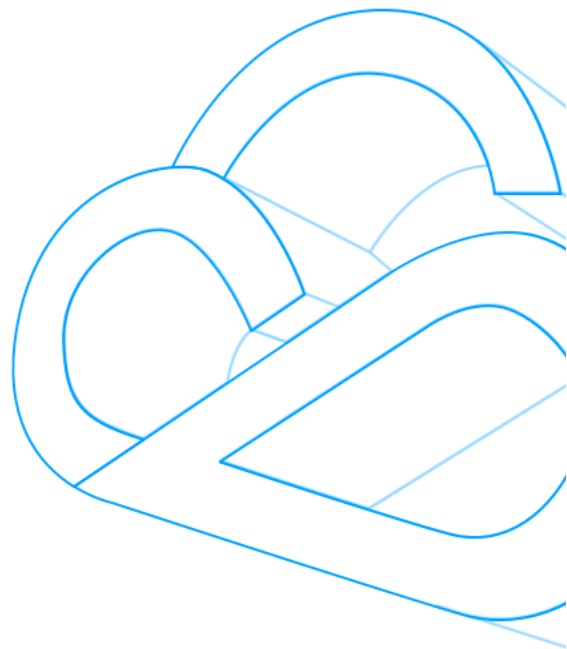




腾讯云数据库 TDSQL MySQL 版 分布式 V10.3.22.8/集中式 V8.0.22.8 命令参考



文档版本：

发布日期：

腾讯云计算（北京）有限责任公司

版权声明

本文档著作权归腾讯云计算（北京）有限责任公司（以下简称“腾讯云”）单独所有，未经腾讯云事先书面许可，任何主体不得以任何方式或理由使用本文档，包括但不限于复制、修改、传播、公开、剽窃全部或部分本文档内容。

本文档及其所含内容均属腾讯云内部资料，并且仅供腾讯云指定的主体查看。如果您非经腾讯云授权而获得本文档的全部或部分内容，敬请予以删除，切勿以复制、披露、传播等任何方式使用本文档或其任何内容，亦请切勿依本文档或其任何内容而采取任何行动。

商标声明



“腾讯”、“腾讯云”及其它腾讯云服务相关的商标、标识等均为腾讯云及其关联公司各自所有。若本文档涉及第三方主体的商标，则应依法由其权利人所有。

免责声明

本文档旨在向客户介绍本文档撰写时，腾讯云相关产品、服务的当时的整体概况，部分产品或服务在后续可能因技术调整或项目设计等任何原因，导致其服务内容、标准等有所调整。因此，本文档仅供参考，腾讯云不对其准确性、适用性或完整性等做任何保证。您所购买、使用的腾讯云产品、服务的种类、内容、服务标准等，应以您和腾讯云之间签署的合同约定为准，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。

修订记录

文档版本	发布日期	修订人	修订内容
------	------	-----	------

目录

修订记录.....	ii
目录.....	iii
前言.....	v
1 定时任务脚本说明.....	1
2 DB 命令参考	3
2.1 mysqlcheck.....	3
2.2 mysqldumpslow	3
2.3 mysqlbinlog.....	4
2.4 mysqldump	5
2.5 mysqlrouter_passwd	5
2.6 ibd2sd	6
2.7 mysqlimport.....	7
2.8 lz4_decompress.....	7
2.9 mysqlshow.....	7
2.10 mysqlpump	8
2.11 mysqlslap.....	9
2.12 perror	9
2.13 mysqladmin	9
2.14 mysqld_safe	10
3 mysqlagent 主要执行文件	11
4 oc_agent 主要执行文件.....	20
5 proxy 主要执行文件.....	21
6 zookeeper 主要执行文件	24
7 Scheduler/Manager 命令参考	26
7.1 scheduler/manager 主要执行文件	26
7.2 manual_set.....	30
7.3 modify_election	36

7.4 resource_tool.....	37
8 oss 主要执行文件.....	44
9 monitor(collector&analyze)主要执行文件	45
10 clouddba 主要执行文件	46
11 kafka 主要执行文件	48
12 consumer 主要执行文件	52
13 onlineddl 主要执行文件.....	56
14 filebeat 主要执行文件	57
15 网关参数说明.....	59
15.1 server	59
15.2 instance.....	60
15.3 log	61
15.4 mode	63
15.5 security	67
15.6 statistics.....	70
15.7 flowcontrol	71
15.8 xa.....	72
15.9 join.....	74
16 usercheck 参数说明.....	76

前言

文档目的

本文档用于帮助用户掌握云产品的操作方法与注意事项。





目标读者

本文档主要适用于如下对象群体：

- 客户
- 交付 PM
- 交付技术架构师
- 交付工程师
- 产品交付架构师
- 研发工程师
- 运维工程师

符号约定

本文档中可能采用的符号约定如下：

符号	说明
 说明：	表示是正文的附加信息，是对正文的强调和补充。
 注意：	表示有低度的潜在风险，主要是用户必读或较关键信息，若用户忽略注意消息，可能会因误操作而带来一定的不良后果或者无法成功操作。
 警告：	表示有中度的潜在风险，例如用户应注意的高危操作，如果忽视这些文本，可能导致设备损坏、数据丢失、设备性能降低或不可预知的结果。
 禁止：	表示有高度潜在危险，例如用户应注意的禁用操作，如果不能避免，会导致系统崩溃、数据丢失且无法修复等严重问题。

1 定时任务脚本说明

模块	任务详细	任务说明
tdsql_begining	cd /data/oc_agent/scripts; ./agent_monitor.sh	监控节点 agent 运行情况
	cd /data/monitorcmd; ./monitoriotop.sh	监控所有的磁盘和文件系统的 磁盘 I/O 信息
tdsql_zk_cluster	/bin/bash /data/tools/check_zk_alive.sh	zookeeper 集群存活检测
tdsql_chitu	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron cluster statistics	定期刷新集群状态和信息
	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron oss2db run	定期从 OSS 获取实例， mysql，proxy 信息同步到监控 库，判断是否下架
	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron workflow run	定期后台执行没有运行的异步 任务
	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron dbcluster statistics_client	定期统计实例的 request_total_select_sum， request_total_modify_sum 等信 息到监控库
	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron dcnjob run	定期从 OSS 获取 dcn 信息同 步到赤兔系统库
	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron syncjob run	定期从 OSS 获取多源同步信 息同步到赤兔系统库
	/usr/local/php/bin/php /data/website/chitu/tdsqlpcloud/index.php cron dbuser_privileges run	对尚未执行完的授权单任务进 行离线处理
	/usr/local/php/bin/php /data/website/tdsqlpcloud/index.php cron dbuser_apply run	对尚未执行完的申请单任务进 行离线处理
	/usr/local/php/bin/php /data/website/chitu/tdsqlpcloud/index.php cron cluster sync_to_local	#集群信息同步到本地文件

	/bin/bash /data/tools/check_nginx_alive.sh	nginx 进程存活监控
	/bin/bash /data/tools/check_php_alive.sh	php-fpm 进程存活监控
tdsql_oss	/bin/bash /data/tools/check_oss_alive.sh	oss 进程存活监控
tdsql_collector_analyzer	/bin/bash /data/tools/check_coll_any_alive.sh	collector/analyze 进程存活监控
tdsql_scheduler	/bin/bash /data/tools/check_scheduler_alive.sh	scheduler 进程存活监控
	cd /data/application/scheduler/bin;./backupZkInfo	zookeeper 备份
tdsql_clouddba	/bin/bash /data/tools/check_clo_alive.sh	tdsql-diagnosis 进程(clouddba) 存活监控
tdsql_onlineddl	/bin/bash /data/tools/check_onlineddl_alive.sh	ddlperformermng 进程存活监控
tdsql_consumer	/usr/bin/python /data/application/consumer/bin/multisrcsync_log_bak_clean.py	定期备份和清理由 binlogconsumer 生成的日志

2 DB 命令参考

2.1 mysqlcheck

命令功能

mysqlcheck 是一个维护、分析、优化和修复数据库表的命令行工具。

命令格式

```
mysqlcheck [options] db_name [tbl_name ...]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -A (--all-databases) 检查所有数据库。
- -a (--analyze) 分析给定的表。
- -o (--optimize) 优化给定的表。
- -r (--repair) 修复给定的表。
- --auto-repair 如果检查的表已损坏，则自动修复它。检查完所有表格后进行任何必要的维修。
- -c (--check) 检查表中是否有错误。
- --host=host_name, -h host_name: 连接到给定主机上的 TDSQL 服务器。
- --port=port_num, -P port_num: 端口号。
- -u user_name 用于连接到服务器的 TDSQL 帐户的用户名。
- -p 用于连接服务器的 TDSQL 帐户的密码。密码值是可选的。如果没有给出，请根据 mysqlcheck 提示输入。
- --auto-repair 如果检查的表已损坏，则自动修复它。检查完所有表格后进行任何必要的维修。

使用示例

```
mysqlcheck -h 172.16.0.11 -P 4001 -u tdsqplcloud -p --auto-repair --optimize --all-databases
```

2.2 mysqldumpslow

命令功能

mysqldumpslow 用于解析 MySQL 服务器的慢查询日志文件。默认情况下，mysqldumpslow 按平均查询时间排序（相当于 -s at）。

命令格式

```
mysqldumpslow [options] [log_file ...]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -s sort_type 按照某种方式排序。
 - t, at（默认值）：按查询时间或平均查询时间排序。
 - l, al: 按锁定时间或平均锁定时间排序。
 - r, ar: 按发送的行数或平均发送的行数排序。
 - c: 按数量排序。
- -r (--reverse) 反向排序结果。
- -t N 只返回前 N 个查询。

使用示例

```
mysqldumpslow -t 10 -s at /var/log/mysql/mysql-slow.log
```

2.3 mysqlbinlog

命令功能

mysqlbinlog 用于处理并解析二进制日志文件（binlog）。

命令格式

```
mysqlbinlog [options] log_file ...
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -v (--verbose) 详细输出。
- -s (--short-form) 只显示最简单的查询。
- -d (--database=database) 只显示指定数据库的查询。
- --start-datetime 指定解析开始时间。
- --stop-datetime 指定解析结束时间。
- --base64-output=value 使用 base-64 编码打印二进制日志。
 - AUTO（默认值）：BINLOG 在必要时自动显示语句。
 - NEVER: 不显示 BINLOG 语句。

- DECODE-ROWS: 与--verbose 配合使用, 显示 BINLOG 行事件。

使用示例

```
mysqlbinlog --start-datetime="2024-01-01 17:55:00" --stop-datetime="2024-01-01 18:00:00" --base64-output=decode-rows --verbose bin.000025
```

2.4 mysqldump

命令功能

mysqldump 用于备份 MySQL 数据库。

命令格式

```
mysqldump [options] db_name [tbl_name ...]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -u (--user=name) 用于连接到服务器的 TDSQL 帐户的用户名。
- -p (--password[=name]) 用于连接服务器的 TDSQL 帐户的密码。密码值是可选的。如果没有给出, 请根据提示输入。
- -B (--databases) 处理指定的数据库。
- -A (--all-databases) 处理所有数据库。

使用示例

```
mysqldump -u root -p --all-databases > alldb_backup.sql
```

2.5 mysqlrouter_passwd

命令功能

mysqlrouter_passwd 工具是一个命令行应用程序, 用于管理 passwd 文件中的账户。操作系统和 MySQL Router 的权限设置可能会要求你使用 sudo 或以管理员身份运行 mysqlrouter_passwd 命令。

命令格式

```
mysqlrouter_passwd [opts] <cmd> <filename> [<username>]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

cmd

- **delete:** 从<filename>中删除用户名。
- **list:** 列出<filename>的一个或全部帐户。
- **set:** 在<filename>中添加或覆盖<username>的帐户。
- **verify:** 验证密码在<filename>中<username>的凭据是否匹配。

opts

- **-?, --help:** 显示此帮助并退出。
- **-V, --version:** 显示版本信息和退出。

使用示例

执行以下命令将提示用户输入新的密码，然后将其加密并存储在
/etc/mysqlrouter/mysqlrouter.conf 配置文件的 DEFAULT 节下的 user_password 键中。

```
mysqlrouter_passwd set /etc/mysqlrouter/mysqlrouter.conf DEFAULT user_password
```

2.6 ibd2sd

命令功能

ibd2sdi 是一个实验性工具，用于从 InnoDB 表空间文件（.ibd）生成 SDI（Serialized Dictionary Information，序列化字典信息）文件。

命令格式

```
ibd2sdi [options] file_name1 [file_name2 file_name3 ...]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- **-h, --help** 显示帮助信息并退出。
- **-v, --version** 显示版本信息并退出。
- **-#, --debug[=name]** 输出调试日志。详细调试选项请参见 [The DBUG Package](#)。
- **-d, --dump-file=name** 将表空间 SDI 转存到用户传入的文件中。如果没有文件名，则默认为 stdout。
- **-s, --skip-data** 跳过检索 SDI 记录中的数据。只检索 id 和类型。
- **-i, --id=#** 检索与用户传入 id 匹配的 SDI 记录。
- **-t, --type=#** 检索与用户传入的类型相匹配的 SDI 记录。
- **-c, --strict-check=name** 由用户指定校验算法，支持 innodb、crc32、none。
- **-n, --no-check** 忽略校验和验证。
- **-p, --pretty** SDI 输出可读性更好的格式，默认开。

使用示例

```
ibd2sdi --dump-file=file_name ../data/test/t1.ibd
```

2.7 mysqlimport

命令功能

mysqlimport 用于将文本文件导入 TDSQL 数据库。

命令格式

```
mysqlimport [options] database textfile1 [textfile2 ...]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -u (--user=name) 用于连接到服务器的 TDSQL 帐户的用户名。
- -p (--password[=name]) 用于连接服务器的 TDSQL 帐户的密码。密码值是可选的。如果没有给出，请根据提示输入。
- -L (--local) 从客户端导入文件。

使用示例

```
mysqlimport -u root -p --local my_database my_table.txt
```

2.8 lz4_decompress

命令功能

lz4_decompress 用于解压缩使用 LZ4 算法压缩的文件。

命令格式

```
lz4_decompress input_file output_file
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

无

使用示例

```
lz4_decompress input.lz4 output.txt
```

2.9 mysqlshow

命令功能

mysqlshow 用于显示数据库、表和列的信息。

命令格式

```
mysqlshow [options] [db_name [tbl_name [col_name]]]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -u (--user=name) 用于连接到服务器的 TDSQL 帐户的用户名。
- -p (--password[=name]) 用于连接服务器的 TDSQL 帐户的密码。密码值是可选的。如果没有给出，请根据提示输入。
- --host=host_name, -h host_name: 连接到给定主机上的 TDSQL 服务器。
- --port=port_num, -P port_num: 端口号。

使用示例

```
mysqlshow -h 172.16.0.11 -P 4001 -u tdsqplcloud -p
```

2.10 mysqlpump

命令功能

mysqlpump 是一个数据库备份工具，类似于 mysqldump，但提供了更多的选项和更好的性能。

命令格式

```
mysqlpump [options] [db_name [tbl_name ...]]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -u (--user=name) 用户名。
- -p (--password[=name]) 密码。
- -B (--databases) 处理指定的数据库。
- -A (--all-databases) 处理所有数据库。

使用示例

```
mysqlpump -u root -p --all-databases > alldb_backup.sql
```

2.11 mysqlslap

命令功能

mysqlslap 是一个负载模拟工具，用于压力测试 TDSQL 服务器。

命令格式

```
mysqlslap [options]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -u (--user=name) 用户名。
- -p (--password[=name]) 密码。
- -c (--concurrency) 并发客户端数量。
- -i (--iterations) 测试迭代次数。

使用示例

```
mysqlslap -u root -p -c 50 -i 100
```

2.12 perror

命令功能

perror 用于解释系统错误代码或 MySQL 内部错误代码。

命令格式

```
perror [options] [errorcode ...]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

无

使用示例

```
perror 1231
```

2.13 mysqladmin

命令功能

mysqladmin 是一个执行管理操作的客户端，如创建和删除数据库，检查服务器状态等。

命令格式

```
mysqladmin [options] command [command-arg] [command [command-arg]] ...
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin

参数说明

- -u (--user=name) 用户名。
- -p (--password[=name]) 密码。

使用示例

```
mysqladmin -u root -p status
```

2.14 mysqld_safe

命令功能

mysqld_safe 是一个启动 mysqld 的脚本，它可以在 mysqld 崩溃时自动重启 mysqld，并记录错误信息。

命令格式

```
mysqld_safe [options]
```

命令路径

/data/tdsql_run/4001/mysql-server-8.0.24/bin/

参数说明

- --basedir=path TDSQL 安装目录。
- --datadir=path 数据目录路径。

使用示例

```
mysqld_safe --basedir=/usr/local/mysql --datadir=/usr/local/mysql/data
```


3 mysqlagent 主要执行文件

文件所在路径:

/data/tdsql_run/4001/mysqlagent/bin/

文件名	功能说明(常用使用场景)	使用方法
checkgtid	支持 DCN 同步目标实例和源实例一致性切换，通常由 DCN 同步工具调用。	<pre># ./checkgtid --help 假设 DCN 同步， 源实例为: set_1660567084_1 目标实例为: set_1660618467_6 可以参考下面命令进行检测，得到 gtid list is same 这样的结论这说明 同步源和目标实例的 gtid 一致。 ./checkgtid --agent- conf=../conf/mysqlagent_4001.xml -- name='set_1660567084_1' zookeeper timeout:10000 msec,msg timeout 30000 msec zookeeper timeout:10000 msec,msg timeout 30000 msec dcn master is set_1660567084_1, dcn slave is set_1660618467_6: set_1660567084_1: --> master and slave executed gtid list is same. set_1660618467_6: --> master and slave executed gtid list is same. 备注: checkgtid 在有两个跨集群备 dcn 时 执行失败，原因是对于不同 dcn slave，应该采用不同 zk 连接去 查，不能用一个全局的 zkClient， 10.3.22.4.x 版本已经解决，方法 为: 1. 添加 checkgtid --version 功能，</pre>

		<p>用来查看不同的 checkgtid 工具版本。</p> <p>2. 对于 1 拖 N 的 DCN 关系，对每个 slave DCN 都重新建立 zk 连接，保证跨集群情况下工具也能正常工作。</p>
agent_config	agent 配置工具，例如修改 agent 的端口等	<p>示例 2:</p> <pre>./agent_config --mode modify remove -- option="ocagent_port" -- value="8966"</pre> <p>示例 1: 开启 flashback 日志的备份，备份保留时间为 180 天,备份间隔默认为 3600 秒的示例。</p> <pre>./agent_config --mode=modify -- option="flashbacklog" -- value="openBackup=1&backupSave Days=180&backupIntervalSec=3600 "</pre>
alldump.sh	支持不同存储引擎数据库实例的压缩多线程流备份	<pre>./alldump.sh cnffile user pass unixsock tmpdir loginpass dstip dsttmpdir dstbindir srchost srcport limit oc_port</pre> <p>for example:</p> <pre>./alldump.sh /data/home/tdengine/mariadb- 10.0.10-prefix/etc/my_4001.cnf agent agent123456 /data/4001/prod/mysql.sock '/data/log/4001/dblogs/tmp' tdengine 10.187.143.217 '/data/4002/dbdata_raw/xtrabackuptm p' '/data/home/tdengine/mysqlagent/bin' 10.187.143.217 4001 0</pre>
autocheckhigh2	指定 pstack 开始执行分析的线程阈值	<pre>./autocheckhigh2 --conf my3318.cnf - -active 10 --wait 5 -- pstack_wait_thread 3 -- pstack_run_max 30</pre>
autorepair	自动修复	
binlogproducter	解析所在节点的 binlog 并	<pre>./binlogproducter ../conf/mysqlagen</pre>

	push 到消息队列	t_4001.xml
binlogproducer_percona	binlogproducer_percona 版本	./binlogproducer_percona ../conf/mysqlagent_4001.xml
cgroup_check.sh	检查指定 port 对应的 cgroup pid	./cgroup_check.sh port
checkdisk	磁盘检查	./checkdisk -p /data5
cold_mydumper_to_cos.sh	利用 mysqldumper 将数据冷备到腾讯云对象存储上 (Cloud Object Storage)	./cold_mydumper_to_cos.sh user pass unixsock hdfsdire filename
cold_mydumper_to_hdfs.sh	利用 mysqldumper 将数据冷备到 HDFS 上	./cold_mydumper_to_hdfs.sh user pass unixsock hdfsdire filename resultfile mydumperParam[optional]
cold_mydumper_to_localfile.sh	利用 mysqldumper 将数据冷备到本地路径	./cold_mydumper_to_localfile.sh user pass unixsock localdire filename resultfile mydumperParam[optional]
cold_xtrabackup_increment_to_cos.sh	利用_xtrabackup 将数据增备到腾讯云对象存储上 (Cloud Object Storage)	./cold_xtrabackup_increment_to_cos.sh cnffile user pass unixsock tmpdire limit hdfspath
cold_xtrabackup_increment_to_hdfs.sh	利用_xtrabackup 将数据增备到 HDFS 上	./cold_xtrabackup_increment_to_hdfs.sh cnffile user pass unixsock tmpdire limit hdfspath
cold_xtrabackup_increment_to_localfile.sh	利用_xtrabackup 将数据增备到本地路径	./cold_xtrabackup_increment_to_localfile.sh cnffile user pass unixsock tmpdire limit path
cold_xtrabackup_to_cos.sh	利用_xtrabackup 将数据全备到腾讯云对象存储上 (Cloud Object Storage)	./cold_xtrabackup_to_cos.sh cnffile user pass unixsock tmpdire limit hdfspath
cold_xtrabackup_to_hdfs.sh	利用_xtrabackup 将数据全备到 HDFS 上	./cold_xtrabackup_to_hdfs.sh cnffile user pass unixsock tmpdire limit hdfspath
cold_xtrabackup_to_localfile.sh	利用_xtrabackup 将数据全备到本地路径	cold_xtrabackup_to_localfile.sh cnffile user pass unixsock tmpdire limit hdfspath
cos_tool.py	由 cold_mydumper_to_cos.sh、cold_xtrabackup_increment_to_cos.sh、cold_xtrabackup_to_cos.sh 脚本调用	无需单独使用

dcntool	DCN 同步工具	./dcntool buildddcslave masterzklist masterzk_homepath master_set slavezklist slavezk_homepath slave_set
decompression	解压缩工具	./decompression file.compress
drop_backup	备份删除工具	./drop_backup --zkdir /tdsqlzk -- savedays 10
encrypt	加密工具	./encrypt 要加密的字符串
encrypt_tool	加密工具(只支持 0 和 1 的加密)	./encrypt_tool 0 111111
getlastgtid	获取最新的 GTID	./getlastgtid --mysqlctfile my.cnf -- binlogprefix mysql-bin
getlastgtidbytime	获取最新的 GTID 对应的 时间	./getlastgtid --mysqlctfile my.cnf -- binlogprefix mysql-bin
getNetSpeed.sh	获取指定 IP 的网速信息	./getNetSpeed.sh ip_addr
gtidlistcache_percona	获取指定 binlog cache 中 的 GTID 信息	./gtidlistcache_percona --mysqlctfile my.cnf --binlogprefix mysql-bin
gtidlist_percona	获取指定 binlog 中的 GTID 信息	/gtidlist_percona binlog.000016
gtidprint	获取连续的 gtid 信息	无需单独使用
gtidprint_percona	获取连续的 gtid 信息	无需单独使用
gtidtofilepos	根据 gtid 获取 binlog 的位 点信息	/gtidtofilepos_percona -- mysqlctfile="" -- binlogprefix="binlog" --gtidlist=""
gtidtofilepos_percona	根据 gtid 获取 binlog 的位 点信息	/gtidtofilepos_percona -- mysqlctfile="" -- binlogprefix="binlog" --gtidlist=""
initcgroupierarchy.sh	cgroup 初始化脚本	
install_single_proxy	安装 single proxy, 该文 件位于 /data/home/tdsql/tdsqlinstal l/目录下	
install_tdsq_zkparam	安装 zk, 该文件位于 /data/home/tdsql/tdsqlinstal l/目录下	
kms_tool.py	通过备份文件恢复实例 时, 用于解密加密的备份 数据	python ./kms_tool.py -- role="qcs::cam::uin/xxxxxxxx:role Name/kmsTDSQRole" -- secret_id="xxxxxxxxxxxxxxxxxxxx xxxxxxxx" --

		secret_key="xxxxxxxxxxxxxxxxxxxx xxxxxxxxxxxx" --region="ap- hongkong" -- ciphertext="CtFlCxx0+LilyvZ5xxqSI A/KEhVexxIGfBwzXiMShQZxxxW AUsHcfLQ0xxxDH2D49/nOw==k- fKVP3WxxxMW4jEkQ==k- zudP3Tz4jxrxxxKkuKU+0V/gVVax xIaRl/+83qCinaBxxU5e1MpW4q/IJ Kpxxb9N9/rO 5Es03fxxxn8Sjex6mnl+YKV1SMQo g+RJ1xxxNmwx/22hhHb/1B5LGpw B8tbXKD3gL0tZwSxxxUnONh5+6s sb2cxxxBhGj9oXtbL6OC74PuDO1 D/AsQ6qBxxxTSA68s8Q="
libiconv.so.2	动态链接库文件，不可执行的二进制程序文件，允许程序共享执行特殊任务所必需的代码和其他资源	
libmysqlclient.so.20	动态链接库文件，不可执行的二进制程序文件，允许程序共享执行特殊任务所必需的代码和其他资源	
lz4	解压缩工具	压缩： lz4 filename 解压缩： lz4 -d filename.lz4
migrate.sh	迁移工具	
mydumper	tdsql for mysql 多线程数据 dump 工具	./mydumper -u root -p password -B test -O /mydumper/
myloader	将 mysqldump 出来的 sql 以并行的方式进行恢复	./myloader -u root -p password -B test -d /mydumper
mysqlbinlog	获取当前二进制日志列表，mysqlbinlog 默认行为等	./mysqlbinlog [options] log-files
mysqlbinlog_flashback	生成反向 SQL 完成回退 _mariadb 版本	./mysqlbinlog_flashback --user arg -- pass arg --host arg --port arg --start- position arg --maxTimeRange arg
mysqlbinlog_flashback_percona	生成反向 SQL 完成回退 _percona 版本	./mysqlbinlog_flashback --user arg -- pass arg --host arg --port arg --start- position arg --maxTimeRange arg
mysql_insert_test	多线程 insert 工具	./mysql_insert_test -h 100.10.1.100 -

		P 15000 -u clouddba -pclouddba -t 8
mysql_param_modify	参数指定 set 修改工具	./mysql_param_modify --agent-conf arg --mode modify --name set
mysqlrecoverfromxtrabackup	xtrabackup 实例恢复工具	./mysqlrecoverfromxtrabackup --etcfile my.cnf --user dbuser --pass password
mysqlrecoverset	set 恢复工具	./mysqlrecoverset --setname arg --zkhhome [zookeeper root dir] --etcfile my.cnf --endtimestamp 需要恢复到的时间点
mysqlreport	agent 上报进程	./mysqlreport ../conf/mysqlagent_4001.xml
oc_encrypt	oc 加密工具	./oc_encrypt 要加密的字符
oc_tool	oc 配置工具	oc_tool [-fdkcvn] [-o option] [-P port] [-l ratelimit] [-t timeout] [-p password] [user@]hostname [command]
parse-processlist-log.sh	线程日志分析	./parse-processlist-log.sh since_time(int) until_time(int) 'processlist-log1 processlist-log2 ...'
procmonitor	进程监视器	./procmonitor --conffile arg
pt-query-digest	慢查询分析工具	./pt-query-digest /usr/local/mysql/data/slow.log
pt-table-checksum	校验主从数据一致性	./pt-table-checksum h=MASTER_HOST,u=repl_user,p='repl_pass',P=3306 \ --databases=d_test --tables=t_user,t_user_detail,t_user_group --nocheck-replication-filters
pt-table-sync	用于修复主从不一致的数据	对指定的表进行 sync: ./pt-table-sync --charset=utf8--ignore-databases=mysql,sys --databases=test_tdsq1 --tables=test_nu--no-check-slave dsn=u=root,p=root,h=源端 IP,P=3306dsn=u=root,p=root,h=目标端 IP,P=3306 --execute --print
pv	Pipe Viewer，管道的过程进度对用户透明	./pv -cN source access.log gzip pv -cN gzip > access.log.gz
reinstall_mysql	重装 mysqlagent	./reinstall_mysql --agent-conf ../conf/mysqlagent_4001.xml -

		-mysql-param innodb_page_size=1638 -- ignore_check_master 1 --setname newset
relayhbts_mariadb	查看从库延迟情况 _mariadb 版本	./relayhbts_mariadb --mysqltcfile my.cnf --relaylogprefix relay --delay 2
relayhbts_percona	查看从库延迟情况 _percona 版本	./relayhbts_percona --mysqltcfile my.cnf --relaylogprefix relay --delay 2
relaylogrepair	relaylog 修复工具	./relaylogrepair port option[relaylognum resetslave]
relaymasterpos_percona	relaylog 位置信息_percona 版本	./relaymasterpos_percona -- mysqltcfile my.cnf --relaylogprefix relay
repairsysdb_mariadb	sysdb 修复工具_mariadb 版	./repairsysdb_mariadb --relaylog arg --start-position arg --port arg
repairsysdb_percona	sysdb 修复工具_percona 版	./repairsysdb_percona --relaylog arg - -start-position arg --port arg
replacetemplate	模板替换	./replacetemplate templatefile outputfile req
restartbinlogproduct_cgrou p.sh	重启 binlogproduct 进程， 会调用 stopbinlogproduct.sh 和 startbinlogproduct_cgrou p.sh	./restartbinlogproduct_cgrou p.sh ../conf/mysqlagent_4001.xml
restartbinlogproduct.sh	重启 binlogproduct 进程， 会调用 stopbinlogproduct.sh 和 startbinlogproduct.sh	./restartbinlogproduct.sh ../conf/mysq lagent_4001.xml
restartbinlogtofile.sh	重启 binlogproduct 进程， 会调用 stopbinlogtofile.sh 和 startbinlogtofile.sh	./restartbinlogtofile ../conf/mysqlag ent_4001.xml
restartreport_cgrou p.sh	重启 report 进程，会调用 stopreport.sh 和 startreport_cgrou p.sh 脚本	./restartreport_cgrou p.sh ../conf/mysq lagent_4001.xml
restartreport.sh	重启 report 进程，会调用 stopreport.sh 和 startreport.sh 脚本	./restartreport.sh ../conf/mysqlagent_ 4001.xml
restarttransfer.sh	重启 transfer 进程，会调	./restarttransfer.sh ../conf/mysqlagent

	用 stoptransfer.sh 和 starttransfer.sh 脚本	_4001.xml
safeshell	用于 Ruby 开发的安全 shell 环境(SafeShell lets you execute shell commands and get the resulting output, but without the security problems of Ruby's backtick operator.)	./safeshell '[cmd]'
srm	tdsql 慢删除工具	./srm [filename]
srmnew	tdsql 慢删除工具	./srmnew [-c countPerSec(default is 4)] filelist
sshpas	用于非交互的 ssh 密码验证，使用 -p 参数指定明文密码，然后直接登录远程服务器	sshpas -p xxx ssh root@ip
startbinlogproduct_cgroup.sh	启动 binlogproduct 进程，会调用 startbinlogproduct.sh 脚本	./startbinlogproduct_cgroup.sh ../conf/mysqlagent_4001.xml
startbinlogproduct.sh	启动 binlogproduct 进程	./startbinlogproduct.sh ../conf/mysqlagent_4001.xml
startbinlogtofile.sh	启动 binlogtofile 进程	
startreport_cgroup.sh	启动 agent 上报，会调用到 startreport.sh 脚本	./startreport_cgroup.sh ../conf/mysqlagent_4001.xml
startreport.sh	启动 agent 上报	./startreport.sh ../conf/mysqlagent_4001.xml
stopbinlogconsumer.sh	停止 binlogconsumer 进程	./stopbinlogconsumer.sh
stopbinlogproduct.sh	停止 binlogproduct 进程	./stopbinlogproduct.sh
stopreport.sh	停止 agent 上报	./stopreport.sh ../conf/mysqlagent_4001.xml
tdsql_addto_cgroup	用于将 instance 加入到 cgroup	./tdsql_addto_cgroup --port mysql_instance_port \ --pid xxx --cgroup_type cpu,blkio
tdsql_cgroup_build	用于 cgroup 对 instance 配置编排	./tdsql_cgroup_build --instanceport instanceport --cpu_percent 20 --mode add modify init
tdsql_keyring	tdsql 秘钥配置工具，生成或者获取指定 mysql 实例的 keyring_file	./tdsql_keyring --keyring-file arg --server-uuid --mode fetch store

transfer_account	用来验证主备切换数据一致性，首先建立账户表，并完成初始化账户余额。然后，多线程转账，对于 update 成功的程序同步更新内存中的数据，对于 update 超时，程序尝试 select 二次查询的方式确认更新是否成功，最终比较数据库中的值和内存中的值是否一致。	./transfer_account -h 127.0.0.1 -P 4001 -u clouddba -p clouddba -t 10 -T 20 --tables 4
truncatebinlog	截断 binlog	./truncatebinlog endTime
truncatebinlog_percona	截断 binlog_percona 版本	./truncatebinlog_percona endTime
truncatFileFromEnd.sh	截断文件，从末尾开始	./truncatFileFromEnd.sh file reserveSize(MB)
uninstall_single_proxy	卸载 single proxy，该文件位于 /data/home/tdsql/tdsqlinstall/目录下	./uninstall_single_proxy
uninstall_single_tdsq	卸载 single tdsq，该文件位于 /data/home/tdsql/tdsqlinstall/目录下	./uninstall_single_tdsq
xatool	xa 分析工具	./xatool --begin-timestamp arg --end-timestamp arg
zkinfo_copy	zk 信息复制	./zkinfo_copy --src-zklist arg -src-zkdir arg --srcname arg --dstname arg

4 oc_agent 主要执行文件

文件所在路径:

/data/oc_agent/bin/

文件名	功能说明(常用使用场景)	使用方法
ewp_tdsq_l_oc	用于运行 oc_agent	/ewp_tdsq_l_oc /data/oc_agent/bin -- config=./conf/oc_agent.xml
ewp_tdsq_l_proc	用于运行 agent proc	./ewp_tdsq_l_proc /data/oc_agent/bin
ewp_tdsq_l_proc.err	ewp_tdsq_l_proc 进程错误日志	tail -100f /data/oc_agent/bin/ewp_tdsq_l_proc.err
oc_encrypt	密码加密	./oc_encrypt -[s c] password
oc_passwd	用来从密码文件(/etc/passwd)中读取一项用户数据, 该用户的数据以 passwd 结构返回.	./oc_passwd tdsq_l
oc_tool	oc 配置工具	usage: oc_tool [-fdkcvn] [-o option] [-P port] [-l ratelimit] [-t timeout] [-p password] [user@]hostname [command]
restart.sh	重启 ewp_tdsq_l_proc 进程	/data/oc_agent/bin/restart.sh
safeshell	用于 Ruby 开发的安全 shell 环境(SafeShell lets you execute shell commands and get the resulting output, but without the security problems of Ruby's backtick operator.)	非执行文件
start_agent.sh	启动 ewp_tdsq_l_proc 进程	/data/oc_agent/bin/start_agent.sh
stop_agent.sh	停止 ewp_tdsq_l_proc、ewp_tdsq_l_oc、AgentCollector 进程	/data/oc_agent/bin/stop_agent.sh

5 proxy 主要执行文件

文件所在路径:

/data/tdsql_run/15001/gateway/bin/

文件名	功能说明(常用使用场景)	使用方法
change_rootdir_all.sh	修改 zookeeper 根目录至 cgroup,xa,noshard 模式	./change_rootdir_all.sh instance /newrootdir/dir 1 1 1
change_rootdir_no_cgroup.sh	修改 zookeeper 根目录至非 cgroup 模式, 可以自定义是否启用 xa 模式	./change_rootdir_no_cgroup.sh instance /newrootdir/dir 0 1 1
change_rootdir.sh	修改 zookeeper 根目录至 cgroup 模式, 可以自定义是否启用 xa 模式	./change_rootdir.sh instance /newrootdir/dir 1 1 0
change_rootdir_to_noshard.sh	修改 zookeeper 根目录至 noshard 模式	./change_rootdir_to_noshard.sh instance /newrootdir/dir
change_rootdir_to_shard.sh	修改 zookeeper 根目录至 shard 模式	./change_rootdir_to_shard.sh instance /newrootdir/dir
dcagent_tokafka	代理进程, 用于和 kafka 交互	
load_data	数据加载	./load_data mode0/mode1 proxy_host proxy_port user password db_table shardkey_index/auto file terminate enclosed [split_size]
md5	文件校验	./md5
mysql-proxy	mysql 代理, 用于实现读写分离、负载均衡	./mysql-proxy /data/tdsql_run/15065/gateway/conf/instance_15065.cnf
mysql-proxyd	用于启停 proxy 实例	./mysql-proxyd instance {start stop restart status}
mysql-proxy_old	mysql-proxy 的旧版本	./mysql-proxyd_old instance {start stop restart status}
replacetemplate	模板替换	./replacetemplate templatefile outputfile req
restartbinlogproduct.sh	重启 binlogproduct.sh	./restartbinlogproduct.sh
restart_cgroup.sh	重启指定的 proxy instance	./restart_cgroup.sh instance_15065

restartdcagent_tokafka_cgroup.sh	cgroup 下重启 agent_tokafka	./restartdcagent_tokafka_cgroup.sh
restartdcagent_tokafka.sh	重启 agent_tokafka	./restartdcagent_tokafka.sh
restart_proxy.sh	重启 proxy instance	./restart_proxy.sh instance_test
restart_router_cgroup.sh	cgroup 下重启 proxy instance	./restart.sh instance_test
restart.sh	重启 proxy instance	./restart.sh instance_test
router_update	router 更新	./router_update /data/tdsql_run/15065/gateway/conf/instance_15065.cnf
safeshell	用于 Ruby 开发的安全 shell 环境(SafeShell lets you execute shell commands and get the resulting output, but without the security problems of Ruby's backtick operator.)	非执行文件
srm	tdsql 慢删除工具	./srm [filename]
srmnew	tdsql 慢删除工具	./srmnew [-c countPerSec(default is 4)] filelist
start_cgroup.sh	cgroup 下启动指定的 proxy instance	./start_cgroup.sh instance_15065
startdcagent_tokafka_cgroup.sh	cgroup 下启动 agent_tokafka	./startdcagent_tokafka_cgroup.sh
startdcagent_tokafka.sh	启动 agent_tokafka 进程	./startdcagent_tokafka.sh
start_proxy.sh	使用 mysql-proxyd 启动指定 instance	./start_proxy.sh instance_test
start_router_cgroup.sh	cgroup 下启动指定的 proxy instance	./start_router_cgroup.sh instance_test
start_router.sh	启动指定 proxy instance	./start_router.sh instance_test
start.sh	启动指定的 mysql-proxy instance	./start.sh instance_test
stopdcagent_tokafka.sh	停止 agent_tokafka	./stopdcagent_tokafka.sh conf
stop_proxy.sh	停止指定的 proxy instance	./stop_proxy.sh instance_15065
stop_router.sh	停止指定的 router	./stop_router.sh instance_15065
stop.sh	停止指定的 proxy instance	./stop.sh instance_15065
stop_version.sh	停止指定版本的 proxy	./stop_version.sh instance_3337 pid
switchdcagent_tokafka.sh	agent_tokafka 模式切换	./switchdcagent_tokafka.sh conf [openDC/closeDC]
tdsql_addto_cgroup	用于将 instance 加入到	./tdsql_addto_cgroup --port

	cgroup	mysql_instance_port \ --pid xxx -- cgrouptype cpu,blkio
tdsql_cgroup_build	用于 cgroup 对 instance 配置 编排	./tdsql_cgroup_build --instanceport instanceport --cpu_percent 20 --mode add modify init
update_version.sh	版本更新	./update_version.sh instance_15065
zk_tool	zookeeper 管理工具	./zk_tool del_node/del_node_force zkiplist path

6 zookeeper 主要执行文件

文件所在路径:

/data/application/zookeeper/bin/

文件名	功能说明(常用使用场景)	使用方法
zkCleanup.sh	清理旧的事务日志和快照	./zkCleanup.sh 参数 1 -n 参数 2。其中： <ul style="list-style-type: none">参数 1, zk data 目录, 即 zoo.cfg 文件中 dataDir 值。参数 2, 保存最近的多少个快照。
zkCli.cmd	zk 的简易客户端工具, 可以对 zookeeper 服务端数据进行各种操(Windows 平台)	./zkCli.cmd -server IP:port
zkCli.sh	zk 的简易客户端工具, 可以对 zookeeper 服务端数据进行各种操(Linux 平台)	./zkCli.sh -server IP:port
zkEnv.cmd	设置 zk 环境变量 (Windows 平台)	设置 zookeeper 启动时的环境变量, 这个脚本不要单独执行,它需要嵌入到 zkServer.cmd 或者其他脚本中使用
zkEnv.sh	设置 zk 环境变量 (Linux 平台)	设置 zookeeper 启动时的环境变量, 这个脚本不要单独执行,它需要嵌入到 zkServer.sh 或者其他脚本中使用
zkServer.cmd	zookeeper 服务器的启动停止重启和状态查询(Windows 平台)	./zkServer.cmd {start start-foreground stop restart status upgrade print-cmd}
zkServer.sh	zookeeper 服务器的启动停止重启和状态查询(Linux 平台)	./zkServer.sh {start start-foreground stop restart status upgrade print-cmd}
zkTxnLogToolkit.cmd	ZooKeeper 附带的命令行工具, 能够恢复带有损坏 CRC 的事务日志条目(Windows 平台)	./zkTxnLogToolkit.cmd log.100000001
zkTxnLogToolkit.sh	ZooKeeper 附带的命令行工具, 能够恢复带有损坏 CRC 的事务日志条目(Linux 平台)	./zkTxnLogToolkit.sh log.100000001

7 Scheduler/Manager 命令参考

7.1 scheduler/manager 主要执行文件

文件所在路径:

/data/application/scheduler/bin

文件名	功能说明(常用使用场景)	使用方法
agent_config	agent 全局配置修改	./agent_config --mode modify remove --option ocagent_port --value 8966
alter_slave_threads	修改 MariaDB 并行复制线程数量（废弃，勿用）	弃用
backupZkInfo	备份 ZK 元数据信息	./backupZkInfo
base64_decrypt	base64 解密	./base64_decrypt [passwd] [key]
base64_encrypt	base64 加密	./base64_encrypt [plaintext] [key]
change_proxygroup	老版本修改网关组 reserved_count（目前已废弃）	弃用
change_proxyport	老版本升级网关组用（已废弃）	弃用
check_and_create_pidmap	检查主资源 DB 的 pidmap 节点，节点不存在则创建	./check_and_create_pidmap [ip]
check_port	检查资源节点端口状态是否正确	./check_port all
check_sets_is_specid	检查实例是否为按照 specid 比例购买	./check_sets_is_specid [set id]
close_percona_encrypt	关闭 percona 加密	./close_percona_encrypt setname/groupid
collection_mariadb_zk_mistake	mariadb/zk 做错误信息采集	./collection_mariadb_zk_mistake
create_db_stat	创建 db 状态信息	./create_db_stat config_path config_path setname
create_lvs_mode	创建 lvs 模式(可以指定 Tunneling 或者 Non-arp device 模式)	./create_lvs_mode [MODE] (TUN/DR)

create_stopdir	创建 stopdir	./create_stopdir
dbinsert	数据插入	./dbinsert ip port user passwd rule
delete_unfinish_jobs	删除未完成任务（已废弃）	弃用
del_set	删除指定 set	./del_set subrootdir setname
fix_appid	功能未知，无法使用，已废弃	弃用
fix_logdisk_by_setname	修正实例的 logdisk 资源	./fix_logdisk_by_setname groupid/nos hard setname
fix_machine	功能未知，无法使用，已废弃	弃用
fix_pidmap	修复 pidmap 节点（已废弃）	弃用
fix_setrun_mem	修正 DB 的内存资源	./fix_setrun_mem master_ip
get_hb	用于取得节点心跳信息	./get_hb subrootdir all host
get_set_info	获取指定 set 相关信息	./get_set_info subrootdir [setname]
get_sets	获取所有 set 相关信息	./get_sets
lvs_tool	lvs 管理工具	./lvs_tool queryLvsMode all
manager	启动 manager	./manager
manager_mig	与 manager 程序完全相同，仅名字区别	./manager
manaul_switch	主从切换	/manaul_switch user subrootdir setname master [special_slave]
modify_manager_variable	manager 变量设置	./modify_manager_variable --offlinehost_timeout 10
modify_specinfo_auto	节点规格信息更改 (CPU/MEMORY/DATA_DISK/LOG_DISK)	./modify_specinfo_auto --machine Machine type
monitor_sche	监控 scheduler 程序	./monitor_sche
newslicejob	用于手动发起垂直扩容任务（建议使用 OSS 发起任务）	./newslicejob subrootdir myuser mypassword srcset dstset begin_time end_time
noshard_create_set	创建非分布式 set	./noshard_create_set subrootdir user setname masterip_port
noshard_privs	非分布式 set 权限管理_旧版本	./noshard_privs setname grant user password password_mode 1 db * table * privileges SELECT, INSERT, UPDATE, DELETE
noshard_privs-new	非分布式 set 权限管理_新	./noshard_privs-new setname grant user

	版本	password password_mode 1 db * table * privileges SELECT, INSERT, UPDATE, DELETE
oc_tool	参考 oc agent, 同一工具	
proxy_machine_remove	proxy 机器删除	./proxy_machine_remove --ip
proxy_machine_report	proxy 机器信息上报	./proxy_machine_report --ip
proxy_tool	proxy_tool 管理工具, 例如可以停止指定 task	./proxy_tool toptask type groupid(nos hard) setid(jobid)
recoverZkInfo	将备份的 zookeeper 数据恢复到 zookeeper	
replacehost	节点替换	./replacehost subrootdir setname jobtype[add, replace, delete] dsthost srchost[only replace host need]
resource_fport	功能未知, 源码丢失, 废弃	弃用
resource_fport_mig	功能未知, 源码丢失, 废弃	弃用
resource_report	资源信息上报	./resource_report --ip "host ip"
safeshell	用于 Ruby 开发的安全 shell 环境(SafeShell lets you execute shell commands and get the resulting output, but without the security problems of Ruby's backtick operator.)	非执行文件
scan_db_port	扫描 db 端口	./scan_db_port
scheduler	启动 scheduler	./scheduler /data/application/scheduler/bin
sshpas	用于非交互的 ssh 密码验证, 使用 -p 参数指定明文密码, 然后直接登录远程服务器	sshpas -p xxx ssh root@ip
sshpas_pack.sh	远程服务器管理工具	./sshpas_pack.sh [password] [user] [ip] [port] <command> <oc_tool path>
sshpas_pack.sh_old	用于 Keeper 调用 oc_tool 工具发送指令的脚本 (已废弃)	./sshpas_pack.sh_old [password] [user] [ip] [port] <command> <oc_tool path>

start_manager.sh	启动 scheduler/manager	./start_manager.sh
stop_manager.sh	停止 scheduler/manager	./stop_manager.sh
update	升级 MIG 版本，例如从 MIG7.1 升级到 MIG8.0	./update version
zk_operate	zk 配置工具	./zk_operate get path 'content'
zk_recursion_print	递归打印根目录树	./zk_recursion_print zklist rootdir
zk_recursion_setacl	递归设置目录 ACL 权限	./zk_recursion_setacl zklist rootdir 0/1 0 mean set acl by word:anyone 1 mean set acl by tdsql user
zk_sets	用于实例数据拷贝（源码丢失，废弃）	弃用
zone_init	zone 初始化，需要先停止 scheduler	./zone_init
zone_privs	groupshard 版本授权工具，非 ip 透传模式	./zone_privs zone myuser mypassword dropdatabase db
zone_privs-new	groupshard 版本授权工具，ip 透传模式	./zone_privs-new zone myuser mypassword dropdatabase db

7.2 manual_set

命令功能

manual_set 用于实例管理，设置实例免切等。

命令路径

/data/application/scheduler/bin

命令格式

强制主备切换

#集中式实例：

```
./manual_set force_sw noshard setname master_host slave_host
```

#分布式实例：

```
./manual_set force_sw groupid setname master_host slave_host
```

- setname 要切换的 set 名。
- master_host 要切换的主机。
- slave_host 要切换到的备机。

显示 applylog 配置信息

```
./manual_set list_applylog all
```

操作示例：

```
$ ./manual_set list_applylog all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
wait_applylog value: 10
manual_set ok
```

修改 applylog 时间

```
./manual_set modify_applylog time(s)
```

显示 kickout 时间

```
./manual_set list_kickout all
```

操作示例：

```
$ ./manual_set list_kickout all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
```

```
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:.,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
kickout_delay value: 300
manual_set ok
```

修改 kickout 时间

```
./manual_set modify_kickout time(s)
```

显示免切换配置

```
./manual_set list_noswitch all
```

操作示例：

```
$ ./manual_set list_noswitch all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:.,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
noswitch time value: 3600
manual_set ok
```

修改免切换时间

```
./manual_set modify_noswitch time(s)
```

显示免切换日志入库时间

```
./manual_set list_noswlogtime all
```

操作示例：

```
$ ./manual_set list_noswlogtime all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:.,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
noswitch time value: 60
manual_set ok
```

修改免切换日志入库时间

```
./manual_set modify_noswlogtime time(s)
```

修改 set 的 host idc 名

#集中式实例：

```
./manual_set modify_idc noshard setname hosts idc_name
```

#分布式实例：

```
./manual_set modify_idc groupid setname hosts idc_name
```

- setname 修改的 set 名。
- hosts 要修改的 host。
- idc_name 修改的 idc 名。

修改 set 的 host 权重

#集中式实例：

```
./manual_set modify_weight noshard setname hosts weight
```

#分布式实例：

```
./manual_set modify_weight groupid setname hosts weight
```

- setname 修改的 set 名。
- hosts 要修改的 host。
- weight 修改的 idc 权重。

修改 set 退化以及退化时间

#集中式实例：

```
./manual_set modify_detime noshard setname degrade[0/1] time(s)
```

#分布式实例：

```
./manual_set modify_detime groupid setname degrade[0/1] time(s)
```

- setname 设置的 set 名。
- degrade[0/1] 是否退化。0：不退化；1：退化。
- time(s) 退化时间。

修改 set 的状态

#集中式实例：

```
./manual_set modify_status noshard setname status
```

#分布式实例：

```
./manual_set modify_status groupid setname status
```

- setname 设置的 set 名。
- status 状态值。

修改 set 最大丢失心跳时间

#集中式实例：

```
./manual_set modify_smaxlosthb noshard setname time(s)
```

#分布式实例：

```
./manual_set modify_smaxlosthb groupid setname time(s)
```

- setname 设置的 set 名。
- time(s) 设置时间。

修改 set 最大延迟时间

#集中式实例：

```
./manual_set modify_smaxdelay noshard setname time(s)
```

#分布式实例：

```
./manual_set modify_smaxdelay groupid setname time(s)
```

- setname 设置的 set 名。
- time(s) 设置时间。

修改 set applylog 时间

#集中式实例：

```
./manual_set modify_sapplylog noshard setname time(s)
```

#分布式实例：

```
./manual_set modify_sapplylog groupid setname time(s)
```

- setname 设置的 set 名。
- time(s) 设置时间。

修改 setinfo 节点 kickout_delay 字段

适用于高可用场景，备节点与主节点数据同步断开多长时间设置为 watch 节点。

系统优先从 keeper 配置文件读取，默认为 300 秒；如果是 0，则从 zk setinfo@setname 节点读取。

#集中式实例：

```
./manual_set modify_skickout noshard setname value
```

#分布式实例：

```
./manual_set modify_skickout groupid setname value
```

- setname 设置的 set 名。
- value 设置时间，单位为秒。

修改 set 免切时间

#集中式实例：

```
./manual_set modify_snoswitch noshard setname time(s)
```

#分布式实例：

```
./manual_set modify_snoswitch groupid setname time(s)
```

- setname 设置的 set 名。
- time(s) 设置时间。

修改 set 自动免切时间

#集中式实例：

```
./manual_set modify_anoswitch noshard setname sw_time
```

#分布式实例：

```
./manual_set modify_anoswitch groupid setname sw_time
```

- setname 设置的 set 名。
- sw_time 设置时间。

删除 set 自动免切节点

#集中式实例：

```
./manual_set delete _anoswitch noshard setname all
```

#分布式实例：

```
./manual_set delete _anoswitch groupid setname all
```

- setname 设置的 set 名。
- all 所有。

显示 set 自动免切状态

#集中式实例：

```
./manual_set list _anoswitch noshard setname
```

#分布式实例：

```
./manual_set list _anoswitch groupid setname
```

添加 set 手动免切节点

#集中式实例：

```
./manual_set add _mnoswitch noshard setname type time(h)
```

#分布式实例：

```
./manual_set add _mnoswitch groupid setname type time(h)
```

- setname 设置的 set 名。
- type 免切类型。0 永久免切； 1 时间段免切。
- time(h) 时间端免切时间。

修改 set 手动免切状态

#集中式实例：

```
./manual_set modify _mnoswitch noshard setname type=xxx ...
```

#分布式实例：

```
./manual_set modify _mnoswitch groupid setname type=xxx ...
```

- setname 设置的 set 名。
- type=xxx ... 修改类型或者时间。

删除 set 手动免切节点

#集中式实例：

```
./manual_set delete _mnoswitch noshard setname
```

#分布式实例：

```
./manual_set delete _mnoswitch groupid setname
```

setname 设置的 set 名。

显示 set 手动免切状态

#集中式实例：

```
./manual_set list _mnoswitch noshard setname
```

#分布式实例：

```
./manual_set list _mnoswitch groupid setname
```


- 参数说明：
setname 设置的 set 名。
- 操作示例：

```
$ ./manual_set list_mnoswitch noshard set_1722067317_1
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
List a node named [set_1722067317_1] in confailmanu for noshard.

path: /tdsqlzk/confailstatus/confailmanu@set_1722067317_1
content: {"timestamp":"1722067377","trace_id":"c0a87a011722067236829103827912","type":1}
noswitch: false, switch time: 2024-07-27 16:02:57

Affected nodes: 32642
manual_set ok
```

查看实例慢日志存放的 DB 信息

获取的是 setconfig@setname 节点的 slow_log 字段。

```
#集中式实例
./manual_set list_slowlog noshard setname
#分布式实例
./manual_set list_slowlog groupid setname
```

- 参数说明：
setname 设置的 set 名。
- 操作示例：

```
$ ./manual_set list_slowlog noshard set_1722067317_1
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
slow_log content
{
  "db" : "tdsqlpcloud_monitor",
  "host" : "1.1.1.1",
  "password" : "n59fp0J+*skp1K",
  "port" : "15001",
  "trace_id" : "c0a87a011722067236829103827912",
```

```
"username" : "tdsqlpcloud"
}
```

```
day: 30
manual_set ok
```

修改 slowlog 备份天数

#集中式实例

```
./manual_set modify_slowlogday noshard setname days
```

#分布式实例

```
./manual_set modify_slowlogday groupid setname days
```

- setname 设置的 set 名。
- days 天数。

修改 slowlog 配置

#集中式实例

```
./manual_set modify_slowlogconfig noshard setname "db=xxx;host=xxx..."
```

#分布式实例

```
./manual_set modify_slowlogconfig groupid setname "db=xxx;host=xxx..."
```

- setname 设置的 set 名。
- "db=xxx;host=xxx..." 修改 db, host 等。

修改 set 强同步状态

#集中式实例

```
./manual_set modify_sync noshard setname sync_type
```

#分布式实例

```
./manual_set modify_sync groupid setname sync_type
```

- setname 设置的 set 名。
- sync_type 同步模式。
 - 1: 强同步
 - 0: 异步
 - 2: 同 IDC 异步, 异 IDC 强同步。

7.3 modify_election

命令功能

modify_election 用于设置选举节点, 设置备机是否为 watch 节点。

命令格式

```
#集中式实例：
./modify_election noshard set host_ip_port is_election
#分布式实例：
./modify_election groupid set host_ip_port is_election
```

命令路径

/data/application/scheduler/bin

参数说明

- set 要设置的 set 名。
- host_ip_port 要设置的 host，ip+port 格式。
- is_election 是否选举节点。0 为非选举节点；1 为选举节点。

7.4 resource_tool

命令功能

resource_tool 用于 TDSQL 集群资源管理，可以添加删除实例。

命令路径

/data/application/scheduler/bin

命令格式

给 set 添加一个备机

```
./resource_tool addhost pattern set_name ip idc_name sync_type
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- set_name 要添加的 Set 名。
- ip 要添加的机器 IP。
- idc_name 要添加的 IP 的 IDC 名。
- sync_type 添加的备机同步模式。

给 set 删除一个备机

```
./resource_tool delhost pattern set_name ip release_flag
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- set_name 要删除备机的 set 名。
- ip 要删除的机器 IP。
- release_flag
 - 1（默认值）：登录服务器删除进程。

- 0：不登录服务器。

给 set 替换一个备机

```
./resource_tool relhost pattern set_name dst_ip idc_name src_ip sync_type
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- set_name 要替换的 set 名。
- dst_ip 要替换为的机器 IP。
- idc_name 要替换为的机器 IP 的 IDC 名。
- src_ip 要被替换的机器 IP。
- sync_type 替换后备机同步模式。

给机器组添加一台机器

```
./resource_tool addip src_ip dst_ip
```

- src_ip 原机器组中 master 机器 IP。
- dst_ip 要添加的机器 IP。

给机器组删除一台机器

```
./resource_tool delip src_ip dst_ip
```

- src_ip 原机器组中 master 机器 IP。
- dst_ip 要删除的机器 IP。

给机器组替换一台机器

```
./resource_tool replaceip src_ip dst_ip
```

- src_ip 要被替换的机器 IP。
- dst_ip 新加入的机器 IP。

从资源池中卸载资源

```
./resource_tool unload_ip ip
```

- ip 要卸载机器 IP。

将 disable 和 isolate 资源恢复

```
./resource_tool recover_res ip(master_ip)
```

- ip 有 disable 和 isolate 端口机器组中主机 IP。

强制中进行中的锁定任务

```
./resource_tool stop_lockInstance pattern taskid
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- taskid 锁定任务 id 号。

强制停止进行中的垂直扩容任务

```
./resource_tool stop_expandInstance pattern taskid
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- taskid 垂直扩容任务 id 号。

强制中止进行中的回档任务

```
./resource_tool stop_retreatInstance pattern taskid
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- taskid 回档任务 ID 号。

查询当前资源池状态

```
./resource_tool status_res all
```

查询当前集群下所有 set

```
./resource_tool list_sets all
```

操作示例：

```
$ ./resource_tool list_sets all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
noshard rootdir:
  setname: set_1722067317_1
group rootdir: group_1722482858_16
  setname: set_1722483048_3
  setname: set_1722483012_1
resouce_tool ok
```

查询 setrun 状态

```
./resource_tool status_setrun pattern all|setname
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- all|setname 所有 set 或者具体 set。

查询 setinfo 状态

```
./resource_tool status_setinfo pattern all|setname
```

- pattern 如果是 noshard 则直接输入 noshard，如果是 groupshard 则输入 groupid。
- all|setname 所有 set 或者具体 set。

查询当前集群主备切换配置信息

```
./resource_tool show_recovery all
```

操作示例：

```
$ ./resource_tool show_recovery all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
```

recovery node content:

```
{"check_mreport_timestamp":"on","consist":"on","degrade_timeout":"20","highload_threshold":"3",
"kickout_delay":"300","master_agent_alive_timeout":"600","max_timeout":"200","maxdelay":"100",
"maxlosthbtime":"20","no_switch_time":"3600","noswitch_logtime":"60","nset_noswitch_time":"60",
"optimize_allow_timestamp":"180","optimize_losthb":"on","optimize_sd_timestamp":"3","slave_agent_alive_timeout":"600",
"sw_checktime":"60","wait_applylog":"10"}
resouce_tool ok
```

手动删除当前集群主备切换配置信息

```
./resource_tool clean_recovery all
```

添加 IDC 信息

```
./resource_tool idc_add name weight
```

- name IDC 名。
- weight IDC 的权重。

删除 IDC 信息

```
./resource_tool idc_del name
```

name IDC 名。

修改 IDC 权重

```
./resource_tool idc_mod name new_weight
```

- name IDC 名。
- new_weight IDC 新权重。

查询 IDC 信息

```
./resource_tool idc_query all
```

添加机型信息

```
./resource_tool spec_add machine cpu data_disk log_disk mem data_dir log_dir install_dir
module_dir reserve_percent
```

- machine 机型。
- cpu 核数。
- data_disk 数据盘大小。
- log_disk 日志盘大小。
- mem 内存大小。
- data_dir 数据盘位置。
- log_dir 日志盘位置。
- install_dir DB 安装包位置(公共目录)。
- module_dir DB 安装位置（分离目录）。
- reserve_percent 资源预留百分比。

添加机型（新版本）

```
./resource_tool spec_add machine cpu mem install_dir module_dir reserve_percent
```

- machine 机型
- cpu 核数
- mem 内存大小
- install_dir DB 安装包位置(公共目录)
- module_dir DB 安装位置 (分离目录)
- reserve_percent 资源预留百分比

给某个机型添加数据盘/日志盘

```
./resource_tool spec_disk_add machine type[0|1] data_dir data_disk[log_dir log_disk]
```

- machine 机型。
- type[0|1]
 - 0: 代表后面两个参数是数据盘配置。
 - 1: 代表后面是日志盘配置。
- data_dir 数据盘位置。
- data_disk 数据盘大小。
- log_dir 日志盘位置。
- log_disk 日志盘大小。

删除机型信息

```
./resource_tool spec_del machine
```

machine 机型

修改机型信息

```
./resource_tool spec_mod machine data_disk=xxx ...
```

- machine 机型
- data_disk|log_disk|mem|cpu 修改数据盘大小|日志盘大小|内存|cpu

获取所有机型信息

```
./resource_tool spec_query all
```

操作示例:

```
$ ./resource_tool spec_query all
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[1],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
sldbconf:master[192.168.225.44:15001],db:tdsqlpcloud_monitor,user:tdsqlpcloud,pass:,days:30
dbconf:master[192.168.225.44:15001],slave[192.168.225.55:15001],m_timeout[5],db:tdsqlpcloud_
monitor,user:tdsqlpcloud,pass:
zookeeper iplist:tdsql_sh9u32w9w_zk1:2118,rootdir:/tdsqlzk
get /tdsqlzk/specinfo
{
  "spec" : [
    {
```

```
"cpu" : 100,
"data_disk" : [
  {
    "dir" : "/data",
    "trace_id" : "c0a87a011722067199433100113507",
    "value" : 1000
  }
],
"install_dir" : "/data/home/tdsql/tdsqlinstall",
"limit_dbnum" : "0",
"log_disk" : [
  {
    "dir" : "/data",
    "trace_id" : "c0a87a011722067199433100113507",
    "value" : 1000
  }
],
"look_as_numa" : 0,
"machine" : "PROXY",
"mem" : 1000,
"module_dir" : "/data/tdsql_run",
"rsv_percent" : 1,
"trace_id" : "c0a87a011722067199433100113507"
},
{
  "cpu" : 800,
  "cpu_excess_ratio" : 0,
  "data_disk" : [
    {
      "dir" : "/data1/tdengine/data",
      "value" : 70000
    }
  ],
  "data_excess_ratio" : 0,
  "description" : "",
  "install_dir" : "/data/home/tdsql/tdsqlinstall",
  "limit_dbnum" : "0",
  "log_disk" : [
    {
      "dir" : "/data1/tdengine/log",
      "value" : 40000
    }
  ]
}
```



```
    ],  
    "log_excess_ratio" : 0,  
    "look_as_numa" : 0,  
    "machine" : "TS80",  
    "mem" : 10000,  
    "mem_excess_ratio" : 0,  
    "module_dir" : "/data/tdsql_run",  
    "rsv_percent" : 1,  
    "trace_id" : "c0a87a011722067226679101127912"  
  }  
],  
"trace_id" : "c0a87a011722067226679101127912"  
}  
  
resouce_tool ok
```

8 oss 主要执行文件

文件所在路径:

/data/application/oss/bin/

文件名	功能说明(常用使用场景)	使用方法
oss_server	提供 http 操作 TDSQL 的接口	/data/application/oss/boot/start.sh /data/application/oss/boot/stop.sh
compare_schema.py	oss 表结构升级校验工具	/data/application/oss/bin/compare_schema.py

9 monitor(collector&analyze)主要执行文件

collector 组件

文件所在路径:

/data/application/tdsql_collector/bin

文件名	功能说明(常用使用场景)	使用方法
encrypt_tool	加密工具	./encrypt_tool 0 12345
monitor.sh	collector 进程监控脚本	由 crontab 定期执行
restart.sh	重启 CollectorApplication	./restart.sh
start.sh	启动 CollectorApplication	./start.sh
stop.sh	停止 CollectorApplication	./stop.sh
tnm2_link.sh	创建指定文件的软链接	
upgrade.sh	升级 CollectorApplication 脚本	./upgrade.sh
up.sh	升级工具	弃用

analyze 组件

文件所在路径:

/data/application/tdsql_analysis/bin

文件名	功能说明(常用使用场景)	使用方法
alarm_test.sh	告警脚本	./alarm_test.sh dir msg reciever mLevel clusterIds alarmIds
confsync.sh	启动脚本	java 参数初始化，不单独使用
encrypt_tool	加密工具	./encrypt_tool 0 12345
monitor.sh	analysis 进程监控脚本	由 crontab 调度
restart.sh	重启 analysisApplication	./restart.sh
start.sh	启动 analysisApplication	./start.sh
stop.sh	停止 analysisApplication	./stop.sh
upgrade.sh	升级 analysisApplication 脚本	./upgrade.sh

10 clouddba 主要执行文件

文件所在路径:

/data/application/clouddba/bin/

文件名	功能说明(常用使用场景)	使用方法
analyze_monitorlog.py	监控日志分析	python analyze_monitorlog.py
clouddba-client.py	clouddba 客户端工具	./clouddba-client.py [-h] [-c CONFIG] [-H HOST] [-p PORT] -q REQUEST
clouddba-client.py.bak	clouddba-client.py 的旧版本	./clouddba-client.py.bak [-h] [-c CONFIG] [-H HOST] [-p PORT] -q REQUEST
dt2ts.sh	date 类型转换为 timestamp 类型	./dt2ts.sh "2021-03-30 19:12:36"
git-version	代码版本信息	非执行文件
master-switch-analysis.py	节点切换原因探测	usage: master-switch-analysis.py [-h] [-d DUMP_LOG_DIR] [-st SWITCH_TIME] [-o OUTPUT] [-dn DUMP_LOG_DIR_NAME] [--since_time SINCE_TIME] [--until_time UNTIL_TIME] [--getdir]
oc_tool	oc 配置管理工具	usage: oc_tool [-fdkcvn] [-o option] [-P port] [-l ratelimit] [-t timeout] [-p password] [user@]hostname [command]
parse-processlist-log.sh	分析指定时间段的慢查询	USAGE: ./parse-processlist-log.sh since_time(int) until_time(int) 'processlist-log1 processlist-log2 ...'
password_encrypt	对指定密码加密	./password_encrypt your_password
proxy_log_analyzer	可用于分析 proxy 的常规日志和事务型日志	./proxy_log_analyzer -t 1
restart.sh	重启扁鹊模块	./restart.sh
safeshell	用于 Ruby 开发的安全 shell 环境	./safeshell '[cmd]'

	(SafeShell lets you execute shell commands and get the resulting output, but without the security problems of Ruby's backtick operator.)	
sql-optimizer	sql 优化模块，可以给出指定 SQL 的优化建议	./sql-optimizer -utdsqcloud -h172.27.32.110 -P15065 -p'tdsqcloud' -Dtdsqcloud -q 'select * from t_db_parameter_config_items'
srm	tdsql 慢删除工具	./srm [filename]
start.sh	启动扁鹊模块	./start.sh
stop.sh	停止扁鹊模块	./stop.sh
tags	用于代码管理	非执行文件
tdsql-diagnosis	用于 tdsq 问题诊断	USAGE: tdsq-diagnosis options: Options: -h [--help] show usage message -v [--version] show version info -c [--config] arg config file path -d [--daemon] run as daemon
tdsq_inspection.py	tdsq 巡检脚本	基础巡检： ./tdsq_inspection.py -b
ts2dt.sh	timestamp 类型转换为 date 类型	
update-config.py	配置更新脚本	./update-config.py diagnosis_conf_file

11 kafka 主要执行文件

文件所在路径:

/data/application/kafka/bin/

文件名	功能说明(常用使用场景)	使用方法
connect-distributed.sh	用于启动多节点的 Distributed 模式的 Kafka Connect 组件	./connect-distributed.sh config/connect-distributed.properties
connect-standalone.sh	用于启动单节点的 Standalone 模式的 Kafka Connect 组件	./connect-standalone.sh config/connect-standalone.properties connector1.properties [connector2.properties ...]
kafka-acls.sh	Kafka ACL 权限管理客户端工具	设置哪些用户可以访问 Kafka 的哪些 TOPIC 权限: ./kafka-acls.sh \ --authorizer-properties zookeeper.connect=zookeeper:port \ --add \ --cluster \ --operation alter \ --deny-principal User:ANONYMOUS
kafka-broker-api-versions.sh	用于验证不同 Kafka 版本之间服务器和客户端的适配性(需要指定 Kafka Broker)	./kafka-broker-api-versions.sh \ --bootstrap-server kafka-host:port
kafka-configs.sh	Kafka 动态配置管理脚本	./kafka-configs.sh \ --zookeeper zookeeper_host:port \ --entity-type topics \ --entity-name topic_name \ --alter \ --add-config max.message.bytes=10485760
kafka-console-consumer.sh	kafka 消费者控制	./kafka-console-consumer.sh \ --bootstrap-server kafka-host:port \ --topic test \ --from-beginning
kafka-console-producer.sh	kafka 生产者控制	./kafka-console-producer.sh \ --bootstrap-server kafka-host:port \ --topic test
kafka-consumer-groups.sh	kafka 消费者组相关配置	重设消费者组位移: ./kafka-consumer-groups.sh \ --bootstrap-server kafka-host:port \ --group test-group \ --reset-offsets \ --all-topics \ --to-latest \ --execute
kafka-consumer-	kafka 消费者性能测试	./kafka-consumer-perf-test.sh \ --broker-list

perf-test.sh	脚本	kafka-host:port \ --messages 10000000 \ --topic test
kafka-delegation-tokens.sh	使用 Token 认证机制时对 Token 的操作	为用户生成 token: ./kafka-delegation-tokens.sh \ --create \ --bootstrap-server
kafka-delete-records.sh	用于删除 Kafka 的分区消息，由于 Kafka 有自己的自动消息删除策略，使用频率不高	删除指定的 partition 和 offset: ./kafka-delete-records.sh \ --bootstrap-server localhost:port \ --offset-json-file ./offset.json offset.json 的内容: {"partitions": [{"topic": "mytest", "partition": 0, "offset": 90}], "version": 1 }
kafka-dump-log.sh	查看 Kafka 消息文件的内容，包括消息的各种元数据信息、消息体数据	./kafka-dump-log.sh \ --files /var/local/kafka/data/test-topic-0/0000000100.log
kafka-log-dirs.sh	查询各个 Broker 上的各个日志路径的磁盘占用情况	./kafka-log-dirs.sh \ --bootstrap-server broker:port \ --describe \ --broker-list broker.id_list \ --topic-list topics
kafka-mirror-maker.sh	用于在 Kafka 集群间实现数据镜像	./kafka-mirror-maker.sh --consumer.config consumer.properties --producer.config producer.properties --whitelist "my-topic1,my-topic2" 以上 consumer.properties 配置文件: bootstrap.servers=broker_ip:port group.id=mirror_maker-group enable.auto.commit=true auto.offset.reset=earliest auto.commit.interval.ms=1000 以上 producer.properties 配置文件: bootstrap.servers=broker_ip:port acks=1 linger.ms=100 batch.size=16384 retries=3
kafka-preferred-replica-election.sh	触发 preferred replica 选举	对所有 topic 进行 preferred replica election: ./kafka-preferred-replica-election --zookeeper cdh-002/kafka
kafka-producer-perf-test.sh	生产者性能测试脚本	设置不同消息数进行压测: ./kafka-producer-perf-test.sh --topic test_perf --num-records 100000 --record-size 2000 --throughput 3000 --producer-props bootstrap.servers=IP:PORT ./kafka-producer-perf-test.sh --topic test_perf --num-records 1000000 --record-size C4000 --throughput 6000 --producer-props bootstrap.servers=IP:PORT ./kafka-

		producer-perf-test.sh --topic test_perf --num-records 10000000 --record-size 6000 --throughput 8000 --producer-props bootstrap.servers=IP:PORT
kafka-reassign-partitions.sh	用于执行分区副本迁移以及副本文件路径迁移	./kafka-reassign-partitions.sh --zookeeper hostname:port --topics-to-move-json-file topic.json --broker-list "0,1,2,4" --generate 以上 topic.json 文件: { "topics": [{"topic": "test"}], "version": 1 }
kafka-replica-verification.sh	复制进度验证脚本	./kafka-replica-verification.sh \ --broker-list IP1:PORT,IP2:PORT,IP3:PORT -topic-white-list test
kafka-run-class.sh	用于查看消费了多少条数据	./kafka-run-class.sh kafka.tools.ConsumerOffsetChecker \ --zookeeper IP1:PORT,IP2:PORT,IP3:PORT \ --group G1 --topic test
kafka-server-start.sh	启动 Kafka Broker 进程	./kafka-server-start.sh -daemon config/server.properties
kafka-server-stop.sh	停止 Kafka Broker 进程	./kafka-server-stop.sh
kafka-streams-application-reset.sh	用于给 Kafka Streams 应用程序重设位移, 以便重新消费数据	./kafka-streams-application-reset.sh \ --application-id
kafka-topics.sh	用于管理所有 topic, 比如创建一个 topic	./kafka-topics.sh \ --bootstrap-server broker_host:port \ --create \ --topic my_topic_name \ --partitions 1 --replication-factor 1
kafka-verifiable-consumer.sh	测试验证消费者功能	./kafka-verifiable-consumer.sh \ --broker-list IP:PORT \ --topic first \ --group-id group.demo \ --max-messages 2
kafka-verifiable-producer.sh	测试验证生产者功能	./kafka-verifiable-producer.sh \ --broker-list IP:PORT \ --topic first \ --message-create-time 1527351382000 \ --value-prefix 1 \ --repeating-keys 10 \ --max-messages 20
trogdor.sh	Kafka 的测试框架, 用于执行各种基准测试和负载测试	./trogdor.sh [action] [options]
zookeeper-security-migration.sh	更新 kafka 数据中的 zookeeper 的 ACL	./zookeeper-security-migration \ --zookeeper.acl=secure \ --zookeeper.connect={host}:{port}/{path}

zookeeper-server-start.sh	启动 zookeeper，Kafka 启动前需要先启动 zookeeper	./zookeeper-server-start.sh -daemon config/zookeeper.properties
zookeeper-server-stop.sh	停止 zookeeper，先停止 Kafka，再停止 zookeeper	./zookeeper-server-stop.sh
zookeeper-shell.sh	查看 kafka 在 zookeeper 中的配置	./zookeeper-shell.sh zookeeper_host:port[/path] [args...]

12 consumer 主要执行文件

文件所在路径:

/data/application/consumer/bin/

文件名	功能说明(常用使用场景)	使用方法
synctool	指定位点开启同步功能	<pre># ./synctool --help synctool: --help just a help info --zklist arg required, zookeeper ip list --zkrootpath arg required, zookeeper root path,for example: /tdsqlzk --groupid arg relevant group id, use to find instance --instanceid arg the product instance id --jobid arg the job id use to find right instance id --partitionnum arg src zookeeper root path,for example: /tdsqlzk --operation arg (=change_partition) operation type, now default change_partition, current support: change_partition, reset_product --resetproducttimestamp arg (=0) reset product timestamp, eg: 1654732800, tool will try find the gtid just before the timestamp --resetproductgtid arg reset product gtid, eg: GTID_FROM_BEGINNING, tool set the gtid for reader to restart --execute arg (=0) real do operation or only echo some</pre>

		<pre> prepare info --srczkiplist arg src set zookeeper ip list --srczkrootpath arg src zookeeper root path,for example: /tdsqlzk --kafkazkiplist arg kafka set zookeeper ip list --kafkazkrootpath arg (=kafka) kafka zookeeper root path,for example: /kafka --version MULTISRCSYNC-2.0.21- 6-R762D002 示例: #./synctool --zklist 9.30.1.233:2181,9.30.1.159:2181,9.30.1.246:2181 -- zkrootpath /tdsqlzk --jobid=0000028481 -- partitionnum=2 --execute=1 输出: op: change_partition log_path: ../log/synctool_change_partition_202206140 15342.log.2022-06-14.0 zookeeper timeout:10000 msec,msg timeout 30000 msec affect topic: alex-test-gs-7, current partition: 3, target partition: 1 affect job ids: setsyncjob@0000000038 will start operate operate success, see more log in ../log/synctool_change_partition_20220614015342.l og.2022-06-14.0 </pre>
binlogconsumer	binlog 日志消费和重放模块	<pre> ./binlogconsumermgn --zklist 172.27.32.13:2118,172.27.32.6:2118,172.27.32.10:211 8 --zkrootpath /tdsqlzk --kafkazklist 172.27.32.13:2118,172.27.32.6:2118,172.27.32.10:211 8 --kafkazkrootpath /kafka --dev eth0 </pre>
binlogconsumermgn	binlogconsumer 的管理模块	<pre> 启动: ./binlogconsumermgn --zklist IP:PORT -- zkrootpath /noshard1 \ --kafkazklist IP:PORT -- kafkazkrootpath /kafka --dev eth0 停止: ps -ef grep binlogconsumermgn grep -v 'grep' awk '{print \$2}' xargs kill -9 </pre>
binlogproducter	解析所在节点	<pre> ./binlogproducter ../conf/mysqlagent_4001.xml </pre>

	的 binlog 并 push 到消息队列	
binlogproducer_percona	binlogproducer 的 percona 版本	./binlogproducer_percona ../conf/mysqlagent_4001.xml
createZkFlag		
incrementalcheck	日志消费增量检查	./incrementalcheck --orgzkiplist IP:PORT \ --checklogpath ../data/jobid/checklog
install_dep.sh	安装 oracle 相关文件	./install_dep.sh
kafkamsgtool	Kafka 管理工具，可用于设置 batchsize 等参数	./kafkamsgtool --zklist IP:PORT --zkrootpath /noshard1 \ --kafkazklist IP:PORT --kafkazkrootpath /kafka --batchsize 1
kafkaUpdate		./kafkaUpdate zkip ziroot file_of_setname productor_switch
listUniqueId		./listUniqueId zkip zkroot setnames
multisrcsync_log_backup_clean.py	对于 binlogconsumer 产生的日志进行间歇性备份和清理	由 crontab 调用
mydumper_tdsq	tdsql for mysql 多线程数据 dump 工具	./mydumper_tdsq --lock-all-tables=FALSE --host=127.0.0.1 --port=3306 --user=app --password=123456 --trx-consistency-only=true --databases "test" > /data/mysqldump/test.sql
myloader_tdsq	tdsql for mysql 多线程数据加载工具	./myloader_tdsq -u root -p 123456 -B test -d /data/mysqldump/
mysql	mysql 客户端工具	mysql -u root -p 123456
mysqldump	mysql 单线程 dump 工具	./mydump --lock-tables=FALSE --protocol=tcp --set-gtid-purged=OFF --host=127.0.0.1 --port=3306 --user=app --password=123456 --single-transaction=TRUE --master-data=2 --databases "test" > /data/mysqldump/test.sql
newjoboncesync		./newjoboncesync --zkiplist IP:PORT --zkrootpath /noshard1 \ --mode execute --truncate 1
oncesynctable		
oracel.dep.tgz	oracle 部分组件	由 install_dep.sh 执行安装

	安装包	
runtime.err	记录运行出错的日志	tail -f runtime.err
safeshell	用于 Ruby 开发的安全 shell 环境	非执行文件
sqlldr	oracle 数据加载工具	sqlldr userid=scott/tiger@emp control=/data/data.ctl log=/data/log20210330.log bad=/data/bad20210330.bad /data/data.ctl 文件内容: options(SKIP=1,ROWS=1000,ERRORS=0) load data CHARACTERSET utf8 infile '/data/emp.csv' APPEND into table scott.emp fields terminated by ',' trailing nullcols (a, b, c, d)
sqlplus	oracle 客户端访问工具	sqlplus username/password@service_name

13 onlineddl 主要执行文件

文件所在路径:

/data/application/onlineddl/bin/

文件名	功能说明(常用使用场景)	使用方法
ddlperformer	online-ddl 执行进程	由 ddlperformermng 调度
ddlperformermng	online-ddl 管理进程	由 crontab 调度
plugin.pl	pt_online_schema_change 插件	无需单独使用
pt-online-schema-change	在线进行表结构变更	添加表字段 new_student: pt-online-schema-change \ --alter="add new_student varchar(10) NOT NULL DEFAULT " \ COMMENT '新学生';" \ --execute \ --print \ --max-lag=5 D=test,t=users,u=root,p=123,S=/tmp/mysql.sock \ --no-check-replication-filters \ --max-load="Threads_running=100" \ --critical-load="Threads_running=120" \ --charset=utf8 \ --chunk-size=100

14 filebeat 主要执行文件

文件所在路径：

/data/filebeat_helper/bin

文件名	功能说明
filebeat_zk_init_tool	filebeat 对 zk 的设置

filebeat_zk_init_tool

10.3.22.3.x-1 版本在输出类型为 kafka 时，支持按顺序上报 partition。因此提供批量刷新历史上报日志的节点。

```
#!/filebeat_zk_init_tool init -i 'tdsql_new_zk1:2181;tdsql_new_zk2:2181;tdsql_new_zk3:2181' -z '/tdsqlzk' -m 2 -r 1
```

说明:

实际使用时，需要修改上面参数为实际值：

- -i: zk 的 host 信息,多个使用;分隔。
- -z: zk 的根目录。
- -m: 2 表示修改 grouppriv、setconfig 节点内容。
- -r: 1 表示添加的 partition_rule 为 1，不填默认为 0。

使用范例

1. filebeat_zk_init_tool 工具提供刷新历史 zk 节点 grouppriv、setconfig 节点功能，具体使用如下。

```
./filebeat_zk_init_tool init -i 'tdsql_new_zk1:2181;tdsql_new_zk2:2181;tdsql_new_zk3:2181' -z '/tdsqlzk' -m 2 -r 1
```

2. 通过 filebeat_zk_init_tool 工具将 match_config.json 文件的内容写入到 zk 里面，写入的 zk 节点为：get /tdsqlzk/filebeat/matchcfg。

```
./filebeat_zk_init_tool init -i 'tdsql_test_zk1:2181;tdsql_test_zk2:2181;tdsql_test_zk3:2181' -g './conf' -f 'match_config.json' -z '/tdsqlzk' -m 0
```

通过 filebeat_zk_init_tool 解析 globalconfig_template.json 文件的命令不变。

```
./filebeat_zk_init_tool init -i 'tdsql_test_zk1:2181;tdsql_test_zk2:2181;tdsql_test_zk3:2181' -g './conf' -f 'globalconfig_template.json' -z '/tdsqlzk'
```

3. 如果要升级 globalconfig_template.json 配置，请使用下面的工具进行升级。

```
./filebeat_zk_init_tool init -i 'tdsql_test_zk1:2181;tdsql_test_zk2:2181;tdsql_test_zk3:2181' -g './conf' -f 'globalconfig_template.json' -z '/tdsqlzk'
```


15 网关参数说明

15.1 server

示例：

```
<server>
  <zookeeper quiet="1" iplist="tdsql_sh9u32w9w_zk1:2118" timeout="10000" rootdir="/tdsqlzk"
safeacl="0" />
  <basic updateinterval="1" setname="" setname_fromzk="192.168.225.44_15001"/>
  <meta_cluster iplist="mc-host-1:12381,mc-host-2:12381,mc-host-3:12381" timeout="1"/>
  <dcn open="0"/>
  <log>
    <log_info name="../log/route_" log_size="1000000000" log_level="2"/>
  </log>
  <router_update exception_restart_num="3" enable_proxy_monitor="1"/>
</server>
```

服务器信息配置参数为网关的路由模块配置参数。

表 15.1-1 服务器信息配置参数

参数名称	参数说明	配置方式	默认值
zookeeper quiet	ZK 是否进入安静模式打印不必要的信息。	取值：0、1 ● 0：打印信息 ● 1：不打印信息	1
iplist	ZK 的 IP 列表信息，即 IP 和端口信息。	取值：IP 地址和端口号	ZK 的地址列表
timeout	路由模块 router_update 和 ZK 的连接超时时间。	取值：0~100000 单位：ms	10000
rootdir	实例在 ZooKeeper 上的根目录。	实例在 zk 上的根目录。	/
basic updateinterval	路由配置更新的间隔时间。	取值：0~10 单位：s	1
setname	非分布式实例静态名称，single_backend flag 为 1 时生效。	输入实例名称。	/

setname_fromzk	非分布式实例动态名称，设置获取实例名的 ZK 的 IP 和端口。	<ul style="list-style-type: none"> 如果 setname 为空时，网关将对应的 setname 从 ZK 中动态获取实例名。 如果 setname、setname_fromzk 都为空，则获取本地 IP 端口对应节点组（Set）。 	192.168.225.44_15001
log_info name	路由模块 router_update 的日志配置信息路径和名称前缀。	日志根目录 + 日志文件前缀	../log/route_
log_size	单个路由模块 router_update 的日志文件的大小。	取值：0~(2 ²³)-1 单位：字节	1000000000
log_level	路由模块 router_update 的日志级别。	取值：0、1、2 <ul style="list-style-type: none"> 0：debug 级别，打印所有日志。 1：info 级别。 2：error 级别，不仅记录错误日志，还记录关键信息，以帮助定位问题。 	0
meta_cluster iplist	需要的 meta cluster 的 iplist	ip 和 port 信息： xxx.xxx.xxx.xxx:11113,xxx.xxx.xx x.xxx:123123	ip1ist:8080
meta cluster timeout	连接 meta cluster 超时时间	> 0	5

15.2 instance

示例：

```
<instance>
  <plugins name="proxy"/>
  <run event_threads="24" />
  <listen address=":::15001" back_log="65535" ipv6_flag="0" enable_both_ipv4_and_ipv6="1" />
</instance>
```

表 15.2-1 实例信息配置参数

参数名称	参数说明	配置方式	默认值
plugins name	网关名称前缀。	/	proxy

	注意：当前版本已取消该配置参数。		
run event_threads	网关工作线程的数量。 注意：重启网关生效。	取值：0~240	24
listen address	网关监听的端口。 注意：重启网关生效。	设置为需要监听的端口。	0.0.0.0:3336

15.3 log

示例：

```
<log>
  <dbfw name="../log/dbfw_" log_size="1000000000" log_level="1"/>
  <system name="../log/sys_" log_size="1000000000" log_level="2"/>
  <interface name="../log/interf_" log_size="1000000000" sql_size="2048" openflag="1"
event="1" timeflag="0" extra="0" status_info="1" />
  <sql name="../log/sql_" log_size="1000000000" openflag="1" status_info="1" />
  <slow_sql name="../log/slow_sql_" log_size="1000000000" openflag="1"
long_query_time="1000" report="1" report_retry_time="3" report_interval="5"/>
  <clean time="14" size="10" thresh="15" interval="3" speed="32" multiple="2" timeout="600"
/>
  <moniter bid="dtsql_gw" subsysid="10254" intfid="1" />
</log>
```

日志配置参数为网关的主进程（mysql-proxy）配置参数，包括系统日志、接口日志以及接口日志相关的监控接口参数信息。

表 15.3-1 网关信息配置参数

参数名称	参数说明	配置方式	默认值
system name	网关系统名称日志。	日志根目录+日志名前缀。	../log/sys_
log_size	单个日志文件的大小。	取值：0 ~ (2 ²³ -1) 单位：字节	1000000000
log_level	日志级别。	取值：0、1、2 ● 0：debug 级别，打印所有日志。 ● 1：info 级别。 ● 2：error 级别，不仅记录错误日志，还记录关键信息，以帮助定位问题。	2

interface name	接口名称日志，该日志级别可调。记录 proxy 接收到的 sql。	日志根目录+日志名前缀。	../log/interf_
openflag	打开标记。	取值：0、1 ● 0：关闭标记 ● 1：打开标记	1
timeflag	时间标记，记录 proxy 每个阶段的耗时，一般用于时耗分析。	取值：0、1 ● 0：关闭标记 ● 1：打开标记	0
sql name	SQL 语句名称日志，该日志级别可调。记录 proxy 发送到 db 的请求。	日志根目录+日志名前缀。	../log/sql_
slow_sql name	显示 SQL 慢查询语句名称。	日志根目录+日志名前缀。	../log/slow_sql_
long_query_time	慢查询时间。进行 SQL 语句查询时，超过设置时间仍未显示查询结果为慢查询，该时间为慢查询时间。	取值：0~1000 单位：ms ● 0：记录所有 SQL 查询日志。 ● 1~1000：记录超过该查询时间的 SQL 日志。	100
clean time	清除日志，当超过设置天数时，日志将被自动删除。	取值：0~140 单位：天	14
size	清除日志大小，当超过设置值时，日志将被自动删除。 注意：优先按照超过设置天数的时间顺序删除日志。	取值：0~10000 单位：G	10
moniter bid	监控模块 ID。 注意：当前版本已取消该配置参数。	取值：0	/
subsysid	网关子系统模块 ID。 注意：当前版本已取消该配置参数。	取值：0~102540	10254
intfid	外部模块 ID。 注意：当前版本已取消该配置参数。	取值：0、1	/
slow_sql report	控制是否上报慢查询	取值：0、1 ● 0，不上报 ● 1，上报	1

15.4 mode

示例：

```
<mode>
  <user name="tdsql"/>
  <ddl delay="100" ddl_job_switch="1" view_job_switch="0" ddl_support_rollback="1"
ddl_retry_repeat_number="60" ddl_repeat_interval_ms="1000" scan_ddl_interval_ms="1000"
ddl_task_over_s="60" ddl_heart_beat_s="3" ddl_client_timeout_s="172800"
ddl_precheck_switch="1" ddl_precheck_lock_wait_timeout="1"
meta_data_from_zookeeper_no_table_exists="1" ddl_support_broadcast="0"
ddl_task_clean_time="0" auto_add_check="0"/>
  <recycle_bin recycle_bin_switch="1"/>
  <single_backend flag="1" rw_split="1" delay_thresh="10" ip_passthrough="1"/>
  <groupshard strict="0" ping_delay_timeout="0" ping_delay_number="3"
ping_delay_interval="500" reject_risky_sql="0" reject_flush_table="0"
partition_key_part_unique_key="0"/>
  <other only_bin_cursor="0"/>
  <init cap_add="" cap_del=""/>
  <slave keep_time="0" close_connection="0" change_time="300" connect="0"/>
  <ssl open="0"/>
  <priv open="0"/>
  <create delay="3"/>
  <prepare per_root="1" cache_prepared_stmt_count="0" prepare_rand_set="1"
use_syntax_tree="0" print_binary_protocol_sql="0"/>
  <save_in_db open="1"/>
  <sequence mode="0" oldstyle="0" cache="0" max_step_num="0" returning_detect_mode="2"
batch_num="0" enable_simple_sql_sequence="0" />
  <pool flag="0" mode="0"/>
  <cache size="10"/>
  <subp limit="1"/>
  <result rand="1"/>
  <information_schema randomness="0"/>
  <error ignore_unexpected_error="0"/>
  <multi_query open="1"/>
  <sub_partition auto_create='1' truncate='0' enable_cache_list_keyvalue='1'/>
  <compatible update_limit="0" strict_consistent_collation="0"/>
  <qid_prefix enable="1"/>
  <epoll reduce_event="0" event_threshold="100000"/>
  <optimize distinct_to_group_by="0" force_index_optimize="1" dirent_send="0"
noshard_user_val_pushdown="0"/>
```

```

<global_index max_modified_rows="10000" enable_select="1" enable_optimistic_commit="1"
enable_optimistic_dml="1"/>
<rebalance lock_wait_timeout="15" write_db_retry_time="60" sync_task_delay_time="4"
sync_task_waiting_time="21600" sync_task_thread="10" is_drop_deltable="1"
wait_proxy_update_route_time="3" dumpdir="." client_timeout="10" switchsql_wait_time="180"/>
<jemalloc profiling="0"/>
</mode>

```

Gateway 模块配置参数，是网关的主进程（mysql-proxy）配置参数。

表 15.4-1 模式信息配置参数

参数名称	参数说明	配置方式	默认值
single_backend flag	gateway 的连接方式。	取值：0、1 <ul style="list-style-type: none"> 1：1 个 gateway 后面只有一个 set，即为非分布式实例。 0：1 个 gateway 可能对应多个 Set，即为分布式实例。 	0
rw_split	“非只读账户”的读写分离模式开关。	取值：0、1、2 <ul style="list-style-type: none"> 0：关闭读写分离模式。 1：开启读写分离，但需手动加入 slave 关键字。 2：网关自动把读请求发往备机。 	0
delay_thresh	备机延时阈值，超过设置值的备机将不进行读写分离。	取值：1~3600 <ul style="list-style-type: none"> 1~3600：备机延时超过该值时不进行读写分离。 	10
ip_passthrough	IP 透传功能开关。	取值：0、1 <ul style="list-style-type: none"> 0：关闭 IP 透传。 1：开启 IP 透传。 	1
groupshard strict	分布式实例支持严格模式，是否禁止 insert、update 对应的 SQL 语句同时对 2 个以上 Set 进行写操作。	取值：0、1 <ul style="list-style-type: none"> 0：执行写操作。 1：拒绝写操作。 	0
other only_bin_cursor	光标判断。	取值：0、1 <ul style="list-style-type: none"> 0：判断光标。 1：只进行二进制协议时判断光标。 	0
init cap_add	连接后端默认的 cap。	默认的 cap。	" cap_del

			="
slave keep_time	在指定时间内，使用同一台备机。	取值：0~3600 单位：s	0
change_time	超过指定时间，尝试访问其它备机。	取值：0~3600 单位：s	300
connect	客户端建立连接时，是否连接备机进行认证。	取值：0、1 ● 0：不连接备机。 ● 1：连接备机。	0
ssl open	客户端和 Proxy 的 SSL 加密开关。不限制协议版本，只允许 tls1.2 协议版本。	取值：0、1、2 ● 0：关闭 SSL 加密。 ● 1：开启 SSL 加密。 ● 2：开启 SSL 加密。	0
priv open	SQL 语句类型的权限操作。	取值：0、1 ● 0：关闭权限操作。 ● 1：开启权限操作。	0
create delay	创建表之后，返回用户请求时，设置需要等待 ZK 信息的同步时间。 注意：新版本已经弃用。	取值：0~3600 单位：s	3
prepare per_root	是否为每个 prepare 二进制请求分配独立的内存池。	取值：0、1 ● 0：同一 session 上的所有 prepare 请求共用内存池，session 关闭后释放内存。 ● 1：每个 prepare 请求使用独立的内存池，prepare 请求关闭后释放内存。	1
save_in_db open	是否禁止 ZK 信息存入后端 DB 中，同时禁止使用 Sequence 和 Step 功能。	取值：0、1 ● 0：关闭该功能。 ● 1：开启该功能。	0
sequence mode	是否对 sequence 的获取进行合并优化。	取值：0、1 ● 0：关闭优化。 ● 1：开启 S 优化。	0
pool flag	是否打开连接池标识。	取值：0、1 ● 0：关闭连接池标识。 ● 1：打开连接池标识。	0
cache size	每个会话（session）最多	取值：0~100	10

	能够缓存的内存。	单位：块	
subp limit	是否将分页结果放入二级分区表。	取值：0、1 ● 0：不放入。 ● 1：放入。	0
information_schema randomness	控制 information_schema 库时，设置请求发往的节点组（Set）。	取值：0、1、2 ● 0：发往第一个 Set。 ● 1：访问 information_schema.tables 时，发往第一个 Set；访问其它表，则发往随机 Set。 ● 2：随机选择一个 Set 进行访问。	1
Multi_query open	groupshard 下，是否支持应用一次发送多条 SQL。	取值：0、1 ● 0，不支持 ● 1、支持	0
sub_partition auto_create	二级分区，是否自动新建分区。	取值：0、1 ● 0，关闭 ● 1，开启	1
compatible update_limit	控制是否兼容老版本处理“UPDATE/DELETE ... limit ...”的行为（即直接下推）。	取值：0、1 ● 0，不兼容 ● 1，兼容	0
compatible strict_consistent_collation	当执行 order by 时，sets 使用的 collation 不一致时是否报错。	取值：0、1 ● 0，不报错 ● 1，报错	0
qid_prefix enable	控制是否在发往 db 的 sql 中添加/*proxy_id: qid*/的前缀。例如： /*9:37847041-1620703640-15*/select * from t1;	取值：0、1 ● 0，不添加前缀 ● 1，添加前缀	0
epoll reduce_event	控制是否开启 epoll 优化，降低 epoll_ctl 的调用次数。用于多 SET、高并发场景下。	取值：0、1 ● 0，不开启优化 ● 1，开启优化	0
result rand	没有指定 order by 的 sql，强行加上一个随机排序。	取值：0、1 ● 0：不排序 ● 1：排序	1

15.5 security

示例：

```
<security>
  <lock lock_min="0" lock_num="5" />
  <offline offline_min="0" />
  <conn conn_limit="0" account_auth_last_set="0" net_buffer_size="4096"
simple_sql_first_set="0" />
  <shardkey update="0" type="0"/>
  <server user="IGS3xjSFsycRBBUGBPfT" pwd="M2bEJ2SnJzUkV2GIIXQx1MUIYTY="/>
  <table max="5000"/>
  <name restrict="1"/>
  <server_close timeout="0" explicit_broken="0" />
  <partial select="0"/>
  <unsafe_sysfunc openflag="1" blacklist="" sysdate_is_now="0" />
  <reject nosk="0" nopk="1" partfail="0" nowhere="1" global_table_partdiff="0" />
  <timeout connect="10"/>
  <keepalive idle="5" intvl="2" cnt="5" tcp_user_timeout="0"/>
  <update quick="0" stop_time="0"/>
  <trace host="0" />
</security>
```

Gateway 模块配置参数，是网关的主进程（mysql-proxy）配置参数。

表 15.5-1 安全配置信息参数

参数名称	说明	配置方式	默认值
lock lock_min	noshard 下，用户输入密码错误，当超过认证次数时，设置被锁定的等待时间。	取值：0、1~60 单位：min ● 0：不锁定 ● 1~60：锁定时间 1~60	0
lock_num	noshard 下，用户密码输入错误的认证次数。	取值：0，1~50 ● 0：不限定认证次数。 ● 1~50：认证错误次数 1~50。	5
offline offline_min	用户连接网关超过设置时间时，则断开网关与用户的连接。	单位：min 取值：0、1~60 ● 0：不限定连接时	0

		<p>间。</p> <ul style="list-style-type: none"> ● 1~60: 用户连接网关超时 1~60, 则断开网关与用户的连接。 	
conn conn_limit	配置每个 IP 最多的连接数。	<p>取值: 0 ~ 100000</p> <ul style="list-style-type: none"> ● 0: 每个 IP 无连接数限制。 ● 1~10000: 每个 IP 最多的连接数。 	0
server user	连接后端默认的内部用户名。	内部加密算法进行加密, 不能改动。	IGS3xjSFsycRB BUGBPfT
pwd	连接后端默认密码。	内部加密算法进行加密, 不能改动。	M2bEJ2SnJzUkV 2G IIXQx1MUIYTY =
timeout connect	Proxy 和前段、后端的连接超时时间。	<p>取值: 0~ 3600</p> <p>单位: s</p>	300
server_close timeout	如果 Proxy 与后端断连, 默认不关闭前端连接, 在事务中如果与前端连接超过这个时间则一直报错, 否则断连。	<p>取值: 0~600</p> <p>单位: s</p>	60
reject nosk	用户建表时, 没有指定 Shardkey 字段, 是否执行报错。	<p>取值: 0、1</p> <ul style="list-style-type: none"> ● 0: 不报错 ● 1: 报错 	0
nopk	用户建表时, 没有指定主键, 是否执行报错。	<p>取值: 0、1</p> <ul style="list-style-type: none"> ● 0: 不报错 ● 1: 报错 	1
table max	分布式实例中最大表的数量限制。	取值: 1000~10000	1000
partial select	在分布式实例中, 当某个 Set 出现问题, 是否返回部分成功的数据。	<p>取值: 0 或 1</p> <ul style="list-style-type: none"> ● 1: 返回。 ● 0: 不返回。 	0
keepalive idle	控制套接字的 TCP_KEEPIIDLE 属性, 即超过设置时间没有发送数据时, 开始发送 Keep-Alive 包。	<p>取值: 0~ 3600</p> <p>单位: s</p>	10
intvl	控制套接字的 TCP_KEEPIINTVL 属性, 即设置等待对方 keep	<p>取值: 0~ 3600</p> <p>单位: s</p>	10

	alive probe 响应的超时时间，超时后重新发送 probe。		
cnt	控制套接字的 TCP_KEEPCNT 属性，即连接 keep alive probe 失效的重试次数。	取值：0~90 ● 0：不设置重试次数。 ● 1~90：重试次数为 1~90。	9
update quick	设置旧 Proxy 网络断开情形。该参数动态生效，无需重启，在 SQL 语句执行完成后执行 quick 检测。	取值：0、1 ● 0：当 stop_time=0 时，旧 proxy 在只有客户端全部断连时退出。 ● 1：某个连接一旦结束事务，旧 Proxy 会自动断开连接。	0
stop_time	设置旧 Proxy 存活时间，超过后不管是否还有连接，直接退出。该参数动态生效，无需重启，从升级开始算计算时间，而不是修改配置文件。	取值：0~3600 单位：s ● 0：当 update quick=0 时，旧 Proxy 在只有客户端全部断连时退出。 ● 1~3600：旧 Proxy 存活最长时间 1~3600。	0
shardkey update	设置带 shardkey 字段的 SQL 语句更新时，网关是否执行该语句。	取值：0、1 ● 0：拒绝更新该 SQL 语句。 ● 1：执行该 SQL 语句更新，让后端数据库判断能否更新。	0
shardkey type	是否限制分区键的类型必须为整数类型或者字符串类型	取值：0、1 ● 0：不限制 ● 1：限制	0

15.6 statistics

示例：

```
<statistics>
  <basic report_memory="1" update_interval="30" update_mem_interval="5"
ip="192.168.225.44"/>
  <time_cost divide_keys="5 20 30" dcn_divide_keys="25 40 50" stat_percentile="1"/>
  <error_code system="1290,1526"/>
</statistics>
```

Gateway 模块配置参数，是网关的主进程（mysql-proxy）配置参数。

表 15.6-1 状态信配置信息参数

参数名称	说明	配置方式	默认值
basic update_interval	网关向 ZK 上报监控信息的周期。	取值：0~600	60
time_cost divide_keys	SQL 延迟统计图的延迟分割点列表，例如，默认值(5 20 30)，则分别统计 0~5ms 的 sql 数量、5~20 的 sql 数量、20~30 的 sql 数量、>30 的 sql 数量。	数值不能修改。	5 20 30
time_cost dcn_divide_keys	同 time_cost divide_keys，但只对跨城访问的 SQL 语句进行统计。	数值不能修改。	25 40 50
error_code system	表示后端出现异常，需要上报错误的错误码。 注意：建议采用默认配置，不要改动。	取值：1290,1526 ● 1290：为后端只读错误，单独上报。 ● 1526：为分区异常导致的后端系统错误。如要添加其它错误码，会和 1526 联合上报，不单独区分。	1290,1526
time_cost stat_percentile	控制是否上报 SQL 的平均耗时，p95/p99 耗时等基于直方图的耗时统计信息	取值：0、1 ● 0，关闭上报 ● 1，开启上报	1

15.7 flowcontrol

示例：

```
<flowcontrol>
  <basic fc_interval="0" avr_num="5"/>
  <threshold cpu="26" io="31"/>
  <time min="1" max="10" inc="1" dec="0.5"/>
  <percent inc="10" dec="5"/>
</flowcontrol>
```

Gateway 模块配置参数，是网关的主进程（mysql-proxy）配置参数。

表 15.7-1 流量控制配置参数

参数名称	说明	配置方式	默认值
basic fc_interval	流控每次取值间隔时间。	取值：0~3600 单位 s	0
avr_num	参考的取值个数。	取值：0~50	5
threshold cpu	CPU 阈值，超过该值就开启流控。	取值：0~900	90
io	IO 阈值。	取值：0~600	60
time min	流控通过增加 sleep 实现最小 sleep 时间。	取值：0~3600 单位 ms	1
max	流控最大 sleep 时间。	取值：0~3600 单位 ms	10
inc	流控每次增加的时间。	取值：0~3600 单位 s	1
dec	流控每次减少的时间。	取值：0~3600 单位 s	0.5
percent inc	流控每次增加的百分比。	取值：0~100 单位：%	10
dec	流控每次减少的百分比。	取值：0~100 单位：%	5

15.8 xa

示例：

```
<xa>
  <basic xa_deadlock_timeout='2' xa_deadlock_detect_interval="10"
xa_deadlock_detect_sql_info="1" seqno_reserve_interval="1024" enable_xa="1"
proxy_server_id="0" max_gwid="3000" seqno_file_path="../data/seqno" log_gtid_timeout="60"
enable_consistent_read="0" consistent_read_optimize="1" enable_snapshot_optimize="2"
enable_global_trx_order="0" request_gts_timeout="5" request_gts_expire_time="60000"
request_gts_reuse_time="0" request_gts_fuzzy_time="100" snapshot_read_timeout="1"
xa_explore_interval="300" />
</xa>
```

Gateway 模块配置参数，是网关的主进程（mysql-proxy）配置参数。

表 15.8-1 xa 模式信息配置参数

参数名称	说明	配置方式	默认值
basic xa_deadlock_time out	在 XA 模式下，设置计时器的超时时间。 网关向后端发送 SQL 语句后，会启动该计时器，如果超时，将触发后台线程死锁检测。	取值：0~3600 单位：s	2
seqno_reserve_inte rval	在分布式实例下，自动为每个事务分配 ID 值，该值规则为按序号递增，同时将该 ID 最大值保存在文件中，以避免重启时与之前的 ID 值冲突。为了性能考虑，每次分配时将该文件中的最大值递增。	取值：0~10240	1024
enable_xa	在分布式实例中，是否开启 XA 模式。	取值：0、1、2 ● 0：关闭 XA 模式。 ● 1：开启 XA 模式。 ● 2：退化为伪分布式事务。	0
proxy_server_id	网关的 ID 值。	模板配置为 0。旧版本需要 changerootdir 自动修改数值；新版本网关内部自动产生全局唯一 ID，不再	0

seqno_file_path	保存事务最大 ID 值的文件路径。	需要通过配置文件指定。 输入最大 ID 值文件路径	../data/seqno
log_gtid_timeout	XA 模式下，将事务 id 插入到 xa.gtid_log_t 表的插入超时时间。 网关提交事务时会将事务 id 插入到 xa.gtid_log_t 表，如果插入超时，则插入操作会被后端拒绝，事务将会被自动回滚。	取值：0~3600 单位：s	3
enable_consistent_read	一致行读的开关	取值：0、1 ● 0：关闭 ● 1：开启 注意：可以通过 OSS 接口配置	0
max_gwid	网关注册的唯一 ID 的最大值	3000	/
consistent_read_optimize	一致性读的优化，针对修改单个 set 的事务（update，insert，delete 更新操作），去掉 xa 框架，在 enable_consistent_read=1 有效	取值：0、1 ● 0：关闭。 ● 1：打开。	1

15.9 join

示例：

```
<join>
  <basic enable_trace='0' max_pushdown_index_values="16" async_join_trigger='10000'
async_join_threads='24' enable_fast_filling="1" enable_metacache="1" optimize_simplejoin='1'
simplejoin_max_varchar_size="256" optimize_simpleunion="1"
optimize_simple_derived_table="1" optimize_simple_sub_partition="0" enable_column_cutoff="0"
derived_merge="1" enable_materialize_subquery="0" enable_subpartition_join_optimize="0"
pushdown_limit_to_simple_derivedtable="1" enable_optimize_using_global_index="1"
ignore_unused_dynstat="0" writeio_limit="0" merge_limit_sql_number="4000"
max_disk_size="1024" max_table_size="1024" enable_optimize_curdate="1"
curdate_disable_interval="60" enable_optimize_date_func="1" enable_lock='0'
optimize_sub_partition_count='0' optimize_sub_partition_cond='0'
optimize_pushdown_subpartition_orderby_limit="0" optimize_simple_join_equal_cond="0"
optimize_range_one_set="0" db_sync_embedded_val="" />
</join>
```

join 为 Gateway 模块配置参数，是网关的主进程（mysql-proxy）配置参数。

表 15.9-1 join 信息配置参数

参数名称	说明	配置方式	默认值
basic enable_trace	是否在 sys 日志中记录 join 的 trace 信息。	取值：0、1 ● 0：不记录。 ● 1：记录。	0
async_join_trigger	当参与连接的表的行数和列数的乘积超过该值时，使用后台线程执行查询。	取值：0~100000	10000
enable_fast_filling	调用 myisam 的快速接口将数据写入 myisam 表	取值：0、1 ● 0：不调用。 ● 1：调用。	1
async_join_threads	执行 join 以及写临时表的后台线程数。	取值：0~240	24
enable_metacache	网关是否缓存后端表的结构。	取值：0、1 ● 0：不缓存。 ● 1：缓存。	1
optimize_simplejoin	是否对简单查询进行直接下推	取值：0、1	1

	优化（不写临时表）。	<ul style="list-style-type: none"> ● 0：不下推优化。 ● 1：下推优化。 	
optimize_simpleunion	是否对简单的 union 进行下推优化（不写临时表）。	取值：0、1 <ul style="list-style-type: none"> ● 0：不下推优化。 ● 1：下推优化。 	1
enable_column_cutoff	网关加载数据到临时表时，是否只加载参与查询的列。	取值：0、1 <ul style="list-style-type: none"> ● 0：不是。 ● 1：是。 	1
derived_merge	是否对 derived table 进行上提，例如，select * from (select * from t1) x 转换为 select * from t1;	取值：0、1 <ul style="list-style-type: none"> ● 0：不是。 ● 1：是。 	1
writeio_limit	对加载数据时，写临时表进行 IO 限速（每秒写入的字节数）	取值：0、1~1000000000 单位：B/s <ul style="list-style-type: none"> ● 0：不限速。 ● 1~1000000000：限速设为 1~1000000000。 	0
enable_cs_introducer	是否在拉取数据的过程中，为字符串加上字符集标记。	取值：0、1 <ul style="list-style-type: none"> ● 0：不加字符集标记。 ● 1：加字符集标记。 	0
max_disk_size	限制网关嵌入式数据库占用的最大存储空间。	取值：0~（263-1） 单位：G <ul style="list-style-type: none"> ● 0：不限制。 ● 1~（263-1）：总的临时表的最大存储空间。 	1024
max_table_size	限制网关嵌入式数据库单个表占用的最大存储空间。	取值：0~（263-1） 单位：G <ul style="list-style-type: none"> ● 0：不限制。 ● 1~（263-1）：单个临时表的最大存储空间为 1~（263-1）。 	1024
ignore_unused_dynstat	执行复杂查询时，是否统计不使用的列。	取值：0、1 <ul style="list-style-type: none"> ● 0：统计。 ● 1：不统计。 	0

16 usercheck 参数说明

示例：

连接进入 zookeeper：

```
cd /data/application/zookeeper/bin/
```

```
./zkCli.sh -server host:port
```

```
get
```

```
/tdsqlzk/group_1722482858_16/userchecks/usercheck@group_1722482858_16/user@user1_w_*  
{"delay_thresh":"0","expire":"0","host":"","master_rw_split":"0","password":"*F826477DCED1CA  
F96CAE99DF0B1DE849BC3C7B63","readonly":"0","same_idc":"0","slave_const":"0","slave_rw_s  
plit":"0","trace_id":"c0a87a011722483138574174207185","user":"hexin_w","watch":"0"}
```

说明：

user1_w：分布式实例 group_1722482858_16 创建的数据库用户。

表 16-1 usercheck 参数说明

主体数据	对象	是否必须	描述
delay_thresh	数字	否	选择主备延迟小于该值的备机。 不填写时使用配置文件中配置的值。
expire	数字	是	过期时间。 取值为 0，表示是永远不超时。
host	字符串	是	用户的 ip，也支持 ipv6。 %代表任意地址后缀匹配。
master_rw_split	数字	否	readonly=4 时，主 DCN 读写分离。
password	字符串	是	用户密码。
readonly	数字	否	<ul style="list-style-type: none">0：普通用户。1：优先选择备机进行读操作，如果备机的延迟都大于设置的延迟，则从主机读取。2：只选择备机进行读操作，如果所有的备机都大于设置的延迟，则直接报错。3：只选择备机进行读取，忽略延迟参数，一般用于拉取 binlog 同步。默认是 0。

same_idc	数字	否	<ul style="list-style-type: none">● 4: DCN 读写分离账号。● 0: 不区分备机, 默认是 0。● 1: 优先选择跟主节点同一个 idc 的备机。● 2: 只选择与主机在同一个 idc 的备机。
slave_const	数字	否	这个参数在 readonly 为 1 或者 2 的时候生效。目前 proxy 在选择备机的时候会在主备延迟满足条件的情况下自动选择不同的备机, 如果 slave_const 为 1, 则当前备机不满足条件的话, 断开连接。默认值为 0, 和之前逻辑一致, 会在多个备机之间选择。
slave_rw_split	数字	否	readonly=4 时, 备 DCN 读写分离。
user	字符串	是	用户名字, 不能添加 tdsq1 用户以及 tdsq1sys_*开头的用户。
watch	数字	否	<ul style="list-style-type: none">● 0: 选择正常的备机进行读操作, 默认是 0。● 1: 选择 watch 状态的节点进行读操作。