Optimizarea interogarilor bazelor de date relationale (partea 2)

Ioana Ciuciu
ioana.ciuciu@ubbcluj.ro
http://www.cs.ubbcluj.ro/~oana/

Planificare

Sunt posibile mici modificari ale planificarii in timpul semestrului

Saptamana	Curs	Seminar	Laborator
S1	1. Concepte fundamentale ale bazelor de date. Modelare conceptuala	1. Modelul Entitate-Relatie. Modelul relational	1. Modelarea unei BD in modelul ER si implementarea ei in SQL Server
S2	2. Modelul relational de organizare a bazelor de date. Modelare conceptuala		
S3	3. Gestiunea bazelor de date relationale cu limbajul SQL (DDL)	2. Limbajul SQL – definirea si actualizarea datelor	2. Interogari SQL
S4	4. Gestiunea bazelor de date relationale cu limbajul SQL (DML)		
S5-6	5-6. Dependente functionale, forme normale	3. Limbajul SQL – regasirea datelor	3. Interogari SQL avansate
S7	7. Interogarea bazelor de date relationale cu operatori din algebra relationala	4. Proceduri stocate	4. Proceduri stocate. View. Trigger
S8	8. Structura fizica a bazelor de date relationale		
S9	9. FARA CURS (30 nov.)	5. View-uri. Functii definite de utilizator. Trigger	
S10-11	10-11. Indecsi. Arbori B. Fisiere cu acces direct	6. Formele normale ale unei relatii. Indecsi	
S12	12. Extensii ale modelului relational si baze de date NoSQL – curs invitat UBB, vineri 22 Dec. 2023		
S13	13. Evaluarea interogarilor in bazele de date relationale	7. Probleme	Examen practic
S14	14. Aplicatii		

Planul cursului

- Curs 13-14 (partea 2)
  - Executia instructiunilor SQL
  - Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare
  - Forma optimala pentru expresia din algebra relationala

Executia instructiunilor SQL

- **Executia instructiunilor SQL la serverul de date**:
  - se face **analiza** instructiunii SQL (analiza sintactica, semantica)
  - se translateaza intr-o **forma interna** (ca expresie in algebra relationala)
    - Eventualele view-uri se inlocuiesc cu expresiile corespunzatoare
  - forma interna se transforma intr-o **forma optimala**
  - se genereaza un **plan de executie** (procedural)
  - se **evalueaza** planul generat si se trimite rezultatul la client

Executia instructiunilor SQL

- La generarea planului de executie se folosesc **operatorii din algebra relationala.** Pentru **interogare** sunt necesari urmatorii operatori:

  - **Selectia**: $\sigma_c(R)$
  - **Proiectia**: $\Pi_\alpha(R)$
  - **Produsul cartezian**: R1×R2
  - **Reuniunea**: $R_1 \cup R_2$
  - **Diferenta**: $R_1 - R_2$
  - **Intersectia**: $R_1 \cap R_2$
  - **Joinul conditional**: $R_1 \otimes_\Theta R_2$
  - **Joinul natural**: $R_1 * R_2$
  - **Joinul extern stanga**: R1 ⟕c R2
  - **Joinul extern dreapta**: R1 ⟖c R2
  - **Joinul extern complet**: R1 ⟗c R2
  - **Semi joinul stanga**: R1 ⊳ R2
  - **Semi joinul dreapta**: R1 ⊲ R2
  - **Câtul**: R1÷R2
  - **Eliminarea duplicarilor**: $\partial\ (R)$
  - **Sortarea inregistrarilor**: $S_{\{lista\}}(R)$
  - **Gruparea**: $G_{\{lista1\}\ group\ by\ \{lista2\}}(R)$

Algoritmi pentru evaluarea operatorilor relationali

- **A1. Table Scan**
- **A2. Index Seek**
- **A3. Index Scan**
- **A4. Cross Join**
- **A5. Nested Loops Join**
- **A6. Indexed Nested Loops Join**
- **A7. Merge Join**
- **A8. Hybrid Merge Join**
- **A9. Hash Join**
- **Joinuri externe**
- **Joinuri multiple**
- **Operatii pe mulimi de inregistrari: R1 ∪ R2, R1 - R2, R1 ∩ R2**

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

A10. Sortarea

- Ceruta explicit:

```
select ... order by lista
select distinct ...
```

- Necesara la evaluarea unor operatori (join, intersectie, reuniune, diferenta)

- Necesara la gruparea inregistrarilor:

```
select ... group by lista
```

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

- A10. Sortarea
- Varianta 1: sursa pentru sortare incape in memoria interna, atunci se foloseste un algoritm de **sortare interna**
- Varianta 2: sursa pentru sortare nu incape in memoria interna:
  - se creaza **monotonii** (inregistrari din sursa care incape in memoria interna si se ordoneaza)
  - monotoniile rezultate se distribuie in fisiere temporare
  - cu un algoritm de **sortare externa** se face o interclasare a monotoniilor

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

A10. Sortarea

- **Precizari**. In exemplele urmtoare (la descrierea algoritmilor de sortare externa):
  1. se foloseste un tabel cu 3100 înregistrari
  2. o monotonie poate contine cel mult 100 înregistrari
  3. pentru distribuirea monotoniilor se folosesc patru fisiere
  4. cu x^y s-a notat repetarea de **y** ori a unei monotonii de lungime relativa **x** (unde monotonia de lungime relativa 1 este cea rezultata printr-un algoritm de sortare interna)
- In continuare se vor descrie urmatorii algoritmi de sortare externa:
  - Interclasarea echilibrata
  - Interclasarea polifazica
  - Interclasarea in cascada

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

A10. Sortarea

- 1. Interclasarea echilibrata
  - monotoniile initiale se distribuie relativ uniform pe jumatate din fisiere
  - prin interclasari se creaza monotonii care se distribuie pe fisierele ramase
    - aceasta etapa se repeta pana la obtinerea unei singure monotonii

Exemplu:

F1	F2	F3	F4	Observaţii
1^{16}	1^{15}	-	-	distribuirea monotoniilor
-	-	2^{8}	$2^{7}1^{1}$	interclasare, alternativ pe F3, F4; copierea unei monotonii
4^{4}	$4^{3}3^{1}$	-	-	interclasare, alternativ pe F1, F2
-	-	8^{2}	$8^{1}7^{1}$	interclasare, alternativ pe F3, F4
16^{1}	15^{1}	-	-	interclasare, alternativ pe F1, F2
-	-	31^{1}	-	interclasare, **obţinere rezultat final** în F3

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

A10. Sortarea

- 2. Interclasarea polifazica
  - Monotoniile se distribuie in fisiere, dupa o anumita regula (trebuie determinata o configuratie initiala), iar dupa distribuire ramane liber un singur fisier
  - In fisierul liber se creaza monotonii prin interclasare din toate celelalte fisiere pana la eliberarea unui fisier
    - aceasta etapa se repeta pana la obtinerea unei singure monotonii

Exemplu:

F1	F2	F3	F4	Observaţii
1^{13}	1^{11}	1^{7}	-	distribuirea monotoniilor
1^{6}	1^{4}	-	3^{7}	interclasare în F4 până la golire F3
1^{2}	-	5^{4}	3^{3}	interclasare în F3 până la golire F2
-	9^{2}	5^{2}	3^{1}	interclasare în F2 până la golire F1
17^{1}	9^{1}	5^{1}	-	interclasare în F1 până la golire F4
-	-	31^{1}	-	interclasare, **obţinere rezultat final** în F4

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

Probleme:

a. Determinarea **configuratiei initiale** (ca numar de monotonii)

F1	F2	F3	F4	
1	-	-	-	monotonia din F1 se obţine din celelalte fişiere
-	1	1	1	monotonia din F4 se obţine din celelalte fişiere
1	2	2	-	monotoniile din F3 se obţin din celelalte fişiere
3	4	-	2	monotoniile din F2 se obţin din celelalte fişiere
7	-	4	6	monotoniile din F1 se obţin din celelalte fişiere
-	7	11	13	monotoniile din F4 se obţin din celelalte fişiere
13	20	24	-	monotoniile din F3 se obţin din celelalte fişiere
c_n	b_n	a_n	-	$a_n \geq b_n \geq c_n$
c_n+a_n	b_n+a_n	-	a_n	

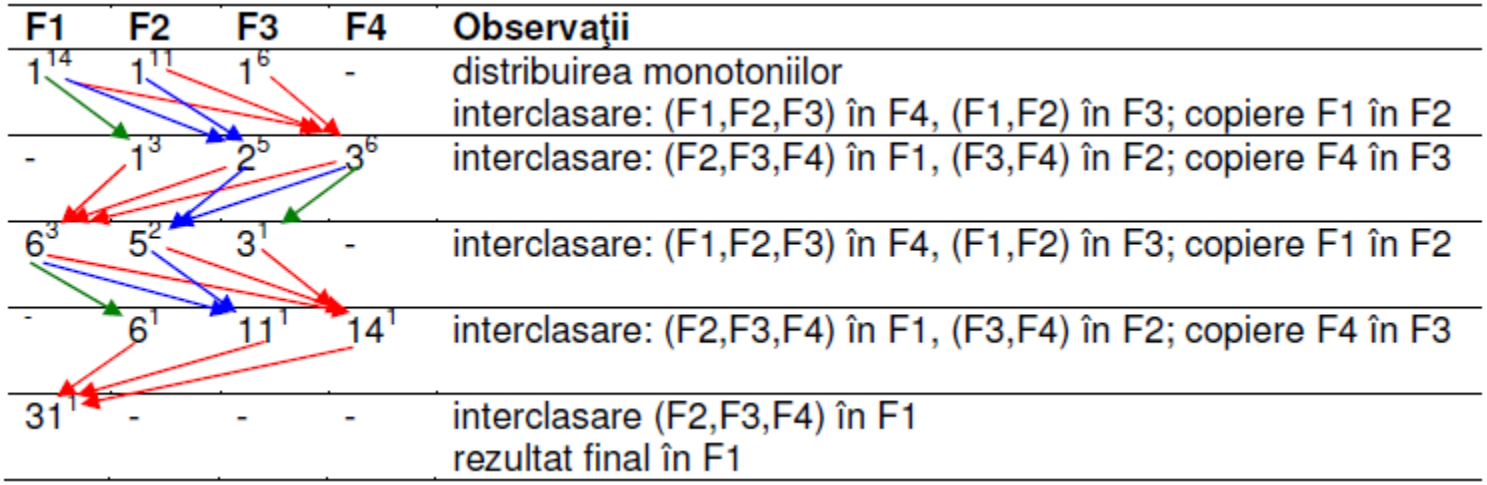
- **Procedeul se poate folosi pentru orice numar de fisiere**

b. Rezolvarea cazului in care nu exista suficiente monotonii pentru crearea unei "**distributii initiale perfecte**": folosirea unor "monotonii vide", care se interclaseaza cu monotonii reale

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

A10. Sortarea

- 3. Interclasarea in cascada
  - Monotoniile se distribuie in fisiere, dupa o anumita regula (este necesara o configuratie initiala pentru distribuirea monotoniilor), iar în final ramane liber un singur fisier
  - In fisierul ramas liber se creeaza monotonii prin interclasare pana la eliberarea unui fisier. Se continua interclasarea in fisierul eliberat cu monotonii din restul de fisiere. Aceasta etapa se repeta pana raman monotonii intr-un singur fisier, care se copiaza in fisierul liber
  - Etapa precedenta se repeta pana la obtinerea unei singure monotonii finale



F1	F2	F3	F4	Observaţii
1^{14}	1^{11}	1^{6}	-	distribuirea monotoniilor interclasare: (F1,F2,F3) în F4, (F1,F2) în F3; copiere F1 în F2
-	1^{3}	2^{5}	3^{6}	interclasare: (F2,F3,F4) în F1, (F3,F4) în F2; copiere F4 în F3
6^{3}	5^{2}	3^{1}	-	interclasare: (F1,F2,F3) în F4, (F1,F2) în F3; copiere F1 în F2
-	6^{1}	11^{1}	14^{1}	interclasare: (F2,F3,F4) în F1, (F3,F4) în F2; copiere F4 în F3
31^{1}	-	-	-	interclasare (F2,F3,F4) în F1 rezultat final în F1

Algoritmi pentru evaluarea operatorilor relationali: Algoritmi de sortare

Probleme:

- a. Determinarea configuratiei initiale

F1	F2	F3	F4
1	-	-	-
-	1	1	1
1	1	1	
1	1		
1			
3	2	1	-
	3	3	3
		2	2
			1
-	3	5	6
6	6	6	
5	5		
3			
14	11	6	-
	14	14	14
		11	11
			6
-	14	25	31

c_n	b_n	a_n	-	$a_n \geq b_n \geq c_n$
$c_n+b_n+a_n$	b_n+a_n	-	a_n	

- **Observatie.** Procedeul se poate folosi pentru orice numar de fisiere
- b. Rezolvarea cazului in care nu exista suficiente monotonii pentru o "distributie perfecta"

Forma optimala pentru expresia din algebra relationala

- **Instructiunea SQL** se transforma intr-o **expresie din algebra relationala** (se poate genera usor, conform unor reguli de transformare pentru fiecare clauza din instructiunea SQL)
- **Fara informatii** din baza de date, **expresia relationala se poate transforma la o forma optimala** (algoritmul de evaluare are o complexitate mai mica)
- Se folosesc anumite reguli de transformare (proprietati matematice ale operatorilor relationali)
- Fiecare SGBD foloseste anumite reguli de transformare
- La scrierea (generarea) unei instructiuni SQL trebuie luate in considerare si optimizarile pe care le face serverul de date

Forma optimala pentru expresia din algebra relationala

- **RI**. $\sigma_C(\Pi_\alpha(R)) = \Pi_\alpha(\sigma_C(R))$

  - selectia reduce numarul de inregistrari pentru proiectie, deci proiectia in varianta din dreapta va analiza mai putine inregistrari
  - se poate crea un algoritm care evalueaza cei doi operatori printr-o singura parcurgere a relatiei R (optimizare in unele SGBD)

Forma optimala pentru expresia din algebra relationala

- **R2**. In loc de doua parcurgeri se face una singura:

$$\sigma_{C1}\left(\sigma_{C2}(R)\right) = \sigma_{C1-and-C2}(R)$$

$$\Pi_{\alpha}\left(\Pi_{\beta}(R)\right) = \Pi_{\alpha \cap \beta}(R)$$

- **R3**. Inlocuire produs cartezian si selectie cu join conditional (pentru joinul conditional exista algoritmi care nu evalueaza produsul cartezian)

$$\sigma_{C}(R \times S) = R \otimes_{C} S$$

unde C este conditia de legatura dintre R si S

Forma optimala pentru expresia din algebra relationala

- **R4**. Folosirea unor proprietati de distributivitate (a unui operator unar "**f**" relativ cu un operator binar "⊙"):

$$f(a \odot b) = f(a) \odot f(b)$$

  - **R4.a**. σ este distributiv fata de ∪, ∩, - (sursele R si S trebuie sa aiba scheme compatibile)

$$\sigma_C(R \cup S) = \sigma_C(R) \cup \sigma_C(S)$$
$$\sigma_C(R \cap S) = \sigma_C(R) \cap \sigma_C(S)$$
$$\sigma_C(R - S) = \sigma_C(R) - \sigma_C(S)$$

  - **R4.b**. σ Este distributiv fata de "**x**":

$$\sigma_C(R \times S) = \sigma_C(R) \times \sigma_C(S)$$

Forma optimala pentru expresia din algebra relationala

Cazuri particulare (R4):

- Daca in C se folosesc numai atribute din R: $\sigma_C(R \times S) = \sigma_C(R) \times S$

- In cazul in care ***C = C1 and C2***, iar in C1 se folosesc numai atribute din R si în C2 se folosesc numai atribute din S:

$$\sigma_{C1-and-C2}(R \times S) = \sigma_{C1}(R) \times \sigma_{C2}(S)$$

- In cazul in care ***C = C1 and C2***, unde **C2** este o conditie de legatura intre R si S:

$$\sigma_{C1-and-C2}(R \times S) = \sigma_{C1}(R \otimes_{C2} S)$$

Forma optimala pentru expresia din algebra relationala

- **R4**. Folosirea unor proprietati de distributivitate (a unui operator unar "**f**" relativ cu un operator binar "⊙"):

$$f(a \odot b) = f(a) \odot f(b)$$

  - **R4.c**. Π este distributiv fata de $\cup$, $\cap$:

$$\Pi_\alpha(R \cup S) = \Pi_\alpha(R) \cup \Pi_\alpha(S)$$
$$\Pi_\alpha(R \cap S) = \Pi_\alpha(R) \cap \Pi_\alpha(S)$$

  - **Observatie.** Π nu este distributiv fata de "-“

  - **Exemplu:** $\Pi_{(x)}\left(\begin{bmatrix} x & y \\ 1 & 2 \\ 1 & 3 \\ 2 & 3 \end{bmatrix} - \begin{Bmatrix} x & y \\ 2 & 3 \\ 1 & 4 \end{Bmatrix}\right)$

Forma optimala pentru expresia din algebra relationala

- **R4**. Folosirea unor proprietati de distributivitate (a unui operator unar "**f**" relativ cu un operator binar "⊙"):

$$f(a \odot b) = f(a) \odot f(b)$$

  - **R4.d**. Π si s (sortarea) sunt distributive fata de "join" (fata de orice operator de join)
    - Pentru optimizare se mai poate face o transformare

$$\Pi_{\alpha}(R - join - S) = \Pi_{\alpha}\left(\Pi_{\alpha 1}(R) join \Pi_{\alpha 2}(S)\right)$$

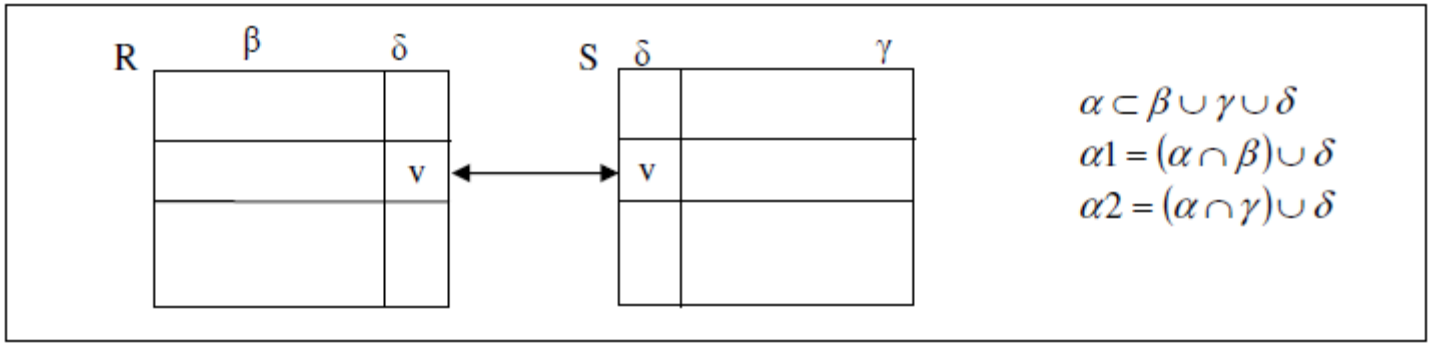
unde:

$$\alpha 1 = \left((atributele - din - \alpha) \cap (atributele - din - R)\right) \cup (atributele - pt - join)$$

$$\alpha 2 = \left((atributele - din - \alpha) \cap (atributele - din - S)\right) \cup (atributele - pt - join)$$

Forma optimala pentru expresia din algebra relationala

- **Obs.** La $\alpha 1$ si $\alpha 2$ se vor adauga atributele utile ***operatorilor precedenti (in expresia relationala).*** Din sursele R si S se selecteaza numai atributele utile pentru join si cele dorite in final



Forma optimala pentru expresia din algebra relationala

- **Asociativitate** si **comutativitate** pentru unii operatori relationali
  - **R5. a**. Asociativitate si comutativitate pentru ∪ si ∩
  - **R5. b**. Asociativitate pentru produs cartezian si join natural
  - **R5. c**. Rezultate "echivalente" (aceleasi înregistrari, dar coloanele apar în alta ordine) la comutarea operanzilor pentru "x" si "join"

Obs. Este posibil ca sa nu se poata folosi asociativitatea pentru "join conditional" (depinde de conditiile de legatura)

R x S = S x R cu algoritmul **Cross Join** este importanta ordinea surselor de date

R join S = S join R cu algoritmul **Hash Join** este important ca functia hash sa fie folosita pentru sursa cu dimensiune mai mica

Forma optimala pentru expresia din algebra relationala

- **<u>R6</u>. Tranzitivitatea** unor operatori relationali pentru operatori de join - pot apare unele filtrari suplimentare inainte de join:
  - **(A>B and B>3) ≡ (A>B and B>3 and A>3)**

A apare in R, B apare in S:

$$R \otimes_{A>B\ and\ B>3} S = \left(\sigma_{A>3}(R)\right) \otimes_{A>B} \left(\sigma_{B>3}(S)\right)$$

  - **(A=B and B=3) ≡ (A=B and B=3 and A=3)**

A apare in R, B apare in S:

$$R \otimes_{A=BandB=3} S = \left(\sigma_{A=3}(R)\right) \otimes_{A=B} \left(\sigma_{B=3}(S)\right)$$

Forma optimala pentru expresia din algebra relationala

- **R7**. Evaluare $\sigma_C(R)$, unde $C \equiv \left(R.A \in \delta\left(\Pi_{\{B\}}(S)\right)\right)$

- Pentru a nu evalua C pentru fiecare inregistrare din R, evaluarea initiala este echivalenta cu: $R \otimes_{R.A=S.B} \left(\delta\left(\Pi_{\{B\}}(S)\right)\right)$

Forma optimala pentru expresia din algebra relationala

- **Exemplu** pentru urmatoarea bdr (cheile primare sunt subliniate, cheile externe sunt marcate cu albastru):
  - **studenti(cnp, nume, prenume, grupa, media,…)**
  - **sectii(cod, denumire,…)**
  - **grupe(cod, sectia, anstudiu,…)**

- **Interogare**. Se cer studentii (nume, prenume, anstudiu, denumire sectie, media) de la o sectie data (ex. cu codul 2, poate fi parametru), cu media cel putin 9 (poate fi parametru). Inregistrarire se cer ordonate alfabetic pe sectii si ani de studiu

```
select nume,prenume, anstudiu, denumire, media
from studenti, grupe, sectii
where studenti.grupa= grupe.cod and grupe.sectia= sectii.cod and sectia=2
      and media>=9
order by anstudiu, nume, prenume
```

Forma optimala pentru expresia din algebra relationala

- Daca notam:

`C = (studenti.grupa= grupe.cod and grupe.sectia= sectii.cod and sectia=2 and media>=9)`

α = `{anstudiu, nume, prenume}` - atributele de la sortare

β = `{nume, prenume, anstudiu, denumire, media}`- atributele de la selectie

atunci expresia relaionala atasata interogarii este:

$$e = \Pi_{\beta}\left(S_{\alpha}\left(\sigma_{c}(studenti \times grupe \times sec\,t\,ii)\right)\right)$$

Forma optimala pentru expresia din algebra relationala

Conform regulilor precedente vom face unele transformări.

- asociativitate pentru "x":

studenti x grupe x sectii = (studenti x grupe) x sectii

sau:

studenti x grupe x sectii = studenti x (grupe x sectii)

- distributivitate σ faţă de "x", folosirea unui caz particular şi folosirea tranzitivităţii operatorului de egalitate:

(**grupe.sectia = sectii.cod** and **sectia=2**)
≡ (**grupe.sectia = sectii.cod** and **sectia=2 and sectii.cod=2**)

studenti.grupa = grupe.cod and **grupe.sectia = sectii.cod** and **sectia=2** and **media>=9** and sectii.cod=2				
C1	C2	C3	C4	C5

σ_C((studenti x grupe) x sectii)=

= $\sigma_{C1 \text{ and } C2}$(($\sigma_{C4}$(**studenti**) x σ_{C3}(**grupe**)) x σ_{C5}(**sectii**))

sau:

= $\sigma_{C1 \text{ and } C2}$ (σ_{C4}(**studenti**) x (σ_{C3}(**grupe**) x σ_{C5}(**sectii**)))

Forma optimala pentru expresia din algebra relationala

- înlocuire selecţie şi produs cartezian prin join condiţional:

$= ((\sigma_{C4}(\text{studenti})) \otimes_{C1} (\sigma_{C3}(\text{grupe}))) \otimes_{C2} \sigma_{C5}(\text{sectii})$

sau:

$= (\sigma_{C4}(\text{studenti})) \otimes_{C1} ((\sigma_{C3}(\text{grupe})) \otimes_{C2} (\sigma_{C5}(\text{sectii})))$

Pe baza informaţiilor statistice din baza de date se alege una dintre variante (vom lua în considerare prima variantă).

$\Rightarrow e = \prod_{\beta}(s_{\alpha}(((\sigma_{C4}(\text{studenti})) \otimes_{C1} (\sigma_{C3}(\text{grupe}))) \otimes_{C2} (\sigma_{C5}(\text{sectii})))$

- distributivitate $\prod$ faţă de "join":

$\beta 1 = \{\text{nume, prenume, media, grupa}\}$ - utile pentru β şi join

$\beta 2 = \{\text{cod, denumire}\}$ - utile pentru β şi join

$\beta 3 = \{\text{cod, sectia, anstudiu}\}$ - utile pentru β şi join

$e = \prod_{\beta}(s_{\alpha}(((\prod_{\beta 1}(\sigma_{C4}(\text{studenti}))) \otimes_{C1} (\prod_{\beta 2}(\sigma_{C3}(\text{grupe})))) \otimes_{C2} (\prod_{\beta 3}((\sigma_{C5}(\text{sectii}))))))$

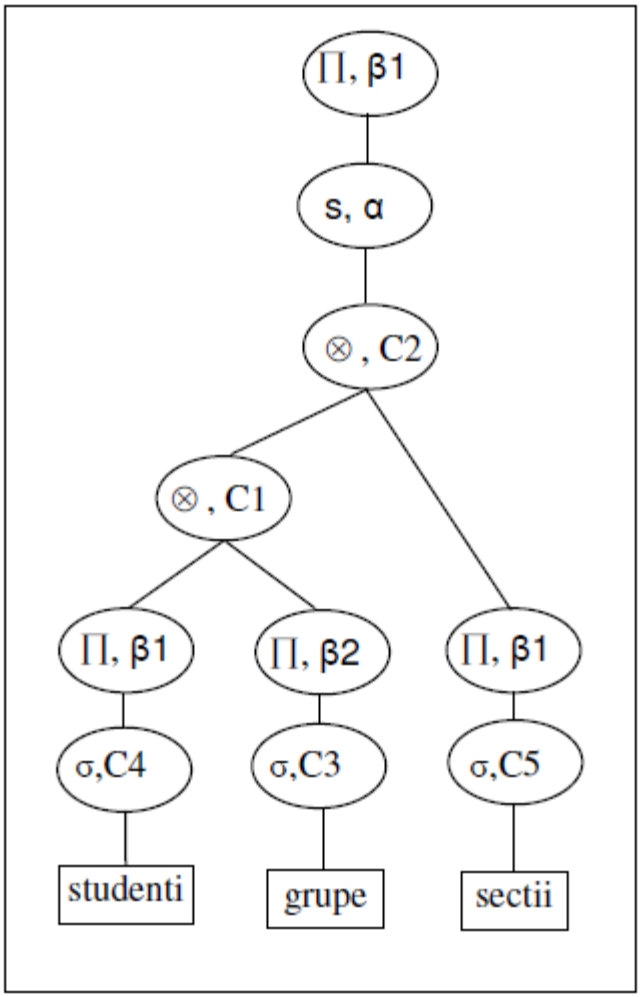
Forma optimala pentru expresia din algebra relationala

Ultima expresie corespunde la instrucţiunea:

```
select nume,prenume, anstudiu, denumire, media
from
((select nume, prenume, media, grupa from studenti where media>=9) st
  inner join
 (select cod, sectia, anstudiu from grupe where sectia=2) gr
  on st.grupa = gr.cod
)
inner join
(select cod, denumire from sectii where cod=2) se
on gr.sectia = se.cod
order by anstudiu, nume, prenume
```

Pentru ultima formă a expresiei din algebra relaţională se poate construi un arbore de evaluare.

Forma optimala pentru expresia din algebra relationala



Cu informaţii din dicţionarul bazei de date şi cu eventuale informaţii statistice, din ultima formă a expresiei se poate genera un plan de execuţie, unde fiecare **operator relaţional** se înlocuieşte cu un **algoritm de evaluare**.

Bibliografie

- Leon Tambulea, Curs de Baze de Date, UBB Cluj
- [Si10] SILBERSCHATZ A., KORTZ H., SUDARSHAN S., *Database System Concepts*, McGraw-Hill, 2010, cap. 13
- [Si10a] SILBERSCHATZ A., KORTZ H., SUDARSHAN S., http://codex.cs.yale.edu/avi/dbbook/db6/slide-dir/PDF-dir/ch13.pdf, 2010
- [Ga08] GARCIA-MOLINA, H., ULLMAN, J., WIDOM, J., *Database Systems: The Complete Book*, Pearson Prentice Hall, 2008, cap.16
- [Da04] DATE, C.J., An Introduction to Database Systems (8th Edition), Addison-Wesley, 2004, cap 13
- [Kn76] KNUTH, D.E., *Tratat de programare a calculatoarelor. Sortare si cautare*. Ed. Tehnica, Bucuresti 1976