# 背景

备份的几个重要理由：

1. 灾难恢复
2. 审计
3. 测试
4. 误操作恢复

# 概述

## frm

与表相关的元数据信息都存放在.frm文件中，主要是表结构的定义信息，不论什么存储引擎，每一个表都会有一个以表名命名的.frm文件。

## .MYD和.MYI

.MYD：MY Data，是MyISAM存储引擎专用的用于存放MyISAM表的数据；

.MYI：MY Index，也是专属于MyISAM存储引擎的主要存放MyISAM表的索引相关信息。

## .ibd和.ibdata

两者都是专属于InnoDB存储引擎的数据库文件。

当采用共享表空间时所有InnoDB表的数据均存放在.ibdata中，所以当表越来越多时，这个文件会变得很大；

相对应的.ibd就是采用独享表空间时InnoDB表的数据文件。

## rbd

# 备份/恢复策略

# 数据库备份

## 背景

在生产环境中我们数据库可能会遭遇各种各样的不测从而导致数据丢失，大概分为以下几种：

1. 硬件故障
2. 软件故障
3. 自然灾害
4. 黑客攻击
5. 误操作 (占比最大)

所以, 为了在数据丢失之后能够恢复数据, 我们就需要定期的备份数据, 备份数据的策略要根据不同的应用场景进行定制, 大致有几个参考数值, 我们可以根据这些数值从而定制符合特定环境中的数据备份策略：

1. 能够容忍丢失多少数据
2. 恢复数据需要多长时间
3. 需要恢复哪一些数据

一般情况下, 我们需要备份的数据分为以下几种：

1、数据

2、二进制日志, InnoDB事务日志

3、代码(存储过程、存储函数、触发器、事件调度器)

4、服务器配置文件

## 分类

### 完全备份/部分备份

根据备份数据集可以分为：

1. 完全备份
2. 部分备份

完全备份指的是备份整个数据集( 即整个数据库 )、部分备份指的是备份部分数据集(例如: 只备份一个表)

#### 增量备份/差异备份

而部分备份又分为以下两种：

1. 增量备份
2. 差异备份

增量备份指的是备份自上一次备份以来(增量或完全)以来变化的数据; 特点: 节约空间、还原麻烦；

差异备份指的是备份自上一次完全备份以来变化的数据 特点: 浪费空间、还原比增量备份简单。

### 冷备/热备/温备

**在MySQl中我们备份数据一般有几种方式：**

热备份：热备份指的是当数据库进行备份时, 数据库的读写操作均不是受影响

温备份：温备份指的是当数据库进行备份时, 数据库的读操作可以执行, 但是不能执行写操作

冷备份：冷备份指的是当数据库进行备份时, 数据库不能进行读写操作, 即数据库要下线

**MySQL中进行不同方式的备份还要考虑存储引擎是否支持：**

MyISAM

热备 ×

温备 √

冷备 √

InnoDB

热备 √

温备 √

冷备 √

### 物理备份/逻辑备份

根据MySQL数据库中数据的备份方式

物理备份一般就是通过tar,cp等命令直接打包复制数据库的数据文件达到备份的效果

逻辑备份一般就是通过特定工具从数据库中导出数据并另存备份(逻辑备份会丢失数据精度)

物理备份

逻辑备份

## 备份策略

针对不同的场景下, 我们应该制定不同的备份策略对数据库进行备份, 一般情况下, 备份策略一般为以下三种：

1、直接cp/tar复制数据库文件

2、mysqldump+复制BIN LOGS

3、lvm2快照+复制BIN LOGS

4、xtrabackup

以上的几种解决方案分别针对于不同的场景：

1. 如果数据量较小, 可以使用第一种方式, 直接复制数据库文件
2. 如果数据量还行, 可以使用第二种方式, 先使用mysqldump对数据库进行完全备份, 然后定期备份BINARY LOG达到增量备份的效果
3. 如果数据量一般, 而又不过分影响业务运行, 可以使用第三种方式, 使用lvm2的快照对数据文件进行备份, 而后定期备份BINARY LOG达到增量备份的效果
4. 如果数据量很大, 而又不过分影响业务运行, 可以使用第四种方式, 使用xtrabackup进行完全备份后, 定期使用xtrabackup进行增量备份或差异备份

## 逻辑备份

逻辑备份工具主要有：mysqldump、mysqlpump、mydumper，物理备份工具主要有：xtrabackup。

**使用mysqldump命令备份**

　　mysqldump命令将数据库中的数据备份成一个文本文件。表的结构和表中的数据将存储在生成的文本文件中。

　　mysqldump先查出需要备份的表的结构，再在文本文件中生成一个CREATE语句。然后，将表中的所有记录转换成一条INSERT语句。然后通过这些语句，就能够创建表并插入数据。

### mysqldump

#### 备份一个数据库

　　mysqldump基本语法：

　　mysqldump -u username -p dbname table1 table2 ...-> BackupName.sql

　　其中：

dbname参数表示数据库的名称；

table1和table2参数表示需要备份的表的名称，为空则整个数据库备份；

BackupName.sql参数表设计备份文件的名称，文件名前面可以加上一个绝对路径。通常将数据库被分成一个后缀名为sql的文件；

　　使用root用户备份test数据库下的person表

mysqldump -u root -p test person > D:\backup.sql

#### 备份多个数据库

语法：

mysqldump -u username -p --databases dbname2 dbname2 > Backup.sql

加上了--databases选项，然后后面跟多个数据库

mysqldump -u root -p --databases test mysql > D:\backup.sql

#### 备份所有数据库

mysqldump命令备份所有数据库的语法如下：

mysqldump -u username -p -all-databases > BackupName.sql

### SELECT INTO OUTFILE

MySQL中，可以使用SELECT...INTO OUTFILE语句将表的内容导出为一个文本文件。其基本的语法格式如下：

SELECT [列名] FROM table [WHERE 语句]

INTO OUTFILE '目标文件' [OPTION];

该语句分为两个部分。前半部分是一个普通的SELECT语句，通过这个SELECT语句来查询所需要的数据；后半部分是导出数据的。其中，“目标文件”参数指出将查询的记录导出到哪个文件中；“OPTION”参数为可选参数选项，其可能的取值有：

FIELDS TERMINATED BY '字符串'：设置字符串为字段之间的分隔符，可以为单个或多个字符。默认值是“\t”。

FIELDS ENCLOSED BY '字符'：设置字符来括住字段的值，只能为单个字符。默认情况下不使用任何符号。

FIELDS OPTIONALLY ENCLOSED BY '字符'：设置字符来括住CHAR、VARCHAR和TEXT等字符型字段。默认情况下不使用任何符号。

FIELDS ESCAPED BY '字符'：设置转义字符，只能为单个字符。默认值为“\”。

LINES STARTING BY '字符串'：设置每行数据开头的字符，可以为单个或多个字符。默认情况下不使用任何字符。

LINES TERMINATED BY '字符串'：设置每行数据结尾的字符，可以为单个或多个字符。默认值是“\n”。

FIELDS和LINES两个子句都是自选的，但是如果两个子句都被指定了，FIELDS必须位于LINES的前面。

### 文件系统快照

## 物理备份

### cp/tar直接复制整个数据库目录

　　MySQL有一种非常简单的备份方法，就是将MySQL中的数据库文件直接复制出来。这是最简单，速度最快的方法。

不过在此之前，要先将服务器停止（冷备），这样才可以保证在复制期间数据库的数据不会发生变化。如果在复制数据库的过程中还有数据写入，就会造成数据不一致。这种情况在开发环境可以，但是在生产环境中很难允许备份服务器。

注意：这种方法不适用于InnoDB存储引擎的表，而对于MyISAM存储引擎的表很方便。同时，还原时MySQL的版本最好相同。

冷备的优点：

1. 备份简单，只要复制相关文件即可；
2. 备份文件易于在不同操作系统，不同MySQL版本上进行恢复；
3. 恢复相当简单，只需要把文件恢复到指定位置即可；
4. 恢复速度非常快，不需要执行任何SQL语句，也不需要重建索引。

冷备的缺点：

1. InnoDB存储引擎冷备的文件通常比逻辑文件大很多，因为表空间存放着需要其他的数据，如undo段，插入缓冲等信息；
2. 冷备也不总是可以轻易地跨平台，操作系统、MySQL版本、文件大小敏感和浮点数格式都会成为问题。

### mysqlhotcopy

　　一看名字就知道是热备份。因此,mysqlhotcopy支持不停止MySQL服务器备份。而且，mysqlhotcopy的备份方式比mysqldump快。mysqlhotcopy是一个perl脚本，主要在Linux系统下使用。其使用LOCK TABLES、FLUSH TABLES和cp来进行快速备份。

　　原理：先将需要备份的数据库加上一个读锁，然后用FLUSH TABLES将内存中的数据写回到硬盘上的数据库，最后，把需要备份的数据库文件复制到目标目录。

命令格式如下：

[root@localhost ~]# mysqlhotcopy [option] dbname1 dbname2 backupDir/

dbname：数据库名称；

backupDir：备份到哪个文件夹下；

**常用选项：**

--help：查看mysqlhotcopy帮助；

--allowold：如果备份目录下存在相同的备份文件，将旧的备份文件加上\_old；

--keepold：如果备份目录下存在相同的备份文件，不删除旧的备份文件，而是将旧的文件更名；

--flushlog：本次备份之后，将对数据库的更新记录到日志中；

--noindices：只备份数据文件，不备份索引文件；

--user=用户名：用来指定用户名，可以用-u代替；

--password=密码：用来指定密码，可以用-p代替。使用-p时，密码与-p之间没有空格；

--port=端口号：用来指定访问端口，可以用-P代替；

--socket=socket文件：用来指定socket文件，可以用-S代替；

mysqlhotcopy并非mysql自带，需要安装Perl的数据库接口包；下载地址为:http://dev.mysql.com/downloads/dbi.html

目前，该工具也仅仅能够备份MyISAM类型的表。

### ibbackup

### Xtrabackup

### MEB

MySQL Enterprise Backup

https://mp.weixin.qq.com/s?\_\_biz=MzU1NzkwMjQ2MQ==&mid=2247485358&idx=1&sn=e5b0386126fd907273cee11e6f6c683d&chksm=fc2ff8bccb5871aafbb1561490a5f151c95f0af855122d636f898d7bdc34104da3bbbe4df427&mpshare=1&scene=24&srcid=&sharer\_sharetime=1591058230737&sharer\_shareid=33f795d236f19ac7c128b2e279563f84#rd

## 备份工具

常用的几种备份工具：

mysqldump：逻辑备份工具, 适用于所有的存储引擎, 支持温备、完全备份、部分备份、对于InnoDB存储引擎支持热备

cp, tar等归档复制工具: 物理备份工具, 适用于所有的存储引擎, 冷备、完全备份、部分备份

lvm2 snapshot：几乎热备, 借助文件系统管理工具进行备份

Mysqlhotcopy：名不副实的的一个工具, 几乎冷备, 仅支持MyISAM存储引擎

Xtrabackup：一款非常强大的InnoDB/XtraDB热备工具, 支持完全备份、增量备份, 由percona提供

# 数据库恢复

## 背景

根据我以往的一些经验来说，通常需要从备份恢复数据的场景有如下几种：

1. 被误删库了
2. 被误删表了，类型为TRUNCATE或者DROP
3. 被误删列了，类型为ALTER ... DROP COLUMN
4. 被误删数据了，类型为DELETE或者UPDATE或者REPLACE
5. 表空间损坏或出现坏块了

根据场景来说，我们可以大致分为两类：

第一类为不可逆恢复，也就是通常的DDL，比如上述的1、2、3、5等场景

第二类为可逆的恢复，通常可以利用binlog进行回滚（要求binlog格式为ROW，binlog\_image为FULL），也就是对应上述的场景4

对于第二类的恢复需求一般来说都比较容易处理，可以利用binlog回滚工具，例如业界比较著名的有binlog2sql以及MyFlash等，这里暂不赘述，我们重点来讨论第一类需求。

为了达到快速恢复的目的，业界DBA经常会采用的方式就是部署一个延迟从库来解决，我们公司目前 所有的核心DB都部署了延迟从库。但是即便有了延迟从库，假设我们错过了延迟的时间，或者在后续利用延迟从库恢复的时候指定错了位点，导致了误删DDL同样应用到了从库，这个时候我们就没有办法利用延迟从库这根救命稻草了。

## 分类

### 全备恢复（异机恢复）

此时，我们只能通过备份来进行数据恢复了。首先我们需要恢复全备，通常来说就是xtrabackup备份的物理备份了。假设你的备份在远程的机器上，那么你可能需要做如下几步动作来进行全备恢复：

1. 将备份scp或者rsync到目标实例机器上
2. 假设备份文件是压缩的情况下，需要解压
3. 解压完成后，需要apply redo log
4. 更改文件权限
5. 假设你直接将文件拷贝到的目标实例的datadir目录下，那么这一步你就可以直接启动mysqld，假设不是，那么你还需要将数据文件move-back或者copy-back到目标实例的datadir
6. 实例启动

### 增备恢复

到这里，全备已经恢复完成了，接下来需要做的就是增量恢复了。按照我们之前的备份方案，我们需要通过binlog来完成增量数据的恢复。对于binlog恢复，我们通常需要以下几个步骤

1. 确定全备对应的binlog位点，也就是需要恢复的起始点
2. 解析主库的binlog，确定误删数据的位点，作为我们恢复的终点
3. 利用mysqlbinlog —start-position —stop-position+管道的方式，将binlog恢复到目标实例上

binlog恢复的方式有很多种，你可以用的是原先master上的binlog，也可以用binlogserver上的binlog，需要做的就是找到binlog恢复的终点即可。

### 增备恢复优化

到这里，你可能会觉得，利用binlog恢复有点麻烦。确实是这样的，利用mysqlbinlog命令并没有办法指定恢复到哪个GTID，只能通过解析binlog，找到需要恢复到的GTID对应的pos位点才行，这对于自动化来说实现起来会比较麻烦。另外，如果利用mysqlbinlog命令恢复，属于单线程恢复，假设需要恢复的binlog量比较多的话，那么这个增量恢复的时间可想而知。

那么有什么办法能加速binlog应用呢？这里我们就想到了MySQL5.7的并行复制，如果我们能用到sql thread的并行复制，是不是这个问题就解决了呢？

### master上binlog恢复

我们回到全备恢复的位点，我们将新实例作为原先的master的slave，然后恢复到指定的GTID位置就可以了呢？没错，这是一种非常简便又轻松还不容易出错的方式，并且还可以利用并行复制的原理来加速binlog应用的目的。但是这种方式的一个要求就是原先的master最老的binlog包含了我们需要的起始恢复位点，这个很容易想到，所以，这将成为我们首选的恢复方式。

### binlogserver上binlog恢复

假设原先master上的binlog已经被purge了，那么我们那需要从binlog上去恢复。有人可能会想到将binlogserver上的binlog拷贝到原先的master上，然后通过修改binlog index来达到注册的目的，实际上这并不可取，具体原因可以见《手动注册binlog文件造成主从异常》。

我们可以采取的方式是什么呢？就是利用binlogserver做成伪装master，然后将从库change上去，其思想就是欺骗slave，让slave的io\_thread将缺失的binlog拉取过来，sql\_thread并行应用binlog event（我们将在下一节具体演示这种方式）。

**优化后的恢复流程**

经过优化以后，我们的增备恢复流程就变成了，首先通过master上的binlog进行恢复，如果发现master上的binlog已经被purge了，那么通过binlogserver上的binlog进行恢复，这样一来我认为是比较科学合理的恢复流程。

各种恢复方式时效性对比



### 业务恢复

到这里，我们已经完成了全量+增量的备份数据恢复，这个时候需要同研发确认数据，确认完成以后将对应的表恢复到原先的master，通常采用的方式有：

mysqldump导出+导入目标实例

表空间传输

## 思路

1. 利用全备的sql文件中记录的CHANGE MASTER语句，binlog文件及其位置点信息，找出binlog文件中增量的那部分。
2. 用mysqlbinlog命令将上述的binlog文件导出为sql文件，并剔除其中的drop语句。
3. 通过全备文件和增量binlog文件的导出sql文件，就可以恢复到完整的数据。

## 逻辑恢复

### mysqldump

还原mysqldump命令备份的数据库，语法如下：

　　mysql -u root -p [dbname] < backup.sq

　　注：这种是采用加载SQL语句的方式实现恢复。

示例：

mysql -u root -p < C:\backup.sql

### mysqladmin

mysqladmin -uroot -p create db\_name

mysql -uroot -p db\_name < /backup/mysqldump/db\_name.db

注：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。

### source

soure方法：

mysql > use db\_name

mysql > source /backup/mysqldump/db\_name.db

### mysqlimport

### LOAD DATA INFILE

如果是通过SELECT INTO OUTFILE导出的符号分割文件，可以使用LOAD DATA INFILE通过相同的参数来加载。也可以使用mysqlimport，这是LOAD DATA INFILE的一个包装。

### alter table xxx import tablespace

将需要备份的数据库表文件.frm和.ibd文件拷贝到对应的目录下，在当前的数据库上执行建表语句，创建一个与待备份表一样的空表。

然后执行释放表空间语句：

alter table xxx drop tablespace; //此时删除data目录下该表对应的ibd文件

然后将对应的ibd文件拷贝到这个目录，接着执行：

alter table xxx import tablespace;

### 基于时间点恢复

### 基于位置恢复

即binlog。

## 物理恢复

### 还原直接复制目录的备份

　　通过这种方式还原时，必须保证两个MySQL数据库的版本号是相同的。MyISAM类型的表有效，对于InnoDB类型的表不可用，InnoDB表的表空间不能直接复制。