PROWESS

Behind the Report—Testing Addendum: Dell™ PowerEdge™ R750 Servers with NVMe® RAID Unleash Microsoft® SQL Server® Database Performance

This document provides the system-configuration details and step-by-step procedures that Prowess Consulting used to collect performance data for running a Microsoft® SQL Server® database on a Dell™ PowerEdge™ R750 server.

For the full analysis, read the report, "Dell™ PowerEdge™ R750 Servers with NVMe® RAID Unleash Microsoft® SQL Server® Database Performance."

Testing was concluded in November 2021.

Dell™ PowerEdge™ R750 Server	
Processor	
Processor	2 x Intel® Xeon® Gold 6338 processor
Processor base frequency	2.0 GHz
Cores/threads	32/64
Total cores	64
Memory	
Total memory in system	256 GB
Number and type of memory modules	16 x 16 GB Hynix® DDR4 DIMMs
Memory speed	3,200 megatransfers per second (MT/s)
RAID array 1 (Microsoft® SQL Server® 2019 database data volume)	
Controller	Dell™ PowerEdge RAID Controller 11 (PERC 11) H755N Front
Disks	8 x 3.2 TB Dell Enterprise NVM Express® (NVMe®) CM6 Multiuse (Kioxia® CM6-V)
RAID array 2 (Microsoft® SQL Server® 2019 database log volume)	
Controller	Dell™ PowerEdge RAID Controller 11 (PERC 11) H755N Front
Disks	8 x 3.2 TB Dell Enterprise NVM Express® (NVMe®) CM6 Multiuse (Kioxia® CM6-V)
RAID array 1 (boot volume)	
Controller	Dell™ Boot Optimized Server Storage (BOSS)-S2
Disks	2 x 480 GB Micron® 5300 PRO solid-state drive (SSD)

Networking	
Network adapters	1 Gb, dual-port Broadcom® BCM5720 NetXtreme® Ethernet controller 10/25 Gb, dual-port BCM57414 NetXtreme-E Remote Direct Memory Access (RDMA) Ethernet controller
Server details	
BIOS version	1.3.8
Operating system (OS) name and version	Red Hat® Enterprise Linux® 8.4
OS kernel	4.18.0-305.el8.x86_64
Database software	
SQL Server	SQL Server 2019 (RTM-CU14) (KB50077182)—15.0.4188.2 (x64)

Note: The Dell PowerEdge R750 server testing was performed on bare-metal servers with no hypervisor/virtualization layer.

Testing Procedures

The following sections describe the step-by-step procedures used to configure the systems, verify the configurations, and run the benchmark tests.

RAID Configuration

Deploy a Dell PowerEdge R750 server with sixteen 3.2 TB Dell Enterprise NVMe® CM6 Multiuse PCIe® Gen4 SSD drives connected equally to two Dell™ PERC 11 H755N Front raid controllers with the following configuration:

SQL_Data virtual disk:

- RAID configuration: RAID 5
- Number of disks: 8
- Virtual disk properties:
  - Strip size = 256 KB
  - Number of blocks = 43750260736
  - Span depth = 1
  - Number of drives per span = 8
  - Write cache (initial setting) = WriteThrough
  - Disk cache policy = Disabled
  - Encryption = None
  - Data protection = None
  - Active operations = None

- Exposed to OS = Yes
- Emulation type = Default
- Cachebypass size = Cachebypass-64k
- Cachebypass mode = Cachebypass Intelligent

SQL_Log virtual disk:

- RAID configuration: RAID 10
- Number of disks: 8
- Virtual disk properties:
  - Strip size = 256 KB
  - Number of blocks = 25000148992
  - Span depth = Not applicable (N/A)
  - Number of drives per span = N/A
  - Write cache (initial setting) = WriteThrough
  - Disk cache policy = Disabled
  - Encryption = None
  - Data protection = None
  - Active operations = None
  - Exposed to OS = Yes
  - Emulation type = Default
  - Cachebypass size = Cachebypass-64k
  - Cachebypass mode = Cachebypass Intelligent
  - Is LD ready for OS requests = Yes

Red Hat® Enterprise Linux® Installation

Install Red Hat® Enterprise Linux® 8.4 and SQL Server 2019:

1. Download the latest Red Hat Enterprise Linux 8.4 installation media.
   Note: If you do not already have a Red Hat Enterprise Linux subscription, you can get one by creating a developer account with Red Hat.
2. Create a boot USB drive from which to install.
3. Install Red Hat Enterprise Linux:
   a. Select **Time & Date**, and then set the time zone to **Central**.
   b. Set the **Root Password**.

c. Select **Installation Source**.

d. Select **Server**.

e. Select **Installation Destination**.

f. Select the **BOSS** drives as the installation location.

g. Select **Subscription**, and then fill in the details.

h. Select **Install**.

4. Download and install the Linux PERCCLI utility.

5. Enter the following commands to ensure that the correct device drive is being configured for the correct volume name:

```
./perccli64 show all

./perccli64 /c0 show
```

6. Locate the virtual disk (VD) list.

7. Note the **VD ID** for each drive name.

8. On the Dell PowerEdge R750 server, set the write cache using the following commands:

```
./perccli64 /c0/v239 set wrcache=wt

./perccli64 /c1/v239 set wrcache=wt
```

9. For each VD, enter:

```
./perccli64 /c0 /v<VD ID> show all
```

10. Copy the **SCSI NAA ID** value.

11. Enter the following command to get the corresponding worldwide name (WWN) values for each drive:

```
ls -la /dev/disk/by-id/ | grep <SCSI NAA ID>
```

12. Locate the device ID.

13. Format the drives with the following file system:

- SQL_Data:

```
mkfs.xfs /dev/<device id> -f -L datavolume
```

- SQL_Log:

```
mkfs.xfs /dev/<device id> -f -L logvolume
```

14. Run the following commands to optimize the disks:

```
echo none > /sys/block/$1/queue/scheduler

echo 0 > /sys/block/$1/queue/add _ random

echo 2 > /sys/block/$1/queue/nomerges
```

```
echo 2048 > /sys/block/$1/queue/nr _ requests
echo 0 > /sys/block/$1/queue/rotational
echo 2 > /sys/block/$1/queue/rq _ affinity
echo 1024 > /sys/block/$1/queue/max _ sectors _ kb
echo 1024 > /sys/block/$1/device/queue _ depth
echo 1 > /sys/block/$1/queue/iostats
```

SQL Server 2019 Installation and Configuration

1. Run the following commands to install prerequisites:

```
sudo yum install python2 compat-openssl10
sudo alternatives --config python
```

2. Run the following commands to enable TuneD and install the SQL Server TuneD profile:

```
systemctl enable tuned
dnf install tuned-profiles-mssql
```

3. The tuned.conf file in /usr/lib/tuned/mssql has the following configuration:

```
#
# tuned configuration
#

[main]
summary=Optimize for MS SQL Server
include=throughput-performance

[vm]
transparent _ hugepage.defrag=always

[sysctl]
vm.max _ map _ count=800000
kernel.numa _ balancing=0
kernel.sched _ latency _ ns=60000000
kernel.sched _ min _ granularity _ ns=15000000
kernel.sched _ wakeup _ granularity _ ns=2000000
```

4. Run the following command to set the SQL Server repository:

```
sudo curl -o /etc/yum.repos.d/mssql-server.repo
https://packages.microsoft.com/config/rhel/8/mssql-server-2019.repo
```

5. Run the following command to install SQL Server:

```
sudo dnf install -y mssql-server
```

6. Run the following command to configure SQL Server:

```
/opt/mssql/bin/mssql-conf setup
```

   a. Select **1** for evaluation.

   b. Enter **Yes** to accept the license terms.

   c. Enter a **SQL Server Admin Password**.

7. Run the following command to verify that SQL Server is running:

```
systemctl status mssql-server
```

8. For testing purposes only, stop and disable the firewall service with the following commands:

```
systemctl stop firewalld
systemctl disable firewalld
```

9. For testing purposes only, disable SELINUX with the following comments:

```
vi /etc/selinux/config
SELINUX=disabled
```

10. Install the SQL Server tools with the following commands:

```
curl -o
/etc/yum.repos.d/msprod.repo https://packages.microsoft.com/config/rhel/8/prod.repo
dnf install -y mssql-tools unixODBC-devel
```

11. Configure .bash_profile and .bashrc to source the tools and SQL Server install paths using the following commands:

```
echo 'export PATH="$PATH:/opt/mssql-tools/bin"' >> ~/.bash _ profile
echo 'export PATH="$PATH:/opt/mssql-tools/bin"' >> ~/.bashrc
echo 'export PATH="$PATH:/opt/mssql/bin"' >> ~/.bash _ profile
echo 'export PATH="$PATH:/opt/mssql/bin"' >> ~/.bashrc
source ~/.bashrc
```

12. Test connectivity to SQL Server by running the following command:

```
sqlcmd -s localhost -U SA -P <sa password>
```

13. Enter the following SQL command to verify the version:

```
Select @@version

go
```

14. Enable trace flag 3979 to support SQL Server and the Forced Unit Access (FUA) input/output (I/O) subsystem using the following commands:

    a. Enable **traceflag 3979**:

```
mssql-conf traceflag 3979 on
```

    b. Set **control.writethrough** in the **mssql-conf configuration** option to **1**:

```
mssql-conf set control.writethrough 1
```

    c. Set **control.alternatewritethrough** in the **mssql-conf configuration** option to **0**:

```
mssql-conf set control.alternatewritethrough 0
```

15. Run the following commands to complete setup of SQL Server:

```
mssql-conf set telemetry.customerfeedback false

sysctl -w kernel.numa _ balancing=0

sysctl -w vm.max _ map _ count=262144

mssql-conf set network.tlsprotocols 1.2
```

16. Set SQL Server memory to 90 percent of available memory with the following command:

```
mssql-conf set memory.memorylimitmb 230400
```

17. Run the following command to create the SQL Server directory:

```
mkdir -p mssql/data mssql/log/log mssql/log/tempdb
```

18. Run the following command to change ownership of the newly created directories:

```
sudo chown mssql:mssql /mssql/data

sudo chown mssql:mssql /mssql/log
```

19. Run the following command to enable execution on the directories:

```
sudo chmod 777 /mssql/data

sudo chmod 777 /mssql/log
```

20. Run the following command to get the universally unique identifier (UUID) of each mounted disk:

```
blkid
```

    Note the UUID for **datavolume** and **logvolume**.

21. Update **/etc/fstab** with the following command:

```
UUID="<UUID datavolume>" /mssql/data xfs rw,attr2,noatime 0 0

UUID="<UUID datavolume>" /mssql/log xfs rw,attr2,noatime 0 0
```

22. Update the SQL Server configuration data and log file locations with the following commands:

```
mssql-conf set filelocation.defaultdatadir /mssql/data/

mssql-conf set filelocation.defaultlogdir /mssql/log/log
```

23. Restart the SQL Server service with the following command:

```
systemctl restart mssql-server.service
```

24. Launch the SQL Server management console from a client system.

25. Connect to the Linux SQL Server instance.

26. Run the following commands to modify the location of TempDB:

```
ALTER DATABASE tempdb MODIFY FILE

      (NAME = tempdev, FILENAME = '/mssql/log/tempdb/tempdb01.mdf', SIZE = 1024,
      FILEGROWTH = 8192MB)

GO

ALTER DATABASE tempdb MODIFY FILE

      (NAME = templog, FILENAME = '/mssql/tempdb/templog.ldf', SIZE = 1024, FILEGROWTH
      = 8192MB)

GO

ALTER DATABASE tempdb REMOVE FILE tempdev2

GO

ALTER DATABASE tempdb REMOVE FILE tempdev3

GO

ALTER DATABASE tempdb REMOVE FILE tempdev4

GO

ALTER DATABASE tempdb REMOVE FILE tempdev5

GO

ALTER DATABASE tempdb REMOVE FILE tempdev6

GO

ALTER DATABASE tempdb REMOVE FILE tempdev7

GO

ALTER DATABASE tempdb REMOVE FILE tempdev8

GO
```

```
ALTER DATABASE tempdb

ADD FILE (NAME = tempdev2, FILENAME = '/mssql/tempdb/tempdb02.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)

ALTER DATABASE tempdb

ADD FILE (NAME = tempdev3, FILENAME = '/mssql/tempdb/tempdb03.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)

ALTER DATABASE tempdb

ADD FILE (NAME = tempdev4, FILENAME = '/mssql/tempdb/tempdb04.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)

ALTER DATABASE tempdb

ADD FILE (NAME = tempdev5, FILENAME = '/mssql/tempdb/tempdb05.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)

ALTER DATABASE tempdb

ADD FILE (NAME = tempdev6, FILENAME = '/mssql/tempdb/tempdb06.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)

ALTER DATABASE tempdb

ADD FILE (NAME = tempdev7, FILENAME = '/mssql/tempdb/tempdb07.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)

ALTER DATABASE tempdb

ADD FILE (NAME = tempdev8, FILENAME = '/mssql/tempdb/tempdb08.ndf', SIZE = 1024,
FILEGROWTH = 8192MB)
```

27. Download and install Atop to capture system performance information using the following commands:

```
wget https://www.atoptool.nl/download/atop-2.6.0-1.el8.x86_64.rpm

chmod +x atop-2.6.0-1.el8.x86_64.rpm

rpm -ivh atop-2.6.0-1.el8.x86_64.rpm

service atop start
```

28. Capture Atop data using the following command:

```
atop -r -b <beginning time> -e <ending time> > /tmp/pass#.txt
```

29. Capture DSTAT data using the following command:

```
dstat -trdlD total,sda,sdb,sdc 60 --output /tmp/pass#.csv
```

Benchmark Testing New Orders per Minute (NOPM)[1]

After software is installed and pretesting has verified proper configurations, run the benchmark tests to compare the performance between the systems. Run the following workload three times, taking the median of the scores.

Configure the Windows® 10/Windows Server® 2019 Client Test Machine

1. Configure an Open Database Connectivity (ODBC) data source for the Linux SQL Server:
   a. Launch Microsoft® ODBC Data Source Administrator (32-bit).
   b. Click **Add**.
   c. Select **SQL Server Native Client 11.0**, and then click **Finish**.
   d. In the **Create a New Data Source to SQL Server** window, in the **Name** field, enter a unique name to identify the Linux SQL Server.
   e. In the **Server** field, enter a name or IP address to connect to the Linux SQL Server, and then click **Next**.
   f. In the **How should SQL Server verify the authenticity of the login ID** field, select **With SQL Server authentication**.
   g. In the **Login ID** field, enter **SA**.
   h. In the **Password** field, enter the SA password.
   i. Click **Next**.
   j. Click **Next** again.
   k. Click **Finish**.
   l. Click **Test Data Source** to verify connectivity.
2. Download and install the following SQL Server client tools:
   a. Client Tools Connectivity
   b. Client Tools Backwards Compatibility
   c. Client Tools SDK
3. Copy the provided **MSTPCC.470.tar** file to the testing client.
4. Unzip the .tar file to **C:\MSTPCC.470**.
5. In File Explorer, browse to **C:\ MSTPCC.470\SETUP\scripts\1400.war\database**.
6. Open **createdb.sql**, and then update the **FILENAME** path for each file.
7. In File Explorer, browse to **C:\MSTPCC.470\BC-TPCC-v1.6.0**.
8. Double-click **BenchCraft-TPCC.msi** to install Benchcraft®, accepting the default settings.[2]
9. Open a command prompt or PowerShell session.
10. Run the following command to build the test database:

```
.\setup.cmd /s:<IP or Hostname> /U:sa /P:<password> /W:1400 /B:full /D:normal
/v:true
```

11. Once the database build is complete, take a backup of the database.
12. Launch the Benchcraft application:
    a. Click **New Profile**.
    b. Click **Browse**, and then select the **TPCC_SQLServer.bdll** plugin file.
    c. Click **Next**.
    d. Select **ThreadGroup-1**, and then enter the following values for each setting:
        - **Thread Count: 100**
        - **W_ID Range End: 1400**
        - **W_ID Maximum: 1400**
        - **Server Name or IP:** IP or name of the Linux SQL Server
        - **User Name: SA**
        - **Password:** *SA password*
13. Save the profile.
14. Select the **Drivers** page.
15. Select **Launch**.
16. Wait 5 minutes before pressing **F1** to run the load.
17. Run the load for 45 minutes or until the load is at a steady state, and then take the last 15 minutes of the load as the scored value.
18. Once the test is complete, delete and restore the database.
19. Double-click the saved profile to launch the Benchcraft workload utility to run the workload again.

[1] Servers were configured and tested serially, which resulted in updates/patches being applied on different days to achieve the same patch version.
[2] Note that this workload is derived from the TPC-C® benchmark and is not comparable to published TPC-C benchmark results, as this implementation does not comply with all requirements of the TPC-C benchmark.

PROWESS