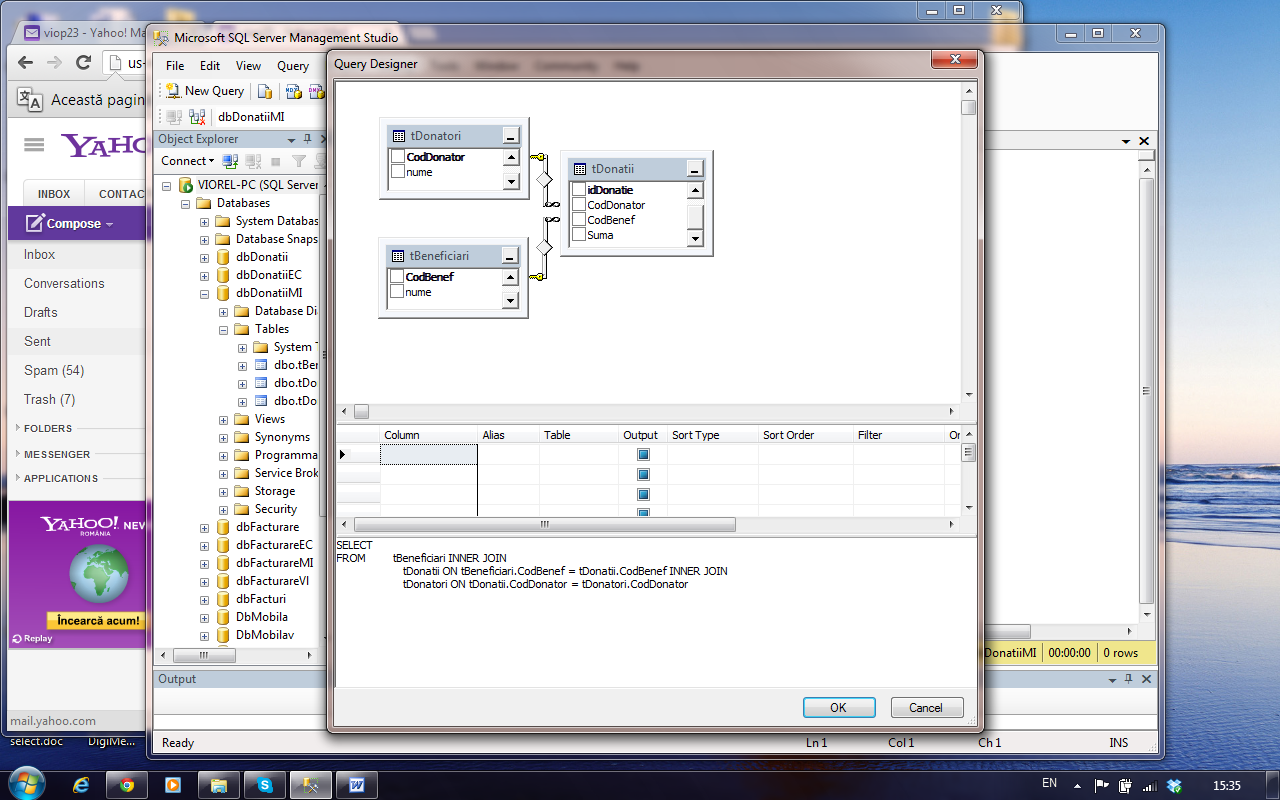
## Interogarea datelor din mai multe tabele

1. Obiectivul lucrării:

Formarea și dezvoltarea abilităților de interogare a bazelor de date.

1. Breviar teoretic cu exerciții și probleme rezolvate

Pentru exemplificarea lucrului cu mai multe tabele, considerăm următorul model relațional:



Scriptul următor crează și populează cu date tabelele aplicației.

create database dbDonatii

go

use dbDonatii

create table tDonatori

( CodDonator char(10) primary key,

nume char(30) not null

)

create table tBeneficiari

( CodBenef char(10) primary key,

nume char(30) not null

)

create table tDonatii

( idDonatie int identity primary key,

CodDonator char(10) null foreign key

references tDonatori(CodDonator),

CodBenef char(10)null foreign key

references tBeneficiari(CodBenef),

Suma int

)

insert into tDonatori(CodDonator,nume)

values ('d1','Popescu'),

('d2','Andreescu'),

('d3','Radulescu'),

('d4','Ionescu'),

('d5','Georgescu')

insert into tBeneficiari(CodBenef,nume)

values ('b1','Ionut'),

('b2','Andreea'),

('b3','Danut'),

('b4','Irina'),

('b5','Marius')

insert into tDonatii(CodDonator,CodBenef,Suma)

values ('d2','b1',5000),

('d1','b2',200),

('d1','b1',400),

('d3','b2',500),

(NULL,'b2',1000), --donator anonim

(NULL,'b1',600), --donator anonim

('d2',NULL,300), --beneficiar anonim

(NULL, NULL,800), --donator si beneficiar anonimi

(NULL, NULL,400) --donator si beneficiar anonimi

*Produsul cartezian*.

Dacă în clauza FROM a unei comenzi SELECT apar mai multe tabele separate prin virgulă sau prin *cross join*, atunci se realizează *produsul cartezian* între rândurile acestor tabele, astfel, rezultatul furnizat de interogare constă în toate combinațiile posibile ale rândurilor tabelelor.

Exemplu:

select \* from tDonatori,tBeneficiari

echivalentă cu:

select \* from tDonatori cross join tBeneficiari

Vom obține un rezultat format din m\*n rânduri, unde m și respectiv n reprezintă numărul de rânduri ale tabelelor tDonatori respectiv tBeneficiari

|  |  |  |  |
| --- | --- | --- | --- |
| CodDonator | nume | CodBenef | nume |
| d1 | Popescu | b1 | Ionut |
| d2 | Andreescu | b1 | Ionut |
| d3 | Radulescu | b1 | Ionut |
| d4 | Ionescu | b1 | Ionut |
| d5 | Georgescu | b1 | Ionut |
| d1 | Popescu | b2 | Andreea |
| d2 | Andreescu | b2 | Andreea |
| d3 | Radulescu | b2 | Andreea |
| d4 | Ionescu | b2 | Andreea |
| d5 | Georgescu | b2 | Andreea |
| d1 | Popescu | b3 | Danut |
| d2 | Andreescu | b3 | Danut |
| d3 | Radulescu | b3 | Danut |
| d4 | Ionescu | b3 | Danut |
| d5 | Georgescu | b3 | Danut |
| d1 | Popescu | b4 | Irina |
| d2 | Andreescu | b4 | Irina |
| d3 | Radulescu | b4 | Irina |
| d4 | Ionescu | b4 | Irina |
| d5 | Georgescu | b4 | Irina |
| d1 | Popescu | b5 | Marius |
| d2 | Andreescu | b5 | Marius |
| d3 | Radulescu | b5 | Marius |
| d4 | Ionescu | b5 | Marius |
| d5 | Georgescu | b5 | Marius |

*Joncțiuni.*

O *joncțiune* între două sau mai multe tabele este un produs cartezian cu restricții. O joncțiune permite asocierea logică a rândurilor diferitelor tabele. Rândurile unui tabel pot fi puse în legătură cu rândurile altui tabel conform valorilor unor coloane ale lor sau ale unor expresii obținute pe baza coloanelor. In felul acesta între tabele apar condiţii de corelare, asociere sau joncțiune (join).

Condiţiile de corelare se realizează de obicei pe baza valorilor comune ale coloanelor cheie primară şi cheie străină.

Putem crea joncțiuni (asocieri) între tabele atât în clauza WHERE cât şi în clauza FROM a interogării.

Sintaxa uzuală, pentru folosirea asocierilor în clauza WHERE (în stilul ANSI SQL 89) este următoarea:

SELECT lista\_de\_coloane

FROM tabel\_A, tabel\_B

WHERE tabel\_A.coloana1=tabel\_B.coloana2;

Asocierile în clauza FROM au fost introduse în varianta SQL 92 a standardului ANSI (American National Standards Institute). Sintaxa uzuală pentru folosirea asocierilor în clauza FROM (în stilul ANSI SQL 92) este următoarea:

SELECT lista\_de\_coloane

FROM tabel\_A INNER | LEFT [OUTER] | RIGHT [OUTER] | FULL [OUTER] join tabel\_B ON tabel\_A.coloana1=tabel\_B.coloana2;

OUTER este optional.

Pentru asocieri multiple sintaxa clauzei FROM este:

FROM tabel\_A JOIN tabel\_B ON conditia1 JOIN tabel\_C ON conditia2 JOIN…

Tipurile de joncțiuni utilizate în interogarea datelor din mai multe tabele sunt:

* INNER JOIN (joncțiune internă)
* LEFT OUTER JOIN (joncțiune externa la stânga)
* RIGHT OUTER JOIN (joncțiune externa la dreapta)
* FULL OUTER JOIN (joncțiune externa şi la stânga şi la dreapta)
* **Inner join** (asociere internă) – rezultatul include numai acele rânduri ale produsului cartezian al celor două tabele care corespund condiţiilor de asociere.

SELECT ... FROM tabel\_A INNER JOIN tabel\_B ON condiţii\_de\_asociere

selectează rândurile din tabelele A şi B care corespund condiţiilor de asociere.

* **Outer join** (asociere externă) - este văzută ca opusul lui *Inner join*. Se includ rânduri dintr-un tabel chiar dacă nu există corespondent în cealaltă tabel. Poate fi de tip:

1. **left join** (asociere la stânga)

*Sintaxa*

SELECT ...

FROM tabel\_A **LEFT OUTER JOIN** tabel\_B ON condiţii\_de\_asociere

selectează toate randurile din *tabel\_A* pe care le completează cu informaţii din *tabel\_B*, în măsura în care satisfac condiţiile de join; acolo unde nu vor exista informaţii corespondente în tabel\_B, acestea vor primi valoarea NULL.

1. **right join** (asociere la dreapta)

*Sintaxa*

SELECT ...FROM tabel\_A RIGHT OUTER JOIN tabel\_B

ON condiţii\_de\_asociere

selectează toate rîndurile din tabel\_B, pe care le completează cu informaţii din tabel\_A, în măsura în care satisfac condiţiile de join; acolo unde nu vor exista informaţii corespondente în tabel\_A, acestea vor primi loarea NULL.

1. **full join**

*Sintaxa*

SELECT ...FROM tabel\_A FULL OUTER JOIN tabel\_B

ON condiţii\_de\_asociere

Este un INNER JOIN completat cu toate randurile din tabel\_A care nu au corespondent in tabel\_B şi toate randurile din tabel\_B care nu au corespondent în tabel\_A. Acolo unde nu vor exista informaţii, acestea vor fi completate cu NULL.

**Auto-asocieri**

Sunt utile atunci când avem de-a face cu o asociere recursivă (între o tabelă şi ea însăşi)

SELECT lista\_de\_coloane FROM tabel\_A as A INNER | LEFT [OUTER] | RIGHT [OUTER] | FULL [OUTER] tabel\_A as B ON A.coloana1=B.coloana2;

Exerciții

1. Să se afișeze conținutul tabelelor tDonatori, tBeneficiari, tDonatii

select \* from tDonatori

CodDonator nume

---------- ------------------------------

d1 Popescu

d2 Andreescu

d3 Radulescu

d4 Ionescu

d5 Georgescu

(5 row(s) affected)

select \* from tBeneficiari

CodBenef nume

---------- ------------------------------

b1 Ionut

b2 Andreea

b3 Danut

b4 Irina

b5 Marius

(5 row(s) affected)

select \* from tDonatii

idDonatie CodDonator CodBenef Suma

----------- ---------- ---------- -----------

1 d2 b1 5000

2 d1 b2 200

3 d1 b1 400

4 d3 b2 500

5 NULL b2 1000

6 NULL b1 600

7 d2 NULL 300

8 NULL NULL 800

9 NULL NULL 400

(9 row(s) affected)

**2. Să se afișeze donatorii cu donațiile lor**

select nume as [nume donator],suma as [suma donata]

from

tDonatori as A

inner join

tDonatii as B on A.CodDonator=B.CodDonator

nume donator suma donata

------------------------------ -----------

Andreescu 5000

Popescu 200

Popescu 400

Radulescu 500

Andreescu 300

(5 row(s) affected)

Se observă că in rezultat apar doar rândurile din cele două tabele care se potrivesc la cheie. Nu apar în rezultat donatorii care încă nu au făcut donații și nici sumele donate de donatori anonimi.

3. Să se afișeze toti donatorii impreună cu eventualele lor donații.

select nume as [nume donator],suma as [suma donata]

from

tDonatori as A

left join

tDonatii as B on A.CodDonator=B.CodDonator

nume donator suma donata

------------------------------ -----------

Popescu 200

Popescu 400

Andreescu 5000

Andreescu 300

Radulescu 500

Ionescu NULL

Georgescu NULL

(7 row(s) affected)

In rezultat apar toti donatorii impreună cu donațiile lor. Lipsesc donațiile făcute de donatori anonimi.

**4. Să se afișeze toate donațiile împreună cu donatorii corespunzători lor.**

select nume as [nume donator],suma as [suma donata]

from

tDonatori as A

right join

tDonatii as B on A.CodDonator=B.CodDonator

nume donator suma donata

------------------------------ -----------

Andreescu 5000

Popescu 200

Popescu 400

Radulescu 500

NULL 1000

NULL 600

Andreescu 300

NULL 800

NULL 400

(9 row(s) affected)

In rezultat apar toate donațiile. Unde nu există donator corespondent, numele donatorului este completat cu NULL.

**5. Să se afișeze toți donatorii împreună cu toate donațiile existente.**

select nume as [nume donator],suma as [suma donata]

from tDonatori as A full join tDonatii as B

on A.CodDonator=B.CodDonator

Popescu 200

Popescu 400

Andreescu 5000

Andreescu 300

Radulescu 500

Ionescu NULL

Georgescu NULL

NULL 1000

NULL 600

NULL 800

NULL 400

(11 row(s) affected)

**6. Să se afișeze nume donator, nume beneficiar și suma donată**

select A.nume as Donator,C.nume as Beneficiar,suma

from

tDonatori as A

inner join

tDonatii as B on A.CodDonator=B.CodDonator

inner join

tBeneficiari as C on B.CodBenef=C.CodBenef

Donator Beneficiar suma

------------------------------ ------------------------------ -----------

Andreescu Ionut 5000

Popescu Andreea 200

Popescu Ionut 400

Radulescu Andreea 500

(4 row(s) affected)

Se observă că această listă nu conține toate donatiile

**7. Să se afișeze toate donațiile cu donatorii și beneficiarii corespunzători**

select A.nume as Donator,C.nume as Beneficiar,suma

from

tDonatori as A right join tDonatii as B on A.CodDonator=B.CodDonator

left join tBeneficiari as C on B.CodBenef=C.CodBenef

Donator Beneficiar suma

------------------------------ ------------------------------ -----------

Andreescu Ionut 5000

Popescu Andreea 200

Popescu Ionut 400

Radulescu Andreea 500

NULL Andreea 1000

NULL Ionut 600

Andreescu NULL 300

NULL NULL 800

NULL NULL 400

(9 row(s) affected)

Comentariu:

Asocierea

tDonatori as A

right join

tDonatii as B on A.CodDonator=B.CodDonator

crează un tabel virtual ce conține toate donațiile împreună cu donatorii corespondenți (dacă există). Mai departe, prin asocierea

left join

tBeneficiari as C on B.CodBenef=C.CodBenef

acest tabel virtual va fi asociat la stânga( prin urmare nu se vor pierde randuri ale tabelului virtual creat) cu tBeneficiari, astfel că tabelul virtual precedent va fi completat cu beneficiarii corespondenti sau null in cazul inexistentei beneficiarului.

**8. Să se afișeze toti donatorii(inclusiv pe cei care si-au declarat intentia de a dona dar inca nu au facut-o) cu donațiile și beneficiarii corespunzători**

select A.nume as Donator,C.nume as Beneficiar,suma

from

tDonatori as A left join tDonatii as B on A.CodDonator=B.CodDonator

**left join** tBeneficiari as C on B.CodBenef=C.CodBenef

Donator Beneficiar suma

------------------------------ ------------------------------ -----------

Popescu Andreea 200

Popescu Ionut 400

Andreescu Ionut 5000

Andreescu NULL 300

Radulescu Andreea 500

Ionescu NULL NULL

Georgescu NULL NULL

(7 row(s) affected)

**9. Să se afișeze toti beneficiarii inregistrati in tabelul tBeneficiari impreuna cu donațiile și donatorii corespunzatori**

select A.nume as Donator,C.nume as Beneficiar,suma

from ( tDonatori as A left join tDonatii as B

on A.CodDonator=B.CodDonator)

right join tBeneficiari as C on

B.CodBenef=C.CodBenef

Donator Beneficiar suma

------------------------------ ------------------------------ -----------

Andreescu Ionut 5000

Popescu Ionut 400

Popescu Andreea 200

Radulescu Andreea 500

NULL Danut NULL

NULL Irina NULL

NULL Marius NULL

(7 row(s) affected)

**10. Exemple de utilizare a asocierii full join**

select A.nume as Donator,C.nume as Beneficiar,suma

from ( tDonatori as A left join tDonatii as B

on A.CodDonator=B.CodDonator)

full join tBeneficiari as C

on B.CodBenef=C.CodBenef

Donator Beneficiar suma

------------------------------ ------------------------------ -----------

Popescu Andreea 200

Popescu Ionut 400

Andreescu Ionut 5000

Andreescu NULL 300

Radulescu Andreea 500

Ionescu NULL NULL

Georgescu NULL NULL

NULL Danut NULL

NULL Irina NULL

NULL Marius NULL

(10 row(s) affected)

**select A.nume as Donator,C.nume as Beneficiar,suma**

**from ( tDonatori as A full join tDonatii as B**

**on A.CodDonator=B.CodDonator)**

**full join tBeneficiari as C on**

**B.CodBenef=C.CodBenef**

Donator Beneficiar suma

------------------------------ ------------------------------ -----------

Popescu Andreea 200

Popescu Ionut 400

Andreescu Ionut 5000

Andreescu NULL 300

Radulescu Andreea 500

Ionescu NULL NULL

Georgescu NULL NULL

NULL Andreea 1000

NULL Ionut 600

NULL NULL 800

NULL NULL 400

NULL Irina NULL

NULL Marius NULL

NULL Danut NULL

(14 row(s) affected)

**11. Sa se afiseze donatiile catre beneficiarul ‘andreea’**

select A.nume,C.nume as donator,B.suma

from tBeneficiari as A inner join tDonatii as B

on A.codBenef=B.CodBenef

left join tDonatori as C on B.codDonator=C.CodDonator

where A.nume='andreea'

**Conditia** A.nume='andreea' **poate fi plasata si in clauza de asociere:**

select A.nume,C.nume as donator,B.suma

from tBeneficiari as A inner join tDonatii as B

on A.codBenef=B.CodBenef **and A.nume='andreea'**

left join tDonatori as C on B.codDonator=C.CodDonator

nume donator suma

--------------- ------------ -----------

Andreea Popescu 200

Andreea Radulescu 500

Andreea NULL 1000

(3 row(s) affected)

**12. Utilizarea clauzei group by**

select A.codBenef as [Cod beneficiar], Nume as [Nume beneficiar], sum(suma) as [total donatii]

from tBeneficiari as A inner join tDonatii as B on A.CodBenef=B.CodBenef

group by A.CodBenef,Nume

Cod beneficiar Nume beneficiar total donatii

b1 Ionut 6000

b2 Andreea 1700

select A.codBenef as [Cod beneficiar], Nume as [Nume beneficiar],sum(suma) as [total donatii]

from tBeneficiari as A left join tDonatii as B on A.CodBenef=B.CodBenef

group by A.CodBenef,Nume

Cod beneficiar Nume beneficiar total donatii

b1 Ionut 6000

b2 Andreea 1700

b3 Danut NULL

b4 Irina NULL

b5 Marius NULL

select A.codBenef as [Cod beneficiar], Nume as [Nume beneficiar],sum(suma) as [total donatii]

from tBeneficiari as A right join tDonatii as B on A.CodBenef=B.CodBenef

group by A.CodBenef,Nume

Cod beneficiar Nume beneficiar total donatii

NULL NULL 1500

b1 Ionut 6000

b2 Andreea 1700

select A.codBenef as [Cod beneficiar], Nume as [Nume beneficiar],sum(suma) as [total donatii]

from tBeneficiari as A full join tDonatii as B on A.CodBenef=B.CodBenef

group by A.CodBenef,Nume

Cod beneficiar Nume beneficiar total donatii

b1 Ionut 6000

b2 Andreea 1700

b3 Danut NULL

b4 Irina NULL

b5 Marius NULL

NULL NULL 1500

select A.nume,sum(suma) as Total

from tBeneficiari as A inner join tDonatii as B

on A.codBenef=B.CodBenef

where A.nume='andreea' group by A.nume

nume Total

------------------------------ -----------

Andreea 1700

(1 row(s) affected)

**13. Auto asocieri**

create table tAnimale

( cod int primary key,

nume varchar(30),

codMama int

)

insert into tAnimale

values

(1,'Bura',null),

(2,'Azorel',1),

(3,'Florica',1),

(4,'Roca',null),

(5,'Bobita',4)

select A.Nume ,B.nume as [Nume mama]

from tAnimale as A left join tAnimale as B on A.codmama=B.cod

Nume Nume mama

Bura NULL

Azorel Bura

Florica Bura

Roca NULL

Bobita Roca

Următoarele comenzi select exemplifică utilizarea câmpurilor calculate, a funcțiilor de agregare și a clauzei *into tabel*

select codFactura,data,A.codClient,nume,C.codProd,

denumire,cantitate,pret, cantitate\*pret as valoare

from

tFacturi as A

inner join

tClienti as B on A.codClient = B.codClient

inner join

tDetaliiFacturi as C on A.codFactura = C.nrFactura

inner join

tProduse as D on C.codProd = D.codProdus

select A.codClient,nume,A.codFactura,sum(cantitate\*pret)

as [Total factura]

from tFacturi as A

inner join tClienti as B

on A.codClient = B.codClient

inner join tDetaliiFacturi as C

on A.codFactura = C.nrFactura

inner join tProduse as D

on C.codProd = D.codProdus

group by A.codClient,nume,codFactura

drop table Centralizator

select codProdus,denumire,isnull(sum(cantitate\*pret),0) as Total

into Centralizator

from tProduse as A

left join tDetaliiFacturi as B

on A.codProdus = B.codProd

group by codProdus,Denumire

insert into Centralizator

select codProdus,denumire,isnull(sum(cantitate\*pret),0) as Total

from tProduse as A

left join tDetaliiFacturi as B

on A.codProdus = B.codProd

group by codProdus,Denumire