AI for drafting and editing text.



Facts matter.®

Principled Technologies is a registered trademark of Principled Technologies, Inc. All other product names are the trademarks of their respective owners. For additional information, review the science behind this report.

DISCLAIMER OF WARRANTIES; LIMITATION OF LIABILITY:

Principled Technologies, Inc. has made reasonable efforts to ensure the accuracy and validity of its testing, however, Principled Technologies, Inc. specifically disclaims any warranty, expressed or implied, relating to the test results and analysis, their accuracy, completeness or quality, including any implied warranty of fitness for any particular purpose. All persons or entities relying on the results of any testing do so at their own risk, and agree that Principled Technologies, Inc., its employees and its subcontractors shall have no liability whatsoever from any claim of loss or damage on account of any alleged error or defect in any testing procedure or result.

In no event shall Principled Technologies, Inc. be liable for indirect, special, incidental, or consequential damages in connection with its testing, even if advised of the possibility of such damages. In no event shall Principled Technologies, Inc.'s liability, including for direct damages, exceed the amounts paid in connection with Principled Technologies, Inc.'s testing. Customer's sole and exclusive remedies are as set forth herein.