

## CAPITOLUL 7. APLICATII INFORMATICE UTILIZAND LIMBAJUL SQL

### 7.1. Aplicație informatică pentru activitatea de salarizare

1) Folosindu-se instrucțiunile SQL, să se creeze tabelele DatePers, DateSal, Impozitar, Pontaj, SporVechime, Taxe, Deduceri.

**CREATE TABLE DatePers**

```
(  
    codang          number (5) primary key,  
    nume           varchar2 (35),  
    cnp            varchar2 (13),  
    datan          date,  
    adresa         varchar2 (30),  
    localitate    varchar2 (15),  
    telefon       varchar2 (12)  
);
```

**CREATE TABLE DateSal**

```
(  
    codang          number(5) references DatePers (codang),  
    functia        varchar2 (10),  
    salbaza        number (15),  
    persintr       number (2),  
    vechime        number (3),  
    codsef number (5) references DatePers(codang)  
);
```

**CREATE TABLE Impozitar**

```
(  
    linie          number (5) primary key,  
    dela           number (15),  
    panala        number (15),  
    suma          number (15),  
    procent       number (3)  
);
```

**CREATE TABLE Pontaj**

```
(
```

```

        codang      number (5) references DatePers(codang),
        luna        number (3),
        zilelucr     number(3),
        orezi       number(3),
        zileco       number(3),
        zilecm       number(3),
        orelucrate   number(4),
        constraint pk primary key(codang,luna)
);

```

**CREATE TABLE SporVechime**

```

(
    nr            number (3) primary key,
    dela         number (3),
    panala       number (3),
    procent      number (3)
);

```

**CREATE TABLE Taxe**

```

(
    den          varchar2 (10) primary key,
    procent      number (2),
    cotamax     number (15)
);

```

**CREATE TABLE Deduceri**

```

(
    den          varchar2 (30) primary key,
    cotasuma     number(15),
    cotaproc     number(2),
    cotamax     number(15)
);

```

2) Să se încarce cu date tabelele create.

```

SQL> DELETE FROM DatePers;
INSERT INTO DatePers VALUES
(100, 'Ion Ion', '1234567890100', '10-JAN-1970', 'Mangaliei 100',
'Constanta', '0722123456');
INSERT INTO DatePers VALUES

```

*(200, 'Popescu Ion', '1234567890200', '10-FEB-1975', 'Tomis 232', 'Constanta', '0744123456');  
INSERT INTO DatePers VALUES  
(300, 'Ionescu Gheorghe', '1234567890300', '10-MAR-1980', 'Ferdinand 48', 'Mangalia', '0788123456');*

*SQL> DELETE FROM DateSal;  
INSERT INTO DateSal VALUES (100, 'Ec', 5000000, 1, 10, 300);  
INSERT INTO DateSal VALUES (200, 'Inginer', 6500000, 2, 5, 300);  
INSERT INTO DateSal VALUES (300, 'Director', 15000000, 0, 15, null);*

*SQL> DELETE FROM Impozitar;  
INSERT INTO Impozitar VALUES (1, 0, 2100000, 0, 18);  
INSERT INTO Impozitar VALUES (2, 2100001, 5200000, 378000,23);  
INSERT INTO Impozitar VALUES (3, 5200001, 8300000, 1091000,28);  
INSERT INTO Impozitar VALUES (4, 8300001, 11600000, 1959000,34);  
INSERT INTO Impozitar VALUES (5, 11600001, 9999999999, 3081000, 40);*

*SQL> DELETE FROM Pontaj;  
INSERT INTO Pontaj VALUES (100, '1', 22, 8, 3, 0, 170);  
INSERT INTO Pontaj VALUES (200, '1', 30, 8, 5, 10, 200);  
INSERT INTO Pontaj VALUES (300, '1', 22, 8, 0, 0, 176);*

*SQL> DELETE FROM SporVechime;  
INSERT INTO SporVechime VALUES (1, 0, 3, 5);  
INSERT INTO SporVechime VALUES (2, 4, 10, 10);  
INSERT INTO SporVechime VALUES (3, 11, 20, 15);  
INSERT INTO SporVechime VALUES (4, 21, 40, 20);*

*SQL> DELETE FROM Taxe;  
INSERT INTO Taxe VALUES ('CASS', 6.5, 9999999999999999);  
INSERT INTO Taxe VALUES ('CAS', 9.5, 15000000);  
INSERT INTO Taxe VALUES ('Somaj', 1, 9999999999999999);*

*SQL> DELETE FROM Deduceri;  
INSERT INTO Deduceri VALUES ('Deducere de baza', 1800000, 0, 1800000);*

**INSERT INTO Deduceri VALUES ('Deducere suplimentara', 0, 0.5, 3600000);**  
**INSERT INTO Deduceri VALUES ('Chelt profesionale', 0, 15, 270000);**

### Interogarea tabelelor bazei de date

1) Să se afișeze informațiile despre angajații firmei.

**SQL> SELECT \* FROM DatePers;**

| CODANGNUME | CNP              | DataN         | ADRESA      | LOCALITATE    | TELEFON              |
|------------|------------------|---------------|-------------|---------------|----------------------|
| 100        | Ion Ion          | 1234567890100 | 10-JAN-1970 | Mangaliei 100 | Constanta 0722123456 |
| 200        | Popescu Ion      | 1234567890200 | 10-FEB-1975 | Tomis 232     | Constanta 0744123456 |
| 300        | Ionescu Gheorghe | 1234567890300 | 10-MAR-1980 | Ferdinand 48  | Mangalia 0788123456  |

2) Să se selecteze toți angajații din Constanța.

**SQL> SELECT \* FROM DatePers**  
**WHERE localitate ='Constanta';**

| CODANGNUME | CNP         | DataN         | ADRESA      | LOCALITATE    | TELEFON              |
|------------|-------------|---------------|-------------|---------------|----------------------|
| 100        | Ion Ion     | 1234567890100 | 10-JAN-1970 | Mangaliei 100 | Constanta 0722123456 |
| 200        | Popescu Ion | 1234567890200 | 10-FEB-1975 | Tomis 232     | Constanta 0744123456 |

3) Să se afișeze numele tuturor angajaților care sunt din localitățile a căror nume începe cu litera M.

**SQL> SELECT nume, localitate**  
**FROM DatePers**  
**WHERE localitate LIKE 'M%';**

| CODANGNUME | CNP              | DataN         | ADRESA      | LOCALITATE   | TELEFON             |
|------------|------------------|---------------|-------------|--------------|---------------------|
| 300        | Ionescu Gheorghe | 1234567890300 | 10-MAR-1980 | Ferdinand 48 | Mangalia 0788123456 |

4) Să se afișeze codul și salariile angajaților care au salariul de bază între 6000000 și 7000000

**SQL> SELECT codang, salbaza**  
**FROM DateSal**  
**WHERE salbaza BETWEEN 6000000 AND 7000000;**

| CODANG | SALBAZA |
|--------|---------|
| 200    | 500000  |

```
SQL> SELECT codang, vechime
FROM DateSal
WHERE vechime IN (10,15);
```

| CODANG | VECHIME |
|--------|---------|
| 100    | 10      |
| 300    | 15      |

```
SQL> SELECT      dela || ' - ' || panala || ' ' || suma || ' + ' ||
                procent || ' % pentru ceea ce depaseste ' || dela
FROM Impozitar;
```

| DELA   '-   PANALA   '   | SUMA    ' + | PROCENT   '%PENTRU CEEA CE DEPASESTE' |
|--------------------------|-------------|---------------------------------------|
| 0 - 2100000              | 0 +         | 18% pentru ceea ce depaseste 0        |
| 2100001 - 5200000        | 378000 +    | 23% pentru ceea ce depaseste 2100001  |
| 5200001 - 8300000        | 1091000 +   | 28% pentru ceea ce depaseste 5200001  |
| 8300001 - 11600000       | 1959000 +   | 34% pentru ceea ce depaseste 8300001  |
| 11600001 - 9999999999999 | 3081000 +   | 40% pentru ceea ce depaseste 11600001 |

```
SQL> SELECT CONCAT (codang, luna), ANGAJAT_LUNA  
LENGTH (concat (codang,luna)) LUNGIME_SIR  
FROM Pontaj;
```

| ANGAJAT_LUNA | LUNGIME_SIR |
|--------------|-------------|
| 1001         | 4           |
| 2001         | 4           |
| 3001         | 4           |

```
SQL> SELECT      ROUND (41000/32000, 2)  2_ZECIMALE,  
                  ROUND (41000/32000, 3)  3_ZECIMALE  
                  FROM DUAL;
```

| 2_ZECIMALE | 3_ZECIMALE |
|------------|------------|
| 1.28       | 1.281      |

9) Să se afișeze angajații care au cuvântul “Ion” în nume (nume și prenume) împreună cu vârsta acestora. Vârsta se va afișa în două moduri: rotunjită în ani și în ani cu luni.

```
SQL> SELECT nume,  
        ROUND ((sysdate-datan)/365, 0)    ANI,  
        ROUND ((sysdate-datan)/365, 1)    ANI_CU_LUNI  
FROM DatePers  
WHERE nume LIKE '%Ion%';
```

| NUME             | ANI | ANI_CU_LUNI |
|------------------|-----|-------------|
| Ion Ion          | 34  | 34.2        |
| Popescu Ion      | 29  | 29.1        |
| Ionescu Gheorghe | 24  | 24          |

10) Să se afișeze numele angajaților și data împlinirii limitei de vârstă pentru pensionare (62 de ani) precum și numărul de luni rămase până la pensionare (62ani\*12luni).

```
SQL> SELECT nume,  
        ADD_MONTHS (datan, 62*12) DATA_PENSIONARE  
        MONTHS_BETWEEN(ADD_MONTHS(datan,62*12),sysdate)  
        LUNI_PENSIONARE  
FROM DatePers;
```

| NUME             | DATA_PENSIONARE | LUNI_PENSIONARE |
|------------------|-----------------|-----------------|
| Ion Ion          | 10-JAN-32       | 334.11          |
| Popescu Ion      | 10-FEB-37       | 395.11          |
| Ionescu Gheorghe | 10-MAR-42       | 456.11          |

11) Să afișeze numele angajaților și ultima zi a lunii corespunzătoare datei de naștere a angajaților din localitatea Mangalia

```
SQL> SELECT nume,  
        LAST_DAY (datan) ULTIMA_ZI_DIN_LUNA  
FROM DatePers  
WHERE localitate='Mangalia';
```

| NUME             | ULTIMA_ZI_DIN_LUNA |
|------------------|--------------------|
| Ionescu Gheorghe | 31-MAR-80          |

12) Să se afișeze următoarea zi de Sămbătă (după DataCurentă->sysdate)

```
SQL> SELECT NEXT_DAY (sysdate,'Saturday') URMATOAREA_SAMBATA  
FROM DUAL;
```

URMATOAREA\_SAMBATA

08-OCT-05

13) Să se afișeze numele angajaților și data nașterii acestora într-un format 'MM/YYYY' (M=Month=Luna, Y=Year=An)

```
SQL> SELECT nume,  
TO_CHAR(datan,'MM/YYYY') LUNA_AN  
FROM DatePers;
```

| NUME             | LUNA_AN |
|------------------|---------|
| Ion Ion          | 01/1970 |
| Popescu Ion      | 02/1975 |
| Ionescu Gheorghe | 03/1980 |

14) Să se afișeze numele și CNP-ul angajaților născuți pe 10 ianuarie 1970

```
SQL> SELECT cnp  
FROM DatePers  
WHERE datan= TO_DATE ('10 January 1970', 'dd Month YYYY');
```

| NUME    | CNP           |
|---------|---------------|
| Ion Ion | 1234567890100 |

15) Să se afișeze codul angajaților, numele și salariul acestora indexat cu 5% pentru economiști și 10% pentru director. Se stabilește un JOIN pe tabelele DatePers și DateSal pentru identificarea numelui și, respectiv, codul angajaților al căror salariu va fi indexat. Salariul care nu va fi indexat va fi trecut cu SAL\_BAZA în coloana SAL\_INDEXAT (opțiunea DEFAULT din funcția DECODE).

```
SQL> SELECT ds.codang, dp.nume,  
ds.salbaza SAL_BAZA,  
DECODE (functia, 'Ec', salbaza*1.05, 'Director', salbaza*1.1, salbaza )  
SAL_INDEXAT  
FROM DateSal ds, DatePers dp  
WHERE ds.codang = dp.codang;
```

| CODANG | NUME             | SAL_BAZA | SAL_INDEXAT |
|--------|------------------|----------|-------------|
| 100    | Ion Ion          | 5000000  | 5250000     |
| 200    | Popescu Ion      | 6500000  | 6500000     |
| 300    | Ionescu Gheorghe | 15000000 | 16500000    |

16) Să se afișeze suma salariilor de bază

```
SQL> SELECT 'Suma este' || sum (salbaza) SUMA
FROM DatePers dp, DateSal ds
WHERE dp.codang=ds.codang(+);
```

SUMA

Suma este 26500000

17) Să se afișeze numele fiecărui angajat și codul șefului direct superior

```
SQL> SELECT nume || 'lucreaza pentru' || codsef ANGAJAT_SEF
FROM DatePers dp, DateSal ds
WHERE dp.codang=ds.codang;
```

ANGAJAT\_SEF

|                  |                 |     |
|------------------|-----------------|-----|
| Ion Ion          | lucreaza pentru | 300 |
| Popescu Ion      | lucreaza pentru | 300 |
| Ionescu Gheorghe | lucreaza pentru | -   |

18) Să se afișeze salariu de bază mediu, salariu minim și salariu maxim pentru toți salariații cu codul cuprins între 10 și 1000.

```
SQL> SELECT Avg (salbaza) MEDIU,
Min (salbaza) MINIM,
Max (salbaza) MAXIM
FROM DateSal
WHERE codang BETWEEN 10 AND 1000;
```

MEDIU MINIM MAXIM

7875000 3500000 16500000

19) Să se afișeze toți angajații cu funcția de Director, din localitatea Mangalia și cu un salariu mai mare de 14000000.

```
SQL> SELECT nume, functia, salbaza
```



```

FROM DatePers dp, DateSal ds
WHERE
    dp.codang=ds.codang AND
    functia= 'Director' AND
    dp.localitate='Mangalia' AND
    salbaza>14000000;

```

| NUME             | FUNCTIA  | SALBAZA  |
|------------------|----------|----------|
| Ionescu Gheorghe | Director | 16500000 |

20) Să se afișeze toți angajații din structura ierarhică a societății. Rădăcina arborelui este Directorul .

```

SQL> SELECT LPAD (' ',5*(LEVEL-1)) || codang, functia
FROM DateSal ds
START WITH functia='Director'
CONNECT BY PRIOR codang=codsef;

```

Rezultatul este:

| LPAD(' ',5*(LEVEL-1))  CODANG | FUNCTIA   |
|-------------------------------|-----------|
| 300                           | Director  |
| 100                           | Ec        |
| 200                           | Inginer   |
| 400                           | Tehnician |

21) Să se blocheze rândurile selectate de o cerere

```

SQL> SELECT * FROM Impozitar
FOR UPDATE

```

Tabelă blocată pentru update-area tuplurilor:

| LINIE | DELA     | PANALA     | SUMA    | PROCENT |
|-------|----------|------------|---------|---------|
| 1     |          | 0          | 210000  | 18      |
| 2     | 2100001  | 5200000    | 378000  | 23      |
| 3     | 5200001  | 8300000    | 1091000 | 28      |
| 4     | 8300001  | 11600000   | 1959000 | 34      |
| 5     | 11600001 | 9999999999 | 3081000 | 40      |

22) Să se adauge un nou angajat în tabela DatePers și să se selecteze angajatul adăugat după prima literă din nume și după apartenența sa o localitate.

```
SQL> INSERT INTO DatePers VALUES
(400, 'Popa Vasile', '1234567890400', '10-APR-1980', 'Zorelelor 12'
,'Medgidia', '0721333333');
SELECT * from DatePers
WHERE nume LIKE 'P%'
AND localitate IN ('Mangalia', 'Medgidia');
```

| CODANGNUME | CNP         | DataN         | ADRESA    | LOCALITATE  | TELEFON  |            |
|------------|-------------|---------------|-----------|-------------|----------|------------|
| 400        | Popa Vasile | 1234567890400 | 10-APR-80 | Zorelelor12 | Medgidia | 0721333333 |

23) Să se adauge datele salariale pentru angajatul nou introdus. Să se selecteze codul, numele și datele salariale introduse pentru noul angajat.

```
SQL> INSERT INTO DateSal
VALUES (400,'Tehnician',3500000,4,25,200);

SELECT ds.codang, dp.nume, ds.functia, ds.salbaza, ds.persintr,
ds.vechime, ds.codsef
FROM DatePers dp, DateSal ds
WHERE dp.codang=ds.codang
AND ds.codang=400
OR dp.nume = '%Vasile';
```

| CODANG | NUME        | FUNCTIA   | SALBAZA | PERSINTR | VECHIME | CODSEF |
|--------|-------------|-----------|---------|----------|---------|--------|
| 400    | Popa Vasile | Tehnician | 3500000 | 4        | 25      | 200    |

24) Să se adauge în tabela Pontaj datele pentru noul angajat (cu date introduse de la tastatura).

```
SQL> PROMPT Să se adauge în Tabela Pontaj datele pentru:
INSERT INTO Pontaj (codang, luna, zilelucr, orezi, zileco, zilecm,
orelucrate)
VALUES('&CodAngajat','&LunaPontaj','&ZileLucr','&OrePeZi',
'&ZileConOdihna','&ZileConMed',' &OreLucrEfectiv');
```

Să se adauge în Tabela Pontaj datele pentru:

```
Enter value for codangajat: 400
Enter value for lunapontaj: 1
Enter value for zilelucr: 22
Enter value for orepezi: 8
Enter value for zileconodihna: 1
Enter value for zileconmed: 1
Enter value for orelucrefectiv: 8
1 row created.
```

Ulterior de poate adăuga la linia de stare (o selecție explicită, prin introducerea codului corespunzător noul angajat inserat în tabelă) .

```
SQL> SELECT * FROM Pontaj
WHERE codang=&CodAngajat;
```

Enter value for codangajat: 400

| CODANG | LUNA | ZILELUCR | OREZI | ZILECO | ZILECM | ORELUCRATE |
|--------|------|----------|-------|--------|--------|------------|
| 400    | 1    | 22       | 8     | 1      | 1      | 8          |

SAU, o selecție implicită prin specificarea directă a codului angajatului:

```
SQL> SELECT * FROM Pontaj
WHERE codang= 400;
```

| CODANG | LUNA | ZILELUCR | OREZI | ZILECO | ZILECM | ORELUCRATE |
|--------|------|----------|-------|--------|--------|------------|
| 400    | 1    | 22       | 8     | 1      | 1      | 8          |

25) Să se adauge o noua taxă, în tabela TAXE, utilizând variabile de memorie

```
SQL> ACCEPT den PROMPT 'Denumire:'
ACCEPT procent PROMPT 'Procent:'
ACCEPT cotamax PROMPT 'Cota maxima:'
INSERT INTO Taxe VALUES('&den','&procent','&cotamax');
```

Denumire: TAXA NOUA  
Procent: 2  
Cota maxima: 3

Old 1: INSERT INTO Taxe VALUES ('&den','&procent','&cotamax')  
New 1: INSERT INTO Taxe VALUES ('TAXA NOUA','2','3')  
1 row created.

Ulterior, după rulare, se va putea selecta.

```
SQL> SELECT * FROM TAXE
WHERE den = '&Denumire ';
```

| DEN       | PROCENT | COTAMAX |
|-----------|---------|---------|
| TAXA NOUA | 2       | 3       |

26) Să se creeze o nouă tabelă pentru Datele Personale ale Angajaților din Constanța (DatePersCta) și să se adauge ulterior în această tabelă datele personale ale angajaților din Constanța existente în tabela inițială DatePers.

```
SQL> CREATE TABLE DatePersCta
(
    codang number(5) primary key,
    nume varchar2(35),
    cnp varchar2(13),
    datan date,
    adresa varchar2(30),
    localitate varchar2(15),
    telefon varchar2(10)
);

INSERT INTO DatePersCta
SELECT * FROM DatePers
WHERE localitate='Constanta';

COMMIT;
```

```
SELECT * FROM DatePersCta;
```

| CODANG | NUME        | CNP           | DataN       | ADRESA        | LOCALITATE | TELEFON    |
|--------|-------------|---------------|-------------|---------------|------------|------------|
| 100    | Ion Ion     | 1234567890100 | 10-JAN-1970 | Mangaliei 100 | Constanta  | 0722123456 |
| 200    | Popescu Ion | 1234567890200 | 10-FEB-1975 | Tomis 232     | Constanta  | 0744123456 |

27) Să se majoreze salariul directorului cu 10 procente.

```
SQL> UPDATE DateSal
SET salbaza=salbaza*1.1
WHERE functia='Director';
```

Rezultatul se poate vizualiza utilizând variabila “Functia”:

```
SQL> SELECT * FROM DateSal
WHERE functia='&Functia';
```

Enter value for functia: Director

| NRCRT | CODANG | FUNCTIA  | SALBAZA  | PERSINTR | VECHIME | CODSEF |
|-------|--------|----------|----------|----------|---------|--------|
| 3     | 300    | Director | 19965000 | 0        | 15      |        |

28) Să se șteargă toate înregistrările din DatePersCta unde numărul de telefon începe cu "0744..."

```
SQL> DELETE FROM DatePersCta  
WHERE telefon LIKE '0744%';
```

```
SELECT * FROM DatePersCta;
```

| CODANG | NUME    | CNP           | DataN       | ADRESA        | LOCALITATE | TELEFON    |
|--------|---------|---------------|-------------|---------------|------------|------------|
| 100    | Ion Ion | 1234567890100 | 10-JAN-1970 | Mangaliei 100 | Constanta  | 0722123456 |

29) Să se afișeze numele tabelelor create în schema proprie de obiecte

```
SQL> SELECT table_name from USER_TABLES;
```

```
TABLE_NAME  
-----  
DATEPERS  
DATEPERSCTA  
DATESAL  
DEDUCERI  
DEPT  
EMP  
IMPOZITAR  
PONTAJ  
SALGRADE  
TAXE
```

30) Să se adauge atributul TMP de tip NUMBER în tabela DatePersCta.

```
SQL> ALTER TABLE DatePersCta  
ADD ( TMP NUMBER (3) );
```

```
DESCRIBE DatePersCta;
```

| Name       | Null?    | Type         |
|------------|----------|--------------|
| CODANG     | NOT NULL | NUMBER(5)    |
| NUME       |          | VARCHAR2(35) |
| CNP        |          | VARCHAR2(13) |
| DATAN      |          | DATE         |
| ADRESA     |          | VARCHAR2(30) |
| LOCALITATE |          | VARCHAR2(15) |
| TELEFON    |          | VARCHAR2(10) |
| TMP        |          | NUMBER (3)   |

31) Să se modifice atributul TMP la o lungime de 5 poziții

```
SQL> ALTER TABLE DatePersCta
```

***MODIFY ( TMP NUMBER (5) );***

***DESCRIBE DatePersCta;***

| Name       | Null?    | Type         |
|------------|----------|--------------|
| CODANG     | NOT NULL | NUMBER(5)    |
| NUME       |          | VARCHAR2(35) |
| CNP        |          | VARCHAR2(13) |
| DATAN      |          | DATE         |
| ADRESA     |          | VARCHAR2(30) |
| LOCALITATE |          | VARCHAR2(15) |
| TELEFON    |          | VARCHAR2(10) |
| TMP        |          | NUMBER (5)   |

32) Să se redenumescă tabela DatePersCta în CONST

**SQL> ALTER TABLE DatePersCta  
RENAME TO Const;**

33) Să se șteargă tabela DatePersCta

***SQL> DROP TABLE DatePersCta;***

34) Să se adauge la DatePers restricția de Validare Codang>0.

***SQL> ALTER Table DatePers  
ADD (CONSTRAINT check\_comp CHECK (codang>0) );***

35) Să se creeze tabela virtuală CONSTANTA care va conține date despre angajații din Constanța

***SQL> CREATE VIEW Constanta  
AS SELECT \*  
FROM DatePers  
WHERE localitate='Constanta';***

View created.

36) Să se șteargă tabela virtuală CONSTANTA

***SQL> DROP VIEW Constanta;***

View dropped.

37) Să se afișeze numărul de înregistrări din tabela DatePers

```
SQL> SELECT count (*) NR_INREG  
        FROM DatePers;
```

| NR_INREG |
|----------|
|----------|

|   |
|---|
| 4 |
|---|

38) Să se vizualizeze restricțiile tablei DatePers

```
SQL> SELECT  
        CONSTRAINT_TYPE,  
        CONSTRAINT_NAME,  
        STATUS  
        FROM USER_CONSTRAINTS  
        WHERE TABLE_NAME= 'DATEPERS';
```

| C CONSTRAINT_NAME | STATUS  |
|-------------------|---------|
| C CHECK_COMP      | ENABLED |

## ***7.2. Aplicație informatică pentru activitatea de aprovizionare și desfacere a unei firme***

### **1. Crearea bazei de date.**

1) Să se creeze tabelele clienți, furnizori, produse, tranzacții, documente și proddoc.

***CREATE TABLE clienti***

```
(  
    codc      varchar2 (5),  
    denc      varchar2 (30),  
    adr        varchar2 (30),  
    loc        varchar2 (20),  
    cont       varchar2 (11),  
    banca      varchar2 (15),  
                constraint pk_codc primary key (codc)  
);
```

***CREATE TABLE furnizori***

```
(  
    codf      varchar2 (5),  
    denf      varchar2 (30),  
    adr        varchar2 (30),  
    loc        varchar2 (20),  
    cont       varchar2 (11),  
    banca      varchar2 (15),  
                constraint pk_codf primary key (codf)  
);
```

***CREATE TABLE produse***

```
(  
    codp      varchar2 (5),  
    denp      varchar2 (25),  
    um        varchar2 (5),  
    pret      number (10),  
    stoc      number (5),  
    termen    date,  
                constraint pk_codp primary key (codp)  
);
```



**CREATE TABLE tranzactii**

```
(  
    codt      varchar2 (5),  
    dent      varchar2 (1)  
        constraint nn_dent not null  
        constraint ck_dent check (upper (dent) in ('L','R')),  
    dataora   date default sysdate,  
    codf      varchar2 (5),  
    codc      varchar2 (5),  
        constraint pk_codt primary key (codt),  
        constraint fk_codf foreign key (codf) references furnizori (codf),  
        constraint fk_codc foreign key (codc) references clienti (codc)  
);
```

**CREATE TABLE documente**

```
(  
    codd      number (5)  
        constraint ck_codd check (codd>0),  
    dend      varchar2 (4)  
        constraint nn_dend not null  
        constraint ck_dend check (upper (dend) in  
('FACT','AVIZ','NIR','CHIT')),  
    data      date default sysdate,  
    codt      varchar2 (5),  
        constraint pk_codd primary key (codd),  
        constraint fk_codt foreign key (codt) references tranzactii (codt)  
);
```

**CREATE TABLE proddoc**

```
(  
    codd      number(5),  
    codp      varchar2(5),  
    um        varchar2(5),  
    cant      number(5),  
        constraint pk_coddp primary key (codd,codp)  
);
```

## **2. Modificarea structurii tabelelor bazei de date**

1) Să se modifice dimensiunea atributului CodP din tabela Produse, la 4 caractere.

**SQL> PROMPT** Modificati dimensiunea atributului Codp din tabela Produe la 4 caractere

**SQL> ALTER TABLE produse MODIFY (codp varchar2 (4));**

**SQL> DESCRIBE produse;**

| Name   | Null?    | Type          |
|--------|----------|---------------|
| CODP   | NOT NULL | VARCHAR2 (4)  |
| DENP   |          | VARCHAR2 (40) |
| UM     |          | VARCHAR2 (5)  |
| PRET   |          | NUMBER (13)   |
| STOC   |          | NUMBER (7)    |
| TERMEN |          | DATE          |

2) Adăugați atributul IE (number(2)) tablei ProdDoc

**SQL> PROMPT** Adaugati atributul IE (number (2) ) tablei proddoc

**SQL> ALTER TABLE proddoc ADD (IE NUMBER (2));**

**SQL> DESCRIBE proddoc;**

| Name | Null?    | Type         |
|------|----------|--------------|
| CODD | NOT NULL | NUMBER (5)   |
| CODP | NOT NULL | VARCHAR2 (5) |
| UM   |          | VARCHAR2 (5) |
| CANT |          | NUMBER (6)   |
| IE   |          | NUMBER (2)   |

3) Adăugați atributul Valoare, numeric de 20 caractere, la tabela Documente.

**SQL> PROMPT** Adaugati atributul valoare la tabela documente

**SQL> ALTER TABLE documente ADD (valoare number (20));**

**SQL> DESCRIBE documente;**

| Name    | Null?    | Type         |
|---------|----------|--------------|
| CODD    | NOT NULL | NUMBER (5)   |
| DEND    | NOT NULL | VARCHAR2 (4) |
| DATA    |          | DATE         |
| CODT    |          | VARCHAR2 (5) |
| VALOARE |          | NUMBER (20)  |

**3.Inserare înregistrări în tabele.**

**SQL> DELETE FROM clienti;**

**SQL> DELETE FROM furnizori;**

**SQL> DELETE FROM produse;**  
**SQL> DELETE FROM tranzactii;**  
**SQL> DELETE FROM documente;**  
**SQL> DELETE FROM proddoc;**

**SQL> PROMPT INSERARE ÎN TABELA CLIENTI;**  
**SQL> INSERT INTO clienti VALUES ('1','GOODS ','PIPERA**  
**135','BUCURESTI','A1234567890','BRD');**  
**SQL> INSERT INTO clienti VALUES ('2','DepozitPC','Stefan cel**  
**Mare 110','Bucuresti', 'A1231231234','BCR');**  
**SQL> INSERT INTO clienti VALUES ('3','Flamingo','Mihai**  
**Eminescu 18','Cluj','A1231231235','BCR');**  
**SQL> INSERT INTO clienti VALUES ('4','Ultra Pro','Mihai Bravu**  
**11','Timisoara','B1231231234','BRD');**  
**SQL> INSERT INTO clienti VALUES ('5','Flanco','Dorobantilor**  
**130','Cluj','C1231231234','BCR');**

**SQL> PROMPT INSERARE ÎN TABELA FURNIZORI;**  
**SQL> INSERT INTO furnizori VALUES ('1','GOODS ','PIPERA**  
**135','BUCURESTI','A1234567890','BRD');**  
**SQL> INSERT INTO furnizori VALUES ('2','ComputerNT','Gral**  
**Popescu 13','Iasi','A1234123412','BRD');**  
**SQL> INSERT INTO furnizori VALUES ('3','Python','Charles de**  
**Gaule 117','Cluj','A1234512345','BCR');**  
**SQL> INSERT INTO furnizori VALUES ('4','Blue Ridge','Magheru**  
**307','Bucuresti','B1234554321','BRD');**  
**SQL> INSERT INTO furnizori VALUES ('5','Deck**  
**Electronics','Lacul Alb 35','Iasi','B1234567777','BCR');**

**SQL> PROMPT INSERARE ÎN TABELA PRODUSE;**  
**SQL> INSERT INTO produse VALUES('P1','Monitor**  
**7inch','buc',3500000,1000,**  
**TO\_DATE('01/08/2006','DD/MM/YYYY'));**  
**SQL> INSERT INTO produse VALUES('P2','CD-RW ASUS**  
**24x10x40x','buc',1000000,500,**  
**TO\_DATE('01/08/2005','DD/MM/YYYY'));**  
**SQL> INSERT INTO produse VALUES('P3','Tastatura**  
**qwerty','buc',300000,100,**  
**TO\_DATE('01/06/2004','DD/MM/YYYY'));**

**SQL> INSERT INTO produse VALUES('P4','CPU AMD Athlon  
1.4GHz','buc',2700000,700,  
TO\_DATE('01/12/2004','DD/MM/YYYY'));**

**SQL> INSERT INTO produse VALUES('P5','Mouse  
A4TECH','buc',100000,150,  
TO\_DATE('01/06/2004','DD/MM/YYYY'));**

**SQL> PROMPT INSERARE ÎN TABELA TRANZACȚII;**

**SQL> INSERT INTO tranzactii VALUES  
( 'T1','R',TO\_DATE('01/08/2003 02:12:39','MM/DD/YYYY  
HH:MI:SS'),'3','1');**

**SQL> INSERT INTO tranzactii VALUES  
( 'T2','R',TO\_DATE('11/10/2003 10:20:09','MM/DD/YYYY  
HH:MI:SS'),'4','1');**

**SQL> INSERT INTO tranzactii VALUES  
( 'T3','L',TO\_DATE('12/10/2003 12:12:30','MM/DD/YYYY  
HH:MI:SS'),'1','5');**

**SQL> INSERT INTO tranzactii VALUES  
( 'T4','L',TO\_DATE('02/11/2003 04:55:39','MM/DD/YYYY  
HH:MI:SS'),'1','2');**

**SQL> PROMPT INSERARE ÎN TABELA DOCUMENTE;**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(10123,'FACT',TO\_DATE('01/08/2003','MM/DD/YYYY'),'T1');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(20123,'NIR',TO\_DATE('01/08/2003','MM/DD/YYYY'),'T1');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(10124,'FACT',TO\_DATE('11/10/2003','MM/DD/YYYY'),'T2');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(20124,'NIR',TO\_DATE('11/10/2003','MM/DD/YYYY'),'T2');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(30122,'AVIZ',TO\_DATE('12/10/2003','MM/DD/YYYY'),'T3');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(10125,'FACT',TO\_DATE('12/10/2003','MM/DD/YYYY'),'T3');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(30123,'AVIZ',TO\_DATE('02/11/2003','MM/DD/YYYY'),'T4');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(10126,'FACT',TO\_DATE('02/11/2003','MM/DD/YYYY'),'T4');**

**SQL> INSERT INTO documente (codd,dend,data,codt) VALUES  
(40123,'CHIT',TO\_DATE('02/11/2003','MM/DD/YYYY'),'T4');**

Valorile pentru câmpul valoare din tabela Documente nu au fost direct introduse în tabelă, deoarece acest câmp este unul calculat, iar valorile sale se vor introduce printr-o formulă.

```
SQL> PROMPT INSERARE ÎN TABELA PRODDOC;  
SQL> INSERT INTO proddoc VALUES (10123,'P1','buc',500,null);  
SQL> INSERT INTO proddoc VALUES (10123,'P2','buc',500,null);  
SQL> INSERT INTO proddoc VALUES (20123,'P1','buc',500,null);  
SQL> INSERT INTO proddoc VALUES (20123,'P2','buc',500,null);  
SQL> INSERT INTO proddoc VALUES (10124,'P3','buc',100,null);  
SQL> INSERT INTO proddoc VALUES (10124,'P4','buc',500,null);  
SQL> INSERT INTO proddoc VALUES (10124,'P5','buc',100,null);  
SQL> INSERT INTO proddoc VALUES (20124,'P3','buc',100,null);  
SQL> INSERT INTO proddoc VALUES (20124,'P4','buc',450,null);  
SQL> INSERT INTO proddoc VALUES (20124,'P5','buc',100,null);  
SQL> INSERT INTO proddoc VALUES (30122,'P1','buc',100,null);  
SQL> INSERT INTO proddoc VALUES (30122,'P2','buc',200,null);  
SQL> INSERT INTO proddoc VALUES (10125,'P1','buc',100,null);  
SQL> INSERT INTO proddoc VALUES (10125,'P2','buc',200,null);  
SQL> INSERT INTO proddoc VALUES (30123,'P1','buc',300,null);  
SQL> INSERT INTO proddoc VALUES (30123,'P4','buc',500,null);  
SQL> INSERT INTO proddoc VALUES (10126,'P1','buc',300,null);  
SQL> INSERT INTO proddoc VALUES (10126,'P4','buc',500,null);
```

#### **4. Definirea generatorului de numere de secvență:**

1) Să se creeze o secvență SECV care începe cu valoarea 10127 și se termina cu valoarea 10130 și pasul 1. Această secvență secv se va folosi ulterior pentru generarea automată de numere unice pentru câmpul codd din tabela Documente. Se vor genera succesiv, crescător, numerele cuprinse între 10127 și 10130.

```
SQL>CREATE SEQUENCE secv // nume secvență  
INCREMENT BY 1 // pasul de incrementare  
START WITH 10127 // valoarea de pornire a secvenței  
MAXVALUE 10130 // valoarea maximă a secvenței  
NOCACHE NOCYCLE; // secvență finită
```

2) Să se adauge o nouă valoare pentru atributul cheii primare codd din tabela Documente folosindu-se succesiunea generată de secvența SECV anterior creată.

```
SQL> INSERT INTO documente VALUES (secv.nextval, 'FACT', sysdate-2, 'T5', null);
```

```
SQL> SELECT * from documente;
```

| CODD  | DEND | DATA      | CODT | VALOARE |
|-------|------|-----------|------|---------|
| 10123 | FACT | 08-JAN-05 | T1   |         |
| 20123 | NIR  | 08-JAN-05 | T1   |         |
| 10124 | FACT | 10-NOV-05 | T2   |         |
| 20124 | NIR  | 10-NOV-05 | T2   |         |
| 30122 | AVIZ | 10-DEC-05 | T3   |         |
| 10125 | FACT | 10-DEC-05 | T3   |         |
| 30123 | AVIZ | 11-FEB-05 | T4   |         |
| 10126 | FACT | 11-FEB-05 | T4   |         |
| 40123 | CHIT | 11-FEB-05 | T4   |         |
| 10127 | FACT | 26-FEB-06 | T5   |         |

La execuție se observă adăugarea tuplului 10127-FACT-26FEB04-T5, cheia primară astfel definită, pentru câmpul codd, fiind prima valoare a secvenței SECV. La fiecare apelare a cuplului INSERT-SELECT secvența SECV anterior creată va incrementa automat cheia primară Codd din tabela Documente.

3) Să se adauge înregistrările corespunzătoare pentru o recepția a 100 de bucăți din produsul P3 și alte 200 de bucăți din produsul P4 de la furnizorul 4, știindu-se că factura a fost emisă de furnizor cu 2 zile înainte de recepția produselor.

```
SQL> INSERT INTO tranzactii VALUES ('T5','R', sysdate, '4','1');
```

```
SQL> INSERT INTO documente VALUES (secv.nextval, 'FACT', sysdate-2, 'T5',0);
```

```
SQL> INSERT INTO documente VALUES (20125,'NIR', sysdate,'T5',0);
```

```
SQL> INSERT INTO proddoc VALUES (secv.currval,'P3','buc',100,0);*
```

```
SQL> INSERT INTO proddoc VALUES (secv.currval,'P4','buc',200,0);*
```

```
SQL> INSERT INTO proddoc VALUES (20125,'P3','buc',100,1);
```

```
SQL> INSERT INTO proddoc VALUES (20125,'P4','buc',200,1);
```

Pentru a se putea defini cerința ca factura să fie emisă cu două zile înainte de recepția produselor, s-a optat pentru varianta sysdate-2 (data curentă-două zile), întrucât recepția produselor intrate în gestiune se face în ziua curentă de lucru. Cele două tupluri sunt cele care definesc aprovizionarea produselor P3 și P4, având drept valori, pentru unul din cele două atribute ale cheii primare, codd - numărul de secvență curent (secv.currval) definit anterior și preluat de la generatorul secv.nextval (din înregistrarea: secv.nextval - FACT- sysdate-2, T5, 0 )

Adăugările la stoc ale celor două produse se vor regăsi în tabela Proddoc.

Înainte de inserare:

**SQL> select \* from documente;**

| CODD  | DEND | DATA      | CODT |
|-------|------|-----------|------|
| 20123 | NIR  | 08-JAN-05 | T1   |
| 10124 | FACT | 10-NOV-05 | T2   |
| 20124 | NIR  | 10-NOV-05 | T2   |
| 30122 | AVIZ | 10-DEC-05 | T3   |
| 10125 | FACT | 10-DEC-05 | T3   |
| 30123 | AVIZ | 11-FEB-05 | T4   |
| 10126 | FACT | 11-FEB-05 | T4   |
| 40123 | CHIT | 11-FEB-05 | T4   |
| 10123 | FACT | 08-JAN-05 | T1   |

După inserare:

**SQL> select \* from documente;**

| CODD  | DEND | DATA      | CODT | VAL |
|-------|------|-----------|------|-----|
| 20123 | NIR  | 08-JAN-05 | T1   |     |
| 10124 | FACT | 10-NOV-05 | T2   |     |
| 20124 | NIR  | 10-NOV-05 | T2   |     |
| 30122 | AVIZ | 10-DEC-05 | T3   |     |
| 10125 | FACT | 10-DEC-05 | T3   |     |
| 30123 | AVIZ | 11-FEB-05 | T4   |     |
| 10126 | FACT | 11-FEB-05 | T4   |     |
| 40123 | CHIT | 11-FEB-05 | T4   |     |
| 10123 | FACT | 08-JAN-05 | T1   |     |
| 10127 | FACT | 27-FEB-06 | T5   | 0   |
| 20125 | NIR  | 29-FEB-06 | T5   | 0   |

Câmpurile adăugate în cele două tabele și având valoarea cheii primare generată ca fiind 10127 (primul număr din secvența SECV) sunt cele corespunzătoare instrucțiunilor SECV.NEXTVAL și SECV.CURRVAL, care au generat, respectiv, preluat valorile pentru cheia primară.

4) Să se afișeze ultimul număr utilizat din secvența SECV:

```
SQL> SELECT secv.CURRVAL  
FROM DUAL;
```

```
CURRVAL  
-----  
10127
```

5) Să se modifice pasul secvenței SECV de la 1 la 10000

```
SQL> ALTER SEQUENCE secv  
INCREMENT BY 10000;  
Sequence altered.
```

6) Să se ștergă secvența SECV .

```
SQL> DROP SEQUENCE secv;  
Sequence dropped.
```

## 5. Actualizări la nivelul aplicației:

1) Să se insereze în atributul IE din tabela Proddoc, valorile: “1”, pentru NIR (I) ; „-1”, pentru AVIZE (E); “0”, pentru celelalte documente.

```
SQL> UPDATE proddoc SET IE=-1  
WHERE SUBSTR (TO_CHAR (codd), 1, 1)= ' 3 ';  
SQL> UPDATE proddoc SET IE=1  
WHERE SUBSTR (TO_CHAR (codd), 1, 1)= ' 2 ';  
SQL> UPDATE proddoc SET IE=0  
WHERE SUBSTR (TO_CHAR (codd), 1, 1) NOT IN( ' 2 ', ' 3 ');  
SQL> SELECT * FROM proddoc;
```

| <b>CODD</b> | <b>CODP</b> | <b>UM</b> | <b>CANT</b> | <b>IE</b> |
|-------------|-------------|-----------|-------------|-----------|
| 10123       | P2          | buc       | 500         | 0         |
| 20123       | P1          | buc       | 500         | 1         |
| 10123       | P1          | buc       | 500         | 0         |
| 20123       | P2          | buc       | 500         | 1         |
| 30123       | P1          | buc       | 300         | -1        |
| 30123       | P4          | buc       | 500         | -1        |
| 10126       | P1          | buc       | 300         | 0         |
| 10126       | P4          | buc       | 500         | 0         |
| 10127       | P3          | buc       | 100         | 0         |
| 10127       | P4          | buc       | 200         | 0         |
| 20125       | P3          | buc       | 100         | 1         |
| 20125       | P4          | buc       | 200         | 1         |



2) Să se calculeze și să se afișeze valoarea totală pentru fiecare document, din tabela Documente.

```
SQL> UPDATE documente D SET
      valoare =
      (SELECT SUM (cant*pret) valoare
       FROM proddoc PD, produse P
       WHERE P.codp=PD.codp AND D.codd=PD.codd
       GROUP BY D.codd);
SQL> SELECT codd, valoare
      FROM documente
      WHERE valoare IS NOT NULL;
```

| CODD  | VALOARE   |
|-------|-----------|
| 20123 | 2.250E+09 |
| 10124 | 1.390E+09 |
| 20124 | 1.255E+09 |
| 30122 | 550000000 |
| 10125 | 550000000 |
| 30123 | 2.400E+09 |
| 10126 | 2.400E+09 |
| 10127 | 570000000 |
| 20125 | 570000000 |
| 10123 | 2.250E+09 |

3) Să se diminueze stocul aferent produsului P5 cu 50 de bucăți.

```
SQL> UPDATE produse
      SET stoc= stoc - 50
      WHERE codp= 'P5';
SQL> SELECT codp, denp, stoc from produse ;
```

| CODP | DENP         | STOC |
|------|--------------|------|
| P5   | Mouse A4TECH | 150  |

## 6. Funcțiile pentru șiruri de caractere:

1) Să se selecteze numele și localitatea unde își au sediul clienții, folosind formatul de afișare cu prima literă majusculă.

```
SQL> SELECT
      INITCAP (DENC) LITERA_MARE_NUME,
      INITCAP (LOC)
```

***FROM clienti;***

| LITERA_MARE_NUME          | INITCAP (LOC) |
|---------------------------|---------------|
| Interconn                 | Bucuresti     |
| Depozitul De Calculatoare | Bucuresti     |
| Flamingo                  | Cluj          |
| Ultra Pro                 | Timisoara     |
| Flanco                    | Cluj          |

2) Să se concateneze șirurile corespunzătoare atributelor „Adresa” și „Localitate” din tabela furnizori, pentru furnizorul “Blue Ridge” .

***SQL> SELECT denf, CONCAT (adr, loc) "Adesa\_din\_Localitatea"***  
***FROM Furnizori***  
***WHERE denf= 'Blue Ridge ';***

| DENF       | Adresa _ din _ Localitatea |
|------------|----------------------------|
| Blue Ridge | Magheru 307 Bucuresti      |

3) Să se selecteze toți furnizorii, aducând codd, denf și loc la lungimea de 20 de caractere fiecare, utilizând LPAD și RPAD.

***SQL> SELECT LPAD (codd, 20, ' \* '),***  
***LPAD (denf, 20),***  
***LPAD (loc, 20, '-')***  
***FROM furnizori ;***

| LPAD(CODF,20,'*') | LPAD(DENF,20)    | LPAD(LOC,20,'-') |
|-------------------|------------------|------------------|
| *****1            | INTERCONN        | -----Bucuresti   |
| *****2            | Computer Network | -----Iasi        |
| *****3            | Python           | -----Cluj        |
| *****4            | Blue Ridge       | -----Bucuresti   |
| *****5            | Deck Electronics | -----Iasi        |

***SQL> SELECT RPAD (codd, 20, ' \* '),***  
***RPAD (denf, 20),***  
***RPAD (loc, 20, '-')***  
***FROM furnizori ;***

| RPAD(CODF,20,'*') | RPAD(DENF,20)    | RPAD(LOC,20,'-') |
|-------------------|------------------|------------------|
| 1 * * * * *       | INTERCONN        | BUCURESTI----    |
| 2 * * * * *       | Computer Network | Iasi-----        |
| 3 * * * * *       | Python           | Cluj-----        |
| 4 * * * * *       | Blue Ridge       | Bucuresti-----   |
| 5 * * * * *       | Deck Electronics | Iasi-----        |

4) Să se afișeze furnizorii din alte localități decât București.

```
SQL> SELECT codf, denf, loc FROM furnizori  
WHERE UPPER (loc) <> 'BUCURESTI' ;
```

| <b>CODF</b> | <b>DENF</b>      | <b>LOC</b> |
|-------------|------------------|------------|
| 2           | Computer Network | Iasi       |
| 3           | Python           | Cluj       |
| 5           | Deck Electronics | Iasi       |

5) Să se afișeze clienții a căror denumire începe cu litera "F"

```
SQL> SELECT codc, denc FROM clienti  
WHERE SUBSTR (denc,1,1)= 'F';
```

| <b>CODC</b> | <b>DENC</b> |
|-------------|-------------|
| 3           | Flamingo    |
| 5           | Flanco      |

## 9. Funcțiile de dată

1) Să se afișeze denumirea furnizorilor cu care nu s-au mai încheiat tranzacții în ultimele 6 luni.

```
SQL> SELECT codf, denf FROM furnizori  
WHERE codf NOT IN  
(  
    SELECT codf FROM tranzactii  
    WHERE MONTHS_BETWEEN (sysdate,dataora)<=6  
);
```

| <b>CODF</b> | <b>DENF</b>      |
|-------------|------------------|
| 2           | Computer Network |
| 3           | Python           |
| 5           | Deck Electronics |

2) Să se afișeze perioada (lunile) de garanție rămasă până la expirarea produselor (inventariate în tabela Produse) cu enumerarea doar a celor care mai au ca valabilitate minimum 3 luni.

```
SQL> SELECT codp, denp,
```

***MONTHS\_BETWEEN (termen, sysdate) LUNI\_GARANTIE  
FROM produse  
where MONTHS\_BETWEEN (termen, sysdate) >3;***

| CODP | DENP                  | LUNI_GARANTIE |
|------|-----------------------|---------------|
| P1   | Monitor 17inch        | 29.0727       |
| P2   | CD-RW ASUS 24x10x40x  | 17.0727       |
| P3   | Tastatura qwerty      | 3.0727001     |
| P4   | CPU AMD Athlon 1.4GHz | 9.0727001     |
| P5   | Mouse A4TECH          | 3.0727001     |

3) Să se selecteze produsul cu termenul de garanție cel mai îndepărtat (August 2007) și să se evidențieze lunile de garanție rămase de la data curentă la termen.

***SQL> SELECT codp, denp,  
MONTHS\_BETWEEN ('01-Aug-06', sysdate)  
LUNI\_MAXIME\_GARANTIE  
FROM produse  
WHERE termen='01-Aug-07' ;***

| CODP | DENP            | LUNI_MAXIME_GARANTIE |
|------|-----------------|----------------------|
| P1   | Monitor 17 inch | 29.072489            |

4) Să se afișeze, codul, denumirea, termenul de garanție, precum și data decalată cu trei luni față de termenul de garanție și data anetrioară cu trei luni termenului de garanție. Se vor evidenția produsele ale căror termene de valabilitate nu au expirat.

***SQL> SELECT codp, denp, termen,  
ADD\_MONTHS (termen, 3) PESTE\_TREI\_L,  
ADD\_MONTHS (termen, -3) CU\_TREI\_L\_IN\_URMA  
FROM produse  
WHERE termen>sysdate;***

| CODP | DENP           | TERMEN     | PESTE_TREI_L | CU_TREI_L_ÎN_URMĂ |
|------|----------------|------------|--------------|-------------------|
| P1   | Monitor 17inch | 01-AUG-07  | 01-NOV-07    | 01-MAY-07         |
| P2   | CD-RW AS       | 01-AUG-06  | 01-NOV-06    | 01-MAY-06         |
| P3   | Tastatura      | 01- JUN-05 | 01- SEP- 05  | 01-MAR-05         |
| P4   | CPU AMD 1.4GHz | 01- DEC-05 | 01-MAR-06    | 01-SEP- 05        |
| P5   | Mouse A4TECH   | 01- JUN-05 | 01- SEP- 05  | 01-MAR-05         |

5) Să se afișeze data următoarei zile a săptămânii (*char*) după o dată declarată.

```
SQL> SELECT NEXT_DAY ('01-MAR-05', 1)
      FROM dual;
```

```
NEXT_DAY
-----
06-MAR-05
```

```
SQL> SELECT NEXT_DAY ('01-MAR-05', 2)
      FROM dual;
```

```
NEXT_DAY
-----
07-MAR-05
```

5) Să se afișeze ultima zi a lunii (*char*) după o dată declarată.

```
SQL> SELECT LAST_DAY ('01-jun-05')
      FROM dual;
```

```
LAST_DAY
-----
30-JUN-05
```

```
SQL> SELECT codp, denp, termen,
      LAST_DAY (termen) ULTIMA_ZI_LUNA
      FROM produse;
```

| CODP DENP |                       | TERMEN     | ULTIMA_ZI_LUNA |
|-----------|-----------------------|------------|----------------|
| P1        | Monitor 17inch        | 01-AUG-07  | 31- AUG-07     |
| P2        | CD-RW ASUS 24x10x40x  | 01-AUG-06  | 31 -AUG-06     |
| P3        | Tastatura qwerty      | 01 -JUN-05 | 30 - JUN-05    |
| P4        | CPU AMD Athlon 1.4GHz | 01 -DEC-05 | 31 - DEC-05    |
| P5        | Mouse A4TECH          | 01 -JUN-05 | 30 -JUN-05.    |

Funcția *ROUND* poate fi aplicată pe date calendaristice. *Round (data1)* întoarce *data1* cu timpul setat la 12:00AM (noaptea). Aceasta este folosită atunci când se compară date care au timpuri diferite.

***ROUND (data1, 'MONTH')*** întoarce:

- prima zi a lunii conținând *data1*, dacă *data1* este în prima parte a lunii,
- prima zi a următoarei luni, dacă *data1* este în a doua jumătate a lunii

- *ROUND(data1, 'YEAR')* întoarce:
- prima zi a anului conținând *data1*, dacă *data1* este în prima jumătate a anului,
- prima zi a următorului an, dacă *data1* este în a doua jumătate a lunii

De exemplu:

6) Să se folosească funcția *ROUND* pentru a returna prima zi a lunii sau anului sau prima zi a următoarei luni sau an, în funcție de data declarată.

```
SQL> SELECT SYSDATE,
ROUND (SYSDATE, 'MONTH') LUNA_ROTUNJITA,
ROUND (SYSDATE, 'YEAR') ANUL_ROTUNJIT
FROM DUAL;
```

| <b>SYSDATE</b>   | <b>LUNA_ROTUNJITA</b> | <b>ANUL_ROTUNJIT</b> |
|------------------|-----------------------|----------------------|
| <b>02-SEP-05</b> | <b>01-SEP-05</b>      | <b>01-JAN-06</b>     |

7) Analog, să se identifice rezultatele întoarse de funcția *LAST\_DAY* pentru valorile atributului *TERMEN* din tabela *Produse*:

```
SQL> SELECT codp, denp, termen,
LAST_DAY (termen) ULTIMA_ZI_LUNA
FROM produse;
```

| <b>CODP</b> | <b>DENP</b>                  | <b>TERMEN</b>    | <b>ULTIMA_ZI_LUNA</b> |
|-------------|------------------------------|------------------|-----------------------|
| <b>P1</b>   | <b>Monitor 17inch</b>        | <b>01-AUG-07</b> | <b>31-AUG-07</b>      |
| <b>P2</b>   | <b>CD-RW ASUS 24x10x40x</b>  | <b>01-AUG-06</b> | <b>31-AUG-06</b>      |
| <b>P3</b>   | <b>Tastatura qwerty</b>      | <b>01-JUN-05</b> | <b>30-JUN-05</b>      |
| <b>P4</b>   | <b>CPU AMD Athlon 1.4GHz</b> | <b>01-DEC-05</b> | <b>31-DEC-05</b>      |
| <b>P5</b>   | <b>Mouse A4TECH</b>          | <b>01-JUN-05</b> | <b>30-JUN-05</b>      |

8) Funcția *TRUNC(data1, 'char')* găsește prima zi a lunii care e conținută în *data1*, dacă *char = 'MONTH'* sau găsește prima zi a anului care conține *data1* dacă *char = 'YEAR'*. Să se utilizeze facilitățile acestei funcții.

```
SQL> SELECT SYSDATE DATA_CURENTA,
TRUNC (SYSDATE, 'MONTH') PRIMA_ZI_LUNA,
TRUNC (SYSDATE, 'YEAR') PRIMA_ZI_AN
FROM SYS.DUAL;
```

| DATA_CURENTA | PRIMA_ZI_LUNA | PRIMA_ZI_AN |
|--------------|---------------|-------------|
| 02-OCT-05    | 01-OCT-05     | 01-JAN-05   |

## 8. Funcții matematice:

1) Să se afișeze lungimea atributului Denumire client din tabela Clienți

```
SQL> SELECT denc,
          LENGTH(denc) LUNGIME_NUME
FROM clienti;
```

| DENC                      | LUNGIME_NUME |
|---------------------------|--------------|
| INTERCONN                 | 9            |
| Depozitul de calculatoare | 25           |
| Flamingo                  | 8            |
| Ultra Pro                 | 9            |
| Flanco                    | 6            |

2) Să se afișeze comisionul corespunzător vânzării fiecărui produs, în mii lei

```
SQL> ACCEPT comision PROMPT 'Introduceti comision: ';
SQL> SELECT denp, pret,
          &comision COMISION(%)
          pret*&comision/1000 VALOARE_COMISION
FROM produse;
```

|                       |    |
|-----------------------|----|
| Introduceti comision: | 10 |
|-----------------------|----|

  

| DENP                  | PRET    | COMISION (%) | VALOARE_COMISION |
|-----------------------|---------|--------------|------------------|
| Monitor 17inch        | 3500000 | 10           | 35000            |
| CD-RW ASUS 24x10x40x  | 1000000 | 10           | 10000            |
| Tastatura qwerty      | 300000  | 10           | 3000             |
| CPU AMD Athlon 1.4GHz | 2700000 | 10           | 27000            |
| Mouse A4TECH          | 100000  | 10           | 1000             |

3) Să se calculeze și afișeze stocul inițial pentru fiecare produs în parte.

```
SQL> SELECT P.codp,
          stoc STOC_INITIAL,
          SUM(stoc+IE*cant) STOC_CURENT
FROM produse P, proddoc PD
```

**WHERE** *P.codp = PD.codp*  
**GROUP BY** *P.codp, stoc;*

| CODP | STOC_INITIAL | STOC_CURENT |
|------|--------------|-------------|
| P1   | 1000         | 6100        |
| P2   | 500          | 2300        |
| P3   | 100          | 600         |
| P4   | 700          | 4350        |
| P5   | 100          | 300         |

4) Să se afișeze denumirea, prețul și stocul actual al produselor, sub forma: PRODUSUL <<nume>> ARE PRETUL UNITAR: <<pret>> LEI. STOCUL ACTUAL ESTE: <<stoc>> <<um>>

**SQL> SELECT 'PRODUSUL ' || LOWER (denp)  
 || 'ARE PRETUL UNITAR: ' ||pret||  
 'LEI. STOCUL ACTUAL ESTE: ' ||stoc||  
 'DE' || um  
 FROM produse;**

PRODUSUL Monitor 17inch ARE PRETUL UNITAR: 3500000 LEI. STOCUL ACTUAL ESTE: 1000 DE buc  
 PRODUSUL Cd-rw asus 24x10x40x ARE PRETUL UNITAR: 1000000 LEI. STOCUL ACTUAL ESTE: 500 DE buc  
 PRODUSUL Tastatura qwerty ARE PRETUL UNITAR: 300000 LEI. STOCUL ACTUAL ESTE: 100 DE buc  
 PRODUSUL CPU amd athlon 1.4ghz ARE PRETUL UNITAR: 2700000 LEI. STOCUL ACTUAL ESTE: 700 DE buc  
 PRODUSUL Mouse a4tech ARE PRETUL UNITAR: 100000 LEI. STOCUL ACTUAL ESTE: 100 DE buc

5) Să se afișeze codul produsului și prețul mărit cu 1.1 pentru Monitoare și cu 1.2 pentru Mouse.

**SQL> SELECT codp,  
 pret PRET\_INITIAL,  
 DECODE (denp, 'monitor 17inch', pret\*1.1,  
 'mouse A4TECH', pret\*1.2, pret) PRET\_MARIT  
 FROM produse;**

| CODP | PRET_INITIAL | PRET_MARIT |
|------|--------------|------------|
| P1   | 3500000      | 3850000    |
| P2   | 1000000      | 1000000    |
| P3   | 300000       | 300000     |
| P4   | 2700000      | 2700000    |
| P5   | 100000       | 120000     |

6) Să se afișeze Cantitatea Medie cumpărată din fiecare produs și să se ordoneze tuplurile după Cantitate.



```
SQL> SELECT P.codp, AVG (cant) CANT_MEDIE
      FROM produse P, proddoc PR
      WHERE P.codp= PR.codp AND IE =1
      GROUP BY P.codp
      ORDER BY AVG (cant);
```

| CODP | CANT_MEDIE |
|------|------------|
| P3   | 100        |
| P5   | 100        |
| P4   | 325        |
| P1   | 500        |
| P2   | 500        |

Ca algoritm de analiză al funcției DECODE se poate observa că, pentru coloana DENP (primul argument) are loc căutarea valorilor “monitor 17 inch” și ”mouse A4 Tech”, iar în cazul în care acestea sunt regăsite pe coloana denumirilor, prețurile lor sunt actualizate cu 1.1 și, respectiv, 1.2.

Pentru restul produselor care nu fac obiectul căutării, se trece implicit, ultimul argument, în cazul de față coloana PRET, sau se poate trece o expresie ‘Pret\_Nemodificat’

Fiind vorba de cumpărare, implicit se ia în calcul ca document de intrare NIR-ul (pentru aprovizionare), acest lucru necesitând o condiție suplimentară IE=1 (alături de cea care identifică din tabela Produse doar acele produse care au făcut obiectul tranzacției și au la bază un document justificativ).

7) Să se afișeze doar acele produse care au cantitatea minimă vândută mai mare decât cantitatea minimă a produsului P3.

```
SQL> SELECT codp,
      MIN (cant) CANT_MINIMA
      FROM proddoc WHERE IE= -1
      GROUP BY codp
      HAVING MIN (cant) >
      (SELECT MIN (cant)
      FROM proddoc
      WHERE codp='P3');
```

| CODP | CANT_MINIMA |
|------|-------------|
| P2   | 200         |
| P4   | 500         |

Fiind vorba de vânzare se pornește de la ideea că documentul justificativ aferent ieșirii din gestiune este avizul, ca atare, se va trece condiția IE=-1. Totodată, în această situație este vorba de o clauză select imbricată pentru a permite selecția doar a celor produse care respectă o condiție față de produsul P3. Ca și în cazul anterior nu se va trece în clauza select atributul *cant* după pentru care se calculează funcțiile și se face gruparea.

8) Să se afișeze cantitatea medie doar pentru produsele care apar mai mult de două ori în tabela Proddoc.

```
SQL> SELECT codp,  
      AVG (cant) CANT_MEDIE  
      FROM proddoc  
      GROUP BY codp  
      HAVING COUNT (*) > 2;
```

**CODP CANT\_MEDIE**

|    |           |
|----|-----------|
| P1 | 300       |
| P2 | 350       |
| P3 | 100       |
| P4 | 391.66667 |

9) Să se afișeze doar acele produse pentru care cantitatea este mai mare sau egală cu 200.

```
SQL> SELECT codp, MAX (cant) CANT_MAXIMA  
      FROM proddoc  
      HAVING MAX (cant) >= 200  
      GROUP BY codp;
```

**CODP CANT\_MAXIMA**

|    |     |
|----|-----|
| P1 | 500 |
| P2 | 500 |
| P4 | 500 |

10) Să se afișeze doar acele produse pentru care cantitatea medie este mai mare sau egală cu 200.

```
SQL> SELECT codp, AVG (cant) MEDIE  
      FROM proddoc
```

**GROUP BY codp  
HAVING AVG (cant) > 200;**

| CODP | MEDIE     |
|------|-----------|
| P1   | 300       |
| P2   | 350       |
| P4   | 391.66667 |

11) Să se afișeze cantitatea medie pe tip de produs, pentru toate codurile de produs mai puțin P1.

**SQL> SELECT codp, AVG (cant) MEDIE\_CANT  
FROM proddoc  
WHERE codp != 'P1'  
GROUP BY codp;**

| CODP | MEDIE_CANT |
|------|------------|
| P2   | 350        |
| P3   | 100        |
| P4   | 391.66667  |
| P5   | 100        |

12) Să se calculeze cantitatea medie pentru fiecare produs distinct, din tabela Proddoc.

**SQL> SELECT codp,  
AVG (cant) MEDIE  
FROM proddoc  
GROUP BY codp;**

| CODP | MEDIE     |
|------|-----------|
| P1   | 300       |
| P2   | 350       |
| P3   | 100       |
| P4   | 391.66667 |
| P5   | 100       |

13) Determinați prețul mediu pentru fiecare produs în afară de produsul 'Monitor 17inch' din tabela Produse.

**SQL> SELECT codp,  
AVG (pret) PRET\_MEDIU  
FROM produse  
WHERE denp!= 'Monitor 17inch'**

**GROUP BY codp;**

| <b>CODP</b> | <b>PRET_MEDIU</b> |
|-------------|-------------------|
| P2          | 1000000           |
| P3          | 300000            |
| P4          | 2700000           |
| P5          | 100000            |

14) Afișați prețul minim pe produs.

```
SQL> SELECT denp,
        MIN (pret) PRET_MINIM
        FROM produse
        GROUP BY denp;
```

| <b>DENP</b>           | <b>PRET_MINIM</b> |
|-----------------------|-------------------|
| CD-RW ASUS 24x10x40x  | 1000000           |
| CPU AMD Athlon 1.4GHz | 2700000           |
| Monitor 17inch        | 3500000           |
| Mouse A4TECH          | 100000            |
| Tastatura qwerty      | 300000            |

15) Să se afișeze toate produsele cu diferențe cantitative în documente.

```
SQL> SELECT a.codp
        FROM (
                SELECT p.codp, SUM (cant) cant
                FROM proddoc p,documente d
                WHERE p.codd=d.codd
                     AND dend='FACT'
                GROUP BY p.codp
            ) a,
        (SELECT p.codp, SUM(cant) cant
        FROM proddoc p,documente d
        WHERE p.codd=d.codd
             AND dend<>'FACT'
        GROUP BY p.codp
      ) b
  WHERE a.codp=b.codp
        AND a.cant-b.cant <> 0;
```

| <b>CODP</b> |
|-------------|
| P4          |

Se identifică cu documentele FACTURA care se regăsesc atât în nomenclatorul de documente (tabela Documente) cât și în nomenclatorul de documente “tranzacționate” (participante la tranzacțiile de produse, din tabela Proddoc). Apoi se identifică cu restul de documente (în afară de FACTURA) aflate atât în nomenclator, cât și în tranzacții. În final se trec condițiile de identificare a produselor tranzacționate și se stabilesc diferențele cantitative.

16) Să se afișeze denumirea, prețul și valoarea totală a vânzărilor pentru fiecare produs, ținând cont de comisionul de 5%.

```
SQL> SELECT denp, pret,  
      SUM(pret*cant*1.05) TOTAL_VANZARI  
FROM produse, proddoc  
WHERE produse.codp=proddoc.codp  
      AND IE= -1  
GROUP BY denp,pret;
```

| DENP                  | PRET    | TOTAL_VANZARI |
|-----------------------|---------|---------------|
| CD-RW ASUS 24x10x40x  | 1000000 | 210000000     |
| CPU AMD Athlon 1.4GHz | 2700000 | 1.418E+09     |
| Monitor 17inch        | 3500000 | 1.470E+09     |

S-au identificat doar produsele pentru care IE=-1, respectiv au ieșit din gestiune (au fost vândute) fiind însoțite de documentul AVIZ (de expediție).

17) Să se afișeze valoarea maximă, valoarea medie, valoarea minimă și valoarea totală pentru livrările (IE= -1) de produse efectuate.

```
SQL> SELECT MAX (1.05*cant*pret) VZ_MAX,  
      AVG (1.05*cant*pret) VZ_MED,  
      MIN (1.05*cant*pret) VZ_MIN,  
      SUM (1.05*cant*pret) VZ_TOTAL  
FROM produse, proddoc  
WHERE produse.codp=proddoc.codp  
      AND IE= -1  
GROUP BY produse.codp;
```

| VZ_MAX    | VZ_MED    | VZ_MIN    | VZ_TOTAL  |
|-----------|-----------|-----------|-----------|
| 1.103E+09 | 735000000 | 367500000 | 1.470E+09 |
| 210000000 | 210000000 | 210000000 | 210000000 |
| 1.418E+09 | 1.418E+09 | 1.418E+09 | 1.418E+09 |

18) Să se calculeze și afișeze profiturile rezultate din vânzări (IE= -1) cu comision de 5%

```
SQL> SELECT p.codp,
      pret*0.95      PROFIT
      FROM produse p, proddoc pd
      WHERE p.codp=pd.codp
      AND IE= -1;
```

| CODP | PROFIT  |
|------|---------|
| P1   | 3325000 |
| P2   | 950000  |
| P1   | 3325000 |
| P4   | 2565000 |

19) Să se afișeze tranzacțiile cu valoare mai mică decât cea mai mare valoare a unei tranzacții cu furnizorul 4

```
SQL> SELECT codt, valoare
      FROM documente
      WHERE valoare < ANY
      (
        SELECT valoare
        FROM documente d, tranzactii t
        WHERE d.codt=t.codt
        AND codf='4'
      );
```

| CODT | VALOARE   |
|------|-----------|
| T2   | 1.255E+09 |
| T3   | 550000000 |
| T3   | 550000000 |
| T5   | 570000000 |
| T5   | 570000000 |

20) Afișați produsele care au cantitatea mai mare decât cea mai mică cantitate a produsului “P4” ( $\min(cant)P4=200$ ).

```

SQL> SELECT codp, cant
      FROM proddoc
      WHERE
          Codp!= 'P4' AND cant > SOME
              (SELECT DISTINCT cant
               FROM proddoc
               WHERE codp='P4')

      ORDER BY cant DESC;

```

| CODP | CANT |
|------|------|
| P2   | 500  |
| P1   | 500  |
| P1   | 500  |
| P2   | 500  |
| P1   | 300  |
| P1   | 300  |

Cea mai mică cantitate a produsului 'P4' este de 200 bucăți, astfel că, cererea principală întoarce toate produsele, cu excepția lui 'P4' (specificată explicit) care sunt într-o cantitatea mai mare decât minimul cantității produsului 'P4' specificat.

Astfel, condiția '> ANY' înseamnă "mai mare ca minim" iar '=ANY' este echivalent cu operatorul *IN*.

Când se folosește SOME/ANY, DISTINCT este frecvent utilizat pentru a împiedica să se selecteze liniile de mai multe ori.

21) Să se afișeze produsele care au cantitatea mai mare sau egală cu cea mai mare cantitate a produsului "P4" ( $\max(cant)P4=5200$ ), inclusiv produsul 'P4'.

```

SQL> SELECT codp, cant
      FROM proddoc
      WHERE cant >= SOME
          (select MAX (cant)
           FROM proddoc
           WHERE codp='P4')

      ORDER BY cant DESC;

```

| CODP | CANT |
|------|------|
| P2   | 500  |
| P1   | 500  |
| P1   | 500  |
| P2   | 500  |
| P4   | 500  |

P4 500  
P4 500

22) Să se afișeze, pentru fiecare document în parte, ce procent reprezintă produsele din totalul de produse de pe document.

```
SQL> SELECT a.codd, a.codp, a.PROC/b.TOTAL*100 PROCENT
FROM
      (SELECT codd, codp, cant PROC
        FROM proddoc
        GROUP BY codd, codp,cant) a,
      (SELECT codd, SUM (cant) TOTAL
        FROM proddoc
        GROUP BY codd) b
WHERE a.codd=b.codd ;
```

| CODD  | CODP | PROCENT   |
|-------|------|-----------|
| 10123 | P1   | 50        |
| 10123 | P2   | 50        |
| 10124 | P3   | 14.285714 |
| 10124 | P4   | 71.428571 |
| 10124 | P5   | 14.285714 |
| 10125 | P1   | 33.333333 |
| 10125 | P2   | 66.666667 |
| 10126 | P1   | 37.5      |
| 10126 | P4   | 62.5      |
| 10127 | P3   | 33.333333 |
| 10127 | P4   | 66.666667 |
| 20123 | P1   | 50        |
| 20123 | P2   | 50        |
| 20124 | P3   | 15.384615 |
| 20124 | P4   | 69.230769 |
| 20124 | P5   | 15.384615 |
| 20125 | P3   | 33.333333 |
| 20125 | P4   | 66.666667 |
| 30122 | P1   | 33.333333 |
| 30122 | P2   | 66.666667 |
| 30123 | P1   | 37.5      |
| 30123 | P4   | 62.5      |

23) Să se afișeze documentele având valorile totale cuprinse între 1.500.000.000 și 6.500.000.000 sau cele ca sunt NIR-uri și Facturi.

```
SQL> SELECT * from documente
      WHERE valoare BETWEEN 1500000000 AND 6500000000
      OR (dend='NIR' AND dend='FACT')
      ORDER BY data ASC;
```



| CODD  | DEND | DATA      | CODT | VALOARE   |
|-------|------|-----------|------|-----------|
| 20123 | NIR  | 08-JAN-03 | T1   | 2.250E+09 |
| 10123 | FACT | 08-JAN-03 | T1   | 2.250E+09 |
| 30123 | AVIZ | 11-FEB-03 | T4   | 2.400E+09 |
| 10126 | FACT | 11-FEB-03 | T4   | 2.400E+09 |

24) Să se afișeze perioada de timp, în săptămâni rămase până la expirarea fiecărui produs. Săptămânile (cu perioadele interimare rezultate) se vor rotunji (prin funcțiile ROUND sau TRUNC ) la valorile întregi.

**SQL> SELECT termen,  
ROUND ((termen-sysdate)/7) SAPT\_GARANTIE  
FROM produse;**

| TERMEN    | SAPT_GARANTIE |
|-----------|---------------|
| 01-AUG-06 | 126           |
| 01-AUG-05 | 74            |
| 01-JUN-04 | 13            |
| 01-DEC-04 | 39            |
| 01-JUN-04 | 13            |

25) Să se afișeze denumirea și valoarea documentelor împreună cu data încheierii lor, doar pentru tranzacțiile încheiate în luna februarie 2005.

**SQL> SELECT dend, valoare, data DATA\_INCHEIERII  
FROM documente  
WHERE TO\_CHAR (data, 'MM/YY') = '02/05';**

| DEND | VALOARE   | DATA_INCHEIERII |
|------|-----------|-----------------|
| FACT | 570000000 | 27-FEB-05       |
| NIR  | 570000000 | 29-FEB-05       |

26) Să se creeze un index nou pe atributul denumire produs (Denp) din tabela Produse.

**SQL> CREATE INDEX prod\_idx  
ON produse (denp);**

Index created.

27) Să se afișeze indecșii creați pentru tabela Produse și dacă asigură unicitatea.

```
SQL> SELECT  
IC.index_name, IC.column_name,  
IC.column_position COL_POZ,  
IX.uniqueness  
FROM user_indexes IX, user_ind_columns IC  
WHERE IC.index_name=IX.index_name  
AND IC.table_name= 'PRODUSE ';
```

No rows selected.

28) Să se șteargă indexul creat anterior.

```
SQL> DROP INDEX prod_idx;
```

Index dropped.

29) Să se afișeze restricțiile definite pentru tabela Tranzacții.

```
SQL> SELECT CONSTRAINT_TYPE    TIP_RESTR,  
CONSTRAINT_NAME                NUME_RESTR,  
STATUS                          STAREA_RESTR  
FROM USER_CONSTRAINTS  
WHERE TABLE_NAME= 'TRANZACTII';
```

| TIP_RESTR | NUME_RESTR | STAREA_RESTR |
|-----------|------------|--------------|
| C         | NN_DENT    | ENABLED      |
| C         | CK_DENT    | ENABLED      |
| P         | PK_CODT    | ENABLED      |
| R         | FK_CODF    | ENABLED      |
| R         | FK_CODC    | ENABLED      |

30) Să se afișeze numele tabelor create în schema proprie de obiecte

```
SQL> SELECT table_name FROM USER_TABLES ;
```

```
TABLE_NAME  
-----  
BONUS  
CLIENTI  
DEPT  
DOCUMENTE  
EMP  
FURNIZORI  
PRODDOC  
PRODUSE
```

SALGRADE  
TRANZACTII  
10 rows selected.

31) Să se afișeze tipurile de obiecte create în schema proprie de obiecte

**SQL> SELECT DISTINCT OBJECT\_TYPE  
FROM USER\_OBJECTS;**

OBJECT\_TYPE  
-----  
INDEX  
SEQUENCE  
TABLE

32) Să se redenumescă tabela Clienți în tabela “Clienți\_Redenumiți”.

**SQL> ALTER TABLE clienti RENAME TO clienti\_redenumiti;**

Table altered.

**SQL> SELECT \* FROM Clienti\_Redenumiti;**

| CODC | DENC  | ADR              | LOC       | CONT        | BANCA |
|------|-------|------------------|-----------|-------------|-------|
| 1    | CONN  | PIPERA 135       | BUCURESTI | A1234567890 | BRD   |
| 5    | Flanx | Dorobantilor 130 | Cluj      | C1231231234 | BCR   |

33) Să se șteargă tabela Clienți\_Redenumiți și să se elibereze spațiul ocupat de aceasta.

**SQL> TRUNCATE TABLE clienti\_redenumiti;**

34) Să se creeze un sinonim public pentru tabela Produse din schema de obiecte Student.

**SQL> CREATE PUBLIC SYNONYM prod FOR student.produse;**

35) Să se creeze utilizatorul AGENT001 cu parola Agent.

**SQL> CREATE USER AGENT001  
IDENTIFIED BY agent;**

36) Să se creeze o serie de drepturi la nivel de sistem pentru utilizatorul AGENT001.

**SQL> GRANT CREATE TABLE,  
CREATE SEQUENCE,**

***CREATE VIEW  
TO AGENT001;***

37) Să se modifice parola utilizatorului AGENT001 cu “noua\_parola”

***SQL> ALTER USER AGENT001  
IDENTIFIED BY noua\_parola;***

38) Să se creeze rolul AgVanz cu drepturile RESOURCE si CONNECT la nivel de sistem.

***SQL> CREATE ROLE agvanz;  
SQL> SET ROLE AgvVanz;  
SQL> GRANT RESOURCE, CONNECT TO AgVanz;***

39) Să se atașeze rolul AgVanz utilizatorului AGENT001.

***SQL> GRANT AgVanz TO AGENT001;***

40) Să se anuleze drepturile primite de utilizatorul AGENT001 pe tabela Documente.

***SQL> REVOKE ALL ON documente FROM AGENT001;***

41) Să se creeze tabela partiționată Vânzari (codt, data, suma) cu partiții pentru vânzarile din ultimele 3 luni.

***SQL> CREATE TABLE  
vanzari (codt varchar2 (5), data date, suma number (11) )  
STORAGE (INITIAL 100K NEXT 50K) LOGGING  
PARTITION BY RANGE(data)  
(PARTITION LUNA03 VALUES LESS THAN (4)  
TABLESPACE T0,  
PARTITION LUNA02 VALUES LESS THAN (3)  
TABLESPACE T1,  
PARTITION LUNA01 VALUES LESS THAN (2)  
TABLESPACE T2);***

42) Să se adauge noi tupluri din tabela Tranzacții în partiția LUNA02 din tabela Vânzări.

```

SQL> INSERT INTO vanzari
PARTITION (LUNA02)
SELECT T.codt,
TO_CHAR (dataora,'MM-DD-YYYY'), valoare
FROM tranzactii T, documente D
WHERE T.codt=D.codt
AND MONTHS_BETWEEN (data, sysdate)=2;

```

43) Să se creeze un raport care să afișeze informațiile despre tranzacțiile încheiate în ultimul an, documentele și produsele aferente și un total general.

```

SQL> SET PAGESIZE 200
SQL> SET LINESIZE 100
SQL> SET FEEDBACK OFF
SQL> SET ECHO OFF
SQL> SET VERIFY OFF
SQL> COLUMN CODT FORMAT a5 HEADING 'Nr'
SQL> COLUMN DENT FORMAT a1 HEADING 'Tip'
SQL> COLUMN DATAORA FORMAT a10 HEADING 'Data'
SQL> COLUMN DEND FORMAT a4 HEADING 'Doc'
SQL> COLUMN CODD FORMAT 99999 HEADING 'Nr Doc'
SQL> COLUMN CODF FORMAT a5 HEADING 'Fz'
SQL> COLUMN CODC FORMAT a5 HEADING 'Cl'
SQL> COLUMN VALOARE FORMAT 99999999999
HEADING 'Valoare Tranzactie'
SQL> COLUMN CODP FORMAT a5 HEADING 'Produs'
COLUMN CANT FORMAT 99999 HEADING 'Cantitate'
SQL> SELECT T.codt, dent, dataora, dend,
D.codd, codf, codc, valoare, codp, cant
FROM tranzactii T,documente D, proddoc P
WHERE T.codt=D.codt
AND D.codd=P.codd
ORDER BY T.codt, Dcodd, codp

```

| Nr | T | Data      | Doc  | Nr Doc | Fz | Cl | Valoare    | Produs | Cantitate |
|----|---|-----------|------|--------|----|----|------------|--------|-----------|
| T1 | R | 08-JAN-04 | FACT | 10123  | 3  | 1  | 2250000000 | P1     | 500       |
| T1 | R | 08-JAN-04 | FACT | 10123  | 3  | 1  | 2250000000 | P2     | 500       |
| T1 | R | 08-JAN-04 | NIR  | 20123  | 3  | 1  | 2250000000 | P1     | 500       |
| T1 | R | 08-JAN-04 | NIR  | 20123  | 3  | 1  | 2250000000 | P2     | 500       |
| T2 | R | 10-NOV-04 | FACT | 10124  | 4  | 1  | 1390000000 | P3     | 100       |
| T2 | R | 10-NOV-04 | FACT | 10124  | 4  | 1  | 1390000000 | P4     | 500       |
| T2 | R | 10-NOV-04 | FACT | 10124  | 4  | 1  | 1390000000 | P5     | 100       |