Bài thực hành 1: Làm quen với SQL Sever và các thao tác trên bảng

I/ LÝ THUYẾT:

1/ Ràng buộc toàn vẹn:

+ Ràng buộc toàn vẹn là những điều kiện bất biến mà tất cả các bộ của những quan hệ có liên quan trong CSDL đều phải thoả mãn ở mọi thời điểm

Ví dụ:

- Ngày sinh của sinh viên phải nhỏ hơn ngày nhập học
- Điểm của sinh viên phải từ 0 đến 10 là một qui định
- Giới tính của sinh viên chỉ có thể là nam hoặc nữ.
- Sinh viên đăng ký học với những môn học thuộc khoa mà sinh viên đó học

+ Phân loại ràng buộc toàn vẹn: Ràng buộc toàn vẹn được chia làm 3 loại:

- Toàn vẹn thực thể: RB thực thể đảm bảo toàn vẹn cho các thực thể được mô hình bởi hệ thống. Ở mức độ đơn giản nhất, sự tồn tại của khóa chính là một RB thực thể nhằm đảm bảo qui tắc 'mỗi thực thể phải là duy nhất'.
- Toàn vẹn miền: là tập các qui tắc xác định giá trị hợp lệ của thuộc tính.
- Toàn vẹn tham chiếu: Khi mô hình quan hệ thì cần thiết phải chia các quan hệ để giảm thiểu sự dư thừa và phải thiết lập các khóa ngoại để liên kết các quan hệ. Nếu các liên kết này bị phá vỡ, trường hợp tốt nhất là hệ thống sẽ không còn đáng tin cậy, xấu nhất là không thể sử dụng được nữa. RBTV tham chiếu đảm bảo cho những liên kết này. Nói cách khác, RB tham chiếu đảm bảo cho các

- SQL Server cung cấp một lượng lớn các công cụ để thực thi toàn vẹn dữ liệu. Chúng được liệt kê trong bảng sau:

Kiểu toàn vẹn	Các công cụ SQL Server
Toàn vẹn thực thể	1. Ràng buộc PRIMARY KEY 2. Ràng buộc UNIQUE 3. Ràng buộc IDENTITY
Toàn vẹn miền	1. Ràng buộc DEFAULT 2. Ràng buộc CHECK 3. Ràng buộc NOT NULL 4. Rule
Toàn vẹn tham chiếu	1. Ràng buộc FOREIGN KEY 2. Ràng buộc CHECK

2/ Tổng quan về SQL Server:

+ Hệ quản trị CSDL: Hệ quản trị cơ sở dữ liệu (Database Management System - DBMS), là phần mềm hay hệ thống được thiết kế để quản trị một cơ sở dữ liệu. Cụ thể, các chương trình

thuộc loại này hỗ trợ khả năng lưu trữ, sửa chữa, xóa và tìm kiếm thông tin trong một cơ sở dữ liệu (CSDL).

Các hệ quản trị CSDL phổ biến được nhiều người biết đến là MySQL, Oracle, PostgreSQL, SQL Server, DB2, Infomix...

+ SQL Server là hệ thống quản trị cơ sở dữ liệu quan hệ (Relational DataBase Management System- RDBMS) sử dụng các lệnh chuyển Transaction-SQL để trao đổi dữ liệu giữa Client Computer và Server Computer.

+ SQL Server có một số đặc tính sau:

- Cho phép quản trị một hệ CSDL lớn (lên đến vài tega byte), có tốc độ xử lý dữ liệu nhanh đáp ứng yêu cầu về thời gian.
- Cho phép nhiều người cùng khai thác trong một thời điểm đối với một CSDL và toàn bộ quản trị CSDL (lên đến vài chục ngàn user).
- Có hệ thống phân quyền bảo mật tương thích với hệ thống bảo mật của công nghệ NT (Network Technology), tích hợp với hệ thống bảo mật của Windows NT hoặc sử dụng hệ thống bảo vệ độc lập của SQL Server.
- Hỗ trợ trong việc triển khai CSDL phân tán và phát triển ứng dụng trên Internet
- Cho phép lập trình kết nối với nhiều ngôn ngữ lập trình khác dùng xây dựng các ứng dụng đặc thù (Visual Basic, C, C++, ASP, ASP.NET, XML,...)
- Sử dụng câu lệnh truy vấn dữ liệu Transaction-SQL (Access là SQL, Oracle là PL/SQL).

+ Một hệ quản trị CSDL thường có 4 thành phần sau:

- **SQL (Structured Query Language - ngôn ngữ truy vấn mang tính cấu trúc)** là một loại ngôn ngữ máy tính phổ biến để tạo, sửa, và lấy dữ liệu từ một hệ quản trị cơ sở dữ liệu quan hệ.
- **Data Definition Language (DDL):** Các lệnh mô tả CSDL, gồm: tạo (**Create**), sửa (**Alter**), xoá (**Drop**) các bảng và ràng buộc
- **Data Manipulation Language (DML):** Các lệnh thao tác truy vấn dữ liệu, gồm: **Chèn(Insert)**, Cập **nhật(Update)**, **Xóa(Delete)**, Lựa chọn (**Select**)
- **Data Control Language (DCL):** Các lệnh điều khiển CSDL, dùng để quản lý quyền hạn của user, Gồm: **grand**, **revoke**, **deny**

+ Các kiểu dữ liệu trong SQL Server

Kiểu dữ liệu	Mô tả	Kích thước
Char	Kiểu kí tự với độ dài không thay đổi	Lưu trữ tối đa 8000 kí tự
Nchar	Giống Char nhưng hỗ trợ Unicode	Tối đa 4000 kí tự

Varchar	Dạng kí tự với độ dài thay đổi tùy theo độ dài thực của dữ liệu	Tối đa 4000 kí tự
Nvarchar	Giống Varchar nhưng hỗ trợ Unicode	Tối đa 4000 kí tự
Text	Kiểu văn bản gồm cả ký tự xuống dòng	Cỡ GB tùy dung lượng máy
Ntext	Giống Text hỗ trợ Unicode	Cỡ GB tùy dung lượng máy
Date/Time	Dữ liệu ngày giờ chia 2 dạng: Date/Time đầy đủ ngày và thời gian . SmallDateTime chỉ ngày hoặc thời gian	8 byte
Numeric	Dữ liệu dạng số nguyên: int, smallint, bigint, số thực: float, real, decimal, numeric	2 – 4 – 8 - 12 byte tùy theo kiểu số được chọn
Monetary	Kiểu tiền tệ: Money và SmallMoney	8 byte

+ Quy tắc viết lệnh trong SQL Sever:

- Không phân biệt chữ hoa, chữ thường
- Nội dung 1 lệnh SQL có thể viết trên nhiều dòng.
- Từ khoá không viết tắt hay phân cách trên nhiều dòng
- Các mệnh đề thường được đặt trên nhiều dòng khác nhau
- Ta có thể sử dụng các ký tự đặc biệt như: +, -, /, *,... để biểu diễn giá trị trong câu lệnh.

+ Chú thích và tiếng việt

- Dòng đơn ⇔ --
- Nhóm dòng ⇔ /* … */
- Sử dụng tiếng việt trong truy vấn
  - Chọn kiểu dữ liệu hỗ trợ Unicode (nchar, nvarchar, ntext)
  - Thêm tiền tố N (National Characters) vào trước chuỗi cần nhập để báo cho SQL Server đây là chuỗi Unicode

2/ Các lệnh làm việc với CSDL:

a/ Tạo CSDL:

- Cú pháp cơ bản:

CREATE DATABASE <***database-name***>
[ON <filespec>]

- Trong đó:
  - <database-name>
  - <**filespec**> ::= (**NAME** = logical_file_name,

FILENAME = os_file_name
[, **SIZE** = *n* [KB|MB|GB|TB]]
[, **MAXSIZE** = *max* [KB|MB|GB|TB] | Unlimited]
[, **FILEGROWTH** = *grow* [KB|MB|GB|TB|%]

<filespec>:để điều khiển các thuộc tính cho file được tạo

- Name: chỉ định tên logical cho file
- Filename: chỉ định tên, đường dẫn file hệ điều hành (file vật lý)
- Size: chỉ định kích thước của file, tối thiểu là 3MB
- Maxsize: chỉ định kích thước tối đa lớn nhất mà file có thể phát triển đến, có từ khóa unlimited chỉ định file được phát triển cho đến khi đĩa bị đầy
- Filegrowth: chỉ định độ tự động gia tăng của file

+ Ví dụ : Tạo CSDL QLSV

CREATE DATABASE QLSV;

b. Sử dụng CSDL:

+ Cú pháp: USE <tên CSDL>;

+ Ví dụ: USE QLSV;

c. Đổi tên CSDL

+ Cú pháp**: ALTER Database <Tên_CSDL> modify name = <tên mới>**

+ Ví dụ: Thay đổi tên CSDL

ALTER Database QLSV modify QL_SV;

d. Xóa CSDL: Khi sử dụng lệnh xóa, CSDL sẽ bị xóa khỏi vùng lưu trữ, muốn tạo thì phải thực thi lại lệnh

+ Cú Pháp: DROP DATABASE <tên CSDL>;

+ Ví dụ: DROP Database Quan_ly_SV;

Chú ý: Không dùng cách xóa thông thường để xóa tệp dữ liệu của SQL mà phải dùng lệnh Drop.

3/ Các lệnh làm việc với bảng

3.1/ Tạo cấu trúc bảng

+ Cú pháp:

CREATE TABLE <***table-name***>

(

column1 data-type ***[RBTV],***

[column2 data-type ***[RBTV],]***

...

[columnn data-type ***[RBTV],]***

[Constraint <name_RB> RBTV (column1, column2, …)]

)

Trong đó,

- **Table-name:** là tên bảng cần tạo, tuân thủ nguyên tắc định danh, không quá 128 ký tự
- **Column**: tên cột cần tạo trong bảng, mỗi bảng có ít nhất một cột
- **Data-type**: xác định kiểu dữ liệu được lưu trữ trong cột, ***Kiểu dữ liệu là thuộc tính bắt buộc***
- ***RBTV****:* gồm các ràng buộc về khuôn dạng dữ liệu hay các ràng buộc về bảo toàn dữ liệu, có thể: **NOT NULL, NULL, UNIQUE, DEFAULT, PRIMARY KEY, IDENTITY, CHECK**,…
- ***Constrain:*** dùng khi có nhiều hơn một RBTV cùng loại, đặc biệt là với RBTV khóa chính.

Có các RBTV sau:

- CHECK: kiểm tra giá trị của cột thỏa mãn điều kiện sau CHECK
- NOT NULL: dữ liệu trên cột không được bỏ trống
- PRIMARY KEY: chỉ định khóa chính cho bảng
- DEFAULT: gán giá trị mặc định
- UNIQUE: giá trị trên cột là duy nhất
- FOREIGN KEY: ràng buộc khóa ngoại. Sử dụng để tham chiếu đến bảng dữ liệu nguồn.
- IDENTITY: giá trị tự tăng, bắt đầu từ giá trị 1 và tự động tăng lên 1 đơn vị; Nếu khai báo dạng IDENTITY (n,m) thì bắt đầu từ giá trị n và tự động tăng lên m đơn vị.

+ Ví dụ:

Tạo cơ sở dữ liệu quản lý sinh viên gồm các bảng (table)

Sinhvien: các thông tin liên quan tới sinh viên

Tên trường	Kiểu dữ liệu	Chú thích
Masv	int (tự tăng)	Khóa
Tensv	Nvarchar	Tên sinh viên, dữ liệu không được bỏ trống
Gioitinh	Nvarchar	Giới tính, mặc định là giá trị Nam
Ngaysinh	Date	Ngày tháng năm sinh, nhỏ hơn ngày hiện tại
Que	Nvarchar	Quê, giá trị không được bỏ trống
Lop	Nvarchar	Lớp

Bảng Monhoc: Thông tin về các môn học trong nhà trường

Tên trường	Kiểu dữ liệu	Chú thích
Mamh	Int (tự tăng)	Khóa
Tenmh	Nvarchar	Tên môn học, giá trị trên cột là duy nhất
DVHT	Int	Số đơn vị học trình, nằm trong khoảng từ 2 đến 9

Bảng Ketqua: Điểm thi của sinh viên

Tên trường	Kiểu dữ liệu	Chú thích

Masv	Int	khóa
Mamh	Int	
Diem	Float	Điểm từ 0-10

```
create database QLSV --Tạo CSDL sinh viên
use QLSV –Sử dụng CSDL
go
Create Table Sinhvien
(
 MaSV int identity primary key,
 TenSV Nvarchar(30) not null,
 GT NVarchar (5) default 'Nam',
 Ngaysinh Date check (Ngaysinh<getdate()),
 Que Nvarchar(50) not null,
 Lop Nvarchar (10)
 )
 drop table Sinhvien
 --Tạo bảng Môn học
 Create table Monhoc
 (
 MaMH int identity primary key,
 TenMH Nvarchar (20) unique,
 DVHT int check (DVHT between 2 and 9)
)
--Tạo bảng điểm thi
create table Ketqua
(
 MaSV int,
 MaMH int,
 Diem float check (Diem between 0 and 10)
 constraint RB_Khoa primary key (MaSV,MaMH)--Định nghĩa khóa có hai thuộc tính
 )
```

3.2 Các thao tác trên bảng

+ Xem thông tin về bảng: **sp_help <tên bảng>**

+ Đổi tên bảng: **exec sp_rename <tên cũ> <tên mới>**

+ Xóa bảng: **Drop Table <tên bảng>**

+ Thêm cột vào bảng:

ALTER Table <tên_bảng>

ADD <tên_Cột> Data_type [RBTV][,..]

+ Thay đổi kiểu dữ liệu của cột

ALTER Table <tên_bảng>

ALTER Column <tên_cột> <kiểu dl mới>

+ Hủy bỏ một cột trong bảng:

ALTER Table <tên_bảng>

DROP Column <ds tên_cột>

+ Thêm ràng buộc cho cột

ALTER Table <tên_bảng>

ADD Constraint <tên_ràng_buộc> <Loại_ràng_buộc> (tên cột)

+ Hủy ràng buộc đã đặt

ALTER Table <tên_bảng>

DROP Constraint <tên_RB>

+ Đổi tên cột:

EXEC SP_Rename '<tên bảng.tên cột>', 'tên mới", 'COLUMN'

+**Bật /tắt các ràng buộc**

- Bật ràng buộc:

ALTER TABLE Tên_bảng

NOCHECK CONSTRAINT ALL | Tên_constraint [,...]

- Tắt ràng buộc:

ALTER TABLE Tên_bảng

CHECK CONSTRAINT ALL | Tên_constraint [,...]

Ví dụ:

```
--Xem thông tin về các bảng
 sp_help sinhvien
 sp_help monhoc
 sp_help Ketqua
 -- Chỉnh sửa bảng
 -- Đổi tên bảng
 exec sp_rename "sinhvien", "SV"
 exec sp_rename "SV","sinhvien"
 --
/*Chèn thêm 1 cột*/
              Alter table Sinhvien
              ADD
              Dienthoai varchar(11),
              CMTND nvarchar(40)
       /*thay đổi kiểu dữ liệu của cột*/
              Alter table sinhvien
              Alter column CMTND int not null
       /*thêm ràng buộc*/
              alter table Monhoc
              ADD
              constraint RB_TenMH Unique(TenMH)
              /*Từ khóa unique giúp trường đó có giá trị duy nhất VD như CMTND*/
```

```
/*Xóa cột*/
	Alter table Sinhvien
	Drop column CMTND, Dienthoai
/*Xóa ràng buộc*/
	Alter table monhoc
	Drop constraint RB_TenMH
```

3.3 Tạo liên kết giữa các bảng (Tạo ràng buộc khóa Ngoại)

Một khóa ngoại là một cột hoặc sự kết hợp của các cột được sử dụng để tạo một liên kết giữa dữ liệu trong hai bảng. Một số chú ý khi tạo liên kết giữa các bảng:

- Cột được tham chiếu trong bảng tham chiếu phải là khoá chính (hoặc là khoá phụ).
- Bảng tham chiếu phải được định nghĩa trước. Do đó, nếu các bảng có mối quan hệ vòng, ta có thể không thể định nghĩa ràng buộc FOREIGN KEY ngay trong câu lệnh CREATE TABLE mà phải định nghĩa thông qua lệnh ALTER TABLE.

+ Cú pháp: có hai cách tạo liên kết khóa ngoại

- Cách 1: Tạo lập ngay trong quá trình tạo bảng với điều kiện bảng khóa chính đã được định nghĩa trước

  CREATE TABLE Tên_bảng
  (
  …..
  FOREIGN KEY (Tên_cột) **REFERENCES** <Tên bảng tham chiếu_khóachính>
)
- Cách 2: Tạo liên kết sau khi tạo các bảng (cách 2 hay được sử dụng để tránh tình trạng các bảng có mối quan hệ vòng)

  ALTER TABLE Tên_bảng
  ADD
  CONSTRAINT <Tên_liên_kết> **FOREIGN KEY** (Tên_cột_khóa_chính) **REFERENCES** <Tên bảng tham chiếu> (tên cột tham chiếu)

Ví dụ 1: Tạo liên kết ngay trong quá trình tạo bảng

```
create table Ketqua
(
 MaSV int,
 MaMH int,
 Diem int check (Diem between 0 and 10)
 constraint RB_Khoa primary key (MaSV,MaMH),
 Foreign key (MaSV) References sinhvien, --Tạo liên kết với bảng Sinhvien
 Foreign key (MaMH) References Monhoc    --Tạo liên kết với bảng môn học
 )
```

Ví dụ 2: Tạo liên kết sau khi đã tạo bảng

```
Alter Table Ketqua
Add
```

```
	Constraint R1 foreign key (MaSV) References Sinhvien(MaSV)

Alter Table Ketqua
Add
	Constraint R2 Foreign Key (MaMH) References Monhoc(MaMH)
```

Chú ý: Sau khi tạo liên kết khóa ngoại, muốn xóa các bảng có trong liên kết ta phải thực hiện lệnh xóa ràng buộc khóa ngoại trước. Ví dụ, sau khi tạo ràng buộc giữa ba bảng Sinhvien, Monhoc và Ketqua, muốn xóa bảng Sinhvien ta phải xóa ràng buộc bằng lệnh sau:

```
Alter Table Ketqua
Drop constraint R1,R2
```

4. Tạo chỉ mục Index:

- Index là chỉ mục quan trọng trong CSDL đặc biệt với CSDL lớn.
- Index có thể thiết lập cho 1 hoặc nhiều cột của bảng
- Index được sắp xếp nhằm hỗ trợ việc tìm kiếm, truy vấn dữ liệu một cách nhanh chóng.

CREATE [UNIQUE] [CLUSTERED] [NONCLUSTERED] INDEX <tên index>
ON <tên bảng>(tên cột,..)

- **Unique**: dữ liệu cột Index là duy nhất không lặp lại
- **Clustered**: dữ liệu được sắp xếp vật lý trên ổ đĩa
- **Nonclustered**: dữ liệu được sắp xếp logic, nhanh trong nhập liệu

+ Ví dụ: tạo index trên cột MaNV của bảng Nhân viên

CREATE INDEX ID_MANV ON NHANVIEN(MANV)

+ Xóa INDEX

- DROP INDEX<tên index> ;
- Ví dụ: xóa index vừa thiết lập

DROP INDEX ID_MANV;

II. BÀI TẬP

Bài 1: Cho lược đồ CSDL quan hệ của bài toán quản lý SPJ sau:

- **NCC**(MaNCC, Ten, Heso, ThPho): mô tả thông tin về nhà cung cấp vật tư. Mỗi nhà cung cấp có một mã số duy nhất, một tên, một hệ số xếp hạng và ở một thành phố nào đó.
- **VATTU**(MaVT, Ten, Mau, TrLuong): mô tả thông tin về vật tư. Mỗi vật tư có một mã số duy nhất, một tên, quy cách màu sắc, trọng lượng và được lưu trữ tại thành phố nào đó.
- **DUAN**(MaDA, Ten, ThPho): mô tả thông tin về dự án sản xuất. Mỗi dự án có một mã số duy nhất, một tên và được thực hiện tại một thành phố nào đó.
- **CC**(MaNCC, MaVT, MaDA, SoLuong): mô ta sự cung cấp vật tư cho dự án sản xuất của các nhà cung cấp. Mỗi nhà cung cấp có thể cung cấp nhiều vật tư cho một hoặc nhiều dự án. Mỗi dự án sản xuất có thế cần nhiều vật tư. Mỗi một bộ dữ liệu trong bảng cho biết nhà cung cấp vật tư cho dự án sản xuất với một số lượng cụ thể.

+ Thuộc tính và kiểu dữ liệu:

Thuộc tính	Miền giá trị
MaNCC	char(5)
Ten	varchar(40)
Heso	int
ThPho	varchar(20)
MaDA	char(5)
MaVT	char(5)
Mau	varchar(15)
TrLuong	float
SLuong	int

Yêu cầu:

1. Tạo một CSDL có tên là **SPJ** có các quan hệ trên với các ràng buộc như sau:

- Tên vật tư, tên thành phố trong bảng VATTU không được bỏ trống.
- Hệ số xếp hạng nằm trong khoảng từ 0-100, mặc định là 0.
- Màu của sản phẩm là duy nhất.
- Số lượng mặc định là 0.
- Trọng lượng tối thiểu của một sản phẩm là 2.0.
- Các ràng buộc khóa chính, khóa ngoại (Bằng 2 cách)
- Tạo chỉ mục Index cho các bảng (chọn cột index là cột khóa chính)

2. Thực hiện các thao tác trên bảng:

- Xem thông tin về các bảng đã tạo.
- Đổi tên các bảng: NCC → NhaCC; CC → CungCap.
- Thêm cột ThPho vào bảng VATTU để thể hiện vật tư đó được lưu trữ tại thành phố nào với giá trị mặc định là Hà Nội, thêm cột DIADIEM kiểu nText vào bảng DUAN với giá trị mặc định là Hà Nội.
- Bổ sung ràng buộc cho các cột như sau: Cột Màu chỉ có các màu cơ bản "Xanh", "Đỏ", 'Tím", "Vàng", "Trắng", "Đen"; Tên Vật tư, dự án, ThPho phải là duy nhất.
- Đổi tên cột DIADIEM thanh DD.
- Hủy bỏ ràng buộc mặc định tại cột ThPho của các bảng.
- Xóa cột DD trong bảng DUAN.

Bài 2: Cho lược đồ CSDL quan hệ sau:

CHUYENBAY(MaCB, GaDi, GaDen, DoDai, GioDi, GioDen, ChiPhi): mô tả thông tin về chuyến bay.. Mỗi chuyến bay có một mã số duy nhất, đường bay, giờ đi và giờ đến. Thông tin về đường bay được mô tả bởi ga đi, ga đến, độ dài đường bay, chi phí phải trả cho phi công.

MAYBAY(MaMB, Hieu, TamBay): mô tả thông tin về máy bay. Mỗi máy bay có một mã số duy nhất, tên phân loại(Hieu) và tầm bay là khoảng cách xa nhất máy bay có thể bay mà không cần tiếp nhiên liệu. Một máy bay chỉ có thể thực hiện các chuyến bay có độ dài đường bay nhỏ hơn tầm bay của máy bay đó.

NHANVIEN(MaNV, Ten, Luong) mô tả thông tin về nhân viên phi hành đoàn gồm phi công và tiếp viên. Mỗi nhân viên có một mã số duy nhất, tên và mức lương.

CHUNGNHAN(MaNV, MaMB): mô tả thông tin về khả năng điều khiển máy bay của phi công nào đó. Một phi công chỉ có thể lái một chuyến bay nếu như phicông đó được công nhận có khả năng lái được loại máy bay có thể thực hiện chuyến bay đó.

+ **Mô tả các thuộc tính:**Tạo các ràng buộc khóa ngoại, khóa chính và not null.

Thuộc tính	Miền xác định
MaCB	char(5)
GaDi	varchar(50)
GaDen	varchar(50)
DoDai	int
GioDi	time
GioDen	time
ChiPhi	int
MaMB	int
Hieu	varchar(50)
TamBay	int
MaNV	char(9)
Ten	varchar(50)
Luong	int

ĐỌC THÊM VỀ CÁC CÔNG CỤ TOÀN VẸN DỮ LIỆU

1/ Toàn vẹn thực thể:

SQL Server đưa ra ba cơ chế để cung cấp toàn vẹn thực thể: **PRIMARY KEY**, ràng buộc **UNIQUE**, và **thuộc tính IDENTITY**

1.1. Ràng buộc Primary Key

Một bảng có một cột, hoặc sự kết hợp của các cột, giá trị của chúng có thể được sử dụng để định dạng mỗi duy nhất mỗi hàng trong bảng. Khi một cột đơn định dạng mỗi hàng của bảng là duy nhất, nó được gọi là **Primary Key** của bảng. Nó thực thi toàn vẹn thực thể trên bảng. Một bảng chỉ có thể có một khóa chính. Khóa chính có thể bao gồm một hoặc nhiều cột.

Tạo và sửa đổi khóa chính

+ Tạo nó như là một phần của định nghĩa bảng, khi một bảng được tạo.

Cú pháp:

CREATE TABLE Tên_bảng

(<tên cột> <kiểu dữ liệu> PRIMARY KEY)

Ví dụ: Tạo ràng buộc khóa chính cho bảng NV là MaNV

CREATE TABLE NV

(MaNV int PRIMARY KEY)

+ Thêm nó tới một bảng sẵn có, nếu không có ràng buộc khóa chính khác đã tồn tại trong bảng. Nhớ rằng chỉ có thể có duy nhất một ràng buộc khóa chính. **Chú ý, trước khi thêm ràng buộc khóa chính**, cần phải thêm ràng buộc not null trước. Cần đảm bảo thuộc tính khóa chính là not null, nếu không sẽ bị báo lỗi.

Cú pháp:

ALTER TABLE Tên_bảng

ADD CONSTRAINT <Tên_ràngbuộc>

PRIMARY KEY (<tên cột>)

Ví dụ: Thêm ràng buộc khóa chính cho bảng **NV** là MaNV

alter table NV add constraint RB1 Primary key(MaNV)

Sửa đổi hoặc xóa nó, nếu nó tồn tại. Để sửa đổi một khóa chính, đầu tiên bạn phải xóa khóa chính hiện có, và sau đó tạo lại một khóa chính mới.

Các điểm cần lưu ý khi tạo một khóa chính

✓ Không thể thay đổi độ dài của một cột một khi nó được định nghĩa ràng buộc khóa chính.

✓ Khi thêm một khóa chính tới một cột, hoặc các cột sẵn có trong một bảng, SQL Server kiểm tra dữ liệu sẵn có trong cột để đảm bảo rằng dữ liệu không có các giá trị lặp lại hoặc giá trị NULL. Nếu một cột có giá trị null hoặc dữ liệu lặp lại, thì SQL Server trả lại một lỗi, và không thêm ràng buộc.

✓ Không thể xóa một ràng buộc khóa chính, nếu một ràng buộc khóa ngoài đang tham chiếu tới nó trong một bảng khác; đầu tiên bạn phải xóa ràng buộc khóa ngoài trước.

1.2. Ràng buộc UNIQUE

Ràng buộc UNIQUE chỉ ra rằng tất cả các giá trị trong một cột cho trước phải là duy nhất; cột đó phải có một giá trị khác nhau ở tất cả mọi hàng của bảng. Một bảng có thể có nhiều ràng buộc duy nhất, trong trường hợp đó chúng phải thỏa mãn cho mọi hàng. Ràng buộc UNIQUE rõ ràng mang lại toàn vẹn thực thể cho bảng, vì nó đảm bảo tất cả các hàng đều khác nhau

+ Tạo mới:

Cú pháp:

CREATE TABLE Tên_bảng

<tên_cột><kiểu dữ liệu> UNIQUE

Ví dụ: Tạo ràng buộc Unique cho cột TenP của bảng PHONG

CREATE TABLE phong

(TenP VARCHAR(20) UNIQUE)

+ Bổ sung ràng buộc UNIQUE

Cú pháp:

ALTER TABLE <tên bảng>

ADD CONSTRAINT <Tên RB> UNIQUE (<tên cột>)

Ví dụ: bổ sung ràng buộc UNIQUE cho cột TenP của bảng PHONG

ALTER TABLE PHONG

Add constraint RB3 UNIQUE (TenP)

Có thể sử dụng ràng buộc UNIQUE để đảm bảo rằng các giá trị lặp lại thì không được nhập vào trong một cột xác định. Các chức năng của ràng buộc UNIQUE giống như ràng buộc khóa PRIMARY, ngoại trừ nó cho phép giá trị NULL.

1.3. Thuộc tính IDENTITY

Bạn có thể áp dụng thuộc tính IDENTITY tới một cột với kiểu dữ liệu là decimal, int, smallint, hoặc numeric. Nó tự động phát sinh một giá trị là duy nhất bảng. Bạn chỉ có thể gắn thuộc tính IDENTITY cho một cột trong một bảng. Mặc định, giá trị bắt đầu của thuộc tính này thiết lập là 1. Bạn có thể thiết lập thuộc tính IDENTITY ở thời điểm tạo bảng.

Cú pháp:

CREATE TABLE Tên_bảng

(Tên_cột Kiểu_Dữ_liệu IDENTITY

[(<giá_trị_khởi_đầu>, <giá_trị_gia_tăng>)])

Trong đó:

Tên_cột: Tên của cột mà thuộc tính IDENTITY được gắn vào.

<giá_trị_khởi_đầu>: Giá trị bắt đầu hay khởi tạo của cột IDENTITY.

<giá_trị_gia_tăng>: Giá trị bước để phát sinh giá trị tiếp theo cho cột. Nó có thể là số âm.

Ví dụ: Phát sinh mã phòng trong bảng PHONG như sau:

```
create table Phong
( MaP int Identity(1,1))
```

Ở đây giá trị ban đầu của MaP được thiết lập là 1, và mỗi lần một bản ghi được thêm vào bảng, nó tự động được tăng thêm 1.

Chú ý: Không thể thông thể thay đổi một sột có sẵn để nó trở thành cột IDENTITY hoặc bỏ thuộc tính IDENTITY.

2. Các công cụ để thực thi toàn vẹn miền.

SQL Server cung cấp bốn cơ chế để đảm bảo toàn vẹn miền.

2.1. Định nghĩa Default.

Tất cả các cột trong một bản ghi phải có một giá trị, ngay cả khi giá trị đó là NULL. Nó phải không được bỏ trắng. Khi các giá trị NULL không phù hợp với yêu cầu, và dữ liệu mong muốn không được cung cấp, hãy chỉ ra một định nghĩa DEFAULT cho cột.

Định nghĩa DEFAULT cung cấp giá trị sẽ được lưu trữ trong một cột, nếu người dùng không chỉ rõ giá trị nào.

Cú pháp:

CREATE TABLE <Tên_bảng>

(<Tên_cột> < Kiểu_Dữ_liệu> DEFAULT <giá_trị_mặc_định>)

Giá trị mặc định phải thích hợp với kiểu dữ liệu của cột để định nghĩa DEFAULT có thể áp dụng được. Ví dụ, giá trị mặc định của một cột int phải là một số nguyên, không thể là một xâu ký tự.

Ví dụ: tạo giá trị mặc định cho cột HSL thuộc bảng NV là 2.34

```
create table NV
(HSL float DEFAULT 2.34)
```

+ Thêm ràng buộc DEFAULT cho cột đã có:

Alter Table <name>

Add Default <giá trị> for <tên cột>

Ví dụ: Thêm giá trị mặc định cho cột HSL là 2.34

```
alter table NV
Add Default 2.34 for HSL
```

2.2. Ràng buộc khóa FOREIGN

Một khóa ngoài là một cột hoặc sự kết hợp của các cột được sử dụng để tạo một liên kết giữa dữ liệu trong hai bảng. Mục đích chính của một ràng buộc FOREIGN KEY là điều khiển dữ liệu có thể được lưu trữ trong bảng con. Một ràng buộc khóa ngoài đảm bảo rằng chỉ dữ liệu tồn tại trong khóa chính của bảng chủ mới được phép vào trường khóa ngoài. Một bảng có thể có nhiều khóa ngoài.

Cú pháp:

CREATE TABLE Tên_bảng

(Tên_cột Kiểu_dữ_liệu,
FOREIGN KEY (Tên_cột) REFERENCES Tênbảng_khóachính)

Trong đó: + Tên_cột: Tên của cột khóa ngoài.

+ Tênbảng_khóachính: Tên bảng chứa khóa chính mà khóa ngoài tham chiếu.

Ví dụ: Tạo liên kết giữa bảng NV và bảng PHONG. Bảng NV và bảng PHONG có chung cột MaP. Cột MaP là khóa chính của bảng PHONG và là khóa ngoại của bảng NV, nên sẽ đặt ràng buộc trên bảng NV như sau:

```
create table NV
( MaP int
  foreign key (MaP) references PHONG )
```

+ Tạo ràng buộc Foreign key cho cột có sẵn:

Alter Table <tên bảng>
Add foreign key (<tên cột khóa ngoài> references <tên bảng>(tên cột khóa chính)

Ví dụ: Thêm ràng buộc foreign key cho cột MaP của bảng NV, tham chiếu sang cột MaP của bảng PHONG.

```
alter table NV
add foreign key (MaP) references PHONG(MaP)
```

2.3. Ràng buộc CHECK

Ràng buộc CHECK nhằm kiểm tra miền giới hạn của các giá trị được lưu trữ trong một cột.

Cú pháp:

CREATE TABLE <Tên_bảng>
(<Tên_cột>< Kiểu_dữ_liệu >CHECK (<biểu thức điều kiện>)

Ví dụ: Đặt điều kiện ràng buộc cho sột HSL>0

```
create table NV
(HSL float CHECK (HSL>0))
```

+ Tạo ràng buộc Check cho cột có sẵn:

Alter Table <tên bảng>
Add Check (<điều kiện>)

Ví dụ: Thêm điều kiện SoNV>0 cho cột SoNV của bảng PHONG

```
ALTER Table PHONG
Add Check (SoNV>0)
```

2.4. Ràng buộc Not Null

Tự chọn cơ bản nhất để thực thi toàn vẹn miền là chỉ ra ràng buộc NOT NULL trên một cột. NOT NULL có nghĩa là một cột không thể có một giá trị NULL. Người dùng phải cung cấp một vài giá trị cho cột. Do vậy thực thi một ràng buộc NOT NULL bắt người dùng nhập một vài giá trị cho cột đó.

Cú pháp:

CREATE TABLE < Tên_bảng >

(<Tên_cột> <Kiểu_dữ_liệu> NOT NULL, ...)

Ví dụ: Tạo ràng buộc NOT NULL cho cột MaNV của bảng NV

```
create table NV
( MaNV int not null)
```

+ Thêm ràng buộc NOT NULL cho cột có sẵn:

Alter Table <tên bảng>

Alter Column <tên cột> <tên kiểu dữ liệu> NOT NULL

Ví dụ: Thêm ràng buộc NOT NULL cho cột HT của bảng NV

```
Alter Table NV
Alter Column HT varchar(30) Not Null
```

2.5. Rule

Rule cung cấp một ý nghĩa luân phiên của việc thực thi toàn vẹn người dùng định nghĩa trong cơ sở dữ liệu. Rule giống với ràng buộc CHECK, nhưng được tạo như một đối tượng riêng độc lập, và sau đó gắn cho một cột. Một cột đơn chỉ có thể có một luật đơn gắn với nó. Nếu bạn đòi hỏi nhiều ràng buộc, thì sử dụng các ràng buộc CHECK thay vì các luật.

Cú pháp

CREATE RULE tên_luật

AS biểuthức_điềukiện

trong đó:

- tên_luật: Xác định một tên cho luật.
- biểuthức_điềukiện: Chỉ ra giá trị đúng cho cột mà luật sẽ được gắn vào.

Cú pháp để gắn luật tới một cột:

sp_bindrule <tên_luật>, <tên_bảng.tên_cột >

Ví dụ, chúng ta sử dụng một luật để đảm bảo rằng hệ số lương luôn nằm từ 2.34 đến 8.0. Đầu tiên bạn sẽ phải tạo luật, và sau đó gắn nó tới cột HSL như sau:

```
create Rule check_hsl
as @pnr between 2.34 and 8.00
go
sp_bindrule 'check_hsl','NV.HSL'
```

3. Các công cụ để thực thi toàn vẹn tham chiếu

Bạn có thể sử dụng các ràng buộc khóa ngoài và CHECK để thực thi toàn vẹn tham chiếu. Để cung cấp toàn vẹn tham chiếu, bạn phải liên kết các ràng buộc khóa ngoài tới khóa chính trong bảng khác.

4. Các công cụ để thực thi toàn vẹn người dùng định nghĩa

Tất cả các ràng buộc chúng ta đã thảo luận cho đến bây giờ thực thi toàn vẹn người dùng định nghĩa. Thêm vào đó, bạn có thể tạo những thủ tục được lưu và các trigger để thực thi toàn vẹn người dùng định nghĩa.