

2.előadás: Adatbázisok-I.

dr. Hajas Csilla (ELTE IK)

<http://sila.hajas.elte.hu/>

Relációs algebra egy táblára vonatkozó műveletei és az SQL SELECT utasítás

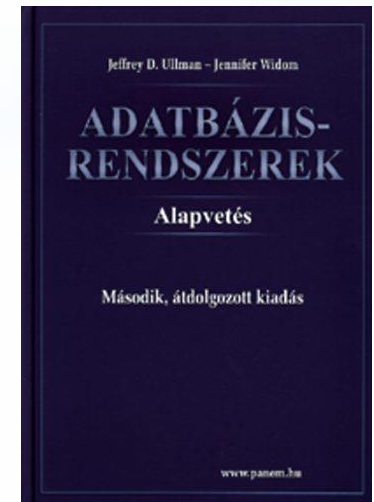
Tankönyv:

2.4. Relációs algebra unér műveletei

5.1-5.2. Kiterjesztése multihalmazokra

6.1. Egy táblás lekérdezések SQL-ben

Kiegészítés: Leckék Oracle gyakorlatra



Egy táblára vonatkozó lekérdezések

- Emlékeztető: **Egy reláció (vagy tábla) két részből áll:**
 - (1) **relációs sémából** (sortípus és megszorítások)
 - (2) **reláció előfordulásából** (véges sok sor halmaza)
- **Egy táblára vonatkozó műveletek** relációs algebrában:
Mi lesz az eredmény tábla sémája és előfordulása?
(Itt alap relációs algebrában halmazként, SQL-ben és kiterjesztett relációs algebrában majd multihalmazként)

Vetítés

1. táblán

Kiválasztás

1. táblán

Vetítés (project, jelölése pí: Π)

- **Vetítés (projekció).** Adott relációt vetít le az alsó indexben szereplő attribútumokra (attribútumok számát csökkentik)
- $\Pi_{\text{lista}}(R)$ ahol lista: $\{A_{i_1}, \dots, A_{i_k}\}$ R-sémájában levő attribútumok egy részhalmazának felsorolása
eredmény típusa $\langle A_{i_1} : \text{értéktípus}_{i_1}, \dots, A_{i_k} : \text{értéktípus}_{i_k} \rangle$
 $\Pi_{\text{lista}}(R) := \{ t.A_{i_1}, t.A_{i_2}, \dots, t.A_{i_k} \mid t \in R \} = \{ t[\text{lista}] \mid t \in R \}$
- Reláció soraiból kiválasztja az attribútumoknak megfelelő A_{i_1}, \dots, A_{i_k} -n előforduló értékeket, ha többször előfordul akkor a duplikátumokat kiszűrjük (hogy halmazt kapjunk)

➤ **Példa:**

A	B	C
a	b	c
c	d	e
c	d	d

$\Pi_{A, B}(R)$



A	B
a	b
c	d

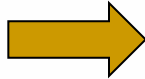
Kiválasztás (select, jelölése szigma: σ)

- **Kiválasztás (szűrés).** Kiválasztja az argumentumban szereplő reláció azon sorait, amelyek eleget tesznek az alsó indexben szereplő feltételnek.
- $\sigma_{\text{Feltétel}}(R)$ és R sémája megegyezik
- $\sigma_{\text{Feltétel}}(R) := \{ t \mid t \in R \text{ és } t \text{ kielégíti } \sigma \text{-ban szereplő Feltételt} \}$
- $R(A_1, \dots, A_n)$ séma feletti reláció esetén a σ_F kiválasztás F feltétele a következőképpen épül fel:
 - **elemi feltétel:** $A_i \theta A_j$, $A_i \theta c$, ahol c konstans, θ pedig $=, \neq, <, >, \leq, \geq$
 - **összetett feltétel:** ha B_1, B_2 feltételek, akkor $\neg B_1, B_1 \wedge B_2, B_1 \vee B_2$ és zárójelezésekkel is feltételek

➤ **Példa:**

A	B	C
a	b	c
c	d	e
g	a	d

$\sigma_{A='a' \vee C \leq 'd'}(R)$



A	B	C
a	b	c
g	a	d

Lekérdezések az SQL-ben

- 1.) Az SQL-ben halmazok helyett **multihalmazokat** használunk (vagyis egy sor többször is előfordulhat)
- 2.) Lekérdezésekben $\Pi_{\text{select-lista}} \sigma_{\text{where-feltétel}}(\text{from-tábla})$ a select-listán és where-feltételben az attribútumnevek helyén olyan **kifejezések** állhatnak az SQL-ben, amely függvényeket és műveleti jeleket is tartalmazhat
- Az attribútumnevek helyén álló kifejezésekben használt **legfontosabb sorfüggvények**:
 - Numerikus, karakteres, dátum, konverziós függvények
 - NULL hiányzó értéket megadott értékkel helyettesítő függvények, például NVL, COALESCE használata, lásd részletesen a kiegészítő leckékben (EA végén).

A kiterjesztett relációs algebra

- Az eddig tanult egy táblára vonatkozó műveleteket: **vetítés** (Π_{lista}) és **kiválasztás** (σ_{felt}) műveletét kiterjesztjük **multihalmazokra**, ahogyan az **SQL-ben**, egy reláció nem sorok halmazából, hanem sorok **multihalmazából** áll, vagyis megengedett a sorok ismétlődése.
- Ezekon kívül a **SELECT kiegészítéseinek és záradékainak** megfeleltetett **új műveletekkel** is kibővítjük a rel. algebrát:
 - **Ismétlődések megszüntetése** (δ) - **select distinct**
 - **Vetítési művelet kiterjesztése** (Π_{lista}) - **select kif [as onev]**
 - **Rendezési művelet** (τ_{lista}) - **order by**
 - **Következő héten folytatjuk:** kifejezések, sorfüggvények, összesítő függvények, összesítő művelet, csoportosítás **GROUP BY** és **HAVING záradékok** ($\sigma_{csop.felt}$, $\gamma_{csop.attr.lista}$)

A relációs algebrai unér műveletek értelmezése multihalmazok fölött

- A projekció és szelekció végrehajtása során nem küszöböljük ki az ismétlődéseket.

R			$\Pi_A(R)$
A	B		A
1	2	➔	1
1	5		1
2	3		2

Új műveletek a kiterjesztett algebrában: Ismétlődések megszüntetése: **DISTINCT**

- **Ismétlődések megszüntetése**: $R1 := \delta(R2)$
- A művelet jelentése: $R2$ multihalmazból $R1$ halmazt állít elő, vagyis az $R2$ -ben egyszer vagy többször előforduló sorok csak egyszer szerepelnek az $R1$ -ben.
- A **DISTINCT** reprezentálására szolgál (jele: δ kis-delta)
- A δ speciális esete lesz az általánosabb γ műveletnek

$$R = \left(\begin{array}{|c|c|} \hline A & B \\ \hline 1 & 2 \\ 3 & 4 \\ 1 & 2 \\ \hline \end{array} \right)$$
$$\delta(R) = \begin{array}{|c|c|} \hline A & B \\ \hline 1 & 2 \\ 3 & 4 \\ \hline \end{array}$$

SQL SELECT lekérdezések

Példa – Sörivók adatbázisséma

- Az előadások SQL lekérdezései az alábbi Sörivók adatbázissémán alapulnak
(aláhúzás jelöli a kulcs attribútumokat)

Sörök(név, gyártó)

Sörözők(név, város, tulaj, engedély)

Sörivók(név, város, tel)

Szeret(név, sör)

Felhasználó(söröző, sör, ár)

Látogat(név, söröző)

Egyszerű példa Select-From-Where-re

- Használjuk Sörök(név, gyártó) illetve Felszolgál(söröző, sör, ár) relációsémát
- Mely sörök olcsóbbak 2.00-nél?

```
SELECT sör  
FROM Felszolgál  
WHERE ár < 2.00; /* numerikus */
```

- Mely söröket gyártja a Dreher?

```
SELECT név  
FROM Sörök  
WHERE gyártó = 'Dreher'; /* char */
```

A lekérdezés eredménye

név
Arany Ászok
Dreher Classic
...

A lekérdezés eredménye egy reláció, amelynek egy attribútuma van (név) és a sorai az összes olyan sör neve, amelyet a Dreher gyárt.

Eltérés a relációs algebrától: Az SQL alapértelmezésben nem szűri ki a duplikátumokat, az eredmény multihalmaz.

A műveletek szemantikája

név	gyártó
Arany Ászok	Dreher

1) Ellenőrizzük a feltételt, hogy a gyártó Dreher-e

2) Ha a feltétel teljesült, akkor képezünk egy t eredménysort

3) Ebből a t sorból a SELECT listának megfelelő típusú sort képezzük, példa: t.név

Az egytáblás SFW alapértelmezése

```
SELECT [DISTINCT] kif1 [[AS] onév1], ... , kifn [onévn]  
FROM táblanév [sorváltozó]  
[WHERE feltétel]
```

Alapértelmezés (a műveletek szemantikája -- általában)

- A FROM záradékban levő relációhoz tekintünk egy **sorváltozót**, amely a reláció minden sorát bejárja
- Minden egyes „aktuális” sorhoz **kiértékeljük** a WHERE záradékot
- Ha helyes (vagyis **igaz**) választ kaptunk, akkor képezünk egy sort a SELECT záradékban szereplő kifejezéseknek megfelelően.

SELECT záradékban * jelentése

- Amikor csak egy reláció van a FROM záradékban, akkor a SELECT záradékban levő * jelentése: „a reláció minden attribútuma”
- **Példa:** Keressük a Sörök(név, gyártó) tábla alapján a Dreher-sörök adatait.
- A lekérdezés eredménye

```
SELECT *
```

```
FROM Sörök
```

```
WHERE gyártó = 'Dreher' ;
```

- A lekérdezés eredménye a Sörök tábla összes attribútumát tartalmazza. Első lépésben (kezdő gyakorlásnál kicsi táblákra) mindig lekérdezzük előbb a tábla tartalmát: `SELECT * FROM Táblanév;`

Attribútumok átnevezése

- Ha az eredményben (a fejlécben) más attribútumnevet szeretnénk használni, akkor “[AS] új_oszlopnév” segítségével tudunk más oszlopnévet kiírni.
(Oracle: másodnévben nem kell ‘AS’, csak szóköz)
- Listán azt értjük, hogy vesszővel vannak elválasztva az elemek (attribútumnevek), ha a másodnévben szóköz szerepel, akkor azt macskaköröm közé kell tenni: ” ... ”
- **Példa:** Sörök(név, gyártó)

```
SELECT név sör, gyártó "Dreher gyártó"  
FROM Sörök  
WHERE gyártó = 'Dreher';
```
- A lekérdezés eredményében az új oszlopnévek lesznek.

SELECT záradékban levő kifejezések

- SELECT utasításban attribútumnév helyett kifejezéseket, függvényeket is lehet használni (és attól függően, hogy az attribútumnak mi a típusa, numerikus, karakteres vagy dátum a megfelelő kifejezéseket, függvényeket).

- **Példa:** Felszolgál(söröző, sör, ár)

```
SELECT söröző, sör, ár*114 árYenben  
FROM Felszolgál;
```

- Konstansok a kifejezésekben Szeret(név, sör):

```
SELECT név DABkedvelő  
FROM Szeret  
WHERE upper(sör) = 'DAB';
```


WHERE záradék (összetett feltételek)

- Hasonlóan, mint a relációs algebra kiválasztás (σ) feltételében **elemi feltételekből építkezünk**, ahol elemi feltételen két kifejezés =, <>, <, >, <=, >= aritmetikai összehasonlítását, a theta műveletet értjük.
- **Logikai műveletek** AND, OR, NOT és zárójel () segítségével kapjuk **az összetett feltételeket**.
- **Példa: Felszolgál (söröző, sör, ár)** relációséma esetén keressük a „Joe's Bar”-ban árult „DAB” sörök árát:

```
SELECT ár
FROM Felszolgál
WHERE söröző = 'Joe' 's Bar' AND
       sör = 'DAB' ;
```

WHERE záradék (további lehetőségek)

SQL specialitások, amelyek könnyen átírhatóak relációs algebrai kifejezésre (összetett kiválasztási feltételre)

- **BETWEEN .. AND ..** intervallumba tartozás
- **IN (értékhalmoz)** egyszerű értékek halmaza

SQL specialitások, nem írhatók át relációs algebraiba:

(--- ezek jönnek a köv.lapon...)

- Karakterláncok **LIKE** összehasonlítása mintákkal
- **IS NULL** összehasonlítás

LIKE

- **Karakterláncok összehasonlítása mintákkal:**
 - <attribútum> LIKE <minta> vagy
 - <attribútum> NOT LIKE <minta>
- **Minta** egy olyan karakterlánc, amelyben használhatjuk a speciális % és _ karaktereket. A mintában % megfelel bármilyen karakterláncnak és _ bármilyen karakternek.
- **Példa:** Azokat a sörözőket keressük, amelyek nevének a második betűje „a” vagy a nevében van „s”, mint ahogyan például a „*Joe's Bar*” névben is szerepel:

```
SELECT név FROM Sörözők
WHERE név LIKE '_a%' OR
       név LIKE '%s%';
```

NULL (hiányzó) értékek

- Az SQL lehetővé teszi, hogy a relációk soraiban az attribútum értéke egy speciális NULL nullérték legyen.
- **A nullérték értelmezésére** több lehetőségünk is van:
 - **Hiányzó érték:** például tudom, „Joe's Bár”-jának van valamilyen címe, de nem tudom, hogy mi az.
 - **Nem-definiált érték:** például a házastárs attribútumnak egyedülálló embereknél nincs olyan értéke, aminek itt értelme lenne, nincs házastársa, ezért nullérték.
- **Where záradékban a nullérték vizsgálata:**
 - IS NULL
 - IS NOT NULL

NULL értékek használata

- **Where záradékban a nullérték** használata:
 - Amikor egy aritmetikai műveletben az egyik tag **NULL**, akkor az eredmény is **NULL**.
 - Amikor egy **NULL** értéket hasonlítunk össze bármely más értékkel (beleértve a NULL-t is) az összehasonlítási operátorok (=, <>, <, <=, >, >=) segítségével, akkor az eredmény **UNKNOWN** (ismeretlen).

Az ismeretlen (unknown) igazságérték

- Az SQL-ben szereplő logikai feltételek valójában **háromértékű logika**: TRUE, FALSE, UNKNOWN (magyarban igaz, hamis, ismeretlen rövidítése miatt inkább meghagyjuk az angol T, F, U rövidítéseket).
- A WHERE záradékban szereplő logikai feltételt a rendszer minden egyes sorra ellenőrzi és a logikai érték TRUE, FALSE vagy UNKNOWN valamelyike lehet, de az eredménybe csak azok a sorok kerülnek, amelyeknek a feltétel kiértékelése TRUE értéket adott.

A háromértékű logika

- **Hogyan működnek** az AND, OR, és NOT logikai műveletek a 3-értékű logikában?
- A szabályt könnyű megjegyezni, ha úgy tekintjük, hogy TRUE = 1, FALSE = 0, és UNKNOWN = $\frac{1}{2}$.
- Ekkor AND = MIN, OR = MAX, NOT(x) = 1-x.
- **Példa:**
TRUE AND (FALSE OR NOT(UNKNOWN)) =
MIN(1, MAX(0, (1 - $\frac{1}{2}$))) =
MIN(1, MAX(0, $\frac{1}{2}$)) =
MIN(1, $\frac{1}{2}$) = $\frac{1}{2}$ = UNKNOWN
- A 3-értékű logika AND, OR és NOT igazságtáblázatát lásd a **Tk. 6.2.ábráját** (vagy kitöltése a fenti szabállyal)

A háromértékű logika (Tk.6.2. ábra)

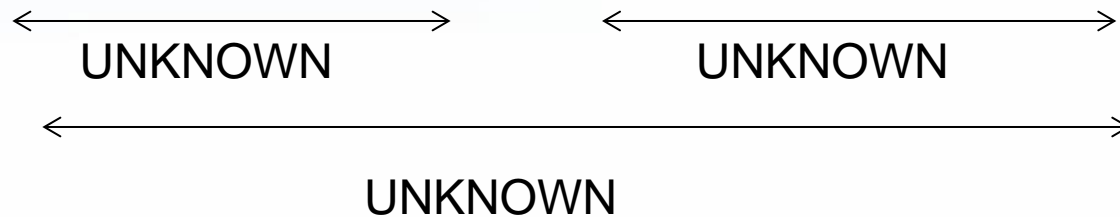
x	y	x AND y	x OR y	NOT x
T	T	T	T	F
T	U	U	T	F
T	F	F	T	F
U	T	U	T	U
U	U	U	U	U
U	F	F	U	U
F	T	F	T	T
F	U	F	U	T
F	F	F	F	T

Egy meglepő példa

- Példa: Felszolgál reláció legyen az alábbi:

söröző	sör	ár
Joe's Bar	Bud	NULL

```
SELECT söröző  
FROM Felszolgál  
WHERE ár < 2.00 OR ár >= 2.00;
```



Oka: a 2-értékű \neq 3-értékű szabályok

- Bizonyos általános szabályok, mint például, hogy az AND kommutatív érvényes a 3-értékű logikában is.
- Ellenben nem igaz, például a **kizáró szabály**, vagyis $p \text{ OR } \text{NOT } p = \text{TRUE}$ nem teljesül, ha $p = \text{UNKNOWN}$, mert ekkor a baloldal: $\text{MAX}(\frac{1}{2}, (1 - \frac{1}{2})) = \frac{1}{2} \neq 1$ vagyis a 3-értékű logikában baloldal értéke nem TRUE.
- Ezért az előző példában nem az eredeti egy soros táblát, hanem az üres táblát (amelynek egy sora sincs) kaptuk meg az eredménytáblaként.

Az eredmény rendezése

- SQL SELECT utasításban a záradékok
- Az SQL lehetővé teszi, hogy a lekérdezés eredménye bizonyos sorrendben legyen rendezve. Az első attribútum egyenlősége esetén a 2.attribútum szerint rendezve, stb, minden attribútumra lehet növekvő vagy csökkenő sorrend.
- Select-From-Where utasításhoz a következő záradékot adjuk, a WHERE záradék és minden más záradék (mint például GROUP BY és HAVING) után következik:

SELECT ... FROM ... [WHERE ...][...]

ORDER BY {attribútum [DESC], ...}

- **Példa: SELECT * FROM Felszolgal
ORDER BY ár DESC, sör**

Kérdés / Válasz

- **Köszönöm a figyelmet! Kérdés/Válasz?**
- **Összefoglalva:** SQL lekérdezések megalapozása
- Az egy táblára vonatkozó lekérdezések, az alap relációs algebra vetítés a kiválasztás műveletei
- SELECT utasítás SELECT és WHERE záradékai
- **Kiegészítés:** Leckék Oracle gyakorlatra
 - [./sql/lecke01_select_alap.pdf](#) (2.gyak)
 - [./sql/lecke02_where_feltetel.pdf](#) (2.gyak)
 - [./sql/lecke03_fuggvenyek.pdf](#) (folyt.köv.héten)
- **Gyakorlatra:** Oracle Példatár 1.fej. feladatsora