SQL

Gugolya László

Aktív elemek SQL-ben

Aktív elem: olyan programrész, amely bizonyos szituációban automatikusan végrehajtódik.

Megszorítás: olyan aktív elem, amely bizonyos feltételek ellenőrzését jelenti.

Attribútumértékre vonatkozó megszorítások

CREATE TABLE-ben adhatók meg:

- **PRIMARY KEY**: *elsődleges kulcs*
- **UNIQUE**: *kulcs*
- **REFERENCES**: *külső kulcs*
- **NOT NULL**: *kötelező kitöltés*
- **CHECK (feltétel)**: *általános feltétel (logikai kifejezés)*

Példa:

```
CREATE TABLE Osztály
 ( osztálykód  CHAR(3) PRIMARY KEY,
   osztálynév  CHAR(20) UNIQUE,
   vezAdószám  DECIMAL(10)
 );
CREATE TABLE Dolgozó
( adószám DECIMAL(10) PRIMARY KEY,
  név     CHAR(30) NOT NULL,
  nem     CHAR(1)
    CHECK (nem IN ('F','N')),
  lakcím  CHAR(40),
  osztkód CHAR(3)
    REFERENCES Osztály(osztálykód)
);
```

Példa: *külső kulcs feltétel részleges ellenőrzése*

```
CREATE TABLE Osztály
 ( osztálykód  CHAR(3) PRIMARY KEY,
   osztálynév  CHAR(20),
   vezAdószám  DECIMAL(10)
 );
CREATE TABLE Dolgozó
 ( adószám DECIMAL(10) PRIMARY KEY,
   név     CHAR(30) NOT NULL,
   lakcím  CHAR(40),
   osztkód CHAR(3)
     CHECK (osztkód IN
              (SELECT osztálykód
               FROM Osztály))
 );
```

Értéktartományok kezelése

Létrehozás:

CREATE DOMAIN név típus
[DEFAULT érték] [CHECK (feltétel)];

Módosítás:

ALTER DOMAIN név ...;

Törlés:

DROP DOMAIN név;

Példa:

```
CREATE DOMAIN NemÉrték CHAR(1)
  CHECK (VALUE IN ('F','N'));

CREATE TABLE Dolgozó
   ( adószám DECIMAL(10) PRIMARY KEY,
     név      CHAR(30),
     nem      NemÉrték,
     lakcím   CHAR(40)
   );
```

Táblára vonatkozó megszorítások

A teljes táblára vonatkoznak.

CREATE TABLE végére, a táblafeltételekhez írandók.

- **PRIMARY KEY**: *elsődleges kulcs*
- **UNIQUE**: *kulcs*
- **FOREIGN KEY**: *külső kulcs*
- **CHECK (feltétel)**: *általános feltétel (logikai kifejezés)*

Példa:

```
CREATE DOMAIN NapÉrték CHAR(2)
  CHECK (VALUE IN ('H','K','Sz','Cs','P'));
CREATE TABLE Tanóra
  ( órakód  DECIMAL(8) PRIMARY KEY,
    óranév  CHAR(30),
    nap     NapÉrték,
    tól     DECIMAL(2),
    ig      DECIMAL(2),
    CHECK (tól>7 AND ig<21 AND tól<ig)
  );
```

Általános megszorítások

Több táblára vonatkoznak.

Létrehozás:

CREATE ASSERTION név CHECK (feltétel);

Törlés:

DROP ASSERTION név;

Példa:

Dolgozó (adószám, név, lakcím, fizetés, *osztálykód*)
Osztály (osztálykód, osztálynév, *vezAdószám*)

```
CREATE ASSERTION VezetőFizetés
  CHECK (NOT EXISTS
    (SELECT * FROM Dolgozó,Osztály
      WHERE Dolgozó.Adószám =
            Osztály.vezAdószám
        AND fizetés < 100000
    ));
```

Megszorítások módosítása

Megszorítás elnevezése: **CONSTRAINT név**
Példa:

```
CREATE TABLE Dolgozó
( adószám DECIMAL(10) PRIMARY KEY,
  név CHAR(30) CONSTRAINT NévKulcs UNIQUE,
  lakcím CHAR(40),
);
```

Kulcsfeltétel elvetése:

```
ALTER TABLE Dolgozó
  DROP CONSTRAINT NévKulcs;
```

A kulcsfeltétel újra érvényesítése táblafeltételként:

```
ALTER TABLE Dolgozó
  ADD CONSTRAINT NévKulcs UNIQUE(név);
```

Triggerek

Trigger: aktualizálási művelet esetén végrehajtandó programrészletet definiál. Adatbázis-objektumként tárolódik.

CREATE TRIGGER név
{ BEFORE | AFTER | INSTEAD OF }
{ DELETE | INSERT | UPDATE [OF oszlopok] }
ON tábla
[REFERENCING [OLD AS régi] [NEW AS új]]
[FOR EACH ROW]
[WHEN (feltétel)]
programblokk;

Példa. *Fizetésváltozások naplózása*
Dolgozó (adószám, név, lakcím, fizetés)
FizetésNapló (dátum, *adószám*, régifiz, újfiz)

```
CREATE TRIGGER fiz_napló
   AFTER UPDATE OF fizetés ON Dolgozó
   REFERENCING OLD AS régi NEW AS új
   FOR EACH ROW
INSERT INTO FizetésNapló
   VALUES (SYSDATE, régi.adószám,
           régi.fizetés, új.fizetés);
```

Megjegyzés: Sok trigger áttekinthetetlenné teheti a rendszer működését (A1 táblára írt T1 trigger módosítja az A2 táblát, amelyre írt T2 trigger módosítja...)