**《数据库系统原理》实验报告**

实验题目：数据类型的使用

姓名：刘俊傲 实验日期：2018年 12月 11日

**实验内容及完成情况：**

1. **了解并尝试使用 MySQL 的各数据类型，不作硬性要求**
   1. **日期和时间数据类型**

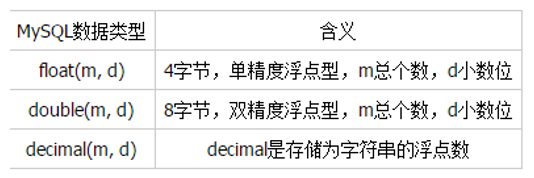
****

**1.2 整型**

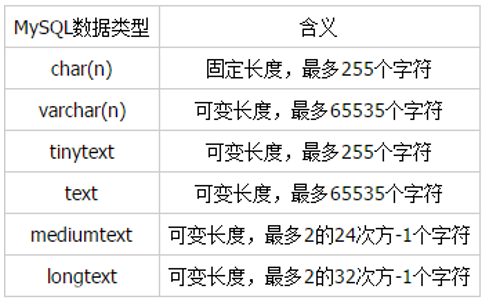
****

**上面定义的都是有符号的，也可以加上 unsigned 关键字，定义成无符号的类型，那么对应 的取值范围就要翻番了，比如：tinyint unsigned 的取值范围为 0~255**

**1.3 浮点型**

****

**1.4 字符串类型**

****

**①char（n）和 varchar（n）中括号中 n 代表字符的个数，并不代表字节个数，所以当使用了 中文的时候(UTF8)意味着可以插入 m 个中文，但是实际会占用 m\*3 个字节**

**②同时 char 和 varchar 最大的区别就在于 char 不管实际 value 都会占用 n 个字符的空间，而varchar 只会占用实际字符应该占用的空间+1，并且实际空间+1<=n**

**③超过 char 和 varchar 的 n 设置后，字符串会被截断**

**④char 在存储的时候会截断尾部的空格，varchar 和 text 不会**

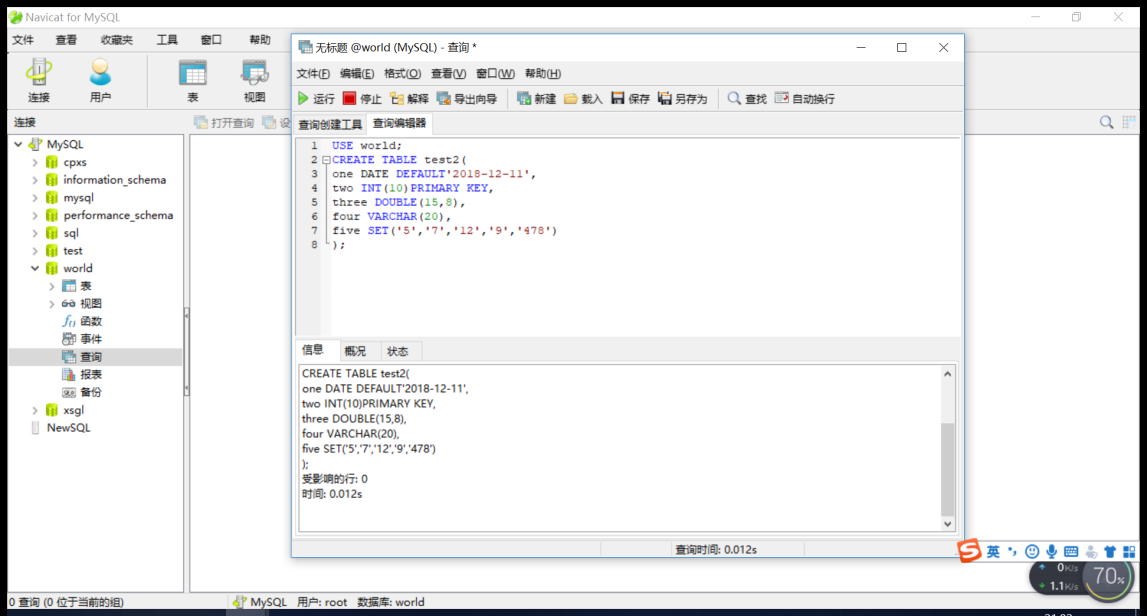
**⑤varchar 会使用 1-3 个字节来存储长度，text 不会**

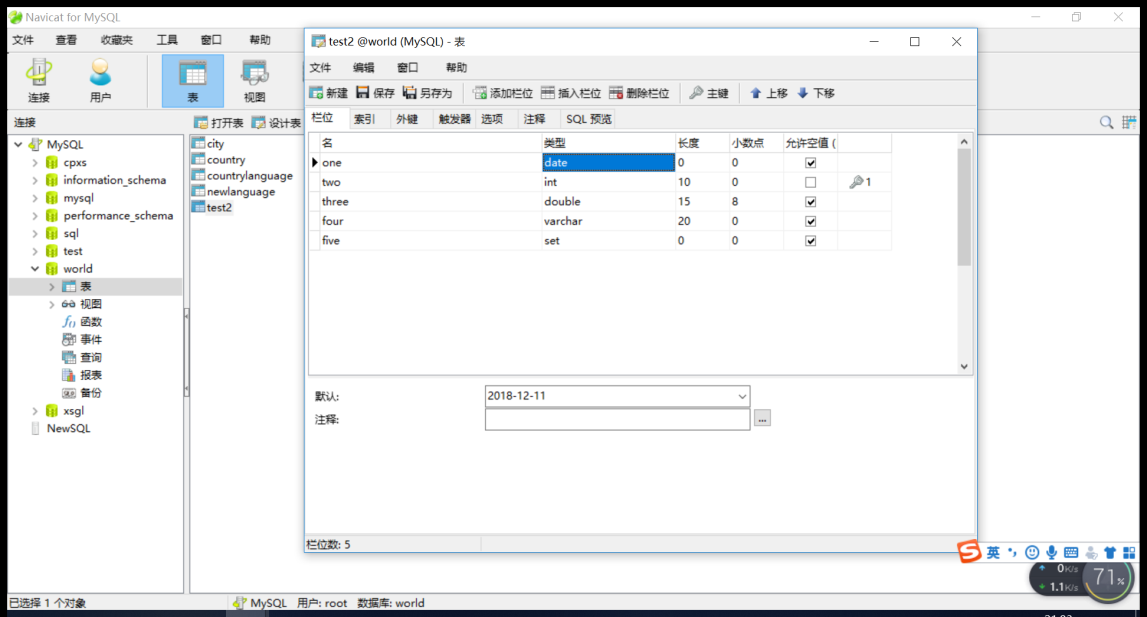
**1.5 其他类型**

**① enum(“member1″, “member2″, … “member65535″)：enum 数据类型就是定义了 一种枚举，最多包含 65535 个不同的成员。当定义了一个 enum 的列时，该列的值限制为列 定义中声明的值。如果列声明包含 NULL 属性，则NULL 将被认为是一个有效值，并且是 默认值。如果声明了 NOT NULL，则列表的第一个成员是默认值。**

**②set(“member”, “member2″, … “member64″)：set 数据类型为指定一组预定义值中 的零个或多个值提供了一种方法，这组值最多包括 64 个成员。值的选择限制为列定义中声 明的值。**

**选择每个类型中的一个数据建立的表test2**

****

****

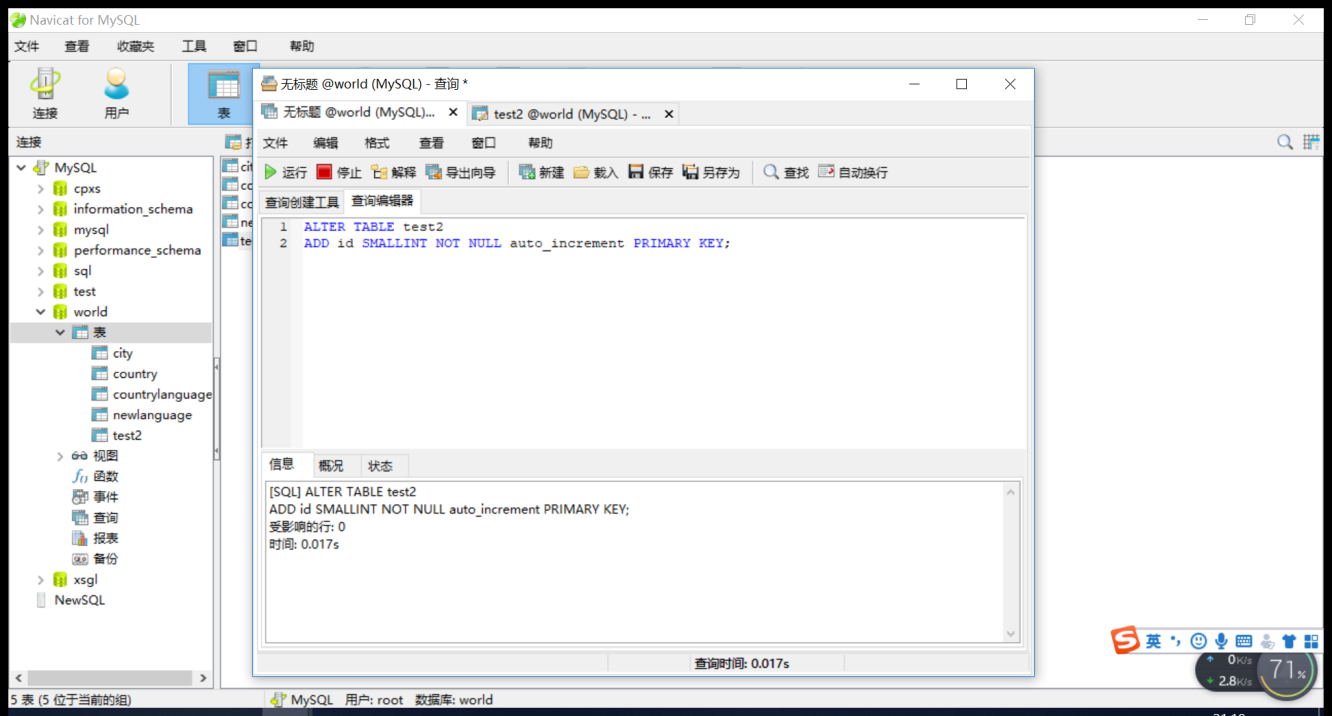
**2.了解 MySQL 数据类型属性，并依次实际操作、验证**

**2.1 auto\_increment**

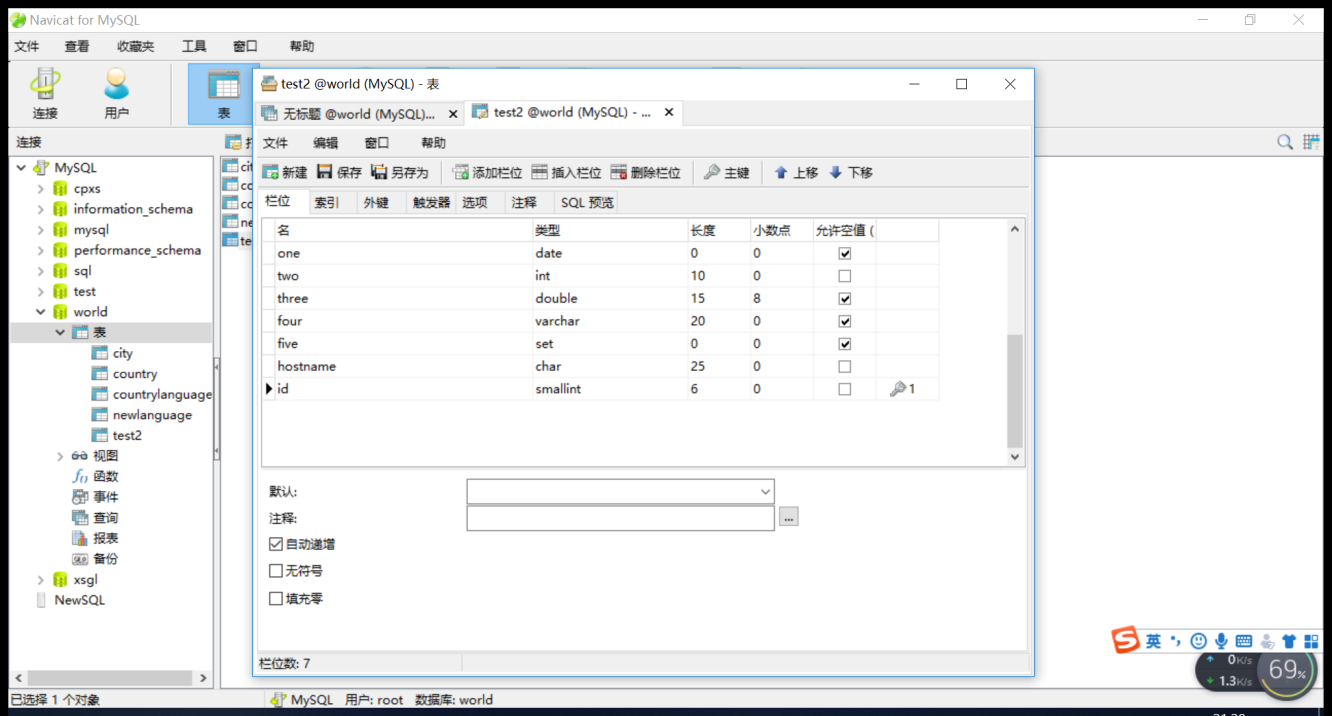
**auto\_increment 能为新插入的行赋一个唯一的整数标识符。为列赋此属性将为每个新插入的行赋值为上一次插入的 ID+1。**

**MySQL要求将 auto\_increment 属性用于作为主键的列。此外，每个表只允许有一个auto\_increment 列。**

**SQL语言如下：id smallint not null auto\_increment primary key**

****

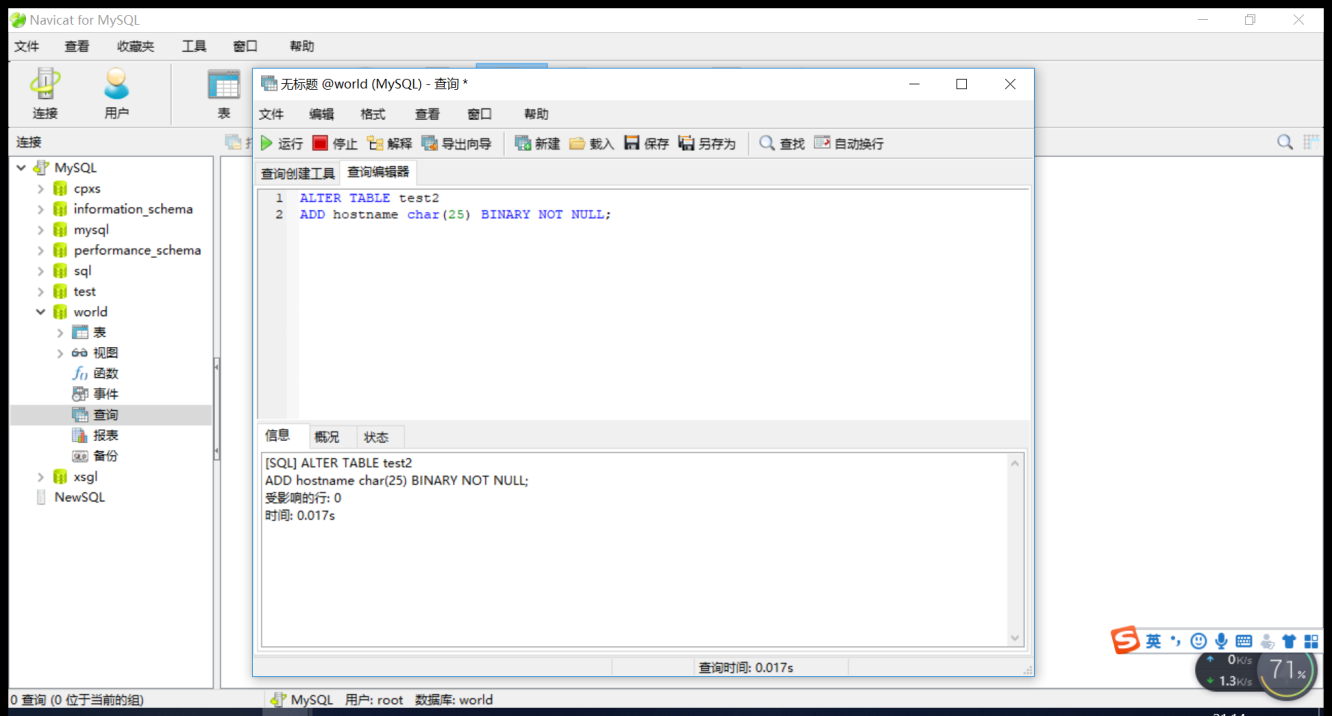
**检测结果：**

****

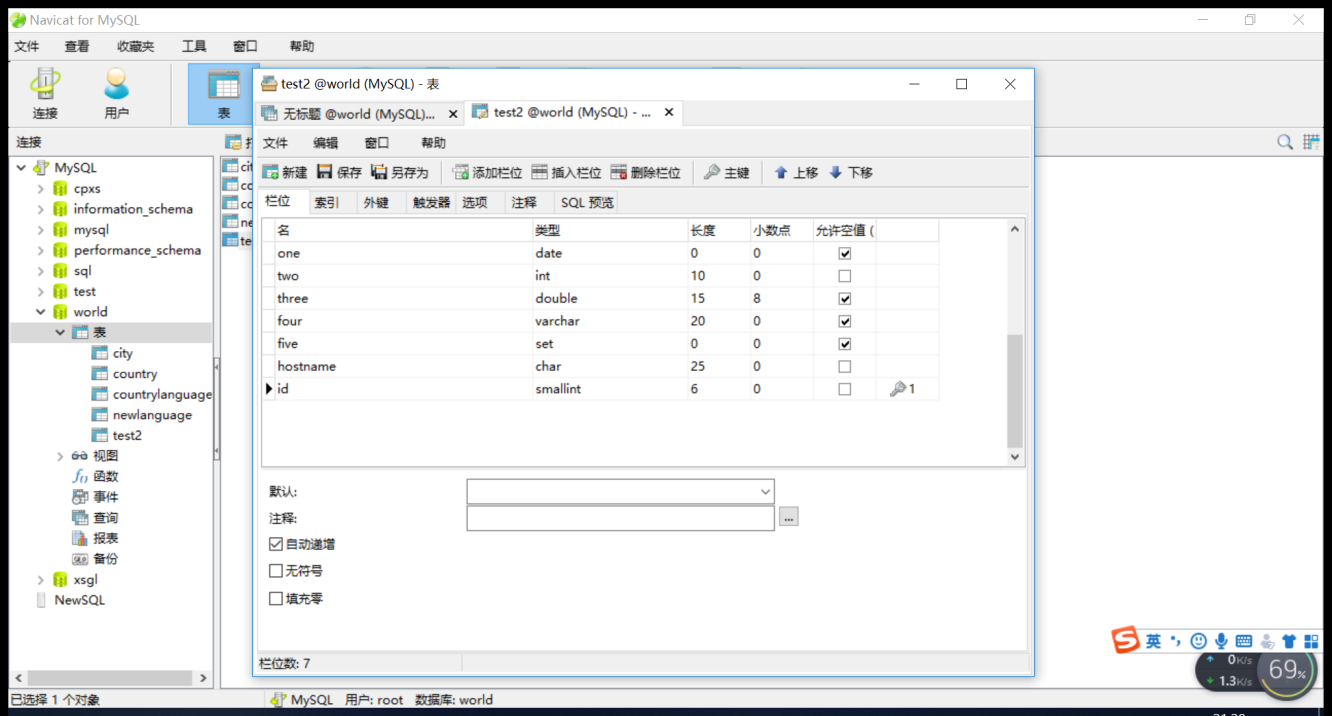
**2.2 binary**

**binary 属性只用于 char 和 varchar 值。当为列指定了该属性时，将以区分大小写的方式 排序。与之相反，忽略 binary 属性时，将使用不区分大小写的方式排序。**

**SQL语言如下：hostname char(25) binary not null**

****

**检测结果：**

****

**2.3 default**

**default 属性确保在没有任何值可用的情况下，赋予某个常量值，这个值必须是常量，因 为 MySQL 不允许插入函数或表达式值。此外，此属性无法用于 BLOB 或 TEXT 列。如果已 经为此列指定了 NULL 属性，没有指定默认值时默认值将为 NULL，否则默认值将依赖于 字段的数据类型**

**例如：subscribed enum('0', '1') not null default '0'**

**2.4 index**

**如果所有其他因素都相同，要加速数据库查询，使用索引通常是最重要的一个步骤。索 引一个列会为该列创建一个有序的键数组，每个键指向其相应的表行。以后针对输入条件可 以搜索这个有序的键数组，与搜索整个未索引的表相比，这将在性能方面得到极大的提升**

**create table employees (**

**id varchar(9) not null,**

**firstname varchar(15) not null,**

**lastname varchar(25) not null,**

**email varchar(45) not null,**

**phone varchar(10) not null,**

**index lastname(lastname),**

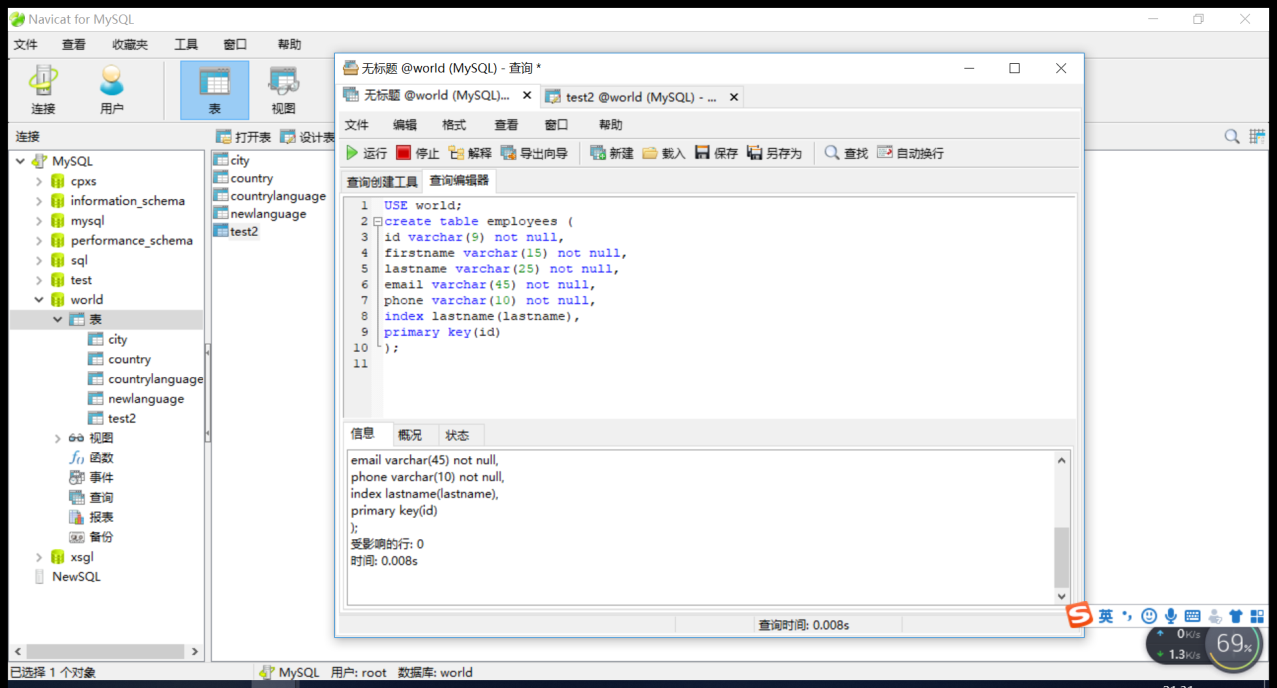
**primary key(id)**

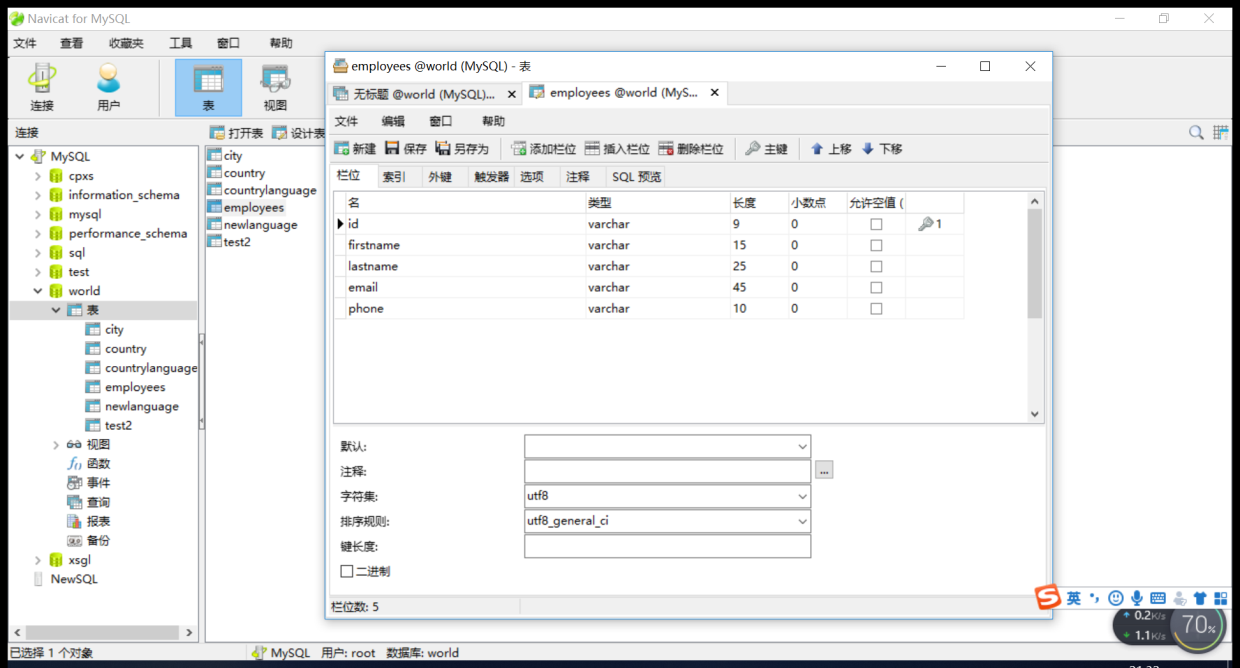
**);**

**我们也可以利用 MySQL 的 create index 命令在创建表之后增加索引：**

**create index lastname on employees (lastname(7));**

**这一次只索引了名字的前 7 个字符，因为可能不需要其它字母来区分不同的名字。因为使用较小的索引时性能更好，所以应当在实践中尽量使用小的索引**

****

****

**2.5 not null**

**如果将一个列定义为 not null，将不允许向该列插入 null 值。建议在重要情况下始终使 用 not null 属性，因为它提供了一个基本验证，确保已经向查询传递了所有必要的值**

**2.6 null**

**为列指定 null 属性时，该列可以保持为空，而不论行中其它列是否已经被填充。记住，null 精确的说法是“无”，而不是空字符串或 0**

**2.7 primary key**

**primary key 属性用于确保指定行的唯一性。指定为主键的列中，值不能重复，也不能 为空。为指定为主键的列赋予 auto\_increment 属性是很常见的，因为此列不必与行数据有任 何关系，而只是作为一个唯一标识符。主键又分为以下两种：**

**①单字段主键: 如果输入到数据库中的每行都已经有不可修改的唯一标识符，一般会使 用单字段主键。注意，此主键一旦设置就不能再修改**

**②多字段主键: 如果记录中任何一个字段都不可能保证唯一性，就可以使用多字段主 键。这时，多个字段联合起来确保唯一性。如果出现这种情况，指定一个 auto\_increment 整 数作为主键是更好的办法**

**2.8 unique**

**被赋予 unique 属性的列将确保所有值都有不同的值，只是 null 值可以重复。一般会指 定一个列为 unique，以确保该列的所有值都不同。**

**例如：email varchar(45) unique**

**2.9 zerofill**

**zerofill 属性可用于任何数值类型，用 0 填充所有剩余字段空间。例如，无符号 int 的默 认宽度是 10；因此，当“零填充”的 int 值为 4 时，将表示它为 0000000004**

**例如：orderid int unsigned zerofill not null**

**3.如果要设计一个表用来存放商品的基本信息，包括商品编号,商品名称,品牌商标,型号,产地,生产厂商,生产日期,保质期,进货价格,销售价格和商品图片等。请设计出这个表的表结构，包括表名,列名,列的数据类型和长度等等**

**CREATE TABLE information1(**

**cpbh CHAR(10) PRIMARY KEY,**

**spmc VARCHAR(8) NOT NULL,**

**ppsb CHAR(10) UNIQUE NOT NULL,**

**xh CHAR(12) UNIQUE NOT NULL,**

**sccs CHAR(9) NOT NULL,**

**scrq DATE DEFAULT'2018-12-11',**

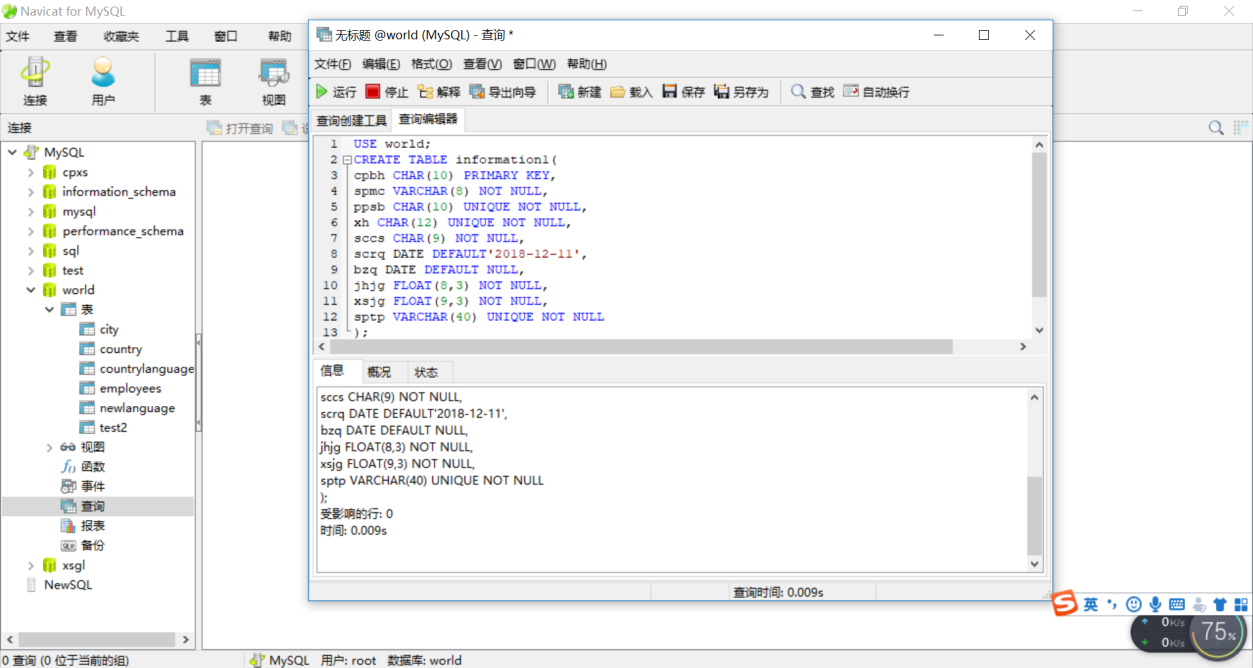
**bzq DATE DEFAULT NULL,**

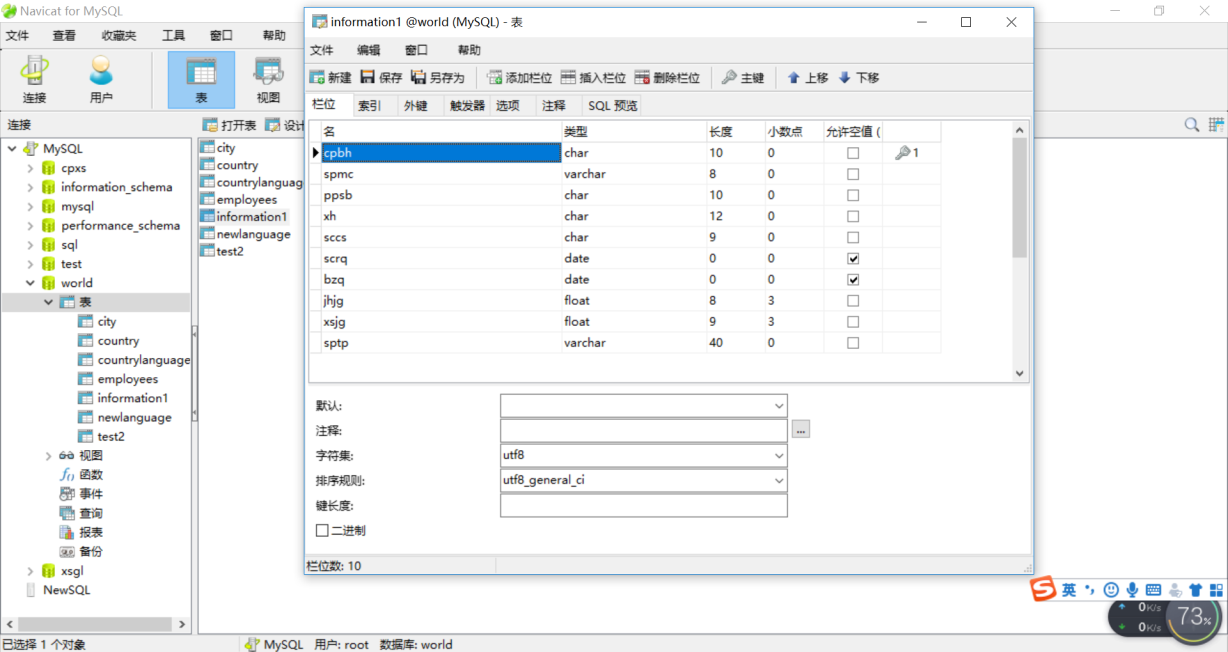
**jhjg FLOAT(8,3) NOT NULL,**

**xsjg FLOAT(9,3) NOT NULL,**

**sptp VARCHAR(40) UNIQUE NOT NULL**

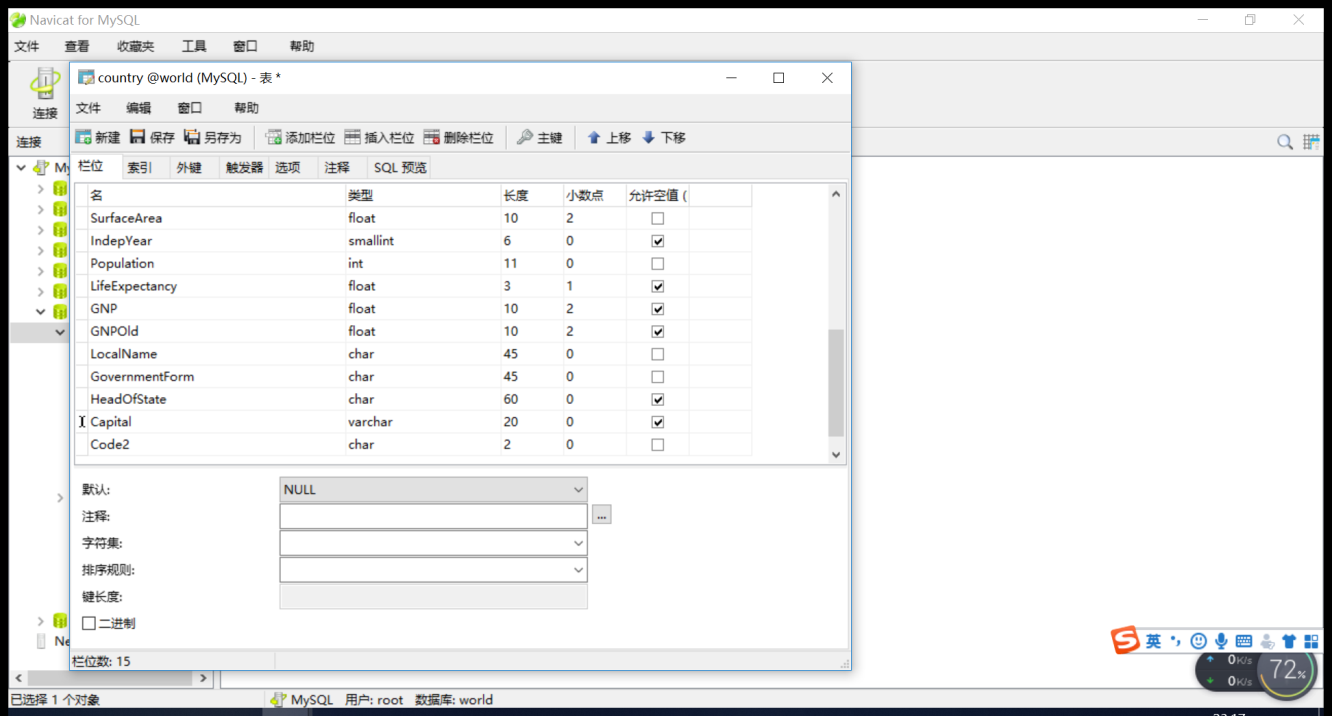
**);**

****

****

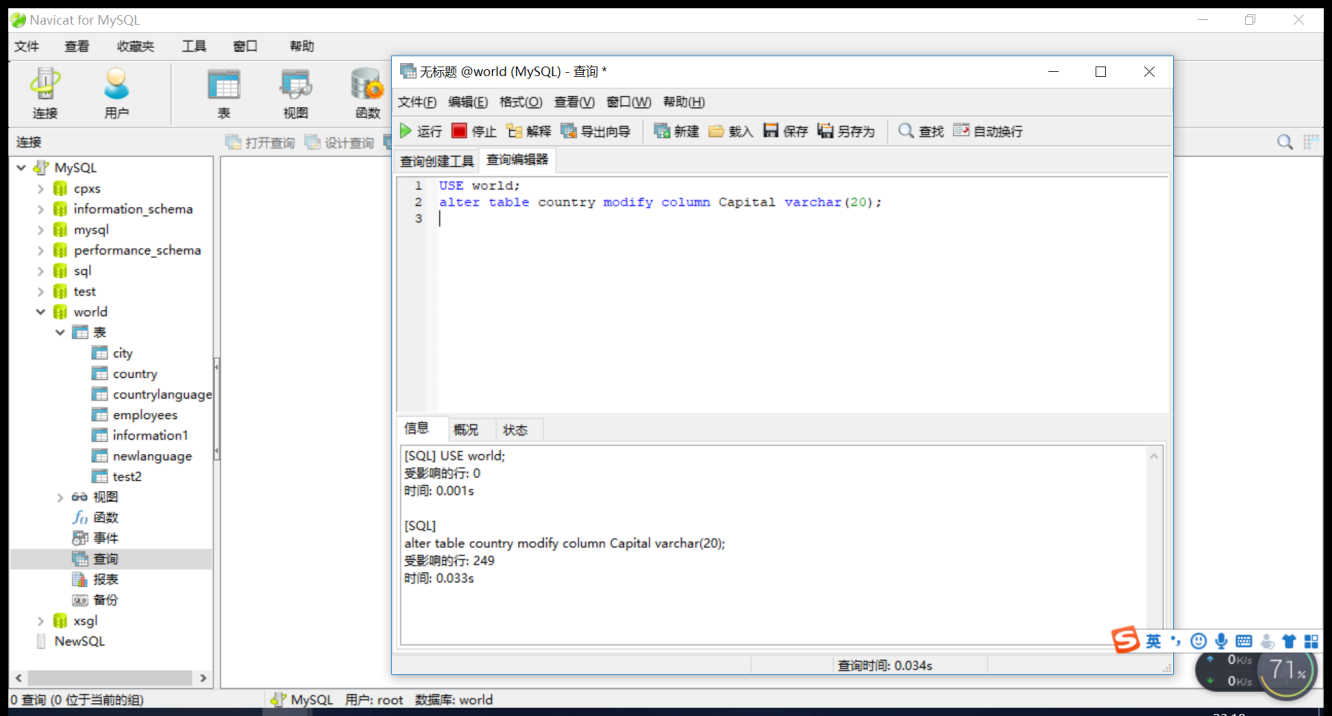
**4.将数据库 world 中 country 表的 capital 字段的数据类型改为 varchar（20）**

**① Navicat—设计表**

****

**② 查询编辑器，输入以下语句：**

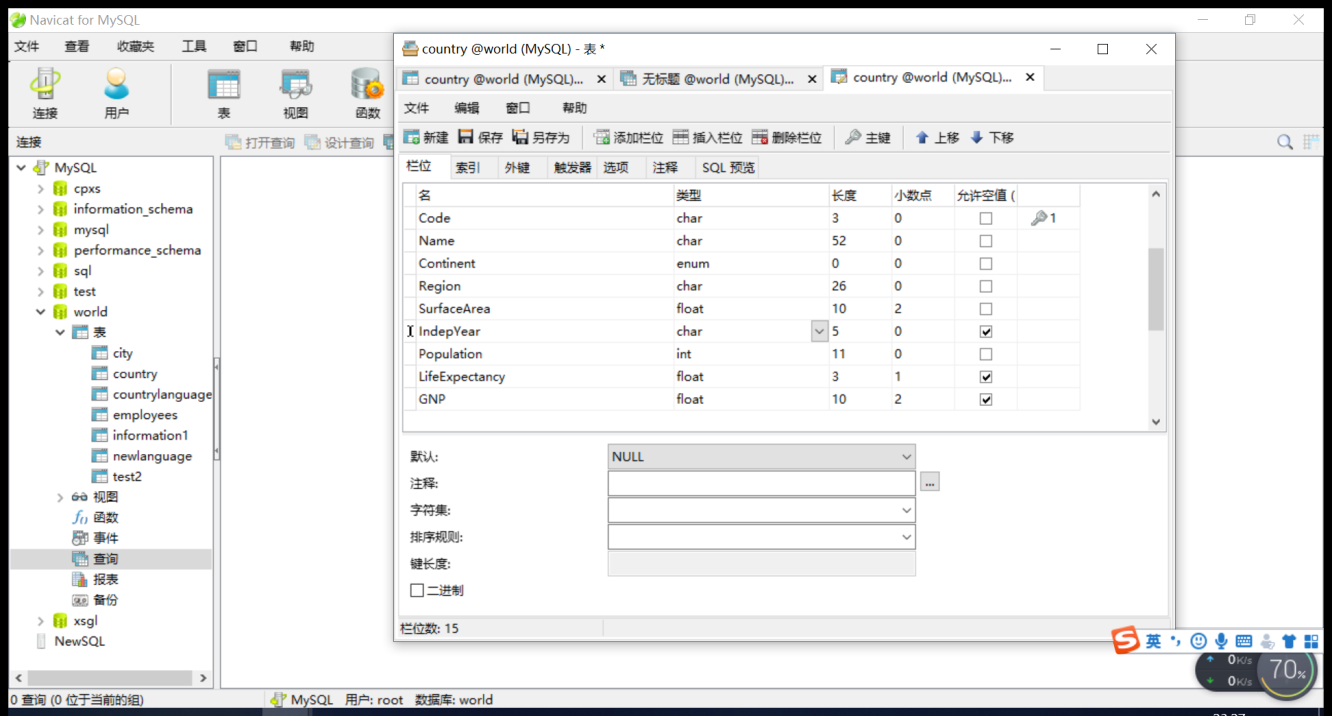
**alter table country modify column Capital varchar(20);**

****

**三、课后练习题**

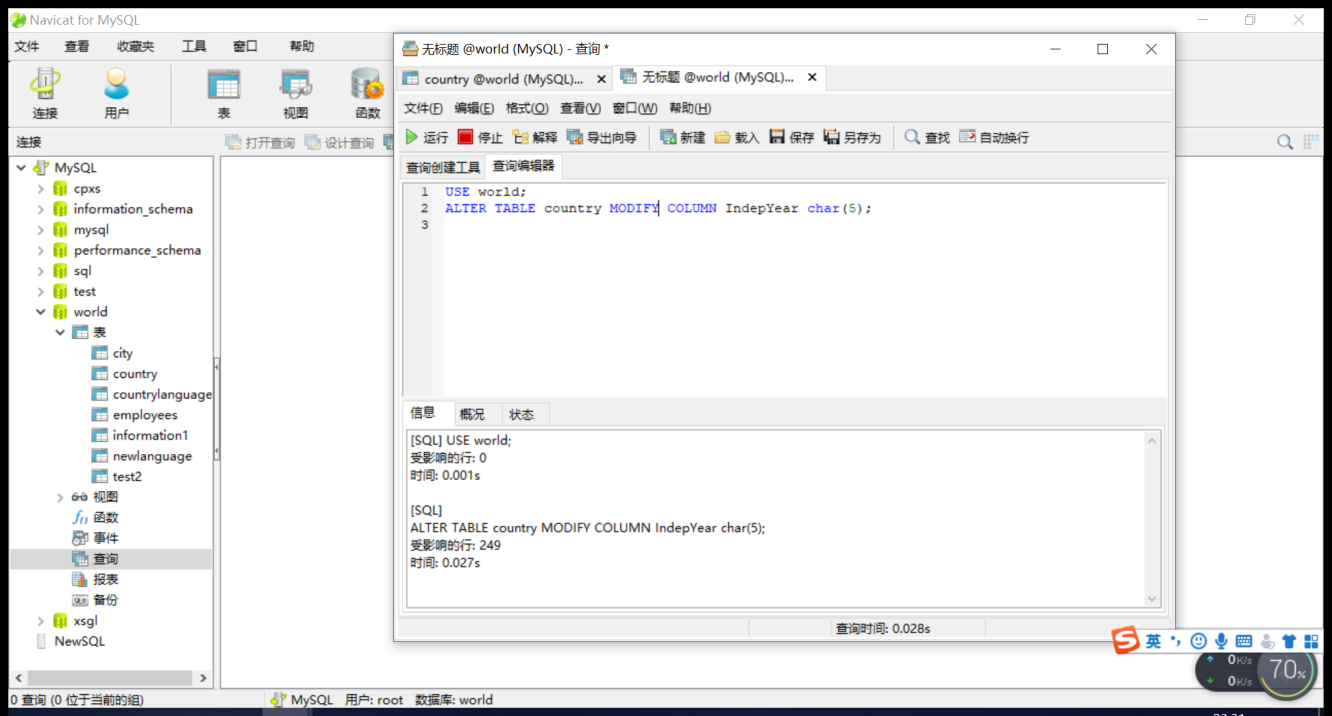
**1.用 Navicat 和 SQL 语言两种方式将数据库 world 中 Indepyear 字段的数据类型改为char（5）**

**① Navicat—设计表**

****

**② 查询编辑器，输入以下语句：**

**ALTER TABLE country MODIFY COLUMN IndepYear char(5);**

****

**2.在 MySQL 中建立一个表，有一列为 float(5, 3).做以下试验：**

**①插入12 .345，成功则查询该表得到结果；**

**②插入12.3456，成功则查询该表得到结果；**

**③插入123.456，成功则查询该表得到结果。**

**比较三次结果的差异，并分析原因**

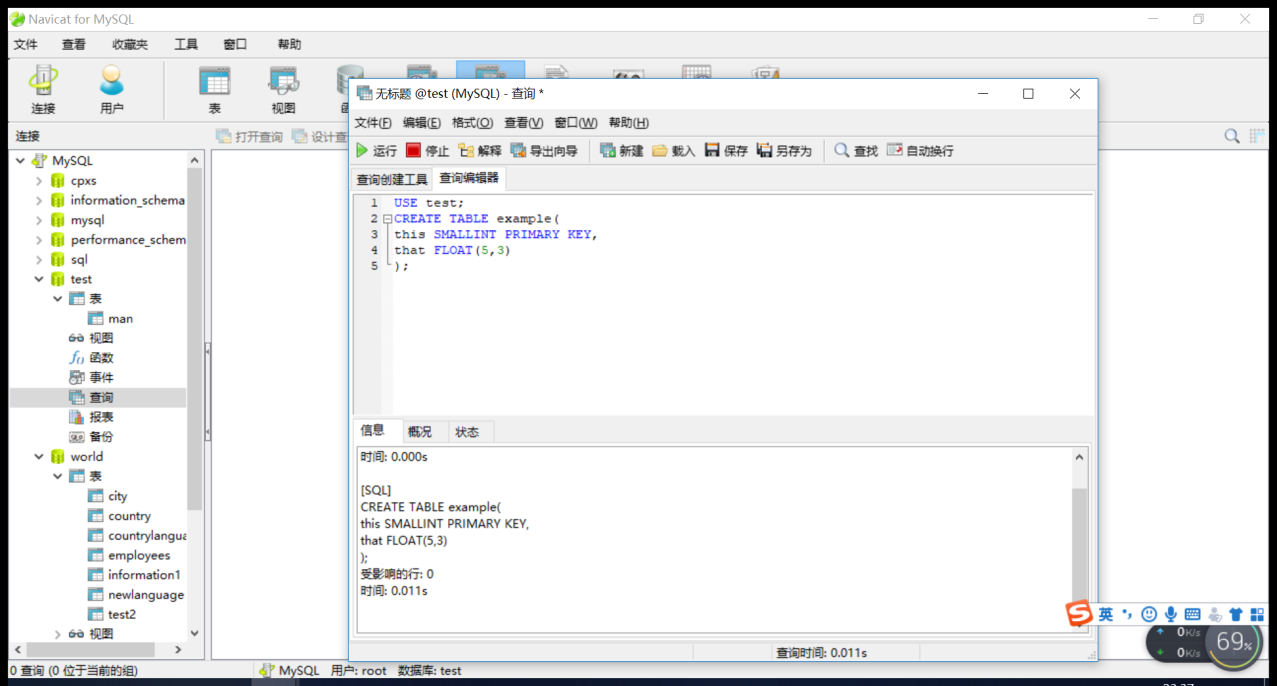
**新建表example**

**CREATE TABLE example(**

**this SMALLINT PRIMARY KEY,**

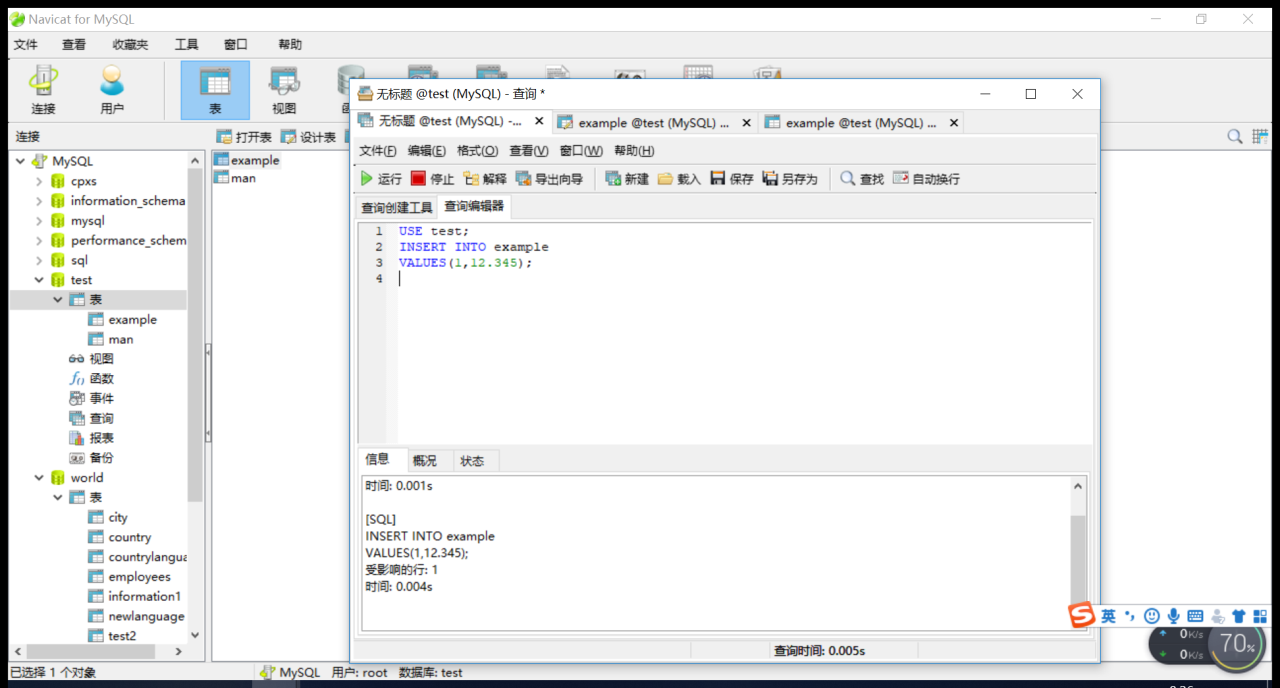
**that FLOAT(5,3)**

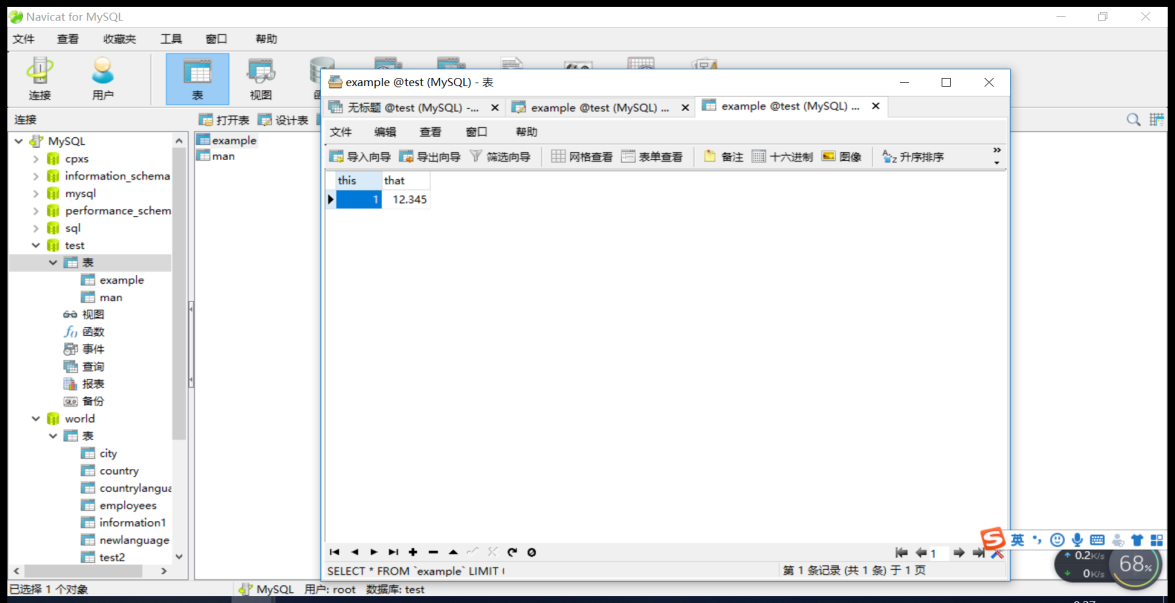
**);**

****

**分别插入上述三个数据，得到如下结果**

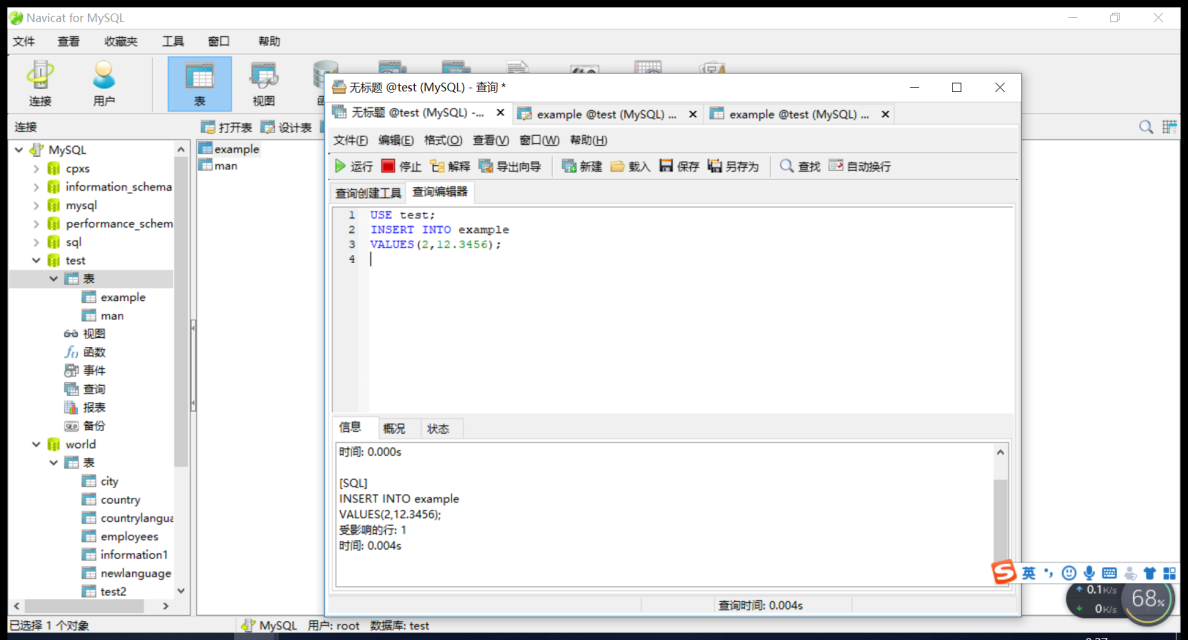
**① 插入12 .345： INSERT INTO example VALUES(1,12.345);**

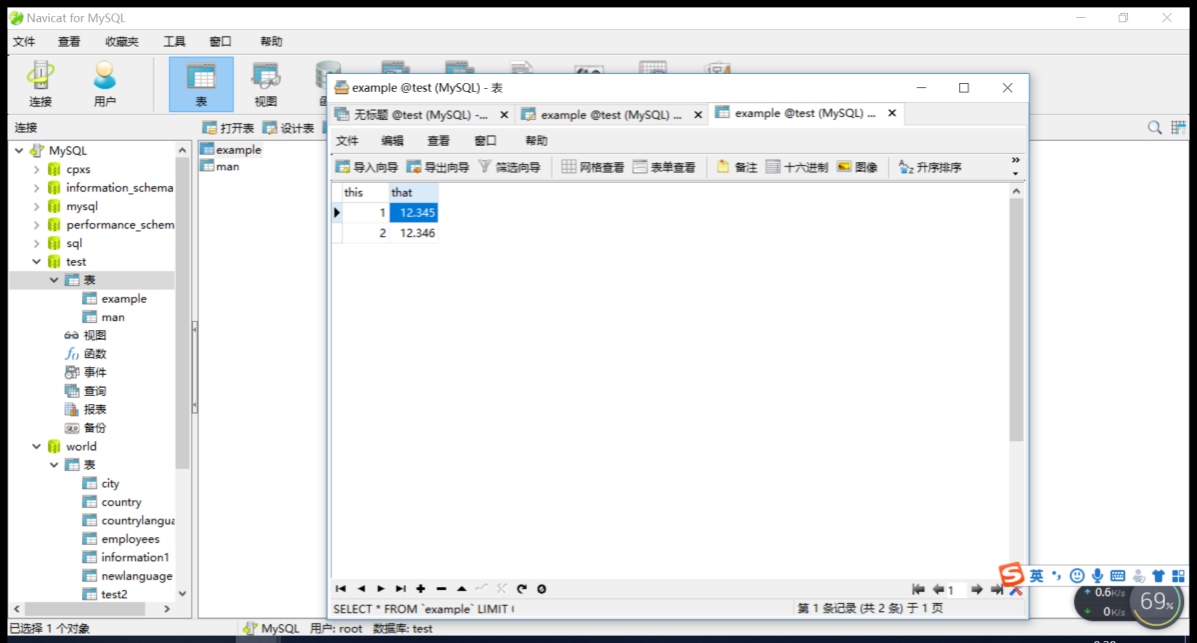
****

****

**正常显示12.345**

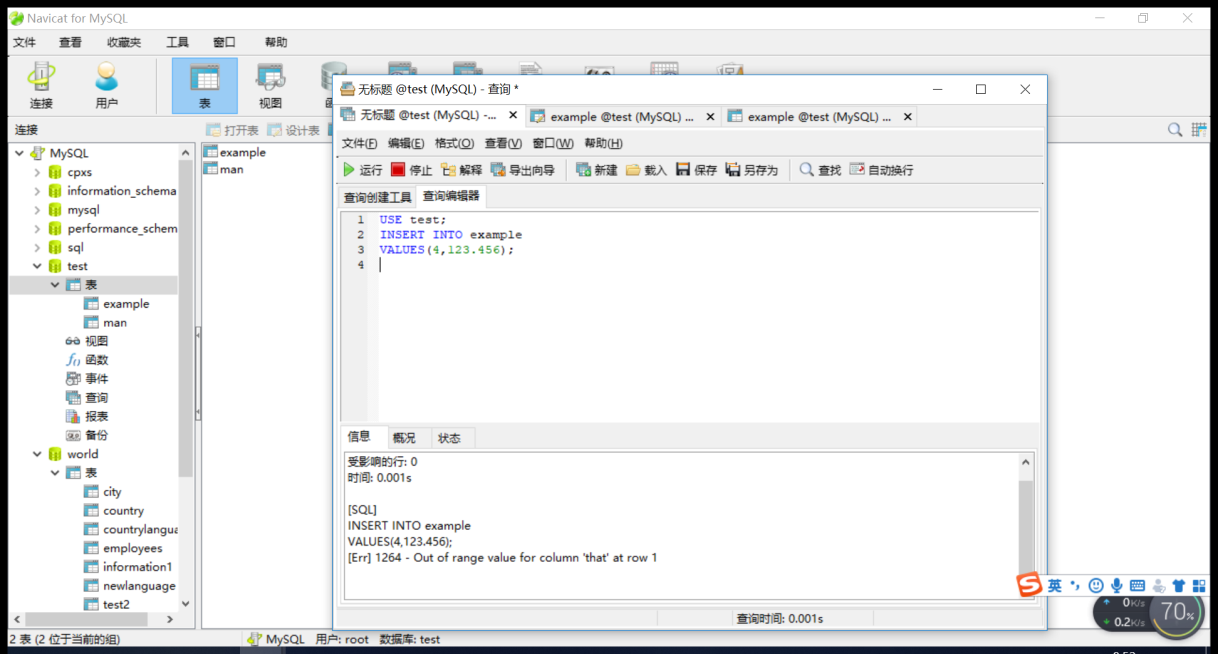
**② 插入12.3456： INSERT INTO example VALUES(2,12.3456);**

****

****

**显示12.346**

**③ 插入123.456： INSERT INTO example VALUES(3,123.456);**

****

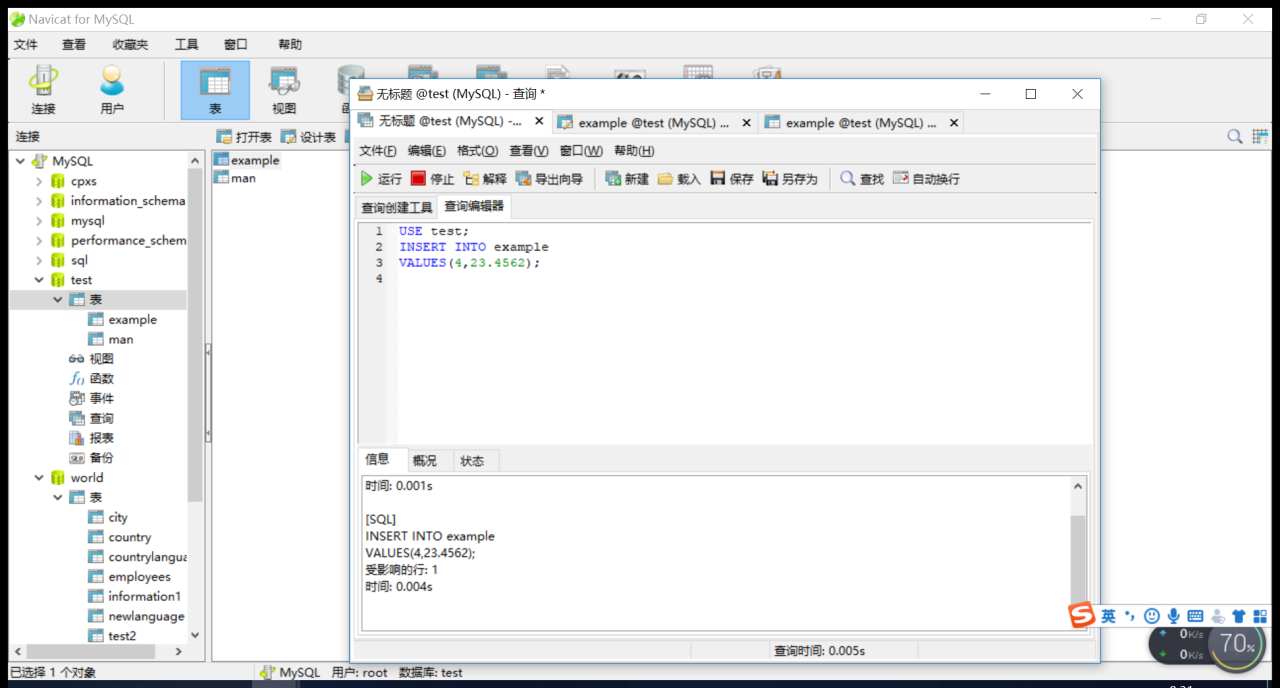
**[Err] 1264 - Out of range value for column 'that' at row 1超出范围报错**

**猜想：对于第②种情况的分析，觉得应该是超出范围后小数位四舍五入进位缩小范围**

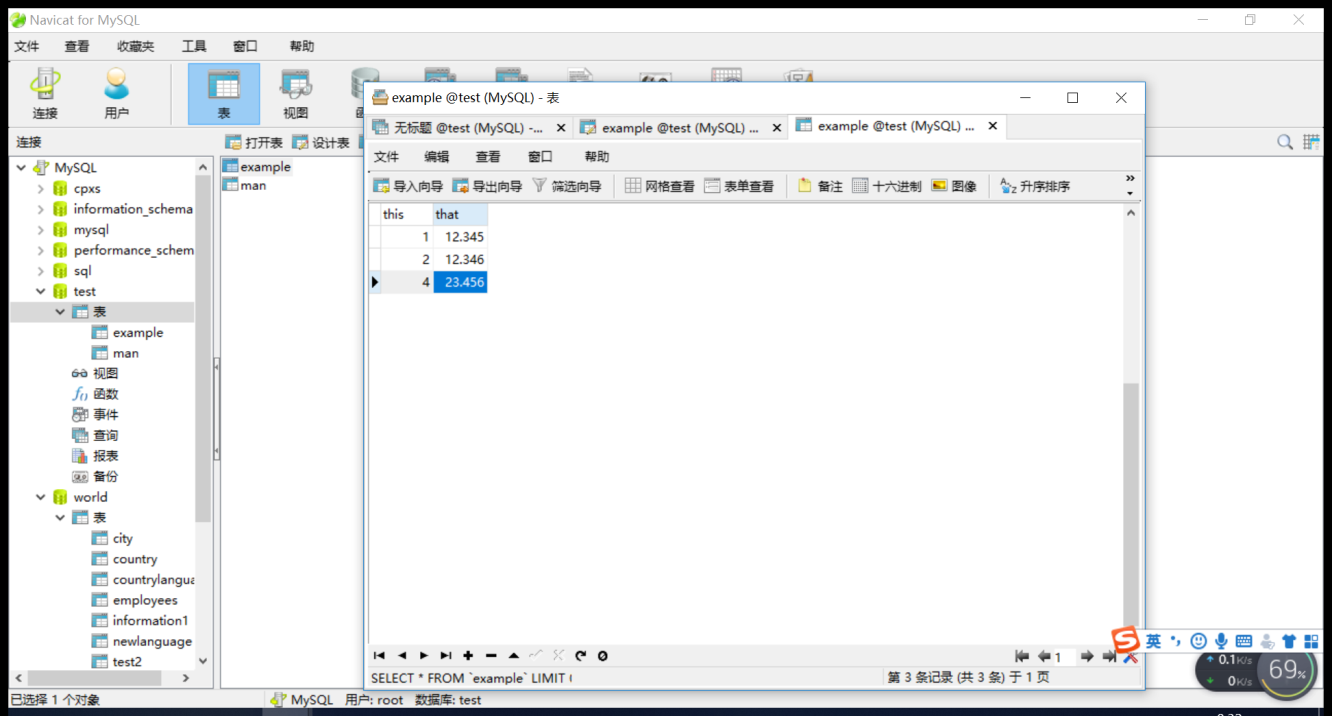
**下面插入23.4562进行分析，若猜想成立则应该存入23.456**

**INSERT INTO example VALUES(4,23.4562);**

**插入23.4562**

****

**结果如下**

****

**存入的数据确为23.456，初步验证猜测正确**

出现的问题及解决方案：

操作正常

教师评语及成绩：