# 5. előadás – SQL

## Berezvai Dániel jegyzete <http://elte.3ice.hu/>

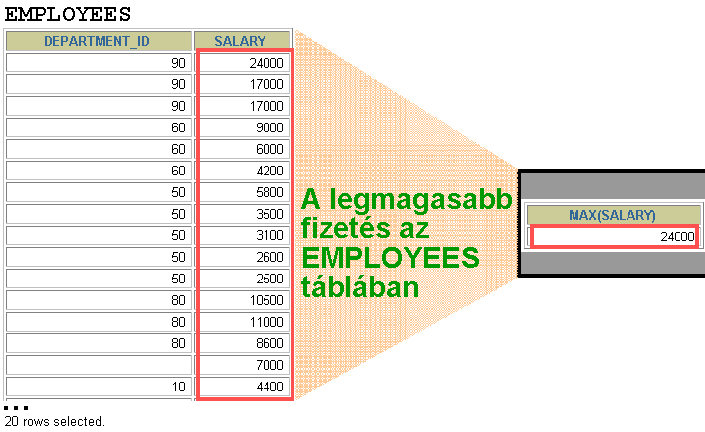
Hétfőn nem voltam.

Új diák reggel kerültek feltöltésre.

<http://people.inf.elte.hu/sila/edu14feb/AB1EA.html>

## Gyakorlati példák (ismétlés)

Az összesítő függvény csoportosított sorok halmazain működik, és egyetlen eredményt ad vissza csoportonként.

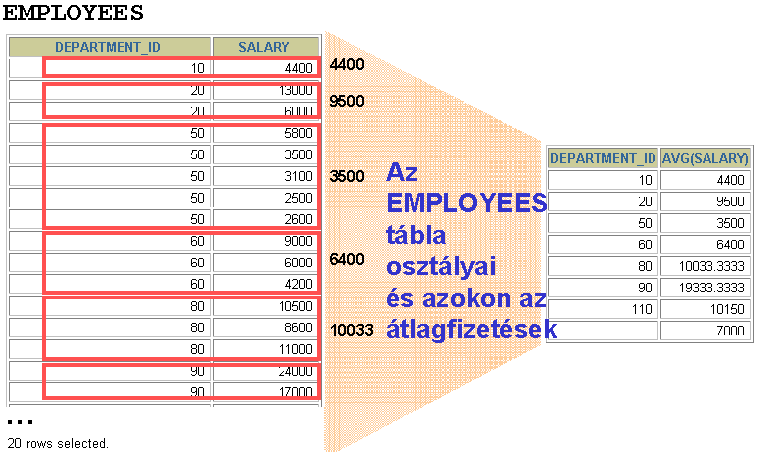


## Az aggregáló függvények típusai

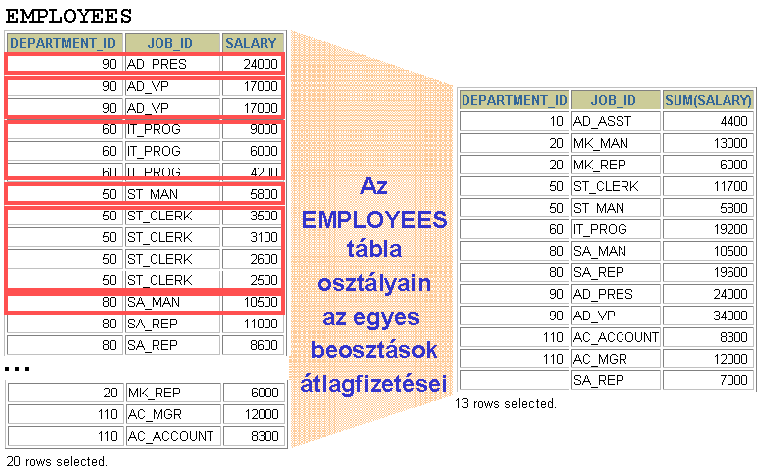
|  |  |
| --- | --- |
| Függvény | Leírása |
| AVG([DISTINCT|ALL]n) | n átlagértéke (nullértékeket kihagyva) |
| COUNT({\*|[DISTINCT|ALL]expr}) | Azon sorok száma, amelyekre expr kiértékelése nem null (DE: \* esetén az összes sorok száma, beleértve az ismétlődőket és a null értéket tartalmazókat is) |
| MAX([DISTINCT|ALL]expr) | Expr legnagyobb értéke (nullértékeket kihagyva) |
| MIN([DISTINCT|ALL]expr) | Expr legkisebb értéke (nullértékeket kihagyva) |
| STDDEV([DISTINCT|ALL]x) | n szórása (nullértékeket kihagyva) |
| SUM([DISTINCT|ALL]n) | n értékeinek összege (nullértékeket kihagyva) |
| VARIANCE([DISTINCT|ALL]x) | n szórásnégyzete (nullértékeket kihagyva) |

A MIN és MAX numerikus, karakteres és dátum típusú adatokra is használható, természetesen.

## Adatcsoportok létrehozása



## Csoportosítás több oszlopnév alapján



A GROUP BY után a HAVING a használandó szűrő, nem a WHERE. Először csoportosítunk, majd leszűrjük a kívánt csoportokat.

Az ORDER BY mindig a legutolsó.

## Összesítő függvények egymásba ágyazása

Csak két mélységig lehet. Az összesítő függvények csak kétszeres mélységig ágyazhatóak egymásba!

Például - Legnagyobb átlagfizetés:

SELECT MAX(AVG(salary))  
FROM employees  
GROUP BY department\_id;

Gyakorlás ZH-ra: Melyik az az osztály, ahol legmagasabb az átlagfizetés?

TODO

Most jön az új anyag:

# Kibővített relációs algebra

Az eddig tanult műveleteket: vetítés (Π), kiválasztás (σ), halmazműveletek: unió (∪), különbség (−), metszet (∪), szorzás: természetes összekapcsolás(⋈), direkt-szorzat (x), stb. multihalmazok fölött értelmezzük, mint az SQL-ben, egy reláció nem sorok halmazából, hanem multihalmazából áll, vagyis megengedett a sorok ismétlődése.

Ezeken kívül a SELECT kiegészítéseinek és záradékainak megfeleltetett új műveletekkel is kibővítjük a rel. algebrát:

* Delta – distinct: Ismétlődések megszüntetése () - select distinct
* Gamma – group: Összesítő műveletek és csoportosítás () - group by
* Pi – project: Vetítési művelet kiterjesztése () - select kif [as onev]
* Tau – tuple: Rendezési művelet () - order by
* Csokornyakkendő feletti o – outer: Külső összekapcsolások () – [left | right | full] outer join

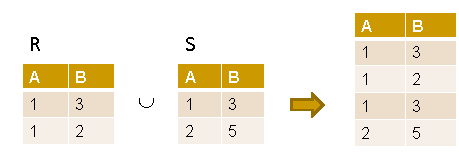
ZH-n kell tudni ezeket a szimbólumokat.

## Multihalmazok egyesítése, metszete, különbsége

Unió: R ∪ S-ben egy t sor annyiszor fordul elő, ahányszor előfordul R-ben, plusz ahányszor előfordul S-ben: n+m

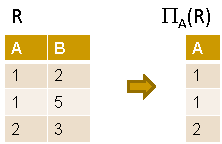
Metszet: R ∩ S-ben egy t sor annyiszor fordul elő, amennyi az R-ben és S-ben lévő előfordulások minimuma: min[n, m]

Különbség: R − S-ben egy t sor annyiszor fordul elő, mint az R-beli előfordulások mínusz az S-beli előfordulások száma, ha ez pozitív, egyébként pedig 0, vagyis max[0, n-m]



## A "többi művelet" multihalmazok fölött

A projekció, szelekció, Descartes-szorzat, természetes összekapcsolás és Théta-összekapcsolás végrehajtása során nem küszöböljük ki az ismétlődéseket.



## Ismétlődések megszüntetése – duplikátumok kiszűrése

Ismétlődések megszüntetése: R1:= δ(R2)

A művelet jelentése: R2 multihalmazból R1 halmazt állít elő, vagyis az R2-ben egyszer vagy többször előforduló sorok csak egyszer szerepelnek az R1-ben.

A DISTINCT reprezentálására szolgál

## Összesítő függvények

SUM, COUNT, MIN, MAX, AVG

## Csoportosítás és összesítés

A művelet a csoportosítást (GROUP BY), a csoportokon végezhető összesítő függvényeket (AVG, SUM, COUNT, MIN, MAX) reprezentálja.

Jele: Gamma alsóindex lista

Itt az L lista valamennyi eleme a következők egyike:

* R egy attribútuma: ez az attribútum egyike a csoportosító attribútumoknak (a GROUP BY után jelenik meg).
* a reláció egyik attribútumára (ez az összesítési attribútumra) alkalmazott összesítő operátor, ha az összesítés eredményére névvel szeretnénk hivatkozni, akkor nyilat és új nevet használunk.

### Menete

Osszuk R sorait csoportokba. Egy csoport azokat a sorokat tartalmazza, amelyek az L listán szereplő csoportosítási attribútumokhoz tartozó értékei megegyeznek

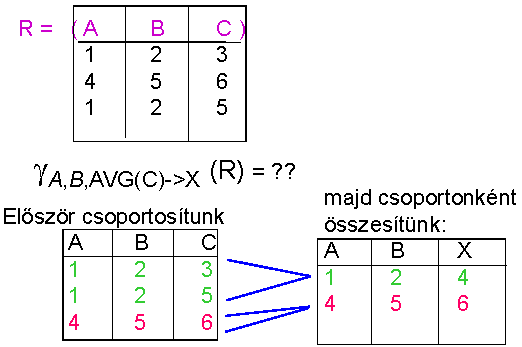
* Vagyis ezen attribútumok minden egyes különböző értéke egy csoportot alkot.

Minden egyes csoporthoz számoljuk ki az L lista összesítési attribútumaira vonatkozó összesítéseket

Az eredmény minden egyes csoportra egy sor:

1. A csoportosítási attribútumok és
2. Az összesítési attribútumra vonatkozó összesítések (az adott csoport összes sorára)

### Példa



### Tankönyv példája

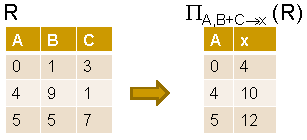
Adjuk meg azokat a színészeket, akik már szerepeltek legalább három filmben illetve adjuk meg az első olyan filmet is, amiben szerepelt:

SELECT  
 HAVING  
 GROUP BY

## Általánosított vetítés

L listája az SQL SELECT záradékához hasonlóan tartalmazhatja:

* R egy attribútumát,
* , ahol x, y attribútumnevek, s itt -et -ra nevezzük át,
* , ahol R attribútumait, konstansokat, aritmetikai operátorokat és karakterlánc operátorokat tartalmazhat például:



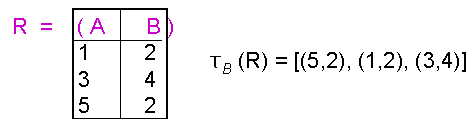
## Rendezés

Rendezés:

Először attribútum szerint rendezzük sorait. Majd azokat a sorokat, amelyek értéke megegyezik az attribútumon, szerint, és így tovább.

Hasonlóan az ORDER BY működéséhez.

Ez az egyetlen olyan művelet, amelynek az eredménye se nem halmaz se nem multihalmaz.



## Külső összekapcsolások

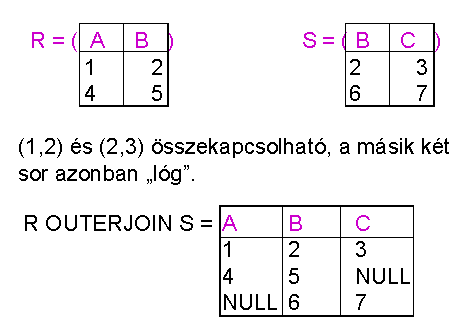
Ez nem relációs algebrai művelet, ugyanis kilép a modellből.

Lehet baloldali, jobboldali, teljes külső összekapcsolás.

R, S sémái , ill.

relációt kiegészítjük az R és S soraival, a hiányzó helyekre NULL értéket írva megőrzi a "lógó sorokat"

### Példa



# Lekérdezések az SQL-ben

Most jön… Visszatérünk az SQL 1999-es szabványban a FROM záradékban az összekapcsolások új szintaxisára és a külső összekapcsolásokra

## Összekapcsolások az SQL-ben

Az SQL-ben összekapcsolások számos változata megtalálható:

SELECT tábla1.oszlop, tábla2.oszlop

FROM tábla1

[NATURAL JOIN tábla2] |

[JOIN tábla2 USING (oszlopnév)] |

[JOIN tábla2 ON (tábla1.oszlopnév = tábla2.oszlopnév)]

[{LEFT | RIGHT | FULL} OUTER JOIN tábla2 ON (tábla1.oszlopnév = tábla2.oszlopnév)]

[CROSS JOIN tábla2] vagy simán tábla2, mert CROSS JOIN a default.

## Descartes szorzat és természetes összekapcsolás

Descartes szorzat (ez megegyezik: R, S)

R CROSS JOIN S;

Természetes összekapcsolás:

R NATURAL JOIN S;

Példa:

Kedvel NATURAL JOIN Felszolgál;

A relációk helyén zárójelezett alkérdések is szerepelhetnek. (Rendeljünk hozzá változót, hogy tudunk rá hivatkozni.)

## Théta-összekapcsolás

R JOIN S ON <feltétel>

Példa: Tegyük fel, hogy a Látogat táblában most ivó oszlopként szerepelnek a sörivók nevei Ivók(név, cím) és Látogat(ivó, bár) táblákból:

Ivók JOIN Látogat ON név = ivó;

Azokat (n, c, i, b) négyeseket adja vissza, ahol a n sörivó c címen lakik és a b bárt látogatja.

## Külső összekapcsolás

Összekapcsoljuk R és S relációkat: .

R azon sorait, melyeknek nincs S-beli párja lógó soroknak nevezzük. S-nek is lehetnek lógó sorai.

A külső összekapcsolás megőrzi a lógó sorokat NULL értékkel helyettesítve a hiányzó értékeket.

## Külső összekapcsolás (SQL)

R OUTER JOIN S: a külső összekapcsolásoknál mindig ez szerepel.

1. Opcionális NATURAL az OUTER előtt.

2. Opcionális ON <feltétel> JOIN után.

3. Opcionális LEFT, RIGHT, vagy FULL az OUTER előtt.

LEFT = csak R lógó sorait őrzi meg.

RIGHT = csak S lógó sorait őrzi meg.

FULL = az összes lógó sort megőrzi.

Csak az egyik szerepelhet.

## Ismétlődések megszüntetése

SELECT DISTINCT … FROM …

A művelet SQL-beli megfelelője

## Ismétlődések kezelése halmazművelet során

(SELECT … FROM … )  
{UNION | INTERSECT | EXCEPT} [ALL]  
(SELECT … FROM … )

Lekérdezések között lehetnek a halmazműveletek.

Az ALL kulcsszóval ezek a műveletek multihalmaz-szemantika szerint működnek.

# Az SQL adatbázisnyelv: DML

INSERT, DELETE, UPDATE

DML már nem lesz a jövő heti ZH-n, de vizsgán már igen.

## Adatbázis tartalmának módosítása

A módosító utasítások nem adnak vissza eredményt, mint a lekérdezések, hanem az adatbázis tartalmát változtatják meg.

3-féle módosító utasítás létezik:

INSERT - sorok beillesztése, beszúrása

DELETE – sorok törlése

UPDATE – sorok komponensei értékeinek módosítása

### Tranzakciók az SQL-ben

Ez nem törzsanyag - nincs vizsgán, később lesz az Adatbázisok-2-n

SAVEPOINT, COMMIT és ROLLBACK

→ AB1\_05D\_SQL8tranz.pdf

## Beszúrás

Ha egyetlen sort szúrunk be:

INSERT INTO <reláció>

VALUES ( … );

Példa: a Kedvel(név, sör) táblában rögzítjük, hogy Zsu szereti a Bud sört.

INSERT INTO Likes

VALUES(’Zsu’, ’Bud’);

## Attribútumok megadása

A reláció neve után megadhatjuk az attribútumait.

Ennek alapvetően 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:

INSERT INTO Kedvel(sör, név)

VALUES(’Bud’, ’Zsu’);

## Default értékek 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.

### Példa

CREATE TABLE Sörivók(

név CHAR(30) PRIMARY KEY,

cím CHAR(50) DEFAULT ’Sesame St’

telefon CHAR(16) );

INSERT INTO Sörivók(név)

VALUES(’Zsu’);

Az eredmény sor:

|  |  |  |
| --- | --- | --- |
| név | cím | telefon |
| Zsu | Sesame St | NULL |

## Több sor beszúrása

Ezt ne felejtsük el vizsgán leírni, ha az insert delete update jellemzése a kérdés!

Egy lekérdezés eredményét is beszúrhatjuk a következő módon alkérdéssel:

INSERT INTO <reláció>

( <alkérdés> );

A Látogat(név, bár) tábla felhasználásával adjuk hozzá a LehetBarát(név) táblához Zsu „lehetséges barátait”, vagyis azokat a sörivókat, akik legalább egy olyan bárat látogatnak, ahova Zsu is szokott járni.

INSERT INTO LehetBarát

(SELECT l2.név  
FROM Látogat l1, Látogat l2  
WHERE l1.név = ’Zsu’ AND l2.név <> ’Zsu’ AND l1.bár = l2.bár  
);

(SELECT) a másik sörivó

(FROM) névpárok: az első Zsu, a második nem Zsu, de van olyan bár, amit mindketten látogatnak.

## Törlés

A törlendő sorokat egy WHERE feltétel segítségével adjuk meg:

DELETE FROM <reláció>

WHERE <feltétel>;

Példa:

DELETE FROM Kedvel

WHERE nev = ’Zsu’ AND sör = ’Bud’;

Az összes sor törlése:

DELETE FROM Kedvel;

### Példa: Több sor törlése

A Sörök(név, gyártó) táblából töröljük azokat a söröket, amelyekhez létezik olyan sör, amit ugyanaz a cég gyártott:

DELETE FROM Sörök s

WHERE EXISTS (  
SELECT név FROM Sörök  
WHERE gyártó = s.gyártó AND név <> s.név  
);

(WHERE) azok a sörök, amelyeknek ugyanaz a gyártója, mint az s éppen aktuális sorának, a nevük viszont különböző.

## A törlés szemantikája

Tegyük fel, hogy az Anheuser-Busch csak Bud és Bud Lite söröket gyárt.

Tegyük fel még, hogy s sorai közt a Bud fordul elő először.

Az alkérdés nem üres, a későbbi Bud Lite sor miatt, így a Bud törlődik.

Kérdés, hogy a Bud Lite sor törlődik-e?

Válasz: igen, a Bud Lite 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>;

### Példa

Fecó telefonszámát 555-1212-re változtatjuk (Fecó itt egy sörivó neve):

UPDATE Sörivók

SET telefon = ’555-1212’

WHERE név = ’Fecó’;

### Példa: Több sor módosítása

Legfeljebb 4 dollárba kerülhessenek a sörök:

UPDATE Felszolgál

SET ár = 4.00

WHERE ár > 4.00;

Olcsó sörök árát duplázzuk

UPDATE Felszolgál

SET ár = 2 \* ár

WHERE ár < 1.00;

Tankönyv 6.39 példa:

UPDATE GyártásIrányító

SET név = ‘Ig.’ || név

WHERE azonosító IN (SELECT elnökAzon FROM Stúdió)

Előadás vége.