ALIBABA CLOUD

Alibaba Cloud

ApsaraDB for RDS Performance White Paper

Document Version: 20220530

Alibaba Cloud

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

Document conventions

Style	Description	Example
Danger	A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	Danger: Resetting will result in the loss of user configuration data.
Warning	A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	Warning: Restarting will cause business interruption. About 10 minutes are required to restart an instance.
Notice	A caution notice indicates warning information, supplementary instructions, and other content that the user must understand.	Notice: If the weight is set to 0, the server no longer receives new requests.
Note	A note indicates supplemental instructions, best practices, tips, and other content.	Note: You can use Ctrl + A to select all files.
>	Closing angle brackets are used to indicate a multi-level menu cascade.	Click **Settings**> **Network**> **Set network type**.
Bold	Bold formatting is used for buttons , menus, page names, and other UI elements.	Click **OK**.
`Courier font`	Courier font is used for commands	Run the `cd /d C:/window` command to enter the Windows system folder.
Italic	Italic formatting is used for parameters and variables.	`bae log list --instanceid` *Instance_ID*
[] or [a|b]	This format is used for an optional value, where only one item can be selected.	`ipconfig [-all|-t]`
{} or {a|b}	This format is used for a required value, where only one item can be selected.	`switch {active|stand}`

Table of Contents

1.RDS for MySQL

1.1. ApsaraDB RDS for MySQL performance overview

ApsaraDB RDS is a stable, reliable, and scalable online database service that is built on top of the Apsara Distributed File System and high-performance SSDs of Alibaba Cloud. ApsaraDB RDS supports the MySQL, SQL Server, PostgreSQL, and MariaDB TX database engines and provides a complete suite of database O&M solutions, such as disaster recovery, backup, restoration, monitoring, and migration. These solutions relieves the need for tedious O&M workloads.

All parameters that are used in ApsaraDB RDS have been tested and optimized over years of production practices that are conducted by a team of experienced database administrators (DBAs) from Alibaba Cloud. These DBAs have continued to optimize each ApsaraDB RDS instance throughout the lifecycle of the instance to ensure that the instance runs at its optimal configuration. In addition, all server hardware that is used by ApsaraDB RDS has passed the tests of multiple concerned parties. This ensures that ApsaraDB RDS can deliver optimal performance and high stability.

For more information about the performance test results of ApsaraDB RDS for MySQL, see the following topics:

- Test results of ApsaraDB RDS instances that run MySQL 8.0
- Test results of ApsaraDB RDS instances that run MySQL 5.7
- Test results of ApsaraDB RDS instances that run MySQL 5.6

1.2. Test guidelines

This topic provides guidelines for new users to test the performance of an ApsaraDB RDS for MySQL instance and submit a test report. After you submit a test report, you can receive additional discounts for instance renewal. If your test report is rated excellent, you are offered with substantial incentives from Alibaba Cloud.

Prerequisites

- An ApsaraDB RDS for MySQL instance is created in the ApsaraDB for RDS console. For more information, see Create an ApsaraDB RDS for MySQL instance.
- An ECS instance is created. For more information, see Create an instance by using the wizard.

Context

ApsaraDB RDS for MySQL test activities.

The following performance metrics are tested:

- Transactions per second (TPS)

  The number of transactions that are executed by the RDS instance per second. A transaction is executed only after it is committed.

  - Use SysBench to test the performance of online transactional processing (OLTP) for executing a read/write transaction that consists of 18 read and write SQL statements.

- Use SysBench to test the performance of OLTP for executing a read-only transaction that consists of 14 read SQL statements: 10 SQL statements are used to query data based on primary keys and 4 SQL statements are used to query data based on specified ranges.
- Use SysBench to test the performance of OLTP for executing a write-only transaction that consists of four write SQL statements: two UPDATE statements, one DETELE statement, and one INSERT statement.

- Queries per second (QPS)

  The number of SQL statements that are executed by the RDS instance per second. These SQL statements include INSERT, SELECT, UPDATE, DETELE, and COMMIT.

SysBench parameters

Parameter	Description
db-driver	The database engine that the RDS instance runs.
mysql-host	The endpoint used to connect to the RDS instance.
mysql-port	The port used to connect to the RDS instance.
mysql-user	The username of the account used to manage the RDS instance.
mysql-password	The password of the account used to manage the RDS instance.
mysql-db	The name of the RDS instance
table_size	The size of tables used for the test.
tables	The number of tables used for the test.
events	The number of requests sent for the test.
time	The time taken for the test.
threads	The number of threads invoked for the test.
percentile	The percentage of execution durations that you want to analyze for the test to obtain an average execution duration. The default value is 95%. It allows you to obtain the average time that is required to execute a request for 95% of all the scenarios.
report-interval	The time interval at which you want to generate a test progress report. The value 0 specifies not to generate test progress reports but only to generate a final test report.

Parameter	Description
skip-trx	Specifies whether to skip transactions. Valid values: • 1: specifies to skip transactions. • 0: specifies not to skip transactions.

Test procedure

1. Log on to your ECS instance and run the following commands to install SysBench:

```
yum install gcc gcc-c++ autoconf automake make libtool mysql-devel git mysql
git clone https://github.com/akopytov/sysbench.git
## Download the SysBench software package from GitHub.
cd sysbench
## Open the SysBench installation directory.
git checkout 1.0.18
## Switch to SysBench 1.0.18.
./autogen.sh
## Execute the autogen.sh script.
./configure --prefix=/usr --mandir=/usr/share/man
make
## Compile SysBench.
make install
```

2. Test the performance of OLTP for executing read/write, read-only, and write-only transactions.
   - Test the performance of OLTP for executing read/write transactions.

     Run the following commands to perform the test (for more information, see SysBench parameters):

```
##Prepare the data used for the test.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX oltp_read_write prepare
##Run your workload.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX --percentile=95 --report-interval=1 oltp_read_write run
##Delete the data used for the test.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX --percentile=95  oltp_read_write cleanup
```

     Sample test results:

     - QPS: 23869.32
     - TPS: 1193.47
     - Response time (RT): 36.89 ms

```
[ 55s ] thds: 16 tps: 1238.00 qps: 24720.98 (r/w/o: 17294.99/4951.00/2475.00) lat (ms,95%): 35.59 err/s: 0.00 reconn/s: 0.00
[ 56s ] thds: 16 tps: 1195.00 qps: 23924.02 (r/w/o: 16752.01/4782.00/2390.00) lat (ms,95%): 36.24 err/s: 0.00 reconn/s: 0.00
[ 57s ] thds: 16 tps: 1235.00 qps: 24714.10 (r/w/o: 17308.07/4935.02/2471.01) lat (ms,95%): 36.24 err/s: 0.00 reconn/s: 0.00
[ 58s ] thds: 16 tps: 1230.99 qps: 24594.83 (r/w/o: 17205.88/4926.97/2461.98) lat (ms,95%): 36.24 err/s: 0.00 reconn/s: 0.00
[ 59s ] thds: 16 tps: 1187.00 qps: 23786.05 (r/w/o: 16655.03/4757.01/2374.00) lat (ms,95%): 36.89 err/s: 0.00 reconn/s: 0.00
[ 60s ] thds: 16 tps: 1212.00 qps: 24252.93 (r/w/o: 16964.95/4863.99/2423.99) lat (ms,95%): 36.24 err/s: 0.00 reconn/s: 0.00
SQL statistics:
    queries performed:
        read:                            1003268
        write:                           286648
        other:                           143324
        total:                           1433240
    transactions:                        71662  (1193.47 per sec.)
    queries:                             1433240 (23869.32 per sec.)
    ignored errors:                      0      (0.00 per sec.)
    reconnects:                          0      (0.00 per sec.)

General statistics:
    total time:                          60.0439s
    total number of events:              71662

Latency (ms):
         min:                                    4.48
         avg:                                   13.40
         max:                                  142.82
         95th percentile:                       36.89
         sum:                                 960169.64

Threads fairness:
    events (avg/stddev):           4478.8750/88.00
    execution time (avg/stddev):   60.0106/0.00
```

- Test the performance of OLTP for executing read-only transactions.

  Run the following commands to perform the test (for more information, see SysBench parameters):

```
##Prepare the data used for the test.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX  oltp_read_only prepare
##Run your workload.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX --percentile=95 --skip-trx=1 --report-interval=1 oltp_read_only run
##Delete the data used for the test.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX --percentile=95 oltp_read_only cleanup
```

  Sample test results:

  - QPS: 26130.73
  - RT: 33.72 ms

```
[ 55s ] thds: 16 tps: 1881.00 qps: 26417.97 (r/w/o: 26417.97/0.00/0.00) lat (ms,95%): 33.72 err/s: 0.00 reconn/s: 0.00
[ 56s ] thds: 16 tps: 1831.01 qps: 25564.09 (r/w/o: 25564.09/0.00/0.00) lat (ms,95%): 34.33 err/s: 0.00 reconn/s: 0.00
[ 57s ] thds: 16 tps: 1914.99 qps: 26829.90 (r/w/o: 26829.90/0.00/0.00) lat (ms,95%): 33.12 err/s: 0.00 reconn/s: 0.00
[ 58s ] thds: 16 tps: 1930.00 qps: 27013.02 (r/w/o: 27013.02/0.00/0.00) lat (ms,95%): 33.12 err/s: 0.00 reconn/s: 0.00
[ 59s ] thds: 16 tps: 1901.00 qps: 26569.06 (r/w/o: 26569.06/0.00/0.00) lat (ms,95%): 33.72 err/s: 0.00 reconn/s: 0.00
[ 60s ] thds: 16 tps: 1933.99 qps: 27157.92 (r/w/o: 27157.92/0.00/0.00) lat (ms,95%): 33.12 err/s: 0.00 reconn/s: 0.00
SQL statistics:
    queries performed:
        read:                            1568952
        write:                           0
        other:                           0
        total:                           1568952
    transactions:                        112068 (1866.48 per sec.)
    queries:                             1568952 (26130.73 per sec.)
    ignored errors:                      0      (0.00 per sec.)
    reconnects:                          0      (0.00 per sec.)

General statistics:
    total time:                          60.0410s
    total number of events:              112068

Latency (ms):
         min:                                    3.03
         avg:                                    8.57
         max:                                   84.14
         95th percentile:                       33.72
         sum:                                960237.13

Threads fairness:
    events (avg/stddev):           7004.2500/96.69
    execution time (avg/stddev):   60.0148/0.01
```

- Test the performance of OLTP for executing write-only transactions.

  Run the following commands to perform the test (for more information, see SysBench parameters):

```
##Prepare the data used for the test.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX  oltp_write_only prepare
##Run your workload.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX --percentile=95 --report-interval=1 oltp_write_only run
##Delete the data used for the test.
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql
-password=XXX --mysql-db=sbtest --table_size=25000 --tables=100 --events=0 --time=300
--threads=XXX --percentile=95  oltp_write_only cleanup
```

  Sample test results:

  - TPS: 4255.01
  - RT: 16.71 ms

```
[ 56s ] thds: 16 tps: 4458.17 qps: 26789.02 (r/w/o: 0.00/17871.68/8917.34) lat (ms,95%): 14.21 err/s: 0.00 reconn/s: 0.00
[ 57s ] thds: 16 tps: 4416.02 qps: 26472.15 (r/w/o: 0.00/17640.10/8832.05) lat (ms,95%): 16.41 err/s: 0.00 reconn/s: 0.00
[ 58s ] thds: 16 tps: 3919.11 qps: 23508.64 (r/w/o: 0.00/15670.43/7838.21) lat (ms,95%): 17.63 err/s: 0.00 reconn/s: 0.00
[ 59s ] thds: 16 tps: 4089.00 qps: 24559.02 (r/w/o: 0.00/16381.01/8178.01) lat (ms,95%): 18.28 err/s: 0.00 reconn/s: 0.00
[ 60s ] thds: 16 tps: 4353.28 qps: 26100.68 (r/w/o: 0.00/17394.12/8706.56) lat (ms,95%): 19.65 err/s: 0.00 reconn/s: 0.00
SQL statistics:
    queries performed:
        read:                            0
        write:                           1021552
        other:                           510776
        total:                           1532328
    transactions:                        255388 (4255.01 per sec.)
    queries:                             1532328 (25530.04 per sec.)
    ignored errors:                      0      (0.00 per sec.)
    reconnects:                          0      (0.00 per sec.)

General statistics:
    total time:                          60.0192s
    total number of events:              255388

Latency (ms):
         min:                                    1.28
         avg:                                    3.76
         max:                                  182.50
         95th percentile:                       16.71
         sum:                               959762.69

Threads fairness:
    events (avg/stddev):           15961.7500/525.08
    execution time (avg/stddev):   59.9852/0.00
```

3. Download the ApsaraDB RDS for MySQL performance test report template, prepare your own test report based on the template, and then submit your test report.

1.3. Test details

1.3.1. Test environment

This topic describes the environment that is used to test the performance of an ApsaraDB RDS for MySQL instance.

- The ECS and RDS instances that you use for testing must reside in the same zone of the same region. All the tests must be performed in Zone I of the China (Hangzhou) region.
- The ECS and RDS instances that you use for testing must use the VPC network type and reside in the same VPC.
- The ECS instance that you use for testing must meet the following requirements:
  - The ECS instance must use the ecs.g6.8xlarge instance type.
  - The ECS instance must run the 64-bit CentOS 7.4 operating system.

> **Note** The ECS instance uses high specifications to ensure that it does not limit the optimal performance of the RDS instance during testing.

- The RDS instance that you use for testing must meet the following requirements:
  - The RDS instance must use local SSDs.
  - The RDS instance must belong to the general-purpose instance family.
  - The RDS instance must use the default parameter template that is designed for ApsaraDB RDS for MySQL in the High-availability Edition equipped with the InnoDB storage engine. For more information, see Manage the parameters of an ApsaraDB for RDS instance by using a parameter template.

1.3.2. Test scenarios

This topic describes the scenarios that are used to test the performance of an ApsaraDB RDS for MySQL instance.

Table schema

```
CREATE TABLE `sbtest100` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `k` int(11) NOT NULL DEFAULT '0',
  `c` char(120) NOT NULL DEFAULT '',
  `pad` char(60) NOT NULL DEFAULT '',
  PRIMARY KEY (`id`),
  KEY `k_100` (`k`)
) ENGINE=InnoDB AUTO_INCREMENT=25001 DEFAULT CHARSET=utf8
```

OLTP with read/write operations

SQL type	Ratio	SQL statement
point_selects	10	`SELECT c FROM sbtest100 WHERE id=?`
simple_ranges	1	`SELECT c FROM sbtest100 WHERE id BETWEEN ? AND ?`
sum_ranges	1	`SELECT SUM(k) FROM sbtest100 WHERE id BETWEEN ? AND ?`
order_ranges	1	`SELECT c FROM sbtest100 WHERE id BETWEEN ? AND ? ORDER BY c`
distinct_ranges	1	`SELECT DISTINCT c FROM sbtest100 WHERE id BETWEEN ? AND ? ORDER BY c`
index_updates	1	`UPDATE sbtest100 SET k=k+1 WHERE id=?`
non_index_updates	1	`UPDATE sbtest100 SET c=? WHERE id=?`

SQL type	Ratio	SQL statement
deletes	1	`DELETE FROM sbtest100 WHERE id=?`
inserts_ignore	1	`INSERT IGNORE INTO sbtest100 (id, k, c, pad) VALUES (?, ?, ?, ?)`

1.3.3. Test tools

This topic describes how to install the SysBench test tool on an ECS instance.

Background information

SysBench is a modular, cross-platform, and multi-threaded benchmark tool that is used to evaluate the core operating parameters of a system running a heavily loaded database. It allows you to obtain the performance of your database system without the need to configure complex database benchmark parameters or install database software.

Procedure

The stress tests of ApsaraDB RDS for MySQL use SysBench 1.0.20. For more information, visit SysBench.

1. Run the following commands to install SysBench on an ECS instance:

```
yum install gcc gcc-c++ autoconf automake make libtool bzr mysql-devel git mysql
git clone https://github.com/akopytov/sysbench.git
## Download the SysBench software package from GitHub.
cd sysbench
## Open the SysBench installation directory.
git checkout 1.0.20
## Switch to SysBench 1.0.20.
./autogen.sh
## Execute the autogen.sh script.
./configure --prefix=/usr --mandir=/usr/share/man
make
## Compile SysBench.
make install
```

2. Run the following commands to configure the SysBench client. Then the kernel can use all available CPUs to process data packets with the minimum context switches between CPUs. Two CPUs are used by default.

```
sudo sh -c 'for x in /sys/class/net/eth0/queues/rx-*; do echo ffffffff>$x/rps_cpus; don
e'
sudo sh -c "echo 32768 > /proc/sys/net/core/rps_sock_flow_entries"
sudo sh -c "echo 4096 > /sys/class/net/eth0/queues/rx-0/rps_flow_cnt"
sudo sh -c "echo 4096 > /sys/class/net/eth0/queues/rx-1/rps_flow_cnt"
```

> **Note** ffffffff specifies to use 32 CPUs. Each letter f represents four CPUs. You can change the value of this parameter based on your business requirements. For example, if your ECS instance uses eight CPUs, enter ff.

1.3.4. Test procedure

This topic describes how to test the performance of an ApsaraDB RDS for MySQL instance.

Procedure

Follow these steps to test the read/write performance of an ApsaraDB RDS for MySQL instance:

1. Prepare the data used for the test.

```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql-password=XXX --mysql-db=sbtest --table_size=25000 --tables=250 --events=0 --time=600  oltp_read_write prepare
```

2. Run your workload.

```
sysbench --db-driver=mysql  --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql-password=XXX --mysql-db=sbtest --table_size=25000 --tables=250 --events=0 --time=600 --threads=XXX --percentile=95 --report-interval=1 oltp_read_write run
```

3. Delete the data used for the test.

```
sysbench --db-driver=mysql  --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql-password=XXX --mysql-db=sbtest --table_size=25000 --tables=250 --events=0 --time=600 --threads=XXX --percentile=95  oltp_read_write cleanup
```

1.3.5. Performance metrics

This topic describes the metrics that are used to test the performance of ApsaraDB RDS for MySQL instances.

Performance metrics

Reads/Writes: the total number of read and write operations on a database within 60 seconds

1.4. Test results of ApsaraDB RDS instances that run MySQL 8.0

This topic describes the performance test results of general-purpose ApsaraDB RDS instances that run MySQL 8.0.

> **Note**
> - To better simulate the production environment, the Reads/Writes metric is used for this stress testing.
> - The performance test results are for reference only. For more information about how to use an ApsaraDB RDS instance that runs MySQL 8.0, see Troubleshoot slow SQL statements on an ApsaraDB RDS for MySQL instance.

Environment

In this stress testing, SysBench is used to test the performance of the five RDS instances that use local SSDs. These RDS instances use different instance types.

- Instance types: rds.mysql.t1.small, rds.mysql.s2.large, rds.mysql.m1.medium, rds.mysql.c1.xlarge, and rds.mysql.c2.xlarge
- Instance family: general-purpose
- Edition: RDS High-availability Edition
- Storage type: local SSD

Configurations

The performance is significantly affected by the **data volume**, **stress testing duration**, and **parameter settings**. The following configurations are made for this stress testing:

- Data volume: The data volume and the number of tables on each RDS instance that you want to test are different. Therefore, the test results for some of the five RDS instances may be similar.
- Stress testing duration: The stress testing duration is 60 seconds for all the RDS instances.
- Parameter settings:
  - `sync_binlog=1` and `innodb_flush_log_at_trx_commit=1`: ensure that the data submitted each time is completely written to disks.
  - `rpl_semi_sync_master_enabled=ON`: enables the semi-synchronous mode for an RDS instance to ensure data consistency between the primary and secondary RDS instances.
  - `Performance_schema=ON`: automatically enables Performance Schema for RDS instances that use instance types with a memory size greater than or equal to 8 GB.

    > **Note** These parameter settings are contained in the standard parameter template for ApsaraDB RDS for MySQL. If the standard parameter template is applied to all the RDS instances, data consistency is maximized, and the test environment is more similar to the production environment.

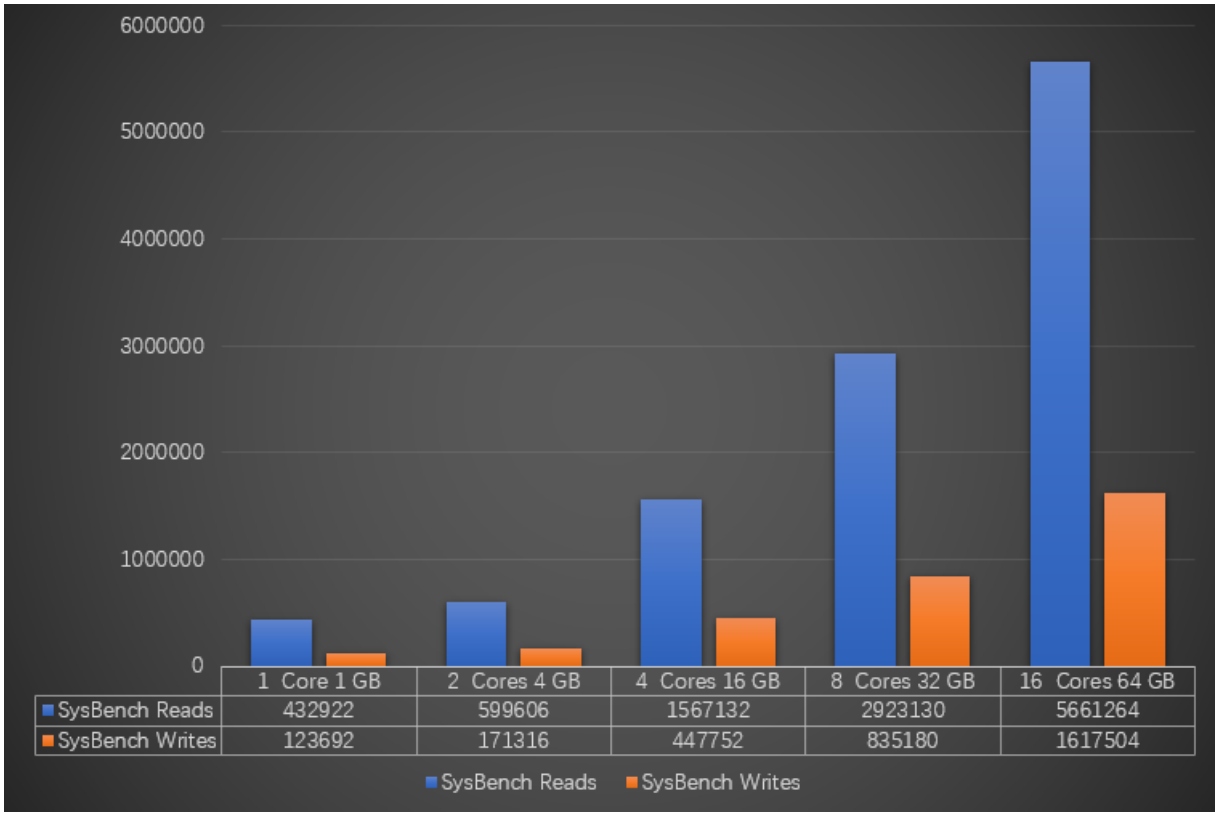
Results

Two types of queries can be used to perform stress testing. You can determine the type of stress testing based on your data volume.

- Stress testing for cache-based queries: This type of stress testing is suitable for scenarios that involve a small amount of data. You can store all data in the InnoDB buffer pool for access. For more information about how to change the size of the InnoDB buffer pool, see Change the size of the InnoDB buffer pool for an ApsaraDB RDS for MySQL instance.
- Stress testing for disk I/O-based queries: This type of stress testing is suitable for scenarios that

involve a large amount of data. You can store only the most frequently used data in the InnoDB buffer pool for access. During the stress testing, data is read from or written to disks, and the InnoDB buffer pool is updated.

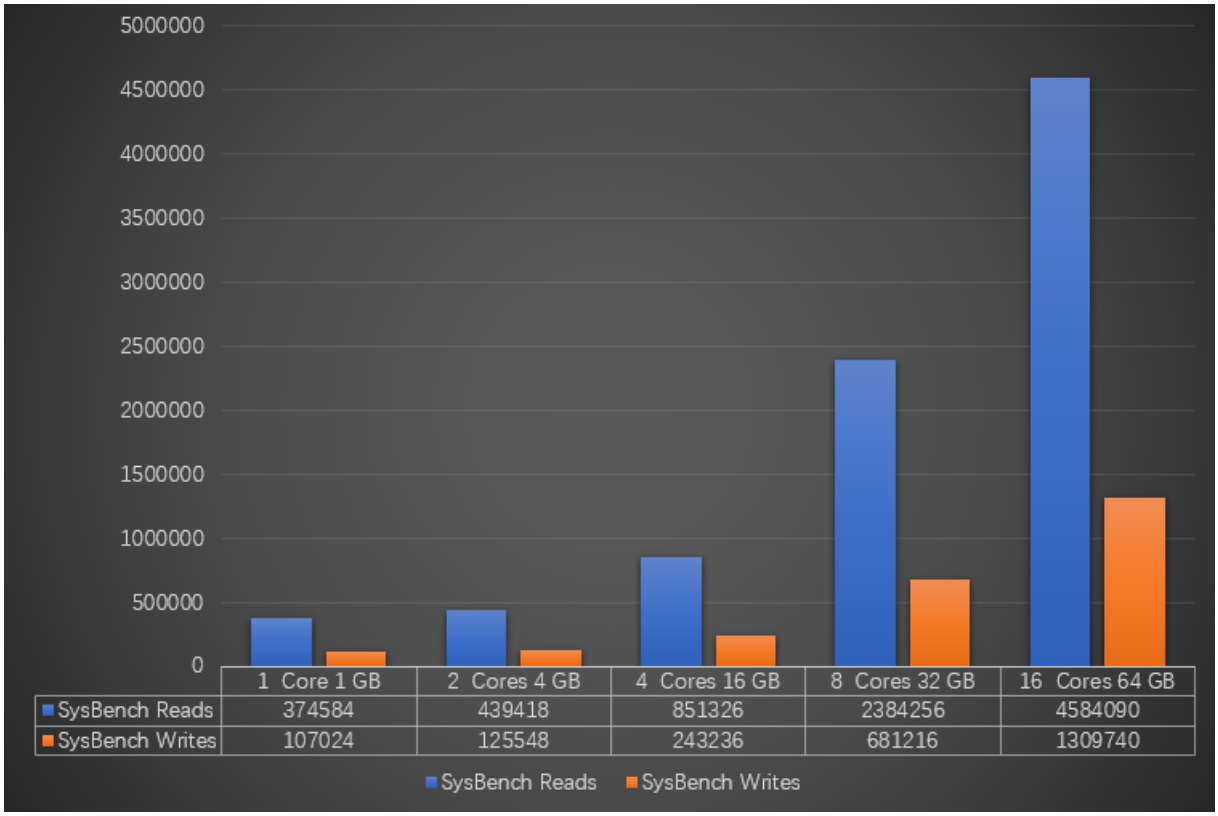
Type 1: Stress testing for cache-based queries



Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
1 core and 1 GB of memory (rds.mysql.t1.small)	25,000	32	300	600	8	432,922	123,692
2 cores and 4 GB of memory (rds.mysql.s2.large)	25,000	32	1,200	2,000	8	599,606	171,316

Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
4 cores and 16 GB of memory (rds.mysql.m1.medium)	25,000	128	4,000	7,000	16	1,567,132	447,752
8 cores and 32 GB of memory (rds.mysql.c1.xlarge)	25,000	128	8,000	12,000	32	2,923,130	835,180
16 cores and 64 GB of memory (rds.mysql.c2.xlarge)	25,000	128	16,000	14,000	64	5,661,264	1,617,504

Type 2: Stress testing for disk I/O-based queries



Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
1 core and 1 GB of memory (rds.mysql.t1.small)	80,000	32	300	600	8	374,584	107,024
2 cores and 4 GB of memory (rds.mysql.s2.large)	80,000	32	1,200	2,000	8	439,418	125,548
4 cores and 16 GB of memory (rds.mysql.m1.medium)	800,000	128	4,000	7,000	16	851,326	243,236
8 cores and 32 GB of memory (rds.mysql.c1.xlarge)	800,000	128	8,000	12,000	32	2,384,256	681,216
16 cores and 64 GB of memory (rds.mysql.c2.xlarge)	800,000	128	16,000	14,000	64	4,584,090	1,309,740

1.5. Test results of ApsaraDB RDS instances that run MySQL 5.7

This topic describes the performance test results of general-purpose ApsaraDB RDS instances that run MySQL 5.7.

> **Note**
>
> - To better simulate the production environment, the Reads/Writes metric is used for this stress testing.
> - The performance test results are for reference only. For more information about how to use an ApsaraDB RDS instance that runs MySQL 5.7, see Troubleshoot slow SQL statements on an ApsaraDB RDS for MySQL instance.

Environment

In this stress testing, SysBench is used to test the performance of the five RDS instances that use local SSDs. These RDS instances use different instance types.

- Instance types: rds.mysql.t1.small, rds.mysql.s2.large, rds.mysql.m1.medium, rds.mysql.c1.xlarge, and rds.mysql.c2.xlarge
- Instance family: general-purpose
- Edition: RDS High-availability Edition
- Storage type: local SSD

Configurations

The performance is significantly affected by the **data volume**, **stress testing duration**, and **parameter settings**. The following configurations are made for this stress testing:

- Data volume: The data volume and the number of tables on each RDS instance that you want to test are different. Therefore, the test results for some of the five RDS instances may be similar.
- Stress testing duration: The stress testing duration is 60 seconds for all the RDS instances.
- Parameter settings:
  - `sync_binlog=1` and `innodb_flush_log_at_trx_commit=1`: ensure that the data submitted each time is completely written to disks.
  - `rpl_semi_sync_master_enabled=ON`: enables the semi-synchronous mode for an RDS instance to ensure data consistency between the primary and secondary RDS instances.
  - `Performance_schema=ON`: automatically enables Performance Schema for RDS instances that use instance types with a memory size greater than or equal to 8 GB.

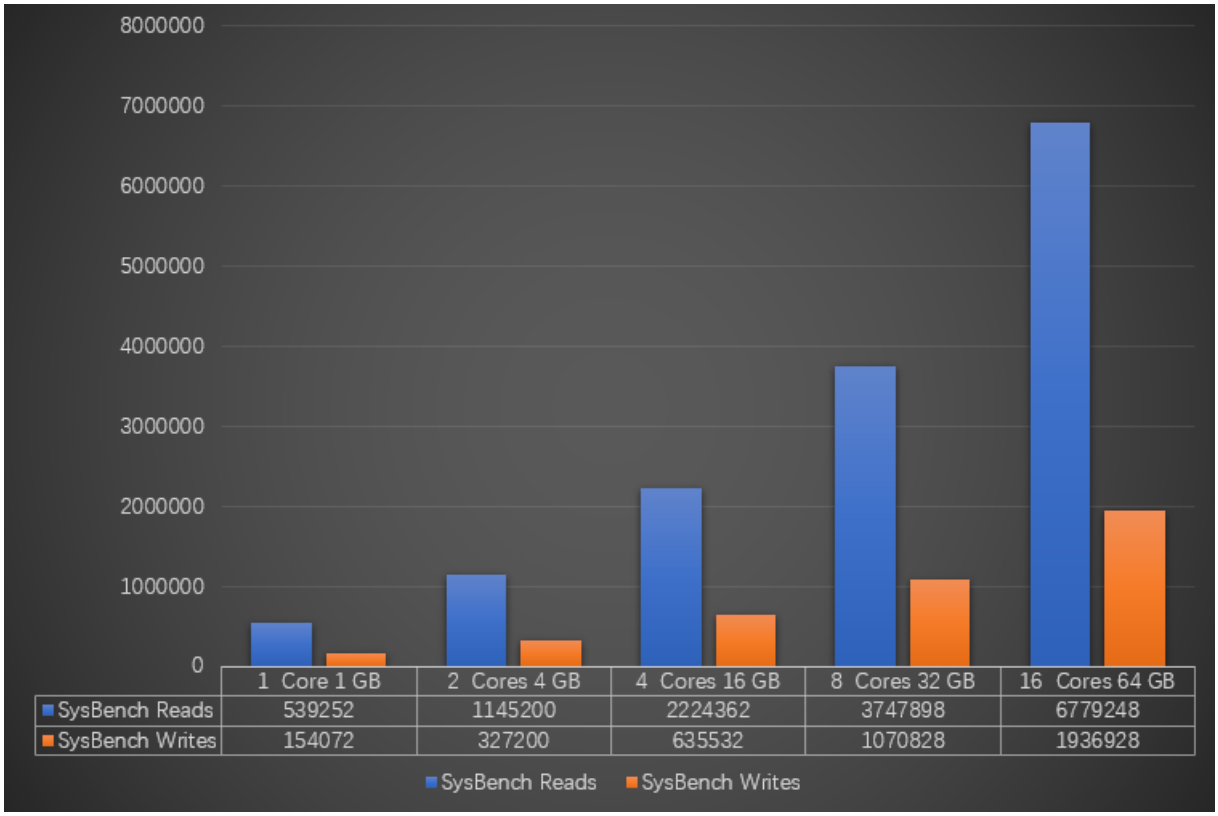
    > **Note** These parameter settings are contained in the standard parameter template for ApsaraDB RDS for MySQL. If the standard parameter template is applied to all the RDS instances, data consistency is maximized, and the test environment is more similar to the production environment.

Results

Two types of queries can be used to perform stress testing. You can determine the type of stress testing based on your data volume.

- Stress testing for cache-based queries: This type of stress testing is suitable for scenarios that involve a small amount of data. You can store all data in the InnoDB buffer pool for access. For more information about how to change the size of the InnoDB buffer pool, see Change the size of the InnoDB buffer pool for an ApsaraDB RDS for MySQL instance.
- Stress testing for disk I/O-based queries: This type of stress testing is suitable for scenarios that involve a large amount of data. You can store only the most frequently used data in the InnoDB buffer pool for access. During the stress testing, data is read from or written to disks, and the InnoDB buffer pool is updated.

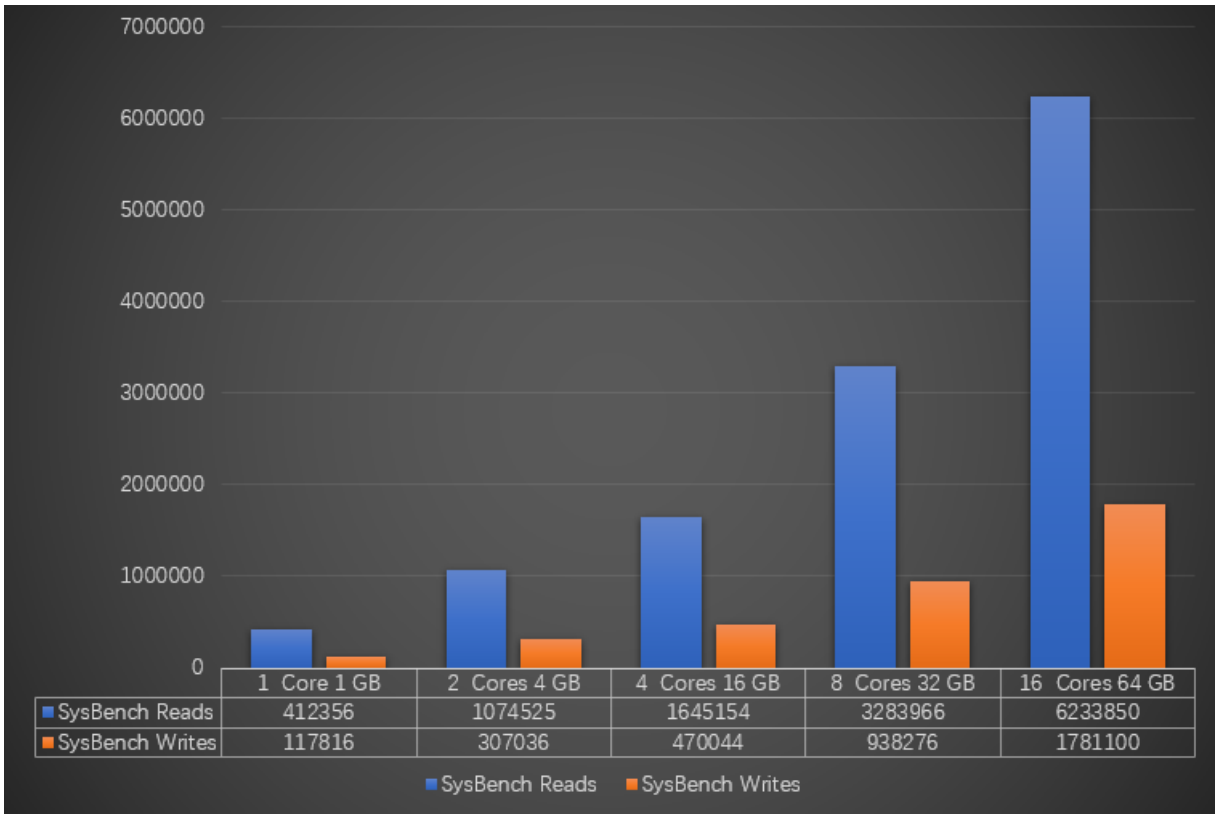
Type 1: Stress testing for cache-based queries



Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
1 core and 1 GB of memory (rds.mysql.t1.small)	25,000	32	300	600	8	539252	154072
2 cores and 4 GB of memory (rds.mysql.s2.large)	25,000	32	1,200	2,000	8	1145200	327200
4 cores and 16 GB of memory (rds.mysql.m1.medium)	25,000	128	4,000	7,000	16	2224362	635532

Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
8 cores and 32 GB of memory (rds.mysql.c1.xlarge)	25,000	128	8,000	12,000	32	3747898	1070828
16 cores and 64 GB of memory (rds.mysql.c2.xlarge)	25,000	128	16,000	14,000	64	6779248	1936928

Type 2: Stress testing for disk I/O-based queries



Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
1 core and 1 GB of memory (rds.mysql.t1.small)	80,000	32	300	600	8	412356	117816
2 cores and 4 GB of memory (rds.mysql.s2.large)	80,000	32	1,200	2,000	8	1074525	307036
4 cores and 16 GB of memory (rds.mysql.m1.medium)	800,000	128	4,000	7,000	16	1645154	470044
8 cores and 32 GB of memory (rds.mysql.c1.xlarge)	800,000	128	8,000	12,000	32	3283966	938276
16 cores and 64 GB of memory (rds.mysql.c2.xlarge)	800,000	128	16,000	14,000	64	6233850	1781100

1.6. Test results of ApsaraDB RDS instances that run MySQL 5.6

This topic describes the performance test results of general-purpose ApsaraDB RDS instances that run MySQL 5.6.

> **Note**
>
> - To better simulate the production environment, the Reads/Writes metric is used for this stress testing.
> - The performance test results are for reference only. For more information about how to use an ApsaraDB RDS instance that runs MySQL 5.6, see Troubleshoot slow SQL statements on an ApsaraDB RDS for MySQL instance.

Environment

In this stress testing, SysBench is used to test the performance of the five RDS instances that use local SSDs. These RDS instances use different instance types.

- Instance types: rds.mysql.t1.small, rds.mysql.s2.large, rds.mysql.m1.medium, rds.mysql.c1.xlarge, and rds.mysql.c2.xlarge
- Instance family: general-purpose
- Edition: RDS High-availability Edition
- Storage type: local SSD

Configurations

The performance is significantly affected by the **data volume**, **stress testing duration**, and **parameter settings**. The following configurations are made for this stress testing:

- Data volume: The data volume and the number of tables on each RDS instance that you want to test are different. Therefore, the test results for some of the five RDS instances may be similar.
- Stress testing duration: The stress testing duration is 60 seconds for all the RDS instances.
- Parameter settings:
  - `sync_binlog=1` and `innodb_flush_log_at_trx_commit=1`: ensure that the data submitted each time is completely written to disks.
  - `rpl_semi_sync_master_enabled=ON`: enables the semi-synchronous mode for an RDS instance to ensure data consistency between the primary and secondary RDS instances.
  - `Performance_schema=ON`: automatically enables Performance Schema for RDS instances that use instance types with a memory size greater than or equal to 8 GB.

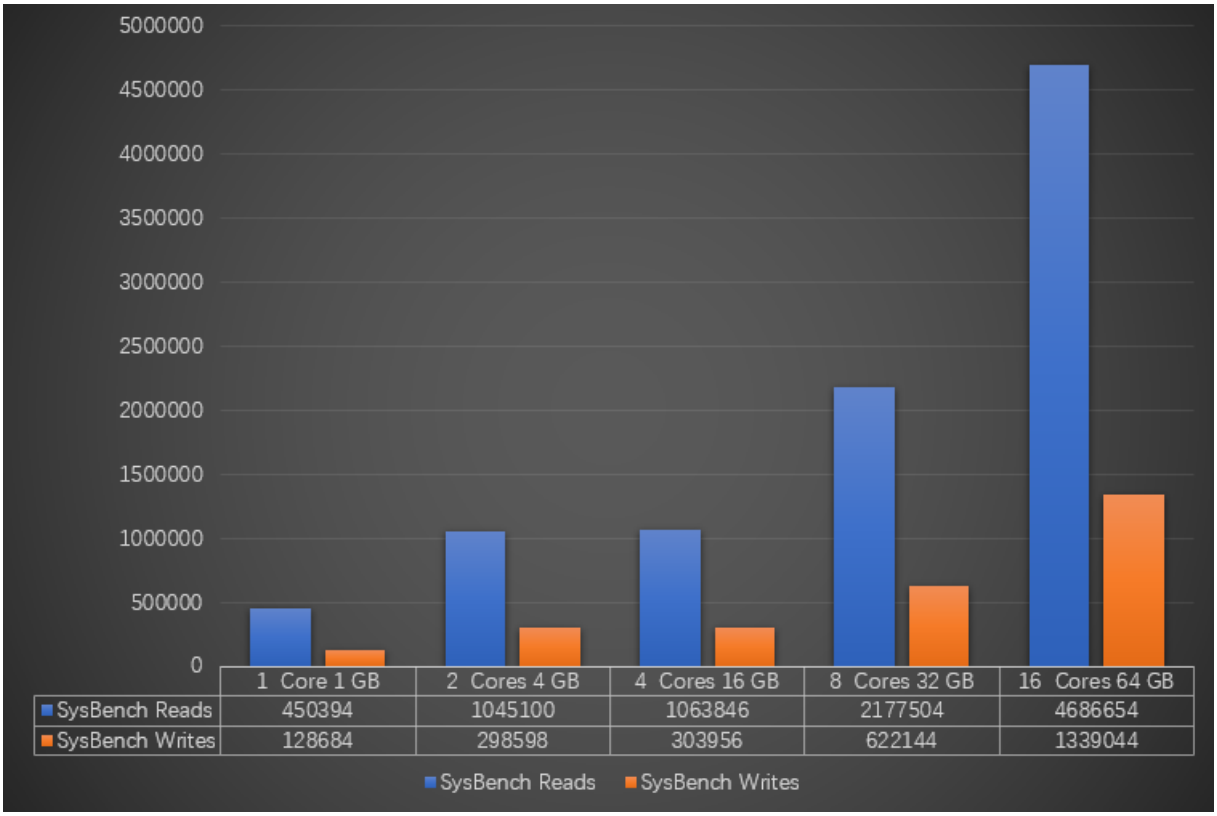
    > **Note** These parameter settings are contained in the standard parameter template for ApsaraDB RDS for MySQL. If the standard parameter template is applied to all the RDS instances, data consistency is maximized, and the test environment is more similar to the production environment.

Results

Two types of queries can be used to perform stress testing. You can determine the type of stress testing based on your data volume.

- Stress testing for cache-based queries: This type of stress testing is suitable for scenarios that involve a small amount of data. You can store all data in the InnoDB buffer pool for access. For more information about how to change the size of the InnoDB buffer pool, see Change the size of the InnoDB buffer pool for an ApsaraDB RDS for MySQL instance.
- Stress testing for disk I/O-based queries: This type of stress testing is suitable for scenarios that involve a large amount of data. You can store only the most frequently used data in the InnoDB buffer pool for access. During the stress testing, data is read from or written to disks, and the InnoDB buffer pool is updated.

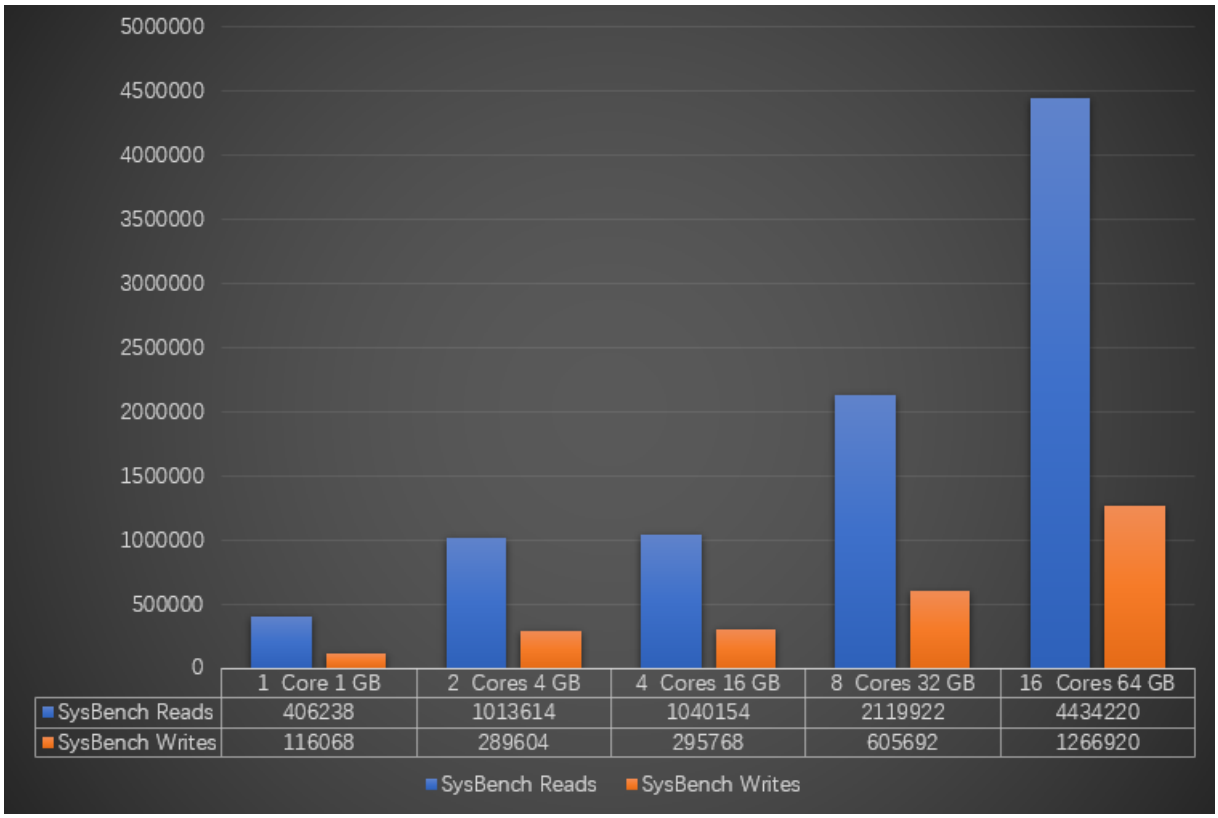
Type 1: Stress testing for cache-based queries



Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
1 core and 1 GB of memory (rds.mysql.t1.small)	25,000	32	300	600	8	450394	128684
2 cores and 4 GB of memory (rds.mysql.s2.large)	25,000	32	1,200	2,000	8	1045100	298598
4 cores and 16 GB of memory (rds.mysql.m1.medium)	25,000	128	4,000	7,000	16	1063846	303956

Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
8 cores and 32 GB of memory (rds.mysql.c1.xlarge)	25,000	128	8,000	12,000	32	2177504	622144
16 cores and 64 GB of memory (rds.mysql.c2.xlarge)	25,000	128	16,000	14,000	64	4686654	1339044

Type 2: Stress testing for disk I/O-based queries



Specifications (instance type)	Data volume in a single table	Number of tables	Maximum number of connections	IOPS	Number of SysBench threads	Number of SysBench reads	Number of SysBench writes
1 core and 1 GB of memory (rds.mysql.t1.small)	80,000	32	300	600	8	406238	116068
2 cores and 4 GB of memory (rds.mysql.s2.large)	80,000	32	1,200	2,000	8	1013614	289604
4 cores and 16 GB of memory (rds.mysql.m1.medium)	800,000	128	4,000	7,000	16	1040154	295768
8 cores and 32 GB of memory (rds.mysql.c1.xlarge)	800,000	128	8,000	12,000	32	2119922	605692
16 cores and 64 GB of memory (rds.mysql.c2.xlarge)	800,000	128	16,000	14,000	64	4434220	1266920

1.7. Test method and results of hot data updates on a single row

This topic describes how to test and analyze the performance of an ApsaraDB RDS for MySQL instance in updating hot data in a single row of a table.

Test environment

In this example, two instances are used. One runs the High-availability Edition, and the other runs the Enterprise Edition. In addition, one uses the rds.mysql.st.v52 instance type, and the other uses the mysql.st.12xlarge.25 instance type.

- Database engine and version: MySQL 5.7
- Specifications: 90 CPU cores, 720 GB of memory (dedicated host instance family)

- RDS editions: High-availability Edition and Enterprise Edition
- Storage type: local SSD

Test data

The table that is used for the test contains 100 rows. The table complies with the following schema:

```
CREATE TABLE `sbtest1`
(
`id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT
,`k` INT(10) UNSIGNED NOT NULL DEFAULT '0'
,`c` CHAR(120) NOT NULL DEFAULT ''
,`pad` CHAR(60) NOT NULL DEFAULT ''
,PRIMARY KEY (`id`)
,KEY `k_1` (`k`)
)
ENGINE=InnoDB AUTO_INCREMENT=101 DEFAULT
CHARSET=utf8 MAX_ROWS=1000000
```

Test script

Execute the following SQL statement to perform concurrent updates on the row whose ID is 100:

```
UPDATE sbtest1 SET k=k+1 WHERE id=100
```

The following Lua script is used for the test:

```
pathtest = string.match(test,"(.*/)")
if pathtest then
   dofile(pathtest .."common.lua")
else
   require("common")
end
function thread_init(thread_id)
   set_vars()
end
function event(thread_id)
   local table_name
   table_name ="sbtest".. sb_rand_uniform(1, oltp_tables_count)
   rs = db_query("begin")
   rs = db_query("update /*+commit_on_success rollback_on_fail target_affect_row(1) */ sbte
st1 SET k=k+1 WHERE id=100")
   rs =db_query("commit")
end
```

Test results

Instance configuration	Maximum TPS on the single row
High-availability Edition	12,000
Enterprise Edition	31,000

Test results for Enterprise Edition

```
OLTP test statistics:
    queries performed:
        read:                            0
        write:                           1865967
        other:                           3731934
        total:                           5597901
    transactions:                        1865967 (30822.67 per sec.)
    read/write requests:                 1865967 (30822.67 per sec.)
    other operations:                    3731934 (61645.34 per sec.)
    ignored errors:                      0      (0.00 per sec.)
    reconnects:                          0      (0.00 per sec.)
```

2.RDS for SQL Server

2.1. ApsaraDB RDS for SQL Server performance overview

ApsaraDB RDS is a stable, reliable, and scalable online database service that is built on top of the Apsara Distributed File System and high-performance SSDs of Alibaba Cloud. ApsaraDB RDS supports the MySQL, SQL Server, PostgreSQL, and MariaDB TX database engines and provides a complete suite of database O&M solutions, such as disaster recovery, backup, restoration, monitoring, and migration. These solutions relieves the need for tedious O&M workloads.

All parameters that are used in ApsaraDB RDS have been tested and optimized over years of production practices that are conducted by a team of experienced database administrators (DBAs) from Alibaba Cloud. These DBAs have continued to optimize each ApsaraDB RDS instance throughout the lifecycle of the instance to ensure that the instance runs at its optimal configuration. In addition, all server hardware that is used by ApsaraDB RDS has passed the tests of multiple concerned parties. This ensures that ApsaraDB RDS can deliver optimal performance and high stability.

2.2. Test results

2.2.1. Test results of SQL Server 2008 R2 on RDS High-availability Edition

This topic describes the results of a performance test on an ApsaraDB RDS instance that runs SQL Server 2008 R2 on RDS High-availability Edition.

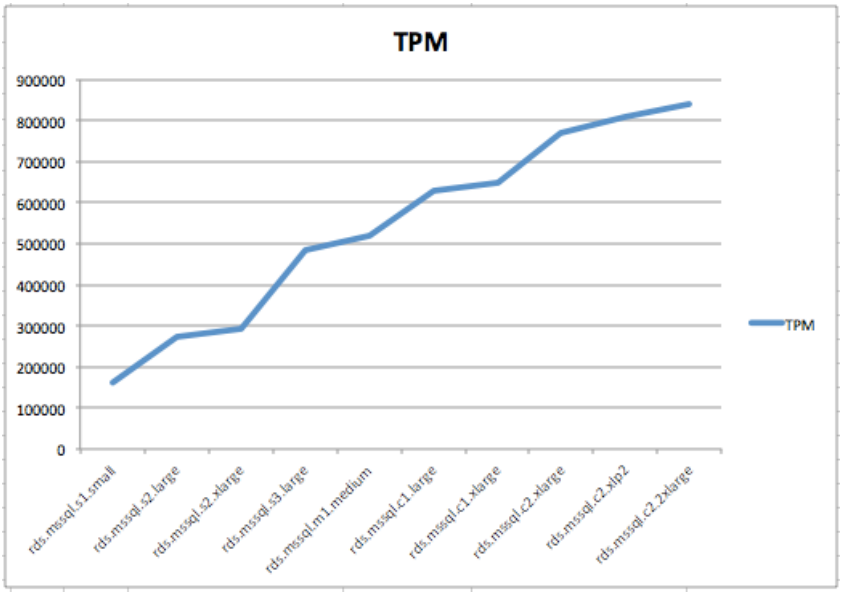
> **Note** The test is based on the TPC-C benchmarks. However, the test is not an official TPC-C benchmark test. Therefore, the test results that are provided in this topic cannot be compared with the test results of the official TPC-C benchmarks.

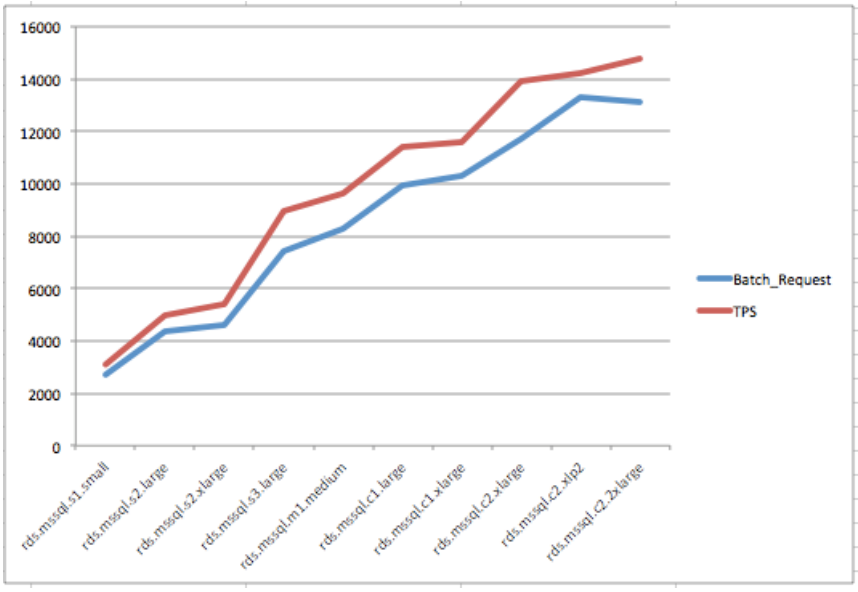
Tested instance types

Instance type	Number of cores	Memory (GB)	IOPS
rds.mssql.s1.small	1	2	1,000
rds.mssql.s2.large	2	4	2,000
rds.mssql.s2.xlarge	2	8	4,000
rds.mssql.s3.large	4	8	5,000
rds.mssql.m1.medium	4	16	7,000
rds.mssql.c1.large	8	16	8,000
rds.mssql.c1.xlarge	8	32	12,000

Instance type	Number of cores	Memory (GB)	IOPS
rds.mssql.c2.xlarge	16	64	14,000
rds.mssql.c2.xlp2	16	96	16,000
rds.mssql.c2.2xlarge	16	128	16,000

Test results





No.	Instance type	TPM	Batch Request	TPS
1	rds.mssql.s1.small	162,000	2,680	3,100
2	rds.mssql.s2.large	273,000	4,370	4,980
3	rds.mssql.s2.xlarge	293,000	4,600	5,400

No.	Instance type	TPM	Batch Request	TPS
4	rds.mssql.s3.large	483,000	7,450	8,970
5	rds.mssql.m1.medium	517,800	8,260	9,640
6	rds.mssql.c1.large	630,000	9,950	11,400
7	rds.mssql.c1.xlarge	647,000	10,300	11,600
8	rds.mssql.c2.xlarge	769,000	11,700	13,900
9	rds.mssql.c2.xlp2	810,000	13,300	14,200
10	rds.mssql.c2.2xlarge	840,000	13,100	14,800

2.2.2. Test results of SQL Server 2012 EE

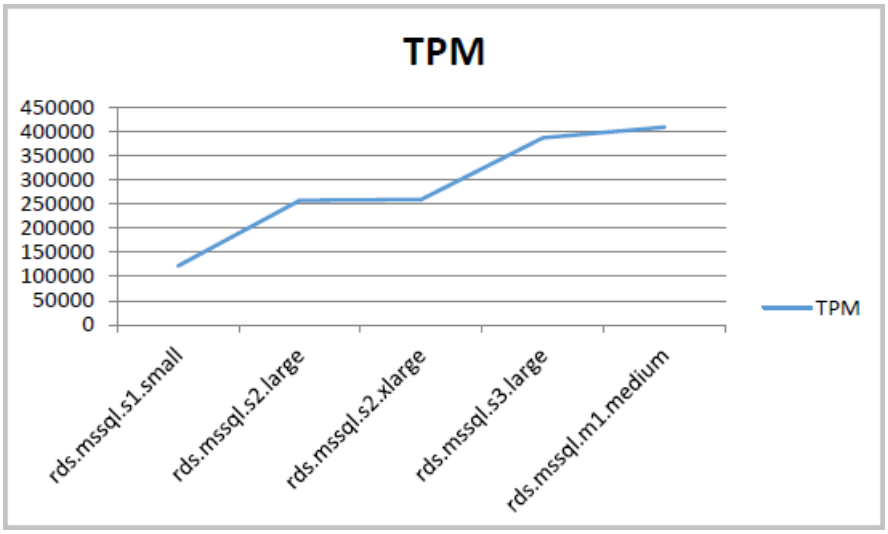
This topic describes the results of a performance test on an ApsaraDB RDS instance that runs SQL Server 2012 EE.

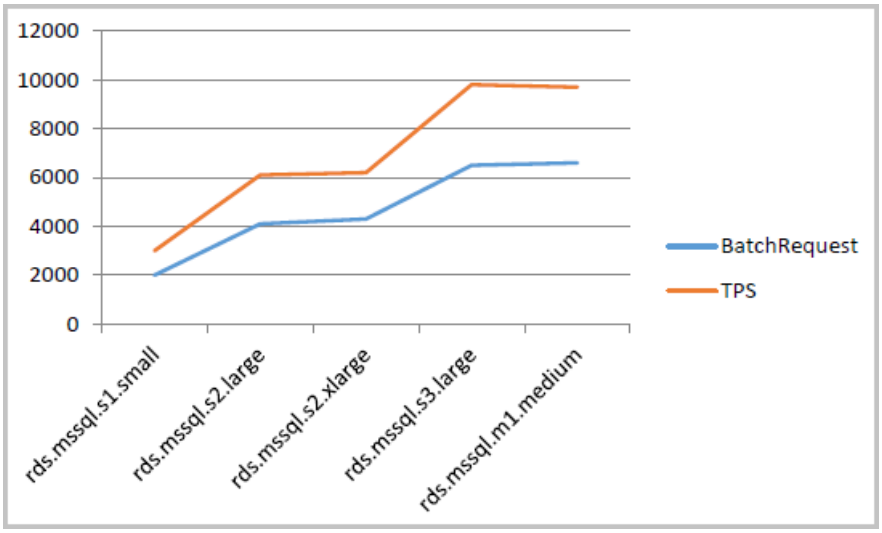
> **Note**
>
> - The test is based on TPC-C benchmarks. However, the test is not an official TPC-C benchmark test. Therefore, the test results that are provided in this topic cannot be compared with the results of TPC-C benchmark tests.
> - For more information about ApsaraDB RDS for SQL Server, see Troubleshoot the issues of high CPU utilization on an ApsaraDB RDS for SQL Server instance.

Tested instance types

Instance type	Number of cores	Memory (GB)
rds.mssql.s1.small	1	2
rds.mssql.s2.large	2	4
rds.mssql.s2.xlarge	2	8
rds.mssql.s3.large	4	8
rds.mssql.m1.medium	4	16

Test results





No.	Instance type	TPM	Batch_Request	TPS
1	rds.mssql.s1.small	122000	2000	3000
2	rds.mssql.s2.large	258000	4100	6100
3	rds.mssql.s2.xlarge	260000	4300	6200
4	rds.mssql.s3.large	388000	6500	9800
5	rds.mssql.m1.medium	410000	6600	9700

3.RDS for PostgreSQL

3.1. ApsaraDB RDS for PostgreSQL performance overview

ApsaraDB RDS is a stable, reliable, and scalable online database service that is built on top of the Apsara Distributed File System and high-performance SSDs of Alibaba Cloud. ApsaraDB RDS supports the MySQL, SQL Server, PostgreSQL, and MariaDB TX database engines and provides a complete suite of database O&M solutions, such as disaster recovery, backup, restoration, monitoring, and migration. These solutions relieves the need for tedious O&M workloads.

All parameters that are used in ApsaraDB RDS have been tested and optimized over years of production practices that are conducted by a team of experienced database administrators (DBAs) from Alibaba Cloud. These DBAs have continued to optimize each ApsaraDB RDS instance throughout the lifecycle of the instance to ensure that the instance runs at its optimal configuration. In addition, all server hardware that is used by ApsaraDB RDS has passed the tests of multiple concerned parties. This ensures that ApsaraDB RDS can deliver optimal performance and high stability.

For more information about the performance test results of ApsaraDB RDS for PostgreSQL, see Test results.

3.2. Test method

3.2.1. Test environment

- Region and zone: All test activities must be conducted in the China (Beijing) region with your ApsaraDB RDS for PostgreSQL instance and ECS instance located in the same zone.
- ECS instance type: ecs.g5.16xlarge (64 vCPUs, 256 GiB).
- ECS storage capacity: 200 GiB provided by local SSDs.
- Network type: VPC.
- Operating system: 64-bit CentOS 7.6.

3.2.2. Test tool

pgbench is a lightweight stress testing tool provided by PostgreSQL for running benchmark tests on PostgreSQL. It executes repetitions of the same sequence of SQL statements, sometimes in multiple concurrent database sessions.

Install pgbench

1. Run the following commands to install PostgreSQL 11 on your ECS instance:

```
yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
yum install -y https://download.postgresql.org/pub/repos/yum/11/redhat/rhel-7-x86_64/pgdg-centos11-11-2.noarch.rpm
yum install -y postgresql11*
su - postgres
vi .bash_profile
export PS1="$USER@`/bin/hostname -s`-> "
export LANG=en_US.utf8
export PGHOME=/usr/pgsql-11
export LD_LIBRARY_PATH=$PGHOME/lib:/lib64:/usr/lib64:/usr/local/lib64:/lib:/usr/lib:/usr/local/lib:$LD_LIBRARY_PATH
export DATE=`date +"%Y%m%d%H%M"`
export PATH=$PGHOME/bin:$PATH:.
export MANPATH=$PGHOME/share/man:$MANPATH
alias rm='rm -i'
alias ll='ls -lh'
unalias vi
```

3.2.3. Test metrics

Read-only QPS

Indicates the number of SELECT statements executed for read-only operations per second.

Read and write QPS

Indicates the number of INSERT, SELECT, and UPDATE statements executed for read/write operations per second.

3.2.4. Test procedure

Describes the steps to test the performance of RDS for PostgreSQL instance.

1. Initialize the test data based on the data size of the target database.
   - To initialize 5 billion data records, run the `pgbench -i -s 50000` command.
   - To initialize 1 billion data records, run the `pgbench -i -s 10000` command.
   - To initialize 500 million data records, run the `pgbench -i -s 5000` command.
   - To initialize 100 million data records, run the `pgbench -i -s 1000` command.
2. Run the following commands to configure environment variables:

```
export PGHOST=<The internal endpoint of your primary ApsaraDB RDS for PostgreSQL instance>
export PGPORT=<The internal port number of your primary ApsaraDB RDS for PostgreSQL instance>
export PGDATABASE=postgres
export PGUSER=<The username to log on to the target database>
export PGPASSWORD=<The password to log on to the target database>
```

3. Create the scripts for testing read-only operations and read/write operations.
   - Create a script file named ro.sql to test read-only operations. The file needs to include the

following code:

```
\set aid random_gaussian(1, :range, 10.0)
SELECT abalance FROM pgbench_accounts WHERE aid = :aid;
```

- Create a script file named rw.sql to test read/write operations. The file needs to include the following code:

```
\set aid random_gaussian(1, :range, 10.0)
\set bid random(1, 1 * :scale)
\set tid random(1, 10 * :scale)
\set delta random(-5000, 5000)
BEGIN;
UPDATE pgbench_accounts SET abalance = abalance + :delta WHERE aid = :aid;
SELECT abalance FROM pgbench_accounts WHERE aid = :aid;
UPDATE pgbench_tellers SET tbalance = tbalance + :delta WHERE tid = :tid;
UPDATE pgbench_branches SET bbalance = bbalance + :delta WHERE bid = :bid;
INSERT INTO pgbench_history (tid, bid, aid, delta, mtime) VALUES (:tid, :bid, :aid, :
delta, CURRENT_TIMESTAMP);
END;
```

4. Test the read-only performance and read/write performance.
   - Run the following commands to test the read-only performance:

```
rds.pg.st.h43 with a total of 5 billion data records including 100 million hot data r
ecords:
pgbench -M prepared -v -r -P 1 -f ./ro.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=100000000
rds.pg.st.h43 with a total of 5 billion data records including 500 million hot data r
ecords:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=500000000
rds.pg.st.h43 with a total of 5 billion data records including 1 billion hot data rec
ords:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=1000000000
rds.pg.st.h43 with a total of 5 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=5000000000
pg.x4.4xlarge.2 with a total of 1 billion data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./ro.sql -c 128 -j 128 -T 120 -D scale=10000 -D ran
ge=100000000
pg.x4.4xlarge.2 with a total of 1 billion data records including 500 million hot data
records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 128 -j 128 -T 120 -D scale=10000 -D ran
ge=500000000
pg.x4.4xlarge.2 with a total of 1 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 128 -j 128 -T 120 -D scale=10000 -D ran
ge=1000000000
pg.x8.2xlarge.2 with a total of 1 billion data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./ro.sql -c 64 -j 64 -T 120 -D scale=10000 -D range
=100000000
pg.x8.2xlarge.2 with a total of 1 billion data records including 500 million hot data
```

```
records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 64 -j 64 -T 120 -D scale=10000 -D range
=500000000
pg.x8.2xlarge.2 with a total of 1 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 64 -j 64 -T 120 -D scale=10000 -D range
=1000000000
pg.x8.xlarge.2 with a total of 1 billion data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./ro.sql -c 32 -j 32 -T 120 -D scale=10000 -D range
=100000000
pg.x8.xlarge.2 with a total of 1 billion data records including 500 million hot data
records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 32 -j 32 -T 120 -D scale=10000 -D range
=500000000
pg.x8.xlarge.2 with a total of 1 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 32 -j 32 -T 120 -D scale=10000 -D range
=1000000000
pg.x8.large.2 with a total of 500 million data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./ro.sql -c 16 -j 16 -T 120 -D scale=5000 -D range=
100000000
pg.x8.large.2 with a total of 500 million data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 16 -j 16 -T 120 -D scale=5000 -D range=
500000000
pg.x8.medium.2 with a total of 100 million data records including 50 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./ro.sql -c 8 -j 8 -T 120 -D scale=1000 -D range=50
000000
pg.x8.medium.2 with a total of 100 million data records that are all hot data records
:
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 8 -j 8 -T 120 -D scale=1000 -D range=10
0000000
```

- Run the following commands to test the read/write performance:

```
rds.pg.st.h43 with a total of 5 billion data records including 100 million hot data r
ecords:
pgbench -M prepared -v -r -P 1 -f ./rw.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=100000000
rds.pg.st.h43 with a total of 5 billion data records including 500 million hot data r
ecords:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=500000000
rds.pg.st.h43 with a total of 5 billion data records including 1 billion hot data rec
ords:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=1000000000
rds.pg.st.h43 with a total of 5 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 240 -j 240 -T 120 -D scale=50000 -D ran
ge=5000000000
pg.x4.4xlarge.2 with a total of 1 billion data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./rw.sql -c 128 -j 128 -T 120 -D scale=10000 -D ran
ge=100000000
```

```
pg.x4.4xlarge.2 with a total of 1 billion data records including 500 million hot data
records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 128 -j 128 -T 120 -D scale=10000 -D ran
ge=500000000
pg.x4.4xlarge.2 with a total of 1 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 128 -j 128 -T 120 -D scale=10000 -D ran
ge=1000000000
pg.x8.2xlarge.2 with a total of 1 billion data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./rw.sql -c 64 -j 64 -T 120 -D scale=10000 -D range
=100000000
pg.x8.2xlarge.2 with a total of 1 billion data records including 500 million hot data
records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 64 -j 64 -T 120 -D scale=10000 -D range
=500000000
pg.x8.2xlarge.2 with a total of 1 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 64 -j 64 -T 120 -D scale=10000 -D range
=1000000000
pg.x8.xlarge.2 with a total of 1 billion data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./rw.sql -c 32 -j 32 -T 120 -D scale=10000 -D range
=100000000
pg.x8.xlarge.2 with a total of 1 billion data records including 500 million hot data
records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 32 -j 32 -T 120 -D scale=10000 -D range
=500000000
pg.x8.xlarge.2 with a total of 1 billion data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 32 -j 32 -T 120 -D scale=10000 -D range
=1000000000
pg.x8.large.2 with a total of 500 million data records including 100 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./rw.sql -c 16 -j 16 -T 120 -D scale=5000 -D range=
100000000
pg.x8.large.2 with a total of 500 million data records that are all hot data records:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 16 -j 16 -T 120 -D scale=5000 -D range=
500000000
pg.x8.medium.2 with a total of 100 million data records including 50 million hot data
records:
pgbench -M prepared -v -r -P 1 -f ./rw.sql -c 8 -j 8 -T 120 -D scale=1000 -D range=50
000000
pg.x8.medium.2 with a total of 100 million data records that are all hot data records
:
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 8 -j 8 -T 120 -D scale=1000 -D range=10
0000000
```

Note

- The value of scale multiplied by 100,000: indicates the number of test data records.
- range: indicates the number of hot data records.
- -c: indicates the number of connections to be tested. This number does not represent the maximum number of connections supported by this instance type. For more information, see Primary ApsaraDB RDS instance types.

3.3. Test results

Note The performance test results of ApsaraDB RDS for PostgreSQL in the following table are for reference only. For more information about ApsaraDB RDS for PostgreSQL, see Performance optimization and diagnosis and Development and O&M recommendations for ApsaraDB RDS for PostgreSQL.

Instance type	Number of data records that can be stored	Number of tested data records	Number of hot (active) data records	Read-only QPS	Read and write QPS
rds.pg.st.h43 (60 cores, 470 GB of memory, and 3 TB of storage)	15 billion	5 billion	0.1 billion	573651	336850
rds.pg.st.h43 (60 cores, 470 GB of memory, and 3 TB of storage)	15 billion	5 billion	0.5 billion	559255	309410
rds.pg.st.h43 (60 cores, 470 GB of memory, and 3 TB of storage)	15 billion	5 billion	1 billion	550090	284155
rds.pg.st.h43 (60 cores, 470 GB of memory, and 3 TB of storage)	15 billion	5 billion	5 billion	430596	204160
pg.x4.4xlarge.2 (32 cores, 128 GB of memory, and 2 TB of storage)	10 billion	1 billion	0.1 billion	400144	254915
pg.x4.4xlarge.2 (32 cores, 128 GB of memory, and 2 TB of storage)	10 billion	1 billion	0.5 billion	375522	228440
pg.x4.4xlarge.2 (32 cores, 128 GB of memory, and 2 TB of storage)	10 billion	1 billion	1 billion	360007	200250

Instance type	Number of data records that can be stored	Number of tested data records	Number of hot (active) data records	Read-only QPS	Read and write QPS
pg.x8.2xlarge.2 (16 cores, 128 GB of memory, and 2 TB of storage)	10 billion	1 billion	0.1 billion	235592	156330
pg.x8.2xlarge.2 (16 cores, 128 GB of memory, and 2 TB of storage)	10 billion	1 billion	0.5 billion	221153	133125
pg.x8.2xlarge.2 (16 cores, 128 GB of memory, and 2 TB of storage)	10 billion	1 billion	1 billion	211268	115915
pg.x8.xlarge.2 (8 cores, 64 GB of memory, and 1 TB of storage)	5 billion	1 billion	0.1 billion	129323	71820
pg.x8.xlarge.2 (8 cores, 64 GB of memory, and 1 TB of storage)	5 billion	1 billion	0.5 billion	115498	58140
pg.x8.xlarge.2 (8 cores, 64 GB of memory, and 1 TB of storage)	5 billion	1 billion	1 billion	102735	58555
pg.x8.large.2 (4 cores, 32 GB of memory, and 500 GB of storage)	2.5 billion	0.5 billion	0.1 billion	75729	45110
pg.x8.large.2 (4 cores, 32 GB of memory, and 500 GB of storage)	2.5 billion	0.5 billion	0.5 billion	63818	36375

Instance type	Number of data records that can be stored	Number of tested data records	Number of hot (active) data records	Read-only QPS	Read and write QPS
pg.x8.medium.2 (2 cores, 16 GB of memory, and 250 GB of storage)	1.25 billion	0.1 billion	0.05 billion	37245	24520
pg.x8.medium.2 (2 cores, 16 GB of memory, and 250 GB of storage)	1.25 billion	0.1 billion	0.1 billion	35321	19880

Note

- Instance type: indicates the code that represents the specifications of an ApsaraDB RDS for PostgreSQL instance.
- Number of data records that can be stored: indicates the maximum number of data records that can be stored for the instance type.
- Number of tested data records: indicates the number of data records that are used in the test.
- Number of hot (active) data records: indicates the number of data records that are queried and updated by using SQL statements in the test.
- Read-only QPS: indicates the number of read requests that are processed per second.
- Read and write QPS: the number of read requests and write requests that are processed per second.

4.Considerations for performance comparison between a user-created database and an ApsaraDB for RDS instance

This topic describes the issues that you need to consider when you perform a test to compare the performance of a user-created database with that of an ApsaraDB for RDS instance. The user-created database and the ApsaraDB for RDS instance must be compared under the same conditions. These conditions include the network environment, specifications, and database engine version.

In actual business scenarios, you can select your user-created database or purchase an ApsaraDB for RDS instance based on your business requirements. We recommend that you purchase an ApsaraDB for RDS instance. An ApsaraDB for RDS instance is fully hosted. It includes a complete suite of solutions such as security, backup, restoration, scaling, and performance optimization. This allows you to mitigate the need for security and maintenance workloads. For example, you do not need to create secondary databases. You need to focus only on the development and innovation of your business.

For more information about the comparison between a user-created database and an ApsaraDB for RDS instance, see Competitive advantages of ApsaraDB for RDS instances over user-created databases.

Network environment

- Your application and the user-created database must be deployed on different ECS instances and reside in the same region as the ApsaraDB for RDS instance. This allows you to use the application to communicate with the ApsaraDB for RDS instance and the user-created database over internal networks.

  > **Note** If the application and the user-created database are deployed on the same ECS instance, the network path from the application to the user-created database is shorter than that from the application to the ApsaraDB for RDS instance. In addition, the application occupies CPU resources. These affect the performance of the user-created database. As a result, the test results are compromised and do not show accurate comparison results for the user-created database.

- You can deploy the application, user-created database, and ApsaraDB for RDS instance by using one of the following architectures:
  - The application, user-created database, and ApsaraDB for RDS instance reside in the same zone.
  - The user-created database and the ApsaraDB for RDS instance reside in the same zone, and the application resides in a different zone within the same region.

Specifications

The ECS instance where the user-created database resides provides the same number of CPU cores and memory capacity as the ApsaraDB for RDS instance.

Database engine version

The user-created database and the ApsaraDB for RDS instance are running the same database engine version. For example, they both are running MySQL 5.6.

Data replication mode

Data can be replicated between the primary and secondary user-created databases or between the primary and secondary ApsaraDB for RDS instances in one of the following modes: asynchronous, semi-synchronous, and synchronous. For more information about the data replication modes, see Modify the data replication mode.

The following table lists the data replication modes that are supported by the user-created database and the ApsaraDB for RDS instance.

User-created database	ApsaraDB for RDS instance
No secondary databases are provided, and data replication is not required.	Your database system is running the High-availability Edition, and data is replicated in asynchronous mode.
One secondary database is provided, and data replication is in asynchronous mode.	Your database system is running the High-availability Edition, and data is replicated in asynchronous mode.
One secondary database is provided, and data is replicated in semi-synchronous mode.	Your database system is running the High-availability Edition, and data is replicated in semi-synchronous mode.
Two secondary databases are provided, and data is replicated in synchronous mode.	Your database system is running the Enterprise Edition, and data is replicated in synchronous mode. Only the synchronous mode is supported in the Enterprise Edition.

Parameter settings

The user-created database and the ApsaraDB for RDS instance must use the same parameter settings.

For more information about reconfiguring the parameters of an ApsaraDB for RDS instance, see Use the console to set parameters.

> **Note** For security purposes, ApsaraDB for RDS does not allow you to reconfigure certain parameters. If the user-created database and the ApsaraDB for RDS instance have different settings for a parameter and ApsaraDB for RDS does not allow you to reconfigure this parameter, you must reconfigure this parameter for the user-created database.

Use case

Scenario: A customer is migrating their business system from a user-created database in their on-premises data center to an ApsaraDB for RDS instance. The time that is required to execute SQL statements on the ApsaraDB for RDS instance is twice the time that is required in the user-created database.

Cause: The user-created database and the ApsaraDB for RDS instance use different parameter settings. For example:

- The user-created database uses the following parameter settings:

join_buffer_size = 128M

read_rnd_buffer_size = 128M

tmp_table_size = 128M

- The ApsaraDB for RDS instance uses the following parameter settings:

  join_buffer_size = 1M

  read_buffer_size = 1M

  tmp_table_size =256K