





专栏 / 管理与运维

-  1
-  0
-  0
-  0

TiDB与MySQL在备份容灾体系的衡量对比

👤 大数据模型 发表于 2024-04-16

🏷️ 原创 # 管理与运维

前文

如果把数据库的工作建设分为三大体系，从前中后的角度来看，笼统一点可以分为开发设计体系【前】、监控优化体系【中】、备份恢复体系【后】，类似规划设计、运营管理、安全保障的关系，构成数据库整个生命周期的管理。

三大体系中可以看出一个产品的覆盖度广窄，三大体系可以洞悉一个产品的成熟度深浅。三大体系简单简述如下。

- 开发设计体系考虑的更多是产品技术造型、基准测试报告、最佳施工应用实践、开发者使用标准规范等设计因素。让产品对准业务有的放矢。
- 监控优化体系考虑更多是流量、延迟、饱和度、关键指标、查询优化等运维因素，让产品健康正常的在生活环境中运行。
- 备份容灾体系考虑的高可用、业务连续性、不可抵抗力的灾后建设。必要通过额外的技术手段，保障产品的售后安全

对业务和DBA而言，开发设计体系和监控优化体系是DBA经常要做的事情，而备份容灾体系则是一劳永逸的工作，初次进行全面的数据备份策略要花较大的精力。

MySQL的备份容灾体系成熟度已经很高，针对MySQL的备份容灾和TiDB的备份容灾做了技术对比。

MySQL备份体系

MySQL的备份技术栈有mysqldumper、mydumper等逻辑备份工具，也有xtabackup等物理备份工具。

逻辑备份的定义，从数据库的接口层出发，数据库客户端访问数据库服务端，像其它SQL语句一样进行正常的解析，最后输出结果为SQL语句或者CSV文件。

物理备份区别于逻辑文件，直接针对硬盘存在的持久化的物理文件进行备份，因为不需要SQL解析，所以处理速度会很快。

mysqldumper是MySQL典型的逻辑备份工具，可以指定某个库、某个表、甚至全库进行备份，当进行备份操作的时候，打开Gernal日志，mysqldumper会启用全局锁，flush table with read lock 会阻止其它表和数据进行DDL和DML操作，必须备份完成才会解锁。

mysqldumper这样的目的是为了备份一致性。数据库工作生产时，一边在大量写入数据，一边需要对数据进行备份，备份的数据要接近生产数据的真实状态，这个就是备份一致性的作用。MySQL为了做到这一点，提供全局锁flush table with read lock

mydumper相对于mysqldumper，它的备份速度要快的多，因为mysqldumper是单线程，而mydumper是多线程备份，mydumper在满足mysqldumper的基本功能上，还提供对目标文件进行压缩编码。

mydumper同样存在不足，它比mysqldumper需要占用更多的资源，而且数据进行恢复的时候，同样是在MySQL的逻辑层进行SQL解析，这个需要花费大量的时间。进行数据恢复的时候mydumper比mysqldumper占不了多少优势。

xtabackup是针对innodb引擎的备份工具，目前有2.4和8.0版本，其中xtabackup2.4是针对MySQL5.6和MySQL5.7，而xtabackup8.0针对的是MySQL8.0。

xtabackup的备份技术原理分为三步走。

- 第一步，物理备份开始，针对MySQL最近一次checkpoint点进行redo日志的日志备份。
- 第二步，针对IBD文件【公共表空间文件和独立表空间、undo文件】、非IBD文件【上全局锁】、进行备份。
- 第三步，当最后一个非IBD文件传输完成，代表redo日志拷贝结束，整个物理备份结束。

xtabackup的每一步都为了备份一致性

- 第一步的作用 识别并获取目标数据源，对那些刚刚到硬盘上，没有来得及进入innodb引擎的数据进行回放，对一些表空间的脏页尚未递交成功，马上放弃通过redo日志进行回放。
- 第二步的作用 拷贝物理文件，针对非IBD文件进行全局锁，防止这个时候有人对非IBD进行DML和DDL操作。在8.0.27版本，只允许进行DDL操作，不允许进行DML操作，进一步细化备份一致性的粒度。
- 第三步的作用 非IBD文件传输完成结束，自行进行解锁，同行监控REDO的线程也结束，备份到了最后阶段。

关于增量备份，基于初次全量备份后，xtabackup通过识别REDO日志的LSN是否发现变化，LSN的序列号是往前递进的，是否变化一目了然，增量备份只叠加备份后的数据。

关于数据恢复，可以基于全量备份恢复或者增量恢复，或者基于全量备份恢复+ 日志备份恢复的方式。

物理备份上，xtabackup使用了全局锁技术，REDO日志监控技术，非IBD文件传输等同于REDO监控结束等方法保障了全量备份、增量备份，实现了基于时间点的数据恢复，恢复到指定的数据恢复点上。

TiDB备份体系

前文

MySQL备份体系

TiDB备份体系

备份容灾总结

TiDB支持mysqldumper和mydumper工具， 但是只能逻辑层面对已经持久化的数据进行备份，无法进行一致性备份。

mysqldump使用 --single-transaction 进行一致性备份

```
root@henley-Inspiron-7447:/tmp# mysqldump -h192.168.10.14 -u root -pGmcc@1234 -P4000 --single-transaction
mysqldump: [Warning] Using a password on the command line interface can be insecure.
mysqldump: Couldn't execute 'ROLLBACK TO SAVEPOINT sp': SAVEPOINT sp does not exist (1305)
```

mydumper使用备份的时候，提示更是明确

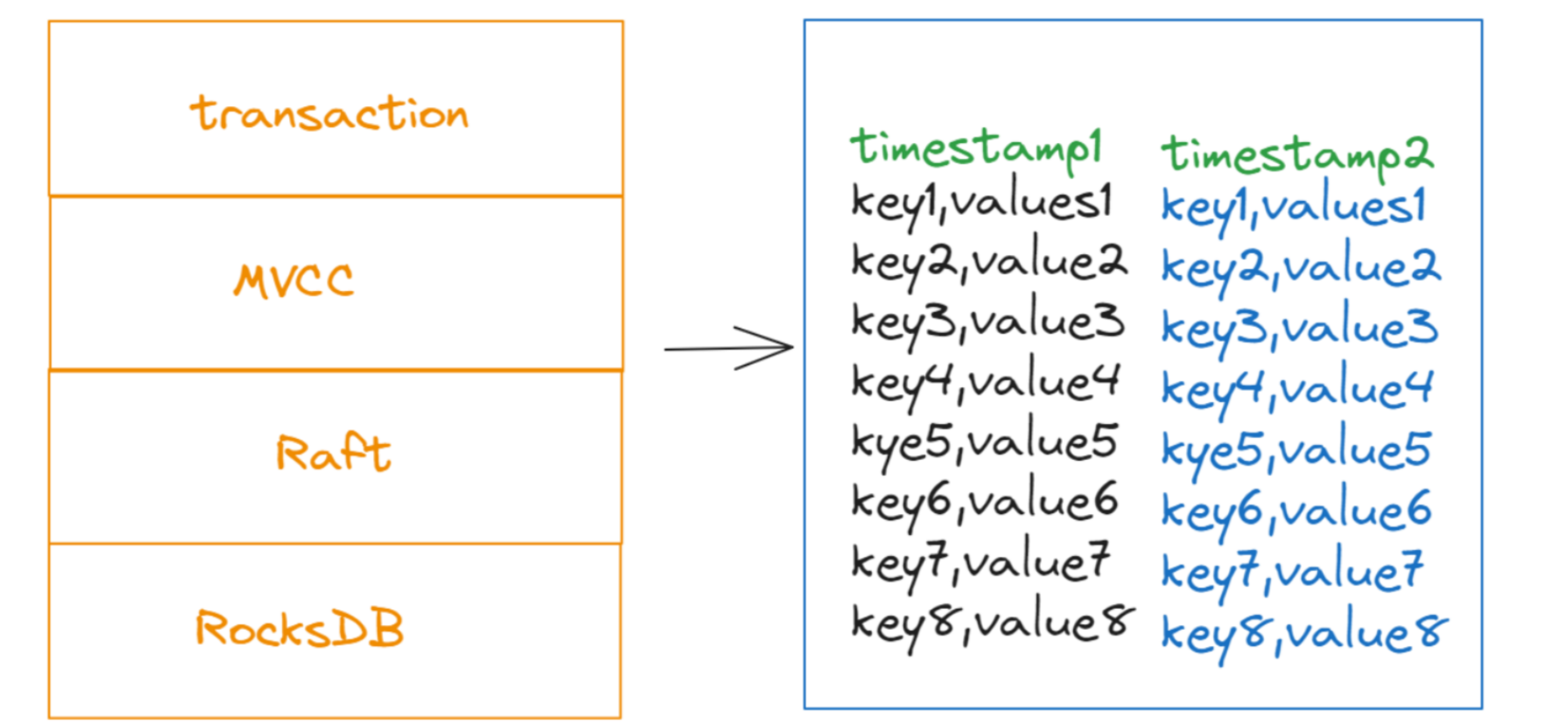
```
** (mydumper:6185): CRITICAL **: 16:34:20.943: Couldn't acquire global lock, snapshots will not be consistent
```

究底根因，目前TiDB没有支持全局锁,只支持tidb_snapshot，那么tidb_snapshot是什么？

```
tidb> FLUSH TABLES WITH READ LOCK;
ERROR 1105 (HY000): FLUSH TABLES WITH READ LOCK is not supported. Please use @@tidb_snapshot
```

这个要说到TiDB的内存管理机制，对于写进来的数据，TiDB把增量数据放到内存，提交成功的数据放在基线数据里面，内存的数据待到适当时机，会冻结合并到基线数据里面。

基线数据是持久化刷新到硬盘里面的数据，每一行数据都会对应一个唯一的时间戳



TiDB的物理备份工具BR的技术原理，就是把已经提交后的物理文件进行时间戳的扫描，把对应的时间点的数据文件信息都集中在一起，分为三步走。

- 第一步扫描KEY VALUE，从TIKV所在的Region读取备份时间点对应的数据。
- 第二步生成SST，将数据保存到SST文件中，这些文件存储在内存中。
- 第三步上传SST，将SST文件上传到存储路径。

下面做个小测试，三个时间点 2024-04-15 10:18:32 , 2024-04-15 10:18:35 , 2024-04-15 10:22:32 ,其中第一个时间连续 备份三次， 期间数据一直在不停的增加中。

```
tiup br backup full --pd "192.168.153.128:2379" --backupts '2024-04-15 10:18:32' --storage "/tidb/backup"
tiup br backup full --pd "192.168.153.128:2379" --backupts '2024-04-15 10:18:32' --storage "/tidb/backup"
tiup br backup full --pd "192.168.153.128:2379" --backupts '2024-04-15 10:18:32' --storage "/tidb/backup"
tiup br backup full --pd "192.168.153.128:2379" --backupts '2024-04-15 10:18:35' --storage "/tidb/backup"
tiup br backup full --pd "192.168.153.128:2379" --backupts '2024-04-15 10:22:32' --storage "/tidb/backup"
```

在TiDB的不断的增加数据的过程中，第1个时间点三次备份都是122M，第2个时间点是比第1个时间点多3秒，备份后数据是138M，第3个时间点备份是174M，时间越往前，识别的数据就越多。

```
[tidb@server128 backup]$ du -sm ./*
122      ./1      ➡ 2024-04-15 10:18:32
122      ./2      ➡ 2024-04-15 10:18:35
122      ./3      ➡ 2024-04-15 10:18:35
138      ./4      ➡ 2024-04-15 10:22:32
174      ./5
```

而且TiDB的物理备份的过程中，只从后端爬物理文件，不需要与其它模块协同。而MySQL会加上全局锁，防止前端写入数据。两者比较相对而言，MySQL的备份机制显得谨慎一点，延迟一点，多些安全。

快照的好处在于记住某个时间点的数据总集，可以进行某个时间点的数据恢复，例如误操作把表truncate了，由于表以前做过快照，我依然可以通过相同的时间点把数据物理备出来。


```
// 对大表进行truncate
mysql> truncate table 表名3;
Query OK, 0 rows affected (55.24 sec)
mysql>
mysql> truncate table 表名2;
Query OK, 0 rows affected (0.11 sec)
mysql> truncate table 表名1;
Query OK, 0 rows affected (0.13 sec)
```



```
// truncate后已经是28分的事，按照 11:26:30去备份依然有数据，备分到2文件夹上面
root@server128 tidb)# tiup br backup full --pd "192.168.153.128:2379" --backupts '2024-04-15 11:26:30'
Starting component br: /root/.tiup/components/br/v8.0.0/br backup full --pd 192.168.153.128:2379 --backupts
Detail BR log in /tmp/br.log.2024-04-15T11.28.38+0800
Full Backup <-----
Checksum <-----
[2024/04/15 11:28:48.091 +08:00] [INFO] [collector.go:77] ["Full Backup success summary"] [total-ranges=47]
```

```
[tidb@server128 backup]$ du -sm ./*
108  ./1
108  ./2 ➡truncate后的表，按照时间点去备份，依然有数据
```

除了内置的快照，TiDB也支持外置的快照，通过LVM也可以对数据进行备份。

如果使用LVM进行备份，注意安装的时候，需要把TiDB的数据盘安装在LVM的 逻辑卷上面，以下TiDB的数据盘安装在 /dev/mapper/centos_server153-tidb--data 下面。

```
[tidb@server128 backup]$ df -h
Filesystem                                Size  Used Avail Use% Mounted on
/dev/mapper/centos_server153-root          173G  104G   63G   63% /
devtmpfs                                  6.3G     0   6.3G    0% /dev
tmpfs                                       6.3G     0   6.3G    0% /dev/shm
tmpfs                                       6.3G   75M   6.2G    2% /run
tmpfs                                       6.3G     0   6.3G    0% /sys/fs/cgroup
/dev/sda1                                  269M  117M  135M   47% /boot
/dev/mapper/centos_server153-tidb--deploy  4.7G  1.3G  3.2G   29% /tidb/tidb-deploy
/dev/mapper/centos_server153-tidb--data    20G   13G   5.2G   72% /tidb/tidb-data
tmpfs                                       1.3G     0   1.3G    0% /run/user/0
```

```
// 直接在外围进数据快照
lvcreate -L 5G -s -n lv-mysql-snap01 /dev/centos_server153/tidb-data
lvcreate -L 5G -s -n lv-mysql-snap02 /dev/centos_server153/tidb-data
```

```
[root@server128 ~]# lvscan
ACTIVE                '/dev/centos_server153/swap' [4.00 GiB] inherit
ACTIVE                '/dev/centos_server153/root' [<175.71 GiB] inherit
ACTIVE                '/dev/centos_server153/tidb-deploy' [4.88 GiB] inherit
ACTIVE  Original      '/dev/centos_server153/tidb-data' [19.53 GiB] inherit
ACTIVE  Snapshot      '/dev/centos_server153/lv-mysql-snap01' [5.00 GiB] inherit
ACTIVE  Snapshot      '/dev/centos_server153/lv-mysql-snap02' [5.00 GiB] inherit
```

//删除数据

```
rm /tidb/tidb-data/* -rf
umount /tidb/tidb-data/
```

//恢复数据

```
[root@server128 ~]# lvconvert --merge /dev/centos_server153/lv-mysql-snap02
Merging of volume centos_server153/lv-mysql-snap02 started.
centos_server153/tidb-data: Merged: 80.49%
centos_server153/tidb-data: Merged: 100.00%
```

//重新上线

```
mount /dev/mapper/centos_server153-tidb--data /tidb/tidb-data/
```

备份容灾总结

针对数据库的备份恢复存在多种方法，例如针对数据库的接入层备份恢复、针对数据库的服务层进行备份恢复、针对数据库的物理进行备份恢复。

以MySQL为例，支持接入层、服务层、物理层的备份恢复，重视锁处理生产数据的矛盾冲突。

接入层的备份有mysqldumper和mydumper，接入层在数据恢复的时候，需要进行昂贵的SQL解析，花费大量的时间，虽然mydumper备份快，但是数据导入速度也不快。

服务层的备份导出例如CSV的规范式文件，再通过load data的方式进行导入，如果接入层的恢复是直路导入，那么服务层是旁路导入，旁路导入按照规律快速生成有序组织，避免SQL的消耗。

物理层的备份有Xtrabackup，它可以把把原有的MySQL相关物理文件迁移过来，技术层面重视已经持久化的数据和没有持久化的数据，通过redo回放数据，释放表空间尚未提交的脏页。经过清洗的数据文件可以马上拿来就用，不需要额外转换。

快照备份，MySQL支持LVM备份

TiDB同样支持接入层、服务层、物理层的备份恢复，主要通过快照处理生产数据的矛盾冲突。

接入层的备份有mysqldumper和mydumper,经过TiDB的适配二开，由于TiDB 分布式能力，资源能力潜力较大，间接也提升mysqldumper和mydumper的使用率

服务层的备份有Lightning导入和Dumpling导出，Lightning导入绕过接入层，直接在服务层导入，通过转化成KEV_VALUE形式，性能较快。

物理层的备份有BR工具，直接对TiKV 进行时间戳扫描，扫描后的文件转换成sst文件，如果要恢复数据文件，需要专业工具进行转换。

快照备份，TiDB支持LVM备份

在备份上的策略，MySQL积极与生产数据协同，达成一致，沟通的方式先上一个全局刷，让它们稍停一会儿。MySQL甚至有元数据锁，解决DDL层面数据结构稳定不变的锁机制。而备份上TiDB的落地则是使用快照确定米已成炊的事实，在锁方面的应用较少，加大了存储成本，但是提升了性能。

版权声明：本文为 TiDB 社区用户原创文章，遵循 CC BY-NC-SA 4.0 版权协议，转载请附上原文出处链接和本声明。

评论



添加评论

评论



dba-kit 2024-04-24 14:13

dumpling其实是mydumper的高位替代，常用参数都是兼容mydumper来设置的，所以逻辑备份完全可以用dumpling来搞。甚至我现在备份MySQL数据也使用dumpling的，支持S3，GCP这些云存储协议太香了。

回复

<

1

>

互助与交流

活动

问答论坛

TiKV 社区

Chaos Mesh 社区

学习与应用

文档

专栏

视频课程

考试认证

典型案例

开发者指南

发现社区

TiDB User Group

问答之星

社区准则

联系我们

电子书

