

Structura bazei de date utilizata ca exemplu la seminar - Se considera activitatea de evidenta a comenzilor incheiate de o societate comerciala cu diverse firme prin intermediul agentilor angajati in cadrul societatii. Comenzile contin produse aflate in depozitul societatii, iar pentru fiecare produs se cunoaste in permanenta stocul existent.

FIRME

<u>CODFIRMA</u>	DENFIRMA	LOC	CONTBANCA	ZONA
-----------------	----------	-----	-----------	------

AGENTI

<u>CODAGENT</u>	NUMEAGENT	DATAANG	DATANAST	ZONA	FUNCTIE	CODSEF
-----------------	-----------	---------	----------	------	---------	--------

COMENZI

<u>NRCOM</u>	<u>CODFIRMA</u>	<u>CODAGENT</u>	DATA
--------------	-----------------	-----------------	------

RANDCOM

<u>NRCOM</u>	<u>CODPRODUS</u>	CANT	PRET	TERMENLIVR
--------------	------------------	------	------	------------

PRODUSE

<u>CODPRODUS</u>	DENPRODUS	UM	STOC
------------------	-----------	----	------

RECAPITULARE COMENZI LDD

1. CREAREA TABELELOR – COMANDA CREATE TABLE

```
CREATE TABLE nume_tabelă  
(  
    .... definirea câmpurilor și a tipurilor de date aferente  
    ... definirea restricțiilor de integritate  
);
```

Definirea restricțiilor – se poate realiza la nivel de câmp (*in-line*) sau la nivelul tabelului (*out-of-line*):

Sintaxa generală:

Constraint *nume_restricție* **tip_restricție** [(*câmpurile cărora li se aplică restricția*)]

Tipuri de restricții:

1. Restricția de tip PRIMARY KEY:

Constraint *nume_restricție* PRIMARY KEY [(*câmpuri care formează cheia primară*)]

2. Restricția de tip FOREIGN KEY:

Constraint *nume_restricție* FOREIGN KEY (*câmpul cheie externă*) REFERENCES *Tabelă_părinte* (*câmp cheie primară*)

3. Restricția de tip NOT NULL:

Se definește **numai** la nivelul câmpului căruia i se aplică restricția:

Ex: *nume* VARCHAR2(20) NOT NULL

4. Restricția de tip UNIQUE:

Constraint *nume_restricție* UNIQUE [(*câmp cheie unică*)]

5. Restricția de tip CHECK:

Constraint *nume_restricție* CHECK [(*condiție asupra unui câmp*)]

2. MODIFICAREA STRUCTURII TABELELOR - COMANDA ALTER

Realizează următoarele:

- Modificarea structurii tabelului: ADD, MODIFY, DROP COLUMN, SET UNUSED
- Modificarea restricțiilor de integritate: ADD, MODIFY, DROP, DISABLE CONSTRAINT
- Redenumeste tabela: RENAME

```
ALTER TABLE nume_tabelă  
- ADD (definire câmpuri);
```

- **MODIFY** (*redefinire câmpuri existente*);
- **DROP COLUMN** *câmp*;
- **ADD CONSTRAINT** *nume_restricție* **TIP_RESTRICȚIE**;
- **DROP CONSTRAINT** *nume_restricție*;
- **DISABLE CONSTRAINT** *nume_restricție*;
- **ENABLE CONSTRAINT** *nume_restricție*;
- **RENAME TO** *nume_nou_tabelă*;

3. STERGEREA TABELELOR – COMANDA DROP

DROP TABLE *nume_tabelă* **CASCADE CONSTRAINTS**;

DROP TABLE *nume_tabela*

PARTEA II - ACTUALIZAREA TABELELOR PRIN COMENZI LMD

Comanda	Scop
INSERT	Adauga o noua inregistrare in tabela
SELECT	Regaseste inregistrari in tabele sau vederi.
DELETE	Sterge inregistrari din tabele
UPDATE	Modifica valorile unor inregistrari din tabele

*Dupa actualizarea datelor se va utiliza comanda `select * from [nume_tabela_actualizata];` pentru a vizualiza rezultatele!*

2.1. ADAUGAREA DATELOR – COMANDA *INSERT*

Datele se pot adauga in tabele in mai multe moduri:

- precizand explicit in sintaxa comenzii valorile
- cu ajutorul variabilelor de substitutie
- pe baza valorilor din alte tabele

Precizarea explicita a valorilor introduse se realizeaza cu comanda:

INSERT INTO TABELA VALUES ([LISTA DE VALORI PENTRU FIECARE ATRIBUT]);

*insert into firme values('10','SC ALFA SRL','Cluj','bcr1000','TRANSILVANIA');
 insert into firme values('20','SC MEDIA SA','Bucuresti','brd1111','MUNTENIA');
 insert into firme values('30','SC SOFTY SRL','Ploiesti','bcr2222','MUNTENIA');
 insert into firme values('40','SC MEGA SRL','Iasi','brd3333','MOLDOVA');
 insert into firme values('50','SC STAR SA','Timisoara','bcr4444','BANAT');
 insert into firme values('60','SC Sas SA','Timisoara','bcr333','BANAT');*

*insert into agenti
 values('1','Toma Alina',to_date('feb 3,04','mon dd,yy'),to_date('jan 23,04','mon dd,yy'),'BANAT','ECONOMIST','2');
 insert into agenti
 values('2','Rotaru Maria',to_date('apr 12,03','mon dd,yy'),to_date('feb 13,54','mon dd,yy'),'MOLDOVA','DIRECTOR','2');
 insert into agenti
 values('3','Popescu Ionel',to_date('may 30,04','mon dd,yy'),to_date('Sep 3,65','mon dd,yy'),'MUNTENIA','CONTABIL','1');*

*insert into produse
 values('111','napolitane','buc','1000');
 insert into produse
 values('222','ciocolata','buc','4000');
 insert into produse
 values ('333','biscuiti','pac','3000');*

```
insert into produse  
values('444','servetele','pac','1100');
```

```
insert into comenzi  
values('100','10','2',to_date('oct 12,04','mon dd,yy'));  
insert into comenzi  
values('200','20','3',to_date('oct 6,04','mon dd,yy'));  
insert into comenzi  
values('300','40','1',to_date('nov 30,04','mon dd,yy'));  
insert into comenzi  
values('400','30','2',to_date('dec 12,04','mon dd,yy'));  
insert into comenzi  
values('500','50','1',to_date('jan 15,05','mon dd,yy'));
```

```
insert into rindcom  
values('100','111','150','5000',to_date('oct 31,04','mon dd,yy'));  
insert into rindcom  
values('200','222','300','20000',to_date('nov 30,04','mon dd,yy'));  
insert into rindcom  
values('300','444','1000','1500',to_date('dec 25,04','mon dd,yy'));  
insert into rindcom  
values('300','111','200','5000',to_date('jan 31,05','mon dd,yy'));  
insert into rindcom  
values('400','333','1500','5000',to_date('jan 31,05','mon dd,yy'));  
insert into rindcom  
values('500','111','100','5000',to_date('feb 20,05','mon dd,yy'));
```

Atentie! Pt data se va utiliza functia de conversie to_date.

Ex: to_date('jan 20,05','mon dd,yy')

sau

to_date('17-12-1980','dd-mm-yyyy')

b) adaugarea datelor pe baza valorilor din alte tabele:

```
INSERT INTO TABELA SELECT [LISTA DE CAMPURI]  
FROM TABELA_SURSA  
WHERE [CONDITIE];
```

Exemple:

Sa se creeze tabela STOC_MIN cu aceeasi structura cu a tablei PRODUSE care sa contina informatii despre produsele cu stocul mai mic decat 1000 unitati.

Creare tabela stoc_min, cream tabela goala initial :

Create table stoc_min

as select * from produse where 2=3;

Inserare date in tabela stoc_min :

Insert into stoc_min select *

from produse where stoc<=1000;

2.2. MODIFICAREA DATELOR – COMANDA UPDATE

Sintaxa este:

**UPDATE [TABELA]
SET [COLOANA] = [VALOARE]
WHERE [CONDITIE];**

Exemple:

Sa se scada cu 100 de pachete stocul de servetele:

```
UPDATE PRODUSE  
SET STOC=STOC-100  
WHERE lower(denprodus)='servetele';
```

```
SELECT * FROM PRODUSE;
```

Executati comanda
rollback;

Ce observati?

Sa se modifice stocul produsului 333 cu stocul produsului cu codul 111:

```
UPDATE PRODUSE  
SET STOC=(SELECT STOC FROM PRODUSE  
           WHERE CODPRODUS='111')  
WHERE CODPRODUS='333';
```

Rollback;

Sa se modifice stocul produselor cu stocul produsului cu codul 111, dar numai pentru produsele aflate in comanda cu nr 400:

```
UPDATE PRODUSE  
SET PRET=(SELECT PRET FROM PRODUSE WHERE CODPRODUS='111')  
WHERE CODPRODUS IN (SELECT CODPRODUS FROM RINDCOM WHERE  
NRCOM=400);
```

2.3. STERGEREA DATELOR – COMANDA DELETE

Sintaxa este:

**DELETE FROM [TABELA]
WHERE [CONDITIE];**

Exemple:

Sa se stearga produsele comandate pt care cant<200;

```
DELETE FROM rindcom  
WHERE cant<200;
```

Sa se stearga produsele comandate pt care termen livrare>feb 2005:

```
DELETE FROM rindcom  
WHERE termenliv>TO_DATE('01-02-05','DD-MM-YY');
```

Sa se stearga toate inregistrarile din tabela stoc_min:

```
DELETE FROM stoc_min;
```

2.4. SELECTIA DATELOR – COMANDA SELECT

SELECT – realizeaza selectia si regasirea datelor din tabele

```
SELECT [DISTINCT] { * , tabelă1.câmp1 [alias] , expresii AS ALIAS ...}  
FROM tabelă1, tabelă2,....  
WHERE {condiții, precizarea legăturilor dintre tabele}  
GROUP BY tabelă .câmp  
HAVING {condiții impuse valorilor de grup}  
ORDER BY tabelă .câmp ASC/DESC;
```

unde:

SELECT	specifică atributele selectate;
DISTINCT	suprimă valorile duplicate;
*	selectează toate atributele;
atribut	selectează coloana numită;
expresie	permite construirea de expresii si valori noi
alias	denumiri pentru atributele selectate;
FROM tabele	specifică tabelele ce conțin coloanele selectate.
WHERE	clauza permite specificarea conditiilor si a criteriilor de selectie a datelor
GROUP BY	se precizeaza campul dupa care vor fi grupate datele in cazul expresiilor si functiilor de grup (SUM(), AVG(), COUNT(), MIN(), MAX())
HAVING	in cazul functiilor de grup conditiile impuse acestora se precizeaza in clauza HAVING
ORDER BY	precizeaza ordonarea in functie un anumite campuri ascendent (ASC) –implicit sau descendent (DESC)

Frazele SQL:

Nu sunt case sensitive;
Pot fi scrise pe mai multe linii;

Cuvintele cheie nu pot fi prescurtate sau scrise pe mai multe linii.

În construirea frazelor SQL se utilizează următorii operatori:

<, >, =, >=, <=, NOT	Operatori de comparație
BETWEEN ... AND ...	între două valori (inclusiv).
SELECT codprodus, cant FROM rindcom WHERE cant BETWEEN 10 AND 15;	
IN(listă)	egal cu oricare valoare din listă
SELECT codagent, numeagent FROM agenti WHERE codagent IN ('1', '3');	
SELECT codprodus, cant FROM rindcom WHERE cant IN (400,500,1000);	
LIKE	similar cu un șablon % - oricâte caractere; _ - un caracter;
SELECT denfirma, loc FROM firme WHERE loc LIKE 'B%';	
SELECT denfirma, loc FROM firme WHERE loc LIKE '_I%';	
IS NULL	are valoarea NULL
SELECT denfirma, loc FROM firme WHERE loc IS NULL;	

- Atributele se specifică în ordinea în care se doresc a fi afișate, nu obligatoriu în ordinea în care apar în descrierea tabelor.
- Afișarea se face implicit cu litere mari, la stânga pentru datele de tip dată calendaristică și caracter, și la dreapta pentru datele numerice.
- Valoarea **NULL**, ca și în alte limbaje, semnifică valoare ne-disponibilă, ne-alocată, și nu este același lucru cu „blank” sau „zero”.
- Cu ajutorul frazelor select se pot extrage informații din baza de date.
- Utilizând această instrucțiune se pot realiza toate cele trei operații specifice modelului relațional.

Selecția: Operator unar, prin care se obține o nouă relație care conține toate atributele relației inițiale și un număr redus de tupluri. Reducerea se face după o condiție numită condiție de selecție.

Proiecția: Operator unar, prin care se obține o nouă relație care conține un număr redus de atribute față de relația inițială și toate valorile sau combinațiile distincte de valori ale acestor atribute. Atributele care se regăsesc în relația rezultată se numesc atribute de proiecție.

Joncțiunea: Operator binar, caz particular de produs cartezian. Relația rezultat va conține tuplurile corespunzătoare atributelor de join care se află într-o anumită relație. Cel mai utilizat caz de joncțiune este acela în care attributele de join au aceeași semnificație, relația în care acestea trebuie să se afle este de egalitate, unul dintre attribute e cheie primară într-una dintre relații, iar celălalt este cheie externă.

EXEMPLE:

1. Să se selecteze toți agenții din tabela agenți:

```
SELECT * FROM agenți;
```

2. Să se selecteze campurile codagent, numeagent și zona din tabela agenți:

```
SELECT codagent, numeagent, zona FROM agenți;
```

3. Să se selecteze numai firmele din București:

```
SELECT * FROM firme
WHERE upper(loc) LIKE '%BUCURESTI%';
```

4. Să se selecteze comenzile încheiate de agentul cu codul = 1:

```
SELECT * FROM comenzi
WHERE codagent = '1';
```

Realizarea Join-urilor (joncțiunilor) între relații. Tipuri de join-uri

a. Join de egalitate (de echivalență - equijoin)

5. Să se selecteze comenzile emise de agenții societății (în clauza WHERE se va preciza condiția de legătură dintre tabele)

```
SELECT agenți.*, comenzi.*
FROM agenți, comenzi
```

```
WHERE agenti.codagent= comenzi.codagent;
```

6. Sa se selecteze comenzile incheiate de agentul 'Toma Alina' numai in luna octombrie:

```
SELECT agenti.*, comenzi.*  
FROM agenti, comenzi  
WHERE agenti.codagent = comenzi.codagent  
AND lower(comenzi.data) like '%oct%'  
AND upper(agenti.numagent) = 'TOMA ALINA';
```

7. Sa se calculeze valoarea fiecarui produs ($val_prod = cant * pret$) si sa se afiseze pretul, cantitatea, valoarea si stocul disponibil:

```
SELECT produse.denprodus, rindcom.cant, rindcom.pret,  
rindcom.cant * rindcom.pret AS Val_Prod, produse.stoc  
FROM produse, rindcom  
WHERE produse.codprodus= rindcom. codprodus;
```

8. Sa se selecteze numai produsele cu valoarea cuprinsa intre 1 si 3 mil:

```
SELECT produse.denprodus, rindcom.cant, rindcom.pret,  
rindcom.cant * rindcom.pret AS Val_Prod, produse.stoc  
FROM produse, rindcom  
WHERE produse.codprodus= rindcom. codprodus  
and rindcom.cant * rindcom.pret BETWEEN 500000 AND 2000000;
```

b. Join extern

9. Să se afişeze codul produsului, denumirea produsului şi cantitatea comandata corespunzatoare:

```
SELECT p.codprodus, p.denprodus, rc.cant, rc.pret  
FROM produse p, rindcom rc  
WHERE p.codprodus = rc.codprodus (+);
```

c. Join tabelă cu aceeaşi tabelă

10. Să se afişeze numele fiecarui agent şi numele sefului direct superior:

```
SELECT agent.numagent||' lucreaza pentru: '||sefi. numagent  
FROM agenti agent, agenti sefi  
WHERE agent.codsef=sefi.codagent;
```

Realizarea Subcererilor (se utilizeaza 2 fraze SELECT imbricate)

11. Sa se selecteze firmele care sunt in aceaasi zona cu firma SC MEDIA SA:

```
SELECT * FROM firme  
WHERE zona =  
(SELECT zona FROM firme WHERE denfirma= 'SC MEDIA SA');
```

12. Să se afișeze produsele care au prețul unitar cel mai mic:

```
SELECT p.denprodus, rc.pret  
FROM produse p, rindcom rc  
WHERE p.codprodus = rc.codprodus  
And rc.pret=(select min(rindcom.pret) from rindcom);
```