Vincoli

Chiavi esterne

Vincoli locali e globali

Triggers

Leggere Cap 2 Riguzzi et al. Sistemi Informativi

Lucidi derivati da quelli di Jeffrey D. Ullman

Vincoli e Triggers

- Un *vincolo* e' una relazione tra dati che il DBMS deve assicurare.
  - Esempio: vincoli di chiave.
- *I triggers* sono eseguiti solo quando una condizione specifica accade, ad es., l'inserimento di una tupla.
  - Piu' facile da implementare di molti vincoli.

Tipo di vincoli

- Chiavi.
- Chiavi esterne, o integrita' referenziale.
- Vincoli basati su un solo attributo.
  - Vincola i valori di uno specifico attributo.
- Vincoli basati sulla tupla.
  - Relazioni tra attributi.
- Asserzioni: qualsiasi espressione booleana SQL.

Chiavi esterne

- Si consideri la relazione Sells(bar, beer, price).
- Ci possiamo aspettare che un valore beer sia una birra reale, ovvero qualcosa che appare in Beers.name.
- Un vincolo che richiede che una birra in Sells sia una birra in Beers e' chiamato un vincolo di *chiave esterna* (foreign-key constraint).

Esprimere le chiavi esterne

- Si usa la parola chiave REFERENCES:
  1. Nella dichiarazione di un attributo, quando un solo attributo e' coinvolto, oppure
  2. Come un elemento dello schema, come in:

  FOREIGN KEY (<lista di attributi>)

  REFERENCES <relazione> (<attributi>)
- Gli attributi referenziati devono essere dichiarati PRIMARY KEY o UNIQUE.

Esempio: con attributo

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

Esempio: come elemento

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

Applicare i vincoli di chiave esterna

- Se c'e' un vincolo di chiave esterna da attributi della relazione *R* alla chiave primaria della relazione *S*, sono possibili due violazioni:
  1. Un inserimento o una modifica in *R* introducono valori non presenti in *S*.
  2. Una cancellazione o una modifica in *S* fa si' che alcune tuple di *R* rimangano "a penzoloni"

Azioni intraprese - 1

- Si supponga *R* = Sells, *S* = Beers.
- Un inserimento o una modifica di Sells che introduce un birra non esistente in Beers deve essere rifiutato.
- Una cancellazione o una modifica di Beers che rimuove un valore di birra presente in alcune tuple di Sells puo' essere gestito in quattro modi.

Azioni intraprese -- 2

- I quattro modi possibili per gestire le birre che improvvisamente smettono di esistere sono:
  1. NO ACTION : rifiuta la modifica.
  2. CASCADE : effettua gli stessi cambiamenti in Sells.
     - Birra cancellata: cancella la tupla di Sells.
     - Birra modificata: modifica il valore in Sells.
  3. SET NULL : modifica la birra in NULL.
  4. SET DEFAULT : assegna alla birra il suo valore di default

Esempio: CASCADE

- Si supponga di cancellare la tupla per la Bud da Beers.
  - Allora si cancellano tutte le tuple di Sells che hanno beer = ’Bud’.
- Si supponga di aggiornare la tupla per la Bud cambiando ’Bud’ in ’Budweiser’.
  - Allora si cambiano tutte le tuple di Sells con beer = ’Bud’ in modo che beer = ’Budweiser’.

Esempio: SET NULL

- Si supponga di cancellare la tupla per la Bud da Beers.
  - Si cambiano tutte le tuple di Sells che hanno beer = ’Bud’ in beer = NULL.
- Si supponga di aggiornare la tuple per la Bud cambiando ’Bud’ in ’Budweiser’.
  - Stesso cambiamento.

Esempio: SET DEFAULT

- Si supponga di cancellare la tupla per la Bud da Beers.
  - Si assegna a tutte le tuple di Sells che hanno beer = 'Bud' il valore di default per beer
- Si supponga di aggiornare le tuple per la Bud cambiando 'Bud' in 'Budweiser'.
  - Stesso cambiamento.
- La colonna beer deve avere un vincolo di default. Se non ce l'ha ed è nullable, allora NULL diventa il valore della colonna. Altrimenti errore

Scegliere una politica

- Quando dichiariamo una chiave esterna, possiamo scegliere la politica SET NULL, CASCADE o SET DEFAULT indipendentemente per le cancellazioni e le modifiche.
- Far seguire la dichiarazione di chiave esterna da:

ON [UPDATE | DELETE][SET NULL | CASCADE | SET DEFAULT]

- Due clausole di questo tipo possono essere usate.
- Altrimenti, il default (impedisci (NO ACTION)) e' usato.

Esempio

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

Check basati sugli attributi

- Mettono un vincolo sul valore di un particolare attributo.
- CHECK(<condizione>) deve essere aggiunto alla dichiarazione dell'attributo.
- La condizione puo' essere una qualunque condizione che puo' apparire in una clausola WHERE
- La condizione puo' usare il nome dell'attributo ma ogni altra relazione o nome di attributo devono apparire in una sottoquery

Esempio

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

Temporizzazione delle verifiche

- Un check basato sugli attributi e' verificato solo quando un valore per quell'attributo viene inserito o modificato.
  - Esempio: CHECK (price <= 5.00) verifica ogni nuovo prezzo e lo rifuta se e' piu di $5.
  - Esempio: CHECK (beer IN (SELECT name FROM Beers)) non e' verificato se una birra viene cancellata da Beers (a differenza delle chiavi esterne).

Check basati sulle tuple

- CHECK (<condizione>) puo' essere aggiunto come un altro elemento di una definizione di schema.
- La condizione puo' essere una qualunque condizione che puo' apparire in una clausola WHERE
- La condizione puo' fare riferimento a ogni attributo della relazione ma ogni altro attributo o relazione richiede una sottoquery.
- Verificato solo agli inserimenti e alle modifiche.

Esempio: check

- Solo Joe's Bar puo' vendere birra a piu' di 5$:

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

Soluzione: triggers

- I checks basati su attributi e su tuple hanno capacita' limitate.
- Un trigger consente all'utente di specificare quando deve essere fatta la verifica.
- Un trigger ha una condizione generale e puo' inoltre effettuare una qualunque sequenza di modifiche SQL al database.

Regole Evento-Condizione-Azione

- Un altro nome per "trigger" e' *regola ECA*, o regola evento-condizione-azione.
- *Evento* : tipicamente un tipo di modifica di database, ad es. "insert on Sells."
- *Condizione* : una qualunque espressione SQL con valore booleano.
- *Azione* : una qualunque istruzione SQL.