--1

CREATE OR REPLACE PROCEDURE Nume\_Titluri

IS

    TYPE Lista\_Titluri IS TABLE OF carte\_DP.titlu%TYPE INDEX BY BINARY\_INTEGER;

    L\_Titluri Lista\_Titluri;

BEGIN

    SELECT c.titlu

    BULK COLLECT INTO L\_Titluri

    FROM CARTE\_DP c

    JOIN IMPRUMUTA\_DP i ON (c.id\_carte = i.cod\_carte)

    ORDER BY i.dataim ASC;

    FOR i in 1..L\_Titluri.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE(L\_Titluri(i));

    END LOOP;

END;

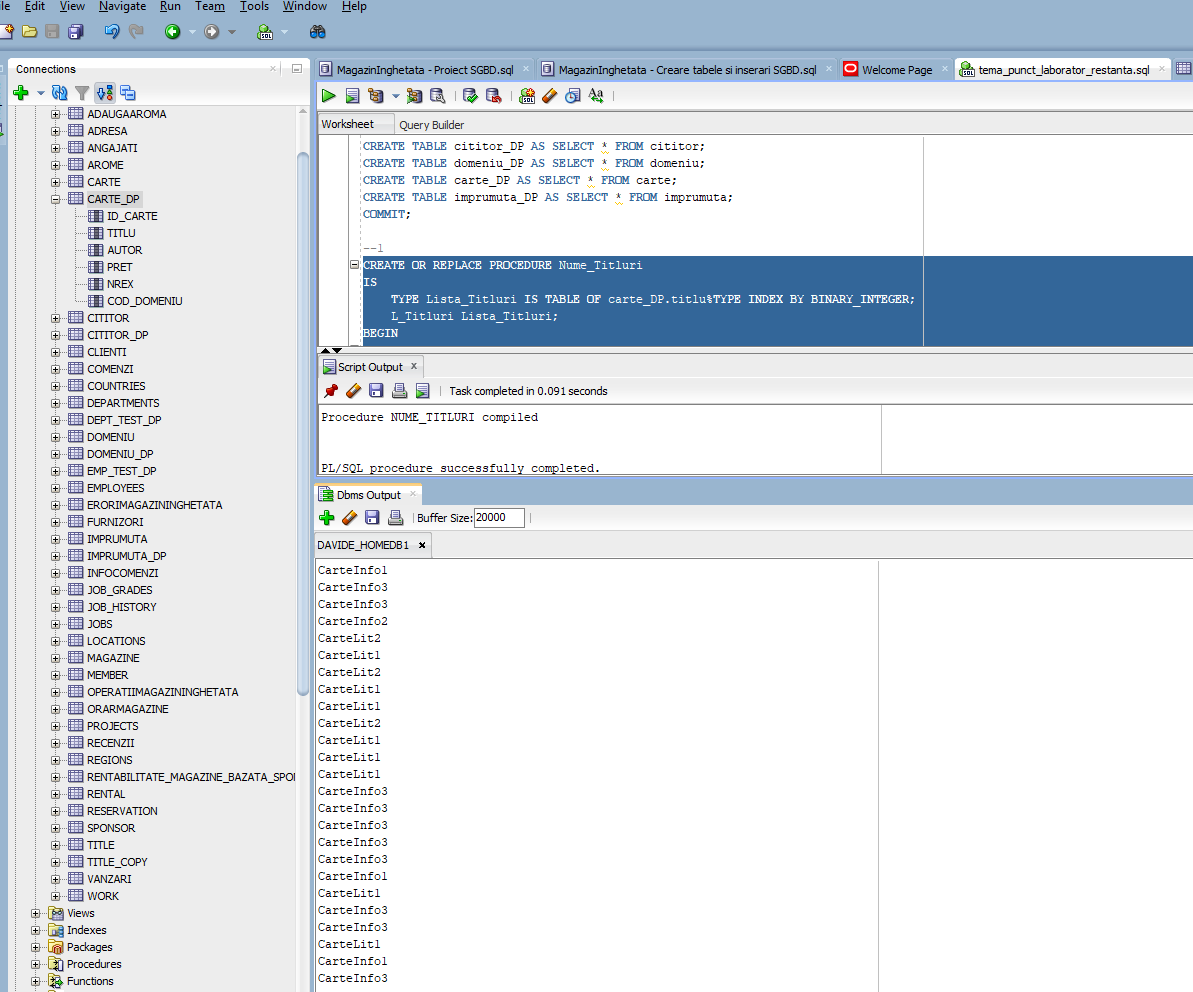
/

BEGIN

    Nume\_Titluri;

END;

/



--2

CREATE OR REPLACE PROCEDURE Numar\_Imprumuturi

IS

    TYPE Lista\_Imprumuturi IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

    NR\_Imprumuturi Lista\_Imprumuturi;

BEGIN

    SELECT COUNT(i.cod\_carte)

    BULK COLLECT INTO NR\_Imprumuturi

    FROM IMPRUMUTA\_DP i

    GROUP BY i.cod\_carte;

    FOR i in 1..NR\_Imprumuturi.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE(NR\_Imprumuturi(i));

    END LOOP;

END;

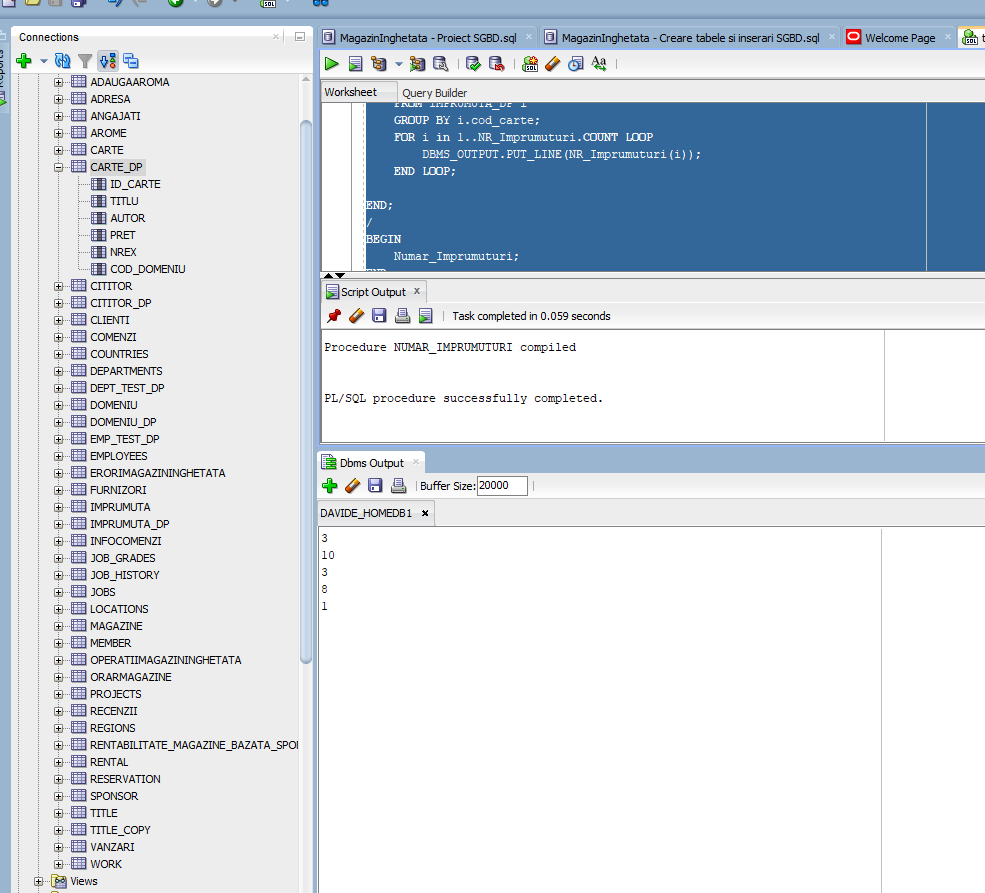
/

BEGIN

    Numar\_Imprumuturi;

END;

/



--3

CREATE OR REPLACE PROCEDURE Domeniu\_Detalii

IS

    TYPE detalii\_domeniu is RECORD(

        denumire domeniu.denumire%TYPE,

        numar\_carti NUMBER,

        pret\_mediu NUMBER,

        nr\_total\_carti CARTE\_DP.nrex%TYPE

    );

    TYPE Lista\_Domenii IS TABLE OF detalii\_domeniu INDEX BY BINARY\_INTEGER;

    Domenii Lista\_Domenii;

BEGIN

    SELECT d.denumire, COUNT(c.id\_carte), AVG(c.pret), SUM(c.nrex)

    BULK COLLECT INTO Domenii

    FROM DOMENIU\_DP d

    JOIN CARTE\_DP c on (d.id\_domeniu = c.cod\_domeniu)

    GROUP BY d.denumire;

    FOR i in 1..Domenii.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE('Denumire '|| Domenii(i).denumire || ' Numar Carti '|| Domenii(i).numar\_carti || ' Pret mediu domeniu '|| Domenii(i).pret\_mediu || ' Numar total carti '|| Domenii(i).nr\_total\_carti);

    END LOOP;

END;

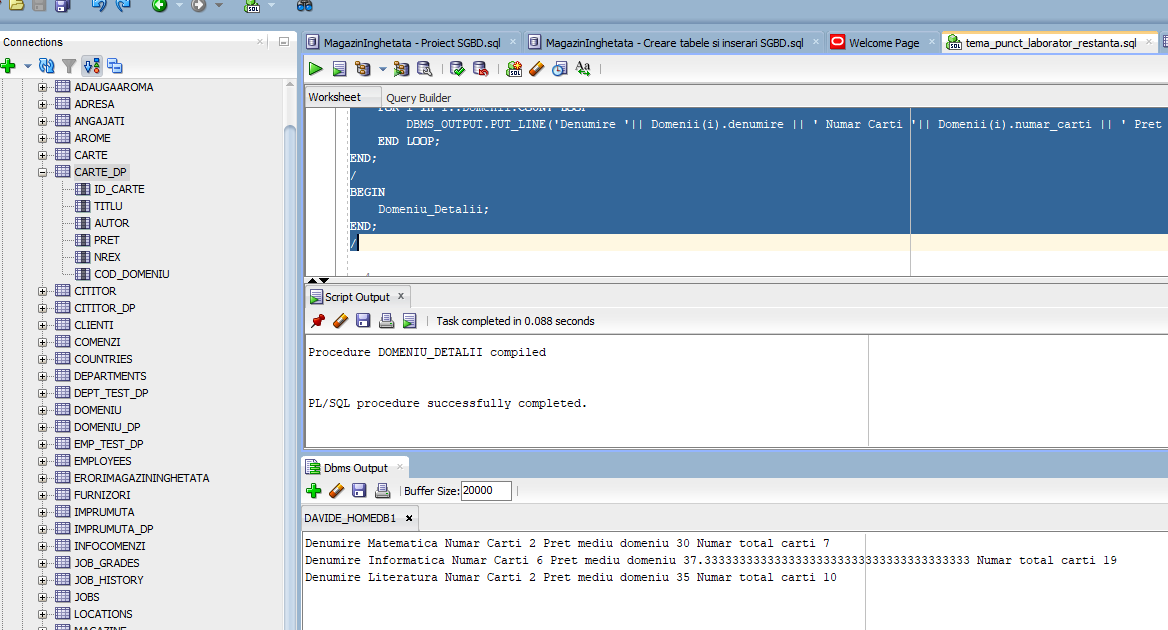
/

BEGIN

    Domeniu\_Detalii;

END;

/



--4

CREATE OR REPLACE PROCEDURE Numar\_Imprumuturi

IS

    TYPE Lista\_Imprumuturi IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

    L\_Imprumuturi Lista\_Imprumuturi;

BEGIN

    SELECT COUNT(cod\_carte)

    BULK COLLECT INTO L\_Imprumuturi

    FROM IMPRUMUTA\_DP

    GROUP BY cod\_carte;

    FOR i in 1..L\_Imprumuturi.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE(L\_Imprumuturi(i));

    END LOOP;

END;

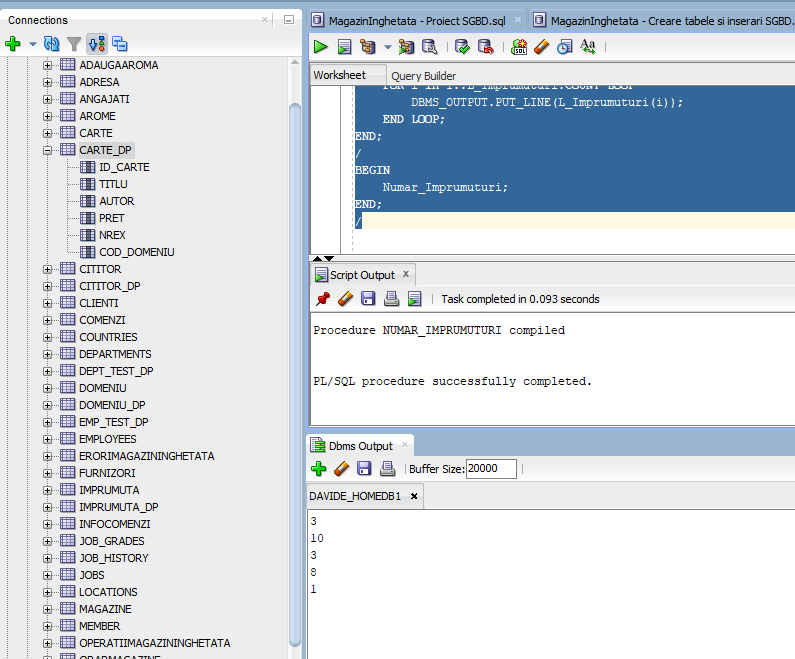
/

BEGIN

    Numar\_Imprumuturi;

END;

/



--5

CREATE OR REPLACE PROCEDURE Numar\_NeImprumuturi

IS

    TYPE Lista\_NeImprumuturi IS TABLE OF CARTE\_DP.titlu%TYPE INDEX BY BINARY\_INTEGER;

    L\_NeImprumuturi Lista\_NeImprumuturi;

BEGIN

    SELECT c.titlu

    BULK COLLECT INTO L\_NeImprumuturi

    FROM CARTE\_DP c

    LEFT JOIN IMPRUMUTA\_DP i ON (c.id\_carte = i.cod\_carte)

    WHERE i.cod\_carte IS NULL;

    FOR i IN 1..L\_NeImprumuturi.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE(L\_NeImprumuturi(i));

    END LOOP;

END;

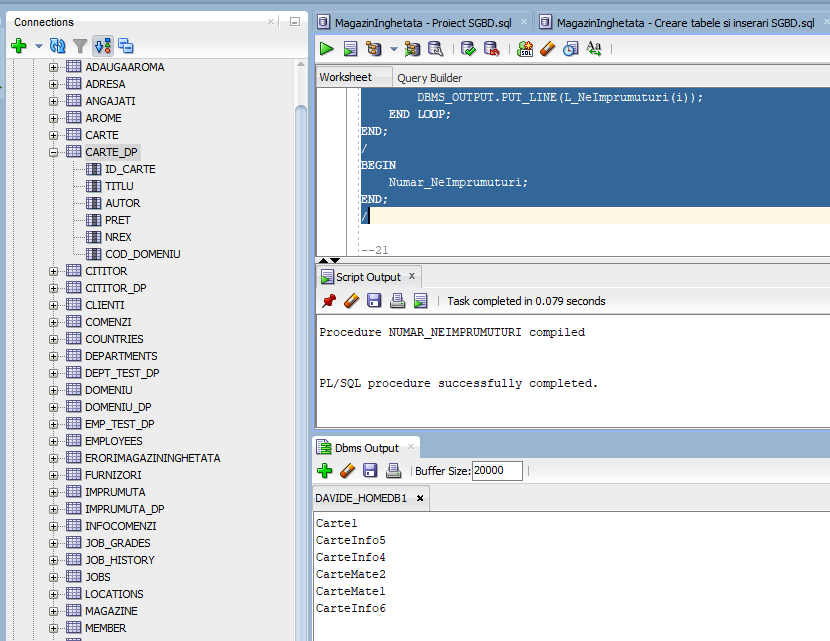
/

BEGIN

    Numar\_NeImprumuturi;

END;

/



--21

CREATE OR REPLACE PROCEDURE Cititor\_Detalii

IS

    CURSOR cititori\_cursor IS

        SELECT DISTINCT c.nume

        FROM CITITOR\_DP c

        JOIN IMPRUMUTA\_DP i ON (c.id\_cititor = i.cod\_cititor)

        JOIN CARTE\_DP carte ON (i.cod\_carte = carte.id\_carte)

        JOIN DOMENIU\_DP d ON (d.id\_domeniu = carte.cod\_domeniu)

        WHERE EXTRACT(DAY FROM c.data\_nasterii) = 23 AND carte.nrex = 1 AND d.denumire = 'Informatica';

    nume cititor.nume%TYPE;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Nascut pe 23 si a imprumutat carti de un singur exemplar din sectiunea Informatica');

    OPEN cititori\_cursor;

    LOOP

        FETCH cititori\_cursor INTO nume;

        EXIT WHEN cititori\_cursor%NOTFOUND;

        DBMS\_OUTPUT.PUT\_LINE(nume);

    END LOOP;

    CLOSE cititori\_cursor;

END;

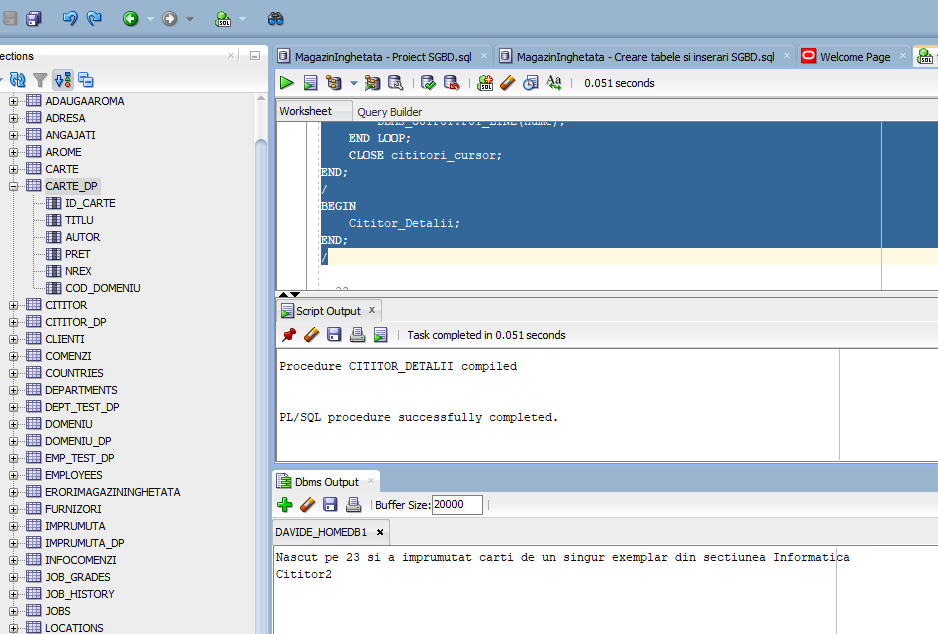
/

BEGIN

    Cititor\_Detalii;

END;

/



--22

CREATE OR REPLACE PROCEDURE Carti\_Scumpe

IS

    CURSOR carti\_cursor IS

        SELECT titlu, pret

        FROM CARTE\_DP

        ORDER BY pret DESC

        FETCH FIRST 4 ROWS ONLY;

    titlu carte\_DP.titlu%TYPE;

    pret carte\_DP.pret%TYPE;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('4 cele mai scumpe carti');

    OPEN carti\_cursor;

    LOOP

        FETCH carti\_cursor INTO titlu, pret;

        EXIT WHEN carti\_cursor%NOTFOUND;

        DBMS\_OUTPUT.PUT\_LINE(titlu||' '||pret);

    END LOOP;

    CLOSE carti\_cursor;

END;

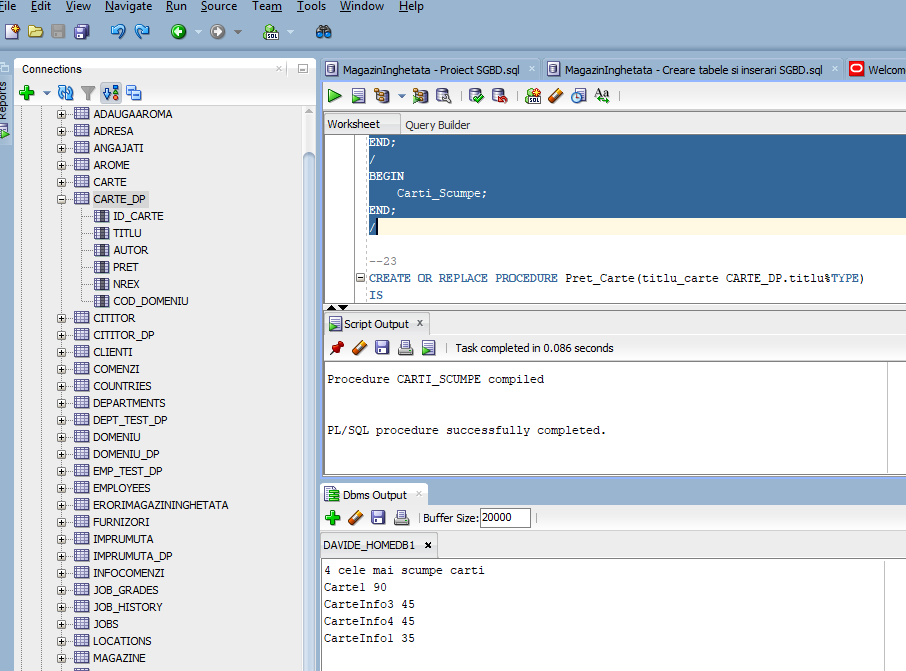
/

BEGIN

    Carti\_Scumpe;

END;

/



--23

CREATE OR REPLACE PROCEDURE Pret\_Carte(titlu\_carte CARTE\_DP.titlu%TYPE)

IS

    pret\_carte CARTE\_DP.pret%TYPE;

BEGIN

    SELECT pret

    INTO pret\_carte

    FROM CARTE\_DP

    WHERE titlu = titlu\_carte;

    IF pret\_carte > 60 THEN

        DBMS\_OUTPUT.PUT\_LINE('Scumpa, peste 60');

    ELSIF pret\_carte BETWEEN 35 AND 60 THEN

        DBMS\_OUTPUT.PUT\_LINE('Decent, intre 35 si 60');

    ELSIF pret\_carte < 35 THEN

        DBMS\_OUTPUT.PUT\_LINE('Ieftina, sub 35');

    END IF;

END;

/

BEGIN

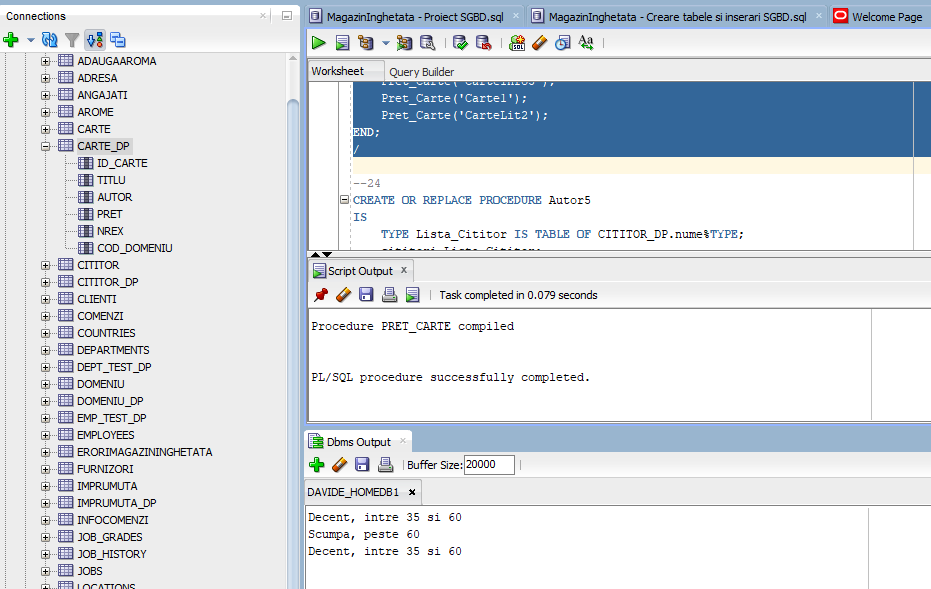
    Pret\_Carte('CarteInfo5');

    Pret\_Carte('Carte1');

    Pret\_Carte('CarteLit2');

END;

/

****

--24

CREATE OR REPLACE PROCEDURE Autor5

IS

    TYPE Lista\_Cititor IS TABLE OF CITITOR\_DP.nume%TYPE;

    cititori Lista\_Cititor;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Cine a imprumutat tot ce a scris Autor5');

    SELECT c.nume

    BULK COLLECT INTO cititori

    FROM cititor\_DP c

    JOIN imprumuta\_DP i ON c.id\_cititor = i.cod\_cititor

    JOIN carte\_DP carte ON i.cod\_carte = carte.id\_carte

    WHERE carte.autor = 'Autor5'

    GROUP BY c.nume, c.id\_cititor

    HAVING COUNT(DISTINCT carte.id\_carte) =

    (   SELECT COUNT(DISTINCT id\_carte)

        FROM carte\_DP

        WHERE autor = 'Autor5' );

    FOR i IN 1..cititori.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE(cititori(i));

    END LOOP;

END;

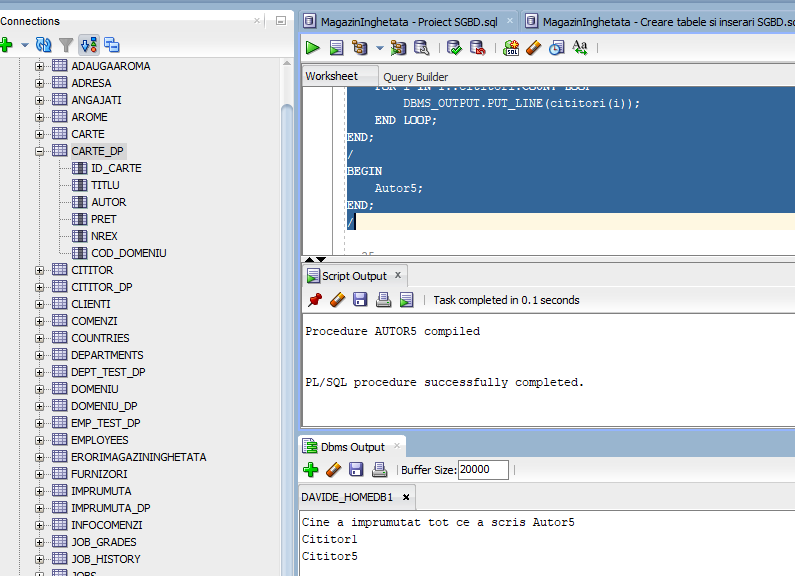
/

BEGIN

    Autor5;

END;

/

****

--25

CREATE OR REPLACE PROCEDURE Neimprumutat\_Info AS

    TYPE Lista\_Titluri IS TABLE OF CARTE\_DP.titlu%TYPE;

    titluri Lista\_Titluri;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Neimprumutat de Info de nimeni');

    SELECT c.titlu

    BULK COLLECT INTO titluri

    FROM carte\_DP c

    JOIN domeniu\_DP d ON (d.id\_domeniu = c.cod\_domeniu)

    WHERE c.autor IS NULL AND d.denumire = 'Informatica'

    AND c.id\_carte NOT IN

    (   SELECT i.cod\_carte

        FROM imprumuta i

        WHERE i.dataef IS NULL );

    FOR i IN 1..titluri.COUNT LOOP

        DBMS\_OUTPUT.PUT\_LINE(titluri(i));

    END LOOP;

END;

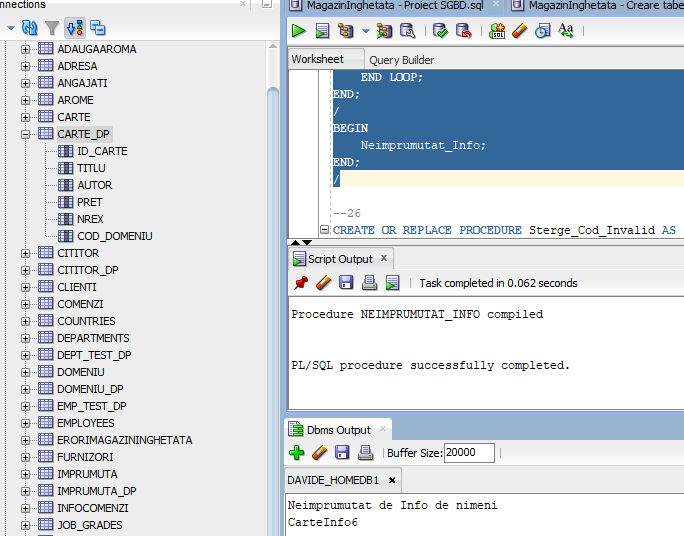
/

BEGIN

    Neimprumutat\_Info;

END;

/

****

--26

CREATE OR REPLACE PROCEDURE Sterge\_Cod\_Invalid AS

BEGIN

    UPDATE carte\_DP

    SET cod\_domeniu = NULL

    WHERE cod\_domeniu NOT IN (SELECT id\_domeniu FROM domeniu);

END;

/

BEGIN

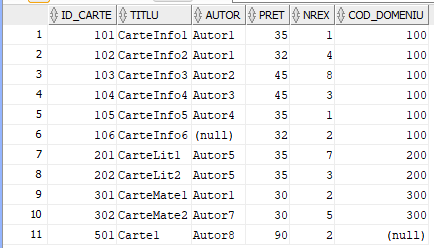
    INSERT INTO CARTE\_DP VALUES(606, 'BioMecanica', 'Davide', 55, 4, 4352);

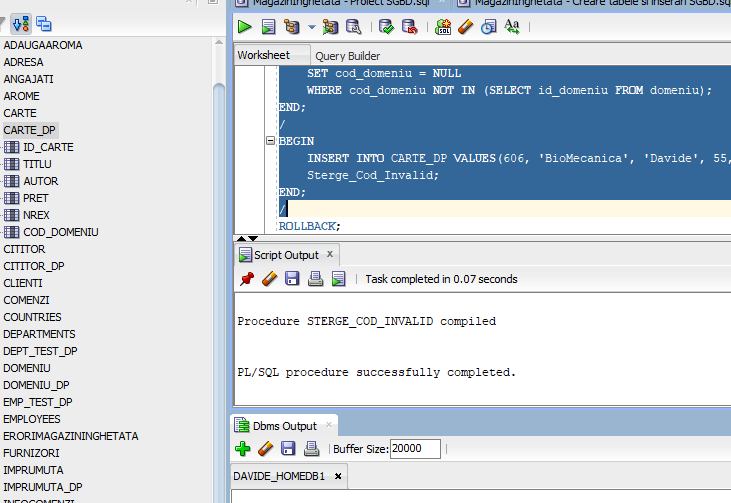
    Sterge\_Cod\_Invalid;

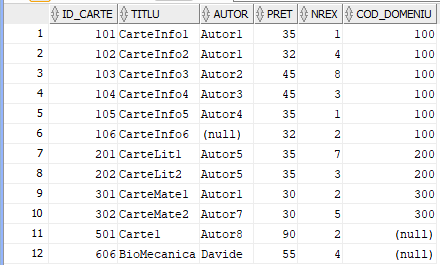
END;

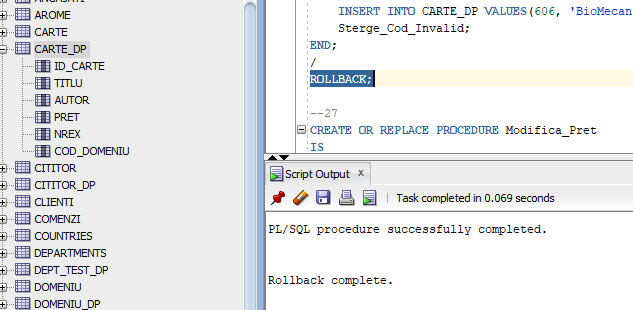
/

ROLLBACK;

****

****

****

****

--27

CREATE OR REPLACE PROCEDURE Modifica\_Pret

IS

    pret\_max NUMBER;

BEGIN

    SELECT MAX(c.pret)

    INTO pret\_max

    FROM carte\_DP c

    JOIN domeniu\_DP d ON (d.id\_domeniu = c.cod\_domeniu)

    WHERE d.denumire = 'Literatura';

    UPDATE carte\_DP

    SET pret = pret\_max

    WHERE autor = 'Autor5';

END;

/

BEGIN

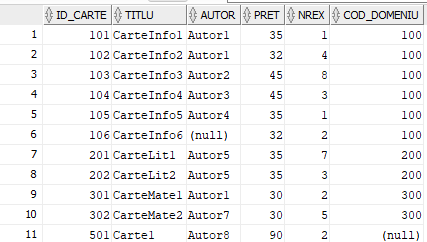
    INSERT INTO CARTE\_DP VALUES(606, 'DEX', 'Autor7', 100, 4, 200);

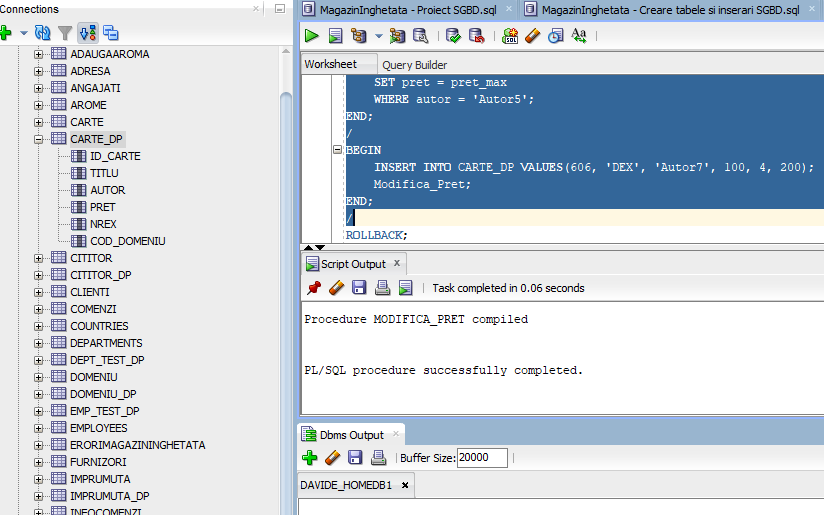
    Modifica\_Pret;

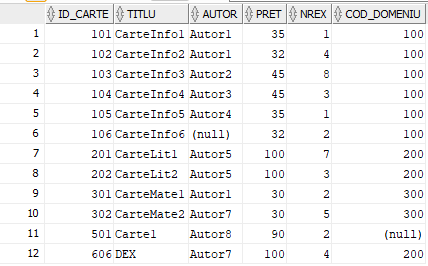
END;

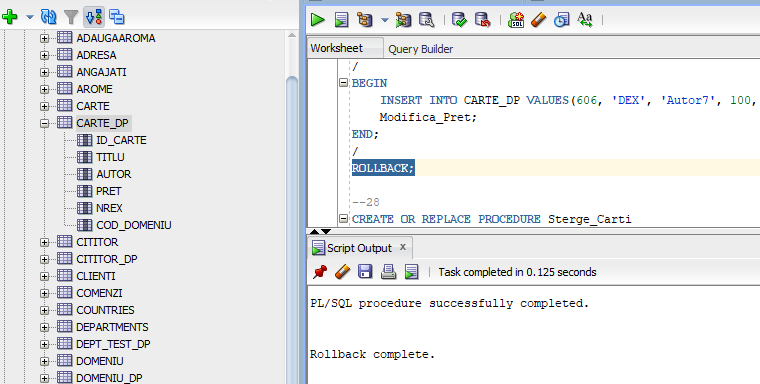
/

ROLLBACK;

****

****

****

****

--28

CREATE OR REPLACE PROCEDURE Sterge\_Carti

IS

    pret\_mediu NUMBER;

BEGIN

    SELECT AVG(pret)

    INTO pret\_mediu

    FROM carte\_DP;

    DELETE FROM carte\_DP

    WHERE id\_carte NOT IN

    (   SELECT cod\_carte

        FROM imprumuta\_DP )

    AND pret < pret\_mediu;

END;

/

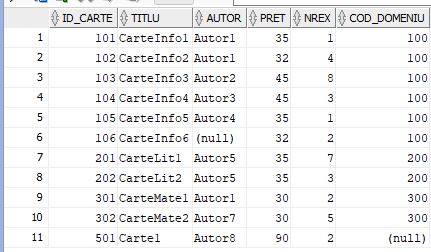
BEGIN

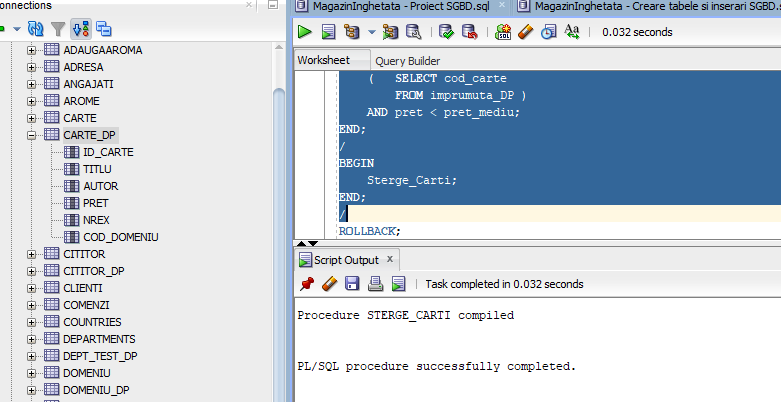
    Sterge\_Carti;

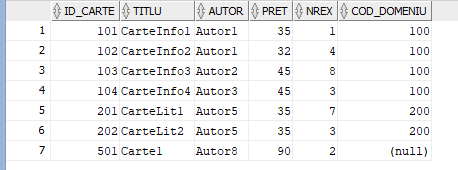
END;

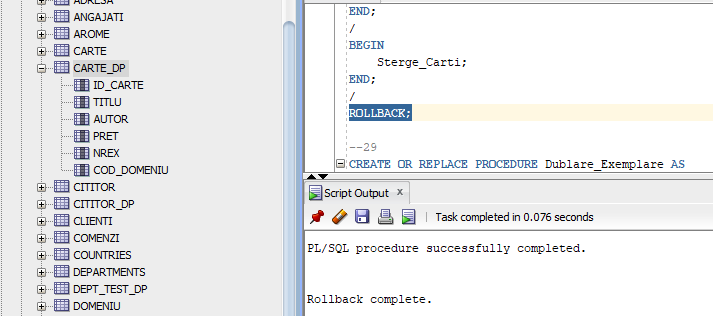
/

ROLLBACK;

****

****

****

****

--29

CREATE OR REPLACE PROCEDURE Dublare\_Exemplare AS

    max\_imprumuturi NUMBER;

BEGIN

    SELECT MAX(COUNT(cod\_carte))

    INTO max\_imprumuturi

    FROM imprumuta\_DP

    GROUP BY cod\_carte;

    UPDATE carte\_DP

    SET nrex = nrex\*2

    WHERE id\_carte IN

    (   SELECT cod\_carte

        FROM imprumuta\_DP

        GROUP BY cod\_carte

        HAVING COUNT(\*) = max\_imprumuturi );

END;

/

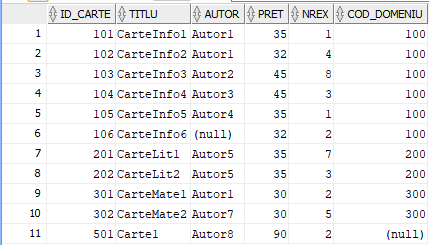
BEGIN

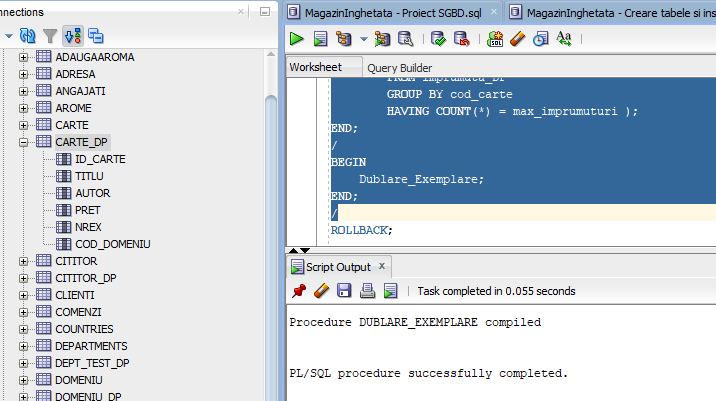
    Dublare\_Exemplare;

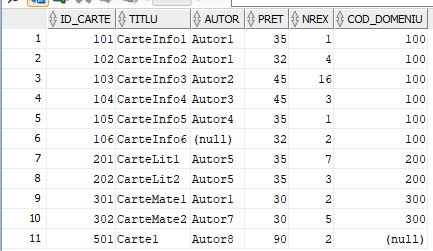
END;

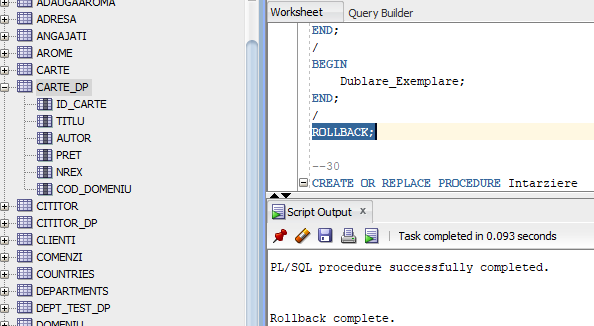
/

ROLLBACK;

****

****

****

****

--30

CREATE OR REPLACE PROCEDURE Intarziere

IS

    suma\_totala NUMBER := 0;

    suma\_curenta NUMBER := 0;

    numar\_intarzieri NUMBER := 0;

    suma\_pe\_carte NUMBER := 0;

    CURSOR cititor\_cursor IS

        SELECT id\_cititor, nume

        FROM cititor\_DP;

    id\_cititor cititor\_DP.id\_cititor%TYPE;

    nume cititor\_DP.nume%TYPE;

    CURSOR imprumuturi\_cursor(id\_primit cititor\_DP.id\_cititor%TYPE) IS

        SELECT i.dataim, i.dataef

        FROM imprumuta\_DP i

        WHERE i.cod\_cititor = id\_primit;

    data\_imprumut imprumuta\_DP.dataim%TYPE;

    data\_restituire imprumuta\_DP.dataef%TYPE;

BEGIN

    OPEN cititor\_cursor;

    LOOP

        FETCH cititor\_cursor INTO id\_cititor, nume;

        EXIT WHEN cititor\_cursor%NOTFOUND;

        DBMS\_OUTPUT.PUT\_LINE('Cititorul ' ||nume);

        suma\_totala := 0;

        suma\_curenta := 0;

        numar\_intarzieri := 0;

        suma\_pe\_carte := 0;

        OPEN imprumuturi\_cursor(id\_cititor);

        LOOP

            FETCH imprumuturi\_cursor INTO data\_imprumut, data\_restituire;

            EXIT WHEN imprumuturi\_cursor%NOTFOUND;

            IF SYSDATE > data\_imprumut THEN

                numar\_intarzieri := numar\_intarzieri + 1;

            END IF;

            IF data\_restituire IS NOT NULL THEN

                IF data\_restituire > data\_imprumut THEN

                    suma\_totala := suma\_totala+(TRUNC(data\_restituire)-TRUNC(data\_imprumut))\*2;

                END IF;

            ELSE

                IF SYSDATE > data\_imprumut THEN

                    suma\_curenta := suma\_curenta+(TRUNC(SYSDATE)-TRUNC(data\_imprumut))\*2;

                END IF;

            END IF;

            IF data\_restituire IS NOT NULL THEN

                suma\_pe\_carte := suma\_pe\_carte+(TRUNC(data\_restituire)-TRUNC(data\_imprumut))\*2;

            ELSE

                suma\_pe\_carte := suma\_pe\_carte+(TRUNC(SYSDATE)-TRUNC(data\_imprumut))\*2;

            END IF;

        END LOOP;

        CLOSE imprumuturi\_cursor;

        DBMS\_OUTPUT.PUT\_LINE('Suma totala data din cauza intarzierii '||suma\_totala);

        DBMS\_OUTPUT.PUT\_LINE('Suma curenta de plata din cauza intarzierii '||suma\_curenta);

        DBMS\_OUTPUT.PUT\_LINE('De cate ori a intarziat '||numar\_intarzieri);

        DBMS\_OUTPUT.PUT\_LINE('Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) '||suma\_pe\_carte);

        DBMS\_OUTPUT.PUT\_LINE('');

    END LOOP;

    CLOSE cititor\_cursor;

END;

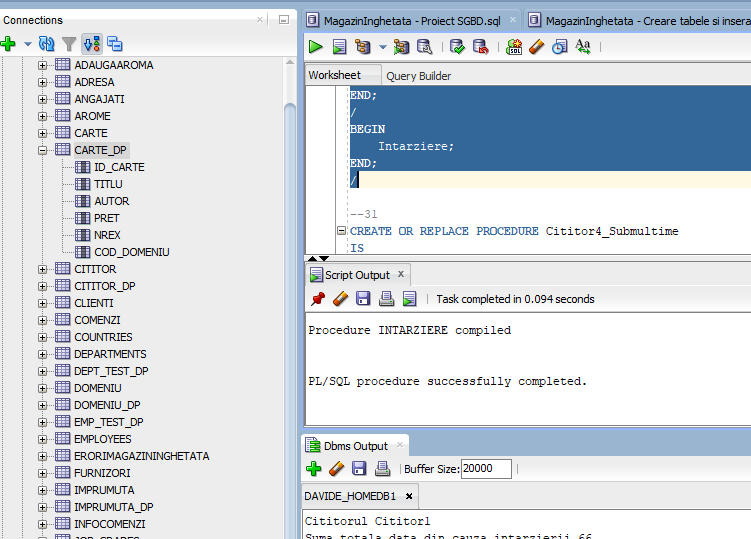
/

BEGIN

    Intarziere;

END;

/

****

Cititorul Cititor1

Suma totala data din cauza intarzierii 66

Suma curenta de plata din cauza intarzierii 5238

De cate ori a intarziat 4

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 5304

Cititorul Cititor2

Suma totala data din cauza intarzierii 58

Suma curenta de plata din cauza intarzierii 5238

De cate ori a intarziat 5

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 5296

Cititorul Cititor3

Suma totala data din cauza intarzierii 0

Suma curenta de plata din cauza intarzierii 0

De cate ori a intarziat 0

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 0

Cititorul Cititor4

Suma totala data din cauza intarzierii 6

Suma curenta de plata din cauza intarzierii 5272

De cate ori a intarziat 3

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 5278

Cititorul Cititor5

Suma totala data din cauza intarzierii 0

Suma curenta de plata din cauza intarzierii 10384

De cate ori a intarziat 3

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 10384

Cititorul Cititor6

Suma totala data din cauza intarzierii 0

Suma curenta de plata din cauza intarzierii 5214

De cate ori a intarziat 2

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 5214

Cititorul Cititor7

Suma totala data din cauza intarzierii 0

Suma curenta de plata din cauza intarzierii 100

De cate ori a intarziat 1

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 100

Cititorul Cititor8

Suma totala data din cauza intarzierii 0

Suma curenta de plata din cauza intarzierii 5094

De cate ori a intarziat 1

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 5094

Cititorul Cititor9

Suma totala data din cauza intarzierii 0

Suma curenta de plata din cauza intarzierii 5174

De cate ori a intarziat 2

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 5174

Cititorul Cititor10

Suma totala data din cauza intarzierii 40

Suma curenta de plata din cauza intarzierii 0

De cate ori a intarziat 4

Suma totala toate cartile imprumutate (inclusiv taxa pe intarziere) 40

--31

CREATE OR REPLACE PROCEDURE Cititor4\_Submultime

IS

    nume cititor\_DP.nume%TYPE;

    id\_cititor cititor\_DP.id\_cititor%TYPE;

    CURSOR cititori\_cursor IS

        SELECT DISTINCT c.id\_cititor, c.nume

        FROM cititor\_DP c

        WHERE c.id\_cititor IN

        (   SELECT i.cod\_cititor

            FROM imprumuta\_DP i

            WHERE i.cod\_carte IN

            (   SELECT cod\_carte

                FROM imprumuta\_DP

                WHERE cod\_cititor = (SELECT id\_cititor FROM cititor\_DP WHERE nume = 'Cititor4') )

            GROUP BY i.cod\_cititor

            HAVING COUNT(DISTINCT i.cod\_carte) =

            (   SELECT COUNT(DISTINCT cod\_carte)

                FROM imprumuta\_DP

                WHERE cod\_cititor = (SELECT id\_cititor FROM cititor\_DP WHERE nume = 'Cititor4') ) );

    carte imprumuta\_DP.cod\_carte%TYPE;

    CURSOR carti\_cursor(id\_primit cititor\_DP.id\_cititor%TYPE) IS

        SELECT cod\_carte

        FROM imprumuta\_DP

        WHERE cod\_cititor = id\_primit;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Cititorii care au imprumutat aceleasi carti ca Cititor4 (pot sa aiba carti in plus care nu au fost citite de Cititor4)');

    OPEN cititori\_cursor;

    LOOP

        FETCH cititori\_cursor INTO id\_cititor, nume;

        EXIT WHEN cititori\_cursor%NOTFOUND;

        DBMS\_OUTPUT.PUT\_LINE(nume);

        OPEN carti\_cursor(id\_cititor);

        LOOP

            FETCH carti\_cursor INTO carte;

            EXIT WHEN carti\_cursor%NOTFOUND;

            DBMS\_OUTPUT.PUT\_LINE('  Cartea '||carte);

        END LOOP;

        CLOSE carti\_cursor;

    END LOOP;

    CLOSE cititori\_cursor;

END;

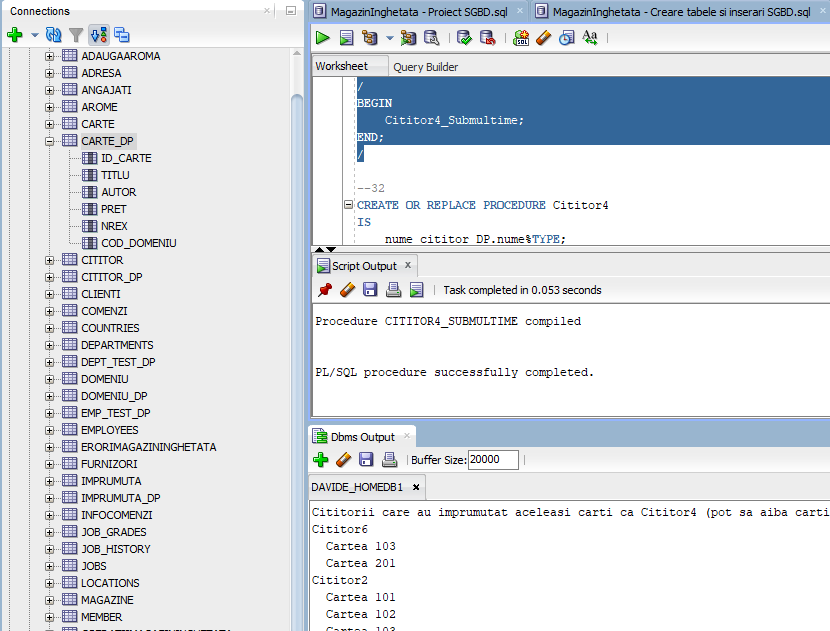
/

BEGIN

    Cititor4\_Submultime;

END;

/

****

Cititorii care au imprumutat aceleasi carti ca Cititor4 (pot sa aiba carti in plus care nu au fost citite de Cititor4)

Cititor6

Cartea 103

Cartea 201

Cititor2

Cartea 101

Cartea 102

Cartea 103

Cartea 103

Cartea 201

Cititor10

Cartea 101

Cartea 103

Cartea 201

Cartea 201

Cititor5

Cartea 103

Cartea 201

Cartea 202

Cititor1

Cartea 101

Cartea 103

Cartea 201

Cartea 202

Cititor4

Cartea 103

Cartea 103

Cartea 201

--32

CREATE OR REPLACE PROCEDURE Cititor4

IS

    nume cititor\_DP.nume%TYPE;

    id\_cititor cititor\_DP.id\_cititor%TYPE;

    CURSOR cititori\_cursor IS

        SELECT DISTINCT c.id\_cititor, c.nume

        FROM cititor\_DP c

        WHERE NOT EXISTS

        (   SELECT cod\_carte

            FROM imprumuta\_DP i

            WHERE i.cod\_cititor = c.id\_cititor

            MINUS

            SELECT cod\_carte

            FROM imprumuta\_DP i4

            WHERE i4.cod\_cititor = (SELECT id\_cititor FROM cititor WHERE nume = 'Cititor4') );

    carte imprumuta\_DP.cod\_carte%TYPE;

    CURSOR carti\_cursor(id\_primit cititor\_DP.id\_cititor%TYPE) IS

        SELECT cod\_carte

        FROM imprumuta\_DP

        WHERE cod\_cititor = id\_primit;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Cititorii care au imprumutat cel mult aceleasi carti ca Cititor4');

    OPEN cititori\_cursor;

    LOOP

        FETCH cititori\_cursor INTO id\_cititor, nume;

        EXIT WHEN cititori\_cursor%NOTFOUND;

        DBMS\_OUTPUT.PUT\_LINE(nume);

        OPEN carti\_cursor(id\_cititor);

        LOOP

            FETCH carti\_cursor INTO carte;

            EXIT WHEN carti\_cursor%NOTFOUND;

            DBMS\_OUTPUT.PUT\_LINE('  Cartea '||carte);

        END LOOP;

        CLOSE carti\_cursor;

    END LOOP;

    CLOSE cititori\_cursor;

END;

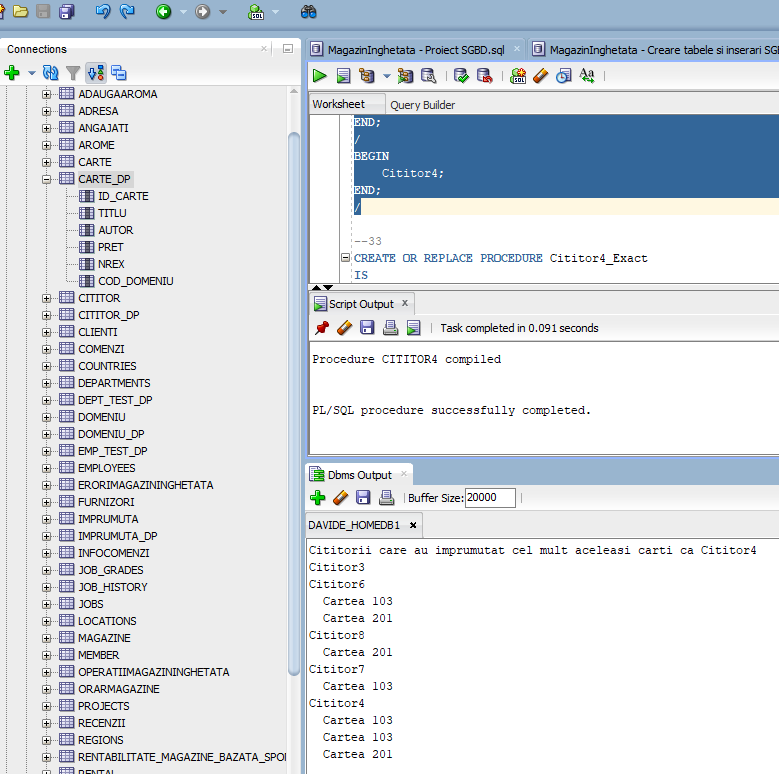
/

BEGIN

    Cititor4;

END;

/

****

--33

CREATE OR REPLACE PROCEDURE Cititor4\_Exact

IS

    nume cititor\_DP.nume%TYPE;

    id\_cititor cititor\_DP.id\_cititor%TYPE;

    CURSOR cititori\_cursor IS

        SELECT c.id\_cititor, c.nume

        FROM cititor c

        WHERE NOT EXISTS

        (   SELECT cod\_carte

            FROM imprumuta i4

            WHERE i4.cod\_cititor = (SELECT id\_cititor FROM cititor WHERE nume = 'Cititor4')

            MINUS

            SELECT cod\_carte

            FROM imprumuta i

            WHERE i.cod\_cititor = c.id\_cititor )

        AND NOT EXISTS

        (   SELECT cod\_carte

            FROM imprumuta i

            WHERE i.cod\_cititor = c.id\_cititor

            MINUS

            SELECT cod\_carte

            FROM imprumuta i4

            WHERE i4.cod\_cititor = (SELECT id\_cititor FROM cititor WHERE nume = 'Cititor4') );

    carte imprumuta\_DP.cod\_carte%TYPE;

    CURSOR carti\_cursor(id\_primit cititor\_DP.id\_cititor%TYPE) IS

        SELECT cod\_carte

        FROM imprumuta\_DP

        WHERE cod\_cititor = id\_primit;

BEGIN

    DBMS\_OUTPUT.PUT\_LINE('Cititorii care au imprumutat exact aceleasi carti ca Cititor4');

    OPEN cititori\_cursor;

    LOOP

        FETCH cititori\_cursor INTO id\_cititor, nume;

        EXIT WHEN cititori\_cursor%NOTFOUND;

        DBMS\_OUTPUT.PUT\_LINE(nume);

        OPEN carti\_cursor(id\_cititor);

        LOOP

            FETCH carti\_cursor INTO carte;

            EXIT WHEN carti\_cursor%NOTFOUND;

            DBMS\_OUTPUT.PUT\_LINE('  Cartea '||carte);

        END LOOP;

        CLOSE carti\_cursor;

    END LOOP;

    CLOSE cititori\_cursor;

END;

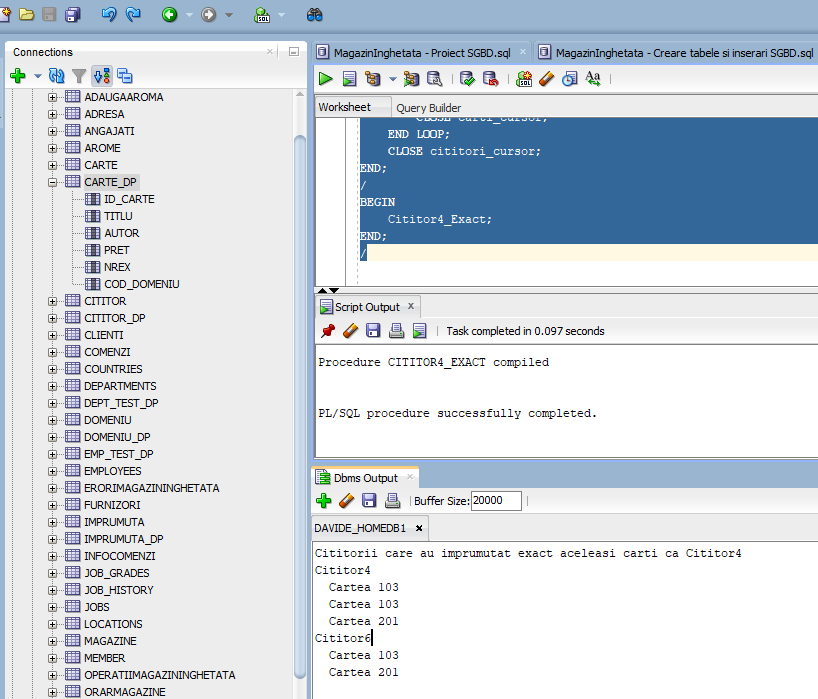
/

BEGIN

    Cititor4\_Exact;

END;

/

****