title: MySQL日志
date: 2019-12-08 21:33:00
tags: MySQL技巧

categories: MySQL

haha

MySQL日志模块

redo-log（重做日志）

- MySQL如果每一次的更新操作都需要写进磁盘，然后磁盘也要找到对应的那条记录，然后再更新，整个过程 IO 成本、查找成本都很高
- MySQL 里经常说到的 WAL 技术，WAL 的全称是 Write-Ahead Logging，它的关键点就是先写日志，再写磁盘。InnoDB 引擎就会先把记录写到 redo log（粉板）里面，并更新内存，在系统不是那么繁忙时再将内存中数据写入磁盘
- redo-log大小固定，类似循环队列
- 为MySQL提供cache-safe功能，中途crash也能恢复

binlog（归档日志）

- MySQL 整体来看，其实就有两块：一块是 Server 层，它主要做的是 MySQL 功能层面的事情；还有一块是引擎层，负责存储相关的具体事宜。上面我们聊到的粉板 redo log 是 InnoDB 引擎特有的日志，而 Server 层也有自己的日志，称为 binlog（归档日志

两种日志比较

1. redo-log是innodb存储引擎的日志系统；binlog是MySQL的server层的日志系统
2. redo-log是物理日志，记录数据库那个页上发生了改变；binlog是逻辑日志，记录某一条具体的数据库操作
3. redo-log大小固定，循环写，类似循环队列；binlog采用追加方式，写满后跳转下一个日志文件，不会覆盖前面的文件

两阶段提交

- prepare状态与commit状态
- 如果不使用“两阶段提交”，那么数据库的状态就有可能和用它的日志恢复出来的库的状态不一致

小结

- redo log 用于保证 crash-safe 能力

```
innodb_flush_log_at_trx_commit=1
#这个参数设置成 1 的时候，表示每次事务的 redo log 都直接持久化到磁盘

sync_binlog=1
#这个参数设置成 1 的时候，表示每次事务的 binlog 都持久化到磁盘
```

- 两阶段提交保证两个日志保持逻辑上一致