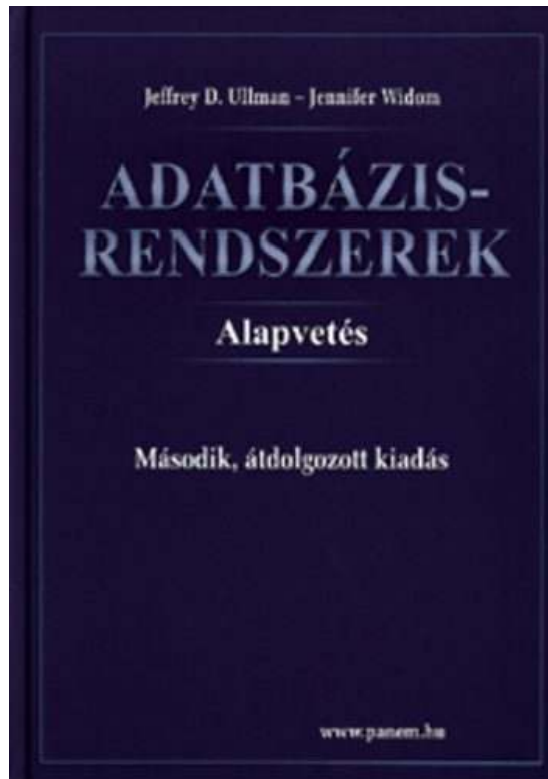


Megszorítások (Constraints)



Ullman-Widom: Adatbázisrendszerek
Alapvetés

Második, átdolgozott kiadás, Panem,
2009

7.1. Kulcsok és idegen kulcsok

7.2. Attribútumra vonatkozó
megszorítások

7.3. Megszorítások módosítása

Adatbázis relációsémák definiálása

- Tankönyv 2.3. fejezete
- Az SQL tartalmaz **adateleíró részt (DDL)** is, az adatbázis **objektumainak** a leírására és megváltoztatására.
Objektumok leíró parancsa a **CREATE** utasítás.
- Objektumok, például tábla, nézettábla, indextábla, stb.
- A relációt az SQL-ben táblának (TABLE) nevezik, az SQL alapvetően háromféle táblát kezel:
 - ❑ Alaptáblák (permanens) CREATE TABLE
 - ❑ Nézettáblák CREATE VIEW (ezt később nézzük)
 - ❑ Átmeneti munkatáblák
- **Alaptáblák** létrehozása: **CREATE TABLE** (köv.oldal)

Táblák létrehozása

- A legegyszerűbb formája:
CREATE TABLE táblanév (
attribútum deklarációk listája,
további kiegészítések);
- Az attribútum deklaráció legalapvetőbb elemei:
attribútumnév értéktípus [DEFAULT érték]
itt olyan értéktípus, amit az SQL konkrét megvalósítása támogat (gyakorlaton Oracle környezetben nézzük meg),
- **Típusok, például:** INTEGER, REAL, CHAR, VARCHAR, DATE, stb
- **DEAFULT kiegészítéssel** alapértelmezett értéket tudunk megadni.

Példa: sörivők adatbázis

Sörök(név, gyártó)

Bár(név, város, tulaj, engedély)

Ivó(név, város, tel)

Kedvel(név, sör)

Felhasználó(bár, sör, ár)

Látogat(név, bár)

- Az aláhúzás jelöli a **kulcsot** (a sorok a kulcs összes attribútumán nem vehetik fel ugyanazt az értékeket).
 - Ez a kulcs, külső kulcs és hivatkozási épség megszorításoknak lesz később kiváló példája.

Egyszerű példák táblák létrehozására

```
CREATE TABLE Bár (  
    név CHAR(20) ,  
    város VARCHAR2(40) ,  
    tulaj CHAR(30) ,  
    engedély DATE DEFAULT SYSDATE  
);
```

```
CREATE TABLE Felszolgál (  
    bár CHAR(20) ,  
    sör VARCHAR2(20) ,  
    ár NUMBER(10,2) DEFAULT 100  
);
```

Kulcsok megadása

- Egy attribútumot vagy attribútum listát kulcsként deklarálnak (PRIMARY KEY vagy UNIQUE).
- Mindkét formája a megszorításnak azt követeli meg, hogy relációnak ne legyen két olyan sora, melyek megegyeznek a kulcs attribútumokon.
- Kulcs esetén nincs értelme a DEFAULT értéknek.
- Kulcsok megadásának két változata van (köv.oldalak)
 - Egyszerű kulcs (egy attribútum) vagy
 - Összetett kulcs (attribútumok listája)

Egyszerű kulcs megadása

- Ha a kulcs egyetlen attribútum, akkor ez az attribútum deklarációban megadható, az attribútumnév és típus után a **PRIMARY KEY** vagy **UNIQUE** kulcsszót írjuk.
- Példa:

```
CREATE TABLE Sörök (  
    név          CHAR(20) UNIQUE,  
    gyártó       CHAR(20)  
);
```

Összetett kulcs megadása

- A CREATE TABLE utasításban az attribútum deklarációk után a **kiegészítő részben** meg lehet adni további tábla elemeket, például egyik eleme lehet **a kulcs deklaráció**.
- **Több attribútumú kulcsokat** csak ebben a formában deklarálhatunk (ugyanakkor az egyetlen attribútumból álló kulcsokat is megadhatjuk így módon).
- Példa:

```
CREATE TABLE Felszolgal (
    bár          CHAR(20) ,
    sör          VARCHAR2(20) ,
    ár           NUMBER(10,2) ,
    PRIMARY KEY (bár, sör) );
```

PRIMARY KEY vs. UNIQUE

- Egy relációhoz csak egyetlen **PRIMARY KEY** tartozhat, viszont **UNIQUE** több is lehet.
- A **PRIMARY KEY** egyetlen attribútuma sem kaphat **NULL** értéket. A **UNIQUE** megszorításnál viszont szerepelhetnek **NULL értékek vagyis hiányzó értékek** egy soron belül akár több is.

DDL – adatleíró részben módosítás

- Hogyan tudjuk a leíró részt módosítani?
 - CREATE – létrehozni
 - DROP – eldobni, a teljes leírást és mindazt, ami ehhez kapcsolódott hozzáférhetetlenné válik
 - ALTER – módosítani a leírást
- Ha ezt táblára használjuk
 - **DROP TABLE** táblanév;
 - **ALTER TABLE** táblanév
 - DROP attribútumnév - - oszlopot tudunk törölni
 - ADD attribútumnév értéktípus - - új oszlopot adni
 - kiegészítő részek például megszorítások
- Például mikor adhatunk meg UNIQUE feltételt?

Megszorítások és triggerek

- Tankönyv 7. fejezet
- Aktív elemek – olyan kifejezés vagy utasítás, amit egyszer eltároltunk az adatbázisban és az azt várjuk tőle, hogy a megfelelő pillanatban lefusson (pl. adatok helyességének ellenőrzése)
- A *megszorítás* adatelemek közötti kapcsolat, amelyet az AB rendszernek fent kell tartania.
 - *Példa*: kulcs megszorítások.
- *Triggerek* olyankor hajtódnak végre, amikor valamilyen megadott esemény történik, mint például sorok beszúrása egy táblába.

Megszorítások típusai

- **Kulcsok és idegen kulcsok,**
 - A hivatkozási épség fenntartása
 - Megszorítások ellenőrzésének késleltetése
- **Attribútumokra vonatkozó megszorítások**
 - NOT NULL feltételek
 - Egy attribútumra vonatkozó CHECK feltételek
- **Sorokra vonatkozó megszorítások**
 - Sorra vonatkozó CHECK feltételek
- **Önálló megszorítások (Assertions)**

Idegen kulcsok megadása

- Még egy kiegészítő lehetőség Mi köthet össze két táblát? **Idegen kulcs (foreign key) megadása**
- Az egyik tábla egyik oszlopában szereplő értékeknek szerepelnie kell egy másik tábla bizonyos attribútumának az értékei között.
- **A hivatkozott attribútumoknak** a másik táblában kulcsnak kell lennie! (PRIMARY KEY vagy UNIQUE)
- **Példa: Felszolgál(bár, sör, ár)** táblára megszorítás, hogy a sör oszlopában szereplő értékek szerepeljenek a **Sörök(név, gyártó)** táblában a név oszlop értékei között.

Idegen kulcsok megadása: attribútumként

REFERENCES kulcsszó használatának két lehetősége:
attribútumként vagy sémaelemként lehet megadni.

1.) Attribútumként (egy attribútumból álló kulcsra)

PÉLDA:

```
CREATE TABLE Sörök (  
    név CHAR(20) PRIMARY KEY,  
    gyártó CHAR(20) );
```

```
CREATE TABLE Felszolgál (  
    bár CHAR(20),  
    sör CHAR(20) REFERENCES Sör(név),  
    ár REAL );
```

Idegen kulcsok megadása: sémaelemként

2. Sémaelemként (egy vagy több attr.-ból álló kulcsra)

FOREIGN KEY (list of attributes)

REFERENCES relation (attributes)

PÉLDA: CREATE TABLE Sörök (
 név CHAR(20) PRIMARY KEY,
 gyártó CHAR(20));

CREATE TABLE Felszolgál (
 bár CHAR(20) ,
 sör CHAR(20) ,
 ár REAL,
 FOREIGN KEY (sör) REFERENCES Sörök (név));

Idegen kulcs megszorítások megőrzése

- Példa: $R = \text{Felszolgál}$, $S = \text{Sörök}$.
- Egy idegen kulcs megszorítás R relációról S relációra kétféleképpen sérülhet:
 1. Egy R -be történő beszúrásnál vagy R -ben történő módosításnál S -ben nem szereplő értéket adunk meg.
 2. Egy S -beli törlés vagy módosítás „lógó” sorokat eredményez R -ben.

Hogyan védekezzünk? --- (1)

- Példa: $R = \text{Felszolgál}$, $S = \text{Sörök}$.
- Nem engedjük, hogy **Felszolgál** táblába a **Sörök** táblában nem szereplő sört szűrjanak be vagy **Sörök** táblában nem szereplő sörre módosítsák (nincs válsztási lehetőségünk, a rendszer visszautasítja a megszorítást sértő utasítást)
- A **Sörök** táblából való törlés vagy módosítás, ami a **Felszolgál** tábla sorait is érintheti (mert sérül az idegen kulcs megszorítás) 3-féle módon kezelhető (lásd köv.oldal)

Hogyan védekezzünk? --- (2)

1. **Alapértelmezés (Default)** : a rendszer nem hajtja végre a törlést.
2. **Továbbgyűrűzés (Cascade)**: a Felszolgál tábla értékeit igazítjuk a változáshoz.
 - **Sör törlése**: töröljük a Felszolgál tábla megfelelő sorait.
 - **Sör módosítása**: a Felszolgál táblában is változik az érték.
3. **Set NULL**: a sör értékét állítsuk NULL-ra az érintett sorokban.

Példa: továbbgyűrűzés

- Töröljük a Bud sort a **Sörök** táblából:
 - az összes sort töröljük a **Felzolgál** táblából, ahol sör oszlop értéke 'Bud'.
- A 'Bud' nevet 'Budweiser'-re változtatjuk:
 - a **Felzolgál** tábla soraiban is végrehajtjuk ugyanezt a változtatást.

Példa: Set NULL

- A Bud sort töröljük a **Sörök** táblából:
 - a **Felhasználó** tábla **sör** = 'Bud' soraiban a Budot cseréljük NULL-ra.
- 'Bud'-ról 'Budweiser'-re módosítunk:
 - ugyanazt kell tennünk, mint törléskor.

A stratégia kiválasztása

- Ha egy idegen kulcsot deklarálnak megadhatjuk a SET NULL és a CASCADE stratégiát is beszúrásra és törlésre is egyaránt.
- Az idegen kulcs deklarálása után ezt kell írunk:
ON [UPDATE, DELETE][SET NULL CASCADE]
- Ha ezt nem adjuk meg, a default stratégia működik.

Példa: stratégia beállítása

```
CREATE TABLE Felszolgál (  
    bár      CHAR(20),  
    sör      CHAR(20),  
    ár       REAL,  
    FOREIGN KEY(sör)  
        REFERENCES Sörök(név)  
        ON DELETE SET NULL  
        ON UPDATE CASCADE  
);
```

Tankönyv példája

Módszer megadása

a hivatkozási épség megőrzésére:

```
CREATE TABLE Stúdió (  
    név CHAR(30) PRIMARY KEY,  
    cím VARCHAR(255),  
    elnökAzon INT REFERENCES  
        GyártásIrányító(azonosító)  
    ON DELETE SET NULL  
    ON UPDATE CASCADE  
);
```

Megszorítások ellenőrzésének késleltetése

- Körkörös megszorítások miatt szükség lehet arra, hogy a megszorításokat ne ellenőrizze, amíg az egész tranzakció be nem fejeződött.
- Bármelyik megszorítás deklarálnak **DEFERRABLE** (késleltethető) vagy **NOT DEFERRABLE**-ként (vagyis minden adatbázis módosításkor a megszorítás közvetlenül utána ellenőrzésre kerül). **DEFERRABLE**-ként deklaráljuk, akkor lehetőségünk van arra, hogy a megszorítás ellenőrzésével várjon a rendszer a tranzakció végéig.
- Ha egy megszorítás késleltethető, akkor lehet
 - **INITIALLY DEFERRED** (az ellenőrzés a tranzakció jóváhagyásáig késleltetve lesz) vagy
 - **INITIALLY IMMEDIATE** (minden utasítás után ellenőrzi)

Tankönyv példája

Az elnökAzon egyedisége és a megszorítás
késleltetése

```
CREATE TABLE Stúdió (  
    név CHAR(30) PRIMARY KEY  
    cím VARCHAR(255)  
    elnökAzon INT UNIQUE  
        REFERENCES GyártásIrányító(azonosító)  
        DEFERRABLE INITIALLY DEFERRED  
);
```

Attribútumokra vonatkozó megszorítások

- Egy adott oszlop értékeire vonatkozóan adhatunk meg megszorításokat.
- A CREATE TABLE utasításban az attribútum megadása után **NOT NULL** kulcsszóval
- Adjuk hozzá a **CHECK(<condition>)** feltételt az attribútum deklarációjához.
- A feltételben csak az adott attribútum neve szerepelhet, **más attribútumok (más relációk attribútumai is) csak alkérdésben szerepelhetnek.**

Példa: attribútum alapú ellenőrzés

```
CREATE TABLE Felszolgál (  
    bár CHAR(20),  
    sör CHAR(20) CHECK ( sör IN  
        (SELECT name FROM Sörök)),  
    ár REAL CHECK ( ár <= 5.00 )  
);
```

Mikor ellenőrzi?

- **Attribútum-alapú ellenőrzést csak **beszúrásnál** és **módosításnál** hajt végre a rendszer.**
 - **Példa:** CHECK (ár <= 5.00) a beszúrt vagy módosított sor értéke nagyobb 5, a rendszer nem hajtja végre az utasítást.
 - **Példa:** CHECK (sör IN (SELECT név FROM Sörök)), ha a Sörök táblából törölünk, ezt a feltételt nem ellenőrzi a rendszer.

Sorokra vonatkozó megszorítások

- A **CHECK (<feltétel>)** megszorítás a séma elemeként is megadható.
- A feltételben tetszőleges oszlop és reláció szerepelhet.
 - De más relációk attribútumai csak alkérdésben jelenhetnek meg.
- Csak beszúrásnál és módosításnál ellenőrzi a rendszer.

Példa: sor-alapú megszorítások

- Csak Joe bárjában lehetnek drágábbak a sörök 5 dollárnál:

```
CREATE TABLE Felszolgál (  
    bár CHAR(20),  
    sör CHAR(20),  
    ár REAL,  
    CHECK (bár = 'Joe bárja' OR  
           ár <= 5.00)  
);
```

Tankönyv példája

Attribútumokra és sorokra vonatkozó megszorítások

Példa: Ha egy színész neme férfi, akkor
a neve nem kezdődhet 'Ms.'-el

```
CREATE TABLE FilmSzínész (  
  név CHAR(30) PRIMARY KEY,  
  cím VARCHAR(255) NOT NULL,  
  nem CHAR(1),  
  születésiDátum DATE,  
  CHECK (nem = 'N' OR név NOT LIKE 'Ms.%')  
);
```

Megszorítások elnevezése

Tankönyv példái:

- név CHAR(30) **CONSTRAINT** NévKulcs PRIMARY KEY,
- nem CHAR(1) CONSTRAINT FérfiVagyNő
CHECK (nem IN ('F', 'N')),
- CONSTRAINT Titulus CHECK (nem = 'N' OR
név NOT LIKE 'Ms.\%')

Megszorítások módosítása

Tankönyv példái:

- ALTER TABLE FilmSzínész **ADD CONSTRAINT** NévKulcs PRIMARY KEY (név);
- ALTER TABLE FilmSzínész ADD CONSTRAINT FérfiVagyNő CHECK (nem IN ('F', 'N'));
- ALTER TABLE FilmSzínész ADD CONSTRAINT Titulus CHECK (nem = 'N' OR név NOT LIKE 'Ms.\%');