



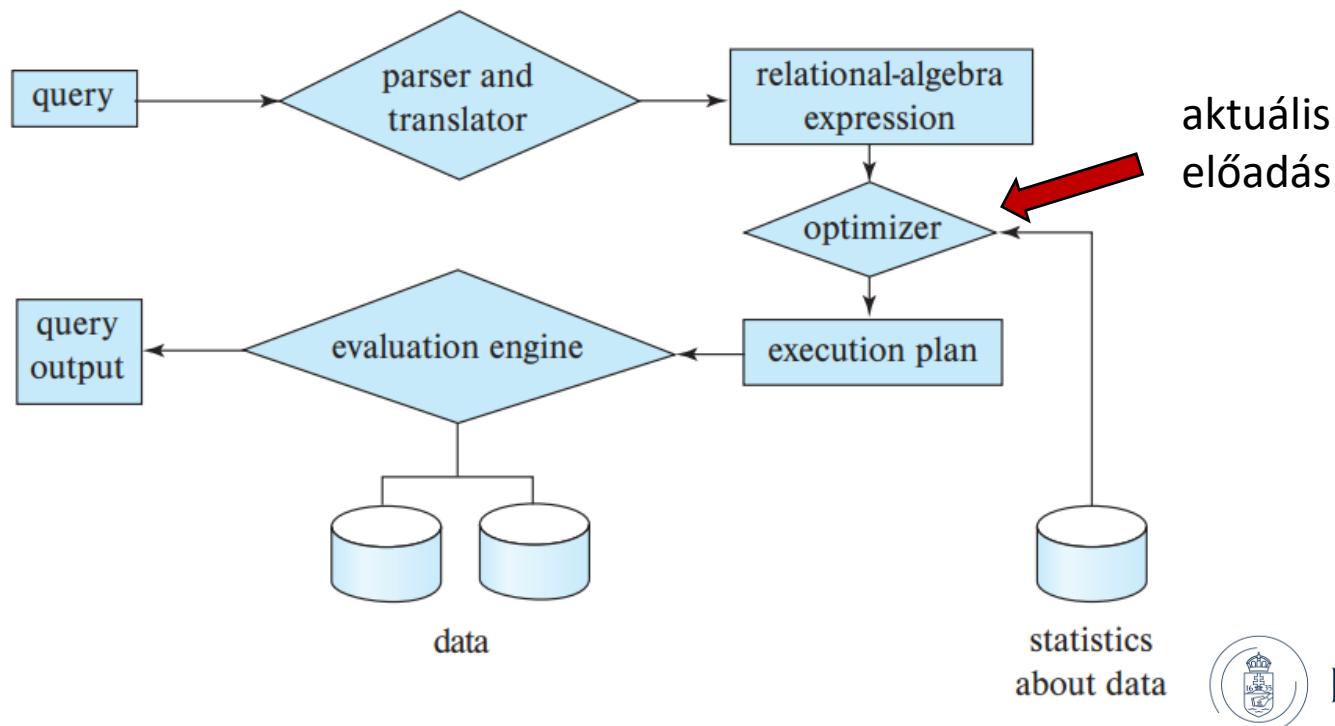
ELTE | IK
INFORMATIKAI KAR

Adatbázisok 2

Lekérdezések optimalizációja

Lekérdezésfeldolgozás lépései

- Parser and translator – logikai lekérdezéstervet készít (kifejezés fa).
- **Optimizer** – relációs algebrai (szabály alapú) és költség alapú optimalizálás.
- Evaluation engine – végrehajtja az elkészült fizikai tervet.



Lekérdezés optimalizáció

1. Megfelelő (ekvivalens) kifejezés fa kiválasztása. Ez egy NP-nehéz feladat.
 2. A kifejezés fához fizikai lekérdezés tervez készítése. Az egyes tervez költségének a becslése.
 3. A legkisebb költségű terv kiválasztása.
- **Cél:** A lekérdezéseket gyorsabbá akarjuk tenni a táblákra vonatkozó paraméterek, statisztikák, indexek ismeretében.

Két féle optimalizáció

- Heurisztikus / szabály alapú / algebrai
 - Ekvivalencia szabályok alapján a kifejezés fa átalakítása.
- Költség alapú
 - A műveletek költségének és a kimenetek méreteinek a felhasználásával becsüljük a költségeket.

Motiváció

- Legyenek *Dolgozo(dnev, foglalkozas, oazon)* és *Osztaly(oazon, onev)* relációk.
- **Lekérdezés:** Adjuk meg a SALESMAN foglalkozású dolgozók nevét és annak az osztálynak a nevét, ahová tartoznak.
- SQL-ben:

```
SELECT dnev, onev  
FROM Dolgozo D, Osztaly O  
WHERE D.oazon=O.oazon AND D.foglalkozas='SALESMAN';
```



Motiváció

Dolgozo(dnev, foglalkozas, oazon)

dnev	foglalkozas	oazon
KING	PRESIDENT	10
MARTIN	SALESMAN	20
LOLA	CLERK	30
MILLER	SALESMAN	20
SMITH	CLERK	20

Osztaly(oazon, onev)

oazon	onev
10	ACCOUNTING
20	SALES
30	OPERATIONS
40	RESEARCH

- A lekérdezés eredménye:

dnev	onev
MARTIN	SALES
MILLER	SALES



Egy lehetséges terv

- Lekérdezés SQL-ben:

```
SELECT dnev, onev  
FROM Dolgozo D, Osztaly O  
WHERE D.oazon=O.oazon AND D.foglalkozas='SALESMAN';
```

- Relációs algebrában (átnevezések nélkül, D és O relációkkal):

$$\pi_{dnev,oazon}(\sigma_{D.oazon=O.oazon \wedge D.foglalkozas="SALESMAN"}(D \times O))$$

- Végrehajtás:

- Vegyük a két tábla szorzatát.
- Válasszuk ki a megfelelő sorokat (a feltételek alapján)
- Hajtsuk végre a vetítést.

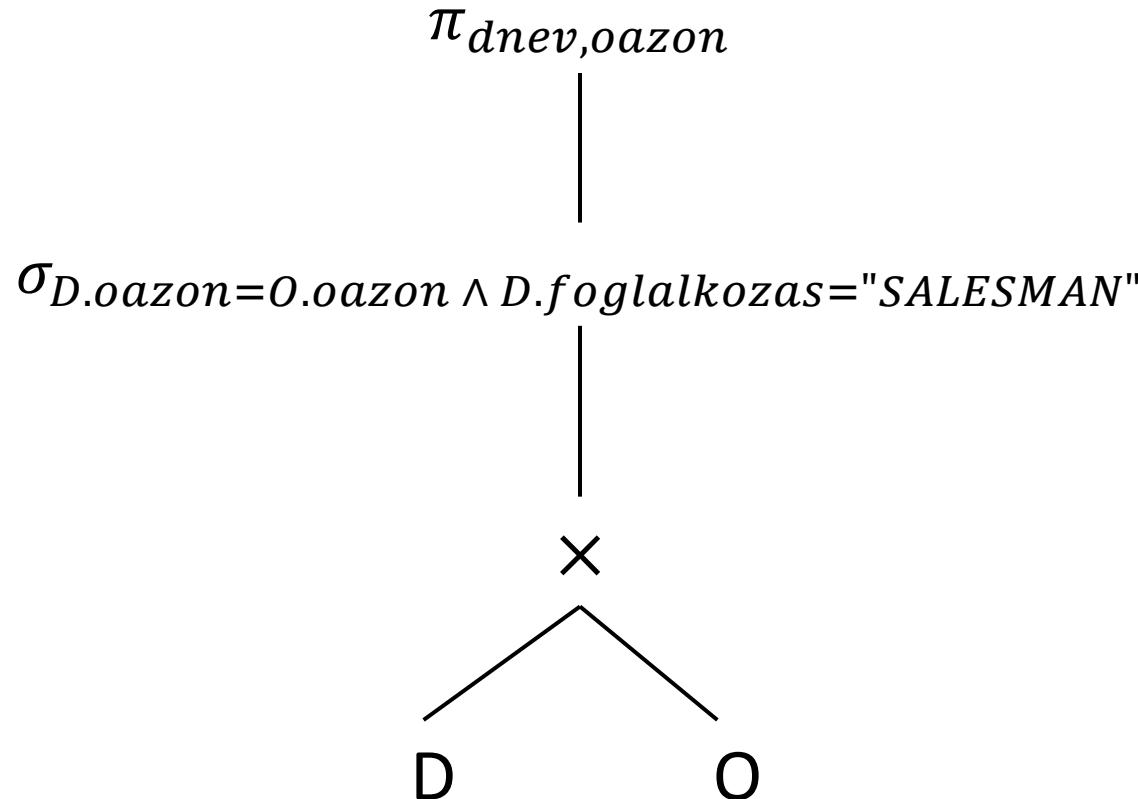


Egy lehetséges terv

- Relációs algebrában (átnevezések nélkül, D és O relációkkal):

$$\pi_{dnev,oazon}(\sigma_{D.oazon=O.oazon \wedge D.foglalkozas="SALESMAN"}(D \times O))$$

- Kifejezés fa:



Egy lehetséges terv

- A kereszt szorzat után megkeressük azokat a sorokat, amelyek kielégítik a feltételeket.

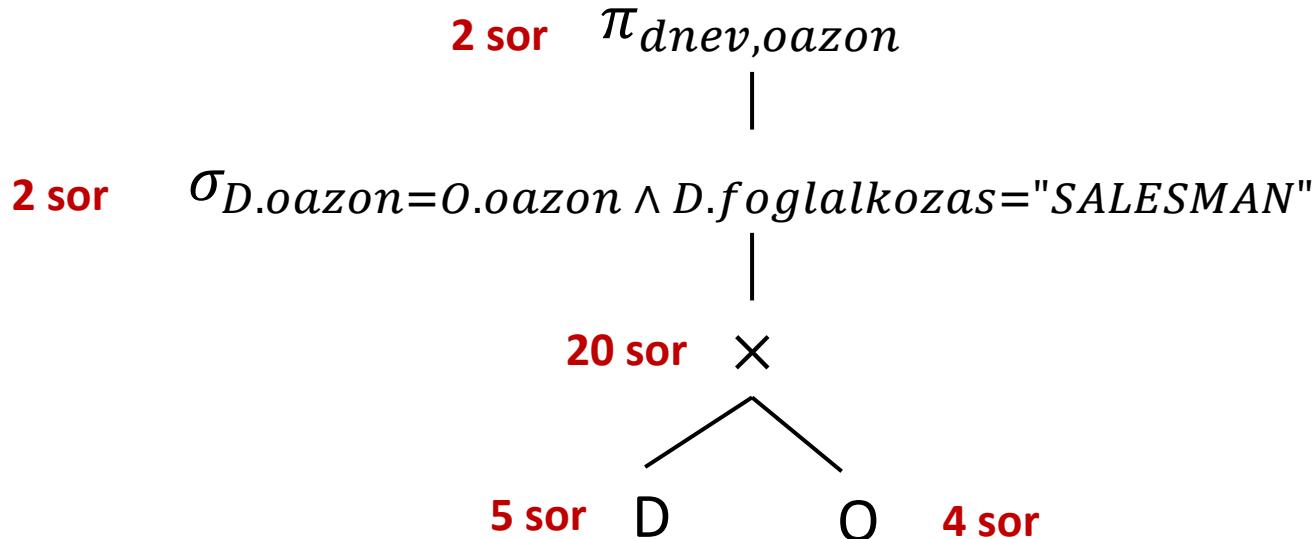


D.dnev	D.foglalkozas	D.oazon	O.oazon	O.onev
KING	PRESIDENT	10	10	ACCOUNTING
KING	PRESIDENT	10	20	SALES
KING	PRESIDENT	10	30	OPERATIONS
KING	PRESIDENT	10	40	RESEARCH
MARTIN	SALESMAN	20	10	ACCOUNTING
MARTIN	SALESMAN	20	20	SALES
MARTIN	SALESMAN	20	30	OPERATIONS
MARTIN	SALESMAN	20	40	RESEARCH
...

$$D \times O$$

Egy lehetséges terv

- Mi a baj a bemutatott tervvel?
 - A direktszorzathoz NESTED LOOP algoritmust kell használnunk, amely a legköltségesebb összekapcsolás.
 - Nagyon sok sort kapunk az összekapcsolás eredményeképp, de a feltételeknek igazából csak néhány sor felel meg.

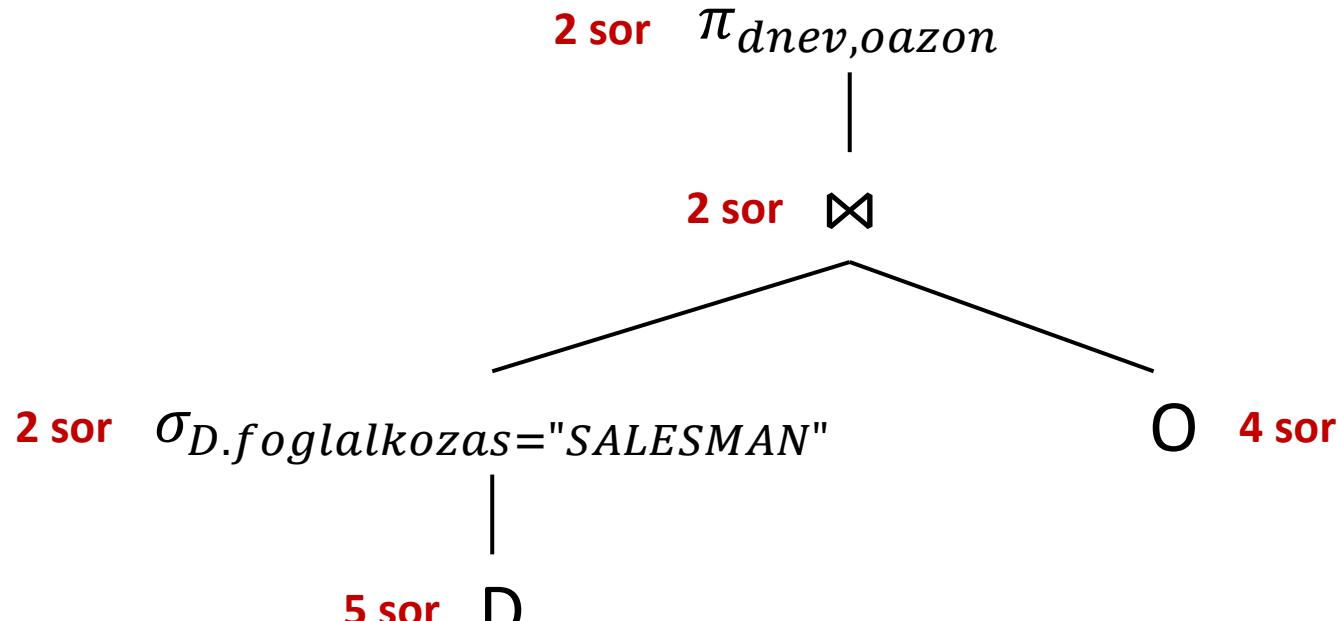


Egy másik lehetséges terv

- **Ekvivalens** relációs algebrai kifejezés:

$$\pi_{dnev,oazon}(\sigma_{D.foglalkozas="SALESMAN"}(D) \bowtie O)$$

- **Módosítás:** vigyük lentebb a kiválasztást! Az eredmény ugyanaz lesz
- Kifejezés fa:



További optimalizáció

- Tegyük fel, hogy van indexünk a **Dolgozo** tábla **foglalkozas** oszlopára és az **Osztaly** tábla **oazon** oszlopára.
- Egy újabb terv, indexek használatával:
 - Lineáris keresés helyett használjunk **indexet** a feltételnek (=”SALESMAN”) megfelelő sorok kiválasztására a **Dolgozo** táblából.
 - A kiválasztott sorok **oazon** értékeivel azonos sorokat keressük meg az **Osztaly** táblában egy index segítségével.
 - Adjuk vissza a megfelelő oszlopokat.



Szabály alapú optimalizáció

- **Cél:** a relációs algebrai kifejezések minél gyorsabb végrehajtása.
- **Költségmodell:** a részkifejezések mérete.
- **Módszer:** a műveleti tulajdonságokon alapuló ekvivalens átalakításokat alkalmazunk úgy, hogy várhatóan kisebb relációk keletkezzenek.
- **Megjegyzések:**
 - Heurisztikus, nem a relációk valódi méretével számol.
 - Az eredmény nem determinisztikus, mivel az átalakítások sorrendje nem kötött.



Szabály alapú optimalizáció

- A relációs algebrai kifejezéseket gráffal (**kifejezés fával**) fogjuk ábrázolni.
- Két féle csúcsok fordulnak elő:
 - Levél csúcsok: relációk vagy konstans relációk.
 - Nem levél csúcsok: relációs algebrai műveletek.
 - **Unáris** műveletek, amelyeknek egy gyereke van (σ, π, ρ).
 - **Bináris** műveletek, amelyeknek két gyerek van ($-$, \cup , \times).



Ekvivalencia

- $E_1(r_1, r_2 \dots r_k)$ és $E_2(r_1, r_2 \dots r_k)$ relációs algebrai kifejezések **ekvivalensek** ($E_1 \cong E_2$), ha tetszőleges $r_1, r_2 \dots r_k$ relációkat véve $E_1(r_1, r_2 \dots r_k) = E_2(r_1, r_2 \dots r_k)$
- 11 ekvivalencia szabályt adunk meg.
- A szabályok olyan állítások, amelyek kifejezések ekvivalenciáját fogalmazzák meg. Bizonyításuk könnyen végiggondolható.



Ekvivalencia szabályok

1. Kommutativitás

- Direkt szorzat: $E_1 \times E_2 \cong E_2 \times E_1$
- Természetes összekapcsolás: $E_1 \bowtie E_2 \cong E_2 \bowtie E_1$
- Theta-összekapcsolás: $E_1 \bowtie_\theta E_2 \cong E_2 \bowtie_\theta E_1$

2. Asszociativitás

- Direkt szorzat: $(E_1 \times E_2) \times E_3 \cong E_1 \times (E_2 \times E_3)$
- Természetes összekapcsolás: $(E_1 \bowtie E_2) \bowtie E_3 \cong E_1 \bowtie (E_2 \bowtie E_3)$
- Theta-összekapcsolás: $(E_1 \bowtie_\theta E_2) \bowtie_\theta E_3 \cong E_1 \bowtie_\theta (E_2 \bowtie_\theta E_3)$
- Megjegyzés: Ezek a szabályok azt jelentik, hogy több tábla összekapcsolása esetén az eredmény szempontjából mindegy, hogy milyen sorrendben tesszük ezt meg (teljesítmény szempontjából viszont számít).



Ekvivalencia szabályok

3. Vetítések összevonása, bővítése.

- Legyen A és B két részhalmaza az E reláció oszlopainak úgy, hogy $A \subseteq B$.
- Ekkor: $\pi_A(\pi_B(E)) \cong \pi_A(E)$

4. Kiválasztások felcserélhetősége, felbontása.

- Legyen F_1 és F_2 az E reláció oszlopain értelmezett kiválasztási feltételek.
- Ekkor: $\sigma_{F_1 \wedge F_2}(E) \cong \sigma_{F_1}(\sigma_{F_2}(E)) \cong \sigma_{F_2}(\sigma_{F_1}(E))$



Ekvivalencia szabályok

5. Kiválasztás és vetítés felcserélhetősége.

- Legyen F az E relációnak csak az A oszlopain értelmezett kiválasztási feltétel.
- Ekkor: $\pi_A(\sigma_F(E)) \cong \sigma_F(\pi_A(E))$
- Általános eset: Legyen F az E relációnak csak az $A \cup B$ oszlopain értelmezett kiválasztási feltétel, ahol $A \cap B = \emptyset$.
 - Ekkor: $\pi_A(\sigma_F(E)) \cong \pi_A(\sigma_F(\pi_{A \cup B}(E)))$
- Megjegyzés: Mi haszna van az utolsó ekvivalenciának, hiszen látszólag több műveletet végzünk. Ha $A \cup B$ számossága kisebb, mint az E oszlopainak a száma, akkor a kiválasztásnak kevesebb adaton kell dolgoznia.



Ekvivalencia szabályok

6. Kiválasztás és szorzás felcserélhetősége.

- Legyen F az E_1 reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel. **Ekkor:** $\sigma_F(E_1 \times E_2) \cong \sigma_F(E_1) \times E_2$
- Speciális eset, amikor konjukciós feltétel esetén az elemi feltételek az egyes relációkra vonatkoznak, azaz $i = 1, 2$ estén az F_i relációs az E_i reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel és $F = F_1 \wedge F_2$. **Ekkor:** $\sigma_F(E_1 \times E_2) \cong \sigma_{F_1}(E_1) \times \sigma_{F_2}(E_2)$
- Általánosabban, legyen F_1 legyen az E_1 reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel, F_2 az $E_1 \times E_2$ reláció oszlopainak egy részhalmazán értelmezett feltétel és $F = F_1 \wedge F_2$. **Ekkor:** $\sigma_F(E_1 \times E_2) \cong \sigma_{F_2}(\sigma_{F_1}(E_1) \times E_2)$.



Ekvivalencia szabályok

7. Kiválasztás és egyesítés felcserélhetősége.

- Legyen E_1, E_2 relációk sémája megegyező és F a közös sémán értelmezett kiválasztási feltétel.
- Ekkor: $\sigma_F(E_1 \cup E_2) \cong \sigma_F(E_1) \cup \sigma_F(E_2)$

8. Kiválasztás és kivonás felcserélhetősége.

- Legyen E_1, E_2 relációk sémája megegyező és F a közös sémán értelmezett kiválasztási feltétel.
- Ekkor: $\sigma_F(E_1 - E_2) \cong \sigma_F(E_1) - \sigma_F(E_2)$
- **Megjegyzés:** Feltűnhet, hogy ebben az esetben elég az F kiválasztási feltételt csak az E_1 relációra alkalmazni, és akkor is fennáll az ekvivalencia. A szabályban azért alkalmaztuk az E_2 -re is, mivel így kisebb relációt kapunk és gyorsíthatjuk a végrehajtást.



Ekvivalencia szabályok

9. Kiválasztás és természetes összekapcsolás felcserélhetősége.

- Legyen F az E_1 és E_2 közös oszlopainak egy részhalmazán értelmezett kiválasztási feltétel.
- Ekkor: $\sigma_F(E_1 \bowtie E_2) \cong \sigma_F(E_1) \bowtie \sigma_F(E_2)$

10. Vetítés és szorzás felcserélhetősége.

- Legyen $i = 1, 2$ estén A_i az E_i reláció oszlopainak egy halmaza, valamint legyen $A = A_1 \cup A_2$.
- Ekkor: $\pi_A(E_1 \times E_2) \cong \pi_{A_1}(E_1) \times \pi_{A_2}(E_2)$



Ekvivalencia szabályok

11. Vetítés és egyesítés felcserélhetősége.

- Legyen E_1 és E_2 relációk sémája megegyező, és legyen a sémában szereplő oszlopok egy részhalmaza A .
- Ekkor: $\pi_A(E_1 \cup E_2) \cong \pi_A(E_1) \cup \pi_A(E_2)$
- **Megjegyzés:** A vetítés és a kivonás **nem** cserélhető fel.
- Azaz: $\pi_A(E_1 - E_2) \not\cong \pi_A(E_1) - \pi_A(E_2)$
- Feladat: Adjunk erre egy példát!
- **Megfontolásra érdemes:** Milyen ekvivalencia szabályokat tudunk megadni a csoportosítás és kiválasztás felcserélhetőségére és a külső összekapcsolásokra (bal oldali, jobb oldali, teljes)?

Szabály alapú optimalizáció

- A szabály alapú optimalizáció a következő heurisztikus elveken alapul:
 - **Minél hamarabb szelektáljunk**, hogy a részkifejezések várhatóan kisebb relációk legyenek (a szigorúbb kiválasztásokat végezzük el leghamarabb).
 - A szorzás utáni kiválasztásokból próbálunk **természetes (vagy feltétel alapú) összekapcsolásokat képezni**, mert az hatékonyabban kiszámolható.
 - A vetítéseket vigyük minél lentebb a kifejezés fában.
 - **Vonjuk össze az egymás utáni unáris műveleteket** és ezekből lehetőleg egy kiválasztás utáni vetítést képezzünk, így csökken a műveletek száma.
 - **Keressünk közös részkifejezéseket**, amiket így elég csak egyszer kiszámolni a kifejezés kiértékelése során.



Optimalizációs algoritmus

- **INPUT:** relációs algebrai kifejezés kifejezésfája.
- **OUTPUT:** optimalizált kifejezésfa.
- Lépései:
 1. A kiválasztásokat bontsuk fel a **4. szabály** segítségével.
 2. A kiválasztásokat az **5., 6., 7., 8., 9. szabályok** segítségével vigyük olyan mélyre a fában, amennyire csak lehet.
 3. A vetítéseket a **3., 5., 10., 11. szabályok** segítségével vigyük olyan mélyre a fában, amennyire csak lehet.
 4. Az egymás utáni kiválasztásokat és vetítéseket vonjuk össze egy kiválasztássá vagy egy vetítéssé, vagy egy kiválasztás utáni vetítéssé (ha lehetséges).



Példa optimalizációra

- A következő relációkat (táblákat) fogjuk használni a példákban:
 - **kv:** könyv(id, szerző, cím)
 - **kő:** kölcsönző(azon, név, lakcím)
 - **ks:** kölcsönzés(id, azon, dátum)
- **Lekérdezés:** Kik és milyen könyveket kölcsönöztek 2025.10.01 után?

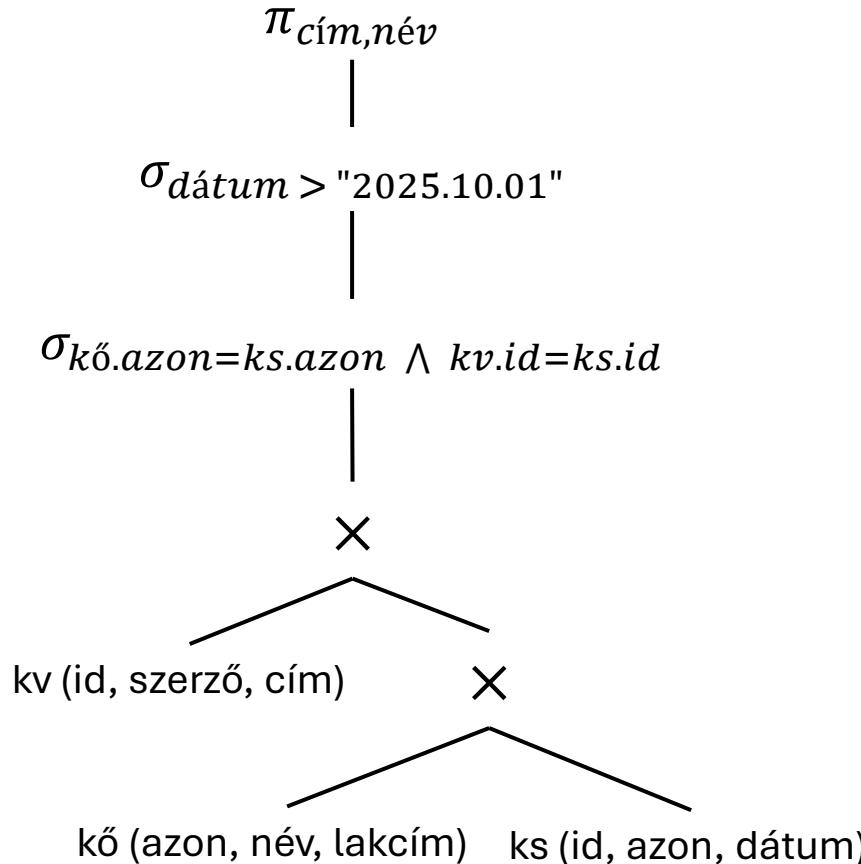


Relációs algebrai kifejezés és kifejezésfa

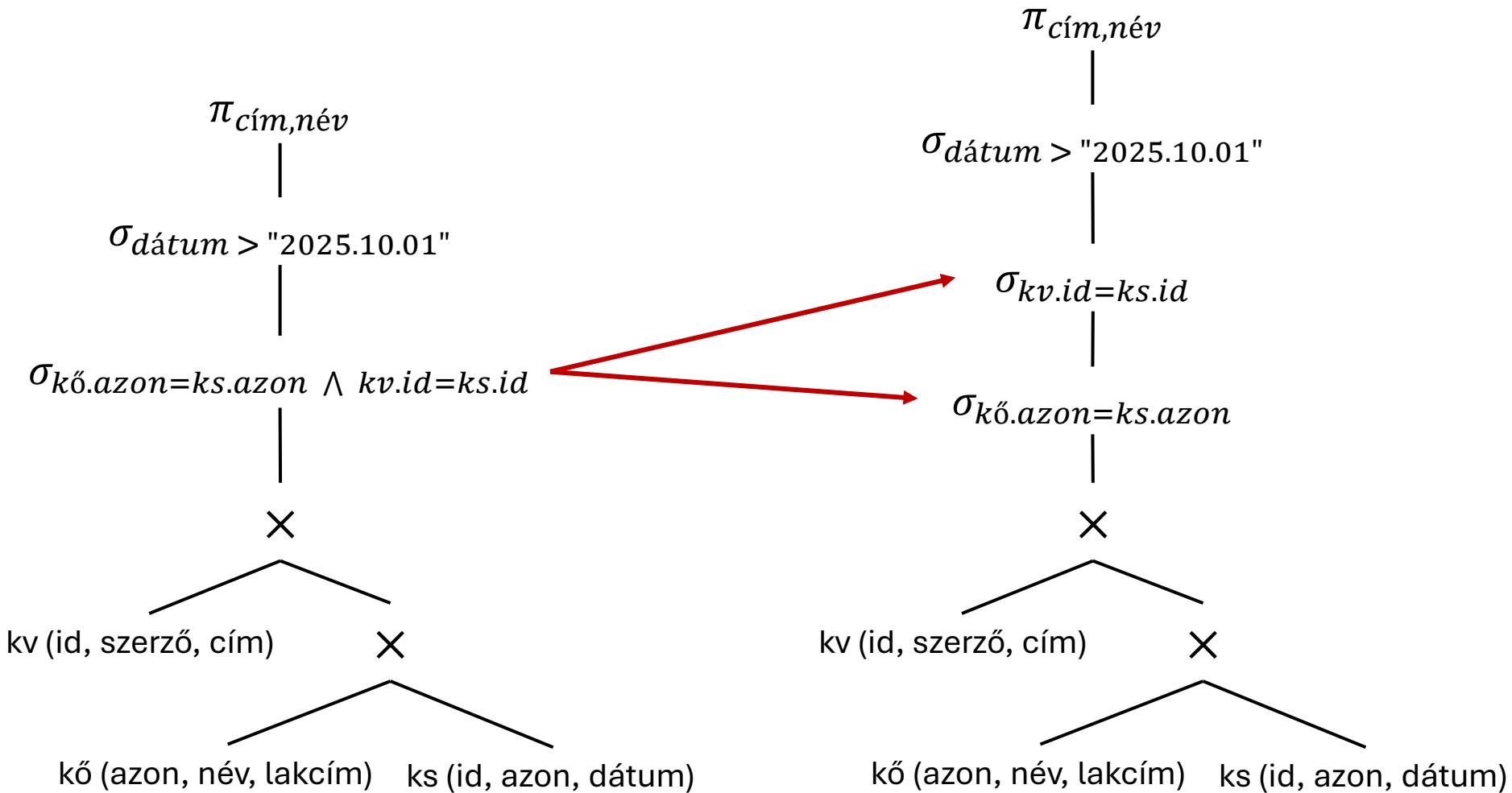
- Relációs algebrában:

$$\pi_{cím,név}(\sigma_{dátum > "2025.10.01"}(\sigma_{kő.azon=ks.azon \wedge kv.id=ks.id}(kv \times (kő \times ks))))$$

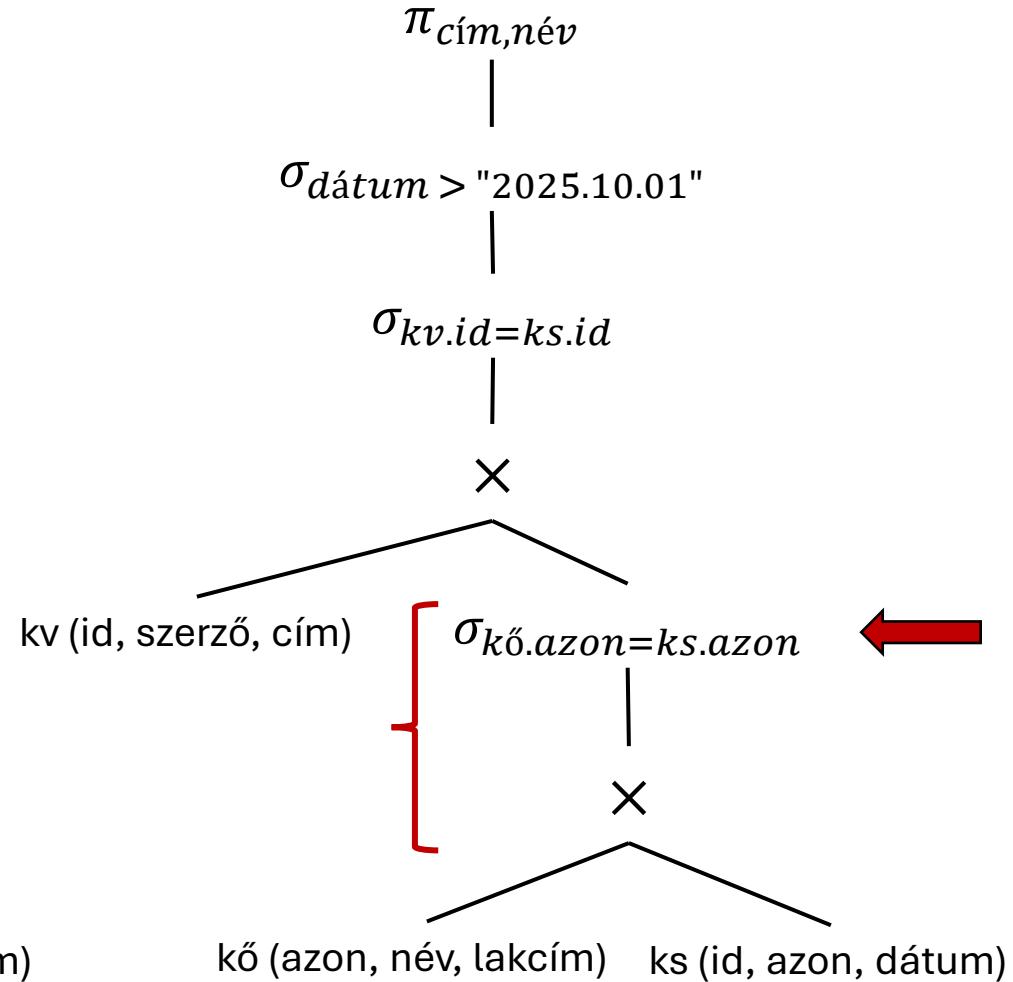
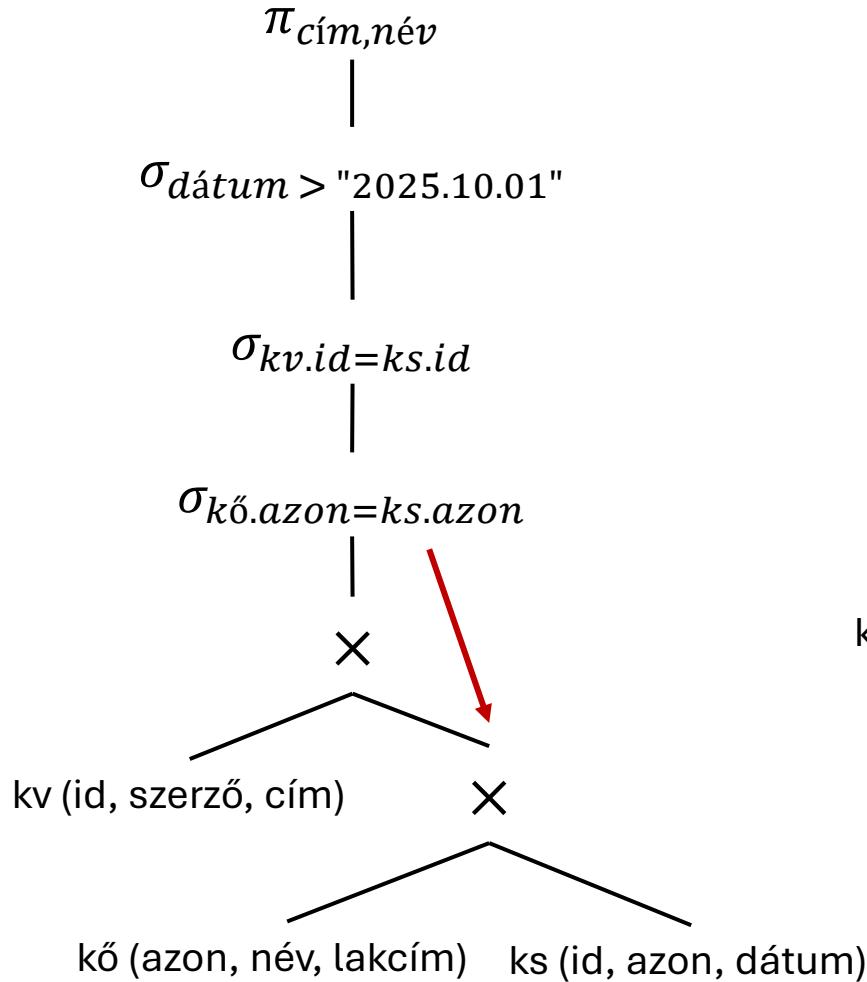
- Kifejezésfa:



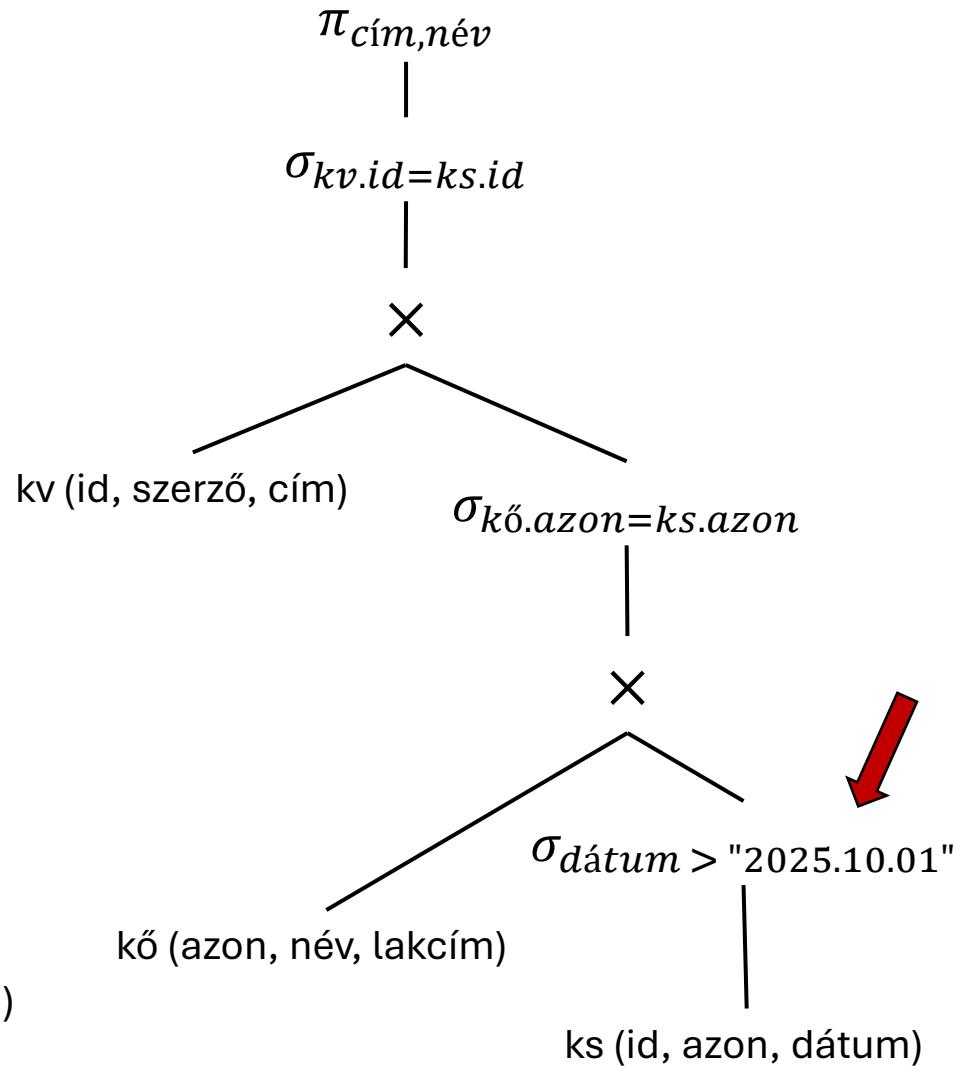
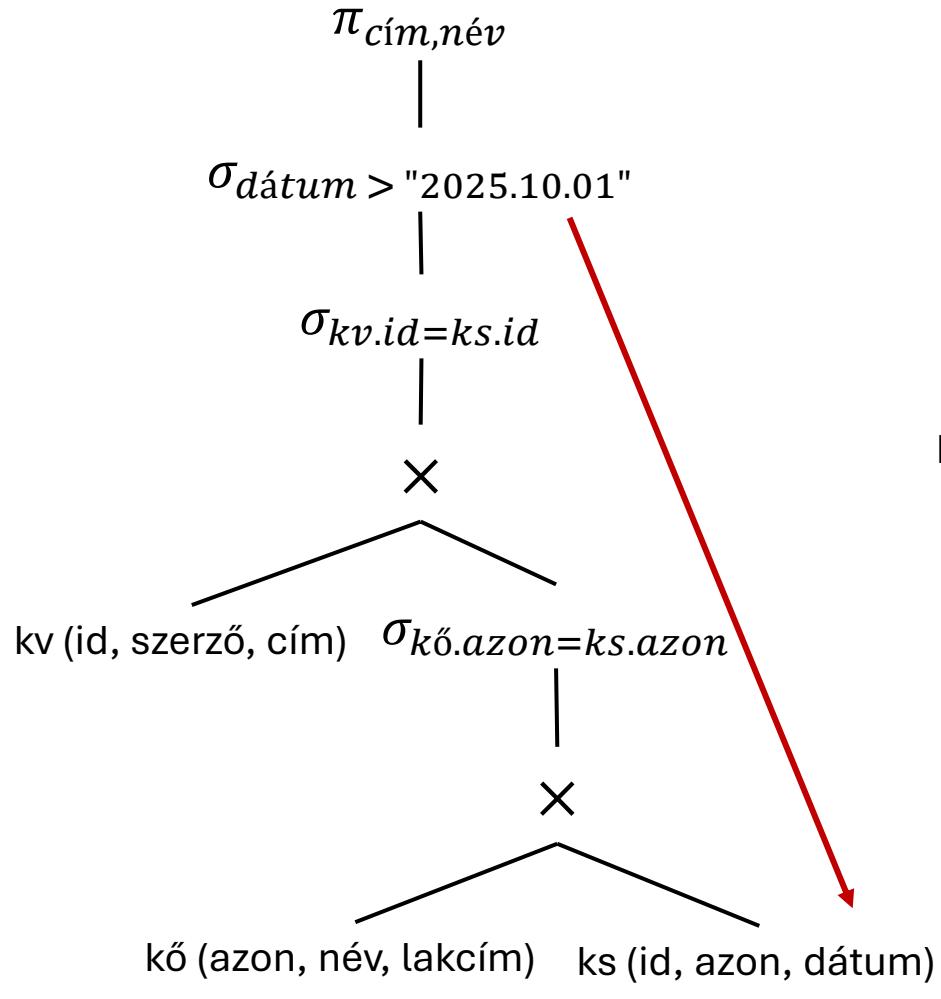
Optimalizáció: kiválasztás felbontása



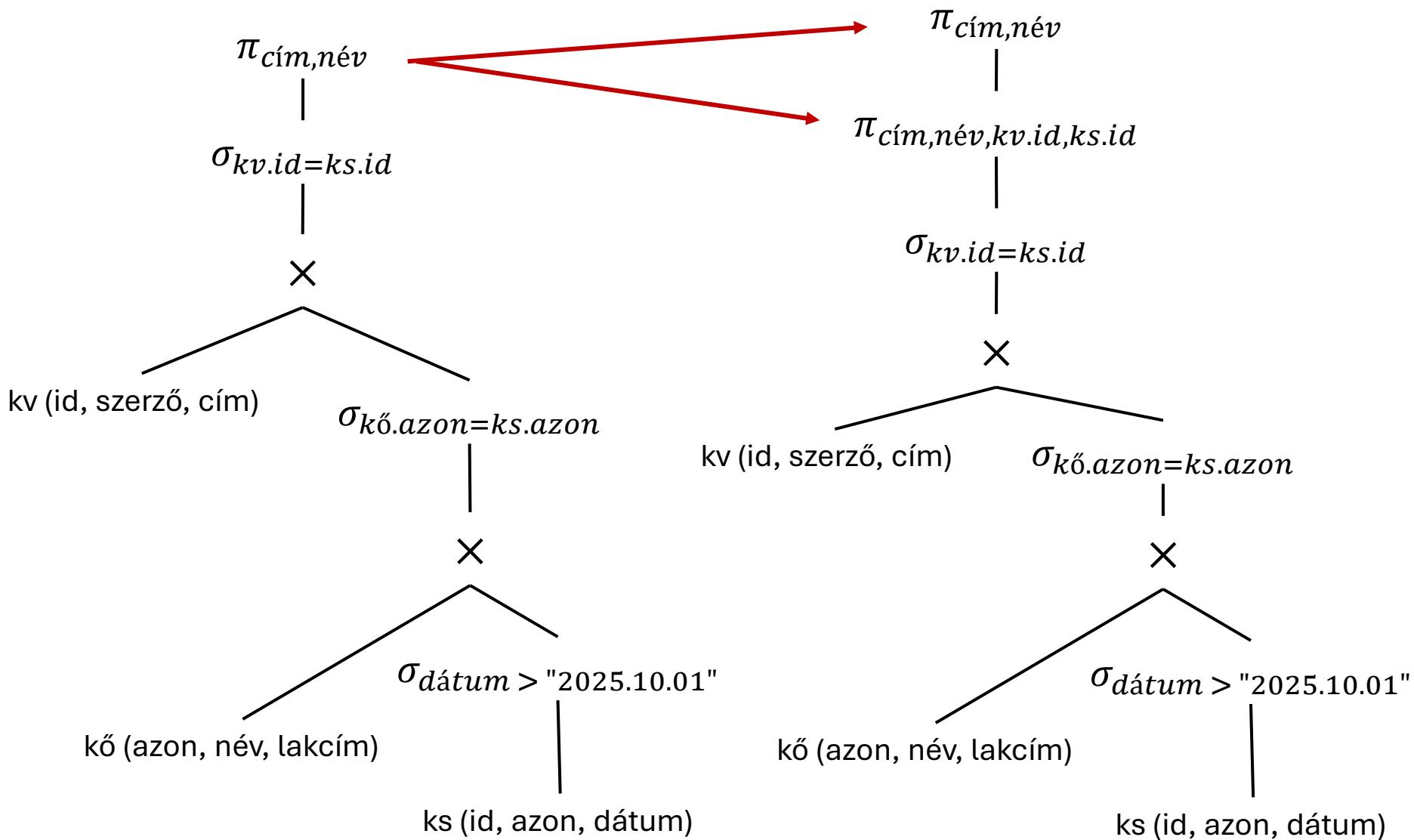
Optimalizáció: kiválasztás lejebb vitele



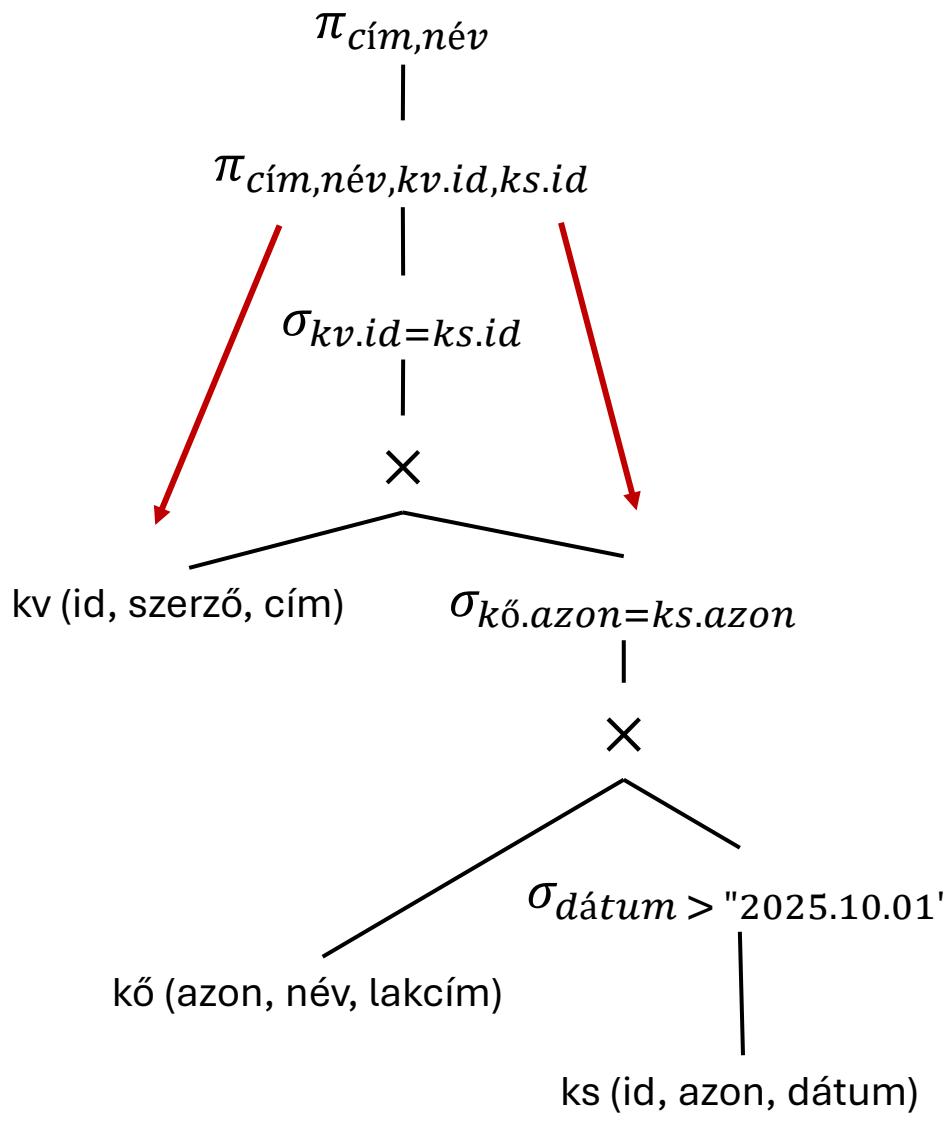
Optimalizáció: kiválasztás lejjebb vitele



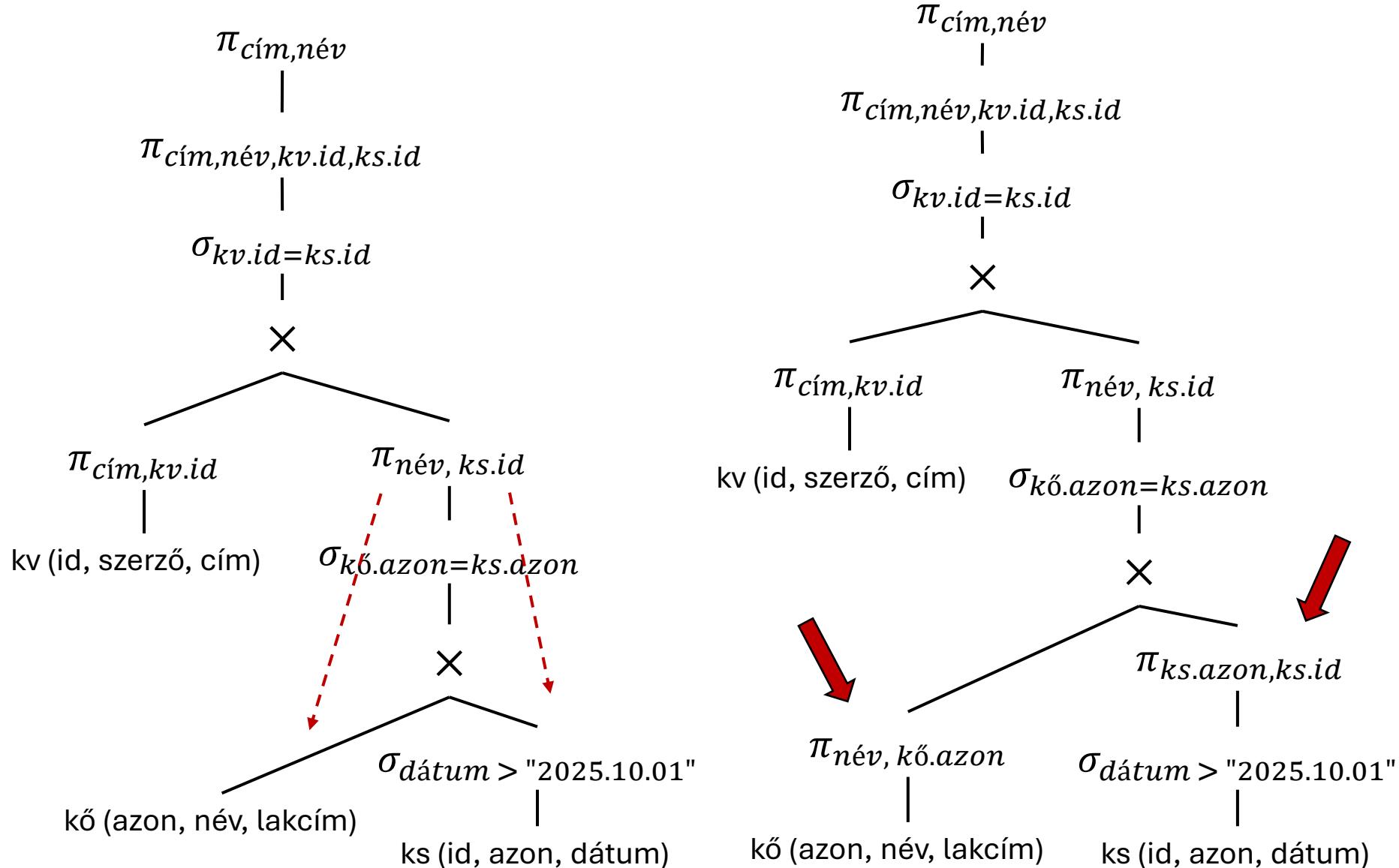
Optimalizáció: vetítések bővítése



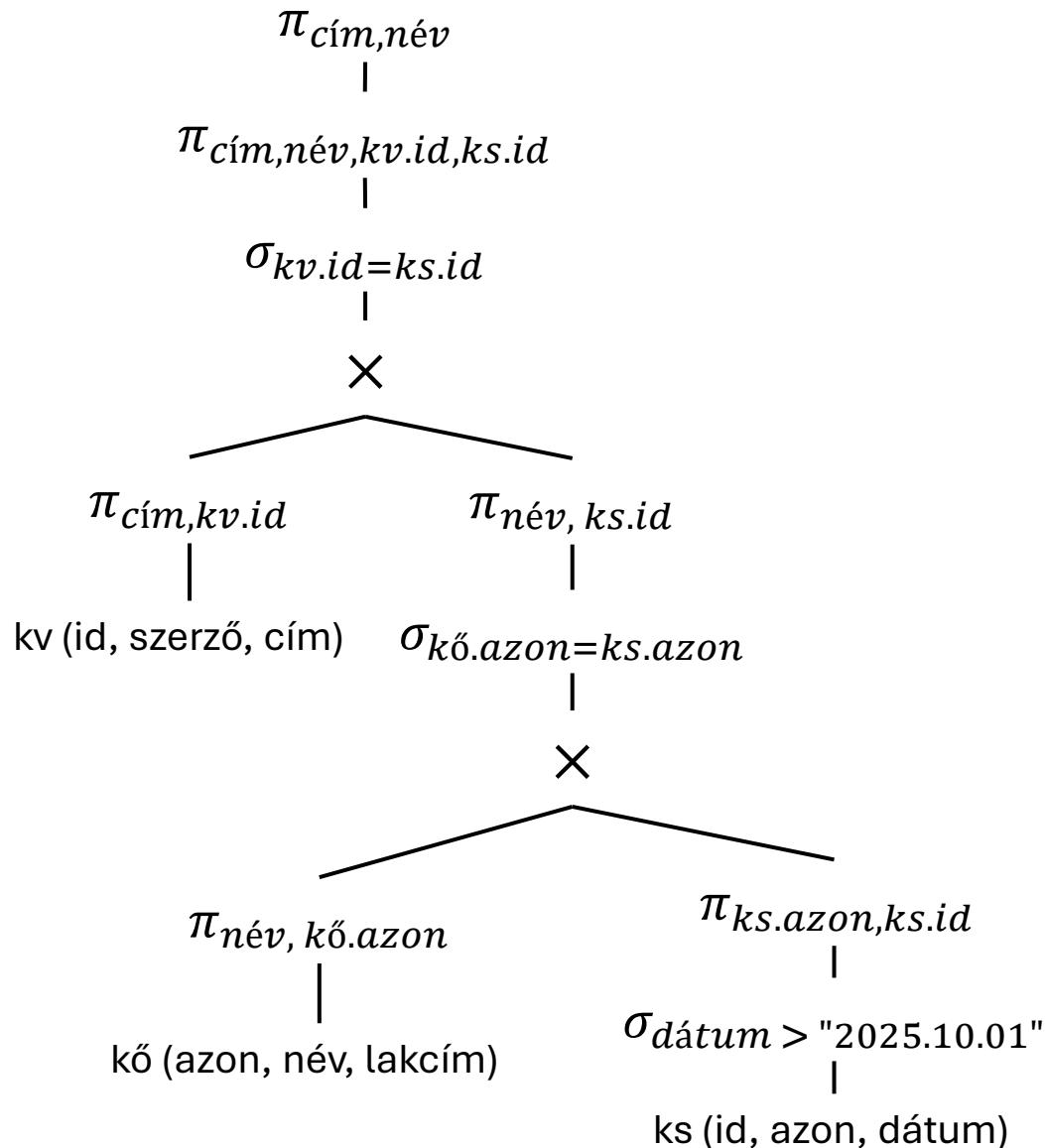
Optimalizáció: vetítések lejebb vitele



Optimalizáció: vetítés bővítése és lejjebb vitele



Optimalizáció: végső logikai terv

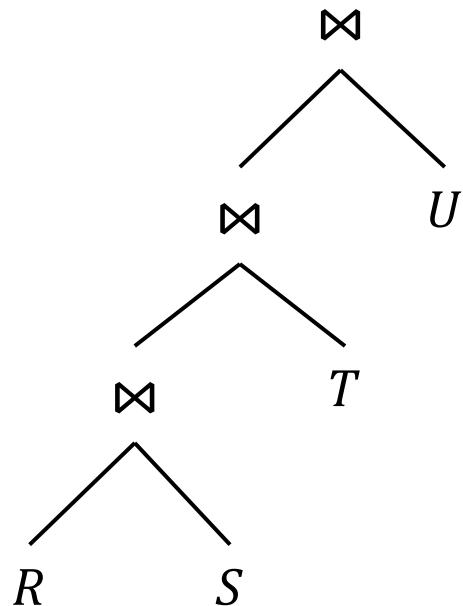


Egytáblás lekérdezések optimalizációja

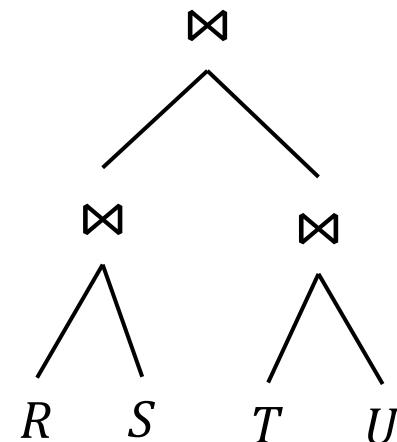
- A logikai terv esetén hajtsuk végre a korábban megadott algoritmust.
- A fizikai tervben, a tábla eléréséhez használjuk a lehető legjobb elérési módot (pl. bináris keresés rendezett értékek esetén; index használata, ha van).

Többtáblás lekérdezések optimalizációja

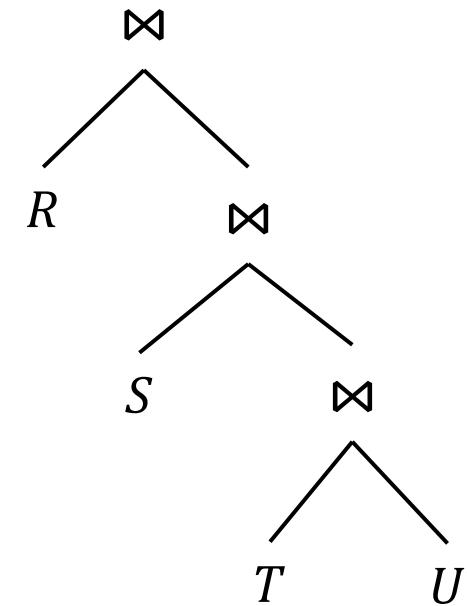
- Több tábla esetén meg kell határoznunk az összekapcsolások sorrendjét.
- Pl. 4 tábla estén (R, S, T, U) három különböző összekapcsolási fát is rajzolhatunk:



Bal-mély fa (left-deep tree)



Bozótszerű (bushy)



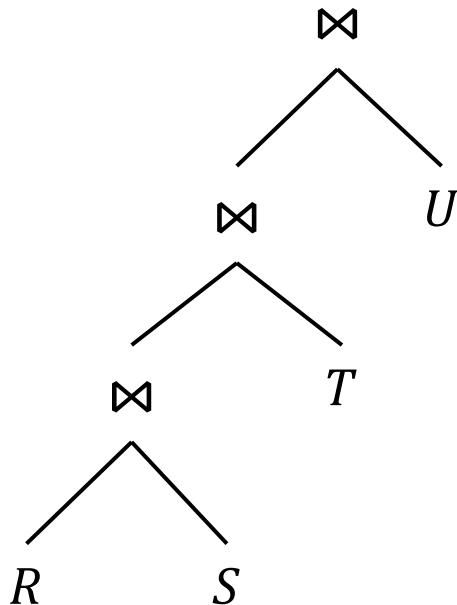
Jobb-mély fa (right-deep tree)

Többtáblás lekérdezések optimalizációja

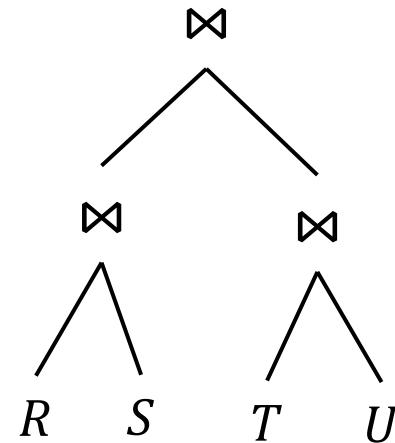
- Az előző példában még a táblák sorrendjét is módosíthatjuk, összesen $n!$ képen. Azaz $3 \cdot 2 \cdot 1 = 6$ féle összekapcsolási fa lehetséges.
- Általánosan n tábla lehetséges összekapcsolási fáinak a számát a következő rekurzív módon kapjuk meg:
 - $T(1) = 1$
 - $T(n) = \sum_{i=1}^{n-1} T(i) * T(n - i)$
 - Már 6 tábla esetén is $30\ 240$ különböző eset lehetséges.
 - Szerencsére nem kell minden esetet megvizsgálnunk.

Összekapcsolási fák alakja

- Csak a **bal-mély** fákat fogjuk figyelembe venni. Előnyei:
 - Így csökken a lehetséges esetek száma.
 - Elkerülhető a köztes relációk lemezre írása.



Bal-mély fa (left-deep tree)



Bozótszerű (bushy)



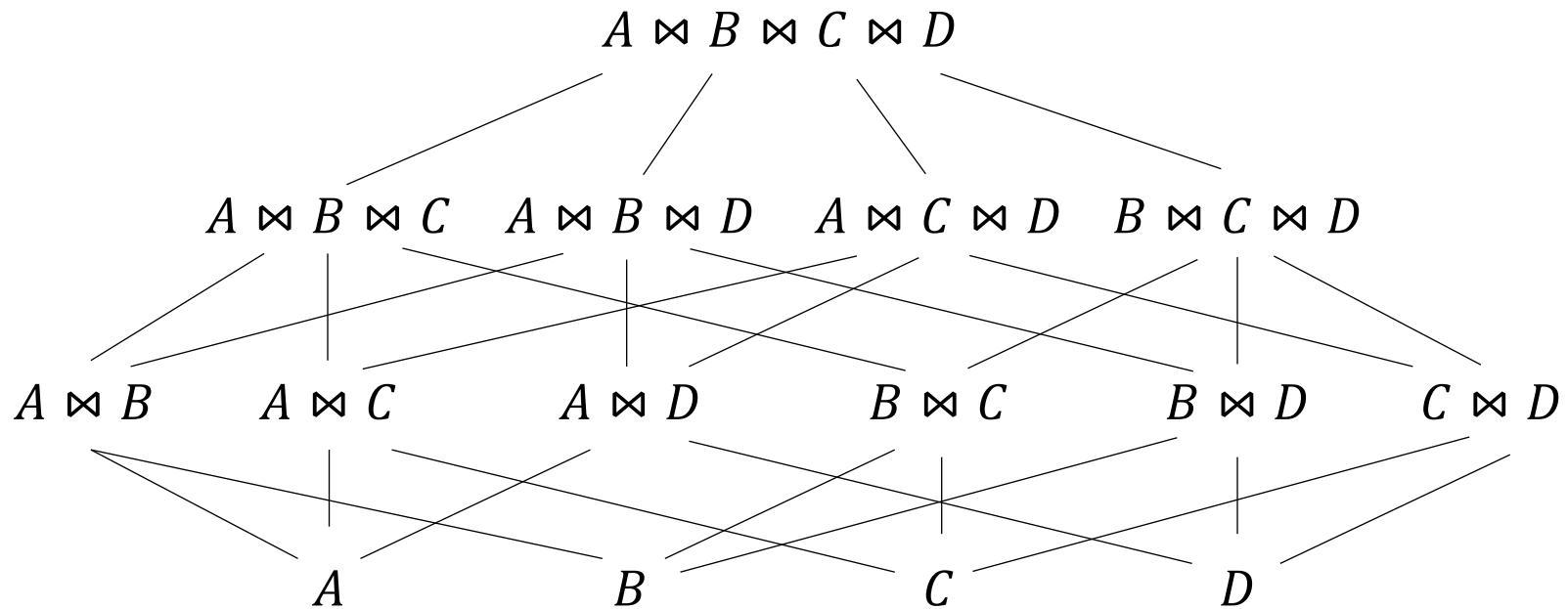
Összekapcsolási sorrend kiválasztása

- A sorrend kiválasztását **dinamikus programozási feladatként** értelmezhetjük.
- Pl. ahhoz, hogy megkapjuk k reláció legjobb összekapcsolást, ki kell választanunk a minimumot a $k-1$ reláció legjobb összekapcsolása és a k .-adik reláció összekapcsolásának a költségei közül.
- $\text{BestPlan}(A,B,C,D) = \min($
 - $\text{BestPlan}(A,B,C) \bowtie D,$
 - $\text{BestPlan}(A,C,D) \bowtie B,$
 - $\text{BestPlan}(B,C,D) \bowtie A,$
 - $\text{BestPlan}(A,B,D) \bowtie C)$



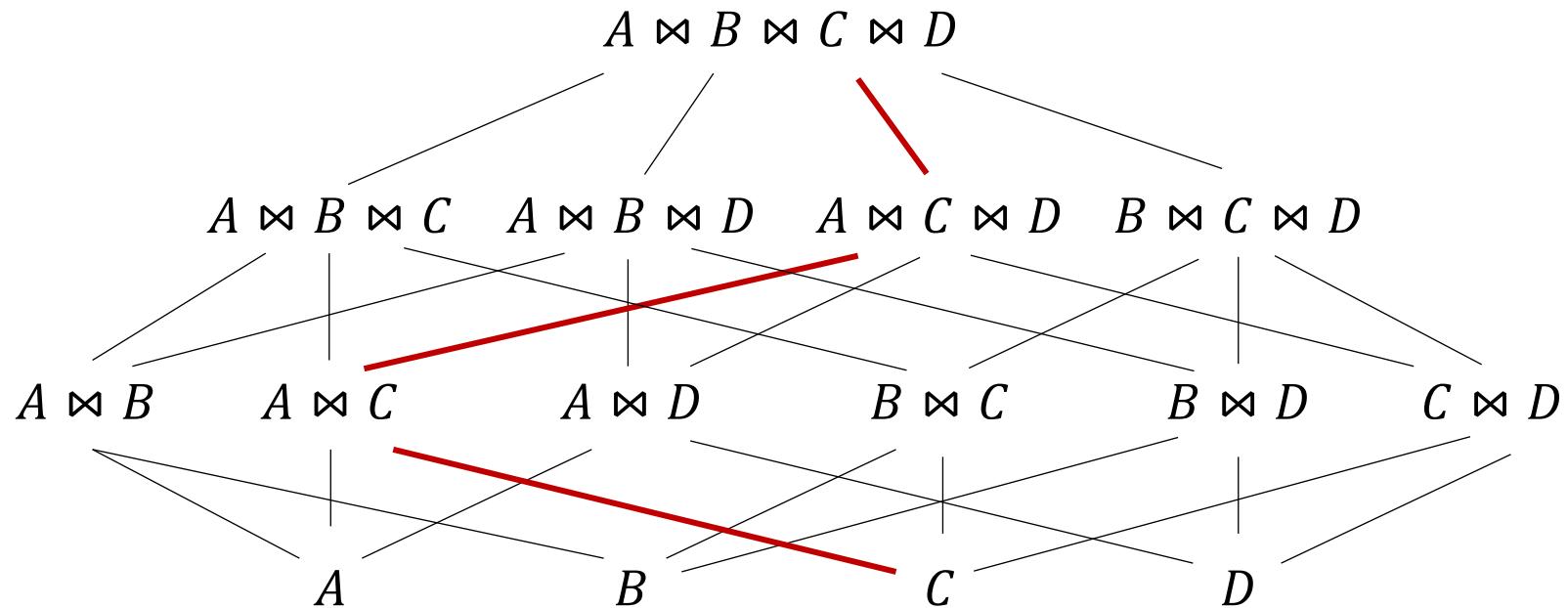
Összekapcsolási sorrend kiválasztása

- **Alulról-felfelé optimalizálás:** egy utat keresünk a gráfban a levelektől a gyökérig.
- Az egyes éleknél még azt is el kell döntenünk, hogy milyen összekapcsolási algoritmust használunk (nested loops, hash join, sort-merge join).



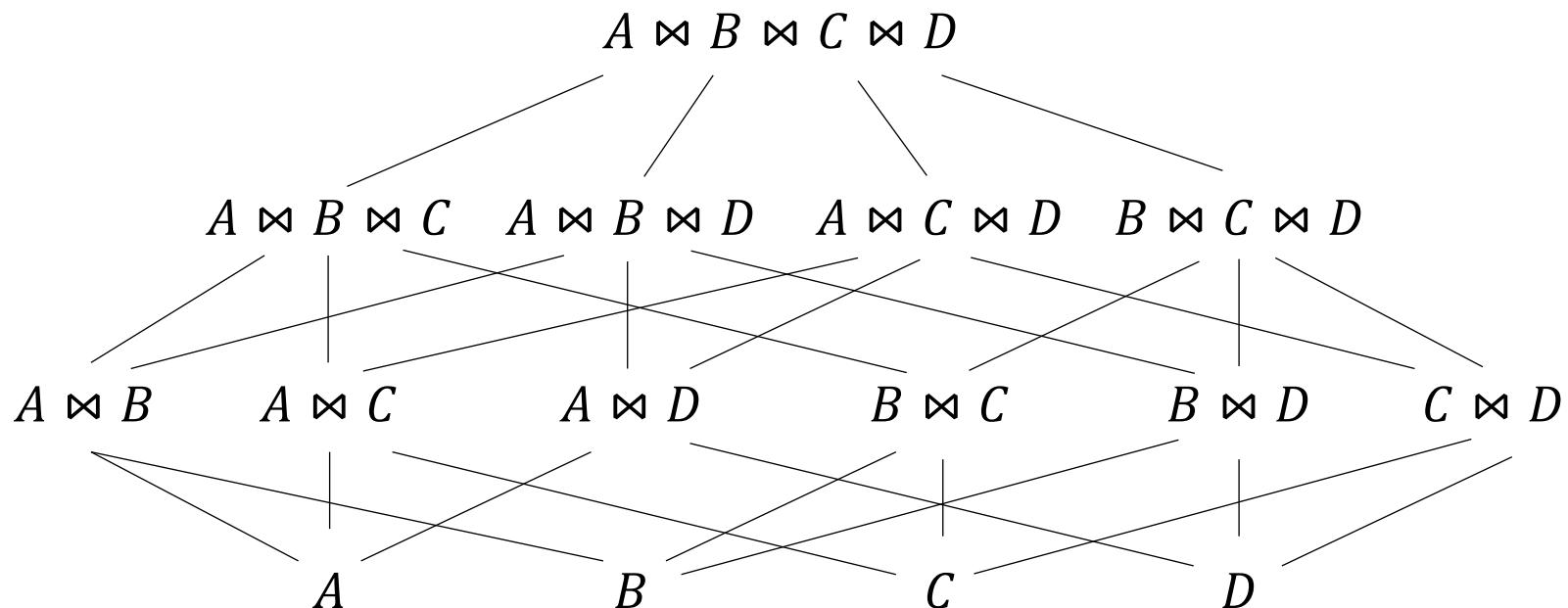
Egy mohó algoritmus

- Egyszerű **költségmodell**: az összekapcsolások során csak a keletkezett relációk **méreteit** vegyük figyelembe és minden a legkisebbet válasszuk.
- Első lépés: válasszuk ki azt a reláció párt, amelyek összekapcsolása a legjobb.
- További lépések: a bal oldali reláció lesz a korábbi összekapcsolás eredménye, a jobb oldalra pedig válasszuk azt, amelyikkel ismét minimális lesz a költség.



Selinger-féle optimalizálás

- Az út kiválasztásakor vegyük figyelembe az „érdekes” rendezettségű összekapcsolásokat. Ezek olyan rendezettségek, amelyek egy későbbi lépésben hasznosak lehetnek (pl. a végén rendeznünk kell az eredményt).
- Tehát ne csak egy utat nézzünk meg, hanem az „érdekes” rendezettségű utakat is, a végén pedig válasszuk azt, amely költsége jobb lesz.



Haladó témák, amiket nem érintünk

- Alkérdések optimalizációja.
 - Átírás összekapcsolásra.
 - Ideiglenes táblák használata.
- Kifejezések átírása.
 - Pl. $1=0$ helyett *false*.
- Top-K lekérdezések.
- Módosítások optimalizálása.
 - Halloween probléma (IBM, System R, 1976).
 - Amikor egy módosítás megváltoztatja egy rekord fizikai helyét, ezért az adatok beolvasása során többször is találkozunk vele (rendezett fájlszervezésnél vagy indexelt keresésnél bukkanhat fel).
 - Megoldás: jegyezzük fel a módosított rekordok azonosítóját.



Gyakorlati megjegyzések

- Az összekapcsolás eredményének a becslését az előző előadáson vizsgáltuk és felhasználtuk hozzá a képméretet: $V(R, A)$ vagy $I_A(R)$.
- A gyakorlatban a képméretet **statisztikákból** becsüljük. Az egyes oszlopokhoz **hisztogram** készül, amely segít megbecsülni az egyes értékek előfordulási gyakoriságát.
- A statisztikák gyakran mintavételezzel készülnek, hogy ne legyen túl költséges. Ezeket a statisztikákat manuálisan is frissíthetjük.
- Az fizikai terveket a rendszerek általában elmentik. Ha valaki ugyanazt a lekérdezést újra futtatná, akkor nem kell az optimalizációt mindig újra elvégezni, elő lehet venni a már elkészült tervet.



Összefoglalás

- Az SQL lekérdezésekkel (az elemzések után) logikai, majd fizikai lekérdezési terv készül.
- A logikai tervet **szabály- és költség** alapú módon is optimalizáljuk:
 - Szabály alapú: ekvivalencia szabályok használatával módosítjuk a kifejezésfát.
 - Költség alapú: a kimeneti méretek becslését felhasználva választjuk ki az összekapcsolások sorrendjét.
- A **fizikai tervhez** kiválasztjuk a megfelelő elérési módot (pl. indexek), összekapcsolási algoritmusokat és más fizikai operátorokat a műveleti költségek és kimeneti méretek becslése alapján.
- A költségeket és kimeneti méreteket az adatbázisban találhat statisztikák alapján becsülhetjük.



Tankönyv fejezetek

- Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom:
Adatbázisrendszerek megvalósítása
 - 7. fejezet: A lekérdezésfordító
- Silberschatz, Korth, & Sudarshan: **Database System Concepts**
 - Chapter 16. Query Optimization

