

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`

2005

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 alkérdés 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.

Alkérdés FROM záradékban

A kiválasztáshoz használt relációk lehetnek alkérdés által származtatott relációk is.

Példa 1: 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 1: 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 1: 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 2: 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 2: 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 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 3: 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 3: 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 4: 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 4: 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 5: 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 5: 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 5: 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.

HAVING megkerülése alkérdéssel

Nézzük egy példán, de általában is így megy:

HAVING-gel:

Azokra a városokra számolunk legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám)
```

```
FROM mozi
```

```
GROUP BY város
```

```
HAVING COUNT(*)>1
```

HAVING megkerülése alkérdéssel

Nézzük egy példán, de általában is így megy:

HAVING-gel:

Azokra a városokra számolunk legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám)
```

```
FROM mozi
```

```
GROUP BY város
```

```
HAVING COUNT(*)>1
```

HAVING nélkül, alkérdéssel:

HAVING megkerülése alkérdéssel

Nézzük egy példán, de általában is így megy:

HAVING-gel:

Azokra a városokra számolunk legkisebb és legnagyobb mozit, ahol van legalább 2 mozi

```
SELECT város, MIN(székszám), MAX(székszám)
```

```
FROM mozi
```

```
GROUP BY város
```

```
HAVING COUNT(*)>1
```

HAVING nélkül, alkérdéssel:

```
SELECT város, minszékszám, maxszékszám
```

```
FROM (SELECT város, MIN(székszám) AS minszékszám, MAX(székszám)
```

```
AS maxszékszám, COUNT(*) AS darab
```

```
FROM mozi
```

```
GROUP BY város)
```

```
WHERE darab >1
```


NULL érték az SQL-ben

Az SQL-ben az ismeretlen vagy nem létező értéket a **NULL** érték jelképezi.

A **NULL** használatakor ügyelni kell rá, hogy az aritmetikai és összehasonlító operátorok speciálisan értelmezettek rá.

NULL érték az SQL-ben

Az SQL-ben az ismeretlen vagy nem létező értéket a **NULL** érték jelképezi.

A **NULL** használatakor ügyelni kell rá, hogy az aritmetikai és összehasonlító operátorok speciálisan értelmezettek rá.

Például:

NULL * 0 értéke nem 0, hanem **NULL**.

NULL érték az SQL-ben

Az SQL-ben az ismeretlen vagy nem létező értéket a **NULL** érték jelképezi.

A **NULL** használatakor ügyelni kell rá, hogy az aritmetikai és összehasonlító operátorok speciálisan értelmezettek rá.

Például:

NULL * 0 értéke nem 0, hanem **NULL**.

rendező = **NULL** értéke nem IGAZ, nem HAMIS, hanem a logikai ISMERETLEN érték
⇒ UN.

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

\vee	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>I</i>	<i>I</i>
<i>H</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>UN</i>	<i>I</i>	<i>UN</i>	<i>UN</i>

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

\vee	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>I</i>	<i>I</i>
<i>H</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>UN</i>	<i>I</i>	<i>UN</i>	<i>UN</i>

\wedge	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>H</i>	<i>H</i>	<i>H</i>	<i>H</i>
<i>UN</i>	<i>UN</i>	<i>H</i>	<i>UN</i>

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

\vee	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>I</i>	<i>I</i>
<i>H</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>UN</i>	<i>I</i>	<i>UN</i>	<i>UN</i>

\wedge	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>H</i>	<i>H</i>	<i>H</i>	<i>H</i>
<i>UN</i>	<i>UN</i>	<i>H</i>	<i>UN</i>

Tehát egy állítás nem csak igaz vagy hamis lehet, hanem „ismeretlen” is és egy WHERE-beli állításnál természetesen csak az számít találatnak, ha igaz az állítás, az „ismeretlen” nem lesz jó.

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

\vee	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>I</i>	<i>I</i>
<i>H</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>UN</i>	<i>I</i>	<i>UN</i>	<i>UN</i>

\wedge	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>H</i>	<i>H</i>	<i>H</i>	<i>H</i>
<i>UN</i>	<i>UN</i>	<i>H</i>	<i>UN</i>

Tehát egy állítás nem csak igaz vagy hamis lehet, hanem „ismeretlen” is és egy WHERE-beli állításnál természetesen csak az számít találatnak, ha igaz az állítás, az „ismeretlen” nem lesz jó.

Furán viselkedik ez a logika:

SELECT mozilD, filmID

FROM vetít

WHERE idő > "12:00" OR idő <= "12:00"

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

\vee	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>I</i>	<i>I</i>
<i>H</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>UN</i>	<i>I</i>	<i>UN</i>	<i>UN</i>

\wedge	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>H</i>	<i>H</i>	<i>H</i>	<i>H</i>
<i>UN</i>	<i>UN</i>	<i>H</i>	<i>UN</i>

Tehát egy állítás nem csak igaz vagy hamis lehet, hanem „ismeretlen” is és egy WHERE-beli állításnál természetesen csak az számít találatnak, ha igaz az állítás, az „ismeretlen” nem lesz jó.

Furán viselkedik ez a logika:

SELECT mozilD, filmID

FROM vetít

WHERE idő > "12:00" OR idő <= "12:00"

Az lenne jó, ha ez minden filmet felsorol, de sajnos aminek nincs ideje, azt nem sorolja fel.

Háromértékű logika

	\neg
<i>I</i>	<i>H</i>
<i>H</i>	<i>I</i>
<i>UN</i>	<i>UN</i>

\vee	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>I</i>	<i>I</i>
<i>H</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>UN</i>	<i>I</i>	<i>UN</i>	<i>UN</i>

\wedge	<i>I</i>	<i>H</i>	<i>UN</i>
<i>I</i>	<i>I</i>	<i>H</i>	<i>UN</i>
<i>H</i>	<i>H</i>	<i>H</i>	<i>H</i>
<i>UN</i>	<i>UN</i>	<i>H</i>	<i>UN</i>

Tehát egy állítás nem csak igaz vagy hamis lehet, hanem „ismeretlen” is és egy WHERE-beli állításnál természetesen csak az számít találatnak, ha igaz az állítás, az „ismeretlen” nem lesz jó.

Furán viselkedik ez a logika:

```
SELECT mozID, filmID
FROM vetít
WHERE idő > "12:00" OR idő <= "12:00"
```

Az lenne jó, ha ez minden filmet felsorol, de sajnos aminek nincs ideje, azt nem sorolja fel.

$\Rightarrow A \vee \neg A \neq I$ a háromértékű logikában, azaz nem teljesül az, amit megszoktunk, hogy vagy az állítás vagy a tagadása igaz lesz.

Háromértékű logika

Hasonlóan: egy mező értéke nem hasonlítható össze a szokásos módon a **NULL** értékkel (mivel **NULL** nem egy konstans).

Háromértékű logika

Hasonlóan: egy mező értéke nem hasonlítható össze a szokásos módon a **NULL** értékkel (mivel **NULL** nem egy konstans).

Erre az **IS NULL**, illetve az **IS NOT NULL** használatosak.

Háromértékű logika

Hasonlóan: egy mező értéke nem hasonlítható össze a szokásos módon a **NULL** értékkel (mivel **NULL** nem egy konstans).

Erre az **IS NULL**, illetve az **IS NOT NULL** használatosak.

Példa 6: Azon filmek címe és rendezője, melyeknek ismerjük a rendezőjét.

```
SELECT cím, rendező FROM film WHERE rendező IS NOT NULL
```

Relációk összekapcsolása (join) SQL2-ben

A következőkben ismertetett nyelvi elemek egy része csak szintaktikai édesítőszer, kifejezhető a

SELECT <attribútumok> FROM R, S WHERE <feltételek> (*)

segítségével.

Relációk összekapcsolása (join) SQL2-ben

A következőkben ismertetett nyelvi elemek egy része csak szintaktikai édesítőszer, kifejezhető a

```
SELECT <attribútumok> FROM R, S WHERE <feltételek> (*)
```

segítségével.

Relációk összekapcsolásakor meg kell adni az összekapcsolás módját (belső vagy külső) és a sorok összekapcsolásának feltételét.

Az összekapcsolás módja

Relációk összekapcsolása (join) SQL2-ben

A következőkben ismertetett nyelvi elemek egy része csak szintaktikai édesítőszer, kifejezhető a

SELECT <attribútumok> FROM R, S WHERE <feltételek> (*)

segítségével.

Relációk összekapcsolásakor meg kell adni az **összekapcsolás módját (belső vagy külső)** és a **sorok összekapcsolásának feltételét**.

Az összekapcsolás módja

- **Belső összekapcsolás** (mint \bowtie -nél): **R INNER JOIN S**
R-nek és S-nek csak azon sorai kerülnek az eredményrelációba, melyekhez van kapcsolódó sor S-ben, illetve R-ben (azaz a másik relációban). (Az, hogy mikor kapcsolódó két sor, az majd a kapcsolódás feltételeinél derül ki.)

Relációk összekapcsolása (join) SQL2-ben

A következőkben ismertetett nyelvi elemek egy része csak szintaktikai édesítőszer, kifejezhető a

SELECT <attribútumok> FROM R, S WHERE <feltételek> (*)

segítségével.

Relációk összekapcsolásakor meg kell adni az **összekapcsolás módját (belső vagy külső)** és a **sorok összekapcsolásának feltételét**.

Az összekapcsolás módja

- **Belső összekapcsolás** (mint \bowtie -nél): **R INNER JOIN S**
R-nek és S-nek csak azon sorai kerülnek az eredményrelációba, melyekhez van kapcsolódó sor S-ben, illetve R-ben (azaz a másik relációban). (Az, hogy mikor kapcsolódó két sor, az majd a kapcsolódás feltételeinél derül ki.)
- **Bal oldali külső összekapcsolás** (mint \ltimes -nél): **R LEFT [OUTER] JOIN S**
R-nek azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik S-beli sor.
Ezekben a csak S-ben szereplő mezők **NULL** értéket kapnak.

Relációk összekapcsolása (join) SQL2-ben

A következőkben ismertetett nyelvi elemek egy része csak szintaktikai édesítőszer, kifejezhető a

SELECT <attribútumok> FROM R, S WHERE <feltételek> (*)

segítségével.

Relációk összekapcsolásakor meg kell adni az **összekapcsolás módját (belső vagy külső)** és a **sorok összekapcsolásának feltételét**.

Az összekapcsolás módja

- **Belső összekapcsolás** (mint \bowtie -nél): **R INNER JOIN S**
R-nek és S-nek csak azon sorai kerülnek az eredményrelációba, melyekhez van kapcsolódó sor S-ben, illetve R-ben (azaz a másik relációban). (Az, hogy mikor kapcsolódó két sor, az majd a kapcsolódás feltételeinél derül ki.)
- **Bal oldali külső összekapcsolás** (mint \ltimes -nél): **R LEFT [OUTER] JOIN S**
R-nek azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik S-beli sor.
Ezekben a csak S-ben szereplő mezők **NULL** értéket kapnak.
- **Jobb oldali külső összekapcsolás** (mint \rightharpoonup -nél): **R RIGHT [OUTER] JOIN S**
Mint a LEFT OUTER JOIN, de R és S szerepe megcserélődik.

- **Teljes külső összekapcsolás** (mint \bowtie -nél): **R FULL [OUTER] JOIN S**
Mind R, mind S azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik sor a másik relációból.
Az ezáltal üresen maradó mezők itt is **NULL** értéket kapnak.

- **Teljes külső összekapcsolás** (mint \bowtie -nél): **R FULL [OUTER] JOIN S**
Mind R, mind S azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik sor a másik relációból.
Az ezáltal üresen maradó mezők itt is **NULL** értéket kapnak.

A direkt szorzat létrehozására az **R CROSS JOIN S** alak használható, ilyenkor nincs feltétele a sorok összekapcsolásának.

Ez az alapértelmezés, (*) használatkor ilyen illesztés történik.

- **Teljes külső összekapcsolás** (mint \bowtie -nél): **R FULL [OUTER] JOIN S**
Mind R, mind S azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik sor a másik relációból.
Az ezáltal üresen maradó mezők itt is **NULL** értéket kapnak.

A direkt szorzat létrehozására az **R CROSS JOIN S** alak használható, ilyenkor nincs feltétele a sorok összekapcsolásának.

Ez az alapértelmezés, (*) használatkor ilyen illesztés történik.

A sorok összekapcsolásának feltételei

- **Természetes illesztés: R NATURAL [INNER | LEFT | RIGHT | FULL] JOIN S**
R és S azon sorai illesztődnek, ahol az azonos nevű attribútumok értéke is megegyezik.
Ez az alapértelmezés.

- **Teljes külső összekapcsolás** (mint \bowtie -nél): **R FULL [OUTER] JOIN S**
Mind R, mind S azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik sor a másik relációból.
Az ezáltal üresen maradó mezők itt is **NULL** értéket kapnak.

A direkt szorzat létrehozására az **R CROSS JOIN S** alak használható, ilyenkor nincs feltétele a sorok összekapcsolásának.

Ez az alapértelmezés, (*) használatakor ilyen illesztés történik.

A sorok összekapcsolásának feltételei

- **Természetes illesztés: R NATURAL [INNER | LEFT | RIGHT | FULL] JOIN S**
R és S azon sorai illesztődnek, ahol az azonos nevű attribútumok értéke is megegyezik.
Ez az alapértelmezés.
- **Illesztés azonos nevű attribútumokkal: R [INNER | LEFT | RIGHT | FULL] JOIN S USING (<attribútumok>)**
R és S azon sorai illesztődnek, ahol az azonos nevű és <attribútumok>-ban felsorolt attribútumok értéke is megegyezik.

- **Teljes külső összekapcsolás** (mint \bowtie -nél): **R FULL [OUTER] JOIN S**
Mind R, mind S azon sorai is bekerülnek az eredményrelációba, melyekhez nem kapcsolódik sor a másik relációból.
Az ezáltal üresen maradó mezők itt is **NULL** értéket kapnak.

A direkt szorzat létrehozására az **R CROSS JOIN S** alak használható, ilyenkor nincs feltétele a sorok összekapcsolásának.

Ez az alapértelmezés, (*) használatakor ilyen illesztés történik.

A sorok összekapcsolásának feltételei

- **Természetes illesztés:** **R NATURAL [INNER | LEFT | RIGHT | FULL] JOIN S**
R és S azon sorai illesztődnek, ahol az azonos nevű attribútumok értéke is megegyezik.
Ez az alapértelmezés.
- **Illesztés azonos nevű attribútumokkal:** **R [INNER | LEFT | RIGHT | FULL] JOIN S USING (<attribútumok>)**
R és S azon sorai illesztődnek, ahol az azonos nevű és <attribútumok>-ban felsorolt attribútumok értéke is megegyezik.
- **Illesztés tetszőleges feltétellel (θ -join):** **R [INNER | LEFT | RIGHT | FULL] JOIN S ON (<feltétel>)**
R és S azon sorai illesztődnek, melyek attribútumai eleget tesznek a megadott feltételnek.

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példa 10: ugyanez USING használatával

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példa 10: ugyanez USING használatával

```
SELECT cím, rendező, nap, idő FROM film INNER JOIN vetít USING (filmID)
```

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példa 10: ugyanez USING használatával

```
SELECT cím, rendező, nap, idő FROM film INNER JOIN vetít USING (filmID)
```

Példa 11: Összes film címe, rendezője és vetítési időpontja (ha van)

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példa 10: ugyanez USING használatával

```
SELECT cím, rendező, nap, idő FROM film INNER JOIN vetít USING (filmID)
```

Példa 11: Összes film címe, rendezője és vetítési időpontja (ha van)

```
SELECT cím, rendező, nap, idő FROM film NATURAL LEFT OUTER JOIN vetít
```

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példa 10: ugyanez USING használatával

```
SELECT cím, rendező, nap, idő FROM film INNER JOIN vetít USING (filmID)
```

Példa 11: Összes film címe, rendezője és vetítési időpontja (ha van)

```
SELECT cím, rendező, nap, idő FROM film NATURAL LEFT OUTER JOIN vetít
```

Példa 12: Az összes film-mozi pár

Példák relációk összekapcsolására

Példa 7: Kusturica vetített filmjei és vetítési időpontjaik

```
SELECT cím, nap, idő FROM film INNER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 8: Kusturica összes filmje és vetítési időpontjaik (ha van)

```
SELECT cím, nap, idő FROM film LEFT OUTER JOIN vetít ON rendező='E. Kusturica' AND  
film.filmID=vetít.filmID
```

Példa 9: Vetített filmek címe, rendezője és vetítési időpontjaik

```
SELECT cím, rendező, nap, idő FROM film NATURAL INNER JOIN vetít
```

Példa 10: ugyanez USING használatával

```
SELECT cím, rendező, nap, idő FROM film INNER JOIN vetít USING (filmID)
```

Példa 11: Összes film címe, rendezője és vetítési időpontja (ha van)

```
SELECT cím, rendező, nap, idő FROM film NATURAL LEFT OUTER JOIN vetít
```

Példa 12: Az összes film-mozi pár

```
SELECT * FROM film CROSS JOIN mozi
```

DML utasítások — INSERT

Sorokat a relációba az **INSERT** utasítással szúrhatunk be.

Szintakszis: **INSERT INTO** <reláció> (<attrib₁>, ..., <attrib_n>) **VALUES** (<érték₁>, ..., <érték_n>)

DML utasítások — INSERT

Sorokat a relációba az **INSERT** utasítással szúrhatunk be.

Szintakszis: `INSERT INTO <reláció> (<attrib1>, ..., <attribn>) VALUES (<érték1>, ..., <értékn>)`

Hatása: a <reláció> relációba egy új sor kerül, amiben <attrib₁> attribútum értéke <érték₁>, stb. A nem meghatározott értékű attribútumok a reláció létrehozásakor az attribútumhoz rendelt alapértelmezett értéket veszik fel.

DML utasítások — INSERT

Sorokat a relációba az **INSERT** utasítással szúrhatunk be.

Szintakszis: `INSERT INTO <reláció> (<attrib1>, ..., <attribn>) VALUES (<érték1>, ..., <értékn>)`

Hatása: a <reláció> relációba egy új sor kerül, amiben <attrib₁> attribútum értéke <érték₁>, stb. A nem meghatározott értékű attribútumok a reláció létrehozásakor az attribútumhoz rendelt alapértelmezett értéket veszik fel.

Példa 13: Egy új film felvétele

```
INSERT INTO film (cím, rendező) VALUES ('Egy csodálatos elme', 'Ron Howard')
```

DML utasítások — INSERT

Sorokat a relációba az **INSERT** utasítással szúrhatunk be.

Szintakszis: `INSERT INTO <reláció> (<attrib1>, ..., <attribn>) VALUES (<érték1>, ..., <értékn>)`

Hatása: a <reláció> relációba egy új sor kerül, amiben <attrib₁> attribútum értéke <érték₁>, stb. A nem meghatározott értékű attribútumok a reláció létrehozásakor az attribútumhoz rendelt alapértelmezett értéket veszik fel.

Példa 13: Egy új film felvétele

```
INSERT INTO film (cím, rendező) VALUES ('Egy csodálatos elme', 'Ron Howard')
```

Megjegyzés:

- a filmID mező az alapértelmezett értékét kapja, de egy trigger (később lesz) segítségével akár automatikusan növekvő számozást is létrehozhatunk.

DML utasítások — INSERT

Sorokat a relációba az **INSERT** utasítással szúrhatunk be.

Szintakszis: `INSERT INTO <reláció> (<attrib1>, ..., <attribn>) VALUES (<érték1>, ..., <értékn>)`

Hatása: a <reláció> relációba egy új sor kerül, amiben <attrib₁> attribútum értéke <érték₁>, stb. A nem meghatározott értékű attribútumok a reláció létrehozásakor az attribútumhoz rendelt alapértelmezett értéket veszik fel.

Példa 13: Egy új film felvétele

```
INSERT INTO film (cím, rendező) VALUES ('Egy csodálatos elme', 'Ron Howard')
```

Megjegyzés:

- a filmID mező az alapértelmezett értékét kapja, de egy trigger (később lesz) segítségével akár automatikusan növekvő számozást is létrehozhatunk.
- Ha az összes attribútum értékét megadjuk, akkor nem kell őket felsorolni, ebben az esetben a beadott értékek a default attribútumsorrend szerint lesznek hozzárendelve az attribútumokhoz)

- a beszúrt adatokat egy alkérdésből is vehetjük:

Ha van egy filmregi(filmID, cím, rendező) tábla és azokat az adatokat szeretnénk átvinni a film táblába, amik ott nem szerepelnek:

- a beszúrt adatokat egy alkérdésből is vehetjük:

Ha van egy filmregi(filmID, cím, rendező) tábla és azokat az adatokat szeretnénk átvinni a film táblába, amik ott nem szerepelnek:

```
INSERT INTO film
```

```
    SELECT filmregi.filmID, filmregi.cím, filmregi.rendező
```

```
    FROM filmregi
```

```
    WHERE filmregi.filmID NOT IN
```

```
        (SELECT filmID FROM film)
```

DML utasítások — UPDATE

Sorokat a relációban az **UPDATE** utasítással módosíthatunk.

Szintakszis: UPDATE <reláció> SET <attrib₁>=<érték₁>, ..., <attrib_n>=<érték_n> WHERE <feltétel>

DML utasítások — UPDATE

Sorokat a relációban az **UPDATE** utasítással módosíthatunk.

Szintakszis: UPDATE <reláció> SET <attrib₁>=<érték₁>, ..., <attrib_n>=<érték_n> WHERE <feltétel>

Hatása: a <reláció> reláció minden sorában, amelyik illeszkedik a <feltétel> feltételre <attrib_i> értéke <érték_i> lesz.

DML utasítások — UPDATE

Sorokat a relációban az **UPDATE** utasítással módosíthatunk.

Szintakszis: UPDATE <reláció> SET <attrib₁>=<érték₁>, ..., <attrib_n>=<érték_n> WHERE <feltétel>

Hatása: a <reláció> reláció minden sorában, amelyik illeszkedik a <feltétel> feltételre <attrib_i> értéke <érték_i> lesz.

Példa 14: Az előbb beszúrt rendező nevének átírása rövidített alakba

DML utasítások — UPDATE

Sorokat a relációban az **UPDATE** utasítással módosíthatunk.

Szintakszis: UPDATE <reláció> SET <attrib₁>=<érték₁>, ..., <attrib_n>=<érték_n> WHERE <feltétel>

Hatása: a <reláció> reláció minden sorában, amelyik illeszkedik a <feltétel> feltételre <attrib_i> értéke <érték_i> lesz.

Példa 14: Az előbb beszúrt rendező nevének átírása rövidített alakba
UPDATE film SET rendező='R. Howard' WHERE rendező='Ron Howard'

DML utasítások — DELETE

Sorokat egy relációból a **DELETE** utasítással törölhetünk.

Szintakszis: **DELETE FROM** <reláció> **WHERE** <feltétel>

DML utasítások — DELETE

Sorokat egy relációból a **DELETE** utasítással törölhetünk.

Szintakszis: **DELETE FROM** <reláció> **WHERE** <feltétel>

Hatása: a <reláció> reláció feltételre illeszkedő sorait törli.

DML utasítások — DELETE

Sorokat egy relációból a **DELETE** utasítással törölhetünk.

Szintakszis: **DELETE FROM** <reláció> **WHERE** <feltétel>

Hatása: a <reláció> reláció feltételre illeszkedő sorait törli.

Megjegyzés: a **WHERE** <feltétel> rész elhagyása esetén a reláció összes sorát törli.

DML utasítások — DELETE

Sorokat egy relációból a **DELETE** utasítással törölhetünk.

Szintakszis: **DELETE FROM** <reláció> **WHERE** <feltétel>

Hatása: a <reláció> reláció feltételre illeszkedő sorait törli.

Megjegyzés: a **WHERE** <feltétel> rész elhagyása esetén a reláció összes sorát törli.

Példa 15: Azon filmek törlése, amiknek a rendezője E. K. monogrammú
DELETE FROM film **WHERE** rendező **LIKE** 'E.% K%'

SQL Data Definition Language

- Séma létrehozása
- Séma törlése
- Séma módosítása

SQL Data Definition Language

- Séma létrehozása
- Séma törlése
- Séma módosítása
- Indexek létrehozása és kezelése (az indexekről később lesz szó részletesen)

SQL Data Definition Language

- Séma létrehozása
- Séma törlése
- Séma módosítása
- Indexek létrehozása és kezelése (az indexekről később lesz szó részletesen)
- Nézetek létrehozása

SQL Data Definition Language

- Séma létrehozása
- Séma törlése
- Séma módosítása
- Indexek létrehozása és kezelése (az indexekről később lesz szó részletesen)
- Nézetek létrehozása
- Kényszerek létrehozása

SQL Data Definition Language

- Séma létrehozása
- Séma törlése
- Séma módosítása
- Indexek létrehozása és kezelése (az indexekről később lesz szó részletesen)
- Nézetek létrehozása
- Kényszerek létrehozása
- Triggerek (kicsit)

A triggerek már az SQL3-hoz tartoznak, a triggerek segítségével az adatbázis valamely változásakor egy tárolt eljárást hajthatunk végre. Itt fogunk beszélni a rekurzióról is, mert az is SQL3-as dolog, de az lekérdezés és nem DDL, a segítségével egy reláció tranzitív lezárását lehet kiszámolni rekurzívan.

Relációk létrehozása

Relációk létrehozására a **CREATE TABLE** használandó.

Minden attribútumnak típust és a reláción belül egyedi nevet kell adni.

Relációk létrehozása

Relációk létrehozására a **CREATE TABLE** használandó.

Minden attribútumnak típust és a reláción belül egyedi nevet kell adni.

Szükség esetén megadható alapértelmezett érték is (**DEFAULT** kulcsszóval), melyeket az attribútum azon sorokban vesz fel, ahol a beszúrásakor nem adtuk meg a konkrét értékét.

Relációk létrehozása

Relációk létrehozására a **CREATE TABLE** használandó.

Minden attribútumnak típust és a reláción belül egyedi nevet kell adni.

Szükség esetén megadható alapértelmezett érték is (**DEFAULT** kulcsszóval), melyeket az attribútum azon sorokban vesz fel, ahol a beszúrásakor nem adtuk meg a konkrét értékét.

Lehetőség van arra is, hogy kényszereket definiáljunk az attribútumokra (pl. attribútum nem **NULL**, elsődleges kulcs, idegen kulcs, stb.), ezekről később lesz szó.

Relációk létrehozása

Relációk létrehozására a **CREATE TABLE** használandó.

Minden attribútumnak típust és a reláción belül egyedi nevet kell adni.

Szükség esetén megadható alapértelmezett érték is (**DEFAULT** kulcsszóval), melyeket az attribútum azon sorokban vesz fel, ahol a beszúrásakor nem adtuk meg a konkrét értékét.

Lehetőség van arra is, hogy kényszereket definiáljunk az attribútumokra (pl. attribútum nem **NULL**, elsődleges kulcs, idegen kulcs, stb.), ezekről később lesz szó.

Szintakszis:

```
CREATE TABLE <relációnév> (  
<attrib1> <adattípus1> [DEFAULT <érték>], ...  
<attribn> <adattípusn> [DEFAULT <érték>]  
)
```

Példa 16: Film reláció létrehozása

```
CREATE TABLE film(  
  filmID number(5),  
  cím varchar(50),  
  rendező char(30),  
  év number(4),  
  hossz number(3) DEFAULT 90)
```

Példa 16: Film reláció létrehozása

```
CREATE TABLE film(  
  filmID number(5),  
  cím varchar(50),  
  rendező char(30),  
  év number(4),  
  hossz number(3) DEFAULT 90)
```

A lehetséges adattípusok függenek az adatbáziskezelőtől.

Példa 16: Film reláció létrehozása

```
CREATE TABLE film(  
  filmID number(5),  
  cím varchar(50),  
  rendező char(30),  
  év number(4),  
  hossz number(3) DEFAULT 90)
```

A lehetséges adattípusok függenek az adatbáziskezelőtől.

A főbb típusok:

char(<i>n</i>)	<i>n</i> hosszú karaktersorozat
varchar(<i>n</i>)	maximum <i>n</i> hosszú karaktersorozat
number(<i>n</i>,<i>m</i>)	<i>n</i> hosszú, <i>m</i> tizedesjegyű szám
date	dátum

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(
DROP TABLE film

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(

```
DROP TABLE film
```

Relációk módosítására az **ALTER TABLE** használandó.

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(
DROP TABLE film

Relációk módosítására az **ALTER TABLE** használandó.

Lehetőség van új attribútum definiálására (**ADD**), attribútum törlésére (**DROP**), attribútum adattípusának módosítására (**MODIFY**), kényszerek módosítására, alapértelmezett érték megadására.

Mindezek csak a jelenlegi adatokkal konzisztensen végezhetők el.
Például nem törölhető olyan attribútum, amire még van hivatkozás.

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(
DROP TABLE film

Relációk módosítására az **ALTER TABLE** használandó.

Lehetőség van új attribútum definiálására (**ADD**), attribútum törlésére (**DROP**), attribútum adattípusának módosítására (**MODIFY**), kényszerek módosítására, alapértelmezett érték megadására.

Mindezek csak a jelenlegi adatokkal konzisztensen végezhetők el.
Például nem törölhető olyan attribútum, amire még van hivatkozás.

Szintaxis: **ALTER TABLE** <relációnév> <opciók>

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(
DROP TABLE film

Relációk módosítására az **ALTER TABLE** használandó.

Lehetőség van új attribútum definiálására (**ADD**), attribútum törlésére (**DROP**), attribútum adattípusának módosítására (**MODIFY**), kényszerek módosítására, alapértelmezett érték megadására.

Mindezek csak a jelenlegi adatokkal konzisztensen végezhetők el.
Például nem törölhető olyan attribútum, amire még van hivatkozás.

Szintaxis: **ALTER TABLE** <relációnév> <opciók>

Példa 18: Vetít relációhoz helyár hozzáadása
ALTER TABLE vetít **ADD** helyár number(4)

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(
DROP TABLE film

Relációk módosítására az **ALTER TABLE** használandó.

Lehetőség van új attribútum definiálására (**ADD**), attribútum törlésére (**DROP**), attribútum adattípusának módosítására (**MODIFY**), kényszerek módosítására, alapértelmezett érték megadására.

Mindezek csak a jelenlegi adatokkal konzisztensen végezhetők el.

Például nem törölhető olyan attribútum, amire még van hivatkozás.

Szintaxis: **ALTER TABLE** <relációnév> <opciók>

Példa 18: Vetít relációhoz helyár hozzáadása
ALTER TABLE vetít **ADD** helyár number(4)

Példa 19: Mozi relációból a város eltávolítása
ALTER TABLE mozi **DROP** város

Relációk törlése, módosítása

Relációk törlésére a **DROP TABLE** használandó.

Szintaxis: **DROP TABLE** <relációnév>

Példa 17: Film reláció törlése :-(
DROP TABLE film

Relációk módosítására az **ALTER TABLE** használandó.

Lehetőség van új attribútum definiálására (**ADD**), attribútum törlésére (**DROP**), attribútum adattípusának módosítására (**MODIFY**), kényszerek módosítására, alapértelmezett érték megadására.

Mindezek csak a jelenlegi adatokkal konzisztensen végezhetők el.
Például nem törölhető olyan attribútum, amire még van hivatkozás.

Szintaxis: **ALTER TABLE** <relációnév> <opciók>

Példa 18: Vetít relációhoz helyár hozzáadása
ALTER TABLE vetít **ADD** helyár number(4)

Példa 19: Mozi relációból a város eltávolítása
ALTER TABLE mozi **DROP** város

Példa 20: Ha a helyárakat ezentúl dollárban számoljuk ...
ALTER TABLE vetít **MODIFY** helyár number(4,2)

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: `CREATE INDEX <indexnév> ON <relációnév>(attribútumok listája)`

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: `CREATE INDEX <indexnév> ON <relációnév>(attribútumok listája)`

Példa 21: index a filmcím, rendező párra

```
CREATE INDEX cím-rend ON film(cím, rendező)
```

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: **CREATE INDEX** <indexnév> **ON** <relációnév>(attribútumok listája)

Példa 21: index a filmcím, rendező párra

```
CREATE INDEX cím-rend ON film(cím, rendező)
```

Ha meguntuk, el lehet dobni: **DROP INDEX** cím-rend

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: `CREATE INDEX <indexnév> ON <relációnév>(attribútumok listája)`

Példa 21: index a filmcím, rendező párra

```
CREATE INDEX cím-rend ON film(cím, rendező)
```

Ha meguntuk, el lehet dobni: `DROP INDEX cím-rend`

- **előny:** gyors keresés lehetséges az index segítségével

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: `CREATE INDEX <indexnév> ON <relációnév>(attribútumok listája)`

Példa 21: index a filmcím, rendező párra

```
CREATE INDEX cím-rend ON film(cím, rendező)
```

Ha meguntuk, el lehet dobni: `DROP INDEX cím-rend`

- **előny:** gyors keresés lehetséges az index segítségével
- **hátrány:** az adatszerkezetet karban kell tartani, így lassítja a beszúrást, törlést, módosítást

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: **CREATE INDEX** <indexnév> **ON** <relációnév>(attribútumok listája)

Példa 21: index a filmcím, rendező párra

```
CREATE INDEX cím-rend ON film(cím, rendező)
```

Ha meguntuk, el lehet dobni: **DROP INDEX** cím-rend

- **előny:** gyors keresés lehetséges az index segítségével
- **hátrány:** az adatszerkezetet karban kell tartani, így lassítja a beszúrást, törlést, módosítást
- az a fontos, hogy miből van több, módosításból vagy lekérdezésből?

Indexek

Egy reláció bizonyos attribútumaira indexet készíthetünk: ez egy adatszerkezetet jelent, ami olyan sorok gyors keresését teszi lehetővé amelyek az adott attribútumokon valami adott értékeket vesznek fel.

SQL2-nek sem része, de gyakori, ezért tanuljuk.

Szintaxis: **CREATE INDEX** <indexnév> **ON** <relációnév>(attribútumok listája)

Példa 21: index a filmcím, rendező párra

```
CREATE INDEX cím-rend ON film(cím, rendező)
```

Ha meguntuk, el lehet dobni: **DROP INDEX** cím-rend

- **előny:** gyors keresés lehetséges az index segítségével
- **hátrány:** az adatszerkezetet karban kell tartani, így lassítja a beszúrást, törlést, módosítást
- az a fontos, hogy miből van több, módosításból vagy lekérdezésből?
- néha a rendszer magától létrehoz indexet (lásd kulcsok)