

Proiect Sisteme de gestiune a bazelor de date

Tema: Administrarea unei echipe de football



Matei Adrian

Seria D

Grupa 1055

Cuprins

Descrierea temei	3
Schema conceptuală pentru modelarea bazei de date	4
Interacțiunea cu serverul Oracle prin intermediul comenzilor SQL (LDD și LMD)	5
Structuri alternative și repetitive	7
Tratarea excepțiilor	9
Gestionarea cursorilor	12
Funcții, proceduri, includerea acestora în pachete	13
Declanșatori	16
Aplicație APEX	19

Descrierea temei

Tema pe care am ales-o este “Administrarea unei echipe de football”. Am ales această temă pentru că evidența jucătorilor și a angajaților dintr-o echipă de football trebuie ținută obligatoriu, cât și contractele cu jucătorii și sponsorii echipei, de cele mai multe ori în joc fiind vorba de sume de bani destul de mari.

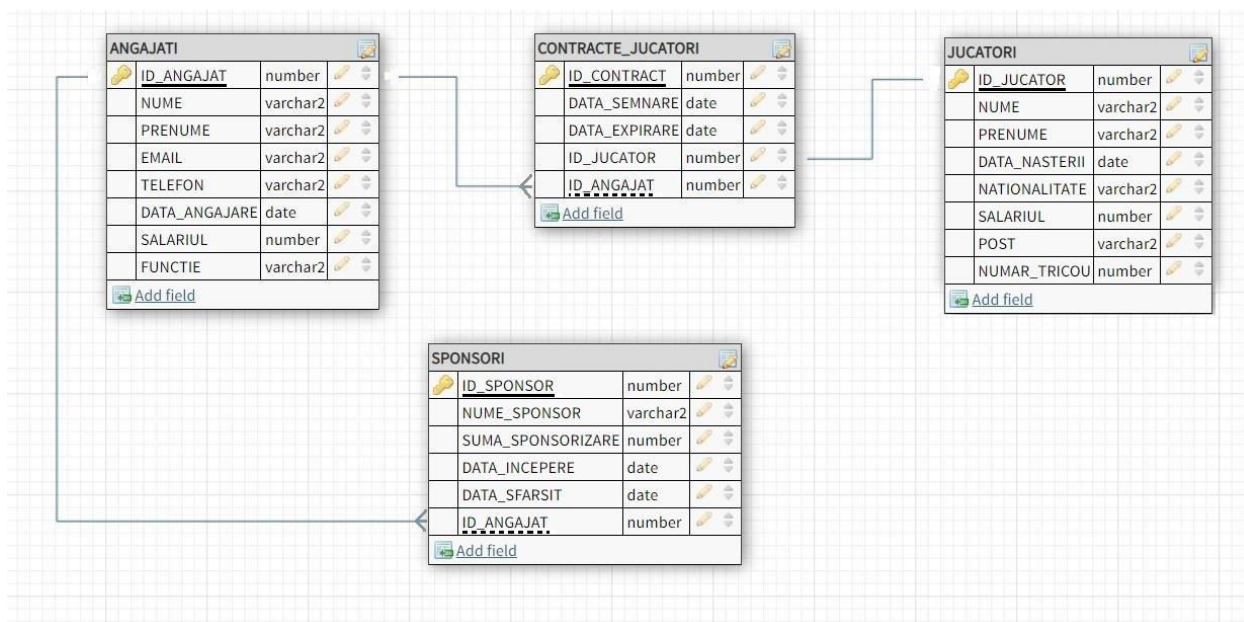
Prima tabelă este cea a angajaților, a oamenilor care se ocupă de buna funcționare a echipei (manager, director sportiv/marketing, îngrijitor teren, doctor, antrenor, etc). Am ales să pun coloana FUNCTIE tot în această tabelă (fără a mai face o tabela separată pentru funcții) deoarece echipa la care voi face referire este una fictivă, aceasta având un număr restrâns de angajați, respectiv un angajat/funcție. Restul coloanelor reprezintă date de identificare (ID_ANGAJAT, NUME, PRENUME, EMAIL, TELEFON), dar și 2 coloane pentru a putea ține evidența angajaților (DATA_ANGAJARE, SALARIUL).

A doua tabelă este cea a jucătorilor, având de asemenea câteva coloane pentru identificare (ID_JUCATOR, NUME, PRENUME, DATA_NASTERII, NATIONALITATE), dar și o coloană cu postul (coloană POST) pe care aceștia joacă, salariul (coloana SALARIUL) pe care aceștia îl iau lunar, dar și numărul de pe tricou (coloana NUMAR_TRICOU).

A treia tabelă este cea în care sunt stocate datele despre contractele jucătorilor, data semnării și data expirării acestora (coloanele DATA_SEMNARII, DATA_EXPIRARII). În această tabelă se află și coloanele ID_JUCATOR și ID_ANGAJAT, acestea fiind chei externe. În football jucătorii au contracte încheiate pe perioade determinate de timp, semnătura fiind dată de directorul sportiv al clubului sau de o altă persoană suplitoare în caz de acesta nu este disponibil.

A patra tabela este tabela sponsorilor, aceasta fiind o tabela importantă în ceea ce privesc finanțele clubului, aici regăsindu-se coloana cu numele sponsorilor, suma de bani cu care aceștia sponsorizează clubul, dar și data începerii și încheierii perioadei de sponsorizare. Coloana ID_ANGAJAT reprezintă angajatul care s-a ocupat de parteneriatul respectiv, de obicei acesta fiind directorului departamentului de marketing, în situații speciale acesta fiind înlocuit de manager.

Schema conceptuală pentru modelarea bazei de date



Legăturile sunt de tip 1:M:

Legătura dintre tabela ANGAJATI și tabela SPONSORI se face prin intermediul coloanei ID_ANGAJAT (cheie primară în tabela ANGAJATI), contractele încheiate cu sponsorii putând fi semnate de anumiți angajați, fiecare sponsor interacționând doar cu un singur angajat;

Legătura dintre tabela ANGAJATI și tabela CONTRACTE_JUCATORI se face prin intermediul coloanei ID_ANGAJAT (cheie primară în tabela ANGAJATI), contractele jucătorilor putând fii semnate de anumiți angajați, fiecare contract fiind semnat doar de un singur angajat.

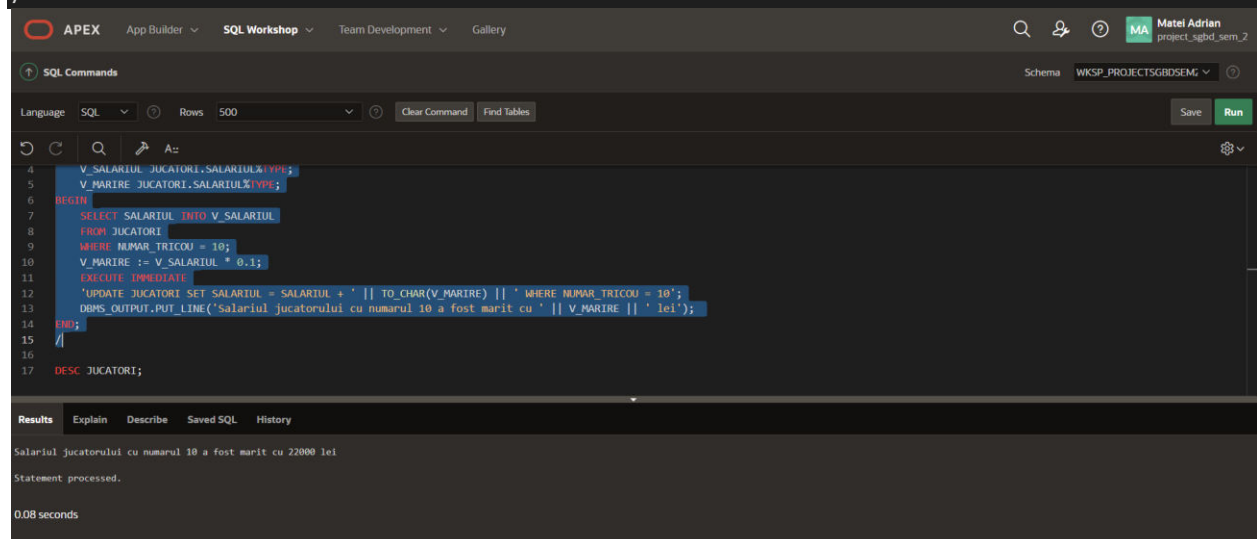
Legătura de tip 1:1:

Legătura dintre tabela JUCATORI și tabela CONTRACTE_JUCATORI se face prin intermediul coloanei ID_JUCATOR (cheie primară în tabela JUCATORI), fiecare jucator având un singur contract, iar fiecare contract aparține unui singur jucător.

Schema a ramas aceeaasi ca pe semestrul 1.

Interacțiunea cu serverul Oracle prin intermediul comenzilor SQL (LDD și LMD)

```
-- 1. Sa se mareasca salariul jucatorului cu numarul tricoului 10 cu 10%
DECLARE
    V_SALARIUL JUCATORI.SALARIUL%TYPE;
    V_MARIRE JUCATORI.SALARIUL%TYPE;
BEGIN
    SELECT SALARIUL INTO V_SALARIUL
    FROM JUCATORI
    WHERE NUMAR_TRICOU = 10;
    V_MARIRE := V_SALARIUL * 0.1;
    EXECUTE IMMEDIATE
        'UPDATE JUCATORI SET SALARIUL = SALARIUL + ' || TO_CHAR(V_MARIRE) || ' WHERE
NUMAR_TRICOU = 10';
    DBMS_OUTPUT.PUT_LINE('Salariul jucatorului cu numarul 10 a fost marit cu ' ||
V_MARIRE || ' lei');
END;
/
```



The screenshot shows the Oracle SQL Workshop interface. The SQL Commands window contains the script. The Results window shows the output: "Salariul jucatorului cu numarul 10 a fost marit cu 22000 lei". The status bar indicates "Statement processed." and "0.08 seconds".

```
- 2. Sa se stearga angajatul cu functia de ASISTENT MEDICAL, deoarece a fost
concediat
DECLARE
    V_FUNCTIE ANGAJATI.FUNCTIE%TYPE;
    V_COMANDA VARCHAR2(100);
BEGIN
    V_FUNCTIE := 'ASISTENT_MEDICAL';
```

```

V_COMANDA := 'DELETE FROM ANGAJATI WHERE FUNCTIE = ''V_FUNCTIE''';
EXECUTE IMMEDIATE V_COMANDA;
END;
/

```

```

-- 3. Sa se afiseze media salariilor angajatilor folosind PL/SQL
DECLARE
    V_MEDIE ANGAJATI.SALARIUL%TYPE;
BEGIN
    SELECT AVG(SALARIUL) INTO V_MEDIE
    FROM ANGAJATI;
    DBMS_OUTPUT.PUT_LINE('Media salariului angajatilor este de ' || V_MEDIE || ' lei'
);
END;
/

```

```

-
4. Sa se schimbe tipul coloanei POST din tabela JUCATORI din varchar2(50) in varc
har2(25)
BEGIN
    EXECUTE IMMEDIATE
    'ALTER TABLE JUCATORI MODIFY POST VARCHAR2(25)';
    DBMS_OUTPUT.PUT_LINE('S-a acutualizat cu succes');
END;
/

```

```

-
5. Sa se afiseze data pana la care ar fi valabil contractul cu ID 4 daca acesta
se va prelungi cu 6 luni
DECLARE
    V_CONTRACT CONTRACTE_JUCATORI%ROWTYPE;
BEGIN
    SELECT * INTO V_CONTRACT
    FROM CONTRACTE_JUCATORI WHERE ID_CONTRACT=4;
    DBMS_OUTPUT.PUT_LINE('Contractul cu id-
u1 ' || TO_CHAR(V_CONTRACT.ID_CONTRACT) || ' va expira pe data de ' || TO_CHAR(AD
D_MONTHS(V_CONTRACT.DATA_EXPIRARE, 6)) ||
    ' daca se va prelungi cu 6 luni. ');
END;
/

```

Structuri alternative și repetitive

```
- 6. Sa se reduca salariul angajatului cu id
4 dupa urmatoarea schema: daca salariul este intre 1000 si 5000 de lei, se va red
uce cu 5%
-
daca salariul este intre 5000 si 10000 de lei se va reduce cu 10%, iar daca este
mai mare de 10000 se va reduce cu 15%
DECLARE
    V_SALARIUL NUMBER;
    V_REducERE VARCHAR2(10);
BEGIN
    SELECT SALARIUL INTO V_SALARIUL FROM ANGAJATI WHERE ID_ANGAJAT = 4;
    DBMS_OUTPUT.PUT_LINE('Salariul angajatului inainte de reducere = ' || V_SALARIU
L);
    IF V_SALARIUL BETWEEN 0 AND 5000 THEN
        V_SALARIUL := V_SALARIUL * 0.95;
        V_REducERE := '5%';
    ELSIF V_SALARIUL BETWEEN 5000 AND 10000 THEN
        V_SALARIUL := V_SALARIUL * 0.9;
        V_REducERE := '10%';
    ELSE
        V_SALARIUL := V_SALARIUL * 0.85;
        V_REducERE := '15%';
    END IF;
    DBMS_OUTPUT.PUT_LINE('Salariul angajatului dupa o reducere de ' || V_REducERE
|| ' = ' || V_SALARIUL);
END;
/
```

```
-
7. Sa se afiseze lista angajatilor cu idul cuprins intre 2 si 5 odata, urmand ca
apoi sa se afiseze cei cu idul cuprins intre 4 si 7 odata
DECLARE
    V_ANGAJAT ANGAJATI%ROWTYPE;
BEGIN
    FOR i IN 1..2 LOOP
        IF i = 1 THEN
            DBMS_OUTPUT.PUT_LINE('Angajatii cu id-ul cuprins intre 2 si 5');
```

```

        FOR i IN 2..5 LOOP
            SELECT ID_ANGAJAT, NUME, PRENUME INTO V_ANGAJAT.ID_ANGAJAT, V_ANGAJAT.NUME, V_ANGAJAT.PRENUME FROM ANGAJATI WHERE ID_ANGAJAT = i;
            DBMS_OUTPUT.PUT_LINE(V_ANGAJAT.ID_ANGAJAT || ' ' || V_ANGAJAT.NUME || ' ' || V_ANGAJAT.PRENUME);
        END LOOP;
    ELSIF i = 2 THEN
        DBMS_OUTPUT.PUT_LINE('Angajatii cu id-ul cuprins intre 4 si 7');
        FOR i IN 4..7 LOOP
            SELECT ID_ANGAJAT, NUME, PRENUME INTO V_ANGAJAT.ID_ANGAJAT, V_ANGAJAT.NUME, V_ANGAJAT.PRENUME FROM ANGAJATI WHERE ID_ANGAJAT = i;
            DBMS_OUTPUT.PUT_LINE(V_ANGAJAT.ID_ANGAJAT || ' ' || V_ANGAJAT.NUME || ' ' || V_ANGAJAT.PRENUME);
        END LOOP;
    END IF;
END LOOP;
END;
/

```

```

-
8. Sa se studieze care este salariul angajatului cu id 4 daca este redus dupa urmatoarea schema: daca salariul este intre 1000 si 5000 de lei, se va reduce cu 5%
-
daca salariul este intre 5000 si 10000 de lei se va reduce cu 10%, iar daca este mai mare de 10000 se va reduce cu 15%. Se va folosi CASE
DECLARE
    V_SALARIUL NUMBER;
    V_REDCERE VARCHAR2(10);
BEGIN
    SELECT SALARIUL INTO V_SALARIUL FROM ANGAJATI WHERE ID_ANGAJAT = 4;
    DBMS_OUTPUT.PUT_LINE('Salariul angajatului inainte de reducere = ' || V_SALARIUL);
    CASE WHEN V_SALARIUL BETWEEN 0 AND 5000 THEN
        V_SALARIUL := V_SALARIUL * 0.95;
        V_REDCERE := '5%';
    WHEN V_SALARIUL BETWEEN 5000 AND 10000 THEN
        V_SALARIUL := V_SALARIUL * 0.9;
        V_REDCERE := '10%';
    ELSE
        V_SALARIUL := V_SALARIUL * 0.85;
        V_REDCERE := '15%';
    END CASE;

```



```

        DBMS_OUTPUT.PUT_LINE('Salariul angajatului dupa o reducere de ' || V_REDUCERE
        || ' = ' || V_SALARIUL);
END;
/
-- 9. Sa se prelungeasca cu o luna toate contractele jucatorilor
DECLARE
    V_COUNT NUMBER;
    i NUMBER;
BEGIN
    SELECT COUNT(*) INTO V_COUNT FROM CONTRACTE_JUCATORI;
    i := 1;
    WHILE(i < V_COUNT) LOOP
        UPDATE CONTRACTE_JUCATORI SET DATA_EXPIRARE = ADD_MONTHS(DATA_EXPIRARE, 1
) WHERE ID_CONTRACT = i;
        DBMS_OUTPUT.PUT_LINE('S-a prelungit cu o luna contractul cu id-
ul ' || TO_CHAR(i));
        i := i + 1;
    END LOOP;
END;

```

Tratarea excepțiilor

```

-
10.Sa se afiseze contractul jucatorului cu data semnarii 03/04/2020. Sa se trate
ze eroare aparuta in
-- cazul in care exista mai multe contracte cu aceeasi data de semnare.
DECLARE
    V_DATA CONTRACTE_JUCATORI.DATA_SEMNARE%TYPE;
    V_ID CONTRACTE_JUCATORI.ID_CONTRACT%TYPE;
BEGIN
    SELECT ID_CONTRACT, DATA_SEMNARE INTO V_ID, V_DATA FROM CONTRACTE_JUCATORI WH
ERE DATA_SEMNARE=TO_DATE('03/04/2020','MM/DD/YYYY');
    DBMS_OUTPUT.PUT_LINE('ID-
ul contractului semnat pe 03/04/2020 este ' || V_ID);
EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multe contracte semnate la aceasta data!
');
END;
/

```

```

-
11.Sa se trateze exceptia atunci cand se incearca impartirea la 0 a salariului u
nui angajat
DECLARE
    V_SALARIUL ANGAJATI.SALARIUL%TYPE;
BEGIN
    UPDATE ANGAJATI SET SALARIUL = SALARIUL/0 where ID_ANGAJAT=4;
EXCEPTION
    WHEN ZERO_DIVIDE THEN
        DBMS_OUTPUT.PUT_LINE('Exceptie! Impartirea la 0 nu se poate realiza!');
END;
/

```

```

-
12. Sa se afiseze salariul jucatorului cu numarul tricoului 15. Daca acesta nu e
xista, se fa trata exceptia
DECLARE
    V_SALARIUL JUCATORI.SALARIUL%TYPE;
BEGIN
    SELECT SALARIUL INTO V_SALARIUL FROM JUCATORI WHERE NUMAR_TRICOU = 15;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Exceptie! Acest jucator nu exista in baza de date!');
END;
/

```

```

-
13.Sa se afiseze contractul jucatorului cu data semnarii 03/04/2020. Sa se trate
ze eroare aparuta in
--
cazul in care exista mai multe contracte cu aceeasi data de semnare. In cazul in
troducerii unei date gresite, sa se trateze exceptia la mod general.
DECLARE
    V_DATA CONTRACTE_JUCATORI.DATA_SEMNARE%TYPE;
    V_ID CONTRACTE_JUCATORI.ID_CONTRACT%TYPE;
BEGIN
    SELECT ID_CONTRACT, DATA_SEMNARE INTO V_ID, V_DATA FROM CONTRACTE_JUCATORI WH
ERE DATA_SEMNARE=TO_DATE('03/04/2020','MM/DD/YYYY');
    DBMS_OUTPUT.PUT_LINE('ID-
ul contractului semnat pe 03/04/2020 este ' || V_ID);
EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multe contracte semnate la aceasta data!
');

```

```

        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('Exceptie!');
END;
/

```

```

-
14. Sa se mareasca cu 10% salariul angajatului cu functia de Traducator. Daca ac
easta functie nu exista, sa se trateze exceptia
DECLARE
    EX_FUNCTIE EXCEPTION;
BEGIN
    UPDATE ANGAJATI SET SALARIUL = SALARIUL * 0.1 WHERE FUNCTIE = 'Traducator';
    IF SQL%NOTFOUND THEN
        RAISE EX_FUNCTIE;
    END IF;
EXCEPTION
    WHEN EX_FUNCTIE THEN
        DBMS_OUTPUT.PUT_LINE('Functia nu exista!');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Exceptie!');
END;
/

```

```

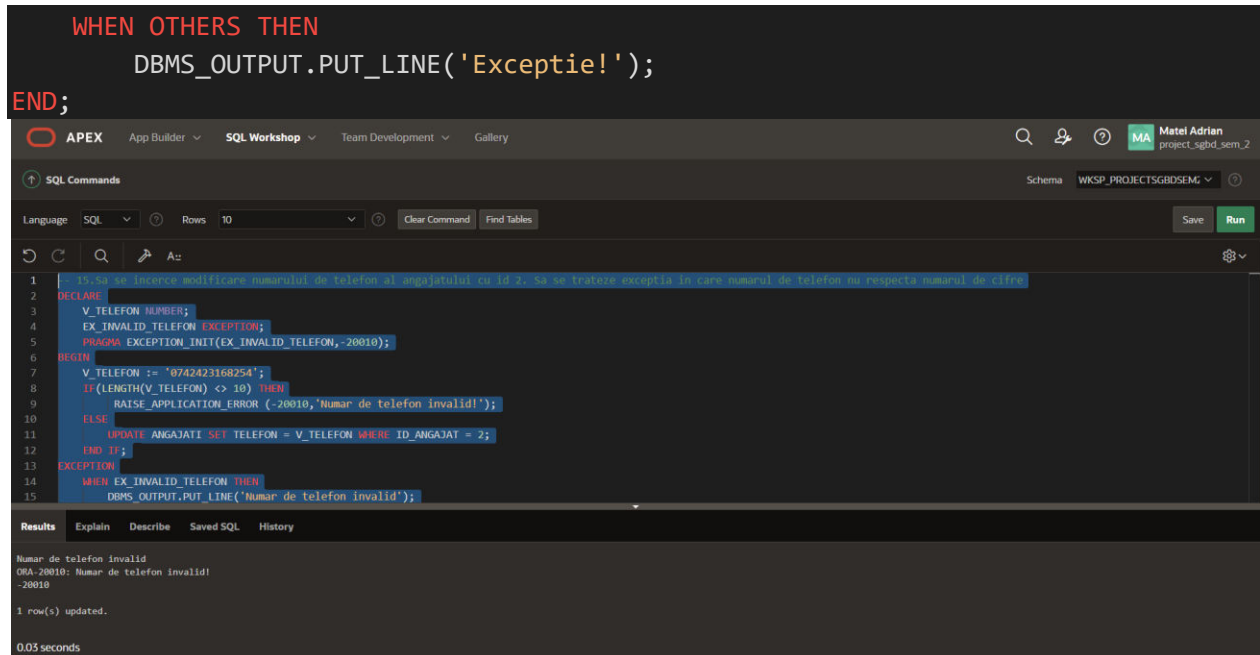
-
15.Sa se incerce modificare numarului de telefon al angajatului cu id 2. Sa se t
rateze exceptia in care numarul de telefon nu respecta numarul de cifre
DECLARE
    V_TELEFON NUMBER;
    EX_INVALID_TELEFON EXCEPTION;
    PRAGMA EXCEPTION_INIT(EX_INVALID_TELEFON,-20010);
BEGIN
    V_TELEFON := '0742423168254';
    IF(LENGTH(V_TELEFON) <> 10) THEN
        RAISE_APPLICATION_ERROR (-20010,'Numar de telefon invalid!');
    ELSE
        UPDATE ANGAJATI SET TELEFON = V_TELEFON WHERE ID_ANGAJAT = 2;
    END IF;
EXCEPTION
    WHEN EX_INVALID_TELEFON THEN
        DBMS_OUTPUT.PUT_LINE('Numar de telefon invalid');
        DBMS_OUTPUT.PUT_LINE(SQLERRM);
        DBMS_OUTPUT.PUT_LINE(SQLCODE);

```

```

    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Exceptie!');
END;

```



The screenshot shows the APEX SQL Workshop interface. The top bar includes the APEX logo and navigation links: App Builder, SQL Workshop, Team Development, and Gallery. The user is logged in as 'Matel Adrian' with the schema 'project_sgbd_sem_2'. The 'SQL Commands' tab is active, showing a PL/SQL script. The script defines an exception 'EX_INVALID_TELEFON' and attempts to update the 'TELEFON' column of the 'ANGAJATI' table for 'ID_ANGAJAT = 2'. It includes a length check and a custom error message. The 'Run' button is highlighted in green. Below the script, the 'Results' tab shows the execution output: 'Numar de telefon invalid', 'ORA-20010: Numar de telefon invalid! -20010', and '1 row(s) updated.' The execution took 0.03 seconds.

```

1  15:56 se incerca modificarea numarului de telefon al angajatului cu id 2. Sa se trateze exceptia in care numarul de telefon nu respecta numarul de cifre
2  DECLARE
3      V_TELEFON NUMBER;
4      EX_INVALID_TELEFON EXCEPTION;
5      PRAGMA EXCEPTION_INIT(EX_INVALID_TELEFON, -20010);
6  BEGIN
7      V_TELEFON := '0742423168254';
8      IF (LENGTH(V_TELEFON) <> 10) THEN
9          RAISE_APPLICATION_ERROR (-20010, 'Numar de telefon invalid!');
10     ELSE
11         UPDATE ANGAJATI SET TELEFON = V_TELEFON WHERE ID_ANGAJAT = 2;
12     END IF;
13 EXCEPTION
14     WHEN EX_INVALID_TELEFON THEN
15         DBMS_OUTPUT.PUT_LINE('Numar de telefon invalid');

```

Results

Numar de telefon invalid
ORA-20010: Numar de telefon invalid!
-20010

1 row(s) updated.

0.03 seconds

Gestionarea cursorilor

```

-
16. Sa se mareasca salariul angajatului cu idul introdus de la tastatura. Daca u
pdate-ul nu se va realiza, sa se afiseze un anunt in acest sens
ACCEPT a_id PROMPT 'Introduceti id-ul angajatului:'
DECLARE
    V_ID ANGAJATI.ID_ANGAJAT%TYPE := &a_id;
BEGIN
    UPDATE ANGAJATI SET SALARIUL = SALARIUL * 0.1 WHERE ID_ANGAJAT = V_ID;
    IF SQL%NOTFOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista angajatul cu acest cod');
    END IF;
END;
/

```

```

17.Sa se micsoareze cu 5% salariile peste 10000 lei si sa se afiseze numarul salar
iilor micsoarate
BEGIN
UPDATE ANGAJATI SET SALARIUL = SALARIUL * 0.95 WHERE SALARIUL > 10000;
DBMS_OUTPUT.PUT_LINE('S-au micsoarat ' || TO_CHAR(SQL%ROWCOUNT) || ' salarii.');
```

```

-
18. Sa se afiseze angajatii care au salariul peste 10.000 de lei folosind un cursor explicit
DECLARE
    CURSOR ANG_TOP_CURSOR IS SELECT ID_ANGAJAT, NUME, SALARIUL FROM ANGAJATI WHERE SALARIUL > 10000;
    ANG angajati%ROWTYPE;
BEGIN
    DBMS_OUTPUT.PUT_LINE('Lista angajatilor cu salariul peste 10.000 de lei');
    OPEN ANG_TOP_CURSOR;
    LOOP
        FETCH ANG_TOP_CURSOR INTO ANG.ID_ANGAJAT, ANG.NUME, ANG.SALARIUL;
        EXIT WHEN ANG_TOP_CURSOR%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Angajatul: ' || ANG.ID_ANGAJAT || ' ' || ANG.NUME || ' Salariul: ' || ANG.SALARIUL);
    END LOOP;
    CLOSE ANG_TOP_CURSOR;
END;
/

-- 19. Sa se afiseze datele jucatorului cu id-ul transmis prin parametru intr-un cursor explicit
DECLARE
    CURSOR C_JUCATOR (P_ID NUMBER) IS SELECT NUME, POST, SALARIUL FROM JUCATORI WHERE ID_JUCATOR = P_ID;
    REC_JUCATOR C_JUCATOR%ROWTYPE;
BEGIN
    IF NOT C_JUCATOR%ISOPEN THEN
        OPEN C_JUCATOR(1);
    END IF;

    FETCH C_JUCATOR INTO REC_JUCATOR;
    DBMS_OUTPUT.PUT_LINE('Jucatorul gasit pentru id-ul dat: ' || REC_JUCATOR.NUME || ' ' || REC_JUCATOR.POST || ' ' || REC_JUCATOR.SALARIUL);
    CLOSE C_JUCATOR;
END;
/

```

Funcții, proceduri, includerea acestora în pachete

```

-
20. Creati o functie care returneaza valoarea salariului unui angajat conform id-ului acestuia primit ca parametru

```

```

CREATE OR REPLACE FUNCTION GET_SAL(P_ID ANGAJATI.ID_ANGAJAT%TYPE)
RETURN NUMBER IS
    V_SAL ANGAJATI.SALARIUL%TYPE := 0;
BEGIN
    SELECT SALARIUL INTO V_SAL FROM ANGAJATI WHERE ID_ANGAJAT = P_ID;
    RETURN V_SAL;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN NULL;
END GET_SAL;

DECLARE
    V_SAL ANGAJATI.SALARIUL%TYPE;
BEGIN
    V_SAL := GET_SAL(1);
    DBMS_OUTPUT.PUT_LINE('Salariul corespunzator angajatului cu id-
ul specificat este: ' || V_SAL);
END;

```

```

-
21. Sa se creeze o functie care verifica daca exista un anumit sponsor in tabela
    cu sponsori primind drept parametru numele acestuia
CREATE OR REPLACE FUNCTION VALID_SPONSOR (P_SPONSOR SPONSORI.NUME_SPONSOR%TYPE)
RETURN BOOLEAN IS
    V_ID SPONSORI.ID_SPONSOR%TYPE;
BEGIN
    SELECT ID_SPONSOR INTO V_ID FROM SPONSORI
    WHERE NUME_SPONSOR = P_SPONSOR;
    RETURN TRUE;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN FALSE;
END;

DECLARE
    V_VALID BOOLEAN;
    V_NUME_SPONSOR SPONSORI.NUME_SPONSOR%TYPE;
BEGIN
    V_NUME_SPONSOR := UPPER('DACSIA');
    IF VALID_SPONSOR(UPPER(V_NUME_SPONSOR)) THEN
        DBMS_OUTPUT.PUT_LINE('Sponsorul ' || V_NUME_SPONSOR || ' exista in tabela
cu sponsori');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Sponsorul ' || V_NUME_SPONSOR || ' nu exista in tab
ela cu sponsori');
    END IF;
END;

```

```

    END IF;
END;

```

```

-
22. Sa se creeze o functie care returneaza suma totala a sponsorizarilor din tab
ela cu sponsori
CREATE OR REPLACE FUNCTION GET_SUMA_TOTALA_SPONSORIZARI
RETURN NUMBER IS
    V_SUMA SPONSORI.SUMA_SPONSORIZARE%TYPE;
BEGIN
    SELECT SUM(SUMA_SPONSORIZARE) INTO V_SUMA FROM SPONSORI;
    RETURN V_SUMA;
END;

DECLARE
    V_SUMA SPONSORI.SUMA_SPONSORIZARE%TYPE;
BEGIN
    V_SUMA := GET_SUMA_TOTALA_SPONSORIZARI;
    DBMS_OUTPUT.PUT_LINE('Suma totala a sponsorizarilor este de ' || V_SUMA || '
lei.');
```

```

-- 23. Creeaza o procedura care face acelasi lucru ca si functia de mai sus
CREATE OR REPLACE PROCEDURE SUMA_SPONSORIZARI_TOTALA (P_SUMA OUT NUMBER) AS
BEGIN
    SELECT SUM(SUMA_SPONSORIZARE) INTO P_SUMA FROM SPONSORI;
END;

DECLARE
    V_SUMA SPONSORI.SUMA_SPONSORIZARE%TYPE;
BEGIN
    SUMA_SPONSORIZARI_TOTALA(V_SUMA);
    DBMS_OUTPUT.PUT_LINE('Suma totala a sponsorizarilor este de ' || V_SUMA || '
lei.');
```

```

-
24. Sa se creeze o procedura ce permite reducerea salariului unui anumit angajat
cu un anumit procent trimis ca si parametru
CREATE OR REPLACE PROCEDURE REDUCERE_SALARIU (P_ID IN ANGAJATI.ID_ANGAJAT%TYPE, P
_PROCENT IN FLOAT) AS
BEGIN
```

```

    IF P_PROCENT < 1 THEN
        UPDATE ANGAJATI SET SALARIUL = SALARIUL -
(SALARIUL * P_PROCENT) WHERE ID_ANGAJAT = P_ID;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Procent invalid!');
    END IF;
END;

BEGIN
    REDUCERE_SALARIU(1, 0.1);
END;

```

```

-
25. Sa se creezeze un pachet care contine functiile si procedurile specifice angaj
atilor
CREATE OR REPLACE PACKAGE ANG_PACKAGE AS
    FUNCTION GET_SAL(P_ID ANGAJATI.ID_ANGAJAT%TYPE) RETURN NUMBER;
    PROCEDURE REDUCERE_SALARIU(P_ID ANGAJATI.ID_ANGAJAT%TYPE, P_PROCENT FLOAT);
END ANG_PACKAGE;

CREATE OR REPLACE PACKAGE BODY ANG_PACKAGE AS
    FUNCTION GET_SAL(P_ID ANGAJATI.ID_ANGAJAT%TYPE)
    RETURN NUMBER IS
        V_SAL ANGAJATI.SALARIUL%TYPE := 0;
    BEGIN
        SELECT SALARIUL INTO V_SAL FROM ANGAJATI WHERE ID_ANGAJAT = P_ID;
        RETURN V_SAL;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            RETURN NULL;
    END;

    PROCEDURE REDUCERE_SALARIU (P_ID IN ANGAJATI.ID_ANGAJAT%TYPE, P_PROCENT IN FL
OAT) AS
    BEGIN
        IF P_PROCENT < 1 THEN
            UPDATE ANGAJATI SET SALARIUL = SALARIUL -
(SALARIUL * P_PROCENT) WHERE ID_ANGAJAT = P_ID;
        ELSE
            DBMS_OUTPUT.PUT_LINE('Procent invalid!');
        END IF;
    END;
END ANG_PACKAGE;

```



```

DECLARE
    V_SAL ANGAJATI.SALARIUL%TYPE;
BEGIN
    V_SAL := ANG_PACKAGE.GET_SAL(1);
    DBMS_OUTPUT.PUT_LINE('Salariul corespunzator angajatului cu id-
ul specificat este: ' || V_SAL);
    ANG_PACKAGE.REDUCERE_SALARIU(1, 0.1);
END;

```

Declanșatori

```

-
26. Sa se creeze un trigger care afiseaza un status de fiecare data cand una din
comenzile INSERT, DELETE, UPDATE este rulata in tabela JUCATORI
CREATE OR REPLACE TRIGGER DISPLAY_OPERATION_TRIGGER
AFTER INSERT OR UPDATE OR DELETE ON JUCATORI
BEGIN
    IF INSERTING THEN
        DBMS_OUTPUT.PUT_LINE('Comanda de inserare in tabela JUCATORI a fost rulat
a cu succes!');
    ELSIF UPDATING THEN
        DBMS_OUTPUT.PUT_LINE('Comanda de actualizare in tabela JUCATORI a fost ru
lata cu succes!');
    ELSIF DELETING THEN
        DBMS_OUTPUT.PUT_LINE('Comanda de stergere in tabela JUCATORI a fost rulat
a cu succes!');
    END IF;
END;

UPDATE JUCATORI SET SALARIUL = SALARIUL * 1.1 WHERE ID_JUCATOR = 13133;

```

```

-
27. Sa se creeze un trigger care se declanseaza inaintea fiecarui update al sala
riului unui angajat si adauga userul si data la care s-a facut operatiunea
CREATE TABLE LOG_TABLE
(USER_ID VARCHAR(50),
LOGON_DATE DATE);

CREATE OR REPLACE TRIGGER LOG_SAL_UPDATE

```

```

BEFORE UPDATE OF SALARIUL ON ANGAJATI
BEGIN
    INSERT INTO LOG_TABLE(USER_ID, LOGON_DATE)
    VALUES(USER,SYSDATE);
END;

-- verificare
UPDATE ANGAJATI SET SALARIUL = SALARIUL * 1.3 WHERE ID_ANGAJAT = 3;
SELECT * FROM LOG_TABLE;

```

```

-
28. Sa se creeze un trigger care arunca o eroare daca valoarea ce se doreste sa
modifice salariul sau sa fie introdusa in acest camp este mai mica decat minimul
pe economie
CREATE OR REPLACE TRIGGER TOO_SMALL_SALARY_TRIGGER
    BEFORE INSERT OR UPDATE OF SALARIUL ON ANGAJATI FOR EACH ROW
BEGIN
    IF :NEW.SALARIUL < 2300 THEN
        RAISE_APPLICATION_ERROR (-
20204, 'Angajatul trebuie sa aiba asigurat macar minimul pe economie!');
    END IF;
END;

UPDATE ANGAJATI SET SALARIUL = 1000 WHERE ID_ANGAJAT IN (1, 2, 3);
-
29. Sa se creeze un trigger care arunca o exceptie daca la incercarea modificari
i id-
ului angajatului cu o functie ce permite semnarea contractelor se introduce un id
al unui angajat fara functie ce permite semnarea contractelor
CREATE OR REPLACE TRIGGER FUNCTIE_INVALIDA_TRIGGER
BEFORE UPDATE OF ID_ANGAJAT ON CONTRACTE_JUCATORI FOR EACH ROW
DECLARE
    V_ID_ANGAJAT ANGAJATI.ID_ANGAJAT%TYPE;
    FUNCTIE_INVALIDA EXCEPTION;
    PRAGMA EXCEPTION_INIT(FUNCTIE_INVALIDA, -20404);
BEGIN
    IF :NEW.ID_ANGAJAT NOT IN (1, 2) THEN
        RAISE_APPLICATION_ERROR(-
20404, 'Doar managerul si directorul sportiv au drept de semnare!');
    END IF;
EXCEPTION
    WHEN FUNCTIE_INVALIDA THEN
        DBMS_OUTPUT.PUT_LINE('Doar managerul si directorul sportiv au drept de se
mnare!');

```

```

RAISE;
END;

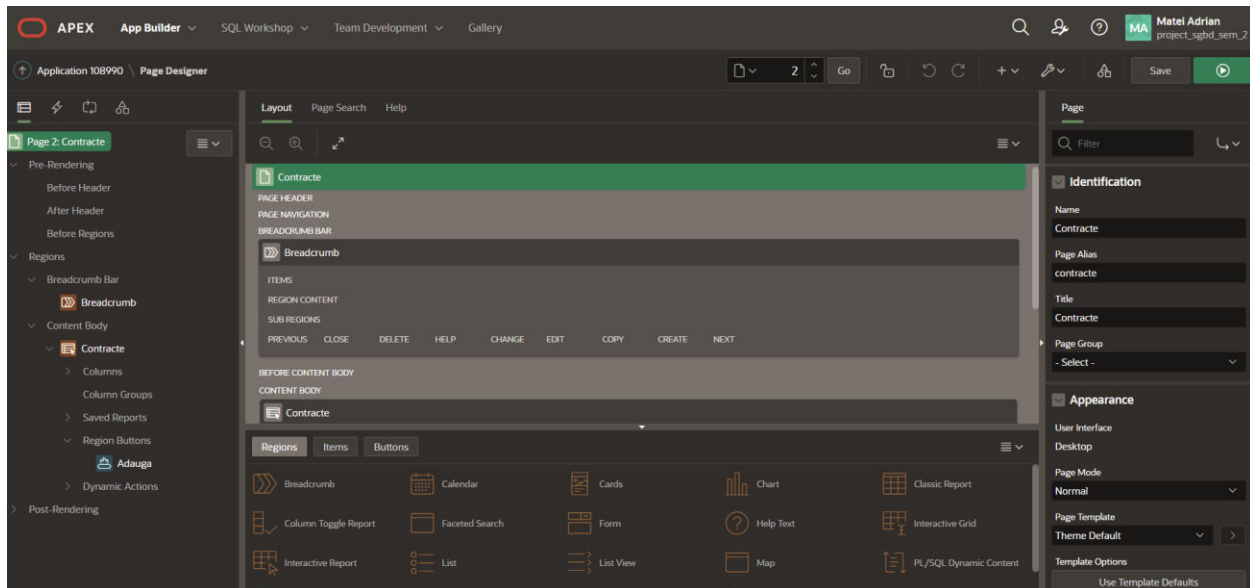
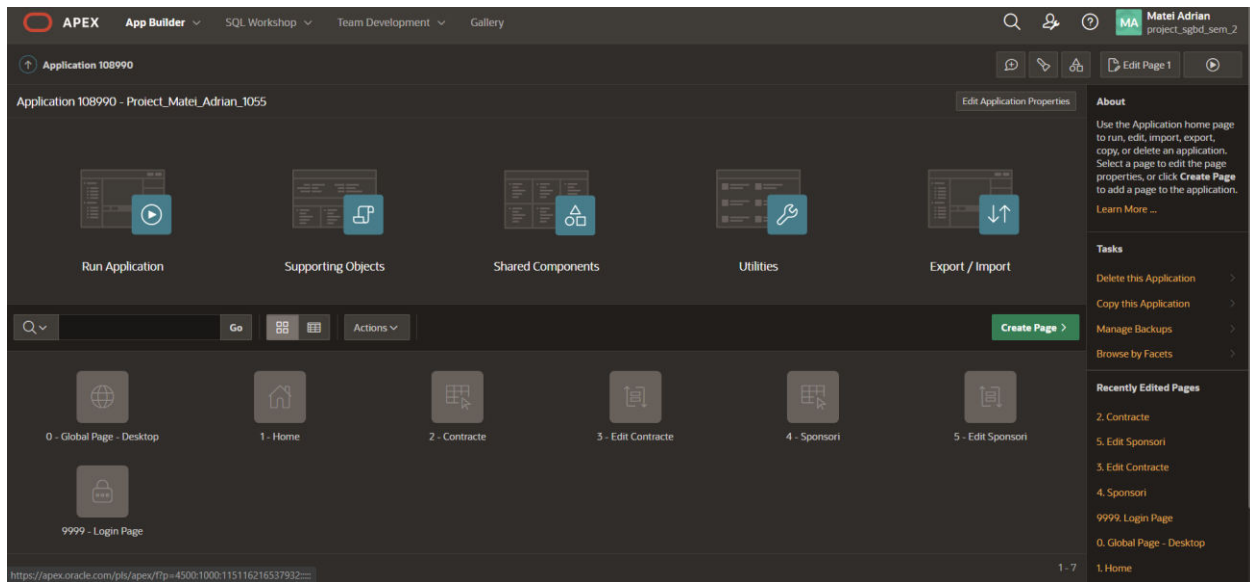
UPDATE CONTRACTE_JUCATORI SET ID_ANGAJAT = 5 WHERE ID_CONTRACT = 1;

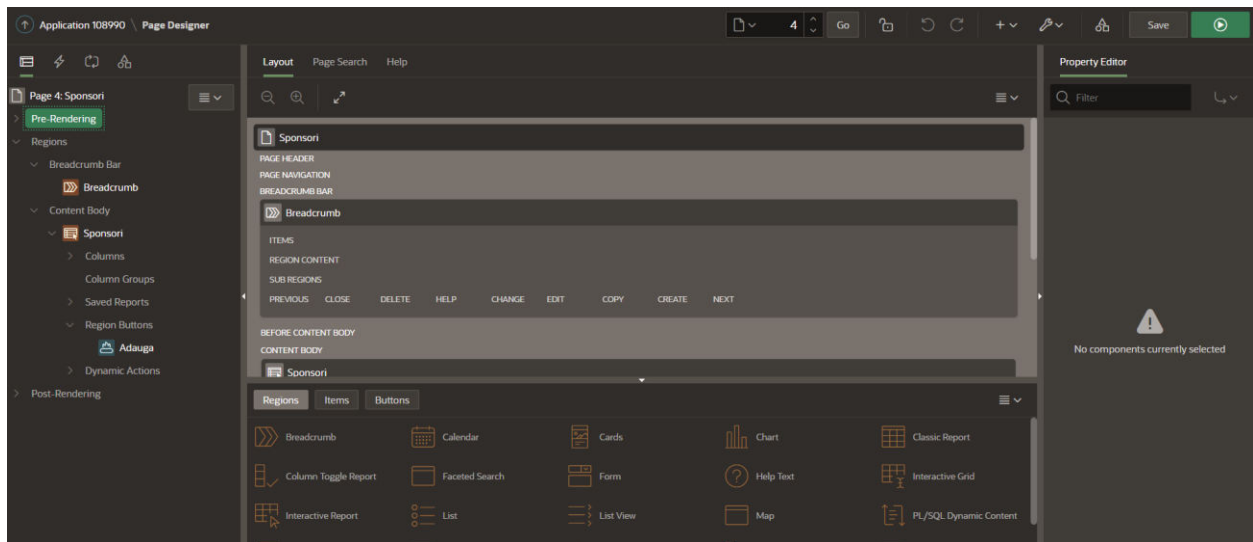
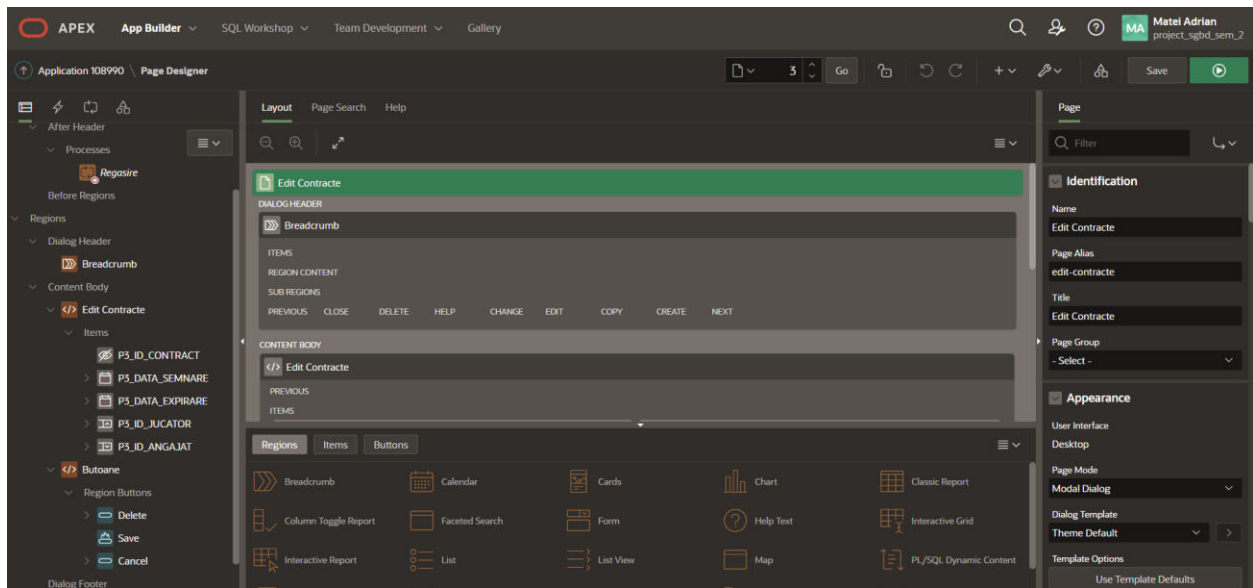
```

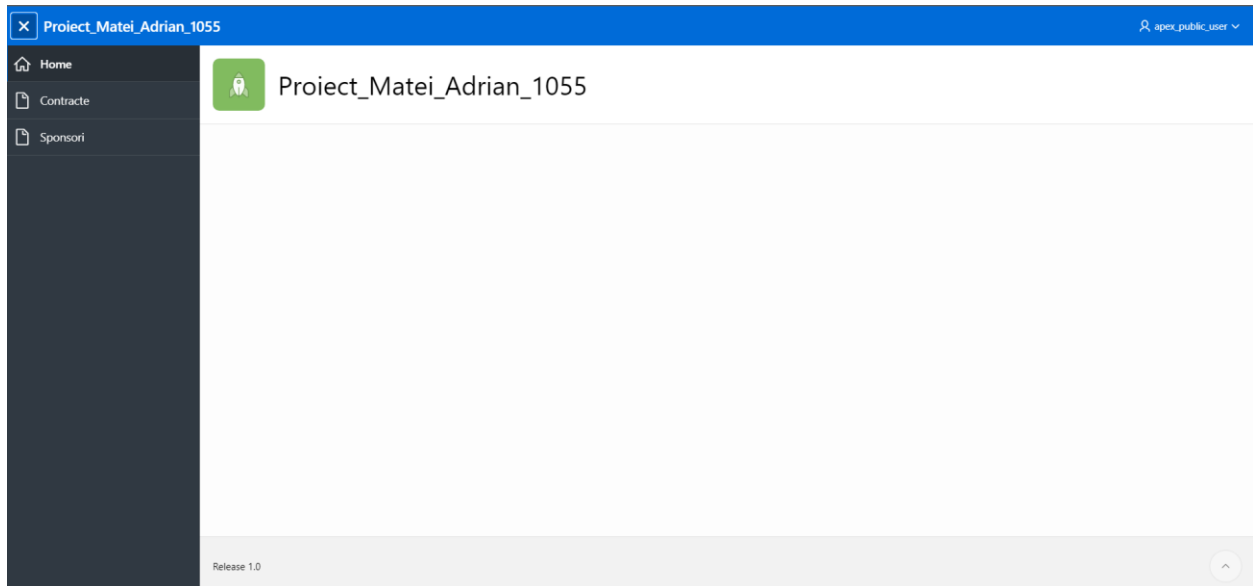
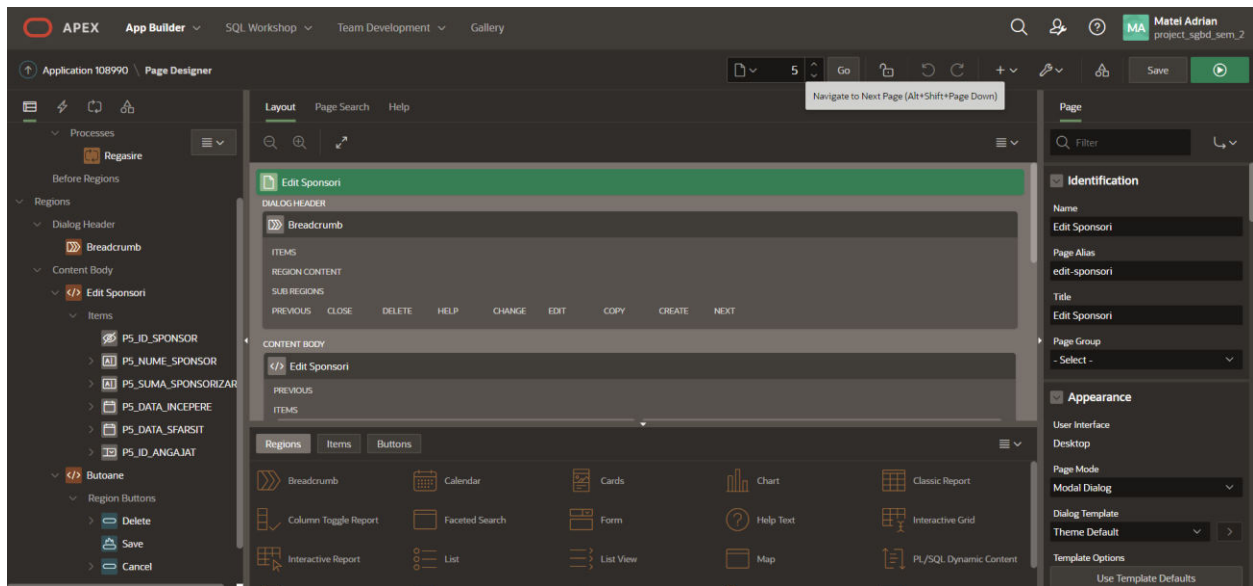
The screenshot displays the APEX SQL Workshop interface. The main area shows a PL/SQL trigger named `FUNCTION_INVALIDA_TRIGGER` and an `UPDATE` statement. The trigger is designed to handle errors when updating the `ID_ANGAJAT` field. It includes a `WHEN` clause that checks if the new `ID_ANGAJAT` is 1 or 2, and if so, it raises an application error with the message "Doar managerul si directorul sportiv au drept de semnare!". The update statement attempts to set `ID_ANGAJAT` to 5 for the contract with `ID_CONTRACT` 1. The results pane shows an error: `ORA-06512: at 'WKSP_PROJECTSGRDESEM2.FUNCTION_INVALIDA_TRIGGER', line 12`, indicating the trigger was fired during the update operation.

Aplicație APEX

Link: [Proiect Matei Adrian 1055 \(oracle.com\)](https://www.oracle.com/aex/proj/matei-adrian-1055/)







Project_Matei_Adrian_1055

apex_public_user

Contracte

	Data Semnare	Data Expirare	Jucator	Semnata de
	4/1/2020	11/1/2025	LIONEL MESSI	MATEI
	3/6/2020	8/4/2023	IKER CASILLAS	PUSCASU
	3/4/2020	9/4/2023	CARLES PUYOL	PUSCASU
	3/5/2020	8/5/2024	GERARD PIQUE	PUSCASU
	6/8/2020	9/8/2023	DANI ALVES	MATEI
	3/8/2020	8/8/2023	JORDI ALBA	PUSCASU
	3/10/2020	9/10/2024	ANDRES INIESTA	PUSCASU
	3/11/2020	8/11/2024	XAVI HERNANDEZ	PUSCASU
	3/11/2020	9/11/2023	PEDRI GONZALEZ	PUSCASU
	3/11/2020	8/11/2025	CRISTIANO RONALDO	MATEI
	3/12/2020	10/14/2024	ZLATAN IBRAHIMOVIC	PUSCASU
	5/19/2021	7/12/2023	ADRIAN MUTU	MATEI

1 - 12

Project_Matei_Adrian_1055

apex_public_user

Contracte

	Data Semnare	Data Expirare	Jucator	Semnata de
	4/1/2020	11/1/2025	LIONEL MESSI	MATEI
	3/6/2020	8/4/2023	IKER CASILLAS	PUSCASU
	3/4/2020	9/4/2023	CARLES PUYOL	PUSCASU
	3/5/2020	8/5/2024	GERARD PIQUE	PUSCASU
	6/8/2020	9/8/2023	DANI ALVES	MATEI
	3/8/2020	8/8/2023	JORDI ALBA	PUSCASU
	3/10/2020	9/10/2024	ANDRES INIESTA	PUSCASU
	3/11/2020	8/11/2024	XAVI HERNANDEZ	PUSCASU
	3/11/2020	9/11/2023	PEDRI GONZALEZ	PUSCASU
	3/11/2020	8/11/2025	CRISTIANO RONALDO	MATEI
	3/12/2020	10/14/2024	ZLATAN IBRAHIMOVIC	PUSCASU
	5/19/2021	7/12/2023	ADRIAN MUTU	MATEI

1 - 12

Contracte \ Edit Contracte

Data Semnare

4/1/2020

Data Expirare

11/1/2025

Jucator

LIONEL MESSI

Semnata de

MATEI

Cancel

Delete

Save

Proiect_Matei_Adrian_1055

apex_public_user

Contracte

Q

Go

Rows 50

Actions

Adauga

	Data Semnare				Semnat de
	4/1/2020				MATEI
	3/6/2020				PUSCASU
	3/4/2020				PUSCASU
	3/5/2020				PUSCASU
	6/8/2020				MATEI
	3/8/2020				PUSCASU
	3/10/2020				PUSCASU
	3/11/2020				PUSCASU
	3/11/2020		8/11/2025		MATEI
	3/12/2020		10/14/2024		PUSCASU
	5/19/2021		7/12/2023		MATEI

1 - 12

Edit Contracte

Contracte Edit Contracte

Data Semnare

Data Expirare

Va rugam selectati data semnarii

Va rugam sa introduceti data expirarii

Jucator

Semnat de

Va rugam selectati numele jucatorului

Va rugam alegeți angajatul care a semnat contractul

Cancel

Delete

Save

Proiect_Matei_Adrian_1055

apex_public_user

Sponsori

Q

Go

Rows 50

Actions

Adauga

	Nume Sponsor	Suma Sponsorizare	Data Incepere	Data Sfarsit	Incheiata de
	DACIA	500000	4/5/2020	3/6/2025	MATEI
	INTERSPORT	305000	5/18/2021	6/9/2022	MATEI
	MAXBET	2132127	6/24/2021	5/14/2024	MATEI
	DEDEMAN	600000	3/12/2020	3/12/2026	DOBROTICI
	UNIBET	40000	5/17/2021	5/9/2023	DOBROTICI
	ROLEX	8646739	5/18/2021	6/18/2026	DOBROTICI

1 - 6

Proiect_Matei_Adrian_1055 apex_public_user

Sponsori

Search: Go Rows: 50

	Nume Sponsor	Data Sfasit	Incheiata de
	DACIA	3/6/2025	MATEI
	INTERSPORT	6/9/2022	MATEI
	MAXBET	5/14/2024	MATEI
	DEDEMAN	3/12/2026	DOBROTICI
	UNIBET	5/9/2023	DOBROTICI
	ROLEX	6/18/2026	DOBROTICI

1 - 6

Release 1.0

Edit Sponsori

Sponsori \ Edit Sponsori

Nume Sponsor: DACIA

Suma Sponsorizare: 500000

Data Incepere: 4/5/2020

Data Sfasit: 3/6/2025

Incheiata de: MATEI

Cancel Delete Save

Proiect_Matei_Adrian_1055 apex_public_user

Sponsori

Search: Go Rows: 50

	Nume Sponsor	Data Sfasit	Incheiata de
	DACIA	3/6/2025	MATEI
	INTERSPORT	6/9/2022	MATEI
	MAXBET	5/14/2024	MATEI
	DEDEMAN	3/12/2026	DOBROTICI
	UNIBET	5/9/2023	DOBROTICI
	ROLEX	6/18/2026	DOBROTICI

1 - 6

Release 1.0

Edit Sponsori

Sponsori \ Edit Sponsori

Nume Sponsor

Suma Sponsorizare

Data Incepere

Data Sfasit

Incheiata de

Cancel Delete Save