云数据库 MariaDB

开发指南

腾讯云

版权所有：腾讯云计算（北京）有限责任公司

【版权声明】

©2013-2025 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【商标声明】

腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

开发指南
- 功能限制
- 性能检测
- 同城双活解决方案
- Binlog 日志消费格式
- 慢查询分析
- 保留关键字
- 数据库审计
- 数据库审计已支持语法
- SEQUENCE

开发指南
功能限制

最近更新时间：2024-12-16 17:42:03

1. 不能更改 mysql、information_schema、performance_schema 和 sysdb 库中的任何数据。
2. 无法直接通过 SQL 语句进行账号和赋权相关操作，只能通过管理控制台进行。
3. 支持常见的19种权限，少量不常见权限不支持，具体支持的权限列表如下：
   - SELECT、INSERT、UPDATE、DELETE、CREATE、DROP、REFERENCES、INDEX、ALTER
   - CREATE TEMPORARY TABLES、LOCK TABLES、EXECUTE、CREATE VIEW、SHOW VIEW
   - CREATE ROUTINE、ALTER ROUTINE、EVENT、TRIGGER、SHOW DATABASES
4. 不提供 root 账号。
5. 建议使用 InnoDB 存储引擎 。

性能检测

最近更新时间：2023-02-07 11:24:19

简介

性能检测是针对数据库实例性能及运行状况综合分析服务，针对 SQL 语句的性能、CPU 使用率、IOPS 使用率、内存使用率、磁盘空间使用率、连接数、锁信息、热点表、事务等进行综合分析，性能检测提供了智能的诊断及优化功能，能最大限度发现数据库存在的或潜在的健康问题。

> **说明**
> 针对某些检测结论，性能检测报告提供了一系列的优化建议，请您在应用这些建议前谨慎测试，以防加重实例的性能问题。

功能介绍

健康评分：您可以看到当前数据库性能综合打分，满分100分；长期低于60分请注意优化业务或数据库配置。
报告生成、查看与保存：您可以任意创建报告，并查看最近一次生成的报告；报告可以通过另存网页的操作下载到本地查阅。
性能检测主要包含如下功能：

资源分析

分析一定时间内数据库实例资源（CPU、磁盘、连接）的使用情况，并综合评分概况。

> **说明**
> 由于多数实例默认开启了闲时超用策略，因此您可能观察到最大 CPU 使用率可能超过100%。如果您的 CPU 长期超过100%，且平均值高于建议值，建议您尽快扩容。

系统状态

梳理实例关键指标，列举其状态和出现时间，并提出对应修改建议。

表空间分布

列出按数据空间倒序的当前 TOP 10 表，协助您分析超大表的情况。

冗余索引检测

列出当前可能的冗余索引（区分度小于1%的冗余），并提出优化建议。

> **说明**

由于查询语句要先查询索引，再通过索引去查询表；如果索引列相同数据过多，不利于减少筛选的数据量，反而不如直接全表扫描性能快。

死锁诊断

死锁诊断通过诊断 `show engine innodb status` 获取数据库最后一次死锁信息，如果死锁发生时间在用户选择的诊断时间段内，便展示出来。

> **说明**
> 如果死锁出现频率过高，则代表事务内的 SQL 在并发执行场景中的持锁容易产生环路，根本解决方案是修改 SQL 运行逻辑顺序，优化加锁机制，减少死锁产生概率。临时解决方案是 kill 掉领头的阻塞会话。

锁等待诊断

当前时间段内的锁等待超过60s的报告。

> **说明**
> 有锁等待是正常现象，但有时候业务会出现 `Lock wait timeout exceeded;try restarting transaction` 锁等待超时等报错。MySQL 的 InnoDB 锁信息保存在系统库 `information_schema`中 的 `innodb_trx`，`innodb_lock_waits`，`innodb_locks` 三张表下，锁等待诊断通过分析诊断实例中的三张表的锁依赖关系，找出持有锁时间超过一定阈值并阻塞的其他会话的领头事务信息和会话信息，以及被阻塞事务的会话信息，并 kill 掉该领头会话。

> **注意**
> 当前锁等待只支持 InnoDB 引擎。

长会话诊断

通过诊断实例中的 `information_schema.processlist` 获取 Command 不为 Sleep 但执行时间（Time）超过10s的会话。

> **说明**
> 最佳解决长会话的手段是优化 SQL，并在业务代码中主动植入 session 失效配置；当然，您也可以通过调整 `interactive_timeout` 、 `wait_timeout` 两个参数，让过期 session 主动失效。

慢查询分析

基于执行次数倒序，列出当前 TOP 20 的慢查询语句。

> **说明**

> 慢查询可以通过 `long_query_time` 配置调整；慢查询产生的原因产生较多。通常，如果您的实例消耗资源合理且慢查询较多，则建议您关注业务 SQL、索引是否合理；如果您实例消耗性能较高且慢查询较多，建议您关注实例配置是否合理，并优化业务 SQL、索引等。慢查询数据可以在慢查询分析功能下，查询更多详细数据。

DB 状态检查

检查当前数据库 DB 层的健康状态。

其他

列出需要 DBA 关注的其他值。

同城双活解决方案

最近更新时间：2024-12-16 17:42:03

TDSQL MySQL 版集中式数据库（以下简称为 TDSQL）目前已经支持同城双中心双活能力，主要特性如下：

- 同城双中心部署
- 双中心可写：如果您的服务器部署在两中心不同子网中，可分别从两中心各自的服务器连接数据库写入数据。
- 故障自动转移/恢复
- 双中心唯一访问 IP

然而，仅是数据库同城双中心双活并不能实现业务系统级的容灾；事实上，单一系统/模块切换到同城灾备中心容易，但企业级系统业务内部复杂的关联、配置都是双中心面临的难题。
因此，构建双活业务系统，需要业务在系统的设计、使用、管理、系统升级过程中时刻都以双中心为基础，双中心实时使用，配置互通为基础，这样才能做到故障后，业务不修改或较少修改，即可快速恢复运行。这也是，腾讯云数据库 TDSQL 同城双中心双活设计的目标，让两个中心的业务系统都能通过本地网络，完全正常读写数据库系统，且能够保证数据库强一致性。

设计标准

腾讯云数据库 TDSQL 双活的设计标准参考《GB/T 20988-2007 信息安全技术 信息系统灾难恢复规范》，由于是数据库单一模块：

- RTO ≤ 60秒
- RPO ≤ 5秒
- 故障切换时间 ≤ 5秒
- 故障检测时间 ≤ 30秒

这意味着，含故障检测时间从故障发生到完全切换完成约需40秒。
风险提示：真实环境测试下，需确保业务系统具有自动重连数据库机制；然而业务系统往往存在多个模块，每个模块可能与多个数据源相关，因此越是复杂系统恢复时间越长。

支持情况

已支持情况

实例版本：

- 标准版：一主一从（双节点）/一主二从（三节点）
- 金融定制版：一主一从（双节点）/一主二从（三节点）

网络要求：仅 VPC 网络
已支持地域：

- 北京（北京一区、北京三区）
- 上海金融（金融一区、金融二区）
- 深圳金融（金融一区、金融二区）

价格说明

双可用区售价与单可用区相同，详见 价格说明 。

购买与使用

请至 云数据库 TDSQL 购买页 单击购买。

- 主、从可用区相同时，即单可用区部署。
- 主、从可用区不同时，即同城双中心部署。

> **注意：**
> - 主可用区即您主服务器所在区域，数据库应尽量分配在于主服务器相同的 VPC 子网内，减少访问延迟。从可用区，数据库从节点所在可用区；如果是一主二从3节点，主可用区将部署2个节点；如果是一主一从2节点，主可用区将部署1个节点。
> - 金融云围笼方案如需支持同城双中心策略，可能需要先建设同城双中心围笼方案，详情请咨询您对应商务、架构师。

查看实例可用区详情

您可至 TDSQL 控制台 ，单击实例 ID 或**操作**列的**管理**，进入实例详情页查看。

主从切换

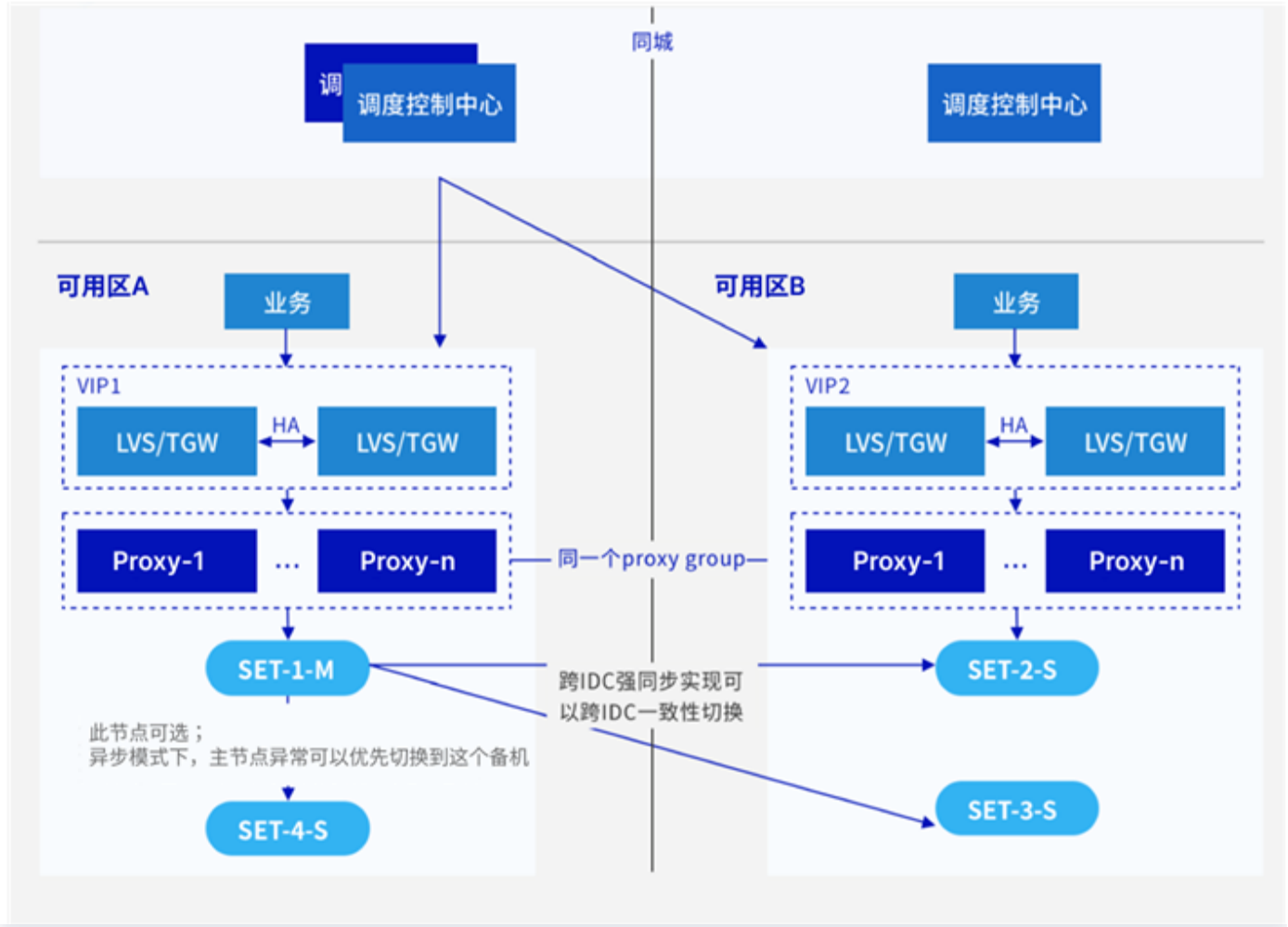
如果您要将主节点从某一可用区切换到另一可用区，您可以直接单击**主从切换**即可。主从切换是高危操作，需求验证登录账号的 IP 地址；切换过程可能导致数据库连接闪断（≤1s），请保证业务有数据库重连机制；频繁切换将可能业务系统异常甚至数据异常。

技术原理简介

基于 TDSQL 高可用的主从架构和 VPC 可用区虚拟 IP 漂移特性的有机整合，实现了双中心同时读写，架构特点如下：

- TDSQL 每一个 DB 节点前端，混合部署 Proxy 模块；Proxy 模块负责将数据请求路由到对应的 DB 节点。

- 在 Proxy 模块前部署跨地域 VPC 网关，并支持虚拟 IP 漂移功能。



如上图，以写入数据为例，假设业务服务器部署在 A 可用区，PC 网关转发数据请求到 A 可用区的 Proxy 网关，再由 Proxy 做透明转发到 Master 节点。而假设业务服务器部署在 B 可用区，VPC 网关转发数据请求到 B 可用区的 Proxy 网关，再由 Proxy 做透明转发（通过腾讯云 BGP 专网）到 Master 节点。
无论是读请求还是写请求，整个过程对业务透明。如果是数据库异常，数据库集群则按如下原则处理：

- 若 Master 与 Proxy 故障，集群自动选举最优 slave 提升为新 Master，系统通知 VPC 修改虚 IP 与物理 IP 关联关系，业务仅感知部分写连接中断。
- 若 Master 故障，但 Proxy 正常，集群自动选举最优 slave 提升为新 Master，Proxy 将阻塞请求，直到主从切换完成；此时，业务仅感知到部分请求超时。
- 若 Slave 故障（无论 Proxy 的是否故障），读写分离情况下，根据预先配置的只读账号**只读策略**（有3种）执行。
- 若 A 可用区完全故障，VPC、数据库在 B 可用区仍然存活，此时，slave2 节点自动提升为 Master 后，**根据强同步策略调整该节点读写策略**，VPC 网络 IP 漂移到 B 可用区；此时集群将重试恢复 A 可用区节点，如果超过30分钟无法恢复，将自动在 B 节点重建至少1个 Slave 节点。由于有 IP 漂移策略，业务不需要修改数据库配置。
- 假设 B 机房完全故障，TDSQL 集群相当于故障了 Slave 节点，处理方法与第3.相同。

常见问题

相对于同单中心，同城双中心是否会导致性能下降?

在基于强同步复制方案下，由于跨中心延迟会略大于同机房内设备，因此 SQL 响应会有理论下降5%左右。

是否存在 Master 节点从主可用区切换到从可用区?

会的，如果不影响您的业务使用，可不用管它。如果您担心影响，可在业务低谷期，通过控制台主从切换功能，切换回去。

如何知道数据库集群做了主从切换?

请到 腾讯云可观测平台控制台 告警策略 > 云数据库 MariaDB > 配置主从切换告警。

如果一部分的读写请求读写到从可用区，因为网络延迟原因导致性能下降，但又不想放弃同城双中心这个特性怎么办?

您可以 提交工单 说明实例 ID 和您服务器在可用区部署方案，以及读写请求比例，可由腾讯云 DBA 帮您调整双可用区负载机制，让从可用区的承担的读写请求尽量少些。

如果期望从单中心更换为同城双中心架构，应该如何操作?

首先确认您所在区域是否支持同城双中心方案，目前已开放北京、上海金融、深圳金融三个地域；其次，您需要提交工单，说明需要调整的账号信息、实例 ID、计划哪两个可用区、以及建议运维操作时间；最后，腾讯云工作人员会进行审核（已支持双可用区的），审核通过后即可按需操作，否则将驳回该需求。

版权所有：腾讯云计算（北京）有限责任公司

Binlog 日志消费格式

最近更新时间：2024-12-16 17:42:03

数据订阅功能帮助客户实时获取 TDSQL MySQL 版数据库中的增量数据，客户能够根据自身业务的需求，自由地对实时增量数据进行处理。

消息格式

数据订阅功能对实例的 Binlog（row 格式）进行解析，并将 Binlog 事件封装成 json 格式的消息上传至 Kafka 集群。消息类型包括 DML 事件、GTID 事件、XID 事件、QUERY 事件。其中 DML 事件包括 insert、update、delete 事件，表征对数据行的更改；GTID 事件表征事务的开始；XID 事件表征事务的提交；QUERY 事件表征 DDL 语句。

DML 消息格式如下：

```
{
    "logtype":"mysqlbinlog",          //日志类型，取值唯一，为 mysqlbinlog
    "eventtype":23,                   //事件类型码，对应 mysql 中 binlog 事
件类型码

    "eventtypestr":"insert",          /*事件类型字符串，包括 insert、
update、delete、gtid、xid、query 事件。
                                        其中insert、update、delete 事件对应
DML 语句；tid 事件标识事务开始；
                                        xid 事件标识事务结束；query 事件标识
DDL 语句 */
    "db":"testdb",                    //库名
    "table":"testtable",              //表名
    "localip":"000.00.000.000",       //实例所在机器 IP
    "localport":0000,                 //实例端口
    "begintime":1511350073,           //事务开始时间，当前事件所在的事务的开始
时间
    "gtid":"0-2670193178-726233561",  //GTID，当前事件所在事务的 gtid
    "event_index":"4",                //表征该事件在该事务中的序号
    "where":[                         //where 字段，标识该行变更前的各个字段的
值

    ],
    "field":[                         //field 字段，标识该行变更后的各个字段的
值
        "1",
        "'name1'"
    ]
```

```
}
```

GTID 消息格式如下：

```
{                                   //GTID 事件表征一个事务的开始
    "logtype":"mysqlbinlog",
    "eventtype":33,
    "eventtypestr":"gtid",
    "db":"sysdb",
    "table":"statustableforhb",
    "localip":"10.231.23.241",
    "localport":8810,
    "begintime":1511419963,
    "gtid":"35be190b-d019-11e7-ab7a-a0423f32c225:469",
    "event_index":"1"
}
```

XID 消息格式如下：

```
{                                   //XID 事件表征事务已经提交
    "logtype":"mysqlbinlog",
    "eventtype":16,
    "eventtypestr":"xid",
    "db":"testsummer",
    "table":"test_table1",
    "localip":"10.231.23.241",
    "localport":8810,
    "begintime":1511419963,
    "gtid":"35be190b-d019-11e7-ab7a-a0423f32c225:469",
    "event_index":"5",
    "xid":"11866"
}
```

QUERY 消息格式如下：

```
{
    "logtype":"mysqlbinlog",
    "eventtype":2,
    "eventtypestr":"query",         //QUERY 事件对应 DDL 语句
    "db":"testsummer",
    "table":"statustableforhb",
    "localip":"10.231.23.241",
```

```
    "localport":8810,
    "begintime":1511419941,
    "gtid":"35be190b-d019-11e7-ab7a-a0423f32c225:452",
    "event_index":"2",
    "sql":"create table test_table1 (id int primary key,name
varchar(20))"
}
```

订阅方式

客户通过获取存储在 Kafka 集群的消息事件来实时获取数据。通过腾讯云提供的数据订阅 API 来实时获取消息并进行处理。

慢查询分析

最近更新时间：2024-12-16 17:42:03

> **注意：**
> TDSQL MySQL 版的性能检测功能已停止更新；建议您使用 数据库智能管家 DBbrain 对数据库进行性能分析和优化。

1. 功能说明

超过指定时间的 SQL 语句查询称为“慢查询”，对应语句称为“慢查询语句”，而数据库管理员（DBA）对慢查询语句进行分析并找到慢查询出现原因的过程叫做慢查询分析。
TDSQL MySQL 版数据库在实例管理页的**性能优化**模块下，提供慢查询分析能力。

2. 主要参数说明

2.1. 主要默认设置

- 慢查询功能：默认开启
- 慢查询时间（long_query_time）：默认配置为1s；即慢查询语句查询时间超过1s的才被记录。
- 分析数据输出延迟：1分钟 – 5分钟。
- 日志记录时长：30天，根据备份和日志设置周期决定。

2.2. 分析列表字段说明

- 校验值（checksum）：标示慢查询语句的一串序列数字，默认64bit。
- 抽象后的慢查询语句（fingerprint）：隐去用户数据以后的慢查询语句。
- 数据库：出现慢查询语句的数据库。
- 账号：出现慢查询语句的账号。
- 最后执行时间（last_seen）：时间范围内，最后一次出现慢查询语句的时间。

- 首次执行时间（first_seen）：时间范围内，第一次出现慢查询语句的时间。
- 总次数（ts_cnt）：时间范围内，慢查询语句出现的次数。
- 总次数占比：时间范围内，慢查询语句占所有慢查询语句次数的占比。
- 总时间（query_time_sum）：时间范围内，慢查询语句查询总耗时。
- 总时间占比：时间范围内，慢查询语句查询总耗时的占比。
- 平均时间（query_time_avg）：慢查询语句总时间除以总次数的平均时间。
- 最小时间（query_time_min）：慢查询语句出现的最小时间。
- 最大时间（query_time_max）：慢查询语句出现的最大时间。
- 总锁时间（lock_time_sum）：慢查询语句出现锁的总耗时。
- 总锁时间占比：时间范围内，慢查询语句占所有慢查询语句锁时间的占比。
- 平均锁时间（lock_time_avg）：慢查询语句总锁时间除以总锁次数的平均时间。
- 最小锁时间（lock_time_min）：慢查询语句锁出现的最小时间。
- 最大锁时间（lock_time_max）：慢查询语句锁出现的最大时间。
- 发送行数（Rows_sent_sum）：该条慢查询语句发送的数据行数总和。
- 扫描行数（Rows_examined_sum）：该条慢查询语句扫描的数据行数总和。
- 主机地址（Host）：该条慢查询所来自的主机。

版权所有：腾讯云计算（北京）有限责任公司

保留关键字

最近更新时间：2024-12-16 17:42:04

此文章将列出 TDSQL MySQL 版集中式数据库所涉及的保留关键字，在 SQL 中使用时需添加 `‘’` 被解析。

主要保留关键字变更记录

关键字	变更内容	引入版本
CYCLE	允许 CYCLE 关键字做表名字段名	22.3.0
SEQUENCE	允许 SEQUENCE 关键字做表名字段名	22.3.0
TDSQL_SUBDISTRIBUTED	proxy 多级分区表	22.3.0
WITHOUT	proxy 回收站语法	22.3.0
RECYCLE_BIN	proxy 回收站语法	22.3.0
RECYCLE_ID	proxy 回收站语法	22.3.0
CLEAR	proxy 回收站语法	22.3.0
TEMPLATE	set 级全局索引	22.2.0
TDSQL_AUTOINCVAL	通过命令手动获取自增 id	22.1.4
MANUAL_SWITCH	rebalance 手动切换路由	22.1.0
SWITCH	rebalance 手动切换路由	22.1.0
SWITCH_DATETIME	rebalance 手动切换路由	22.1.0
DDL	查询 ddl 任务相关信息	20.8
OFFLINE	离线 rebalance	19.1
REBALANCE_TASK	查询 rebalance 任务相关信息	19.1
TDSQL_RESETVAL	sequence 重置序列号	17.2
TDSQL_DISTRIBUTED	range/list 分区表	16.3
SKIP	select for update 的 option	15.1
LOCKED	select for update 的 option	15.1
NOWAIT	select for update 的 option	15.1

版权所有：腾讯云计算（北京）有限责任公司

OF	select for update 的 option	15.1
CYCLE	proxy兼容 db 的 sequence 语法	15.1
INCREMENT	proxy 兼容 db 的 sequence 语法	15.1
LASTVAL	proxy兼容 db 的 sequence 语法	15.1
MAXVALUE	proxy 兼容 db 的 sequence 语法	15.1
NEXTVAL	proxy 兼容 db 的 sequence 语法	15.1
NOCACHE	proxy兼容 db 的 sequence 语法	15.1
NOCYCLE	proxy兼容 db 的 sequence 语法	15.1
NOMAXVALUE	proxy 兼容 db 的 sequence 语法	15.1
NOMINVALUE	proxy 兼容 db 的 sequence语法	15.1
PREVIOUS	proxy 兼容 db 的 sequence 语法	15.1
RESTART	proxy兼容 db 的 sequence 语法	15.1
REUSE	proxy兼容 db 的 sequence 语法	15.1
SEQUENCE	proxy 兼容 db 的 sequence 语法	15.1
SETVAL	proxy 兼容 db 的 sequence 语法	15.1
TDSQL_CYCLE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_INCREMENT	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_LASTVAL	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_MINVALUE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_NEXTVAL	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_NOCACHE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_NOCYCLE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_NOMAXVALUE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_NOMINVALUE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_PREVIOUS	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21

版权所有：腾讯云计算（北京）有限责任公司

TDSQL_RESTART	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_REUSE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_SEQUENCE	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21
TDSQL_SETVAL	proxy 支持带 tdsql_ 前缀的 sequence 语法	13.21

更早的保留关键字列表

```
&&, <, <=, <>, !=, =, >, >=, <<, >>, <=>, ACCESSIBLE, ACTION, ADD,
ADMIN, AFTER, AGAINST, AGGREGATE, ALL, ALGORITHM, ALTER, ALWAYS,
ANALYZE, AND, ANY, AS, ASC, ASCII, ASENSITIVE, AT, AUTHORS,
AUTO_INCREMENT,
AUTOEXTEND_SIZE, AUTO, AVG, AVG_ROW_LENGTH, BACKUP, BEFORE, BEGIN,
BETWEEN, BIGINT, BINARY, BINLOG, BIT, BLOB, BLOCK, BODY, BOOL, BOOLEAN,
BOTH, BTREE, BY, BYTE, CACHE, CALL, CASCADE, CASCADED, CASE,
CATALOG_NAME, CHAIN, CHANGE, CHANGED, CHAR, CHARACTER, CHARSET, CHECK,
CHECKPOINT, CHECKSUM, CIPHER, CLASS_ORIGIN, CLIENT, CLIENT_STATISTICS,
CLOSE, COALESCE, CODE, COLLATE, COLLATION, COLUMN, COLUMN_NAME,
COLUMNS, COLUMN_ADD, COLUMN_CHECK, COLUMN_CREATE, COLUMN_DELETE,
COLUMN_GET, COMMENT, COMMIT, COMMITTED, COMPACT, COMPLETION, COMPRESSED,
CONCURRENT, CONDITION, CONNECTION, CONSISTENT, CONSTRAINT,
CONSTRAINT_CATALOG, CONSTRAINT_NAME, CONSTRAINT_SCHEMA, CONTAINS,
CONTEXT, CONTINUE, CONTRIBUTORS, CONVERT, CPU, CREATE, CROSS, CUBE,
CURRENT, CURRENT_DATE, CURRENT_POS, CURRENT_ROLE, CURRENT_TIME,
CURRENT_TIMESTAMP, CURRENT_USER, CURSOR, CURSOR_NAME, CYCLE, DATA,
DATABASE, DATABASES, DATAFILE, DATE, DATETIME, DAY, DAY_HOUR,
DAY_MICROSECOND, DAY_MINUTE, DAY_SECOND, DEALLOCATE, DEC, DECIMAL,
DECLARE,
DEFAULT, DEFINER, DELAYED, DELAY_KEY_WRITE, DELETE, DESC, DESCRIBE,
DES_KEY_FILE, DETERMINISTIC, DIAGNOSTICS, DIRECTORY, DISABLE, DISCARD,
DISK, DISTINCT, DISTINCTROW, DIV, DO, DOUBLE, DROP, DUAL, DUMPFILE,
DUPLICATE, DYNAMIC, EACH, ELSE, ELSEIF, ENABLE, ENCLOSED, END, ENDS,
ENGINE, ENGINES, ENUM, ERROR, ERRORS, ESCAPE, ESCAPED, EVENT, EVENTS,
EVERY, EXAMINED, EXCHANGE, EXECUTE, EXISTS, EXIT, EXPANSION,
EXPORT, EXPLAIN, EXTENDED, EXTENT_SIZE, FALSE, FAST, FAULTS, FETCH,
FIELDS, FILE, FIRST, FIXED, FLOAT, FLOAT4, FLOAT8, FLUSH, FOR, FORCE,
FOREIGN, FOUND, FROM, FULL, FULLTEXT, FUNCTION, GENERAL, GENERATED,
GEOMETRY, GEOMETRYCOLLECTION, GET_FORMAT, GET, GLOBAL, GRANT, GRANTS,
GROUP, HANDLER, HARD, HASH, HAVING, HELP, HIGH_PRIORITY, HOST, HOSTS,
HOUR, HOUR_MICROSECOND, HOUR_MINUTE, HOUR_SECOND, ID, IDENTIFIED,
IF, IGNORE, IGNORE_SERVER_IDS, IMPORT, IN, INCREMENT, INDEX, INDEXES,
INDEX_STATISTICS, INFILE, INITIAL_SIZE, INNER, INOUT, INSENSITIVE,
```

版权所有：腾讯云计算（北京）有限责任公司

```
INSERT, INSERT_METHOD, INSTALL, INT, INT1, INT2, INT3, INT4, INT8,
INTEGER, INTERVAL, INTO, IO, IO_THREAD, IPC, IS, ISOLATION, ISSUER,
ITERATE, INVOKER, JOIN, JSON, KEY, KEYS, KEY_BLOCK_SIZE, KILL, LANGUAGE,
LAST, LAST_VALUE, LASTVAL, LEADING, LEAVE, LEAVES, LEFT, LESS,
LEVEL, LIKE, LIMIT, LINEAR, LINES, LINESTRING, LIST, LOAD, LOCAL,
LOCALTIME, LOCALTIMESTAMP, LOCK, LOCKS, LOGFILE, LOGS, LONG, LONGBLOB,
LONGTEXT, LOOP, LOW_PRIORITY, MASTER, MASTER_CONNECT_RETRY,
MASTER_GTID_POS, MASTER_HOST, MASTER_LOG_FILE, MASTER_LOG_POS,
MASTER_PASSWORD, MASTER_PORT, MASTER_SERVER_ID, MASTER_SSL,
MASTER_SSL_CA, MASTER_SSL_CAPATH, MASTER_SSL_CERT, MASTER_SSL_CIPHER,
MASTER_SSL_CRL, MASTER_SSL_CRLPATH, MASTER_SSL_KEY,
MASTER_SSL_VERIFY_SERVER_CERT, MASTER_USER, MASTER_USE_GTID,
MASTER_HEARTBEAT_PERIOD, MATCH, MAX_CONNECTIONS_PER_HOUR,
MAX_QUERIES_PER_HOUR, MAX_ROWS,
MAX_SIZE, MAX_UPDATES_PER_HOUR, MAX_USER_CONNECTIONS, MAXVALUE, MEDIUM,
MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, MEMORY, MERGE, MESSAGE_TEXT,
MICROSECOND, MIDDLEINT, MIGRATE, MINUTE, MINUTE_MICROSECOND,
MINUTE_SECOND, MINVALUE, MIN_ROWS, MOD, MODE, MODIFIES, MODIFY, MONTH,
MULTILINESTRING, MULTIPOINT, MULTIPOLYGON, MUTEX, MYSQL_ERRNO, NAME,
NAMES, NATIONAL, NATURAL, NDB, NDBCLUSTER, NCHAR, NEW, NEXT,
NEXTVAL, NO, NOMAXVALUE, NOMINVALUE, NOCACHE, NOCYCLE, NO_WAIT,
NODEGROUP, NONE, NOT, NO_WRITE_TO_BINLOG, NULL, NUMBER, NUMERIC,
NVARCHAR, OFFSET, OLD_PASSWORD, ON, ONE, ONLINE, ONLY, OPEN, OPTIMIZE,
OPTIONS, OPTION, OPTIONALLY, OR, ORDER, OUT, OUTER, OUTFILE, OWNER,
PACK_KEYS, PAGE, PAGE_CHECKSUM, PARSER, PARSE_VCOL_EXPR, PARTIAL,
PARTITION, PARTITIONING, PARTITIONS, PASSWORD, PERSISTENT, PHASE,
PLUGIN,
PLUGINS, POINT, POLYGON, PORT, PRECISION, PREPARE, PRESERVE, PREV,
PREVIOUS, PRIMARY, PRIVILEGES, PROCEDURE, PROCESS, PROCESSLIST, PROFILE,
PROFILES, PROXY, PURGE, QUARTER, QUERY, QUICK, RANGE, READ,
READ_ONLY, READ_WRITE, READS, REAL, REBUILD, RECOVER, REDO_BUFFER_SIZE,
REDOFILE, REDUNDANT, REFERENCES, REGEXP, RELAY, RELAYLOG,
RELAY_LOG_FILE, RELAY_LOG_POS, RELAY_THREAD, RELEASE, RELOAD, REMOVE,
RENAME,
REORGANIZE, REPAIR, REPEATABLE, REPLACE, REPLICATION, REPEAT, REQUIRE,
RESET, RESIGNAL, RESTART, RESTORE, RESTRICT, RESUME, RETURNED_SQLSTATE,
RETURN, RETURNING, RETURNS, REUSE, REVERSE, REVOKE, RIGHT, RLIKE,
ROLE, ROLLBACK, ROLLUP, ROUTINE, ROW, ROW_COUNT, ROWS, ROW_FORMAT,
RTREE, SAVEPOINT, SCHEDULE, SCHEMA, SCHEMA_NAME, SCHEMAS,
SECOND,SECOND_MICROSECOND,SECURITY, SELECT, SENSITIVE, SEPARATOR,
SEQUENCE, SERIAL,
SERIALIZABLE, SESSION, SERVER, SET, SETVAL, SHARDKEY, SHARE, SHOW,
SHUTDOWN, SIGNAL, SIGNED, SIMPLE, SLAVE, SLAVES, SLAVE_POS, SLOW,
```

```
SNAPSHOT, SMALLINT, SOCKET, SOFT, SOME, SONAME, SOUNDS, SOURCE, SPATIAL,
SPECIFIC, SQL, SQLEXCEPTION, SQLSTATE, SQLWARNING, SQL_BIG_RESULT,
SQL_BUFFER_RESULT, SQL_CACHE, SQL_CALC_FOUND_ROWS, SQL_NO_CACHE,
SQL_SMALL_RESULT, SQL_THREAD, SQL_TSI_SECOND, SQL_TSI_MINUTE,
SQL_TSI_HOUR,
SQL_TSI_DAY, SQL_TSI_WEEK, SQL_TSI_MONTH, SQL_TSI_QUARTER, SQL_TSI_YEAR,
SSL, START, STARTING, STARTS, STATS_AUTO_RECALC, STATS_PERSISTENT,
STATS_SAMPLE_PAGES, STATUS, STOP, STORAGE, STORED, STRAIGHT_JOIN,
STRING, SUBCLASS_ORIGIN, SUBJECT, SUBPARTITION, SUBPARTITIONS, SUPER,
SUSPEND, SWAPS, SWITCHES, TABLE, TABLE_NAME, TABLES, TABLESPACE,
TABLE_STATISTICS, TABLE_CHECKSUM, TEMPORARY, TEMPTABLE, TERMINATED,
TEXT,
THAN, THEN, TIME, TIMESTAMP, TIMESTAMPADD, TIMESTAMPDIFF, TINYBLOB,
TINYINT, TINYTEXT, TO, TRAILING, TRANSACTION, TRANSACTIONAL, TRIGGER,
TRIGGERS, TRUE, TRUNCATE, TYPE, TYPES, UNCOMMITTED, UNDEFINED,
UNDO_BUFFER_SIZE, UNDOFILE, UNDO, UNICODE, UNION, UNIQUE, UNKNOWN,
UNLOCK, UNINSTALL, UNSIGNED, UNTIL, UPDATE, UPGRADE, USAGE, USE, USER,
USER_RESOURCES, USER_STATISTICS, USE_FRM, USING, UTC_DATE, UTC_TIME,
UTC_TIMESTAMP, VALUE, VALUES, VARBINARY, VARCHAR, VARCHARACTER,
VARIABLES, VARYING, VIA, VIEW, VIRTUAL, WAIT, WARNINGS, WEEK,
WEIGHT_STRING, WHEN, WHERE, WHILE, WITH, WORK, WRAPPER, WRITE, X509,
XOR, XA, XML,
YEAR, YEAR_MONTH, ZEROFILL, ||, BOOST
```

版权所有：腾讯云计算（北京）有限责任公司

数据库审计

最近更新时间：2024-12-16 17:42:04

> **注意：**
> 数据库审计功能重构升级中，敬请期待；在此期间数据库新购实例不再开放审计功能。

概述

背景说明

企业使用数据库，可能面临如下安全风险，该类风险需要完整的事后审计和追溯机制，数据库审计能力由此诞生。

管理风险

- 系统管理员存在的误操作、违规操作、越权操作，损害业务系统安全运行。
- 多人公用一个账号，责任难以分清。
- 第三方开发维护人员的误操作，恶意操作和篡改。
- Root 账号权限过大，无法审计监控。

技术风险

- 应用系统开发商后门或漏洞。
- 离职员工留下后门。

政策风险

- 无法达到国家等级保护（三级）明确要求（7.1.3.3）。
- 满足不了行业信息安全合规性文件要求——如人行《金融行业信息系统信息安全等级保护实施指引》。

术语定义

审计策略： 定义对哪些用户行为进行审计以及如何响应的策略。 **审核策略 = 审核对象 + 审计规则 + 响应动作** 即配置一条审计策略，需要指定审计内容，如果经过解析，某些（用户或系统）行为的特征正好符合某个审计规则，且恰好在策略生效时间，审计引擎就会按照此策略定义的响应方式进行响应，例如告警等。

审计规则： 审计策略中，规定了需要审计的一系列行为的集合，称为规则。规则由规则参数组成，每个规则参数定义了一种具体的行为匹配特征。

产品能力与限制条件

腾讯云提供数据库审计能力，审计日志默认保存7天，帮助企业对可能存在的数据库访问进行风险控制，提高数据安全等级。

审计操作

开通数据库审计

使用 TDSQL MySQL 版集中式数据库（以下简称为 TDSQL）的用户，均可开通数据库审计；开通入口在 数据库审计 页面。

开通审计存在以下注意事项：

- 您至少须拥有1个 TDSQL 实例，且未下线或隔离，否则系统会自动关闭您的审计功能。
- 2016年6月5日之前购买的 TDSQL 实例，需重启升级后方可支持该能力，由于重启升级可能会导致1秒 – 5秒的业务中断，您可以联系腾讯云工作人员预约升级时间。
- 数据库审计日志将以明文形式展示，因此建议您开启 二次登录认证 。
  开通审计可能存在几分钟初始化时间，请耐心等待。

新建审计规则

审计功能开通后，日志会自动通过 TDSQL 网关集群转发到审计集群，由于没有建立审计规则和审计策略，日志不会持久化记录并展示。因此您可以通过**新建审计规则** > **关联审计策略**让日志存储在审计集群中。

1. 进入 审计规则 页，单击**新建规则**。
2. 填写审计规则名称，单击**下一步**。
3. 进入参数设置页面，填写规则参数（所列规则参数需至少填写一个，但不必全部填写）。
   - **规则中参数的条件关系：** 与关系；规则中各参数之间是与的关系，即各参数都满足条件规则才会匹配成功。
   - **特征串：** 定义参数的具体内容，也就是操作对象的具体特征。为了达到精确匹配，用户只定义自己关心的参数的关键字，这样审计系统就只需记录用户定义的规则，提高审计检索效率。注意：特征串为空表示不关心该参数，即“匹配所有”。
   - **匹配类型：** 参数对象和特征的关系。
     - **包含：** 表示网络字段中出现了特征串就匹配成功。
     - **不包含：** 表示网络字段中没有出现特征串则匹配成功。
     - **等于：** 表示网络字段等于特征串则匹配成功。
     - **不等于：** 表示网络字段不等于特征串则匹配成功。
     - **正则表达式：** 表示特征串，支持标准的正则表达式语法。
4. 所有新建规则，可以在规则列表中看到。
5. 设置完成审计规则后，您可以随时修改。类似的规则，您可以通过**克隆规则**的方式新建，以提高效率。

新建审计策略

审计策略是将审计规则、审计对象、响应方式组合起来形成完整的审计方案。用户可以对一个实例同时制定多条审计策略，审计引擎解析时，将**按用户的策略配置顺序从前到后的优先级匹配**。

1. 选择**审计策略**页，单击**新建策略**。
2. 填写策略要求，根据需求选择需审计实例，并选择对应规则（目前暂不支持配置告警）。
3. 调整优先级：对于相同实例下的多条策略，可以进行优先级调整；优先级数字越小，优先等级越高。优先级调整后，预计1分钟内生效。
4. 您可以通过修改功能，实时修改审计策略；修改完成后预计5分钟内生效，并按新策略进行审计监控，修改审计策略前的日志不会被修改。

查看日志

匹配到审计策略的 SQL 语句将展现在审计日志页面，您可以直接单击查看或搜索。注意事项：

- 审计日志明文方式展现，建议您开启 二次登录认证 ，以确保日志可控。
- 日志将以审计策略创建时开始记录，历史数据不做记录。
- 事务、存储过程等可能会被记录为单条语句，详见 数据库审计已支持语法说明 。
- 目前支持单条 SQL 语句最大1k，超过部分会被截断。

数据库审计已支持语法

最近更新时间：2023-02-07 11:18:44

> **注意**
> 数据库审计功能重构升级中，敬请期待；在此期间数据库新购实例不再开放审计功能。

数据库审计目前已经支持绝大多数 SQL 语句，如发现存在不足，请 联系我们 反馈。

- 已支持 DCL、DDL、DML 语句的解析。

```
Insert,Replace,Select,Union,Update,Delete,CreateDatabase:,CreateEvent,
CreateFunction,CreateIndex,CreateLog,
CreateTable,CreateServer,CreateProcedure,CreateTablespace,CreateTrigge
r,CreateView,CreateUDF,CreateUser,
ShowCharset,ShowCollation,ShowColumns,ShowCreate,ShowCreateDatabase,Sh
owDatabases,ShowEngines,ShowErrors,
ShowEvents,ShowFunction,ShowGrants,ShowLogEvents,ShowLogs,ShowProcedur
e,ShowOpenTables,ShowPlugins,
ShowProcessList,ShowMasterStatus,ShowPrivileges,ShowProfiles,ShowSlave
Hosts,ShowSlaveStatus,ShowTableStatus,
ShowWarnings,ShowVariables,ShowStatus,ShowTriggers,Call,DropProcedure,
DropDatabase,DropEvent,DropFunction,
DropIndex,DropLogfile,DropServer,DropTables,DropTablespace,DropTrigger
,DropUser,DropView,AlterDatabase,
AlterEvent,AlterFunction,AlterLogfile,AlterProcedure,AlterServer,Alter
Table,AlterTablespace,AlterUser,
AlterView,Rollback,Commit,Begin,Set,SetTrans,SetPassword,Release,Grant
,RenameTable,RenameUser,Revoke,
Install,StopSlave,StartSlave,StartTrans,Use,DescribeTable,DescribeStmt
,Flush,Load,LoadIndex,FlushTables,
Reset,CacheIndex,TruncateTable,Lock,Unlock,SavePoint,Help,Do,SubQuery,
ShowTables,Execute,Deallocate,Binlog,
Kill,Partition,PrepareRepairXACheckCheckSumAnalyzeChangeOptimizePurgeH
andlerSignalResignal
```

- 事务和存储过程可能被拆分为多条语句。

SEQUENCE

最近更新时间：2024-11-25 19:27:01

CREATE

```
-- 建表语句
CREATE TDSQL_SEQUENCE {DATABASE | SCHEMA}  sequence_name
    [ TDSQL_INCREMENT BY increment ]
    [ START  WITH  startvalue ]
    [ TDSQL_MINVALUE minvalue] | [ TDSQL_NOMINVALUE ]
    [ TDSQL_MAXVALUE  maxvalue] | [ TDSQL_NOMAXVALUE ]
    [ TDSQL_CACHE  cachevalue ] | [ TDSQL_NOCACHE ]
    [ TDSQL_CYCLE ] | [ TDSQL_NOCYCLE ]
    [ TDSQL_ORDER ] | [ TDSQL_NOORDER ]

-- 查看 sequence 元数据
SHOW CREATE TDSQL_SEQUENCE sequence_name;
```

关键字解释

- TDSQL_SEQUENCE：标识为序列操作。
- TDSQL_INCREMENT BY：设置序列步长，正数表示升序序列，负数表示降序序列。缺省则默认为1。
- START WITH：设置序列起始值，缺省则默认为1。
- TDSQL_MAXVALUE 和 TDSQL_NOMAXVALUE：设置序列最大值，TDSQL_MAXVALUE 缺省或者设置 TDSQL_NOMAXVALUE 时，递增序列最大值为 9223372036854775807，递减序列最大值为-1。
- TDSQL_MINVALUE 和 TDSQL_NOMINVALUE：设置序列最小值，TDSQL_MINVALUE 缺省或者设置 TDSQL_NOMINVALUE 时，递增序列最小值为1，递减序列最小值为-9223372036854775808。
- TDSQL_CYCLE 和 TDSQL_NOCYCLE：设置序列是否循环，缺省默认不循环。
- TDSQL_CACHE 和 TDSQL_NOCACHE：设置缓存序列数量，缺省默认缓存20。
- TDSQL_ORDER 和 TDSQL_NOORDER：设置序列对于并发请求保证完全有序。TDSQL 目前仅支持无序。

语法示例

```
-- 设置最大值，其他未定义项均选择默认值，创建序列成功后，可在
MYSQL.TDSQL_SEQUENCES 表中查看到序列元数据
> CREATE DATABASE dbtest_seq_alter_start;
> use dbtest_seq_alter_start;
> CREATE TDSQL_SEQUENCE  sequence_01 maxvalue 100;
```

```
-- 查看到序列元数据使用SHOW CREATE SEQUENCE或mysql.tdsql_sequences
> SHOW CREATE TDSQL_SEQUENCE  sequence_01;
Sequence Create Sequence
sequence_01 CREATE SEQUENCE `dbtest_seq_alter_start`.`sequence_01` start
with 1 minvalue 1 maxvalue 100 increment by 1 cache 20 nocycle

> SELECT * from mysql.tdsql_sequences where name='sequence_01';
+------+-------------+---------------------+-------+------+-----------
+-----------+----------+---------+
| db   | name        | curval              | start | step | max_value |
min_value | do_cycle | n_cache |
+------+-------------+---------------------+-------+------+-----------
+-----------+----------+---------+
| test | sequence_01 | -9223372036854775808 |     1 |    1 |       100 |
1 |        0 |      20 |
+------+-------------+---------------------+-------+------+-----------
+-----------+----------+---------+

-- 创建升序序列
CREATE TDSQL_SEQUENCE   sequence_001  TDSQL_INCREMENT BY 1  START WITH 1
TDSQL_MAXVALUE 10000000 TDSQL_MINVALUE 1 TDSQL_CYCLE TDSQL_CACHE 100
TDSQL_NOORDER;

-- 创建降序序列，是否为降序序列，通过TDSQL_INCREMENT BY 步长决定，步长为负时表示此
序列为降序序列
CREATE TDSQL_SEQUENCE sequence_002 TDSQL_INCREMENT BY -1    START WITH
10000000 TDSQL_MAXVALUE 10000000 TDSQL_MINVALUE 1 TDSQL_CYCLE
TDSQL_CACHE 100 ;
```

ALTER

```
ALTER TDSQL_SEQUENCE {DATABASE | SCHEMA}  sequence_name
    [ TDSQL_INCREMENT BY increment ]
    [ TDSQL_MINVALUE minvalue ] | [ TDSQL_NOMINVALUE ]
    [ TDSQL_MAXVALUE  maxvalue ] | [ TDSQL_NOMAXVALUE ]
    [ TDSQL_CACHE  cachevalue ]  | [ TDSQL_NOCACHE ]
    [ TDSQL_CYCLE ] | [ TDSQL_NOCYCLE ]
    [ TDSQL_ORDER ] | [ TDSQL_NOORDER ]
    [ START  WITH  startvalue ]
    [ RESTART [WITH restartvalue] ]
```

关键字说明：

- TDSQL_INCREMENT BY：设置序列步长，正数表示升序序列，负数表示降序序列。缺省则默认1。
- TDSQL_MAXVALUE 和 TDSQL_NOMAXVALUE：设置序列最大值，TDSQL_MAXVALUE 缺省或者设置 TDSQL_NOMAXVALUE 时，递增序列最大值为 9223372036854775807，递减序列最大值为-1。
- TDSQL_MINVALUE 和 TDSQL_NOMINVALUE：设置序列最小值，TDSQL_MINVALUE 缺省或者设置 TDSQL_NOMINVALUE 时，递增序列最小值为1，递减序列最小值为-9223372036854775808。
- TDSQL_CYCLE 和 TDSQL_NOCYCLE：设置序列是否循环，缺省默认不循环。
- TDSQL_CACHE 和 TDSQL_NOCACHE：设置缓存序列数量，缺省默认缓存20。
- START WITH 设置序列起始值，缺省则默认为1。设置 TDSQL_CYCLE 时下一次循环生效。
- RESTART WITH 重置序列的值，下次调用 nextval 则按照设置的值重新开始，如未设置，取 START 当前值。

注意：

- `START WITH | RESTART WITH` 设置范围需在 [TDSQL_MINVALUE，TDSQL_MAXVALUE] 区间。
- RESTART WITH 语法10.3.22.2及之后版本开始支持。
- RESTART WITH 在 TDSQL_NOCYCLE 下不支持回退取号。

语法示例

```
--更新序列步长
> ALTER TDSQL_SEQUENCE sequence_001 TDSQL_INCREMENT BY 2;

-- 修改tdsql sequence的起始值
> ALTER tdsql_sequence sequence_001 start with 110;

-- 重置tdsql sequence的序列号
> ALTER tdsql_sequence sequence_001 restart with 1;

-- 在由TDSQL_CYCLE 变为 TDSQL_NOCYCLE时,数值不允许回退。当前值55, restart 语句
想回退到50, restart不满足约束条件, 导致报错
> alter tdsql_sequence sequence_01 increment by 1 restart start with 50
maxvalue 100 cycle;
> SELECT sequence_01.CURRVAL;
+---------------------+
| sequence_01.CURRVAL |
+---------------------+
|                  55 |
+---------------------+
```

```
> alter tdsql_sequence sequence_01 increment by 1 restart start with 50
maxvalue 100 nocycle;
ERROR HY000: Wrong sequence argument(restart) value specified: restart
valueexceeds [min, max] without cycle option given
```

DROP

语法示例

```
-- 更新序列步长
DROP TDSQL_SEQUENCE sequence_001;
```

SELECT

```
-- 使用Sequence获取下一个数值
select  sequence_001.nextval
-- 查看序列值:
查看当前序列值:
SELECT sequence_001.CURRVAL;
-- 查看下一序列值:
SELECT sequence_001.NEXTVAL;
```

```
-- 批量使用SEQUENCE,使SEQUENE递增，sequence_same_nextval_in_query 需为OFF，
默认为ON保持相同。在赤兔-数据库参数中设置该参数为OFF状态
> show variables like '%same%';
+-------------------------------+-------+
| Variable_name                 | Value |
+-------------------------------+-------+
| sequence_same_nextval_in_query | OFF   |
+-------------------------------+-------+
1 row in set (0.00 sec)

> select * from s1;
+------+
| name |
+------+
| Amy  |
| Bob  |
+------+
```

```
2 rows in set (0.00 sec)

-- 可以看到SEQUENCE为递增显示


> select seq1.nextval seq,name from s1;
+-----+------+
| seq | name |
+-----+------+
|   1 | Amy  |
|   2 | Bob  |
+-----+------+
2 rows in set (0.01 sec)
```