

Adatbázisok 1.

SQL haladó – 1. rész

Külső összekapcsolások, Csoportosítás/Összesítés,
Beszúrás/Törlés/Módosítás,
Táblák létrehozása/Kulcs megszorítások

Összekapcsolás típusok

- Descartes-szorzat (CROSS JOIN)
- Belső összekapcsolás (NATURAL JOIN, INNER JOIN...)
- Külső összekapcsolás (OUTER JOIN)

Belső összekapcsolások

- Természetes összekapcsolás
- Théta összekapcsolás
- Félig összekapcsolás (semi join, jelölés: \bowtie):
 - $R(A_1, \dots, A_n), S(B_1, \dots, B_m)$ sémájú táblák esetén: $R \bowtie S \equiv \Pi_{A_1, A_2, \dots, A_n}(R \mid X \mid S)$
- Anti összekapcsolás (anti join, jelölés: \Join):
 - $R \Join S \equiv R - R \bowtie S$

Relációs algebra és az SQL

R

A	B
a1	1
a2	2

$R \bowtie S$

R.A	B
a1	1

$R \triangleright S$

R.A	B
a2	2

S

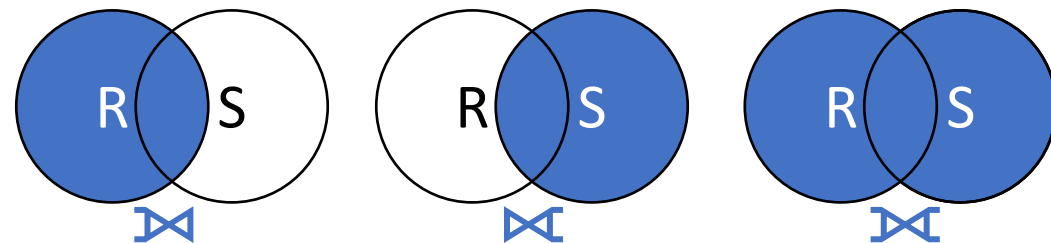
A	C
a1	2
a3	3

```
SELECT R.* FROM R
WHERE EXISTS(
  SELECT * FROM S
  WHERE R.A = S.A);
```

```
SELECT R.* FROM R
WHERE NOT EXISTS(
  SELECT * FROM S
  WHERE R.A = S.A);
```

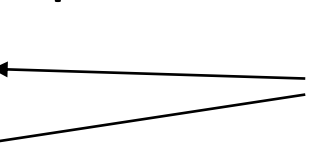


Külső összekapcsolás



- Kiterjesztett relációs algebra
- Összekapcsoljuk **R** és **S** relációkat: $R \bowtie_c S$, $R \bowtie_c S$, $R \bowtie_c S$
 - A „természetes” változatnál nincs az alsó indexben „C”.
- Három különböző művelet
- **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 <típus> OUTER JOIN S: a külső összekapcsolásoknál mindig ez szerepel.
 1. Opcionális NATURAL az OUTER előtt.
 2. Opcionális ON <feltétel> az S után.
 3. 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.
- 

Példa: Külső összekapcsolások

(Kiterjesztett relációs algebra és az SQL)

R

A	B
a1	1
a2	2

R ⋈_{R.A=S.A} **S**

R.A	B	S.A	C
a1	1	a1	2
a2	2	NULL	NULL

S

A	C
a1	2
a3	3

```
SELECT * FROM R  
LEFT OUTER JOIN S  
ON R.A = S.A;
```

Példa: Külső összekapcsolások

(Kiterjesztett relációs algebra és az SQL)

R

A	B
a1	1
a2	2

R ⋈_{R.A=S.A} **S**

R.A	B	S.A	C
a1	1	a1	2
a2	2	NULL	NULL

R ⋈_{R.A=S.A} **S**

R.A	B	S.A	C
a1	1	a1	2
NULL	NULL	a3	3

S

A	C
a1	2
a3	3

```
SELECT * FROM R  
LEFT OUTER JOIN S  
ON R.A = S.A;
```

```
SELECT * FROM R  
RIGHT OUTER JOIN S  
ON R.A = S.A;
```


Példa: Külső összekapcsolások

(Kiterjesztett relációs algebra és az SQL)

R

A	B
a1	1
a2	2

R ⋈_{R.A=S.A} **S**

R.A	B	S.A	C
a1	1	a1	2
a2	2	NULL	NULL

R ⋈_{R.A=S.A} **S**

R.A	B	S.A	C
a1	1	a1	2
NULL	NULL	a3	3

R ⋈_{R.A=S.A} **S**

R.A	B	S.A	C
a1	1	a1	2
a2	2	NULL	NULL
NULL	NULL	a3	3

S

A	C
a1	2
a3	3

```
SELECT * FROM R  
LEFT OUTER JOIN S  
ON R.A = S.A;
```

```
SELECT * FROM R  
RIGHT OUTER JOIN S  
ON R.A = S.A;
```

```
SELECT * FROM R  
FULL OUTER JOIN S  
ON R.A = S.A;
```

Példa: Külső összekapcsolások

(Kiterjesztett relációs algebra és az SQL)

R

A	B
a1	1
a2	2

R ⋈ S

A	B	C
a1	1	2
a2	2	NULL

R ⋈ S

A	B	C
a1	1	2
a3	NULL	3

```
SELECT * FROM R  
NATURAL LEFT OUTER JOIN S;
```

```
SELECT * FROM R  
NATURAL RIGHT OUTER JOIN S;
```

S

A	C
a1	2
a3	3

R ⋈ S

A	B	C
a1	1	2
a2	2	NULL
a3	NULL	3

```
SELECT * FROM R  
NATURAL FULL OUTER JOIN S;
```

Összesítések (aggregációk)

- SUM, AVG, COUNT, MIN, és MAX összesítő függvényeket a SELECT záradékban alkalmazhatjuk egy-egy oszlopra.
- COUNT(*) az eredmény sorainak számát adja meg.

Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgál(kocsma, sör, ár)
Látogat(alkesz, kocsma)

Példa: Összesítés

- A Felszolgál(kocsm, sör, ár) tábla segítségével adjuk meg a Bud átlagos árát:

```
SELECT AVG(ár)
FROM Felszolgál
WHERE sör = 'Bud';
```

Ismétlődések kiküszöbölése összesítésben

- Az összesítő függvényen belül DISTINCT.
- **Példa:** hány *különféle* áron árulják a Bud sört?

```
SELECT COUNT(DISTINCT ár)
FROM Felszolgál
WHERE sör = 'Bud';
```

NULL értékek nem számítanak az összesítésben

- NULL soha nem számít a SUM, AVG, COUNT, MIN, MAX függvények kiértékelésekor.

NULL értékek nem számítanak az összesítésben

- NULL soha nem számít a SUM, AVG, COUNT, MIN, MAX függvények kiértékelésekor.
- De ha nincs NULL értéktől különböző érték az oszlopban, akkor az összesítés eredménye NULL.
 - **Kivétel:** COUNT az üreshalmazon 0-t ad vissza.

Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgal(kocsma, sör, ár)
Látogat(alkesz, kocsma)

Példa: NULL értékek összesítésben

```
SELECT count(*)  
FROM Felszolgal  
WHERE sör = 'Bud';
```

A Bud sört árusító
kocsmák száma.



```
SELECT count(ár)  
FROM Felszolgal  
WHERE sör = 'Bud';
```


Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgal(kocsma, sör, ár)
Látogat(alkesz, kocsma)

Példa: NULL értékek összesítésben

```
SELECT count(*)  
FROM Felszolgal  
WHERE sör = 'Bud';
```

A Bud sört árusító
kocsmák száma.

```
SELECT count(ár)  
FROM Felszolgal  
WHERE sör = 'Bud';
```

Azon kocsmák száma,
ahol ismerjük a Bud sör
árát.

Csoportosítás

- Egy SELECT-FROM-WHERE kifejezést GROUP BY záradékkal folytathatunk, melyet attribútumok listája követ.
- A SELECT-FROM-WHERE eredménye a megadott attribútumok értékei szerint csoportosítódik, az összesítéseket ekkor minden csoportra külön alkalmazzuk.

Példa: Csoportosítás

- A **Felszolgál(kocsmá, sör, ár)** tábla segítségével adjuk meg a sörök átlagos árát.

```
SELECT sör, AVG(ár)
FROM Felszolgál
GROUP BY sör;
```

sör	AVG(ár)
Bud	2.33
Miller	2.45

Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgál(kocsmá, sör, ár)
Látogat(alkesz, kocsmá)

Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgál(kocsma, sör, ár)
Látogat(alkesz, kocsma)

Példa: Csoportosítás

- SELECT alkesz, AVG(ár)
FROM Látogat, Felszolgál
WHERE sör = 'Bud' AND
Látogat.kocsm =
Felszolgál.kocsm
GROUP BY alkesz;

Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgal(kocsma, sör, ár)
Látogat(alkesz, kocsma)

Példa: Csoportosítás

- SELECT alkesz, AVG(ár)
FROM Látogat, Felszolgal
WHERE sör = 'Bud' AND
Látogat.kocsm =
Felszolgal.kocsm
GROUP BY alkesz;

Alkesz-
kocsm-ár
hármask a
Bud sörre.

Sörök(név, gyártó)
Kocsmák(név, cím, engedélySzám)
Alkeszek(név, cím, telefon)
Szeret(alkesz, sör)
Felszolgal(kocsma, sör, ár)
Látogat(alkesz, kocsma)

Példa: Csoportosítás

- SELECT alkesz, AVG(ár)

```
FROM Látogat, Felszolgal  
WHERE sör = 'Bud' AND  
      Látogat.kocsm =  
      Felszolgal.kocsm
```

```
GROUP BY alkesz;
```

Alkesz-
kocsm-ár
hármask a
Bud sörre.

Alkeszek
szerinti
csoportosítás.

A SELECT lista és az összesítések

- Ha összesítés is szerepel a lekérdezésben, a SELECT-ben felsorolt attribútumokra a következő érvényes
 1. Összesítések, amelyekben egy összesítési operátort alkalmazunk egy attribútumra vagy egy attribútumot tartalmazó kifejezésre. Ezek a kifejezések csoportonként kerülnek kiértékelésre.
 2. Attribútumok, amelyek a GROUP BY záradékban szerepelnek, mint a példában *alkesz*. Egy összesítéseket tartalmazó SELECT záradékban csak a GROUP BY záradékban is megtalálható attribútumok jelenhetnek meg összesítési operátor nélkül.

Helytelen lekérdezés

- Elsőre sokan gondolhatják azt, hogy az alábbi lekérdezés a Bud sört legolcsóbban áruló kocsmát adja vissza:
- ```
SELECT kocsma, MIN(ár)
FROM Felszolgál
WHERE sör= 'Bud';
```
- Valójában ez egy helytelen SQL lekérdezés.



# HAVING záradék

- A GROUP BY záradékot egy HAVING <feltétel> záradék követheti.
- Ebben az esetben a feltétel az egyes csoportokra vonatkozik, ha egy csoport nem teljesíti a feltételt, nem lesz benne az eredményben.

Sörök(név, gyártó)  
Kocsmák(név, cím, engedélySzám)  
Alkeszek(név, cím, telefon)  
Szeret(alkesz, sör)  
Felszolgál(kocsma, sör, ár)  
Látogat(alkesz, kocsma)

## Példa: HAVING

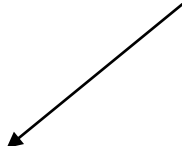
- A Felszolgál(kocsm, sör, ár) és Sörök(név, gyártó) táblák felhasználásával adjuk meg az átlagos árát azon söröknek, melyeket legalább három kocsmában felszolgálnak, vagy Pete a gyártójuk.

Sörök(név, gyártó)  
Kocsmák(név, cím, engedélySzám)  
Alkeszek(név, cím, telefon)  
Szeret(alkesz, sör)  
Felszolgál(kocsma, sör, ár)  
Látogat(alkesz, kocsma)

# Megoldás

```
SELECT sör, AVG(ár)
FROM Felszolgál
GROUP BY sör
```

Sör csoportok, melyeket  
legalább 3 nem-NULL  
kocsmában árulnak



```
HAVING COUNT(kocsm
```

```
sör IN (SELECT név
```

```
FROM Sörök
```

```
WHERE gyártó = 'Pete');
```

Sörök(név, gyártó)  
Kocsmák(név, cím, engedélySzám)  
Alkeszek(név, cím, telefon)  
Szeret(alkesz, sör)  
Felszolgál(kocsma, sör, ár)  
Látogat(alkesz, kocsma)

# Megoldás

```
SELECT sör, AVG(ár)
FROM Felszolgál
GROUP BY sör
```

Sör csoportok, melyeket  
legalább 3 nem-NULL  
kocsmában árulnak, vagy  
Pete a gyártójuk.

```
HAVING COUNT(kocsm
```

```
sör IN (SELECT név
FROM Sörök
WHERE gyártó = 'Pete');
```

Sörök,  
melyeket  
Pete gyárt.

# A HAVING feltételére vonatkozó megszorítások

- Az alkérdésre nincs megszorítás.
- Az alkérdésen kívül ugyanazok a szabályok érvényesek, mint a SELECT záradéknál
- A FROM záradékban megadott relációk bármely attribútumára képezhetünk a HAVING záradékban összesítést, összesítés nélkül a HAVING záradékban csak azok az attribútumok fordulhatnak elő, amelyek a GROUP BY listában is szerepeltek. (Ugyanaz a szabály, mint ami a SELECT záradéokra is vonatkozott.)
- A HAVING záradékban hivatkozott összesítés csak az éppen feldolgozott csoport soraira vonatkozik.