

Adatbázisok elmélete 8. 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

Példák oszlopkalkulus alkalmazására

ÁRU(ÁRUKÓD, ÁRUNÉV, EGYSÉGÁR)

MENNYISÉG(DÁTUM, ÁRUKÓD, DB)

BEVÉTEL(DÁTUM, ÖSSZEG)

BEFIZ(ÖSSZEG, BEFIZ) **BEFIZ=ÖSSZEG-4000**

Példák oszlopkalkulus alkalmazására

ÁRU(ÁRUKÓD, ÁRUNÉV, EGYSÉGÁR)

MENNYISÉG(DÁTUM, ÁRUKÓD, DB)

BEVÉTEL(DÁTUM, ÖSSZEG)

BEFIZ(ÖSSZEG, BEFIZ) **BEFIZ=ÖSSZEG-4000**

Hány darabot adtak el 2004. jan. 15-én az A123 kódú áruból, mi a neve és az ára?

$$\left\{ s^{(3)} \mid \exists u \exists v \left(\text{MENNYISÉG}(u) \wedge \text{ÁRU}(v) \wedge u[1] = 2004-01-15 \wedge u[2] = 'A123' \wedge \right. \right. \\ \left. \left. v[1] = 'A123' \wedge s[1] = u[3] \wedge s[2] = v[2] \wedge s[3] = v[3] \right) \right\}$$

Példák oszlopkalkulus alkalmazására

ÁRU(ÁRUKÓD, ÁRUNÉV, EGYSÉGÁR)

MENNYISÉG(DÁTUM, ÁRUKÓD, DB)

BEVÉTEL(DÁTUM, ÖSSZEG)

BEFIZ(ÖSSZEG, BEFIZ) BEFIZ=ÖSSZEG-4000

Hány darabot adtak el 2004. jan. 15-én az A123 kódú áruból, mi a neve és az ára?

$$\{s^{(3)} \mid \exists u \exists v (\text{MENNYISÉG}(u) \wedge \text{ÁRU}(v) \wedge u[1] = 2004-01-15 \wedge u[2] = 'A123' \wedge \\ v[1] = 'A123' \wedge s[1] = u[3] \wedge s[2] = v[2] \wedge s[3] = v[3])\}$$

$$\{x, y, z \mid \exists u \exists v (\text{MENNYISÉG}(u, v, x) \wedge \text{ÁRU}(v, y, z) \wedge u = 2004-01-15 \wedge v = 'A123')\}$$

Példák oszlopkalkulus alkalmazására

ÁRU(ÁRUKÓD, ÁRUNÉV, EGYSÉGÁR)

MENNYISÉG(DÁTUM, ÁRUKÓD, DB)

BEVÉTEL(DÁTUM, ÖSSZEG)

BEFIZ(ÖSSZEG, BEFIZ) BEFIZ=ÖSSZEG-4000

Hány darabot adtak el 2004. jan. 15-én az A123 kódú áruból, mi a neve és az ára?

$$\{s^{(3)} \mid \exists u \exists v (\text{MENNYISÉG}(u) \wedge \text{ÁRU}(v) \wedge u[1] = 2004-01-15 \wedge u[2] = 'A123' \wedge \\ v[1] = 'A123' \wedge s[1] = u[3] \wedge s[2] = v[2] \wedge s[3] = v[3])\}$$

$$\{x, y, z \mid \exists u \exists v (\text{MENNYISÉG}(u, v, x) \wedge \text{ÁRU}(v, y, z) \wedge u = 2004-01-15 \wedge v = 'A123')\}$$

Mely nevű áruk azok, amelyekkel van azonos egységárú másik áru?

$$\{s^{(1)} \mid \exists u \exists v (\text{ÁRU}(v) \wedge \text{ÁRU}(u) \wedge s[1] = v[2] \wedge v[3] = u[3] \wedge \neg(v[1] = u[1]))\}$$

Példák oszlopkalkulus alkalmazására

ÁRU(ÁRUKÓD, ÁRUNÉV, EGYSÉGÁR)

MENNYISÉG(DÁTUM, ÁRUKÓD, DB)

BEVÉTEL(DÁTUM, ÖSSZEG)

BEFIZ(ÖSSZEG, BEFIZ) **BEFIZ=ÖSSZEG-4000**

Hány darabot adtak el 2004. jan. 15-én az A123 kódú áruból, mi a neve és az ára?

$$\{s^{(3)} \mid \exists u \exists v (\text{MENNYISÉG}(u) \wedge \text{ÁRU}(v) \wedge u[1] = 2004-01-15 \wedge u[2] = 'A123' \wedge \\ v[1] = 'A123' \wedge s[1] = u[3] \wedge s[2] = v[2] \wedge s[3] = v[3])\}$$

$$\{x, y, z \mid \exists u \exists v (\text{MENNYISÉG}(u, v, x) \wedge \text{ÁRU}(v, y, z) \wedge u = 2004-01-15 \wedge v = 'A123')\}$$

Mely nevű áruk azok, amelyekkel van azonos egységárú másik áru?

$$\{s^{(1)} \mid \exists u \exists v (\text{ÁRU}(v) \wedge \text{ÁRU}(u) \wedge s[1] = v[2] \wedge v[3] = u[3] \wedge \neg(v[1] = u[1]))\}$$

$$\{v \mid \exists x \exists y \exists w \exists u (\text{ÁRU}(x, v, u) \wedge \text{ÁRU}(y, w, u) \wedge x \neq y)\}$$

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopalkulus ekvivalensek.*

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopkalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopkalkulus ekvivalensek.*

Bizonyítás: *Vázlat:*

$$\{t^{(k)} \mid \phi(t^{(k)})\} \longleftrightarrow \{u_1, \dots, u_k \mid \psi(u_1, \dots, u_k)\}$$

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopalkulus ekvivalensek.*

Bizonyítás: *Vázlat:*

$$\begin{aligned} \{t^{(k)} \mid \phi(t^{(k)})\} &\longleftrightarrow \{u_1, \dots, u_k \mid \psi(u_1, \dots, u_k)\} \\ t^{(k)} &\longleftrightarrow u_1, \dots, u_k \end{aligned}$$

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopkalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopkalkulus ekvivalensek.*

Bizonyítás: *Vázlat:*

$$\{t^{(k)} \mid \phi(t^{(k)})\} \longleftrightarrow \{u_1, \dots, u_k \mid \psi(u_1, \dots, u_k)\}$$

$$t^{(k)} \longleftrightarrow u_1, \dots, u_k$$

$$R(t^{(k)}) \longleftrightarrow R(u_1, \dots, u_k)$$

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopkalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopkalkulus ekvivalensek.*

Bizonyítás: Vázlat:

$$\{t^{(k)} \mid \phi(t^{(k)})\} \longleftrightarrow \{u_1, \dots, u_k \mid \psi(u_1, \dots, u_k)\}$$

$$t^{(k)} \longleftrightarrow u_1, \dots, u_k$$

$$R(t^{(k)}) \longleftrightarrow R(u_1, \dots, u_k)$$

$$t^{(k)}[j] \longleftrightarrow u_j$$

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopkalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopkalkulus ekvivalensek.*

Bizonyítás: Vázlat:

$$\{t^{(k)} \mid \phi(t^{(k)})\} \longleftrightarrow \{u_1, \dots, u_k \mid \psi(u_1, \dots, u_k)\}$$

$$t^{(k)} \longleftrightarrow u_1, \dots, u_k$$

$$R(t^{(k)}) \longleftrightarrow R(u_1, \dots, u_k)$$

$$t^{(k)}[j] \longleftrightarrow u_j$$

$$\exists t^{(k)} \phi(t^{(k)}) \longleftrightarrow \exists u_1 \dots \exists u_k \psi(u_1, \dots, u_k)$$

Biztonságos kifejezés

$\text{Dom}(\psi)$ és a biztonságos formula és kifejezés ugyanaz, mint sorkalkulusnál.

Ugyanolyan technikák vannak a biztonságosság elérésére, mint sorkalkulusnál.

Tétel. *A sorkalkulus és az oszlopkalkulus ekvivalensek. A biztonságos sorkalkulus és a biztonságos oszlopkalkulus ekvivalensek.*

Bizonyítás: Vázlat:

$$\{t^{(k)} \mid \phi(t^{(k)})\} \longleftrightarrow \{u_1, \dots, u_k \mid \psi(u_1, \dots, u_k)\}$$

$$t^{(k)} \longleftrightarrow u_1, \dots, u_k$$

$$R(t^{(k)}) \longleftrightarrow R(u_1, \dots, u_k)$$

$$t^{(k)}[j] \longleftrightarrow u_j$$

$$\exists t^{(k)} \phi(t^{(k)}) \longleftrightarrow \exists u_1 \dots \exists u_k \psi(u_1, \dots, u_k)$$

$$\text{biztonságos} \longleftrightarrow \text{biztonságos}$$

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)
- aritmetika

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)
- aritmetika
- aggregátumok

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)
- aritmetika
- aggregátumok
- többszörös sorok kezelése/megengedése

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)
- aritmetika
- aggregátumok
- többszörös sorok kezelése/megengedése
- attribútumok sorrendje kötött

Lekérdezőnyelvek típusai, általános jellemzőik

- Lehetnek **algebrai alapúak**: relációs algebrán alapuló lekérdezés, procedurális leírás
Előny: lekérdezésoptimalizálásra jobb
- Lehetnek **logikai alapúak**: sor vagy oszlopkalkulusra épülő lekérdezés, deklaratív leírás
Előny: könnyebben átlátható

Általában a konkrét lekérdezőnyelvek eltérnek a modelltől (algebrai és logikai esetben is), van amiben többet tudnak, van amiben kevesebbet, vagy csak máshogy.

Lehetséges eltérések:

- logikai alapúakban eleve csak biztonságos kifejezéseknek megfelelő kérdéseket lehet írni (nincsenek kvantorok)
- aritmetika
- aggregátumok
- többszörös sorok kezelése/megengedése
- attribútumok sorrendje kötött

De az igaz mindre, hogy relációsan teljesek (esetleg bizonyos műveletek nehezebben mennek) és általában van a lekérdező funkció mellett DML és DDL is.

Példák relációs adatbáziskezelő nyelvekre

- Information System Base Language \Rightarrow ISBL
 - ★ relációs algebra alapú
 - ★ kifejlesztő: IBM's United Kingdom Scientific Center
 - ★ Peterlee Relational Test Vehicle

Példák relációs adatbáziskezelő nyelvekre

- **Information System Base Language** \Rightarrow **ISBL**
 - ★ relációs algebra alapú
 - ★ **kifejlesztő:** IBM's United Kingdom Scientific Center
 - ★ **Peterlee Relational Test Vehicle**

- **QUERy Language** \Rightarrow **QUEL**
 - ★ sorkalkulus alapú
 - ★ **kifejlesztő:** University of California at Berkly
 - ★ **INGRES lekérdezőnyelve**

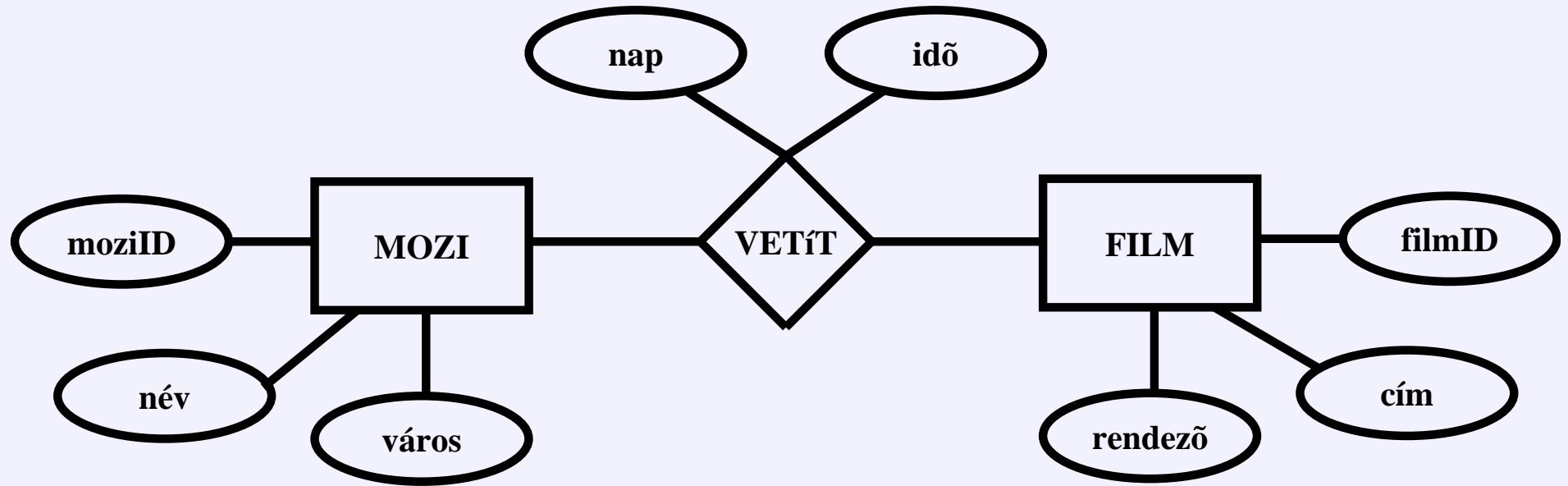
Példák relációs adatbáziskezelő nyelvekre

- **Information System Base Language** \Rightarrow **ISBL**
 - ★ relációs algebra alapú
 - ★ **kifejlesztő**: IBM's United Kingdom Scientific Center
 - ★ **Peterlee Relational Test Vehicle**
- **QUERy Language** \Rightarrow **QUEL**
 - ★ sorkalkulus alapú
 - ★ **kifejlesztő**: University of California at Berkly
 - ★ **INGRES lekérdezőnyelve**
- **Query-By-Example** \Rightarrow **QBE**
 - ★ oszlopkalkulus alapú
 - ★ **kifejlesztő**: IBM's Watson Research Center
 - ★ **pl. MS Access**

Példák relációs adatbáziskezelő nyelvekre

- **Information System Base Language** \Rightarrow **ISBL**
 - ★ relációs algebra alapú
 - ★ **kifejlesztő**: IBM's United Kingdom Scientific Center
 - ★ **Peterlee Relational Test Vehicle**
- **QUERy Language** \Rightarrow **QUEL**
 - ★ sorkalkulus alapú
 - ★ **kifejlesztő**: University of California at Berkly
 - ★ **INGRES lekérdezőnyelve**
- **Query-By-Example** \Rightarrow **QBE**
 - ★ oszlopkalkulus alapú
 - ★ **kifejlesztő**: IBM's Watson Research Center
 - ★ **pl. MS Access**
- **Structured Query Language** \Rightarrow **SQL**
 - ★ oszlopkalkulus-szerű alapjai vannak, némi sorkalkulus elemmel
 - ★ **kifejlesztő**: IBM's San Jose Research Laboratory
 - ★ **pl. Oracle, MS SQL, IBM DB2, MySQL, PostgreSQL**

A példákban használt relációs séma



MOZI	mozilD	név	város
	101	Művész	Budapest
	102	Uránia	Pécs
	⋮		

FILM	filmID	cím	rendező
	1	Macskajaj	E. Kusturica
	2	Moszkva tér	Török F.
	⋮		

VETÍT	mozilD	filmID	nap	idő
	101	1	péntek	16:00
	101	2	szombat	19:00
	⋮			

ISBL

- Relációs algebra alapú lekérdezések

- ★ operátorok

művelet	relációs algebra	ISBL
unió	$R \cup S$	$R + S$
metszet	$R \cap S$	$R \cdot S$
természetes illesztés	$R \bowtie S$	$R * S$
különbség (általánosabb)	$R \setminus (R \bowtie S)$	$R - S$
kiválasztás	$\sigma_F(R)$	$R:F$
vetítés	$\pi_{A,B,C}(R)$	$R \% A,B,C$

ISBL

- Relációs algebra alapú lekérdezések

- ★ operátorok

művelet	relációs algebra	ISBL
unió	$R \cup S$	$R + S$
metszet	$R \cap S$	$R \cdot S$
természetes illesztés	$R \bowtie S$	$R * S$
különbség (általánosabb)	$R \setminus (R \bowtie S)$	$R - S$
kiválasztás	$\sigma_F(R)$	$R:F$
vetítés	$\pi_{A,B,C}(R)$	$R \% A,B,C$

- ★ egyéb elemek

- * eredmény megjelenítése: **list** kulcsszó
- * értékadás relációnak: **=**
- * és, vagy, tagadás jelei: **&**, **|**, **¬**

ISBL

- Relációs algebra alapú lekérdezések

- ★ operátorok

művelet	relációs algebra	ISBL
unió	$R \cup S$	$R + S$
metszet	$R \cap S$	$R . S$
természetes illesztés	$R \bowtie S$	$R * S$
különbség (általánosabb)	$R \setminus (R \bowtie S)$	$R - S$
kiválasztás	$\sigma_F(R)$	$R:F$
vetítés	$\pi_{A,B,C}(R)$	$R \% A,B,C$

- ★ egyéb elemek

- * eredmény megjelenítése: **list** kulcsszó
- * értékadás relációnak: **=**
- * és, vagy, tagadás jelei: **&**, **|**, **¬**

- További nyelvi elemek

- ★ aggregátumok (min., max., összeg, átlag, darabszám) kezelése
- ★ adatfrissítő műveletek
- ★ átnevezés
- ★ kimenet formázása

ISBL példák

- Budapesti mozik nevei
`list MOZI : város = "Budapest" % név`

ISBL példák

- Budapesti mozik nevei
list MOZI : város = "Budapest" % név
- Pénteken 16 órakor kezdődő filmek címe, rendezője
list FILM * VETÍT : nap = "péntek" & idő = "16:00" % cím, rendező

ISBL példák

- Budapesti mozik nevei
list MOZI : város = "Budapest" % név
- Pénteken 16 órakor kezdődő filmek címe, rendezője
list FILM * VETÍT : nap = "péntek" & idő = "16:00" % cím, rendező
- Pénteken nem vetített filmek címe
list FILM – (VETÍT : nap = "péntek") % cím
(Figyelem! Különbségnek más a definíciója.)

ISBL példák

- Budapesti mozik nevei
list MOZI : város = "Budapest" % név
- Pénteken 16 órakor kezdődő filmek címe, rendezője
list FILM * VETÍT : nap = "péntek" & idő = "16:00" % cím, rendező
- Pénteken nem vetített filmek címe
list FILM – (VETÍT : nap = "péntek") % cím
(Figyelem! Különbségnek más a definíciója.)
- Pénteken és szombaton is vetített filmek címe
list (FILM * VETÍT : nap = "péntek" % cím) .
(FILM * VETÍT : nap = "szombat" % cím)

ISBL példák

- Budapesti mozik nevei
list MOZI : város = "Budapest" % név
- Pénteken 16 órakor kezdődő filmek címe, rendezője
list FILM * VETÍT : nap = "péntek" & idő = "16:00" % cím, rendező
- Pénteken nem vetített filmek címe
list FILM – (VETÍT : nap = "péntek") % cím
(Figyelem! Különbségnek más a definíciója.)
- Pénteken és szombaton is vetített filmek címe
list (FILM * VETÍT : nap = "péntek" % cím) .
(FILM * VETÍT : nap = "szombat" % cím)
vagy köztes relációk bevezetésével:
r1 = FILM * VETÍT : nap = "péntek" % cím
r2 = FILM * VETÍT : nap = "szombat" % cím
list r1.r2

QUEL

Sorkalkulus alapú.

Pl. Budapesti mozik nevei:

$\{t[2] \mid \text{MOZI}(t) \wedge t[3] = \text{'Budapest'}\}$

QUEL

Sorkalkulus alapú.

Pl. Budapesti mozik nevei:

$\{t[2] \mid \text{MOZI}(t) \wedge t[3] = \text{'Budapest'}\}$

Meg kell mondani, hogy a sorváltozó melyik reláció sorain fut:

$R(t) \iff \text{range of } t \text{ is } R$

QUEL

Sorkalkulus alapú.

Pl. Budapesti mozik nevei:

$\{t[2] \mid \text{MOZI}(t) \wedge t[3] = \text{'Budapest'}\}$

Meg kell mondani, hogy a sorváltozó melyik reláció sorain fut:

$R(t) \iff \text{range of } t \text{ is } R$

Hivatkozni kell a sorváltozó komponenseire:

$t[i] \iff t.<i\text{-edik attributum neve}>$

QUEL

Sorkalkulus alapú.

Pl. Budapesti mozik nevei:

$\{t[2] \mid \text{MOZI}(t) \wedge t[3] = \text{'Budapest'}\}$

Meg kell mondani, hogy a sorváltozó melyik reláció sorain fut:

$R(t) \longleftrightarrow \text{range of } t \text{ is } R$

Hivatkozni kell a sorváltozó komponenseire:

$t[i] \longleftrightarrow t.<i\text{-edik attributum neve}>$

Pl. $t[3] \longleftrightarrow t.\text{város}$

QUEL

Sorkalkulus alapú.

Pl. Budapesti mozik nevei:

$\{t[2] \mid \text{MOZI}(t) \wedge t[3] = \text{'Budapest'}\}$

Meg kell mondani, hogy a sorváltozó melyik reláció sorain fut:

$R(t) \longleftrightarrow \text{range of } t \text{ is } R$

Hivatkozni kell a sorváltozó komponenseire:

$t[i] \longleftrightarrow t.<i\text{-edik attributum neve}>$

Pl. $t[3] \longleftrightarrow t.\text{város}$

Lekérdezés:

retrieve (\langle lekérdezendő attribútumok \rangle) **where** \langle feltétel \rangle

QUEL példák

range of *m* is MOZI

range of *f* is FILM

range of *v* is VETÍT

range of *u* is VETÍT

QUEL példák

range of *m* is MOZI

range of *f* is FILM

range of *v* is VETÍT

range of *u* is VETÍT

- Budapesti mozik nevei
 retrieve (*m.név*) where *m.város* = 'Budapest'

QUEL példák

range of *m* is MOZI

range of *f* is FILM

range of *v* is VETÍT

range of *u* is VETÍT

- Budapesti mozik nevei
retrieve (*m.név*) where *m.város* = 'Budapest'
- Pénteken 16 órakor kezdődő filmek címe, rendezője
retrieve unique (*f.cím*, *f.rendező*)
where *f.filmID*=*v.filmID* and *v.nap* = 'péntek' and *v.idő* = '16:00'

QUEL példák

range of m is MOZI

range of f is FILM

range of v is VETÍT

range of u is VETÍT

- Budapesti mozik nevei
retrieve ($m.név$) where $m.város = 'Budapest'$
- Pénteken 16 órakor kezdődő filmek címe, rendezője
retrieve unique ($f.cím, f.rendező$)
where $f.filmID=v.filmID$ and $v.nap = 'péntek'$ and $v.idő = '16:00'$
- Pénteken nem vetített filmek címe:
 $\{t^{(3)}[2] \mid \text{FILM}(t) \wedge \nexists s (\text{VETÍT}(s) \wedge s[2] = t[1] \wedge s[3] = 'péntek')\}$

QUEL példák

range of m is MOZI

range of f is FILM

range of v is VETÍT

range of u is VETÍT

- Budapesti mozik nevei
retrieve ($m.név$) where $m.város = 'Budapest'$
- Pénteken 16 órakor kezdődő filmek címe, rendezője
retrieve unique ($f.cím, f.rendező$)
where $f.filmID=v.filmID$ and $v.nap = 'péntek'$ and $v.idő = '16:00'$
- Pénteken nem vetített filmek címe:
 $\{t^{(3)}[2] \mid \text{FILM}(t) \wedge \nexists s (\text{VETÍT}(s) \wedge s[2] = t[1] \wedge s[3] = 'péntek')\}$
 QUEL-ben nincsenek igazi kvantorok, csak egy **any** nevű aggregatum!

QUEL példák

range of m is MOZI

range of f is FILM

range of v is VETÍT

range of u is VETÍT

- Budapesti mozik nevei

retrieve ($m.név$) **where** $m.város = 'Budapest'$

- Pénteken 16 órakor kezdődő filmek címe, rendezője

retrieve unique ($f.cím, f.rendező$)

where $f.filmID=v.filmID$ **and** $v.nap = 'péntek'$ **and** $v.idő = '16:00'$

- Pénteken nem vetített filmek címe:

$\{t^{(3)}[2] \mid \text{FILM}(t) \wedge \nexists s (\text{VETÍT}(s) \wedge s[2] = t[1] \wedge s[3] = 'péntek')\}$

QUEL-ben nincsenek igazi kvantorok, csak egy **any** nevű aggregatum!

retrieve ($f.cím$)

where any ($f.filmID$ **where** $f.filmID=v.filmID$ **and** $v.nap = 'péntek'$)=0

Lehet beszúrni és törölni:

retrieve into FILM1 **unique** *f.cím*

retrieve into FILM2 **unique** *f.cím* **where** *f.filmID=v.filmID* **and** *v.nap='péntek'*

range of *f1* **is** FILM1

range of *f2* **is** FILM2

delete *f1* **where** *f1.filmID=f2.filmID*

retrieve FILM1

Lehet beszúrni és törölni:

retrieve into FILM1 **unique** *f.cím*

retrieve into FILM2 **unique** *f.cím* **where** *f.filmID=v.filmID* **and** *v.nap='péntek'*

range of *f1* **is** FILM1

range of *f2* **is** FILM2

delete *f1* **where** *f1.filmID=f2.filmID*

retrieve FILM1

- Pénteken és szombaton is vetített filmek címe

retrieve unique (*f.cím*)

where *f.filmID=v.filmID* **and** *v1.filmID=v.filmID* **and**
v.nap = 'péntek' **and** *v1.nap = 'szombat'*

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós
 - ★ A lekérdezés elemei **változókkal** és **konstansokkal** kitöltött **sablon(ok)**
 - * jelölések

változó	aláhúzott változónév
konstans	nem aláhúzott érték
egyszer említett változó	üres cella
kimenetre kerülő attribútum	P. prefix

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós
- ★ A lekérdezés elemei **változókkal** és **konstansokkal** kitöltött **sablon(ok)**
- * jelölések

változó	aláhúzott változónév
konstans	nem aláhúzott érték
egyszer említett változó	üres cella
kimenetre kerülő attribútum	P. prefix

- * Példa: Budapesti mozik nevei:

MOZI	mozID	név	város
	<u>1</u>	P. <u>mozinév</u>	Budapest

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós
 - ★ A lekérdezés elemei **változókkal** és **konstansokkal** kitöltött **sablon(ok)**
 - * jelölések

változó	aláhúzott változónév
konstans	nem aláhúzott érték
egyszer említett változó	üres cella
kimenetre kerülő attribútum	P. prefix

- * Példa: Budapesti mozik nevei:

MOZI	mozID	név	város
	<u>1</u>	P. <u>mozinév</u>	Budapest

vagy

MOZI	mozID	név	város
		P.	Budapest

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós
- ★ A lekérdezés elemei **változókkal** és **konstansokkal** kitöltött **sablon(ok)**
- * jelölések

változó	aláhúzott változónév
konstans	nem aláhúzott érték
egyszer említett változó	üres cella
kimenetre kerülő attribútum	P. prefix

- * Példa: Budapesti mozik nevei:

MOZI	mozID	név	város
	<u>1</u>	P. <u>mozinév</u>	Budapest

vagy

MOZI	mozID	név	város
		P.	Budapest

- ★ **Összetett lekérdezések** is lehetségesek (használatukkor az **azonos nevű változók** illesztése történik meg).
- * **használható több soros sablon** (ekkor a kiértékeléskor mindegyik sornak egy-egy futó oszlopváltozó fog megfelelni, és ha illeszkedés van, akkor megtörténik a kiírás)

QBE

- Oszlopkalkulus alapú lekérdezések, kétdimenziós
- ★ A lekérdezés elemei **változókkal** és **konstansokkal** kitöltött **sablon(ok)**
- * jelölések

változó	aláhúzott változónév
konstans	nem aláhúzott érték
egyszer említett változó	üres cella
kimenetre kerülő attribútum	P. prefix

- * Példa: Budapesti mozik nevei:

MOZI	mozID	név	város
	<u>1</u>	P. <u>mozinév</u>	Budapest

vagy

MOZI	mozID	név	város
		P.	Budapest

- ★ **Összetett lekérdezések** is lehetségesek (használatukkor az **azonos nevű változók** illesztése történik meg).
- * **használható több soros sablon** (ekkor a kiértékeléskor mindegyik sornak egy-egy futó oszlopváltozó fog megfelelni, és ha illeszkedés van, akkor megtörténik a kiírás)
- * **használható több sablon** (kiértékelés hasonlóan, mint a többsoros kérdésnél, csak az oszlopváltozók nem ugyanazon reláció sorait futják be)

★ A kiválasztás feltételeinek megadása

- * Az **egyenlőség** konstanshoz való illesztéssel vizsgálható (mint az előbb),

★ A kiválasztás feltételeinek megadása

- * Az **egyenlőség** konstanshoz való illesztéssel vizsgálható (mint az előbb),
- * **egyéb egyszerű relációkhoz** a \neg , $=$, $>$, $<$, $>=$, $<=$ operátorok használhatók,

★ A kiválasztás feltételeinek megadása

- * Az **egyenlőség** konstanshoz való illesztéssel vizsgálható (mint az előbb),
- * **egyéb egyszerű relációkhoz** a \neg , $=$, $>$, $<$, $>=$, $<=$ operátorok használhatók,
- * **összetett feltételeket** (pl. két változó közt a $<$ relációt) külön feltételsablon megadásával lehet vizsgálni.

- ★ A kiválasztás feltételeinek megadása
 - * Az **egyenlőség** konstanshoz való illesztéssel vizsgálható (mint az előbb),
 - * **egyéb egyszerű relációkhoz** a \neg , $=$, $>$, $<$, $>=$, $<=$ operátorok használhatók,
 - * **összetett feltételeket** (pl. két változó közt a $<$ relációt) külön feltételsablon megadásával lehet vizsgálni.
- További nyelvi elemek
 - ★ mintaillesztés
 - ★ aritmetika
 - ★ kimenet rendezése
 - ★ csoportosítás
 - ★ aggregátumok kezelése
 - ★ reláció tranzitív lezártjának kezelése
 - ★ adatmódosító műveletek
 - ★ típusdefiníció, sémalétrehozás

Megjegyzés: dupla példányt kiirtja, azaz többszörös sorok nincsenek

QBE példák még

- Nem budapesti mozik nevei

MOZI	mozID	név	város
		P.	\neg = Budapest

QBE példák még

- Nem budapesti mozik nevei

MOZI	mozilD	név	város
		P.	\neg = Budapest

- Pénteki és szombati kezdési időpontok

VETÍT	mozilD	filmID	nap	idő
			P. péntek	P.
			P. szombat	P.

QBE példák még

- Nem budapesti mozik nevei

MOZI	mozilD	név	város
		P.	\neg = Budapest

- Pénteki és szombati kezdési időpontok

VETÍT	mozilD	filmID	nap	idő
			P. péntek	P.
			P. szombat	P.

- Időpontok, amikor pénteken és szombaton is kezdődik film

VETÍT	mozilD	filmID	nap	idő
			péntek	P. <u>kezdes</u>
			szombat	<u>kezdes</u>

- Pénteken vetített filmek adatai

FILM	filmID	cím	rendező
	<u>1</u>	P.	P.

VETÍT	mozilD	filmID	nap	idő
		<u>1</u>	péntek	

- Pénteken vetített filmek adatai

FILM	filmID	cím	rendező
	<u>1</u>	P.	P.

VETÍT	mozilD	filmID	nap	idő
		<u>1</u>	péntek	

- Azok a városok, ahol van legalább két mozi:

MOZI	mozilD	név	város
	<u>1</u>		P. <u>városnév</u>
	<u>2</u>		<u>városnév</u>

CONDITIONS
<u>1</u> ≠ <u>2</u>

Az SQL nyelv

- Relációs nyelv, mint az eddigiek

Az SQL nyelv

- Relációs nyelv, mint az eddigiek
- oszlopkalkulus jellegű, de némi sorkalkulusos beütéssel

Az SQL nyelv

- Relációs nyelv, mint az eddigiek
- oszlopkalkulus jellegű, de némi sorkalkulusos beütéssel

Termékek, (amik szükségszerűen relációs nyelvet is tartalmaztak):

- IBM: System/R
- Relational Software: Oracle
- Relational Systems: Ingres

Az SQL nyelv

- Relációs nyelv, mint az eddigiek
- oszlopkalkulus jellegű, de némi sorkalkulusos beütéssel

Termékek, (amik szükségszerűen relációs nyelvet is tartalmaztak):

- IBM: System/R
- Relational Software: Oracle
- Relational Systems: Ingres

Szabványok

- SQL89 (SQL1)
- SQL92 (SQL2, mi nagyrészt ezt nézzük most)
- SQL99 (SQL3, ebből is pár dolog, pl. triggererek, rekurzió)

Az SQL nyelv

- Relációs nyelv, mint az eddigiek
- oszlopkalkulus jellegű, de némi sorkalkulusos beütéssel

Termékek, (amik szükségszerűen relációs nyelvet is tartalmaztak):

- IBM: System/R
- Relational Software: Oracle
- Relational Systems: Ingres

Szabványok

- SQL89 (SQL1)
- SQL92 (SQL2, mi nagyrészt ezt nézzük most)
- SQL99 (SQL3, ebből is pár dolog, pl. triggererek, rekurzió)

Működő rendszerekben ezek verziói vannak (főleg SQL2).

Fontosabb utasítások

Data Definition Language:

- CREATE - séma létrehozása
- ALTER - séma módosítása
- DROP - séma törlése

Fontosabb utasítások

Data Definition Language:

- CREATE - séma létrehozása
- ALTER - séma módosítása
- DROP - séma törlése

Data Modification Language:

- INSERT - adatok beszúrása
- UPDATE - adatok módosítása
- DELETE - adatok törlése
- SELECT - adatok lekérdezése

Fontosabb utasítások

Data Definition Language:

- CREATE - séma létrehozása
- ALTER - séma módosítása
- DROP - séma törlése

Data Modification Language:

- INSERT - adatok beszúrása
- UPDATE - adatok módosítása
- DELETE - adatok törlése
- SELECT - adatok lekérdezése

Természetesen előbb mindig a sémát kell létrehozni, és utána dolgozhatunk vele, de most fordítva tárgyaljuk mert eddig a lekérdező nyelvekről volt szó.

DML utasítások — SELECT

Ezzel valósítható meg a kiválasztás, vetítés és a szorzat.

Szintaxis: **SELECT** <reláció_i>.<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):

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

Példa 1: A budapesti mozik azonosítói és nevei

```
SELECT mozi.mozilD, mozi.név FROM mozi WHERE mozi.város="Budapest"
```

Példa 2: A pénteken hétkor kezdődő filmek azonosítói

```
SELECT vetít.filmID FROM vetít WHERE vetít.nap="péntek" AND vetít.idő="19:00"
```

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHERE

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztán arra csinálnánk a kiválasztást és a vetítést.

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztán arra csinálnánk a kiválasztást és a vetítést.
- ha többszörös sorokat nem akarunk: SELECT DISTINCT (ennek ára van!!!)

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztán arra csinálnánk a kiválasztást és a vetítést.
- ha többszörös sorokat nem akarunk: SELECT DISTINCT (ennek ára van!!!)
- WHERE el is hagyható

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztán arra csinálnánk a kiválasztást és a vetítést.
- ha többszörös sorokat nem akarunk: SELECT DISTINCT (ennek ára van!!!)
- WHERE el is hagyható
- WHERE-ben mi állhat: erről később

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztán arra csinálnánk a kiválasztást és a vetítést.
- ha többszörös sorokat nem akarunk: SELECT DISTINCT (ennek ára van!!!)
- WHERE el is hagyható
- WHERE-ben mi állhat: erről később
- az eredmény az ORDER BY kulcsszó segítségével rendezhető, megadható hogy mely oszlopok szerint és hogy növekvőleg vagy csökkenőleg

Megjegyzés:

- **kiértékelés:** minden egyes FROM utáni relációnak megfelel egy-egy sorváltozó, ami az egyes relációk sorain megy végig (egymásba ágyazott ciklusokkal például). Ha találat van, azaz a WHEREfeltétel igaz az aktuális értékekre, akkor a SELECT utáni mezők kiíródnak
- úgy gondolhatunk a kiértékelésre, mintha először vennénk a FROM utáni relációk direkt szorzatát és aztán arra csinálnánk a kiválasztást és a vetítést.
- ha többszörös sorokat nem akarunk: SELECT DISTINCT (ennek ára van!!!)
- WHERE el is hagyható
- WHERE-ben mi állhat: erről később
- az eredmény az ORDER BY kulcsszó segítségével rendezhető, megadható hogy mely oszlopok szerint és hogy növekvőleg vagy csökkenőleg
- A fenti két példa mutatja, hogy a kiválasztás és a vetítés megy, a szorzatra a sorváltozók bevezetése után nézünk példát