Gestionarea unui lanț
de magazine de jocuri video

Student: Anghel Fabian

Cuprins

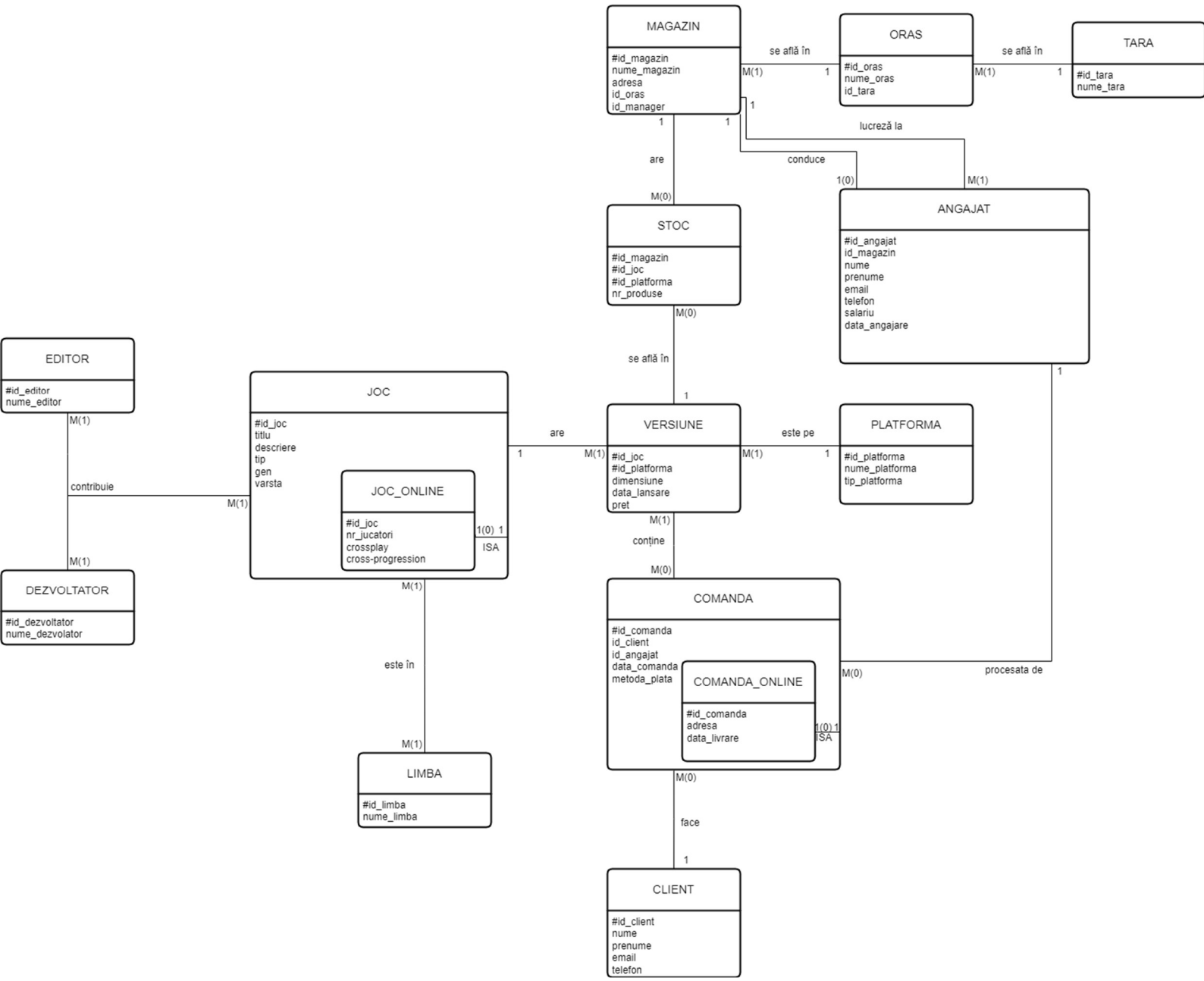
1. Descrierea bazei de date

Modelul de date va gestiona informații legate de organizarea și gestionarea vânzărilor din cadrul unui lanț de magazine de jocuri video. Aceste magazine se află la o adresă în anumite orașe și anumite țări. De asemenea fiecare magazin are un stoc limitat de jocuri de care se ține evidența. Mai mulți angajați lucrează într-un magazin, iar unul dintre ei este manager în acea locație. Despre un angajat se reține mai multe informații precum nume, salariu și date de contact.

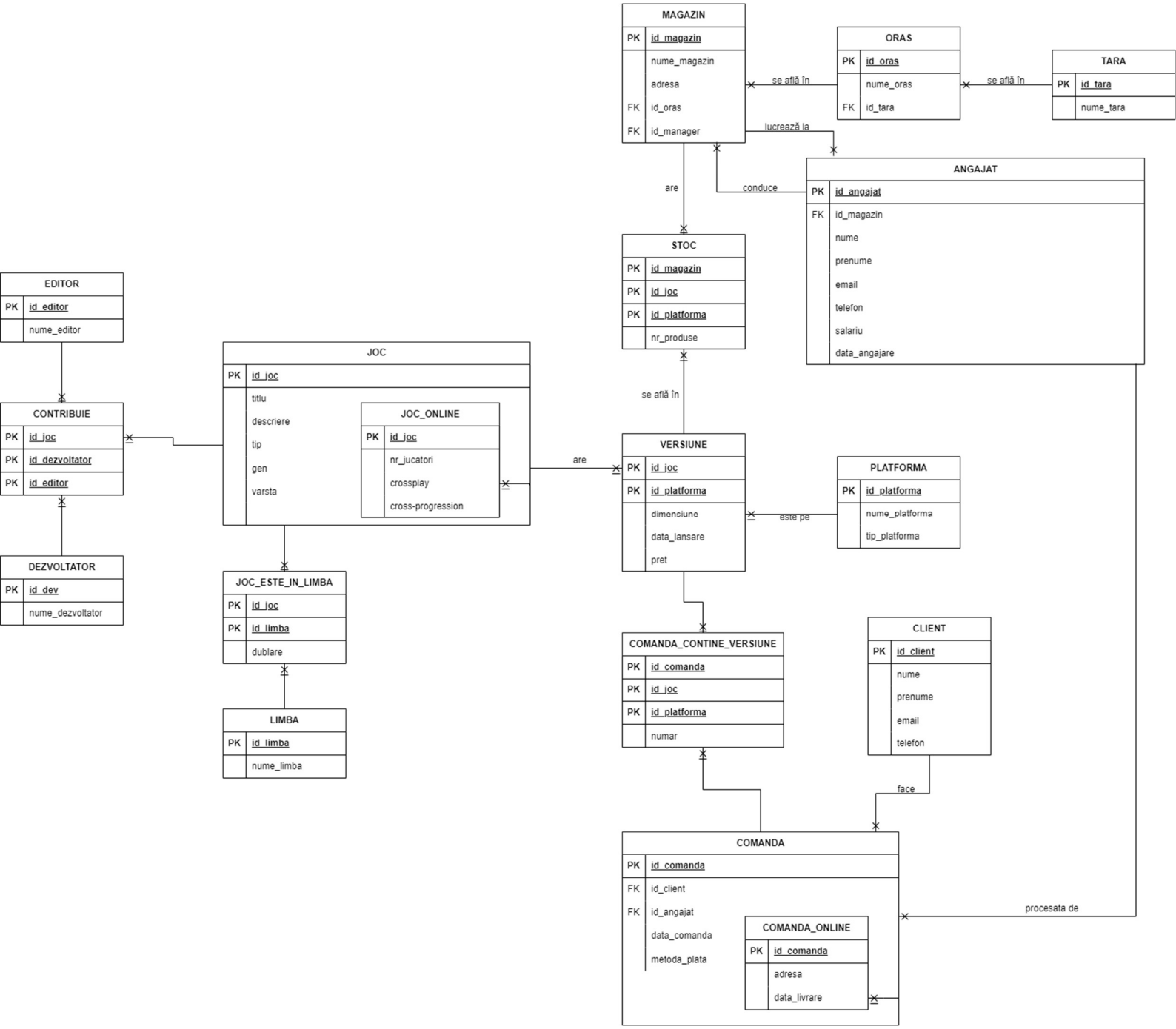
Un joc, care are anumite specificații, are mai multe versiuni care depind de platforma pe care se află. De asemenea, un joc a fost făcut de anumiți dezvoltatori și publicat de editori. De asemenea el poate avea mai multe limbi disponibile.

Un client, despre care se reține mai multe date de contact, poate da o comandă de mai multe jocuri de anumite versiuni. O camandă poate fi și online, fiind livrată la o anumită adresă. O comandă este procesată de un angajat.

2. Diagrama entitate-relație



3. Diagrama conceptuală



4. Crearea tabelelor

```
CREATE TABLE tara(
  id_tara varchar2(2) CONSTRAINT tara_pk PRIMARY KEY,
  nume_tara varchar2(30) CONSTRAINT tara_nume_nn NOT NULL
            CONSTRAINT tara_nume_unq UNIQUE
);

CREATE TABLE oras(
  id_oras number(3) CONSTRAINT oras_pk PRIMARY KEY,
  nume_oras varchar2(20) CONSTRAINT oras_nume_nn NOT NULL,
  id_tara varchar2(2) CONSTRAINT oras_fk REFERENCES tara(id_tara) ON DELETE SET NULL
);

CREATE TABLE magazin(
  id_magazin number(3) CONSTRAINT mag_pk PRIMARY KEY,
  nume_magazin varchar2(30),
  adresa varchar2(100),
  id_oras CONSTRAINT mag_fk_oras REFERENCES oras(id_oras) ON DELETE SET NULL,
  id_manager number(4)
);

CREATE TABLE joc(
  id_joc number(6) CONSTRAINT joc_pk PRIMARY KEY,
  titlu varchar2(30) CONSTRAINT joc_titlu_nn NOT NULL,
  descriere varchar2(300),
  gen varchar2(15),
  multiplayer number(1) CONSTRAINT joc_mp_chk CHECK(multiplayer = 0 or multiplayer = 1),
  varsta number(2) DEFAULT 3
);

CREATE TABLE joc_online(
  id_joc number(6) CONSTRAINT joc_on_pk PRIMARY KEY
          CONSTRAINT joc_on_fk REFERENCES joc(id_joc) ON DELETE CASCADE,
  nr_jucatori number(8) CONSTRAINT joc_on_juc_chk CHECK(nr_jucatori > 0),
  crossplay number(1) CONSTRAINT joc_cplay_chk CHECK(crossplay = 0 or crossplay = 1),
  cross_progression number(1) CONSTRAINT joc_cprog_chk CHECK(cross_progression = 0 or
cross_progression = 1)
);

CREATE TABLE platforma(
  id_platforma number(2) CONSTRAINT plat_pk PRIMARY KEY,
  nume_platforma varchar2(15) CONSTRAINT plat_nume_nn NOT NULL
          CONSTRAINT plat_nume_unq UNIQUE,
  tip_platforma varchar2(10) CONSTRAINT plat_tip_nn NOT NULL
);

CREATE TABLE versiune(
  id_joc number(6) CONSTRAINT vers_joc_fk REFERENCES joc(id_joc) ON DELETE CASCADE,
  id_platforma number(2) CONSTRAINT vers_plat_fk REFERENCES platforma(id_platforma) ON
DELETE CASCADE,
```

```
    dimensiune number(4,2),
    data_lansare date,
    pret number(4,2) CONSTRAINT vers_pret_nn NOT NULL,
    CONSTRAINT vers_pk PRIMARY KEY(id_joc, id_platforma)
);

CREATE TABLE stoc(
    id_magazin number(3) CONSTRAINT stoc_mag_fk REFERENCES magazin(id_magazin) ON DELETE CASCADE,
    id_joc number(6),
    id_platforma number(2),
    nr_produse number(3) CONSTRAINT stoc_nr_nn NOT NUll,
    CONSTRAINT stoc_pk PRIMARY KEY(id_magazin, id_joc, id_platforma),
    CONSTRAINT stoc_vers_fk FOREIGN KEY(id_joc, id_platforma) REFERENCES versiune(id_joc, id_platforma) ON DELETE CASCADE
);

CREATE TABLE dezvoltator(
    id_dezvoltator number(3) CONSTRAINT dezv_pk PRIMARY KEY,
    nume_dezvoltator varchar2(15) CONSTRAINT dezv_nume_unq UNIQUE
);

CREATE TABLE editor(
    id_editor number(3) CONSTRAINT editor_pk PRIMARY KEY,
    nume_editor varchar2(15) CONSTRAINT editor_nume_unq UNIQUE
);

CREATE TABLE contribuie(
    id_joc number(6) CONSTRAINT contr_joc_fk REFERENCES joc(id_joc) ON DELETE CASCADE,
    id_dezvoltator number(3) CONSTRAINT contr_dezv_fk REFERENCES dezvoltator(id_dezvoltator) ON DELETE CASCADE,
    id_editor number(3) CONSTRAINT contr_edit_fk REFERENCES editor(id_editor) ON DELETE CASCADE,
    CONSTRAINT contr_pk PRIMARY KEY(id_joc, id_dezvoltator, id_editor)
);

CREATE TABLE limba(
    id_limba varchar2(2) CONSTRAINT limba_pk PRIMARY KEY,
    nume_limba varchar2(15) CONSTRAINT limba_nume_unq UNIQUE
);

CREATE TABLE joc_in_limba(
    id_joc number(6) CONSTRAINT joc_lim_fk_joc REFERENCES joc(id_joc) ON DELETE CASCADE,
    id_limba varchar2(2) CONSTRAINT joc_lim_fk_lim REFERENCES limba(id_limba) ON DELETE CASCADE,
    dublare number(1) CONSTRAINT joc_lim_dub_chk CHECK(dublare = 0 or dublare = 1),
    CONSTRAINT joc_lim_pk PRIMARY KEY(id_joc, id_limba)
);

CREATE TABLE angajat(
    id_angajat number(4) CONSTRAINT ang_pk PRIMARY KEY,
    id_magazin number(3) CONSTRAINT ang_fk REFERENCES magazin(id_magazin)
```

```
                    CONSTRAINT ang_mag_nn NOT NULL,
        nume varchar2(20) CONSTRAINT ang_nume_nn NOT NULL,
        prenume varchar2(20),
        email varchar2(30),
        telefon varchar2(12),
        salariu number(5) CONSTRAINT ang_sal_chk CHECK (salariu > 500),
        data_angajare date DEFAULT SYSDATE
    );

    CREATE TABLE client(
        id_client number(7) CONSTRAINT clnt_pk PRIMARY KEY,
        nume varchar2(20),
        prenume varchar2(20),
        email varchar2(30),
        telefon varchar2(12)
    );

    CREATE TABLE comanda(
        id_comanda number(10) CONSTRAINT comd_pk PRIMARY KEY,
        id_client number(7) CONSTRAINT comd_clnt_fk REFERENCES client(id_client) ON DELETE SET
NULL,
        id_angajat number(4) CONSTRAINT comd_ang_fk REFERENCES angajat(id_angajat) ON
DELETE SET NULL,
        data_comanda date DEFAULT SYSDATE,
        metoda_plata varchar2(10)
    );

    CREATE TABLE comanda_online(
        id_comanda number(10) CONSTRAINT comd_on_pk PRIMARY KEY
                    CONSTRAINT comd_on_fk REFERENCES comanda(id_comanda) ON DELETE
CASCADE,
        adresa varchar2(100),
        data_livrare date
    );

    CREATE TABLE comanda_contine(
        id_joc number(6),
        id_platforma number(2),
        id_comanda number(10) CONSTRAINT comd_cont_fk_comd REFERENCES
comanda(id_comanda) ON DELETE CASCADE,
        numar number(2) CONSTRAINT comd_cont_chk CHECK(numar > 0),
        CONSTRAINT comd_cont_fk_ver FOREIGN KEY(id_joc, id_platforma) REFERENCES
versiune(id_joc, id_platforma) ON DELETE CASCADE,
        CONSTRAINT comd_cont_pk PRIMARY KEY(id_joc, id_platforma, id_comanda)
    );

    ALTER TABLE magazin
    ADD CONSTRAINT mag_fk_mng FOREIGN KEY(id_manager)
REFERENCES angajat(id_angajat);
```

```
Table TARA created.
Table ORAS created.
Table MAGAZIN created.
Table JOC created.
Table JOC_ONLINE created.
Table PLATFORMA created.
Table VERSIUNE created.
Table STOC created.
Table DEZVOLTATOR created.
Table EDITOR created.
Table CONTRIBUIE created.
Table LIMBA created.
Table JOC_IN_LIMBA created.
Table ANGAJAT created.
Table CLIENT created.
Table COMANDA created.
Table COMANDA_ONLINE created.
Table COMANDA_CONTINE created.
Table MAGAZIN altered.
```

5. Inserarea datelor

```
INSERT ALL
INTO tara VALUES ('RO', 'Romania')
INTO tara VALUES ('NL', 'Olanda')
INTO tara VALUES ('GB', 'Marea Britanie')
INTO tara VALUES ('IT', 'Italia')
INTO tara VALUES ('HU', 'Ungaria')
SELECT * FROM DUAL;

INSERT ALL
INTO oras VALUES(11, 'Bucuresti', 'RO')
INTO oras VALUES(12, 'Galati', 'RO')
INTO oras VALUES(13, 'Cluj', 'RO')
INTO oras VALUES(14, 'Amsterdam', 'NL')
INTO oras VALUES(15, 'Brasov', 'RO')
INTO oras VALUES(16, 'Londra', 'GB')
INTO oras VALUES(17, 'Roma', 'IT')
INTO oras VALUES(18, 'Rotterdam', 'NL')
INTO oras VALUES(19, 'Budapesta', 'HU')
INTO oras VALUES(20, 'Milano', 'IT')
SELECT * FROM DUAL;

INSERT ALL
INTO magazin
VALUES(300, 'Gameshop AFI', 'Bulevardul General Paul Teodorescu 4', 11, NULL)
INTO magazin
VALUES(301, 'Gameshop Centru Vechi', 'Strada Lipscani 55, Bucure?ti', 11, NULL)
```

```
    INTO magazin
    VALUES(302, 'Gamecorner Tiglina', 'Strada Br?ilei nr. 163, 800309', 12, NULL)
    INTO magazin
    VALUES(303, 'Gamecorner Gorjului', 'Bulevardul Iuliu Maniu 67', 11, NULL)
    INTO magazin
    VALUES(304, 'Gameshop Cluj', 'Strada Alexandru Vaida Voevod 59, 400436', 13, NULL)
    INTO magazin
    VALUES(305, 'Gameshop Amsterdam', NULL, 14, NULL)
    INTO magazin
    VALUES(306, 'Megagame Brasov', 'Bulevardul 15 Noiembrie 78', 15, NULL)
    INTO magazin
    VALUES(307, 'Gamecorner Walworth', 'East St, London SE17 1EL', 16, NULL)
    INTO magazin
    VALUES(308, 'Gameshop Roma', NULL, 17, NULL)
    INTO magazin
    VALUES(309, 'Gameshop Rotterdam', 'Willem Ruyslaan 225, 3063 ER', 18, NULL)
    INTO magazin
    VALUES(310, 'Megashop Londra', '12 Walbrook, London EC4N 8AA', 16, NULL)
    INTO magazin
    VALUES(311, 'Gamecorner Budapesta 1', 'Maglódi út 14/B, 1106', 19, NULL)
    INTO magazin
    VALUES(312, 'Gamecorner Potcoava', 'Strada Brailei, Nr.17, Complex Comercial Potcoava de Aur',
12, NULL)
    INTO magazin
    VALUES(313, 'Gameshop Milano', 'Via Lorenteggio, 219, 20147 Milano MI', 20, NULL)
    INTO magazin
    VALUES(314, 'Gamecorner Budapesta 2', 'Könyves Kálmán krt. 12-14, 1097', 19, NULL)
    SELECT * FROM DUAL;

    INSERT ALL
    INTO joc
    VALUES(100290, 'Nier: Automata', 'Nier: Automata spune povestea androizilor 2B, 9S si A2 si
lupta lor pentru recapatarea distopiei controlate de masinarii puternice.', 'Action JRPG', 0, 18)
    INTO joc
    VALUES(40741, 'Persona 5 Royal', 'Scoate-ti masca si alatura-te Hotilor Fantoma si furturilor lor
magnifice, cum infiltreaza mintilor oamenilor corupti, facandu-i sa-si schimbe caile.', 'Turn-based
JRPG', 0, 16)
    INTO joc
    VALUES(40742, 'Persona 5 Strikers', NULL, 'Hack and Slash', 0, 16)
    INTO joc
    VALUES(460000, 'Rainbow Six: Siege', 'Rainbow Six: Siege este un shooter tactic de elita pe
echipe, unde planificarile superioare si executiile trimfa.', 'Team Shooter', 1, 18)
    INTO joc
    VALUES(101000, 'Shin Megami Tensei V', NULL, 'Turn-based JRPG', 0, 16)
    INTO joc
    VALUES(102000, 'Counter Strike 2', NULL, 'Team Shooter', 1, 18)
    INTO joc
    VALUES(103000, 'Brawl Stars', NULL, 'Topdown Shooter', 1, 7)
    INTO joc
    VALUES(104000, 'Minecraft Bedrock', NULL, 'Sandbox', 1, 7)
    INTO joc
    VALUES(105000, 'Ghost Recon: Wildlands', NULL, 'Open World', 1, 18)
```

```
INTO joc
VALUES(106000, 'Cyberpunk 2077', NULL, 'Action RPG', 0, 18)
INTO joc
VALUES(107000, 'The Witcher 3', NULL, 'Open World', 0, 18)
INTO joc
VALUES(108000, 'Geometry Dash', NULL, 'Platformer', 0, 3)
INTO joc
VALUES(25200, 'Soul Hackers 2', NULL, 'Turn-based JRPG', 0, 16)
INTO joc
VALUES(109000, 'Roblox', NULL, 'Sandbox', 1, 3)
SELECT * FROM DUAL;

INSERT ALL
INTO joc_online VALUES(460000, 3800000, 1, 1)
INTO joc_online VALUES(102000, 10000000, 0, 0)
INTO joc_online VALUES(103000, 1000000, 1, 1)
INTO joc_online VALUES(104000, 30000000, 1, 0)
INTO joc_online VALUES(105000, 20000, 1, 1)
INTO joc_online VALUES(109000, 17800000, 1, 1)
SELECT * FROM DUAL;
INSERT ALL
INTO platforma VALUES(1, 'Windows', 'PC')
INTO platforma VALUES(2, 'Playstation 4', 'Consola')
INTO platforma VALUES(3, 'Playstation 5', 'Consola')
INTO platforma VALUES(4, 'Xbox One', 'Consola')
INTO platforma VALUES(5, 'Xbox Series X', 'Consola')
INTO platforma VALUES(6, 'Xbox Series S', 'Consola')
INTO platforma VALUES(7, 'Nintendo Switch', 'Handheld')
INTO platforma VALUES(8, 'Steam Deck', 'Handheld')
INTO platforma VALUES(9, 'Android', 'Telefon')
INTO platforma VALUES(10, 'IOS', 'Telefon')
INTO platforma VALUES(11, 'Stadia', 'Cloud')
INTO platforma VALUES(12, 'Linux', 'PC')
SELECT * FROM DUAL;

INSERT ALL
INTO versiune VALUES(100290, 1, 40.8, TO_DATE('17-03-2017', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(100290, 2, 38, TO_DATE('23-02-2017', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(100290, 4, 38, TO_DATE('26-06-2018', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(100290, 7, 15, TO_DATE('06-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(100290, 8, 40.8, TO_DATE('17-03-2017', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(40741, 1, 39.3, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40741, 2, 33.2, TO_DATE('31-03-2020', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(40741, 3, 39.3, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40741, 4, 39.3, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40741, 5, 39.3, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40741, 6, 39.3, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40741, 7, 21.1, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40741, 8, 39.3, TO_DATE('21-10-2022', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(40742, 1, 31.5, TO_DATE('23-02-2021', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(40742, 2, 31.5, TO_DATE('23-02-2021', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(40742, 3, 31.5, TO_DATE('23-02-2021', 'dd-mm-yyyy'), 40)
```

```
INTO versiune VALUES(40742, 7, 18.2, TO_DATE('23-02-2021', 'dd-mm-yyyy'), 40)
INTO versiune VALUES(460000, 1, 54.5, TO_DATE('01-12-2015', 'dd-mm-yyyy'), 20)
INTO versiune VALUES(460000, 2, 40.2, TO_DATE('01-12-2015', 'dd-mm-yyyy'), 20)
INTO versiune VALUES(460000, 4, 40.2, TO_DATE('01-12-2015', 'dd-mm-yyyy'), 20)
INTO versiune VALUES(460000, 3, 50.7, TO_DATE('01-12-2020', 'dd-mm-yyyy'), 30)
INTO versiune VALUES(460000, 5, 50.7, TO_DATE('01-12-2020', 'dd-mm-yyyy'), 30)
INTO versiune VALUES(460000, 6, 50.7, TO_DATE('01-12-2020', 'dd-mm-yyyy'), 30)
INTO versiune VALUES(460000, 11, 0, TO_DATE('30-6-2021', 'dd-mm-yyyy'), 8)
INTO versiune VALUES(101000, 7, 16, TO_DATE('12-11-2021', 'dd-mm-yyyy'), 60)
INTO versiune VALUES(102000, 1, 30.7, TO_DATE('21-08-2012','dd-mm-yyyy'), 10)
INTO versiune VALUES(102000, 8, 30.7, TO_DATE('21-08-2012','dd-mm-yyyy'), 10)
INTO versiune VALUES(102000, 12, 30.7, TO_DATE('23-09-2014','dd-mm-yyyy'), 10)
INTO versiune VALUES(103000, 9, 2.99, TO_DATE('12-12-2018','dd-mm-yyyy'), 0)
INTO versiune VALUES(103000, 10, 2.99, TO_DATE('12-12-2018','dd-mm-yyyy'), 0)
INTO versiune VALUES(104000, 1, 2, TO_DATE('29-07-2015','dd-mm-yyyy'), 20)
INTO versiune VALUES(104000, 2, 2, TO_DATE('29-07-2015','dd-mm-yyyy'), 20)
INTO versiune VALUES(104000, 4, 2, TO_DATE('29-07-2015','dd-mm-yyyy'), 20)
INTO versiune VALUES(104000, 7, 2, TO_DATE('29-07-2015','dd-mm-yyyy'), 20)
INTO versiune VALUES(104000, 9, 0.33, TO_DATE('07-10-2011','dd-mm-yyyy'), 8)
INTO versiune VALUES(104000, 10, 0.33, TO_DATE('17-11-2011','dd-mm-yyyy'), 8)
INTO versiune VALUES(105000, 1, 76.3, TO_DATE('07-03-2017','dd-mm-yyyy'), 50)
INTO versiune VALUES(105000, 2, 76.3, TO_DATE('07-03-2017','dd-mm-yyyy'), 50)
INTO versiune VALUES(105000, 4, 76.3, TO_DATE('07-03-2017','dd-mm-yyyy'), 50)
INTO versiune VALUES(105000, 11, 0, TO_DATE('07-03-2017','dd-mm-yyyy'), 50)
INTO versiune VALUES(106000, 1, 70, TO_DATE('10-12-2020','dd-mm-yyyy'), 60)
INTO versiune VALUES(106000, 3, 70, TO_DATE('10-12-2020','dd-mm-yyyy'), 60)
INTO versiune VALUES(106000, 5, 70, TO_DATE('10-12-2020','dd-mm-yyyy'), 60)
INTO versiune VALUES(106000, 6, 70, TO_DATE('10-12-2020','dd-mm-yyyy'), 60)
INTO versiune VALUES(107000, 1, 43.7, TO_DATE('19-05-2015','dd-mm-yyyy'), 30)
INTO versiune VALUES(107000, 2, 43.7, TO_DATE('19-05-2015','dd-mm-yyyy'), 30)
INTO versiune VALUES(107000, 4, 43.7, TO_DATE('19-05-2015','dd-mm-yyyy'), 30)
INTO versiune VALUES(107000, 3, 43.7, TO_DATE('14-12-2022','dd-mm-yyyy'), 30)
INTO versiune VALUES(107000, 5, 43.7, TO_DATE('14-12-2022','dd-mm-yyyy'), 30)
INTO versiune VALUES(107000, 6, 43.7, TO_DATE('14-12-2022','dd-mm-yyyy'), 30)
INTO versiune VALUES(107000, 7, 19.2, TO_DATE('15-10-2019','dd-mm-yyyy'), 40)
INTO versiune VALUES(108000, 1, 0.18, TO_DATE('22-12-2014','dd-mm-yyyy'), 4)
INTO versiune VALUES(108000, 9, 0.18, TO_DATE('13-08-2013','dd-mm-yyyy'), 4)
INTO versiune VALUES(108000, 10, 0.18, TO_DATE('13-08-2013','dd-mm-yyyy'), 4)
INTO versiune VALUES(108000, 12, 0.18, TO_DATE('22-12-2014','dd-mm-yyyy'), 4)
INTO versiune VALUES(25200, 1, 20.5, TO_DATE('26-08-2022','dd-mm-yyyy'), 60)
INTO versiune VALUES(25200, 2, 20.5, TO_DATE('26-08-2022','dd-mm-yyyy'), 60)
INTO versiune VALUES(25200, 4, 20.5, TO_DATE('26-08-2022','dd-mm-yyyy'), 60)
INTO versiune VALUES(109000, 1, 1, TO_DATE('01-09-2006','dd-mm-yyyy'), 0)
INTO versiune VALUES(109000, 9, 1, TO_DATE('16-07-2014','dd-mm-yyyy'), 0)
INTO versiune VALUES(109000, 10, 1, TO_DATE('11-12-2012','dd-mm-yyyy'), 0)
INTO versiune VALUES(109000, 12, 1, TO_DATE('01-09-2006','dd-mm-yyyy'), 0)
SELECT * FROM DUAL;

INSERT ALL
INTO stoc VALUES(300, 100290, 1, 20)
INTO stoc VALUES(300, 100290, 2, 15)
INTO stoc VALUES(300, 40741, 4, 4)
```

```
INTO stoc VALUES(300, 40741, 1, 30)
INTO stoc VALUES(300, 40741, 7, 10)
INTO stoc VALUES(300, 101000, 7, 3)
INTO stoc VALUES(300, 102000, 1, 300)
INTO stoc VALUES(300, 106000, 1, 98)
INTO stoc VALUES(300, 25200, 1, 134)
INTO stoc VALUES(301, 101000, 7, 201)
INTO stoc VALUES(301, 102000, 1, 153)
INTO stoc VALUES(301, 103000, 10, 40)
INTO stoc VALUES(301, 104000, 1, 329)
INTO stoc VALUES(301, 105000, 1, 104)
INTO stoc VALUES(301, 106000, 1, 30)
INTO stoc VALUES(301, 107000, 1, 309)
INTO stoc VALUES(301, 108000, 1, 980)
INTO stoc VALUES(301, 109000, 1, 189)
INTO stoc VALUES(302, 100290, 1, 4)
INTO stoc VALUES(302, 100290, 4, 5)
INTO stoc VALUES(302, 100290, 7, 1)
INTO stoc VALUES(302, 101000, 7, 1)
INTO stoc VALUES(302, 104000, 1, 35)
INTO stoc VALUES(302, 105000, 1, 16)
INTO stoc VALUES(304, 460000, 1, 20)
INTO stoc VALUES(304, 460000, 2, 20)
INTO stoc VALUES(304, 460000, 4, 20)
INTO stoc VALUES(304, 105000, 1, 30)
INTO stoc VALUES(304, 105000, 2, 30)
INTO stoc VALUES(304, 105000, 4, 30)
INTO stoc VALUES(305, 40741, 7, 178)
INTO stoc VALUES(305, 40742, 7, 341)
INTO stoc VALUES(305, 101000, 7, 250)
INTO stoc VALUES(305, 25200, 2, 999)
INTO stoc VALUES(306, 100290, 1, 20)
INTO stoc VALUES(306, 100290, 2, 15)
INTO stoc VALUES(306, 40741, 4, 4)
INTO stoc VALUES(306, 40741, 1, 30)
INTO stoc VALUES(306, 40741, 7, 10)
INTO stoc VALUES(306, 101000, 7, 3)
INTO stoc VALUES(306, 102000, 1, 300)
INTO stoc VALUES(306, 106000, 1, 98)
INTO stoc VALUES(306, 25200, 1, 134)
INTO stoc VALUES(306, 102000, 8, 153)
INTO stoc VALUES(306, 103000, 10, 40)
INTO stoc VALUES(306, 104000, 1, 329)
INTO stoc VALUES(306, 105000, 1, 104)
INTO stoc VALUES(306, 106000, 3, 30)
INTO stoc VALUES(306, 107000, 1, 309)
INTO stoc VALUES(306, 108000, 1, 980)
INTO stoc VALUES(306, 109000, 1, 189)
INTO stoc VALUES(307, 101000, 7, 539)
INTO stoc VALUES(308, 100290, 1, 12)
INTO stoc VALUES(308, 101000, 7, 13)
INTO stoc VALUES(308, 105000, 1, 15)
```

```
INTO stoc VALUES(308, 108000, 10, 10)
INTO stoc VALUES(310, 100290, 1, 20)
INTO stoc VALUES(310, 100290, 2, 15)
INTO stoc VALUES(310, 40741, 4, 4)
INTO stoc VALUES(310, 40741, 1, 30)
INTO stoc VALUES(310, 40741, 7, 10)
INTO stoc VALUES(310, 101000, 7, 3)
INTO stoc VALUES(310, 102000, 1, 300)
INTO stoc VALUES(310, 106000, 1, 98)
INTO stoc VALUES(310, 25200, 1, 134)
INTO stoc VALUES(310, 102000, 8, 153)
INTO stoc VALUES(310, 103000, 10, 40)
INTO stoc VALUES(310, 104000, 1, 329)
INTO stoc VALUES(310, 105000, 1, 104)
INTO stoc VALUES(310, 106000, 3, 30)
INTO stoc VALUES(310, 107000, 1, 309)
INTO stoc VALUES(310, 108000, 1, 980)
INTO stoc VALUES(310, 109000, 1, 189)
INTO stoc VALUES(310, 100290, 4, 12)
INTO stoc VALUES(310, 108000, 10, 10)
INTO stoc VALUES(312, 101000, 7, 5)
INTO stoc VALUES(312, 104000, 9, 5)
INTO stoc VALUES(312, 103000, 9, 5)
INTO stoc VALUES(312, 108000, 9, 5)
INTO stoc VALUES(313, 109000, 1, 10)
INTO stoc VALUES(313, 109000, 9, 10)
INTO stoc VALUES(313, 109000, 10, 10)
INTO stoc VALUES(313, 109000, 12, 10)
SELECT * FROM DUAL;

INSERT ALL
INTO dezvoltator VALUES(100, 'PlatinumGames')
INTO dezvoltator VALUES(110, 'Atlus')
INTO dezvoltator VALUES(210, 'P Studio')
INTO dezvoltator VALUES(120, 'Omega Force')
INTO dezvoltator VALUES(130, 'Ubisoft')
INTO dezvoltator VALUES(140, 'Valve')
INTO dezvoltator VALUES(150, 'Hidden Path')
INTO dezvoltator VALUES(160, 'Supercell')
INTO dezvoltator VALUES(170, 'CD Projekt Red')
INTO dezvoltator VALUES(180, 'RobTop')
INTO dezvoltator VALUES(190, 'Roblox Corp.')
INTO dezvoltator VALUES(200, 'Mojang')
SELECT * FROM DUAL;

INSERT ALL
INTO editor VALUES(100, 'Square Enix')
INTO editor VALUES(210, 'Atlus')
INTO editor VALUES(220, 'Sega')
INTO editor VALUES(300, 'Ubisoft')
INTO editor VALUES(400, 'Valve')
INTO editor VALUES(500, 'Supercell')
```

```
 INTO editor VALUES(600, 'CD Projeckt')
 INTO editor VALUES(700, 'RobTop')
 INTO editor VALUES(800, 'Roblox Corp.')
 INTO editor VALUES(900, 'Mojang')
 SELECT * FROM DUAL;

 INSERT ALL
  INTO contribuie VALUES(100290, 100, 100)
  INTO contribuie VALUES(40741, 110, 210)
  INTO contribuie VALUES(40741, 210, 210)
  INTO contribuie VALUES(40741, 110, 220)
  INTO contribuie VALUES(40741, 210, 220)
  INTO contribuie VALUES(40742, 210, 210)
  INTO contribuie VALUES(40742, 120, 210)
  INTO contribuie VALUES(460000, 130, 300)
  INTO contribuie VALUES(101000, 110, 210)
  INTO contribuie VALUES(101000, 110, 220)
  INTO contribuie VALUES(102000, 140, 400)
  INTO contribuie VALUES(102000, 150, 400)
  INTO contribuie VALUES(103000, 160, 500)
  INTO contribuie VALUES(104000, 200, 900)
  INTO contribuie VALUES(105000, 130, 300)
  INTO contribuie VALUES(106000, 170, 600)
  INTO contribuie VALUES(107000, 170, 600)
  INTO contribuie VALUES(108000, 180, 700)
  INTO contribuie VALUES(25200, 110, 210)
  INTO contribuie VALUES(25200, 110, 220)
  INTO contribuie VALUES(109000, 190, 800)
 SELECT * FROM DUAL;

 INSERT ALL
  INTO limba VALUES('EN', 'Engleza')
  INTO limba VALUES('JP', 'Japoneza')
  INTO limba VALUES('RO', 'Romana')
  INTO limba VALUES('IT', 'Italiana')
  INTO limba VALUES('SP', 'Spaniola')
 SELECT * FROM DUAL;

 INSERT ALL
  INTO joc_in_limba VALUES(100290, 'EN', 1)
  INTO joc_in_limba VALUES(100290, 'JP', 1)
  INTO joc_in_limba VALUES(40741, 'EN', 1)
  INTO joc_in_limba VALUES(40741, 'JP', 1)
  INTO joc_in_limba VALUES(40741, 'IT', 0)
  INTO joc_in_limba VALUES(40741, 'SP', 0)
  INTO joc_in_limba VALUES(40742, 'EN', 1)
  INTO joc_in_limba VALUES(40742, 'JP', 1)
  INTO joc_in_limba VALUES(101000, 'EN', 1)
  INTO joc_in_limba VALUES(101000, 'JP', 1)
  INTO joc_in_limba VALUES(25200, 'EN', 1)
  INTO joc_in_limba VALUES(25200, 'JP', 1)
  INTO joc_in_limba VALUES(460000, 'EN', 1)
```

```
    INTO joc_in_limba VALUES(460000, 'JP', 0)
    INTO joc_in_limba VALUES(460000, 'IT', 0)
    INTO joc_in_limba VALUES(460000, 'SP', 0)
    INTO joc_in_limba VALUES(102000, 'EN', 1)
    INTO joc_in_limba VALUES(103000, 'EN', 1)
    INTO joc_in_limba VALUES(104000, 'EN', 0)
    INTO joc_in_limba VALUES(104000, 'JP', 0)
    INTO joc_in_limba VALUES(104000, 'RO', 0)
    INTO joc_in_limba VALUES(104000, 'SP', 0)
    INTO joc_in_limba VALUES(104000, 'IT', 0)
    INTO joc_in_limba VALUES(109000, 'EN', 0)
    INTO joc_in_limba VALUES(109000, 'JP', 0)
    INTO joc_in_limba VALUES(109000, 'RO', 0)
    INTO joc_in_limba VALUES(109000, 'SP', 0)
    INTO joc_in_limba VALUES(109000, 'IT', 0)
    INTO joc_in_limba VALUES(105000, 'EN', 1)
    INTO joc_in_limba VALUES(105000, 'IT', 1)
    INTO joc_in_limba VALUES(106000, 'EN', 1)
    INTO joc_in_limba VALUES(106000, 'SP', 1)
    INTO joc_in_limba VALUES(107000, 'EN', 1)
    INTO joc_in_limba VALUES(108000, 'EN', 0)
    SELECT * FROM DUAL;

    INSERT ALL
    INTO angajat
    VALUES(1000, 300, 'Floricel', 'Andreea Renata', 'renata@gmail.com', '+40756701314', 501,
TO_DATE('11-09-2006','dd-mm-yyyy'))
    INTO angajat
    VALUES(1001, 300, 'Ionescu', 'Ion', 'the_ion@gmail.com', '+40756742812', 1700, TO_DATE('11-
09-2019','dd-mm-yyyy'))
    INTO angajat
    VALUES(1002, 301, 'Pinduchi', 'Andreea Corina', 'corina@gmail.com', '+40756319217', 2050,
TO_DATE('07-05-2022','dd-mm-yyyy'))
    INTO angajat
    VALUES(1003, 301, 'Vasiliu', 'Ana', 'ana@ana.com', '+40726791325', 2000, TO_DATE('11-05-
2022','dd-mm-yyyy'))
    INTO angajat
    VALUES(1004, 301, 'Tenea', 'Andrada', 'andrada@gmail.com', '+40759312203', 1970,
TO_DATE('13-05-2023','dd-mm-yyyy'))
    INTO angajat
    VALUES(1005, 302, 'Burlacu', 'Eduard Daniel', NULL, '+40756771315', 2300, TO_DATE('10-10-
2022','dd-mm-yyyy'))
    INTO angajat
    VALUES(1006, 303, 'Smadu', 'Andrei', 'smadu@andrei.ro', NULL, 2500, TO_DATE('12-02-
2023','dd-mm-yyyy'))
    INTO angajat
    VALUES(1007, 304, 'Coman', 'Andrei David', 'coman@coman.com', '+40770649949', 2400,
TO_DATE('11-11-2022','dd-mm-yyyy'))
    INTO angajat(id_angajat, id_magazin, nume, prenume, email, telefon, salariu)
    VALUES(1008, 305, 'Sefer', 'Emanuel Adrian', 'emanu@gamil.com', '+31756771315', 3000)
    INTO angajat
```

```
    VALUES(1009, 306, 'Adamita', 'Stefan David', NULL, '+40756781315', 1000, TO_DATE('23-09-2020','dd-mm-yyyy'))
     INTO angajat
    VALUES(1010, 306, 'Melinte', 'Florin', NULL, NULL, 3000, TO_DATE('19-07-2019','dd-mm-yyyy'))
     INTO angajat
    VALUES(1011, 306, 'Berechet', 'Tudor', 'budor@gmail.com', '+40756781315', 2000, sysdate)
     INTO angajat
    VALUES(1012, 307, 'Williams', 'Will', 'will@will.will', NULL, 1000, TO_DATE('15-04-1998','dd-mm-yyyy'))
     INTO angajat
    VALUES(1013, 308, 'Giovanna', 'Giorno', NULL, NULL, 10000, TO_DATE('16-04-2001','dd-mm-yyyy'))
     INTO angajat(id_angajat, id_magazin, nume, salariu)
    VALUES(1014, 309, 'Theodore', 700)
     INTO angajat
    VALUES(1015, 310, 'White', 'Sandra', 'london1@gameshop.com', '+44000111222', 632, TO_DATE('13-05-2023','dd-mm-yyyy'))
     INTO angajat
    VALUES(1016, 310, 'Black', 'Jonathan Joe', 'london2@gameshop.com', '+44333111222', 701, TO_DATE('12-05-2023','dd-mm-yyyy'))
     INTO angajat
    VALUES(1017, 311, 'Popescu', 'Popa', 'popa@gameshop.com', '+40700111222', 502, TO_DATE('13-05-2023','dd-mm-yyyy'))
     INTO angajat
    VALUES(1018, 312, 'Lovin', 'Raluca', 'raluca@gameshop.com', '+40710111222', 3000, TO_DATE('09-05-2023','dd-mm-yyyy'))
     INTO angajat
    VALUES(1019, 313, 'Petru', 'Mihai', 'mihai@gameshop.com', '+40700611222', 1200, TO_DATE('08-05-2023','dd-mm-yyyy'))
     INTO angajat
    VALUES(1020, 314, 'Popescu', 'Ana', NULL, '+40700111262', 1029, TO_DATE('13-05-2023','dd-mm-yyyy'))
    SELECT * FROM DUAL;

    UPDATE magazin SET id_manager = 1001 WHERE id_magazin = 300;
    UPDATE magazin SET id_manager = 1002 WHERE id_magazin = 301;
    UPDATE magazin SET id_manager = 1005 WHERE id_magazin = 302;
    UPDATE magazin SET id_manager = 1006 WHERE id_magazin = 303;
    UPDATE magazin SET id_manager = 1007 WHERE id_magazin = 304;
    UPDATE magazin SET id_manager = 1008 WHERE id_magazin = 305;
    UPDATE magazin SET id_manager = 1010 WHERE id_magazin = 306;
    UPDATE magazin SET id_manager = 1013 WHERE id_magazin = 308;
    UPDATE magazin SET id_manager = 1014 WHERE id_magazin = 309;
    UPDATE magazin SET id_manager = 1015 WHERE id_magazin = 310;
    UPDATE magazin SET id_manager = 1018 WHERE id_magazin = 312;
    UPDATE magazin SET id_manager = 1019 WHERE id_magazin = 313;

    INSERT INTO client
    VALUES(secventa_client.nextval, 'Cazan', 'George', 'george@gmail.com', '+40763192452');
    INSERT INTO client
    VALUES(secventa_client.nextval, 'Onodi', 'Paul', 'paul@gmail.com', '+40763693212');
    INSERT INTO client
```

```
    VALUES(secventa_client.nextval, 'Jansen', 'Liam', NULL, '+31756821387');
    INSERT INTO client
    VALUES(secventa_client.nextval, 'Greyson', 'Henry', 'henry@yahoo.com', NULL);
    INSERT INTO client
    VALUES(secventa_client.nextval, 'Cocu', 'Elena', 'elena@cocu.com', '+40769695674');
    INSERT INTO client
    VALUES(secventa_client.nextval, 'De Lucca', 'Antonio', NULL, NULL);
    INSERT INTO client
    VALUES(secventa_client.nextval, 'Nagy', 'Zoltan', 'nagy@zoltan.com', '+40711112222');

    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000000, 1001, TO_DATE('11-05-2023','dd-mm-yyyy'),
'cash');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000000, 1002, TO_DATE('09-05-2023','dd-mm-yyyy'),
'card');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000001, 1005, TO_DATE('05-05-2023','dd-mm-yyyy'),
'rate');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000003, 1012, TO_DATE('02-05-2023','dd-mm-yyyy'),
'card');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000003, 1015, TO_DATE('03-05-2023','dd-mm-yyyy'),
'card');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000002, 1008, TO_DATE('13-05-2023','dd-mm-yyyy'),
'mixt');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000000, 1006, TO_DATE('13-05-2023','dd-mm-yyyy'),
'cash');
    INSERT INTO comanda(id_comanda, id_client, id_angajat)
    VALUES(secventa_comanda.nextval, 1000004, 1007);
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000005, 1013, TO_DATE('15-05-2023','dd-mm-yyyy'),
'cash');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000006, 1020, TO_DATE('11-05-2023','dd-mm-yyyy'),
'card');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000001, 1018, TO_DATE('07-05-2023','dd-mm-yyyy'),
'cash');
    INSERT INTO comanda
    VALUES(secventa_comanda.nextval, 1000005, 1019, TO_DATE('04-05-2023','dd-mm-yyyy'),
'cash');

    INSERT ALL
     INTO comanda_online VALUES(1037182341, 'Str. Preciziei 24, Bucure?ti 062204', TO_DATE('10-
05-2023','dd-mm-yyyy'))
     INTO comanda_online VALUES(1111547023, 'Wandsworth Bridge Rd., London SW6 2NY',
TO_DATE('05-05-2023','dd-mm-yyyy'))
```

```
    INTO comanda_online VALUES(1185911705, 'Vijzelstraat 15, 1017 HD Amsterdam', TO_DATE('14-05-2023','dd-mm-yyyy'))
    INTO comanda_online VALUES(1409005751, 'Parcheggio pubblico, Via Borgo Palazzo, 217, 24125 Bergamo BG', TO_DATE('20-05-2023','dd-mm-yyyy'))
    INTO comanda_online VALUES(1371823410, 'Strada General Ioan Dragalina 10, Gala?i 800402', TO_DATE('07-05-2023','dd-mm-yyyy'))
SELECT * FROM DUAL;
INSERT ALL
 INTO comanda_contine VALUES(101000, 7, 1000000000, 2)
 INTO comanda_contine VALUES(101000, 7, 1037182341, 1)
 INTO comanda_contine VALUES(40741, 7, 1037182341, 1)
 INTO comanda_contine VALUES(100290, 7, 1037182341, 1)
 INTO comanda_contine VALUES(460000, 1, 1074364682, 1)
 INTO comanda_contine VALUES(105000, 1, 1074364682, 1)
 INTO comanda_contine VALUES(25200, 2, 1111547023, 5)
 INTO comanda_contine VALUES(104000, 9, 1148729364, 12)
 INTO comanda_contine VALUES(104000, 10, 1148729364, 3)
 INTO comanda_contine VALUES(106000, 1, 1185911705, 1)
 INTO comanda_contine VALUES(101000, 7, 1223094046, 1)
 INTO comanda_contine VALUES(101000, 7, 1260276387, 1)
 INTO comanda_contine VALUES(102000, 1, 1260276387, 1)
 INTO comanda_contine VALUES(103000, 9, 1260276387, 1)
 INTO comanda_contine VALUES(40741, 1, 1297458728, 1)
 INTO comanda_contine VALUES(40742, 1, 1297458728, 1)
 INTO comanda_contine VALUES(100290, 1, 1334641069, 1)
 INTO comanda_contine VALUES(100290, 1, 1371823410, 2)
 INTO comanda_contine VALUES(102000, 1, 1371823410, 2)
 INTO comanda_contine VALUES(106000, 1, 1371823410, 2)
 INTO comanda_contine VALUES(40741, 1, 1371823410, 2)
 INTO comanda_contine VALUES(107000, 1, 1371823410, 2)
 INTO comanda_contine VALUES(104000, 1, 1371823410, 2)
 INTO comanda_contine VALUES(101000, 7, 1409005751, 20)
SELECT * FROM DUAL;

COMMIT;
```

5 rows inserted.	5 rows inserted.	1 row inserted.	
10 rows inserted.	34 rows inserted.	1 row inserted.	
15 rows inserted.	21 rows inserted.	1 row inserted.	
14 rows inserted.	1 row updated.	1 row inserted.	1 row inserted.
6 rows inserted.	1 row updated.	1 row inserted.	1 row inserted.
12 rows inserted.	1 row updated.	1 row inserted.	1 row inserted.
62 rows inserted.	1 row updated.	1 row inserted.	5 rows inserted.
83 rows inserted.	1 row updated.	1 row inserted.	24 rows inserted.
12 rows inserted.	1 row updated.	1 row inserted.	Commit complete.
10 rows inserted.	1 row updated.	1 row inserted.	
21 rows inserted.	1 row updated.	1 row inserted.	
5 rows inserted.	1 row updated.	1 row inserted.	
34 rows inserted.	1 row updated.	1 row inserted.	
21 rows inserted.	1 row updated.	1 row inserted.	
	1 row updated.		

6. Subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate.

Să se ieftineasca cu 20% cele mai putin cumparate 5 jocuri pentru cele 3 platforme cu cele mai puține jocuri vandute pentru.

```
CREATE OR REPLACE PROCEDURE ieftinire_jocuri
  IS
```

```
    TYPE tablou_indexat IS TABLE OF joc%ROWTYPE
                INDEX BY BINARY_INTEGER;
    TYPE tablou_imbricat IS TABLE OF versiune%ROWTYPE;
    TYPE vector IS VARRAY(3) OF platforma%ROWTYPE;

    t_jocuri tablou_indexat;
    t_platforme vector := vector();
    t_versiuni tablou_imbricat := tablou_imbricat();
    k INTEGER;
    v_nume_joc joc.titlu%TYPE;
    v_nume_platforma platforma.nume_platforma%TYPE;
    v_pret versiune.pret%TYPE;
    v_ver versiune%ROWTYPE;
    v_count NUMBER;

  BEGIN
    SELECT *
    BULK COLLECT INTO t_jocuri
    FROM joc
    WHERE id_joc IN (SELECT id_joc
              FROM (SELECT id_joc
                FROM comanda_contine
                GROUP BY id_joc
                ORDER BY SUM(numar))
              WHERE ROWNUM <= 5);

    SELECT *
    BULK COLLECT INTO t_platforme
    FROM platforma
    WHERE id_platforma IN (SELECT id_platforma
              FROM (SELECT id_platforma
                FROM comanda_contine
                GROUP BY id_platforma
                ORDER BY SUM(numar))
              WHERE ROWNUM <= 3);

    FOR i in t_jocuri.FIRST..t_jocuri.LAST LOOP
      FOR j in t_platforme.FIRST..t_platforme.LAST LOOP
        SELECT COUNT(*)
        INTO v_count
        FROM versiune
        WHERE id_joc = t_jocuri(i).id_joc AND id_platforma = t_platforme(j).id_platforma;
        IF v_count = 1 THEN
          SELECT *
          INTO v_ver
          FROM versiune
          WHERE id_joc = t_jocuri(i).id_joc AND id_platforma = t_platforme(j).id_platforma;
          t_versiuni.EXTEND;
          t_versiuni(t_versiuni.LAST) := v_ver;
        END IF;
      END LOOP;
    END LOOP;
```

```
    k := t_versiuni.FIRST;
    WHILE k <= t_versiuni.LAST LOOP
      t_versiuni(k).pret := t_versiuni(k).pret * 4 / 5;
      k := t_versiuni.NEXT(k);
    END LOOP;

    k := t_versiuni.FIRST;
    WHILE k <= t_versiuni.LAST LOOP
      SELECT titlu
      INTO v_nume_joc
      FROM joc
      WHERE id_joc = t_versiuni(k).id_joc;

      SELECT nume_platforma
      INTO v_nume_platforma
      FROM platforma
      WHERE id_platforma = t_versiuni(k).id_platforma;

      SELECT pret
      INTO v_pret
      FROM versiune
      WHERE id_joc = t_versiuni(k).id_joc AND id_platforma = t_versiuni(k).id_platforma;

      DBMS_OUTPUT.PUT_LINE('Jocul ' || v_nume_joc || ' de pe platforma ' || v_nume_platforma
|| ' avea pretul ' || v_pret || ' si a scazut la ' || t_versiuni(k).pret || '.');

      UPDATE versiune
      SET pret = t_versiuni(k).pret
      WHERE id_joc = t_versiuni(k).id_joc AND id_platforma = t_versiuni(k).id_platforma;

      k := t_versiuni.NEXT(k);
    END LOOP;
END;
/
```

```
Procedure IEFTINIRE_JOCURI compiled
```

```
Jocul Persona 5 Strikers de pe platforma Playstation 4 avea pretul 40 si a scazut la 32.
Jocul Brawl Stars de pe platforma Android avea pretul 0 si a scazut la 0.
Jocul Brawl Stars de pe platforma IOS avea pretul 0 si a scazut la 0.
Jocul Ghost Recon: Wildlands de pe platforma Playstation 4 avea pretul 50 si a scazut la 40.
Jocul The Witcher 3 de pe platforma Playstation 4 avea pretul 30 si a scazut la 24.
Jocul Rainbow Six: Siege de pe platforma Playstation 4 avea pretul 20 si a scazut la 16.


PL/SQL procedure successfully completed.
```

7. Subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat, dependent de celălalt cursor.

Să se afiseze date despre un magazin (nume, adresa, numar angajati, media salariilor) si lista angajatilor săi.

```
CREATE OR REPLACE PROCEDURE date_magazin
  IS
    TYPE refcursor IS REF CURSOR RETURN magazin%ROWTYPE;
    c_magazine refcursor;
    CURSOR c_angajati (id_mag magazin.id_magazin%TYPE) IS
      SELECT *
      FROM angajat
      WHERE id_magazin = id_mag;
    v_mag magazin%ROWTYPE;
    v_nr_ang NUMBER;
    v_avg_sal NUMBER;
  BEGIN
    OPEN c_magazine FOR
      SELECT * FROM magazin;
    LOOP
      FETCH c_magazine INTO v_mag;
      EXIT WHEN c_magazine%NOTFOUND;

      DBMS_OUTPUT.PUT_LINE('---------------------------------------');
      DBMS_OUTPUT.PUT_LINE(v_mag.nume_magazin);
      DBMS_OUTPUT.PUT_LINE('Adresa: ' || v_mag.adresa);
      DBMS_OUTPUT.PUT_LINE('---------------------------------------');

      v_nr_ang := 0;
      v_avg_sal := 0;

      FOR v_ang IN c_angajati(v_mag.id_magazin) LOOP
        DBMS_OUTPUT.PUT_LINE(v_ang.nume || ' ' || v_ang.prenume || ' ' || v_ang.salariu);
        v_nr_ang := v_nr_ang + 1;
        v_avg_sal := v_avg_sal + v_ang.salariu;
      END LOOP;

      IF v_nr_ang = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Magazinul nu are angajati');
      ELSE
        v_avg_sal := v_avg_sal / v_nr_ang;
        DBMS_OUTPUT.PUT_LINE('---------------------------------------');
        DBMS_OUTPUT.PUT_LINE('Numarul de angajati: ' || v_nr_ang);
        DBMS_OUTPUT.PUT_LINE('Salariul mediu: ' || v_avg_sal);
      END IF;
      DBMS_OUTPUT.PUT_LINE('---------------------------------------');
      DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
    CLOSE c_magazine;
```

```
  END;
/
```

```
Procedure DATE_MAGAZIN compiled
```

```
--------------------------------------
Gameshop AFI
Adresa: Bulevardul General Paul Teodorescu 4
--------------------------------------
Floricel Andreea Renata 501
Ionescu Ion 1700
--------------------------------------
Numarul de angajati: 2
Salariul mediu: 1100.5
--------------------------------------

--------------------------------------
Gameshop Centru Vechi
Adresa: Strada Lipscani 55, Bucure?ti
--------------------------------------
Pinduchi Andreea Corina 2050
Vasiliu Ana 2000
Tenea Andrada 1970
--------------------------------------
Numarul de angajati: 3
Salariul mediu: 2006.6666666666666666666666666666666666667
--------------------------------------

--------------------------------------
Gamecorner Tiglina
Adresa: Strada Br?ilei nr. 163, 800309
--------------------------------------
Burlacu Eduard Daniel 2300
--------------------------------------
Numarul de angajati: 1
Salariul mediu: 2300
--------------------------------------
```

```
-------------------------------------
Gamecorner Gorjului
Adresa: Bulevardul Iuliu Maniu 67
-------------------------------------
Smadu Andrei 2500
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 2500
-------------------------------------

-------------------------------------
Gameshop Cluj
Adresa: Strada Alexandru Vaida Voevod 59, 400436
-------------------------------------
Coman Andrei David 2400
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 2400
-------------------------------------

-------------------------------------
Gameshop Amsterdam
Adresa:
-------------------------------------
Sefer Emanuel Adrian 3000
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 3000
-------------------------------------

-------------------------------------
Megagame Brasov
Adresa: Bulevardul 15 Noiembrie 78
-------------------------------------
Adamita Stefan David 1000
Melinte Florin 3000
Berechet Tudor 2000
-------------------------------------
```

```
-------------------------------------
Gamecorner Walworth
Adresa: East St, London SE17 1EL
-------------------------------------
Williams Will 1000
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 1000
-------------------------------------

-------------------------------------
Gameshop Roma
Adresa:
-------------------------------------
Giovanna Giorno 10000
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 10000
-------------------------------------

-------------------------------------
Gameshop Rotterdam
Adresa: Willem Ruyslaan 225, 3063 ER
-------------------------------------
Theodore  700
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 700
-------------------------------------
```

```
-------------------------------------
Megashop Londra
Adresa: 12 Walbrook, London EC4N 8AA
-------------------------------------
White Sandra 632
Black Jonathan Joe 701
-------------------------------------
Numarul de angajati: 2
Salariul mediu: 666.5
-------------------------------------

-------------------------------------
Gamecorner Budapesta 1
Adresa: Maglódi út 14/B, 1106
-------------------------------------
Popescu Popa 502
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 502
-------------------------------------

-------------------------------------
Gamecorner Potcoava
Adresa: Strada Brailei, Nr.17, Complex Comercial Potcoava de Aur
-------------------------------------
Lovin Raluca 3000
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 3000
-------------------------------------

-------------------------------------
Gameshop Milano
Adresa: Via Lorenteggio, 219, 20147 Milano MI
-------------------------------------
Petru Mihai 1200
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 1200
-------------------------------------

-------------------------------------
Gamecorner Budapesta 2
Adresa: Könyves Kálmán krt. 12-14, 1097
-------------------------------------
Popescu Ana 1029
-------------------------------------
Numarul de angajati: 1
Salariul mediu: 1029
-------------------------------------



PL/SQL procedure successfully completed.
```

8. Subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții proprii.

Pentru un magazin dat să se returneze numărul total de jocuri vândute de angajații săi.

```
CREATE OR REPLACE FUNCTION vanzari_magazin
  (v_nume_mag magazin.nume_magazin%TYPE)
RETURN NUMBER IS
    v_id_mag magazin.id_magazin%TYPE;
    v_nr_ang NUMBER;
    v_nr_comenzi NUMBER;
    v_joc_vand NUMBER;
    NO_EMPLOYEES EXCEPTION;
    NO_SALES EXCEPTION;
  BEGIN
    SELECT id_magazin
    INTO v_id_mag
    FROM magazin
    WHERE UPPER(nume_magazin) LIKE ('%' || UPPER(v_nume_mag) || '%');

    SELECT COUNT(*)
    INTO v_nr_ang
    FROM angajat
    WHERE id_magazin = v_id_mag;
    IF v_nr_ang = 0 THEN
      RAISE NO_EMPLOYEES;
    END IF;

    SELECT COUNT(*)
    INTO v_nr_comenzi
    FROM comanda
    WHERE id_angajat IN (SELECT id_angajat FROM angajat
                WHERE id_magazin = v_id_mag);
    IF v_nr_comenzi = 0 THEN
      RAISE NO_SALES;
    END IF;

    SELECT SUM(numar)
    INTO v_joc_vand
    FROM comanda_contine JOIN comanda USING (id_comanda)
      JOIN angajat USING (id_angajat)
    WHERE id_magazin = v_id_mag;

    RETURN v_joc_vand;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Nu exista magazin cu acel nume.');
      RETURN -1;
    WHEN TOO_MANY_ROWS THEN
      DBMS_OUTPUT.PUT_LINE('Exista mai multe magazine cu acel nume.');
```

```
        RETURN -1;
      WHEN NO_EMPLOYEES THEN
        DBMS_OUTPUT.PUT_LINE('Magazinul nu are angajati.');
        RETURN 0;
      WHEN NO_SALES THEN
        DBMS_OUTPUT.PUT_LINE('Magazinul nu are vanzari.');
        RETURN 0;
   END;
/
```

```
Function VANZARI_MAGAZIN compiled
```

Dacă rulăm următoarea secvență de cod, rezultatele sunt:

```
BEGIN
   DBMS_OUTPUT.PUT_LINE(vanzari_magazin('Gameshop Galati'));
   DBMS_OUTPUT.PUT_LINE(vanzari_magazin('Gameshop'));
   DBMS_OUTPUT.PUT_LINE(vanzari_magazin('Brasov'));
   DBMS_OUTPUT.PUT_LINE(vanzari_magazin('Londra'));
END;
/
```

```
Nu exista magazin cu acel nume.
-1
Exista mai multe magazine cu acel nume.
-1
Magazinul nu are vanzari.
0
15


PL/SQL procedure successfully completed.
```

Dacă ar exista un magazin nou fără angajați, rezultatul ar fi:

```
INSERT INTO magazin
VALUES(315, 'Gamecorner Budapesta 3', 'Könyves Kálmán krt. 12-14, 1097', 19, NULL);

BEGIN
   DBMS_OUTPUT.PUT_LINE(vanzari_magazin('Gamecorner Budapesta 3'));
END;
/

ROLLBACK;
```

```
1 row inserted.

Magazinul nu are angajati.
0


PL/SQL procedure successfully completed.


Rollback complete.
```

9. Subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, incluzând excepțiile NO_DATA_FOUND și TOO_MANY_ROWS.

Să se afișeze stocul și numărul de jocuri vândute pentru un dezvoltator și toate orasele dintr-o anumită țară.

```
CREATE OR REPLACE PROCEDURE stoc_dezvoltator
  (v_nume_dez dezvoltator.nume_dezvoltator%TYPE,
  v_nume_tara tara.nume_tara%TYPE)
  IS
    TYPE tablou_orase IS TABLE OF oras%ROWTYPE;

    v_id_dez dezvoltator.id_dezvoltator%TYPE;
    v_id_tara tara.id_tara%TYPE;

    v_nr_orase NUMBER;
    v_stoc NUMBER;
    v_vanzari NUMBER;
    t_orase tablou_orase;
    k NUMBER;

    NO_CITIES EXCEPTION;
  BEGIN
    BEGIN
      SELECT id_dezvoltator
      INTO v_id_dez
      FROM dezvoltator
      WHERE UPPER(nume_dezvoltator) LIKE ('%' || UPPER(v_nume_dez) || '%');
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista dezvoltator cu acel nume.');
        RAISE;
      WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi dezvoltatori cu acel nume.');
        RAISE;
    END;
```

```
BEGIN
   SELECT id_tara
   INTO v_id_tara
   FROM tara
   WHERE UPPER(nume_tara) LIKE ('%' || UPPER(v_nume_tara) || '%');
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Nu exista tara cu acel nume.');
      RAISE;
   WHEN TOO_MANY_ROWS THEN
      DBMS_OUTPUT.PUT_LINE('Exista mai multe tari cu acel nume.');
      RAISE;
END;

SELECT COUNT(*)
INTO v_nr_orase
FROM oras
WHERE id_tara = v_id_tara;
IF v_nr_orase = 0 THEN
   RAISE NO_CITIES;
END IF;

SELECT *
BULK COLLECT INTO t_orase
FROM oras
WHERE id_tara = v_id_tara;

k := t_orase.FIRST;
WHILE k <= t_orase.LAST LOOP
   SELECT SUM(nr_produse)
   INTO v_stoc
   FROM contribuie JOIN stoc USING (id_joc)
      JOIN magazin USING (id_magazin)
   WHERE id_dezvoltator = v_id_dez
      AND id_oras = t_orase(k).id_oras;

   SELECT SUM(numar)
   INTO v_vanzari
   FROM contribuie JOIN comanda_contine USING (id_joc)
      JOIN comanda USING (id_comanda)
      JOIN angajat USING (id_angajat)
      JOIN magazin USING (id_magazin)
   WHERE id_dezvoltator = v_id_dez
      AND id_oras = t_orase(k).id_oras;

   IF v_stoc IS NULL THEN
      v_stoc := 0;
   END IF;
   IF v_vanzari IS NULL THEN
      v_vanzari := 0;
   END IF;
```

```
        DBMS_OUTPUT.PUT_LINE('In orasul ' || t_orase(k).nume_oras || ' dezvolatorul are stocul ' ||
v_stoc || ' si suma totala de vanzari ' || v_vanzari || '.');
        k := t_orase.NEXT(k);
      END LOOP;
   EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Eroare NO_DATA_FOUND');
      WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Eroare TOO_MANY_ROWS');
      WHEN NO_CITIES THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista orase in acea tara.');
   END;
/
```

```
Procedure STOC_DEZVOLTATOR compiled
```

Dacă rulăm următoarea secvență de cod, rezultatele sunt:

```
BEGIN
   stoc_dezvoltator('RGG Studios', 'Romania');
   stoc_dezvoltator('Rob', 'Romania');
   stoc_dezvoltator('Atlus', 'Bulgaria');
   stoc_dezvoltator('Atlus', 'ia');

   stoc_dezvoltator('Atlus', 'Romania');
END;
/
```

```
Nu exista dezvoltator cu acel nume.
Eroare NO_DATA_FOUND
Exista mai multi dezvoltatori cu acel nume.
Eroare TOO_MANY_ROWS
Nu exista tara cu acel nume.
Eroare NO_DATA_FOUND
Exista mai multe tari cu acel nume.
Eroare TOO_MANY_ROWS
In orasul Bucuresti dezvolatorul are stocul 764 si suma totala de vanzari 10.
In orasul Galati dezvolatorul are stocul 12 si suma totala de vanzari 4.
In orasul Cluj dezvolatorul are stocul 0 si suma totala de vanzari 2.
In orasul Brasov dezvolatorul are stocul 362 si suma totala de vanzari 0.


PL/SQL procedure successfully completed.
```

Dacă ar exista o țară fără orașe, rezultatul ar fi:

```
INSERT INTO tara
VALUES ('PO', 'Polonia');
BEGIN
   stoc_dezvoltator('Atlus', 'Polonia');
END;
/
ROLLBACK;
```

```
1 row inserted.

Nu exista orase in acea tara.


PL/SQL procedure successfully completed.


Rollback complete.
```

10. Trigger de tip LMD la nivel de comandă.

Să nu se poată face o comandă în afara programului (8 - 22 de luni până vineri, 10 - 20 sâmbată, 10 - 16 duminică)

```
CREATE OR REPLACE TRIGGER comanda_in_program
   BEFORE INSERT ON comanda
DECLARE
   v_zi_sapt NUMBER;
   v_ora NUMBER;
BEGIN
   v_zi_sapt := TO_NUMBER(TO_CHAR(SYSDATE, 'D'));
   v_ora := TO_NUMBER(TO_CHAR(SYSDATE, 'HH24'));
   IF (v_zi_sapt BETWEEN 1 AND 5) AND (v_ora < 8 OR v_ora >= 22) THEN
      RAISE_APPLICATION_ERROR(-20001, 'Comanda facuta in afara programului');
   ELSIF (v_zi_sapt = 6) AND (v_ora < 10 OR v_ora >= 20) THEN
      RAISE_APPLICATION_ERROR(-20001, 'Comanda facuta in afara programului');
   ELSIF (v_zi_sapt = 7) AND (v_ora < 10 OR v_ora >= 16) THEN
      RAISE_APPLICATION_ERROR(-20001, 'Comanda facuta in afara programului');
   END IF;
END;
/
```

```
Trigger COMANDA_IN_PROGRAM compiled
```

Dacă încercăm să inserăm o comandă în diferite momente ale săptămânii, rezultatele sunt:

```
1 row inserted.
```

10:25
10/01/2024

```
Error starting at line : 377 in command -
INSERT INTO comanda
VALUES(17311230, 1000000, 1000, SYSDATE, 'CASH')
Error report -
ORA-20001: Comanda facuta in afara programului
ORA-06512: at "C##TEST6.COMANDA_IN_PROGRAM", line 8
ORA-04088: error during execution of trigger 'C##TEST6.COMANDA_IN_PROGRAM'
```

06:27
10/01/2024

```
Error starting at line : 377 in command -
INSERT INTO comanda
VALUES(17311230, 1000000, 1000, SYSDATE, 'CASH')
Error report -
ORA-20001: Comanda facuta in afara programului
ORA-06512: at "C##TEST6.COMANDA_IN_PROGRAM", line 12
ORA-04088: error during execution of trigger 'C##TEST6.COMANDA_IN_PROGRAM'
```

17:27
07/01/2024

```
Error starting at line : 377 in command -
INSERT INTO comanda
VALUES(17311230, 1000000, 1000, SYSDATE, 'CASH')
Error report -
ORA-20001: Comanda facuta in afara programului
ORA-06512: at "C##TEST6.COMANDA_IN_PROGRAM", line 10
ORA-04088: error during execution of trigger 'C##TEST6.COMANDA_IN_PROGRAM'
```

20:28
06/01/2024

11. Trigger de tip LMD la nivel de linie.

Să se valideze datele la inserarea, updatarea și ștergerea unui angajat (telefon și email valide, dată de angajare după începerea companiei, salariu peste 500, salariu să nu scadă sub 75% din cel anterior, să nu fie manager la un magazin când e șters).

```
CREATE OR REPLACE TRIGGER validare_angajat
  BEFORE INSERT OR UPDATE OR DELETE ON angajat
  FOR EACH ROW
DECLARE
  v_manager NUMBER;
BEGIN
  IF INSERTING OR UPDATING THEN
    IF (:NEW.telefon IS NOT NULL) AND (:NEW.telefon NOT LIKE '+%') THEN
      RAISE_APPLICATION_ERROR(-20002, 'Numar de telefon invalid');
    ELSIF (:NEW.email IS NOT NULL) AND (:NEW.email NOT LIKE '%_@__%.__%') THEN
      RAISE_APPLICATION_ERROR(-20003, 'Email invalid');
    ELSIF :NEW.salariu < 500 THEN
      RAISE_APPLICATION_ERROR(-20004, 'Salariu invalid');
    END IF;
  END IF;

  IF UPDATING THEN
```

```
    IF :NEW.salariu < :OLD.salariu * 3 / 4 THEN
      RAISE_APPLICATION_ERROR(-20004, 'Salariu invalid');
    END IF;
  ELSIF DELETING THEN
    SELECT COUNT(*)
    INTO v_manager
    FROM magazin
    WHERE id_manager = :OLD.id_angajat;

    IF v_manager != 0 THEN
      RAISE_APPLICATION_ERROR(-20005, 'Angajatul inca este manager intr-un magazin');
    END IF;
  END IF;
END;
/
```

```
Trigger VALIDARE_ANGAJAT compiled
```

Dacă rulăm următoarea secvență de cod, rezultatele sunt:

```
DELETE FROM angajat WHERE id_angajat = 1001;

UPDATE angajat
SET salariu = 400
WHERE id_angajat = 1001;

UPDATE angajat
SET salariu = 600
WHERE id_angajat = 1001;

UPDATE angajat
SET telefon = '012834114'
WHERE id_angajat = 1001;

UPDATE angajat
SET email = '@i.c'
WHERE id_angajat = 1001;
```

```
Error starting at line : 421 in command -
DELETE FROM angajat WHERE id_angajat = 1001
Error report -
ORA-20005: Angajatul inca este manager intr-un magazin
ORA-06512: at "C##TEST6.VALIDARE_ANGAJAT", line 25
ORA-04088: error during execution of trigger 'C##TEST6.VALIDARE_ANGAJAT'


Error starting at line : 423 in command -
UPDATE angajat
SET salariu = 400
WHERE id_angajat = 1001
Error report -
ORA-20004: Salariu invalid
ORA-06512: at "C##TEST6.VALIDARE_ANGAJAT", line 10
ORA-04088: error during execution of trigger 'C##TEST6.VALIDARE_ANGAJAT'


Error starting at line : 427 in command -
UPDATE angajat
SET salariu = 600
WHERE id_angajat = 1001
Error report -
ORA-20004: Salariu invalid
ORA-06512: at "C##TEST6.VALIDARE_ANGAJAT", line 16
ORA-04088: error during execution of trigger 'C##TEST6.VALIDARE_ANGAJAT'



Error starting at line : 431 in command -
UPDATE angajat
SET telefon = '012834114'
WHERE id_angajat = 1001
Error report -
ORA-20002: Numar de telefon invalid
ORA-06512: at "C##TEST6.VALIDARE_ANGAJAT", line 6
ORA-04088: error during execution of trigger 'C##TEST6.VALIDARE_ANGAJAT'


Error starting at line : 435 in command -
UPDATE angajat
SET email = '@i.c'
WHERE id_angajat = 1001
Error report -
ORA-20003: Email invalid
ORA-06512: at "C##TEST6.VALIDARE_ANGAJAT", line 8
ORA-04088: error during execution of trigger 'C##TEST6.VALIDARE_ANGAJAT'
```

12. Trigger de tip LDD

Să se interzică crearea, alterarea sau ștergerea de tabele noi.

```
CREATE OR REPLACE TRIGGER fara_schimbari
   BEFORE CREATE OR DROP OR ALTER ON SCHEMA
BEGIN
   IF SYS.DICTIONARY_OBJ_TYPE LIKE 'TABLE' THEN
      IF SYS.SYSEVENT LIKE 'CREATE' THEN
         RAISE_APPLICATION_ERROR(-20006, 'Nu aveti voie sa mai adaugati tabele');
```

```
    ELSIF SYS.SYSEVENT LIKE 'DROP' THEN
      RAISE_APPLICATION_ERROR(-20007, 'Nu aveti voie sa stergeti tabele');
    ELSIF SYS.SYSEVENT LIKE 'ALTER' THEN
      RAISE_APPLICATION_ERROR(-20008, 'Nu aveti voie sa alterati tabele');
    END IF;
  END IF;
END;
/
```

```
Trigger FARA_SCHIMBARI compiled
```

Dacă se încearcă inserarea, ștergerea sau alterarea unui tabel, rezultatul este:

```
Trigger FARA_SCHIMBARI compiled


Error starting at line : 462 in command -
CREATE TABLE tabel (coloana_1 NUMBER(2))
Error report -
ORA-04088: error during execution of trigger 'C##TEST6.FARA_SCHIMBARI'
ORA-00604: error occurred at recursive SQL level 1
ORA-20006: Nu aveti voie sa mai adaugati tabele
ORA-06512: at line 4
04088. 00000 -  "error during execution of trigger '%s.%s'"
*Cause:    A runtime error occurred during execution of a trigger.
*Action:   Check the triggers which were involved in the operation.

Error starting at line : 463 in command -
ALTER TABLE angajat ADD (coloana_2 NUMBER(2))
Error report -
ORA-04088: error during execution of trigger 'C##TEST6.FARA_SCHIMBARI'
ORA-00604: error occurred at recursive SQL level 1
ORA-20008: Nu aveti voie sa alterati tabele
ORA-06512: at line 8
04088. 00000 -  "error during execution of trigger '%s.%s'"
*Cause:    A runtime error occurred during execution of a trigger.
*Action:   Check the triggers which were involved in the operation.

Error starting at line : 464 in command -
DROP TABLE angajat
Error report -
ORA-04088: error during execution of trigger 'C##TEST6.FARA_SCHIMBARI'
ORA-00604: error occurred at recursive SQL level 1
ORA-20007: Nu aveti voie sa stergeti tabele
ORA-06512: at line 6
04088. 00000 -  "error during execution of trigger '%s.%s'"
*Cause:    A runtime error occurred during execution of a trigger.
*Action:   Check the triggers which were involved in the operation.
```

13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.

```
CREATE OR REPLACE PACKAGE pachet_proiect AS
   PROCEDURE ieftinire_jocuri_ex6;
   PROCEDURE date_magazin_ex7;
   FUNCTION vanzari_magazin_ex8
      (v_nume_mag magazin.nume_magazin%TYPE)
      RETURN NUMBER;
   PROCEDURE stoc_dezvoltator_ex9
      (v_nume_dez dezvoltator.nume_dezvoltator%TYPE,
      v_nume_tara tara.nume_tara%TYPE);
END pachet_proiect;
/

CREATE OR REPLACE PACKAGE BODY pachet_proiect AS
   PROCEDURE ieftinire_jocuri_ex6
   IS
      TYPE tablou_indexat IS TABLE OF joc%ROWTYPE
                  INDEX BY BINARY_INTEGER;
      TYPE tablou_imbricat IS TABLE OF versiune%ROWTYPE;
      TYPE vector IS VARRAY(3) OF platforma%ROWTYPE;

      t_jocuri tablou_indexat;
      t_platforme vector := vector();
      t_versiuni tablou_imbricat := tablou_imbricat();
      k INTEGER;
      v_nume_joc joc.titlu%TYPE;
      v_nume_platforma platforma.nume_platforma%TYPE;
      v_pret versiune.pret%TYPE;
      v_ver versiune%ROWTYPE;
      v_count NUMBER;

   BEGIN
      SELECT *
      BULK COLLECT INTO t_jocuri
      FROM joc
      WHERE id_joc IN (SELECT id_joc
                  FROM (SELECT id_joc
                     FROM comanda_contine
                     GROUP BY id_joc
                     ORDER BY SUM(numar))
                  WHERE ROWNUM <= 5);

      SELECT *
      BULK COLLECT INTO t_platforme
      FROM platforma
      WHERE id_platforma IN (SELECT id_platforma
                  FROM (SELECT id_platforma
                     FROM comanda_contine
```

```
                    GROUP BY id_platforma
                    ORDER BY SUM(numar))
                WHERE ROWNUM <= 3);

    FOR i in t_jocuri.FIRST..t_jocuri.LAST LOOP
      FOR j in t_platforme.FIRST..t_platforme.LAST LOOP
        SELECT COUNT(*)
        INTO v_count
        FROM versiune
        WHERE id_joc = t_jocuri(i).id_joc AND id_platforma = t_platforme(j).id_platforma;
        IF v_count = 1 THEN
          SELECT *
          INTO v_ver
          FROM versiune
          WHERE id_joc = t_jocuri(i).id_joc AND id_platforma = t_platforme(j).id_platforma;
          t_versiuni.EXTEND;
          t_versiuni(t_versiuni.LAST) := v_ver;
        END IF;
      END LOOP;
    END LOOP;

    k := t_versiuni.FIRST;
    WHILE k <= t_versiuni.LAST LOOP
      t_versiuni(k).pret := t_versiuni(k).pret * 4 / 5;
      k := t_versiuni.NEXT(k);
    END LOOP;

    k := t_versiuni.FIRST;
    WHILE k <= t_versiuni.LAST LOOP
      SELECT titlu
      INTO v_nume_joc
      FROM joc
      WHERE id_joc = t_versiuni(k).id_joc;

      SELECT nume_platforma
      INTO v_nume_platforma
      FROM platforma
      WHERE id_platforma = t_versiuni(k).id_platforma;

      SELECT pret
      INTO v_pret
      FROM versiune
      WHERE id_joc = t_versiuni(k).id_joc AND id_platforma = t_versiuni(k).id_platforma;

      DBMS_OUTPUT.PUT_LINE('Jocul ' || v_nume_joc || ' de pe platforma ' || v_nume_platforma
|| ' avea pretul ' || v_pret || ' si a scazut la ' || t_versiuni(k).pret || '.');

      UPDATE versiune
      SET pret = t_versiuni(k).pret
      WHERE id_joc = t_versiuni(k).id_joc AND id_platforma = t_versiuni(k).id_platforma;

      k := t_versiuni.NEXT(k);
```

```
    END LOOP;
END;

PROCEDURE date_magazin_ex7
IS
    TYPE refcursor IS REF CURSOR RETURN magazin%ROWTYPE;
    c_magazine refcursor;
    CURSOR c_angajati (id_mag magazin.id_magazin%TYPE) IS
        SELECT *
        FROM angajat
        WHERE id_magazin = id_mag;
    v_mag magazin%ROWTYPE;
    v_nr_ang NUMBER;
    v_avg_sal NUMBER;
BEGIN
    OPEN c_magazine FOR
        SELECT * FROM magazin;
    LOOP
        FETCH c_magazine INTO v_mag;
        EXIT WHEN c_magazine%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('-------------------------------------');
        DBMS_OUTPUT.PUT_LINE(v_mag.nume_magazin);
        DBMS_OUTPUT.PUT_LINE('Adresa: ' || v_mag.adresa);
        DBMS_OUTPUT.PUT_LINE('-------------------------------------');

        v_nr_ang := 0;
        v_avg_sal := 0;

        FOR v_ang IN c_angajati(v_mag.id_magazin) LOOP
            DBMS_OUTPUT.PUT_LINE(v_ang.nume || ' ' || v_ang.prenume || ' ' || v_ang.salariu);
            v_nr_ang := v_nr_ang + 1;
            v_avg_sal := v_avg_sal + v_ang.salariu;
        END LOOP;

        IF v_nr_ang = 0 THEN
            DBMS_OUTPUT.PUT_LINE('Magazinul nu are angajati');
        ELSE
            v_avg_sal := v_avg_sal / v_nr_ang;
            DBMS_OUTPUT.PUT_LINE('-------------------------------------');
            DBMS_OUTPUT.PUT_LINE('Numarul de angajati: ' || v_nr_ang);
            DBMS_OUTPUT.PUT_LINE('Salariul mediu: ' || v_avg_sal);
        END IF;
        DBMS_OUTPUT.PUT_LINE('-------------------------------------');
        DBMS_OUTPUT.PUT_LINE('');
    END LOOP;
    CLOSE c_magazine;
END;

FUNCTION vanzari_magazin_ex8
    (v_nume_mag magazin.nume_magazin%TYPE)
RETURN NUMBER IS
```

```
    v_id_mag magazin.id_magazin%TYPE;
    v_nr_ang NUMBER;
    v_nr_comenzi NUMBER;
    v_joc_vand NUMBER;
    NO_EMPLOYEES EXCEPTION;
    NO_SALES EXCEPTION;
  BEGIN
    SELECT id_magazin
    INTO v_id_mag
    FROM magazin
    WHERE UPPER(nume_magazin) LIKE ('%' || UPPER(v_nume_mag) || '%');

    SELECT COUNT(*)
    INTO v_nr_ang
    FROM angajat
    WHERE id_magazin = v_id_mag;
    IF v_nr_ang = 0 THEN
      RAISE NO_EMPLOYEES;
    END IF;

    SELECT COUNT(*)
    INTO v_nr_comenzi
    FROM comanda
    WHERE id_angajat IN (SELECT id_angajat FROM angajat
                  WHERE id_magazin = v_id_mag);
    IF v_nr_comenzi = 0 THEN
      RAISE NO_SALES;
    END IF;

    SELECT SUM(numar)
    INTO v_joc_vand
    FROM comanda_contine JOIN comanda USING (id_comanda)
      JOIN angajat USING (id_angajat)
    WHERE id_magazin = v_id_mag;

    RETURN v_joc_vand;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Nu exista magazin cu acel nume.');
      RETURN -1;
    WHEN TOO_MANY_ROWS THEN
      DBMS_OUTPUT.PUT_LINE('Exista mai multe magazine cu acel nume.');
      RETURN -1;
    WHEN NO_EMPLOYEES THEN
      DBMS_OUTPUT.PUT_LINE('Magazinul nu are angajati.');
      RETURN 0;
    WHEN NO_SALES THEN
      DBMS_OUTPUT.PUT_LINE('Magazinul nu are vanzari.');
      RETURN 0;
  END;

  PROCEDURE stoc_dezvoltator_ex9
```

```
    (v_nume_dez dezvoltator.nume_dezvoltator%TYPE,
    v_nume_tara tara.nume_tara%TYPE)
  IS
    TYPE tablou_orase IS TABLE OF oras%ROWTYPE;

    v_id_dez dezvoltator.id_dezvoltator%TYPE;
    v_id_tara tara.id_tara%TYPE;

    v_nr_orase NUMBER;
    v_stoc NUMBER;
    v_vanzari NUMBER;
    t_orase tablou_orase;
    k NUMBER;

    NO_CITIES EXCEPTION;
  BEGIN
    BEGIN
      SELECT id_dezvoltator
      INTO v_id_dez
      FROM dezvoltator
      WHERE UPPER(nume_dezvoltator) LIKE ('%' || UPPER(v_nume_dez) || '%');
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista dezvoltator cu acel nume.');
        RAISE;
      WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multi dezvoltatori cu acel nume.');
        RAISE;
    END;

    BEGIN
      SELECT id_tara
      INTO v_id_tara
      FROM tara
      WHERE UPPER(nume_tara) LIKE ('%' || UPPER(v_nume_tara) || '%');
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista tara cu acel nume.');
        RAISE;
      WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Exista mai multe tari cu acel nume.');
        RAISE;
    END;

    SELECT COUNT(*)
    INTO v_nr_orase
    FROM oras
    WHERE id_tara = v_id_tara;
    IF v_nr_orase = 0 THEN
      RAISE NO_CITIES;
    END IF;
```

```
      SELECT *
      BULK COLLECT INTO t_orase
      FROM oras
      WHERE id_tara = v_id_tara;

      k := t_orase.FIRST;
      WHILE k <= t_orase.LAST LOOP
        SELECT SUM(nr_produse)
        INTO v_stoc
        FROM contribuie JOIN stoc USING (id_joc)
          JOIN magazin USING (id_magazin)
        WHERE id_dezvoltator = v_id_dez
          AND id_oras = t_orase(k).id_oras;

        SELECT SUM(numar)
        INTO v_vanzari
        FROM contribuie JOIN comanda_contine USING (id_joc)
          JOIN comanda USING (id_comanda)
          JOIN angajat USING (id_angajat)
          JOIN magazin USING (id_magazin)
        WHERE id_dezvoltator = v_id_dez
          AND id_oras = t_orase(k).id_oras;

        IF v_stoc IS NULL THEN
          v_stoc := 0;
        END IF;
        IF v_vanzari IS NULL THEN
          v_vanzari := 0;
        END IF;

        DBMS_OUTPUT.PUT_LINE('In orasul ' || t_orase(k).nume_oras || ' dezvolatorul are stocul ' ||
v_stoc || ' si suma totala de vanzari ' || v_vanzari || '.');
        k := t_orase.NEXT(k);
      END LOOP;
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Eroare NO_DATA_FOUND');
      WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Eroare TOO_MANY_ROWS');
      WHEN NO_CITIES THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista orase in acea tara.');
    END;
  END pachet_proiect;
  /
```

```
Package PACHET_PROIECT compiled
```

```
Package Body PACHET_PROIECT compiled
```

14. Definiți un pachet care să includă tipuri de date complexe și obiecte necesare unui flux de acțiuni integrate, specifice bazei de date definite (minim 2 tipuri de date, minim 2 funcții, minim 2 proceduri).

Un flux de acțiuni pentru facerea comenzilor:

- Suplimentarea stocului
- Creearea unei comenzi
- Adăugarea de produse la comandă
- Verificarea stocului
- Returnarea prețului comenzii
- Facerea comenzii

```
CREATE OR REPLACE PACKAGE flux_comanda AS
   PROCEDURE suplimentare_stoc
      (v_mag magazin.id_magazin%TYPE, v_joc joc.id_joc%TYPE, v_plat
platforma.id_platforma%TYPE, cantitate stoc.nr_produse%TYPE);

   PROCEDURE creare_comanda
      (v_client client.id_client%TYPE, v_plata comanda.metoda_plata%TYPE, v_angajat
angajat.id_angajat%TYPE, v_online BOOLEAN, v_adresa comanda_online.adresa%TYPE);

   PROCEDURE adaugare_item
      (v_joc joc.id_joc%TYPE, v_plat platforma.id_platforma%TYPE, cantitate
comanda_contine.numar%TYPE);

   FUNCTION verificare_stoc
      (v_mag magazin.id_magazin%TYPE, v_joc joc.id_joc%TYPE, v_plat
platforma.id_platforma%TYPE, cantitate stoc.nr_produse%TYPE)
      RETURN BOOLEAN;

   FUNCTION pret_comanda
      RETURN NUMBER;

   PROCEDURE facere_comanda;
END flux_comanda;
/

CREATE OR REPLACE PACKAGE BODY flux_comanda AS
   TYPE item_comanda IS RECORD
   (
      id_joc joc.id_joc%TYPE,
      id_platforma platforma.id_platforma%TYPE,
      cantitate comanda_contine.numar%TYPE
   );

   TYPE tablou_iteme IS TABLE OF item_comanda;

   TYPE comanda_noua IS RECORD
   (
```

```
    id_client client.id_client%TYPE,
    metoda_plata comanda.metoda_plata%TYPE,
    id_angajat angajat.id_angajat%TYPE,
    is_online BOOLEAN,
    adresa comanda_online.adresa%TYPE,
    lista_iteme tablou_iteme := tablou_iteme()
  );

  comanda_curenta comanda_noua;

  INVALID_CLIENT EXCEPTION;
  INVALID_EMPLOYEE EXCEPTION;
  INVALID_GAME EXCEPTION;
  INVALID_STOC EXCEPTION;
  INVALID_SHOP EXCEPTION;

  PROCEDURE suplimentare_stoc
    (v_mag magazin.id_magazin%TYPE, v_joc joc.id_joc%TYPE, v_plat
platforma.id_platforma%TYPE, cantitate stoc.nr_produse%TYPE)
  IS
    v_in_stoc NUMBER;
    v_valid_joc NUMBER;
    v_valid_mag NUMBER;
  BEGIN
    SELECT COUNT(*)
    INTO v_valid_joc
    FROM versiune
    WHERE id_joc = v_joc AND id_platforma = v_plat;

    IF v_valid_joc = 0 THEN
      RAISE INVALID_GAME;
    END IF;

    SELECT COUNT(*)
    INTO v_valid_mag
    FROM magazin
    WHERE id_magazin = v_mag;

    IF v_valid_mag = 0 THEN
      RAISE INVALID_SHOP;
    END IF;

    SELECT COUNT(*)
    INTO v_in_stoc
    FROM stoc
    WHERE id_magazin = v_mag AND id_joc = v_joc AND id_platforma = v_plat;

    IF v_in_stoc = 0 THEN
      INSERT INTO stoc
      VALUES(v_mag, v_joc, v_plat, cantitate);
    ELSE
      UPDATE stoc
```

```
        SET nr_produse = nr_produse + cantitate
        WHERE id_magazin = v_mag AND id_joc = v_joc AND id_platforma = v_plat;
    END IF;
  EXCEPTION
    WHEN INVALID_GAME THEN
      DBMS_OUTPUT.PUT_LINE('Joc invalid');
    WHEN INVALID_SHOP THEN
      DBMS_OUTPUT.PUT_LINE('Magazin invalid');
  END;

  PROCEDURE creare_comanda
    (v_client client.id_client%TYPE, v_plata comanda.metoda_plata%TYPE, v_angajat
angajat.id_angajat%TYPE, v_online BOOLEAN, v_adresa comanda_online.adresa%TYPE)
  IS
    v_valid_client NUMBER;
    v_valid_angajat NUMBER;
  BEGIN
    SELECT COUNT(*)
    INTO v_valid_client
    FROM client
    WHERE id_client = v_client;

    SELECT COUNT(*)
    INTO v_valid_angajat
    FROM angajat
    WHERE id_angajat = v_angajat;

    IF v_valid_client = 0 THEN
      RAISE INVALID_CLIENT;
    ELSIF v_valid_angajat = 0 THEN
      RAISE INVALID_EMPLOYEE;
    END IF;

    comanda_curenta.id_client := v_client;
    comanda_curenta.metoda_plata := v_plata;
    comanda_curenta.id_angajat := v_angajat;
    comanda_curenta.is_online := v_online;
    comanda_curenta.adresa := v_adresa;
  EXCEPTION
    WHEN INVALID_CLIENT THEN
      DBMS_OUTPUT.PUT_LINE('Clientul nu exista');
    WHEN INVALID_EMPLOYEE THEN
      DBMS_OUTPUT.PUT_LINE('Angajatul nu exista');
  END;

  PROCEDURE adaugare_item
    (v_joc joc.id_joc%TYPE, v_plat platforma.id_platforma%TYPE, cantitate
comanda_contine.numar%TYPE)
  IS
    v_in_lista BOOLEAN := FALSE;
    v_valid_joc NUMBER;
    v_nou_item item_comanda;
```

```
    k INTEGER;
  BEGIN
    SELECT COUNT(*)
    INTO v_valid_joc
    FROM versiune
    WHERE id_joc = v_joc AND id_platforma = v_plat;

    IF v_valid_joc = 0 THEN
      RAISE INVALID_GAME;
    END IF;

    k := comanda_curenta.lista_iteme.FIRST;
    WHILE k <= comanda_curenta.lista_iteme.LAST LOOP
      IF comanda_curenta.lista_iteme(k).id_joc = v_joc AND
comanda_curenta.lista_iteme(k).id_platforma = v_plat THEN
        v_in_lista := TRUE;
        comanda_curenta.lista_iteme(k).cantitate := comanda_curenta.lista_iteme(k).cantitate +
cantitate;
      END IF;
      k := comanda_curenta.lista_iteme.NEXT(k);
    END LOOP;
    IF v_in_lista = FALSE THEN
      v_nou_item.id_joc := v_joc;
      v_nou_item.id_platforma := v_plat;
      v_nou_item.cantitate := cantitate;
      comanda_curenta.lista_iteme.EXTEND;
      comanda_curenta.lista_iteme(comanda_curenta.lista_iteme.LAST) := v_nou_item;
    END IF;
  EXCEPTION
    WHEN INVALID_GAME THEN
      DBMS_OUTPUT.PUT_LINE('Joc invalid');
  END;

  FUNCTION verificare_stoc
    (v_mag magazin.id_magazin%TYPE, v_joc joc.id_joc%TYPE, v_plat
platforma.id_platforma%TYPE, cantitate stoc.nr_produse%TYPE)
    RETURN BOOLEAN
  IS
    v_in_tabel NUMBER;
    v_nr_stoc stoc.nr_produse%TYPE;
    v_in_stoc BOOLEAN;
  BEGIN
    SELECT COUNT(*)
    INTO v_in_tabel
    FROM stoc
    WHERE id_magazin = v_mag AND id_joc = v_joc AND id_platforma = v_plat;

    IF v_in_tabel = 0 THEN
      v_in_stoc := FALSE;
    ELSE
      SELECT nr_produse
      INTO v_nr_stoc
```

```
        FROM stoc
        WHERE id_magazin = v_mag AND id_joc = v_joc AND id_platforma = v_plat;

        IF v_nr_stoc < cantitate THEN
          v_in_stoc := FALSE;
        ELSE
          v_in_stoc := TRUE;
        END IF;
      END IF;

      RETURN v_in_stoc;
    END;

    FUNCTION pret_comanda
      RETURN NUMBER
    IS
      v_total NUMBER;
      v_pret versiune.pret%TYPE;
      k INTEGER;
    BEGIN
      v_total := 0;
      k := comanda_curenta.lista_iteme.FIRST;
      WHILE k <= comanda_curenta.lista_iteme.LAST LOOP
        SELECT pret
        INTO v_pret
        FROM versiune
        WHERE id_joc = comanda_curenta.lista_iteme(k).id_joc AND id_platforma =
comanda_curenta.lista_iteme(k).id_platforma;

        v_total := v_total + v_pret * comanda_curenta.lista_iteme(k).cantitate;
        k := comanda_curenta.lista_iteme.NEXT(k);
      END LOOP;

      RETURN v_total;
    EXCEPTION
      WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Joc invalid in lista comenzii');
    END;

    PROCEDURE facere_comanda
    IS
      k INTEGER;
      v_valid_stoc BOOLEAN;
      v_mag magazin.id_magazin%TYPE;
      v_id_comanda comanda.id_comanda%TYPE;
    BEGIN
      SELECT id_magazin
      INTO v_mag
      FROM angajat
      WHERE id_angajat = comanda_curenta.id_angajat;

      k := comanda_curenta.lista_iteme.FIRST;
```

```
    WHILE k <= comanda_curenta.lista_iteme.LAST LOOP
      v_valid_stoc := verificare_stoc(v_mag, comanda_curenta.lista_iteme(k).id_joc,
comanda_curenta.lista_iteme(k).id_platforma, comanda_curenta.lista_iteme(k).cantitate);
      IF v_valid_stoc = FALSE THEN
        RAISE INVALID_STOC;
      END IF;

      k := comanda_curenta.lista_iteme.NEXT(k);
    END LOOP;

    v_id_comanda := secventa_comanda.NEXTVAL;

    INSERT INTO comanda
    VALUES (v_id_comanda, comanda_curenta.id_client, comanda_curenta.id_angajat, SYSDATE,
comanda_curenta.metoda_plata);

    IF comanda_curenta.is_online = TRUE THEN
      INSERT INTO comanda_online
      VALUES (v_id_comanda, comanda_curenta.adresa, SYSDATE + 2);
    END IF;

    k := comanda_curenta.lista_iteme.FIRST;
    WHILE k <= comanda_curenta.lista_iteme.LAST LOOP
      UPDATE stoc
      SET nr_produse = nr_produse - comanda_curenta.lista_iteme(k).cantitate
      WHERE id_joc = comanda_curenta.lista_iteme(k).id_joc AND id_platforma =
comanda_curenta.lista_iteme(k).id_platforma AND id_magazin = v_mag;

      INSERT INTO comanda_contine
      VALUES (comanda_curenta.lista_iteme(k).id_joc,
comanda_curenta.lista_iteme(k).id_platforma, v_id_comanda,
comanda_curenta.lista_iteme(k).cantitate);

      k := comanda_curenta.lista_iteme.NEXT(k);
    END LOOP;

    comanda_curenta.lista_iteme.DELETE();

    comanda_curenta.id_client := NULL;
    comanda_curenta.metoda_plata := NULL;
    comanda_curenta.id_angajat := NULL;
    comanda_curenta.is_online := NULL;
    comanda_curenta.adresa := NULL;
  EXCEPTION
    WHEN INVALID_STOC THEN
      comanda_curenta.lista_iteme.DELETE();
      comanda_curenta.id_client := NULL;
      comanda_curenta.metoda_plata := NULL;
      comanda_curenta.id_angajat := NULL;
      comanda_curenta.is_online := NULL;
      comanda_curenta.adresa := NULL;
      DBMS_OUTPUT.PUT_LINE('Nu exista stoc sufficient');
```

```
      WHEN OTHERS THEN
         comanda_curenta.lista_iteme.DELETE();
         comanda_curenta.id_client := NULL;
         comanda_curenta.metoda_plata := NULL;
         comanda_curenta.id_angajat := NULL;
         comanda_curenta.is_online := NULL;
         comanda_curenta.adresa := NULL;
         DBMS_OUTPUT.PUT_LINE('Data invalide');
   END;
END flux_comanda;
/
```

Dacă se rulează următoarea secvență de cod rezultatul este:

```
EXECUTE flux_comanda.creare_comanda(1000007, 'CASH', 1000, TRUE, NULL);
EXECUTE flux_comanda.creare_comanda(1000000, 'CASH', 999, TRUE, NULL);
EXECUTE flux_comanda.creare_comanda(1000000, 'CASH', 1000, TRUE, NULL);

EXECUTE flux_comanda.adaugare_item(100290, 2, 1);
EXECUTE flux_comanda.adaugare_item(1, 2, 1);

BEGIN
   DBMS_OUTPUT.PUT_LINE(flux_comanda.pret_comanda());
END;
/

EXECUTE flux_comanda.suplimentare_stoc(300, 1, 2, 1);
EXECUTE flux_comanda.suplimentare_stoc(299, 100290, 2, 1);
EXECUTE flux_comanda.suplimentare_stoc(300, 100290, 2, 1);

EXECUTE flux_comanda.facere_comanda();
EXECUTE flux_comanda.facere_comanda();

EXECUTE flux_comanda.creare_comanda(1000000, 'CASH', 1000, TRUE, NULL);
EXECUTE flux_comanda.adaugare_item(40741, 4, 5);
BEGIN
   DBMS_OUTPUT.PUT_LINE(flux_comanda.pret_comanda());
END;
/
EXECUTE flux_comanda.facere_comanda();

ROLLBACK;
```

```
Clientul nu exista

PL/SQL procedure successfully completed.

Angajatul nu exista

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Joc invalid

PL/SQL procedure successfully completed.

40

PL/SQL procedure successfully completed.

Joc invalid

PL/SQL procedure successfully completed.

Magazin invalid
```

```
PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Data invalide

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

300

PL/SQL procedure successfully completed.

Nu exista stoc sufficient

PL/SQL procedure successfully completed.

Rollback complete.
```