

DIMA NARCISA

# APLICAȚII PL/SQL ÎN ORACLE APEX



EDITURA  
EVOMIND

# APLICAȚII PL/SQL ÎN ORACLE APEX

*Copyright* © 2020  
Autor: DIMA NARCISA

*Toate drepturile rezervate.*

ISBN 978-606-94690-9-5

*Editura Evomind, 2020*

*<https://evomind.org/>*

## CUPRINS

1. CREAREA UNEI APLICAȚII ÎN ORACLE APEX 18.2 .....	4
2. ACTUALIZAREA TABELELOR PRIN ACTIUNI DINAMICE IN ORACLE APEX 19.1 .....	12
3. TRATAREA EXCEPTIILOR IN ORACLE PL/SQL .....	17
4. CURSORI ÎN ORACLE PL/SQL .....	20
5. ROW TRIGGERS IN ORACLE APEX .....	26
6. GRAFICE ȘI PROCESE ÎN APLICAȚII ORACLE APEX .....	34
7. FOLOSIREA LISTELOR DE VALORI SI A PROCESELOR IN APLICATII ORACLE APEX .....	47
8. ITEMI CU ACȚIUNE DINAMICĂ ÎN ORACLE APEX .....	55
BIBLIOGRAFIE .....	65

## 1. CREAREA UNEI APLICAȚII ÎN ORACLE APEX 18.2

În cele ce urmează, vom crea o aplicație cu baze de date, folosind versiunea 18.2 a utilitarului ORACLE APEX, lansată în septembrie 2018. Vom considera, spre exemplificare, crearea unei aplicații în care să ținem evidența medicamentelor dintr-o farmacie. Vom crea patru tabele în care vom reține denumirea, categoria, producătorii, precum și evidența stocului pentru fiecare medicament în parte, după cum urmează:

### Tabela **Medicamente**

```
create table "medicamente"  
(  
    "cod" number not null enable,  
    "medicament" varchar2(40) not null enable,  
    "grupa_terapeutica" number,  
    constraint "medicamente_pk" primary key ("cod")  
using index enable  
)  
alter table "medicamente" add constraint "medicamente_fk" foreign key ("grupa_terapeutica")  
    references "grupe_terapeutice" ("cod") on delete cascade enable  
create or replace editionable trigger "bi_medicamente"  
before insert on "medicamente"  
for each row  
begin  
    if :new."cod" is null then  
        select "medicamente_seq".nextval into :new."cod" from sys.dual;  
    end if;  
end;  
alter trigger "bi_medicamente" enable
```

Tabela **Grupe\_Terapeutice** în care vom reține grupele terapeutice din care fac parte medicamentele:

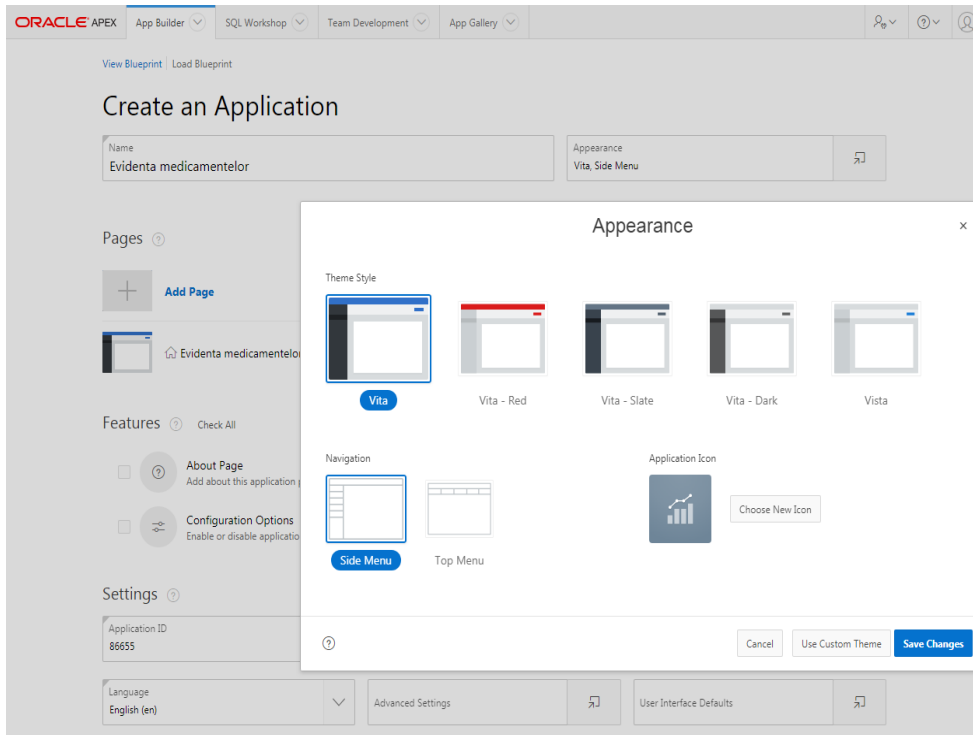
```
create table "grupe_terapeutice"  
  
  (  
    "cod" number not null enable,  
    "grupa_terapeutica" varchar2(40) not null enable,  
    constraint "grupe_terapeutice_pk" primary key  
("cod")  
  using index enable,  
    constraint "grupe_terapeutice_uk1" unique  
("grupa_terapeutica")  
  using index enable  
  )  
create or replace editionable trigger "bi_grupe_terapeutice"  
  before insert on "grupe_terapeutice"  
  for each row  
begin  
  if :new."cod" is null then  
    select "grupe_terapeutice_seq".nextval into :new."cod"  
  from sys.dual;  
  end if;  
end;  
alter trigger "bi_grupe_terapeutice" enable
```

Tabela **Firme** în care reținem denumirea firmelor producătoare de medicamente

```
create table "firme"  
  (    "cod" number not null enable,  
    "firma" varchar2(20) not null enable,  
    constraint "firme_pk" primary key ("cod")  
  using index enable,  
    constraint "firme_uk1" unique ("firma")  
  using index enable  
  )  
create or replace editionable trigger "bi_firme"  
  before insert on "firme"  
  for each row  
begin  
  if :new."cod" is null then  
    select "firme_seq".nextval into :new."cod" from sys.dual;  
  end if;  
end;  
  
alter trigger "bi_firme" enable
```

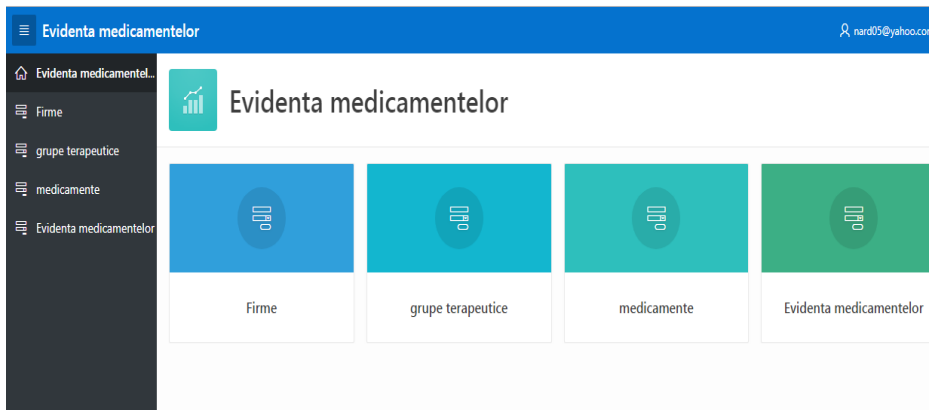
Crearea aplicației presupune parcurgerea următorilor pași:

*Alegerea interfeței grafice, denumirea aplicației*



**Figura 1.1** Alegerea interfeței grafice, denumirea aplicației

*Lansarea în execuție a aplicației:*



**Figura 1.2 Lansarea în execuție a aplicației**



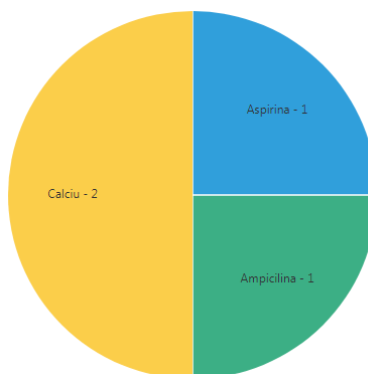
### Crearea listelor de valori și atașarea acestora la itemi

The screenshot displays the Oracle APEX Builder interface for editing a page item. The left sidebar shows the page structure for 'Page 7: Medicamente', including regions like 'Content Body' and 'Buttons', and page items like 'PT\_COD' and 'PT\_MEDICAMENT'. The main workspace is divided into two panels: 'Page Rendering' and 'Page Processing'. The 'Page Rendering' panel shows the layout of the page item 'PT\_GRUPA\_TERAPEUTICA' as a 'Group Therap...' with a 'Select List' type. The 'Page Processing' panel shows the configuration for the 'List of Values' component, including a dropdown menu with 'GRUPA TERAPEUTIC' selected and 'LOGIN\_REMEMBER\_USERNAME' as the parent item. The right sidebar shows the configuration for the 'List of Values' component, including a dropdown menu with 'GRUPA TERAPEUTIC' selected and 'LOGIN\_REMEMBER\_USERNAME' as the parent item.

**Figura 1.3** Crearea listelor de valori și atașarea acestora la itemi

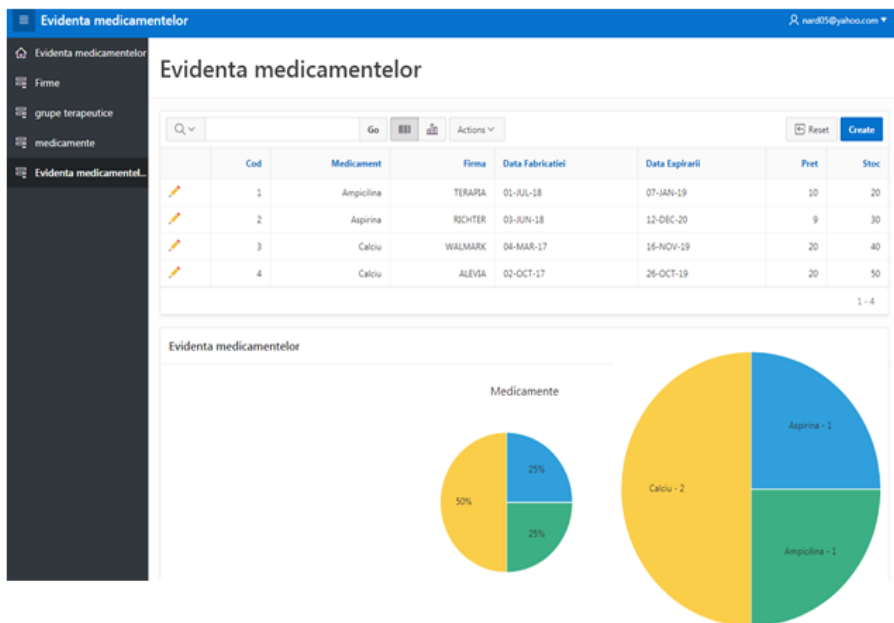
### Crearea unui grafic în aplicație

The screenshot displays the Oracle APEX Page Editor interface for 'Page 8: Evidenta medicamentelor'. The left-hand 'Regions' tree is expanded to show the configuration for the 'Evidenta medicamentelor' region, which is set to be a 'Chart'. The 'Columns' list includes: COD, MEDICAMENT, FIRMA, DATA\_FABRICATIEI, DATA\_EXPIRARII, PRET, and STOC. The right-hand 'Page Rendering' pane shows a summary of the page layout, with the 'Evidenta medicamentelor' region highlighted as a 'Chart'.



**Fig. 1.4 Crearea unui grafic în aplicație**

*Raport cu grafic atașat*



**Figura 1.5** *Raport cu grafic atașat*

## 2. ACTUALIZAREA TABELOR PRIN ACTIUNI DINAMICE IN ORACLE APEX 19.1

În cele ce urmează, ne propunem să actualizăm o tabelă prin intermediul unei acțiuni dinamice ce se va executa la crearea unei înregistrări în altă tabelă. Spre exemplificare, vom crea o aplicație ce va ține evidența vânzărilor dintr-o librărie. Tabela *Evidențe* reține evidența stocului existent pentru fiecare carte prezentă în librărie, iar tabela *Vânzări* reține vânzările efectuate. Ne propunem ca la introducerea unei vânzări, să se actualizeze automat stocul cărții respective din tabela *Evidențe*. Pentru aceasta, vom crea o acțiune dinamică ce se va executa la introducerea unei vânzări.

<pre> Cazul TRUE: declare s number; BEGIN select stoc into s from evidente where cod=:P13_EVIDENTA; IF (s&gt;=:P13_EXEMPLARE and :P13_EXEMPLARE&gt;0) THEN BEGIN update evidente set stoc= stoc- :P13_EXEMPLARE where cod= :P13_EVIDENTA; </pre>	<pre> :P13_REZULTAT:='VANZARE EFECTUATA'; END; END IF; :P13_AUTORUL:=NULL; :P13_EVIDENTA:=NULL; END; Cazul False: begin :P13_REZULTAT:='NU SE POATE EFECTUA VANZAREA'; end; </pre>
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

The screenshot displays the Oracle APEX Page Designer interface for Application 79680, Page Designer. The main workspace shows a form layout with various regions and items. A 'CREATE' button is highlighted, and its dynamic action configuration is shown in the right-hand pane.

**Dynamic Action Configuration:**

- Identification:** Action: Execute PL/SQL Code
- Settings:**
  - PL/SQL Code:**

```
declare
s number;
BEGIN
select stoc into s from evidentelib12b
where cod=:P15_EVIDENTA;
IF (s)>=:P15_EXEMPLARE and :P15_EXEMPLARE=0)
THEN
BEGIN
update evidentelib12b
set stoc= stoc- :P15_EXEMPLARE
where cod= :P15_EVIDENTA;
```
  - Items to Submit:** P15\_EVIDENTA,P15\_EXEMPLARE
  - Items to Return:** P15\_REZULTAT,P15\_EVIDENTA,P...
  - Suppress Change Event:** Yes
- Execution Options:**
  - Sequence:** 10
  - Event:** efectuare vanzare
  - Fire When Event Result Is:** True
  - Fire on Initialization:** Yes

**Figura 2.1** Acțiune dinamică atașată butonului CREATE din formularul VÂNZĂRI

	Carte ↑	Autor	Editura	Domeniu	Anapar	Pret	Stoc
	Aminții din copilărie	Ion Creanga	Carminis	beletristica	1,980	15	18
	Emotiile	Osho	For You	dezvoltare personala	1,996	35	6
	Poezii	George Cosbuc	Carminis	beletristica	2,010	24	23
	Poezii	Mihai Eminescu	Carminis	beletristica	2,005	20	21
	Poezii	Mihai Eminescu	Paralela 45	beletristica	2,011	25	25
	Proza	Mihai Eminescu	Paralela 45	beletristica	2,006	35	38

1 - 6

**Figura 2.2 Tabela Evidențe**

**Vanzarilib12b**

SELECȚATI AUTORUL DORIT  
Ion Creanga

SELECȚATI CODUL CARTII DORIT  
1

Data  
05-OCT-19

INTRODUCETI NUMARUL DE EXEMPLARE DORIT  
2

NUMARUL BONUSULUI  
81

PREȚUL TOTAL AL VANZĂRII  
30

Rezultat  
Stoc suficient! Se poate da comanda!

Go

Actions

Cod	Carte	Autor	Editura	Domeniul	Anapar	Pret	Stoc
1	Amintiri din copilărie	Ion Creanga	Carminis	beletristica	1980	15	18

Cancel Efectuare vânzare

**Figura 2.3** Introducerea unei vânzări prin intermediul formularului **Vânzări**

The screenshot shows the 'Vanzari' (Sales) report in an Oracle APEX application. The interface includes a sidebar with navigation options like 'Librerie', 'Autori', 'Carti', 'Domenii', 'Edituri', 'Bonuri', 'Evidente', 'Vanzari', 'Grafice', and 'Administration'. The main content area displays a table with columns for 'Evidenta', 'Data', 'Exemplare', 'Bon', and 'Prettotal'. Below this, there is a search bar and a detailed table with columns for 'Cod', 'Carte', 'Autor', 'Data Vanzarii', 'Exemplare Vandute', 'Bon', 'Pret Total Vanzare', 'Anapar', 'Pret Unitar', 'Stoc', and 'Editura'.

	Evidenta ↑	Data	Exemplare	Bon	Prettotal
	1	05-OCT-19	2	81	30
	21	11-FEB-19	2	21	70
	21	18-FEB-19	2	41	70
	42	11-FEB-19	2	1	70
	43	18-FEB-19	3	41	72
	43	11-FEB-19	2	1	48
	43	02-OCT-19	2	61	48
	44	11-FEB-19	2	1	50
	44	11-FEB-19	3	21	75

Cod	Carte	Autor	Data Vanzarii	Exemplare Vandute	Bon	Pret Total Vanzare	Anapar	Pret Unitar	Stoc	Editura
1	Proza	Mihai Eminescu	11-FEB-19	2	1	70	2006	35	38	Paralela 45
21	Poezii	Mihai Eminescu	11-FEB-19	2	1	50	2011	25	25	Paralela 45
23	Poezii	Mihai Eminescu	11-FEB-19	3	21	75	2011	25	25	Paralela 45
24	Emotile	Osho	11-FEB-19	2	21	70	1996	35	6	For You
61	Poezii	George Cosbuc	18-FEB-19	3	41	72	2010	24	23	Carminis
121	Aminții din copilărie	Ion Creanga	05-OCT-19	2	81	30	1980	15	16	Carminis

**Figura 2.4 Actualizarea automată a stocului pentru cartea vândută**



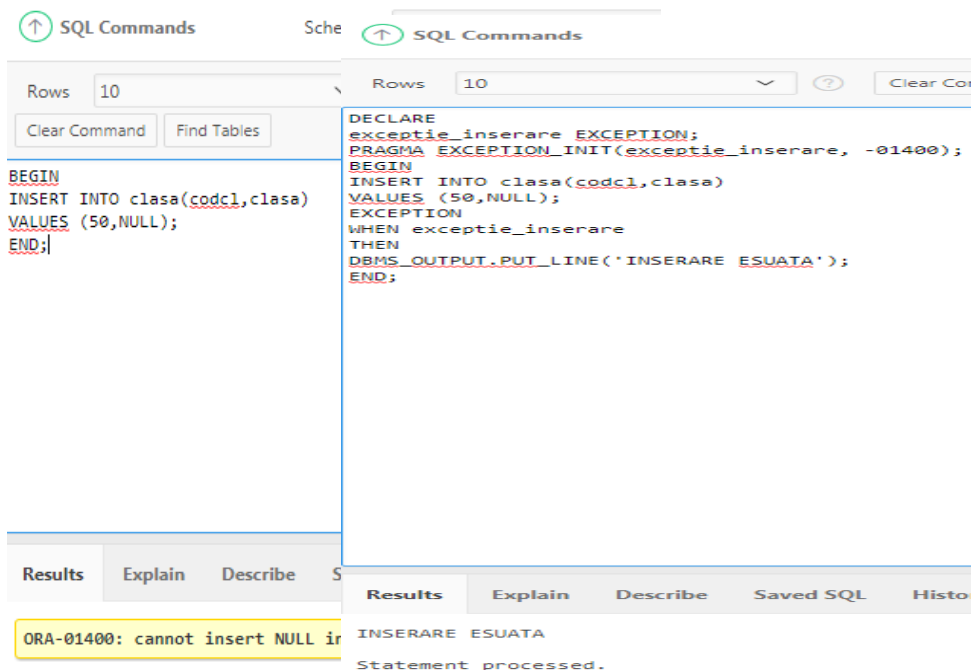
### 3. TRATAREA EXCEPȚIILOR ÎN ORACLE PL/SQL

Apariția unei erori în timpul execuției unui bloc de program în ORACLE PL/SQL, are ca efect întreruperea acestuia. Restabilirea execuției programului presupune tratarea erorii respective prin folosirea unei excepții. O *excepție* reprezintă un cod special ce se adaugă la finalul blocului de program unde se anticipează apariția erorii. În scrierea codului respectiv se va ține cont de tipul erorii ce ar putea să apară în acel bloc.

În funcție de erorile ce pot apărea în cadrul execuției unui bloc de program, există două tipuri de excepții: generate de server-ul ORACLE și cele definite de utilizator. Excepțiile generate de server-ul ORACLE pot fi la rândul lor de două tipuri: predefinite sau non-predefinite. Cele predefinite au un nume, un cod de eroare, un mesaj și se generează automat de server-ul ORACLE, neavând nevoie de o declarație explicită. Ca exemplu, în cazul unei comenzi SQL ce nu returnează nicio valoare, se generează excepția `NO_DATA_FOUND` asociată codului de eroare `ORA-01403`.

Spre deosebire de excepțiile predefinite, cele non-predefinite generate de server-ul ORACLE, nu au un nume. De aceea, acestea trebuie declarate explicit de către utilizator în secțiunea declarativă

a blocului de program, asociind numele excepției la codul erorii respective prin folosirea funcției PRAGMA EXCEPTION\_INIT. Ca exemplu, în *Figura 3.1* s-a generat o eroare non-predefinită de către server-ul ORACLE când s-a încercat introducerea valorii NULL în câmpul clasa ce nu admite astfel de valoare.



The screenshot shows the Oracle APEX SQL Commands interface. The left pane contains the following PL/SQL code:

```
BEGIN
INSERT INTO clasa(codcl,clasa)
VALUES (50,NULL);
END;
```

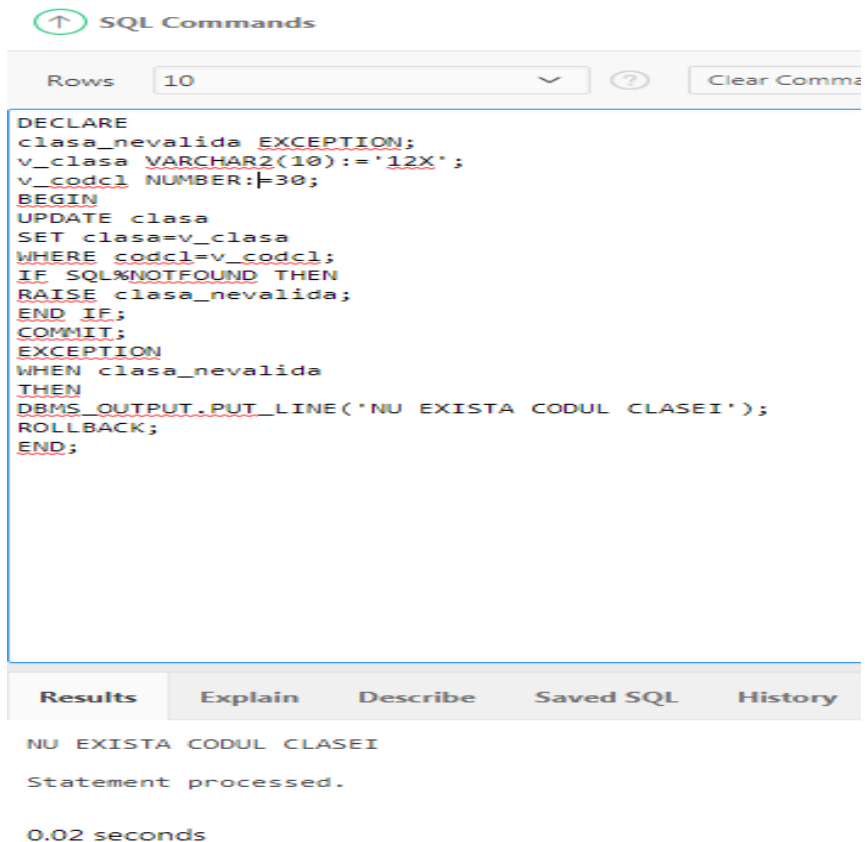
The right pane shows the execution results:

```
DECLARE
exceptie_inserare EXCEPTION;
PRAGMA EXCEPTION_INIT(exceptie_inserare, -01400);
BEGIN
INSERT INTO clasa(codcl,clasa)
VALUES (50,NULL);
EXCEPTION
WHEN exceptie_inserare
THEN
DBMS_OUTPUT.PUT_LINE('INSERARE ESUATA');
END;
```

The results pane shows the error message: **ORA-01400: cannot insert NULL in** (highlighted in yellow) and the output: **INSERARE ESUATA** and **Statement processed.**

**Fig. 3.1** Excepție non-predefinită generată de server-ul ORACLE

Excepțiile definite de utilizator se declară în secțiunea declarativă a blocului de program în care se anticipează că pot apărea erori la execuție și se lansează explicit prin folosirea cuvântului cheie RAISE, ca în *Figura 3.2*.



```
SQL Commands

Rows: 10

DECLARE
  clasa_nevalida EXCEPTION;
  v_clasa VARCHAR2(10):='12X';
  v_codcl NUMBER:=30;
BEGIN
  UPDATE clasa
  SET clasa=v_clasa
  WHERE codcl=v_codcl;
  IF SQL%NOTFOUND THEN
    RAISE clasa_nevalida;
  END IF;
  COMMIT;
EXCEPTION
  WHEN clasa_nevalida
  THEN
    DBMS_OUTPUT.PUT_LINE('NU EXISTA CODUL CLASEI');
  ROLLBACK;
END;
```

Results   Explain   Describe   Saved SQL   History

```
NU EXISTA CODUL CLASEI
Statement processed.
0.02 seconds
```

**Figura 3.2** Excepție definită de utilizator

#### 4. CURSORI ÎN ORACLE PL/SQL

Pentru afișarea mai multor rânduri de valori dintr-o tabelă prin intermediul unui bloc de program ORACLE PL/SQL, vom folosi un cursor explicit. Spre exemplificare, vom crea o aplicație ce va reține evidența cărților dintr-o bibliotecă(Figura 4.1).



**Figura 4.1 Aplicația Biblioteca**

Se vor crea două tabele, *carti\_t* și *editura\_t* în care se vor reține datele despre cărți, având structura prezentată în Figura 4.2 respectiv Figura 4.3.

Biblioteca /  
Carti

Carti

<input type="checkbox"/>	<b>Id</b>	<b>Titlu</b>	<b>Autor</b>	<b>Editura</b>
<input type="checkbox"/>	1	Matricea divina	Gregg Braden	For You ▾
<input type="checkbox"/>	2	Imageria inimii	Daniel Mitel	For You ▾
<input type="checkbox"/>	3	Cartea Secretelor	Deepak Chopra	For You ▾
<input type="checkbox"/>	4	Calculatoare personal	Radu Marsanu	All ▾
<input type="checkbox"/>	5	Poezii	Mihai Eminescu	All ▾
<input type="checkbox"/>	6	Zei subterani	Cristovam Buarque	Albatros ▾
<input type="checkbox"/>	7	Teste de analiza matei	Catalin-Petru Nicolescu	Albatros ▾

1 - 7

**Figura 4.2 Tabela Cărți\_t**

Biblioteca /  
Editura T

EDITURA\_T

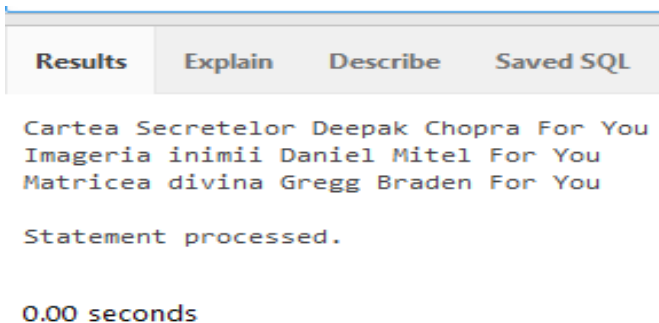
<input type="checkbox"/>	Editura 	Id
<input type="checkbox"/>	Albatros	2
<input type="checkbox"/>	All	1
<input type="checkbox"/>	For You	3

1 - 3

**Figura 4.3** Tabela *Editura\_t*

În continuare, dorim să afișăm toate cărțile unei edituri date, printr-un bloc de program PL/SQL. Pentru aceasta, vom crea un cursor explicit(Figura 4-din SQL Command și Fig 4.5-din aplicație).

```
DECLARE
CURSOR carti_t_cursor IS
  SELECT a.titlu, a.autor, b.editura from carti_t a, editura_t b
  where (a.editura=b.id) and (b.editura=:Editura);
v_carti_t_record carti_t_cursor%ROWTYPE;
BEGIN
  OPEN carti_t_cursor;
  LOOP
    FETCH carti_t_cursor INTO v_carti_t_record;
    EXIT WHEN carti_t_cursor%NOTFOUND;
    DBMS_OUTPUT.PUT_LINE(v_carti_t_record.titlu || ' ' ||
v_carti_t_record.autor ||'|'|v_carti_t_record.editura);
  END LOOP;
  CLOSE carti_t_cursor;
END;
```



Results	Explain	Describe	Saved SQL
Cartea Secretelor Deepak Chopra For You			
Imageria inimii Daniel Mitel For You			
Matricea divina Gregg Braden For You			

Statement processed.

0.00 seconds

**Figura 4.4 Cursor explicit**

În aplicație, pentru crearea cursorului explicit vom folosi aceeași comandă ca în Fig 4.4, cu deosebirea că la afișare vom utiliza comanda *htp.p()* în locul comenzii *dbms\_output.put\_line()*. Pentru alegerea editurii din lista editurilor disponibile în tabela *editura\_t*, vom crea un item de tip listă pe care îl vom asocia acestei tabele. La alegerea editurii din itemul respectiv, se vor afișa cărțile disponibile din acea editură (Figura 4.5).

```
DECLARE
CURSOR carti_t_cursor IS
  SELECT a.titlu, a.autor, b.editura from carti_t a, editura_t b
  where (a.editura=b.id) and (a.editura=:P5_EDITURA);
v_carti_t_record carti_t_cursor%ROWTYPE;
BEGIN
  OPEN carti_t_cursor;
  LOOP
    FETCH carti_t_cursor INTO v_carti_t_record;
    EXIT WHEN carti_t_cursor%NOTFOUND;
    htp.p('Titlu: ' || v_carti_t_record.titlu || ' - Autor: ' ||
v_carti_t_record.autor || ' - Editura: '||v_carti_t_record.editura);
    htp.nl();
  END LOOP;
  CLOSE carti_t_cursor;
END;
```



Biblioteca /

## Cursor explicit- afisarea cartilor unei edituri date

CURSOR EXPLICIT

Editura

Titlu: Matricea divina - Autor: Gregg Braden - Editura: For You  
Titlu: Imageria inimii - Autor: Daniel Mitel - Editura: For You  
Titlu: Cartea Secretelor - Autor: Deepak Chopra - Editura: For You

---

**Figura 4.5 Cursor explicit creat din aplicație**

## 5. ROW TRIGGERS IN ORACLE APEX

Ne propunem, în cele ce urmează, să creăm o aplicație în care să ținem evidența jucătorilor și a echipelor în care aceștia activează. În cazul în care un jucător se transferă la o altă echipă, acesta să nu fie primit dacă a mai evoluat anterior acolo. Pentru aceasta, este necesar să reținem istoricul fiecărui jucător într-o tabelă ale cărei date se vor încărca automat la modificarea tabeli ce reține evidența jucătorilor și a echipelor în care aceștia activează.

Fie tabela *jucatori1* care reține evidența jucătorilor și a echipelor în care activează aceștia (Figura 5.1)

JUCATORI1							
Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults

```

CREATE TABLE "JUCATORI1"
  (
    "CODJ" NUMBER NOT NULL ENABLE,
    "JUCATOR" VARCHAR2(30) NOT NULL ENABLE,
    "CODE" NUMBER NOT NULL ENABLE,
    CONSTRAINT "JUCATORI1_PK" PRIMARY KEY ("CODJ")
  USING INDEX ENABLE
  )
/
ALTER TABLE "JUCATORI1" ADD CONSTRAINT "JUCATORI1_FK" FOREIGN KEY ("CODE")
  REFERENCES "ECHIPE1" ("CODE") ON DELETE CASCADE ENABLE
/

```

**Figura 5.1 Crearea tabelii Jucatori1**

Jucatorii			
<input type="checkbox"/>	Codj	Jucator	Code
<input type="checkbox"/>	1	JUCATOR 1	echipa 1 ↕
<input type="checkbox"/>	22	JUCATOR 3	echipa 1 ↕
<input type="checkbox"/>	41	JUCATOR 4	echipa 1 ↕
<input type="checkbox"/>	21	JUCATOR 2	echipa 3 ↕

**Figura 5.2 Tabela Jucatori1**

Tabela *echipe1* reține echipele în care pot activa jucătorii(Figura 5.4).



The screenshot shows the Oracle SQL Developer interface with the 'ECHIPE1' table selected. The 'Table' tab is active, displaying the table's structure and associated trigger. The SQL code is as follows:

```
CREATE TABLE "ECHIPE1"
(
  "CODE" NUMBER NOT NULL ENABLE,
  "ECHIPA" VARCHAR2(10) NOT NULL ENABLE,
  CONSTRAINT "ECHIPE1_PK" PRIMARY KEY ("CODE")
  USING INDEX ENABLE
)
/

CREATE OR REPLACE TRIGGER "BI_ECHIPE1"
before insert on "ECHIPE1"
for each row
begin
  if :NEW."CODE" is null then
    select "ECHIPE1_SEQ".nextval into :NEW."CODE" from sys.dual;
  end if;
end;
/

ALTER TRIGGER "BI_ECHIPE1" ENABLE
/
```

**Figura 5.3 Crearea tabelii Echipe1**



The screenshot shows a table titled "Echipe" with the following data:

	Code	Echipa
<input type="checkbox"/>	1	echipa 1
<input type="checkbox"/>	2	echipa 2
<input type="checkbox"/>	3	echipa 3

Page number: 1 - 3

**Figura 5.4 Tabela Echipe1**

Tabela *istoric\_echipe1* reține istoricul fiecărui jucător și echipele în care acesta a mai evoluat(Figura 5.6

```
CREATE TABLE "ISTORIC_ECHIPE1"  
(  
  "COD" NUMBER NOT NULL ENABLE,  
  "CODJ" NUMBER NOT NULL ENABLE,  
  "CODE" NUMBER NOT NULL ENABLE,  
  CONSTRAINT "ISTORIC_ECHIPE1_PK" PRIMARY KEY ("COD")  
  USING INDEX ENABLE  
)  
/
```

**Figura 5.5 Crearea tabelii Istoricechipe1**

ISTORIC ECHIPE			
ISTORIC ECHIPE			
<input type="checkbox"/>	Cod	Codj	Code
<input type="checkbox"/>	41	JUCATOR 1 ▾	echipa 2 ▾
<input type="checkbox"/>	42	JUCATOR 3 ▾	echipa 1 ▾
<input type="checkbox"/>	43	JUCATOR 2 ▾	echipa 3 ▾
<input type="checkbox"/>	44	JUCATOR 4 ▾	echipa 1 ▾
<input type="checkbox"/>	61	JUCATOR 1 ▾	echipa 1 ▾

**Figura 5.6 Tabela istoric\_echipe1**

Pentru încărcarea automată a informațiilor în tabela *istoric\_exhipe1*, la fiecare actualizare a tabelii *jucători1* se execută trigger-ul *JUCATORII\_T1* (Figura 5.7).


Object Details	Code	Errors	SQL
	<pre>CREATE OR REPLACE TRIGGER "JUCATORI1_T1" AFTER insert or update or delete on "JUCATORI1" for each row begin BEGIN     INSERT INTO istoric_echipe1         (codj, code) VALUES (:NEW.codj, :NEW.code); END; end;  / ALTER TRIGGER "JUCATORI1_T1" ENABLE /</pre>		

**Figura 5.7 Trigger-ul JUCATORI1\_T1**

În tabela *istoric\_echipe1* se observă că *jucătorul 1* a mai evoluat anterior la *echipa 2*, iar în prezent face parte din *echipa 1*. În cazul în care dorim să-l transferăm înapoi la *echipa 2* în tabela *jucatori1*, se declanșează trigger-ul *jucatori1\_T3*(Figura 5.7) care nu permite acest lucru(Figura 5.8).

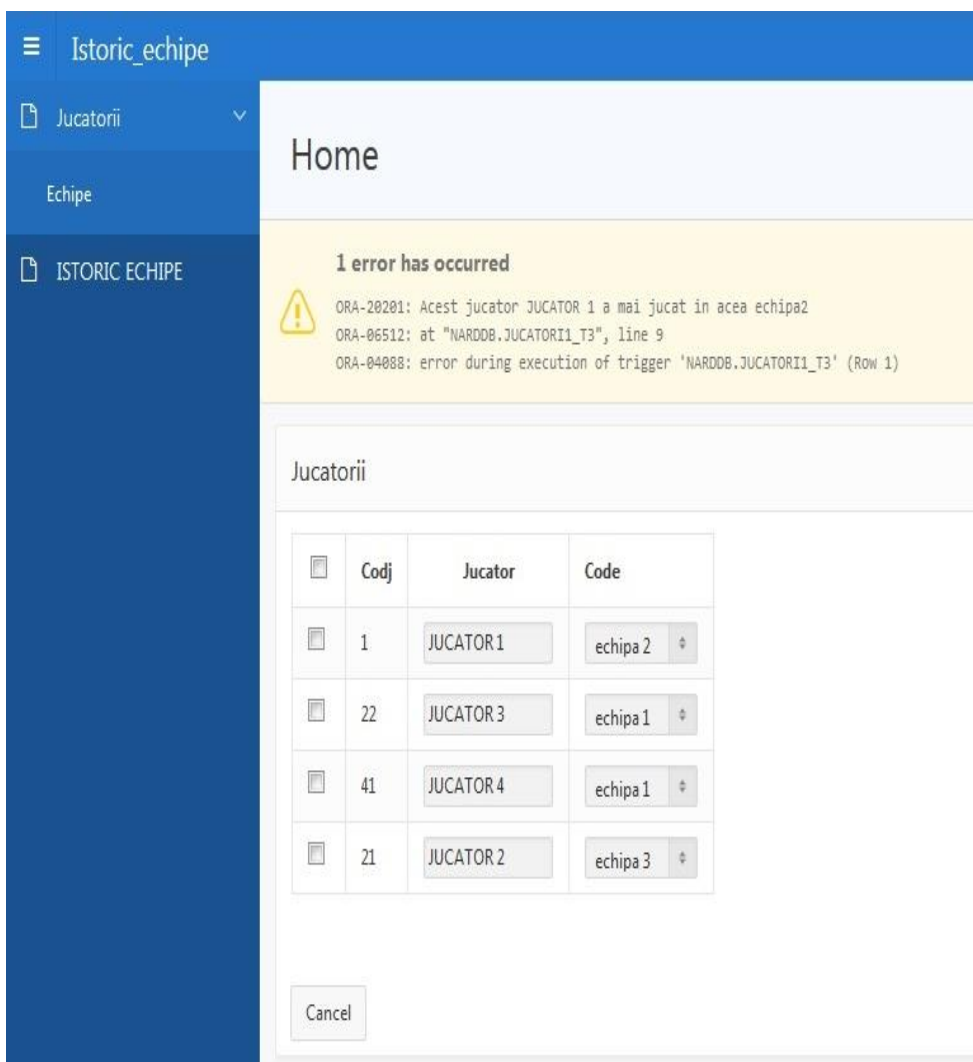
```
1 create or replace trigger "JUCATORI1_T3"  
2 BEFORE  
3 UPDATE OF "CODE" ON "JUCATORI1"  
4 for EACH ROW  
5 DECLARE  
6   nrechipe INTEGER;  
7 BEGIN  
8   SELECT count(*) into nrechipe  
9   FROM istoric_echipe1  
10  where (codj= :OLD.codj)  
11  AND (code= :NEW.code);  
12  IF (nrechipe>0) THEN  
13    RAISE_APPLICATION_ERROR (-20201,'Acest jucator ' || :OLD.jucator || ' a mai jucat in acea echipa' || :NEW.code);  
14  END IF;  
15 END;
```

**Figura 5.8** Trigger-ul jucatori1\_T3

ISTORIC ECHIPE	<p><b>1 error has occurred</b></p> <p> ORA-20201: Acest jucator JUCATOR 1 a mai jucat in acea echipa2 ORA-06512: at "NARDDB.JUCATORI1_T3", line 9 ORA-04088: error during execution of trigger 'NARDDB.JUCATORI1_T3' (Row 1)</p>
----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Figura 5.9** Declanșarea trigger-ului jucatori1\_T3





The screenshot displays the Oracle APEX application interface. The left sidebar contains navigation items: 'Istoric\_echipe', 'Jucatorii', 'Echipe', and 'ISTORIC ECHIPE'. The main content area is titled 'Home' and features a yellow error banner with a warning icon. The error message reads: '1 error has occurred', followed by three lines of Oracle error details: 'ORA-20201: Acest jucator JUCATOR 1 a mai jucat in acea echipa2', 'ORA-06512: at "NARDDDB.JUCATORII\_T3", line 9', and 'ORA-04088: error during execution of trigger 'NARDDDB.JUCATORII\_T3' (Row 1)'. Below the error banner is a table titled 'Jucatorii' with columns 'Codj', 'Jucator', and 'Code'. The table contains five rows of data. A 'Cancel' button is located at the bottom left of the table area.

Codj	Jucator	Code
1	JUCATOR 1	echipa 2
22	JUCATOR 3	echipa 1
41	JUCATOR 4	echipa 1
21	JUCATOR 2	echipa 3

**Figura 5.10** Execuția trigger-ului JUCATORII\_T3

## 6. GRAFICE ȘI PROCESE ÎN APLICAȚII ORACLE APEX

În aplicația descrisă mai jos, ne propunem să construim o tabelă pe baza informațiilor dintr-o altă tabelă, precum și un grafic al acesteia.

Se consideră o bază de date în care se reține evidența rezultatelor unei clase de elevi la testul inițial pe o disciplină. Pentru fiecare elev se cunosc următoarele informații: *numele și prenumele*, precum și *nota* acestuia la testul inițial respectiv. Ne propunem să realizăm o aplicație în ORACLE APEX prin care să putem gestiona și vizualiza datele respective, precum și rezultatele pe tranșe de note, la care vom atașa un grafic.

Se vor crea tabelele *TEST\_INIȚIAL1* și *TRANȘE\_DE\_NOTE* în care vom reține informațiile de care avem nevoie.

TEST_INITIAL1					TRANSE_DE_NOTE												
Table	Data	Indexes	Model	Constraints	Grants	Statistics	UIDefaults	Triggers	Table	Data	Indexes	Model	Constraints	Grants	Statistics	UIDefaults	Triggers
Add Column	Modify Column	Rename Column	Drop Column	Rename	Copy	Drop	Truncate	Cr	Add Column	Modify Column	Rename Column	Drop Column	Rename	Copy	Drop	Truncate	Creat
Column Name	Data Type	Nullable	Default	Primary Key													
COD	NUMBER	No	-	1													
NUME	VARCHAR2(50)	No	-	-													
NOTA	NUMBER	Yes	-	-													
				1 - 3													
Column Name	Data Type	Nullable	Default	Primary Key													
COD	NUMBER	No	-	1													
TRANSA_NOTE	VARCHAR2(30)	Yes	-	-													
NUMAR	NUMBER	Yes	-	-													
				1 - 3													

**Figura 6.1** Structură tabelle TEST\_INIȚIAL1 și TRANȘE\_DE\_NOTE

În continuare vom construi aplicația *Tranșe de note* ce va include trei pagini:

- O pagină de tip Blank ce va conține intrările către celelalte două pagini
- O pagină de tip Tabular form prin intermediul căreia vom gestiona informațiile din tabela Test\_initial1 ( pagina rezultate)
- O pagină de tip Tabular form prin intermediul căreia vom gestiona informațiile din tabela Transe\_de\_note. Informațiile din câmpul Număr vor conține numărul de note cuprinse în intervalul specificat în câmpul Transa\_note. Aceste informații

*vor fi introduse automat prin intermediul unui proces creat și atașat acestui tabular form.*

The screenshot displays the Oracle APEX Page Designer interface for 'Application 37248' in 'Page Designer' mode. The 'Processing' tab is active, showing a tree view on the left with a process named 'note2-3' highlighted. The main workspace shows a 'Tabular Form' with various regions and buttons. The right-hand panel is open to the 'Process' configuration for 'note2-3', showing the following details:

- Identification:** Name: note2-3, Type: Execute Code, Editable Region: - Select -
- Source:** Location: Local Database, Language: PL/SQL, PL/SQL Code:

```
begin
select count(*) into :P4_X
from TEST_INITIAL1
where nota between 2 and 2.99;
end;
```
- Execution Options:** Sequence: 10, Point: Processing, Run Process: Once Per Page Visit
- Success Message:** Success Message: ok

**Figura 6.2** Procesul Note2-3

The screenshot displays the Oracle APEX Page Designer interface for 'Application 37248 \ Page Designer'. The 'Processing' tab is active, showing a tree view on the left with 'note 4-5' selected. The main workspace shows a 'Tabular Form' with various buttons like 'MULTI\_ROW\_DELETE', 'SUBMIT', and 'ADD'. The right-hand 'Process' configuration panel is open, showing the following details:

- Identification:** Name: note 4-5, Type: Execute Code, Editable Region: - Select -
- Source:** Location: Local Database, Language: PL/SQL, PL/SQL Code:

```
begin
select count(cod) into :P6_X
from TEST_INITIAL1
where nota between 4 and 4.99;
end;
```
- Execution Options:** Sequence: 20, Point: Processing, Run Process: Once Per Page Visit
- Success Message:** Success Message: ok

**Figura 6.3** Procesul Note 4-5

The screenshot displays the Oracle APEX Page Designer interface for Application 37248, Page Designer. The interface is divided into several sections:

- Left Panel (Navigation):** Shows a tree view of the page components. Under the 'Processing' section, the process 'Note intre 3 si 4' is highlighted with a green dashed border.
- Center Panel (Tabular Form):** Displays a preview of a tabular form with various buttons such as 'MULTI\_ROW\_DELETE', 'SUBMIT', 'CANCEL', 'EDIT', 'CHANGE', 'CREATE', 'EXPAND', 'COPY', and 'HELP'. The form includes regions for 'ITEMS', 'REGION CONTENT', 'SUB-REGIONS', 'BELOW REGION', and 'BOTTOM OF REGION'.
- Right Panel (Process Configuration):**
  - Identification:** Name: 'Note intre 3 si 4', Type: 'Execute Code', Editable Region: '- Select -'.
  - Source:** Location: 'Local Database', Language: 'PL/SQL', PL/SQL Code:

```

begin
select count(COD) into :PE_X
from TEST_INITIAl1
where nota between 3 and 3.99;
end;

```
  - Execution Options:** Sequence: 30, Point: 'Processing', Run Process: 'Once Per Page Visit'.
  - Success Message:** Success Message: 'ok'.

**Figura 6.4** Procesul Note intre 3 si 4

*Procesul TRANSE DE NOTE*

*declare*

*v1 number;*

*v2 number;*

*v3 number;*

*v4 number;*

*v5 number;*

*v6 number;*

*v7 number;*

*v8 number;*

*v9 number;*

*v10 number;*

*begin*

*select count(cod) into v1 from test\_initial1 where nota between 1 and 1.99;*

*select count(cod) into v2 from test\_initial1 where nota between 2 and 2.99;*

*select count(cod) into v3 from test\_initial1 where nota between 3 and 3.99;*

*select count(cod) into v4 from test\_initial1 where nota between 4 and 4.99;*

*select count(cod) into v5 from test\_initial1 where nota between 5 and 5.99;*

*select count(cod) into v6 from test\_initial1 where nota between 6 and 6.99;*

*select count(cod) into v7 from test\_initial1 where nota between 7 and 7.99;*

*select count(cod) into v8 from test\_initial1 where nota between 8 and 8.99;*

```
select count(cod) into v9 from test_initial1 where nota between 9
and 9.99;
select count(cod) into v10 from test_initial1 where nota=10;
update transe_de_note
set numar = v1 where transa_note = 'intre 1-1.99';
update transe_de_note
set numar = v2 where transa_note = 'intre 2-2.99';
update transe_de_note
set numar = v3 where transa_note = 'intre 3-3.99';
update transe_de_note
set numar = v4 where transa_note = 'intre 4-4.99';
update transe_de_note
set numar = v5 where transa_note = 'intre 5-5.99';
update transe_de_note
set numar = v6 where transa_note = 'intre 6-6.99';
update transe_de_note
set numar = v7 where transa_note = 'intre 7-7.99';
update transe_de_note
set numar = v8 where transa_note = 'intre 8-8.99';
update transe_de_note
set numar = v9 where transa_note = 'intre 9-9.99';
update transe_de_note
set numar = v10 where transa_note = 'nota 10';
end;
```



The screenshot displays the Oracle APEX Page Designer interface for Application 37248. The 'Processing' tab is active, showing a list of processes on the left. The 'TRANSE DE NOTE' process is selected and highlighted in green. The main workspace shows a 'Tabular Form' with various regions and buttons. The right-hand pane is configured for the 'TRANSE DE NOTE' process, showing the following settings:

- Source:** Local Database, PL/SQL
- PL/SQL Code:**

```

DECLARE
V1 NUMBER;
V2 NUMBER;
V3 NUMBER;
V4 NUMBER;
V5 NUMBER;
V6 NUMBER;
V7 NUMBER;
V8 NUMBER;
V9 NUMBER;
V10 NUMBER;

```
- Execution Options:** Sequence 40, Point Processing, Run Process Once Per Page Visit
- Success Message:** Success Message: ok

**Figura 6.5** Procesul TRANSE DE NOTE

test initial > Transe de note

okOK X

Transe de note

Note 3-4

Note intre 4 si 4.99

Note de la 2 la 3

**Figura 6.6 Executia proceselor prin intermediul unor itemi**

Ne propunem în continuare să atașăm un grafic la tabela *Transe\_de\_note*. În graficul respective dorim să afișăm numărul de note din fiecare tranșă prezentă în tabela *Transe\_de\_note*.

rezultate			Tabular Form				
<input type="checkbox"/>	Cod	Nume	Nota	<input type="checkbox"/>	Cod	Transa Note	Numar
<input type="checkbox"/>	4	Dicu Ileana	2	<input type="checkbox"/>	1	INTRE 1-1.99	0
<input type="checkbox"/>	3	Nanu Oana	2.4	<input type="checkbox"/>	2	INTRE 2-2.99	3
<input type="checkbox"/>	1	Ilie Ion	2.5	<input type="checkbox"/>	3	INTRE 3-3.99	0
<input type="checkbox"/>	2	Vitan Ana	4.5	<input type="checkbox"/>	4	INTRE 4-4.99	1
<input type="checkbox"/>	5	Farcas Liliana	5	<input type="checkbox"/>	5	INTRE 5-5.99	2
<input type="checkbox"/>	6	Petrescu Gigi	5.5	<input type="checkbox"/>	6	INTRE 6-6.99	0
				<input type="checkbox"/>	7	INTRE 7-7.99	0
				<input type="checkbox"/>	8	INTRE 8-8.99	0
				<input type="checkbox"/>	9	INTRE 9-9.99	0
				<input type="checkbox"/>	10	NOTA 10	0

1 - 6 1 - 10

**Figura 6.7 Tabelele Test\_initial1 si Transe\_de\_note**

The figure consists of two screenshots of the Oracle APEX 'Create Region' wizard interface.

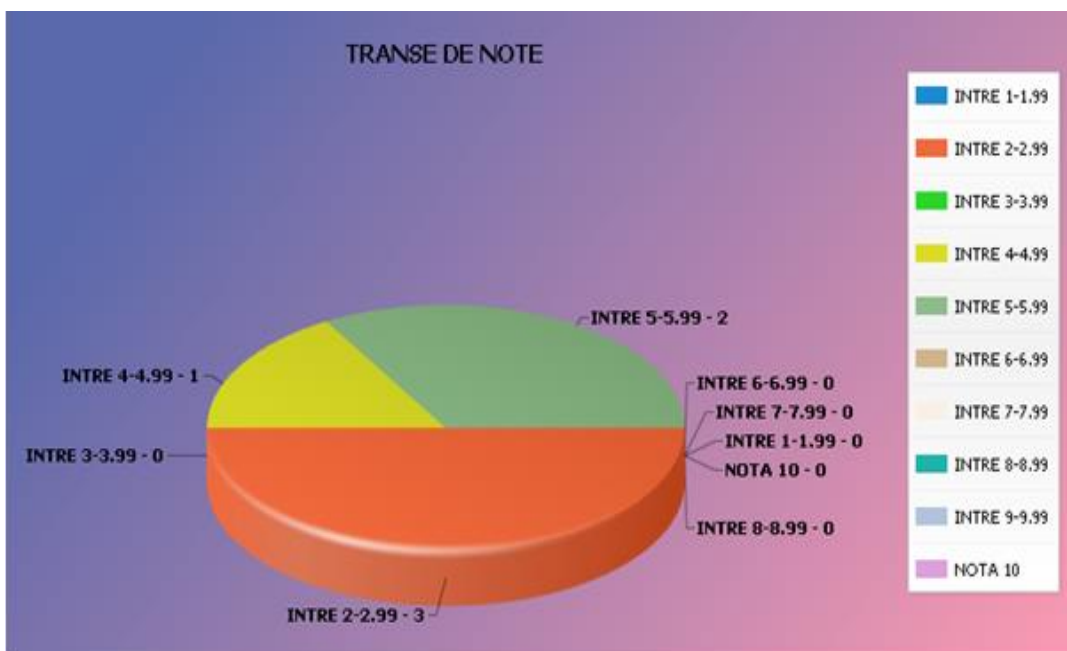
**Top Screenshot:** Shows the 'Create Region' wizard with the 'Chart Type' step selected. The 'Chart Type' is set to '3D Pie'. The 'Chart Title' is 'TRANSE DE NOTE'. The 'Background Type' is 'Transparent'. The 'Background Color 1' is '#F966A8' and 'Background Color 2' is '#F799B2'. The 'Gradient Angle' is '30'. The 'Color Scheme' is 'Look 6'. The 'Show Hints', 'Show Labels', and 'Show Values' checkboxes are checked. The 'Show Legend' options are 'None', 'Left', 'Right', 'Top', and 'Bottom'.

**Bottom Screenshot:** Shows the 'Create Region' wizard with the 'Source' step selected. The 'Page' is '4 - Transe de note'. The 'Region Source Type' is 'Flash Chart'. The 'Title' is 'Transe de note'. The 'Region Template' is 'Chart Region'. The 'Parent Region' is '- Select a Parent -'. The 'Display Point' is 'Page Template Body (3 items above region content)'. The 'Sequence' is '11' and the 'Column' is '1'. The 'Source' field contains the following SQL query:

```
select mail link, TRANSA_NOTE label, NOTAS value1
from "TRAF00B"."TRANSE_DE_NOTE"
```

The 'Build Query' button is visible at the bottom right of the wizard.


**Figura 6.8** Atașarea unui grafic la tabela *Transe\_de\_note*



**Figura 6.9** Vizualizarea graficului

La o modificare a tabelii *Test\_initial1* se vor modifica automat datele din câmpul *Număr* al tabelii *Transe\_de\_note*, precum și graficul atașat acesteia.

rezultate

<input type="checkbox"/>	Cod	Nume	Nota 
<input type="checkbox"/>	4	Dicu Ileana	2
<input type="checkbox"/>	3	Nanu Oana	2.4
<input type="checkbox"/>	1	Ilie Ion	2.5
<input type="checkbox"/>	2	Vitan Ana	4.5
<input type="checkbox"/>	5	Farcas Liliana	5
<input type="checkbox"/>	6	Petrescu Gigi	5.5
<input type="checkbox"/>	24	Tinca Eduard	8
<input type="checkbox"/>	21	Mihaela Carmen	9.55
<input type="checkbox"/>	22	Sandu Ion	10
<input type="checkbox"/>	23	Draghici Cristina	10

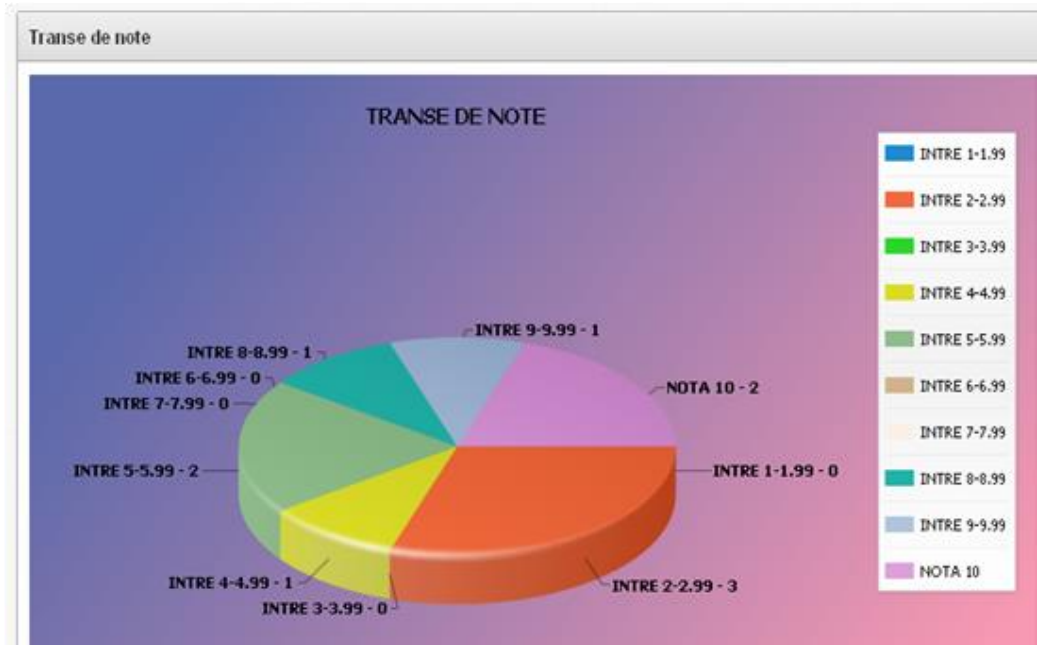
1 - 10

**Figura 6.10** Tabela *Test\_initial1* modificată

Tabular Form

<input type="checkbox"/>	Cod	Transa Note	Numar
<input type="checkbox"/>	1	INTRE 1-1.99	0
<input type="checkbox"/>	2	INTRE 2-2.99	3
<input type="checkbox"/>	3	INTRE 3-3.99	0
<input type="checkbox"/>	4	INTRE 4-4.99	1
<input type="checkbox"/>	5	INTRE 5-5.99	2
<input type="checkbox"/>	6	INTRE 6-6.99	0
<input type="checkbox"/>	7	INTRE 7-7.99	0
<input type="checkbox"/>	8	INTRE 8-8.99	1
<input type="checkbox"/>	9	INTRE 9-9.99	1
<input type="checkbox"/>	10	NOTA 10	2

**Figura 6.11** Tabela *Transe\_de\_note* modificată



*Figura 6.12 Noul grafic rezultat în urma modificării*

## 7. FOLOSIREA LISTELOR DE VALORI SI A PROCESELOR IN APLICATII ORACLE APEX

Nu de puține ori, în cadrul unei aplicații ORACLE, atunci când dorim să introducem valori într-un câmp ce reprezintă o cheie străină într-o tabelă, ne întrebăm: *Ce valoare are câmpul de legătură ce reprezintă cheia primară în tabela de referință(tabela părinte), pentru ceea ce dorim să introducem în câmpul respectiv din tabela-fiu? Ar fi cu mult mai ușor dacă am avea la dispoziție o listă din care să putem selecta aceste valori, la introducerea datelor într-un câmp ce reprezintă o cheie străină.* Ne propunem în aplicația descrisă mai jos, să exemplificăm acest lucru, prin folosirea listelor de valori. Totodată vom realiza completarea automată a datelor din anumite câmpuri ale unei tabele, prin intermediul proceselor create în cadrul aplicației.

Se consideră o bază de date în care se reține evidența elevilor la un concurs școlar. Pentru fiecare elev se cunosc următoarele informații: *numele și prenumele, clasa, nota1, nota2, media* precum și *rezultatul* obținut în cadrul concursului. Ne propunem să realizăm o aplicație în APEX prin care să putem gestiona și vizualiza datele respective.

Se vor crea două tabele *CLASE*, respectiv *ELEVII* în care vom reține informațiile de care avem nevoie. Legătura dintre cele două tabele se va face prin intermediul câmpului *codclasa* ce va fi cheia străină în tabela *ELEVII* și care va face referire la câmpul *cod*, cheia primară din tabela *CLASE*.

CLASE										
Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL
Add Column	Modify Column	Rename Column	Drop Column	Rename	Copy	Drop	Truncate	Create Lookup Table		
Column Name	Data Type	Nullable	Default	Primary Key						
COD	NUMBER	No	-	1						
DENUMIRE	VARCHAR2(4000)	No	-	-						
				1 - 2						

**Figura 7.1 Tabela CLASE**

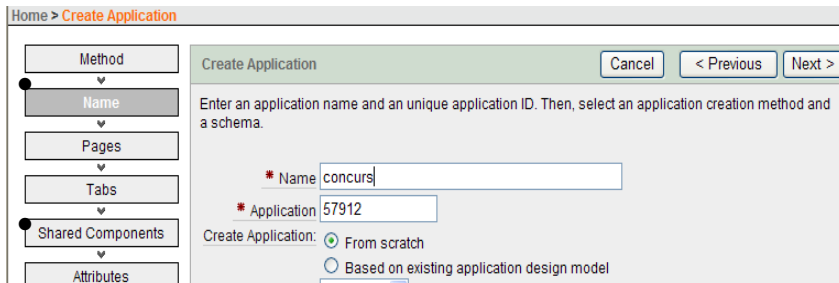
ELEVII										
Table	Data	Indexes	Model	Constraints	Grants	Statistics	UI Defaults	Triggers	Dependencies	SQL
Add Column	Modify Column	Rename Column	Drop Column	Rename	Copy	Drop	Truncate	Create Lookup Table		
Column Name	Data Type	Nullable	Default	Primary Key						
COD	NUMBER	No	-	1						
NUME	VARCHAR2(4000)	No	-	-						
CODCLASA	NUMBER	No	-	-						
NOTA1	NUMBER	Yes	-	-						
NOTA2	NUMBER	Yes	-	-						
MEDIA	NUMBER	Yes	-	-						
REZULTAT	VARCHAR2(4000)	Yes	-	-						
				1 - 7						

**Figura 7.2 Tabela ELEVII**



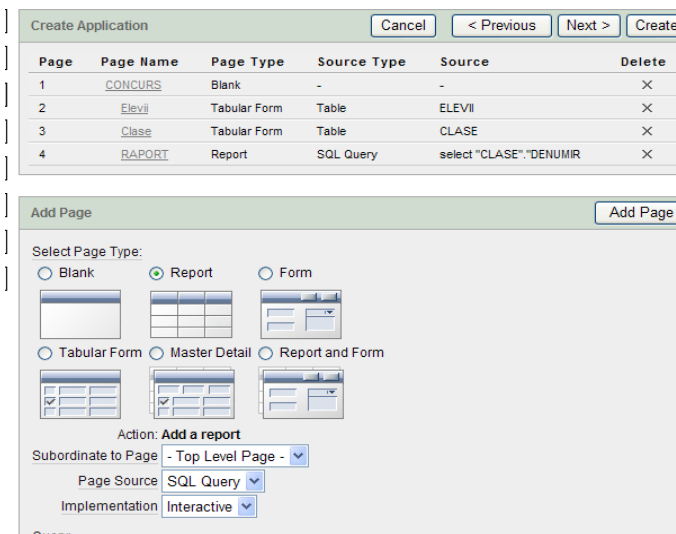
În continuare vom construi aplicația *CONCURS* ce va include 4 pagini:

- pagină de tip *Blank* ce va conține un cuprins din care vom accesa celelalte pagini



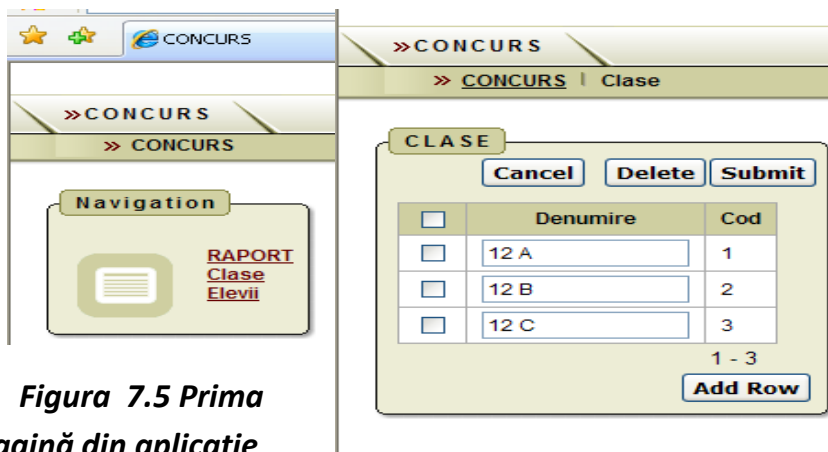
**Figura 7.3 Crearea aplicației CONCURS**

- Două pagini de tip *Tabular Form* prin intermediul cărora vom actualiza datele din cele două tabele
- O pagină de tip *Report* prin intermediul căreia vom vizualiza informațiile din cele două tabele



**Figura 7.4** Paginile aplicației

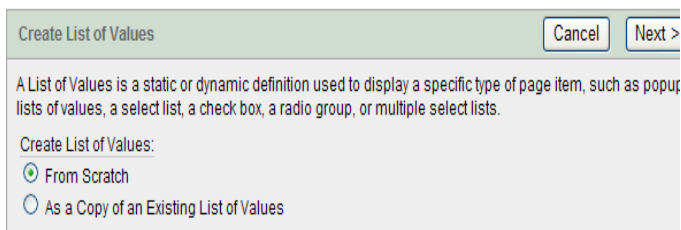
Aplicația va arăta în felul următor:



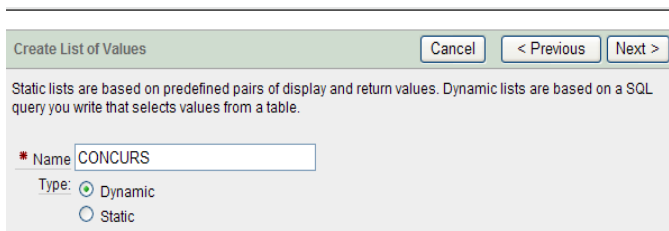
**Figura 7.5** Prima pagină din aplicație

**Figura 7.6** A treia pagină din aplicație

Pentru introducerea datelor în tabela ELEVII ne propunem să creăm o listă de valori pentru a selecta clasa din care face parte fiecare elev.

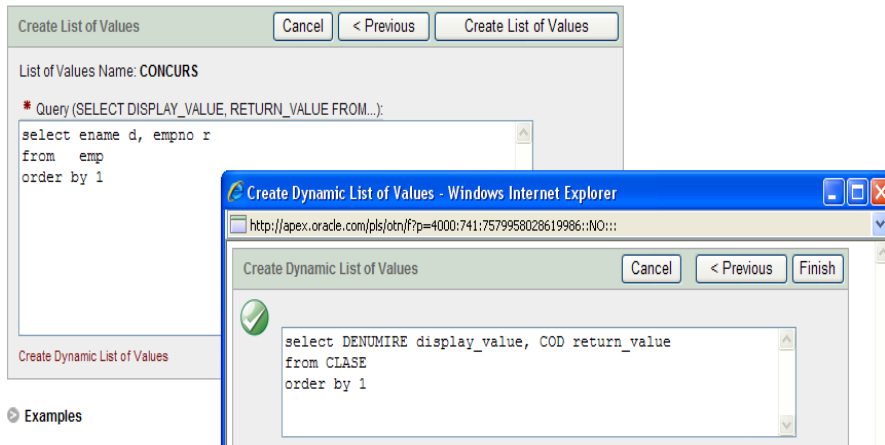


**Figura 7.7 Crearea listei de valori**



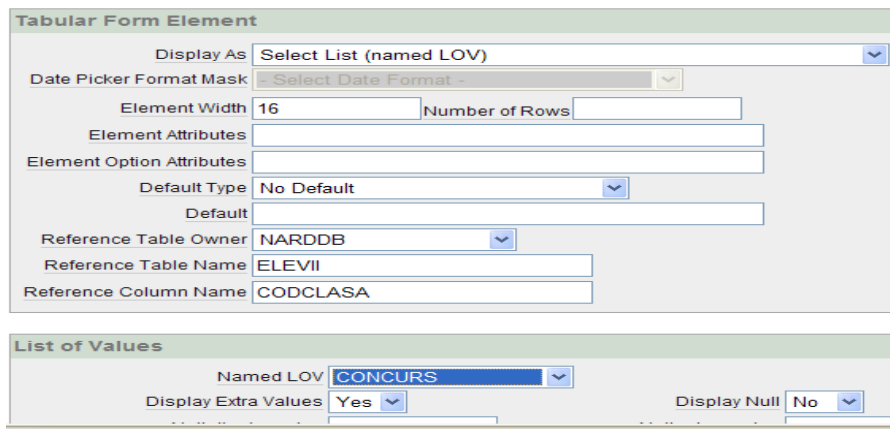
**Figura 7.8 Selectarea tipului listei de valori**

Selectăm tabela- referință, câmpul ale cărui valori dorim să se afișeze în listă, precum și câmpul ale cărui valori se vor introduce în fundal(în câmpul FK).



**Figura 7.9** Descrierea listei de valori

În continuare, edităm atributele câmpului *codclasa* din tabela *ELEVII*, ale cărui valori se vor introduce din lista de valori creată anterior



**Figura 7.10** Selectarea listei de valori

<input type="checkbox"/>	Cod	Nume	CLASA	Nota1	Nota2	Media	Rezultat
<input type="checkbox"/>	1	ION ELENA	12 A	8.52	7.45		
<input type="checkbox"/>	(null)	DINU IONUT					

**Figura 7.11 Folosirea listei de valori creată**

Pentru completarea automată a valorilor din câmpurile *Media* și *Rezultat* în cardul tablei *Elevii*, vom crea un PL/SQL proces în aplicație.

Page Processing

Computations

Validations

**Processes**

After Submit

10	Apply MRU	Multi Row Update	Conditional
20	Apply MRD	Multi Row Delete	Conditional
30	Apply MRU	Multi Row Update	Conditional
40	Add Rows	Add rows to tabular form	Conditional

Branches

After Processing

1	Go To Page_2	Unconditional
---	--------------	---------------

Create Page Process

Page: 2 - Elevii

Point: On Submit - After Computations and Validations

Enter PL/SQL Page Process

```
BEGIN
UPDATE ELEVII
SET MEDIA=(NOTA1+NOTA2)/2;
UPDATE ELEVII
SET REZULTAT='ADMIS'
WHERE MEDIA>=5;
UPDATE ELEVII
SET REZULTAT='RESPINS'
WHERE MEDIA<5;
END;
```

Do not validate PL/SQL code (parse PL/SQL code at runtime only).

**Figura 7.12 Crearea unui proces**

The screenshot displays an Oracle APEX application interface. At the top, there is a breadcrumb trail: >> CONCURS >> CONCURS | Elevii. Below this, a message box says "AM ACTUALIZAT" with a close button. The main content area is titled "ELEVII" and contains a table with columns: Cod, Nume, CLASA, Nota1, Nota2, Media, and Rezultat. There are checkboxes in the first column and buttons for "Cancel", "Delete", and "Submit" at the top right. The table has two rows of data. Below the table, there is a page indicator "1 - 2" and an "Add Row" button.

<input type="checkbox"/>	Cod	Nume	CLASA	Nota1	Nota2	Media	Rezultat
<input type="checkbox"/>	2	DINU IONUT	12 B	4	5	4.5	RESPINS
<input type="checkbox"/>	1	ION ELENA	12 A	8.52	7.45	7.985	ADMIS

**Figura 7.13** Execuția procesului în cadrul aplicației

## 8. ITEMI CU ACȚIUNE DINAMICĂ ÎN ORACLE APEX

La introducerea datelor în formularele unei aplicații cu baze de date, precum și la afișarea acestora, apare uneori necesitatea controlării acestor operații, în funcție de anumite condiții pe care trebuie să le îndeplinească aceste date. Introducerea datelor sau afișarea acestora în câmpurile unui formular poate fi condiționată prin crearea unor **acțiuni dinamice** pe care le vom atașa itemilor ce rețin acele câmpuri.

În cele ce urmează, ne propunem să exemplificăm acest lucru prin intermediul unei aplicații create în *ORACLE APEX 4.0*. Aplicația respectivă va gestiona datele unui examen alcătuit din patru probe, din care primele două sunt eliminatorii și au ca rezultat ADMIS sau RESPINS. La introducerea datelor, în cazul în care un candidat nu a promovat una dintre cele două probe eliminatorii, dorim ca următoarele câmpuri din formular să fie dezactivate, iar rezultatul să fie afișat automat. Un candidat se va considera admis, în cazul în care va promova cele două probe eliminatorii și va avea minim 5 la următoarele două probe, iar media va fi minim 6. Valoarea mediei se va calcula automat pentru candidații admiși la probele eliminatorii. Aplicația va gestiona datele din două tabele, *Rezultat* și *Examen* ce vor reține valorile rezultatului unui examen, respectiv datele candidaților la examen.

Tabelele *Rezultat* și *Examen* au structura prezentată în Figura 8.1, respectiv Figura 8.2.

Column Name	Data Type	Nullable	Default	Primary Key
COD	NUMBER	No	-	1
REZULTAT	VARCHAR2(10)	No	-	-
				1 - 2

**Figura 8.1 Tabela Rezultat**

Column Name	Data Type	Nullable	Default	Primary Key
COD	NUMBER	No	-	1
CANDIDAT	VARCHAR2(50)	No	-	-
PROBA_A	NUMBER	No	-	-
PROBA_B	NUMBER	Yes	-	-
PROBA_C	NUMBER	Yes	-	-
PROBA_D	NUMBER	Yes	-	-
MEDIA	NUMBER	Yes	-	-
REZULTAT	VARCHAR2(50)	Yes	-	-

**Figura 8.2 Tabela Examen**

Create Application					
Page	Page Name	Page Type	Source Type	Source	Delete
1	<a href="#">Examen</a>	Report	Table	EXAMEN	X
2	<a href="#">Examen</a>	Form	Table	EXAMEN	X

**Figura 8.3 Aplicația EXAMEN**



The image shows a screenshot of an Oracle APEX form titled "FORMULAR EXAMEN". The form is displayed on a light yellow background. It contains the following fields from top to bottom: a text input field for "Candidat"; a dropdown menu for "Proba A" with "RESPINS" selected; another dropdown menu for "Proba B" also with "RESPINS" selected; a text input field for "Proba C"; a text input field for "Proba D"; a text input field for "Media"; and a text input field for "Rezultat".

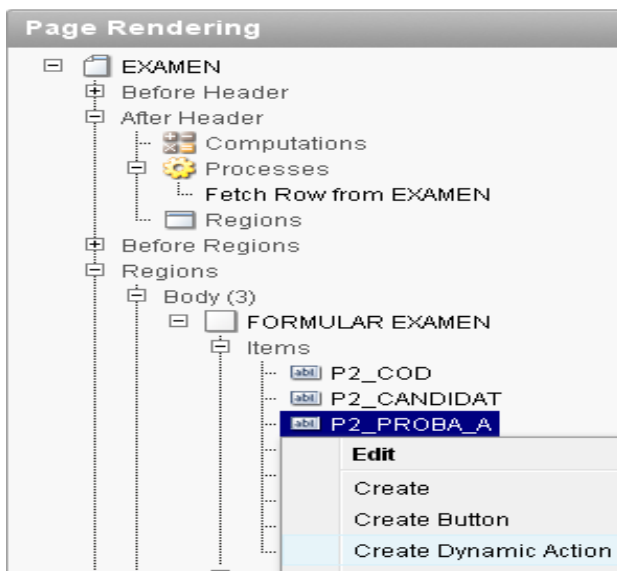
**Figura 8.4 Formularul EXAMEN**

Se crează aplicația *EXAMEN* ce are în componența sa un raport și un formular.

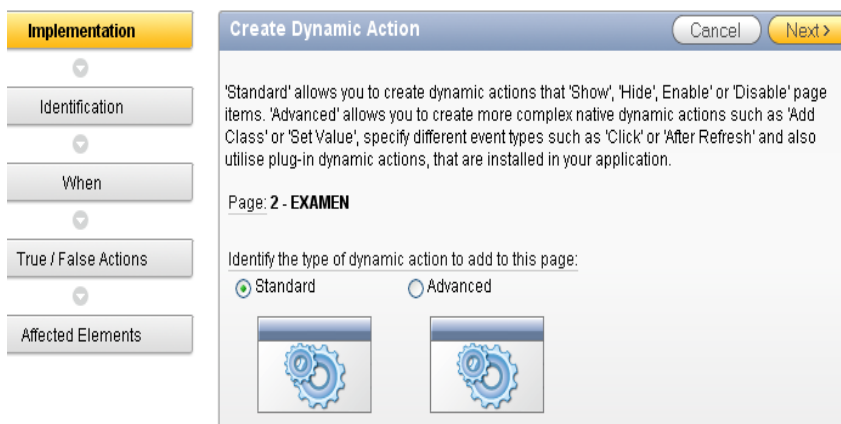
În continuare, vom edita itemul *P2\_PROBA\_A* corespunzător câmpului **Proba A** din cadrul formularului, astfel încât să putem selecta pentru acesta, unul din rezultatele *ADMIS* sau *RESPINS* din tabela *REZULTAT*. Pentru aceasta vom crea o listă dinamică de valori pe care o vom atașa acestui item. Analog vom proceda pentru itemul *P2\_PROBA\_B* corespunzător **Probei B** din formular.

Formularului de introducere a datelor va arăta ca în Figura 8.4. Având în vedere faptul că primele două probe din examen sunt eliminatorii, dorim să blocăm introducerea datelor în continuare pentru candidații ce nu au promovat la una din cele două probe, dezactivând următoarele câmpuri din formular. Pentru aceasta

vom crea o acțiune dinamică standard, atașată itemului PROBA\_A. Analog vom proceda pentru itemul PROBA\_B.



*Pasul 1: Selectarea itemului și atașarea unei acțiuni dinamice*



**Pasul 2: Crearea unei acțiuni dinamice standard**

Implementation

Identification

When

True / False Actions

Affected Elements

**Create Dynamic Action**

Page: 2 - EXAMEN

\* Name: PROBA\_A ELIMINATORIE

\* Sequence: 10

Existing Dynamic Actions

**Pasul 3: Denumirea acțiunii dinamice**

Implementation

Identification

When

True / False Actions

Affected Elements

**Create Dynamic Action**

Specify when you would like the Dynamic Action

Page: 2 - EXAMEN

Name: PROBA\_A ELIMINATORIE

\* Selection Type: Item(s)

Item, Region

\* Item(s): P2\_PROBA\_A

Condition: equal to

equal to, is null

\* Value: 2

**Pasul 4: Stabilirea valorii itemului respectiv, pentru care se va declanșa acțiunea dinamică**

Implementation

Identification

When

**True / False Actions**

Affected Elements

**Create Dynamic Action** Cancel < Previous Next

Specify the 'True' and 'False' action for this dynamic action. 'True' actions fire either when the 'When Condition' is met, or when 'No Condition' has been specified. 'False' actions fire only when the 'When Condition' is not met. Therefore, if no 'When Condition' has been specified, only a 'True' action can be created.

Page: **2 - EXAMEN**  
Name: **PROBA\_A ELIMINATORIE**

\* Specify the True Action:  Show  
 Hide  
 Enable  
 Disable

Create Opposite False Action

***Pasul 5: Selectarea acțiunii(Enable) ce se va declanșa la valoarea selectată la pasul anterior, pentru itemul respectiv***

Implementation

Identification

When

True / False Actions

**Affected Elements**

**Create Dynamic Action**

Select which page elements you would like the dynamic action to control.

Page: **2 - EXAMEN**  
Name: **PROBA\_A ELIMINATORIE**  
True Action: **Enable**

Selection Type:

Item(s)

P2\_CANDIDAT  
P2\_COD  
P2\_PROBA\_A  
P2\_REZULTAT

P2\_PROBA\_B  
P2\_PROBA\_C  
P2\_PROBA\_D  
**P2\_MEDIA**

***Pasul 6: Selectarea itemilor pentru care se va declanșa acțiunea dinamică nou creată***

The image shows three sequential screenshots of a form titled 'FORMULAR EXAMEN'. Each form contains the following fields: 'Candidat' (text input), 'Proba A' (dropdown menu), 'Proba B' (dropdown menu), 'Proba C' (text input), 'Proba D' (text input), 'Media' (text input), and 'Rezultat' (text input).  
 - Screenshot 1: 'Proba A' is set to 'RESPINS'.  
 - Screenshot 2: 'Proba A' is set to 'ADMIS'.  
 - Screenshot 3: 'Proba A' is set to 'RESPINS'.

### ***Pasul 7: Declanșarea acțiunii dinamice pentru itemul corespunzător câmpului Proba A din formularul aplicației***

În continuare vom crea o acțiune dinamică de tip avansat, pentru câmpul **Media**, ce va avea ca efect calculul automat al mediei probelor scrise, în cazul în care se vor edita notele în câmpurile respectivelor probe.

#### ***Pasul 1: Selectarea itemilor la a căror modificare se va declanșa acțiunea dinamică***

The screenshot shows the 'Create Dynamic Action' dialog box. The configuration is as follows:  
 - Page: 2 - EXAMEN  
 - Name: MEDIA  
 - Event: Change  
 - Selection Type: Item(s)  
 - Item(s): P2\_PROBA\_C, P2\_PROBA\_D  
 - Condition: - No Condition -  
 - Equal to, is null

The screenshot shows the 'Settings' section of the 'Create Dynamic Action' dialog box. The configuration is as follows:  
 - The following action will fire when the 'When Condition' is met or w  
 - Page: 2 - EXAMEN  
 - Name: MEDIA  
 - Action: Set Value  
 - Fire On Page Load:   
 - Set Type: PL/SQL Expression  
 - PL/SQL Expression: (: P2\_PROBA\_C+ : P2\_PROBA\_D) / 2  
 - Page Items to Submit: P2\_PROBA\_C, P2\_PROBA\_D  
 - Escape Special Characters: Yes

**Create Dynamic Action**

Select which page elements you would like the dynamic action to control.

Page: **2 - EXAMEN**  
 Name: **MEDIA**  
 True Action: **Set Value**

Selection Type:

Item(s):

- P2\_CANDIDAT
- P2\_COD
- P2\_PROBA\_A
- P2\_PROBA\_B
- P2\_PROBA\_C
- P2\_PROBA\_D
- P2\_REZULTAT

P2\_MEDIA

**Pasul 2: Selectarea acțiunii ce se va declanșa pentru câmpul Media (calculul mediei folosind o expresie PL/SQL)**  
**Selectarea itemului corespunzător câmpului Media, unde se va declanșa acțiunea dinamică**

**FORMULAR EXAMEN**

<b>Candidat</b>	<input type="text"/>
<b>Proba A</b>	ADMIS <input type="button" value="v"/>
<b>Proba B</b>	ADMIS <input type="button" value="v"/>
<b>Proba C</b>	<input type="text" value="8"/>
<b>Proba D</b>	<input type="text" value="7"/>
<b>Media</b>	<input type="text" value="7.5"/>
<b>Rezultat</b>	<input type="text"/>

**Pasul 3: Declanșarea acțiunii dinamice atașată câmpului Media din cadrul formularului**

În continuare, în câmpul *Rezultat* dorim să se afișeze automat rezultatul la examen pentru un candidat. Pentru aceasta, vom crea în mod analog, o acțiune dinamică de tip avansat pe care o vom atașa acestui câmp. Această acțiune dinamică se va declanșa la orice modificare a informațiilor din câmpurile ce rețin rezultatele la probele de examen. Pentru crearea acestei acțiuni dinamice vom folosi următoarea funcție PL/SQL ce va întoarce valoarea pentru câmpul *Rezultat*:

**declare**

**r1 Varchar2(50);**

**r2 varchar2(50);**

**r varchar2(50);**

**begin**

**select rezultat into r1 from rezultat where :P2\_PROBA\_A=cod;**

**select rezultat into r2 from rezultat where :P2\_PROBA\_B=cod;**

**if (r1=r2)and(r1='ADMIS')and(:P2\_PROBA\_C>=5)AND**

**(:P2\_PROBA\_D>=5)AND(:P2\_MEDIA>=6)**

**then r:='ADMIS' ;**

**else r := 'RESPINS';**

**end if;**

**return r;**

**END;**

**FORMULAR EXAMEN**

**Candidat**

**Proba A**  ▼

**Proba B**  ▼

**Proba C**

**Proba D**

**Media**

**Rezultat**

**FORMULAR EXAMEN**

**Candidat**

**Proba A**  ▼

**Proba B**  ▼

**Proba C**

**Proba D**

**Media**

**Rezultat**

**EXAMEN**

CANDIDAT ▼	PROBA A	PROBA B	PROBA C	PROBA D	MEDIA	REZULTAT
Ana Marin	ADMIS	ADMIS	5	5.5	5.25	RESPINS
Ion Sandu	ADMIS	RESPINS				RESPINS
ion	ADMIS	ADMIS	7	6	6.5	ADMIS

1 - 3

**Figura 8.5** Aplicația Examen



## BIBLIOGRAFIE

1. **ORACLE ACADEMY**- *Database Programming with PL\SQL* ,  
<https://academy.ORACLE.com/>, 2018
2. **ORACLE® APPLICATION EXPRESS** 18.2 -  
<http://apex.oracle.com/>