

Indrumar Baze de Date - Capitolul 3

INTEROGAREA BAZELOR DE DATE

Pentru formularea interogarilor bazelor de date s-au dezvoltat doua formalisme, ca limbaje abstracte de interogare, algebra relationala si calculul relational. Algebra relationala exprima interogarile prin aplicarea unor operatori specializati (operatorii algebrei relationale) asupra relatiilor. Rezultatul unei operatii din algebra relationala este tot o relatie, asigurând astfel proprietatea de închidere a operatiilor. Calculul relational este bazat pe calculul predicatelor si exprima o interogare formulând o definitie a rezultatului dorit (de regula o relatie) printr-o expresie de calcul relational. Aceste *limbaje de interogare abstracte*, algebra relationala, calculul relational sunt echivalente din punct de vedere al capacitatii de exprimare a interogarilor, diferentele constând în modul de formulare a acestora. S-a demonstrat ca, pentru orice expresie de algebra relationala, se poate gasi o expresie de calcul relational echivalenta si invers. În continuare se va studia algebra relationala pe care se bazeaza majoritatea comenzilor SQL.

3.1 ALGEBRA RELATIONALA

Algebra relationala consta dintr-o multime de operatii care au ca operanzi relatii, iar rezultatul este tot o relatie. E.F. Codd a propus opt operatii ale algebrei relationale, grupati în doua categorii:

- *Operatii pe multimi*: reuniunea (*union*), intersectia (*intersection*), diferenta (*difference*) si produsul cartezian (*Cartesian product*). Aceste operatii reprezinta adaptarea operatiilor corespunzatoare din teoria multimilor si actioneaza asupra relatiilor vazute ca multimi de elemente (tupluri), fara a lua în considerare compozitia fiecarui element.
- *Operatii relationale speciale*: restrictia (*restriction*), proiectia (*projection*), jonctiunea (*join*) si diviziunea (*division*). Aceste operatii iau în considerare compozitia tuplurilor, formate din valori ale atributelor relatiilor.

Reuniunea (*union*) a doua relatii compatibile R si S este o relatie $T = R \cup S$ care contine toate tuplurile care apartin fie relatiei R, fie relatiei S, fie ambelor relatii. Tuplurile care apartin ambelor relatii se introduc în relatia rezultat o singura data, adica nu se duplica. Operatia de reuniune se exprima în SQL ca o reuniune a doua tabele obtinute ca rezultat a doua comenzi SELECT, cu sintaxa:

```
SELECT lista_coloane_1 FROM tabel_1 [WHERE conditie_1]
UNION
SELECT lista_coloane_2 FROM tabel_2 [WHERE conditie_2];
```

Intersectia (*intersection*) a doua relatii compatibile R si S este o relatie $T = R \cap S$ care contine toate tuplurile care apartin atât relatiei R cât si relatiei S. La fel ca si reuniunea, operatia de intersectie se exprima în SQL ca intersectie a doua tabele obtinute ca rezultat a doua instructiuni SELECT, cu sintaxa:

```
SELECT lista_atribute_1 FROM tabel_1 [WHERE conditie_1]
INTERSECT
SELECT lista_atribute_2 FROM tabel_2 [WHERE conditie_2];
```

Diferenta (*difference*) a doua relatii compatibile R si S este o relatie $T = R - S$ care contine toate tuplurile care apartin relatiei R, dar nu apartin relatiei S. Operatia de diferenta se exprima în SQL ca diferenta a doua tabele obtinute ca rezultat a doua comenzi SELECT, cu sintaxa:

```
SELECT lista_atribute_1 FROM nume_tabel_1 [WHERE conditie_1]
MINUS
SELECT lista_atribute_2 FROM nume_tabel_2 [WHERE conditie_2];
```

Produsul cartezian (*Cartesian product*). Produsul cartezian al doua relatii $R(A_1, A_2, \dots, A_n)$ si $S(B_1, B_2, \dots, B_m)$ este o relatie $T: R \times S = T(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$ care are ca atribute toate atributurile primei relatii plus toate atributurile celei de-a doua relatii, deci gradul relatiei rezultat este egal cu suma gradelor celor doua relatii operanzi. Pentru a se obtine relatia rezultat se combina (se concateneaza) valorile atributelor fiecarui tuplu din prima relatie cu valorile tuturor atributelor unui tuplu din cea de-a doua relatie. În limbajul SQL, produsul cartezian a doua tabele R si S se obtine ca o varianta a instructiunii `SELECT`, într-una din formele:

```
SELECT * FROM R,S;
SELECT lista_coloane FROM R,S;
```

În prima forma, limbajul SQL admite operatia produs cartezian si în situatia în care în cele doua relatii operand exista doua atribute cu acelasi nume, subînțelegându-se calificarea atributelor cu numele fiecărei relatii. Pentru cea de-a doua forma, atributele cu acelasi nume trebuie sa fie calificate cu numele relatiei respective.

Restrictia (*restriction*) este o operatie unara care selecteaza dintre tuplurile relatiei operand acele tupluri care îndeplinesc o conditie data. Operatia de restrictie se noteaza: $\sigma_{\theta}(R)$, unde θ este o expresie booleana specificata asupra atributelor relatiei R . În relatia rezultat sunt selectate acele tupluri ale relatiei R pentru care expresia θ are valoarea 1 (TRUE). Relatia rezultat are aceleasi atribute ca si relatia operand. Operatia de *restrictie* se mai numeste si *selectie* (si, într-adevar, restrictia face o selectie a tuplurilor), dar este mai bine sa fie evitata aceasta denumire care se poate confunda cu instructiunea `SELECT` din SQL, care are rolul de instructiune generala de interogare.

În limbajul SQL restrictia se exprima printr-o forma particulara a instructiunii `SELECT`, în care lista de atribute este formata din toate atributele unei singure relatii, iar clauza `WHERE` este obligatorie si introduce conditia de restrictie:

```
SELECT * FROM nume_tabel WHERE conditie [clauze_secundare];
```

Proiectia este o operatie unara prin care se selecteaza o submultime a atributelor relatiei operand. Notatia obisnuita pentru proiectie este: $\Pi_{\text{lista_atribute}}(\text{nume_relatie})$. Relatia rezultat a operatiei de proiectie contine numai atributele din lista de atribute data ca parametru, care este o submultime nevida a multimii atributelor relatiei operand.

Daca lista atributelor de proiectie este o cheie (sau contine o cheie) a relatiei operand, atunci relatia rezultat are toate tuplurile distincte. Daca lista de atribute nu este o cheie (sau nu contine o cheie) a relatiei operand, atunci este posibil ca prin proiectie sa se obtina doua sau mai multe tupluri identice, dar în relatia rezultat sunt eliminate tuplurile duplicat. În acesta situatie numarul de tupluri ale relatiei rezultat este mai mic decât numarul de tupluri ale relatiei operand.

În limbajul SQL, operatia de proiectie se obtine tot prin instructiunea de interogare `SELECT`; lista de coloane introdusa în instructiunea `SELECT` este lista atributelor de proiectie:

```
SELECT DISTINCT lista_coloane FROM nume_tabel;
```

Daca lipseste clauza `DISTINCT` rezultatul operatiei poate contine tupluri duplicat (deci nu este o relatie în sensul definitiei din modelul relational).

Jonctiunea (*join*) este o operatie binara a algebrei relationale prin care se combina tuplurile a doua relatii într-o singura relatie. Jonctiunea se noteaza cu semnul \bowtie si este o operatie foarte importanta în bazele de date relationale deoarece ea permite prelucrarea asocierilor între relatii. În continuare vor fi prezentate doua forme ale operatiei de jonctiune: θ -jonctiunea si jonctiunea naturala.

θ -Jonctiunea a doua relatii $R(A_1, A_2, \dots, A_n)$ si $S(B_1, B_2, \dots, B_m)$ este o relatie T :

$$R \bowtie_{\theta} S = T(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$$

în care fiecare tuplu este o combinatie a doua tupluri, unul din relatia R (cu atributele A_1, A_2, \dots, A_n), celalalt din relatia S (cu atributele B_1, B_2, \dots, B_m), care satisfac conditia de jonctiune.

Cea mai utilizata forma de θ -jonctiune este *echi-jonctiunea*, în care se foloseste numai operatorul de comparatie de egalitate ($=$).

Jonctiunea naturala. Jonctiunea naturala este o echi-jonctiune în care fiecare pereche de atribute comparate pentru egalitate (în conditia de jonctiune) se înlocuieste cu un singur atribut. Se

poate spune ca jonctiunea naturala este o echi-jonctiune urmata de o proiectie pe multimea atributelor celor doua relatii minus câte un atribut din fiecare pereche de attribute comparate pentru egalitate. Jonctiunea naturala se reprezinta numai cu semnul \bowtie , fara sa mai fie însoțit de conditia de jonctiune, înțelegând prin aceasta ca jonctiunea are loc pe atributul (sau attributele) comune ale celor doua relatii.

Operatia de jonctiune naturala este utilizata pentru a combina date din doua sau mai multe relatii, astfel încât informatia rezultata sa fie cuprinsa într-o singura relatie. În cazul cel mai frecvent, jonctiunea naturala se calculeaza între o relatie care refera si relatia referita, atributul de jonctiune fiind cheia straina (în relatia care refera), respectiv cheia primara (sau candidata) în relatia referita. Rezultatul obtinut reflecta asocierea dintre cele doua relatii.

În limbajul SQL, instructiunea `SELECT` poate exprima o *q-jonctiune* a doua sau mai multe tabele, conditia de jonctiune θ fiind introdusa prin clauza `WHERE`. De exemplu, jonctiunea tabelelor `SECTII` si `ANGAJATI` se poate obtine prin instructiunea:

```
SELECT * FROM ANGAJATI,SECTII WHERE ANGAJATI.IdSectie= SECTII.IdSectie;
```

Daca exista attribute cu aceleasi nume în cele doua tabele, se califica attributele cu numele tabelului respectiv, la fel ca la produsul cartezian.

O jonctiune naturala se poate exprima în limbajul SQL numai în mod explicit, adica trebuie ca lista de attribute a instructiunii `SELECT` sa contina un atribut de jonctiune o singura data, iar în clauza `WHERE` trebuie introdusa conditia de egalitate a atributelor corespondente. De exemplu, jonctiunea naturala `SECTII` \bowtie `ANGAJATI` se poate introduce prin comanda SQL:

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

Diviziunea. Fie doua multimi de attribute: $A = \{A_1, A_2, \dots, A_n\}$ si $B = \{B_1, B_2, \dots, B_m\}$ si doua relatii $R (A \cup B)$ si $S (B)$ astfel încât multimea atributelor relatiei S sa fie o submultime a multimii atributelor relatiei R . Relatia T obtinuta prin operatia de diviziune $T(A) = R \div S$ are ca attribute toate attributele diferentei celor doua multimi de attribute (adica acele attribute care apartin relatiei R si nu apartin relatiei S) si contine acele tupluri t care au proprietatea ca pentru orice tuplu s din S exista un tuplu în R care are atributul B (simplu sau compus) egal cu atributul B al tuplului s :

$$T(A) = R \div S = \Pi_A \sigma_{R.B=S.B} (R)$$

În limbajul SQL, diviziunea se exprima printr-o instructiune `SELECT`, introducând explicit lista atributelor de proiectie si conditia de egalitate a atributelor corespondente din cele doua relatii prin clauza `WHERE`.

3.2 CREAREA INTEROGARILOR

Interogariile se pot crea pe baza operatiilor din algebra relationala sau a celor din calculul relational. Majoritatea comenzilor SQL se bazeaza pe algebra relationala, si numai un numar mic de comenzi (`EXISTS`, `NOT EXISTS`) provin din operatori din calculul relational.

În algebra relationala o interogare se formuleaza printr-o expresie constând dintr-o secventa de identificatori (nume de relatii, nume de attribute), constante (literale), operatori ai algebrei relationale ($\cup, \cap, -, \times, \sigma, \Pi, \bowtie, \div$), operatori de comparatie ($=, \neq, <, \leq, >, \geq$) si operatori logici (`NOT`, `AND`, `OR`).

În expresii se pot folosi si paranteze pentru a specifica o anumita ordine de efectuare a operatiilor. Pentru exprimarea unei interogari printr-o expresie de algebra relationala, trebuie sa fie precizate urmatoarele elemente:

- Lista atributelor relatiei rezultat, care se numeste lista atributelor de proiectie;
- Lista relatiilor din care se extrag informatiile;
- Conditia pe care trebuie sa o îndeplineasca tuplurile relatiei rezultat.

În functie de aceste elemente, se pot studia doua situatii de exprimare a interogarilor: interogari care se rezolva în cadrul unei singure relatii si interogari care se rezolva folosind doua sau mai multe relatii ale bazei de date. Pe lângă acestea, mai exista si interogari imbricate si subinterogari.

Interogari care se rezolva în cadrul unei singure relatii. Daca toate attributele care intervin în interogare (attributele de proiectie si attributele de conditie) sunt attribute ale unei singure relatii R , atunci interogarea se poate rezolva la nivelul acelei relatii, ca o proiectie (pe attributele relatiei rezultat) a restrictiei cu conditia impusa asupra relatiei date:

$$T = \Pi_{A_1, A_2, \dots, A_k} \sigma_{\text{conditie}}(R)$$

Fie, de exemplu, relatia $ANGAJATI(IdAngajat, Nume, Prenume, DataNasterii, Adresa, Salariu, IdSectie)$ si interogarea: "Care sunt numele, prenumele, data nasterii si salariul angajatilor care lucreaza în sectia 1?"

Analizând aceasta interogare se constata ca toate attributele de proiectie (nume, prenume, data nasterii si salariul unui angajat) si atributul din conditia de interogare (numarul sectiei) sunt attribute ale relatiei $ANGAJAT$, deci interogarea poate fi rezolvata la nivelul acestei relatii. Expresia de algebra relationala care exprima interogarea data este:

$$T = \Pi_{Nume, Prenume, DataNasterii, Salariu} \sigma_{IdSectie = 1}(ANGAJAT)$$

Instructiunea SQL care realizeaza aceasta interogare este:

```
SELECT Nume, Prenume, DataNasterii, Salariu
FROM ANGAJATI
WHERE IdSectie = 1;
```

Interogari care se rezolva folosind doua sau mai multe relatii. În situatia în care attributele de proiectie si attributele din conditia de interogare nu apartin unei singure relatii, pentru rezolvarea interogarii trebuie sa fie folosite cel putin toate acele relatii care, împreuna, contin aceste attribute.

Conceptual, o astfel de interogare se rezolva construind mai întâi o relatie care sa contina toate attributele necesare prin combinarea a doua sau mai multe relatii folosind operatii de produs cartezian sau jonctiuni, iar rezultatul interogarii se obtine prin restrictia (cu conditia de interogare) si proiectia (pe attributele de proiectie) a acestei relatii.

Cazul cel mai frecvent de interogare necesita jonctiunea naturala a doua sau mai multe relatii aflate în situatie de referire, folosind perechea de attribute cheia straina - cheia primara referita pentru fiecare operatie de jonctiune.

Fie relatiile $SECTII$ si $ANGAJATI$ si interogarea "Care sunt numele, prenumele, data nasterii si salariul angajatilor care lucreaza în sectia cu numele Productie?".

Attributele de proiectie ($Nume, Prenume, DataNasterii, Salariu$) sunt attribute ale relatiei $ANGAJATI$; atributul numele sectiei (care apare în conditia de interogare) nu se afla în aceeași relatie, ci în relatia $SECTII$ si de aceea, pentru a rezolva aceasta interogare, este necesara combinarea celor doua relatii. Expresia de algebra relationala care exprima interogarea data este:

$$T = \Pi_{ANGAJAT.Nume, Prenume, DataNasterii, Salariu} \sigma_{SECTIE.Nume='Productie'}(ANGAJAT \bowtie SECTIE)$$

Instructiunea SQL care realizeaza aceasta interogare este:

```
SELECT ANGAJAT.Nume, Prenume, DataNasterii, Salariu FROM SECTIE, ANGAJAT
WHERE SECTIE.IdSectie=ANGAJAT.IdSectie AND SECTIE.Nume='Productie';
```

Asa cum sa mai precizat, în SQL trebuie sa fie introdusa explicit conditia de jonctiune naturala ($SECTIE.IdSectie = ANGAJAT.IdSectie$), împreuna (prin conjunctia AND) cu celelalte conditii de interogare ($SECTIE.Nume = 'Productie'$).

Interogari imbricate. Subinterogari. Instructiunile SELECT se pot imbrica pe mai multe niveluri, o instructiune având ca argument rezultatul unei altei instructiuni, numita subinterogare. Exista mai multe moduri de construire a subinterogarilor, una din formele cele mai frecvent folosite fiind urmatoarea:

```
SELECT lista_atribute FROM tabel1
WHERE colx IN (SELECT colx FROM tabel2 WHERE conditie);
```

Într-o astfel de constructie valoarea de comparatie (pentru operatorul de comparatie IN) din clauza WHERE a primei instructiuni SELECT se defineste printr-o subinterogare care consta dintr-o alta instructiune SELECT.

Tot o subinterogare este folosita si în comanda SQL EXISTS, care corespunde cuantificatorului universal din calculul relational. Expresia SQL este:

```
SELECT lista_coloane FROM lista_tabele
WHERE EXISTS(SELECT * FROM R WHERE P(X));
```

este echivalenta cu o formula de calcul relational al tuplurilor în care o variabila de tuplu x cu domeniu al valorilor definit pe relatia R este cuantificata existential. Subinterogarea (interogarea interna) careia i se aplica functia EXISTS este, în mod normal, corelata cu interogarea externa, adica în conditia WHERE se pot include unele coloane ale tabelelor din interogarea externa. Clauza WHERE din interogarea externa este evaluata la TRUE daca subinterogarea returneaza un rezultat nevid, care contine una sau mai multe linii, si ia valoarea FALSE daca subinterogarea nu returneaza nici o linie.

3.3 PARTICULARITATI DE CREARE A INTEROGARILOR IN DIFERITE SISTEME SGBD

Implementarea limbajului SQL în diferite sisteme de gestiune prezinta unele particularitati în ceea ce priveste modul de realizare a operatiilor algebrei relationale si formularea interogarilor.

Operatiile pe multimi nu sunt implementate complet în toate sistemele de gestiune, cu exceptia sistemului Oracle. Produsul cartezian este implementat în toate sistemele, dat fiind ca este cuprins în sintaxa instructiunii SELECT, care nu poate sa lipseasca din nici o implementare, fiind instructiunea de interogare de baza. În MySQL nu sunt implementate operatiile UNION, INTERSECT si MINUS, iar în SQL Server nu este implementate operatiile INTERSECT si MINUS.

În orice sistem de gestiune, operatiile INTERSECT si MINUS se pot crea prin subinterogari cu comanda EXISTS. De exemplu, intersectia si, respectiv diferenta, dintre doua relatii obtinute prin proiectia relatiilor ANGAJATI si FURNIZORI pe attributele Nume si Prenume se pot realiza astfel:

```
SELECT DISTINCT Nume,Prenume FROM ANGAJATI
WHERE EXISTS SELECT * FROM FURNIZORI
WHERE ANGAJATI.Nume = FURNIZORI.Nume
AND ANGAJATI.Prenume = FURNIZORI.Prenume )

SELECT DISTINCT Nume,Prenume FROM ANGAJATI
WHERE NOT EXISTS (SELECT * FROM FURNIZORI
WHERE ANGAJATI.Nume = FURNIZORI.Nume
AND ANGAJATI.Prenume = FURNIZORI.Prenume )
```

3.3.1 CREAREA INTEROGARILOR IN ACCESS

În general, comenzile de interogare se transmit sistemului de gestiune al bazelor de date ca si instructiuni SQL, dar mediul MS Access ofera o modalitate mai simpla de construire a interogarilor si memorare a interogarilor si anume printr-un *limbaj de interogare prin exemple* (Query by Example - QBE). Acest limbaj ofera o interfata utilizator care faciliteaza formularea interactiva a interogarilor folosind variabile de domeniu sau constante pentru a forma modelul (schema conceptuala) a tuplurilor (înregistrarilor) rezultat.

Pentru a crea o interogare folosind limbajul QBE, în fereastra *Database*, se selecteaza panoul *Queries*, si apoi comanda *New*. Pe ecran este afisata caseta de dialog *New Query* care contine mai multe optiuni pentru crearea interogarilor. Pentru exemplificarea modului de creare a unei interogari, se va alege optiunea *Design View*. La comanda *OK* cu aceasta selectie, se deschide caseta de dialog *Show Table* si afiseaza grila de construire a interogarii QBE.

Deoarece majoritatea interogarilor se bazeaza pe datele existente în tabele, mediul Access asteapta ca utilizatorul sa selecteze o sursa de date valida care poate fi atât tabel, dar si o interogare existenta în baza de date, deoarece ambele tipuri returneaza obiecte de tip tabel, din care interogările pot extrage date. Dupa selectarea unei surse de date, caseta de dialog *Show Table* poate fi închisa pentru a elibera ecranul si a vedea întreaga grila QBE. Aceasta grila este formata din doua sectiuni: sectiunea din partea de sus contine tabelele (sau interogările) selectate, cu prezentarea asocierilor dintre acestea, iar sectiunea din partea de jos a ferestrei contine grila interogarii (*Query Grid*) si este cea în care se lucreaza efectiv. Rândurile din grila interogarii sunt folosite astfel:

Field (Câmp). Intrarea în fiecare celulă de pe acest rând este numele unui câmp din sursa de date. Introducerea numelui unui câmp aici se poate face din două motive: fie se urmărește ca datele din câmpul respectiv să apară în rezultatele interogării, fie se dorește sortarea sau alegerea înregistrărilor din sursa de date în funcție de o anumită valoare plasată în acest câmp.

Table (Tabelul). Este sursa de date în care se găsesc câmpurile listate mai sus. Poate fi o tabelă sau o altă interogare.

Sort (Sortare). Reprezintă modul în care trebuie sortate înregistrările returnate de interogare. Intrările valide în această celulă sunt: *ascending* (0-9, A-Z), *descending* (9-0, Z-A) și *not sorted* (nesortate sau necompletate). Sortarea se aplică asupra câmpului afișat deasupra ordinii de sortare; pot fi sortate oricâte câmpuri din grila.

Show (Afișare). Această casetă este validată automat, indicând astfel că datele din câmpul selectat trebuie afișate ca parte a rezultatelor interogării. În cazul în care caseta nu este validată, câmpul respectiv este folosit pentru sortare și/sau criterii, dar nu este afișat pe ecran.

Criteria (Criterii). Un șir introdus în această celulă indică faptul că respectivul câmp trebuie să corespundă șirului pentru ca datele din înregistrările asociate să fie incluse în rezultat. Acest șir poate include oricâte criterii pentru câmpul listat, separate prin cuvântul cheie AND.

or (sau). Orice șir introdus în această celulă face parte din criteriile de selecție pentru câmpul corespunzător, dar aceste criterii sunt diferite de cele introduse în celula anterioară. Dacă datele din câmpul listat respectă criteriile din celula *Criteria* sau pe cele din celula *or*, înregistrarea asociată va fi inclusă în setul de rezultate.

Când se deschide grila QBE, sistemul Access presupune că utilizatorul dorește să construiască o interogare corespunzătoare unei comenzi SQL de tip *SELECT*. Există o serie de caracteristici și funcții suplimentare utile în procesul de construire a interogărilor *SELECT* cum ar fi utilizarea funcțiilor statistice pe linia *Totals* a unei interogări în vederea calculării totalurilor și a altor valori. Dacă în modul de afișare *Design View* a unei interogări se selectează de pe bara de instrumente butonul *Totals* (are o pictogramă reprezentând litera grecească sigma - Σ), în grila QBE apare o linie nouă intitulată *Total*. În câmpul *Total* pot fi selectate dintr-o listă derulantă mai multe funcții care operează asupra câmpului corespunzător și schimbă modul în care acesta este afișat în rezultat: *Group by*, *Sum*, *Avg*, *Min*, *Max*, ș.a.. Pentru crearea celorlalte tipuri de interogări, se selectează din meniul *Query* tipul de interogare dorit ceea ce conduce la actualizarea câmpurilor din grila (*Update Query*, *Delete Query*, etc). Atunci când se specifică mai multe tabele pentru o interogare, tabelele sunt considerate asociate și se efectuează operația de joncțiune (*join*). În Fig. 3.1 este prezentată grila QBE pentru proiectarea interogării “Care sunt numele, prenumele și salariul angajaților care lucrează în secția cu numele *Productie*?” în sistemul de baze de date Microsoft Access.

Field:	Nume	Prenume	Salariu	IdSectie	Nume
Table:	ANGAJATI	ANGAJATI	ANGAJAT	SECTII	SECTII
Sort:	Ascending				
Show:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Criteria:				[ANGAJAT].[IdSectie]	'Productie'
or:					

Fig. 3.1 Grila QBE pentru construirea interogării “Care sunt numele, prenumele și salariul angajaților care lucrează în secția cu numele *Productie*?”

Exercitii - Capitolul 3

Pentru toate exercitiile care urmeaza se va folosi baza de date (schema) `INTREPRINDERE` realizata în lucrarile precedente. Acesta este numele generic al bazei de date; în realitate fiecare student va folosi propria baza de date care, conform conventiilor stabilite, are acelasi nume cu al contului propriu pentru fiecare din sistemele de gestiune studiate.

În fiecare tabel se vor insera mai întâi un numar oarecare de linii, având în vedere referintele între tabele (care sunt reprezentate în Fig. 2.2). Pentru a obtine ca rezultat al interogarilor multimi de linii nevide, este necesar sa fie introduse cel puțin liniile care contin valorile atributelor specificate în interogari.

Pentru sistemele de gestiune Oracle, SQL Server, MySQL, interogarile se pot memora ca scripturi de comenzi care se pot executa din programele utilitare oferite de fiecare sistem de gestiune (*Oracle SQL* Plus Worksheet, SQL Server Query Analyzer si mysql*). Pentru sistemul de gestiune MS Access se poate folosi grila QBE de proiectare vizuala a interogarilor.

3.1 Selectati si afisati attributele `Nume`, `Prenume` ale tuturor tuplurilor din tabelele `FURNIZORI`, `CLIENTI` care au valoarea `'Bucuresti'` a atributului `Adresa`, folosind operatia de reuniune (`UNION`). Retineti care dintre sistemele de gestiune studiate implementeaza aceasta operatie.

3.2 Selectati si afisati produsul cartezian al tabelor `SECTII` si `ANGAJATI`.

3.3 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele furnizorilor care au livrat componente în cantitati egale sau mai mari ca 200 ?" Care va fi rezultatul interogarii?

3.4 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele angajatilor care s-au ocupat de achizitionarea componentelor cu denumirea `'Condensator'` ?" Care va fi rezultatul interogarii? Se precizeaza ca atributul cheie straina `IdAchizitor` din relatia `ACHIITITII` refera atributul `IdAngajat` din relatia `ANGAJATI`.

3.5 Scrieti si executati instructiunea SQL pentru interogarea: "Care sunt numele, prenumele si adresa furnizorilor care au livrat componente cu denumirea `Rezistentă`?". Care va fi rezultatul interogarii?

3.6 Scrieti si executati instructiunea SQL pentru interogarea: "Care sunt numele, prenumele si adresa furnizorilor care au livrat componenta cu denumirea `'Condensator'` în cantitati mai mari sau egale 150?". Care va fi rezultatul interogarii?

3.7 Scrieti si executati comanda SQL pentru interogarea: "Care sunt numele, prenumele si adresa clientilor care au cumparat produsul cu denumirea `'Monitor'` în cantitati mai mari sau egale 10?". Care va fi rezultatul interogarii?

3.8 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele angajatilor care s-au ocupat de vânzarea produselor cu denumirea `'Monitor'` ?" Care va fi rezultatul interogarii? Se precizeaza ca atributul cheie straina `IdVanzator` din relatia `VANZARII` refera atributul `IdAngajat` din relatia `ANGAJATI`.

3.9 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele, prenumele si data nasterii angajatilor care participa la proiectul cu denumirea `'Sistem de achizitie de date'` ?" Care va fi rezultatul interogarii?

3.10 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele sectiilor în care lucreaza angajatii care participa la proiectul cu denumirea 'Sistem de achizitie de date' ?" Care va fi rezultatul interogarii?

3.11 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele, prenumele si data nasterii tuturor inginerilor cu specialitatea 'Electronica'?" Care va fi rezultatul interogarii?

3.12 Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele angajatilor din sectia 'Productie' care au fii cu prenumele 'Ion' ?" Care va fi rezultatul interogarii?

3.13* Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele inginerilor din sectia 'Productie'?" Care va fi rezultatul interogarii?

3.14* Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele angajatilor cu care a colaborat furnizorul Popescu Razvan ? Care va fi rezultatul interogarii?

3.15* Scrieti si executati instructiunea SQL pentru interogarea "Care sunt numele si prenumele angajatilor cu care a colaborat clientul Marinescu Ion ? Care va fi rezultatul interogarii?

3.16* Scrieti si executati instructiunea SQL pentru interogarea "Care sunt produsele în care se folosesc componente furnizate de furnizorul Danescu Ovidiu ?" Care va fi rezultatul interogarii?

3.17* Scrieti si executati instructiunea SQL pentru interogarea "Care sunt produsele în care se folosesc componente furnizate de furnizorul Danescu Ovidiu ?" Care va fi rezultatul interogarii?