**Morning：**

1. 数据库备份

**-数据备份方式**

物理备份：备份数据文件(cp、tar、scp、rsync)

缺陷：跨平台性差；速度慢，浪费存储空间

当只备份数据库文件，会导致日志丢失无法读取数据。

逻辑备份：使用软件提供的命令对数据作备份，如mysqldump、mysql。

执行命令时，将数据存储到备份文件或命令，保存在备份目录下。

**-数据备份策略**

完全备份：备份所有数据（服务器/库/表）

增量备份：参照上一次备份后，所有新产生的数据

差异备份：参照完全备份后，所有新产生的数据

生产环境：每周一完全备份、每天增量/差异备份。

命名格式：2018-11-21\_gamedb.sql

1. 完全备份与恢复

**-命令行语法**

|  |  |
| --- | --- |
| 完全备份 | **mysqldump -uroot -p密码 库名 > 路径/xxx.sql** |
| 完全恢复 | **mysql -uroot -p密码 库名 < 路径/xxx.sql** |
| **库名**表示方式 | -A 或 --all-databases #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 #多个库 |

注意：备份与恢复的用户，需要有相应的权限；

恢复库时，需要先手动创建数据库；

完全恢复库时，已有表记录会按备份的数据进行覆盖，新增的表保留。

缺点：恢复数据时会锁表；完全备份后新增的数据无法恢复；

备份与还原效率低。

**-完全备份案例**

[root@mysql50 ~]#

mysqldump -uroot -p123456 -B db4 db5 > /root/2018-11-21\_dbs.sql

#备份库db4、db5

mysql -uroot -p123456 db4 < /root/2018-11-21\_dbs.sql

mysql -uroot -p123456 db5 < /root/2018-11-21\_dbs.sql

#恢复库

**Afternoon：**

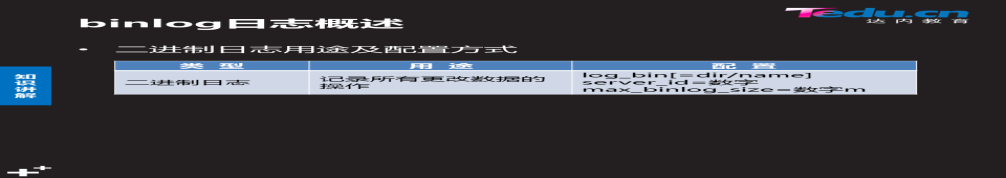
1、binlog日志

**-binlog概述**

记录所有更改数据(insert、update、delete)操作的SQL命令。

自动实现数据实时增量备份，配置mysql主从同步的必要条件。

缺点：数据量大时，恢复效率低



log\_bin[=dir/name]：启用日志[自定义存放位置]

#默认存储在/var/lib/mysql/主机名-bin.00000n

#只指定name，存储在/var/lib/mysql/name.00000n

#xxx.index记录已有的日志文件

#mysql用户必须对存放目录有w权限

server\_id=数字：服务器ID(1-255)

#多台服务器不能重复

max\_binlog\_size=数字m

#默认500m，日志文件大小超过时生成新文件

binlog\_format=”记录格式”

#row（不记录sql命令，仅记录执行后的结果）

#statement（记录sql命令）

#mixed（既记录sql命令，又记录结果）

**-binlog日志操作**

1）启动binlog日志

[root@mysql50 ~]# vim /etc/my.cnf

log\_bin=/logdir/gjq

server\_id=50

binlog\_format="mixed"

[root@mysql50 ~]# mkdir /logdir

[root@mysql50 ~]# chown mysql /logdir

[root@mysql50 ~]# systemctl restart mysqld;

[root@mysql50 ~]# ls /logdir/

gjq.000001 gjq.index

2）查看binlog日志

[root@mysql50 ~]# mysqlbinlog 日志文件名

[root@mysql50 ~]# mysqlbinlog /logdir/gjq.000001 | grep insert

[root@mysql50 ~]# mysqlbinlog /logdir/gjq.000001 | grep delete

[root@mysql50 ~]# mysqlbinlog /logdir/gjq.000001 | grep update

1. 手动生成新的日志文件

1. 重启mysqld服务

2. 执行SQL操作mysql> flush logs;

3. [root@mysql50 ~]# mysql -uroot -p密码 -e ‘flush logs’

4. [root@mysql50 ~]# mysqldump -uroot -p密码 --flush-logs 库名 > 文件名 #做完全备份并生成新的日志文件

Tips：编号最大的日志有效；

每次生成的新日志为空，用于记录该操作后的实时增量备份

1. 删除binlog日志文件

mysql> purge master logs to “gjq.000004”;

#删除日志文件gjq.01~gjq.03

mysql> show master status;

#显示正在使用的binlog文件

mysql> reset master;

#删除所有日志，重建新日志

注意：不建议使用rm删除，不会同步到index文件中

**-分析binlog日志**

1. 日志文件如何记录多条命令？

[root@mysql50 ~]# mysqlbinlog /logdir/gjq.000001

# at 296

#181121 15:32:30 server id 50 end\_log\_pos 404 ...

...

insert into t2 values("bob",1,"aaa")

# at 404

#181121 15:32:30 server id 50 end\_log\_pos 482 ...

...

COMMIT

/\*!\*/;

按偏移量记录：at ～ end\_log\_pos

按时间记录：yymmdd hh:mm:ss

1. 使用mysqlbinlog工具分析日志

**mysqlbinlog [选项] 日志文件名**

常用选项：

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

--stop-datetime=”yyyy-mm-dd hh:mm:ss”

--start-position=数字

--stop-position=数字

**-使用binlog恢复数据**

mysqlbinlog [选项] 日志文件名 | mysql -uroot -p密码

案例：结合完全备份+binlog日志，实现删表后数据恢复

1. 对db4.t2表进行完全备份

[root@mysql50 ~]# mysqldump -uroot -p123456 db4 t2 > /root/gjq.sql

2）增加db4.t2表记录，模拟删表情景

mysql> insert into t2 values("test4",999,"test4");

mysql> drop table db4.t2;

3）执行完全恢复

[root@mysql50 ~]# mysql -uroot -p123456 db4 < /root/gjq.sql

4）执行增量恢复

[root@mysql50 ~]# mysqlbinlog /logdir/gjq.000001 | cat -n | grep "test"

#cat -n可以显示日志行数

[root@mysql50 ~]# mysqlbinlog --stop-position=1490 /logdir/gjq.000001 | mysql -uroot -p123456

2、innobackupex备份/恢复

**-XtraBackup工具**

优点：在线热备，备份过程不锁表，适合生产环境

主要含有两个组件：

-xtrabackup（只支持InnoDB/XtraDB引擎）

-Innobackupex（支持InnoDB/XtraDB/MyISAM），InnoDB/XtraDB可以做增量备份，MyISAM只能做完全备份

**-安装percona-xtrabackup**

[root@mysql50 ~]# rpm -ivh libev-4.15-1...rpm

#安装依赖包

[root@mysql50 ~]# rpm -ivh percona-xtrabackup-24...rpm

[root@mysql50 ~]# rpm -ql percona-xtrabackup-24

/usr/bin/innobackupex #备份innodb、xtradb、myisam的表

...

/usr/bin/xtrabackup #备份innodb、xtradb引擎的表

**-innobackupex命令**

|  |  |
| --- | --- |
| 常用选项 | 含义 |
| --host | 主机名 |
| --user | 用户名 |
| --port | 端口号，不写默认为3306 |
| --password | 密码 |
| --databases | 不写默认为所有数据库  单个库：--databases=”库名”  多个库：--databases=”库1 库2”  单个表：--databases=”库.表” |
| --no-timestamp | 不用日期命名备份文件存储的子目录名 |
| --slave-info | 记录binlog偏移量 |

注意：

1. 恢复前/var/lib/mysql/数据库目录必须清空；因此3个系统库必须备份;

mysql库：存放授权信息

performance\_schema、sys库：存放当前配置、存储过程、视图文件、服务器的额外设置  
2）备份目录：备份数据+配置文件(记录目录下备份数据的备份信息)

3）备份目录无需提前创建



**案例1：完全备份与恢复**

#######完全备份数据库########

innobackupex --user root --password 123456 --no-timestamp --slave-info /allbak

#######完全恢复数据#########

1）准备恢复数据

innobackupex --user root --password 123456 --apply-log /allbak

2）拷贝备份文件到数据库目录下

rm -rf /var/lib/mysql/\* #清空数据库目录

innobackupex --user root --password 123456 --copy-back /allbak

3）修改目录所有者/组为mysql

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

4）重启数据库服务

systemctl restart mysqld

**案例2：增量备份与恢复**

#######增量备份数据库#######

1）首先做完全备份

innobackupex --user root --password 123456 /fullbak --no-timestamp

2）完全备份后添加新数据，作增量备份

mysql> insert into a values(999);

innobackupex --user root --password 123456 --incremental /new1dir --incremental-basedir=/fullbak --no-timestamp

#--incremental指定本次增量备份存放的目录

#--incremental-basedir指定上一次备份目录

#######程序如何检测新数据？#######

./xtrabackup\_checkpoints：记录本次备份数据的备份信息

lsn：日志序列号

Innodb中的事务日志文件是按照lsn来记录不同表的修改

做增量备份时，先检查上一次备份checkpoints配置文件中的to\_lsn，作为本次备份的from\_lsn，然后将该序列号以后的所有指令+内容执行备份。

#######增量恢复数据#######

1. 准备恢复数据

systemctl stop mysqld #停止服务

rm -rf /var/lib/mysql/\* #清空数据库目录

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

cat /fullbak/xtrabackup\_checkpoints

backup\_type=log-applied #状态改变

1. 合并日志文件

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

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

#合并日志的同时，会把增量备份的数据也合并到/fullbak目录

3）拷贝备份文件到数据库目录下

innobackupex --copy-back /fullbak

4）修改目录所有者/组为mysql

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

5）重启数据库服务

systemctl start mysqld

注意：

-停止服务后，innobackupex无需指定--user、--password

-完成增量恢复数据后，可以把之前的增量备份的目录删除，下一次增做量备份指定的目录为/fullbak

**案例3：恢复完全备份中的单个表**

1）对db5做完全备份

innobackupex --user root --password 123456 --databases="db5" /db5bak --no-timestamp

2）删除db5中的表b

mysql> drop table b;

3）使用备份文件恢复删除的表b

mysql> create table b(name char(10));

#创建b表，表结构必须与之前一致

mysql> alter table db5.b discard tablespace;

#删除表空间，相当于删除.idb文件

i[root@mysql50 ~]# innobackupex --user root --password 123456 --databases="db5" --apply-log --export /db5bak

#导出表信息，会在备份目录下多出b.{cfg,exp}

[root@mysql50 ~]# cp /db5bak/db5/b.{cfg,exp,ibd} /var/lib/mysql/db5/

#拷贝表信息、表空间文件到数据库目录

[root@mysql50 ~]# chown mysql:mysql /var/lib/mysql/db5/b.\*

#修改文件所有者/所属组

mysql> alter table db5.b import tablespace;

#导入表空间，相当于让运行的mysqld服务重新识别表文件

[root@mysql50 ~]# rm -rf /var/lib/mysql/db5/b.{cfg,exp}

#恢复表数据后，移除表信息文件

Tips：对于InnoDB引擎，所有数据=数据文件+日志文件