

Tehnici de programare cu baze de date

#1

PL/SQL

Concepte generale

<https://www.runceanu.ro/adrian/activitate-didactica/>

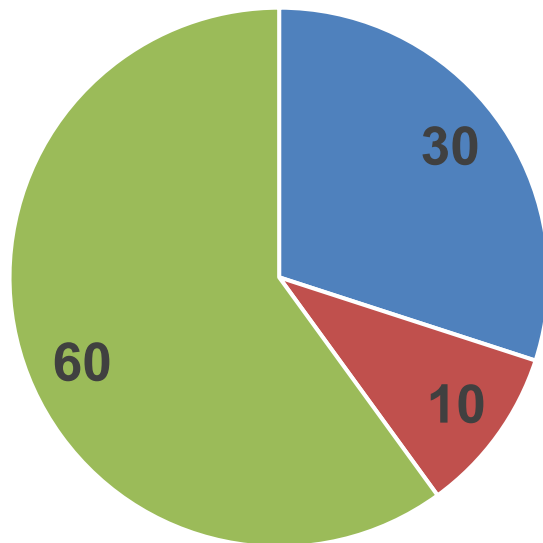
Câteva precizări

Structura cursului

- **2 ore curs** – titular curs:
Conf. univ. dr. Adrian Runceanu
- **2 ore laborator** – titular aplicații practice:
Conf. univ. dr. Adrian Runceanu

Câteva precizări

Procentaje evaluare



- Evaluare pe parcursul semestrului

Forme de examinare:

- Examen final = 60%
- Activitate la curs și laborator = 10%
- Verificare finală lucrări de laborator = 30%



Câteva precizări

Bibliografia necesară cursului:

1. Abiteboul S. etc: "Foundations of Databases", Addison Wesley, 95
2. Date C.J.: "An Introduction to Database Systems, ed.8, Addison Wesley, 2004
3. Fotache M. etc. : "Oracle 9i – Ghidul dezvoltarii aplicatiilor profesionale", Polirom, 2003
4. Garcia-Molina H., Ullman J.D. : "Database Systems.The Complete Book", 2000
5. Popescu Ileana: "Modelarea bazelor de date", Editura Tehnica, Bucuresti, 2000
6. Felea V. : "Elemente ale implementarii modelului relational in sisteme de gestiune de baze de date". Ed.MatrixROM, 2007
7. Felea V., Matei C. si Balta M.: "Interogarea bazelor de date. Aplicatii in Oracle si SQL Server", Ed.MatrixROM, 2005

Referințe electronice:

- <http://thor.info.uaic.ro/~felea/>
- Documentatia produselor Oracle



Câteva precizări

1) Hub-ul colaborativ Microsoft Teams: cursul **FI-3-Tehnici de programare cu baze de date-2024/2025** va conține toate cursurile și laboratoarele.



2) Suport curs - varianta electronică disponibilă pe site-ul: www.runceanu.ro/adrian

Notă: Actualizarea site-ului se face săptămânal.

Tehnici de programare cu baze de date

Concepte generale

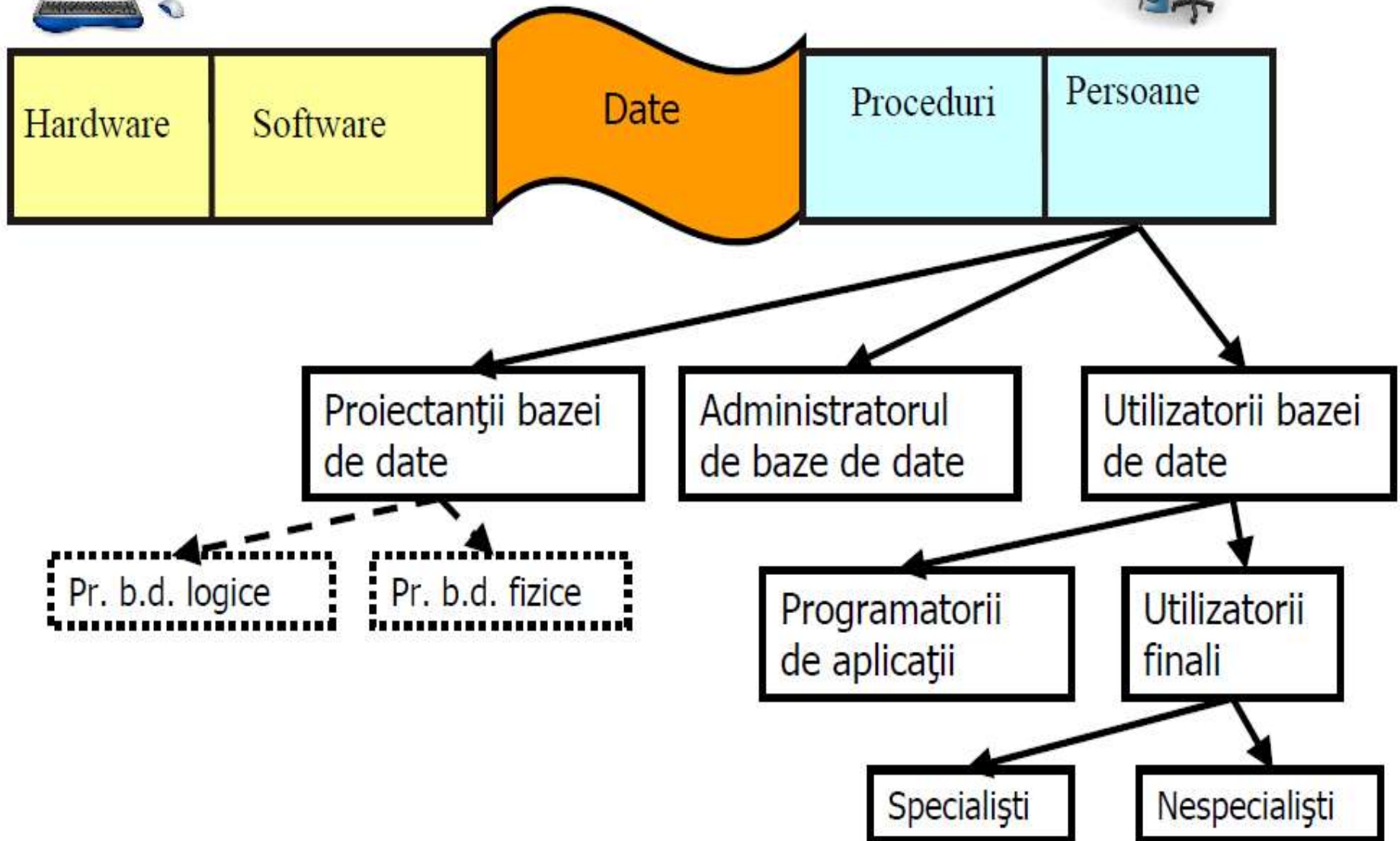
GENERALITĂȚI DESPRE BAZE DE DATE

- **Baza de date** este un ansamblu structurat de date coerente, fără redundanță inutilă, astfel încât acestea pot fi prelucrate eficient de mai mulți utilizatori într-un mod concurent.



GENERALITĂȚI DESPRE BAZE DE DATE

- Un **sistem de gestiune al bazei de date** (SGBD – *Data Base Management System*) este un *produs software* cu elemente hardware, care asigură interacțiunea cu o bază de date, permițând *definirea*, *consultarea* și *actualizarea datelor* din baza de date.



PL/SQL



- *Procedural Language/Structured Query Language (PL/SQL)* este extensia **procedurală** a limbajului SQL.
- **PL/SQL** este un limbaj de programare sofisticat care asigură accesarea datelor unei **baze de date relaționale orientate obiect** și permite gruparea unei mulțimi de comenzi într-un **bloc** unic de tratare a datelor.
- Programul este format din unul sau mai multe blocuri care pot conține blocuri imbricate.

PL/SQL include:

1. instrucțiuni *SQL* pentru manipularea datelor și pentru gestiunea tranzacțiilor
 2. instrucțiuni proprii
- Limbajul combină construcțiile procedurale ale unui limbaj de generația 3 (*LG3*) cu puterea și flexibilitatea lui *SQL* (*LG4*).
 - Combinația a generat un limbaj puternic pentru modelarea aplicațiilor complexe.

PL/SQL extinde **SQL** prin construcții specifice limbajelor procedurale:

- definirea variabilelor
- declararea tipurilor
- utilizarea structurilor de control
- implementarea procedurilor și funcțiilor
- introducerea tipurilor obiect și a metodelor, etc.

PL/SQL oferă posibilități moderne de tratare a informației:

- încapsularea datelor
- analiza specială a erorilor
- mascarea informației
- orientarea obiect

Posibilitățile lui **SQL** sunt folosite pentru un acces rafinat la date, iar facilitățile oferite de **PL/SQL** sunt folosite pentru fluxul controlului procesării datelor.

The logo for PL/SQL programming. The text "PL/SQL" is written in a large, bold, red, serif font. Below it, the word "programming" is written in a smaller, black, lowercase, serif font. The entire logo has a slight drop shadow effect.

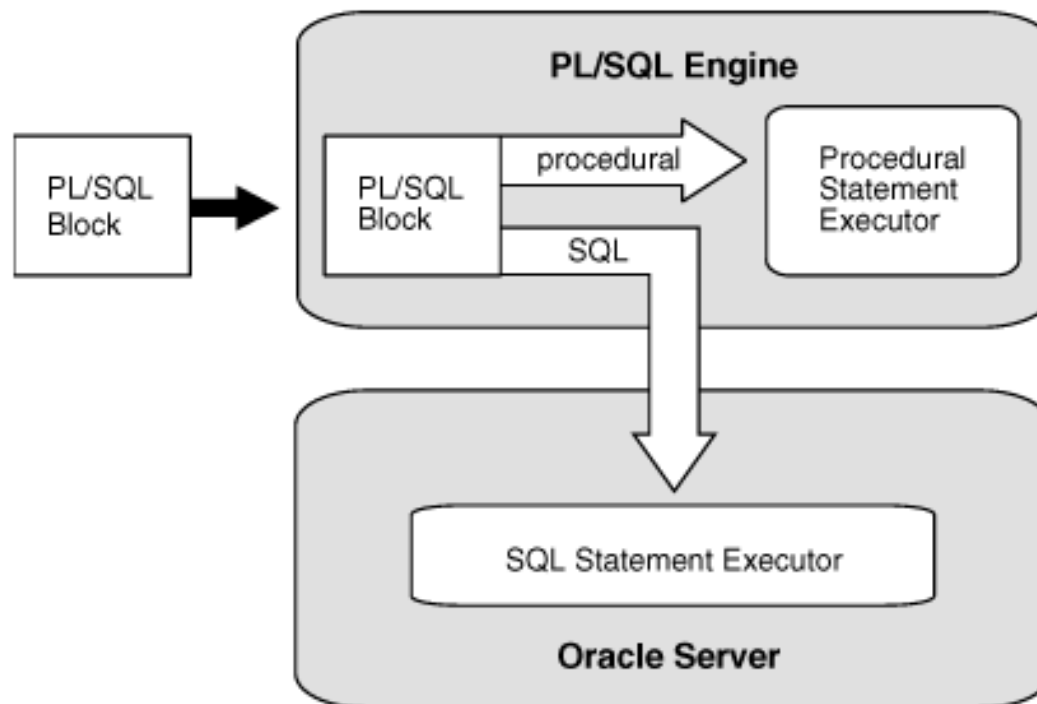
Dintre funcționalitățile limbajului **PL/SQL** care determină ca acesta să fie frecvent utilizat se remarcă următoarele facilități:

1. integrarea comenzilor **SQL** de bază
2. integrarea cu server-ul **Oracle** și cu aplicații **Oracle**
3. oferirea unui suport pentru programarea orientată obiect
4. asigurarea securității informației
5. definirea și gestiunea blocurilor de instrucțiuni

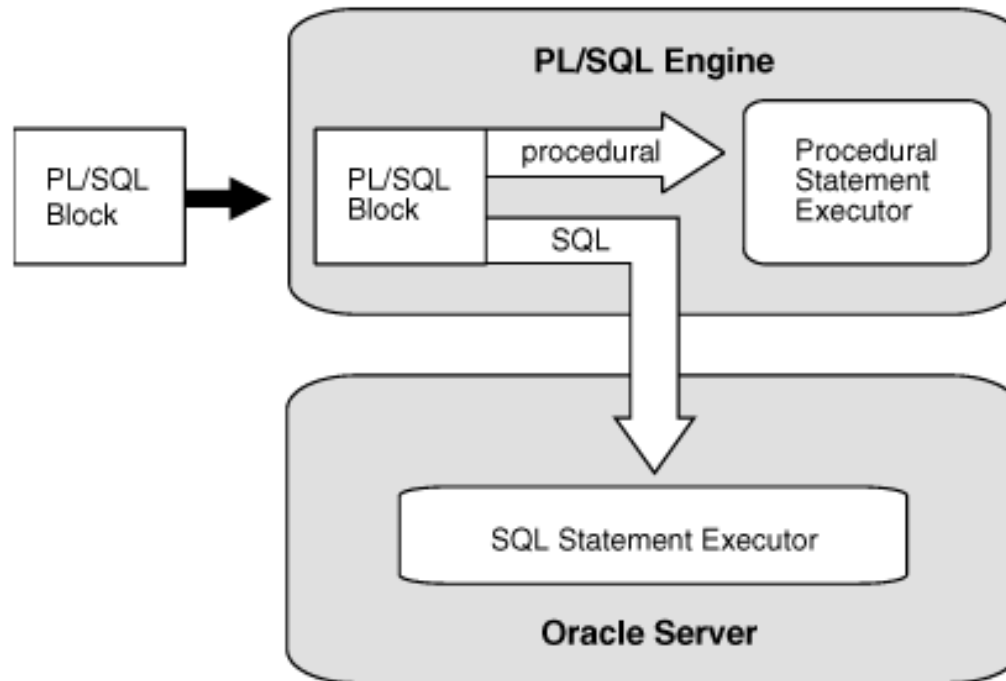
6. gestiunea variabilelor, constantelor și a cursoarelor
7. modularizarea programelor (subprograme, pachete)
8. implementarea și utilizarea declanșatorilor
9. utilizarea structurilor de control fundamentale
10. detectarea și gestiunea erorilor de execuție și a situațiilor excepționale
11. dezvoltarea de *aplicații Web*

- *PL/SQL* este o tehnologie utilizată de *server-ul Oracle* și de anumite *aplicatii Oracle*.
- Blocurile *PL/SQL* sunt transmise unui **motor *PL/SQL*** și procesate (compilate și executate) de acesta.
- *Motorul PL/SQL* poate să se afle pe *server-ul Oracle* sau într-un utilitar, iar utilizarea sa depinde de unde se invocă *PL/SQL*.
- Multe *aplicatii Oracle* (inclusiv *Developer/2000*) au propriul lor *motor PL/SQL* care este independent de motorul prezent pe *server-ul Oracle*.

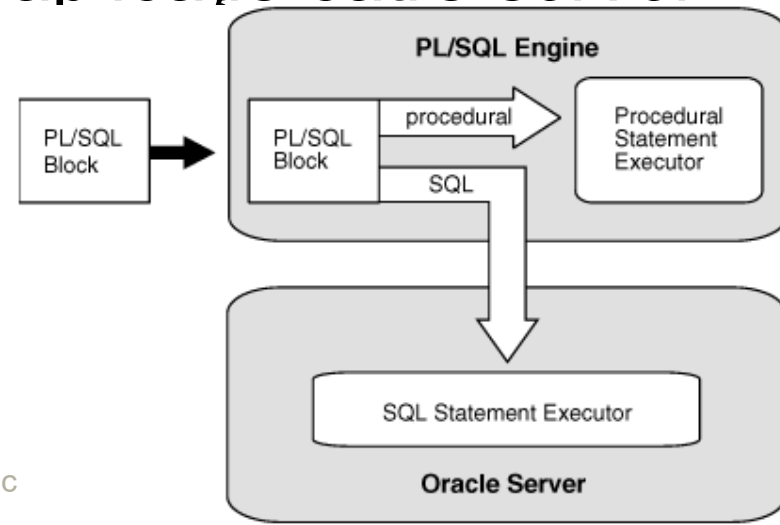
- Blocurile *PL/SQL* pot fi executate:
 1. pe stația *client* fără interacțiune cu *server*-ul
 2. sau în întregime pe *server*.
- Când blocurile *PL/SQL* sunt referite dintr-un program *PRO**, din *iSQL*Plus*, sau de către *Server Manager*, motorul *PL/SQL* de pe *server-ul Oracle* va procesa aceste blocuri.



- Acesta descompune blocul în instrucțiuni SQL și le trimite executorului de instrucțiuni SQL (*SQL Statement Executor*) de pe **server-ul Oracle**.
- Fără *PL/SQL*, instrucțiunile SQL ar fi procesate separat, fiecare la un moment dat, fiecare implicând un apel la **server-ul Oracle**.



- Restul comenzilor (procedurale) sunt procesate de către executorul instrucțiunilor procedurale (*PSE – Procedural Statement Executor*) care este în motorul *PL/SQL*.
- *PSE* procesează datele care sunt locale aplicației, reducându-se astfel activitatea de transfer spre *server-ul Oracle* și numărul de cursoare solicitate.
- În felul acesta, este necesar un singur transfer pentru a trimite blocul din aplicație către *server*.



O aplicație bază de date poate fi structurată în trei părți:

1. **interfața utilizator** (utilizatorul introduce anumite informații și obține niște rezultate în urma executării aplicației)
2. **aplicația logică efectivă**
3. **baza de date**

Există două modele pentru proiectarea unei aplicații bază de date:

1. modelul *client-server (two-tier)*
2. modelul *three-tier*

1. Modelul *client-server*

- Multe dintre aplicațiile baze de date sunt construite folosind modelul clasic *client-server*, descris succint anterior pentru *PL/SQL*.
- Modelul este caracterizat de cele două componente:
 1. *client*
 2. *server*
- *Client*-ul folosește interfața,
- iar *server*-ul conține baza de date.

1. Modelul *client-server*

- Aplicația logică este scindată între *client* și *server*.
- De remarcat din *această caracteristică fundamentală a modelului că aplicația comunică direct cu server-ul*.
- Există un motor *PL/SQL* pe *server*, iar în anumite cazuri și pe *client*.

- Dacă motorul *PL/SQL* este pe *server*, atunci aplicația (care poate fi scrisă în *Pro*C*, *JDBC*, *OCI* sau alte limbaje) care rezidă pe *client* trimite cereri la un *server* de date.
- Cererile sunt rezolvate utilizând *SQL*.
- Diferite cereri *SQL* pot fi grupate într-un bloc *PL/SQL* și trimise ca o singură entitate *server*-ului.

2. Modelul *three-tier*

În modelul *three-tier* interfața utilizator, aplicația logică și baza de date sunt scindate în trei părți separate:

1. *Client Browser*
2. *Application Server (Oracle Internet Application Server – IAS)*
3. *Oracle Database Server*

- Clientul (cel mai frecvent un *browser*) apelează subprograme care generează ca rezultat pagini *HTML*, iar *Application Server* procesează solicitările.
- În *Oracle9i*, aplicațiile *Forms* și *Reports* se execută numai ca parte a unui model *three-tier*.
- De exemplu, *Oracle Web Forms* permite desfășurarea unei aplicații într-un mediu *multi-tier Internet* fără a modifica o linie de cod.

- În general, comenzile *SQL* și *PL/SQL* sunt trimise *server-ului* pentru a fi executate.
- Pentru a realiza acest lucru, trebuie stabilită **conectarea la BD**.
- Aceasta presupune ca baza să **autentifice utilizatorul** (parolă și identificator).
- *PL/SQL* nu conține o sintaxă pentru realizarea conectării la baza de date.
- Conectarea poate fi realizată de alte componente ale sistemului (de exemplu, *iSQL*Plus*).

- *PL/SQL* are un rol important atât la nivelul *server*-ului *Oracle* (prin subprograme stocate, declanșatori și pachete), cât și la nivelul utilitărelor *Oracle* (de exemplu, *Developer/2000* – componenta declanșatori).
- Partea procedurală este atât la nivel de *client*, cât și la nivel de *server*.
- Cererile însă nu se execută pe stații *client*, ci numai pe *server*.

PL/SQL constituie fundamentul pentru realizarea diverselor aplicații:

- fișiere *iSQL*Plus* (*script file*);
- proceduri și funcții stocate (*stored procedure*):
 1. o procedură sau o funcție stocată este un subprogram *PL/SQL* care poate fi invocat de o aplicație *client*
 2. un declanșator BD
 3. un declanșator aplicație

1. Declanșatori bază de date (*database trigger*)

- un declanșator bază de date este un bloc *PL/SQL* care se execută la apariția unui eveniment:

- modificarea unui tabel al bazei
- modificarea unei vizualizări
- anumite acțiuni sistem
- sau chiar anumite acțiuni ale utilizatorului

2. Pachete (*package*): un pachet este un bloc *PL/SQL* care încapsulează într-o unitate logică în baza de date o mulțime de proceduri, funcții, variabile, constante, tipuri, cursoare și excepții;

3. Declanșatori aplicație (*application trigger*) - un declanșator aplicație este un bloc *PL/SQL* care se execută în urma unui eveniment provocat de sistem.

- Aplicațiile *Oracle (Oracle Forms și Oracle Reports)* sunt dotate cu motoare *PL/SQL* care permit construirea acestor declanșatori.

Portabilitatea PL/SQL

Referitor la portabilitatea limbajului, pot fi remarcate două aspecte:

1. Aplicațiile *PL/SQL* se pot executa pe orice platformă sau sistem de operare pe care poate fi executat *Oracle*.
2. *PL/SQL* permite transfer de cod între *server-ul Oracle* și aplicații. Pot fi scrise pachete program portabile și crea biblioteci ce pot fi utilizate în diferite situații, de diverse aplicații.

Controlul execuției unui bloc *PL/SQL*

PL/SQL este un limbaj cu structură de bloc, adică programele sunt compuse din blocuri care pot fi complet separate sau imbricate.

- Structura unui bloc poate fi obținută combinând subprograme, pachete, blocuri imbricate.
- Blocurile pot fi folosite în utilitarele *Oracle*.

Pentru modularizarea unui program este necesară:

1. gruparea logică a instrucțiunilor în blocuri
2. imbricarea de subblocuri în blocuri mai mari
3. descompunerea unei probleme complexe într-o mulțime de module logice și implementarea acestora cu ajutorul blocurilor
4. plasarea în biblioteci a codului *PL/SQL* reutilizabil, de unde poate fi folosit de aplicații
5. depunerea codului într-un *server Oracle*, de unde este accesibil oricărei aplicații care interacționează cu baza de date *Oracle*

Un program *PL/SQL* poate cuprinde unul sau mai multe blocuri.

Un bloc poate fi:

1. anonim
2. neanonim

Block Types

Anonymous

```
[DECLARE]

BEGIN
  --statements

[EXCEPTION]

END;
```

Procedure

```
PROCEDURE name
IS
BEGIN
  --statements

[EXCEPTION]

END;
```

Function

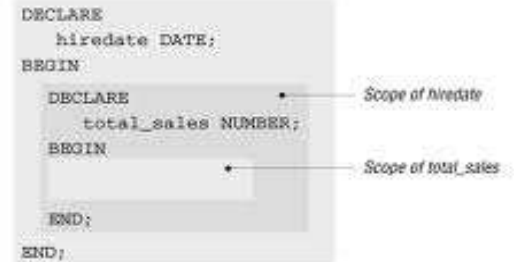
```
FUNCTION name
RETURN datatype
IS
BEGIN
  --statements
  RETURN value;

[EXCEPTION]

END;
```

Blocuri anonime

```
DECLARE
  hiredate DATE;
BEGIN
  DECLARE
    total_sales NUMBER;
  BEGIN
  END;
END;
```



1. **Blocurile anonime** sunt blocuri *PL/SQL* fără nume, care sunt construite dinamic și sunt executate o singură dată.

- Acest tip de bloc nu are argumente și nu returnează un rezultat.
- Ele sunt declarate într-un punct al aplicației, unde vor fi executate (trimise motorului *PL/SQL*).
- În blocurile anonime pot fi declarate proceduri și funcții *PL/SQL*.

Blocuri neanonime

2. Blocurile **neanonime** sunt:

- fie blocuri cu nume (etichetate) construite static sau dinamic și executate o singură dată,
- fie **subprograme**, **pachete** sau **declanșatori**.

1. Subprogramele sunt **proceduri** sau **funcții** depuse în baza de date.

- Aceste blocuri sunt executate de mai multe ori și, în general, nu mai sunt modificate după ce au fost construite.
- Procedurile și funcțiile stocate sunt depuse pe **server-ul Oracle**, acceptă parametri și pot fi apelate prin nume.

Block Types

Anonymous	Procedure	Function
<pre>[DECLARE] BEGIN --statements [EXCEPTION] END;</pre>	<pre>PROCEDURE name IS BEGIN --statements [EXCEPTION] END;</pre>	<pre>FUNCTION name RETURN datatype IS BEGIN --statements RETURN value; [EXCEPTION] END;</pre>

2. Pachetele (stocate sau aplicație)
sunt blocuri neanonime care grupează:

1. proceduri
2. funcții
3. cursoare
4. tipuri
5. constante
6. variabile - într-o unitate logică, în baza de date

3. Declanșatorii sunt blocuri *PL/SQL* neanonime depuse în baza de date, care pot fi asociați bazei, iar în acest caz sunt executați implicit ori de câte ori apare un anumit eveniment declanșator:

The slide, titled "Triggers...", contains the following content:

- Code tied to a table
- Executes automatically when DML executes

```

SQL> @trigger1
SQL> create trigger customer_name
2:  before update or insert on customer
3:  for each row
4:  begin
5:  /* convert character values to upper case */
6:  new.lastname := upper(new.lastname);
7:  new.firstname := upper(new.firstname);
8:  dbms_output.put_line('trigger fired');
9:  end;
10 /
Trigger created.
  
```

Callouts on the slide:

- Trigger tied to CUSTOMER table (points to the table name in the code)
- Use the UPPER function to convert characters inserted into the database (points to the UPPER function in the code)

© 2008 OracleBrowsers, Inc. | www.OracleBrowsers.com

- De exemplu, instrucțiuni **INSERT**, **UPDATE** sau **DELETE** ce se execută asupra unui tabel al bazei de date
- sau pot fi asociați unei aplicații (de exemplu, declanșator *SQL*Forms*), ceea ce presupune că se execută automat, în funcție de anumite condiții sistem.

Triggers...

- > Code tied to a table
- > Executes automatically when DML executes

```
SQL> #trigger1
SQL> create trigger customer_name
2  before update or insert on customer
3  for each row
4  begin
5  /* convert character values to upper case */
6  :new.lastname := upper(:new.lastname);
7  :new.firstname := upper(:new.firstname);
8  dbms_output.put_line('trigger fired');
9  end;
10 /

Trigger created.
```

Trigger tied to CUSTOMER table

Use the UPPER function to convert characters inserted into the database

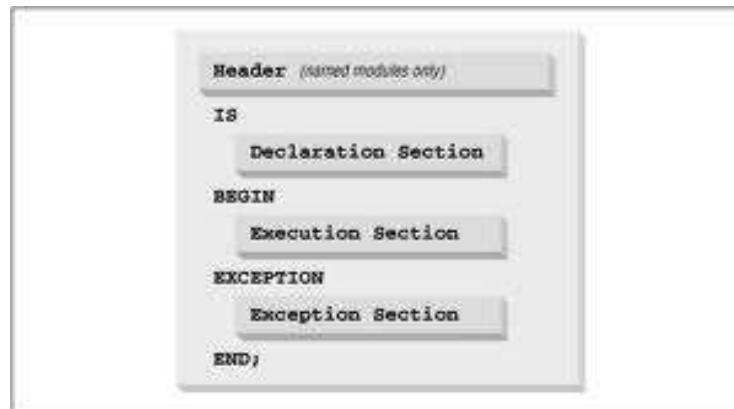
© 2008 OracleBrowsers, Inc. | www.OracleBrowsers.com

Structura unui bloc *PL/SQL*

Un bloc *PL/SQL* este compus din trei secțiuni distincte:

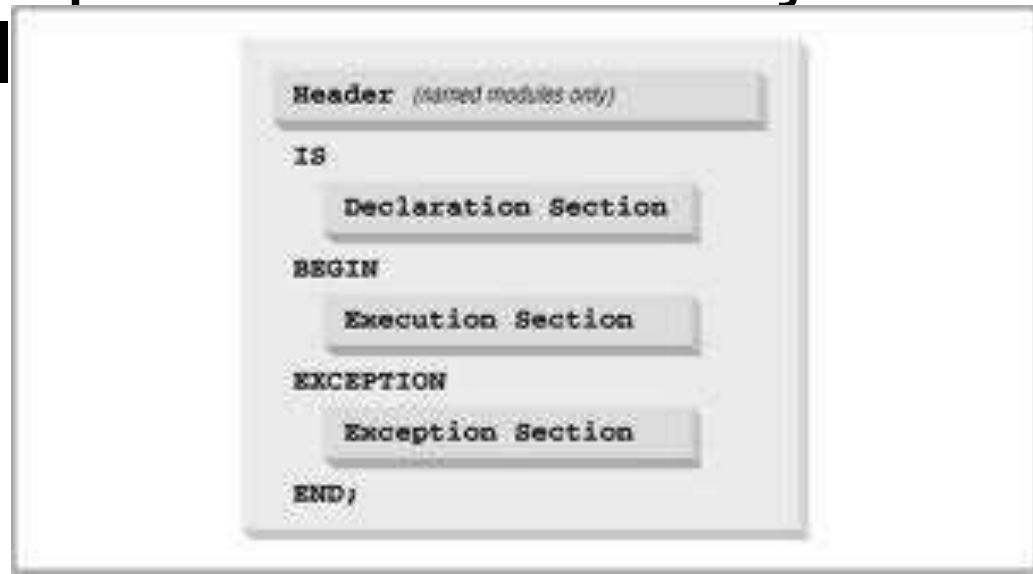
1. **Secțiunea declarativă (opțională)** conține declarații pentru toate variabilele, constantele, cursoarele și erorile definite de utilizator la care se face referință în secțiunea executabilă sau chiar în cea declarativă.

- De asemenea, pot fi declarate subprograme locale care sunt vizibile doar în blocul respectiv.



2. Secțiunea executabilă conține instrucțiuni neprocedurale SQL pentru prelucrarea datelor din baza de date și instrucțiuni *PL/SQL* pentru prelucrarea datelor în cadrul blocului.

3. Secțiunea pentru tratarea erorilor (opțională) specifică acțiunile ce vor fi efectuate atunci când în execuția blocului apar erori sau condiții anormal



Structura unui bloc PL/SQL

Blocul *PL/SQL* are următoarea structură generală:

[*<<nume_bloc>>*]

[**DECLARE**

instrucțiuni de declarare]

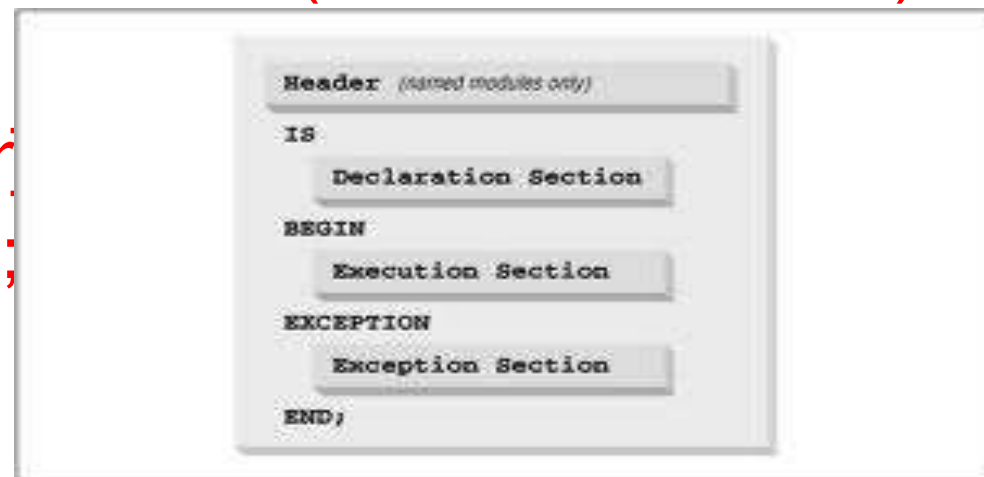
BEGIN

instrucțiuni executabile (SQL sau PL/SQL)

[**EXCEPTION**

tratarea erorilor]

END [*nume_bloc*];



Dacă blocul *PL/SQL* este executat fără erori, invariant va apărea mesajul:

PL/SQL procedure successfully completed

Compatibilitate cu SQL

Din punct de vedere al compatibilității dintre *PL/SQL* și *SQL*, se remarcă următoarele reguli de bază:

- *PL/SQL* furnizează:
 - toate comenzile *LMD* (*limbajul de manipulare a datelor*) ale lui *SQL*
 - comanda *SELECT* cu clauza *INTO*
 - comenzile *LCD* (*limbajul de control al datelor*)
 - funcțiile
 - pseudocoloanele
 - și operatorii *SQL*
- *PL/SQL* nu furnizează comenzile *LDD* (*limbajul de definire a datelor*).

- Majoritatea funcțiilor SQL sunt disponibile în *PL/SQL*.
- Există însă funcții specifice *PL/SQL*, cum sunt funcțiile *SQLCODE* și *SQLERRM*.
- De asemenea, există funcții SQL care nu sunt disponibile în instrucțiuni procedurale (*DECODE*, funcțiile grup), dar care sunt disponibile în instrucțiunile SQL dintr-un bloc *PL/SQL*.
- SQL nu poate folosi funcții sau atribute specifice *PL/SQL*.

- Funcțiile de grup trebuie folosite cu atenție, deoarece clauza **GROUP BY** nu are sens să apară în instrucțiunea **SELECT ... INTO**.
- *Oracle9i* introduce clauza **OVER**, care permite ca funcția grup căreia îi este asociată să fie considerată o funcție analitică (poate returna mai multe linii pentru fiecare grup).

Următoarele funcții **SQL** nu sunt permise
în **PL/SQL**:

1. *WIDTH_BUCKET*
2. *BIN_TO_NUM*
3. *COMPOSE*
4. *DECOMPOSE*
5. *TO_LOB*
6. *DECODE*
7. *DUMP*
8. *EXISTSNODE*
9. *TREAT*
10. *NULLIF*
11. *SYS_CONNECT_BY_PATH*
12. *SYS_DBURIGEN*
13. *EXTRACT*

Întrebări?