Evaluating Microsoft Azure SQL Managed Instance Next-gen GP for SQL workloads: Performance and value analysis

Our tests demonstrated that upgrading from baseline classic GP Instances to Next-gen GP instances running on 3rd Gen Intel® Xeon® Scalable Processors can improve performance and value

Highlights of this report

Up to 10.8x the OLTP performance

Handle more database requests[1]

Up to 3.8x the OLAP/ analytics performance

Gain quicker insights from datasets[2]

Up to 7.8x better performance/price ratio

Improve the return on your cloud investment[3]

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

Executive summary

Microsoft Azure SQL Managed Instance offers two General Purpose (GP) tiers: classic GP and Next-gen GP. While the classic GP tier provides a proven managed SQL Server platform, its storage I/O architecture can limit performance as workloads grow. The newer Next-gen GP tier addresses these constraints with higher IOPS and expanded memory.

PT used online transaction processing (OLTP) and online analytical processing (OLAP) benchmarks to see if those enhancements deliver performance gains that justify the cost. We tested classic GP (baseline) and Next-gen GP (baseline), both with default storage performance, and Next-gen GP (boosted), with expanded memory and higher IOPS ceiling. We calculated raw performance and cost-per-operation metrics under three pricing models.

Next-gen GP outperformed classic GP baseline on both OLTP and OLAP workloads. Factoring in cost, Next-gen GP also demonstrated a better performance/price ratio.

Key outcomes of moving from classic GP

OLTP

- 2.66x the throughput with Next-gen GP (baseline)
- 10.88x the throughput with Next-gen GP (boosted)
- Up to 7.80x the performance per dollar

OLAP

- 1.40x the query sets per hour with Next-gen GP (baseline)
- 3.89x the query sets per hour with Next-gen GP (boosted)
- Up to 2.98x the performance per dollar

Table of contents

Highlights of this report 1

Executive summary 2

The case for re-evaluating your General Purpose tier 4

How we tested 4

Benchmark performance results 5

- TPROC-C (OLTP) 5
- TPROC-H (OLAP/analytics) 6

Price/performance analysis with cost-optimization purchasing options 7

- TPROC-C (OLTP) 7
- TPROC-H (OLAP/analytics) 8

Key takeaways 9

Conclusion 9

Appendix A: Detailed methodology 10

Appendix B: Configuration tables 16

Appendix C: Results 18

Appendix D: Monthly cost by configuration and purchasing option 19

Appendix E: Cost analysis 20

The case for re-evaluating your General Purpose tier

Enterprise organizations running SQL workloads in the cloud face a persistent tension between performance headroom and cost control. As databases expand and the volume of transactions increases, the cost of over-provisioning to maintain acceptable response times can threaten the economic case for cloud migration. Many organizations have come to rely on the Azure SQL Managed Instance General Purpose tier, which provides a fully managed, highly available SQL Server environment without the premium price of Business Critical. However, the classic GP architecture carries storage I/O constraints that can become a bottleneck for workloads with high concurrency or large analytical query sets.

With the Next-gen General Purpose tier, Microsoft addresses this directly, introducing a higher IOPS ceiling and expanded memory options within the same managed service model and general price band. For database administrators and cloud architects evaluating whether to migrate existing classic GP instances or provision new workloads on Next-gen GP, the practical question is straightforward: Does the architectural improvement translate to measurable gains under real workloads, and do those gains hold up when you factor in cost? In this report, we answer that question with the results of hands-on benchmark testing across both tiers.

How we tested

PT used HammerDB, a widely adopted open-source database benchmarking tool, to evaluate performance across three Azure SQL Managed Instance configurations. Testing covered two workload types:

- TPROC-C (OLTP): Simulates an online transaction processing workload modeled on the TPC-C benchmark, measuring the number of new-order transactions completed per minute (NOPM).
- TPROC-H (OLAP/analytics): Simulates a decision-support workload modeled on the TPC-H benchmark, measuring total time to complete a set of complex analytical queries.

The results of these two workloads are not comparable to those on the TPC-C and TPC-H benchmarks. All configurations used Premium-series hardware. Table 1 presents the three configurations we tested.

Table 1: Configuration details for the Microsoft Azure instances we tested. Source: PT.

Configuration	vCores	Memory (GB)	Storage (GB)	IOPS	Hardware
Classic GP (baseline)	8	56	1,024 for TPROC-C 2,048 for TPROC-H	2,300	Premium series
Next-gen GP (baseline)	8	56	1,024 for TPROC-C 2,048 for TPROC-H	3,072 for TPROC-C 6,144 for TPROC-H	Premium series
Next-gen GP (boosted)	8	96	1,024 for TPROC-C 2,048 for TPROC-H	12,800	Premium series

Configuration notes:

- Next-gen GP (boosted) uses expanded memory (96 GB) and a significantly higher IOPS ceiling (12,800) available in the Next-gen GP tier. These are not available in the classic GP (baseline) tier.
- IOPS for Next-gen (baseline) difference is greater than for classic GP (baseline) because storage size determines IOPS for Next-gen instances, while file size determines IOPS for classic GP instances.

PT ran multiple iterations of each benchmark on each configuration and recorded median results to account for variability. For each configuration, we captured:

- TPROC-C: New order transactions per minute (NOPM)
- TPROC-H: Total elapsed time to complete the full set of queries, which we convert to the number of query sets each configuration can complete in 1 hour
- Cost per operation: Hourly instance cost divided by throughput expressed as new orders per dollar (OLTP), or as query sets per dollar (OLAP)

Our source for pricing data was Azure public pricing pages as of June 23, 2026. We evaluated three purchasing scenarios:

- Pay-as-you-go: Standard on-demand pricing, which offers the greatest flexibility at the highest price
- 3-year reserved instances: Committed 3-year pricing, which typically offers a significant discount vs. a pay-as-you-go plan
- Azure Hybrid Benefit (AHB): Reserved pricing with an existing SQL Server license applied, further reducing instance cost. This is a good choice for organizations that wish to bring their own SQL Server licensing.

Benchmark performance results

TPROC-C (OLTP)

Figure 1 shows the TPROC-C results across the three test configurations. Higher NOPM indicates the ability to process more database transactions per minute, giving an organization greater throughput, lower latency, and more headroom to scale transaction-heavy workloads without over-provisioning resources.

Figure 1: TPROC-C results (NOPM). Higher is better. Source: PT.

Next-gen GP (baseline) delivered 2.66 times as many NOPM as classic GP (baseline) using the same vCore, memory, and storage configuration. The additional IOPS available to the Next-gen GP baseline configuration—3,072 vs. the default classic GP ceiling—enabled the instance to sustain higher transaction rates without storage contention.

Next-gen GP (boosted) produced the highest throughput in our TPROC-C testing, benefiting from both the expanded IOPS ceiling (12,800) and the additional memory (96 GB). Next-gen GP (boosted) delivered 10.88 times the NOPM of classic GP (baseline) and 4.08 times the NOPM of Next-gen GP (baseline).

TPROC-H (OLAP/analytics)

Figure 2 shows the TPROC-H results in terms of the number of query sets per hour that each instance processed. Higher numbers indicate the ability to execute complex analytical queries faster across large datasets, giving an organization greater throughput for decision-support and business intelligence workloads without over-provisioning resources.

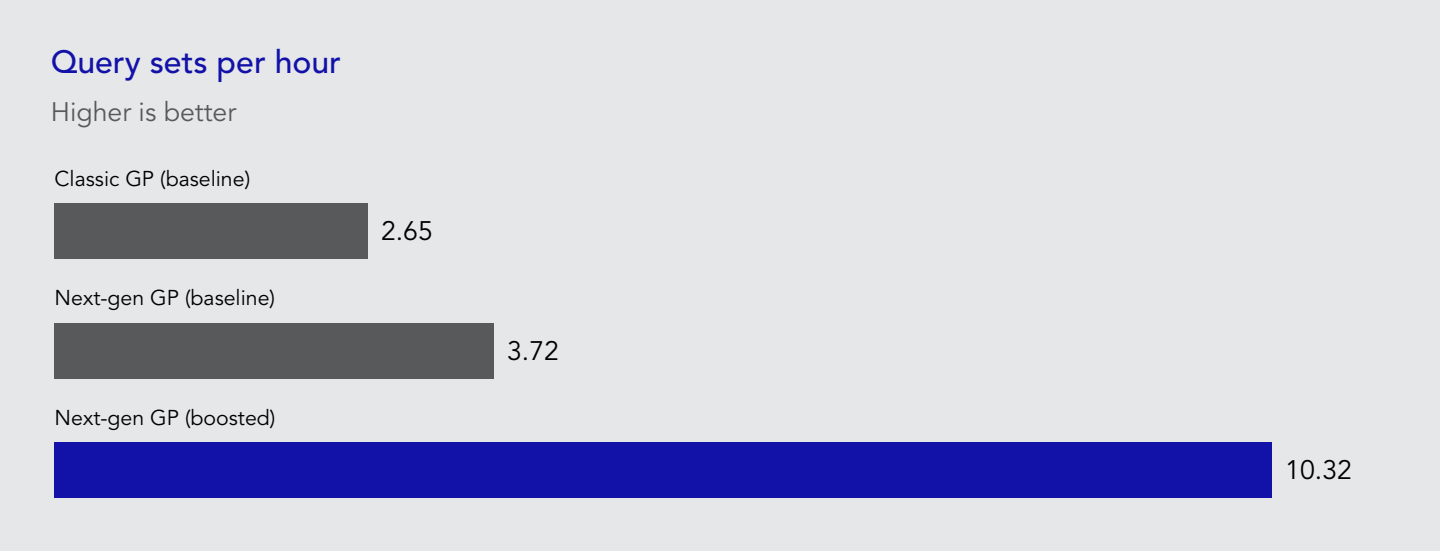

Figure 2: The number of query sets per hour each instance processed. Higher is better. Source: PT.

For analytics workloads, Next-gen GP (baseline) delivered 1.40 times the query sets per hour of classic GP (baseline). The higher baseline IOPS available to Next-gen GP reduced I/O wait time during the large sequential scans typical of analytical queries. Next-gen GP (boosted) processed the most query sets per hour, delivering 3.89 times as many query sets per hour as classic GP (baseline) and 2.77 times as many as Next-gen GP (baseline). The combination of higher memory and elevated IOPS allowed the system to process the data faster.

Price/performance analysis with cost-optimization purchasing options

Raw throughput numbers tell only part of the story. For organizations planning cloud database deployments, the more relevant question is how much throughput they can get per dollar of monthly spend, and how that ratio changes as they commit to longer terms or apply existing license entitlements. Azure SQL Managed Instance supports two cost-optimization purchasing options: 3-year reserved instance pricing and Azure Hybrid Benefit (AHB), which is available to customers with qualifying existing SQL Server licenses. We include pay-as-you-go pricing as a baseline for comparison.

A key characteristic of the classic and Next-gen GP Baseline configurations is that they carry identical monthly costs at every pricing tier.[4] This means the decision to move from classic GP to Next-gen GP hardware is purely a performance decision; no cost trade-off is necessary, and any throughput improvement flows directly to the price/performance ratio. Organizations that have already standardized on a purchasing option can upgrade to Next-gen compute without revisiting their budget.

TPROC-C (OLTP)

A higher number of new orders per dollar translates to more transactional capacity for the same spend, which directly affects how many concurrent users, application threads, or business processes a deployment can support before requiring a scale-up. Tables 2 through 4 show new orders per dollar for the three pricing tiers.

Table 2: TPROC-C new orders per dollar, pay-as-you-go pricing. Higher new orders per dollar is better. Source: PT.

Configuration	Hourly cost	New orders per dollar	vs. classic GP (baseline)
Classic GP (baseline)	$2.36	253,779.66	—
Next-gen GP (baseline)	$2.36	675,661.01	**2.66x**
Next-gen GP (boosted)	$3.29	1,981,969.60	**7.80**x

Table 3: TPROC-C new orders per dollar, 3-year reserved pricing. Higher new orders per dollar is better. Source: PT.

Configuration	Hourly cost	New orders per dollar	vs. classic GP (baseline)
Classic GP (baseline)	$1.58	379,063.29	—
Next-gen GP (baseline)	$1.58	1,009,215.18	**2.66x**
Next-gen GP (boosted)	$2.52	2,587,571.42	**6.82x**

Table 4: TPROC-C new orders per dollar, Azure Hybrid Benefit pricing. Higher new orders per dollar is better. Source: PT.

Configuration	Hourly cost	New orders per dollar	vs. classic GP (baseline)
Classic GP (baseline)	$0.78	767,846.15	—
Next-gen GP (baseline)	$0.78	2,044,307.69	**2.66x**
Next-gen GP (boosted)	$1.72	3,791,093.02	**4.93x**

Across every pricing tier, Next-gen GP Baseline delivered 2.66 times as many new orders per dollar as classic GP. For an organization running classic GP today, upgrading to Next-gen GP hardware means **handling more than twice the transactional load without any change to their monthly bill or purchasing model**. That gain would come entirely from generational improvements in compute efficiency, not from spending more.

For organizations with headroom to increase spend, Next-gen GP (boosted) would offer the highest absolute throughput per dollar of any configuration tested. At 3-year reserved pricing it reached 6.82 times the rate achieved by classic GP at a 58 percent higher monthly cost. Companies running high-concurrency applications, real-time processing pipelines, or workloads with strict SLA requirements on transaction latency are the natural fit for Next-gen GP (boosted): The additional monthly spend buys a disproportionate increase in capacity.

TPROC-H (OLAP/analytics)

Analytical workloads place different demands on the database. Rather than many short transactions, they involve complex multi-table queries that scan large portions of the dataset. We calculated how many query sets per hour the instances could perform and divided that by the hourly cost of the instances. Organizations running reporting, business intelligence, or mixed OLTP/analytics workloads on the same instance benefit directly from improvements in this metric—more query throughput per dollar means faster time-to-insight. Tables 5 through 7 show the number of TPROC-H query sets per dollar that each configuration achieved for the three pricing tiers.

Table 5: TPROC-H query sets per dollar, pay-as-you-go pricing. Higher is better. Source: PT.

Configuration	Hourly cost	Query sets per dollar	vs. classic GP (baseline)
Classic GP (baseline)	$2.52	1.05	—
Next-gen GP (baseline)	$2.52	1.47	**1.4x**
Next-gen GP (boosted)	$3.29	3.13	**2.98x**

Table 6: TPROC-H query sets per dollar, 3-year reserved pricing. Higher is better. Source: PT.

Configuration	Hourly cost	Query sets per dollar	vs. classic GP (baseline)
Classic GP (baseline)	$1.75	1.51	—
Next-gen GP (baseline)	$1.75	2.12	**1.4x**
Next-gen GP (boosted)	$2.68	3.85	**2.54x**

Table 7: TPROC-H query sets per dollar, Azure Hybrid Benefit pricing. Higher is better. Source: PT.

Configuration	Hourly cost	Query sets per dollar	vs. classic GP (baseline)
Classic GP (baseline)	$0.95	2.78	—
Next-gen GP (baseline)	$0.95	3.91	**1.4x**
Next-gen GP (boosted)	$1.88	5.48	**1.97x**

Next-gen GP Baseline delivered 1.4x the analytics throughput per dollar than classic GP at identical cost across all pricing tiers. The consistency of this improvement across pay-as-you-go, 3-year reserved, and Azure Hybrid Benefit pricing means the benefit is structural, not contingent on any particular purchasing decision. Organizations already on AHB or reserved pricing get the upgrade improvement without spending any more.

Next-gen GP Boosted extended the advantage further for analytics-heavy workloads. At 3-year reserved pricing, it reached 3.85 query sets per dollar, 2.54 times the rate achieved by classic GP. For organizations whose analytical workloads are growing faster than their budgets, this configuration offers a path to meaningful throughput increases without a proportional increase in spend.

Key takeaways

1. **Upgrading from classic GP to Next-gen GP (baseline) dramatically improved performance and costs nothing.** Simply moving to Next-gen GP (baseline)—with no increase in vCores, memory, or storage—more than doubled OLTP throughput. Because classic and Next-gen GP (baseline) cost the same, organizations running classic GP are leaving throughput capacity on the table.
2. **Next-gen GP (boosted) delivered outsized throughput gains for a moderate cost increase.** Starting at just a 30 percent cost increase (with increases up to 117 percent), the Boosted configuration returned up to 7.80 times the OLTP throughput per dollar and up to 2.98 times the analytics throughput per dollar. For workloads where throughput is the binding constraint—high-concurrency OLTP, time-sensitive analytics, or applications with aggressive growth trajectories—the additional spend buys capacity that would otherwise require significant architectural changes or horizontal scaling.

Conclusion

Next-gen GP delivered consistent performance and cost-efficiency advantages over classic GP across the OLTP and analytics workloads we tested. Next-gen GP (baseline) outperformed classic GP (baseline) on transaction throughput and analytics query throughput using identical vCore, memory, and storage configurations—a difference attributable directly to the higher IOPS floor built into the Next-gen GP architecture. Next-gen GP (boosted), with its expanded memory and elevated IOPS ceiling, extended that advantage, delivering 3.89 times the query sets per hour of classic GP (baseline). When we factor in pricing, Next-gen GP yielded a better performance/price ratio than classic GP across all three pricing scenarios we evaluated.

For organizations evaluating whether to migrate existing classic GP instances or provision new workloads, our results suggest that Next-gen GP can deliver meaningful gains without requiring a move to a higher service tier.

The science behind the report

In this section, we list our complete results and describe the solutions on which we tested and our test methodologies.

We concluded our hands-on testing on June 30, 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 30, 2026 or earlier. Unavoidably, these configurations may not represent the latest versions available when this report appears.

Appendix A: Detailed methodology

We used HammerDB 5.0 to perform TPROC-C and TPROC-H workload benchmarking on three Azure SQL Managed Instance configurations. The following procedure documents environment setup, database creation, and test execution.

Setting up the Azure SQL Managed Instance environment

Creating the SQL Managed Instance

1. Log into the Azure Portal and navigate to the SQL Managed Instance service.
2. Click Create.
3. Select your Subscription and Resource group.
4. Enter a name for the instance, and select the region you wish to work in. We used (US) West US 3.
5. Under Compute + storage, click Configure Managed Instance.
6. Under Service tier, select General Purpose.
7. Under Hardware generation, select your preferred hardware series. See Tables 9 and 10 for the generations we used for each instance.
8. Under vCores, select your desired number of vCores. We used 8 vCores for each of our Managed Instances.
9. Use the slider to specify the amount of Storage in GB to use. See Tables 9 and 10 for the volume size we used for each instance.
10. Choose your desired Backup storage redundancy, and click Apply. We used Locally-redundant backup storage.
11. Under Authentication, leave Use SQL authentication selected, and enter credentials for logging into the Azure SQL MI.
12. Click Next: Networking.
13. On the Networking tab, leave all defaults for the first instance created. For the three Managed Instances created after the first, select the virtual network created during the Configuring the Managed Instance Virtual Network stage below. Then click Next: Security.
14. On the Security tab, leave all defaults, and click Next: Additional settings.
15. On the Additional settings tab, set your time zone and maintenance window to your desired values, and click Next: Tags. We used (UTC-05:00) Eastern Time.
16. On the Tags tab, enter in any desired tags, and click Review + create.
17. Review your settings, and click Create.

Configuring the Managed Instance virtual network for client connection

1. Log into the Azure Portal, and navigate to the Resource groups service.
2. Select the Resource group containing your SQL MI Virtual network (vnet).
3. Select the virtual network for your MI.
4. In the left pane, select Subnets.
5. Click + Subnet.
6. Give the subnet a name, and leave the rest of the options default.
7. To create the client subnet, click OK.

Creating the client VM

1. Log into the Azure Portal, and navigate to the Virtual Machines service.
2. To open the Add VM wizard, click Create.
3. On the Basics tab, set the following:
4. From the drop-down menu, choose your Subscription.
5. From the drop-down menu, choose your Resource group.
6. Name the Virtual Machine.
7. From the drop-down menu, choose your Region.
8. Leave the Availability options set to No infrastructure redundancy required.
9. From the Image drop-down menu, choose Windows Server 2025 Datacenter.
10. Leave Azure Spot instance set to No.
11. Select the instance size you wish to use; we used Standard D8ds_v5.
12. Choose a new Username and Password for the Administrator account.
13. Leave Public inbound ports set to Allow selected ports.
14. For Select inbound ports, choose SSH (22).
15. On the Disks tab, set the following:
16. From the drop-down menu, for the OS disk type, choose Standard SSD.
17. Leave the default Encryption type.
18. On the Networking tab, set the following:
19. From the drop-down menu, choose the Virtual network subnet you created in the Configuring the Managed Instance virtual network for client connection section.
20. To create a new Public IP, select Create New.
21. Leave the rest of the settings at defaults.
22. On the Management tab, leave all defaults.
23. On the Advanced tab, leave all defaults.
24. On the Tags tab, add any tags you wish to use.
25. On the Review + create tab, review your settings, and click Create.

Configuring Microsoft Windows Server 2025 client

Configuring Windows Server 2025

1. Open Server Manager, and click Local Server.
2. Disable IE Enhanced Security Configuration.
3. Change the time zone to your local time zone.
4. Change the name of your server, and when prompted, reboot.
5. Open Server Manager again, and click Local Server.
6. Click to run updates.
7. Run updates, rebooting when prompted, until the server shows no new updates to install.

Installing C++ Redistributable

1. In a browser, navigate to: https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170#latest-supported-redistributable-version.
2. Download the latest x64 version.
3. Double-click the VC_redist.x64 executable.
4. Click OK to install.

Installing Microsoft ODBC 18 driver for SQL Server

1. In a browser, navigate to https://learn.microsoft.com/uk-ua/sql/connect/odbc/download-odbc-driver-for-sql-server?view=sql-server-ver15.
2. Download the newest version of Microsoft ODBC Driver 18 for SQL Server (x64). We used 18.6.2.1.
3. Double-click the msodbcsql.msi to open the installer, and click Next.
4. Click I accept the terms in the license agreement, and click Next.
5. Click Next.
6. Click Install.

Installing HammerDB 5.0

1. In a browser, navigate to https://github.com/TPC-Council/HammerDB/releases.
2. Download the HammerDB-5.0-Win-x64-Setup.exe executable.
3. Double-click the .exe file, choose English, and click OK.
4. Click Yes.
5. Click Next.
6. Choose a destination location, and click Next.
7. Click Next.
8. Click Finish.

Creating, copying, and restoring the test databases

Installing SSMS on client VM

1. In a browser, navigate to the SSMS download page: https://learn.microsoft.com/en-us/ssms/install/install.
2. Download vs_SSMS.exe
3. Double-click to open.
4. Click Continue.
5. Don't select any of the five add-ons from the Visual Studio add-on (just the SSMS Core components)
6. Click Install.
7. Launch and log into your Microsoft account.

Creating the TPROC-C database

1. Open SQL Server Management Studio.
2. Right-click Databases → New Database.
3. Name the database. We named ours tpcc.
4. Click the Files tab and pre-grow the data file to 300GB.
5. Pre-grow the log file to 100GB.
6. In the Options tab, set the Recovery Model to Full, and click OK to begin the creation process.

Populating the TPROC-C database

1. Open HammerDB, and click Options → Benchmark.
2. Choose MSSQL Server and TPROC-C.
3. Expand SQL Server → TPROC-C → Schema Build.
4. Double-click Options.
5. In the SQL Server field, enter the Endpoint FQDN of the system under test.
6. For Authentication, select SQL Server and enter the User ID and Password established when creating the SQL Managed Instance.
7. Ensure the TPROC-C SQL Server Database field lists the correct name of the database, in our case tpcc.
8. Under the Settings tab, Select 3,000 warehouses and 48 virtual users.
9. Uncheck "Use BCP Option"
10. Click OK.
11. Double-click Build.
12. To change the retention period of Azure's automatic backups to cover the length of the testing window, log in to the Azure portal and navigate to the Managed Instance you build the database on.
13. In the left column, navigate to Data management → Backups.
14. Select Retention Policies.
15. Enable the check next to the database, then select Configure Policies.
16. Adjust the slider to set the needed period of time for the PITR backup.
17. Select Apply.

Creating the TPROC-H database and its tables

1. Open SQL Server Management Studio.
2. Right-click Databases → New Database.
3. Name the database. We named ours tpch.
4. Click the Files tab, and pre-grow the data file to 300GB.
5. Pre-grow the log file to 100GB.
6. On the Options tab, set the Recovery Model to Full.
7. To start the build, click OK.

Populating the TPROC-H database

1. Open HammerDB, and click Options → Benchmark.
2. Choose MSSQL Server and TPROC-H.
3. Expand SQL Server → TPROC-H → Schema Build.
4. Double-click Options.
5. In the SQL Server field, enter the IP of the system under test.
6. For Authentication, select SQL Server and enter the User ID and Password established when creating the SQL Managed Instance.
7. Ensure the TPROC-H SQL Server Database field lists the correct name of the database, in our case tpch.
8. Under the Settings tab, set the MAXDOP to 8.
9. Check the box for Clustered Columnstore.
10. Set the Scale Factor to 300.
11. Click OK.
12. Double-click Build.
13. To change the retention period of Azure's automatic backups to cover the length of the testing window, log in to the Azure portal and navigate to the Managed Instance you build the database on.
14. In the left column, navigate to Data management → Backups.
15. Select Retention Policies.
16. Enable the check next to the database, and select Configure Policies.
17. Adjust the slider to set the needed period of time for the PITR backup.
18. Select Apply.

Copying a database to another Managed Instance

1. In the Azure portal, navigate to the Managed Instance containing the database you will copy.
2. In the left column, navigate to Data management → Backups.
3. Locate the database you will copy from the list and select Restore.
4. Enter the name for the database (tpcc or tpch in our case) and under Managed Instance select the Managed Instance you will copy the database to.
5. On the Data Source tab, set the Restore Point to be just after the Earliest Restore Point.
6. On the Security tab, leave all defaults.
7. On the Additional Settings tab, select Configure Retention to adjust the timespan of the backup to cover the testing window.
8. On the Tags tab, enter any relevant tags.
9. On the Review + Create tab, confirm that all the information is correct.
10. Select Create.

Restoring a database from Azure point-in-time backups

1. In the Azure portal, navigate to the Managed Instance you will restore to.
2. In the left column, navigate to Data management → Backups.
3. Under Show Databases, select Deleted.
4. Locate the correct database using the earliest PITR restore point timestamp, and click Restore.
5. Under Database Details, enter the name for the database (tpcc or tpch in our case) and ensure the Managed Instance selected is the correct Managed Instance you wish to restore to.
6. On the Data Source tab, set the Restore Point to be just after the Earliest Restore Point. Record this timestamp value and use it for every restore.
7. On the Security tab, leave all defaults.
8. On the Additional Settings tab, leave the default.
9. On the Tags tab, enter any relevant tags.
10. On the Review + Create tab, confirm that all the information is correct.
11. Select Create.

Performing the TPROC-C test on Windows Server 2025

On the client system, open SSMS and run the following query at the master level to ensure an automatic Managed Instance database backup is not occurring. If it is, wait until the backup completes (run the query again and ensure no rows appear) before moving on.

```
SELECT session_id, command, percent_complete, wait_type
FROM sys.dm_exec_requests
WHERE command = 'BACKUP DATABASE'
```

1. Start HammerDB.
2. Select Options → Benchmark.
3. Set the database server to SQL Server, and set the workload to TPROC-C.
4. Open the Options panel for the Driver Script.
5. In the SQL Server field, enter the Endpoint FQDN of the system under test.
6. Choose SQL Server Authentication, and change the SQL Server User Password to the password you chose during SQL Server setup.
7. Choose Timed Driver Script.
8. Change the Total Transactions per User to 1000000000.
9. Enter 15 for the Rampup Time, and 30 for the Test Duration.
10. Click OK.
11. Open the Options panel for the Virtual Users.
12. Set the Virtual User count. Refer to Table 12 to see the Virtual User count we used for each Managed Instance.
13. Select the following: Show Output, Log Output to Temp, and Use Unique Log Name.
14. Click OK.
15. Click Create Users.
16. Click Start to begin the test.
17. Once the test is complete, save the HammerDB results text file and Azure metrics output.
18. To repeat: Delete the SQL Managed Instance database file, then restore a new database following the steps in the Restoring a database from Azure's point-in-time backups section. Repeat the backup check query, VM reboot, and all steps above.
19. Run the test three times, and report the median run.

Performing the TPROC-H test on Windows Server 2025

On the client system, open SSMS and run the following query at the master level to ensure an automatic Managed Instance database backup is not occurring. If it is, wait until the backup completes (run the query again and ensure no rows appear) before moving on.

```
SELECT session_id, command, percent_complete, wait_type
FROM sys.dm_exec_requests
WHERE command = 'BACKUP DATABASE'
```

1. Start HammerDB.
2. Select Options → Benchmark.
3. Set the database server to SQL Server, and set the workload to TPROC-H.
4. Open the Options panel for the Schema.
5. In the SQL Server field, enter the Endpoint FQDN of the system under test.
6. Choose SQL Server Authentication, and change the SQL Server User Password to the password you chose during SQL Server setup.
7. Set the MAXDOP to 8, scale to 300, and check the box for Clustered Columnstore.
8. Click OK.
9. Open the Options panel for the Driver Script and confirm the Endpoint FQDN, SQL Server Authentication, and MAXDOP settings you just entered are correct.
10. Click OK to load the driver script.
11. Expand Virtual User, and double-click Options.
12. For the power test, choose one user. Check the boxes for Show Output, Log Output to Temp, and Use Unique Log Name.
13. Click OK.
14. Double-click Create users.
15. To begin the run, click Start.
16. Once the power test has finished, record the results, and click Stop.
17. Double-click Virtual User → Options.
18. Set the number of users to 5 for the throughput test.

19. Double-click Create users.
20. To begin the run, click Start.
21. Once the throughput test has finished, record the results, and click Stop.
22. To repeat, reboot the client VM, run the backup check query, and repeat all steps above.
23. Run the test three times, and report the median run.

Appendix B: Configuration tables

Table 8: General information of our configuration

General information	
Date testing ended	6/30/26
Cloud service provider (CSP)	Microsoft Azure
Region	West US 3

Table 9: Detailed information on the managed instances and test parameters we used for the TPROC-C workload.

System configuration information	Classic GP (baseline)	Next-gen GP (baseline)	Next-gen GP (boosted)
Tested by	Principled Technologies	Principled Technologies	Principled Technologies
Test date	6/9/2026	6/11/2026	6/12/2026
Instance generation	Classic General Purpose	Next-gen General Purpose	Next-gen General Purpose
Region	West US 3	West US 3	West US 3
Workload and version	HammerDB v5.0 TPROC-C	HammerDB v5.0 TPROC-C	HammerDB v5.0 TPROC-C
Workload-specific parameters	3,000 warehouses, 8 virtual users, timed test, 15-minute ramp up, 30-minute run time	3,000 warehouses, 8 virtual users, timed test, 15-minute ramp up, 30-minute run time	3,000 warehouses, 8 virtual users, timed test, 15-minute ramp up, 30-minute run time
Iterations and result choice	Three runs, median	Three runs, median	Three runs, median
Server platform	8vCore General Purpose Premium Non-Memory Optimized	8vCore General Purpose Premium Non-Memory Optimized	8vCore General Purpose Premium Non-Memory Optimized
SQL version	SQL Server 2025	SQL Server 2025	SQL Server 2025
Backup	Locally redundant	Locally redundant	Locally redundant
Processor			
Number of vCPU	8	8	8
Vendor and model	Intel® Xeon® Platinum 8370C	Intel Xeon Platinum 8370C	Intel Xeon Platinum 8370C
Core count (per processor)	32	32	32
Core frequency (GHz)	2.8	2.8	2.8
Stepping	6	6	6
Hyper-threading	Yes	Yes	Yes
Turbo	Yes	Yes	Yes
Memory module(s)			
Total memory in system (GB)	56	56	96
Data drive			
Drive size (GB)	1,024	1,024	1,024
IOPS	2,300	3,072	12,800

Table 10: Detailed information on the managed instances and test parameters we used for the TPROC-H workload.

System configuration information	Classic GP (baseline)	Next-gen GP (baseline)	Next-gen GP (boosted)
Tested by	Principled Technologies	Principled Technologies	Principled Technologies
Test date	6/26/2026	6/29/2026	6/30/2026
Instance generation	Classic General Purpose	Next-gen General Purpose	Next-gen General Purpose
Region	West US 3	West US 3	West US 3
Workload and version	HammerDB v5.0 TPROC-H	HammerDB v5.0 TPROC-H	HammerDB v5.0 TPROC-H
Workload-specific parameters	300 scale, 1 and 5 users, MAXDOP 8	300 scale, 1 and 5 users, MAXDOP 8	300 scale, 1 and 5 users, MAXDOP 8
Iterations and result choice	Three runs, median	Three runs, median	Three runs, median
Server platform	8vCore General Purpose Premium Non-Memory Optimized	8vCore General Purpose Premium Non-Memory Optimized	8vCore General Purpose Premium Non-Memory Optimized
SQL version	SQL Server 2025	SQL Server 2025	SQL Server 2025
Backup	Locally redundant	Locally redundant	Locally redundant
Processor			
Number of vCPU	8	8	8
Vendor and model	Intel Xeon Platinum 8370C	Intel Xeon Platinum 8370C	Intel Xeon Platinum 8370C
Core count (per processor)	32	32	32
Core frequency (GHz)	2.8	2.8	2.8
Stepping	6	6	6
Hyper-threading	Yes	Yes	Yes
Turbo	Yes	Yes	Yes
Memory module(s)			
Total memory in system (GB)	56	56	96
Data drive			
Drive size (GB)	2,048	2,048	2,048
IOPS	2,300	6,144	12,800

Table 11: VM Client system configuration

General information	
Operating system	Windows Server 2025 Datacenter
VM size	D8ds_v5
Type	General Purpose
vCPUs	8
Memory (GB)	32
Max IOPS	12,800
Local storage (GB)	300

Appendix C: 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 12: HammerDB TPROC-C new orders per minute (NOPM). Higher is better. Source: PT.

	Classic GP (baseline)	Next-gen GP (baseline)	Next-gen GP (boosted)
Virtual users	8	16	48
NOPM	9,982	26,576	108,678

Table 13: HammerDB TPROC-H throughput test results. Each stream ran the five query sets concurrently; completion time is the longest individual stream time. Query sets per hour = 3,600 / throughput completion time × 5 streams. Higher query sets per hour is better; lower completion times are better. Source: PT.

Metric	Classic GP (baseline)	Next-gen GP (baseline)	Next-gen GP (boosted)
Throughput run completion time (seconds)	6,772	4,827	1,744
Query sets per hour	2.65	3.72	10.32

Appendix D: Monthly cost by configuration and purchasing option

Note that Azure lists pay-as-you-go pricing for Next-gen GP (baseline) at $1,725.72 per month, ten cents higher than classic GP (baseline) at $1,725.62. This trivial gap likely reflects rounding in Azure's pricing calculator rather than a true cost difference, and does not materially affect the claims in this report that the two configurations carry identical monthly costs.

Table 14: Monthly Azure SQL Managed Instance cost for TPROC-C test configurations. Source: Azure pricing calculator.

Purchasing option	Classic GP (baseline)	Next-gen GP (baseline)	Next-gen GP (boosted)
Pay-as-you-go	$1,725.62	$1,725.72	$2,404.99
3-year reserved	$1,160.43	$1,160.43	$1,839.70
Azure Hybrid Benefit	$576.62	$576.62	$1,255.90

Table 15: Monthly Azure SQL Managed Instance cost for TPROC-H test configurations. Source: Azure pricing calculator.

Purchasing option	Classic GP (baseline)	Next-gen GP (baseline)	Next-gen GP (boosted)
Pay-as-you-go	$1,843.48	$1,843.48	$2,406.02
3-year reserved	$1,278.19	$1,278.19	$1,957.46
Azure Hybrid Benefit	$694.38	$694.38	$1,373.66

Appendix E: Cost analysis

- To determine a per-hour cost for each instance, we took the monthly costs listed on the Azure site/pricing catalog and divided them by 730 hours. (Azure claims to base monthly costs on 730 hours per month.)
- To determine OLTP new orders per hour (NOPH), we multiplied new orders per minute by 60. To determine new orders per dollar, we divided NOPH by the cost per hour.
- To determine OLAP query sets per dollar, we divided query sets per hour by the cost per hour.

Table 16: TPROC-C new orders per dollar by configuration and purchasing option. Higher new orders per dollar is better. Source: PT.

Configuration	Pay-as-you-go		3-year reserved		Azure Hybrid Benefit	
	Hourly cost	New orders per dollar	Hourly cost	New orders per dollar	Hourly cost	New orders per dollar
Classic GP (baseline)	$2.36	253,779.66	$1.58	379,063.29	$0.78	767,846.15
Next-gen GP (baseline)	$2.36	675,661.01	$1.58	1,009,215.18	$0.78	2,044,307.69
Next-gen GP (boosted)	$3.29	1,981,969.60	$2.52	2,587,571.42	$1.72	3,791,093.02

Table 17: TPROC-H query sets per dollar by configuration and purchasing option. Higher query sets per dollar is better. Source: PT.

Configuration	Pay-as-you-go		3-year reserved		Azure Hybrid Benefit	
	Hourly cost	Query sets per dollar	Hourly cost	Query sets per dollar	Hourly cost	Query sets per dollar
Classic GP (baseline)	$2.52	1.05	$1.75	1.51	$0.95	2.78
Next-gen GP (baseline)	$2.52	1.47	$1.75	2.12	$0.95	3.91
Next-gen GP (boosted)	$3.29	3.13	$2.68	3.85	$1.88	5.48

1. Next-gen GP (boosted) achieved 10.88x the OLTP throughput of classic GP (baseline) on the HammerDB TPROC-C workload.
2. Next-gen GP (boosted) achieved 3.89x the OLAP query sets per hour of classic GP (baseline) on the HammerDB TPROC-H workload.
3. Next-gen GP (boosted) delivered up to 7.80x the OLTP performance per dollar of classic GP, based on HammerDB TPROC-C workload and pay-as-you-go pricing.
4. Note that Azure lists pay-as-you-go pricing for Next-gen GP (baseline) at $1,725.72 per month, ten cents higher than classic GP (baseline) at $1,725.62. This trivial gap likely reflects rounding in Azure's pricing calculator rather than a true cost difference, and does not materially affect the claims in this report that the two configurations carry identical monthly costs.

July 2026

Commissioned by

Microsoft | intel

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 to generate a T-SQL query used to check for an in-progress automated Managed Instance backup and to confirm its completion before test execution began.

Primary contributors

Tech: Travis Hall

Writing: Laura W.

Design: Laura K.

PM: Greg C.

Principled Technologies®

Principled Technologies is a registered trademark of Principled Technologies, Inc. All other product names are the trademarks of their respective owners.

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.