Baze de date relationale

* 1. Generalitati

<http://www.scritub.com/stiinta/informatica/baze-de-date/Baze-de-date-relationale83395.php>

Modelul relational al datelor a fost primit cu entuziasm si acceptat aproape fara rezerve atat de specialistii din domeniul bazelor de date cat si de utilizatori, inca de la aparitia primelor articole ale lui Codd E. F., in 1970 prin care erau puse bazele acestui model. Ideea unui model asamblist al datelor a fost lansata in 1968 de catre Childs D. F. care a subliniat faptul ca orice structura de date poate fi reprezentata printr-una sau mai multe tabele de date, in cadrul carora este necesar sa existe si informatii de legatura, pentru  asigurarea legaturilor intre tabele. Codd are meritul de a fi articulat si dezvoltat ideile cu privire la utilizarea teoriei apartenentei la ansambluri sub forma unui model coerent de structurare a datelor numit '**modelul relational**'. Despre acest model s-a scris mult, astfel incat in prezent, lucrarile consacrate modelului relational si sistemelor de gestiune a bazelor de date relationale sunt cu mult mai numeroase de cat cele dedicate altor tipuri de modele si sisteme.

S-a constatat ca prin utilizarea sistemelor relationale este posibila atingerea unor obiective importante ale organizarii datelor si anume: asigurarea unei mai mari independente a programelor de aplicatie fata de organizarea datelor precum si a unor instrumente eficace de control a coerentei si redundantei datelor.

Utilizatorii cuceriti de simplitatea modelului relational au manifestat o puternica preferinta pentru  sistemele relationale. Trebuie subliniat faptul ca  folosirea modelului relational nu necesita cunostinte teoretice deosebite . Modelul relational poate fi prezentat utilizatorilor cu ajutorul unor concepte extrem de simple si intuitive. De exemplu, ***relatia*** conceptul central in modelul relational poate fi prezentata sub forma unui ***tablou de date***. Reprezentarea ***relatiei*** in acest mod este comoda, usor de inteles si de utilizat, in special in cadrul operatiilor asupra datelor.

Definirea unui SGBD relational impune analizarea carcteristicilor pe care trebuie sa le prezinte un model de date pentru a fi considerat relational. Exista diferite modalitati pentru a defini acest concept:

·       Prezentarea datelor in tabele supuse anumitor operatii: selectie, proiectie, reuniune, compunere, intersectie, etc.

·       Un sistem de baze de date ce suporta un limbaj de tip SQL - Structured Query Language.

·       Un sistem de baze de date care respecta principiile modelului relational introdus de Codd.

1.2. Caracteristicile modelului relational.

Un model relational este caracterizat de trei concepte. Conceptele modelului relational sunt :

1.     ***Structura relationala a datelor***. În cadrul bazelor de date relationale, datele sunt organizate sub forma de tablouri bidimensionale (tabele de date), numite ***relatii***. Asocierile dintre relatii se reprezinta explicit prin ***atribute de legatura***. Aceste atribute figureaza intr-una din relatiile implicate in asociere (de regula , in cazul legaturilor de tip 'unu la multi') sau sunt plasate intr-o relatie distincta, construita special pentru  examinarea legaturilor intre relatii (in cazul legaturilor de tip 'multi la multi'). O baza de date relationala (BDR) reprezintă 333j92d ; un ansamblu de relatii, prin care se reprezinta atat datele cat si legaturile dintre date.

2.     ***Operatorii modelului relational***. Definesc operatiile care se pot efectua asupra relatiilor, in scopul realizarii functiilor de prelucrare asupra bazei de date, respectiv consultarea, inserarea, modificarea si stergerea datelor.

3.     ***Restrictiile de integritate ale modelului relational***.  Permit definirea starilor coerente ale bazei de date.

1.2.1. Structura relationala a datelor

  Prezentarea structurii relationale a datelor impune definirea notiunilor de: domeniu, relatie, atribut si schema a unei relatii.

1.2.1.1.Domeniu

Domeniul reprezinta o multime de valori, caracterizata printr-un nume. Un domeniu se poate defini explicit, prin enumerarea tuturor valorilor apartinand acestuia sau implicit, prin precizarea proprietatilor pe care le au valorile din cadrul domeniului respectiv.

Sa consideram, domeniile D1, D2, D3 definite astfel:

D1:

D2:

D3:

D1, D2 - definire explicita

D3       - definire implicita

Pentru un ansamblu de domenii D1, D2,…,Dn produsul cartezian al acestora reprezinta multimea tuplurilor <v1, v2,.., vn >, unde v1 este o valoare apartinand domeniului D1, v2 o valoare din D2, s.a.m.d.

De exemplu, tuplurile :<“Maria”, “F”, 50>, <”Vasile”, ”M”,15> va apartine produsului cartezian: D3http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image002.gifD1http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image002.gifD2.

1.2.1.2.Relatie

Relatia reprezinta o submultime a produsului cartezian al mai multor domenii, submultime caracterizata printr-un nume si care contine tupluri cu semnificatie.

Considerand, de exemplu ca se cunosc doua persoane, vom defini relatia R prin tuplurile care descriu aceste persoane si anume:

R:

Observatie: Într-o relatie tuplurile trebuie sa fie distincte, nu se admit duplicari ale tuplurilor.

O reprezentare a relatiei este tabelul bidimensional (tabela de date), in care liniile reprezinta tuplurile, iar coloanele corespund domeniilor.

    R:    D3              D1      D2

|  |  |  |
| --- | --- | --- |
| 'Maria' | 'F' | 30 |
| 'Vasile' | 'M' | 35 |

 Relatia poate avea semnificatia unui fisier, iar tuplul poate fi considerat drept o inregistrare, iar valorile din cadrul tuplului pot fi interpretate drept valori ale campurilor inregistrare.

Numarul tuplurilor dintr-o relatie reprezinta ***cardinalul relatiei*** iar numarul valorilor (atributelor) dintr-un tuplu defineste ***gradul relatiei***.

1.2.1.3.Atribut

În timp ce tuplurile dintr-o relatie trebuie sa fie unice, un domeniu poate apare de mai multe ori in produsul cartezian pe baza caruia este definita relatia.

Sa consideram, de exemplu ca o persoana dispune de urmatoarele date: nume, sex, varsta si numele sotului.

O posibilitate de organizare a acestor date o reprezinta relatia din urmatorea figura.

     PERS: D3               D1                   D2                   D3

|  |  |  |  |
| --- | --- | --- | --- |
| 'Maria' | 'F' | 30 | 'Vasile' |
| 'Vasile' | 'M' | 35 | 'Maria' |

Relatia PERS reprezinta o submultime a produsului cartezian: D3http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image002.gifD1http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image002.gifD2http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image002.gifD3.

Semnificatia valorilor din cadrul unui tuplu se stabileste in acest caz nu numai pe baza domeniului de care apartine valorile, ci li in functie de pozitia ocupata in cadrul tuplului. Dependenta fata de ordine a datelor inseamna o reducere a flexibilitati organizarii datelor. Într-o organizare eficienta, flexibila ordinea liniilor si coloanelor din cadrul tabelei de date nu trebuzie sa prezinte nici o importanta. Pentru a diferentia coloanele care contin valori ale aceluiasi domeniu si a elimina astfel dependenta de pozitie in cadrul tabelei se associaza fiecarei coloane un nume distinct, ceea ce duce la aparitia notiunii de atribut.

Atributul reprezinta coloana unei tabele de date, caracterizata printr-un nume. Numele coloanei (atributului) exprima de obicei semnificatia valorilor din cadrul coloanei respective. Relatia PERS poate fi prezentata astfel:

 PERS:

|  |  |  |  |
| --- | --- | --- | --- |
| Nume:D3 | Sex:D1 | Varsta:D2 | Sot:D3 |
| 'Maria' | 'F' | 30 | 'Vasile' |
| 'Vasile' | 'M' | 35 | 'Maria' |

Cand inseram tupluri intr-o relatie, de multe ori un atribut este necunoscut sau neaplicabil. Pentru a reprezenta acest atribut a fost introdusa o valoare conventionala in relatie, si anume valoarea *null*. Rezultatul operatorilor <, >, =,+,-,\*, / este *null* cand unul din argumente este *null*. Prin urmare, *'null = null'*are valoarea *null*.

1.2.1.4.Schema unei relatii

   Prin **schema** unei relatii se intelege numele relatiei, urmat de lista atributelor, pentru fiecare atribut precizandu-se domeniul asociat. Astfel, pentru o relatie R cu atributele A1, A2,…, An si domeniile D1, D2,…, Dm, cu m ≤ n, schema relatiei R poate fi reprezentata intr-una din formele prezentate mai jos:

R(A1 : D1,…, An : Dm)  sau

R:

|  |  |  |
| --- | --- | --- |
| A1:D1 | ….. | An:Dm |
|  |  |  |

Schema unei relatii se mai numeste si **intensia** unei relatii. Intensia unei relatii  este o expresie a proprietatilor comune si invariante ale tuplurilor care compun relatia.

**Extensia** unei relatii reprezinta multimea tuplurilor care compun la un moment dat relatia, multime care este variabilia in timp.

De obicei, extensia unei relatii este stocata fizic in spatiul asociat bazei de date relationale, caz in care relatia poarta numele de **relatie de baza**.

Exista insa si situatii in care extensia nu este memorata in baza de date. Este cazul asa numitelor **relatii virtuale**, cunoscute sub denumirea de vizualizare ( tabelul view).

***Tabelul view*** (relatia virtuala, vizualizare) constituie un filtru asupra tabelului initial care contine numai informatia necesara unei anumite abordari, unei anumite aplicatii.  Daca baza de date este memorata fizic pe disc, tabelul view este virtual  deoarece datele pe care le contine nu sunt in realitate memorate in baza de date, este memorata numai definitia tabelului view.

Vizualizarea nu este definita explicit, ca relatiile de baza, prin multimea tuplurilor componente ci implicit, pe baza altor relatii prin intermediul unor expresii relationale. Stabilirea efectiva a tuplurilor care compun vizualizarea se realizeaza prin evaluarea expresiei relationale ori de cate ori utilizatorul se refera la aceea vizualizare.

Utilizarea vizualizarilor este avantajoasa, deoarece asigura securitatea tabelului initial, care este protejat de stergeri, modificari etc. Exista totusi limitari in utilizarea acestor tabele, de exemplu coloanele calculate nu pot fi reactualizate.

 Pentru crearea vizualizarilor exista implementari specifice in diferite limbaje de definire a datelor. De exemplu, in SQL :

CREATE VIEW student(nume, prenume, an, grupa )

AS SELECT nume, prenume, cod\_an, cod\_grupa

FROM stud

WHERE cod\_sectie = 'SSC';

1.2.2. Operatorii. modelului relational

Operatorii modelului relational definesc operatiile care se pot efectua asupra relatiilor, in scopul realizarii functiilor de prelucrare asupra bazei de date.

Modelul relational ofera doua multimi de operatori pe relatii si anume: algebra relationala si calculul relational.                      Calculul relational este de doua tipuri: orientat pe tuplu (pe inregistrare) sau orientat pe domenii.

Operatorii algebrei relationale sunt fie operatori traditionali pe multimi (UNION, INTERSECT, PRODUCT, DIFFERENCE), fie operatori relationali speciali (PROJECT, SELECT, JOIN, DIVISION).

Calculul realtional reprezinta o adaptare a calcului predicatelor la domeniul bazelor de date relationale. Ideea de baza este de a identifica o relatie cu un predicat. Pe baza unor predicate initiale, prin aplicarea unor operatori ai calculului cu predicate (conjuctia, disjunctia, negatia, cuantificatorul existential si cel universal) se pot defini noi predicate, noi relatii. Variabilele care apar in constructiile calculului relational  orientat pe domenii sunt variabile definite asupra domeniilor, iar cele care apar in constructiile calculului relational orientat pe tupluri sunt variabile definite asupra relatiilor, adica valorile acestora reprezinta tupluri.

Exista doua moduri diferite care sunt utilizate pentru a exprima operatiile intr-un model relational:

·       ***Algebric***, in care filtrele (noile relatii) se obtin aplicand operatori specializati asupra uneia sau mai multor relatii din cadrul bazei relationale (algebra relationala).

·       ***Logic,*** in care filtrele se obtin cu ajutorul unor formule logice pe care tuplurile relatiei rezultat trebuie sa le satisfaca (calculul relational).

Echivalenta dintre algebra relationala si calculul relational a fost demonstrata de Ullman. Aceasta echivalenta arata ca orice relatie posibil de definit in algebra relationala poate fi definita si in cadrul calcului relational, si reciproc.

1.2.2.1. Algebra relationala

Algebra relationala a fost introdusa de Codd  ca o multime de operatii formale actionand asupra unor relatii si producand ca rezultat alte relatii. Baza teoretica fundamentala pentru limbajele de interogare relationale o constituie cei opt operatori introdusi de Codd pentru manipularea relatiilor.

 Operatorii realizeaza urmatoarele functii:

SELECT (selectie) - extrage tupluri ce satisfac o conditie specificata;

PROJECT (proiectie) - extrage atributele specificate;

DIFFERENCE (diferenta) - elimina tupluri care apar intr-o relatie, dar nu apar in cealalta;

PRODUCT (produs cartezian) - genereaza toate perechile posibile de tupluri, primul element al perechi fiind luat din prima relatie, iar cel de-al doilea element din cealalta relatie;

UNION (reuniune) - reuneste doua relatii;

INTERSECT (intersectie) - extrage tupluri care apar in ambele relatii;

DIVISION(diviziune) - extrage valorile atributelor dintr-o relatie, care apar in toate valorile atributelor din cealalta relatie;

JOIN(compunere) - extrage tupluri din mai multe relatii corelate astfel:

·       NATURAL JOIN (compunere naturala) - combina tupluri din doua relatii, cu conditia ca atributele comune sa aiba valori identice;

·       SEMI-JOIN(semi-compunere) - selecteaza toate tuplurile din prima relatie care vor fi corelate cu tupluri din cea de-a doua relatie;

·       http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image004.gif-JOIN (compunere) - combina tupluri din doua relatii, cu conditia ca valorile atributelor sa satisfaca o anumita conditie;

·       OUTER JOIN (compunere externa) - combina tupluri din doua relatii, astfel incat conditiile de corelare sa fie satisfacute. Tuplurile din orice relatie care nu satisfac aceste conditii sunt completate cu valori *null*.

Pentru operatorii UNION, INTERSECT si DIFFERENCE, se presupune ca sunt aplicati numai la relatii avand aceeasi aritate, iar ordinea atributelor este aceeasi.

Toate operatiile de baza ale algebrei relationale se regasesc sub forma instructiunilor de manipulare a datelor din SQL. Exemplele au fost introduse cu scopul de-a marca analogia intre transpunerea cererilor in limbaj algebric si implementarea in SQL. Toate exemplele care vor fi intalnite se refera la urmatoarele  tabele de date:

SALARIAT(cod\_salariat, nume, prenume, sex, job, cod\_sef, data\_ang, salar, comision, cod\_dept),

DEPARTAMENT (cod\_dept, nume\_dept, loc),

ATASAT\_LA(cod\_salariat, nr\_proiect, functia),

PROIECT(nr\_proiect, descriere, buget).

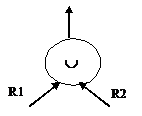
***1.*** ***Operatorul UNION (reuniunea)***

Reuniunea reprezinta o operatie a algebrei relationale definita pe doua relatii: R1 si R2 ambele cu o aceeasi schema, operatie care consta din construirea unei noi relatii R3, cu schema identica cu R1 si R2 si avand drept extensie tuplurile din  R1 si R2, luate impreuna o singura data.

Operatorul UNION permite obtinerea tuplurilor distincte a doua relatii si poate fi utilizat pentru adaugarea de noi tupluri intr-o relatie.

   Notatiile uzuale  intalnite pentru reuniune sunt: R1ÈR2, OR(R1ÈR2), APPEND(R1,R2), UNION(R1,R2).

Reprezentarea grafica a operatiei de reuniune a doua relatii este:

Fig. 4.2‑1 Reprezentarea grafica a operatorului UNION

Ø     Afisarea tuturor persoanelor care sunt programator sau sunt de sex masculin din baza de date **salariat**se va realiza prin operatia de reuniune.

Reuniunea in SQL :

SELECT nume, prenume, salar, comision

FROM salariat

WHERE job = 'programator'

  UNION

SELECT nume, prenume, salar, comision

FROM salariat

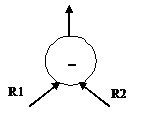
WHERE sex = 'm';

***2***. ***Operatorul DIFFERENCE (diferenta)***

Diferenta reprezinta  o operatie a algebrei relationale definita pe doua relatii R1 si R2 ambele cu o aceeasi schema, operatia consta din construirea unei noi relatii R3, cu schema identica cu a relatiilor R1 si R2, cu extensia formata din acele tupluri ale relatiei R1 care nu se regasesc si in relatia R2.

Diferenta este o operatie binara necomutativa care permite obtinerea tuplurilor ce apar numai intr-o relatie.   Notatiile uzuale pentru operatia de diferenta sunt: R1 - R2, REMOVE(R1, R2), MINUS(R1, R2), DIFFERENCE(R1, R2).

Reprezentarea grafica a operatiei de diferenta este:

Fig. 4.2‑2 Reprezentarea grafica a operatorului DIFFERENCE

***3.*** ***Operatorul INTERSECT (intersectia)***

Intersectia reprezinta o operatie a algebrei relationale definita pe doua relatii: R1 si R2, ambele cu aceeasi schema, operatie care consta din construirea unei noi relatii R3, cu schema identica cu a relatiilor R1 si R2 si cu extensia formata din tuplurile comune ale relatiilor R1 si R2.

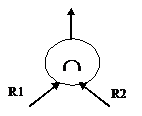
Operatorul INTERSECT este un operator binar, comutativ, derivat:

 R1 Ç R2 = R1 - (R1 - R2) sau

R1 Ç R2 = R2 - (R2 - R1)

Notatiile uzuale pentru operatia de intersectie sunt: R1 Ç R2, INTERSECT(R1, R2), AND(R1,R2)

Reprezentarea grafica a operatiei de intersectie este:

Fig. 4.2‑3 Reprezentarea grafica a operatorului INTERSECT

Operatorii DIFFERENCE si INTERSECT pot fi simulati in SQL cu ajutorul optiunilor EXISTS, NOT EXISTS, IN, != ANY, in cadrul comenzii SELECT. În SQL\*PLUS exista operatori speciali INTERSECT, MINUS care realizeaza aceste operatii.

Ø     Afisarea tuturor angajatilor (nume, prenume, salar, comision, sex) din baza de date **salariat** care lucreaza la proiectul cu numarul 22, se va realiza prin operatorul  INTERSECT.

Intersectie in SQL :

SELECT nume, prenume, salar, comision, sex

FROM salariat s

WHERE s. cod\_salariat EXISTS

(SELECT a.cod\_salariat

FROM atasat\_la a

WHERE a.nr\_proiect = 22

AND s.cod\_salariat = a.cod\_salariat);

Ø     Afisarea tuturor angajatilor (nume, prenume, salar, comision, sex) din baza de date **salariat** care nu lucreaza la proiectul cu numarul 22, se va realiza prin operatorul  DIFFERENCE.

Diferenta in SQL :

SELECT nume, prenume, salar

FROM salariat s

WHERE s.cod\_salariat NOT EXISTS

(SELECT a.cod\_salariat

FROM atasat\_la a

WHERE a.nr\_proiect = 22

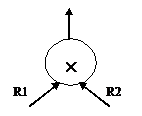
AND s.cod\_salariat = a.cod\_salariat);

***4.*** ***Operatorul PRODUCT (produs cartezian)***

Produsul cartezian reprezinta o operatie a algebrei relationale definita pe doua relatii: R1 si R2, operatie care consta din construirea unei noi relatii R3, a carei schema se obtine prin concatenarea schemelor relatiilor R1 si R2 si a carei extensie cuprinde toate combinatiile tuplurilor din R1 cu cele din R2.

Notatiile uzuale pentru aceasta operatie sunt: R1 ´ R2, PRODUCT(R1, R2), TIMES(R1, R2).

Reprezentarea grafica a produsului cartezian este:

Fig. 4.2‑4 Reprezentarea grafica a operatorului PRODUCT

Produsul cartezian al bazei de date **salariat**cu baza de date **departament** poate fi exprimat in SQL astfel:

SELECT \*

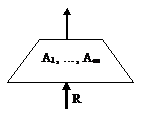
FROM salariat, departament;

***5.*** ***Operatorul PROJECT (proiectia)***

Proiectia reprezinta o operatie din algebra relationala definita asupra unei relatii R, operatie care consta din construirea unei noi relatii P, in care se regasesc numai acele atribute din R specificate explicit in cadrul operatiei. Suprimarea unor atribute din R inseamna efectuarea unor taieturi verticale asupra lui R, care pot avea ca efect aparitia unor tupluri duplicate ce se cer a fi eliminate.

Prin operatia de proiectie se trece de la o relatie de gradul n la o operatie de gradul p, mai mic decat cel initial (p < n).

Notatiile uzuale pentru operatia de proiectie sunt urmatoarele:http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image014.gif, R[ A1, A2, …, Am], PROJECT(R, A1, A2, …, Am), unde A1, A2, …, Am sunt parametrii proiectiei relativ la relatia R. Reprezentarea grafica a proiectiei este:

Fig. 4.2‑5 Reprezentarea grafica a operatorului  PROJECT

Proiectia poate fi exprimata in SQL astfel:

(Proiectie cu inregistrari nedistincte)

SELECT nume, prenume, salar, comision

FROM salariat;

(Proiectie cu inregistrari distincte, fara dubluri)

SELECT  DISTINCT nume, prenume, salar, comision

FROM salariat;

***6.*** ***Operatorul SELECT (selectia)***

Selectia reprezinta o operatie din algebra relationala definita asupra unei relatii R, operatie care consta din construirea unei relatii S, a carei schema este identica cu cea a relatiei R si a carei extensie este construita din acele tupluri din R care satisfac o conditie mentionata explicit in cadrul operatiei.

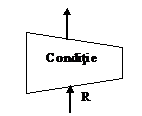
Conditia precizata in cadrul operatiei de selectie este in general de forma:

atribut < operator de comparatie > valoare

unde: < operator de comparatie>  poate fi: <, <=, >=, > sau ¹

Notatiile utilizate sunt: sconditie(R), R[conditie], RESTRICT(R, conditie), SELECT(R, conditie).

Reprezentarea grafica a operatiei de selectie este:

Fig. 4.2‑6 Reprezentarea grafica a operatorului SELECT

Ø    Afisarea angajatilor din departamentul cu numarul10 se va realiza utilizand operatia de selectie pe baza de date **salariat**.

Selectia poate fi exprimata in SQL astfel:

SELECT \*

FROM salariat

WHERE cod\_dept =10;

***7.*** ***Operatorul JOIN (Jonctiune, Compunere)***

Operatorul JOIN permite regasirea informatiei din mai multe relatii corelate.

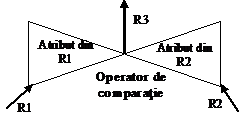
Compunerea (jonctiunea) reprezinta o operatie din algebra relationala definita pe doua relatii: R1 si R2, operatie care consta din construirea unei noi relatii R3, prin concatenarea unor tupluri din R1 cu tupluri din R2. Se concateneaza acele tupluri din R1 si R2 care satisfac o anumita conditie, specificata explicit in cadrul operatiei. Extensia relatiei R3 va contine acele tupluri care satisfac conditia de compunere. Conditia necesara aplicarii operatorului JOIN este ca tuplurile ce se combina sa fie similare.

Notatiile utilizate sunt: JOIN( R1, R2, conditie).

Operatorul JOIN combina produsul cartezian, selectia si proiectia, astfel se construieste un produs cartezian, se elimina tupluri prin selectie, si se elimina atribute prin proiectie, adica:

JOIN(R1,R2,conditie)=RESTRICT(PRODUCT(R1,R2), conditie)

Reprezentarea grafica a operatorului JOIN este:

Fig. 4.2‑7 Reprezentarea grafica a operatorului JOIN

În functie de operatorul de comparatie din cadrul conditiei de compunere, exista mai multe variante ale operatorului JOIN pe care le vom prezenta in continuare.

**Operatorul NATURAL JOIN (equijoinul, Jonctiunea naturala)**

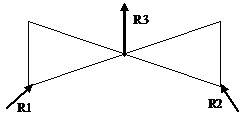
Jonctiunea naturala este o operatie definita pe doua relatii R1 si R2, prin care este constituita o noua relatie R3, a carei schema este obtinuta prin reuniunea atributelor din relatiile R1 si R2 (atributele cu acelasi nume se iau o singura data), si a carei extensie contine tuplurile obtinute prin concatenarea tuplurilor din R1 cu tuplurile din R2 care reprezinta aceleasi valori pentru atributele cu acelasi nume.

Cel mai important tip de JOIN, in sensul celei mai frecvente utilizari este **equijoinul**. Equijoinul reprezinta jonctiunea dirijata de o conditie de forma:

          atribut din R1 = atribut din R2

Notatia uzuala pentru jonctiunea naturala (equijoin) este: JOIN(R1, R2), R1>< R2.

Reprezentarea grafica a jonctiunii naturale este:

Fig. 4.2‑8 Reprezentarea grafica a operatorului NATURAL JOIN

Algoritmul care realizeaza jonctiunea naturala este urmatorul:

1.       Se calculeaza produsul cartezian R1 ´ R2;

2.       Pentru fiecare atribut comun A care defineste o coloana in R1 si o coloana in R2, se selecteaza din R1 ´ R2 tuplurile ale caror valori coincid in coloanele R1.A si R2.A. Atributul R1.A reprezinta numele coloanei din R1 ´ R2 corespunzatoare coloanei A din R1;

3.     Pentru fiecare astfel de atribut A se proiecteaza coloana R2.A, iar coloana R1.A se va numi A.

Astfel, R1>< R2 =http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image024.gif, unde A1, …, Am sunt atribute comune lui R1 si R2, iar i1, …imreprezinta lista componentelor din R1 ´ R2 (pastrand ordinea initiala) din care au fost eliminate componentele R2.A1, …, R2.Ak.

Ø     Afisam numele angajatilor si departamentele la care acestia lucreaza, utilizind operatorul equijoin.

Equijoinul poate fi exprimat in SQL astfel:

SELECT nume, prenume, job, nume\_dept, loc

FROM salariat s, departament d

WHERE s.cod\_dept =d.cod\_dept;

            În acest exemplu, atributul 'd.cod\_dept' este cheie principala in tabelul 'departament', iar 's.cod\_dept' este cheie externa in tabelul  salariat.

**Operatorul Q-JOIN**

            Operatorul Q-JOIN combina tupluri din doua relatii, cu conditia ca valorile atributelor specificate sa satisfaca o anumita conditie. Conditia este specificata explicit in cadrul operatiei. Operatorul se reprezinta cu ajutorul simbolului >< sub care se scrie conditia, sau prin JOIN(R1, R2, conditie).

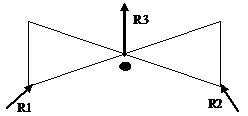
            Operatorul Q-JOIN este un operator derivat, fiind o combinatie de produs scalar si selectie: JOIN(R1, R2, conditie) = http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image026.gif

**Operatorul OUTER JOIN (Jonctiunea externa)**

Atunci cand relatiile care participa la jonctiune nu au proiectii identice pe atributul de jonctiune (atributul de jonctiune nu prezinta aceleasi valori in relatiile care se jonctioneza), operatia de jonctiune conduce la pierderea de tupluri, cel putin dintr-o relatie. În scopul evitarii pierderilor de informatie a fost definita jonctiunea externa, ca operatie prin care din doua relatii: R1 si R2 se obtine o noua relatie, R3 prin jonctionarea relatiilor R1 si R2 relatie la care sunt adaugate si tuplurile din R1 si R2 care nu au participat la join. Aceste tupluri sunt completate in relatia R3 cu valori “null” pentru atributele relatiei corespondente (R2 respectiv R1).

Notatiile uzuale pentru jonctiunea externa sunt: http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image028.jpg, EXT-JOIN(R1,R2) si OUTERJOIN(R1, R2).

Reprezentarea grafica a operatiei de outer join este:

Fig. 4.2‑9 Reprezentarea grafica a operatorului OUTER JOIN

**Operatorul SEMI-JOIN (Semijonctiunea)**

Semijonctiunea reprezinta o operatie definita pe doua relatii: R1 si R2, operatie care consta din construirea unei noi relatii R3, a carei extensie contine tuplurile relatiei R1 care participa la jonctiunea celor doua relatii. Semijonctiunea produce acelasi rezultat cu cel al operatiei de proiectie pe atributele din relatia R1 efectuata asupra jonctiunii relatiilor R1 si R2 sau cu cel al operatiei de selectie asupra relatiei R1 realizata pe baza valorilor din R2 ale atributului de jonctiune.

Notatiile utilizate pentru operatia de semijonctiune sunt:

http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image032.jpg  , SEMIJOIN(R1,R2, conditie)

SEMI-JOIN este un operator derivat, fiind o combinatie de proiectie si natural join sau proiectie si Q-join: SEMIJOIN(R1, R2)= http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image034.gif(R1>< R2) sau SEMIJON(R1, R2, conditie)= http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image034.gif (R1>< R2, conditie), unde am notat prin M atributele relatiei R.

***8. Operatorul DIVISION (Diviziunea)***

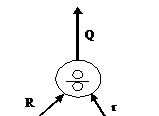
Reprezinta o operatie din algebra relationala definita asupra unei relatii R cu schema: R(A1:D1, …, Ap:Dk, Ap+1:Dl, …,An:Dm), operatie care consta din construirea cu ajutorul unei relatii   r(Ap+1:Dl,…,An:Dm) a relatiei Q(A1:D1,…,Ap:Dk). Tuplurile relatiei Q, concatenate cu tuplurile relatiei r permit obtinerea tuplurilor relatiei R.

Operatia de diviziune reprezinta o operatie derivata, care se poate exprima prin intermediul diferentei, produsului carterzian si proiectiei astfel:

http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image036.gifhttp://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image038.gif

Notatiile utilizate sunt: R ÷ r, DIVISION(R, r).

Reprezentarea grafica  a operatiei de diviziune este:

Fig. 4.2‑10 Reprezentarea grafica a operatorului DIVISION

Operatia de diviziune reprezinta o operatie derivata, care se poate exprima prin intermediul diferentei, produsului carterzian si proiectiei astfel:

http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image036.gifhttp://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image038.gif

Operatorul DIVISION este legat de cuantificatorul universal (') care nu exista in SQL. Cuantificatorul universal poate fi insa simulat cu ajutorul cuantificatorului existential ($) care se gaseste in limbajul SQL, utilizand relatia:

'x P(x) ºØ $ x Ø P(x).

Operatorul DIVISION poate fi exprimat in SQL prin succesiunea a doi operatori NOT EXISTS.

Ø     Diviziunea in SQL prin care se obtin codurile salariatilor atasati tuturor proiectelor pentru care s-a alocat un buget de 100.000 va fi exprimata astfel:

SELECT UNIQUE(cod\_salariat)

FROM atasat\_la a

WHERE a.nr\_proiect NOT EXISTS

            (SELECT p.nr\_proiect

              FROM proiect p

              WHERE  buget = 100.000

              AND p.nr\_proiect NOT EXISTS

                        (SELECT b.nr\_proiect

                          FROM atasat\_la b

                          WHERE p.nr\_proiect = b. nr\_proiect

                           AND b.cod\_salariat = a.cod\_salariat));

Exemplul arata dificultatea exprimarii operatiei de diviziune in SQL. Diviziunea poate fi exprimata si cu ajutorul functiei COUNT:

SELECT cod\_salariat

FROM atasat\_la a

WHERE nr\_proiect IN

            (SELECT nr\_proiect

            FROM proiect

            WHERE buget =100.000)

GROUP BY cod\_salariat

HAVING COUNT(nr\_proiect) = (SELECT COUNT (\*)

                                                  FROM proiect

                                                  WHERE buget =100.000);

Pentru manipularea optima a relatiilor au fost introduse si operatii aditionale.

***9. Complementarea***

Reprezinta o operatie aditionala din algebra relationala care permite determinarea complementului unei relatii. Complementul unei relatii este definit drept ansamblul tuplurilor produsului cartezian al domeniilor asociate atributelor relatiei, care nu figureaza in extensia relatiei considerate. Este obligatoriu ca domeniile sa fie finite. Cardinalitatea rezultatului poate fi extrem de mare, ceea ce face ca operatia de complementare sa fie relativ putin utilizata.

   Notatiile pentru operatia de complementare sunt: ¬ R,          NOT(R), COMP(R)

Consideram relatia R pentru care domeniile DA1: si DA2 :

R:

|  |  |
| --- | --- |
| A1:DA1 | A2:DA2 |
| 'Rosu' 'Rosu'  'Galben' | 1  2  3 |

a)     Relatia R

Complementul reatiei R este:

Not(R):

|  |  |
| --- | --- |
| A1:DA1 | A2:DA2 |
| 'Rosu' '  'Galben'  'Galben'  'Albastru'  'Albastru'  'Albastru' | 3  1  2  1  2  3 |

b) Complementul relatiei R

***10. Splitarea (Spargerea)***

Reprezinta o operatie aditionala din algebra relationala definita asupra unei relatii R, operatie care pe baza unei conditii definite asupra atributelor din R permite construirea a doua relatii: R1 si R2, cu o aceeasi schema cu R. Extensia lui R1 contine tuplurile din R care verifica  conditia specificata, iar R2 contine tuplurile din R care nu verifica aceasta conditie.

Considerand, relatia R definita mai sus si conditia de splitare  A2 >2, operatia de splitare a relatiei R produce ca rezultat relatiile R1 si R2 prezentate mai jos:

R1:

|  |  |
| --- | --- |
| A1:DA1 | A2:DA2 |
| 'Galben' | 3 |

a)     Relatia R1

R2:

|  |  |
| --- | --- |
| A1:DA1 | A2:DA2 |
| 'Rosu' 'Rosu' | 1  2 |

b)     Relatia R2

***11. Închiderea tranzitiva***

Reprezinta o operatie aditionala din algebra relationala prin care se pot adauga tupluri la o relatie. Operatia de inchidere tranzitiva presupune executarea in mod repetat a secventei de operatii: jonctiune, proiectie, reuniune. Numarul de executii depinde de continutul relatiei, nefiind fixat dinainte.

Închiderea tranzitiva este o operatie definita asupra unei relatii R, a carei schema contine doua atribute: A1, A2 cu acelasi domeniu asociat, operatie care consta din adaugarea la relatia R a tuplurilor care se obtin succesiv prin tranzitivitate, in sensul ca, daca exista in R tuplurile <a, b> si <b, c>  se va adauga la R si tuplu <a, c>.

Notatiile uzuale pentru operatia de inchidere tranzitiva sunt: t(R), R+, CLOSE(R).

1.2.3. Restrictiile de integritate ale modelului relational.

Restrictiile de integritate, denumite si reguli de integritate definesc cerintele pe care trebuie sa le satisfaca datele din cadrul bazei de date pentru a putea fi considerate corecte, coerente in raport cu lumea reala pe care o reflecta.

În teoria sistemelor relationale, restrictiile de integritate sunt studiate in principal sub aspectul puterii lor de modelare si al posibilitatilor de verificare eficace a respectarii lor.

Restrictiile de integritate ale modelului relational sunt de doua tipuri si anume :

**1. Restrictii de integritate structurale,** care sunt inerente modelului relational, care se definesc prin egalitatea sau inegalitatea unor valori din cadrul relatiilor. Din categoria restrictiilor de integritate structurala fac parte :

-   restrictia de unicitate a cheii

-   restrictia referentiala

-   restrictia entitatii

-   dependetele intre date

**2. Restrictii de integritate de comportament,** proprii unei BDR, restrictii care tin cont  de semnificatia valorilor din cadrul bazei de date. Putem considera, de exemplu restrictia incadrarii virstei (sau inaltimi, greutati etc.) unei persoane intre anumite limite acceptabile.

Utilizarea modelului relational nu impune definirea si verificarea tuturor acestor tipuri de restrictii de integritate, structurale si comportament. Din acest punct de vedere putem considera ca exista:

1. **Restrictii de integritate minimale,** obligatoriu de definit si de respectat atunci cind se lucreaza cu modelul relational. Din aceasta categorie fac parte :

-        restrictia de unicitate a cheii

-        restrictia referentiala

-        restrictia entitatii

2. **Alte restrictii de integritate**, categorie din care fac parte :

    -   dependetele intre date

    -   restrictiile de comportament

1.2.3.1. Restrictiile de integritate minimale.

Restrictiile de integritate minimale sunt definite in raport cu notiunea de cheie a unei relatii.

Relatia se prezinta ca o multime de tupluri. Întrucat o multime nu poate contine elemente  duplicate, relatia nu poate prezenta tupluri identice. Identificarea unui tuplu in cadrul unei relatii fara a se face apel la toate valorile din tuplu a dus la aparitia notiunii de cheie.

**Cheia** unei relatii R reprezinta ansamblul minimal de atribute prin care se poate identifica in mod unic orice tuplu din R. Orice relatie poseda cel putin o cheie. Cheia este construita fie dintr-un singur atribut, fie din totalitatea atributelor din schema relatiei respective. Cand cheia este construita dintr-un singur atribut poarta numele de ***cheie simpla***, iar cand este formata din mai multe atribute este denumita de ***cheie compusa***.

Într-o relatie pot exista mai multe combinatii de atribute cu proprietatea de identificare unica a tuplurilor. În acest caz se spune ca relatia poseda mai multi ***candidati cheie*** sau mai multe ***chei candidate***.

În aceasta situatie, administratorul bazei de date va alege una dintre cheile candidate care sa serveasca in mod efectiv la identificarea tuplurilor si care se va numi ***cheie primara***. Restul cheilor candidate vor purta numele de ***chei alternate***.

Cheia unei relatii trebuie sa fie minimala, in sensul ca nici o parte a sa nu trebuie sa posede proprietatea de identificare unica a tuplurilor relatiei. Un grup de atribute din din cadrul unei relatii care contine o cheie a relatiei poarta numele de ***supercheie***.

Modelul relational serveste la reprezentarea entitatilor din lumea reala si a asocierilor dintre acestea. Modelarea asocierilor dintre entitati impune recurgerea la conceptul de **cheie externa**.

**O cheie externa** reprezinta un atribut sau un grup de atribute dintr-o relatie R1 ale carui sau caror valori sunt definite pe acelasi/aceleasi domeniu/domenii ca si cheia primara a unei relatii R2 si care are rolul de a modela asocierea intre entitatile reprezentate prin relatiile Rı si R2. În acest context R1 este numita 'relatie care refera' iar R2 este numita 'relatie referita'.

Atributele care reprezinta cheia primara sunt fie subliniate, fie urmate de semnul #.

De exemplu, presupunem 2 relatii DEPART si ANGAJATI care sunt prezentate in figura Fig. 4.2-11.

ANGAJATI(Codang, Nume, Codded) sau

ANGAJATI(Codang#, Nume, Codded)

DEPART(Coddep, Dendept, Tel)

            \_\_\_\_\_\_\_

ANGAJATI:                            |           |    DEPART:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Codang | Nume | Coddep |  | Coddep | Dendept | Tel |
| 111  222  333 | Pop Ion  Alb Ion  Sas Paul | 1  2  1 |  | 1  2  3 | Financiar  Personal  Tehnic | 112  113134 |

            Fig. 4.2‑11 Reprezentarea legaturii dintre relatiile ANGAJATI si DEPART

Într-un departament lucreaza mai multi angajati in timp ce un angajat lucreza intr-un departament.

Atributul 'Coddep' din relatia ANGAJATI reprezinta o cheie externa servind la asocierea dintre relatiile ANGAJATI si DEPART.

Modelul reltional prezinta urmatoarele restrictii de integritate minimale :

**1.     Restrictia de unicitate  a cheii.**

Reprezinta restrictia de integritate care impune ca intr-o relatie, R care are cheia k, oricare ar fi tuplurile t1 si t2 sa fie satisfacuta inegalitatea :  tı (k) ≠  t2 (k).

Aceasta inegalitate semnifica faptul ca intr-o relatie nu pot exista tupluri cu o aceeasi valoare pentru atributele cheie.

**2.     Restrictia referentiala.**

Reprezinta restrictia de integritate care impune ca intr-o relatie Rı care refera o relatie R2, valorile cheii externe sa figureze printre valorile cheii primare sau sa fie valori ' null ' nedefinite. R1 si R2 nu trebuie sa fie neaparat distincte.

Relatiile ANGAJATI si DEPART respecta restrictia de integritate.

3.     **Restrictia entitatii.**

Restrictia entitatii reprezinta restrictia de integritate care impune ca intr-o relatie, atributele cheii primare sa fie nenule. Unicitatea cheii impune ca la incarcarea unui tuplu, valoarea cheii sa fie cuoscuta, pentru a se putea verifica faptul ca aceasta valoare nu exista deja incarcata. Cu valori ' null ' cheia isi pierde rolul de identificator de tuplu.

Restrictia de integritate a entitatii nu se aplica cheilor externe dintr-o relatie, daca acestea nu apartin cheii primare.

1.2.3.2.Alte restrictii de integritate.

Din categoria restrictiilor de integritate neminimale fac parte:

-        restrictii referitoare la dependenta datelor.

-        restrictii de comportament

**Restrictiile referitoare la dependenta datelor**

Semnifca modul in care datele depin unele de altele. Aceasta depedenta intre date poate fi de mai multe tipuri si anume :

-        dependenta functionala

-        dependenta multivaloare

-        dependenta jonctiune

***Dependentele functionale.*** Reprezinta dependenta intre date prin care se poate identifica un atribut sau  un grup de atribute prin intermediul unui atribut sau grup de atribute.

Fiind data o relatie R, un atribut Y este dependent functional de un alt atribut X din R, daca si numai daca fiecare valoare a lui X are asociata o valoare precisa a lui Y.

http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image041.gifhttp://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image042.gifNotatia utilizata pentru desemnarea dependentei functionale este X           Y. Partea stanga a dependentei functionale poarta numele de **determinant**, iar partea dreapta a dependentei poarta numele de **determinat**. Astfel in cadrul dependentei functionale X    Y, X este deterninantul iar Y este determinatul.

http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image043.gifDependenta functionala : X     Y reprezinta o restrictie de integritate aplicata tuplurilor relatiei R, in sensul ca oricare doua tupluri din R care prezinta o aceeasi valoare pentru X trebuie sa prezinte o aceeasi valoare si pentru Y

***Dependentele multivaloare***. Reprezinta acel tip de dependenta intre date in care un atribut sau un grup de atribute poate prezenta mai multe valori pentru o singura valoare a unui atribut sau a unui grup de atribute.

Fie o relatie R, in care apar atributele sau grupurile de atribute: X, Y si Z. În cadrul relatiei R exista o dependenta multivaloare daca si numai daca multimea valorilor lui Y corespunzatoare unei perechi: (valoare X, valoare Z) depinde numai de valoare lui X, nu si de valoarea lui Z. aceasta dependenta multivaloare se va nota cu:

X ->> Y.

Pentru a avea X ->> Y ese necesar sa avem si : X ->> Z. Din acest motiv se obisnuieste ca dependenta multivaloare sa se noteze cu:

X ->> Y/Z

Exista o dependenta multivaloare : X ->> Y daca si numai daca YXZ depinde numai de valorile lui X, adica :

YXZ1= YXZ2 ,    Z1http://www.scritub.com/files/informatica/baze%20de%20date/85_poze/image045.gifZ2

**Dependentele jonctiune.**

Se considera o relatie R, cu schema R(X : Dx , Y : Dy , Z : Dz) pentru care nu se manifesta dependente functionale sau dependente multivaloare, adica relatia R se poate asimila unei chei compuse. Asupra acestei relatii se formuleaza urmatoarea restrictie : daca in relatia R figureaza tuplurile <X1, Y1, Z2>, <X2, Y1, Z1> si <X1, Y2, Z1> atunci in R trebuie sa figureze si tuplul <X1, Y1, Z1>. Aceasta restrictie exprima o dependenta intre date similara dependentei functionale sau dependentei multivaloare. Descompunerea fara pierderi de date a relatiei R se poate realiza sub forma urmatoarelor proiectii:

P1 (X:Dx , Y:Dy ), P2 (Y:Dy , Z:Dz ) si P3 (Z:Dz , X:Dx )

Reconstituirea relatiei R se realizeaza prin jonctionarea proiectiilor: P1, P2 si P3. Din aceasta cauza , dependenta din cadrul relatiei R a primit numele de **dependenta jonctiune**. Dependenta jonctiune exprima o restrictie mai generala decat cea exprimata de dependenta multivaloare sau dependenta functionala.

**Restrictiile de comportament**

În functie de realitatea descrisa in baza de date, se pot defini de catre utilizatori mai multe tipuri de restrictii de integritate de comportament si anume :

-        restrictii de domeniu

-        restrictii temporale etc.

Conform restrictiei de domeniu  valorile unui atribut al unei relatii trebuie sa se incadreze intre anumite valori (de exemplu atributul 'salariu' sa se incadreze intre anumite valori).

O restrictie temporala se poate referi la valorile atributului 'Stoc' din cadrul unei relatii MATERIALE, prin care sa se impuna ca valorile rezultate in urma actualizarii sa depaseasca valorile anterioare actualizarii.

SGBD-urile relationale sunt dotate cu mecanisme pentru controlul integritatii datelor, prin care se verifica respectarea restrictiilor de integritate de catre continutul bazei de date.

1.3. Definirea sistemelor de gestiune a bazelor de date relationale.

Se poate considera un sistem de gestiune a bazelor de date relationale (SGBDR) ca reprezentand un SGBD care utilizeaza drept conceptie de organizare a datelor modelul relational.

Dintre principalele caracteristici ale modelului relational amintim: nu exista tupluri identice, ordinea liniilor si a coloanelor este arbitrara, articolele unui domeniu sunt omogene, fiecare coloana defineste un domeniu distinct si nu se poate repeta in cadrul aceleiasi relatii, toate valorile unui domeniu corespunzatoare tuturor cazurilor nu mai pot fi descompuse in alte valori (sunt atomice).

Codd a publicat un set de 13 reguli, in raport cu care un SGBD poate fi apreciat ca relational. Ulterior, cele 13 reguli ale lui Codd au fost extinse la un numar de 100. Trebuie mentionat faptul ca nu exista un SGBD care respecta toate regulile de Codd, deci nu trebuie sa apreciem un SGBD ca fiind relational sau nu, ci masura in care acesta este relational, deci numarul regulilor pe care le respecta.

1.3.1. Regulile lui Codd.

Definirea unui SGBDR impune o detaliere a caracteristicilor pe care trebuie sa le prezinte un SGBD pentru a putea fi considerat relational. În acest sens, Codd a formulat 13 reguli care exprima cerintele pe care trebuie sa le satisfaca un SGBDR.

**R1 :  Regula privind gestionarea datelor la nivel de relatie**

Sistemul trebuie sa gestioneze baza de date numai prin mecanisme relationale, adica sa utilizeze limbaje precum SQL care sa opereze la un moment dat pe o intreaga relatie.

**R2 :  Regula privind reprezentarea logica a datelor**

Toate datele din baza de date relationala trebuie sa fie reprezentate explicit la nivel logic intr-un singur mod si anume ca valori in tabele de date.

**R3 :  Regula privind garantarea accesului la datelor**

Orice data din baza de date relationala trebuie sa poata fi accesata prin specificarea numelui de tabela, valorii cheii primare si numelui de coloana.

**R4 :  Regula privind valorile null**

Sistemele trebuie sa permita declararea si manipularea sistematica a valorilor null, cu semnificatia unor date lipsa sau in aplicatie. Într-un SGBD relational trebuie sa putem face diferenta intre valoarea zero, un sir vid de caractere si o valoare necunoscuta.

**R5 :  Regula privind metadatele (dictionarele de date)**

Descrierea bazei de date trebuie sa se prezinte la nivel logic in aceleasi mod cu desrierea datelor propriu zise, astfel incit utilizatorii autorizati sa poata aplica asupra descrierii bazei de date (informatii relative la relatii, vizualizari, indecsi etc) aceleasi operatii ca si asupra datelor obisnuite.

**R6 :  Regula limbajului de interogare**

Un sistem relational trebuie sa faca posibila utilizarea mai multor limbaje in mai multe moduri. Trebuie sa existe insa cel putin un limaj de nivel inalt ale carui instructiuni sa poata exprima oricare din urmatoarele operatii : definirea relatiilor, definirea tabelelor virtuale, manipularea datelor, definirea restrictiilor de integritate, autorizarea accesului, precizarea limitelor tranzactiilor.

**R7 :  Regula privind actualizarea tabelelor virtuale**

Toate tabelele virtuale care teoretic sunt posibil de actualizat trebuie sa poata fi efectiv actualizate. Trebuie sa existe un mecanism prin care sa se poata determina daca anumite vizualizari pot fi actualizate sau nu. Majoritatea implementarilor SQL stabilesc aceasta, in functie de variantele instructiunii de selectie utilizate.

**R8 :  Regula privind inserarile, modificarile si stergerile in baza de date**

Sistemul trebuie sa ofere posibilitatea manipularii unei tabele nu numai in cadrul operatiilor de regasire, ci si in actiunile de inserare, modificare si stergere a datelor.

**R9 :  Regula privind independenta fizica a datelor**

Programele de aplicatie nu trebuie sa fie afectate de schimbarile efectuate in modul de reprezentare a datelor sau in metodele de acces. O schimbare a structuri fizice a datelor nu trebuie sa blocheze functionarea programelor de aplicatie.

**R10 :  Regula privind independenta logica a datelor**

Programele de aplicatie nu trebuie sa fie afectate de schimbarile efectuate asupra relatiilor bazei de date, schimbari care conserva datele si teoretic garanteaza valabilitatea programelor de aplicatie existente.

**R11 :  Regula privind restrictiile de integritate**

Regulile de integritate trebuie sa fie definite intr-un sublimbaj relational, nu in cadrul programelor de aplicatie.

SQL permite definirea de restrictii privind integritatea datelor si stocarea lor in catalogul de sistem (de exemplu, atributele NOT NULL, si UNIQUE)

**R12 :  Regula privind distribuirea geografica a datelor**

Limbajul de manipulare a datelor utilizat de sistem trebuie sa permita ca, in situatia in care datele sunt distribuite, programele de aplicatie sa fie logic acelasi cu cele utilizate in cazul in care datele sunt fizic cetralizate.

Distribuirea datelor pe mai multe calculatoare dintr-o retea de comunicatii de date nu trebuie sa afecteze programele de aplicatie.

ANSI-SQL nu mentioneaza aceasta regula in specificatiile sale, deoarece este greu de respectat.

**R13 :  Regula versiunii procedurale a SGBD**

Orice componenta procedurala a unui sistem SGBD trebuie sa respecte aceleasi restrictii de integritate ca si componenta relationala.

De exemplu, daca in partea de manipulare a datelor a limbajului relational valoarea dintr-o coloana este de tipul *not null,*orice alta metoda procedurala de accesare a acestei coloane nu trebuie sa permita introducerea unei valori *null* in aceasta coloana. Prin urmare, regulile de integritate exprimate intr-un limbaj relational de un anumit nivel nu pot fi distruse de un limbaj de nivel inferior.

Deoarece regulile lui Codd sunt prea severe pentru a fi respectate de un SGBD operational, s-au formulat criterii minimale de definire a unui sistem relational.

Un SGBD este minimal relational daca satisface urmatoarele conditii:

-        Toate datele din cadrul bazei de date sunt reprezentate prin valori in tabele;

-        Nu exista pointeri observabili de catre utilizator;

-        Sistemul suporta operatorii relationali de proiectie, selectie si join natural, fara limitari impuse din considerente interne.

Un SGBD este *complet relational* daca este minimal relational si satisface in plus conditiile:

-        Sistemul suporta restrictiile de integritate de baza (unicitatea cheii primare, constrangerile de referinta, integritatea entitatii).

-        Sistemul suporta toate operatiile de baza ale algebrei relationale.

Un SGBD relational indeplineste functiile unui SGBD, cu anumite particularitati care decurg din conceptia de organizare a datelor, respectiv din modelul relational. Fiecare SGBD relational implementeaza modelul relational intr-o maniera proprie, care il diferentiaza de restul sistemelor relationale.