      1.2 mysql主从复制

mysql的主从复制方案,和 scp/rsync等文件级别同步时类似的 ,都是数据传输,只不过 mysql 无需借用第三方工具 ,而是自带多复制功能有一点不同 ,，另外一点，mysql 的主从复制并不是数据库磁盘上的文件直接拷贝复制 ,而是通过逻辑的binlog日志复制到同步数据库本地然后读取 SQL语句应用到数据库的过程 .

1.2.1mysql**主从复制介绍**

mysql的主从复制是一个异步的复制过程（虽然一般情况下是实时同步的），数据库数据从一个mysql数据库（我们称之为master）复制到另一个mysql数据库（我们称之为slave）。在master与slave之间实现整个主从复制到的过程是由三个线程参与完成是，其中有两个线程（SQL线程和IO线程在slave端），另一个线程（IO线程）在mater端。

要实现mysql的主从复制，首先必须打开master端的binlog（Mysql\_binxxxxxx）功能，否则无法实现主从复制。因为整个复制过程实际上就是slave从master端获取Binlog日志，然后再在slave自身上以相同顺序执行获得的binlog日志中所记录的各种操作。

mysql数据库支持单项双向链式级联等场景的复制，在复制过程中一台服务器充当主服务器（master）而一个或者多个其他服务器充当从服务器（slave）

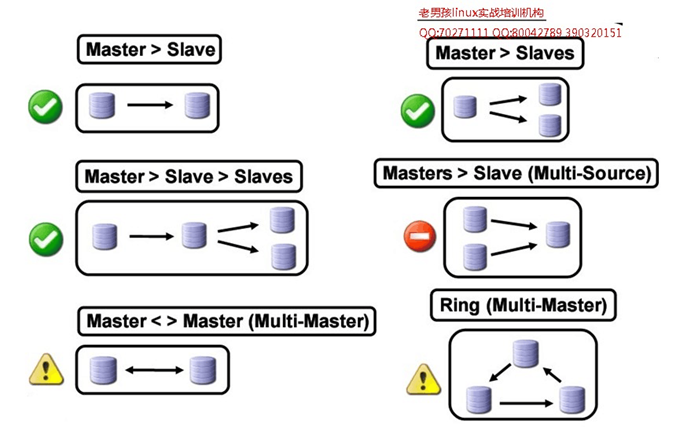
打开mysql的binlog可以通过在mysql的配置文件my.conf中的mysql模块增加

log-bin

[mysqld]

log-bin=/data/3306/mysql-bin

**mysql官方架构图：**

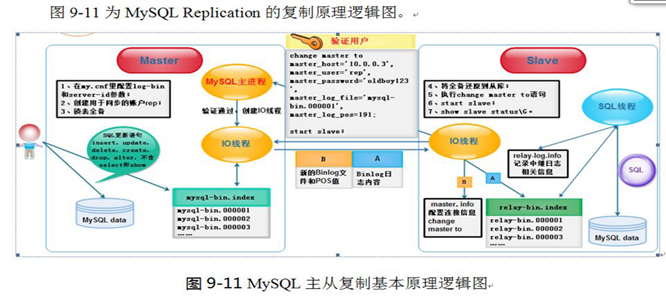


在当前的输出工作中，mysql主从复制都是异步复制方式，既不是严格的实时的数据同步。

1.2.2 mysql主从复制的应用场景

mysql主从复制有利于数据库的架构的健壮性，提升访问速度和易于维护管理。

mysql主从复制集群功能使得 MySQL数据库支持大规模高并发读写成为可能，同时有效保护物理服务器宕机场景的数据备份。



**应用场景1） 从服务器作为主服务器的实时数据备份**

主从服务器架构的设置, 可以大大的加强数据库架构的健壮性 。 例如: 当主服务器出现问题时, 我们可以人工或自动切换到从服务器继续提供服务, 此时从服务器的数据和宕机时的主数据库几乎是一致的 。 。

这类似我们前面课程中的 NFS存储数据通过 inotify+rsync同步到各份的NFS服务器非常类似,只不过MysQL的复制方案,是其自带的工具。

利用 MysQL的复制功能做数据备份,在硬件宕机,服务故障的场景数据各份是有效的, 但对于人为的执行drop,delete等语句删除数据的情况, 从库的各份功能就没用了, 因为从服务器也会执行删除的语句。

**应用场景2 )主从服务器实现读写分高,从服务器实现负裁均衡**

主从服务器架构可通过程序(php, java)或代理软件(mysql\_proxy,amoeba)实现对用户(客户端)的请求读写分离,即让从服务器仅仅处理用户的select查询请求,降低用户査询响应时间及读写同时在主阳長务器带来的压力 。 对于更新的数据 (update,insert,delete) 仍然交给主服务器处理, 确保主服务器和从服务器保持实时同步,

百度, 淘宝, 新浪等绝大多数的网站都是用户浏览页面多余用户发布内容, 因此通过在从服务器只按收读请求, 就可以很好的减轻主库的读压力, 且从服务器可以很容易的扩展到多台,且可以加上 Lvs做负载均衡,效果就非常棒了,这就是传说中的数据库读写分离架构，上述架构的逻辑图图下图所示：

**错误!未指定文件名。**

**应用场景3 )把多个从服务器根据业务重要性进行拆分访间**

可以把几个不同的从服务器 , 根据公司的业务进行拆分 。 例如: 有为外部用户提供査询服务的从服务器 ,有内部 DBA 用来数据各份的从服务器 , 还有为公司内部人员提供访问的后台、 脚本, 目志分析及开发人员使用的从服务器。 这样的拆分除了减轻主服务器的压力外。 使得数据库对外部用户浏览、 对内] 部用户业务处理 , 及DBA 人员各份等互不影响。 具体可以用下面的简单架构来说明 :

图：mysql主从复制根据业务重要性拆分

**提示：**

主从复制原理小结：

主从复制是异步的，逻辑的 SQL语句级的复制。

同步时，主库有一个 IO线程，从库有两个线程， IO和SQL 线程.

实现主从复制的必要条件，主库要开启 binlog功能.

binlog 文件只记录对数据库更改的 SQL语句（来自主数据库内容的变更） .

**主从复制条件：**

1. 主库开起bin.log

查询：grep -i "log-bin" my.cnf

配置文件添加log-bin = /data/3307/mysql-bin

2. 主从库之间的serverid不能一样。

grep server-id /data/{3306,3307}/my.cnf

3. 查看mysql参数

4. 检查二进制日志是否开启

show variables like “ log\_bin”;

5. 查看server-id

show variables like “ server\_id”;

6. 添加同步账户并限制权限

grant replication slave on \*.\* to rep@’ 172.16.1.%’ identified by  ‘123456’ ;

7. 检查用户是否创建成功

select user,host from mysql.user;

show grants for rep@'172.16.1.%';

8. 锁表(所有表只读)（不要退出当前会话，退出失效）

flush table with read loak;

9. 查看主库锁表后状态

show master status;

10. 备份主库

mysql –uroot – p123456 –events  –S /data/3306/mysql.sock – A –B|gzip >/opt/bak\_$(date +%F).sql.gz

11. 解锁（开放用户写入）

unlock table

12. 将备份数据恢复到从库上：

mysql -uroot -poldboy456 -S /data/3307/mysql.sock <bak\_2015-12-25.sql

13. 在从库上添加master.info信息

change master to master\_host=’ 172.16.1.52’,master\_port=3306,master\_user= ’rep’ ,master\_password=’qwer1234 ’,master\_log\_file=’ mysql-bin.000010’,master\_log\_pos=107;

14. 打开从库的slave，同步开关

start slave;

15. 查看同步状态：

show slave status\G

1.2.3实现MySQL 读写分离的方案

1、通过程序实现读写分离 (性能, 效率最佳 ,推荐)

php和 java程序都可以通过设置多个连接文件轻松的实现对数据库的读写分离 , 即当select 时,就去連接读库的连接文件 ,当 update,insert,delete时就连接写库的连接文件。

通过程序实现读,写分离的缺点就是需要开发对程序改造 , 对下层不透明, 但这种方式更容易开发和实现 , 适合互联网场景 。

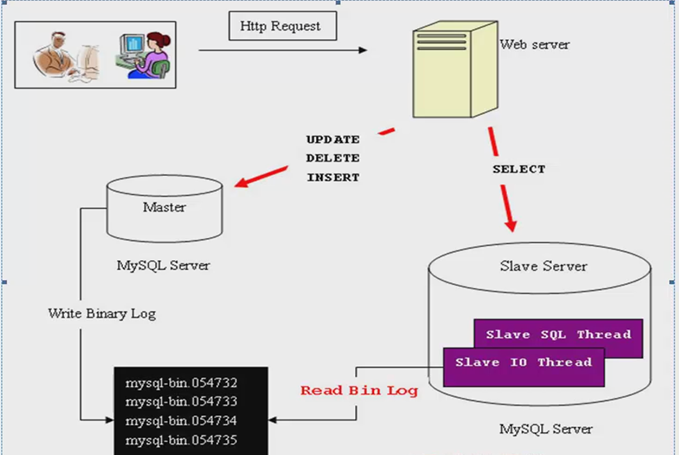
2、通过开源的软件实现读写分离

MySQL-proxy, Amoeba等代理软件也可以实现读写分离功能 , 这些软件的稳定性和功能一般 , 不建议生产使用。普通公司常用的还是通过客户端程序实现读写分离。

3、大型门户独立开发 DAL层综合软件

百度, 阿里等大型门户有牛人 , 会花大力气开发适合自己业务的读写分离、 负载均街、监控报警、 自动扩容、 自动收缩等一系列功能的 DAL层软件,

MySQL读写分离的基本逻辑图展示如下 :



1.3主从复制实践准备

1.3.1定义主从复制需要的服务器角色

主库及从库 IP端口：

主库   （mysql  master ）：[IP为 10.0.0.52   port3306]

从库 1（mysql   slave ） : [IP为 10.0.0.52  port3307]

从库 2 （mysql  slave ）：[IP为 10.0.0.52  port3308]

1.3.2主从复制数据库实战环境准备

[root@db02 application]# netstat -lnt|grep 330

tcp        0      0 0.0.0.0:3306                0.0.0.0:\*                   LISTEN

tcp        0      0 0.0.0.0:3307                0.0.0.0:\*                   LISTEN

tcp        0      0 0.0.0.0:3308                0.0.0.0:\*                   LISTEN

[root@db02 application]# grep server-id /data/{3306,3307,3308,3309}/my.cnf

/data/3306/my.cnf:server-id = 1

/data/3307/my.cnf:server-id = 3

/data/3308/my.cnf:server-id = 8

注意server-id 的值为 232-1

1.1.3数据库中英名称规定

mysql主库，也称master对应端口3306！

mysql从库，也称 slave 对应端口 3307！

mysql从库，也称 slave 对应端口 3308！

1.2.1实质server-id值并开启binlog参数

  根据前文的mysql的主从同步原理首先要开启binlog日志参数

1.    修改主库配置文件

执行vi /data/3306/my.cnf,编辑实例3306的在配置文件my.cnf，两个参数按如下内容修改

[mysqld]

server-id       = 1

log-bin=/data/3306/mysql-bin 可以省略红色部分

2.检查配置参数后的结果

 [root@db02 ~]# egrep "log-bin|server-id" /data/3306/my.cnf

/data/3306/my.cnf:log-bin = /data/3306/mysql-bin

/data/3306/my.cnf:server-id = 1

3.重启主库mysql服务

[root@db02 ~]# /data/3306/mysql stop

MySQL is stopped...

 [root@db02 ~]# /data/3306/mysql start

Starting MySQL...

[root@db02 ~]# netstat -lntup|grep 330

tcp        0      0 0.0.0.0:3306                0.0.0.0:\*                   LISTEN      11796/mysqld

登录数据库检查参数的更改情况：

[root@db02 ~]# mysql -uroot -p123456 -S /data/3306/mysql.sock

Welcome to the MySQL monitor.  Commands end with ; or \g.

Your MySQL connection id is 1

Server version: 5.5.32-log Source distribution

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show variables like 'server\_id';

+---------------+-------+

| Variable\_name | Value |

+---------------+-------+

| server\_id     | 1     |

+---------------+-------+

1 row in set (0.00 sec)

mysql> show variables like 'log\_bin';

+---------------+-------+

| Variable\_name | Value |

+---------------+-------+

| log\_bin       | ON    |

+---------------+-------+

1 row in set (0.00 sec)

mysql> grant replication slave on \*.\* to 'rep'@'10.0.0.%' identified by '123456';  授权

Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges; 刷新

Query OK, 0 rows affected (0.00 sec)

4.检查主库创建rep复制账号：

mysql> select user,host from mysql.user; 查看

+------+-----------+

| user | host      |

+------+-----------+

| rep  | 10.0.0.%  |

| root | 127.0.0.1 |

| root | localhost |

+------+-----------+

3 rows in set (0.00 sec)

mysql> show grants for [rep@"10.0.0.%](mailto:rep@%2210.0.0.%25) "; 查看权限

+-----------------------------------------------------------------------------------------------------------------------+

| Grants for rep@10.0.0.%                                                                                               |

+-----------------------------------------------------------------------------------------------------------------------+

| GRANT REPLICATION SLAVE ON \*.\* TO 'rep'@'10.0.0.%' IDENTIFIED BY PASSWORD '\*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9' |

+-----------------------------------------------------------------------------------------------------------------------+

1 row in set (0.09 sec)

mysql>

## 1.2.2对主数据库锁表只读（当前窗口不要关掉）

1.2.2.1 无需锁表备份的方案

在生产环境时, 一般会每天备份一份完整数据, 在备份时锁表各份并且记录下备份时的bin1og对应的文件以及位置点, 这样在实现主从复制的时候, 就无需现锁表备份了, 直接用夜里的全各即可完成主从同步的配置,另外,如果没有全各,那么最好等到夜里进行全备,也可能需要申请停机时间因为锁表期间,会影响业务小规模就不用了直接夜里mvsqldump锁表备份即可, 当然了务必要记录 bin1og的位置或者增加一一master-data=1。

mysql> flush table with read lock; 设置读锁

Query OK, 0 rows affected (0.00 sec)

  注意:锁表不能退出,退出即失效

mysqldump -uroot -p123456 -A -B --master-data=2 -x --events|gzip >/opt/all.sql.gz 企业用

[root@db02 ~]# mysqldump -uroot -poldboy123 -A -B --events -S /data/3306/mysql.sock|gzip >/server/backup/rep\_bak\_$(date +%F).sql.gz

[root@db02 ~]# ls -lrt /server/backup/

总用量 156

-rw-r--r-- 1 root root   2874 12月 22 00:00 oldboy\_2015-12-22.sql

-rw-r----- 1 root root    758 12月 22 10:04 mysql-bin.000138

-rw-r--r-- 1 root root   3020 12月 22 10:06 138bin.sql

-rw-r--r-- 1 root root 144736 12月 22 12:22 rep\_bak\_2015-12-22.sql.gz

 [root@db02 ~]#

mysql> show master status;  查看状态

+------------------+----------+--------------+------------------+

| File             | Position | Binlog\_Do\_DB | Binlog\_Ignore\_DB |

+------------------+----------+--------------+------------------+

| mysql-bin.000005 |      486 |              |                  |

+------------------+----------+--------------+------------------

1 row in set (0.00 sec)

mysql> unlock tables;     解锁

Query OK, 0 rows affected (0.00 sec)

 还原到从数据库:

[root@db02 3306]# cd /opt/

[root@db02 opt]# gzip -d bak\_2015-12-17.sql.gz

[root@db02 opt]# mysql -uroot -p123456 -S /data/3307/mysql.sock <bak\_2015-12-17.sql

[root@db02 opt]#

inodb引擎企业生产备份命令的推荐

mysqldump -uroot -p123456 -A -B -F --master-data=2 -x --single-transactin|gzip>/opt/all.sql.gz

--maste-data作用

1.使用—master-data=2进行备份文件会增加下面内容：适当普通备份增量恢复

--CHANGE MASTER TO MASTER\_LOG\_FILE=’mysql-bin.000020’,MASTER\_LOG\_POS=1191;



从库配置参数：

CHANGE MASTER TO

MASTER\_HOST='172.16.1.41',

MASTER\_PORT=3306,

MASTER\_USER='rep',

MASTER\_PASSWORD='123456',

MASTER\_LOG\_FILE='mysql-bin.000004',

MASTER\_LOG\_POS= 107;

开启从库开关

mysql> start slave;

查看状态

mysql> show slave status\G

解决查看状态错误的方法：（跳过不应该出现的错误）：

方法一：

stop slave; #<==临时停止同步开关。

set global sql\_slave\_skip\_counter =1 ; #<==将同步指针向下移动一个，如果多次不同步，可以重复操作。

start slave;

方法二：

[root@db02 backup]# grep slave-skip /data/3308/my.cnf

slave-skip-errors = 1032,1062

查看状态最重要的三个状态：

**错误!未指定文件名。**

生产场景下如何快速部署mysql主从复制

1.    快速配置mysql主从复制

步骤如下：

1）    安装好要配置从库的数据库，配置好log-bin和server-id参数

2）    无需配置my.cnf文件，主库的log-bin和server-id参数默认是配置好的

3）    登录主库，增加从库连接主库同步的账户，例如：rep 并授权replication同步权限

4）    使用曾经在半夜通过定时任务备份mysql=dump带-x和—master-data=1的命令及参数定时备份文件，把它恢复到从库

5）    在从库执change master to 语句无需binlog文件及对应位置点

6）    从库开启同步开关，start slave

7）    从库show slave status\G  检查同步状态，并在主库更新测试

2.    无需熬夜，轻松部署mysql主从复制

实战如下：

1）    半夜在主库通过定时任务执行如下命令，备份导出主库数据

mysqldump -uroot -poldboy123 -S /data/3306/mysql.sock -B -F -R -x --master-data=2 -A --events|gzip >/server/backup/rep3308\_$(date +%F).sql.gz

2）    白天找机会在需要做复制的备份数据库上导入全备做从库

gzip -d rep\_bak\_2015-12-27.sql.gz

mysql -uroot -p123456 -S /data/3307/mysql.sock </server/backup/rep\_bak\_2015-12-27.sql

mysql -uroot -p123456 -S /data/3307/mysql.sock<<EOF

CHANGE MASTER TO

MASTER\_HOST=10.0.0.52',

MASTER\_PORT=3306,

MASTER\_USER='rep',

MASTER\_PASSWORD='123456';

EOF

mysql主从复制线程状态说明及用途

1.    mysql主从复制主库I/O线程状态说明

1）    登录数据库从库mysql线程同步状态

mysql> show processlist\G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

     Id: 3

   User: system user

   Host:

     db: NULL

Command: Connect

   Time: 745

  State: Waiting for master to send event

   Info: NULL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

     Id: 6

   User: root

   Host: localhost

     db: NULL

Command: Query

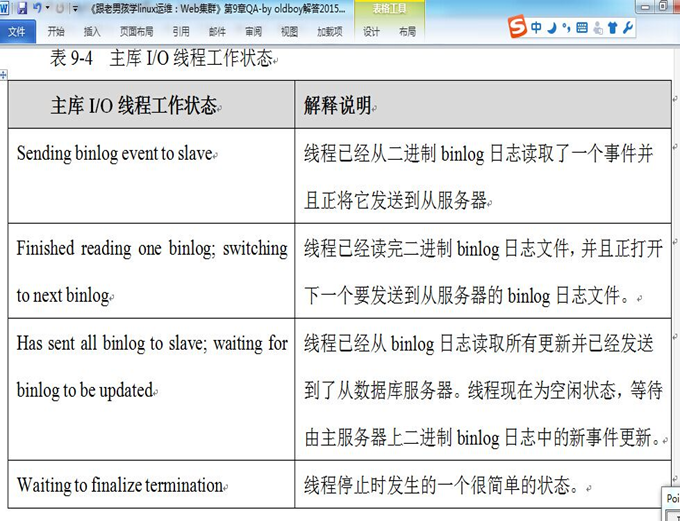
   Time: 0

  State: NULL

   Info: show processlist

2 rows in set (0.00 sec)

主库 I/O线程工作状态;



2)登录从库从库mysql线程工作状态，从库有两个线程即 I/O和SQL 线程

  下面是从库IO线程状态

mysql> show processlist\G

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

     Id: 3

   User: system user

   Host:

     db: NULL

Command: Connect

   Time: 745

  State: Waiting for master to send event

   Info: NULL

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

     Id: 6

   User: root

   Host: localhost

     db: NULL

Command: Query

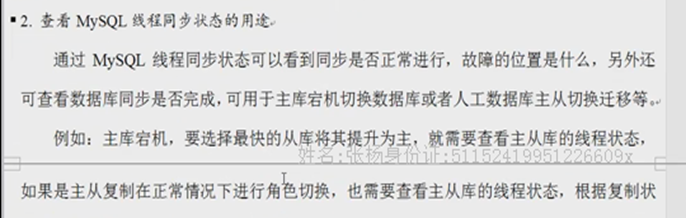
   Time: 0

  State: NULL

   Info: show processlist

2 rows in set (0.00 sec)

2.    从库mysql线程同步状态的用途



mysql主从复制更多应用技巧实践

工作中mysql从库停止复制故障案例

mysql> show slave status\G

             Slave\_IO\_Running: Yes

            Slave\_SQL\_Running: No

              Replicate\_Do\_DB:

          Replicate\_Ignore\_DB: mysql

           Replicate\_Do\_Table:

       Replicate\_Ignore\_Table:

      Replicate\_Wild\_Do\_Table:

  Replicate\_Wild\_Ignore\_Table:

                   Last\_Errno: 1008

                   Last\_Error: Error 'Can't drop database 'kkk'; database doesn't exist' on query. Default

**对于该冲突，**

解决方法1为：

mysql> stop slave  ####临时停止同步

mysql> set global sql\_slave\_skip\_counter=1;

mysql> start slave;

解决方法2：跳过错误

[root@db02 backup]# grep slave-skip /data/3308/my.cnf

slave-skip-errors = 1032,1062，1007

其他可能引起复制故障的问题：

1. mysql自身及人为重复插入数据.

2. 一般数据库版本也会引起不停步，低版本到高版本可以，但是高版本不能往低版本

3. mysql的运行错误或者程序BUG

4 binlog记录模式，例如row level 模式比默认的语句模式要好.

## 让mysql从库记录binlog日志的方法：

从库复制记录binlog的应用场景为：当前的从库还要作为其他从库的主库，例如：

级联复制或双主互从场景的情况下。下面介绍一下从库记录binlog的方法：

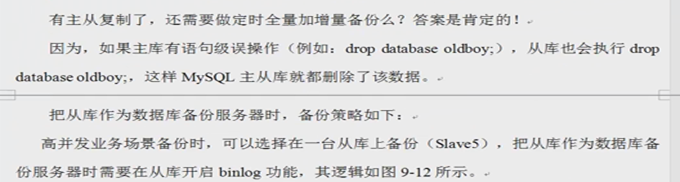
在从库my.cnf加入如下参数，重启生效：

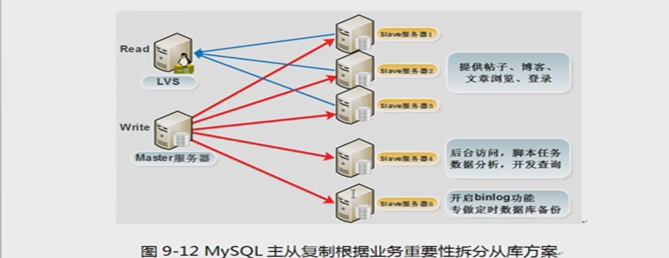
log-slave-updates #必须有这个参数

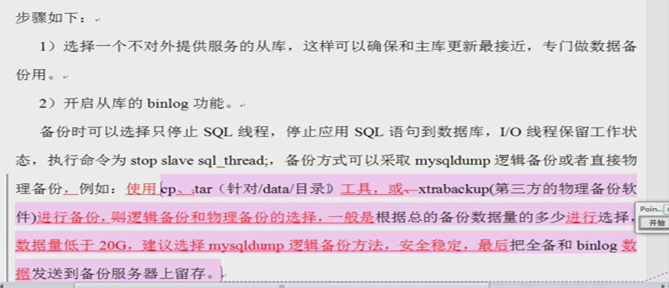
log-bin = /data/3307/mysql-bin

expire\_logs\_days = 7

1. mysql 主从复制集群架构的数据备份策略







MySQL主从复制延迟问题原因及解决方案

问题一：一个主库的从库太多，导致复制延迟。

建议从库数量3~5个位宜，要复制的从节点数量过多，会导致复制延迟。

问题二：从库硬件比主库差，导致复制延迟。

查看 master和slave 的系统配置，可能会因为机器配置的问题。

问题三：慢 SQL语句过多

例如一条 SQL语句，执行时间是20秒，那么从执行完毕，到从库上能查到数据也至少是 20秒，这样就延迟20秒了。

问题四：主从复制的设计问题。

例如，主从复制单线程，如果主库写并发太大，来不及传送到从库就会导致延迟。更高版本的 MySQL可以支持多线程复制，门户网站则会自己开发多线程同步功能。

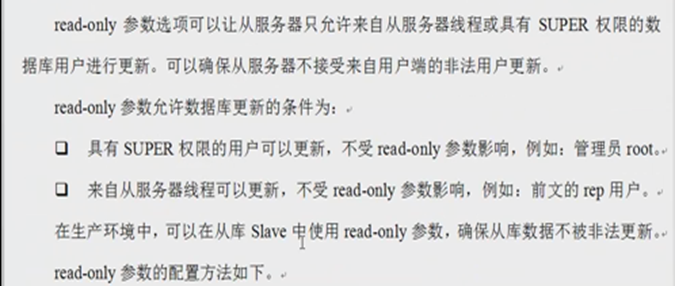
问题五：主从库之间的网络延迟。

主从库的网卡、网线、连接的交换机等网络设备都可能成为复制的瓶颈，导致复制延迟，另外，跨公网主从复制很容易导致主从复制延迟。

问题六：主库读写压力大，导致复制延迟。

主库硬件要搞好一点，架构的前端加 buffer以及缓存层。

通过read-only参数从库只读访问



方法一：启动数据库直接带 —read-only参数启动或重启，使用：

killall mysqld

或

mysqldump –uroot –p123456 –S /data/3307/mysql.sock shutdown

mysql\_safe –defaults-file=/data/3307/my.cnf –read-only &

方法二：在配置文件 [mysqld]模块加read-only 参数，并重启数据库

      [mysqld]

      read-only

授权

mysql> grant insert,update,create on oldboy.\* to 'web'@'172.16.1.%' identified by '123456';

mysql> flush privileges;

[root@db02 3307]# mysql -uweb -p123456 -h 172.16.1.52 -P3307

mysql> use oldboy

mysql> select \* from t1;

+----+----------+

| id | name     |

+----+----------+

|  1 | oldboy   |

|  3 | oldboy   |

|  5 | oldgirl  |

|  6 | oldboy1  |

|  8 | oldgirl2 |

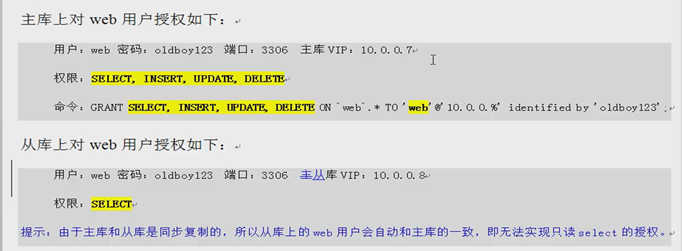
+----+----------+

5 rows in set (0.00 sec)

mysql> delete from t1;

ERROR 1290 (HY000): The MySQL server is running with the --read-only option so it cannot execute this statement

2.    web用户专业设置方案：mysql主从复制读写分离集群



主库

mysql> grant insert,update,delete,select on oldboy.\* to web@'172.16.1.%' identified by '123456';

mysql> flush privileges;

mysql> select user,host from mysql.user;

+------+------------+

| user | host       |

+------+------------+

| root | 127.0.0.1  |

| rep  | 172.16.1.% |

| web  | 172.16.1.% |

| root | localhost  |

+------+------------+

4 rows in set (0.00 sec)

从库

mysql> show grants for web@'172.16.1.%';

+-------------------------------------------------------------------------------------------------------------+

| Grants for web@172.16.1.%                                                                                   |

+-------------------------------------------------------------------------------------------------------------+

| GRANT USAGE ON \*.\* TO 'web'@'172.16.1.%' IDENTIFIED BY PASSWORD '\*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9' |

| GRANT SELECT, INSERT, UPDATE, DELETE ON `oldboy`.\* TO 'web'@'172.16.1.%'                                    |

+-------------------------------------------------------------------------------------------------------------+

2 rows in set (0.00 sec)

mysql> REVOKE insert,update,delete ON oldboy.\* from 'web'@'172.16.1.%';  回收权限

mysql> show grants for web@'172.16.1.%';

+-------------------------------------------------------------------------------------------------------------+

| Grants for web@172.16.1.%                                                                                   |

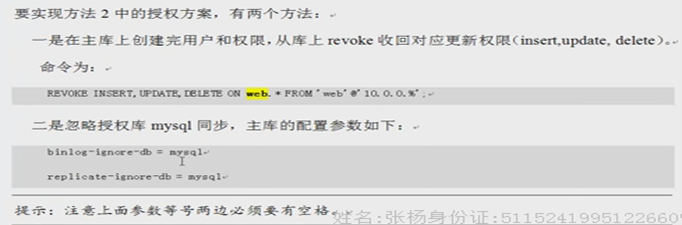
+-------------------------------------------------------------------------------------------------------------+

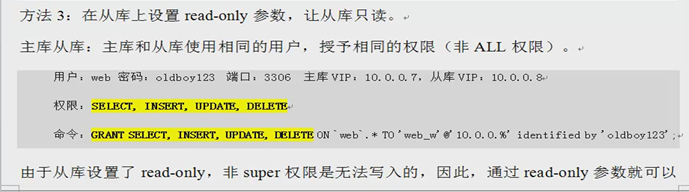
| GRANT USAGE ON \*.\* TO 'web'@'172.16.1.%' IDENTIFIED BY PASSWORD '\*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9' |

| GRANT SELECT ON `oldboy`.\* TO 'web'@'172.16.1.%'                                                            |

+-------------------------------------------------------------------------------------------------------------+

2 rows in set (0.00 sec)





DO：同步少量库

binlog-do-db=db\_oldboy

replicate-do-db=db\_oldboy-------------->#如需跨数据库更新并且不想复制这些更新，不应使用该选项

replicate-do-table=db\_oldboy----------->#允许跨数据库更新。

replicate-wild-do-table=db\_oldboy------>#用于跨数据库更新

ignore：排除

binlog-ignore-db=mysql

replicate-ignore-db=mysql-------------->#如需跨数据库更新并且不想复制这些更新，不应使用该选项

replicate-ignore-table=mysql----------->#该选项可以跨数据库进行更新。

replicate-wild-ignore-table=db\_oldboy-->#该选项可以跨数据库进行更新

**一主多从主库master宕机**

M --->S1

  --->S2

  --->S3

  --->S4

  --->S5

是否事先指定接班人，太子，接班人如何选择：

1）半同步从库（谷歌半同步插件 MySQL5.5是自带的）

-----S1作为太子，

第一：主库插入数据后，同时写入到 S1，成功返回。

优点：

两台库同时写入数据。

缺点：

写入会慢，网络不稳定时，主库持续等待。

解决措施：

1、连不上S1的时候会自动转为异步。

2、设置10秒超时，超过10秒转为异步。

3、S1网络，硬件要好，不提供服务，干等接管。

2）：S1 ，啥也不干，只做同步的从库， 500台服务器，百度。

3）皇帝驾崩现选（耽误事，容易被篡位）

确定主之后，要进行角色切换，将 S1提升为新主M1：

主库宕机有两种情况：

1、如果主库可以SSH连接，binlog数据没丢，要把主库的binlog补全到所有从库。

   a.提升S1位M1的操作

      1）调配置read-only。授权用户select，变成增删改查，开启binlog。如果做了双主同步，就直接切即可，啥都不用做。

      2）删除 rm -f master.info relay-log\*

      3）登录数据库reset master

      4）重启数据库，提升S1位M1完毕。

   b.所有从库：

CHANGE MASTER TO MASTER\_LOG\_FILE='mysql-bin.000002', MASTER\_LOG\_POS=191;

 M1 --

  --->S2

  --->S3

  --->S4

  --->S5

2、如果主库连不上:

   a.选择半同步从库提升主库，把半同步数据补全到所有从库。

     半同步从库提升主库操作如下，所有从库CHANGE MASTER同1-b。

      1）调配置read-only。授权用户select，变成增删改查，开启binlog。如果做了双主同步，就直接切即可，啥都不用做。

      2）删除 rm -f master.info relay-log\*

      3）登录数据库reset master

      4）重启数据库，提升S1位M1完毕。

   b.如果是S1啥也不干的方法

     提升啥也不干的S1为主库M1，操作见1-a步骤，所有从库CHANGE MASTER同1-b.

   c.主库宕机，没有事先指定哪个从库为主库。

     选主的方法：

1）确保IO和SQL都读取了自己的log，并运用到数据库

     1）登录所有从库 show processlist;看两个线程的更新状态。

     2）登录从库 32 33分别查看谁更新的更快（双主5从）：

cat /data/3306/data/master.info

cat /data/3307/data/master.info

确保更新完毕，看看4个从库哪个最快，经过测试没有延迟的情况POS差距很小，甚至是一致的

14

mysql-bin.001508

3024235

192.168.1.35

repl

XXXX

3306

60

0

相同文件选个POS最大的作为主库，例如：32 3306

选择文件及位置点最大的为主库，

 3）切换角色：

提升选择的S1为主库M1，操作见1-a步骤，所有从库CHANGE MASTER同1-b.

补全（通过中继日志）所有其他从库，和当前准备为主的数据一致。

把选定的太子提升为主的操作步骤：

第九章 mysql主从复制

1. 确保所有relay log全部更新完毕

在每个从库上执行stop slave io\_thread；show processlist;

直到看到has read all relay log 表示从库更新都执行完毕；

1. 从库提升为主

登录：mysql – uroot –p123456 –S /data/3306/mysql.scok

stop slave

retset master

quit

1. 进到数据库目录，删除master.info relay.info

cd /data/3306/data

rm – rf master.info relay.info

检查mysql授权表（web用户权限从库同步的权限）是不是正确的，

read-only等参数

确认bin-log是否开启生效

4.32 3306 提升为主库

vi /data/3306/my.cnf

开启

log-bin =/data/3306/mysql-bin

//如果存在log-slave-updates read-only一定要注释掉

/data/3306/mysql restart

到此为止，提升主库完毕

5.如果主库服务器没有挡掉，需要去主库拉取binlog补全提升主库的从库

其他从库操作

已经检查（同步user rep 都存在）

登录从库

stop slave

change master to master\_host = ‘ 192.16.1.32’; //如果不同步就指定位置点

start salve

show slave status\G

=====================主库宕机切换成功

6.修改程序配置文件主数据库35的指向32

平时访问数据库用域名则直接可以修改hosts解析，

如果做了VIP漂移就指定位置点

7.修理损坏 的主库，完成后作为从库使用，或者切换回来（尽量不切，除非配置有区别）

MHA高可用工具就是利用上面讲的原理实现的；

MySQL+HA+DRBD高可用场景

MySQL+MMM

MySQL +MHA

我们现在讲的是主库以外宕机，加入我们有计划切换怎么做？

1. 主库锁表
2. 登录所有数据库查看同步状态，是否完成，其他与前面一样

1