The science behind the report:

Get higher performance for your MySQL databases with Dell APEX Private Cloud

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 Get higher performance for your MySQL databases with Dell APEX Private Cloud.

We concluded our hands-on testing on April 18, 2023. 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 April 18, 2023 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. We ran each test three times and report the median result.

	Dell APEX and PowerMax solution	Amazon EC2 with EBS solution
Number of new orders per minute	5,215,469	4,157,909
Average number of new orders per minute	325,967	259,869

System configuration information

Table 2: Detailed information on the systems we tested.

System configuration information	Dell VxRail® E660F
BIOS name and version	Dell™ 1.3.8
Non-default BIOS settings	Performance system profile
Operating system name and version/build number	VMware ESXi®, 7.0.3, 19193900
Date of last OS updates/patches applied	N/A
Power management policy	Performance
Processor	
Number of processors	2
Vendor and model	Intel® Xeon® Platinum 8358
Core count (per processor)	32
Core frequency (GHz)	2.60
Stepping	1
Memory module(s)	
Total memory in system (GB)	512 (256 assigned)
Number of memory modules	16
Vendor and model	Hynix® HMAA4GR7CJR8N-XN
Size (GB)	32
Type	DDR4
Speed (MHz)	3,200
Speed running in the server (MHz)	3,200
Storage controller	
Vendor and model	Dell HBA355i Front
Cache size (GB)	0
Firmware version	17.15.08.00
Vendor and model	Dell BOSS-S2
Cache size	0
Firmware version	2.5.13.4008
Local storage (type A)	
Number of drives	2
Drive vendor and model	Micron® MTFDDAV480TDS
Drive size (GB)	480
Drive information (speed, interface, type)	6Gbps, SATA, SSD

System configuration information	Dell VxRail® E660F
Network adapter	
Vendor and model	Broadcom® BCM57414 Dual 25Gb Ethernet
Number and type of ports	2 x 25GbE
Driver version	21.85.21.91
Vendor and model	Emulex® LPe35002-M2-D 2-Port 32Gb Fibre Channel
Number and type of ports	2 x 32GbE
Driver version	03.05.23
Vendor and model	Broadcom BCM5720 Gigabit Ethernet
Number and type of ports	2x 1GbE
Driver version	21.81.3
Cooling fans	
Vendor and model	N/A
Number of cooling fans	16
Power supplies	
Vendor and model	Dell PWR SPLY,1400W,RDNT,LTON
Number of power supplies	2
Wattage of each (W)	1,400

Table 3: Detailed information on the Dell EMC PowerMax 2500.

System configuration information	Dell EMC® PowerMax 2500
Software version	10.0.0.0 (Release 01, Build 6079_125/0001)
Number of storage shelves	1
Total number of drives	20
Drive size (TB)	3.8

Table 4: Detailed information on the Amazon EC2 with EBS solution.

System configuration information	Amazon EC2 with EBS solution
Date testing ended	4/12/2023
Cloud service provider (CSP)	AWS®
Region	us-east-1b
Workload information	
Workload name and version	HammerDB v4.6 TPROC-C
Workload or software specific parameters	500 warehouses, 64 vusers
Iterations and result choice	3 test runs, median reported

System configuration information	Amazon EC2 with EBS solution
Cloud VM or instance details	
Number of VMs	16
VM or instance size	m6i.4xlarge
BIOS name	Amazon EC2 1.0
vCPU	16
Number of cores/threads	16
Memory (GB)	64
Underlying processor model	Intel Xeon Platinum 8375C
Operating system information	
Image or template name and UUID	RHEL 9 64-bit (x86) ami-016eb5d644c333ccb
Operating system name	Red Hat Enterprise Linux 9
Kernel version	5.14.0-162.18.1.el9_1.x86_64
Date patches last applied	4/12/2023
Changes made from CSP image	No
Instance storage (volume type 1)	
Number of volumes	1
Volume use in this test	Data drive
CSP volume type	EBS io2
Volume size (GB)	200GB
IOPS requested	16000
Throughput requested (MB/s)	N/A
Encryption type	Not encrypted
Instance storage (volume type 2)	
Number of volumes	1
Volume use in this test	OS
CSP volume type	EBS gp3
Volume size (GB)	50
IOPS requested	Default
Throughput requested (MB/s)	Default
Encryption type	Not encrypted

How we tested

We deployed 16 VMs on a cluster of four Dell APEX Private Cloud servers with Intel Xeon Platinum 8358 processors that Dell configured with VMware® vSphere®. The PowerMax volumes backed the VMware vSphere datastores that held the VMs running MySQL Community Server, as well as the VMs with the workload utility HammerDB. On AWS, we tested m6i instances using Intel Xeon Platinum 8375C processors with gp3 storage for OS volumes and io2 storage for data volumes. We chose this configuration because it best matched the Dell APEX Private Cloud offering we compared it to.

Each VM on both the AWS and Dell APEX solutions had 16 vCPUs, 64 GB of memory, a 50GB OS drive, and a 200GB data drive. We deployed 16 VMs on the Dell APEX Private Cloud environment. Each of the Dell APEX Private Cloud nodes had 64 cores and 512 GB of memory, and we allocated 256 GB of memory to the VMs to match the configuration of the core-to-RAM configuration of commercially available compute-optimized Dell APEX Private Cloud nodes, which gave us room for four VMs per node. We confirmed this resource utilization during discovery testing, where the 16-VM configuration showed the Dell APEX Private Cloud cluster CPU utilization exceeding 90 percent. We then matched the configuration on AWS. On each VM, we installed MySQL Community Server, version 8.0.32.

To test the database performance of each environment, we used the HammerDB benchmark suite with the TPROC-C workload. Each VM had a single MySQL instance with a 500-warehouse TPROC-C database. We targeted the maximum NOPM each environment could achieve by increasing the user count until performance degraded. The maximum user count was 64 users per VM for both Dell APEX Private Cloud and our AWS VMs.

We deployed four physical servers for HammerDB clients to drive the workload on Dell APEX Private Cloud. Each of these four servers ran four client VMs, for a total of 16 client VMs. We configured each client VM with eight vCPUs, 32 GB of memory, and 50 GB of OS storage. We used m6i.2xlarge instances on AWS for clients, with specs matching those of the Dell APEX Private Cloud clients.

We also created a controller VM on each environment so we could easily configure changes and run scripts across the various test and client VMs.

For more information about our testing and detailed results, see the science behind the report.

Installing VMware vCenter Server Appliance version 7.0.3 build 20990077

Note: We deployed the vCenter appliance on one of the client hosts for management of the vSphere environment.

1. From the VMware support portal at https://my.vmware.com, download VMware vCenter version 7.0.3 build 20990077.
2. Mount the image on your local system, and browse to the vcsa-ui-installer folder. Expand the folder for your OS, and launch the installer if it doesn't automatically begin.
3. When the vCenter Server Installer wizard opens, click Install.
4. To begin installation of the new vCenter server appliance, click Next.
5. Check the box to accept the license agreement, and click Next.
6. Enter the IP address of the infrastructure server with VMware ESXi version 7.0.3 build 19193900. Provide the root password, and click Next.
7. To accept the SHA1 thumbprint of the server's certificate, click Yes.
8. Accept the VM name, and provide and confirm the root password for the VCSA. Click Next.
9. Set the size for environment you're planning to deploy. We selected Medium. Click Next.
10. Select the datastore on which to install vCenter. Accept the datastore defaults, and click Next.
11. Enter the FQDN, IP address information, and DNS servers you want to use for the vCenter server appliance. Click Next.
12. To begin deployment, click Finish.
13. When Stage 1 has completed, click Close. To confirm, click Yes.
14. Open a browser window and connect to https://[vcenter.FQDN:5480/.
15. On the Getting Started - vCenter Server page, click Set up.
16. Enter the root password, and click Log in.
17. Click Next.
18. Enable SSH access, and click Next.
19. To confirm the changes, click OK.
20. Type vsphere.local for the Single Sign-On domain name. Enter a password for the administrator account, confirm it, and click Next.
21. Click Next.
22. Click Finish.

Installing vSphere version 7.0.3 build 19193900 on the Dell APEX Private Cloud servers

1. From https://my.vmware.com/group/vmware/evalcenter?p=vsphere-eval-7#tab_download, download ESXi Version 7.0.3 build 19193900.
2. Open a new browser tab, and connect to the IP address of the Dell server iDRAC.
3. Log in with the iDRAC credentials.
4. In the main screen, click Launch Virtual Console.
5. In the console menu bar, select Virtual Media.
6. In the Virtual Media popup window, from the dropdown menu, select ISO, and click Open Image to browse local computers and select the image you downloaded in step 1.
7. On the console menu bar, click the Power Control, and select Power Reset.
8. The system will boot to the mounted image and the Loading ESXi installer screen will appear. When prompted, press Enter to continue.
9. To Accept the EULA and Continue, press F11.
10. Select the storage device to target for installation. To continue, press Enter.
11. To confirm the storage target, press Enter.
12. Select the keyboard layout, and press Enter.
13. Provide a root password, and confirm the password. To continue, press Enter.
14. To install, press F11.
15. Upon completion, reboot the server by pressing Enter.
16. Complete steps 1 through 15 for each server under test.

Creating a vSphere cluster in vCenter

1. Open a browser, and enter the address of the vCenter server you deployed. For example: https://[vcenter.FQDN]/ui
2. In the left panel, select the vCenter server, right-click, and select New Datacenter.
3. Provide a name for the new data center, and click OK.
4. Select the data center you just created, right-click, and select New Cluster.
5. Give a name to the cluster, and click OK.
6. In the cluster configuration panel, under Add hosts, click Add.
7. Check the box for Use the same credentials for all hosts. Enter the IP address and root credentials for the first host, and enter the IP addresses of all remaining hosts. Click Next.
8. To select all hosts, check the box beside Hostname/IP Address. Click OK.
9. Click Next.
10. Click Finish.

Creating a distributed vSwitch and port group

1. From vSphere client, click HomeàNetworking.
2. Select your Datastore.
3. On the right panel, in the Actions drop-down menu, select Distributed vSwitchàNew Distributed vSwitch.
4. Give your vSwitch a name, or accept the default. Click Next.
5. Select 7.0.2 - ESXi 7.0.2 and later as the version, and click Next.
6. Select the number of uplinks per ESXi host you'll give to the vSwitch (we selected 2). Click Next.
7. Click Finish.
8. Select new DSwitch, and click configure.
9. Select Properties, and click edit.
10. Under advanced, set MTU (Bytes) to 9000.
11. Click OK.
12. Select LACP, click + NEW, and set mode to active.
13. Click OK.
14. Enter appropriate name, set ports to two, and click OK.
15. Right-click the new DSwitch, and select Add and Manage Hosts.
16. Leave Add hosts selected, and click Next.
17. To add new hosts, click the +.
18. To select all the hosts in your target cluster, check the box for Host. Click OK. Click Next.
19. Select the NIC you want to use for this DSwitch, and click Assign Uplink.
20. Select the first LAG port.
21. Check the box for Apply this uplink assignment to the rest of the hosts. Click OK.

Creating the gold test, client, and controller VMs (APEX)

1. In vCenter, select a host to create a VM.
2. Right click and select New Virtual Machine.
3. Click Create a new virtual machine, and click next.
4. Enter an appropriate VM name, select Datacenter, and click next.
5. Select an appropriate host, and click next.
6. Choose an appropriate datastore, and click next.
7. Select an appropriate compatibility mode (ESXi 7.0 U2 and later), and click next.
8. Select Linux for the Guest OS Family and Red Hat Enterprise Linux 9 (64-bit) for the Guest OS Version. Click next.
9. In customize hardware, select the following:
   - 16 vCPU
   - 64GB memory
   - 50GB OS hard drive
   - 200GB data hard drive
   - Choose the appropriate VM Network
   - Point the dvd drive to the installation media ISO in your datastore
   - Click Next
10. Review, and click finish.
11. Repeat steps 1-10 twice to create a client VM and a controller VM with the following specs:
    - 8 vCPU
    - 16GB memory
    - 50GB OS hard drive

Creating the gold test, client, and controller VMs (AWS)

1. Log into AWS, and navigate to the AWS Management Console.
2. Click EC2.
3. Click Launch instance, and to open the Launch Instance in the dropdown wizard, click Launch instance.
4. In the search window, type Windows Server, and press enter.
5. On Quick Start, next to Red Hat Enterprise Linux 9, click the Select button.
6. On Choose Instance Type, select m6i.4xlarge, and click "Next: Configure Instance Details."
7. On Configure Instance, set the following:
   a. Number of instances: 1
   b. Purchasing option: Leave unchecked
   c. Network: Default VPC
   d. Subnet: Choose the region you are working in. We used east-1b.
   e. Auto-assign Public IP: Enable
   f. Placement Group: Leave unchecked
   g. Capacity Reservation: Open
   h. Domain join directory: No Directory
   i. IAM role: None
   j. Shutdown behavior: Stop
   k. Click Next: Add Storage
8. On Add Storage, set the following:
   a. Size: 50GB
   b. Volume Type: gp2
   c. Delete on Termination: Checked
   d. Encryption: Not Encrypted
   e. Add an additional drive:
      i. Size: 200GB
      ii. Volume Type: io2
      iii. IOPs: 16000

f. Delete on Termination: Checked
g. Encryption: Not Encrypted
h. Click Next: Add Tags

9. On Add Tags, add any tags you wish to use. Click Next: Configure Security Group.
10. On Configure Security Group, leave defaults, and click Review and Launch.
11. On Review, click Launch.
12. Choose the appropriate option for the key pair, and click Launch Instances.
13. Repeat these steps (minus the data volume creation) to deploy a client instance and a controller instance, changing the instance type in step 6 to m6i.2xlarge.

Installing Red Hat Enterprise Linux 9 on the gold test, client and controller VMs

1. Power on the gold test VM.
2. Boot to the installation media attached to the VM.
3. Select Install Red Hat Enterprise Linux 9.1.
4. Select English, and click Continue.
5. For the Installation destination, select automatic partitioning, and click Done.
6. For the Software Selection, select Minimal Install, and click Done.
7. Configure a static IP address, and click Done.
8. Set a password for the root user account, and click Done.
9. Click Begin Installation.
10. Repeat steps 1-9 for the gold client VM and controller VM.

Configuring Red Hat Enterprise Linux 9 and install MySQL on gold test VM

1. Log into the MySQL instance via ssh.
2. In /etc/selinux/config, set SELINUX=Permissive.
3. Stop and disable the firewall:

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

4. Set the tuned profile to virtual-guest:

```
tuned-adm profile virtual-guest
```

5. Add the following lines to the end of /etc/sysctl.conf:

```
vm.swappiness = 1
fs.aio-max-nr = 1048576
vm.nr_hugepages = 27648
```

6. Install prerequisite tools:

```
yum -y install wget vim tar zip unzip lz4 pigz nmon sysstat numactl ksh screen
```

7. Create a directory for MySQL:

```
mkdir -p /mnt/mysqldata
```

8. Create an XFS filesystem on the data drive:

```
mkfs.xfs /dev/sdb
```

9. Mount the data drive to the MySQL directory:

```
mount -o defaults,nofail,x-systemd.device-timeout=5 /dev/sdb /mnt/mysqldata
```

10. Edit the /etc/fstab file, and add the following line to the end:

```
/dev/sdb /mnt/mysqldata xfs defaults,nofail,x-systemd.device-timeout=5 0 2
```

11. Download the appropriate MySQL bundle for either x86 or Arm architecture on Red Hat Enterprise Linux from https://dev.mysql.com/downloads/mysql/.
12. Extract the MySQL bundle:

```
tar -xf mysql-community-server-8.0.32-1.el9.rpm-bundle.tar
```

13. Install MySQL Community Server 8 and all dependencies:

```
sudo yum --disablerepo=* localinstall mysql-community-server-8.0.32-1.el9.x86_64.rpm
```

14. Stop the MySQL service:

```
sudo service mysqld stop
```

15. Copy the mysql data directory to your data disk:

```
sudo cp -R -p /var/lib/mysql /mnt/mysqldata/
```

16. Start the MySQL service:

```
sudo service mysqld start
```

17. Log into the MySQL instance as the root user:

```
mysql -u root -p
```

18. Create a new user named mysql with full permissions:

```
CREATE USER 'mysql'@'localhost' IDENTIFIED BY '[password]';
GRANT ALL PRIVILEGES ON *.* TO 'mysql'@'localhost'
->  WITH GRANT OPTION;
CREATE USER 'mysql'@'%' IDENTIFIED BY '[password]';
GRANT ALL PRIVILEGES ON *.* TO 'mysql'@'%'
->  WITH GRANT OPTION;
```

19. Enable mysql_native_password authentication on the mysql user:

```
ALTER USER 'mysql'@'localhost' IDENTIFIED WITH mysql_native_password BY '[password]';
ALTER USER 'mysql'@'%' IDENTIFIED WITH mysql_native_password BY '[password]';
```

20. Edit the /etc/my.cnf file to match what we have in the Scripts we used for testing section.
21. Reboot the gold test VM:

```
reboot
```

Configuring RHEL 9 and installing HammerDB 4.6 on the gold client VM

1. SSH into the gold client VM.
2. Download HammerDB 4.6:

```
wget https://github.com/TPC-Council/HammerDB/releases/download/v4.6/HammerDB-4.6-Linux.tar.gz
```

3. Unzip HammerDB:

```
tar -xzf HammerDB-4.6-Linux.tar.gz
```

4. Install prerequisites:

```
dnf install -y epel-release
dnf install -y wget vim tar zip unzip lz4 pigz nmon sysstat numactl ksh screen
```

5. Download the appropriate MySQL bundle from https://dev.mysql.com/downloads/mysql/.
6. Extract the MySQL bundle:

```
tar -xf mysql-community-server-8.0.32-1.el9.rpm-bundle.tar
```

7. Install MySQL client:

```
sudo yum --disablerepo=* localinstall mysql-community-client-8.0.32-1.el9.x86_64.rpm
```

8. Place the run.tcl file from the Scripts we used for testing section into the HammerDB directory.
9. Create a scripts directory:

```
mkdir ~/scripts
```

10. Change directories into the scripts directory:

```
cd ~/scripts
```

11. Create a bash script called run_hammerdb.sh with the contents from the Scripts we used for testing section.

Setting up password-less SSH on the VMs and hosts

1. Open two different terminals, one for the gold test VM and one for the gold client VM.
2. In each VM, generate an RSA keypair:

```
ssh-keygen
```

3. Accept the default storage location, and leave the password blank.
4. Cat out the public key on each VM:

```
cat ~/.ssh/id_rsa.pub
```

5. Copy the output, and paste it into the ~/.ssh/authorized_keys file on the other VM.
6. Open one more terminal, and connect to the controller VM.
7. Repeat steps 2-5 on the controller VM, copying the public key into the authorized_keys file on each of the gold client and test VMs.

Building the 500-warehouse TPROC-C schema on the gold test VM

1. On the gold client VM, change directories into the HammerDB directory.
2. Enter the hammerdb cli:

```
./hammerdbcli
```

3. Setup the build parameters:

```
dbset db mysql
diset connection mysql_host [Hostname or IP Address]
diset tpcc mysql_count_ware 500
diset tpcc mysql_num_vu 16
diset tpcc mysql_pass [Password]
diset tpcc mysql_partition true
```

4. Build the schema:

```
buildschema
```

Backing up the 500-warehouse schema

1. SSH into the gold test VM.
2. Stop the MySQL service:

```
systemctl stop mysqld
```

3. Create a compressed backup of the mysql directory:

```
tar -cf- /mnt/mysqldata/mysql/ | pigz -9 -c > /mnt/mysqldata/backup/mysql_tpcc_500wh.tar.gz
```

4. Once the backup is complete, start the MySQL service:

```
systemctl start mysqld
```

5. Shut down the VM:

```
poweroff
```

Running the test

Prior to running the test, we cloned out the gold test VM and the gold client VM so that we had 16 test VMs, and 16 client VMs from which to run the test. We ran all of our tests from scripts in our scripts directory on our controller VM. Our run script sets up all the clients with the correct parameters, starts performance collection, runs the test, restores the database, and stops, collects, and parses the performance data.

1. SSH into the controller VM.
2. Change directories into the scripts directory:

```
cd ~/scripts
```

3. Run the test:

```
./run.sh [apex | aws]
```

4. After the test is finished, wait for the restore to complete before rebooting the VMs and running again.
5. We tested three times on each environment, and used the median for reporting purposes.

Scripts

run.sh

```
#!/bin/bash

CPU=${1}
TIMESTAMP=$(date '+%Y%m%d_%H%M%S')
HDB_SCRIPT=run.tcl
HDB_RUN=${HDB_DIR}/${HDB_SCRIPT}
HDB_LOG=/tmp/hammerdb.log

if [[ ${CPU} == "apex" ]];
then
    RESULTS_DIR=/root/results
    HDB_DIR=/root/HammerDB-4.6
    SCRIPT_DIR=/root/scripts/

elif [[ ${CPU} == "aws" ]];
then
    RESULTS_DIR=/home/ec2-user/results
    HDB_DIR=/home/ec2-user/HammerDB-4.6
    SCRIPT_DIR=/home/ec2-user/scripts/
else
    echo "CPU type not set. Enter either apex or aws after run.sh."
    exit
fi

#HammerDB run parameters
WAREHOUSE_COUNT=500
RAMPUP=5     #  minutes
DURATION=10  #  minutes
VIRTUAL_USERS=64
TOTAL_ITERATIONS=1000000000

#Run length
WARMUP=$((RAMPUP*60))
RUNTIME=$((DURATION*60))
RUN_LENGTH=$(((WARMUP+RUNTIME)+60))

#nmon parameters
STEP=2  # seconds
SAMPLES_TOTAL=$(((WARMUP+RUNTIME)/STEP+5))

#Prepare HammerDB run script
if [[ ${CPU} == "apex" ]];
then
    for i in {1..16};
    do
        ssh hammer-${i} "sed -i 's/_host.*/_host mysql-${i}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_total_iterations.*/_total_iterations ${TOTAL_
ITERATIONS}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_count_ware.*/_count_ware ${WAREHOUSE_COUNT}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_rampup.*/_rampup ${RAMPUP}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_duration.*/_duration ${DURATION}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_num_vu.*/_num_vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/vuset vu .*/vuset vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
    done

elif [[ ${CPU} == "aws" ]];
then
    for i in {1..16};
    do
        ssh hammer-${i} "sed -i 's/_host.*/_host mysql-${i}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_total_iterations.*/_total_iterations ${TOTAL_
ITERATIONS}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_count_ware.*/_count_ware ${WAREHOUSE_COUNT}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_rampup.*/_rampup ${RAMPUP}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_duration.*/_duration ${DURATION}/' ${HDB_RUN}"
        ssh hammer-${i} "sed -i 's/_num_vu.*/_num_vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
```

```
        ssh hammer-${i} "sed -i 's/vuset vu .*/vuset vu ${VIRTUAL_USERS}/' ${HDB_RUN}"
    done
else
    echo "CPU type not set. Enter either apex or aws after run.sh."
    exit
fi

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

for i in {1..16};
do
    scp hammer-${i}:${HDB_RUN} ${CLIENT_CFG_DIR}/hammerdb/hammer-${i}.run.tcl
done

#Start performance monitoring on hosts and clients
if [[ ${CPU} == "apex" ]];
then
    for i in {1..16};
    do
        ssh mysql-${i} "killall -q -w nmon; sync; rm -f /tmp/mysql-${i}.nmon"
    done

    for i in {1..4};
    do
        sshpass -p "APEX4us1234!" ssh root@esxi-${i} "rm -rf /tmp/esxtop_*" &
    done

    for i in {1..4};
    do
        sshpass -p "APEX4us1234!" ssh root@esxi-${i} "/tmp/startesxtop.sh" &
    done

    for i in {1..16};
    do
        ssh mysql-${i} "nmon -F /tmp/mysql-${i}.nmon -s${STEP} -c$((SAMPLES_TOTAL)) -J -t"
    done

elif [[ ${CPU} == "aws" ]];
then
    for i in {1..16};
    do
        ssh mysql-${i} "killall -q -w nmon; sync; rm -f /tmp/mysql-${i}.nmon"
    done

    for i in {1..16};
    do
        ssh mysql-${i} "nmon -F /tmp/mysql-${i}.nmon -s${STEP} -c$((SAMPLES_TOTAL)) -J -t"
    done
fi

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

for i in {1..16};
do
    ssh hammer-${i} "rm -f ${HDB_LOG}"
done

sleep 5

for i in {1..16};
do
    ssh hammer-${i} ${SCRIPT_DIR}/run_hammerdb.sh &
done

sleep ${RUN_LENGTH}
```

```
#Stop performance monitoring and copy files
mkdir -p ${RUN_DIR}/nmon
mkdir -p ${RUN_DIR}/esxtop

NMON_DIR=${RUN_DIR}/nmon
ESXTOP_DIR=${RUN_DIR}/esxtop

if [[ ${CPU} == "apex" ]];
then
    for i in {1..16};
    do
        ssh mysql-${i} "killall -q -w nmon; sync"
        done

    for i in {1..4};
        do
        sshpass -p "APEX4us1234!" ssh root@esxi-${i} "pkill esxtop" &
        done

    for i in {1..4};
        do
        sshpass -p "APEX4us1234!" scp root@esxi-${i}:/tmp/esxtop_* ${ESXTOP_DIR}
        done

    for i in {1..16};
        do
            scp mysql-${i}:/tmp/mysql-${i}.nmon ${NMON_DIR}
        done

elif [[ ${CPU} == "aws" ]];
then
        for i in {1..16};
        do
            ssh aws${i} "killall -q -w nmon; sync"
        done

    for i in {1..16};
        do
            scp aws${i}:/tmp/aws${i}.nmon ${NMON_DIR}
        done
fi

#Copy HammerDB results

for i in {1..16};
do
    scp hammer-${i}:${HDB_LOG} ${RUN_DIR}/hammer-${i}-hammerdb.log
done

sleep 60

#Restore database after the run finishes
if [[ ${CPU} == "apex" ]];
then
    for i in {1..16};
    do
         ssh mysql-${i} "screen -d -m /root/restoredb.sh"
    done
elif [[ ${CPU} == "aws" ]];
then
        #for i in {1..16};
        do
            ssh aws${i} "screen -d -m /home/ec2-user/restoredb.sh"

        done
fi

#Parse results and output run summary
for i in {1..16};
do
    cat ${RUN_DIR}/hammer-${i}-hammerdb.log | grep TPM | awk '{ print $10 }' >> ${RUN_DIR}/TPM.txt
        cat ${RUN_DIR}/hammer-${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 {1..16};
do
    echo "hammer-${i} TPM: $(cat ${RUN_DIR}/hammer-${i}-hammerdb.log | grep TPM | awk '{ print $10
}')" >> ${SUMMARY}
done

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

for i in {1..16};
do
    echo "hammer-${i} NOPM: $(cat ${RUN_DIR}/hammer-${i}-hammerdb.log | grep NOPM | awk '{ print $7
}')" >> ${SUMMARY}
done

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

if [[ ${CPU} == "apex" ]];
then
        for i in {1..16};
        do
            /usr/bin/nmonchart ${NMON_DIR}/mysql-${i}.nmon ${NMON_DIR}/mysql-${i}.html
        done
elif [[ ${CPU} == "aws" ]];
then
        for i in {1..16};
        do
            ~/nmonchart ${NMON_DIR}/aws${i}.nmon ${NMON_DIR}/aws${i}.html
    done
fi
```

run_hammerdb.sh

```
#!/bin/bash
cd ~/HammerDB-4.6
./hammerdbcli auto run.tcl
```

run.tcl

```
#!/bin/tclsh
puts "SETTING CONFIGURATION"
global complete
proc wait_to_complete {} {
global complete
set complete [vucomplete]
if {!$complete} { after 5000 wait_to_complete } else { exit }
}

dbset db mysql

diset connection mysql_host mysql-1
diset tpcc mysql_count_ware 500
diset tpcc mysql_user mysql
diset tpcc mysql_pass [password]
diset tpcc mysql_partititon true
diset tpcc mysql_driver timed
diset tpcc mysql_rampup 5
diset tpcc mysql_duration 10
diset tpcc mysql_num_vu 64

vuset logtotemp 1

loadscript

vuset vu 64
vuset showoutput 1
vucreate

vurun

wait_to_complete
vwait forever
```

restoredb.sh

```
#!/bin/bash

systemctl stop mysqld
rm -rf /mnt/mysqldata/mysql
pigz -d -c /mnt/mysqldata/backup/mysql_tpcc_500wh.tar.gz | tar -C /mnt/mysqldata/ -xf-
systemctl start mysqld
```

my.cnf

```
[mysqld]

datadir=/mnt/mysqldata/mysql
default_authentication_plugin=mysql_native_password
socket=/mnt/mysqldata/mysql/mysqld.sock
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
port=3306
bind_address=0.0.0.0
large-pages

# general

max_connections=4000
table_open_cache=8000
table_open_cache_instances=16
back_log=1500
default_password_lifetime=0
ssl=0
performance_schema=OFF
max_prepared_stmt_count=128000
skip_log_bin=1
character_set_server=latin1
collation_server=latin1_swedish_ci
transaction_isolation=REPEATABLE-READ

# files

innodb_file_per_table
innodb_log_file_size=1024M
innodb_log_files_in_group=16 #scale
innodb_open_files=4000

# buffers
innodb_buffer_pool_size=53248M #scale
innodb_buffer_pool_instances=64
innodb_log_buffer_size=1024M

# tune

innodb_doublewrite=0
innodb_thread_concurrency=0
innodb_flush_log_at_trx_commit=0
innodb_max_dirty_pages_pct=90
innodb_max_dirty_pages_pct_lwm=10
join_buffer_size=32K
sort_buffer_size=32K
innodb_use_native_aio=1
innodb_stats_persistent=1
innodb_spin_wait_delay=6
innodb_max_purge_lag_delay=300000
innodb_max_purge_lag=0
innodb_flush_method=O_DIRECT_NO_FSYNC
innodb_checksum_algorithm=none
innodb_io_capacity=8000
innodb_io_capacity_max=16000
innodb_lru_scan_depth=9000
innodb_change_buffering=none
innodb_read_only=0
innodb_page_cleaners=4
innodb_undo_log_truncate=off

# perf special

innodb_adaptive_flushing=1
innodb_flush_neighbors=0
innodb_read_io_threads=16
innodb_write_io_threads=16
innodb_purge_threads=4
```

```
innodb_adaptive_hash_index=0

# monitoring

innodb_monitor_enable='%'

[client]

socket=/mnt/mysqldata/mysql/mysqld.sock
```

Read the report at https://facts.pt/S8KfF8j ▶

This project was commissioned by Dell Technologies.

Principled Technologies®

Facts matter.®