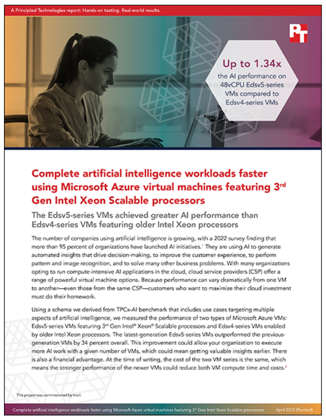

A Principled Technologies report: Hands-on testing. Real-world results.

Up to 1.34x the AI performance on 48vCPU Edsv5-series VMs compared to Edsv4-series VMs

Complete artificial intelligence workloads faster using Microsoft Azure virtual machines featuring 3rd Gen Intel Xeon Scalable processors

The Edsv5-series VMs achieved greater AI performance than Edsv4-series VMs featuring older Intel Xeon processors

The number of companies using artificial intelligence is growing, with a 2022 survey finding that more than 95 percent of organizations have launched AI initiatives.[1] They are using AI to generate automated insights that drive decision-making, to improve the customer experience, to perform pattern and image recognition, and to solve many other business problems. With many organizations opting to run compute-intensive AI applications in the cloud, cloud service providers (CSP) offer a range of powerful virtual machine options. Because performance can vary dramatically from one VM to another—even those from the same CSP—customers who want to maximize their cloud investment must do their homework.

Using a schema we derived from TPCx-AI benchmark that includes use cases targeting multiple aspects of artificial intelligence, we measured the performance of two types of Microsoft Azure VMs: Edsv5-series VMs featuring 3rd Gen Intel® Xeon® Scalable processors and Edsv4-series VMs enabled by older Intel Xeon processors. The latest-generation Edsv5-series VMs outperformed the previous-generation VMs by 34 percent overall. This improvement could allow your organization to execute more AI work with a given number of VMs, which could mean getting valuable insights earlier. There is also a financial advantage. At the time of writing, the cost of the two VM series is the same, which means the stronger performance of the newer VMs could reduce both VM compute time and costs.[2]

This project was commissioned by Intel.

Complete artificial intelligence workloads faster using Microsoft Azure virtual machines featuring 3rd Gen Intel Xeon Scalable processors April 2023 (Revised)

The science behind the report:

Complete artificial intelligence workloads faster using Microsoft Azure virtual machines featuring 3rd Gen Intel Xeon Scalable processors

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 Complete artificial intelligence workloads faster using Microsoft Azure virtual machines featuring 3rd Gen Intel Xeon Scalable processors.

We concluded our hands-on testing on July 20, 2022. 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 July 14, 2022 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 http://facts.pt/calculating-and-highlighting-wins. Unless we state otherwise, we have followed the rules and principles we outline in that document.

Table 1: Results of our testing

VM type	Overall score on TPCx-AI-like workload	Edsv5-series advantage
Standard_E48ds_v5	214.126	N/A
Standard_E48ds_v4	159.204	34.5%

This project was commissioned by Intel.

System configuration information

Table 2: Detailed information on the system we tested.

System configuration information	16vCPU Edsv5-series VM (One head node)	48vCPU Edsv5-series VMs (Five worker nodes)
Tested by	Principled Technologies	Principled Technologies
Test date	2022/7/20	2022/7/20
CSP / Region	Microsoft Azure – East US	Microsoft Azure – East US
WL specific parameters	100GB dataset, two concurrent users	100GB dataset, two concurrent users
Iterations and result choice	3 runs, median	3 runs, median
Server platform	Standard_D16_v4	Standard_D48ds_v4
BIOS name and version	American Megatrends Inc. 090008 (12/07/2018)	American Megatrends Inc. 090008 (12/07/2018)
Operating system name and version/build number	Centos 8.4 4.18.0-305.7.1.el8_4.x86_64	Centos 8.4 4.18.0-305.7.1.el8_4.x86_64
Date of last OS updates/patches applied	2022/7/20	2022/7/20
Processor		
Number of processors	1	2
Vendor and model	Intel® Xeon® Platinum 8370C CPU @ 2.80GHz	Intel Xeon Platinum 8370C CPU @ 2.80GHz
Core count (per processor)	32	32
Core frequency (GHz)	2.8	2.8
Family, Model, Stepping	6,106,6	6,106,6
Hyper-threading	Yes	Yes
Turbo	Yes (3.5 GHz)	Yes (3.5 GHz)
Number of vCPU per VM	16	48
Memory module(s)		
Total memory in system	128GB	384GB
NVMe memory present?	No	No
Total memory (DDR+NVMe RAM)	128GB	384GB
General HW		
Storage: NW or Direct Att / Instance	Direct Att / Instance	Direct Att / Instance

System configuration information	16vCPU Edsv5-series VM (One head node)	48vCPU Edsv5-series VMs (Five worker nodes)
Local storage		
OS		
Number of drives	1	1
Drive size (GB)	256	256
Drive information (speed, interface, type)	StandardSSD_LRS	StandardSSD_LRS
Data drives		
Number of drives	4	4
Drive size (GB)	160	160
Drive information (speed, interface, type)	StandardSSD_LRS	StandardSSD_LRS
Network adapter		
Vendor and model	Azure virtual network adapter	Azure virtual network adapter
Number and type of ports	1x 50Gb	1x 50Gb

Table 3: Detailed information on the system we tested.

System configuration information	16vCPU Edsv4-series VM (One head node)	48vCPU Edsv4-series VMs (Five worker nodes)
Tested by	Principled Technologies	Principled Technologies
Test date	2022/6/21	2022/6/21
CSP / Region	Microsoft Azure – East US	Microsoft Azure – East US
WL specific parameters	100GB dataset, two concurrent users	100GB dataset, two concurrent users
Iterations and result choice	3 runs, median	3 runs, median
Server platform	Standard_D16_v4	Standard_D48ds_v4
BIOS name and version	American Megatrends Inc. 090008 (12/07/2018)	American Megatrends Inc. 090008 (12/07/2018)
Operating system name and version/build number	Centos 8.4 4.18.0-305.7.1.el8_4.x86_64	Centos 8.4 4.18.0-305.7.1.el8_4.x86_64
Date of last OS updates/patches applied	2022/6/20	2022/6/20
Processor		
Number of processors	1	1
Vendor and model	Intel Xeon Platinum 8272CL CPU @ 2.60GHz	Intel Xeon Platinum 8272CL CPU @ 2.60GHz
Core count (per processor)	26	26
Core frequency (GHz)	2.6	2.6
Family, Model, Stepping	6,85,7	6,85,7
Hyper-threading	Yes	Yes
Turbo	Yes (3.7 GHz)	Yes (3.7 GHz)
Number of vCPU per VM	16	48

System configuration information	16vCPU Edsv4-series VM (One head node)	48vCPU Edsv4-series VMs (Five worker nodes)
Memory module(s)		
Total memory in system	128GB	384GB
NVMe memory present?	No	No
Total memory (DDR+NVMe RAM)	128GB	384GB
General HW		
Storage: NW or Direct Att / Instance	Direct Att / Instance	Direct Att / Instance
Local storage		
OS		
Number of drives	1	1
Drive size (GB)	256	256
Drive information (speed, interface, type)	StandardSSD_LRS	StandardSSD_LRS
Data drives		
Number of drives	4	4
Drive size (GB)	160	160
Drive information (speed, interface, type)	StandardSSD_LRS	StandardSSD_LRS
Network adapter		
Vendor and model	Azure virtual network adapter	Azure virtual network adapter
Number and type of ports	1x 50Gb	1x 50Gb

How we tested

This experiment consisted of running benchmark programs derived from TPCx-AI-on Cloudera Data Platform hosted on Azure virtual machine instances with Intel processors. We provisioned Azure VMs, formatted and configured storage, installed CDP with trial license, installed prerequisite software, and installed the benchmark. We tuned system configuration to maximize performance on use cases 2, 5, 8, and 9 at 100G scale factor. We then completed three tests and identified the test with the median primary metric (100GB dataset).

Azure instance types

Table 4: The Microsoft Azure instance types we used in testing. Source: Principled Technologies.

Instance**	Processor	HT	VCPUs	Memory (GiB)	OS drive* (GiB)	Data drive*(GiB)
Standard_E16_v4	Intel Xeon Platinum 8272CL CPU @ 2.60GHz	Yes	16	64	256	4x160
Standard_E48ds_v4	Intel Xeon Platinum 8272CL CPU @ 2.60GHz	Yes	48	64	256	4x160
Standard_E16_v5	Intel Xeon Platinum 8370C CPU @ 2.80GHz	Yes	16	64	256	4x160
Standard_E48ds_v5	Intel Xeon Platinum 8370C CPU @ 2.80GHz	Yes	48	64	256	4x160

*: All drive volumes were StandardSSD_LRS volumes.
**: All instances used Centos 8.4 image from OpenLogic, available on the Azure marketplace.

Software versions

Table 5: The software versions we used in testing. Source: Principled Technologies.

Software	Version
Schema derived from TPCx-AI	1.0.2
CDP	7.4.4
Hadoop	3.1.1
Spark	2.4.5
Scala	2.11
GCC, G++	7.3.0
openmpi-mpicc	4.0
Tensorflow	2.4.0
numpy	1.19.2
horovod	0.21
petastorm	0.9.8
tensorflow-addons	0.14
Python	3.7
py-opencv	4.5
pyarrow	3.0
librosa	0.8
Hypy	2.10

Provisioning Azure instances

1. Log into the Azure Portal, and navigate to the Virtual Machines.
2. Click Create, and click Virtual Machine.
3. Enter the subscription and resource group names, and give the virtual machine a name.
4. Choose the region for the instance. We used East US.
5. For Availability Options, select No infrastructure redundancy required.
6. For Security type, select Standard.
7. For the image, click See all images to navigate to the marketplace.
8. Search for OpenLogic.
9. Select CentOS 8 from OpenLogic by Perforce, generation 1.
10. Select Size according to Table 1.
11. For Authentication type, select SSH public key.
12. Set username to centos.
13. If no prior keypair exists, select Generate new keypair for SSH public key source, and enter a name.
14. If a prior key exists, select Use an existing key stored in Azure for SSH public key source, and select the key from the drop-down menu.
15. Select Allow selected ports, and ensure SSH is enabled.
16. Click Next: Disks.
17. For OS disk type, select Standard SSD.
18. Create four data disks using Standard SSD type and custom size of 160 GiB.
19. Click Review + create.
20. Click Create.
21. When prompted, download the SSH key to a safe place.
22. Resize the OS boot drive from 30 to 256 GiB by following instructions at https://learn.microsoft.com/en-us/azure/virtual-machines/windows/expand-os-disk.

Allowing ingress traffic to port 7180 (Cloudera Manager) on Azure VMs

1. Log into the Azure portal.
2. Click Virtual Machines.
3. Locate the VM in the list, and click it.
4. In the settings panel, click Networking.
5. Under Inbound port rules, click Add inbound port rule and use the following settings:
    - Source: *Your public ip address*
    - Source port ranges: *
    - Destination: Any
    - Service: Custom
    - Destination port ranges: 7180
    - Protocol: TCP
    - Action: Allow
    - Priority: *Add 1 to the highest priority id less than 65000 of all existing rules*
6. Click Add.

Creating an unprivileged user with sudo ability

1. Connect to the host with SSH.
2. Create the user:

```
sudo adduser benchmark
```

3. Add user to the wheel group (allows user to use sudo):

```
sudo usermod -aG wheel benchmark
```

Enabling passwordless sudo

1. Connect to the host with SSH.
2. Edit the sudoers:

```
sudo visudo
```

3. Add the following line to the end of the file:

```
benchmark ALL=(ALL) NOPASSWD:ALL
```

Disabling firewall

1. Connect to the host with SSH.
2. Stop firewall:

```
sudo systemctl stop firewalld
```

3. Disable firewall on startup:

```
sudo systemctl disable firewalld
```

Disabling selinux

1. Connect to the host with SSH.
2. Disable selinux runtime:

```
sudo setenforce 0
```

3. Set selinux to permissive mode:

```
sudo sed -i 's/^SELINUX=.*$/SELINUX=permissive' /etc/selinux/config
```

4. Set selinux type to targeted:

```
sudo sed -i 's/^SELINUXTYPE=.*$/SELINUXTYPE=targeted' /etc/selinux/config
```

5. Reboot:

```
sudo shutdown -r now
```

Configuring CentOS repositories

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Edit the AppStream repository definition:

```
nano /etc/yum.repos.d/CentOS-Linux-AppStream.repo
```

4. Disable the mirrorlist by adding a `#` to the start of the mirrorlist line.
5. Change the repository's baseurl (uncomment if necessary) to the following:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/AppStream/$basearch/os/
```

6. Save the file by typing `[CTRL]+O`.
7. Exit the editor by typing `[CTRL]+X`.
8. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-BaseOS.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/BaseOS/$basearch/os/
```

9. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-ContinuousRelease.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/cr/$basearch/os/
```

10. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-Devel.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/Devel/$basearch/os/
```

11. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-Extras.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/extras/$basearch/os/
```

12. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-FastTrack.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/fasttrack/$basearch/os/
```

13. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-HighAvailability.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/HighAvailability/$basearch/os/
```

14. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-Plus.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/centosplus/$basearch/os/
```

15. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-PowerTools.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/PowerTools/$basearch/os/
```

16. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-Sources.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/cern/centos/8.5.2111/AppStream/$basearch/os/
```

17. Repeat steps 3 through 7 for `/etc/yum.repos.d/CentOS-Linux-Debuginfo.repo` repository:

```
baseurl=https://linuxsoft.cern.ch/centos-debuginfo/8/$basearch
```

Configuring SSH between nodes for unprivileged user

1. Connect to the host with SSH.
2. Create the SSH directory:

```
mkdir ~/.ssh
chmod 700 ~/.ssh
```

3. Edit the SSH configuration file:

```
nano ~/.ssh/config
```

4. Add the lines to disable host key checking for all hosts:

```
Host *
    StrictHostKeyChecking no
```

5. Save by pressing `[CTRL]+O`.
6. Exit by pressing `[CTRL]+X`.
7. Generate the SSH keypair:

```
ssh-keygen -t rsa -b 4096 -N"" -f ~/.ssh/id_rsa
```

8. Copy the SSH key to all other hosts (repeat once per host, replacing **host** with the host):

```
ssh-copy-id benchmark@host
```

Configuring SSH between nodes for root user

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Create the SSH directory:

```
mkdir ~/.ssh
chmod 700 ~/.ssh
```

4. Edit the SSH configuration file:

```
nano ~/.ssh/config
```

5. Add the lines to disable host key checking for all hosts:

```
Host *
    StrictHostKeyChecking no
```

6. Save by pressing `[CTRL]+O`.
7. Exit by pressing `[CTRL]+X`.
8. Generate the SSH keypair:

```
ssh-keygen -t rsa -b 4096 -N"" -f ~/.ssh/id_rsa
```

9. Copy the SSH key to all other hosts (repeat once per host, replacing **host** with the host):

```
ssh-copy-id benchmark@host
```

Configuring root SSH authorized keys

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Copy authorized keys from unprivileged user to root:

```
cp /home/benchmark/.ssh/authorized_keys /root/.ssh/authorized_keys
```

4. Set authorized keys file ownership:

```
chown root:root ~/.ssh/authorized_keys
```

5. Set authorized keys file permissions:

```
chmod 600 ~/.ssh/authorized_keys
```

Disabling transparent huge pages

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Disable transparent huge pages:

```
echo never > /sys/kernel/mm/transparent_hugepage/defrag
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

4. Open /etc/rc.local:

```
nano /etc/rc.local
```

5. Add lines to disable transparent huge pages on next boot:

```
echo never > /sys/kernel/mm/transparent_hugepage/defrag
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

6. Save by pressing `[CTRL]+O`.
7. Exit by pressing `[CTRL]+X`.

Adding Enterprise Linux repositories

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Install packages:

```
dnf install -y epel-release
```

Installing system packages

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Install packages:

```
dnf install -y nmon python3 vim tar wget java-1.8.0-openjdk   \
cmake maven git blas64 lapack64 bc curl htop nano bind-utils  \
expect libglvnd-glx libglvnd-devel libSM libSM-devel          \
libXcursor libXcursor-devel libXrandr libXrandr-devel         \
libXt libXt-devel libXtst libXtst-devel openmpi openmpi-devel \
pssh zip unzip "@Development Tools"
```

4. Create pssh symlinks:

```
ln -s /usr/bin/parallel-scp /usr/local/bin/pscp.pssh
ln -s /usr/bin/parallel-ssh /usr/local/bin/pssh
```

Installing Miniconda

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Create installer directory:

```
mkdir -p /opt/installers
```

4. Change to the installers directory:

```
cd /opt/installers
```

5. Download the installer:

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh
```

6. Set installer permissions:

```
chmod +x Miniconda3-latest-Linux-x86_64.sh
```

7. Run the installer:

```
/opt/installers/Miniconda3-latest-Linux-x86_64.sh -b -p '/opt/miniconda'
```

8. Activate the base conda environment:

```
. '/opt/miniconda/bin/activate'
```

9. Initialize the conda environment:

```
conda init
```

10. Install mamba in the base environment:

```
conda install -y -n base -c conda-forge "mamba>=0.21"
```

Formatting data drives

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Identify the data disk Linx device names:

```
lsblk
```

4. Format each device:

Note: Be sure to replace **DEVICE_IDENTIFIER** *with the device identifier you learned in step 2.*

```
    mkfs -t xf3 DEVICE_IDENTIFIER
```

5. Mount each device:

Note: Be sure to replace **DEVICE_IDENTIFIER** *with the device identifier you learned in step 2.*

Note: Be sure to replace **N** *with an integer to assign an arbitrary number to the disk.*

```
mount -t xfs DEVICE_IDENTIFIER /var/data/hadoop/drive-N
```

6. Determine each filesystem UUID:

```
blkid
```

7. Add the data disks to /etc/fstab:

```
vi /etc/fstab
```

8. Add a line for each drive, being sure to replace **THE_FS_UUID** with the filesystem UUID determined in step 6 and **N** with the assigned drive number in step 5. Note: whitespaces are tab characters.

```
UUID=THE_FS_UUID /var/data/hadoop/drive-N xfs defaults 0 0
```

9. Reboot the virtual machine:

```
reboot
```

10. Reconnect to the virtual machine via SSH.
11. Confirm the data disks are mounted in the correct locations:

```
mount | grep /var/data/hadoop
```

12. Create HDFS initial folders on each disk:

```
mkdir -p /var/data/hadoop/drive-N/hdfs
mkdir -p /var/data/hadoop/drive-N/hdfs/nn
mkdir -p /var/data/hadoop/drive-N/hdfs/snn
mkdir -p /var/data/hadoop/drive-N/hdfs/dn
mkdir -p /var/data/hadoop/drive-N/hdfs/tmp
```

Setting swappiness

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Edit the sysctl configuration file:

```
sudo nano /etc/sysctl.conf
```

4. Add a line to set vm.swappiness:

```
vm.swappiness=10
```

5. Save by pressing `[CTRL]+O`.
6. Exit by pressing `[CTRL]+X`.

Setting hostname on the head node

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Set the host name:

```
hostnamectl set-hostname head.cdp.local
```

Setting hostname on all worker nodes

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Set the host name (Replace **N** with the worker node's number):

```
hostnamectl set-hostname workerN.cdp.local
```

Adding cluster VMs to /etc/hosts on all hosts

1. Connect to the host with SSH.
2. For each head node and worker node virtual machine, determine the internal IP address from the cloud provider.
3. Become root:

```
sudo su
```

4. Open /etc/hosts in the nano editor:

```
nano /etc/hosts
```

5. For each VM, add the following line, replacing **IP_ADDRESS and HOSTNAME** with the values you determined in step 1.

```
IP_ADDRESS HOSTNAME
```

6. Save the file by typing `[CTRL]+O`.
7. Close the file editor by typing `[CTRL]+X`.

Creating conda environment for the benchmark workload python

1. Contact info@principledtechnologies.com to request the conda environment file for the benchmark python.
2. Save the file from step 1 as /opt/benchmark/dist/tools/spark/conda-env-workload.yml.
3. Connect to the host with SSH.
4. Become root:

```
sudo su
```

5. Create env directory:

```
mkdir -p /opt/benchmark/envs/workload
```

6. Activate the base conda environment:

```
. '/opt/miniconda/bin/activate'
```

7. Initialize the conda environment:

```
conda init
```

8. Create the workload environment:

```
conda create -y -p /opt/benchmark/envs/workload
```

9. Install mamba package manager:

```
conda install -y -p /opt/benchmark/envs/workload -c conda-forge "mamba>=0.21"
```

10. Activate the workload environment:

```
conda activate /opt/benchmark/envs/workload
```

11. Install workload dependencies:

```
mamba env update -p /opt/benchmark/envs/workload --file /opt/benchmark/dist/tools/spark/conda-
env-workload.yml
```

12. Change workload environment file ownership:

```
chown -R benchmark:benchmark /opt/benchmark/envs/workload
```

13. Change workload environment folder mode:

```
chmod 700 /opt/benchmark/envs/workload
```

Creating conda environment for the benchmark driver python

1. Contact info@principledtechnologies.com to request the conda environment file for the benchmark driver python.
2. Save the file from step 1 as /opt/benchmark/dist/tools/spark/conda-env-driver.yml.
3. Connect to the host with SSH.
4. Become root:

```
sudo su
```

5. Create env directory:

```
mkdir -p /opt/benchmark/dist/lib/python-venv
```

6. Activate the base conda environment:

```
. '/opt/miniconda/bin/activate'
```

7. Initialize the conda environment:

```
conda init
```

8. Create the driver environment:

```
conda create -y -p /opt/benchmark/dist/lib/python-venv
```

9. Install mamba package manager:

```
conda install -y -p /opt/benchmark/dist/lib/python-venv -c conda-forge "mamba>=0.21"
```

10. Activate the driver environment:

```
conda activate /opt/benchmark/dist/lib/python-venv
```

11. Install driver dependencies:

```
mamba env update -p /opt/benchmark/dist/lib/python-venv --file /opt/benchmark/dist/tools/spark/
conda-env-driver.yml
```

12. Change driver environment file ownership:

```
chown -R benchmark:benchmark /opt/benchmark/dist/lib/python-venv
```

13. Change driver environment folder mode:

```
chmod 700 /opt/benchmark/dist/lib/python-venv
```

Configuring the benchmark

1. Contact info@principledtechnologies.com to request a spark configuration file for the benchmark for your system.
2. Save the file from step 1 as /opt/benchmark/dist/config-spark.yaml.
3. Contact info@principledtechnologies.com to request an environment script file for the benchmark for your system.
4. Save the file from step 3 as /opt/benchmark/dist/setenv.sh.

Creating HDFS directories for the benchmark

1. Connect to the host with SSH.
2. Become root:

```
sudo su
```

3. Add Hadoop to the PATH environment variable:

```
export PATH="/opt/yarn/hadoop-3.1.1/bin:$PATH"
```

4. Set the HDFS binary:

```
export HDFS="$(which hdfs)"
```

5. Create HDFS directory:

```
sudo -u hdfs "$HDFS" dfs -mkdir -p /user/benchmark
```

6. Set HDFS directory ownership:

```
sudo -u hdfs "$HDFS" dfs -chown -R root:supergroup /user/benchmark
```

7. Set HDFS directory access permissions:

```
sudo -u hdfs "$HDFS" dfs -chmod -R 777 /user/benchmark
```

8. Verify the directory exists and has correct ownership and permissions:

```
sudo -u hdfs "$HDFS" dfs -ls /user/ &&
```

Running the benchmark

1. Connect to the head node with SSH.
2. Become root:

```
sudo su
```

3. Change to the benchmark directory:

```
cd /opt/benchmark/dist
```

4. Set the benchmark environment:

```
. ./setenv.sh
```

5. Set the Hadoop user name:

```
export HADOOP_USER_NAME=benchmark
```

6. Run the benchmark script:

```
/opt/benchmark/dist/Benchmarkrun.sh
```

7. Collect data captured in /opt/benchmark/dist/logs.

Read the report at http://facts.pt/dYLb2KL ▶

This project was commissioned by Intel.