

ONUTU RADU-CONSTANTIN  
GRUPA 212

# CUPRINS:

## Contents

EX1.....	3
EX2.....	3
EX3.....	3
EX4.....	4
EX5.....	5
EX6.....	7
EX7.....	8
EX8.....	9
EX9.....	9
EX10.....	10
EX11.....	11
EX12.....	23
EX13.....	27

## EX1

Baza mea de date contine informatii despre un lant de pizzerii. Fiecare pizzerie are cate un singur patron care este nascut intr-un singur oras. Nu pot exista meniuri asemanatoare in pizzerii. Fiecare pizzerie are cate o echipa cu angajati diferiti.

## EX2

Reguli:

- Echipa este unica pentru fiecare pizzerie;
- Angajatii sunt unici pentru fiecare echipa;
- Un patron poata sa aiba mai multe pizzerii;
- Patronul este nascut intr-un singur oras;
- Fiecare meniu e diferit pentru fiecare pizzerie

## EX3

Pizzerie = contine numele pizzeriei. Cheia primara este id\_pizzerie.

Meniu = contine numele categoriilor de mancare pe care le poti comanda si pretul. Cheia primara este id\_meniu.

Portie = contine numele portiilor si gramajul lor. Cheia primara este id\_portie.

Departament = contine numele departamentelor. Cheia primara este id\_departament.

Echipa = contine numele rolurilor pe care le poate avea un angajat. Cheia primara este id echipa.

Angajat = contine numele, varsta si sexul fiecarui angajat. Cheia primara este id\_angajat.

Patron = contine numele fiecarui patron. Cheia primara este id\_patron.

Oras = contine numele orasului in care este nascut patronul si numarul de locuitori al acestuia. Cheia primara este id\_oras.

Tara = contine numele tarii si suprafata acesteia. Cheia primara este id\_tara.

## EX4

Pizzeria --- Meniu = One to One. O pizzeria are un singur meniu, un meniu e diferit pentru fiecare pizzeria.

Meniu --- Portie = Many to Many. Un meniu are mai multe portii, o portie se regaseste in mai multe meniuri.

Pizzeria --- Departament = Many to Many. O pizzeria are mai multe departamente, un departament apare in mai multe pizzerii.

Departament --- Echipa = One to Many. Un departament are mai multe echipe, o echipa face parte doar dintr-un singur departament.

Echipa --- Angajat = One to Many. O echipa are mai multi angajati, un angajat face parte doar dintr-o echipa.

Pizzeria --- Patron = Many to One. O pizzeria este condusa doar de un patron, un patron poate sa detina mai multe pizzerii.

Patron --- Oras = Many to One. Un patron provine dintr-un singur oras, dintr-un oras pot sa provina mai multi patroni.

Oras --- Tara = Many to One. Un oras poate sa apara doar intr-o singura tara, o tara contine mai multe orase.

## EX5

Pizzeria:

- id\_pizzeria INT
- nume CHAR (numele pizzeriei)

Meniu:

- id\_meniu INT
- denumire CHAR (numele categoriei de mancare)
- pret INT (pretul unei farfurii)
- id\_pizzeria INT
- id\_portie INT

Portie:

- id\_portie INT
- nume CHAR (numele portiei)
- gramaj INT (gramajul unei portii)

Departament:

- id\_departament INT
- nume CHAR (numele departamentului)
- id\_pizzeria INT
- id echipa INT

Departament\_Echipa\_mapping:

- id\_departament INT
- id echipa INT

Echipa:

- id echipa INT
- rol CHAR (numele rolului)
- id departament INT
- id angajat INT

#### Angajat:

- id\_angajat INT
- nume CHAR (numele unui angajat)
- varsta INT (varsta unui angajat; Restrictie: min. 16, max. 150)
- sex CHAR (sexul unui angajat; Restrictie: M/F)

#### Patron:

- id\_patron INT
- nume CHAR (numele patronului)
- id\_pizzeria INT
- id\_oras INT

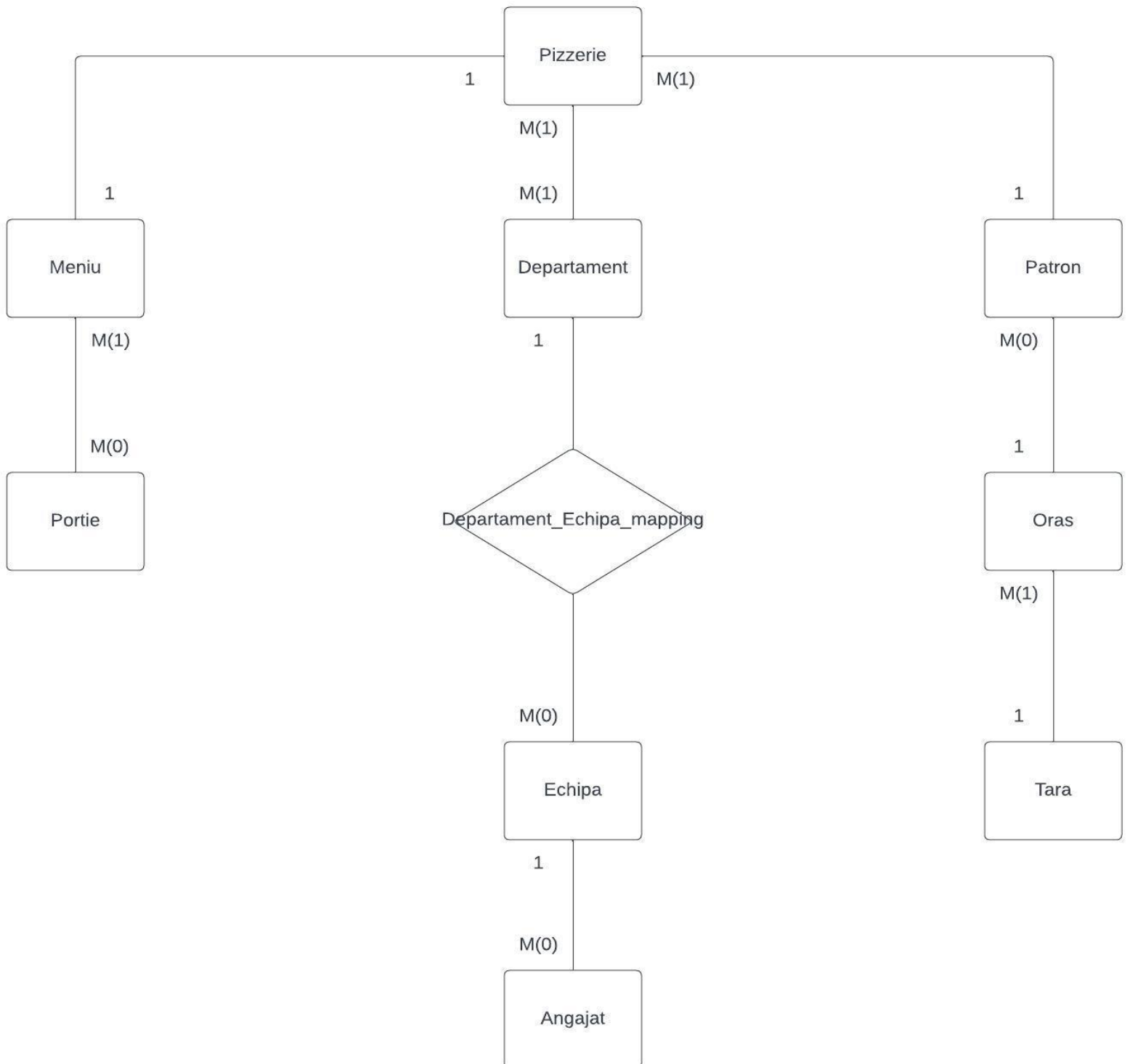
#### Oras:

- id\_oras INT
- nume CHAR (numele orasului)
- nr\_locuitori INT (numarul de locuitori al orasului)
- id\_tara INT

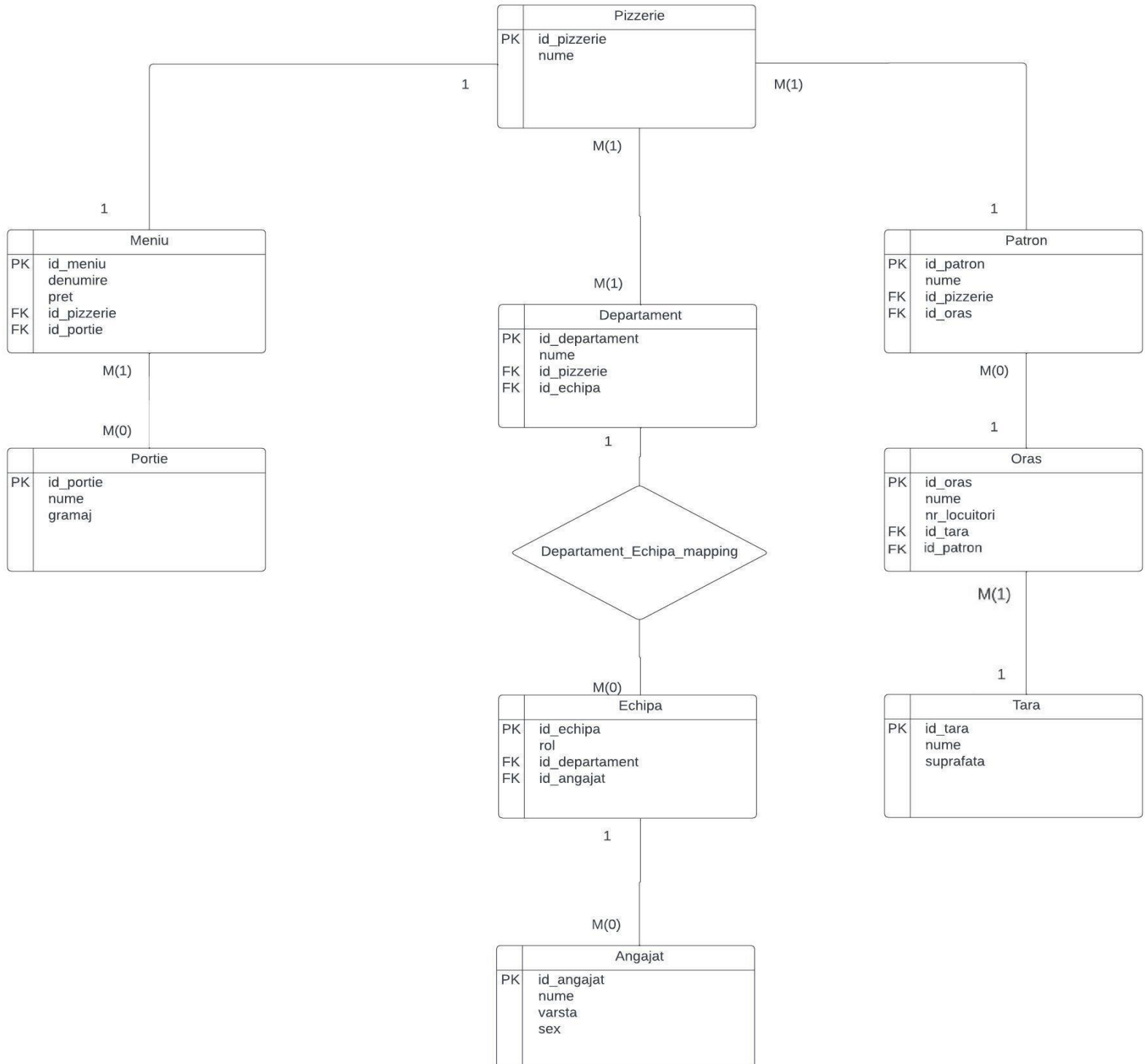
#### Tara:

- id\_tara INT
- nume CHAR (numele tarii)
- suprafata INT (suprafata tarii)

## EX6



## EX7





## EX8

Pizzeria (#id\_pizzeria, nume)

Meniu (#id\_meniu, denumire, pret, id\_pizzeria, id\_portie)

Portie (#id\_portie, nume, gramaj)

Departament (#id\_departament, nume, id\_pizzeria, id echipa)

Departament\_Echipa\_mapping (id\_departament, id echipa)

Echipa (#id echipa, rol, id\_departament, id\_angajat)

Angajat (#id\_angajat, nume, varsta, sex)

Patron (#id\_patron, nume, id\_pizzeria, id\_oras)

Oras (#id\_oras, nume, nr\_locuitori, id\_patron, id\_tara)

Tara (#id\_tara, nume, suprafata)

## EX9

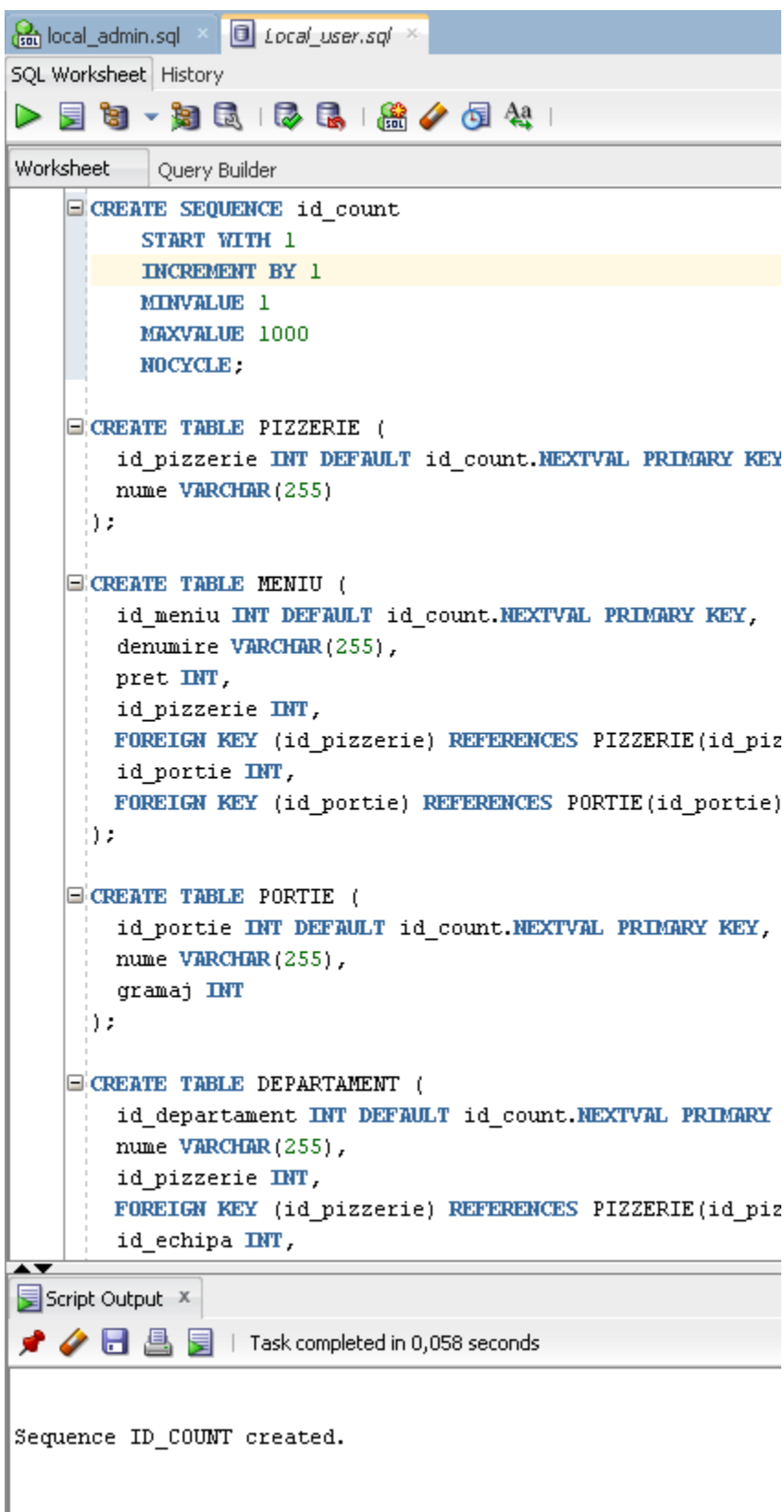
FN1: Tabelul Patron avea in plus coloana Oras, dar am facut un tabel separat cu Oras ca sa aiba o cheie primara.

FN2: Tabelul Echipa avea inainte ca PK 2 id-uri asa ca am lasat unul dintre ele acolo si pe celalalt l-am pus la un tabel nou.

FN3: Departamentul angajatilor depinde de id\_angajat si pentru asta am facut tabelul Departament.

## EX10

Secvente pt id-urile de la tabele



The screenshot shows the SQL Developer interface with two tabs: 'local\_admin.sql' and 'Local\_user.sql'. The 'Local\_user.sql' tab is active, displaying a SQL script. The script includes a 'CREATE SEQUENCE' statement for 'id\_count' and four 'CREATE TABLE' statements for 'PIZZERIE', 'MENIU', 'PORTIE', and 'DEPARTAMENT'. The 'id\_count' sequence is configured to start at 1, increment by 1, with a minimum value of 1 and a maximum value of 1000, and it is set to 'NOCYCLE'. The 'PIZZERIE' table has a primary key 'id\_pizzeria' that uses the 'id\_count' sequence's next value. The 'MENIU' table has a primary key 'id\_meniu' that also uses the 'id\_count' sequence's next value. The 'PORTIE' table has a primary key 'id\_portie' that uses the 'id\_count' sequence's next value. The 'DEPARTAMENT' table has a primary key 'id\_departament' that uses the 'id\_count' sequence's next value. The script is executed, and the 'Script Output' window at the bottom shows the message 'Sequence ID\_COUNT created.' and 'Task completed in 0,058 seconds'.

```
CREATE SEQUENCE id_count
  START WITH 1
  INCREMENT BY 1
  MINVALUE 1
  MAXVALUE 1000
  NOCYCLE;

CREATE TABLE PIZZERIE (
  id_pizzeria INT DEFAULT id_count.NEXTVAL PRIMARY KEY
  nume VARCHAR(255)
);

CREATE TABLE MENIU (
  id_meniu INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
  denumire VARCHAR(255),
  pret INT,
  id_pizzeria INT,
  FOREIGN KEY (id_pizzeria) REFERENCES PIZZERIE(id_piz
  id_portie INT,
  FOREIGN KEY (id_portie) REFERENCES PORTIE(id_portie)
);

CREATE TABLE PORTIE (
  id_portie INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
  nume VARCHAR(255),
  gramaj INT
);

CREATE TABLE DEPARTAMENT (
  id_departament INT DEFAULT id_count.NEXTVAL PRIMARY
  nume VARCHAR(255),
  id_pizzeria INT,
  FOREIGN KEY (id_pizzeria) REFERENCES PIZZERIE(id_piz
  id echipa INT,
```

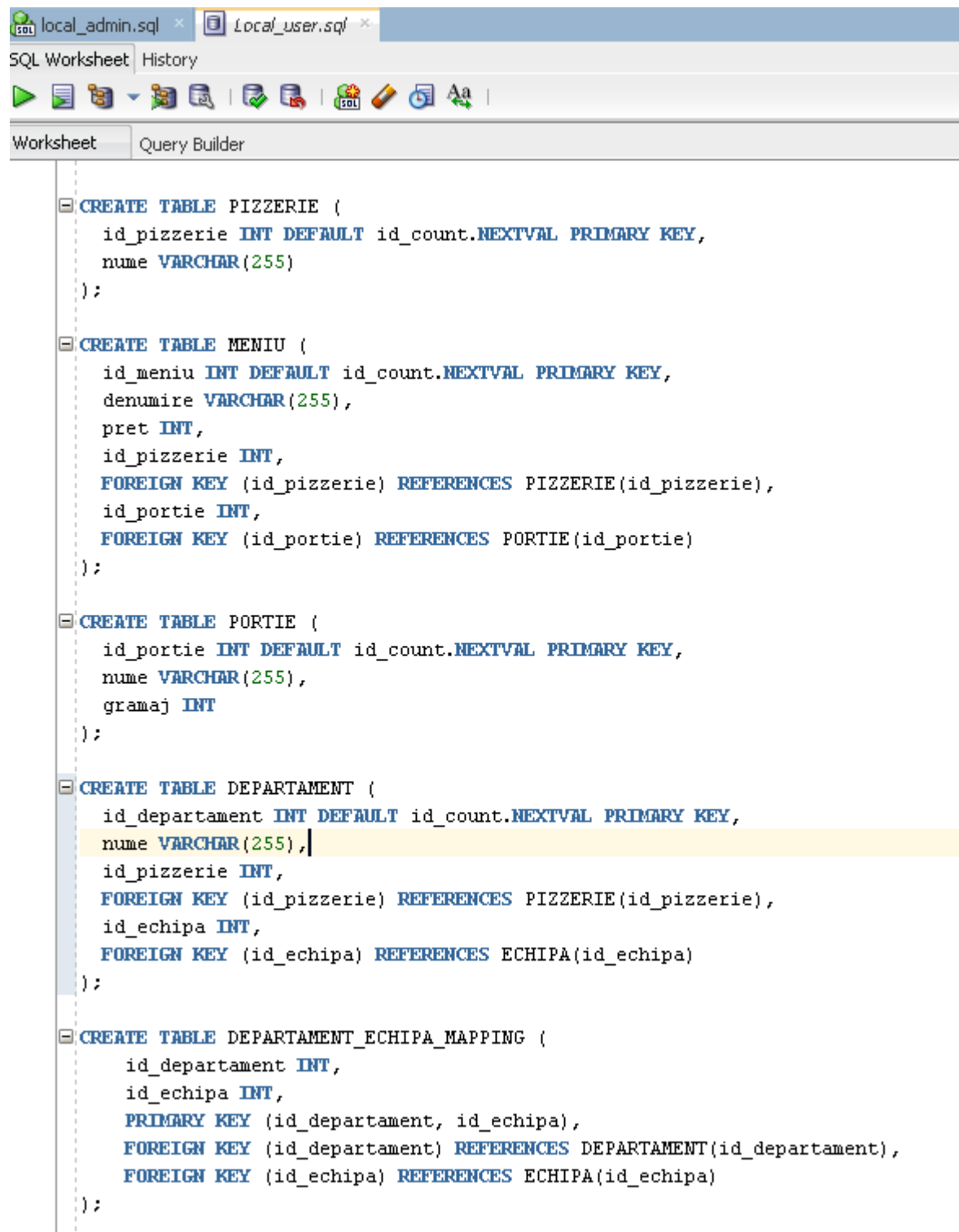
Script Output x

Task completed in 0,058 seconds

Sequence ID\_COUNT created.

## EX11

### Creearea tabelelor



```
local_admin.sql x Local_user.sql x
SQL Worksheet History
Worksheet Query Builder

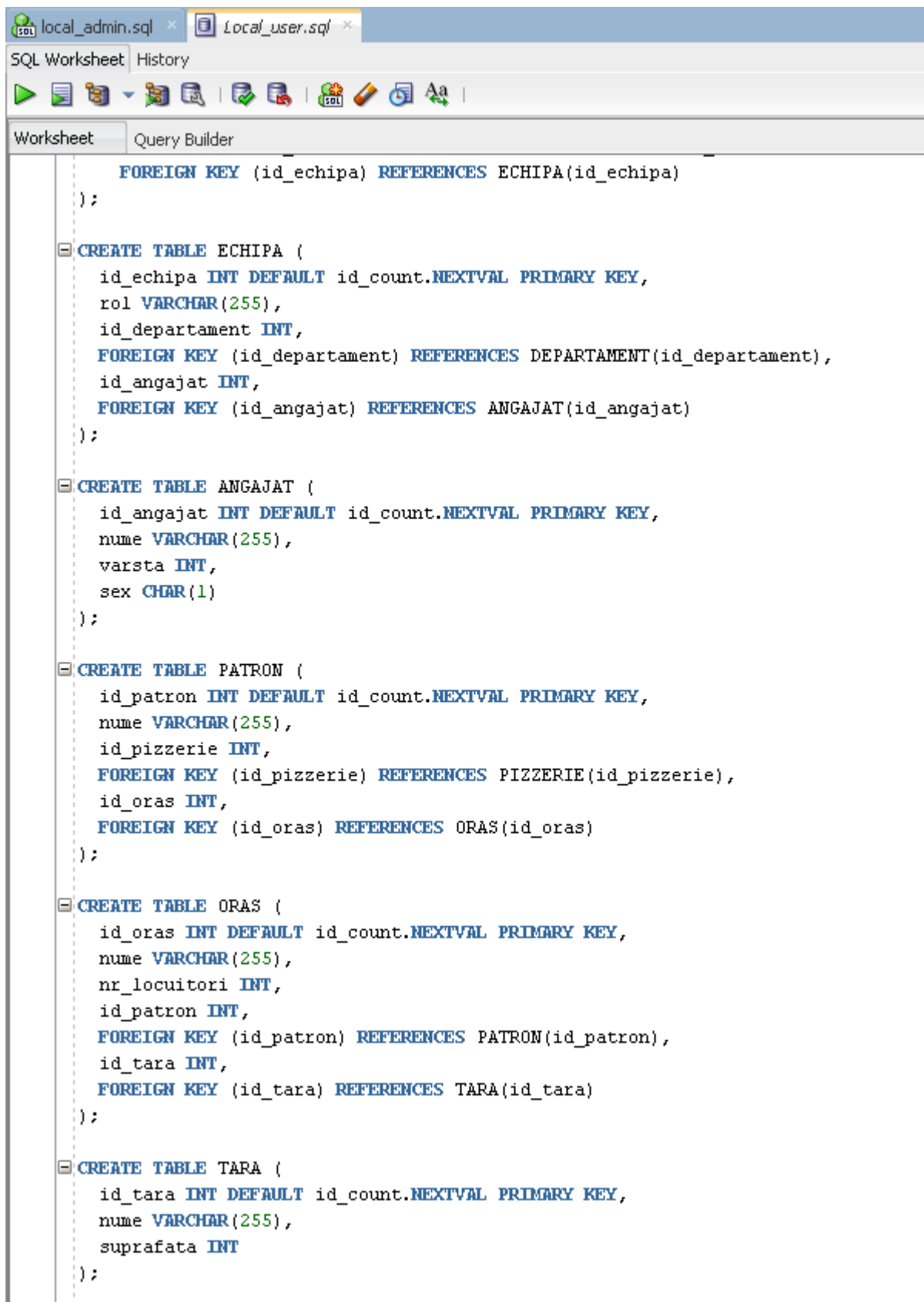
CREATE TABLE PIZZERIE (
    id_pizzeria INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255)
);

CREATE TABLE MENU (
    id_menu INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    denumire VARCHAR(255),
    pret INT,
    id_pizzeria INT,
    FOREIGN KEY (id_pizzeria) REFERENCES PIZZERIE(id_pizzeria),
    id_portie INT,
    FOREIGN KEY (id_portie) REFERENCES PORTIE(id_portie)
);

CREATE TABLE PORTIE (
    id_portie INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255),
    gramaj INT
);

CREATE TABLE DEPARTAMENT (
    id_departament INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255),
    id_pizzeria INT,
    FOREIGN KEY (id_pizzeria) REFERENCES PIZZERIE(id_pizzeria),
    id_echipa INT,
    FOREIGN KEY (id_echipa) REFERENCES ECHIPA(id_echipa)
);

CREATE TABLE DEPARTAMENT_ECHIPA_MAPPING (
    id_departament INT,
    id_echipa INT,
    PRIMARY KEY (id_departament, id_echipa),
    FOREIGN KEY (id_departament) REFERENCES DEPARTAMENT(id_departament),
    FOREIGN KEY (id_echipa) REFERENCES ECHIPA(id_echipa)
);
```



The screenshot shows a SQL IDE window with two tabs: 'local\_admin.sql' and 'Local\_user.sql'. The 'Local\_user.sql' tab is active, displaying a SQL script. The script defines five tables: ECHIPA, ANGAJAT, PATRON, ORAS, and TARA. Each table has a primary key and several foreign keys. The tables are created sequentially, with each new table's creation starting with a comment icon (a small square with a minus sign) in the left margin. The script is as follows:

```
FOREIGN KEY (id_echipa) REFERENCES ECHIPA(id_echipa)
);

CREATE TABLE ECHIPA (
    id_echipa INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    rol VARCHAR(255),
    id_departament INT,
    FOREIGN KEY (id_departament) REFERENCES DEPARTAMENT(id_departament),
    id_angajat INT,
    FOREIGN KEY (id_angajat) REFERENCES ANGAJAT(id_angajat)
);

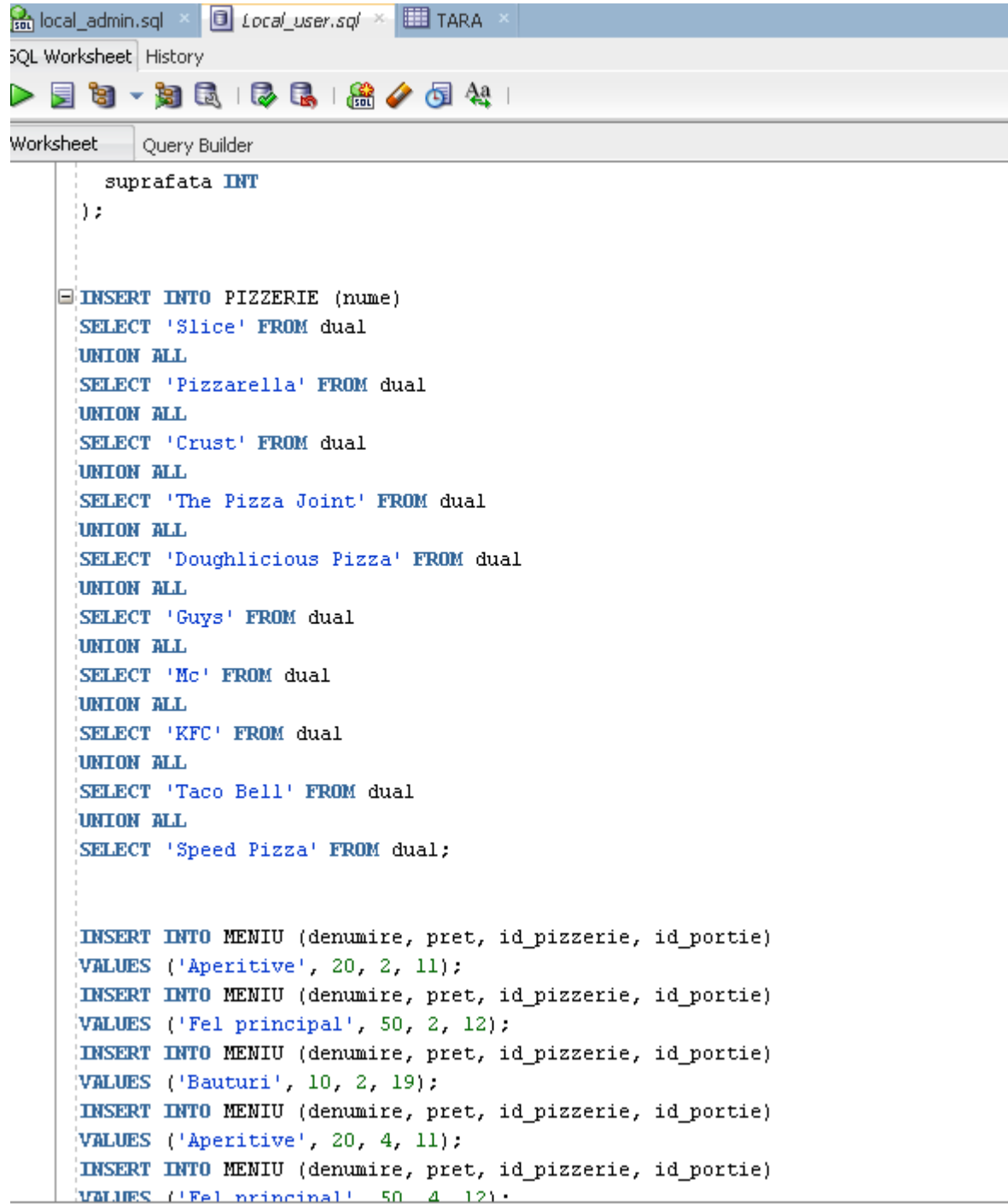
CREATE TABLE ANGAJAT (
    id_angajat INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255),
    varsta INT,
    sex CHAR(1)
);

CREATE TABLE PATRON (
    id_patron INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255),
    id_pizzerie INT,
    FOREIGN KEY (id_pizzerie) REFERENCES PIZZERIE(id_pizzerie),
    id_oras INT,
    FOREIGN KEY (id_oras) REFERENCES ORAS(id_oras)
);

CREATE TABLE ORAS (
    id_oras INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255),
    nr_locuitori INT,
    id_patron INT,
    FOREIGN KEY (id_patron) REFERENCES PATRON(id_patron),
    id_tara INT,
    FOREIGN KEY (id_tara) REFERENCES TARA(id_tara)
);

CREATE TABLE TARA (
    id_tara INT DEFAULT id_count.NEXTVAL PRIMARY KEY,
    nume VARCHAR(255),
    suprafata INT
);
```

## Popularea tabelelor



The screenshot shows a SQL Worksheet interface with three tabs: 'local\_admin.sql', 'Local\_user.sql', and 'TARA'. The 'TARA' tab is active. The interface includes a toolbar with various icons for execution, saving, and editing. Below the toolbar, there are two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab is selected, displaying a SQL query. The query defines a table 'suprafata' with an integer column 'INT' and then inserts data into 'PIZZERIE' and 'MENIU' tables using 'UNION ALL' and 'VALUES' clauses.

```
suprafata INT
);

INSERT INTO PIZZERIE (nume)
SELECT 'Slice' FROM dual
UNION ALL
SELECT 'Pizzarella' FROM dual
UNION ALL
SELECT 'Crust' FROM dual
UNION ALL
SELECT 'The Pizza Joint' FROM dual
UNION ALL
SELECT 'Doughlicious Pizza' FROM dual
UNION ALL
SELECT 'Guys' FROM dual
UNION ALL
SELECT 'Mc' FROM dual
UNION ALL
SELECT 'KFC' FROM dual
UNION ALL
SELECT 'Taco Bell' FROM dual
UNION ALL
SELECT 'Speed Pizza' FROM dual;

INSERT INTO MENIU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Aperitive', 20, 2, 11);
INSERT INTO MENIU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Fel principal', 50, 2, 12);
INSERT INTO MENIU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Bauturi', 10, 2, 19);
INSERT INTO MENIU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Aperitive', 20, 4, 11);
INSERT INTO MENIU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Fel principal', 50, 4, 12);
```



local\_admin.sql x local\_user.sql x TARA x

SQL Worksheet History

Worksheet Query Builder

```
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Bauturi', 10, 2, 19);
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Aperitive', 20, 4, 11);
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Fel principal', 50, 4, 12);
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Bauturi', 10, 4, 19);
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Aperitive', 20, 6, 11);
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Fel principal', 50, 6, 12);
INSERT INTO MENU (denumire, pret, id_pizzerie, id_portie)
VALUES ('Bauturi', 10, 6, 19);
INSERT INTO MENU (denumire, pret, id_pizzerie)
VALUES ('Desert', 20, 8);

INSERT INTO PORTIE (nume, gramaj)
VALUES ('Burger', 500);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Pizza', 400);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Apa', 50);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Paste', 300);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Salata', 200);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Sandwich', 250);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Supa', 350);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Frigarui', 450);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Sushi', 350);
```

local\_admin.sqlLocal\_user.sqlTARA

SQL WorksheetHistory

WorksheetQuery Builder

```
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Supa', 350);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Frigarui', 450);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Sushi', 350);
INSERT INTO PORTIE (nume, gramaj)
VALUES ('Tortilla', 400);

INSERT INTO DEPARTAMENT (nume, id_pizzerie) VALUES ('Bucatarie', 2);
INSERT INTO DEPARTAMENT (nume, id_pizzerie) VALUES ('Personal', 2);
INSERT INTO DEPARTAMENT (nume, id_pizzerie) VALUES ('Contabilitate', 2);
INSERT INTO DEPARTAMENT (nume, id_pizzerie) VALUES ('Curatenie', 2);
INSERT INTO DEPARTAMENT (nume, id_pizzerie) VALUES ('Marketing', 2);

INSERT INTO DEPARTAMENT_ECHIPA_MAPPING (id_departament, id echipa) VALUES (51, 52);
INSERT INTO DEPARTAMENT_ECHIPA_MAPPING (id_departament, id echipa) VALUES (51, 54);
INSERT INTO DEPARTAMENT_ECHIPA_MAPPING (id_departament, id echipa) VALUES (56, 57);
INSERT INTO DEPARTAMENT_ECHIPA_MAPPING (id_departament, id echipa) VALUES (59, 55);
INSERT INTO DEPARTAMENT_ECHIPA_MAPPING (id_departament, id echipa) VALUES (60, 55);

INSERT INTO ECHIPA (rol, id_departament, id_angajat) VALUES ('Bucatar', 51, 41);
INSERT INTO ECHIPA (rol, id_departament, id_angajat) VALUES ('Ajutor', 51, 42);
INSERT INTO ECHIPA (rol, id_departament, id_angajat) VALUES ('Chelner', 51, 43);
INSERT INTO ECHIPA (rol, id_departament, id_angajat) VALUES ('Chelner', 53, 44);
INSERT INTO ECHIPA (rol, id_departament, id_angajat) VALUES ('Contabil', 56, 45);

INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Radu', 50, 'M');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Elena', 35, 'F');
INSERT INTO ANGAJAT (nume, varsta, sex)
```

The screenshot shows an SQL Worksheet application window with three tabs: 'local\_admin.sql', 'Local\_user.sql', and 'TARA'. The 'Local\_user.sql' tab is active. The window has a menu bar with 'SQL Worksheet' and 'History', and a toolbar with various icons. Below the toolbar is a tabbed interface with 'Worksheet' and 'Query Builder'. The 'Worksheet' tab is selected, displaying a list of SQL INSERT statements. The statements are color-coded: keywords in blue, table names in green, and values in red. The first block of statements inserts data into the 'ANGAJAT' table, and the second block inserts data into the 'TARA' table.

```
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Radu', 50, 'M');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Elena', 35, 'F');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Andrei', 28, 'M');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Maria', 42, 'F');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Alexandru', 31, 'M');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Ana', 55, 'F');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Cristian', 39, 'M');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Simona', 27, 'F');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Mihai', 48, 'M');
INSERT INTO ANGAJAT (nume, varsta, sex)
VALUES ('Ioana', 33, 'F');

INSERT INTO TARA (nume, suprafata)
VALUES ('Romania', 238);
INSERT INTO TARA (nume, suprafata)
VALUES ('SUA', 962);
INSERT INTO TARA (nume, suprafata)
VALUES ('Canada', 997);
INSERT INTO TARA (nume, suprafata)
VALUES ('China', 964);
INSERT INTO TARA (nume, suprafata)
VALUES ('Rusia', 170);
INSERT INTO TARA (nume, suprafata)
VALUES ('Australia', 769);
INSERT INTO TARA (nume, suprafata)
```



local\_admin.sql x local\_user.sql x TARA x

SQL Worksheet History

Worksheet Query Builder

```
VALUES ('China', 964);
INSERT INTO TARA (nume, suprafata)
VALUES ('Rusia', 170);
INSERT INTO TARA (nume, suprafata)
VALUES ('Australia', 769);
INSERT INTO TARA (nume, suprafata)
VALUES ('Brazil', 851);
INSERT INTO TARA (nume, suprafata)
VALUES ('India', 328);
INSERT INTO TARA (nume, suprafata)
VALUES ('Argentina', 278);
INSERT INTO TARA (nume, suprafata)
VALUES ('Kazakhstan', 272);

INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Bucuresti', 20, 61);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Craiova', 2, 61);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Brasov', 6, 61);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Suceava', 8, 61);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Constanta', 10, 61);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Los Angeles', 200, 62);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('California', 30, 62);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('San Francisco', 50, 62);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('Alaska', 45, 62);
INSERT INTO ORAS (nume, nr_locuitori, id_tara) VALUES ('New York', 150, 62);

INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Marian', 2, 71);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Marius', 2, 72);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Andreea', 4, 73);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Dima', 4, 74);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Ghita', 2, 75);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Andrei', 2, 75);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Marcus', 6, 76);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Ciolacu', 2, 77);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Marcel', 6, 77);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Traian', 2, 78);
```

Script Output x

```

INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Dima', 4, 74);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Ghita', 2, 75);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Andrei', 2, 75);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Marcus', 6, 76);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Ciolacu', 2, 77);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Marcel', 6, 77);
INSERT INTO PATRON (nume, id_pizzerie, id_oras) VALUES ('Traian', 2, 78);

```

local\_admin.sql Local\_user.sql ANGAJAT

Columns Data Model Constraints Grants Statistics Triggers Flashback Depe

Sort.. Filter:

	ID_ANGAJAT	NUME	VARSTA	SEX
1	41	Radu	50	M
2	42	Elena	35	F
3	43	Andrei	28	M
4	44	Maria	42	F
5	45	Alexandru	31	M
6	46	Ana	55	F
7	47	Cristian	39	M
8	48	Simona	27	F
9	49	Mihai	48	M
10	50	Ioana	33	F

local\_admin.sql Local\_user.sql DEPARTAMENT

Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies Details

Sort.. Filter:

	ID_DEPARTAMENT	NUME	ID_PIZZERIE	ID_ECHIPA
1	51	Bucatarie	2	(null)
2	53	Personal	2	(null)
3	56	Contabilitate	2	(null)
4	59	Curatenie	2	(null)
5	60	Marketing	2	(null)

local\_admin.sql Local\_user.sql DEPARTAMENT\_ECHIPA\_MAPPING

Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies Details | P.

Sort.. Filter:

	ID_DEPARTAMENT	ID_ECHIPA
1	51	52
2	51	54
3	56	57
4	59	55
5	60	55

local\_admin.sql Local\_user.sql ECHIPA

Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies De

Sort.. Filter:

	ID_ECHIPA	ROL	ID_DEPARTAMENT	ID_ANGAJAT
1	52	Bucatar	51	41
2	54	Chelner	51	42
3	55	Chelner	53	43
4	57	Contabil	56	44
5	58	Ajutor	51	45

local\_admin.sql Local\_user.sql MENU

Columns Data Model Constraints Grants Statistics Triggers Flashback Dependencies t

Sort.. Filter:

	ID_MENU	DENUMIRE	PRET	ID_PIZZERIE	ID_PORTIE
1	31	Aperitive	20	2	11
2	32	Fel principal	50	2	12
3	33	Bauturi	10	2	19
4	34	Aperitive	20	4	11
5	35	Fel principal	50	4	12
6	36	Bauturi	10	4	19
7	37	Aperitive	20	6	11
8	38	Fel principal	50	6	12
9	39	Bauturi	10	6	19

local\_admin.sql

Local\_user.sql

ORAS

Columns

Data

Model

Constraints

Grants

Statistics

Triggers

Flashback

Dependencies

Details

Pa

Sort..

Filter:

	ID_ORAS	NUME	NR_LOCUITORI	ID_PATRON	ID_TARA	
1	71	Bucuresti	20	(null)	61	
2	72	Craiova	2	(null)	61	
3	73	Brasov	6	(null)	61	
4	74	Suceava	8	(null)	61	
5	75	Constanta	10	(null)	61	
6	76	Los Angeles	200	(null)	62	
7	77	California	30	(null)	62	
8	78	San Francisco	50	(null)	62	
9	79	Alaska	45	(null)	62	
10	80	New York	150	(null)	62	

local_admin.sql	Local_user.sql	PATRON
Columns	Data	Model
Constraints	Grants	Statistics
Triggers	Flashback	
Sort..	Filter:	
ID_PATRON	NUME	ID_PIZZERIE
1	1 Marian	2
2	2 Marius	2
3	3 Andreea	4
4	4 Dima	4
5	5 Ghita	2
6	6 Andrei	2
7	7 Marcus	6
8	8 Ciolacu	2
9	9 Marcel	6
10	10 Traian	2

local\_admin.sql x Local\_user.sql x PIZZERIE x

Columns Data Model Constraints Grants Statistics Triggers Flashback

Sort.. Filter:

	ID_PIZZERIE	NUME
1	2	Slice
2	4	Pizzarella
3	6	Crust
4	8	The Pizza Joint
5	10	Doughlicious Pizza
6	22	Guys
7	24	Mc
8	26	KFC
9	28	Taco Bell
10	30	Speed Pizza

local\_admin.sql x Local\_user.sql x PORTIE x

Columns Data Model Constraints Grants Statistics Triggers Flashback

Sort.. Filter:

	ID_PORTIE	NUME	GRAMAJ
1	11	Salata	200
2	12	Sandwich	250
3	13	Supa	350
4	14	Frigarui	450
5	15	Sushi	350
6	16	Tortilla	400
7	17	Burger	500
8	18	Pizza	400
9	19	Apa	50
10	20	Paste	300

	ID_TARA	NUME	SUPRAFATA
1	61	Romania	238
2	62	SUA	962
3	63	Canada	997
4	64	China	964
5	65	Rusia	170
6	66	Australia	769
7	67	Brazil	851
8	68	India	328
9	69	Argentina	278
10	70	Kazakhstan	272

## EX12

A)

```
-- cerere in care folosesc 5 tabele si folosesc clauza FROM
-- afiseaza cati angajati are o pizzerie
SELECT p.nume nume_pizzerie, COUNT(*) numar_angajati
FROM pizzerie p JOIN departament d
    ON p.id_pizzerie = d.id_pizzerie
    JOIN DEPARTAMENT_ECHIPA_MAPPING de
    ON d.id_departament = de.id_departament
    JOIN ECHIPA e
    ON de.id echipa = e.id echipa
    JOIN angajat a ON
    e.id_angajat = a.id_angajat
WHERE p.nume LIKE 'Pizzarella'
GROUP BY p.nume;
```

	NUME_PIZZERIE	NUMAR_ANGAJATI
1	Pizzarella	5

B)

```
-- cerere care contine UPPER, LOWER
-- scrie cu majuscule numele pizzeriilor so cu litere mici numele patronilor

SELECT
    CONCAT('Pizzeria ', UPPER(p.num)) nume_pizzeria,
    CONCAT('Patronul ', LOWER(pa.num)) nume_patron
FROM
    PIZZERIA p
JOIN PATRON pa ON p.id_pizzeria = pa.id_pizzeria;
```


Script Output x Query Result x	
SQL   All Rows Fetched: 10 in 0,002 seconds	
NUME_PIZZERIA	NUME_PATRON
1 Pizzeria SLICE	Patronul marian
2 Pizzeria SLICE	Patronul marius
3 Pizzeria PIZZARELLA	Patronul andreea
4 Pizzeria PIZZARELLA	Patronul dima
5 Pizzeria SLICE	Patronul ghita
6 Pizzeria SLICE	Patronul andrei
7 Pizzeria CRUST	Patronul marcus
8 Pizzeria SLICE	Patronul ciolacu
9 Pizzeria CRUST	Patronul marcel
10 Pizzeria SLICE	Patronul traian

C)

```
-- cerere care contine functie pe date calendaristice
-- afiseaza ziua si luna curenta
SELECT P.*, EXTRACT(DAY FROM CURRENT_DATE) ZiuaCurenta, TO_CHAR(CURRENT_DATE, 'Month') LunaCurenta
FROM PIZZERIA P;

-- cerere care foloseste nvl si decode
```

Script Output x Query Result x

 All Rows Fetched: 5 in 0,002 seconds

ID_PIZZERIA	NUME	ZIUA_CURENTA	LUNA_CURENTA
1	2 Slice	18 June	
2	4 Pizzarella	18 June	
3	6 Crust Restaurant	18 June	
4	8 The Pizza Joint	18 June	
5	10 Doughlicious Pizza	18 June	



D)

```
-- cerere care foloseste nvl si decode
-- afiseaza angajatii si ii filtreaza in functie de varsta
SELECT a.numa nume_angajat, a.varsta,
       NVL(a.sex, 'Necunoscut') AS sex_angajat,
       DECODE(SIGN(a.varsta - 30), -1, 'Tanar', SIGN(a.varsta - 50), 'Matur', 'In Varsta') categorie_varsta
FROM angajat a
ORDER BY a.numa;
```

Script Output x Query Result x

All Rows Fetched: 5 in 0,002 seconds

NUME_PIZZERIA	DENUMIRE_MENIU
1 Pizzeria SLICE	Meniul paste
2 Pizzeria SLICE	Meniul ciorba
3 Pizzeria SLICE	Meniul platou
4 Pizzeria SLICE	Meniul burger
5 Pizzeria SLICE	Meniul pizza

E)

```
-- cerere care foloseste with si case
-- filtreaza salariile si le afiseaza doar pe cele mari
WITH salarii_angajati AS (
  SELECT a.id_angajat, a.numa, c.salariu,
         CASE
           WHEN c.salariu >= 5000 THEN 'Mare'
           WHEN c.salariu BETWEEN 3000 AND 4999 THEN 'Mediu'
           ELSE 'Mic'
         END categorie_salariu
  FROM angajat a
  JOIN contabilitate c ON a.id_angajat = c.id_angajat
)
SELECT s.numa, s.salariu, s.categorie_salariu
FROM salarii_angajati s
WHERE s.categorie_salariu = 'Mare'
ORDER BY s.salariu DESC;
```

UPDATE MENTII

Script Output x Query Result x

All Rows Fetched: 14 in 0,004 seconds

	NUME	SALARIU	CATEGORIE_SALARIU
1	Simona	85000	Mare
2	Simona	85000	Mare
3	Simona	85000	Mare
4	Ioana	53000	Mare
5	Ioana	53000	Mare
6	Ioana	53000	Mare
7	Mihai	52000	Mare
8	Mihai	52000	Mare
9	Alexandru	51000	Mare
10	Ana	50600	Mare
11	Andrei	8000	Mare
12	Cristian	7000	Mare
13	Elena	6000	Mare
14	Radu	5000	Mare

F)

```
-- cerere care foloseste with si case
-- filtreaza preturile si spune daca sunt scumpe sau ieftine
WITH cte AS (
    SELECT
        m.id_meniu, m.denumire, m.pret, p.nume
    FROM
        MENUU m
        INNER JOIN PIZZERIE p ON m.id_pizzeria = p.id_pizzeria
    ORDER BY
        m.pret DESC
)
SELECT
    id_meniu, denumire, pret, nume,
    CASE
        WHEN pret >= 50 THEN 'Scump'
        WHEN pret >= 30 THEN 'Ieftin'
        ELSE 'Ieftin'
    END AS price_category
FROM
    cte
ORDER BY
    pret DESC;
```

Script Output x Query Result x					
SQL   All Rows Fetched: 10 in 0,005 seconds					
	ID_MENIU	DENUMIRE	PRET	NUME	PRICE_CATEGORY
1	35	Fel principal	50	Pizzarella	Scump
2	38	Fel principal	50	Crust	Scump
3	32	Fel principal	50	Slice	Scump
4	34	Aperitive	20	Pizzarella	Ieftin
5	40	Desert	20	The Pizza Joint	Ieftin
6	37	Aperitive	20	Crust	Ieftin
7	31	Aperitive	20	Slice	Ieftin
8	36	Bauturi	10	Pizzarella	Ieftin
9	33	Bauturi	10	Slice	Ieftin
10	39	Bauturi	10	Crust	Ieftin

## EX13

```
UPDATE MENU
SET pret = (
    SELECT pret + 5
    FROM MENU
    WHERE id_menu = 12
)
WHERE id_menu = 12;

UPDATE PIZZERIA
SET nume = (
    SELECT CONCAT(nume, ' Restaurant')
    FROM PIZZERIA
    WHERE id_pizzeria = 6
)
WHERE id_pizzeria = 6;

DELETE FROM MENU WHERE id_menu = 31;
```