MySQL基础命令

第一、基础部分

1. 连接MySQL：`mysql -h host_address -u user_name -p user_password`

```
mysql -h110.110.110.110 -uroot -p123;
```

2. 修改密码：`mysqladmin -u user_name -p old_password password new_password`

```
mysqladmin -uroot -pabc123 password def456;
```

3. 增加新用户：`grant select on db_name.* to user_name@login_host identified by 'user_password'`

```
/* mysql grant命令添加用户常用的三种模式 */
grant all PRIVILEGES on *.* to 'test'@'localhost' identified by '123';
grant all PRIVILEGES on *.* to 'test'@'%' identified by '123';
grant all PRIVILEGES on *.* to 'test'@'10.22.225.18' identified by '123';
```

> 说明：
> 第一条命令添加一个本地用户 'test'，一般用于web服务器和数据库服务器在一起的情况；第二条命令添加一个用户 'test'，只要能连接数据库服务器的机器都可以使用，这个比较危险，一般不用；
> 最后条命令在数据库服务器上给 '10.22.225.18' 机器添加一个用户'test'，一般用于web服务器和数据库服务器分离的情况。

> 注意：
> 真正使用的时候不会用 `grant all PRIVILEGES on *.*`，而是根据实际需要设定相关的权限。
> 比如 `grant select,insert,delete,update on test.* to 'test'@'localhost' identified by '123';`

4. 创建数据库：`create database db_name`

```
create database news;
```

5. 显示数据库：show databases`

6. 删除数据库：`drop database db_name`

```
drop database news;
```

7. 连接数据库：`use db_name`

```
use news;
```

`use` 语句可以通告MySQL把 `db_name` 数据库作为默认（当前）数据库使用，用于后续语句。该数据库保持为默认数据库，直到语段的结尾，或者直到发布一个不同的 `USE` 语句：

```
mysql> USE db1;
mysql> SELECT COUNT(*) FROM mytable;   # selects from db1.mytable
mysql> USE db2;
mysql> SELECT COUNT(*) FROM mytable;   # selects from db2.mytable
```

8. 选择的数据库：`select method()`

MySQL中 `SELECT` 命令类似于其他编程语言里的 `print` 或者 `write`，你可以用它来显示一个字符串、数字、数学表达式的结果等等。如何使用 `MySQL` 中 `SELECT` 命令的特殊功能?

① 显示MYSQL的版本

```
mysql> select version();
+-----------------------+
| version()             |
+-----------------------+
| 6.0.4-alpha-community |
+-----------------------+
1 row in set (0.02 sec)
```

② 显示当前时间

```
mysql> select now();
+---------------------+
| now()               |
+---------------------+
| 2009-09-15 22:35:32 |
+---------------------+
1 row in set (0.04 sec)
```

③ 显示年月日

```
SELECT DAYOFMONTH(CURRENT_DATE);
+--------------------------+
| DAYOFMONTH(CURRENT_DATE) |
+--------------------------+
|                       15 |
+--------------------------+
1 row in set (0.01 sec)

SELECT MONTH(CURRENT_DATE);
+---------------------+
| MONTH(CURRENT_DATE) |
+---------------------+
|                   9 |
+---------------------+
1 row in set (0.00 sec)
```

```
SELECT YEAR(CURRENT_DATE);
+--------------------+
| YEAR(CURRENT_DATE) |
+--------------------+
|               2009 |
+--------------------+
1 row in set (0.00 sec)
```

④ 显示字符串

```
mysql> SELECT "welecome to my blog!";
+----------------------+
| welecome to my blog! |
+----------------------+
| welecome to my blog! |
+----------------------+
1 row in set (0.00 sec)
```

⑤ 当计算器用

```
select ((4 * 4) / 10 ) + 25;
+----------------------+
| ((4 * 4) / 10 ) + 25 |
+----------------------+
|                26.60 |
+----------------------+
1 row in set (0.00 sec)
```

⑥ 串接字符串

```
select CONCAT(f_name, " ", l_name)
AS Name
from employee_data
where title = 'Marketing Executive';
+---------------+
| Name          |
+---------------+
| Monica Sehgal |
| Hal Simlai    |
| Joseph Irvine |
+---------------+
3 rows in set (0.00 sec)
```

> 注意：这里用到 `CONCAT()` 函数，用来把字符串串接起来。另外，我们还用到以前学到的 `AS` 给结果列 `'CONCAT(f_name, " ", l_name)'` 起了个假名。

9. 创建数据表：`create table table_name (field_1_name field_1_type [ ,... field_n_name field_n_type ])`

```
CREATE TABLE IF NOT EXISTS `user` (
  `uid` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `user_name` varchar(20) DEFAULT NULL,
  `user_password` varchar(32) DEFAULT NULL,
  `user_email` varchar(40) DEFAULT NULL,
  PRIMARY KEY (`uid`),
) ENGINE=MyISAM DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;
```

10. 获取表结构：`desc table_name` 或者 `show columns from table_name`

> 使用MySQL数据库desc 表名时，我们看到Key那一栏，可能会有4种值，即 `' '`， `'PRI'`，`'UNI'`， `'MUL'`。

① 如果 `Key` 是空的, 那么该列值的可以重复, 表示该列没有索引, 或者是一个非唯一的复合索引的非前导列;

② 如果 `Key` 是 `PRI` , 那么该列是主键的组成部分;

③ 如果 `Key` 是 `UNI` , 那么该列是一个唯一值索引的第一列(前导列)，并别不能含有空值(`NULL`);

④ 如果 `Key` 是 `MUL` , 那么该列的值可以重复, 该列是一个非唯一索引的前导列(第一列)或者是一个唯一性索引的组成部分但是可以含有空值 `NULL` 。 如果对于一个列的定义，同时满足上述4种情况的多种，比如一个列既是 `PRI` ，又是 `UNI` ，那么 `desc table_name` 的时候，显示的 `Key` 值按照优先级来显示 `PRI->UNI->MUL` 。那么此时，显示 `PRI` 。 一个唯一性索引列可以显示为 `PRI` ，并且该列不能含有空值，同时该表没有主键。 一个唯一性索引列可以显示为 `MUL` ，如果多列构成了一个唯一性复合索引，因为虽然索引的多列组合是唯一的，比如 `ID+NAME` 是唯一的，但是没一个单独的列依然可以有重复的值，只要 `ID+NAME` 是唯一的即可。

11. 删除表：`drop table table_name`

`DROP TABLE` 用于取消一个或多个表。您必须有每个表的 `DROP` 权限。所有的表数据和表定义会被取消，所以使用本语句要小心!

12. 表插入数据：`insert into table_name ( field_1_name [ ,... field_n_name ]) values ( value_1 [,... value_n ] )`

```
INSERT INTO user (`uid`, `user_name`, `user_password`, `user_email`) VALUES (1,
'admin', 'admin',  'admin@example.com');
```

`insert into` 每次只能向表中插入一条记录。

13. 查询表数据：`select field_1_name [,... field_n_name ] from table_name where sql_expression`

① 查询所有行：

查看表 `user` 中所有数据 `select * from user;`

② 查询前几行数据：

查看表 user 中前2行数据 `select * from user order by id limit 0,2;`

> 注：`select` 一般配合 `where` 使用，以查询更精确更复杂的数据。

14. 删除表中数据：`delete from table_name where sql_expression`

删除表 user 中编号为1 的记录 `delete from user where uid=1;`

15. 修改表中数据：`update table_name set field_name = new_value [ ,…] where sql_expression`

如更新 `id` 为 `1` 的 `user`，设置 `user_name`字段值为 `Mary`。

```
update user set user_name='Mary' where id=1;
```

① 单表的MySQL UPDATE语句：

```
UPDATE [LOW_PRIORITY] [IGNORE] tbl_name SET col_name1=expr1 [, col_name2=expr2
...] [WHERE where_definition] [ORDER BY ...] [LIMIT row_count]
```

② 多表的UPDATE语句：

```
UPDATE [LOW_PRIORITY] [IGNORE] table_references SET col_name1=expr1 [,
col_name2=expr2 ...] [WHERE where_definition]
```

`UPDATE` 语法可以用新值更新原有表行中的各列。`SET` 子句指示要修改哪些列和要给予哪些值。`WHERE` 子句指定应更新哪些行。如果没有 `WHERE` 子句，则更新所有的行。如果指定了 `ORDER BY` 子句，则按照被指定的顺序对行进行更新。 `LIMIT` 子句用于给定一个限值，限制可以被更新的行的数目。

16. 增加字段：`alter table table_name [ add field_name field_type / other_sql_expression ]`

在表 `user` 中添加了一个字段 `user_pic` ，类型为 `varchar(40)`，默认值为 `NULL` `alter table user add user_pic varchar(40) default NULL;`

加索引 `alter table employee add index emp_name (name);`

加主关键字的索引 `alter table employee add primary key(id);`

加唯一限制条件的索引 `alter table employee add unique emp_name2(cardnumber);`

删除某个索引 `alter table employee drop index emp_name;`

增加字段 `alter table user add user_pic varchar(40) default NULL;`

修改原字段名称及类型 `ALTER TABLE table_name CHANGE old_field_name new_field_name field_type;`

删除字段 `ALTER TABLE table_name DROP field_name;`

17. 修改表名：`rename table old_table_name to new_table_name`

当你执行 `RENAME` 时，你不能有任何锁定的表或活动的事务。你同样也必须有对原初表的 `ALTER` 和 `DROP` 权限，以及对新表的 `CREATE` 和 `INSERT` 权限。 如果在多表更名中，`MySQL` 遭遇到任何错误，它将对所有被更名的表进行倒退更名，将每件事物退回到最初状态。

18.备份数据库：

① 导出整个数据库，导出文件默认是存在mysql\bin目录下 `mysqldump -uuser_name -puser_password db_name > new_db_name.sql`

② 导出一个表 `mysqldump -uuser_name -puser_password database_name table_name > outfile_name.sql`

③ 导出一个数据库结构 `mysqldump -uuser_name -puser_password -d -add-drop-table database_name > outfile_name.sql` `-d` 没有数据 `-add-drop-table` 在每个 `create` 语句之前增加一个 `drop table`

④带语言参数导出 `mysqldump -u user_name -p user_password -default-character-set=latin1 -set-charset=gbk -skip-opt database_name > outfile_name.sql`

19. 建库建表示例：

```
drop database if exists school; //如果存在SCHOOL则删除
create database school; //建立库SCHOOL
use school; //打开库SCHOOL
create table teacher //建立表TEACHER
(
    id int(3) auto_increment not null primary key,
    name char(10) not null,
    address varchar(50) default '深圳',
    year date
); //建表结束

//以下为插入字段
insert into teacher values('','allen','大连一中','1976-10-10');
insert into teacher values('','jack','大连二中','1975-12-23');
```

如果你在 `mysql` 提示符键入上面的命令也可以，但不方便调试。

① 你可以将以上命令原样写入一个文本文件中，假设为 `school.sql` ，然后复制到 `c:\` 下，并在 `DOS` 状态进入目录 `mysql\bin` ，然后键入以下命令： `mysql -u user_name -p user_password < c:\school.sql`

如果成功，空出一行无任何显示；如有错误，会有提示。（以上命令已经调试，你只要将//的注释去掉即可使用）。

② 或者进入命令行后使用 `mysql> source c:\school.sql;` 也可以将 `school.sql` 文件导入数据库中。

第二、常见应用场景分析？非常重要!!

- **多表链接查询?**

多　表　连　接

案例: 电商订单详情页多方面信息综合展示

SQL语句: 多表连接

应用场景:

1.员工、部门、 工资等级
2.书籍、类别、出版社
3.订单、详情、物流

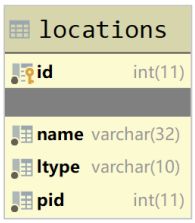
- **省市区级联设计?**

多　　级　　级　　联

效果：

序号	名字	邮箱	地址
1	张三	zs@qq.com	山东省-济南市-历下区-趵北路

表结构：



id	name	ltype	pid
1	山东省	省	-1
2	山西省	省	-1
3	北京市	直辖市	-1
4	济南市	市	1
5	青岛市	市	1
6	天桥区	区	4
7	历下区	区	4
8	济阳县	县	4
9	市南区	区	5
10	市北区	区	5

SQL语句

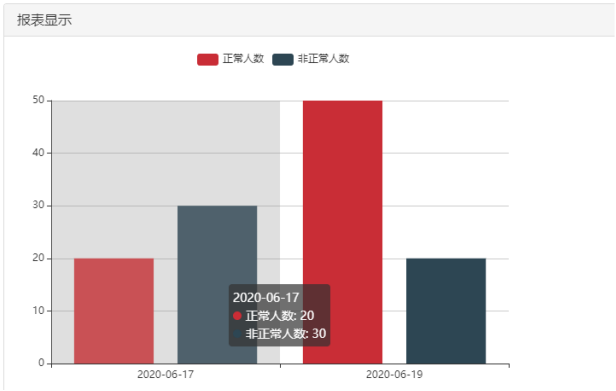
```
select a.id,flag,concat_ws('-',pro.name,city.name ,area.name),a.info
       from address  a left join locations pro on a.proid = pro.id
            left join locations city on a.cityid = city.id
            left join locations area on a.area = area.id;
```

id	flag	`concat_ws('-',pro...	info
1	家	山东省-济南市-历下区	趵北路6号401
2	公司	山西省-太原市-小店区	趵北路6号411
3	老家	山东省-青岛市-市南区	海洋大厦4

- **数据报表统计?**

报　表　展　示

效果:



分析报表背后的要查询的数据

SQL:

```
select returndate,sum(decode(health,'正常',1,0)) as "正常",(count(*)-sum(decode(health,'正常',1,0))) as "非正常"
from person group by returndate having sysdate-returndate<=30;
```

- ## 权限系统设计?

RBAC(Role-Based Access Control)基于角色的权限控制

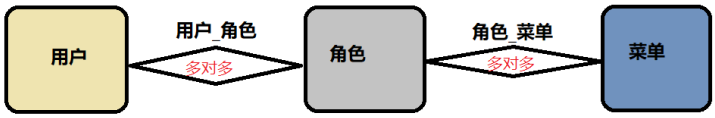
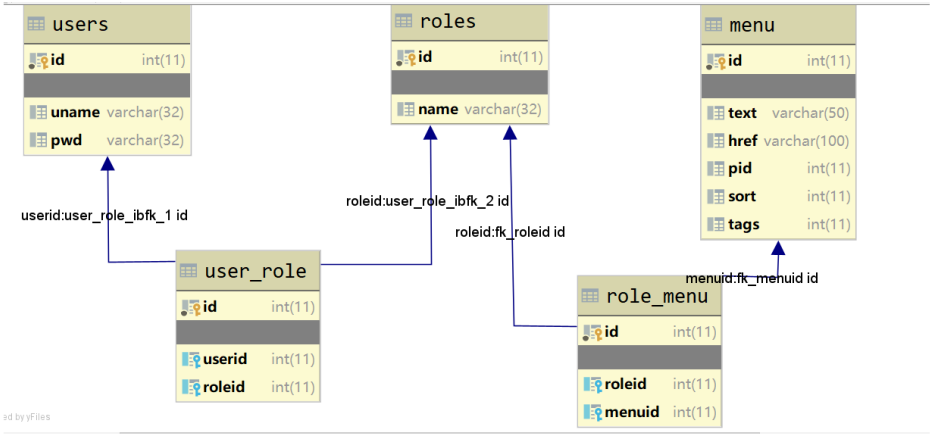
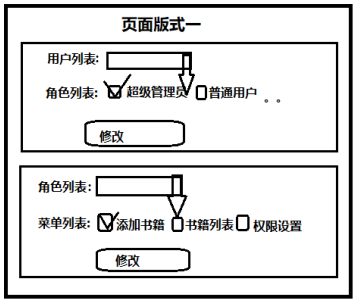


表　　设　　计



页　　面



页面版式二

序号	用户名	角色名	菜单名
1	etoak	超级管理员，普通用户	添加书籍、书籍列表、添加类别。。
2	et	普通用户	查询书籍，查询类别，查询。。
3			
。。。			

SQL　　语　　句

1.根据用户查询角色

```
select * from roles where id in (select roleid from user_role where userid=?)
```

2.根据角色查询菜单

```
select * from menu where id in( select menuid from role_menu where roleid=?)
```

3.一起查询

```
select u.id,u.uname ,group_concat(distinct (r.name)),group_concat(m.text)
from users u
    left join user_role ur
        on u.id = ur.userid
    left join roles r
        on ur.roleid = r.id
    left join role_menu rm
        on ur.roleid = rm.roleid
    left join menu m on m.id = rm.menuid group by u.id
```