# Lekérdezések optimalizálása

**CÉL**: A lekérdezéseket gyorsabbá akarjuk tenni a táblákra vonatkozó paraméterek, statisztikák, indexek ismeretében és általános érvényű tulajdonságok, heurisztikák segítségével.

Például, hogyan, milyen procedúrával értékeljük ki az alábbi SQL (deklaratív) lekérdezést?

**Legyen adott R(A,B,C) és S(C,D,E). Melyek azok az R.B és S.D értékek azokban az R, illetve S táblabeli sorokban, amely sorokban R.A='c' és S.E=2 és R.C=S.C?**

**Ugyanez SQL-ben**:

**Select B,D**
**From R,S**
**Where R.A = 'c' and S.E = 2 and R.C=S.C;**

# Lekérdezések optimalizálása

R

| A | B | C |
|---|---|---|
| a | 1 | 10 |
| b | 1 | 20 |
| c | 2 | 10 |
| d | 2 | 35 |
| e | 3 | 45 |

S

| C | D | E |
|---|---|---|
| 10 | x | 2 |
| 20 | y | 2 |
| 30 | z | 2 |
| 40 | x | 1 |
| 50 | y | 3 |

**A lekérdezés eredménye:**

| B | D |
|---|---|
| 2 | x |

# Lekérdezések optimalizálása

Hogy számoljuk ki tetszőleges tábla esetén az eredményt?

Egy lehetséges terv

- Vegyük a két tábla szorzatát!
- Válasszuk ki a megfelelő sorokat!
- Hajtsuk végre a vetítést!

- Ez a direktszorzaton alapuló összekapcsolás.
- Oracleben: NESTED LOOP.
- Nagyon költséges!

# Lekérdezések optimalizálása

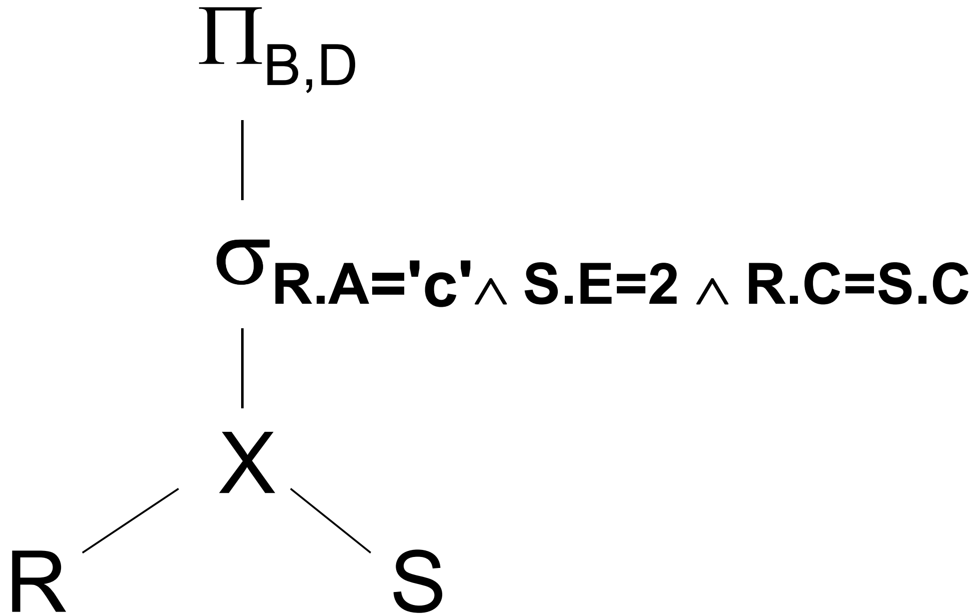
RXS

| R.A | R.B | R.C | S.C | S.D | S.E |
|---|---|---|---|---|---|
| a | 1 | 10 | 10 | x | 2 |
| a | 1 | 10 | 20 | y | 2 |
| . | | | | | |
| . | | | | | |
| c | 2 | 10 | 10 | x | 2 |
| . | | | | | |
| . | | | | | |

Ez a sor kell!

# Lekérdezések optimalizálása

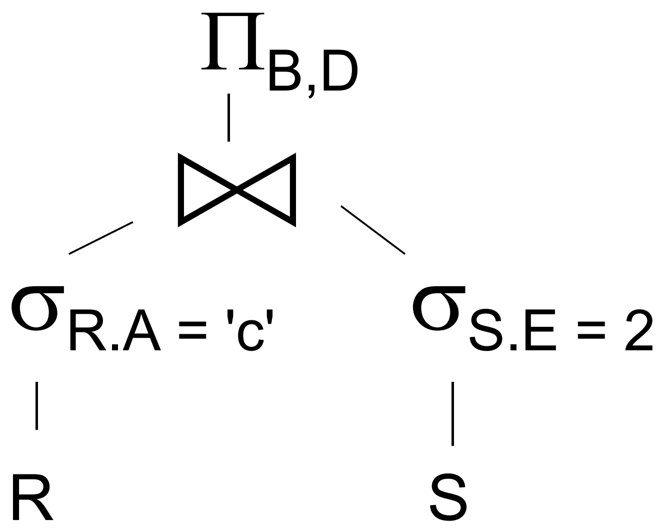
## Ugyanez a terv relációs algebrában:




$$\Pi_{B,D} [\, \sigma_{R.A='c' \wedge S.E=2 \wedge R.C = S.C} \, (RXS)]$$

# Lekérdezések optimalizálása

**Egy másik lehetséges kiszámítási javaslat:**



# Lekérdezések optimalizálása

R

| A | B | C |
|---|---|---|
| a | 1 | 10 |
| b | 1 | 20 |
| c | 2 | 10 |
| d | 2 | 35 |
| e | 3 | 45 |

$\sigma$ (R)

| A | B | C |
|---|---|---|
| c | 2 | 10 |

$\sigma$(S)

| C | D | E |
|---|---|---|
| 10 | x | 2 |
| 20 | y | 2 |
| 30 | z | 2 |

S

| C | D | E |
|---|---|---|
| 10 | x | 2 |
| 20 | y | 2 |
| 30 | z | 2 |
| 40 | x | 1 |
| 50 | y | 3 |

$\bowtie$

$\Pi_{B,D}$

Ugyanazt számolja ki!

| B | D |
|---|---|
| 2 | x |

# Lekérdezések optimalizálása

Használjuk ki az R.A és S.C oszlopokra készített **indexeket**:

(1) Az R.A index alapján keressük meg az R azon sorait, amelyekre R.A = 'c'!

(2) Minden megtalált R.C értékhez az S.C index alapján keressük meg az S-ből az ilyen értékű sorokat!

(3) Válasszuk ki a kapott S-beli sorok közül azokat, amelyekre S.E = 2!

(4) Kapcsoljuk össze az R és S így kapott sorait, és végül vetítsünk a B és D oszlopokra.

Lekérdezések optimalizálása

R

| A | B | C |
|---|---|---|
| a | 1 | 10 |
| b | 1 | 20 |
| c | 2 | 10 |
| d | 2 | 35 |
| e | 3 | 45 |

A ='c'

I1

<c,2,10>

C

I2

<10,x,2>

Ellenőrzés:
=2?

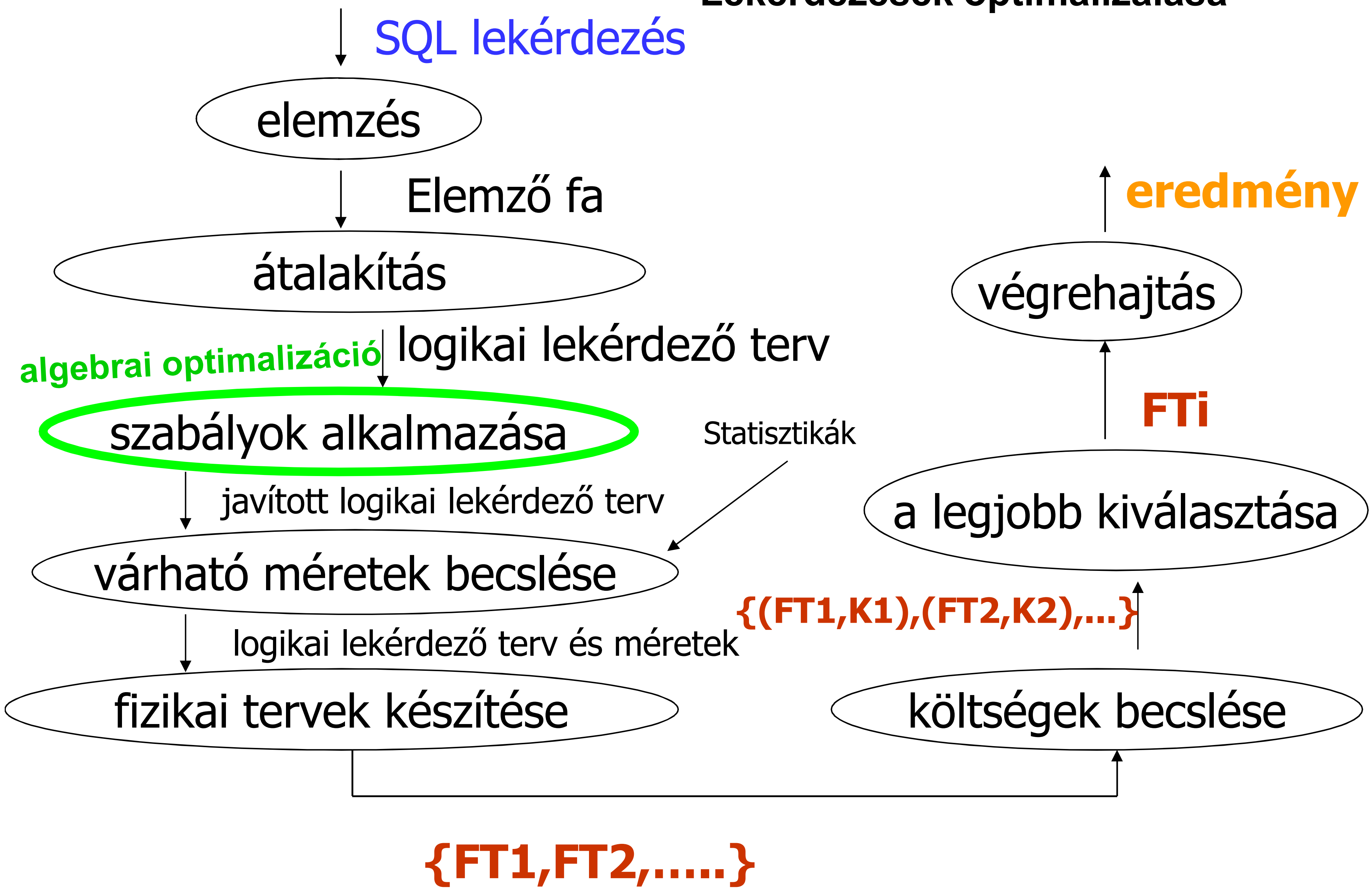
Eredmény: <2,x>

S

| C | D | E |
|---|---|---|
| 10 | x | 2 |
| 20 | y | 2 |
| 30 | z | 2 |
| 40 | x | 1 |
| 50 | y | 3 |

# INDEXES ÖSSZEKAPCSOLÁS

## Lekérdezések optimalizálása

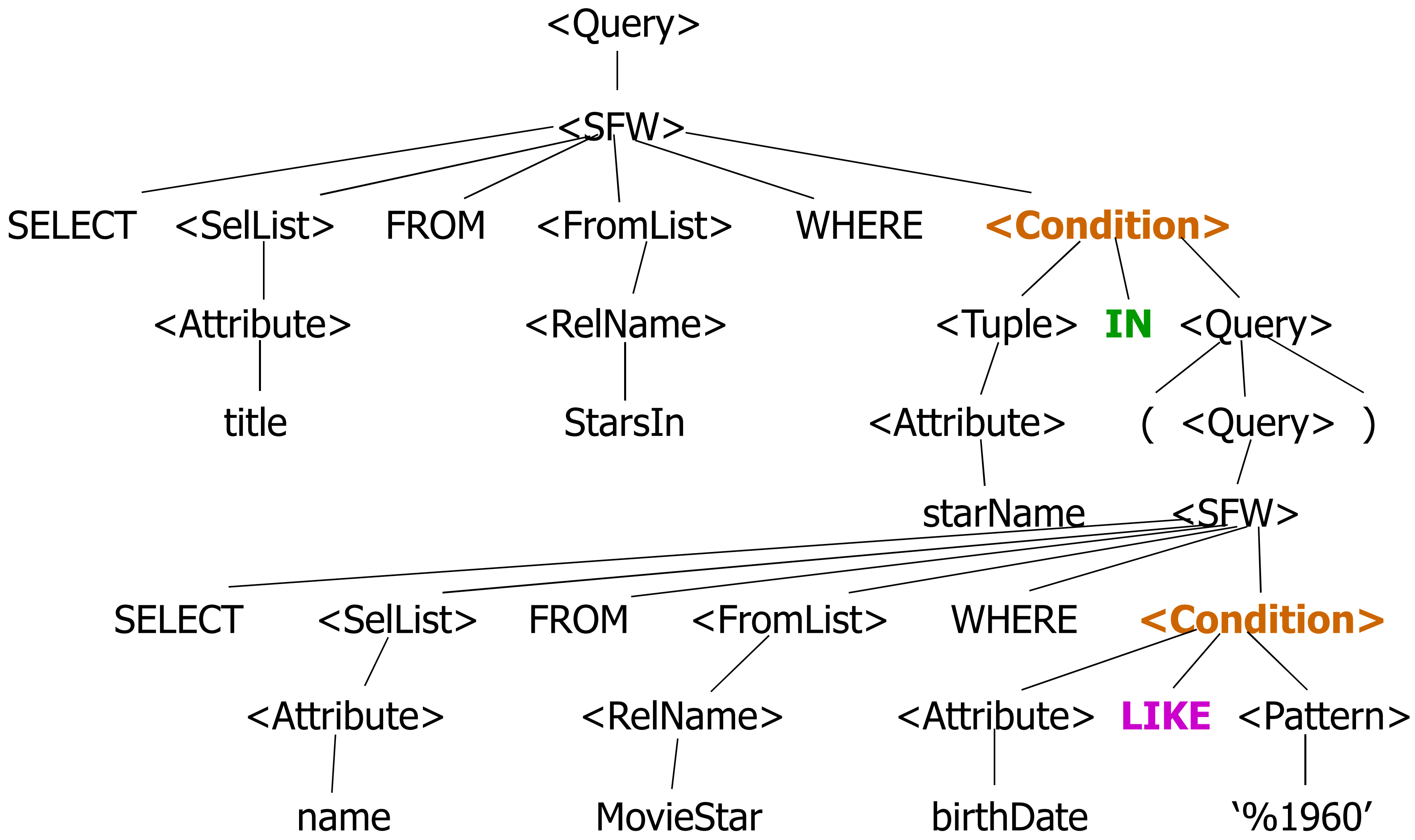


# Példa: SQL lekérdezés

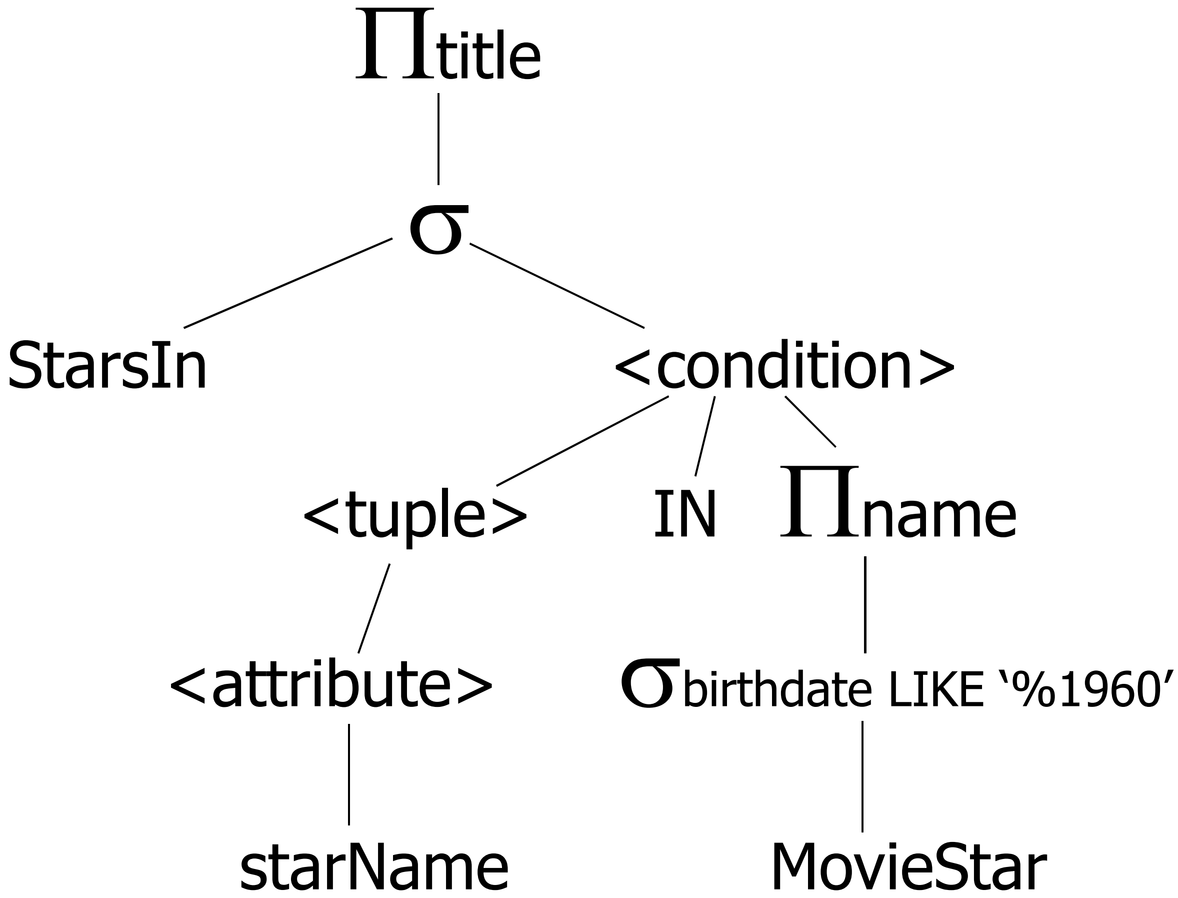
```
SELECT title
FROM StarsIn
WHERE starName IN (
        SELECT name
        FROM MovieStar
        WHERE birthdate LIKE ‘%1960’
);
```

**Milyen filmekben szerepeltek 1960-as születésű színészek?**

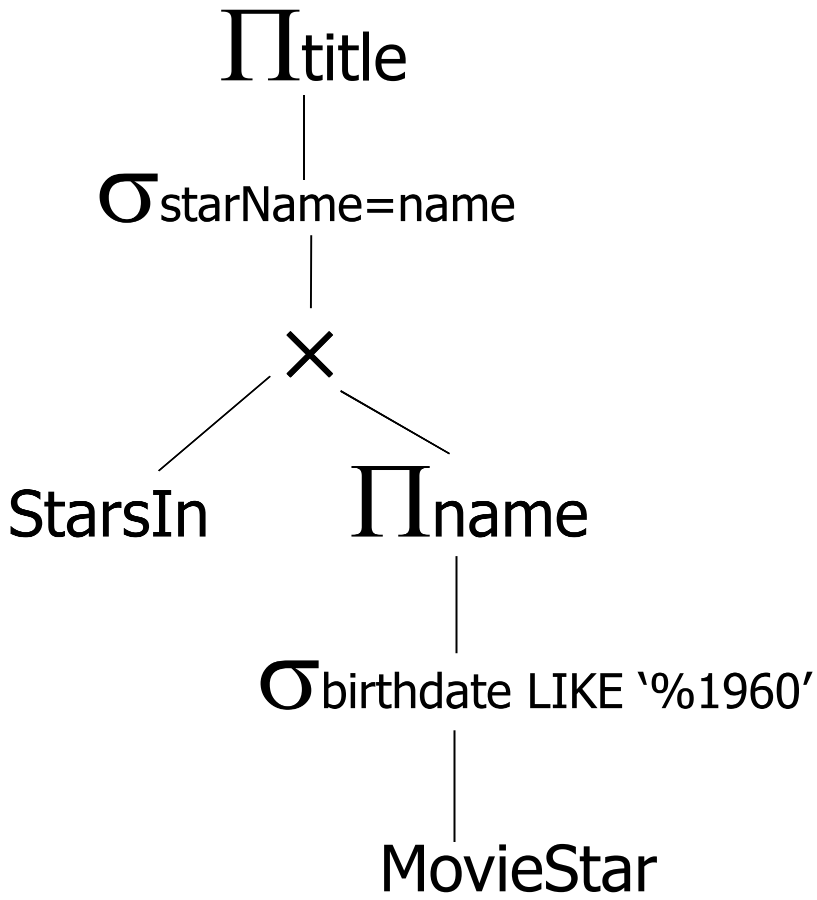
# Elemzőfa:   Parse Tree



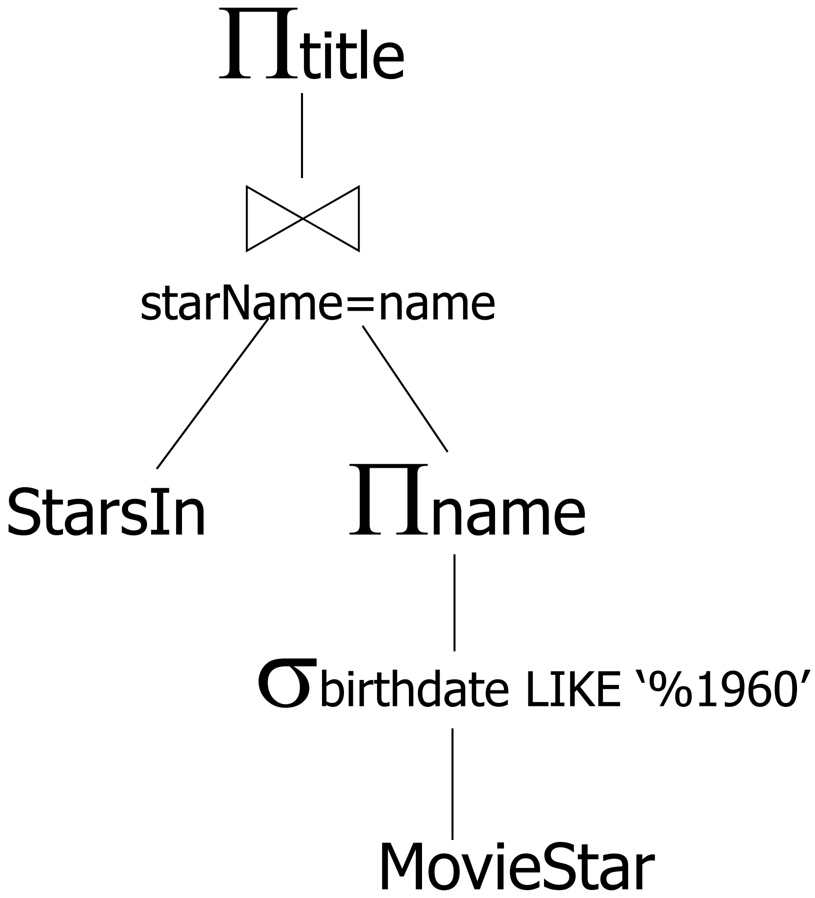
# Ugyanez relációs algebrában:



# Átalakított logikai lekérdező terv



# Továbbjavított logika lekérdező terv



# Algebrai optimalizáció

- **Cél:** a relációs algebrai kifejezéseket minél gyorsabban akarjuk kiszámolni.
- **Költségmodell:** a kiszámítás költsége arányos a relációs algebrai kifejezés részkifejezéseinek megfelelő relációk tárolási méreteinek összegével.
- **Módszer:** a műveleti tulajdonságokon alapuló ekvivalens átalakításokat alkalmazunk, hogy várhatóan kisebb méretű relációk keletkezzenek.
- **Az eljárás heurisztikus**, tehát nem az argumentum relációk valódi méretével számol.
- **Az eredmény nem egyértelmű:** Az átalakítások sorrendje nem determinisztikus, így más sorrendben végrehajtva az átalakításokat más végeredményt kaphatunk, de mindegyik általában jobb költségű, mint amiből kiindultunk.
- **Megjegyzés:** Mivel az SQL bővebb, mint a relációs algebra, ezért az optimalizálást bővített relációs algebrára is meg kell adni, de először a hagyományos algebrai kifejezéseket vizsgáljuk.

# Algebrai optimalizáció

- A relációs algebrai kifejezést **gráffal** ábrázoljuk.
- **Kifejezésfa:**
  - a **nem levél csúcsok**: a relációs algebrai műveletek:
    - unáris ( $\sigma, \Pi, \rho$ ) – egy gyereke van
    - bináris ($-, \cup, \times$) – két gyereke van (bal oldali az első, jobb oldali a második argumentumnak felel meg)
  - a **levél csúcsok**: konstans relációk vagy relációs változók

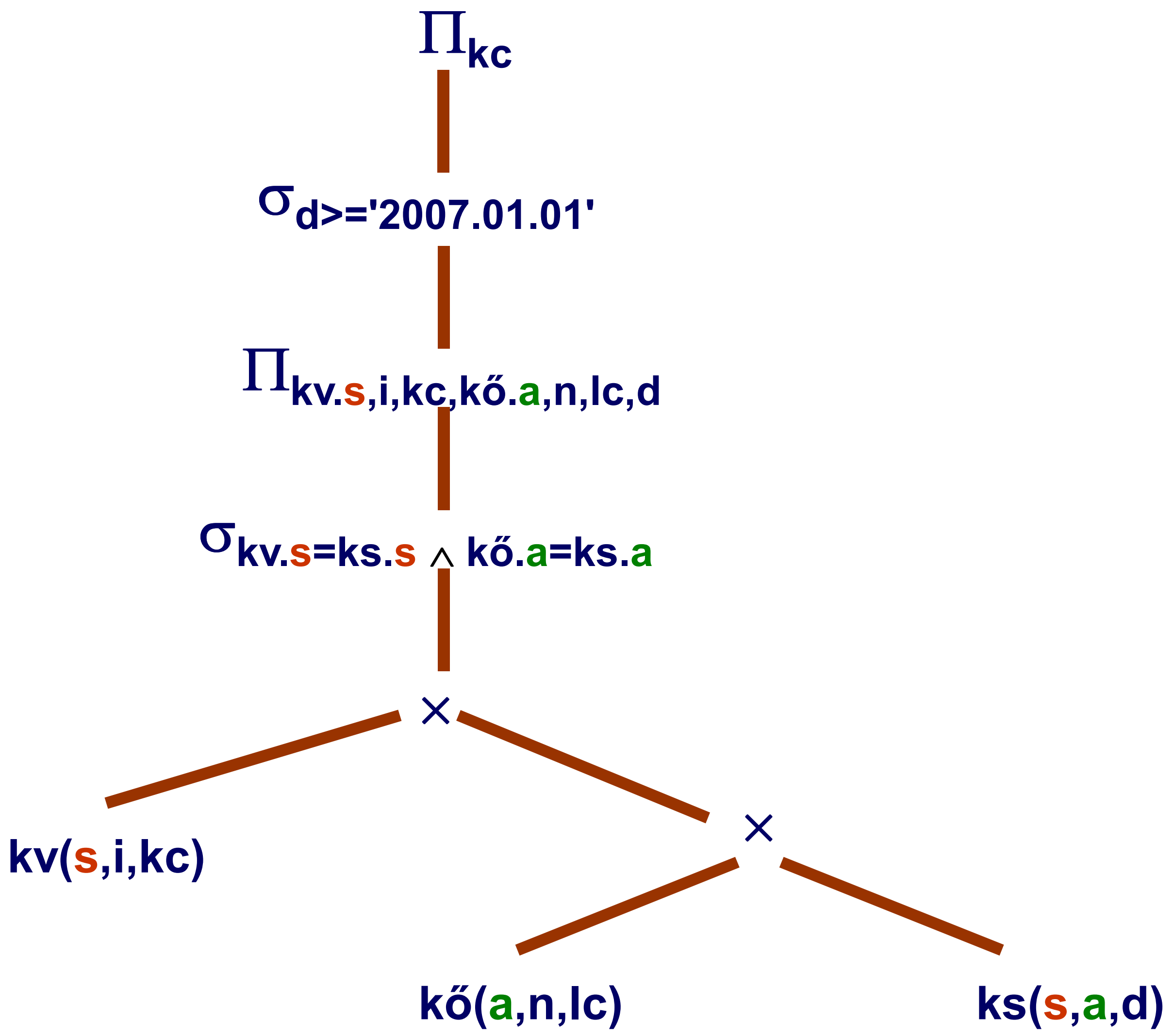
# Algebrai optimalizáció

- könyv(sorszám,író,könyvcím)
  - **kv(s,i,kc)**
- kölcsönző(azonosító,név,lakcím)
  - **kő(a,n,lc)**
- kölcsönzés(sorszám,azonosító,dátum)
  - **ks(s,a,d)**
- Milyen című könyveket kölcsönöztek ki 2007-től kezdve?
- $\Pi_{\mathbf{kc}}(\sigma_{\mathbf{d>='2007.01.01'}}(\mathbf{kv}|\times|\mathbf{kő}|\times|\mathbf{ks}))$
- Az összekapcsolásokat valamilyen sorrendben kifejezzük az alapműveletekkel:

$$\Pi_{\mathbf{kc}}(\sigma_{\mathbf{d>='2007.01.01'}}(\Pi_{\mathbf{kv.s,i,kc,kő.a,n,lc,d}}(\sigma_{\mathbf{kv.s=ks.s} \wedge \mathbf{kő.a=ks.a}}(\mathbf{kv}\times(\mathbf{kő}\times\mathbf{ks})))))$$

# Algebrai optimalizáció

$$\Pi_{kc}(\sigma_{d>='2007.01.01'}(\Pi_{kv.s,i,kc,kő.a,n,lc,d}(\sigma_{kv.s=ks.s \wedge kő.a=ks.a}(kv\times(kő\times ks)))))$$



# Algebrai optimalizáció

- E1(r1,...,rk) és E2(r1,...,rk) **relációs algebrai kifejezések ekvivalensek** (E1 $\cong$ E2 ), ha tetszőleges r1,...,rk relációkat véve E1(r1,...,rk)=E2(r1,...,rk).
- 11 szabályt adunk meg. A szabályok olyan állítások, amelyek kifejezések ekvivalenciáját fogalmazzák meg. Bizonyításuk könnyen végiggondolható.
- Az állítások egy részében a kifejezések szintaktikus helyessége egyben elégséges feltétele is az ekvivalenciának.

**1. Kommutativitás (szorzás, természetes összekapcsolás, téta-összekapcsolás)**

- **E1 $\times$ E2 $\cong$ E2 $\times$ E1**
- **E1 $|\times|$ E2 $\cong$ E2 $|\times|$ E1**
- **E1 $|\times|_{\Theta}$ E2 $\cong$ E2 $|\times|_{\Theta}$ E1**

# Algebrai optimalizáció

2. **Asszociativitás (szorzás, természetes összekapcsolás, téta-összekapcsolás)**
   - $(E1 \times E2) \times E3 \cong E2 \times (E1 \times E3)$
   - $(E1 \,|\!\times\!|\, E2) \,|\!\times\!|\, E3 \cong E1 \,|\!\times\!|\, (E2 \,|\!\times\!|\, E3)$
   - $(E1 \,\underset{\Theta}{|\!\times\!|}\, E2) \,\underset{\Theta}{|\!\times\!|}\, E3 \cong E1 \,\underset{\Theta}{|\!\times\!|}\, (E2 \,\underset{\Theta}{|\!\times\!|}\, E3)$
3. **Vetítések összevonása, bővítése**
   - Legyen $\underline{A}$ és $\underline{B}$ két részhalmaza az E reláció oszlopainak úgy, hogy $\underline{\mathbf{A}} \subseteq \underline{\mathbf{B}}$.
   - Ekkor $\Pi_{\underline{A}}(\Pi_{\underline{B}}(E)) \cong \Pi_{\underline{A}}(E)$.
4. **Kiválasztások felcserélhetősége, felbontása**
   - Legyen F1 és F2 az E reláció oszlopain értelmezett kiválasztási feltétel.
   - Ekkor $\sigma_{F1 \wedge F2}(E) \cong \sigma_{F1}(\sigma_{F2}(E)) \cong \sigma_{F2}(\sigma_{F1}(E))$.

# Algebrai optimalizáció

5. **Kiválasztás és vetítés felcserélhetősége**

- Legyen F az E relációnak csak az A oszlopain értelmezett kiválasztási feltétel.

a) - Ekkor $\Pi_{\underline{A}}(\sigma_F(E)) \cong \sigma_F(\Pi_{\underline{A}}(E))$.

- Általánosabban: Legyen F az E relációnak csak az $\underline{A} \cup \underline{B}$ oszlopain értelmezett kiválasztási feltétel, ahol $\underline{A} \cap \underline{B} = \emptyset$.

b) - Ekkor $\Pi_{\underline{A}}(\sigma_F(E)) \cong \Pi_{\underline{A}}(\sigma_F(\Pi_{\underline{A} \cup \underline{B}}(E)))$.

6. **Kiválasztás és szorzás felcserélhetősége**

- Legyen F az E1 reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel.

a) - Ekkor $\sigma_F(E1 \times E2) \cong \sigma_F(E1) \times E2$.

- Speciálisan: Legyen i=1,2 esetén Fi az Ei reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel, legyen továbbá $F = F1 \wedge F2$.

b) - Ekkor $\sigma_F(E1 \times E2) \cong \sigma_{F1}(E1) \times \sigma_{F2}(E2)$.

- Általánosabban: Legyen F1 az E1 reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel, legyen F2 az E1×E2 reláció oszlopainak egy részhalmazán értelmezett kiválasztási feltétel, úgy hogy mindkét sémából legalább egy oszlop szerepel benne, legyen továbbá $F = F1 \wedge F2$.

c) - Ekkor $\sigma_F(E1 \times E2) \cong \sigma_{F2}(\sigma_{F1}(E1) \times E2)$.

# Algebrai optimalizáció

7. **Kiválasztás és egyesítés felcserélhetősége**
   - Legyen E1, E2 relációk sémája megegyező, és F a közös sémán értelmezett kiválasztási feltétel.
   - Ekkor $\sigma_F(E1 \cup E2) \cong \sigma_F(E1) \cup \sigma_F(E2)$.
8. **Kiválasztás és kivonás felcserélhetősége**
   - Legyen E1, E2 relációk sémája megegyező, és F a közös sémán értelmezett kiválasztási feltétel.
   - Ekkor $\sigma_F(E1 - E2) \cong \sigma_F(E1) - \sigma_F(E2)$.
9. **Kiválasztás és természetes összekapcsolás felcserélhetősége**
   - Legyen F az E1 és E2 közös oszlopainak egy részhalmazán értelmezett kiválasztási feltétel.
   - Ekkor $\sigma_F(E1 |\times| E2) \cong \sigma_F(E1)\ |\times|\ \sigma_F(E2)$.

# Algebrai optimalizáció

## 10. Vetítés és szorzás felcserélhetősége

- Legyen i=1,2 esetén Ai az Ei reláció oszlopainak egy halmaza, valamint legyen A=A1∪A2.
- Ekkor $\Pi_{\underline{A}}(\mathbf{E1}\times\mathbf{E2}) \cong \Pi_{\underline{A1}}(\mathbf{E1})\times\Pi_{\underline{A2}}(\mathbf{E2})$.

## 11. Vetítés és egyesítés felcserélhetősége

- Legyen E1 és E2 relációk sémája megegyező, és legyen A a sémában szereplő oszlopok egy részhalmaza.
- Ekkor $\Pi_{\underline{A}}(\mathbf{E1}\cup\mathbf{E2}) \cong \Pi_{\underline{A}}(\mathbf{E1})\cup\Pi_{\underline{A}}(\mathbf{E2})$.

- Megjegyzés: A vetítés és kivonás nem cserélhető fel, azaz $\Pi_{\underline{A}}(\mathbf{E1 - E2}) \not\cong \Pi_{\underline{A}}(\mathbf{E1}) - \Pi_{\underline{A}}(\mathbf{E2})$. Például:

E1:

| A | B |
|---|---|
| 0 | 0 |
| 0 | 1 |

E2:

| A | B |
|---|---|
| 0 | 0 |

esetén $\Pi_{\mathbf{A}}(\mathbf{E1 - E2})$:

| A |
|---|
| 0 |

míg

$\Pi_{\mathbf{A}}(\mathbf{E1}) - \Pi_{\mathbf{A}}(\mathbf{E2}) = \emptyset$

# Algebrai optimalizáció

- Az optimalizáló algoritmus a következő **heurisztikus elveken** alapul:

1. **Minél hamarabb szelektáljunk**, hogy a részkifejezések várhatóan kisebb relációk legyenek.
2. A szorzás utáni kiválasztásokból **próbáljunk természetes összekapcsolásokat képezni**, mert az összekapcsolás hatékonyabban kiszámolható, mint a szorzatból történő kiválasztás.
3. **Vonjuk össze az egymás utáni unáris műveleteket** (kiválasztásokat és vetítéseket), és ezekből lehetőleg egy kiválasztást, vagy vetítést, vagy kiválasztás utáni vetítést képezzünk. Így csökken a műveletek száma, és általában a kiválasztás kisebb relációt eredményez, mint a vetítés.
4. **Keressünk közös részkifejezéseket**, amiket így elég csak egyszer kiszámolni a kifejezés kiértékelése során.

# Algebrai optimalizáció

- **Algebrai optimalizációs algoritmus:**
- INPUT: relációs algebrai kifejezés kifejezésfája
- OUTPUT: optimalizált kifejezésfa optimalizált kiértékelése

Hajtsuk végre az alábbi lépéseket a megadott sorrendben:

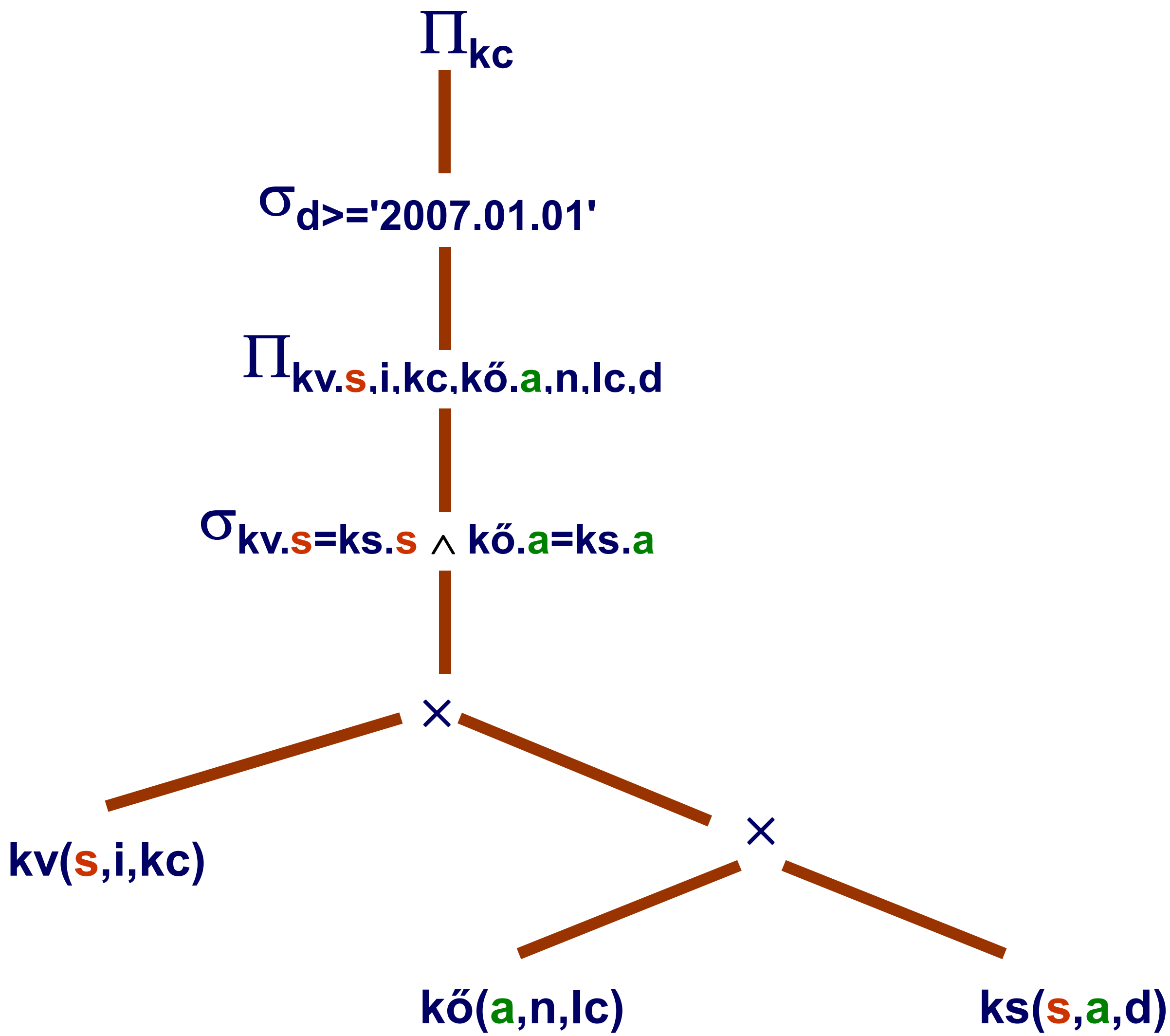
1. **A kiválasztásokat bontsuk fel** a **4. szabály** segítségével:
   - $\sigma_{F1\wedge\ldots\wedge Fn}(E) \cong \sigma_{F1}(\ldots(\sigma_{Fn}(E)))$
2. **A kiválasztásokat** a **4., 5., 6., 7., 8., 9. szabályok** segítségével **vigyük** olyan **mélyre** a kifejezésfában, amilyen mélyre csak lehet.
3. **A vetítéseket** a **3., 5., 10., 11. szabályok** segítségével **vigyük** olyan **mélyre** a kifejezésfában, amilyen mélyre csak lehet. Hagyjuk el a triviális vetítéseket, azaz az olyanokat, amelyek az argumentum reláció összes attribútumára vetítenek.
4. Ha egy relációs változóra vagy konstans relációra közvetlenül egymás után kiválasztásokat vagy vetítéseket alkalmazunk, akkor ezeket a **3., 4., 5. szabályok** segítségével **vonjuk össze egy kiválasztássá, vagy egy vetítéssé, vagy egy kiválasztás utáni vetítéssé, ha lehet** (azaz egy $\Pi.(\sigma.())$ alakú kifejezéssé). **Ezzel megkaptuk az optimalizált kifejezésfát.**
5. A gráfot **a bináris műveletek alapján bontsuk részgráfokra**. Minden részgráf egy bináris műveletnek feleljen meg. A részgráf csúcsai legyenek: a bináris műveletnek **($\cup$, —, $\times$)** megfelelő csúcs és a csúcs felett a következő bináris műveletig szereplő kiválasztások ($\sigma$) és vetítések ($\Pi$). Ha a bináris művelet szorzás ($\times$), és a részgráf equi-joinnak felel meg, és a szorzás valamelyik ága nem tartalmaz bináris műveletet, akkor ezt az ágat is vegyük hozzá a részgráfhoz.
6. Az előző lépésben kapott részgráfok is fát képeznek. **Az optimális kiértékeléshez** ezt a fát értékeljük ki alulról felfelé haladva, tetszőleges sorrendben.

**Megjegyzés**. Az **equi-join** azt jelenti, hogy a kiválasztás feltétele egyenlőség, amely a szorzás két ágának egy-egy oszlopát hasonlítja össze.

# Algebrai optimalizáció

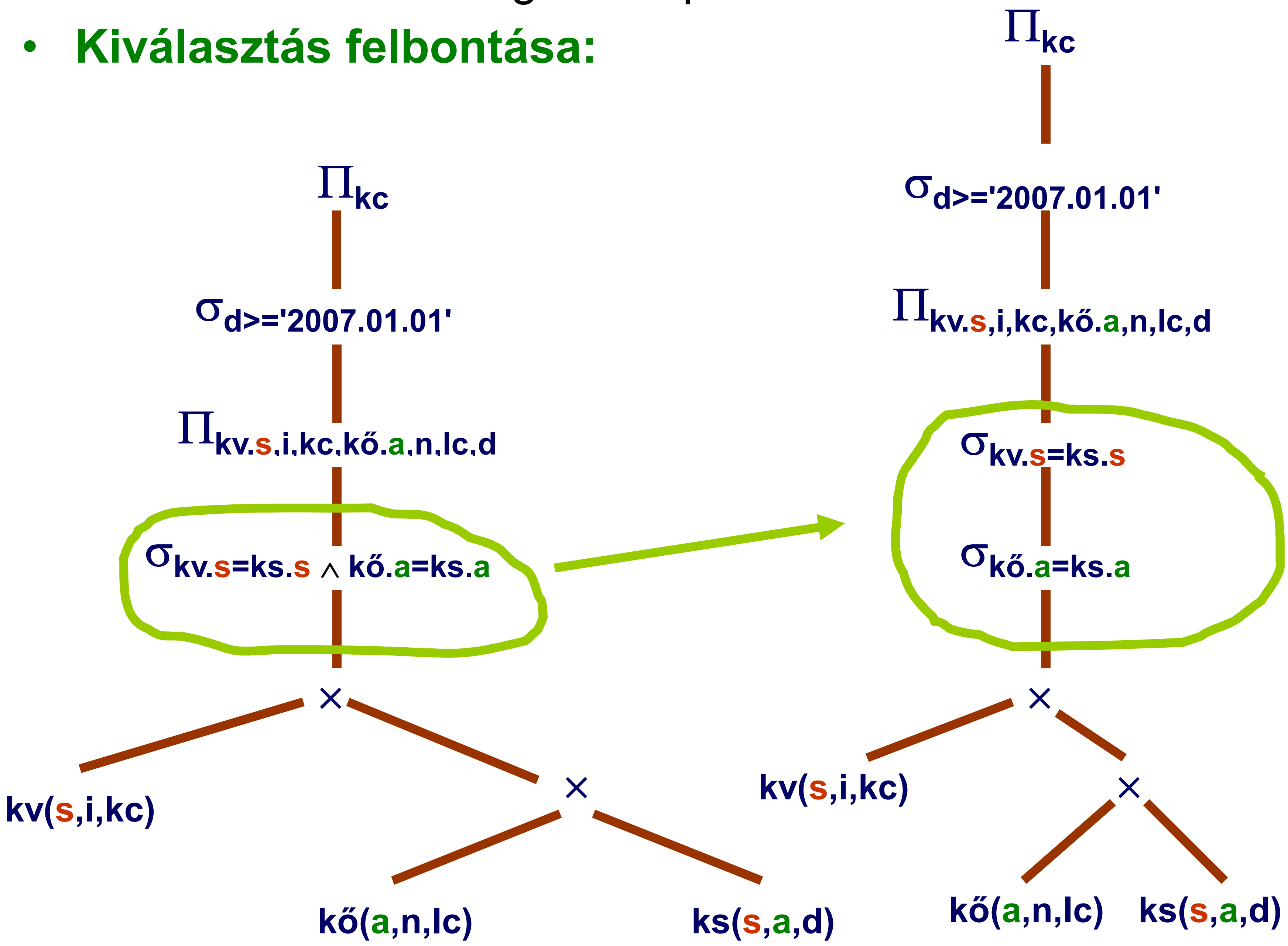
- Optimalizáljuk a következő kifejezést:

$$\Pi_{kc}(\sigma_{d>='2007.01.01'}(\Pi_{kv.s,i,kc,kő.a,n,lc,d}(\sigma_{kv.s=ks.s \wedge kő.a=ks.a}(kv \times (kő \times ks)))))$$

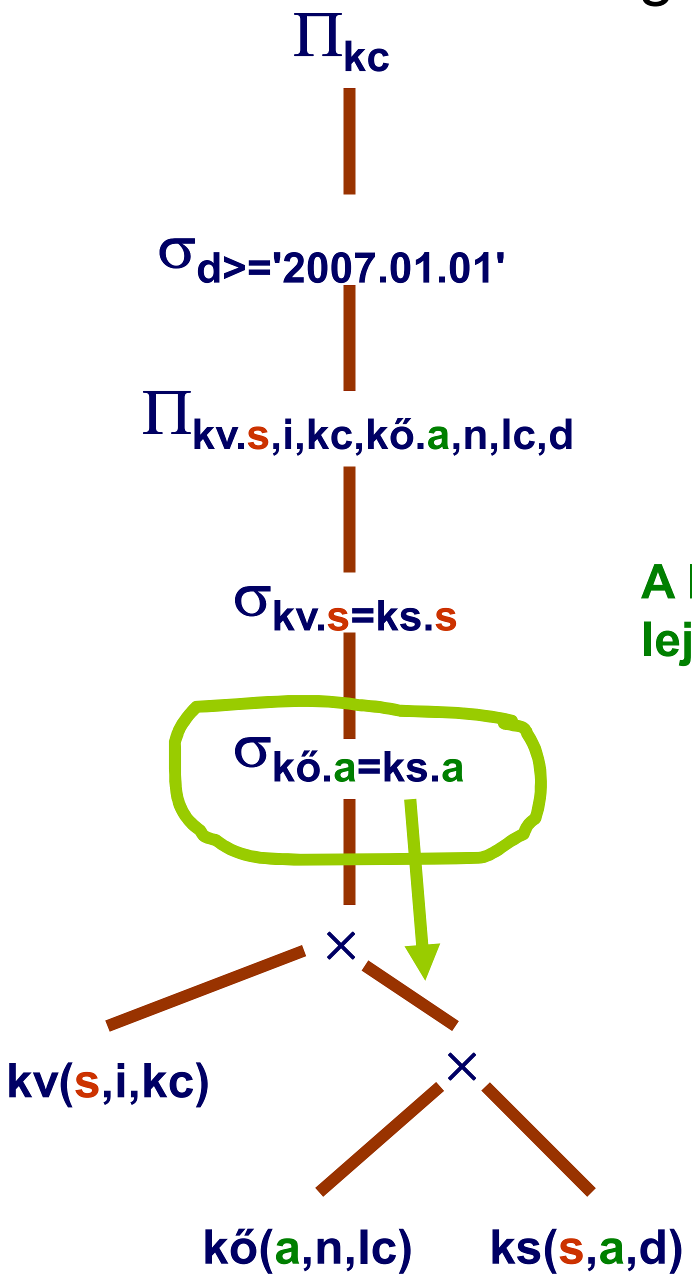


# Algebrai optimalizáció

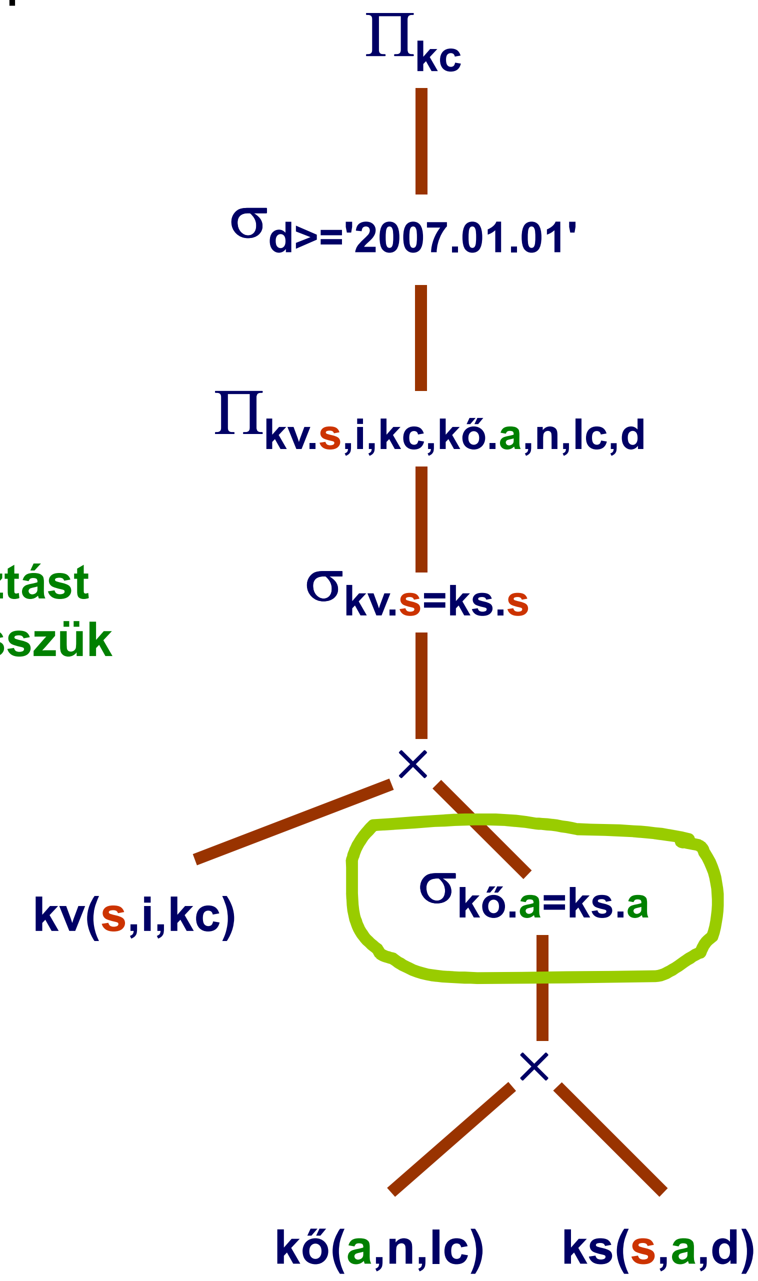
- **Kiválasztás felbontása:**



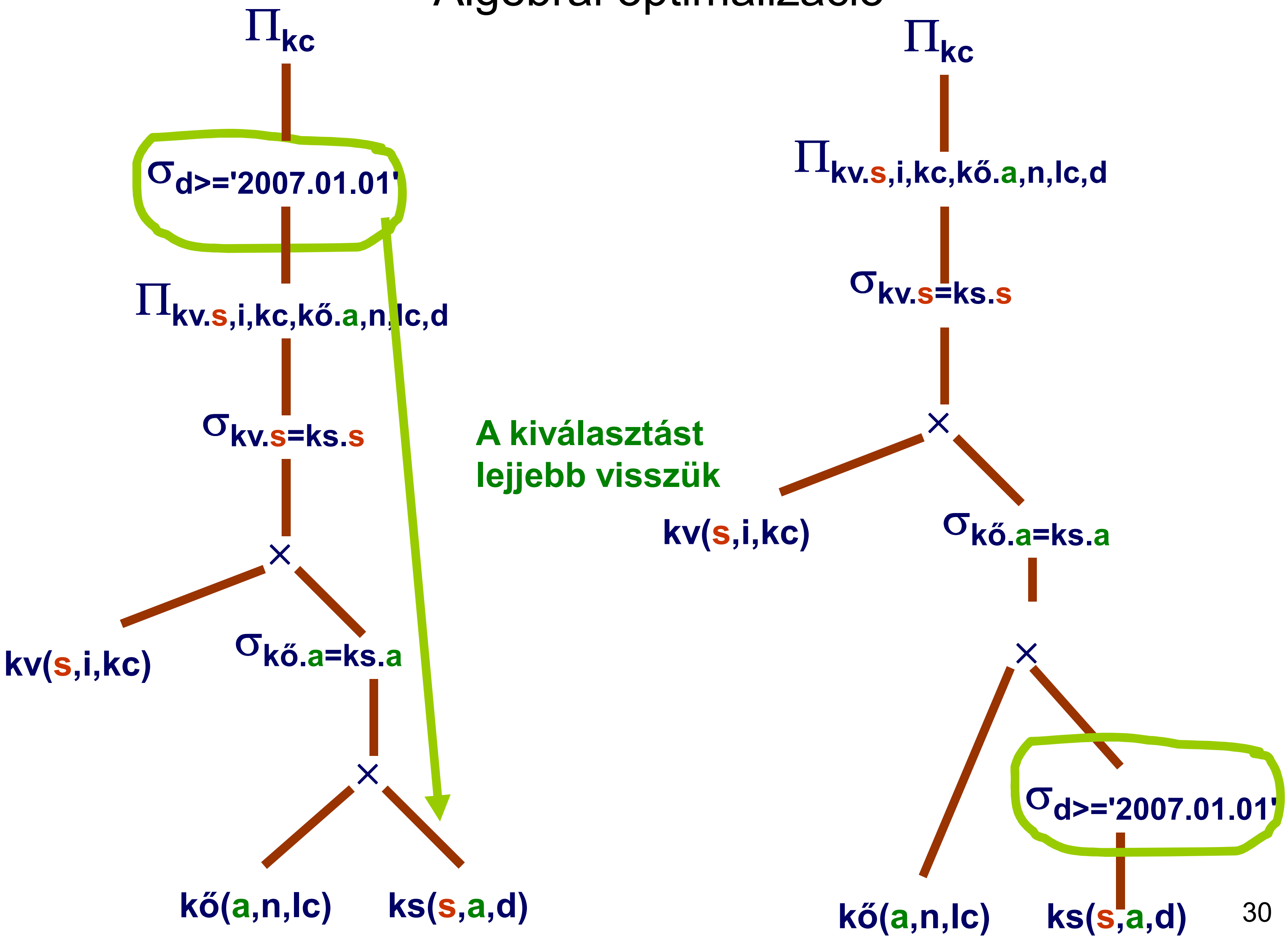
# Algebrai optimalizáció



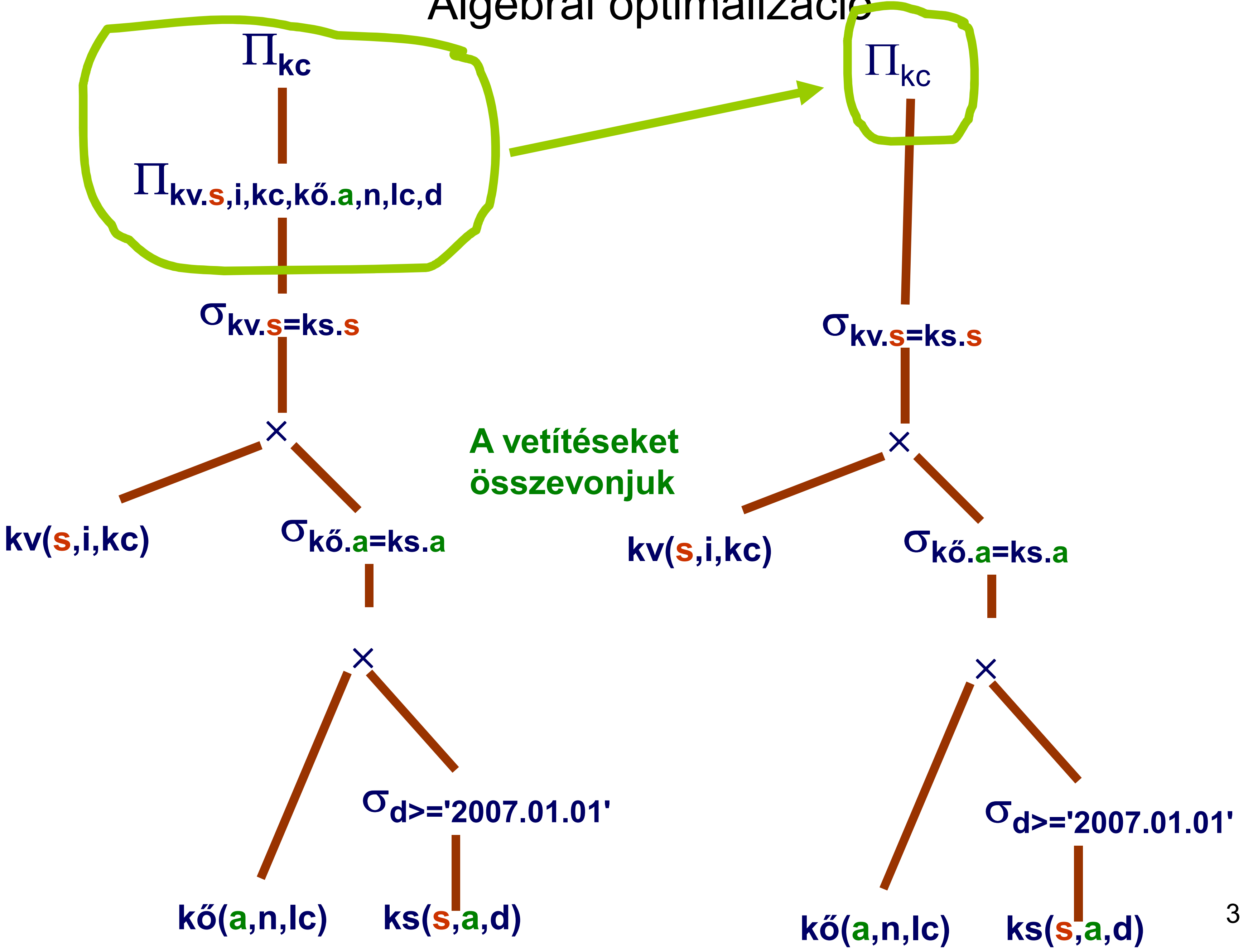

**A kiválasztást lejjebb visszük**



# Algebrai optimalizáció

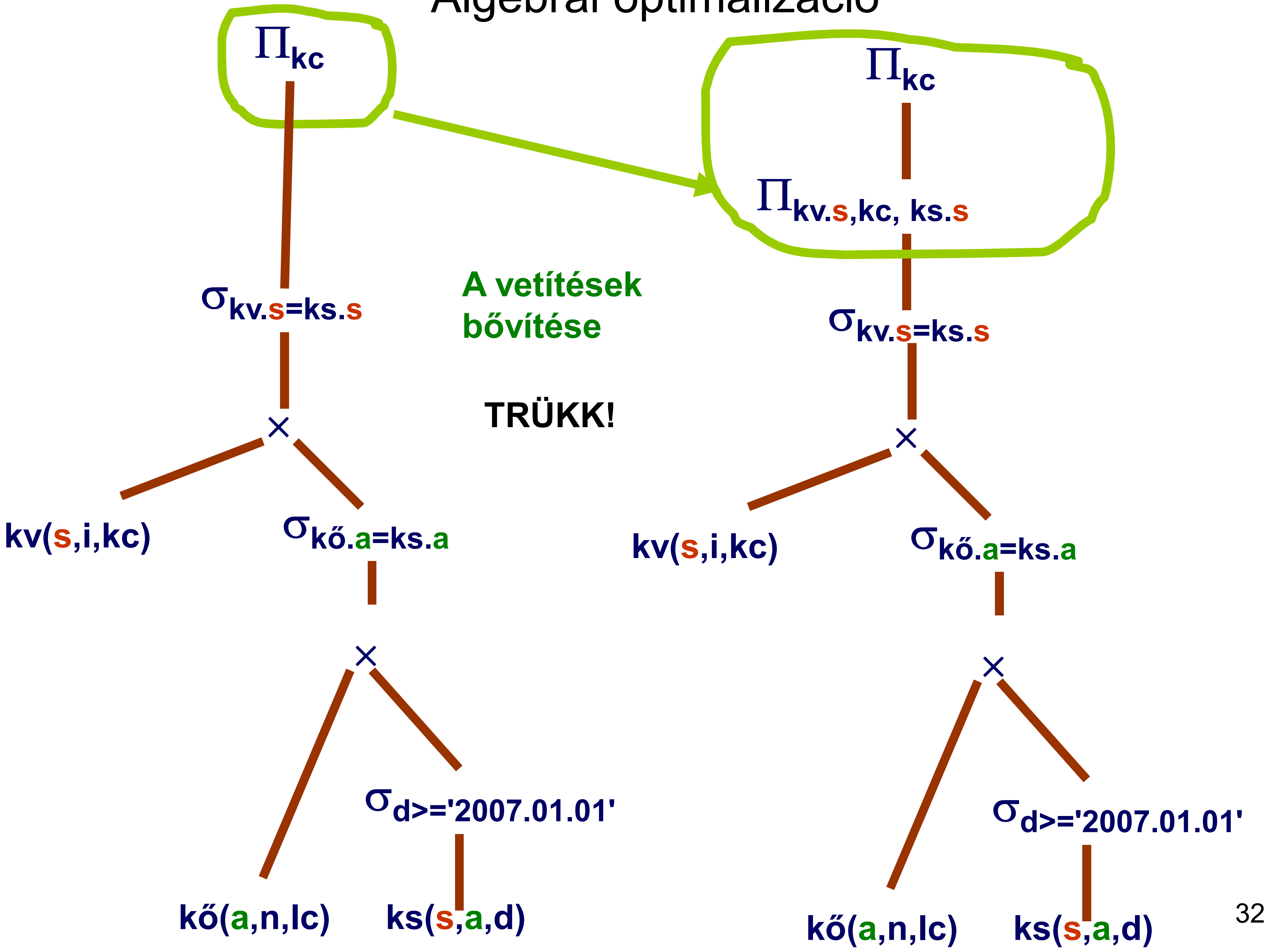


# Algebrai optimalizáció




A vetítéseket összevonjuk

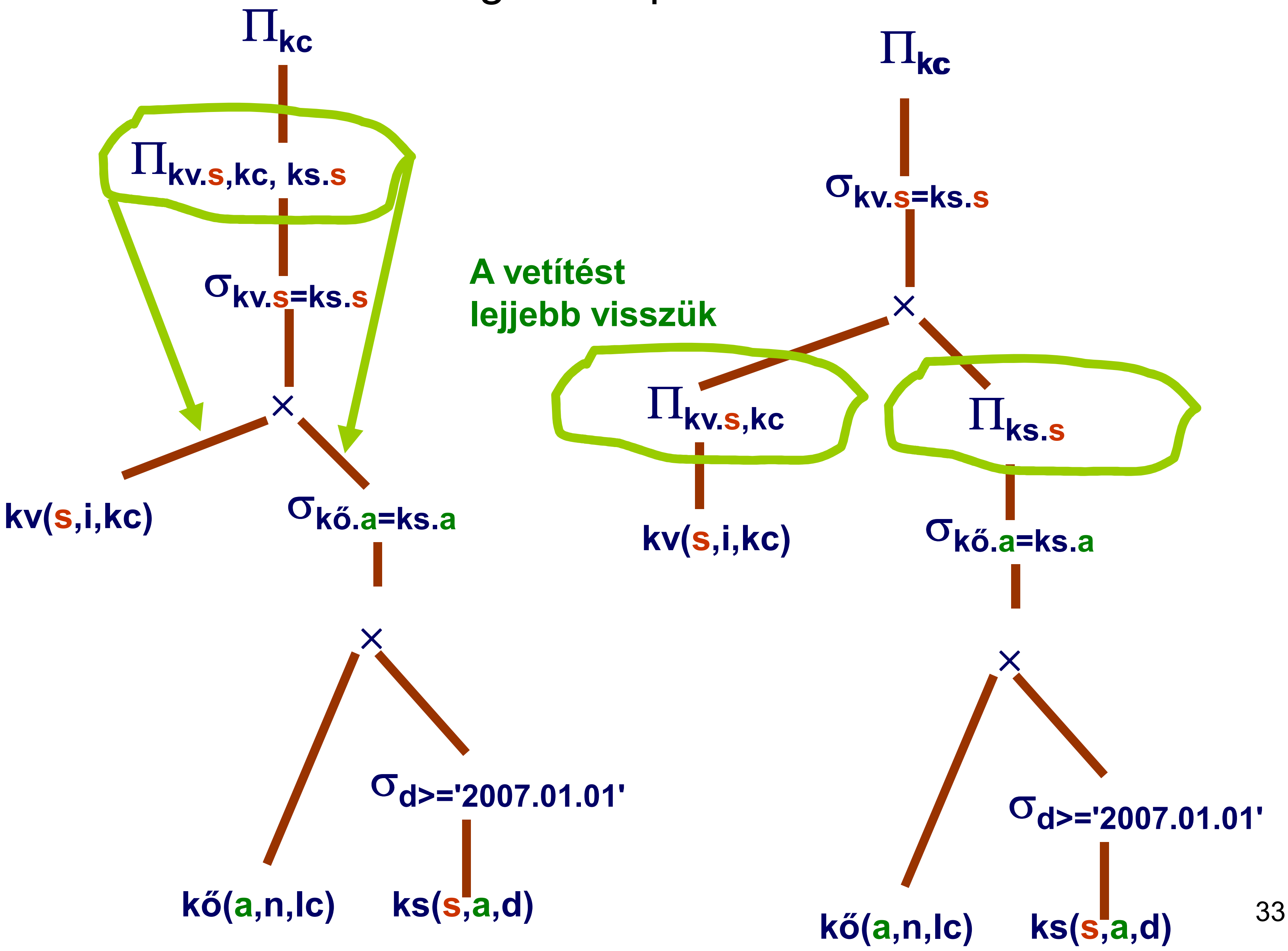
# Algebrai optimalizáció



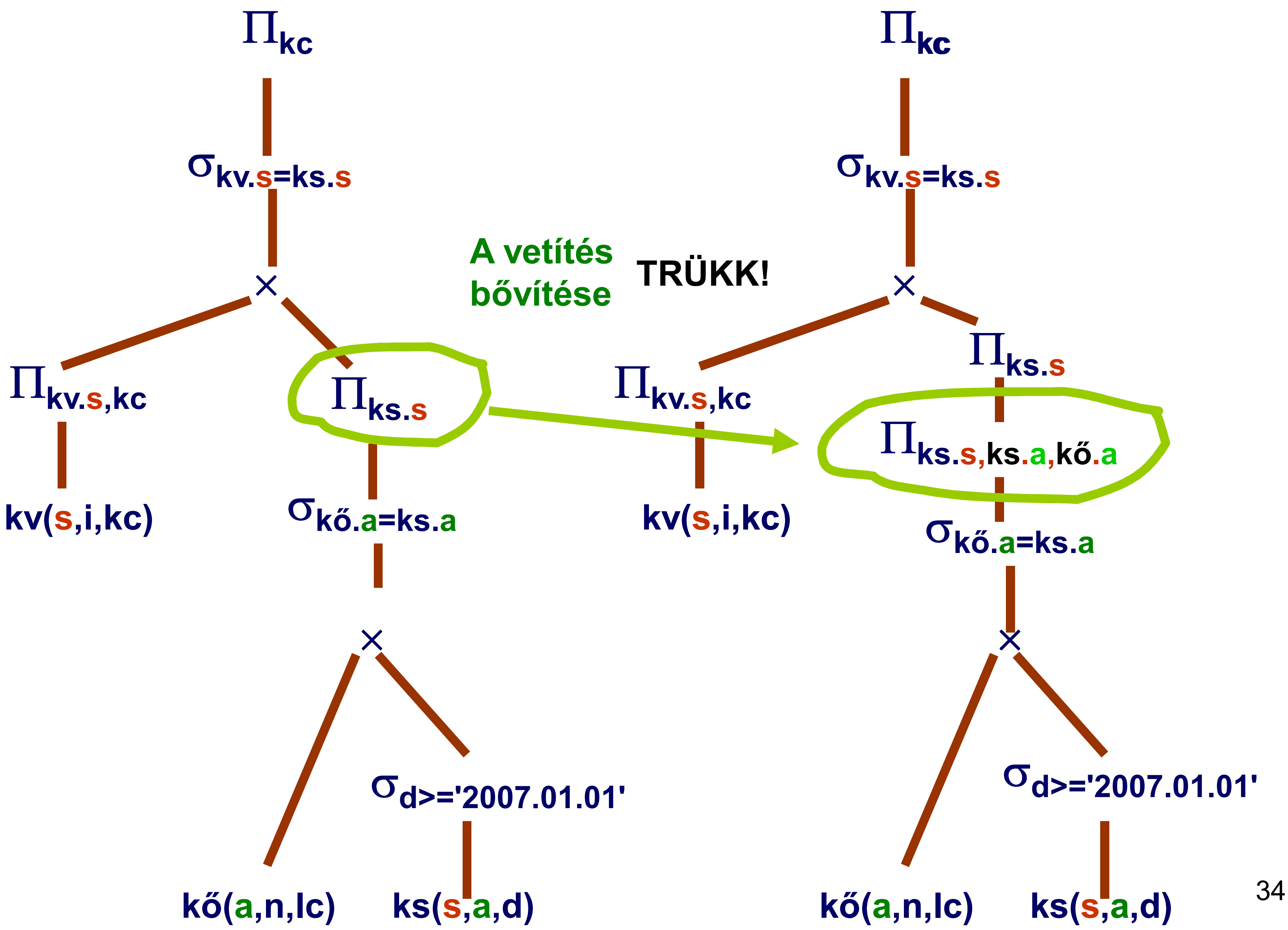

A vetítések bővítése

TRÜKK!

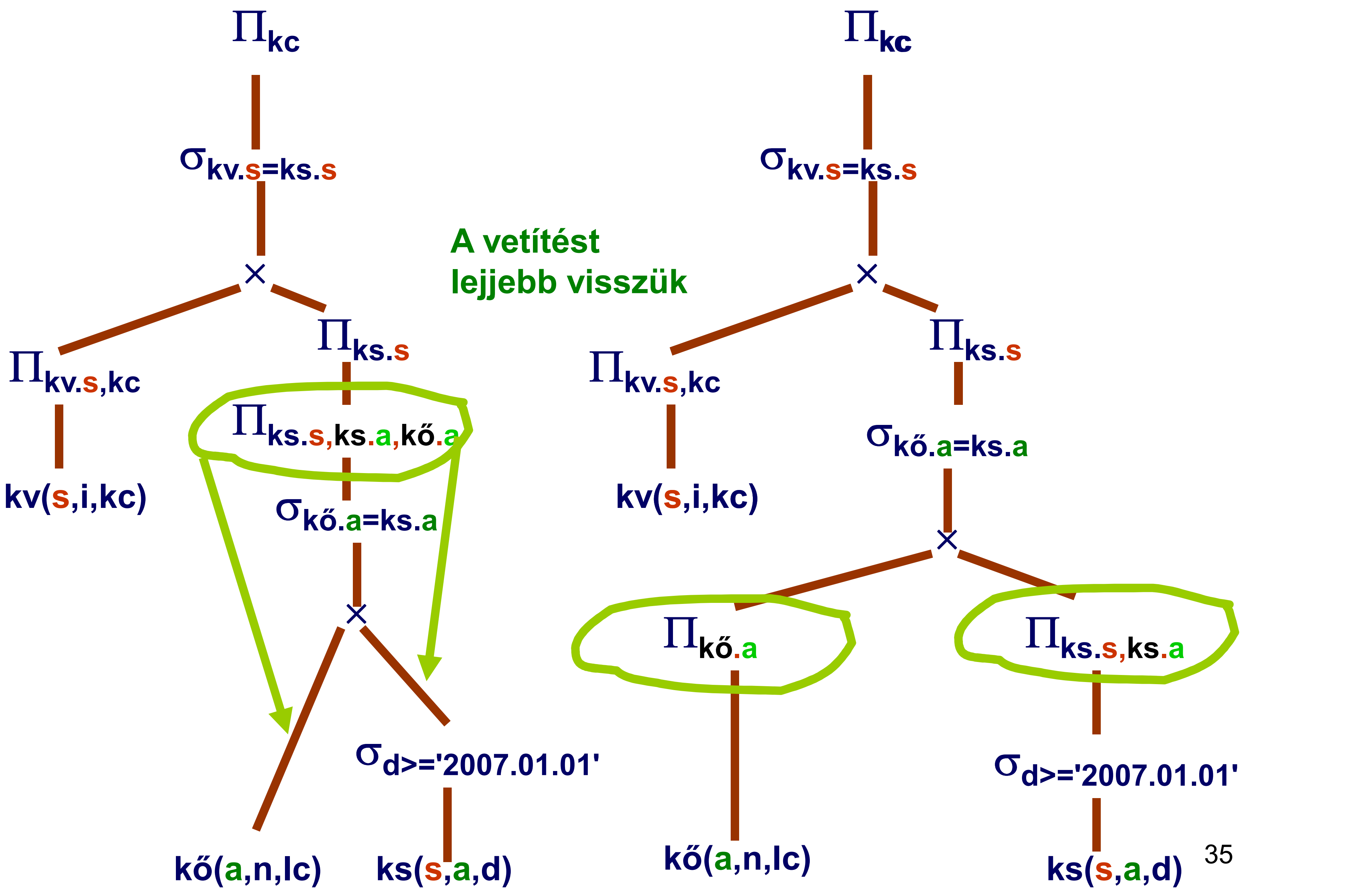
# Algebrai optimalizáció



# Algebrai optimalizáció

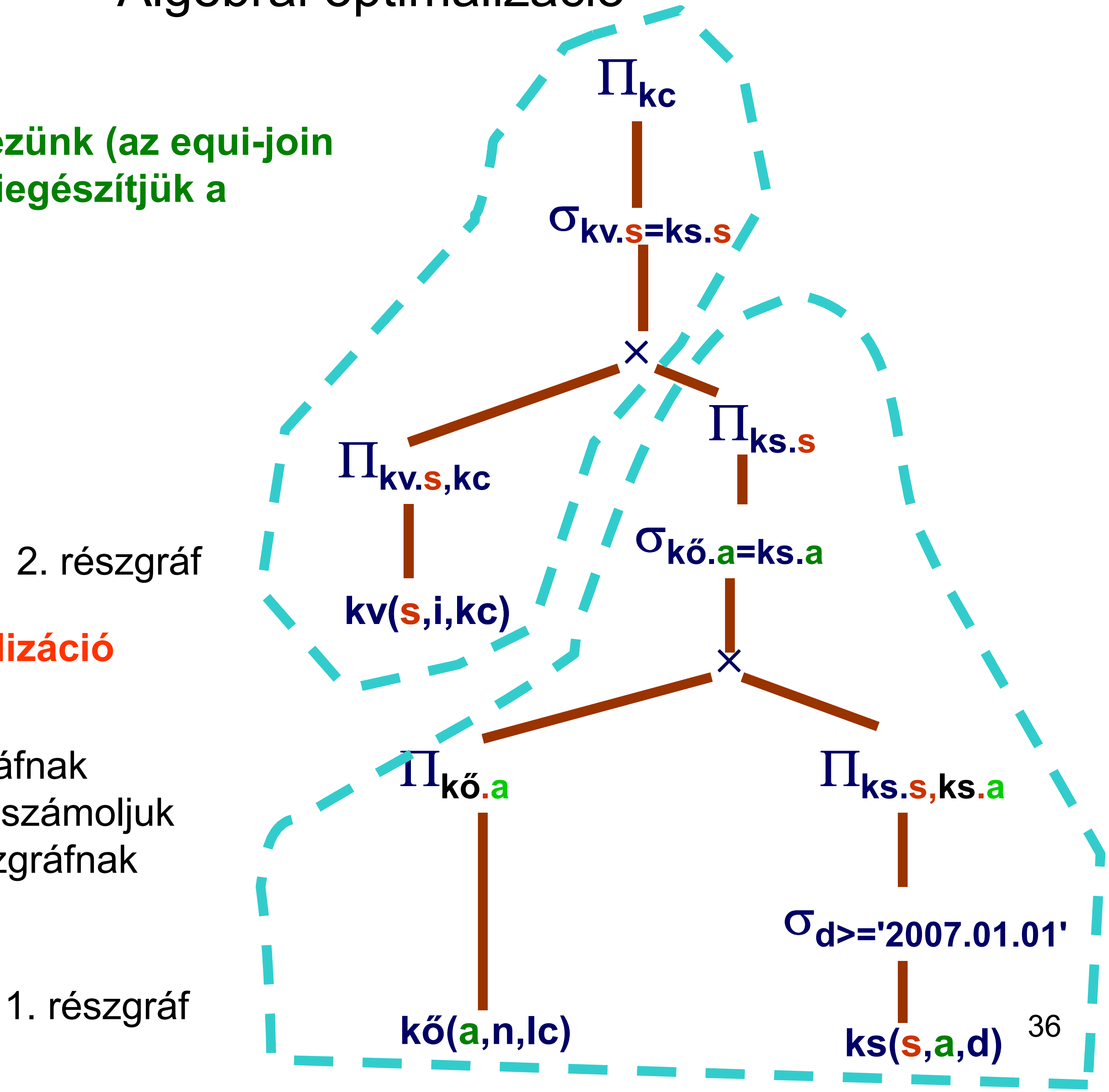


# Algebrai optimalizáció



# Algebrai optimalizáció

**Részgráfokat képezünk (az equi-join miatt a levelekig kiegészítjük a csoportokat)**




**Az algebrai optimalizáció eredménye:**

Először az 1. részgráfnak megfelelő kifejezést számoljuk ki, és utána a 2. részgráfnak megfelelő kifejezést.