6.22 数据库备份、恢复

分两种：

完全备份/恢复

Mysqldump命令

增量备份/恢复

1. 启用mysql服务的binlog日志 实现增量备份
2. 安装第3软件，提供innonbackupex命令做增量备份

数据备份的目的：

误操作（管理者、使用者）

硬件损坏 磁盘

物理、逻辑备份都不适合用innodb引擎

数据备份的方式：

物理备份： （innodb不适合物理备份，因为他有事务日志）

直接拷贝库和被对应的系统文件

Cp -rp /var/lib/mysql /opt/mysqlir.bak

拷贝回去时，对应相对接的目录，所有者和所有人是mysql

优点：简单

缺点：占内存大、不适合跨平台

逻辑备份：（生产环境中常用的）

执行备份操作时，根据备份的库表产生对应的sql命令，把sql命令存储到指定文件里（备份的是create、insert，自己创建的字段和插入的命令）

完全备份：备份所有数据

只备份新产成的数据：

差异备份：备份完全备份后所有新产生的数据 （在完全备份后，第二天备份的，在第三天也会把第二天的备份备份一遍）

完全 10 10

差异 2 2

差异 5 7

增量备份：备份上次备份后所有新产生的数据（每天只备份新增加的内容）

完全 10 10

增量 2 2

增量 5 5

生产环境下对数据做备份的方法

Crond 执行 备份脚本 （shell、python、go）

使用脚本对数据作备份要考虑的问题

备份频率

备份策略： 完全+增量 完全+差异

备份时间：（在服务器不忙的时候备份）

备份文件名 日期\_xx.sql

存储空间 lv + raid

搭建mysql主从结构实现数据的自动备份

操作：

完全备份/恢复

Mysqldump命令

备份：

mysqldump 源库名 > 路径 /xxx.sql

恢复：

mysql [目标库名] < 路径 /xxx.sql

源库名的表示

– --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 数据库 2 备份多个库

注意事项

– 无论备份还是恢复,都要验证用户及权限

列：

mysqldump -uroot -p654321 teadb > /aa.sql/teadb.sql

//备份teadb库到/aa.sql/teadb.sql

[root@host50 aa.sql]# mysqldump -uroot -p654321 teadb usertab > /aa.sql/teadb\_user.sql

//备份库中的usertab表到/aa.sql/teadb\_user.sql

[root@host50 ~]# mysqldump -uroot -p654321 teadb > /aa.sql/teadb.sql

//备份teadb库

[root@host50 aa.sql]# mysql -uroot -p654321 teadb < /aa.sql/teadb\_user.sql

//将/aa.sql/teadb\_user.sql 恢复到 teadb库中

[root@host50 aa.sql]# mysql -uroot -p654321 teadb < /aa.sql/teadb.sql s

第二种方法恢复数据

mysql> create databaes tt;

mysql> use tt;

mysql> source /aa.sql/teadb\_user.sql

写脚本,备份库，并指定时间

#!/bin/bash

if [ ! -e /mydatadb1 ];then

mkdir /mydatadb1

fi

day=`date +%F`

mysqldump -uroot -p654321 teadb > /mydatadb1/${day}-db1.sql

[root@host50 mydatadb1]# crontab -l

00 23 \* \* 1 /shell/allbakdb1.sh

增量备份/恢复

1.启用mysql服务的binlog日志 实现增量备份

Binlog二进制日志

记录除查询之外的sql命令

查看日志当前记录格式

mysql> show variables like "binlog\_format";

三种记录格式 :

1. statement :每一条修改数据的 sql 命令都会记录在 binlog 日志中

2. row: 不记录 sql 语句上下文相关信息,仅保存哪条记录被修改

3. mixed: 是以上两种格式的混合使用。

日志默认的内存为500M，超过会在新建一个文件

可修改：log\_bin[=dir/name]

server\_id= 数字

max\_binlog\_size= 数字 m

修改日志记录格式

修改配置文件 /etc/my.conf

[mysqld]

server\_id=50 //用来标示数据库服务器，可从1-254里选择值，默认写本机的ip

log-bin

binlog\_format="mixed"

改完配置文件后，会在/var/lib/mysql/默认目录下生成日志文件，生成两个文件bin.index（记录已有的日志文件名） 和 bin.000001（第一个二进值日志）

使用 mysqlbinlog 工具

格式: mysqlbinlog [ 选项 ] binlog 日志文件名

创建一个库和表，插入或修改，然后查看日志文件

[root@host50 mysql]# mysqlbinlog host50-bin.000001

[root@host50 mysql]# mysqlbinlog host50-bin.000001 | grep -i insert

创建自定义日志目录

修改配置文件

#log-bin

log-bin=/logdir/plj //要创建目录，给权限

测试：

创建库和表，插入或更改

[root@host50 ~]# mysqlbinlog /logdir/plj.000001 | grep insert

Binlog日志文件

偏移量

时间点

执行binlog日志文件里的sql命令恢复数据

Mysqlbinlog 选项 日志文件名 | mysql -uroot -p654321

选项：

--start-position=数值 --stop-position=数值

--start-datetime="yyyy-mm-dd hh:mm:ss

--stop-datetime="yyyy-mm-dd hh:mm:ss"

列：

[root@~]# mysqlbinlog --start-position=616 --stop-position=714 /logdir/plj.000001 | mysql -uroot -p654321

[root@host50 ~]# mysqlbinlog --start-datetime="2018-06-22 15:27:09" --stop-datetime="2018-06-22 15:27:58" /logdir/plj.000001 | mysql -uroot -p654321

//时间不能一样，当时间一样时，他默认会当成没有结束位置

生成新的日志的四种方法

mysql> flush logs；

[root@host50 ~]# mysql -uroot -p654321 -e "flush logs"

[root@host50 logdir]# systemctl restart mysqld //一般不要用

[root@host50 logdir]# mysqldump -uroot -p654321 --flush-logs t2 > /root/t2.sql

删除已有的日志文件

删除早于指定版本的 binlog 日志

mysql> show master status; //查看当前的日志文件

mysql> purge master logs to "plj.000005";

//删除日志文件5以前的

mysql> reset master; //删除所有，从新初始化第一个

//系统默认使用的是最大的日志文件

2.安装第3软件，提供innobackupex命令做增量备份

一款强大的在线热备份工具

开源 备份过程中不锁库表,适合生产环境

Innobackupex 完全/恢复 增量/恢复

1. 安装软件包

安装时，会报错需要安装一些依赖包

[root@host50 ~]# rpm -ivh percona-xtrabackup-24-2.4.7-1.el7.x86\_64.rpm

[root@host50 ~]# yum -y install perl-Digest-MD5.x86\_64 perl-DBD-MySQL.x86\_64

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

[root@host50 ~]# rpm -ivh percona-xtrabackup-24-2.4.7-1.el7.x86\_64.rpm

查看是否安装

[root@host50 ~]# rpm -qa | grep -i percona

[root@host50 ~]# rpm -ql percona-xtrabackup-24

完全备份

在本机登陆是不用加--host --port

[root@host50 ~]# innobackupex --user root --password 654321 /allbak --no-timestamp

//不加库名是完全备份,备份服务器上的所有数据

--no-timestamp //不用日期命名备份文件存储的子目录名

--

[root@host50 ~]# innobackupex --user root --password 654321 --databases="buydb mysql" /towdballbak --no-timestamp

//备份多个库

--databases //数据库名

[root@host50 ~]# innobackupex --user root --password 654321 --databases="buydb" /buydballbak --no-timestamp

//备份库

[root@host50 ~]# innobackupex --user root --password 654321 --databases="buydb.t1" /allbakatab --no-timestamp

//备份库里的表

完全恢复：

恢复时，要求数据库目录是空的

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

重做日志 --apply-log

恢复数据 --copy-back

合并还原 --redo-only

1. 重做日志

查看日志时，日志=full-backuped时，没有重做日志

[root@host50 buydbfull]# cat xtrabackup\_checkpoints

backup\_type = full-backuped

[root@host50 buydbfull]# innobackupex --user root --password 654321 --apply-log /buydbfull

2.清空数据库目录

[root@host50 ~]# rm -rf /var/lib/mysql/\*

[root@host50 ~]# systemctl stop mysqld

3.恢复数据

[root@host50 ~]# innobackupex --user root --password 654321 --copy-back /buydbfull

[root@host50 ~]# chown -R mysql:mysql /var/lib/mysql

[root@host50 ~]# systemctl start mysqld

增量备份/恢复

1. 做首次备份（完全备份）
2. 写数据，做增量备份

[root@host55 ~]# innobackupex --user root --password plj666 --databases="gamedb mysql performance\_schema sys " /bak --no-timestamp //完全备份

//增量备份，在备份之前先要插入新的内容

[root@host55 ~]# innobackupex --user root --password plj666 --databases="gamedb mysql performance\_schema sys" --incremental /new1 --incremental-basedir=/bak --no-timestamp

//指定库，备份到new1目录里，指定上一个备份文件

[root@host55 ~]# innobackupex --user root --password plj666 --databases="gamedb mysql performance\_schema sys" --incremental /new2 --incremental-basedir=/new1 --no-timestamp

//--incremental /new //增量备份

//--incremental-basedir=/fullbuydb //指定上一次备份的文件存储的目录

1. lsn 日志序列号

[root@host55 bak]# cat xtrabackup\_checkpoints

[root@host55 ~]# cat /new1/xtrabackup\_checkpoints

[root@host55 ~]# cat /new2/xtrabackup\_checkpoints

//查看序列号，可以看出他们的排序，第1个从0开始，第二个是从第1个结束的序号开始，依此类推

1. 增量恢复

增量备份目录名 --incremental-dri=

重做并合并 --redo-only

--apply-log 准备恢复数据

1. 重做并合并日志（合并以后，重新增量备份，指定上一次的备份文件是合并以后的文件）

innobackupex --user root --password 654321 \

--databases="gamedb mysql performance\_schema sys" \

--apply-log --redo-only /fullbak

innobackupex --user root --password 654321 \

--databases="gamedb mysql performance\_schema sys" \

--apply-log --redo-only /fullbak --incremental-dir=/new1dir

innobackupex --user root --password 654321 \

--databases="gamedb mysql performance\_schema sys" \

--apply-log --redo-only /fullbak --incremental-dir=/new2dir

/

1. 清空数据库目录

rm -rf /var/lib/mysql

]# mkdir /var/lib/mysql

1. 停止服务
2. 恢复数据，权限

innobackupex --user root --password 654321 \

--databases="gamedb mysql performance\_schema sys" \

--copy-back /fullbak

chown -R mysql:mysql /var/lib/mysql

1. 启动服务

systemctl restart mysqld

使用完全备份文件恢复单个表

完全备份

innobackupex --user root --password 654321 --databases=”buydb” /buydbfull2 --no-timestamp

Drop table buydb.b;

恢复步骤：

从备份数据里导出表信息

innobackupex --user root --password 654321 --databases=”buydb” --apply-log --export /buydbfull2

创建删除的表（表结构要和删除的时候相同）

Create table buyddb.b(name char(10));

删除表空间

Alter table buydb.b discard tablespace;

把导出的表信息文件拷贝到数据库目录下

Cp /buydbfull2/buydb/b.{cfg,exp,ibd} /var/lib/mysql/buydb/

Cd /var/lib/mysql/buydb

Chown mysql:mysql b.\*

导入表空间

Mysql> alter table buydb.b import tablespace;

Rm -rf /var/lib/mysql/buydb/b.{cfg.exp}

查看数据select \* from buydb.b;