

Adatbázisok elmélete 9. előadás

Katona Gyula Y.

Budapesti Műszaki és Gazdaságtudományi Egyetem
Számítástudományi Tsz.

I. B. 137/b

`kiskat@cs.bme.hu`

`http://www.cs.bme.hu/~kiskat`

2004

SELECT

Ezzel valósítható meg a kiválasztás, vetítés és a szorzat.

Szintaxis: **SELECT** <reláció_{*j*}>.<attrib₁>, ..., <reláció_{*j*}>.<attrib_{*n*}>
FROM <reláció₁>, ..., <reláció_{*m*}>
WHERE <kifejezés>

Relációs algebrabeli megfelelője (de nem pontosan, mert SQL-ben SELECT nem küszöböli ki a többszörös sorokat):

$$\pi_{\langle \text{attrib}_1, \dots, \text{attrib}_n \rangle} \sigma_{\langle \text{kifejezés} \rangle} (\langle \text{reláció}_1 \rangle \times \dots \times \langle \text{reláció}_m \rangle)$$

SQL Sor- és oszlopváltozók

A FROM után felsorolt relációkhoz **sorváltozókat** rendelhetünk.

Szintaxis (FROM után <reláció_i> helyén): <reláció_i> AS <sorváltozó>

SQL Sor- és oszlopváltozók

A FROM után felsorolt relációkhoz **sorváltozókat** rendelhetünk.

Szintaxis (FROM után <reláció_i> helyén): <reláció_i> AS <sorváltozó>

A SELECT után elhelyezett attribútum-hivatkozásokhoz **oszlopváltozókat** rendelhetünk.

Szintaxis (SELECT után <reláció_i>.<attrib_j> helyén):

<reláció_i>.<attrib_j> AS <oszlopváltozó>

SQL Sor- és oszlopvátozók

A FROM után felsorolt relációkhoz **sorvátozókat** rendelhetünk.

Szintaxis (FROM után <reláció_i> helyén): <reláció_i> AS <sorvátozó>

A SELECT után elhelyezett attribútum-hivatkozásokhoz **oszlopvátozókat** rendelhetünk.

Szintaxis (SELECT után <reláció_i>.<attrib_j> helyén):

<reláció_i>.<attrib_j> AS <oszlopvátozó>

Így átnevezés lehetséges az eredmény megjelenítésekor:

Például:

SELECT név AS Filmszínház,
város AS Hely FROM mozi

⇒

	Filmszínház	Hely
	⋮	

SQL Sor- és oszlopváltozók

A FROM után felsorolt relációkhoz **sorváltozókat** rendelhetünk.

Szintaxis (FROM után <reláció_i> helyén): <reláció_i> AS <sorváltozó>

A SELECT után elhelyezett attribútum-hivatkozásokhoz **oszlopváltozókat** rendelhetünk.

Szintaxis (SELECT után <reláció_i>.<attrib_j> helyén):

<reláció_i>.<attrib_j> AS <oszlopváltozó>

Így átnevezés lehetséges az eredmény megjelenítésekor:

Például:

SELECT név AS Filmszínház,
város AS Hely FROM mozi

⇒

	Filmszínház	Hely
	⋮	

Az oszlopváltozók valójában csak az eredményreláció attribútumainak elnevezésére használhatók, a SELECT utasításon belül nem hivatkozhatunk rájuk.

SQL Sor- és oszlopváltozók

A FROM után felsorolt relációkhoz **sorváltozókat** rendelhetünk.

Szintaxis (FROM után <reláció_i> helyén): <reláció_i> AS <sorváltozó>

A SELECT után elhelyezett attribútum-hivatkozásokhoz **oszlopváltozókat** rendelhetünk.

Szintaxis (SELECT után <reláció_i>.<attrib_j> helyén):

<reláció_i>.<attrib_j> AS <oszlopváltozó>

Így átnevezés lehetséges az eredmény megjelenítésekor:

Például:

SELECT név AS Filmszínház,
város AS Hely FROM mozi

⇒

	Filmszínház	Hely
	⋮	

Az oszlopváltozók valójában csak az eredményreláció attribútumainak elnevezésére használhatók, a SELECT utasításon belül nem hivatkozhatunk rájuk.

A <reláció_i>. előtag elhagyható, ha egyértelmű, hogy melyik relációról van szó, továbbá a <reláció_i>. előtag helyett <sorváltozó>. előtag is szerepeltethető.

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Példa 1: A pénteken vetített filmek címei és rendezői (természetes illesztés)

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Példa 1: A pénteken vetített filmek címei és rendezői (természetes illesztés)

```
SELECT cím, rendező FROM film, vetít WHERE vetít.filmID = film.filmID AND nap='péntek'
```

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Példa 1: A pénteken vetített filmek címei és rendezői (természetes illesztés)

SELECT cím, rendező **FROM** film, vetít **WHERE** vetít.filmID = film.filmID **AND** nap='péntek'

Példa 2: Azok a várospárok, ahol vannak azonos nevű mozik

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Példa 1: A pénteken vetített filmek címei és rendezői (természetes illesztés)

```
SELECT cím, rendező FROM film, vetít WHERE vetít.filmID = film.filmID AND nap='péntek'
```

Példa 2: Azok a várospárok, ahol vannak azonos nevű mozik

```
SELECT m1.város, m2.város FROM mozi AS m1, mozi AS m2 WHERE m1.név = m2.név  
AND m1.város <> m2.város
```

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Példa 1: A pénteken vetített filmek címei és rendezői (természetes illesztés)

```
SELECT cím, rendező FROM film, vetít WHERE vetít.filmID = film.filmID AND nap='péntek'
```

Példa 2: Azok a várospárok, ahol vannak azonos nevű mozik

```
SELECT m1.város, m2.város FROM mozi AS m1, mozi AS m2 WHERE m1.név = m2.név  
AND m1.város <> m2.város
```

Megjegyzés: a várospárok mindkét sorrendben megjelennek, és több azonos nevű mozi esetén többször is megjelennek.

Attribútumhivatkozások

Amikor egy attribútumra akarunk hivatkozni, három lehetőségünk van:

- **<attribútum>** (ha ez egyértelmű)
- **<reláció>.<attribútum>** (ha ez egyértelmű – N.B.: egy reláció többször is szerepelhet a FROM után, lesz példa)
- **<sorváltozó>.<attribútum>** (mindig használható)

Példa 1: A pénteken vetített filmek címei és rendezői (természetes illesztés)

```
SELECT cím, rendező FROM film, vetít WHERE vetít.filmID = film.filmID AND nap='péntek'
```

Példa 2: Azok a várospárok, ahol vannak azonos nevű mozik

```
SELECT m1.város, m2.város FROM mozi AS m1, mozi AS m2 WHERE m1.név = m2.név  
AND m1.város <> m2.város
```

Megjegyzés: a várospárok mindkét sorrendben megjelennek, és több azonos nevű mozi esetén többször is megjelennek.

Az elsőre megoldás: <> helyett legyen <, amúgy meg **DISTINCT**

```
SELECT DISTINCT m1.város, m2.város FROM mozi AS m1, mozi AS m2  
WHERE m1.név = m2.név AND m1.város < m2.város
```


A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...
- karakterlánc műveletek, összehasonlítás: CONCAT (||), LENGTH, LOWER, SUBSTR, SOUNDEX, ...

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...
- karakterlánc műveletek, összehasonlítás: CONCAT (||), LENGTH, LOWER, SUBSTR, SOUNDEX, ...
- halmazba tartozás: IN (halmaz), ...

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...
- karakterlánc műveletek, összehasonlítás: CONCAT (||), LENGTH, LOWER, SUBSTR, SOUNDEX, ...
- halmazba tartozás: IN (halmaz), ...
- változóhivatkozások: <sorváltozó>.<attribútum>, <reláció>.<attribútum>, <attribútum>

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...
- karakterlánc műveletek, összehasonlítás: CONCAT (||), LENGTH, LOWER, SUBSTR, SOUNDEX, ...
- halmazba tartozás: IN (halmaz), ...
- változóhivatkozások: <sorváltozó>.<attribútum>, <reláció>.<attribútum>, <attribútum>
- konstans (szám,karakterlánc): 137, 42e-3, 'füzér', ...

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...
- karakterlánc műveletek, összehasonlítás: CONCAT (||), LENGTH, LOWER, SUBSTR, SOUNDEX, ...
- halmazba tartozás: IN (halmaz), ...
- változóhivatkozások: <sorváltozó>.<attribútum>, <reláció>.<attribútum>, <attribútum>
- konstans (szám,karakterlánc): 137, 42e-3, 'füzér', ...
- NULL érték vizsgálata: IS NULL, IS NOT NULL (később lesz)

A WHERE kifejezés

Kifejezés felépítése:

- logikai műveletek: AND, OR, NOT
- összehasonlítás: =, <>, >=, <=, LIKE, BETWEEN
- aritmetikai műveletek: +, -, *, /, MOD, POWER, LN, SIN, COS, ...
- karakterlánc műveletek, összehasonlítás: CONCAT (||), LENGTH, LOWER, SUBSTR, SOUNDEX, ...
- halmazba tartozás: IN (halmaz), ...
- változóhivatkozások: <sorváltozó>.<attribútum>, <reláció>.<attribútum>, <attribútum>
- konstans (szám,karakterlánc): 137, 42e-3, 'füzér', ...
- NULL érték vizsgálata: IS NULL, IS NOT NULL (később lesz)
- alkérdés is lehet itt: (majd erről később)

LIKE és BETWEEN használata

LIKE használata:

- '_' egy tetszőleges karakterre illeszkedik

LIKE és BETWEEN használata

LIKE használata:

- '_' egy tetszőleges karakterre illeszkedik
- '%' tetszőleges karakterláncra illeszkedik

LIKE és BETWEEN használata

LIKE használata:

- '_' egy tetszőleges karakterre illeszkedik
- '%' tetszőleges karakterláncra illeszkedik

BETWEEN használata: BETWEEN a AND b jelentése $a \leq . \leq b$

LIKE és BETWEEN használata

LIKE használata:

- '_' egy tetszőleges karakterre illeszkedik
- '%' tetszőleges karakterláncra illeszkedik

BETWEEN használata: BETWEEN a AND b jelentése $a \leq . \leq b$

Példa 3: A 150 és 200 közötti azonosítójú mozik közül azok, amelyek B-vel kezdődő nevű városban vannak, és a nevük hárombetűs.

LIKE és BETWEEN használata

LIKE használata:

- '_' egy tetszőleges karakterre illeszkedik
- '%' tetszőleges karakterláncra illeszkedik

BETWEEN használata: BETWEEN a AND b jelentése $a \leq . \leq b$

Példa 3: A 150 és 200 közötti azonosítójú mozik közül azok, amelyek B-vel kezdődő nevű városban vannak, és a nevük hárombetűs.

```
SELECT név FROM mozi WHERE moziID BETWEEN 150 AND 200 AND város LIKE 'B%' AND név LIKE ' _ _ _ '
```

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Unió (valamely eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> UNION <eredményreláció2>

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Unió (valamely eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> UNION <eredményreláció2>
- **Példa 4:** A pénteken vagy szombaton játszott filmek :

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Unió (valamely eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> UNION <eredményreláció2>
- **Példa 4:** A pénteken vagy szombaton játszott filmek :
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
UNION
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

(nem hatékony!)

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Unió (valamely eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> UNION <eredményreláció2>
- **Példa 4:** A pénteken vagy szombaton játszott filmek :
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
UNION
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

(nem hatékony!)

Metszet (mindkét eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> INTERSECT <eredményreláció2>

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Unió (valamely eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> UNION <eredményreláció2>
- **Példa 4:** A pénteken vagy szombaton játszott filmek :
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
UNION
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

(nem hatékony!)

Metszet (mindkét eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> INTERSECT <eredményreláció2>
- **Példa 5:** A pénteken és szombaton is játszott filmek:

Műveletek relációkkal

A részeredményül kapott relációkkal **(ha azok sémája lényegében azonos!)** halmazműveleteket (unió, metszet, különbség) végezhetünk.

Unió (valamely eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> UNION <eredményreláció2>
- **Példa 4:** A pénteken vagy szombaton játszott filmek :
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
UNION
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

(nem hatékony!)

Metszet (mindkét eredményrelációban szereplő sorok):

- **Szintaxis:** <eredményreláció1> INTERSECT <eredményreláció2>
- **Példa 5:** A pénteken és szombaton is játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
INTERSECT
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- Szintaxis: $\langle \text{eredményreláció1} \rangle \text{ MINUS } \langle \text{eredményreláció2} \rangle$

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- Szintaxis: $\langle \text{eredményreláció1} \rangle \text{ MINUS } \langle \text{eredményreláció2} \rangle$
- Példa 6: A pénteken igen, de szombaton nem játszott filmek:

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- Szintaxis: <eredményreláció1> MINUS <eredményreláció2>
- Példa 6: A pénteken igen, de szombaton nem játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
MINUS
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- **Szintaxis:** <eredményreláció1> MINUS <eredményreláció2>
- **Példa 6:** A pénteken igen, de szombaton nem játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
MINUS
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

A szabványban **MINUS** helyett **EXCEPT** szerepel, de a gyakorlatban a **MINUS** használatos.

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- **Szintaxis:** <eredményreláció1> MINUS <eredményreláció2>
- **Példa 6:** A pénteken igen, de szombaton nem játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
MINUS
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

A szabványban **MINUS** helyett **EXCEPT** szerepel, de a gyakorlatban a **MINUS** használatos.

Állítás. Az SQL relációsan teljes.

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- Szintaxis: <eredményreláció1> MINUS <eredményreláció2>
- Példa 6: A pénteken igen, de szombaton nem játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
MINUS
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

A szabványban MINUS helyett EXCEPT szerepel, de a gyakorlatban a MINUS használatos.

Állítás. Az SQL relációsan teljes.

Bizonyítás: Most láttuk az **uniót** és **különbséget**, a többi pedig már volt, de újra:

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- Szintaxis: <eredményreláció1> MINUS <eredményreláció2>
- Példa 6: A pénteken igen, de szombaton nem játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
MINUS
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

A szabványban MINUS helyett EXCEPT szerepel, de a gyakorlatban a MINUS használatos.

Állítás. Az SQL relációsan teljes.

Bizonyítás: Most láttuk az **uniót** és **különbséget**, a többi pedig már volt, de újra:

vetítés: $\pi_{A_{i_1}, A_{i_2}, \dots, A_{i_k}}(R)$ -nek megfelelő lekérdezés: **SELECT $A_{i_1}, A_{i_2}, \dots, A_{i_k}$ FROM R**

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- **Szintaxis:** <eredményreláció1> MINUS <eredményreláció2>
- **Példa 6:** A pénteken igen, de szombaton nem játszott filmek:
 (SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
 MINUS
 (SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

A szabványban **MINUS** helyett **EXCEPT** szerepel, de a gyakorlatban a **MINUS** használatos.

Állítás. Az SQL relációsan teljes.

Bizonyítás: Most láttuk az **uniót** és **különbséget**, a többi pedig már volt, de újra:

vetítés: $\pi_{A_{i_1}, A_{i_2}, \dots, A_{i_k}}(\mathbf{R})$ -nek megfelelő lekérdezés: **SELECT $A_{i_1}, A_{i_2}, \dots, A_{i_k}$ FROM R**

kiválasztás: $\sigma_F(\mathbf{R})$ -nek megfelel a

SELECT * FROM R WHERE F'

ahol F'az, ami F-ből jön átírással (\wedge, \vee, \neg helyett AND, OR, NOT)

Különbség (az első reláció azon sorai, melyek a másodikban nem szerepelnek):

- **Szintaxis:** <eredményreláció1> MINUS <eredményreláció2>
- **Példa 6:** A pénteken igen, de szombaton nem játszott filmek:
(SELECT cím FROM film, vetít WHERE vetít.nap = 'péntek' AND film.filmID = vetít.filmID)
MINUS
(SELECT cím FROM film, vetít WHERE vetít.nap = 'szombat' AND film.filmID = vetít.filmID)

A szabványban **MINUS** helyett **EXCEPT** szerepel, de a gyakorlatban a **MINUS** használatos.

Állítás. Az SQL relációsan teljes.

Bizonyítás: Most láttuk az **uniót** és **különbséget**, a többi pedig már volt, de újra:

vetítés: $\pi_{A_{i_1}, A_{i_2}, \dots, A_{i_k}}(\mathbf{R})$ -nek megfelelő lekérdezés: **SELECT $A_{i_1}, A_{i_2}, \dots, A_{i_k}$ FROM R**

kiválasztás: $\sigma_F(\mathbf{R})$ -nek megfelel a

SELECT * FROM R WHERE F'

ahol F'az, ami F-ből jön átírással (\wedge, \vee, \neg helyett AND, OR, NOT)

szorzat: **SELECT $R.A_1, R.A_2, \dots, R.A_k, S.B_1, \dots, S.B_l$ FROM R,S** ✓

Multihalmazok-halmazok

- Az SQL alapértelmezésben nem tünteti el a többszörös sorokat, kivétel: UNION, INTERSECT, EXCEPT, ennél a háromnál eltűnnek az ismétlődések

Multihalmazok-halmazok

- Az SQL alapértelmezésben nem tünteti el a többszörös sorokat, **kivétel:** UNION, INTERSECT, EXCEPT, ennél a háromnál eltűnnek az ismétlődések
- Ha el akarjuk tüntetni az ismétlődéseket:
SELECT DISTINCT

Multihalmazok-halmazok

- Az SQL alapértelmezésben nem tünteti el a többszörös sorokat, **kivétel:** UNION, INTERSECT, EXCEPT, ennél a háromnál eltűnnek az ismétlődések
- Ha el akarjuk tüntetni az ismétlődéseket:
SELECT DISTINCT
- Ha a halmazműveleteknél mégsem akarom eltüntetni az ismétlődéseket:
UNION ALL, EXCEPT ALL, INTERSECT ALL

Multihalmazok-halmazok

- Az SQL alapértelmezésben nem tünteti el a többszörös sorokat, **kivétel:** UNION, INTERSECT, EXCEPT, ennél a háromnál eltűnnek az ismétlődések
- Ha el akarjuk tüntetni az ismétlődéseket:
SELECT DISTINCT
- Ha a halmazműveleteknél mégsem akarom eltüntetni az ismétlődéseket:
UNION ALL, EXCEPT ALL, INTERSECT ALL
- Nem (mindig) éri meg közben is törekedni arra, hogy ne legyen ismétlődés, elég a végén, mert:

Multihalmazok-halmazok

- Az SQL alapértelmezésben nem tünteti el a többszörös sorokat, **kivétel:** UNION, INTERSECT, EXCEPT, ennél a háromnál eltűnnek az ismétlődések
- Ha el akarjuk tüntetni az ismétlődéseket:
SELECT DISTINCT
- Ha a halmazműveleteknél mégsem akarom eltüntetni az ismétlődéseket:
UNION ALL, EXCEPT ALL, INTERSECT ALL
- Nem (mindig) éri meg közben is törekedni arra, hogy ne legyen ismétlődés, elég a végén, mert:
Az ismétlődés kiküszöbölése sok munka, mert rendezni kell az egész relációt hozzá.

Aggregátumok

- Aggregátumok számolása: SUM, MIN, MAX, AVG, COUNT,...

Aggregátumok

- Aggregátumok számolása: SUM, MIN, MAX, AVG, COUNT,...
- Az, hogy COUNT hogyan kezeli a többszörös sorokat, az rendszerfüggő.
Ha biztosra akarunk menni: COUNT (DISTINCT <attribútum>), COUNT (ALL <attribútum>)

Aggregátumok

- Aggregátumok számolása: SUM, MIN, MAX, AVG, COUNT,...
- Az, hogy COUNT hogyan kezeli a többszörös sorokat, az rendszerfüggő.
Ha biztosra akarunk menni: COUNT (DISTINCT <attribútum>), COUNT (ALL <attribútum>)
- Lehetőségünk van bizonyos attribútumok értéke szerint csoportosítani az eredményt, és így aggregált sorokat képezni.

Aggregátumok

- Aggregátumok számolása: SUM, MIN, MAX, AVG, COUNT,...
- Az, hogy COUNT hogyan kezeli a többszörös sorokat, az rendszerfüggő.
Ha biztosra akarunk menni: COUNT (DISTINCT <attribútum>), COUNT (ALL <attribútum>)
- Lehetőségünk van bizonyos attribútumok értéke szerint csoportosítani az eredményt, és így aggregált sorokat képezni.

Erre az utóbbira példa a következő reláció:

MOZI	mozilD	név	város	székszám
	1	Corvin	Budapest	2500
	2	Elit	Sopron	300
	3	Sopron Plaza Megaflex	Sopron	2000
	4	Szindbád	Budapest	600
	5	Tabán	Budapest	200
	6	Uránia	Pécs	500

Aggregátumok

Csoportosítsunk a város attribútum szerint:

MOZI	mozilD	név	város	székszám
	1	Corvin	Budapest	2500
	4	Szindbád	Budapest	600
	5	Tabán	Budapest	200
	6	Uránia	Pécs	500
	2	Elit	Sopron	300
	3	Sopron Plaza Megaflex	Sopron	2000

Aggregátumok

Csoportosítsunk a város attribútum szerint:

MOZI	mozilD	név	város	székszám
	1	Corvin	Budapest	2500
	4	Szindbád	Budapest	600
	5	Tabán	Budapest	200
	6	Uránia	Pécs	500
	2	Elit	Sopron	300
	3	Sopron Plaza Megaflex	Sopron	2000

Képezzük minden városra a székszámok összegét:

MOZI	város	össz_székszám
	Budapest	3300
	Pécs	500
	Sopron	2300

Példa 7: Mindez SQL-ben

Aggregátumok

Csoportosítsunk a város attribútum szerint:

MOZI	mozilD	név	város	székszám
	1	Corvin	Budapest	2500
	4	Szindbád	Budapest	600
	5	Tabán	Budapest	200
	6	Uránia	Pécs	500
	2	Elit	Sopron	300
	3	Sopron Plaza Megaflex	Sopron	2000

Képezzük minden városra a székszámok összegét:

MOZI	város	össz_székszám
	Budapest	3300
	Pécs	500
	Sopron	2300

Példa 7: Mindez SQL-ben

`SELECT város, SUM(székszám) AS össz_székszám FROM mozi GROUP BY város`

Példa 8: Az egyes városok legkisebb és legnagyobb mozijának mérete

Példa 8: Az egyes városok legkisebb és legnagyobb mozijának mérete
`SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város`

Példa 8: Az egyes városok legkisebb és legnagyobb mozijának mérete
`SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város`

Példák, ahol nincs csoportosítás:

Példa 9: A létező legnagyobb és a legkisebb székszám

Példa 8: Az egyes városok legkisebb és legnagyobb mozijának mérete
`SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város`

Példák, ahol nincs csoportosítás:

Példa 9: A létező legnagyobb és a legkisebb székszám
`SELECT MIN(székszám), MAX(székszám) FROM mozi`

Példa 8: Az egyes városok legkisebb és legnagyobb mozijának mérete
`SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város`

Példák, ahol nincs csoportosítás:

Példa 9: A létező legnagyobb és a legkisebb székszám
`SELECT MIN(székszám), MAX(székszám) FROM mozi`

Példa 10: Az összes székszám

Példa 8: Az egyes városok legkisebb és legnagyobb mozijának mérete
`SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város`

Példák, ahol nincs csoportosítás:

Példa 9: A létező legnagyobb és a legkisebb székszám
`SELECT MIN(székszám), MAX(székszám) FROM mozi`

Példa 10: Az összes székszám
`SELECT SUM(székszám) FROM mozi`

Aggregátumok

- **Kiértékelés:** Vesszük a FROM utáni relációk direkt szorzatát (egy reláció szerepelhet többször is a szorzatban, ha sorváltozókat adtunk meg hozzá),

Aggregátumok

- **Kiértékelés:** Vesszük a FROM utáni relációk direkt szorzatát (egy reláció szerepelhet többször is a szorzatban, ha sorváltozókat adtunk meg hozzá), a WHERE feltételt teljesítő eseteket a GROUP BY szerint csoportosítjuk,

Aggregátumok

- **Kiértékelés:** Vesszük a FROM utáni relációk direkt szorzatát (egy reláció szerepelhet többször is a szorzatban, ha sorváltozókat adtunk meg hozzá), a WHERE feltételt teljesítő eseteket a GROUP BY szerint csoportosítjuk, majd kiszámoljuk minden csoportra az aggregátumot és kiírjuk.

Aggregátumok

- **Kiértékelés:** Vesszük a FROM utáni relációk direkt szorzatát (egy reláció szerepelhet többször is a szorzatban, ha sorváltozókat adtunk meg hozzá), a WHERE feltételt teljesítő eseteket a GROUP BY szerint csoportosítjuk, majd kiszámoljuk minden csoportra az aggregátumot és kiírjuk.
- Amennyiben aggregátumokat képzünk a GROUP BY segítségével, akkor csak azokra az attribútumokra hivatkozhatunk közvetlenül a SELECT-ben, ami szerint csoportosítottunk.

Aggregátumok

- **Kiértékelés:** Vesszük a FROM utáni relációk direkt szorzatát (egy reláció szerepelhet többször is a szorzatban, ha sorváltozókat adtunk meg hozzá), a WHERE feltételt teljesítő eseteket a GROUP BY szerint csoportosítjuk, majd kiszámoljuk minden csoportra az aggregátumot és kiírjuk.
- Amennyiben aggregátumokat képzünk a GROUP BY segítségével, akkor csak azokra az attribútumokra hivatkozhatunk közvetlenül a SELECT-ben, ami szerint csoportosítottunk. Ezen attribútumok értékei ugyanis egy aggregátumon belül jól meghatározottak. A többi attribútum az aggregátumon belül többféle értéket is felvehet. Ezért rájuk csak oszlopfüggvényeken (aggregátumokon) keresztül hivatkozhatunk.

Aggregátumok

- **Kiértékelés:** Vesszük a FROM utáni relációk direkt szorzatát (egy reláció szerepelhet többször is a szorzatban, ha sorváltozókat adtunk meg hozzá), a WHERE feltételt teljesítő eseteket a GROUP BY szerint csoportosítjuk, majd kiszámoljuk minden csoportra az aggregátumot és kiírjuk.
- Amennyiben aggregátumokat képzünk a GROUP BY segítségével, akkor csak azokra az attribútumokra hivatkozhatunk közvetlenül a SELECT-ben, ami szerint csoportosítottunk. Ezen attribútumok értékei ugyanis egy aggregátumon belül jól meghatározottak. A többi attribútum az aggregátumon belül többféle értéket is felvehet. Ezért rájuk csak oszlopfüggvényeken (aggregátumokon) keresztül hivatkozhatunk.
- Lehet több oszlop szerint is GROUP BY, ekkor azok a sorok lesznek egy csoportban, ahol mindegyik GROUP BY után felsorolt oszlop értéke megegyezik.

- Lehet GROUP BY aggregátum nélkül is

Példa 11:

```
SELECT város FROM mozi GROUP BY város
```

- Lehet GROUP BY aggregátum nélkül is

Példa 11:

SELECT város FROM mozi GROUP BY város

Kiírja az összes várost (pontosan egyszer), ahol van mozi.

- Lehet GROUP BY aggregátum nélkül is

Példa 11:

SELECT város FROM mozi GROUP BY város

Kiírja az összes várost (pontosan egyszer), ahol van mozi.

Ugyanaz, mint a

SELECT DISTINCT város FROM mozi

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város HAVING  
COUNT(név)>1
```

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város HAVING  
COUNT(név)>1
```

- a csoportra vonatkozó feltételt a **HAVING** kulcsszó vezeti be

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város HAVING  
COUNT(név)>1
```

- a csoportra vonatkozó feltételt a **HAVING** kulcsszó vezeti be
- olyan feltételt írunk ide, ami csoportra vonatkozik (különben **WHERE**-be írnánk)

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város HAVING  
COUNT(név)>1
```

- a csoportra vonatkozó feltételt a **HAVING** kulcsszó vezeti be
- olyan feltételt írunk ide, ami csoportra vonatkozik (különben **WHERE**-be írnánk)
- csak **GROUP BY**-jal együtt használható

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város HAVING  
COUNT(név)>1
```

- a csoportra vonatkozó feltételt a **HAVING** kulcsszó vezeti be
- olyan feltételt írunk ide, ami csoportra vonatkozik (különben **WHERE**-be írnánk)
- csak **GROUP BY**-jal együtt használható
- a kiértékelés során a csoportosítás után minden egyes csoportra megnézzük a feltételt és eldobjuk azokat a csoportokat, amikre a feltétel nem áll és a maradékkal dolgozunk tovább

Feltétel a csoportokra — HAVING

A csoportosítással együtt tehetünk feltételt a csoportokra. Ebben az esetben csak azokra a csoportokra számolódik ki az aggregátum, amik a feltételnek eleget tesznek.

Példa 12: Azokra a városokra számolunk csak legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám) FROM mozi GROUP BY város HAVING COUNT(név)>1
```

- a csoportra vonatkozó feltételt a **HAVING** kulcsszó vezeti be
- olyan feltételt írunk ide, ami csoportra vonatkozik (különbön **WHERE**-be írnánk)
- csak **GROUP BY**-jal együtt használható
- a kiértékelés során a csoportosítás után minden egyes csoportra megnézzük a feltételt és eldobjuk azokat a csoportokat, amikre a feltétel nem áll és a maradékkal dolgozunk tovább
- **HAVING** megkerülhető, mindent, amit lehet **HAVING**-gel, lehet máshogy is (majd lesz erről szó az alkérdéseknél)

A hat alapkulcsszó

- SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY

A hat alapkulcsszó

- SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY
- Ebben a sorrendben jönnek

A hat alapkulcsszó

- SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY
- Ebben a sorrendben jönnek
- SELECT és FROM kell, a többi opcionális

A hat alapkulcsszó

- SELECT, FROM, WHERE, GROUP BY, HAVING, ORDER BY
- Ebben a sorrendben jönnek
- SELECT és FROM kell, a többi opcionális
- HAVING csak GROUP BY-jal

Alkérdeések

- Az alkérdeés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.

Alkérdeések

- Az alkérdés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.
- Tipikusan **WHERE feltételében áll**, ezáltal sokkal összetettebb kiválasztási feltételek jönnek létre, mint a relációs algebrában.

Alkérdeések

- Az alkérdés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.
- Tipikusan **WHERE** feltételében áll, ezáltal sokkal összetettebb kiválasztási feltételek jönnek létre, mint a relációs algebrában.

Alkérdés FROM záradékban

Alkérdeések

- Az alkérdés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.
- Tipikusan **WHERE feltételében áll**, ezáltal sokkal összetettebb kiválasztási feltételek jönnek létre, mint a relációs algebrában.

Alkérdés FROM záradékban

A kiválasztáshoz használt relációk lehetnek alkérdés által származtatott relációk is.

Alkérdeések

- Az alkérdés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.
- Tipikusan **WHERE feltételében áll**, ezáltal sokkal összetettebb kiválasztási feltételek jönnek létre, mint a relációs algebrában.

Alkérdeés FROM záradékban

A kiválasztáshoz használt relációk lehetnek alkérdés által származtatott relációk is.

Példa 13: A filmek címe, rendezője és a rendező filmjeinek száma

Alkérdeések

- Az alkérdés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.
- Tipikusan **WHERE feltételében áll**, ezáltal sokkal összetettebb kiválasztási feltételek jönnek létre, mint a relációs algebrában.

Alkérdeés FROM záradékban

A kiválasztáshoz használt relációk lehetnek alkérdés által származtatott relációk is.

Példa 13: A filmek címe, rendezője és a rendező filmjeinek száma

```
SELECT f1.cím, f1.rendező, f2.filmszám FROM
```

```
film AS f1,
```

```
(SELECT rendező, COUNT(*) AS filmszám FROM film GROUP BY rendező) AS f2
```

```
WHERE f1.rendező = f2.rendező
```

Alkérdeések

- Az alkérdeés eredménye mindig egy reláció, szintaxisa pedig a lekérdezés szintaxisával azonos.
- Tipikusan **WHERE feltételében áll**, ezáltal sokkal összetettebb kiválasztási feltételek jönnek létre, mint a relációs algebrában.

Alkérdeés FROM záradékban

A kiválasztáshoz használt relációk lehetnek alkérdeés által származtatott relációk is.

Példa 13: A filmek címe, rendezője és a rendező filmjeinek száma

```
SELECT f1.cím, f1.rendező, f2.filmszám FROM
```

```
film AS f1,
```

```
(SELECT rendező, COUNT(*) AS filmszám FROM film GROUP BY rendező) AS f2
```

```
WHERE f1.rendező = f2.rendező
```

Vigyázat! Itt nem jött létre f2 nevű reláció, csak annyi történik, hogy az f2 nevű sorváltozó befutja az alkérdeés eredményéül kapott reláció sorait. Egyszer kiszámolódik az alkérdeés és ennek eredményét használjuk a továbbiakban.

Alkérés WHERE záradékban

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz.

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz. Ezeknek az attribútumok számában meg kell egyezniük az alkérés eredményének oszlopszámával.

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz. Ezeknek az attribútumok számában meg kell egyezniük az alkérés eredményének oszlopszámával.

- **Egyenlőség vizsgálata**

Csak akkor lehetséges, ha az alkérés egysoros relációt ír le (azaz az eredménye egyetlen érték vagy érték-vektor).

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz. Ezeknek az attribútumok számában meg kell egyezniük az alkérés eredményének oszlopszámával.

- **Egyenlőség vizsgálata**

Csak akkor lehetséges, ha az alkérés egysoros relációt ír le (azaz az eredménye egyetlen érték vagy érték-vektor).

Az egyenlőség fennáll, ha az adott attribútumok értékei megegyeznek az alkérés által adott reláció megfelelő attribútumainak értékével.

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz. Ezeknek az attribútumok számában meg kell egyezniük az alkérés eredményének oszlopszámával.

- **Egyenlőség vizsgálata**

Csak akkor lehetséges, ha az alkérés egysoros relációt ír le (azaz az eredménye egyetlen érték vagy érték-vektor).

Az egyenlőség fennáll, ha az adott attribútumok értékei megegyeznek az alkérés által adott reláció megfelelő attribútumainak értékével.

Szintaxis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) = (SELECT <attrib21>, ..., <attrib2n> FROM ...)
```

A nem egyenlőség vizsgálatára a <> használandó.

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz. Ezeknek az attribútumok számában meg kell egyezniük az alkérés eredményének oszlopszámával.

- **Egyenlőség vizsgálata**

Csak akkor lehetséges, ha az alkérés egysoros relációt ír le (azaz az eredménye egyetlen érték vagy érték-vektor).

Az egyenlőség fennáll, ha az adott attribútumok értékei megegyeznek az alkérés által adott reláció megfelelő attribútumainak értékével.

Szintaxis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) = (SELECT <attrib21>, ..., <attrib2n> FROM ...)
```

A nem egyenlőség vizsgálatára a <> használandó.

Példa 14: A legnagyobb mozik nevei

Alkérés WHERE záradékban

Az alkérés eredményét valamely attribútumok értékeivel hasonlítjuk össze a kiválasztáshoz. Ezeknek az attribútumok számában meg kell egyezniük az alkérés eredményének oszlopszámával.

- **Egyenlőség vizsgálata**

Csak akkor lehetséges, ha az alkérés egysoros relációt ír le (azaz az eredménye egyetlen érték vagy érték-vektor).

Az egyenlőség fennáll, ha az adott attribútumok értékei megegyeznek az alkérés által adott reláció megfelelő attribútumainak értékével.

Szintaxis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) = (SELECT <attrib21>, ..., <attrib2n> FROM ...)
```

A nem egyenlőség vizsgálatára a <> használandó.

Példa 14: A legnagyobb mozik nevei

```
SELECT név FROM mozi
```

```
WHERE mozi.székszám = (SELECT MAX(székszám) FROM mozi)
```

- **Tartalmazás vizsgálata**
Több sort adó alkérdésre is értelmezett.

- **Tartalmazás vizsgálata**

Több sort adó alkérdésre is értelmezett.

A tartalmazás fennáll, ha a vizsgált attribútumok értéke megegyezik az alkérdés eredményének valamely sorával.

- **Tartalmazás vizsgálata**

Több sort adó alkérdésre is értelmezett.

A tartalmazás fennáll, ha a vizsgált attribútumok értéke megegyezik az alkérdés eredményének valamely sorával.

Szintaxis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) IN (SELECT <attrib21>, ..., <attrib2n> FROM ...)
```

- **Tartalmazás vizsgálata**

Több sort adó alkérésre is értelmezett.

A tartalmazás fennáll, ha a vizsgált attribútumok értéke megegyezik az alkérés eredményének valamely sorával.

Szintaxis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) IN (SELECT <attrib21>, ..., <attrib2n> FROM ...)
```

A nem tartalmazás vizsgálatára a **NOT IN** használandó.

Példa 15: A nem vetített filmek címe és rendezője

- **Tartalmazás vizsgálata**

Több sort adó alkérdésre is értelmezett.

A tartalmazás fennáll, ha a vizsgált attribútumok értéke megegyezik az alkérdés eredményének valamely sorával.

Szintaxis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) IN (SELECT <attrib21>, ..., <attrib2n> FROM ...)
```

A nem tartalmazás vizsgálatára a **NOT IN** használandó.

Példa 15: A nem vetített filmek címe és rendezője

```
SELECT cím, rendező FROM film AS f1
```

```
WHERE f1.filmID NOT IN (SELECT v1.filmID FROM vetít AS v1)
```

Alkérdeések

- **Alkérdeés valamely vagy minden sorának vizsgálata**

Tipikusan több sort szolgáló alkérdeés esetén, valamilyen összehasonlító operátorral együtt használatosak az **ANY** és az **ALL** kulcsszavak.

Alkérdeések

- **Alkérdeés valamely vagy minden sorának vizsgálata**

Tipikusan több sort szolgáltató alkérdeés esetén, valamilyen összehasonlító operátorral együtt használatosak az **ANY** és az **ALL** kulcsszavak.

Az **ANY**-t (**ALL**-t) tartalmazó feltétel teljesül, ha a vizsgált attribútumok értéke és az alkérdeés valamely (minden) sorára az összehasonlító operátor igaz értéket ad.

Alkérdeések

- **Alkérdeés valamely vagy minden sorának vizsgálata**

Tipikusan több sort szolgáltató alkérdeés esetén, valamilyen összehasonlító operátorral együtt használatosak az **ANY** és az **ALL** kulcsszavak.

Az **ANY**-t (**ALL**-t) tartalmazó feltétel teljesül, ha a vizsgált attribútumok értéke és az alkérdeés valamely (minden) sorára az összehasonlító operátor igaz értéket ad.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) <op> [ ANY | ALL ] (SELECT <attrib21>, ..., <attrib2n>  
FROM ...)
```

Alkérdeések

- **Alkérdeés valamely vagy minden sorának vizsgálata**

Tipikusan több sort szolgáltató alkérdeés esetén, valamilyen összehasonlító operátorral együtt használatosak az **ANY** és az **ALL** kulcsszavak.

Az **ANY**-t (**ALL**-t) tartalmazó feltétel teljesül, ha a vizsgált attribútumok értéke és az alkérdeés valamely (minden) sorára az összehasonlító operátor igaz értéket ad.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) <op> [ ANY | ALL ] (SELECT <attrib21>, ..., <attrib2n>  
FROM ...)
```

A „semelyik”, illetve a „nem mind” leírására a **NOT ANY**, illetve a **NOT ALL** használatosak.

Alkérdeések

- **Alkérdeés valamely vagy minden sorának vizsgálata**

Tipikusan több sort szolgáltató alkérdeés esetén, valamilyen összehasonlító operátorral együtt használatosak az **ANY** és az **ALL** kulcsszavak.

Az **ANY**-t (**ALL**-t) tartalmazó feltétel teljesül, ha a vizsgált attribútumok értéke és az alkérdeés valamely (minden) sorára az összehasonlító operátor igaz értéket ad.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) <op> [ ANY | ALL ] (SELECT <attrib21>, ..., <attrib2n>  
FROM ...)
```

A „semelyik”, illetve a „nem mind” leírására a **NOT ANY**, illetve a **NOT ALL** használatosak.

Példa 16: Ismét a legnagyobb mozi(k)

Alkérdeések

- **Alkérdeés valamely vagy minden sorának vizsgálata**

Tipikusan több sort szolgáltató alkérdeés esetén, valamilyen összehasonlító operátorral együtt használatosak az **ANY** és az **ALL** kulcsszavak.

Az **ANY**-t (**ALL**-t) tartalmazó feltétel teljesül, ha a vizsgált attribútumok értéke és az alkérdeés valamely (minden) sorára az összehasonlító operátor igaz értéket ad.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE (<attrib11>, ..., <attrib1n>) <op> [ ANY | ALL ] (SELECT <attrib21>, ..., <attrib2n>  
FROM ...)
```

A „semelyik”, illetve a „nem mind” leírására a **NOT ANY**, illetve a **NOT ALL** használatosak.

Példa 16: Ismét a legnagyobb mozi(k)

```
SELECT m2.város, m2.név, m2.székszám FROM mozi AS m2
```

```
WHERE m2.székszám >= ALL (SELECT m1.székszám FROM mozi AS m1)
```

- **Alkérés ürességének vizsgálata**

Az **EXISTS** kulcsszóval megvizsgálhatjuk, hogy van-e egyáltalán sora az alkérés által leírt relációnak.

- **Alkérés ürességének vizsgálata**

Az **EXISTS** kulcsszóval megvizsgálhatjuk, hogy van-e egyáltalán sora az alkérés által leírt relációnak.

Az ezt tartalmazó feltétel teljesül, ha van legalább egy sor.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE EXISTS (SELECT <attrib1>, ..., <attribn> FROM ...)
```

- **Alkérés ürességének vizsgálata**

Az **EXISTS** kulcsszóval megvizsgálhatjuk, hogy van-e egyáltalán sora az alkérés által leírt relációnak.

Az ezt tartalmazó feltétel teljesül, ha van legalább egy sor.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE EXISTS (SELECT <attrib1>, ..., <attribn> FROM ...)
```

A nem létezés vizsgálatára a **NOT EXISTS** használandó.

- **Alkérés ürességének vizsgálata**

Az **EXISTS** kulcsszóval megvizsgálhatjuk, hogy van-e egyáltalán sora az alkérés által leírt relációnak.

Az ezt tartalmazó feltétel teljesül, ha van legalább egy sor.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE EXISTS (SELECT <attrib1>, ..., <attribn> FROM ...)
```

A nem létezés vizsgálatára a **NOT EXISTS** használandó.

Példa 17: Azok a városok, ahol van legalább két mozi

- **Alkérés ürességének vizsgálata**

Az **EXISTS** kulcsszóval megvizsgálhatjuk, hogy van-e egyáltalán sora az alkérés által leírt relációnak.

Az ezt tartalmazó feltétel teljesül, ha van legalább egy sor.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE EXISTS (SELECT <attrib1>, ..., <attribn> FROM ...)
```

A nem létezés vizsgálatára a **NOT EXISTS** használandó.

Példa 17: Azok a városok, ahol van legalább két mozi

```
SELECT m1.város FROM mozi AS m1
```

```
WHERE EXISTS (SELECT * FROM mozi AS m2 WHERE m1.város =m2.város AND  
m1.név<>m2.név)
```

- **Alkérés ürességének vizsgálata**

Az **EXISTS** kulcsszóval megvizsgálhatjuk, hogy van-e egyáltalán sora az alkérés által leírt relációnak.

Az ezt tartalmazó feltétel teljesül, ha van legalább egy sor.

Szintakszis:

```
SELECT ... FROM ...
```

```
WHERE EXISTS (SELECT <attrib1>, ..., <attribn> FROM ...)
```

A nem létezés vizsgálatára a **NOT EXISTS** használandó.

Példa 17: Azok a városok, ahol van legalább két mozi

```
SELECT m1.város FROM mozi AS m1
```

```
WHERE EXISTS (SELECT * FROM mozi AS m2 WHERE m1.város =m2.város AND  
m1.név<>m2.név)
```

Ez úgy nevezett korrelált alkérés:

ennek kiértékelése során minden egyes lehetséges értékére az m1 sorváltozónak lefut az alkérés és kiírás van, ha az alkérés eredménye nem üres.

A korábbi esetekben csak egyszer kellett kiértékelni az alkérést, itt annyiszor, ahány sora a mozi-nak van.