# **[MySQL数据类型——数值类型](https://www.cnblogs.com/wangyanzi/p/6115354.html)**

### **1.1.1 **整型****

|  |  |  |  |
| --- | --- | --- | --- |
| 整型 | 占用字节 | 范围 | 范围 |
| tinyint | 1 | -27~27-1 | -128~127 |
| smallint | 2 | -215~215-1 | -32768~32767 |
| mediumint | 3 | -223~223-1 | -8388608~8388607 |
| int | 4 | -231~231-1 | -2147483648~2147483647 |
| bigint | 8 | -263~263-1 |  |

**多学一招：一个字节=8位**

**帮助的位置：列类型——数值类型**

### **1.1.2 **显示宽度****

显示宽度：最小的显示位数，比如int(11)，最少用11位数字显示值。例如存放了“12”,则显示“00000000012”。

IMG_256

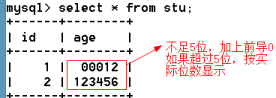
**脚下留心：显示宽度必须结合zerofill才起作用。**

IMG_257

插入测试数据

IMG_258

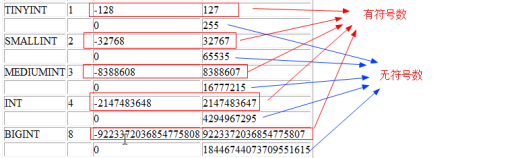
查询结果



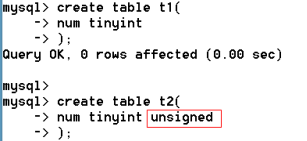
**脚下留心：显示宽度不决定显示范围，只是在数值不够指定位数的时候用0做前导。**

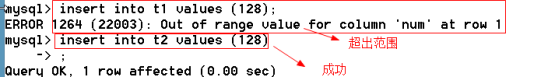
### **1.1.3 **无符号数（unsigned）****

无符号数就是没有负数，无符号数的正数的范围是有符号数正数范围的2倍。



测试：创建两个表



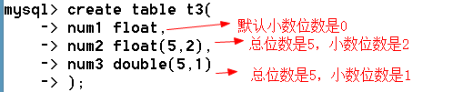


### **1.1.4 **浮点数****

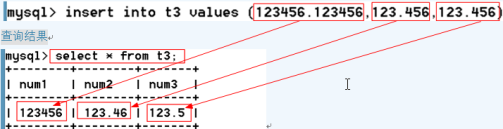
|  |  |  |
| --- | --- | --- |
| 浮点数 | 占用字节 | 范围 |
| float(单精度) | 4 | -3.4E+38~3.4E+38 |
| double(双精度) | 8 | -1.8E+308~1.8E+308 |

浮点数的声明：float(M,D)，double(M,D)

M：总位数 D：小数位数

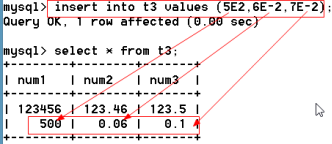


插入测试数据



**脚下留心：没有指定小数位数，默认小数位数是0位**

**多学一招：MySQL浮点数支持科学计数法**



**5E2=5\*1026E-2=6\*10-2**

**脚下留心：浮点数的精度会丢失**

mysql> create table t4(

    -> num1 float

    -> );

Query OK, 0 rows affected (0.01 sec)

mysql> insert into t4 values (99.999999999);

Query OK, 1 row affected (0.00 sec)

mysql> select \* from t4;

+------+

| num1 |

+------+

|  100 |

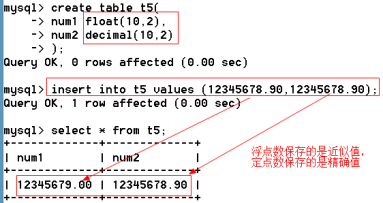
+------+

1 row in set (0.00 sec)

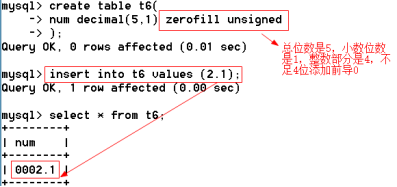
### **1.1.1 **定点数（小数）****

语法：decimal(M,D)

M的最大值是65，D的最大值是30，默认是(10,0)



**多学一招：定点数和浮点数都支持显示宽度、无符号数**



**脚下留心：decimal是变长的，大致每9个数字用4个字节存储，整数和小数分开存储。定点数占用的资源可能会比浮点数大很多。**

## **1.1 **MySQL数据类型——字符型****

|  |  |
| --- | --- |
| 数据类型 | 描述 |
| char(L) | 定长字符 |
| varchar(L) | 可变长度字符 |
| tinytext | 大段文本（大块数据）   28-1=255个字符 |
| text | 大段文本（大块数据）   216-1=65535个字符 |
| mediumtext | 224-1 |
| longtext | 232-1 |

### **1.1.1 **char****

1、 char(L)：MySQL不回收多余的空间

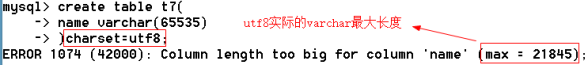
2、 L的最大长度是255

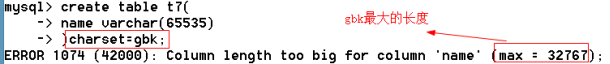
IMG_268

### **1.1.2 **varchar****

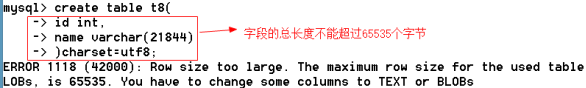
1、 varchar(L):MySQL回收多余的空间

2、 L的理论最大长度是65535，但事实上达不到，因为有的字符是多字节字符，比如一个utf8的字符占用3个字节，65535/3大约保存2万多字符；如果是gbk,一个字符占两个字节，65535/2大约保存3万多个字符。



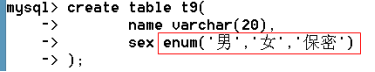


**脚下留心：一个记录的所有字段（不包含大数据）的总长度不能超过65535个字节**

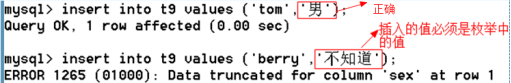


## **1.2 **MySQL数据类型——枚举（enum）****

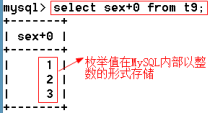
从集合中选择一个值作为数据（单选）



插入测试数据



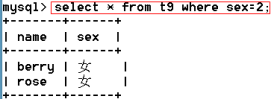
MySQL管理枚举值是通过整型的数字来管理的，第一个值是1，第二个值是2，以此类推。



既然枚举值以整数的形式存储，插入值的时候就可以直接插入整数

IMG_275

也可以通过整型数字来做查询



### **1.2.1 **枚举优点****

1、 限制值

2、 节省空间

3、 运行速度快（整数比字符串运行速度快）

**思考题：已知枚举占用2个字节，请问最多可以有多个枚举值。**

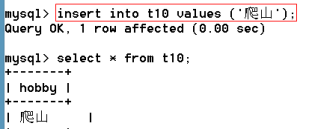
**答：2个字节16位，可以保存216方个值（65536，0-65535），因为枚举值从1开始，所以最多可以有65535个枚举值。**

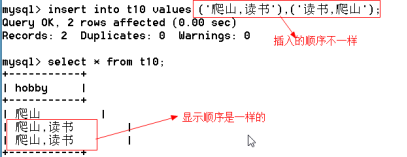
## **1.3 **MySQL数据类型——集合（set）****

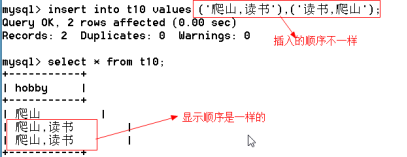
从集合中选择一些值作为数据（多选）

IMG_277

插入数据



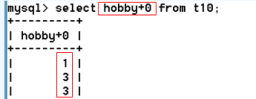




IMG_281

**多学一招：集合和枚举一样，在MySQL内部也是通过数字来管理的。MySQL为每个集合元素分配一个固定的值。分配方式从前往后依次是20,21,22，…。**

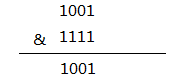
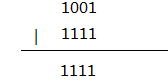
**如果有多个选项，值是单个选项的和**



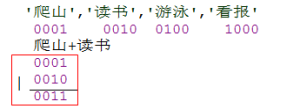
**多学一招：按位或和按位与**

**按位与：所有的位都是1结果才是1**

**按位或：只要有一位是1结果就是1**

### **1.3.1 **集合是一个按位或的关系****



**思考题：一直集合占用8个字节，可以表示多少个选项？**

**答：8个字节是64个位，可以表示64个选项**

## **1.4 **MySQL数据类型——日期时间型****

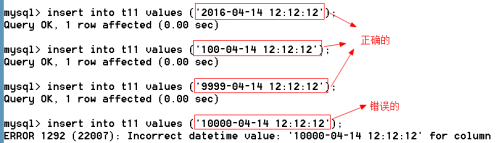
|  |  |
| --- | --- |
| 数据类型 | 描述 |
| datetime | 日期时间   占8个字节 |
| date | 日期        占四个字节 |
| time | 时间 |
| year | 年份，占用1个字节 |
| timestamp | 时间戳，占用4个字节 |

### **1.4.1 **datetime  (日期时间)****

格式：年-月-日  小时:分钟:秒

IMG_286

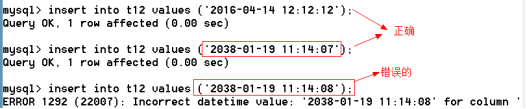
--插入数据



### **1.4.2 **timestamp  （时间戳）****

IMG_288

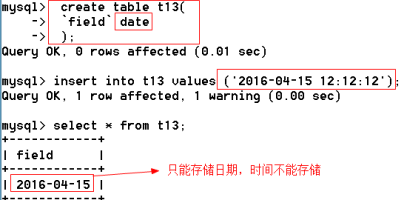
插入数据



**脚下留心：datetime和timestamp在表现上是一样的，他们的区别在于：**

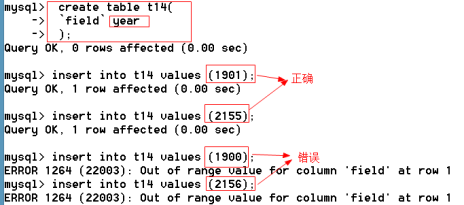
**Datetime从1到9999，而timestamp从1970年到2038年（原因是timestamp只占用了4个字节），2038年01月19日11:14:07秒后就超过了4个字节的长度。**

### **1.4.3 **date  只保存日期，不保存时间****



### **1.4.4 **year****

因为year占用1个字节，所以只能保存255个年份，范围是1901~2155



### **1.4.5 **time****

可以表示时间，还可以表示时间间隔，范围是-838:59:59~838:59:59

IMG_292

插入数据

IMG_293

IMG_294

IMG_295

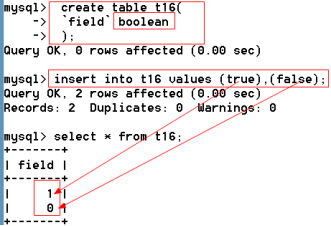
IMG_296

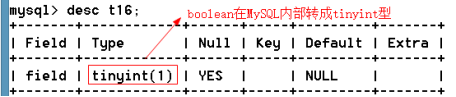
**多学一招：time也支持以天的方式表示时间间隔**

IMG_297

## **1.5 **MySQL数据类型——boolean****

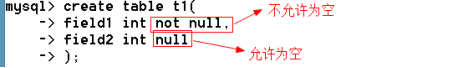
MySQL不支持boolean型，true和false在数据库中对应的是1和0



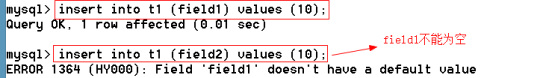


## **1.6 **列属性——是否为空（null|not null）****

用来约束字段的值是否为null



插入测试数据



## **1.7 **列属性——默认值 （default）****

如果一个字段没有插入值，自动插入一个默认值。

IMG_302

IMG_303

## **1.8 **列属性——自动增长（auto\_increment）****

字段的值从1开始，每次递增1。特点字段中不会有重复的值，合适为记录生成一个唯一的id，主键常用。

IMG_304

插入测试数据

IMG_305

**脚下留心：**

1、 **在MySQL中，auto\_increment列必须做主键**

2、 **auto\_increment是从1开始增长的，所以定义的时候一般和unsigned一起使用（使用无符号整数）**

3、 **自动增长列一般用null插入**

4、 **自动增长列上的数据被删除，默认不再重复使用。**

5、 **delete from 清空数据后编号继续增长，truncate table 清空表后编号从1开始增长.因为truncate table 是删除原来的表创建一个新表。**

**思考题：一个自动增长列最后一个记录的编号是10，再插入一条编号一定是11？**

**答：不对，因为10后面的记录可能被删除过。**

**思考题：在一个自动增长列中，插入3条，删除2条，插入3条，删除2条，插入3条，删除2条，再插入编号是多少？**

**答：10，删除和不删除对结果不产生影响。**

## **1.9 **列属性——主键（primary key）****

1、 主键：唯一标识表中记录的一个或一组列。

2、 主键特点：不能重复，不能为空

3、 一个表只能有一个主键，但是主键可以由多个字段组成。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 学校编号 | 班级号 | 学号 | 姓名 | 成绩 |
| 1 | 1 | 1 | tom | 99 |
| 2 | 1 | 1 | tom | 88 |

### **1.9.1 **主键的作用****

1、 保证数据完整性

2、 加快查询速度

### **1.9.2 **添加主键方法一：****

mysql> create table stu(

    -> id int primary key,

    -> name varchar(20)

    -> );

Query OK, 0 rows affected (0.00 sec)

### **1.9.3 **添加主键方法二：****

mysql> create table stu(

    -> id int,

    -> name varchar(20),

    -> primary key(id)

    -> );

**多学一招：如果主键由多个字段组成**

mysql> create table stu(

    -> id int,

    -> name varchar(20),

    -> primary key(id,name)

    -> );

### **1.9.4 **添加主键方法三****

通过更改表的语法添加主键

语法：alter table 表名 add primary key(字段)

mysql> create table stu(

    -> id int,

    -> name varchar(20)

    -> );

Query OK, 0 rows affected (0.00 sec)

mysql> alter table stu add primary key (id);

Query OK, 0 rows affected (0.06 sec)

Records: 0  Duplicates: 0  Warnings: 0

mysql> desc stu;

+-------+-------------+------+-----+---------+-------+

| Field | Type        | Null | Key | Default | Extra |

+-------+-------------+------+-----+---------+-------+

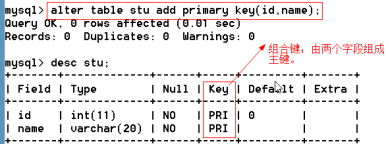
| id    | int(11)     | NO   | PRI | 0       |       |

| name  | varchar(20) | YES  |     | NULL    |       |

+-------+-------------+------+-----+---------+-------+

2 rows in set (0.06 sec)

**添加多个列做主键**



### **1.9.5 **删除主键****

语法：alter table 表名 drop primary key;

IMG_307

### **1.9.6 **选择主键的原则****

1、 最少性：尽量选择单个键做主键

2、 稳定性：尽量选择更新少的列做主键。

3、 能用数字做主键的就不要用字符串

## **1.10 **列属性——唯一键（unique）****

1、 不能重复可以为空

2、 一个表可以有多个唯一键

### **1.10.1 **创建唯一键方法一：****

mysql> create table stu(

    -> id int,

    -> name varchar(20) unique,

    -> stuadd varchar(20) unique

    -> );

Query OK, 0 rows affected (0.00 sec)

mysql> desc stu;

+--------+-------------+------+-----+---------+-------+

| Field  | Type        | Null | Key | Default | Extra |

+--------+-------------+------+-----+---------+-------+

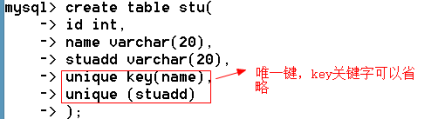
| id     | int(11)     | YES  |     | NULL    |       |

| name   | varchar(20) | YES  | UNI | NULL    |       |

| stuadd | varchar(20) | YES  | UNI | NULL    |       |

+--------+-------------+------+-----+---------+-------+

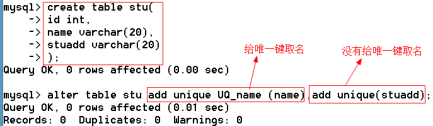
### **1.10.2 **创建唯一键方法二：****



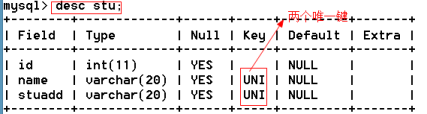
### **1.10.3 **创建唯一键的方法三****

更改表的时候创建唯一键

语法：alter table 表名 add unique [key] [唯一键名](字段名)

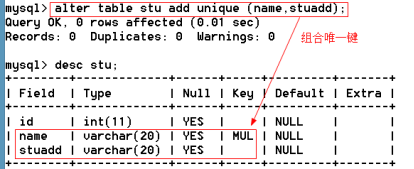


可以理解为添加一个字段。

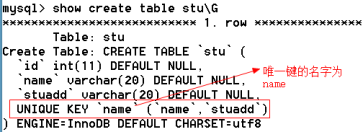


如果不给唯一键显式取名，MySQL会自动给键取一个名字。

### **1.10.4 **创建组合唯一键****



### **1.10.5 **查看唯一键的名字****



### **1.10.6 **删除唯一键****

通过唯一键的名字来删除

语法：alter table 表名 drop index 唯一键名

IMG_313

## **1.11 **列属性——备注（comment）****

为了程序员之间相互交流，可以给字段添加备注。

mysql> create table stu(

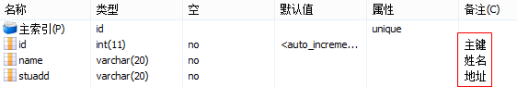
    -> id int auto\_increment comment '主键',

    -> name varchar(20) not null comment '姓名',

    -> stuadd varchar(20) not null comment '地址',

    -> primary key(id)

    -> );

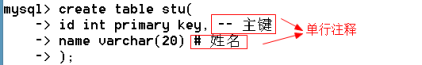


IMG_315

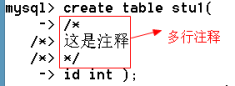
## **1.12 **SQL注释****

单行注释：--或 #

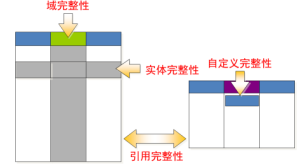
多行注释：/\*    \*/



多行注释



## **1.13 **数据完整性****



如果记录之间无法区分称为失去了实体完整性

如果字段的值不正确称为失去了域完整性

两个表的引用出现错误称为失去了引用完整性

### **1.13.1 **保证实体完整性****

1、 主键约束

2、 唯一约束

3、 自动增长列（标识列）

### **1.13.2 **保证域完整性****

1、 非空约束

2、 默认值约束

3、 数据类型约束



### **1.13.3 **引用完整性****

外键约束

### **1.13.4 **保证自定义完整性****

1、 存储过程

2、 函数

3、 触发器

## **1.14 **外键（foreign key）****

1、 外键：从表中的公共字段称为外键。外键约束用来保证引用完整性。

2、 公共字段名字可以不一样，但是属性必须一样

### **1.14.1 **创建外键方法一：创建表的时候就创建外键****

create table stuinfo(

stuno char(4) primary key,

name varchar(20) not null

);

create table stumarks(

id char(4) primary key,

score tinyint unsigned not null default 0,

foreign key (id) references stuinfo(stuno)

);

### **1.14.2 **创建外键的方法二：修改表的时候创建外键****

语法：alter table 从表 add [constraint 外键名] foreign key(公共字段) references 主表(公共字段)

constraint：约束     references：参照

create table stuinfo(

stuno char(4) primary key,

name varchar(20) not null

);

create table stumarks(

id char(4) primary key,

score tinyint unsigned not null default 0

);

--添加外键

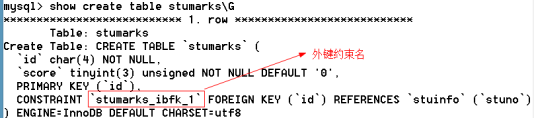
alter table stumarks add foreign key(id) references stuinfo(stuno);

--添加外键的时候给外键取名

alter table stumarks add constraint `FK1` foreign key(id) references stuinfo(stuno);

**提示：如果在某个字段上添加外键，该字段必须具有索引才行，如果该字段上有索引就用已经存在索引；如果没有MySQL会自动创建索引。**

### **1.14.3 **查看外键名****



### **1.14.4 **删除外键****

语法：alter table 表名 drop foreign key 外键约束名

IMG_321

### **1.14.5 **外键约束中的主表和从表****

1、 主表中没有的从表中不能插入

2、 从表中有的主表中不能删除

3、 先删从表，再删主表

## **1.15 **外键操作****

1、 严格限制：默认是严格显示，知识点参见主表和从表

2、 置空操作(set null)：如果主表记录删除，从表对应的记录外键字段设为null。

3、 级联操作(cascade)：如果主表更新，从表也更新

语法：foreign key(外键字段) references 主表(关联字段) [主表记录删除时动作] [主表记录更新时的动作]

一般来说：主表删除，从表置空，主表修改，从表级联

--主表

create table stuinfo(

stuno char(4) primary key,

name varchar(20) not null

);

--从表

create table stumarks(

id int unsigned auto\_increment primary key,

stuno char(4),

score tinyint unsigned not null default 0,

foreign key(stuno) references stuinfo(stuno) on delete set null on update cascade   --删除时置空，更新时级联

);

--测试

mysql> insert into stuinfo values ('1001','tom');

Query OK, 1 row affected (0.00 sec)

mysql> insert into stumarks values (null,'1001',88);

Query OK, 1 row affected (0.00 sec)

mysql> select \* from stuinfo;

+-------+------+

| stuno | name |

+-------+------+

| 1001  | tom  |

+-------+------+

1 row in set (0.00 sec)

mysql> select \* from stumarks;

+----+-------+-------+

| id | stuno | score |

+----+-------+-------+

|  1 | 1001  |    88 |

+----+-------+-------+

1 row in set (0.00 sec)

mysql> update stuinfo set stuno='1002' where stuno='1001';

Query OK, 1 row affected (0.01 sec)

Rows matched: 1  Changed: 1  Warnings: 0

mysql> select \* from stuinfo;

+-------+------+

| stuno | name |

+-------+------+

| 1002  | tom  |

+-------+------+

1 row in set (0.00 sec)

mysql> select \* from stumarks;

+----+-------+-------+

| id | stuno | score |

+----+-------+-------+

|  1 | 1002  |    88 |

+----+-------+-------+

1 row in set (0.00 sec)

mysql> delete from stuinfo where stuno='1002';

Query OK, 1 row affected (0.00 sec)

mysql> select \* from stumarks;

+----+-------+-------+

| id | stuno | score |

+----+-------+-------+

|  1 | NULL  |    88 |

+----+-------+-------+

1 row in set (0.00 sec)

**脚下留心：只用Innodb引擎才支持外接约束。**

## **1.16 **思考题****

q 手机号码一般使用什么数据类型存储? Char

q QQ号 varchar

q 性别一般使用什么数据类型存储? char

q 年龄信息一般使用什么数据类型存储? tinyint

q 照片信息一般使用什么数据类型存储? binary

q 薪水一般使用什么数据类型存储? Decimal

q 学员姓名允许为空吗? 不允许

q 家庭地址允许为空吗? 不允许

q 电子邮件信息允许为空吗? 允许

q 考试成绩允许为空吗? 允许

q 在主键列输入的数值，允许为空吗? 不允许

q 一个表可以有多个主键吗? 不可以

q 在一个学校数据库中，如果一个学校内允许重名的学员，但是一个班级内不允许学员重名，可以组合班级和姓名两个字段一起来作为主键吗？ 可以

q 标识列允许为字符数据类型吗？ 不允许

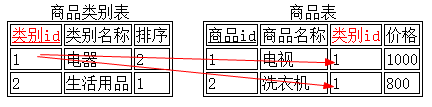
q 表中没有合适的列作为主键怎么办？ 添加一个自动增长列做主键

q 如果标识列A的初始值为1，增量为1，则输入三行数据以后，再删除两行，下次再输入数据行的时候，标识值从多少开始？ 从4开始

**一个字段是用字符型还是数字型，就看字段的值是否有计算的可能。如果可能会计算则用数字，否则用字符型。**

## **1.17 **实体之间的关系（表与表之间的关系）****

### **1.17.1 **一对多：(1:N)主表中的一条记录对应从表中多条记录****



一个商品类别对应多个商品

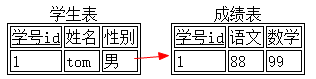
一个班级对应多个学生

一个客户对应多个订单

**脚下留心：主键和非主键建关系实现一对多**

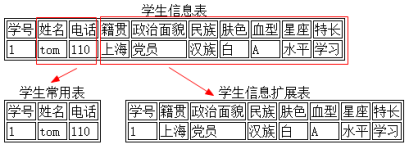
### **1.17.2 **多对一 (N:1)（和一对多是一样的）****

### **1.17.3 **一对一 （1:1）****



**脚下留心：主键和主键建关系实现一对一**

应用：如果一记录字段很多，所有字段放在一个表中会影响效率，可以将多个字段分成常用字段和不常用字段两个表。



### **1.17.4 **多对多(N:M)****

要实现多对多必须要有关系表

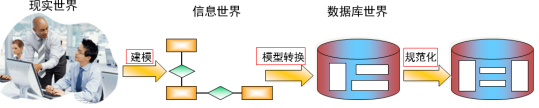
班级和讲师的关系



科目和学生的关系



## **1.1 **数据库设计步骤****



### **1.17.5 **项目需求****

BBS论坛的基本功能：

1、用户注册和登录，后台数据库需要存放用户的注册信息和在线状态信息；

2、用户发贴，后台数据库需要存放贴子相关信息，如贴子内容、标题等；

3、用户可以对发帖进行回复（跟帖）；

4、论坛版块管理：后台数据库需要存放各个版块信息，如版主、版块名称、贴子数等；

### **1.17.6 **标识实体（Entiry）****

在项目需求中用到的实体标识出来。实体是名词。

1、 用户

2、 主帖

3、 跟帖

4、 板块

### **1.17.7 **标识每个实体的属性（Attribute）****

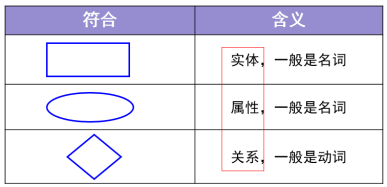


### **1.17.8 **标识实体之间的关系（Relationship）****

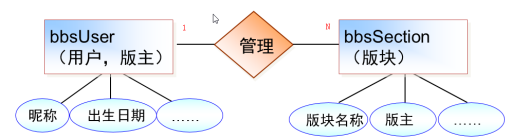
标识两个表的关系（参照实体之间的关系）

### **1.17.9 **绘制E-R图（实体关系图）****

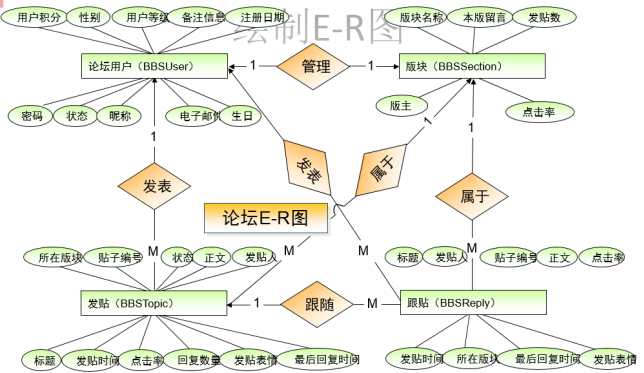
语法：



例题：



### **1.17.10 **E-R图实现如下****



### **1.17.11 **将E-R转成表****

1、 一个实体就是一个表

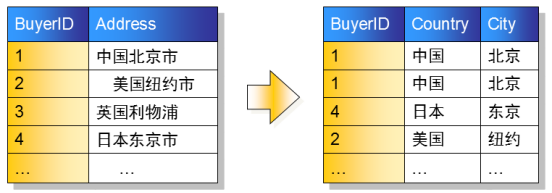
2、 一个属性就是一个字段

3、 建立关系

**脚下留心：如果在转换的过程中，没有一个属性可以住主键，就添加一个自动增长列做主键**

## **1.18 **数据规范化****

第一范式：确保每列原子性（每个字段不可再分）

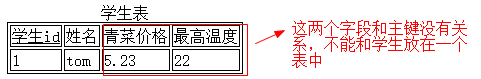


**脚下留心：如果要通过地址做统计或查询，就把地址分开，如果仅仅是一个地址字符串，没有统计或查询的需求，地址可以不拆分。**

**思考：如下表结构是否合理？**



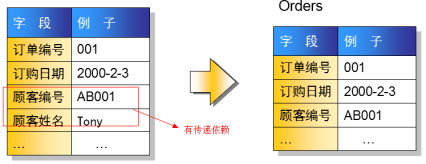
第二范式：非键字段必须依赖于键字段。（一个表只描述的一件事情）





第三范式：消除传递依赖

第三范式约束的是非主键字段部分。



**脚下留心：必须满足低范式以后才能满足高范式，**

**思考题：如何实现高考分数表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 考试号 | 姓名 | 语文 | 数学 | 总分 |

满足第一范式

满足第二范式

不满足第三范式    语文+数学=总分，但实际中确实添加了“总分”字段，就是为了提高性能。

**当规范化和性能发生冲突的时候，我们要性能**

### **1.18.1 **例题****

需求

1、公司承担多个工程项目，每一项工程有：工程号、工程名称、施工人员等

2、公司有多名职工，每一名职工有：职工号、姓名、性别、职务（工程师、技术员）等

3、公司按照工时和小时工资率支付工资，小时工资率由职工的职务决定（例如，技术员的小时工资率与工程师不同）

工资报表



将工资报表转成数据表



插入异常：有一个员工入职，还没有参加工程，但工程名称不能为空，必须给插入一个虚拟的工程名称

更新异常：员工从“技术员”升到“工程师”，但可能小时工资率没有更改。

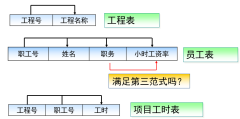
删除异常：员工离职，将员工删除，员工删除后工程也删除了。

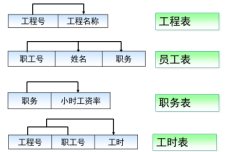
**总结：此表不能做数据表，增、删、改都有问题。**

解决：



**项目工时表就是工程信息和员工的关系表**





## **1.19 **运算符****

### **1.19.1 **算术运算符****

+

-

\*

/

%

**脚下留心：MySQL的算术运算符没有++和--**

### **1.19.2 **关系运算符****

>

>=

<

<=

=

!=或<>

**脚下留心：MySQL的“=”既是等于也是赋值。**

### **1.19.3 **逻辑运算符****

and 与

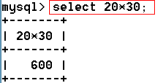
or 或

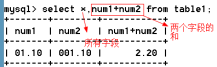
not 非

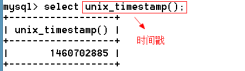
## **1.20 **查询语句****

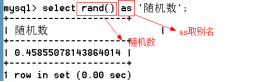
语法：select  [选项] 列名 from 表名 [where 条件] [group by 分组] [having 条件] [order by 排序] [limit 限制]

### **1.20.1 **字段表达式列表****

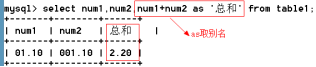


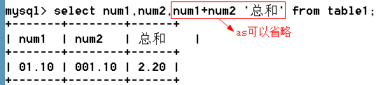






**多学一招：as用来给字段去别名，as可以省略**





### **1.20.2 **from子句****

from后面跟的是数据源，如果数据源有多张表，返回的是笛卡尔积。

--插入测试数据

create table stu\_info(

       name varchar(10),

       sex char(1)

);

create table stu\_marks(

       ch tinyint,

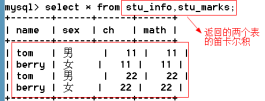
       math tinyint

);

insert into stu\_info values ('tom','男'),('berry','女');

insert into stu\_marks values (11,11),(22,22);

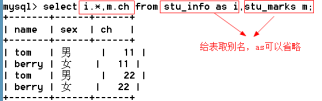
--测试from后面跟多个表



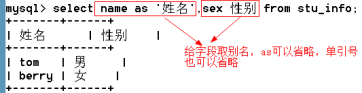
--查询表的部分字段



--给表取别名

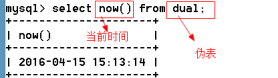


--给字段取别名



### **1.20.3 **dual表（伪表）****

dual表是一个虚拟表，在有些特定的情况下不能省略from子句，但是又没有一个表，这时候就用伪表。dual表就是为了保证查询语句的完整性。



### **1.20.4 **where子句****

插入测试数据

create table stu

(

stuNo char(6) primary key,

stuName varchar(10) not null,

stuSex char(2) not null,

stuAge tinyint not null ,

stuSeat tinyint not null,

stuAddress varchar(10) not null,

ch tinyint,

math tinyint

);

insert into stu values ('s25301','张秋丽','男',18,1,'北京',80,null);

insert into stu values ('s25302','李文才','男',31,3,'上海',77,76);

insert into stu values ('s25303','李斯文','女',22,2,'北京',55,82);

insert into stu values ('s25304','欧阳俊雄','男',28,4,'天津',null,74);

insert into stu values ('s25305','诸葛丽丽','女',23,7,'河南',72,56);

insert into stu values ('s25318','争青小子','男',26,6,'天津',86,92);

insert into stu values ('s25319','梅超风','女',23,5,'河北',74,67);

在数据源中进行查询

--年龄超过20的女生

IMG_353

IMG_354

IMG_355

**思考：既然查询得到的“结果集”的结构类似于一张表，那么可以在“结果集”上继续进行查询吗？**

**答：可以的，这就是子查询或高级查询。**

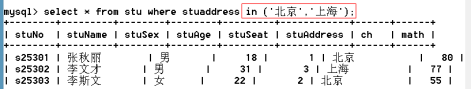
### **1.20.5 **in | not in****

把字段中内容与所列出的查询内容列表匹配的记录查询出来。

--查询“北京、上海”的学生

IMG_356

--用in语句实现



--查询不是“北京”和“上海”的学生

IMG_358

### **1.20.6 **between and | not between and****

在指定范围之间 | 不在指定范围之间

--查询年龄在20到30之间的学生

IMG_359

--用between and实现

IMG_360

--查询年龄不在20到30之间的学生

IMG_361

### **1.20.7 **is null | is not null****

--取出语文或数学缺考的学生

IMG_362

--取出没有缺考的学生

IMG_363

### **1.20.8 **聚合函数****

|  |  |
| --- | --- |
| 聚合函数 | 描述 |
| sum() | 求和 |
| avg() | 求平均值 |
| max() | 最大值 |
| min() | 最小值 |
| count() | 记录数 |

select sum(ch) as '语文总分', avg(ch) as '语文平均分',max(ch) as '语文最高分',min(ch) as '语文最低分',count(\*) 总人数 from stu;

IMG_364

### **1.20.9 **通配符****

|  |  |
| --- | --- |
| 通配符 | 描述 |
| \_  (下划线) | 表示任意一个字符 |
| % | 表示任意字符 |

1、’t\_m’下列选项符合的条件的是( B,D)

A：tm B：tom C:toom D:tam E:tmo

2、’张%’ 下列选项符合的条件的是（A、B、C、D）

A：张三 B：张三丰 C：张牙舞爪  D：张   E：小张

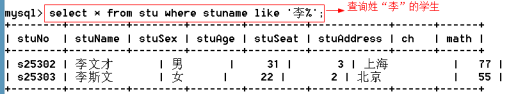
3、’%诺基亚%’  下列选项符合的条件的是(A、B、C、D )

A：诺基亚2100   B :2100诺基亚  C：我的诺基亚质量很好 D：诺基亚

4、SELECT \* FROM 数据表 WHERE 编号 LIKE '00\_A5%C’  (D)

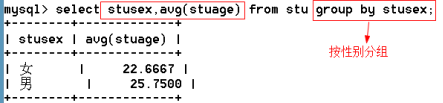
A、0890AC   B、007\_A51C    C、00A54C   D、00AA5C

### **1.20.10 **模糊查询 （like）****

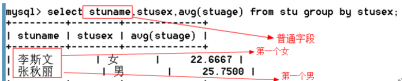


### **1.20.11 **分组查询  （group by）****

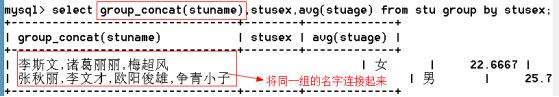
将查询的结果分组展示，分组的目的在于统计数据。



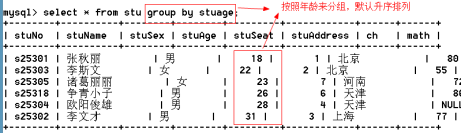
**脚下留心：一般来说，分组语句中的显示字段是聚合函数和分组字段名，如果用普通字段，只取分组字段的第一个值。**



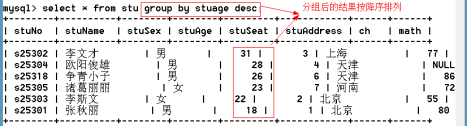
**可以通过group\_concat()函数将同一组的值连接起来**



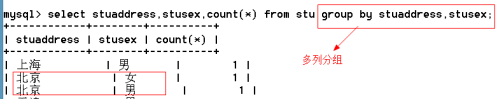
分组后的结果默认按照升序排列



可以指定分组后的结果按降序排列

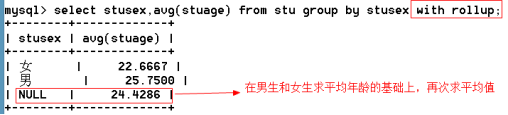


多列分组



### **1.20.12 **回溯统计****

在分组统计的基础上再次进行相同的统计（with rollup）



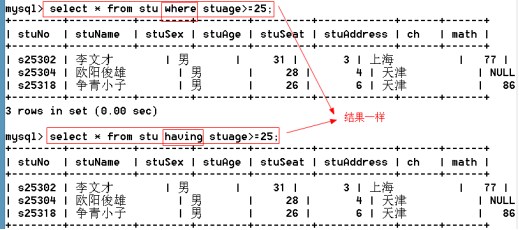
### **1.20.13 **having条件****

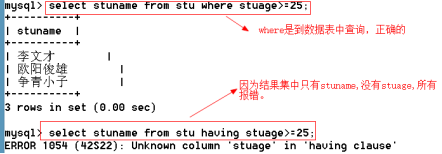
where和having的区别：

1、 where是对表的数据进行筛选

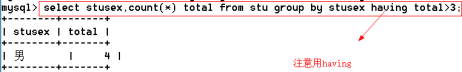
2、 having是对结果集的数据进行筛选，having是对查询的结果再次进行筛选。

having设置了查询条件，条件字段必须在查询结果集中存在。





**思考：查找班级总人数超过3个人的性别**



### **1.20.14 **order by排序****

排序有升序和降序（asc | desc）

mysql> select \* from stu order by stuage asc; --按年龄升序

mysql> select \* from stu order by stuage;   -- 默认是升序

mysql> select \* from stu order by stuage desc; --降序

多列排序：

mysql> select \* from stu order by stuage desc,ch desc; --按年龄从大到小排列，如果年龄一样大，按照语文成绩从大到小排列

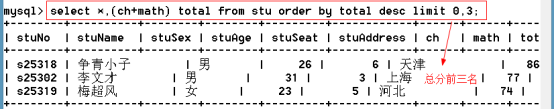
select \* from stu order by stuage,ch desc  --按年龄从小到大排列，如果年龄一样大，按照语文成绩从大到小排列

select \* from stu order by stuage,ch;  --按年龄从小到大排列，如果年龄一样大，按照语文成绩从小到大排列

### **1.20.15 **limit 限制****

语法：limit 起始位置，显示长度

起始位置可以省略，如果省略默认是0



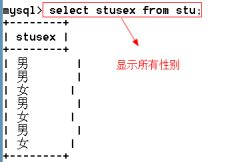
IMG_377

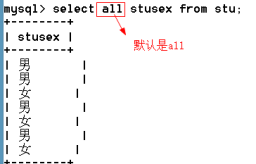
### **1.20.16 **查询语句中的选项****

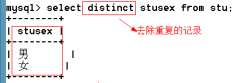
查询语句中的选项有两个

1、 all：显示所有的记录（默认）

2、 distinct：去除相同的记录







## **1.21 **insert…select语句****

将一个表中的数据插入到另一个表中

mysql> create table user1(

    -> name varchar(20),

    -> sex char(1)

    -> );

Query OK, 0 rows affected (0.01 sec)

--将stu表中的stuname和stusex插入到user1表中

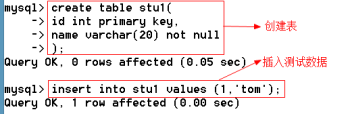
mysql> insert into user1 select stuname,stusex from stu;

Query OK, 7 rows affected (0.00 sec)

Records: 7  Duplicates: 0  Warnings: 0

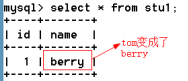
## **1.22 **on duplicate key update****

如果插入一个已经存在的主键或不满足唯一约束的值就执行update操作。



--测试   插入1 berry ,如果不能插入就执行更新

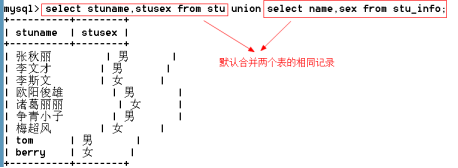
mysql> insert into stu1 values (1,'berry') on duplicate key update name='berry';



## **1.23 **union****

作用：将多条select语句纵向联合起来。

语法：select 语句 union [选项] select 语句 union [选项] select 语句…

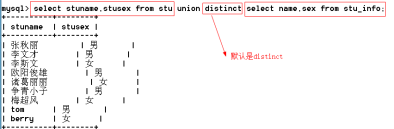


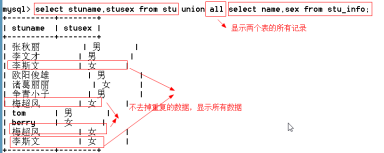
### **1.23.1 **union的选项****

union的选项有两个

1、 all：显示两个表中的所有记录

2、 distinct：去掉两个表中相同的记录 （默认）





### **1.23.2 **例题****

1、 查找上海的男生和北京的女生

-- 方法一：

select \* from stu where (stuaddress='上海' and stusex='男') or (stuaddress='北京' and stusex='女');

--方法二：

select \* from stu where (stuaddress='上海' and stusex='男') union

select \* from stu where (stuaddress='北京' and stusex='女');

IMG_386

**结论：将复杂的条件简化成两个简单的条件**

2、 将多个表的数据插入到一个表中

--创建雇员表

mysql> create table emp(

    -> id int unsigned auto\_increment primary key,

    -> name varchar(20) not null,

    -> sex char(1) not null

    -> );

--插入数据

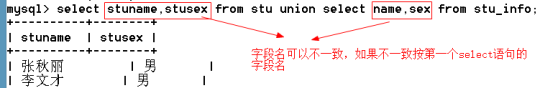
insert into emp (name,sex) select stuname,stusex from stu union all select name,sex from stu\_info;

### **1.23.3 **union的使用要求****

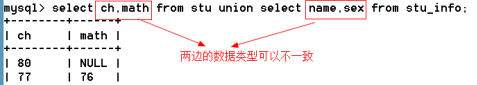
1、 union两边的select语句的字段个数必须一致。

IMG_387

2、 字段名可以不一致，最终按第一个select语句的字段名。



3、 union两边的select语句中的字段的数据类型可以不一致

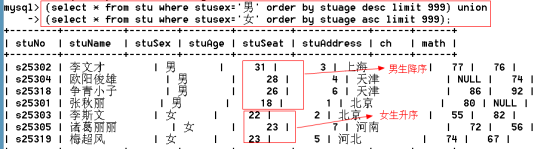


### **1.23.4 **union与order by一起使用****

1、 union两边select语句必须用括号括起来

2、 每条select语句必须配合limit才起作用

男生按年龄的降序排列，女生按年龄的升序排列



**[好文要顶](https://www.cnblogs.com/wangyanzi/p/javascript:void(0);)** **[关注我](https://www.cnblogs.com/wangyanzi/p/javascript:void(0);)** **[收藏该文](https://www.cnblogs.com/wangyanzi/p/javascript:void(0);)** **[IMG_391](https://www.cnblogs.com/wangyanzi/p/javascript:void(0);)** **[IMG_392](https://www.cnblogs.com/wangyanzi/p/javascript:void(0);)**