1.介绍

2.实践

2.1 分别在两台centos 7系统上安装mysql 5.7

2.2 master主服务器的配置

2.2.1 配置文件my.cnf的修改

2.2.2 创建从服务器的用户和权限

2.3 slave从服务器的配置

2.3.1 配置文件my.cnf的修改

2.3.2 连接master主服务器

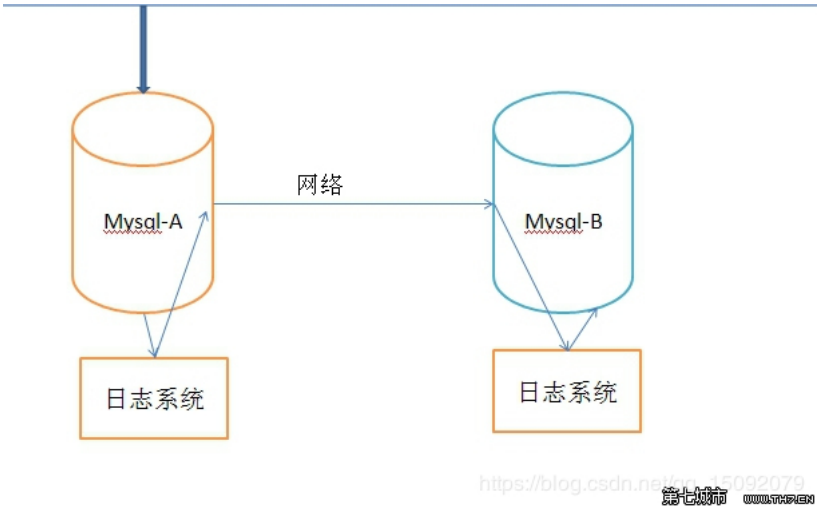
2.3.3 启动slave数据同步

2.3.4 读写分离的延迟问题

# 1.介绍

利用主从数据库来实现读写分离，从而分担主数据库的压力。在多个服务器上部署mysql，将其中一台认为主数据库，而其他为从数据库，实现主从同步。其中主数据库负责主动写的操作，而从数据库则只负责主动读的操作（slave从数据库仍然会被动的进行写操作，为了保持数据一致性），这样就可以很大程度上的避免数据丢失的问题，同时也可减少数据库的连接，减轻主数据库的负载。

原文：https://blog.csdn.net/qq_15092079/article/details/81672920




> 在上面的模型中，Mysql-A就是主服务器，即master，Mysql-B就是从服务器，即slave。

> 在Mysql-A的数据库事件（例如修改数据库的sql操作语句），都会存储到日志系统A中，在相应的端口（默认3306）通过网络发送给Mysql-B。Mysql-B收到后，写入本地日志系统B，然后一条条的将数据库事件在数据库Mysql-B中完成。
>
> 日志系统A，是MYSQL的日志类型中的二进制日志，也就是专门用来保存修改数据库表的所有动作，即bin log，注意MYSQL会在执行语句之后，释放锁之前，写入二进制日志，确保事务安全。
>
> 日志系统B，不是二进制日志，由于它是从MYSQL-A的二进制日志复制过来的，并不是自己的数据库变化产生的，有点接力的感觉，称为中继日志，即relay log。
>
> 通过上面的机制，可以保证Mysql-A和Mysql-B的数据库数据一致，但是时间上肯定有延迟，即Mysql-B的数据是滞后的。因此，会出现这样的问题，Mysql-A的数据库操作是可以并发的执行的，但是Mysql-B只能从relay log中一条一条的读取执行。若Mysql-A的写操作很频繁，Mysql-B很可能就跟不上了。

> 原文：https://blog.csdn.net/qq_15092079/article/details/81672920
>
> 大致流程：主数据库 负责写,会把所有的写操作记录到日志文件(二进制文件)中, 然后 从服务器来 同步这个日志文件,然后以此更新 从服务器上 的日志

主从同步复制有以下几种方式：

（1）同步复制，master的变化，必须等待slave-1,slave-2,...,slave-n完成后才能返回。

（2）异步复制，master只需要完成自己的数据库操作即可，至于slaves是否收到二进制日志，是否完成操作，不用关心。MYSQL的默认设置。

（3）半同步复制，master只保证slaves中的一个操作成功，就返回，其他slave不管。这个功能，是由google为MYSQL引入的。

本文说的是在centos 7系统上，实现的mysql5.7数据库的主从同步配置，从而实现读写分离操作。

原文：https://blog.csdn.net/qq_15092079/article/details/81672920

以下操作未经自己实践!!原文：https://blog.csdn.net/qq_15092079/article/details/81672920

# 2.实践

## 2.1 分别在两台centos 7系统上安装mysql 5.7

具体的安装步骤可以见此链接，https://blog.csdn.net/qq_15092079/article/details/81629238。

本文中的两台服务器的IP地址分别为主服务器（192.168.17.130）和从服务器（192.168.17.132）。

分别在这两个服务器上创建test数据库，以备后面测试。

## 2.2 master主服务器的配置

### 2.2.1 配置文件my.cnf的修改

```
#根据上一篇文章，编辑my.cnf文件
[root@localhost mysql]# vim /etc/my.cnf

#在[mysqld]中添加:
server-id=1
log_bin=master-bin
log_bin_index=master-bin.index
binlog_do_db=test
#备注:
#server-id 服务器唯一标识。
#log_bin 启动MySQL二进制日志，即数据同步语句，从数据库会一条一条的执行这些语句。
#binlog_do_db 指定记录二进制日志的数据库，即需要复制的数据库名，如果复制多个数据库，重复设置这个选项即可。
#binlog_ignore_db 指定不记录二进制日志的数据库，即不需要复制的数据库名，如果有多个数据库，重复设置这个选项即可。
#其中需要注意的是，binlog_do_db和binlog_ignore_db为互斥选项，一般只需要一个即可。
```

### 2.2.2 创建从服务器的用户和权限

```
#进入mysql数据库
[root@localhost mysql]# mysql -uroot -p
Enter password:

#创建从数据库的masterbackup用户和权限
mysql> grant replication slave on *.* to masterbackup@'192.168.17.%' identified by '123456';
#备注
#192.168.17.%通配符，表示0-255的IP都可访问主服务器，正式环境请配置指定从服务器IP
#若将 192.168.17.% 改为 %，则任何ip均可作为其从数据库来访问主服务器

#退出mysql
mysql> exit;
```

重启mysql服务

```
service mysql restart
```

查看主服务器状态

```
show master status;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| master-bin.000001 |      154 | test         |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
```

## 2.3 slave从服务器的配置

### 2.3.1 配置文件my.cnf的修改

#根据上一篇文章，编辑my.cnf文件
[root@localhost mysql]# vim /etc/my.cnf

#在[mysqld]中添加:
server-id=2
relay-log=slave-relay-bin
relay-log-index=slave-relay-bin.index
#replicate-do-db=test
#备注:
#server-id 服务器唯一标识，如果有多个从服务器，每个服务器的server-id不能重复，跟IP一样是唯一标识，如果你没设置server-id或者设置为0，则从服务器不会连接到主服务器。
#relay-log 启动MySQL二进制日志，可以用来做数据备份和崩溃恢复，或主服务器挂掉了，将此从服务器作为其他从服务器的主服务器。
#replicate-do-db 指定同步的数据库，如果复制多个数据库，重复设置这个选项即可。若在master端不指定binlog-do-db，则在slave端可用replication-do-db来过滤。
#replicate-ignore-db 不需要同步的数据库，如果有多个数据库，重复设置这个选项即可。
#其中需要注意的是，replicate-do-db和replicate-ignore-db为互斥选项，一般只需要一个即可

重启mysql服务
service mysql restart

### 2.3.2 连接master主服务器

change master to
master_host='192.168.17.130',master_port=3306,master_user='masterbackup',master_password='123456',master_log_file='mast
bin.000001',master_log_pos=154;

> #备注:
> #master_host对应主服务器的IP地址。
> #master_port对应主服务器的端口。
> #master_log_file对应show master status显示的File列：master-bin.000001。
> #master_log_pos对应show master status显示的Position列：154。

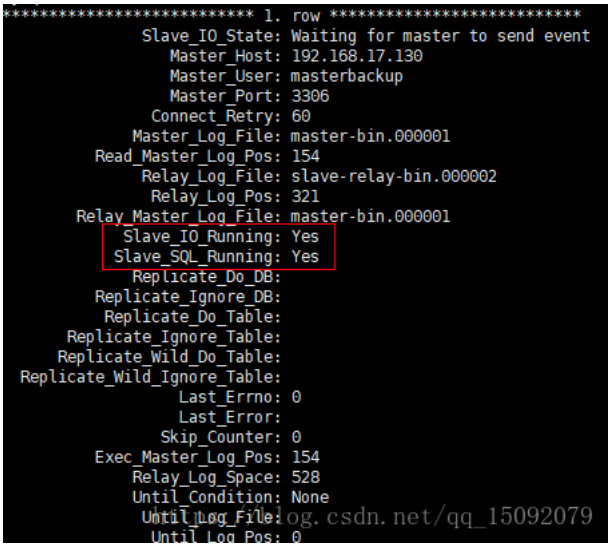
### 2.3.3 启动slave数据同步

#启动slave数据同步
mysql> start slave;

#停止slave数据同步（若有需要）
mysql> stop slave;

# 查看slave信息

show slave status\G; # 好像不要 \G



Slave_IO_Running和Slave_SQL_Running都为yes，则表示同步成功。

至此已完成!!!!,如果对 主数据库 进行写操作，在 从数据库 则会看到

> windows 下 https://blog.csdn.net/u010509052/article/details/80449134

#### 2.3.4 读写分离的延迟问题

> 使用场景: **数据量大的情况下使用的技术不是读写分离，是分表和分库，或者使用分布式存储引擎，读写分离不能解决数据量大的问题。**
> **https://bbs.csdn.net/topics/393486535**

如果对数据库进行了读写分离,那可能存在查不到刚刚写入的数据,因为同步数据需要时间,(不是高并发的情况下,一般不会出现这种问题,因为同步数据是从节点IO读取master的binlog日志)

方案大致有以下几种

1. 一个是半同步复制，用来解决主库数据丢失问题;

semi-sync复制，指的就是主库写入binlog日志之后，就会将强制此时立即将数据同步到从库，从库将日志写入自己本地的relay log之后，接着会返回一个ack给主库，主库接收到至少一个从库的ack之后才会认为写操作完成了

2. 一个是并行复制，用来解决主从同步延时问题。
3. 对于这种需要立即读的场景,指定(写库)数据库去查询
4. sleep一下,反正就是稍微等一下(例如插入之后做一些其他操作再读取)

> 原文链接：https://blog.csdn.net/wolf_love666/article/details/90444154