



知识点36: 表的简介



表

- 数据库中包含许多对象，其中最重要的就是表
- 数据库中的数据或信息都是存储在表中
- 表中数据的组织形式是行和列的组合，其中每一行代表一条记录
- 表必须建在某一数据库中，不能单独存在，也不以操作系统文件形式存在



表的分类

■ 按照存储时间分

- 永久表：除非人工删除，否则一直保存
- 临时表：只在数据库运行期间临时保存数据

■ 按照表的用途分类

- 系统表：维护服务器和数据库正常工作的数据表，不允许用户进行更改
- 用户表



临时表

- 临时表与永久表相似，但临时表存储在 **tempdb** 中，当不再使用时会自动删除。
- 可以创建本地临时表和全局临时表。
 - 本地临时表只对于创建者是可见的。当用户与 SQL Server 实例断开连接后，将删除本地临时表。
 - 全局临时表在创建后对任何用户和任何连接都是可见的，当引用该表的所有用户都与 SQL Server 实例断开连接后，将删除全局临时表。
 - 本地临时表的名称前面有一个数字符号 (**#table_name**)，而全局临时表的名称前面有两个数字符号 (**##table_name**)。
 - **CREATE TABLE #MyTempTable (cola INT PRIMARY KEY)**



临时表的作用

- 当对数据库执行大数据量的排序等操作时，要产生大量的中间运算结果，因此需要消耗大量的内存资源。
- 如果内存资源不够用，那么SQL Server将在临时数据库tempdb中创建临时表用于存放这些中间结果。



知识点37: 创建表



创建表

■ 说明:

- 表在数据库中被创建
- 表名和列名要有意义（不允许使用关键字）
- 不允许重名。
- 表名最多128个字符。
- 列名在一个表中必须是唯一的，最多128个字符。



创建表（续）

- 两种创建表的方式：
 - Management Studio中创建
 - 执行SQL语句
- 说明：
 - 创建表只能在**当前数据库**中进行
 - 当前数据库指的是可以操作的数据库
 - Use 数据库名
 - 默认数据库是系统数据库master

对象资源管理器

连接(O) ▾

- School_management
 - 数据库关系图
 - 表
 - 系统表
 - dbo.Student
 - 列
 - sno (int, not null)
 - sname (varchar(10), not null)
 - sex (varchar(2), null)
 - hometown (varchar(20), null)
 - introuction (varchar(400), null)
 - birthdate (datetime, not null)
 - 键
 - 约束
 - 触发器
 - 索引
 - 统计信息
 - 视图
 - 同义词
 - 可编程性
 - Service Broker
 - 存储
 - 安全性

THINK-THINK.Scho...ent - dbo.Student SQLQuery2.sql - TH...agement (sa (55))*

列名	数据类型	允许 Null 值
sno	int	<input type="checkbox"/>
sname	varchar(10)	<input type="checkbox"/>
sex	varchar(2)	<input checked="" type="checkbox"/>
hometown	varchar(20)	<input checked="" type="checkbox"/>
introuction	varchar(400)	<input checked="" type="checkbox"/>
birthdate	datetime	<input type="checkbox"/>

列属性

(常规)

(名称)	sname
长度	10
默认值或约束	

(常规)



数据类型

■ 字符串类型——单引号

■ 编码

- 普通字符编码——多种字符集，统称ANSI字符集
- 统一字符编码——Unicode字符集（固定使用两个字节来表示一个字符）

■ 非Unicode字符类型：

■ `char [(n)]`

- 固定长度，非 Unicode 字符数据，长度为 n 个字节。 n 的取值范围为 1 至 8,000，存储大小是 n 个字节。

■ `varchar [(n | max)]`

- 可变长度，非 Unicode 字符数据。 n 的取值范围为 1 至 8,000。

■ `text`



数据类型（续）

■ Unicode字符

■ `nchar [(n)]`

- n 个字符的固定长度的 Unicode 字符数据。 n 值必须在 1 到 4,000 之间（含）。

■ `nvarchar [(n | max)]`

- 可变长度 Unicode 字符数据。 n 值在 1 到 4,000 之间（含）。

■ 前面有一个N标识符

- 如N'dog'



数据类型（续）

- 精确数字

- bigint、decimal、int、numeric、smallint、money、tinyint、smallmoney、bit

- 近似数字

- float real

- 日期和时间

- Datetime、 smalldatetime
- 用单引号括起来
 - '20060507'



NULL的含义

■ NULL的含义

- NULL（空值）表示数值未知或将在以后添加的数据。
- 在“查询编辑器”中查看表数据时，空值在结果中显示为NULL。
- 用户也可显式的输入NULL
- NULL不同于空白、0或长度为零的字符串。



NULL的含义（续）

- 没有两个相等的空值。两个NULL值相比或NULL与其他数值相比，返回是未知，因为每个空值均为未知。
- NOT NULL（不允许空值）表示数据列不允许空值。
- 如果一列设为NOT NULL，那么在向表中写数据时必须在列中输入一个值，否则该行不被接收，有助于维护数据完整性。



创建表的SQL语句

CREATE TABLE <表名>

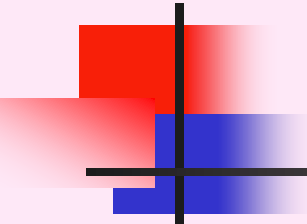
(
<列名> <数据类型> [列级完整性约束条件]
[, <列名> <数据类型> [列级完整性约束条件]]...
[, <表级完整性约束条件>]
)



SQL语句创建表——例题（续）

建立一个Student表，结构如下：

字段名称	字段类型	字段宽度	是否NULL	说明
sno	int		NOT NULL	学号
sname	Varchar	10	NOT NULL	姓名
sex	Varchar	2	NULL	性别
hometown	Datetime		NULL	籍贯
introduction	Varchar	400	NULL	简介
birthdate	Datetime		NULL	出生日期



```
create table Student  
(sno int not null,  
  sname varchar(10) not null,  
  sex varchar(2),  
  hometown varchar(20),  
  introuction varchar(50) null,  
  birthdate datetime not null )
```



SQL语句创建表——例题

创建一个表EmployeeLeave:

列	数据类型	检查
EmployeeID	int	NOT NULL
LeaveStartDate	date	NOT NULL
LeaveEndDate	date	NOT NULL
LeaveReason	varchar(100)	NOT NULL
LeaveType	char(2)	NOT NULL



SQL语句创建表——例题（续）

```
CREATE TABLE EmployeeLeave  
(  
    EmployeeID int NOT NULL,  
    LeaveStartDate datetime NOT NULL,  
    LeaveEndDate datetime NOT NULL,  
    LeaveReason varchar(100),  
    LeaveType char(2)NOT NULL  
)
```



知识点38：列属性说明与设置

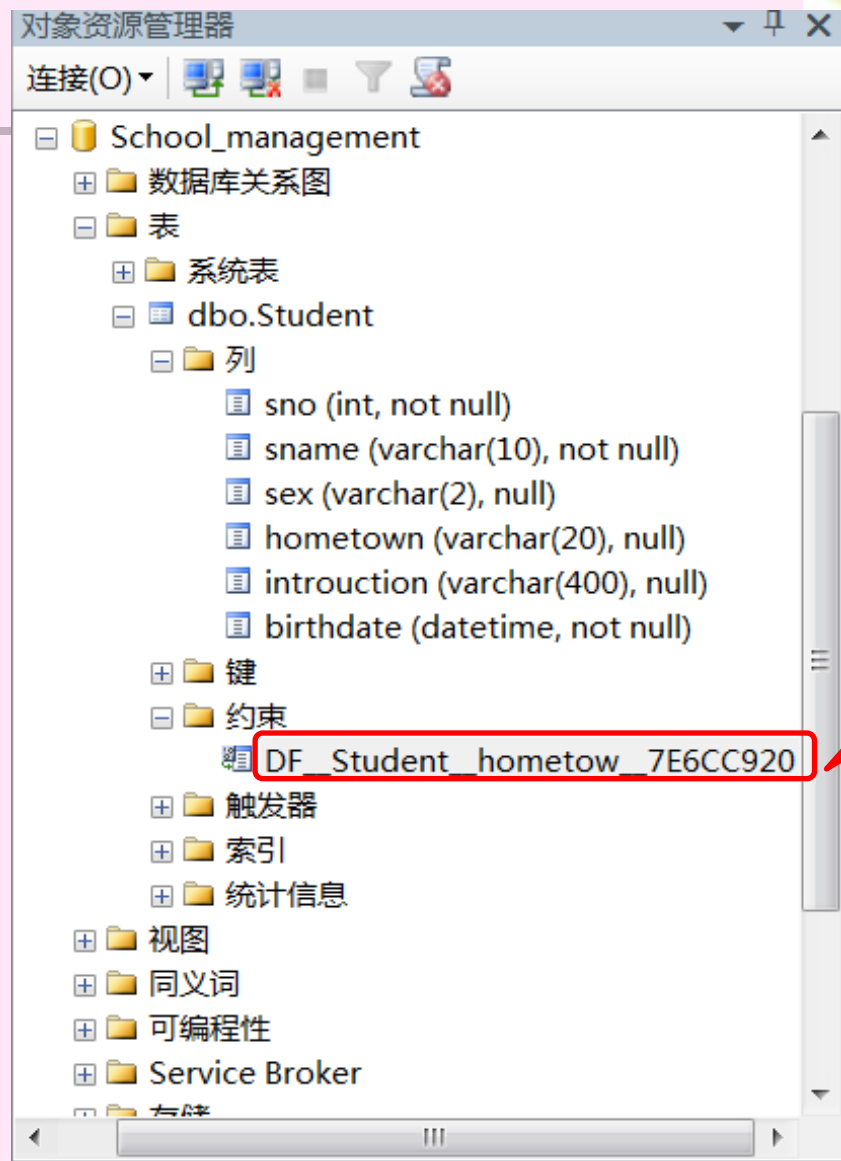


- The screenshot shows the 'Properties' window for the 'hometown' column in a SQL Server database. The 'Default Value or Binding' tab is selected, and the default value is set to '江苏' (Jiangsu). The column's data type is 'varchar' and it is marked as 'Allow Nulls'.



默认值

- 默认值，实际上是为表加了一个**DEFAULT**（默认值）约束。



默认约束名



默认值（续）

```
create table Student
(sno int not null,
sname varchar(10) not null,
sex varchar(2),
hometown varchar(20) null default('江苏'), /*设置默认值*/
introuction varchar(50) null,
birthdate datetime not null )
```



IDENTITY属性

- 使用**IDENTITY**关键字定义的字段又称标识字段。
- 为标识字段指定标识种子（**Identity Seed** 属性）和标识增量（**Identity Increment** 属性），系统按照给定的种子和增量自动生成标识号，该标识号是唯一的。

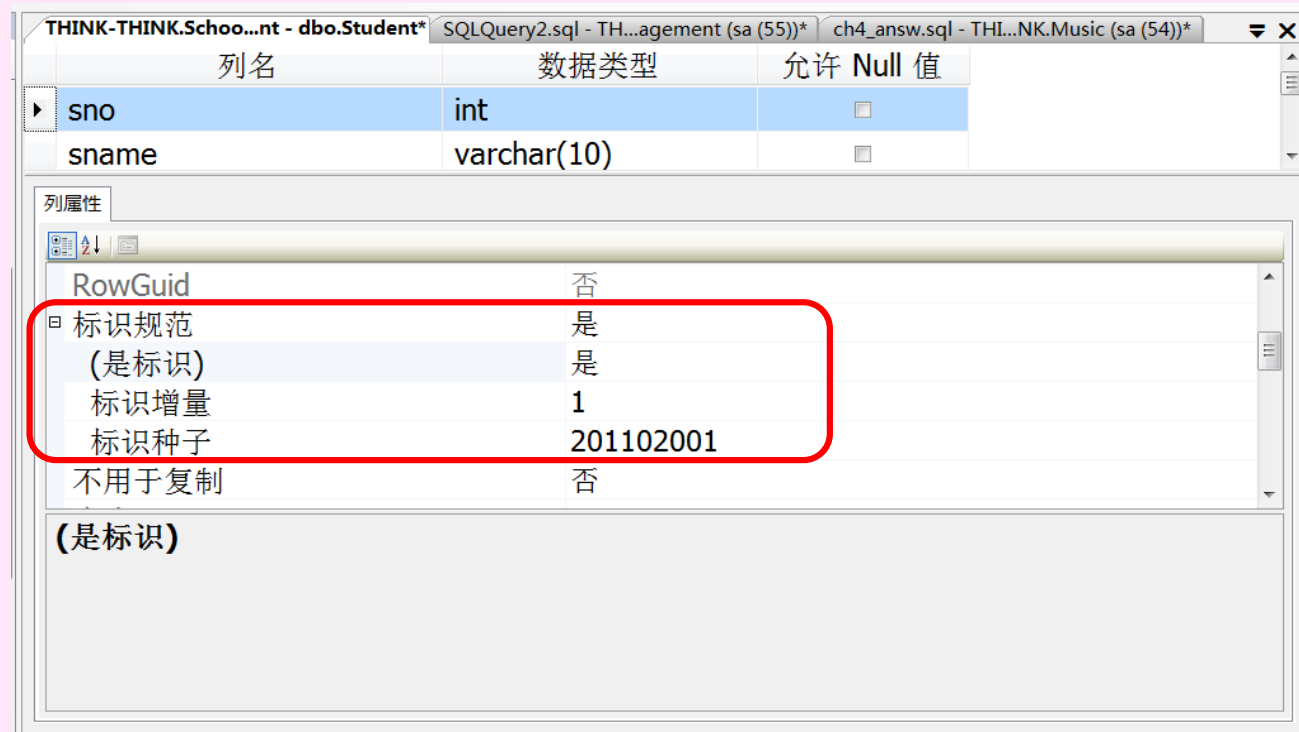


说明

- 使用**IDENTITY**属性时要注意：
 - 一个表只能有一个使用 **IDENTITY** 属性定义的列；
 - **IDENTITY**只适用于decimal（小数部分为0）、int、numeric（小数部分为0）、smallint、bigint 或 tinyint 数据类型；
 - 标识种子和增量的默认值均为 1；
 - 标识符列不能允许为空值，也不能包含 **DEFAULT** 定义或对象；
 - 标识符列不能更新；
 - 如果在经常进行删除操作的表中存在标识符列，那么标识值之间可能会出现断缺。



IDENTITY属性设置





IDENTITY属性设置 (续)

create table Student

(

sno int not null identity(201102001,1), /*设置IDENTITY*/

sname varchar(50) not null,

sex varchar(2),

hometown varchar(50) null default('江苏'), /*设置默认值*/

introduction varchar(50) null,

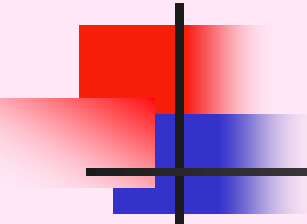
birthdate datetime not null)



练习

■ 在“Music”数据库中建立Songs（歌曲）表

字段名称	字段类型	字段宽度	说明	要求
SongID	Char	10	歌曲编号	非空
Name	Varchar	50	歌曲名字	
Lyricist	Varchar	20	词作者	
Composer	Varchar	20	曲作者	
Lang	Varchar	16	语言类别	默认“中文”



CREATE TABLE Songs
(SongID CHAR(10) not null,
Name VARCHAR(50),
Lyricist VARCHAR(20),
Composer VARCHAR(20),
Lang VARCHAR(16) default('中文'))



知识点39：数据完整性约束



数据完整性约束

- 数据库完整性（Database Integrity）是指数据库中数据的正确性和相容性
- 数据库完整性设计就是数据库完整性约束的设计
- SQL Server为了保证数据完整性共提供了以下6种约束：
 - 非空（NOT NULL）约束
 - 主键（PRIMARY KEY）约束
 - 外键（FOREIGN KEY）约束
 - 唯一性（UNIQUE）约束
 - 检查（CHECK）约束
 - 默认（DEFAULT）约束



数据完整性约束（续）

完整性类型	约束类型	完整性功能描述
实体完整性	PRIMARY KEY(主键)	指定主键，确保主键不重复，不允许主键为空值
	UNIQUE(惟一值)	指出数据应具有惟一值，防止出现冗余
用户定义完整性	CHECK(检查)	指定某个列可以接受值的范围，或指定数据应满足的条件
参照完整性	FOREIGN KEY(外键)	定义外键、被参照表和其主键



知识点39：实体完整性及其实现



PRIMARY KEY约束

- 表通常具有包含唯一标识表中每一行值的一列或一组列，这样的一列或多列称为表的主键（**PRIMARY KEY**）
- 表主键就是用来约束数据表中不能存在相同的两行数据。而且，**PRIMARY KEY** 约束的列应具有确定的数据，不能接受空值
- 在管理数据时，应确保每一个数据表都拥有一个唯一的主键，从而实现实体的完整性



说明

- 一个表只能有一个 **PRIMARY KEY** 约束，并且 **PRIMARY KEY** 约束中的列不能接受空值。
- 如果对多列定义了 **PRIMARY KEY** 约束，则一列中的值可能会重复，但来自 **PRIMARY KEY** 约束定义中所有列的任何值组合必须唯一。
- 当定义主键约束时，SQL Server在主键列上建立唯一聚簇索引，以加快查询速度。



单一主键约束的实现

字段名称	字段类型	字段宽度	说明	要求
SongID	Char	10	歌曲编号	非空
Name	Varchar	50	歌曲名字	
Lyricist	Varchar	20	词作者	
Composer	Varchar	20	曲作者	
Lang	Varchar	16	语言类别	默认“中文”



单一主键约束的实现（续）

THINK-THINK.Music - dbo.Songs SQLQuery2.sql - THI...NK.Mytest (sa (52))

列名	数据类型	允许 Null 值
SongID	char(10)	<input type="checkbox"/>
	varchar(50)	<input checked="" type="checkbox"/>
	varchar(20)	<input checked="" type="checkbox"/>
	varchar(20)	<input checked="" type="checkbox"/>
	varchar(16)	<input checked="" type="checkbox"/>

设置主键(Y)
插入列(M)
删除列(N)
关系(H)...
索引/键(I)...
全文索引(F)...
XML 索引(X)...
CHECK 约束(O)...
空间索引(P)...
生成更改脚本(S)...

对象资源管理器

连接(O) ▾

- dbo.Singer
 - dbo.Songs
 - SongID (PK, char(10), not null)
 - Name (varchar(50), null)
 - Lyricist (varchar(20), null)
 - Composer (varchar(20), null)
 - Lang (varchar(16), null)
 - 键
 - PK_Songs
 - 约束
 - 触发器
 - 索引
 - 统计信息
- dbo.Track
- 视图
- 同义词
- 可编程性
- Service Broker
- 存储
- 安全性
- mydb
- Mytest

主键约束名

THINK-THINK.Music - dbo.Songs SQLQuery2.sql - THI...NK.Mytest (sa (52))

列名	数据类型	允许 Null 值
SongID	char(10)	<input type="checkbox"/>
Name	varchar(50)	<input checked="" type="checkbox"/>
Lyricist	varchar(20)	<input checked="" type="checkbox"/>
Composer	varchar(20)	<input checked="" type="checkbox"/>
Lang	varchar(16)	<input checked="" type="checkbox"/>

列属性

长度

SongID

10

(常规)



单一主键约束的实现 (续)

CREATE TABLE Songs

(SongID CHAR(10) not null PRIMARY KEY, /*主键约束*/,

Name VARCHAR(50),

Lyricist VARCHAR(20),

Composer VARCHAR(20),

Lang VARCHAR(16) default('中文')) /*DEFAULT约束 */



组合主键的设置

字段名称	字段类型	字段宽度	说明	要求
SongID	Char	10	歌曲编号	非空
SingerID	Char	10	歌手编号	非空
Album	Varchar	50	专辑	
Style	Varchar	20	曲风类别	默认“流行”
Circulation	INT		发行量	
PubYear	INT		发行时间	



组合主键的设置 (续)

列名	数据类型	允许 Null 值
SongID	char(10)	<input type="checkbox"/>
SingerID	char(10)	<input type="checkbox"/>
Album	varchar(50)	<input checked="" type="checkbox"/>
Style	varchar(20)	<input checked="" type="checkbox"/>
Circulation	int	<input checked="" type="checkbox"/>
PubYear	int	<input checked="" type="checkbox"/>

设置主键(Y)

插入列(M)

删除列(N)

关系(H)...

索引/键(I)...

全文索引(F)...

XML 索引(X)...

CHECK 约束(O)...

空间索引(P)...

生成更改脚本(S)...

对象资源管理器

dbo.Track

SongID (PK, char(10), not null)

SingerID (PK, char(10), not null)

Album (varchar(50), null)

Style (varchar(20), null)

Circulation (int, null)

PubYear (int, null)

键

PK Track

THINK-THINK.Music - dbo.Track

THINK-THINK.Music - dbo.Songs

SQLQuery2.sql - TH...K.Mytest (sa (52))*

列名	数据类型	允许 Null 值
SongID	char(10)	<input type="checkbox"/>
SingerID	char(10)	<input type="checkbox"/>
Album	varchar(50)	<input checked="" type="checkbox"/>
Style	varchar(20)	<input checked="" type="checkbox"/>
Circulation	int	<input checked="" type="checkbox"/>
PubYear	int	<input checked="" type="checkbox"/>

列属性

(常规)

(名称)

默认值或绑定

数据类型

PubYear

int

(常规)



组合主键的设置 (续)

```
CREATE TABLE Track
(SongID CHAR(10) not null,
SingerID CHAR(10) not null,
Album VARCHAR(50),
Style VARCHAR(20) default('流行'),          /*DEFAULT约束*/
Circulation INTEGER, PubYear INTEGER,
PRIMARY KEY(SongID, SingerID) /*主键约束, 该约束是表级完
整性约束*/
)
```



UNIQUE约束

- 唯一约束被用来增强非主键列的唯一性。
- 设置了唯一约束的列不能有重复值，可以但最多允许一个NULL值。
- 每个UNIQUE约束都生成一个唯一索引，所以唯一约束和唯一索引的创建方法相同



说明

- 主键约束与唯一约束的异同：
 - 两者都要求约束的列不能有重复值；
 - 主键约束要求主键列不能为空；
 - 唯一约束允许有空值，但只允许一个NULL值。
- 在一个表上可以定义多个UNIQUE约束；
- 可以在多个列上定义一个UNIQUE约束，表示这些列组合起来不能有重复值。



设置唯一约束

- 例：为Singer中的Name设置唯一约束

THINK-THINK.Music - dbo.Singer*

列名	数据类型	允许 Null 值
SingerID	char(10)	<input type="checkbox"/>
Name	varchar(50)	<input type="checkbox"/>
	varchar(2)	<input checked="" type="checkbox"/>
	datetime	<input checked="" type="checkbox"/>
	varchar(20)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

索引/键(I)...

- 设置主键(Y)
- 插入列(M)
- 删除列(N)
- 关系(H)...
- 索引/键(I)...
- 全文索引(F)...
- XML 索引(X)...
- CHECK 约束(O)...
- 空间索引(P)...
- 生成更改脚本(S)...

索引/键

选定的 主/唯一键或索引(S):

IX_Singer*
PK_Singer

正在编辑新的 唯一键或索引 的属性。

(常规)

类型	索引
列	SingerID (ASC)
是唯一的	否

标识

(名称)	IX_Singer
说明	

表设计器

包含的列	
创建为聚集的	否
忽略重复键	否

数据空间规范

	PRIMARY
--	---------

添加(A) 删除(D) 关闭(C)



设置唯一约束（续）

索引列

指定用于此索引的列和排序顺序(C):

列名	排序顺序
Name	升序

确定 取消

索引/键

选定的 主/唯一键或索引(S):

IX_Singer
PK_Singer

正在编辑现有 主/唯一键或索引 的属性。

(常规)	
类型	索引
列	Name (ASC)
是唯一的	是
标识 (名称)	是
说明	否
表设计器	
包含的列	
创建为聚集的	否
忽略重复键	否
数据空间规范	PRIMARY

添加(A) 删除(D) 关闭(C)

对象资源管理器

连接(O)

- 表
 - 系统表
 - dbo.Singer
 - 列
 - SingerID (PK, char(10), not null)
 - Name (varchar(50), not null)
 - Gender (varchar(2), null)
 - Birth (datetime, null)
 - Nation (varchar(20), null)
 - 键
 - 约束
 - 触发器
 - 索引
 - IX_Singer (唯一, 非聚集)
 - PK_Singer (聚集)
 - 统计信息
 - dbo.Songs
 - dbo.Track
- 视图
- 同义词
- 可编程性
- Service Broker
- 存储



设置唯一约束 (续)

```
CREATE TABLE Singer(
```

```
-- 主键约束
```

```
SingerID char(10) NOT NULL PRIMARY KEY,
```

```
-- 唯一约束
```

```
Name varchar(50) NOT NULL UNIQUE,
```

```
Gender varchar(2) NULL, Birth datetime NULL,
```

```
Nation varchar(20) NULL)
```



知识点40：用户定义完整性及其实现



用户定义完整性

- 用户定义的完整性是通过检查约束来实现的。
- 检查约束用于限制列的取值在指定范围内，这样使得数据库中存放的值都是有意义的。
- 当一列被设置了**CHECK**约束，该列的取值必须符合检查约束所设置的条件。
- 约束条件应是逻辑表达式，多个条件可以用**AND**或**OR**组合。



检查约束

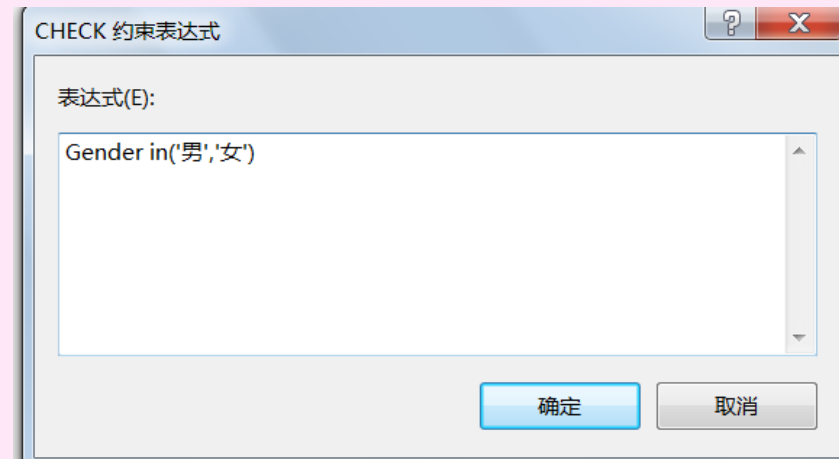
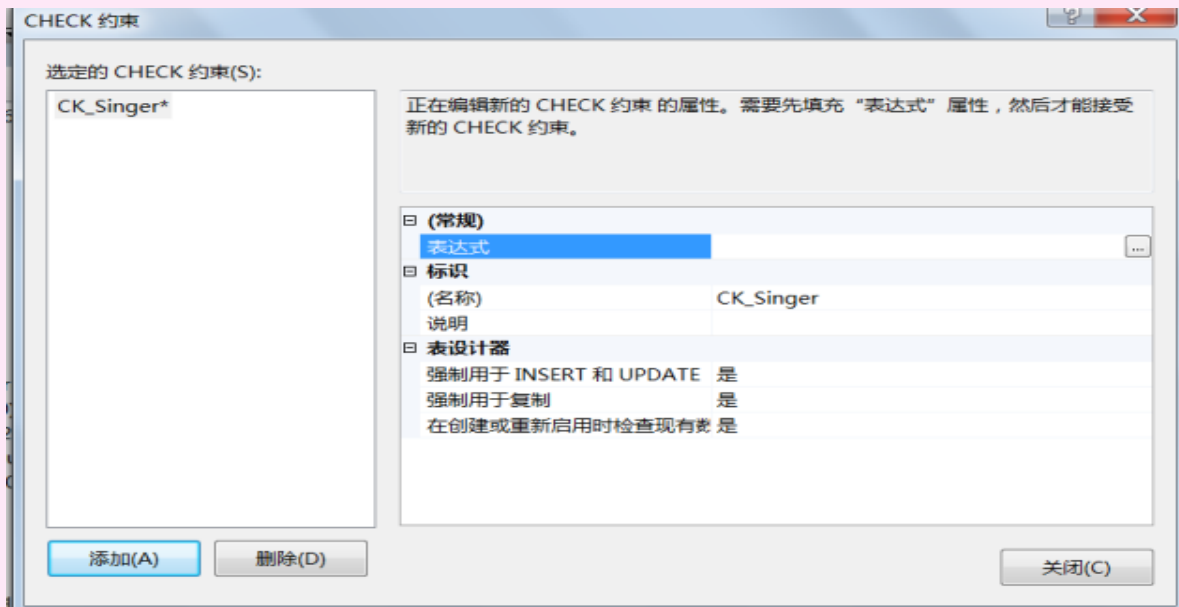
Singer表

字段名称	字段类型	字段宽度	是否NULL	说明
SingerID	Char	10	NOT NULL	歌手编号
Name	Varchar	8	NOT NULL	歌手名称
Gender	Varchar	2	NULL	性别
Birth	Datetime		NULL	出生日期
Nation	Varchar	20	NULL	籍贯



设置检查约束

- 设置Singer表中Gender（性别）列取值为“男”或“女”





设置检查约束 (续)

```
CREATE TABLE Singer(  
  SingerID char(10) NOT NULL PRIMARY KEY,      /* 主键约束 */  
  Name varchar(50) NOT NULL  
  UNIQUE,      /* 唯一约束 */  
  Gender varchar(2) NULL CHECK(Gender in('男','女')), /* 检查约束 */  
  Birth datetime NULL, Nation varchar(20) NULL)
```



知识点41：参照完整性及其实现



参照完整性

- 参照完整性主要通过主键与外键的联系来实现的。
- 主键所在的表称为主表，外键所在的表称为子表。
- 外键的取值参照主键的取值，即外键列的值有两种可能：一是等于主键的某个值；二是为空值，否则将返回违反外键约束的错误信息。



外键约束

- 通过设置参照完整性，SQL Server将禁止用户在外键列中引用主键中不存在的值；
- 在默认级联规则下，禁止修改主键中的值而不修改外键中的值；禁止删除被外键引用的主键值。
- SQL Server通过FOREIGN KEY（外键）约束来实现参照完整性
- 一张表可含有多个FOREIGN KEY约束。



说明

- **FOREIGN KEY**约束只能引用同一个服务器上的同一数据库中的表。跨数据库的参照完整性必须通过触发器实现。
- **FOREIGN KEY**可引用同一个表中的其他列，这称为自引用。
- **FOREIGN KEY** 约束并不仅仅可以与另一表的 **PRIMARY KEY** 约束相链接，它还可以定义为引用另一表的 **UNIQUE** 约束。
- 不能更改定义了**FOREIGN KEY**约束的列的长度，因为外键列和主键列的数据类型和长度需一致。

SingerID	Name	Gender	Birth	Nation
GC001	王菲	女	1969-07-01	中国
GA001	Michael_Jackson	男	1958-06-11	美国
GC002	李健	男	1974-10-02	中国
GC003	李小东	男	1970-08-12	中国
GA002	John_Denver	男	1943-02-16	美国

SongID	SingerID	Album	Style	Circulation	PubYear
S0102	GA001	SPIRIT	摇滚	20	1983
S0101	GA002	MEMORY	乡村	10	1971
S0001	GC001	流金岁月	流行	12	2003
S0001	GC002	想念你	流行	8	2010
S0002	GA001	GOD BLESS	爵士	NULL	1975



设置外键约束

- 设置“Track”表中的“SongID”为外键

外键关系

选定的 关系(S):

FK_Track_Track*

正在编辑新的 关系 的属性。需要先填充“表和列规范”属性，然后才能接受新的关系。

▢ (常规)

▢ 表和列规范

在创建或重新启用时检查现有表 是

▢ 标识

(名称) FK_Track_Track

说明

▢ 表设计器

▢ INSERT 和 UPDATE 规范

强制外键约束 是

强制用于复制 是

添加(A) 删除(D) 关闭(C)

表和列

关系名(N):

FK_Track_Singer

主键表(P):

Singer

外键表:

Track

SingerID

SongID

<无>

确定 取消



设置外键约束 (续)

```
CREATE TABLE Track
```

```
(/*外键约束*/
```

```
SongID CHAR(10) not null FOREIGN KEY REFERENCES Songs(SongID),
```

```
SingerID CHAR(10) not null
```

```
FOREIGN KEY REFERENCES Singers(SingerID),
```

```
Album VARCHAR(50),
```

```
Style VARCHAR(20) default('流行'),          /*DEFAULT约束*/
```

```
Circulation INTEGER, PubYear INTEGER,
```

```
PRIMARY KEY(SongID, SingerID) )
```



例题

CREATE TABLE 员工信息

(员工编号 char(6) PRIMARY KEY,

员工姓名 char(10) Not Null, 出生日期 SmallDateTime,

性别 char(2) DEFAULT '男',

地址 varchar(40), 邮政编码 char(6), 电话号码 char(11), 部门编号char(3),

UNIQUE(员工姓名),

CHECK(性别 IN('男', '女')),

CHECK(邮政编码LIKE '[0-9][0-9][0-9][0-9][0-9][0-9]'),

CHECK (电话号码LIKE '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9] [0-9][0-9][0-9] '),

FOREIGN KEY (部门编号) REFERENCE 部门信息(部门编号))

练习

NewProduct	
属性名	数据类型
ProductId	char(6)
ProductName	varchar(20)
ProductDescription	varchar(250)
CategoryId	char(3)
Photo	image
Qoh	smallint
ProductImgPath	varchar(50)

Category	
属性名	数据类型
CategoryId	char(3)
Category	char(20)
Description	varchar(100)

- 创建Category表。当创建表的时候，增强下面的数据完整性规则
 1. category id应该是主键。
 2. Category属性应该是唯一的但不是主键。
 3. Category表的描述可以允许存储NULL值。

• 创建带有下面数据完整性规则的NewProduct表：

- a. product id应该是主键。
- b. Photo 和 ProductImgPath属性可以允许存储NULL值。
- c. ProductName 和 ProductDescription属性不应该允许NULL值。
- d. CategoryId属性的值应该在Category表中表示。
- e. Qoh 大于100



知识点42：修改表



修改表

ALTER TABLE table_name

{ ALTER COLUMN column_name

--修改列定义

→ [WITH { CHECK → NOCHECK }]

--是否检查现有数据

选项

→ADD COLUMN <column_definition>

--添加新列

→ADD [CONSTRAINT constraint_name] constraint_type

--添加约束

→DROP [CONSTRAINT] constraint_name }

--删除约束



例题

- 例：建立歌曲表Songs增加一列“翻唱歌曲编号”（DupSongID），其数据类型是字符型。

ALTER TABLE Songs ADD DupSongID CHAR(10)

- 说明：不论表中原来是否有数据，新增加的一列值全部为空（NULL）。



例题（续）

■ 删除列

- `ALTER TABLE doc_exb DROP COLUMN column_b ;`

■ 更改列的数据类型

- `ALTER TABLE doc_exy ALTER COLUMN column_a DECIMAL (5, 2) ;`

■ 添加约束

- `alter table stu_test2 add unique(sname)`

■ 添加默认约束

- `ALTER TABLE stu_test2 ADD DEFAULT '江苏' FOR hometown`





例题（续）

- 为Student添加约束，要求：主键是sno；sex只能“男”或“女”；sname值不重复。

```
alter table dbo.Student
```

```
add constraint pk_sno primary key(sno),
```

--主键约束

```
constraint ck_sex check(sex in('男','女')),
```

--检查约束

```
constraint unique_sname unique(sname)
```

- 删除unique_sname约束。

```
alter table dbo.Student drop unique_sname
```



说明

- 在删除某列前要先解除与该列相关联的约束等关系。
- 删除列使用ALTER TABLE的 **DROP COLUMN**子句，其中 **COLUMN**不能省略，因为如果省略，则成了ALTER TABLE **DROP**，这是删除约束的语句。
- 删除**PRIMARY KEY**约束时，如果另一个表中的**FOREIGN KEY**约束引用了该**PRIMARY KEY**约束，则必须先删除**FOREIGN KEY**约束。



WITH NOCHECK的说明

- 表创建以后添加约束的时候，默认是with check选项
 - 这种情况下，会检查已有数据，如果现有数据违反了拟添加的约束，则数据库引擎将返回错误信息，并且不添加约束。
- 如果不想根据现有数据验证新的约束，可以使用 **WITH NOCHECK**选项



例题

- 在现有列中添加一个未经验证的 CHECK 约束

alter table dbo.student

with nocheck

add constraint ck_dept check(dept in('computer','math'))



知识点43: 删除表



删除表

- 删除表的操作将删除关于该表的所有定义（包括表的结构、索引、触发器、约束等）和表的数据。
- 如果要删除主表，必须先删除子表或先在子表中删除对应的 **FOREIGN KEY** 约束
- **SQL Server** 中，删除表后，建立在该表上的视图定义仍然保留在数据字典中，但是当用户引用时出错。
 - 任何引用已删除表的视图或存储过程都必须使用 **DROP VIEW** 或 **DROP PROCEDURE** 显式删除。。



删除表（续）

- 用T-SQL的**DROP TABLE**语句删除表
 - **DROP TABLE mytest**
- 如果要删除表中的所有数据而不删除表本身，即清空表。可使用**TRUNCATE TABLE** 语句。
 - 语法为：**TRUNCATE TABLE** 表名