MySQL学习笔记（Day003：升级/参数/连接/权限）

MySQL学习

一. 数据库升级

1. 环境说明：

一般说来，MySQL数据库的二进制数据文件，也就是 `my.cnf` 中的配置项 `datadir` 所在的位置，和我们MySQL应用程序安装的位置，是分开的，仅仅通过配置项告诉MySQL，数据库的数据存在 `datadir` 这个目录下。当程序和数据分离以后，方便我们对数据库应用程序做版本的升级或者回退。

2. 环境举例：

- *MySQL*安装目录：
  - **MySQL 5.6.27:** /usr/local/mysql-5.6.27-linux-glibc2.5-x86_64
  - **MySQL 5.7.9 :** /usr/local/mysql-5.7.9-linux-glibc2.5-x86_64
- *datadir*目录：
  - /data/mysq_data/
- 初始环境：

```
shell> ll | grep mysql
lrwxrwxrwx   1 root root    34 Nov 16 13:40 mysql -> mysql-5.6.27-linux-glibc2.5-x86_64
drwxr-xr-x  13 root mysql  4096 Nov 16 13:37 mysql-5.6.27-linux-glibc2.5-x86_64
drwxr-xr-x   9 7161 wheel 4096 Oct 12 00:29 mysql-5.7.9-linux-glibc2.5-x86_64

shell> ll /data/mysql_data/
total 13540
-rw-rw---- 1 mysql mysql    65468 Nov 16 13:50 bin.000001
-rw-rw---- 1 mysql mysql  1176237 Nov 16 13:50 bin.000002
-rw-rw---- 1 mysql mysql       26 Nov 16 13:50 bin.index
-rw-rw---- 1 mysql mysql     6882 Nov 16 13:50 error.log
-rw-rw---- 1 mysql mysql      865 Nov 16 13:50 ib_buffer_pool
-rw-rw---- 1 mysql mysql 12582912 Nov 16 13:50 ibdata1
drwx------ 2 mysql mysql     4096 Nov 16 13:50 mysql
drwx------ 2 mysql mysql     4096 Nov 16 13:50 performance_schema
drwx------ 2 mysql mysql     4096 Nov 16 13:49 test
```

3. 版本升级

```
shell> /etc/init.d/mysqld stop  #安全的停止数据库的运行
shell> cd /usr/local/
shell> unlink mysql
shell> ln -s mysql-5.7.9-linux-glibc2.5-x86_64 mysql
        #此时，MySQL的应用程序版本已经升级完成
        #/etc/init.d/mysqld
        #/etc/profile中PATH增加的/usr/local/mysql/bin
        #都不需要做任何的改变，即可将当前系统的mysql版本升级完成
        #注意：此时只是应用程序升级完成，系统表仍然还是5.6的版本


shell> cd /usr/local/mysql
shell> chown root.mysql . -R
#5.7.x -> 5.6.X 降级存在问题，这里暂且注释掉
#shell> cp /data/mysql_data/mysql /你的备份路径/mysql_5_6_27.backup -r
       #该步骤将mysql5.6.27版本的系统表进行了备份，以便将来可以回退

shell> /etc/init.d/mysqld start
#此时 /etc/init.d/mysqld start  # 可以启动
#      且可以使用 mysql -u root -p （原密码） 进入数据库
#      show databases;存在test表，而没有sys表（数据的二进制文件兼容）
#      但是如果去看error.log会发现好多的WARNNING
#      所以，这个时候我们要去 upgrade 去升级


shell> mysql_upgrade -p -s
        #参数 -s 一定要加,表示只更新系统表，-s: upgrade-system-tables
        #如果不加-s,则会把所有库的表以5.7.9的方式重建，线上千万别这样操作
        #因为数据库二进制文件是兼容的，无需升级
        #什么时候不需要-s ？当一些老的版本的存储格式需要新的特性，
        #                   来提升性能时，不加-s
        #即使通过slave进行升级，也推荐使用该方式升级，速度比较快

Enter password:
The --upgrade-system-tables option was used, databases wont be touched.
Checking if update is needed.
Checking server version.
Running queries to upgrade MySQL server.
Upgrading the sys schema.
Upgrade process completed successfully.
Checking if update is needed.

shell> mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.9-log MySQL Community Server (GPL)

Copyright (c) 2000, 2015, 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              |  # 这个就是升级后的系统库，如果回退，将备份的拷贝回来覆盖即可
| performance_schema |
| sys                |  # 5.7 新的sys库
| test               |  # 5.6 中的test库
+--------------------+
5 rows in set (0.00 sec)
```

> `5.1.X` 、 `5.5.X` 、 `5.6.X` 是可以直接通过该方式升级到 `5.7.X` 。 `5.0.X` 未知，需要测试

注意：
如果原来数据二进制文件保存在**/usr/local/ `mysql-5.6.27` -linux-glibc2.5-x86_64/data**目录下,在升级之前，要么将该目录的数据拷贝到新的你指定的data目录（比如**/usr/local/ `mysql-5.7.9` -linux-glibc2.5-x86_64/data** ），要么修改 `my.cnf` ，将 `datadir` 指向 `/usr/local/mysql-5.6.27-linux-glibc2.5-x86_64/data` ，总之一定要确保 `my.cnf` 中的数据位置和你实际的数据位置是一致的，不管是默认的也好，还是你 `datadir` 指定的也好

4.关于降级问题的说明

通过覆盖 `mysql` 系统表的方式存在问题，会导致启动不起来。官方建议如下：

> 官方MySQL5.7降级建议
> 上述建议中使用的SQL语句可在 `mysql5.7` 的源码的 `srcipts/mysql_system_tables_fix_for_downgrade.sql` 中找到，或者直接运行这个sql脚本。

姜老师测试后发现，有bug; 可以启动，但是原来的用户表，无法访问。
当前记录时间是2015-11-17，等待下一步解决方案。

二. MySQL的连接登录

1. 几种登录方式

- 方式一 `mysql -p`
  - 该方法默认使用root用户, 可使用 `select user();` 查看当前用户
- 方式二 `mysql -S /tmp/mysql.sock -u root -p` 密码*A*
  - 该方法适用于在安装MySQL主机上进行本地登录
- 方式三 `mysql -h 127.0.0.1 -u root -p` 密码*B*
  - 使用 `'root'@'127.0.0.1'` 这个用户登录
- 方式四 `mysql -h localhost -u root -p` 密码*A*
  - 该方式等价与【方式二】，且和【方式三】属于两个不同的"用户"

2. 免密码登录

- 方式一 `my.cnf` 增加 `[client]` 标签

```
[client]
user="root"
password="你的密码"



#单对定义不同的客户端

[mysql] # 这个是给/usr/loca/mysql/bin/mysql 使用的
user=root
password="你的密码"

[mysqladmin] # 这个是给/usr/local/mysql/bin/mysqladmin使用的
user=root
password="你的密码"
```

每个不同的客户端需要定义不同的标签，使用 `[client]` 可以统一

- 方式二 `login-path`

```
shell> mysql_config_editor set -G vm1 -S /tmp/mysql.sock -u root -p
Enter password [输入root的密码]

shell> mysql_config_editor print --all
[vm1]
user=root
password=*****
socket=/tmp/mysql.sock


#login

shell> mysql --login-path=vm1 # 这样登录就不需要密码，且文件二进制存储 ,位置是 ~/.mylogin.cnf
```

该方式相对安全。如果server被黑了，该二进制文件还是会被破解

- 方式三 `~/.my.cnf`，自己当前家目录

```
#Filename: ~/.my.cnf

[client]
user="root"
password="你的密码"
```

三. MySQL 参数介绍和设置

1. 参数的分类

- 全局参数：GLOBAL
  - 可修改参数
  - 不可修改参数
- 会话参数：SESSION
  - 可修改参数
  - 不可修改参数

> 1: 用户可在线修改 `非只读参数` ， `只读参数` 只能预先在配置文件中进行设置，通过重启数据库实例,方可生效。
>
> 2: 所有的在线修改过的参数(GLOBAL/SESSION)，在重启后，都会丢失，不会写如 `my.cnf` ，无法将修改进行持久化
>
> 3: 有些参数，即存在于 `GLOBAL` 又存在于 `SESSION`，比如 `autocommit` (PS：MySQL默认是提交的)

2. 查看参数

```
mysql> show variables; # 显示当前mysql的所有参数，且无隐藏参数
mysql> show variables like "max_%"; #查以max_开头的变量
```

3. 设置参数

- 设置全局(GLOBAL)参数

```
mysql> set global slow_query_log = off; #不加global，会提示错误
                                        #slow_query_log是全局参数

mysql> set slow_query_log = off;  # 下面就报错了，默认是会话参数
ERROR 1229 (HY000): Variable 'slow_query_log' is a GLOBAL variable and should be set with SET GLOBAL
```

- 设置会话(SESSION)参数

```
mysql> set autocommit = 0;  # 当前会话生效

# 或者

mysql> set session autocommit = 0;  # 当前会话生效
```

`autocommit` 同样在 `GLOBAL` 中，也有同样的参数

```
mysql> set global autocommit = 1; #当前实例，全局生效
```

注意：如果这个时候/etc/init.d/mysqld restart，则全局的autocommit的值会变成默认值，或者依赖于my.cnf的设置值。

执行的效果如下：

```
mysql> show variables like "slow%"; # 原值为ON
+---------------------+----------+
| Variable_name       | Value    |
+---------------------+----------+
| slow_launch_time    | 2        |
| slow_query_log      | OFF      |
| slow_query_log_file | slow.log |
+---------------------+----------+
3 rows in set (0.00 sec)

mysql> select @@session.autocommit; # 等价于 slect @@autocomit;
+----------------------+
| @@session.autocommit |
+----------------------+
|                    0 |
+----------------------+
1 row in set (0.00 sec)

mysql> select @@global.autocommit;
+---------------------+
| @@global.autocommit |
+---------------------+
|                   1 |
+---------------------+
1 row in set (0.00 sec)
```

四. 权限管理

1. “用户 + IP”的概念

MySQL中同一个用户名，比如Bob,能否登录，以及用什么密码登录，可以访问什么库等等，都需要加上IP，才可以表示一个完整的用户标识

> `bob@127.0.0.1` 和 `bob@loalhost` 以及 `bob@192.168.1.100` 这三个其实是 `不同` 的 **用户标识**

2. 用户权限管理

- **系统表权限信息:**
  - **a) 用户名和IP是否允许**
  - **b) 查看mysql.user表** `// 查看全局所有库的权限`
  - **c) 查看mysql.db表** `// 查看指定库的权限`
  - **d) 查看mysql.table_priv表** `// 查看指定表的权限`
  - **e) 查看mysql.column_priv表** `// 查看指定列的权限`

  tips: mysql> desc [tablename]; 可以查看表的结构信息；
- **常用权限：**
  - SQL语句：SELECT、INSERT、UPDATE、DELETE、INDEX
  - 存储过程：CREATE ROUTINE、ALTER ROUTINE、EXECUTE、TRIGGER
  - 管理权限：SUPER、RELOAD、SHOW DATABASE、SHUTDOWN、

  所有权限猛戳这里
- **可选资源:**
  - MAX_QUERIES_PER_HOUR *count*
  - MAX_UPDATES_PER_HOUR *count*
  - MAX_CONNECTIONS_PER_HOUR *count*
  - MAX_USER_CONNECTIONS *count*

  tips:只能精确到小时，对于部分场景不适用，可以考虑中间件方式
- **显示当前用户的权限**

```
#这三个是同一个意思

mysql> show grants;
mysql> show grants for current_user;
mysql> show grants for current_user();
```

3. 基本操作

```
mysql> create user 'bob'@'127.0.0.1' identified by '123';
       #创建一个认证用户为'bob'@'127.0.0.1',密码是123
mysql> grant all on NWDB.* to 'bob'@'127.0.0.1';
       #授予他NWDB库下面所有表的所有访问权限; *.*表示所有库的所有表

mysql> grant all on NWDB.* to 'alice'@'127.0.0.1' identified by '123';
       #这个grant语句会搜索用户，如果用户不存在，则自动创建用户，
       #如果不带identified by，则该用户名密码为空

mysql> grant all on *.* to 'tom'@'192.168.10.%' identified by '123' with grant option;
       #表示这个用户'tom'@'127.0.0.1'可以访问所有库的所有表，
       #同时，他还可以给其他用户授予权限(with grant option)，
       #注意如果，*.*改成了某一个指定的非USER库，
       #则tom没法去新建其他用户了，因为User库没有权限了
       #192.168.10.% 表示属于192.168.10.0/24网段的用户可以访问
```

4. 撤销权限

- `revoke` 关键字，该关键字只删除用户权限，不删除用户
- `revoke` 语法同 `grant` 一致, 从 `grant ... to` 变为 `revoke ... from`