# Mysql

## 授权远程访问,远程登录

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

flush privileges;

/usr/bin/mysqladmin -u root password 'sjcsfwq'

4、如果此时还无法连接，检测配置文件（/etc/mysql/mysql.conf.d/mysqld.cnf） bind-address配置是否开启，应该是关闭

#bind-address = 127.0.0.1 注释掉；

## 数据目录

/var/lib/mysql

## 恢复

把上述文件夹下的对应数据库拷过去就可以用了

如果出了错误:

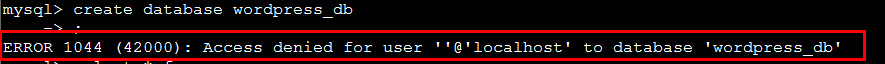
# mysql ERROR 1018 (HY000): Can't read dir of '.xxxx' (errno: 13)解决方法

mysql问题：   
mysql> use asterisk;   
Database changed   
mysql> show tables;   
ERROR 1018 (HY000): Can't read dir of './asterisk/' (errno: 13)   
  
原因：

查看这个数据库的目录属性：（#cd /usr/local/mysql/var; #ll -t ）  
这个库目录的属主不对，使用chown修改之，表信息都正常了。   
此时可以#ls -l 或 #ll  查看该目录的详细信息，就可发现库目录的属主不是mysql   
  
解决方法：   
#chown -R mysql:mysql /var/lib/mysql/SJCS-GF  
其中 /var/lib/mysql/SJCS-GF  
 为mysql数据库存储目录

## [ERROR 1044 (42000): Access denied for user ''@'localhost' to database 'mysql'](http://www.cnblogs.com/Anker/p/3551610.html)

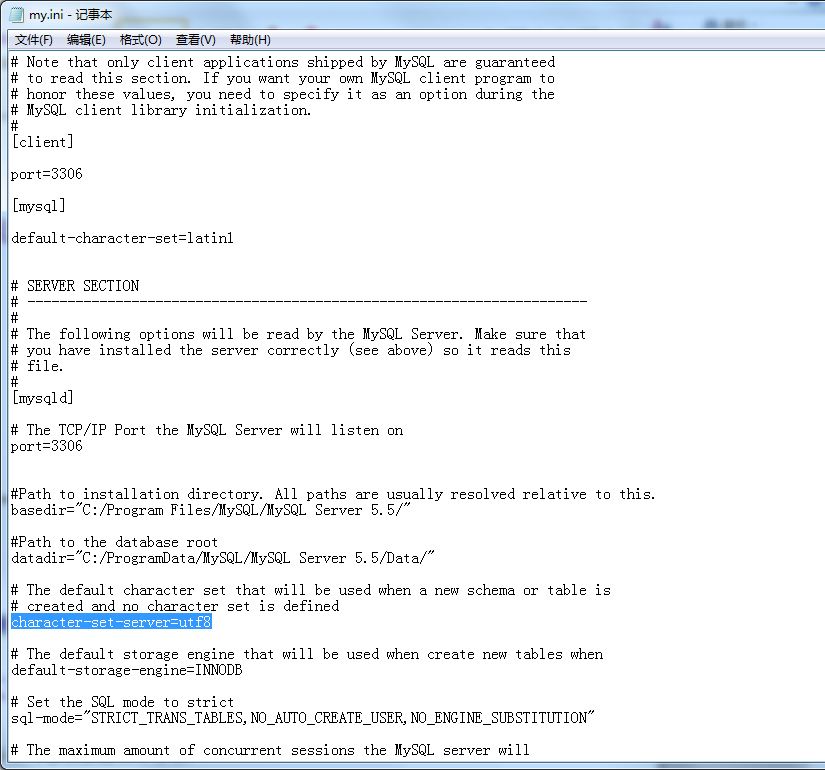
　　在centos下安装好了mysql，用root帐号连上mysql，然后创建一个数据库，提示下图错误：



　　提示：ERROR 1044 (42000): Access denied for user ''@'localhost' to database 'mysql'。网上找了一个比较流行的方法（见方法一），搞定了。今天又用这个试了试，却搞不定，在网上找了半天，终于发现是因为mysql数据库的user表里，存在用户名为空的账户即匿名账户，导致登录的时候是虽然用的是root，但实际是匿名登录的，通过错误提示里的''@'localhost'可以看出来，于是解决办法见方法二。  
  
**方法一：**  
**1.关闭mysql**  
   # service mysqld stop  
**2.屏蔽权限**  
   # mysqld\_safe --skip-grant-table  
   屏幕出现： Starting demo from .....  
**3.新开起一个终端输入**  
   # mysql -u root mysql  
   mysql> UPDATE user SET Password=PASSWORD('newpassword') where USER='root';  
   mysql> FLUSH PRIVILEGES;//记得要这句话，否则如果关闭先前的终端，又会出现原来的错误  
   mysql> \q  
  
方法二：  
**1.关闭mysql**  
   # service mysqld stop  
**2.屏蔽权限**  
   # mysqld\_safe --skip-grant-table  
   屏幕出现： Starting demo from .....  
**3.新开起一个终端输入**  
   # mysql -u root mysql  
   mysql> delete from user where USER='';  
   mysql> FLUSH PRIVILEGES;//记得要这句话，否则如果关闭先前的终端，又会出现原来的错误  
   mysql> \q

## navicat显示乱码

### windows



### java

改配置文件字符集改成utf8,navicat不要改,不然导数据容易出问题

### C++

**去掉“使用mysql字符集”前的对勾，在编码里选择utf-8，**

## rpm安装方式更改data目录

rpm方式安装的MySQL服务如何修改数据文件目录

**mkdir -p /data/mysql   
修改mysql用户家目录为/data/mysql   
mysql:x:27:27:MySQL Server:/data/mysql:/bin/bash   
chown mysql.mysql /data/mysql/   
vi /etc/my.cnf   
[mysqld]   
#datadir=/var/lib/mysql   
datadir=/home/yigou/www/htmlBack/log\_mysql/mysql  
mysql\_install\_db --datadir=/home/yigou/www/htmlBack/log\_mysql/mysql --user=mysql   
/usr/bin/mysqld\_safe &**

**/usr/bin/mysqld\_safe --datadir=/home/yigou/www/htmlBack/log\_mysql/mysql --pid-file=/var/lib/mysql/test.pid &**

## mysql整合版(一个文件夹那种)

### mysql5.6.26-glibc tar.gz安装

解压

解压完之后 将目录改名为mysql

chown -R yigou:yigou mysql

su yigou

cd mysql

./scripts/mysql\_install\_db --basedir=. --datadir=./data

vim my.cnf

[mysqld]

lower\_case\_table\_names=1

explicit\_defaults\_for\_timestamp=true

innodb\_buffer\_pool\_size = 128M

log\_bin = yigou.bin

basedir =.

datadir =./data

port = 3306

server\_id = 1

join\_buffer\_size = 128M

sort\_buffer\_size = 2M

read\_rnd\_buffer\_size = 2M

sql\_mode=NO\_ENGINE\_SUBSTITUTION,STRICT\_TRANS\_TABLES

max\_allowed\_packet = 50M

log-error=./error.log

1. ./bin/mysqld --defaults-file=my.cnf &
2. ./bin/mysqladmin -uroot password 'sjcsfwq'

这样配置之后，应该就可以移动文件夹到任意位置重新启动了。

不要设置服务啥的，如果非要设置，得用绝对路径，但我觉得这样就行了

### 启动服务

./mysqld --defaults-file=../my.cnf

### 登陆

./mysql -h10.104.24.221 -uroot -pnh6QAew1Wr82d

## 开放防火墙端口

**第二部分，在linux防火墙上开放3306端口（mysql远程访问端口）**

**1、打开防火墙配置文件  
vi  /etc/sysconfig/iptables Linux学习，http:// linux.it.net.cn**

**2、增加下面一行**

**-A INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT**

**3、重启防火墙**

**service  iptables restart**

放到icmp上边

## 授权远程访问,远程登录

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

flush privileges;

备份数据

mysqldump -uroot -pnh6QAew1Wr82d yigou >yigou.sql

4、如果此时还无法连接，检测配置文件（/etc/mysql/mysql.conf.d/mysqld.cnf） bind-address配置是否开启，应该是关闭

#bind-address = 127.0.0.1 注释掉；

## 导入数据

./mysql -uroot -pnh6QAew1Wr82d yigou </root/yigou.sql

mysql  -h10.104.24.221 -uroot -p --force dbname <   ./test\_gznow\_bbs.sql

./mysql -h10.104.24.221 -uroot -pnh6QAew1Wr82d yigou </root/yigou.sql

### 远程导入到数据库

内网:

mysql -hrds5mv15x7abta401daj.mysql.rds.aliyuncs.com -uyigou\_1 -pyigou\_1 yigou < /home/yigou/xxx.sql

外网:

mysql -h rds5mv15x7abta401dajo.mysql.rds.aliyuncs.com -uyigou\_1 -pyigou\_1 yigou < /home/yigou/www/htmlBack/yigou.sql

注释:aspchina\_net.sql,如果用户没有创建数据库的权限将不能导入……..是吗,怎么测试成功了……aspchina\_net.sql数据库,否则spchina\_net.sql只能是多张表不然会出错,这点切记!..

## 强制,跳过错误

mysqlbinlog /root/Desktop/bk/log-bin.000005 | mysql -uroot -psjcsfwq –force 跳过错误

## 备份命令

### 本地sql备份

mysqldump -u root -pnh6QAew1Wr82d yigou > /home/yigou/yigou.sql

### 从远程备份到本地

假设我们的任务是要备份一个位于192.168.1.2服务器上，名称为backup\_test的数据库，并把备份文件放到D:\bak\的文件夹中。又假设我们的[操作系统](http://www.myhack58.com/Article/48/Article_048_1.htm)为windows xp，于是我们可以用以下的方法执行备份操作：

我们可以使用windows自带的命令提示符（调用命令提示符的方法有多种，最常用的就是点击开始菜单，选择“运行”，输入cmd的命令就可以启动了），在命令提示符中输入以下命令：

mysqldump -**uyigou\_1** -pyigou\_1 -hrds5mv15x7abta401daj.mysql.rds.aliyuncs.com yigou > /home/yigou/yigou.sql

5.6为防错误加参数

mysqldump -hrds5mv15x7abta401daj.mysql.rds.aliyuncs.com -uyigou\_1 -pyigou\_1 yigou --set-gtid-purged=OFF> /home/yigou/yigou.sql

mysqlbinlog --no-defaults --database=yigou --stop-datetime='2015-10-21 17:00:00' /root/Desktop/bk/log-bin.000005 >x.txt

mysql 的日志的启动与查看  
错误日志： -log-err  
查询日志： -log  
慢查询日志: -log-slow-queries  
更新日志: -log-update  
二进制日志： -log-bin  
默 认情况下，所有日志创建于mysqld数据目录中。通过刷新日志，你可以强制 mysqld来关闭和重新打开日志文件（或者在某些情况下切换到一个新的日志）。当你执行一个FLUSH LOGS语句或执行mysqladmin flush-logs或mysqladmin refresh时，出现日志刷新  
1. 错误日志  
用--log- error[=file\_name]选项来指定mysqld保存错误日志文件的位置。如果没有给定file\_name值，mysqld使用错误日志名 host\_name.err 并在数据目录中写入日志文件。如果你执行FLUSH LOGS，错误日志用-old重新命名后缀并且mysqld创建一个新的空日志文件。(如果未给出--log-error选项，则不会重新命名）。

如果不指定--log-error，或者(在Windows中)如果你使用--console选项，错误被写入标准错误输出stderr。通常标准输出为你的终端。

2. 通用查询日志

用--log[=file\_name]或-l [file\_name]选项启动它。如果没有给定file\_name的值，默认名是host\_name.log。

3. 慢速查询日志

用--log-slow-queries[=file\_name]选项启动时，mysqld 写一个包含所有执行时间超过long\_query\_time秒的SQL语句的日志文件.如果没有给出file\_name值，默认未主机名，后缀为 -slow.log。如果给出了文件名，但不是绝对路径名，文件则写入数据目录。

3. 更新日志

用--log-update[=file\_name]选项启动,不推荐使用.

是否启用了日志  
mysql>show variables like 'log\_%';  
怎样知道当前的日志  
mysql> show master status;  
顯示二進制日志數目  
mysql> show master logs;  
看二进制日志文件用mysqlbinlog  
shell>mysqlbinlog mail-bin.000001  
或者shell>mysqlbinlog mail-bin.000001 | tail  
在配置文件中指定log的輸出位置.  
Windows：Windows 的配置文件为 my.ini，一般在 MySQL 的安装目录下或者 c:\Windows 下。  
Linux：Linux 的配置文件为 my.cnf ，一般在 /etc 下。  
在linux下：  
Sql代码  
# 在[mysqld] 中輸入  
#log  
log-error=/usr/local/mysql/log/error.log  
log=/usr/local/mysql/log/mysql.log  
long\_query\_time=2  
log-slow-queries= /usr/local/mysql/log/slowquery.log

# 在[mysqld] 中輸入 #log

log-error=/usr/local/mysql/log/error.log

log=/usr/local/mysql/log/mysql.log

long\_query\_time=2

log-slow-queries= /usr/local/mysql/log/slowquery.log

windows下:  
Sql代码  
# 在[mysqld] 中輸入  
#log  
log-error="E:/PROGRA~1/EASYPH~1.0B1/mysql/logs/error.log"  
log="E:/PROGRA~1/EASYPH~1.0B1/mysql/logs/mysql.log"  
long\_query\_time=2  
log-slow-queries= "E:/PROGRA~1/EASYPH~1.0B1/mysql/logs/slowquery.log"

# 在[mysqld] 中輸入 #log

log-error="E:/PROGRA~1/EASYPH~1.0B1/mysql/logs/error.log"

log="E:/PROGRA~1/EASYPH~1.0B1/mysql/logs/mysql.log"

long\_query\_time=2

log-slow-queries= "E:/PROGRA~1/EASYPH~1.0B1/mysql/logs/slowquery.log"

开启慢查询  
long\_query\_time =2 --是指执行超过多久的sql会被log下来，这里是2秒  
log-slow-queries= /usr/local/mysql/log/slowquery.log --将查询返回较慢的语句进行记录  
log-queries-not-using-indexes = nouseindex.log --就是字面意思，log下来没有使用索引的query  
log=mylog.log --对所有执行语句进行记录  
windows下开启mysql日志：

在[mysql]下加入这些（基本上等于加在最后面）：

log-error=

#Enter a [name](http://hi.baidu.com/xiaoxiaolq/blog/item/%20:;) for the query log file. Otherwise a default name [will](http://hi.baidu.com/xiaoxiaolq/blog/item/%20:;) be used.

#注:（写成txt文件editplus可以及时重载,不过有时要放在C盘下editplus才可以及时重载）

log= c:/mysql\_query.log.txt

#Enter a name for the slow query log file. Otherwise a default name will be used.

log-slow-queries=

#Enter a name for the update log file. Otherwise a default name will be used.

log-update=

#Enter a name for the binary log. Otherwise a default name will be used.

log-bin=

## 查找备份日志中是否有xx内容

mysqlbinlog --no-defaults log-bin.000005 | grep xx\_order

## 设置外键约束:

SET FOREIGN\_KEY\_CHECKS=1;

**查看外键约束的值**

**SELECT  @@FOREIGN\_KEY\_CHECKS;**

## 双机热备

### 过程

首先，主服务器的mysql版本不能高于从服务器的版本 ，可以是一样的，其次端口号必须统一。

好了，现在开始配置双机热备（随后将会提到循环备份以及互备）我使用的版本为redhat5.4两台ip为192.168.1.10、192.168.0.11，备份数据库为test。

热备中数据库分类：master主数据库、slave从数据库

一、两台mysql均可以在本地登录，并设定好密码，打开防火墙3306端口否则，无法进行通讯。

vi /etc/sysconfig/iptables

#在里面添加3306端口

-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 3306 -j ACCEPT

：wq

二、在主数据库中添加用于备份的账号

mysql> GRANT REPLICATION SLAVE ON \*.\* #注意 这里是星号

　　　　-> TO "backupUser"@"IP" IDENTIFIED BY "password";

backupUser填写建立的用户名，IP为从服务器IP，password就不用说了吧~

三、修改主服务器的my.cnf在配置文件中添加

　　　　server-id=1 #数据库id，默认1

　　　　log-bin=log-bin #日志文件的名称，这里可以制定日志到别的目录 如果没有设置则默认主机名的一个日志名称

　　　　binlog-do-db=db\_name #记录日志的数据库 ,可选

　　　　binlog-ignore-db=mysql #不记录日志的数据库 可选

log-bin是mysql产生的二进制日志，热备份就是读取二进制日志使数据得到时时同步，建议在binlog-ignore中添加mysql这个库，因为在我测试中这个库也被同步过来了，所以...你懂得

四、重启服务器载入新配置文件

进入数据库

mysql -uroot -p

password:

锁定数据库

mysql> FLUSH TABLES WITH READ LOCK;

备份所需要的数据表

mysqldump -uroot -ppassword test > /home/backup/mysql/test.sql

#如果有多个就备份多个

锁定数据库是为了准备记录二进制日志的记录点，否则会在从机配置的时候不能正常读取二进制日志

查看在锁定数据库时二进制日志的记录点

mysql> show master status;

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

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

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

| bin\_log.000001 | 805 | | |

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

1 row in set (0.00 sec)

用笔记录二进制日志名称、以及pos点 bin-log.000001、805

解锁数据表

mysql> UNLOCK TABLES;

下载备份sql然后上传到从机上去

五、首先修改从机mysql配置文件

vi /etc/my.cnf

#添加代码

　　　　server-id=2 //设置数据库id，可以随便设置。但不能重复

　　　　master-host=192.168.1.10 //主服务器的IP地址或者域名

　　　　master-port=3306 //主数据库的端口号

　　　　master-user=backup //同步数据库的用户

　　　　master-password=\*\*\*\*\*\*\*\*\* //同步数据库的密码

　　　　master-connect-retry=60 //如果从服务器发现主服务器断掉，重新连接的时间差

master-retry-count = 999 //尝试连接次数

replicate-do-db=database //表示同步的数据库如多个，请重复添加

binlog-ignore-db=mysql //不同步mysql数据库

设定好上述参数后重启从机mysql

六、进入从机mysql、停止slave（slave开机自动启动)

mysql> slave stop;

更新master信息（之前配置文件中是基础数据，此处是详细配置包括后期更改master信息也可以在mysql>中进行更新，但是端口变更要修改配置文件）

　　　mysql> change master to #进入修改master -> master\_host='192.168.1.10', #定义master地址

-> master\_user='backup', #定义备份时使用的用户

-> master\_password=’password‘, #定义备份用户的密码

-> master\_log\_file='bin\_log.000001', #指定从机数据在master中的二进制日志名称 -> master\_log\_pos=805; #指定从机数据在master中的二进制日志记录点

Query OK, 0 rows affected (0.01 sec)

更新完成，启动slave

mysql> slave start；

查看slave的运行状态

mysql> show slave status\G;

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

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

Master\_Host: 192.168.38.7

Master\_User: backup

Master\_Port: 3306

Connect\_Retry: 60

Master\_Log\_File: bin\_log.000001

Read\_Master\_Log\_Pos: 53229

Relay\_Log\_File: localhost-relay-bin.000002

Relay\_Log\_Pos: 52665

Relay\_Master\_Log\_File: bin\_log.000001

Slave\_IO\_Running: Yes 可以看到此处两个yes代表主从已经开始运行

Slave\_SQL\_Running: Yes 可以做个数据测试了

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: 53229

Relay\_Log\_Space: 52824

Until\_Condition: None

Until\_Log\_File:

Until\_Log\_Pos: 0

Master\_SSL\_Allowed: No

Master\_SSL\_CA\_File:

Master\_SSL\_CA\_Path:

Master\_SSL\_Cert:

Master\_SSL\_Cipher:

Master\_SSL\_Key:

Seconds\_Behind\_Master: 0

Master\_SSL\_Verify\_Server\_Cert: No

Last\_IO\_Errno: 0

Last\_IO\_Error:

Last\_SQL\_Errno: 0

Last\_SQL\_Error:

1 row in set (0.00 sec)

IO和SQL均为yes则表示连同，那么可以去测试了，到此双机热备结束。

### change 命令

change master to master\_host='192.168.1.10',master\_user='root',master\_password='nh6QAew1Wr82d',master\_log\_file='bin\_log.000004',master\_log\_pos=15754;

### 错误

Last\_IO\_Error: Got fatal error 1236 from master when reading data from binary log: 'Could not find first log file name in binary log index file'

本地MySQL环境，是两台MySQL做M-M复制。今天发现错误信息：

mysql 5.5.28-log> show slave status\G  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1. row \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
               Slave\_IO\_State:   
                  Master\_Host: 88.88.88.88  
                  Master\_User: replicate  
                  Master\_Port: 3306  
                Connect\_Retry: 60  
              Master\_Log\_File: testdbbinlog.000005  
          Read\_Master\_Log\_Pos: 98359687  
               Relay\_Log\_File: mysql-relay-bin.000020  
                Relay\_Log\_Pos: 4  
        Relay\_Master\_Log\_File: testdbbinlog.000005  
             Slave\_IO\_Running: No  
            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: 98359687  
              Relay\_Log\_Space: 107  
              Until\_Condition: None  
               Until\_Log\_File:   
                Until\_Log\_Pos: 0  
           Master\_SSL\_Allowed: No  
           Master\_SSL\_CA\_File:   
           Master\_SSL\_CA\_Path:   
              Master\_SSL\_Cert:   
            Master\_SSL\_Cipher:   
               Master\_SSL\_Key:   
        Seconds\_Behind\_Master: NULL  
Master\_SSL\_Verify\_Server\_Cert: No  
                Last\_IO\_Errno: 1236  
                Last\_IO\_Error: Got fatal error 1236 from master when reading data from binary log: 'Could not find first log file name in binary log index file'  
               Last\_SQL\_Errno: 0  
               Last\_SQL\_Error:   
  Replicate\_Ignore\_Server\_Ids:   
             Master\_Server\_Id: 1  
1 row in set (0.00 sec)

mysql 5.5.28-log>

参考网上文档： <http://blog.csdn.net/billfanggs/article/details/8905991>

在source那边，执行：

flush logs;  
show master status;

记下File, Position。

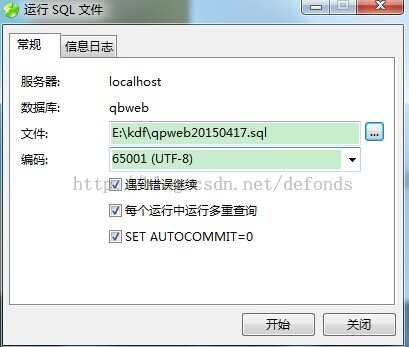
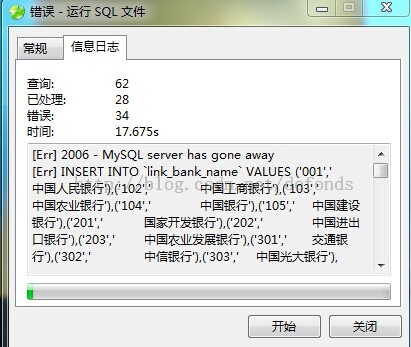
在target端，执行：

CHANGE MASTER TO MASTER\_LOG\_FILE='log-bin.000014',MASTER\_LOG\_POS=106;  
slave start;  
show slave status \G

一切正常。

## MySQL 数据批量恢复时 [Err] 2006 - MySQL server has gone away 错误的解决

2015-04-19 21:31:41CSDN-defonds-点击数:425

Win7、Win8 下，MySQL5.5，执行数据恢复 sql(批量导入)  
  
  
报 [Err] 2006 - MySQL server has gone away 错误：  
  
  
解决方法如下：  
1. 编辑 MySQL 安装目录下的 my.ini，在最后添加以下内容：  
**max\_allowed\_packet=16M**  
2. 重启 MySQL 服务即可

## 重设修改root密码

mysqladmin -u root flush-privileges password "newpassword" 重设root密码

系统安装完成后

/usr/bin/mysqladmin -u root password 'sjcsfwq'

update user set password=password('sjcsfwq') where user='root';

## mysql5.6.26(rpm)

### 如果安装了先卸载旧的版本

a.rpm -qa | grep -i mysql

yum -y remove mysql...

### b.下载需要的安装包，下载地址：

http://cdn.mysql.com/Downloads/MySQL-5.6/MySQL-client-5.6.20-1.el6.x86\_64.rpm

http://cdn.mysql.com/Downloads/MySQL-5.6/MySQL-devel-5.6.20-1.el6.x86\_64.rpm

http://cdn.mysql.com/Downloads/MySQL-5.6/MySQL-server-5.6.20-1.el6.x86\_64.rpm

### c、开始逐个安装

rpm -ivh MySQL-server-5.6.20-1.el6.x86\_64.rpm

rpm -ivh MySQL-devel-5.6.20-1.el6.x86\_64.rpm

rpm -ivh MySQL-client-5.6.20-1.el6.x86\_64.rpm

会生成随机密码,密码在/root/.mysql\_secret

,先用这个密码登上,然后改密码

SET PASSWORD = PASSWORD('sjcsfwq');

### 修改配置文件位置并做相关设置

cp /usr/share/mysql/my-default.cnf /etc/my.cnf

vi /etc/my.cnf

#做如下配置

#[client]

#password = 123456

#port = 3306

#default-character-set=utf8

[mysqld]

port = 3306

character\_set\_server=utf8

character\_set\_client=utf8

collation-server=utf8\_general\_ci

**max\_allowed\_packet=16M**

#default-collation = utf8\_general\_ci

#linux下mysql安装完后是默认：表名区分大小写，列名不区分大小写； 0：区分大小写，1：不区分大小写

lower\_case\_table\_names=1

#设置最大连接数，默认为 151，MySQL服务器允许的最大连接数16384

max\_connections=1000

[mysqld]

default-character-set = utf8

### e.初始化MySQL及设置密码

/usr/bin/mysql\_install\_db

service mysql start

### f.登录到mysql,第一次装没有密码，直接回车

mysql -uroot -p

#设置root用户的密码

mysql> update user set password=password('sjcsfwq') where user='root';

### g.设置允许远程登录

mysql> use mysql;

mysql> select host,user,password from user;

mysql> update user set host='%' where user='root' and host='localhost';

mysql> flush privileges;

mysql> exit;

或

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

flush privileges;

### h.设置开机自启动

chkconfig mysql on

chkconfig --list | grep mysql

### i. MySQL的默认安装位置说明

/var/lib/mysql/ #数据库目录

/usr/share/mysql #配置文件目录

/usr/bin #相关命令目录

/etc/init.d/mysql #启动脚本 注：卸载mysql的时候，将这些目录下的文件也删掉。

### j.可能遇到的错误（一）

2014-01-21 06:03:29 14964 [ERROR] InnoDB: auto-extending data file ./ibdata1 is of a different size 640 pages (rounded down to MB) than specified in the .cnf file: initial 768 pages, max 0 (relevant if non-zero) pages!

2014-01-21 06:03:29 14964 [ERROR] InnoDB: Could not open or create the system tablespace. If you tried to add new data files to the system tablespace, and it failed here, you should now edit innodb\_data\_file\_path in my.cnf back to what it was, and remove the new ibdata files InnoDB created in this failed attempt. InnoDB only wrote those files full of zeros, but did not yet use them in any way. But be careful: do not remove old data files which contain your precious data!

2014-01-21 06:03:29 14964 [ERROR] Plugin 'InnoDB' init function returned error.

2014-01-21 06:03:29 14964 [ERROR] Plugin 'InnoDB' registration as a STORAGE ENGINE failed.

2014-01-21 06:03:29 14964 [ERROR] Unknown/unsupported storage engine: InnoDB

2014-01-21 06:03:29 14964 [ERROR] Aborting

在/var/lib/mysql/目录下删掉这三个文件：ibdata1 ib\_logfile0 ib\_logfile1 然后重启mysql

cd /var/lib/mysql

rm ibdata1 ib\_logfile0 ib\_logfile1

service mysql start

### k.可能遇到的错误（二）

[root@localhost local]# mysql -uroot -p

Enter password:

ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: YES)

方法操作很简单，如下：

# /etc/init.d/mysql stop

# mysqld\_safe --user=mysql --skip-grant-tables --skip-networking &

# mysql -u root mysql

//把空的用户密码都修改成非空的密码。

mysql> UPDATE user SET Password=PASSWORD('sjcsfwq') where USER='root' and host='root' or host='localhost';

mysql> FLUSH PRIVILEGES;

mysql> quit

# /etc/init.d/mysql restart

# mysql -uroot -p

Enter password: <输入新设的密码newpassword>

另

d:\mysql\bin\mysqladmin -u root flush-privileges password "newpassword" 重设root密码

### l.可能遇到的错误（三）

mysql> show databases;

ERROR 1820 (HY000): You must SET PASSWORD before executing this statement

这句话要求你重新设置一次密码！

mysql>  SET PASSWORD = PASSWORD('sjcsfwq');

Query OK, 0 rows affected (0.03 sec)

mysql> create database roger;

Query OK, 1 row affected (0.00 sec)

参考：<http://blog.csdn.net/liumm0000/article/details/18841197>

### 可能遇到的错误（四）

[MySQL错误Another MySQL daemon already running with the same unix socket.](http://blog.csdn.net/acmilanvanbasten/article/details/17502811)

在国外网站发现的解决方法。

原因多个Mysql进程使用了同一个socket。

两个方法解决：

第一个是立即关机 使用命令 shutdown -h now 关机，关机后在启动，进程就停止了。

第二个直接把mysql.sock文件改名即可。也可以删除，推荐改名。

然后就可以启动mysql了。

### the server quit without updating pid file :

### 看日志发现 can’t create/write to file /var/run/mysqld/mysqld.pid

没有目录,先创建/var/run/mysqld



解决办法：

将 /var/run/mysqld/权限赋给mysql   
  
执行以下命令即可：

chown -R mysql /var/run/mysqld  
chgrp -R mysql /var/run/mysqld

### ERROR! MySQL is running but PID file could not be found ?

第一种方法：可能是硬盘满了，清理下垃圾文件。

第二种：

查看下数据库运行状态

 /etc/init.d/mysql status  
提示  
ERROR! MySQL is running but PID file could not be found  
先打印MYSQL进程  
ps aux | grep mysql  
然后KILL进程  
kill -9 pid1 pid2 …  
再启动MYSQL  
/etc/init.d/mysql start  
再检查mysql运行状态  
/etc/init.d/mysql status

提示成功,一般是这种情况

第三种：是修改过my.cnf配置出错引起 执行命令：vi /etc/my.cnf  修改内容：

# The MySQL server  
[mysqld]  
port　　　= 3306  
socket　 = /var/lib/mysql/mysql.sock

在执行第二种命令，OK

## Warning: World-writable config file '/etc/my.cnf' is ignored

1. 问题描述：

重启mysql服务时出现以下信息：

Warning: World-writable config file '/etc/my.cnf' is ignored

出现这种情况的原因是：mysql故意不让用户重写my.cnf，权限默认是644, 即-rw-r--r--, 如果权限不是644， 则会出现这种情况。

2. 解决方法:

(1) 查看my.cnf文件的权限：

ls -l /etc/my.cnf

信息如下：

cryhelyxx@ada:~$ ls -l /etc/my.cnf

-rw-r--r-- 1 root root 1126 4月 13 16:46 /etc/my.cnf

cryhelyxx@ada:~$

可以看到my.cnf文件权限为644， 如果你的my.cnf文件权限不是"-rw-r--r--"， 而是如“-rwxrwxrwx”等等， 则执行以下命令：

sudo chmod 644 /etc/my.cnf

(2) 进入mysql目录下：

cd /usr/local/mysql/

(3) 重启mysql服务：

sudo ./support-files/mysql.server restart

3. OK, Enjoy it!!!

#### Linux下MySQL5.6的修改字符集编码为UTF8（解决中文乱码问题）

Linux下MySQL5.6的修改字符集编码为UTF8（解决中文乱码问题）

一、登录MySQL查看用SHOW VARIABLES LIKE ‘character%’;下字符集，显示如下：

|  |  |
| --- | --- |
|  | +--------------------------+----------------------------+  | Variable\_name | Value |  +--------------------------+----------------------------+  | character\_set\_client | utf8 |  | character\_set\_connection | utf8 |  | character\_set\_database | latin1 |  | character\_set\_filesystem | binary |  | character\_set\_results | utf8 |  | character\_set\_server | latin1 |  | character\_set\_system | utf8 |  | character\_sets\_dir | /usr/share/mysql/charsets/ |  +--------------------------+----------------------------+ |

character\_set\_database和character\_set\_server的默认字符集还是latin1。

二、最简单的完美修改方法，修改mysql的my.cnf文件中的字符集键值（注意配置的字段细节）：

|  |  |
| --- | --- |
|  | 1、在[client]字段里加入default-character-set=utf8，如下：    [client]  port = 3306  socket = /var/lib/mysql/mysql.sock  default-character-set=utf8    2、在[mysqld]字段里加入character-set-server=utf8，如下：  [mysqld]  port = 3306  socket = /var/lib/mysql/mysql.sock  character-set-server=utf8  3、在[mysql]字段里加入default-character-set=utf8，如下：  [mysql]  no-auto-rehash  default-character-set=utf8 |

修改完成后，service mysql restart重启mysql服务就生效。注意：[mysqld]字段与[mysql]字段是有区别的。这点在网上没人反馈过。

使用SHOW VARIABLES LIKE ‘character%’;查看，发现[数据库](http://www.2cto.com/database/)编码全已改成utf8。

#### MYSQL安装过程中出现错误ERROR 1820 (HY000): You must SET PASSWORD before executing this statement解决

最近新装好的mysql在进入mysql工具时，总是有错误提示:

# mysql -u root -p

Enter password:

ERROR 1045 (28000): Access denied for user 'root'@'localhost' (using password: NO)

或者

# mysql -u root -p password 'newpassword'

Enter password:

mysqladmin: connect to server at 'localhost' failed

error: 'Access denied for user 'root'@'localhost' (using password: YES)'

现在终于找到解决方法了。本来准备重装的，现在不必了。

方法操作很简单，如下：

# /etc/init.d/mysql stop

# mysqld\_safe --user=mysql --skip-grant-tables --skip-networking &

# mysql -u root mysql

mysql> UPDATE user SET Password=PASSWORD('newpassword') where USER='root' and host='root' or host='localhost';//把空的用户密码都修改成非空的密码就行了。

mysql> FLUSH PRIVILEGES;

mysql> quit # /etc/init.d/mysqld restart

# mysql -uroot -p

Enter password: <输入新设的密码newpassword>

MySql5.6操作时报错：You must SET PASSWORD before executing this statement解决

mysql>  SET PASSWORD = PASSWORD('123456');

Query OK, 0 rows affected (0.03 sec)

mysql> create database roger;

Query OK, 1 row affected (0.00 sec)

也就是用mysql>  SET PASSWORD = PASSWORD('123456');这句话重新设置一次密码！

如果碰到

Starting MySQL.The server quit without updating PID file (/var/lib/mysql/loc

删除一下/var/lib/mysql/mysql.sock再重启试试

## mysql数据库安装配置(rpm老版本)

### （安装mysql安装）

1. 也可直接
2. yum install mysql
3. yum install mysql-server
4. yum install mysql-devel
5. service msyqld start

rpm sever,client,devel，安装前先检查是否安装过，用rpm -qa|Mysql rpm -qa|mysql,如果装了要先卸载 rpm -e 查出来的名字，如果还不能卸载，到add/remove 里卸载依赖的，以及所有带mysql的

先启动服务

装完之后配置，先为root用户加密码 复制sever装完以后的加密码命令 /usr/bin/mysqladmin -u root password 'sjcsfwq'

然后登录 mysql -u root -p 或者明文登录mysql -uroot -psjcsfwq

登录进去之后查看数据库 show databases;然后use 数据库名称，比如use mysql;然后查看所有当前数据库的表 show tables

select host,user,password from user;从user表里面查询这三个字段

新建root,dingkai用户 insert into user(host,user,password) 新建%改成localhost values('%','root',password('sjcsfwq'));然后把root改成dingkai 密码dingkai再来一遍

insert into user(host,user,password) values('%','root', 'sjcsfwq');

然后给权限:grant all privileges on \*.\* to 'root'@'%' identified by 'sjcsfwq' with grant option;用户名哎他所有主机 密码，然后root用户来一次

然后刷新权限，flush privileges;

然后windows 装navicat连接dingkai导入数据库sql

新建数据库，要名字，一般是SJCS，与集中监控配置文件中的数据库配置项数据库名对应，还有客户端的，字符集和整理都选最后一个utf8

配置文件

Vim /etc/my.cnf

[mysqld]

character\_set\_server=utf8

**max\_allowed\_packet=16M**

### [[MySQL] - MySQL的Grant命令](http://www.cnblogs.com/hcbin/archive/2010/04/23/1718379.html)

**本文实例，运行于** MySQL 5.0 及以上版本。

MySQL 赋予用户权限命令的简单格式可概括为：

grant 权限 on 数据库对象 to 用户

一、grant 普通数据用户，查询、插入、更新、删除 数据库中所有表数据的权利。

grant select on testdb.\* to common\_user@'%'  
grant insert on testdb.\* to common\_user@'%'  
grant update on testdb.\* to common\_user@'%'  
grant delete on testdb.\* to common\_user@'%'

或者，用一条 MySQL 命令来替代：

grant select, insert, update, delete on testdb.\* to common\_user@'%'

二、grant 数据库开发人员，创建表、索引、视图、存储过程、函数。。。等权限。

grant 创建、修改、删除 MySQL 数据表结构权限。

grant create on testdb.\* to developer@'192.168.0.%';  
grant alter on testdb.\* to developer@'192.168.0.%';  
grant drop on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 外键权限。

grant references on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 临时表权限。

grant create temporary tables on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 索引权限。

grant index on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 视图、查看视图源代码 权限。

grant create view on testdb.\* to developer@'192.168.0.%';  
grant show view on testdb.\* to developer@'192.168.0.%';

grant 操作 MySQL 存储过程、函数 权限。

grant create routine on testdb.\* to developer@'192.168.0.%'; -- now, can show procedure status  
grant alter routine on testdb.\* to developer@'192.168.0.%'; -- now, you can drop a procedure  
grant execute on testdb.\* to developer@'192.168.0.%';

三、grant 普通 DBA 管理某个 MySQL 数据库的权限。

grant all privileges on testdb to dba@'localhost'

其中，关键字 “privileges” 可以省略。

四、grant 高级 DBA 管理 MySQL 中所有数据库的权限。

grant all on \*.\* to dba@'localhost'

五、MySQL grant 权限，分别可以作用在多个层次上。

1. grant 作用在整个 MySQL 服务器上：

grant select on \*.\* to dba@localhost; -- dba 可以查询 MySQL 中所有数据库中的表。  
grant all on \*.\* to dba@localhost; -- dba 可以管理 MySQL 中的所有数据库

2. grant 作用在单个数据库上：

grant select on testdb.\* to dba@localhost; -- dba 可以查询 testdb 中的表。

3. grant 作用在单个数据表上：

grant select, insert, update, delete on testdb.orders to dba@localhost;

这里在给一个用户授权多张表时，可以多次执行以上语句。例如：

grant select(user\_id,username) on smp.users to mo\_user@'%' identified by '123345';  
grant select on smp.mo\_sms to mo\_user@'%' identified by '123345'  
4. grant 作用在表中的列上：

grant select(id, se, rank) on testdb.apache\_log to dba@localhost;

5. grant 作用在存储过程、函数上：

grant execute on procedure testdb.pr\_add to 'dba'@'localhost'  
grant execute on function testdb.fn\_add to 'dba'@'localhost'

六、查看 MySQL 用户权限

查看当前用户（自己）权限：

show grants;

查看其他 MySQL 用户权限：

show grants for dba@localhost;

七、撤销已经赋予给 MySQL 用户权限的权限。

revoke 跟 grant 的语法差不多，只需要把关键字 “to” 换成 “from” 即可：

grant all on \*.\* to dba@localhost;  
revoke all on \*.\* from dba@localhost;

八、MySQL grant、revoke 用户权限注意事项

1. grant, revoke 用户权限后，该用户只有重新连接 MySQL 数据库，权限才能生效。

2. 如果想让授权的用户，也可以将这些权限 grant 给其他用户，需要选项 “grant option“

grant select on testdb.\* to dba@localhost with grant option;

这个特性一般用不到。实际中，数据库权限最好由 DBA 来统一管理。\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

遇到 SELECT command denied to user '用户名'@'主机名' for table '表名' 这种错误，解决方法是需要把吧后面的表名授权，即是要你授权核心数据库也要。

我遇到的是SELECT command denied to user 'my'@'%' for table 'proc'，是调用存储过程的时候出现，原以为只要把指定的数据库授权就行了，什么存储过程、函数等都不用再管了，谁知道也要把数据库mysql的proc表授权

mysql授权表共有5个表：user、db、host、tables\_priv和columns\_priv。

授权表的内容有如下用途：  
**user表**  
user表列出可以连接服务器的用户及其口令，并且它指定他们有哪种全局（超级用户）权限。在user表启用的任何权限均是全局权限，并适用于所有数据库。例如，如果你启用了DELETE权限，在这里列出的用户可以从任何表中删除记录，所以在你这样做之前要认真考虑。

**db表**  
db表列出数据库，而用户有权限访问它们。在这里指定的权限适用于一个数据库中的所有表。

**host表**  
host表与db表结合使用在一个较好层次上控制特定主机对数据库的访问权限，这可能比单独使用db好些。这个表不受GRANT和REVOKE语句的影响，所以，你可能发觉你根本不是用它。

**tables\_priv表**  
tables\_priv表指定表级权限，在这里指定的一个权限适用于一个表的所有列。

**columns\_priv表**  
columns\_priv表指定列级权限。这里指定的权限适用于一个表的特定列

### 错误，问题

MySQL错误Another MySQL daemon already running with the same unix socket.

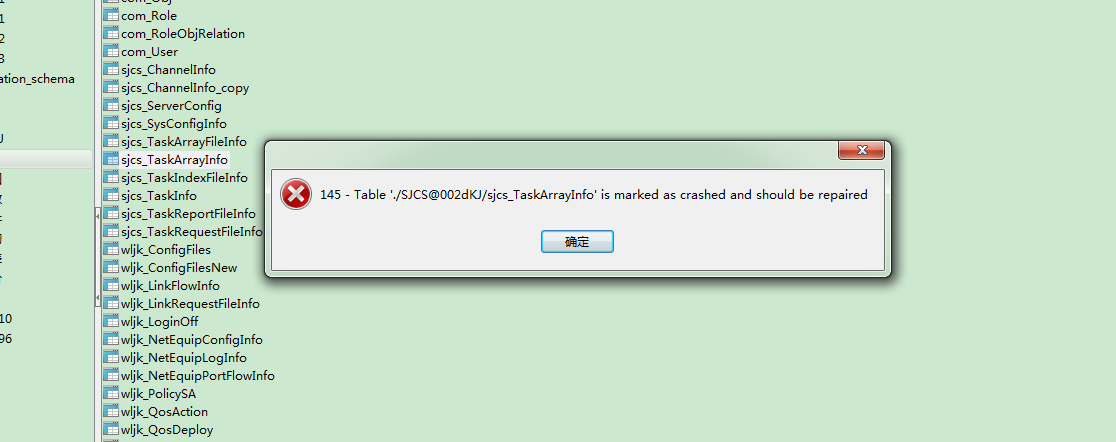
mysql.sock改名，centos中在/var/lib/mysql/mysql.sock

什么 through sock啥玩意的,先看看服务开了没再说

## MySQL manager or server PID file could not be found



删除数据之后后台连数据库获取数据可能会崩,mysq\_fetch\_row时,有可能是数据错误,navicat点击表试一下,如果发现打不开的表,



右键维护修复,重启就好了

# Oracle

## 时间日期格式转换为字符串后20点变08点

yyyy-mm-dd hh24:mi:ss

# Sql语句

## 字段中含有字符串

select \* from data where type like '

%road%'

## In

select \* from SysLog where UserName in (select UserName from Users)

UserName在另一个表中

## 查找个数

Select(distinct field) from table

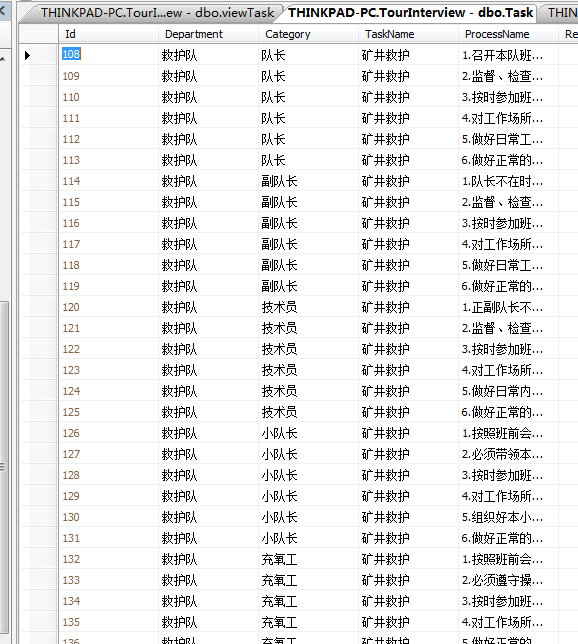
## 视图

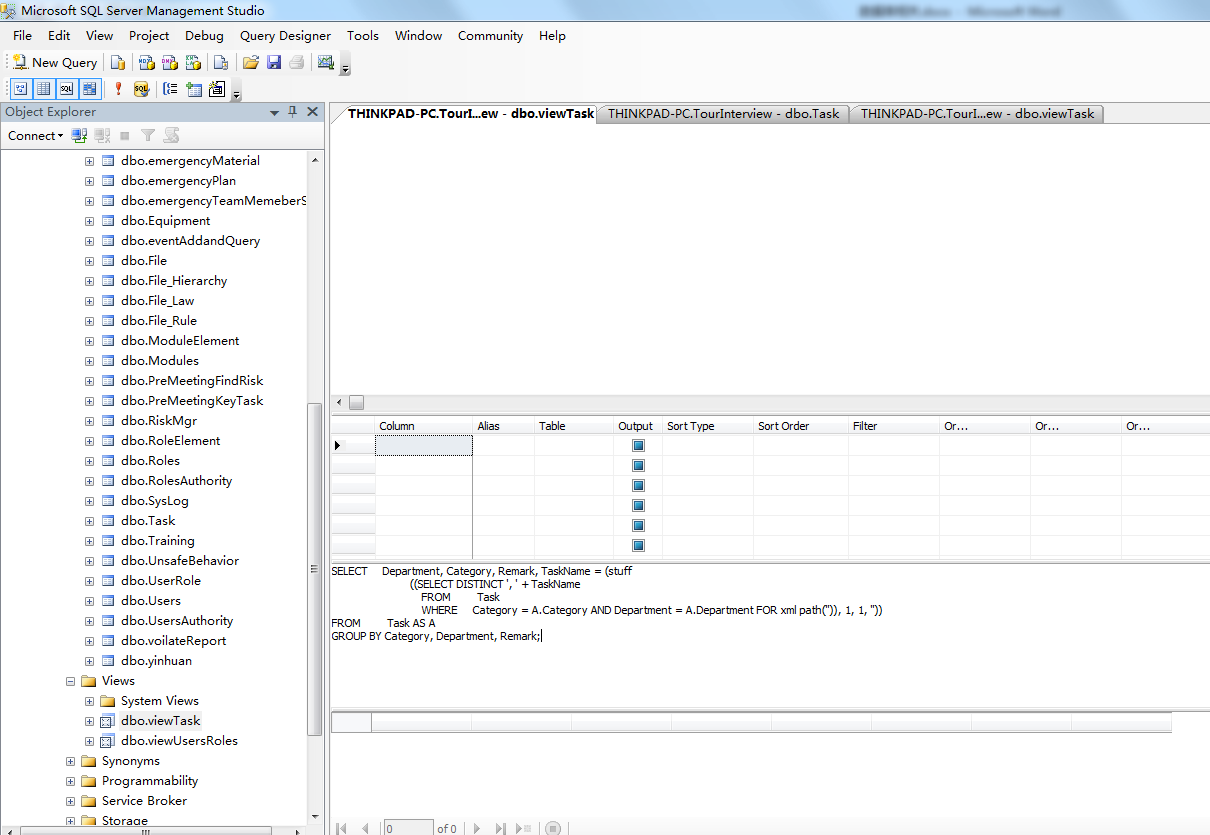
即建立一个虚拟表

## 复制表

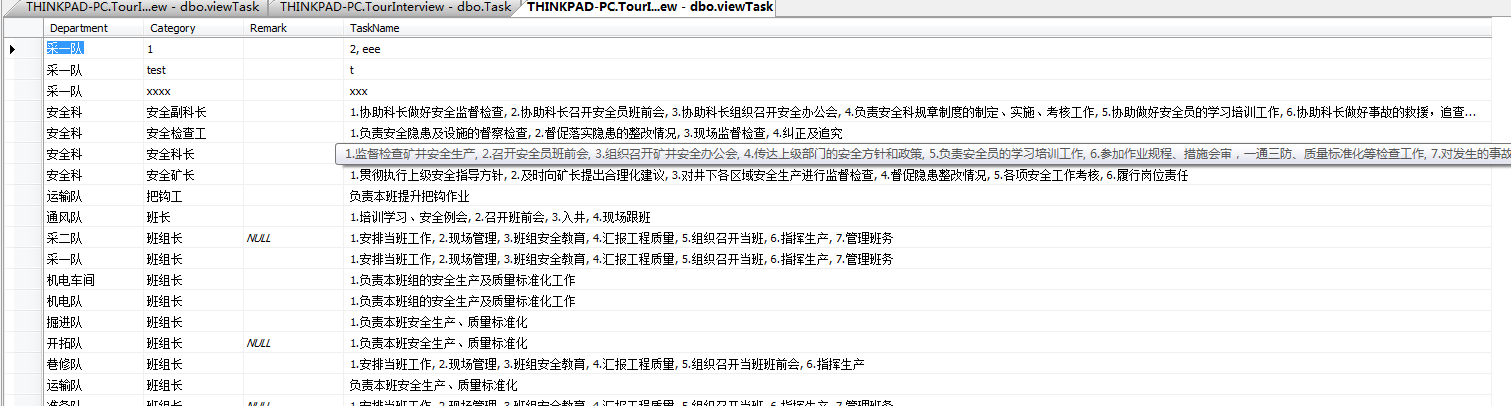
select \* into Equipment2 from Equipment

## 清空表

truncate table Equipment2



这个是将同一工种的不同任务组合到一条,结果如下



## 知识

### mysql\_real\_query()

对于包含[二进制](http://baike.baidu.com/subview/18536/18536.htm)数据的查询，你*必须*使用mysql\_real\_query()而不是mysql\_query()，因为[二进制代码](http://baike.baidu.com/view/2418160.htm)数据可能包含“\0”字符，而且，mysql\_real\_query()比mysql\_query()更快，因为它不会在查询字符串上调用[strlen](http://baike.baidu.com/subview/736226/5092487.htm)()。如果查询成功，函数返回零。如果发生一个错误，函数返回非零。出错的代码及原因如下所示：

## 清空表

TRUNCATE TABLE teacher

如果不对的话

mysql> SET FOREIGN\_KEY\_CHECKS = 0;

Query OK, 0 rows affected (0.02 sec)

mysql> delete from repositories where repo\_name = "watch/sinawatch\_agent\_update";

Query OK, 1 row affected (0.02 sec)

mysql> SET FOREIGN\_KEY\_CHECKS = 1;

Query OK, 0 rows affected (0.00 sec)

MYSQL 在navicat中数据库右键---命令行

## 获取刚插入的自增id

select @@IDENTITY

# navicat

## cannot delete or update a parent row a foreign key constraint fails

是因为要导入的表里面有别的表的外键对应,这个字段必须有,且类型相同.

# mysql数据库设置远程连接

# windows和linux系统开启mysql远程连接权限的方法

来源：本站转载 作者：佚名 时间：2012-11-20 TAG： [我要投稿](http://www.myhack58.com/public.html)

|  |  |
| --- | --- |
| |  | | --- | | 大家在远程连接mysql的时候应该碰到过，root用户的mysql，只可以本地连，对外拒绝连接。  我们需要建立一个允许远程登录的数据库帐户，这样你可以方便进行在本地远程操作数据库了。  方法如下：  一：windows环境  1。 改表法。可能是你的帐号不允许从远程登陆，只能在localhost。这个时候只要在localhost的那台电脑，登入mysql后，更改 “mysql” 数据库里的 “user” 表里的 “host” 项，从”localhost”改称”%”  mysql -u root -pvmwaremysql>use mysql;mysql>update user set host = ‘%’ where user = ‘root’;mysql>select host, user from user;  2. 授权法。例如，你想myuser使用mypassword从任何主机连接到mysql服务器的话。  GRANT ALL PRIVILEGES ON \*.\* TO ‘myuser’@'%’ IDENTIFIED BY ‘mypassword’ WITH GRANT OPTION;  如果你想允许用户myuser从ip为192.168.1.3的主机连接到mysql服务器，并使用mypassword作为密码  GRANT ALL PRIVILEGES ON \*.\* TO ‘myuser’@’192.168.1.3′ IDENTIFIED BY ‘mypassword’ WITH GRANT OPTION;  我 的mysql.user里root用户的host果然是localhost，先用改表法给localhost改成“％”，还是不行，仍然报1130的错 误，又按“从任何主机连接到mysql服务器”方法授权，还是报一样的错，最后给自己的ip授权之后，终于登录上了。。。。  乎乎。。。  mysql的ERROR 1045 在上面情况后如再出现客户段1045可在服务器执行如下  UPDATE user SET Password=PASSWORD(’123456′) where USER=’myuser’;  FLUSH PRIVILEGES;//用户付完权限后对数据进行刷新时用!要不Mysql数据库识别不了  例如:(必须参考以上再看下面的例子)  GRANT ALL ON \*.\* TO admin@’%’ IDENTIFIED BY ‘admin’ WITH GRANT OPTION;  mysql “192.168.50.85″ “admin” “admin” ;  GRANT ALL ON \*.\* TO admin@’localhost’ IDENTIFIED BY ‘admin’ WITH GRANT OPTION;  GRANT ALL PRIVILEGES ON \*.\* TO ‘myuser’@'%’ IDENTIFIED BY ’123456′ WITH GRANT OPTION;  GRANT ALL PRIVILEGES ON \*.\* TO ‘myuser’@’192.168.50.85′ IDENTIFIED BY ’123456′ WITH GRANT OPTION;  UPDATE user SET Password=PASSWORD(’123456′) where USER=’myuser’;  FLUSH PRIVILEGES;  mysql默认是没有开启远程控制的，必须添加远程访问的用户。  mysql默认是没有开启远程控制的，必须添加远程访问的用户。 g&00 x$ Y  0VE }0(fQ  如果是安装版的话，从mysql提供的控制台进入。 1s+<( 0R/  用root用户登陆，然后： Q k C jgS0  grant all privileges on \*.\* to 创建的用户名 @”%” identified by “密码”; o! q\*)`xuQ  flush privileges; \* 刷新刚才的内容\* )/ z (5 3X  ; (Q:LY \v  格式：grant 权限 on 数据库名.表名 to 用户@登录主机 identified by “用户密码”; .=h j Qi[  @ 后面是访问mysql的客户端IP地址（或是 主机名） % 代表任意的客户端，如果填写 localhost 为本地访问（那此用户就不能远程访问该mysql数据库了）。 U <T 1O 6  lp 8 z Rc  同时也可以为现有的用户设置是否具有远程访问权限。如下： \ch3 X X#  use mysql; o$`> :>N e  update db set host = '%' where user = '用户名'; （如果写成 host=localhost 那此用户就不具有远程访问权限） B Gnl: R \*  FLUSH PRIVILEGES; zn 6en;&?  3 Pj\_2B=  查看结果，执行： $4b 5-U  use mysql; ){[' 66!  select host,user,password from user; J):H\B:j\_M  i oZ 0 " H  +--------------+--------+---------------+ t75 JSdQ  | Host | User | Password | XKL Di{ U  +--------------+--------+-----------------+ B\*1Q< Cx5B  | localhost | | | U 8RZq(f  | % | John | 123456 | r ` Al B  | localhost | root | | H fd ZE  +--------------+------ --+---------------+ pd7= 8 8  ArJ~Vs "+  总结： ? Av L`B A  其实MySQL默认有两个数据库分别为 mysql 和 test 而MySQL用户的信息都放在数据库 mysql 的相关表中 gb: hj /  也可以使用 Navicat （推荐）这样的GUI工具来管理用户。 HUD?}U BB5  update user set host='%' where host='localhost'  工作中用到[web服务器](http://www.myhack58.com/Article/sort099/sort0100/Article_100_1.htm)和mysql服务器不在同一台计算机上安装的运用。需要通过mysql的远程账户访问mysql。  先上语法：  grant [权限] on [数据库名].[表命] to ['用户命']@['[web服务器](http://www.myhack58.com/Article/sort099/sort0100/Article_100_1.htm)的ip地址'] identified by ['密码'];  实例：  grant all on lportal.\* to ‘root’@’192.168.1.21 identified by ‘root’;  grant select,update,insert,delete on lportal.\* to ‘root’@’192.168.1.21 identified by ‘root’;    二：linux类环境  1.修改/etc/mysql/my.conf,修改bind-address,指定为本机实际IP地址,你的my.con修改完大概是如下的样子 [mysqld] # #　\*　Basic　Settings # default-character-set=utf8 default-storage-engine=INNODB bind-address　　　　=　192.168.1.107 user　　　　　　　　　　　　=　mysql pid-file　　　　　　　　=　/var/run/mysqld/mysqld.pid socket　　　　　　　　　　=　/var/run/mysqld/mysqld.sock port　　　　　　　　　　　　=　3306 basedir　　　　　　　　　=　/usr datadir　　　　　　　　　=　/var/lib/mysql tmpdir　　　　　　　　　　=　/tmp language　　　　　　　　=　/usr/share/mysql/english skip-external-locking 2.用root登陆mysql执行如下命令 grant　all　on　sonardb.\*　to　sonar@’%’　identified　by　’123456′; grant　all　on　sonardb.\*　to　sonar@localhost　identified　by　’123456′; sonardb替换为你想访问的数据库名,sonar是你的想使用的用户名,123456替换为你的密码,这样就开启了远程访问功能 | |

# 单机 mysql 主从服务器配置

mysql的安装位置：/home/45MJ/workspace/

使用的mysql版本：mysql-5.6.19-linux-glibc2.5-x86\_64.tar.gz（copy到安装位置）

配置过程如下：

1. **安装mysql(master和slave)**

**$** tar -zxvf mysql-5.6.19-linux-glibc2.5-x86\_64.tar.gz

**$** mv mysql-5.6.19-linux-glibc2.5-x86\_64 mysql-master

**$** cp -r mysql-master/ mysql-slave

**$** cd mysql-master

**$** ./scripts/mysql\_install\_db

**$** cd ../mysql-slave

**$** ./scripts/mysql\_install\_db

1. **主服务器配置**

1) 修改master下的配置文件my.cnf：

[mysqld]

**lower\_case\_table\_names=1**

**explicit\_defaults\_for\_timestamp=true**

# Remove leading # and set to the amount of RAM for the most important data

# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.

# innodb\_buffer\_pool\_size = 128M

# Remove leading # to turn on a very important data integrity option: logging

# changes to the binary log between backups.

**log\_bin=mysql-master.bin**

# These are commonly set, remove the # and set as required.

**basedir = /home/45MJ/workspace/mysql-master**

**datadir = /home/45MJ/workspace/mysql-master/data**

**port = 3306**

**server\_id = 1**

**socket = /tmp/mysql.sock**

**# binlog-do-db=repl #需要同步的数据库，如果没有本行，即表示同步所有的数据库，多项的话重复即可**

**# binlog-ignore-db=mysql #被忽略的数据库，多项重复即可**

# Remove leading # to set options mainly useful for reporting servers.

# The server defaults are faster for transactions and fast SELECTs.

# Adjust sizes as needed, experiment to find the optimal values.

**join\_buffer\_size = 128M**

**sort\_buffer\_size = 2M**

**read\_rnd\_buffer\_size = 2M**

sql\_mode=NO\_ENGINE\_SUBSTITUTION,STRICT\_TRANS\_TABLES

1. 授予slave replication权限：

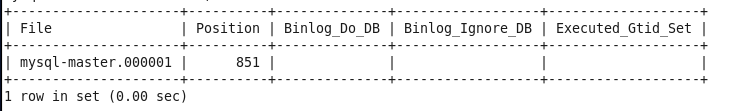
**$** ./bin/mysqld --defaults-file=my.cnf & //启动myql-master下的mysql

**$** ./bin/mysql -h127.0.0.1 -P3306 -uroot -p //登录mysql，初始密码为空，可自行修改

mysql >grant replication slave on \*.\* to 'repl'@'127.0.0.1' identified by 'root'; //为slave设置 访问master端mysql数据库的用户’repl’，密码为’root’, ip为本地（127.0.0.1），因为mysql的master和slave在同一个机器上，如下：

mysql > flush tables with read lock; //锁定数据库，此时不允许更改任何数据

mysql > show master status; //查看状态，这些数据是要记录的，一会要在slave端用到



1. **从服务器配置**

1) 修改slave下的配置文件my.cnf：

[mysqld]

**lower\_case\_table\_names=1**

**explicit\_defaults\_for\_timestamp=true**

# Remove leading # and set to the amount of RAM for the most important data

# cache in MySQL. Start at 70% of total RAM for dedicated server, else 10%.

# innodb\_buffer\_pool\_size = 128M

# Remove leading # to turn on a very important data integrity option: logging

# changes to the binary log between backups.

**log\_bin=mysql-slave.bin**

# These are commonly set, remove the # and set as required.

**basedir = /home/45MJ/workspace/mysql-slave**

**datadir = /home/45MJ/workspace/mysql-slave/data**

**port = 3307**

**server\_id = 2**

**socket = /tmp/mysql2.sock**

**master-host=127.0.0.1**

**master-user=repl**

**master-password=root**

**master-port=3306**

**#replicate-do-db=test #需要备份的数据库名**

**#replicate-ignore-db=mysql #忽略的数据库**

# Remove leading # to set options mainly useful for reporting servers.

# The server defaults are faster for transactions and fast SELECTs.

# Adjust sizes as needed, experiment to find the optimal values.

**join\_buffer\_size = 128M**

**sort\_buffer\_size = 2M**

**read\_rnd\_buffer\_size = 2M**

ql\_mode=NO\_ENGINE\_SUBSTITUTION,STRICT\_TRANS\_TABLES

2) slave设置同步参数

$ ./bin/mysqld --defaults-file=my.cnf & //启动myql-slave下的mysql

$ ./bin/mysql -h127.0.0.1 -P3306 -uroot –p //登录mysql，初始密码为空，可自行修改

mysql> stop slave (注：网上有用slave stop的，应该是mysql版本的问题)

mysql> change master to

-> master\_host='127.0.0.1',

-> master\_user='repl',

-> master\_password='root',

-> master\_log\_file='mysql-master.000001',

-> master\_log\_pos=851;

mysql> start slave;

mysql> show slave status\G; //查看slave状态，其中 Slave\_IO\_Running 和

Slave\_SQL\_Running 两列的值都为 "Yes"，表明 Slave 的 I/O 和 SQL 线程  
都在正常运行。

1. **主服务器解锁**

mysql> unlock tables;

到此主从库搭建成功。可以在主库上插入数据测试同步是否正常。

1. **测试**

**主服务器在创建表并在表中插入数据，查看slave是否同步**：

mysql> show databases;

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

| Database |

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

| information\_schema |

| db1 |

| mysql |

| performance\_schema |

| test |

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

5 rows in set (0.00 sec)

mysql> use db1;

mysql> use db1;

mysql> create table masterslavetest(id int,name char(10));

mysql> insert into masterslavetest values(1,'master');

mysql> insert into masterslavetest values(1,'slave');

mysql> select \* from masterslavetest;

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

| id | name |

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

| 1 | master |

| 2 | slave |

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

**从服务器查看表内容：**

mysql> use db1;

mysql> select \* from masterslavetest;

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

| id | name |

+------+--------+s

| 1 | master |

| 2 | slave |

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

查到数据，同步成功

注：过程中遗漏一些参数，如master-connect-retry=60 #如果从服务器发现主服务器断掉，重新连接的时间差(秒），未添加没有影响到此次搭建，但是应该都是些挺重要的参数，下一版再细研究。另，配置出现异常问题暂时没有解决办法，基本靠度娘。