# MySQL 定时备份数据库（非常全）

letcafe java学习 今天

在操作数据过程中，可能会导致数据错误，甚至数据库奔溃，而有效的定时备份能很好地保护数据库。本篇文章主要讲述了几种方法进行 MySQL 定时备份数据库。

## 一. mysqldump命令备份数据

在MySQL中提供了命令行导出数据库数据以及文件的一种方便的工具**mysqldump**,我们可以通过命令行直接实现数据库内容的导出dump,首先我们简单了解一下mysqldump命令用法:

```
#MySQLdump常用
mysqldump -u root -p --databases 数据库1 数据库2 > xxx.sql
```

## 二. mysqldump常用操作示例

1. 备份全部数据库的数据和结构

```
mysqldump -uroot -p123456 -A > /data/mysqlDump/mydb.sql
```

2.备份全部数据库的结构（加 -d 参数）

```
mysqldump -uroot -p123456 -A -d > /data/mysqlDump/mydb.sql
```

3. 备份全部数据库的数据(加 -t 参数)

```
mysqldump -uroot -p123456 -A -t > /data/mysqlDump/mydb.sql
```

4.备份单个数据库的数据和结构(,数据库名mydb)

```
mysqldump -uroot-p123456 mydb > /data/mysqlDump/mydb.sql
```

5. 备份单个数据库的结构

```
mysqldump -uroot -p123456 mydb -d > /data/mysqlDump/mydb.sql
```

6. 备份单个数据库的数据

```
mysqldump -uroot -p123456 mydb -t > /data/mysqlDump/mydb.sql
```

7. 备份多个表的数据和结构（数据，结构的单独备份方法与上同）

```
mysqldump -uroot -p123456 mydb t1 t2 > /data/mysqlDump/mydb.sql
```

8. 一次备份多个数据库

```
mysqldump -uroot -p123456 --databases db1 db2 > /data/mysqlDump/mydb.sql
```

## 三. 还原 MySQL 备份内容

有两种方式还原，第一种是在 MySQL 命令行中，第二种是使用 SHELL 行完成还原

1. 在系统命令行中，输入如下实现还原：

```
mysql -uroot -p123456 < /data/mysqlDump/mydb.sql
```

2. 在登录进入mysql系统中,通过source指令找到对应系统中的文件进行还原：

```
mysql> source /data/mysqlDump/mydb.sql
```

在 Linux中，通常使用**BASH**脚本对需要执行的内容进行编写，加上定时执行命令**crontab**实现日志自动化生成。

以下代码功能就是针对mysql进行备份，配合crontab，实现备份的内容为近一个月（31天）内的每天的mysql数据库记录。

**编写BASH维护固定数量备份文件**

在Linux中，使用vi或者vim编写脚本内容并命名为：mysql_dump_script.sh

```
#!/bin/bash

#保存备份个数，备份31天数据
number=31
#备份保存路径
backup_dir=/root/mysqlbackup
#日期
dd=`date +%Y-%m-%d-%H-%M-%S`
#备份工具
tool=mysqldump
```

```
#用户名
username=root
#密码
password=TankB214
#将要备份的数据库
database_name=edoctor

#如果文件夹不存在则创建
if [ ! -d $backup_dir ];
then
    mkdir -p $backup_dir;
fi

#简单写法 mysqldump -u root -p123456 users > /root/mysqlbackup/users-$filename.sql
$tool -u $username -p$password $database_name > $backup_dir/$database_name-$dd.sql

#写创建备份日志
echo "create $backup_dir/$database_name-$dd.dupm" >> $backup_dir/log.txt

#找出需要删除的备份
delfile=`ls -l -crt $backup_dir/*.sql | awk '{print $9 }' | head -1`

#判断现在的备份数量是否大于$number
count=`ls -l -crt $backup_dir/*.sql | awk '{print $9 }' | wc -l`

if [ $count -gt $number ]
then
  #删除最早生成的备份，只保留number数量的备份
  rm $delfile
  #写删除文件日志
```

```
  echo "delete $delfile" >> $backup_dir/log.txt
fi
```

如上代码主要含义如下：

1.首先设置各项参数，例如number最多需要备份的数目，备份路径，用户名，密码等。

2.执行mysqldump命令保存备份文件，并将操作打印至同目录下的log.txt中标记操作日志。

3.定义需要删除的文件：通过ls命令获取第九列，即文件名列，再通过实现定义操作时间最晚的那个需要删除的文件。

4.定义备份数量：通过ls命令加上

统计以sql结尾的文件的行数。

5.如果文件超出限制大小，就删除最早创建的sql文件

**使用crontab定期执行备份脚本**

在 Linux 中，周期执行的任务一般由cron这个守护进程来处理[ps -ef|grep cron]。cron读取一个或多个配置文件，这些配置文件中包含了命令行及其调用时间。
cron的配置文件称为“crontab”，是“cron table”的简写。

**cron服务**

cron是一个 Liunx 下 的定时执行工具，可以在无需人工干预的情况下运行作业。

service crond start　//启动服务

service crond stop　//关闭服务

service crond restart　//重启服务

service crond reload　//重新载入配置

service crond status　//查看服务状态

**crontab语法**

crontab命令用于安装、删除或者列出用于驱动cron后台进程的表格。用户把需要执行的命令序列放到crontab文件中以获得执行。每个用户都可以有自己的crontab文件。/var/spool/cron下的crontab文件不可以直接创建或者直接修改。该crontab文件是通过crontab命令创建的。

在crontab文件中如何输入需要执行的命令和时间。该文件中每行都包括六个域，其中前五个域是指定命令被执行的时间，最后一个域是要被执行的命令。
每个域之间使用空格或者制表符分隔。

格式如下:
minute hour day-of-month month-of-year day-of-week commands
合法值 00-59 00-23 01-31 01-12 0-6 (0 is sunday)

除了数字还有几个个特殊的符号就是"*"、"/"和"-"、","，*代表所有的取值范围内的数字，"/"代表每的意思,"/5"表示每5个单位，"-"代表从某个数字到某个数字,","分开几个离散的数字。

-l 在标准输出上显示当前的crontab。
-r 删除当前的crontab文件。

-e 使用VISUAL或者EDITOR环境变量所指的编辑器编辑当前的crontab文件。当结束编辑离开时，编辑后的文件将自动安装。

**创建cron脚本**

第一步：写cron脚本文件,命名为mysqlRollBack.cron。

15,30,45,59 * * * * echo "xgmtest....." >> xgmtest.txt 表示，每隔15分钟，执行打印一次命令

第二步：添加定时任务。执行命令“crontab crontest.cron”。搞定

第三步："crontab -l" 查看定时任务是否成功或者检测/var/spool/cron下是否生成对应cron脚本

注意：这操作是直接替换该用户下的crontab，而不是新增

定期执行编写的定时任务脚本（记得先给shell脚本执行权限）

```
0 2 * * * /root/mysql_backup_script.sh
```

随后使用crontab命令定期指令编写的定时脚本

```
crontab mysqlRollback.cron
```

再通过命令检查定时任务是否已创建：

附 crontab 的使用示例：

1. 每天早上6点

```
0 6 * * * echo "Good morning." >> /tmp/test.txt //注意单纯echo，从屏幕上看不到任何输出，因为cron把任何输出都email到root的信箱了。
```

2. 每两个小时

```
0 */2 * * * echo "Have a break now." >> /tmp/test.txt
```

3. 晚上11点到早上8点之间每两个小时和早上八点

```
0 23-7/2，8 * * * echo "Have a good dream" >> /tmp/test.txt
```

4. 每个月的4号和每个礼拜的礼拜一到礼拜三的早上11点

```
0 11 4 * 1-3 command line
```

5.1 月 1 日早上 4 点

```
0 4 1 1 * command line SHELL=/bin/bash PATH=/sbin:/bin:/usr/sbin:/usr/bin MAILTO=root //如果出现错误，或者有数据输出，数据作为邮件发给这个
```

6. 每小时执行/etc/cron.hourly内的脚本

```
01 * * * * root run-parts /etc/cron.hourly
```

7. 每天执行/etc/cron.daily内的脚本

```
02 4 * * * root run-parts /etc/cron.daily
```

8. 每星期执行/etc/cron.weekly内的脚本

```
22 4 * * 0 root run-parts /etc/cron.weekly
```

9. 每月去执行/etc/cron.monthly内的脚本

```
42 4 1 * * root run-parts /etc/cron.monthly
```

注意: "run-parts" 这个参数了，如果去掉这个参数的话，后面就可以写要运行的某个脚本名，而不是文件夹名。

10. 每天的下午4点、5点、6点的5 min、15 min、25 min、35 min、45 min、55 min时执行命令。

```
5，15，25，35，45，55 16，17，18 * * * command
```

11. 每周一，三，五的下午3：00系统进入维护状态，重新启动系统。

```
00 15 * * 1，3，5 shutdown -r +5
```

12. 每小时的10分，40分执行用户目录下的innd/bbslin这个指令：

```
10，40 * * * * innd/bbslink
```

13. 每小时的1分执行用户目录下的bin/account这个指令：

以下是我的测试每分钟的截图效果，其对应代码如下：

```
* * * * * /root/mysql_backup_script.sh
```

效果截图：

```
-rw-r--r-- 1 root root 259114 Mar  9 19:57 edoctor-2018-03-09-19-57-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 19:58 edoctor-2018-03-09-19-58-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 19:59 edoctor-2018-03-09-19-59-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:00 edoctor-2018-03-09-20-00-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:01 edoctor-2018-03-09-20-01-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:02 edoctor-2018-03-09-20-02-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:03 edoctor-2018-03-09-20-03-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:04 edoctor-2018-03-09-20-04-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:05 edoctor-2018-03-09-20-05-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:06 edoctor-2018-03-09-20-06-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:07 edoctor-2018-03-09-20-07-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:08 edoctor-2018-03-09-20-08-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:09 edoctor-2018-03-09-20-09-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:10 edoctor-2018-03-09-20-10-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:11 edoctor-2018-03-09-20-11-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:12 edoctor-2018-03-09-20-12-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:13 edoctor-2018-03-09-20-13-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:14 edoctor-2018-03-09-20-14-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:15 edoctor-2018-03-09-20-15-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:16 edoctor-2018-03-09-20-16-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:17 edoctor-2018-03-09-20-17-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:18 edoctor-2018-03-09-20-18-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:19 edoctor-2018-03-09-20-19-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:20 edoctor-2018-03-09-20-20-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:21 edoctor-2018-03-09-20-21-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:22 edoctor-2018-03-09-20-22-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:23 edoctor-2018-03-09-20-23-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:24 edoctor-2018-03-09-20-24-02.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:25 edoctor-2018-03-09-20-25-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:26 edoctor-2018-03-09-20-26-01.sql
-rw-r--r-- 1 root root 259114 Mar  9 20:27 edoctor-2018-03-09-20-27-01.sql
-rw-r--r-- 1 root root  12572 Mar  9 20:27 log.txt
```

其中的log.txt记录备份的操作详细日志：

```
delete /root/mysqlbackup/edoctor-2018-03-09-19-58-01.sql
```

本文参考：

```
1.MySQLdump常用命令

www.cnblogs.com/smail-bao/p/6402265.html

2.利用Shell脚本实现对mysql数据库的备份：

www.cnblogs.com/mracale/p/7251292.html

3.Linux下的Crontab定时执行任务命令详解：

www.cnblogs.com/longjshz/p/5779215.html
```

## 往期推荐

### 1、java学习路线

喜欢此内容的人还喜欢

一张900w的数据表，16s执行的SQL优化到300ms?

我是程序汪