1. Descrierea modelului real, a utilității acestuia și a regulilor de funcționare :

Aceasta baza de date este una care gestioneaza activitatea unei companii ce furnizeaza servicii de cursuri de dans. Aceasta are cinci sedii cu cate doua sali de dans fiecare de dimensiuni egale si coregrafi(coregrafii pot preda la unul sau ambele sedii).

Exista bine inteles si receptionisti si angajati sanitari, dar acestia nu sunt relevanti acestei baze de date.

Clientii acestei companii pot sa achizitioneze abonamente(de un anumit tip), sedinte singulare, sau pot folosi vouchere promotionale pentru a participa la cursuri. Toate abonamentele si vocherele au valabilitate de o luna.

In afara de cursurile obisnuite ce au un program fix se pot lua sedinte private cu un anumit coregraf . Un curs reprezinta o sedinta indiferent de durata lui (un curs poate fi de o ora/o ora jumate).Clubul de dans organizeaza si diferite proiecte pe parcursul anului in afara sediilor de dans.

Informatiile stocate in aceasta baza de date sunt cele necesare gestionarii rezervarilor precum informatii client, coregraful angajat, curs/sedinta privata/proiect, sala , sediu, metoda de plata.

Modelul de date respectă anumite restricţii de funcţionare:

* Daca clientul are abonament/voucher se scad din sedintele din el in momentul in care se face o rezervare.Daca nu,sedintele sunt platite pe moment nefiind nevoie sa fie stocate.
* Proiectele si sedintele private pot fi platite doar pe moment, nu sunt incluse in abonament sau voucher, nu e nevoie sa fie stocata plata lor
* Cursurile se desfasoara pe trei nivele de dificultate: incepator, intermediar si avansat si nivelele pot fi predate de acelasi coregraf dar nu este obligatoriu, insa ele vor fi predate mereu la acelasi sediu, in aceeasi sala, vor avea acelasi pret,durata si aceeasi limita de varsta
* Clubul de dans organizeaza diferite proiecte pe parcursul anului in afara sediilor de dans.Vom lua in considerare doar coregraful sef al evenimentului cand retinem datele angajatului responsabil.
* Clientii care vin fara rezervare dar mai exista locuri se considera tot ca au facut rezervare si nu sunt tratati drept un caz special
* Un client poate rezerva un curs doar daca este in limita varstei cursului
* Pretul unui curs fara abonament este intre 40-55 de lei (avantajul abonamentului e ca platesti in functie de abonament intre 20-35 pt orice curs)
* In datele clientului sunt stocate si data incepere abonament, data incepere voucher(nule daca clientul nu detine nimic la momentul curent) care cand sunt la o luna fata de ziua curenta fac abonamentul/voucher-ul null, acestea se actualizeaza la achizitionarea unui alt abonament/primirea unui voucher(clientul primeste abonat\_id sau voucher\_id care pot fi si nule)
* Cursurile au un program fix iar salile sunt ocupate mereu in perioada cursurilor, sedintele private se fac in afara cursurilor cu coregrafii dupa un orar stabilita. Se scade din numarul de participanti existent in tabela “curs” care e initializata cu 25(numarul de participanti maxim universal tuturor cursurilor care e reinitializata duminica cu numarul maxim(deoarece duminica nu se tin cursuri)
* Proiectele sunt unice deci o data ce nr de locuri ajunge la zero,nu isi atinge nr minim de locuri cumparate cu 1 zi inainte de eveniment(15) sau ziua curenta este dupa ziua desfasurarii proietului, proiectul este sters din baza de date
* Sedintele private urmeaza un program saptamanal statusul lor de valabilitate este setat la zero in fiecare duminica, deoarece duminica nu exista sedinte private, includ rezervarea printr-un singur client a sedintei pentru 1-4 persoane, pretul nu e influentat de numarul de personae
* Voucher-ul e valabil doar pt u tip de curs si se extrag sedinte din el doar daca tipul cursului rezervarii este egal cu cel de pe voucher
* Un curs figureaza ca fiind predat de un singur coregraf fara sa fie contorizati eventualii invitati speciali, parteneri
* Tipurile de dans bagate in retea sunt cele care au si un curs asociat
* In timpul programului de lucru salile sunt mereu ocupate
* Cursurile se tin mereu, daca nu poate un anumit anagajat sa isi tina cursul in saptamana curenta se modifica provizoriu id-ul angajatului din curs urmand sa fie schimbat inapoi saptamana viiitoare
* Proiectele care nu strang participanti sunt sterse manual la nevoie cand se crede momentul potrivit(fiind unice)
* Orice activitate va incepe la ore fixe deci ora va fi memorata ca numar
* Sedintele private trebuie anulate cu o zi inainte
* Cursurile se pot anula doar in ziua desfasurarii acestuia
* Clientul poate sa anuleze o rezervare atata timp cat ora la care anularea este anuntata este cu doua ore inainte de inceperea cursului, altfel nu este luat in considerare
* Locul la un proiect e platit prealabil si nu poate fi anulat, un client poate da locul unei cunostinte, in baza de date ins ava figura tot cel initial
* Sedintele private sunt unice pe saptamana .
* Toate cursurile si sedintele private au durata de 2 ore deci nu se contorizeaza asta
* Fiecare rezervare e unica in system chiar daca ele sunt facute de acelasi client(rezervare\_id unic)

1. Diagrama entitate- relaţie

Rezerva(tabel asoc)

Predat de

are

are

are

TIPDANS

SEDIU

SALA\_DANS

COREGRAF

CURS

VOUCHER

ABONAMENT

PROIECT

SEDINTA\_PRIVATA

CLIENT

se tine in

contine

Predata de

Predat de

1

rezerva

rezerva

M(1)

M(0)

M(M)

M(M)

M(M)

M(1)

1

M(1)

1

1

M(0)

1

M

1

M(0)

1(0)

1(0)

M(0)

1. Diagrama conceptuală

este

este

este

face

Predat de

are

are

are

TIPDANS

SEDIU

SALA\_DANS

COREGRAF

CURS

VOUCHER

ABONAMENT

PROIECT

SEDINTA\_PRIVATA

CLIENT

REZERVARE

se tine in

contine

are

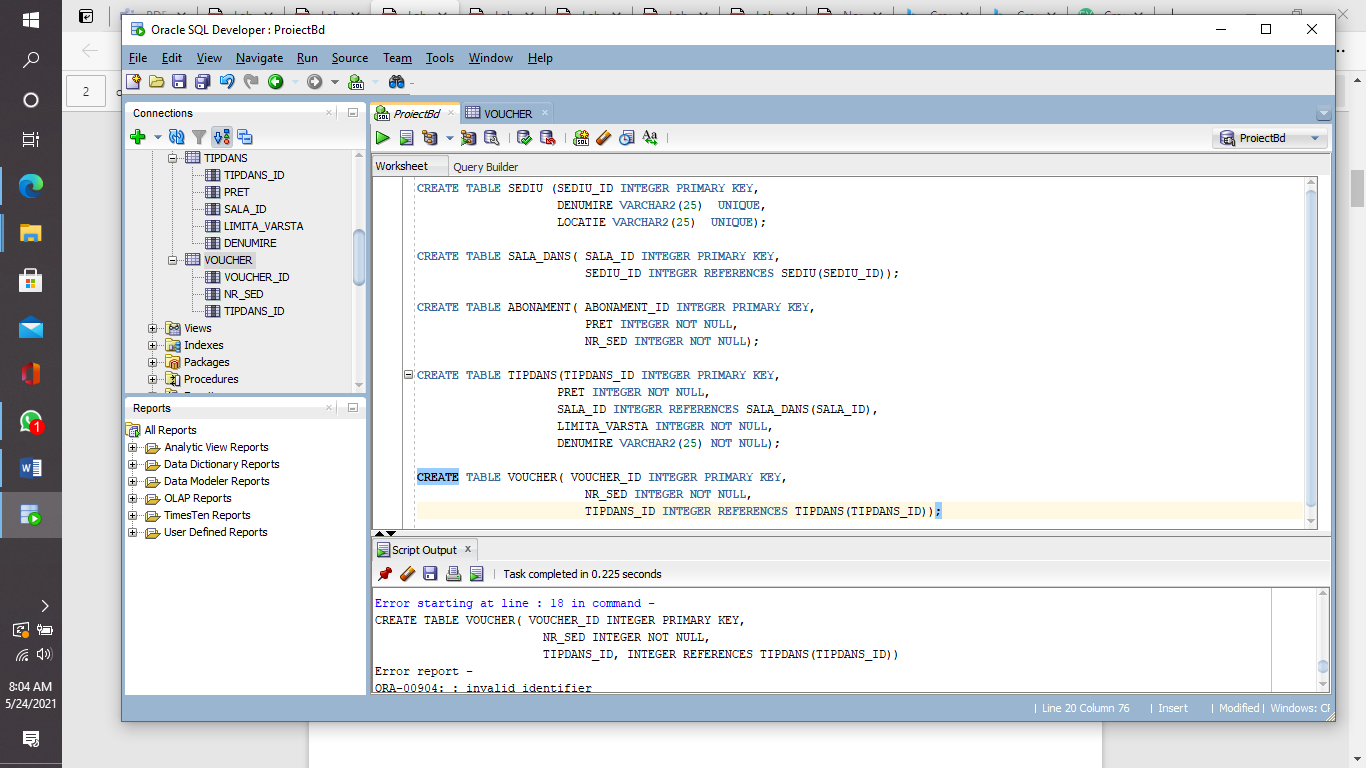
Predata de

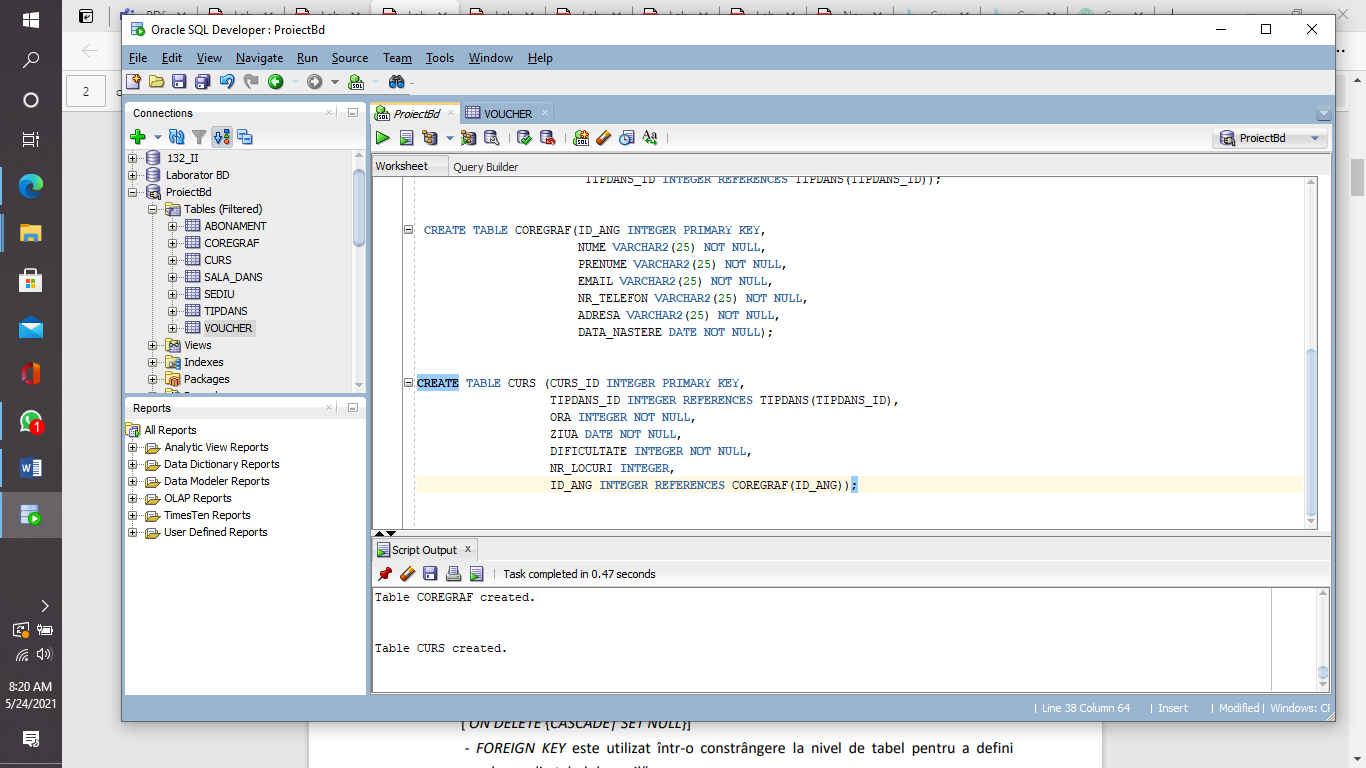
Predat de

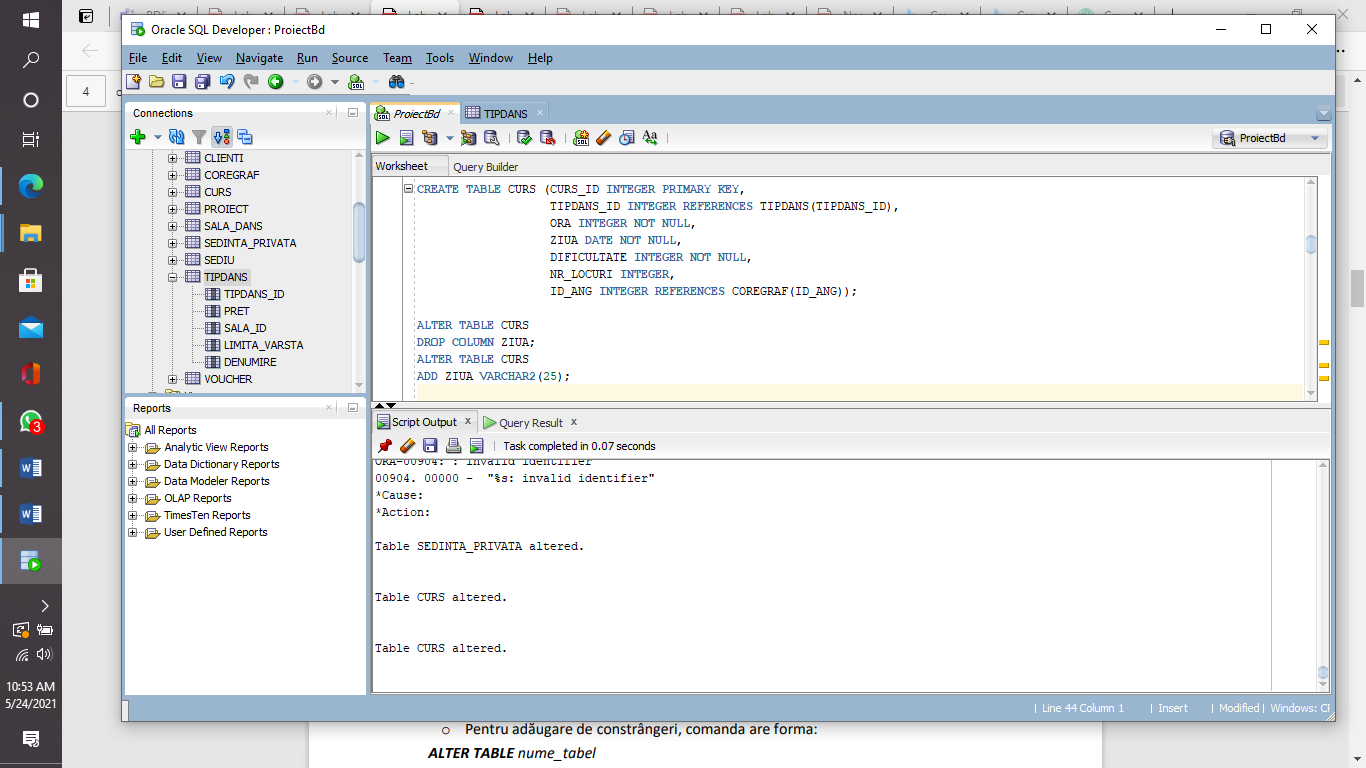
ANULARE

4.5. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum5înregistrări înfiecare tabelneasociativ; minimum 10 înregistrări întabelele asociative).

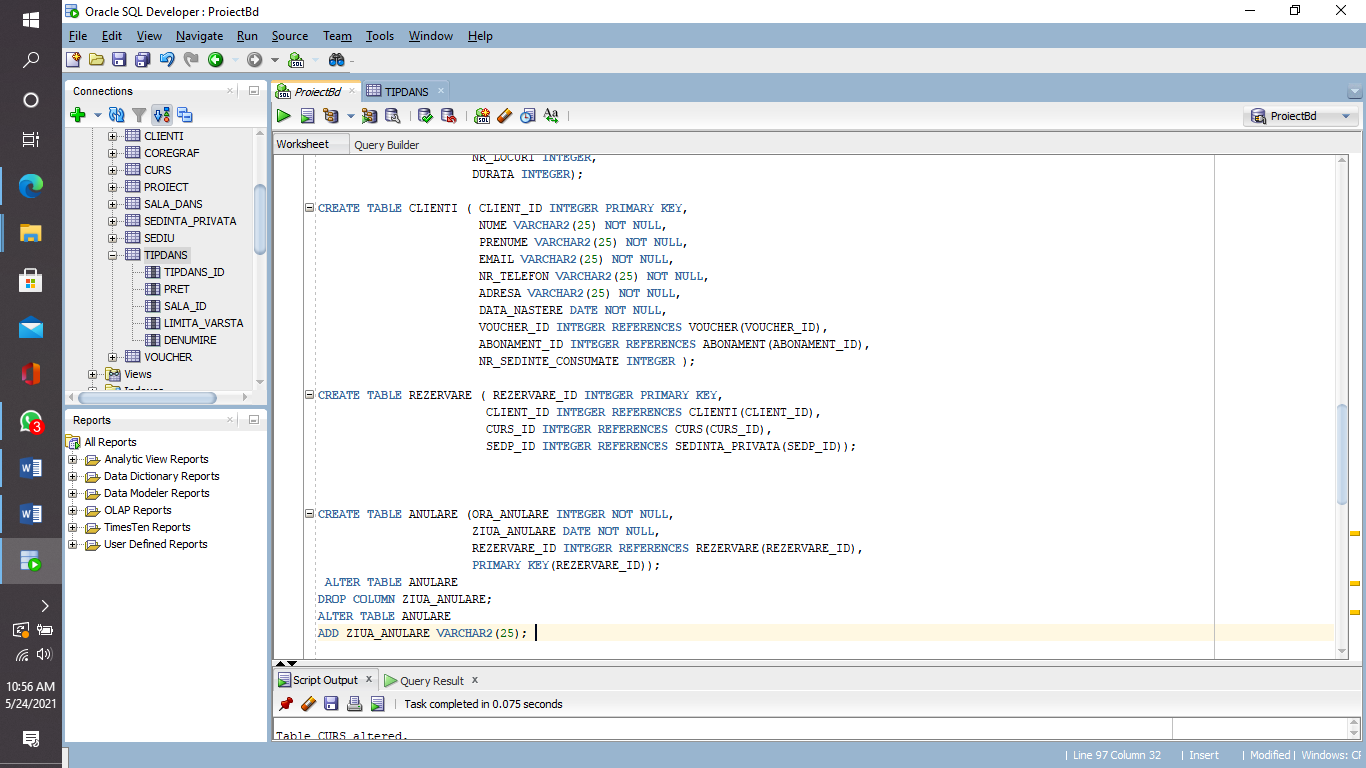
* In baza de date am denumit entitatea Client, Clienti pentru ca mi-o facea comanda altfel si cod de la voucher in voucher\_id deoarece e mai sugestiv

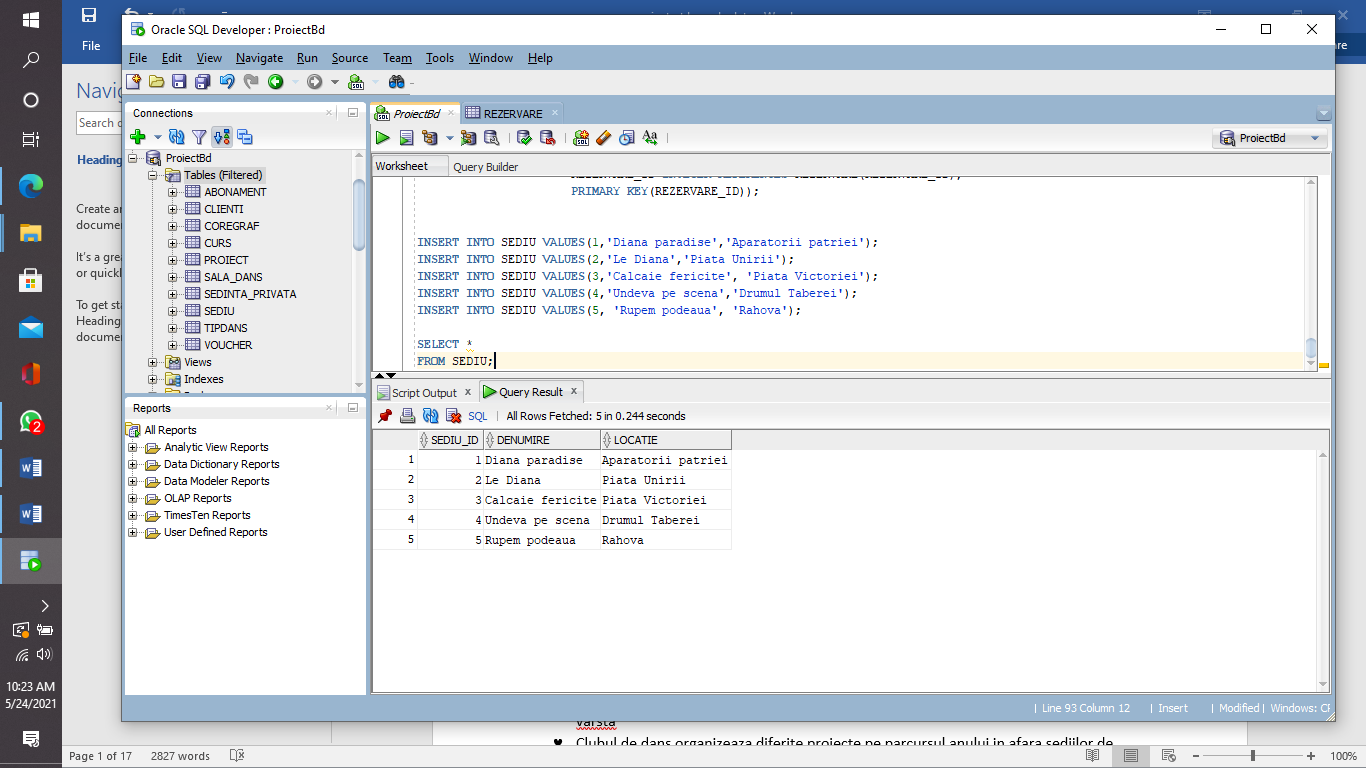


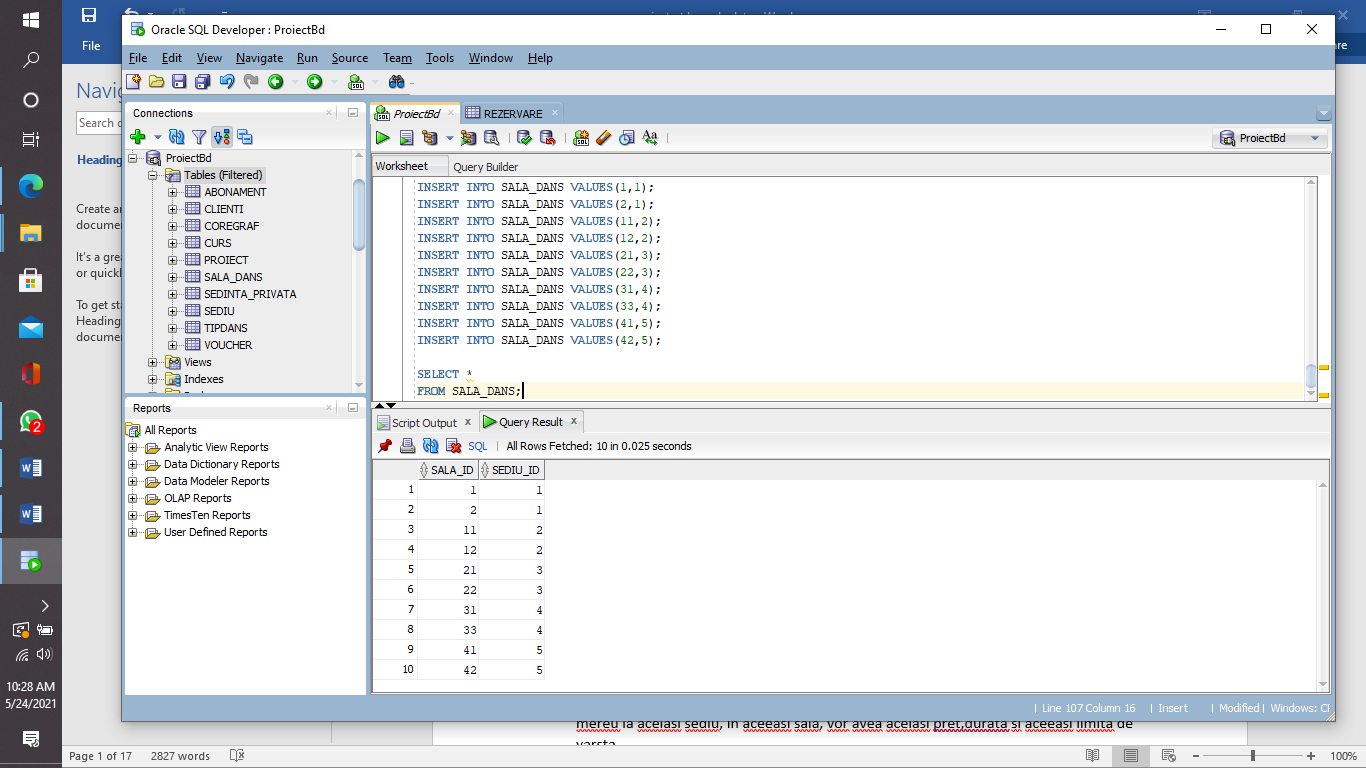


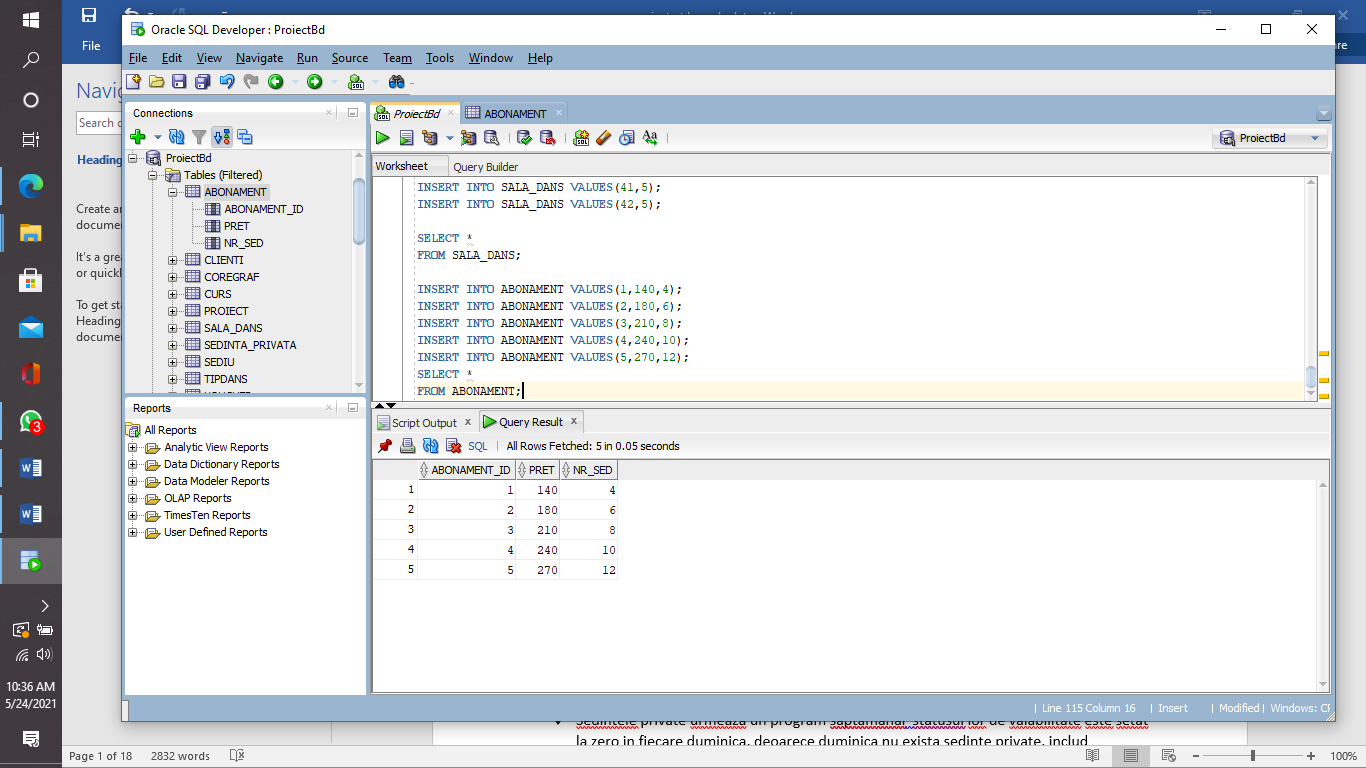


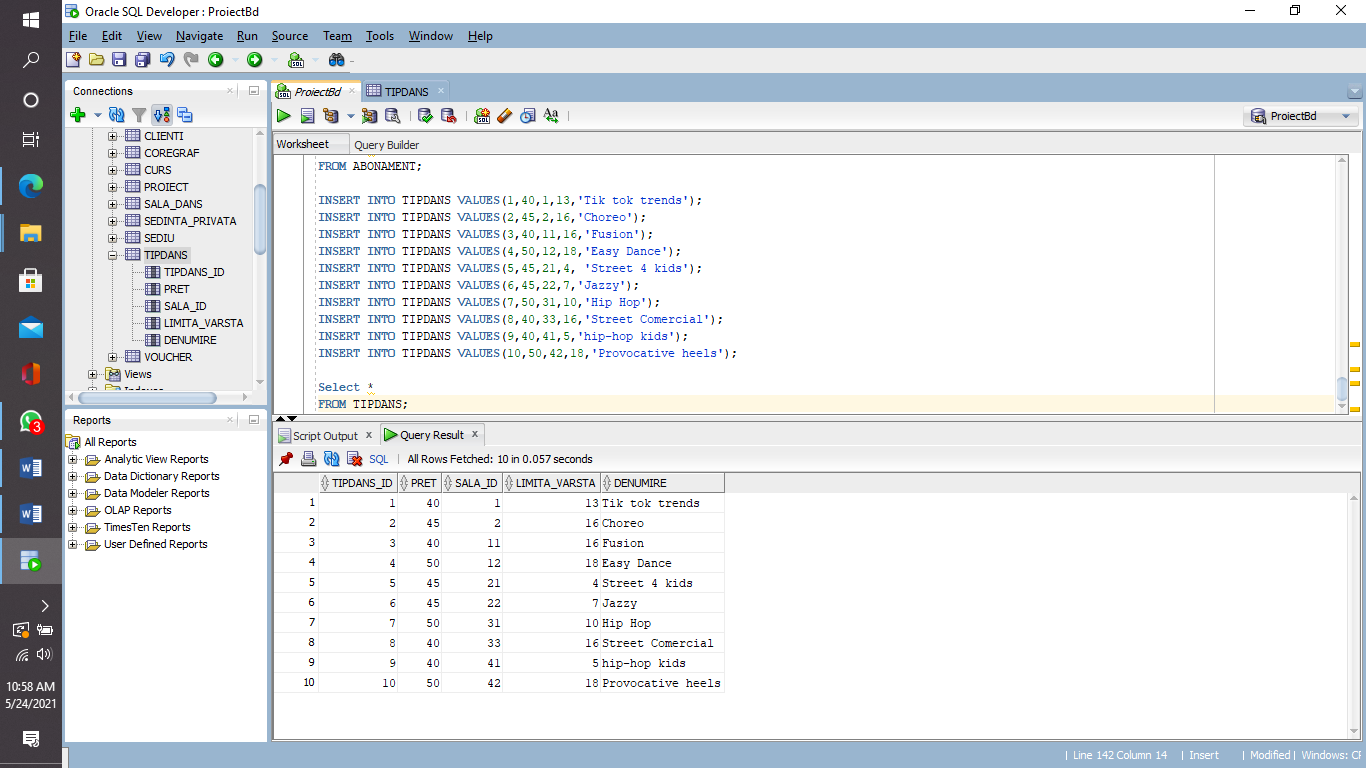


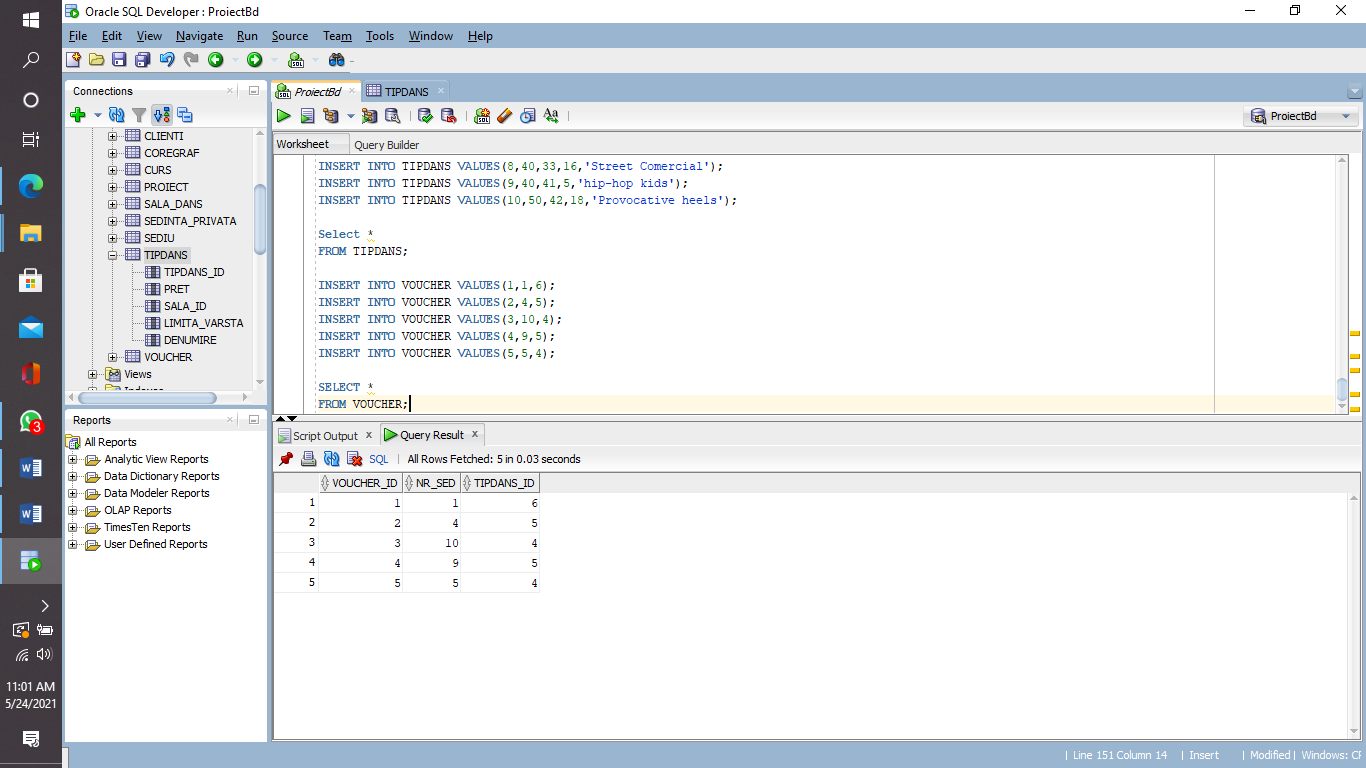


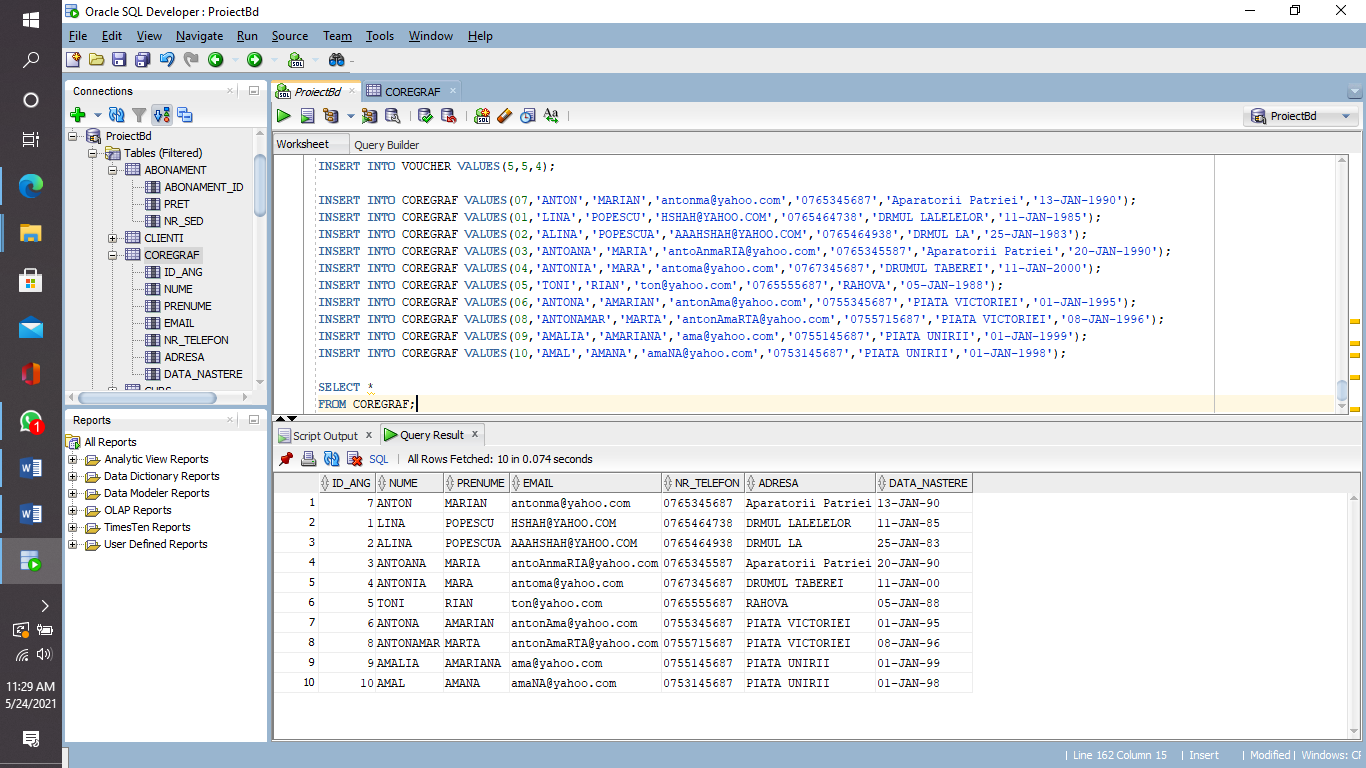
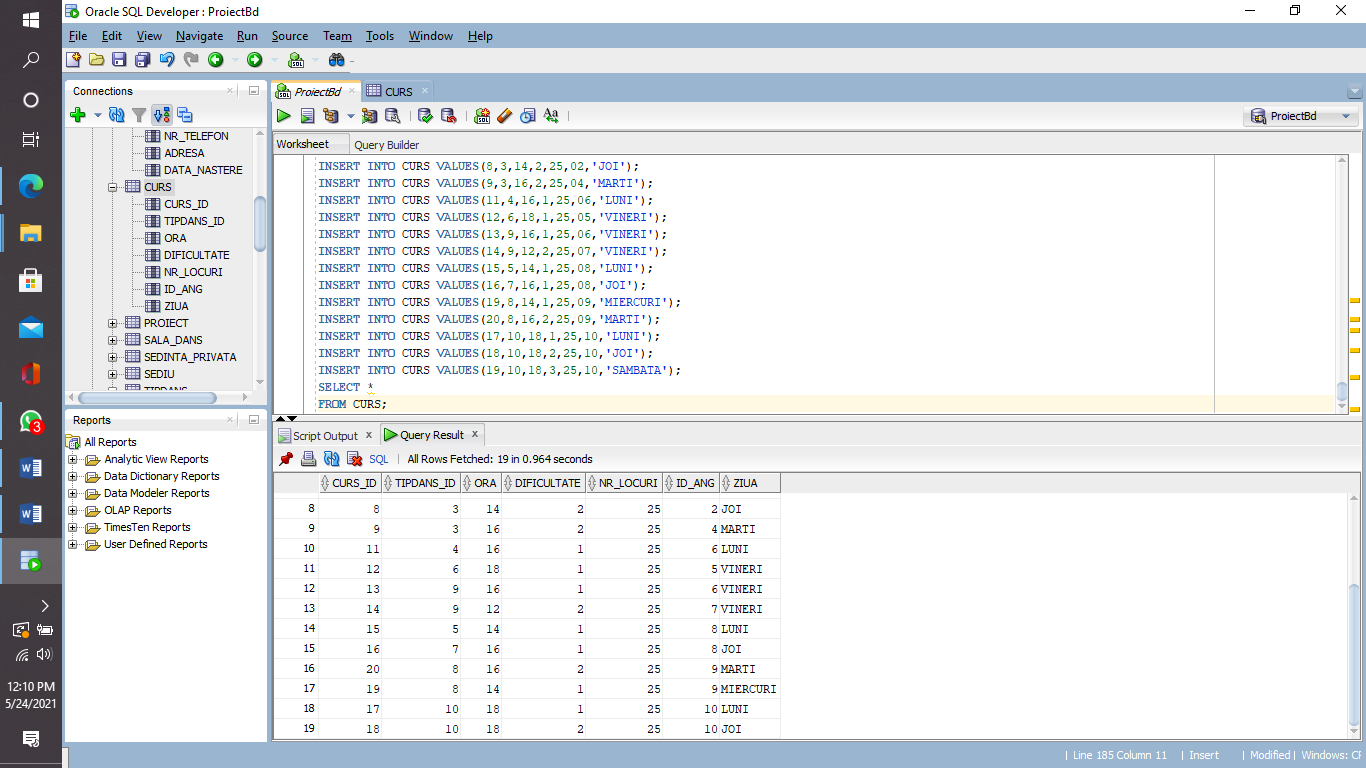


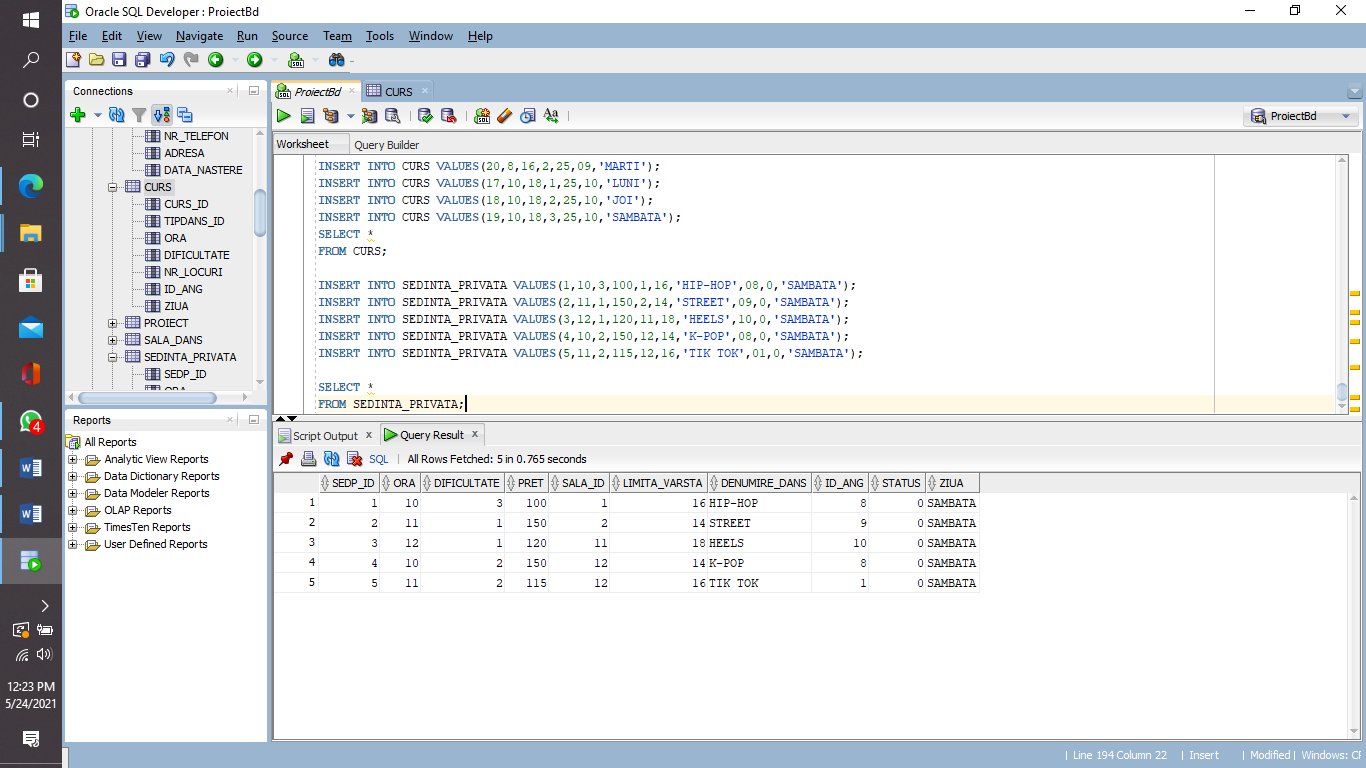


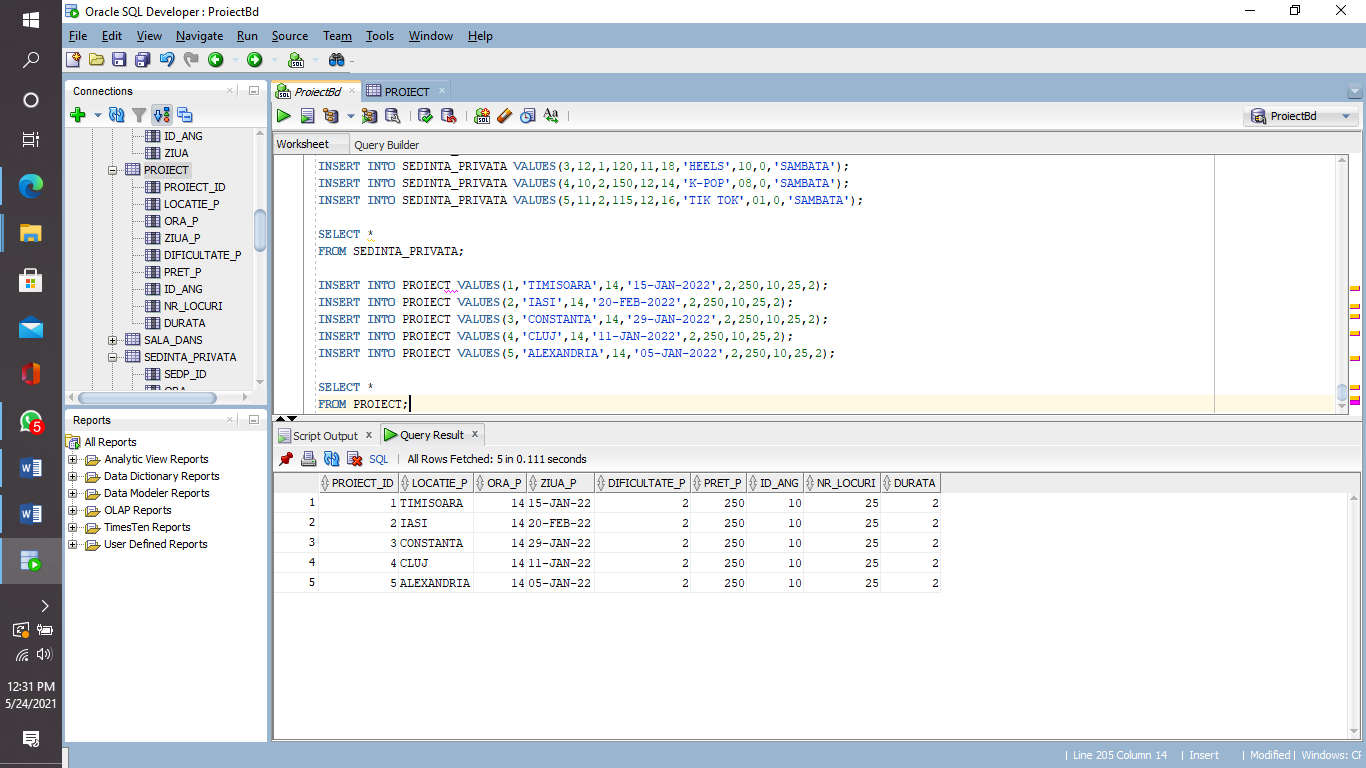


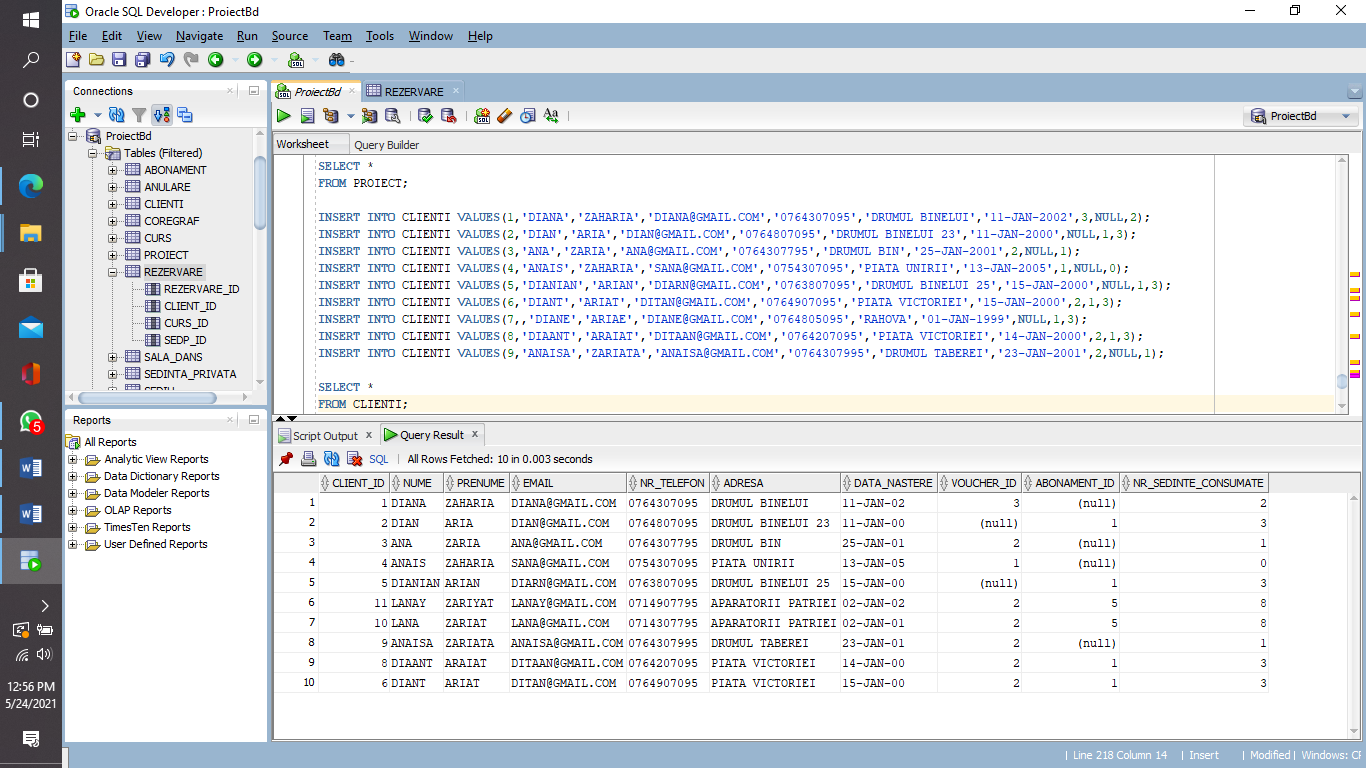
















6. Scrie care sunt cursurile predate, cate cursuri preda si ce tipuri de dans preda un anumit angajat cu id-ul dat (nr de la 1 la 10 cu exceptia lui 3).

set serveroutput on;

create or replace type vector is varray(11) of number;

CREATE OR REPLACE FUNCTION zdc1 (v\_id number)

RETURN NUMBER IS

t vector;

TYPE tablou2 IS TABLE OF NUMBER(30);

tb tablou2;

BEGIN

SELECT CURS\_ID

BULK COLLECT INTO t

FROM CURS

WHERE ID\_ANG=v\_id;

DBMS\_OUTPUT.PUT('COREGRAFUL ARE ' || t.COUNT ||' CURSURI: ');

FOR i IN t.FIRST..t.LAST LOOP

DBMS\_OUTPUT.PUT(t(i)|| ' ');

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

SELECT tipdans\_id BULK COLLECT INTO tb from curs WHERE ID\_ANG=v\_id;

FOR idx IN 1..tb.COUNT

LOOP

DBMS\_OUTPUT.PUT\_LINE('TIP DE DANS '||idx ||': ' || tb(idx));

END LOOP;

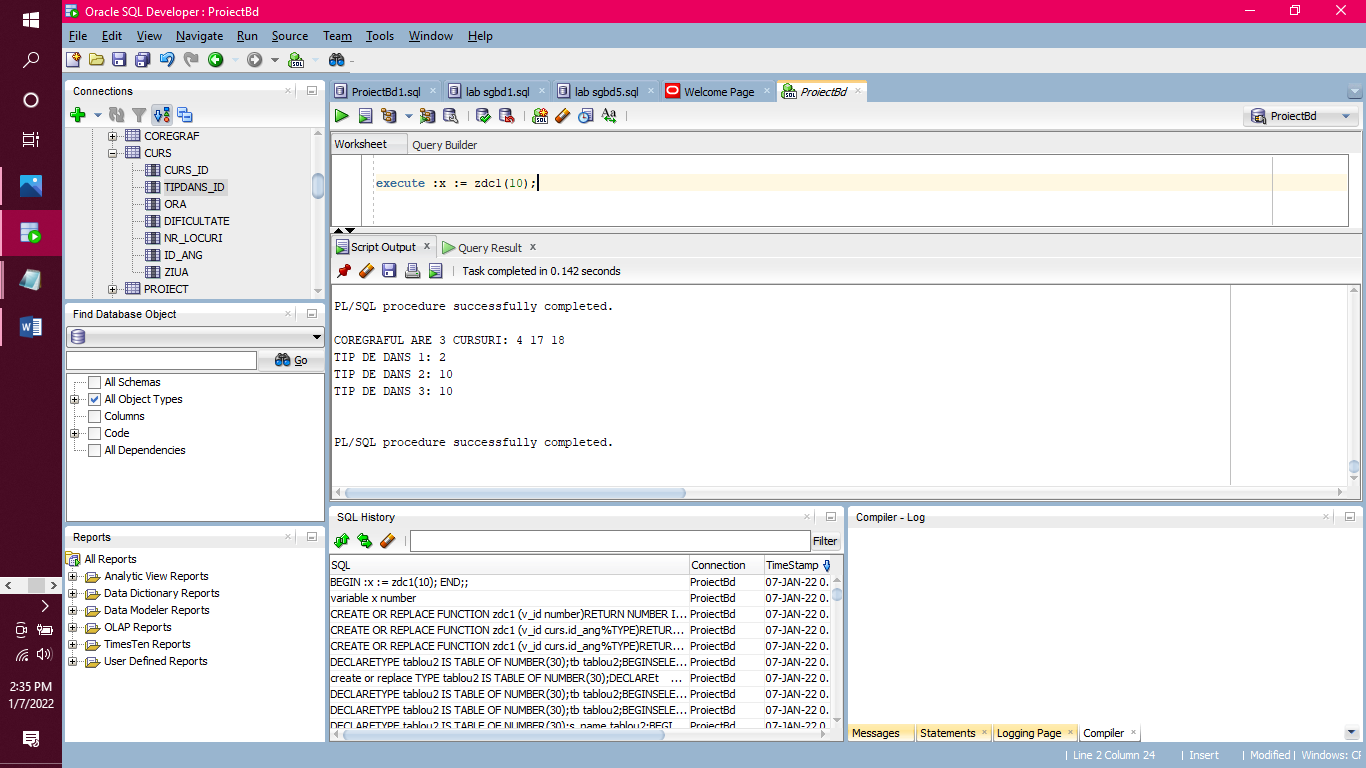
return 0;

END zdc1;

--testare

variable x number

execute :x := zdc1(10);



7. Scrie denumirea si limita de varsta in ordinea descrescatoare a varstelor pentru tipurile de dans cu id-ul mai mic decat cel dat.

CREATE OR REPLACE FUNCTION zdc2 (v\_id number)

RETURN NUMBER IS

CURSOR cZ

IS

SELECT

denumire, limita\_varsta

FROM

tipdans

WHERE tipdans\_id<=v\_id

ORDER BY

limita\_varsta DESC;

BEGIN

FOR rZ IN cZ

LOOP

dbms\_output.put( rZ.denumire || ': ' || rZ.limita\_varsta );

dbms\_output.new\_line;

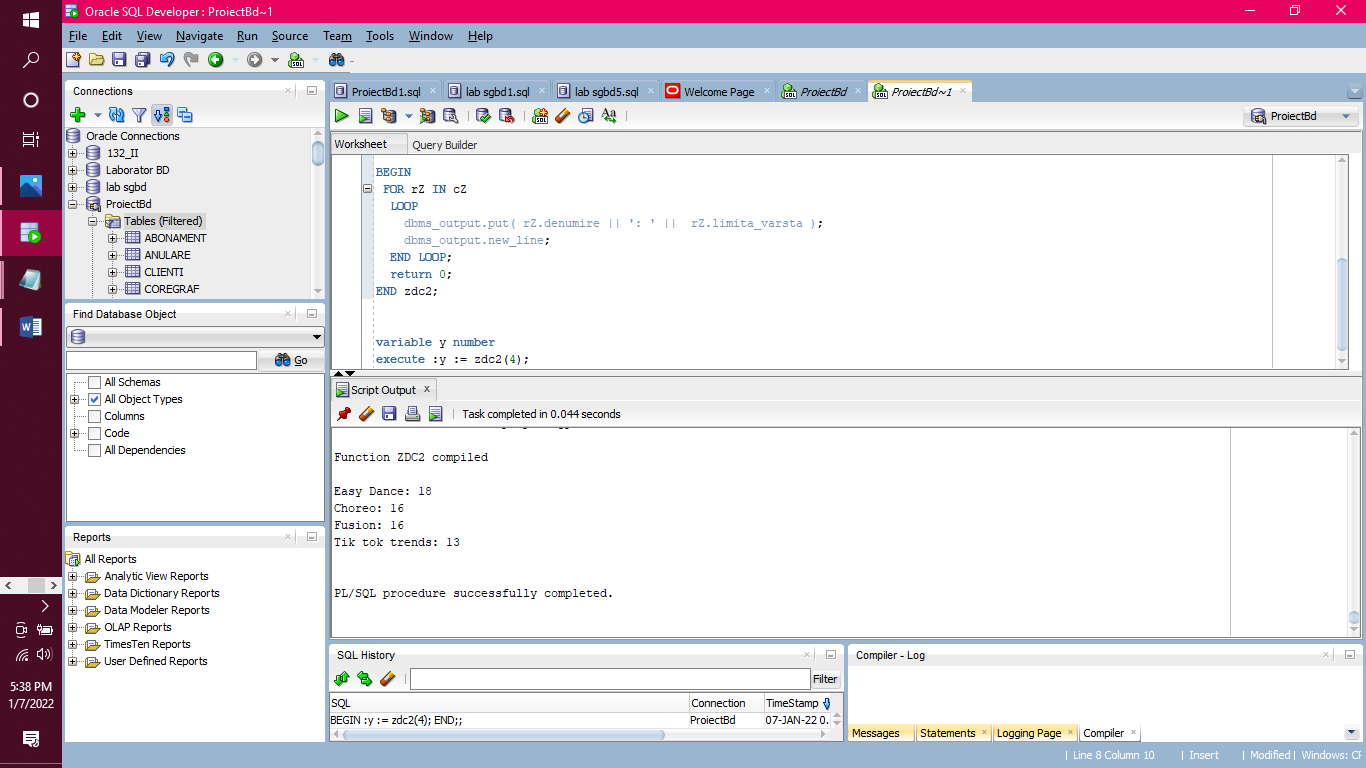
END LOOP;

return 0;

END zdc2;

variable y number

execute :y := zdc2(4);



8. Scrieti cate si ce cursuri are angajatul cu un id dat(1-10) intr-o sala cu id dat(0,1,2,11,22,33)

Daca nu are niciun curs in sala nu afisa nimic.

CREATE OR REPLACE FUNCTION zdc3 (v\_id number, s\_id number)

RETURN NUMBER IS

t vector;

id\_c number;

BEGIN

SELECT curs.curs\_id

INTO id\_c

FROM (curs inner join tipdans on curs.tipdans\_id=tipdans.tipdans\_id)inner join sala\_dans on tipdans.sala\_id=sala\_dans.sala\_id

WHERE curs.id\_ang=v\_id and sala\_dans.sala\_id=s\_id;

DBMS\_OUTPUT.PUT('COREGRAFUL ARE UN CURS in sala ' || s\_id || ': ');

DBMS\_OUTPUT.PUT(id\_c|| ' ');

DBMS\_OUTPUT.NEW\_LINE;

return 0;

EXCEPTION

when NO\_DATA\_FOUND then

return -1;

when TOO\_MANY\_ROWS then

SELECT curs.curs\_id

BULK COLLECT INTO t

FROM (curs inner join tipdans on curs.tipdans\_id=tipdans.tipdans\_id)inner join sala\_dans on tipdans.sala\_id=sala\_dans.sala\_id

WHERE curs.id\_ang=v\_id and sala\_dans.sala\_id=s\_id;

DBMS\_OUTPUT.PUT('COREGRAFUL ARE ' || t.COUNT ||' CURSURI IN SALA '|| s\_id || ': ');

FOR i IN t.FIRST..t.LAST LOOP

DBMS\_OUTPUT.PUT(t(i)|| ' ');

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

return -2;

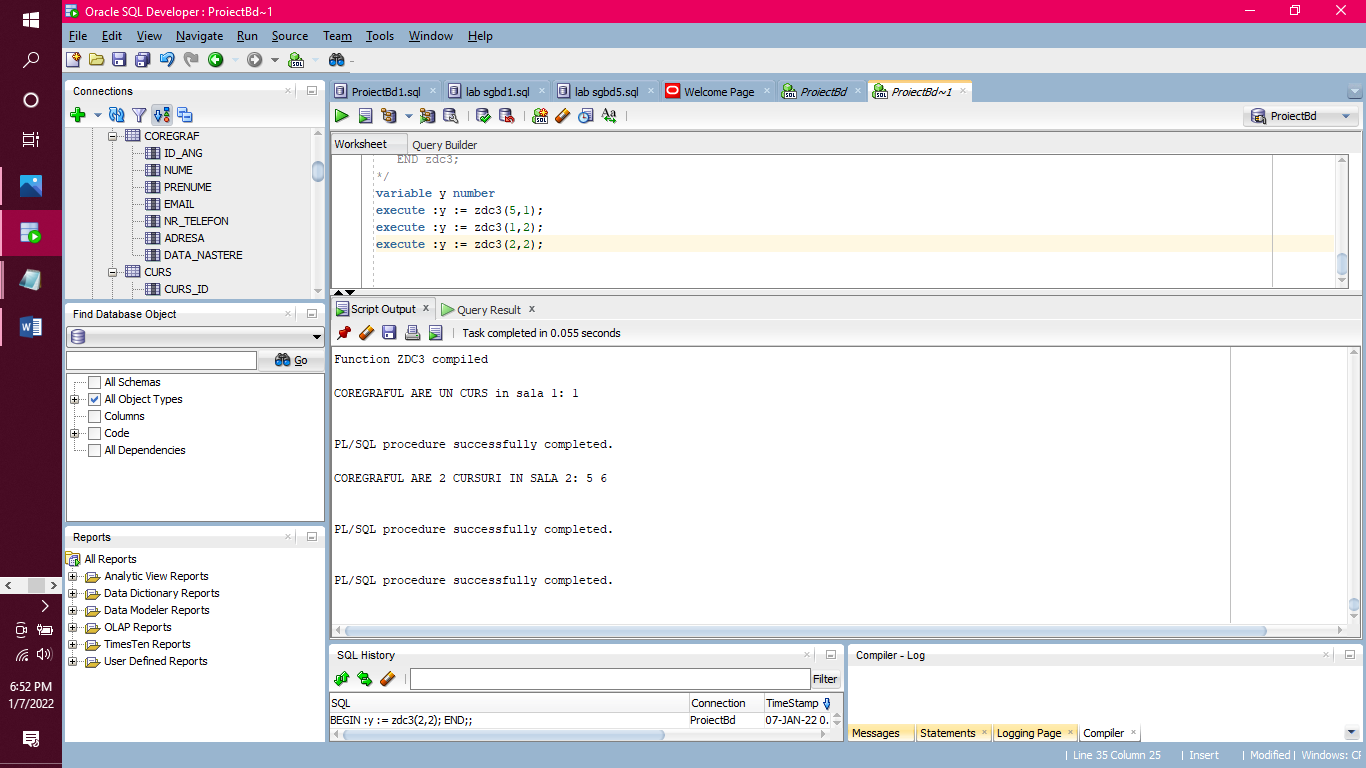
END zdc3;

variable y number

execute :y := zdc3(5,1);

execute :y := zdc3(1,2);

execute :y := zdc3(2,2);



9. Scrie daca se sustine doar un curs de AMAL AMANA intr-un sediu dat altfel scrie eroarea potrivita.

CREATE OR REPLACE PROCEDURE zdc8( nr number)

IS

id\_c number;

BEGIN

SELECT curs.curs\_id

INTO id\_c

FROM (((curs inner join tipdans on curs.tipdans\_id=tipdans.tipdans\_id) inner join sala\_dans on tipdans.sala\_id=sala\_dans.sala\_id) inner join sediu on sala\_dans.sediu\_id=sediu.sediu\_id) inner join coregraf on coregraf.id\_ang=curs.id\_ang

WHERE coregraf.nume='AMAL' AND COREGRAF.PRENUME='AMANA' and sediu.sediu\_id<nr;

DBMS\_OUTPUT.PUT\_LINE('cursul este '|| id\_c);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RAISE\_APPLICATION\_ERROR(-20000,'Nu exista ');

WHEN TOO\_MANY\_ROWS THEN

RAISE\_APPLICATION\_ERROR(-20001,'Exista mai multe');

WHEN OTHERS THEN

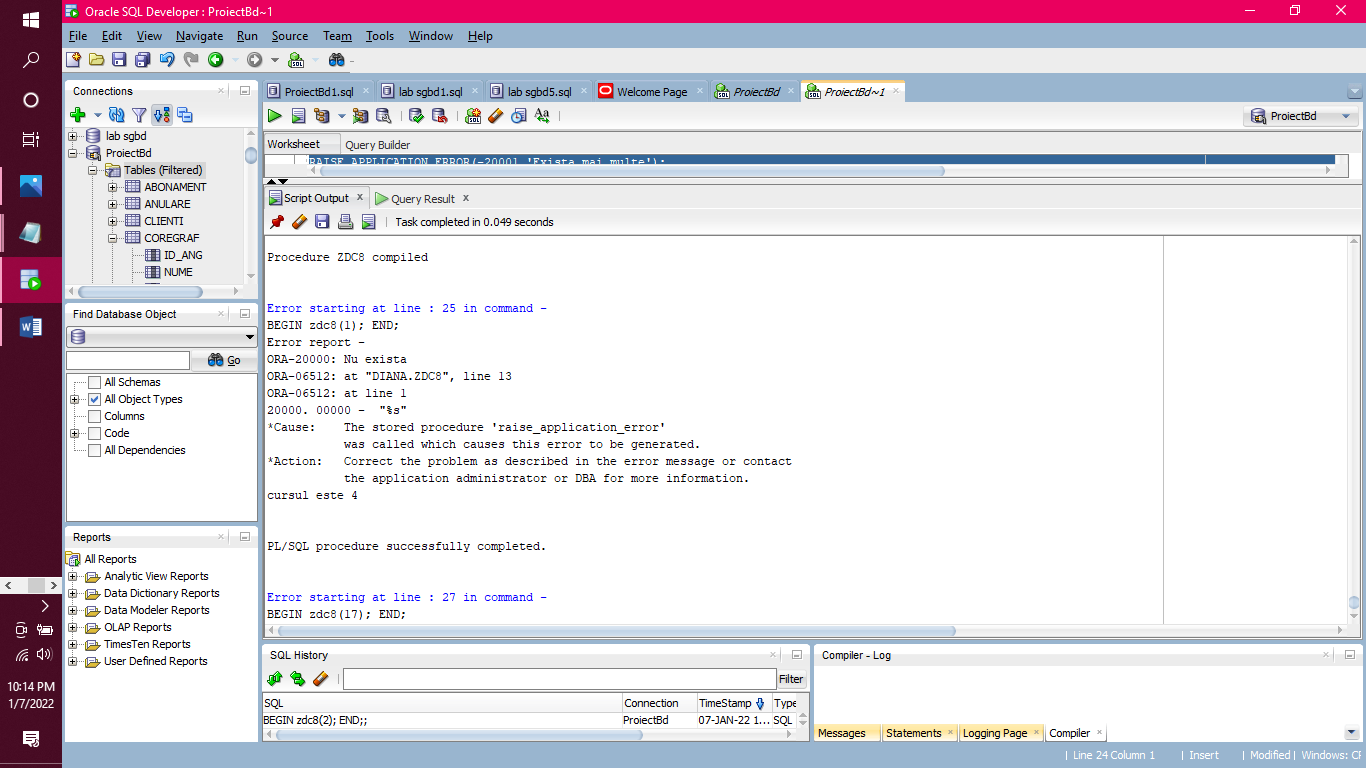
RAISE\_APPLICATION\_ERROR(-20002,'Alta eroare!');

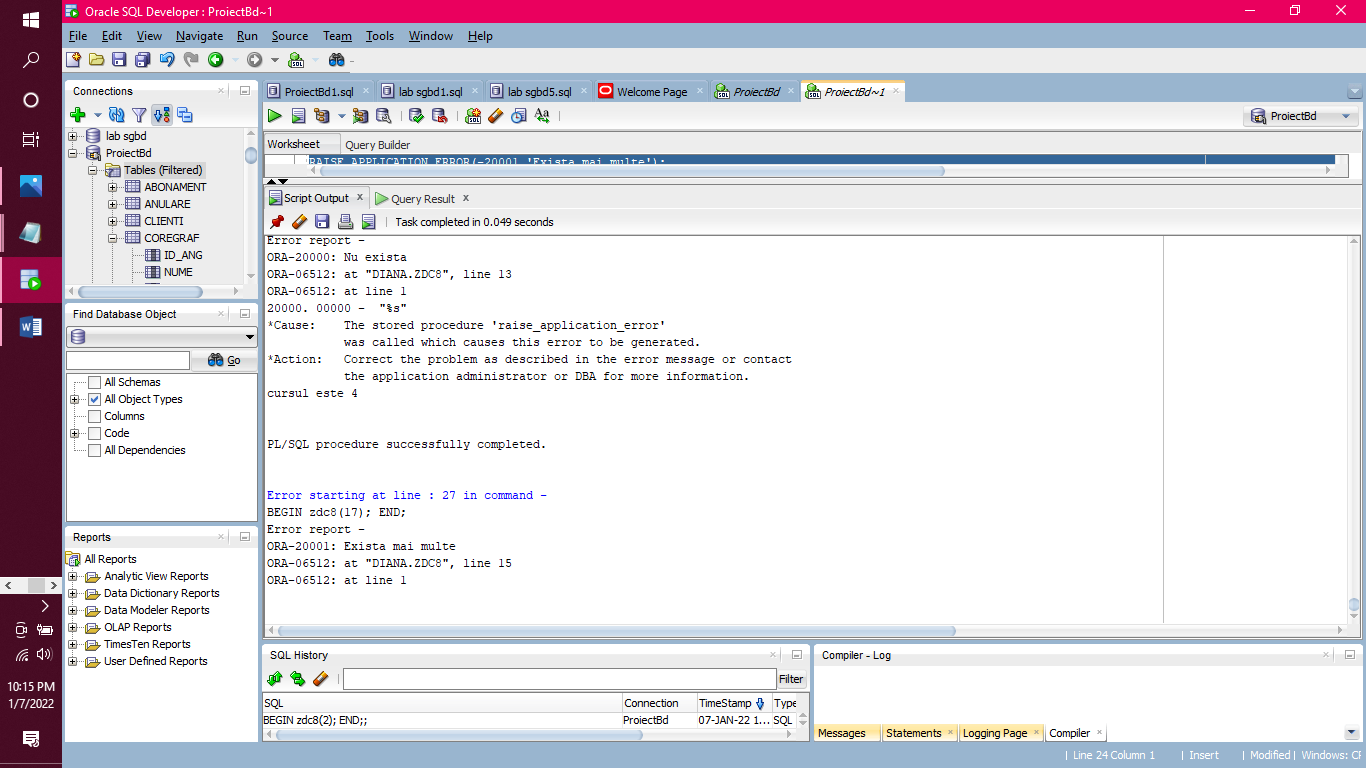
END zdc8;

EXECUTE zdc8(1);

EXECUTE zdc8(2);

EXECUTE zdc8(17);





10. -Definiți un declanșator care să permită lucrul asupra tabelului anulare (INSERT, UPDATE, DELETE) decât în intervalul de ore 8:00 -20:00, de luni până sâmbătă(declanșator la nivel de instrucțiune).

CREATE OR REPLACE TRIGGER trig1

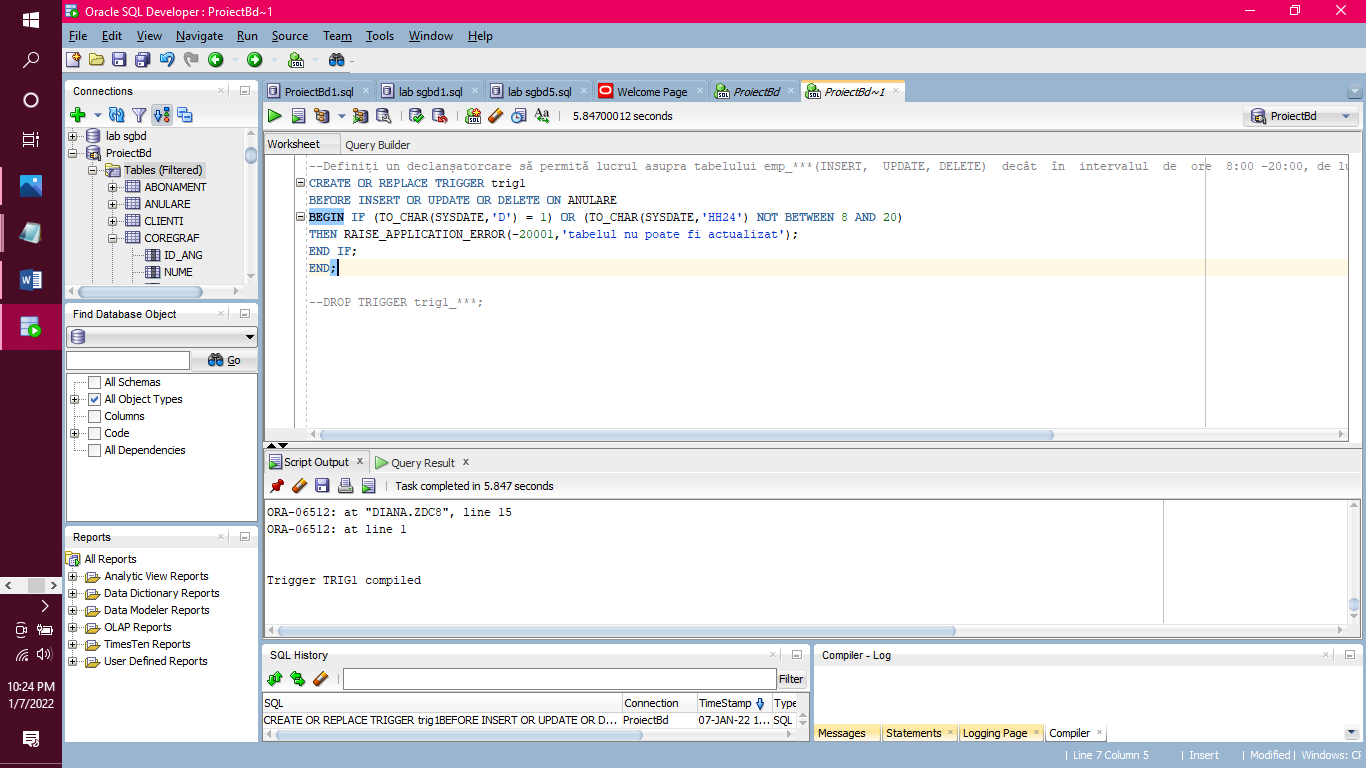
BEFORE INSERT OR UPDATE OR DELETE ON ANULARE

BEGIN IF (TO\_CHAR(SYSDATE,'D') = 1) OR (TO\_CHAR(SYSDATE,'HH24') NOT BETWEEN 8 AND 20)

THEN RAISE\_APPLICATION\_ERROR(-20001,'tabelul nu poate fi actualizat');

END IF;

END;

--DROP TRIGGER trig1\_\*\*\*;

11. Definițiun declanșatorprin care să nu se permită micșorarea pretului dansurilor din tabelul tipdans (declanșator la nivel de linie).

CREATE OR REPLACE TRIGGER trig21

BEFORE UPDATE OF pret ON TIPDANS

FOR EACH ROW

BEGIN

IF (:NEW.pret < :OLD.pret) THEN

RAISE\_APPLICATION\_ERROR(-20002,'pretul nu poate fi micsorat');

END IF;

END;

UPDATE tipdans

SET pret=pret-10;

--DROP TRIGGER trig21;

12. a.Creați un tabel cu următoarele coloane:

-user\_id(SYS.LOGIN\_USER);-nume\_bd(SYS.DATABASE\_NAME);-erori(‘eroare’);-data.

b.Definițiun declanșator sistem(la nivel de bază de date) care să introducă date în acest tabel referitoare la erorile apărute.

CREATE TABLE tabel\_erori (

user\_id VARCHAR2(200),

nume\_bd VARCHAR2(200),

erori VARCHAR2(200),

data DATE

);

CREATE OR REPLACE TRIGGER trgg

AFTER SERVERERROR

ON DATABASE

BEGIN

INSERT INTO tabel\_erori

VALUES (

SYS.LOGIN\_USER,

SYS.DATABASE\_NAME,

'eroare',

SYSDATE

);

END;