MySQL备份及恢复

一、MySQL备份概述

1、为什么要备份

- 能够防止由于机械故障以及人为误操作带来的数据丢失，例如将数据库文件保存在了其它地方。
- 冗余： 数据有多份冗余，但不等备份，只能防止机械故障还来的数据丢失，例如主备模式、数据库集群。

2、备份必须重视的内容

- 备份内容 databases　Binlog my.conf
- 所有备份数据都应放在非数据库本地，而且建议有多份副本。
- 测试环境中做日常恢复演练，恢复较备份更为重要。

3、备份过程中必须考虑因素：

- 数据的一致性
- 服务的可用性

4、MySQL 备份类型

1、物理备份

对数据库操作系统的物理文件（如数据文件、日志文件等）的备份。物理备份又可分为脱机备份（冷备份）和联机备份（热备份）。这种类型的备份适用于出现问题时需要快速恢复的大型重要数据库。

1、热备(hot backup)

- 在线备份，数据库处于运行状态，这种备份方法依赖于数据库的日志文件
- 对应用基本无影响(应用程序读写不会阻塞,但是性能还是会有下降,所以尽量不要在主上做备份,在从库上做)

2、冷备(cold backup)

- 备份数据文件,需要停机，是在关闭数据库的时候进行的
- 备份 datadir 目录下的所有文件

3、温备(warm backup)

- 针对myisam的备份(myisam不支持热备),备份时候实例只读不可写，数据库锁定表格（不可写入但可读）的状态下进行的
- 对应用影响很大
- 通常加一个读锁

2、逻辑备份

对数据库逻辑组件（如表等数据库对象）的备份，表示为逻辑数据库结构create database、createtable等语句）和内容（insert 语句或分割文本文件）的信息。这种类型的备份适用于可以编辑数据值或表结构较小的数据量，或者在不同机器体系结构上重新创建数据。

3、物理和逻辑备份的区别

-	逻辑备份	物理备份
备份方式	备份数据库逻辑内容	备份数据库物理文件
优点	备份文件相对较小,只备份表中的数据与结构	恢复速度比较快(物理文件恢复基本已经完成恢复)
缺点	恢复速度较慢(需要重建索引,存储过程等)	备份文件相对较大(备份表空间,包含数据与索引,碎片)
对业务影响	缓冲池污染(把所有数据读一遍,读到bp中),I/O负载加大	I/O负载加大
代表工具	mysqldump	ibbackup、xtrabackup，mysqlbackup

4、备份方式的选择

- 从以下几个维度考虑备份方式
    - 备份速度
    - 恢复速度
    - 备份大小
    - 对业务影响

5、MySQL 备份工具

1、ibbackup

- 官方备份工具
- 收费
- 物理备份

2、xtrabackup

- 开源社区备份工具
- 开源免费,上面那东西的免费版本(老版本有问题,备份出来的数据可能有问题)
- 物理备份

3、mysqldump

- 官方自带备份工具 开源免费
- 逻辑备份(速度慢)
- 不阻塞dml,阻塞ddl

4、mysqlbackup

- mysql 官方备份工具
- innodb 引擎的表mysqlbackup可以进行热备
- 非innodb表mysqlbackup就只能温备
- 物理备份，备份还原速度快
- 适合大规模数据使用

6、MySQL 备份策略

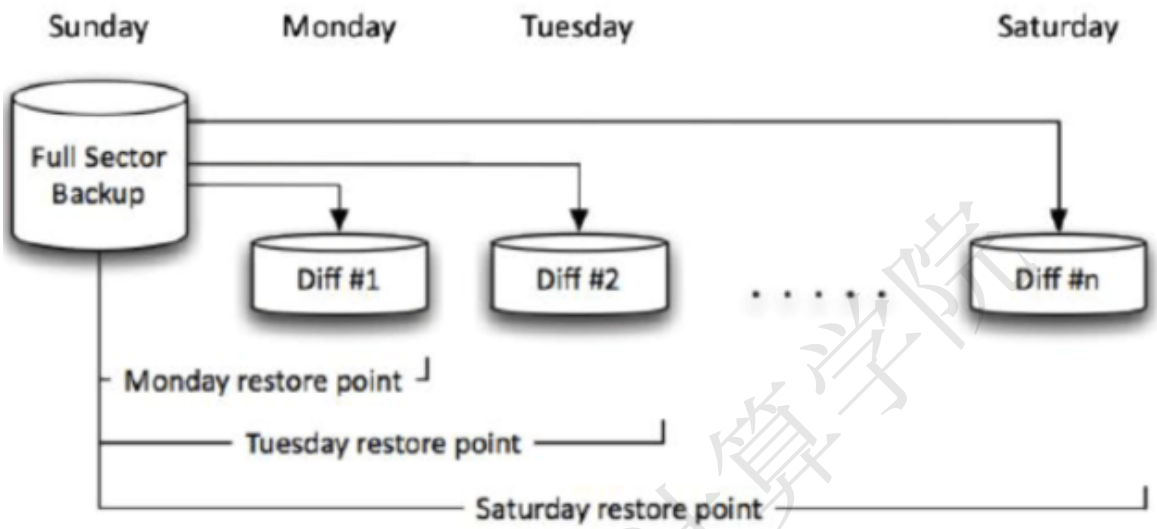
完全备份

每次对数据进行完整的备份，即对整个数据库的备份、数据库结构和文件结构的备份，保存的是备份完成时刻的数据库，是差异备份与增量备份的基础。

优点：备份与恢复操作简单方便. 缺点：数据存在大量的重复；占用大量的空间；备份与恢复时间长。

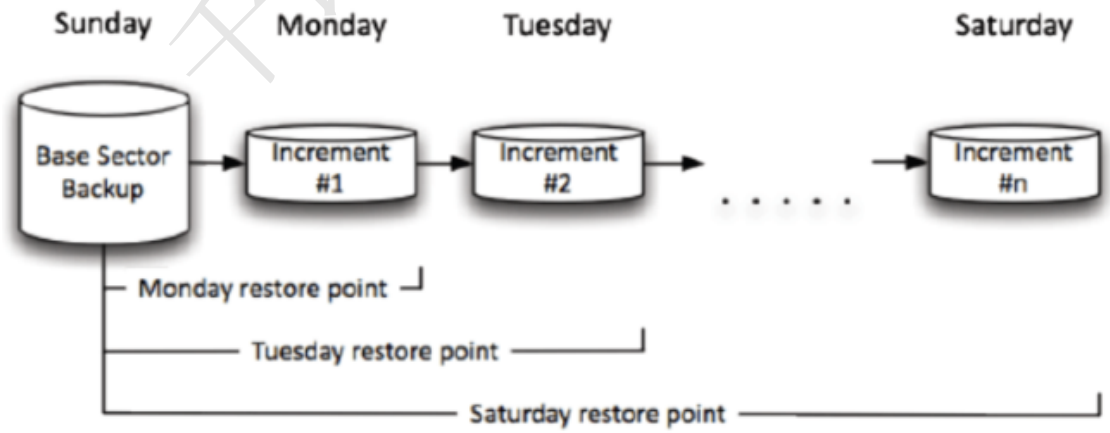
差异备份

备份那些自从上次完全备份之后被修改过的所有文件，备份的时间起点是从上次完整备份起，备份数据量会越来越大。恢复数据时，只需恢复上次的完全备份与最近的一次差异备份。



增量备份

只有那些在上次完全备份或者增量备份后被修改的文件才会被备份。以上次完整备份或上次的增量备份的时间为时间点，仅备份这之间的数据变化，因而备份的数据量小，占用空间小，备份速度快。但恢复时，需要从上一次的完整备份起到最后一次增量备份依次恢复，如中间某次的备份数据损坏，将导致数据的丢失。



二、MySQL逻辑备份mysqldump

1、mysqldump 简介

- mysqldump 是 MySQL 自带的逻辑备份工具。可以保证数据的一致性和服务的可用性。

- 它的备份原理是通过协议连接到 MySQL 数据库，将需要备份的数据查询出来，将查询出的数据转换成对应的 insert 语句，当我们需要还原这些数据时，只要执行这些 insert 语句，即可将对应的数据还原。

2、备份命令

1、命令格式

```
mysqldump [选项] 数据库名 [表名] > 脚本名
```

或

```
mysqldump [选项] --数据库名 [选项 表名] > 脚本名
```

或

```
mysqldump [选项] --all-databases [选项]  > 脚本名
```

2、选项说明

参数名	缩写	含义
--host	-h	服务器IP地址
--port	-P	服务器端口号
--user	-u	MySQL 用户名
--pasword	-p	MySQL 密码
--databases	-B	指定要备份的数据库
--all-databases	-A	备份mysql服务器上的所有数据库
--compact		压缩模式，产生更少的输出
--comments		添加注释信息
--complete-insert		输出完成的插入语句
--lock-tables		备份前，锁定所有数据库表
--no-create-db/--no-create-info		禁止生成创建数据库语句
--force		当出现错误时仍然继续备份操作
--default-character-set		指定默认字符集
--add-locks		备份数据库表时锁定数据库表
-single-transaction		保证数据的一致性和服务的可用性
--master-data=1|2		通常等于1，记录binlog日志位置与文件名，追加至备份文件中
--flush-logs	-F	备份之前刷新日志
--events	-E	备份事件调度器代码
--triggers	-T	备份触发器
--routines	-R	备份存储过程和存储函数

3、备份实例

备份所有数据库：

```
[root@qfedu.com ~]# mysqldump -uroot -p --all-databases > /backup/mysqldump/all.db
```

备份指定数据库：

```
[root@qfedu.com ~]# mysqldump -uroot -p test > /backup/mysqldump/test.db
```

备份指定数据库指定表(多个表以空格间隔)

```
[root@qfedu.com ~]# mysqldump -uroot -p  mysql db event >
/backup/mysqldump/2table.db
```

备份指定数据库排除某些表

```
[root@qfedu.com ~]# mysqldump -uroot -p test --ignore-table=test.t1 --ignore-
table=test.t2 > /backup/mysqldump/test2.db
```

4、还原命令

1、系统行命令

```
[root@qfedu.com ~]# mysqladmin -uroot -p create db_name
[root@qfedu.com ~]# mysql -uroot -p  db_name < /backup/mysqldump/db_name.db
```

- 在导入备份数据库前，db_name如果没有，是需要创建的； 而且与db_name.db中数据库名是一样的才可以导入。

2、soure 方法

```
mysql > use db_name
mysql > source /backup/mysqldump/db_name.db
```

3、MySQL 逻辑备份

1、MySQL 环境

系统版本	mysql版本	安装方式
centos7	5.7.28	YUM安装

2、完整备份与恢复

1、修改配置文件开启二进制日志

```
[root@qfedu.com ~]# vim /etc/my.cnf
[mysqld]
basedir=/soft/mysql
datadir=/soft/mysql/data
default_password_lifetime=0
server-id = 2                    # id是做标识，随便填写
log-bin=/var/log/mysql/bin-log   # 设置二进制日志存放的位置
```

2、创建存放二进制日志文件的目录并赋权限

```
[root@qfedu.com ~]# mkdir -p /var/log/mysql
[root@qfedu.com ~]# chown -R mysql:mysql /var/log/mysql
```

3、创建全量备份文件存放目录并赋权限

```
[root@qfedu.com ~]# mkdir /backup/mysql -p
[root@qfedu.com ~]# chown -R mysql:mysql /backup/mysql/
```

4、重启数据库

```
[root@qfedu.com ~]# systemctl restart mysqld
```

5、进入mysqld创建一个数据库 test1

```
[root@qfedu.com ~]# mysql -uroot -hlocalhost -p'‘Qfedu.123com’'
mysql> create database test1;
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
```

6、进行全量备份

```
[root@localhost ~]# mysqldump -uroot -hlocalhost -p'Qfedu.123com' -P3306 --all-databases --triggers --routines --events --single-transaction --master-data=1 --flush-logs --set-gtid-purged=OFF > /backup/mysql/$(date +%F%H)-mysql-all.sql
```

7、删除数据库文件

```
[root@qfedu.com ~]# systemctl stop mysqld
[root@qfedu.com ~]# rm -rf /var/lib/mysql/*
```

8、向全量备份文件里面追加不记录二进制日志的命令

```
[root@qfedu.com ~]# sed -i '23a SET sql_log_bin=0;' /backup/mysql/2019-11-2810-mysql-all.sql
```

- 向全量备份文件里面追加不记录二进制日志的命令的原因是因为我们在恢复的时候要重新执行一次SQL语句，这个语句没有记录的必要，如果记录的话还可能会导致恢复失败。

9、重启初始化数据库、启动数据库、并修改密码

```
[root@qfedu.com ~]# systemctl restart mysqld
[root@qfedu.com ~]# mysql -uroot -hlocalhost -p'‘Qfedu.123com’'
[root@qfedu.com ~]# grep 'temporary password' /var/log/mysqld.log
[root@qfedu.com ~]# mysql -u root -p'UOln8LE!ue=#'
mysql> alter user 'root'@'localhost' identified by 'Qfedu.123com';
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
```

- 由于这是一个新的数据库，里面只有默认的库，并没有`test1`数据库。

10、导入全备的数据

```
[root@qfedu.com ~]# mysql -u root -p'Qfedu.123com' < /backup/mysql/2019-11-2810-mysql-all.sql
[root@qfedu.com ~]# mysql -uroot -p'Qfedu.123com'
mysql>  set  sql_log_bin=1;
Query OK, 0 rows affected (0.00 se
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| test1              |
+--------------------+
```

- 在数据库内部也可以进行恢复

```
mysql> set  sql_log_bin=0;
mysql> source /backup/mysql/2019-11-2810-mysql-all.sql
mysql>  set  sql_log_bin=1;
Query OK, 0 rows affected (0.00 se
```

- 导入之后当前的密码会不变，当进入数据库 flush privileges 之后，密码又恢复到备份时的密码

```
mysql> flush privileges
```

3、增量备份与恢复

- 备份与恢复环境
- 数据库完整备份+数据库增量备份
- 新建数据表, 进行全量备份, 随着时间推移, 数据库突然奔溃

1、备份之前

```
mysql> create database test2;
mysql> create table test2.t1 (id int,name varchar(20));
mysql> insert into test2.t1 values (1,"test21");
mysql> insert into test2.t1 values (2,"test22");
mysql> select * from test2.t1;
+------+--------+
| id   | name   |
+------+--------+
|    1 | test21 |
|    2 | test22 |
+------+--------+
2 rows in set (0.00 sec)
```

2、基于当前状态做一次全备

```
[root@localhost ~]# mysqldump -uroot -hlocalhost -p'Qfedu.123com' -P3306 --all-databases --triggers --routines --events --single-transaction --master-data=1 --flush-logs --set-gtid-purged=OFF > /backup/mysql/$(date +%F%H)-mysql-all.sql
```

3、进入数据库再插入数据

```
mysql> insert into test2.t1 values (3,"test23");
mysql> insert into test2.t1 values (5,"tt");
mysql> select * from test2.t1;
+------+--------+
| id   | name   |
+------+--------+
|    1 | test21 |
|    2 | test22 |
|    3 | test23 |
|    5 | tt     |
+------+--------+
4 rows in set (0.00 sec)
```

4、模拟数据库崩溃

- 重启初始化，启动数据库,更改默认密码

```
[root@qfedu.com ~]# systemctl stop mysqld
[root@qfedu.com ~]# rm -rf /var/lib/mysql/*
[root@qfedu.com ~]# systemctl start mysqld
[root@qfedu.com ~]# grep 'temporary password' /var/log/mysqld.log
[root@qfedu.com ~]# mysql -u root -p'UOln8LE!ue=#'
mysql> alter user 'root'@'localhost' identified by 'Qfedu.123com';
mysql> \q
[root@qfedu.com ~]# mysql -uroot -p'Qfedu.123com'
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

5、恢复全量数据

```
[root@qfedu.com ~]# sed -i "23aSET sql_log_bin=0;" /backup/mysql/2019-11-2810-mysql-all.sql
[root@qfedu.com ~]# mysql -uroot -p'Qfedu.123com' < /backup/mysql/2019-11-2810-mysql-all.sql
[root@qfedu.com ~]# mysql -u root -p'Qfedu.123com' -e "select * from test2.t1"
+------+--------+
| id   | name   |
+------+--------+
|    1 | test21 |
|    2 | test22 |
+------+--------+
```

6、恢复增量备份

1、获取全备截至点

查看一下全量备份，备份到哪个点了，如下所示是154这个点，000001这个日志文件**

```
[root@qfedu.com ~]# sed -n '22p' /backup/mysql/2019-11-2810-mysql-all.sql
CHANGE MASTER TO MASTER_LOG_FILE='bin-log.000001', MASTER_LOG_POS=154;
```

- 全量仅备份到了154这个点，154后面的点全备文件里面就没有了，需要去000002以后的二进制文件里面找

2、根据 MASTER_LOG_POS 恢复增量的数据

```
[root@qfedu.com mysql]# pwd
/log/mysql
[root@qfedu.com mysql]# mysqlbinlog --start-position=154 bin-log.000001 bin-log.000002 bin-log.000003 bin-log.000003 | mysql -uroot -pQfedu.123com;
[root@mysql02 ~]# mysql -u root -pQfedu.123com -e "select * from test2.t1"
+------+--------+
| id   | name   |
+------+--------+
|    1 | test21 |
|    2 | test22 |
|    3 | test23 |
|    5 | tt     |
+------+--------+
```

4、误操作删除了库（练习）

- 新来的开发删了库，这件事不想再回忆了，以后打死也不会把数据库的 `root` 权限轻易给别人了。今天把当时的场景用虚拟机还原一下，然后复现一下数据恢复的过程，就当是个总结吧！说多了都是泪啊~

1、模拟环境准备

```
[root@qfedu.com ~]# mysql -uroot -p'Qfedu.123com'
mysql> create database test2db;
mysql> use test2db;
mysql> create table t1 (id int,name varchar(20));
mysql> insert into t1 values (1,"ccr");
mysql> insert into t1 values (2,"tfr");
mysql> select * from t1;
+------+------+
| id   | name |
+------+------+
|    1 | ccr  |
|    2 | tfr  |
+------+------+
```

2、全备

```
[root@qfedu.com ~]# mysqldump -uroot -hlocalhost -p'Qfedu.123com' -P3306 --all-databases --triggers --routines --events --single-transaction --master-data=1 --flush-logs --set-gtid-purged=OFF > /backup/mysql/$(date +%F%H)-mysql-all.sql
```

3、再次插入数据

```
[root@qfedu.com ~]# mysql -uroot -p'Qfedu.123com'
mysql> insert into test2db.t1 values(3,'tr1'),(4,'zx'),(5,'wq'),(6,'tj'),
(7,'gwt');
mysql> select * from test2db.t1;
+------+------+
| id   | name |
+------+------+
|    1 | ccr  |
|    2 | tfr  |
|    3 | tr1  |
|    4 | zx   |
|    5 | wq   |
|    6 | tj   |
|    7 | gwt  |
+------+------+
```

4、开发误操作

```
mysql> delete from test2db.t1 where id = '2';
mysql> drop database test2db;
```

5、恢复全备

```
[root@mysql02 ~]# sed -i '23aSET sql_log_bin=0;' /backup/mysql/2019-11-2812-
mysql-all.sql
[root@mysql02 ~]# mysql -u root -p'Qfedu.123com' < /backup/mysql/2019-11-2812-
mysql-all.sql
[root@mysql02 ~]# mysql -u root -p'Qfedu.123com' -e "select * from test2db.t1"
+------+------+
| id   | name |
+------+------+
|    1 | ccr  |
|    2 | tfr  |
+------+------+
```

6、跳过 DELETE 和 DROP 语句

- 下面的操作就要小心翼翼了，不能一下子把二进制日志里面全备以后的操作全部恢复，一旦全部恢复了，那开发删除操作也会恢复，我们只能跳过误操作的地方。

```
[root@qfedu.com ~]# sed -n '22p' /backup/mysql/2019-11-2812-mysql-all.sql
CHANGE MASTER TO MASTER_LOG_FILE='bin_log.000002', MASTER_LOG_POS=154;
[root@qfedu.com ~]# ls /log/mysql/  #全备之后只有一个` bin_log.000002`二进程日志文件
[root@mysql02 ~]# mysql -u root -p'Qfedu.123com'
mysql> show binlog events in 'bin-log.000002';
+----------------+-----+----------------+-----------+-------------+-------------
--------------------------+
| Log_name       | Pos | Event_type     | Server_id | End_log_pos | Info
                         |
+----------------+-----+----------------+-----------+-------------+-------------
--------------------------+
| bin-log.000008 |   4 | Format_desc    |         2 |         123 | Server ver:
5.7.29-log, Binlog ver: 4 |
```

```
| bin-log.000008 | 123 | Previous_gtids |         2 |         154 |
                          |
| bin-log.000008 | 154 | Anonymous_Gtid |         2 |         219 | SET
@@SESSION.GTID_NEXT= 'ANONYMOUS'  |
| bin-log.000008 | 219 | Query          |         2 |         294 | BEGIN
                          |
| bin-log.000008 | 294 | Table_map      |         2 |         345 | table_id:
179 (test2db.t1)            |
| bin-log.000008 | 345 | Write_rows     |         2 |         422 | table_id:
179 flags: STMT_END_F       |
| bin-log.000008 | 422 | Xid            |         2 |         453 | COMMIT /*
xid=980 */                  |
| bin-log.000008 | 453 | Anonymous_Gtid |         2 |         518 | SET
@@SESSION.GTID_NEXT= 'ANONYMOUS'  |
| bin-log.000008 | 518 | Query          |         2 |         593 | BEGIN
                          |
| bin-log.000008 | 593 | Table_map      |         2 |         644 | table_id:
179 (test2db.t1)            |
| bin-log.000008 | 644 | Delete_rows    |         2 |         688 | table_id:
179 flags: STMT_END_F       |
| bin-log.000008 | 688 | Xid            |         2 |         719 | COMMIT /*
xid=982 */                  |
| bin-log.000008 | 719 | Anonymous_Gtid |         2 |         784 | SET
@@SESSION.GTID_NEXT= 'ANONYMOUS'  |
| bin-log.000008 | 784 | Query          |         2 |         885 | drop
database test2db                |
+----------------+-----+----------------+-----------+-------------+-------------
-------------------------+
[root@qfedu.com ~]# mysqlbinlog --start-position=154 --stop-position=453  bin-
log.000002 | mysql -p'Qfedu.1234com'
[root@qfedu.com ~]# mysql -uroot -p'Qfedu.123com' -e "select * from test2db.t1"
+------+------+
| id   | name |
+------+------+
|    1 | ccr  |
|    2 | tfr  |
|    3 | tr1  |
|    4 | zx   |
|    5 | wq   |
|    6 | tj   |
|    7 | gwt  |
+------+------+
```

- **注：上述案例在全备之后仅产生了多个二进制日志文件可进行合并处理**

```
[root@qfedu.com ~]# mysqlbinlog --base64-output="decode-rows" -v bin_log.000001
bin_log.000002 > test3.sql
```

三、MySQL 物理备份 xtrabackup

1、xtrabackup 介绍

Xtrabackup 是一个开源的免费的热备工具，在 Xtrabackup 包中主要有 Xtrabackup 和 innobackupex 两个工具。其中 Xtrabackup 只能备份 InnoDB 和 XtraDB 两种引擎; innobackupex则是封装了Xtrabackup，同时增加了备份MyISAM引擎的功能。

Xtrabackup备份时不能备份表结构、触发器等等，也不能智能区分.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

1、下载安装包

- Xtrabackup 下载地址 https://www.percona.com/downloads/ 选择自己的版本下载。
  - https://www.percona.com/downloads/Percona-XtraBackup-2.4/Percona-XtraBackup-2.4.18/binary/redhat/7/x86_64/Percona-XtraBackup-2.4.18-r29b4ca5-el7-x86_64-bundle.tar
- 依赖包下载，下载地址 http://rpmfind.net/linux/atrpms/el6-x86_64/atrpms/stable/libev-4.04-2.el6.x86_64.rpm。

2、解压安装

```
[root@qfedu.com ~]# ls
libev-4.04-2.el6.x86_64.rpm  Percona-XtraBackup-2.4.14-ref675d4-el7-x86_64-
bundle.tar

[root@qfedu.com ~]# rpm -ivh libev-4.04-2.el6.x86_64.rpm
warning: libev-4.04-2.el6.x86_64.rpm: Header V4 DSA/SHA1 Signature, key ID
66534c2b: NOKEY
Preparing...                          ################################# [100%]
Updating / installing...
   1:libev-4.04-2.el6                 ################################# [100%]

[root@qfedu.com ~]# tar -xf Percona-XtraBackup-2.4.14-ref675d4-el7-x86_64-
bundle.tar
[root@qfedu.com ~]# ls
libev-4.04-2.el6.x86_64.rpm
Percona-XtraBackup-2.4.14-ref675d4-el7-x86_64-bundle.tar
percona-xtrabackup-24-2.4.14-1.el7.x86_64.rpm
percona-xtrabackup-24-debuginfo-2.4.14-1.el7.x86_64.rpm
percona-xtrabackup-test-24-2.4.14-1.el7.x86_64.rpm

[root@qfedu.com ~]# yum -y install percona-xtrabackup-24-2.4.14-
1.el7.x86_64.rpm    # 这里用yum安装，因为还有其他的依赖包
```

3、配置文件

修改配置文件/etc/my.cnf，保证[mysqld]模块存在参数datadir=/var/lib/mysql（指向数据目录），因为xtrbackup是根据/etc/my.cnf配置文件来获取需要备份的文件。

4、重启mysqld。

3、xtrbackup 使用

一般使用的是 innobackupex 脚本，因为 innobackupex 是 perl 脚本对 xtrbackup 的封装和功能扩展。

1、用户权限说明

备份数据库时会涉及到两个用户：系统用户与数据库内部的用户。

1、系统用户

- 需要在 datadir（配置文件内设置的目录）上具有读写执行权限（rwx）。

2、数据库内部用户

- RELOAD 和 LOCK TABLES 权限，执行 FLUSH TABLES WITH READ LOCK；
- REPLICATION CLIENT 权限，获取 binary log（二进制日志文件）位置；
- CREATE TABLESPACE权限，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。

2、常用命令格式和常用参数

1、命令格式：

innobackupex [参数] [目的地址|源地址]

2、常用参数

```
--user                  # 以什么用户身份进行操作
--password              # 数据库用户的密码
--port                  # 数据库的端口号，默认3306
--stream                # 打包（数据流）
--defaults-file         # 指定默认配置文件，默认读取/etc/my.cnf
--no-timestamp          # 不创建时间戳文件，而改用目的地址（可以自动创建）
--copy-back             # 备份还原的主要选项
--incremental           # 使用增量备份，默认使用的完整备份
--incremental-basedir   # 与--incremental选项联合使用，该参数指定上一级备份的地址来做增量备份
```

3、xtrbackup 完整备份和还原

1、完整备份

```
[root@qfedu.com ~]# innobackupex --user=root --password=Qfedu.123com /backup/mysql/   # 有大量输出备份信息
[root@qfedu.com ~]# innobackupex --user=root --password=Qfedu.123com /backup/mysql/ 2>>/backup/mysql/backup.log   # 将备份输出信息保存到文件
[root@qfedu.com ~]# ls /backup/mysql/
2019-06-16_15-49-44  2019-06-16_15-51-23  backup.log
[root@qfedu.com ~]# ls /backup/mysql/2019-06-16_15-49-44/
backup-my.cnf ibdata1 performance_schema xtrabackup_checkpoints
xtrabackup_logfile
ib_buffer_pool mysql sys xtrabackup_info
[root@qfedu.com ~]# innobackupex --user=root --password=Qfedu.123com --no-timestamp /backup/mysql/test/ 2>>/backup/mysql/backup.log   # 不使用时间戳创建目录，可自动创建目的地址
[root@qfedu.com ~]# ls /backup/mysql/
2019-06-16_15-49-44  2019-06-16_15-51-23  backup.log  test
[root@qfedu.com ~]# ls /backup/mysql/test/
2019-06-16_15-54-20
[root@qfedu.com ~]# ls /backup/mysql/test/2019-06-16_15-54-20
backup-my.cnf   ibdata1  performance_schema  xtrabackup_checkpoints
xtrabackup_logfile
ib_buffer_pool  mysql    sys                 xtrabackup_info
```

2、数据还原

- 注意：innobackupex -copy-back不会覆盖已存在的文件。而且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

```
[root@qfedu.com ~]# systemctl stop mysqld
[root@qfedu.com ~]# rm -rf /var/lib/mysql/*                    # 危险操作，请在测试环境测试
[root@qfedu.com ~]# innobackupex --copy-back /backup/mysql/2019-06-16_15-49-44/ 2>>/backup/mysql/copyback.log
[root@qfedu.com ~]# ll /var/lib/mysql
总用量 12324
-rw-r----- 1 root root      292 6月  16 17:08 ib_buffer_pool
-rw-r----- 1 root root 12582912 6月  16 17:08 ibdata1
drwxr-x--- 2 root root     4096 6月  16 17:08 mysql
drwxr-x--- 2 root root     8192 6月  16 17:08 performance_schema
drwxr-x--- 2 root root     8192 6月  16 17:08 sys
-rw-r----- 1 root root      423 6月  16 17:08 xtrabackup_info
[root@qfedu.com ~]# chown -R mysql:mysql /var/lib/mysql    # 重新授权，否则mysqld无法启动
[root@qfedu.com ~]# systemctl start mysqld
[root@qfedu.com ~]# mysql -uroot -pQfedu.123com
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.26 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.00 sec)
```

4、xtrbackup 增量备份的还原

- 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。

1、查看完整备份的LSN

```
[root@qfedu.com ~]# innobackupex --user=root --password=Qfedu.123com
/backup/mysql/

[root@qfedu.com ~]# cat /backup/mysql/2020-03-12_16-45-17/xtrabackup_checkpoints
backup_type = full-backuped                              # 代表完整备份
from_lsn = 0
to_lsn = 2525919
last_lsn = 2525928
compact = 0
recover_binlog_info = 0
flushed_lsn = 2525928
```

2、以全备创建增量备份

创建一些数据，以2020-03-12_16-45-17时间戳创建第一个增量备份，并查看 LSN

```
[root@qfedu.com ~]# mysql -uroot -pQfedu.123com
mysql> create database test_db;
Query OK, 1 row affected (0.00 sec)

mysql> use test_db;
Database changed
mysql> create table user_tb(id int,name varchar(20));
Query OK, 0 rows affected (0.01 sec)

mysql> insert into user_tb values(1,'zhangsan');
Query OK, 1 row affected (0.00 sec)

mysql> exit
Bye


[root@qfedu.com ~]# innobackupex --user=root --password=Qfedu.123com --
incremental /backup/mysql/ --incremental-basedir=/backup/mysql/2020-03-12_16-45-
17/

[root@qfedu.com ~]# innobackupex --user=root --password=Qfedu.123com --
incremental --incremental-basedir=/backup/mysql/2020-03-12_16-45-17/
/backup/mysql/ 2>>/backup/mysql/backup.log


[root@qfedu.com ~]# cat /backup/mysql/2020-03-12_16-48-08/xtrabackup_checkpoints
backup_type = incremental                    # 表示增量备份
from_lsn = 2525919
to_lsn = 2530689
last_lsn = 2530698
compact = 0
recover_binlog_info = 0
flushed_lsn = 2530698
```

3、恢复数据准备全量备份

```
[root@qfedu.com ~]# innobackupex --apply-log --redo-only /backup/mysql/2020-03-
12_16-45-17/ 2>>/backup/mysql/copyback.log
```

4、应用第一次增量备份到全量备份

```
[root@qfedu.com ~]# innobackupex --apply-log --redo-only /backup/mysql/2020-03-12_16-45-17/ --incremental-dir=/backup/mysql/2020-03-12_16-48-08 2>>/backup/mysql/copyback.log
```

5、查看全量备份的 xtrabackup_checkpoints

```
[root@qfedu.com ~]#cat /backup/mysql/2019-06-16_15-49-44/xtrabackup_checkpoints
```

- 对比之前查看的第一次增量备份的 last_lsn 位置，在应用第一次增量备份到全量后，可以看到 last_lsn 已经被应用和第一次全量备份的位置相同了

6、把备份整体进行一次apply操作

```
[root@qfedu.com ~]# innobackupex --apply-log /backup/mysql/2020-03-12_16-45-17/ 2>>/backup/mysql/copyback.log
```

7、停止 mysql 服务并移除数据目录（生产环境建议用 mv ）

```
[root@qfedu.com ~]# systemctl stop mysqld
[root@qfedu.com ~]# rm -rf /var/lib/mysql/*    # 危险操作建议用 mv
```

8、使用--copy-back 参数恢复拷贝到data目录

```
[root@qfedu.com ~]# innobackupex --copy-back /backup/mysql/2020-03-12_16-45-17/ 2>/backup/mysql/copyback.log
```

9、验证数据还原

```
[root@qfedu.com ~]# ls -l /var/lib/mysql
```

10、授权并启动MySQL

```
[root@qfedu.com ~]# chown -R mysql:mysql /var/lib/mysql/
[root@qfedu.com ~]# systemctl start mysqld
```

11、验证数据完整性

```
[root@qfedu.com ~]# mysql -uroot -pQfedu.123com
mysql> use test_db;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> select * from user_tb;
+------+----------+
| id   | name     |
+------+----------+
| 1    | zhangsan |
| 2    | lisi     |
+------+----------+
2 rows in set (0.00 sec)
```

（1）增量备份需要使用参数--incremental指定需要备份到哪个目录，使用incremental-dir指定全备目录；

（2）进行数据备份时，需要使用参数--apply-log redo-only先合并全备数据目录数据，确保全备数据目录数据的一致性；

（3）再将增量备份数据使用参数--incremental-dir合并到全备数据当中；

（4）最后通过最后的全备数据进行恢复数据，注意，如果有多个增量备份，需要逐一合并到全备数据当中，再进行恢复。

5、xtrbackup 数据流压缩

- -stream 选项
- 注意：使用--stream选项时，会输出打包的数据流，并不会直接生成打包文件，此时需要使用重定向或其他命令对数据流进行处理。

1、使用重定向生成压缩文件

- 将标准输出重定向为tar文件，将标准错误重定向到日志文件

```
[root@qfedu.com ~]# innobackupex --databases=test_db --user=root --
password=Qfedu.123com --stream=tar /backup/mysql/> /backup/mysql/`date +%F`.tar
2> /backup/mysql/backup.log
[root@qfedu.com ~]# ls //backup/mysql/
2019-09-29.tar  backup.log
[root@qfedu.com ~]# cd //backup/mysql/
[root@qfedu.com ~]# mkdir test_db.bak
[root@qfedu.com mysql]# tar xf 2019-09-29.tar -C ./test_db.bak      # 注意: 解压时
指定解压目录，压缩中没有归档目录
[root@qfedu.com mysql]# ls test_db.bak
ib_buffer_pool  xtrabackup_binlog_info  xtrabackup_logfile
backup.log      ibdata1                 xtrabackup_checkpoints
backup-my.cnf   test_db                 xtrabackup_info
```

2、使用 ssh 和 cat 组合命令，直接备份到其他服务器上

```
[root@qfedu.com ~]# ssh-keygen
[root@qfedu.com ~]# ssh-copy-id 192.168.5.70
[root@qfedu.com ~]# ssh root@192.168.5.70 "mkdir /backup/mysql"
[root@qfedu.com ~]# innobackupex --databases=test_db --user=root --
password=Qfedu.123com --stream=tar 2>/backup/mysql/backup.log | ssh
root@192.168.5.70 "cat - > /backup/mysql/`date +%F`.tar "
[root@qfedu.com ~]# ssh 192.168.5.70
[root@1000phone mysql]# ls /backup/mysql        # 查看备份文件
2019-09-29.tar
[root@1000phone mysql]# mkdir /backup/mysql/test_db.bak
[root@1000phone ~]# tar xf /2019-09-29.tar -C /backup/mysql/test_db.bak/    # 解
压查看解压结果
[root@1000phone ~]# ls /backup/mysql/test_db.bak
backup-my.cnf   ibdata1  xtrabackup_binlog_info  xtrabackup_info
ib_buffer_pool  test_db  xtrabackup_checkpoints  xtrabackup_logfile
[root@1000phone mysql]# exit
```

3、使用 gzip 再压缩一下

```
[root@qfedu.com ~]# rm -rf /backup/mysql/*
[root@qfedu.com ~]# innobackupex --databases=test_db --user=root --
password=Qfedu.123com --stream=tar /backup/mysql/ 2>/backup/mysql/backup.log |
gzip > /backup/mysql/`date +%F`.tar.gz
[root@qfedu.com ~]# ls /backup/mysql/
2019-09-29.tar.gz  backup.log
[root@qfedu.com ~]# mkdir /backup/mysql/test_db.bak
[root@qfedu.com ~]# tar zxf /backup/mysql/2019-09-29.tar.gz -C
/backup/mysql/test_db.bak
[root@qfedu.com ~]# ls /backup/mysql/test_db.bak        # 解压查看
ib_buffer_pool     xtrabackup_binlog_info  xtrabackup_logfile
backup.log         ibdata1         xtrabackup_checkpoints
backup-my.cnf      test_db         xtrabackup_info
```