ALIBABA CLOUD

阿里云

云数据库RDS
性能白皮书

文档版本：20220422

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.MySQL版

1.1. 性能概述

阿里云关系型数据库RDS（Relational Database Service）是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和SSD盘高性能存储，RDS支持MySQL、SQL Server、PostgreSQL和MariaDB TX引擎，并且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案，彻底解决数据库运维的烦恼。

自RDS推出市场以来，阿里云数据库行业专家在实践中针对RDS的所有参数不断优化。RDS使用的所有服务器硬件也经过多方的严格评测，保证产品拥有高稳定性和高可用性。

RDS MySQL的性能测试结果请参见：

- MySQL 8.0测试结果
- MySQL 5.7测试结果
- MySQL 5.6测试结果

1.2. 性能测试指导

新用户首次购买实例，您可以参见本文测试实例性能并提交测试报告。提交测试报告后可以享受续费优惠，如果您的测试报告被评为优秀测试报告，还会赠送精美礼品。

前提条件

- 创建RDS MySQL实例。
- 创建ECS实例。

背景信息

RDS MySQL性能测试活动详情请参见性能测试活动。

实例性能测试的指标包括：

- 每秒执行事务数TPS（Transactions Per Second）

  数据库每秒执行的事务数，以COMMIT成功次数为准。

  - SysBench标准OLTP读写混合场景中一个事务包含18个读写SQL。
  - SysBench标准OLTP只读场景中一个事务包含14个读SQL（10条主键点查询、4条范围查询）。
  - SysBench标准OLTP只写场景中一个事务包含4个写SQL（2条UPDATE、1条DETELE、1条INSERT）。

- 每秒执行请求数QPS（Queries Per Second）

  数据库每秒执行的SQL数，包含INSERT、SELECT、UPDATE、DETELE、COMMIT等。

SysBench参数说明

参数	说明
db-driver	数据库引擎。
mysql-host	RDS实例连接地址。
mysql-port	RDS实例连接端口。

参数	说明
mysql-user	RDS实例账号。
mysql-password	RDS实例账号对应的密码。
mysql-db	RDS实例数据库名。
table_size	测试表大小。
tables	测试表数量。
events	测试请求数量。
time	测试时间。
threads	测试线程数。
percentile	需要统计的百分比，默认值为95%，即请求在95%的情况下的执行时间。
report-interval	表示N秒输出一次测试进度报告，0表示关闭测试进度报告输出，仅输出最终的报告结果。
skip-trx	是否跳过事务。 • 1：跳过 • 0：不跳过

性能测试步骤

本文以CentOS系统为例，演示性能测试的步骤。

1. 登录ECS实例，通过如下步骤安装SysBench。

```
yum install gcc gcc-c++ autoconf automake make libtool mysql-devel git mysql
git clone https://github.com/akopytov/sysbench.git
##从Git中下载Sysbench
cd sysbench
##打开sysbench目录
git checkout 1.0.18
##切换到sysbench 1.0.18版本
./autogen.sh
##运行autogen.sh
./configure --prefix=/usr --mandir=/usr/share/man
make
##编译
make install
```

2. 进行性能测试，包括OLTP读写混合场景压测、OLTP只读场景压测和OLTP只写场景压测。
   - OLTP读写混合场景压测

     参见如下命令进行测试，参数说明请参见SysBench参数说明。

```
##准备数据
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
##运行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
##清理数据
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
```

测试结果示例如下：

- QPS：23869.32
- TPS：1193.47
- 响应时间（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
```

- OLTP只读场景压测

  参见如下命令进行测试，参数说明请参见SysBench参数说明。

```
##准备数据
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
##运行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
##清理数据
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
```

测试结果示例如下：

- QPS：26130.73
- 响应时间（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
```

- OLTP只写场景压测

  参见如下命令进行测试，参数说明请参见SysBench参数说明。

```
##准备数据
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
##运行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
##清理数据
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
```

测试结果示例如下：

- TPS：4255.01
- 响应时间（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. 下载MySQL性能测试报告模板，填写后在性能测试数据提交页提交。

1.3. 测试方法

1.3.1. 测试环境

本文介绍云数据库RDS MySQL性能测试所使用的环境。

- 测试的ECS和RDS实例均在同一地域、同一可用区。所有测试均在华东1（杭州）可用区I完成。
- 测试的ECS和RDS实例网络类型均为专有网络（VPC）且在同一VPC。
- 测试的ECS实例信息如下：
  - 实例规格为ecs.g6.8xlarge。
  - 实例镜像为CentOS 7.4 64位。

> **说明** ECS实例的规格较高是为了保证测试时性能瓶颈不在ECS端。

- 测试的RDS实例信息如下：
  - 存储类型为本地SSD盘。
  - 实例规格为通用型的各类规格。
  - 参数模板为MySQL_InnoDB_高可用版_默认参数模板。

1.3.2. 测试场景

本文介绍云数据库RDS MySQL性能测试的场景信息。

表结构

```
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读写混合场景

SQL类型	比例	SQL语句
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`

SQL类型	比例	SQL语句
index_updates	1	`UPDATE sbtest100 SET k=k+1 WHERE id=?`
non_index_updates	1	`UPDATE sbtest100 SET c=? WHERE id=?`
deletes	1	`DELETE FROM sbtest100 WHERE id=?`
inserts_ignore	1	`INSERT IGNORE INTO sbtest100 (id, k, c, pad) VALUES (?, ?, ?, ?)`

1.3.3. 测试工具

本文介绍RDS MySQL性能测试工具SysBench以及如何在ECS实例上安装SysBench。

SysBench工具介绍

SysBench是一个跨平台且支持多线程的模块化基准测试工具，用于评估系统在运行高负载的数据库时相关核心参数的性能表现。可绕过复杂的数据库基准设置，甚至在没有安装数据库的前提下，快速了解数据库系统的性能。

安装方法

本压测使用SysBench 1.0.20版本，更多信息，请参见SysBench。

1. 在ECS实例执行如下命令安装SysBench。

```
yum install gcc gcc-c++ autoconf automake make libtool bzr mysql-devel git mysql
git clone https://github.com/akopytov/sysbench.git
##从Git中下载SysBench
cd sysbench
##打开SysBench目录
git checkout 1.0.20
##切换到SysBench 1.0.20版本
./autogen.sh
##运行autogen.sh
./configure --prefix=/usr --mandir=/usr/share/man
make
##编译
make install
```

2. 执行如下命令配置SysBench Client，使内核可以使用所有的CPU处理数据包（默认设置为使用2个CPU），同时减少CPU之间的上下文切换。

```
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"
```

> 说明 ffffffff表示使用32个CPU（1个f表示4个CPU）。请根据实际配置修改，例如ECS为8核CPU，则输入ff。

1.3.4. 测试方法

本文介绍RDS MySQL性能测试的方法。

操作步骤

使用SysBench测试RDS实例的读写混合性能。

1. 准备数据。

```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mysql-p
assword=XXX --mysql-db=sbtest --table_size=25000 --tables=250 --events=0 --time=600  ol
tp_read_write prepare
```

2. 运行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. 清理数据。

```
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. 测试指标

本文介绍RDS MySQL性能测试的测试指标。

测试指标

60秒内数据库读取和写入的总次数。

1.4. MySQL 8.0测试结果

本文介绍RDS MySQL 8.0通用型实例的性能测试结果。

> 说明
> - 为了更接近生产环境，本次压测取60秒的读写次数总量（Reads/Writes）作为测试指标。
> - 以下性能测试结果仅供参考。为了帮助您更好地了解和使用RDS MySQL 8.0版本实例，请参见性能优化与诊断。

测试环境

本次压测采用业界标准的SysBench，分别对RDS MySQL的5个本地盘规格进行性能测试。

- 实例规格：rds.mysql.t1.small、rds.mysql.s2.large、rds.mysql.m1.medium、rds.mysql.c1.xlarge、rds.mysql.c2.xlarge
- 实例规格族：通用型
- 实例系列：高可用版
- 实例存储类型：本地盘

测试限制

由于**数据量**、**压测时长**、**参数配置**会大幅影响性能数据，本测试做如下限制：

- 数据量：对不同实例规格配置不同的表个数和表数据量。部分规格看似测试结果相近，其实是整体数据量不同。
- 压测时长：由于不同压测时长对测试结果影响较大，因此本次压测时长统一为60秒。
- 参数配置：
  - `sync_binlog=1`、`innodb_flush_log_at_trx_commit=1`：确保每次提交的数据完整写入磁盘中。
  - `rpl_semi_sync_master_enabled=ON`：开启数据库半同步模式，保证主备库数据的一致性。
  - `Performance_schema=ON`：内存大于等于8 GB的实例规格默认开启Performance Schema。

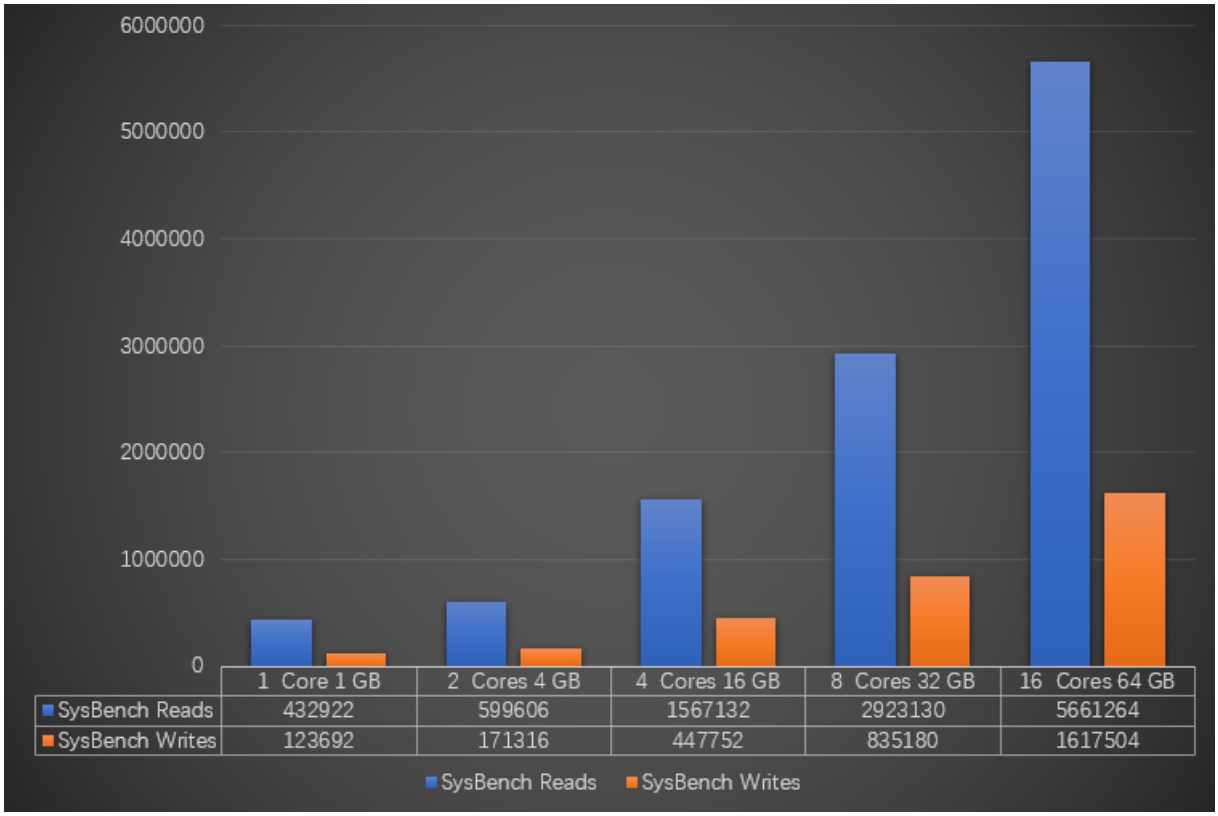
> **说明** 上述参数为RDS MySQL标准参数模板，统一参数模板可在最大程度上保证数据一致性，同时更加接近生产环境。

测试结果

本次压测分两个场景进行，您可以根据自身数据量判断使用哪种场景。

- 内存命中型：适用于数据量较小的场景，可将数据全量放入Buffer Pool进行存取。如何更改Buffer Pool大小，请参见调整实例Buffer Pool大小。
- 磁盘I/O型：适用于数据量大的场景，只将最常访问的数据放入Buffer Pool进行存取，压测时会读写磁盘以及更新Buffer Pool。

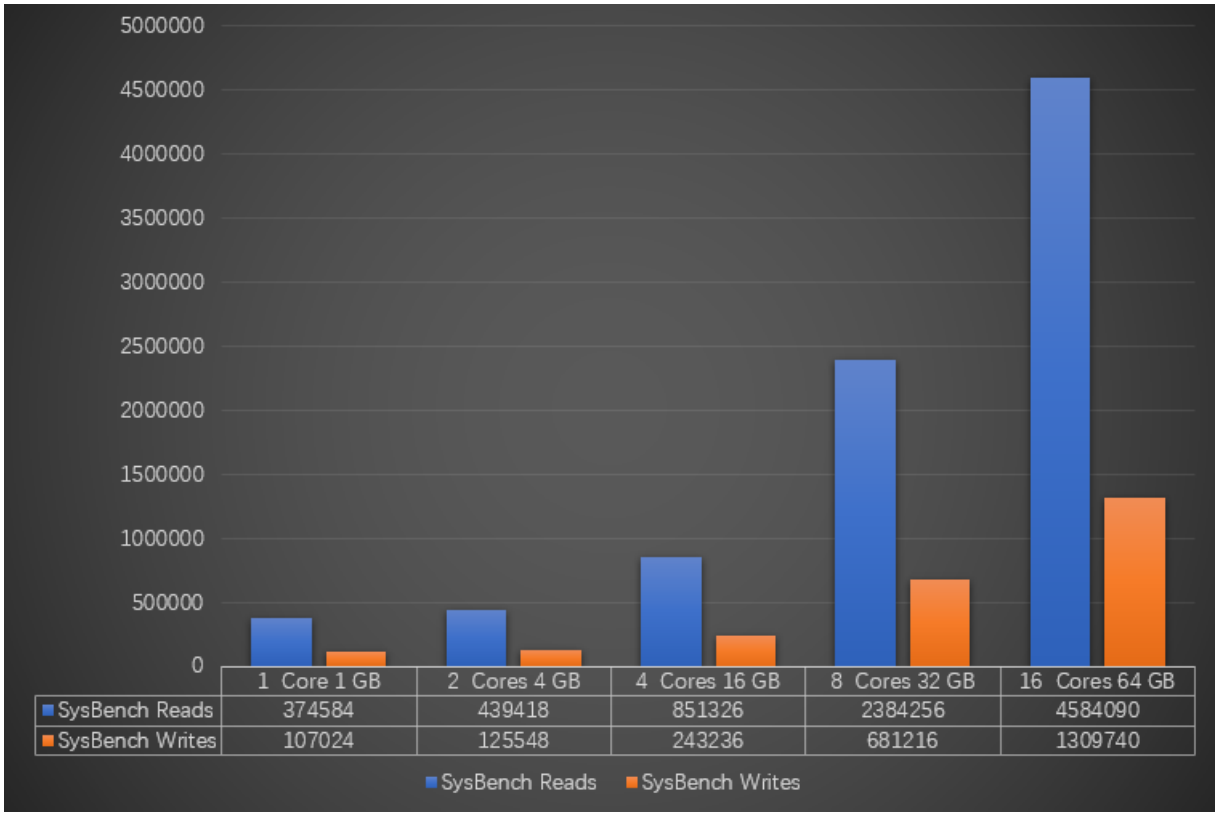
场景一：内存命中型



规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
1核1 GB（rds.mysql.t1.small）	25000	32	300	600	8	432922	123692
2核4 GB（rds.mysql.s2.large）	25000	32	1200	2000	8	599606	171316
4核16 GB（rds.mysql.m1.medium）	25000	128	4000	7000	16	1567132	447752
8核32 GB（rds.mysql.c1.xlarge）	25000	128	8000	12000	32	2923130	835180

规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
16核64 GB（rds.mysql.c2.xlarge）	25000	128	16000	14000	64	5661264	1617504

场景二：磁盘I/O型



规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
1核1 GB（rds.mysql.t1.small）	80000	32	300	600	8	374584	107024
2核4 GB（rds.mysql.s2.large）	80000	32	1200	2000	8	439418	125548

规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
4核16 GB（rds.mysql.m1.medium）	800000	128	4000	7000	16	851326	243236
8核32 GB（rds.mysql.c1.xlarge）	800000	128	8000	12000	32	2384256	681216
16核64 GB（rds.mysql.c2.xlarge）	800000	128	16000	14000	64	4584090	1309740

1.5. MySQL 5.7测试结果

本文介绍RDS MySQL 5.7通用型实例的性能测试结果。

> **说明**
> - 为了更接近生产环境，本次压测取60秒的读写次数总量（Reads/Writes）作为测试指标。
> - 以下性能测试结果仅供参考。为了帮助您更好地了解和使用RDS MySQL 5.7版本实例，请参见性能优化与诊断。

测试环境

本次压测采用业界标准的SysBench，分别对RDS MySQL的5个本地盘规格进行性能测试。

- 实例规格：rds.mysql.t1.small、rds.mysql.s2.large、rds.mysql.m1.medium、rds.mysql.c1.xlarge、rds.mysql.c2.xlarge
- 实例规格族：通用型
- 实例系列：高可用版
- 实例存储类型：本地盘

测试限制

由于**数据量**、**压测时长**、**参数配置**会大幅影响性能数据，本测试做如下限制：

- 数据量：对不同实例规格配置不同的表个数和表数据量。部分规格看似测试结果相近，其实是整体数据量不同。
- 压测时长：由于不同压测时长对测试结果影响较大，因此本次压测时长统一为60秒。
- 参数配置：
  - `sync_binlog=1`、`innodb_flush_log_at_trx_commit=1`：确保每次提交的数据完整写入磁盘中。

- `rpl_semi_sync_master_enabled=ON`：开启数据库半同步模式，保证主备库数据的一致性。
- `Performance_schema=ON`：内存大于等于8 GB的实例规格默认开启Performance Schema。

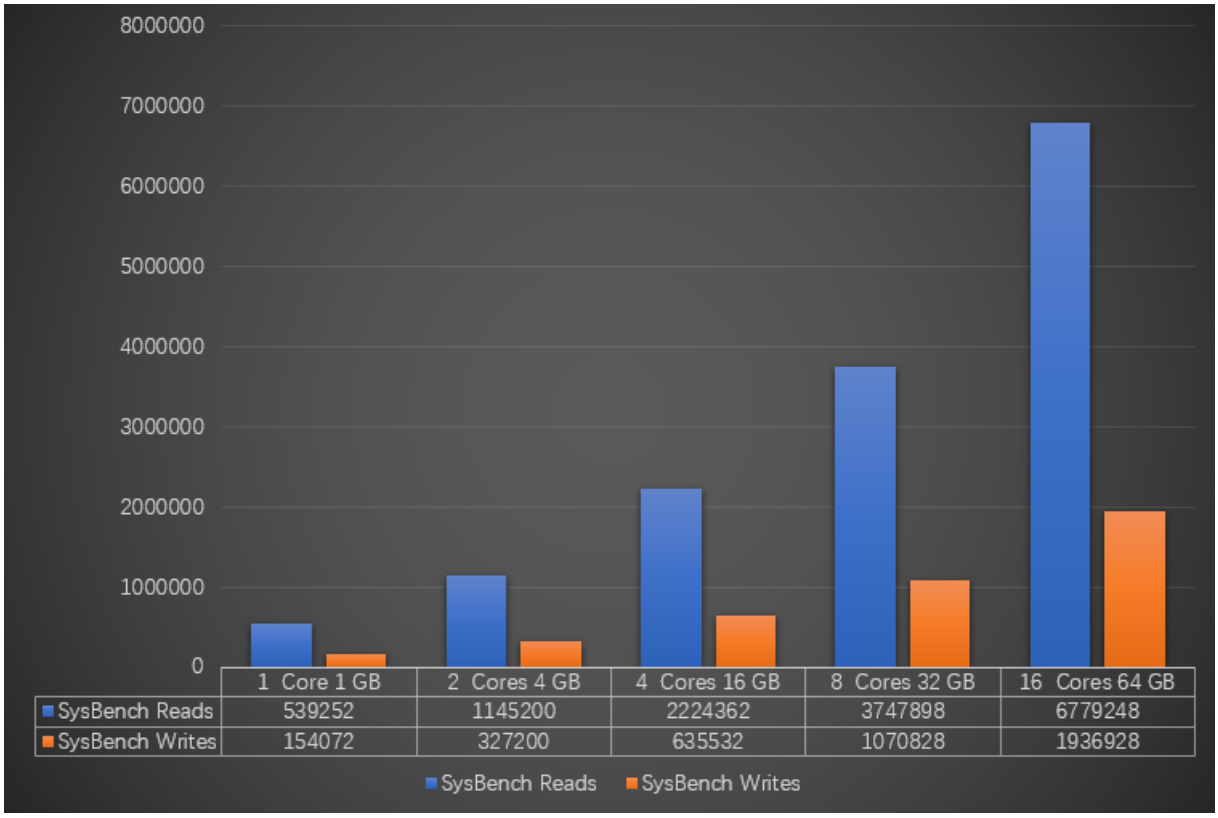
> **说明** 上述参数为RDS MySQL标准参数模板，统一参数模板可在最大程度上保证数据一致性，同时更加接近生产环境。

测试结果

本次压测分两个场景进行，您可以根据自身数据量判断使用哪种场景。

- 内存命中型：适用于数据量较小的场景，可将数据全量放入Buffer Pool进行存取。如何更改Buffer Pool大小，请参见调整实例Buffer Pool大小。
- 磁盘I/O型：适用于数据量大的场景，只将最常访问的数据放入Buffer Pool进行存取，压测时会读写磁盘以及更新Buffer Pool。

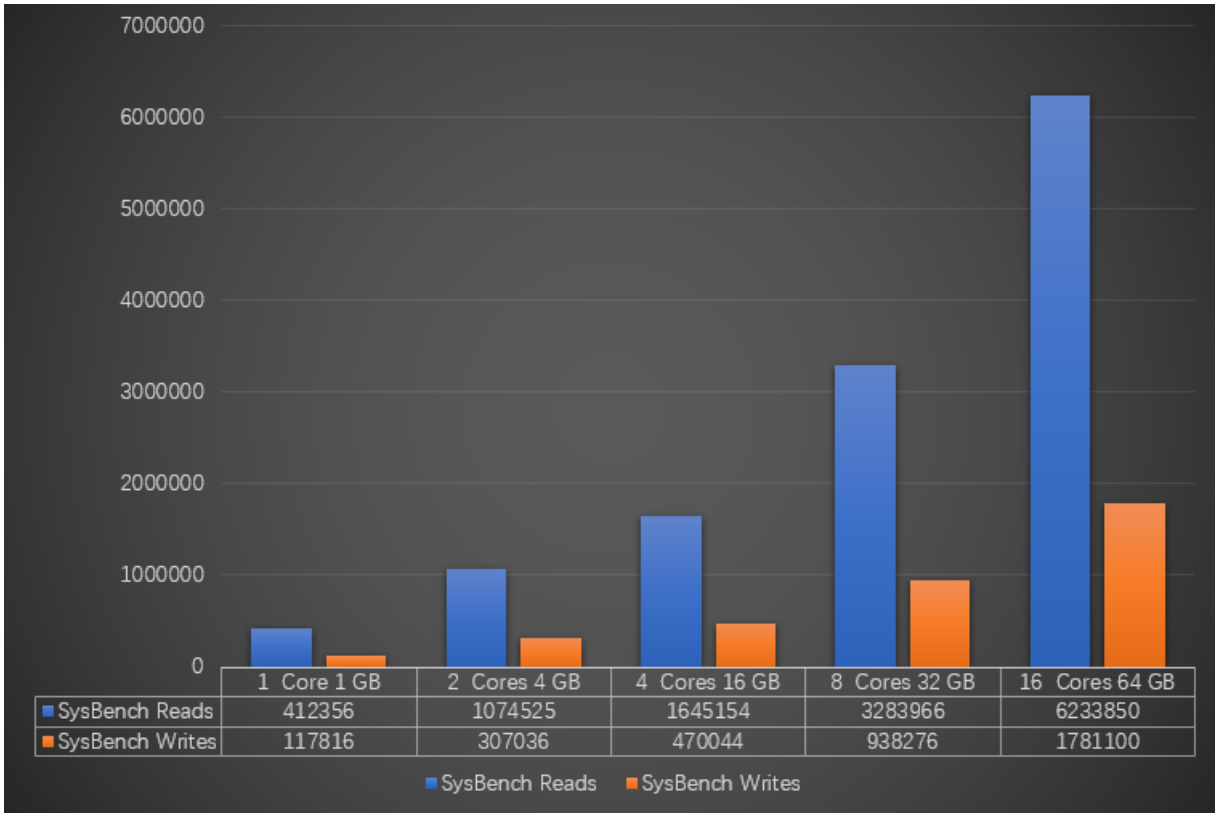
场景一：内存命中型



规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
1核1 GB（rds.mysql.t1.small）	25000	32	300	600	8	539252	154072

规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
2核4 GB（rds.mysql.s2.large）	25000	32	1200	2000	8	1145200	327200
4核16 GB（rds.mysql.m1.medium）	25000	128	4000	7000	16	2224362	635532
8核32 GB（rds.mysql.c1.xlarge）	25000	128	8000	12000	32	3747898	1070828
16核64 GB（rds.mysql.c2.xlarge）	25000	128	16000	14000	64	6779248	1936928

场景二：磁盘I/O型



规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
1核1 GB（rds.mysql.t1.small）	80000	32	300	600	8	412356	117816
2核4 GB（rds.mysql.s2.large）	80000	32	1200	2000	8	1074525	307036
4核16 GB（rds.mysql.m1.medium）	800000	128	4000	7000	16	1645154	470044
8核32 GB（rds.mysql.c1.xlarge）	800000	128	8000	12000	32	3283966	938276
16核64 GB（rds.mysql.c2.xlarge）	800000	128	16000	14000	64	6233850	1781100

1.6. MySQL 5.6测试结果

本文介绍RDS MySQL 5.6通用型实例的性能测试结果。

> **说明**
> - 为了更接近生产环境，本次压测取60秒的读写次数总量（Reads/Writes）作为测试指标。
> - 以下性能测试结果仅供参考。为了帮助您更好地了解和使用RDS MySQL 5.6版本实例，请参见性能优化与诊断。

测试环境

本次压测采用业界标准的SysBench，分别对RDS MySQL的5个本地盘规格进行性能测试。

- 实例规格：rds.mysql.t1.small、rds.mysql.s2.large、rds.mysql.m1.medium、rds.mysql.c1.xlarge、rds.mysql.c2.xlarge
- 实例规格族：通用型
- 实例系列：高可用版
- 实例存储类型：本地盘

测试限制

由于**数据量**、**压测时长**、**参数配置**会大幅影响性能数据，本测试做如下限制：

- 数据量：对不同实例规格配置不同的表个数和表数据量。部分规格看似测试结果相近，其实是整体数据量不同。
- 压测时长：由于不同压测时长对测试结果影响较大，因此本次压测时长统一为60秒。
- 参数配置：
  - `sync_binlog=1`、`innodb_flush_log_at_trx_commit=1`：确保每次提交的数据完整写入磁盘中。
  - `rpl_semi_sync_master_enabled=ON`：开启数据库半同步模式，保证主备数据的一致性。
  - `Performance_schema=ON`：内存大于等于8 GB的实例规格默认开启Performance Schema。

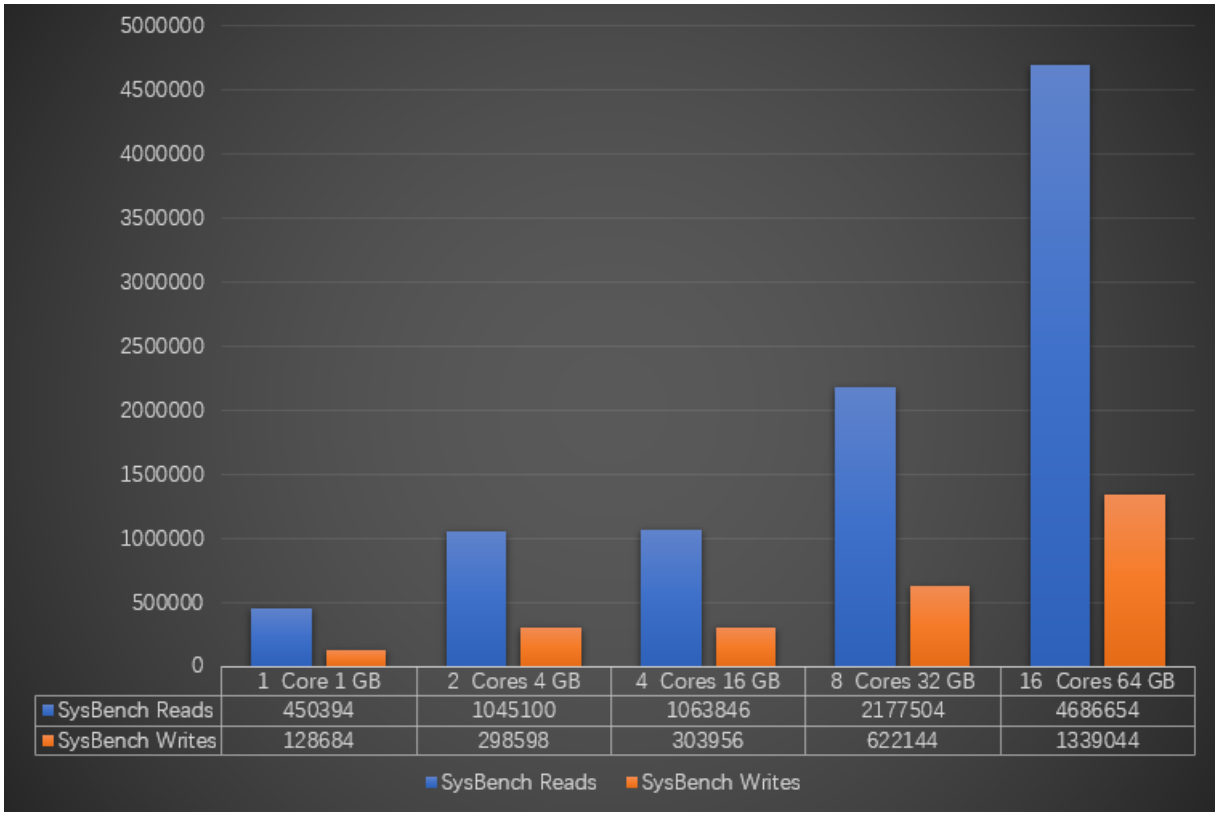
> **说明** 上述参数为RDS MySQL标准参数模板，统一参数模板可在最大程度上保证数据一致性，同时更加接近生产环境。

测试结果

本次压测分两个场景进行，您可以根据自身数据量判断使用哪种场景。

- 内存命中型：适用于数据量较小的场景，可将数据全量放入Buffer Pool进行存取。如何更改Buffer Pool大小，请参见调整实例Buffer Pool大小。
- 磁盘I/O型：适用于数据量大的场景，只将最常访问的数据放入Buffer Pool进行存取，压测时会读写磁盘以及更新Buffer Pool。

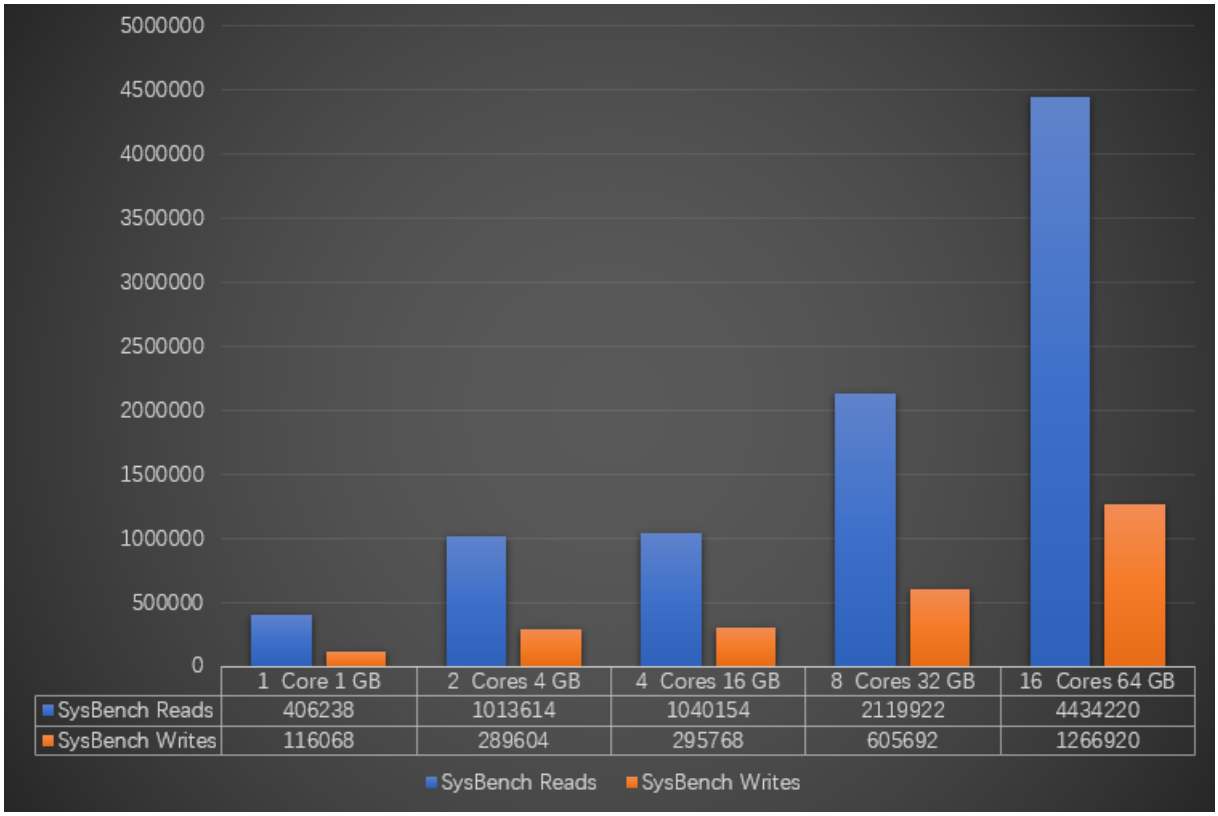
场景一：内存命中型



	1 Core 1 GB	2 Cores 4 GB	4 Cores 16 GB	8 Cores 32 GB	16 Cores 64 GB
SysBench Reads	450394	1045100	1063846	2177504	4686654
SysBench Writes	128684	298598	303956	622144	1339044

规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
1核1 GB（rds.mysql.t1.small）	25000	32	300	600	8	450394	128684
2核4 GB（rds.mysql.s2.large）	25000	32	1200	2000	8	1045100	298598
4核16 GB（rds.mysql.m1.medium）	25000	128	4000	7000	16	1063846	303956
8核32 GB（rds.mysql.c1.xlarge）	25000	128	8000	12000	32	2177504	622144
16核64 GB（rds.mysql.c2.xlarge）	25000	128	16000	14000	64	4686654	1339044

场景二：磁盘I/O型



规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
1核1 GB（rds.mysql.t1.small）	80000	32	300	600	8	406238	116068
2核4 GB（rds.mysql.s2.large）	80000	32	1200	2000	8	1013614	289604
4核16 GB（rds.mysql.m1.medium）	800000	128	4000	7000	16	1040154	295768
8核32 GB（rds.mysql.c1.xlarge）	800000	128	8000	12000	32	2119922	605692

规格	单表数据量	表数量	最大连接数	IOPS	Sysbench线程数	Sysbench读取（单位：次）	SysBench写入（单位：次）
16核64 GB（rds.mysql.c2.xlarge）	800000	128	16000	14000	64	4434220	1266920

1.7. 单行热点更新测试

本文介绍云数据库RDS MySQL单行热点更新的性能测试方法和结果。

测试环境

本示例中，分别使用两个实例进行测试（高可用版和三节点企业版），规格码为rds.mysql.st.v52和mysql.st.12xlarge.25。

- 实例版本：MySQL 5.7
- 实例规格：90核720GB（独占物理机型）
- 实例系列：高可用版和三节点企业版
- 实例存储类型：本地盘

测试数据

测试数据为单表，表内100行记录。表结构如下：

```
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
```

测试脚本

对id=100的记录进行并发更新，SQL如下：

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

测试的Lua脚本如下：

```
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
```

测试结果

实例类型	单行记录更新峰值（TPS）
RDS高可用版	1.2万
RDS三节点企业版	3.1万

三节点企业版测试结果

```
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.SQL Server版

2.1. 性能概述

阿里云关系型数据库RDS（Relational Database Service）是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和SSD盘高性能存储，RDS支持MySQL、SQL Server、PostgreSQL和MariaDB TX引擎，并且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案，彻底解决数据库运维的烦恼。

自RDS推出市场以来，阿里云数据库行业专家在实践中针对RDS的所有参数不断优化。RDS使用的所有服务器硬件也经过多方的严格评测，保证产品拥有高稳定性和高可用性。

2.2. 测试方法

2.2.1. 测试环境

- 所有测试均在华东1（杭州）地域的可用区B完成。
- 测试用的ECS的实例规格族为计算型C1。
- 配置为8核8GB。
- 网络类型为经典网络。
- 压测用的镜像为Win2012 64位标准版。

2.2.2. 测试工具

HammerDB简介

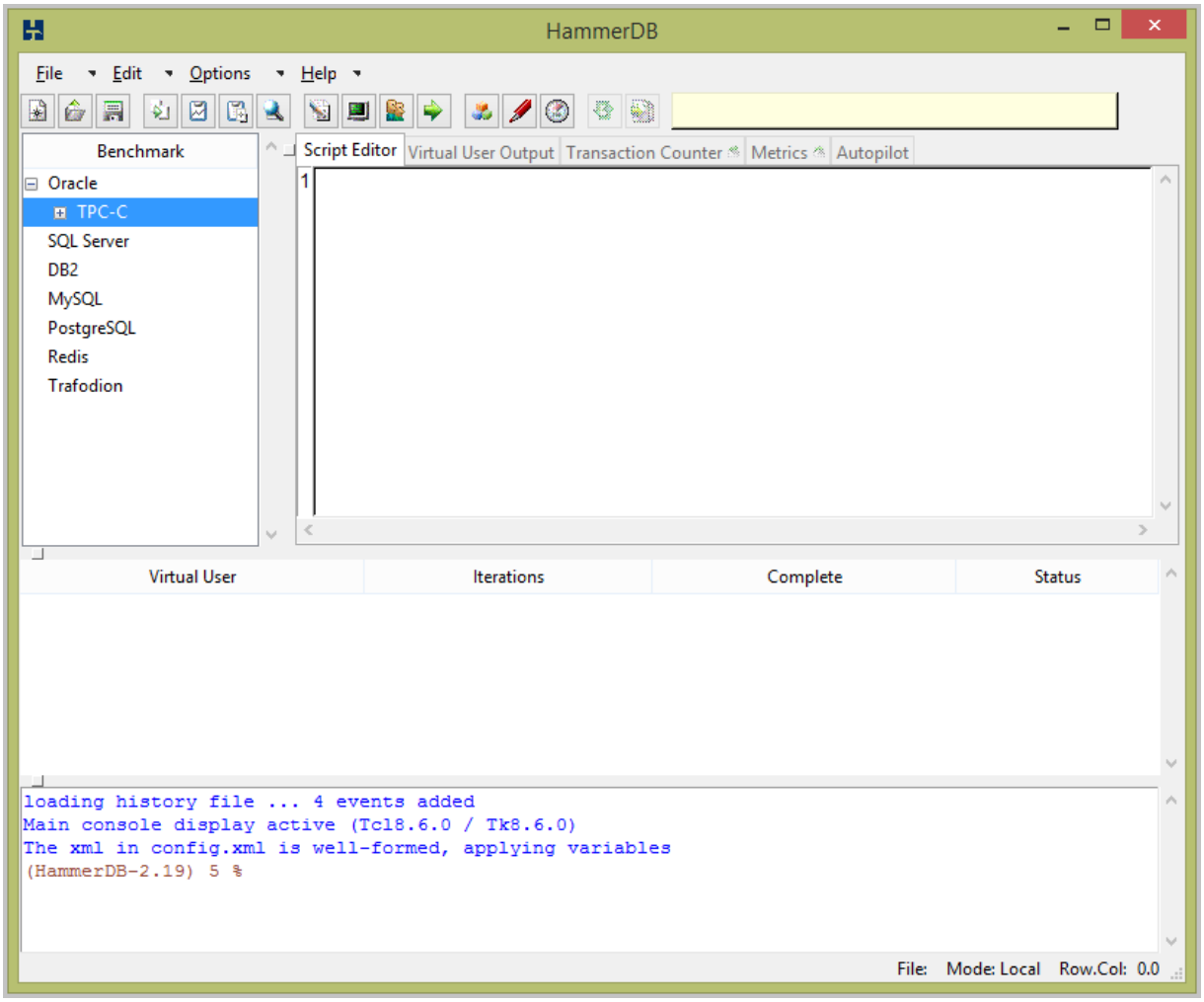
HammerDB是一个开源的数据库负载测试和基准工具，有Linux和Windows版本，可以测试运行在任意系统上的数据库系统。HammerDB具有自动化的，多线程和动态脚本可扩展特点。HammerDB目前支持的数据库种类很多，主流的数据库都已经覆盖，例如Oracle、SQL Server、DB2、TimesTen、MySQL、MariaDB、PostgreSQL、Greenplum、Postgres Plus Advanced Server、Redis 和 Trafodion SQL on Hadoop。HammerDB包含一个内嵌的基于TPC-C工业标准基线工作负载。

安装方法

本文用的HammerDB版本为2.19，单击此处下载。

下载后，请根据安装向导提示安装HammerDB。

安装后界面如下：



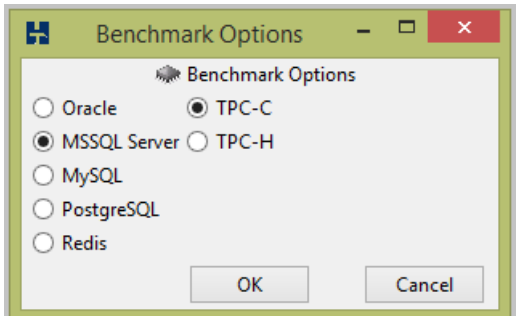
2.2.3. 测试步骤

操作步骤

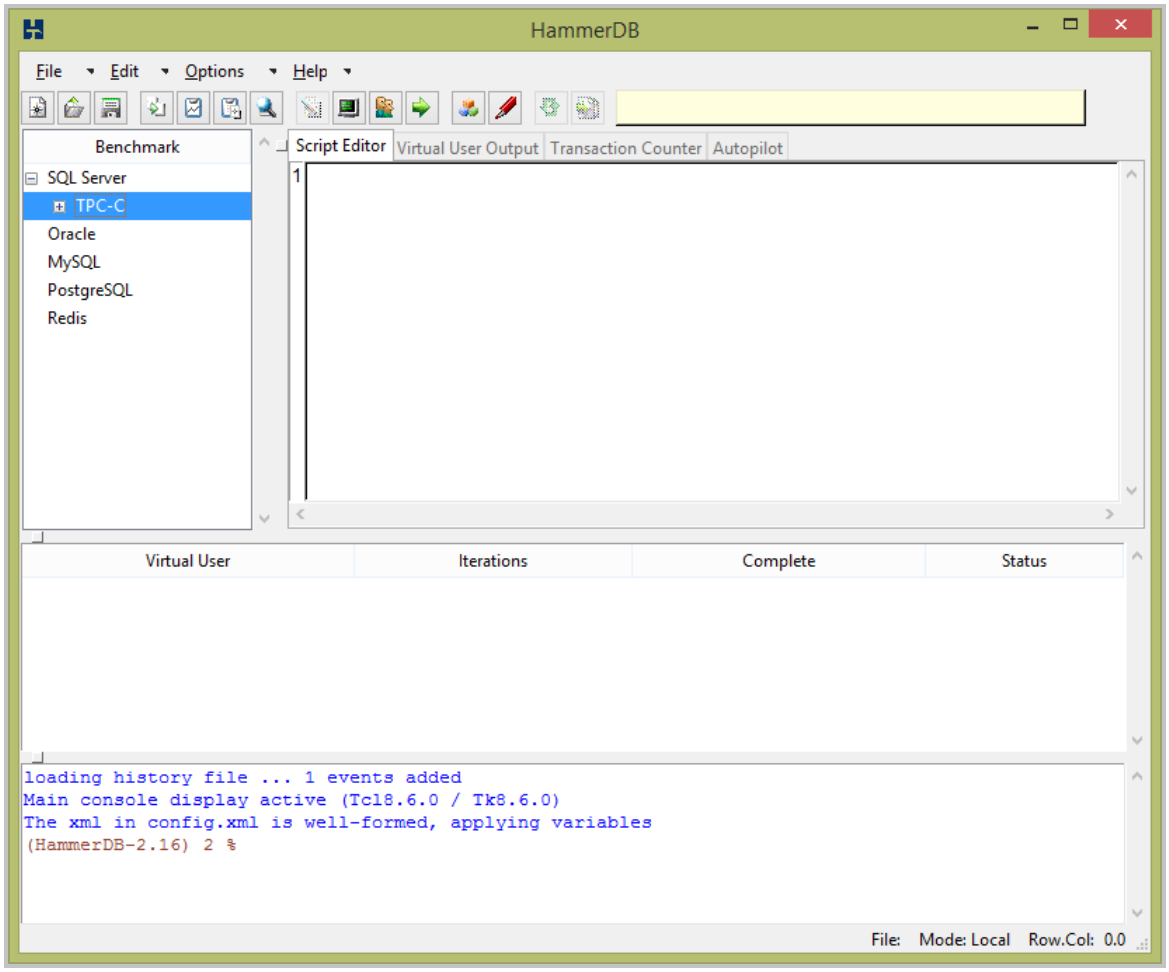
在进行基于TPC-C模式的测试之前，请先完成如下操作步骤：

需要特别说明的是，这个测试建议在前30分钟到2小时之间取值，因为随着数据量的不断变化，TPC-C模式会出现性能瓶颈，后期需要为[dbo].[STOCK]、[dbo].[ORDER_LINE]、[dbo].[ORDERS]这几张表增加Index才可以正常完成测试。

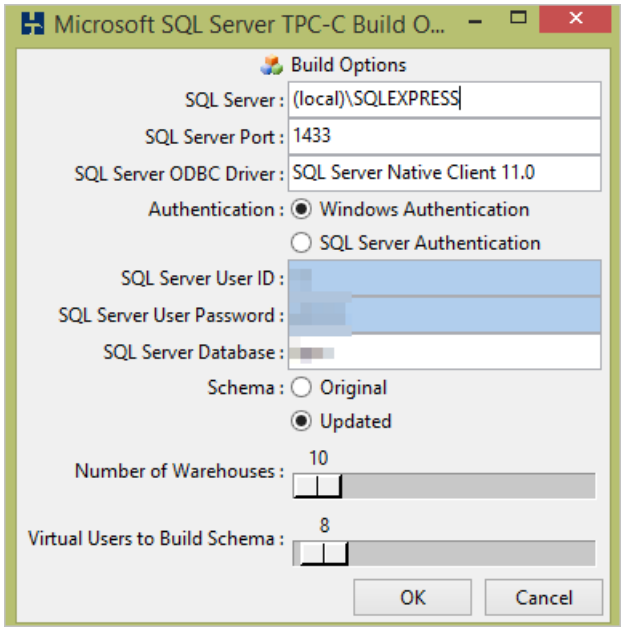
1. 打开**HammerDB**。
2. 选择**SQL SERVER**和**TPC-C**，如下图所示。



3. 准备构建架构，如下图所示。

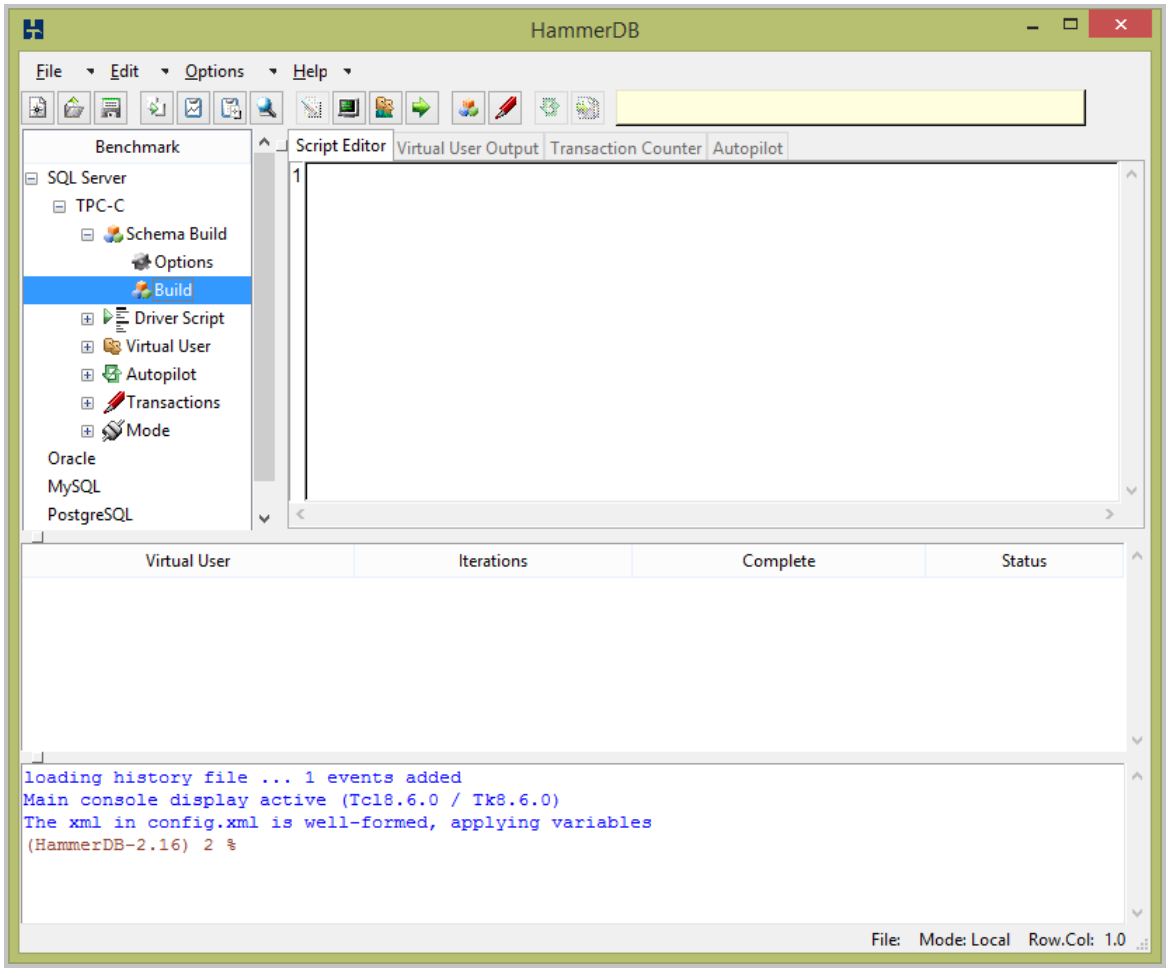


4. 设置连接信息和初始化仓库（所有规格都设置为10）和并发用户数（根据压力调整以测试最佳性能），双击**Schema Build/Option**。如下图所示。

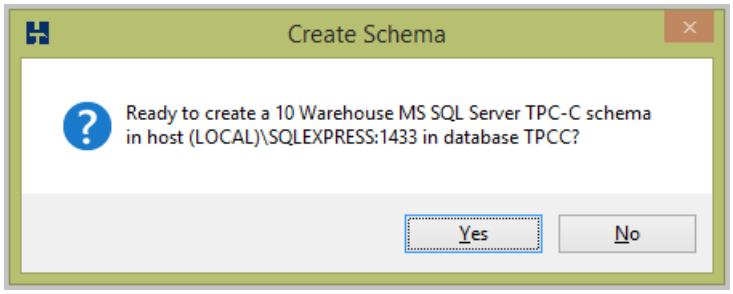


> **说明** 在测试RDS时，虽然有指定端口，还需要在SQL Server上指明端口号，例如：**.sqlserver.rds.aliyuncs.com,3433。

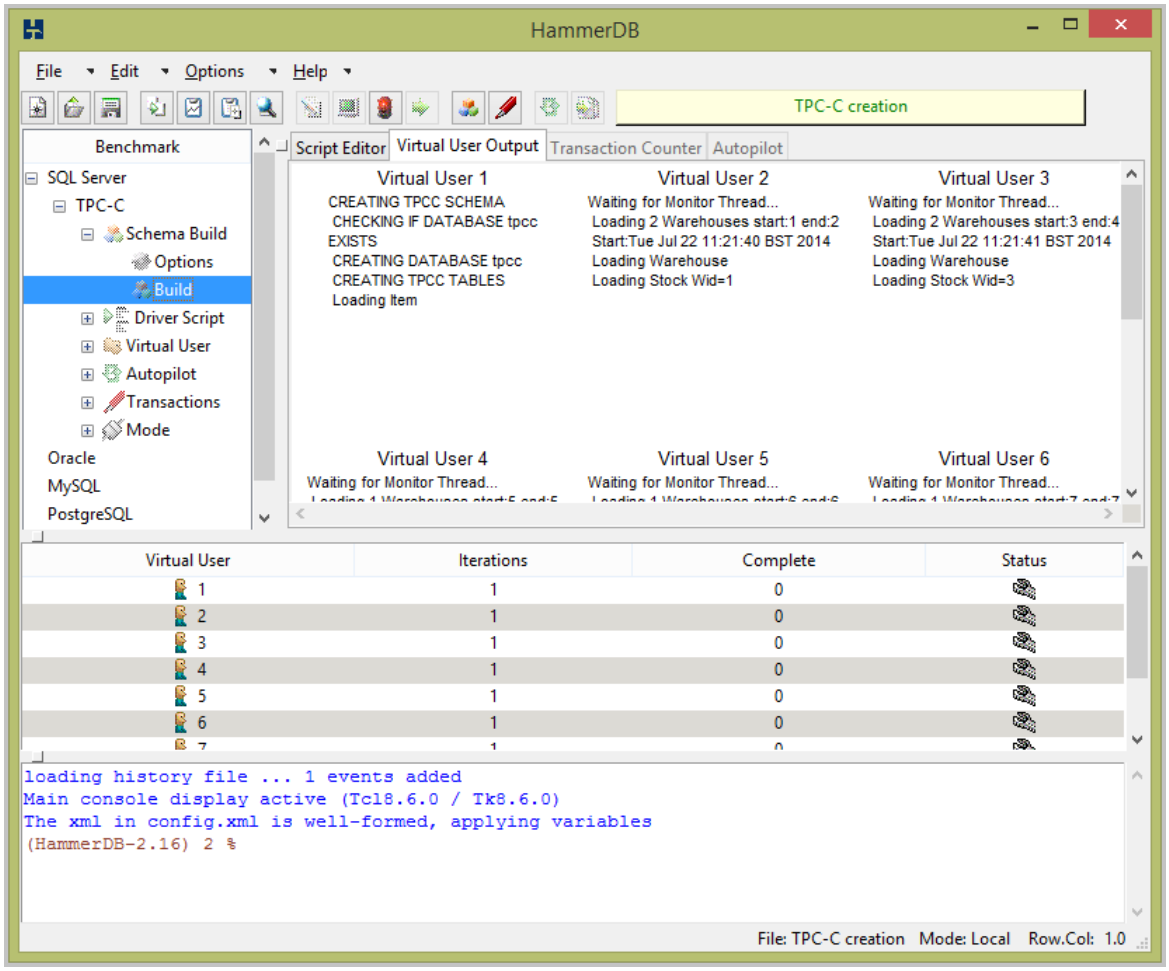
5. 双击**Schema Build/Build**，如下图所示。



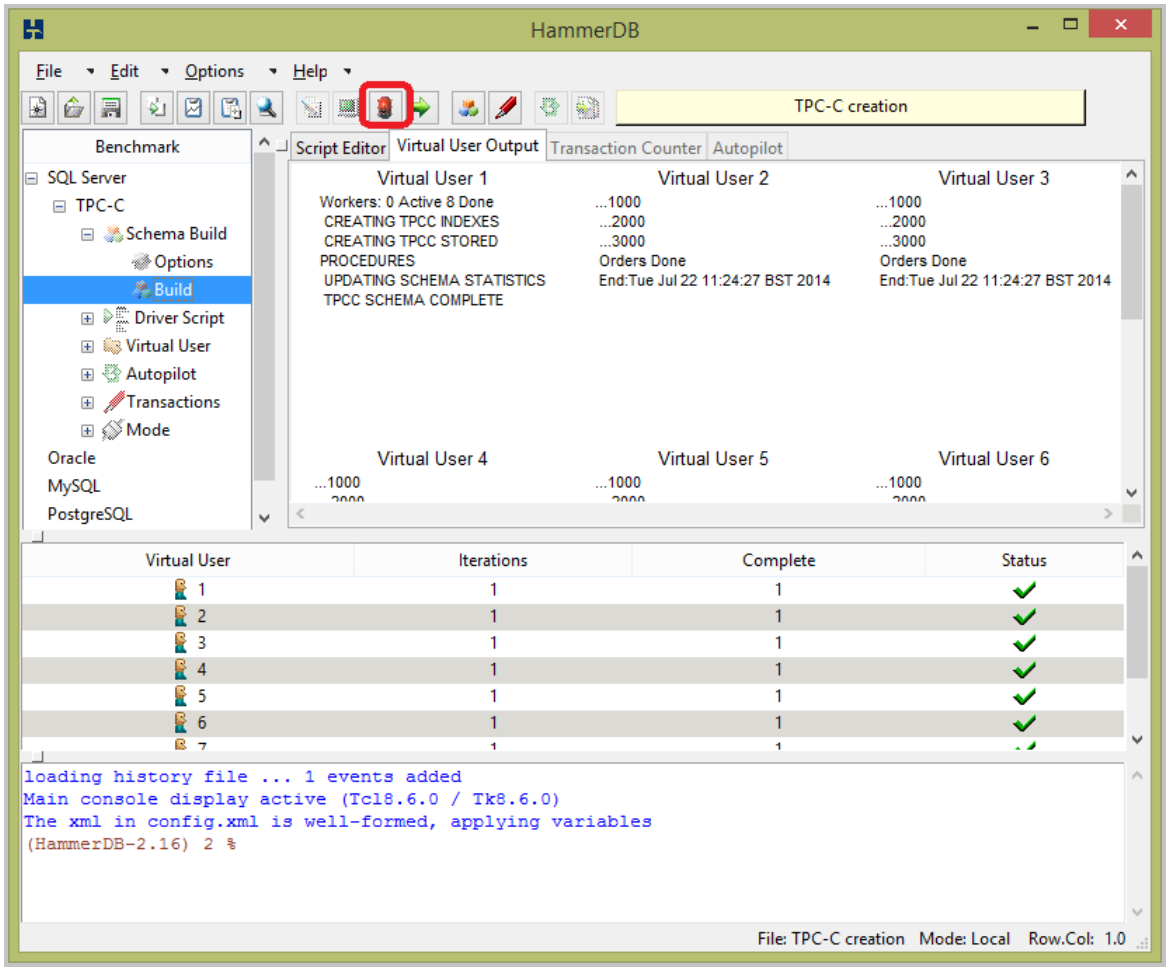
6. 单击**YES**创建架构，如下图所示。



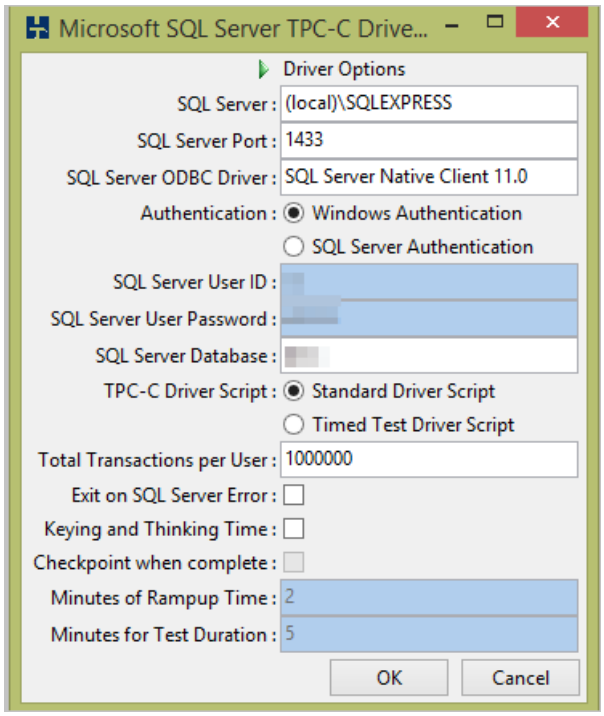
7. 等待初始化架构完成，如下图所示。



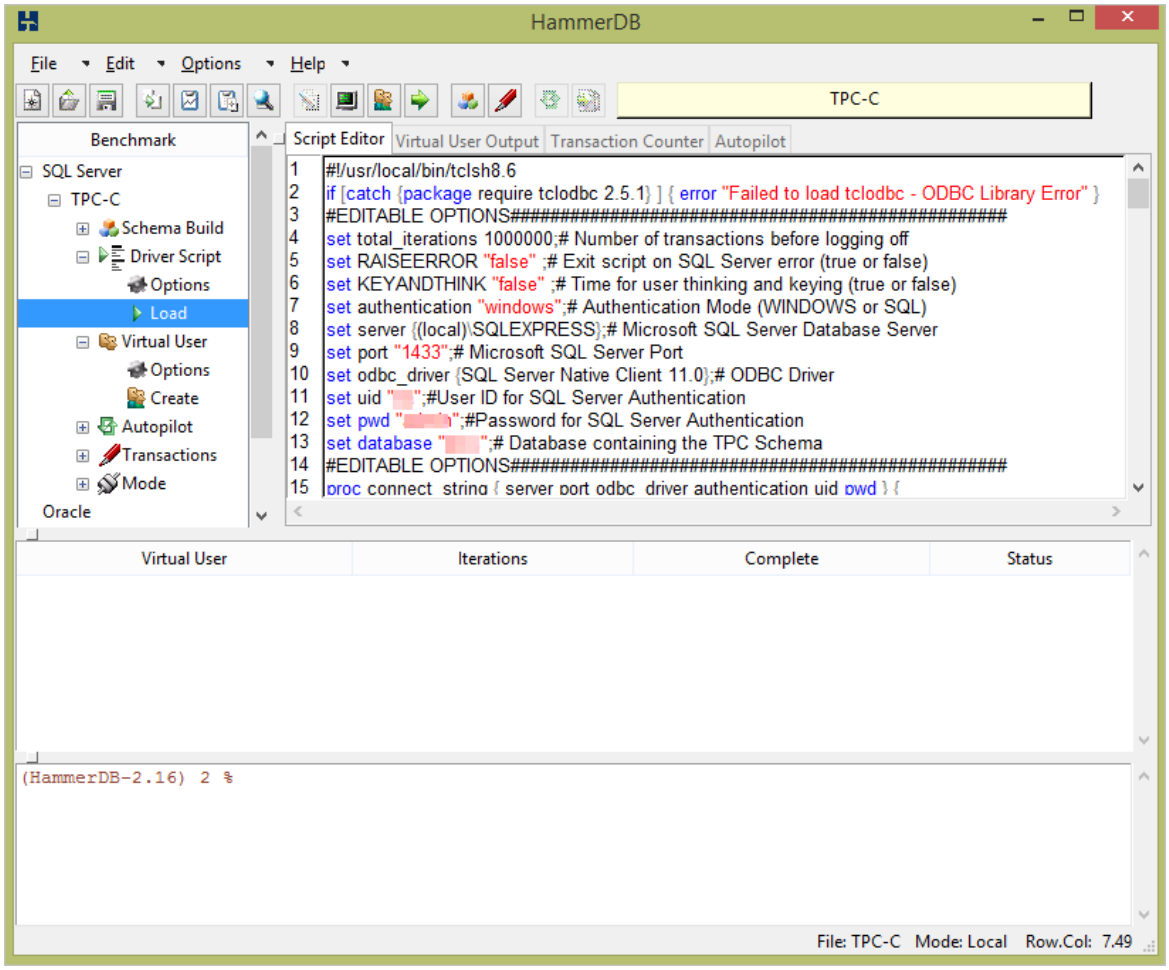
8. 当初始化都显示complete后，单击红色方框停止执行，如下图所示。



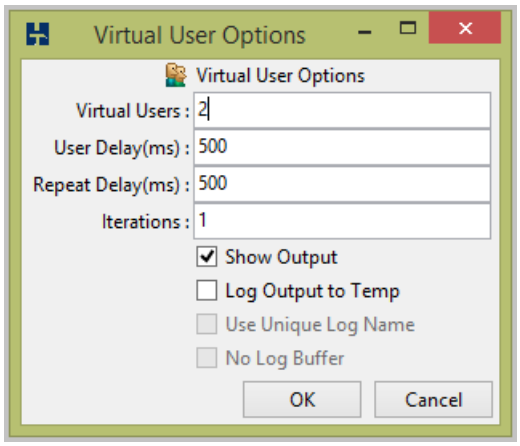
9. 在左侧导航栏中选择**Driver script / option**，确保数据库连接信息正确。



10. 在左侧导航栏中选择**Driver script / load**，如下图所示。

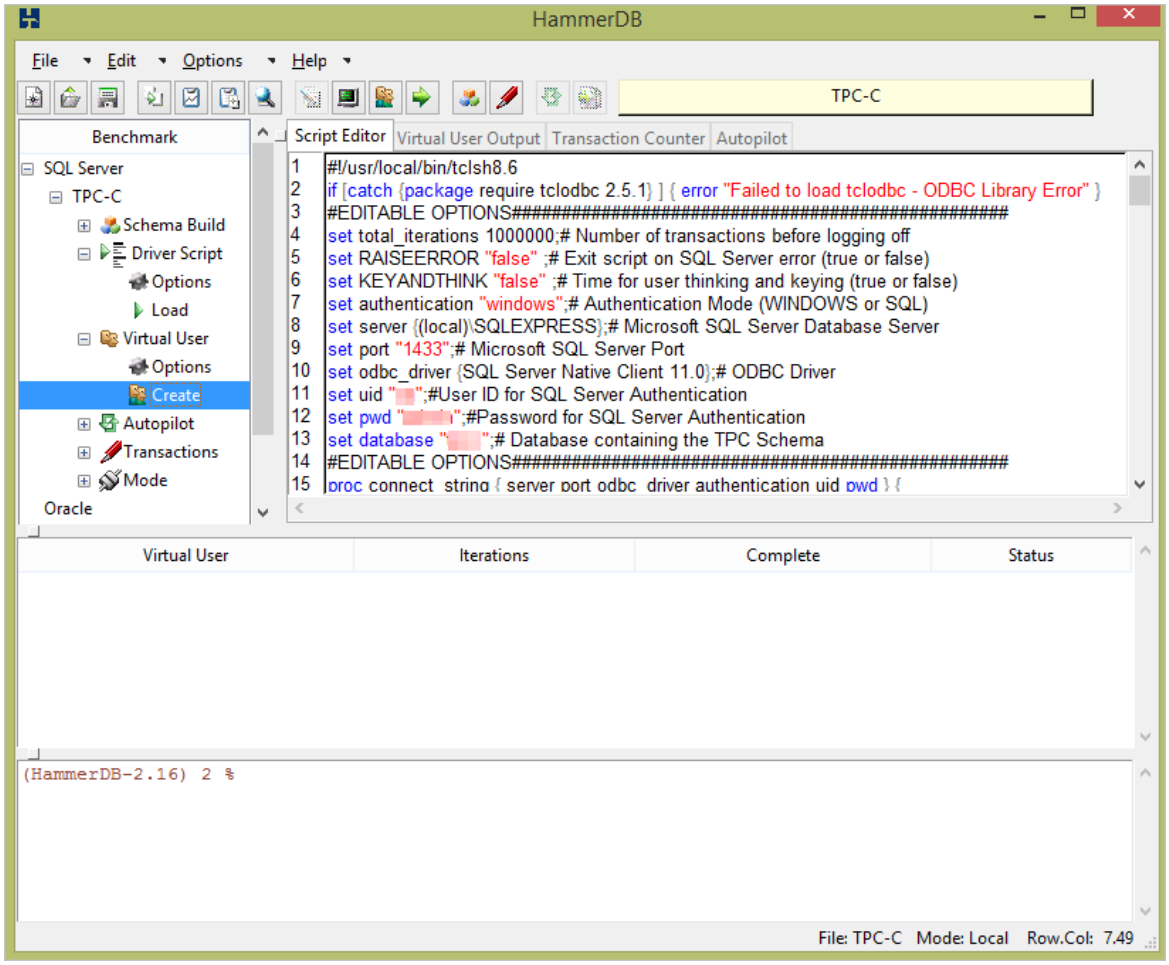


11. 设置Virtual User参数，根据规格配置选择用户数，直到数据库被压出最高TPM。如下图所示。

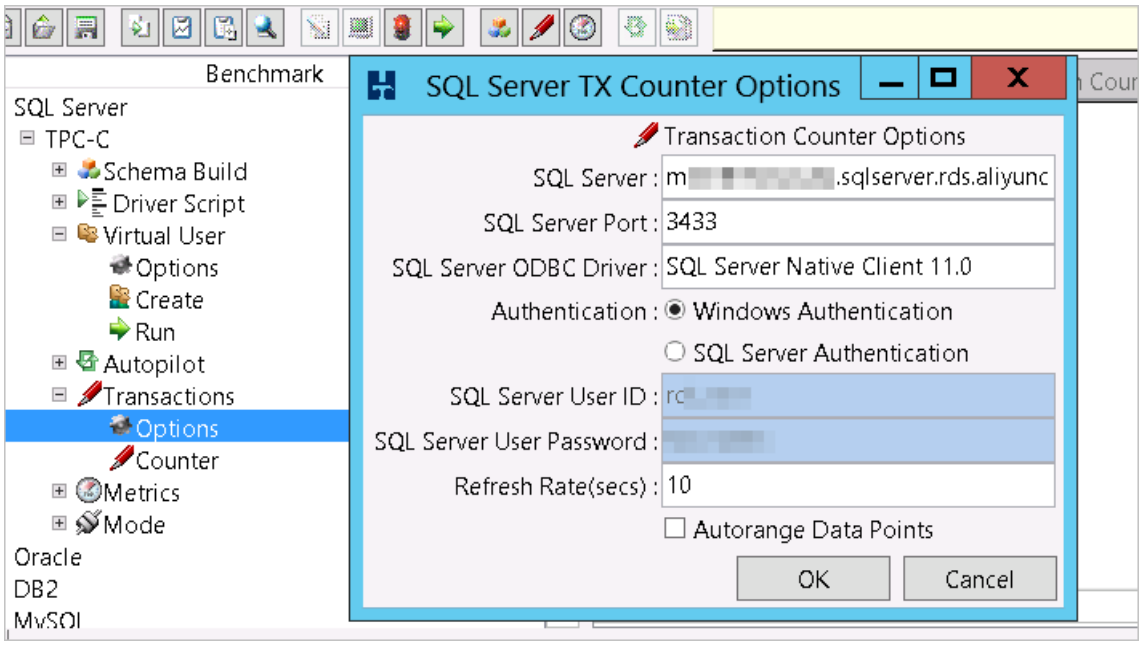


> **说明** 建议不要选择Show Output这个选项，可能会导致客户端无响应。

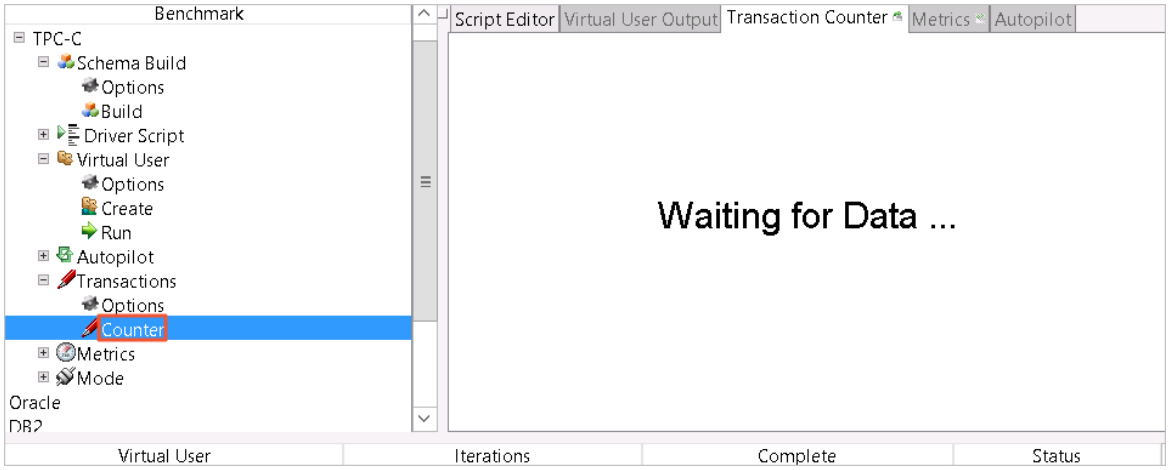
12. 在左侧导航栏中选择**Virtual User/Create**，如下图所示。



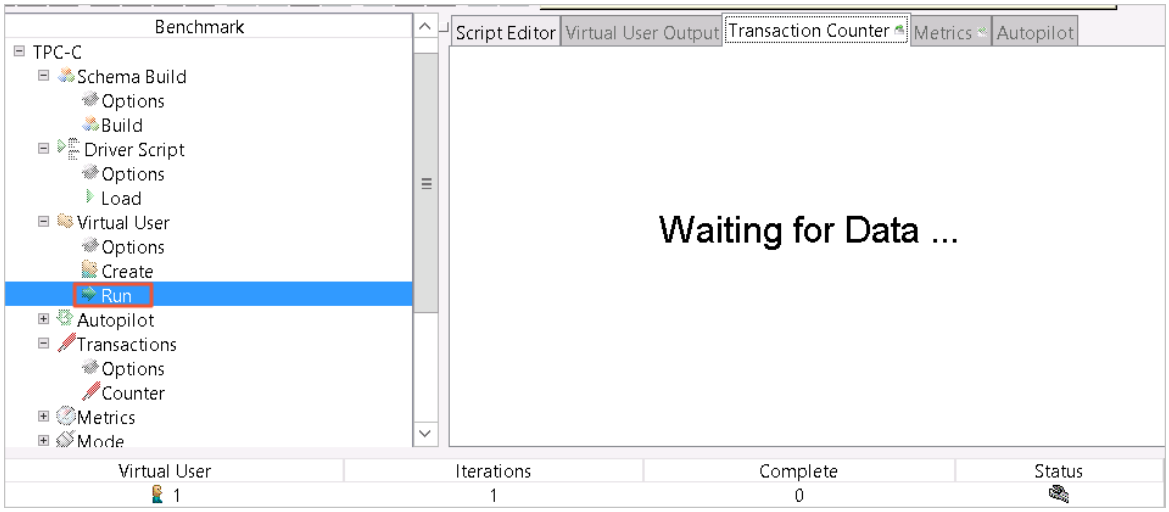
13. 在左侧导航栏中选择**Transcations/Option**，如下图所示。



14. 在左侧导航栏中选择**Transcations/Counter**，如下图所示。



15. 在左侧导航栏中选择**Virtual User/Run**，如下图所示。



2.2.4. 测试模型

库表结构

请参见文档Introduction to Transactional (TPC-C) Testing for all Databases。

表

```
[dbo].[CUSTOMER]
[dbo].[DISTRICT]:
[dbo].[HISTORY]:
[dbo].[ITEM]:
[dbo].[NEW_ORDER]
[dbo].[ORDER_LINE]
[dbo].[ORDERS]
[dbo].[STOCK]
[dbo].[WAREHOUSE]
```

存储过程

```
[dbo].[DELIVERY]
[dbo].[NEWORD]
[dbo].[OSTAT]
[dbo].[PAYMENT]
[dbo].[SLEV]
```

2.2.5. 测试指标

HammerDB本身只提供TPM指标，未提供TPS和Batch Request指标，不过我们在以下的测试中会提供这两个指标。

TPM

Transactions Per Minute，表示数据库每分钟执行的事务数量。

TPS

Transactions Per Second，表示数据库每秒执行的事务数量。

Batch Request

每秒批处理请求数。

> **说明** 这个不是简单地将TPS和查询叠加。

2.3. 测试结果

2.3.1. SQL Server 2008 R2高可用版

本文介绍RDS SQL Server 2008 R2高可用版实例的性能测试结果。

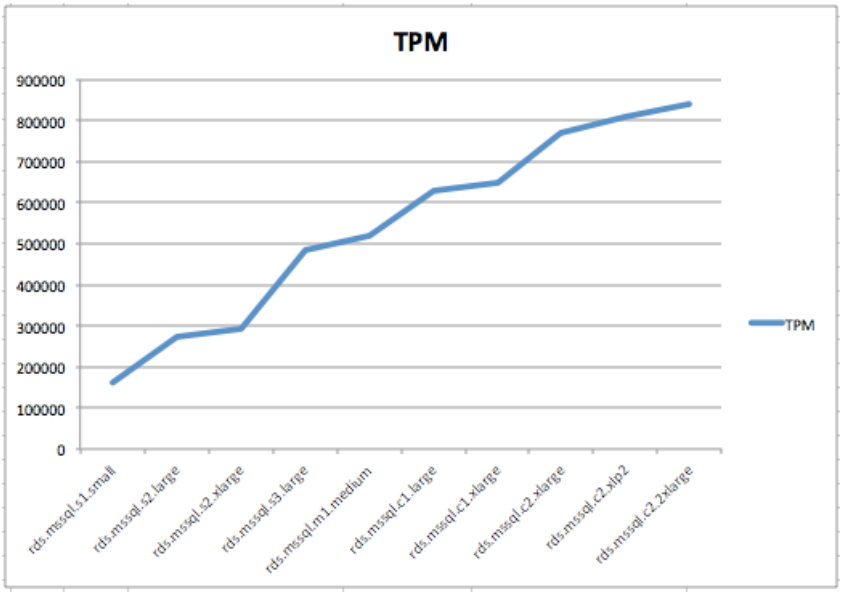
> **说明**
> - 本测试的实现基于TPC-C官方的基准测试，但并不满足TPC基准测试的所有要求。因此，测试结果不能与TPC官方发布的基准测试结果相比较。
> - 为了帮助您更好地了解和使用RDS SQL Server，请参见性能优化与诊断。

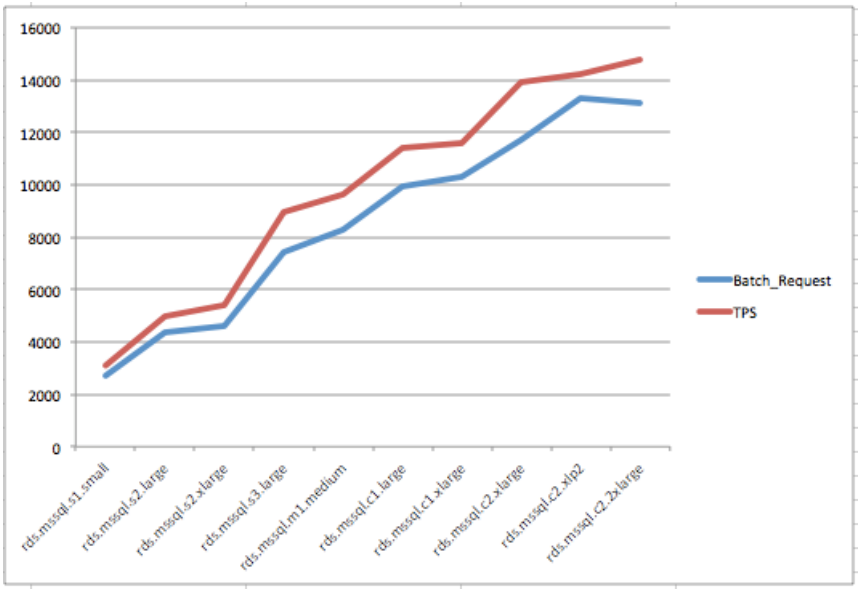
测试规格

实例规格	CPU核数	内存（GB）	IOPS
rds.mssql.s1.small	1	2	1000
rds.mssql.s2.large	2	4	2000
rds.mssql.s2.xlarge	2	8	4000
rds.mssql.s3.large	4	8	5000

实例规格	CPU核数	内存（GB）	IOPS
rds.mssql.m1.medium	4	16	7000
rds.mssql.c1.large	8	16	8000
rds.mssql.c1.xlarge	8	32	12000
rds.mssql.c2.xlarge	16	64	14000
rds.mssql.c2.xlp2	16	96	16000
rds.mssql.c2.2xlarge	16	128	16000

测试结果





编号	规格	TPM	Batch Request	TPS
1	rds.mssql.s1.small	162000	2680	3100
2	rds.mssql.s2.large	273000	4370	4980
3	rds.mssql.s2.xlarge	293000	4600	5400
4	rds.mssql.s3.large	483000	7450	8970
5	rds.mssql.m1.medium	517800	8260	9640
6	rds.mssql.c1.large	630000	9950	11400
7	rds.mssql.c1.xlarge	647000	10300	11600
8	rds.mssql.c2.xlarge	769000	11700	13900
9	rds.mssql.c2.xlp2	810000	13300	14200
10	rds.mssql.c2.2xlarge	840000	13100	14800

2.3.2. SQL Server 2012企业版

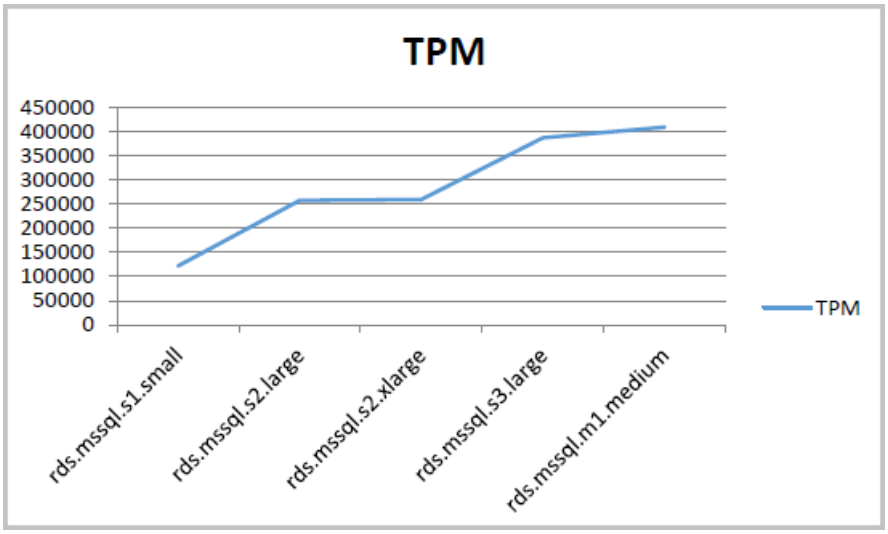
本文介绍RDS SQL Server 2012企业版实例的性能测试结果。

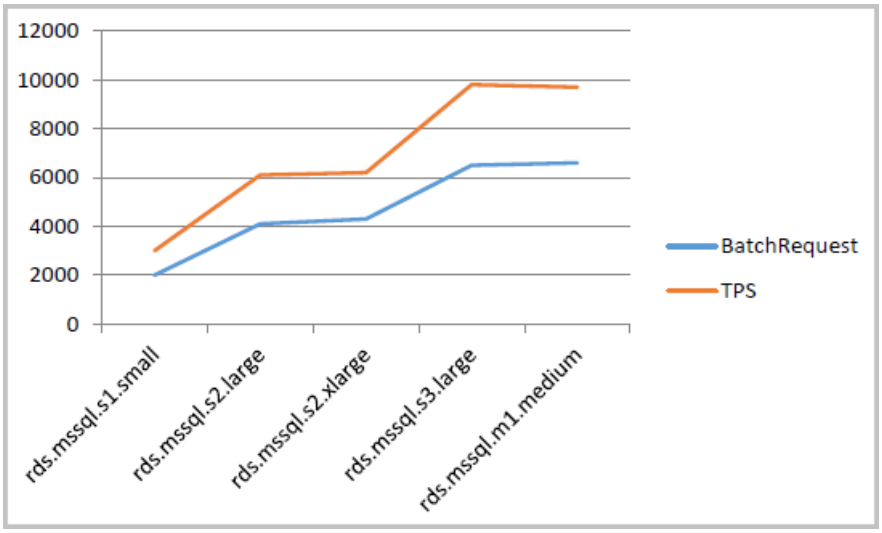
> **说明**
> - 本测试的实现基于TPC-C官方的基准测试，但并不满足TPC基准测试的所有要求。因此，测试结果不能与TPC官方发布的基准测试结果相比较。
> - 为了帮助您更好地了解和使用RDS SQL Server，请参见性能优化与诊断。

测试规格

实例规格	CPU核数	内存（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

测试结果





编号	规格	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.PostgreSQL版

3.1. 性能概述

阿里云关系型数据库RDS（Relational Database Service）是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和SSD盘高性能存储，RDS支持MySQL、SQL Server、PostgreSQL和MariaDB TX引擎，并且提供了容灾、备份、恢复、监控、迁移等方面的全套解决方案，彻底解决数据库运维的烦恼。

自RDS推出市场以来，阿里云数据库行业专家在实践中针对RDS的所有参数不断优化。RDS使用的所有服务器硬件也经过多方的严格评测，保证产品拥有高稳定性和高可用性。

RDS PostgreSQL的性能测试结果请参见测试结果。

3.2. 测试方法

3.2.1. 测试环境

- 所有测试均在华北2（北京）地域完成，PostgreSQL实例和ECS实例在同一可用区。
- ECS的实例规格：ecs.g5.16xlarge（64核 256GiB）
- ECS存储规格： SSD本地盘 200GiB。
- 网络类型：专有网络。
- 操作系统：CentOS 7.6 x64

3.2.2. 测试工具

PostgreSQL自带一款轻量级的压力测试工具：pgbench。pgbench是一种在PostgreSQL上运行基准测试的简单程序。它可以在并发的数据库会话中一遍一遍地运行相同的SQL命令。

安装方法

1. 执行如下命令在ECS实例中安装PostgreSQL 11。

```
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/pg
dg-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:/us
r/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. 测试指标

只读QPS

数据库只读时每秒执行的SQL数（仅包含select）。

读写QPS

数据库读写时每秒执行的SQL数（含insert、select、update）。

3.2.4. 测试步骤

本文介绍测试RDS PostgreSQL实例性能的步骤。

1. 根据目标库大小初始化测试数据，具体命令如下：
   - 初始化数据50亿：`pgbench -i -s 50000`
   - 初始化数据10亿：`pgbench -i -s 10000`
   - 初始化数据5亿：`pgbench -i -s 5000`
   - 初始化数据1亿：`pgbench -i -s 1000`
2. 通过以下命令配置环境变量：

```
export PGHOST=<PostgreSQL实例内网地址>
export PGPORT=<PostgreSQL实例端口>
export PGDATABASE=postgres
export PGUSER=<PostgreSQL数据库用户名>
export PGPASSWORD=<PostgreSQL对应用户的密码>
```

3. 创建只读和读写的测试脚本。
   - 创建只读脚本ro.sql内容如下：

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

   - 创建读写脚本rw.sql内容如下：

```
\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. 使用如下命令测试：
   - 只读测试：

```
rds.pg.st.h43，总数据量50亿，热数据1亿
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，总数据量50亿，热数据5亿
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，总数据量50亿，热数据10亿
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，总数据量50亿，热数据50亿
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，总数据量10亿，热数据1亿
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，总数据量10亿，热数据5亿
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，总数据量10亿，热数据10亿
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，总数据量10亿，热数据1亿
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，总数据量10亿，热数据5亿
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，总数据量10亿，热数据10亿
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，总数据量10亿，热数据1亿
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，总数据量10亿，热数据5亿
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，总数据量10亿，热数据10亿
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，总数据量5亿，热数据1亿
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，总数据量5亿，热数据5亿
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，总数据量1亿，热数据5000万
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，总数据量1亿，热数据1亿
pgbench -M prepared -n -r -P 1 -f ./ro.sql -c 8 -j 8 -T 120 -D scale=1000 -D range=10
0000000
```

- 读写测试：

```
rds.pg.st.h43，总数据量50亿，热数据1亿
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，总数据量50亿，热数据5亿
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，总数据量50亿，热数据10亿
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，总数据量50亿，热数据50亿
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，总数据量10亿，热数据1亿
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，总数据量10亿，热数据5亿
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，总数据量10亿，热数据10亿
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，总数据量10亿，热数据1亿
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，总数据量10亿，热数据5亿
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，总数据量10亿，热数据10亿
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，总数据量10亿，热数据1亿
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，总数据量10亿，热数据5亿
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，总数据量10亿，热数据10亿
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，总数据量5亿，热数据1亿
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，总数据量5亿，热数据5亿
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，总数据量1亿，热数据5000万
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，总数据量1亿，热数据1亿
pgbench -M prepared -n -r -P 1 -f ./rw.sql -c 8 -j 8 -T 120 -D scale=1000 -D range=10
0000000
```

> 说明
> - scale乘以10万：表示测试数据量。
> - range：表示活跃数据量。
> - -c：表示测试连接数，测试连接数不代表该规格的最大连接数，最大连接数请参见主实例规格列表。

3.3. 测试结果

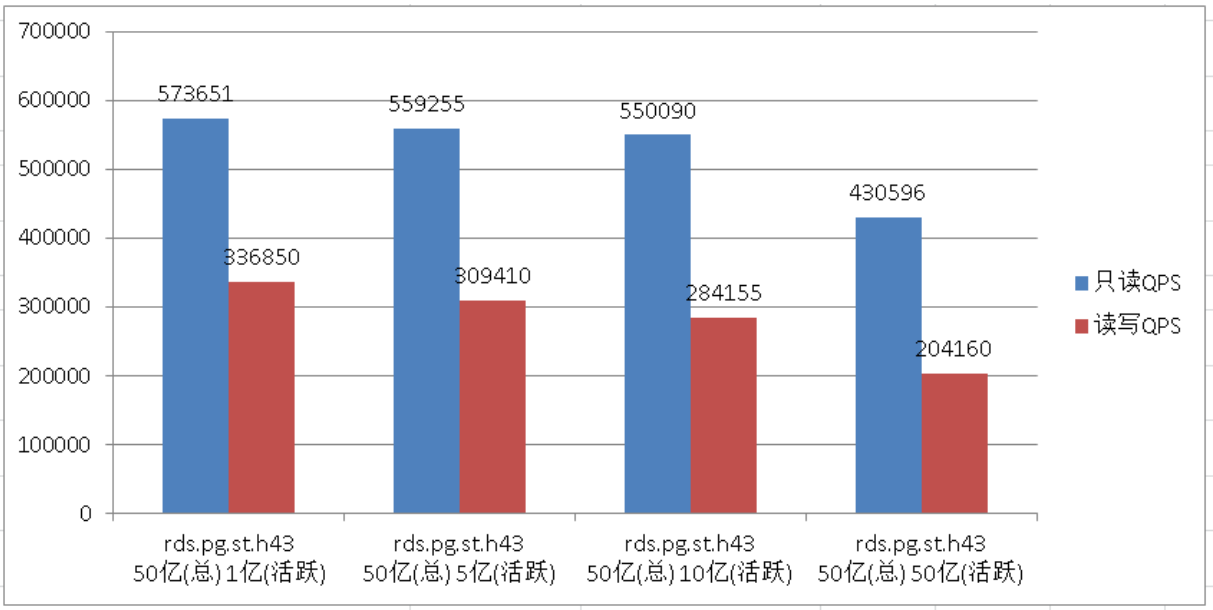
> 说明 以下为RDS PostgreSQL的性能测试结果，仅供参考。为了帮助您更好地了解和使用RDS PostgreSQL，请参见性能优化与诊断和开发运维建议。

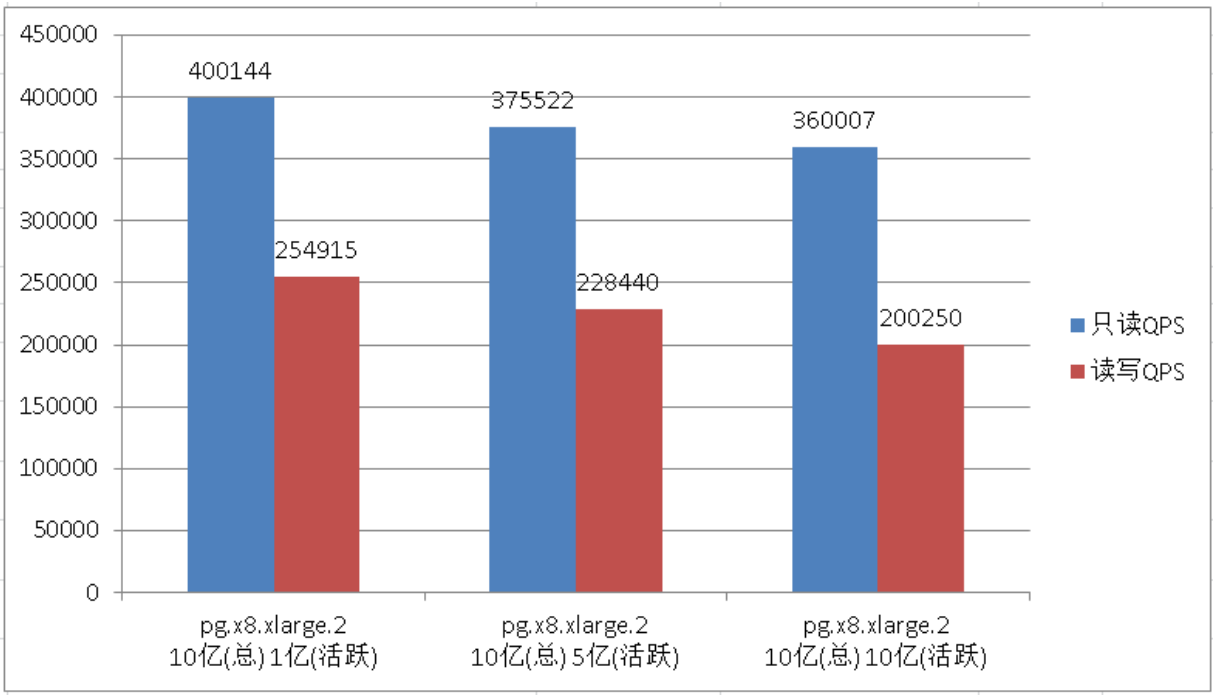
规格	预计可存储数据量	测试数据量	热（活跃）数据量	只读QPS	读写QPS
rds.pg.st.h43 60核470G3T	150亿	50亿	1亿	573651	336850
rds.pg.st.h43 60核470G3T	150亿	50亿	5亿	559255	309410
rds.pg.st.h43 60核470G3T	150亿	50亿	10亿	550090	284155
rds.pg.st.h43 60核470G3T	150亿	50亿	50亿	430596	204160
pg.x4.4xlarge.2 32核128G2T	100亿	10亿	1亿	400144	254915
pg.x4.4xlarge.2 32核128G 2T	100亿	10亿	5亿	375522	228440
pg.x4.4xlarge.2 32核128G 2T	100亿	10亿	10亿	360007	200250
pg.x8.2xlarge.2 16核128G2T	100亿	10亿	1亿	235592	156330
pg.x8.2xlarge.2 16核128G2T	100亿	10亿	5亿	221153	133125

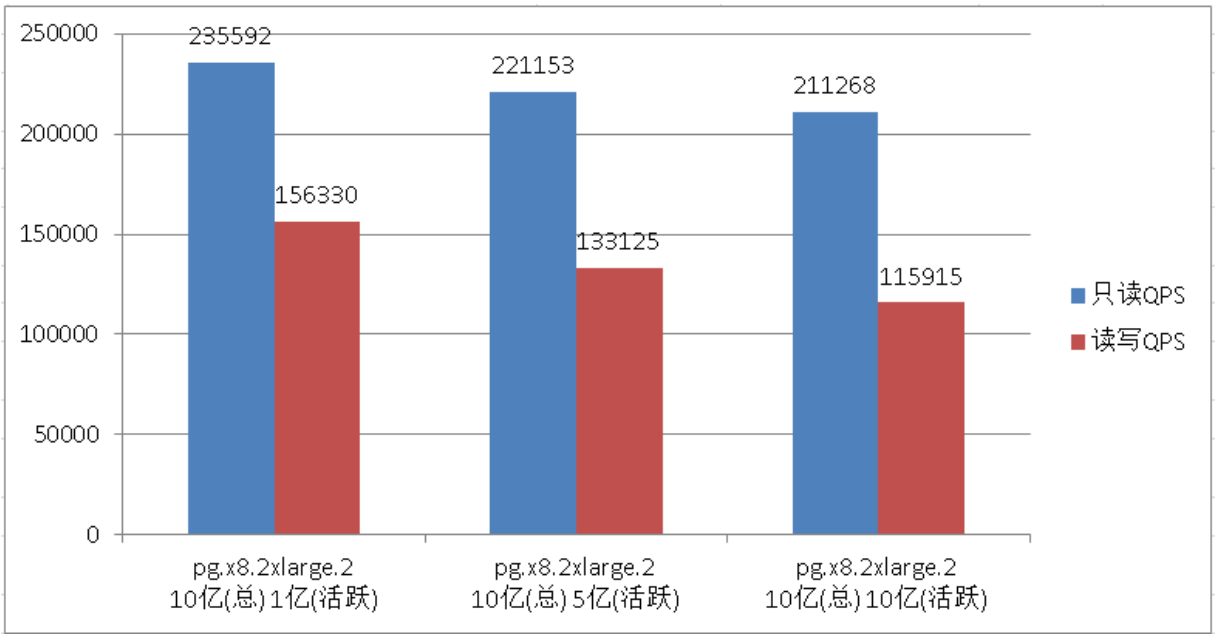
规格	预计可存储数据量	测试数据量	热（活跃）数据量	只读QPS	读写QPS
pg.x8.2xlarge.2 16核128G2T	100亿	10亿	10亿	211268	115915
pg.x8.xlarge.2 8核64G1T	50亿	10亿	1亿	129323	71820
pg.x8.xlarge.2 8核64G1T	50亿	10亿	5亿	115498	58140
pg.x8.xlarge.2 8核64G1T	50亿	10亿	10亿	102735	58555
pg.x8.large.2 4核32G500G	25亿	5亿	1亿	75729	45110
pg.x8.large.2 4核32G500G	25亿	5亿	5亿	63818	36375
pg.x8.medium.2 2核16G250G	12.5亿	1亿	5000万	37245	24520
pg.x8.medium.2 2核16G250G	12.5亿	1亿	1亿	35321	19880

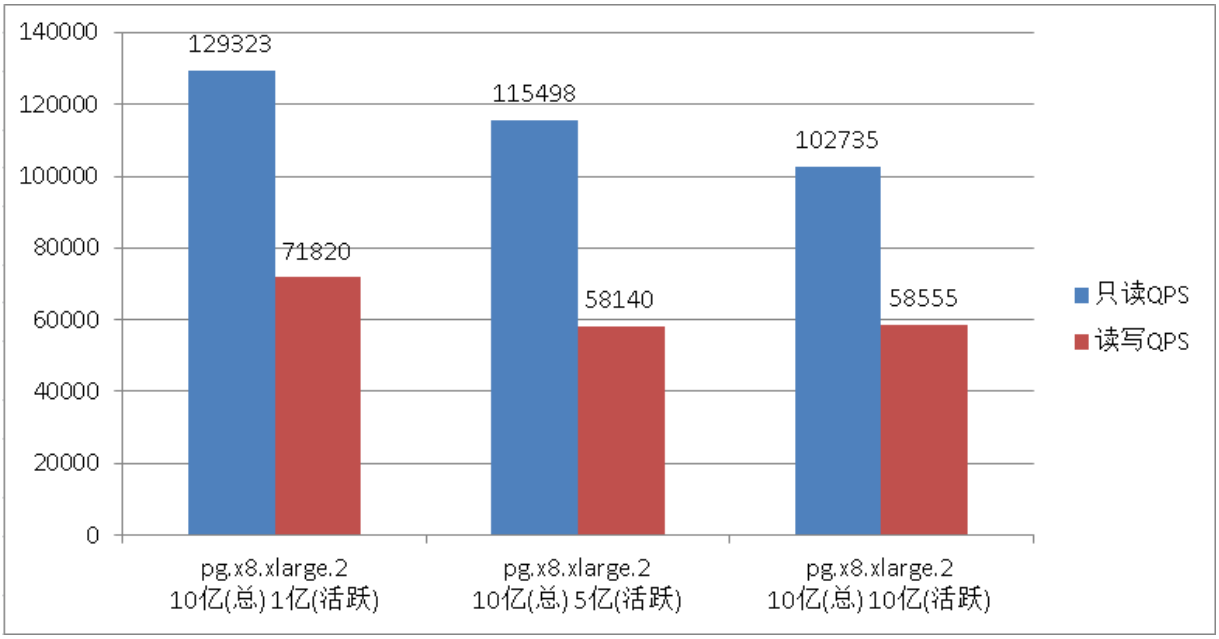
说明

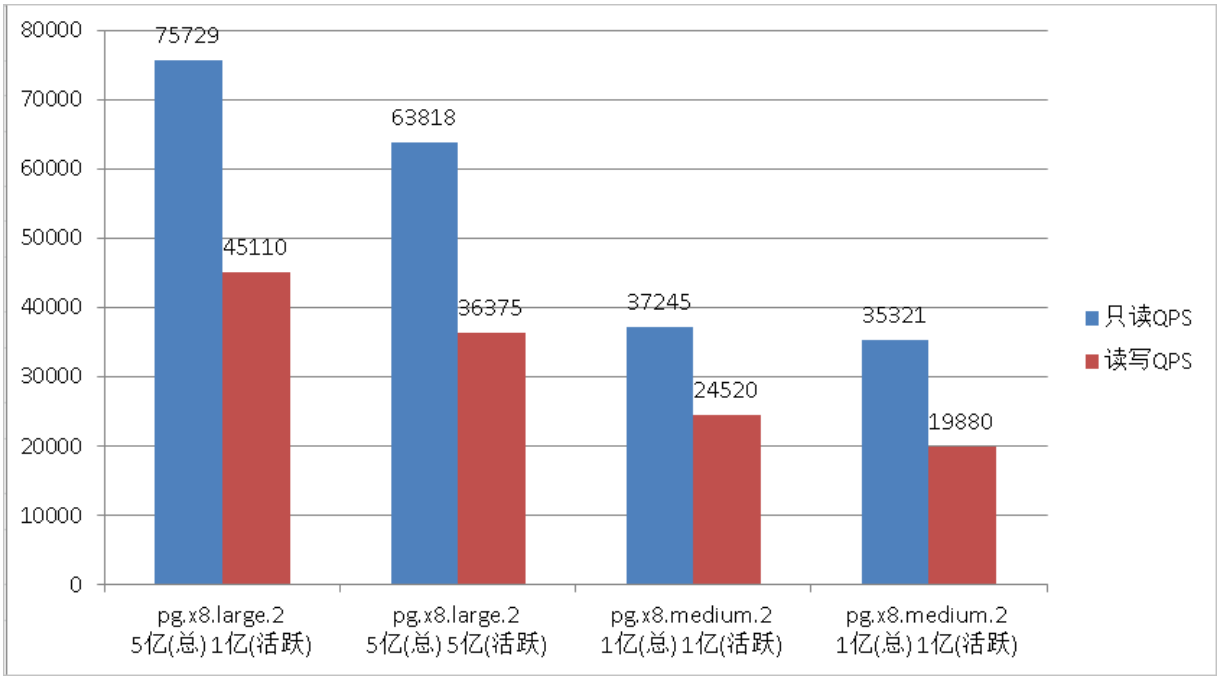
- 规格：RDS for PostgreSQL的规格代码。
- 预计可存储数据量：预计该规格实例可以存储多少条记录。
- 测试数据量：本轮测试数据的记录数。
- 热（活跃）数据量：本轮测试的查询、更新SQL的记录数范围。
- 只读QPS：只读测试的结果，表示每秒请求数。
- 读写QPS：读写测试的结果，表示每秒请求数。


700000
600000
500000
400000
300000
200000
100000
0
573651
336850
559255
309410
550090
284155
430596
204160
只读QPS
读写QPS
rds.pg.st.h43
50亿(总) 1亿(活跃)
rds.pg.st.h43
50亿(总) 5亿(活跃)
rds.pg.st.h43
50亿(总) 10亿(活跃)
rds.pg.st.h43
50亿(总) 50亿(活跃)


450000
400000
350000
300000
250000
200000
150000
100000
50000
0
400144
254915
375522
228440
360007
200250
只读QPS
读写QPS
pg.x8.xlarge.2
10亿(总) 1亿(活跃)
pg.x8.xlarge.2
10亿(总) 5亿(活跃)
pg.x8.xlarge.2
10亿(总) 10亿(活跃)


250000
200000
150000
100000
50000
0
235592
156330
221153
133125
211268
115915
只读QPS
读写QPS
pg.x8.2xlarge.2
10亿(总) 1亿(活跃)
pg.x8.2xlarge.2
10亿(总) 5亿(活跃)
pg.x8.2xlarge.2
10亿(总) 10亿(活跃)


140000
120000
100000
80000
60000
40000
20000
0
129323
71820
115498
58140
102735
58555
只读QPS
读写QPS
pg.x8.xlarge.2
10亿(总) 1亿(活跃)
pg.x8.xlarge.2
10亿(总) 5亿(活跃)
pg.x8.xlarge.2
10亿(总) 10亿(活跃)


80000
70000
60000
50000
40000
30000
20000
10000
0
75729
45110
63818
36375
37245
24520
35321
19880
pg.x8.large.2
5亿(总) 1亿(活跃)
pg.x8.large.2
5亿(总) 5亿(活跃)
pg.x8.medium.2
1亿(总) 1亿(活跃)
pg.x8.medium.2
1亿(总) 1亿(活跃)
只读QPS
读写QPS

4.自建数据库与RDS性能对比的注意事项

您可以通过测试来对比自建数据库与RDS的性能差异，但是对比时需要保证二者具有相同的条件，如相同的网络环境、性能规格、数据库版本等。本文介绍具体的注意事项。

您可以自行搭建数据库，或者购买阿里云RDS实例。推荐您购买RDS实例，因为它完全由阿里云托管，而且有完备的安全、备份、恢复、扩容、性能优化等机制，您无需执行各种安全措施（如搭建备库）和维护工作，只需专注于业务的发展和创新。

关于自建数据库与RDS的各方面对比，请参见RDS与自建数据库对比优势。

网络环境

- 应用和自建数据库都需要部署于ECS实例，且与RDS实例位于同一地域，使得应用与RDS之间，以及应用与自建数据库之间都是通过内网进行通信。

  > **说明** 应用与自建数据库需要部署于两台不同的ECS实例。如果部署于同一台ECS实例，则应用到自建数据库的网络路径短于应用到RDS的网络路径，而且应用对CPU资源的占用也会影响自建数据库的性能表现，对比测试将不公平。

- 您可以采用以下其中一种部署架构：
  - 应用、自建数据库和RDS实例的主节点位于同一个可用区。
  - 自建数据库与RDS实例的主节点位于同一个可用区。应用位于同一地域下的另一个可用区。

性能规格

自建数据库所在ECS实例和RDS拥有相同的CPU核数与内存。

数据库版本

自建数据库和RDS的数据库版本相同，例如，两者都为MySQL 5.6。

数据复制方式

主备节点之间的数据复制方式分为异步、半同步和强同步。关于数据复制方式的介绍，请参见数据复制方式。

具体要求如下表所示。

自建数据库	RDS
无备库，不涉及数据复制方式。	高可用版，且数据复制方式为异步。
有1个备库，且数据复制方式为异步。	高可用版，且数据复制方式为异步。
有1个备库，且数据复制方式为半同步。	高可用版，且数据复制方式为半同步。
有2个备库，且数据复制方式为强同步。	三节点企业版（数据复制方式为强同步，且无法修改）。

数据库参数

自建数据库和RDS的参数设置需要一致。

关于如何修改RDS的参数设置，请参见修改实例参数。

> **说明** 出于安全考虑，RDS不支持部分参数的修改。如果某参数不一致，而RDS不支持修改，请修改自建数据库的参数。

案例分析

场景：某客户正在将本地的业务系统迁移上云，在RDS上的SQL执行时间明显要比线下自建数据库执行时间要长1倍。

原因：自建数据库与RDS的参数配置不同，如下所示：

- 用户本地参数配置：

  join_buffer_size = 128M

  read_rnd_buffer_size = 128M

  tmp_table_size = 128M

- RDS参数配置：

  join_buffer_size = 1M

  read_buffer_size = 1M

  tmp_table_size =256K