

Proiect - Sisteme de Baze de Date

- Gestiunea unui lanț de restaurante -

Popescu Paullo Robertto Karloss

Grupa 406

2025 Ianuarie

Cuprins:

1. Prezentarea bazei de date.
 - 1.1 Tehnologii folosite pentru realizarea proiectului.
 - 1.2 Descrierea temei alese.
 - 1.3 Prezentarea constrângerilor impuse asupra modelului.
 - 1.4 Descrierea entităților.
 - 1.5 Descrierea relațiilor.
 - 1.6 Descrierea atributelor.
2. Diagrama Entitate-Relație (ER).
3. Diagrama Conceptuală.
4. Schemele relaționale corespunzătoare diagramei conceptuale.
5. Normalizarea până în forma normal 3 (FN1-FN3).
6. Implementarea bazei de date în Oracle.
 - 6.1 Crearea tabelelor și a constrângerilor.
 - 6.2 Inserarea datelor coerente în tabele (minimum 5 înregistrări în fiecare tabel neasociativ, minimum 10 înregistrări în tabelele associative) + crearea unei secvențe.
 - 6.3 Diagrama generată în SQL Developer după crearea tabelelor și inserarea datelor.
7. Crearea a 15 interogări complexe în SQL.
8. Crearea tabelului MESAJE pentru excepții.
9. PL / SQL.
 - 9.1 Subprogram stocat independent (inclusiv apelare) care utilizează toate cele 3 tipuri de colecții învățate.
 - 9.2 Subprogram stocat independent (inclusiv apelare) care utilizează toate cele 2 tipuri de cursoare învățate, unul dintre acestea fiind cursor parametrizat, dependent de celălalt cursor.
 - 9.3 Subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite și tratarea excepțiilor care pot apărea.
 - 9.4 Crearea unui Trigger LMD la nivel de comandă.
 - 9.5 Crearea unui Trigger LMD la nivel de linie (TRIGGER COMPOUND).
 - 9.6 Crearea unui Trigger LDD.
 - 9.7 Crearea unui pachet pentru obiectele definite.
10. Inserarea excepțiilor în tabelul mesaje.

1. Prezentarea bazei de date

1.1 Tehnologii folosite pentru realizarea proiectului.

Pentru proiectul din cadrul cursului de Baze de Date am folosit versiunea **21c** a *Oracle Database*.

Ce aplicații am folosit?

- Oracle SQL Developer

1.2 Descrierea temei alese. Pentru ce ar fi folosită

În cadrul acestui proiect, am ales ca temă **Gestiunea unui lanț de restaurante**. Scopul ei este de a face mai ușoară ținerea în evidență a mai multor restaurante (comenzi, angajați, facturi etc).

1.3 Prezentarea constrângerilor impuse asupra modelului.

Un angajat trebuie să lucreze la un singur restaurant.

Un restaurant poate avea mai mulți angajați.

Angajații pot fi chelneri, casieri, bucătari sau manageri.

Un angajat poate avea subordonați, iar subordonații pot avea la rândul lor alți subordonați.

Într-o locație se poate găsi un singur restaurant.

Produsele sunt făcute cu ingrediente care sunt fabricate de un producător.

Într-o factură se poate găsi o singură comandă.

O comandă conține unul sau mai multe produse.

Un produs poate să aparțină mai multor restaurante.

Produsele pot fi preparate de un bucătar, dar există și produse care nu trebuie preparate (de exemplu vin, şampanie etc).

La o factură se pot preciza detalii, cum ar fi: dacă a fost achitată cash sau cu ajutorul unui card de credit.

1.4 Descrierea entităților.

Cum am precizat mai sus, ideea proiectului este Gestionarea unui lanț de restaurante, de la care clienții pot comanda diferite produse. Acesta cuprinde următoarele entități:

- Entitatea **Restaurant**, care va conține numele restaurantului. Cheia primară fiind *id-ul restaurantului*.
- Entitatea **Locație**, care va conține numele țării unde se află restaurantul, orașul, respectiv codul poștal și strada. Cheia primară fiind *id-ul locației*.
- Entitatea **Produs**, care va conține numele fiecărui produs, cantitatea și o scurtă descriere a sa. Cheia primară fiind *id-ul produsului*.
- Entitatea **Ingredient**, care va conține numele ingredientului. Cheia primară fiind *id-ul ingredientului*.
- Entitatea **Producător**, care va conține numele și numărul de telefon al unui producător de ingrediente. Cheia primară fiind *id-ul producătorului*.
- Entitatea **Comandă**, care va conține prețul și data pentru fiecare comandă plasată. Cheia primară fiind *id-ul comenzi*.
- Entitatea **Angajat**, care va conține id-ul șefului (dacă are un șef), numele, prenumele și data angajării. Cheia primară fiind *id-ul angajatului*.
- Subentitatea **Bucătar**, care va conține numărul de stele al fiecărui bucătar. Cheia primară fiind *id-ul angajatului*.
- Subentitatea **Casier**, care va conține numărul de ani de studii al fiecărui casier. Cheie primară fiind *id-ul angajatului*.
- Subentitatea **Chelner**, care va conține numărul de ani de experiență al fiecărui chelner. Cheie primară fiind *id-ul angajatului*.
- Subentitatea **Manager**, care nu are atrbute, are doar cheia primară *id-ul angajatului*.
- Entitatea **Client**, care va conține numele, prenumele și numărul personal de telefon, întrucât acestea sunt necesare atunci când se plasează o comandă. Cheia primară fiind *id-ul clientului*.
- Entitatea **Factură**, care va conține valoarea pe care clientul este nevoit să o plătească pentru comanda sa, dar și câteva detalii cum ar fi dacă a fost achitată cash sau cu ajutorul unui card de credit. Cheia primară fiind *id-ul facturii*.

1.5 Descrierea relațiilor.

Un anumit produs poate apartine mai multor restaurante.

Într-o locație poate fi găsit un singur restaurant.

Un produs se poate regăsi în mai multe comenzi.

Într-o comandă se pot găsi mai multe produse, dar cel puțin unul.

Un produs poate fi preparat de mai mulți bucătari.

Un ingredient trebuie să fie produs de un producător.

Produsele trebuie să conțină măcar un ingredient.

Un bucătar poate să prepare mai multe produse, dar trebuie să prepare minim un produs.

Un client poate plasa una sau mai multe comenzi (devine client după ce plasează minim o comandă).

O factură poate conține o singură comandă.

La o factură este atașat un singur casier, dar un casier poate fi atașat la mai multe facturi, nu doar una.

Un angajat trebuie să fie chelner, casier, bucătar sau manager.

1.6 Descrierea atributelor.

- Entitatea **restaurant** va avea ca atrbute id-ul și numele restaurantului curent (de exemplu "Yamas", "Ivans" etc).
Tipul de date pentru id-ul restaurantului va fi number(10), id-locație number(10), iar pentru nume va fi varchar2(32).
Cheia externă va fi id_locație (provenită din tabelul locație), cu tipul de date number(10).
PK-ul va fi id-ul restaurantului (cheia primară).
Constrângere pentru numele restaurantului: Unique.

- Entitatea **locație** va avea ca atrbute: id-ul locației, țara, orașul, codul poștal și strada pentru restaurantul respectiv.
Tipul de date pentru id-ul locației va fi number(10), țara varchar2(20), orașul varchar2(20), codul poștal varchar2(15) și strada varchar2(50).
PK-ul va fi id-ul locației (cheia primară).
Constrângere pentru codul poștal: Unique.

- Entitatea **produs** va avea ca atrbute: id-ul produsului, numele, cantitatea și o scurtă descriere pentru produsul comercializat.
Tipul de date pentru id-ul produsului va fi number(10), nume varchar2(25), gramaj number(10) și descriere varchar2(100).
PK-ul va fi id-ul produsului (cheia primară).
Constrângere pentru gramajul produsului: check gramaj > 0.

- Tabelul asociativ **meniu** va avea ca atrbute: id-ul produsului, id-ul restaurantului și prețul. Avem o cheie primară compusă formată din id-ul produsului (FK din tabelul *produs*) și id-ul restaurantului (FK din tabelul *restaurant*).
 Tipul de date pentru id-ul produsului va fi *number(10)*, id-ul restaurantului *number(10)* și prețul *number(10)*.
 Constrângere pentru prețul meniului: check preț > 0.
- Tabelul asociativ **cantitate_produs** va avea ca atrbute: id-ul produsului, id-ul ingredientului și cantitatea.
 Avem o cheie primară compusă formată din id-ul produsului (FK din tabelul *produs*) și id-ul ingredient (FK din tabelul *ingredient*).
 Tipul de date pentru id-ul produsului va fi *number(10)*, id-ul ingredientului *number(12)* și cantitatea *number(10)*.
 Constrângere pentru cantitate: check cantitate >= 0.
- Entitatea **ingredient** va avea ca atrbute: id-ul și numele ingredientului.
 Ingredient va avea ca cheie externă id-producător (provenită din tabelul *producător*).
 Tipul de date pentru id-ul ingredientului va fi *number(12)*, id-ul producătorului *number(10)*, iar pentru numele ingredientului *varchar2(32)*.
 PK-ul va fi id-ul ingredientului (cheia primară).
- Entitatea **producător** va avea ca atrbute: id-ul, numele și numărul de telefon al producătorului respectiv.
 Tipul de date pentru id va fi *number(10)*, numele *varchar2(32)* și numărul de telefon *varchar2(15)*.
 PK-ul va fi id-ul producătorului (cheia primară).
 Constrângere pentru numărul de telefon: Unique.
- Entitatea **comandă** va avea ca atrbute: id-ul, prețul și data unei comenzi plasate de client, și va avea ca cheie externă id-ul clientului (provenită din tabelul *client*), dar și id-ul facturii (provenită din tabelul *factură*).
 Tipul de date pentru id-ul comenzi va fi *number(20)*, id-ul clientului *number(10)*, id-ul facturii *number(15)*, data *date*, iar prețul *number(10)*.
 PK-ul va fi id-ul comenzi (cheia primară).
 Constrângere pentru prețul comenzi: NOT_NULL, dar și check preț > 0.
 Atributul data va avea setat ca default ziua curentă (sysdate).
- Tabelul asociativ **conținut_comandă** va avea ca atrbute: id-ul produsului, id-ul comenzi și numărul de produse.

Avem o cheie primară compusă formată din id-ul produsului (FK din tabelul *produs*) și id-ul comenzi (FK din *comandă*).

Tipul de date pentru id-ul produsului va fi number(10), id-ul comenzi number(20) și numărul de produse din comandă number(10).

Constrângere pentru numărul de produse: check *număr_produse > 0*.

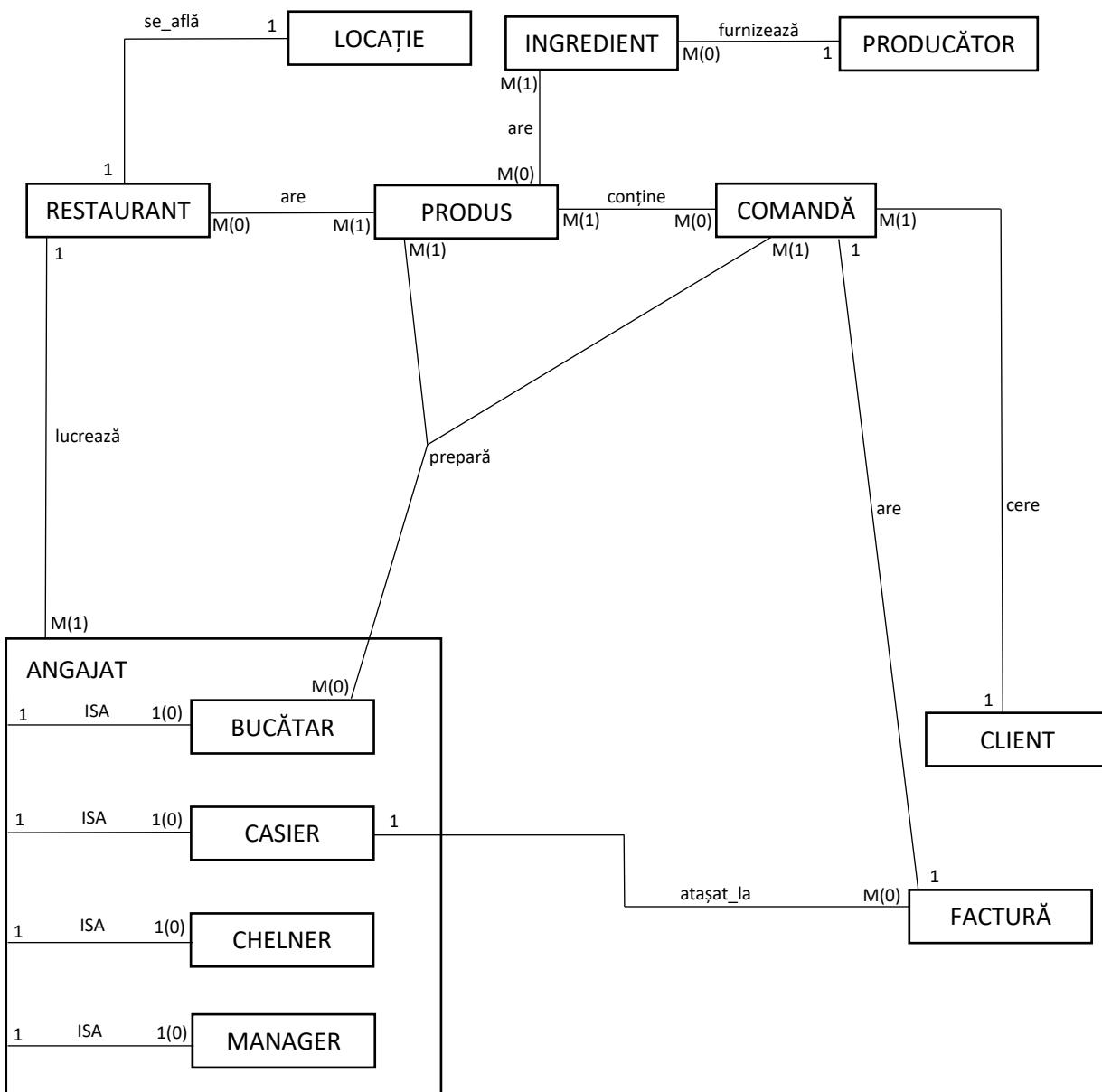
- Entitatea **angajat** va avea ca atrbute: id-ul, id-ul şefului, numele, prenumele și data angajării (am ales această abordare pentru că niciun subtip nu avea atrbute separate de angajat).
Angajat va avea ca cheie externă id-restaurant (provenită din tabelul *restaurant*) și id-şef ca fiind cheie externă auto-referențială (provenită din același tabel, *angajat*).
Tipul de date pentru id-ul angajatului va fi number(10), id-ul şefului number(10) dar care poate fi NULL, id-ul restaurantului number(10), nume varchar2(32), prenume varchar2(32), iar data angajării date.
Atributul *data_angajare* va avea setat ca default ziua curentă (sysdate).
- Subentitatea **bucătar** va avea ca atrbute id-ul angajatului și numărul de stele.
Tipul de date pentru id-ul angajatului va fi number(10), iar pentru numărul de stele number(7).
PK va fi *id_angajat* (care provine din tabelul *angajat*).
- Subentitatea **casier** va avea ca atrbute id-ul angajatului și numărul de ani de studii.
Tipul de date pentru id-ul angajatului va fi number(10), iar pentru numărul de ani de studii number(5).
PK va fi *id_angajat* (care provine din tabelul *angajat*).
- Subentitatea **chelner** va avea ca atrbute id-ul angajatului și numărul de ani de experiență.
Tipul de date pentru id-ul angajatului va fi number(10), iar pentru numărul de ani de experiență number(5).
PK va fi *id_angajat* (care provine din tabelul *angajat*).
- Subentitatea **manager** va avea ca atrbute id-ul angajatului.
Tipul de date pentru id-ul angajatului va fi number(10).
PK va fi *id_angajat* (care provine din tabelul *angajat*).
- Tabelul asociativ **preparare** va avea ca atrbute: id-ul produsului, id-ul comenzi, id-ul bucătarului și durata.

Avem o cheie primară compusă formată din id-ul produsului (FK din tabelul produs), id-ul comenzi (FK din comandă) și id-ul bucătarului (FK din angajat), acestea reies din relația de tip 3 dintre produs, comandă și bucătar.

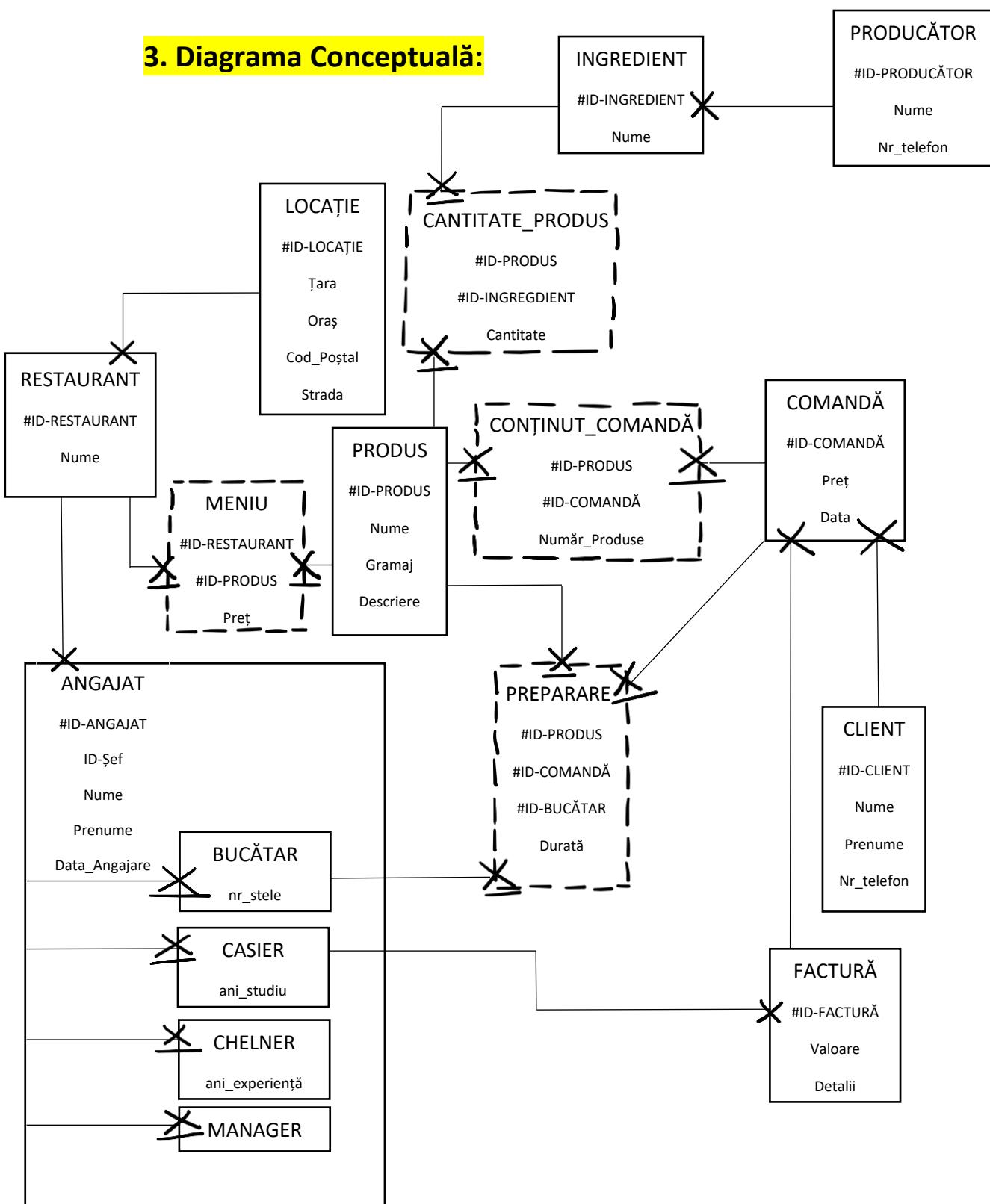
Tipul de date pentru id-ul produsului va fi number(10), id-ul comenzi number(20), id-ul bucătarului number(10) și durata number(10).

- Entitatea **client** va avea ca atrbute: id-ul, numele, prenumele și numărul de telefon al unui client.
Tipul de date pentru id-ul clientului va fi number(10), numele varchar2(32), prenumele varchar2(32) și numărul de telefon varchar2(15).
Constrângere pentru numele și prenumele: NOT_NULL.
PK-ul va fi id-ul clientului (cheia primară).
- Entitatea **factură** va avea ca atrbute: id-ul, valoarea și câteva mici detalii legate de factură.
Constrângere pentru valoare : NOT_NULL. Va avea cheia externă id-ul id-ul casierului.
Tipul de date pentru id-ul facturii va fi number(15), valoarea number(10), detalii varchar2(50) și id-ul casierului number(10).
Constrângere pentru valoarea facturii: check valoare > 0.
PK-ul va fi id-ul facturii (cheia primară).

2. Diagrama Entitate-Relație (ER):



3. Diagrama Conceptuală:



4. Schema relatională:

- RESTAURANT (#ID-RESTAURANT, id-locație, Nume)
- LOCAȚIE (#ID-LOCAȚIE, Tara, Oraș, Cod_Poștal, Strada)
- MENIU (#ID-RESTAURANT, #ID-PRODUS, Preț)
- PRODUS (#ID-PRODUS, Nume, Gramaj, Descriere)
- CANTITATE_PRODUS (#ID-PRODUS, #ID-INGREDIENT, Cantitate)
- INGREDIENT (#ID-INGREDIENT, id-producător, Nume)
- PRODUCĂTOR (#ID-PRODUCĂTOR, Nume, Nr_telefon)
- CONȚINUT_COMANDĂ (#ID-PRODUS, #ID-COMANDĂ, Număr_Produse)
- COMANDĂ (#ID-COMANDĂ, Preț, Data, id-factură, id-client)
- CLIENT (#ID-CLIENT, Nume, Prenume, Nr_telefon)
- ANGAJAT (#ID-ANGAJAT, id-șef, id-restaurant, Nume, Prenume, data_angajare)
- BUCĂTAR (#ID-ANGAJAT, nr_stele)
- CASIER (#ID-ANGAJAT, ani_studiu)
- CHELNER (#ID-ANGAJAT, ani_experiență)
- MANAGER (#ID-ANGAJAT)
- FACTURĂ (#ID-FACTURĂ, Valoare, Detalii, id-casier)
- PREPARARE (#ID-PRODUS, #ID-COMANDĂ, #ID-BUCĂTAR, Durată)

5. Normalizarea până la forma normal 3 (FN1 – FN3):

Schema noastră ar fi scoasă din FN1 dacă entitatea produs ar conține atributul ingredient, întrucât un produs are mai multe ingrediente (de exemplu piper, boia, sare etc).

Schema Non-FN1:

PRODUS (#ID-PRODUS, Nume, Gramaj, Descriere, **Ingrediente**)

Schema noastră ar fi scoasă din FN2 dacă în loc să existe două entități ingredient și producător ar exista doar entitatea Ingredient (#id-ingredient, #id-producător, nume_ingredient, nume_producător, nr_telefon_producător), pentru ca nr de telefon nu depinde de id-ul ingredientului.

Schema Non-FN2:

INGREDIENT (#ID-INGREDIENT, #ID-PRODUCĂTOR, Nume_ingredient, Nume_Producător, **Nr_telefon**)

Schema noastră ar fi scoasă din FN3 dacă entitatea comandă ar conține ca atribut numărul de telefon al clientului (dependență tranzitivă între oraș-țara).

Schema Non-FN3:

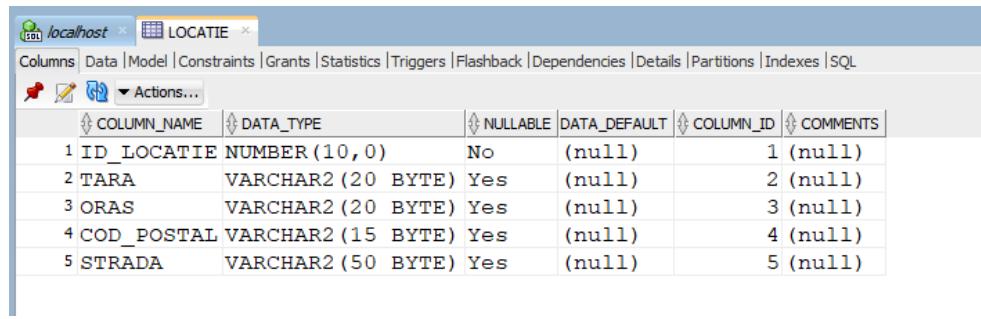
COMANDĂ (#ID-COMANDĂ, Preț, Data, id_factură, id_client, **nr_telefon_client**)

6. Implementarea bazei de date în Oracle:

6.1 Crearea tabelelor și a constrângerilor.

```
--- CREAREA TABELELOR ---  
  
-- LOCATIE --  
CREATE TABLE locatie (  
    id_locatie    NUMBER(10) PRIMARY KEY,  
    tara          VARCHAR2(20),  
    oras          VARCHAR2(20),  
    cod_postal    VARCHAR2(15) UNIQUE,  
    strada        VARCHAR2(50)  
);
```

Print-Screen:

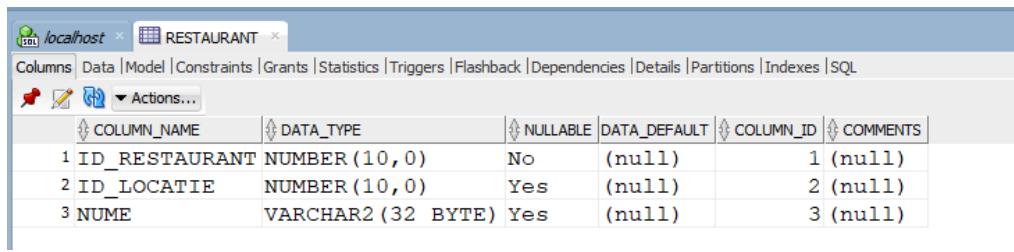


The screenshot shows the Oracle SQL Developer interface with a database connection to 'localhost'. A table named 'LOCATIE' is selected. The 'Columns' tab is active, displaying the following column information:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_LOCATIE	NUMBER(10, 0)	No	(null)	1	(null)
2 TARA	VARCHAR2(20 BYTE)	Yes	(null)	2	(null)
3 ORAS	VARCHAR2(20 BYTE)	Yes	(null)	3	(null)
4 COD_POSTAL	VARCHAR2(15 BYTE)	Yes	(null)	4	(null)
5 STRADA	VARCHAR2(50 BYTE)	Yes	(null)	5	(null)

```
-- RESTAURANT --  
CREATE TABLE restaurant (  
    id_restaurant    NUMBER(10) PRIMARY KEY,  
    id_locatie        NUMBER(10),  
    nume              VARCHAR2(32) UNIQUE,  
    CONSTRAINT fk_restaurant_locatie FOREIGN KEY ( id_locatie )  
        REFERENCES locatie ( id_locatie )  
);
```

Print-Screen:



The screenshot shows the Oracle SQL Developer interface with a database connection to 'localhost'. A table named 'RESTAURANT' is selected. The 'Columns' tab is active, displaying the following column information:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_RESTAURANT	NUMBER(10, 0)	No	(null)	1	(null)
2 ID_LOCATIE	NUMBER(10, 0)	Yes	(null)	2	(null)
3 NUME	VARCHAR2(32 BYTE)	Yes	(null)	3	(null)

```
-- PRODUS --
CREATE TABLE produs (
    id_produs    NUMBER(10) PRIMARY KEY,
    nume         VARCHAR2(25),
    gramaj       NUMBER(10),
    descriere    VARCHAR2(100),
    CONSTRAINT chk_gramaj CHECK ( gramaj > 0 )
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER(10,0)	No	(null)	1	(null)
2 NUME	VARCHAR2(25 BYTE)	Yes	(null)	2	(null)
3 GRAMAJ	NUMBER(10,0)	Yes	(null)	3	(null)
4 DESCRIERE	VARCHAR2(100 BYTE)	Yes	(null)	4	(null)

```
-- MENIU --
CREATE TABLE meniu (
    id_produs      NUMBER(10),
    id_restaurant NUMBER(10),
    pret           NUMBER(10),
    CONSTRAINT meniu_pk PRIMARY KEY ( id_produs,
                                         id_restaurant ),
    CONSTRAINT meniu_produs_fk FOREIGN KEY ( id_produs )
        REFERENCES produs ( id_produs ),
    CONSTRAINT meniu_restaurant_fk FOREIGN KEY ( id_restaurant )
        REFERENCES restaurant ( id_restaurant ),
    CONSTRAINT chk_pret CHECK ( pret > 0 )
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER(10,0)	No	(null)	1	(null)
2 ID_RESTAURANT	NUMBER(10,0)	No	(null)	2	(null)
3 PRET	NUMBER(10,0)	Yes	(null)	3	(null)

```
-- PRODUCATOR --
CREATE TABLE producator (
    id_producator      NUMBER(10) PRIMARY KEY,
    nume                VARCHAR2(32),
    nr_telefon          VARCHAR2(15) UNIQUE
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUCATOR	NUMBER (10, 0)	No	(null)	1	(null)
2 NUME	VARCHAR2 (32 BYTE)	Yes	(null)	2	(null)
3 NR_TELEFON	VARCHAR2 (15 BYTE)	Yes	(null)	3	(null)

```
-- INGREDIENT --
CREATE TABLE ingredient (
    id_ingredient      NUMBER(12) PRIMARY KEY,
    id_producator      NUMBER(10),
    nume                VARCHAR2(32),
    CONSTRAINT fk_ingredient_producator FOREIGN KEY ( id_producator )
        REFERENCES producator ( id_producator )
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_INGREDIENT	NUMBER (12, 0)	No	(null)	1	(null)
2 ID_PRODUCATOR	NUMBER (10, 0)	Yes	(null)	2	(null)
3 NUME	VARCHAR2 (32 BYTE)	Yes	(null)	3	(null)

```
-- CANTITATE_PRODUS --
CREATE TABLE cantitate_produs (
    id_produs      NUMBER(10),
    id_ingredient  NUMBER(12),
    cantitate      NUMBER(10),
    CONSTRAINT cantitate_produs_pk PRIMARY KEY ( id_produs,
                                                id_ingredient ),
    CONSTRAINT cantitate_produs_fk FOREIGN KEY ( id_produs )
        REFERENCES produs ( id_produs ),
    CONSTRAINT cantitate_ingredient_fk FOREIGN KEY ( id_ingredient )
        REFERENCES ingredient ( id_ingredient ),
    CONSTRAINT chk_cantitate_produs CHECK ( cantitate > 0 )
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER(10,0)	No	(null)	1	(null)
2 ID_INGREDIENT	NUMBER(12,0)	No	(null)	2	(null)
3 CANTITATE	NUMBER(10,0)	Yes	(null)	3	(null)

```
-- ANGAJAT --
CREATE TABLE angajat (
    id_angajat      NUMBER(10) PRIMARY KEY,
    id_restaurant   NUMBER(10) NOT NULL,
    id_sef           NUMBER(10),
    nume             VARCHAR2(32),
    prenume          VARCHAR2(32),
    data_angajare    DATE DEFAULT to_date(sysdate, 'dd-mm-yy'),
    CONSTRAINT angajat_restaurant_fk FOREIGN KEY ( id_restaurant )
        REFERENCES restaurant (id_restaurant),
    CONSTRAINT angajat_sef_fk FOREIGN KEY ( id_sef )
        REFERENCES angajat (id_angajat)
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_ANGAJAT	NUMBER(10,0)	No	(null)	1	(null)
2 ID_RESTAURANT	NUMBER(10,0)	No	(null)	2	(null)
3 ID_SEF	NUMBER(10,0)	Yes	(null)	3	(null)
4 NUME	VARCHAR2(32 BYTE)	Yes	(null)	4	(null)
5 PRENUME	VARCHAR2(32 BYTE)	Yes	(null)	5	(null)
6 DATA_ANGAJARE	DATE	Yes	to_date(sysdate, 'dd-mm-yy')	6	(null)

```
-- CHELNER --
CREATE TABLE chelner (
    id_angajat      NUMBER(10),
    ani_experienta  NUMBER(5),
    CONSTRAINT chelner_pk PRIMARY KEY ( id_angajat ),
    CONSTRAINT angajat_fk FOREIGN KEY ( id_angajat )
        REFERENCES angajat ( id_angajat )
);
```

Print-Screen:

The screenshot shows the 'CHELNER' table structure in Oracle SQL Developer. The table has two columns: 'ID_ANGAJAT' and 'ANI_EXPERIENTA'. Both columns are of type NUMBER with precision 10 and scale 0. The column 'ID_ANGAJAT' is nullable ('No') and has a default value of '(null)'. The column 'ANI_EXPERIENTA' is nullable ('Yes') and has a default value of '(null)'. The table has a primary key constraint named 'chelner_pk' on the 'ID_ANGAJAT' column and a foreign key constraint named 'angajat_fk' on the 'ID_ANGAJAT' column, referencing the 'id_angajat' column of the 'angajat' table.

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_ANGAJAT	NUMBER (10, 0)	No	(null)	1	(null)
2 ANI_EXPERIENTA	NUMBER (5, 0)	Yes	(null)	2	(null)

```
-- CASIER --
CREATE TABLE casier (
    id_angajat      NUMBER(10),
    ani_studiu     NUMBER(5),
    CONSTRAINT casier_pk PRIMARY KEY ( id_angajat ),
    CONSTRAINT angajat_casier_fk FOREIGN KEY ( id_angajat )
        REFERENCES angajat ( id_angajat )
);
```

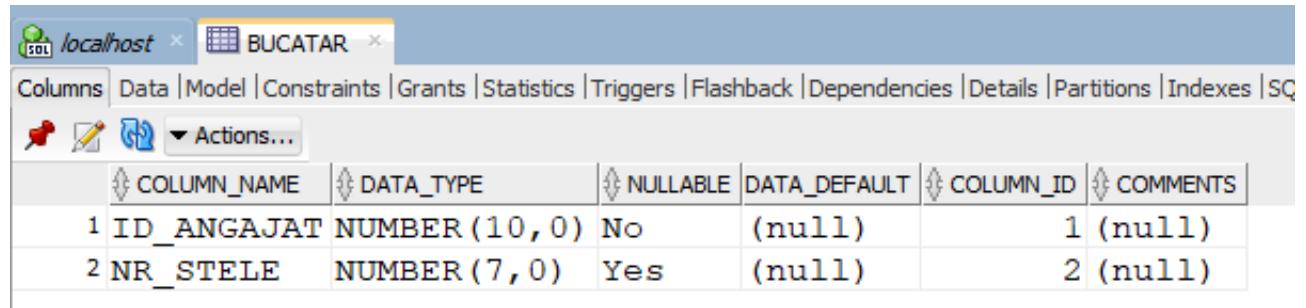
Print-Screen:

The screenshot shows the 'CASIER' table structure in Oracle SQL Developer. The table has two columns: 'ID_ANGAJAT' and 'ANI_STUDIU'. Both columns are of type NUMBER with precision 10 and scale 0. The column 'ID_ANGAJAT' is nullable ('No') and has a default value of '(null)'. The column 'ANI_STUDIU' is nullable ('Yes') and has a default value of '(null)'. The table has a primary key constraint named 'casier_pk' on the 'ID_ANGAJAT' column and a foreign key constraint named 'angajat_casier_fk' on the 'ID_ANGAJAT' column, referencing the 'id_angajat' column of the 'angajat' table.

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_ANGAJAT	NUMBER (10, 0)	No	(null)	1	(null)
2 ANI_STUDIU	NUMBER (5, 0)	Yes	(null)	2	(null)

```
-- BUCATAR --
CREATE TABLE bucatar (
    id_angajat    NUMBER(10),
    nr_stele      NUMBER(7),
    CONSTRAINT bucatar_pk PRIMARY KEY ( id_angajat ),
    CONSTRAINT angajat_bucatar_fk FOREIGN KEY ( id_angajat )
        REFERENCES angajat ( id_angajat )
);
```

Print-Screen:



localhost > BUCATAR >

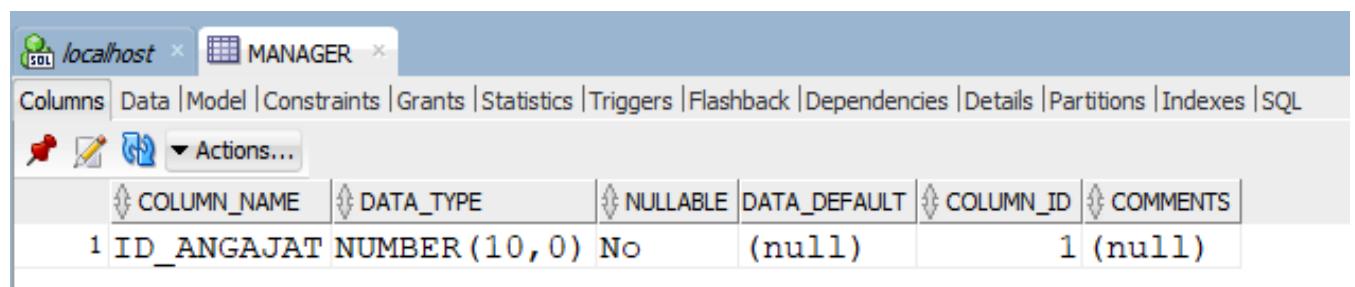
Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies Details Partitions Indexes SQL

Actions... ▾

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_ANGAJAT	NUMBER (10, 0)	No	(null)	1	(null)
2 NR_STELE	NUMBER (7, 0)	Yes	(null)	2	(null)

```
-- MANAGER --
CREATE TABLE manager (
    id_angajat NUMBER(10),
    CONSTRAINT manager_pk PRIMARY KEY ( id_angajat ),
    CONSTRAINT angajat_manager_fk FOREIGN KEY ( id_angajat )
        REFERENCES angajat ( id_angajat )
);
```

Print-Screen:



localhost > MANAGER >

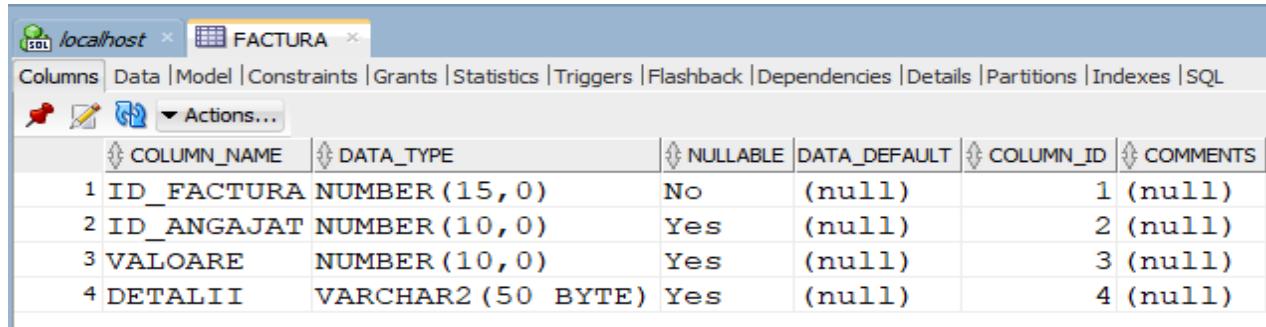
Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies Details Partitions Indexes SQL

Actions... ▾

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_ANGAJAT	NUMBER (10, 0)	No	(null)	1	(null)

```
-- FACTURA --
CREATE TABLE factura (
    id_factura    NUMBER(15) PRIMARY KEY,
    id_angajat    NUMBER(10),
    valoare        NUMBER(10),
    detalii        VARCHAR2(50),
    CONSTRAINT fk_factura_casier FOREIGN KEY ( id_angajat )
        REFERENCES casier ( id_angajat ),
    CONSTRAINT chk_factura_valoare CHECK ( valoare > 0 )
);
```

Print-Screen:

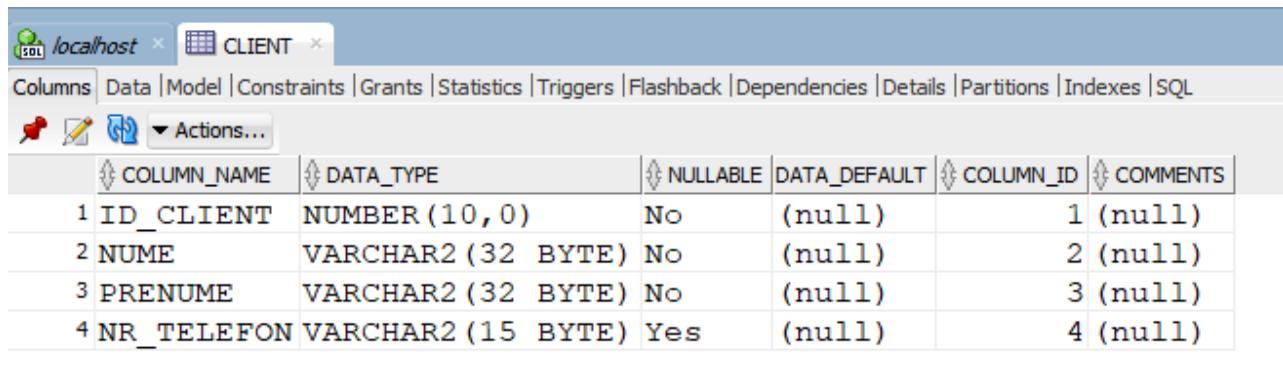


The screenshot shows the 'Columns' tab for the FACTURA table in Oracle SQL Developer. The table has four columns: ID_FACTURA, ID_ANAJAT, VALOARE, and DETALII. The primary key is ID_FACTURA.

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_FACTURA	NUMBER (15, 0)	No	(null)	1	(null)
2 ID_ANAJAT	NUMBER (10, 0)	Yes	(null)	2	(null)
3 VALOARE	NUMBER (10, 0)	Yes	(null)	3	(null)
4 DETALII	VARCHAR2 (50 BYTE)	Yes	(null)	4	(null)

```
-- CLIENT --
CREATE TABLE client (
    id_client    NUMBER(10) PRIMARY KEY,
    nume          VARCHAR2(32) NOT NULL,
    prenume       VARCHAR2(32) NOT NULL,
    nr_telefon   VARCHAR2(15)
);
```

Print-Screen:



The screenshot shows the 'Columns' tab for the CLIENT table in Oracle SQL Developer. The table has four columns: ID_CLIENT, NUME, PRENUME, and NR_TELEFON. The primary key is ID_CLIENT.

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_CLIENT	NUMBER (10, 0)	No	(null)	1	(null)
2 NUME	VARCHAR2 (32 BYTE)	No	(null)	2	(null)
3 PRENUME	VARCHAR2 (32 BYTE)	No	(null)	3	(null)
4 NR_TELEFON	VARCHAR2 (15 BYTE)	Yes	(null)	4	(null)

```
-- COMANDA --
CREATE TABLE comanda (
    id_comanda    NUMBER(20) PRIMARY KEY,
    id_client     NUMBER(10),
    id_factura    NUMBER(15),
    pret          NUMBER(10) NOT NULL,
    data          DATE DEFAULT to_date(sysdate, 'dd-mm-yy'),
    CONSTRAINT fk_comanda_client FOREIGN KEY ( id_client )
        REFERENCES client ( id_client ),
    CONSTRAINT fk_comanda_factura FOREIGN KEY ( id_factura )
        REFERENCES factura ( id_factura ),
    CONSTRAINT chk_comanda CHECK ( pret > 0 )
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_COMANDA	NUMBER(20, 0)	No	(null)	1	(null)
2 ID_CLIENT	NUMBER(10, 0)	Yes	(null)	2	(null)
3 ID_FACTURA	NUMBER(15, 0)	Yes	(null)	3	(null)
4 PRET	NUMBER(10, 0)	No	(null)	4	(null)
5 DATA	DATE	Yes	to_date(sysdate, 'dd-mm-yy')	5	(null)

```
-- CONTINUT_COMANDA --
CREATE TABLE continut_comanda (
    id_produs      NUMBER(10),
    id_comanda     NUMBER(20),
    numar_produse  NUMBER(10),
    CONSTRAINT continut_comanda_pk PRIMARY KEY ( id_produs,
                                                id_comanda ),
    CONSTRAINT continut_produs_fk FOREIGN KEY ( id_produs )
        REFERENCES produs ( id_produs ),
    CONSTRAINT continut_comanda_fk FOREIGN KEY ( id_comanda )
        REFERENCES comanda ( id_comanda ),
    CONSTRAINT chk_continut_comanda CHECK ( numar_produse > 0 )
);
```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER(10, 0)	No	(null)	1	(null)
2 ID_COMANDA	NUMBER(20, 0)	No	(null)	2	(null)
3 NUMAR_PRODUSE	NUMBER(10, 0)	Yes	(null)	3	(null)

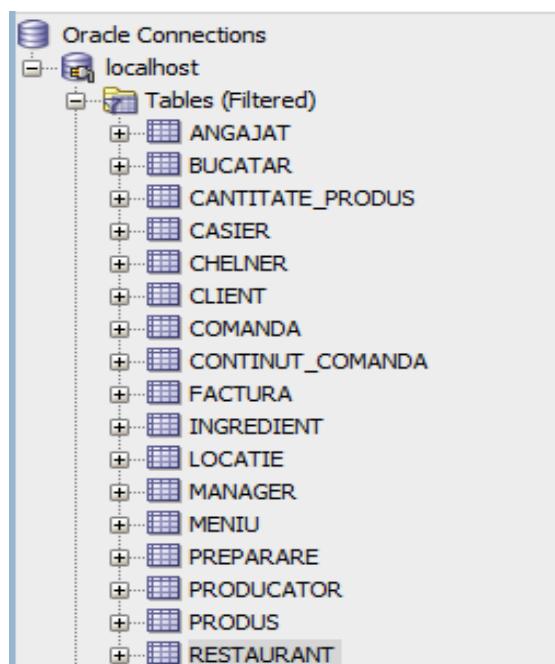
```
-- PREPARARE --
CREATE TABLE preparare (
    id_produs      NUMBER(10),
    id_comanda     NUMBER(20),
    id_angajat     NUMBER(10),
    durata         NUMBER(10),
    CONSTRAINT preparare_pk PRIMARY KEY ( id_produs,
                                            id_comanda,
                                            id_angajat ),
    CONSTRAINT preparare_casier_fk FOREIGN KEY ( id_angajat )
        REFERENCES bucatar ( id_angajat ),
    CONSTRAINT preparare_produs_fk FOREIGN KEY ( id_produs )
        REFERENCES produs ( id_produs ),
    CONSTRAINT preparare_comanda_fk FOREIGN KEY ( id_comanda )
        REFERENCES comanda ( id_comanda )
);

```

Print-Screen:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 ID_PRODUS	NUMBER (10, 0)	No	(null)	1	(null)
2 ID_COMANDA	NUMBER (20, 0)	No	(null)	2	(null)
3 ID_ANGAJAT	NUMBER (10, 0)	No	(null)	3	(null)
4 DURATA	NUMBER (10, 0)	Yes	(null)	4	(null)

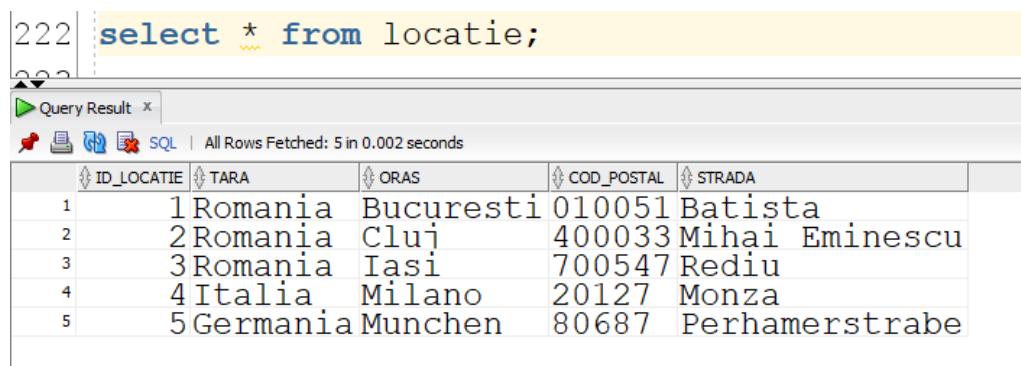
Print-Screen:



6.2 Inserarea datelor coerente în tabele (minimum 5 înregistrări în fiecare tabel neasociativ, minimum 10 înregistrări în tabelele asociative) + crearea unei secvențe.

```
--- INSERAREA DATELOR IN TABELE ---  
  
-- PENTRU TABELUL LOCATIE --  
create sequence id_locatie  
start with 1  
increment by 1  
minvalue 0  
maxvalue 9999  
nocycle;  
  
insert into locatie  
values (id_locatie.nextval, 'Romania' , 'Bucuresti', '010051',  
'Batista'); --1  
  
insert into locatie  
values (id_locatie.nextval, 'Romania', 'Cluj', '400033', 'Mihai  
Eminescu'); --2  
  
insert into locatie  
values (id_locatie.nextval, 'Romania', 'Iasi', '700547', 'Rediu'); --  
3  
  
insert into locatie  
values (id_locatie.nextval, 'Italia', 'Milano', '20127', 'Monza'); --  
4  
  
insert into locatie  
values (id_locatie.nextval, 'Germania', 'Munchen', '80687',  
'Perhamerstrabe'); --5
```

Print-Screen:



The screenshot shows a MySQL Workbench interface with a query editor and a results grid. The query editor contains the following SQL code:

```
222 | select * from locatie;
```

The results grid displays the following data:

ID_LOCATIE	TARA	ORAS	COD_POSTAL	STRADA
1	Romania	Bucuresti	010051	Batista
2	Romania	Cluj	400033	Mihai Eminescu
3	Romania	Iasi	700547	Rediu
4	Italia	Milano	20127	Monza
5	Germania	Munchen	80687	Perhamerstrabe

```
-- PENTRU TABELUL RESTAURANT --
create sequence id_restaurant
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into restaurant
values (id_restaurant.nextval, 2, 'Gurmandul'); --1

insert into restaurant
values (id_restaurant.nextval, 3, 'Yamas'); --2

insert into restaurant
values (id_restaurant.nextval, 1, 'Ivans'); --3

insert into restaurant
values (id_restaurant.nextval, 5, 'Savanna'); --4

insert into restaurant
values (id_restaurant.nextval, 4, 'Grande appetito!'); --5
```

Print-Screen:

The screenshot shows a SQL query being run in Oracle SQL Developer. The query is:

```
249 | select * from restaurant;
```

The results are displayed in a table titled "Query Result". The table has three columns: ID_RESTAURANT, ID_LOCATIE, and NUME. The data is as follows:

ID_RESTAURANT	ID_LOCATIE	NUME
1	1	Gurmandul
2	2	Yamas
3	3	Ivans
4	4	Savanna
5	5	Grande appetito!

At the bottom of the interface, it says "All Rows Fetched: 5 in 0.002 seconds".

```
-- PENTRU TABELUL MENIU --
-- pretul este in lei --
insert into meniu (id_restaurant, id_produs, pret)
values (4, 1, 35); --1

insert into meniu (id_restaurant, id_produs, pret)
values (4, 2, 50); --2

insert into meniu (id_restaurant, id_produs, pret)
```

```

values (1, 5, 20); --3

insert into meniu (id_restaurant, id_produs, pret)
values (1, 4, 33); --4

insert into meniu (id_restaurant, id_produs, pret)
values (5, 2, 60); --5

insert into meniu (id_restaurant, id_produs, pret)
values (2, 4, 15); --6

insert into meniu (id_restaurant, id_produs, pret)
values (5, 4, 13); --7
commit;

insert into meniu (id_restaurant, id_produs, pret)
values (3, 5, 12); --8

insert into meniu (id_restaurant, id_produs, pret)
values (4, 4, 30); --9

insert into meniu (id_restaurant, id_produs, pret)
values (3, 4, 15); --10

insert into meniu (id_restaurant, id_produs, pret)
values (2, 3, 27); --11

```

Print-Screen:

The screenshot shows the MySQL Workbench interface. The top part is the 'Worksheet' tab where the SQL query `select * from meniu;` is entered. The bottom part is the 'Query Result' tab, which shows the results of the query in a table format.

	ID_PRODUS	ID_RESTAURANT	PRET
1	1	4	35
2	2	4	50
3	5	1	20
4	4	1	33
5	2	5	60
6	4	2	15
7	4	5	13
8	5	3	12
9	4	4	30
10	4	3	15
11	3	2	27

```
-- PENTRU TABELUL PRODUS --
-- aici cantitatea este in grame
create sequence id_produs
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into produs
values (id_produs.nextval, 'Spaghete', 200, 'picante'); --1

insert into produs
values (id_produs.nextval, 'Pizza', 150, 'dulce aromata'); --2

insert into produs
values (id_produs.nextval, 'Spaghete', 200, 'picante'); --3

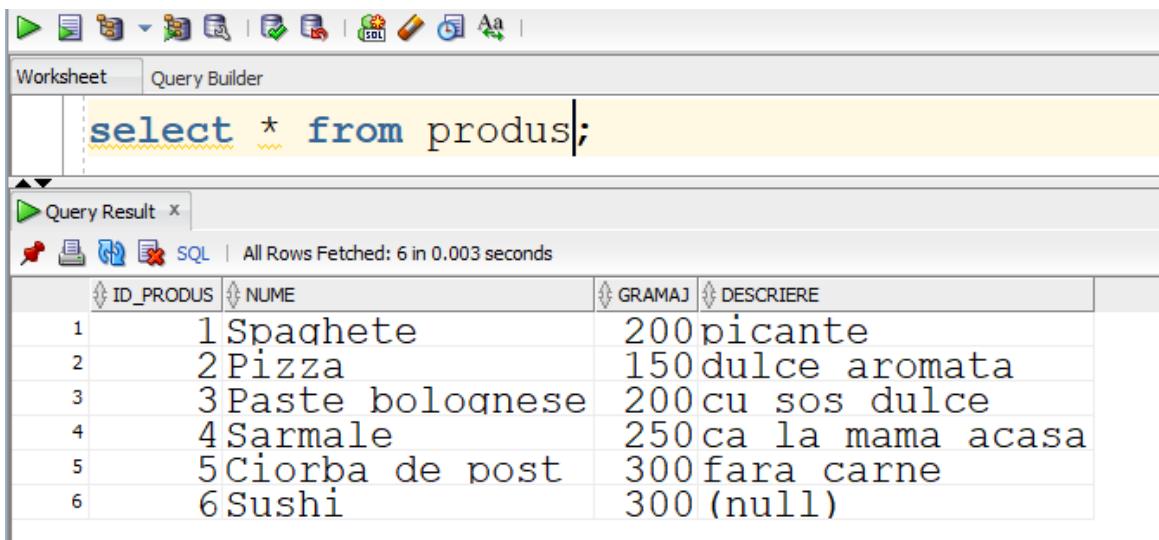
-- aici am updatat linia pentru paste bolognese
update produs
set nume = 'Paste bolognese', descriere = 'cu sos dulce'
where id_produs = 3;

insert into produs
values (id_produs.nextval, 'Sarmale', 250, 'ca la mama acasa'); --4

insert into produs
values (id_produs.nextval, 'Ciorba de post', 300, 'fara carne'); --5

insert into produs(id_produs,nume,gramaj)
values (id_produs.nextval, 'Sushi', 300); --6
```

Print-Screen:



ID_PRODUS	NUME	GRAMAJ	DESCRIERE
1	Spaghete	200	picante
2	Pizza	150	dulce aromata
3	Paste bolognese	200	cu sos dulce
4	Sarmale	250	ca la mama acasa
5	Ciorba de post	300	fara carne
6	Sushi	300	(null)

```
-- PENTRU TABELUL CANTITATE_PRODUS --
-- cantitatea este masurata in grame --
insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (1, 1, 85); --1

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (1, 4, 50); --2

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (2, 4, 60); --3

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (2, 3, 150); --4

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (2, 1, 100); --5

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (3, 4, 200); --6

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (3, 5, 55); --7

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (4, 4, 100); --8

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (4, 3, 500); --9

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (5, 5, 350); --10

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (2, 6, 50); --11

insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (2, 7, 35); --12
```

Print-Screen:

--

551 | **select * from cantitate_produs;**

Script Output x Query Result x

SQL | All Rows Fetched: 12 in 0.004 seconds

	ID_PRODUS	ID_INGREDIENT	CANTITATE
1	1	1	85
2	1	4	50
3	2	4	60
4	2	3	150
5	2	1	100
6	3	4	200
7	3	5	55
8	4	4	100
9	4	3	500
10	5	5	350
11	2	6	50
12	2	7	35

```
-- PENTRU TABELUL INGREDIENT --
create sequence id_ingredient
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into ingredient
values (id_ingredient.nextval, 1, 'rosii'); --1

insert into ingredient
values (id_ingredient.nextval, 3, 'varza murata'); --2

insert into ingredient
values (id_ingredient.nextval, 4, 'ulei floarea soarelui'); --3

insert into ingredient
values (id_ingredient.nextval, 5, 'condimente'); --4

insert into ingredient
values (id_ingredient.nextval, 2, 'bors'); --5

insert into ingredient
values (id_ingredient.nextval, 2, 'parmezan'); --6
```

```
insert into ingredient
values (id_ingredient.nextval, 2, 'cascaval'); --7
```

Print-Screen:

ID_INGREDIENT	ID_PRODUCATOR	NUME
1	1	1 rosii
2	2	3 varza murata
3	3	4 ulei floarea soarelui
4	4	5 condimente
5	5	2 bors
6	6	2 parmezan
7	7	2 cascaval

```
-- PENTRU TABELUL PRODUCATOR --
create sequence id_producator
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into producator
values (id_producator.nextval, 'Livada cu de toate', '0770573182'); --1

insert into producator
values (id_producator.nextval, 'Olivers', '0754754318'); --2

insert into producator
values (id_producator.nextval, 'FreshOnly', '021999123'); --3

insert into producator
values (id_producator.nextval, 'ION MOS', '021129123'); --4

insert into producator
values (id_producator.nextval, 'FUCHS', '021592555'); --5

insert into producator
values (id_producator.nextval, 'Bergenbier', '026662555'); --6

insert into producator
values (id_producator.nextval, 'Tuborg', '072592555'); --7
```

Print-Screen:

The screenshot shows a SQL query being run in Oracle SQL Developer. The query is:

```
select * from producator;
```

The results are displayed in a table titled "Query Result". The table has three columns: ID_PRODUCATOR, NUME, and NR_TELEFON. The data is as follows:

ID_PRODUCATOR	NUME	NR_TELEFON
1	Livada cu de toate	0770573182
2	Olivers	0754754318
3	FreshOnly	021999123
4	ION MOS	021129123
5	FUCHS	021592555
6	Bergenbier	026662555
7	Tubora	072592555

```
-- PENTRU TABELUL CONTINUT_COMANDA --
insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (1, 4, 2); --1

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (1, 2, 1); --2

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (2, 2, 1); --3

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (2, 1, 1); --4

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (3, 5, 1); --5

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (3, 4, 1); --6

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (4, 5, 1); --7

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (5, 4, 1); --8

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (5, 2, 1); --9

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (3, 3, 1); --10

insert into continut_comanda (id_produs, id_comanda, numar_produse)
```

```

values (2, 6, 2); --11

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (1, 7, 2); --12

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (2, 7, 1); --13

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (3, 7, 1); --14

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (1, 8, 1); --15

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (2, 8, 3); --16

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (3, 8, 1); --17

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (4, 9, 2); --18

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (5, 9, 1); --19

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (2, 10, 1); --20

```

Print-Screen:

The screenshot shows a MySQL Workbench interface with the following details:

- Query Editor:** The query `select * from continut_comanda;` is entered.
- Results Tab:** The results are displayed in a table titled "Query Result".
- Table Headers:** The columns are labeled `ID_PRODUS`, `ID_COMANDA`, and `NUMAR_PRODUSE`.
- Data Rows:** There are 20 rows of data, each consisting of three values: ID_PRODUS, ID_COMANDA, and NUMAR_PRODUSE.

ID_PRODUS	ID_COMANDA	NUMAR_PRODUSE
1	1	4
2	1	2
3	2	2
4	2	1
5	3	5
6	3	4
7	4	5
8	5	4
9	5	2
10	3	3
11	2	6
12	1	7
13	2	7
14	3	7
15	1	8
16	2	8
17	3	8
18	4	9
19	5	9
20	2	10

```
-- PENTRU TABELUL COMANDA --
create sequence id_comanda
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 4, 50); --1

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 3, 5, 130); --2

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 2, 3, 20); --3

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 4, 2, 230); --4

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 5, 1, 80); --5

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 4, 100); --6

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 1, 50); --7

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 2, 30); --8

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 3, 70); --9

insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 4, 20); --10
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. At the top, there is a command line window with the number '553' and the SQL command 'select * from comanda;'. Below it is a tab bar with 'Script Output', 'Query Result', 'Query Result 1', 'Query Result 2', and the active tab 'SQL'. A status message 'All Rows Fetched: 10 in 0.006 seconds' is displayed. The main area is a grid table with the following data:

ID_COMANDA	ID_CLIENT	ID_FACTURA	PRET	DATA
1	1	1	4	50 30-JAN-25
2	2	3	5	130 30-JAN-25
3	3	2	3	20 30-JAN-25
4	4	4	2	230 30-JAN-25
5	5	5	1	80 30-JAN-25
6	6	1	4	100 30-JAN-25
7	7	1	1	50 30-JAN-25
8	8	1	2	30 30-JAN-25
9	9	1	3	70 30-JAN-25
10	10	1	4	20 30-JAN-25

```
-- PENTRU TABELUL CLIENT --
create sequence id_client
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into client
values (id_client.nextval, 'Ionescu', 'Marian', '0721666212'); --1

insert into client (id_client, nume, prenume, nr_telefon)
values (id_client.nextval, 'Dumitrescu', 'Mircea', '0721611211'); --2

insert into client (id_client, nume, prenume, nr_telefon)
values (id_client.nextval, 'Gheorghe', 'Sebastian', '0744573419'); --3

insert into client (id_client, nume, prenume, nr_telefon)
values (id_client.nextval, 'Salam', 'Florin', '0210116666'); --4

insert into client (id_client, nume, prenume, nr_telefon)
values (id_client.nextval, 'Biju', 'Costel', '0211999913'); --5

insert into client (id_client, nume, prenume)
values (id_client.nextval, 'Hagi', 'Gica'); --6

insert into client (id_client, nume, prenume)
values (id_client.nextval, 'Becali', 'Gigi'); --7
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. The top bar displays the title "PROJECT BD ROBERTTO". Below the title bar is a toolbar with various icons. The main area has two tabs: "Worksheet" and "Query Builder". The "Worksheet" tab is active, showing the SQL query: "select * from client;". The "Query Result" tab is also visible below it, showing the results of the query. The results are presented in a table with columns: ID_CLIENT, NUME, PRENUME, and NR_TELEFON. The data is as follows:

ID_CLIENT	NUME	PRENUME	NR_TELEFON
1	Ionescu	Marian	0721666212
2	Dumitrescu	Mircea	0721611211
3	Gheorghe	Sebastian	0744573419
4	Salam	Florin	0210116666
5	Biju	Costel	0211999913
6	Hagi	Gica	(null)
7	Becali	Gigi	(null)

```
-- PENTRU TABELUL FACTURA --
-- fiecare factura are o valoare care include mai mult taxe etc --
create sequence id_factura
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into factura (id_factura, id_angajat, valoare, detalii)
values (id_factura.nextval, 16, 100, 'CASH'); --1

insert into factura (id_factura, id_angajat, valoare, detalii)
values (id_factura.nextval, 12, 250, 'CARD'); --2

insert into factura (id_factura, id_angajat, valoare, detalii)
values (id_factura.nextval, 8, 50, 'CASH'); --3

insert into factura (id_factura, id_angajat, valoare, detalii)
values (id_factura.nextval, 3, 70, 'CASH'); --4

insert into factura (id_factura, id_angajat, valoare, detalii)
values (id_factura.nextval, 5, 150, 'CARD'); --5
```

Print-Screen:

The screenshot shows a SQL developer interface with a query editor and a results window. The query editor contains the following SQL statement:

```
520 | select * from factura;
521 | 
```

The results window displays the data from the 'factura' table:

ID_FACTURA	ID_ANGAJAT	VALOARE	DETALII
1	16	100	CASH
2	12	250	CARD
3	8	50	CASH
4	3	70	CASH
5	5	150	CARD

```
-- PENTRU TABELUL CASIER --
insert into casier(id_angajat, anि_studiu)
values (3, 10); --1

insert into casier(id_angajat, anि_studiu)
values (16, 2); --2

insert into casier(id_angajat, anि_studiu)
values (5, 0); --3

insert into casier(id_angajat, anि_studiu)
values (12, 1); --4
```

```
insert into casier(id_angajat, ani_studiu)
values (8, 4); --5
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, the number '538' is displayed. Below it, the SQL command 'select * from casier;' is entered in the SQL editor. The top-right corner shows the status 'All Rows Fetched: 5 in 0.003 seconds'. The bottom half of the window displays the 'Query Result' grid, which contains five rows of data with columns labeled 'ID_ANGAJAT' and 'ANI_STUDIU'. The data is as follows:

ID_ANGAJAT	ANI_STUDIU
1	3
2	16
3	5
4	12
5	8
	10
	2
	0
	1
	4

```
-- PENTRU TABELUL ANGAJAT --
create sequence id_angajat
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume)
values (id_angajat.nextval, 1, NULL, 'Popescu', 'Robertto'); --1

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 2, 1, 'Escobar', 'Ricardo', to_date('15-
02-21', 'dd-mm-yy')); --2

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 4, 1, 'Marinescu', 'Teodora',
to_date('10-01-20','dd-mm-yy')); --3

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 3, 2, 'Marinescu', 'Petre', to_date('15-
01-21','dd-mm-yy')); --4

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 2, 2, 'Manole', 'Alexandru', to_date('20-
06-19','dd-mm-yy')); --5

insert into angajat (id_angajat, id_sef, id_restaurant, nume,
prenume)
```

```

values (id_angajat.nextval, 1, 1, 'Voicu', 'Andrei'); --6

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 3, 3, 'Radoi', 'Raisa', to_date('15-05-
20','dd-mm-yy')); --7

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 4, 3, 'Stan', 'Mihnea', to_date('01-01-
20','dd-mm-yy')); --8

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 5, 1, 'Filip', 'Mihnea', to_date('21-09-
20','dd-mm-yy')); --9

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 1, 1, 'Peste', 'Florin', to_date('25-07-
20','dd-mm-yy')); --10

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 2, 2, 'Raducanu', 'Sorin', to_date('01-
04-21','dd-mm-yy')); --11

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 3, 3, 'Bida', 'Marian', to_date('25-02-
21','dd-mm-yy')); --12

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 4, 4, 'Fredo', 'Magiore', to_date('01-05-
21','dd-mm-yy')); --13

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 5, 4, 'Cercel', 'Florin', to_date('02-09-
20','dd-mm-yy')); --14

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 4, 4, 'Stanescu', 'Gigel', to_date('02-
07-20','dd-mm-yy')); --15

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 2, 5, 'Dociu', 'Mihai', to_date('17-08-
20','dd-mm-yy')); --16

```

```

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 1, 1, 'Dumitrescu', 'Florin',
to_date('04-01-25', 'dd-mm-yy')); --17

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 3, 3, 'Sociu', 'Razvan', to_date('29-12-
20', 'dd-mm-yy')); --18

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 5, 6, 'Scarlatescu', 'Catalin',
to_date('01-01-25', 'dd-mm-yy')); --19

insert into angajat (id_angajat, id_restaurant, id_sef, nume,
prenume, data_angajare)
values (id_angajat.nextval, 2, 6, 'Bontea', 'Sorin', to_date('02-01-
25', 'dd-mm-yy')); --20

insert into angajat (id_angajat, id_sef, id_restaurant, nume,
prenume)
values (id_angajat.nextval, 1, 1, 'Daniel', 'Andrei'); --21

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. At the top, there is a code editor window with two lines of SQL: line 552 is empty, and line 553 contains the query `select * from angajat;`. Below the code editor is a toolbar with icons for Script Output, Query Result, Query Result 1, and Query Result 2. The "Query Result" tab is selected. A status bar at the bottom indicates "All Rows Fetched: 21 in 0.007 seconds". The main area displays a grid of 21 rows of data from the `angajat` table, with columns labeled `ID_ANGAJAT`, `ID_RESTAURANT`, `ID_SEF`, `NUME`, `PRENUME`, and `DATA_ANGAJARE`.

ID_ANGAJAT	ID_RESTAURANT	ID_SEF	NUME	PRENUME	DATA_ANGAJARE
1	1	1 (null)	Popescu	Robertto	30-JAN-25
2	2	2	1 Escobar	Ricardo	15-FEB-21
3	3	4	1 Marinescu	Teodora	10-JAN-20
4	4	3	2 Marinescu	Petre	15-JAN-21
5	5	2	2 Manole	Alexandru	20-JUN-19
6	6	1	1 Voicu	Andrei	30-JAN-25
7	7	3	3 Radoi	Raisa	15-MAY-20
8	8	4	3 Stan	Mihnea	01-JAN-20
9	9	5	1 Filip	Mihnea	21-SEP-20
10	10	1	1 Peste	Florin	25-JUL-20
11	11	2	2 Raducanu	Sorin	01-APR-21
12	12	3	3 Bida	Marian	25-FEB-21
13	13	4	4 Fredo	Magiore	01-MAY-21
14	14	5	4 Cercel	Florin	02-SEP-20
15	15	4	4 Stanescu	Gigel	02-JUL-20
16	16	2	5 Dociu	Mihai	17-AUG-20
17	17	1	1 Dumitrescu	Florin	04-JAN-25
18	18	3	3 Sociu	Razvan	29-DEC-20
19	19	5	6 Scarlatescu	Catalin	01-JAN-25
20	20	2	6 Bontea	Sorin	02-JAN-25
21	21	1	1 Daniel	Andrei	30-JAN-25

```
-- PENTRU TABELUL PREPARARE --
-- durata este masurata in minute --
insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (1, 1, 17, 60); --1

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (1, 2, 9, 45); --2

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (2, 4, 19, 35); --3

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (3, 4, 19, 25); --4

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (4, 4, 19, 120); --5

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (5, 4, 17, 75); --6

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (3, 3, 20, 70); --7

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (3, 5, 9, 16); --8

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (1, 2, 17, 130); --9

insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (4, 2, 18, 240); --10
```

Print-Screen:

The screenshot shows a MySQL Workbench interface. In the top-left, the number '644' is displayed. To its right is a text input field containing the SQL query: `select * from preparare;`. Below the query is a 'Query Result' tab. A status bar at the bottom indicates 'All Rows Fetched: 10 in 0.004 seconds'. The main area displays a table with 10 rows of data.

ID_PRODUS	ID_COMANDA	ID_ANGAJAT	DURATA
1	1	17	60
2	1	9	45
3	2	19	35
4	3	19	25
5	4	19	120
6	5	17	75
7	3	20	70
8	3	9	16
9	1	17	130
10	4	18	240

```
-- PENTRU TABELUL BUCATAR --
insert into bucatar (id_angajat, nr_stele)
values (9, 1); --1

insert into bucatar (id_angajat, nr_stele)
values (17, 3); --2

insert into bucatar (id_angajat, nr_stele)
values (18, 4); --3

insert into bucatar (id_angajat, nr_stele)
values (19, 5); --4

insert into bucatar (id_angajat, nr_stele)
values (20, 5); --5

insert into bucatar (id_angajat, nr_stele)
values (21, 2); --6
```

Print-Screen:

```
400 | select * from bucatar;
401 |
402 | -- PENTRU TABELUL CASIER --
```

ID_ANGAJAT	NR_STELE
1	9
2	17
3	18
4	19
5	20
6	21

```
-- PENTRU TABELUL CHELNER --
insert into chelner (id_angajat, ani_experienta)
values (4, 2); --1

insert into chelner (id_angajat, ani_experienta)
values (7, 5); --2
```

```

insert into chelner (id_angajat, ani_experienta)
values (11, 1); --3

insert into chelner (id_angajat, ani_experienta)
values (14, 10); --4

insert into chelner (id_angajat, ani_experienta)
values (15, 7); --5

```

Print-Screen:

680 | **select * from chelner;**

ID_ANGAJAT	ANI_EXPERIENTA
1	4
2	7
3	11
4	14
5	15

```

-- PENTRU TABELUL MANAGER --
insert into manager (id_angajat)
values (1); --1

insert into manager (id_angajat)
values (2); --2

insert into manager (id_angajat)
values (6); --3

insert into manager (id_angajat)
values (10); --4

insert into manager (id_angajat)
values (13); --5

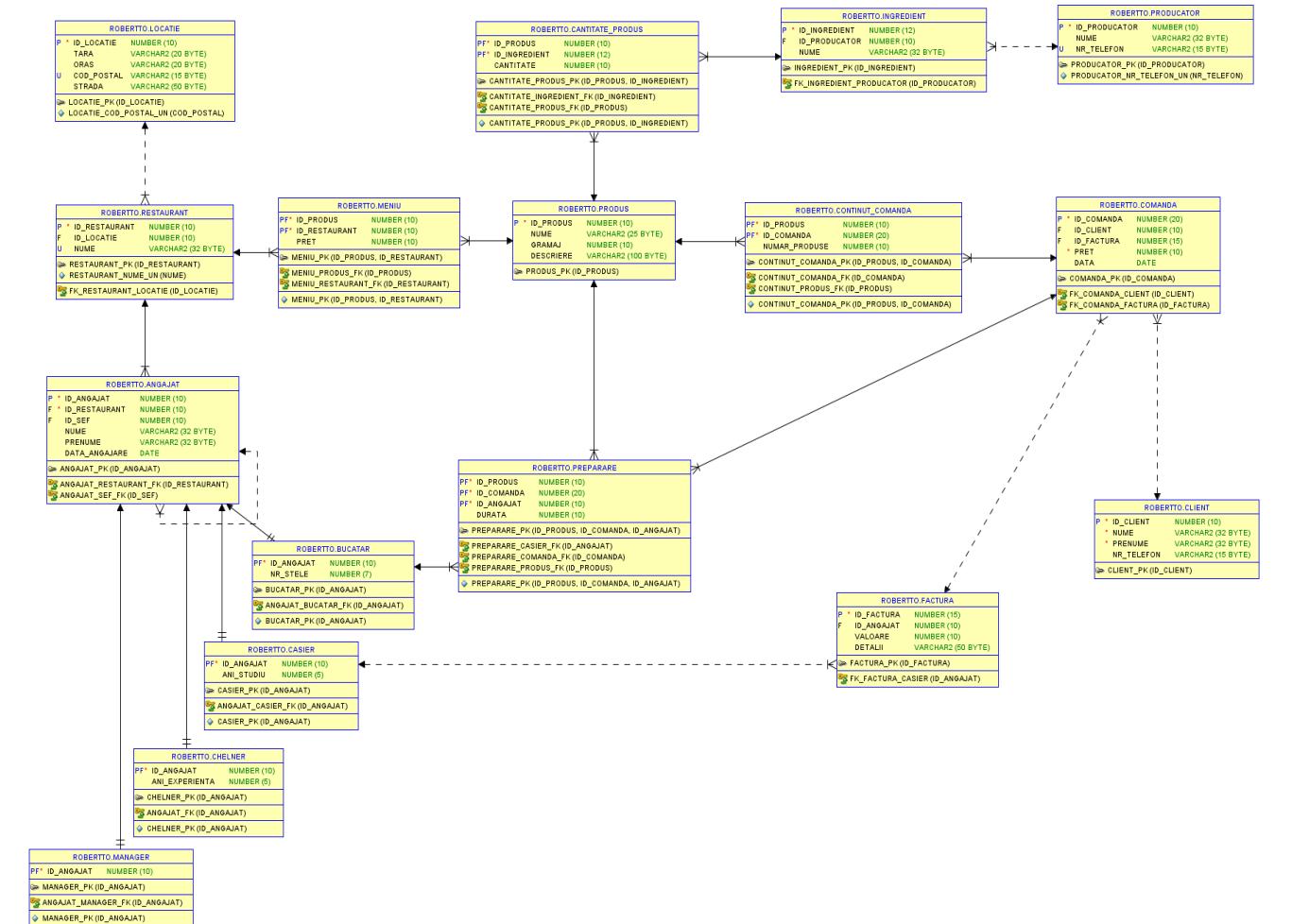
```

Print-Screen:

698 | **select * from manager;**

ID_ANGAJAT
1
2
3
4
5

6.3 Diagrama generată în SQL Developer după crearea tabelelor și inserarea datelor.



7. Crearea a 15 interogări complexe în SQL:

1) Să se afișeze numele și id-ul tuturor produselor comandate de un client vreodată.

```
SELECT DISTINCT
    cl.id_client,
    cl.numere,
    cl.prenume,
    prod.id_produs,
    prod.numere
FROM
    comanda      c,
    client        cl,
    continut_comanda cmd,
    produs       prod
WHERE
    cl.id_client = c.id_client
    AND c.id_comanda = cmd.id_comanda
    AND cmd.id_produs = prod.id_produs
ORDER BY
    id_client,
    id_produs;
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, there's a 'Worksheet' tab where the SQL query is typed. Below it, in the bottom-left, is a 'Script Output' tab. In the center, there are two 'Query Result' tabs, one labeled 'Query Result 1' and the other 'Query Result 2'. The 'Query Result 1' tab is active and displays a table with 14 rows of data. The table has columns: ID_CLIENT, NUME, PRENUME, ID_PRODUS, and NUME_1. The data is as follows:

ID_CLIENT	NUME	PRENUME	ID_PRODUS	NUME_1
1	Ionescu	Marian	1	Spaghete
2	Ionescu	Marian	2	Pizza
3	Ionescu	Marian	3	Paste bolognese
4	Ionescu	Marian	4	Sarmale
5	Ionescu	Marian	5	Ciorba de post
6	Dumitrescu	Mircea	3	Paste bolognese
7	Gheorghe	Sebastian	1	Spaghete
8	Gheorghe	Sebastian	2	Pizza
9	Gheorghe	Sebastian	5	Ciorba de post
10	Salam	Florin	1	Spaghete
11	Salam	Florin	3	Paste bolognese
12	Salam	Florin	5	Ciorba de post
13	Biju	Costel	3	Paste bolognese
14	Biju	Costel	4	Sarmale

2) Pentru fiecare bucătar angajat în anul curent, afişați profitul mediu pe care îl poate aduce dacă ar vinde din fiecare mâncare pe care știe să o prepare exact o bucată.

```
SELECT
    a.id_angajat,
    lower(a.nume),
    lower(a.prenume),
    r.id_restaurant,
    upper(r.nume),
    AVG(m.pret) "castig mediu"
FROM
    angajat      a,
    bucatar      b,
    preparare    prep,
    produs       prod,
    meniu        m,
    restaurant   r
WHERE
    a.id_restaurant = r.id_restaurant
    AND a.id_angajat = b.id_angajat
    AND b.id_angajat = prep.id_angajat
    AND prod.id_produs = prep.id_produs
    AND prod.id_produs = m.id_produs
    AND m.id_restaurant = r.id_restaurant
    AND to_char(a.data_angajare, 'yyyy') LIKE to_char(sysdate,
'yyyy')
GROUP BY (
    a.id_angajat,
    a.nume,
    a.prenume,
    r.id_restaurant,
    r.nume
);
```

Print-Screen:

3) Afişați pentru fiecare produs, id-ul bucătarului care îl prepară și câte stele are bucătarul sau mesaj dacă nu e preparat de nimeni, dacă e preparat de mai mulți, ii afişați pe toți.

```
-- Nesicronizata
with myProd as (
    select distinct prod.ID_PRODUS, prod.NUME, prep.ID_ANGAJAT
    from produs prod,
         PREPARARE prep
    where prod.ID_PRODUS = prep.ID_PRODUS(+) )
select myProd.ID_PRODUS,
       myProd.NUME,
       decode(nvl(om.ID_ANGAJAT, -1), -1, 'nu e preparat de nimeni',
om.id_angajat) preparator,
       om.NR_STELE,
       om.ID_RESTAURANT
from myProd,
     (select a.ID_ANGAJAT, a.NUME, a.PRENUME, b.NR_STELE,
r.ID_RESTAURANT
      from ANGAJAT a,
           BUCATAR b,
           RESTAURANT r
     where a.ID_ANGAJAT = b.ID_ANGAJAT
       and a.ID_RESTAURANT = r.ID_RESTAURANT) om
where myProd.ID_ANGAJAT = om.ID_ANGAJAT(+);
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface with the following details:

- Script Output:** Contains the SQL query code.
- Query Result:** Shows the results of the query execution.
- Table Results:** Displays the output data in a grid format.

Table Results Data:

ID_PRODUS	NUME	PREPATOR	NR_STELE	ID_RESTAURANT
1	1 Spaghete	9	1	5
2	3 Paste bolognese	9	1	5
3	1 Spaghete	17	3	1
4	5 Ciocan de post	17	3	1
5	4 Sarmale	18	4	3
6	2 Pizza	19	5	5
7	3 Paste bolognese	19	5	5
8	4 Sarmale	19	5	5
9	3 Paste bolognese	20	5	2
10	6 Sushi	nu e preparat de nimeni	(null)	(null)

4) Afişaţi toți clienții cu toate comenziile lor și facturile aferente, a căror sumă cheltuită până acum depășește average-ul comenziilor cu cel puțin 3 produse.

```
-- Sincronizata
select *
from CLIENT cl,
      COMANDA cmd,
      FACTURA f
where cmd.ID_CLIENT = cl.ID_CLIENT
  and f.ID_FACTURA = cmd.ID_FACTURA
  and (select sum(cmd2.PRET)
        from COMANDA cmd2
        where cmd2.ID_CLIENT = cl.ID_CLIENT) >
(select avg(t1.pr)
  from (select cmd3.ID_COMANDA, cmd3.PRET pr
          from COMANDA cmd3,
              CONTINUT_COMANDA cnt
         where cmd3.ID_COMANDA = cnt.ID_COMANDA
         group by cmd3.ID_COMANDA, cmd3.PRET
        having count(*) >= 3) t1);
```

Print-Screen:

```
78 --- 4) Afisati toti clientii cu toate comenzile lor si facturile aferente,
79 -- a caror suma cheltuita pana acum depaseste average-ul comenziilor cu cel putin 3 produs
80 -- Sincronizata
81 select *
82 from CLIENT cl,
83      COMANDA cmd,
84      FACTURA f
85 where cmd.ID_CLIENT = cl.ID_CLIENT
86   and f.ID_FACTURA = cmd.ID_FACTURA
87   and (select sum(cmd2.PRET)
88         from COMANDA cmd2
89         where cmd2.ID_CLIENT = cl.ID_CLIENT) >
90 (select avg(t1.pr)
91   from (select cmd3.ID_COMANDA, cmd3.PRET pr
92          from COMANDA cmd3,
93              CONTINUT_COMANDA cnt
94             where cmd3.ID_COMANDA = cnt.ID_COMANDA
95             group by cmd3.ID_COMANDA, cmd3.PRET
96            having count(*) >= 3) t1);
97
```

ID_CLIENT	NUME	PRENUME	NR_TELEFON	ID_COMANDA	ID_CLIENT_1	ID_FACTURA	PRET	DATA	ID_FACTURA_1	ID_ANGAJAT	VALOARE	DETALII
1	Ionescu	Marian	0721666212	1	1	4	50	30-JAN-25	4	3	70	CASH
2	Ionescu	Marian	0721666212	6	1	4	100	30-JAN-25	4	3	70	CASH
3	Ionescu	Marian	0721666212	7	1	1	50	30-JAN-25	1	16	100	CASH
4	Ionescu	Marian	0721666212	8	1	2	30	30-JAN-25	2	12	250	CARD
5	Ionescu	Marian	0721666212	9	1	3	70	30-JAN-25	3	8	50	CASH
6	Ionescu	Marian	0721666212	10	1	4	20	30-JAN-25	4	3	70	CASH
7	Gheorghe	Sebastian	0744573419	2	3	5	130	30-JAN-25	5	5	150	CARD
8	Salam	Florin	0210116666	4	4	2	230	30-JAN-25	2	12	250	CARD

5) Pentru fiecare pereche de angajați, afișați dacă aceștia se cunosc de cel puțin un an.

```
select case
    when months_between(a1.DATA_ANGAJARE, a2.DATA_ANGAJARE) >=
12 then 'Angajatii ' || a1.ID_ANGAJAT || ' si ' ||
a2.ID_ANGAJAT ||
' se cunosc de peste un an'
    else 'Angajatii ' || a1.ID_ANGAJAT || ' si ' ||
a2.ID_ANGAJAT || ' se cunosc de mai putin de un an' end
from ANGAJAT a1,
ANGAJAT a2
where a1.ID_ANGAJAT < a2.ID_ANGAJAT;
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab is active, displaying the SQL query. The 'Query Result' tab shows the output of the query, which consists of 43 rows of text. Each row contains two parts: an identifier (e.g., 'Angajatii 1 si 2') followed by a statement ('se cunosc de peste un an'). The output is timestamped at 'Fetched 100 rows in 0.011 seconds'.

Row Number	Output Content
1	Angajatii 1 si 2 se cunosc de peste un an
2	Angajatii 1 si 3 se cunosc de peste un an
3	Angajatii 1 si 4 se cunosc de peste un an
4	Angajatii 1 si 5 se cunosc de peste un an
5	Angajatii 1 si 6 se cunosc de mai putin de un an
6	Angajatii 1 si 7 se cunosc de peste un an
7	Angajatii 1 si 8 se cunosc de peste un an
8	Angajatii 1 si 9 se cunosc de peste un an
9	Angajatii 1 si 10 se cunosc de peste un an
10	Angajatii 1 si 11 se cunosc de peste un an
11	Angajatii 1 si 12 se cunosc de peste un an
12	Angajatii 1 si 13 se cunosc de peste un an
13	Angajatii 1 si 14 se cunosc de peste un an
14	Angajatii 1 si 15 se cunosc de peste un an
15	Angajatii 1 si 16 se cunosc de peste un an
16	Angajatii 1 si 17 se cunosc de mai putin de un an
17	Angajatii 1 si 18 se cunosc de peste un an
18	Angajatii 1 si 19 se cunosc de mai putin de un an
19	Angajatii 1 si 20 se cunosc de mai putin de un an
20	Angajatii 1 si 21 se cunosc de mai putin de un an
21	Angajatii 2 si 3 se cunosc de peste un an
22	Angajatii 2 si 4 se cunosc de mai putin de un an
23	Angajatii 2 si 5 se cunosc de peste un an
24	Angajatii 2 si 6 se cunosc de mai putin de un an
25	Angajatii 2 si 7 se cunosc de mai putin de un an
26	Angajatii 2 si 8 se cunosc de peste un an
27	Angajatii 2 si 9 se cunosc de mai putin de un an
28	Angajatii 2 si 10 se cunosc de mai putin de un an
29	Angajatii 2 si 11 se cunosc de mai putin de un an
30	Angajatii 2 si 12 se cunosc de mai putin de un an
31	Angajatii 2 si 13 se cunosc de mai putin de un an
32	Angajatii 2 si 14 se cunosc de mai putin de un an
33	Angajatii 2 si 15 se cunosc de mai putin de un an
34	Angajatii 2 si 16 se cunosc de mai putin de un an
35	Angajatii 2 si 17 se cunosc de mai putin de un an
36	Angajatii 2 si 18 se cunosc de mai putin de un an
37	Angajatii 2 si 19 se cunosc de mai putin de un an
38	Angajatii 2 si 20 se cunosc de mai putin de un an
39	Angajatii 2 si 21 se cunosc de mai putin de un an
40	Angajatii 3 si 4 se cunosc de mai putin de un an
41	Angajatii 3 si 5 se cunosc de mai putin de un an
42	Angajatii 3 si 6 se cunosc de mai putin de un an
43	Angajatii 3 si 7 se cunosc de mai putin de un an

6) Afisați pentru toate produsele din sistem, la ce restaurant se servesc, sau null dacă nu e servit nicăieri.

```
select p.ID_PRODUS, p.NUME, m.ID_RESTAURANT, r.NUME, l.TARA, l.ORAS,
l.STRADA
from meniu m, PRODUS p, RESTAURANT r, locatie l
where m.ID_PRODUS(+) = p.ID_PRODUS
and m.ID_RESTAURANT = r.ID_RESTAURANT(+)
and r.ID_RESTAURANT = l.ID_LOCATIE(+) --outer join
```

Print-Screen:

The screenshot shows a SQL query editor with the following details:

- Query Text:**

```
109 -- 6)
110 -- Afisati pentru toate produsele din sistem, la ce restaurant se servesc, sau null daca nu
111 select p.ID_PRODUS, p.NUME, m.ID_RESTAURANT, r.NUME, l.TARA, l.ORAS, l.STRADA
112 from meniu m, PRODUS p, RESTAURANT r, locatie l
113 where m.ID_PRODUS(+) = p.ID_PRODUS
114 and m.ID_RESTAURANT = r.ID_RESTAURANT(+)
115 and r.ID_RESTAURANT = l.ID_LOCATIE(+) --outer join
116
117 -- Afisati produsele care nu au locatii in toata retea restaurantelor
```
- Execution Details:** All Rows Fetched: 12 in 0.018 seconds
- Results Table:**

ID_PRODUS	NUME	ID_RESTAURANT	NUME_1	TARA	ORAS	STRADA
1	4 Sarmale		1 Gurmandul	Romania	Bucuresti	Batista
2	5 Ciorba de post		1 Gurmandul	Romania	Bucuresti	Batista
3	3 Paste bolognese		2 Yamas	Romania	Cluj	Mihai Eminescu
4	4 Sarmale		2 Yamas	Romania	Cluj	Mihai Eminescu
5	4 Sarmale		3 Ivans	Romania	Iasi	Rediu
6	5 Ciorba de post		3 Ivans	Romania	Iasi	Rediu
7	1 Spaghete		4 Savanna	Italia	Milano	Monza
8	2 Pizza		4 Savanna	Italia	Milano	Monza
9	4 Sarmale		4 Savanna	Italia	Milano	Monza
10	2 Pizza		5 Grande appetito!	Germania	Munchen	Perhamerstrabe
11	4 Sarmale		5 Grande appetito!	Germania	Munchen	Perhamerstrabe
12	6 Sushi	(null)	(null)	(null)	(null)	(null)

Afişați produsele care se găsesc în toate restaurantele.

```
-- Folosim division
select distinct p.ID_PRODUS,p.NUME
from PRODUS p, MENIU m
where p.ID_PRODUS in (select distinct m.ID_PRODUS from MENIU m) and
m.ID_PRODUS=p.ID_PRODUS
group by p.ID_PRODUS, p.NUME
having count(p.ID_PRODUS) = (select count(r1.ID_RESTAURANT) from
RESTAURANT r1);
```

Print-Screen:

The screenshot shows a SQL query editor with the following code:

```
-- Afisati produsele care se gasesc in toate restaurantele.
-- Folosim division
select distinct p.ID_PRODUS,p.NUME
from PRODUS p, MENIU m
where p.ID_PRODUS in (select distinct m.ID_PRODUS from MENIU m) and m.ID_PRODUS=p.ID_PRODUS
group by p.ID_PRODUS, p.NUME
having count(p.ID_PRODUS) = (select count(r1.ID_RESTAURANT) from RESTAURANT r1);
-- Afisati produsele care se gasesc in toate comenzile dintr-o anumita zi.
-- Folosim Division
insert into FACTURA values(1000,3,100,'nu');
insert into COMANDA values(1000,1,1000,1000,to_date('11/07/2001 12:00:00','MM/DD/YYYY'));
```

The results of the query are displayed in a table:

ID_PRODUS	NUME
1	4 Sarmale

Below the table, the status bar indicates "All Rows Fetched: 1 in 0.007 seconds".

Afişați produsele care se găsesc în toate comenzi dintr-o anumită zi. Folosim Division.

```
insert into FACTURA values(1000,3,100,'nu');
insert into COMANDA values (1000,1,1000,1000,to_date('13-07-
2000','dd-mm-yyyy'));
insert into CONTINUT_COMANDA values (4,1000,3);

with t as(select * from CONTINUT_COMANDA cnt0, COMANDA cmd0 where
cnt0.ID_COMANDA=cmd0.ID_COMANDA)
select distinct p.ID_PRODUS, p.NUME
from PRODUS p, t t1
where p.ID_PRODUS=t1.ID_PRODUS
and to_char(t1.DATA,'dd-mm-yyyy')='13-07-2000'
and p.ID_PRODUS in (select distinct t2.ID_PRODUS from t t2
                     where to_char(t2.DATA, 'dd-mm-yyyy')='13-07-
2000')
group by p.ID_PRODUS, p.NUME
having count(p.ID_PRODUS)=(select count(*) from t t3
                           where to_char(t3.DATA, 'dd-mm-
yyyy')='13-07-2000');
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface with the following details:

- Worksheet Tab:** Labeled "Query builder".
- Code Area:** Displays the SQL query with line numbers 125 to 141. Lines 125 and 126 are comments. Lines 128 to 141 form the main query.
- Status Bar:** Shows "All Rows Fetched: 1 in 0.038 seconds".
- Results Area:** Shows a table with one row:

ID_PRODUS	NUME
1	4 Sarmale

7) Afişaţi combinaţiile posibile între restaurante şi produse, inclusiv acelea unde restaurantul nu vinde niciun produs (coloanele de produs vor fi NULL) sau unde există produse ce nu apar în niciun meniu (coloanele de restaurant vor fi NULL).

```
SELECT r.id_restaurant,
       r.numere AS nume_restaurant,
       p.id_produs,
       p.numere AS nume_produs
  FROM restaurant r
 FULL JOIN meniu m
    ON r.id_restaurant = m.id_restaurant
 FULL JOIN produs p
    ON p.id_produs = m.id_produs
 ORDER BY r.id_restaurant, p.id_produs;
```

Print-Screen:

The screenshot shows the SQL Server Management Studio interface. The top pane displays the SQL query with line numbers 143 to 159. The bottom pane shows the results of the query execution.

ID_RESTAURANT	NUME_RESTAURANT	ID_PRODUS	NUME_PRODUS
1	1 Gurmandul	4	Sarmale
2	1 Gurmandul	5	Ciorba de post
3	2 Yamas	3	Paste bolognese
4	2 Yamas	4	Sarmale
5	3 Ivans	4	Sarmale
6	3 Ivans	5	Ciorba de post
7	4 Savanna	1	Spaghete
8	4 Savanna	2	Pizza
9	4 Savanna	4	Sarmale
10	5 Grande appetito!	2	Pizza
11	5 Grande appetito!	4	Sarmale
12	(null) (null)	6	Sushi

8) Pentru fiecare produs din restaurantele aflate într-o locație specifică (Ex România, București), să se afișeze id-ul produsului, numele produsului, descrierea, primele 5 caractere din descriere, poziția primului spațiu, numele și id-ul restaurantului, și datele despre locația fiecărui restaurant(tara, orașul și codul postal).

```

SELECT DISTINCT
    p.id_produs          AS id_produs,
    p.nume               AS nume_produs,
    p.descriere,
    SUBSTR(p.descriere, 1, 5) AS primele_5_caractere,
    INSTR(p.descriere, ' ')  AS pozitie_prim_spatiu,
    r.nume               AS nume_restaurant,
    r.id_restaurant      AS id_restaurant,
    l.tara                AS tara,
    l.oras               AS oras,
    l.cod_postal         AS cod_postal
FROM produs p
JOIN meniu m
    ON p.id_produs = m.id_produs
JOIN restaurant r
    ON m.id_restaurant = r.id_restaurant
JOIN locatie l
    ON r.id_locatie = l.id_locatie
WHERE l.tara = 'Romania'
    AND l.oras = 'Bucuresti'
ORDER BY p.id_produs, r.id_restaurant;

```

Print-Screen:

```

160 -- 8)
161 -- Pentru fiecare produs din restaurantele aflate intr-o locatie specifica (Ex Romania, Bucuresti),
162 -- sa se afiseze id-ul produsului, numele produsului, descrierea, primele 5 caractere din descriere,
163 -- pozitia primului spatiu, numele si id-ul restaurantului, si datele despre locatia
164 -- fiecarui restaurant(tara, orasul si codul postal).
165
166 SELECT DISTINCT
167     p.id_produs      AS id_produs,
168     p.nume           AS nume_produs,
169     p.descriere,
170     SUBSTR(p.descriere, 1, 5) AS primele_5_caractere,
171     INSTR(p.descriere, ' ')  AS pozitie_prim_spatiu,
172     r.nume           AS nume_restaurant,
173     r.id_restaurant  AS id_restaurant,
174     l.tara            AS tara,
175     l.oras           AS oras,
176     l.cod_postal     AS cod_postal
177 FROM produs p
178 JOIN meniu m
179     ON p.id_produs = m.id_produs
180 JOIN restaurant r
181     ON m.id_restaurant = r.id_restaurant
182 JOIN locatie l
183     ON r.id_locatie = l.id_locatie
184 WHERE l.tara = 'Romania'
185     AND l.oras = 'Bucuresti'
186 ORDER BY p.id_produs, r.id_restaurant;
187

```

Script Output | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | All Rows Fetched: 2 in 0.024 seconds

ID_PRODUS	NUME_PRODUS	DESCRIERE	PRIMELE_5_CARACTERE	POZITIE_PRIM_SPATIU	NUME_RESTAURANT	ID_RESTAURANT	TARA	ORAS	COD_POSTAL
1	4 Sarmale	ca la mama acasa ca la	4 Sarmale	3	Ivans	3	Romania	Bucuresti	010051
2	5 Ciocan de post fara carne	fara	5 Ciocan de post fara carne	5	Ivans	3	Romania	Bucuresti	010051

9) Pentru fiecare restaurant din România sau Germania, să se afișeze prețul minim, prețul maxim din meniu, dar și diferența de preț între prețul minim și prețul maxim din meniu.

```
SELECT r.id_restaurant,
       r.nume          AS nume_restaurant,
       l.tara          AS tara,
       MIN(m.pret)     AS pret_minim_meniu,
       MAX(m.pret)     AS pret_maxim_meniu,
       MAX(m.pret) - MIN(m.pret) AS diferența_preturi_meniu
  FROM   restaurant r
  JOIN   meniu m ON r.id_restaurant = m.id_restaurant
  JOIN   locatie l ON r.id_locatie = l.id_locatie
 WHERE  LOWER(l.tara) = 'romania' or LOWER(l.tara) = 'germania'
 GROUP BY r.id_restaurant, r.nume, l.tara
 ORDER BY r.id_restaurant;
```

Print-Screen:

The screenshot shows a SQL query editor interface. The query itself is identical to the one provided above, with minor line number differences (188-205). The results pane displays a table with four rows of data, corresponding to the four restaurants in the database.

ID_RESTAURANT	NUME_RESTAURANT	TARA	PRET_MINIM_MENUU	PRET_MAXIM_MENUU	DIFERENTA_PRETURI_MENUU
1	Gurmandul	Romania	20	33	13
2	Yamas	Romania	15	27	12
3	Ivans	Romania	12	15	3
4	Savanna	Germania	30	50	20

10) Pentru fiecare angajat care este casier, să se afișeze id-ul acestuia, nume, prenume și valoarea totală încasată de pe urma facturilor emise de acesta.

```
SELECT c.id_angajat,
       a.nume,
       a.prenume,
       sub.total_facturi
  FROM   casier c
 JOIN   angajat a ON c.id_angajat = a.id_angajat
 JOIN  (
        SELECT f.id_angajat,
               SUM(f.valoare) AS total_facturi
          FROM factura f
         GROUP BY f.id_angajat
      ) sub ON c.id_angajat = sub.id_angajat
 ORDER BY c.id_angajat;
```

Print-Screen:

```
206 -- 10)
207 -- Pentru fiecare angajat care este casier, sa se afiseze id-ul acestuia, nume, prenume si
208 -- valoarea totala incasata de pe urma facturilor emise de acesta.
209
210 SELECT c.id_angajat,
211       a.nume,
212       a.prenume,
213       sub.total_facturi
214  FROM   casier c
215 JOIN   angajat a ON c.id_angajat = a.id_angajat
216 JOIN  (
217        SELECT f.id_angajat,
218               SUM(f.valoare) AS total_facturi
219          FROM factura f
220         GROUP BY f.id_angajat
221      ) sub ON c.id_angajat = sub.id_angajat
222 ORDER BY c.id_angajat;
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | Query Result 6 x

SQL | All Rows Fetched: 5 in 0.013 seconds

ID_ANGAJAT	NUME	PRENUME	TOTAL_FACTURI	
1	3	Marinescu	Teodora	170
2	5	Manole	Alexandru	150
3	8	Stan	Mihnea	50
4	12	Bida	Marian	250
5	16	Dociu	Mihai	100

11) Pentru fiecare bucătar angajat în luna curentă, afișați data la care i-ar expira contractul de muncă, știind că fiecare angajat are un contract semnat pe 12 luni de la data angajării.

```
SELECT a.id_angajat,
       a.nume,
       a.prenume,
       a.data_angajare,
       ADD_MONTHS(a.data_angajare, 12) AS expirare_contract_munca
  FROM   angajat a
 JOIN bucatar b ON a.id_angajat = b.id_angajat
 WHERE  EXTRACT(MONTH FROM a.data_angajare) = EXTRACT(MONTH FROM
              SYSDATE)
        AND EXTRACT(YEAR FROM a.data_angajare) = EXTRACT(YEAR FROM
              SYSDATE)
 ORDER BY a.id_angajat;
```

Print-Screen:

```
224 -- 11)
225 -- Pentru fiecare bucătar angajat în luna curentă, afișați data la care
226 -- i-ar expira contractul de muncă, știind că fiecare angajat
227 -- are un contract semnat pe 12 luni de la data angajării
228
229 SELECT a.id_angajat,
230       a.nume,
231       a.prenume,
232       a.data_angajare,
233       ADD_MONTHS(a.data_angajare, 12) AS expirare_contract_munca
234  FROM   angajat a
235 JOIN bucatar b ON a.id_angajat = b.id_angajat
236 WHERE  EXTRACT(MONTH FROM a.data_angajare) = EXTRACT(MONTH FROM SYSDATE)
237       AND EXTRACT(YEAR FROM a.data_angajare) = EXTRACT(YEAR FROM SYSDATE)
238 ORDER BY a.id_angajat;
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | Query Result 6 x

SQL | All Rows Fetched: 4 in 0.009 seconds

ID_ANGAJAT	NUME	PRENUME	DATA_ANGAJARE	EXPIRARE_CONTRACT_MUNCA
1	17 Dumitrescu	Florin	04-JAN-25	04-JAN-26
2	19 Scarlatescu	Catalin	01-JAN-25	01-JAN-26
3	20 Bontea	Sorin	02-JAN-25	02-JAN-26
4	21 Daniel	Andrei	30-JAN-25	30-JAN-26

12) Să se afișeze lista completă de produse din meniu, chiar dacă apare un produs în meniu care nu mai este pe stoc (nu mai există înregistrarea în tabelul produs), împreună cu informațiile despre restaurant (id și numele restaurantului) și categoria de preț pentru fiecare produs (Sub 50 lei / Exact 50 lei / Peste 50 lei).

```

SELECT r.id_restaurant,
       r.numere
          AS nume_restaurant,
       m.id_produs,
       p.nume
          AS nume_produs,
       m.pret,
       DECODE(SIGN(m.pret - 50),
              -1, 'Sub 50 lei',
              0, 'Exact 50 lei',
              1, 'Peste 50 lei') AS categorie_pret
FROM produs p
RIGHT JOIN meniu m
      ON p.id_produs = m.id_produs
JOIN restaurant r
      ON m.id_restaurant = r.id_restaurant
ORDER BY r.id_restaurant, m.id_produs;

```

Print-Screen:

```

240 -- 12)
241 -- Sa se afisze lista completa de produse din meniu,
242 -- chiar daca apare un produs in meniu care nu mai este pe stoc
243 -- (nu mai exista inregistrarea in tabelul produs), impreuna cu informatiile despre
244 -- restaurant (id si numele restaurantului) si categoria de pret pentru fiecare produs
245 -- (Sub 50 lei / Exact 50 lei / Peste 50 lei).
246
247 SELECT r.id_restaurant,
248       r.numere
          AS nume_restaurant,
249       m.id_produs,
250       p.nume
          AS nume_produs,
251       m.pret,
252       DECODE(SIGN(m.pret - 50),
253              -1, 'Sub 50 lei',
254              0, 'Exact 50 lei',
255              1, 'Peste 50 lei') AS categorie_pret
256 FROM produs p
257 RIGHT JOIN meniu m
258      ON p.id_produs = m.id_produs
259 JOIN restaurant r
260      ON m.id_restaurant = r.id_restaurant
261 ORDER BY r.id_restaurant, m.id_produs;
262

```

ID_RESTAURANT	NUME_RESTAURANT	ID_PRODUS	NUME_PRODUS	PRET	CATEGORIE_PRET
1	1 Gurmandul	4	Sarmale	33	Sub 50 lei
2	1 Gurmandul	5	Ciorba de post	20	Sub 50 lei
3	2 Yamas	3	Paste bolognese	27	Sub 50 lei
4	2 Yamas	4	Sarmale	15	Sub 50 lei
5	3 Ivans	4	Sarmale	15	Sub 50 lei
6	3 Ivans	5	Ciorba de post	12	Sub 50 lei
7	4 Savanna	1	Spaghete	35	Sub 50 lei
8	4 Savanna	2	Pizza	50	Exact 50 lei
9	4 Savanna	4	Sarmale	30	Sub 50 lei
10	5 Grande appetito!	2	Pizza	60	Peste 50 lei
11	5 Grande appetito!	4	Sarmale	13	Sub 50 lei

13) Să se afișeze angajații care sunt și casieri și chelneri în același timp.

```
INSERT INTO chelner (id_angajat, ani_experienta)
VALUES (12, 3);

SELECT a.id_angajat,
       a.nume,
       a.prenume
  FROM angajat a
 WHERE a.id_angajat IN (
      SELECT id_angajat FROM casier
      INTERSECT
      SELECT id_angajat FROM chelner
) ;
```

Print-Screen:

The screenshot shows a SQL editor window with the following content:

```
-- 263 -- 13)
-- Sa se afiseze angajatii care sunt si casieri si chelneri in acelasi timp.
264 INSERT INTO chelner (id_angajat, ani_experienta)
265 VALUES (12, 3);
266
267
268
269 SELECT a.id_angajat,
270       a.nume,
271       a.prenume
272  FROM angajat a
273 WHERE a.id_angajat IN (
274     SELECT id_angajat FROM casier
275     INTERSECT
276     SELECT id_angajat FROM chelner
277 ) ;
```

Below the code, the status bar indicates "All Rows Fetched: 1 in 0.011 seconds". The results pane displays a single row:

ID_ANGAJAT	NUME	PRENUME
1	12	Bida Marian

14) Dacă cel mai comandat produs este pizza atunci să se afișeze null, iar dacă există mai multe produse cu același număr maxim de comenzi să se returneze din nou null iar toate celelalte cazuri să se returneze numele produsului cu cel mai mare număr de comenzi.

```

WITH pr AS (
    SELECT LOWER(numé) AS numé, COUNT(*) AS nr
    FROM PRODUS
    GROUP BY LOWER(numé)
),
ans AS (
    SELECT pr.numé
    FROM pr
    WHERE pr.nr = (SELECT MAX(pr2.nr) FROM pr pr2)
)
SELECT NULLIF(
    CASE (SELECT COUNT(*) FROM ans)
        WHEN 1 THEN (SELECT * FROM ans)
        ELSE NULL
    END, 'pizza')
FROM dual;

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface with the following details:

- Script Output:** Shows the SQL code for question 14.
- Query Result 1:** Shows the result of the query: 1 (null).
- Status Bar:** Shows "All Rows Fetched: 1 in 0.009 seconds".
- Bottom Bar:** Shows icons for Script, Output, Log, SQL, and Help, along with the message "NULLIF(CASE(SELECTCOUNT(*)FROMANS)WHEN1THEN(SELECT*FROMANS)ELSENULLEND,PIZZA)".

```

280 -- 14)
281 -- Daca cel mai comandat produs este pizza atunci sa se afiseze null, iar daca
282 -- exista mai multe produse cu acelasi numar maxim de comenzi sa se returneze din nou null
283 -- iar toate celelalte cazuri sa se returneze numele produsului cu cel mai mare numar de comenzi.
284
285 WITH pr AS (
286     SELECT LOWER(numé) AS numé, COUNT(*) AS nr
287     FROM PRODUS
288     GROUP BY LOWER(numé)
289 ),
290 ans AS (
291     SELECT pr.numé
292     FROM pr
293     WHERE pr.nr = (SELECT MAX(pr2.nr) FROM pr pr2)
294 )
295 SELECT NULLIF(
296     CASE (SELECT COUNT(*) FROM ans)
297         WHEN 1 THEN (SELECT * FROM ans)
298         ELSE NULL
299     END, 'pizza')
300 FROM dual;
301
302 -- 15)

```

15) Să se afișeze pentru fiecare angajat id-ul, numele, prenumele, id-ul șefului dacă există nivelul ierarhic (nivel 1 = ceo firmei, nivel 2 sunt subordonații direcți ai ceo-lui, nivel 3 sunt sub-subordonații etc.) și drumul complet de la CEO până la angajatul curent.

```

SELECT
    a.id_angajat,
    a.numere,
    a.prenume,
    a.id_sef,
    LEVEL AS nivel_ierarhie,
    SYS_CONNECT_BY_PATH(a.numere, ' -> ') AS path_angajati
FROM angajat a
-- Angajatul fara sef (id_sef IS NULL) este radacina ierarhiei
START WITH a.id_sef IS NULL
-- Definim relatia parinte-copil: parintele este randul anterior (PRIOR)
-- iar copilul are id_sef = id_angajat al parintelui
CONNECT BY PRIOR a.id_angajat = a.id_sef
ORDER BY a.id_angajat;

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface with the following details:

- Script Output:** Shows the executed SQL code from line 302 to 321, including comments and the final query.
- Results:** A table titled "Exercitiul 8" showing the results of the query. The columns are: ID_ANGAJAT, NUME, PRENUME, ID_SEF, NIVEL_IERARHIE, and PATH_ANGAJATI.
- Data:** The table contains 21 rows of data, corresponding to the employees listed in the script output.

ID_ANGAJAT	NUME	PRENUME	ID_SEF	NIVEL_IERARHIE	PATH_ANGAJATI
1	Popescu	Robertto	(null)	1	1 -> Popescu
2	Escobar	Ricardo	1	2	2 -> Popescu -> Escobar
3	Marinescu	Teodora	1	2	2 -> Popescu -> Marinescu
4	Marinescu	Petre	2	3	3 -> Popescu -> Escobar -> Marinescu
5	Manole	Alexandru	2	3	3 -> Popescu -> Escobar -> Manole
6	Voicu	Andrei	1	2	2 -> Popescu -> Voicu
7	Radoi	Raisa	3	3	3 -> Popescu -> Marinescu -> Radoi
8	Stan	Mihnea	3	3	3 -> Popescu -> Marinescu -> Stan
9	Filip	Mihnea	1	2	2 -> Popescu -> Filip
10	Peste	Florin	1	2	2 -> Popescu -> Peste
11	Raducanu	Sorin	2	3	3 -> Popescu -> Escobar -> Raducanu
12	Bida	Marian	3	3	3 -> Popescu -> Marinescu -> Bida
13	Fredo	Magiore	4	4	4 -> Popescu -> Escobar -> Marinescu -> Fredo
14	Cercel	Florin	4	4	4 -> Popescu -> Escobar -> Marinescu -> Cercel
15	Stanescu	Gigel	4	4	4 -> Popescu -> Escobar -> Marinescu -> Stanescu
16	Dociu	Mihai	5	4	4 -> Popescu -> Escobar -> Manole -> Dociu
17	Dumitrescu	Florin	1	2	2 -> Popescu -> Dumitrescu
18	Sociu	Razvan	3	3	3 -> Popescu -> Marinescu -> Sociu
19	Scarlatescu	Catalin	6	3	3 -> Popescu -> Voicu -> Scarlatescu
20	Bontea	Sorin	6	3	3 -> Popescu -> Voicu -> Bontea
21	Daniel	Andrei	1	2	2 -> Popescu -> Daniel

8. Crearea tabelului MESAJE pentru excepții:

```
--- CREAREA TABELULUI MESAJE ---
CREATE TABLE MESAJE
(
    cod_mesaj      NUMBER,
    mesaj          VARCHAR2(255),
    tip_mesaj      VARCHAR2(1),
    creat_de       VARCHAR2(40) NOT NULL,
    creat_la       DATE        NOT NULL,
    CONSTRAINT pk_message PRIMARY KEY (cod_mesaj),
    CHECK (tip_mesaj = 'E' OR tip_mesaj = 'W' OR tip_mesaj = 'I')
);

-- PENTRU TABELUL MESAJE --
create sequence cod_mesaj
start with 1
increment by 1
minvalue 0
maxvalue 9999
nocycle;
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface with the 'Relational_1 (Untitled_1)' connection selected. The 'MESAJE' table is currently open. The 'Columns' tab is active, displaying the following column information:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1 COD_MESAJ	NUMBER	No	(null)	1	(null)
2 MESAJ	VARCHAR2(255 BYTE)	Yes	(null)	2	(null)
3 TIP_MESAJ	VARCHAR2(1 BYTE)	Yes	(null)	3	(null)
4 CREAT_DE	VARCHAR2(40 BYTE)	No	(null)	4	(null)
5 CREAT_LA	DATE	No	(null)	5	(null)

9. PL / SQL:

9.1 Subprogram stocat independent (inclusiv apelare) care utilizează toate cele 3 tipuri de colecții învățate.

Produsele preferate pentru un client citit de la tastatură și produsele cele mai puțin comandate vreodată de acesta. Se afișează numele și prenumele clientului, produsele preferate și de câte ori le-a comandat. Se afișează numele și prenumele clientului, produsele cele mai puțin comandate vreodată și de câte ori le-a comandat.

Aici voi simula cu ajutorul unui tabel indexat, un vector de frecvență, iar return type-ul funcției va fi un obiect format dintr-un tabel imbricat și un vector varray.

```
CREATE OR REPLACE TYPE tab_imb IS
    TABLE OF NUMBER(10);
/

CREATE OR REPLACE TYPE t_array IS
    VARRAY(10) OF NUMBER(10);
/

CREATE OR REPLACE TYPE result_obj AS OBJECT (
    v_id_feluri_preferate tab_imb,
    varray_ids             t_array
);
/

CREATE OR REPLACE FUNCTION afis_produse_preferate (my_id_client IN
client.id_client%TYPE)
RETURN result_obj IS
    TYPE tab_ind IS TABLE OF PLS_INTEGER INDEX BY PLS_INTEGER;
    v_contor                tab_ind;
    varray_ids               t_array;
    v_index_max              PLS_INTEGER;
    v_index_min              PLS_INTEGER;
    id                       PLS_INTEGER;
    v_id_feluri_preferate   tab_imb;
    ok                      BINARY_INTEGER := 0;
    client_fara_comenzi     EXCEPTION;
    v_err_msg                VARCHAR2(4000);
BEGIN
    v_id_feluri_preferate := tab_imb();
    varray_ids := t_array();

    FOR i IN (SELECT p.id_produs, cont.numar_produse -- CURSOR
IMPLICIT
        FROM produs p, continut_comanda cont, comanda cmd
        WHERE my_id_client = cmd.id_client
        AND cont.id_produs = p.id_produs
        AND cmd.id_comanda = cont.id_comanda) LOOP BEGIN
```

```

        v_contor(i.id_produs) := v_contor(i.id_produs) +
i.numar_produse; -- simulez un vector de frecventa, unde tin de cate
ori a fost comandat produsul respectiv
        IF ok = 1 THEN
            IF v_contor(i.id_produs) > v_contor(v_index_max) THEN
                v_index_max := i.id_produs;
            END IF;
        END IF;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN -- intr-un tabel indexat daca nu
exista indexul respectiv
            -- se arunca exceptia no data found
            v_contor(i.id_produs) := i.numar_produse; -- initializam
numarul de aparitii cu 1
            IF ok = 0 THEN -- daca e primul produs selectat de cursor
                ok := 1;
                v_index_max := i.id_produs;
            END IF;
        END;
    END LOOP;
    IF ok = 0 THEN
        RAISE client_fara_comenzi;
    ELSE
        v_index_min := v_contor.FIRST;

        IF v_index_min IS NOT NULL THEN
            id := v_index_min;
            LOOP
                EXIT WHEN id IS NULL;
                IF v_contor(id) < v_contor(v_index_min) THEN
                    v_index_min := id;
                END IF;

                id := v_contor.NEXT(id);
            END LOOP;
        END IF;

        IF v_contor(v_index_max) != v_contor(v_index_min) THEN
--            DBMS_OUTPUT.PUT_LINE('v_index_max: ' || v_index_max);
--            DBMS_OUTPUT.PUT_LINE('v_contor(v_index_max): ' || 
v_contor(v_index_max));
--            DBMS_OUTPUT.PUT_LINE('v_index_min: ' || v_index_min);
--            DBMS_OUTPUT.PUT_LINE('v_contor(v_index_min): ' || 
v_contor(v_index_min));
            id := v_contor.first; -- prima valoare din tabelul
indexat (vector frecventa)
            LOOP
                EXIT WHEN id IS NULL;
                IF v_contor(id) = v_contor(v_index_max) THEN
                    v_id_feluri_preferate.extend;
                END IF;
            END LOOP;
        END IF;
    END;

```

```

        v_id_feluri_preferate(v_id_feluri_preferate.LAST)
:= id;
        END IF;

        IF v_contor(id) = v_contor(v_index_min) THEN
            IF varray_ids.COUNT < varray_ids.LIMIT THEN
                varray_ids.EXTEND;
                varray_ids(varray_ids.LAST) := id;
            END IF;
        END IF;
        id := v_contor.NEXT(id);
    END LOOP;
ELSE
    id := v_contor.first; -- prima valoare din tabelul
indexat (vector frecventa)
    LOOP
        EXIT WHEN id IS NULL;
        IF v_contor(id) = v_contor(v_index_max) THEN
            v_id_feluri_preferate.extend;
            v_id_feluri_preferate(v_id_feluri_preferate.LAST)
:= id;
        END IF;
        id := v_contor.NEXT(id);
    END LOOP;
END IF;
END IF;
RETURN result_obj(v_id_feluri_preferate, varray_ids);

EXCEPTION
    WHEN client_fara_comenzi THEN
        INSERT INTO mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
        VALUES (cod_mesaj.NEXTVAL, 'Pentru clientul dat de la
tastatura cu id ' || my_id_client || ' nu s-a gasit nicio comanda!',
'W', USER, SYSDATE);
        COMMIT;
        DBMS_OUTPUT.PUT_LINE('Acest client nu a comandat nimic in
viata lui!');
    WHEN OTHERS THEN
        v_err_msg := 'Alta eroare in afis_produse_preferate: ' ||
SQLERRM;
        INSERT INTO mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
        VALUES (cod_mesaj.NEXTVAL, 'Alta eroare in
afis_produse_preferate: ' || v_err_msg, 'E', USER, SYSDATE);
        COMMIT;
END;
/

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface during the execution of a PL/SQL procedure. The main window displays the code in the Worksheet tab, and the right pane shows the results of the DBMS_OUTPUT.PUT_LINE statements.

```
131  v_nume_produs      produs.nume%TYPE;
132  v_nume            client.nume%TYPE;
133  v_prenume         client.prenume%TYPE;
134  v_contor          PLS_INTEGER := 0;
135  BEGIN
136    v_result := afis_produse_preferate(v_id_client_citit);
137    my_tab := v_result.v_id_feluri_preferate;
138    my_varray := v_result.varray_ids;
139
140    SELECT c.nume, c.prenume INTO v_nume, v_prenume
141    FROM client c
142    WHERE c.id_client = v_id_client_citit;
143    DBMS_OUTPUT.PUT_LINE('Clientul ' || v_nume || ' ' || v_prenume || ' are urmatoarele produse favorite:');
144    id := my_tab.FIRST; -- prima valoare din tabelul indexat (vector frecventa)
145  LOOP
146    EXIT WHEN id IS NULL;
147    SELECT p.nume INTO v_nume_produs
148    FROM produs p
149    WHERE p.id_produs = my_tab(id);
150    -- afisam produsele cu numarul maxim de aparitii
151    v_contor := v_contor + 1;
152    DBMS_OUTPUT.PUT_LINE(v_contor || '.' || v_nume_produs);
153    id := my_tab.NEXT(id);
154  END LOOP;
155
156
157  IF my_varray.COUNT = 0 THEN
158    DBMS_OUTPUT.PUT_LINE('Niciun produs nu a fost comandat de mai putine ori fata de celelalte.');
159  ELSE
160    DBMS_OUTPUT.PUT_LINE('');
161    DBMS_OUTPUT.PUT_LINE('Produse cel mai putin comandate: ');
162    v_contor := 0;
163    FOR idx IN 1..my_varray.COUNT LOOP
164      SELECT p.nume
165        INTO v_nume_produs
166        FROM produs p
167        WHERE p.id_produs = my_varray(idx);
168      v_contor := v_contor + 1;
169      DBMS_OUTPUT.PUT_LINE(v_contor || '.' || v_nume_produs);

```

DBMS Output pane content:

```
Clientul Ionescu Marian are urmatoarele produse favorite:
1. Pizza

Produse cel mai putin comandate:
1. Ciorba de post
```

Script Output pane content:

```
DBMS_OUTPUT.PUT_LINE(v_contor || '.' || v_nume_produs);
END LOOP;
END IF;
END;

PL/SQL procedure successfully completed.
```

9.2 Subprogram stocat independent (inclusiv apelare) care utilizează toate cele 2 tipuri de cursoare învățate, unul dintre acestea fiind cursor parametrizat, dependent de celălalt cursor.

Să se afișeze pentru fiecare bucătar numele și prenumele acestuia, dacă a preparat vreun produs de când s-a angajat.

Dacă a preparat minim un produs se vor afișa date corespunzătoare legate de acesta (numele produsului, descrierea acestuia și cât a durat să îl prepare).

```
CREATE OR REPLACE PROCEDURE afis_durata_preparare_produs
IS
    CURSOR c_bucatari IS SELECT a.id_angajat, a.nume, a.prenume,
b.nr_stele
                           FROM angajat a
                           JOIN bucatar b ON a.id_angajat =
b.id_angajat;
    CURSOR c_prepares(p_id_angajat in angajat.id_angajat%TYPE) IS
SELECT p.id_produs, p.nume, p.descriere, prep.durata
                           FROM preparare prep
                           JOIN produs p ON
p.id_produs = prep.id_produs
                           WHERE prep.id_angajat
= p_id_angajat;
    TYPE rec IS RECORD (id_angajat.id_angajat%TYPE, dur
prepares.durata%TYPE);
    TYPE tab_ind IS TABLE OF rec INDEX BY PLS_INTEGER;
    t                      tab_ind;
    v_id_bucatar           angajat.id_angajat%TYPE;
    v_nume_bucatar          angajat.nume%TYPE;
    v_prenume_bucatar        angajat.prenume%TYPE;
    v_nr_stele_bucatar       bucatar.nr_stele%TYPE;
    v_durata_prepares        preparare.durata%TYPE;
    v_id_produs              produs.id_produs%TYPE;
    v_nume_produs             produs.nume%TYPE;
    v_descriere_produs        produs.descriere%TYPE;
BEGIN
    OPEN c_bucatari;
    LOOP
        FETCH c_bucatari INTO v_id_bucatar, v_nume_bucatar,
v_prenume_bucatar, v_nr_stele_bucatar;
        EXIT WHEN c_bucatari%NOTFOUND;

        OPEN c_prepares(v_id_bucatar);
        FETCH c_prepares INTO v_id_produs, v_nume_produs,
v_descriere_produs, v_durata_prepares;

        IF c_prepares%NOTFOUND THEN
```

```

        DBMS_OUTPUT.PUT_LINE('ATENȚIE! Bucătarul ' ||
v_nume_bucatar || ' ' || v_prenume_bucatar || ' nu a preparat
niciodată un produs!');
        DBMS_OUTPUT.NEW_LINE();
    ELSE
        DBMS_OUTPUT.PUT_LINE('Bucătarul ' || v_nume_bucatar || ' '
|| v_prenume_bucatar || ' a preparat următoarele produse:');
        WHILE c_prepare%FOUND
        LOOP
            DBMS_OUTPUT.PUT_LINE(c_prepare%ROWCOUNT || '.');
            Produsul: ' || v_nume_produs || ' (' || v_descriere_produs || ') a
durat ' || v_durata_preparare || ' minute.');
            IF t.EXISTS(v_id_produs) THEN
                IF v_durata_preparare < t(v_id_produs).dur THEN
                    t(v_id_produs) := rec(v_id_bucatar,
v_durata_preparare);
                END IF;
            ELSE
                t(v_id_produs) := rec(v_id_bucatar,
v_durata_preparare);
            END IF;
            FETCH c_prepare INTO v_id_produs, v_nume_produs,
v_descriere_produs, v_durata_preparare;
        END LOOP;
    END IF;
    CLOSE c_prepare;
END LOOP;
CLOSE c_bucatari;

FOR i IN t.FIRST..t.LAST LOOP
    SELECT nume INTO v_nume_produs
    FROM produs
    WHERE id_produs = i;

    SELECT a.nume, a.prenume INTO v_nume_bucatar,
v_prenume_bucatar
    FROM angajat a, bucatar b
    WHERE a.id_angajat = t(i).id_ang
    AND a.id_angajat = b.id_angajat;
    DBMS_OUTPUT.PUT_LINE('Bucatarul ' || v_nume_bucatar || ' ' ||
v_prenume_bucatar || ' a preparat produsul ' || v_nume_produs || ' in
' || t(i).dur || ' minute(cel mai scurt timp).');
    END LOOP;
END;
/

BEGIN
    afis_durata_preparare_produs();
END;
/

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Worksheet' tab displays a PL/SQL script named 'Exercitiul 9.sql'. The script contains several loops, IF statements, and DBMS_OUTPUT.PUT_LINE statements. On the right, the 'Dms Output' tab shows the results of the script execution, listing various clients and the products they have prepared along with their preparation times.

```
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
```

Clientul Ionescu Marian are urmatoarele produse favorite:
1. Pizza

Produse cel mai putin comandate:
1. Ciorba de post

Bucatarul Filip Mihnea a preparat urmatoarele produse:
1. Produsul: Spaghete (picante) a durat 45 minute.
2. Produsul: Paste bolognese (cu sos dulce) a durat 16 minute.
Bucatarul Dumitrescu Florin a preparat urmatoarele produse:
1. Produsul: Spaghete (picante) a durat 60 minute.
2. Produsul: Spaghete (picante) a durat 130 minute.
3. Produsul: Ciorba de post (fara carne) a durat 75 minute.
Bucatarul Sociu Razvan a preparat urmatoarele produse:
1. Produsul: Sarmale (ca la mama acasa) a durat 240 minute.
Bucatarul Scarlatescu Catalin a preparat urmatoarele produse:
1. Produsul: Pizza (dulce aromata) a durat 35 minute.
2. Produsul: Paste bolognese (cu sos dulce) a durat 25 minute.
3. Produsul: Sarmale (ca la mama acasa) a durat 120 minute.
Bucatarul Bontea Sorin a preparat urmatoarele produse:
1. Produsul: Paste bolognese (cu sos dulce) a durat 70 minute.
ATENȚIE! Bucatarul Daniel Andrei nu a preparat niciodată un produs!

Bucatarul Filip Mihnea a preparat produsul Spaghete in 45 minute(cel mai scurt timp).
Bucatarul Scarlatescu Catalin a preparat produsul Pizza in 35 minute(cel mai scurt timp).
Bucatarul Filip Mihnea a preparat produsul Paste bolognese in 16 minute(cel mai scurt timp).
Bucatarul Scarlatescu Catalin a preparat produsul Sarmale in 120 minute(cel mai scurt timp).
Bucatarul Dumitrescu Florin a preparat produsul Ciorba de post in 75 minute(cel mai scurt timp)

Procedure AFIS_DURATA_PREPARARE_PRODUS compiled
PL/SQL procedure successfully completed.

9.3 Subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite și tratarea exceptiilor care pot apărea.

Să se afișeze profitul generat de un client citit de la tastatură, de-a lungul timpului. Dacă acesta a beneficiat de reduceri de-a lungul timpului se va afișa valoarea totală a acestor reduceri. Dacă profitul adus de-a lungul timpului este unul negativ înseamnă că clientul a beneficiat de prea multe reduceri și se va afișa un mesaj corespunzător și suma (chiar dacă este negativă). De asemenea se va afișa profitul maxim oferit de un client și profitul obținut de companie până la momentul actual.

```
CREATE OR REPLACE FUNCTION profit_client (my_id_client IN
client.id_client%TYPE)
RETURN PLS_INTEGER IS
```

```

TYPE tab_ind IS TABLE OF PLS_INTEGER INDEX BY PLS_INTEGER;
v_nume_client           client.nume%TYPE;
v_prenume_client         client.prenume%TYPE;
v_nr_telefon_client     client.nr_telefon%TYPE;
v_profit_maxim_companie PLS_INTEGER := 0;
v_profit_maxim_client   PLS_INTEGER := 0;
v_benef_red              BINARY_INTEGER := 0;
v_valoare_red            PLS_INTEGER := 0;
invalid                  EXCEPTION;
profit_negativ           EXCEPTION;
v_exista                PLS_INTEGER := 0;
t                        tab_ind;
v_err_msg                VARCHAR2(4000);

BEGIN
    SELECT COUNT(*) INTO v_exista
    FROM client
    WHERE id_client = my_id_client;
    IF v_exista = 0 THEN
        RAISE invalid;
    END IF;

    SELECT nume, prenume, nr_telefon INTO v_nume_client,
    v_prenume_client, v_nr_telefon_client
    FROM client
    WHERE id_client = my_id_client;

    FOR i IN (SELECT cli.id_client, cli.nume, cli.prenume, cmd.pret,
    f.valoare, f.detalii
    FROM client cli, comanda cmd, factura f
    WHERE cli.id_client = cmd.id_client
    AND cmd.id_factura = f.id_factura) LOOP
        BEGIN
            IF i.id_client = my_id_client AND i.valoare < i.pret THEN
                v_benef_red := 1;
                v_valoare_red := v_valoare_red + (i.pret -
i.valoare);
            --
            ELSIF i.id_client = my_id_client THEN
                t(i.id_client) := t(i.id_client) + (i.valoare -
i.pret);
            END IF;
            v_profit_maxim_companie := v_profit_maxim_companie +
(i.valoare - i.pret);
            t(i.id_client) := t(i.id_client) + (i.valoare - i.pret);
        EXCEPTION -- pentru prima inserare in tabel
            WHEN NO_DATA_FOUND THEN
                insert into mesaje(cod_mesaj, mesaj, tip_mesaj,
creat_de, creat_la)
                    values (cod_mesaj.NEXTVAL, 'Prima inserare in tabelul
pentru clientul cu id: ' || i.id_client, 'I', USER, sysdate);
                commit;
            t(i.id_client) := i.valoare - i.pret;
        END;
    END LOOP;
END;

```

```

        END;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Profitul obtinut in urma clientilor de pana
acum este de ' || v_profit_maxim_companie || ' lei.');
    FOR i IN 1..t.LAST LOOP
        IF t(i) > v_profit_maxim_client THEN
            v_profit_maxim_client := t(i);
        END IF;
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('Profitul maxim adus de un client este de '
|| v_profit_maxim_client || ' lei.');
    DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul ' || my_id_client || ' '
este ' || v_nume_client || ' ' || v_prenume_client || ' cu nr de
telefon: ' || v_nr_telefon_client || '.');
    IF v_benef_red = 1 THEN
        DBMS_OUTPUT.PUT_LINE('A beneficiat de reduceri in valoare de
' || v_valoare_red || ' lei.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Nu beneficiaza de reduceri!');
    END IF;
    IF t.EXISTS(my_id_client) AND t(my_id_client) < 0 THEN
        RAISE profit_negativ;
    END IF;
    RETURN t(my_id_client);
EXCEPTION
    WHEN invalid THEN
        insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
        values (cod_mesaj.NEXTVAL, 'Client inexistent cu id=' ||
my_id_client, 'E', USER, sysdate);
        commit;
        RAISE_APPLICATION_ERROR(-20021, 'Nu a fost gasit niciun
client cu id-ul introdus!');
    WHEN profit_negativ THEN
        insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
        values (cod_mesaj.NEXTVAL, 'Profit negativ pentru clientul '
|| my_id_client, 'I', USER, sysdate);
        commit;
        RAISE_APPLICATION_ERROR(-20022, 'Acest client a generat un
profit negativ (prea multe reduceri)!');
    WHEN OTHERS THEN
        v_err_msg := 'Alta eroare la profit_client, cod Oracle: ' ||
SQLERRM;
        insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
        values (cod_mesaj.NEXTVAL, v_err_msg, 'E', USER, sysdate);
        commit;
        RAISE_APPLICATION_ERROR(-20022, 'Alta eroare!');
END;
/

```

```

DECLARE
    v_id_client_citit      client.id_client%TYPE := &client_id;
    v_rezultat            PLS_INTEGER;

BEGIN
    v_rezultat := profit_client(v_id_client_citit);
    IF v_rezultat > 0 THEN
        DBMS_OUTPUT.PUT_LINE('El a contribuit la profitul companiei
pana acum cu ' || v_rezultat || ' lei.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Acesta a beneficiat de prea multe
reduceri! Aducand un profit negativ companiei: ' || v_rezultat || '
lei.');
    END IF;
END;

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface with two main panes: a Script Editor and a Database Output pane.

Script Editor (Left Pane):

```

Window Help
localhost MESAIE Relatorie_1 (Untitled_0)
insert into
Worksheet Query Builder
16 IF v_benef_red = 1 THEN
17     DBMS_OUTPUT.PUT_LINE('A beneficiat de reduceri in valoare de ' || v_valoare_red || ' lei.');
18 ELSE
19     DBMS_OUTPUT.PUT_LINE('Nu beneficiaza de reduceri!');
20 END IF;
21 IF t.EXISTS(my_id_client) AND t(my_id_client) < 0 THEN
22     RAISE profit_negativ;
23 END IF;
24 RETURN t(my_id_client);
25 EXCEPTION
26 WHEN invalid THEN
27     insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
28         values (cod_mesaj.NEXTVAL, 'Client inexistent cu id=' || my_id_client, 'E', USER, sysdate);
29     commit;
30     RAISE_APPLICATION_ERROR(-20021, 'Nu a fost gasit niciun client cu id-ul introdus!');
31 WHEN profit_negativ THEN
32     insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
33         values (cod_mesaj.NEXTVAL, 'Profit negativ pentru clientul ' || my_id_client, 'I', USER, sysdate);
34     commit;
35     RAISE_APPLICATION_ERROR(-20022, 'Acest client a generat un profit negativ (prea multe reduceri!)');
36 WHEN OTHERS THEN
37     v_err_msg := 'Alte eroare la profit_client, cod Oracle: ' || SQLERRM;
38     insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
39         values (cod_mesaj.NEXTVAL, v_err_msg, 'E', USER, sysdate);
40     commit;
41     RAISE_APPLICATION_ERROR(-20022, 'Alta eroare!');
42 END;
43 /
44
45 DECLARE
46     v_id_client_citit      client.id_client%TYPE := &client_id;
47     v_rezultat            PLS_INTEGER;
48
49 BEGIN
50     v_rezultat := profit_client(v_id_client_citit);
51     IF v_rezultat > 0 THEN
52         DBMS_OUTPUT.PUT_LINE('El a contribuit la profitul companiei pana acum cu ' || v_rezultat || ' lei.');
53     ELSE
54         DBMS_OUTPUT.PUT_LINE('Acesta a beneficiat de prea multe reduceri! Aducand un profit negativ companiei: ' || v_rezultat || ' lei.');
55     END IF;
56 END;
57
58 -- Exercitiul 9.4

```

Database Output (Right Pane):

```

localhost Dbsn Output Buffer Size:20000
localhost
Profital obtinut in urma clientilor de pana acum este de 60 lei.
Profitul maxim adus de un client este de 30 lei.
Clientul cu id-ul 3 este Gheorghe Sebastian cu nr de telefon: 0744573419.
Nu beneficiaza de reduceri!
El a contribuit la profitul companiei pana acum cu 20 lei.

```

At the bottom of the interface, it says: //SQL procedure successfully completed.

Pentru id_client = 100;

```
345 DECLARE
346   v_id_client_citit      client.id_client%TYPE := &client_id;
347   v_rezultat              PLS_INTEGER;
348
349 BEGIN
350   v_rezultat := profit_client(v_id_client_citit);
351   IF v_rezultat > 0 THEN
352     DBMS_OUTPUT.PUT_LINE('El a contribuit la profitul companiei pana acum cu ' || v_rezultat || ' lei.');
353   ELSE
354     DBMS_OUTPUT.PUT_LINE('Acesta a beneficiat de prea multe reduceri! Aducand un profit negativ companiei: ' || v_rezultat);
355   END IF;
356 END;
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1188
1189
1190
1191
1192
1193
1194
1195
1196
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1288
1289
1290
1291
1292
1293
1294
1295
1296
1296
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1388
1389
1390
1391
1392
1393
1394
1395
1396
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1488
1489
1490
1491
1492
1493
1494
1495
1496
1496
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1588
1589
1590
1591
1592
1593
1594
1595
1596
1596
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1688
1689
1690
1691
1692
1693
1694
1695
1696
1696
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1795
1796
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1890
1891
1892
1893
1894
1895
1896
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1990
1991
1992
1993
1994
1995
1996
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2090
2091
2092
2093
2094
2095
2096
2096
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2188
2189
2190
2191
2192
2193
2194
2195
2196
2196
2197
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2288
2289
2290
2291
2292
2293
2294
2295
2296
2296
2297
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2388
2389
2390
2391
2392
2393
2394
2395
2396
2396
2397
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2488
2489
2490
2491
2492
2493
2494
2495
2496
2496
2497
2498
2499
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2588
2589
2590
2591
2592
2593
2594
2595
2596
2596
2597
2598
2599
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2688
2689
2690
2691
2692
2693
2694
2695
2696
2696
2697
2698
2699
269
```

9.4 Crearea unui Trigger LMD la nivel de comandă.

Să se creeze un Trigger care nu permite clienților să plaseze comenzi de Crăciun, 25 decembrie și de Anul Nou, 1 ianuarie; astfel încât înainte de a insera în tabelul comandă să se arunce o eroare dacă ziua este una din cele menționate anterior.

```

CREATE OR REPLACE TRIGGER trig_vacanta BEFORE
    INSERT ON comanda
DECLARE
    ziua    NUMBER := extract(DAY FROM sysdate);
    luna    NUMBER := extract(MONTH FROM sysdate);
BEGIN
    IF ( ziua = 25 AND luna = 12 ) OR ( ziua = 1 AND luna = 1 ) THEN
        raise_application_error(-20067, 'Astazi nu se pot plasa
comenzi, este zi libera!');
    END IF;
END;
/

```

Print-Screen:

```
358 -- Exercitiul 9.4
359 -- Sa se creeze un Trigger care nu permite clientilor sa plaseze comenzi de Craciun, 25 decembrie si de Anul Nou, 1 ianuarie
360 -- astfel incat inainte de a insera in tabelul comanda sa se arunce o eroare daca ziua este una din cele mentionate ant
361
362 CREATE OR REPLACE TRIGGER trig_vacanta BEFORE
363     INSERT ON comanda
364 DECLARE
365     ziua    NUMBER := extract(DAY FROM sysdate);
366     luna    NUMBER := extract(MONTH FROM sysdate);
367 BEGIN
368     IF ( ziua = 25 AND luna = 12 ) OR ( ziua = 1 AND luna = 1 ) THEN
369         raise_application_error(-20067, 'Astazi nu se pot plasa comenzi, este zi libera!');
370     END IF;
371 END;
372 /
```

Trigger TRIG_VACANTA compiled

Declanșarea acestuia:

```
insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 4, 50);
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. In the code editor, a trigger named 'trig_vacanta' is defined. It checks if the current date is December 25 or January 1. If either condition is true, it raises an application error. Otherwise, it inserts a new row into the 'comanda' table with specific values. The code is as follows:

```
361 CREATE OR REPLACE TRIGGER trig_vacanta BEFORE
362   INSERT ON comanda
363 DECLARE
364   ziua NUMBER := extract(DAY FROM sysdate);
365   luna NUMBER := extract(MONTH FROM sysdate);
366 BEGIN
367   IF ( ziua = 25 AND luna = 12 ) OR ( ziua = 1 AND luna = 1 ) THEN
368     raise_application_error(-20067, 'Astazi nu se pot plasa comenzi, este zi libera!');
369   END IF;
370 END;
371 /
372
373
374 insert into comanda (id_comanda, id_client, id_factura, pret)
375 values (id_comanda.nextval, 1, 4, 50);
376
377 -- Exercitiul 9.5
```

In the 'Script Output' tab, the message 'Trigger TRIG_VACANTA compiled' is displayed. Below it, the message '1 row inserted.' indicates that the insertion was successful.

Am putut inseră pentru că data de astăzi este 7 ianuarie. Dacă modificăm trigger-ul la data de astăzi, 30 ianuarie, se va declanșa:

The screenshot shows the Oracle SQL Developer interface. The same trigger 'trig_vacanta' is created, but this time it checks for December 30 or January 1. The code is identical to the previous one except for the date comparison in the IF statement.

```
358 /-- Exercitiul 9.4
359 -- Sa se creeze un Trigger care nu permite clientilor sa plaseze comenzi de Craciun, 25 decembrie si de Anul Nou, 1 ianuarie
360 -- astfel incat inainte de a insera in tabelul comanda sa se arunce o eroare daca ziua este una din cele mentionate ant
361
362 CREATE OR REPLACE TRIGGER trig_vacanta BEFORE
363   INSERT ON comanda
364 DECLARE
365   ziua NUMBER := extract(DAY FROM sysdate);
366   luna NUMBER := extract(MONTH FROM sysdate);
367 BEGIN
368   IF ( ziua = 30 AND luna = 1 ) OR ( ziua = 1 AND luna = 1 ) THEN
369     raise_application_error(-20067, 'Astazi nu se pot plasa comenzi, este zi libera!');
370   END IF;
371 END;
372 /
373
374 insert into comanda (id_comanda, id_client, id_factura, pret)
375 values (id_comanda.nextval, 1, 4, 50);
```

In the 'Script Output' tab, an error message is shown: 'Error starting at line : 374 in command -'. The full error log is as follows:

```
Error starting at line : 374 in command -
insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 1, 4, 50)
Error at Command Line : 375 Column : 9
Error report -
SQL Error: ORA-20067: Astazi nu se pot plasa comenzi, este zi libera!
ORA-06512: at "ROBERTTO.TRIG_VACANTA", line 6
ORA-04088: error during execution of trigger 'ROBERTTO.TRIG_VACANTA'

https://docs.oracle.com/error-help/db/ora-20067/

More Details :
https://docs.oracle.com/error-help/db/ora-20067/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04088/
```

9.5 Crearea unui Trigger LMD la nivel de linie (TRIGGER COMPOUND).

Aici voi folosi un Trigger Compound pentru a nu putea insera un produs care nu se află în niciun meniu.

```
CREATE OR REPLACE TRIGGER trig_produs_in_menui FOR
    INSERT OR UPDATE OF id_produs ON continut_comanda
COMPOUND TRIGGER
    v_id_restaurant menui.id_restaurant%TYPE;
    BEFORE STATEMENT IS BEGIN
        dbms_output.put_line('Verificare daca produs exista in
menui');
    END BEFORE STATEMENT;
    BEFORE EACH ROW IS BEGIN
        SELECT
            id_restaurant
        INTO v_id_restaurant
        FROM
            menui m
        WHERE
            m.id_produs = :new.id_produs;

        dbms_output.put_line('Aces produs se afla doar in meniul
restaurantului ' || v_id_restaurant);
    EXCEPTION
        WHEN no_data_found THEN
            raise_application_error(-20065, 'Acest produs nu exista
in niciun meniu');
        WHEN too_many_rows THEN
            dbms_output.put_line('Acest produs se afla in meniurile
mai multor restaurante');
    END BEFORE EACH ROW;
    AFTER EACH ROW IS BEGIN
        dbms_output.put_line('Produsul cu id '
                            || :new.id_produs
                            || ' in comanda');

    END AFTER EACH ROW;
    AFTER STATEMENT IS BEGIN
        dbms_output.put_line('Verificare completa. Pofta buna!');
    END AFTER STATEMENT;
END trig_produs_in_menui;
/
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. In the top-left corner, there is a green box containing the text "Print-Screen:". Below it is a code editor window displaying PL/SQL code for creating a trigger named "trig_produs_in_menu". The code includes various clauses like CREATE OR REPLACE TRIGGER, FOR INSERT OR UPDATE OF, COMPOUND TRIGGER, and several BEFORE and AFTER clauses. It also uses DBMS_OUTPUT.PUT_LINE to print messages to the screen. The code is numbered from 377 to 402. At the bottom of the code editor, there is a status bar with tabs for Script Output, Query Result, and others, followed by the message "Task completed in 0.069 seconds". Below the code editor, a message says "Trigger TRIG_PRODUS_IN_MENU compiled".

```
377 | -- Exercitiul 9.5
378 | CREATE OR REPLACE TRIGGER trig_produs_in_menu FOR
379 |   INSERT OR UPDATE OF id_produs ON continut_comanda
380 | COMPOUND TRIGGER
381 |   v_id_restaurant meniu.id_restaurant%TYPE;
382 |   BEFORE STATEMENT IS BEGIN
383 |     dbms_output.put_line('Verificare daca produs exista in meniu');
384 |   END BEFORE STATEMENT;
385 |   BEFORE EACH ROW IS BEGIN
386 |     SELECT
387 |       id_restaurant
388 |     INTO v_id_restaurant
389 |     FROM
390 |       meniu m
391 |     WHERE
392 |       m.id_produs = :new.id_produs;
393 |
394 |     dbms_output.put_line('Acest produs se afla doar in meniul restaurantului ' || v_id_restaurant);
395 |   EXCEPTION
396 |     WHEN no_data_found THEN
397 |       raise_application_error(-20065, 'Acest produs nu exista in niciun meniu');
398 |     WHEN too_many_rows THEN
399 |       dbms_output.put_line('Acest produs se afla in meniurile mai multor restaurante');
400 |   END BEFORE EACH ROW;
401 |   AFTER EACH ROW IS BEGIN
402 |     dbms_output.put_line('Produsul cu id '
```

Declansarea acestuia:

```
insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (6, 4, 2); --nu se poate insera un produs care nu apare in
niciun meniu

delete from continut_comanda
where id_produs = 6 and id_comanda = 4;

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (1, 6, 2); --in meniul carui restaurant apare produsul (doar
in restaurant 4)

delete from continut_comanda
where id_produs = 1 and id_comanda = 6;

insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (4, 3, 2); --produs care se afla in mai multe meniuri

delete from continut_comanda
where id_produs = 4 and id_comanda = 3;
```

localhost > MESAIE > Relational_1 (Untitled_1) > insert into

Query Builder | Dbsm Output | localhost | Buffer Size: 20000 |

Verificare daca produs exista in meniu

```

394     dbms_output.put_line('Aces produs se afla doar in meniu restaurantului ' || v_id_restaurant);
395  EXCEPTION
396      WHEN no_data_found THEN
397          raise_application_error(-20065, 'Acest produs nu exista in niciun meniu');
398      WHEN too_many_rows THEN
399          dbms_output.put_line('Acest produs se afla in meniurile mai multor restaurante');
400  END BEFORE EACH ROW;
401  AFTER EACH ROW IS BEGIN
402      dbms_output.put_line('Produsul cu id ' ||
403                           || :new.id_produs ||
404                           || ' in comanda');
405  END AFTER EACH ROW;
406  AFTER STATEMENT IS BEGIN
407      dbms_output.put_line('Verificare completa. Pofta buna!');
408  END AFTER STATEMENT;
409  END trig_produs_in_menui;
410 /
411
412 insert into continut_comanda (id_produs, id_comanda, numar_produse)
413 values (6, 4, 2); --nu se poate insera un produs care nu apare in niciun meniu
414
415 delete from continut_comanda
416 where id_produs = 6 and id_comanda = 4;
417
418 insert into continut_comanda (id_produs, id_comanda, numar_produse)
419 values (1, 6, 2); --in meniul carui restaurant apare produsul (doar in restaurant 4)
420
421 delete from continut_comanda
422 where id_produs = 1 and id_comanda = 6;
423

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 |

Task completed in 0.062 seconds

sqlplus rtrv0000@192.168.1.100

Error starting at line : 412 in command -
insert into continut_comanda (id_produs, id_comanda, numar_produse)
values (6, 4, 2)
Error at Command Line : 412 Column : 13
Error report -
SQL Error: ORA-20065: Acest produs nu exista in niciun meniu
ORA-06512: at "ROBERTTO.TRIG_PRODUS_IN_MENUI", line 18
ORA-04088: error during execution of trigger 'ROBERTTO.TRIG_PRODUS_IN_MENUI'
<https://docs.oracle.com/error-help/db/ora-20065/>
More Details :
<https://docs.oracle.com/error-help/db/ora-20065/>
<https://docs.oracle.com/error-help/db/ora-06512/>
<https://docs.oracle.com/error-help/db/ora-04088/>

localhost > MESAIE > Relational_1 (Untitled_1) > insert into

Query Builder | Dbsm Output | localhost | Buffer Size: 20000 |

Verificare daca produs exista in meniu

```

403     || :new.id_produs ||
404     || ' in comanda';
405  END AFTER EACH ROW;
406  AFTER STATEMENT IS BEGIN
407      dbms_output.put_line('Verificare completa. Pofta buna!');
408  END AFTER STATEMENT;
409  END trig_produs_in_menui;
410 /
411
412 insert into continut_comanda (id_produs, id_comanda, numar_produse)
413 values (6, 4, 2); --nu se poate insera un produs care nu apare in niciun meniu
414
415 delete from continut_comanda
416 where id_produs = 6 and id_comanda = 4;
417
418 insert into continut_comanda (id_produs, id_comanda, numar_produse)
419 values (1, 6, 2); --in meniul carui restaurant apare produsul (doar in restaurant 4)
420
421 delete from continut_comanda
422 where id_produs = 1 and id_comanda = 6;
423
424 insert into continut_comanda (id_produs, id_comanda, numar_produse)
425 values (4, 3, 2); --produs care se afla in mai multe meniuri
426
427 delete from continut_comanda
428 where id_produs = 4 and id_comanda = 3;
429
430 -- Un client nu poate astepta mai mult decat 300 minute pentru ca comanda sa sa fie preparata.
431 -- Daca timpul de preparare este mai mare decat 300 minute sa se arunce o eroare la inserarea in tabelul preparare.
432

```

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 |

Task completed in 0.065 seconds

sqlplus rtrv0000@192.168.1.100

Error report -
SQL Error: ORA-20065: Acest produs nu exista in niciun meniu
ORA-06512: at "ROBERTTO.TRIG_PRODUS_IN_MENUI", line 18
ORA-04088: error during execution of trigger 'ROBERTTO.TRIG_PRODUS_IN_MENUI'
<https://docs.oracle.com/error-help/db/ora-20065/>
More Details :
<https://docs.oracle.com/error-help/db/ora-20065/>
<https://docs.oracle.com/error-help/db/ora-06512/>
<https://docs.oracle.com/error-help/db/ora-04088/>

0 rows deleted.

1 row inserted.

Anshor - MESAIE Relational_1 (Untitled_1)

Qn: insert into

Worksheet Query Builder

```

403      || :new.id_produs
404      || ' in comanda');
405  END AFTER EACH ROW;
406  AFTER STATEMENT IS BEGIN
407    dbms_output.put_line('Verificare completa. Pofta bună!');
408  END AFTER STATEMENT;
409  END trig_produc_in_menu;
410 /
411
412 insert into continut_comanda (id_produc, id_comanda, numar_produc)
413 values (6, 4, 2); --nu se poate inseră un produs care nu apare în niciun meniu
414
415 delete from continut_comanda
416 where id_produc = 6 and id_comanda = 4;
417
418 insert into continut_comanda (id_produc, id_comanda, numar_produc)
419 values (1, 6, 2); --în meniul carui restaurant apare produsul (doar în restaurant 4)
420
421 delete from continut_comanda
422 where id_produc = 1 and id_comanda = 6;
423
424 Insert into continut_comanda (id_produc, id_comanda, numar_produc)
425 values (4, 3, 2); --produs care se află în mai multe meniu;
426
427 delete from continut_comanda
428 where id_produc = 4 and id_comanda = 3;
429
430 -- Un client nu poate aștepta mai mult decât 300 minute pentru ca comanda să fie preparată.
431 -- Dacă timpul de prepară este mai mare decât 300 minute să se arunce o eroare la inserarea în tabelul preparare.
432

```

Items Output

localhost

Verificare daca produs exista in meniu
Verificare daca produs exista in meniu
Acest produs se afla doar in meniul restaurantului 4
Produsul cu id 1 in comanda
Verificare completa. Pofta bună!

Verificare daca produs exista in meniu
Acest produs se afla in meniurile mai multor restaurante
Produsul cu id 4 in comanda
Verificare completa. Pofta bună!

Script Output

Task completed in 0.095 seconds
http://127.0.0.1:5500/oracle/mesaie/relational_1/insert

More Details :

https://docs.oracle.com/error-help/db/ora-20065/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04086/

0 rows deleted.

1 row inserted.

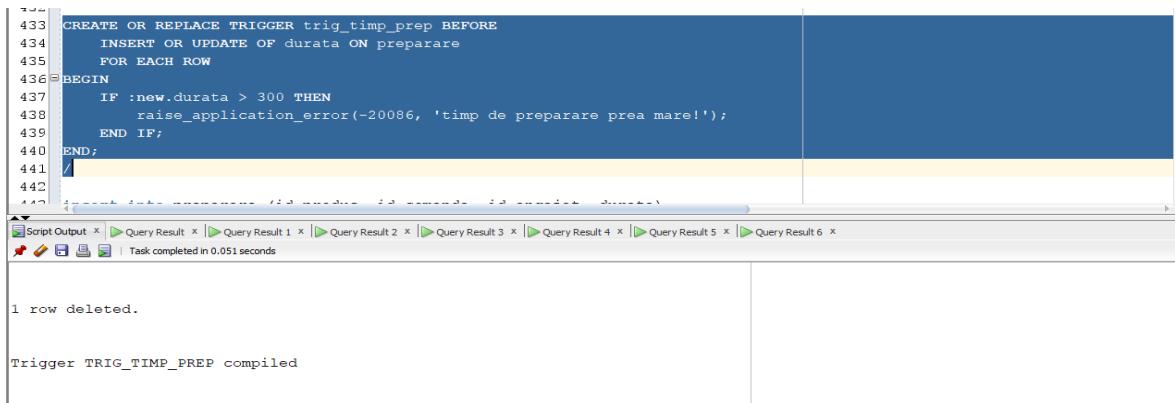
1 row deleted.

Aici am mai făcut încă un Trigger:

Un client nu poate aștepta mai mult decât 300 minute pentru ca comanda să fie preparată. Dacă timpul de preparare este mai mare decât 300 minute să se arunce o eroare la inserarea în tabelul preparare.

```
CREATE OR REPLACE TRIGGER trig_timp_prep BEFORE
    INSERT OR UPDATE OF durata ON preparare
    FOR EACH ROW
BEGIN
    IF :new.durata > 300 THEN
        raise_application_error(-20086, 'timp de preparare prea
mare!');
    END IF;
END;
/
```

Print-Screen:



The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor with the following PL/SQL code:

```
433 CREATE OR REPLACE TRIGGER trig_timp_prep BEFORE
434     INSERT OR UPDATE OF durata ON preparare
435     FOR EACH ROW
436 BEGIN
437     IF :new.durata > 300 THEN
438         raise_application_error(-20086, 'timp de preparare prea mare!');
439     END IF;
440 END;
441 /
442
```

In the bottom-left pane, the output window shows the results of the execution:

```
1 row deleted.

Trigger TRIG_TIMP_PREP compiled
```

L-am declanșat:

```
insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (1, 1, 17, 600);
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. In the code editor, a trigger named 'trig_ldd_client' is being created. The code includes an insert statement into the 'parapare' table and a raise_application_error statement. An error message is displayed in the output pane, indicating ORA-20086: timp de preparare prea mare! (prep time too long!).

```
441 /
442
443 insert into parapare (id_produs, id_comanda, id_angajat, durata)
444 values (1, 1, 17, 600);
445
446 -- Exercitiul 9.6
447 CREATE OR REPLACE TRIGGER trig_ldd_client BEFORE DROP ON SCHEMA BEGIN
  insert into parapare (id_produs, id_comanda, id_angajat, durata)
  values (1, 1, 17, 600)
  Error at Command Line : 443 Column : 13
  Error report -
  SQL Error: ORA-20086: timp de preparare prea mare!
  ORA-06512: at "ROBERTTO.TRIG_TIMP_PREP", line 3
  ORA-04088: error during execution of trigger 'ROBERTTO.TRIG_TIMP_PREP'
```

9.6 Crearea unui Trigger LDD.

```
CREATE OR REPLACE TRIGGER trig_ldd_client BEFORE DROP ON SCHEMA BEGIN
  IF lower(ora_dict_obj_name) = lower('Client') THEN
    raise_application_error(-20777, 'Nu aveti voie sa stergeti
tabela cu clienti');
  END IF;
END;
/
```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. A trigger named 'trig_ldd_client' is being created with logic to raise an application error if the schema object name is 'Client'. The output pane shows the error details for ORA-20086, ORA-06512, and ORA-04088.

```
446 -- Exercitiul 9.6
447 CREATE OR REPLACE TRIGGER trig_ldd_client BEFORE DROP ON SCHEMA BEGIN
  IF lower(ora_dict_obj_name) = lower('Client') THEN
    raise_application_error(-20777, 'Nu aveti voie sa stergeti tabela cu clienti');
  END IF;
END;
/
Trigger TRIG_LDD_CLIENT compiled
```

Declansarea acestuia:

```
drop table client;
```

Print-Screen:

The screenshot shows a SQL developer interface. In the top-left, there's a code editor with the following content:

```
453
454 drop table client;
```

Below the code editor is a toolbar with icons for script output, query result, and other database operations. A status bar at the bottom indicates "Task completed in 0.06 seconds".

The main pane displays the following error message:

```
Trigger TRIG_LDD_CLIENT compiled

Error starting at line : 454 in command -
drop table client
Error report -
ORA-04088: error during execution of trigger 'ROBERTTO.TRIG_LDD_CLIENT'
ORA-00604: error occurred at recursive SQL level 1
ORA-20777: Nu aveti voie sa stergeti tabela cu clienti
ORA-06512: at line 3

https://docs.oracle.com/error-help/db/ora-04088. 00000 - "error during execution of trigger '%s.%s'"
*Cause:   A runtime error occurred during execution of a trigger.
*Action:  Check the triggers which were involved in the operation.
```

9.7 Crearea unui unui pachet pentru obiectele definite.

```
CREATE OR REPLACE PACKAGE pachet_cerinte IS
    FUNCTION afis_produse_preferate (my_id_client IN
client.id_client%TYPE) RETURN result_obj;
    PROCEDURE afis_durata_preparare_produs;
    FUNCTION profit_client (my_id_client IN client.id_client%TYPE)
RETURN PLS_INTEGER;
END pachet_cerinte;
/

CREATE OR REPLACE PACKAGE BODY pachet_cerinte AS
    FUNCTION afis_produse_preferate (my_id_client IN
client.id_client%TYPE)
RETURN result_obj IS
    TYPE tab_ind IS TABLE OF PLS_INTEGER INDEX BY PLS_INTEGER;
    v_contor                      tab_ind;
    varray_ids                     t_array;
    v_index_max                    PLS_INTEGER;
    v_index_min                    PLS_INTEGER;
    id                            PLS_INTEGER;
    v_id_feluri_preferate         tab_imb;
    ok                           BINARY_INTEGER := 0;
    client_fara_comenzi          EXCEPTION;
    v_err_msg                      VARCHAR2(4000);
BEGIN
    v_id_feluri_preferate := tab_imb();
    varray_ids := t_array();
```

```

FOR i IN (SELECT p.id_produs, cont.numar_produse -- CURSOR
IMPLICIT
    FROM produs p, continut_comanda cont, comanda cmd
    WHERE my_id_client = cmd.id_client
    AND cont.id_produs = p.id_produs
    AND cmd.id_comanda = cont.id_comanda) LOOP BEGIN
    v_contor(i.id_produs) := v_contor(i.id_produs) +
i.numar_produse; -- simulez un vector de frecventa, unde tin de cate
ori a fost comandat produsul respectiv
    IF ok = 1 THEN
        IF v_contor(i.id_produs) > v_contor(v_index_max) THEN
            v_index_max := i.id_produs;
        END IF;
    END IF;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN -- intr-un tabel indexat daca nu
exista indexul respectiv
            -- se arunca exceptia no data found
            v_contor(i.id_produs) := i.numar_produse; --
initializam numarul de aparitii cu 1
        IF ok = 0 THEN -- daca e primul produs selectat de
cursor
            ok := 1;
            v_index_max := i.id_produs;
        END IF;
    END;
    END LOOP;
    IF ok = 0 THEN
        RAISE client_fara_comenzi;
    ELSE
        v_index_min := v_contor.FIRST;

        IF v_index_min IS NOT NULL THEN
            id := v_index_min;
            LOOP
                EXIT WHEN id IS NULL;
                IF v_contor(id) < v_contor(v_index_min) THEN
                    v_index_min := id;
                END IF;

                id := v_contor.NEXT(id);
            END LOOP;
        END IF;

        IF v_contor(v_index_max) != v_contor(v_index_min) THEN
--            DBMS_OUTPUT.PUT_LINE('v_index_max: ' ||
v_index_max);
--            DBMS_OUTPUT.PUT_LINE('v_contor(v_index_max): ' ||
v_contor(v_index_max));
--            DBMS_OUTPUT.PUT_LINE('v_index_min: ' ||
v_index_min);

```

```

--          DBMS_OUTPUT.PUT_LINE('v_contor(v_index_min): ' ||
v_contor(v_index_min));
          id := v_contor.first; -- prima valoare din tabelul
indexat (vector frecventa)
LOOP
    EXIT WHEN id IS NULL;
    IF v_contor(id) = v_contor(v_index_max) THEN
        v_id_feluri_preferate.extend;
    END IF;

v_id_feluri_preferate(v_id_feluri_preferate.LAST) := id;
    END IF;

    IF v_contor(id) = v_contor(v_index_min) THEN
        IF varray_ids.COUNT < varray_ids.LIMIT THEN
            varray_ids.EXTEND;
            varray_ids(varray_ids.LAST) := id;
        END IF;
    END IF;
    id := v_contor.NEXT(id);
END LOOP;
ELSE
    id := v_contor.first; -- prima valoare din tabelul
indexat (vector frecventa)
LOOP
    EXIT WHEN id IS NULL;
    IF v_contor(id) = v_contor(v_index_max) THEN
        v_id_feluri_preferate.extend;
    END IF;

v_id_feluri_preferate(v_id_feluri_preferate.LAST) := id;
    END IF;
    id := v_contor.NEXT(id);
END LOOP;
END IF;
RETURN result_obj(v_id_feluri_preferate, varray_ids);

EXCEPTION
    WHEN client_fara_comenzi THEN
        INSERT INTO mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
        VALUES (cod_mesaj.NEXTVAL, 'Pentru clientul dat de la
tastatura cu id ' || my_id_client || ' nu s-a gasit nicio comanda!',',
'W', USER, SYSDATE);
        COMMIT;
        DBMS_OUTPUT.PUT_LINE('Acest client nu a comandat nimic in
viata lui!');
    WHEN OTHERS THEN
        v_err_msg := 'Alta eroare in afis_produse_preferate: ' ||
SQLERRM;
        INSERT INTO mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)

```

```

        VALUES (cod_mesaj.NEXTVAL, 'Alta eroare in
afis_produse_preferate: ' || v_err_msg, 'E', USER, SYSDATE);
        COMMIT;
    END;

PROCEDURE afis_durata_preparare_produs
IS
    CURSOR c_bucatari IS SELECT a.id_angajat, a.nume, a.prenume,
b.nr_stele
                           FROM angajat a
                           JOIN bucatar b ON a.id_angajat =
b.id_angajat;
    CURSOR c_prepares(p_id_angajat in angajat.id_angajat%TYPE)
IS SELECT p.id_produs, p.nume, p.descriere, prep.durata
                           FROM preparare
prep
                           JOIN produs p ON
p.id_produs = prep.id_produs
                           WHERE
prep.id_angajat = p_id_angajat;
    TYPE rec IS RECORD (id_ang angajat.id_angajat%TYPE, dur
preparare.durata%TYPE);
    TYPE tab_ind IS TABLE OF rec INDEX BY PLS_INTEGER;
    t
                           tab_ind;
    v_id_bucatar
                           angajat.id_angajat%TYPE;
    v_nume_bucatar
                           angajat.nume%TYPE;
    v_prenume_bucatar
                           angajat.prenume%TYPE;
    v_nr_stele_bucatar
                           bucatar.nr_stele%TYPE;
    v_durata_prepares
                           preparare.durata%TYPE;
    v_id_produs
                           produs.id_produs%TYPE;
    v_nume_produs
                           produs.nume%TYPE;
    v_descriere_produs
                           produs.descriere%TYPE;
BEGIN
    OPEN c_bucatari;
    LOOP
        FETCH c_bucatari INTO v_id_bucatar, v_nume_bucatar,
v_prenume_bucatar, v_nr_stele_bucatar;
        EXIT WHEN c_bucatari%NOTFOUND;

        OPEN c_prepares(v_id_bucatar);
        FETCH c_prepares INTO v_id_produs, v_nume_produs,
v_descriere_produs, v_durata_prepares;

        IF c_prepares%NOTFOUND THEN
            DBMS_OUTPUT.PUT_LINE('ATENȚIE! Bucătarul ' ||
v_nume_bucatar || ' ' || v_prenume_bucatar || ' nu a preparat
niciodată un produs!');
            DBMS_OUTPUT.NEW_LINE();
        ELSE
            DBMS_OUTPUT.PUT_LINE('Bucătarul ' || v_nume_bucatar
|| ' ' || v_prenume_bucatar || ' a preparat următoarele produse:');
        END IF;
    END LOOP;
END;

```

```

        WHILE c_prepare%FOUND
        LOOP
            DBMS_OUTPUT.PUT_LINE(c_prepare%ROWCOUNT || '.');
Produsul: ' || v_nume_produs || '(' || v_descriere_produs || ') a
durat ' || v_durata_preparare || ' minute.');
            IF t.EXISTS(v_id_produs) THEN
                IF v_durata_preparare < t(v_id_produs).dur THEN
                    t(v_id_produs) := rec(v_id_bucatar,
v_durata_preparare);
                END IF;
            ELSE
                t(v_id_produs) := rec(v_id_bucatar,
v_durata_preparare);
            END IF;
            FETCH c_prepare INTO v_id_produs, v_nume_produs,
v_descriere_produs, v_durata_preparare;
        END LOOP;
        END IF;
        CLOSE c_prepare;
    END LOOP;
    CLOSE c_bucatari;

    FOR i IN t.FIRST..t.LAST LOOP
        SELECT nume INTO v_nume_produs
        FROM produs
        WHERE id_produs = i;

        SELECT a.nume, a.prenume INTO v_nume_bucatar,
v_prenume_bucatar
        FROM angajat a, bucatar b
        WHERE a.id_angajat = t(i).id_ang
        AND a.id_angajat = b.id_angajat;
        DBMS_OUTPUT.PUT_LINE('Bucatarul ' || v_nume_bucatar || '
' || v_prenume_bucatar || ' a preparat produsul ' || v_nume_produs || '
in ' || t(i).dur || ' minute(cel mai scurt timp).');
    END LOOP;
END;

FUNCTION profit_client (my_id_client IN client.id_client%TYPE)
RETURN PLS_INTEGER IS
    TYPE tab_ind IS TABLE OF PLS_INTEGER INDEX BY PLS_INTEGER;
    v_nume_client           client.nume%TYPE;
    v_prenume_client         client.nume%TYPE;
    v_nr_telefon_client     client.nr_telefon%TYPE;
    v_profit_maxim_companie PLS_INTEGER := 0;
    v_profit_maxim_client   PLS_INTEGER := 0;
    v_benef_red              BINARY_INTEGER := 0;
    v_valoare_red            PLS_INTEGER := 0;
    invalid                  EXCEPTION;
    profit_negativ          EXCEPTION;

```

```

        v_exista          PLS_INTEGER := 0;
        t
        v_err_msg         VARCHAR2(4000);
BEGIN
    SELECT COUNT(*) INTO v_exista
    FROM client
    WHERE id_client = my_id_client;
    IF v_exista = 0 THEN
        RAISE invalid;
    END IF;

    SELECT nume, prenume, nr_telefon INTO v_nume_client,
v_prenume_client, v_nr_telefon_client
    FROM client
    WHERE id_client = my_id_client;

    FOR i IN (SELECT cli.id_client, cli.nume, cli.prenume,
cmd.pret, f.valoare, f.detalii
              FROM client cli, comanda cmd, factura f
             WHERE cli.id_client = cmd.id_client
               AND cmd.id_factura = f.id_factura) LOOP
        BEGIN
            IF i.id_client = my_id_client AND i.valoare < i.pret
THEN
                v_benef_red := 1;
                v_valoare_red := v_valoare_red + (i.pret -
i.valoare);
                --
                ELSIF i.id_client = my_id_client THEN
                --
                    t(i.id_client) := t(i.id_client) + (i.valoare -
i.pret);
                END IF;
                v_profit_maxim_companie := v_profit_maxim_companie
+ (i.valoare - i.pret);
                t(i.id_client) := t(i.id_client) + (i.valoare -
i.pret);
            EXCEPTION -- pentru prima inserare in tabel
            WHEN NO_DATA_FOUND THEN
                insert into mesaje(cod_mesaj, mesaj, tip_mesaj,
creat_de, creat_la)
                    values (cod_mesaj.NEXTVAL, 'Prima inserare in
tabelul pentru clientul cu id: ' || i.id_client, 'I', USER, sysdate);
                commit;
                t(i.id_client) := i.valoare - i.pret;
            END;
        END LOOP;
        DBMS_OUTPUT.PUT_LINE('Profitul obtinut in urma clientilor de
pana acum este de ' || v_profit_maxim_companie || ' lei.');
        FOR i IN 1..t.LAST LOOP
            IF t(i) > v_profit_maxim_client THEN
                v_profit_maxim_client := t(i);
            END IF;
        
```

```

        END LOOP;
        DBMS_OUTPUT.PUT_LINE('Profitul maxim adus de un client este
de ' || v_profit_maxim_client || ' lei.');
        DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul ' || my_id_client ||
' este ' || v_nume_client || ' ' || v_prenume_client || ' cu nr de
telefon: ' || v_nr_telefon_client || '.');
        IF v_benef_red = 1 THEN
            DBMS_OUTPUT.PUT_LINE('A beneficiat de reduceri in valoare
de ' || v_valoare_red || ' lei.');
        ELSE
            DBMS_OUTPUT.PUT_LINE('Nu beneficiaza de reduceri!');
        END IF;
        IF t.EXISTS(my_id_client) AND t(my_id_client) < 0 THEN
            RAISE profit_negativ;
        END IF;
        RETURN t(my_id_client);
EXCEPTION
    WHEN invalid THEN
        insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
            values (cod_mesaj.NEXTVAL, 'Client inexistent cu id=' ||
my_id_client, 'E', USER, sysdate);
            commit;
            RAISE_APPLICATION_ERROR(-20021, 'Nu a fost gasit niciun
client cu id-ul introdus!');
    WHEN profit_negativ THEN
        insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
            values (cod_mesaj.NEXTVAL, 'Profit negativ pentru
clientul ' || my_id_client, 'I', USER, sysdate);
            commit;
            RAISE_APPLICATION_ERROR(-20022, 'Acest client a generat
un profit negativ (prea multe reduceri)!');
    WHEN OTHERS THEN
        v_err_msg := 'Alta eroare la profit_client, cod Oracle: '
|| SQLERRM;
        insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de,
creat_la)
            values (cod_mesaj.NEXTVAL, v_err_msg, 'E', USER,
sysdate);
            commit;
            RAISE_APPLICATION_ERROR(-20022, 'Alta eroare!');
    END;
END pachet_cerinte;
/

```

Print-Screen:

The screenshot shows the Oracle SQL Developer interface. In the top pane, the code for the package body PACHET_CERINTE is displayed. The code includes logic to calculate profit, handle negative profits, and log errors. In the bottom pane, the results of the compilation process are shown, indicating that the package and its body were successfully compiled.

```
680     END;
681   END LOOP;
682   DBMS_OUTPUT.PUT_LINE('Profitul obtinut in urma clientilor de pana acum este de ' || v_profit_maxim_companie || ' lei.');
683   FOR i IN 1..t.LAST LOOP
684     IF t(i) > v_profit_maxim_client THEN
685       v_profit_maxim_client := t(i);
686     END IF;
687   END LOOP;
688   DBMS_OUTPUT.PUT_LINE('Profitul maxim adus de un client este de ' || v_profit_maxim_client || ' lei.');
689   DBMS_OUTPUT.PUT_LINE('Clientul cu id-ul ' || my_id_client || ' este ' || v_nume_client || ' ' || v_prenume_client);
690   IF v_benef_red = 1 THEN
691     DBMS_OUTPUT.PUT_LINE('A beneficiat de reduceri in valoare de ' || v_valoare_red || ' lei.');
692   ELSE
693     DBMS_OUTPUT.PUT_LINE('Nu beneficiaza de reduceri!');
694   END IF;
695   IF t.EXISTS(my_id_client) AND t(my_id_client) < 0 THEN
696     RAISE profit_negativ;
697   END IF;
698   RETURN t(my_id_client);
699 EXCEPTION
700   WHEN invalid THEN
701     insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
702     values (cod_mesaj.NEXTVAL, 'Client inexistent cu id=' || my_id_client, 'E', USER, sysdate);
703     commit;
704     RAISE_APPLICATION_ERROR(-20021, 'Nu a fost gasit niciun client cu id-ul introdus!');
705   WHEN profit_negativ THEN
706     insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
707     values (cod_mesaj.NEXTVAL, 'Profit negativ pentru clientul ' || my_id_client, 'I', USER, sysdate);
708     commit;
709     RAISE_APPLICATION_ERROR(-20022, 'Acest client a generat un profit negativ (prea multe reduceri)!');
710   WHEN OTHERS THEN
711     v_err_msg := 'Alta eroare la profit_client, cod Oracle: ' || SQLERRM;
712     insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
713     values (cod_mesaj.NEXTVAL, v_err_msg, 'E', USER, sysdate);
714     commit;
715     RAISE_APPLICATION_ERROR(-20022, 'Alta eroare!');
716   END;
717 END pachet_cerinte;
718 /
719
```

Script Output | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Task completed in 0.318 seconds

*Action: Check the triggers which were involved in the operation.

Package PACHET_CERINTE compiled

Package Body PACHET_CERINTE compiled

Apelare din pachet:

```
-- Exercitiul 9.1 Apel din pachet
DECLARE
    v_id_client_citit      client.id_client%TYPE := &client_id;
    v_result                result_obj;
    my_tab                  tab_imb;
    my_varray               t_array;
    id                      PLS_INTEGER;
    v_nume_produs           produs.nume%TYPE;
    v_nume                  client.nume%TYPE;
    v_prenume               client.prenume%TYPE;
    v_contor                PLS_INTEGER := 0;
BEGIN
    v_result :=
    pachet_cerinte.afis_produse_preferate(v_id_client_citit);
    my_tab := v_result.v_id_feluri_preferate;
    my_varray := v_result.varray_ids;

    SELECT c.nume, c.prenume INTO v_nume, v_prenume
    FROM client c
    WHERE c.id_client = v_id_client_citit;
```

```

DBMS_OUTPUT.PUT_LINE('Clientul ' || v_nume || ' ' || v_prenume ||
' are urmatoarele produse favorite:');
    id := my_tab.FIRST; -- prima valoare din tabelul indexat (vector
frecventa)
    LOOP
        EXIT WHEN id IS NULL;
        SELECT p.nume INTO v_nume_produs
        FROM produs p
        WHERE p.id_produs = my_tab(id);
        -- afisam produsele cu numarul maxim de aparitii
        v_contor := v_contor + 1;
        DBMS_OUTPUT.PUT_LINE(v_contor || '. ' || v_nume_produs);
        id := my_tab.NEXT(id);
    END LOOP;

    IF my_varray.COUNT = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Niciun produs nu a fost comandat de mai
putine ori fata de celealte.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('');
        DBMS_OUTPUT.PUT_LINE('Produse cel mai putin comandate: ');
        v_contor := 0;
        FOR idx IN 1..my_varray.COUNT LOOP
            SELECT p.nume
                INTO v_nume_produs
                FROM produs p
                WHERE p.id_produs = my_varray(idx);
            v_contor := v_contor + 1;
            DBMS_OUTPUT.PUT_LINE(v_contor || '. ' || v_nume_produs);
        END LOOP;
    END IF;
END;
/

```

localhost MESAJE Relational_1 (Untitled_1)

Query Builder

localhost x

DBMS Output Buffer Size:20000

```
v_result := pachet_cerinte.afis_produse_preferate(v_id_client_citit);
my_tab := v_result.v_id_feluri_preferate;
my_varray := v_result.varray_ids;

SELECT c.nume, c.prenume INTO v_nume, v_prenume
FROM client c
WHERE c.id_client = v_id_client_citit;
DBMS_OUTPUT.PUT_LINE('Clientul ' || v_nume || ' ' || v_prenume || ' are urmatoarele produse favorite:');
id := my_tab.FIRST; -- prima valoare din tabelul indexat (vector frecventa)
LOOP
  EXIT WHEN id IS NULL;
  SELECT p.num INTO v_nume_produs
  FROM produs p
  WHERE p.id_produs = my_tab(id);
  -- afisam produsele cu numarul maxim de aparitii
  v_contor := v_contor + 1;
  DBMS_OUTPUT.PUT_LINE(v_contor || '.' || v_nume_produs);
  id := my_tab.NEXT(id);
END LOOP;

IF my_varray.COUNT = 0 THEN
  DBMS_OUTPUT.PUT_LINE('Niciun produs nu a fost comandat de mai putine ori fata de celealte.');
ELSE
  DBMS_OUTPUT.PUT_LINE('');
  DBMS_OUTPUT.PUT_LINE('Produse cel mai putin comandate: ');
  v_contor := 0;
  FOR idx IN 1..my_varray.COUNT LOOP
    SELECT p.num
    INTO v_nume_produs
    FROM produs p
    WHERE p.id_produs = my_varray(idx);
    v_contor := v_contor + 1;
    DBMS_OUTPUT.PUT_LINE(v_contor || '.' || v_nume_produs);
  END LOOP;
END IF;
END;
```

localhost x

Clientul Gheorghe Sebastian are urmatoarele produse favorite:

1. Spaghete
2. Pizza
3. Ciocan de post

Niciun produs nu a fost comandat de mai putine ori fata de celealte.

```
-- Exercitiul 9.2 Apel din pachet
BEGIN
    pachet_cerinte.afis_durata_preparare_produs();
END;
/

```

The screenshot shows the Oracle SQL Developer interface with the following details:

- Left Panel (Query Builder):** Displays the PL/SQL code for Exercise 9.2. The code includes logic to print favorite dishes for clients based on their preparation times.
- Right Panel (Dom Output):** Shows the output of the executed procedure. It lists favorite dishes for different clients and their preparation times.
- Bottom Status Bar:** Shows the message "PL/SQL procedure successfully completed." twice.

```

-- Exercitiul 9.2 Apel din pachet
BEGIN
    pachet_cerinte.afis_durata_preparare_produs();
END;
/

```

```

Clientul Gheorghe Sebastian are următoarele produse favorite:
1. Spaghete
2. Pizza
3. Ciorba de post
Niciun produs nu a fost comandat de mai putine ori fata de celelalte.

Bucătarul Filip Mihnea a preparat următoarele produse:
1. Produsul: Spaghete ( picante ) a durat 45 minute.
2. Produsul: Paste bolognese ( cu sos dulce ) a durat 16 minute.
Bucătarul Dumitrescu Florin a preparat următoarele produse:
1. Produsul: Spaghete ( picante ) a durat 60 minute.
2. Produsul: Spaghete ( picante ) a durat 130 minute.
3. Produsul: Ciorba de post ( fara carne ) a durat 75 minute.
Bucătarul Sociu Razvan a preparat următoarele produse:
1. Produsul: Sarmale ( ca la mama acasa ) a durat 240 minute.
Bucătarul Scarlatescu Catalin a preparat următoarele produse:
1. Produsul: Pizza ( dulce aromata ) a durat 35 minute.
2. Produsul: Paste bolognese ( cu sos dulce ) a durat 25 minute.
3. Produsul: Sarmale ( ca la mama acasa ) a durat 120 minute.
Bucătarul Bontea Sorin a preparat următoarele produse:
1. Produsul: Paste bolognese ( cu sos dulce ) a durat 70 minute.
ATENȚIE! Bucătarul Daniel Andrei nu a preparat niciodată un produs!

Bucătarul Filip Mihnea a preparat produsul Spaghete in 45 minute(cel mai scurt timp).
Bucătarul Scarlatescu Catalin a preparat produsul Pizza in 35 minute(cel mai scurt timp).
Bucătarul Filip Mihnea a preparat produsul Paste bolognese in 16 minute(cel mai scurt timp).
Bucătarul Scarlatescu Catalin a preparat produsul Sarmale in 120 minute(cel mai scurt timp).
Bucătarul Dumitrescu Florin a preparat produsul Ciorba de post in 75 minute(cel mai scurt timp)

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

```

```
-- Exercitiul 9.3 Apel din pachet
DECLARE
    v_id_client_citit      client.id_client%TYPE := &client_id;
    v_rezultat              PLS_INTEGER;

BEGIN
    v_rezultat := pachet_cerinte.profit_client(v_id_client_citit);
    IF v_rezultat > 0 THEN
        DBMS_OUTPUT.PUT_LINE('El a contribuit la profitul companiei
pana acum cu ' || v_rezultat || ' lei.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Acesta a beneficiat de prea multe
reduceri! Aducand un profit negativ companiei: ' || v_rezultat || '
lei.');
    END IF;
END;
/
```

The screenshot shows the Oracle SQL Developer interface with the following details:

- Tools Bar:** Includes standard options like File, Window, Help, and various icons for connection management.
- Connections:** Shows a connection to "localhost" with a buffer size of 20000.
- Script Output:** Displays the PL/SQL code for Exercise 9.3, which calculates profit based on client ID input.
- Output Window:** Shows the results of the DBMS_OUTPUT.PUT_LINE statements:
 - Profitul obtinut in urma clientilor de pana acum este de 60 lei.
 - Profitul maxim adus de un client este de 30 lei.
 - Cliențul cu id-ul 3 este Gheorghe Sebastian cu nr de telefon: 0744573419.
 - Nu beneficiază de reduceri!
 - El a contribuit la profitul companiei pana acum cu 20 lei.
- Bottom Status:** Shows "PL/SQL procedure successfully completed." and a note about completing in 1.534 seconds.

â

10. Inserarea exceptiilor în tabelul mesaje:

La exercițiul 9.1:

```
543     id := v_contor.first; -- prima valoare din tabelul indexat (vector frecventa)
544     LOOP
545         EXIT WHEN id IS NULL;
546         IF v_contor(id) = v_contor(v_index_max) THEN
547             v_id_feluri_preferate.extend;
548             v_id_feluri_preferate(v_id_feluri_preferate.LAST) := id;
549         END IF;
550         id := v_contor.NEXT(id);
551     END LOOP;
552     END IF;
553 END IF;
554 RETURN result_obj(v_id_feluri_preferate, varray_ids);
555
556 EXCEPTION
557     WHEN client_fara_comenzi THEN
558         INSERT INTO mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
559             VALUES (cod_mesaj.NEXTVAL, 'Pentru clientul dat de la tastatura cu id ' || my_id_client || ' nu s-a gasit niciun client');
560             COMMIT;
561             DBMS_OUTPUT.PUT_LINE('Acest client nu a comandat nimic in viata lui!');
562     WHEN OTHERS THEN
563         v_err_msg := 'Alta eroare in afis_produse_preferate: ' || SQLERRM;
564         INSERT INTO mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
565             VALUES (cod_mesaj.NEXTVAL, 'Alta eroare in afis_produse_preferate: ' || v_err_msg, 'E', USER, SYSDATE);
566             COMMIT;
567 END;
```

La exercițiul 9.3:

```
571     END IF;
572     v_profit_maxim_companie := v_profit_maxim_companie + (i.valoare - i.pret);
573     t(i.id_client) := t(i.id_client) + (i.valoare - i.pret);
574 EXCEPTION -- pentru prima inserare in tabel
575     WHEN NO_DATA_FOUND THEN
576         insert into mesaje(cod_mesaj, mesaj, tip_mesaj, creat_de, creat_la)
577             values (cod_mesaj.NEXTVAL, 'Prima inserare in tabelul pentru clientul cu id: ' || i.id_client, 'I',
578             commit;
579             t(i.id_client) := i.valoare - i.pret;
580     END;
581 END LOOP;

582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718 /
```

Mesajele stocate în tabelul mesaje:

```
791
792 select * from mesaje;
793
```

Script Output x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | Query Result 6 x

SQL | All Rows Fetched: 60 in 0.007 seconds

COD_MESAJ	MESAJ	TIP_MESAJ	CREAT_DE	CREAT_LA
36	36 Prima inserare in tabelul pentru clientul cu id: 4	I	ROBERTTO	30-JAN-25
37	37 Prima inserare in tabelul pentru clientul cu id: 5	I	ROBERTTO	30-JAN-25
38	38 Alta eroare la profit_client, cod Oracle: ORA-01403: no data found	E	ROBERTTO	30-JAN-25
39	39 Prima inserare in tabelul pentru clientul cu id: 1	I	ROBERTTO	30-JAN-25
40	40 Prima inserare in tabelul pentru clientul cu id: 2	I	ROBERTTO	30-JAN-25
41	41 Prima inserare in tabelul pentru clientul cu id: 3	I	ROBERTTO	30-JAN-25
42	42 Prima inserare in tabelul pentru clientul cu id: 4	I	ROBERTTO	30-JAN-25
43	43 Prima inserare in tabelul pentru clientul cu id: 5	I	ROBERTTO	30-JAN-25
44	44 Profit negativ pentru clientul 1	I	ROBERTTO	30-JAN-25
45	45 Prima inserare in tabelul pentru clientul cu id: 1	I	ROBERTTO	30-JAN-25
46	46 Prima inserare in tabelul pentru clientul cu id: 2	I	ROBERTTO	30-JAN-25
47	47 Prima inserare in tabelul pentru clientul cu id: 3	I	ROBERTTO	30-JAN-25
48	48 Prima inserare in tabelul pentru clientul cu id: 4	I	ROBERTTO	30-JAN-25
49	49 Prima inserare in tabelul pentru clientul cu id: 5	I	ROBERTTO	30-JAN-25
50	50 Client inexistent cu id=100	E	ROBERTTO	30-JAN-25
51	51 Prima inserare in tabelul pentru clientul cu id: 1	I	ROBERTTO	30-JAN-25
52	52 Prima inserare in tabelul pentru clientul cu id: 2	T	ROBERTTO	30-JAN-25