

PROGRAMARE ORACLE 10g

**SOLUȚII INFORMATICE PENTRU DEZVOLTAREA
APLICAȚIILOR ECONOMICE UTILIZÂND
PL/SQL și ORACLE DEVELOPER**

www.editurauniversitara.ro

Lect. univ. dr. IONEL IACOB

PROGRAMARE ORACLE 10g

**SOLUȚII INFORMATICE PENTRU DEZVOLTAREA
APLICAȚIILOR ECONOMICE UTILIZÂND
PL/SQL și ORACLE DEVELOPER**

www.editurauniversitara.ro



EDITURA UNIVERSITARĂ
București, 2012

Redactor: Gheorghe Iovan
Tehnoredactor: Ionel Iacob
Coperta: Angelica Mălăescu

Editură recunoscută de Consiliul Național al Cercetării Științifice (C.N.C.S.)

Descrierea CIP a Bibliotecii Naționale a României

IACOB, IONEL

**Programare Oracle 10g : soluții informatice pentru dezvoltarea
aplicațiilor economice utilizând PL/SQL și Oracle Developer / Ionel Iacob. -**

București : Editura Universitară, 2011

Bibliogr.

ISBN 978-606-591-313-4

004.4

DOI: (Digital Object Identifier): 10.5682/9786065913134

© Toate drepturile asupra acestei lucrări sunt rezervate, nicio parte din această lucrare nu poate
fi copiată fără acordul Editurii Universitare

Copyright © 2012

Editura Universitară

Director: Vasile Muscalu

B-dul. N. Bălcescu nr. 27-33, Sector 1, București

Tel.: 021 – 315.32.47 / 319.67.27

www.editurauniversitara.ro

e-mail: redactia@editurauniversitara.ro

Distribuție: tel.: 021-315.32.47 / 319.67.27 / 0744 EDITOR / 07217 CARTE

comenzi@editurauniversitara.ro

O.P. 15, C.P. 35, București

www.editurauniversitara.ro

CUPRINS

PREFAȚĂ	9
CAPITOLUL I	
PROGRAMARE ORACLE UTILIZÂND FUNDAMENTE S.Q.L*PLUS AVANSAT PENTRU ANALIZĂ ECONOMICĂ	11
I.1 PROIECTAREA APLICAȚIEI INFORMATICE PRIVIND EVIDENȚA PRODUSELOR FABRICATE ȘI IMPLEMENTAREA ÎN SQL*PLUS	11
I.2 FUNCȚII ORACLE PENTRU ANALIZA ECONOMICĂ A REZULTATELOR	15
I.2.1 GRUPAREA LINIILOR TOTALIZATOARE - CLAUZA „GROUP BY”	15
I.2.2 STRUCTURAREA MULȚIMILOR REZULTAT (20)	19
CAPITOLUL II	
ELEMENTE DEFINITORII ALE PROGRAMĂRII S.Q.L.	47
II.1 PREZENTARE P.L./S.Q.L	47
II.2 FUNDAMENTE ALE LIMBAJULUI P.L./S.Q.L	48
II.2.1 STRUCTURA FUNCTIONALA	48
II.2.2 REGULI PENTRU APLICAREA INSTRUCȚIUNILOR	49
II.2.3 SIMBOLURI UTILIZATE	49
II.2.4 DOMENIUL DE VALABILITATE AL OBIECTELOR	50
II.2.5 ATRIBUIRI ȘI EXPRESII	51
II.2.6 PRECEDENȚA OPERATORILOR ÎN PL/SQL	51
II.2.7 FUNCȚII INTERNE	51
II.2.8 APELUL FUNCȚIILOR ȘI AL PROCEDURILOR	52
II.3 CURSOARE	52
II.3.1. DECLARAREA UNUI CURSOR ÎN P.L./S.Q.L	53
II.3.2. UTILIZAREA RÂNDURILOR DIN CURSOR	53
II.3.3. DESCHIDEREA ȘI ÎNCHIDEREA UNUI CURSOR	53
II.3.4 ATRIBUTE ALE CURSORULUI	54
II.3.5 CURSOARE CU PARAMETRII	55
II.3.6 STRUCTURI REPETITIVE ÎN CURSOR	56
II.4 INSTRUCȚIUNI DE SALT ȘI DE CONTROL	57
II.4.1 INSTRUCȚIUNEA „IF”	57
II.4.2 INSTRUCȚIUNEA „WHILE”	59
II.4.3 INSTRUCȚIUNEA „GOTO”	59
II.4.4 INSTRUCȚIUNEA „EXIT”	60
II.5 VARIABILE TIP ÎNREGISTRARE ÎN P.L./S.Q.L	61
II.6 ÎNREGISTRĂRI DEFINITE DE UTILIZATOR. INSTRUCȚIUNEA „TYPE”	63
II.7 DEFINIREA TABLOURILOR DE DATE	64
CAPITOLUL III	
APLICAȚIE INFORMATICĂ PENTRU EVIDENȚA ACTIVITĂȚII COMERCIALE DIN CADRUL UNEI SOCIETĂȚI ECONOMICE	66
III.1 CREAREA TABELELOR BAZEI DE DATE	66
III.2 INSERAREA DATELOR	67
III.3 APLICAȚII CURSOARE	69

CAPITOLUL IV	
APLICAȚIE INFORMATICĂ PENTRU EVIDENȚA PROCESULUI COMERCIAL DINTR-O SOCIETATE ECONOMICĂ	79
IV.1 STUDIUL SISTEMULUI INFORMAȚIONAL	79
IV.2 CREAREA TABELELOR APLICAȚIEI	80
IV.3 APLICAȚII INFORMATICE	84
CAPITOLUL V	
APLICAȚIE INFORMATICĂ PENTRU EVIDENȚA VÂNZĂRILOR DE CĂRȚI LA O BIBLIOTECĂ UNIVERSITARĂ	99
V.1 CREAREA TABELELOR APLICAȚIEI	99
V.2 CREAREA SECVENȚELOR APLICAȚIEI	100
V.3 CREARE TRIGGERI APLICAȚIE	102
V.4 PROCEDURI PENTRU ADĂUGARE ÎNREGISTRĂRI ÎN TABELE	103
V.5 PROCEDURI PENTRU MODIFICARE ÎNREGISTRĂRI DIN TABELE	110
V.6 PROCEDURI PENTRU ȘTERGERE ÎNREGISTRĂRI DIN TABELE	113
V.7 PROCEDURI PENTRU REALIZARE APLICAȚII INFORMATICE	119
CAPITOLUL VI	
APLICAȚIE INFORMATICĂ PENTRU URMĂRIREA ACTIVITĂȚII ÎNTR-O AGENȚIE IMOBILIARĂ	126
VI.1 CREAREA TABELELOR APLICAȚIEI	126
VI.2 CREAREA SECVENȚELOR APLICAȚIEI	127
VI.3 CREARE TRIGGERI APLICAȚIE	128
VI.4 PROCEDURI PENTRU ADĂUGARE ÎNREGISTRĂRI ÎN TABELE	128
VI.5 PROCEDURI PENTRU MODIFICARE ÎNREGISTRĂRI DIN TABELE	131
VI.6 PROCEDURI PENTRU ȘTERGERE ÎNREGISTRĂRI DIN TABELE	136
VI.7 PROCEDURI PENTRU REALIZARE APLICAȚII INFORMATICE	139
CAPITOLUL VII	
APLICAȚIE INFORMATICĂ PENTRU EVIDENȚA FACTURILOR DINTR-O SOCIETATE COMERCIALĂ	143
VII.1 CREAREA TABELELOR APLICAȚIEI	143
VII.2 CREAREA SECVENȚELOR APLICAȚIEI	145
VII.3 CREAREA PROCEDURILOR APLICAȚIEI	146
VII.4 REALIZARE APLICAȚII SI CREARE RAPOARTE	150
CAPITOLUL VIII	
APLICAȚIE INFORMATICĂ PENTRU EVIDENȚA INTRĂRILOR DE PRODUSE ÎN GESTIUNE	153
VIII.1 CREAREA TABELELOR APLICAȚIEI	153
VIII.2 CREARE TRIGGERI APLICAȚIE	153
VIII.1 CREARE PROCEDURI APLICAȚIE	154
CAPITOLUL IX	
SOLUȚII INFORMATICE PENTRU APLICAȚII MATEMATICE ȘI TEHNICI DE PROGRAMARE UTILIZÂND PL/SQL	157
IX.1 STABILIREA CERINȚELOR	157

IX.2 DESCRIEREA APLICAȚIEI	158
IX.3 MENIURILE APLICAȚIEI	159
IX.4 APLICAȚII TEHNICI DE PROGRAMARE	166
IX.5 APLICAȚII MATEMATICE	183
IX.6 APLICAȚII INFORMATICE	217
CAPITOLUL X	
EXERCIȚII P.L./S.Q.L	244
CAPITOLUL XI	
TESTE GRILĂ P.L./S.Q.L	261
CAPITOLUL XII	
FUNDAMENTE TEORETICE ALE PACHETULUI DE PRODUSE SOFTWARE	
„ORACLE DEVELOPER”	268
XII.1 DESCRIERE „ORACLE DEVELOPER”	268
XII.2 PREZENTARE FORM BUILDER SAU SQL*FORMS	269
XII.2.1 PREZENTARE FORM BUILDER	269
XII.2.2 PREZENTARE FORM COMPILER	271
XII.2.3 PREZENTARE FORM RUNTIME	272
XII.2.4 CREAREA UNUI VIDEOFORMAT	272
XII.2.4.1 ELEMENTELE UNUI VIDEOFORMAT	275
XII.2.4.2 CREAREA FORMELOR DE TIP „SINGLE BLOCK”	276
XII.2.4.3 RULAREA FORMELOR DE TIP „SINGLE BLOCK”	280
XII.2.4.4 UTILIZAREA EDITORULUI DE AFIȘARE	281
XII.2.4.5 CREAREA FORMELOR DE TIP „MASTER-DETAIL”	283
XII.3 PREZENTARE REPORT BUILDER SAU SQL*REPORT	285
XII.3.1 UTILIZARE PRODUS SOFTWARE „REPORTS WIZARDS”	287
XII.3.2 REALIZAREA RAPOARTELOR	288
CAPITOLUL XIII	
REALIZAREA UNEI APLICAȚII INFORMATICE PENTRU ACTIVITATEA COMERCIALĂ	
DINTR-O SOCIETATE UTILIZÂND „ORACLE DEVELOPER”	292
XIII.1 OBIECTIVELE APLICAȚIEI INFORMATICE	292
XIII.2 DOCUMENTELE APLICAȚIEI INFORMATICE	293
XIII.3 RAPOARTELE APLICAȚIEI INFORMATICE	296
XIII.4 VIDEOFORMATELE APLICAȚIEI INFORMATICE	299
XIII.5 TABELELE APLICAȚIEI INFORMATICE	301
BIBLIOGRAFIE	308

PREFAȚĂ

Într-un mediu de afaceri modern, adaptabilitatea rapidă la cerințele complexe și diversificate ale pieței, poate fi factorul definitoriu care plasează o companie înaintea altor zeci sau poate sute de societăți de profil, care ignoră competitivitatea, flexibilitatea și eficiența. Dar, eforturile manageriale sau strategiile de marketing, fie ele și de ultimă generație, pot deveni ineficiente sau zadarnice, dacă partea economică nu este susținută și de partea informatică din cadrul societății. Eficiența folosirii informațiilor de afaceri depinde de strategia managerială și de suportul hardware și software implementat. Ambele cu aceleași priorități!

La peste 10 ani de secolul XXI, punctele de reper ale unei companii moderne se identifică cu: capacitatea de a se adapta și organiza rapid și eficient la cerințele pieței, flexibilitatea de a prelua și prelucra un volum foarte mare și cu caracter complex de date, analiza într-un timp foarte scurt a rezultatelor obținute în urma prelucrării datelor cu ajutorul aplicațiilor informatice moderne și îmbunătățirea permanentă a platformelor hardware și software.

Soluția rezolvării acestor deziderate este și, cu certitudine...subiectivă, va fi: platforma ORACLE! ORACLE...ORACLE 10g, ORACLE 11g, ORACLE 11g WebLogic SUITE sau, pe viitor...alte versiuni „ORACLE n”, unde $\{n/n>11, n<k, \text{pt } k \leq 100\}g/i$?!

SGBD ORACLE 10g, păstrat în continuare și în acest manual, ca versiune de lucru pentru exemplificarea fundamentelor și realizarea aplicațiilor utilizând platforma ORACLE, permite utilizatorilor dezvoltarea unor soluții informatice complexe, integrarea și utilizarea eficientă a conceptelor de programare precum și derularea și administrarea aplicațiilor economice.

Pentru a ordona volumul foarte mare de informații prelucrate (coroborat cu dorința „perpetuum mobile” a autorului de a îmbunătății și diversifica cunoștințele studenților în domeniu) – cuprinsul lucrării a fost structurat în 3 secțiuni reprezentative, fiecare dintre acestea tratând o arie specifică: „SQL AVANSAT”, „PROGRAMARE SQL”, „ORACLE DEVELOPER-FORMS&REPORTS”.

La rândul lor, fiecare dintre aceste părți include un număr distinct de capitole, reprezentând în fapt „mini”aplicații informatice care, dezvoltate pe baza exemplelor implementate, pot constitui ulterior un punct viabil de plecare în construirea unor produse software complexe.

Fundamentul informatic al lucrării (din punct de vedere didactic) sau „cireașa de pe tort ” (într-o abordare specifică camerelor TV) – este reprezentat(ă) de Limbajul de Programare SQL (PL/SQL), ce se regăsește exemplificat și utilizat în peste 75% din conținutul informațional al materialului propus pentru studiu.

Aplicațiile propuse în această lucrare acoperă o paletă largă de cerințe, atât în ceea ce privește tematica economică abordată, cât mai ales a modului de realizare a lor, prin prisma conceptelor, noțiunilor și exemplurilor informatice implementate.

Noutatea acestei lucrări constă în integrarea, pe de o parte a conceptelor specifice „SQL AVANSAT” (concretizate în tipuri de funcții ORACLE ce permit o serie de analize economice complexe) și, pe de altă parte, realizarea unui capitol special al „Programării SQL”, dedicat „tehnicienilor de programare, inclusiv cu problemele specifice metodelor „Backtracking” sau „Divide et impera”, și exercițiilor de logică matematică.

Datorită simbiozei unitare dintre aspectele teoretice și numeroasele exemple prezentate, lucrarea se adresează studenților și tuturor cititorilor dornici de a explora tărâmul vast, dar șipericulos de atractiv numit: ORACLE. Acest „ocol de minimum 4 ani, 3 luni și 2 zile” poate cauza pasionaților, „dependență” sau mai grav „virusul programatorului ORACLE”: acesta se infiltrează subtil în aplicațiile ORACLE și la intervale de 12 ore fără pauză, virusează aplicația programatorului ORACLE și, printr-un mesaj de avertizare din secțiunea „EXCEPTION” a unui bloc PL/SQL, lansează atacul: „ORA 30000 - LA MASĂĂĂĂĂĂĂĂĂĂ!

Acest sindrom se poate trata doar printr-un „consum suplimentar de programare ORACLE, în cantități moderate de până la 2-3 capitole pe zi de funcții, proceduri, triggere și exemple, oricând în timpul zilei și doar la nevoie noaptea, atunci când se susțin testele și examene la SGBD ORACLE”.

În speranța captării interesului privind programarea și a inițierii în domeniul Programării SQL, doresc să urez studenților succes la examenele universitare, să li se îndeplinească visul de a fi programatori de succes și închei, prin povesti următoarea aneddotă cu tâlc: „într-o grădină stăteau 3 pitici vorbitori; la un moment dat proprietarul se hotărăște să-i vândă; vine un cumpărător și întreabă cât costă primul pitic: „100” răspunde proprietarul; „Dar de ce așa de mult?” întreabă curios cumpărătorul - „Păi e special: știe programare Visual Basic”; „Dar al doilea?” - „200” răspunde proprietarul, și el este un pitic special: știe programare Visual C”. Dar ultimul? întreabă descumpănit cumpărătorul. „1000” răspunde sec vânzătorul. „1000!” Dar de ce? – “Păi scrie pe spatele lui „ORACLE” răspunde plin el vânzătorul. „Păi și dacă scrie, ce este? întreabă cumpărătorul; „Păi nu știu nici eu”, zise vânzătorul, dar îi tot aud pe ceilalți doi zicându-i „șefu”! Deci e mai scump!

**AUTORUL
APRILIE 2012**

CAPITOLUL I

PROGRAMARE ORACLE UTILIZÂND FUNDAMENTE S.Q.L*PLUS AVANSAT PENTRU ANALIZĂ ECONOMICĂ

Pentru exemplificarea fundamentelor SQL*PLUS avansat care stau la baza interpretării rezultatelor economice obținute în urma prelucrării datelor din sistemul economic, se propun funcțiile „**ROLLUP**”, „**CUBE**” și „**GROUPING**”, dar și clauza „**GROUP BY**” foarte des utilizată - care permit o analiză economică detaliată a rezultatelor obținute.

În acest context, s-a optat pentru proiectarea unei structuri mai puțin detaliate a unei baze de date aferente gestionării producției (în contextul exemplificării programării și nu a proiectării), care să permită reliefaarea funcțiilor SQL avansat ale platformei Oracle în ceea ce privește analiza economică a datelor prelucrate.

I.1 PROIECTAREA APLICAȚIEI INFORMATICE PRIVIND EVIDENȚA PRODUSELOR FABRICATE ȘI IMPLEMENTAREA ÎN SQL*PLUS

PREZENTARE ACTIVITATE

Într-o societate economică, al cărui principal obiect de activitate este „*Producția de bunuri*”, se impune, la un moment dat, proiectarea și implementarea unei baze de date, astfel încât să se asigure informatizarea procesului de gestiune a produselor fabricate, prin evidența „intrărilor” (materile prime), evidența „ieșirilor” (produsele finite) și a proceselor „intermediare” tehnologice.

Aplicația solicitată se va dezvolta utilizând standardul SQL, dar implementat în cadrul platformei Oracle (respectiv, SQL*PLUS) și va asigura o interpretare de ansamblu a rezultatelor obținute în urma prelucrării datelor preluate din sistem.

PROIECTAREA STRUCTURII BAZEI DE DATE

Baza de date privind evidența „*Producției de bunuri*” se denumește „*ProdData*” și include *trei* tabele: tabela „*Materii prime*”, tabela „*Procese*” și tabela „*Produce_Fabr*” – fiecare dintre aceste entități ale sistemului economic real identificând o parte a activității proiectate: *gestiunea materiilor prime pentru fabricație*, *prelucarea materiilor prime* respectiv, *evidența produselor finite rezultate*”.

Fiecare dintre cele trei tabele proiectate vor „surpinde” *caracteristicile definitorii* ale entităților identificate: „*Materii prime*”, „*Procese de fabricație*” și „*Produse finite*”, astfel încât, într-o prezentare succintă a fundamentelor procesului economic privind „*evidența produselor finite rezultate în urma derulării procesului de fabricație*” - să poată fi dezvoltată o bază de date cât mai reprezentativă din punct de vedere informațional, dar și o aplicație informatică capabilă să permită automatizarea procesului de gestiune.

Produsele finite rezultate în urma implementării proceselor tehnologice de fabricație privesc îndeosebi componentele auto, descrise în mod generic prin „Bloc motor”, „Panou bord” sau „Sisteme ABS”, și sunt rezultatul implementării etapelor de realizare definite prin procesele „Prelucrare Tehnologică”, „Ansamblare tehnică” și „Finisare operațională”.

Pentru proiectarea tabelelor bazei de date „*ProdData*” se vor identifica cu „*N*” câmpurile de tip „*numeric*”, cu „*C*”, câmpurile de tip „*caracter*” și cu „*D*”, câmpurile de tip „*dată*”, urmând ca pentru fiecare dintre aceste tipuri de date să fie stabilită în structura SQL*PLUS identitatea corespunzătoare: „*Number*”, „*Varchar2*” și respectiv, „*Date*”.

Totodată, atributul unei tabele identificat prin simbolul „*Pk*” (sau „*Primary Key*”) stabilește „*cheia primară*” a tabelei (care permite identificarea unică a tuplurilor), iar simbolul „*Fk*” (sau „*Foreign Key*”) stabilește „*cheia externă*” a tabelei (cea care permite definirea asocierii dintr o tabelă de bază și o tabelă de referită).

În tabela „*Produse_Fabr*”, atributul „*cheie primară*” este compus (fiind definit pe baza valorilor atributelor „*CodP*” și „*CodMP*”), iar atributul „*cheie externă*” este definit prin „*CodProc*”; astfel se presupune ipoteza că „un produs este realizat pe baza mai multor categorii de materii prime”; procesele de fabricație sunt identificate generic, astfel încât pot să cuprindă o succesiune variată de faze și operații de lucru.

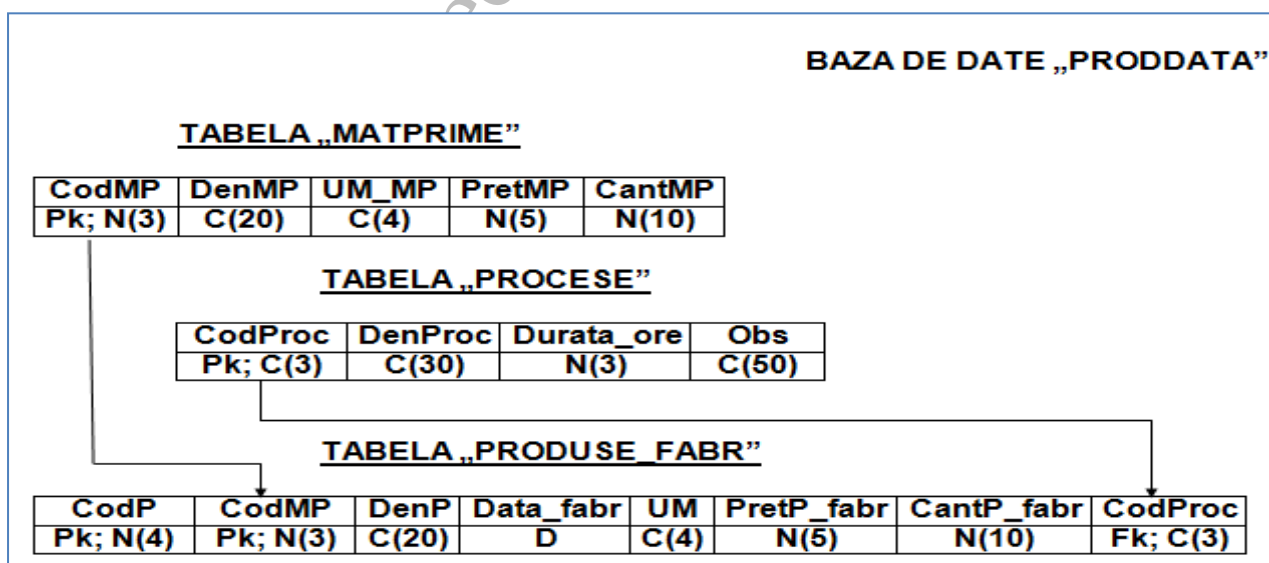


Figura 1: Proiectarea structurii bazei de date „*ProdData*” pentru evidența produselor finite rezultate în urma procesului de fabricație

PROGRAMAREA SQL*PLUS

Implementarea în mediul de programare SQL*PLUS a cerințelor de proiectare prezentate anterior, presupune: *crearea tabelor aplicației, vizualizarea structurii tabelor, inserarea de tupluri în cadrul tabelor și vizualizarea înregistrărilor adăugate.*

CREAREA TABELOR APLICAȚIEI

Pentru ușurință în programarea aplicației, se recomandă, preliminar creării tabelor aplicației, utilizarea unui fișier „.sql”, denumit „**proddata.sql**”, în care se vor introduce *toate comenzile aferente procesului de creare tabelle și inserare tupluri.*

PROMPT CREARE TABELE

```
SQL> DROP TABLE Produse_Fabr;  
      DROP TABLE MatPrime;  
      DROP TABLE Procese;
```

PROMPT CREARE TABELA „MATERII PRIME”

```
SQL> CREATE TABLE MatPrime (  
      CodMP      Number(3),  
      DenMP      Varchar2(20),  
      UM_MP      Varchar2(4),  
      PretMP     Number(5),  
      CantMP     Number(10),  
      CONSTRAINT pk_codmp PRIMARY KEY(CodMP) );
```

```
SQL> DESCRIBE MatPrime;
```

PROMPT CREARE TABELA „PROCESE”

```
SQL> CREATE TABLE Procese (  
      CodProc    Varchar2 (3),  
      DenProc    varchar2(30),  
      Durata_ore Number(3),  
      Obs        Varchar2(50),  
      CONSTRAINT pk_codproc PRIMARY KEY(CodProc));
```

```
SQL> DESCRIBE PROCESE;
```

PROMPT CREARE TABELA „PRODUSE_FABR”

```
SQL> CREATE TABLE Produse_Fabr(  
      CodP      Number(4),  
      CodMP     Number(3),  
      DenP      Varchar2(20),
```

```
Data_fabr Date default Sysdate,
UM_P Varchar2(4),
PretP_fabr Number(5),
CantP_fabr Number(10),CodPROC varchar2(3),
CONSTRAINT pk_codp_codmp PRIMARY KEY(CodP, CodMp),
CONSTRAINT fk_codproc FOREIGN KEY (CodPROC)
REFERENCES Procese (CodPROC));
```

```
SQL> DESCRIBE Prognose_Fabr;
```

INSERAREA TUPLURILOR ÎN TABELE

Adăugarea înregistrărilor în tabele se va realiza în funcție de informațiile prelevate din analiza sistemului, astfel încât datele introduse să reflecte cât mai exact realitatea economică.

PROMPT INSERARE TUPLURI IN TABELE

```
SQL> DELETE FROM Prognose_Fabr;
SQL> DELETE FROM MatPrime;
SQL> DELETE FROM Procese;
```

PROMPT INSERARE TUPLURI IN TABELA „MATERII PRIME”

```
SQL> INSERT INTO MatPrime VALUES(1, 'Aliaj', 'Tone', 1200, 110);
INSERT INTO MatPrime VALUES(2, 'Subansamble','Buc', 350, 500);
INSERT INTO MatPrime VALUES(3, 'Prefabricate','Buc', 2000, 880);
SQL> SELECT * FROM MatPrime;
```

PROMPT INSERARE TUPLURI IN TABELA „PROCESE”

```
SQL> INSERT INTO Procese VALUES ('P1','Prelucrare tehnologica', 10,
'Se prelucreaza materialul pentru ansamblare');
INSERT INTO Procese VALUES ('P2','Ansamblare tehnica', 25,
'Se assembleaza produsul nefinisat');
INSERT INTO Procese VALUES ('P3','Finisare operationala', 15,
'Se finiseaza produsul finit');
SQL> SELECT * FROM Procese;
```

PROMPT INSERARE TUPLURI IN TABELA „PRODUSE_FABR”

```
SQL> INSERT INTO Prognose_Fabr VALUES (101, 1, 'Bloc motor',
TO_DATE('01-01-11','DD-MM-YY'), 'Tone', 1500, 200, 'P1');
INSERT INTO Prognose_Fabr VALUES (102, 1, 'Caroserie',
TO_DATE('10-10-10','DD-MM-YY'), 'Buc', 1200, 410, 'P1');
INSERT INTO Prognose_Fabr VALUES (102, 1, 'Caroserie',
TO_DATE('11-11-11','DD-MM-YY'), 'Buc', 11500, 100, 'P2');
```

```

SQL> INSERT INTO Produse_Fabr VALUES (103, 1, 'Panou bord',
    TO_DATE('10-11-11','DD-MM-YY'), 'Buc', 10000, 100, 'P2');
INSERT INTO Produse_Fabr VALUES (103, 2, 'Panou bord',
    TO_DATE('17-11-11','DD-MM-YY'), 'Buc', 11500, 200, 'P3');
INSERT INTO Produse_Fabr VALUES (103, 3, 'Panou bord',
    TO_DATE('12-11-11','DD-MM-YY'), 'Buc', 9000, 100, 'P3');
INSERT INTO Produse_Fabr VALUES (104, 2, 'Sisteme ABS',
    TO_DATE('01-12-11','DD-MM-YY'), 'Buc', 11550, 100, 'P2');
INSERT INTO Produse_Fabr VALUES (104, 3, 'Sisteme ABS',
    TO_DATE('08-12-11','DD-MM-YY'), 'Buc', 8500, 350, 'P3');
SQL> SELECT * FROM Produse_Fabr;

```

I.2 FUNCȚII ORACLE PENTRU ANALIZA ECONOMICĂ A REZULTATELOR

I.2.1 GRUPAREA LINIILOR TOTALIZATOARE - CLAUZA „GROUP BY”

Clauza „GROUP BY” este utilizată pentru a diviza liniile unui tabel în grupuri. Pentru a returna informația corespunzătoare fiecărui astfel de grup, se pot utiliza funcțiile agregat. Funcțiile agregat pot fi definite în clauzele **SELECT**, **ORDER BY** și **HAVING**.

Server-ul Oracle aplică aceste funcții fiecărui grup de linii și returnează un singur rezultat pentru fiecare mulțime.

Aplicații „GROUP BY”:

În tabela „Produse fabricate” se pot stabili mai multe „grupuri de tupluri” cu ajutorul cărora se pot obține rezultate semnificative și se pot defini o serie de interpretări economice. Exemplificarea fundamentelor de bază privind gruparea datelor și interpretarea rezultatelor, se va realiza utilizând facilitățile oferite de clauza „GROUP BY”.

Pentru o mai bună reprezentativitate a informațiilor prelevate, structurarea exemplelor propuse se va face alegând ca puncte de reper attributele cheii primare compuse din tabela „Produse_fabr”: „CodP” și „CodMP”.

```
SQL> SELECT*FROM Produse_fabr;
```

```
SQL> PROMPT GRUPAREA PRODUSELOR IN FUNCTIE DE COD PRODUS
PROMPT CODP ESTE PRIMUL FACTOR DE GRUPARE
```

```
SQL> SELECT Codp, SUM(Cantp_fabr) as "CANT_PROD_FABR", CodMP
FROM Produse_Fabr
WHERE CodP=103
GROUP BY (CodP, CodMP);
```