Capitolul 3: Interogarea bazelor de date

- Limbaje de interogare
- Algebra relationala si calculul relational
- Operatiile pe multimi ale algebrei relationale
  - Reuniunea
  - Intersecția
  - Diferenta
  - Produsul Cartesian
- Operatiile speciale ale algebrei relationale
  - Selectia
  - Proiectia
  - Jonctiunea
  - Diviziunea
- Interogarea bazelor de date
  - Interogarea intr-o singura relatie
  - Interogarea in doua sau mai multe relatii

Limbaje de interogare

- ***Interogarea*** (*query*)*: operația prin care se obțin informatiile dorite dintr-o bază de date, selectate conform unui anumit criteriu* (*condiție*);
- Limbaje de interogare: abstracte si “concrete” (reale - implementari in diferite SGBD-uri)
- Limbaje de interogare abstracte: algebra relationala si calculul relational
- ***Algebra relationala*** (*relational algebra*) **-** constă dintr-o mulțime de operații care au ca operanzi relații, iar rezultatul este tot o relație
- ***Calculul relațional*** (*relational calculus*) - bazat pe calculul predicatelor - exprimă o interogare definind rezultatul dorit ca expresie de calcul relațional
- ***Calculul relational al tuplurilor*** foloseste *variabile de tuplu* (variabile definite pe mulțimea tuplurilor unei relații)
- ***Calculul relational al domeniilor*** foloseste *variabile de domeniu* (variabile definite pe domenii de definiție ale atributelor)
- Cele trei limbaje formale sunt echivalente din punct de vedere al interogarilor
- ***Limbajele de interogare reale*** sunt definite pe baza unuia sau altuia din limbajele de interogare abstracte, sau pe o combinație a acestora.
- De exemplu, limbajul SQL2 este în cea mai mare parte bazat pe algebra relațională, dar mai conține şi construcții derivate din calculul relațional.

Algebra relațională

- Algebra relațională (*relational algebra*) - interogările sunt expresii compuse din operatii care au ca operanzi relatii si rezultatul este o relatie
- Operatiile algebrei relationale: operatii pe multimi si operatii speciale, la care se adauga operatia de redenumire (*rename*) a atributelor (E.Codd)
- Operatiile relationale pe multimi acționează asupra relațiilor văzute ca mulțimi (de tupluri), fără a lua în considerație compoziția fiecărui tuplu; acestea sunt:
  - Reuniunea
  - Intersecția
  - Diferența
  - Produsul cartezian
- Operațiile relaționale speciale iau în considerație compoziția tuplurilor, formate din valori ale atributelor relațiilor; acestea sunt:
  - Restricția
  - Proiecția
  - Joncțiunea
  - Diviziunea
- Proprietatea de închidere: operanzii (unul sau doi operanzi) sunt relații, rezultatul este o relație; această proprietate permite operații imbricate: proiecția unei joncțiuni etc.

Operația de reuniune

- ***Reuniunea*** (*set-union*) a două relații compatibile r(R) şi s(S):

  $q = r \cup s = \{ t \mid t \in r \text{ or } t \in s\}$

- Pentru ca r si s sa fie compatibile, trebuie ca:
  - r si s să aiba acelasi grad (acelasi numar de atribute)
  - Atributele corespondente (în ordine pozitională) să fie compatibile (adica sa fie definite pe domenii compatibile ca tip de date si semantic), nu ca denumire
- Tuplurile care aparțin ambelor relații se introduc în relația rezultat o singură dată (nu se duplică)
- Relatia rezultat are un numar de tupluri (cardinalitatea) mai mic sau egal cu suma numerelor de tupluri ale celor doi operanzi
- Exemplu:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

A	B
α	1
α	2
β	1
β	3

$r \cup s$

Operațiile de intersecție și diferență

- ***Intersectia*** (*set-intersection*) a două relații compatibile r(R) şi s(S):

  $q = r \cap s = \{ t \mid t \in r \text{ and } t \in s\}$

Exemplu:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

A	B
α	2

$r \cap s$

- ***Diferența*** (*set-difference*) a două relații compatibile r(R) şi s(S):

  $q = r - s = \{ t \mid t \in r \text{ and } t \notin s\}$

- Exemplu:

A	B
α	1
α	2
β	1

r

A	B
α	2
β	3

s

A	B
α	1
β	1

r - s

- Reuniunea si intersectia sunt comutative si asociative ($r \cup s = s \cup r$; $r \cup (s \cup t) = (r \cup s) \cup t$); diferenta nu este nici nici comutativa, nici asociativa

Operatia de produs Cartesian

- ***Produsul Cartesian*** (*Cartesian-Product*) a două relații r(R) şi s(S):

  $q = r \times s = \{ tp \mid t \in r \text{ and } p \in s\}$, $Q = R \cup S$

- Se presupune că multimile R si S sunt disjuncte, adica $R \cap S = \varnothing$
- Daca multimile de atribute R si S nu sunt disjuncte (adica exista atribute cu aceeasi denumire in cele doua relaii), atunci unele atribute:
  - se pot redenumi (RENAME nume_atribut AS noul_nume_atribut) sau
  - se pot califica cu numele relatiei careia ii apartin (folosind operatorul “punct”)
- Tuplurile relației rezultat se obtin prin concatenarea valorilor atributelor fiecărui tuplu din prima relație cu valorile atributelor tuturor tuplurilor din a doua relație
- Relația rezultată are numărul de tupluri (cardinalitatea) egal cu produsul numarului de tupluri ale relatiilor operand
- Exemplu:

A	B
α	1
β	2

r

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

s

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

r x s

Exprimarea operatiilor pe multimi in SQL (1)

- ***Reuniunea***:

  SELECT lista_coloane1 FROM tabel1 [WHERE condiție1]
  UNION
  SELECT lista_coloane2 FROM tabel2 [WHERE condiție2];

  Exemplu (MySQL, Intreprindere):

  SELECT Nume, Prenume, Adresa FROM FURNIZORI
  UNION
  SELECT Nume, Prenume, Adresa FROM CLIENTI;

  Afiseaza numele tuturor furnizorilor si al clientilor, precum si adresa acestora

- ***Intersectia***:

  SELECT lista_coloane1 FROM tabel1 [WHERE condiție1]
  INTERSECT
  SELECT lista_coloane2 FROM tabel2 [WHERE condiție2];

- ***Diferenta***:

  SELECT lista_coloane1 FROM tabel1 [WHERE condiție1]
  MINUS
  SELECT lista_coloane2 FROM tabel2 [WHERE condiție2];

Exprimarea operatiilor pe multimi in SQL (2)

- ***Produsul Cartesian***:

  SELECT * from R, S;

  SELECT lista_col_R, lista_col_S from R, S;

  Exemplu (MySQL, Intreprindere):

  SELECT * FROM ANGAJATI, SECTII;

  SELECT IdAngajat, Nume, Prenume, DataNasterii, Adresa, Functia,

  Salariu, ANGAJATI.IdSectie, SECTII.IdSectie, Denumire, Buget

  FROM ANGAJATI, SECTII;

- Produsul Cartesian este implementat in toate SGBD-urile (instructiunea SQL SELECT)
- In sistemul Oracle sunt implementate toate operatiile pe multimi
- In alte SGBD-uri nu sunt implementate toate operațiile pe mulțimi : in SQL Server 2000 si in MySQL 5.0 nu sunt implementate operațiile INTERSECT şi MINUS

Operația de selecție

- ***Selecța*** (sau *restricția – select, restriction*) într-o relatie r(R) - definita astfel:

  $\sigma_p(r) = \{t \mid t \in r \text{ and } p(t)\}$

  unde p, *predicatul selectiei*, este o expresie logică compusă din termeni conectati prin operatorii and, or, not (și, eventual, paranteze)

- Fiecare termen este o valoare logică obținută ca rezultat al unei operații de comparație de forma:

  <atribut> *op* <atribut> sau

  <atribut> *op* <constanta>, unde

  op este un operator de comparatie aritmetic (=, ≠, >, ≥. <. ≤) sau special(IS NULL etc.)
  Tuplul t este selectat (introdus in rezultat) daca p(t) = TRUE

- În limbajul SQL:

  SELECT * FROM tabel WHERE p;

  În termenii folosiți în limbajul SQL, *restricția selectează o parte din liniile tabelului operand*

- Exemple (MySQL - WORLD):

  SELECT * FROM city where CountryCode='ROM';

  SELECT * FROM country WHERE Continent='Europe';

  SELECT * FROM country WHERE Continent='Europe' and Population > 10000000;

Operatia de proiecție

- ***Proiectia*** (*project*) pe atributele A1, A2, .. Ak intr-o relatie r(R) se notează:

  $\Pi_{A1, A2, \ldots Ak}$ (r), unde A1 $\in$ R, A2 $\in$ R, …Ak $\in$ R

- Rezultatul este o relatie cu k atribute, cele din lista data
- Daca {A1, A2, …Ak} nu contine o supercheie, pot sa apara tupluri duplicat; teoretic, tuplurile duplicat se elimina, dat fiind ca rezultatul este o multime
- Exemplu:

r

A	B	C
α	10	1
α	20	1
β	30	1
β	40	2

A	C
α	1
α	1
β	1
β	2

=

A	C
α	1
β	1
β	2

$\Pi_{A,C}$ (r)

- In limbajul SQL proiectia se exprima astfel:

  SELECT DISTINCT A1, A2, …Ak FROM R

  Daca nu se introduce parametrul DISTINCT, nu se elimina tuplurile duplicat

- Exemplu (MySQL, WORLD):

  SELECT DISTINCT CountryCode FROM City;

Operația de joncțiune naturală (1)

- Joncțiunea naturală (natural join) combină tuplurile din doua relatii
- Fie multimile de atribute: A = {A1,A2,...Am} , B= {B1,B2,...Bn}, C={C1,C2,…Ck} si doua relatii r(R) si s(S), unde:
  - R ={A, B}, S = {B, C}
  - deci atributele $R \cap S = B =$ {B1, B2,...Bn} sunt comune celor două relații
- Joncțiunea naturală este o relatie q = r >< s, care se obține în felul următor:
  - se calculează produsul cartesian al celor doua relatii: p = r x s, P = {A, R.B, S.B, C};
  - din tuplurile produsului cartesian se selecteza acele tupluri care au valori egale pentru atributele comune (B1, B2,...Bn): R.B = S.B, adică R.B1=S.B1, R.B2=S.B2,..
  - se face proiectia rezultatului pe multimea de atribute $R \cup S$ = {A, B, C}
- Schema relatiei rezultat este $Q = R \cup S$ = {A, B, C}

  $q = r >< s = \Pi_{A,B,C}\, \sigma_{(r.B1=s.B1 \text{ AND } r.B2=s.B2 \ldots \text{ AND } r.Bn = s.Bn)} (r \times s)$
- Atributele comune R.B si S.B trebuie să fie compatibile în cele doua relatii; dacă sunt compatibile, ele se consideră identice chiar dacă au denumiri diferite, si în reuniunea atributelor $R \cup S$ se introduc o singură dată
- Cazul cel mai frecvent de jonctiune naturala: intre doua relatii asociate (1: N), atributul comun fiind cheia străină – cheia primară (candidată) referită

Operatia de jonctiune naturala (2)

- Exemplul 1: r ⋈ s = $\Pi_{A,B,C,D,E}\ \sigma_{(r.D = s.D)}$ (r x s)

A	B	C	D
α	1	α	a
β	2	γ	a
γ	4	β	b
α	1	γ	a
δ	2	β	b

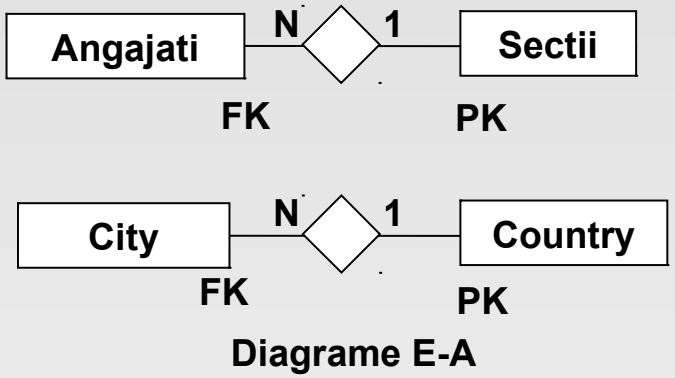
r

D	E
a	α
b	β
c	γ
d	δ
e	∈

s

A	B	C	D	E
α	1	α	a	α
β	2	γ	a	α
γ	4	β	b	β
α	1	γ	a	α
δ	2	β	b	β

r >< s



Diagrame E-A

- In SQL trebuie sa fie introduse explicit lista atributelor rezultatului si conditiile de egalitate ale atributelor comune:

  SELECT A,B,C,R.D,E FROM R, S WHERE R.D = S.D;

- Daca jonctiunea naturala se face intre relații asociate N:1 prin cheie străina, se extind informațiile din relația care refera, cu date din relația referita
- Ex: Angajati **><** Sectii; City **><** Country

  SELECT IdAngajat, ANGAJATI.Nume, Prenume, DataNasterii, Adresa, Salariu, ANGAJATI.IdSectie, SECTII.Nume, Buget FROM ANGAJATI, SECTII WHERE ANGAJATI.IdSectie=SECTII.IdSectie;

  SELECT ID, city.Name Oras, CountryCode 'Cod Tara', city.Population, country.Name Tara, Continent from city, country where city.countryCode=country.CODE order by country.Name;

  Daca nu se afiseaza toate atributele joncțiunii, înseamna ca s-a combinat cu o proiecție

Joncțiuni interne şi externe

- Joncțiunea naturală se mai numeşte şi joncțiune internă şi se mai poate exprima in SQL cu cuvintele cheie INNER JOIN astfel:

SELECT idAngajati, Angajati.Nume, Prenume, DataNasterii, Adresa, CNP, Functia, Salariu, Angajati.idSectii, Sectii.Nume, Buget
from Angajati INNER JOIN Sectii ON Angajati.idSectii=Sectii.idSectii;

- Joncțiunea externă introduce în plus toate liniile care exită în prima relație (pentru LEFT OUTER JOIN) sau în cea de-a doua relație (pentru RIGHT OUTER JOIN) şi pentru care nu există linii în cealălaltă relație care să îndepl. condiția de join;ex:

SELECT idAngajati, Angajati.Nume, Prenume, DataNasterii, Adresa, CNP, Functia, Salariu, Angajati.idSectii, Sectii.Nume, Buget
from Angajati LEFT OUTER JOIN Sectii ON Angajati.idSectii=Sectii.idSectii;

Se vor afişa si liniile din tab. din stânga (Angajați) pentru care nu se indepl. cond. join.

idAngajati	Nume	Prenume	DataNasterii	Adresa	CNP	Functia	Salariu	idSectii	Nume	Buget
1	Ionescu	Ion	1992-01-01	Bucuresti	1234567890	inginer	4000	1	Cercetare	400000
2	Popa	Viorel	1993-02-02	Ploiesti	1122334455	tehnician	2500	2	Proiectare	500000
3	Carol	Mihail	1994-03-03	Craiova	1112223330	inginer	3500	1	Cercetare	400000
4	Marin	Mihnea	1992-01-06	Bucuresti	1234554321	inginer	3500	3	Productie	1000000
5	Ene	Mirela	1995-09-03	Braila	9876543210	secretara	2400	3	Productie	1000000
6	Ionescu	Cezar	1992-06-02	Bucuresti	1243658709	inginer	3800	2	Proiectare	500000
7	Dobre	Paul	1980-06-02	Iasi	9078563412	tehnician	2000	2	Proiectare	500000
8	Oprescu	Victor	0000-00-00	Bucuresti	1900708600	inginer	2500	NULL	NULL	NULL

Operația de diviziune

- Fie relațiile r(R) si s(S), unde:

  R = {A, B} unde A={A1,A2,..Am}, B={ B1,B2,..Bn}

  S = {B}

- Relația q = r ÷ s are schema Q = R – S = {A} si:

$$r \div s = \{ t \mid t \in \prod_{R-S}(r) \wedge \forall u \in s (tu \in r) \}$$

  unde tu inseamna concatenarea tuplurilor t si u; se introduc numai tupluri t distincte

- În limbajul SQL, diviziunea se exprimă printr-o instrucțiune SELECT, introducând explicit toate conditiile impuse valorilor atributelor
- Exemplu:

A	B
α	1
α	2
α	3
β	1
β	2
δ	1
δ	3

r

B
1
2

s

A
α
β

r ÷ s

Concluzii: operatiile algebrei relationale

- Algebra relațională este o colecție de operații asupra relațiilor
- Cele opt operații propuse de E.F.Codd nu constituie o mulțime minimă de operații ale algebrei relaționale
- Mulțimea minimă de operații ale algebrei relaționale consta din cinci operații primitive, pe baza cărora se poate construi orice expresie de algebra relațională:
  - Reuniunea
  - Diferența
  - Produsul Cartesian
  - Restricția (selectia)
  - Proiecția
- Celelalte operații se pot exprima prin intermediul acestora:
  - Intersecția se poate exprima prin expresia: $R \cap S = R - (R - S)$;
  - Joncțiunea este o proiecție a unei restricții a produsului cartezian al relațiilor;
  - Diviziunea este o proiecție a unei restricții asupra relației deîmpărțit
- Si celelalte trei operații sunt deosebit de utile în formularea interogărilor, astfel încât algebra relațională a păstrat toate cele opt operații propuse de E.F.Codd, la care s-a adăugat operația de redenumire a atributelor

Formularea interogarilor

- Interogarea este operatia prin care se obtin informațiile dorite (care indeplinesc o anumita conditie) dintr-o bază de date. O interogare:
  - se formuleaza mai intai în limbaj natural,
  - apoi se exprima într-un limbaj abstract de interogare (algebra relațională sau calculul relațional),
  - se transpune în limbajul de interogare al SGBD-ului folosit (ex., limbajul SQL),
  - iar aplicatia client transmite SGBD-ului instructiunea (sau instructiunile) obtinute
- Sistemul SGBD prelucreaza programul interogarii în mai multe faze:
  - analiza lexicală, sintactică şi semantică
  - optimizarea interogarii
  - generarea codului
  - executia si returnarea rezultatului
- În algebra relațională o interogare se exprima printr-o expresie care defineste următoarele elemente:
  - lista atributelor relației rezultat, care se numesc atribute de proiecție;
  - lista relațiilor din care se extrag informațiile
  - condițiile pe care trebuie să le îndeplinească tuplurile relației rezultat.
- Sunt posibile diferite situații:
  - interogări care se rezolvă în cadrul unei singure relații
  - interogări care se rezolvă folosind două sau mai multe relații ale bazei de date

Interogări într-o singură relație

- Interogarile intr-o relatie folosesc operatiile unare (selectie, proiectie); fie r(R):
  - Expresia de algebra relationala: q = $\prod_{\text{lista\_atribute}} \sigma_{p}(r)$
  - Instructiunea SQL:
  - **SELECT** lista_atribute FROM R WHERE p;
- Exemplul 1: Fie relația ANGAJATI şi interogarea: „Care sunt numele şi prenumele angajaților care au un salariu mai mare sau egal cu 2000?”.
  - Expresia de algebră relațională: q = $\prod_{\text{Nume, Prenume}} \sigma_{\text{Salariu >= 2000}}$ (ANGAJATI)
  - Instrucțiunea SQL:
  - **SELECT** Nume, Prenume FROM ANGAJATI WHERE Salariu >= 2000;
- Exemplul 2: (MySQL - WORLD): “Care sunt numele si populatia oraselor din tara cu codul ‘ROM’ ?”
  - Expresia de algebră relațională: q = $\prod_{\text{Name, Population}} \sigma_{\text{CountryCode='ROM'}}$ (city)
  - Instructiunea SQL:
  - **SELECT** Name, Population FROM city WHERE CountryCode=‘ROM';
- Exemplul 3: Fie relația ANGAJATI şi interogarea: „Care sunt numele, prenumele, adresa și numărul secției în care lucrează angajații?”.
  - Expresia de algebră relațională: q = $\prod_{\text{Nume, Prenume, Adresa, IdSectii}}$(ANGAJATI)
  - Instructiunea SQL:
  - **SELECT** Nume, Prenume, Adresa, IdSectii FROM ANGAJATI;

Interogări în două sau mai multe relații

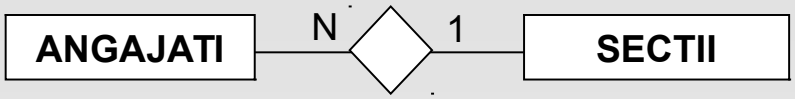
- Daca atributele de proiecție şi atributele din condiția de interogare nu aparțin unei singure relații, pentru rezolvarea interogării trebuie să fie folosite toate acele relațiile care, împreună, conțin atributele şi asocierile necesare
- Conceptual, o astfel de interogare se rezolvă astfel:
  - se construieste mai întâi o relație care să conțină toate atributele implicate prin combinarea relațiilor necesare, folosind operații de produs cartezian sau joncțiuni;
  - in relatia obtinuta se aplica o selectie (restrictie) (cu condiția de interogare *p*);
  - apoi se face proiecția (pe atributele de proiecție).
- Expresia generala de algebra relationala a interogarii este:

  $q = \Pi_{lista\_atribute}\ \sigma_p(r \times s \times t...)$
- Daca intre relatiile din produsul cartesian exista atribute comune care trebuie sa aiba valori egale (de regula, perechile cheie străină - cheie candidata) atunci se pot face operații de joncțiune:

  $q = \Pi_{lista\_atribute}\ \sigma_{p1\ AND\ conditii\text{-}join}(r \times s \times t...) = \Pi_{lista\_atribute}\ \sigma_{p1}\ (r >\!\!< s >\!\!< t\ ...)$
- O astfel de interogare se exprima prin instructiuni SELECT in care:
  - Clauza WHERE combina conditii impuse valorilor atributelor cu conditii de jonctiuni
  - Jonctiunile se pot specifica şi în clauza FROM (cu INNER JOIN, OUTER JOIN)

Interogare în două relații asociate N:1

- Fie interogarea: Care sunt numele, prenumele, adresa şi denumirea secției în care lucrează angajații?
- Expresia de algebră relațională este:



$q = \prod_{Nume,\ Prenume,Adresa,\ Denumire} (ANGAJATI >< SECTII)$

- Instructiunea SQL corespunzatoare acestei interogări:

```
SELECT Nume, Prenume, Adresa, Denumire FROM ANGAJATI, SECTII
    WHERE SECTII.IdSectii = ANGAJATI.IdSectii
```

Se efectueaza o “navigare” în baza de date, pe atributul comun (IdSectii)

ANGAJATI

IdAngajat	Nume	Prenume	DataNasterii	Adresa	Salariu	*IdSectii*

SECTII

Buget	Denumire	IdSectii

- Fie interogarea: Care sunt numele, prenumele, adresa angajaților care lucrează în secția cu denumirea ‘Productie’?

$q = \prod_{Nume,\ Prenume,\ Adresa} \sigma_{Denumire= \text{‘Productie’}} (ANGAJATI >< SECTII)$

```
SELECT Nume, Prenume, Adresa FROM ANGAJATI, SECTII
        WHERE SECTII.IdSectii = ANGAJATI.IdSectii AND Denumire = ‘Productie’;
```

Interogare in trei relatii asociate

- Fie relatiile FILM și ACTOR asociate M:N prin intermediul relației FILM_ACTOR (baza de date SAKILA - MySQL):
  - FILM (film_id, title, description, release_year, ...)
  - ACTOR (actor_id, first_name, last_name, last_update)
  - FILM_ACTOR (film_id, actor_id, last_update)

- Interogarea: "În ce FILME au jucat fiecare din ACTORII din baza de date sakila ?"
- Răspunsul (ca identificatori) este dat de liniile tabelului FILM_ACTOR:

  $q = \prod_{actor\_id,\ film\_id} (film\_actor)$

  In SQL: **SELECT actor_id, film_id from film_actor; -- fig (a) – pagina urmatoare**
- Pentru a afla datele cerute (numele actorilor, ale filmelor) se face joncțiunea rel. FILM_ACTOR cu fiecare din cele două relații, ACTOR și FILM și apoi o proiecție

  $q = \prod_{actor.actor\_id,\ first\_name,\ last\_name,film.film\_id,\ title} (film >< film\_actor >< actor)$
- **In SQL:**

```
select first_name, last_name, actor.actor_id, film.film_id, title
  from film, film_actor, actor
  WHERE film.film_id = film_actor.film_id AND actor.actor_id = film_actor.actor_id
  ORDER BY title,actor.actor_id;
```

Exemplu: interogare in trei relatii (2)

actor_id	first_name	last_name	last_update
1	PENELOPE	GUINESS	2006-02-15
2	NICK	WAHLBERG	2006-02-15
3	ED	CHASE	2006-02-15
4	JENNIFER	DAVIS	2006-02-15
5	JOHNNY	LOLLOBRIGID.	2006-02-15
6	BETTE	NICHOLSON	2006-02-15

film_id	title	description
1	ACADEMY DINOSAUR	A Epic Drama of ...
2	ACE GOLDFINGER	A Astounding Epi...
3	ADAPTATION HOLES	A Astounding Ref...
4	AFFAIR PREJUDICE	A Fanciful Docum...
5	AFRICAN EGG	A Fast-Paced Doc...
6	AGENT TRUMAN	A Intrepid Panor...
7	AIRPLANE SIERRA	A Touching Saga ...

actor_id	film_id
1	1
10	1
20	1
30	1
40	1
53	1
108	1
162	1
188	1
198	1
19	2
85	2

actor_id	first_name	last_name	title	film_id
1	PENELOPE	GUINESS	ACADEMY DINOSAUR	1
10	CHRISTIAN	GABLE	ACADEMY DINOSAUR	1
20	LUCILLE	TRACY	ACADEMY DINOSAUR	1
30	SANDRA	PECK	ACADEMY DINOSAUR	1
40	JOHNNY	CAGE	ACADEMY DINOSAUR	1
53	MENA	TEMPLE	ACADEMY DINOSAUR	1
108	WARREN	NOLTE	ACADEMY DINOSAUR	1
162	OPRAH	KILMER	ACADEMY DINOSAUR	1
188	ROCK	DUKAKIS	ACADEMY DINOSAUR	1
198	MARY	KEITEL	ACADEMY DINOSAUR	1
19	BOB	FAWCETT	ACE GOLDFINGER	2
85	MINNIE	ZELLWEGER	ACE GOLDFINGER	2

Subinterogări

- Subinterogările sunt operații care calculează diferite date (valori scalare, tabele rezultat, număr de elemete etc.) folosite în interogarea de bază
- Subinterogările pot conține la rândul lor alte subinterogări
- Exemplul 1: Care sunt angajații (nume, prenume, adresa) care lucrează în aceeaşi secție cu angajatul cu numele Ionescu şi prenumele Mihai?
  - Se determină printr-o subinterogare în ce secție lucrează angajaul dat
  - Se selectează toți angajații din acea secție:

  SELECT Nume, Prenume, Adresa FROM ANGAJATI

  WHERE IdSectii = (**SELECT IdSectii FROM ANGAJATI WHERE Nume = 'Ionescu' AND Prenume = 'Mihai');**
- Exemplul 2: Care sunt numele, prenumele, denumirea secției şi salariul angajaților care au salariul egal cu salariul maxim pe una din secții:
  - Se determină printr-o subinterogare tabelul cu numarul sectiei și valoarea maxima a salariului în fiecare secție
  - Se selecteză angajații care au salariul maxim in sectia proprie:

```
select nume, prenume, adresa, salariu
from angajati,(select idsectii, max(salariu) salmax
                    from angajati group by idsectii) angmax
where angajati.idsectii = angmax.idsectii and salariu = angmax.salmax;
```