

一. 实验目的

- 学习用户自定义约束，并实践用户完整性，利用短语NOT NULL, UNIQUE, CHECK保证用户定义完整性

二. 实验准备

- 建立worker表，并自定义两个约束U1，U2，其中U1规定Name字段唯一，U2规定sage(级别) 字段的上限为28。并在worker表中插入一条合法记录。

```
CREATE TABLE Worker(  
    Number CHAR(5),  
    Name CHAR(8) CONSTRAINT U1 UNIQUE,  
    Sex CHAR(1),  
    Sage INT CONSTRAINT U2 CHECK (Sage<=28),  
    Department CHAR(20),  
    CONSTRAINT PK_Worker PRIMARY KEY (Number))  
  
INSERT INTO Worker (Number,Name,Sex,Sage,Department)  
VALUES('00001','李勇','M',14,'科技部')  
SELECT * FROM Worker
```

查询结果如下：

结果

消息

	Number	Name	Sex	Sage	Department
1	00001	李勇	M	14	科技部

查询已成功执行。


(local)\CHONOR (10.0 RTM) | DESKTOP-CHONOR\Chonor ... | School | 00:00:00 | 1 行

三. 实验结果

1. 加入约束U3，令sage值大于等于0


```
ALTER TABLE Worker ADD CONSTRAINT U3 CHECK (Sage>=0)
```

结果如下：

 消息

命令已成功完成。

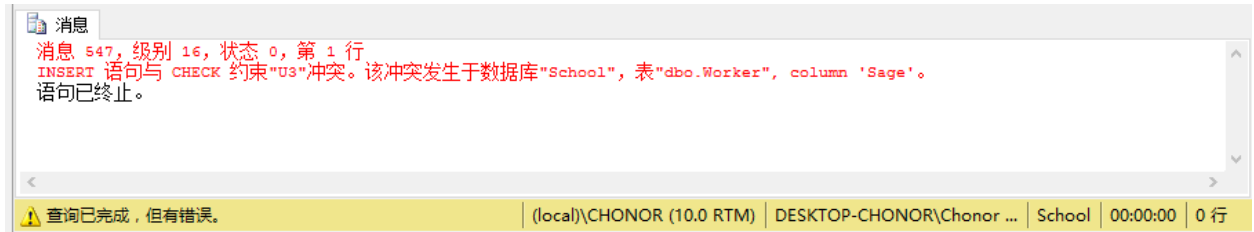
<						>
---	--	--	--	--	--	---

 查询已成功执行。	(local)\CHONOR (10.0 RTM)	DESKTOP-CHONOR\Chonor ...	School	00:00:00	0 行
--	---------------------------	---------------------------	--------	----------	-----

2. 演示插入违反约束U3的记录

```
INSERT INTO Worker (Number,Name,Sex,Sage,Department)
VALUES('00002','王勇','M',-1,'科技部')
```

查询结果如下：



此时的因为Sage<0，违反U3约束，所以数据插入失败。

3. 演示插入不违反约束U3的记录

```
INSERT INTO Worker (Number,Name,Sex,Sage,Department)
VALUES('00002','王勇','M',24,'科技部')
SELECT * FROM Worker
```

查询结果如下：

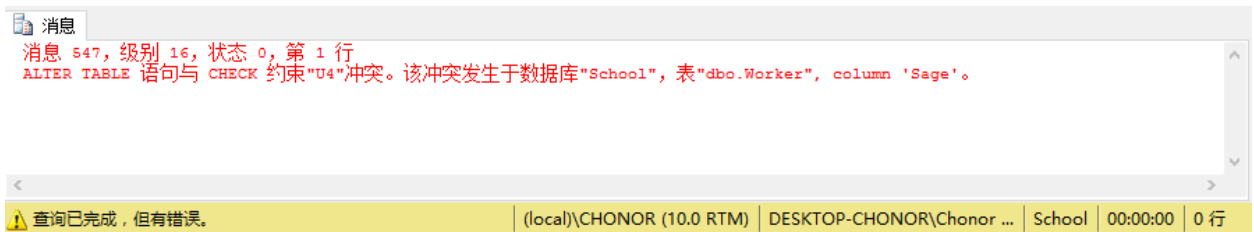
	Number	Name	Sex	Sage	Department
1	00001	李勇	M	14	科技部
2	00002	王勇	M	24	科技部

此时插入的Sage符合U3>=0的约束，同时也符合U2<=28的约束，所以成功插入。

4. 加入约束U4，令sage值<0，观察执行是否成功，分析原因

```
ALTER TABLE Worker ADD CONSTRAINT U4 CHECK (Sage<0)
```

此时结果如下：



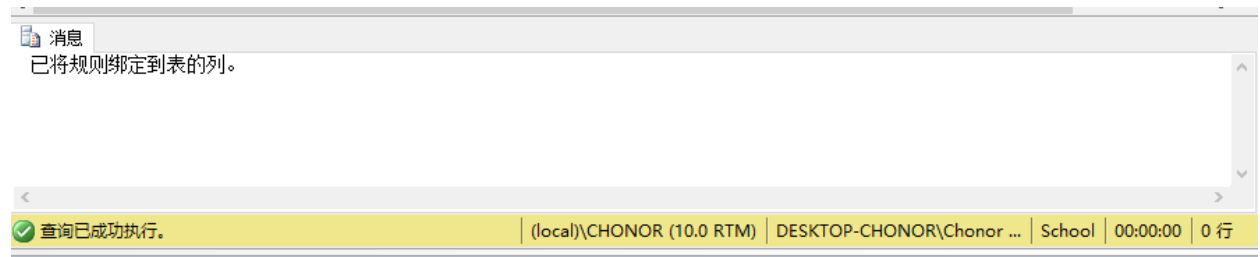
此时因为数据表中有Sage>=0的数据，所以此时加入Sage<0的约束会提示失败，如果要加入改约束那么就应先删除数据，再添加约束，做法如下：

```
DELETE FROM Worker
ALTER TABLE Worker ADD CONSTRAINT U4 CHECK (Sage<0)
```

5. 加入规则R2，确保插入的记录的sage值在1到100之间，并绑定到sage属性。

```
GO
CREATE RULE R2 AS @Sage BETWEEN 1 AND 100
GO
EXEC sp_bindrule R2, 'Worker.[Sage]';
```

结果如下：

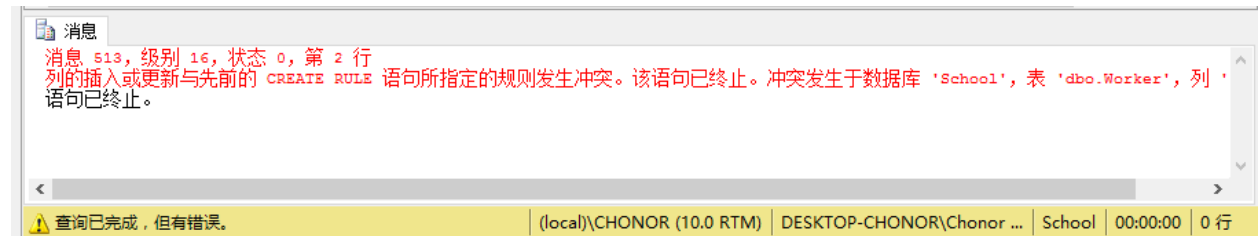


6. 演示插入违反R2的记录

尝试使用如下语句

```
INSERT INTO Worker (Number,Name,Sex,Sage,Department)
VALUES('00004','张勇','M',101,'科技部')
```

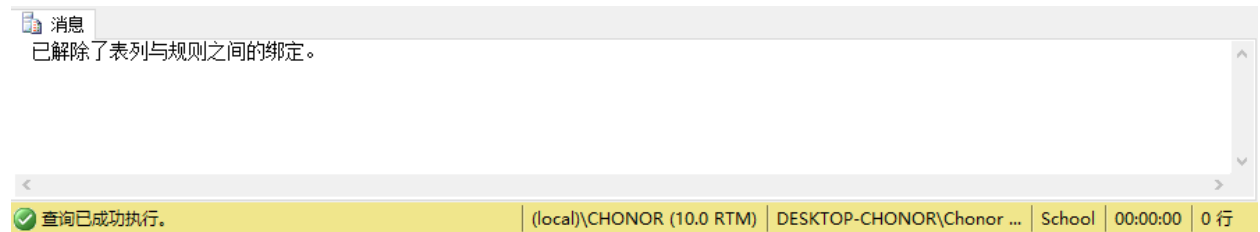
结果如下：



此时因为插入的Sage为101，不符合规则要求的[0,100]，所以报错违反规则不能插入

7. 解除规则R2的绑定，并重复(6)的操作。

```
EXEC sp_unbindrule 'Worker.[Sage]'
```



此时执行

```
INSERT INTO Worker (Number,Name,Sex,Sage,Department)
VALUES('00004','张勇','M',101,'科技部')
```

结果提示：

消息

消息 547, 级别 16, 状态 0, 第 1 行
INSERT 语句与 CHECK 约束"U2"冲突。该冲突发生于数据库"School", 表"dbo.Worker", column 'Sage'。
语句已终止。

查询已完成，但有错误。 | (local)\CHONOR (10.0 RTM) | DESKTOP-CHONOR\Chonor ... | School | 00:00:00 | 0 行

此时因为还存在之前的约束U2所以无法插入，此时删除U2，继续尝试

```
ALTER TABLE Worker DROP U2
```

```
INSERT INTO Worker (Number,Name,Sex,Sage,Department)  
VALUES('00004','张勇','M',101,'科技部')  
SELECT * FROM Worker
```

结果 消息

	Number	Name	Sex	Sage	Department
1	00001	李勇	M	14	科技部
2	00002	王勇	M	24	科技部
3	00004	张勇	M	101	科技部

查询已成功执行。 | (local)\CHONOR (10.0 RTM) | DESKTOP-CHONOR\Chonor ... | School | 00:00:00 | 3 行

此时结果成功插入。

8. 已知示例三中已插入sage为38的记录，那么加入规则R3，令sage大于50。观察加入规则R3的操作是否能成功。

```
GO  
CREATE RULE R3 AS @Sage>50  
GO  
EXEC sp_bindrule R3,'Worker.[Sage]'
```

消息

已将规则绑定到表的列。

查询已成功执行。 | (local)\CHONOR (10.0 RTM) | DESKTOP-CHONOR\Chonor ... | School | 00:00:00 | 0 行

此时提示成功绑定规则，测试插入违反规则数据

```
INSERT INTO Worker (Number,Name,Sex,Sage,Department)  
VALUES('00005','X勇','M',50,'科技部')
```

消息

消息 513, 级别 16, 状态 0, 第 2 行
列的插入或更新与先前的 CREATE RULE 语句所指定的规则发生冲突。该语句已终止。冲突发生于数据库 'School', 表 'dbo.Worker', 列 '
语句已终止。

查询已完成，但有错误。 | (local)\CHONOR (10.0 RTM) | DESKTOP-CHONOR\Chonor ... | School | 00:00:00 | 0 行

此时因为违反规则无法插入，但是我们查询表中信息

SELECT * FROM Worker

结果 消息

	Number	Name	Sex	Sage	Department
1	00001	李勇	M	14	科技部
2	00002	王勇	M	24	科技部
3	00004	张勇	M	101	科技部

查询已成功执行。

(local)\CHONOR (10.0 RTM) | DESKTOP-CHONOR\Chonor ... | School | 00:00:00 | 3 行

我们能发现表中的数据存在有不符合规则的，说明规则只会对之后数据的修改产生限制，对于之前的数据不会有有限制。

四. 实验感想

这次实验是用户完整性，这部分完整性是用户自行定义的，数据库系统只提供这些完整性的验证。实验中的约束和规则都是基于我们自己的定义实现，数据库系统只会对于我们的数据操作根据这些用户完整性进行返回，用户完整性可以规定我们某个值的取值范围或者类型要求。实验中实践了两种用户自定义完整性，约束和规则，约束对所有的数据产生限制，而规则只对之后的数据修改插入产生限制。