

Mô hình quan hệ dữ liệu (Diagram)

[Share](#) [Đọc: 1675-Ngày đăng: 09-06-2009-Ngày sửa: 18-06-2009]

Đối với người sử dụng đã quen thuộc với loại cơ sở dữ liệu Microsoft Access, màn hình Relationship trong Microsoft Access sẽ giúp cho bạn hiển thị và tạo lập các mối quan hệ dữ liệu giữa các bảng trong tập tin cơ sở dữ liệu MDB.

1/- Khái niệm về mô hình quan hệ dữ liệu :

Thực ra việc tạo mối quan hệ giữa các bảng là nhằm trao đổi và chia sẻ thông tin qua lại giữa các bảng với nhau, đồng thời cũng giúp kiểm tra tính tồn tại dữ liệu (khóa ngoại).

Riêng trong môi trường cơ sở dữ liệu của Microsoft SQL Server khi đến giai đoạn này thì bạn chỉ biết được việc tạo cấu trúc các bảng có thể tạo thêm các mối quan hệ dữ liệu ngầm định giữa các bảng thông qua quy tắc kiểm tra ràng buộc về khóa ngoại (foreign key constraint). Tuy nhiên các mối quan hệ này không thể hiện một cách trực quan để giúp cho người sử dụng dễ dàng quan sát.

Thấy được sự thiếu sót này trong các phiên bản cũ trước đây. Vì thế kể từ phiên bản 7.0 trở lên, Microsoft đã đưa một đối tượng mới trong cơ sở dữ liệu của Microsoft SQL Server, đó chính là đối tượng mô hình quan hệ dữ liệu (diagram). Trong mô hình quan hệ dữ liệu này, bạn có thể dễ dàng nhìn thấy mối quan hệ dữ liệu giữa các bảng có liên quan với nhau, thực hiện một số các hành động tác động trực tiếp ngay trong các bảng như là tạo cấu trúc bảng mới, xóa cấu trúc bảng đã có, chỉ định khóa chính cho bảng, thêm các bảng hiện có vào bên trong mô hình quan hệ dữ liệu.

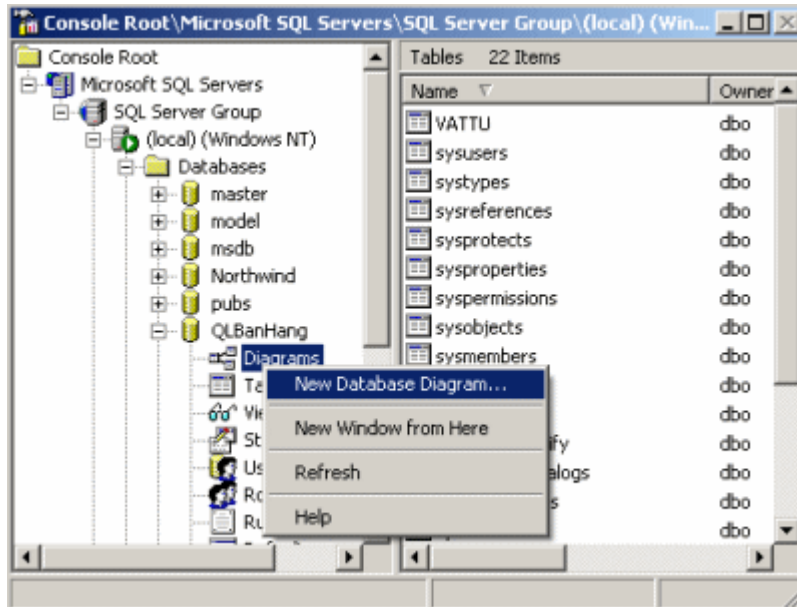
2/- Tạo mới mô hình quan hệ dữ liệu :

Đối tượng mô hình quan hệ dữ liệu chỉ được tạo ra và hiển thị trong tiện ích **Enterprise Manager** nhằm hệ thống Microsoft SQL Server kiểm tra các ràng buộc dữ liệu khóa ngoại trong cơ sở dữ liệu và giúp cho người dùng dễ dàng thấy được các mối liên kết dữ liệu giữa các bảng một cách trực quan hơn.

Thông thường các mối kết hợp bên trong mô hình quan hệ dữ liệu sẽ được tự động tạo ra khi bạn chèn các bảng vào mô hình này. Để tạo mới mô hình quan hệ dữ liệu trong một cơ sở dữ liệu bạn lần lượt thực hiện các bước sau :

• Bước 1 :

Khởi động tiện ích **Enterprise Manager**, chọn cơ sở dữ liệu có chứa các bảng cần tạo mới mô hình quan hệ dữ liệu. Chọn chức năng **New Database Diagram** trong thực đơn tắt sau khi click chuột phải trên đối tượng **Diagrams**.



Tạo mới mô hình quan hệ dữ liệu

• **Bước 2 :**

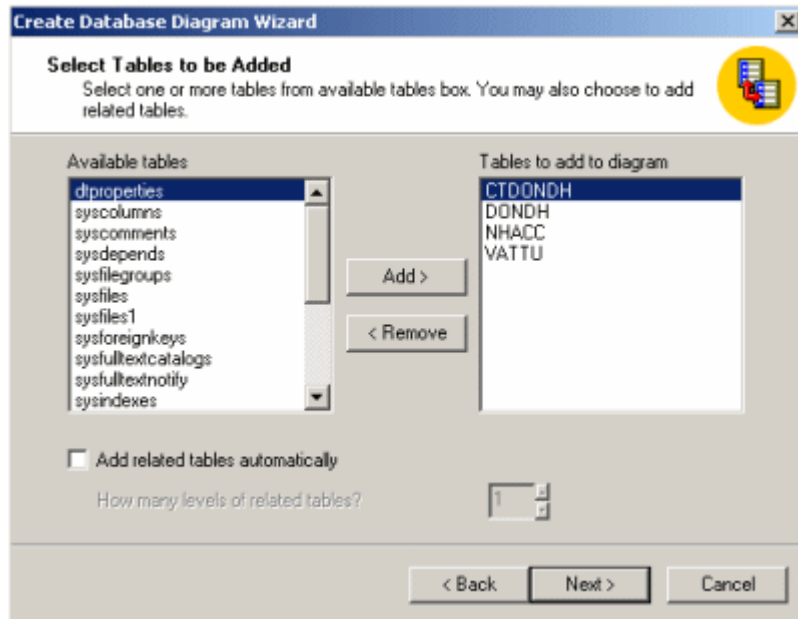
Trong màn hình Welcome việc tạo mới mô hình quan hệ dữ liệu bằng trình trợ giúp thông minh, hệ thống hiển thị cho bạn biết các bước sẽ thực hiện trong quá trình tạo mới mô hình quan hệ dữ liệu. Nhấn **Next** để tiếp tục.



Màn hình tạo mới mô hình quan hệ dữ liệu

• **Bước 3 :**

Trong màn hình chọn các bảng (**Select Tables**), lần lượt chọn các bảng trong danh sách bảng đang có trong cơ sở dữ liệu (**Available tables**) mà bạn muốn đưa chúng vào mô hình quan hệ dữ liệu, sau đó nhấn **Add**. Nhấn **Next** để tiếp tục.

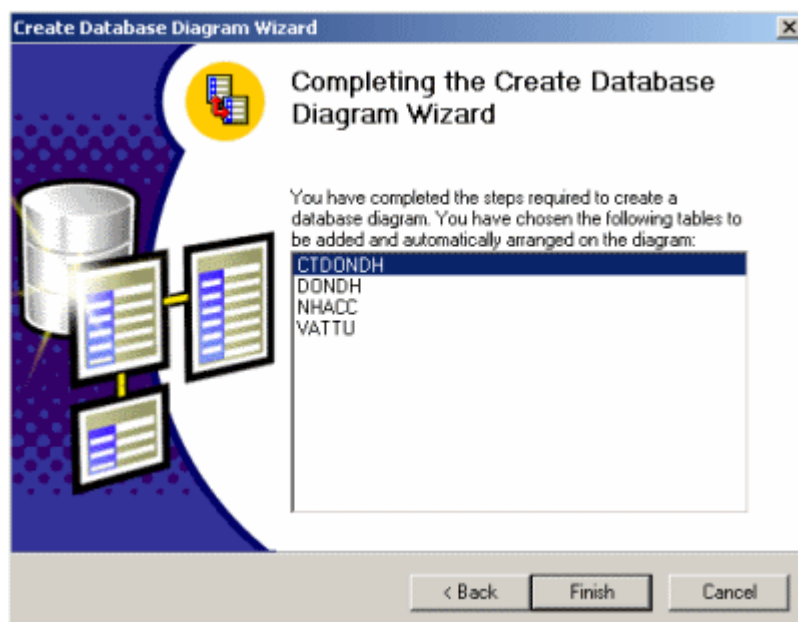


Chọn các bảng đưa vào mô hình quan hệ dữ liệu

Ngược lại khi muốn bỏ đi các bảng có trong mô hình quan hệ dữ liệu thì bạn sẽ nhấn vào nút **Remove** sau khi đã chọn các bảng cần xóa trong danh sách bên phải (**Tables to add to diagram**).

• **Bước 4 :**

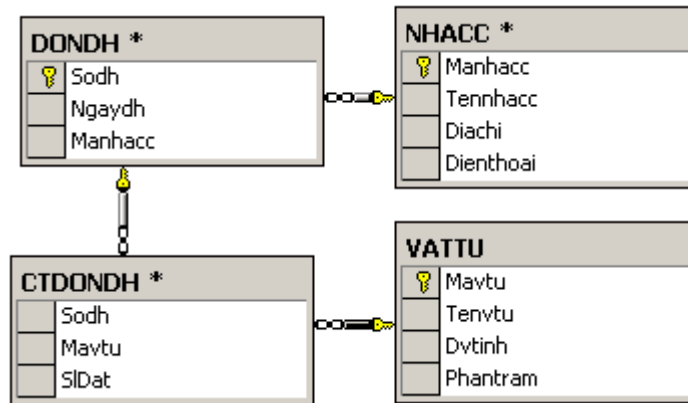
Trong màn hình kết thúc (**Completing**) hệ thống sẽ hiển thị danh sách các bảng mà bạn đã chọn ở bước 3, các bảng này sẽ xuất hiện đầy đủ trong mô hình quan hệ dữ liệu. Nếu bạn thấy còn thiếu bảng nào cần bổ sung thêm thì bạn có thể nhấn nút **Back** để quay lại bước 3. Nhấn nút **Finish** để kết thúc quá trình tạo mới mô hình quan hệ dữ liệu.



Kết thúc quá trình tạo mới mô hình quan hệ dữ liệu

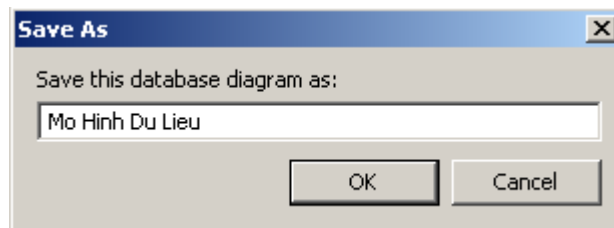
Sau một thời gian ngắn, hệ thống sẽ tự động tạo ra các mối liên kết giữa các bảng đã chọn và sắp xếp vào bên trong mô hình quan hệ dữ liệu.

Bên trong cơ sở dữ liệu Microsoft SQL Server, một mô hình quan hệ dữ liệu đã được tạo lập theo mô hình bên dưới.



Mối liên kết giữa các bảng trong mô hình quan hệ dữ liệu

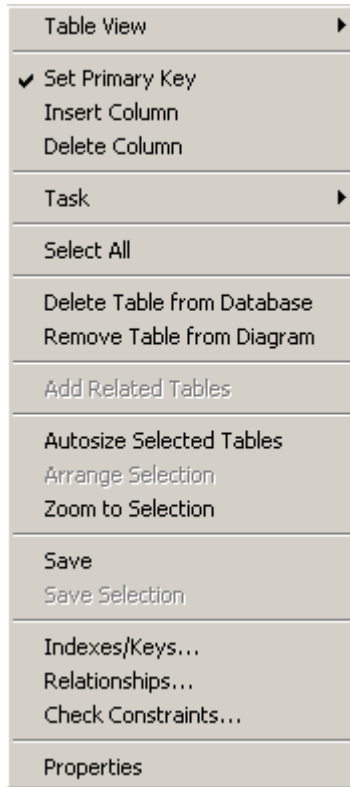
Cuối cùng bạn nhấn vào biểu tượng nút **Save** trên thanh công cụ và gõ vào tên của mô hình quan hệ dữ liệu dùng để lưu lại mô hình quan hệ dữ liệu vừa mới tạo ở các bước trên.



Nhập tên mô hình quan hệ dữ liệu

3/- Các chức năng trong mô hình quan hệ dữ liệu :

Trong mô hình quan hệ dữ liệu có rất nhiều chức năng để người sử dụng có thể thực hiện khi đang làm việc bên trong mô hình này. Tuy nhiên trong phần này chỉ giới thiệu bạn một số chức năng chính tác động vào các bảng dữ liệu bên trong cơ sở dữ liệu. Các chức năng này sẽ thông qua hai thực đơn tắt khi bạn nhấn chuột phải trên tên một bảng bất kỳ hoặc trên một vùng trống trong mô hình quan hệ dữ liệu.



Thực đơn tắt của bảng



Thực đơn tắt của mô hình quan hệ dữ liệu

3.1/- Sửa đổi cấu trúc bảng hiện có :

Bạn có thể chọn chức năng **Insert column** dùng để thêm cột, hoặc chức năng **Delete column** dùng để xóa cột, hoặc chức năng **Set Primary Key** dùng để định nghĩa lại khóa chính của bảng đang được chọn. Trong trường hợp thêm vào các cột mới, để có thể chỉ định được các thuộc tính liên quan đến các cột mới như là tên cột, kiểu dữ liệu, độ rộng, dữ liệu tại cột có cho phép dữ liệu bỏ trống, ... bạn nên chọn chức năng **Properties** để có thể thấy được đầy đủ các thuộc tính liên quan đến các cột dữ liệu giống như màn hình thiết kế bảng trong

tiện ích **Enterprise Manager**. Mặc định trong mô hình quan hệ dữ liệu chỉ hiển thị ra tên các cột có bên trong bảng.

3.2/- Tạo mới, hủy bỏ bảng trong cơ sở dữ liệu :

Bạn có thể chọn chức năng **New Table** để tạo mới bảng và tự động chèn bảng đó vào trong mô hình quan hệ dữ liệu hoặc chức năng **Delete Table From Database** để hủy bỏ bảng hiện chọn ra khỏi cơ sở dữ liệu. Bạn cần thận khi chọn chức năng **Delete Table From Database** bởi vì sau khi đồng ý hủy bỏ thì ngay sau đó dữ liệu và cấu trúc của bảng sẽ không còn được lưu trữ bên trong cơ sở dữ liệu nữa và bạn sẽ không thể nào khôi phục lại được.

3.3/- Chèn, xóa bảng trong mô hình quan hệ dữ liệu :

Bạn có thể chọn chức năng **Add Table** để có thể chọn các bảng đã có trong cơ sở dữ liệu và chèn nó vào mô hình quan hệ dữ liệu hoặc chức năng **Add Related Tables** để hệ thống Microsoft SQL Server tự động chèn các bảng có quan hệ với bảng hiện hành đang chọn vào bên mô hình quan hệ dữ liệu. Các chức năng này vẫn được thường sử dụng để chèn các bảng mới vừa tạo nhưng lại có quan hệ với các bảng hiện đang có trong mô hình quan hệ dữ liệu.

Ngoài ra bạn cũng có thể chọn chức năng **Remove Table From Diagram** để xóa bảng đang chọn ra khỏi mô hình quan hệ dữ liệu, tuy nhiên cấu trúc và dữ liệu của bảng này hoàn toàn không hề mất đi trong cơ sở dữ liệu.

3.4/- Thay đổi cách trình bày :

Bạn có thể chọn chức năng **Arrange Tables** để hệ thống tự động sắp xếp lại vị trí hiển thị của các bảng hiện có trong mô hình quan hệ dữ liệu. Việc sắp xếp lại các bảng này nhằm giúp cho bạn dễ dàng thấy rõ mối liên kết giữa các bảng.

Đối với một mô hình quan hệ dữ liệu lớn có chứa rất nhiều bảng thì bạn có thể chọn chức năng **Zoom** với các tỉ lệ khác nhau (10%, 25%, 50%, 75%, 100%, 150% và 200%) để có thể thu nhỏ hoặc phóng to các bảng và các mối liên kết của chúng cho dễ nhìn.

Ngoài ra bạn có thể chọn chức năng **Show Relationship Labels** để hiển thị tên của các quy tắc kiểm tra dữ liệu khóa ngoại liên kết quan hệ giữa các bảng.

3.5/- In mô hình quan hệ dữ liệu :

Bạn có thể chọn chức năng **View Page Breaks** để thấy được các bảng dữ liệu trong mô hình quan hệ dữ liệu lớn được trình bày trên nhiều trang giấy khác nhau.

Ngoài ra để in mô hình quan hệ dữ liệu ra máy in, bạn có thể sử dụng chức năng **Print** hoặc nhấn và biểu tượng của nút **Print** trên thanh công cụ.

Bên cạnh đó, bạn có thể chèn các ghi chú trong mô hình quan hệ dữ liệu bằng cách chọn chức năng **New Text Annotation** trong thực đơn tắt.

Tóm lại, sau khi tạo xong cấu trúc bảng dữ liệu và thiết lập mô hình quan hệ dữ liệu, công việc kế tiếp mà bạn cần thực hiện trước khi đi tiếp các phần sau là việc nhập dữ liệu mẫu vào cho các bảng. Để làm việc này nhanh nhất, bạn chọn chức năng **Open Table (Return all rows)** để nhập dữ liệu trực tiếp vào bảng. Nguyên tắc trong khi nhập dữ liệu là luôn luôn nhập dữ liệu của bảng bên quan hệ nhánh 1 trước tiên và nhánh N sau đó.



liệu cho các bảng trong ứng dụng theo thứ tự sau : VATTU, NHACC, DONDH và

ấn dữ liệu – Phần 2

ợc: 1297-Ngày đăng: 24-07-2009-Ngày sửa: 24-07-2009]

Với cú pháp SELECT FROM bên dưới kết hợp mệnh đề GROUP BY cho phép bạn có thể nhóm dữ liệu của các dòng bên trong một bảng và được phép sử dụng các hàm thống kê đi kèm theo để tính toán các dữ liệu có tính chất thống kê tổng hợp. Thông thường, sau khi nhóm dữ liệu, bạn nên sắp xếp lại dữ liệu để hiển thị theo một thứ tự nào đó. Do vậy bạn sẽ sử dụng mệnh đề ORDER BY sau mệnh đề GROUP BY.

1.4/- Mệnh đề nhóm dữ liệu :

Cú pháp :

```
SELECT Danh_sách_các_cột | Hàm_thống_kê AS Bí_danh
FROM Tên_bảng
[WHERE Điều_kiện_lọc]
GROUP BY Danh_sách_cột_nhómdl
[ORDER BY Tên_cột [DESC] [, ...]]
```

Trong đó :

- **Hàm thống kê** : là tên của các hàm thống kê và các tham số tương ứng dùng để tính tổng (SUM), tính giá trị thấp nhất (MIN), tính giá trị cao nhất (MAX), đếm các mẫu tin (COUNT), tính giá trị trung bình (AVG) của các dữ liệu bên trong bảng.
- **Bí danh** : là tiêu đề mới của các cột tính toán. Các tiêu đề này chỉ có hiệu lực lúc hiển thị dữ liệu trong câu lệnh truy vấn mà không làm ảnh hưởng đến cấu trúc bên trong của bảng.
- **Danh sách cột nhóm dữ liệu** : là danh sách tên các cột được nhóm dữ liệu để tính toán.

Ví dụ :

Để thống kê tổng số đơn đặt hàng mà công ty đã đặt hàng theo từng nhà cung cấp và sắp xếp dữ liệu hiển thị theo thứ tự tổng số đơn đặt hàng tăng dần. Bạn thực hiện câu lệnh SELECT FROM như sau :

```
SELECT MANHACC, COUNT(*) AS TONG_SODH
FROM DONDH
GROUP BY MANHACC
ORDER BY TONG_SODH
```

Kết quả truy vấn trả về :

```
MANHACC    TONG_SODH
-----
C01         1
C03         1
C05         2
C02         3

(4 row(s) affected)
```

1.5/- Mệnh đề lọc dữ liệu sau khi đã nhóm :

Với cú pháp SELECT FROM bên dưới kết hợp mệnh đề HAVING cho phép bạn có thể lọc lại dữ liệu sau khi đã nhóm dữ liệu của các dòng bên trong một bảng. Khác với mệnh đề WHERE dùng để lọc các dòng dữ liệu hiện đang có bên trong bảng, mệnh đề HAVING chỉ được phép sử dụng đi kèm theo mệnh đề GROUP BY dùng để lọc lại dữ liệu sau khi đã nhóm. Điều này có nghĩa là mệnh đề HAVING chỉ được dùng kèm với mệnh đề GROUP BY.

Cú pháp :

```
SELECT Danh_sách_các_cột | Hàm_thống_kê AS Bí_danh
FROM Tên_bảng
[WHERE Điều_kiện_lọc]
GROUP BY Danh_sách_cột_nhómdl
HAVING Điều_kiện_lọc_nhóm
[ORDER BY Tên_cột [DESC] [, ...]]
```

Trong đó :

- **Điều kiện lọc nhóm** : là điều kiện dùng để lọc lại dữ liệu sau khi đã nhóm dữ liệu. Thông thường là các biểu thức luận lý.

Ví dụ :

Theo ví dụ trên nhưng bạn chỉ cần lọc ra những nhà cung cấp có mã bắt đầu bằng chữ "C" và tổng số các đơn đặt hàng lớn hơn 1 sau khi đã tính toán dữ liệu theo nhóm. Bạn thực hiện câu lệnh SELECT FROM như sau :

```
SELECT MANHACC, COUNT(*) AS TONG_SODH
FROM DONDH
WHERE MANHACC LIKE "C%"
GROUP BY MANHACC
HAVING COUNT(*)>1
ORDER BY TONG_SODH
```

Kết quả truy vấn trả về :

```
MANHACC  TONG_SODH
-----
C05      2
C02      3

(2 row(s) affected)
```

Trong ví dụ này bạn thấy rằng việc lọc dữ liệu được chia ra ở hai mệnh đề khác nhau. Thứ nhất mệnh đề WHERE MANHACC LIKE "C%" dùng để lọc ra các mẫu tin trong bảng DONDH sao cho mã nhà cung cấp phải bắt đầu bằng chữ "C", thứ hai mệnh đề HAVING COUNT(*)>1 dùng để lọc lại các nhà cung cấp nào có tổng số các đơn đặt hàng lớn hơn 1 sau khi đã nhóm để tính ra tổng số các đơn đặt hàng theo từng nhà cung cấp.

1.6/- Mệnh đề liên kết dữ liệu trong hai bảng :

Với cú pháp SELECT FROM bên dưới kết hợp mệnh đề JOIN cho phép bạn liên kết hai bảng có quan hệ với nhau để lấy ra các dữ liệu chung. Điểm quan trọng giữa những bảng này phải có các cột quan hệ chung nhau và thứ tự quan hệ khi bạn chỉ định giữa các bảng cũng sẽ làm ảnh hưởng đến kết quả của truy vấn.

Cú pháp :

```
SELECT Danh_sách_các_cột
FROM Tên_bảng
INNER {LEFT | RIGHT | FULL [OUTER]}
JOIN Tên_bảng_quan_hệ ON Điều_kiện_quan_hệ
```

Trong đó :

- **Từ khóa INNER JOIN** : dùng để chỉ định việc so sánh giá trị trong các cột của các bảng là tương đương (dữ liệu đều có ở cả hai bảng). Hệ thống sẽ trả về các mẫu tin thỏa điều kiện quan hệ ở cả hai bảng.

- **Từ khóa LEFT RIGHT FULL** : dùng để chỉ định việc so sánh giá trị các cột của bảng được ưu tiên cho mối quan hệ bên nhánh trái, phải hoặc cả hai bên. Việc thay đổi thứ tự ưu tiên này sẽ làm ảnh hưởng đến kết quả truy vấn.
- **Từ khóa OUTER** : được dùng kết hợp cho các quan hệ ưu tiên dữ liệu. Tuy nhiên bạn được phép bỏ đi khi sử dụng loại quan hệ ưu tiên LEFT RIGHT FULL.
- **Điều kiện quan hệ** : là một biểu thức so sánh bằng, chỉ ra tên các cột quan hệ giữa hai bảng gần giống như biểu thức sau mệnh đề WHERE dùng để liên kết hai bảng.

Ví dụ :

Để hiển thị thông tin của các đơn đặt hàng trong bảng DONDH kèm theo cột họ tên nhà cung cấp tương ứng trong bảng NHACC và sắp xếp dữ liệu theo cột mã nhà cung cấp tăng dần. Bạn thực hiện câu lệnh SELECT FROM như sau :

```
SELECT NCC.MANHACC, TENNHACC, SODH
FROM DONDH DH INNER JOIN NHACC NCC
                ON DH.MANHACC = NCC.MANHACC
ORDER BY NCC.MANHACC
```

Kết quả truy vấn trả về :

| MANHACC | TENNHACC | SODH |
|---------|--------------------|------|
| C01 | Lê Minh Trí | D002 |
| C02 | Trần Minh Thạch | D003 |
| C02 | Trần Minh Thạch | D005 |
| C02 | Trần Minh Thạch | D007 |
| C03 | Nguyễn Hồng Phương | D001 |
| C05 | Lưu Nguyệt Quế | D004 |
| C05 | Lưu Nguyệt Quế | D006 |

(7 row(s) affected)

Lưu ý :

Trong các truy vấn lấy dữ liệu từ nhiều bảng có quan hệ, bạn phải bắt buộc sử dụng định dạng : tên_bảng.tên_cột cho các cột trùng tên giữa các bảng. Theo ví dụ trên thì cột MANHACC xuất hiện ở cả hai bảng DONDH và NHACC do vậy bạn phải ghi : NCC.MANHACC và DH.MANHACC.

Ngoài ra bạn cũng có thể sử dụng khái niệm bí danh cho các bảng nhằm để làm ngắn gọn câu lệnh mỗi khi tham chiếu đến tên các bảng. Theo ví dụ trên bảng DONDH có bí danh là DH, bảng NHACC có bí danh là

NCC. Các bí danh này bạn có thể đặt tên tùy thích, tuy nhiên bạn nên đặt ngắn gọn, gợi nhớ và không được phép trùng nhau bên trong một câu lệnh truy vấn.

Ví dụ :

Giống như ví dụ trên nhưng yêu cầu hiển thị ra tất cả các nhà cung cấp hiện có trong bảng NHACC. Để làm được điều này, bạn thấy rằng thứ tự quan hệ phải ưu tiên dữ liệu bên bảng NHACC. Bạn thực hiện câu lệnh SELECT FROM như sau :

```
SELECT NCC.MANHACC, TENNHACC, SODH
      FROM DONDH DH RIGHT OUTER JOIN NHACC NCC
                                ON DH.MANHACC=NCC.MANHACC
      ORDER BY NCC.MANHACC
```

Kết quả truy vấn trả về :

| MANHACC | TENNHACC | SODH |
|------------|--------------------------|-------------|
| C01 | Lê Minh Trí | D002 |
| C02 | Trần Minh Thạch | D003 |
| C02 | Trần Minh Thạch | D005 |
| C02 | Trần Minh Thạch | D007 |
| C03 | Nguyễn Hồng Phương | D001 |
| C04 | Trương Nhật Thăng | NULL |
| C05 | Lưu Nguyệt Quế | D004 |
| C05 | Lưu Nguyệt Quế | D006 |
| C07 | Cao Minh Trung | NULL |

(9 row(s) affected)

Bạn thấy rằng có thêm hai nhà cung cấp mới trong kết quả truy vấn sau khi thay đổi thứ tự ưu tiên quan hệ dữ liệu cho bảng NHACC (RIGHT JOIN bởi vì bảng NHACC nằm bên phải trong quan hệ của bảng DONDH và NHACC). Tuy nhiên giá trị dữ liệu tại cột số đơn đặt hàng của hai nhà cung cấp này là NULL bởi vì công ty chưa bao giờ đặt hàng các nhà cung cấp này.

Ví dụ :

Hoàn toàn giống như ví dụ trên nhưng lần này bạn sẽ sử dụng thứ tự ưu tiên quan hệ dữ liệu bên trái. Bạn thực hiện câu lệnh SELECT FROM như sau :



```
SELECT NCC.MANHACC, TENNHACC, SODH
      FROM NHACC NCC LEFT JOIN DONDH DH
                                ON DH.MANHACC=NCC.MANHACC
      ORDER BY NCC.MANHACC
```



ủa truy vấn trả về có gì khác so với ví dụ ở trên khi thay đổi thứ tự ưu tiên quan hệ dữ liệu (JOIN) hay không? Kết quả của hai câu lệnh truy vấn hoàn toàn như nhau. Tuy nhiên khi sử dụng quan hệ dữ liệu bên trái trong ví dụ này đã thay đổi tên bảng NHACC ngay phía sau mệnh lệnh chỉ định thứ tự ưu tiên lấy dữ liệu bên bảng NHACC.

Trong thực tế việc chọn lựa để sử dụng mệnh đề LEFT JOIN hoặc RIGHT JOIN là không quan trọng mà thay vào đó bạn phải hiểu rằng dữ liệu mà bạn cần chọn ra phải ưu tiên nằm bên trong bảng nào. Thông thường có một quy định là bảng nào ưu tiên dữ liệu sẽ được ghi ngay sau mệnh đề FROM, kế tiếp sử dụng mệnh đề LEFT JOIN chỉ định tên của bảng quan hệ cần lấy thông tin.

Kiểu dữ liệu do người dùng định nghĩa

[Share](#)   + [Đọc: 486-Ngày đăng: 16-06-2009-Ngày sửa: 18-06-2009]

Giống như những ngôn ngữ lập trình khác, Microsoft SQL Server cho phép bạn có thể định nghĩa ra các kiểu dữ liệu mới dựa trên các kiểu dữ liệu cơ sở của Microsoft SQL Server.

1/- Định nghĩa :

Các kiểu dữ liệu mới này được gọi là kiểu dữ liệu do người dùng định nghĩa. Mục đích của việc tạo ra các kiểu dữ liệu do người dùng định nghĩa là để đảm bảo cho cấu trúc dữ liệu bên trong cơ sở dữ liệu được nhất quán và dễ sửa đổi.

Ví dụ :

Trong một cơ sở dữ liệu gồm có các bảng : nhân viên, khách hàng, nhà cung cấp. Trong từng bảng này lại có các cột mà dữ liệu của chúng lưu trữ hoàn toàn giống nhau như là : họ, tên, địa chỉ, số điện thoại. Để kiểu dữ liệu và độ dài lưu trữ ở các cột trong mỗi bảng là giống nhau bắt buộc bạn phải ngâm nhớ, cách làm việc này không khoa học lắm. Ngoài ra về sau khi có nhu cầu sửa đổi cấu trúc các bảng sẽ dễ dàng làm mất đi tính nhất quán về cấu trúc giữa các bảng trong cùng ứng dụng.

Bên trong Microsoft SQL Server để có thể khắc phục được yếu điểm này, bạn có thể định nghĩa ra các kiểu dữ liệu mới và sau đó khi tạo cấu trúc bảng mới hoặc sửa đổi cấu trúc bảng thì phải chỉ rõ ra kiểu dữ liệu của các cột bên trong bảng là những kiểu mới vừa định nghĩa này.

Ví dụ :

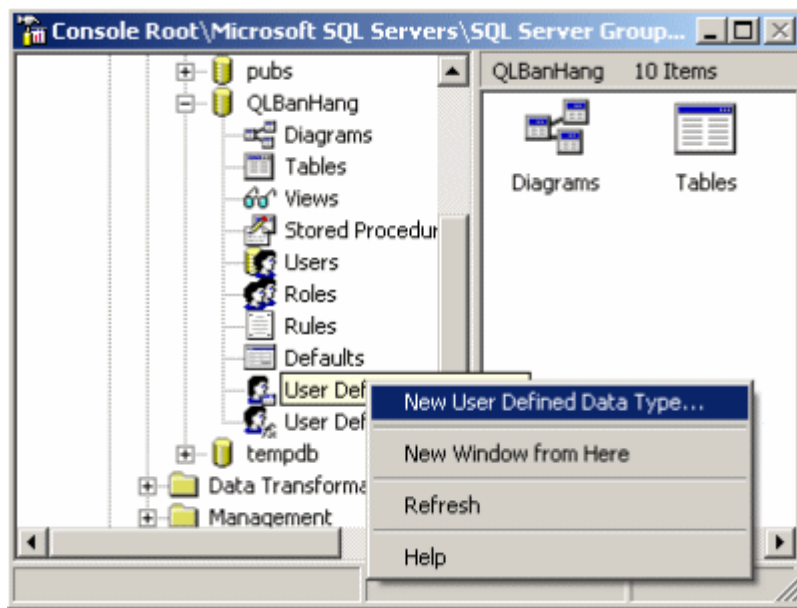
Tiếp theo ví dụ trên bạn sẽ lần lượt tạo ra các kiểu dữ liệu mới như là : kiểu dữ liệu họ, kiểu dữ liệu tên, kiểu dữ liệu địa chỉ ... để chỉ định cho các cột họ, tên, địa chỉ trong các bảng dữ liệu : nhân viên, khách hàng, nhà cung cấp.

2/- Tạo mới kiểu dữ liệu do người dùng định nghĩa :

Giống như việc tạo mới các đối tượng khác mà bạn đã làm quen trước đây trong Microsoft SQL Server, bạn có hai cách để có thể tạo mới kiểu dữ liệu do người dùng định nghĩa. Các bước bên dưới sẽ hướng dẫn bạn cách thức tạo ra kiểu dữ liệu mới do người dùng định nghĩa bằng tiện ích Enterprise Manager.

Bước 1 :

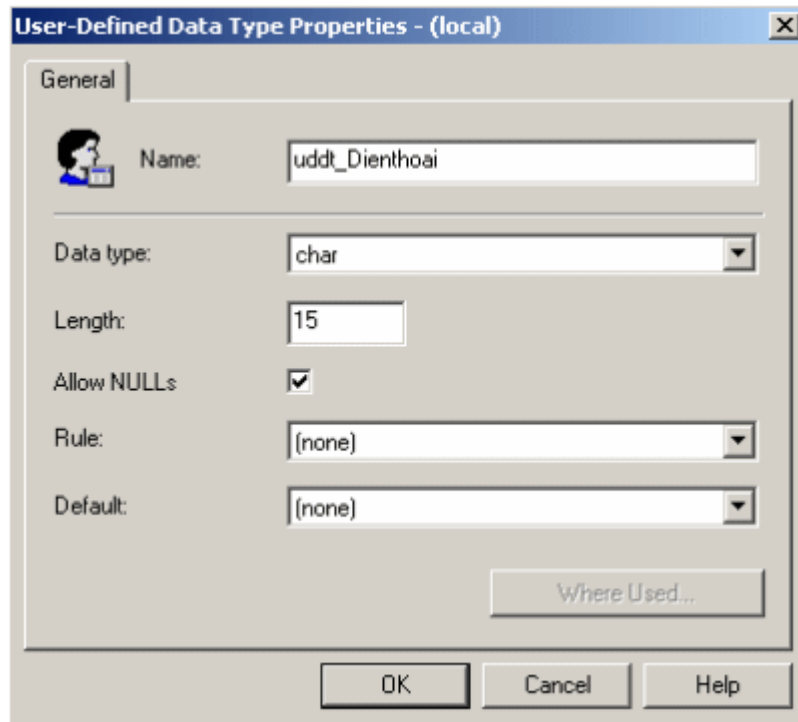
Khởi động tiện ích Enterprise Manager, chọn chức năng **New User Defined Data Type ...** trong thực đơn tắt sau khi nhấn chuột phải trên đối tượng **User Defined Data Types** để tạo mới kiểu dữ liệu do người dùng định nghĩa dùng chung cho các bảng trong cơ sở dữ liệu.



Chọn New User Defined Data Type

Bước 1 :

Trong màn hình định nghĩa kiểu dữ liệu mới lần lượt chỉ định các thuộc tính liên quan đến kiểu dữ liệu mới bao gồm : tên của kiểu dữ liệu mới, kiểu dữ liệu cơ sở, chiều dài dữ liệu, có cho phép bỏ trống dữ liệu khi thêm mới không. Sau cùng nhấn **OK** để lưu lại kiểu dữ liệu mới định nghĩa.



Các thuộc tính liên quan đến kiểu dữ liệu mới

Ngoài ra bạn cũng có thể tạo mới kiểu dữ liệu do người dùng định nghĩa bằng thủ tục nội tại hệ thống `sp_addtype` khi gõ lệnh trong cửa sổ **Query Analyzer**.

Cú pháp :

```
EXEC sp_addtype Tên_kiểu_dl_mới, 'Kiểu_dữ_liệu_cơ_sở'  
[,NULL | NOT NULL]
```

Trong đó :

- **Tên kiểu dữ liệu mới** : tên kiểu dữ liệu do người dùng định nghĩa, tên kiểu dữ liệu mới phải duy nhất trong một cơ sở dữ liệu.
- **Kiểu dữ liệu cơ sở** : tên của các kiểu dữ liệu hiện có trong Microsoft SQL Server. Thông thường đối với các kiểu dữ liệu có chỉ định độ rộng dữ liệu thì bắt buộc phải có mở và đóng nháy.
- **NULL I NOT NULL** : có cho phép hoặc không cho phép cột bỏ trống dữ liệu khi lưu trữ. Mặc định cho phép cột bỏ trống dữ liệu khi thêm mới dữ liệu.

Ví dụ :

Để định nghĩa kiểu dữ liệu có tên `uddt_Soluong` có kiểu dữ liệu cơ sở là decimal lưu trữ tối đa 15 ký số trong đó có 2 số lẻ, không cho phép cột để trống dữ liệu khi lưu trữ. Bạn sẽ thực hiện câu lệnh như sau :

```
EXEC sp_addtype uddt_Soluong, 'Decimal(15,2)', 'NOT NULL'
```

Ví dụ :

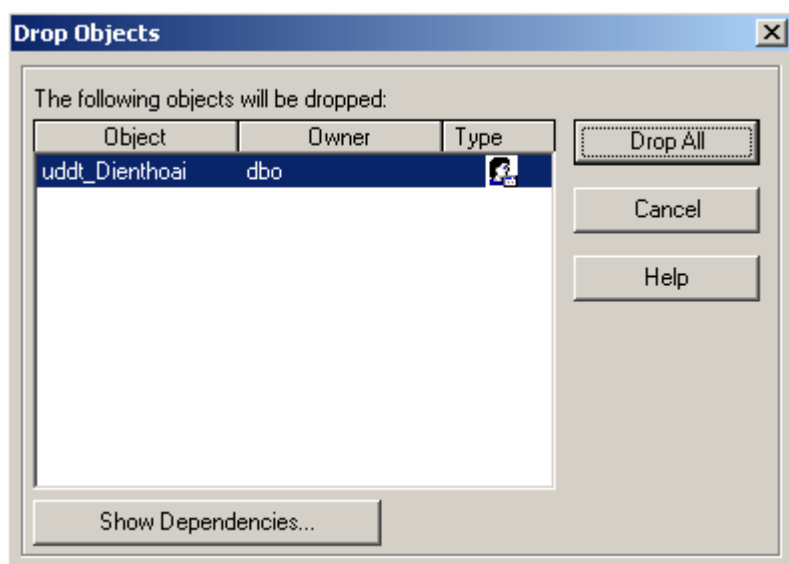
Sau đó khi tạo lập cấu trúc bảng TONKHO (tồn kho) trong đó có các cột số lượng đầu kỳ, tổng số lượng nhập, tổng số lượng xuất và số lượng cuối kỳ. Bạn sẽ sử dụng kiểu dữ liệu mới vừa định nghĩa ở trên cho các cột này bởi vì chúng có cùng chung một kiểu dữ liệu là kiểu số lượng (uddt_Soluong).

```
CREATE TABLE TONKHO
(
    Namthang      CHAR(7),
    Mavtu         CHAR(4),
    Sldk          uddt_Soluong,
    TSNhap       uddt_Soluong,
    TSIXuat      uddt_Soluong,
    Slick        uddt_Soluong
    CONSTRAINT PRK_TONKHO_NAMTHANG_MAVTU
        PRIMARY KEY (Namthang, Mavtu),
    CONSTRAINT FRK_TONKHO_MAVTU
        FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)
```

3/- Xóa kiểu dữ liệu do người dùng định nghĩa :

Khi một kiểu dữ liệu do người dùng định nghĩa trong cơ sở dữ liệu không còn dùng đến nữa, bạn có thể hủy bỏ nó đi. Tuy nhiên nếu một kiểu dữ liệu do người dùng định nghĩa còn được sử dụng ít nhất cho một cột bên trong một bảng nào đó thì bạn sẽ không thể nào hủy được nó.

Để hủy một kiểu dữ liệu do người dùng định nghĩa, bạn chọn chức năng **Delete** trên thực đơn tắt sau khi nhấn chuột phải vào tên kiểu dữ liệu do người dùng định nghĩa muốn hủy bỏ trong tiện ích Enterprise Manager và xác nhận đồng ý hủy bằng cách chọn nút **Drop All** trong màn hình hủy bỏ các đối tượng bên trong cơ sở dữ liệu của Microsoft SQL Server.



Xác nhận hủy bỏ kiểu dữ liệu mới

Ngoài ra bạn cũng có thể sử dụng thủ tục nội tại hệ thống có tên `sp_droptype` để hủy bỏ kiểu dữ liệu do người dùng định nghĩa trong cửa sổ **Query Analyzer**.

Cú pháp :

```
EXEC sp_droptype Tên_kiểu_dl
```

Trong đó :

- **Tên kiểu dữ liệu** : là tên kiểu dữ liệu do người dùng định nghĩa có trong cơ sở dữ liệu hiện hành muốn hủy bỏ (không còn dùng nữa trong cơ sở dữ liệu).

Ví dụ :

Hủy kiểu dữ liệu `uddt_Soluong` trong cơ sở dữ liệu `QLBanHang`.


```
EXEC sp_droptype uddt_Soluong
```

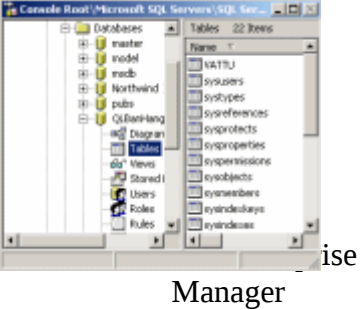
Tuy nhiên khi đó hệ thống sẽ xuất hiện thông báo lỗi bởi vì các cột số lượng trong bảng `TONKHO` đang sử dụng kiểu dữ liệu này nên không thể hủy được.

```
Server: Msg 15180, Level 16, State 1, Line 0  
Cannot drop. The data type is being used.
```

Hết

Thay đổi cấu trúc bảng

[Share](#)   + [Đọc: 554-Ngày đăng: 09-06-2009-Ngày sửa: 09-06-2009]



Trong thực tế việc thay đổi cấu trúc bảng vẫn thường được thực hiện đối với các bảng đã có chứa nhiều dữ liệu. Trong trường hợp này bạn tuyệt đối không hủy bỏ bảng hiện có và tạo lại cấu trúc mới bởi vì làm như thế là bạn sẽ mất đi tất cả các dữ liệu đang hiện có bên trong bảng.

Bạn có thể thay đổi cấu trúc bảng bằng tiện ích **Enterprise Manager** trong màn hình thiết kế bảng. Tuy nhiên, trong bài này sẽ hướng dẫn bạn các tính năng mà câu lệnh **ALTER TABLE** sẽ mang đến khi bạn thực hiện việc thay đổi cấu trúc bảng.

1/- Thêm một cột mới trong bảng :

Với cú pháp **ALTER TABLE** bên dưới cho phép bạn thêm vào một hoặc nhiều cột mới vào trong bảng hiện đang có. Tên các cột mới phải khác tên các cột hiện đang có bên trong bảng. Mặc nhiên dữ liệu của các cột mới thêm vào phải được phép bỏ trống.

Cú pháp :

```
ALTER TABLE Tên_bảng  
ADD Tên_cột Kiểu_dữ_liệu [, ...]
```

Trong đó :

- Tên cột : tên của cột mới được thêm vào bảng.
- Kiểu dữ liệu : kiểu dữ liệu tương ứng của cột mới.

Ví dụ :

Để thêm vào bảng DONDH (đơn đặt hàng) một cột ngày dự kiến nhận hàng có kiểu dữ liệu là ngày. Bạn thực hiện câu lệnh **ALTER TABLE** như sau :

```
ALTER TABLE DONDH  
ADD Ngaydtkh DATETIME
```

2/- Hủy bỏ cột hiện có bên trong bảng :

Với cú pháp **ALTER TABLE** bên dưới cho phép bạn hủy bỏ một hoặc nhiều cột hiện có bên trong bảng. Nên nhớ việc hủy bỏ các cột đồng nghĩa với các dữ liệu mà cột hiện đang lưu trữ cũng sẽ bị mất theo. Do đó cần thận trọng khi sử dụng câu lệnh này :

Cú pháp :

```
ALTER TABLE Tên_bảng  
DROP COLUMN Tên_cột [, ...]
```

Trong đó :

- **Tên cột** : tên cột sẽ bị hủy bỏ ra khỏi bảng.

Ví dụ :

Để hủy bỏ cột ngày dự kiến nhận hàng trong bảng DONDH (đơn đặt hàng) vừa thêm ở ví dụ trên, bạn thực hiện câu lệnh **ALTER TABLE** như sau :

```
ALTER TABLE DONDH
DROP COLUMN Ngaydknh
```

Lưu ý :

Đối với những cột nào đã có định nghĩa các quy tắc kiểm tra toàn vẹn dữ liệu thông qua các constraint thì bạn không thể hủy bỏ. Trong các trường hợp này để có thể hủy bỏ các cột đó, trước tiên bạn phải hủy bỏ các constraint có liên quan đến chúng.

3/- Sửa đổi kiểu dữ liệu của cột :

Với cú pháp **ALTER TABLE** bên dưới cho phép bạn sửa đổi kiểu dữ liệu của cột hiện có trong bảng. Đối với các kiểu dữ liệu dạng text, ntext hoặc image thì không thể đổi sang kiểu dữ liệu khác. Thông thường bạn rất ít khi sửa đổi kiểu dữ liệu của các cột một cách trực tiếp mà thay vào đó bạn thêm một cột mới với kiểu dữ liệu như mong muốn vào bảng, sau đó cập nhật dữ liệu của cột hiện có sang cột mới vừa thêm vào, kiểm tra việc dữ liệu sau khi cập nhật có đúng theo như mong muốn không. Cuối cùng là hủy bỏ cột dữ liệu cũ và đổi tên cột dữ liệu mới thành tên của cột dữ liệu cũ trước đó.

Cú pháp :

```
ALTER TABLE Tên_bảng
ALTER COLUMN Tên_cột Kiểu_dữ_liệu_mới
```

Ví dụ :

Để sửa lại kiểu dữ liệu và tăng độ dài lưu trữ của cột đơn vị tính lên 20 ký tự trong bảng VATTU (vật tư), bạn thực hiện câu lệnh **ALTER TABLE** như sau :

```
ALTER TABLE VATTU
ALTER COLUMN Dvtinh VARCHAR(20)
```

Lưu ý :

Giống như việc hủy bỏ các cột, với những cột nào đã có định nghĩa các quy tắc kiểm tra toàn vẹn dữ liệu thông qua các constraint thì bạn không thể thay đổi kiểu dữ liệu của chúng.

4/- Tắt bỏ quy tắc kiểm tra toàn vẹn dữ liệu :

Với cú pháp **ALTER TABLE** bên dưới cho phép bạn tạm thời bỏ qua việc kiểm tra toàn vẹn dữ liệu thông qua các đối tượng constraint bên trong bảng bằng cách tắt các quy tắc. Các quy tắc mà bạn chỉ định sẽ không còn được áp dụng dùng để kiểm tra tính toàn vẹn dữ liệu cho đến khi nào bạn bật cho nó hoạt động trở lại. Tuy nhiên các quy tắc này sẽ hoàn toàn không bị xóa đi ra khỏi cơ sở dữ liệu.

Cú pháp :

```
ALTER TABLE Tên_bảng  
NOCHECK CONSTRAINT ALL | Tên_constraint [, ...]
```

Trong đó :

- **Tên constraint** : tên của các constraint phải có tồn tại bên trong cơ sở dữ liệu mà bạn muốn tạm ngưng việc kiểm tra tính toàn vẹn dữ liệu trên đó.
- **Từ khóa ALL** : được sử dụng khi bạn muốn tạm ngưng việc kiểm tra tính toàn vẹn dữ liệu cho tất cả các constraint có liên quan đến bảng đã được định nghĩa trước đó.

Lưu ý :

Bạn tạm thời tắt việc kiểm tra dữ liệu trong cột số lượng đặt hàng của bảng CTDONDH (chi tiết đặt hàng) nằm trong miền giá trị từ 10 đến 50 khi người dùng thêm hoặc sửa dữ liệu trong bảng CTDONDH, bạn thực hiện câu lệnh **ALTER TABLE** như sau :

```
ALTER TABLE CTDONDH  
NOCHECK CONSTRAINT CHK_CTDONDH_SLDAT
```

Lưu ý :

Bạn chỉ được phép tắt các quy tắc kiểm tra toàn vẹn dữ liệu của các constraint là khóa ngoại (**FOREIGN KEY**), miền giá trị (**CHECK**) và giá trị mặc định (**DEFAULT**). Hai loại constraint còn lại là khóa chính (**PRIMARY KEY**) và duy nhất (**UNIQUE**) hệ thống Microsoft SQL Server hoàn toàn không cho phép tắt là bởi vì hệ thống cần phải kiểm tra tính duy nhất của dữ liệu bên trong bảng có định nghĩa khóa chính.

5/- Bật lại quy tắc kiểm tra toàn vẹn dữ liệu :

Với cú pháp **ALTER TABLE** bên dưới cho phép bạn bật lại việc áp dụng các quy tắc kiểm tra toàn vẹn dữ liệu thông qua các đối tượng constraint bên trong bảng sau khi bạn tạm thời đã tắt đi trước đó.

Cú pháp :

```
ALTER TABLE Tên_bảng  
CHECK CONSTRAINT ALL | Tên_constraint [, ...]
```

Trong đó :

&# 8226 Tên constraint : tên của các constraint phải tồn tại trong cơ sở dữ liệu mà bạn muốn bật trở lại sau khi đã tắt tạm thời.

&# 8226 Từ khóa ALL : được sử dụng khi muốn bật việc kiểm tra toàn vẹn dữ liệu cho toàn bộ tất cả các constraint liên quan đến bảng.

Ví dụ :

Bạn bật lại việc kiểm tra toàn vẹn dữ liệu trong cột số lượng đặt hàng của bảng CTDONDH (chi tiết đặt hàng) đã bị tắt tạm thời ở ví dụ trên, bạn thực hiện lệnh **ALTER TABLE** như sau :

```
ALTER TABLE CTDONDH  
CHECK CONSTRAINT CHK_CTDONDH_SLDAT
```

6/- **Đổi tên cột, tên bảng dữ liệu :**

Trong thực tế, việc thay đổi tên cột hoặc tên bảng dữ liệu là rất hạn chế bởi vì chúng sẽ làm ảnh hưởng rất nhiều đến các đoạn chương trình có tham chiếu đến tên cột hoặc tên bảng ở đâu đó trong ứng dụng. Bạn chỉ thực hiện việc này khi thật sự cần thiết.

Để có thể đổi tên cột hoặc tên bảng, bạn có thể vào trực tiếp tiện ích **Enterprise Manager**. Tuy nhiên, bạn cần biết thêm về thủ tục nội tại hệ thống (system stored procedure) có tên là `sp_rename`.

Các thủ tục nội tại hệ thống là một tập hợp các thủ tục do Microsoft SQL Server tạo ra và cung cấp cho người sử dụng thực hiện một số xử lý sẵn có bên trong Microsoft SQL Server.

Cú pháp :

```
EXEC sp_rename 'Tên_bảng[.Tên_cột],Tên_mới'  
[,'COLUMN']
```

Trong đó :

- **EXE** : lệnh dùng để thực thi (execute) các thủ tục nội tại bên trong Microsoft SQL Server.
- **Tên bảng** : tên của bảng mà bạn sẽ đổi tên hoặc tên bảng chứa tên cột mà bạn muốn đổi tên.

- **Tên cột** : tên cột mà bạn muốn đổi tên. Khi muốn đổi tên một cột trong bảng thì bạn phải chỉ ra đầy đủ tên bảng chứa tên cột muốn đổi tên.
- **Tên mới** : tên mới của cột hoặc tên mới của bảng sau khi đổi.
- **COLUMN** : từ khóa chỉ được sử dụng khi thay đổi tên cột.

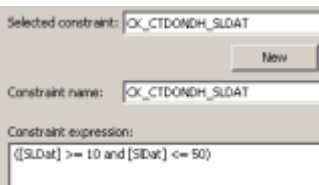
Ví dụ :

Để thay đổi tên cột họ tên nhà cung cấp trong bảng NHACC (nhà cung cấp) từ cột cũ Tennhacc thành cột mới là Hotenncc, bạn thực hiện câu lệnh như sau :

```
EXEC sp_rename 'NHACC.Tennhacc','Hotenncc'
        'COLUMN'
```

Để thay đổi tên bảng hiện có của NHACC thành tên mới là NHACCAP, bạn thực hiện câu lệnh như sau :

```
EXEC sp_rename 'NHACC','NHACCAP'
```



100 câu hỏi về dữ liệu trong cơ sở dữ liệu - Phần 1

[Share](#) + [Đọc: 861-Ngày đăng: 29-05-2009-Ngày sửa: 29-05-2009]

Toàn vẹn dữ liệu là việc đặt ra các quy tắc trong một cơ sở dữ liệu nhằm kiểm tra thuộc tính bảng các giá trị của dữ liệu trước khi được lưu trữ phải đảm bảo tính chính xác và hợp lý bên trong một cơ sở dữ liệu.

Nếu các giá trị dữ liệu nào vi phạm các quy tắc đặt ra thì các dữ liệu đó sẽ không được lưu vào bảng. Các quy tắc này phần lớn được xây dựng dựa vào các quy tắc hiện hữu đang tồn tại trong công việc kinh doanh.

Ví dụ :

Trong cơ sở dữ liệu QLBanHang (quản lý bán hàng) bạn có thể đưa ra một số quy tắc như sau :

- Số lượng đặt hàng phải dương (miễn giá trị).
- Ngày nhập hàng phải sau ngày đặt hàng.
- Số hóa đơn giao hàng không bị cấp trùng số (khóa chính).
- Đơn đặt hàng phải gởi cho một nhà cung cấp có trong danh sách nhà cung cấp (khóa ngoại).
- Số lượng bán hàng phải dương và còn đủ số lượng hiện có trong kho hàng.

Điều gì xảy ra nếu dữ liệu bên trong các bảng có vi phạm các quy tắc ở trên mà vẫn được lưu trữ vào bên trong cơ sở dữ liệu? Do đó, trong các hệ cơ sở dữ liệu quan hệ lớn ngày nay, hệ thống sẽ giúp cho người sử dụng có thể định nghĩa ra các quy tắc thông qua các từ khóa hoặc thuộc tính có liên quan đến cơ sở dữ liệu nhằm đảm bảo dữ liệu khi lưu trữ vào cơ sở dữ liệu phải chính xác và hợp lý.

Bên trong cơ sở dữ liệu của Microsoft SQL Server việc kiểm tra tính toàn vẹn dữ liệu sẽ thông qua hai đối tượng quản lý, đó là các **constraint** và **trigger**. Cả hai đối tượng này đều được liên kết trực tiếp vào bảng dữ liệu.

Các loại quy tắc kiểm tra tính toàn vẹn dữ liệu bao gồm :

1/- Kiểm tra duy nhất dữ liệu :

Loại ràng buộc toàn vẹn này cho phép bạn có thể kiểm tra tính duy nhất của dữ liệu bên trong bảng. Điều này ngăn cản việc người sử dụng tình cờ nhập trùng lại các giá trị dữ liệu bên trong bảng. Bạn có thể sử dụng hai thành phần **PRIMARY KEY** hoặc **UNIQUE** trong câu lệnh **CREATE TABLE** để thực hiện việc kiểm tra tính duy nhất của dữ liệu.

Điểm khác biệt chính giữa **PRIMARY KEY** và **UNIQUE** là sự xuất hiện của các thành phần trong câu lệnh **CREATE TABLE**. Thành phần **PRIMARY KEY** cho phép bạn tạo ra cấu trúc bảng có chứa khóa chính. Do một bảng chỉ có một khóa chính, tuy nhiên khóa chính được phép định nghĩa có nhiều cột tham gia. Vì thế thành phần **PRIMARY KEY** chỉ xuất hiện một lần duy nhất khi tạo cấu trúc bảng.

Thành phần **UNIQUE** cho phép bạn kiểm tra tính duy nhất của các cột không tham gia làm khóa chính của bảng. Thành phần **UNIQUE** được phép xuất hiện nhiều lần khi tạo cấu trúc bảng nếu cần kiểm tra tính duy nhất của các cột không làm khóa chính.

Cú pháp :

```
[ CONSTRAINT Tên_constraint ]  
PRIMARY KEY (Danh_sách_cột_khóa_chính)
```

Trong đó :

- **Tên constraint** : phải là duy nhất trong cơ sở dữ liệu. Thông thường quy định tên constraint gồm có 3 phần. Bắt đầu bằng các chữ PRK, kế tiếp là tên bảng và cuối cùng là tên cột áp dụng quy tắc kiểm tra duy nhất dữ liệu của các cột khóa chính.
- **Danh sách cột khóa chính** : là danh sách tên các cột tham gia làm khóa chính, tên các cột được ngăn cách nhau bởi dấu phẩy (,).

Cú pháp :

```
[ CONSTRAINT Tên_constraint ]  
    UNIQUE (Danh_sách_các_cột)
```

Trong đó :

- **Tên constraint** : phải là duy nhất trong cơ sở dữ liệu. Thông thường quy định tên constraint gồm có 3 phần. Bắt đầu bằng các chữ UNQ, kế tiếp là tên bảng và cuối cùng là tên cột áp dụng quy tắc kiểm tra duy nhất dữ liệu của các cột không tham gia làm khóa chính.
- **Danh sách cột** : là danh sách tên các cột cần kiểm tra duy nhất, tên các cột được ngăn cách nhau bởi dấu phẩy (,).

Lưu ý :

Mặc định tên các constraint sẽ do hệ thống Microsoft SQL Server tạo ra nhằm đảm bảo tính duy nhất bên trong cơ sở dữ liệu. Tuy nhiên bạn có thể chủ động đặt tên cho các constraint khi tạo ra chúng bằng cách đưa thêm từ khóa **CONSTRAINT** trong các thành phần định nghĩa các kiểm tra ràng buộc toàn vẹn dữ liệu.

Ví dụ :

Tạo bảng VATTU kiểm tra dữ liệu của cột mã vật tư phải là duy nhất. Trường hợp chỉ định cột mã vật tư làm khóa chính của bảng. Bạn thực hiện câu lệnh **CREATE TABLE** như sau :

```
CREATE TABLE VATTU  
(  
    Mavtu    CHAR(4)  
            PRIMARY KEY (Mavtu),  
    Tenvtu  CHAR(30),  
    DyTinh  CHAR(10) NOT NULL,  
    Phantram TinyInt DEFAULT 20  
)
```

Với cú pháp như trên bảng VATTU có sử dụng một constraint loại **PRIMARY KEY** dùng định nghĩa khóa chính của bảng là cột Mavtu. Tuy nhiên tên của constraint sẽ do hệ thống Microsoft SQL Server tạo ra.

Trong trường hợp nếu bạn muốn chỉ định tên của constraint do bạn định nghĩa thì bạn thực hiện câu lệnh **CREATE TABLE** có sử dụng từ khóa **CONSTRAINT** như sau :

```

CREATE TABLE VATTU
(
    Mavtu    CHAR(4)    NOT NULL,
            CONSTRAINT PRK_VATTU_Mavtu
            PRIMARY KEY (Mavtu),
    Tenvtu   CHAR(30) ,
    DyTinh   CHAR(10) NOT NULL,
    Phantram  TinyInt  DEFAULT 20
)

```

Khi mà các thành phần của đối tượng constraint xuất hiện ngay phía sau tên cột như hai ví dụ ở trên thì ràng buộc toàn vẹn dữ liệu sẽ được kiểm tra trên cột dữ liệu. Bạn nên chọn cách này khi thực hiện việc kiểm tra các ràng buộc toàn vẹn dữ liệu chỉ trên một cột bên trong bảng dữ liệu.

Trong trường hợp nếu bạn đặt các thành phần của đối tượng constraint bên dưới tất cả các cột dữ liệu bên trong bảng thì ràng buộc toàn vẹn dữ liệu sẽ được kiểm tra trên bảng. Bạn nên chọn cách này khi thực hiện việc kiểm tra các ràng buộc toàn vẹn dữ liệu trên nhiều cột có liên quan bên trong bảng dữ liệu.

Ví dụ :

Để tạo bảng có tên CTDONDH (chi tiết đơn đặt hàng) gồm có những cột như : số đặt hàng có kiểu dữ liệu là chuỗi và chiều dài 4 ký tự, mã vật tư có kiểu dữ liệu là chuỗi và chiều dài 4 ký tự, số lượng đặt có kiểu số nguyên. Dữ liệu tại các cột không được phép trống. Khóa chính gồm có 2 cột là số đặt hàng và mã vật tư. Bạn thực hiện câu lệnh **CREATE TABLE** như sau :

```

CREATE TABLE CTDONDH
(
    Sodh     CHAR(4) ,
    Mavtu    CHAR(4) ,
    SIDat    SMALLINT
            PRIMARY KEY (Sodh, Mavtu)
)

```

Hoặc muốn đặt tên của constraint là PRK_CTDONDH_SodhID

```

CREATE TABLE CTDONDH
(
    Sodh     CHAR(4) ,
    Mavtu    CHAR(4) ,
    SIDat    SMALLINT
            CONSTRAINT PRK_CTDONDH_SodhID
            PRIMARY KEY (Sodh, Mavtu)
)

```


Lưu ý :

Khi sử dụng các loại ràng buộc toàn vẹn dữ liệu được kiểm tra trên cột dữ liệu thì cần nhớ rằng dấu phẩy (,) luôn được đặt ở vị trí sau cùng của thành phần constraint chứ không được đặt ở vị trí phía sau tên kiểu dữ liệu của cột.

Ngược lại khi sử dụng các loại ràng buộc toàn vẹn dữ liệu được kiểm tra trên bảng thì không cần có thêm dấu phẩy (,) ở vị trí phía sau của cột dữ liệu cuối cùng bên trong bảng.

Đôi khi những điều lưu ý này sẽ làm cho bạn cảm thấy khó nhớ. Do vậy để đơn giản khi tạo cấu trúc bảng, bạn thực hiện hai bước :

Bước 1 : Tạo cấu trúc bảng đơn giản bằng lệnh **CREATE TABLE** gồm có tên bảng, tên các cột và các kiểu dữ liệu mong muốn.

Bước 2 : Thêm các loại constraint tương ứng bằng lệnh **ALTER TABLE ADD CONSTRAINT** để áp dụng các kiểm tra ràng buộc toàn vẹn dữ liệu cho bảng dữ liệu.

2/- Kiểm tra tồn tại dữ liệu :

Loại ràng buộc toàn vẹn này cho phép bạn có thể kiểm tra tính tồn tại của dữ liệu (khóa ngoại), bắt buộc phải có bên một bảng khác, còn gọi là bảng tham chiếu. Điều này ngăn cản việc người sử dụng nhập một giá trị dữ liệu không có trong một bảng dữ liệu khác. Bạn có thể sử dụng thành phần **FOREIGN KEY** trong câu lệnh **CREATE TABLE** để thực hiện việc kiểm tra tính tồn tại của dữ liệu.

Trong một bảng được phép có nhiều khóa ngoại, tuy nhiên các cột tham chiếu phải được định nghĩa là khóa chính trong các bảng tham chiếu trước đó. Vì thế thành phần **FOREIGN KEY** được phép xuất hiện nhiều lần khi tạo cấu trúc bảng nếu bạn cần kiểm tra tính tồn tại của nhiều cột khác nhau trong các bảng khác nhau.

Cú pháp :

```
[ CONSTRAINT Tên_constraint ]
    FOREIGN KEY (Danh_sách_cột_khóa_ngoại)
    REFERENCES Tên_bảng_tham_chiếu
    (Danh_sách_cột_tham_chiếu)
```

Trong đó :

- **Tên constraint :** phải là duy nhất trong cơ sở dữ liệu. Thông thường quy định tên constraint gồm có 3 phần. Bắt đầu bằng các chữ : FRK, kế tiếp là tên bảng và cuối cùng là tên cột áp dụng quy tắc kiểm tra tồn tại dữ liệu.

- **Danh sách cột khóa ngoại** : là danh sách tên các cột tham gia làm khóa ngoại, tên các cột được ngăn cách nhau bởi dấu phẩy (,).
- **Tên bảng tham chiếu** : là tên của bảng tham chiếu để kiểm tra tính tồn tại dữ liệu bên bảng tham chiếu.
- **Danh sách cột tham chiếu** : là danh sách tên các cột tham chiếu trong bảng tham chiếu, tên các cột ngăn cách nhau bởi dấu phẩy (,). Lưu ý rằng danh sách các cột khóa ngoại và danh sách các cột tham chiếu phải tương xứng nhau.

Ví dụ :

Tạo lại bảng CTDONDH có bổ sung thêm hai ràng buộc toàn vẹn dữ liệu khóa ngoại : số đặt hàng tham chiếu qua bảng DONDH, nghĩa là giá trị dữ liệu tại cột số đặt hàng phải tồn tại trong bảng DONDH và mã vật tư tham chiếu qua bảng VATTU, nghĩa là giá trị dữ liệu tại cột mã vật tư phải tồn tại trong bảng VATTU. Bạn thực hiện câu lệnh **CREATE TABLE** như sau :

```
CREATE TABLE CTDONDH
(
    Sodh    CHAR(4),
    Mavtu   CHAR(4),
    SIDat   SMALLINT
    PRIMARY KEY (Sodh, Mavtu) ,
    FOREIGN KEY (Sodh) REFERENCES DONDH (Sodh) ,
    FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)
)
```

Hoặc muốn đặt tên cho các constraint liên quan

```
CREATE TABLE CTDONDH
(
    Sodh    CHAR(4),
    Mavtu   CHAR(4),
    SIDat   SMALLINT
    CONSTRAINT PRK_CTDONDH_Sodh
        PRIMARY KEY (Sodh, Mavtu) ,
    CONSTRAINT FRK_CTDONDH_Sodh
        FOREIGN KEY (Sodh) REFERENCES DONDH (Sodh) ,
    CONSTRAINT FRK_CTDONDH_Mavtu
        FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)
)
```

3/- **Kiểm tra miền giá trị** :

Loại ràng buộc toàn vẹn này cho phép bạn có thể kiểm tra miền giá trị của các cột dữ liệu bên trong bảng. Điều này ngăn cản việc người sử dụng nhập một giá trị dữ liệu vượt ra khỏi phạm vi mà bạn quy định trước đó. Bạn có thể sử dụng thành phần **CHECK** trong câu lệnh **CREATE TABLE** để thực hiện việc kiểm tra miền giá trị của dữ liệu.

Có thể thực hiện việc kiểm tra để so sánh giá trị của một cột bên trong bảng với một giá trị cụ thể hoặc một hàm tính toán của Microsoft SQL Server mà kiểu trả về của hàm phải khớp với kiểu dữ liệu của cột. Ngoài ra bạn cũng có thể kiểm tra giá trị của nhiều cột có liên quan trong cùng một bảng dữ liệu.

Cú pháp :

```
[ CONSTRAINT Tên_constraint ]  
CHECK (Biểu_thức_luận_lý)
```

Trong đó :

- **Tên constraint** : phải là duy nhất trong cơ sở dữ liệu. Thông thường quy định tên constraint gồm có 3 phần. Bắt đầu bằng các chữ : CHK, kế tiếp là tên bảng và cuối cùng là tên cột áp dụng quy tắc kiểm tra miền giá trị dữ liệu.
- **Biểu thức luận lý** : là một biểu thức chỉ định quy tắc kiểm tra dữ liệu trong bảng. Thông thường biểu thức luận lý sẽ là biểu thức so sánh.

Ví dụ :

Trong bảng CTDONDH (chi tiết đơn đặt hàng) kiểm tra quy tắc giá trị dữ liệu của cột số lượng đặt tối thiểu là 10 và không lớn hơn 50. Có nghĩa là cột số lượng đặt ≥ 10 và số lượng đặt ≤ 50 , giá trị mặc định sẽ là 10 khi người sử dụng không cung cấp giá trị cột số lượng đặt.

```
CREATE TABLE CTDONDH  
(  
    Sodh      CHAR(4),  
    Mavtu     CHAR(4),  
    SIDat     SMALLINT      DEFAULT 20  
    CHECK (SIDat BETWEEN 10 AND 50)  
    PRIMARY KEY (Sodh, Mavtu),  
    FOREIGN KEY (Sodh) REFERENCES DONDH (Sodh),  
    FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)  
)
```

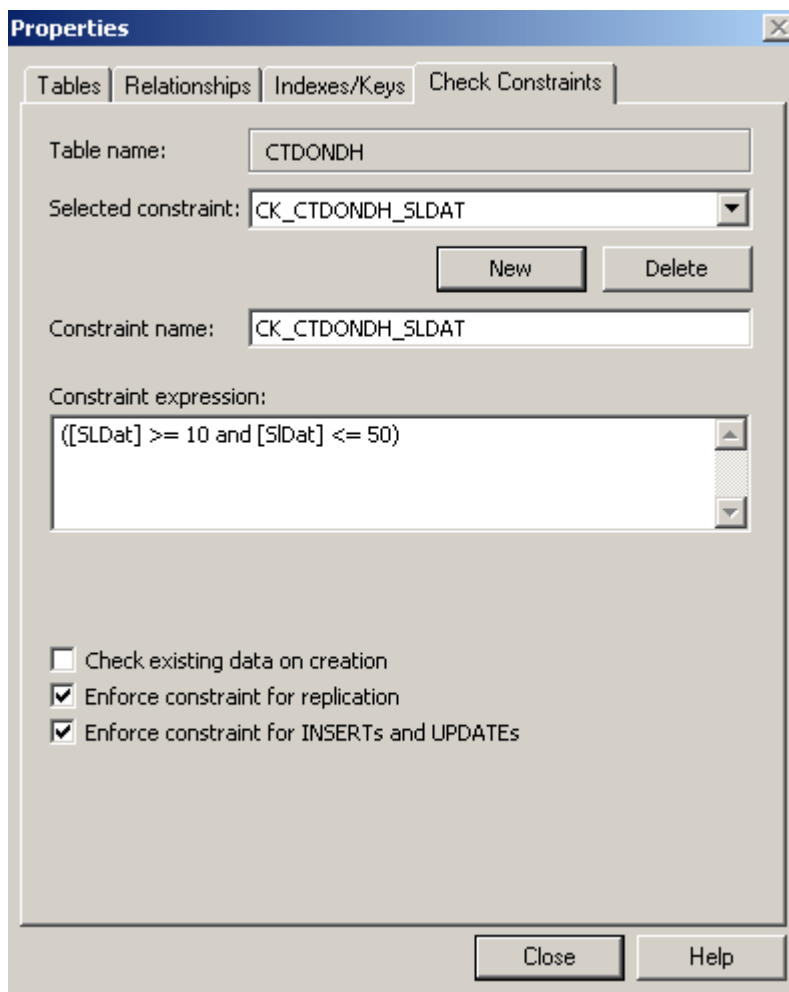
Hoặc muốn đặt tên cho các constraint liên quan

```

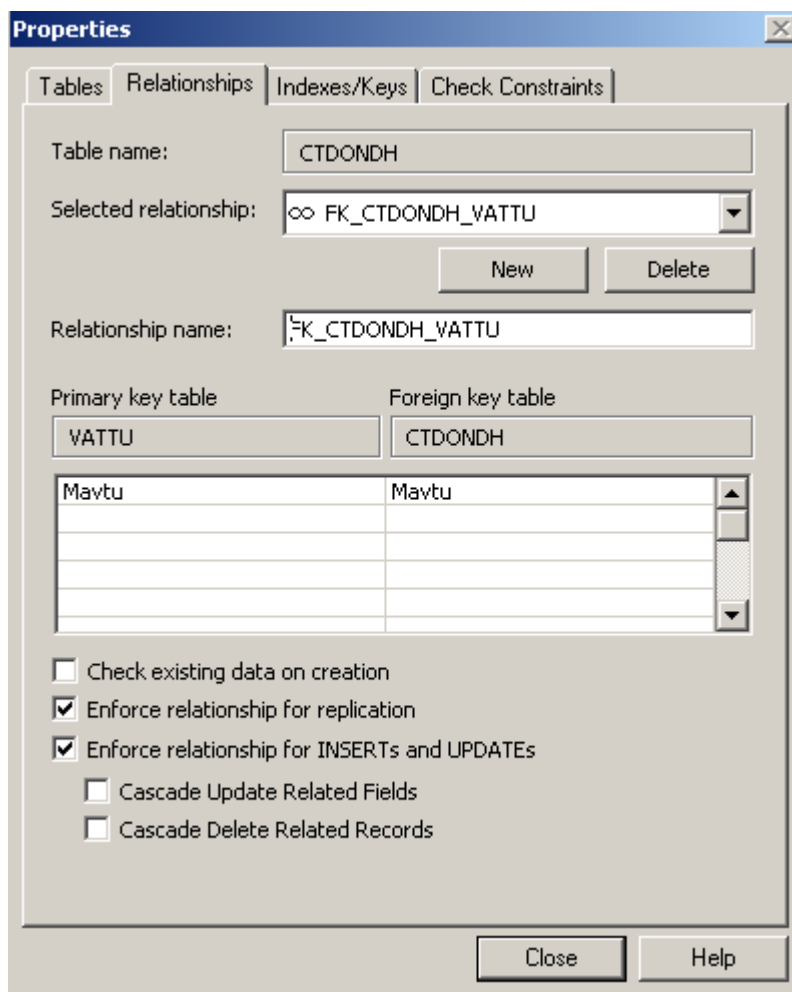
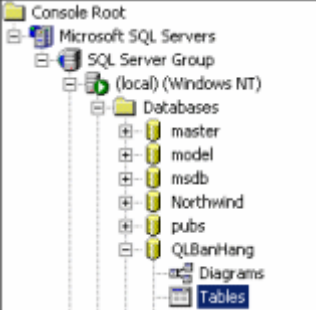
CREATE TABLE CTDONDH
(
    Sodh    CHAR(4),
    Mavtu   CHAR(4),
    SLDat   SMALLINT      DEFAULT 20
    CONSTRAINT CHK_CTDONDH_SLDat
        CHECK (SLDat BETWEEN 10 AND 50)
    CONSTRAINT PRK_CTDONDH_Sodh
        PRIMARY KEY (Sodh, Mavtu)
    CONSTRAINT FRK_CTDONDH_Sodh
        FOREIGN KEY (Sodh) REFERENCES DONDH (Sodh) ,
    CONSTRAINT FRK_CTDONDH_Mavtu
        FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)
)

```

Khi muốn xem tên của các constraint và các quy tắc liên quan được định nghĩa trong cấu trúc bảng, bạn chọn chức năng **Properties** sau khi nhấn vào tên bảng trong màn hình mô hình quan hệ dữ liệu (**diagram**).



Tương tự, bạn có thể di chuyển qua các trang **Relationships** hoặc **Indexes/Keys** để thấy được các quy tắc kiểm tra dữ liệu về tính duy nhất, tính tồn tại của các dữ liệu bên trong bảng.



Còn tiếp

Tính toán vận dữ liệu trong cơ sở dữ liệu - Phần cuối

[Share](#)   + [Đọc: 1052-Ngày đăng: 29-05-2009-Ngày sửa: 29-05-2009]

Thay vì tạo cấu trúc bảng bằng duy nhất một câu lệnh CREATE TABLE với nhiều thành phần của các đối tượng constraint khó nhớ bên trong đó, bạn có thể tạo ra cấu trúc bảng đơn giản và sau đó thêm vào các constraint mới trong bảng dùng để thực hiện quy tắc kiểm tra cho các loại ràng buộc toàn vẹn dữ liệu trên bảng.

4/- Thêm vào constraint mới trong bảng :

Với cú pháp ALTER TABLE tổng quát bên dưới cho phép bạn thêm vào các loại quy tắc kiểm tra toàn vẹn dữ liệu thông qua các đối tượng constraint bên trong bảng.

Cú pháp :

```
ALTER TABLE Tên_bảng
    ADD [ CONSTRAINT Tên_constraint ]
        LOẠI Các_tham_số
    [...]
```

Trong đó :

- **Tên constraint** : phải là duy nhất bên trong cơ sở dữ liệu. Thông thường quy định tên của một constraint bao gồm 3 phần nhỏ. Bắt đầu bằng các chữ : PRK, FRK, UNQ, CHK, DEF chỉ các loại constraint tương ứng sẽ là khóa chính, khóa ngoại, duy nhất, miền giá trị, giá trị mặc định, kế sau đó là tên của bảng và cuối cùng là tên cột sẽ áp dụng quy tắc kiểm tra dữ liệu.
- **Loại** : là các từ khóa tương ứng cho các thành phần theo từng loại constraint như kiểm tra tính duy nhất (**PRIMARY KEY, UNIQUE**), kiểm tra khóa ngoại (**FOREIGN KEY**), kiểm tra miền giá trị (**CHECK**), giá trị mặc định (**DEFAULT**).
- **Các tham số** : là các tham số cần thiết đi kèm theo với các từ khóa của từng loại constraint tương ứng.

Cú pháp chi tiết từng loại constraint :

```
ALTER TABLE Tên_bảng
ADD [ CONSTRAINT Tên_constraint ]
    PRIMARY KEY (Danh_sách_cột_khóa_chính)
Hoặc
    UNIQUE (Danh_sách_các_cột)
Hoặc
    FOREIGN KEY (Danh_sách_cột_khóa_ngoại
REFERENCES Tên_bảng_tham_chiếu _
                (Danh_sách_cột_tham_chiếu)
Hoặc
    CHECK (Biểu_thức_luận_lý)
Hoặc
    DEFAULT Giá_trị_mặc_định FOR Tên_cột
```

Ví dụ :

Tạo cấu trúc bảng NHACC (nhà cung cấp) với khóa chính là cột mã nhà cung cấp. Trước tiên bạn tạo cấu trúc bảng đơn giản.

```
CREATE TABLE NHACC
(
    Manhacc CHAR(3) NOT NULL ,
    Tenrhacc CHAR(30) NOT NULL ,
    Diachi VARCHAR(50) NOT NULL ,
    Dienthoai CHAR(15)
)
```

Sau đó bạn thêm vào các quy tắc kiểm tra duy nhất của dữ liệu tại khóa chính của bảng là cột mã nhà cung cấp bằng lệnh **ALTER TABLE ADD CONSTRAINT**.

```
ALTER TABLE NHACC
ADD CONSTRAINT PRK_NHACC_MANHACC
PRIMARY KEY (MANHACC)
```

Bạn có thể thêm một quy tắc mới dùng để kiểm tra tính duy nhất dữ liệu tại cột địa chỉ của bảng nhà cung cấp và đưa vào giá trị mặc định cho cột điện thoại của bảng nhà cung cấp là "Chưa có" bằng các câu lệnh như sau :

```
ALTER TABLE NHACC
ADD CONSTRAINT UNQ_NHACC_DIACHI
UNIQUE (DIACHI)
CONSTRAINT DEF_NHACC_DIENTHOAI
DEFAULT 'Chưa có' FOR DIENTHOAI
```

Lưu ý :

Trong lệnh **ALTER TABLE ADD CONSTRAINT** khi cần định nghĩa giá trị mặc định cho một cột dữ liệu bên trong bảng thì bắt buộc bạn phải có thêm từ khóa **FOR** tên cột để chỉ định giá trị mặc định được tạo ra sẽ áp dụng cho cột dữ liệu nào.

Ví dụ :

Sử dụng các câu lệnh **CREATE TABLE** để tạo ra cấu trúc các bảng VATTU (vật tư), NHACC (nhà cung cấp), DONDH (đơn đặt hàng) và CTDONDH (chi tiết đơn đặt hàng).

Bảng VATTU có các quy tắc kiểm tra toàn vẹn dữ liệu :

- Khóa chính là cột mã vật tư.
- Giá trị dữ liệu duy nhất tại cột tên vật tư.
- Giá trị mặc định cho cột phần trăm là 20.
- Kiểm tra phạm vi giá trị của phần trăm từ 5 đến 100.

```

CREATE TABLE VATTU
(
    Mavtu    CHAR(4)      NOT NULL ,
    Tenvtu   CHAR(30)     NOT NULL ,
    DvTinh   CHAR(10)    NOT NULL ,
    Phantram  TinyInt
    CONSTRAINT DEF_VATTU_PHANTRAM
        DEFAULT 20 ,
    CONSTRAINT PRK_VATTU_MAVTU
        PRIMARY KEY (Mavtu) ,
    CONSTRAINT UNQ_VATTU_TENMTU
        UNIQUE (Tenvtu) ,
    CONSTRAINT CHK_VATTU_PHANTRAM
        CHECK (Phantram BETWEEN 5 AND 100)
)

```

Hoặc đầu tiên tạo bảng có cấu trúc đơn giản và sau đó thêm vào các ràng buộc toàn vẹn dữ liệu.

```

CREATE TABLE VATTU
(
    Mavtu    CHAR(4)      NOT NULL ,
    Tenvtu   CHAR(30)     NOT NULL ,
    DvTinh   CHAR(10)    NOT NULL ,
    Phantram  TinyInt
)
GO
ALTER TABLE VATTU
    CONSTRAINT PRK_VATTU_MAVTU
        PRIMARY KEY (Mavtu) ,
    CONSTRAINT DEF_VATTU_PHANTRAM
        DEFAULT 20 FOR Phantram ,
    CONSTRAINT UNQ_VATTU_TENMTU
        UNIQUE (Tenvtu) ,
    CONSTRAINT CHK_VATTU_PHANTRAM
        CHECK (Phantram BETWEEN 5 AND 100)

```

Bảng NHACC có các quy tắc kiểm tra toàn vẹn dữ liệu :

- Khóa chính là cột mã nhà cung cấp.
- Giá trị dữ liệu duy nhất tại cột địa chỉ.
- Giá trị mặc định cho cột điện thoại là "Chưa có".


```

CREATE TABLE NHACC
(
    Manhacc CHAR(3) NOT NULL ,
    Tenrhacc CHAR(30) NOT NULL ,
    Diachi VARCHAR(50) NOT NULL ,
    Dienthoai CHAR(15) NOT NULL ,
    CONSTRAINT DEF_NHACC_DIENHAI
        DEFAULT 'Chưa có' ,
    CONSTRAINT PRK_NHACC_MANHACC
        PRIMARY KEY (Manhacc) ,
    CONSTRAINT UNQ_NHACC_DIACHI
        UNIQUE (Diachi)
)

```

Hoặc đầu tiên tạo bảng có cấu trúc đơn giản và sau đó thêm vào các ràng buộc toàn vẹn dữ liệu.

```

CREATE TABLE NHACC
(
    Manhacc CHAR(3) NOT NULL ,
    Tenrhacc CHAR(30) NOT NULL ,
    Diachi VARCHAR(50) NOT NULL ,
    Dienthoai CHAR(15) NOT NULL ,
)
GO
ALTER TABLE NHACC
    ADD CONSTRAINT PRK_NHACC_MANHACC
        PRIMARY KEY (Manhacc) ,
    CONSTRAINT UNQ_DIACHI
        UNIQUE (Diachi) ,
    CONSTRAINT DEF_NHACC_DIENHAI
        DEFAULT 'Chưa có' FOR Dienthoai

```

Bảng DONDH có các quy tắc kiểm tra toàn vẹn dữ liệu :

- Khóa chính là cột số đơn đặt hàng.
- Giá trị mặc định cho các cột ngày đặt hàng và ngày dự kiến nhận hàng là ngày hiện hành.
- Ngày dự kiến nhận hàng phải sau ngày đặt hàng.
- Kiểm tra tính tồn tại dữ liệu của cột mã nhà cung cấp bên bảng NHACC.

```

CREATE TABLE DONDH
(
    Sodh      CHAR(4) ,
    Ngaydh   DATETIME      NOT NULL
    CONSTRAINT DEF_DONDH_NGAYDH
        DEFAULT GETDATE() ,
    Ngaydknh DATETIME      NOT NULL
    CONSTRAINT DEF_DONDH_NGAYDKNH
        DEFAULT GETDATE() ,
    Manhacc  CHAR(3)      NOT NULL
    CONSTRAINT PRK_DONDH_SODH
        PRIMARY KEY (Sodh) ,
    CONSTRAINT CHK_DONDH_NGAYDKNH
        CHECK (Ngaydh <= Ngaydknh) ,
    CONSTRAINT FRK_DONDH_MANHACC
        FOREIGN KEY (Manhacc) REFERENCES NHACC (Manhacc)
)

```

Hoặc đầu tiên tạo bảng có cấu trúc đơn giản và sau đó thêm vào các ràng buộc toàn vẹn dữ liệu.

```

CREATE TABLE DONDH
(
    Sodh      CHAR(4) NOT NULL ,
    Ngaydh   DATETIME      NOT NULL ,
    Ngaydknh DATETIME      NOT NULL ,
    Manhacc  CHAR(3)      NOT NULL ,
)
GO
ALTER TABLE DONDH
    ADD CONSTRAINT PRK_DONDH_SODH
        PRIMARY KEY (Sodh) ,
    CONSTRAINT DEF_DONDH_NGAYDH
        DEFAULT GETDATE() FOR Ngaydh ,
    CONSTRAINT DEF_DONDH_NGAYDKNH
        DEFAULT GETDATE() FOR Ngaydknh ,
    CONSTRAINT CHK_DONDH_NGAYDKNH
        CHECK (Ngaydh <= Ngaydknh) ,
    CONSTRAINT FRK_DONDH_MANHACC
        FOREIGN KEY (Manhacc) REFERENCES NHACC (Manhacc)

```

Bảng CTDONDH có các quy tắc kiểm tra toàn vẹn dữ liệu :

- Khóa chính là cột số đặt hàng và mã vật tư.
- Giá trị mặc định cho cột số lượng đặt là 10.
- Kiểm tra phạm vi giá trị của số lượng đặt từ 10 đến 50.

- Kiểm tra tính tồn tại dữ liệu của cột số đặt hàng trong bảng DONDH và cột mã vật tư trong bảng VATTU.

```
CREATE TABLE CTDONDH
(
    Sodh      CHAR(4),
    Mavtu     CHAR(4),
    SIDat     SMALLINT
    CONSTRAINT DEF_CTDONDH_SLDAT DEFAULT 10,
    CONSTRAINT PRK_CTDONDH_SODH
        PRIMARY KEY (Sodh, Mavtu),
    CONSTRAINT FRK_CTDONDH_SODH
        FOREIGN KEY (Sodh) REFERENCES DONDH (Sodh),
    CONSTRAINT FRK_CTDONDH_MAVTU
        FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)
    CONSTRAINT CHK_CTDONDH_SLDAT
        CHECK (SIDat BETWEEN 10 AND 50)
)
```

Hoặc đầu tiên tạo bảng có cấu trúc đơn giản và sau đó thêm vào các ràng buộc toàn vẹn dữ liệu.

```
CREATE TABLE CTDONDH
(
    Sodh      CHAR(4)      NOT NULL,
    Mavtu     CHAR(4)      NOT NULL,
    SIDat     SMALLINT
)
GO
ALTER TABLE CTDONDH
    ADD CONSTRAINT PRK_CTDONDH_SODH
        PRIMARY KEY (Sodh, Mavtu),
    CONSTRAINT DEF_CTDONDH_SLDAT
        DEFAULT 10 FOR SIDat,
    CONSTRAINT FRK_CTDONDH_SODH
        FOREIGN KEY (Sodh) REFERENCES DONDH (Sodh),
    CONSTRAINT FRK_CTDONDH_MAVTU
        FOREIGN KEY (Mavtu) REFERENCES VATTU (Mavtu)
    CONSTRAINT CHK_CTDONDH_SLDAT
        CHECK (SIDat BETWEEN 10 AND 50)
```

Lưu ý :

Khi thực hiện tạo bảng có cấu trúc đơn giản và sau đó thêm vào các ràng buộc toàn vẹn dữ liệu thì bạn phải luôn luôn thêm từ khóa **NOT NULL** phía sau các cột tham gia làm khóa chính của bảng khi tạo cấu trúc bằng lệnh **CREATE TABLE**.

5/- Hủy bỏ constraint đã có trong bảng :

Với cú pháp **ALTER TABLE** bên dưới cho phép bạn hủy bỏ các quy tắc kiểm tra toàn vẹn dữ liệu thông qua các đối tượng constraint có bên trong bảng đã tạo trước đó. Các quy tắc mà bạn hủy bỏ sẽ không còn thực hiện việc kiểm tra tính toàn vẹn dữ liệu bên trong của bảng cho đến khi bạn phải tạo mới lại nó.

Muốn hủy bỏ được các đối tượng constraint bắt buộc bạn phải ghi nhớ được tên của các constraint lúc tạo ra nó. Hoặc xem lại trong màn hình thuộc tính của bảng. Do vậy khi tạo ra các quy tắc để kiểm tra các ràng buộc toàn vẹn dữ liệu bạn nên chủ động đặt tên cho các đối tượng constraint bên trong cơ sở dữ liệu Microsoft SQL Server .

Cú pháp :

```
ALTER TABLE Tên_bảng  
DROP CONSTRAINT Tên_constraint [, ...]
```

Trong đó :

- Tên constraint : phải tồn tại trong cơ sở dữ liệu mà bạn muốn hủy bỏ. Do đó đối với tên của các constraint mà bạn chủ động đặt khi sử dụng từ khóa **CONSTRAINT** thì sẽ giúp bạn dễ dàng gợi nhớ hơn tên của các constraint do hệ thống tự đặt.

Lưu ý :

Hủy bỏ quy tắc kiểm tra số lượng đặt hàng trong bảng CTDONDH có giới hạn từ 10 đến 50.

```
ALTER TABLE CTDONDH  
DROP CONSTRAINT CHK_CTDONDH_SLDAT
```