# 一：MySQL字符集

## 1：字符集介绍

**字符集就是一套文字符号及其编码、比较规则的集合，第一个计算机字符集ASC2!MySQL数据库字符集包括字符集（CHARACTER）和校对规则（COLLATION）两个概念，其中字符集用来定义MySQL数据字符串的存储方式，而校对规则定义比较字符串的方式**

## 2：MySQL数据库常见字符集介绍



**选择字符集建议使用国际标准的utf8**

## 3：MySQL怎样选择合适的字符集

**1、如果处理各种各样的文字，发布到不同语言国家地区，应选Unicode字符集，对MySQL来说就是utf8(每个汉字三个字节)**

**2、如果只是需要支持中文，并且数据量很大，性能要求也高，可选GBK(定长，每个汉字占双字节，英文也占双字节)，如果是大量运算，比较排序等，定长字符集更快，性能也高**

**3、处理移动互联网业务，可能需要使用utf8mb4字符集，apple**

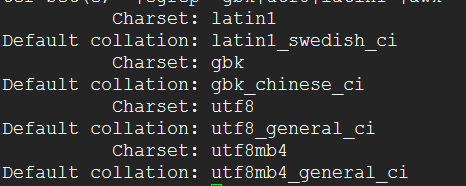
## 4：查看当前MySQL支持的字符集

M**ySQL可以支持多种字符集，同一台服务器，库或表的不同字段都可以指定不同的字符集**

**mysql -uroot -p123456 -e "show character set \G;" 查看所有的字符集**

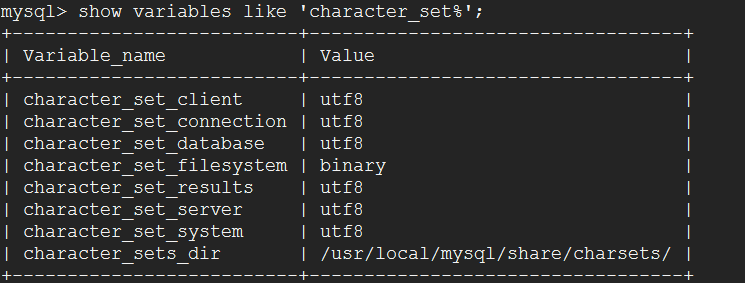
**查看常用的字符集：**

**mysql -uroot -p123456 -e "show character set\G;" |egrep "gbk|utf8|latin1"|awk '{print $0}'**



## 5：查看MySQL当前的字符集设置情况

**mysql> show variables like 'character\_set%';**



## 6：实战，迁移数据

**背景：公司业务数据book,由于之前建表没注意字符集的问题，导致之前写入的数据出现乱码。现在要将之前的数据和现在数据的字符集一致，不出现乱码情况**

**将字符集为latin1已有记录的数据转成utf8，并且已经存在的记录不乱码**

**步骤**

**1：建库及建表的语句导出，sed批量修改为utf8**

**2：导出之前所有的数据**

**3：修改mysql服务端和客户端编码为utf8**

**4：删除原有的库表及数据**

**5：导入新的建库及建表语句**

**6：导入之前的数据**

**1）：导出表结构**

**mysqldump –uroot --default-character-set=latin1 -d book2> booktable.sql**

**2）:编辑booktable.sql 将latin1修改成utf8**

**vim booktable.sql 修改所有latin1为utf8**

**3）:确保数据库不再更新，导出所有数据**

**mysqldump -uroot –p123456 --quick --no-create-info --extended-insert --default-character-set=latin1 book2>bookdata.sql**

**参数说明：**

**--quick:用于转储大的表，强制mysqldump从服务器一次一行的检索数据而不是检索所有行，并输出当前cache到内存中**

**--no-create-info:不要创建create table语句**

**--extended-insert:使用包括几个values列表的多行insert语法，这样文件更小，IO也小，导入数据时会非常快**

**--default-character-set=latin1:按照原有字符集导出数据，这样导出的文件中，所有中文都是可见的，不会保存成乱码**

**4）：打开bookdata.sql 将SET NAME latin1 修改成SET NAME utf8**

**vim bookdata.sql**

**/\*!40101 SET NAMES utf8 \*/;**

**5）:重新建库**

**mysql> create database book2 default charset utf8;**

**6）：建立表，导入我们之前导出的表的数据库**

**mysql –uroot –p123456 book <booktable.sql**

**7）:导入数据**

**mysql -uroot -p123456 book2 <bookdata.sql**

**注意：选择目标字符集时，要注意最好大于等于原字符集（字库更大），否则可能会丢失不被支持的数据**

# 二：MySQL日常维护

## 1：mysqlcheck🡪 mysql修复工具

**mysqlcheck客户端工具可以检查和修复MyISAM表，还可以优化和分析表。**

**实际上，它集成了mysql工具中check、repair、analyze、tmpimize的功能。**

**/usr/local/mysql/bin/mysqlcheck #源码编译安装位置**

**rpm -qf `which mysqlcheck` yum安装查看**

**参数选项：**

**mysqlcheck –-help 查看帮助**

**-c, --check (检查表)；**

**-r, --repair（修复表)；**

**-a, --analyze (分析表)；**

**-o, --tmpimize(优化表)； //其中，默认选项是-c(检查表)**

**-u， 使用mysql中哪个用户进行操作**

**Mysqlcheck使用语法：**

**使用以下3种方式来调用mysqlcheck：**

**mysqlcheck[tmpions] db\_name [tables]**

**mysqlcheck[tmpions] ---database DB1 [DB2 DB3...]**

**mysqlcheck[tmpions] --all--database**

**如果没有指定任何表或使用---database或--all--database选项，则检查整个数据库。**

**举例说明：**

**1：检查表(check)**

**mysqlcheck -uroot -p123456 -c book books**

****

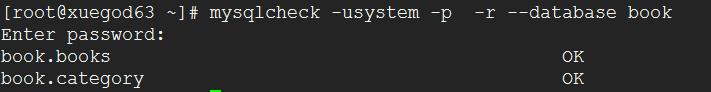
**2:修复表（repair）**

**mysqlcheck -uroot -p123456 -r book books**

****

**3:修复指定的数据库**

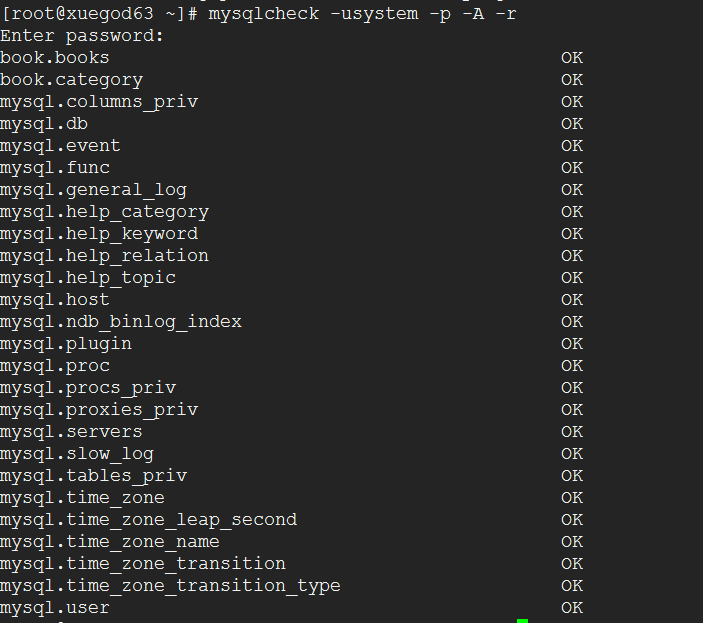
**mysqlcheck -uroot -p -r --database book**

****

**扩展： 修复文件系统。 容易掉失数据**

**fsck -y -f /dev/sdaX**

**4:检查修复所有数据库**

**mysqlcheck -uroot -p -A –r **

**参数-A 等于 --all-databases**

**5：每天定时对mysql数据库进行优化**

**使用计划任务**

**crontab –e**

**#0 1 \* \* \* mysqlcheck -A -o -r -u你的用户名 -p你的密码 > /dev/null 2>&1**

**0 3 \* \* \* mysqlcheck -uroot -p123456 -r -o -A > /dev/null 2>&1 每天3点优化**

## 2：MySQL备份恢复

### 1)MySQL备份的类型

**按照备份时对数据库的影响范围分为：**

**Hot backup(热备) Cold Backup（冷备）Warm Backup（温备）**

**Hot backup：指在数据库运行中直接备份，对正在运行的数据库没有任何影响。（Online Backup）官方手册为在线备份**

**Cold Backup：指在数据库停止的情况下进行备份(OfflineBackup) 官方手册称为离线备份**

**Warm Backup：备份同样在数据库运行时进行，但是会对当前数据库的操作有所影响，例如加一个全局读锁以保证备份数据的一致性**

**按照备份后文件内容：**

**逻辑备份 -->**

**指备份后的文件内容是可读的，通常为文本文件，内容一般是SQL语句，或者是表内的实际数据，如mysqldump和SELECT \* INTO OUTFILE的方法，一般适用于数据库的升级和迁移，恢复时间较长**

**裸文件备份-->**

**拷贝数据库的物理文件，数据库既可以处于运行状态（mysqlhotcopy 、ibbackup、xtrabackup这类工具），也可以处于停止状态，恢复时间较短。**

**按照备份数据库的内容来分，又可以分为：**

**完全备份：对数据库完整的备份**

**增量备份：在上一次完全备份基础上，对更新的数据进行备份（xtrabackup）**

**日志备份：二进制日志备份，主从复制**

### 2)逻辑备份工具mysqldump

**使用的时候MySQL当要导入或者导出数据量大的库的时候,用PHPMYADMIN甚至MySQL Administrator这些工具都会力不从心,这时只能使用MySQL所提供的命令行工具mysqldump进行备份恢复。数据量大的时候不推荐使用，可支持MyISAM,InnoDB**

**MySQL数据的导出和导入工具:mysqldump**

**导出数据：**

**语法： mysqldump [TMPIONS] database [tables] >导出的文件名.sql**

**A:导出所有数据库**

**mysqldump -uroot -p123456 -A >all.sql**

**mysqldump -uroot -p123456 --all-databases >all2.sql**

**参数-A代表所有，等同于—all-databases**

**B:导出某个数据库**

**mysqldump -u 用户名 -p 数据库名 > 导出的文件名.sql # 注意是-p空格后是数据库名，不是密码。**

**mysqldump -uroot -p123456 book >book.sql**

**vim book.sql**

**C:导出单张表**

**mysqldump -uroot -p123456 book books >books.sql #导出book库books表**

**D:导出库的表结构**

**mysqldump -uroot -p123456 -d book>booktable.sql #只导出book库的表结构**

**E：只导出数据**

**mysqldump -uroot -p123456 -t book>bookdata.sql #只导出book库中的数据**

**F:导出数据库，并自动生成库的创建语句**

**mysqldump -uroot -p123456 -B book2 >book2.sql**

**mysql -uroot -p123456 < book2.sql 导入不用指定数据名**

**导入数据：**

**A：导入所有数据库**

**mysql -uroot -p123456 <all.sql**

**B：导入数据库**

**Mysql -uroot -p123456 book <book.sql #如果导入时，没有对应的数据库，需要你手动创建一下：mysql> create database book;**

**使用source导入**

**mysql> create database book;**

**mysql> use book;**

**mysql> source /root/book.sql**

**c:导入表**

**mysql> drop table books;**

**mysql> source /root/books.sql; ##导入表时，不需要重新，创建表。要先进到相应的数据库中**

**mysql> select \* from books;**

**D：导入表结构和数据**

**mysql> create database book;**

**mysql -uroot -p123456 book<booktable.sql**

**mysql -uroot -p123456 book<bookdata.sql**

### 3) mysqlhotcopy-🡪裸文件备份 --5.7版本已经去掉此命令，请在5.5以下测试

**mysqlhotcopy使用lock tables、flush tables和cp或scp来快速备份数据库.它是备份数据库或单个表最快的途径,完全属于物理备份,但只能用于备份MyISAM存储引擎和运行在数据库目录所在的机器上.与mysqldump备份不同,mysqldump属于逻辑备份,备份时是执行的sql语句.**

**Mysqlhotcopy本质是使用锁表语句后再使用cp或scp拷贝数据库**

**1.安装mysqlhotcopy所依赖的软件包(perl-DBD)**

**yum install perl-DBD\* -y**

**2.查看mysqlhotcopy帮助信息：**

**mysqlhotcopy –help**

**--allowold don't abort if target dir already exists (rename it \_old) --不覆盖以前备份的文件**

**--addtodest don't rename target dir if it exists, just add files to it --属于增量备份**

**--noindices don't include full index files in copy --不备份索引文件**

**--debug enable debug --启用调试输出**

**--regexp=# copy all databases with names matching regexp --使用正则表达式**

**--checkpoint=# insert checkpoint entry into specified db.table --插入检查点条目**

**--flushlog flush logs once all tables are locked --所有表锁定后刷新日志**

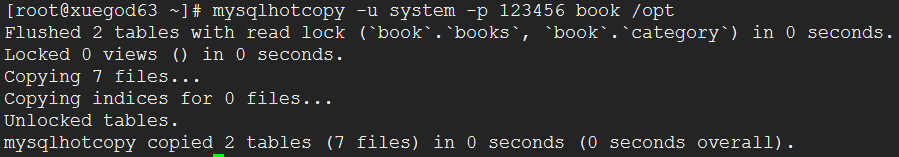
**--resetmaster reset the binlog once all tables are locked --一旦锁表重置binlog文件**

**--resetslave reset the master.info once all tables are locked --一旦锁表重置master.info文件**

**举例说明：**

**A：备份一个数据库到一个目录中**

**mysqlhotcopy -u root -p 123456 book /tmp #-u –p 后面都有空格**

****

****

**对比下大小**

**[root@xuegod70 ~]# du -h /tmp/book/ /var/lib/mysql/book**

**48K /tmp/book/**

**48K /var/lib/mysql/book**

**B：备份多个数据库到一个目录**

**mkdir /tmp/book-mysql**

**mysqlhotcopy -u root -p 123456 book mysql /tmp/book-mysql**

**Flushed 24 tables with read lock (`book`.`books`, `book`.`category`, `mysql`.`columns\_priv`, `mysql`.`db`, `mysql`.`event`, `mysql`.`func`, `mysql`.`help\_category`, `mysql`.`help\_keyword`, `mysql`.`help\_relation`, `mysql`.`help\_topic`, `mysql`.`host`, `mysql`.`ndb\_binlog\_index`, `mysql`.`plugin`, `mysql`.`proc`, `mysql`.`procs\_priv`, `mysql`.`proxies\_priv`, `mysql`.`servers`, `mysql`.`tables\_priv`, `mysql`.`time\_zone`, `mysql`.`time\_zone\_leap\_second`, `mysql`.`time\_zone\_name`, `mysql`.`time\_zone\_transition`, `mysql`.`time\_zone\_transition\_type`, `mysql`.`user`) in 0 seconds.**

**Locked 0 views () in 0 seconds.**

**Copying 7 files...**

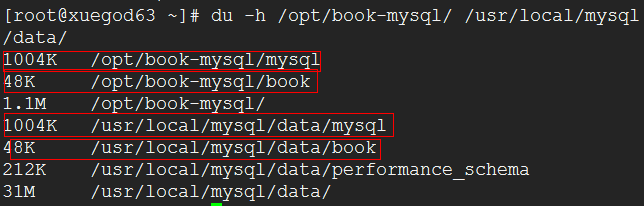
**Copying indices for 0 files...**

**Copying 72 files...**

**Copying indices for 0 files...**

**Unlocked tables.**

**mysqlhotcopy copied 26 tables (79 files) in 0 seconds (0 seconds overall).**

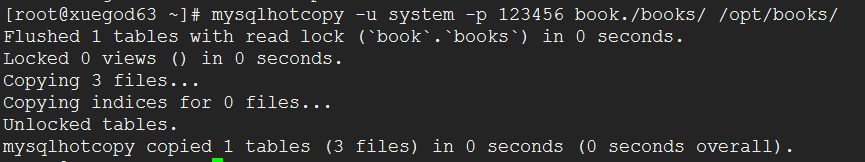
****

**C：备份数据库中的某个表**

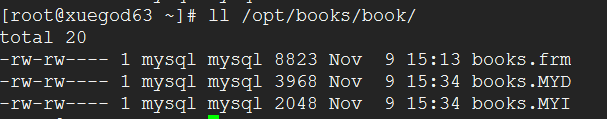
**语法：mysqlhotcopy -u 用户 -p 密码 数据库名./要备份的表名/ 要备份的路径**

**mkdir /tmp/books**

**mysqlhotcopy -u root -p 123456 book./books/ /tmp/books/ #注意 / 一定要**

****

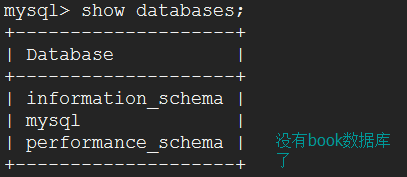
**实际上是把对应的表文件复制到/tmp目录下**

****

**D:恢复数据**

**首先我们破坏数据**

**rm -rf /usr/local/mysql/data/book/**

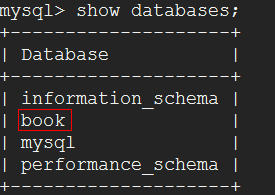
****

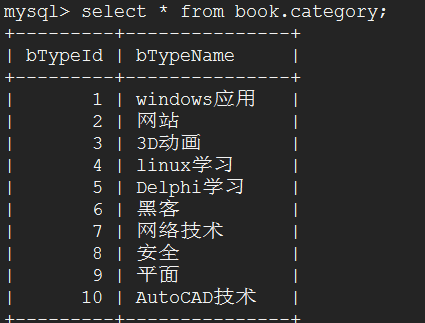
**开始恢复**

**cp -ra /tmp/book/ /usr/local/mysql/data/**

**ll -d /usr/local/mysql/data/book/**

**drwxr-x--- 2 mysql mysql 4096 Nov 9 17:28 /usr/local/mysql/data/book/**

****

****

**总结：**

**mysqldump和mysqlhotcopy的比较：**

**1、mysqldump 是采用SQL级别的备份机制，它将数据表导成 SQL 脚本文件，数据库大时，占用系统资源较多，支持常用的MyISAM，innodb**

**2、mysqlhotcopy只是简单的缓存写入和文件复制的过程，占用资源和备份速度比mysqldump快很多很多。特别适合大的数据库，但需要注意的是：mysqlhotcopy只支持MyISAM 引擎**

**3、mysqlhotcopy只能运行在数据库目录所在的机器上，mysqldump可以用在远程客户端。  
4、相同的地方都是在线执行LOCK TABLES 以及 UNLOCK TABLES  
5、mysqlhotcopy恢复只需要COPY备份文件到源目录覆盖即可，mysqldump需要导入SQL文件到原来库中。**

### 实战：写个自动备份MySQL数据库shell脚本

**Vim mysql-autoback.sh**

**#!/bin/bash**

**export LANG=en\_US.UTF-8**

**savedir=/database\_back/**

**cd "$savedir"**

**time="$(date +"%Y-%m-%d")"**

**mysqldump -u root -p123456 book > book-"$time".sql**

**再添加计划任务就ok**

## xtrabackup备份工具使用

### 1、xtrabackup简介

**我们知道，针对InnoDB存储引擎，MySQL本身没有提供合适的热备工具，ibbackup虽是一款高效的首选热备方式，但它是是收费的。好在Percona公司给大家提供了一个开源、免费的Xtrabackup热备工具，它可实现ibbackup的所有功能，并且还扩展支持真正的增量备份功能，是商业备份工具InnoDB Hotbackup的一个很好的替代品。**

**Xtrabackup包括两个主要工具：Xtrabackup和innobackupex：**

Xtrabackup只能备份InnoDB和XtraDB两种引擎表，而不能备份MyISAM数据表。

innobackupex则封装了xtrabackup，同时可以备份MyISAM数据表。Xtrabackup做备份的时候不能备份表结构、触发器等等，智能区分.idb数据文件。另外innobackupex还不能完全支持增量备份，需要和xtrabackup结合起来实现全备的功能

### 2、xtrbackup 安装 （mysql5.7.20需安装最新版XtraBackup2.4.9）

**下载安装包**

**[root@xuegod70 ~]# wget https://www.percona.com/downloads/XtraBackup/Percona-XtraBackup-2.4.9/binary/redhat/7/x86\_64/Percona-XtraBackup-2.4.9-ra467167cdd4-el7-x86\_64-bundle.tar**

**解压包**

**[root@xuegod70 ~]# tar -xvf Percona-XtraBackup-2.4.9-ra467167cdd4-el7-x86\_64-bundle.tar**

**Yum安装并解决依赖：**

**yum -y install percona-xtrabackup-24-2.4.9-1.el7.x86\_64.rpm**

**注意：安装完成以后记得更改你的/etc/my.cnf配置文件制定数据目录，因为Xtrabackup是根据你的/etc/my.cnf配置文件来获取你备份的文件，比如在/etc/my.cnf的[mysqld] 下添加datadir=/usr/local/mysql/data，然后重启mysql**

**[root@xuegod70 ~]# vim /etc/my.cnf**

**在mysqld的范围添加**

**[mysqld]**

**datadir=/usr/local/mysql/data**

**[root@xuegod70 ~]# systemctl restart mysqld #重启msyql**

### 3、xtrbackup使用

**我们一般使用innobackupex脚本**

**innobackupex是perl脚本对xtrabackup的封装，和功能扩展。**

**备份准备工作：**

**权限和链接**

**xtrabackup需要连接到数据库和datadir操作权限。**

**xtrabackup或者innobackupex在使用过程中涉及到2类用户权限：**

**1.系统用户，用来调用innobackupex或者xtrabackup**

**2.数据库用户，数据库内使用的用户**

**连接到服务**：innobackupex或者xtrabackup通过—user和—password连接到数据库服务

$ innobackupex --user=DBUSER --password=SECRET /path/to/backup/dir/

$ innobackupex --user=LUKE  --password=US3TH3F0RC3 --stream=tar ./ | bzip2 - 压缩

$ xtrabackup --user=DVADER --password=14MY0URF4TH3R --backup --target-dir=/data/bkps/

**其他连接选项：**

**需要的权限**：连接到服务是为了执行备份，需要在datadir上有read，write和execute权限。在数据库中需要以下权限：

RELOAD和LOCK TABLES权限为了执行FLUSH TABLES WITH READ LOCK   。

 REPLICATION CLIENT为了获取binary log 位置

 CREATE TABLESPACE权限为了导入表，用户表级别的恢复

SUPER权限在slave环境下备份用来启动和关闭slave线程

| **Tmpion** | **Description** |
| --- | --- |
| **–port** | **The port to use when connecting to the database server with TCP/IP.** |
| **–socket** | **The socket to use when connecting to the local database.** |
| **–host** | **The host to use when connecting to the database server with TCP/IP.** |

**可以单独创建用来备份数据库的用户，安全，并赋予对应的权限**

### 4、全备和全备还原

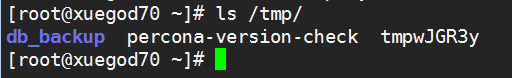
#### 使用innobackupex创建全备

**创建全备：**

**innobackupex --user=root --password=123456 /tmp/db\_backup/**

**innobackupex --user=root --password=123456 /tmp/db\_backup/ 2>>/tmp/db\_backup/db\_backup.log**

**2>>/tmp/db\_backup/db\_backup.log #不显示输出信息，输出信息重定向到db\_backup.log**



**内部机制：在备份的时候innobackupex会调用xtrabackup来备份innodb表，并复制所有的表定义，其他引擎的表(MyISAM,MERGE,CSV,ARCHIVE)。**

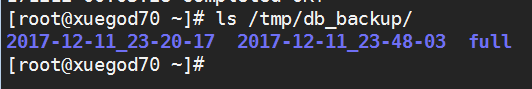
**其他选项:**

**--no-timestamp，指定了这个选项备份会直接备份在BACKUP-DIR，不再创建时间戳文件夹。**

**--default-file，指定配置文件，用来配置innobackupex的选线**

**innobackupex --user=root --password=123456 --no-timestamp /tmp/db\_backup/full**

**innobackupex --user=root --password=123456 --no-timestamp /tmp/db\_backup/full (使用--no-timestamp时，后面的这个full目录必须跟上且不能提前自己建立，它由innobackupex自动建立，否则会报错innobackupex: Error: Failed to create backup directory)**



#### 使用innobackupex还原备份

**使用innobackupex --copy-back来还原备份**

**停止mysql:**

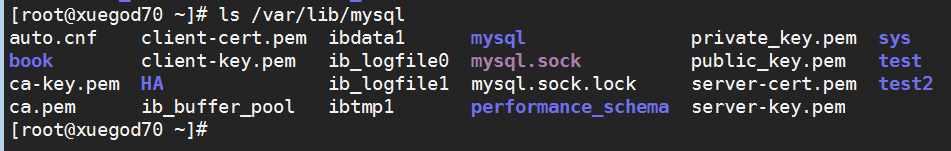
**Systemctl stop mysqld**

**删除数据：（危险动作）：**

**rm -rf /var/lib/mysql/\***

**还原:**

**[root@xuegod70 ~]# innobackupex --copy-back /tmp/db\_backup/2017-12-11\_23-20-17/**



**ll /usr/local/mysql/data/ 查看权限**

**重新授权**

**chown -R mysql:mysql /var/lib/mysql/ 要不然mysql启动不起来**

**systemctl restart mysqld**

**注：datadir必须是为空的，innobackupex –copy-back不会覆盖已存在的文件，还要注意，还原时需要先关闭服务，如果服务是启动的，那么就不能还原到datadir。**

### 5：创建增量备份和还原

**增量备份作用：减少备份数据重复，节省磁盘空间，缩短备份时间**

**增量备份的实现，依赖于innodb页上面的LSN（log sequence number），每次对数据库的修改都会导致LSN自增。增量备份会复制指定LSN<日志序列号>之后的所有数据页。**

#### 创建增量备份

**1：首先创建全备**

**在创建增量备份之前需要一个全备，不然增量备份是没有意义的。**

**innobackupex --user=root --password=123456 /tmp/db\_backup/**

**这样就会在/tmp/db\_backup下创建一个时间戳文件夹 2017-12-11\_23-20-17，文件夹下就是备份文件**

**检查备份文件夹下的xtrabackup-checkpoints，查看信息**

**cat /tmp/db\_backup/2017-12-11\_23-20-17/xtrabackup\_checkpoints**

**backup\_type = full-backuped #全备的意思**

**from\_lsn = 0**

**to\_lsn = 1600459**

**last\_lsn = 16004592：创建第一个增量备份**

**首先我插入一些数据到表里面**

**mysql> use test;**

**mysql> insert into test2 (name) values('HA');**

**2. 使用—incremental创建增量备份**

***innobackupex --user=root --password=123456 --incremental /增量1路径 --incremental-basedir=全备路径 后面指定这哪个全备上进行增量备份***

**[root@xuegod70 mysql]# innobackupex --user=root --password=123456 --incremental /tmp/db\_backup/ --incremental-basedir=/tmp/db\_backup/2017-12-11\_23-20-17/**

**再查看LSN<日志序列号>**

**[root@xuegod70 mysql]# cat /tmp/2017-12-11\_23-58-17/xtrabackup\_checkpoints**

**backup\_type = incremental**

**from\_lsn = 1600459**

**to\_lsn = 1601301**

**last\_lsn = 1601301**

**3：增量备份创建的替代方法**

**可以使用指定—incremental-lsn来代替—incremental-basedir的方法创建增量备份。**

**innobackupex --user=root --password=123456 --incremental /tmp/db\_backup/ --incremental-lsn=1600459 #从这个编号点开始备份**

**注意：xtrabackup只会影响xtradb或者innodb的表，其他引擎的表在增量备份的时候只会复制整个文件，不会差异。**

**测试：**

**停止mysql,删除数据rm –rf /var/lib/mysql/\***

#### 还原增量备份

增量备份的恢复比全备要复杂一点，第一步是在所有备份目录下重做已提交的日志，如：

innobackupex --apply-log --redo-only BASE-DIR

innobackupex --apply-log --redo-only BASE-DIR --incremental-dir=INCREMENTAL-DIR-1

innobackupex --apply-log BASE-DIR --incremental-dir=INCREMENTAL-DIR-2

注意：如果仅有一份增量备份，第2条语句忽然

其中BASE-DIR是指全备目录，INCREMENTAL-DIR-1是指第一次的增量备份，INCREMENTAL-DIR-2是指第二次的增量备份，以此类推。

这里要注意的是：最后一步的增量备份并没有--redo-only选项！

以上语句执行成功之后，最终数据在BASE-DIR（即全备目录）下。

第一步完成之后，我们开始第二步：回滚未完成的日志：

innobackupex --apply-log BASE-DIR

上面执行完之后，BASE-DIR里的备份文件已完全准备就绪，最后一步是拷贝：

innobackupex --copy-back BASE-DIR

恢复mysql权限 chown –R mysql.mysql /var/lib/mysql/

最后启动systemctl start mysqld

检验数据是否恢复正常

### 6：提示

这款工具很强大，更多的地方还要各位自己去摸索

官方有手册可以下载

**官方文档可以参考：**

https://www.percona.com/doc/percona-xtrabackup/2.4/index.html