Baze de Date II

(Proiectarea Sistemelor cu Baze de Date)

Cornelia TUDORIE

Proiectarea Sistemelor cu Baze de Date - Cuprins

A. **Modelul Relaţional.**
B. **Limbajul SQL. Lucrul cu tabele.**
C. **Sisteme cu baze de date. Performanţa.**
D. **Obiecte ale sistemului de baze de date. Performanţa în utilizare**
E. **Programe pentru baze de date. Performanţa în programare.**
F. **Protecţia datelor.**
G. **Sisteme Informatice. Proiectarea sistemelor de baze de date.**

D. Obiecte ale sistemului de baze de date. Performanţa în utilizare.

1. Stocarea informaţiilor în baza de date
2. Obiectele bazei de date:
    1. Tabele
    2. Indecşi
    3. Tabele virtuale
    4. Sinonime
    5. Secvenţe
3. Gestiunea eficientă a spaţiului de stocare
4. Optimizarea cererilor

D. 3. Gestiunea eficientă a spaţiului de stocare

Urmărirea situaţiei utilizării spaţiului

1. Urmărirea situaţiei spaţiului alocat
2. Urmărirea înlănţuirii datelor

Urmărirea situaţiei spaţiului alocat

Dicţionarul de date

- **USER_SEGMENTS**
- **USER_EXTENTS**
- **USER_FREE_SPACE**

Urmărirea situaţiei spaţiului alocat

USER_SEGMENTS

General information

- OWNER
- SEGMENT_NAME
- SEGMENT_TYPE
- TABLESPACE_NAME

Size

- EXTENTS
- BLOCKS

Storage settings

- INITIAL_EXTENT
- NEXT_EXTENT
- MIN_EXTENTS
- MAX_EXTENTS
- PCT_INCREASE

Urmărirea situaţiei spaţiului alocat

USER_EXTENTS

Identification

- **OWNER**
- **SEGMENT_NAME**
- **EXTENT_ID**

Location and size

- **TABLESPACE_NAME**
- **RELATIVE_FNO**
- **FILE_ID**
- **BLOCK_ID**
- **BLOCKS**

Urmărirea situaţiei spaţiului alocat

USER_FREE_SPACE

Location and size

- TABLESPACE_NAME
- RELATIVE_FNO
- FILE_ID
- BLOCK_ID
- BLOCKS

Urmărirea situaţiei spaţiului alocat

Exemplu

```
SQL> describe user_segments
 Name                                      Null?    Type
 ----------------------------------------- -------- ----
 SEGMENT_NAME                                       VARCHAR2(81)
 SEGMENT_TYPE                                       VARCHAR2(18)
 TABLESPACE_NAME                                    VARCHAR2(30)
 BYTES                                              NUMBER
 BLOCKS                                             NUMBER
 EXTENTS                                            NUMBER
 INITIAL_EXTENT                                     NUMBER
 NEXT_EXTENT                                        NUMBER
 MIN_EXTENTS                                        NUMBER
 MAX_EXTENTS                                        NUMBER
 PCT_INCREASE                                       NUMBER
```

Urmărirea situaţiei spaţiului alocat

Exemplu

```
SQL> select segment_name, segment_type, extents,
  2  initial_extent, next_extent, pct_increase
  3  from user_segments;
SEGMENT_NAME SEGMENT_TYPE  EXTENTS INITIAL_EXTENT NEXT_EXTENT PCT_INCREASE
------------ ------------- ------- -------------- ----------- ------------
VANZARE      TABLE               1          65536
CLIENT       TABLE               1          65536
PROD         TABLE               1          65536
CLIENT_PK    INDEX               1          65536
PROD_PK      INDEX               1          65536
```

D. Obiecte ale sistemului de baze de date. Performanţa în utilizare.

1. Stocarea informaţiilor în baza de date
2. Obiectele bazei de date:
    1. Tabele
    2. Indecşi
    3. Tabele virtuale
    4. Sinonime
    5. Secvenţe
3. Gestiunea eficientă a spaţiului de stocare
4. Optimizarea cererilor

D. 4. Optimizarea cererilor

Modelul relaţional - Limbaje neprocedurale

SGBD - responsabil cu generarea unui plan de execuţie optim (minimizarea timpului de răspuns)

- are la dispoziţie:

- **informaţii statistice**
- **dinamism în raport cu evoluţia bazei de date**
- **capacitate de analiză rapidă a numeroase variante**

D. 4. Optimizarea cererilor

Nivelul sistem

Scopul principal: Alegerea celei mai eficiente strategii de evaluare a unei expresii relaţionale

Nivelul utilizator

Totuşi, şi utilizatorul poate contribui la sporirea performanţelor, prin crearea unui context adecvat

Optimizarea cererilor (nivelul Sistem)

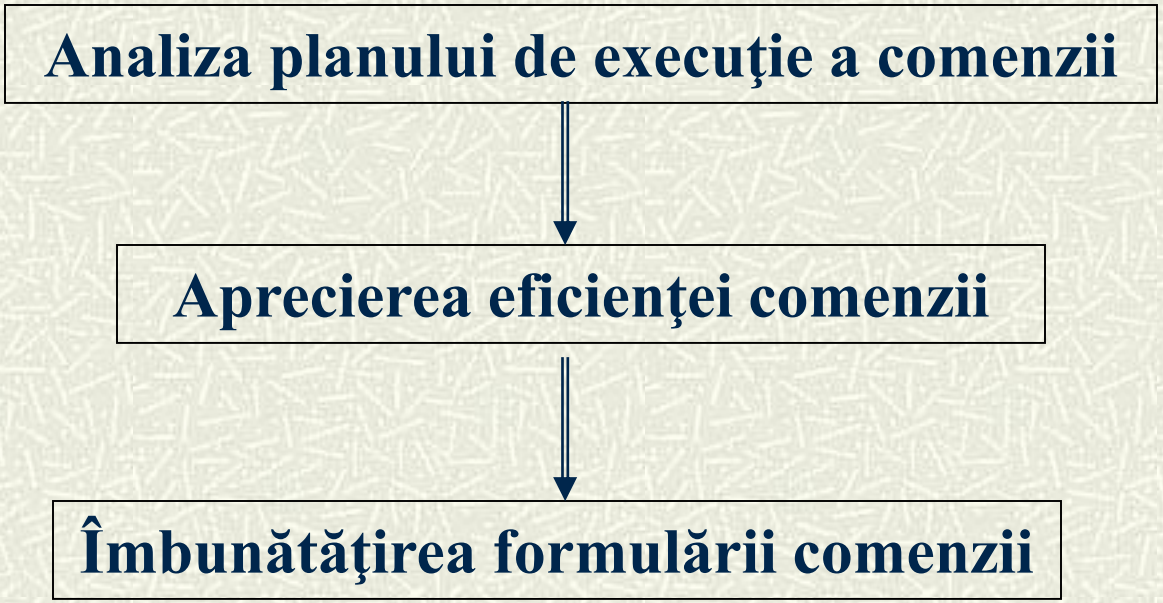
Evaluarea cererii:

I. **Analiza cererii**

II. **Planificarea execuţiei**

III. **Execuţia**

Optimizarea cererilor (nivelul Utilizator)



Proiectarea Sistemelor cu Baze de Date - Cuprins

A. **Modelul Relaţional.**
B. **Limbajul SQL. Lucrul cu tabele.**
C. **Sisteme cu baze de date. Performanţa.**
D. **Obiecte ale sistemului de baze de date. Performanţa în utilizare**
E. **Programe pentru baze de date. Performanţa în programare.**
F. **Protecţia datelor.**
G. **Sisteme Informatice. Proiectarea sistemelor de baze de date.**

E. Programe pentru baze de date. Performanţa în programare.

1. **Fişiere de comenzi**
2. **PL/SQL. Tipuri de programe**
3. **Blocuri anonime**
4. **Subprograme stocate**
5. **Pachete**
6. **Declanşatori**

E. 1. Fişiere de comenzi

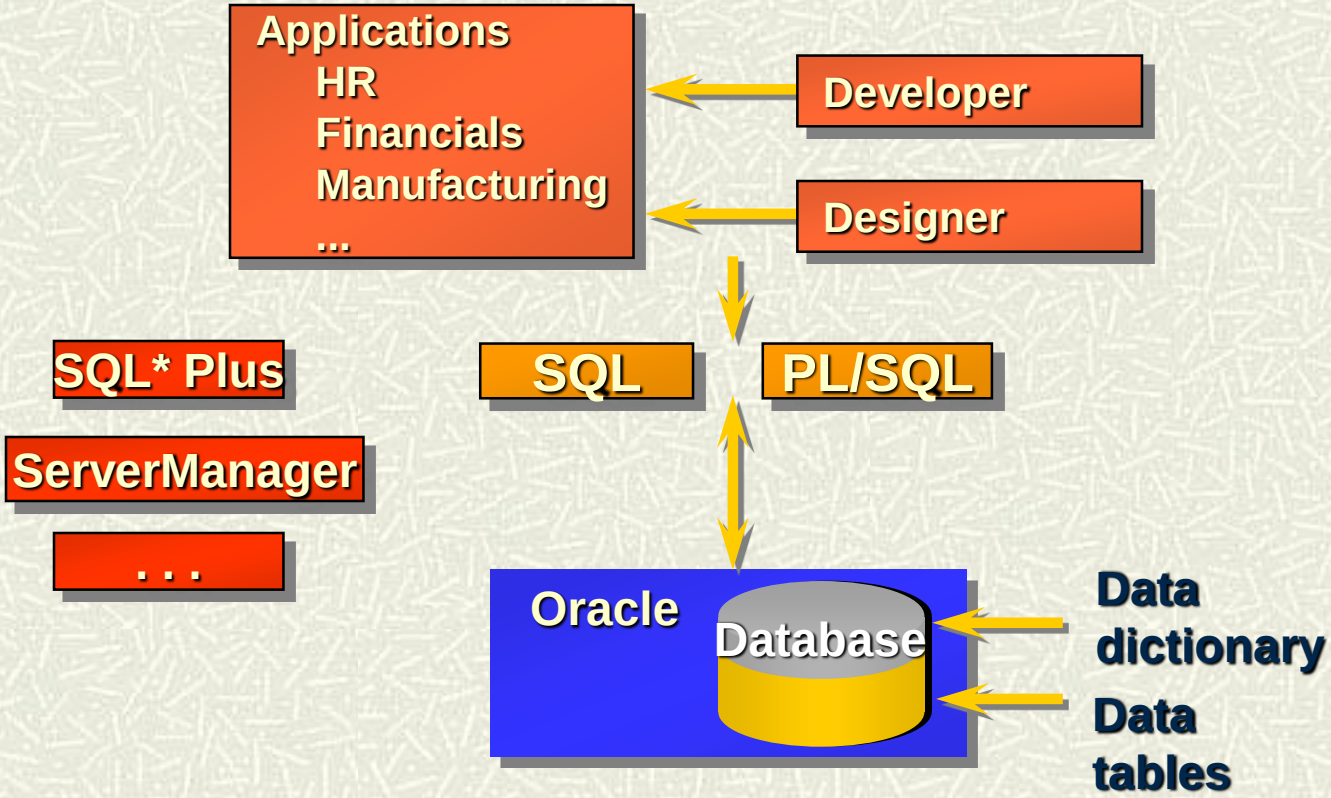
Fişier de comenzi **(script) :**

- forma cea mai simplă de program

- secvenţă de comenzi

- pot fi lansate din orice program de interfaţă în mod comandă

Oracle Complete Solution



Utilitarul SQL*Plus

SQL*Plus

- instrumentul cel mai direct şi la îndemână pentru operare şi dezvoltare de aplicaţii
- util tuturor tipurilor de utilizatori
- acceptă şi trimite serverului:
  - comenzi SQL
  - instrucţiuni PL/SQL
- acceptă şi tratează:
  - comenzi SQL*Plus

Fişiere de comenzi SQL*Plus

Fişierul de comenzi SQL*Plus poate conţine:

- comenzi SQL
- instrucţiuni PL/SQL
- comenzi SQL*Plus
- apelul altor fişiere de comenzi

Fişiere de comenzi SQL*Plus

Fişierul de comenzi SQL*Plus poate fi creat şi modificat:

- în minieditorul SQL*Plus
- în editoarele Sistemului de Operare (Notepad)

Comenzi SQL*Plus pentru fişiere de comenzi

- **SAV[E]** ***filename*** **[CRE[ATE] | REP[LACE] | APP[END]**
- **GET** ***filename***
- **STA[RT]** ***filename***
- ***@ filename***
- **EDI[T]** ***filename***
- **SPO[OL]** ***filename*** **| OFF | OUT**

Fişiere de comenzi. Exemple

Exemplu
Crebd.sql

```
--  Creare database <stud>
create database stud
    controlfile reuse
    logfile 'c:\orastud\DATA\log1a.ora' size 200K reuse,
            'c:\orastud\DATA\log2a.ora' size 200K reuse
    datafile 'c:\orastud\DATA\system01.ora' size 10M reuse
      autoextend on   next 10M maxsize 200M
    character setWE8ISO8859P1;
create tablespace data01
    datafile 'c:\orastud\DATA\data01.ora' size 3M reuse
      autoextend on   next 5M maxsize 150M;
.   .   .   .
```

Fişiere de comenzi. Exemple

Exemplu
Cremaster.sql

```
REM          creare rol master
create role master;
grant connect , resource to master;

REM          creare utilizatori master si tabelele lor
create user master1 identified by oracle
  default tablespace users temporary tablespace temp
  quota unlimited on users;
grant master to master1;
Connect master1/oracle@stud
@cretabs
.   .   .   .   .
```

Parametri de mediu utili în fişiere de comenzi

***parameter* :**

- **echo**
- **heading**
- **pagesize**
- **pause**
- **termout**
- **verify**
- **. . .**

SET *parameter* ON | OFF

SHOW ALL | *parameter*

Parametri de mediu utili în fişiere de comenzi

Exemplu
Cretabs.sql

```
Set TERMOUT OFF
REM       creare 3 tabele prod,vanzare,client
DROP TABLE PROD;
DROP TABLE CLIENT;
DROP TABLE VANZARE;

CREATE TABLE CLIENT (
 codcli                NUMBER(3)  ,
 nume                 VARCHAR2(15),
 loc                      VARCHAR2(15),
 CONSTRAINT CLIENT_PK PRIMARY KEY (codcli));
INSERT INTO CLIENT  VALUES (31,'Ionescu','Galati');
INSERT INTO CLIENT  VALUES (32,'Popescu','Iasi');
. . . .
Set TERMOUT ON
```

Variabile de substituţie SQL*Plus

Exemplu

```
INSERT INTO  dept (deptno, dname, loc)
  VALUES (&department_id,'&department_name','&location');
```

```
Enter value for department_id: 80
Enter value for department_name: EDUCATION
Enter value for location: ATLANTA
1 row created.
```

Variabile de substituţie SQL*Plus

Exemplu

```
ACCEPT   department_id -
         PROMPT 'Please enter the department number:'
ACCEPT   department_name -
         PROMPT 'Please enter the department name:'
ACCEPT   location PROMPT 'Please enter the location: '

INSERT INTO  dept (deptno, dname, loc)
   VALUES (&department_id, '&department_name','&location');
```

ACCEPT *substitution_variable* [PROMPT *text*][HIDE]

Variabile de substituţie SQL*Plus

***parametri pentru substituţie* :**

- **scan (on | off)**
- **define (*character*)**
- **escape (*character*)**
- **verify (on | off)**
- **concat (*character*)**

Fişiere de comenzi cu parametri

Exemplu
Cretabs.sql

```
REM       creare 3 tabele prod,vanzare,client
REM       pentru un utilizator master
Connect master.&1/oracle@stud
CREATE TABLE CLIENT.&1 (
 codcli                NUMBER(3)  ,
 nume                 VARCHAR2(15),
 loc                      VARCHAR2(15),
 CONSTRAINT CLIENT_PK PRIMARY KEY (codcli));
INSERT INTO CLIENT  VALUES (31,'Ionescu','Galati');
. . . .
```

```
SQL>@Cretabs 5
```

@ filename parameter1 [, parameter2 , . . .]

Fişiere de comenzi pentru rapoarte

```
SELECT p.den, p.culoare, c.nume, v.cant
  FROM prod p , vanzare v, client c
  WHERE p.codp=v.codp AND v.codcli=c.codcli
  ORDER BY p.den, p.culoare;
```

```
DEN        CULOARE    NUME                 CANT
---------- ---------- --------------- ---------
panza      alb        Marin                  35
panza      alb        Marin                  20
. . . .
vopsea     rosu       Ionescu                50

10 rows selected.
```

Fişiere de comenzi pentru rapoarte

```
  'Lista vanzarilor'

Denumirea                Numele           Cantitatea
produsului  Culoarea     clientului         cumparata
----------  ----------   ---------------  -----------
panza       alb          Marin                    35
                         Marin                    20
            rosu         Vasile                   50
                         Popescu                  60
            verde        Vasile                  110
                         Popescu                  10
                         Ionescu                  40
**********  **********                    -----------
sum                                                325

vopsea      rosu         Ionescu                  20
                         Marin                    30
                         Ionescu                  50
**********  **********                    -----------
sum                                                100

'Pagina:'    1
```

Fişiere de comenzi pentru rapoarte

Comenzi pentru formatarea rapoartelor :

- **TTITLE [*text* | ON | OFF]**
- **BTITLE [*text* | ON | OFF]**
- **COLUMN *column* [FORMAT *format*] [HEADING "*text*"]**
- **BREAK ON *report_element***
- **COMPUTE *aggr_func* OF *calc_column* ON *report_element***
- **CLEAR COLUMN**
- **COLUMN *column* CLEAR**

Fişiere de comenzi pentru rapoarte

```
SET LINESIZE 50
SET PAGESIZE 25
SET FEEDBACK OFF
TTITLE CENTER 'Lista  vanzarilor' SKIP 1 LINE
BTITLE LEFT 'Pagina: ' FORMAT 999 sql.pno

COLUMN den HEADING "Denumirea |produsului" FORMAT A10
COLUMN culoare HEADING "Culoarea" FORMAT A10
COLUMN nume HEADING "Numele |clientului" FORMAT A15
COLUMN cant HEADING "Cantitatea |cumparata" FORMAT 9999

BREAK ON den SKIP 2 ON culoare
COMPUTE SUM OF cant ON den REPORT

SELECT p.den, p.culoare, c.nume, v.cant
  FROM prod p , vanzare v, client c
  WHERE p.codp=v.codp AND v.codcli=c.codcli
  ORDER BY p.den, p.culoare;

TTITLE OFF
BTITLE OFF
SET FEEDBACK ON
```

Generarea de fişiere de comenzi

Exemplu

Gen.sql

```
SET HEADING OFF
SET FEEDBACK OFF
SELECT 'SELECT * FROM '||table_name||';'FROM  user_tables;
```

SQL*Plus

```
@Gen.sql
SPOOL Fcom.sql
/
SPOOL OFF
```

Fcom.sql

```
SELECT * FROM CLIENT;
SELECT * FROM PLAN_TABLE;
SELECT * FROM PROD;
SELECT * FROM VANZARE;
```

SQL*Plus

```
@Fcom
```

E. Programe pentru baze de date. Performanţa în programare.

1. Fişiere de comenzi
2. PL/SQL. Tipuri de programe
3. Blocuri anonime
4. Subprograme stocate
5. Pachete
6. Declanşatori

E. 2. PL/SQL. Tipuri de programe

***De ce PL/SQL?* :**

- ***SQL - limbaj neprocedural***
- ***limbaj procedural: variabile, structuri de control, subprograme, interfata, dialog cu operatorul, etc***

integrarea SQL-ului în limbaj procedural

E. 2. PL/SQL. Tipuri de programe

***Limbajul Oracle PL/SQL* :**

- extensie procedurală a limbajului SQL, destinată programării

- comenzi SQL cu acces la baza de date sunt incluse în programe PL/SQL

- orientat pe blocuri

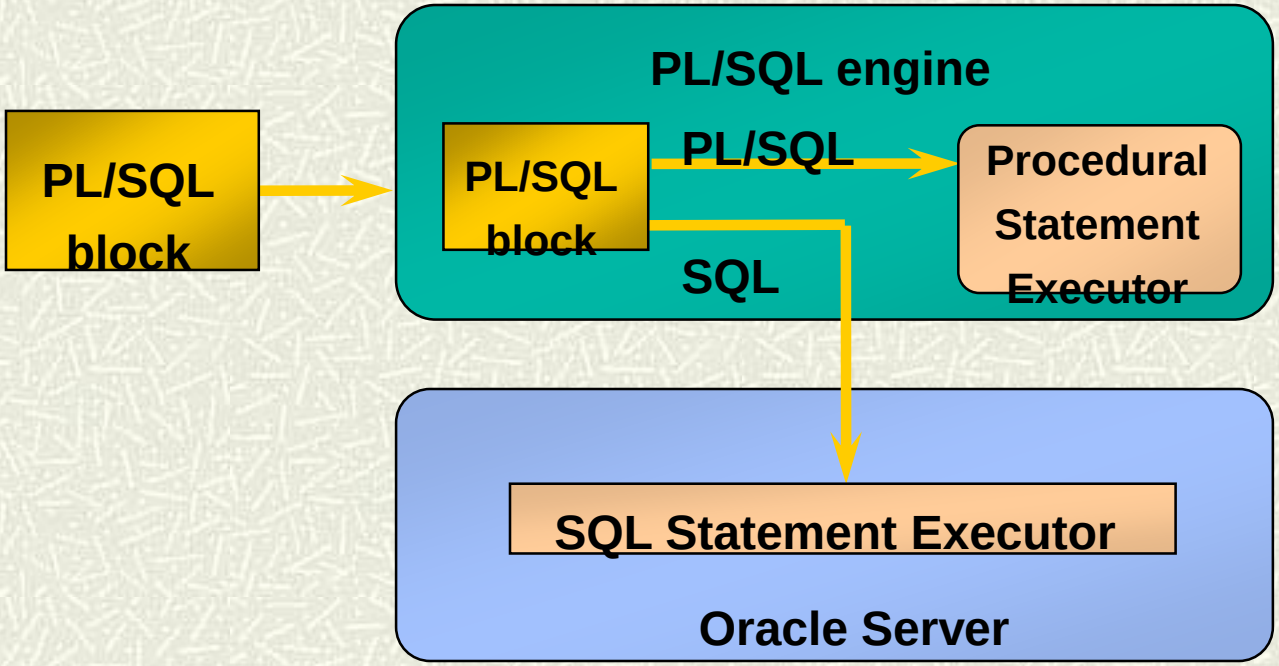
E. 2. PL/SQL. Tipuri de programe

***Limbajul Oracle PL/SQL* :**

- ***Comenzile SQL* sunt executate de motorul SQL al serverului (SQL Statement Executor)**

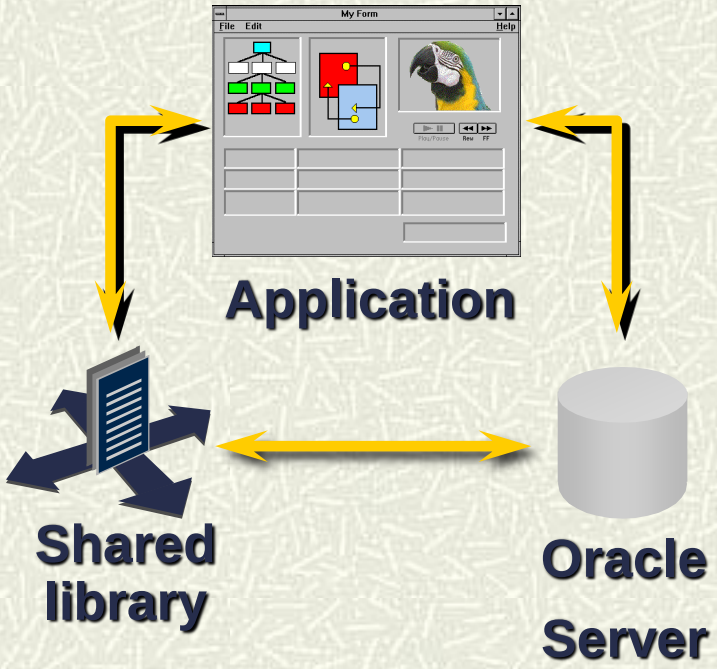
- ***Instructiunile PL/SQL* sunt executate de un motor PL/SQL (Procedural Statement Executor)**
  - ***fie la nivelul serverului***
  - ***fie la nivelul aplicatiei Oracle***

Procesarea blocurilor PL/SQL



Avantaje PL/SQL

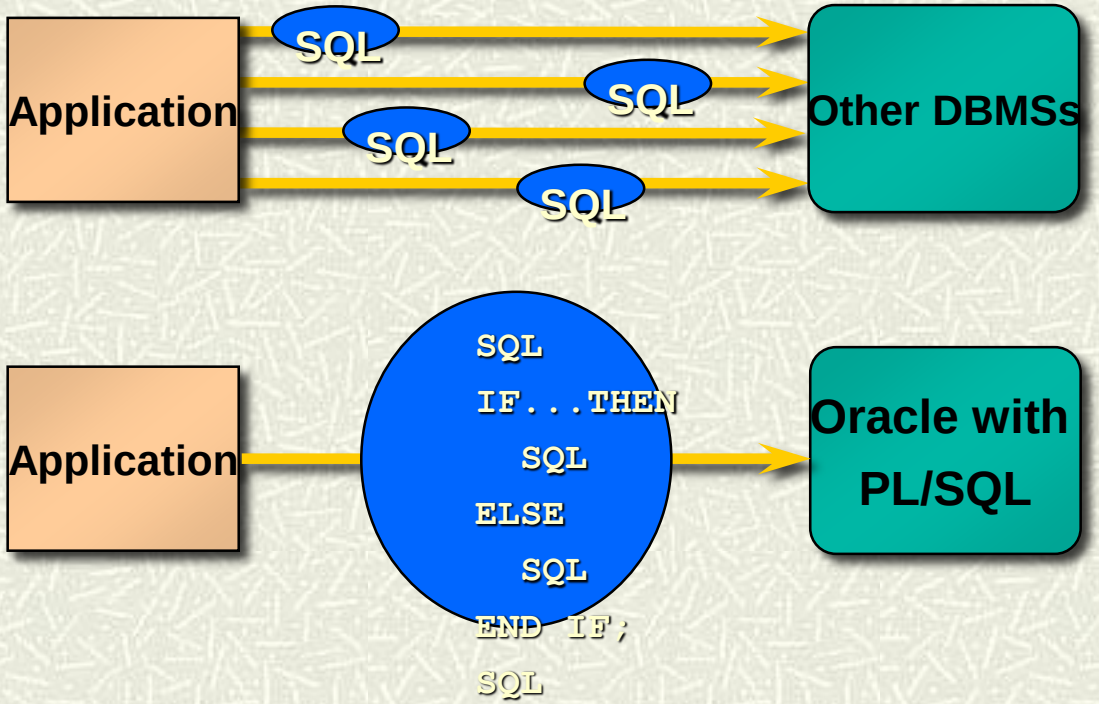
Integrarea blocurilor PL/SQL



Avantaje PL/SQL

Flexibilitate

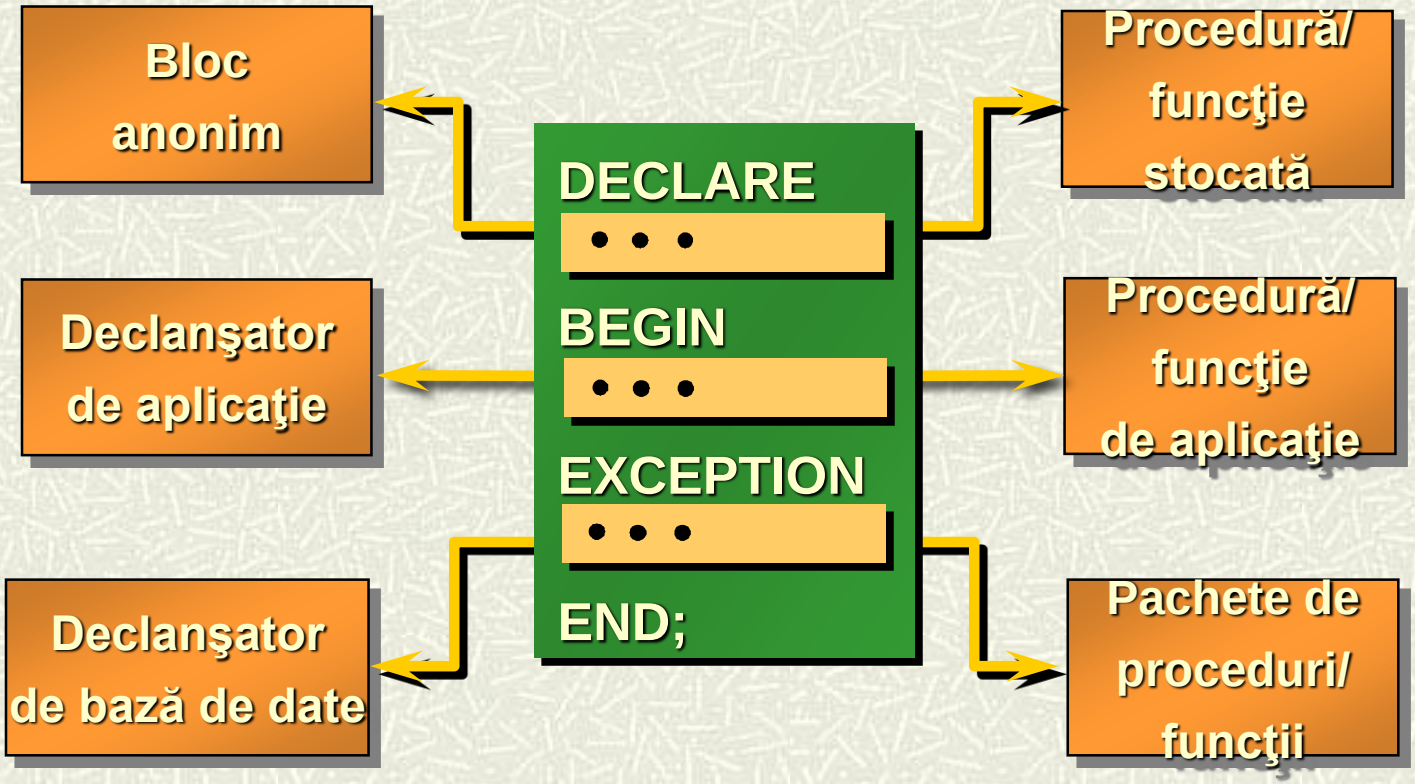
Îmbunătăţirea performanţelor



Avantaje PL/SQL (limbaj procedural)

- **Permite declararea de identificatori**
- **Implementează structuri de control**
- **Interceptează şi tratează erorile**
- **Este portabil**
- **Permite modularizare (blocuri imbricate, subprograme, subprograme catalogate, pachete)**

PL/SQL, limbaj bazat pe blocuri



PL/SQL, limbaj bazat pe blocuri

Sectiune	Descriere	
Declarativa	Contine definitiile structurilor si variabilelor utilizate in bloc.	Optionala
Executabila	Contine comenzi SQL si instructiuni PL/SQL	Obligatorie
Exceptii	Descrie actiunile de executat la aparitia unei erori sau a unei situatii anormale in sectiunea executabila.	Optionala

PL/SQL, limbaj bazat pe blocuri

Exemplu:

```
DECLARE
        v_variable          VARCHAR2(5)
BEGIN
      SELECT    column_name
        INTO    v_variable
        FROM    table_name;
EXCEPTION
         WHEN  exception_name THEN
           …
END;
```

PL/SQL, limbaj bazat pe blocuri

Bloc anonim

[DECLARE]
sectiune declarativa
BEGIN
sectiune executabila
[EXCEPTION]
sectiune de exceptii
END;

Bloc cu nume

antet
IS
sectiune declarativa
BEGIN
sectiune executabila
[EXCEPTION]
sectiune de exceptii
END;

PL/SQL, limbaj bazat pe blocuri

Bloc anonim

```
DECLARE
        astazi  DATE DEFAULT SYSDATE;
BEGIN
        dbms_output.put_line('Buna ziua! Suntem in data de' || astazi);
END;
```

PL/SQL, limbaj bazat pe blocuri

Bloc cu nume

```
CREATE OR REPLACE PROCEDURE salut
IS
        astazi  DATE DEFAULT SYSDATE;
BEGIN
        dbms_output.put_line('Buna ziua! Suntem in data de' || astazi);
END;
```

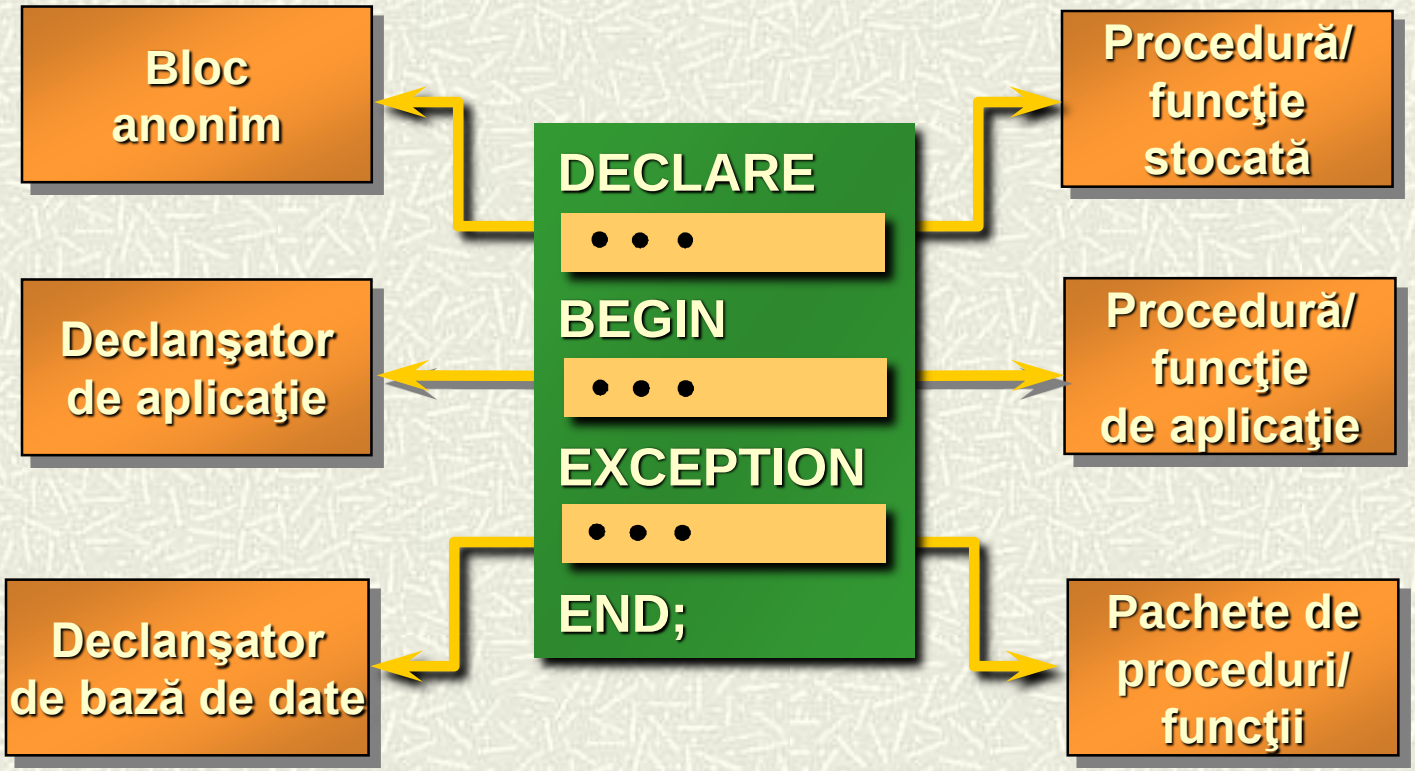
PL/SQL, limbaj bazat pe blocuri

- ***un program PL/SQL poate contine comenzi SQL de tip LMD, dar nu LDD***
- ***gestiunea tranzactiilor este identica, fie ca se lucreaza in SQL sau in PL/SQL***

Avantaje PL/SQL (limbaj procedural)

- **Permite declararea de identificatori**
- **Implementează structuri de control**
- **Interceptează şi tratează erorile**
- **Este portabil**
- **Permite modularizare (blocuri imbricate, subprograme, subprograme catalogate, pachete)**

Tipuri de programe PL/SQL



E. Programe pentru baze de date. Performanţa în programare.

1. Fişiere de comenzi
2. PL/SQL. Tipuri de programe
3. Blocuri anonime
4. Subprograme stocate
5. Pachete
6. Declanşatori

E. 3. Blocuri anonime

- **Structura blocului**
- **Variabile**
- **Accesul la baza de date**
- **Structuri de control**
- **Tipuri de date compuse**
- **Cursori expliciţi**
- **Tratarea excepţiilor**