目录

[第一章 mysql基础 2](#_Toc517305693)

[1、mysql安装操作 2](#_Toc517305694)

[1.1、rpm安装 3](#_Toc517305695)

[1.2、rpm安装修改默认目录 3](#_Toc517305696)

[1.3、编译安装 3](#_Toc517305697)

[1.4、mysql8.0以上版本安装 5](#_Toc517305698)

[2、mysql基本配置 6](#_Toc517305699)

[2.1、用户授权介绍 7](#_Toc517305700)

[2.2、回收权限 7](#_Toc517305701)

[2.3、忘记mysql密码操作 8](#_Toc517305702)

[3、my.cnf配置参考 8](#_Toc517305703)

[3.1、生产my.cnf参考 8](#_Toc517305704)

[3.2、my.cnf配置介绍 8](#_Toc517305705)

[4、mysql主从复制 8](#_Toc517305706)

[4.1、修改主服务器master 8](#_Toc517305707)

[4.2、修改从服务器slave 9](#_Toc517305708)

[4.3、配置主服务器 9](#_Toc517305709)

[4.4、配置从服务器 9](#_Toc517305710)

[4.5、检查复制状态 9](#_Toc517305711)

[第二章 基础sql语句 10](#_Toc517305712)

[1、SQL基础 10](#_Toc517305713)

[1.1、修改mysql密码 10](#_Toc517305714)

[1.2、找回丢失的密码 10](#_Toc517305715)

[1.3、创建用户和删除用户以及授权 10](#_Toc517305716)

[1.4、创建和删除表 11](#_Toc517305717)

[1.5、创建索引 11](#_Toc517305718)

[1.6、创建列及删除列 11](#_Toc517305719)

[1.7、select 表数据 12](#_Toc517305720)

[1.8、增删表字段 12](#_Toc517305721)

[第三章 mysql备份和恢复 12](#_Toc517305722)

[1、mysqldump备份与恢复 12](#_Toc517305723)

[1.1、备份数据库 12](#_Toc517305724)

[1.2、恢复数据库 13](#_Toc517305725)

[2、bin-log日志恢复数据 13](#_Toc517305726)

[3、xtrabackup备份与恢复(推荐) 13](#_Toc517305727)

[3.1、Xtrabackup优点 13](#_Toc517305728)

[3.2、应用安装 13](#_Toc517305729)

[3.3、全量备份 14](#_Toc517305730)

[3.4、增量备份 15](#_Toc517305731)

[3.5、全量还原 15](#_Toc517305732)

[3.6、增量还原 15](#_Toc517305733)

[4、mysql备份脚本整理 16](#_Toc517305734)

[第四章 mysql实现高可用 16](#_Toc517305735)

[1、keepalived+mysql实现高可用 16](#_Toc517305736)

[1.1、keepalived安装部署 16](#_Toc517305737)

[1.2、主keepalived配置 17](#_Toc517305738)

[1.3、备keepalived配置 17](#_Toc517305739)

[2、mysql+drdb+heartbeart实现高可用 20](#_Toc517305740)

[3、MySQL集群高可用架构之MHA 20](#_Toc517305741)

[第五章 mysql读写分离 20](#_Toc517305742)

[1、Mysql+Mycat实现数读写分离 20](#_Toc517305743)

[2、利用oneproxy实现读写分离 20](#_Toc517305744)

[第六章 mysql性能优化 20](#_Toc517305745)

[1、mysql问题排查与优化 20](#_Toc517305746)

[1.1、linux进程占用CPU高的解决方案 20](#_Toc517305747)

[1.2、慢查询排查与优化 20](#_Toc517305748)

[2、生产优化细节 20](#_Toc517305749)

[第七章 生产遇到问题汇总 21](#_Toc517305750)

[1、开户历史库归档整理 21](#_Toc517305751)

[第八章 mysql遇到错误分析 21](#_Toc517305752)

[问题1：mysql8.0以上版本caching\_sha2\_password' cannot be loaded: 21](#_Toc517305753)

[问题2: [ERROR] [MY-010811] [Server] Failed to open log (file './binlog.000061', errno 2) 22](#_Toc517305754)

[问题3：ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2) 22](#_Toc517305755)

[问题4：主从报错Fatal error: The slave I/O thread stops because master and slave have equal MySQL server UUIDs; these UUIDs must be different for replication to work. 22](#_Toc517305756)

[问题5：创建主从关系时copy了同样的my.cnf文件，报错 22](#_Toc517305757)

[Fatal error: The slave I/O thread stops because master and slave have equal MySQL server ids; 22](#_Toc517305758)

# 第一章 mysql基础

## 1、mysql安装操作

检查是否安装其他版本mysql

rpm -qa | grep -i mysql

卸载其他版本程序

yum -y remove mysql-libs-5.1.52-1.el6\_0.1.i686

rpm -e mysql-libs-5.1.61-4.el6.x86\_64 --nodeps

centos默认安装了mariadb如果安装mysql需要卸载mariadb

rpm -qa | grep -i mariadb #查看安装的mariadb

rpm -e mariadb-5.5.44-2.el7.centos.x86\_64 --nodeps 加上--nodeps表示强制卸载

### 1.1、rpm安装

rpm -ivh mysql-community-libs-5.7.11-1.el7.x86\_64.rpm mysql-community-libs-compat-5.7.11-1.el7.x86\_64.rpm mysql-community-client-5.7.11-1.el7.x86\_64.rpm mysql-community-server-5.7.11-1.el7.x86\_64.rpm

mysql密码第一次安装时系统会随机生成一个密码

grep 'temporary password' /var/log/mysqld.log

 [Note] A temporary password is generated for [root@localhost](mailto:root@localhost): u4&y%IsE:CdX

**登陆并设置密码**

mysql -u root -p

Enter password: 输入密码

5.7以上密码必须满足复杂性要求

SET PASSWORD = PASSWORD('Fan2017#$');

**关闭防火墙**

centos 7默认使用的是firewall防火墙

systemctl start firewalld.service #启动firewall

systemctl stop firewalld.service #停止firewall

systemctl disable firewalld.service #禁止firewall开机启动

改成iptables

systemctl start iptables.service

### 1.2、rpm安装修改默认目录

mysql数据库默认安装在/var/lib/mysql目录，如果想更改默认安装目录需要如下修改

1、修改my.cnf文件如下配置

datadir=/mysql/data/ 数据库存放路径

socket=/mysql/data/mysql.sock

2、迁移目录

mv /var/lib/mysql/\* /mysql/data/

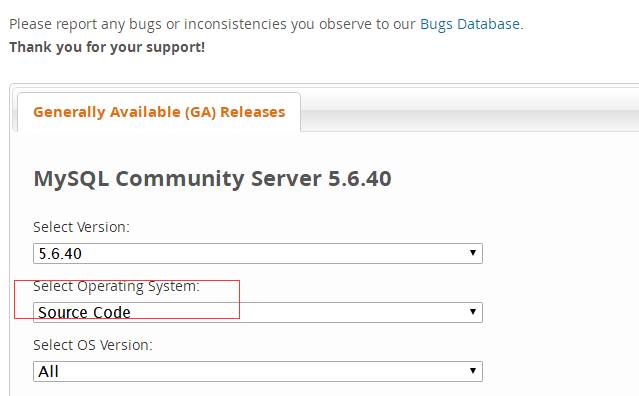
chown -R mysql:mysql /mysql/\*

3、然后可以重新启动了。如果启动报错需要执行 setenforce 0让系统关闭权限校验。

如果要彻底禁用SELinux 需要在/etc/sysconfig/selinux中设置参数selinux=0

4、修改/etc/init.d/mysql 启动文件里的默认路径都修改成现在的数据库存放路径（我自己验证没有找到可能是5.7的原因）

### 1.3、编译安装

1.2.1、官网下载mysql



1.2.2、安装依赖包

yum -y install make gcc-c++ cmake bison-devel ncurses-devel libaio

yum install libaio libaio-devel -y

yum install perl-Data-Dumper -y

yum install net-tools -y

tar -zxvf mysql-5.6.40.tar.gz

cd mysql-5.6.40

#创建mysql用户组

groupadd mysql

useradd -g mysql mysql

* + 1. 编译安装mysql

cmake \

-DCMAKE\_INSTALL\_PREFIX=/usr/local/mysql \

-DMYSQL\_DATADIR=/usr/local/mysql/data \

-DMYSQL\_UNIX\_ADDR=/usr/local/mysql/mysql.sock \

-DSYSCONFDIR=/etc \

-DWITH\_MYISAM\_STORAGE\_ENGINE=1 \

-DWITH\_INNOBASE\_STORAGE\_ENGINE=1 \

-DWITH\_MEMORY\_STORAGE\_ENGINE=1 \

-DWITH\_READLINE=1 \

-DMYSQL\_TCP\_PORT=3306 \

-DENABLED\_LOCAL\_INFILE=1 \

-DWITH\_PARTITION\_STORAGE\_ENGINE=1 \

-DEXTRA\_CHARSETS=all \

-DDEFAULT\_CHARSET=utf8 \

-DDEFAULT\_COLLATION=utf8\_general\_ci

make && make install

1.2.4、执行初始化配置脚本

cd /usr/local/mysq

chown -R mysql:mysql \*

scripts/mysql\_install\_db --basedir=/usr/local/mysql --datadir=/usr/local/mysql/data --user=mysql

1.2.5、拷贝启动脚本并启动

cp support-files/mysql.server /etc/init.d/mysql

chkconfig mysql on

service mysql start

1.2.6、登陆并配置密码

vi /etc/profile

#在末尾添加

PATH=/usr/local/mysql/bin:$PATH

export PATH

source /etc/profile

mysql -u root -p

SET PASSWORD = PASSWORD('fan');

远程登陆权限

GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY 'fan' WITH GRANT OPTION;

### 1.4、mysql8.0以上版本安装

**1、安装依赖关系**

yum -y install openssl-devel

yum install perl-JSON

**2、安装mysql(要一起安装)**

rpm -ivh mysql-community-common-8.0.11-1.el7.x86\_64.rpm

mysql-community-devel-8.0.11-1.el7.x86\_64.rpm

mysql-community-embedded-compat-8.0.11-1.el7.x86\_64.rpm

mysql-community-libs-8.0.11-1.el7.x86\_64.rpm

mysql-community-server-8.0.11-1.el7.x86\_64.rpm

mysql-community-client-8.0.11-1.el7.x86\_64.rpm

mysql-community-test-8.0.11-1.el7.x86\_64.rpm

**3、查看密码**

grep 'temporary password' /var/log/mysqld.log

**4、登陆后重新设置root密码**

ALTER USER 'root'@'localhost' IDENTIFIED BY 'Fan2018!';

**5、创建用户**

CREATE USER 'fangx'@'%' IDENTIFIED WITH mysql\_native\_password BY 'Fan2018!';

**#(需要注意：mysql8.0加密方式修改了)**

**#检查用户**

select user, host, plugin, authentication\_string from user\G;

**6、授权远程数据库**

*#授权所有权限*

GRANT ALL PRIVILEGES ON \*.\* TO 'fangx'@'%';

*#授权基本的查询修改权限，按需求设置*

GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,ALTER ON \*.\* TO 'fangx'@'%';

**7、查看用户权限**  
show grants for 'fangx'@'%';

## 2、mysql基本配置

创建用户

create user 'fangx'@'%' identified by 'Fan2018! ';

grant select,update,insert on koacif.\* to fangx@'%' identified  by '[Fan2018](mailto:Fangx@2017)!';

创建数据库

create database myfirstdb;

删除数据库

 drop database myfirstdb;

查看所有的数据库

 show databases;

选择数据库

use myfirstdb;

查看表

show tabales;

关闭数据库

mysqladmin -uroot -pfan shutdown

### 2.1、用户授权介绍

grant select,update,insert on koacif.\* to fangx@'%' identified  by '[Fan2018](mailto:Fangx@2017)!';

grant all privileges on \*.\* to 'koacif'@'%' identified by 'koacif' with grant option;

mysql权限介绍：<http://dev.mysql.com/doc/refman/5.7/en/privileges-provided.html>

all privileges : 表示将所有权限授给用户，也可以指定具体权限 如（select,create,drop,update,delel）；

on 表示这些权限对那些数据库和表生效 格式：数据库名.表名  \*.\* 表示所有库所有表

to 将权限授给哪个用户。格式：“登录用户名”@“登录IP”%表示所有地址都可以登录。"fangx"@"192.168.0.%"表示这个用户只能在192.168.0.0网段登录

identified by  指定用户登录的密码

with grant option  表示允许用户将自己的权限授权给其他用户

可以使用GRANT给用户添加权限，权限会自动叠加，不会覆盖之前授予的权限，比如你先给用户添加一个SELECT权限，后来又给用户添加了一个INSERT权限，那么该用户就同时拥有了SELECT和INSERT权限。

### 2.2、回收权限

show grants for 'fangx'@'%';       查看fangx 这个账户拥有的权限。

revoke update on koacif.\* from 'fangx'@'%';    回收update权限；

revoke all PRIVILEGES on `koacif`.\* from 'fangx'@'%'; 取消远程登陆权限

koacif.\* 这个库要和查到权限所定义数据库一样。

flush privileges;

操作完成命令需要重新加载权限

查看用户

mysql> use mysql   #选择数据库

mysql> select host,user from user;

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

| host      | user      |

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

| %         | koacif    |

| %         | root       |

| localhost | mysql.sys |

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

3 rows in set (0.00 sec)

删除用户

drop user 'koacif'@'%';

用户重命名

rename user 'test3'@'%' to 'test1'@'%';     把 test3 修改成test1

### 2.3、忘记mysql密码操作

vi /etc/my.cnf    #编辑文件，找到[mysqld]，在下面添加一行skip-grant-tables

然后重启MySQL服务

登陆mysql

mysql -uroot -p   #直接按回车，这时不需要输入root密码。

执行命令：

UPDATE mysql.user SET authentication\_string= PASSWORD('123456') WHERE user = 'root'; #5.7版本修改密码方式

**\*特别提醒注意的一点是，5.7以上的mysql数据库下的user表中已经没有Password字段了**

flush privileges;  #刷新系统授权表

vi /etc/my.cnf   编辑文件，找到[mysqld]，删除skip-grant-tables这一行

重启mysql后在登陆 mysql，root密码已经修改为123456了。

## 3、my.cnf配置参考

my.cnf生成网站：<http://imysql.cn/comment/1126>

### 3.1、生产my.cnf参考

我维护的生产参考，mysql版本是5.6.21



5.7版本的参考，从网站生成的



百度出来的上面有各个配置说明



### 3.2、my.cnf配置介绍

重要参数

## 4、mysql主从复制

主从复制要求mysql版本一致

### 4.1、修改主服务器master

   vi /etc/my.cnf

       [mysqld]

       log-bin=mysql-bin //[必须]启用二进制日志

       server-id=222  //[必须]服务器唯一ID，默认是1，一般取IP最后一段

### 4.2**、修改从服务器slave**

   vi /etc/my.cnf

       [mysqld]

       log-bin=mysql-bin //[不是必须]启用二进制日志

       server-id=226  //[必须]服务器唯一ID，默认是1，一般取IP最后一段

然后重启两台服务器的mysql

### 4.3、配置主服务器

mysql>GRANT REPLICATION SLAVE ON \*.\* to 'fangxslave'@'100.168.50.211' identified by 'fangxslave'; 一般不用root帐号，主从用户用具体IP代替，加强安全。

mysql>show master status; #查询日志节点

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

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

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

   | mysql-bin.000004 |  308 |       |         |

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

   1 row in set (0.00 sec)

   注：执行完此步骤后不要再操作主服务器MYSQL，防止主服务器状态值变化

### 4.4、配置从服务器

mysql>change master to master\_host='100.168.50.203',master\_user='fangxslave',master\_password='fangxslave',master\_log\_file=' mysql-bin.000004',master\_log\_pos=308;   注意不要断开，308数字前后无单引号。

Mysql>start slave;    启动从服务器复制功能

如果有报错为ERROR 1872 (HY000): Slave failed to initialize relay log info structure from the repository

使用reset slave可解决

### 4.5、检查复制状态

从服务器操作show slave status\G

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

              Slave\_IO\_State: Waiting for master to send event

              Master\_Host: 192.168.2.222  //主服务器地址

              Master\_User: fangxslave  //授权帐户名，尽量避免使用root

              Master\_Port: 3306    //数据库端口，部分版本没有此行

              Connect\_Retry: 60

              Master\_Log\_File: mysql-bin.000004

              Read\_Master\_Log\_Pos: 600     //#同步读取二进制日志的位置，大于等于Exec\_Master\_Log\_Pos

              Relay\_Log\_File: ddte-relay-bin.000003

              Relay\_Log\_Pos: 251

              Relay\_Master\_Log\_File: mysql-bin.000004

              Slave\_IO\_Running: Yes    //此状态必须YES

              Slave\_SQL\_Running: Yes     //此状态必须YES

                    ......

注：Slave\_IO及Slave\_SQL进程必须正常运行，即YES状态，否则都是错误的状态(如：其中一个NO均属错误)。

以上操作过程，主从服务器配置完成。

# 第二章 基础sql语句

字符集使用UTF8，创建数据库或者表不要指定字符集了，help (命令)，可以查看命令帮助

show variables like 'character\_set%';查询数据库字符编码

不乱码思想：环境选择UTF8

linux系统contos7设置source /etc/locale.conf 命令更新配置

cat /etc/locale.conf

LANG="zh\_CN.UTF-8"

contos6系统：/etc/sysconfig/i18n和my.cnf里的配置

## 1、SQL基础

文献参考：<https://www.cnblogs.com/benjamin77/p/8457148.html>

### 1.1、修改mysql密码

set password=password("fan")

mysqladmin -uroot -p password "fan"

UPDATE mysql.user SET password = PASSWORD("fan") WHERE user = 'root'

PS:5.7以上有复杂密码限制，不需要可以执行

set global validate\_password\_policy=0

ALTER USER USER() IDENTIFIED BY 'fan'

SHOW VARIABLES LIKE 'validate\_password%'

### 1.2、找回丢失的密码

在/etc/my.cnf添加skip-grant-tables然后重启

无密码登陆后修改密码

UPDATE mysql.user SET authentication\_string= PASSWORD("fan") WHERE user = 'root'

去掉skip-grant-tables然后重启就可以。

注：5.7版本password字段变成了authentication\_string

### 1.3、创建用户和删除用户以及授权

创建用户：create user 'koacif'@'%' identified by 'fan'

查询用户：select user,host,password from mysql.user;

删除用户：drop USER 'koacif'@'%';

授权：grant all privileges on \*.\* to koacif@'%' identified by 'fan';

PS:可以写具体权限(select,update,insert,delete,create,drop)

如：grant select,update,insert,delete on \*.\* to koacif@'%' identified by 'fan';

查看权限：show grants for 'koacif'@'%';

回收权限：revoke update on \*.\* from 'koacif'@'%';

### 1.4、创建和删除表

创建表：create table tutorials\_tbl(

tutorial\_id INT NOT NULL AUTO\_INCREMENT,

tutorial\_title VARCHAR(100) NOT NULL,

tutorial\_author VARCHAR(40) DEFAULT NULL,

submission\_date DATE COMMENT '录入时间',

PRIMARY KEY ( tutorial\_id )

);

字段使用NOT NULL属性，是因为我们不希望这个字段的值为NULL。 因此，如果用户将尝试创建具有NULL值的记录，那么MySQL会产生错误。

字段的AUTO\_INCREMENT属性告诉MySQL自动增加id字段下一个可用编号。

字段的后面加上COMMENT 可以填写备注信息

关键字PRIMARY KEY用于定义此列作为主键。可以使用逗号分隔多个列来定义主键。

删除表：DROP TABLE table\_name;

修改表名：alter table 表名 rename 新表名；

### 1.5、创建索引

用ALTER TABLE语句创建索引

普通索引：alter table table\_name add index index\_name (column\_list) ;

唯一索引：alter table table\_name add unique (column\_list) ;

主键索引：alter table table\_name add primary key (column\_list) ;

用CREATE INDEX语句对表增加索引，不能创建主键索引

普通索引：create index index\_name on table\_name (column\_list) ;

主键索引：create unique index index\_name on table\_name (column\_list) ;

删除普通索引：drop index index\_name on table\_name ;

删除普通索引：alter table table\_name drop index index\_name ;

删除主键索引：alter table table\_name drop primary key ;

查看索引：show keys from 表名

### 1.6、创建列及删除列

查看列：desc 表名

插入列：alter table 表名 add column 列名 varchar(20) COMMENT '备注信息';

删除列：alter table 表名 drop column 列名;

修改列名：alter table 表名 change 原列名 新列名 类型；

如：alter table hmm change tutorial\_title hmm\_title varchar(100);

修改列类型：alter table 表名 modify 列名 类型；

如：alter table hmm modify tutorial\_author varchar(100);

### 1.7、select 表数据

单表查询：select \* from 表名 where 条件

内连接：select \* from 表1 a, 表2 b, where a.a1 = b.b1

关联查询：select a.\*,b.\* from 表1 a left join 表2 b on a.a1 = b.b1

### 1.8、增删表字段

添加表数据：INSERT INTO table\_name (列1, 列2,...) VALUES (值1, 值2,....)

删除表数据：delete from 表名 where 条件

# 第三章 mysql备份和恢复

## 1、mysqldump备份与恢复

参考博客：https://www.cnblogs.com/clsn/archive/2017/12/31/8138015.html

### 1.1、备份数据库

mysqldump -uroot -pPassword [database name] > [dump file]

如：mysqldump --opt -uroot -p123456 oldboy > 2018.sql 只备份单个库

mysqldump --no-data --databases -B mydatabase1 mydatabase2 mydatabase3 > test.dump

--no-data 表示只备份表结构简写 -d

--databases 备份多个数据库，参数后面紧跟着数据库

--all-databases 备份所有数据库简写 -A

-B 表示在备份文件中增加创建库语句一般在备份多个库时使用

--master-data 增加binlog日志文件名及对应的位置点

--lock-all-tables 锁表 简写-x

-t 只备份数据

--single-transaction innodb 事务数据库备份用这个参数

**参数说明 参考：https://www.cnblogs.com/zengkefu/p/5690166.html**

备份建议：

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

innodb: mysqldump -uroot -p123456 --all-databases --flush-privileges --single-transaction --master-data=1 --flush-logs --triggers --routines --events --hex-blob > /opt/all.sql

mysqldump -A -R --triggers --master-data=2 --single-transaction |gzip >/opt/all\_$(date +%F).sql.gz

混合引擎：mysqldump -A -R --triggers --master-data=2 |gzip >/opt/all\_$(date +%F).sql.gz

  比如需要在每天凌晨1:30备份某个主机上的所有数据库并压缩dump文件为gz格式，那么可在/etc/crontab配置文件中加入下面代码行：

**30** **1** \* \* \* root mysqldump -u root -pPASSWORD --all-databases | gzip > /opt /database\_`date '+%m-%d-%Y'`.sql.gz

### 1.2、恢复数据库

mysql -uroot -p123456 [database name] < [backup file name] 如果恢复所有数据库可用不指定数据库

## 2、bin-log日志恢复数据

binlog日志用于记录所有更新数据，当我们的数据库出现故障时，我们可以利用binlog日志来挽回。

如果mysql数据库出现问题需要重新创建binlog二进制文件。

# 关闭当前的binlog日志并创建一个新日志文件，编号加1。

flush logs

# 查看日志，查出需要恢复的时间点

mysqlbinlog --no-defaults fangx-bin.000001 |more

#恢复具体时间导成SQL语句

mysqlbinlog fangx-bin.000001 --database=fangx --start-datetime='2018-06-6 00:00:01' --stop-datetime='2018-06-6 23:59:59' > fangx2018.sql

database #为要恢复的数据库名称

start-datetime #恢复的起始时间

stop-datetime #恢复的结束时间

# 把sql语句导入进去

mysql -uroot -pfan fangx -v -f < fangx2018.sql

-v 是查看导入的详细信息

-f 是当中间遇到错误时，可以skip过去，继续执行下面的语句

# 或者这样直接导入

mysqlbinlog --no-defaults fangx-bin.000001 --start-position="794" --stop-position="1055" | mysql -uroot -pfan fangx

#清除所有binlog日志 从1开始

reset master

## 3、xtrabackup备份与恢复(推荐)

Xtrabackup是由percona开源的免费数据库热备份软件，它能对InnoDB数据库非阻塞地备份

**1**）xtrabackup 是专门用来备份InnoDB表的，和mysql server没有交互；

**2**）innobackupex 是一个封装xtrabackup的Perl脚本，支持同时备份innodb和myisam，但在对myisam备份时需要加一个全局的读锁。

**3**）xbcrypt 加密解密备份工具

**4**）xbstream 流传打包传输工具，类似tar

**5**）物理备份工具，在同级数据量基础上，都要比逻辑备份性能好的多，特别是在数据量较大的时候，体现的更加明显。

### 3.1、Xtrabackup优点

　　1）备份速度快，物理备份可靠,2）备份过程不会打断正在执行的事务（无需锁表）,3）能够基于压缩等功能节约磁盘空间和流量,4）自动备份校验,5）还原速度快,6）可以流传将备份传输到另外一台机器上,7）在不增加服务器负载的情况备份数据,8）物理备份工具，在同级数据量基础上，都要比逻辑备份性能要好的多。几十G到不超过TB级别的条件下。但在同数据量级别，物理备份恢复数据上有一定优势。

### 3.2、应用安装

官网下载地址：https://www.percona.com/downloads/XtraBackup/LATEST/

rpm -ivh percona-xtrabackup-24-2.4.11-1.el7.x86\_64.rpm

# 如果提示缺失依赖包 yum -y install perl perl-devel libaio libaio-devel perl-Time-HiRes perl-DBD-MySQL perl-Digest-MD5

-------------------------

**错误提示安装依赖的库文件没有找到：libmysqlclient.so.18()(64bit)**

问题：使用yum安装软件依赖包时报错

...

2:postfix-2.10.1-6.el7.x86\_64 has missing requires of libmysqlclient.so.18()(64bit)

2:postfix-2.10.1-6.el7.x86\_64 has missing requires of libmysqlclient.so.18(libmysqlclient\_18)(64bit)

**重点关注：libmysqlclient.so.18()(64bit)**

解决:缺少Percona-XtraDB-Cluster-shared-55-5.5.37-25.10.756.el6.x86\_64.rpm这个包

# wget http://www.percona.com/redir/downloads/Percona-XtraDB-Cluster/5.5.37-25.10/RPM/rhel6/x86\_64/Percona-XtraDB-Cluster-shared-55-5.5.37-25.10.756.el6.x86\_64.rpm

# rpm -ivh Percona-XtraDB-Cluster-shared-55-5.5.37-25.10.756.el6.x86\_64.rpm

### 3.3、全量备份

innobackupex --defaults-file=/usr/local/mysql/my.cnf --socket=/usr/local/mysql/mysql.sock --user=root --password=fan --backup /opt/

#可以进入opt目录查看我们的备份文件

fangx opt]#ll

drwxr-x---. 7 root root 229 6月 13 04:30 2018-06-13\_04-30-40

#### 参数介绍

**--defaults-file=[MY.CNF]**  //指定配置文件：只能从给定的文件中读取默认选项。 且必须作为命令行上的第一个选项；必须是一个真实的文件，它不能是一个符号链接。

**--databases=#**  //指定备份的数据库和表，格式为：--database="db1[.tb1] db2[.tb2]" 多个库之间以空格隔开，如果此选项不被指定，将会备份所有的数据库。

--include=REGEXP    //用正则表达式的方式指定要备份的数据库和表，格式为 --include=‘^mydb[.]mytb’ ，对每个库中的每个表逐一匹配，因此会创建所有的库，不过是空的目录。--include 传递给 xtrabackup --tables。

--tables-file=FILE    //此选项的参数需要是一个文件名，此文件中每行包含一个要备份的表的完整名称，格式为databasename.tablename。该选项传递给 xtrabackup --tables-file，与--tables选项不同，只有要备份的表的库才会被创建。

注意：部分备份（--include、--tables-file、--database）需要开启 innodb\_file\_per\_table 。

--compact    //创建紧凑型备份，忽略所有辅助索引页，只备份data page；通过--apply-log中重建索引--rebuild-indexs。

--compress    //此选项指示xtrabackup压缩备份的InnoDB数据文件，会生成 \*.qp 文件。

--decompress    //解压缩qp文件，为了解压缩，必须安装 qpress 工具。 Percona XtraBackup不会自动删除压缩文件，为了清理备份目录，用户应手动删除 \* .qp文件：find /data/backup -name "\*.qp" | xargs rm。

--no-timestamp    //指定了这个选项备份将会直接存储在 BACKUP-DIR 目录，不再创建时间戳文件夹。

**--apply-log**    //应用 BACKUP-DIR 中的 xtrabackup\_logfile 事务日志文件。一般情况下，在备份完成后，数据尚且不能用于恢复操作，因为备份的数据中可能会包含尚未提交的事务或已经提交但尚未同步至数据文件中的事务。因此，此时数据文件仍处于不一致状态。“准备”的主要作用正是通过回滚未提交的事务及同步已经提交的事务至数据文件使得数据文件处于一致性状态。

--use-memory=#    //此选项接受一个字符参数（1M/1MB,1G/1GB，默认100M），仅与--apply-log一起使用，该选项指定prepare时用于崩溃恢复（crash-recovery）的内存。

**--copy-back**    //拷贝先前备份所有文件到它们的原始路径。但原路径下不能有任何文件或目录，除非指定 --force-non-empty-directories 选项。

--force-non-empty-directories    //恢复时指定此选项，可使 --copy-back 和 --move-back 复制文件到非空目录，即原data目录下可以有其他文件，但是不能有与恢复文件中同名的文件，否则恢复失败。

--rsync    //此选项可优化本地文件（非InnoDB）的传输。rsync工具一次性拷贝所有非InnoDB文件，而不是为每个文件单独创建cp，在备份恢复很多数据库和表时非常高效。此选项不能和 --stream 一起使用。

**--incremental**    //这个选项告诉 xtrabackup 创建一个增量备份，而不是完全备份。它传递到 xtrabackup 子进程。当指定这个选项，可以设置 --incremental-lsn 或 --incremental-basedir。如果这2个选项都没有被指定，--incremental-basedir 传递给 xtrabackup 默认值，默认值为：基础备份目录的第一个时间戳备份目录。

--incremental-basedir=DIRECTORY    //该选项接受一个字符串参数，该参数指定作为增量备份的基本数据集的完整备份目录。它与 --incremental 一起使用。

--incremental-dir=DIRECTORY    //该选项接受一个字符串参数，该参数指定了增量备份将与完整备份相结合的目录，以便进行新的完整备份。它与 --incremental 选项一起使用。

--redo-only    //在“准备基本完整备份” 和 “合并所有的增量备份(除了最后一个增备)”时使用此选项。它直接传递给xtrabackup的 xtrabackup --apply-log-only 选项，使xtrabackup跳过"undo"阶段，只做"redo"操作。如果后面还有增量备份应用到这个全备,这是必要的。有关详细信息,请参阅xtrabackup文档。

--parallel=NUMBER-OF-THREADS    //此选项接受一个整数参数，指定xtrabackup子进程应用于同时备份文件的线程数。请注意，此选项仅适用于文件级别，也就是说，如果您有多个.ibd文件，则它们将被并行复制； 如果您的表一起存储在一个表空间文件中，它将不起作用。

**2) xtrabackup 参数选项**

--apply-log-only    //这个选项使在准备备份(prepare)时，只执行重做(redo)阶段，这对于增量备份非常重要。

--backup:指明为备份，此参数可以忽略

### 3.4、增量备份

innobackupex --defaults-file=/usr/local/mysql/my.cnf --socket=/usr/local/mysql/mysql.sock --user=root --password=fan --incremental /backup/ --incremental-basedir=/opt/2018-06-13\_04-30-40/

#--incremental /backup/   指定增量备份文件备份的目录

#--incremental-basedir    指定上一次全备或增量备份的目录

### 3.5、全量还原

innobackupex --apply-log /opt/2018-06-13\_04-30-40/ #**使用此参数使用相关数据性文件保持一致性状态**

**#恢复数据之前需要保证数据目录是空的状态**

innobackupex --datadir=/usr/local/mysql/data --copy-back /opt/2018-06-13\_04-30-40/

#恢复之后需要重新设置下属组,然后启动mysql

chown -R mysql:mysql \*

### 3.6、增量还原

innobackupex --apply-log --redo-only /opt/2018-06-13\_04-30-40/ #首先执行全量备份的文件

innobackupex --apply-log --redo-only /opt/2018-06-13\_04-30-40/ --incremental-dir=/opt/zl/2018-06-13\_05-07-52/ #增量文件的校验，有多少个执行多少次按照备份顺序执行

innobackupex --datadir=/usr/local/mysql/data --copy-back /opt/2018-06-13\_04-30-40/ #然后开始还原

#恢复之后需要重新设置下属组,然后启动mysql

chown -R mysql:mysql \*

文献参考：https://www.cnblogs.com/youkanyouxiao/p/8335173.html

## 4、mysql备份脚本整理

以下脚本已验证



# 第四章 mysql实现高可用

## 1、keepalived+mysql实现高可用

环境



应用

keepalived-2.0.2.tar.gz

openssl-devel-1.0.2k-12.el7.x86\_64.rpm

### 1.1、keepalived安装部署

keepalived下载网址：http://www.keepalived.org/

**准备应用**

keepalived-2.0.2.tar.gz

openssl-devel-1.0.2k-12.el7.x86\_64.rpm

**安装**

rpm -ivh openssl-devel-1.0.2k-12.el7.x86\_64.rpm

tar -zxvf keepalived-2.0.2.tar.gz

./configure --prefix=/usr/local/keepalived

make && make install

**整理文件**

cp /usr/local/keepalived/sbin/keepalived /usr/sbin/

cp /usr/local/keepalived/etc/sysconfig/keepalived /etc/sysconfig/

mkdir -p /etc/keepalived

cd /etc/keepalived

### 1.2、主keepalived配置

vim /etc/keepalived.conf

------

global\_defs {

notification\_email {

}

}

vrrp\_instance VI\_1 {

state MASTER #备用服务器上改为 BACKUP

interface ens33

virtual\_router\_id 11

priority 110 #备用服务器上改为99

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

192.168.10.11

}

}

virtual\_server 192.168.10.132 3306 {

delay\_loop 6 #每隔6秒检查一次real\_server状态

lb\_algo rr

lb\_kind NAT

nat\_mask 255.255.255.0

persistence\_timeout 50 #回话保持时间

protocol TCP

real\_server 192.168.10.132 3306 {

weight 2

notify\_down "/etc/keepalived/mysql.sh" #检测mysql脚本,内容:pkill keepalived

TCP\_CHECK {

connect\_timeout 3

connect\_port 3306

}

}

}

-------

### 1.3、备keepalived配置

vim keepalived.conf

-----------------

global\_defs {

notification\_email {

}

}

vrrp\_instance VI\_1 {

state BACKUP #备用服务器上改为 BACKUP

interface ens33

virtual\_router\_id 11

priority 99 #备用服务器上改为99

advert\_int 1

authentication {

auth\_type PASS

auth\_pass 1111

}

virtual\_ipaddress {

192.168.10.11

}

}

virtual\_server 192.168.10.200 3306 {

delay\_loop 6 #每隔6秒检查一次real\_server状态

lb\_algo rr

lb\_kind NAT

nat\_mask 255.255.255.0

persistence\_timeout 50 #回话保持时间

protocol TCP

real\_server 192.168.10.200 3306 {

weight 1

notify\_down "/etc/keepalived/mysql.sh" #检测mysql脚本,内容:pkill keepalived

TCP\_CHECK {

connect\_timeout 3

connect\_port 3306

}

}

}

-------

**创建关闭keepalived脚本**

vim mysql.sh

------

#!/bin/bash

pkill keepalived

-------

启动相关程序

service keepalived start

添加开机自启动

chkconfig --add keepalived

chkconfig keepalived on

至此keepalived配置完成，可以通过关闭mysql程序验证VIP是否漂移，关闭主keepalived验证VIP是否漂移，关闭主keepalived机器验证keepalived是否漂移

**关于keepalived算法说明**

**rr** **轮询算法**，它将请求依次分配给不同的rs节点，也就是RS节点中均摊分配。这种算法简单，但只适合于RS节点处理性能差不多的情况  
**wrr** **加权轮训调度**，它将依据不同RS的权值分配任务。权值较高的RS将优先获得任务，并且分配到的连接数将比权值低的RS更多。相同权值的RS得到相同数目的连接数。  
**Wlc 加权最小连接数调度**，假设各台RS的全职依次为Wi，当前tcp连接数依次为Ti，依次去Ti/Wi为最小的RS作为下一个分配的RS  
**Dh 目的地址哈希调度**（destination hashing）以目的地址为关键字查找一个静态hash表来获得需要的RS  
**SH 源地址哈希调度**（source hashing）以源地址为关键字查找一个静态hash表来获得需要的RS  
**Lc 最小连接数调度**（least-connection）,IPVS表存储了所有活动的连接。LB会比较将连接请求发送到当前连接最少的RS.  
**Lblc 基于地址的最小连接数调度**（locality-based least-connection）：将来自同一个目的地址的请求分配给同一台RS，此时这台服务器是尚未满负荷的。否则就将这个请求分配给连接数最小的RS，并以它作为下一次分配的首先考虑。

LVS调度算法的生产环境选型:

1、一般的网络服务，如http，mail，mysql等常用的LVS调度算法为：

a.基本轮询调度rr

b.加权最小连接调度wlc

c.加权轮询调度wrc

2、基于局部性的最小连接lblc和带复制的给予局部性最小连接lblcr主要适用于web **cache**和DB **cache**

3、源地址散列调度SH和目标地址散列调度DH可以结合使用在防火墙集群中，可以保证整个系统的出入口唯一。

实际适用中这些算法的适用范围很多，工作中最好参考内核中的连接调度算法的实现原理，然后根据具体的业务需求合理的选型。

**负载均衡转发工作模式**  
1、**DR模式**：  
通过改写请求报文的目标MAC地址，将请求发送给真实服务器，而真实服务器将相应后的处理结果直接返还给客户端用户。极大地提高集群的伸缩性，但LB与RS必须在同一局域网环境。  
2、**NAT模式：**  
通过网络地址转换，LB重写请求报文的目标地址，根据预设的调度算法，将请求分派给后端的真实服务器，真实服务器的响应报文处理之后，返回时必须通过LB，经过LB时报文的源地址被重写，再返回给客户。  
3、**隧道模式：**  
LB把请求的报文通过IP隧道转发至真实服务器，而真实服务器将响应处理后直接返回给客户端用户。  
4、**FULLNAT模式：**  
数据包进入时，除了做DNAT，还做SNAT，从而实现LVS-RealServer间可以跨vlan通讯，RealServer只需要连接到内网。

## 2、mysql+drdb+heartbeart实现高可用

## 3、MySQL集群高可用架构之MHA

## 4、MySQL Utilities管理工具

参考：http://www.ttlsa.com/mysql/mysql-manager-tools-mysql-utilities-tutorial/

# 第五章 mysql读写分离

## 1、读写分离方法

使用proxySQL+MySQL实现读写分离，参考链接：<https://mp.weixin.qq.com/s/7PU0PuZLbIx_WQCUd4hs_g>

使用mycat+mysql实现读写分离，参考：<https://mp.weixin.qq.com/s/m2xP-BSIZaFInG4eEtWD6g>

## 2、利用oneproxy实现读写分离



# 第六章 mysql性能优化

## 1、mysql问题排查与优化

### 1.1、linux进程占用CPU高的解决方案

uptime linux系统命令，查看机器负载

show full processlist 查看进程

### 1.2、慢查询排查与优化

## 2、生产优化细节

mysql优化点：

1、硬件优化：CPU:高性能，硬盘:ssd,raid0>raid10>raid5>raid1，内存:大内存

2、my.cnf参数优化:参考生产my.cnf

3、SQL语句优化:创建索引,SQL语句避免like，复杂的SQL拆分多个小SQL，避免在数据库中计算数据。

4、架构优化:

# 第七章 生产遇到问题汇总

## 1、开户历史库归档整理



# 第八章 mysql遇到错误分析

### 问题1：mysql8.0以上版本caching\_sha2\_password' cannot be loaded:

**/usr/lib64/mysql/plugin/caching\_sha2\_password.so**

61.17. Authentication plugin 'caching\_sha2\_password' cannot be loaded: /usr/lib64/mysql/plugin/caching\_sha2\_password.so: cannot open shared object file: No such fileor directory

这个故障出现在 MySQL 8.0 上，用户使用 mysql client 5.7 链接 MySQL 8.0 提示如下

[root@netkiller ~]# mysql -h 193.112.95.53 -uroot -p

Enter password:

ERROR 2059 (HY000): Authentication plugin 'caching\_sha2\_password' cannot be loaded: /usr/lib64/mysql/plugin/caching\_sha2\_password.so: cannot open shared object file: No such fileor directory

解决方案，创建用户使用 mysql\_native\_password密码

mysql> CREATE USER 'root'@'%' IDENTIFIED WITH mysql\_native\_password BY 'pMQiEge1ikst7S\_6tlXzBOmt\_4b';

Query OK, 0 rows affected (0.08 sec)

mysql> grant all on \*.\* to 'root'@'%';

Query OK, 0 rows affected (0.08 sec)

重新链接

[root@netkiller ~]# mysql -h 193.112.95.53 -uneo -p

Enter password:

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

Your MySQL connection id is 24

Server version: 8.0.11 MySQL Community Server - GPL

Copyright (c) 2000, 2016, 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>

### 问题2: [ERROR] [MY-010811] [Server] Failed to open log (file './binlog.000061', errno 2)

mysqld: File './binlog.000061' not found (OS errno 2 - No such file or directory)

2018-06-13T03:42:52.970224Z 0 [ERROR] [MY-010811] [Server] Failed to open log (file './binlog.000061', errno 2)

2018-06-13T03:42:52.970238Z 0 [ERROR] [MY-010958] [Server] Could not open log file.

2018-06-13T03:42:52.970246Z 0 [ERROR] [MY-010041] [Server] Can't init tc log

2018-06-13T03:42:52.970268Z 0 [ERROR] [MY-010119] [Server] Aborting

2018-06-13T03:42:54.849493Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: Shutdown complete (mysqld 8.0.11) MySQL Community Server - GPL.

**解决办法：愿意是因为无法binlog日志被删除，需要查看my.cnf关闭binlog日志,然后重新生成binlog日志**

### 问题3：ERROR 2002 (HY000): Can't connect to local MySQL server through socket '/var/lib/mysql/mysql.sock' (2)

解决办法：创建一个连接。

ln -s /data/mysql/mysql.sock /var/lib/mysql/mysql.sock

### 问题4：主从报错Fatal error: The slave I/O thread stops because master and slave have equal MySQL server UUIDs; these UUIDs must be different for replication to work.

**原因分析：**

mysql 5.6的复制引入了uuid的概念，各个复制结构中的server\_uuid得保证不一样，但是查看到直接copy  data文件夹后server\_uuid是相同的，show variables like '%server\_uuid%';

**解决方法：**

找到data文件夹下的auto.cnf文件，修改里面的uuid值，保证各个db的uuid不一样，重启db即可

### 问题5：创建主从关系时copy了同样的my.cnf文件，报错Fatal error: The slave I/O thread stops because master and slave have equal MySQL server ids;

**原因分析：**

和server\_uuid类似，servier\_id也得保证不一样

**解决方法：**

找到my.cnf配置文件中的server\_id，修改从库的server\_id保证和复制结构中的其他db不一样，重启db即可

### 问题6：[mysql5.7.1.16出现[Err] 1146 - Table 'performance\_schema.session\_status' doesn't exist的解决办法](http://blog.csdn.net/qq_22642239/article/details/59513956)

执行mysql\_upgrade -u root -p --force命令然后输入password，然后重新启动后，问题解决。

### 问题7：[解决mysql主从不同步](http://www.cnblogs.com/blueel/archive/2013/01/06/2847842.html)

更多主从不同步问题参考：http://blog.51cto.com/13407306/2067333

mysql>show processlist;   查看下进程是否Sleep太多。发现很正常。

show master status; 也正常。

mysql> show master status;

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

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

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

| mysqld-bin.000001 |     3260 |              | mysql,test,information\_schema |

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

1 row in set (0.00 sec)

再到Slave上查看

mysql> show slave status;

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: No

可见是Slave不同步

方法一：忽略错误后，继续同步

该方法适用于主从库数据相差不大，或者要求数据可以不完全统一的情况，数据要求不严格的情况

解决：

stop slave; #表示跳过一步错误，后面的数字可变

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

start slave;

之后再用mysql> show slave status  查看

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

ok，现在主从同步状态正常了。。。

方式二：重新做主从，完全同步

该方法适用于主从库数据相差较大，或者要求数据完全统一的情况

解决步骤如下：

1.先进入主库，进行锁表，防止数据写入

使用命令：mysql> flush tables with read lock;

注意：该处是锁定为只读状态，语句不区分大小写

2.进行数据备份

#把数据备份到mysql.bak.sql文件

[root@server01 mysql]#mysqldump -uroot -p -hlocalhost > mysql.bak.sql

这里注意一点：数据库备份一定要定期进行，可以用shell脚本或者python脚本，都比较方便，确保数据万无一失

3.查看master 状态

mysql> show master status;

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

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

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

| mysqld-bin.000001 |     3260 |              | mysql,test,information\_schema |

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

1 row in set (0.00 sec)

4.把mysql备份文件传到从库机器，进行数据恢复

#使用scp命令

[root@server01 mysql]# scp mysql.bak.sql root@192.168.128.101:/tmp/

5.停止从库的状态

mysql> stop slave;

6.然后到从库执行mysql命令，导入数据备份

mysql> source /tmp/mysql.bak.sql

7.设置从库同步，注意该处的同步点，就是主库show master status信息里的| File| Position两项

change master to master\_host = '192.168.128.100', master\_user = 'rsync', master\_port=3306, master\_password='', master\_log\_file = 'mysqld-bin.000001', master\_log\_pos=3260;

8.重新开启从同步

mysql> start slave;

9.查看同步状态

mysql> show slave status;  查看：

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes

好了，同步完成啦。