TDSQL-C for MySQL

White Paper

Product Documentation

Contents

White Paper
Security White Paper
Overview

Last updated：2023-08-22 15:57:03

TDSQL-C for MySQL is a new-generation cloud-native relational database developed by Tencent Cloud. It combines the strengths of traditional databases, cloud computing, and cutting-edge hardware technologies to deliver high performance and availability. It is fully compatible with MySQL and offers a throughput of over one million QPS and a petabyte-level distributed smart storage, ensuring data security and reliability.
TDSQL-C for MySQL simplifies your IT Ops and allows you to focus more on business development with its various features such as backup, restoration, monitoring, fast scaling, and data transfer.
TDSQL-C for MySQL provides diverse security reinforcement features to ensure the reliability and security of your data. In order to make your databases more secure, we recommend you use the following security features based on your business needs:

Security Feature	Security Capability
Data storage security	Automatic backup Periodic backup retention Data security
Security audit	Compliance audit
Access control	Database account management Access management Custom password strength
Data communication security	VPC Security group
Data disaster recovery	Intra-region disaster recovery

Data Storage Security

Last updated：2023-08-24 09:50:00

TDSQL-C for MySQL stores your data confidentially and maintains its integrity.

Automatic backup

TDSQL-C for MySQL supports both automatic and manual backup to ensure data restorability that guarantees data integrity and reliability. It provides data backup and binlog backup features by default. Automatic backup is performed once a day between 2:00 AM and 6:00 AM by default. You can customize the backup start time and retention period as needed in the console. If you have other backup needs, you can also initiate manual backup through the console or APIs at any time.
For more information on how to use this feature, see Automatic Backup.

Periodic backup retention

TDSQL-C for MySQL allows you to set the backup start time and retention period flexibly for higher data security. You can shorten or extend the retention period based on your business needs, which is 7 days by default and can be up to 1,830 days. Backup files that exceed the retention period will be automatically deleted.
For more information on how to use this feature, see Setting Backup Retention Period.

Security Audit

Last updated：2023-08-22 15:58:04

TDSQL-C for MySQL provides security auditing capabilities to monitor and log access and operation behaviors in the database management system, so as to audit security incidents and implement protection measures in the system.

Compliance audit

TDSQL-C for MySQL provides an enterprise-grade compliance audit feature that offers the benefits of compliance, security, and traceability. You can use this feature to pass relevant compliance certifications since it complies with applicable regulations.

By recording and intelligently analyzing your behaviors of accessing the database through AI technologies, it helps you generate security compliance audit reports and discover risky behaviors in time. In addition, it promptly triggers alarms for access behaviors that violate security policies, which ensures that database operations meet compliance requirements and helps you pass applicable compliance certifications.

Note：

When full audit is performed, the system performance loss does not exceed 5% as a tradeoff of functionality and performance.

Attack Protection

Last updated：2022-03-25 14:51:29

DDoS Attack Prevention

When you use the public network to connect to and access a TDSQL-C for MySQL instance, you may suffer from DDoS attacks. To address this problem, Tencent Cloud provides traffic cleansing and blocking features that are automatically triggered and stopped by the system. When the Anti-DDoS system detects that your instance is under attacks, it will automatically enable traffic cleansing or block the traffic if the attacks cannot be resisted by cleansing or reach the blocking threshold.

Note:

We recommend you access TDSQL-C for MySQL over the private network to avoid DDoS attacks.

Traffic Cleansing

When the public network traffic of a TDSQL-C for MySQL instance exceeds the threshold, Anti-DDoS will automatically cleanse the inbound traffic to the instance. Policy-based routing will be used to redirect the traffic from the original network route to the DDoS cleansing devices of Anti-DDoS, which will identify the public network traffic, discard attack traffic, and forward normal traffic to the instance.

Blocking

When the attack traffic suffered by a TDSQL-C for MySQL instance exceeds the blocking threshold, Tencent Cloud will block all public network access requests to this instance through applicable ISP services to prevent other Tencent Cloud users from being affected. This means that when the bandwidth of the attack traffic suffered by your instance exceeds the maximum protection bandwidth, Tencent Cloud will block all public network access requests to it.

When the following conditions are met, blocking will be triggered:

The bits per second (bps) value reaches 2 Gbps.

Traffic cleansing is not effective.

When the following condition is met, blocking will be stopped:

2 hours elapses after blocking starts.

Access Control

Last updated：2024-01-03 14:39:19

TDSQL-C for MySQL provides access control capabilities. By defining and verifying user permissions, regulating user access to database resources, and managing database resource permissions, you can ensure that only authorized users can access database objects within the scope of their permissions or at their security levels.

Database account management

You can create database accounts through the TDSQL-C for MySQL console or API. You can also grant management permissions at different levels to such accounts. We recommend you authorize accounts based on the principle of least privilege to ensure the data security.
For more information, see Creating Account.

Access management

Cloud Access Management (CAM) helps you securely manage and control access permissions to your Tencent Cloud resources. With CAM, you can create, manage, and terminate users (groups), and control the Tencent Cloud resources that can be used by the specified user through identity and policy management, which implements permission separation.
For more information, see CAM Overview.

Custom password strength

Passwords are the most important means for protecting database security. As more data security regulations are introduced, there are higher requirements for the database password strength. TDSQL-C for MySQL supports the custom password strength feature to protect your database security and meet your needs for compliance with applicable regulations.
You can configure this feature in the console to enable password strength for all password-related operations. This helps protect your passwords from leakage or other risks. The feature offers the following configuration items:
Min Number of Uppercase or Lowercase Letters
Min Number of Digitals
Min Number of Symbols
Min Number of Password Characters
Non-Compliant Dictionary

For more information, see Overview.

Data Communication Security

Last updated：2023-02-08 09:46:36

TDSQL-C for MySQL provides data communication security capabilities to ensure the confidentiality and integrity of data during communication.

VPC

TDSQL-C for MySQL supports using Virtual Private Cloud (VPC) to achieve a higher degree of network isolation and control. A VPC is a logically isolated network space in Tencent Cloud. In a VPC, you can customize IP ranges, IP addresses, and routing policies to implement network isolation at the resource level.
Read-write instances and read-only instances in a TDSQL-C for MySQL cluster deployed in a VPC can only be accessed by CVM instances in the same VPC by default. If the CVM and instances are in different VPCs, they can communicate after you apply for public network access. For the sake of network security, we recommend you not access your databases over the public network. If you have to do so, configure appropriate security groups to implement access control for clients.
For more information on the feature, see VPC Overview.

Security group

TDSQL-C for MySQL supports security group as an important means for network security isolation, which can be used to set network access controls for one or more instances. Instances with the same network security isolation demands in one region can be put into the same security group, which is a logical group.
For more information on how to use this feature, see Creating and Managing TencentDB Security Groups.

Disaster Recovery

Last updated：2023-02-08 09:46:36

TDSQL-C for MySQL provides the cross-AZ data disaster recovery feature to help you deliver continued services at low costs while improving data reliability or meeting compliance requirements.

Intra-region disaster recovery

TDSQL-C for MySQL supports multi-AZ deployment, where physical servers are deployed in different AZs in the same region. When an AZ fails, the business traffic will be switched to another AZ swiftly, which is imperceptible to the business and requires no changes at the application layer to achieve intra-region disaster recovery.
For more information on how to use this feature, see Setting Multi-AZ Deployment.

Network Isolation

Last updated：2022-03-25 11:19:55

TDSQL-C for MySQL supports the use of VPC to achieve a higher degree of network isolation and control. Using security group and VPC together can greatly improve the security of access to TDSQL-C for MySQL.
A VPC is a logically isolated network space established for users in Tencent Cloud. In a VPC, you can freely define IP range segmentation, IP addresses, and routing policies to achieve resource-level network isolation.
TDSQL-C for MySQL instances deployed in a VPC can only be accessed by CVM instances in the same VPC by default. If the CVM and TDSQL-C for MySQL instances are in different VPCs, they can communicate after you apply for public network access. For network security considerations, we recommend you not access your databases over the public network. If you have to do so, configure appropriate security groups to implement access control for clients.

Backup and Restoration

Last updated：2022-03-31 22:08:06

Backup

TDSQL-C for MySQL supports both automatic and manual backup to ensure data restorability that guarantees data integrity and reliability. If you have other backup needs, you can initiate manual backup in the console at any time.
For more information on how to use this feature, see Backing up Data.

Rollback

TDSQL-C for MySQL supports database/table-level rollback. You can use the rollback feature to restore data to any time point within the retention period as needed.
For more information on how to use this feature, see Rolling back Data.

Clone

TDSQL-C for MySQL supports rollback of the entire cluster to a new cluster (clone). This feature can restore a cluster to any time point in the log backup retention period or to the backup set of the specified backup file through clone.
For more information on how to use this feature, see Cloning Cluster.

Audit and Governance

Last updated：2022-03-25 21:17:49

TDSQL-C for MySQL provides an enterprise-grade database audit feature. It complies with applicable national regulations and has the benefits of compliance, security, and traceability, making it a necessary service for you to pass the Cybersecurity Classified Protection (CCP) compliance certification.

By recording and analyzing your behaviors of accessing the database, it helps you generate security compliance audit reports and discover risky behaviors in time. In addition, it promptly triggers alarms for access behaviors that violate security policies, which ensures that database operations meet compliance requirements and helps you pass the CCP compliance certification.

Note:

In terms of both functionality and performance, when audit is enabled and full audit is performed, the system performance loss does not exceed 5%.

Database Inspection

Last updated：2022-02-11 15:22:06

Database inspection is used to automatically and regularly perform health checks on all instances. You can also set up custom inspections based on your own needs to help troubleshoot potential instance issues and provide solutions. A database inspection report contains the following sections: overview, basic information, health, instance status, exception diagnosis, slow SQL analysis, high-risk accounts, big table analysis, and performance curves.

Data Termination

Last updated：2022-03-25 11:45:02

When you terminate your TDSQL-C for MySQL instance, all data (including backup data) stored in it will be terminated. Tencent Cloud will not retain the data or actively restore your instance.
For more information, see Deleting Cluster/Instance.

Version Upgrade

Last updated：2022-03-25 11:43:23

TDSQL-C for MySQL will provide you with the latest version of database services. When a severe bug or security vulnerability occurs in the system, your TDSQL-C for MySQL instances will be upgraded during your maintenance time window, and upgrade notifications will be pushed to you in advance. The version upgrade process may cause a momentary disconnection; therefore, make sure that your business has a reconnection mechanism.

Performance White Paper

Performance Overview

Last updated：2023-05-15 15:33:27

TDSQL-C for MySQL is a new-generation cloud native relational database developed by Tencent Cloud. It combines the strengths of traditional databases, cloud computing, and cutting-edge hardware technologies to deliver high performance and availability. It is fully compatible with MySQL, with a throughput of over one million QPS and a massive distributed smart storage capacity at the petabyte level. It also supports serverless scaling within seconds, helping you accelerate your digital transformation.

TDSQL-C for MySQL simplifies your IT Ops and allows you to focus more on business development with its various features such as backup, restoration, monitoring, fast scaling, and data transfer.

Continuously tested and optimized by a professional team, TDSQL-C for MySQL offers many MySQL Enterprise Edition features and can flexibly and efficiently process transactions. Its engine kernel has also been deeply optimized to deliver advanced and complete security protection capabilities, massive instance capacity, and superior performance.

This document compares the performance of TDSQL-C for MySQL and TencentDB for MySQL under the dataset characteristics of full cache, big dataset, and 1 TB single table in write, read, and read-write scenarios respectively to show TDSQL-C for MySQL's overall performance. The specific test scenarios are as described below.

TDSQL-C for MySQL features an major upgrade. The new architecture uses RDMA over the entire linkage, optimizes the performance in various aspects based on the enterprise-grade TXSQL kernel, upgrades the distributed storage layer architecture, and supports new hardware devices.

Note:

Currently, the new architecture is in beta test in Beijing Zone 6. To try it out, submit a ticket for application.

Dataset Characteristics	Test Scenario	Read Type
Full cache	Write	-
	Read	POINT SELECT
	Read	RANGE SELECT
	Read-write	POINT SELECT
	Read-write	RANGE SELECT
Big dataset	Write	-
	Read	POINT SELECT
	Read	RANGE SELECT

	Read-write	POINT SELECT
	Read-write	RANGE SELECT
1 TB single table	Write	-
	Read	POINT SELECT
	Read	RANGE SELECT
	Read-write	POINT SELECT
	Read-write	RANGE SELECT

Note:

In the above table, **POINT SELECT** and **RANGE SELECT** are defined as follows.

POINT SELECT: Point test, indicating the number of queries for point selection tests in a single transaction.

RANGE SELECT: Range test, indicating the number of queries for range selection tests in a single transaction.

Test Methods
Test Environment

Last updated：2023-11-01 17:28:05

This document describes the environment used for the TDSQL-C for MySQL performance test.

Test Object: TDSQL-C for MySQL

Region/AZ: Beijing - Beijing Zone 6
Client
Client type: CVM
Client specification: S5.4XLARGE32 (Standard S5 with 16 CPU cores and 32 GB memory)
Client operating system: CentOS 7
Number of clients: 2 (one more client is added after the concurrency exceeds 1,000, and so on)
Information of the tested TDSQL-C for MySQL instance:
AZ deployment: Single-AZ
Database version: MySQL 5.7
Parameter template: The default template is used, with the following parameters adjusted:
log_bin=off
thread_handling=pool-of-threads
innodb_log_sync_method=async
Note：
You can't modify the `log_bin` and innodb_log_sync_method` parameters. To do so, submit a ticket for assistance.
VPC network latency: 0.6 ms
Instance type/specification:

Instance type	Instance specification
Dedicated	2-core, 16 GB MEM
	4-core, 16 GB MEM
	4-core, 32 GB MEM
	8-core, 32 GB MEM
	8-core, 64 GB MEM
	16-core, 64 GB MEM

	16-core, 96 GB MEM
	16-core, 128 GB MEM
	32-core, 128 GB MEM
	32-core, 256 GB MEM
	64-core, 256 GB MEM

Test Object: TencentDB for MySQL

Region/AZ: Guangzhou - Guangzhou Zone 6

Client

Client type: CVM

Client specification: S5.4XLARGE32 (Standard S5 with 16 CPU cores and 32 GB memory)

Client operating system: CentOS 7

Number of clients: 2 (one more client is added after the concurrency exceeds 1,000, and so on)

The information of the tested TencentDB for MySQL instance is as follows:

Storage type: Local SSD

AZ deployment: Single-AZ

Database version: MySQL 5.7

Architecture: Two-node

Replication mode: Async replication

Parameter template: High-performance parameter template

VPC network latency: 0.5 ms

Instance type/specification: Same as above

Test Tools

Last updated：2023-03-01 14:33:46

This document describes how to install SysBench, a performance testing tool for TDSQL-C for MySQL, in a CVM instance.

SysBench overview

SysBench is a modular, cross-platform, and multi-threaded benchmark tool for evaluating OS parameters that are important for a system running a database under intensive load. The idea of this benchmark suite is to quickly get an impression about system performance without setting up complex database benchmarks or even without installing a database at all.

SysBench parameter description

Parameter	Description
db-driver	Database engine
mysql-host	TDSQL-C for MySQL server host
mysql-port	TDSQL-C for MySQL server port
mysql-user	TDSQL-C for MySQL account
mysql-password	TDSQL-C for MySQL password
mysql-db	TDSQL-C for MySQL database name
table_size	Test table size
tables	Number of test tables
events	Number of test requests
time	Test time
threads	Number of test threads
percentile	The percentile range to be counted, which is 95% by default, i.e., the execution times of requests

	in 95% of the cases
report-interval	Interval for outputting a test progress report in seconds. 0 indicates to output only the final result but not the test progress report
skip-trx	Whether to skip transactions 1: Yes 0: No

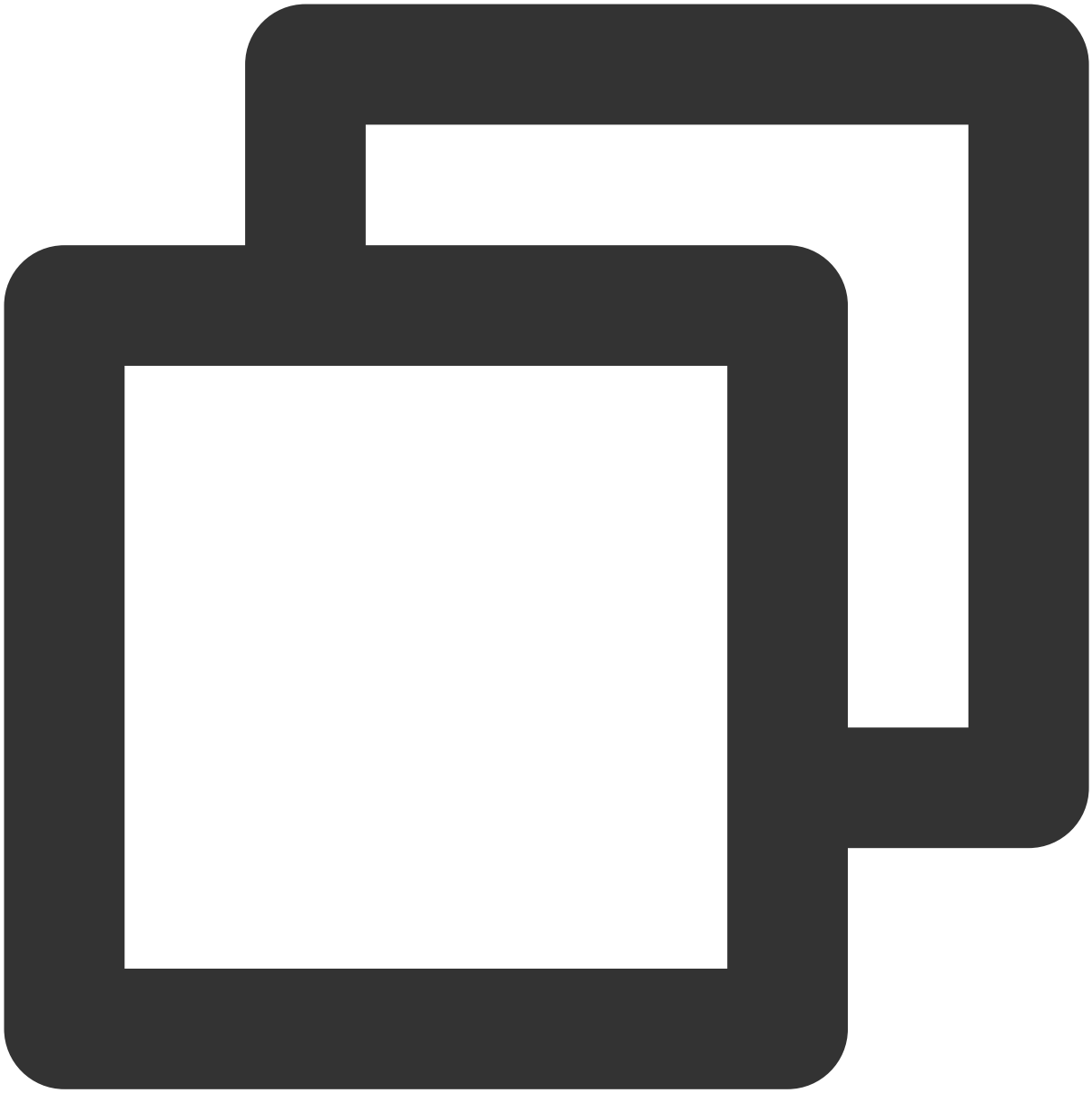
Installation method

This stress test uses SysBench 1.0.20 (using bundled LuaJIT 2.1.0-beta2). For more information, see here.

Note:

One client offers a concurrency of 1,000. One more client is added after the concurrency exceeds 1,000, and so on.

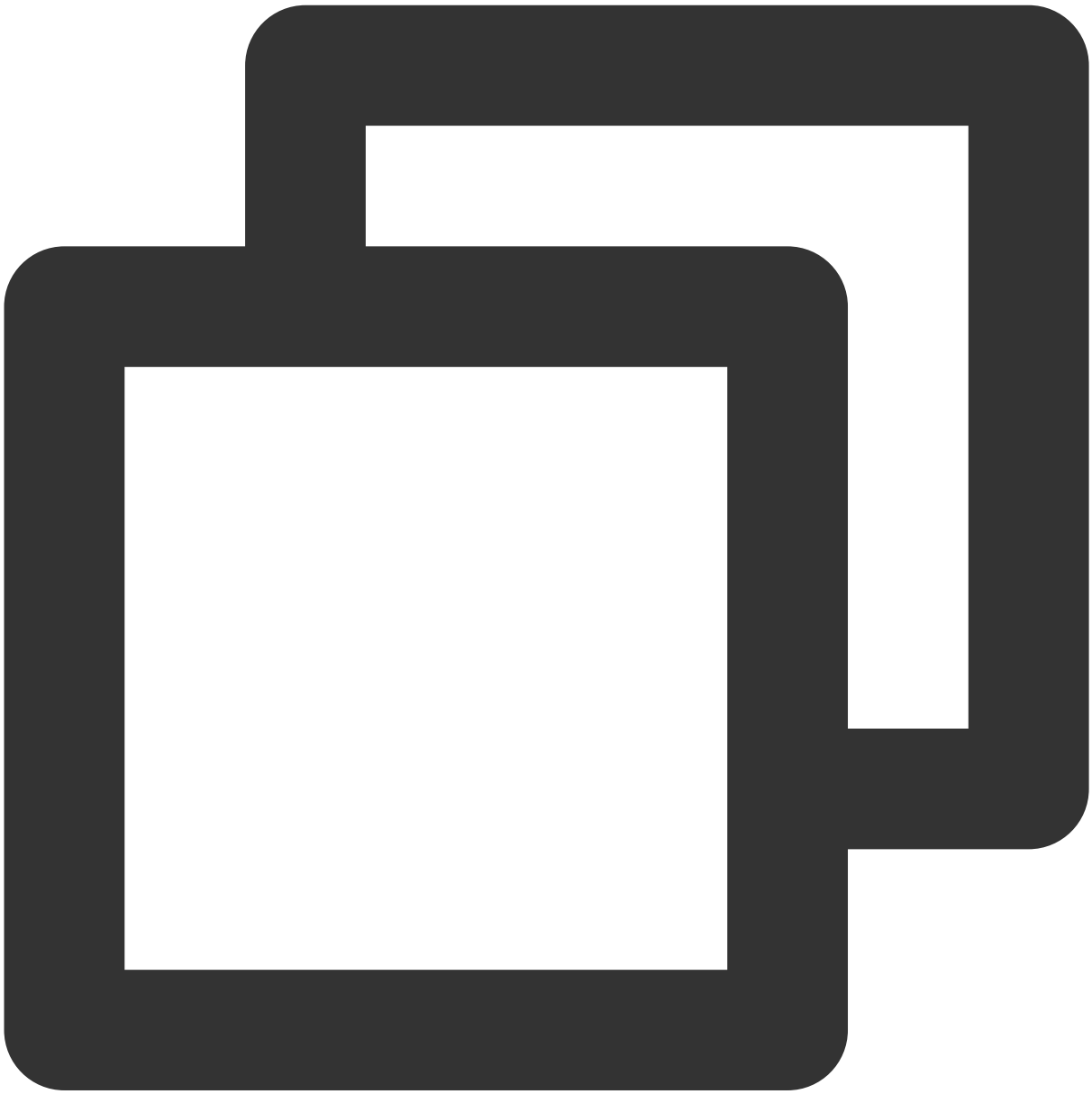
1. Run the following command to install SysBench in a CVM instance:



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

```
##Compile
make install
```

2. Run the following command to configure the client, so that the kernel can use all CPU cores to process data packets and reduce context switches within each CPU core.



```
sudo sh -c 'for x in /sys/class/net/eth0/queues/rx-*; do echo ffffffff>$x/rps_cpus;
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` indicates that 32 CPU cores are used (one `f` represents four CPU cores).

Test Methods

Last updated：2023-11-20 17:08:38

This document describes the method of TDSQL-C for MySQL performance test.

Scenario 1: Full Cache

List of table sizes and table quantities in the full cache test scenario:

Specification	Table Size (table_size)	Total Tables (tables)
2-core 16 GB MEM	25000	250
4-core 16 GB MEM	25000	250
2-core 32 GB MEM	25000	250
8-core 32 GB MEM	25000	250
8-core 64 GB MEM	25000	250
16-core 64 GB MEM	25000	250
16-core 96 GB MEM	25000	250
16-core 128 GB MEM	25000	250
32-core 128 GB MEM	25000	250
32-core 256 GB MEM	25000	250
64-core 256 GB MEM	25000	250

Command execution

Note:

Replace XXX in the following commands with the private network address, port number, username, user password, and database name of the tested TDSQL-C for MySQL cluster, as well as the `table_size` and `tables` corresponding to the test scenario. Specific parameters are as described below:

-host: Private network address of the tested instance

-port: Port number

-user: Username

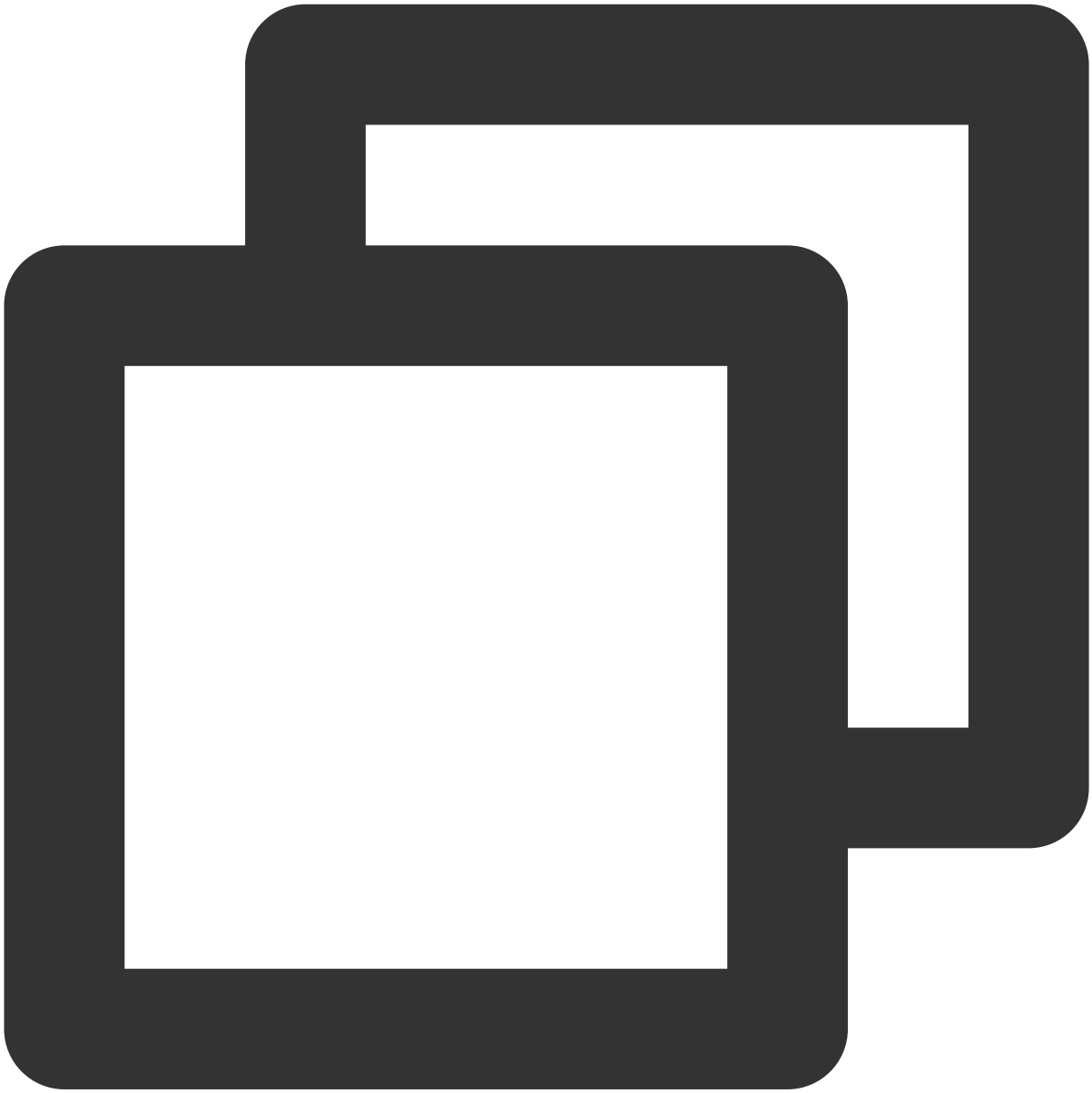
- password: Password of the username

-table_size: Data volume in one single table

-tables: Total number of tables

-mysql-db: Database name

1. Write

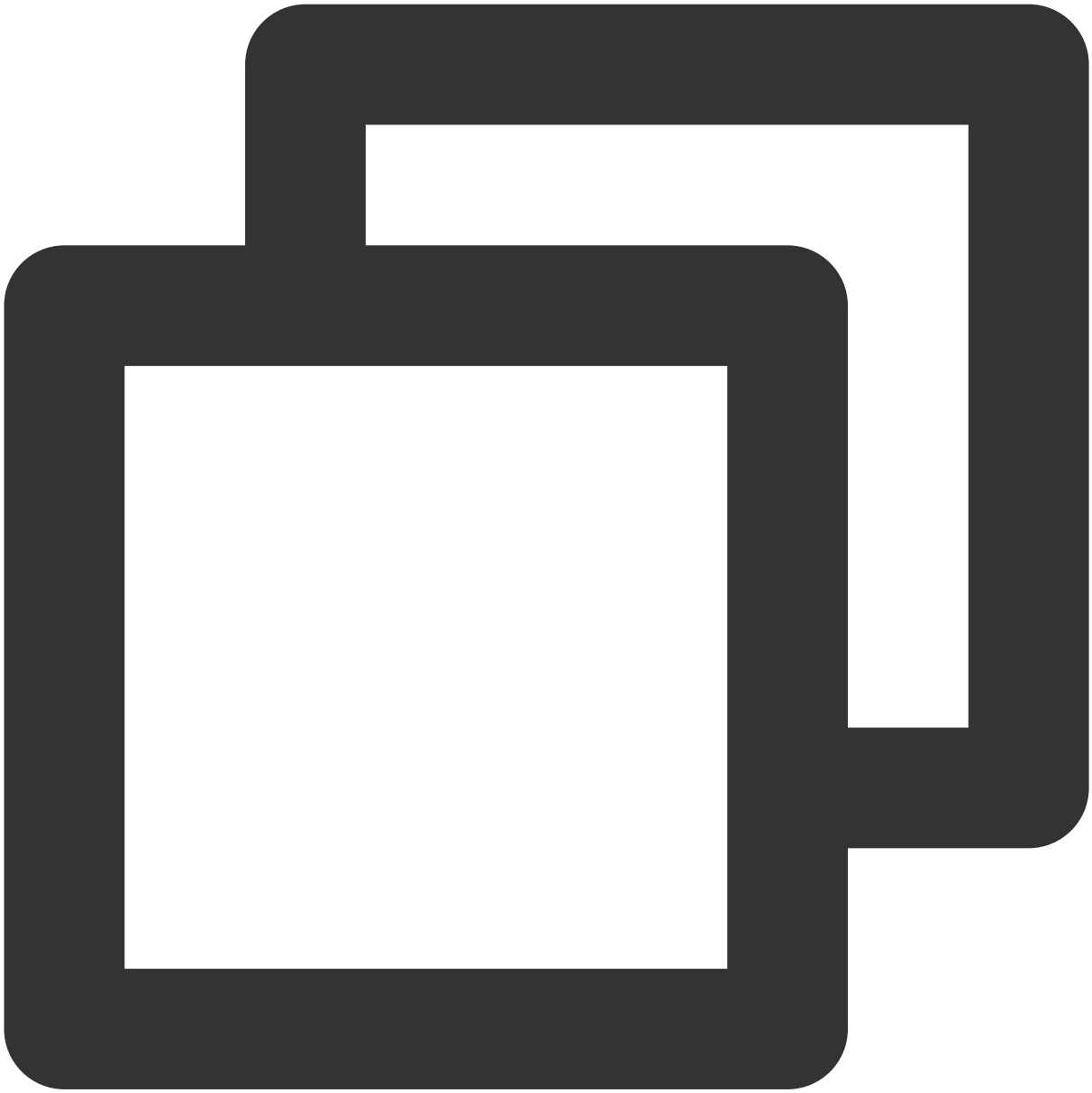


```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Prepare data

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Run the workload
```

```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Clear the data
```

2. Read (POINT SELECT)



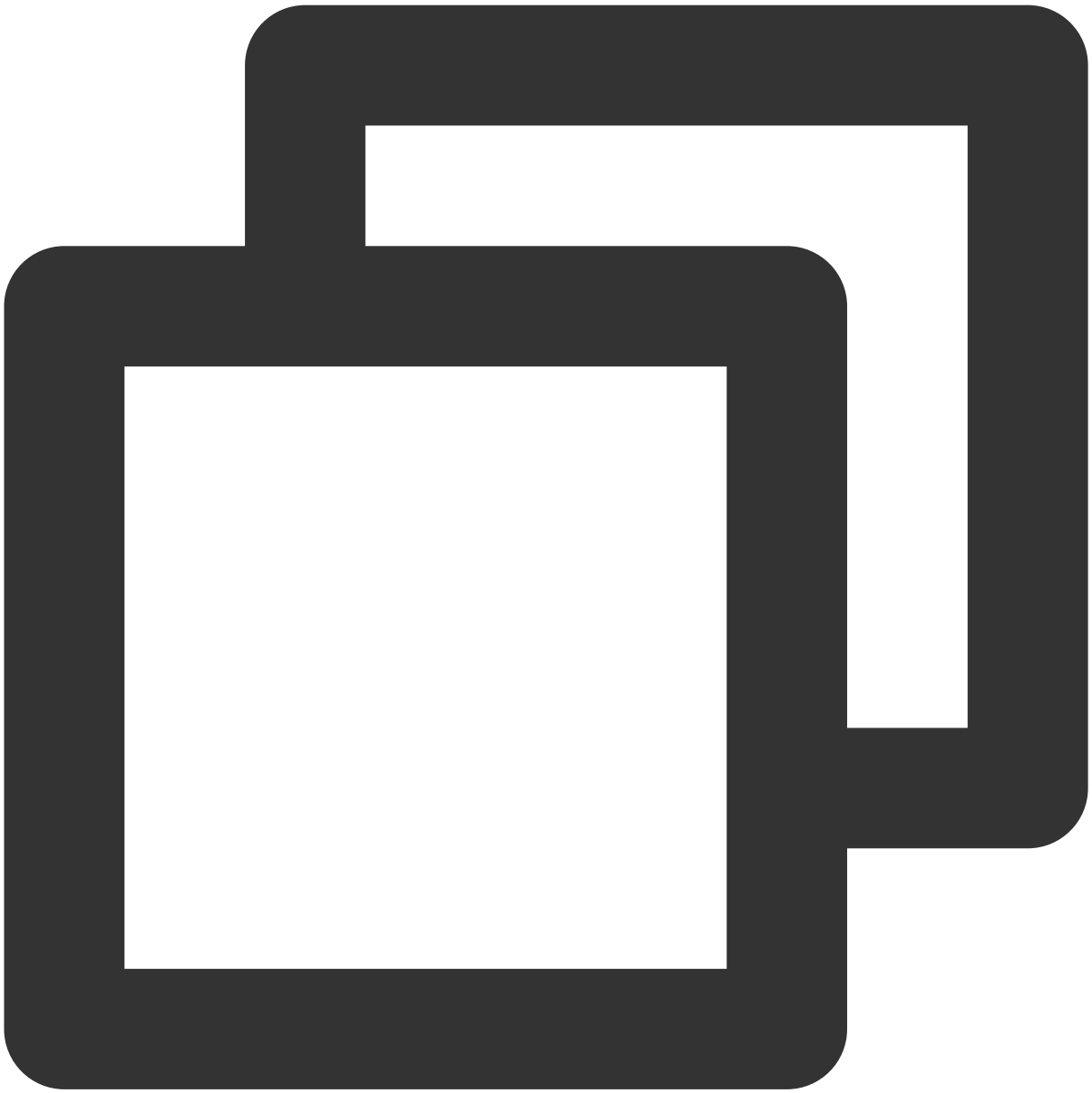
```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Prepare data

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Run the workload

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
```

```
## Clear the data
```

3. Read (RANGE SELECT)

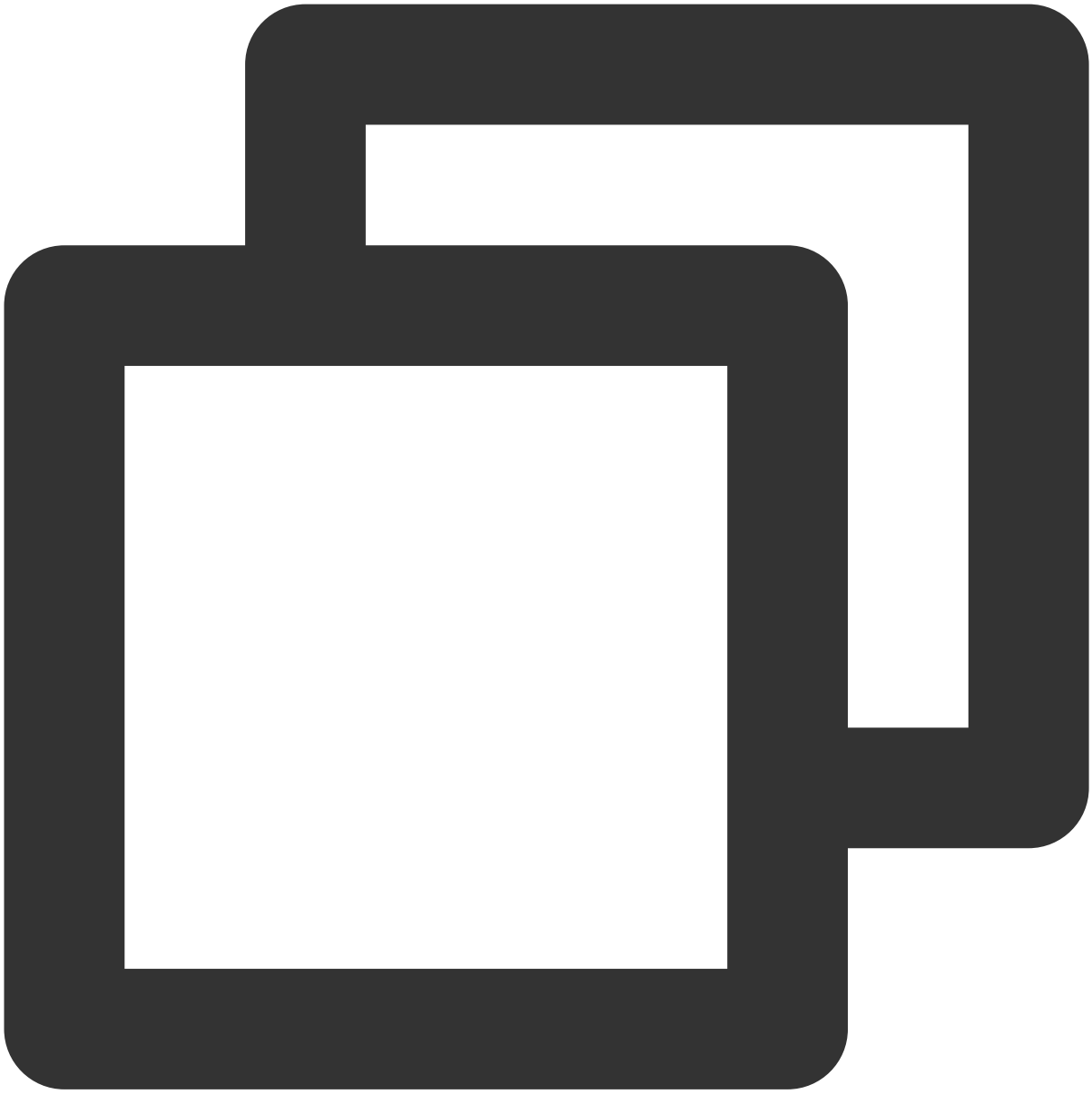


```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Prepare data

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Run the workload

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Clear the data
```

4. Read-write (POINT SELECT)

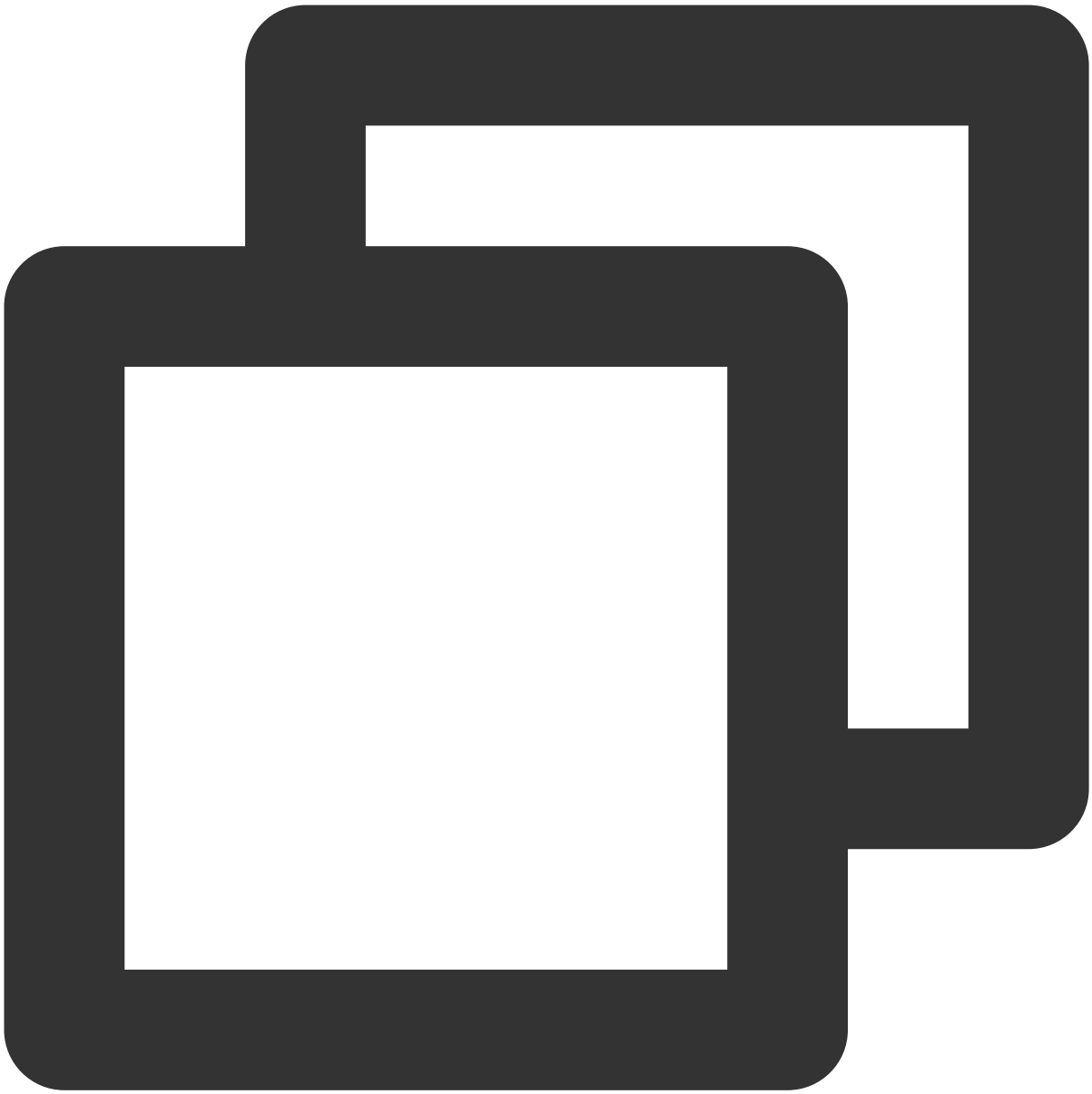


```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Prepare data

sysbench --db-driver=mysql  --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --my
## Run the workload

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Clear the data
```

5. Read-write (RANGE SELECT)



```
sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Prepare data

sysbench --db-driver=mysql  --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --my
## Run the workload

sysbench --db-driver=mysql --mysql-host=XXX --mysql-port=XXX --mysql-user=XXX --mys
## Clear the data
```

Scenario 2: Big Dataset

List of table sizes and table quantities in the big dataset test scenario:

Specification	Table Size (table_size)	Total Tables (tables)
2-core 16 GB MEM	800000	150
4-core 16 GB MEM	800000	300
4-core 32 GB MEM	800000	300
8-core 32 GB MEM	800000	300
8-core 64 GB MEM	800000	450
16-core 64 GB MEM	800000	450
16-core 96 GB MEM	800000	600
16-core 128 GB MEM	5000000	300
32-core 128 GB MEM	5000000	300
32-core 256 GB MEM	5000000	400
64-core 256 GB MEM	6000000	450

Command execution

The commands are the same as those in each full cache test scenario. You only need to replace the `table_size` and `tables` in the commands.

Scenario 3: 1 TB Single Table

List of table sizes and table quantities in the 1 TB single table test scenario:

Specification	Table Size (table_size)	Total Tables (tables)
2-core 16 GB MEM	4000000000	1
4-core 16 GB MEM	4000000000	1
4-core 32 GB MEM	4000000000	1
8-core 32 GB MEM	4000000000	1

8-core 64 GB MEM	4000000000	1
16-core 64 GB MEM	4000000000	1
16-core 96 GB MEM	4000000000	1
16-core 128 GB MEM	4000000000	1
32-core 128 GB MEM	4000000000	1
32-core 256 GB MEM	4000000000	1
64-core 256 GB MEM	4000000000	1

Command execution

The commands are the same as those in each full cache test scenario. You only need to replace the `table_size` and `tables` in the commands.

Test Metrics

Last updated：2023-05-31 15:01:45

This document describes the metrics of TDSQL-C for MySQL performance test.

Metric	Definition
QPS	The number of requests (queries) processed per second
Concurrency	The number of concurrent requests initiated by the client during performance testing

Test Results

Full Cache Scenario Test Results

Last updated：2022-04-13 12:04:10

This document lists the performance comparison test results between TDSQL-C for MySQL and TencentDB for MySQL in the full cache scenario.

Full Cache Scenario Overview

In the full cache scenario, as all data can be put into the cache, the disk doesn't need to be read and written to update the cache during queries.

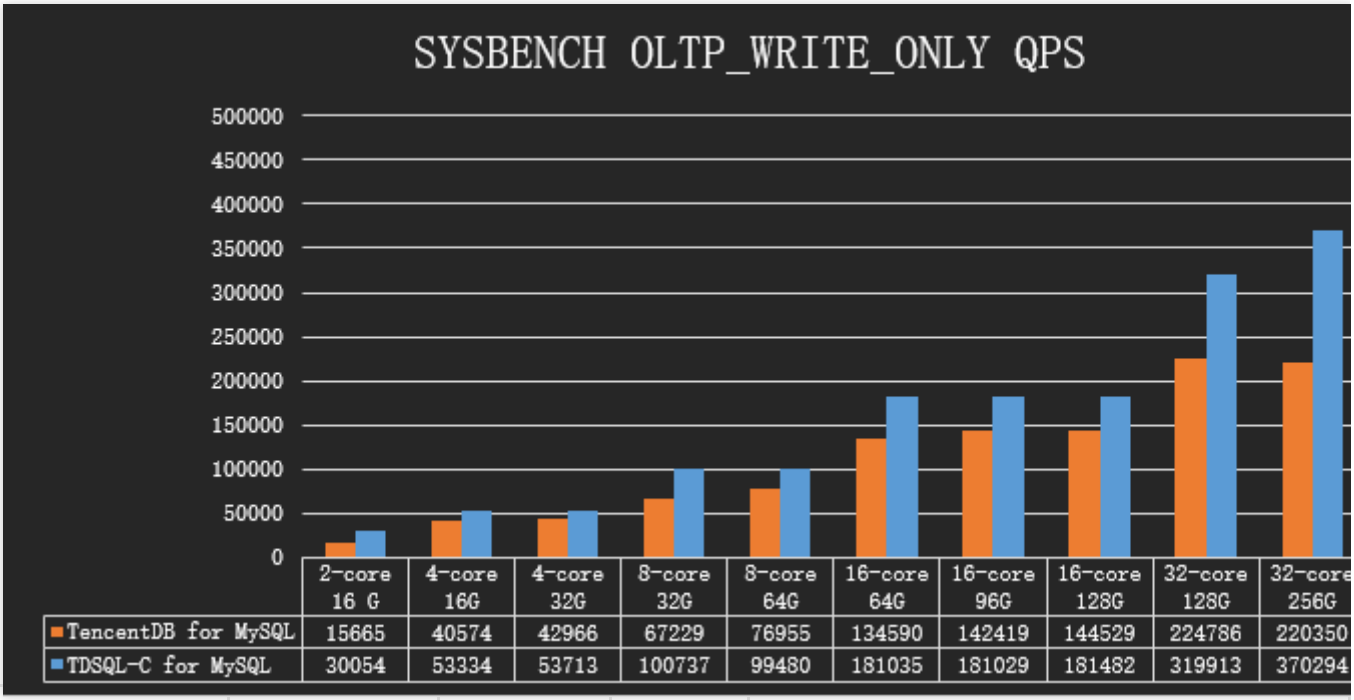
Full Cache Scenario Test Conclusion

The higher the instance specification, the more obvious the performance advantage of TDSQL-C for MySQL. TencentDB for MySQL's write and read-write performance reaches a bottleneck at the 32-core specification, but TDSQL-C for MySQL can further increase the QPS with more CPU cores.
In most scenarios, TDSQL-C for MySQL can achieve a CPU utilization of above 90% on compute nodes. The test shows that the resource utilization of TDSQL-C for MySQL is better than that of TencentDB for MySQL.
TDSQL-C for MySQL is more stable in terms of request delay (RTT) and almost doesn't jitter at all when the dataset is fully cached.

Dataset Characteristics	Test Scenario	Read Type	Conclusion
Full cache	Write	-	TDSQL-C for MySQL has a higher performance
	Read	POINT SELECT	TDSQL-C for MySQL has a higher performance
	Read	RANGE SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under low specifications, but the latter outperforms the former under high specifications
	Read-write	POINT SELECT	TDSQL-C for MySQL has a higher performance
	Read-write	RANGE SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under most specifications

Full Cache Scenario Test Results

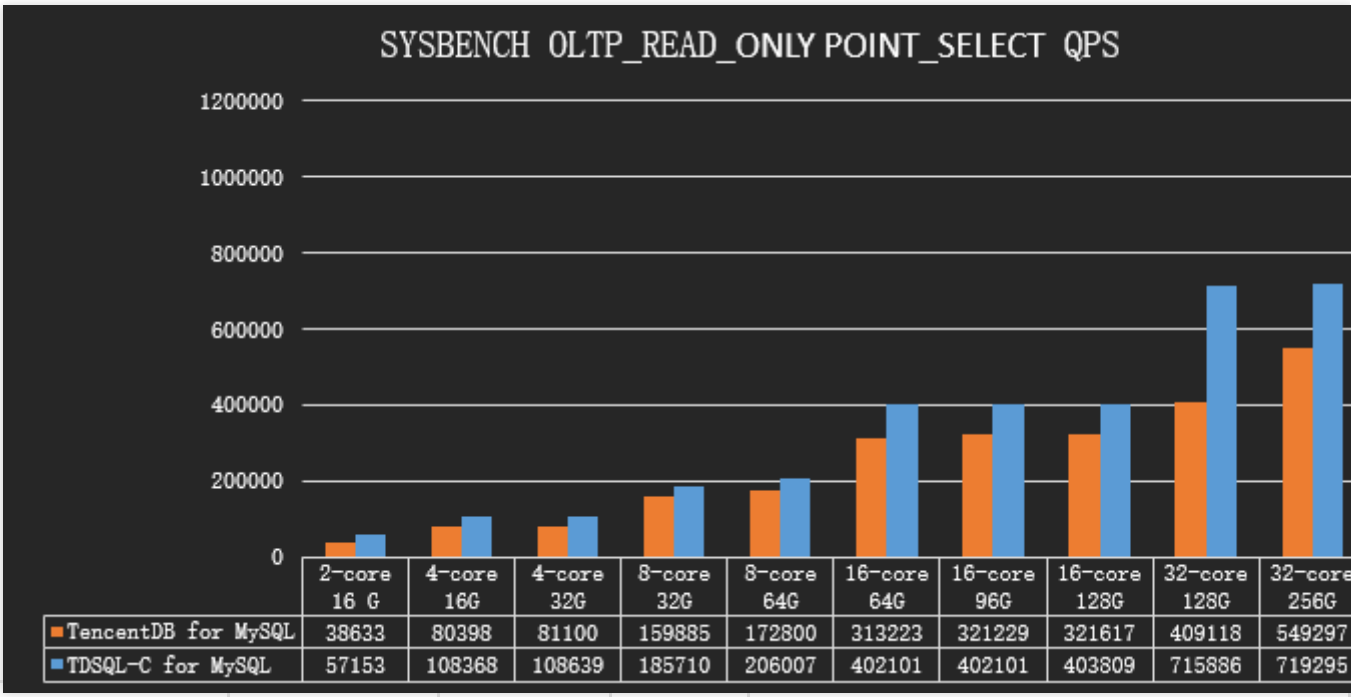
Scenario 1: Write



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	96	25000	250	15665	30054
4-core 16 GB MEM	192	25000	250	40574	53334
4-core 32 GB MEM	192	25000	250	42966	53713
8-core 32 GB MEM	256	25000	250	67229	100737
8-core 64 GB MEM	256	25000	250	76955	99480
16-core 64 GB MEM	512	25000	250	134590	181035
16-core 96 GB MEM	512	25000	250	142419	181029
16-core 128 GB MEM	512	25000	250	144529	181482

32-core 128 GB MEM	1000	25000	250	224786	319913
32-core 256 GB MEM	1000	25000	250	220350	370294
64-core 256 GB MEM	1000	25000	250	236079	448221

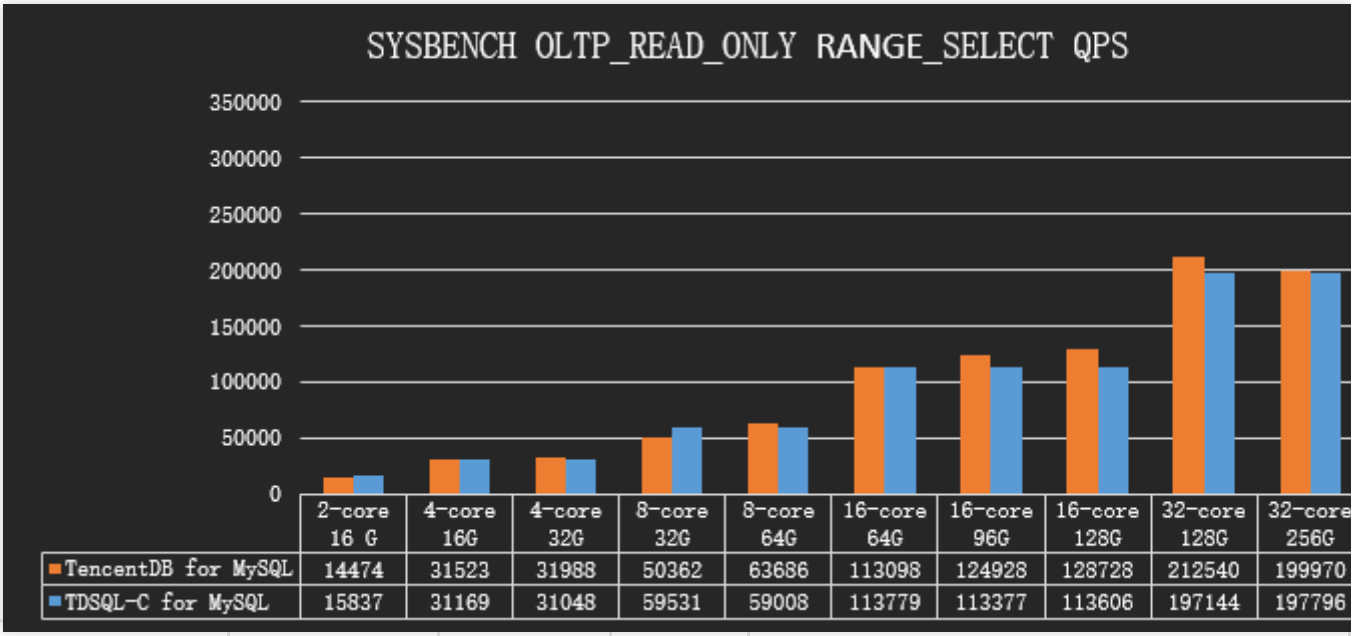
Scenario 2: Read (POINT SELECT)



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	1500	25000	250	38633	57153
4-core 16 GB MEM	1500	25000	250	80398	108368
4-core 32 GB MEM	1500	25000	250	81100	108639
8-core 32 GB MEM	1500	25000	250	159885	185710
8-core 64 GB MEM	1500	25000	250	172800	206007
16-core 64 GB MEM	2000	25000	250	313223	402101

16-core 96 GB MEM	2000	25000	250	321229	402101
16-core 128 GB MEM	2000	25000	250	321617	403809
32-core 128 GB MEM	2000	25000	250	409118	715886
32-core 256 GB MEM	2000	25000	250	549297	719295
64-core 256 GB MEM	2000	25000	250	670026	1125180

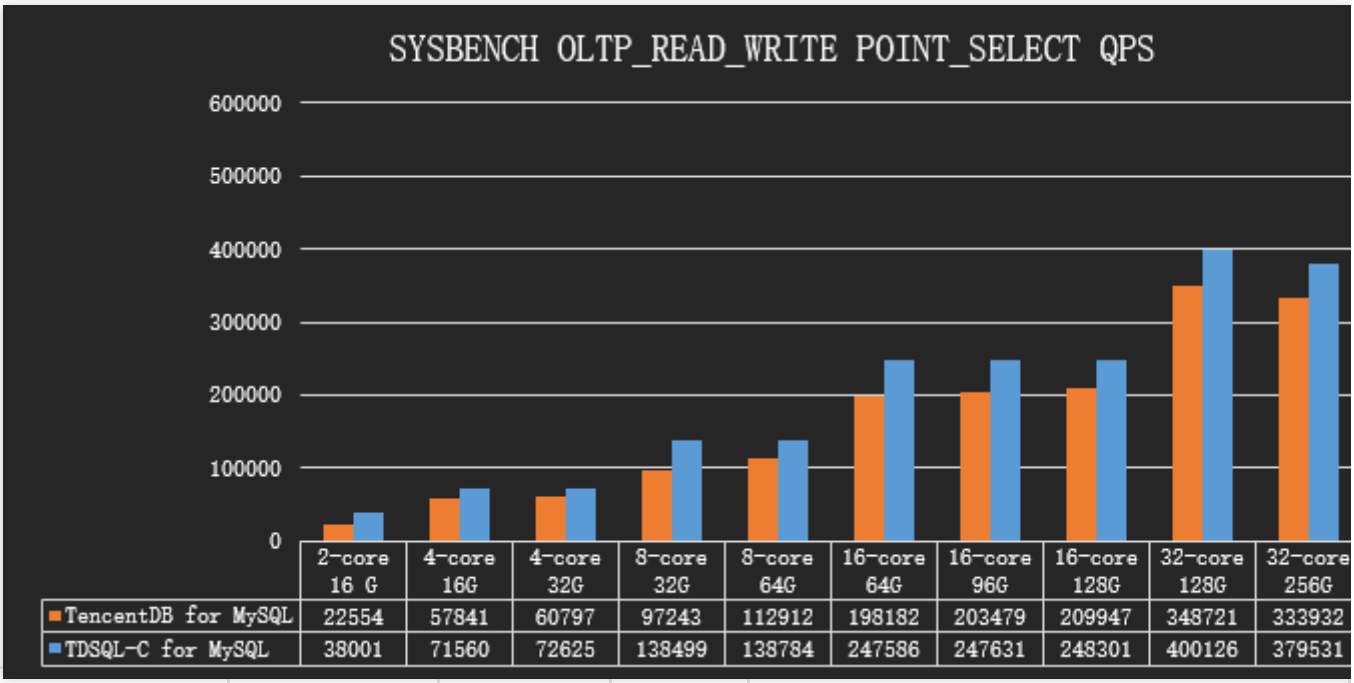
Scenario 3: Read (RANGE SELECT)



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	64	25000	250	14474	15837
4-core 16 GB MEM	64	25000	250	31523	31169
4-core 32 GB MEM	64	25000	250	31988	31048
8-core 32 GB MEM	64	25000	250	50362	59531
8-core 64 GB MEM	64	25000	250	63686	59008

16-core 64 GB MEM	128	25000	250	113098	113779
16-core 96 GB MEM	128	25000	250	124928	113377
16-core 128 GB MEM	128	25000	250	128728	113606
32-core 128 GB MEM	256	25000	250	212540	197144
32-core 256 GB MEM	256	25000	250	199970	197796
64-core 256 GB MEM	256	25000	250	304502	289460

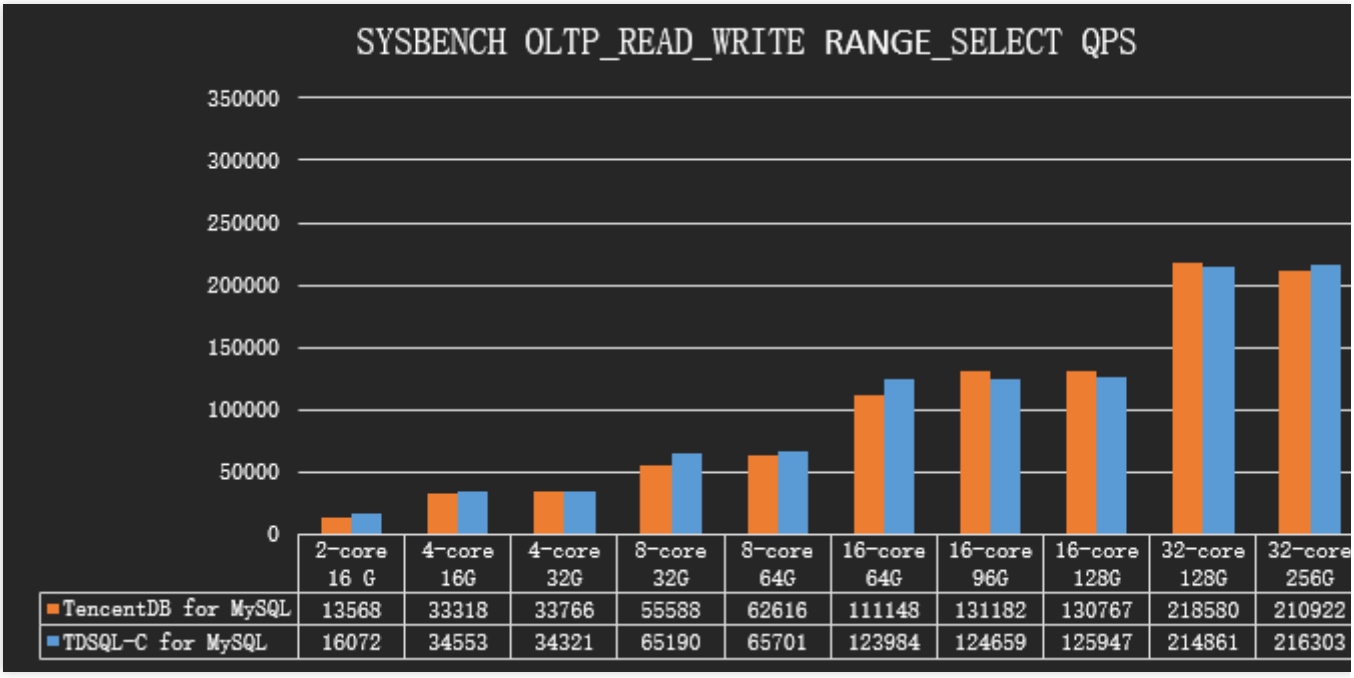
Scenario 4: Read-write (POINT SELECT)



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	512	25000	250	22554	38001
4-core 16 GB MEM	512	25000	250	57841	71560

4-core 32 GB MEM	512	25000	250	60797	72625
8-core 32 GB MEM	512	25000	250	97243	138499
8-core 64 GB MEM	512	25000	250	112912	138784
16-core 64 GB MEM	512	25000	250	198182	247586
16-core 96 GB MEM	512	25000	250	203479	247631
16-core 128 GB MEM	512	25000	250	209947	248301
32-core 128 GB MEM	512	25000	250	348721	400126
32-core 256 GB MEM	512	25000	250	333932	379531
64-core 256 GB MEM	512	25000	250	439984	553040

Scenario 5: Read-write (RANGE SELECT)



Specification	Concurrency	Table Size	Tables	QPS	

				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	64	25000	250	13568	16072
4-core 16 GB MEM	256	25000	250	33318	34553
4-core 32 GB MEM	256	25000	250	33766	34321
8-core 32 GB MEM	256	25000	250	55588	65190
8-core 64 GB MEM	256	25000	250	62616	65701
16-core 64 GB MEM	256	25000	250	111148	123984
16-core 96 GB MEM	256	25000	250	131182	124659
16-core 128 GB MEM	384	25000	250	130767	125947
32-core 128 GB MEM	384	25000	250	218580	214861
32-core 256 GB MEM	384	25000	250	210922	216303
64-core 256 GB MEM	384	25000	250	308399	312941

Big Dataset Scenario Test Results

Last updated：2023-11-20 17:29:52

This document lists the performance comparison test results between TDSQL-C for MySQL and TencentDB for MySQL in the big dataset scenario.

Big Dataset Scenario Overview

In the big dataset scenario, the entire volume of data cannot be all stored in the cache (the data volume is twice the memory size), and the disk needs to be read and written to update the cache during queries.

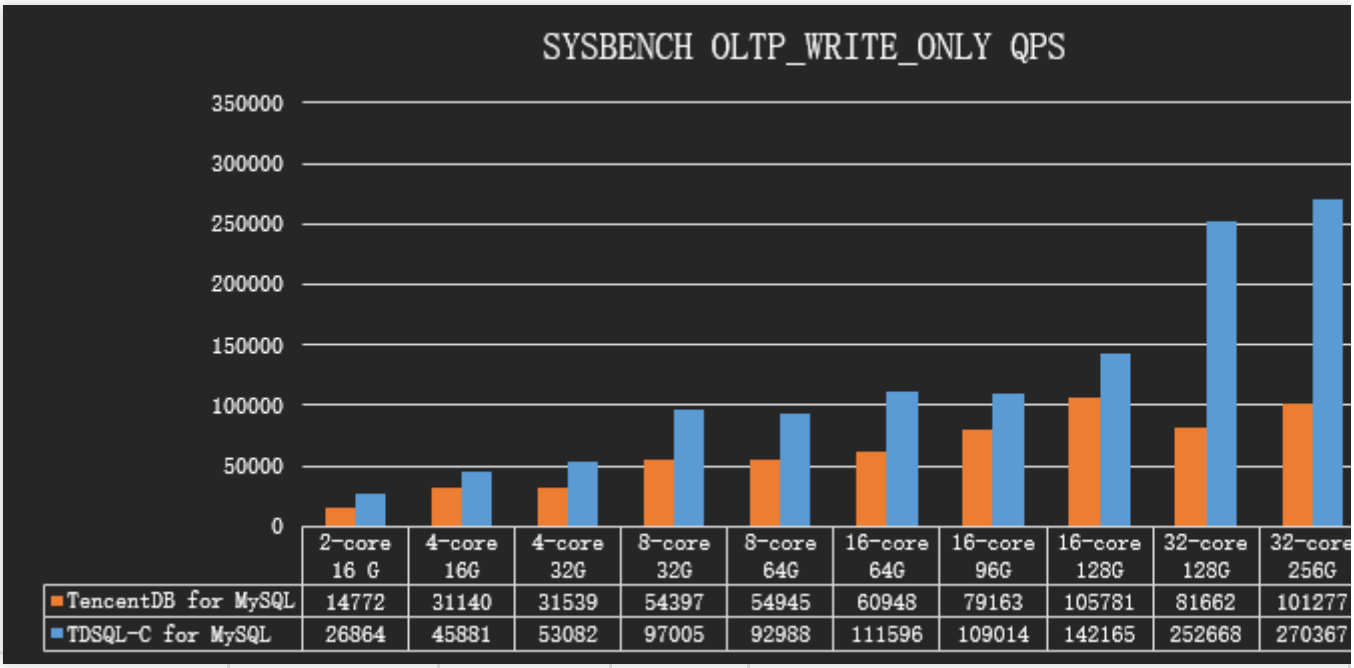
Test Conclusion for Big Dataset Scenario

In the read scenario, TDSQL-C for MySQL can achieve a CPU utilization of above 90% on compute nodes. The test shows that the resource utilization of TDSQL-C for MySQL is better than that of TencentDB for MySQL.
In the write scenario, TDSQL-C for MySQL outperforms TencentDB for MySQL under lower specifications, and as the specifications and data volume increase, its performance advantage gets much greater.

Dataset Characteristics	Test Scenario	Read Type	Conclusion
Big dataset	Write	-	TDSQL-C for MySQL outperforms TencentDB for MySQL, especially under high specifications
	Read	POINT SELECT	TDSQL-C for MySQL has a higher performance
	Read	RANGE SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under most specifications
	Read-write	POINT SELECT	TDSQL-C for MySQL has a higher performance
	Read-write	RANGE SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under most specifications

Test Results for Big Dataset Scenario

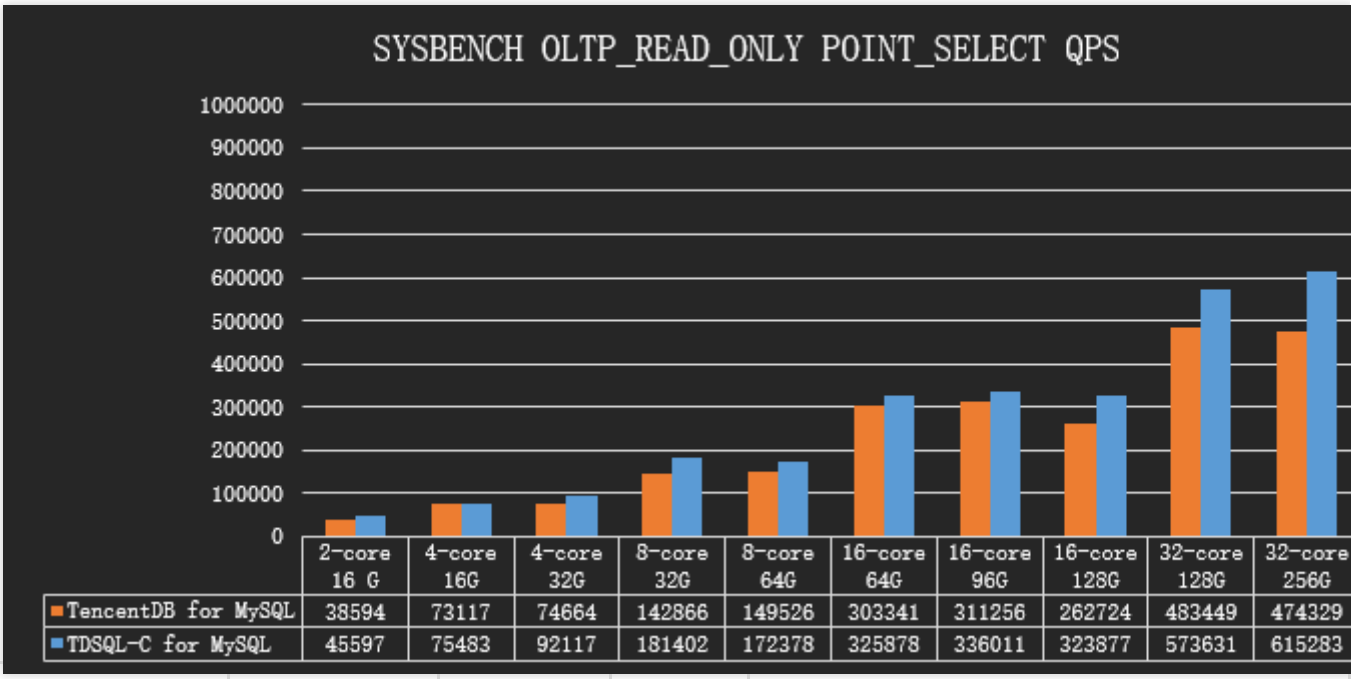
Scenario 1: Write



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	96	800000	150	14772	26864
4-core 16 GB MEM	96	800000	300	31140	45881
4-core 32 GB MEM	192	800000	300	31539	53082
8-core 32 GB MEM	192	800000	300	54397	97005
8-core 64 GB MEM	192	800000	450	54945	92988
16-core 64 GB MEM	192	800000	450	60948	111596
16-core 96 GB MEM	256	800000	600	79163	109014
16-core 128 GB MEM	384	5000000	300	105781	142165
32-core 128 GB MEM	384	5000000	300	81662	252668
32-core 256 GB MEM	384	5000000	400	101277	270367
64-core 256 GB	384	6000000	450	115992	301974

MEM					

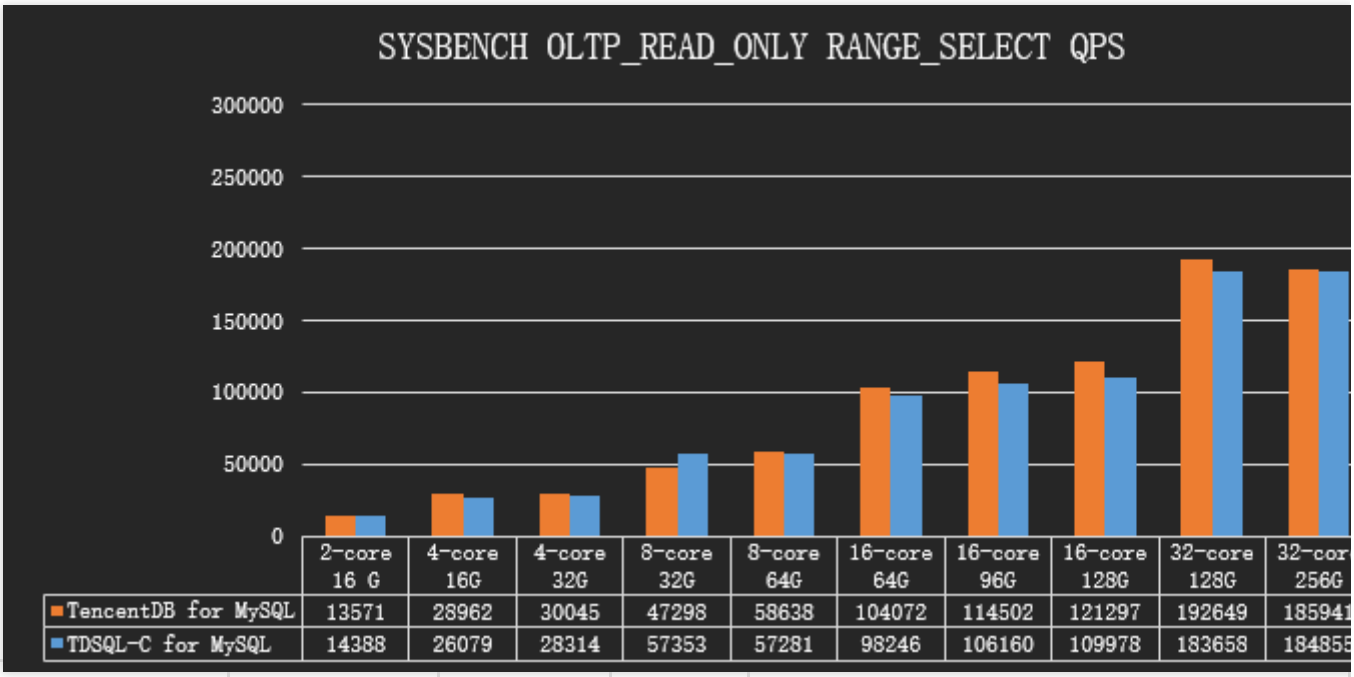
Scenario 2: Read



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	512	800000	150	38594	45597
4-core 16 GB MEM	512	800000	300	73117	75483
4-core 32 GB MEM	1000	800000	300	74664	92117
8-core 32 GB MEM	1000	800000	300	142866	181402
8-core 64 GB MEM	1000	800000	450	149526	172378
16-core 64 GB MEM	1000	800000	450	303341	325878
16-core 96 GB MEM	1000	800000	600	311256	336011
16-core 128 GB MEM	1000	5000000	300	262724	323877
32-core 128 GB MEM	1000	5000000	300	483449	573631

32-core 256 GB MEM	1000	5000000	400	474329	615283
64-core 256 GB MEM	1000	6000000	450	663715	940105

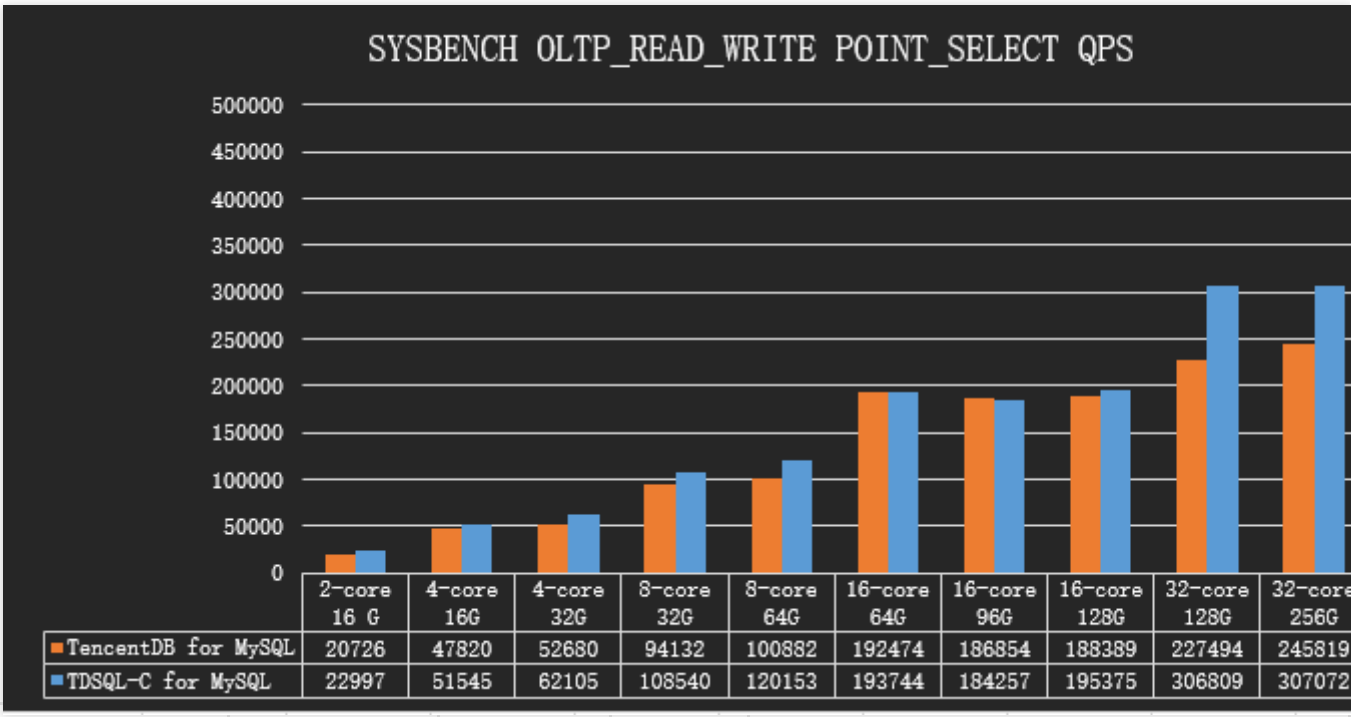
Scenario 3: Read



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	64	800000	150	13571	14388
4-core 16 GB MEM	64	800000	300	28962	26079
4-core 32 GB MEM	64	800000	300	30045	28314
8-core 32 GB MEM	64	800000	300	47298	57353
8-core 64 GB MEM	64	800000	450	58638	57281
16-core 64 GB MEM	128	800000	450	104072	98246
16-core 96 GB MEM	128	800000	600	114502	106160

16-core 128 GB MEM	128	5000000	300	121297	109978
32-core 128 GB MEM	256	5000000	300	192649	183658
32-core 256 GB MEM	256	5000000	400	185941	184855
64-core 256 GB MEM	256	6000000	450	283903	278997

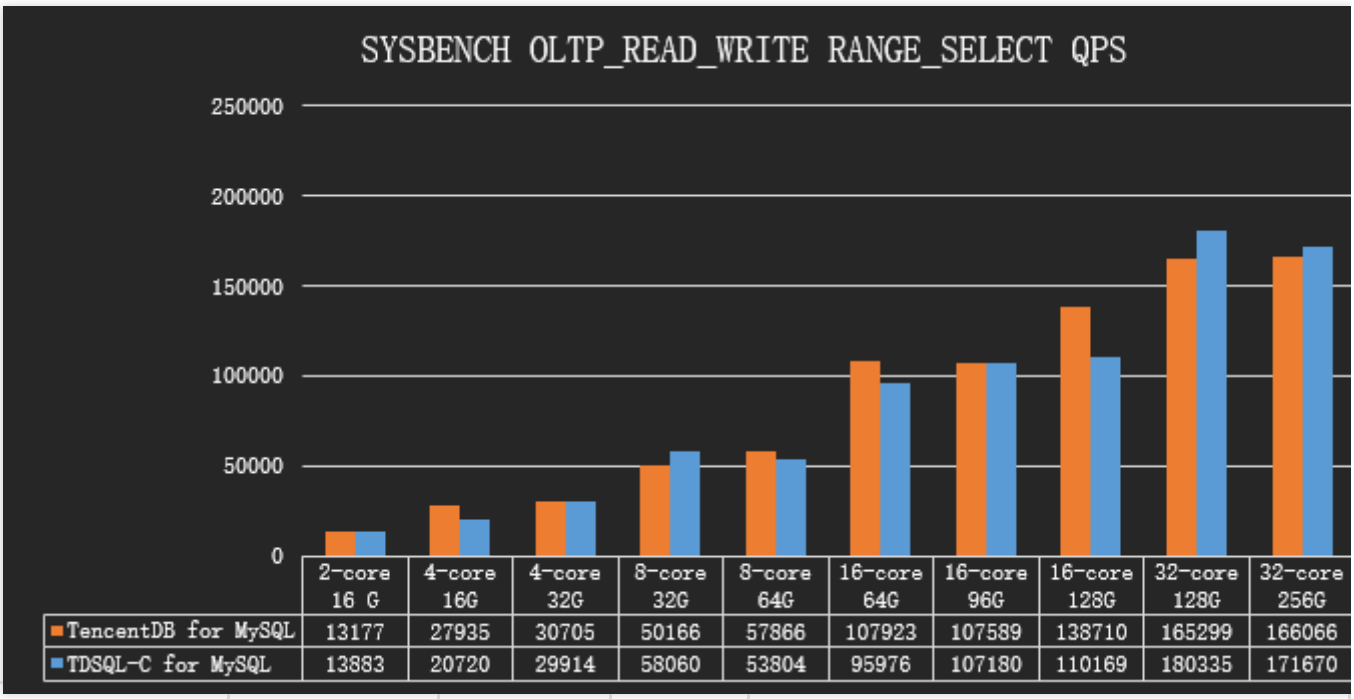
Scenario 4: Read-write



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	64	800000	150	20726	22997
4-core 16 GB MEM	256	800000	300	47820	51545
4-core 32 GB MEM	256	800000	300	52680	62105
8-core 32 GB MEM	256	800000	300	94132	108540
8-core 64 GB MEM	256	800000	450	100882	120153

16-core 64 GB MEM	256	800000	450	192474	193744
16-core 96 GB MEM	256	800000	600	186854	184257
16-core 128 GB MEM	512	5000000	300	188389	195375
32-core 128 GB MEM	512	5000000	300	227494	306809
32-core 256 GB MEM	512	5000000	400	245819	307072
64-core 256 GB MEM	512	6000000	450	335819	453163

Scenario 5: Read-write (RANGE SELECT)



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	32	800000	150	13177	13883
4-core 16 GB MEM	32	800000	300	27935	20720

4-core 32 GB MEM	64	800000	300	30705	29914
8-core 32 GB MEM	96	800000	300	50166	58060
8-core 64 GB MEM	64	800000	450	57866	53804
16-core 64 GB MEM	128	800000	450	107923	95976
16-core 96 GB MEM	128	800000	600	107589	107180
16-core 128 GB MEM	256	5000000	300	138710	110169
32-core 128 GB MEM	256	5000000	300	165299	180335
32-core 256 GB MEM	256	5000000	400	166066	171670
64-core 256 GB MEM	512	6000000	450	208616	190824

1 TB Single Table Scenario Test Results

Last updated：2023-11-20 17:44:29

This document lists the performance comparison test results between TDSQL-C for MySQL and TencentDB for MySQL in the 1 TB single table scenario.

Overview

In the 1 TB single table scenario, the test dataset contains 4 billion data records in one single table with a storage space of 1 TB.

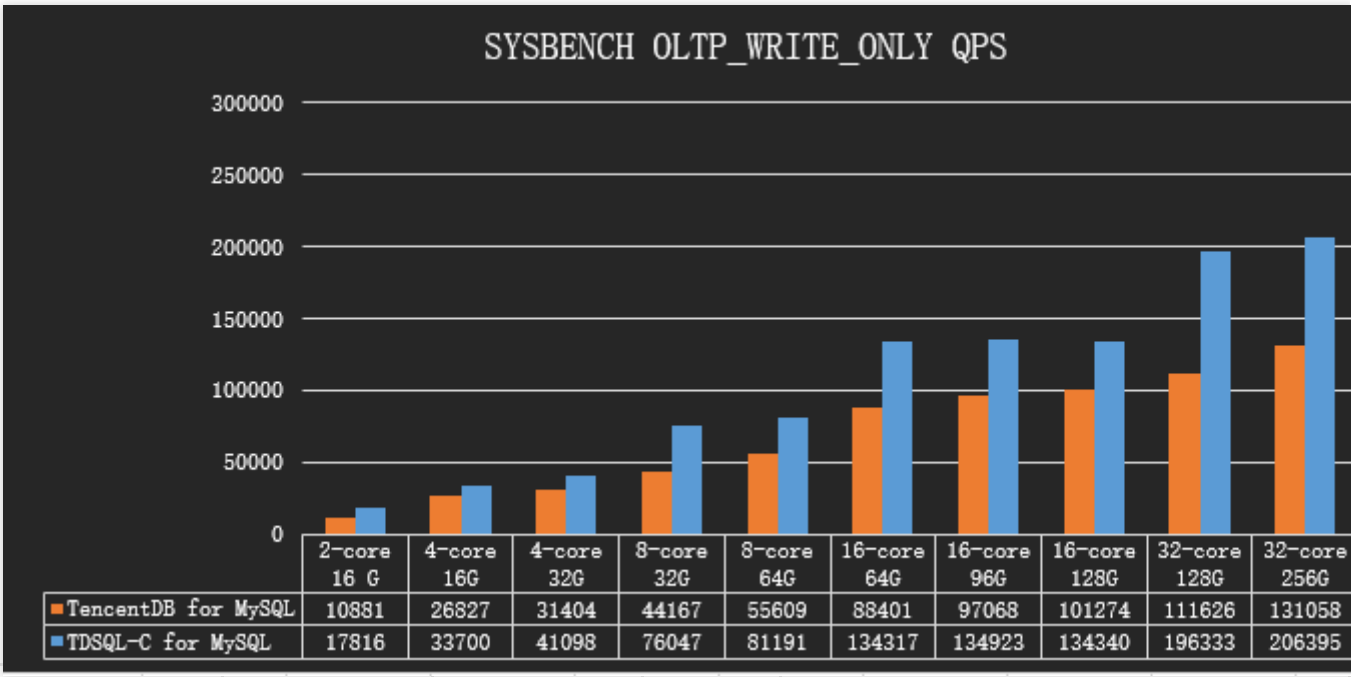
Test Conclusion

In the read scenario, TDSQL-C for MySQL can achieve a CPU utilization of above 90% on compute nodes. The test shows that the resource utilization of TDSQL-C for MySQL is better than that of TencentDB for MySQL.
In the write scenario, TDSQL-C for MySQL outperforms TencentDB for MySQL under lower specifications, and as the specifications and data volume increase, its performance advantage gets much greater.

Dataset Characteristics	Test Scenario	Read Type	Conclusion
1 TB single table	Write	-	TDSQL-C for MySQL has a higher performance
	Read	POINT SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under most specifications, but the former outperforms the latter under the maximum specification
	Read	RANGE SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under most specifications
	Read-write	POINT SELECT	TDSQL-C for MySQL has a higher performance
	Read-write	RANGE SELECT	TDSQL-C for MySQL and TencentDB for MySQL have generally the same performance under most specifications

Test Results

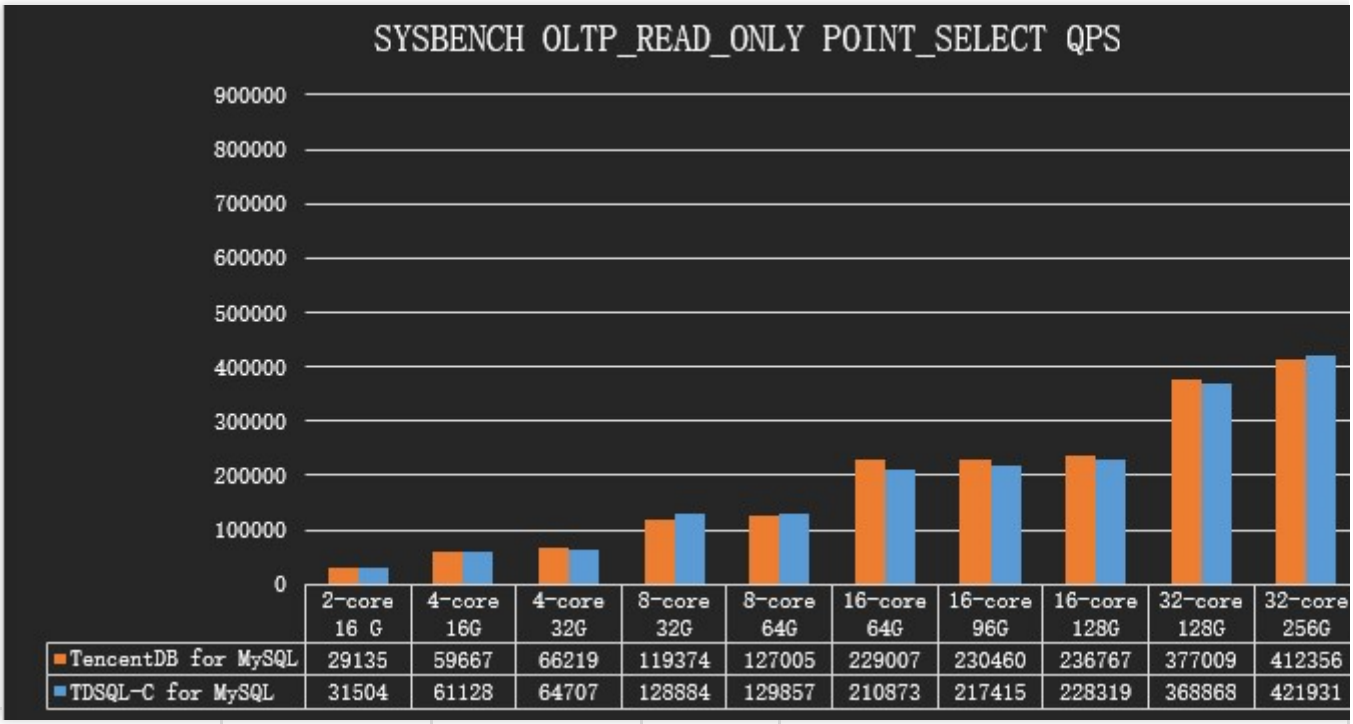
Scenario 1: Write



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	256	4000000000	1	10881	17816
4-core 16 GB MEM	512	4000000000	1	26827	33700
4-core 32 GB MEM	512	4000000000	1	31404	41098
8-core 32 GB MEM	512	4000000000	1	44167	76047
8-core 64 GB MEM	512	4000000000	1	55609	81191
16-core 64 GB MEM	512	4000000000	1	88401	134317
16-core 96 GB MEM	512	4000000000	1	97068	134923
16-core 128 GB MEM	512	4000000000	1	101274	134340
32-core 128 GB MEM	512	4000000000	1	111626	196333
32-core 256 GB MEM	512	4000000000	1	131058	206395
64-core 256 GB	512	4000000000	1	140587	256415

MEM					

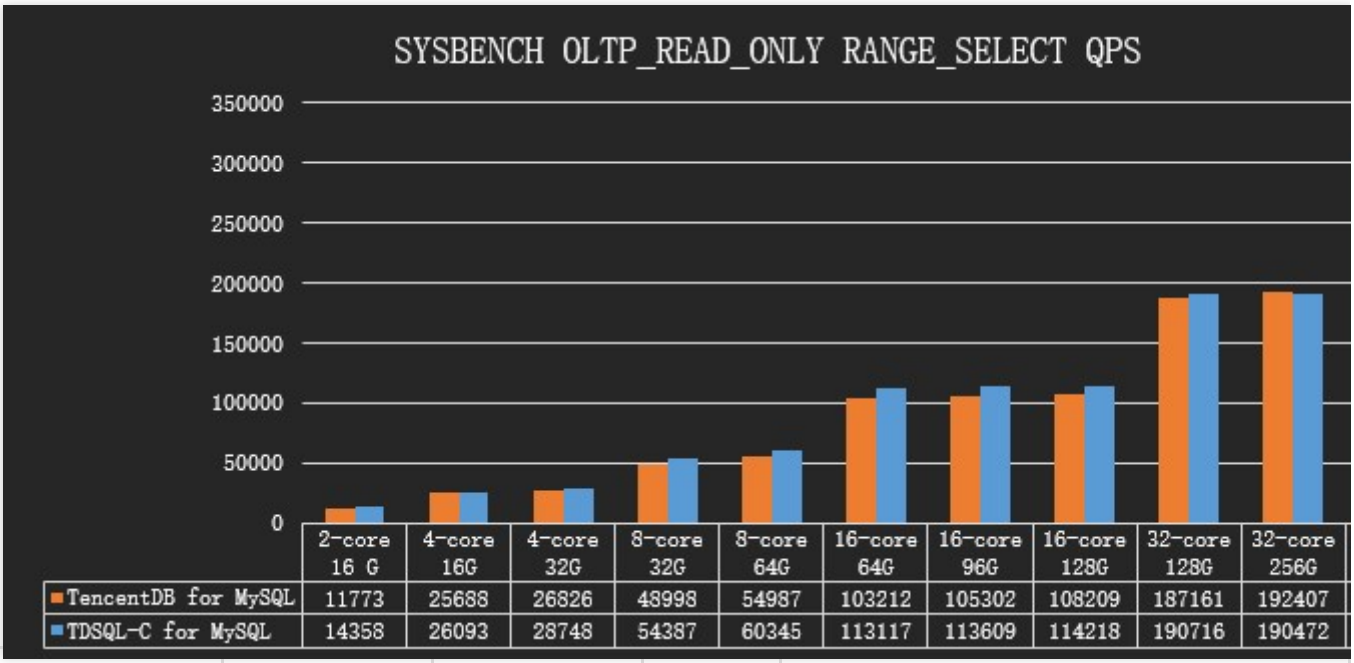
Scenario 2: Read



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	256	4000000000	1	29135	31504
4-core 16 GB MEM	512	4000000000	1	59667	61128
4-core 32 GB MEM	512	4000000000	1	66219	64707
8-core 32 GB MEM	512	4000000000	1	119374	128884
8-core 64 GB MEM	512	4000000000	1	127005	129857
16-core 64 GB MEM	1000	4000000000	1	229007	210873
16-core 96 GB MEM	1000	4000000000	1	230460	217415
16-core 128 GB MEM	1000	4000000000	1	236767	228319
32-core 128 GB	1000	4000000000	1	377009	368868

MEM					
32-core 256 GB MEM	1000	4000000000	1	412356	421931
64-core 256 GB MEM	1000	4000000000	1	569523	794997

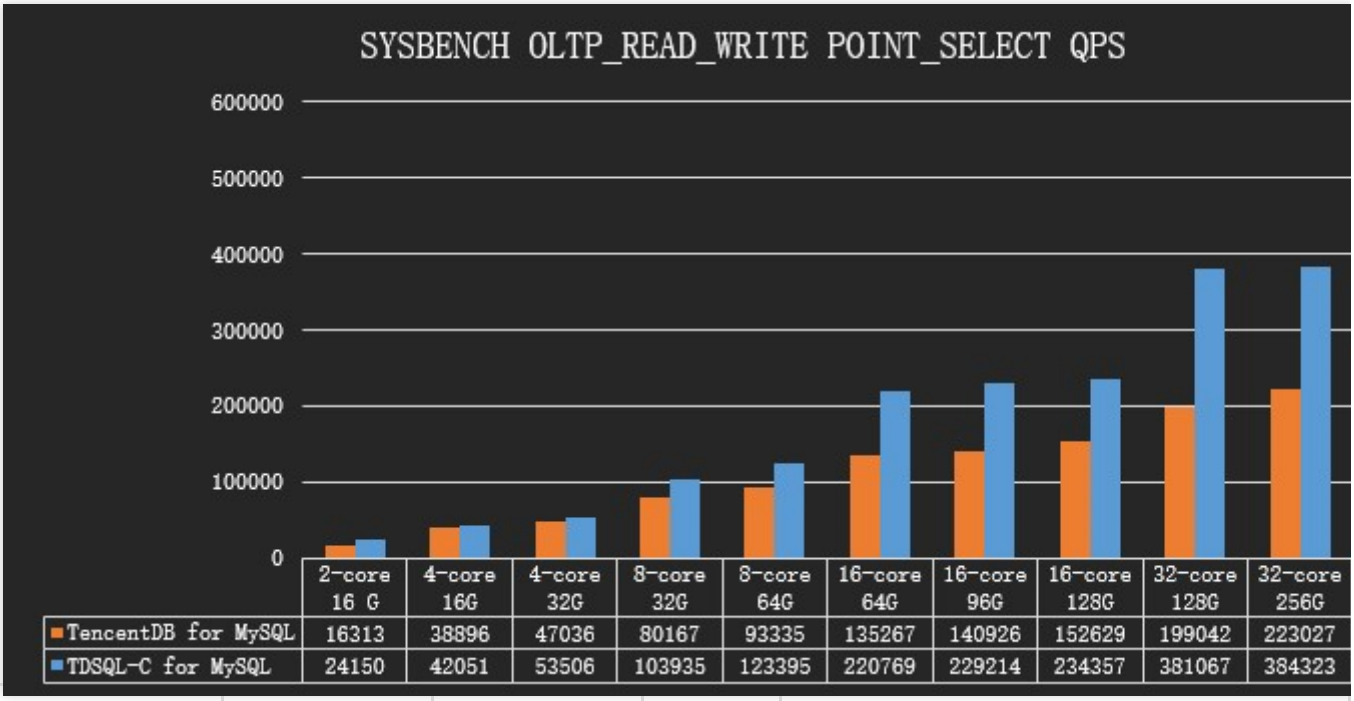
Scenario 3: Read



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	32	4000000000	1	11773	14358
4-core 16 GB MEM	64	4000000000	1	25688	26093
4-core 32 GB MEM	64	4000000000	1	26826	28748
8-core 32 GB MEM	128	4000000000	1	48998	54387
8-core 64 GB MEM	128	4000000000	1	54987	60345
16-core 64 GB MEM	256	4000000000	1	103212	113117
16-core 96 GB MEM	256	4000000000	1	105302	113609

16-core 128 GB MEM	256	4000000000	1	108209	114218
32-core 128 GB MEM	512	4000000000	1	187161	190716
32-core 256 GB MEM	512	4000000000	1	192407	190472
64-core 256 GB MEM	1000	4000000000	1	308631	319047

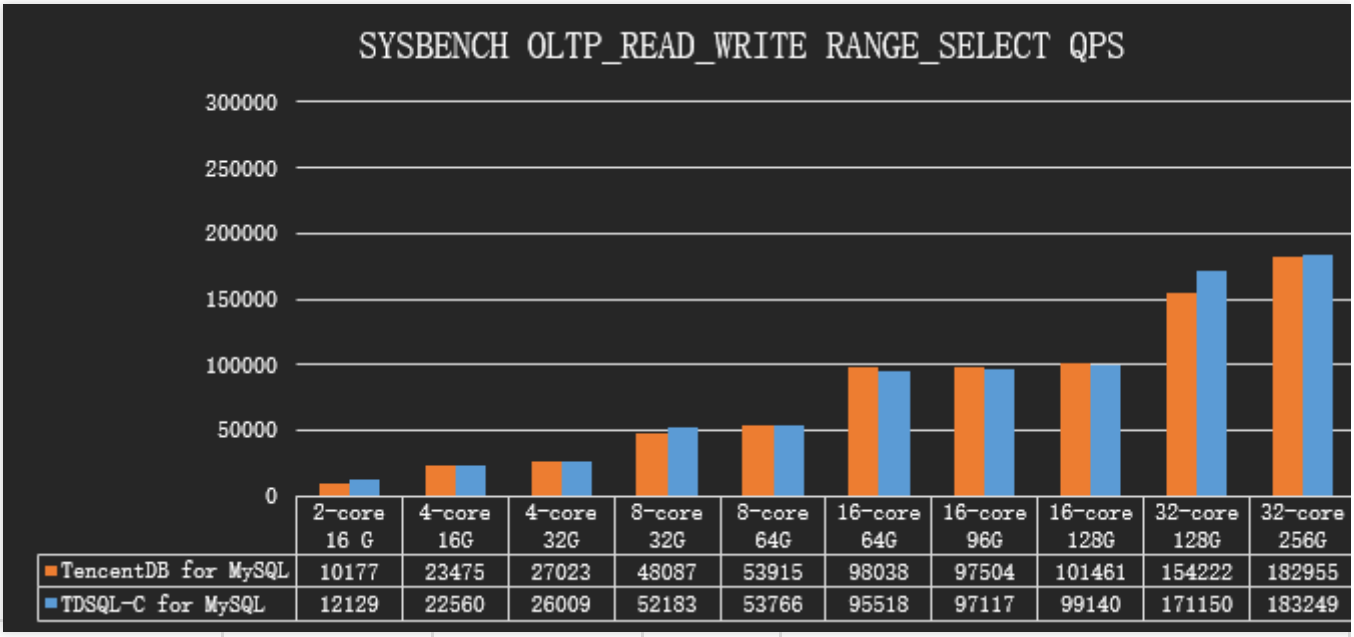
Scenario 4: Read-write



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	64	4000000000	1	16313	24150
4-core 16 GB MEM	64	4000000000	1	38896	42051
4-core 32 GB MEM	128	4000000000	1	47036	53506
8-core 32 GB MEM	256	4000000000	1	80167	103935
8-core 64 GB MEM	256	4000000000	1	93335	123395

16-core 64 GB MEM	512	4000000000	1	135267	220769
16-core 96 GB MEM	512	4000000000	1	140926	229214
16-core 128 GB MEM	512	4000000000	1	152629	234357
32-core 128 GB MEM	512	4000000000	1	199042	381067
32-core 256 GB MEM	512	4000000000	1	223027	384323
64-core 256 GB MEM	1000	4000000000	1	283722	520265

Scenario 5: Read-write (RANGE SELECT)



Specification	Concurrency	Table Size	Tables	QPS	
				TencentDB for MySQL	TDSQL-C for MySQL
2-core 16 GB MEM	256	4000000000	1	10177	12129
4-core 16 GB MEM	512	4000000000	1	23475	22560
4-core 32 GB MEM	512	4000000000	1	27023	26009

8-core 32 GB MEM	512	4000000000	1	48087	52183
8-core 64 GB MEM	512	4000000000	1	53915	53766
16-core 64 GB MEM	512	4000000000	1	98038	95518
16-core 96 GB MEM	512	4000000000	1	97504	97117
16-core 128 GB MEM	512	4000000000	1	101461	99140
32-core 128 GB MEM	512	4000000000	1	154222	171150
32-core 256 GB MEM	512	4000000000	1	182955	183249
64-core 256 GB MEM	512	4000000000	1	246526	266539