10.12 Day04

用户授权

Grant 授权

授权 : 添加用户并设置权限 (在服务器上添加新的连接用户)

命令格式

mysql> grant 权限列表 on 库名 to 用户名@”客户端地址”

identifed by “密码” #授权用户密码

with grant option; #有(支持)授权权限 , 可选项

mysql> grant all on db4.\* to yaya@"%" identified by "123qqq...A";

创建一个名为”yaya”用户 , 授权db4库下的所有表所有权限 ,

% : 网络中所有主机 (客户端服务器)

权限列表 常用6种

-all #所有权限

-ueage #无权限

-select(查询) , update(更新修改) , insert(添加) #个别权限

-select , update(字段1,...,字段N) #指定字段

库名

- \*.\* #所有库所有表

-库名.\* #一个库

-库名.表名 #一张表

用户名

-授权时自定义 要有标识性 , 用户信息存储在MySQL库的user表里

客户端地址

-% //所有主机

-192.168.4.% //网段内的所有主机

-192.168.4.1 //1台主机

-localhost //数据库服务器本机

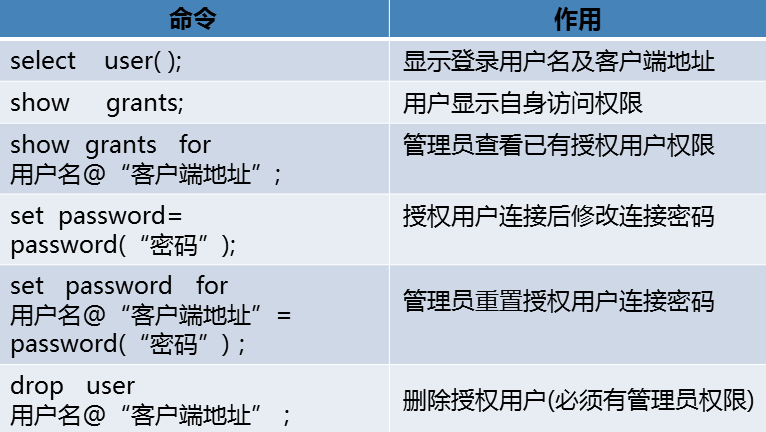
撤销权限 : 删除添加的用户对数据的访问权限

删除用户 : 把新添加的用户删除

授权库 : mysql库 保存新添加的用户信息和权限信息

相关命令

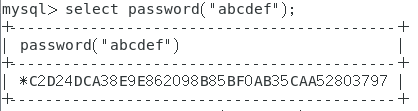
登录用户使用



mysql> show grants for yaya@"%"; #管理员(50)查看授权过的用户与权限

mysql> set password=password("123456"); #客户端(51,授权用户)修改连接密码

注 : password(“”) 为***加密函数*** (原设置密码策略长度最少6位)



mysql> set password for yaya@"%"=password("tarena");

#管理员重置授权用户连接密码

mysql> drop user yaya@"%"; #删除授权用户(必须有管理员权限)

]# which mysql #查看主机上是否有装mysql软件

[root@host51 ~]# rpm -q mariadb ---> 未安装软件包 mariadb

[root@host51 ~]# rpm -qa | grep -i mysql

可由mysql-community-client-5.7.17-1.el7.x86\_64包提供mysql服务

[root@host51 ~]# mysql -h192.168.4.50 -uyaya -p123qqq...A

#连接指定服务器格式 (如 : 51连接50)

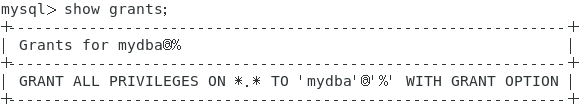
(50)mysql> grant select on \*.\* to root2@"loaclhost" identified by "123456";

注 : 授权root2用户对所有表有查看权限 , 登录密码为123456

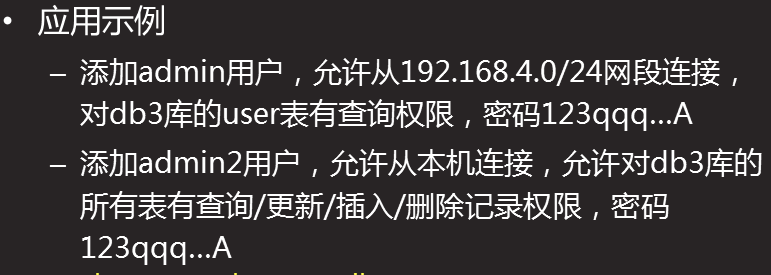
(50)mysql> grant all on \*.\* to mydba@"%" identified by "123qqq...A" with grant option;

注 : 添加用户mydba

51 ~]# mysql -h192.168.4.50 -umydba -p123qqq...A



附 : GRANT PROXY ON~ : 可以把自己权限**复制**给别人



(51)mysql> grant select on db3.user to admin@"192.168.4.%" identified by "123qqq...A";

(51)mysql> grant select,update,insert,delete on db3.\* to admin2@"localhost" identified by "123qqq...A";

授权库

Mysql库 记录授权信息 , 主要表如下 :2

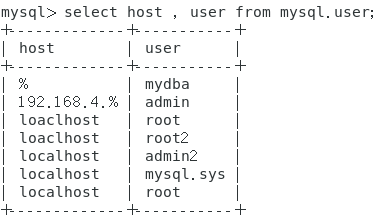
-user表 #记录已有的授权用户及权限

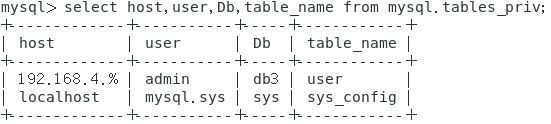
-db表 ~对数据库的访问权限 #只能显示 授权过对库有权限 的用户

-tables\_priv表 ~对表的访问权限 (枚举类型)

-columns\_priv表 ~对字段的访问权限 (授权时间/授权类型)

mysql> select **host**,**user** from mysql.**user**; #**查看所有授权用户**

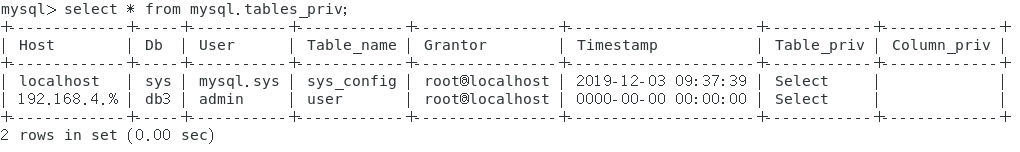




mysql> grant all on \*.\* to mydba@"%" identified by '123qqq...A' with grant option;

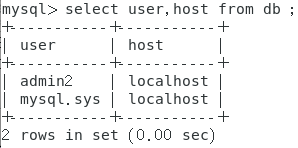
mysql> select \* from mysql.user where user="mydba" and host="%" \G;

mysql> select \* from mysql.tables\_priv;



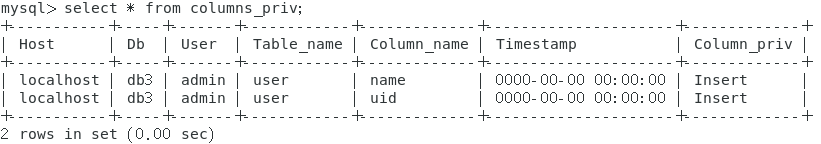
mysql> grant select,update,delete,insert on db3.\* to admin2@"localhost" identified by '123qqq...A';

mysql> select user,host,db from **db** ;



mysql> grant insert(name,uid),select on db3.user to admin@"localhost" identified by "123qqq...A"; #授权用户在特定字段插入功能

mysql> select \* from columns\_priv;



修改/更新已有用户对表的权限=修改表记录

(通过修改表记录修改权限) tables.prive表里有table\_priv字段

mysql> update mysql.tables\_priv set table\_priv="select,insert,delete" where user="admin"; #原权限 select

mysql> flush privileges; #手动刷新权限(即刷新表记录)

mysql> desc mysql.tables\_priv; #**表结构权限位置**

mysql> select \* from mysql.tables\_priv #**表记录权限表**

修改/更新已有用户对数据库的权限

撤销权限

格式 :

mysql> revoke 权限列表 on 库名.表名 from 用户名@”客户端地址”;

mysql> revoke update,delete on db3.\* from admin2@"localhost";

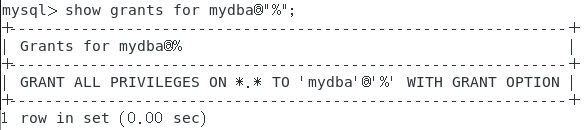
注 : 撤销 admin2 用户的update与delete权限

法1 : mysql> show grants for admin2@"localhost";

# 查看admin2权限

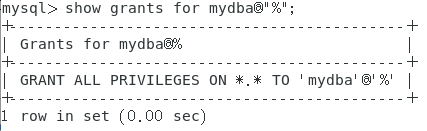
法2 : mysql> select \* from mysql.db where user="admin2" \G;

mysql> show grants for mydba@"%";



mysql> revoke **grant option** on \*.\* from mydba@"%"; #删除用户授权权限

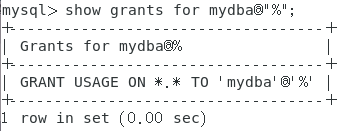
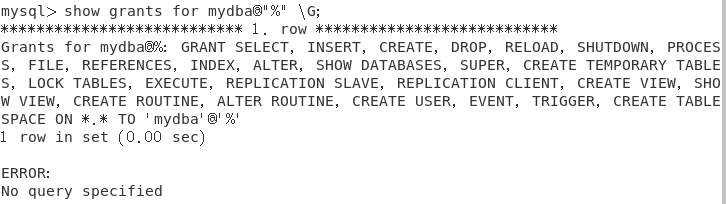
mysql> show grants for mydba@"%";



mysql> revoke **delete,update** on \*.\* from mydba@"%"; #删除用户删除/更新权限

mysql> revoke all on \*.\* from mydba@"%"; #删除所有权限(USAGE)

mysql> show grants for mydba@"%" \G;



root密码 (只有root操作系统管理员才有权限修改)

1.重置root密码

[root@host51 ~]# mysqladmin -hlocalhost -uroot -p123456 password "123aaa...A"

注 : -p123456 为原密码 123aaa...A为新密码

1. 恢复root密码 前提 :　修改配置文件主机用户为root

修改配置文件

host54 ~]# systemctl **stop** mysqld.service #**必须先停服务**

[root@host51 ~]# vim /etc/my.cnf

[mysqld]

skip-grant-tables #添加此行 , **同时注释密码策略** , **否则无法起服务**

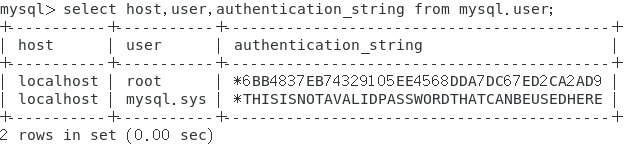
(此处意思为 , 跳过表记录 ,

(可忽略)mysql> show variables like "%password%"; 不显示密码策略设置)

(可忽略)[root@host51 ~]# systemctl restart mysqld #重启服务

mysql> desc mysql.user; ---> **authentication\_string #存放密码字段**

(可忽略)mysql> select host,user,authentication\_string from mysql.user; #查密码



mysql> update mysql.user set authentication\_string=password("123aaa...A") where user="root" and host="localhost";

host54 ~]# systemctl **start** mysqld.service #**启动mysql服务**

注 :

authentication\_string #数据库密码文件(**存放密码字段)**

123aaa...B : 要改回 同样的 密码策略

mysql> flush privileges; #刷新数据库

host51 ~]# vim /etc/my.cnf #删除 密码策略注释 , 再删除/注释skip-grant-tables

host51 ~]# systemctl restart mysqld

备份

1. 数据备份方式

物理备份 (建议数据量**少**时使用)

-冷备 : cp tar ... #停止服务下进行备份热备 (热备是不停服务备份)

逻辑备份 (根据已有数据生成对应的SQL命令 , 并存放到备份时指定的文件里)

(安装其他软件去对数据库上的数据执行备份操作)

(恢复数据时 , 直接去找这些 SQL 命令进行执行操作即可)

(建议数据量**大**时使用)

-mysqldump #备份命令

-mysql #恢复命令

1. 物理备份及恢复

备份操作

-cp -r /var/lib/mysql #备份目录/mysql.bak

-tar -zcvf /root/mysql.tar.gz /var/lib/mysql/\*

物理备份 :

host50 ~]# cp -r /var/lib/mysql /root/mysql.bak #可自定义名字

host50 ~]# ls

host50 ~]# ls /root/mysql.bak/

host50 ~]# scp -r /root/mysql.bak/ root@192.168.4.51:/tmp/

物理恢复 : 192.168.4.51

host51 ~]# cp -r /var/lib/mysql /root/mysql51.bak #此步防止操作失误

host51 ~]# systemctl stop mysqld.service

host51 ~]# *rm -rf /var/lib/mysql*

host51 ~]# cp -r /tmp/mysql.bak /var/lib/mysql

host51 ~]# ls -ld /var/lib/mysql

drwxr-x--x 6 root root 4096 10月 17 20:05 /var/lib/mysql

host51 ~]# chown -R mysql:mysql /var/lib/mysql

host51 ~]# ls /var/lib/mysql

host51 ~]# systemctl start mysqld

host51 ~]# systemctl status mysqld

host51 ~]# mysql -uroot -p123456 #此处用50的密码

逻辑备份

数据备份策略

完全备份 -备份所有数据

增量备份 -备份上次备份后 , 所有新产生的数据

差异备份 -备份完全备份后 , 所有新产生的数据

完全+增量 增量 :

周1 完全 备份10条数据 10 即,周一完全备份10条的数据 为 周二备份的上次

~2 增量 备份3条~ 3 备份数据,周二又产生3条数据 , 共13条数据 , 然

~3 增量 ~6条~ 6 后与周一进行对比 , 所以,周二增量备是备份上次备

~4 增量 ~4条~ 4 份后新产生的数据(即3条) ,周三产生5条 , 所以

~5 增量 13+5共18条数据, 周三与周二对比 ,18-13 ,

周三增量备份即5条

完全+差异

周1 完全 10 10 差异 :

~2 差异 3 3 每次备份都和第一次完全备份去比

~3 差异 6 9 周二产生3条 , 周二与周一(10条)对比 , 备份13条,

19 周三产生6条 , 周三与周一对比 , 备份3+6=9条

~4 差异 4 13 周四产生4条 , 周四与周一对比 , 备份3+6+4=13条

23

**完全备份及恢复**

备份时库名表示方式

- --all-databases 或 **-A** #所有库

- 数据库名 #单个库

-数据库名 表名 #单张表

- -B 数据库1 db2 #多个库

注意事项 :

1. 无论是备份还是恢复 , 都要验证用户权限 ! ! !

]# mysqldump -uroot -p密码 库名 > 目录/xxx.sql

**完全备份** : 50操作

host50 ~]# mkdir /mybak

host50 ~]# ls /mybak

host50 ~]# mysqldump -uroot -p123456 -A > /mybak/all.sql #自定义名称

host50 ~]# vim /mybak/all.sql

host50 ~]# mysqldump -uroot -p123456 db2 > /mybak/db2.sql

host50 ~]# ls /mybak

host50 ~]# scp -r /mybak/db2.sql root@192.168.4.51:/tmp

**完全恢复**

]# mysql -uroot -p密码 **[**库名**]** < 目录/xxx.sql

注 :

1.中括号内的[库名] 表示可选意思

2.运行数据库时 , 删db2库 , 数据库服务数据存在内存里 , 一旦重启mysql就丢失数据

3.用-A(单个库) 或 -B(多个库备份)恢复文件时 , 不需加中括号[]

完全恢复 : 51操作

host51 ~]# mysql -uroot -p123456 db2

mysql> drop database db2; #模拟51 db2数据丢失

mysql> create databases db2;

mysql> exit

host51 ~]# mysql -uroot -p123456 db2 < /tmp/db2.sql #恢复之前要先建库

host51 ~]# mysql -uroot -p123456 < /tmp/all.sql

注 : 若有相同库名 , 会先删除原有同名的文件 , 再用备份的数据进行恢复

===============================================

增量备份

Binlog日志

把mysql上binlog日志启动 , 就是增量备份

日志概述 :

什么是binlog日志

-也称作 二进制日志

-MySQL服务日志文件的一种

-记录除**查询**之外的所有SQL命令 #不记录查询记录(包括desc查表结构,select,show)

-可用于数据备份和恢复

-配置mysql主从同步的必要条件

启用日志



注 :

1.server\_id = 50(主机位的值) 不允许重复 启用日志

2.不指定log\_bin 目录名/文件名 , 就会默认存放在/var/lib/mysql/ ,

文件名并以主机名命名

3.日志文件大小超过1G , 才会自动创建新 #存放位置 /var/lib/mysql/host50-bin.\*

Binlog相关文件

-主机名-bin.index #索引文件

-主机名-bin.000001 #第一个二进制文件

~2 # ~二~

host50 ~]# vim /etc/my.cnf

......

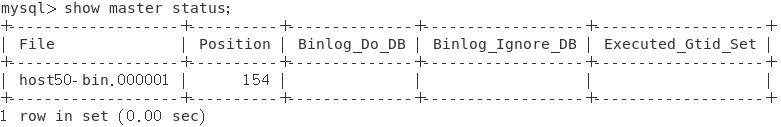
log\_bin #**启用binlog日志**

server\_id=50 #指定id值 (1~255) (非权重)

host50 ~]# systemctl restart mysqld

host50 ~]# mysql -uroot -p123456

mysql> show master status; #查看日志 154表示偏移量(日志初始位置)



host50 ~]# ls /var/lib/mysql #默认存储位置

host50-bin.000001 (二进制内容文件)

host50 ~]# cat /var/lib/mysql/host50-bin.index

./host50-bin.000001 (二进制内容文件)

注 :

通过执行host50-bin.000001进行数据恢复

自定义日志存放位置

host50 ~]# vim /etc/my.cnf

log\_bin=/logdir/abc #新mysql日志文件位置 /logdir

server\_id=50

host50 ~]# chown mysql /logdir/

host50 ~]# systemctl restart mysqld

host50 ~]# mysql -uroot -p123456

mysql> show master status; ----> postion(位置) 154

mysql> create database db5; #建库 db5

mysql> create table db5.t1 (id int); #建表 db1.t1 字段id 类型int

mysql> insert into db5.t1 values(100); #插入表记录 100 200 300

(此时**不**生成日志文件)

mysql> show master status; ---> postion(位置) 898(记录建新表数据)

host50 ~]# systemctl restart mysqld

[root@host50 ~]# ls /logdir/ #重启一次多一个日志文件

abc.000001 abc.000002 abc.index

3.2手动生成新的日志文件

1.host50 ~]# mysqldump -uroot -p123456 --flush-logs db5 > /root/db5.sql

(此时生成 abc.000003) (备份db5数据 , 导出数据)

2.host50 ~]# mysql -uroot -p123456 -e "flush logs"

mysql> flush logs;

(此时生成 abc.000004)

exit

host50 ~]# ls /logdir/

(此时在原来基础上多一个abc.000004二进制日志文件)

(update/insert表记录操作,不会生成新的日志文件,但会记录在二进制日志文件里,超过1G再自动生成新的日志文件)

3.host60 ~]# mysqldump -uroot -p123qqq...A **--flush-logs** -B db5 db6 > /root/3.sql

(备份几个库 , 就创建几个对应的binlog日志文件)

-e : 执行SQL命令

-A : 备份单个库

-B : 备份多个库

3.3清理日志

删除指定编号之前的binlog日志文件

mysql> purge master logs to "abc.000004"; #删除04之前的日志文件(不包括04)

host50 ~]# cat /logdir/abc.index

(已有的文件删除了 , 索引文件也会自动更新)

删除**所有**binlog日志 , 重置日志 , 生成新的索引文件abc.index 与abc.000001

mysql> reset master;

同时清除所有索引文件与binlog日志文件 host50 ~]# cat /logdir/abc.index

host50 ~]# rm -rf /logdir/\* #全删 /logdir/abc.000001

(使用命令单独删除1个文件,如abc.000003 , 索引文件不会更新/变化)

===============================================

3.4使用binlog日志恢复数据

基本思路

-使用mysqlbinlog提前历史SQL命令操作

-通过管道交给mysql命令执行

格式 :

host50 ~]# mysqlbinlog 日志文件 | mysql -uroot -p密码

host60 ~]# scp /logdir/abc.000006 root@192.168.4.61:/root

host61 ~]# mysqlbinlog /root/abc.000006 | mysql -uroot -p123qqq...

mysql> show databases;

准备模拟环境 : (51上db3表记录丢失)

先起服务binlog日志 vim /etc/mysql

host50 ~]# mysqldump -uroot -p123456 db3 > /root/db3.sql #完全备份db3数据

host50 ~]# mysql -uroot -p123456

mysql> reset master; #删除所有binlog日志(仅限练习环境)

mysql> show master status; ---> 154

mysql> create table db3.tea(name char(10)); #建表db3.tea

mysql> insert into db3.tea values ("bob"),(“tom”);

mysql> select \* from db3.tea;

传输 /root/db3.sql 与 /logdir/abc.000001

host50 ~]# scp /root/db3.sql root@192.168.4.51:/root/

host50 ~]# scp /logdir/abc.000001 root@192.168.4.51:/root/

192.168.4.51操作

mysql> drop database db3;

mysql> show databases;

mysql> create database db3;

host51 ~]# mysql -uroot -p123456 db3 < /root/db3.sql

host51 ~]# mysqlbinlog /root/abc.000001 | mysql -uroot -p123456

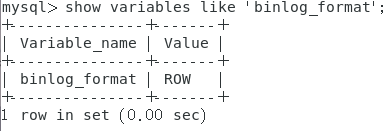
mysql> use db3;

mysql> show tables; --->此时会多出在50上创建的tea表

分析日志

查看日志当前记录格式

mysql> show variables like “binlog\_format”;

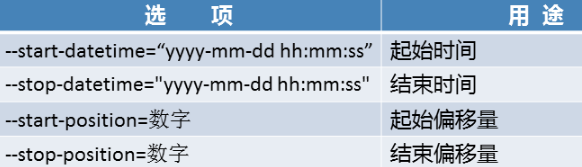


三种记录方式 :

statement #报表模式

**row** #行模式 (默认) (看不到执行SQL命令记录)

mixed #混合模式 (可看到inset/select/update 等SQL命令记录)



50

mysql> insert into db3.tea values (800); ``

mysql> update db3.tea set id=8000;

mysql> show master status; ---> abc.000001 + pos 2912

host50 ~]# scp /logdir/abc.000003 [root@192.168.4.51:/root](mailto:root@192.168.4.51:/root)

mysql> delete from db5.c;

51

host51 ~]# mysqlbinlog /logdir/abc.000003 | grep -i insert

注 :

ERROR 1050 (42S01) at line 30: Table 'tea' already exists

#abc.000003重头读到尾 , 显示报错 , tea表已经存在

host51 ~]# mysqlbinlog /logdir/abc.000003 | mysql -uroot -p123456

#此处恢复数据成功

===============================================

修改日志记录模式

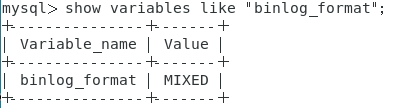
host50 ~]# vim /etc/my.cnf

host50 ~]# mysqlbinlog /logdir/abc.000001 | more #按百分比显示

host50 ~]# vim /etc/my.cnf

binlog\_format="mixed"

host50 ~]# systemctl restart mysqld



Host50 ~]# ls /logdir/

abc.000001 abc.000002 abc.index

host50 ~]# mysqlbinlog /logdir/abc.000001 | grep -i insert

mysql> show master status; ---> abc.000002 + pos 154

mysql> insert into db5.c values (1)~(5); ---> pos 154 ~ 1519

mysql> insert into db5.c values (1);

host50 ~]# mysqlbinlog /logdir/abc.000001 | grep -i insert # 此时有记录

mysql> update db5.c set id= 800 where id=4; #修改操作

mysql> delete from db5.c where id=2; ---> pos 2072 #更新操作

host50 ~]# mysqlbinlog /logdir/abc.000002 | grep -i insert

host50 ~]# mysqlbinlog /logdir/abc.000002 | grep -i update

host50 ~]# mysqlbinlog /logdir/abc.000002 | grep -i delete

注 :

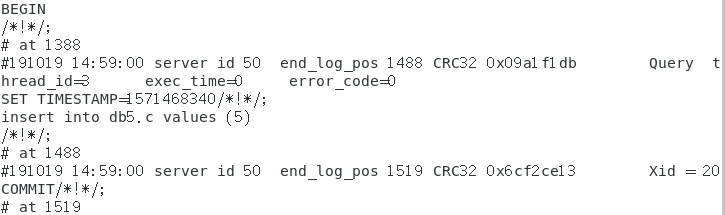
1. 重启服务后多出02文件
2. 日志记录模式改为mixed后 , 不会改变原来的日志文件内容 , 只对修改后新生成的文件有效(02为mixed模式 , 01为row模式)

===============================================

读取binlog日志文件**指定范围内记录的SQL命令**恢复数据

1.使用日志偏移量恢复

host50 ~]# mysqlbinlog /logdir/abc.000002 # mixed模式



注 :

1.commit #回车

2.# at 偏移量(如上1388) #at 后面为偏移量

3.如果要 f 这条命令执行 , 需要包含 commit --> 结束位置 # at 1504 --->

4. 开始位置 a的上方,begin下方 296

51

mysql> delect from db5.c; #此步模拟数据丢失

host51 ~]# mysqlbinlog --**start**-position=296 --stop-position=1519 /logdir/abc.000002 | mysql -uroot -p123456

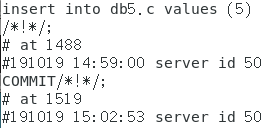
host51 ~]# mysql -uroot -p123456 -e 'select \* from db5.c'

2.通过时间 (起始/结束时间)恢复

host61 ~]# mysql -uroot -p123qqq...A -e 'delete from db5.c'

host61 ~]# mysql -uroot -p123qqq...A -e 'select \* from db5.c'

host61 ~]# mysqlbinlog /root/abc.000002



host61 ~]# mysqlbinlog --**start**-datetime="2019-10-19 14:58:33" --stop-datetime="2019-10-19 14:59:00" /root/abc.000002 | mysql -uroot -p123qqq...A

host61 ~]# mysql -uroot -p123qqq...A -e 'select \* from db5.c'

注 :

若只指定**起始**偏移量 或 **起始**时间 , 默认开始从 **起始**偏移量 或 **起始**时间 读到**末尾**

若只指定**结束**偏移量 或 **结束**时间 , 默认开始从 **头** 读到 **结束**偏移量 或 **结束**时间

===============================================

Day05

MySQL备份工具

常用的MySQL备份工具

物理备份缺点

-跨平台性查

-备份时间长/冗余备份/浪费存储空间

Mysqldump备份缺点

-效率较低/备份和还原速度慢/锁表

-备份过程中 , 数据插入和更新操作被阻塞

(备份需要10分中 , 因为锁表 , 导致无法数据插入与更新)

(锁表 , 是一种MySQL功能)

XtraBackup (可做完全备份 , 差异备份 , 恢复)

一款强大的在热备份工具 (即备份过程中不锁表)

-备份过程中不锁库表 , 适合生产环境

-由专业组织Percona提供(改进MySQL分支)

主要含两个组件

-xtrabackup : C程序 , 支持 InnoDB/XtraDB

-innobackupex : 以Perl脚本封装xtrabakup

安装Percona (在线热备不锁表软件包)

host50 ~]# rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm

host50 ~]# yum -y install percona-xtrabackup-24-2.4.7-1.el7.x86\_64.rpm

(有依赖 10)

查看安装信息

host50 ~]# rpm -ql percona-xtrabackup-24

/usr/bin/innobackupex #备份innodb / xtrdb / myisam**引擎**的表

... ...

/usr/bin/xtrabackup #备份innodb / xtrdb引擎的表

查看命令帮助

host50 ~]# rpm -ql percona-xtrabackup-24

host50 ~]# man innobackupex #详细帮助

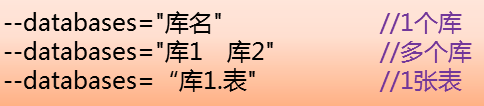
host50 ~]# innobackupex --help #常用选项

innobackupex 命令

常用选项



注 : 不加--databases 默认备份全部数据





完全备份与恢复

完全备份

格式]# innobackupex --user 用户名 --password 密码 备份目录名 --no-timestamp

(不需事先创建)

host50 ~]# innobackupex --user root --password 123456 /allbak --no-timestamp #加--no-timestamp , 不用时间戳做目录 (自动创建)

host50 ~]# ls /allbak/

host50 ~]# scp **-r** /allbak/ root@192.168.4.61:/tmp/

注 :

1. 环境原为空表 ,创建了db5 a表b表,并且各insert了10条数据

完全恢复

格式]# innobackupex --apply-log 目录名 #准备恢复数据

]# innobackupex --copy-back 目录名 #恢复数据

51

host51 ~]# systemctl stop mysqld

host51 ~]# rpm -ivh libev-4.15-1.el6.rf.x86\_64.rpm

host51 ~]# yum -y install percona-xtrabackup-24-2.4.7-1.el7.x86\_64.rpm

host51 ~]# cat /tmp/allbak/xtrabackup\_checkpoints

host51 ~]# innobackupex --apply-log /tmp/allbak/ 准备备份环境

host51 ~]# rm -rf /var/lib/mysql/\* #清空数据库

host51 ~]# innobackupex --copy-back /tmp/allbak/

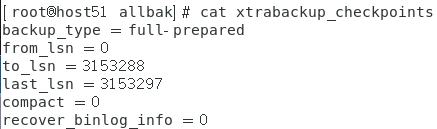
host51 ~]# chown -R **mysql:mysql** /var/lib/mysql #未改权属无法起服务

host51 ~]# systemctl start mysqld

host51 ~]# mysql -uroot -p123456 ---> 此处的密码为原数据库密码

mysql> show databases; ---> 出现db5库

[root@host51 allbak]# cat xtrabackup\_checkpoints



注:

1.记录备份类型+备份信息文件 : xtrabackup\_checkpoints

2.完全备份 :　backup\_type = full-backuped

3.做完准备恢复后变成　：　backup\_type = full-prepared

4.#--copy-back拷贝前要求原mysql目录为空 #要先停服务

===============================================

恢复单张表

准备环境 : 192.168.4.51 :

mysql> delete from b;

mysql> select \* from db5.b;

操作步骤

-删除表空间ibd

-导出表信息cfg

-拷贝表信息文件cfg到数据库目录下

-修改表信息文件cfg的所有者及组用户为mysql

-导入表空间ibd

-删除数据库目录下的表信息文件

-查看表记录

frm**表结构**文件

ibd **表空间**文件 #存储数据的表文件

cfg **表信息**文件

host51 ~]# ls /var/lib/mysql/db5/a3.\* #db5库下的a3表

/var/lib/mysql/db5/a3.frm /var/lib/mysql/db5/a3.ibd

mysql> alter table db5.b discard tablespace; #删除表空间 b.ibd

host51 ~]# innobackupex --apply-log --export /tmp/allbak/ #导出表信息cfg

host51 ~]# cp /tmp/allbak/db5/b.{cfg,exp,ibd} /var/lib/mysql/db5/

#拷贝表信息文件cfg到数据库目录下

host51 ~]# chown mysql:mysql /var/lib/mysql/db5/b.\*

#修改表信息文件cfg的所有者及组用户为mysql

mysql> alter table db5.b import tablespace;

#在登录状态下导入表信息cfg内容 , 导入表空间frm之前前提给权限

(此时.cfg与.exp文件已作废 , 可删除)

mysql> system rm -rf /var/lib/mysql/allbak/db5/b.cfg

mysql> system rm -rf /var/lib/mysql/allbak/db5/b.exp

#删除数据库目录下的表信息文件cfg

若不删除 , 也可以查到此记录 , 但不是数据库格式文件 , 后期对db5做操作时会出问题

===============================================

10.15

增量备份

备份上次备份后 , 所有新增产生的数据

周一 **完全**备份

host50 ~]# innobackupex --user root --password 123456 /fullbak --no-timestamp

mysql> insert into db5.a values (301);

mysql> insert into db5.a values (601);

周二 **增量**备份

host50 ~]# innobackupex --user root --password 123456 --incremental /new1dir --incremental-basedir=/fullbak --no-timestamp

mysql> insert into db5.a values (801);

mysql> insert into db5.a values (608);

指定**增量备份**文件存放位置 : --incremental /new1dir

上次备份文件目录: --incremental-basedir=/fullbak

周三 **增量**备份

50

Innobackupex --user root --password 123456 --incremental /new2dir --incremental-basedir=/new1dir --no-timestamp

mysql> insert into db5.a valuser(501);

mysql> insert into db5.a valuser(708);

注 :

host50 ~]# cat /fullbak/xtrabackup\_checkpoints # 记录了备份信息

backup\_type = full-backuped

from\_lsn = 0 #开始

to\_lsn = 2890275

last\_lsn = 2890284 #结束 lsn 日志序列号 : 记录执行的sql命令的编号

compact = 0

recover\_binlog\_info = 0

事务日志文件 :

host50 ~]# ls /var/lib/mysql #任何对数据库操作都会记录在事务日志文件

ib\_logfile0 ib\_logfile1 与日志序列号对比,数值对不上则数据被改动过

51恢复备份

host51 ~]# systemctl stop mysqld

host51 ~]# rm -rf /var/lib/mysql/\*

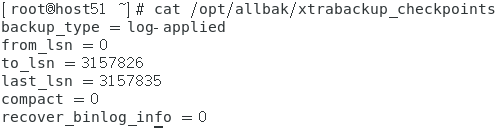
host51 ~]# ls /tmp/ ---> fullbak new1dir new2dir

host51 ~]# innobackupex --apply-log --redo-only /tmp/fullbak/

host51 ~]# innobackupex --apply-log --redo-only /tmp/fullbak/ --incremental-dir=/tmp/new1dir **#redo-only #合并数据new1dir**

host51 ~]# innobackupex --apply-log --redo-only /tmp/fullbak/ --incremental-dir=/tmp/new2dir **#apply-log #准备恢复数据**

host51 ~]# cat /opt/allbak/xtrabackup\_checkpoints



host51 ~]# innobackupex --copy-back /tmp/fullbak/ #拷贝数据

host51 ~]# ls -l /var/lib/mysql

drwxr-x--- 2 root root 290 10月 15 10:00 db1

host51 ~]# chown -R mysql:mysql /var/lib/mysql

host51 ~]# systemctl start mysqld

host51 ~]# mysql -uroot -p123456

mysql> select \* from db5.a;

mysql> select \* from db5.b;

注 :

1.准备恢复数据 : --apply-log

2.日志合并 : --redo-only

3.拷贝数据 : --copy-back

4.增量备份时 , 指定增量备份数据存储的目录名 : --incremental-dir=

5.--increment-basedir #指定上一次备份数据存储的目录名

6.若第一次输入错误 , 要把旧增量备份数据删除 , 再去原数据库复制新的增量备份数据 才可恢复 , 否则一直报错