

# Proiect - Baze de Date

- Gestiunea unui lanț de restaurante -

Popescu Paullo Robertto Karloss

Grupa 131

2021 Mai

# 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 la 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 (minim 5 înregistrări în fiecare tabel neasociativ, minimum 10 înregistrări în tabelele asociative) + crearea unei secvențe.
  - 6.3 Diagrama generată în sql după crearea tabelelor și inserarea datelor.
7. Crearea a 5 cereri complexe în SQL.
8. Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri.
9. O cerere care utilizează operația *outer-join* pe minim 4 tabele și două cereri ce utilizează operația *division*.
10. Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimată prin expresie algebrică, arbore algebric și limbaj (SQL), atât anterior cât și ulterior optimizării.
11. a) Realizarea normalizării BCNF, FN4, FN5.  
b) Aplicarea denormalizării.

# 1. Prezentarea bazei de date

## 1.1 Tehnologii folosite pentru realizarea proiectului.

Pentru proiectul din cadrul cursului de Baze de Date am folosit versiunea **o19c** 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.

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 trebuiesc preparate (de exemplu vin, șampanie 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 postal 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 comenzii*.
- Entitatea **Angajat**, care va conține 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 numărul de ani de experiență al fiecărui chelner. Cheie primară fiind *id-ul angajatului*.
- Subentitatea **Manager**, care nu are attribute, 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 aparține 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 fie chelner, casier, bucătar sau manager.

### 1.6 Descrierea atributelor.

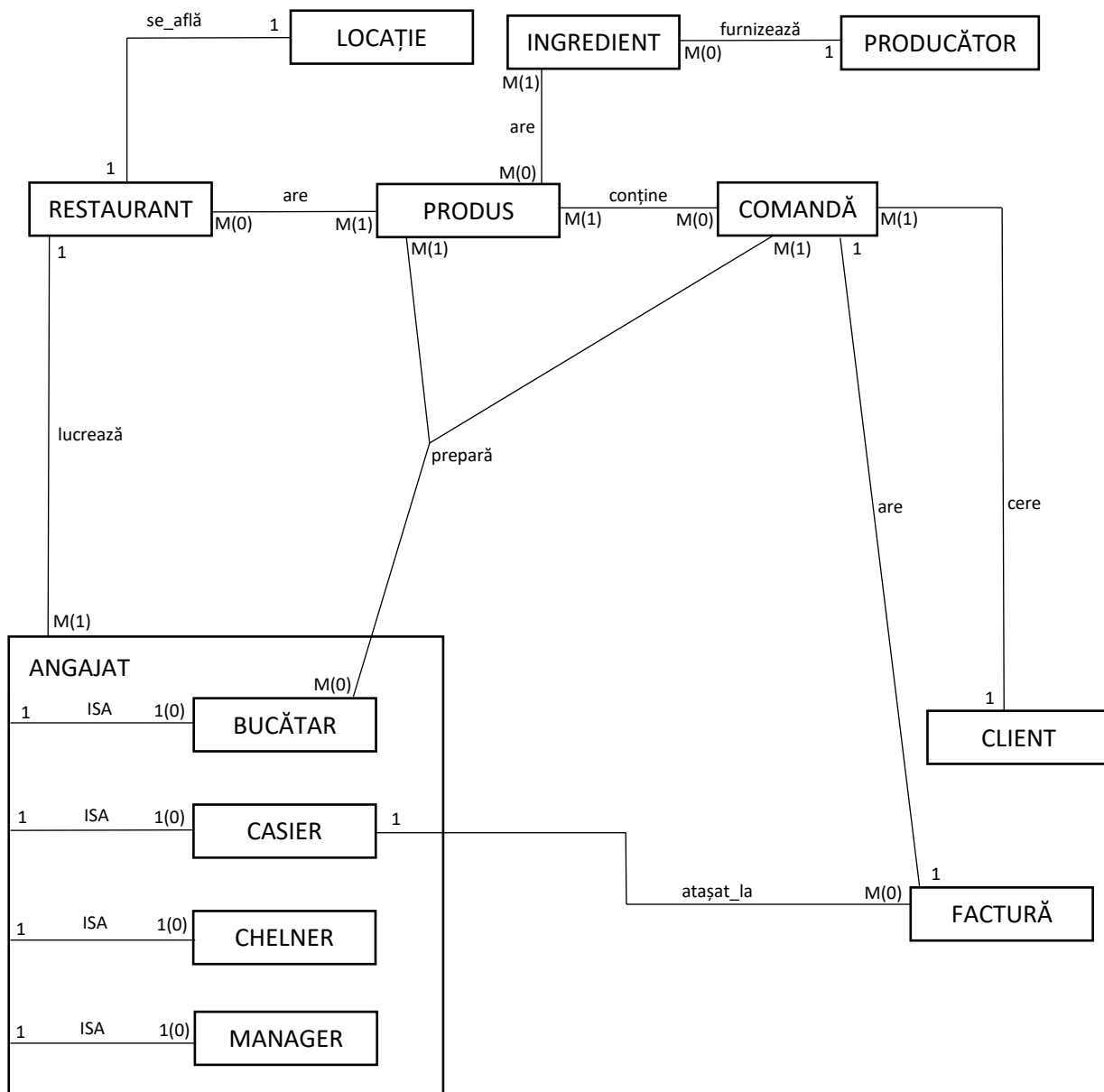
- Entitatea **restaurant** va avea ca atribute id-ul și numele restaurantului curent (de exemplu "Yamas", "Ivans" etc).  
Tipul de date pentru id-ul restaurantului va fi number(10), id-locăție number(10), iar pentru nume va fi varchar2(32).  
Cheia externă va fi id\_locăție (provenită din tabelul locăție), cu tipul de date number(10).  
PK-ul va fi id-ul restaurantului (cheia primară).  
Constrângere pentru numele restaurantului: Unique.
- Entitatea **locăție** va avea ca atribute: id-ul locăției, țara, orașul, codul poștal și strada pentru restaurantul respectiv.  
Tipul de date pentru id-ul locăției va fi number(10), țara varchar2(20), orașul varchar2(20), codul poștal varchar2(15) și strada varchar2(50).  
PK-ul va fi id-ul locăției (cheia primară).  
Constrângere pentru codul poștal: Unique.
- Entitatea **produs** va avea ca atribute: 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 **menu** va avea ca atribute: 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 atribute: 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 atribute: 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 atribute: 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 atribute: 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 comenzii 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 comenzii (cheia primară).  
Constrângere pentru prețul comenzii: NOT\_NULL, dar și check preț > 0.  
Atributul data va avea setat ca default ziua curentă (sysdate).
- Tabelul asociativ **conținut\_comandă** va avea ca atribute: id-ul produsului, id-ul comenzii și numărul de produse.  
Avem o cheie primară compusă formată din id-ul produsului (FK din tabelul produs) și id-ul comenzii (FK din comandă).  
Tipul de date pentru id-ul produsului va fi number(10), id-ul comenzii number(20) și numărul de produse din comandă number(10).  
Constrângere pentru numărul de produse: check număr\_produse > 0.
- Entitatea **angajat** va avea ca atribute: id-ul, numele, prenumele și data angajării (am ales această abordare pentru că niciun subtip nu avea atribute separate de angajat).  
Angajat va avea ca cheie externă id-restaurant (provenită din tabelul restaurant).  
Tipul de date pentru id-ul angajatului va fi number(10), 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 atribute 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 atribute 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).

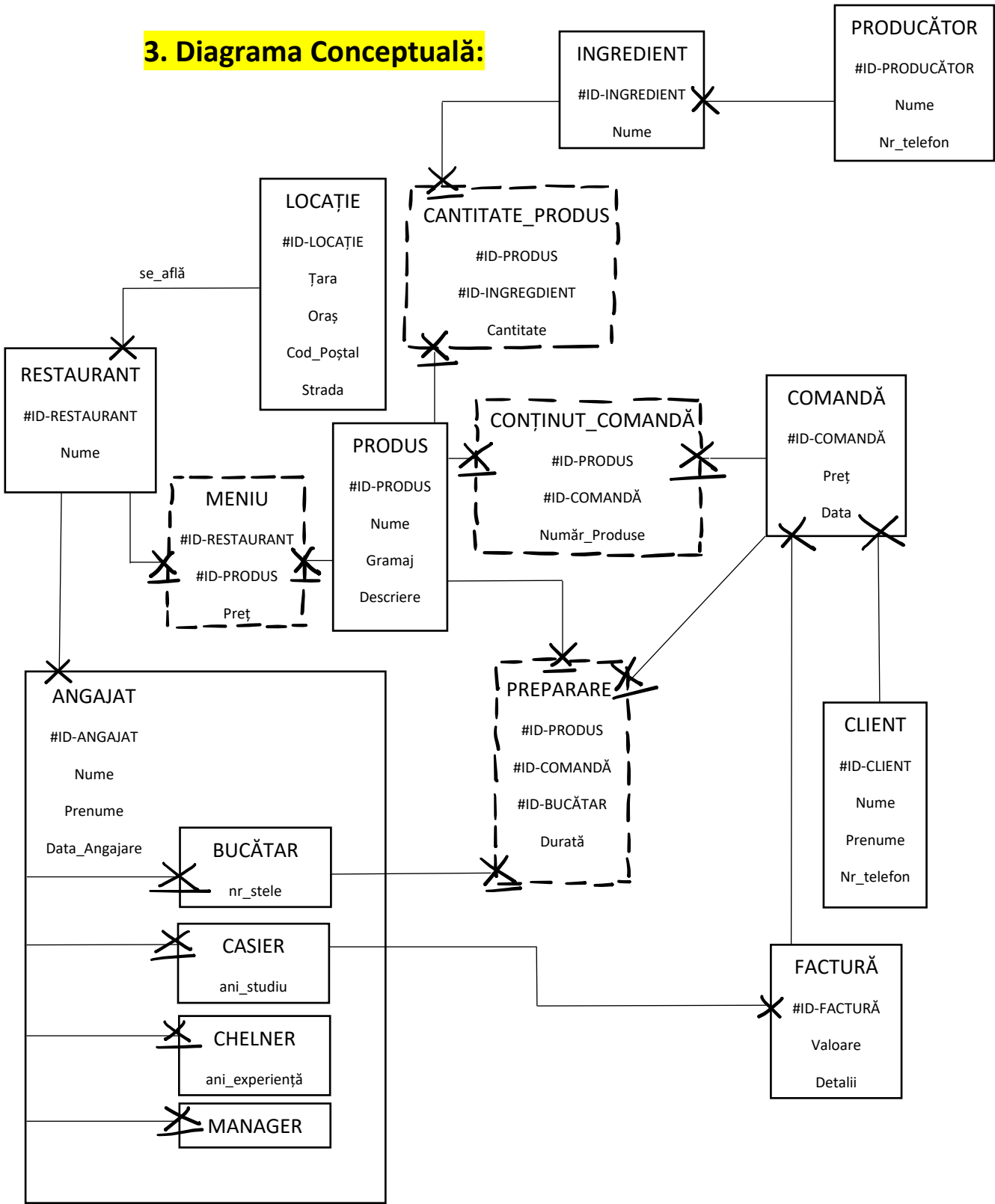
- Subentitatea **chelner** va avea ca atribute id-ul angajatului și numărul de ani de experiență.  
Tipul de date pentru id-ul angajatului va fi number(10), iar pentru numărul de ani de experiență number(5).  
PK va fi id\_angajat (care provine din tabelul angajat).
- Subentitatea **manager** va avea ca atribut 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 atribute: id-ul produsului, id-ul comenzii, id-ul bucătarului și durata.  
Avem o cheie primară compusă formată din id-ul produsului (FK din tabelul produs), id-ul comenzii (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 comenzii number(20), id-ul bucătarului number(10) și durata number(10).
- Entitatea **client** va avea ca atribute: 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 atribute: 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 relațională:

- RESTAURANT (#ID-RESTAURANT, id-locăție, Nume)
- LOCAȚIE (#ID-LOCAȚIE, Țara, 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-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 ingrediente, 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 (dependența 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:

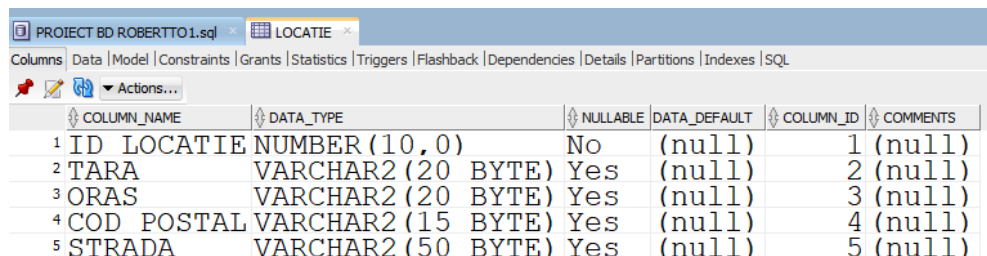
### 6.1 Crearea tabelelor și a constrângerilor.

--- 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 the 'LOCATIE' table selected. The 'Columns' tab is active, displaying the following table structure:

	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)

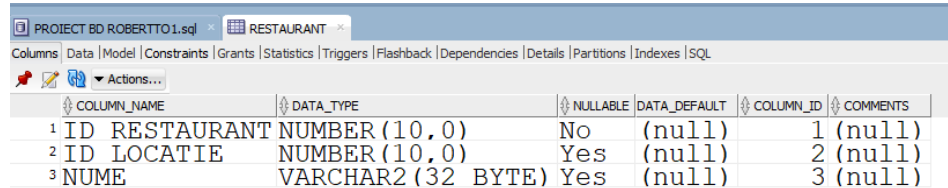
-- RESTAURANT --

```
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 'Columns' tab for the 'RESTAURANT' table in a project named 'PROJECT BD ROBERTO1.sql'. The table has three columns: 'ID RESTAURANT' (NUMBER(10,0), Not Null, Default (null), Column ID 1), 'ID LOCATIE' (NUMBER(10,0), Yes, Default (null), Column ID 2), and 'NUME' (VARCHAR2(32 BYTE), Yes, Default (null), Column ID 3).

	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),

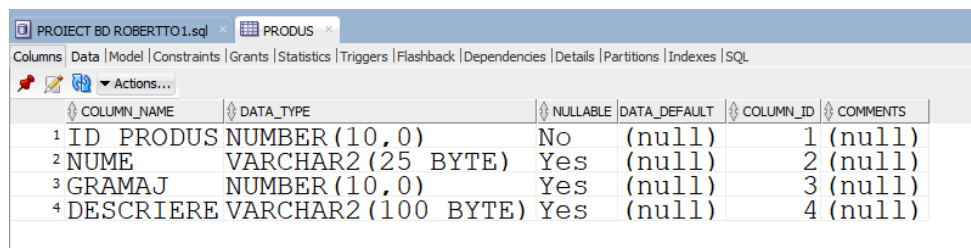
    gramaj    NUMBER(10),

    descriere VARCHAR2(100),

    CONSTRAINT chk_gramaj CHECK ( gramaj > 0 )

);
```

**Print-Screen:**



The screenshot shows the 'Columns' tab for the 'PRODUS' table in a project named 'PROJECT BD ROBERTO1.sql'. The table has four columns: 'ID PRODUS' (NUMBER(10,0), Not Null, Default (null), Column ID 1), 'NUME' (VARCHAR2(25 BYTE), Yes, Default (null), Column ID 2), 'GRAMAJ' (NUMBER(10,0), Yes, Default (null), Column ID 3), and 'DESCRIERE' (VARCHAR2(100 BYTE), Yes, Default (null), Column ID 4).

	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)

```
-- MENUU --

CREATE TABLE meniu (

    id_produs  NUMBER(10),
```

```

id_restaurant NUMBER(10),
pret          NUMBER(10),
CONSTRAINT menu_pk PRIMARY KEY ( id_produ,
                                id_restaurant ),
CONSTRAINT menu_produ_fk FOREIGN KEY ( id_produ )
    REFERENCES produ ( id_produ ),
CONSTRAINT menu_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)

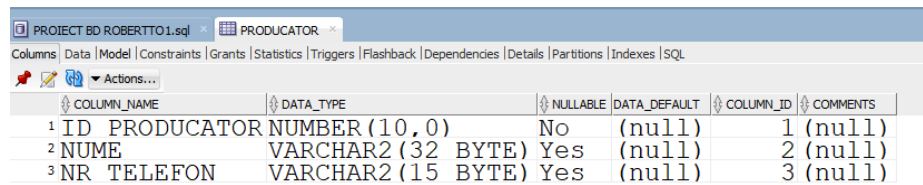
-- PRODUCATOR --

```

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

```

### Print-Screen:



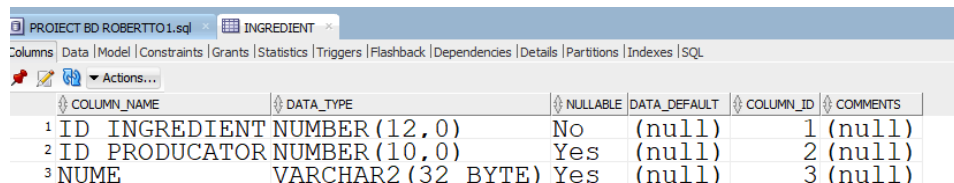
The screenshot shows the 'PRODUCATOR' table structure in SQL Developer. The table has three columns: ID, NUME, and NR TELEFON. The ID column is a NUMBER(10,0) and is the primary key. The NUME column is a VARCHAR2(32 BYTE). The NR TELEFON column is a VARCHAR2(15 BYTE). All columns are nullable by default.

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:



The screenshot shows the 'INGREDIENT' table structure in SQL Developer. The table has three columns: ID INGREDIENT, ID PRODUCATOR, and NUME. The ID INGREDIENT column is a NUMBER(12,0) and is the primary key. The ID PRODUCATOR column is a NUMBER(10,0). The NUME column is a VARCHAR2(32 BYTE). All columns are nullable by default.

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:**

The screenshot shows the 'Columns' tab for the table 'CANTITATE\_PRODUS'. The table has three columns: ID\_PRODUS, ID\_INGREDIENT, and CANTITATE. The data types are NUMBER(10,0), NUMBER(12,0), and NUMBER(10,0) respectively. The NULLABLE column shows 'No' for ID\_PRODUS and ID\_INGREDIENT, and 'Yes' for CANTITATE. The DATA\_DEFAULT column shows '(null)' for all three. The COLUMN\_ID column shows 1, 2, and 3 respectively. The COMMENTS column is empty.

	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),

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 )

);

```

**Print-Screen:**

The screenshot shows the 'Columns' tab for the table 'ANGAJAT'. The table has five columns: ID\_ANGAJAT, ID\_RESTAURANT, NUME, PRENUME, and DATA\_ANGAJARE. The data types are NUMBER(10,0), NUMBER(10,0), VARCHAR2(32 BYTE), VARCHAR2(32 BYTE), and DATE respectively. The NULLABLE column shows 'No' for ID\_ANGAJAT, and 'Yes' for the others. The DATA\_DEFAULT column shows '(null)' for ID\_ANGAJAT, ID\_RESTAURANT, NUME, and PRENUME, and 'to date(sysdate, 'dd-mm-yy')' for DATA\_ANGAJARE. The COLUMN\_ID column shows 1, 2, 3, 4, and 5 respectively. The COMMENTS column is empty.

	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)	Yes	(null)	2	(null)
3	NUME	VARCHAR2(32 BYTE)	Yes	(null)	3	(null)
4	PRENUME	VARCHAR2(32 BYTE)	Yes	(null)	4	(null)
5	DATA_ANGAJARE	DATE	Yes	to date(sysdate, 'dd-mm-yy')	5	(null)

```
-- CHELNER --

CREATE TABLE chelner (

    id_angajat    NUMBER(10),

    ani_experienta  NUMBER(5),

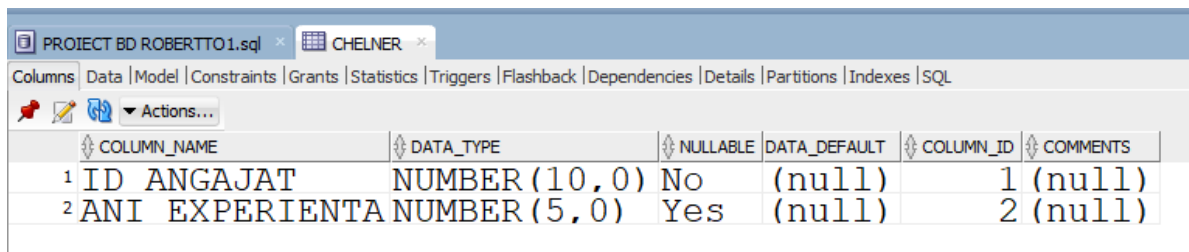
    CONSTRAINT chelner_pk PRIMARY KEY ( id_angajat ),

    CONSTRAINT angajat_fk FOREIGN KEY ( id_angajat )

        REFERENCES angajat ( id_angajat )

);
```

**Print-Screen:**



	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),

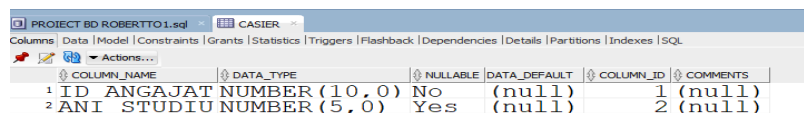
    CONSTRAINT casier_pk PRIMARY KEY ( id_angajat ),

    CONSTRAINT angajat_casier_fk FOREIGN KEY ( id_angajat )

        REFERENCES angajat ( id_angajat )

);
```

**Print-Screen:**



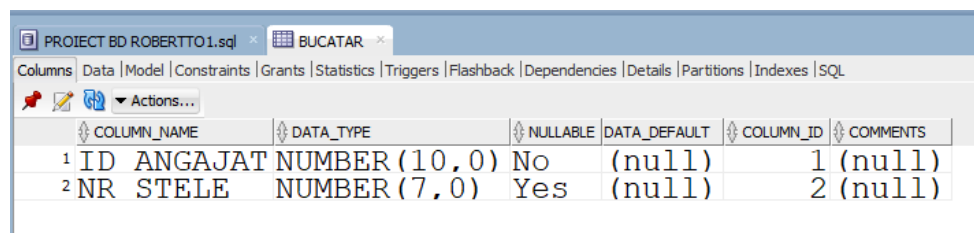
	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID ANGAJAT	NUMBER(10,0)	No	(null)	1	(null)
2	ANI STUDIUM	NUMBER(5,0)	Yes	(null)	2	(null)



-- 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:**



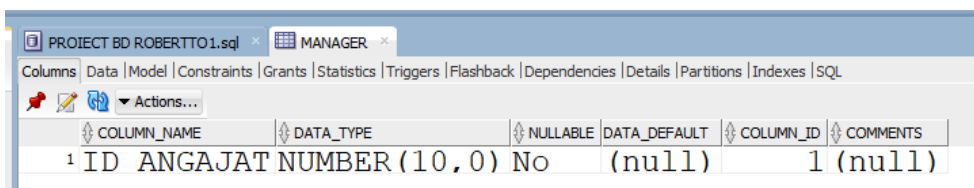
The screenshot shows the 'Columns' tab in SQL Developer for the 'BUCATAR' table. The table has two columns: 'ID\_ANGAJAT' and 'NR\_STELE'. 'ID\_ANGAJAT' is the primary key, is not nullable, and has a default value of null. 'NR\_STELE' is nullable and also has a default value of null.

	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:**



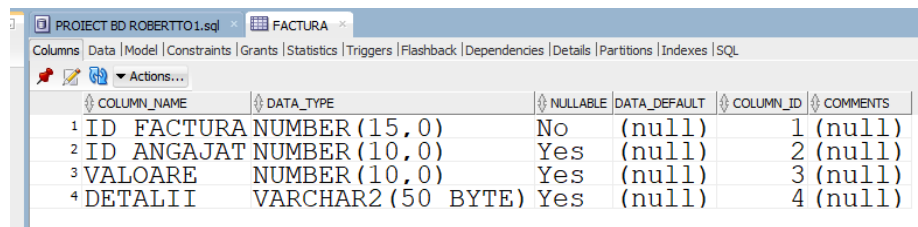
The screenshot shows the 'Columns' tab in SQL Developer for the 'MANAGER' table. The table has one column: 'ID\_ANGAJAT'. It is the primary key, is not nullable, and has a default value of null.

	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:**



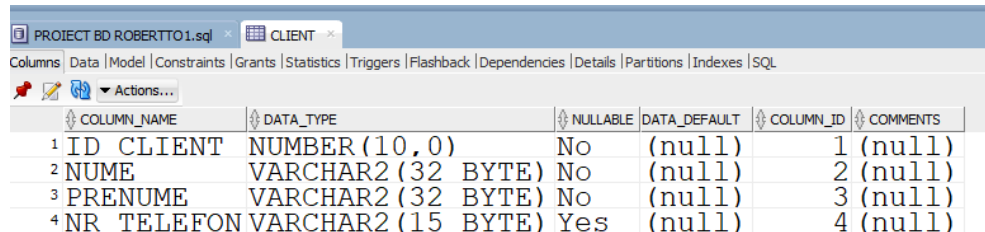
The screenshot shows a database management tool interface with a tab labeled 'FACTURA'. Below the tab, there is a table with columns: COLUMN\_NAME, DATA\_TYPE, NULLABLE, DATA\_DEFAULT, COLUMN\_ID, and COMMENTS. The table contains four rows of data representing the columns of the FACTURA table.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	ID_FACTURA	NUMBER (15,0)	No	(null)	1	(null)
2	ID_ANGAJAT	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 'CLIENT' table structure in Oracle SQL Developer. The table has four columns: ID\_CLIENT, NUME, PRENUME, and NR TELEFON. The first column is a primary key and is not nullable. The other three columns are nullable and have a default value of null.

	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 ),

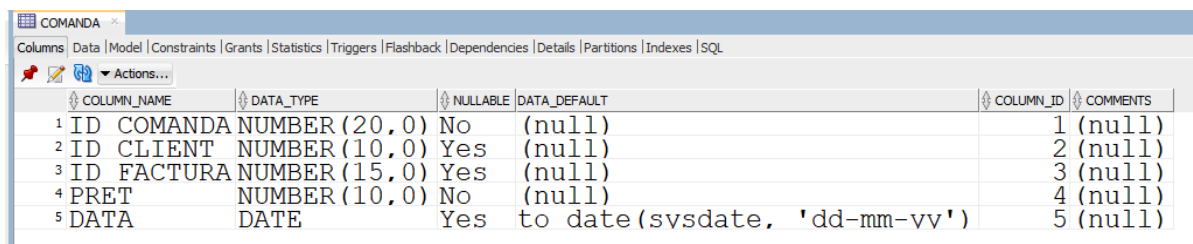
CONSTRAINT fk\_comanda\_factura FOREIGN KEY ( id\_factura )

REFERENCES factura ( id\_factura ),

CONSTRAINT chk\_comanda CHECK ( pret > 0 )

);

### Print-Screen:



The screenshot shows the 'COMANDA' table structure in Oracle SQL Developer. The table has five columns: ID\_COMANDA, ID\_CLIENT, ID\_FACTURA, PRET, and DATA. The first column is a primary key and is not nullable. The other four columns are nullable and have a default value of null. The DATA column has a default value of to\_date(sysdate, 'dd-mm-yy').

	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_produks    NUMBER(10),
id_comanda    NUMBER(20),
numar_produce NUMBER(10),
CONSTRAINT continuit_comanda_pk PRIMARY KEY ( id_produks,
                                             id_comanda ),
CONSTRAINT continuit_produks_fk FOREIGN KEY ( id_produks )
    REFERENCES produks ( id_produks ),
CONSTRAINT continuit_comanda_fk FOREIGN KEY ( id_comanda )
    REFERENCES comanda ( id_comanda ),
CONSTRAINT chk_continuit_comanda CHECK ( numar_produce > 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_PRODUCE	NUMBER(10,0)	Yes	(null)	3	(null)

```
-- PREPARARE --
```

```

CREATE TABLE preparare (
    id_produks    NUMBER(10),
    id_comanda    NUMBER(20),
    id_angajat    NUMBER(10),
    durata        NUMBER(10),
    CONSTRAINT preparare_pk PRIMARY KEY ( id_produks,
                                         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 ),

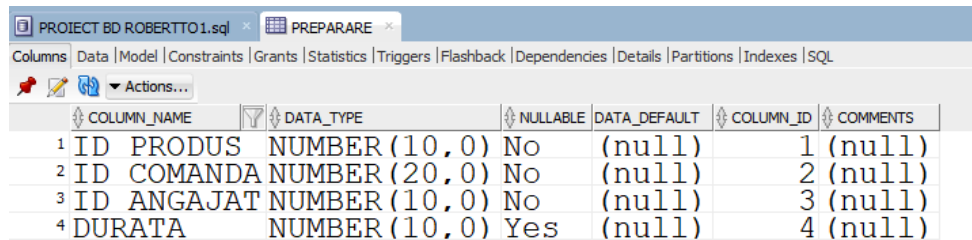
CONSTRAINT preparare_comanda_fk FOREIGN KEY ( id_comanda )

REFERENCES comanda ( id_comanda )

);

```

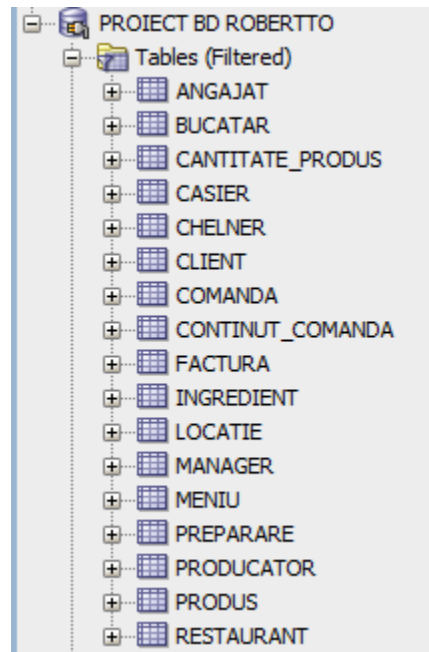
**Print-Screen:**



The screenshot shows the 'PREPARARE' table structure in SQL Developer. The table has four columns: ID\_PRODUS, ID\_COMANDA, ID\_ANGAJAT, and DURATA. The first three columns are of type NUMBER(10,0) and are not nullable, while the last column, DURATA, is of type NUMBER(10,0) and is nullable. The table is part of a database named 'PROJECT BD ROBERTTO'.

	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

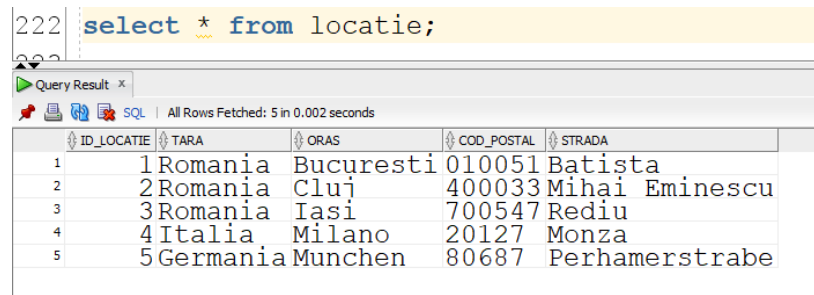
insert into locatie

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 SQL query editor with the command `select * from locatie;` and its results. The results are displayed in a table with 5 rows and 5 columns: ID\_LOCATIE, TARA, ORAS, COD\_POSTAL, and STRADA. The data is as follows:

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

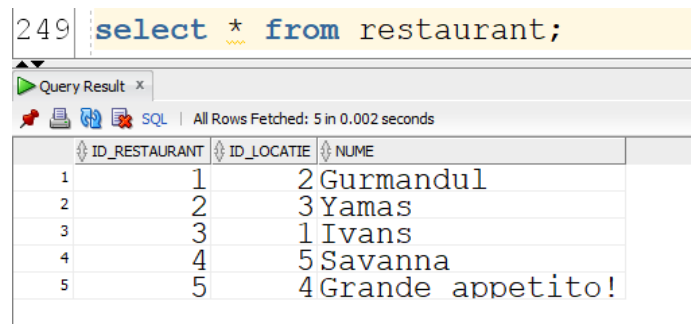
insert into restaurant

values (id\_restaurant.nextval, 5, 'Savanna'); --4

insert into restaurant

values (id\_restaurant.nextval, 4, 'Grande appetito!'); --5

**Print-Screen:**



249 | select \* from restaurant;

Query Result x

All Rows Fetched: 5 in 0.002 seconds

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

-- PENTRU TABELUL MENUU --

-- 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;
```

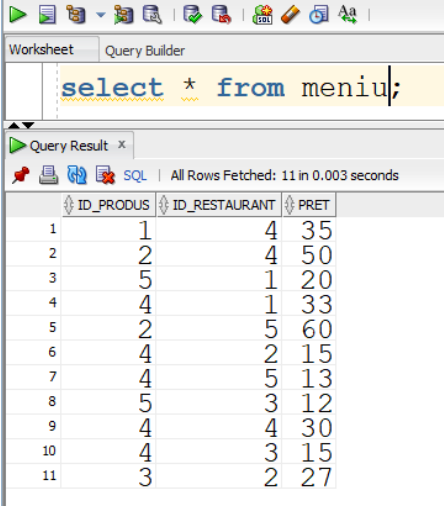
```
insert into meniu (id_restaurant, id_produs, pret)
values (3, 5, 12); --8
```

```
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 a database query interface with a toolbar at the top. Below the toolbar, there are tabs for 'Worksheet' and 'Query Builder'. The 'Query Builder' tab is active, displaying the SQL query: `select * from menu;`. Below the query, there is a 'Query Result' section. It shows a table with 11 rows and 3 columns: `ID_PRODUS`, `ID_RESTAURANT`, and `PRET`. The data is as follows:

	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
```

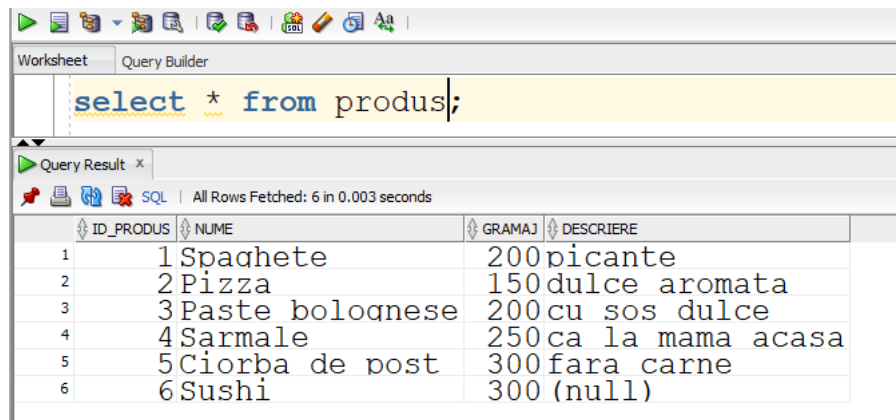
```
insert into produs
```

```
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:**



The screenshot shows a database query builder interface. The top section is labeled 'Query Builder' and contains a text area with the SQL query: `select * from produs;`. Below this is a section labeled 'Query Result' which displays the results of the query. It shows a table with 6 rows and 4 columns: ID\_PRODUS, NUME, GRAMAJ, and DESCRIERE. The data is as follows:

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

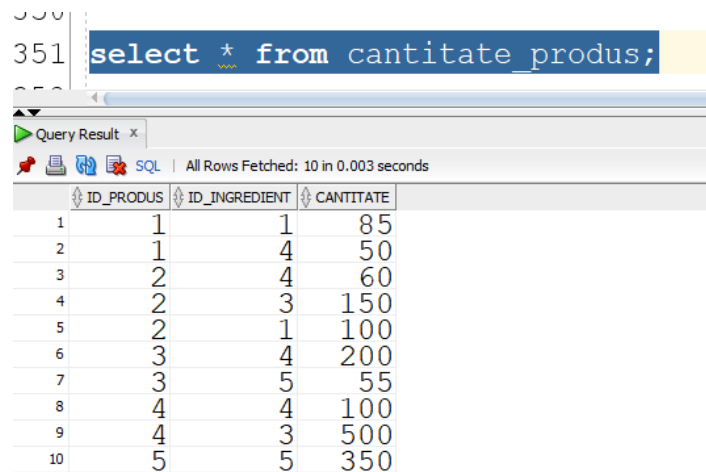
```
insert into cantitate_produs (id_produs, id_ingredient, cantitate)
values (3, 5, 55); --7
```

```
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
```

**Print-Screen:**



The screenshot shows a SQL query editor with the query `select * from cantitate_produs;` and its results. The results are displayed in a table with 10 rows and 3 columns: ID\_PRODUS, ID\_INGREDIENT, and CANTITATE. The data is as follows:

	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

-- 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

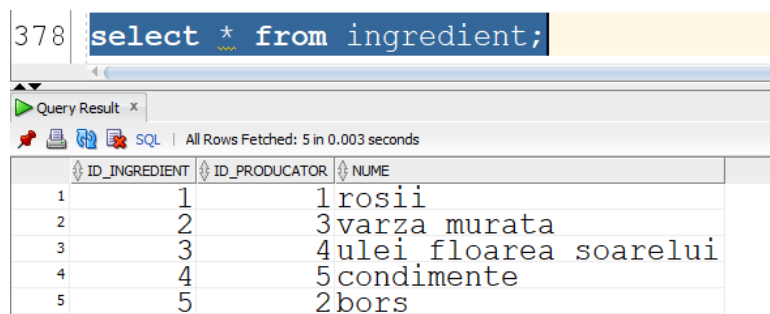
insert into ingredient

values (id\_ingredient.nextval, 5, 'condimente'); --4

insert into ingredient

values (id\_ingredient.nextval, 2, 'bors'); --5

**Print-Screen:**



The screenshot shows a SQL query window with the query `select * from ingredient;` and its results. The results are displayed in a table with 5 rows and 3 columns: `ID_INGREDIENT`, `ID_PRODUCATOR`, and `NUME`. The data is as follows:

	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

-- 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

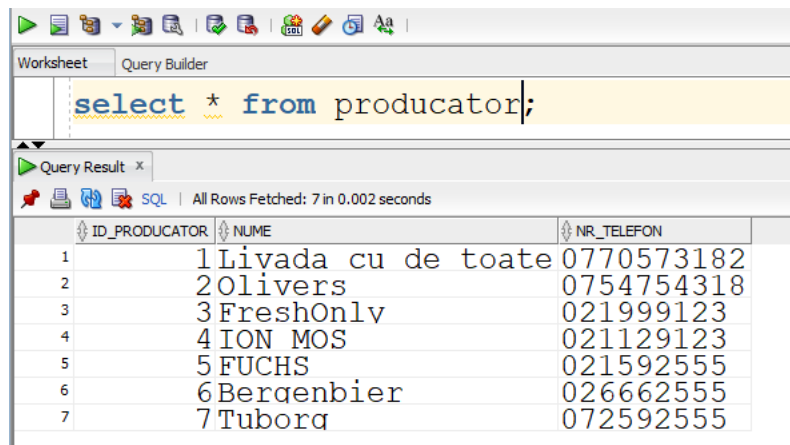
insert into producator

values (id\_producator.nextval, 'Bergenbier', '026662555'); --6

insert into producator

values (id\_producator.nextval, 'Tuborg', '072592555'); --7

### **Print-Screen:**



The screenshot shows a database query tool interface. At the top, there's a toolbar with various icons. Below it, a 'Worksheet' tab is active, displaying a SQL query: `select * from producator;`. Below the query, a 'Query Result' tab is active, showing the results of the query. The results are displayed in a table with three columns: ID\_PRODUCATOR, NUME, and NR\_TELEFON. There are 7 rows of data.

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	Tuborg	072592555

-- PENTRU TABELUL CONTINUT\_COMANDA --

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

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

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

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

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

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

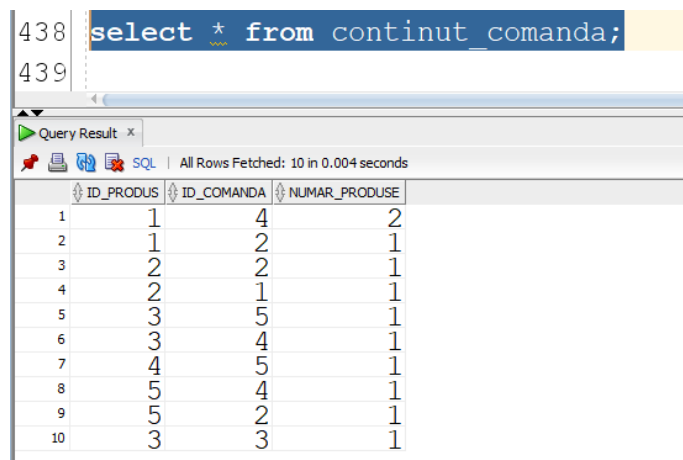
```
insert into continut_comanda (id_produs, id_comanda, numar_produce)
values (4, 5, 1); --7
```

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

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

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

**Print-Screen:**



The screenshot shows a SQL query window with the command `select * from continut_comanda;` and its results. The results are displayed in a table with 10 rows and 3 columns: ID\_PRODUS, ID\_COMANDA, and NUMAR\_PRODUCE. The data is as follows:

	ID_PRODUS	ID_COMANDA	NUMAR_PRODUCE
1	1	4	2
2	1	2	1
3	2	2	1
4	2	1	1
5	3	5	1
6	3	4	1
7	4	5	1
8	5	4	1
9	5	2	1
10	3	3	1

-- 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
```

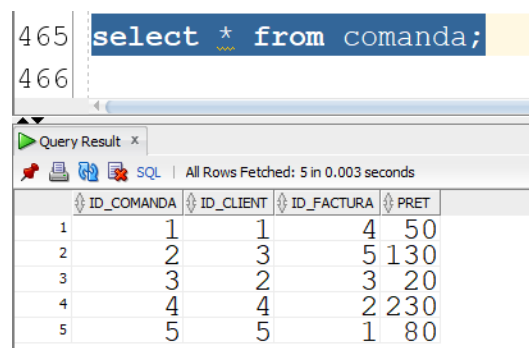
```
insert into comanda (id_comanda, id_client, id_factura, pret)
values (id_comanda.nextval, 3, 5, 130); --2
```

```
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
```

**Print-Screen:**



The screenshot shows a SQL query window with the command `select * from comanda;` and its results. The results are displayed in a table with 5 rows and 4 columns: ID\_COMANDA, ID\_CLIENT, ID\_FACTURA, and PRET. The data is as follows:

ID_COMANDA	ID_CLIENT	ID_FACTURA	PRET
1	1	1	4 50
2	2	3	5 130
3	3	2	3 20
4	4	4	2 230
5	5	5	1 80

-- 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', 'Mircel', '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

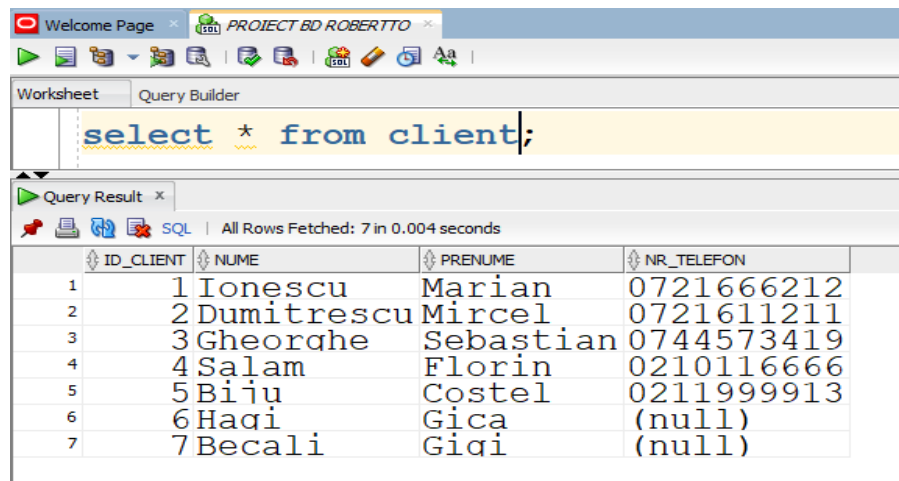
insert into client (id\_client, nume, prenume)

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 a database query tool interface. At the top, there's a 'Welcome Page' and a 'PROJECT BD ROBERTTO' tab. Below the toolbar, the 'Query Builder' tab is active, displaying the SQL query: `select * from client;`. The 'Query Result' tab is also visible, showing the results of the query. The results are displayed in a table with 4 columns: ID\_CLIENT, NUME, PRENUME, and NR\_TELEFON. There are 7 rows of data.

ID_CLIENT	NUME	PRENUME	NR_TELEFON
1	Ionescu	Marian	0721666212
2	Dumitrescu	Mircea	0721611211
3	Gheorghe	Sebastian	0744573419
4	Salam	Florin	0210116666
5	Bişu	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

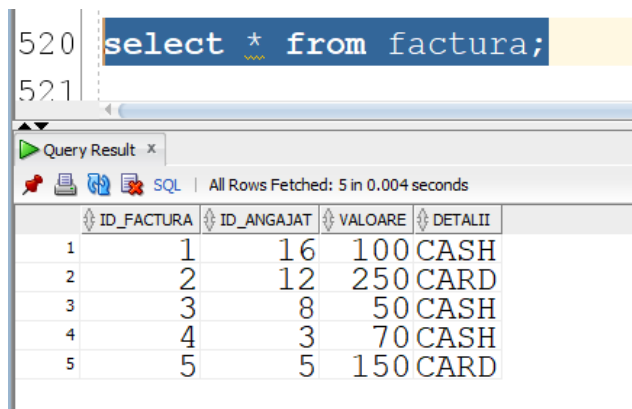
insert into factura (id\_factura, id\_angajat, valoare, detalii)

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 query window with the command `select * from factura;` and its results. The results are displayed in a table with 5 rows and 4 columns: ID\_FACTURA, ID\_ANGAJAT, VALOARE, and DETALII. The data is as follows:

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

-- PENTRU TABELUL CASIER --

```
insert into casier(id_angajat, ani_studiu)
values (3, 10); --1
```

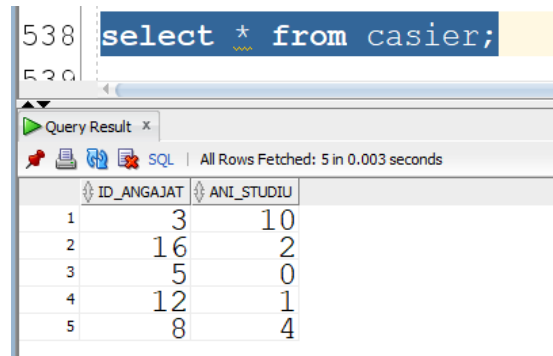
```
insert into casier(id_angajat, ani_studiu)
values (16, 2); --2
```

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

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

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

**Print-Screen:**



The screenshot shows a SQL query window with the query `select * from casier;` and its results. The results are displayed in a table with 5 rows and 2 columns: `ID_ANGAJAT` and `ANI_STUDIU`. The data is as follows:

	ID_ANGAJAT	ANI_STUDIU
1	3	10
2	16	2
3	5	0
4	12	1
5	8	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, nume, prenume)
values (id_angajat.nextval, 1, 'Popescu', 'Robertto'); --1
```

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

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

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

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

```
insert into angajat (id_angajat, id_restaurant, nume, prenume)
values (id_angajat.nextval, 1, 'Voicu', 'Andrei'); --6
```

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

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

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

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

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

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

```
values (id_angajat.nextval, 3, 'Bida', 'Marian', to_date('25-02-21', 'dd-mm-yy')); --12
```

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

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

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

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

```
insert into angajat (id_angajat, id_restaurant, nume, prenume, data_angajare)  
values (id_angajat.nextval, 1, 'Dumitrescu', 'Florin', to_date('04-03-21', 'dd-mm-yy')); --17
```

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

```
insert into angajat (id_angajat, id_restaurant, nume, prenume, data_angajare)  
values (id_angajat.nextval, 5, 'Scarlatescu', 'Catalin', to_date('01-03-21', 'dd-mm-yy')); --19
```

```
insert into angajat (id_angajat, id_restaurant, nume, prenume, data_angajare)  
values (id_angajat.nextval, 2, 'Bontea', 'Sorin', to_date('02-03-21', 'dd-mm-yy')); --20
```

### Print-Screen:

610 `select * from angajat;`

Query Result x

All Rows Fetched: 20 in 0.004 seconds

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

-- PENTRU TABELUL PREPARARE --

-- durata este masurata in minute --

insert into preparare (id\_produș, id\_comanda, id\_angajat, durata)

values (1, 1, 17, 60); --1

insert into preparare (id\_produș, id\_comanda, id\_angajat, durata)

values (1, 2, 9, 45); --2

insert into preparare (id\_produș, id\_comanda, id\_angajat, durata)

values (2, 4, 19, 35); --3

insert into preparare (id\_produș, 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
```

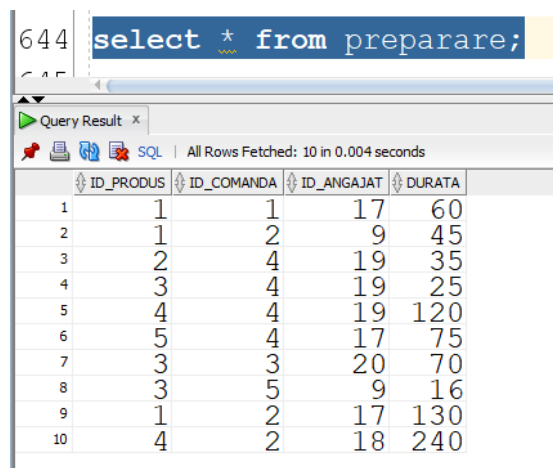
```
insert into preparare (id_produs, id_comanda, id_angajat, durata)
values (3, 3, 20, 70); --7
```

```
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:**



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

-- PENTRU TABELUL BUCATAR --

```
insert into bucatar (id_angajat, nr_steie)
values (9, 1); --1
```

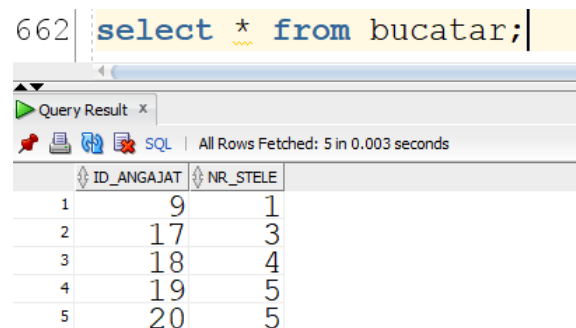
```
insert into bucatar (id_angajat, nr_steie)
values (17, 3); --2
```

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

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

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

**Print-Screen:**



The screenshot shows a SQL query editor with the command `select * from bucatar;` and its results. The results are displayed in a table with two columns: `ID_ANGAJAT` and `NR_STEIE`. The table contains five rows of data, corresponding to the insert statements above.

	ID_ANGAJAT	NR_STEIE
1	9	1
2	17	3
3	18	4
4	19	5
5	20	5

-- PENTRU TABELUL CHELNER --

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

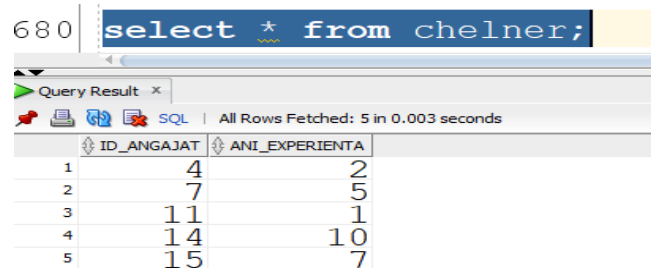
```
insert into chelner (id_angajat, ani_experienta)
values (7, 5); --2
```

```
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:**



The screenshot shows a SQL query editor with the query `select * from chelner;` and its results. The results are displayed in a table with two columns: `ID_ANGAJAT` and `ANI_EXPERIENTA`. The table contains 5 rows of data.

	ID_ANGAJAT	ANI_EXPERIENTA
1	4	2
2	7	5
3	11	1
4	14	10
5	15	7

-- PENTRU TABELUL MANAGER --

```
insert into manager (id_angajat)
values (1); --1
```

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

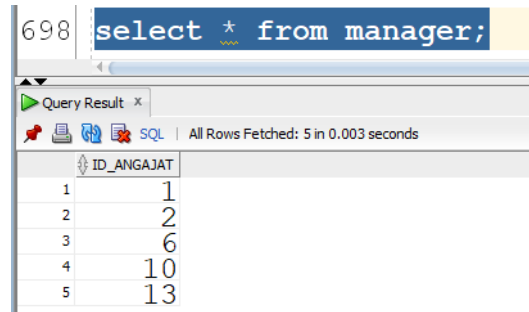
```
insert into manager (id_angajat)
values (6); --3
```

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

insert into manager (id\_angajat)

values (13); --5

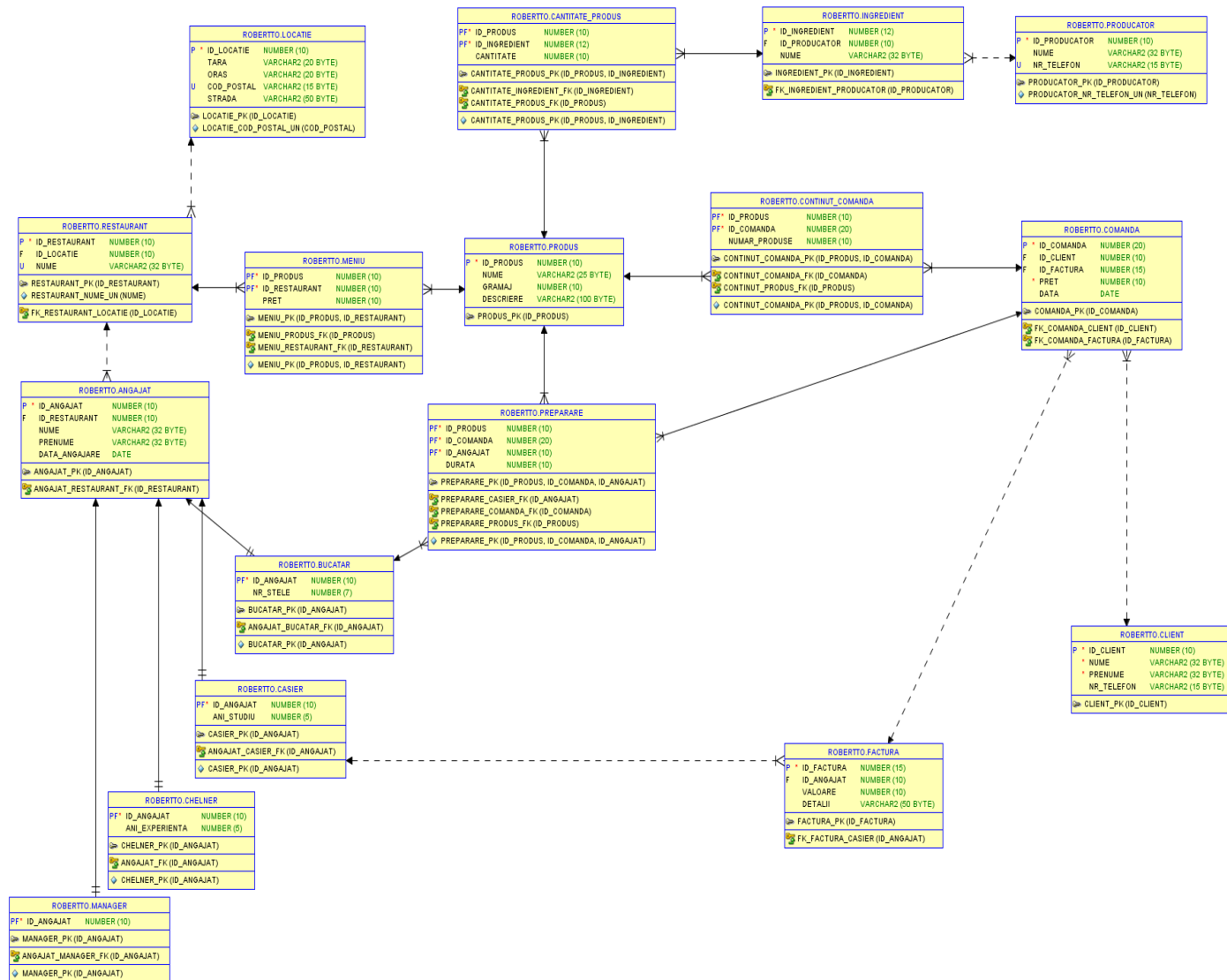
**Print-Screen:**



The screenshot shows a SQL query editor with the query `select * from manager;` entered. Below the editor, a 'Query Result' window displays the results of the query. The results are shown in a table with 5 rows and 1 column named 'ID\_ANGAJAT'. The values in the column are 1, 2, 6, 10, and 13. The status bar indicates 'All Rows Fetched: 5 in 0.003 seconds'.

ID_ANGAJAT
1
2
6
10
13

### 6.2 Diagrama generată în sql după crearea tabelelor și inserarea datelor.



## 7. Crearea a 5 cereri complexe în SQL:

-- CERINTA 11 --

-- 1) Sa se afiseze numele si id-ul tuturor produselor comandate de un client vreodata.

SELECT DISTINCT

cl.id\_client,

cl.nume,

cl.prenume,

prod.id\_produs,

prod.nume

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:

```
5 -- 1) Sa se afiseze numele si id-ul tuturor produselor comandate de un client vreodata.
6 SELECT DISTINCT
7     cl.id_client,
8     cl.numa,
9     cl.prenume,
10    prod.id_produ,
11    prod.numa
12 FROM
13     comanda          c,
14     client            cl,
15     continut_comanda cmd,
16     produs            prod
17 WHERE
18     cl.id_client = c.id_client
19     AND c.id_comanda = cmd.id_comanda
20     AND cmd.id_produ = prod.id_produ
21 ORDER BY
22     id_client,
23     id_produ;
```

Query Result x

All Rows Fetched: 10 in 0.004 seconds

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

-- 2) Pentru fiecare bucatar angajat in anul curent, afisati profitul mediu pe care il poate aduce daca ar vinde

-- din fiecare mancare pe care stie sa o prepare exact o bucata.

SELECT

a.id\_angajat,

lower(a.numa),

lower(a.prenume),

r.id\_restaurant,

upper(r.numa),

AVG(m.pret) "castig mediu"

FROM

angajat a,  
bucatar b,  
preparare prep,  
produs prod,  
menu 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:

```
25 -- 2) Pentru fiecare bucatar angajat in anul curent, afisati profitul mediu pe care il poate aduce daca ar vinde
26 -- din fiecare mancare pe care stie sa o prepare exact o bucata.
27 SELECT
28     a.id_angajat,
29     lower(a.num),
30     lower(a.prenume),
31     r.id_restaurant,
32     upper(r.num),
33     AVG(m.pret) "castig_mediu"
34 FROM
35     angajat      a,
36     bucatar      b,
37     preparare    prep,
38     produs       prod,
39     meniu        m,
40     restaurant   r
41 WHERE
42     a.id_restaurant = r.id_restaurant
43     AND a.id_angajat = b.id_angajat
44     AND b.id_angajat = prep.id_angajat
45     AND prod.id_produs = prep.id_produs
46     AND prod.id_produs = m.id_produs
47     AND m.id_restaurant = r.id_restaurant
48     AND to_char(a.data_angajare, 'yyyy') LIKE to_char(sysdate, 'yyyy')
49 GROUP BY (
50     a.id_angajat,
51     a.num,
52     a.prenume,
53     r.id_restaurant,
54     r.num
55 );
```

Query Result: X | All Rows Fetched: 3 in 0.273 seconds

ID_ANGAJAT	LOWER(A.NUM)	LOWER(A.PRENUME)	ID_RESTAURANT	UPPER(R.NUM)	castig_mediu
17	dumitrescu	florin	1	GURMANDUL	20
20	bontea	sorin	2	YAMAS	27
19	scarlatescu	catalin	5	GRANDE APPETITO!	36.5

-- 3) Afisati pentru fiecare produs, id-ul bucatarului care il prepara si cate stele are bucatarul

-- sau mesaj daca nu e preparat de nimeni

-- daca e preparat de mai multi, ii afisati pe toti.

-- 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:

```

57 -- 3) Afisati pentru fiecare produs, id-ul bucatarului care il prepara si cate stele are bucatarul
58 -- sau mesaj daca nu e preparat de nimeni
59 -- daca e preparat de mai multi, ii afisati pe toti.
60 -- Nesicronizata
61 with myProd as (
62     select distinct prod.ID_PRODUS, prod.NUME, prep.ID_ANGAJAT
63     from produs prod,
64         PREPARARE prep
65     where prod.ID_PRODUS = prep.ID_PRODUS(+)
66 select myProd.ID_PRODUS,
67     myProd.NUME,
68     decode(nvl(om.ID_ANGAJAT, -1), -1, 'nu e preparat de nimeni', om.id_angajat) preparator,
69     om.NR_STELE,
70     om.ID_RESTAURANT
71 from myProd,
72     (select a.ID_ANGAJAT, a.NUME, a.PRENUME, b.NR_STELE, r.ID_RESTAURANT
73     from ANGAJAT a,
74         BUCATAR b,
75         RESTAURANT r
76     where a.ID_ANGAJAT = b.ID_ANGAJAT
77     and a.ID_RESTAURANT = r.ID_RESTAURANT) om
78 where myProd.ID_ANGAJAT = om.ID_ANGAJAT(+);
79
80

```

Query Result: x Query Result 1 x

All Rows Fetched: 10 in 0.005 seconds

ID_PRODUS	NUME	PREPARATOR	NR_STELE	ID_RESTAURANT
1	3 Paste boloanese	9	1	5
2	1 Spachete	9	1	5
3	1 Spachete	17	3	1
4	5 Ciorba de post	17	3	1
5	4 Sarmale	18	4	3
6	3 Paste boloanese	19	5	5
7	2 Pizza	19	5	5
8	4 Sarmale	19	5	5
9	3 Paste boloanese	20	5	2
10	6 Sushi	nu e preparat de nimeni (null)	(null)	

```

--- 4) Afisati toti clientii cu toate comenzile lor si facturile aferente,
--   a caror suma cheltuita pana acum depaseste average-ul comenzilor cu cel putin 3 produs
-- 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:**

```

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

```

ID_CLIENT	NUME	PRENUME	NR_TELEFON	ID_COMANDA	ID_CLIENT_1	ID_FACTURA	PRET	DATA	ID_FACTURA_1	ID_ANGAJAT	VALOARE	DETALII
1	4	Salam Florin	0210116666	4	4	2	230	23-MAY-21	2	12	250	CARD

-- 5) Pentru fiecare pereche de angajati, afisati daca acestia se cunosc de cel putin 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;

-- CERINTA 12 --

-- 1) suprimare

DELETE FROM producator

WHERE nume IN ( SELECT nume

FROM producator

where upper(nume) like '%BERGENBIER%'

);

**Print-Screen:**

```
103 | -- 5) Pentru fiecare pereche de angajati, afisati daca acestia se cunosc de cel putin un an.  
104 | select case  
105 |     when months_between(a1.DATA_ANGAJARE, a2.DATA_ANGAJARE) >= 12 then 'Angajatii ' || a1.ID_ANGAJAT || ' si ' ||  
106 |                                     a2.ID_ANGAJAT ||  
107 |                                     ' se cunosc de peste un an'  
108 |     else 'Angajatii ' || a1.ID_ANGAJAT || ' si ' || a2.ID_ANGAJAT || ' se cunosc de mai putin de un an' end  
109 | from ANGAJAT a1,  
110 |      ANGAJAT a2  
111 | where a1.ID_ANGAJAT < a2.ID_ANGAJAT;
```

```
Query Result: 1 x | Query Output: x | Query Result: 1 x | Query Result: 2 x  
Printed 20 rows in 0.004 seconds  
1 | 1 | 2 | se cunosc de mai putin de un an  
1 | 1 | 3 | se cunosc de peste un an  
1 | 1 | 4 | se cunosc de mai putin de un an  
1 | 1 | 5 | se cunosc de peste un an  
1 | 1 | 6 | se cunosc de mai putin de un an  
1 | 1 | 7 | se cunosc de peste un an  
1 | 1 | 8 | se cunosc de peste un an  
1 | 1 | 9 | se cunosc de mai putin de un an  
1 | 1 | 10 | se cunosc de mai putin de un an  
1 | 1 | 11 | se cunosc de mai putin de un an  
1 | 1 | 12 | se cunosc de mai putin de un an  
1 | 1 | 13 | se cunosc de mai putin de un an  
1 | 1 | 14 | se cunosc de mai putin de un an  
1 | 1 | 15 | se cunosc de mai putin de un an  
1 | 1 | 16 | se cunosc de mai putin de un an  
1 | 1 | 17 | se cunosc de mai putin de un an  
1 | 1 | 18 | se cunosc de mai putin de un an  
1 | 1 | 19 | se cunosc de mai putin de un an  
1 | 1 | 20 | se cunosc de mai putin de un an  
2 | 2 | 1 | se cunosc de peste un an  
2 | 2 | 4 | se cunosc de mai putin de un an  
2 | 2 | 5 | se cunosc de peste un an  
2 | 2 | 6 | se cunosc de mai putin de un an  
2 | 2 | 7 | se cunosc de mai putin de un an  
2 | 2 | 8 | se cunosc de peste un an  
2 | 2 | 9 | se cunosc de mai putin de un an  
2 | 2 | 10 | se cunosc de mai putin de un an  
2 | 2 | 11 | se cunosc de mai putin de un an  
2 | 2 | 12 | se cunosc de mai putin de un an
```

## 8. Implementarea a 3 operații de actualizare sau suprimare a datelor utilizând subcereri:

-- CERINTA 12 --

-- 1) suprimare

delete from producator

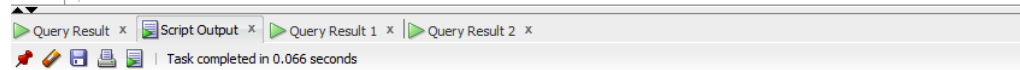
where nume in (select nume

from producator

where upper(nume) like '%BERGENBIER%');

**Print-Screen:**

```
113 -- CERINTA 12 --
114 -- 1) suprimare
115 delete from producator
116 where nume in (select nume
117                 from producator
118                 where upper(nume) like '%BERGENBIER%');
119
```



1 row deleted.

-- 2) update

update client

set nr\_telefon = '666'

where nr\_telefon is null

and nume = (select c1.nume from client c1 where lower(c1.nume) like '%becali%');

select \* from client;

### **Print-Screen:**

```
120 -- 2) update
121 update client
122 set nr_telefon = '666'
123 where nr_telefon is null
124 and nume = (select c1.nume from client c1 where lower(c1.nume) like '%becali%');
125
```

Query Result x Script Output x Query Result 1 x Query Result 2 x  
Task completed in 0.069 seconds

1 row updated.

-- 3) update

update produs

set gramaj = (select avg(gramaj) from produs)

where descriere is null;

### **Print-Screen:**

```
126 -- 3) update
127 update produs
128 set gramaj = (select avg(gramaj) from produs)
129 where descriere is null;
130
```

Query Result x Script Output x Query Result 1 x Query Result 2 x  
Task completed in 0.058 seconds

1 row updated.

## **9. O cerere care utilizează operația outer-join pe minium 4 tabele și două cereri ce utilizează operația division:**

-- CERINTA 16

-- Folosim outer join aici

-- Afisati pentru toate produsele din sistem, la ce restaurant se servesc, sau null daca nu e servit nicaieri.

```

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:

```

131 -- CERINTA 16
132 -- Folosim outer join aici
133 -- Afisati pentru toate produsele din sistem, la ce restaurant se servesc, sau null daca
134 select p.ID_PRODUS, p.NUME, m.ID_RESTAURANT, r.NUME, l.TARA, l.ORAS, l.STRADA
135 from meniu m, PRODUS p, RESTAURANT r, locatie l
136 where m.ID_PRODUS(+) = p.ID_PRODUS
137 and m.ID_RESTAURANT = r.ID_RESTAURANT(+)
138 and r.ID_RESTAURANT = l.ID_LOCATIE(+); --outer join

```

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 bolonnese	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)

-- Afisati produsele care se gasesc in toate restaurantele.

-- Folosim division

```
select distinct p.ID_PRODUS, p.NUME
```

```
from PRODUS p, MENIU m
```

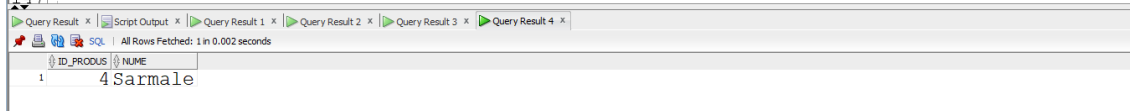
```
where p.ID_PRODUS in (select distinct m.ID_PRODUS from MENIU m) and m.ID_PRODUS = p.ID_PRODUS
```

```
group by p.ID_PRODUS, p.NUME
```

```
having count(p.ID_PRODUS) = (select count(r1.ID_RESTAURANT) from RESTAURANT r1);
```

## Print-Screen:

```
140 -- Afisati produsele care se gasesc in toate restaurantele.
141 -- Folosim division
142 select distinct p.ID_PRODUS,p.NUME
143 from PRODUS p, MENIU m
144 where p.ID_PRODUS in (select distinct m.ID_PRODUS from MENIU m) and m.ID_PRODUS=p.ID_PRODUS
145 group by p.ID_PRODUS, p.NUME
146 having count(p.ID_PRODUS) = (select count(r1.ID_RESTAURANT) from RESTAURANT r1);
147
```



ID_PRODUS	NUME
4	Sarmale

-- 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('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:

```
148 -- Afisati produsele care se gasesc in toate comenzile dintr-o anumita zi.
149 -- Folosim Division
150 insert into FACTURA values(1000,3,100,'nu');
151 insert into COMANDA values (1000,1,1000,1000,to_date('13-07-2000','dd-mm-yyyy'));
152 insert into CONTINUT_COMANDA values (4,1000,3);
153
154 with t as(select * from CONTINUT_COMANDA cnt0, COMANDA cmd0 where cnt0.ID_COMANDA=cmd0.ID_COMANDA)
155 select distinct p.ID_PRODUS, p.NUME
156 from PRODUS p, t t1
157 where p.ID_PRODUS=t1.ID_PRODUS
158 and to_char(t1.DATA,'dd-mm-yyyy')='13-07-2000'
159 and p.ID_PRODUS in (select distinct t2.ID_PRODUS from t t2
160                     where to_char(t2.DATA, 'dd-mm-yyyy')='13-07-2000')
161 group by p.ID_PRODUS, p.NUME
162 having count(p.ID_PRODUS)=(select count(*) from t t3
163                           where to_char(t3.DATA, 'dd-mm-yyyy')='13-07-2000');
```



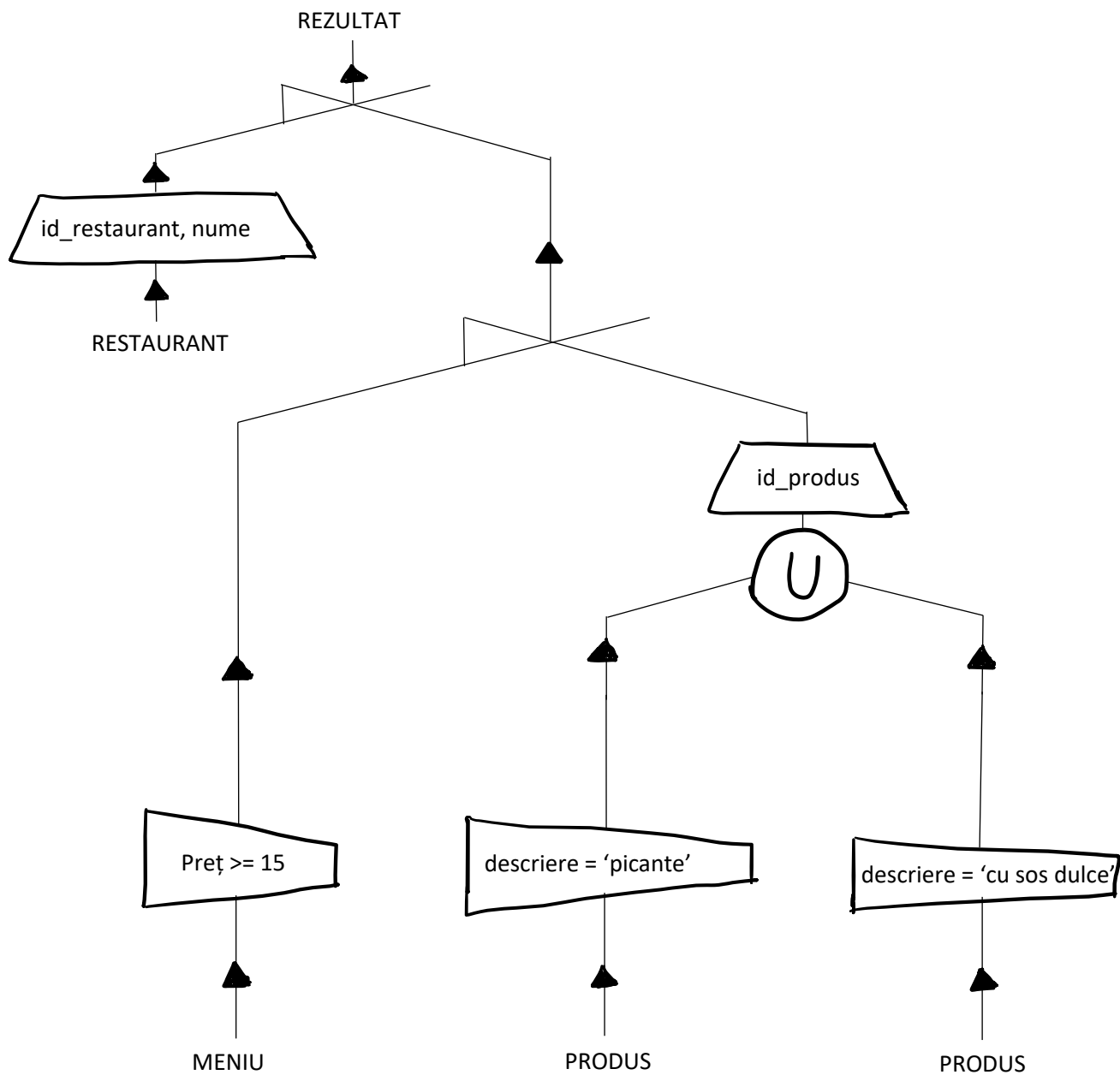
ID_PRODUS	NUME
4	Sarmale

**10. Optimizarea unei cereri, aplicând regulile de optimizare ce derivă din proprietățile operatorilor algebrei relaționale. Cererea va fi exprimată prin expresie algebrică, arbore algebric și limbaj (SQL), atât anterior cât și ulterior optimizării:**

### Cerința:

*Să se afișeze codul și numele restaurantelor care au în meniu prețuri mai mari sau egale cu 15 lei la produse care sunt picante sau cu sos dulce.*

- Mai jos avem arborele neoptimizat:



### ALGEBRA RELATIONALĂ:

R1 = SELECT (PRODUS, descriere = 'picante');

R2 = SELECT (PRODUS, descriere = '%cu%sos%dulce%');

R3 = UNION (R1, R2);

```

R4 = PROJECT (R3, id_produs);
R5 = SELECT (MENIU, Pret >= 15);
R6 = SEMIJOIN (R5, R4);
R7 = SEMIJOIN (RESTAURANT, R6);
REZULTAT = PROJECT (R7, id_restaurant);

```

### **SQL:**

with

```

R1 as (select * from produs where descriere like '%picante%'),
R2 as (select * from produs where descriere like '%cu%sos%dulce%'),
R3 as (select * from R1 union select * from R2),
R4 as (select id_produs from R3),
R5 as (select * from meniu where pret >= 15),
R6 as (select * from R4, R5 where R4.id_produs = R5.id_produs),
R7 as (select t2.id_restaurant, t2.numa from R6 t1, restaurant t2 where t1.id_restaurant =
t2.id_restaurant)
select R7.id_restaurant, R7.numa from R7;

```

### **Print-Screen:**

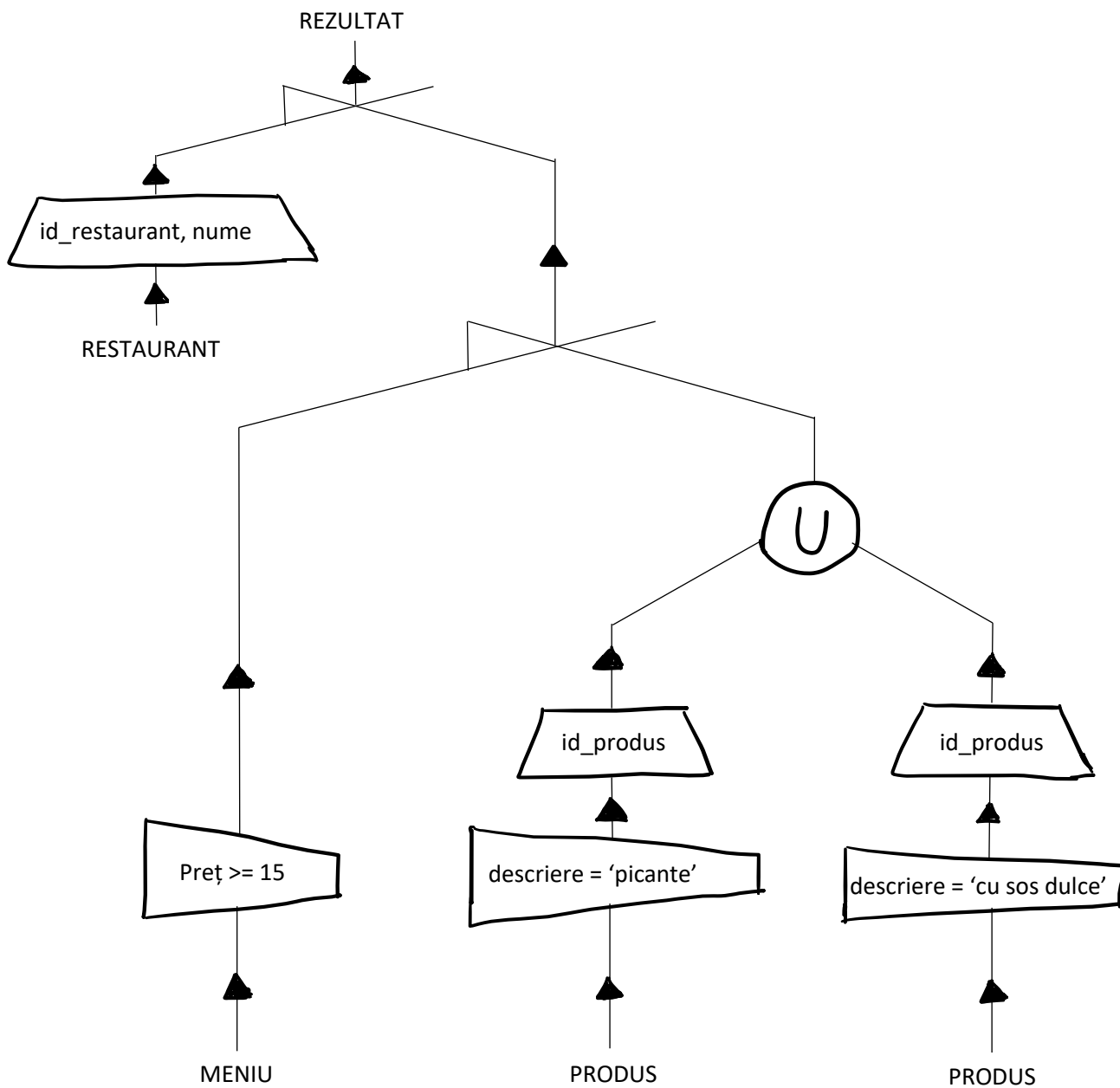
```

167 |-- CERINTA 17 --
168 |-- neoptimizata
169 |with
170 |R1 as (select * from produs where descriere like '%picante%'),
171 |R2 as (select * from produs where descriere like '%cu%sos%dulce%'),
172 |R3 as (select * from R1 union select * from R2),
173 |R4 as (select id_produs from R3),
174 |R5 as (select * from meniu where pret >= 15),
175 |R6 as (select * from R4, R5 where R4.id_produs = R5.id_produs),
176 |R7 as (select t2.id_restaurant, t2.numa from R6 t1, restaurant t2 where t1.id_restaurant = t2.id_restaurant)
177 |select R7.id_restaurant, R7.numa from R7;

```

ID_RESTAURANT	NUMA
1	4 Savanna
2	2 Yamas

- Mai jos avem arborele optimizat



**ALGEBRA RELATIONALĂ:**

R1 = SELECT (PRODUS, descriere = 'picante');

R2 = SELECT (PRODUS, descriere = '%cu%sos%dulce%');

R3 = PROJECT (R1, id\_produș);

```

R4 = PROJECT (R2, id_produs);
R5 = UNION (R4, R3);
R6 = SELECT (MENIU, Pret >= 15);
R7 = SEMIJOIN (R5, R6);
R8 = PROJECT (RESTAURANT, id_restaurant, nume);
REZULTAT = SEMIJOIN (R7, R8);

```

### SQL:

with

```

R1 as (select * from produs where descriere like '%picante%'),
R2 as (select * from produs where descriere like '%cu%sos%dulce%'),
R3 as (select id_produs from R1),
R4 as (select id_produs from R2),
R5 as (select * from R3 union select * from R4),
R6 as (select * from meniu where pret >= 15),
R7 as (select * from R5, R6 where R5.id_produs = R6.id_produs),
R8 as (select id_restaurant, nume from restaurant)
select R8.id_restaurant, R8.nume from R7, R8 where R8.id_restaurant = R7.id_restaurant;

```

### Print-Screen:

```

179 | -- optimizata
180 | with
181 | R1 as (select * from produs where descriere like '%picante%'),
182 | R2 as (select * from produs where descriere like '%cu%sos%dulce%'),
183 | R3 as (select id_produs from R1),
184 | R4 as (select id_produs from R2),
185 | R5 as (select * from R3 union select * from R4),
186 | R6 as (select * from meniu where pret >= 15),
187 | R7 as (select * from R5, R6 where R5.id_produs = R6.id_produs),
188 | R8 as (select id_restaurant, nume from restaurant)
189 | select R8.id_restaurant, R8.nume from R7, R8 where R8.id_restaurant = R7.id_restaurant;

```

ID_RESTAURANT	NUME
1	4 Savanna
2	2 Yamas

## 11. a) Realizarea normalizării BCNF, FN4, FN5:

Schema noastră ar fi scoasă din BCNF întrucât în tabelul meniu, #id\_restaurant și #id\_produs determină prețul. Presupunem pentru modelul nostru că prețul determină #id\_produs.

Tabelul Meniu nu este în BCNF, deci aplicăm regula *Casey-Delobel* și atunci rezultă că:

- **MENIU1** (#Preț, id\_produs)
- **MENIU2** (#id\_restaurant, Preț)

Un alt caz ar putea fi atunci când #id\_produs determină preț, tragem concluzia că nu este în BCNF. Aplicăm regula *Casey-Delobel* și atunci rezultă că:

- **MENIU1** (#id\_produs, preț)
- **MENIU2** (#id\_produs, #id\_restaurant)

Observăm că aici se păstrează dependențele funcționale spre deosebire de prima afirmație.

### Schema Non-BCNF:

**MENIU** (#ID-RESTAURANT, #ID-PRODUS, Preț)

Schema noastră ar fi scoasă din FN4 întrucât în tabelul preparare, #id\_comandă ->-> (multidetermină) #id\_produs, de unde rezultă #id\_comandă ->-> (multidetermină) #id\_bucătar. O aducem în FN4:

- **PREPARARE1** (#id\_comandă, #id\_produs, Durată)
- **PREPARARE2** (#id\_comandă, #id\_bucătar)

### Schema Non-FN4:

**PREPARARE** (#ID-BUCĂTAR, #ID-COMANDĂ, #ID-PRODUS, Durată)

Un bucătar poate prepara mai multe produse care aparțin aceleiași comenzi sau poate prepara un produs pentru o comandă într-o perioadă de timp diferită. Presupunem că mai mulți bucătari pot prepara același produs, în aceeași perioadă de timp sau în perioade de timp diferite.

Datorită dependențelor precizate anterior, relația nu va mai fi în FN5. Ea se poate desface prin proiecție în trei relații:

- **PREPARARE1** (#id\_bucătar, #id\_comandă, #id\_produs)
- **PREPARARE2** (#id\_bucătar, Durată)
- **PREPARARE3** (#id\_comandă, #id\_produs, Durată)

Putem observa foarte clar:

PREPARARE != JOIN (PREPARARE1, PREPARARE2)

PREPARARE != JOIN (PREPARARE1, PREPARARE3)

PREPARARE != JOIN (PREPARARE2, PREPARARE3)

PREPARARE = JOIN (JOIN(PREPARARE1, PREPARARE2), PREPARARE3)

### **Schema Non-FN5:**

**PREPARARE** (#ID-COMANDĂ, #ID-PRODUS, #ID-BUCĂȚAR, Durată)

## **b) Aplicarea denormalizării:**

Tabelul LOCAȚIE (#id\_locație, Țara, Oraș, Cod\_poștal, Strada)

Această relație nu este în FN3 întrucât cod\_poștal -> (determină) {Oraș, Țara}. Ea este mai exact în FN2.

Dacă aplicăm FN3 vom obține:

- LOCAȚIE1 (#id\_locație, Strada, Cod\_poștal)
- COD\_P (#cod\_poștal, Oraș, Țara)

Acest lucru nu este convenabil, deoarece rareori vom accesa adresa unui restaurant fără informații la Țară și Oraș. Din acest motiv am decis să nu folosesc forma FN3 pentru tabelul LOCAȚIE, reducem numărul de join-uri care nu sunt necesare.

Un alt exemplu ar putea fi acela pentru tabelul FACTURĂ, care conține #id\_factură, valoare, id\_casier, detalii. Acesta să conțină și nume\_chelner, respectiv prenume\_chelner, pentru a avea mai multe detalii despre angajatul(chelnerul) care a fost atașat la o factură. Această condiție ar reduce numărul de join-uri pentru a afla numele și prenumele chelnerului. Presupunem că nu ne interesează decât numele și prenumele acestuia. (Să se afișeze numele și prenumele chelnerului care a fost atașat la factura cu cea mai mare valoare.)