chương 2

Ràng buộc, Trình kích hoạt, Lượt xem

Hệ thống cơ sở dữ liệu - Cuốn sách hoàn chỉnh, Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom

Đề cương chương

- Khóa và Ràng buộc khóa ngoại trên
- các thuộc tính và Tuples Sửa đổi các
- ràng buộc
- Cò súng
- Lượt xem

Khai báo khóa ngoại
Hạn chế

- Trong SQL, chúng ta có thể khai báo một thuộc tính hoặc các thuộc tính của một quan hệ là khóa ngoại, tham chiếu đến một số thuộc tính của quan hệ thứ hai (có thể là cùng một quan hệ)
- (Các) thuộc tính được tham chiếu của quan hệ thứ hai phải được khai báo DUY NHẤT hoặc KHÓA CHÍNH cho quan hệ của chúng. Nếu không, chúng tôi không thể thực hiện khai báo khóa ngoại.
- Các giá trị của khóa ngoại xuất hiện trong quan hệ đầu tiên cũng phải xuất hiện trong các thuộc tính được tham chiếu của một số bộ.

Hai cách để khai báo khóa ngoại.

- Ví dụ:

Giả sử chúng ta muốn khai báo mối quan hệ

Phòng thu(Tên , Địa chỉ,presC #)

cái nào có khóa ngoài***presC #***tài liệu tham khảo đó***cert #*** của mối quan hệ:

MovieExec (tên, addreses, cert #, netWorth)

Chúng tôi có thể khai báo presC # trực tiếp với cert # tham chiếu như sau:

TẠO BẢNG Studio (

tên CHAR (30) PRIMARY KEY,

địa chỉ VARCHAR (255),

presC # INT***NGƯỜI GIỚI THIỆU***

MovieExec (cert #$_4$));

Hai lối để khai báo khóa ngoại (tiếp).

- Ví dụ:

Một hình thức thay thế là thêm khai báo khóa ngoại một cách riêng biệt, như

TẠO BẢNG Studio (

tên CHAR (30) PRIMARY KEY,

địa chỉ VARCHAR (255),

presC # INT,

TÀI LIỆU THAM KHẢO TỪ KHÓA NGOẠI TỆ (presC #)

MovieExec (cert #)

);

Duy trì tính toàn vẹn tham chiếu.

- Các hành động sau sẽ bị ngăn chặn bởi DBMS (nghĩa là, một ngoại lệ hoặc lỗi trong thời gian chạy sẽ được tạo ra).

Một)Chèn một tuple Studio mới có giá trị presC # không phải là NULL và không phải là thành phần cert # của bất kỳ tuple MovieExec nào. Là

b)Cập nhật bộ giá trị Studio để thay đổi thành phần presC # thành giá trị không phải NULL không phải là thành phần cert # của bất kỳ bộ tuple MovieExec nào.

C)Xóa bộ mã MovieExec và thành phần cert # của nó, không phải là NULL, xuất hiện dưới dạng thành phần presC # của một hoặc nhiều bộ giá trị Studio.

d)Cập nhật bộ mã MovieExec theo cách thay đổi giá trị cert # và cert # cũ là giá trị của presC # của một số studio phim.

Duy trì tính toàn vẹn tham chiếu (tiếp)

- Đối với hai sửa đổi đầu tiên, trong đó thay đổi liên quan đến mối quan hệ mà ràng buộc khóa ngoại được khai báo, không có thay thế nào; hệ thống phải từ chối sửa đổi vi phạm.
- Đối với các thay đổi đối với quan hệ được tham chiếu, trong đó hai sửa đổi cuối cùng là ví dụ, nhà thiết kế có thể chọn trong số ba tùy chọn:

1. Chính sách mặc định: Từ chối các sửa đổi vi phạm.
2. Chính sách phân tầng: Theo chính sách này, các thay đổi đối với (các) thuộc tính được tham chiếu sẽ được bắt chước ở khóa ngoại.
3. Chính sách Set-Null.

Duy trì tính toàn vẹn tham chiếu (tiếp)

Ví dụ:

TẠO BẢNG Studio (

tên CHAR (30) PRIMARY KEY,

địa chỉ VARCHAR (255),

presC # INT TÀI LIỆU THAM KHẢO

MovieExec (cert #)

ON DELETE SET NULL

TRÊN CẬP NHẬT CASCADE

);

Ràng buộc về Thuộc tính và Bộ quy tắc.

- **Ràng buộc Not-Null:** Ví dụ:

```
TẠO BẢNG Studio (
    tên CHAR (30) PRIMARY KEY,
    địa chỉ VARCHAR (255),
    presC # INT TÀI LIỆU THAM KHẢO MovieExec (cert #)
                                                CÓ GIÁ TRỊ
    );
```

- Chúng tôi không thể chèn một bộ vào Studio bằng cách chỉ xác định tên và địa chỉ.
- Chúng tôi không thể sử dụng chính sách set-null

Các ràng buộc về Thuộc tính và Bộ quy tắc (tiếp).

- **Ràng buộc KIỂM TRA Dựa trên thuộc tính:**

Ví dụ: Giả sử chúng ta muốn yêu cầu số chứng chỉ phải có ít nhất sáu chữ số.

```
TẠO BẢNG Studio (
    tên CHAR (30) PRIMARY KEY,
    địa chỉ VARCHAR (255),
    presC # INT TÀI LIỆU THAM KHẢO MovieExec (cert #)
                                KIỂM TRA (presC #> = 100000)
    );
```

- Chúng tôi không thể chèn một bộ vào Studio bằng cách chỉ xác định tên và địa chỉ.
- Chúng tôi không thể sử dụng chính sách set-null

Các ràng buộc về Thuộc tính và Bộ quy tắc (tiếp).

- **Ràng buộc KIỂM TRA dựa trên**

Tuple: Ví dụ:

```
TẠO BẢNG MovieStar (
        tên CHAR (30) PRIMARY KEY,
        địa chỉ VARCHAR (255),
        giới tính CHAR (1),
        ngày sinh DATE,
    KIỂM TRA (giới tính = 'F' HOẶC tên KHÔNG THÍCH 'Ms.%')
        );
```

Sửa đổi các ràng buộc.

- **Đặt tên cho các ràng buộc:** Ví dụ:

TẠO BẢNG MovieStar (
tên CHAR (30) CONSTRAINT *NamelsKey* KHÓA CHÍNH, địa chỉ VARCHAR (255),
giới tính CHAR (1),
ngày sinh DATE,
HẠN CHẾ *RightTitle*
KIỂM TRA (giới tính = 'F' HOẶC tên KHÔNG THÍCH 'Ms.%')
);

Nhớ lại, bạn nên đặt tên cho từng ràng buộc của mình, ngay cả khi bạn không tin rằng bạn sẽ cần tham khảo nó.

Sửa đổi các ràng buộc.

- **Thay đổi các ràng buộc trên bảng:**

Ví dụ:

BẢNG ALTER MovieStar**DROP CONSTRAINT** NamelsKey;

BẢNG ALTER MovieStar**THÊM CONSTRAINT**NamelsKey KHÓA CHÍNH (tên);

Gây nên.

- Trình kích hoạt là một loạt các hành động được liên kết với các sự kiện nhất định và được thực hiện bất cứ khi nào các sự kiện này phát sinh.
- Kích hoạt khác với các loại ràng buộc ở ba điểm:

1. Trigger chỉ được đánh thức khi xảy ra một số sự kiện nhất định, do người lập trình cơ sở dữ liệu chỉ định (thường là chèn, xóa hoặc cập nhật).

2. Sau khi được đánh thức bởi sự kiện kích hoạt của nó, trình kích hoạt sẽ kiểm tra một điều kiện.

3. Nếu điều kiện của trình kích hoạt được thỏa mãn, thì hành động được kết hợp với trình kích hoạt được thực hiện bởi DBMS.

Trình kích hoạt, đôi khi được gọi là **sự kiện-điều kiện-hành động quy tắc**hoặc**Quy tắc ECA**

Gây nên.

- Ví dụ. MovieExec (tên, địa chỉ, CERT #, netWorth)

1) TẠO TRIGGERNetWorthTrigger

2)SAU KHI CẬP NHẬTOF netWorth ON MovieExec

3) TÀI LIỆU THAM KHẢO

4) CŨ ROW NHƯ Cũ,

5) ROW MỚI AS NewTuple

6) CHO TỪNG ROW

7)KHI NÀO (OldTuple.netWorth> NewTuple.netWorth)

số 8) CẬP NHẬT MovieExec

9) SET netWorth = OldTuple.netWorth

10) WHERE cert # = NewTuple.cert #;

Các tùy chọn cho thiết kế kích hoạt

- Chúng tôi có thể thay thếSAUquaTRƯỚC(dòng 2) trong trường hợp đó điều kiện WHEN được kiểm tra trên trạng thái cơ sở dữ liệu tồn tại trước khi sự kiện kích hoạt được thực thi.

- bên cạnh đóCẬP NHẬT(dòng 2), các sự kiện kích hoạt có thể có khác làCHÈNvàXÓA BỎ.
- CácMệnh đề WHEN là tùy chọn. Nếu nó bị thiếu, thì hành động sẽ được thực hiện bất cứ khi nào trình kích hoạt được đánh thức.
- Có thể có bất kỳ số lượng câu lệnh nào như vậy (dòng 8-10), được phân tách bằng dấu chấm phẩy và được bao quanh bởi BEGIN ... END.

Các tùy chọn cho thiết kế kích hoạt

- Nếu chúng ta bỏ qua FOR MỖI ROW trên dòng (6) hoặc thay thế nó theo mặc định CHO MỖI TRẠNG THÁI, thì atrình kích hoạt cấp độ hàngtrở thành mộttrình kích hoạt cấp tuyên bố.
- MỘTtrình kích hoạt cấp tuyên bốđược thực thi một lần bất cứ khi nào một câu lệnh thuộc loại thích hợp được thực thi, bất kể có bao nhiêu hàng - không, một hoặc nhiều - nó thực sự ảnh hưởng.
- Trong trình kích hoạt cấp câu lệnh, chúng ta không thể tham chiếu trực tiếp đến các bộ giá trị cũ và mới (như dòng 4,5), nhưng sử dụng các khai báo như**BẢNG CŨ NHƯ Cũ**và **BẢNG MỚI AS NewStuff**.

Các tùy chọn cho thiết kế kích hoạt

- Ví dụ

1) TẠO TRIGGER AvgNetWorthTrigger

2) SAU KHI CẬP NHẬT netWorth TRÊN MovieExec

3) TÀI LIỆU THAM KHẢO

4) BẢNG CŨ NHƯ Cũ,

5) BẢNG MỚI AS NewStuff

6) CHO TỪNG BÁO CÁO

7) KHI ((CHỌN AVG (netWorth) TỪ MovieExec) <500000)

8) BẮT ĐẦU

9) XÓA KHỎI MovieExec

10) WHERE (tên, địa chỉ, cert #, netWorth) TRONG NewStuff;

11) CHÈN VÀO MovieExec

12) (CHỌN * TỪ OldStuff);

13) HẾT;

Tạo trình kích hoạt trong máy chủ SQL

TẠO TRIGGER Trigger_name ON {bảng | Quang cảnh}
{CHO | SAU | INSTEAD OF} {[INSERT] [,] [UPDATE] [,] [DELETE]} NHƯ
{sql_statement [;] [,...n] [;]>}

- Tranh luận

- **CHO | SAU**:AFTER chỉ định rằng trình kích hoạt DML chỉ được kích hoạt khi tất cả các hoạt động được chỉ định trong câu lệnh SQL kích hoạt đã thực thi thành công. Tất cả các hành động phân tầng tham chiếu và kiểm tra ràng buộc cũng phải thành công trước khi trình kích hoạt này kích hoạt. AFTER là mặc định khi FOR là từ khóa duy nhất được chỉ định. Các trình kích hoạt SAU KHI không thể được xác định trên các chế độ xem.
- INSTEAD OF: Chỉ định rằng trình kích hoạt DML được thực thi*thay vì*do đó, câu lệnh SQL kích hoạt ghi đè các hành động của câu lệnh kích hoạt. INSTEAD OF không thể được chỉ định cho các trình kích hoạt DDL hoặc đăng nhập. Tối đa, một kích hoạt INSTEAD OF cho mỗi câu lệnh INSERT, UPDATE hoặc DELETE có thể được xác định trên một bảng hoặc dạng xem. INSTEAD OF trình kích hoạt không được phép trên các dạng xem có thể cập nhật sử dụng VỚI CHỌN KIỂM TRA.

- Trình kích hoạt DML thường được sử dụng để thực thi các quy tắc kinh doanh và tính toàn vẹn của dữ liệu.

Tạo trình kích hoạt trong máy chủ SQL

- ví dụ 1

Trình kích hoạt DML sau sẽ in một thông báo cho máy khách khi bất kỳ ai cố gắng thêm hoặc thay đổi dữ liệu trong Customertable trong cơ sở dữ liệu AdventureWorks2012.

NẾU OBJECT_ID ('Sales.reminder1', 'TR') KHÔNG ĐẦY ĐỦ
DROP TRIGGER Sales.reminder1;
ĐI
TẠO nhắc nhở TRIGGER1 VỀ Bán hàng. Khách hàng
SAU KHI CHÈN, CẬP NHẬT
AS RAISERROR ('Thông báo Quan hệ Khách hàng', 16, 10); ĐI

Tạo trình kích hoạt trong máy chủ SQL

- Ví dụ 2

Ví dụ sau đây sẽ gửi một thông điệp e-mail đến một người được chỉ định (MaryM) khi bảng Khách hàng thay đổi.

```
NẾU OBJECT_ID ('Sales.reminder2', 'TR') KHÔNG ĐẦY ĐỦ
DROP TRIGGER Sales.reminder2;
ĐI
TẠO nhắc nhở TRIGGER2 VỀ Bán hàng. Khách hàng
SAU KHI CHÈN, CẬP NHẬT, XÓA
AS EXEC msdb.dbo.sp_send_dbmail @profile_name =
  'AdventureWorks2012 Admin', @recipient = '
  danw@Adventure-Works.com ', @body = 'Đừng quên in
  một báo cáo cho
                    lực lượng bán hàng.',
  @subject = 'Nhắc nhở';
ĐI
```

Tạo trình kích hoạt trong máy chủ SQL

- Ví dụ 3

TẠO TRIGGER NetWorthTrigger trên MovieExec SAU KHI

CẬP NHẬT như

khai báo @new int, @old int, @ssn int

select @ new = ne.netWorth, @ old = ol.netWorth, @ ssn = ol.name từ

ne.netWorth đã chèn, xóa ol

nơi ne.name = ol.name

if (@old> @new)

Bắt đầu

CẬP NHẬT MovieExec

SET netWorth = @old

WHERE name = @ssn

chấm dứt

ĐI

Lượt xem

- MỘT *Quang cảnh* là một bảng ảo có nội dung (cột và hàng) được xác định bởi một truy vấn lấy dữ liệu trong một hoặc nhiều bảng (được gọi là *bảng cơ sở*) hoặc các dạng xem khác trong cơ sở dữ liệu.
- một chế độ xem có thể được sử dụng cho các mục đích sau:

1. Để tập trung, đơn giản hóa và tùy chỉnh nhận thức của mỗi người dùng về cơ sở dữ liệu.
2. Là một cơ chế bảo mật bằng cách cho phép người dùng truy cập dữ liệu thông qua dạng xem, mà không cấp cho người dùng quyền truy cập trực tiếp vào các bảng cơ sở bên dưới.
3. Cung cấp giao diện tương thích ngược để mô phỏng một bảng có giản đồ đã thay đổi

Lượt xem

- Khai báo bởi:

  TẠO VIEW view_name AS
  CHỌN (các) tên cột
  TỪ tên_bảng
  Điều kiện WHERE;

- Ghé qua:

  DROP XEM <tên>

- Sửa đổi bởi:

  ALTER VIEW view_name
  NHƯ CHỌN (các) tên cột
  TỪ tên_bảng
  Điều kiện WHERE;

Ví dụ chạy của chúng tôi

Các loại bia (BiaTên , manf) Bars (Quầy barTên , addr, license) Người uống rượu (DrTên , addr, phone) Lượt thích (Drname ,bia) Bán (tên thanh ,bia , giá cả) Tần suất (Drname ,tên thanh)

- Gạch chân =*Chìa khóa*(các bộ giá trị không thể có cùng giá trị trong tất cả các thuộc tính chính).

Ví dụ: Xem Định nghĩa

- CanDrink (người uống rượu bia)là một chế độ xem “chứa” các cặp đồ uống-bia để người uống rượu bia thường xuyên lui tới ít nhất một quán bar phục vụ bia:

TẠO CHẾ ĐỘ XEM Có thể uống NHƯ

CHỌN người uống, bia

TỪ Tần suất, Bán

WHERE Frequents.bar = Sells.bar;

Ví dụ: Truy cập một Chế độ xem

- Truy vấn một dạng xem như thể nó là một bảng cơ sở.
  - Ngoài ra: một khả năng hạn chế để sửa đổi các chế độ xem nếu nó có ý nghĩa như một sửa đổi của một bảng cơ sở bên dưới.
- Truy vấn mẫu:

```
CHỌN bia TỪ CanDrink WHERE
người uống = 'Sally';
```

Lượt xem

- Vấn đề: mỗi khi một bảng cơ sở thay đổi, chế độ xem có thể thay đổi.
  - **Ghi chú:**Chế độ xem luôn hiển thị dữ liệu cập nhật! Công cụ cơ sở dữ liệu tạo lại dữ liệu, sử dụng câu lệnh SQL của chế độ xem, mỗi khi người dùng truy vấn một chế độ xem.

Chế độ xem không thể cập nhật

- Các chế độ xem được xác định bằng cách sử dụng các nhóm và các hàm tổng hợp không thể cập nhật được
- Các dạng xem được xác định trên nhiều bảng bằng cách sử dụng phép nối thường không thể cập nhật được

Bán tại. Dept (đã, ngân sách, quản lý)

Hoạt động (eid, did, pct_time)

```
tạo chế độ xemccbằng
  Lựa chọnquản lý,
        tối đa(ngân sách)bằng
  ln từDept;
```

```
tạo chế độ xemddnhư Lựa
  chọnd.đã làm,w.eid,
                w.pct_time
  từPhòng d,Hoạt động w ở
  đâud.đã làm=w.đã làm;
```

Ví dụ

Phim (tiêu đề, năm, thời lượng, thể loại, studioName,
producerC #)

Một chế độ xem được định nghĩa là:

TẠO XEM ParamountMovies AS
CHỌN tiêu đề,
năm TỪ Phim
WHERE studioName = Điều tối quan trọng '; Giả sử
chúng ta chèn vào chế độ xem ParamountMovies
CHÈN VÀO ParamountMovies
GIÁ TRỊ ('StarTrek', 1979);

Sao vậy ???

Kích hoạt trên Lượt xem

- Trình kích hoạt INSTEAD OF cho phép chúng tôi giải thích các sửa đổi chế độ xem theo cách có ý nghĩa.

Ví dụ: Dept (đã, ngân sách, quản lý)

Hoạt động (eid, did, pct_time)

```
tạo chế độ xemddnhư Lựa
  chọnd.đã làm,w.eid,
                w.pct_time
  từPhòng d,Hoạt động w ở
  đâud.đã làm=w.đã làm;
```

Chúng tôi không thể chèn trực tiếp vào dd:

CHÈN VÀO GIÁ TRỊ dd (9, 2, 16)

Diễn giải phần chèn chế độ xem

- Chúng tôi không thể chèn vào xem dd.
- Nhưng chúng ta có thể sử dụng trình kích hoạt INSTEAD OF để biến(did, eid, pct_time)nhân ba thành hai phần chèn của các cặp dự kiến, một cho mỗi Bộ phận và Công trình.
  - Dept.budget và Dept.manageid sẽ phải là NULL.

Kích hoạt trên lượt xem

TẠO TRIGGER InsertToView TRÊN dd

INSTEAD OF INSERT

BẰNG

DECLARE @sdid int, @seid int, @spct_time int

SELECT @sdid = insert.did, @seid = insert.eid,

@spct_time = insert.pct_time FROM

đã được chèn

BẮT ĐẦU

CHÈN VÀO Bộ phận (đã làm) GIÁ TRỊ (@sdid)

CHÈN VÀO GIÁ TRỊ CÔNG TRÌNH (@seid, @sdid, @spct_time)

HẾT

Trình kích hoạt trong SQL-Server

TẠO TRIGGER GiveRaise1 ON Emp SAU KHI cập nhật AS

Khai báo @new numeric (18,0), @old numeric (18,0), @eid smallint

SELECT @ new = ne.Salary, @ old = ol.salary, @eid = ne.eid

TỪ Đã chèn ne, Đã xóa ol

trong đó ne.eid = ol.eid

If @new> @old BEGIN

cập nhật Emp

Đặt Emp.salary = @new

nơi Emp.salary <@new và Emp.eid in

(Chọn D.manageid

Từ Emp E, Works W, Dept D

trong đó E.eid = @eid

và E.eid = W.eid và W.did = D.did); CHẤM DỨT

Hỏi & Đáp