MySQL 基本使用

导入

在信息化社会，充分有效地管理和利用各类信息资源，是进行科学研究和决策管理的前提条件。数据库技术是管理信息系统、办公自动化系统、决策支持系统等各类信息系统的核心部分，是进行科学研究和决策管理的重要技术手段。

目录

目标

1. 掌握数据库相关概念，并且能够在 Linux 系统上快速安装 MySQL 数据库服务器和客户端（重点）
2. 能够根据软件业务，设计数据表，并且创建对应数据库、表（重点、难点）
3. 能够根据具体业务需求，进行简单查询和数据操作（重点）
4. 能够快速备份和恢复数据库，并养成备份数据库的好习惯

一、数据库基础

1、数据库基础概述

数据管理主要经历过程：

- 手工管理阶段：应用程序管理数据、数据不保存、不共享、不具有独立性。
- 文件管理阶段：文件系统管理数据、数据可长期保存、但共享性差、冗余度大、独立性差。
- 数据管理阶段：数据库系统管理数据、数据结构复杂、冗余小、易扩充、较高的独立性、统一数据控制。

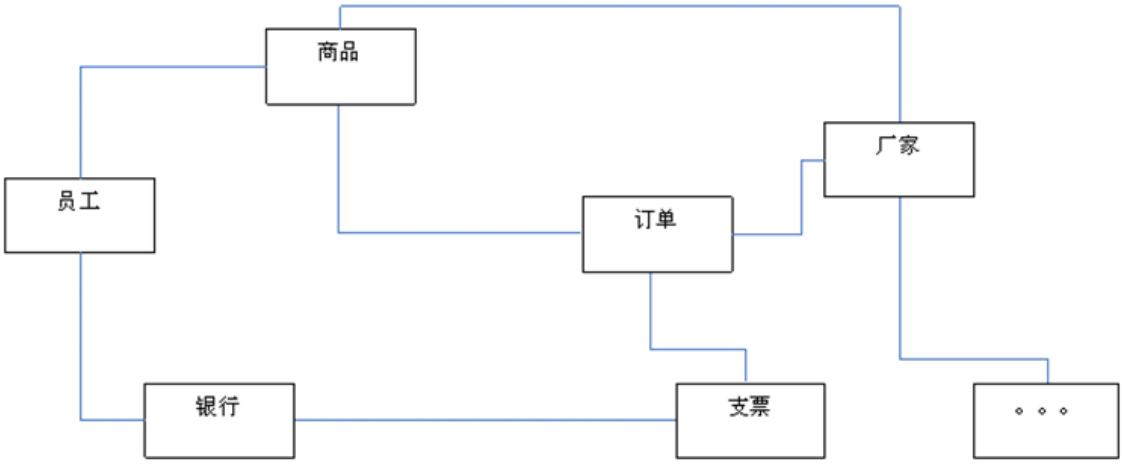
数据库的特征：

- 数据结构化
- 实现数据共享
- 减少数据冗余
- 数据独立性

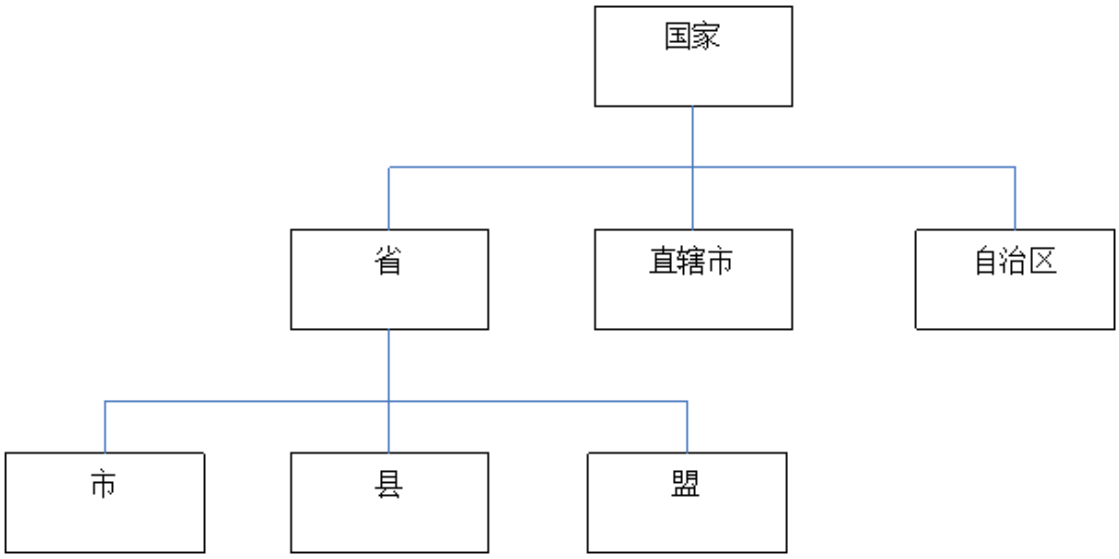
数据库类型（按数据模型特点分）

- 网状型数据库
- 层次型数据库
- 关系型数据库

网状数据库：采用记录类型为节点的网状数据模型



层次数据库：采用层次模型模拟现实世界中按层次组织起来的事物



关系型数据库：采用二维表结构组织和管理数据，并规定了表内和表间数据的依赖关系

学号	姓名	性别	电话
Stu10001	张力	男	83620089
Stu10002	王南	女	86368888
Stu10004	李那	女	23910001

学号	科目	成绩
Stu10001	英语	80
Stu10001	高数	90
Stu10002	物理	60

关系数据库是指一些相关的表和其他数据库对象的集合。对于关系数据库来说，关系就是表的同义词。

表是由行和列组成（类似二维数组的结构）。

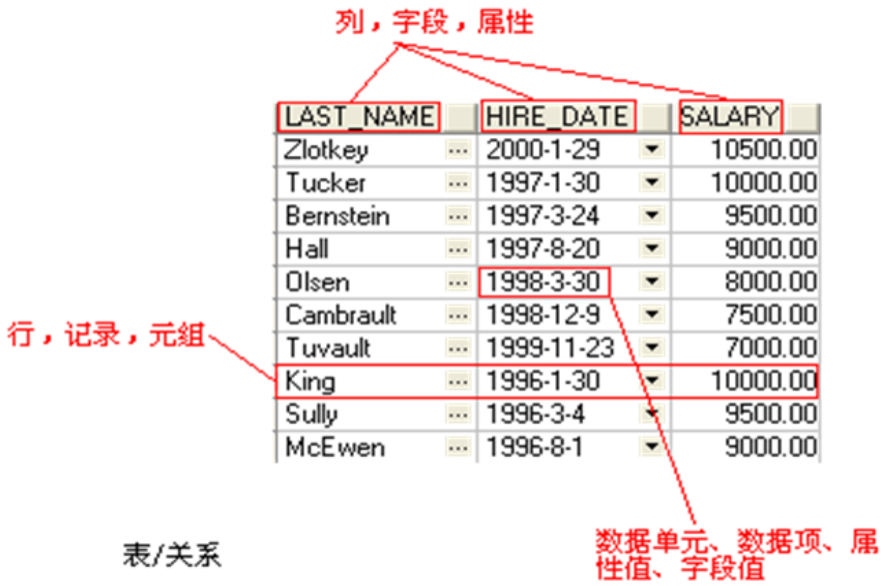
列包含一组命名的属性（也称字段）。

行包含一组记录，每行包含一条记录。

行和列的交集称为数据项，指出了某列对应的属性在某行上的值，也称为字段值。

列需定义数据类型，比如整数或者字符型的数据。

关系数据库的数据结构图示：



常见的数据库

Rank Dec 2017	Rank Nov 2017	Rank Dec 2016	DBMS	Database Model
1.	1.	1.	Oracle	Relational DBMS
2.	2.	2.	MySQL	Relational DBMS
3.	3.	3.	Microsoft SQL Server	Relational DBMS
4.	4.	4.	PostgreSQL	Relational DBMS
5.	5.	5.	MongoDB	Document store
6.	6.	6.	DB2	Relational DBMS
7.	7.	↑ 8.	Microsoft Access	Relational DBMS
8.	↑ 9.	↑ 9.	Redis	Key-value store
9.	↓ 8.	↓ 7.	Cassandra	Wide column store
10.	10.	↑ 11.	Elasticsearch	Search engine
11.	11.	↓ 10.	SQLite	Relational DBMS
12.	12.	12.	Teradata	Relational DBMS
13.	13.	↑ 14.	Solr	Search engine
14.	14.	↓ 13.	SAP Adaptive Server	Relational DBMS
15.	15.	↑ 16.	Splunk	Search engine

2、MySQL 数据库介绍

MySQL 是由瑞典 MySQL AB 公司开的一种开放源代码的关系型数据库管理系统（RDBMS），目前属于 Oracle 旗下产品。MySQL 数据库系统使用最常用的数据库管理语言——结构化查询语言（SQL）进行数据库管理。由于 MySQL 是开放源代码的，因此任何人都可以在 General Public License 的许可下下载并根据个性化的需要对其进行修改。MySQL 因为其速度、可靠性和适应性而备受关注。

SQL 语言主要是用来操作关系型数据库的一本语言，称之为结构化查询语句。

SQL 语句主要分为：

- DQL：数据查询语言，用于对数据进行查询，如 select
- DML：数据操作语言，对数据进行增加、修改、删除，如 insert、udpate、delete

- TPL：事务处理语言，对事务进行处理，包括 begin transaction、commit、rollback
- DCL：数据控制语言，进行授权与权限回收，如 grant、revoke
- DDL：数据定义语言，进行数据库、表的管理等，如 create、drop
- CCL：指针控制语言，通过控制指针完成表的操作，如 declare cursor

MySQL 的特点：

- 使用 C 和 C++编写，并使用了多种编译器进行测试，保证源代码的可移植性
- 全面支持 SQL 的 GROUP BY 和 ORDER BY 子句，支持聚合函数(COUNT()、COUNT(DISTINCT)、AVG()、STD()、SUM()、MAX()和MIN())。你可以在同一查询中混来自不同数据库的表。
- 为多种编程语言提供了 API，如 C、C++、Python、Java、Perl、PHP、Eiffel、Ruby 等
- 支持多种存储引擎
- MySQL 软件采用了双授权政策，它分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放源码这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库

3、MySQL 数据库安装

官方地址，跟随老师安装一遍

```
https://www.mysql.com/
```

MySQL 服务端（在 Linux 系统）

下载安装

```
sudo apt-get install mysql-server
```

启动服务

```
sudo server mysql start
```

查看服务是否启动

```
ps ajx|grep mysql
```

```
sudo server mysql status
```

停止服务

```
sudo server mysql stop
```

重启服务

```
sudo service mysql restart
```

配置：

配置文件目录为/etc/mysql/mysql.conf.d

进入目录，打开 mysqld.cnf，可以看到配置项

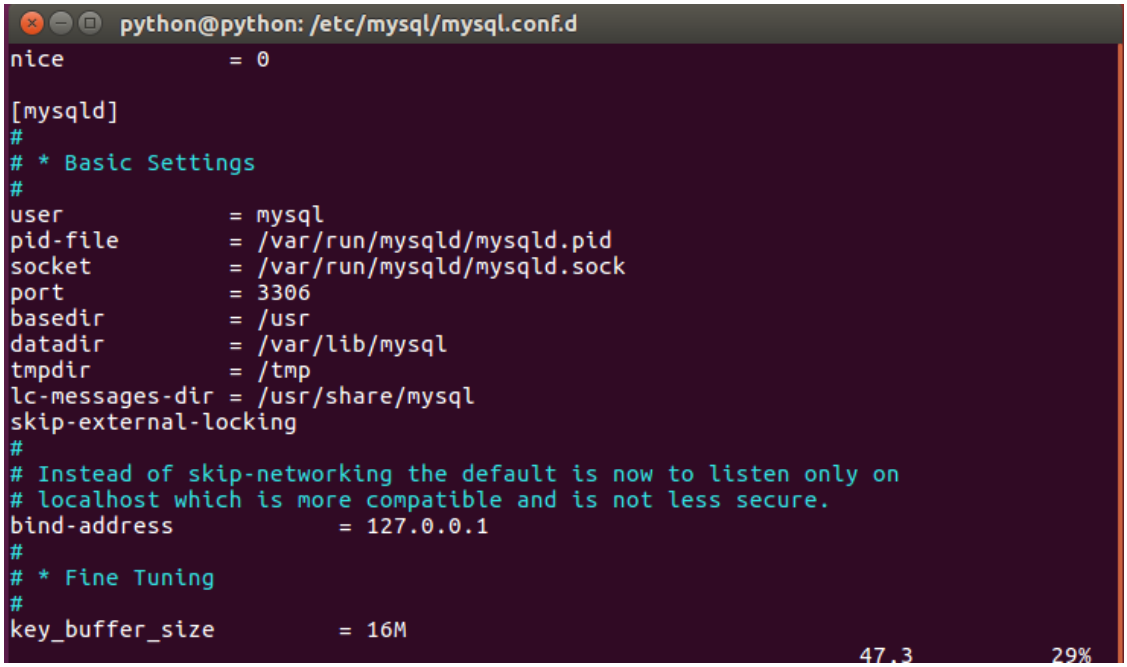
bind-address 表示服务器绑定的 ip，默认为 127.0.0.1

port 表示端口，默认为 3306

datadir 表示数据库目录，默认为/var/lib/mysql

general*log*file 表示普通日志，默认为/var/log/mysql/mysql.log

log_error 表示错误日志，默认为/var/log/mysql/error.log



MySQL 客户端

客户端为开发人员使用，常用的有命令行客户端、navicat 图形界面客户端等。

下载安装命令行客户端

```
sudo apt install mysql-client
```

连接数据库

```
mysql -u root -p123456
```

-u 后面跟的是数据库的账户名，-p 密码 -p 与密码之间不能有空格，看到下面的提示表示已经连接数据库。

如果-p 后面不加密码，那么回车后会要求输入密码。

```
timber@ubuntu: ~
timber@ubuntu:~$ mysql -uroot -p123456
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 4
Server version: 5.7.22-0ubuntu0.16.04.1 (Ubuntu)

Copyright (c) 2000, 2018, 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>
```

按 ctrl+d 或输入 quit 或者 exit 命令退出

二、数据库和数据表管理

1、数据库管理

连接数据库

mysql -u 账号 -p 密码 -h 主机地址 -P 端口

```
mysql -uroot -pmysql
```

查看数据库版本

```
select version();
```

显示当前时间

```
select now();
```

查看所有数据库：

```
show databases;
```

创建数据库

```
create database 数据库名 charset=utf8;
```

注意：创建库的时候一定要指定编码 utf8，utf8 中间没有-，跟 pyhton 中写编码有点区别

```
mysql> create database mydjango charset=utf8;
Query OK, 1 row affected (0.01 sec)

mysql>
```

切换数据库：

```
use 数据库名
```

查看当前正在使用哪个数据库

```
select database();
```

删除数据库

```
drop database 数据库名;
```

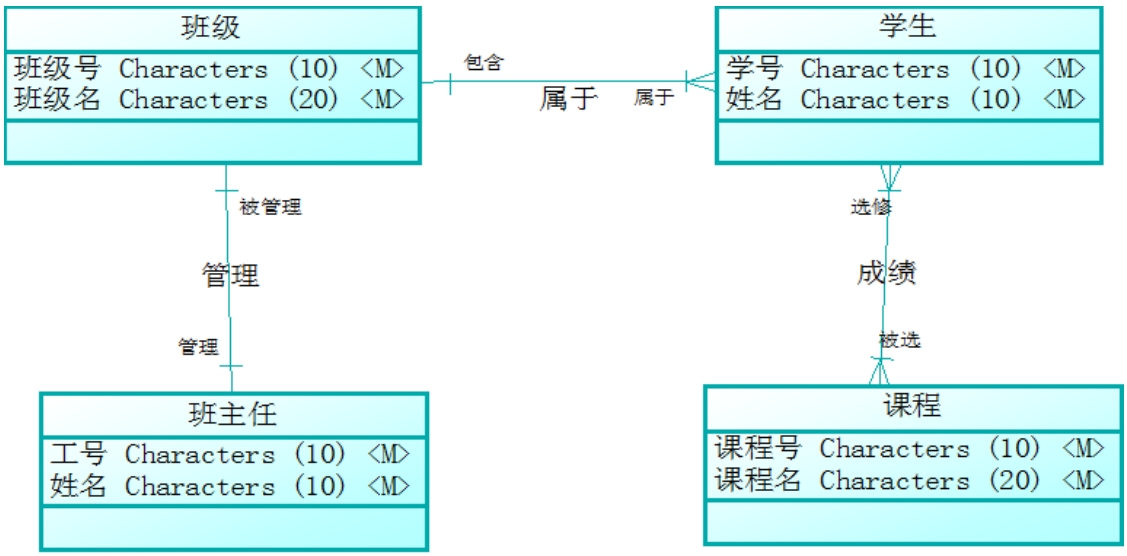
```
mysql> drop database mydjango;
Query OK, 0 rows affected (0.03 sec)

mysql>
```

2、数据表管理

（1）数据表设计

数据表设计包括 ER 图、表的主键、字段、数据类型、约束、表之间关系的设计

E-R（Entity-Relationship）模型即实体-关系模型主要用于定义数据的存储需求，该模型已经广泛用于关系数据库设计中。E-R 模型由实体、属性和关系三个基本要素构成。



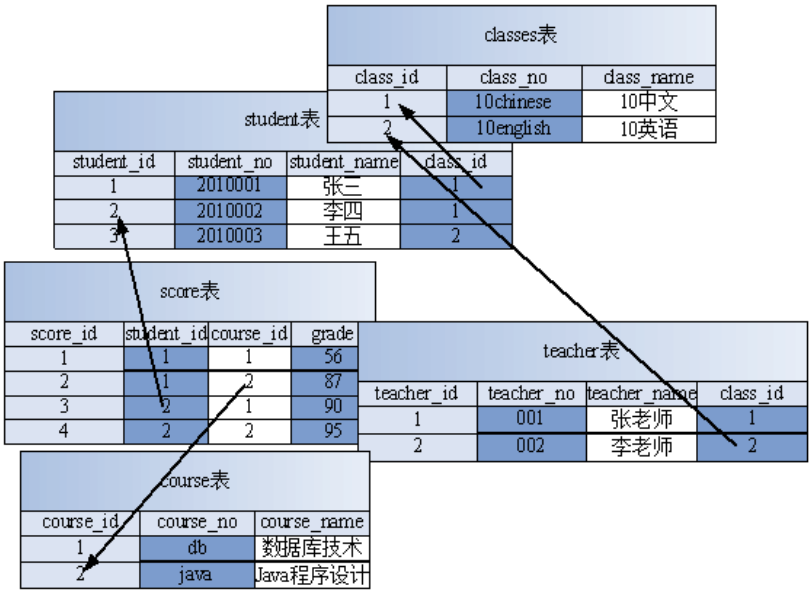
主键（Primary Key） 数据库表要求表中的每一行记录都必须是唯一的，即在同一张表中不允许出现完全相同的两条记录。在设计数据库时，为了保证记录的“唯一性”，最为普遍、最为推荐的做法是为表定义一个主键（primary key）。数据库表中主键有以下两个特征：

- 表的主键可以由一个字段构成，也可以由多个字段构成（这种情况称为复合主键）。
- 数据库表中主键的值具有唯一性且不能取空值（NULL），当数据库表中的主键由多个字段构成时，每个字段的值不能取 NULL 值。

实体间的关系与外键（Foreign Key）

班级实体和班主任实体之间为一对一关系，班级实体和学生实体之间为一对多关系，学生实体和课程实体之间为多对多关系。

实体间的关系可以通过外键来表示。如果表 A 中的一个字段 a 对应于表 B 的主键 b，则字段 a 称为表 A 的外键。此时存储在表 A 中字段 a 的值，同时这个字段值也是表 B 主键 b 的值。



约束（Constraint）

约束是定义在表上的一种强制规则。当为某个表定义约束后，对该表做的所有 SQL 操作都必须满足约束的规则要求，否则操作将失败。

约束类型

约束	说明
NOT NULL	非空约束，指定某列的所有行数据不能包含空值
UNIQUE	唯一性约束，指定列或者列的组合 的所有行数据必须唯一
PRIMARY KEY	外键约束，在列及引用列上建立的一种强制依赖关系
FOREIGN KEY	外键约束，在列及引用列上建立的一种强制依赖关系
CHECK	检查性约束，在列上指定一个必须满足的条件

（2）创建表

查看当前数据库中的表

```
show tables;
```

创建表

```
create table 表名(
id int unsigned auto_increment  primary key not null,
name varchar(10) not null,
is_delete bit(1) not null default 0
);
```

建表主要是前面是字段，字段后面跟的是约束条件。

创建学生表

```
create table students(
id int auto_increment primary key not null,
name varchar(10) not null,
gender bit(1) default 0,
hometown varchar(40) default ""
)
```

comment 注释，在创建表的时候如果字段很多，防止忘记字段是存什么数据的，可以给字段添加注释

```
create table students(
id int auto_increment primary key not null comment '主键',
name varchar(10) not null comment '学生姓名',
gender bit(1) default 0 comment '性别',
hometown varchar(40) default "" comment '家乡地址'
)
```

查看创建表的 sql 语句

```
show create table 表名;
```

（3）修改表

添加字段


```
alter table 表名 add 列名 类型;
```

给 students 添加一个生日字段

```
mysql> alter table students add birthday date;
Query OK, 0 rows affected (0.13 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

修改字段

第一种，不修改字段名只修改类型及约束

```
alter table 表名 modify 列名 类型及约束;
```

```
mysql> alter table students modify hometown varchar(40) default N
Ull;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

第二种，需要修改字段名字

```
alter table 表名 change 原名 新名 类型及约束;
```

将 class 表的 is_delete 字段修改为 delete

```
alter table class change is_delete delete bit(1) NOT NULL DEFAULT
 b'0';
```

删除字段

```
alter table 表名 drop 字段名字;
```

将 students 表中的 gender 字段删除

```
mysql>  alter table students drop gender;
Query OK, 0 rows affected (0.10 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

（4）删除表

```
drop table 表名;
```

删除学生表

```
mysql>  drop table students;
Query OK, 0 rows affected (0.02 sec)

mysql>
```

3、图形化界面操作数据库

下载安装图形界面工具 Navicat

Navicat 官网：https://www.navicat.com.cn/

mysql 刚装好 root 账号默认只能本地登录，不能在其他机器登录的。使用 Navicat 连接之前，先在命令行客户端将 mysql 的用户登录权限进行修改；

修改步骤：

（1）mysql -uroot -p123456 连接数据库

（2）use msyql 进入 mysql 数据库

（3）select host,user from user; 查看账号有哪些权限

```
mysql> select host,user from user;
+-----------+------------------+
| host      | user             |
+-----------+------------------+
| localhost | debian-sys-maint |
| localhost | mysql.session    |
| localhost | mysql.sys        |
| localhost | root             |
+-----------+------------------+
4 rows in set (0.00 sec)

mysql>
```

（4）将 root 登录权限修改成所有主机都能登录

```
grant all privileges on *.* to 'root'@'%';
```

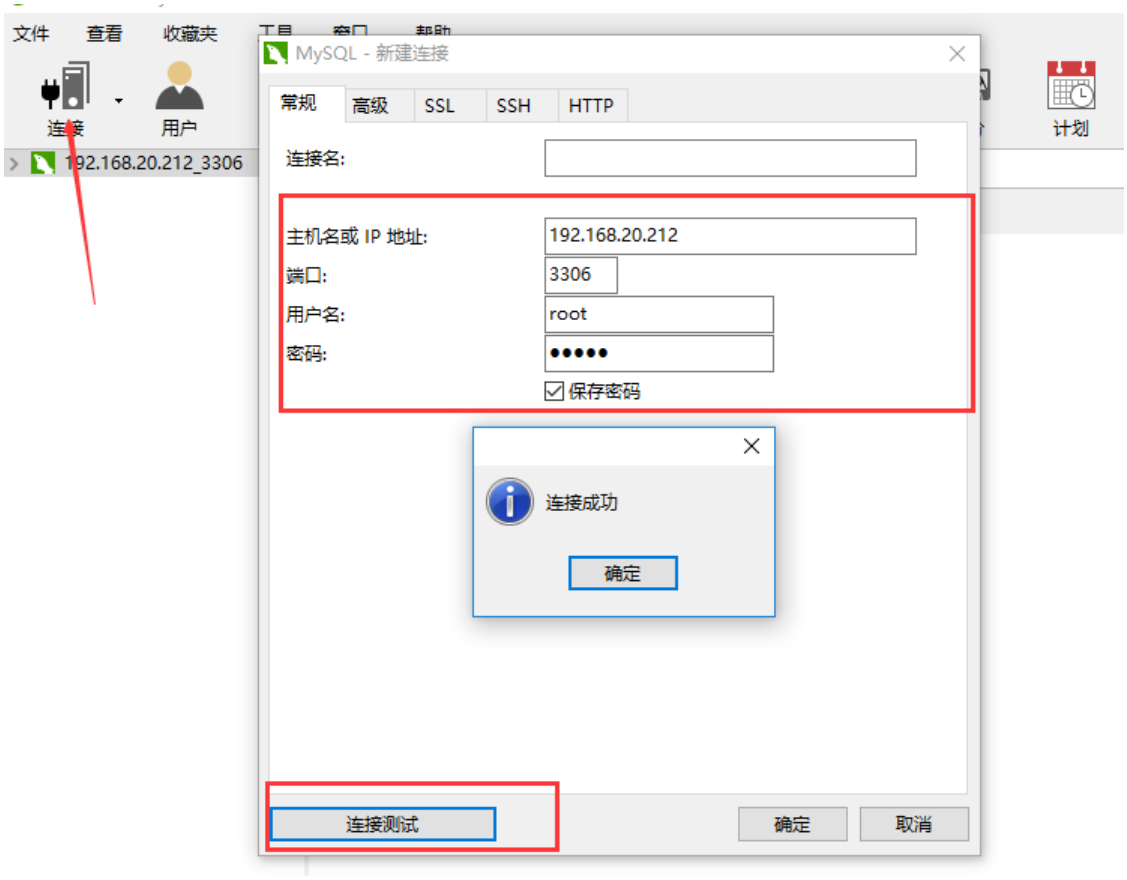
```
mysql> select user,host from user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| root             | %         |
| debian-sys-maint | localhost |
| mysql.session    | localhost |
| mysql.sys        | localhost |
| root             | localhost |
+------------------+-----------+
5 rows in set (0.00 sec)
```

（5）注释配置文件的 bind-address

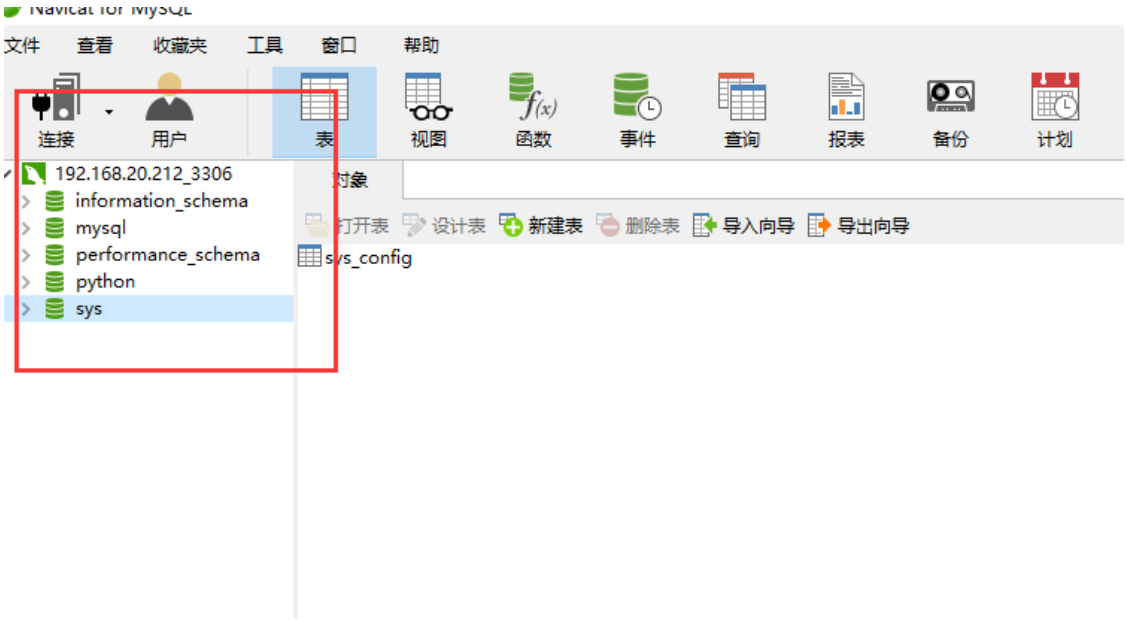
把/etc/mysql/mysql.conf.d/mysqld.cnf 配置文件里面的 bind-address 注释掉

使用 Navicat 连接 mysql

打开安装好的 Navicat 客户端点击 连接--->mysql----填写账号密码，主机地址是你安装 msyql 的 ubuntu ip 地址，点击连接测试弹出连接成功，说明 Navicat 已经连接上了 mysql，点击确定。

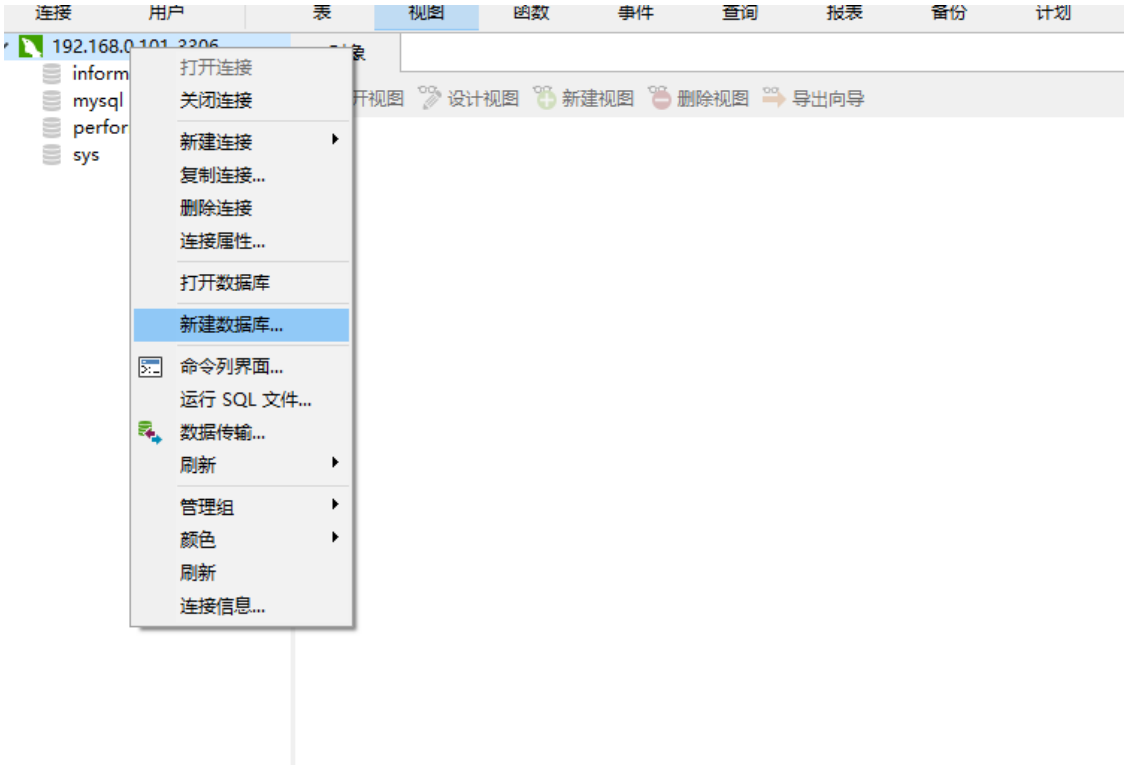


连接成功后可以看到所有数据库

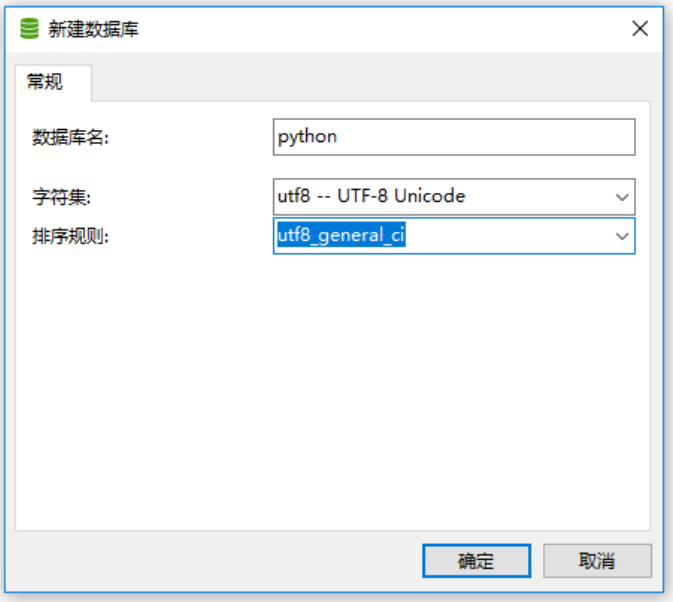


创建数据库

在左侧栏空白处右击，选择“新建数据库”，点击

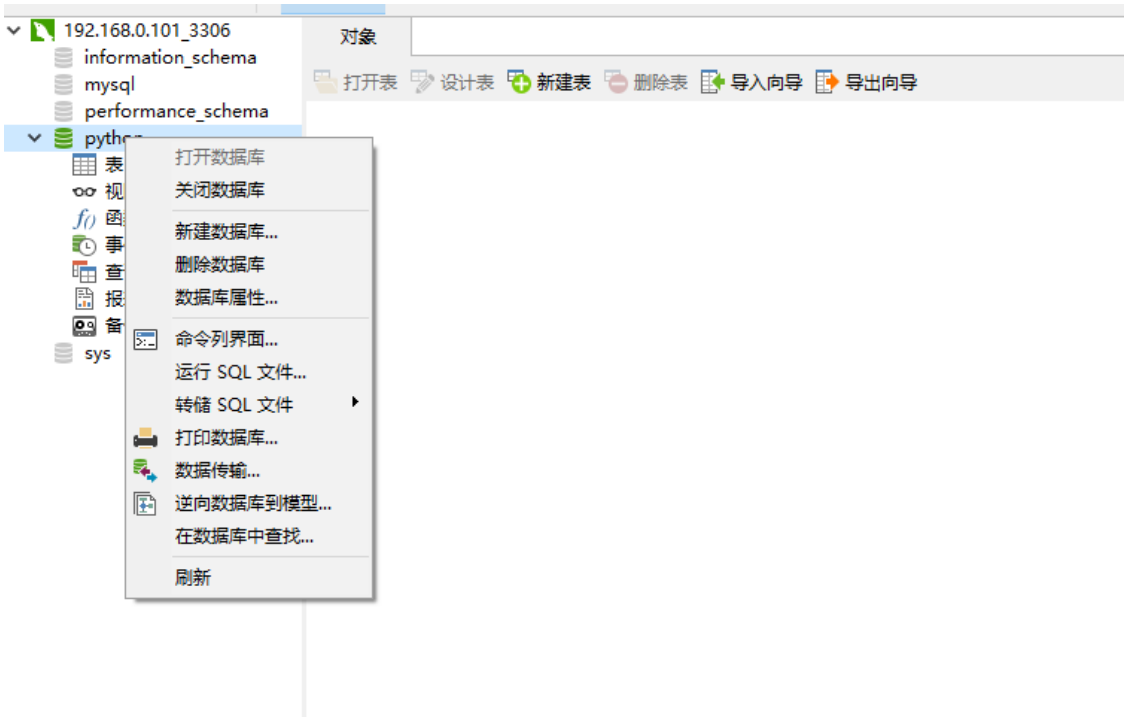


在弹出框里填写数据库名，编码格式，选择 utf-8



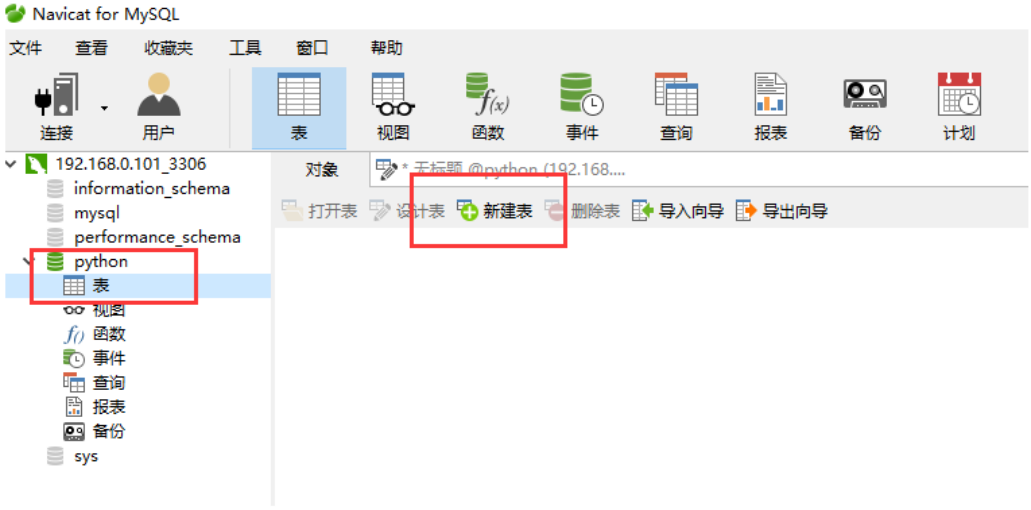
编辑数据库

选择刚才创建的数据库右击可以编辑数据库



创建数据表

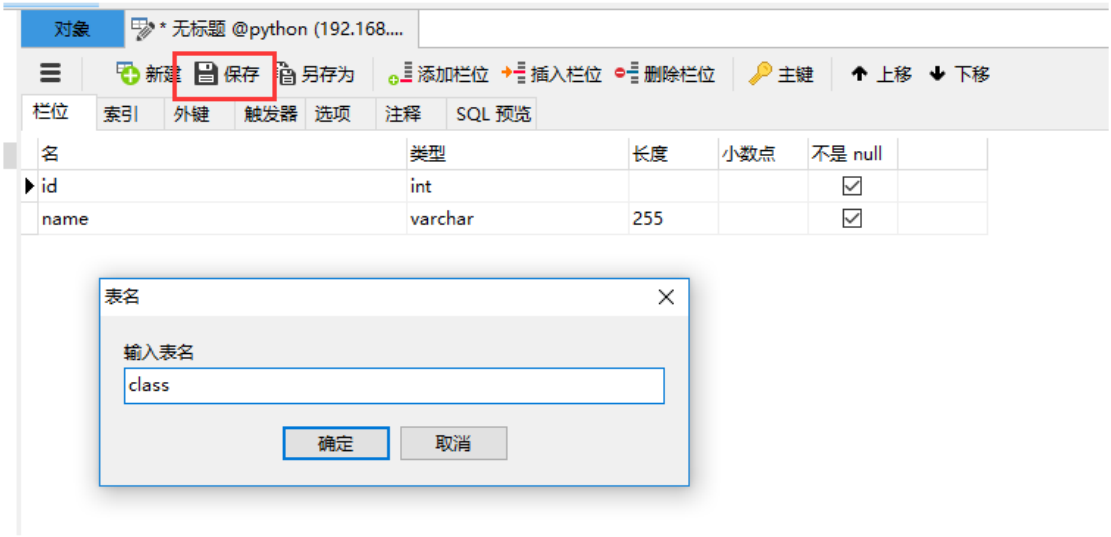
选择表然后点击新建表



弹出的新标签页中按照设计创建表

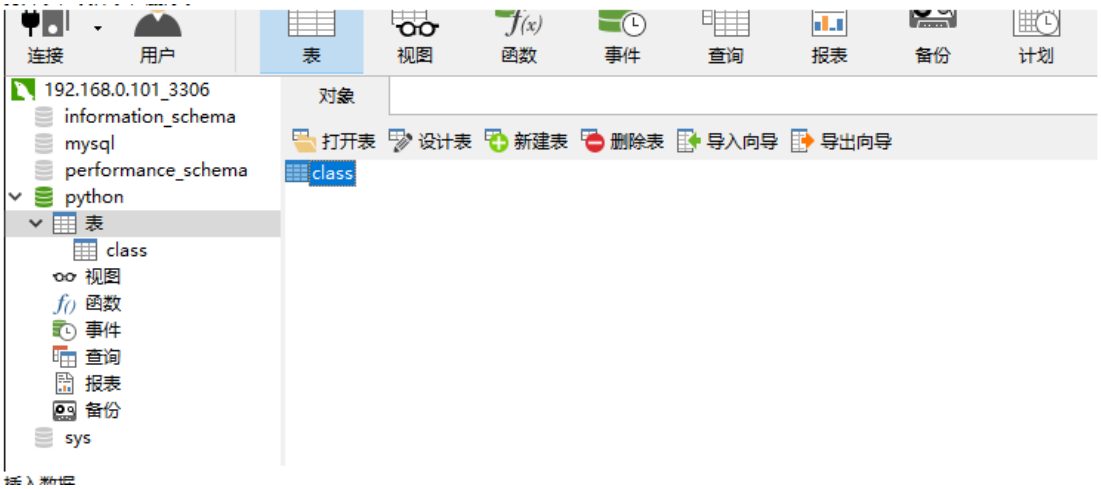
创建一个班级表

对于 id 字段，需要设置为 int 类型，无符号，自动增长，主键，非空



创建好表后，还可以对表进行编辑

打开表，设计表，删除表



三、简单查询与数据操作

1、基本查询语句

select * from 表名;

```
select * from students;
```
查询 students 表中的所有内容

```
mysql> show tables;
+----------------+
| Tables_in_test |
+----------------+
| goods          |
| score          |
| students       |
| subject        |
| t1             |
+----------------+
5 rows in set (0.00 sec)

mysql> select * from students;
+----+--------------+--------+
| id | name         | gender |
+----+--------------+--------+
|  1 | 诸葛亮       | 男     |
|  2 | 安其拉       |        |
|  3 | 白起         | 男     |
|  4 | 不知火舞     |        |
|  5 | 妲己         |        |
|  6 | 狄仁杰       | 男     |
|  7 | 小乔         |        |
|  8 | 芈月         |        |
|  9 | 李白         | 男     |
| 10 | 吕布         | 男     |
| 11 | 嬴政         | 男     |
| 12 | 王昭君       |        |
+----+--------------+--------+
12 rows in set (0.00 sec)
```

指定字段查询

`select 字段1，字段2  from 表名;`

比如只想看 id，name 这两列

`select id,name from students;`

```
mysql>   select id,name from students;
+----+--------------+
| id | name         |
+----+--------------+
|  1 | 诸葛亮       |
|  2 | 安其拉       |
|  3 | 白起         |
|  4 | 不知火舞     |
|  5 | 妲己         |
|  6 | 狄仁杰       |
|  7 | 小乔         |
|  8 | 芈月         |
|  9 | 李白         |
| 10 | 吕布         |
| 11 | 嬴政         |
| 12 | 王昭君       |
+----+--------------+
12 rows in set (0.00 sec)
```

2、插入数据

（1）全列插入

```
insert into 表名 values (......)
```

在上节课中创建的 students 学生表中插入学生信息

```
insert into students values(0,'韩信',0,'广州');
```

```
mysql> desc students;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| id       | int(11)     | NO   | PRI | NULL    | auto_increment |
| name     | varchar(10) | NO   |     | NULL    |                |
| gender   | bit(1)      | YES  |     | b'0'    |                |
| hometown | varchar(40) | YES  |     |         |                |
+----------+-------------+------+-----+---------+----------------+
4 rows in set (0.22 sec)

mysql> insert into students values(0,'韩信',0,'广州');
Query OK, 1 row affected (0.14 sec)
```

注意全列插入时，有多少个字段，必须插入多少个字段，即使默认可以为空的字段也要占位。主键自增也需要占位一般使用 0 占位。

（2）部分插入

```
insert into 表名 [字段 1，字段 2] values (值 1，值 2);
```

```
2 rows in set (0.00 sec)

mysql> insert into students (name,gender) values ('程咬金',0);
Query OK, 1 row affected (0.06 sec)

mysql> select * from students;
+----+-----------+--------+----------+
| id | name      | gender | hometown |
+----+-----------+--------+----------+
|  1 | 韩信      |        | 广州     |
|  2 | 荆轲      |        | 广州     |
|  3 | 程咬金    |        |          |
+----+-----------+--------+----------+
3 rows in set (0.00 sec)

mysql> insert into students (name,gender) values ('貂蝉',1);
Query OK, 1 row affected (0.00 sec)

mysql> select * from students;
+----+-----------+--------+----------+
| id | name      | gender | hometown |
+----+-----------+--------+----------+
|  1 | 韩信      |        | 广州     |
|  2 | 荆轲      |        | 广州     |
|  3 | 程咬金    |        |          |
|  4 | 貂蝉      | 01     |          |
```

（3）全列多行插入

多行插入每一行的内容写在一个小括号内，用逗号分隔多行。

```
insert into 表名 values (...),(....),(....);
```

```
mysql> mysql> insert into students values(0,'后羿',0,'广州'),(0,1,'广州'),(0,'嬴政'
,0,'广州
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
```

（4）部分列多行插入

```
insert into 表名 (字段1，字段2) values (..),(..);
```

```
mysql> insert into students (name,gender,hometown) values('狄仁杰',0,'深圳'),('鲁班
七号',1,'深圳'),('孙尚香',1,'深圳');
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

3、修改数据

```
update 表名 set 字段=xxx where 字段=xxx;
```

注意：修改某一行内容一定要加 where 限定条件，否则会造成全表修改，除非你想要修改整张表。

```
update students set hometown='珠海' where id= 5;
```

```
mysql> update students set  hometown='珠海' where id= 5;
Query OK, 1 row affected (0.08 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from students;
+----+--------------+--------+----------+
| id | name         | gender | hometown |
+----+--------------+--------+----------+
|  1 | 韩信         |        | 广州     |
|  2 | 荆轲         |        | 广州     |
|  3 | 程咬金       |        |          |
|  4 | 貂蝉         | 00     |          |
|  5 | 后羿         |        | 珠海     |
|  6 | 芈月         | 00     | 广州     |
|  7 | 嬴政         |        | 广州     |
|  8 | 狄仁杰       |        | 深圳     |
|  9 | 鲁班七号     | 00     | 深圳     |
| 10 | 孙尚香       | 00     | 深圳     |
+----+--------------+--------+----------+
10 rows in set (0.00 sec)
```

看下不加 where 限定条件的结果将 gender 改成女（表中保存的是 1）

```
mysql>
mysql> update  students set gender=1;
Query OK, 6 rows affected (0.01 sec)
Rows matched: 10  Changed: 6  Warnings: 0

mysql> select * from students;
+----+--------------+--------+----------+
| id | name         | gender | hometown |
+----+--------------+--------+----------+
|  1 | 韩信         |      1 | 广州     |
|  2 | 荆轲         |      1 | 广州     |
|  3 | 程咬金       |      1 |          |
|  4 | 貂蝉         |      1 |          |
|  5 | 后羿         |      1 | 珠海     |
|  6 | 芈月         |      1 | 广州     |
|  7 | 嬴政         |      1 | 广州     |
|  8 | 狄仁杰       |      1 | 深圳     |
|  9 | 鲁班七号     |      1 | 深圳     |
| 10 | 孙尚香       |      1 | 深圳     |
+----+--------------+--------+----------+
10 rows in set (0.00 sec)
```

所以不是全表修改的一定要记得加 where 限定条件

4、删除数据

```
delete from students where id =3;
```

删除 id 为 3 的程咬金，删除行也要加限定条件，不加的话会造成全表删除。

```
mysql> delete from  students where id = 3;
Query OK, 1 row affected (0.01 sec)

mysql>
mysql>
mysql>
mysql>
mysql> select * from students;
+----+--------------+--------+----------+
| id | name         | gender | hometown |
+----+--------------+--------+----------+
|  1 | 韩信         |      1 | 广州     |
|  2 | 荆轲         |      1 | 广州     |
|  4 | 貂蝉         |      1 |          |
|  5 | 后羿         |      1 | 珠海     |
|  6 | 芈月         |      1 | 广州     |
|  7 | 嬴政         |      1 | 广州     |
|  8 | 狄仁杰       |      1 | 深圳     |
|  9 | 鲁班七号     |      1 | 深圳     |
| 10 | 孙尚香       |      1 | 深圳     |
+----+--------------+--------+----------+
9 rows in set (0.00 sec)
```

程咬金已经被删除

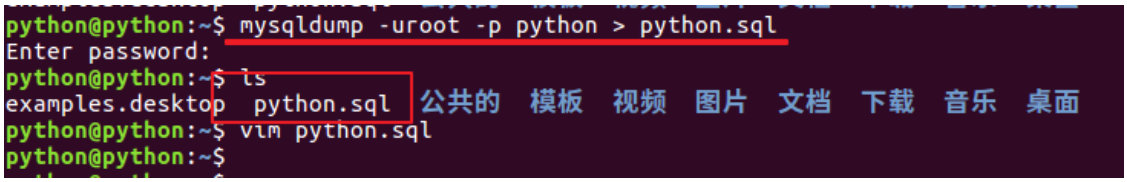
四、备份和恢复数据库

1、备份数据库

备份数据库的所有表的数据

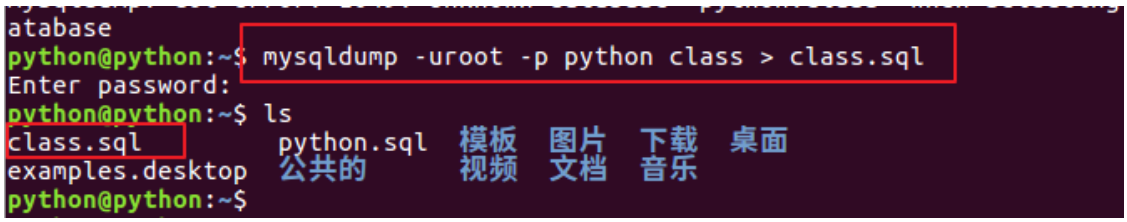
```
mysqldump -uroot -p 数据库名 > python.sql;
mysqldump -uroot -p python >python.sql
```

提示输入密码，mysql 的密码



备份数据库的某个数据表的数据

```
mysqldump -uroot -p 数据库名  数据表名> class.sql;
msyqldump -uroot -p python class > class.sql
```



2、恢复数据库

恢复数据库之前要先创建库，我们来看看备份出来的文件，其实里面全是 sql 语句，有创建表的语句，但是没有创建库的，所以要先手动创建库，

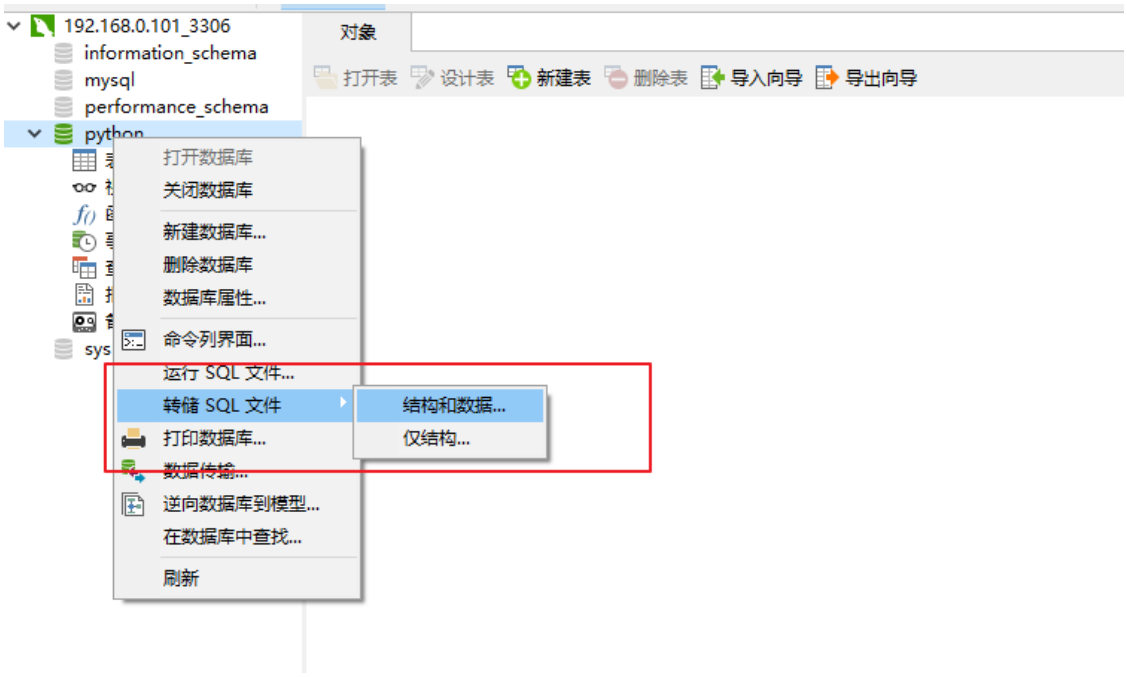
```
-- MySQL dump 10.13  Distrib 5.7.20, for Linux (x86_64)
--
-- Host: localhost    Database: python
-- ------------------------------------------------------
-- Server version       5.7.20-0ubuntu0.16.04.1

/*!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`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `class` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(10) NOT NULL,                    创建表
  `delete` bit(1) NOT NULL DEFAULT b'0',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

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

```
mysql -uroot -p 新数据库名 < python.sql
```

会提示输入数据库的密码

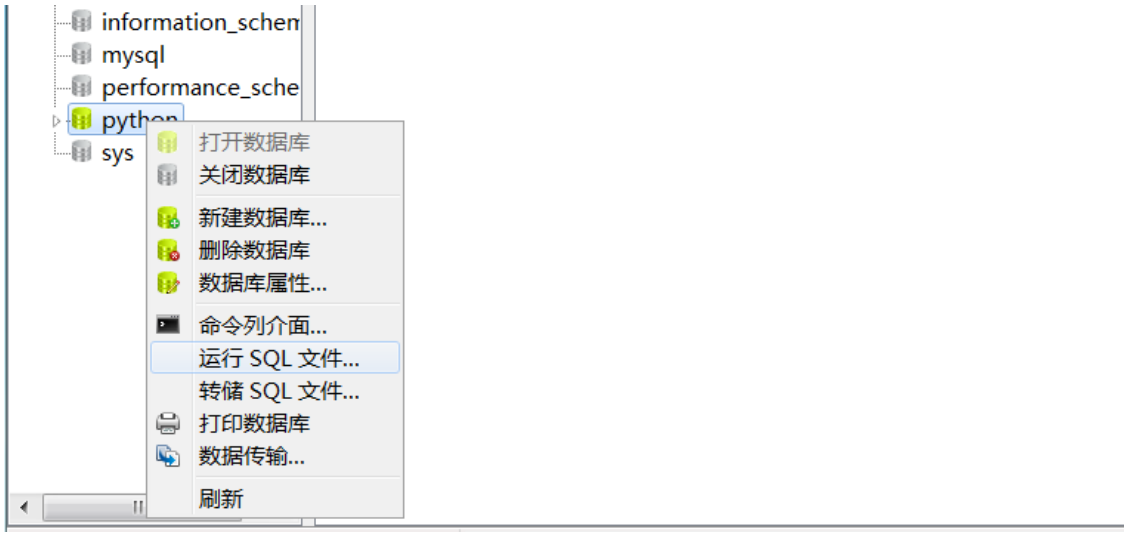
3、图形界面备份和恢复数据库

备份数据库，转储 SQL 文件即可以进行数据库备份



恢复数据库

新建数据库，运行备份好的 SQL 文件即可



小结

数据库基础

- 数据库基础概述
- MySQL 数据库介绍
- MySQL 数据库安装

数据库和数据表管理

- 数据库管理
- 数据表管理
- 图形化界面操作数据库

简单查询与数据操作

- 基本查询语句
- 插入数据
- 修改数据
- 删除数据

备份和恢复数据库

- 备份数据库
- 恢复数据库
- 图形界面备份和恢复数据库

课后作业

课后问答题

1、请写出全列插入与部分插入的语法格式。

课后实操题

1、使用 navicat 图形界面工具，创建一个名为 test 的数据库，编码格式为 utf-8，在 test 库中创建一个表 goods 商品表。字段需要 id,name,price,is_delete.在表中随便插入一些数据

2、利用 navicat 将第一题中的数据库备份。

3、命令行的方式将刚才建立的数据库删除。

4、命令行的方式重复第一题的操作，创建库，表，并插入一些数据。

5、在 goods 表中多插入一些数据，演练课上的修改删除，插入等 sql 语句。

6、用命令的方式备份备份 test 数据库。