MySQL 实现主从复制

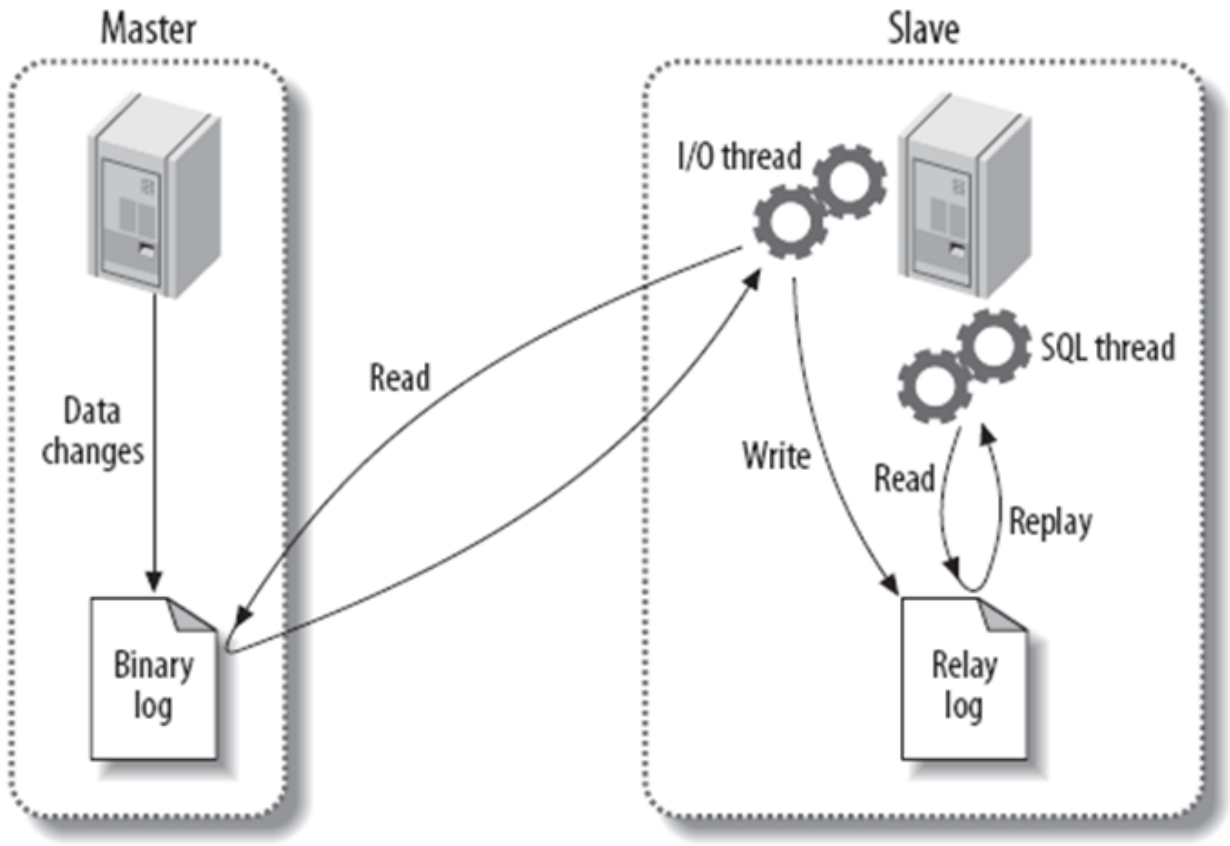
一、前言

随着应用业务数据不断的增大，应用的响应速度不断下降，在检测过程中我们不难发现大多数的请求都是查询操作。

此时，我们可以将数据库扩展成主从复制模式，将读操作和写操作分离开来，多台数据库分摊请求，从而减少单库的访问压力，进而应用得到优化。

> 本次测试使用两个虚拟机： ip：192.168.2.21（主） ip：192.168.2.22（从）

二、主从复制原理



同步操作通过 3 个线程实现，其基本步骤如下：

```
主服务器将数据的更新记录到二进制日志中（记录被称作二进制日志事件）-- 主库线程；
从库将主库的二进制日志复制到本地的中继日志（relay log）-- 从库 I/O 线程；
从库读取中继日志中的事件，将其重放到数据中 -- 从库 SQL 线程。
```

三、配置主库

3.1 创建用户

为了安全起见，准备创建一个新用户用于从库连接主库。

```
# 创建用户
create user 'repl'@'%' identified by 'repl';

# 授权，只授予复制和客户端访问权限
grant replication slave,replication client on *.* to 'repl'@'%' identified
by 'repl';
```

3.2 修改配置文件

1）vim /etc/my.cnf 在[mysqld]下添加：

```
log-bin         = mysql-bin
log-bin-index   = mysql-bin.index
binlog_format   = mixed
server-id       = 21
sync-binlog     = 1
character-set-server = utf8
```

2）保存文件并重启主库：

```
service mysqld restart
```

配置说明：

```
log-bin：设置二进制日志文件的基本名；
log-bin-index：设置二进制日志索引文件名；
binlog_format：控制二进制日志格式，进而控制了复制类型，三个可选值
    -STATEMENT：语句复制
    -ROW：行复制
    -MIXED：混和复制，默认选项
server-id：服务器设置唯一ID，默认为1，推荐取IP最后部分；
sync-binlog：默认为0，为保证不会丢失数据，需设置为1，用于强制每次提交事务时，同步二进制日志到磁盘上。
```

3.3 备份主数据库数据

若主从数据库都是刚刚装好且数据都是一致的，直接执行 **show master status** 查看日志坐标。

若主库可以停机，则直接拷贝所有数据库文件。

若主库是在线生产库，可采用 mysqldump 备份数据，因为它对所有存储引擎均可使用。

1）为了获取一个一致性的快照，需对所有表设置读锁：

```
flush tables with read lock;
```

2）获取二进制日志的坐标：

```
 show master status;
```

返回结果：

```
+------------------+----------+--------------+------------------+---------
----------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
Executed_Gtid_Set |
+------------------+----------+--------------+------------------+---------
----------+
| mysql-bin.000001 |      120 |              |                  |
|
+------------------+----------+--------------+------------------+---------
----------+
1 row in set (0.00 sec)
```

3）备份数据：

```
# 针对事务性引擎
mysqldump -uroot -ptiger --all-database -e --single-transaction --flush-
logs --max_allowed_packet=1048576 --net_buffer_length=16384 >
/data/all_db.sql

# 针对 MyISAM 引擎，或多引擎混合的数据库
mysqldump -uroot --all-database -e -l --flush-logs --
max_allowed_packet=1048576 --net_buffer_length=16384 > /data/all_db.sql
```

4）恢复主库的写操作：

```
unlock tables;
```

四、配置从库

4.1 修改配置文件

1）vim /etc/my.cnf 在[mysqld]下添加：

```
log-bin             = mysql-bin
binlog_format       = mixed
log-slave-updates   = 0
server-id           = 22
relay-log           = mysql-relay-bin
relay-log-index     = mysql-relay-bin.index
read-only           = 1
slave_net_timeout   = 10
```

2）保存文件并重启从库：

```
service mysqld restart
```

配置说明：

```
log-slave-updates：控制 slave 上的更新是否写入二进制日志，默认为0；若 slave 只作为从服务器，则不必启用；若 slave 作为其他服务器的 master，则需启用，启用时需和 log-bin、binlog-format 一起使用，这样 slave 从主库读取日志并重做，然后记录到自己的二进制日志中；
relay-log：设置中继日志文件基本名；
relay-log-index：设置中继日志索引文件名；
read-only：设置 slave 为只读，但具有super权限的用户仍然可写；
slave_net_timeout：设置网络超时时间，即多长时间测试一下主从是否连接，默认为3600秒，即1小时，这个值在生产环境过大，我们将其修改为10秒，即若主从中断10秒，则触发重新连接动作。
```

4.2 导入备份数据

如果 **3.3** 步骤中没进行备份，忽略此步骤。

```
mysql -uroot -p < /data/all_db.sql
```

4.3 统一二进制日志的坐标

根据 3.3 步骤获取的坐标，统一到从库中：

```
change master to
master_host='192.168.2.21',
master_user='repl',
master_password='repl',
master_port=3306,
master_log_file='mysql-bin.000001',
master_log_pos=120;
```

注意：此处使用的是新创建的账户。

4.4 启动主从复制

1）启动从库 slave 线程：

```
start slave;
```

2）查看从服务器复制功能状态：

```
show slave status\G;
```

返回结果：

```
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 192.168.2.21
                  Master_User: repl
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 120
               Relay_Log_File: mysql-relay-bin.000002
                Relay_Log_Pos: 283
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 120
```

此处只张贴部分返回结果。

结果说明：

```
Slave_IO_Running：此进程负责 slave 从 master 上读取 binlog 日志，并写入 slave 上的中继日志。
Slave_SQL_Running：此进程负责读取并执行中继日志中的 binlog 日志。
```

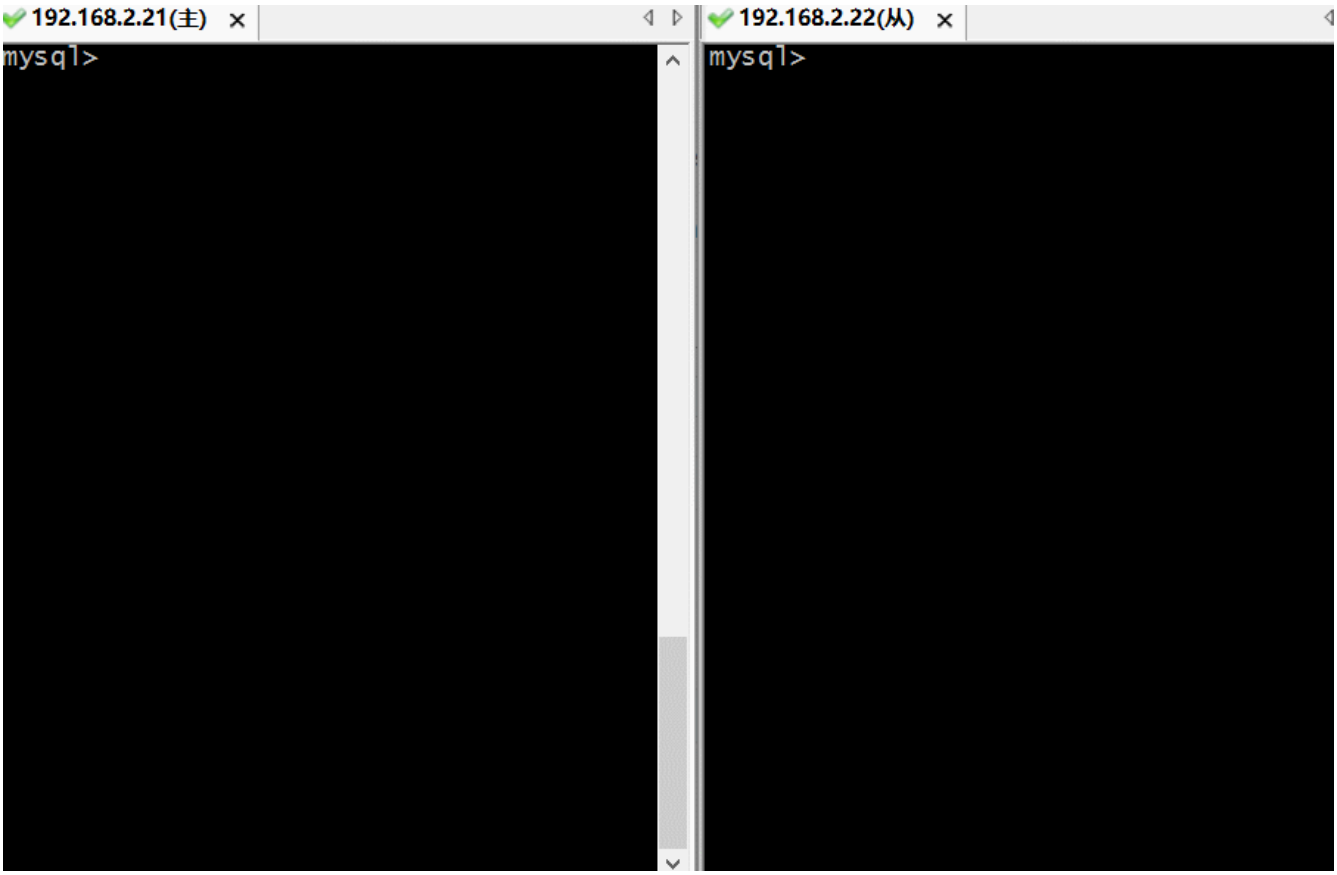
这两个进程的状态需全部为 YES，只要有一个为 NO，则复制就会停止。

当 Relay_Master_Log_File = Master_Log_File 且 Read_Master_Log_Pos = Exec_Master_Log_Pos 时，则表明 slave 和 master 处于完全同步的状态。

五、验证

使用一个简单的例子：

在主库创建名为 mysql_test 的数据库，如果同步成功，那么在从库中也能查询出名为 mysql_test 数据库。



六、参考资料

- MySQL 官网
- 从库提升为主库