MySQL学习笔记（Day004：权限拾遗/Role模拟/Workbench/体系结构）

MySQL学习

一. 权限拾遗

1. GRANT与创建用户

```
mysql> grant select on sys.* to 'perf'@'127.0.0.1' identified by '123';
Query OK, 0 rows affected, 1 warning (0.01 sec) -- 这里有一个warning

mysql> show warnings;
-- 输入warning的Message如下：
-- Using GRANT for creating new user is deprecated and will be removed in future release. Create new user with CREATE USER statement.
```

上面的这个例子使用 `GRANT` 赋权限的同时创建了 `'perf'@'127.0.0.1'` 这个用户，但是出现了 `warning` ，从给出的提示看来，以后的MySQL版本会废弃掉这种方式

- **正确的创建用户并赋权的方式**

```
mysql> create user 'pref'@'127.0.0.1' identified by '123';
Query OK, 0 rows affected (0.00sec)
mysql> grant select on sys.* to 'perf'@'127.0.0.1';
Query OK, 0 rows affected (0.00sec)
```

2. 查看某一个用户的权限

```
mysql> show grants for 'perf'@'127.0.0.1';
+-----------------------------------------------+
| Grants for perf@127.0.0.1                     |
+-----------------------------------------------+
| GRANT USAGE ON *.* TO 'perf'@'127.0.0.1'      | -- USAGE表示用户可以登录
| GRANT SELECT ON `sys`.* TO 'perf'@'127.0.0.1' | -- 对sys库的所有表有select权限
+-----------------------------------------------+
2 rows in set (0.00 sec)
```

3. 删除某一个用户

```
mysql> drop user 'perf'@'127.0.0.1';
Query OK, 0 rows affected (0.00sec)
```

4. MySQL权限信息

```
mysql> select * from mysql.user where user='perf'\G
*************************** 1. row ***************************
                  Host: 127.0.0.1
                  User: perf
           Select_priv: N   ---由于perf用户是对sys库有权限，所以这里（USER）全是N
           Insert_priv: N
           Update_priv: N
           Delete_priv: N
           Create_priv: N
             Drop_priv: N
           Reload_priv: N
         Shutdown_priv: N
          Process_priv: N
             File_priv: N
            Grant_priv: N
       References_priv: N
            Index_priv: N
            Alter_priv: N
          Show_db_priv: N
            Super_priv: N
 Create_tmp_table_priv: N
      Lock_tables_priv: N
          Execute_priv: N
       Repl_slave_priv: N
      Repl_client_priv: N
      Create_view_priv: N
        Show_view_priv: N
   Create_routine_priv: N
    Alter_routine_priv: N
      Create_user_priv: N
            Event_priv: N
          Trigger_priv: N
Create_tablespace_priv: N
              ssl_type:
            ssl_cipher:
           x509_issuer:
          x509_subject:
         max_questions: 0
           max_updates: 0
       max_connections: 0
  max_user_connections: 0
                plugin: mysql_native_password
 authentication_string: *23AE809DDACAF96AF0FD78ED04B6A265E05AA257
      password_expired: N
 password_last_changed: 2015-11-18 12:20:13
     password_lifetime: NULL
        account_locked: N    -- 如果这里为Y，该用户就无法使用了
1 row in set (0.00 sec)

------------------------我是分割线-----------------------------

mysql> select * from mysql.db where user='perf'\G
*************************** 1. row ***************************
                 Host: 127.0.0.1
                   Db: sys  -- sys 数据库
                 User: perf
          Select_priv: Y    -- 有select权限，和我们赋予perf的权限一致
          Insert_priv: N
          Update_priv: N
          Delete_priv: N
          Create_priv: N
            Drop_priv: N
           Grant_priv: N
      References_priv: N
           Index_priv: N
           Alter_priv: N
Create_tmp_table_priv: N
     Lock_tables_priv: N
     Create_view_priv: N
       Show_view_priv: N
  Create_routine_priv: N
   Alter_routine_priv: N
         Execute_priv: N
           Event_priv: N
         Trigger_priv: N
1 row in set (0.00 sec)
```

> **注意：**
> 不建议 **使用 `INSERT` 或者 `GRANT` 对元数据表进行修改，来达到修改权限的目的**

5. information_schema

```
mysql> select user();
+----------------+
| user()         |
+----------------+
| perf@127.0.0.1 |
+----------------+
1 row in set (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |  -- 这是一个特殊的数据库，我们虽然可以看见，但其实没有权限
| sys                |
+--------------------+
2 rows in set (0.00 sec)

mysql> use information_schema;
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  INNODB_CMP;
ERROR 1227 (42000): Access denied; you need (at least one of) the PROCESS privilege(s) for this operation
```

二. MySQL模拟角色

1. 角色的定义：

> 角色 (Role)可以用来批量管理用户，同一个角色下的用户，拥有相同的权限。
> MySQL5.7.X 以后可以模拟角色(Role)的功能，通过 mysql.proxies_priv 模拟实现。
> mysql.proxies_priv 在 5.5.X 和 5.6.X 的时候就存在，但是 无法模拟 角色(Role)功能。

2. 模拟角色操作：

```
mysql> create user 'junior_dba'@'127.0.0.1';  -- 相当于定于一个 角色(Role),
                                               -- 但这只是个普通的用户，名字比较有(Role)的感觉
                                               -- 有点类似用户组
Query OK, 0 rows affected (0.00sec)

mysql> create user 'tom'@'127.0.0.1';         -- 用户1
Query OK, 0 rows affected (0.02sec)

mysql> create user 'jim'@'127.0.0.1';         -- 用户2
Query OK, 0 rows affected (0.02sec)

mysql> grant proxy on 'junior_dba'@'127.0.0.1' to 'tom'@'127.0.0.1';  -- 将junior_dba的权限映射(map)到tom
Query OK, 0 rows affected (0.02sec)

mysql> grant proxy on 'junior_dba'@'127.0.0.1' to 'jim'@'127.0.0.1';  -- 然后映射(map)给jim
Query OK, 0 rows affected (0.01sec)

mysql> grant select on *.* to 'junior_dba'@'127.0.0.1';  -- 给junior_dba（模拟的Role）赋予实际权限
Query OK, 0 rows affected (0.01 sec)


mysql> show grants for 'junior_dba'@'127.0.0.1';        -- 查看 junior_dba的权限
+-----------------------------------------------------+
| Grants for junior_dba@127.0.0.1                     |
+-----------------------------------------------------+
| GRANT SELECT ON *.* TO 'junior_dba'@'127.0.0.1' |
+-----------------------------------------------------+
1 row in set (0.00 sec)

mysql> show grants for 'jim'@'127.0.0.1';               -- 查看jim的权限
+-------------------------------------------------------------+
| Grants for jim@127.0.0.1                                    |
+-------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'jim'@'127.0.0.1'                     |
| GRANT PROXY ON 'junior_dba'@'127.0.0.1' TO 'jim'@'127.0.0.1' |
+-------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql> show grants for 'tom'@'127.0.0.1';               -- 查看tom的权限
+-------------------------------------------------------------+
| Grants for tom@127.0.0.1                                    |
+-------------------------------------------------------------+
| GRANT USAGE ON *.* TO 'tom'@'127.0.0.1'                     |
| GRANT PROXY ON 'junior_dba'@'127.0.0.1' TO 'tom'@'127.0.0.1' |
+-------------------------------------------------------------+
2 rows in set (0.00 sec)

mysql> select * from mysql.proxies_priv;    --  查看 proxies_priv的权限
+-----------+------+--------------+-------------+-------------+-----------------------+---------------------+
| Host      | User | Proxied_host | Proxied_user | With_grant | Grantor               | Timestamp           |
+-----------+------+--------------+-------------+-------------+-----------------------+---------------------+
| localhost | root |              |              |          1 | boot@connecting host | 0000-00-00 00:00:00 |
| 127.0.0.1 | tom  | 127.0.0.1    | junior_dba   |          0 | root@localhost       | 0000-00-00 00:00:00 |
| 127.0.0.1 | jim  | 127.0.0.1    | junior_dba   |          0 | root@localhost       | 0000-00-00 00:00:00 |
+-----------+------+--------------+-------------+-------------+-----------------------+---------------------+
3 rows in set (0.00 sec)
```

> **`mysql.proxies_priv` 仅仅是对Role的 模拟 ，和Oracle的角色还是有所不同.官方称呼为 `Role like`**

三. Workbench与Utilities介绍

1. 下载

- Workbench-win32下载
- Workbench-win64下载
- Utilities下载

2. Workbench功能概述

- **SQL语句格式化**
- **SQL关键字upcase**
- **MySQL Dashboard**
- **SQL语法提示**
- **ER图**
- **Forward Engine** `//ER图 --> DB表结构`
- **Reverse** `//DB表结构 --> ER图`

3. Utilities安装

```
shell> python setup.py install  # 如果安装不成功，查看一下python的版本。推荐2.7.X
```

四. MySQL体系结构

1. 数据库

> 数据库（数据库文件）是一个或者一组二进制文件，通常来说存在与文件系统之上。

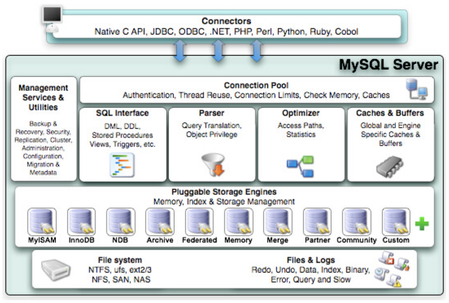
2. 数据库实例

> 由数据库后台进程/线程以及一个共享区域组成(程序的概念),数据库实例是用来操作数据库文件的

注意：`MySQL`中，数据库实例和数据库是一一对应的。没有`Oracle`的一对多`(RAC)`的机制。

3. MySQL体系结构

1. 单进程多线程结构
    - 不会影响MySQL的性能，看程序如何写。（多进程程序，进程间通信开销大于多线程）
2. 存储引擎的概念
    - 可以理解成文件系统，例如FAT32, NTFS, EXT4。 一个表是一个分区，引擎就是分区的文件系统
    - 存储引擎的对象就是表
    - show tables; 可以看到每个表对应的是上面引擎(Engine)
    - 除了特殊情况，我们现在就只考虑 `INNODB`
3. 体系结构图



4. 逻辑存储结构

MySQL逻辑存储结构
instance
database
schema
table
view

- **一个DB对应一个schema**
- **一个DB对应一个文件夹**
- **一个表对应一组文件**

```
                                          |--> table1 ---  | view1 |
MySQL Instance -----> Database ----> Schema ---> |--> table2 ---  | view2 |
                                          |--> table3 ---  | View3 |
```

> **注意：**
> MySQL中一个 `Database` 对应一个 `Schema` ,之所以要有这个 `schema` , 是为了兼容其他数据库
> `information_schema` 数据库不是文件夹，存在于内存中，在启动时创建

4. MySQL物理存储结构

1. MySQL配置文件
    - `datadir`
        - 存储数据二进制文件的路径
2. 表结构的组成
    - frm：表结构定义文件
    - MYI：索引文件
    - MYD：数据文件
    - 可以用 `hexdump -c XXX.frm` 查看二进制文件(意义不大)
    - `show create table tablename;`
    - mysqlfrm (utilities工具包)

        ```
        shell> mysqlfrm --diagnostic /data/mysql_data/aaa/.a.frm  #可将frm文件转成create table的语句
        ```

3. 错误日志文件
    - `log_err`
        - 建议配置成统一的名字
    - 方便定位错误
4. 慢查询日志文件
    - 将运行超过某一个时间 阈 (*yu*四声) 值 的SQL语句记录到文件