MySQL常见问题

典型回答

1. MySQL主从延迟的原因都有什么？会导致什么问题?
答：主从延迟会导致从库读不到最新的数据，降低可用和数据一致性。同时一些旁路监控的中间件如：canal也受影响。导致主从延迟的原因有：

- 大事务，产生大量binglog，导致从库回放时间较长。解决方案：拆分成多个小事务，并开启并行复制。
- 无主键、无索引表更新，从库会产生大量全表扫描，导致回放时间较长。解决方案：表增加主键、索引。
- DDL时间过长，导致从库回放时间较长。解决方案：DDL之前查看官方文档是否支持OnlineDDL，否则在业务低峰期使用pt工具执行DDL。
- 从库提供读服务，大查询、慢查询需要等待MDL锁，阻塞回放binlog。解决方案：kill掉查询。
- 从库负载升高，没法及时回放binlog。解决方案：主要排查负载升高原因，看是否有慢查询、机器配置不够等问题。
- 主从网络问题，从库无法及时接收binlog。解决方案：抓包排查网络问题，检查是否网络带宽不够。

MySQL 主从复制的过程

2. `insert into select` 插入1000万数据怎么操作不影响性能
答：该操作通常用于归档，直接使用会导致大事务问题，可以使用`order by id limit`分批次插入数据，也可以使用pt工具进行归档。

3. SQL经常卡死怎么排查?
答：如果是DML，查看是否存在死锁、锁等待时间过长的问题。出现死锁，要从代码层面解决，也可以临时降低数据库隔离级别解决。如果是SELECT语句，检查慢查询日志中是否有慢SQL，可以通过查询计划观察并优化SQL。同时也需要注意是否因为有DML在执行，尤其是大事务，因为可能存在排它锁。另外需要观察磁盘和机器整体负载是否有升高。
数据库死锁如何解决?

4. 业务高峰CPU使用率高是为什么?
答：MySQL是多线程的，业务高峰时主要消耗资源包括：线程创建和销毁、上下文切换、网络连接建连、锁的争抢、写索引b+树分裂，慢查询，这些操作都可能导致CPU使用率高。
数据库CPU被打满排查过程

5. MySQL占用内存过高会不会被kill?
答：MySQL内存占用过高可能被操作系统OOMKiller给kill掉，尤其是在禁用swap之后。内存只用过高通常是因为buffer pool、连接池等配置过大导致的。

7. update小表，出现慢SQL有哪些原因?
答：小表通常只有几十或几百行记录，全表扫描的速度也很快。所以第一时间应该考虑是否存在锁竞争，其次看慢查询时的数据库和机器整体状态是否是因为繁忙导致的。

8. 数据倾斜怎么办?
答：数据倾斜是指分库分表后，如使用日期做拆分，某一天的流量突然增大单表数据量增大，可产生慢查询。解决方案，第一在事先选择均匀的拆分算法，如：hash(id)，第二对于出现的慢sql试优化，有索引但全表扫描的，可以尝试更新统计信息，或者`force index`强制走索引，第三可以考虑对数据手工拆分成多个表，但需要修改分片查询方法。注意，个问题最靠谱的解决方案应该是选择均匀的拆分算法。

9. 执行计划走错怎么办?
答：执行计划走错了，通常是因为表的统计信息不准确（尤其在大事务进行大量删除，插入时）可以进行`ANALYZE TABLE`或使用`force index`强制走索引。

10. 亿级别数据表加字段，怎么操作不影响业务?
答：首先考虑是否能使用OnlineDDL，不同MySQL版本使用OnlineDDL的类型和支持的SQL不同，可去官网查看。其次如果确定不能使用的话，可以使用PT工具修改表结构。注意，最好在业务低峰间执行。

11. 查询不存在的数据很慢，查询存在的数据很快为什么?
答：需要对比两个sql的查询计划，需要注意的是网上谣传`NOT IN`，`NOT EXISTS`等取反操作不走索引是错误的，这些也是会走索引的，只不过查询分析器在数据量大时是对索引树的叶节点数量和随机抽取几页中有多少行记录计算进行粗略统计的。而取反操作出的结果大部分情况占全表数据的比例较低，所以查询计划认为全表扫描比走索引性能高。

12. 12c48g能抗多少并发?
答：这个问题需要压测得出，MySQL本身能够承载的最大连接数和所能执行的事务是有限制的。其次也取决于你的sql是否存在锁竞争、慢查询。另外配置不合理也会导致MySQL承载的并发量低。需要注意的是，压测需要关注响应耗时，机器资源使用情况，并规定大于一定响应耗时就算作异常。阶梯性增大压测流量。

13. 什么情况全表扫描比索引扫描快?
答：注意这个问题不代表是索引设置不合理，或查询不可使用索引。返回的数据量大会走全表扫描，查询计划走索引同时要回表，所以会选择直接全表扫描避免回表开销。返回的数据超过表20%-30%，查询优化器会倾向于使用全表扫描，不会走索引。

14. 单表过亿，用什么数据库?
答：MySQL innoDB单表达到亿级别可能存在一些性能问题，但也不是绝对，如果你的表结构比

较简单且创建合理的索引问题不大。如果超过2亿或表结构复杂，则推荐进行分库分表或使用分布式数据库。

15. MySQL高可用方案有哪些?
答：官方提供了MGR+innoDB Cluster的方案，MySQL8.0此方案比较成熟。也可以使用第三方公司开发的MHA、PXC，轻量级的方案是keepAlive+双主架构。

16. 为什么创建过多索引会影响性能?
答：查询不会影响性能，索引主要的目的是优化查询性能。建立过多或错误的索引会对增删改产生影响，每个索引是一个b+树，增删改都需要维护b+树结构。同时锁也是加在索引上的，如果查询计划没有选择一个合适的索引，也会导致加锁效率低，影响性能。

17. 正则表达式能走索引吗?
答：MySQL8.0引入了函数索引，可以用其在一些正则表达式的场景上，但不是所有场景都能利用上函数索引。正则表达式放在查询中与全模糊查询基本相同，很难利用索引进行优化。在使用正则表达式的需求上，建议考虑在代码中实现，或fulltext索引进行优化，甚至可以使用ElasticSearch（当然ES上使用正则表达式也是很慢的操作）。

18. 为什么前缀模糊查询走索引，后缀不走?
答：索引是按照从左到右对每个字符进行排序的，多列索引也是按照从左到右进行排序的（即第一列值相同再用第二列排序，以此类推）。后缀模糊查询或全模糊查询是无法确定左侧字或者列有哪些值的，所以无法利用索引。但不是绝对，可以参考下面的文章进行优化：
MySQL 中 like 的模糊查询如何优化

19. 多个外连接会影响性能吗?
答：外连接的作表会固定为驱动表，查询分析器无法进行优化改变表的顺序，所以驱动表若过大，被驱动表扇出也会很大，即使被驱动表小也会影响性能。MySQL5.7使用nested-loop join性能不太好，8.0使用hash join性能有所提升，但驱动表过大会使用临时文件也会影响性能。

20. 对大表批量清洗数据需要注意什么?
答：主要注意清洗数据的读写QPS不要过大影响线上业务，不要出现大事务，不要出现慢查询，如果是删除动作最后要清理表空洞。所以：

- 清洗代码要限流控制QPS，可以使用guava的RateLimiter实现单机限流
- 使用 `order by id desc limit` 限制每次清洗数据的条数，建议不超过100，避免大事务导致占用较长时间的锁，避免undo膨胀，避免binlog主从复制延迟
- 提前使用查询计划看以下 `update` 或 `delete` 语句 `where` 条件是否命中索引
- 删除表后，使用 `ALTER TABLE 表名 ENGINE = InnoDB` 或 `OPTIMIZE TABLE 表名` 清理表空洞
- 建议清洗时打印反向日志，防止清洗错数据。虽然可以使用binlog恢复数据，但操作比较麻烦
- 建议清洗代码实现可暂停和可恢复，在业务低峰期清洗，在高峰期暂停清洗

扩展知识

查看MySQL锁的情况，使用如下SQL

```
SHOW ENGINE INNODB STATUS; //查看InnoDB状态，包含DEADLOCK区域就是死锁的信息

SHOW PROCESSLIST; //状态列中显示 "Locked" 或 "Waiting for lock" 的查询为当前加锁和等锁的SQL

SELECT //查询当前加锁的事务
    r.trx_id waiting_trx_id,
    r.trx_mysql_thread_id waiting_thread,
    r.trx_query waiting_query,
    b.trx_id blocking_trx_id,
    b.trx_mysql_thread_id blocking_thread,
    b.trx_query blocking_query
FROM
    INFORMATION_SCHEMA.INNODB_LOCK_WAITS w
    INNER JOIN INFORMATION_SCHEMA.INNODB_TRX b ON
        b.trx_id = w.blocking_trx_id
    INNER JOIN INFORMATION_SCHEMA.INNODB_TRX r ON
        r.trx_id = w.requesting_trx_id;
```

什么是pt

pt全称为percona-toolkit，源自Maatkit 和 Aspersa 工具。MySQL DBA工具箱必备工具。非常强大，有如下常用功能：

- pt-archiver 归档表
- pt-online-schema-change 修改表结构、索引增删改
- pt-table-checksum 校验主从数据一致性
- pt-table-sync 主从数据修复
- pt-heartbeat 监控 mysql 主从复制延迟
- pt-duplicate-key-checker 从 mysql 表中找出重复的索引和外键
- pt-kill kill掉MySQL命令无法kill的语句
- pt-slave-find 显示主从结构
- pt-slave-delay 设置从服务器落后于主服务器指定时间
- pt-slave-restart 监视 mysql 复制错误，当复制停止尝试重启 mysql 复制
- pt-show-grants 用户权限信息迁移
- pt-upgrade 用来检查在新版本中运行的 SQL 是否与老版本一样

- pt-index-usage 分析慢查询的索引使用情况
- pt-visual-explain 格式化 explain 出来的执行计划
- pt-config-diff 比较 mysql 配置文件和服务器参数
- pt-mysql-summary 精细地对 mysql 的配置和 sataus 信息进行汇总
- pt-query-digest 分析查询执行日志，并产生一个查询报告
- pt-diskstats 收集和显示磁盘使用情况
- pt-summary 收集和显示系统信息概况
- pt-stalk 出现问题的时候收集 mysql 的用于诊断的数据
- pt-find 查找 mysql 表并执行指定的命令

什么是MGR

MGR全称为MySQL Group Replication，即组复制，是MySQL5.7版本引入的新的主从复制策略。可用于InnoDB Cluster的集群方案。MGR+InnoDB Cluster有一些额外的限制：

1. 只能用于InnoDB 存储引擎，使用其他存储引擎可能产生错误
2. 表必须有主键
3. 使用了Paxos协议较为复杂，对网络性能和稳定性有一定要求
4. 必须基于statement和GTID复制
5. 对间隙锁支持的不好
6. 不支持外键
7. 不支持在不同服务器上对同一个表进行DDL或DML
8. 不支持SERIALIZABLE隔离级别
9. 多主模式下，`SELECT FOR UPDATE`容易导致死锁
10. 对事务时长有限制
11. 每个复制组中最多有9个成员

什么是表空洞

InnoDB为了实现MVCC，所以在执行`delete`语句时使用逻辑删除的方式，同时也可以在`insert`指定id插入时复用行记录。行记录中有一个隐藏标记为逻辑删除，而磁盘空间并不会清理掉。所以在大量删除表记录时，会发现表空间大小并没有缩小（反倒可能因为undo导致表空间增大），这时需要清理表空洞。

MySQL最大连接数和最大事务数

最大连接数可以使用如下SQL查询

```
show varaibles like '%max_connections%' //可以承载的最大连接数
show globl status like 'Max_used_connections' //当前使用的最大连接数
```

`max_connections`是可以修改的，同时也需要注意修改服务器最大文件描述符上限。通常保证`Max_used_connections`不超过`max_connections`的80%。
MySQL5.5.4之前最多支持1023个写事务，之后支持更多的回滚段，每个回滚段支持1023个写事务，能够支持更多的写事务。如果有大量未提交的事务超出回滚段最大上限，会报错`too many active concurrent transaction`。

OnlineDDL是什么?
什么是 OnlineDDL
查询哪些SQL支持OlineDDL *注意，右上角可以选择MySQL版本，不同版本支持的OnLineDDL不同。*