# 第9章 数据库、Mysql、表

# 数据库概述

## 什么是数据库

数据库就是存储数据的仓库，其本质是一个文件系统，数据按照特定的格式将数据存储起来，用户可以对数据库中的数据进行增加，修改，删除及查询操作。

## 数据库特点

1).数据结构化

数据库系统实现了整体数据的结构化，这是数据库的最主要的特征之一。这里所说的“整体”结构化，是指在数据库中的数据不再仅针对某个应用，而是面向全组织；不仅数据内部是结构化，而且整体式结构化，数据之间有联系。

2).数据的共享性高，冗余度低，易扩充

因为数据是面向整体的，所以数据可以被多个用户、多个应用程序共享使用，可以大大减少数据冗余，节约存储空间，避免数据之间的不相容性与不一致性。

3).数据独立性高

数据独立性包括数据的物理独立性和逻辑独立性。

物理独立性是指数据在磁盘上的数据库中如何存储是由DBMS管理的，用户程序不需要了解，应用程序要处理的只是数据的逻辑结构，这样一来当数据的物理存储结构改变时，用户的程序不用改变。

逻辑独立性是指用户的应用程序与数据库的逻辑结构是相互独立的，也就是说，数据的逻辑结构改变了，用户程序也可以不改变。

数据与程序的独立，把数据的定义从程序中分离出去，加上存取数据的由DBMS负责提供，从而简化了应用程序的编制，大大减少了应用程序的维护和修改

4).数据由DBMS统一管理和控制

数据库的共享是并发的（concurrency）共享，即多个用户可以同时存取数据库中的数据，甚至可以同时存取数据库中的同一个数据。

DBMS必须提供以下几方面的数据控制功能：

数据的安全性保护（security）

数据的完整性检查（integrity）

数据库的并发访问控制（concurrency）

数据库的故障恢复（recovery）

## 为什么要引入数据库?

数据保存到内存：

优点：读写非常快

缺点：程序关闭导致数据丢失

数据保存到文件：

优点：数据可以永久保存

缺点：1）频繁地IO操作，效率不高！

2）数据管理不方便。例如查询某个数据需要全部读取出来，再匹配。

数据保存到数据库软件：

优点：1）数据永久保存下来

2）数据管理非常方便。（例如查询非常快速和方便）

## 什么是数据库管理系统

数据库管理系统（DataBase Management System，DBMS）：指一种操作和管理数据库的大型软件，用于建立、使用和维护数据库，对数据库进行统一管理和控制，以保证数据库的安全性和完整性。用户通过数据库管理系统访问数据库中表内的数据。

## 常见的数据库管理系统

关系型数据库：

MYSQL ：开源免费的数据库，小型的数据库.已经被Oracle收购了.MySQL6.x版本也开始收费。

Oracle ：收费的大型数据库，Oracle公司的产品。Oracle收购SUN公司，收购MYSQL。

DB2 ：IBM公司的数据库产品,收费的。常应用在银行系统中.

SQLServer：MicroSoft 公司收费的中型的数据库。C#、.net等语言常使用。

SyBase ：已经淡出历史舞台。提供了一个非常专业数据建模的工具PowerDesigner。

SQLite : 嵌入式的小型数据库，应用在手机端。

非关系型数据库：

NoSQL，泛指非关系型的数据库。

NoSQL是对不同于传统的关系数据库的数据库管理系统的统称。

两者存在许多显著的不同点，其中最重要的是NoSQL不使用SQL作为查询语言。其数据存储可以不需要固定的表格模式，也经常会避免使用SQL的JOIN操作，一般有水平可扩展性的特征。

MongoDb、redis、HBase

## SQL语句分类

数据定义语言：简称DDL(Data Definition Language)，用来定义数据库对象：数据库，表，列等。关键字：create，alter，drop等

数据操作语言：简称DML(Data Manipulation Language)，用来对数据库中表的记录进行更新。关键字：insert，delete，update等

数据控制语言：简称DCL(Data Control Language)，用来定义数据库的访问权限和安全级别，及创建用户。

数据查询语言：简称DQL(Data Query Language)，用来查询数据库中表的记录。关键字：select，from，where等

## 数据库客户端、数据库管理系统、数据库、表、字段之间关系

数据库中以表为组织单位存储数据。

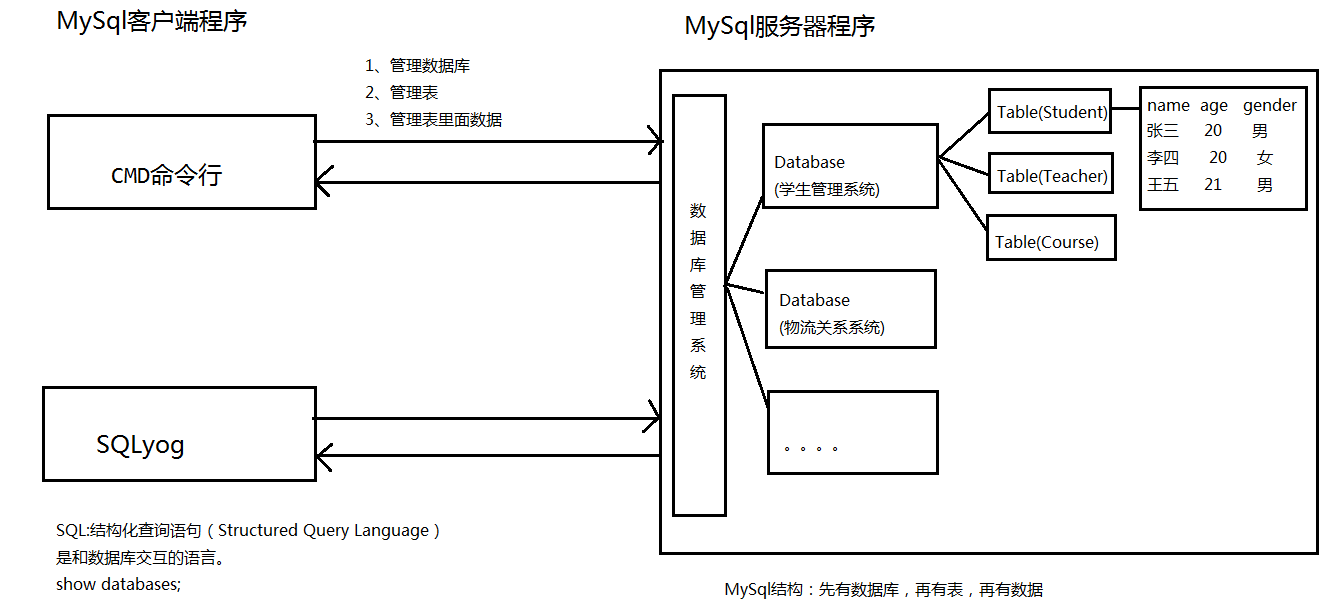
表类似我们的Java类，每个字段都有对应的数据类型。

那么用我们熟悉的java程序来与关系型数据对比，就会发现以下对应关系。

类----------表

类中属性----------表中字段

对象----------记录



# MySql安装与卸载

安装参考：mysql安装图解教程。

MySql卸载:

1、控制面板-管理工具-服务，停掉MySql服务。

2、控制面板-程序和功能，卸载MySql

3、到D:\Program Files安装目录把MySql目录删除

4、C:\ProgramData删除MySql目录

5、控制面板-管理工具-服务里面如果还有MySql服务，需要到注册表中把服务删除,

如果没有MySql服务这一步不需要执行。

到regedit把注册表

HKEY\_LOCAL\_MACHINE/SYSTEM/ControlSet001/Services/Eventlog/Applications/MySQL

HKEY\_LOCAL\_MACHINE/SYSTEM/ControlSet002/Services/Eventlog/Applications/MySQL

HKEY\_LOCAL\_MACHINE/SYSTEM/CurrentControlSet/Services/Eventlog/Applications/MySQL

删除之后需要重启。

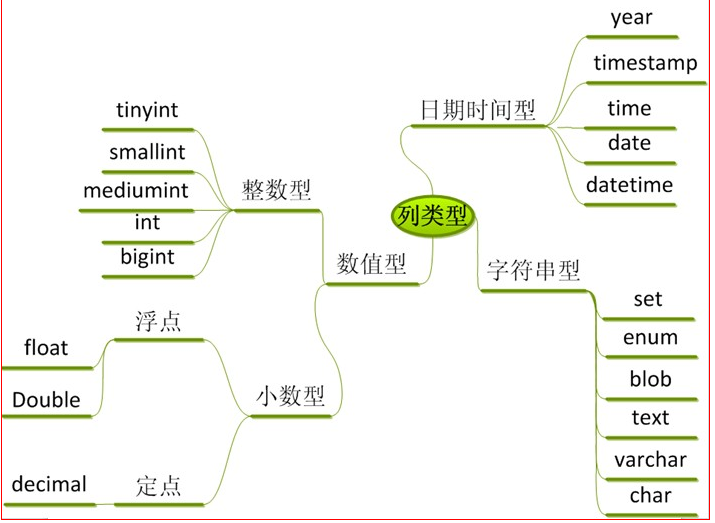
# MySql数据类型

SQL语句可以单行或多行书写，以分号结尾

可使用空格和缩进来增强语句的可读性

MySQL数据库的SQL语句不区分大小写，建议使用大写，例如：SELECT \* FROM user。

同样可以使用/\*\*/的方式完成注释



## 整数类型

tinyint(1字节), smallint(2字节), mediumint(3字节),int(4字节), bigint(8字节)

1字节=8位（8个灯泡）

一个灯泡只能表达2个意思（2个数字）

2个灯泡可以表达4个意思

3个灯泡可以表达8个意思

。。。。。

8个灯泡（1个字节）可以表达256个数字。

默认情况下，这些整数类型都是可正可负的，那么：

tinyInt就只能存储: -128--127这些数



## 小数类型

单精度浮点型：float非精确数，通常不设定长度

双精度浮点：double非精确数，通常不设定长度

定点型：decimal精确数，通常定点型需要设定长度，形式为：decimal(总长, 小数位数）

## 时间日期类型

Date,time,datetime,year,timestamp

注意：

写入数据库时，直接的时间日期数据，应该用单引号引起了。

year类型可以是4位整数或4位纯数字字符串，也可以是2位整数或2位纯数字字符串

timestamp表示的含义是“时间戳”，其实就是指“当前时刻”，本质上是一个数字，代表从1970年1月1日0点0分0秒到某个时间之间的秒数数值。该类型的字段值无需赋值，而是会自动取得当前时间值。

## 字符串类型

### varchar、char

最基本最重要的2个：

varchar类型：可变长度字符串类型。最多能存储65532个字节的字符串——也还要考虑字符编码。设定的长度只是最长长度，但可以不存满，则实际长度以数据长度为准。

char类型：定长字符串类型。最多能存储256个字符。如果存储的数据不足设定的长度，则会自动补空格填满。

设定时都需要给定长度，比如：varchar(20),char(6);

mysql，一行的所有内容的总的存储长度也有个限制，约65535个。

### 二进制文本binary、varbinary

binary：类似char，只是里面不存“文本”，而是存“文本的二进制数据”

varbinary: 类似varchar，同样，不存“文本”，而是存“文本的二进制数据”

### 大文本类型text、Blob

text：可以存储“超大文本”，且其实际的长度并不占用一行的长度。相对char和varchar效率低。

Blob:可以存储“超大二进制文本”，通常用于存储图片这种“二进制数据”。

### 选项文本enum、set

enum：

专门用于方便存储类似表单中的“单选项”的值。

形式：

enum(‘选项1’，‘选项2’，‘选项3’，......）

这些选项的值虽然是字符串，但其数据库内部存储其实是数字（效率高），他们的数字值是：1， 2， 3，4， 5，。。。。。最多6万多个。

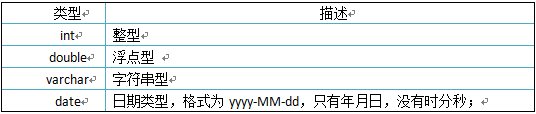
set：专门用于方便存储类似表单中的“多选项”的值。

形式：

set (‘选项1’，‘选项2’，‘选项3’，......）

这些选项的值虽然是字符串，但其数据库内部存储其实是数字（效率高），他们对应的数字值是：1， 2，4， 8， 16，。。。。。最多6万多个。

MySQL中的我们常使用的数据类型如下



# 据库操作

## 创建数据库

create database 数据库名;

create database 数据库名 character set 字符集;

CREATE DATABASE java1711 DEFAULT CHARACTER SET utf8;

## 查看数据库

查看数据库MySQL服务器中的所有的数据库:

show databases;

查看某个数据库的定义的信息:

show create database 数据库名;

## 删除数据库

drop database 数据库名称;

## 其他的数据库操作命令

切换数据库：

use 数据库名;

# 表相关操作

## 创建表

基本形式

create table [if not exists] 表名(字段列表， [约束或索引列表]) [表选项列表];

说明：列表都是表示“多个”，相互之间用逗号分开。

字段基本形式： 字段名 类型 [字段修饰属性]；

CREATE TABLE student(

id INT PRIMARY KEY AUTO\_INCREMENT,

NAME VARCHAR(10),

age INT,

gender VARCHAR(1)

);

## 字段属性设置

not null： 不为空，表示该字段不能放“null”这个值。不写，则默认是可以为空

auto\_increment: 设定int类型字段的值可以“自增长”，即其值无需“写入”，而会自动获得并增加

此属性必须随同 primary key 或 unique key 一起使用。

[primary] key： 设定为主键。是唯一键“加强”：也不能重复并且不能使用null，并且可以作为确定任意一行数据的“关键值”，最常见的类似：where id= 8; 或 where user\_name = ‘zhangsan’;

通常，每个表都应该有个主键，而且大多数表，喜欢使用一个id并自增长类型作为主键。

但：一个表只能设定一个主键。

unique [key] : 设定为唯一键：表示该字段的所有行的值不可以重复（唯一性）。

default ‘默认值’： 设定一个字段在没有插入数据的时候自动使用的值。

comment ‘字段注释’：

CREATE TABLE student(

id INT PRIMARY KEY AUTO\_INCREMENT not null,

NAME VARCHAR(10) comment ‘注释’,

age INT,

gender VARCHAR(1)

);

## 索引设置

什么是索引：

索引是一个“内置表”，该表的数据是对某个真实表的某个（些）字段的数据做了“排序”之后的存储形式。

其作用是：极大提高表查找数据的速度！——其效率（速度）可以匹敌二分查找。

注意：索引在提供查找速度的同时，降低增删改的速度。

对创建（设计）表来说，建立索引是非常简单的事，形式如下：

索引类型 (字段名1，字段名2， .... ） //可以使用多个字段建立索引，但通常是一个

有以下几种索引：

普通索引：key(字段名1，字段名2， .... ）：它只具有索引的基本功能——提速(其他索引不但有提速还有额外功能)

唯一索引：unique key (字段名1，字段名2， .... ）

主键索引：primary key (字段名1，字段名2， .... ）

全文索引：fulltext (字段名1，字段名2， .... ）

CREATE TABLE student(

id INT PRIMARY KEY AUTO\_INCREMENT not null,

NAME VARCHAR(10) comment ‘注释’,

age INT,

gender VARCHAR(1),

unique key(name),

key(age)

);

## 约束设置

什么叫约束：约束就是一种限定数据以符合某种要求的形式（机制）

约束主要有：

### 主键约束

primary key (字段名1，字段名2， .... ）

其实就是主键索引，也是主键属性。即primary key有3个角度的理解（说法）：字段属性设置为主键，或建立的主键索引，或设定一个主键约束，但他们的本质是一样

### 唯一约束

unique key (字段名1，字段名2， .... ），其实也是“3体合一”（类似primary key)

### 外键约束

什么叫外键：就是设定一个表中的某个字段的值，必须“来源于”另一个表的某个主键字段的值。

语法形式：

foreign key (字段名1，字段名2， .... ) references 表名2(字段名1，字段名2， .... )

说明：

对某个（些）字段设定外键，则其相对应的其他表的对应字段需要设置为主键。

CREATE TABLE student(

id INT PRIMARY KEY AUTO\_INCREMENT not null,

NAME VARCHAR(10) comment ‘注释’,

age INT,

gender VARCHAR(1),

class\_id INT,

unique key(name),

key(age),

foreign key (class\_id) references Class(id)

);

### 非空约束

就是要求该字段的值不能为空，其只能在字段上当作字段属性来设定。

### 默认约束

就是要求该字段的值在“空”的时候会自动填充该设定的默认值，也只能字段上设定。

### 检查约束

就是使用一个表达式（逻辑判断）来决定数据是否有效，比如年龄字段，可以使用

tinyint，则可能会超过127就不合适了。

tinynit unsigned，则0-255是可以的。

但：如果考虑现实情况，假设某保险公司只作150岁以下的人的保险。则我们就可以继续对该字段可能存储的数据进行“约束”。比如类似：if（age > 150){return false}

可惜的是：ｍｙｓｑｌ不支持检查约束的语法和功能。

## 表选项

表选项就是对一个表的有关属性的设定，通常都不需要。如果不设定，则有其默认值。

有以下几个可用：

comment = ‘表的注释’;

charset = 字符编码名称； //跟数据的字符编码设定一个意思。

字符编码设定的范围及继承关系：

系统级设定：安装时确定了。

库级设定：建库时设定；

表级设定：就是这里的charset = 字符编码名称

字段级设定：作为字段属性出现。

他们都只对“字符类型”的字段有效。每一级如果没有设定，就会“继承使用”其上一级的设定。

auto\_increment = 起始整数； //自增长类型值的初值，默认是1

engine = “表的存储引擎名”； //

存储引擎就是将数据存入硬盘（或其他媒介）的方式方法。通常就几个可用，默认是InnoDB

存储引擎决定一个数据表的各方面的信息：功能和性能。

## 修改表，删除表，表的其他操作

通常，创建一个表，能搞定（做到）的事情，修改表也能做到。

增删改字段：

增：alter table 表名 add [column] 字段名 字段类型 字段属性；

删： alter table 表名 drop 字段名

改：alter table 表名 change 原字段名 新字段名 新字段类型 新字段属性；

删除表：

drop table [if exists] 表名；

## 表的其他操作

显示所有表show tables：

显示表结构desc 表名；

显示表的创建语句: show create table 表名；。

从已有表复制表结构：create table [if not exists] 新表名 like 原表名;

从已有表复制表结构：create table [if not exists] 新表名 select \* from 原表名 where 1<>1;（不推荐）

# MySQL配置文件-my.ini

my.ini是什么？

my.ini是MySQL数据库中使用的配置文件，修改这个文件可以达到更新配置的目的。

my.ini存放在哪里？

my.ini存放在MySql安装的根目录

参数简介：

1.port参数也是表示数据库的端口。

2.basedir参数表示MySQL的安装路径。

3.datadir参数表示MySQL数据文件的存储位置，也是数据库表的存放位置。

4.default-character-set参数表示默认的字符集，这个字符集是服务器端的。

5.default-storage-engine参数默认的存储引擎。

6.sql-mode参数表示SQL模式的参数，通过这个参数可以设置检验SQL语句的严格程度。

7.max\_connections参数表示允许同时访问MySQL服务器的最大连接数，其中一个连接是保留的，留给管理员专用的。

8.query\_cache\_size参数表示查询时的缓存大小，缓存中可以存储以前通过select语句查询过的信息，再次查询时就可以直接从缓存中拿出信息。

9.table\_cache参数表示所有进程打开表的总数。

10.tmp\_table\_size参数表示内存中临时表的总数。

11.thread\_cache\_size参数表示保留客户端线程的缓存。

12.myisam\_max\_sort\_file\_size参数表示MySQL重建索引时所允许的最大临时文件的大小。

13.myisam\_sort\_buffer\_size参数表示重建索引时的缓存大小。

14.key\_buffer\_size参数表示关键词的缓存大小。

15.read\_buffer\_size参数表示MyISAM表全表扫描的缓存大小。

16.read\_rnd\_buffer\_size参数表示将排序好的数据存入该缓存中。

17.sort\_buffer\_size参数表示用于排序的缓存大小。

# 思途心得

在JavaEE开发体系中数据库部分占有非常重要的作用，我们做的企业级的系统大部分都是要和数据打交道，如何管理数据就显得非常关键，说的简单一点后台系统开发大部分的操作都是对数据的增删改查操作，只不过是中间要涉及要具体的业务逻辑，所以掌握好数据库的开发就显得异常重要，而且数据库的开发是脱离具体的编程语言的，虽然我们这里讲解的是java，但是不同的语言同样可以操作这些数据库，所以数据库的知识使非常通用的知识。

在这一章中大家要重点掌握数据库的数据类型、表单创建和表中字段的约束。

# 作业

如下是电商项目数据库设计：

用户表user：

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 用户表id |
| username | varchar(50) | 用户名 |
| password | varchar(50) | 用户密码，MD5加密 |
| email | varchar(50) | 邮箱 |
| phone | varchar(20) | 电话 |
| question | varchar(100) | 找回密码问题 |
| answer | varchar(100) | 找回密码答案 |
| role | int(4) | 角色0-管理员,1-普通用户 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

收货地址表shipping:

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 收获地址id |
| user\_id | int(11) | 用户id |
| receiver\_name | varchar(20) | 收货姓名 |
| receiver\_phone | varchar(20) | 收货固定电话 |
| receiver\_mobile | varchar(20) | 收货移动电话 |
| receiver\_province | varchar(20) | 省份 |
| receiver\_city | varchar(20) | 城市 |
| receiver\_district | varchar(20) | 区/县 |
| receiver\_address | varchar(200) | 详细地址 |
| receiver\_zip | varchar(6) | 邮编 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

产品表product:

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 商品id |
| category\_id | int(11) | 分类id,对应category表的主键 |
| name | varchar(100) | 商品名称 |
| subtitle | varchar(200) | 商品副标题 |
| main\_image | varchar(500) | 产品主图,url相对地址 |
| sub\_images | text | 图片地址,json格式,扩展用 |
| detail | text | 商品详情 |
| price | decimal(20,2) | 价格,单位-元保留两位小数 |
| stock | int(11) | 库存数量 |
| status | int(6) | 商品状态.1-在售 2-下架 3-删除 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

商品分类表category:

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 类别Id |
| parent\_id | int(11) | 父类别id当id=0时说明是根节点,一级类别 |
| name | varchar(50) | 类别名称 |
| status | tinyint(1) | 类别状态1-正常,2-已废弃 |
| sort\_order | int(4) | 排序编号,同类展示顺序,数值相等则自然排序 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

购物车cart:

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 购物车id |
| user\_id | int(11) | 哪个用户的购物车，对应的用户的id |
| product\_id | int(11) | 商品id |
| quantity | int(11) | 数量 |
| checked | int(11) | 是否选择,1=已勾选,0=未勾选 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

订单表order:

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 订单id |
| order\_no | bigint(20) | 订单号 |
| user\_id | int(11) | 用户id |
| shipping\_id | int(11) | 订单对应的收货地址表的id |
| payment | decimal(20,2) | 实际付款金额,单位是元,保留两位小数 |
| payment\_type | int(4) | 支付类型,1-在线支付 |
| postage | int(10) | 运费,单位是元 |
| status | int(10) | 订单状态:0-已取消-10-未付款，20-已付款，40-已发货，50-交易成功，60-交易关闭 |
| payment\_time | datetime | 支付时间 |
| send\_time | datetime | 发货时间 |
| end\_time | datetime | 交易完成时间 |
| close\_time | datetime | 交易关闭时间 |
| create\_time | datetime | 创建时间 |

订单项表order\_item:

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 订单子表id |
| user\_id | int(11) | 订单项对应的用户表的id |
| order\_no | bigint(20) | 订单项对应的订单表的id |
| product\_id | int(11) | 商品id |
| product\_name | varchar(100) | 商品名称 |
| product\_image | varchar(500) | 商品图片地址 |
| current\_unit\_price | decimal(20,2) | 生成订单时的商品单价，单位是元,保留两位小数 |
| quantity | int(10) | 商品数量 |
| total\_price | decimal(20,2) | 商品总价,单位是元,保留两位小数 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

支付信息表pay\_info：

|  |  |  |
| --- | --- | --- |
| Field | Type | Comment |
| id | int(11) | 支付信息表id |
| user\_id | int(11) | 用户id |
| order\_no | bigint(20) | 订单号 |
| pay\_platform | int(10) | 支付平台:1-支付宝,2-微信 |
| platform\_number | varchar(200) | 支付宝支付流水号 |
| platform\_status | varchar(20) | 支付宝支付状态 |
| create\_time | datetime | 创建时间 |
| update\_time | datetime | 最后一次更新时间 |

根据上面表的表述创建对应的表，并画出对应的ER关系图。