10 MySQL数据备份与还原(mysqldump)

MySQL**数据备份与还原**(mysqldump)

一 mysqldump指令实现数据备份、mysql指令实现数据还原

经常有朋友问我，DBA到底是做什么的，百科上说：数据库管理员（Database Administrator，简称DBA），是从事管理和维护数据库管理系统(DBMS)的相关工作人员的统称，属于运维工程师的一个分支，主要负责业务数据库从设计、测试到部署交付的全生命周期管理。DBA的核心目标是保证数据库管理系统的稳定性、安全性、完整性和高性能。

百科出来的内容总是那么的专业，让人看完之后的感觉是很解释的很好，但是我没有看懂或者似懂非懂的模糊感。。哈哈，其实我认为，DBA主要做三件事情：1.保证公司的数据不丢失不损坏 2.提高数据库管理系统的工作性能

对于现在的公司来讲，数据变得尤为重要，可以说最重要，你的网站可以无法访问，服务器可以宕机，但是数据绝对不能丢，所以我们本节内容就冲着如果保护好数据而来的。本篇博客的内容并不是很深入，毕竟不是专业的DBA，只是作为超哥的讲课内容，让大家学一些数据备份的基本操作，入门级别咱们只讲一下mysqldump指令，至于如果做主从复制，双机热备，数据库高可用，数据库集群，大家可以去看我其他的博客，博客写完了，目前还在整理，整理好之后我就发出来供大家批评指正，共同学习~~~，因为毕竟咱们学的是开发，本篇内容就当作是拓展自己的知识领域吧，对你来讲都是很有好处的~~~大家加油吧

那么我们就来学一下mysqldump指令。

1.首先我们先创建一个名为crm2的库

```
mysql> create database crm2;
mysql> show create database crm2;
```

2.切换到crm2库下

```
mysql> use crm2;
```

3.创建两张表，student表和class表

```
mysql> create table tb1(id int primary key,name char(8) not null,age int,class_id int not null);
Query OK, 0 rows affected (0.63 sec)

mysql> create table class(id int primary key,cname char(20) not null);
Query OK, 0 rows affected (0.34 sec)
```

4.给两张表插入一些数据

```
mysql> insert into class values(1,'一班'),(2,'二班');
mysql> insert into student values(1,'Jaden',18,1),(2,'太白',45,1),(3,'彦涛',30,2);
```

5.查看一下两个表的数据

```
mysql> select * from student;
+----+--------+------+----------+
| id | name | age | class_id |
+----+--------+------+----------+
| 1 | Jaden | 18 | 1 |
| 2 | 太白 | 45 | 1 |
| 3 | 彦涛 | 30 | 2 |
+----+--------+------+----------+
3 rows in set (0.00 sec)

mysql> select * from class;
+----+--------+
| id | cname |
+----+--------+
| 1 | 一班 |
| 2 | 二班 |
+----+--------+
2 rows in set (0.00 sec)
```

好，前期工作准备完毕，下面我们来通过mysqldump指令进行备份，在cmd窗口下执行下面的指令，注意不是进入mysql里面输入的，是在外面面。

C:\WINDOWS\system32>mysqldump -h 127.0.0.1 -u root -p666 crm2 > f:\数据库备份练习\crm2.sql
Warning: Using a password on the command line interface can be insecure. (这个提示是因为我把密码显示出来了，自己在自己电脑上测试的时候，这个警告可以忽略)

然后我们就会发现在这个'f:\数据库备份练习\'路径下面就有了crm2.sql文件
然后我们通过nodepad++(随便一个文本编辑器都可以)，打开看看里面的内容：

```
-- MySQL dump 10.13 Distrib 5.6.42, for Win64 (x86_64)
--
-- Host: 127.0.0.1 Database: crm2
-- ------------------------------------------------------
-- Server version 5.6.42

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0
*/;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `class`
--

DROP TABLE IF EXISTS `class`; --如果之前存在class表，就将之前的class表删除
/*!40101 SET @saved_cs_client = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `class` ( --创建表
`id` int(11) NOT NULL,
`cname` char(20) NOT NULL,
PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `class`
--

LOCK TABLES `class` WRITE; --锁表
/*!40000 ALTER TABLE `class` DISABLE KEYS */;
INSERT INTO `class` VALUES (1,'一班'),(2,'二班'); --插入数据
/*!40000 ALTER TABLE `class` ENABLE KEYS */;
UNLOCK TABLES; --解锁
.....
```

等等大致内容(如果你插入的数据有中文，这里显示的确实乱码的同学，往回看看我的关于修改mysql字符集编码的博客，将编码改为统一的，然后重新操作一遍就可以了)

上面的这个指令的意思就是将crm2这个库，备份到这个'f:\数据库备份练习\'路径下，并且命名为crm2.sql文件。

执行备份语句的时候，其中可以加上很多的参数，用来添加一些备份的时候的特殊要求的，其中有一个-B参数，执行备份语句时，如果加上了-B参数，那么将来再执行数据还原的时候，就不需要自己到数据库里面去先创建一个crm2这个库了，并且执行数据还原语句的时候就不需要指定crm2这个库了，如果没有加-B参数，就需要自行到数据库中先创建一个crm2这个库，并且执行语句是要指定将数据恢复到这个crm2库里面，看对比：
首先上面我们执行的语句中没有加上-B参数，那么恢复数据的时候，怎么恢复呢，看下面的语句
1.连接到数据库中，并创建crm2这个库

```
mysql -u root -p666
mysql> create database crm2;
```

2.退出mysql或者重新启动一个cmd窗口，然后执行

```
mysql -uroot -p 库名 < mysqldump出来的那个sql文件的路径
```

例如：mysql -uroot -p crm2< f:\数据库备份练习\crm2.sql

3.这样就恢复好了，我们连接上数据库并查看里面的内容：

```
mysql -u root -p666
use crm2;
mysql> show tables;
+----------------+
| Tables_in_crm2 |
+----------------+
| class |
| student |
+----------------+
2 rows in set (0.00 sec)

mysql> select * from student;
+----+--------+------+----------+
| id | name | age | class_id |
+----+--------+------+----------+
| 1 | Jaden | 18 | 1 |
| 2 | 太白 | 45 | 1 |
| 3 | 彦涛 | 30 | 2 |
+----+--------+------+----------+
3 rows in set (0.00 sec)

mysql> desc student;
+----------+---------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+----------+---------+------+-----+---------+-------+
| id | int(11) | NO | PRI | NULL | |
| name | char(8) | NO | | NULL | |
| age | int(11) | YES | | NULL | |
| class_id | int(11) | NO | | NULL | |
+----------+---------+------+-----+---------+-------+
4 rows in set (0.02 sec)
```

就这么简单我们就将数据库恢复了，表中的数据和表结构都恢复了。

执行mysqldump的时候加上了-B参数，那么恢复数据的时候，就不需要指定是恢复那个库里面的数据了，也不需要提前到数据库中创建一个crm2库了，因为-B参数导出的文件中自带创建数据库和连接数据库的功能：(使用-B参数备份出来的内容自带create database 库名和use 库名的功能)

1.mysqldump -uroot -p -B crm2> f:\数据库备份练习\crm2.sql

2.在cmd窗口下执行：mysql -uroot -p < f:\数据库备份练习\crm2.sql

3.查看一下是否恢复了：

```
mysql> show databases;
+--------------------+
| Database |
+--------------------+
| information_schema |
| crm2 |
| d1 |
| mysql |
| performance_schema |
| test |
+--------------------+
mysql> use crm2;
Database changed
```

```
mysql> show tables;
+----------------+
| Tables_in_crm2 |
+----------------+
| class |
| student |
+----------------+
2 rows in set (0.00 sec)
mysql> select * from class;
+----+--------+
| id | cname |
+----+--------+
| 1 | 一班 |
| 2 | 二班 |
+----+--------+
2 rows in set (0.00 sec)

mysql> desc student;
+----------+---------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+----------+---------+------+-----+---------+-------+
| id | int(11) | NO | PRI | NULL | |
| name | char(8) | NO | | NULL | |
| age | int(11) | YES | | NULL | |
| class_id | int(11) | NO | | NULL | |
+----------+---------+------+-----+---------+-------+
4 rows in set (0.02 sec)
```

上面我们就完成了一个简单数据库备份和恢复的过程(在linux下面还可以在导出的时候压缩文件内容，减小空间占用mysqldump -uroot -p -B crm2|gzip> f:\数据库备份练习\crm2.sql.gz，windows好像是没有自带的zip压缩指令，大家有兴趣的可以去查一下，作为了解吧)

原理：其实很简单，就是把数据从mysql库里面以逻辑的sql语句的形式直接输出或者生成备份文件的过程。

上面我们说完了单库备份，下面来看看多个库怎么备份呀

C:\WINDOWS\system32>mysqldump -uroot -p -B crm2 mysql> f:\数据库备份练习\crm2mysql.sql

Enter password: ***

就是多个库名用空格分开，这样备份出来的sql文件还是一个，也就是这两个库都备份到一个文件里面了。一般作备份的时候的文件名字都是库名.sql，多个库一般就是库名_库名_库名.sql。

那如果我们将很多的库都是一起备份的，但是我们只想恢复其中一个库怎么办，这样搞是不是就不太合适了(因为一个文件算是一个备份，在进行恢复的时候，一下就将文件里面的所有的库都还原了，效率低不说，还麻烦)，那就需要分库备份了，也就是将每个库分开来进行备份，自己备份自己的，一个一个来

其实就是执行多个单库备份的语句

mysqldump -uroot -p -B crm2> f:\数据库备份练习\crm2.sql

mysqldump -uroot -p -B mysql> f:\数据库备份练习\crm2.sql

...

但是如果库比较多(企业的数据库里一般都会有多个库)，这么写就比较麻烦了，所以需要获取所有数据库的库名，然后根据库名来循环执行上面的单库备份的语句，并将库名作为变量放到语句里面进行循环。需要写脚本(就是一堆系统指令组成的程序)来做这件事情了，具体怎么做，咱们就不说啦，目前知道一下就可以了(写一个.sh文件，然后sh+文件来执行这个文件，文件里面写个for循环就行了，等你学会写shell脚本就会了)

备份表：

上面我们说的是如何备份库，现在我们来看看如果备份其中的某个表：

语法：mysqldump -u 用户名 -p 库名 表名> (路径)备份的文件名

mysqldump -uroot -p crm2 student> f:\数据库备份练习\crm2_table_student.sql

单纯进行表备份的时候，就不用写-B参数了，因为库crm2后面就是student表了，也就是说你的crm2库还在呢

备份多个表：

语法：mysqldump -u 用户名 -p 库名 表名1 表名2> (路径)备份的文件名

和多个库一起备份有一个同样的问题，就是如果我只需要恢复某一张表怎么办，上面的多表备份是不是也不太合适啊，所以又要进行分表备份

又是同样的套路，获取所有的表名，写一个循环脚本，执行单表备份的指令。

分库分表备份有些缺点：文件多，很碎，数据量非常大的时候，效率低

1.做一个完整的全备，再做一个分库分表的备份

2.脚本批量恢复多个sql文件。

备份数据库表结构：

利用mysqldump -d参数只备份表的结果，例如：备份crm2库的所有表的结构：

C:\WINDOWS\system32>mysqldump -uroot -p -B -d crm2> f:\数据库备份练习\crm2stru.sql

Enter password: ***

备份出来的文件打开一看，就没有了插入数据的部分

mysqldump的关键参数说明：

1.-B指定多个库，增加建库语句和use 语句

2.--compact 去掉注释，适合调试输出，生产上不用

3.-A或者--all-databases

例如：C:\WINDOWS\system32>mysqldump -uroot -p -B -A> f:\数据库备份练习\all.sql

Enter password: ***

4.-F刷新binlog日志(binlog具体是什么，后面咱们再解释)

5.--master-data 增加binlog日志文件名及对应的为支点。

6.-x，--lock-all-tables 将所有的表锁住，一般mysql引擎都是锁表，全部都不能使用了，所有不太友好

7.--add-locks这个选项会在INSERT语句中捆上一个LOCK TABLE和UNLOCK TABLE语句。这就防止在这些记录被再次导入数据库时其他用户对表进行的操作(mysql默认是加上的)

8.-l，--lock-tables Lock all tables for read

9.-d 只备份表结构

10.-t 只备份数据

11. --single-transaction 开启事务，适合innodb事务数据库备份

InnoDB表在备份时，通常启用选项--single-transaction来保证备份的一致性，实际上他的工作原理时设定本次会话的隔离界别为：REPEATABLE READ，以确保本次会话(dump)时，不会看到其他会话已经提交了数据。

MyISAM全库备份指令推荐：(gzip是压缩文件为zip类型的)

mysqldump -uroot -p666 -A -B --master-data=2 -x|gzip>f:\数据库备份练习\all.sql.gz

InnoDB全库备份指令推荐：

mysqldump -uroot -p666 -A -B --master-data=2 --single-transaction|gzip>f:\数据库备份练习\all.sql.gz

数据恢复：

一、通过source命令恢复数据库

进入mysql数据库控制台，mysql -uroot -p666登陆后

mysql>use 数据库;

然后使用source命令，后面参数为脚本文件(如这里用到的是.sql文件，如果你备份的是.txt文件，那这里写.txt文件)

mysql>source crm2.sql #这个文件是系统路径下的，默认是登陆mysql前的系统路径，在mysql中查看系统路径的方法是通过system+系统命令来搞的

mysql>system ls

二、利用mysql命名恢复(标准)

mysql -root -p666 -e "use crm2;drop table student;show tables;" 必须是双引号

mysql -uroot -p666 crm2<f:\数据库备份练习\crm2.sql

mysql -uroot -p666 -e "use crm2;show tables;"

注：如果sql文件里面没有use db这样的字样时，在导入时就要指定数据库名了。

mysql -uroot -p666 crm2<.sql文件

建议备份数据库时都指定上-B参数，效果好

说明：mysql不光可以恢复mysqldump的备份，只要文件中是sql语句，都可以通过mysql命令执行到数据库中

mysql 带-e参数实现非交互式对话，就是不需要到mysql里面去，在外面执行里面的指令的方法,例如：mysql -uroot -p666 -e "use crm2;show tables;"，但是语句必须是双引号包裹。

批量恢复库：找到所有的数据库名，然后通过库名去循环恢复

关于binlog，等我整理好在给大家吧~~~

#下面的这些内容是我之前整理的，大家不要看了，我会改版的，等新版出来之后在发出来给大家，下面的太晦涩难懂了~~~

二 MySQL数据备份

```
#1. 物理备份： 直接复制数据库文件，适用于大型数据库环境。但不能恢复到异构系统中如Windows。
#2. 逻辑备份： 备份的是建表、建库、插入等操作所执行SQL语句，适用于中小型数据库，效率相对较低。
#3. 导出表： 将表导入到文本文件中。
```

一、使用mysqldump实现逻辑备份

```
#语法:
# mysqldump -h 服务器 -u用户名 -p密码 数据库名 > 备份文件.sql

#示例:
#单库备份
mysqldump -uroot -p123 db1 > db1.sql
mysqldump -uroot -p123 db1 table1 table2 > db1-table1-table2.sql

#多库备份
mysqldump -uroot -p123 --databases db1 db2 mysql db3 > db1_db2_mysql_db3.sql

#备份所有库
mysqldump -uroot -p123 --all-databases > all.sql
```

二、恢复逻辑备份

```
#方法一:
```

```
[root@localhost backup]# mysql -uroot -p123 < /backup/all.sql

#方法二:
mysql> use db1;
mysql> SET SQL_LOG_BIN=0;
mysql> source /root/db1.sql

#注:如果备份/恢复单个库时,可以修改sql文件
DROP database if exists school;
create database school;
use school;
```

三、备份/恢复案例

```
#数据库备份/恢复实验一:数据库损坏
备份:
1. # mysqldump -uroot -p123 --all-databases > /backup/`date +%F`_all.sql
2. # mysql -uroot -p123 -e 'flush logs' //截断并产生新的binlog
3. 插入数据 //模拟服务器正常运行
4. mysql> set sql_log_bin=0; //模拟服务器损坏
mysql> drop database db;

恢复:
1. # mysqlbinlog 最后一个binlog > /backup/last_bin.log
2. mysql> set sql_log_bin=0;
mysql> source /backup/2014-02-13_all.sql //恢复最近一次完全备份
mysql> source /backup/last_bin.log //恢复最后个binlog文件


#数据库备份/恢复实验二:如果有误删除
备份:
1. mysqldump -uroot -p123 --all-databases > /backup/`date +%F`_all.sql
2. mysql -uroot -p123 -e 'flush logs' //截断并产生新的binlog
3. 插入数据 //模拟服务器正常运行
4. drop table db1.t1 //模拟误删除
5. 插入数据 //模拟服务器正常运行

恢复:
1. # mysqlbinlog 最后一个binlog --stop-position=260 > /tmp/1.sql
# mysqlbinlog 最后一个binlog --start-position=900 > /tmp/2.sql
2. mysql> set sql_log_bin=0;
mysql> source /backup/2014-02-13_all.sql //恢复最近一次完全备份
mysql> source /tmp/1.log //恢复最后个binlog文件
mysql> source /tmp/2.log //恢复最后个binlog文件

注意事项:
1. 完全恢复到一个干净的环境(例如新的数据库或删除原有的数据库)
2. 恢复期间所有SQL语句不应该记录到binlog中
```

四、实现自动化备份

```
备份计划:
1. 什么时间 2:00
```

```
2. 对哪些数据库备份
3. 备份文件放的位置

备份脚本:
[root@localhost~]# vim /mysql_back.sql
#!/bin/bash
back_dir=/backup
back_file=`date +%F`_all.sql
user=root
pass=123

if [ ! -d /backup ];then
mkdir -p /backup
fi

# 备份并截断日志
mysqldump -u${user} -p${pass} --events --all-databases > ${back_dir}/${back_file}
mysql -u${user} -p${pass} -e 'flush logs'

# 只保留最近一周的备份
cd $back_dir
find . -mtime +7 -exec rm -rf {} \;

手动测试:
[root@localhost ~]# chmod a+x /mysql_back.sql
[root@localhost ~]# chattr +i /mysql_back.sql
[root@localhost ~]# /mysql_back.sql

配置cron:
[root@localhost ~]# crontab -l
2 * * * /mysql_back.sql
```

五、表的导出和导入

```
SELECT... INTO OUTFILE 导出文本文件
示例:
mysql> SELECT * FROM school.student1
INTO OUTFILE 'student1.txt'
FIELDS TERMINATED BY ',' //定义字段分隔符
OPTIONALLY ENCLOSED BY '"' //定义字符串使用什么符号括起来
LINES TERMINATED BY '\n' ; //定义换行符


mysql 命令导出文本文件
示例:
# mysql -u root -p123 -e 'select * from student1.school' > /tmp/student1.txt
# mysql -u root -p123 --xml -e 'select * from student1.school' > /tmp/student1.xml
# mysql -u root -p123 --html -e 'select * from student1.school' > /tmp/student1.html

LOAD DATA INFILE 导入文本文件
mysql> DELETE FROM student1;
mysql> LOAD DATA INFILE '/tmp/student1.txt'
INTO TABLE school.student1
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"'
LINES TERMINATED BY '\n';
```

```
#可能会报错
mysql> select * from db1.emp into outfile 'C:\\db1.emp.txt' fields terminated by ',' lines terminated by '\r\n';
ERROR 1238 (HY000): Variable 'secure_file_priv' is a read only variable


#数据库最关键的是数据，一旦数据库权限泄露，那么通过上述语句就可以轻松将数据导出到文件中然后下载拿走，因而mysql对此作了
在配置文件中
[mysqld]
secure_file_priv='C:\\' #只能将数据导出到C:\\下

重启mysql
重新执行上述语句
```

六、数据库迁移

```
务必保证在相同版本之间迁移
# mysqldump -h 源IP -uroot -p123 --databases db1 | mysql -h 目标IP -uroot -p456
```