Adatbázisok 1.
SQL haladó – 2. rész

Külső összekapcsolások, Csoportosítás/Összesítés,

Beszúrás/Törlés/Módosítás,

Táblák létrehozása/Kulcs megszorítások

Adatbázis módosítások (*database modifications*)

- A *módosító* utasítások nem adnak vissza eredményt, mint a lekérdezések, hanem az adatbázis tartalmát változtatják meg.
- 3-féle módosító utasítás létezik:
  1. *Insert* (beszúrás, *insertion*).
  2. *Delete* (törlés, *deletion*).
  3. *Update* (frissítés, *update*, létező sorok értékeinek módosítása).
- Data Manipulation Language (DML).

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Beszúrás

- Ha egyetlen sort szúrunk be:

  INSERT INTO <reláció>

  VALUES (<attribútum lista>);

- Példa: a Szeret(vendég, tea) táblában rögzítjük, hogy Susy szereti a Brisk teát.

```
INSERT INTO Szeret
VALUES('Susy', 'Brisk');
```

Az INSERT-nél megadhatjuk az attribútumokat

- A reláció neve után megadhatjuk az attribútumait.
- Ennek két oka lehet:
  1. elfelejtettük, hogy a reláció definíciójában, milyen sorrendben szerepeltek az attribútumok.
  2. Nincs minden attribútumnak értéke, és azt szeretnénk, ha a hiányzó értékeket NULL vagy default értékkel helyettesítenék.

Példa: Attribútumok megadása

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

```
INSERT INTO Szeret(tea, vendég)
VALUES('Brisk', 'Susy');
```

Alapértelmezett értékek (*default values*) megadása

- A CREATE TABLE utasításban az oszlopnevet DEFAULT kulcsszó követheti és egy érték.
- Ha egy beszúrt sorban hiányzik az adott attribútum értéke, akkor a default értéket kapja.

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Alapértelmezett érték

```
CREATE TABLE Vendégek (
    név CHAR(30) PRIMARY KEY,
    cím CHAR(50) DEFAULT '123 Sesame St.',
    telefon CHAR(16)
);
```

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Alapértelmezett érték

```
INSERT INTO Vendégek(név)
VALUES('Susy');
```

Az eredmény sor:

név	cím	telefon
Susy	123 Sesame St	NULL

Több sor beszúrása

- Egy lekérdezés eredményét is beszúrhatjuk a következő módon:

```
INSERT INTO <reláció>
( <alkérdés> );
```

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Beszúrás alkérdéssel

- A Látogat(vendég, teázó) tábla felhasználásával adjuk hozzá a Ivócimborák (név) táblához Susy ivócimboráit, vagyis azokat, akikkel legalább egy közös teázót látogatnak.

Megoldás

Az ivócimborák.

Vendégpárok: az első Susy, a második nem Susy, de van olyan teázó, amit mindketten látogatnak.

```
INSERT INTO Ivócimborák
(SELECT l2.vendég
FROM Látogat l1, Látogat l2
WHERE l1.vendég = 'Susy' AND
  l2.vendég <> 'Susy' AND
  l1.teázó = l2.teázó
);
```

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Törlés

- A törlendő sorokat egy feltétel segítségével adjuk meg:

```
DELETE FROM <reláció>
WHERE <feltétel>;
```

Példa: Törlés

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

```
DELETE FROM Szeret
WHERE vendég = 'Susy' AND
        tea = 'Brisk';
```

Példa: Az összes sor törlése

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

```
DELETE FROM Szeret;
```

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Több sor törlése

- A Teák(név, gyártó) táblából töröljük azokat a teákat, amelyekhez létezik olyan tea, amit ugyanaz a cég gyártott.

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Több sor törlése

- A Teák(név, gyártó) táblából töröljük azokat a teákat, amelyekhez létezik olyan tea, amit ugyanaz a cég gyártott.

```
DELETE FROM Teák s
WHERE EXISTS (
    SELECT név FROM Teák
    WHERE gyártó = s.gyártó
        AND név <> s.név);
```

Azok a teák, amelyeknek ugyanaz a gyártója, mint az *s* éppen aktuális sorának, a nevük viszont különböző.

A törlés szemantikája --- (1)

- Tegyük fel, hogy a Lipton csak Brisk és Pyramid teákat gyárt.
- Tegyük fel még, hogy *s* sorai közt a Brisk fordul elő először.
- Az alkérdés nem üres, a későbbi Pyramid sor miatt, így a Brisk törlődik.
- Kérdés, hogy a Pyramid sor törlődik-e?

A törlés szemantikája --- (2)

- Válasz: igen, a Pyramid sora is törlődik.
- A törlés ugyanis két lépésben hajtódik végre.
  1. Kijelöljük azokat a sorokat, amelyekre a WHERE feltétele teljesül.
  2. Majd töröljük a kijelölt sorokat.

Módosítás

- Bizonyos sorok bizonyos attribútumainak módosítása.

UPDATE <reláció>

SET <attribútum értékadások listája>

WHERE <sorokra vonatkozó feltétel>;

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Módosítás

- Frank telefonszámát 555-1212-re változtatjuk (Frank itt egy vendég):

```
UPDATE Vendégek
SET telefon = '555-1212'
WHERE név = 'Frank';
```

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa: Több sor módosítása

- Legfeljebb 4 dollárba kerülhessenek a teák:

```
UPDATE Felszolgál
SET ár = 4.00
WHERE ár > 4.00;
```

Adatbázis sémák SQL-ben

- Data Definition Language (DDL), az SQL nyelv része, ennek segítségével hozhatunk létre adatobjektumokat, deklarálhatunk megszorításokat stb.

Relációk létrehozása

- A legegyszerűbb forma:

```
CREATE TABLE <név> (
        <elemek listája>
);
```

- Relációk törlése:

```
DROP TABLE <név>;
```

Tábla definíciók elemei

- A legegyszerűbb elem: az attribútum és annak típusa.
- A legfontosabb típusok a következők:
  - INT vagy INTEGER (szinonimák).
  - REAL vagy FLOAT (szinonimák).
  - CHAR(*n*) = rögzített hosszúságú sztring *n* karakter hosszú.
  - VARCHAR(*n*) = változó hosszúságú sztring legfeljebb *n* karakter hosszú.

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Példa : Create Table

```
CREATE TABLE Felszolgál (
    teázó    CHAR(20),
    tea      VARCHAR(20),
    ár       REAL
);
```

SQL értékek

- Az egészek és lebegőpontos típusú konstansokat csak „szimplán le kell írni”. (Pont jelöli a tizedesvesszőt.)
- A sztringek esetében aposztrófok közé kell tennünk a konstansokat.
  - Két aposztróf = egyetlen aposztróf, például: `'Joe''s Teahouse'`.
- Minden érték lehet NULL is.

Dátum és idő

- A DATE és TIME külön típusok az SQL-ben.
- A dátum típus formátuma:

  DATE ’yyyy-mm-dd’

  - Példa: `DATE '2007-09-30'` (2007. szeptember 30.)

Idő típus

- Az TIME típus formátuma:

  TIME ’hh:mm:ss’

Opcionálisan tizedespont is következhet a másodpercek után.

- Példa: `TIME ’15:30:02.5’` = fél négy múlt két és fél másodperccel.

Kulcsok megadása (deklarálása)

- Egy attribútumot vagy attribútum listát kulcsként deklarálhatunk (PRIMARY KEY vagy UNIQUE).
- Mindkét formája a megszorításnak azt követeli meg, hogy relációnak ne legyen két olyan sora, melyek megegyeznek a kulcs attribútumokon.
- A különbségekről később lesz szó.

Teák(<u>név</u>, gyártó)
Teázók(<u>név</u>, cím, engedélySzám)
Vendégek(<u>név</u>, cím, telefon)
Szeret(<u>vendég</u>, <u>tea</u>)
Felszolgál(<u>teázó</u>, <u>tea</u>, ár)
Látogat(<u>vendég</u>, <u>teázó</u>)

Egy attribútumos kulcs deklarálása

- PRIMARY KEY vagy UNIQUE kulcsszót írhatjuk közvetlenül az attribútum mögé.
- Példa:

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

Több attribútumú kulcsok megadása

- A kulcs deklaráció a CREATE TABLE utasítás egy eleme is lehet az attribútum-típus elemek után.
- Több attribútumú kulcsokat csak ebben a formában deklarálhatunk.
  - Ugyanakkor az egyetlen attribútumból álló kulcsokat is megadhatjuk ily módon.

Példa: Több attribútumú kulcs

Teák(név, gyártó)
Teázók(név, cím, engedélySzám)
Vendégek(név, cím, telefon)
Szeret(vendég, tea)
Felszolgál(teázó, tea, ár)
Látogat(vendég, teázó)

```
CREATE TABLE Felszolgál (
        teázó     CHAR(20),
        tea       VARCHAR(20),
        ár        REAL,
        PRIMARY KEY (teázó, tea)
);
```

PRIMARY KEY vs. UNIQUE

1. Egy relációhoz egyetlen PRIMARY KEY tartozhat és több UNIQUE megszorítás.
2. A PRIMARY KEY egyetlen attribútuma sem kaphat NULL értéket. A UNIQUE megszorításnál szerepelhetnek NULL értékek egy soron belül akár több is.

SQL példák

- Tegyük fel, hogy az alábbi táblát hozták létre:

```
CREATE TABLE Hallgatók (
      név CHAR(30) PRIMARY KEY,
      cím CHAR(50)
);
```

SQL példák

Lehet-e olyan hallgató, akinek nincs kitöltve (NULL) a neve?

Tegyük fel, hogy a név után mégsem szerepelne a PRIMARY KEY kulcsszó. Hány különböző nevű hallgató van?

```
SELECT COUNT(DISTINCT név) FROM Hallgatók;
```

Hogyan tudjuk meghatározni, hogy az azonos nevű hallgatókból hány darab van?

```
SELECT név, COUNT(név) FROM Hallgatók
GROUP BY név
HAVING COUNT(név) > 0;
```