主从同步集群

读写分离的原理

分担MYSQL的压力；提升性能与吞吐量。

读写分离的原理

MYSQL需要有一个主服务器，主服务器执行所有的写操作

若干个从服务器，从服务器负责读操作。

主从同步：MYSQL所有的更改操作都写在BinLog二进制日志当中。

从服务器只需要连接到主服务器，将主服务器的BinLog日志读取。

从服务器将主服务器的操作重演一遍。

主从复制的延迟和网络传输数据丢失的问题：

MYSQL的主从同步提高了性能，但是牺牲了一定的一致性。数据具有延迟。

问题场景，如果一个业务在插入完成后立刻区查询数据库，就会查不到新插入的数据。

典型案例：mybatis的自增主键返回策略。

数据同步中断问题，节点在同步数据时，发生网络中断或者节点停止，就会造成数据不一致

解决同步问题的方法：

半同步第三方插件，当数据插入到主库时，操作并不会返回成功，直到所有的从库都接受到日志后才算作插入成功。半同步牺牲了写操作的性能。

Mysql的数据库拆分

垂直拆分

根据业务的不同，把不同业务的表拆分到多个数据库当中。

跨库的连表操作。

水平拆分

主从复制的结构

一主一从

这种结构并不会提升系统的性能，这种结构的作用主要是提高系统的容灾性（高可用）。主节点挂了，从节点可以顶替主节点的位置

这种方法不能用于系统的备份，因为删除操作也会同步到从库，无法保证数据的安全性。

一主多从

主节点的子节点不宜过多，否则会给主节点带来压力。一般一个主节点的从节点个数为两到三个。

有时候可以单拿出一个从节点，做特殊业务处理，作为单独慢查询的分析，或测试数据使用。

双主

主要用于解决写操作并发压力，插入的数据，根据Id进行散列，每一个主库负责存取不同段的数据，然后主库之间在进行同步。散列避免了两个库的数据重叠。

如果主库挂了，会出现诸多问题，所以非特殊情况之前介绍的两种结构方法即可。

（雪花算法）

环形多主

不建议使用。

淘宝在NOSQL未问世的时候曾经使用过，多个主节点环绕同步，每个主节点下还有多个从节点，用来解决压力非常大的并发问题。

主从的配置文件my.cnf

server-id=【数字】 每一个数据库都有自己独一无二的id

log-bin=mysql-bin文件的前缀，写死既可以

binlog-do-db=【数据库名称】需要同步的数据库

binlog-ignore-db=【数据库名称】不需要同步的数据库

这里不需要配置主从的ip端口号，因为ip和端口号在启动后动态在数据库中创建

MYSQL数据库的架构

MYCAT分库分表中间件

atlas主从的代理同步器。

PXC架构

PXC在接受到中间键的写操作后会将操作分发给其他PXC，直到确认所有的PXC都接受到数据后，才会让数据库执行提交操作。

缺点是性能比主从架构低，但是具有强一致性，数据不会丢失，不会有延迟问题

主从复制与PXC的选择，不需要强一致性就用主从复制，对强一致性要求高就用PXC

MYSQL的主从复制：

同步模式:

SBR：只记录被执行的sql语句，但是在使用某些函数时会出现不一致问题，如now()主库和从库获取的当前时间不一致。

RBR：记录被修改的数据和修改值，缺点是在类似修改表结构的时候，会大量产生日志，但是SBR模式下的日志只需要传输一条sql语句。

MBR：混合前两种模式,根据sql语句,mysql自行判断使用哪种保存方式。