# mysql高可用架构设计 - 凯凯王的技术生涯


https://blog.csdn.net/qq_23864697/article/details/79465772
主要介绍：复制功能介绍，mysql二进制日志，mysql复制拓扑，高可用框架，单点故障，读写分离和负载均衡

一 mysql复制功能介绍

mysql复制功能提供分担读负载

二 复制解决的问题

1 实现不同服务器上的数据分布

1.1 利用二进制增量进行

1.2 不需要太多的带宽

1.3 但是使用基于行的复制在进行大批量修改时会对带宽带来一定的压力，特别是跨IDC环境下进行复制

2 实现数据读取的负载均衡

2.1 需要其他组件配合完成

2.2 利用DNS轮询的方式把程序的连接到不同的备份数据库

2.3 利用LVS,haproxy这样的代理方式

2.4 非共享架构，同样的数据分布在多台服务器上

3 增强了数据安全性

3.1 利用备库的备份来减少主库负载

3.2 复制并不能代替备份

4 实现数据的在线升级

三 mysql二进制日志

1 mysql服务层日志

二进制，慢查日志，通用日志

2 mysql存储引擎层日志

innodb日志，重做日志，回滚日志

记录了所有对mysql数据库的修改事件，包括增删改查事件和对表结构的修改事件

四 二进制日志格式

1 基于段的日志格式（记录sql语句）

binlog_format=statement

优点：日志记录量相对较小，节约磁盘及网络i/o,只对一条记录修改或插入

缺点：必须要记录上下文信息（保证语句在从服务器和主服务器上执行结果一样），对于特定的函数如uuid(),user()这样的非确定函数还是无法复制，可能造成mysql复制的主备服务器数据不一致

2 基于行的格式

binlog_format=ROW

同一sql语句修改10000条数据的情况下，基于段的日志格式只会记录这个sql语句，基于行的日志格式会有10000条记录分别记录每一行的数据修改

优点：使mysql主从复制更加安全，对每一行数据的修改比基于段的复制高效，误操作而修改了数据库中的数据，同时又没有备份可以恢复时，我们就可以通过分析二进制日志，对日志记录的数据修改操作做反向处理的方式来达到恢复数据的目的

缺点：记录日志量较大

3 混合日志格式

binlog_format=minxed

特点：根据sql语句由系统决定在基于段和基于行的日志中进行选择，数据量的大小由所执行的sql决定

五 mysql日志格式对复制的影响

1 基于sql语句的复制

优点：生成日志量少，节约网络传输i/o；并不强制要求主从数据库的表定义完全相同；相对基于行的复制更加灵活

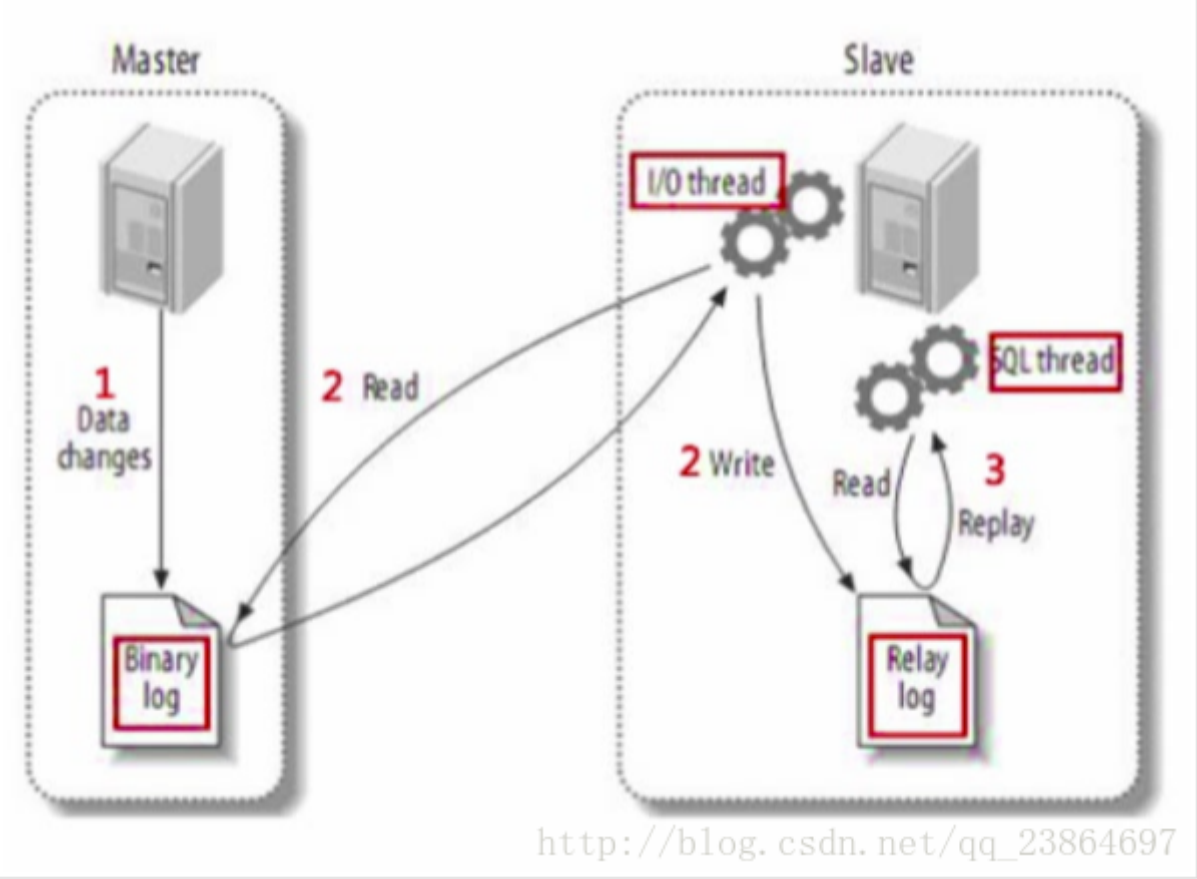
缺点：对于非确定事件，无法保证主从数据的一致性；对于存储过程、触发器、自定义函数进行的修改也可能造成数据不一致；相比基于行的复制方式在从上执行需要更多的行锁

2 基于行的复制

优点：可以应用于任何sql的复制包括非确定性函数，存储过程；可以较少数据库索的使用

缺点：要求主从数据库的表结构相同，否则可能会中断复制；无法在从上单独执行触发器

六 mysql的工作方式




步骤

主将变更写入二进制日志

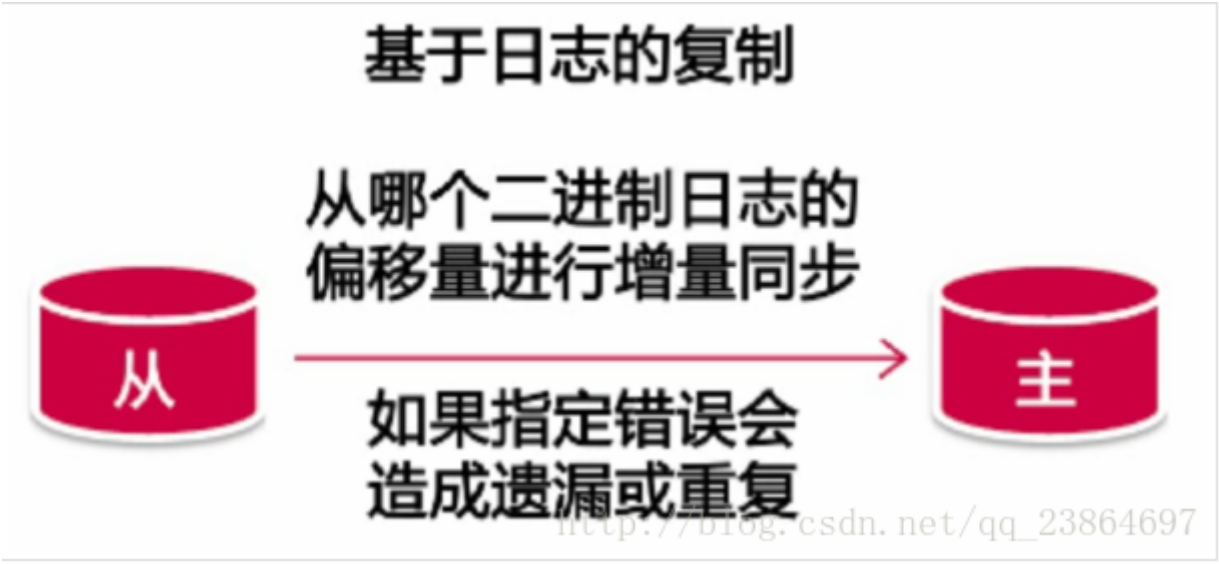
从读取主的二进制日志变更并写入到relay_log中

基于日志点的复制，基于GTID的复制

在从上重放relay_log中的日志

基于sql段的日志是在从库上执行记录的sql，基于行的日志则是在从库上直接应用对数据库的修改

七 基于日志点的复制




配置步骤

在主DB服务器上建立复制账号

create user 'repl'@'ip段' identified by ‘password’

grant replication slave on .to 'repl'@'ip段'

配置从数据库服务器

bin_log=mysql-bin

server_id=101

relay_log=mysql-relay-bin

log_slave_update=on

read_only=on

初始化从服务器数据

mysqldump -master-data=2-single-transaction

xtrabackup -slave-info

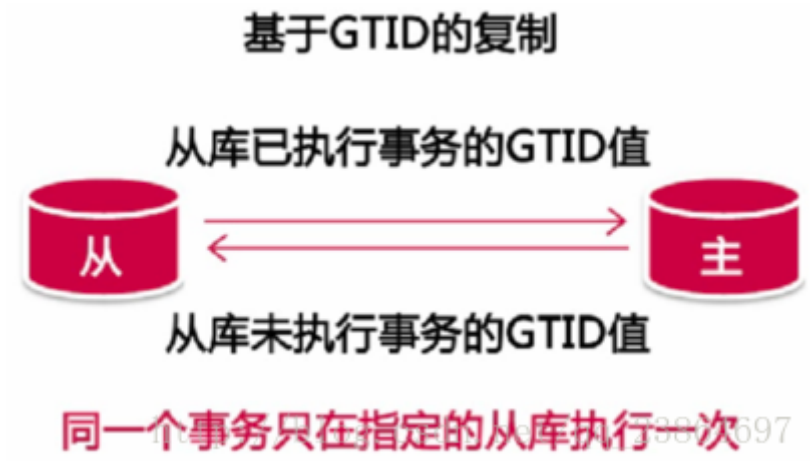
启动复制链路

change master to master_host="master_host_ip",master_user='repl',master_passwd=''password' master_log_file='mysql_log_file_name',master_log_pos=4

优点：是mysql最早支持的复制技术，bug相对较少；对sql查询没有任何限制；处理故障比较容易

缺点：故障转移是重新获取新主的日志点信息比较困难

八 基于GTID的复制




什么是GTID：全局事务id,其保证为每一个在主提交的事务在复制集群中可以生成一个唯一的id；

基于日志点的复制步骤

在主DB服务器上建立复制账号

create user 'repl'@'ip段' identified by 'passwd'

grant replication slave on .to 'repl'@'ip段'

配置从数据库服务器

bin_log=/usr/local/mysql/log/mysql-bin

server_id=100

gtid_mode=on

enforce-gtid-consiste

log_slave_update=on

read_only=on

初始化从服务器数据

mysqldump -master-data=2-single-transaction

xtrabackup -slave-info

启动基于GTID的复制

change master to master_host='master_host_ip',master_user='repl',master_password='password',master_auto_position=1;

优点：可以很方便的进行故障转移；从库上不会丢失上库上的任何修改

缺点：故障处理比较复杂；对执行的sql又一定的限制

选择复制模式要考虑的问题

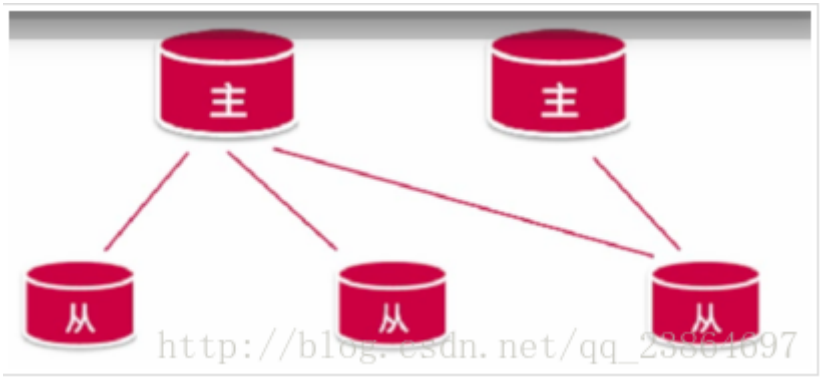
所使用的mysql版本

复制架构及主从切换方式

所使用的高可用管理组件

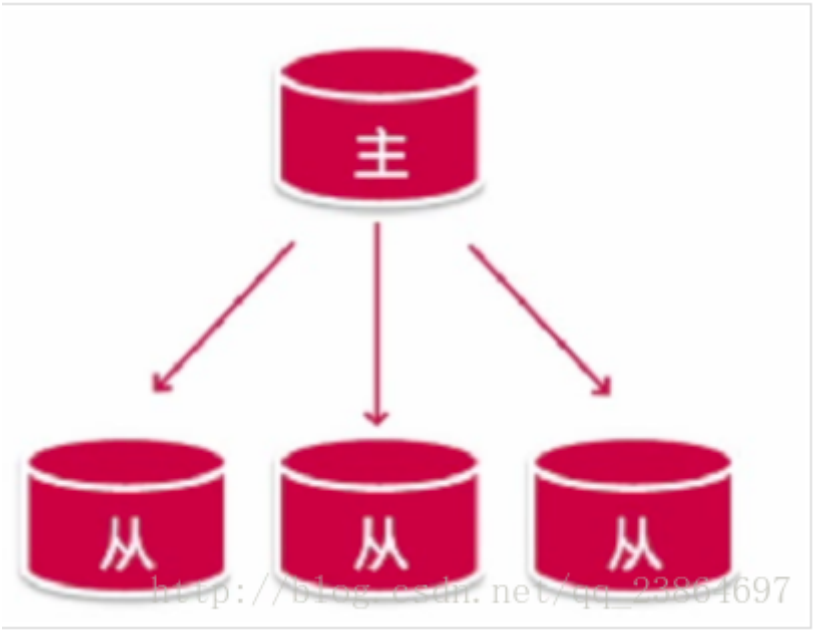
对应用的支持程度

九 mysql复制拓扑




mysql5.7之前，一个从库只能又一个主库，5.7之后，支持一从多主

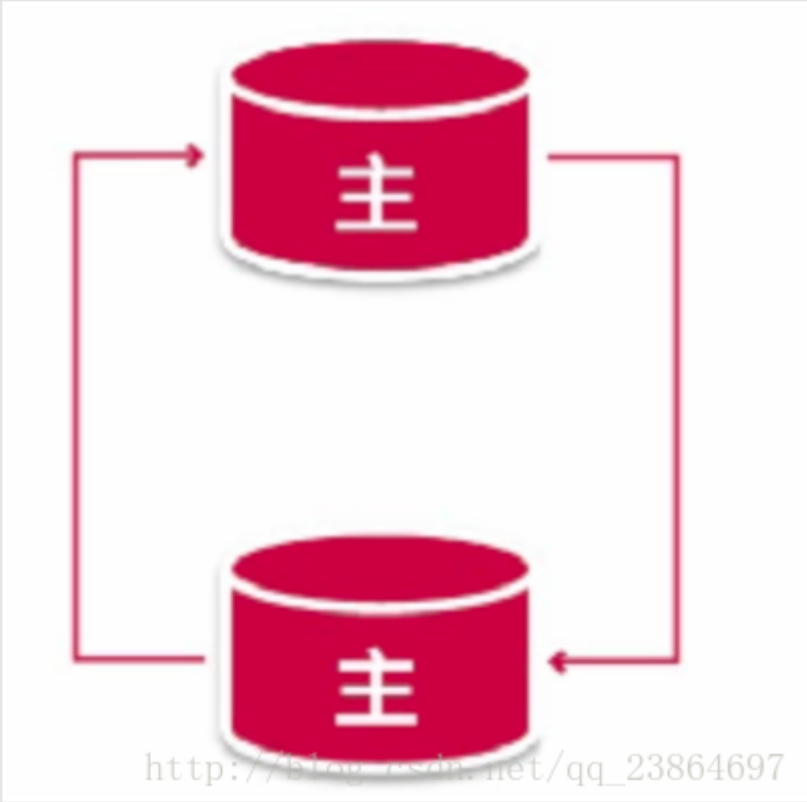
1 一主多从的复制拓扑




优点：配置简单，可以用多个从库分担读负载

用途：为不同的业务使用不同的从库；将一台从库放到远程IDC，用作灾备恢复；分担主库的读负载

2 主主复制拓扑




配置注意事项：两个主中所操作的表最好能够分开；使下面两个参数控制自增id的生成；

auto_increment_increment=2;

auto_increment_offset=1|2

主备模式下的主-主复制配置主要事项

只有一台主服务器对外提供服务

一台服务器处于只读状态并且只作为热备使用

在对外提供服务的主库出现故障或是计划性的维护时才会进行切换

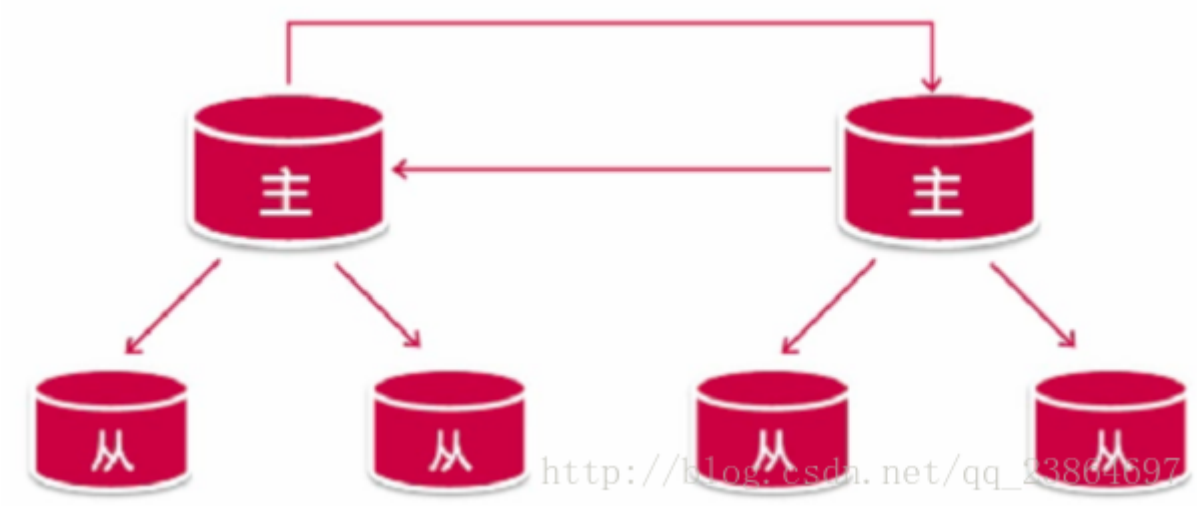
使原来的备库成为主库，而原来的主库会成为新的备库，并处理只读或是下线状态，待维护完成后重新上线

确保两台服务器上的初始数据相同

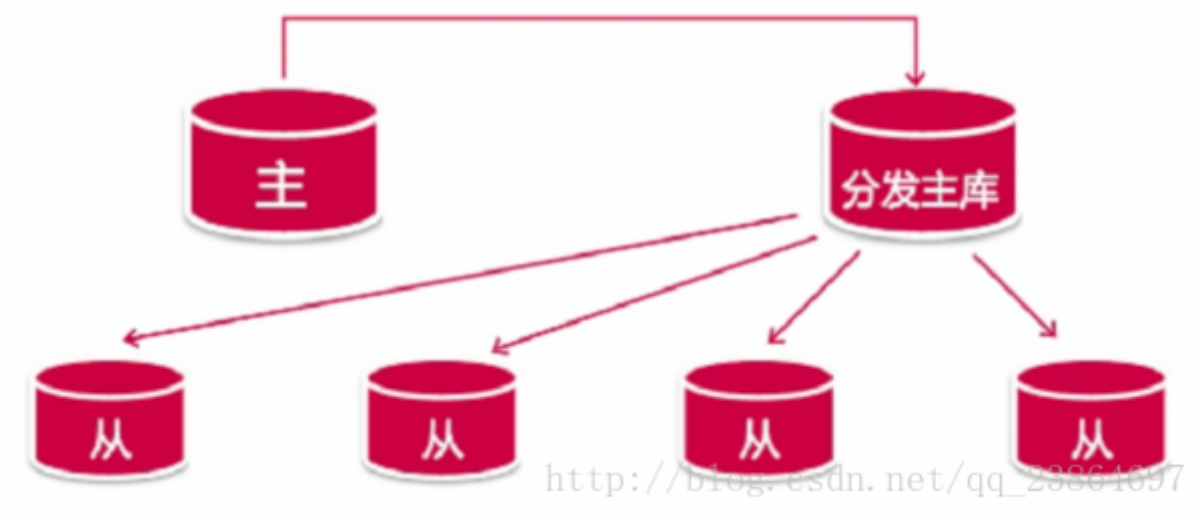
确保两台服务器上已经启动binlog并且又不同的server_id

在初始的备份上启动read_only

也可以给主库分配几个从库

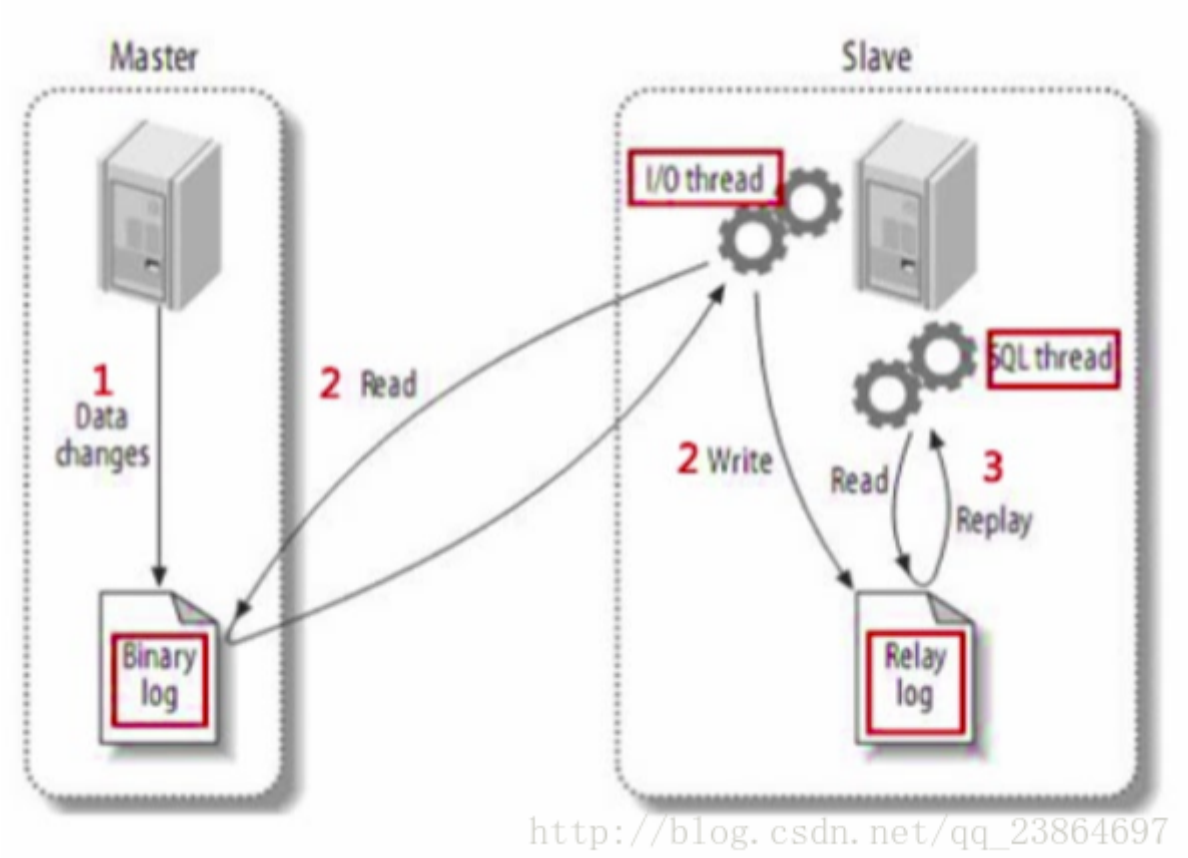



级联复制




十 mysql复制性能优化

影响主从延迟的因素




1 主库写入二进制日志的时间

解决办法：控制主库的事务大小，分割大事务

2 二进制日志的传输时间

解决办法：使用mixed日志格式或设置set binlog_row_image=minimal

3 默认情况下从库只有一个sql线程，主上并发的修改在从库变成了串行

解决办法：使用多线程复制，在mysql5.7中可以按照逻辑时钟的方式来分配线程

```
stop slave
set global slave_parallel_type='logical_clock'
set global slave_parallel_workers=4
start slave
```

十一 mysql复制常见问题处理

mysql数据损坏或丢失所引起的主从复制错误

使用跳过二进制日志事件

注入空事务的方式先恢复中断的复制链路

再使用其他方法来对比主从服务器上的数据

主库上的二进制日志损坏

备库上的中继日志损坏

在从库上进行数据修改造成的主从复制错误

十二 mysql复制无法解决的问题

分担数据库的写负载

自动进行故障转移及主从切换

提供读写分离功能

十三 高可用架构

什么是高可用：通过尽量缩短因日常维护操作（计划）和突发的系统崩溃（非计划）所导致的停机事件，以提高系统和应用的可用性

高可用的因子：正常可用时间，全年时间的百分比

引起系统不可用的原因：严重的主从延迟，主从复制中断，锁引起的大量阻塞，软硬件故障造成的服务器宕机

如何实现高可用

避免导致系统不可用的因素，减少系统不可用的时间

建立完善的监控及报警系统

对备份数据进行恢复测试

正确配置数据进行恢复测试

对不需要的数据进行归档和清理

增加系统冗余，保证发生系统不可用时可以尽快恢复

避免存在单点故障

主从切换及故障转移

原因

有服务器磁盘空间耗尽

性能糟糕的sql

表结构和索引没有优化

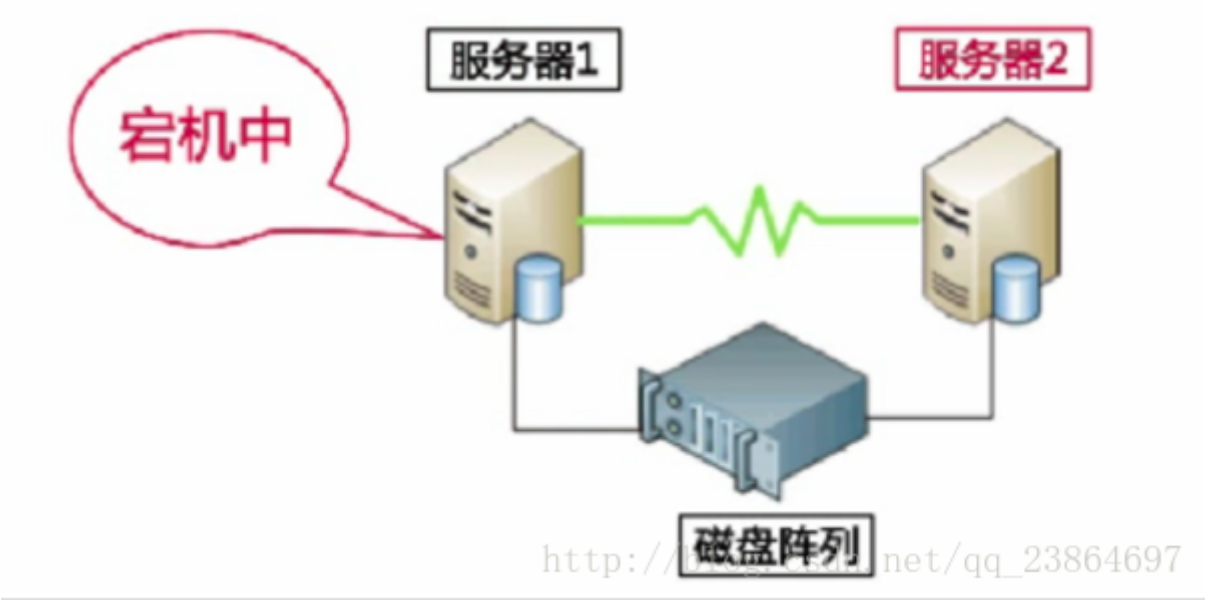
主从数据不一致

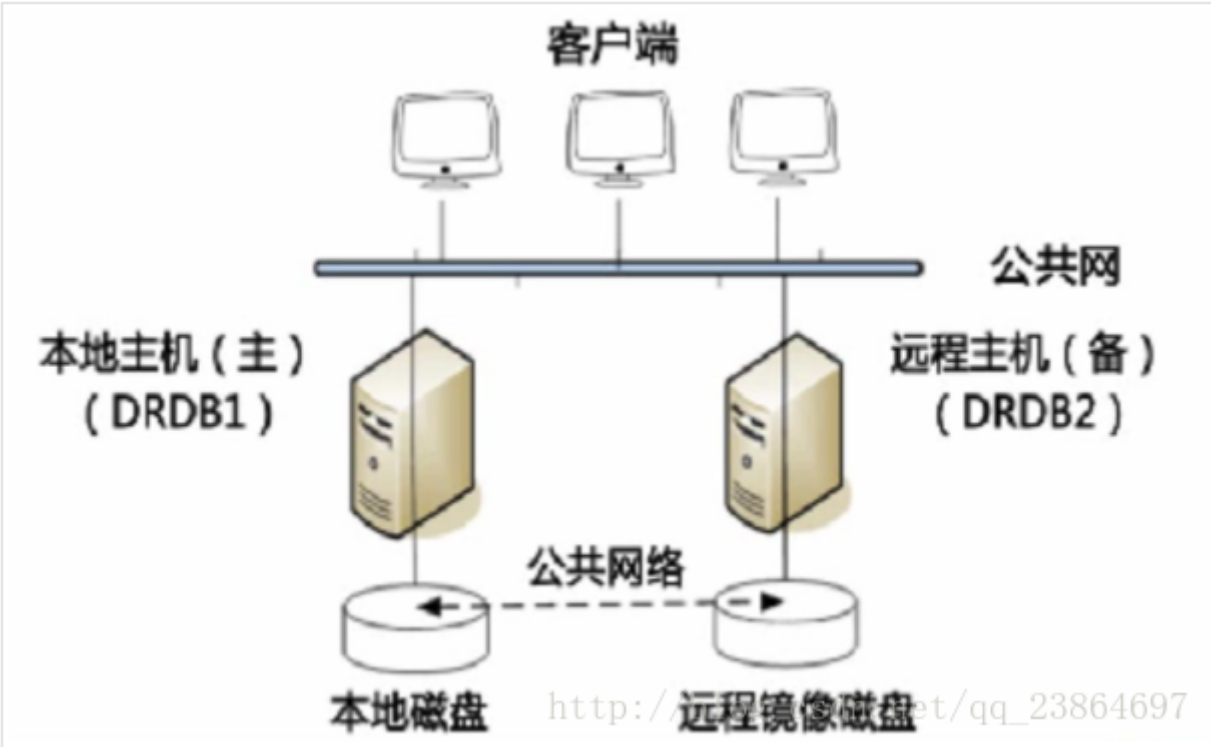
十四 单点故障

单点故障是指一个系统中提供相同功能的组件只有一个，如果这个组件失效来额，就会影响整个系统的正常使用

如何避免mysql单点故障

利用sun共享存储或drdb磁盘复制解决mysql单点故障




利用多写集群或ndb集群解决mysql单点故障


pxc架构图



如何解决主服务器的单点问题

主服务器切换后，如何通知应用新的主服务器ip地址

如何检查mysql主服务器是否可用

如何处理从服务器和新主服务器之间的复制关系

十五 MMM架构

Muti-master Replication Manager

1 MMM提供了什么功能

MMM监控mysql主从复制健康情况

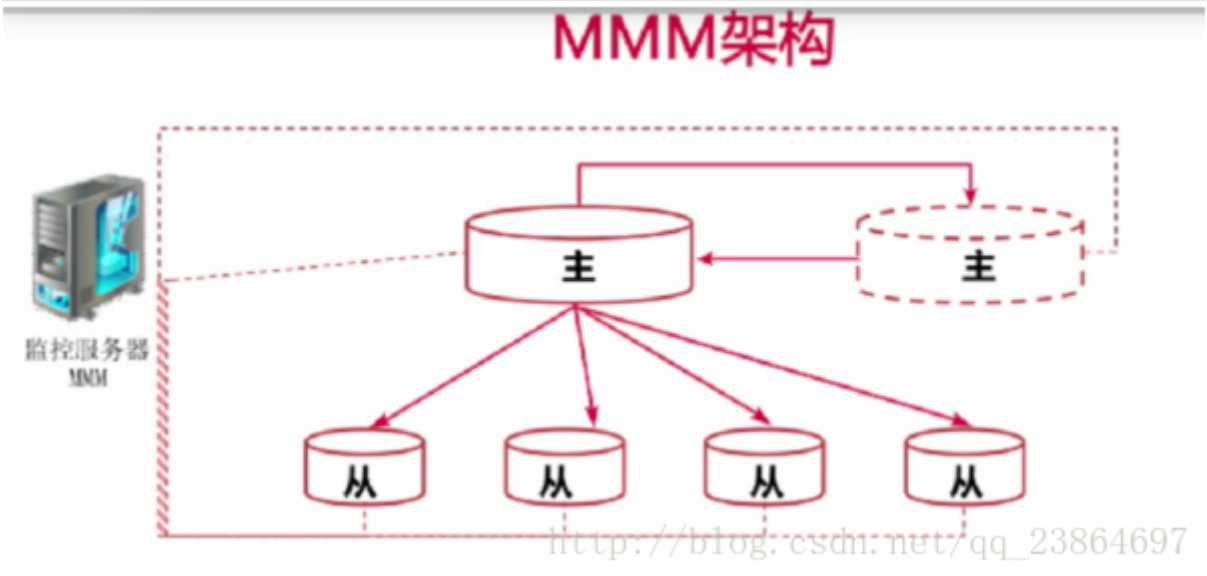
在主库上出现宕机进行故障转移并自动配置其他从对主的复制

如何找到从库对应的新的主库日志点的同步点

如果存在多个从库出现数据不一致的情况如何处理

提供了读写虚拟ip，在主服务器出现问题时，可以自动迁移虚拟ip

2 MMM架构




3 MMM部署所需资源

| 名称资源 | 数量 | 说明 |
| --- | --- | --- |
| 主DB服务器 | 2 | 用于主备模式的主主复制配置 |
| 从DB服务器 | 0-N | 可以配置0台或多台从服务器，但不建议太多 |
| 监控服务器 | 1 | 用于监控MySQL复制集群 |
| Ip 地址 | 2*(n+1) | n为MySQL服务器的数量 |
| 监控用户 | 1 | 用户于监控数据库状态的MySQL用户(replication client) |
| 代理用户 | 1 | 用户MMM代理的MySQL用户(super,replication client,process) |
| 复制用户 | 1 | 用户配置MySQL复制的MySQL用户(replication slave) |



4 MMM优缺点

优点：使用perl脚本语言开发及完全开源；提供了读写vip，使服务器的角色的变更对前端应用透明；mmm提供了从服务器的延迟监控

缺点：发布时间比较早不支持mysql新的复制功能；没有读负载的功能；在进行主从切换时，容易造成数据丢失；mmm监控服务存在单点故障

十六 MHA架构介绍

Master High Avaliablility

1 提供的功能

监控主数据库服务是否可用

当主DB不可用时，从多个从服务器中选举出新的主数据库服务器

提供了主从切换和故障转移功能

2 MHA主从切换过程

尝试从出现故障的主数据库保存二进制日志

从多个备选从服务器中选举新的备选主服务器

在备选主服务器和其他从服务器之间同步差异二进制数据

应用从原DB服务器保存的二进制日志

十七 读写分离和负载均衡介绍

进行mysql主从复制配置的一个主要目的：为了分担主库的读负载

1 为什么要读写分离

只能在主上进行写操作

读操作主和从上都可以

2 读写分离的两种方式

程序实现读写分离

优点：由开发人员控制什么样的查询在从库中执行，因此比较灵活；有程序直接连接数据库，所以性能损耗比较少

缺点：增加了开发工作量，是程序代码更加复杂；人为控制，容易出错

中间件实现读写分离

优点：由中间件根据查询语法分析，自动完成读写分离；对程序透明，对于已有程序不用作任何调整

缺点：增加了中间层，所以对查询效率有损耗；对于延迟敏感业务员无法自动在主库执行

读写分离与读的负载均衡区别

读写分离要解决的是如何在复制集群的不同角色上，去执行不同的语句

读的负载均衡主要解决的是具有相同角色的数据库，如何共同分担相同的负载

如何实现读的负载均衡

软件：lvs,haproxy,maxscale

硬件：f5