**MySQL数据库技术**

# Http协议：

1. **什么是HTTP协议？**
2. **HTTP协议包括哪几种？每种协议的组成有哪些？**
3. **常见的状态码有哪些？**
4. **常见的响应头有哪些？**

指服务器和客户端间进行通信时的约束和规范，客户端与服务端的数据交互并不是杂乱无章的，需要遵照（基于）一定的规范进行。

## 协议：

协议，就是事先的一种约定、规则、规范、标准。（入学协议）

## 常见协议

1、HTTP、HTTPS 超文本传输协议

2、FTP 文件传输协议

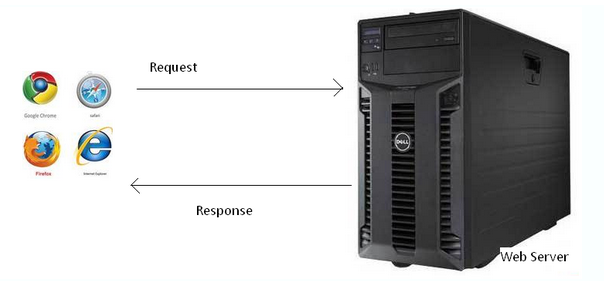
3、SMTP 简单邮件传输协议

## HTTP协议

即超文本传输协议，网站是基于HTTP协议的，例如网站的图片、CSS、JS等都是基于HTTP协议进行传输的。

HTTP协议是由从客户机到服务器的请求(Request)和从服务器到客户机的响应(Response)进行了约束和规范。

即HTTP协议主要由请求和响应构成。



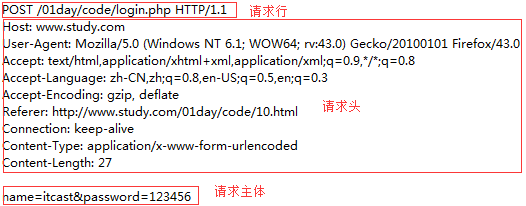
常用请求方法 POST、GET、PUT、DELETE

请求 (request)：请求行、请求头、请求主体。

响应 (response)：状态行、响应头、响应主体。

### 请求/请求报文

请求由客户端发起，其规范格式为：请求行、请求头、请求主体。



1、请求行



由请求方式、请求URL和协议版本构成

2、请求头

* **host**：当前url中所要请求的服务器的主机名（域名）。
* accept：表示浏览器可以接收的数据类型(MIME类型)
* 如text/html,text/css,text/javascript,image/png,image/jpeg….
* accept-language：可以接收的语言类型，zh-cn，en，fr等，
* user-agent：用户代理UA，当前发起请求的浏览器的内核信息，识别客户端设备的重要信息。
* Referer：从哪个URL跳转过来的
* Accept-Encoding：可接受的压缩格式

3、请求主体

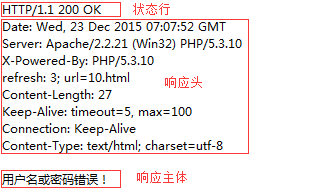
即传递给服务端的数据

**注：当以post形式提交表单的时候，请求头里会设置**

**Content-Type: application/x-www-form-urlencoded，以get形式当不需要**

### 响应/响应报文

响应由服务器发出，其规范格式为：状态行、响应头、响应主体。



1、状态行



由协议版本号、状态码和状态信息构成

2、响应头

Date：响应时间

Server：服务器信息

Content-Length：响应主体长度

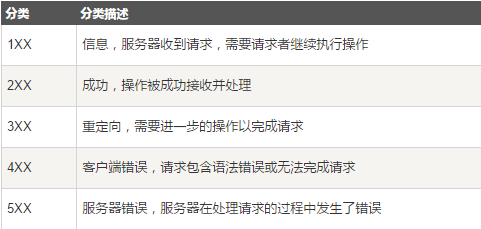
Content-Type：响应资源的MIME类型

MIME是标识文件类型的，文件后缀并不能正确无误的标识文件的类型。

3、响应主体

即服务端返回给客户端的内容；

状态码



常见的有：

200代表成功、

302页面重定向、

304 文档未修改、

404未找到

### php设置响应头

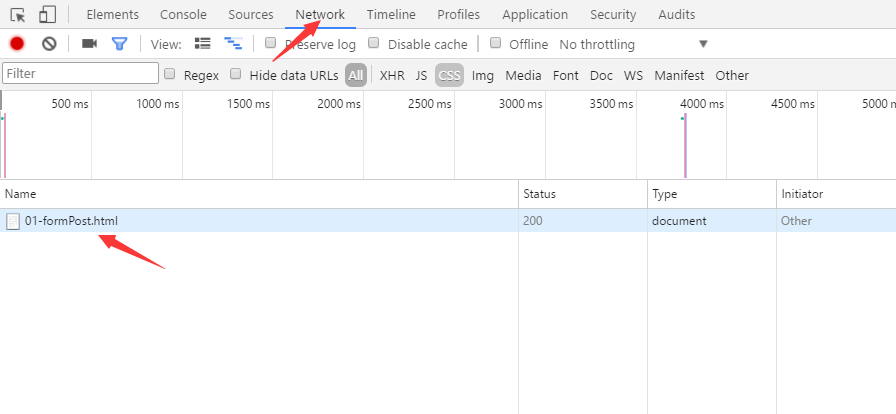
header('location:01-formPost.html'); //立即跳转

header('refresh:3;url=01-formPost.html'); //在指定时间后跳转

header('content-type:text/html;charset=utf-8'); //设置编码

### 调试工具

利用HTTP抓包工具在开发中可以帮我们进行调试，常用抓包工具HttpWatch、Fiddler、Charles、FireBug等



# 数据库简介

程序核心就是对数据的操作！

Var life=90; left+=10; left-=20; hu+=200;

**一个问题? 网站页面展示的大量数据存储在哪里？**

## 数据库概念

专门用来存储、管理数据的仓库（空间），按照数据结构来组织、存储和管理，可以实现高效存储及读取的数据解决方案。英文：Database,DB。 Database Administrator DBA

实际项目中，一般将项目相关的数据全部存储在单个数据库中。

数据库有专门的、高效的数据管理系统（DBMS），可以应对数据量较大、关系较为复杂的情况。

## 关系型数据库

当前使用范围最广的数据库。按照关系模型（数据之间存在联系）组织的数据库。数据存储在硬盘文件中。

订单：订单编号，商品名称，价格，数量，配送地址

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **订单编号** | **商品名称** | **价格** | **数量** | **配送地址** |
| 100123 | 华为mate10 | 4999 | 2 | 北京市京顺路99号 |
| 100124 | 花裙子 | 499 | 1 | 北京市清华大学1号楼 |

关系模型是采用二维表格（行、列）及其之间的联系。

相对于其他模型，更贴合现实世界，更便于理解。

### 数据库的组织架构：

* 数据表（table）

表是以行和列的形式组织起来的数据的集合。项目一般需要不同的数据表，将数据分布存储在不同的数据表中。

如：订单表，用户表，商品表，配送地址表，

* 记录（row，record）
  + 一行有意义的信息的集合
* 字段(column，field)
  + 某一列的值的名称
* 通过行列交叉确定具体数据

### 关系型数据库产品—MySQL介绍

瑞典AB公司开发，现为Oracle公司所有。2010年04月22 发布了MySQL 5.5。

数据库官网地址：<https://dev.mysql.com>

英文手册地址：<https://dev.mysql.com/doc/refman/5.5/en/>

中文参考论坛：<http://www.mysqlpub.com/>

## SQL（了解）

SQL：structured Query Language结构化查询语言。  
 SQL语句，相当于客户端发送的命令（与数据库服务器进行交互）。

### 概念

* 关系型数据库所使用通用语言。为一个ISO标准。 w3c
* 数据库脚本语言的后缀。

### 分类

数据操作基本逻辑：增删改查操作

# MySQL数据库服务器访问

WAMP ：M---MySQL

## 访问步骤

* 开启服务器

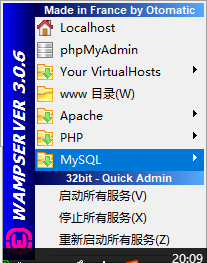
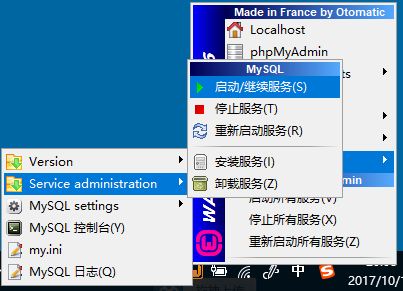
成功访问数据服务器的前提是：数据库服务器在运行状态。

* 通过客户端发送命令(第一步为登录，建立起连接)

MySQL有权限验证（第二步），需要验证账号信息（用户名及密码）及用户的数据操作权限（是否具备查、删除等权限）。

## 开启数据库服务器

打开 wamp菜单

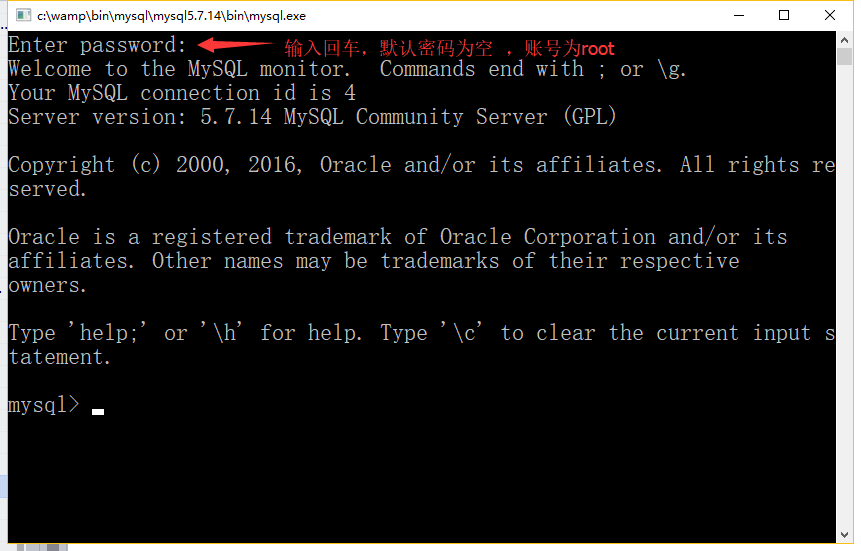
## 客户端连接数据库服务器

软件架构是C/S。

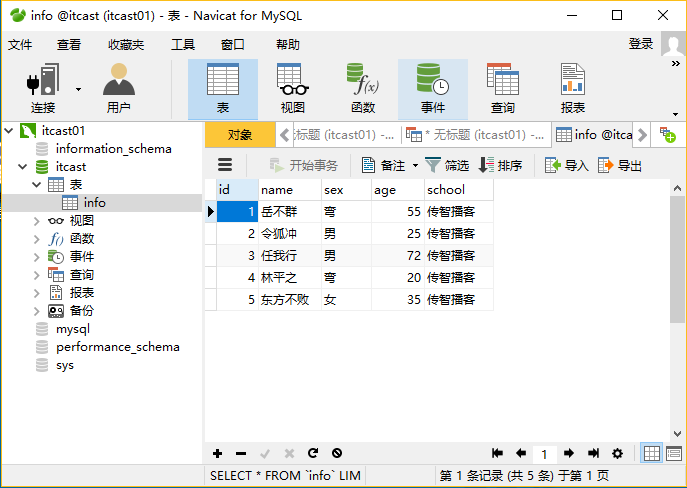
### 常见客户端

* 命令行客户端 ：打开Mysql 控制台





* 图形化界面：Navicat for mysql , phpmyadmin



* **PHP**：

可以直接操作数据库。通过PHP发送SQL命令，对数据操作。

此时，PHP作为MySQL的一个客户端。

### 命令行连接命令

登录完成，建立起客户端与数据库服务器之间的连接。然后可以发送SQL命令。

mysql[.exe] [-h主机名] [–p端口号] –u用户名 –p密码

* mysql.exe数据库服务器登录验证主程序，命令行访问通常需要设置环境变量。
* 服务器主程序是：mysqld.exe
* h，host,主机名。主机地址 (域名或者IP地址)。数据库默认主机名为 localhost，可以省略。当前只有localhost本地主机。----127.0.0.1
* p（第一个p）,port，端口号。服务器监视该端口，提供数据库服务。数据库默认端口号 3306。可以省略。
* u，user，用户名。当前数据库服务器只有root超级管理员账号，即用户名为root
* p（第二个p），password，密码。可以直接输入，也可以在回车之后输入，保护账号信息安全。
* -h，-p,-u,-p和对应的参数之间可以使用空格隔开，也可以之间连接。
* 尚未登录，不是SQL语句，命令末尾没有分号
* 如果提示：cann’t connect to mysql server.原因是：服务器未启动

### 设置字符集

只针对命令行客户端来说。避免后续获取中文字符信息或者发送包含中文的命令时出现乱码。

set names gbk;

//通知数据库服务器在交互过程中使用的字符集为GBK，而不是UTF8

//数据库服务器中数据一般存储为UTF8，但是返回时数据库服务器可以转换为GBK编码

登录成功之后第一件事：set names gbk;

### 退出

exit 或者 \q 或者 quit

# 数据库操作

需要先成功连接数据库服务器。数据库服务器中一般存储多个数据库，每个数据库一般对应单个项目。

## 查看数据库

### 查看数据库服务器中的全部数据库

**show** databases;

注意事项：

* SQL语句：使用英文状态的分号，作为语句结束标志。
* 系统数据库（information\_schema,performance\_schema,mysql）不能删除。

### 查看数据库的创建语句

show create database 数据库名;

显示信息包括：数据库的创建语句及数据库选项（包括字符集）。

默认字符集：数据库数据存储默认使用UTF8(对中文进行编码)该字符集。

## 创建数据库

创建数据库，保存该项目数据。

### 创建语法

create database 数据库名 [charset 字符集];

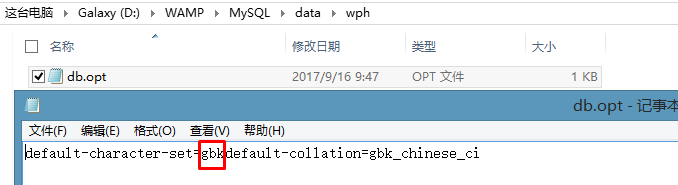
默认字符集为utf-8

### 注意事项

每创建一个数据库，系统在数据的存储路径mysql/data文件夹中，创建数据库名字对应的文件夹，其中包含db.opt文件，存储数据库选项信息。

* 数据库选项包括字符集及校对规则。不指定时，使用默认的字符集及校对规则。

常见字符集：ASCII，GBK，UTF8



### 数据库名的命名规范

* 字母、数字、下划线组成。
* 不建议使用纯数字的名称。
* 命名见名知意。数据表表名或者数据库的库名使用下划线命名法。可以添加项目前缀。
* 长度不超过32个字符，便于使用。

### 调试技巧

* 已经存在的数据库，无法再次创建。
* 查找错误提示：从near开始

## 删除数据库

drop database 数据库名;

删除数据库（删除结构，相当于中介拆房子），全部数据丢失。

提示：不能删除系统数据库（information\_schema, mysql, performance\_schema）,

* 删除特殊名字的数据库，需要添加反引号``
* 删除数据库：数据库中的所有数据表被删除(存储的数据丢失)，需要先备份。

## 修改数据库

alter database 数据库名 charset 新字符集;

数据库名称不能修改。修改数据库的主要操作为修改字符集(修改数据源存储的字符集)。

## 选择数据库

数据真正存储的地方：数据表。

一般先选择数据库(数据表所在的数据库)，然后才能对其中的数据表进行操作。

use dbName;

# 数据表操作 真正存储数据的地方：数据表，table。

项目开发一大重要的技能：将项目的数据,使用合理的数据表存储在数据库中。

## 查看数据表

### 查看数据库中的全部数据表

show tables;

## 字段类型初步介绍

### 整型

存储如年龄，产品数量，编号等。

int（0-21亿） 占据4个字节的空间

### 小数类型

float ,decimal eg.重量，工资，奖金，价格等

使用decimal类型，实现小数的精确存储，原理为可以存储每位数字。一般用来存储与钱有关的数字。

### 字符串型

varchar(M)，char(M) M为该字段可以存储的最多字符数，如varchar(10)。M参数不可以省略。

varchar一般用来存储长度变化比较大的字符串，如文章标题，商品名称，char存储长度比较固定的字符串，如手机号，身份证号，序列号，邮编。

此外可以使用text类型，存储较长的字符串，无需指定字符串的具体长度。

### 日期时间型

datetime,date(年月日)，time（时分秒）。年月日时分秒，也可以使用PHP获取时间戳整型数字存储。

## 创建数据表

### 语法

create table 数据表名(

字段1 类型 [属性],

字段2 类型 [属性],

…

字段n 类型 [属性]

);

创建数据表的核心是定义数据存储的**结构**：**字段**名称 **类型** **属性**

### 需求：创建学生就业信息表，存储：学号,姓名，班级，就业薪资，毕业时间

### 注意事项

* 数据表中有不同的字段。创建数据表之前需要先分析数据表字段的个数，类型及属性。字段定义离不开数据表的定义，同时，数据表定义离不开字段定义。
* 字段的定义包括：字段名称，字段类型，字段属性等
* 不同的字段定义之间，使用英文的逗号隔开
* 最后一个字段定义与括号之间不能有逗号

## 列属性介绍

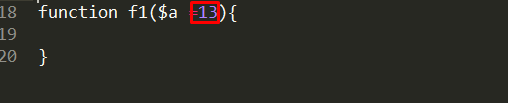
字段数据的属性（特征）

### not null

限制字段值不能为空

### default

设置字段的默认值，在没有录入时自动使用默认值填充。



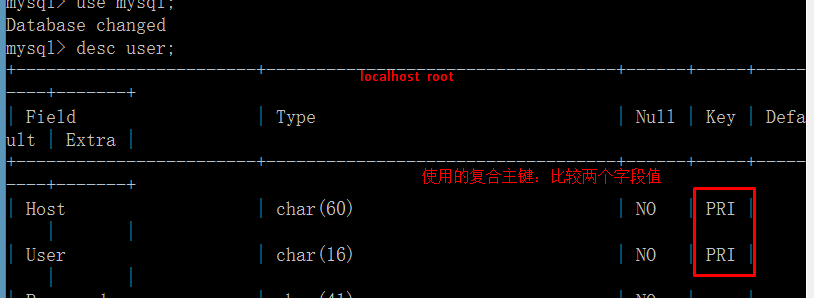
default 默认值

### primary key

设置字段为主键，主键字段的值不能重复，不能为空。而且一个数据表中只能设置一个字段为主键，作为每行记录的唯一身份信息（索引）。

stu\_id int primary key

了解复合主键：



### auto\_increment

设置字段为自动增长，默认从1开始自动分配编号。自增长字段必须为一个key（索引，数据结构，便于快速查找数据，相当于书的目录），一般与primary key结合使用。

stu\_id int primary key auto\_increment

//设置stu\_id字段为自增长字段

类型必须为**整型**。

### unique key

唯一键，设置字段的值为唯一的，可以设置多个字段为唯一键。唯一键字段的值可以为空。

学号设置为主键，要求唯一的，不能为空的，用来标识学生信息，

姓名，不能重名，设置唯一键，该字段的值唯一的 、不重复的、可以为空的

### 需求：创建学生就业信息表，存储：学号,姓名，班级、年龄，就业薪资

## 删除数据表

drop table 数据表表名；

删除数据表的结构定义，会导致数据表结构文件丢失，同时数据表中的全部数据丢失。删除之前，做好数据的备份工作。

# 数据操作初步

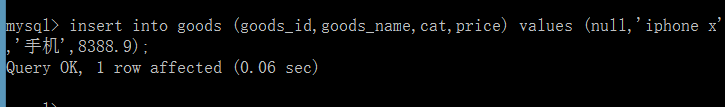
基本操作：CRUD，增（C,create,insert）,删(D,delete),改(U,update)，查（Retrieve）。

## 插入数据

insert into 数据表 （字段1，字段2）values (值1，值2)；

### 注意事项：

* 插入数据时，字段列表(插入数据的字段顺序)与数据列表要一一对应。



* 插入数据的类型，需要与字段设置的类型一致。数据应该不超出字段的表示范围。
* 字符串类型必须使用单引号或者双引号包括。日期时间型也需要使用单双引号。
* 字段列表中，字段出现的顺序不需要与数据表中字段的顺序一致。
* 插入全部字段信息时，可以省略字段列表。此时，必须按照数据表中字段出现的顺序录入。

## 查询数据

select 字段1，字段2 | \* from 数据表名 [where 条件]；

* \*代表查看全部字段的信息，返回数据表的全部字段的信息：
* 所得即所需。可以指定查询的具体信息（字段列表），实现仅查询需要的数据，减少数据的请求量。
* 从数据库语句执行效率角度，尽量不查询全部数据。执行效率高（返回的数据更小，传输占据的带宽小）
* 可以添加where限定条件，返回满足条件的行。
* where执行过程:类似于while循环，逐一扫描磁盘上数据表中的所有行，满足限定条件，该行在结果集中显示。

## 修改数据

update 数据表 set 字段1 =值1,字段2 =值2 where 条件；

* 不添加where条件，修改数据表的全部行对应字段的信息。
* 更新语句，通常需要添加where限定条件。

## 删除数据

删除数据表中的指定行，不对表结构进行操作。

delete from 数据表[where 条件]；

### drop 与delete区别：

**drop**:SQL的结构定义语句（DDL），整个数据表消失。

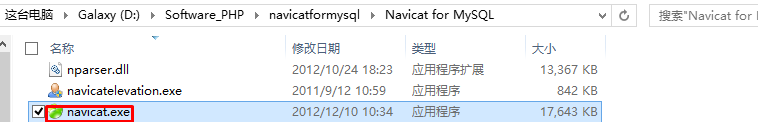
**delete**:SQL的数据操作语言（DML）,一般删除指定行。

### where条件

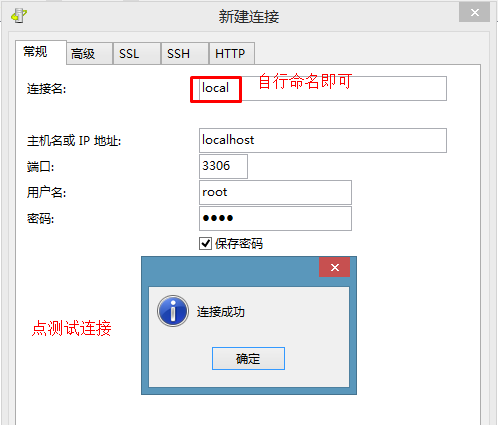
* 不添加where条件，删除数据表中的所有行。数据表的结构依然存在。
* 添加where条件，删除满足条件的所有行。

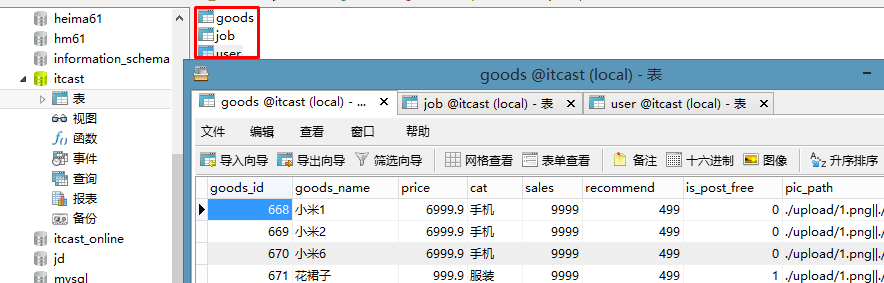
# navicat for mysql的使用

navicat for mysql是一个专业的MySQL数据库图形化界面工具。绿色软件。

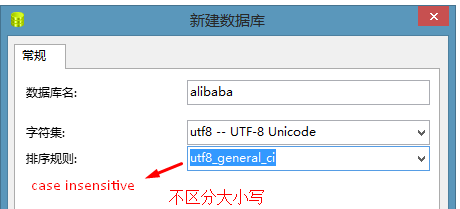


## 建立连接



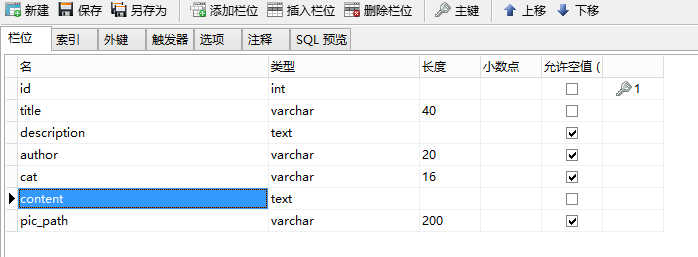


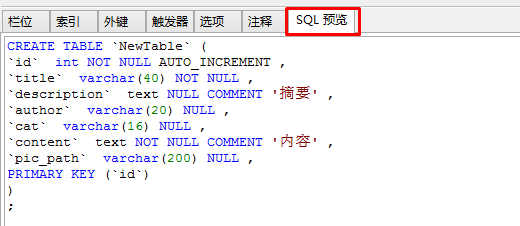
## 创建数据库



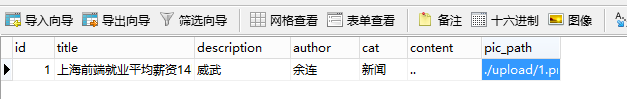
## 创建数据表

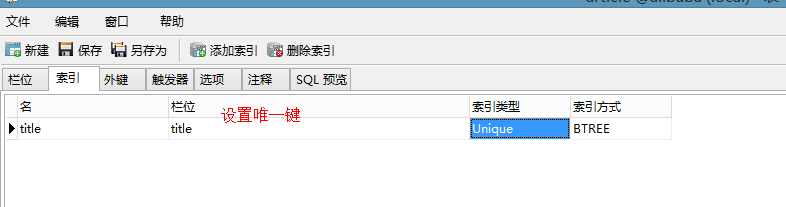
新建表—新建---设置字段名、类型及长度---保存—设置表名





## 插入数据





## 删除数据表

# 查询高级操作

## where子句

查询时，不添加where条件返回数据表所有行。需要添加限定条件，只返回需要的行。

select 字段列表 from table where子句；

Like 模糊匹配 % 通配符

## Count() 获取返回数据的总条数

## limit 子句

返回查找结果中的前n行



select 字段列表 from table limit 偏移量，每页显示的行数

对于一个页面，显示的行一般为固定的。

偏移量：偏移指定行，开始显示。

第n页，偏移前n-1页已经显示的内容。

偏移量 =（页数 -1） \* 每页显示的行数

## 注意事项

where子句、limit子句可以同时使用:

select 字段列表 from table [where 子句] [limit 子句]

注意各个子句之间的顺序。

# 连接查询与子查询

## 连接查询

select 字段列表 from A join B on A.字段 = B.字段 [where子句];

查询学生基本信息以及班级信息：

## 子查询

子查询就是一个查询的结果作为另外一个查询的条件使用。外层的查询称为父查询，作为条件的查询称为子查询（subquery）。

select 字段列表 from table where … （子查询）；

**查询前端专业学生信息：**

select 字段列表 from tb where 字段 in (subquery);

//子查询返回多个值

//父查询字段取其中一个则满足条件