

RELÁCIÓS LEKÉRDEZÉSEK OPTIMALIZÁLÁSA

Dr. Gajdos Sándor

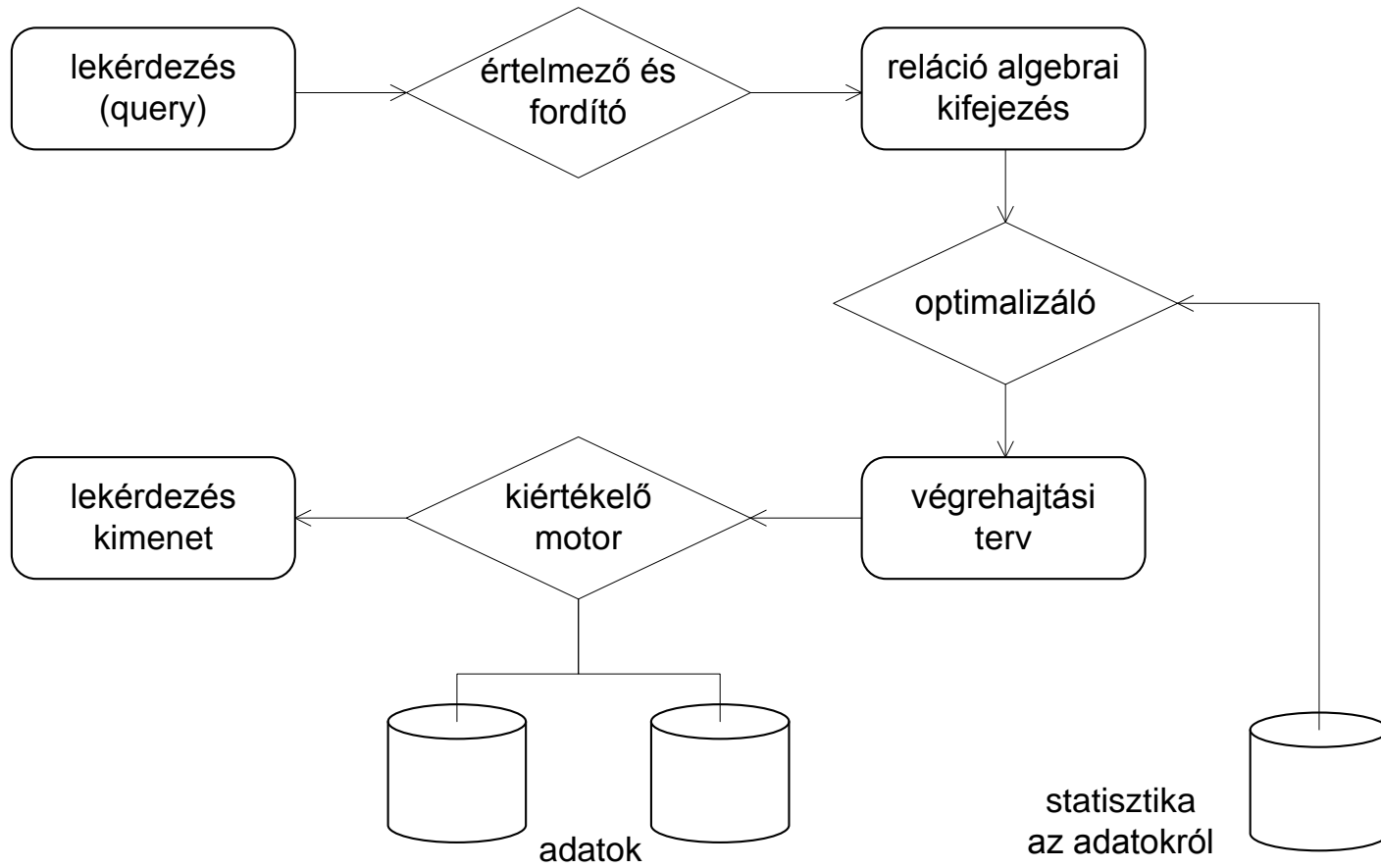
BME-TMIT

2018. okt.

TARTALOM

- Áttekintés
- I. Heurisztikus, szabály alapú optimalizálás
 - Mikor ekvivalens két fa?
- II. Költség alapú optimalizálás
 - Katalógus költségbecslés
 - Operációk, műveletek áttekintése
 - Kifejezéskiértékelés
 - Az optimális végrehajtási terv kiválasztása
- III. Manuális vs. automatikus optimalizálás

ÁTTEKINTÉS



I. HEURISZTIKUS, SZABÁLY ALAPÚ OPTIMALIZÁLÁS

- Relációs algebrai fa alapú optimalizálás
- Lekérdezési fa

EMPLOYEE (EMPLOYEE ID, LAST_NAME, FIRST_NAME, BIRTH_DATE, ...)

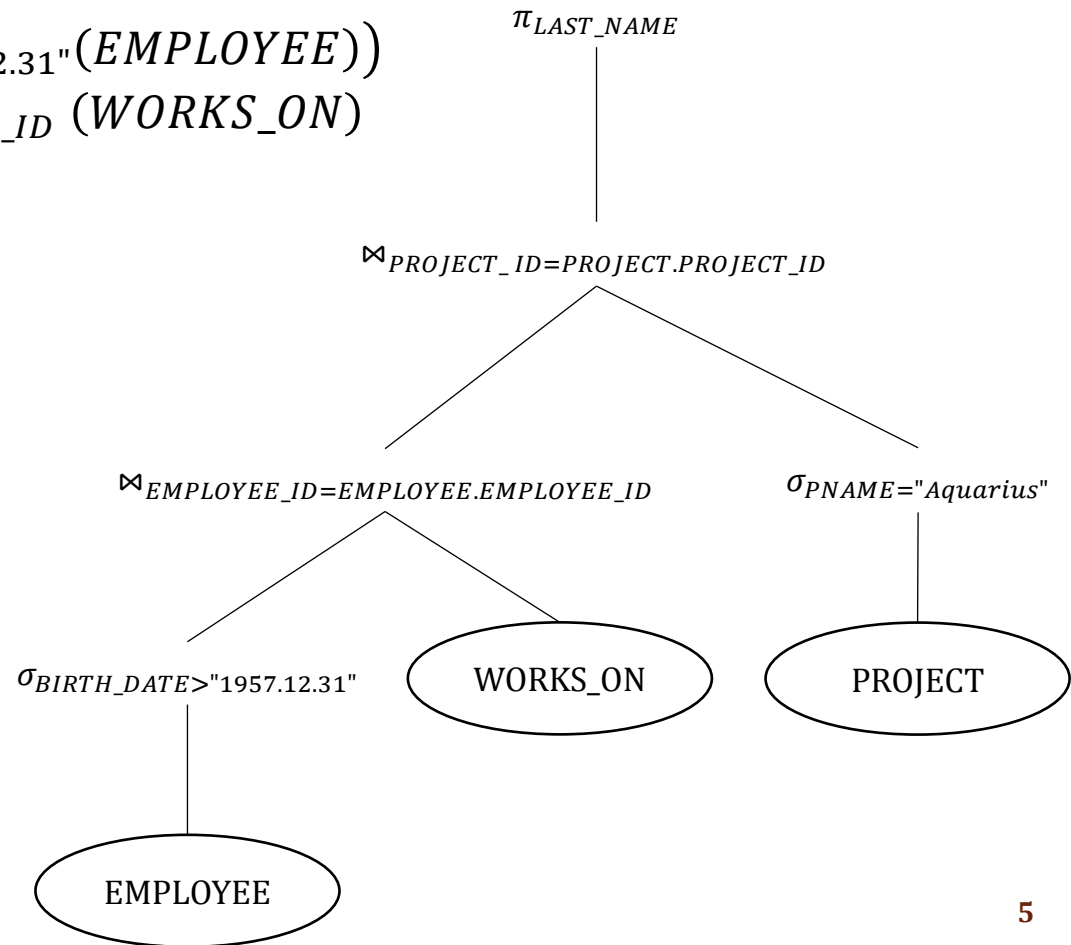
PROJECT (PROJECT ID, PNAME, ...)

WORKS_ON (PROJECT ID, EMPLOYEE ID)

```
select last_name
  from employee, works_on, project
 where employee.birth_date > '1957.12.31'
       and works_on.project_id = project.project_id
       and works_on.employee_id = employee.employee_id
       and project.pname = 'Aquarius'
```

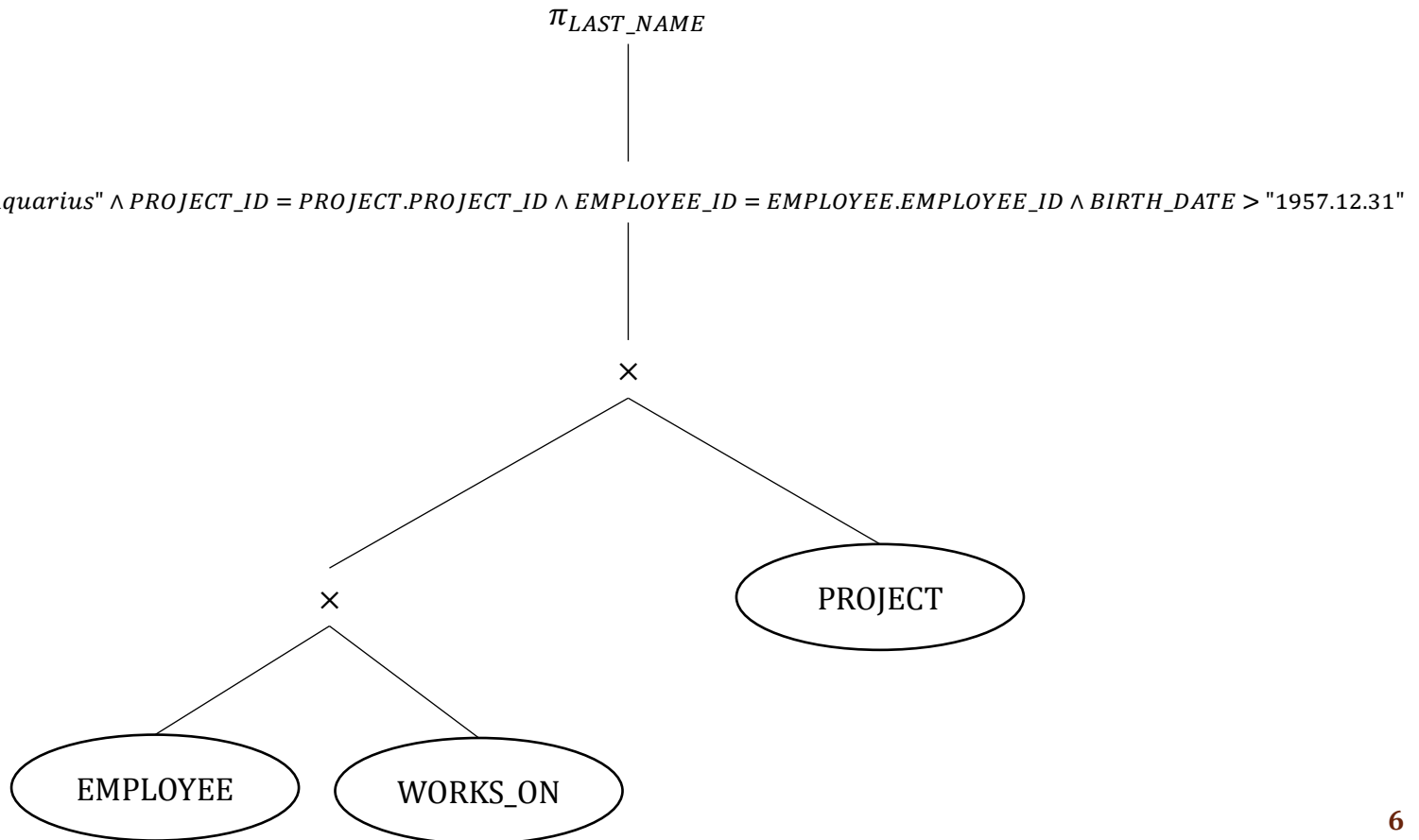
EGY LEHETSÉGES RELÁCIÓS ALGEBRAI MEGFELELŐ

$\pi_{LAST_NAME} \left(\left(\sigma_{BIRTH_DATE > "1957.12.31"}(EMPLOYEE) \right) \right)$
 $\bowtie_{EMPLOYEE_ID=EMPLOYEE.EMPLOYEE_ID} (WORKS_ON)$
 $\bowtie_{PROJECT_ID=PROJECT.PROJECT_ID}$
 $\sigma_{PNAME="Aquarius"}(PROJECT)$

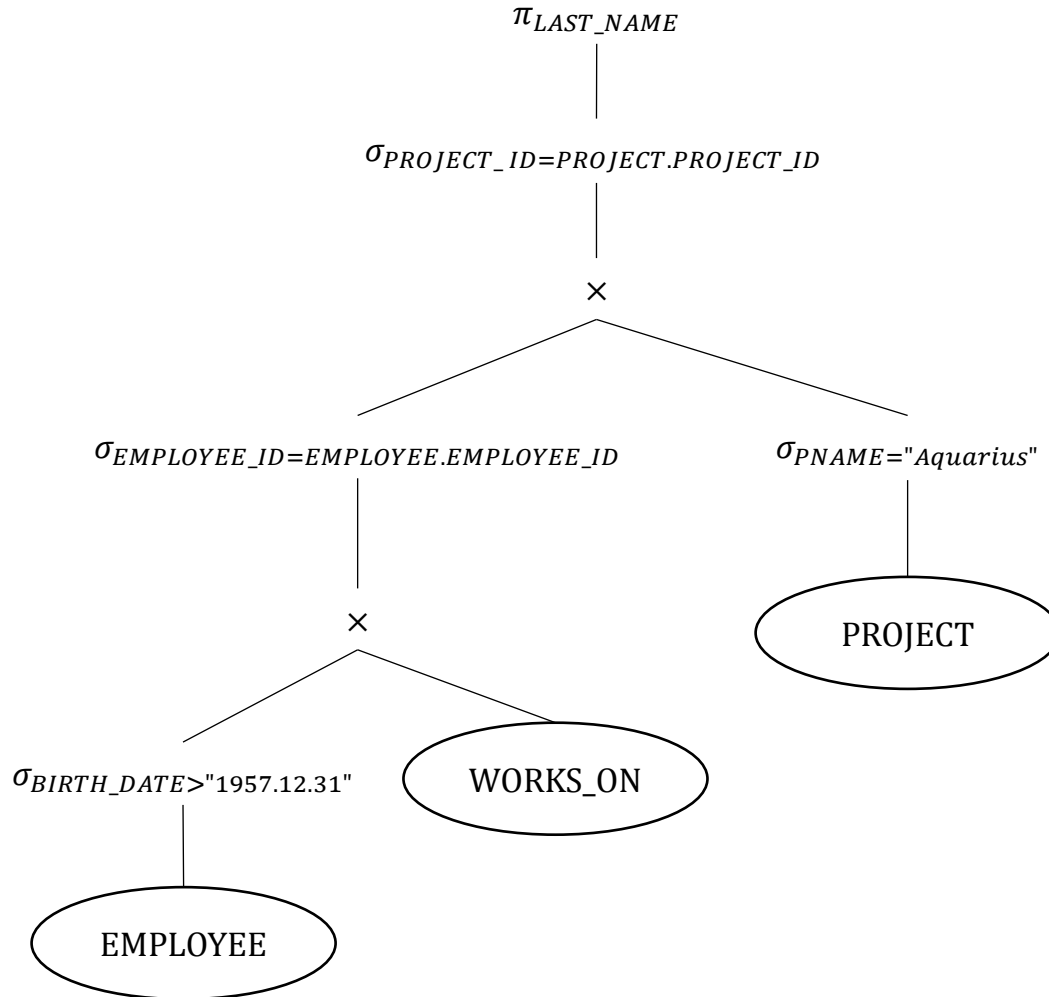


CÉL: A LEGGYORSABB ALAK KIVÁLASZTÁSA

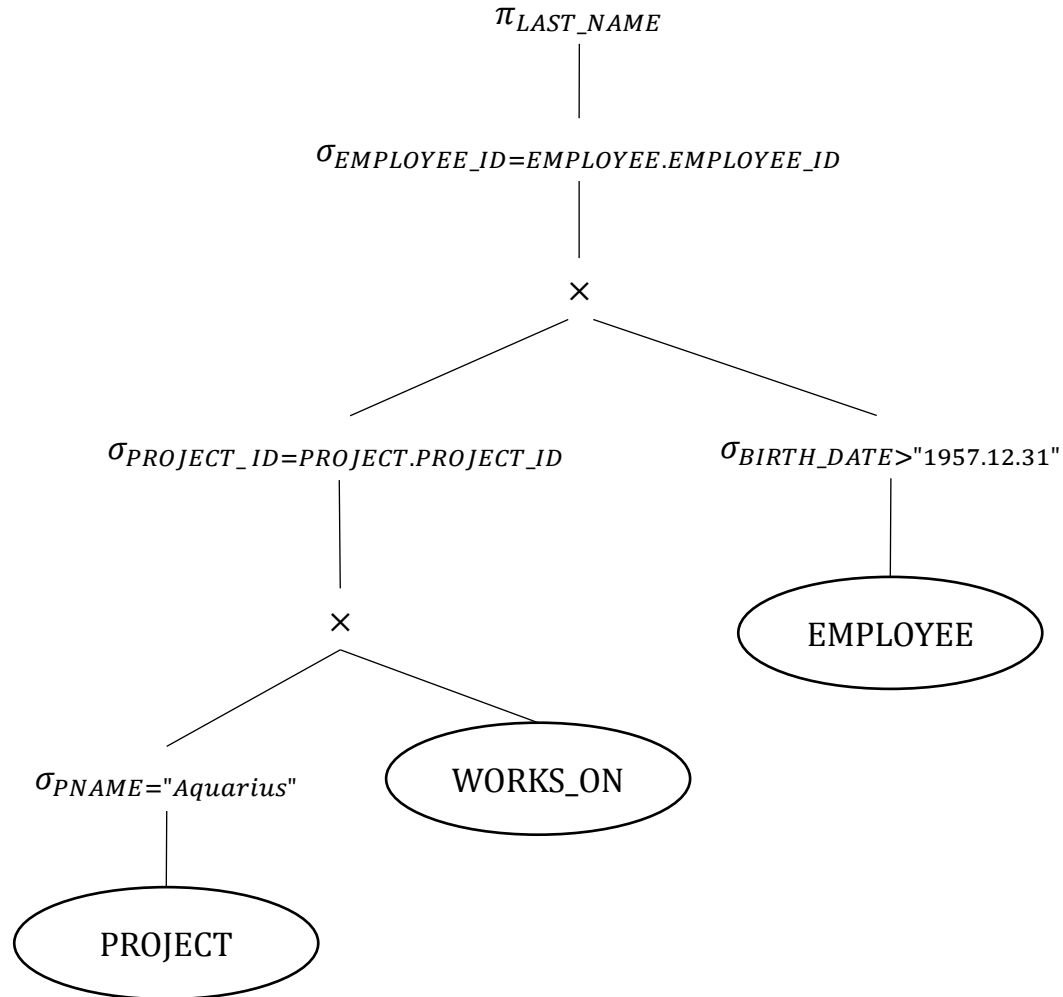
Kiindulás: kanonikus alakból (Descartes, szűrés, projekció)



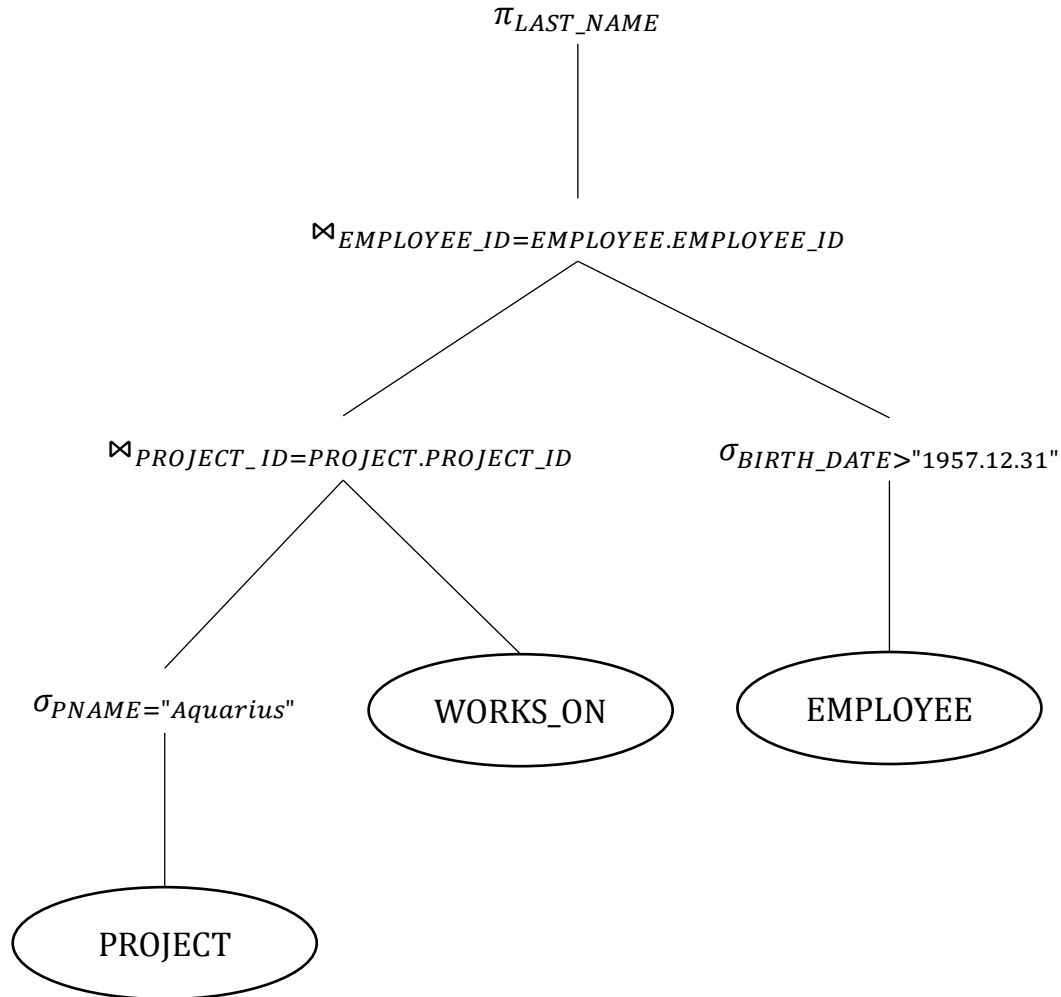
MÁSODIK LÉPÉS: SZELEKCIÓK SÜLLYESZTÉSE



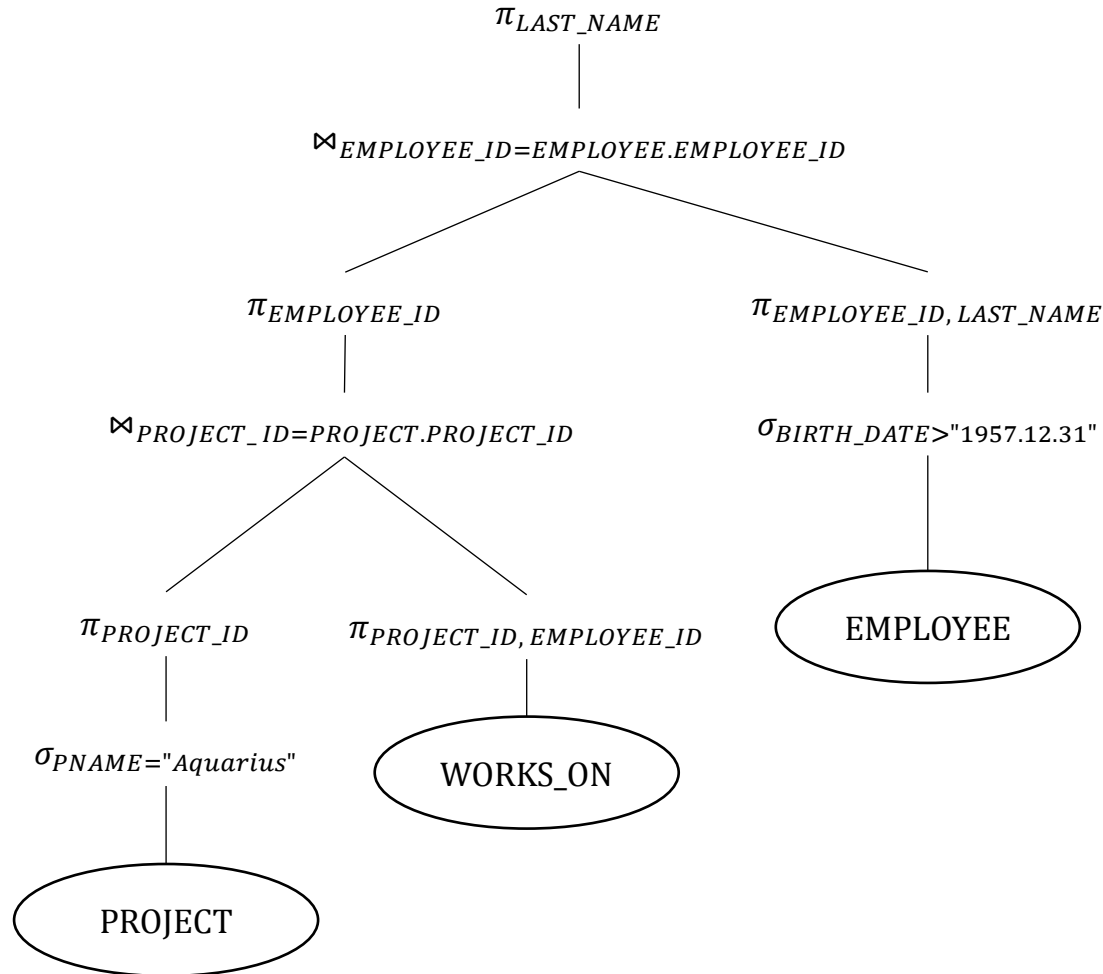
HARMADIK LÉPÉS: LEVELEK ÁTRENDEZÉSE



NEGYEDIK LÉPÉS: JOIN



ÖTÖDIK LÉPÉS: PROJEKCIÓ SÜLLYESZTÉSE



MIKOR EKVIVALENS KÉT FA?

RELÁCIÓS ALGEBRAI TRANSZFORMÁCIÓK I.

- $\sigma_{c_1 \wedge c_2 \wedge \dots \wedge c_n}(r) \equiv \sigma_{c_1}(\sigma_{c_2}(\dots(\sigma_{c_n}(r))\dots))$
- $\sigma_{c_1}(\sigma_{c_2}(r)) \equiv \sigma_{c_2}(\sigma_{c_1}(r))$
- $\pi_{List_1}(\pi_{List_2}(\dots(\pi_{List_n}(r))\dots)) \equiv \pi_{List_1}(r)$
- $\pi_{A_1, A_2, \dots, A_n}(\sigma_c(r)) \equiv \sigma_c(\pi_{A_1, A_2, \dots, A_n}(r))$

MIKOR EKVIVALENS KÉT FA?

RELÁCIÓS ALGEBRAI TRANSZFORMÁCIÓK II.

- $r \bowtie_c s \equiv s \bowtie_c r$
- $\sigma_c(r \bowtie s) \equiv (\sigma_c(r)) \bowtie s$
- $\pi_L(r \bowtie_c s) \equiv (\pi_{A_1, \dots, A_n}(r)) \bowtie_c (\pi_{B_1, \dots, B_m}(s))$

A halmazműveletek (unió, metszet) kommutativitása

A join, Descartes-szorzat, unió és metszet asszociatív:

$$(r\theta s)\theta t \equiv r\theta(s\theta t)$$

MIKOR EKVIVALENS KÉT FA?

RELÁCIÓS ALGEBRAI TRANSZFORMÁCIÓK III.

- $\sigma_C(r \theta s) \equiv (\sigma_C(r)) \theta (\sigma_C(s))$
- $\pi_L(r \theta s) \equiv (\pi_L(r)) \theta (\pi_L(s))$

Egyéb szabályok:

- $c \equiv \neg(c_1 \wedge c_2) \equiv (\neg c_1) \vee (\neg c_2)$
- $c \equiv \neg(c_1 \vee c_2) \equiv (\neg c_1) \wedge (\neg c_2)$

ÖSSZEFOGLALÓ SZABÁLYOK

- Konjunktív szelekciós feltételeket szelekciós feltételek sorozatává bontjuk.
- Szelekciós műveleteket felcseréljük a többi művelettel.
- Átrendezzük a lekérdezési fa leveleit.
- A Descartes-szorzatokat és a fölöttük lévő szelekciós kapcsolási feltételt egy join műveletté vonjuk össze.
- A projekciós műveleteket felcseréljük a többi művelettel .

II. KÖLTSÉG ALAPÚ OPTIMALIZÁLÁS

1. Elemzés (szintaktikus), fordítás
2. Költségoptimalizálás
3. Kiértékelés

II. KÖLTSÉG ALAPÚ OPTIMALIZÁLÁS

- Katalógusadatok alapján történő költségbecslés
 - A katalógusban tárolt egyes relációkra vonatkozó információk
 - Katalógusinformációk az indexekről
 - A lekérdezés költsége
- Megoldás az adatok frissítésére

A KATALÓGUSBAN TÁROLT EGYES RELÁCIÓKRA VONATKOZÓ INFORMÁCIÓK

- n_r : az r relációban levő rekordok száma (number)
- b_r : az r relációban levő rekordokat tartalmazó blokkok (blocks) száma
- s_r : egy rekord nagysága (size) bájtokban
- f_r : mennyi rekord fér egy blokkba (blocking factor)

A KATALÓGUSBAN TÁROLT EGYES RELÁCIÓKRA VONATKOZÓ INFORMÁCIÓK

- $V(A, r)$: hány különböző értéke (Values) fordul elő az A attribútumnak az r relációban (kardinalitás).
 - $V(A, r) = |\pi_A(r)|$
 - Ha A kulcs, akkor $V(A, r) = n_r$
- $SC(A, r)$: (**S**election **C**ardinality) azon rekordok átlagos száma, amelyek egy kiválasztási feltételt kielégítenek.
 - Ha A kulcs, akkor $SC(A, r) = 1$
 - Általános esetben $SC(A, r) = \frac{n_r}{V(A, r)}$
- Ha a relációk rekordjai fizikailag együtt vannak tárolva, akkor:

$$b_r = \left\lceil \frac{n_r}{f_r} \right\rceil$$

KATALÓGUS INFORMÁCIÓK AZ INDEXEKRŐL

- f_i : pointer kimenetek átlagos száma a fa struktúrájú indexeknél, pl. a B* fáknál
- HT_i : az index szintjeinek száma (**H**eight of **T**ree)
 - $HT_i = \lceil \log_{f_i} V(A, r) \rceil$ (B*-fa)
 - $HT_i = 1$ (hash)
- LB_i : a levélszintű indexblokkok száma (**L**owest level index **B**lock)

KÖLTSÉG MEGHATÁROZÁSA

Meghatározása:

- igényelt és felhasznált erőforrások alapján?
- válaszidő alapján?
- kommunikációra fordított idő alapján?

Definíció:

- háttértár blokkolvasások és írások száma a válasz kiírásának költsége nélkül

További egyszerűsítések.

OPERÁCIÓK, MŰVELETEK KÖLTSÉGE

- Select
 - szelekciós algoritmusok (alap, indexelt, összehasonlításos)
 - komplex szelekció
- Join
 - típusai
 - join nagyságbecslés
 - join algoritmusok
 - komplex join
- Egyéb
 - ismétlődés kiszűrése
 - unió, metszet, különbség

ALAP SZELEKCIÓS ALGORITMUSOK (=)

A1: Lineáris keresés

- Költsége:

$$E_{A1} = b_r$$

A2: Bináris keresés

- Feltétele:
 - Blokkok folyamatosan a diszken
 - Az A attribútum szerint rendezettek
 - Szelekció feltétele az egyenlőség az A attribútumon
- Költsége:

$$E_{A2} = \lceil \log_2(b_r + 1) \rceil + \left\lceil \frac{SC(A, r)}{f_r} \right\rceil - 1$$

INDEXELT SZELEKCIÓS ALGORITMUSOK

A3: Elsődleges index használatával, egyenlőségi feltételt a kulcson vizsgálva

- $E_{A3} = HT_i + 1$

A4: Elsődleges index használatával, egyenlőségi feltétel nem kulcson (a nemkulcs attribútumon van az elsődleges index)

- $E_{A4} = HT_i + \left\lceil \frac{SC(A,r)}{f_r} \right\rceil$

A5: Másodlagos index használatával.

- $E_{A5} = HT_i + SC(A, r)$

- $E_{A5} = HT_i + 1$, ha A kulcs

ÖSSZEHASONLÍTÁS ALAPÚ SZELEKCIÓ – $\sigma_{A \leq v}(R)$

Az eredményrekordok számának becslése:

- Ha v -t nem ismerjük: $\frac{n_r}{2}$

- Ha v -t ismerjük, egyenletes eloszlás esetén:

$$n_{\text{átlagos}} = n_r \cdot \frac{v - \min(A, r)}{\max(A, r) - \min(A, r)}$$

ÖSSZEHASONLÍTÁS ALAPÚ SZELEKCIÓ – $\sigma_{A \leq v}(R)$

A6: Elsődleges index használatával.

- Ha v -t nem ismerjük:

$$E_{A6} = HT_i + \frac{b_r}{2}$$

- Ha v -t ismerjük:

$$E_{A6} = HT_i + \left\lfloor \frac{c}{f_r} \right\rfloor,$$

ahol c jelöli azon rekordok számát, ahol $A \leq v$

A7: Másodlagos index használatával

$$E_{A7} = HT_i + \frac{LB_i}{2} + \frac{n_r}{2}$$

JOIN OPERÁCIÓ

Definíció:

$$r_1 \bowtie_{\theta} r_2 = \sigma_{\theta}(r_1 \times r_2)$$

Típusai:

- Természetes illesztés (natural join)

$$r_1 \bowtie r_2 = \pi_{A \cup B}(\sigma_{R1.X=R2.X}(r_1 \times r_2))$$

- Külső illesztés (outer join)

- Bal oldali külső illesztés: $r_1 * (+)r_2$

- Jobb oldali külső illesztés: $r_1 (+) * r_2$

- Teljes külső illesztés: $r_1 (+) * (+)r_2$

- Theta illesztés:

$$r_1 \bowtie_{\theta} r_2 = \sigma_{\theta}(r_1 \times r_2)$$

NESTED-LOOP JOIN (EGYMÁSBA ÁGYAZOTT CIKLIKUS ILLESZTÉS)

Adott két reláció, r és s :

FOR minden $t_r \in r$ rekordra DO BEGIN

 FOR minden $t_s \in s$ rekordra DO BEGIN

 teszteljük (t_r, t_s) párt, hogy kielégíti-e a θ -join feltételt

 IF igen, THEN adjuk a $t_r \cdot t_s$ rekordot az eredményhez

 END

END

- „worst case” költség: $n_r \cdot b_s + b_r$
- ha legalább az egyik befér a memóriába, akkor a költség: $b_r + b_s$

BLOCK NESTED-LOOP JOIN

(BLOKKALAPÚ EGYMÁSBA ÁGYAZOTT CIKLIKUS ILLESZTÉS)

```
FOR minden  $b_r \in r$  blokkra DO BEGIN
    FOR minden  $b_s \in s$  blokkra DO BEGIN
        FOR minden  $t_r \in b_r$  rekordra DO BEGIN
            FOR minden  $t_s \in b_s$  rekordra DO BEGIN
                teszteljük le a  $(t_r, t_s)$  párt
            END
        END
    END
END
END
```

- „worst-case” költsége: $b_r \cdot b_s + b_r$
- sok memóriával: $b_r + b_s$

INDEXED NESTED-LOOP JOIN (INDEXALAPÚ EGYMÁSBA ÁGYAZOTT CIKLIKUS ILLESZTÉS)

Az egyik relációhoz (s) van indexünk

Tegyük az első algoritmus belső ciklusába az indexelt relációt

⇒ A keresés index alapján kisebb költséggel is elvégezhető

Költsége:

$$b_r + n_r \cdot c,$$

ahol c a szelekció költsége s -en.

TOVÁBBI JOIN IMPLEMENTÁCIÓK

- sorted merge join
 - a relációkat a join feltételben meghatározott attribútumok mentén rendezzük, majd összefésüljük
- hash join
 - az egyik relációt hash-táblán keresztül érjük el, miközben a másik reláció egy adott rekordjához illeszkedő rekordokat keressük
- egyéb
 - pl. bitmap indexekkel (bitmap join)

EGYÉB OPERÁCIÓK

- *Ismétlődés kiszűrése* (rendezés, majd törlés)
- *Projekció* (projekció, majd ismétlődés kiszűrés)
- *Unió* (mindkét relációt rendezzük, majd összefésülésnél kiszűrjük a duplikációkat)
- *Metszet* (mindkét relációt rendezzük, fésülésnél csak a másodpéldányokat hagyjuk meg)
- *Különbség* (mindkét relációt rendezzük, fésülésnél csak első relációbeli rekordokat hagyunk)
- *Aggregáció* pl.

márkanév $G_{\text{sum(egyenleg)}}$ (számla)

számítása, pl. rendezéssel márkanévre. Összegzés on-the-fly.

KIFEJEZÉSKIÉRTÉKELÉS MÓDJAI

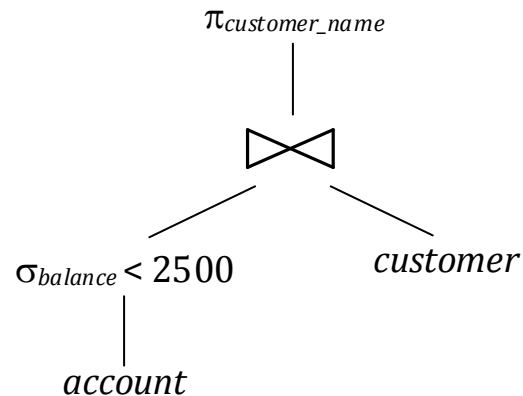
- Materializáció
 - összetett kifejezésnek egyszerre egy műveletét értékeljük ki valamilyen rögzített sorrend szerint
- Pipelining
 - egyszerre több elemi művelet szimultán kiértékelése folyik
 - egy operáció eredményét azonnal megkapja a sorban következő operáció operandusként

MATERIALIZÁCIÓ

- Kanonikus alak:

$$\pi_{customer_name}(\sigma_{balance < 2500}(account) \bowtie customer)$$

- Műveleti fa:



- Eredő költség: a végrehajtott műveletek költsége + részeredmények tárolásának költsége
- Előnye: egyszerű implementálhatóság
- Hátrány: sok háttértár-művelet

PIPELINING

- szimultán kiértékelés
- a részegységek az előttük álló elemtől kapott eredményekből a
 - sorban következő számára állítanak elő részeredményeket
- nem számítja ki előre az egész relációt

Előnye:

- kiküszöböli az ideiglenes tárolás szükségességét
- kis memóriaigény

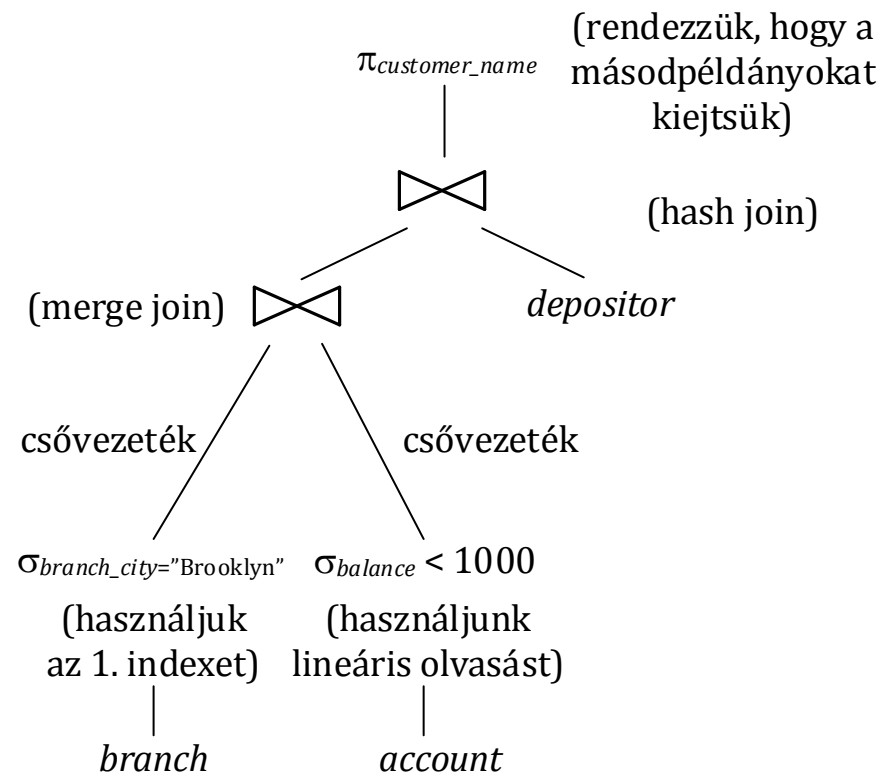
Hátránya:

- szűkíti a felhasználható algoritmusok körét

A KIÉRTÉKELÉSI TERV KIVÁLASZTÁSA

- milyen műveletek
- milyen sorrendben
- milyen algoritmus szerint
- milyen workflow-ban

egy konkrét
kiértékelési terv



KÖLTSÉGALAPÚ OPTIMALIZÁCIÓ

Mohó és egyben rossz stratégia:

- Minden ekvivalens kifejezés felsorolása
- Minden forma kiértékelése
- Az optimális kiválasztása

Pl.: Tekintsük az alábbi kifejezést: $r_1 \bowtie r_2 \bowtie r_3 \rightarrow 12$ ekvivalens

Általános esetben: n reláció illesztésére $\frac{(2(n-1))!}{(n-1)!}$ ekvivalens lehetőség.

Ez túl nagy terhelés lenne a rendszer számára.

A megoldás: heurisztikus költség alapú optimalizálás

III. AUTOMATIKUS VS. MANUÁLIS OPTIMALIZÁLÁS

Az automatikus optimalizáló előnyei:

- Szélesebb ismeret a letárolt adatértékekről.
- Gyorsabb numerikus kiértékelési mechanizmus.
- Szisztematikus értékelés.
- Algoritmusai több szakember együttes tudását hordozza.
- Dinamikusan, minden művelet előtt, az aktuális feltételeket figyelembe véve értékelődik ki.

Az emberi optimalizálás előnyei:

- Szélesebb általános ismeret, a probléma szemantikai tartalmának felhasználása lehetséges.
- Nagyobb szabadság a felhasználható módszerek, eszközök tekintetében.
- Váratlan helyzetekre jobban felkészült.