**MYSQL 全量备份、MYSQL 增量备份。定期还原**

通过mysqldump全量备份，mysqlbinlog增量备份。或者使用第三方工具xtrabackup

**MYSQL 主从复制**

**主从复制原理：**

从库生成两个线程，一个I/O线程，一个SQL线程。

I/O线程去请求主库的binlog，并将得到的binlog日志写到relay log(中继日志)文件中；主库会生产一个log dump线程，用来给从库i/o线程传binlog

SQL线程，会读取relay log文件中的日志，并解析成具体操作，来实现主从的操作一致性，而最终数据一致

**主服务器：**

1. 开启二进制日志
2. 配置server-id
3. 获取master二进制日志文件名及位置
4. 创建一个用于slave和master通信的账号

**从服务器：**

1. 配置唯一的server-id
2. 使用master分配的账号读取master二进制日志
3. 启用slave服务

**MYSQL** **主从复制延迟问题**

**延迟原理：**主库上DDL和DML产生binlog，binlog是顺序写，所以效率很高，slave的slave\_IO\_Running线程到主库取日志，效率也比较高。但是slave的Slave\_SQL\_Running线程将主库的DDL和DML操作在slave实施。DML和DDL的IO操作是随机的，不是顺序的，成本很高，还可能在slave上的其他查询产生lock争用，由于Slave\_SQL\_Running也是单线程的，所以应该DDL卡在了。需要执行10分钟，那么多所有之后的DDL会等待这个DDL执行完才会继续执行，这就导致了延迟

**主从延迟产生：**当主库的TPS并发比较高时，产生的DDL数量超过slave应该sql线程所能承受的范围，那么延迟就产生了，当然还有就是看你与slave的大型query语句产生了锁等待

**主从同步延迟解决方案：**最简单的减少slave同步延迟的方案就是在架构上做优化，尽量让主库的DDL快速执行。还有就是主库是写，对数据安全性要求较高，比如 sync\_binlog=1，innodb\_flush\_log\_at\_trx\_commit = 1 之类的设置，而slave则不需要这么高的数据安全，完全可以讲sync\_binlog设置为0或者关闭binlog，innodb\_flushlog也 可以设置为0来提高sql的执行效率。另外就是使用比主库更好的硬件设备作为slave。

**主从延迟主要几个因素：**网络延迟、master负载、slave负载

**MYSQL 优化**

1. SQL和索引优化
2. 优化数据库对象：选择合适的存储引擎、数据类型、分表分库
3. 优化存储引擎：优化特定参数
4. 优化MySQL Server：MySQL内存管理和优化、innodblog机制及优化、调整跟并发相关的MySQL参数
5. 磁盘IO优化
6. 应用优化

**MYSQL配置优化**

**基础配置**

**innodb\_buffer\_pool\_size**:调整缓冲池大小，缓冲池是数据和索引缓存的地方

**innodb\_log\_file\_size**：5.5以后redo日志默认大小限制在4GB(默认可以有2个log文件)，一开始就把innodb\_log\_file\_size设置成512M(这样有1GB的redo日志)会使你有充裕的写操作空间。如果你知道你的应用程序需要频繁的写入数据并且使用的是MYSQL5.6.你可以一开始就把它设置成4G

**max\_connections**:调整数据库连接数

**InnoDB配置**

**innodb\_file\_per\_table**：是否将所有表的数据和索引存放在共享表空间内

**innodb\_flush\_log\_at\_trx\_commit**：默认值为1，表示Innodb完全支持ACID特性

**innodb\_flush\_method**:这项配置觉得了数据和日志写入硬盘的方式。一般来说，如果你有硬件RAID控制器，并且其独立缓存采用write-back机制，并有着电池断电保护，那么应该设置配置为O\_DIRECT；否则，大多数情况下应将其设为fdatasync（默认值）。sysbench是一个可以帮助你决定这个选项的好工具。

**innodb\_log\_buffer\_size**: 这项配置决定了为尚未执行的事务分配的缓存。其默认值（1MB）一般来说已经够用了，但是如果你的事务中包含有二进制大对象或者大文本字段的话，这点缓存很快就会被填满并触发额外的I/O操作。看看Innodb\_log\_waits状态变量，如果它不是0，增加innodb\_log\_buffer\_size

**query\_cache\_size**:查询缓存， 最佳选项是将其从一开始就停用，设置query\_cache\_size = 0（现在MySQL 5.6的默认值）并利用其他方法加速查询：优化索引、增加拷贝分散负载或者启用额外的缓存（比如memcache或redis）。

**log\_bin**：开启二进制日志

**skip\_name\_resolve**：当客户端连接数据库服务器时，服务器会进行主机名解析，并且当DNS很慢时，建立连接也会很慢。