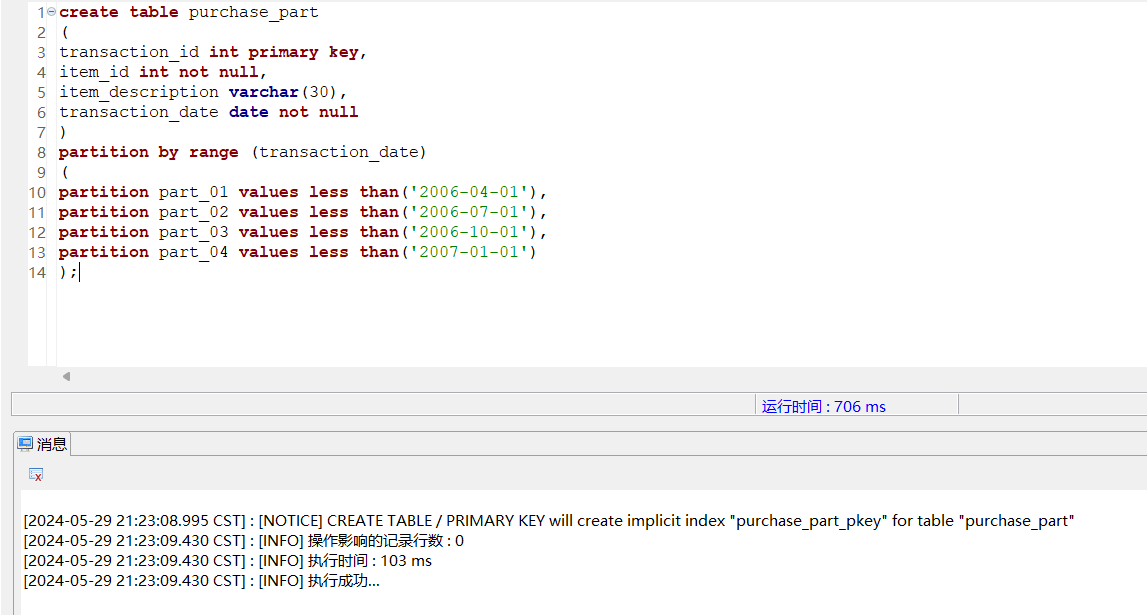
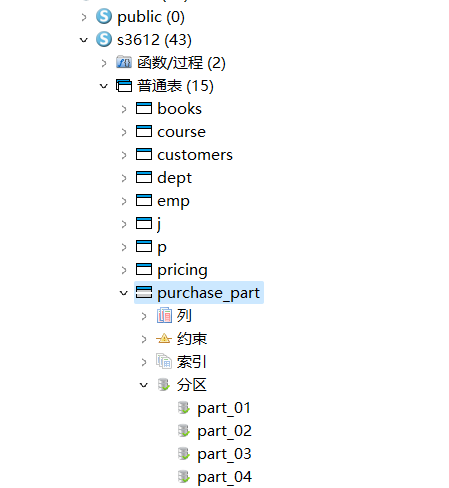
**实验六 数据库内模式**

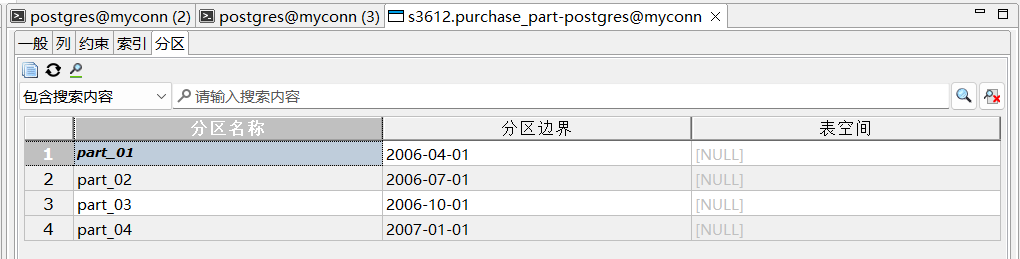
**37220222203612 洪伟鑫**

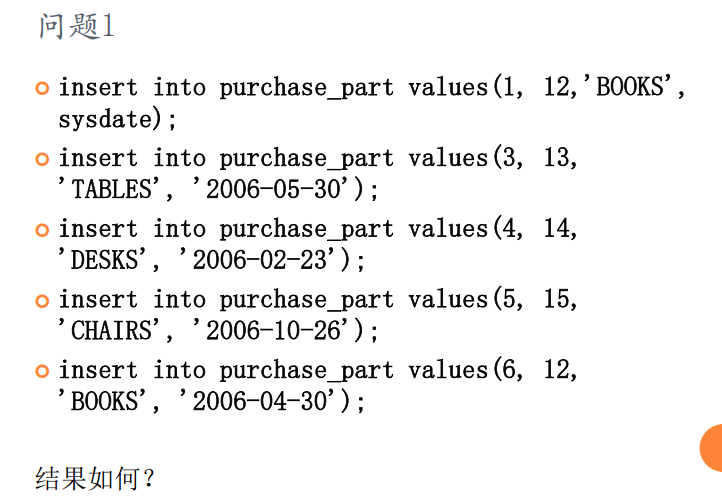


先创建表并进行范围分区



可以看到分区出现了part01到part04



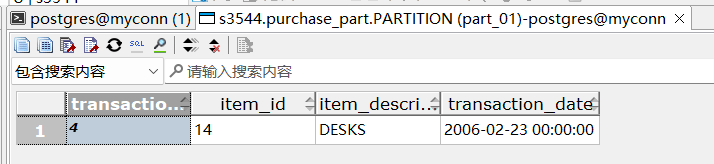


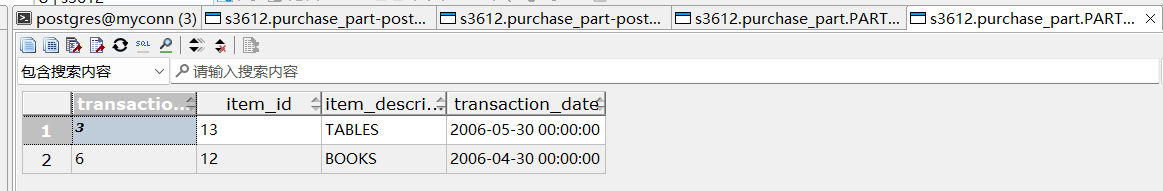
执行后，插入的各元组会根据分区的划分而存储在不同分区中

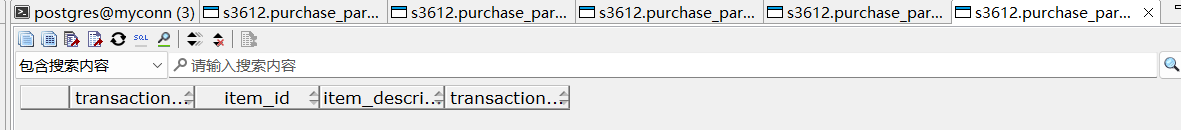
第一条插入语句因为sysdate无法映射到分区中的任何一个，所以会报错，插入失败

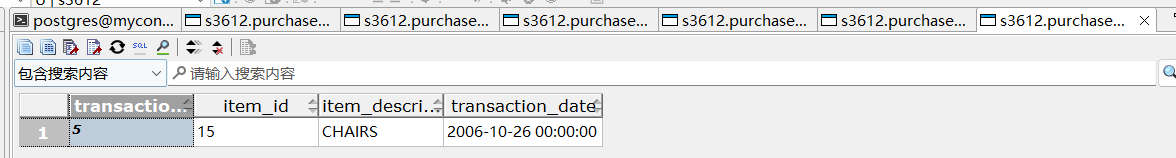
其他语句可以正常执行

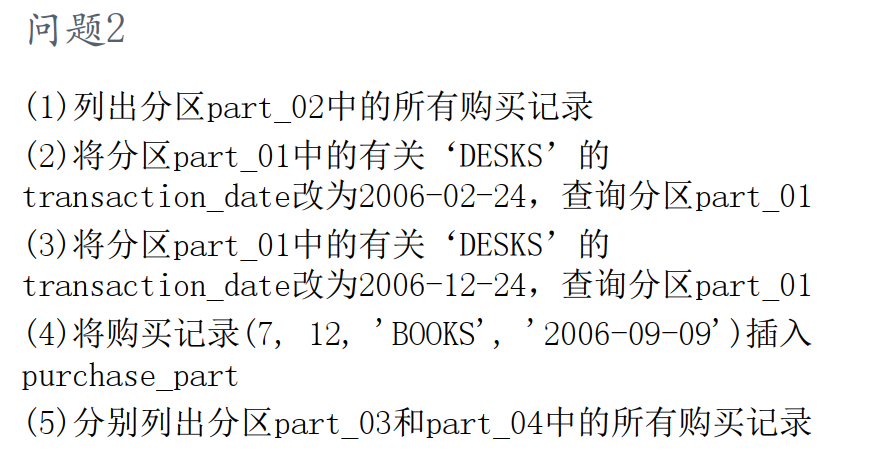
查看part1到part4的四部分分区数据如下







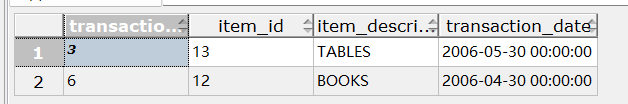




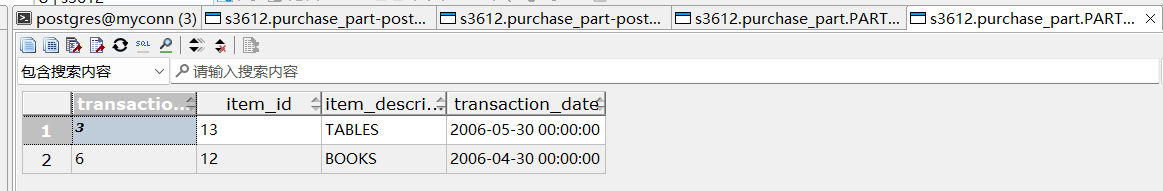
（1）

**SELECT** **\*** **FROM** purchase\_part

**WHERE** transaction\_date **>=** '2006-04-01' **AND** transaction\_date **<** '2006-07-01';



可以看到结果与分区part02中直接查看的一样



（2）

**UPDATE** purchase\_part

**SET** transaction\_date **=** '2006-02-24'

**WHERE** item\_description **=** 'DESKS'

**AND** transaction\_date **<** '2006-04-01';

**SELECT** **\*** **FROM** purchase\_part

**WHERE** transaction\_date **<** '2006-04-01';



（3）

**UPDATE** purchase\_part

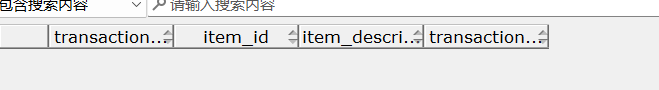
**SET** transaction\_date **=** '2006-12-24'

**WHERE** item\_description **=** 'DESKS'

**AND** transaction\_date **<** '2006-04-01';

**SELECT** **\*** **FROM** purchase\_part

**WHERE** transaction\_date **<** '2006-04-01';

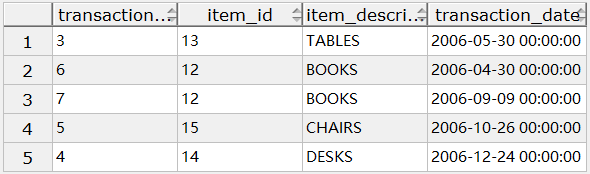


可以看到DESKS已经查询不到了，因为不在part01分区的范围内

（4）

**INSERT** **INTO** purchase\_part (transaction\_id, item\_id, item\_description, transaction\_date)

**VALUES** (7, 12, 'BOOKS', '2006-09-09');

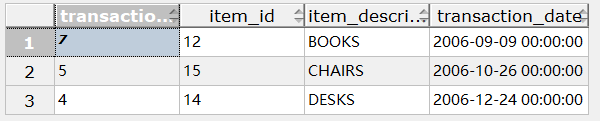


（5）

**SELECT** **\*** **FROM** purchase\_part

**WHERE** transaction\_date **>=** '2006-07-01' **AND** transaction\_date **<** '2007-01-01';

Part03和part04两个区域查询结果如下



**create** **table** user\_by\_city(

userID **int** **primary** **key**,

username **varchar**(25),

city **varchar**(10)

)

**partition** **by** list(city) (

**partition** p1 **values** ('北京', '上海', '天津') ,

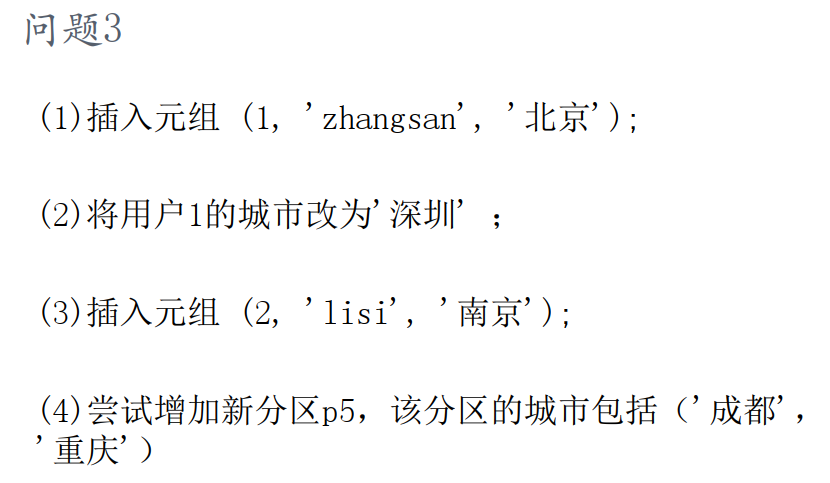
**partition** p2 **values** ('深圳', '广州') ,

**partition** p3 **values** ('郑州', '武汉', '西安') ,

**partition** p4 **values** ('大连', '沈阳')

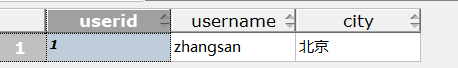
);

创建一个新表并按列表分区



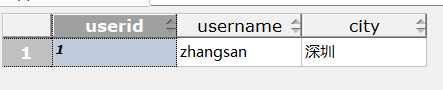
（1）

**INSERT** **INTO** user\_by\_city (userID, username, city) **VALUES** (1, 'zhangsan', '北京');



（2）

**UPDATE** user\_by\_city **SET** city **=** '深圳' **WHERE** userID **=** 1;



（3）

由于'南京'不在任何已定义分区的值列表中，尝试插入(2, 'lisi', '南京')会报错。

（4）

**ALTER** **TABLE** user\_by\_city

**ADD** **PARTITION** p5 **VALUES** ('成都', '重庆');



**create** **table** itemInfo

( item\_id **int**,

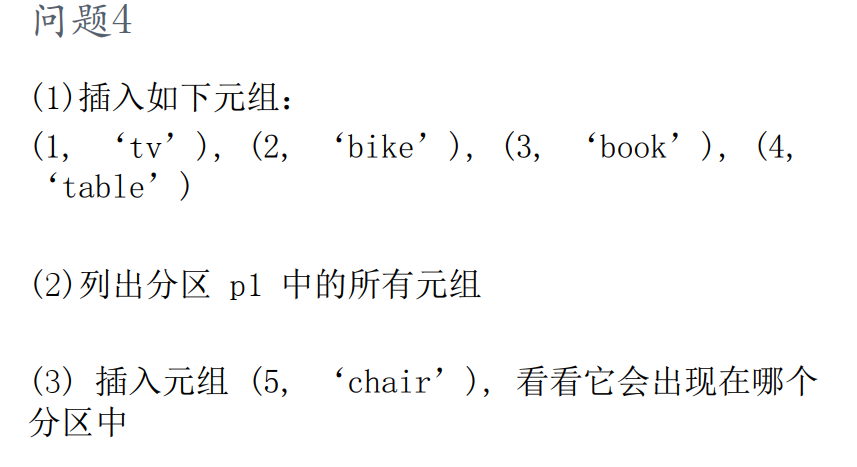
description **varchar**(30))

**partition** **by** hash(item\_id)

(**partition** p1,

**partition** p2);

创建一个新表并创建哈希分区



（1）

**INSERT** **INTO** itemInfo (item\_id, description) **VALUES**

(1, 'tv'),

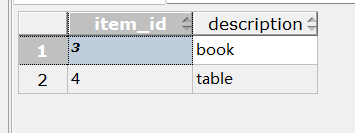
(2, 'bike'),

(3, 'book'),

(4, 'table');

（2）

**SELECT** **\*** **FROM** itemInfo **PARTITION** (p1);

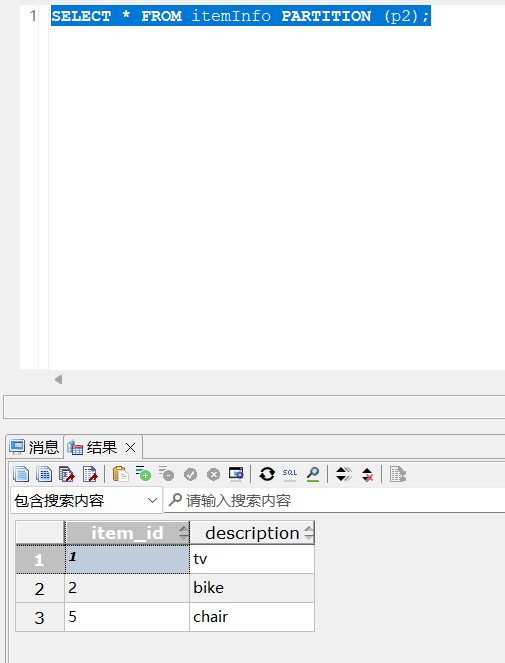


（3）

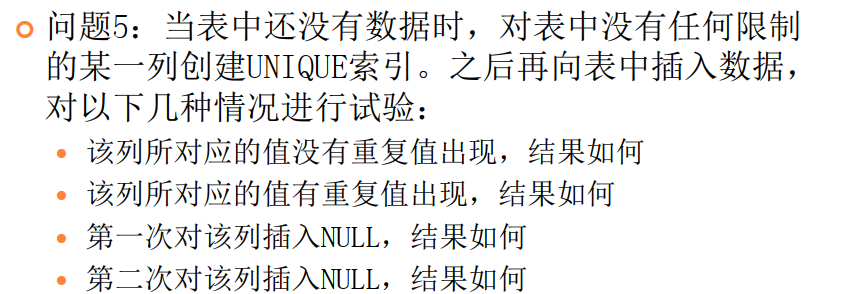
**INSERT** **INTO** itemInfo (item\_id, description) **VALUES** (5, 'chair');

**SELECT** **\*** **FROM** itemInfo **PARTITION** (p1);

**SELECT** **\*** **FROM** itemInfo **PARTITION** (p2);



查询后发现chair在p2分区



创建一个简单的表，分别用四列来模拟以下四种情况，每列都要创建unique索引

**CREATE** **TABLE** **table**1 (

col1 **INT** **UNIQUE**,

col2 **INT** **UNIQUE**,

col3 **INT** **UNIQUE**,

col4 **INT** **UNIQUE**

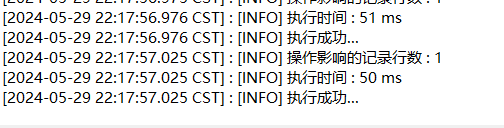
);

对于情况1，对列1插入两个不同的值，

**INSERT** **INTO** **table**1 (col1) **VALUES** (5);

**INSERT** **INTO** **table**1 (col1) **VALUES** (7);

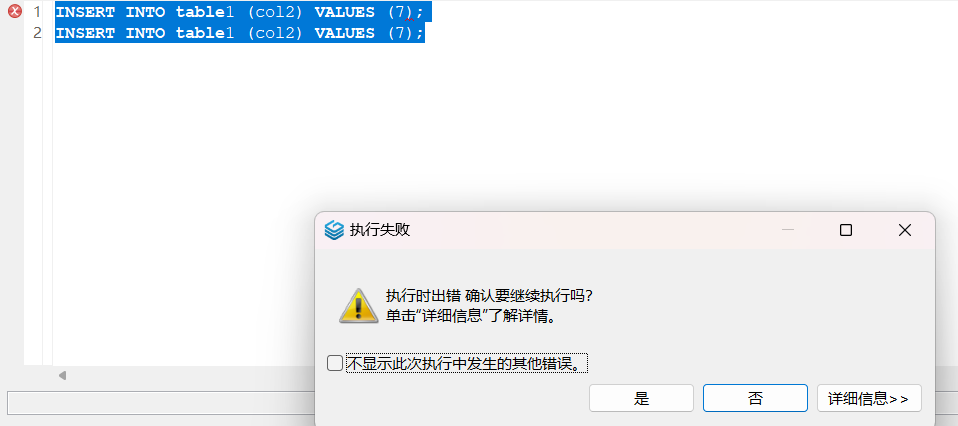
可以看到插入成功



对于情况2，对列2插入两个相同的值

**INSERT** **INTO** **table**1 (col2) **VALUES** (7);

**INSERT** **INTO** **table**1 (col2) **VALUES** (7);



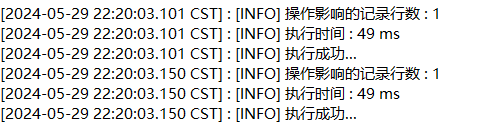
由于列2有unique约束，所以插入两个重复值是不允许的

对于情况3，对列3插入一个值和一个null值

**INSERT** **INTO** **table**1 (col3) **VALUES** (7);

**INSERT** **INTO** **table**1 (col3) **VALUES** (**null**);

操作成功，说明第一次对unique索引的列插入null值是可以的

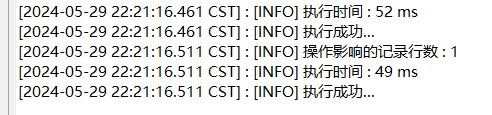


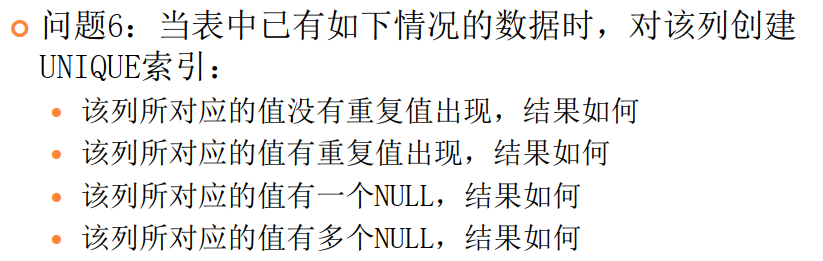
对于情况4，对列4插入两个null值

**INSERT** **INTO** **table**1 (col4) **VALUES** (**null**);

**INSERT** **INTO** **table**1 (col4) **VALUES** (**null**);

运行成功，说明有unique索引下的列也可以有多个null值





**CREATE** **TABLE** table2 (

col1 **INT** ,

col2 **INT** ,

col3 **INT** ,

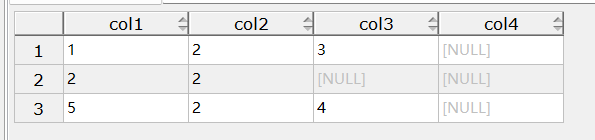
col4 **INT**

);

先创建一个不带列约束的表，然后往里面先准备好数据

**INSERT** **INTO** table2 (col1,col2,col3,col4) **VALUES** (1,2,3,**null**);

**INSERT** **INTO** table2 (col1,col2,col3,col4) **VALUES** (2,2,**null**,**null**);

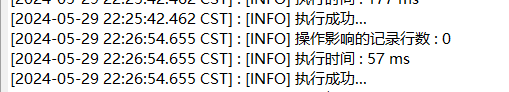


对应了上述四种情况，分别是没有重复值，有重复值，有一个null，有多个null

下面对四种情况的四列分别创建unique索引

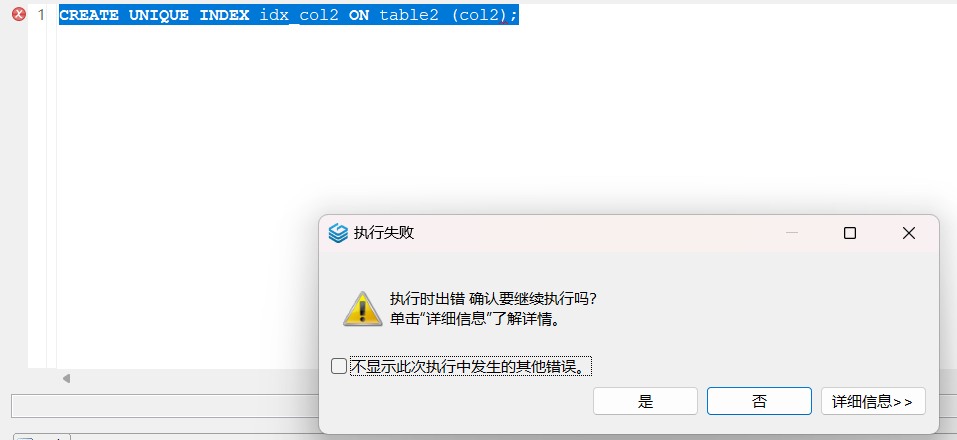
**CREATE** **UNIQUE** **INDEX** idx\_col1 **ON** table2 (col1);

对列1创建unique索引成功



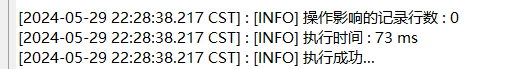
**CREATE** **UNIQUE** **INDEX** idx\_col2 **ON** table2 (col2);

对列2创建unique索引失败，因为列2中的数据不满足unique的列约束



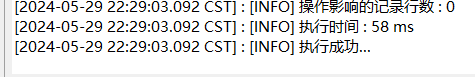
**CREATE** **UNIQUE** **INDEX** idx\_col3 **ON** table2 (col3);

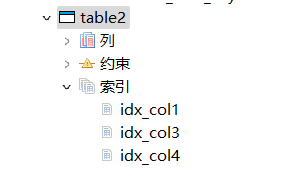
列3创建unique约束成功



**CREATE** **UNIQUE** **INDEX** idx\_col4 **ON** table2 (col4);

列4创建unique约束成功





最终仅能创建1、3、4列的unique索引，说明unique索引对null值是没有约束的，对重复值有约束