|  |
| --- |
|  |
| |  | | --- | | [**Home**](http://www.thuvien-it.net/home/tinhoc/default.asp) >> [**Cơ Sở Dữ Liệu**](http://www.thuvien-it.net/home/tinhoc/cat.asp?cid=3) >> [**MS SQL Server**](http://www.thuvien-it.net/home/tinhoc/sub.asp?sid=74)>> | | |  | | --- | |  | | |  | | --- | | [**Home**](http://www.thuvien-it.net/home/tinhoc/default.asp) >> [**Cơ Sở Dữ Liệu**](http://www.thuvien-it.net/home/tinhoc/cat.asp?cid=3) >> [**MS SQL Server**](http://www.thuvien-it.net/home/tinhoc/sub.asp?sid=74)>> | | **Khái quát về Trigger – Phần 1**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/06/KhaiquatveTrigger_Phan1_d.gif |   **Trigger có thể xem là một dạng đặc biệt của thủ tục nội tại, bởi vì bên trong nội dung của trigger lưu trữ các câu lệnh dùng để thực hiện một số hành động nào đó mà người lập trình sẽ chỉ ra.** | | 1/- Trigger là gì ?  Tuy nhiên khác với thủ tục nội tại, trigger hoàn toàn không có tham số. Ngoài ra chúng ta có thể gọi thực hiện trực tiếp trigger bằng lệnh **EXECUTE** như thủ tục nội tại hoặc bằng bất kỳ một lệnh nào khác, thay vào đó trigger sẽ được thực hiện một cách tự động khi dữ liệu của bảng có liên quan đến trigger bị cập nhật.  Chính nhờ vào tính năng đặc biệt là tự động thực hiện mà nội dung các lệnh bên trong trigger được dùng cho các công việc sau :  • Kiểm tra các ràng buộc toàn vẹn dữ liệu phức tạp.  • Thực hiện các xử lý được thiết kế thi hành tại server (trong mô hình client/server). Các xử lý mà ta muốn chúng sẽ được tự động thực hiện khi có thao tác **INSERT**, **UPDATE** hoặc **DELETE** xảy ra.  • Trigger cũng được dùng để thay thế các constraint trong trường hợp ta muốn việc kiểm tra ràng buộc dữ liệu kèm theo các câu thông báo thích hợp theo ý muốn người dùng.  2/- Mở rộng ràng buộc toàn vẹn dữ liệu với các Trigger  Có 3 thao tác cơ bản làm thay đổi dữ liệu trên các bảng của cơ sở dữ liệu đó là thao tác thêm (insert), thao tác sửa (update) và thao tác xóa (delete). Như vậy để đảm bảo dữ liệu nhất quán và đúng đắn, ta cần kiểm tra việc thực hiện của 3 thao tác này.  Có 2 cách kiểm tra : kiểm tra mức giao diện và kiểm tra mức cơ sở dữ liệu.  • *Kiểm tra mức giao diện :* công việc này chính là công việc lập trình trên các màn hình giao diện.  • *Kiểm tra mức cơ sở dữ liệu :* công việc này được thực hiện bởi các đối tượng constraint hoặc trigger.  Đối với các ràng buộc toàn vẹn dữ liệu đơn giản như kiểm tra các ràng buộc miền giá trị, kiểm tra các ràng buộc giữa các thuộc tính trên cùng một bảng dữ liệu, … ta nên sử dụng đối tượng constraint.  Đối với các ràng buộc toàn xẹn dữ liệu phức tạp khác – là những quy tắc được định nghĩa dùng để kiểm tra tính toàn vẹn của dữ liệu trên nhiều cột hoặc nhiều dòng của các bảng khác nhau. Khi đó bắt buộc chúng ta phải sử dụng đối tượng trigger, đối tượng này cho phép chúng ta xây dựng các câu lệnh bên trong nó với mục tiêu là các câu lệnh này sẽ được thực hiện khi các thao tác làm thay đổi dữ liệu xảy ra trên bảng dữ liệu mà nó gắn.  Một trigger gắn với duy nhất một đối tượng table hoặc một đối tượng view.  3/- Các dạng ràng buộc toàn vẹn dữ liệu nên dùng Trigger  Trên SQL Server, có 2 cách thực hiện ràng buộc toàn vẹn dữ liệu :  • Ràng buộc toàn vẹn dữ liệu bằng phương pháp mô tả (declarative data integrity).  Thực hiện ràng buộc dữ liệu theo phương pháp này chính là các công việc xác định ràng buộc khóa chính, khóa ngoại, kiểm tra miền giá trị, … và mô tả chúng tại thời điểm tạo table.  Đặc điểm cần biết của phương pháp này là việc kiểm tra sẽ được thực hiện trước khi cho phép ghi vào table.  • Ràng buộc toàn vẹn dữ liệu bằng phương pháp theo thủ tục (procedural data integrity).  Trong phương pháp này, việc ràng buộc toàn vẹn dữ liệu được xác định bởi tập hợp các câu lệnh T-SQL. Các câu lệnh chứa bên trong đối tượng Trigger, chúng sẽ được gọi thi hành khi có thao tác thêm, xóa, hoặc sửa dữ liệu xảy ra trên table tương ứng.  Đặc điểm cần biết của phương pháp này là việc kiểm tra sẽ được thực hiện sau khi dữ liệu được ghi vào table.  4/- Khi nào thì sử dụng Trigger ?  Để giải đáp thỏa đáng vấn đề này, chúng ta xem xét ví dụ sau :  Cho một cơ sở dữ liệu có cấu trúc sau :  • KHACH\_HANG(ID\_KHG, Ten\_NCC)  • PHIEU\_XUAT(ID\_PX, So\_PX, Ngay\_PN, #ID\_KHG)  • CT\_PHIEU\_XUAT(#ID\_PX, #ID\_HH, SoLuong, DonGia, ThanhTien)  • DM\_HANGHOA(ID\_HH, Ten\_HH, DonGiaHienHanh)  Để bảo đảm tính nhất quán và chính xác của dữ liệu, cần thực hiện yêu cầu ràng buộc toàn vẹn dữ liệu :  1/- KHACH\_HANG : Một khách hàng có 1 mả duy nhất dùng để phân biệt khách hàng này với khách hàng khác, có 1 tên khách hàng duy nhất.  2/- DM\_HANGHOA : Một hàng hóa có 1 mã hàng hóa duy nhất dùng để phân biệt với hàng hóa khác, có 1 tên hàng hóa và 1 đơn giá bán hiện tại duy nhất.  3/- PHIEU\_XUAT : Một phiếu xuất có 1 mã duy nhất dùng để phân biệt với phiếu xuất khác, có 1 số phiếu xuất, 1 ngày xuất xác định.  Một phiếu xuất liên quan đến duy nhất 1 khách hàng.  Một phiếu xuất có ít nhất 1 chi tiết xuất.  4/- CT\_PHIEU\_XUAT : Một chi tiết xuất có mã phiếu xuất và mã hàng hóa dùng để xác định khóa của quan hệ. Khóa này dùng để phân biệt với chi tiết xuất khác, có 1 giá trị số lượng xác định, 1 đơn giá tương ứng với đơn giá hiện hành của hàng hóa lấy từ thuộc tính DonGiaHienHanh của table DM\_HANG\_HOA. Giá trị của ThanhTien được tính toán dựa trên công thức ThanhTien = SoLuong \* DonGia.  **Câu hỏi đặt ra :** với các yêu cầu ràng buộc toàn vẹn dữ liệu ở trên, yêu cầu nào cần phải sử dụng trigger để cài đặt ?  **Câu trả lời :** Chỉ có ràng buộc 1 đơn giá tương ứng với đơn giá hiện hành của hàng hóa lấy từ thuộc tính DonGiaHienHanh của table DM\_HANG\_HOA là thật sự cần phải sử dụng trigger để giải quyết vấn đề. Các ràng buộc còn lại ta có thể sử dụng theo phương pháp mô tả.  **Các trường hợp sử dụng Trigger :**  • Quan điểm cơ bản cần nghĩ đến trước tiên là chỉ những ràng buộc dữ liệu nào không thể dùng phương pháp mô tả (declarative data integrity) thì mới dùng trigger để giải quyết. Tuy nhiên, bạn không nhất thiết phải tuân thủ theo quan điểm này trên thực tế bạn hoàn toàn có thể dùng trigger để thay thế cho constraint trong trường hợp bạn muốn tự mình kiểm soát và cho ra các câu thông báo thích hợp người dùng.  • Khi có sự thay đổi dữ liệu trên 1 table và chúng ta muốn dữ liệu trên 1 hay nhiều table khác cũng tự động thay đổi theo cho phù hợp.  *Ví dụ :*  Chúng ta có các table có cấu trúc sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/06/KhaiquatveTrigger_Phan1_1.gif  **Yêu cầu :**  Khi thêm mới mẫu tin trên table DON\_GIA\_HANG\_HOA cần update lại mục DonGiaHienHanh trên table DM\_HANG\_HOA sao cho DM\_HANG\_HOA.DonGiaHienHanh = DON\_GIA\_HANG\_HOA.Gia\_DGHH  **Giải quyết vấn đề :**  Với các yêu cầu ràng buộc dữ liệu này, ta cần phải dùng trigger để thực hiện.  • Trên thực tế đô khi người lập trình muốn thay đổi cấu trúc lưu trữ của dữ liệu với mục tiêu tăng tốc độ xử lý hoặc việc xử lý tính toán dễ dàng hơn, điều này thường dẫn đến việc phá vỡ tính chuẩn của cơ sở dữ liệu và làm tăng thêm ràng buộc toàn vẹn dữ liệu. Ví dụ, ta có trường hợp sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/06/KhaiquatveTrigger_Phan1_2.gif  Cột TongSoTien được bổ sung vào table PHIEU\_XUAT với ý nghĩa dùng để chứa giá trị tổng số tiền của phiếu xuất. Giá trị này bằng tổng các giá trị chứa trên cột ThanhTien của cùng phiếu xuất trên table CT\_PHIEU\_XUAT. Vấn đề khó khăn ở đây là khi thêm, xóa mẫu tin hoặc sửa đổi giá trị của cột ThanhTien trên CT\_PHIEU\_XUAT, tính đúng đắn của cột TongSoTien trên PHIEU\_XUAT phải luôn được đảm bảo.  Gặp các tình huống như thế này, bạn không cần phải đắn đo khi quyết định sử dụng triger.  • Nếu bạn muốn cơ sở dữ liệu có những thông báo lỗi thích hợp theo ý muốn khi thao tác thay đổi dữ liệu vi phạm ràng buộc toàn vẹn, bạn nên dùng trigger.  • Tất cả các xử lý mà bạn muốn chúng tự động thực hiện tại phía server khi có thao tác dữ liệu INSERT, UPDATE hoặc DELETE xảy ra.  *Còn tiếp* | | | **Khái quát về Trigger – Phần 2**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/KhaiquatveTrigger_Phan2_d.gif |   **Trigger không thể được tạo trên 1 table tạm (temporary table, là table được tạo với tên table có ký tự # hoặc ## phía trước) hoặc table hệ thống (system table). Tuy nhiên cần chú ý rằng các câu lệnh bên trong trigger hoàn toàn có thể tham chiếu đến nội dung bên trong các table tạm và table hệ thống.** | | 5/- Các đặc trưng và hạn chế  • Từ một trigger có thể xây dựng bên trong nó các câu lệnh tác động lên cột của table bất kỳ trong cơ sở dữ liệu hoặc tớti các đối tượng bên ngoài cơ sở dữ liệu hiện hành.  • Có thể áp dụng trigger cho đối tượng View.  • Một trigger có thể thực hiện nhiều hành động (multiple actions), và nó có thể được kích hoạt bởi nhiều hơn 1 biến cố (các biến cố gồm : insert, delete, update).  • Trigger không thể được tạo trên 1 table tạm (temporary table, là table được tạo với tên table có ký tự # hoặc ## phía trước) hoặc table hệ thống (system table). Tuy nhiên cần chú ý rằng các câu lệnh bên trong trigger hoàn toàn có thể tham chiếu đến nội dung bên trong các table tạm và table hệ thống.  • Một hạn chế quan trọng mà bạn cần quan tâm là các trigger loại INSTEAD OF DELETE và INSTEAD OF UPDATE không thể được định nghĩa trên các table có chứa khóa ngoại và trên dây quan hệ nối từ table chứa nó thông qua khóa ngoại đến table khác đã có thiết đặt tương ứng trên các dây quan hệ tính chất Cascade Delete Related Records và Cascade Update Related Fieds.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/KhaiquatveTrigger_Phan2_1.gif  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/KhaiquatveTrigger_Phan2_2.gif  Trong mối quan hệ ở hình trên, nếu ta thiết đặt tính **Cascade Delete Related Records**của dây quan hệ thì việc tạo trigger loại Instead Of Delete sẽ không thể thực hiện thành công.  6/- Các biến cố trigger  Có 3 biến cố tự kích họat một trigger : INSERT, UPDATE và DELETE. Các biến cố xảy ra trên đối tượng table hoặc view tại thời điểm các thao tác dữ liệu tương ứng tác động lên các đối tượng này. Ta không thể gọi thi hành trực tiếp 1 trigger.  Tồn tại các trường hợp chắc chắn là khi một sự kiện sửa hoặc xóa dữ liệu xảy ra nhưng không kích hoạt trigger, đó là các trường hợp sau :  • Khi sử dụng câu lệnh TRUNCATE TABLE sẽ không làm kích hoạt trigger được định nghĩa cho biến cố DELETE. Một điều quan trọng cần quan tâm đối với các trigger là nếu giao tác (transaction) không thành công sẽ tự động quan lui (rolled back). Bởi vì câu lệnh TRUNCATE TABLE không là một biến cố được ghi nhận vào sổ ghi nhận biến cố, nó không thể được quay lui vì vậy nó không kích hoạt trigger DELETE.  • Câu lệnh WRITETEXT cũng không là nguyên nhân gây kích hoạt trigger INSERT hoặc UPDATE.  7/- Cơ chế hoạt động của trigger  Khi thực hiện việc thêm mới một mẫu tin vào một table, thao tác này sẽ kích hoạt một trigger, trigger lưu trữ dữ liệu của mẫu tin vừa thêm mới vào một table có tên là Inserted. Tương tự khi thực hiện việc xóa mẫu tin của một table, thao tác này sẽ kích hoạt một trigger, trigger lưu trữ dữ liệu của mẫu tin vừa xóa vào một table có tên là Deleted.  Cấu trúc của hai bảng này hòan tòan giống với cấu trúc của bảng dữ liệu liên quan đến trigger khi tạo ra. Thật ra hai bảng này chỉ tồn tại trong bộ nhớ của máy tính (RAM) được xem như là hai bảng luận lý mà chúng ta có thể sử dụng trong các xử lý của trigger. Chúng ta không thể tham chiếu trực tiếp trong tiện ích SQL Query Analyzer hoặc bên trong các thủ tục nội tại.  Chúng ta đã đề cập đến cơ chế hoạt động của trigger liên quan đến 2 thao tác thêm và xóa dữ liệu. Còn thao tác sửa đổi dữ liệu (Update) trong Microsoft SQL Server được xem như là sự phối hợp của hai lệnh DELETE và INSERT (xóa bỏ dữ liệu cũ và thêm vào dữ liệu mới sau khi sửa đổi) do thế mà đối với các trigger liên quan đến việc sửa đổi dữ liệu thì chúng ta có thể tham chiếu đến cả hai bảng trung gian Inserted và Deleted.  Trong đó bảng Deleted sẽ chứa đựng thông tin của các dòng dữ liệu đã bị hủy bỏ, các dòng dữ liệu cũ trước khi sửa đổi, bảng Inserted sẽ chứa đựng thông tin của các dòng dữ liệu mới vừa thêm vào, các dòng dữ liệu sau khi sửa đổi.  Một khái niệm quan trọng liên quan đến cơ chế hoạt động của trigger mà chúng tôi muốn đề cập ở phần này, đây là khái niệm liên quan đến phần bổ sung mới của phiên bản SQL Server 2000, khái niệm INSTEAD OF và AFTER.  • Trigger loại INSTEAD OF sẽ bỏ qua hành động kích hoạt trigger (các thao tác insert, delete, update), thay vào đó nó sẽ thực hiện các câu lệnh bên trong trigger. Cần lưu ý là nó sẽ bỏ qua hành động tác động đến cơ sở dữ liệu nhưng việc lưu trữ dữ liệu vào các bảng Inserted (đối với thao tác insert), deleted (đối với thao tác delete) vẫn được thực hiện.  *Ví dụ :*  Một câu lệnh update tác động đến một table chứa trigger loại INSTEAD OF, các câu lệnh T-SQL bên trong trigger sẽ được thi hành, nó đóng vai trò thay thế câu lệnh Update. Đặc tính này cho phép bạn chỉ định các câu lệnh xử lý phức tạp trong trigger nhằm làm tăng thêm lệnh thay đổi dữ liệu trên table.  • Trigger loại INSTEAD OF có thể định nghĩa trên đối tượng table hoặc view.  • Trigger loại AFTER là loại mặc định (cũng chính là loại có từ khóa khai báo FOR), nó có vai trò như là phần bổ sung của hành động kích hoạt trigger. Các câu lệnh bên trong trigger loại này chỉ được thi hành sau khi hành động kích hoạt đã được thực hiện rồi. Cần chú ý là các ràng buộc toàn vẹn loại mô tả sẽ được thực hiện trước, kế đó các câu lệnh bên trong trigger sẽ được thực hiện sau. Trigger loại AFTER chỉ được phép định nghĩa trên duy nhất đối tượng table (không thể định nghĩa trên view).  Trên một table, nếu muốn bạn có thể định nghĩa cả hai loại trigger. Nếu bạn định nghĩa cả hai loại trigger và các constraint cho một table, thứ tự thi hành sẽ là trigger INSTEAD OF, các constraint được xử lý và sau cùng là trigger AFTER. Nếu các constraint bị vi phạm (có nghĩa là việc kiểm tra ràng buộc toàn vẹn của các constraint phát hiện có vi phạm tính toàn vẹn dữ liệu), các hành động của trigger INSTEAD OF sẽ được quay lui, trigger AFTER không thi hành nếu các constraint bị vi phạm hoặc nếu có một số biến cố khác làm cho việc sửa đổi dữ liệu trên table bị lỗi.  Giống như các thủ tục nội tại, trigger có thể được lồng đến 32 cấp và có thể được kích hoạt đệ quy.  *Hết* |   **Làm việc với trigger – Phần 1**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_d.gif |   **Bạn có thể sử dụng câu lệnh CREATE TRIGGER để tạo một trigger bằng cách sử dụng tiện ích SQL Query Analyzer hoặc sử dụng công cụ dạng dấu nhắc lệnh (command-prompt) điển hình như công cụ osql.** | | 1/- Tạo mới trigger  Giống như các đối tượng khác trong Microsoft SQL Server, bạn có 2 cách để có thể tạo mới một trigger : sử dụng các câu lệnh T-SQL hoặc dùng tiện ích Enterprise Manager.  1.1/- Sử dụng câu lệnh T-SQL :  Bạn có thể sử dụng câu lệnh CREATE TRIGGER để tạo một trigger bằng cách sử dụng tiện ích SQL Query Analyzer hoặc sử dụng công cụ dạng dấu nhắc lệnh (command-prompt) điển hình như công cụ osql.  Khi sử dụng CREATE TRIGGER, bạn phải chú ý thực hiện những điều cốt lõi sau :  • Xác định tên của trigger. Tên của trigger có tính duy nhất trên 1 cơ sở dữ liệu (database).  • Chỉ định table hoặc view mà trên đó trigger được áp dụng.  • Chỉ định trigger thuộc loại INSTEAD OF hoặc AFTER.  • Chỉ định một biến cố hoặc nhiều biến cố kích hoạt trigger.  • Các câu lệnh tương ứng với nhiệm vụ mà bạn muốn trigger thực hiện.  • Các tùy chọn : trigger cho phép nhân bản (replicate) hay không, trigger có được mã hóa (encrypt) hay không.  *Chú ý :*  Mệnh đề WITH APPEND được sử dụng trong câu lệnh CREATE TRIGGER ở các phiên bản trước, không còn được sử dụng trong phiên bản SQL Server 2000.  *Cú pháp :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_1.gif  a/- Mệnh đề CREATE TRIGGER :  Việc tạo trigger bắt đầu với mệnh đề CREATE TRIGGER theo sau là tên của trigger. Trigger không cho phép xác định tên của cơ sở dữ liệu như là tiền tố của tên đối tượng. Do vậy cần chọn cơ sở dữ liệu với mệnh đề USE Tên\_database và từ khóa GO trước câu lệnh tạo trigger. Cần phải chỉ định từ khóa GO bởi vì câu lệnh CREATE TRIGGER phải là câu lệnh đầu tiên trong một query batch (Một batch là một lô bao gồm các câu lệnh T-SQL kết thúc bởi từ khóa GO). Mặc định, quyền chủ sở hữu table được phép tạo trigger.  *Chú ý :*  Nếu bạn sử dụng tiện ích SQL Query Analyzer và trong tiện ích này bạn đã chọn đúng database cần tạo trigger, khi đó bạn có thể không cần phải sử dụng mệnh đề USE Tên\_database.  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_2.gif  Trong trường hợp bạn muốn tên trigger là một chuỗi gồm có cả các khoảng trắng, bạn phải đặt chúng trong cặp dấu [ ].  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_3.gif  b/- Mệnh đề ON :  Trigger phải được gán tới table hoặc view, sử dụng mệnh đề ON để cho biết table hoặc view nào được áp dụng.  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_4.gif  Một trigger chỉ áp dụng cho một table hoặc view. Nếu bạn cần áp dụng thao tác trigger giống như vậy cho một table khác trong cơ sở dữ liệu, bạn phải tạo một trigger với một tên khác, và copy toàn bộ nội dung sang trigger mới, sau đó áp dụng nó cho table khác.  Loại trigger mặc định AFTER chỉ có thể áp dụng cho một table. Loại trigger mới INSTEAD OF có thể áp dụng hoặc cho một table hoặc cho một view.  c/- Các mệnh đề FOR, AFTER, INSTEAD OF :  Trigger phải được gán tới table hoặc view, sử dụng mệnh đề ON để cho biết table hoặc view nào được áp dụng.  Loại biến cố trigger phải được chỉ định khi trigger được tạo. Các biến cố gồm có : INSERT, UPDATE và DELETE. Một trigger có thể được kích hoạt bởi một, hai hoặc cả 3 biến cố xảy ra. Nếu bạn muốn trigger kích hoạt trên tất cả các biến cố thì theo sau các mệnh đề FOR, AFTER hoặc INSTEAD OF là các từ khóa INSERT, UPDATE và DELETE. Các kiểu biến cố này có thể được liệt kê theo thứ tự bất kỳ.  *Ví dụ :*  Trigger loại AFTER :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_5.gif  *Chú ý :*  Mệnh đề FOR có ý nghĩa tương đượng với mệnh đề AFTER.  Trigger loại INSTEAD OF :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_6.gif  *Chú ý :*  Mệnh đề FOR được thay thế với INSTEAD OF.  d/- Mệnh đề AS :  Mệnh đề AS được khai báo trước các câu lệnh T-SQL trong trigger.  *Ví dụ :*  Lấy lại ví dụ ở phần trình bày "các trường hợp sử dụng trigger". Yêu cầu của ví dụ này là khi thêm mới hoặc sửa đổi mẫu tin trên table DON\_GIA\_HANG\_HOA, ta muốn giá trị tại cột DonGiaHienHanh trên table DM-HANG\_HOA phải được cập nhật lại theo giá trị của cột Gia\_DGHH của mẫu tin vừa thêm mới hoặc sửa đổi. Trigger giải quyết vấn đề này có thể được thực hiện như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_7.gif  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Lamviecvoitrigger_Phan1_8.gif  *Còn tiếp* | |
| **Lập trình trigger – Phần 1**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_d.gif |   **Khi có thao tác Insert, Update hoặc Delete xảy ra trên table, các thao tác này sẽ kích hoạt trigger, biến cố tạo ra một hoặc nhiều table giả. Các table này có thể xem như là các sổ ghi nhận giao tác (transaction logs) của biến cố. Có hai loại : table Inserted và Deleted.** |
| 1/- Các table giả (pseudo tables) Inserted và Deleted  Khi có thao tác Insert, Update hoặc Delete xảy ra trên table, các thao tác này sẽ kích hoạt trigger, biến cố tạo ra một hoặc nhiều table giả. Các table này có thể xem như là các sổ ghi nhận giao tác (transaction logs) của biến cố. Có hai loại : table Inserted và Deleted.  Một biến cố insert hoặc update sẽ tạo ra một table : Inserted. Table này chứa tập mẫu tin được thêm mới (insert) hoặc được thay đổi (update).  Trigger Update cũng tạo ra một table : Deleted. Table này chứa tập mẫu tin tồn tại trước khi thao tác Update xảy ra.  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_1.gif  Trigger Them\_HH được tạo và gắn với table DM\_HANG\_HOA. Khi thao tác insert xảy ra trên table DM\_HANG\_HOA, trigger Them\_HH được kích hoạt. Câu lệnh SELECT bên trong trigger sẽ hiển thị tập mẫu tin chứa trên table Inserted. Đoạn lệnh sau dùng để kiểm tra hoạt động của trigger :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_2.gif  Kết quả, trigger sẽ hiển thị nội dung sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_3.gif  Căn cứ trên màn hình hiển thị kết quả thi hành của trigger, ta thấy table Inserted chứa nội dung của mẫu tin vừa insert vào table DM\_HANG\_HOA.  *Chú ý :*  Trong trường hợp này table deleted sẽ không chứa thông tin nào cả.  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_4.gif  Trigger Sua\_HH được tạo và gắn với table DM\_HANG\_HOA. Khi thao tác Update xảy ra trên table DM\_HANG\_HOA, trigger Sua\_HH được kích hoạt. Câu lệnh SELECT thứ nhất bên trong trigger sẽ hiển thị mẫu tin chứa trên table Inserted và câu lệnh SELECT thứ hai sẽ hiển thị tập mẫu tin chứa trong table deleted. Đoạn lệnh sau dùng để kiểm tra hoạt động của trigger :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_5.gif  Kết quả tương ứng với câu lệnh SELECT thứ nhất hiển thị nội dung chứa trong table Inserted :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_6.gif  Kết quả tương ứng với câu lệnh SELECT thứ hai hiển thị nội dung chứa trong table deleted :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_7.gif  Căn cứ trên màn hình hiển thị kết quả thi hành của trigger, ta thấy table Inserted chứa nội dung của mẫu tin vừa được sửa và table deleted chứa nội dung của mẫu tin tại thời điểm trước khi sửa.  Khi có thao tác Delete xảy ra trên table, thao tác này sẽ kích hoạt trigger, biến cố tạo ra một table Deleted. Table này chứa tập mẫu tin vừa bị xóa.  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_8.gif  Trigger Xoa\_HH được tạo và gắn với table DM\_HANG\_HOA. Khi thao tác delete xảy ra trên table DM\_HANG\_HOA, trigger Xoa\_HH được kích hoạt. Câu lệnh SELECT thứ nhất bên trong trigger sẽ hiển thị tập mẫu tin chứa trên table Inserted và câu lệnh SELECT thứ hai sẽ hiển thị tập mẫu tin chứa trong deleted. Đoạn lệnh sau dùng để kiểm tra hoạt động của trigger :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_9.gif  Kết quả tương ứng với câu lệnh SELECT thứ nhất hiển thị nội dung chứa trong table Inserted :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_10.gif  Kết quả tương ứng với câu lệnh SELECT thứ hai hiển thị nội dung chứa trong table deleted :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_11.gif  Căn cứ trên màn hình hiển thị kết quả thi hành của trigger, ta thấy table Inserted không chứa thông tin nào cả, và table deleted chứa nội dung của mẫu tin vừa được xóa.  2/- Các lệnh hệ thống và các hàm sử dụng trong trigger 2.1/- Mệnh đề UPDATE(Tên\_cột) :  *Ý nghĩa sử dụng :* Xác định có hay không một biến cố insert hoặc update xảy ra trên cột đã chỉ ra tại mục tham số Tên\_cột của mệnh đề update.  *Chú ý :*  Khi thao tác insert xảy ra trên table A nào đó, tất cả các cột trên table A đều xảy ra update cho dù có tồn tại một cột mà thao tác insert ta không hề chỉ định (cột không chỉ định chấp nhận mang giá trị NULL).  *Cú pháp :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_12.gif  Trong trường hợp bạn muốn kiểm tra nhiều hơn một cột trên table, có thể sử dụng như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_13.gif  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_14.gif  Trigger Them\_HH được tạo và gắn với table DM\_HANG\_HOA. Khi thao tác insert xảy ra trên table DM\_HANG\_HOA, trigger Them\_HH được kích hoạt. Câu lệnh IF UPDATE(Ten\_HH) dùng để kiểm tra có hay không việc Update trên cột Ten\_HH. UPDATE(Ten\_HH) sẽ trả về giá trị là TRUE nếu có Update xảy ra trên cột Ten\_HH. Đoạn lệnh sau dùng để kiểm tra hoạt động của trigger :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_15.gif  Khi thực hiện câu lệnh kiểm tra này, trigger sẽ xuất câu thông báo ’Cột Ten\_HH đã xảy ra Update’. Điều này chứng tỏ có xảy ra việc Update trên Ten\_HH, mặc dù thao tác INSERT không chỉ định cột Ten\_HH.  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan1_16.gif  Trigger Sua\_DonGiaHangHoa được tạo và gắn với table DON\_GIA\_HANG\_HOA. Khi thao tác Update xảy ra trên table DON\_GIA\_HANG\_HOA, trigger Sua\_DonGiaHangHoa được kích hoạt. Nhiệm vụ chủ yếu của các câu lệnh bên trong trigger này là cập nhật lại đơn giá hiện hành trên table DM\_HANG\_HOA. Trong ví dụ này, chúng tôi muốn trình bày vấn đề ở mức đơn giản nên đã đưa vào câu lệnh "IF (Select Count(\*) From Inserted)>1" với mục tiêu giới hạn chỉ cho phép cập nhật tối đa một mẫu tin trong một thao tác cập nhật. *Còn tiếp* |

|  |  |
| --- | --- |
| **Lập trình trigger – Phần 2**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_d.gif |   **Trên thực tế, trong bối cảnh người sử dụng chương trình là người Việt Nam, các câu thông báo sử dụng trong chương trình phải là ngôn ngữ tiếng Việt và sao cho phải thích hợp với đối tượng sử dụng chương trình, do vậy khi xây dựng các kiểm tra ràng buộc dữ liệu bạn buộc phải sử dụng trigger với câu lệnh RAISERROR.** |
| 2.2/- Mệnh đề (COLUMNS\_UPDATE()) :  a/- Ý nghĩa sử dụng :  Kiểm tra các cột đã được Update. Thay vì trả về giá trị TRUE hoặc FALSE, (COLUMNS\_UPDATE()) trả về một giá trị dạng nhị phân có ý nghĩa chỉ ra những cột nào được kiểm tra khi insert hoặc update.  b/- Cách sử dụng cho trường hợp có không quá 8 cột :  Để chỉ định những cột nào cần kiểm tra, bạn sử dụng một bit mặt nạ (mask) tượng trưng cho vị trí mỗi cột trong table.  Bảng liệt kê sau đây biểu diễn 8 cột đầu tiên của table và bit mặt nạ được gán cho mỗi cột.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_1.gif  Để kiểm tra cột 4 hoặc 6 đã có được Update hay không, bạn có thể sử dụng đoạn chương trình sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_2.gif  Để kiểm tra cột 4 và 6 cả hai đã có được Update hay không, bạn có thể sử dụng đoạn chương trình sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_3.gif  Giá trị 40 có được từ việc cộng giá trị 8 của cột 4 và giá trị 32 của cột 6.  Bạn có thể tính toán giá trị này dựa trên công thức sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_4.gif  Với a = 0 hoặc 1 tương ứng quy tắc : nếu bạn muốn kiểm tra thì bật a=1, ngược lại a=0; Giá trị của b=2; Giá trị n= số cột -1; U = giá trị tính được.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_5.gif  Trong ví dụ trên, có 8 cột và muốn kiểm tra cột 4 hoặc cột 6. Bạn tuân theo quy tắc : cột cần kiểm tra bật lên 1 và các cột còn lại gán giá trị 0. Áp dụng công thức trên, ta có :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_6.gif  *Chú ý :*  Vậy giá trị tương ứng với cột thứ I sẽ là 2i-1=power(2,(i-1))  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_7.gif  c/- Cách sử dụng cho trường hợp có nhiều hơn 8 cột :  Trong trường hợp có nhiều hơn 8 cột, bạn phải sử dụng hàm SUBSTRING để trigger hiểu bit mặt nạ cần kiểm tra.  Nếu cột kiểm tra là một cột thuộc 8 cột đầu tiên, bạn có thể sử dụng cộng thức sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_8.gif  Nếu cột kiểm tra là một cột khác 8 cột đầu tiên, cột thứ 9 sẽ bắt đầu lại tương ứng là 1 và bạn có thể sử dụng công thức sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_9.gif  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_10.gif  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_11.gif  2.3/- Các hàm và lệnh hệ thống :  **• Hàm @@ROWCOUNT :** Trả về số dòng bị ảnh hưởng bởi câu lệnh T-SQL ngay trước đó trong trigger.  **• Câu lệnh RETURN :** Nếu không có dòng nào bị ảnh hưởng bởi biến cố INSERT, UPDATE hoặc DELETE, trigger vẫn kích hoạt. Trong trường hợp này, bạn có thể sử dụng câu lệnh RETURN để chấm dứt việc thi hành không cần thiết các câu lệnh của trigger. Lệnh RETURN được sử dụng tại các vị trí trong trigger mà tại đ1o bạn muốn chấm dứt việc thi hành tiếp tục các câu lệnh khác của trigger.  **• Câu lệnh RAISERROR :** Được sử dụng để hiển thị thông báo lỗi. Bạn cũng có thể dùng nó cho việc gởi ra màn hình câu thông báo mang tính chất cảnh báo. Chuỗi thông báo khi sử dụng RAISERROR sẽ được gởi về cho client từ server.  Trên thực tế, trong bối cảnh người sử dụng chương trình là người Việt Nam, các câu thông báo sử dụng trong chương trình phải là ngôn ngữ tiếng Việt và sao cho phải thích hợp với đối tượng sử dụng chương trình, do vậy khi xây dựng các kiểm tra ràng buộc dữ liệu bạn buộc phải sử dụng trigger với câu lệnh RAISERROR (Mặc dù, về nguyên tắc chỉ những ràng buộc nào không thể sử dụng các constraint mới sử dụng trigger).  *Ví dụ :*  Viết trigger kiểm tra ràng buộc khóa ngoại trên table DON\_GIA\_HANG\_HOA với yêu cầu : khi thêm mới mẫu tin trên table này, kiểm tra giá trị trên cột ID\_HH, nếu giá trị này chưa có trong table DM\_HANG\_HOA thì bỏ qua thao tác thêm và hiển thị câu thông báo lỗi.  Giả sử trước đó ta đã có constraint FK\_DON\_GIA\_H\_HienH\_1BF\_D2C07 dùng để kiểm tra khóa ngoại ID\_HH. Vậy trước hết, bạn cần tắt constraint này đi bằng cách sử dụng câu lệnh sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_12.gif  Nội dung của trigger có thể được viết như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_13.gif  **• Câu lệnh ROLLBACK TRANSACTION :** Đây là câu lệnh quan trọng rất hay dùng trong trigger, nó được sử dụng trong trường hợp bạn muốn bỏ qua toàn bộ thao tác trước đó là nguyên nhân kích hoạt trigger hoặc quay lui toàn bộ lô xử lý (batch) của trigger. Một lỗi không xác định nào đó cũng làm cho quay lui toàn bộ giao tác. Bạn không nên đặt câu lệnh ROLLBACK TRANSACTION trong trigger nếu mục tiêu của bạn là hoàn tất giao tác trong tất cả các trường hợp, ngoại trừ có lỗi không xác định nào đó xảy ra trong lúc thực hiện giao tác.  2.4/- Các câu lệnh T-SQL không được chấp nhận trong một trigger :  Các câu lệnh được liệt kê sau đây không được chấp nhận trong một trigger : ALTER, CREATE, DROP, RESTORE và LOAD DATABASE.  • LOAD và RESTORE LOG  • DISK RESIZE và DISK INIT  • RECONFIGURE  *Chú ý :*  Bạn có thể sử dụng các câu lệnh SELECT và PRINT hoặc gán giá trị cho các biến trong đoạn chương trình của trigger nhưng việc sử dụng chúng để trả về tập kết quả, các thông báo hoặc các giá trị là điều nguy hiểm nên tránh.  *Còn tiếp* |
|  |
| |  | | --- | | [**Home**](http://www.thuvien-it.net/home/tinhoc/default.asp) >> [**Cơ Sở Dữ Liệu**](http://www.thuvien-it.net/home/tinhoc/cat.asp?cid=3) >> [**MS SQL Server**](http://www.thuvien-it.net/home/tinhoc/sub.asp?sid=74)>> | | **Lập trình trigger – Phần 2**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_d.gif |   **Trên thực tế, trong bối cảnh người sử dụng chương trình là người Việt Nam, các câu thông báo sử dụng trong chương trình phải là ngôn ngữ tiếng Việt và sao cho phải thích hợp với đối tượng sử dụng chương trình, do vậy khi xây dựng các kiểm tra ràng buộc dữ liệu bạn buộc phải sử dụng trigger với câu lệnh RAISERROR.** | | 2.2/- Mệnh đề (COLUMNS\_UPDATE()) :  a/- Ý nghĩa sử dụng :  Kiểm tra các cột đã được Update. Thay vì trả về giá trị TRUE hoặc FALSE, (COLUMNS\_UPDATE()) trả về một giá trị dạng nhị phân có ý nghĩa chỉ ra những cột nào được kiểm tra khi insert hoặc update.  b/- Cách sử dụng cho trường hợp có không quá 8 cột :  Để chỉ định những cột nào cần kiểm tra, bạn sử dụng một bit mặt nạ (mask) tượng trưng cho vị trí mỗi cột trong table.  Bảng liệt kê sau đây biểu diễn 8 cột đầu tiên của table và bit mặt nạ được gán cho mỗi cột.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_1.gif  Để kiểm tra cột 4 hoặc 6 đã có được Update hay không, bạn có thể sử dụng đoạn chương trình sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_2.gif  Để kiểm tra cột 4 và 6 cả hai đã có được Update hay không, bạn có thể sử dụng đoạn chương trình sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_3.gif  Giá trị 40 có được từ việc cộng giá trị 8 của cột 4 và giá trị 32 của cột 6.  Bạn có thể tính toán giá trị này dựa trên công thức sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_4.gif  Với a = 0 hoặc 1 tương ứng quy tắc : nếu bạn muốn kiểm tra thì bật a=1, ngược lại a=0; Giá trị của b=2; Giá trị n= số cột -1; U = giá trị tính được.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_5.gif  Trong ví dụ trên, có 8 cột và muốn kiểm tra cột 4 hoặc cột 6. Bạn tuân theo quy tắc : cột cần kiểm tra bật lên 1 và các cột còn lại gán giá trị 0. Áp dụng công thức trên, ta có :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_6.gif  *Chú ý :*  Vậy giá trị tương ứng với cột thứ I sẽ là 2i-1=power(2,(i-1))  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_7.gif  c/- Cách sử dụng cho trường hợp có nhiều hơn 8 cột :  Trong trường hợp có nhiều hơn 8 cột, bạn phải sử dụng hàm SUBSTRING để trigger hiểu bit mặt nạ cần kiểm tra.  Nếu cột kiểm tra là một cột thuộc 8 cột đầu tiên, bạn có thể sử dụng cộng thức sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_8.gif  Nếu cột kiểm tra là một cột khác 8 cột đầu tiên, cột thứ 9 sẽ bắt đầu lại tương ứng là 1 và bạn có thể sử dụng công thức sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_9.gif  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_10.gif  *Ví dụ :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_11.gif  2.3/- Các hàm và lệnh hệ thống :  **• Hàm @@ROWCOUNT :** Trả về số dòng bị ảnh hưởng bởi câu lệnh T-SQL ngay trước đó trong trigger.  **• Câu lệnh RETURN :** Nếu không có dòng nào bị ảnh hưởng bởi biến cố INSERT, UPDATE hoặc DELETE, trigger vẫn kích hoạt. Trong trường hợp này, bạn có thể sử dụng câu lệnh RETURN để chấm dứt việc thi hành không cần thiết các câu lệnh của trigger. Lệnh RETURN được sử dụng tại các vị trí trong trigger mà tại đ1o bạn muốn chấm dứt việc thi hành tiếp tục các câu lệnh khác của trigger.  **• Câu lệnh RAISERROR :** Được sử dụng để hiển thị thông báo lỗi. Bạn cũng có thể dùng nó cho việc gởi ra màn hình câu thông báo mang tính chất cảnh báo. Chuỗi thông báo khi sử dụng RAISERROR sẽ được gởi về cho client từ server.  Trên thực tế, trong bối cảnh người sử dụng chương trình là người Việt Nam, các câu thông báo sử dụng trong chương trình phải là ngôn ngữ tiếng Việt và sao cho phải thích hợp với đối tượng sử dụng chương trình, do vậy khi xây dựng các kiểm tra ràng buộc dữ liệu bạn buộc phải sử dụng trigger với câu lệnh RAISERROR (Mặc dù, về nguyên tắc chỉ những ràng buộc nào không thể sử dụng các constraint mới sử dụng trigger).  *Ví dụ :*  Viết trigger kiểm tra ràng buộc khóa ngoại trên table DON\_GIA\_HANG\_HOA với yêu cầu : khi thêm mới mẫu tin trên table này, kiểm tra giá trị trên cột ID\_HH, nếu giá trị này chưa có trong table DM\_HANG\_HOA thì bỏ qua thao tác thêm và hiển thị câu thông báo lỗi.  Giả sử trước đó ta đã có constraint FK\_DON\_GIA\_H\_HienH\_1BF\_D2C07 dùng để kiểm tra khóa ngoại ID\_HH. Vậy trước hết, bạn cần tắt constraint này đi bằng cách sử dụng câu lệnh sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_12.gif  Nội dung của trigger có thể được viết như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan2_13.gif  **• Câu lệnh ROLLBACK TRANSACTION :** Đây là câu lệnh quan trọng rất hay dùng trong trigger, nó được sử dụng trong trường hợp bạn muốn bỏ qua toàn bộ thao tác trước đó là nguyên nhân kích hoạt trigger hoặc quay lui toàn bộ lô xử lý (batch) của trigger. Một lỗi không xác định nào đó cũng làm cho quay lui toàn bộ giao tác. Bạn không nên đặt câu lệnh ROLLBACK TRANSACTION trong trigger nếu mục tiêu của bạn là hoàn tất giao tác trong tất cả các trường hợp, ngoại trừ có lỗi không xác định nào đó xảy ra trong lúc thực hiện giao tác.  2.4/- Các câu lệnh T-SQL không được chấp nhận trong một trigger :  Các câu lệnh được liệt kê sau đây không được chấp nhận trong một trigger : ALTER, CREATE, DROP, RESTORE và LOAD DATABASE.  • LOAD và RESTORE LOG  • DISK RESIZE và DISK INIT  • RECONFIGURE  *Chú ý :*  Bạn có thể sử dụng các câu lệnh SELECT và PRINT hoặc gán giá trị cho các biến trong đoạn chương trình của trigger nhưng việc sử dụng chúng để trả về tập kết quả, các thông báo hoặc các giá trị là điều nguy hiểm nên tránh.  *Còn tiếp* | |
| **Lập trình trigger – Phần 3**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_d.gif |   **Khái niệm trigger lồng nhau hoàn toàn giống khái niệm thủ tục lồng nhau. Bản thân bên trong trigger, chúng ta được phép gọi thực hiện các lệnh INSERT, UPDATE, DELETE để cập nhật dữ liệu của các bảng khác.** |
| 3/- Các thao tác lập trình trigger phổ biến  3.1/- Trigger lồng nhau :  Khái niệm trigger lồng nhau hoàn toàn giống khái niệm thủ tục lồng nhau. Bản thân bên trong trigger, chúng ta được phép gọi thực hiện các lệnh INSERT, UPDATE, DELETE để cập nhật dữ liệu của các bảng khác, chính các lệnh này sẽ làm kích hoạt các trigger liên quan khác (nếu có) và cứ thế các trigger có thể gọi thực hiện lồng nhau.  Cấp độ lồng tối đa của các trigger không vượt quá 32 cấp. Chúng ta cũng có thể sử dụng biến hệ thống @@NESTLEVEL để biết được cấp độ lồng hiện hành của trigger. Mặc định các trigger được phép lồng nhau. Tuy nhiên chúng ta cũng có thể tạm thời tắt chế độ lồng của trigger bằng lệnh.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_1.gif  Hoặc có thể bật trở lại chế độ lồng nhau của trigger bằng lệnh.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_2.gif  *Ví dụ :*  Thông thường khi dữ liệu cập nhật trong các bảng CTPXUAT hoặc CTPNHAP thì chúng ta phải cập nhật tự động các cột : tổng số lượng nhập, tổng số lượng xuất và số lượng cuối kỳ của các vật tư tương ứng trong bảng TONKHO. Các xử lý này có thể chia ra làm hai nơi :  • Các xử lý cập nhật cột tổng số lượng nhập hoặc tổng số lượng xuất được viết trong trigger của các bảng CTPNHAP hoặc CTPXUAT.  • Xử lý cập nhật lại giá trị của cột số lượng cuối kỳ được viết trong trigger của bảng TONKHO.  Khi đó bản thân các trigger của các bảng CTPNHAP và CTPXUAT sẽ thực hiện các lệnh cập nhật dữ liệu trên bảng TONKHO, lúc này trigger của bảng TONKHO sẽ được kích hoạt thực hiện chỉ để tính lại giá trị cột số lượng cuối kỳ khi giá trị các cột số lượng nhập hoặc số lượng xuất bị thay đổi. Chúng ta thực hiện lệnh CREATE TRIGGER như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_3.gif  3.2/- Trigger kiểm tra ràng buộc dữ liệu :  Trong các phần còn lại, chúng tôi sẽ trình bày chi tiết các xử lý bên trong trigger để có thể thực hiện việc kiểm tra các ràng buộc toàn vẹn dữ liệu phức tạp. Để dễ hiểu chúng tôi chia các hành động cập nhật dữ liệu ra làm 3 sự kiện rời rạc : thêm, sửa và xóa nhằm giúp các bạn phân biệt rõ ràng các xử lý thường dùng bên trong các sự kiện đó.  3.3/- Khi thêm mới mẫu tin :  Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng được thêm mới vào bảng dữ liệu. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là :  • Khóa ngoại.  • Miền giá trị.  • Liên thuộc tính trong cùng một bảng.  • Liên thuộc tính của nhiều bảng khác nhau.  Khi các giá trị dữ liệu thêm mới vi phạm các ràng buộc toàn vẹn dữ liệu thì trigger sẽ thông báo cho người dùng biết và không lưu lại các thông tin của dòng dữ liệu vừa được thêm mới vào bên trong bảng. Trong các trigger thêm mới dữ liệu thì các dữ liệu vừa mới thêm vào sẽ được lưu trữ tạm thời trong bảng Inserted, do đó chúng ta sẽ tham chiếu đến bảng Inserted để lấy ra các giá trị dữ liệu vừa mới thêm dùng trong những kiểm tra ràng buộc toàn vẹn dữ liệu.  *Ví dụ :*  Xây dựng trigger trong bảng PNHAP để kiểm tra các ràng buộc toàn vẹn dữ liệu khi người dùng thêm mới thông tin của một phiếu nhập hàng cho một đơn đặt hàng trước đó. Các ràng buộc toàn vẹn dữ liệu bao gồm :  **• Khóa ngoại :** cần kiểm tra số đặt hàng phải tồn tại trong bảng đơn đặt hàng.  **• Miền giá trị :** cần kiểm tra ngày giao hàng phải sau ngày đặt hàng.  Chúng ta thực hiện lệnh CREATE TRIGGER như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_4.gif  Để kiểm tra các hoạt động bên trong trigger có đúng hay không, chúng ta cần phải lần lượt thực hiện các lệnh INSERT INTO để thêm dữ liệu vào bảng PNHAP cho các trường hợp mà chúng ta đã biện luận trong trigger.  **Trường hợp 1 :** Vi phạm ràng buộc toàn vẹn số đơn đặt hàng không tồn tại trong bảng DONDH.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_5.gif  Kết quả lỗi trả về :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_6.gif  **Trường hợp 2 :** Vi phạm ràng buộc toàn vẹn ngày giao hàng trước ngày đặt hàng.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_7.gif  Kết quả lỗi trả về :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_8.gif  Cuối cùng khi nhập thông tin của một phiếu nhập hàng không vi phạm các ràng buộc toàn vẹn dữ liệu.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_9.gif  Kết quả trả về hành động thêm dữ liệu mới thành công.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan3_10.gif  *Còn tiếp* |
|  |
| |  | | --- | | [**Home**](http://www.thuvien-it.net/home/tinhoc/default.asp) >> [**Cơ Sở Dữ Liệu**](http://www.thuvien-it.net/home/tinhoc/cat.asp?cid=3) >> [**MS SQL Server**](http://www.thuvien-it.net/home/tinhoc/sub.asp?sid=74)>> | | **Lập trình trigger – Phần 4**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_d.gif |   **Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng bị sửa đổi. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là : kiểm tra ràng buộc toàn vẹn dữ liệu khóa ngoại, miền giá trị, liên thuộc tính trong cùng một bảng dữ liệu, liên thuộc tính của nhiều bảng dữ liệu khác nhau.** | | **3.4/- Khi hủy bỏ mẫu tin :**  Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu bên trong bảng bị hủy bỏ. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là : kiểm tra ràng buộc toàn vẹn dữ liệu khóa ngoại dùng để xóa tự động dữ liệu bên bản con có liên quan hoặc thông báo lỗi đã vi phạm ràng buộc toàn vẹn khi xóa dữ liệu bên bảng cha.  Khi giá trị dữ liệu trong bảng bị hủy nếu có vi phạm ràng buộc toàn vẹn dữ liệu khóa ngoại thì trigger sẽ thông báo cho người dùng biết và không hủy thông tin của các dòng dữ liệu. Trong các trigger hủy bỏ dữ liệu thì dữ liệu vừa bị hủy bỏ sẽ được lưu trữ tạm thời trong bảng Deleted., chúng ta sẽ tham chiếu đến bảng Deleted để lấy ra giá trị của các dữ liệu vừa bị hủy bỏ dùng cho các kiểm tra ràng buộc toàn vẹn dữ liệu khóa ngoại.  *Ví dụ :*  Khi xóa một số đặt hàng bên trong bảng DONDH cần phải kiểm tra các ràng buộc toàn vẹn dữ liệu sau :  • Kiểm tra xem đơn đặt hàng bị xóa đã được nhập hàng chưa ? Nếu đã được nhập hàng rồi thì thông báo không thể xóa đơn đặt hàng được.  • Ngược lại thì xóa tự động dữ liệu liên quan bên bảng chi tiết đơn đặt hàng (CTDONDH).  Chúng ta thực hiện lệnh CREATE TRIGGER như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_1.gif  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_2.gif  Cũng hoàn toàn giống như ví dụ trên, chúng ta cần phải lần lượt thực hiện các lệnh DELETE để hủy bỏ các dòng dữ liệu trong bảng DONDH nhằm kiểm tra tính đúng đắn của các hành động bên trong trigger chúng ta đã viết.  **Trường hợp 1 :**  Vi phạm ràng buộc toàn vẹn khóa ngoại vì số đơn đặt hàng D002 đã được nhập hàng rồi.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_3.gif  Kết quả lỗi trả về :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_4.gif  **Trường hợp 2 :**  Xóa thành công toàn bộ thông tin đơn đặt hàng D006 trong cả hai bảng DONDH và CTDONDH (vì đơn đặt hàng này chưa được nhập hàng về).  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_5.gif  Kết quả trả về 3 dòng bị xóa trong bảng CTDONDH và 1 dòng trong DONDH.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_6.gif  **3.5/- Khi sửa đổi mẫu tin :**  Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng bị sửa đổi. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là : kiểm tra ràng buộc toàn vẹn dữ liệu khóa ngoại, miền giá trị, liên thuộc tính trong cùng một bảng dữ liệu, liên thuộc tính của nhiều bảng dữ liệu khác nhau. Tuy nhiên thông thường đối với việc thay đổi dữ liệu trên bảng chúng ta sẽ hạn chế việc sửa đổi dữ liệu, chỉ cho phép người sử dụng sửa đổi dữ liệu trên một số cột nhất định nào đó bên trong bảng.  Để kiểm tra giá trị dữ liệu của một cột bên trong bảng có bị thay đổi trong các trigger sửa đổi dữ liệu, chúng ta sẽ sử dụng hàm UPDATE. Cú pháp đầy đủ của hàm UPDATE sẽ được mô tả như sau :  *Cú pháp :*  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_7.gif  Trong đó :  **• Tên cột :** tên cột mà chúng ta muốn kiểm tra xem dữ liệu tại đó có bị sửa đổi trong trigger không.  **• Biểu thức luận lý :** trả về True khi giá trị dữ liệu của cột đã bị sửa đổi, ngược lại trả về False khi giá trị dữ liệu của cột không bị thay đổi.  Hành động sửa đổi dữ liệu bên dưới của Microsoft SQL Server thật chất là sự kết hợp của hai hành động đi kèm là : xóa dữ liệu cũ hiện có và thêm lại dữ liệu mới đã được sửa đổi. Do đó bên trong trigger sửa đổi dữ liệu khi đó bảng Inserted sẽ chứa đựng dữ liệu mới sau khi sửa đổi và bảng Deleted sẽ chứa đựng dữ liệu cũ trước khi sửa đổi. Thông thường trong trigger sửa đổi chúng ta có thể tham chiếu đến cùng lúc hai bảng luận lý Inserted và Deleted.  *Ví dụ :*  Khi sửa đổi thông tin của một số đặt hàng bên trong bảng DONDH cần phải kiểm tra các ràng buộc toàn vẹn dữ liệu sau :  • Không cho phép sửa đổi dữ liệu tại các cột số đặt hàng hoặc mã nhà cung cấp vì khi đó dữ liệu sẽ bị ảnh hưởng đến nhiều bảng liên quan khác.  • Khi sửa đổi giá trị cột ngày đặt hàng thì phải đảm bảo luôn luôn trước ngày nhập hàng đầu tiên của số đặt hàng đó (nếu đơn đặt hàng đã có nhập hàng).  Chúng ta thực hiện lệnh CREATE TRIGGER như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_8.gif  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_9.gif  Chúng ta cần phải lần lượt thực hiện các lệnh UPDATE để sửa đổi dữ liệu trong bảng DONDH nhằm kiểm tra tính đúng đắn của trigger đã viết.  **Trường hợp 1 :** Sửa đổi dữ liệu cột số đặt hàng hoặc cột mã nhà cung cấp.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_10.gif  Hoặc  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_11.gif  Kết quả lỗi trả về :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_12.gif  **Trường hợp 2 :** Đơn đặt hàng D002 có ngày đặt hàng là 30/01/2002, ngày nhập hàng dự kiến là 02/02/2002 và đã được nhập hàng lần đầu vào ngày 31/01/2002. Giả sử chúng ta ra lệnh để sửa đổi giá trị ngày đặt hàng lại thành ngày 01/02/2002.  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_13.gif  Kết quả lỗi trả về :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan4_14.gif  *Còn tiếp* | |
|  |
| **Lập trình trigger – Phần 5**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan5_d.gif |   **Khi giá trị của các dữ liệu thêm mới đã được kiểm tra hợp lệ so với các ràng buộc toàn vẹn dữ liệu thì các dữ liệu này sẽ được cập nhật tăng giá trị tại một cột nào đó của các bảng liên quan.</** |
| **3.6/- Trigger cập nhật giá trị tự động :**  Bên cạnh việc kiểm tra các ràng buộc toàn vẹn dữ liệu khi dữ liệu bị cập nhật bởi người sử dụng, bên trong trigger còn có thêm một tính năng ưu việt nữa đó là việc tính toán các giá trị và cập nhật tự động vào các cột của một bảng dữ liệu nào đó. Thông thường dữ liệu của các bảng này được hình thành dựa vào số liệu của một hoặc nhiều bảng khác bên trong cơ sở dữ liệu.  *Ví dụ :*  Dữ liệu của bảng TONKHO sẽ được tính tự động từ dữ liệu của các bảng liên quan đến việc nhập hàng và việc xuất hàng, cụ thể sẽ là các bảng : PNHAP, CTPNHAP, PXUAT và CTPXUAT.  Trong phần cuối, chúng tôi muốn giới thiệu thêm các xử lý cập nhật giá trị tự động bên trong các trigger, các xử lý này cũng sẽ được chia ra làm thành các sự kiện rời rạc nhằm giúp các bạn dễ hiểu. Tuy nhiên chúng ta không nên viết các xử lý cập nhật giá trị tự động này thành một trigger mới hoàn toàn mà chỉ cần bổ sung các xử lý này vào bên dưới trong các trigger đã có. Bởi vì các xử lý cập nhật giá trị tự động chỉ được thực hiện khi nào các ràng buộc toàn vẹn dữ liệu đã được kiểm tra hợp lệ trước đó.  **3.7/- Khi thêm mới mẫu tin :**  Khi giá trị của các dữ liệu thêm mới đã được kiểm tra hợp lệ so với các ràng buộc toàn vẹn dữ liệu thì các dữ liệu này sẽ được cập nhật tăng giá trị tại một cột nào đó của các bảng liên quan.  *Ví dụ :*  Khi thêm mới các thông tin của chi tiết một phiếu nhập hàng vào bảng CTPNHAP, chúng ta cần kiểm tra các ràng buộc toàn vẹn dữ liệu :  • Kiểm tra số phiếu nhập phải có trong bảng PNHAP.  • Kiểm tra mã vật tư phải có trong danh sách chi tiết danh sách các mã vật tư có trong chi tiết đơn đặt hàng trước đó.  • Kiểm tra tổng số lượng nhập hàng vẫn còn ít hơn số lượng đặt hàng của vật tư đó.  Nếu tất cả các ràng buộc toàn vẹn dữ liệu ở trên đều hợp lệ thì tăng giá trị của cột tổng số lượng nhập trong bảng TONKHO và cột tổng giá trị trong bảng PNHAP. Chúng ta thực hiện lệnh CREATE TRIGGER như sau :  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan5_1.gif  http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan5_2.gif  *Còn tiếp* |
| **Lập trình trigger – Phần 6**   |  | | --- | | http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan6_d.gif |   **Khi hủy bỏ thông tin chi tiết các phiếu nhập hàng trong bảng CTPNHAP, chúng ta sẽ giảm giá trị cột tổng số lượng nhập trong bảng TONKHO và cột tổng giá trị nhập trong bảng PNHAP.** |
| **3.8/- Khi hủy bỏ mẫu tin :**  **Ngược lại với việc thêm mới, khi giá trị các dòng dữ liệu trong bảng bị hủy bỏ thì chúng ta cập nhật giảm giá trị tại cột nào đó của các bảng liên quan. Ở các trigger trong các ví dụ trước chỉ áp dụng đúng khi việc cập nhật dữ liệu trên cột và chỉ một dòng dữ liệu mà thôi, nếu việc cập nhập tác động cùng lúc trên nhiều dòng dữ liệu thì các trigger trên sẽ hoạt động không còn đúng nữa.**  **Do đó trong ví dụ bên dưới, chúng tôi muốn minh họa cho các bạn thấy được cách viết trigger áp dụng trong các trường hợp người dùng cập nhật dữ liệu cùng lúc một hoặc nhiều dòng dữ liệu. Về nguyên tắc chung thường phải sử dụng hàm SUM để tính ra số tổng của cùng một nhóm dữ liệu và cập nhật giá trị đó vào trong các bảng khác.**  **Ví dụ :**  **Khi hủy bỏ thông tin chi tiết các phiếu nhập hàng trong bảng CTPNHAP, chúng ta sẽ giảm giá trị cột tổng số lượng nhập trong bảng TONKHO và cột tổng giá trị nhập trong bảng PNHAP. Chúng ta thực hiện lệnh CREATE TRIGGER như sau :**  **http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan6_1.gif**  **http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan6_2.gif**  **3.9/- Khi sửa đổi mẫu tin :**  **Giống như các hành động thêm và hủy bỏ trước đây, sau khi mẫu tin bị sửa đổi, chúng ta phải tính toán để cập nhật lại giá trị các cột của các bảng có liên quan khớp với giá trị mới vừa được sửa đổi. Việc cập nhập này có thể là tăng hoặc giảm so với giá trị cũ trước đó hòan toàn tùy thuộc vào giá trị chênh lệch giữa giá trị mới được sửa đổi và giá trị cũ trước đó.**  **Do thế nguyên tắc chung trong các trigger sửa đổi dữ liệu là cập nhật lại giá trị chênh lệch sau khi và trước khi sửa đổi dữ liệu, tùy thuộc vào số chênh lệch này là âm hoặc dương mà giá trị được cập nhật sẽ giảm hoặc tăng so với giá trị hiện hành.**  **Ví dụ :**  **Khi sửa đổi thông tin của chi tiết một phiếu nhập hàng vào bảng CTPNHAP, chúng ta cần kiểm tra các ràng buộc toàn vẹn dữ liệu :**  **• Không cho phép sửa đổi các cột : số phiếu nhập hoặc mã vật tư.**  **• Số lượng nhập hàng sau khi sửa đổi phải đảm bảo : tổng số lượng nhập không vượt quá số lượng đặt hàng ban đầu.**  **Nếu tất cả các ràng buộc toàn vẹn dữ liệu ở trên đều hợp lệ thì cập nhật lại giá trị của cột tổng số lượng nhập trong bảng TONKHO và cột tổng giá trị trong bảng PNHAP. Chúng ta thực hiện lệnh CREATE TRIGGER như sau :**  **http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan6_3.gif**  **http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan6_4.gif**  **http://www.thuvien-it.net/home/tinhoc/2010/12/07/Laptrinhtrigger_Phan6_5.gif**  **Tóm lại, việc phân chia, tổ chức các kiểm tra ràng buộc toàn vẹn dữ liệu phức tạp hoặc các cập nhật dữ liệu tự động trong đối tượng trigger sẽ làm cho các xử lý được tập trung tại máy chủ và độc lập với ngôn ngữ lập trình tại máy trạm. Điều này làm cho tốc độ của các ứng dụng theo mô hình khách chủ được nhanh hơn.**  **Hết** |