اصول طراحی پایگاه داده ها

۷- محدودیت ها در پایگاه داده

رضا سعیدی

برگرفته از اسلایدهای خانم بهکمال

محدودیت ها در پایگاه داده

Constraints & Triggers

- ***constraint*** **محدودیتی بین المان های داده است که DBMS ملزم به رعایت آنها می شود.**

- ***trigger*** **تنها زمانی اجرا می شوند که شرایط یا اتفاق خاصی رخ دهد**

  - گاهی اوقات اجرا کردن قیود نسبت به محدودیت های پیچیده آسان تر می باشد.

انواع محدودیت ها

1. کلیدها (یک صفت یا چند صفت)
2. کلیدهای خارجی، یا پیوستگی های ارجاعی (referential-integrity)
3. محدودیت روی مقادیر (Value-based constraint)
   - محدودیت های کنترل مقادیر یک صفت خاص
4. محدودیت روی سطرهای جدول (Tuple-based constraint)
   - محدودیت برای یک سطر (چک کردن مقادیر چند صفت)
5. Assertion: هرعبارت با مقدار بولین در SQL

۱. کلید اصلی

- کلید تک صفتی

بعد از تعریف نوع صفت، عبارت PRIMARY KEY یا UNIQUE را قرار می دهیم.

```
CREATE TABLE Beers (
        name    CHAR(20) UNIQUE,
        manf    CHAR(20)
);
```

۱. کلید اصلی ...

- کلید چندصفتی

در رابطه Sells ترکیب صفات bar و beer کلید محسوب می شوند:

```
CREATE TABLE Sells (
        bar         CHAR(20),
        beer        CHAR(20),
        price       REAL,
        PRIMARY KEY (bar, beer)
);
```

۲. کلید خارجی

- **مقادیری که برای صفت یک جدول دیده می شوند، باید در صفت کلید خارجی متناظرش در جدول دیگر هم وجود داشته باشد.**

- بعنوان مثال انتظار داریم که مقدار beer در رابطه Sells(bar, beer, price) در صفت Beers.name هم وجود داشته باشد.

۲. کلید خارجی ...

- برای تعریف کلید خارجی، کلمه REFERENCES را به یکی از دو صورت زیر استفاده می کنیم:
  - **بعد از یک صفت (برای کلیدهای تک صفتی)**
  - **بعنوان یک المان در شمای رابطه**

FOREIGN KEY (<list of attributes>)
REFERENCES <relation> (<attributes>)

- صفتی که به آن ارجاع می شود باید PRIMARY KEY یا UNIQUE باشد.

۲. کلید خارجی ...

مثال تعریف کلید خارجی:

```
CREATE TABLE Beers (
      name     CHAR(20) PRIMARY KEY,
      manf     CHAR(20) );
```

- تعریف کلید خارجی بعد از یک صفت

```
CREATE TABLE Sells (
      bar    CHAR(20),
      beer  CHAR(20) REFERENCES Beers(name),
      price   REAL );
```

- تعریف کلید خارجی بعنوان یک المان در شمای رابطه

```
CREATE TABLE Sells (
      bar     CHAR(20),
      beer    CHAR(20),
      price   REAL,
      FOREIGN KEY(beer) REFERENCES      Beers(name) );
```

۲. کلید خارجی ...

- **اگر محدودیت کلید خارجی از رابطه R به رابطه S باشد، دو اتفاق زیر ممکن است بیفتد:**

1. درج یا به روز رسانی در رابطه R باعث ایجاد مقادیری می شود که در S وجود ندارد.

2. حذف یا به روز رسانی در رابطه S باعث ایجاد مشکل در سطرهای رابطه R می شود.

۲. کلید خارجی ...

- مثال

Beers (name, manf)
Sells (bar, beer, price)
FOREIGN KEY Sells.beer REFERENCES Beers.name

- در هنگام درج یا بروز رسانی جدول **Sells** اگر مقادیر جدید **beer** در صفت **name** جدول **Beers** وجود نداشته باشد، درج یا بروز رسانی پذیرفته نمی شود.

- در هنگام حذف یا بروز رسانی جدول **Beers** اگر مقداری از صفت **name** که در بعضی از سطرهای **Sells** وجود دارد حذف شود، سه روش وجود دارد:

۲. کلید خارجی ...

1. **Default: عدم پذیرش تغییر**

2. **Cascade: تمام تغییرات لازم در Sells اعمال شود.**
   - حذف beer: تمام سطرهای Sells که آن beer را دارند حذف میشوند.
   - به روز رسانی beer: تمام سطرهای Sells که آن beer را دارند تغییر داده میشوند.

3. **Set NULL: مقدار beer در Sells به NULL تغییر کند.**

۲. کلید خارجی ...

- **مثال Cascade:**
- **دررابطه Beers سطری که ‘Bud’ = name حذف می شود**
  - بنابراین در رابطه Sells تمام سطرهایی که ’Bud’ = beer حذف میشوند.
- **دررابطه Beers سطری که ‘Bud’ = name از ’Bud’ به ’Budweiser’ تغییر می کند.**
  - بنابراین در رابطه Sells تمام سطرهایی که ’Bud’ = beer به ’Budweiser’ = beer تبدیل میشوند.

۲. کلید خارجی ...

- **مثال Set NULL :**
- **دررابطه Beers سطری که ‘Bud’ = name حذف می شود**
  - بنابراین در رابطه Sells تمام سطرهایی که ’Bud’ = beer به beer = NULL تبدیل می شوند.
- **دررابطه Beers سطری که ‘Bud’ = name از ’Bud’ به ’Budweiser’ تغییر می کند.**
  - بنابراین در رابطه Sells تمام سطرهایی که ’Bud’ = beer به beer = NULL تبدیل می شوند.

۲. کلید خارجی ...

- وقتی کلید خارجی را تعریف می کنیم باید یکی از روش های CASCADE یا SET NULL را برای حذف و به روز رسانی انتخاب کنیم.

FOREIGN KEY (<list of attributes>)
REFERENCES <relation> (<attributes>)
ON [UPDATE, DELETE][SET NULL CASCADE]

- در غیر اینصورت، حالت پیش فرض (default) استفاده می شود و تغییر پذیرفته نمی شود.

۲. کلید خارجی ...

مثال انتخاب روش حذف و بروز رسانی:

```
CREATE TABLE Sells (
    bar  CHAR(20),
    beerCHAR(20),
    price        REAL,
    FOREIGN KEY(beer)
        REFERENCES Beers(name)
        ON DELETE SET NULL
        ON UPDATE CASCADE
);
```

۲. محدودیت مقادیر صفات

- محدودیت می تواند روی مقدار یک صفت خاص اعمال می شوند.
- برای اعمال محدودیت، در هنگام تعریف صفت، عبارت زیر اضافه می شود:

- CHECK(<condition>)

- در عبارت condition می توان از نام یک صفت استفاده کرد، ولی برای اعمال محدودیت روی یک رابطه یا سایر صفات باید از sub query استفاده کرد.

۲. محدودیت مقادیر صفات ...

مثال

```
CREATE TABLE Sells (
    bar         CHAR(20),
    beer        CHAR(20)   CHECK ( beer IN
                (SELECT name FROM Beers)),
    price       REAL CHECK ( price <= 5.00 )
);
```

۲. محدودیت مقادیر صفات ...

- **زمان چک کردن: چک کردن مقدار صفات فقط در زمان درج یا بروز رسانی آن صفت انجام می شود.**
  - **CHECK (price <= 5.00)**

    برای هر قیمت جدید بررسی می شود و در صورتی که قیمت آن از 5 بیشتر باشد، تغییر پذیرفته نمی شود.

  - **CHECK (beer IN (SELECT name FROM Beers))**
    - در هنگام حذف یک beer از رابطه Beers چک نمی شود (برخلاف کلیدهای خارجی)

۴. محدودیت روی سطرها

- **عبارت (<condition>) CHECK می تواند به عنوان یک المان در شمای رابطه اضافه شود.**

- **برای هرکدام از صفات رابطه می توان شرط گذاشت**
  - ولی برای رابطه ها یا خصیصه های دیگر باید از subquery استفاده کرد.

- **فقط در هنگام درج یا به روز رسانی سطر چک می شود.**

۴. محدودیت روی سطرها ...

- تنها Joe’s Bar می تواند beer با قیمت بیشتر از $5 بفروشد.

```
CREATE TABLE Sells (
        bar             CHAR(20),
        beer            CHAR(20),
        price           REAL,
        CHECK (bar = ’Joe’’s Bar’ OR
                                price <= 5.00)
);
```

۵- Assertion

- المان های شمای پایگاه داده هستند، مثل رابطه ها و دیدها.

- به صورت زیر تعریف می شود:

```
CREATE ASSERTION <name>
        CHECK (<condition>);
```

- شرط روی همه رابطه ها و خصیصه ها در شمای پایگاه داده اعمال می شود.

۵- Assertion ...

- مثال. در رابطه Sells(bar, beer, price) میانگین قیمت های bar نباید از $5 بیشتر باشد.

```
CREATE ASSERTION NoRipoffBars CHECK (
   NOT EXISTS (
        SELECT bar FROM Sells
        GROUP BY bar
        HAVING AVG(price) > 5.00
));
```

Bars with an average price above $5

۵- Assertion ...

- مثال. در دو رابطه زیر:

- Drinkers(name, addr, phone)
- Bars(name, addr, license)

تعداد bars نباید از تعداد drinkers بیشتر باشد.

```
CREATE ASSERTION FewBar CHECK (
(SELECT COUNT(*) FROM Bars) <=
(SELECT COUNT(*) FROM Drinkers)
);
```

۵- Assertion ...

زمان چک کردن:

- **بطورکلی، بعد از هر تغییر در جداول پایگاه داده، assertion باید چک شود.**

- **تنها بعضی از تغییرها، می تواند در assertion تاثیر بگذارد.**
  - مثال: هرتغییری در رابطه Beers یا درج هر سطر در رابطه Drinkers، تاثیری روی FewBar نمی گذارد.

Trigger

- Assertions ابزار قدرتمندی است، اما گاهی اوقات برای DBMS تشخیص اینکه چه زمانی باید چک کردن را انجام بدهد کار راحتی نیست.
- اگرچه چک کردن صفات و سطرها زمان معینی دارد، اما خیلی قدرتمند نیست.
- Triggers به کاربر اجازه می دهند که خود برای زمان چک کردن شرایط خاص تصمیم بگیرد.

Trigger …

- نام دیگر "trigger"، قانون *ECA* (*event-condition-action*) *می باشد.*
- **Event**: هر تغییری بر روی پایگاه داده مثلا: درج در رابطه **Sells**
- **Condition**: هر دستور **SQL** که یک مقدار بولین برمی گرداند.
- **Action**: هر دستور **SQL**

Trigger …

- مثال.

Beers (name, manf)
Sells (bar, beer, price)
FOREIGN KEY Sells.beer REFERENCES Beers.name

- استفاده از محدودیت کلید خارجی: هنگام درج یک سطر در رابطه Sells که مقدار beer درجدول Beers تعریف نشده است، با عدم پذیرش درج مواجه می شویم.

- استفاده از trigger: در این حالت، هنگام درج سطر درSells، مقدار beer نامعلوم به رابطه Beers اضافه می شود و سایر صفات آن (یعنی manf) مقدار NULL می گیرند.

Trigger …

```
CREATE  TRIGGER  BeerTrig
    AFTER INSERT ON Sells
    REFERENCING NEW ROW AS NewTuple
    FOR EACH ROW
    WHEN (NewTuple.beer NOT IN
        (SELECT name FROM Beers))
    INSERT INTO Beers(name)
        VALUES(NewTuple.beer);
```

The event

The condition

The action

تعریف یا اصلاح Trigger

- تعریف trigger جدید:

CREATE TRIGGER <name>

- برای وقتی که یک trigger وجود دارد و می خواهیم آن را تغییر دهیم.

REPLACE TRIGGER <name>

The Event

- به جای **AFTER** می توان **BEFORE** را بکار برد.
- به جای **INSERT** می توان از **DELETE** یا **UPDATE** استفاده کرد.
- به جای **UPDATE** می توان از **UPDATE . . . ON** روی یک صفت خاص استفاده کرد.

FOR EACH ROW

Trigger ها در دو سطح اجرا می شوند:

- **Row-level triggers**: برای هر سطر که تغییر پیدا کند یکبار اجرا می شوند.
- **Statement-level triggers**: برای یک دستور SQL یکبار اجرا می شوند، بدون توجه به اینکه در این دستور چند سطر تغییر یافته باشد.
- عبارت FOR EACH ROW نشان دهنده سطح اول است و چنانچه ذکر نشود، Triggerدر سطح دستور عمل می کند

REFERENCING

- عبارت INSERT نشان می دهد که یک سطر جدید (row-level) یا یک جدول جدید (statement-level) درج خواهد شد.
  - یک جدول، یک مجموعه از سطرهای درج شده است.
- عبارت DELETE نشان می دهد که یک سطر یا جدول قدیمی حذف خواهد شد.
- عبارت UPDATE نشان می دهد که یک سطر یا جدول قدیمی به سطر یا جدول جدید تبدیل خواهد شد.
- حالت های بالا به صورت زیر نشان داده می شوند:

[NEW OLD][TUPLE TABLE] AS <name>

The Condition

- **هر دستور SQL که یک مقدار بولین برمیگرداند.**
- **بر اساس اینکه از BEFORE یا AFTER استفاده شده باشد، روی پایگاه داده محاسبه می شود**
  - ولی همیشه قبل از اینکه تغییرات اتفاق بیفتند، اعمال می شود.

The Action

- ممکن است بیش از یک دستور SQL در action استفاده شود.
  - در این حالت از عبارت BEGIN . . . END استفاده می شود.