***PROIECT SGBD***

**PLATFORMĂ DE GAMING**

***Jiglău Fabrizzio***

***SERIA 23, GRUPA 233***

***2024 – 2025***

**CUPRINS**

[DESCRIERE MODEL 3](#_Toc187444880)

[DIAGRAMA ENTITATE-RELAȚIE 5](#_Toc187444881)

[DIAGRAMA CONCEPTUALĂ 6](#_Toc187444882)

[SCHEMELE RELAȚIONALE 7](#_Toc187444883)

[CREAREA TABELELOR ȘI ÎNREGISTRAREA DATELOR 8](#_Toc187444884)

[UTILIZATOR 8](#_Toc187444885)

[NOTIFICARE 9](#_Toc187444886)

[MESAJ 12](#_Toc187444887)

[TARA 15](#_Toc187444888)

[LOCATIE 18](#_Toc187444889)

[DEVELOPER 21](#_Toc187444890)

[JOC 23](#_Toc187444891)

[RECENZIE 26](#_Toc187444892)

[COLECTIE 29](#_Toc187444893)

[COLECTIE\_JOC 32](#_Toc187444894)

[INVENTAR 34](#_Toc187444895)

[ITEM 35](#_Toc187444896)

[INVENTAR\_ITEM 39](#_Toc187444897)

[TRANZACTIE 41](#_Toc187444898)

[6. Subprogram stocat independent care să utilizeze toate cele 3 tipuri de colecții studiate 44](#_Toc187444899)

[7. Subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat, dependent de celălalt cursor. 47](#_Toc187444900)

[8. Subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele create. 50](#_Toc187444901)

[9. Subprogram stocat independent de tip procedură care să aibă minim 2 parametri și să utilizeze într-o singură comandă SQL 5 dintre tabelele create 55](#_Toc187444902)

[10. Definiți un trigger de tip LMD la nivel de comandă 63](#_Toc187444903)

[11. Definiți un trigger de tip LMD la nivel de linie. 68](#_Toc187444904)

[12. Definiți un trigger de tip LDD. 74](#_Toc187444905)

[13. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un pachet 77](#_Toc187444906)

## DESCRIERE MODEL

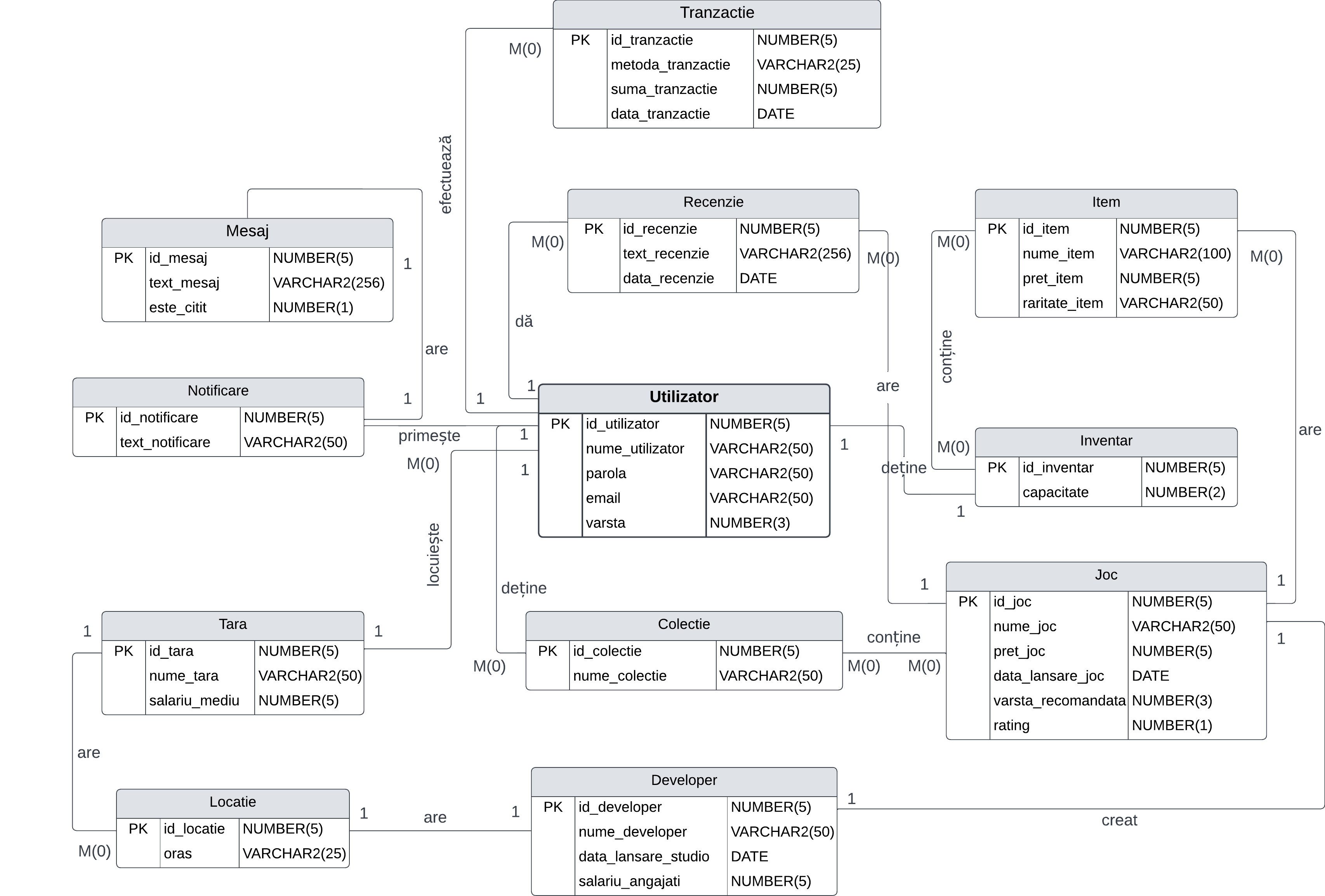
Această bază de date este proiectată pentru a gestiona o platformă de gaming complexă, care include informații despre utilizatori, jocuri, dezvoltatori de jocuri, tranzacții, recenzii, notificări, mesaje, țări, locații, colecții, inventare și iteme. Scopul principal al acestei baze de date este de a organiza și stoca datele necesare pentru funcționarea eficientă și integrată a platformei, asigurând în același timp integritatea și consistența datelor.

Modelul este inspirat din celebrul Steam, un gigant al industriei de jocuri video. Baza de date permite utilizatorului să trăiască experiența adusă de original, adăugând și utilități extra. User-ul are acces atât la o gamă largă de jocuri, cât și la un marketplace centrat în jurul itemelor ce aparțin de jocurile platformei.

Entitățile principale sunt: **Utilizator, Recenzie, Tranzactie, Notificare, Mesaj, Tara, Locatie, Developer, Colectie, Joc, Colectie\_Joc, Inventar, Item și Inventar\_Item**.

* UTILIZATOR – Reprezintă persoanele care își crează un cont pe platformă. Aceștia pot interacționa cu diverse funcționalități ale platformei, cum ar fi jocurile, recenziile și tranzacțiile. Cheia primară este id\_utilizator.
* RECENZIE – Recenziile sunt evaluările lăsate de utilizatori pentru diferite jocuri disponibile pe platformă. Fiecare recenzie este legată de un utilizator și un joc. Cheia primară este id\_recenzie.
* TRANZACTIE – Tranzacțiile reprezintă plățile efectuate de utilizatori. Fiecare tranzacție este asociată unui utilizator. Cheia primară este id\_tranzactie.
* NOTIFICARE – Notificările sunt texte scurte trimise utilizatorilor pentru a-i informa despre diverse evenimente sau promoții. Fiecare notificare este legată de un utilizator. Cheia primară este id\_notificare.
* MESAJ – Mesajele sunt conținutul individual al fiecărei notificări, fiecare mesaj fiind asociat unei notificări. Cheia primară este id\_mesaj.
* TARA – Țările sunt utilizate pentru a specifica locația dezvoltatorilor de jocuri și a altor entități. Cheia primară este id\_tara.
* LOCATIE – Locațiile sunt specificate pentru fiecare dezvoltator și sunt legate de o țară. Cheia primară este id\_locatie.
* DEVELOPER – Developerii sunt entitățile care creează jocurile. Fiecare dezvoltator este asociat unei locații. Cheia primară este id\_developer.
* COLECTIE – Colecțiile sunt seturi de jocuri organizate de utilizatori. Fiecare colecție este legată de un utilizator. Cheia primară este id\_colectie.
* JOC – Jocurile sunt produsele principale ale platformei, fiecare joc fiind creat de un dezvoltator și putând fi inclus în colecții, dar si evaluat de utilizatori prin recenzii. Cheia primară este id\_joc.
* COLECTIE\_JOC – Această entitate gestionează relația de tip many-to-many între jocuri și colecții. Cheile primare sunt id\_colectie și id\_joc.
* INVENTAR – Inventarele sunt seturi de iteme deținute de utilizatori. Fiecare inventar este asociat unui utilizator. Cheia primară este id\_inventar.
* ITEM – Itemele sunt obiecte virtuale care pot fi achiziționate sau câștigate în jocuri. Fiecare item este asociat unui joc. Cheia primară este id\_item.
* INVENTAR\_ITEM – Această entitate gestionează relația de tip many-to-many între iteme și inventare. Cheile primare sunt id\_item și id\_inventar.
* Un utilizator poate avea mai multe recenzii, iar o recenzie este asociată unui singur utilizator și unui singur joc.
* Un utilizator poate efectua mai multe tranzacții, iar o tranzacție este asociată unui singur utilizator.
* Un utilizator poate primi mai multe notificări, iar o notificare este asociată unui singur utilizator.
* Un utilizator poate avea mai multe colecții, iar o colecție este asociată unui singur utilizator.
* Un utilizator poate avea un singur inventar, iar un inventar este asociat unui singur utilizator.
* O notificare are un singur mesaj asociat, iar un mesaj aparține unei singure notificări.
* O colecție poate conține mai multe jocuri, iar un joc poate aparține mai multor colecții.
* Un inventar poate conține mai multe iteme, iar un item poate aparține mai multor inventare.
* Un dezvoltator poate crea mai multe jocuri, iar un joc este creat de un singur dezvoltator.
* O locație poate fi asociată mai multor dezvoltatori, iar un dezvoltator este localizat într-o singură locație.
* O țară poate avea mai multe locații, iar o locație este asociată unei singure țări.

## DIAGRAMA ENTITATE-RELAȚIE



## D:\FMI\Y1S2\BD\Proiect\CONC_DIAGRAM.pngDIAGRAMA CONCEPTUALĂ

## SCHEMELE RELAȚIONALE

**Utilizator** (**id\_utilizator#, id\_tara#**, *nume\_utilizator, parola, email, varsta*)

**Recenzie** (**id\_recenzie#, id\_utilizator#, id\_joc#,** *text\_recenzie, data\_recenzie*)

**Tranzactie** (**id\_tranzactie#, id\_utilizator#**, *metoda\_tranzactie, suma\_tranzactie, data\_tranzactie*)

**Notificare** (**id\_notificare#, id\_utilizator#,** *text\_notificare*)

**Mesaj** (**id\_mesaj#, id\_notificare#,** *text\_mesaj, este\_citit*)

**Tara** (**id\_tara#,** *nume\_tara, salariu\_mediu*)

**Locatie** (**id\_locatie#, id\_tara#**, *oras*)

**Developer** (**id\_developer#, id\_locatie*#,*** *nume\_developer, data\_lansare\_studio, salariu\_angajati*)

**Colectie** (**id\_colectie#, id\_utilizator#,** *nume\_colectie*)

**Joc** (**id\_joc#, id\_developer#**, *nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating*)

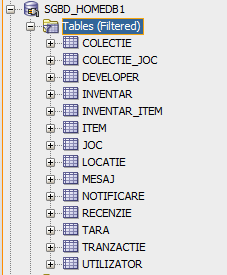
**Colectie\_Joc** (**id\_colectie#, id\_joc#**)

**Inventar** (**id\_inventar#, id\_utilizator#,** *capacitate*)

**Item** (**id\_item#, id\_joc#,** *nume\_item, pret\_item, raritate\_item*)

**Inventar\_Item** (**id\_item#, id\_inventar#**)

## CREAREA TABELELOR ȘI ÎNREGISTRAREA DATELOR



## UTILIZATOR

* CREATE

CREATE TABLE Utilizator (

id\_utilizator NUMBER(5) PRIMARY KEY,

id\_tara NUMBER(5) NOT NULL,

nume\_utilizator VARCHAR2(50) NOT NULL,

parola VARCHAR2(50) NOT NULL,

email VARCHAR2(50) NOT NULL UNIQUE,

varsta NUMBER(3) NOT NULL CHECK (varsta >= 0),

CONSTRAINT fk\_utilizator\_tara FOREIGN KEY (id\_tara) REFERENCES Tara(id\_tara)

);

DESCRIBE Utilizator;

* INSERT

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (1, 'QuantumLeap', 'timeTravel42', 'alex.ramirez@protonmail.com', 18, 1); -- SUA

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (2, 'GlitchWizard', 'codeMancer88', 'nina.schwartz@zoho.com', 19, 7); -- Suedia

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (3, 'NeonRider', 'speedOfLight77', 'leon.tang@yandex.com', 21, 2); -- Japonia

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (4, 'PixelHunter', 'sharpShooter89', 'marco.valenti@gmail.com', 12, 1); -- SUA

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (5, 'FrostMage', 'chillFactor0', 'sarah.muller@web.de', 23, 10); -- Germania

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (6, 'StormCaster', 'stormyWeather22', 'emily.dubois@yahoo.fr', 18, 4); -- Franta

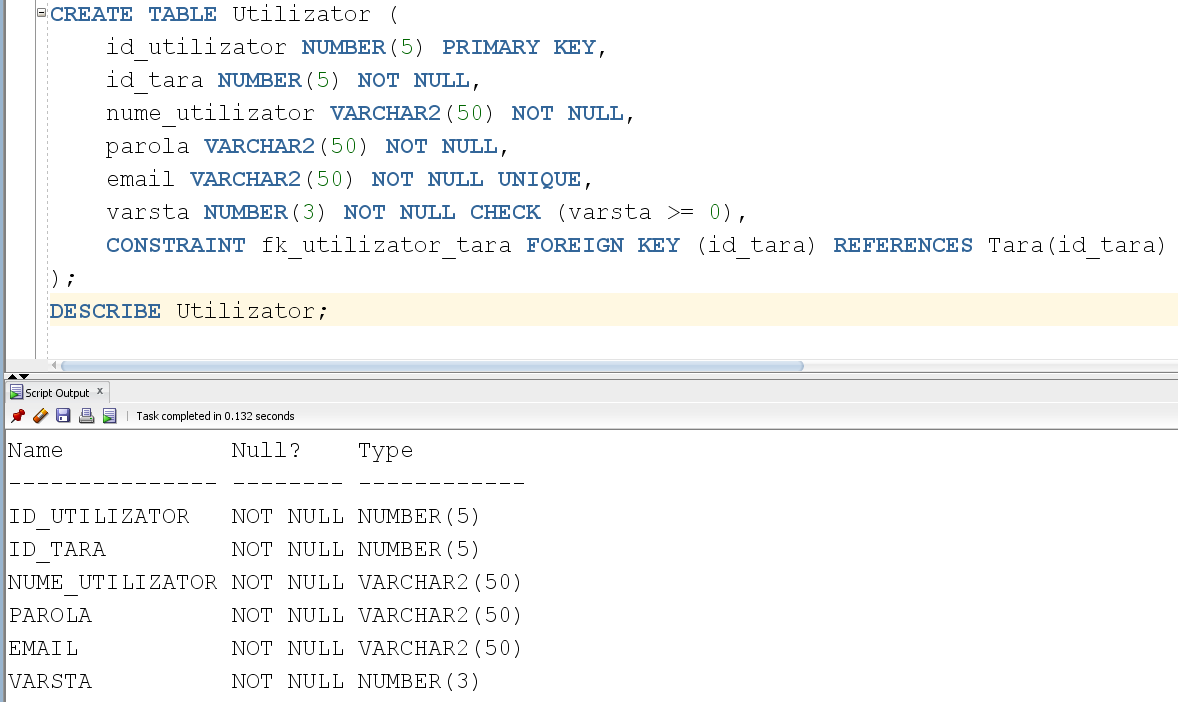
INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

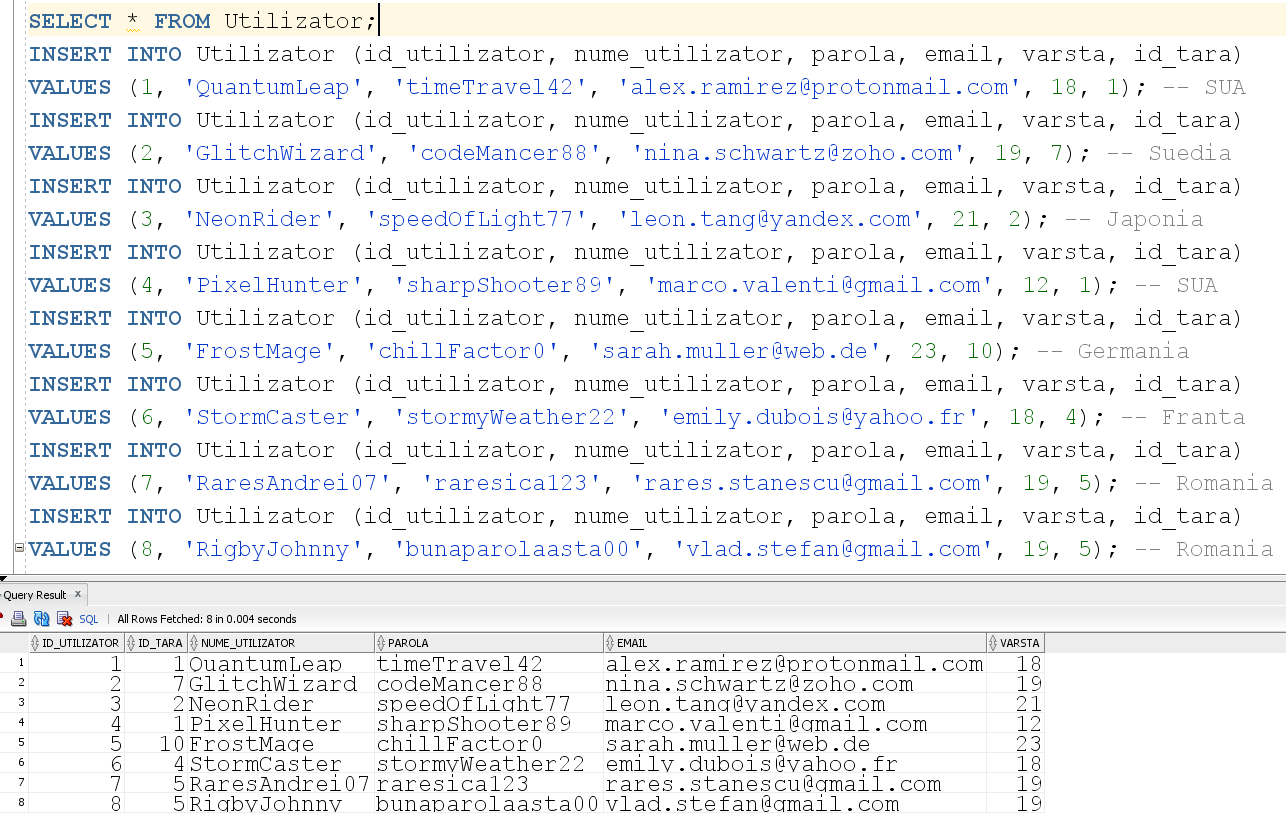
VALUES (7, 'RaresAndrei07', 'raresica123', 'rares.stanescu@gmail.com', 19, 5); -- Romania

INSERT INTO Utilizator (id\_utilizator, nume\_utilizator, parola, email, varsta, id\_tara)

VALUES (8, 'RigbyJohnny', 'bunaparolaasta00', 'vlad.stefan@gmail.com', 19, 5); -- Romania

SELECT \* FROM Utilizator;



****

## NOTIFICARE

* CREATE

CREATE TABLE Notificare (

id\_notificare NUMBER(5) PRIMARY KEY,

id\_utilizator NUMBER(5) NOT NULL,

text\_notificare VARCHAR2(50) NOT NULL,

CONSTRAINT fk\_notificare\_utilizator FOREIGN KEY (id\_utilizator) REFERENCES Utilizator(id\_utilizator)

);

DESCRIBE Notificare;

* INSERT

SELECT \* FROM Notificare;

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (1, 1, 'Noutati in cont. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (2, 1, 'Actualizare importanta. PORTOCALIU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (3, 2, 'Alerta de securitate. ROSU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (4, 2, 'Promotii noi. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (5, 3, 'Feedback procesat. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (6, 4, 'Termeni actualizati. PORTOCALIU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (7, 5, 'Cont expira curand. PORTOCALIU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (8, 6, 'Eligibil pentru upgrade. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (9, 7, 'Oferta speciala expira. PORTOCALIU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (10, 8, 'Suspiciune de frauda. ROSU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (11, 3, 'Joc preferat actualizat. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (12, 3, 'Reduceri de weekend. PORTOCALIU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (13, 4, 'Alerta securitate. ROSU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (14, 4, 'Invitatie beta test. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (15, 5, 'Abonament expira curand. PORTOCALIU');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (16, 6, 'Cadou in inventar. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (17, 7, 'Sondaj comunitate. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

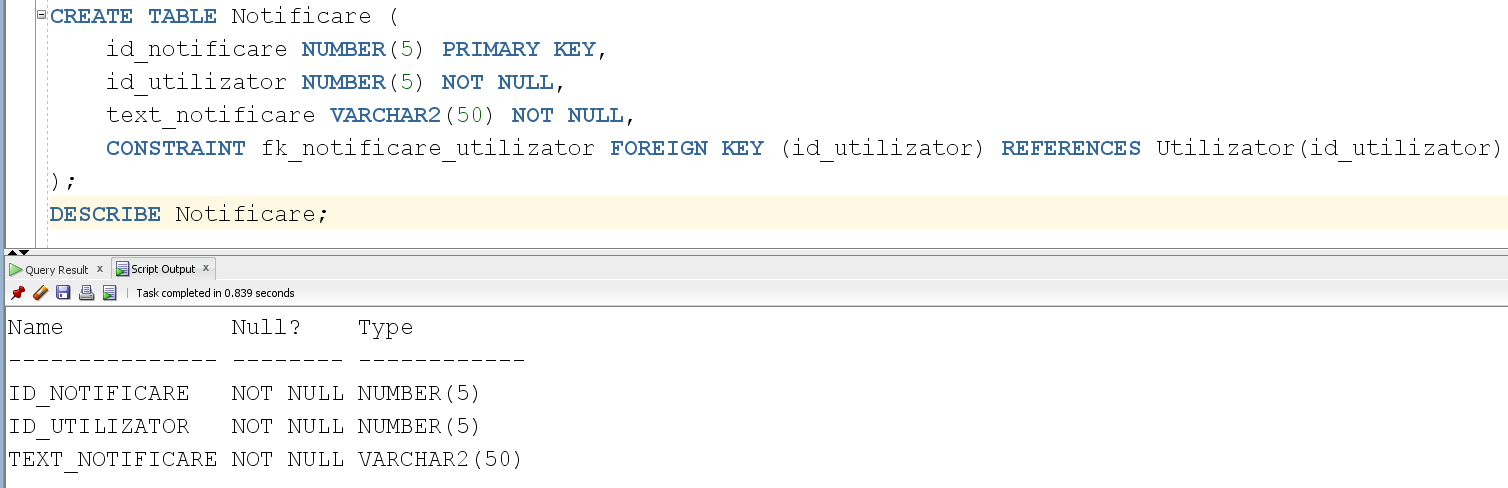
VALUES (18, 7, 'Update joc efectuat. VERDE');

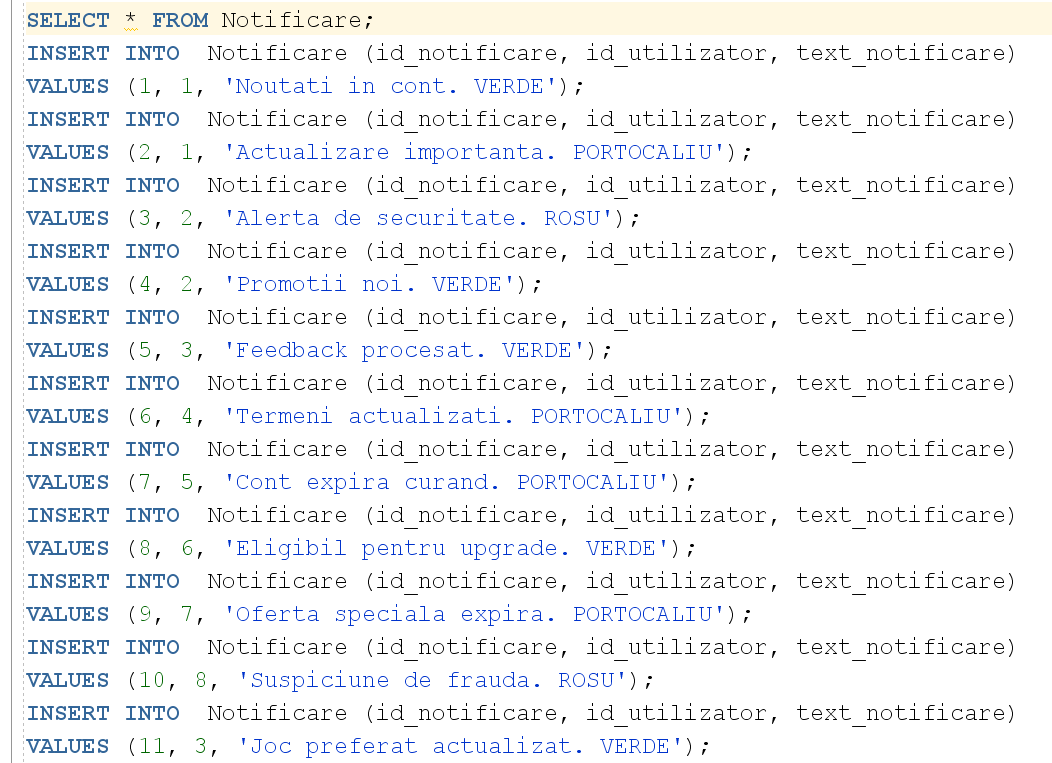
INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

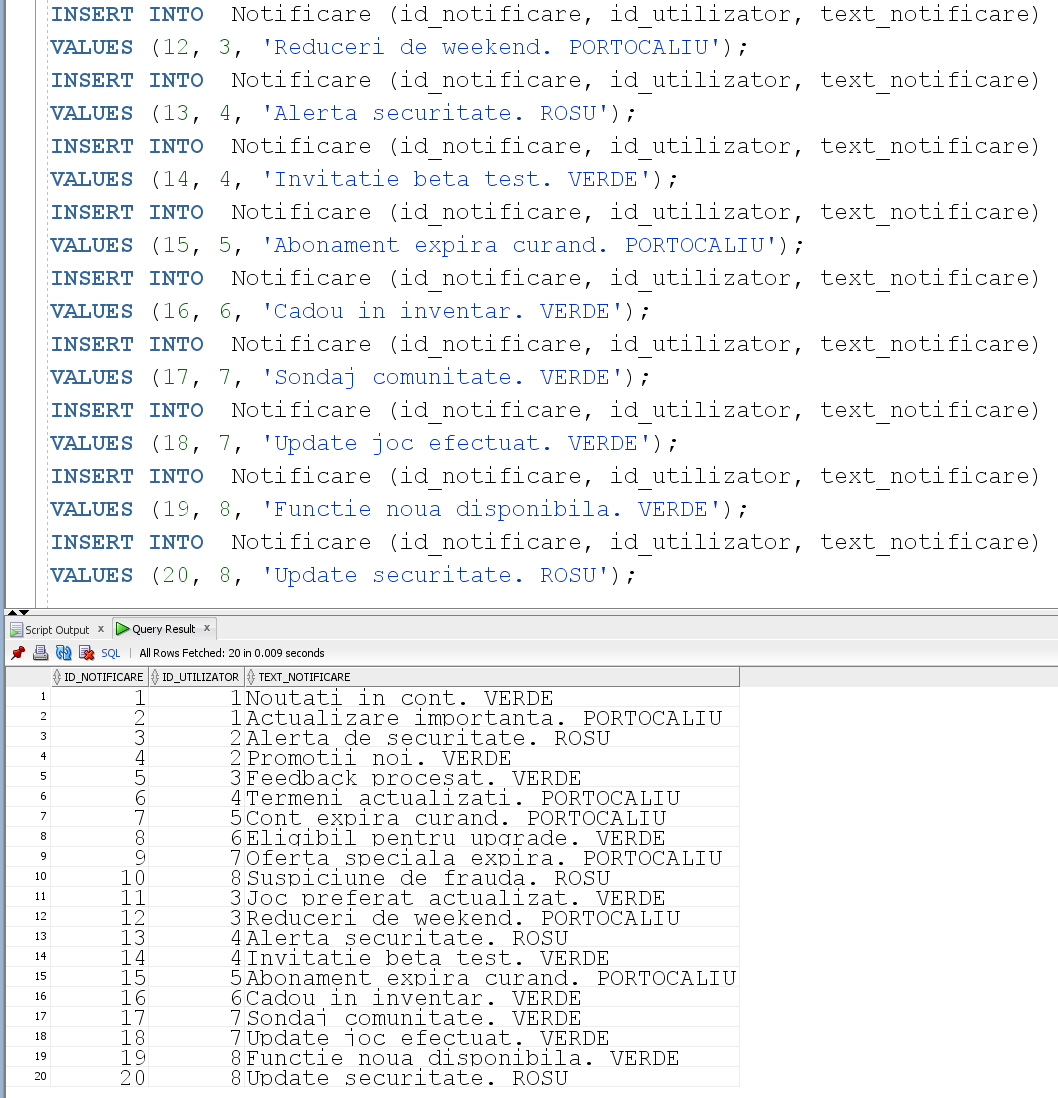
VALUES (19, 8, 'Functie noua disponibila. VERDE');

INSERT INTO Notificare (id\_notificare, id\_utilizator, text\_notificare)

VALUES (20, 8, 'Update securitate. ROSU');







## MESAJ

* CREATE

CREATE TABLE Mesaj (

id\_mesaj NUMBER(5) PRIMARY KEY,

id\_notificare NUMBER(5) NOT NULL,

text\_mesaj VARCHAR2(256) NOT NULL,

este\_citit NUMBER(1) NOT NULL CHECK (este\_citit IN (0, 1)),

CONSTRAINT fk\_mesaj\_notificare FOREIGN KEY (id\_notificare) REFERENCES Notificare(id\_notificare)

);

DESCRIBE Mesaj;

* INSERT

SELECT \* FROM Mesaj;

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (1, 1, 'Ai noutati in contul tau. Verifica-le acum!', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (2, 2, 'O actualizare importanta este disponibila pentru tine.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (3, 3, 'Alerta de securitate! Verifica activitatea recenta.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (4, 4, 'Profita de promotiile noi adaugate recent.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (5, 5, 'Feedback-ul tau a fost procesat cu succes.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (6, 6, 'Termenii si conditiile au fost actualizati.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (7, 7, 'Atentie, contul tau va expira in curand.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (8, 8, 'Esti eligibil pentru un upgrade. Detalii in cont.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (9, 9, 'Ultima sansa sa profiti de oferta speciala.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (10, 10, 'Suspiciune de frauda detectata. Actioneaza imediat!', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (11, 11, 'Jocul tau preferat a fost actualizat recent.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (12, 12, 'Reduceri de weekend disponibile acum.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (13, 13, 'Alerta de securitate majora! Verifica imediat.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (14, 14, 'Ai fost invitat la un beta test. Inscrie-te acum.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (15, 15, 'Abonamentul tau este pe cale sa expire. Reinnoieste-l.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (16, 16, 'Verifica inventarul tau pentru un cadou surpriza.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (17, 17, 'Participa la sondajul nostru si castiga puncte.', 0);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

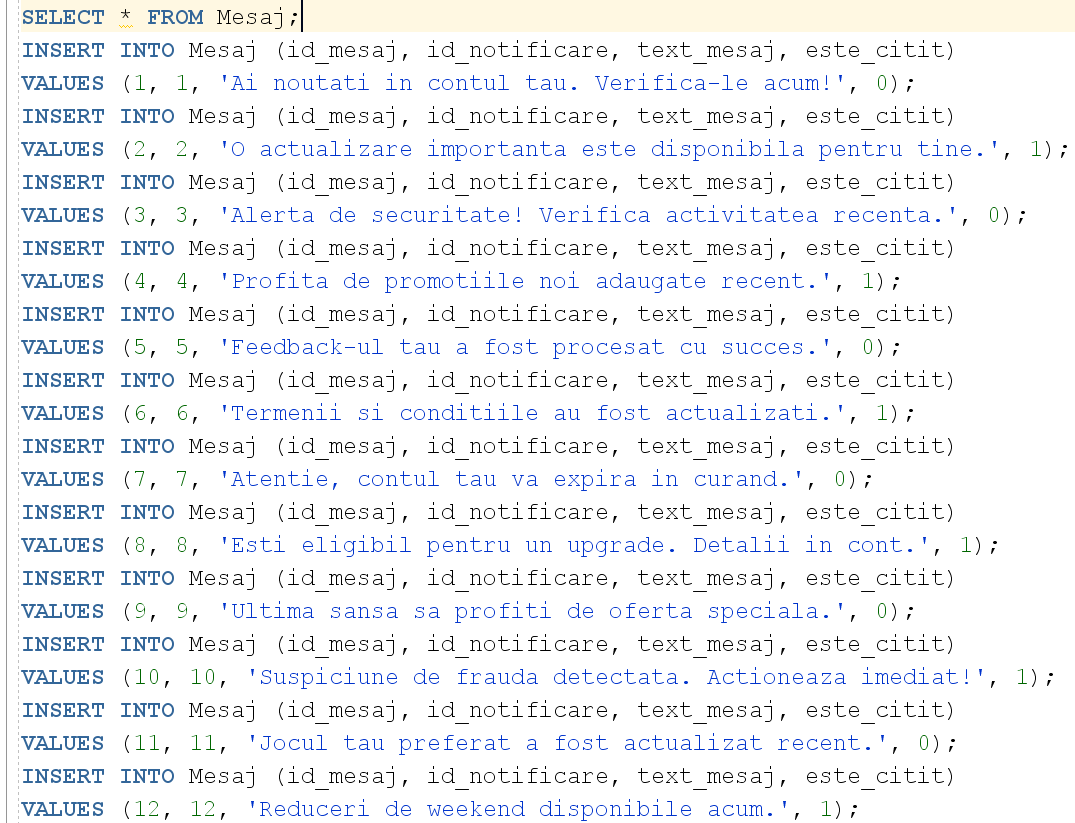
VALUES (18, 18, 'Update-ul jocului tau a fost efectuat cu succes.', 1);

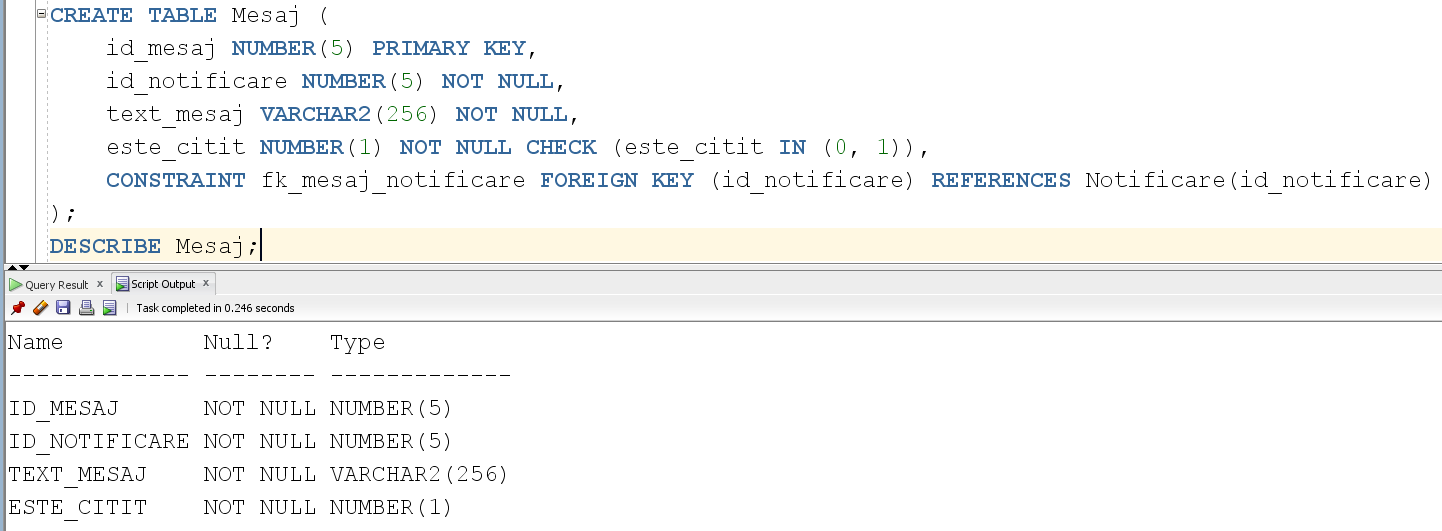
INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

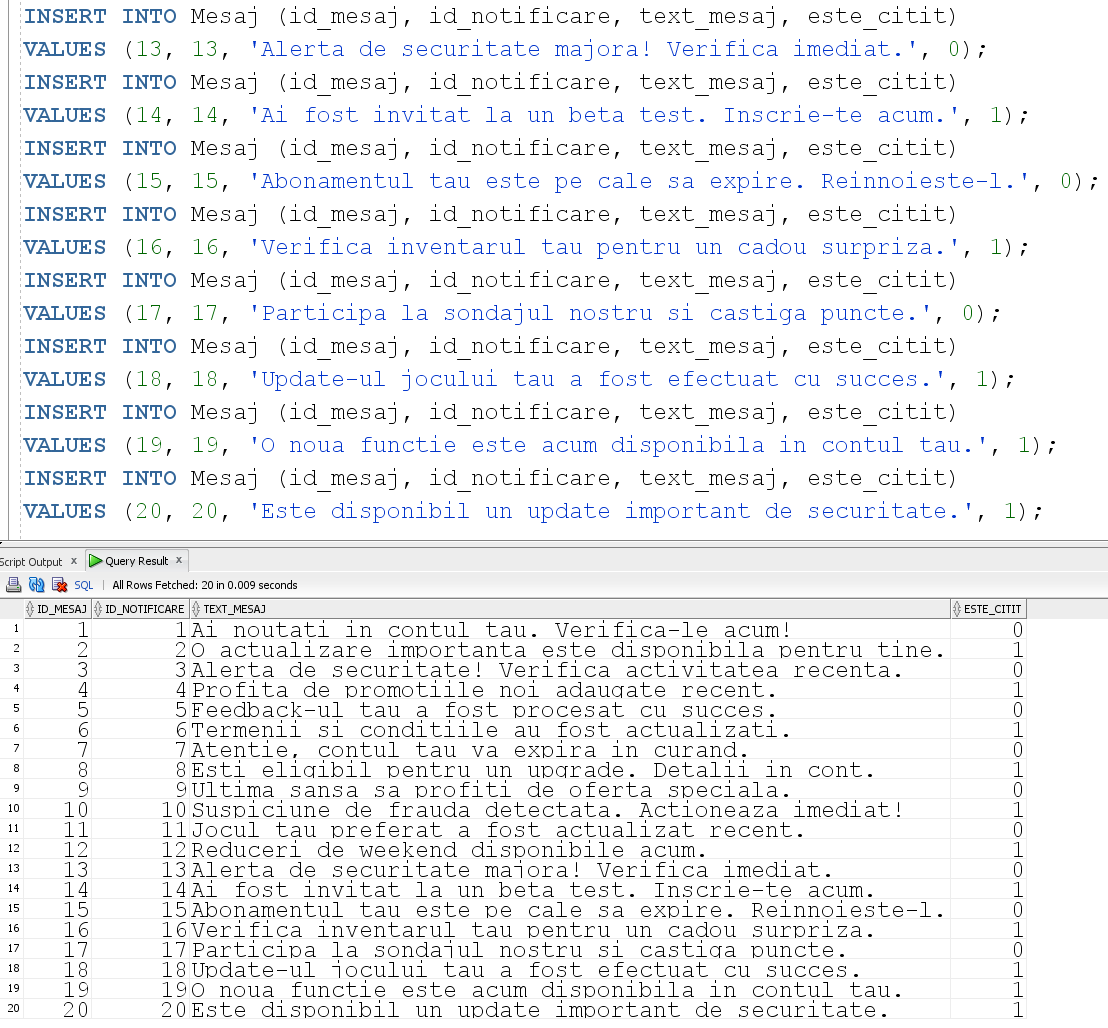
VALUES (19, 19, 'O noua functie este acum disponibila in contul tau.', 1);

INSERT INTO Mesaj (id\_mesaj, id\_notificare, text\_mesaj, este\_citit)

VALUES (20, 20, 'Este disponibil un update important de securitate.', 1);







## TARA

* CREATE

CREATE TABLE Tara (

id\_tara NUMBER(5) PRIMARY KEY,

nume\_tara VARCHAR2(50) NOT NULL UNIQUE,

salariu\_mediu NUMBER(5) NOT NULL CHECK (salariu\_mediu >= 0)

);

DESCRIBE Tara;

* INSERT

CREATE SEQUENCE seq\_tara

START WITH 1

INCREMENT BY 1

MINVALUE 1

MAXVALUE 12;

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'SUA', 55000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Japonia', 40000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Canada', 45000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Franta', 43000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Romania', 12000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Polonia', 27000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Suedia', 42000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Rusia', 15000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Estonia', 23000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Germania', 65000);

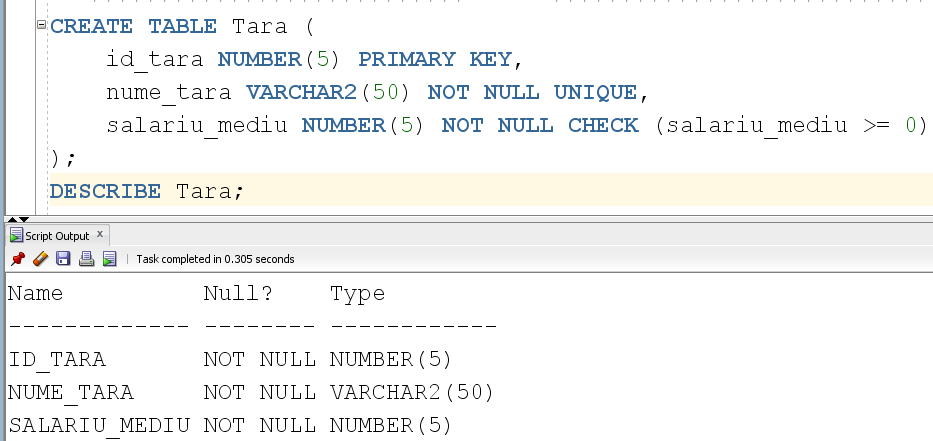
INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

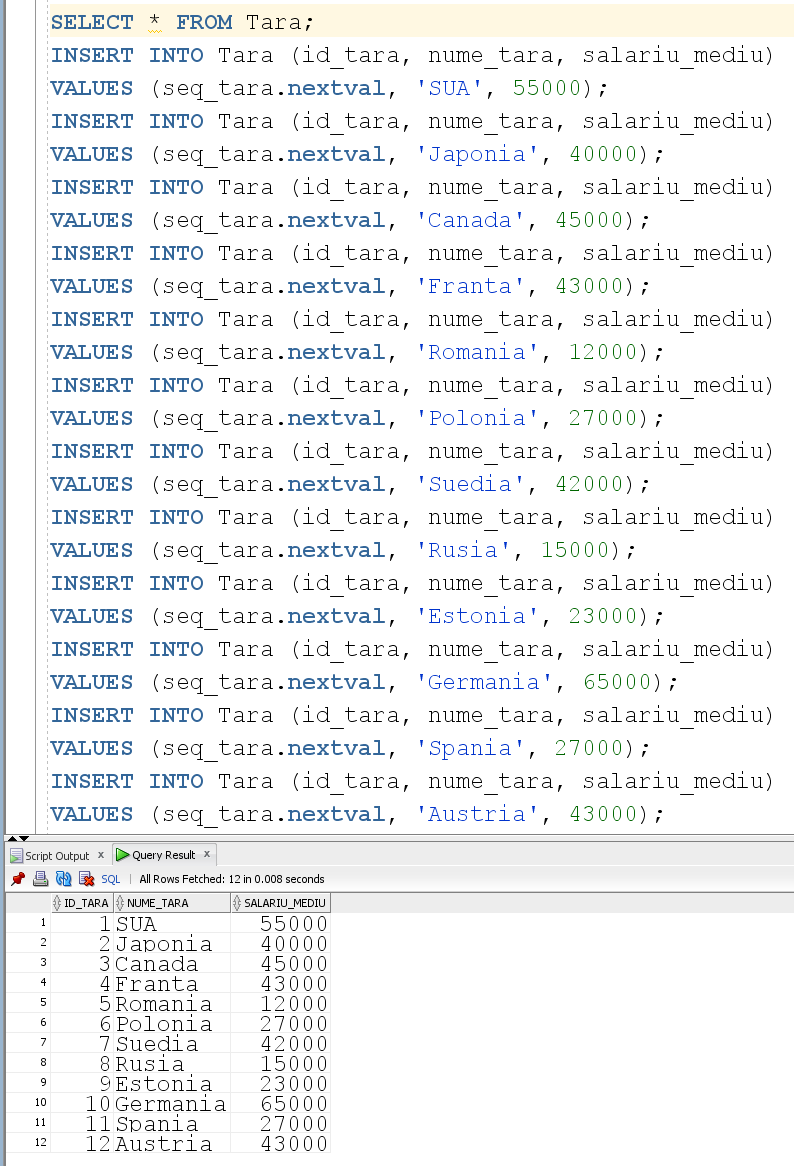
VALUES (seq\_tara.nextval, 'Spania', 27000);

INSERT INTO Tara (id\_tara, nume\_tara, salariu\_mediu)

VALUES (seq\_tara.nextval, 'Austria', 43000);

SELECT \* FROM Tara;





## LOCATIE

* CREATE

CREATE TABLE Locatie (

id\_locatie NUMBER(5) PRIMARY KEY,

id\_tara NUMBER(5) NOT NULL,

oras VARCHAR2(25) NOT NULL UNIQUE,

CONSTRAINT fk\_locatie\_tara FOREIGN KEY (id\_tara) REFERENCES Tara(id\_tara)

);

DESCRIBE Locatie;

* INSERT

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (1, 3, 'Montreal'); -- Canada, Ubisoft Montreal

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (2, 1, 'Redmond'); -- SUA, 343 Industries

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (3, 6, 'Varsovia'); -- Polonia, CD Projekt Red

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (4, 1, 'Santa Monica'); -- SUA, Naughty Dog

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (5, 2, 'Tokyo'); -- Japonia, Kojima Productions

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (6, 1, 'Bellevue'); -- SUA, Valve

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (7, 1, 'New York'); -- SUA, Rockstar Games

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (8, 1, 'Dallas'); -- SUA, id Software

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (9, 1, 'Frisco'); -- SUA, Gearbox Software

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (10, 7, 'Stockholm'); -- Suedia, Mojang Studios

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (11, 3, 'Vancouver'); -- Canada, EA Vancouver, FIFA23

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (12, 5, 'Bucuresti'); -- Romania, EAFC 24

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (13, 8, 'Moscova'); -- Rusia

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (14, 9, 'Tallinn'); -- Estonia

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (15, 10, 'Koln'); -- Germania

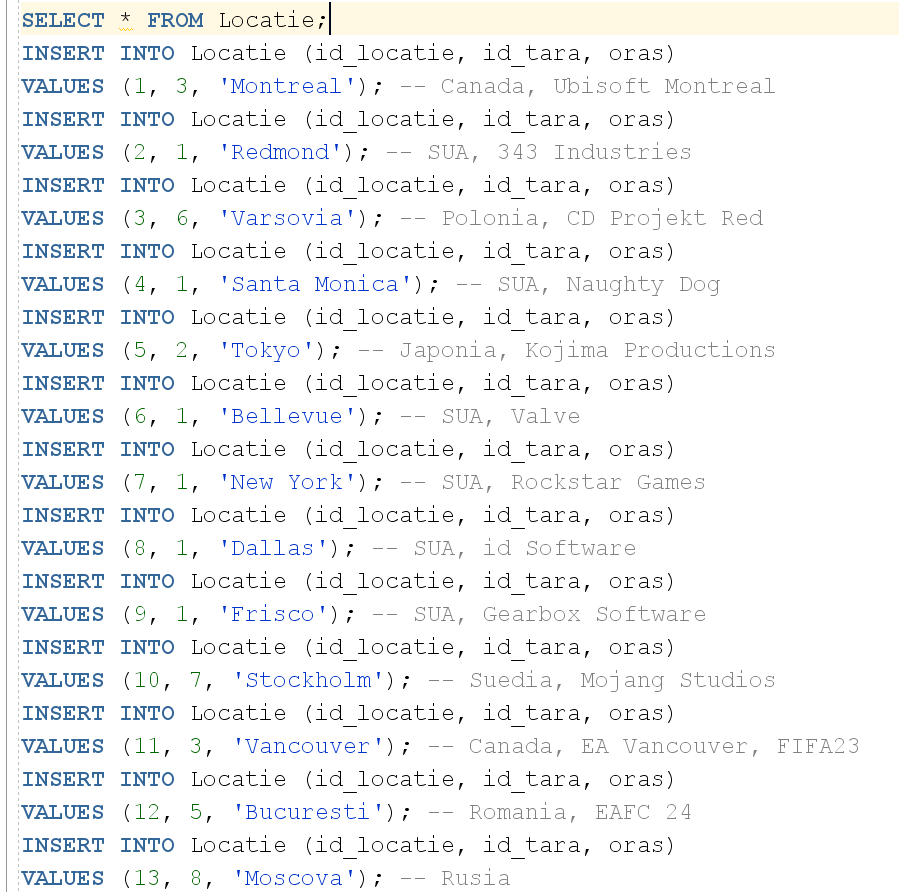
INSERT INTO Locatie (id\_locatie, id\_tara, oras)

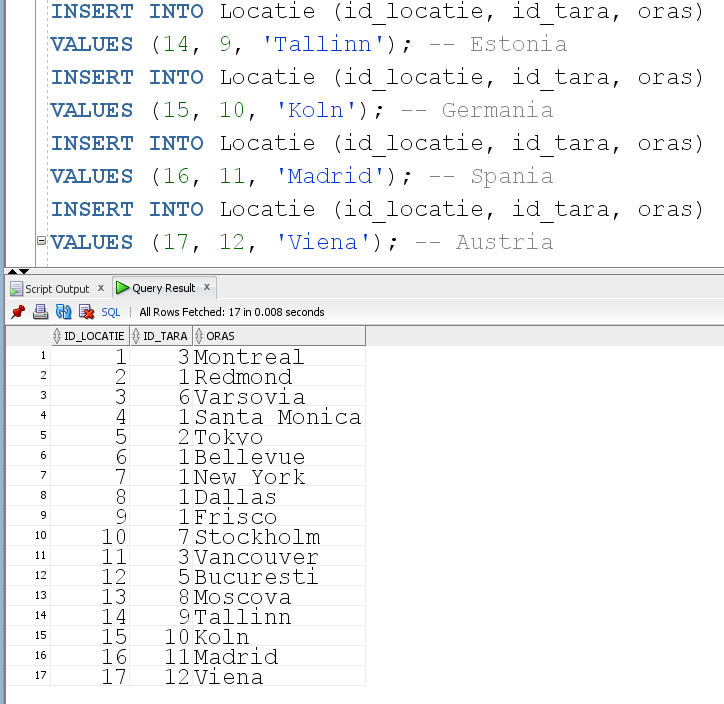
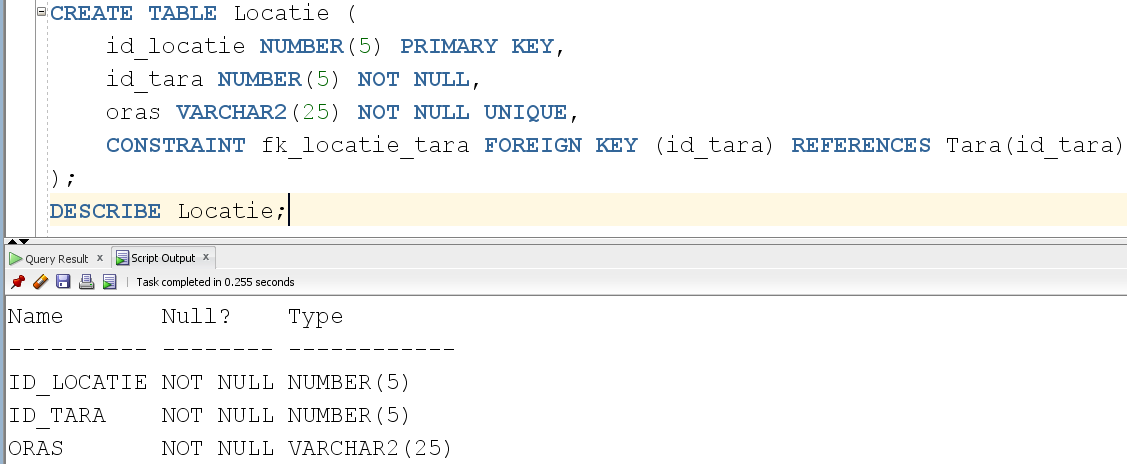
VALUES (16, 11, 'Madrid'); -- Spania

INSERT INTO Locatie (id\_locatie, id\_tara, oras)

VALUES (17, 12, 'Viena'); -- Austria

SELECT \* FROM Locatie;



## DEVELOPER

* CREATE

CREATE TABLE Developer (

id\_developer NUMBER(5) PRIMARY KEY,

id\_locatie NUMBER(5),

nume\_developer VARCHAR2(50) NOT NULL,

data\_lansare\_studio DATE NOT NULL,

salariu\_angajati NUMBER(5) NOT NULL CHECK (salariu\_angajati > 0),

CONSTRAINT fk\_developer\_locatie FOREIGN KEY (id\_locatie) REFERENCES Locatie(id\_locatie)

);

DESCRIBE Developer;

* INSERT

SELECT \* FROM Developer;

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (1, 1, 'Ubisoft Montreal', TO\_DATE('1997-01-01', 'YYYY-MM-DD'), 44000); -- Sub medie in Canada

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (2, 2, '343 Industries', TO\_DATE('2007-05-01', 'YYYY-MM-DD'), 53000); -- Sub medie in SUA

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (3, 3, 'CD Projekt Red', TO\_DATE('2002-06-01', 'YYYY-MM-DD'), 26000); -- Sub medie in Polonia

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (4, 4, 'Naughty Dog', TO\_DATE('1984-09-01', 'YYYY-MM-DD'), 60000); -- Peste medie in SUA

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (5, 5, 'Kojima Productions', TO\_DATE('2005-12-01', 'YYYY-MM-DD'), 45000); -- Peste medie in Japonia

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (6, 6, 'Valve Corporation', TO\_DATE('1996-08-01', 'YYYY-MM-DD'), 70000); -- Peste medie in SUA

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (7, 7, 'Rockstar Games', TO\_DATE('1998-12-01', 'YYYY-MM-DD'), 65000); -- Peste medie in SUA

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (8, 8, 'id Software', TO\_DATE('1991-02-01', 'YYYY-MM-DD'), 62000); -- Peste medie in SUA

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (9, 9, 'Gearbox Software', TO\_DATE('1999-02-01', 'YYYY-MM-DD'), 58000); -- Peste medie in SUA

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

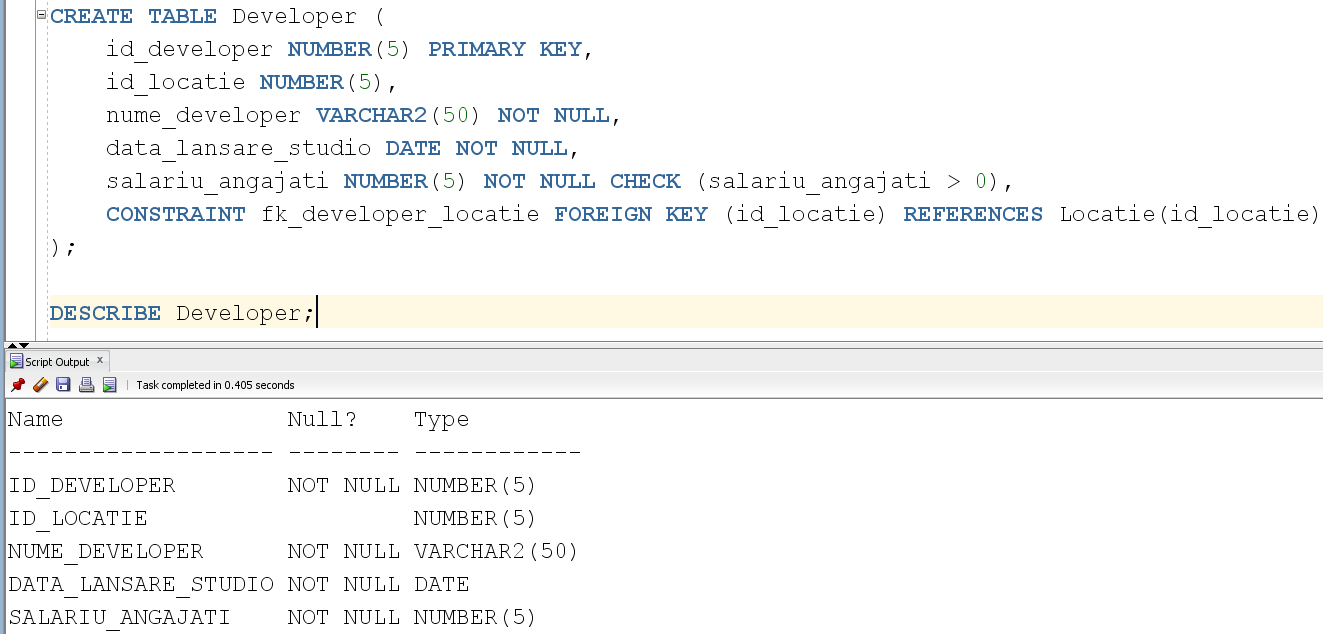
VALUES (10, 10, 'Mojang Studios', TO\_DATE('2009-05-01', 'YYYY-MM-DD'), 50000); -- Peste medie in Suedia

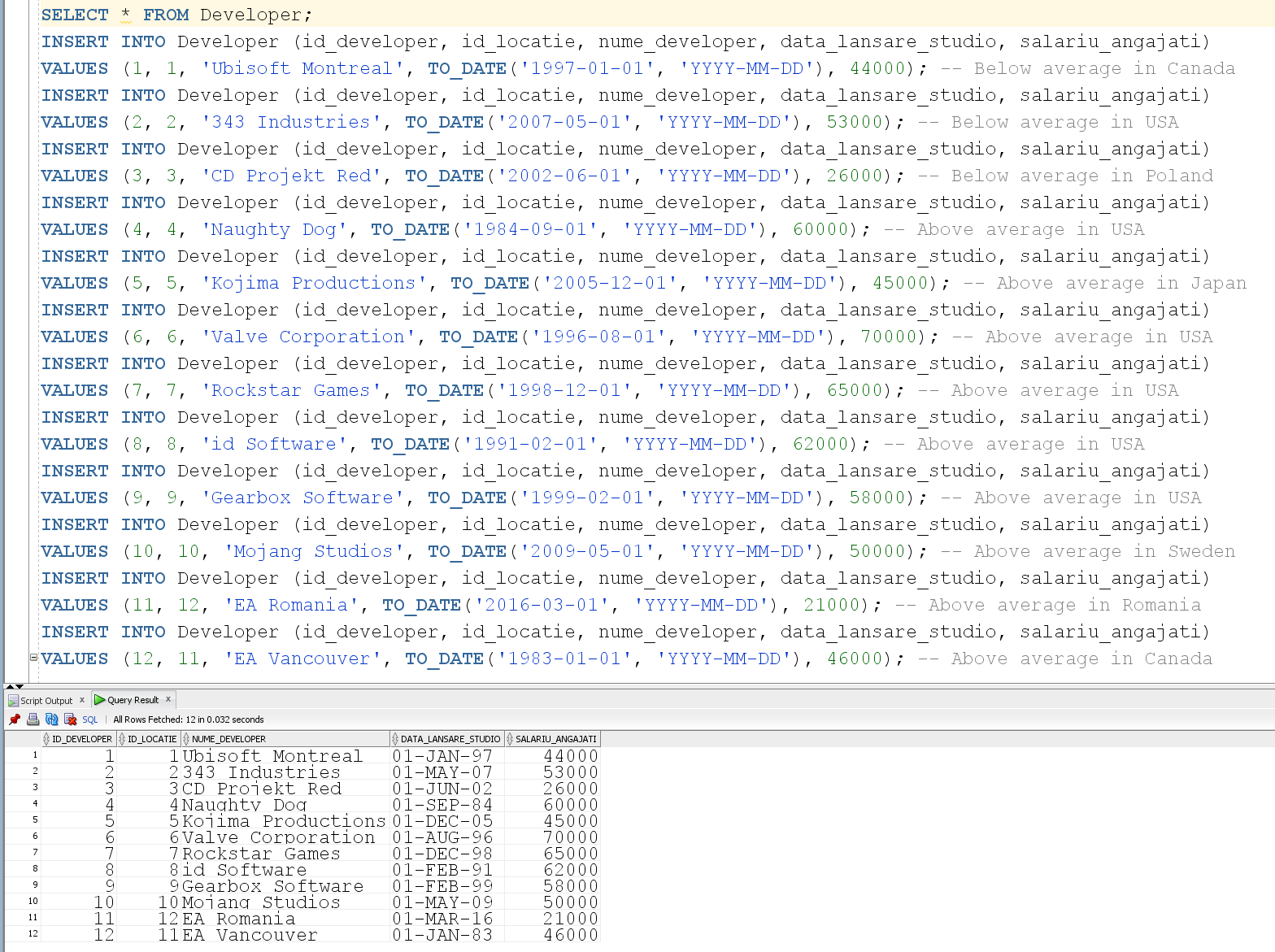
INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (11, 12, 'EA Romania', TO\_DATE('2016-03-01', 'YYYY-MM-DD'), 21000); -- Peste medie in Romania

INSERT INTO Developer (id\_developer, id\_locatie, nume\_developer, data\_lansare\_studio, salariu\_angajati)

VALUES (12, 11, 'EA Vancouver', TO\_DATE('1983-01-01', 'YYYY-MM-DD'), 46000); -- Peste medie in Canada





## JOC

* CREATE

CREATE TABLE Joc (

id\_joc NUMBER(5) PRIMARY KEY,

id\_developer NUMBER(5) NOT NULL,

nume\_joc VARCHAR2(50) NOT NULL UNIQUE,

pret\_joc NUMBER(5) NOT NULL CHECK (pret\_joc >= 0),

data\_lansare\_joc DATE NOT NULL,

varsta\_recomandata NUMBER(3) CHECK (varsta\_recomandata >= 0),

rating NUMBER(1) CHECK (rating BETWEEN 1 AND 5),

CONSTRAINT fk\_joc\_developer FOREIGN KEY (id\_developer) REFERENCES Developer(id\_developer)

);

DESCRIBE Joc;

* INSERT

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (1, 1, 'Assassins Creed Valhalla', 60, TO\_DATE('2020-11-10', 'YYYY-MM-DD'), 18, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (2, 2, 'Halo Infinite', 60, TO\_DATE('2021-12-08', 'YYYY-MM-DD'), 16, 3);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (3, 3, 'Cyberpunk 2077', 60, TO\_DATE('2020-12-10', 'YYYY-MM-DD'), 18, 3);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (4, 4, 'The Last of Us Part II', 60, TO\_DATE('2020-06-19', 'YYYY-MM-DD'), 18, 5);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (5, 5, 'Death Stranding', 60, TO\_DATE('2019-11-08', 'YYYY-MM-DD'), 16, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (6, 6, 'Ghost of Tsushima', 60, TO\_DATE('2020-07-17', 'YYYY-MM-DD'), 18, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (7, 5, 'Monster Hunter World', 60, TO\_DATE('2018-01-26', 'YYYY-MM-DD'), 16, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (8, 7, 'Red Dead Redemption 2', 60, TO\_DATE('2018-10-26', 'YYYY-MM-DD'), 18, 5);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (9, 8, 'Doom Eternal', 60, TO\_DATE('2020-03-20', 'YYYY-MM-DD'), 18, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (10, 9, 'Borderlands 3', 60, TO\_DATE('2019-09-13', 'YYYY-MM-DD'), 18, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (11, 10, 'Minecraft', 30, TO\_DATE('2011-11-18', 'YYYY-MM-DD'), 10, 5);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (12, 11, 'EAFC 24', 60, TO\_DATE('2030-09-30', 'YYYY-MM-DD'), 3, 4);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (13, 12, 'FIFA 23', 60, TO\_DATE('2022-09-30', 'YYYY-MM-DD'), 3, 3);

INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (14, 6, 'CS:GO', 0, TO\_DATE('2012-08-21', 'YYYY-MM-DD'), 16, 5);

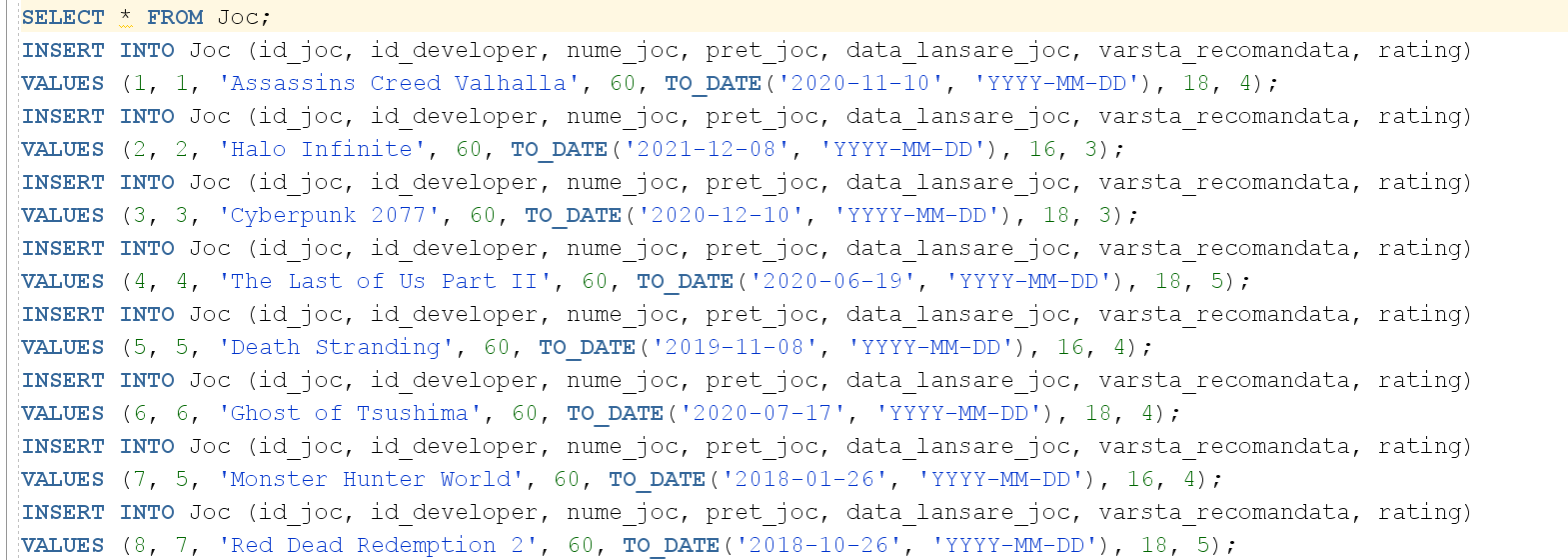
INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

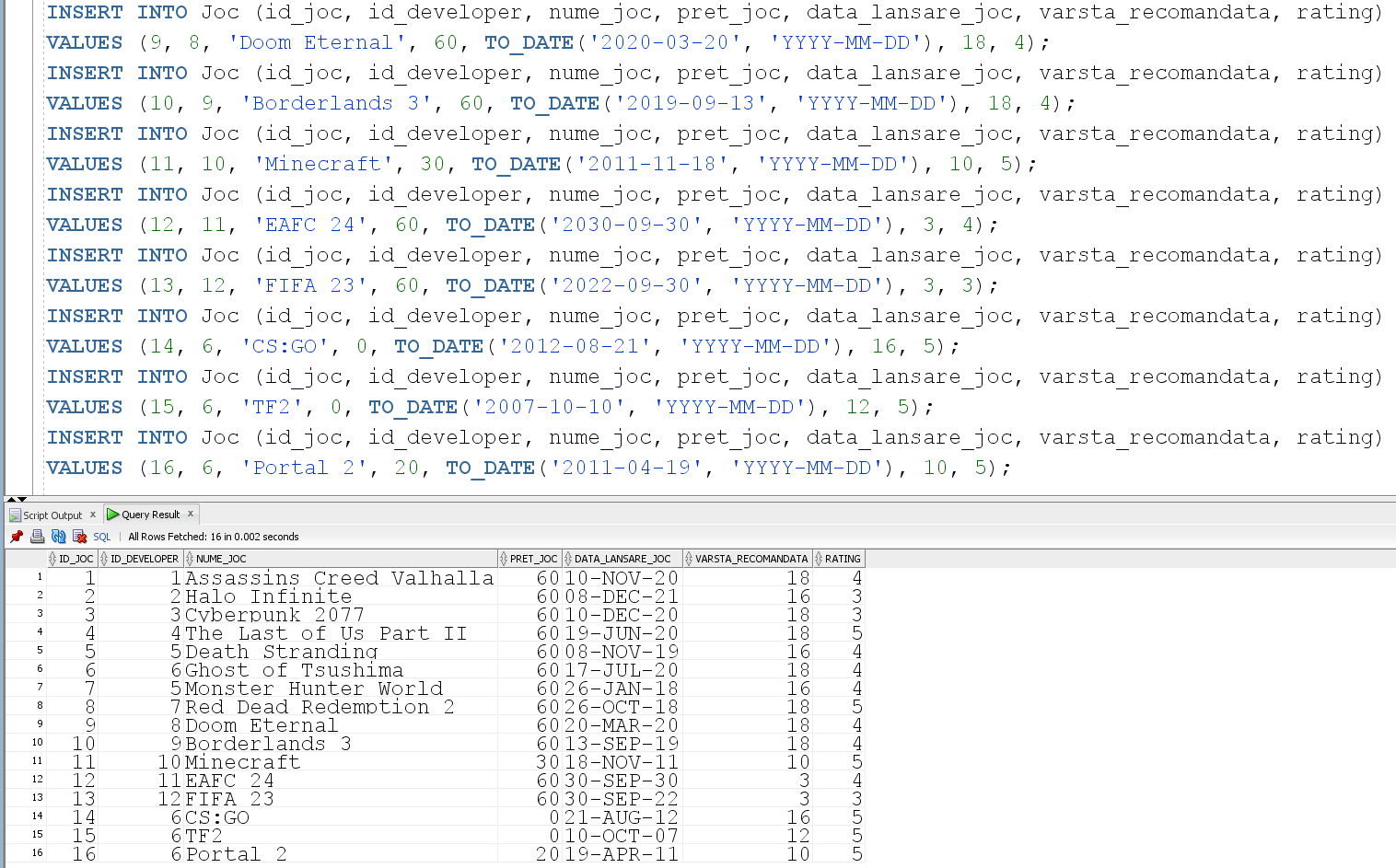
VALUES (15, 6, 'TF2', 0, TO\_DATE('2007-10-10', 'YYYY-MM-DD'), 12, 5);

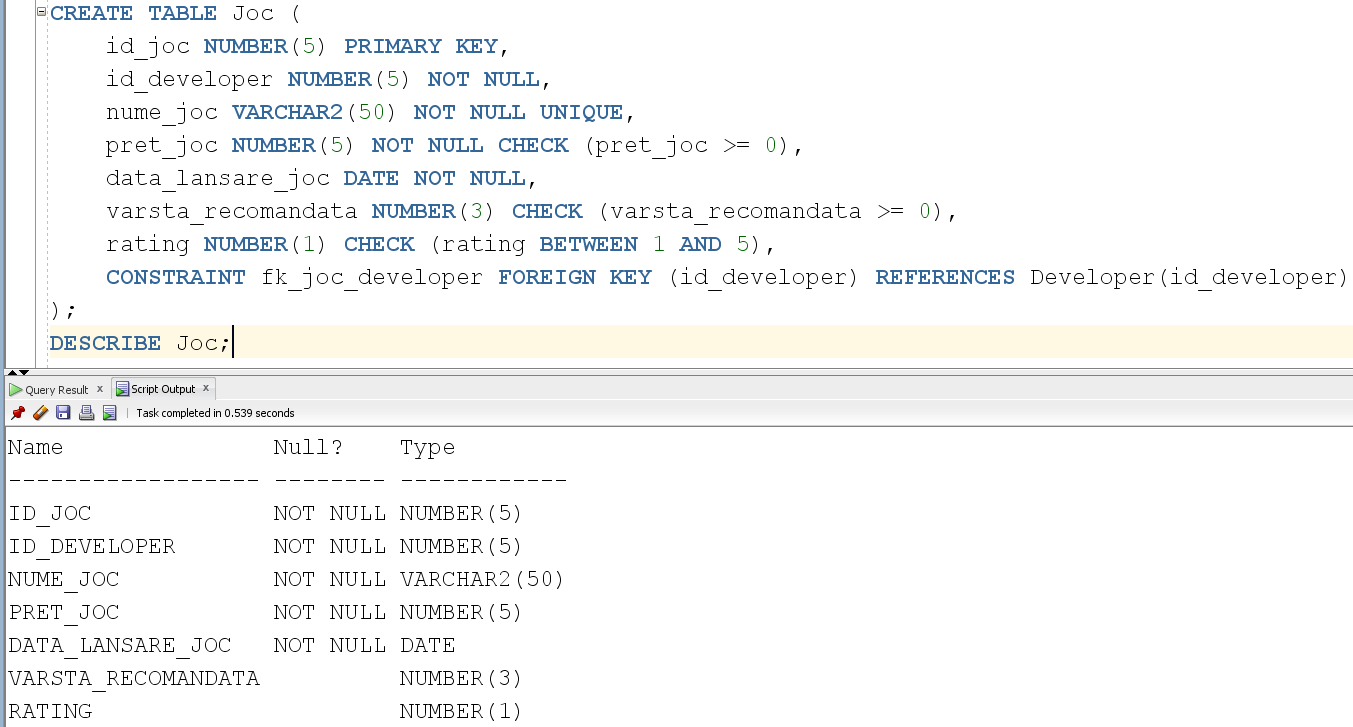
INSERT INTO Joc (id\_joc, id\_developer, nume\_joc, pret\_joc, data\_lansare\_joc, varsta\_recomandata, rating)

VALUES (16, 6, 'Portal 2', 20, TO\_DATE('2011-04-19', 'YYYY-MM-DD'), 10, 5);

SELECT \* FROM Joc;







## RECENZIE

* CREATE

CREATE TABLE Recenzie (

id\_recenzie NUMBER(5) PRIMARY KEY,

id\_utilizator NUMBER(5) NOT NULL,

id\_joc NUMBER(5) NOT NULL,

text\_recenzie VARCHAR2(256) NOT NULL,

data\_recenzie DATE NOT NULL,

CONSTRAINT fk\_recenzie\_utilizator FOREIGN KEY (id\_utilizator) REFERENCES Utilizator(id\_utilizator),

CONSTRAINT fk\_recenzie\_joc FOREIGN KEY (id\_joc) REFERENCES Joc(id\_joc)

);

DESCRIBE Recenzie;

* INSERT

SELECT \* FROM Recenzie;

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (1, 1, 1, 'Explorarea in stil viking este detaliata si captivanta. POZITIV', TO\_DATE('2020-08-10', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (2, 5, 2, 'Gameplay-ul este robust, dar campania este scurta. NEGATIV', TO\_DATE('2022-01-15', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (3, 8, 3, 'Viziunea futurista este impresionanta, dar plina de bug-uri. NEGATIV', TO\_DATE('2020-09-10', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (4, 3, 4, 'Poveste profunda si emotionanta. POZITIV', TO\_DATE('2020-12-20', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (5, 4, 5, 'Jocul redefineste genul, dar nu este pentru toata lumea. NEGATIV', TO\_DATE('2019-08-08', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (6, 2, 6, 'Un peisaj deschis superb si lupte detaliate. POZITIV', TO\_DATE('2020-10-17', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (7, 7, 7, 'Complex si detaliat, dar dificil pentru incepatori. NEGATIV', TO\_DATE('2018-07-26', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (8, 6, 8, 'O capodopera narativa cu un ritm lent. POZITIV', TO\_DATE('2018-04-26', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (9, 1, 9, 'Actiune non-stop si grafica impresionanta. POZITIV', TO\_DATE('2020-03-19', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (10, 5, 10, 'Lume uriasa ce asteapta sa fie explorata. POZITIV', TO\_DATE('2019-09-14', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (11, 1, 11, 'Creativitate fara limite si distractie continua. POZITIV', TO\_DATE('2023-10-01', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (12, 5, 12, 'Evolutie impresionanta in simularea fotbalului, gameplay rafinat. POZITIV', TO\_DATE('2030-09-29', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (13, 8, 13, 'Simulare realista a fotbalului, dar cu microtranzactii excesive. NEGATIV', TO\_DATE('2022-10-01', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

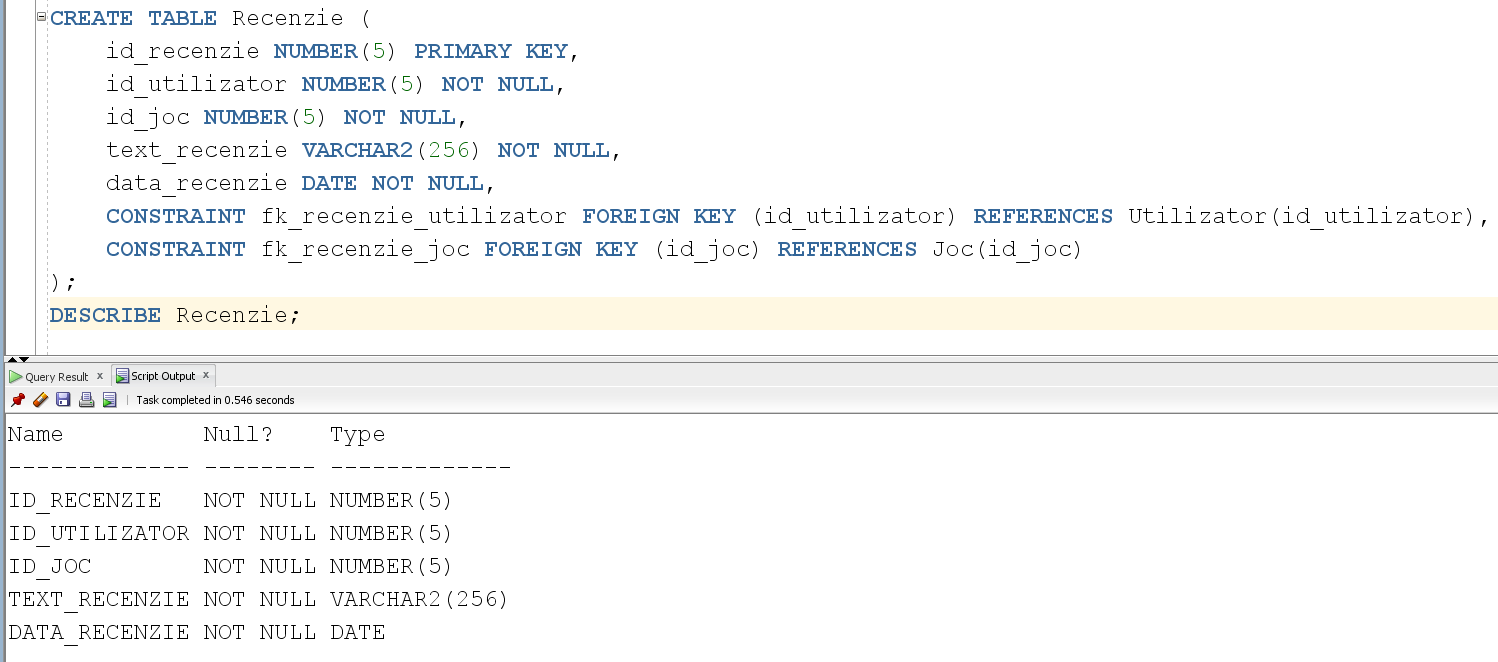
VALUES (14, 1, 14, 'Competitiv si captivant, un clasic al shooterelor. POZITIV', TO\_DATE('2012-09-01', 'YYYY-MM-DD'));

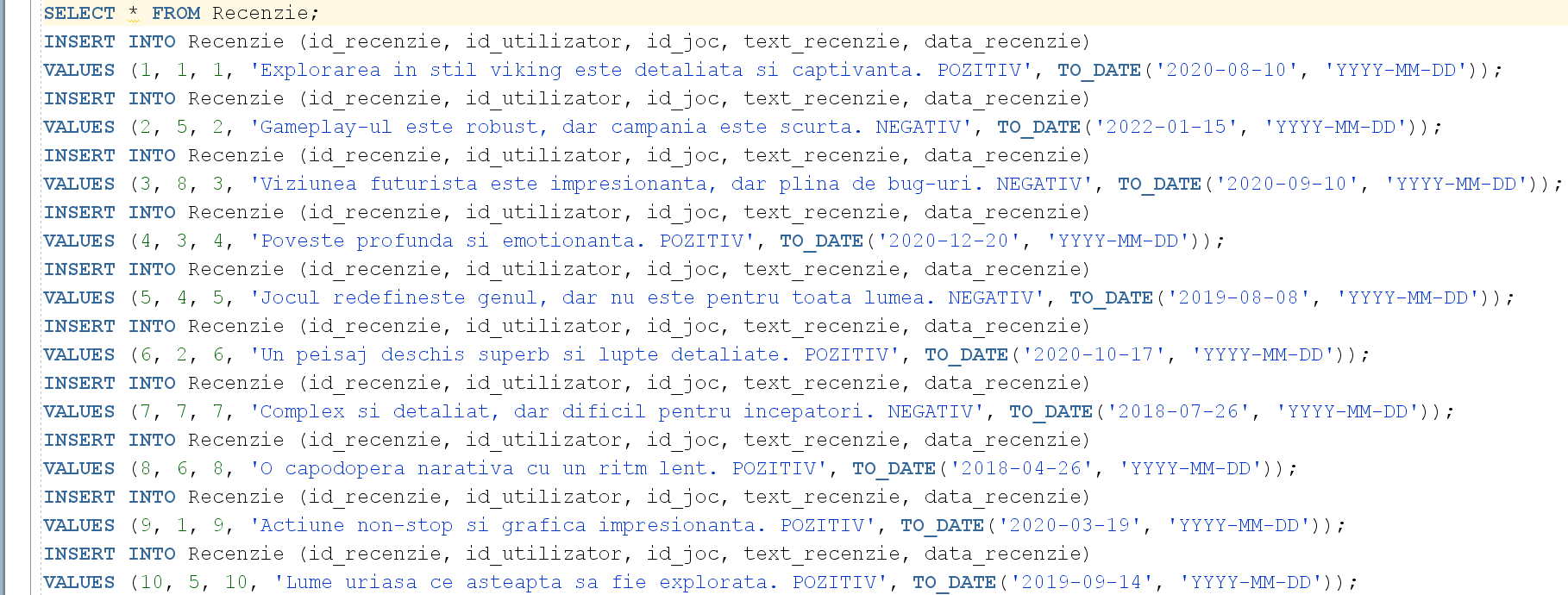
INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

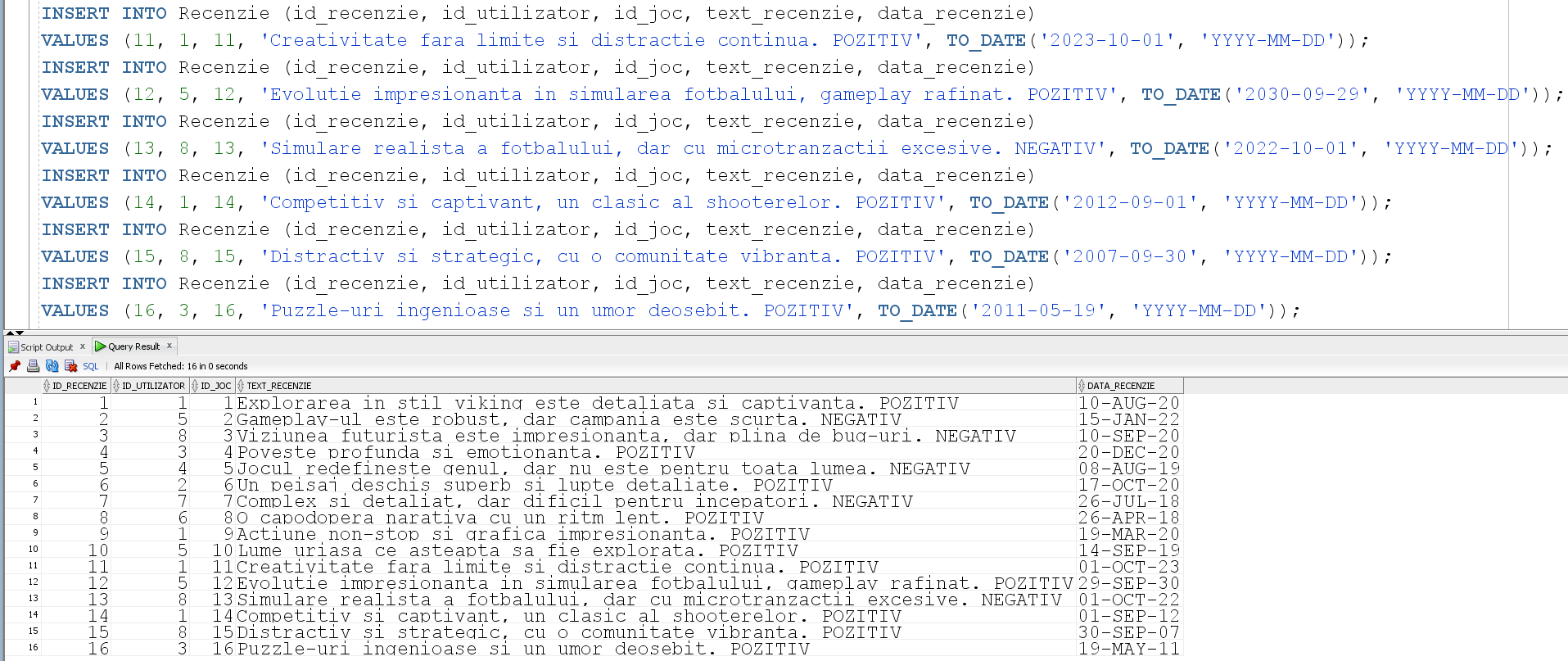
VALUES (15, 8, 15, 'Distractiv si strategic, cu o comunitate vibranta. POZITIV', TO\_DATE('2007-09-30', 'YYYY-MM-DD'));

INSERT INTO Recenzie (id\_recenzie, id\_utilizator, id\_joc, text\_recenzie, data\_recenzie)

VALUES (16, 3, 16, 'Puzzle-uri ingenioase si un umor deosebit. POZITIV', TO\_DATE('2011-05-19', 'YYYY-MM-DD'));







## COLECTIE

* CREATE

CREATE TABLE Colectie (

id\_colectie NUMBER(5) PRIMARY KEY,

id\_utilizator NUMBER(5) NOT NULL,

nume\_colectie VARCHAR2(50) NOT NULL,

CONSTRAINT fk\_colectie\_utilizator FOREIGN KEY (id\_utilizator) REFERENCES Utilizator(id\_utilizator)

);

DESCRIBE Colectie;

* INSERT

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (1, 1, 'Calatorii in Timp');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (2, 1, 'Exploarea spatiului');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (3, 2, 'Cosmar');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (4, 2, 'Lumi Deschise');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (5, 3, 'Actiune Intensa');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (6, 3, 'Cupe Mondiale');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (7, 4, 'RPG-uri Epice');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (8, 4, 'Shootere Tactice');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (9, 5, 'Paranormal');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (10, 5, 'Adrenalina la Maxim');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (11, 7, 'Povesti Epice');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (12, 7, 'Strategie si Tactici');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

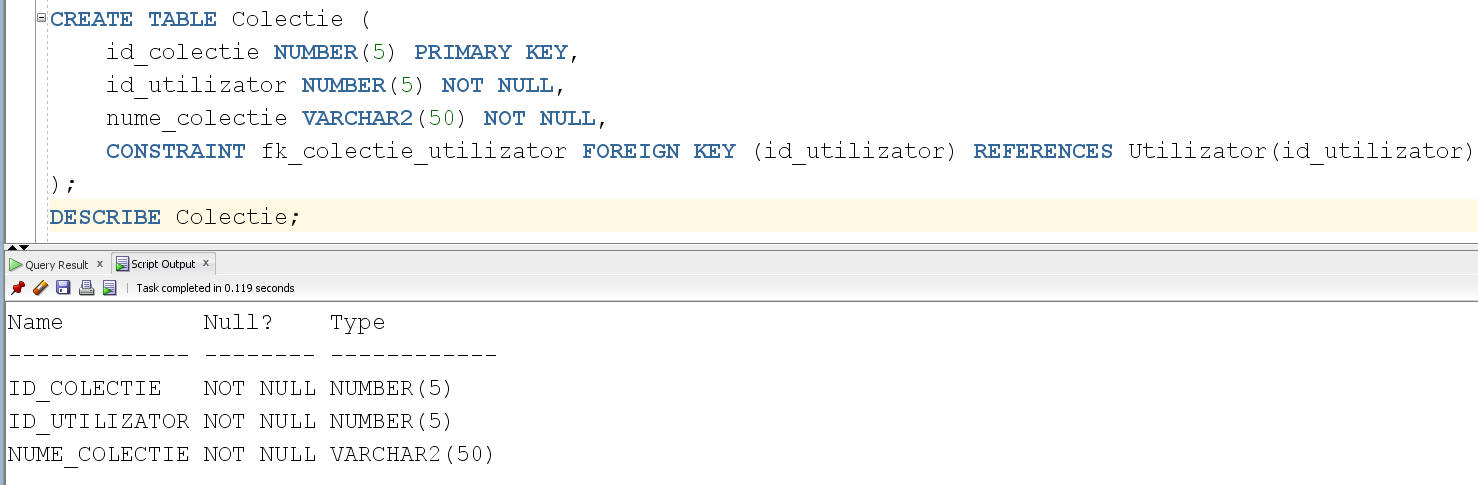
VALUES (13, 8, 'Sci-Fi si Fantezie');

INSERT INTO Colectie (id\_colectie, id\_utilizator, nume\_colectie)

VALUES (14, 8, 'Arena Sportiva');

SELECT \* FROM Colectie;





## COLECTIE\_JOC

* CREATE

CREATE TABLE Colectie\_Joc (

id\_colectie NUMBER(5) NOT NULL,

id\_joc NUMBER(5) NOT NULL,

PRIMARY KEY (id\_colectie, id\_joc),

CONSTRAINT fk\_colectie\_joc\_colectie FOREIGN KEY (id\_colectie) REFERENCES Colectie(id\_colectie),

CONSTRAINT fk\_colectie\_joc\_joc FOREIGN KEY (id\_joc) REFERENCES Joc(id\_joc)

);

DESCRIBE Colectie\_Joc;

* INSERT

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (1, 1);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (2, 2);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (3, 9);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (4, 8);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (5, 4);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (6, 13);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (7, 3);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (8, 14);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (9, 16);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (10, 5);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (11, 11);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (12, 11);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (13, 4);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (14, 2);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (1, 3);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (14, 13);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (4, 6);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (7, 7);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (10, 9);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (5, 10);

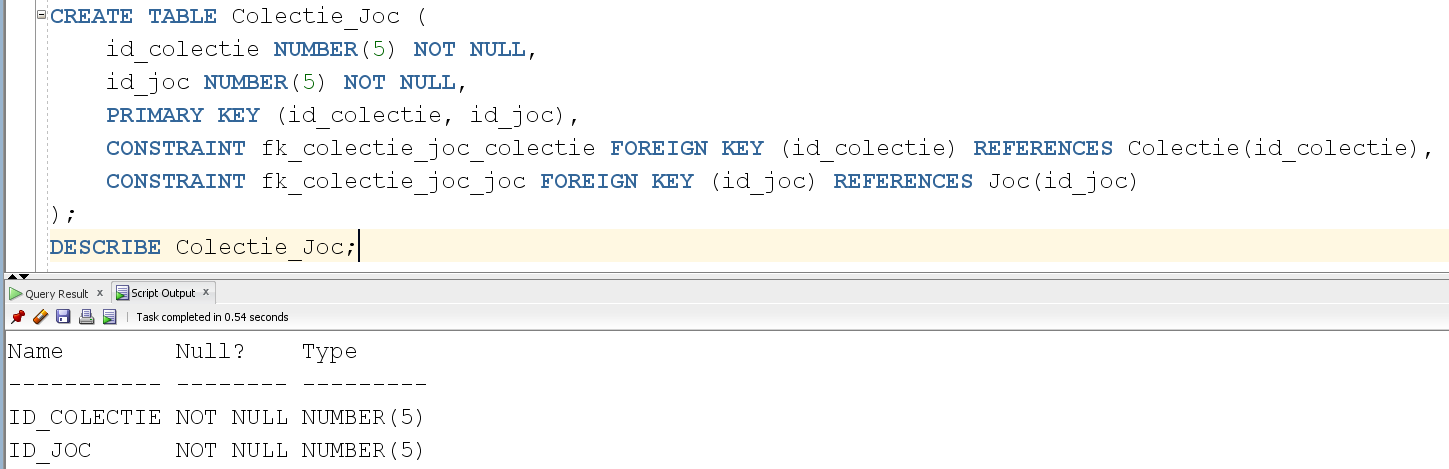
INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (8, 15);

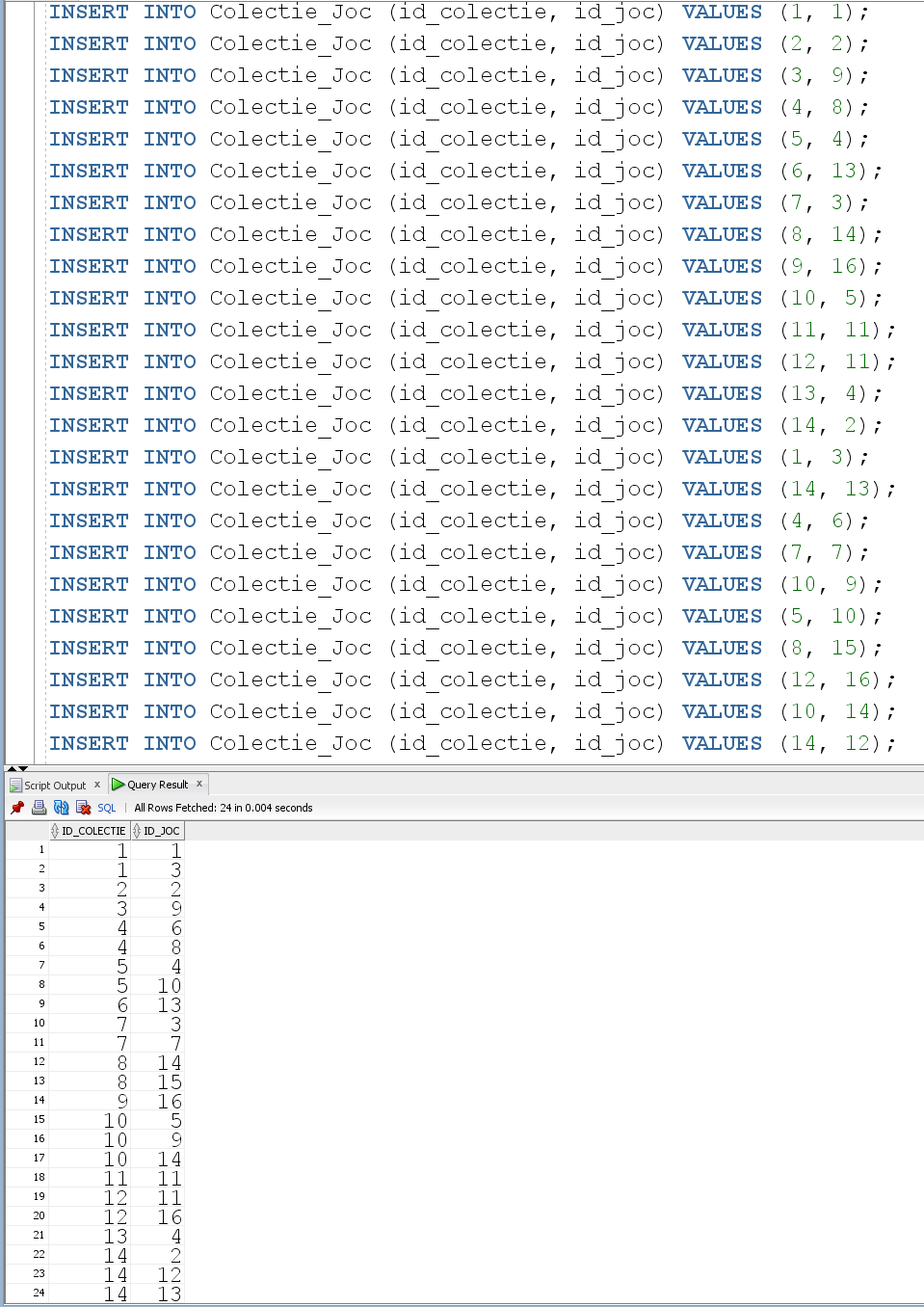
INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (12, 16);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (10, 14);

INSERT INTO Colectie\_Joc (id\_colectie, id\_joc) VALUES (14, 12);

SELECT \* FROM Colectie\_Joc;





## INVENTAR

* CREATE

CREATE TABLE Inventar (

id\_inventar NUMBER(5) PRIMARY KEY,

id\_utilizator NUMBER(5) NOT NULL,

capacitate NUMBER(2) NOT NULL CHECK (capacitate > 0),

CONSTRAINT fk\_inventar\_utilizator FOREIGN KEY (id\_utilizator) REFERENCES Utilizator(id\_utilizator)

);

DESCRIBE Inventar;

* INSERT

INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (1, 1, 10);

INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (2, 2, 3);

INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (3, 3, 5);

INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (4, 4, 6);

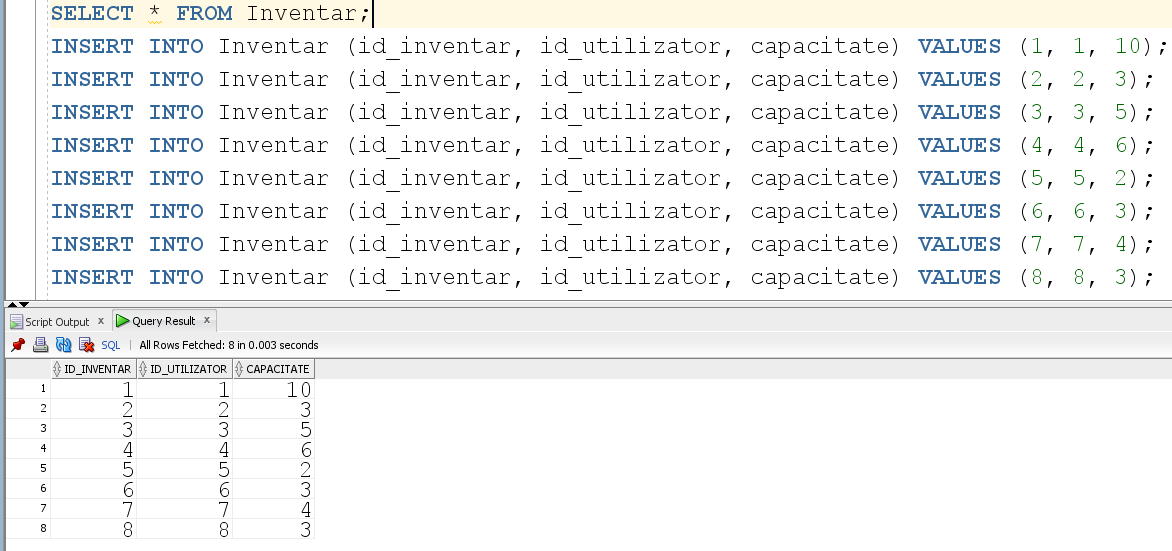
INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (5, 5, 2);

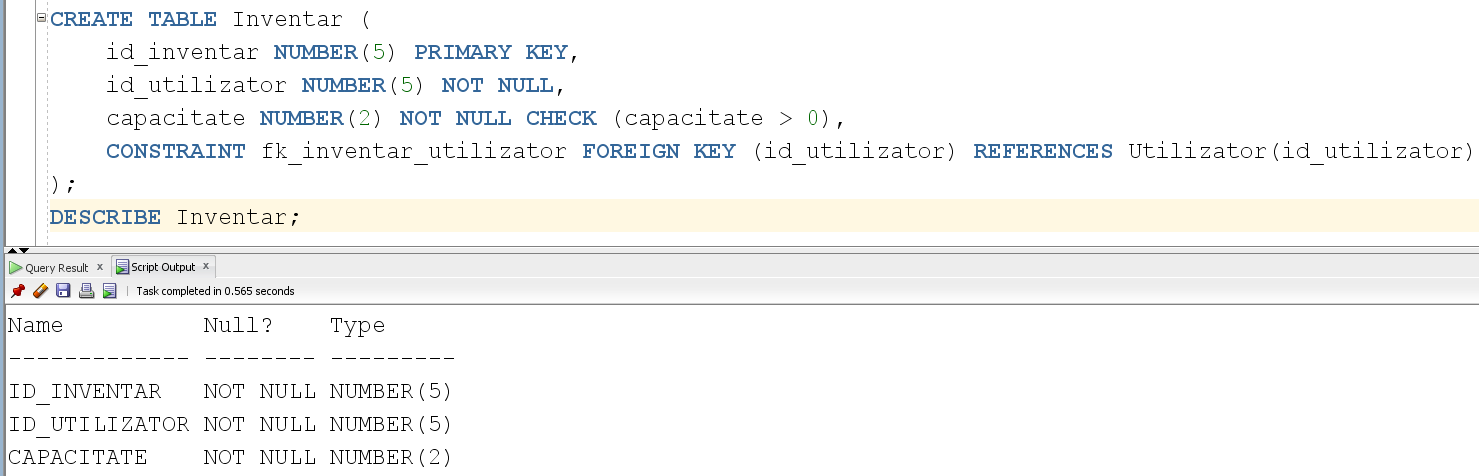
INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (6, 6, 3);

INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (7, 7, 4);

INSERT INTO Inventar (id\_inventar, id\_utilizator, capacitate) VALUES (8, 8, 3);

SELECT \* FROM Inventar;





## ITEM

* CREATE

CREATE TABLE Item (

id\_item NUMBER(5) PRIMARY KEY,

id\_joc NUMBER(5) NOT NULL,

nume\_item VARCHAR2(100) NOT NULL,

pret\_item NUMBER(10, 2),

raritate\_item VARCHAR2(50) NOT NULL,

CONSTRAINT fk\_item\_joc FOREIGN KEY (id\_joc) REFERENCES Joc(id\_joc)

);

DESCRIBE Item;

* INSERT

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (1, 'AK-47 | Redline', 25.00, 'CLASSIFIED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (2, 'AWP | Dragon Lore', 1500.00, 'COVERT', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (3, 'Mann Co. Supply Crate Key', 2.50, 'COMMON', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (4, 'Unusual Hat', 200.00, 'UNUSUAL', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (5, 'Glock-18 | Fade', 300.00, 'COVERT', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (6, 'Strange Rocket Launcher', 50.00, 'STRANGE', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (7, 'M4A4 | Howl', 1800.00, 'CONTRABAND', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (8, 'Strange Professional Killstreak Scattergun', 100.00, 'STRANGE', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (9, 'Desert Eagle | Blaze', 600.00, 'COVERT', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (10, 'Unusual Taunt', 150.00, 'UNUSUAL', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (11, 'Karambit | Doppler', 1200.00, 'COVERT', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (12, 'Australium Wrench', 100.00, 'AUSTRALIUM', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (13, 'P90 | Asiimov', 50.00, 'CLASSIFIED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (14, 'Strange Professional Killstreak Minigun', 200.00, 'STRANGE', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (15, 'USP-S | Kill Confirmed', 100.00, 'COVERT', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (16, 'Unusual Rocket Launcher', 300.00, 'UNUSUAL', 15);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (17, 'AK-47 | Elite Build', 1.50, 'MIL-SPEC', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (18, 'AWP | Worm God', 2.00, 'MIL-SPEC', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (19, 'FAMAS | Pulse', 0.75, 'RESTRICTED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (20, 'Galil AR | Chatterbox', 3.00, 'RESTRICTED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (21, 'UMP-45 | Primal Saber', 1.20, 'RESTRICTED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (22, 'CZ75-Auto | The Fuschia Is Now', 0.90, 'RESTRICTED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (23, 'MP7 | Bloodsport', 1.50, 'RESTRICTED', 14);

INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (24, 'SG 553 | Pulse', 0.70, 'MIL-SPEC', 14);

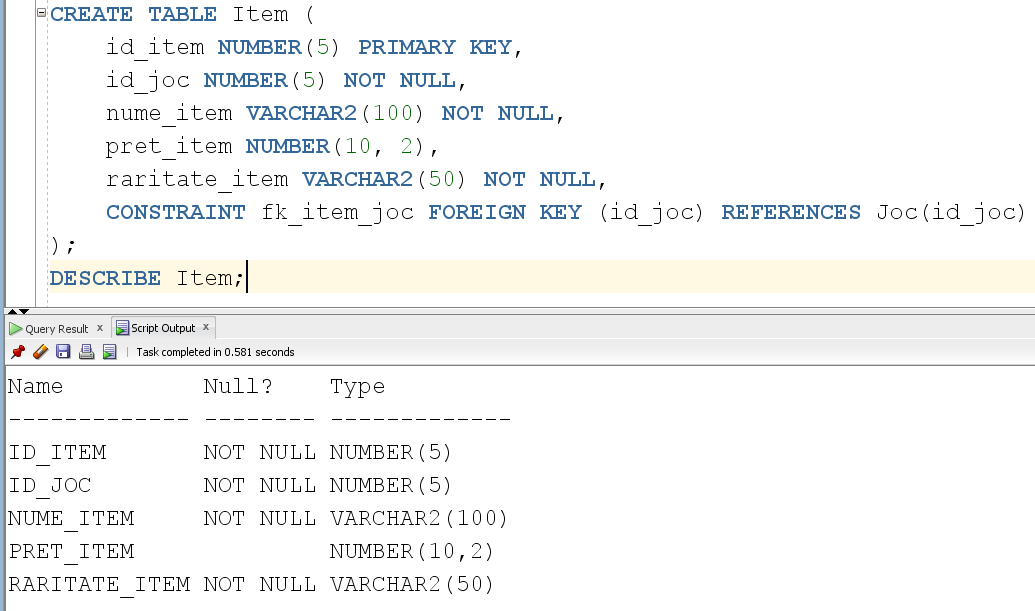
INSERT INTO Item (id\_item, nume\_item, pret\_item, raritate\_item, id\_joc)

VALUES (25, 'P250 | Supernova', 0.65, 'MIL-SPEC', 14);

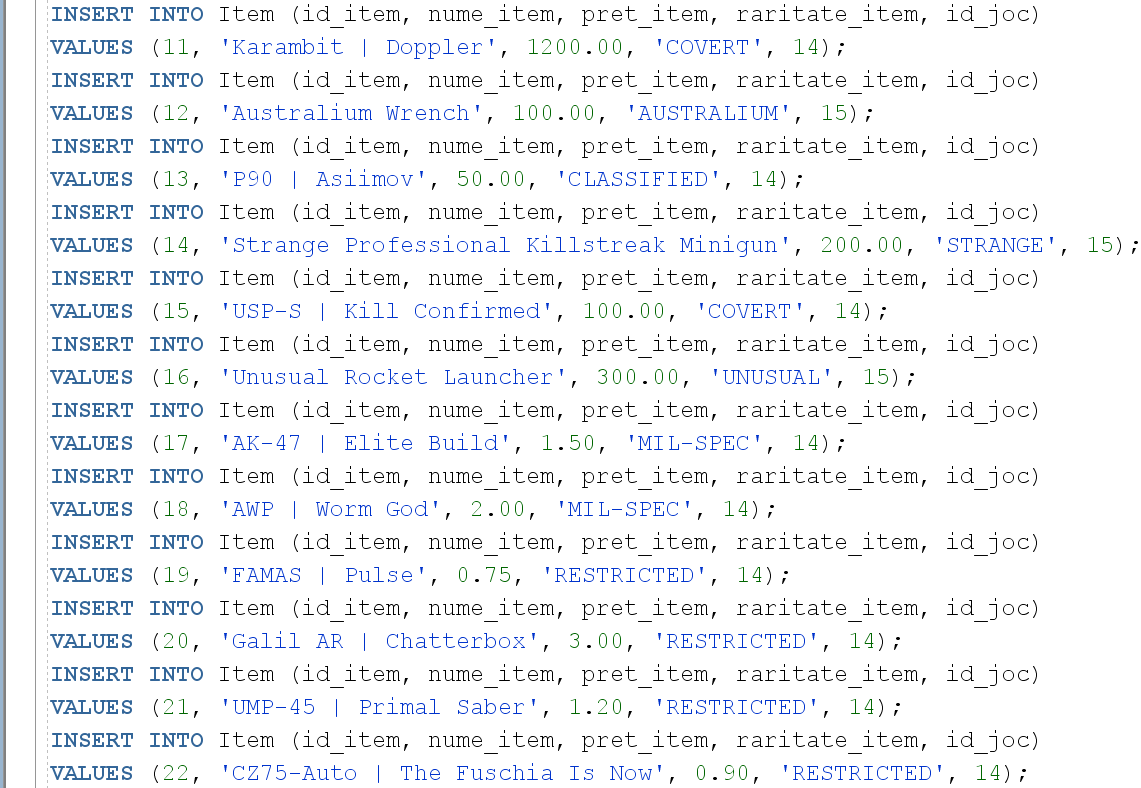
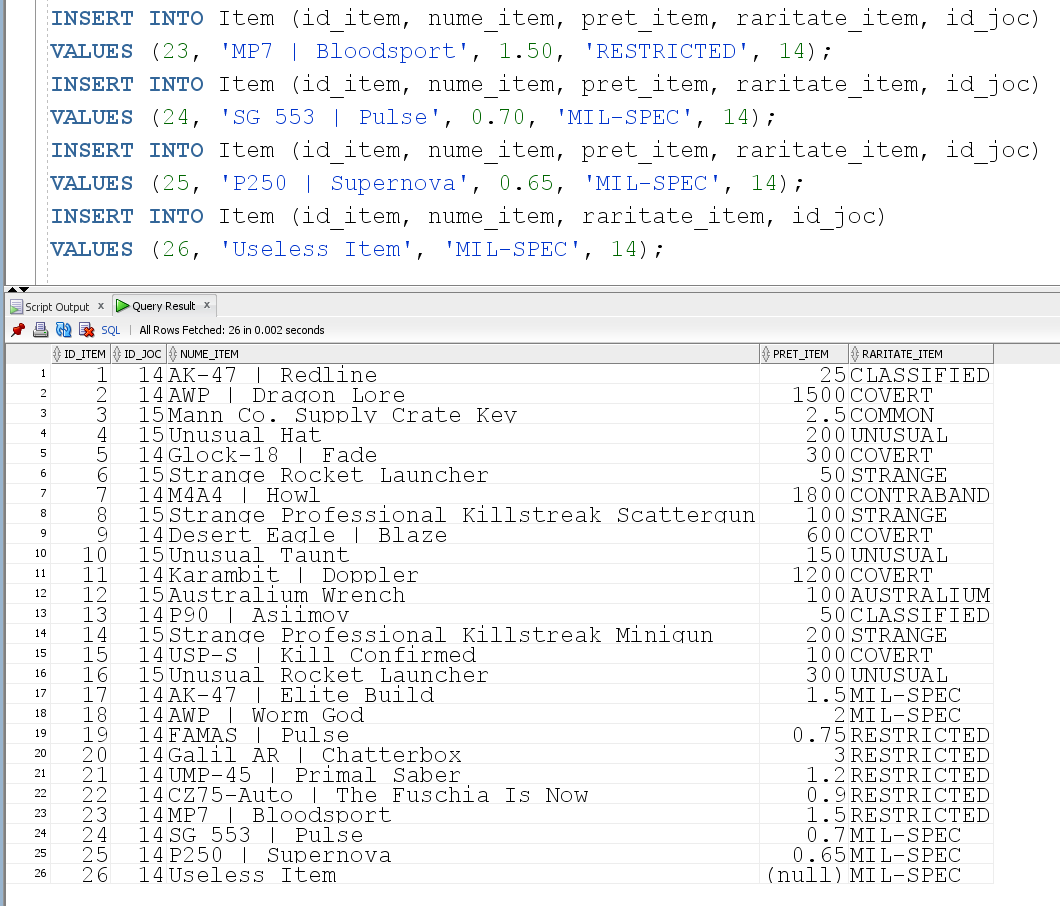
INSERT INTO Item (id\_item, nume\_item, raritate\_item, id\_joc)

VALUES (26, 'Useless Item', 'MIL-SPEC', 14);

SELECT \* FROM Item;





## INVENTAR\_ITEM

* CREATE

CREATE TABLE Inventar\_Item (

id\_item NUMBER(5) NOT NULL,

id\_inventar NUMBER(5) NOT NULL,

PRIMARY KEY (id\_item, id\_inventar),

CONSTRAINT fk\_inventar\_item\_item FOREIGN KEY (id\_item) REFERENCES Item(id\_item),

CONSTRAINT fk\_inventar\_item\_inventar FOREIGN KEY (id\_inventar) REFERENCES Inventar(id\_inventar)

);

DESCRIBE Inventar\_Item;

* INSERT

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (5, 1);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (12, 1);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (7, 1);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (19, 1);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (3, 1);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (26, 1);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (9, 2);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (15, 2);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (22, 2);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (1, 2);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (14, 3);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (6, 3);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (17, 3);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (20, 4);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (21, 4);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (24, 4);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (2, 5);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (11, 5);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (18, 5);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (25, 5);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (4, 6);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (8, 6);

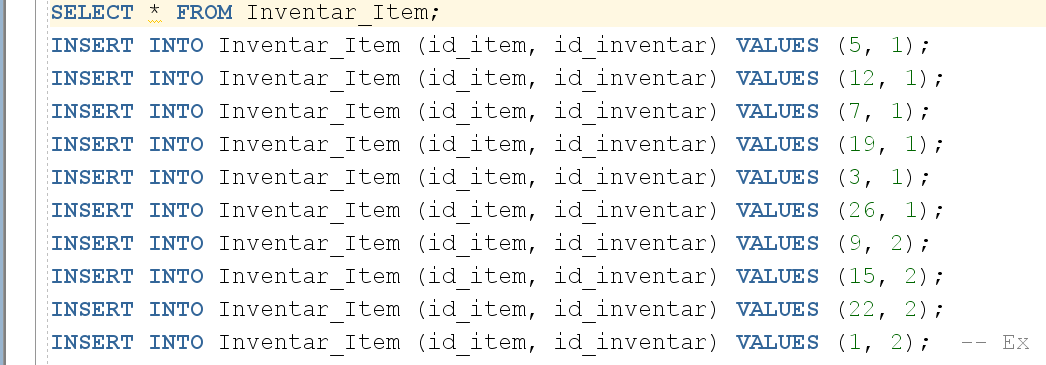
INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (10, 7);

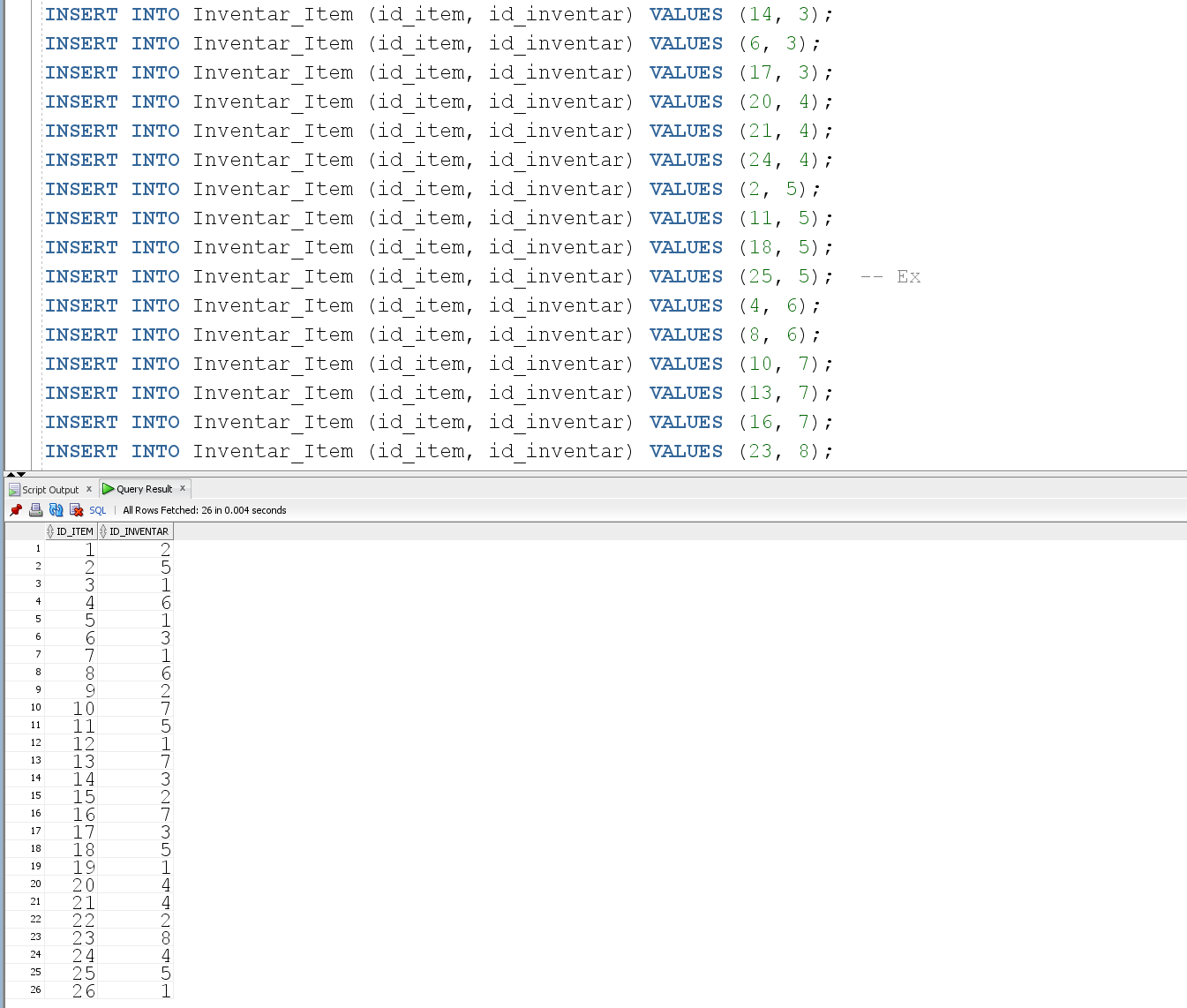
INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (13, 7);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (16, 7);

INSERT INTO Inventar\_Item (id\_item, id\_inventar) VALUES (23, 8);

SELECT \* FROM Inventar\_Item;





## TRANZACTIE

* CREATE

CREATE TABLE Tranzactie (

id\_tranzactie NUMBER(5) PRIMARY KEY,

id\_utilizator NUMBER(5) NOT NULL,

metoda\_tranzactie VARCHAR2(25) NOT NULL,

suma\_tranzactie NUMBER(5) NOT NULL CHECK (suma\_tranzactie >= 0),

data\_tranzactie DATE NOT NULL,

CONSTRAINT fk\_depozit\_utilizator FOREIGN KEY (id\_utilizator) REFERENCES Utilizator(id\_utilizator)

);

DESCRIBE Tranzactie;

* INSERT

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (1, 1, 'Card de credit', 100, TO\_DATE('2023-09-15', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (2, 1, 'PayPal', 50, TO\_DATE('2023-08-21', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (3, 2, 'Card de debit', 25, TO\_DATE('2023-07-30', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (4, 2, 'Transfer bancar', 200, TO\_DATE('2023-09-05', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (5, 3, 'Card de credit', 45, TO\_DATE('2023-10-11', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (6, 4, 'Card de debit', 150, TO\_DATE('2023-08-25', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (7, 5, 'PayPal', 30, TO\_DATE('2023-09-20', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (8, 6, 'Transfer bancar', 75, TO\_DATE('2023-10-02', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (9, 1, 'Card de credit', 120, TO\_DATE('2023-11-01', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (10, 1, 'PayPal', 60, TO\_DATE('2023-11-15', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (11, 2, 'Card de debit', 35, TO\_DATE('2023-12-05', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (12, 2, 'Transfer bancar', 210, TO\_DATE('2023-12-20', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (13, 3, 'Card de credit', 50, TO\_DATE('2023-11-10', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (14, 4, 'Card de debit', 160, TO\_DATE('2023-11-25', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

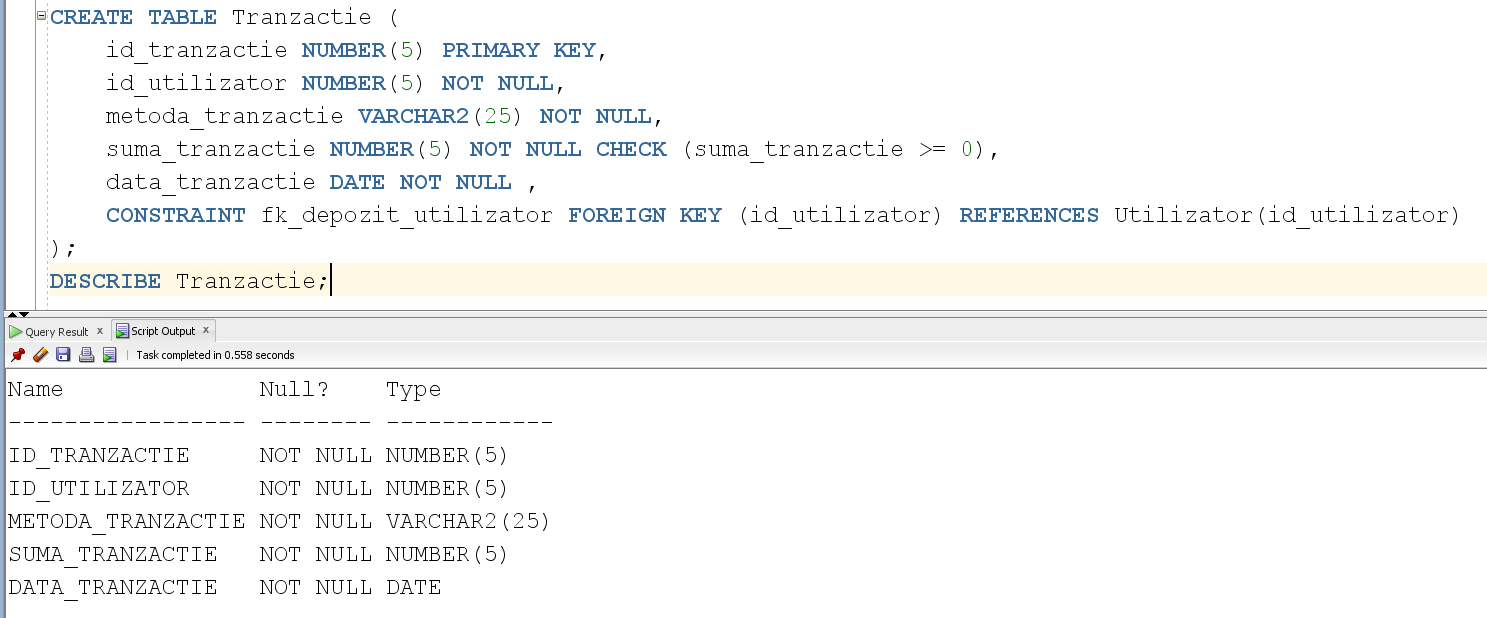
VALUES (15, 5, 'PayPal', 40, TO\_DATE('2023-12-01', 'YYYY-MM-DD'));

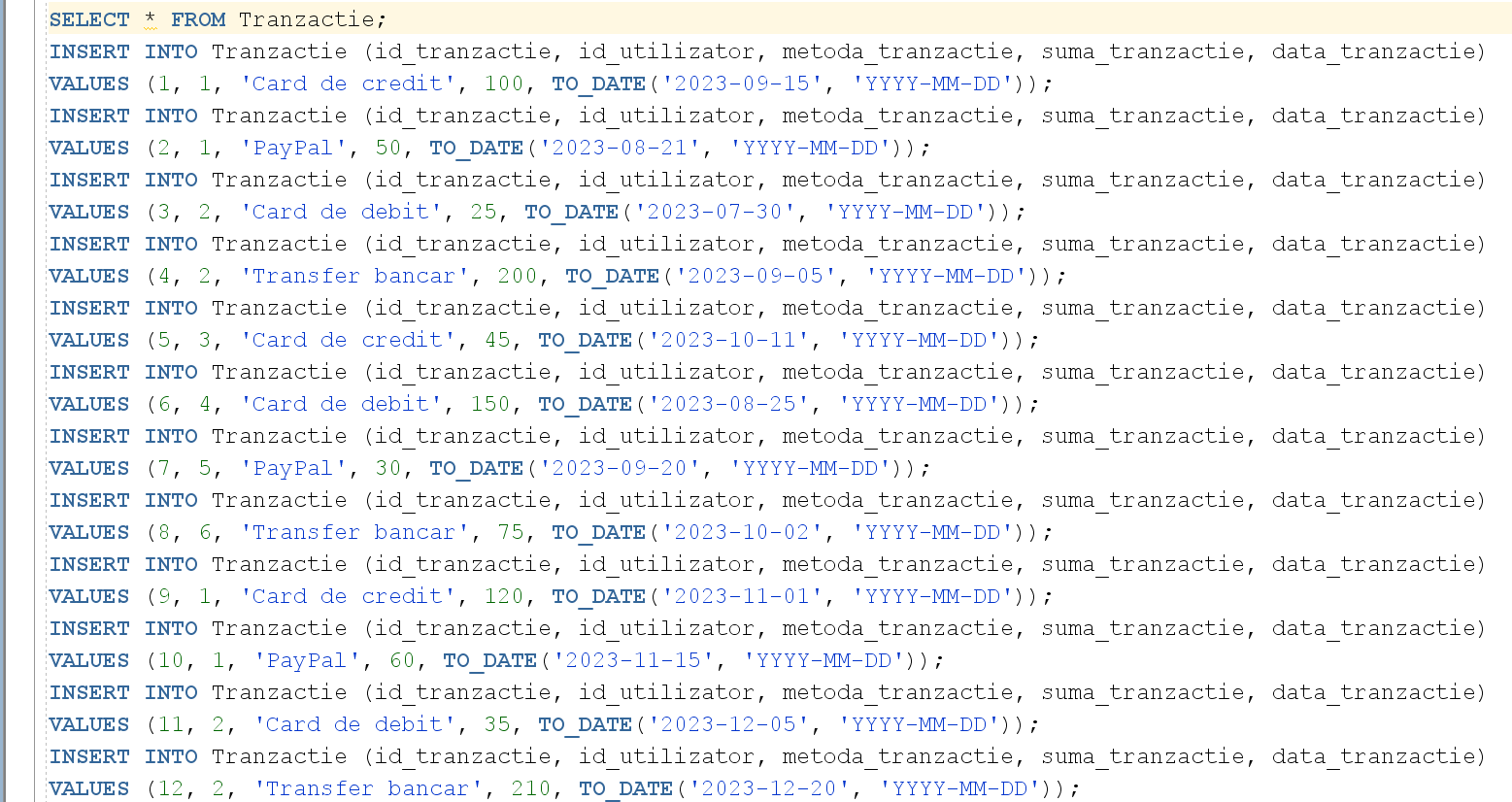
INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

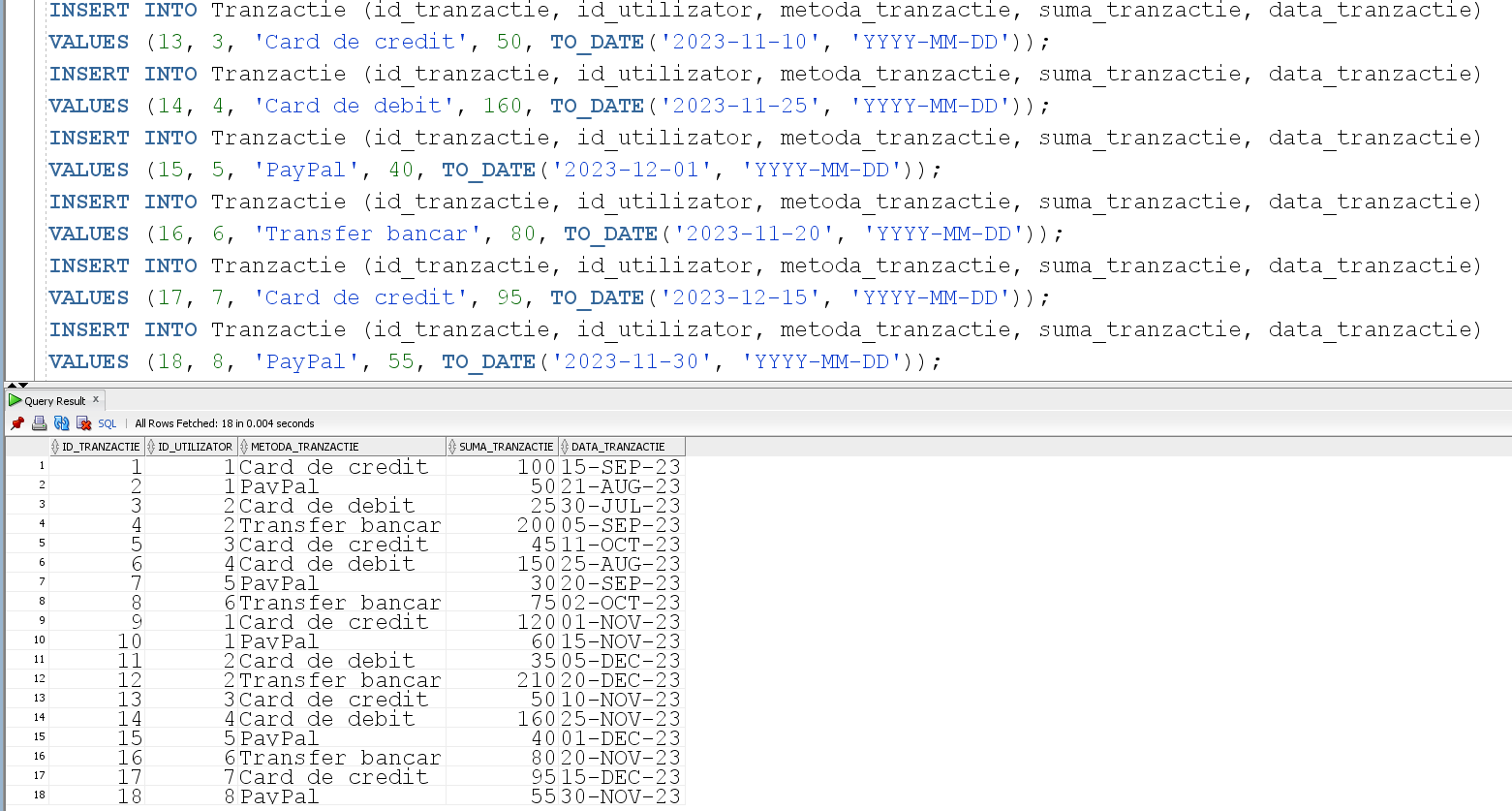
VALUES (16, 6, 'Transfer bancar', 80, TO\_DATE('2023-11-20', 'YYYY-MM-DD'));

INSERT INTO Tranzactie (id\_tranzactie, id\_utilizator, metoda\_tranzactie, suma\_tranzactie, data\_tranzactie)

VALUES (17, 7, 'Card de credit', 95, TO\_DATE('2023-12-15', 'YYYY-MM-DD'));







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

-- Sa se afiseze utilizatorii cu nr de notificari peste media numarului de notificari.

CREATE OR REPLACE PROCEDURE notif\_peste\_medie IS

-- stocam numarul de notificari pentru fiecare utilizator

TYPE tablou\_indexat IS

TABLE OF NUMBER INDEX BY PLS\_INTEGER;

numar\_notificari tablou\_indexat;

-- stocam notificarile

TYPE tablou\_imbricat IS

TABLE OF VARCHAR2(256);

notificari tablou\_imbricat := tablou\_imbricat();

-- stocam utilizatorii cu nr de notificari peste medie

TYPE vector\_utilizatori IS

VARRAY(100) OF VARCHAR2(50);

utilizatori\_peste\_medie vector\_utilizatori := vector\_utilizatori();

v\_numar\_utilizatori PLS\_INTEGER := 0;

v\_total\_notificari NUMBER := 0;

v\_media NUMBER;

BEGIN

-- numarul de notificari pentru fiecare utilizator

FOR rec IN (

SELECT

id\_utilizator,

COUNT(\*) AS numar\_notificari

FROM

notificare

GROUP BY

id\_utilizator

) LOOP

v\_numar\_utilizatori := v\_numar\_utilizatori + 1;

numar\_notificari(v\_numar\_utilizatori) := rec.numar\_notificari;

v\_total\_notificari := v\_total\_notificari + rec.numar\_notificari;

END LOOP;

-- media notificarilor

IF v\_numar\_utilizatori > 0 THEN

v\_media := v\_total\_notificari / v\_numar\_utilizatori;

ELSE

v\_media := 0; -- Evitam diviziunea cu zero

END IF;

-- adaugam toate notificarile in tabloul imbricat

FOR rec IN (

SELECT

text\_notificare

FROM

notificare

) LOOP

notificari.extend;

notificari(notificari.last) := rec.text\_notificare;

END LOOP;

dbms\_output.put\_line('Nr de notificari in medie: ' || v\_media);

-- stocam utilizatorii cu nr de notificari peste medie

dbms\_output.put\_line('Utilizatorii cu numar peste medie de notificari:');

FOR rec IN (

SELECT

u.nume\_utilizator,

COUNT(n.id\_notificare) AS numar\_notificari

FROM

utilizator u

JOIN notificare n ON u.id\_utilizator = n.id\_utilizator

GROUP BY

u.nume\_utilizator

HAVING

COUNT(n.id\_notificare) > v\_media

) LOOP

utilizatori\_peste\_medie.extend;

utilizatori\_peste\_medie(utilizatori\_peste\_medie.last) := rec.nume\_utilizator;

dbms\_output.put\_line(rec.nume\_utilizator

|| ' - '

|| rec.numar\_notificari);

END LOOP;

EXCEPTION

WHEN OTHERS THEN

dbms\_output.put\_line('A aparut o eroare: ' || sqlerrm);

END notif\_peste\_medie;

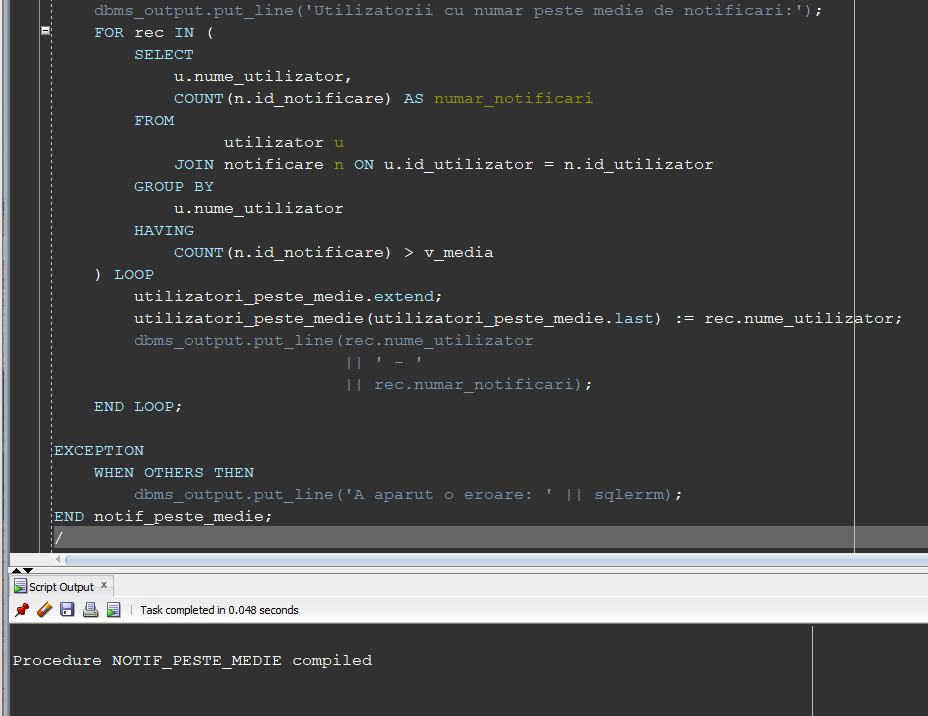
/

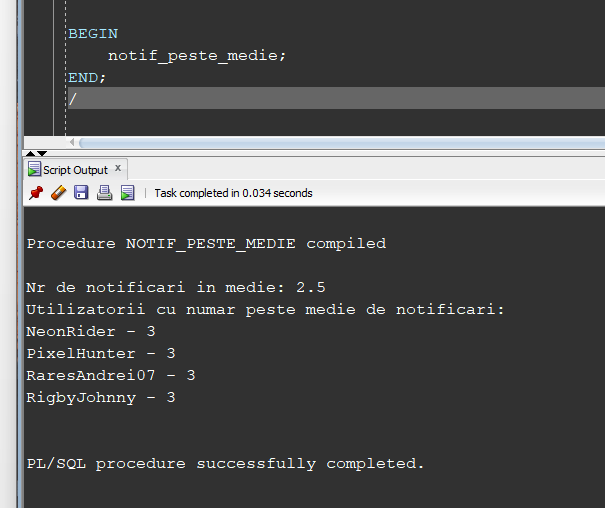
BEGIN

notif\_peste\_medie;

END;

/





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

-- Afiseaza utilizatorii si numarul de recenzii scrise de fiecare, specificand daca recenzia este pozitiva sau negativa, in functie de continutul textului recenziei.

CREATE OR REPLACE PROCEDURE gestionare\_recenzii IS

-- pentru numele utilizatorului si numarul de recenzii

CURSOR c\_utilizatori IS

SELECT

u.id\_utilizator,

u.nume\_utilizator,

COUNT(r.id\_recenzie) AS numar\_recenzii

FROM

utilizator u

LEFT JOIN recenzie r ON u.id\_utilizator = r.id\_utilizator

GROUP BY

u.id\_utilizator,

u.nume\_utilizator;

-- pentru detaliile despre recenzii

CURSOR c\_recenzii (

p\_id\_utilizator NUMBER

) IS

SELECT

j.nume\_joc,

r.text\_recenzie

FROM

recenzie r

JOIN joc j ON r.id\_joc = j.id\_joc

WHERE

r.id\_utilizator = p\_id\_utilizator;

v\_id\_utilizator utilizator.id\_utilizator%TYPE;

v\_nume\_utilizator utilizator.nume\_utilizator%TYPE;

v\_numar\_recenzii NUMBER;

v\_nume\_joc joc.nume\_joc%TYPE;

v\_text\_recenzie recenzie.text\_recenzie%TYPE;

BEGIN

OPEN c\_utilizatori;

LOOP

FETCH c\_utilizatori INTO

v\_id\_utilizator,

v\_nume\_utilizator,

v\_numar\_recenzii;

EXIT WHEN c\_utilizatori%notfound;

IF v\_numar\_recenzii = 1 THEN

dbms\_output.put\_line('Utilizatorul '

|| v\_nume\_utilizator

|| ' a scris o recenzie.');

ELSE

dbms\_output.put\_line('Utilizatorul '

|| v\_nume\_utilizator

|| ' a scris '

|| v\_numar\_recenzii

|| ' recenzii.');

END IF;

OPEN c\_recenzii(v\_id\_utilizator);

LOOP

FETCH c\_recenzii INTO

v\_nume\_joc,

v\_text\_recenzie;

EXIT WHEN c\_recenzii%notfound;

-- verificam textul recenziei

IF instr(v\_text\_recenzie, 'POZITIV') > 0 THEN

dbms\_output.put\_line('Joc: '

|| v\_nume\_joc

|| ', Recenzie: Pozitiva');

ELSIF instr(v\_text\_recenzie, 'NEGATIV') > 0 THEN

dbms\_output.put\_line('Joc: '

|| v\_nume\_joc

|| ', Recenzie: Negativa');

ELSE

dbms\_output.put\_line('Joc: '

|| v\_nume\_joc

|| ', Recenzie: '

|| v\_text\_recenzie);

END IF;

END LOOP;

CLOSE c\_recenzii;

END LOOP;

CLOSE c\_utilizatori;

END;

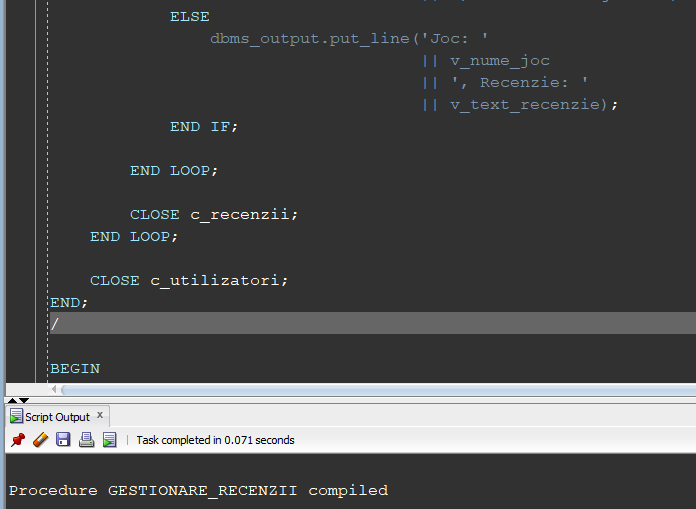
/

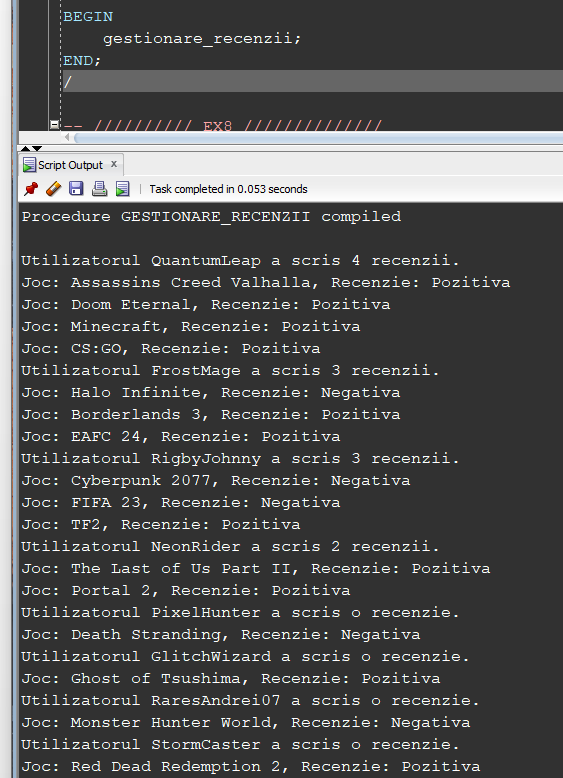
BEGIN

gestionare\_recenzii;

END;

/





## 8. Subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele create.

-- Sa se afiseze pentru un utilizator dat daca suma tranzactiilor sale

-- este mai mare decat media tranzactiilor efectuate de toti utilizatorii.

-- Utilizatorul trebuie sa aiba domiciliu intr-o tara cu salariul mediu peste

-- 20000, iar acesta sa nu aiba mai mult de 3 tranzactii.

CREATE OR REPLACE FUNCTION verifica\_tranzactii\_utilizator (

p\_id\_utilizator NUMBER

) RETURN VARCHAR2 IS

v\_suma\_tranzactii NUMBER;

v\_media\_tranzactii NUMBER;

v\_salariu\_mediu NUMBER;

v\_numar\_tranzactii NUMBER;

v\_exista\_utilizator NUMBER;

BEGIN

SELECT

COUNT(\*)

INTO v\_exista\_utilizator

FROM

utilizator

WHERE

id\_utilizator = p\_id\_utilizator;

IF v\_exista\_utilizator = 0 THEN

raise\_application\_error(-20001, 'Utilizatorul nu exista.');

END IF;

-- verificam numarul de tranzactii ale utilizatorului

SELECT

COUNT(\*)

INTO v\_numar\_tranzactii

FROM

tranzactie

WHERE

id\_utilizator = p\_id\_utilizator;

IF v\_numar\_tranzactii = 0 THEN

RAISE no\_data\_found; -- utilizatorul nu are tranzactii

ELSIF v\_numar\_tranzactii > 3 THEN

RAISE too\_many\_rows; -- are mai mult de 3 tranzactii

END IF;

-- suma tranzactiilor utilizatorului

SELECT

SUM(suma\_tranzactie)

INTO v\_suma\_tranzactii

FROM

tranzactie

WHERE

id\_utilizator = p\_id\_utilizator;

-- media tranzactiilor tuturor utilizatorilor

SELECT

AVG(suma\_tranzactie)

INTO v\_media\_tranzactii

FROM

tranzactie;

-- salariul mediu din tara utilizatorului

SELECT

t.salariu\_mediu

INTO v\_salariu\_mediu

FROM

utilizator u

JOIN tara t ON u.id\_tara = t.id\_tara

WHERE

u.id\_utilizator = p\_id\_utilizator;

IF v\_salariu\_mediu <= 20000 THEN

raise\_application\_error(-20002, 'Salariul mediu din tara utilizatorului este sub 20000.');

END IF;

IF v\_suma\_tranzactii > v\_media\_tranzactii THEN

RETURN 'Suma tranzactiilor utilizatorului este peste media tuturor tranzactiilor ('

|| v\_suma\_tranzactii

|| ' > '

|| round(v\_media\_tranzactii, 2)

|| ').';

ELSE

RETURN 'Suma tranzactiilor utilizatorului este sub media tuturor tranzactiilor ('

|| v\_suma\_tranzactii

|| ' < '

|| round(v\_media\_tranzactii, 2)

|| ').';

END IF;

EXCEPTION

WHEN no\_data\_found THEN

RETURN 'Utilizatorul nu are tranzactii.';

WHEN too\_many\_rows THEN

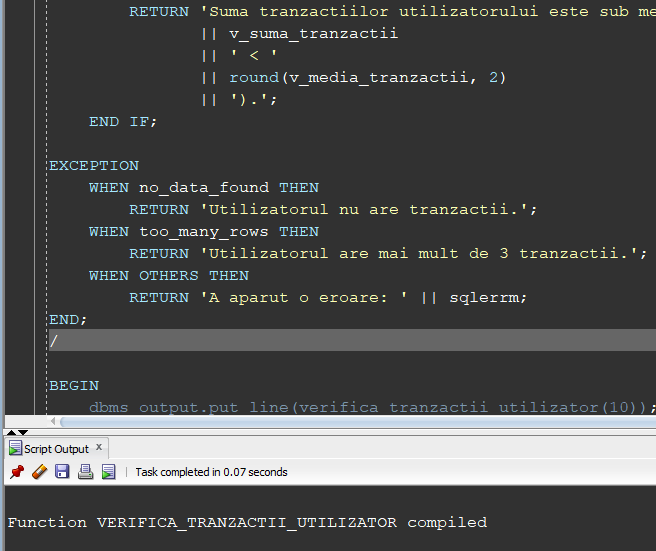
RETURN 'Utilizatorul are mai mult de 3 tranzactii.';

WHEN OTHERS THEN

RETURN 'A aparut o eroare: ' || sqlerrm;

END;

/



--cream un utilizator fara tranzactii

INSERT INTO utilizator (

id\_utilizator,

nume\_utilizator,

parola,

email,

varsta,

id\_tara

) VALUES (

9,

'MonsterSlayer',

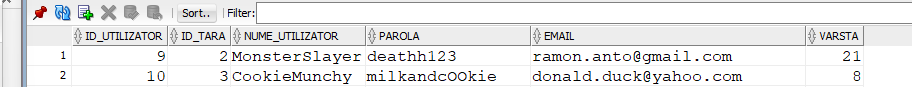
'deathh123',

'ramon.anto@gmail.com',

21,

2

);

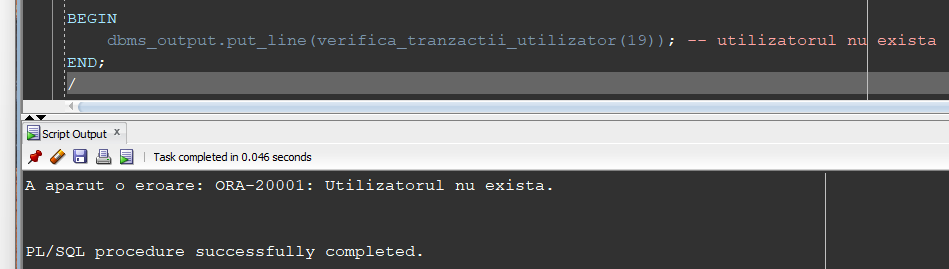


BEGIN

dbms\_output.put\_line(verifica\_tranzactii\_utilizator(19)); -- utilizatorul nu exista

END;

/

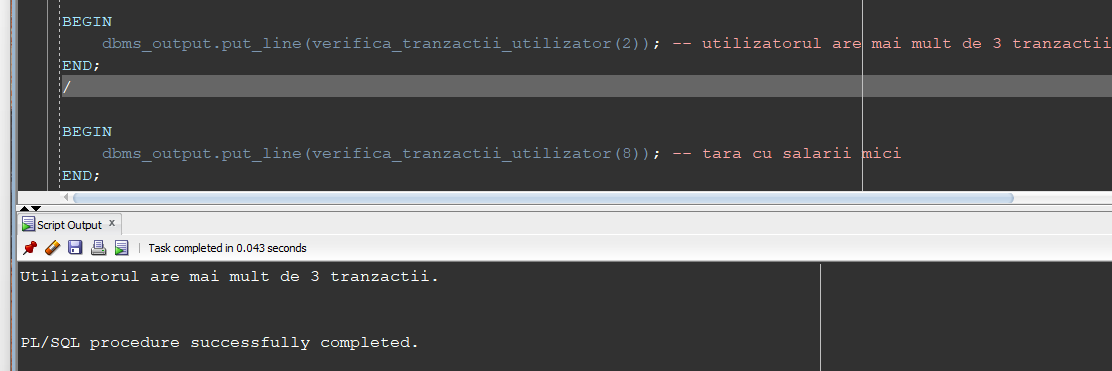


BEGIN

dbms\_output.put\_line(verifica\_tranzactii\_utilizator(2)); -- utilizatorul are mai mult de 3 tranzactii

END;

/

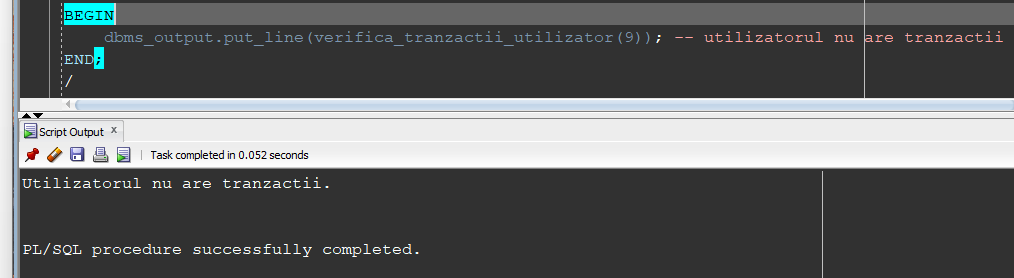


BEGIN

dbms\_output.put\_line(verifica\_tranzactii\_utilizator(9)); -- utilizatorul nu are tranzactii

END;

/

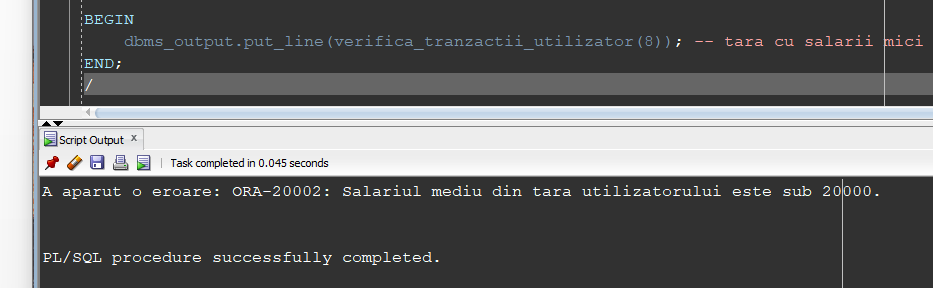


BEGIN

dbms\_output.put\_line(verifica\_tranzactii\_utilizator(8)); -- tara cu salarii mici

END;

/



BEGIN

dbms\_output.put\_line(verifica\_tranzactii\_utilizator(3)); -- peste medie

END;

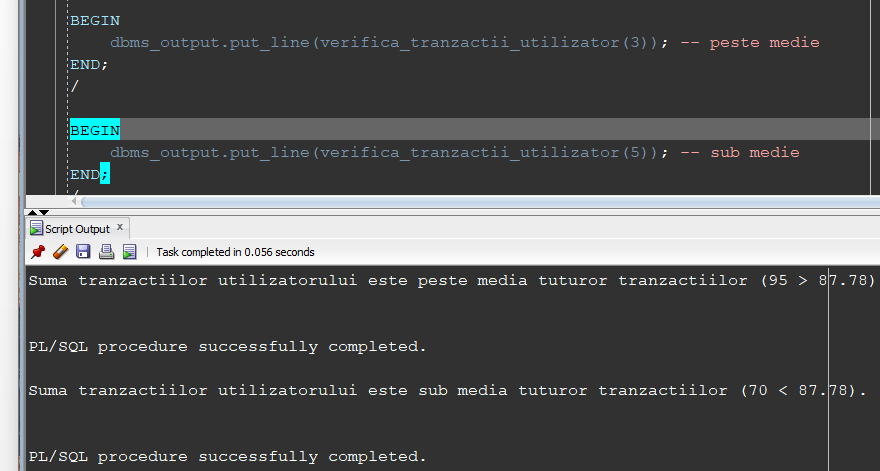
/

BEGIN

dbms\_output.put\_line(verifica\_tranzactii\_utilizator(5)); -- sub medie

END;

/



## 9. Subprogram stocat independent de tip procedură care să aibă minim 2 parametri și să utilizeze într-o singură comandă SQL 5 dintre tabelele create

-- scrie o procedura care permite adaugarea unui item

-- in inventarul unui utilizator (id-uri date ca parametrii)

-- 1. verifica daca utilizatorul si itemul exista

-- 2. verifica capacitatea inventarului utilizatorului si daca

-- este posibila adaugarea unui item

-- 3. verifica varsta utilizatorului si compara cu varsta jocului

-- din care face parte itemul

-- scrie o procedura care permite adaugarea unui item

-- in inventarul unui utilizator (id-uri date ca parametrii)

-- 1. verifica daca utilizatorul si itemul exista

-- 2. verifica capacitatea inventarului utilizatorului si daca

-- este posibila adaugarea unui item

-- 3. verifica varsta utilizatorului si compara cu varsta jocului

-- din care face parte itemul

CREATE OR REPLACE PROCEDURE adauga\_item\_in\_inventar (

p\_id\_utilizator NUMBER,

p\_id\_item NUMBER

) IS

v\_capacitate NUMBER;

v\_numar\_iteme NUMBER;

v\_exista\_item NUMBER;

v\_varsta\_utilizator NUMBER;

v\_varsta\_recomandata NUMBER;

e\_capacitate\_depasita EXCEPTION;

e\_item\_inexistent EXCEPTION;

e\_varsta\_recomandata EXCEPTION;

e\_utilizator\_inexistent EXCEPTION;

BEGIN

-- utilizatorul exista

SELECT

COUNT(\*)

INTO v\_exista\_item

FROM

utilizator

WHERE

id\_utilizator = p\_id\_utilizator;

IF v\_exista\_item = 0 THEN

raise e\_utilizator\_inexistent;

END IF;

-- itemul exista

SELECT

COUNT(\*)

INTO v\_exista\_item

FROM

item

WHERE

id\_item = p\_id\_item;

IF v\_exista\_item = 0 THEN

raise e\_item\_inexistent;

END IF;

-- capacitatea inventarului utilizatorului

SELECT

capacitate

INTO v\_capacitate

FROM

inventar

WHERE

id\_utilizator = p\_id\_utilizator;

-- verificam numarul de iteme din inventar

SELECT

COUNT(\*)

INTO v\_numar\_iteme

FROM

inventar\_item

WHERE

id\_inventar = (

SELECT

id\_inventar

FROM

inventar

WHERE

id\_utilizator = p\_id\_utilizator

);

IF v\_numar\_iteme >= v\_capacitate THEN

raise e\_capacitate\_depasita;

END IF;

-- varsta recomandata a jocului asociat itemului

SELECT

j.varsta\_recomandata

INTO v\_varsta\_recomandata

FROM

item i

JOIN joc j ON i.id\_joc = j.id\_joc

WHERE

i.id\_item = p\_id\_item;

SELECT

varsta

INTO v\_varsta\_utilizator

FROM

utilizator

WHERE

id\_utilizator = p\_id\_utilizator;

IF v\_varsta\_recomandata > v\_varsta\_utilizator THEN

raise e\_varsta\_recomandata;

END IF;

-- adaugam itemul in inventar

INSERT INTO inventar\_item (

id\_item,

id\_inventar

) VALUES (

p\_id\_item,

(

SELECT

id\_inventar

FROM

inventar

WHERE

id\_utilizator = p\_id\_utilizator

)

);

dbms\_output.put\_line('Itemul a fost adaugat cu succes in inventar.');

EXCEPTION

WHEN e\_capacitate\_depasita THEN

dbms\_output.put\_line('Capacitatea inventarului a fost depasita.');

WHEN e\_item\_inexistent THEN

dbms\_output.put\_line('Itemul nu exista.');

WHEN e\_varsta\_recomandata THEN

dbms\_output.put\_line('Itemul nu poate fi adaugat (varsta utilizatorului prea mica).');

WHEN e\_utilizator\_inexistent THEN

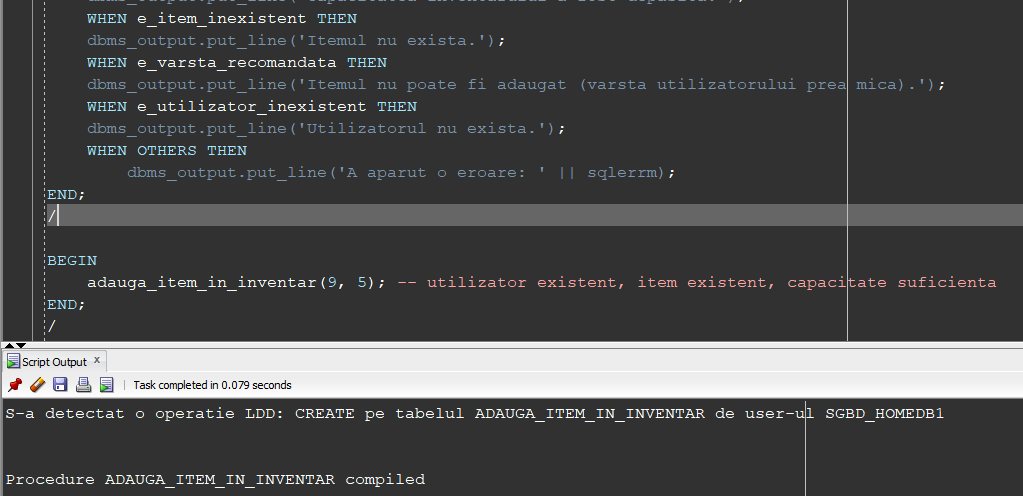
dbms\_output.put\_line('Utilizatorul nu exista.');

WHEN OTHERS THEN

dbms\_output.put\_line('A aparut o eroare: ' || sqlerrm);

END;

/



-- pentru utilizatorul cu id 9 cream un inventar cu capacitate mare

-- (pe acesta il vom folosi pentru testarea procedurii)

INSERT INTO inventar (

id\_inventar,

id\_utilizator,

capacitate

) VALUES (

9,

9,

99

);



-- cream un utilizator cu o varsta mai mica

-- decat varsta recomandata de majoritatea jocurilor

INSERT INTO utilizator (

id\_utilizator,

nume\_utilizator,

parola,

email,

varsta,

id\_tara

) VALUES (

10,

'CookieMunchy',

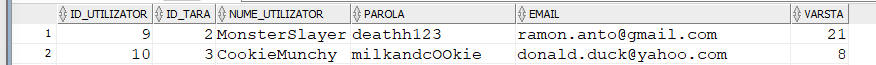
'milkandcOOkie',

'donald.duck@yahoo.com',

8,

3

);



-- cream inventarul utilizatorului nou adaugat

INSERT INTO inventar (

id\_inventar,

id\_utilizator,

capacitate

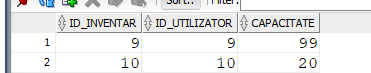
) VALUES (

10,

10,

20

);

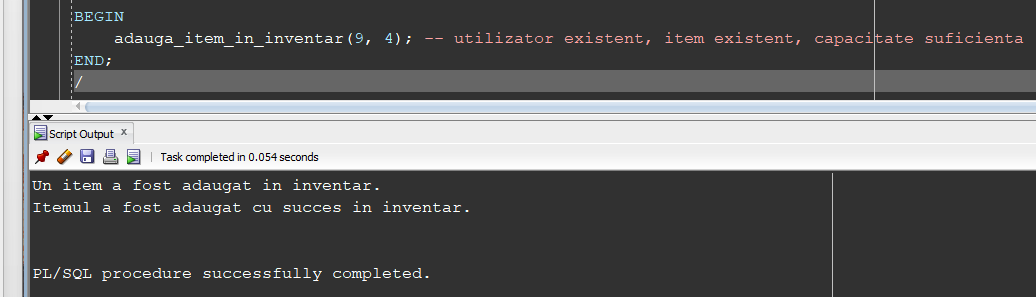


BEGIN

adauga\_item\_in\_inventar(9, 4); -- utilizator existent, item existent, capacitate suficienta

END;

/



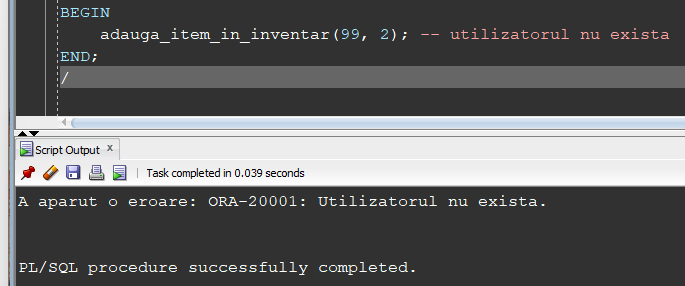


BEGIN

adauga\_item\_in\_inventar(99, 2); -- utilizatorul nu exista

END;

/

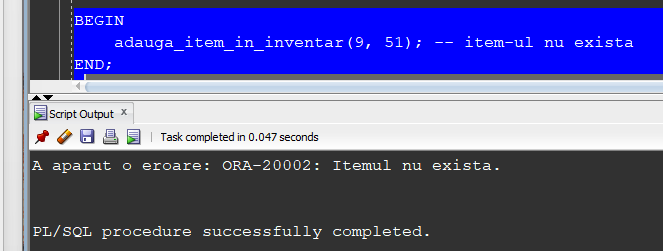


BEGIN

adauga\_item\_in\_inventar(9, 51); -- item-ul nu exista

END;

/

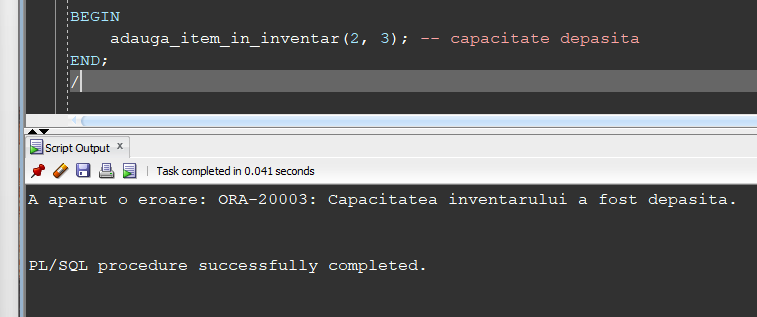


BEGIN

adauga\_item\_in\_inventar(2, 3); -- capacitate depasita

END;

/



De la inceput am creat utilizatori cu inventare care depasesc limita.

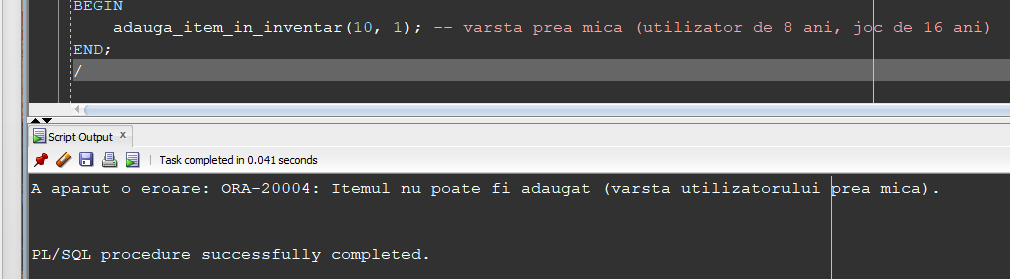


BEGIN

adauga\_item\_in\_inventar(10, 1); -- varsta prea mica (utilizator de 8 ani, joc de 16 ani)

END;

/



## 10. Definiți un trigger de tip LMD la nivel de comandă

-- cream o tabela audit care va stoca toate operatiile LMD

CREATE TABLE log\_inventare (

id\_log NUMBER

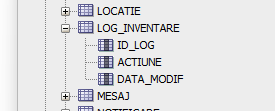
GENERATED BY DEFAULT AS IDENTITY

PRIMARY KEY,

actiune VARCHAR2(10),

data\_modif TIMESTAMP DEFAULT current\_timestamp

);



-- scrieti un trigger care se declanseaza la fiecare adaugare de item in inventar.

CREATE OR REPLACE TRIGGER trg\_lmd\_comanda AFTER

INSERT OR UPDATE OR DELETE ON inventar\_item

DECLARE

actiune VARCHAR2(20);

BEGIN

IF inserting THEN

actiune := 'insert';

dbms\_output.put\_line('Un item a fost adaugat in inventar.');

ELSIF updating THEN

actiune := 'update';

dbms\_output.put\_line('Un item din inventar a fost modificat.');

ELSIF deleting THEN

actiune := 'delete';

dbms\_output.put\_line('Un item a fost sters din inventar.');

END IF;

-- ne folosim de log pentru a inregistra actiunea

INSERT INTO log\_inventare (

actiune,

data\_modif

) VALUES (

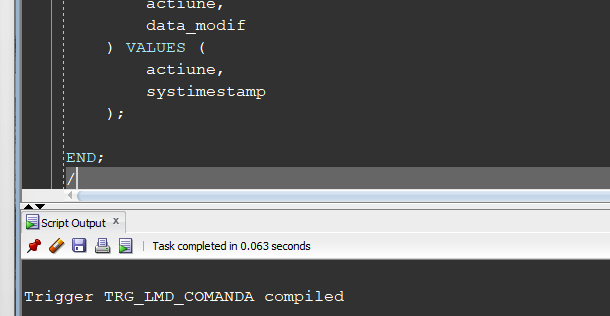
actiune,

systimestamp

);

END;

/



--inserare

INSERT INTO inventar\_item (

id\_item,

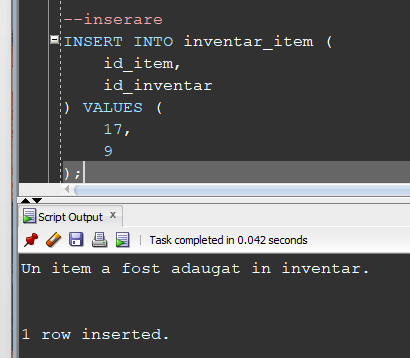
id\_inventar

) VALUES (

17,

9

);





--update

UPDATE inventar\_item

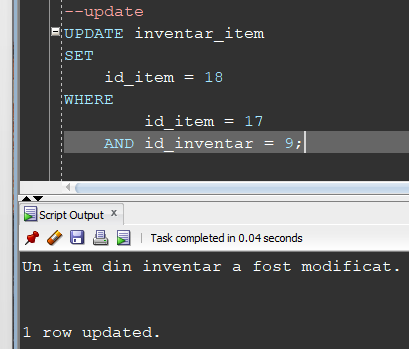
SET

id\_item = 16

WHERE

id\_item = 15

AND id\_inventar = 9;





--delete

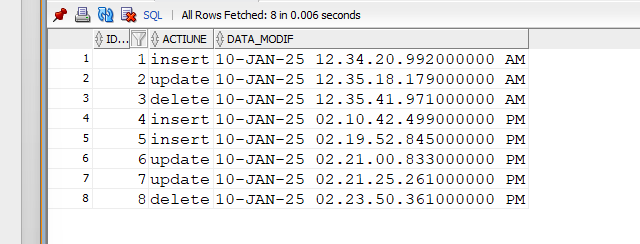
DELETE FROM inventar\_item

WHERE

id\_item = 16

AND id\_inventar = 9;





## 11. Definiți un trigger de tip LMD la nivel de linie.

-- scrieti un trigger care se declanseaza la inserarea in

-- tabelul Developer si care verifica conditiile:

-- 1. studioul sa nu existe deja

-- 2. studioul sa nu fi fost lansat in ultimii 4 ani

-- 3. locatia studioului sa fie valida

CREATE OR REPLACE TRIGGER trg\_verifica\_developer BEFORE

INSERT OR UPDATE ON developer

FOR EACH ROW

DECLARE

v\_count\_dev NUMBER;

v\_count\_locatie NUMBER;

BEGIN

-- verif daca studioul deja exista

SELECT

COUNT(\*)

INTO v\_count\_dev

FROM

developer

WHERE

nume\_developer = :new.nume\_developer;

IF v\_count\_dev > 0 THEN

raise\_application\_error(-20001, 'Studio-ul exista deja in tabela.');

END IF;

-- cel putin 4 ani

IF months\_between(sysdate, :new.data\_lansare\_studio) < 48 THEN

raise\_application\_error(-20002, 'Studio-ul trebuie sa fie lansat de cel putin 4 ani.');

END IF;

-- locatie valida

SELECT

COUNT(\*)

INTO v\_count\_locatie

FROM

locatie

WHERE

id\_locatie = :new.id\_locatie;

IF v\_count\_locatie = 0 THEN

raise\_application\_error(-20003, 'Locatia asociata nu exista in tabela locatie.');

END IF;

END;

/



-- studio lansat in ultimii 4 ani

INSERT INTO developer (

id\_developer,

id\_locatie,

nume\_developer,

data\_lansare\_studio,

salariu\_angajati

) VALUES (

13,

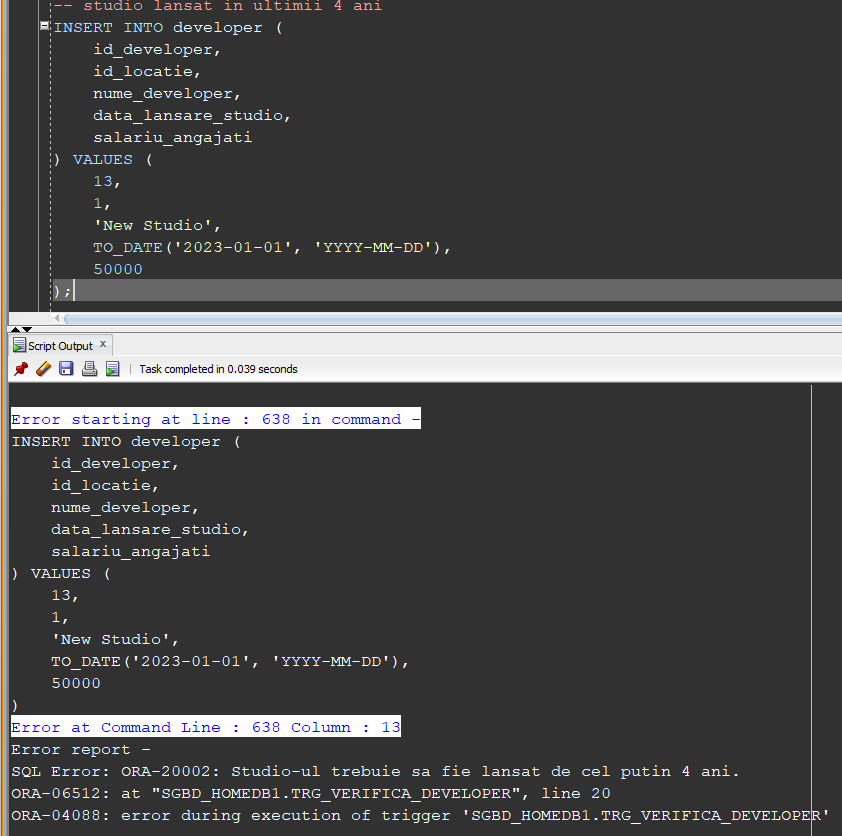
1,

'New Studio',

TO\_DATE('2023-01-01', 'YYYY-MM-DD'),

50000

);



-- studio deja existent

INSERT INTO developer (

id\_developer,

id\_locatie,

nume\_developer,

data\_lansare\_studio,

salariu\_angajati

) VALUES (

14,

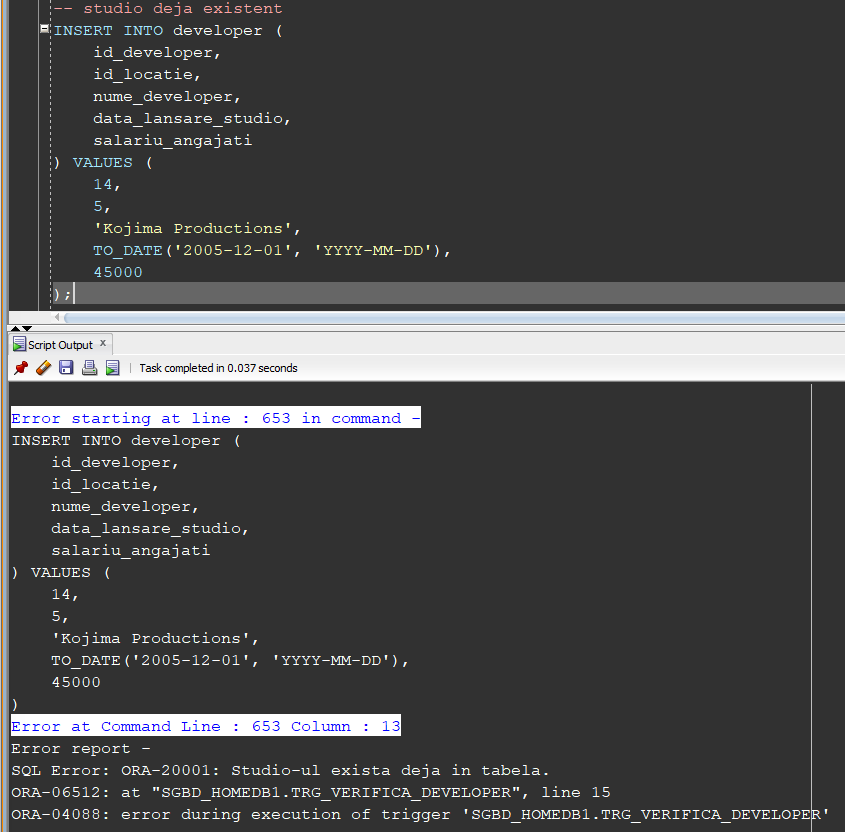
5,

'Kojima Productions',

TO\_DATE('2005-12-01', 'YYYY-MM-DD'),

45000

);



-- locatia nu exista

INSERT INTO developer (

id\_developer,

id\_locatie,

nume\_developer,

data\_lansare\_studio,

salariu\_angajati

) VALUES (

15,

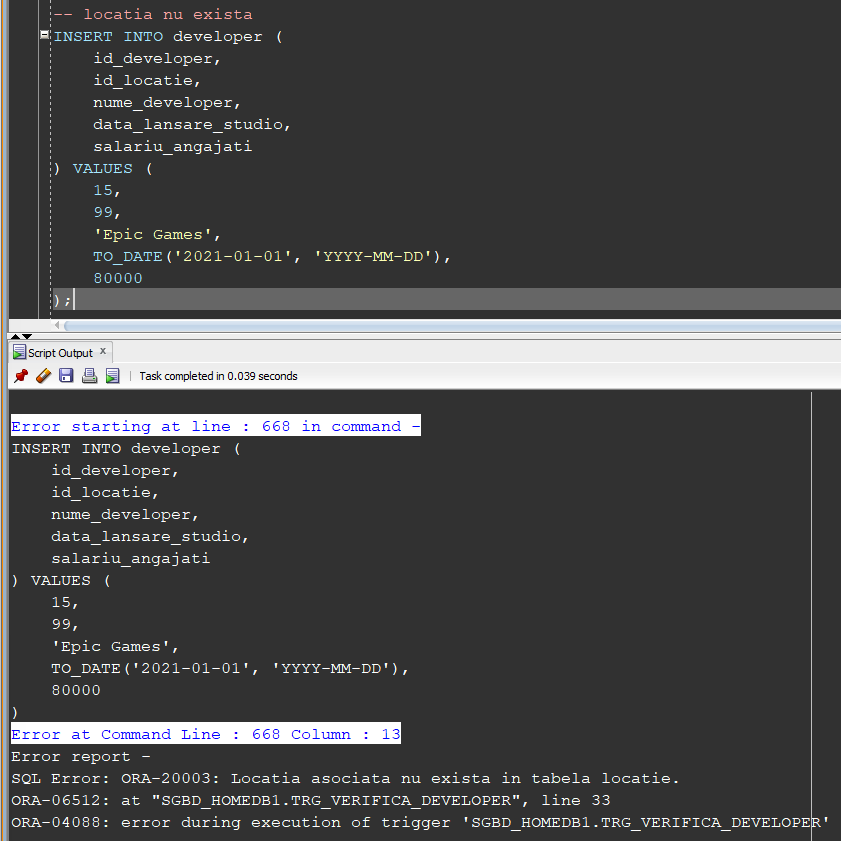
99,

'Epic Games',

TO\_DATE('2021-01-01', 'YYYY-MM-DD'),

80000

);



-- insert valid

INSERT INTO developer (

id\_developer,

id\_locatie,

nume\_developer,

data\_lansare\_studio,

salariu\_angajati

) VALUES (

20,

6,

'Riot',

TO\_DATE('2002-01-01', 'YYYY-MM-DD'),

70000

);





## 12. Definiți un trigger de tip LDD.

--cream un tabel de care se va folosi trigger-ul pentru log-uri

CREATE TABLE log\_ldd (

id\_log NUMBER

GENERATED BY DEFAULT AS IDENTITY

PRIMARY KEY,

username VARCHAR2(50),

actiune VARCHAR2(50),

tabel\_modif VARCHAR2(50),

data\_modif DATE

);

Scrieti un trigger care stocheaza intr-o tabela de log-uri modificarile aduse de utilizator bazei de date.

CREATE OR REPLACE TRIGGER trg\_ldd AFTER CREATE OR ALTER OR DROP ON SCHEMA DECLARE

v\_user VARCHAR2(50);

v\_tabel VARCHAR2(100);

v\_ldd VARCHAR2(50);

BEGIN

v\_user := sys\_context('USERENV', 'SESSION\_USER');

v\_tabel := ora\_dict\_obj\_name;

v\_ldd := ora\_sysevent;

INSERT INTO log\_ldd (

username,

actiune,

tabel\_modif,

data\_modif

) VALUES (

v\_user,

v\_ldd,

v\_tabel,

systimestamp

);

dbms\_output.put\_line('S-a detectat o operatie LDD: '

|| v\_ldd

|| ' pe tabelul '

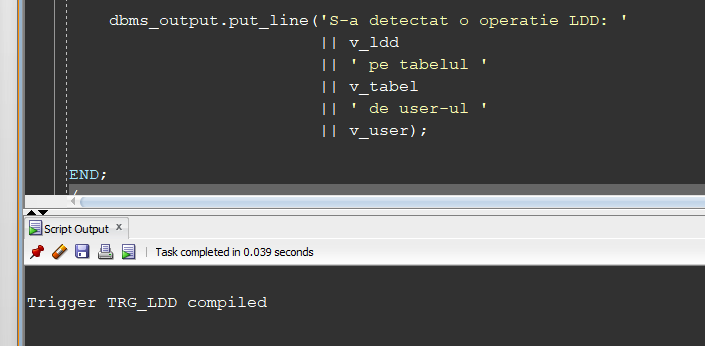
|| v\_tabel

|| ' de user-ul '

|| v\_user);

END;

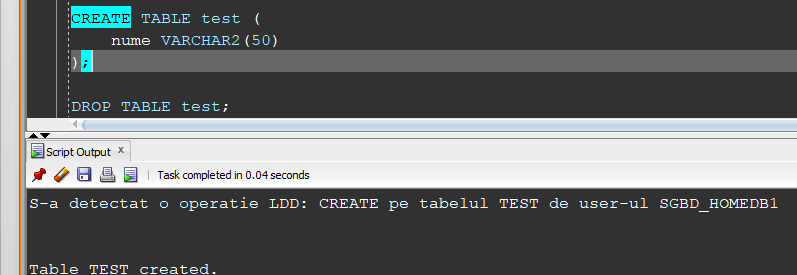
/



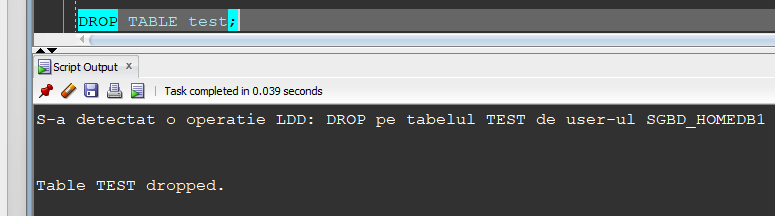
CREATE TABLE test (

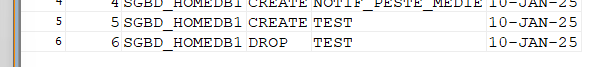
nume VARCHAR2(50)

);



DROP TABLE test;





## 13. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un pachet

-- Creati un package care afiseaza itemele unui user,

-- jocurile unui dezvoltator dat,

-- numarul de iteme ale utilizatorului care au valoare mai mare decat un pret dat

-- si jocurile dezvoltate in tara utilizatorului.

CREATE OR REPLACE PACKAGE steam\_pack AS

-- RECORD pentru jocuri

TYPE joc\_record IS RECORD (

id\_joc NUMBER(5),

id\_developer NUMBER(5),

nume\_joc VARCHAR2(50),

pret\_joc NUMBER(5),

data\_lansare\_joc DATE,

varsta\_recomandata NUMBER(3),

rating NUMBER(1)

);

-- RECORD pentru jocuri din tara utilizatorului

TYPE jocuri\_natale IS RECORD (

id\_joc NUMBER(5),

nume\_joc VARCHAR2(50),

nume\_developer VARCHAR2(50),

tara VARCHAR2(50)

);

TYPE jocuri\_natale\_table IS

TABLE OF jocuri\_natale INDEX BY PLS\_INTEGER;

TYPE joc\_table IS

TABLE OF joc\_record INDEX BY PLS\_INTEGER;

PROCEDURE afiseaza\_iteme\_in\_inventar (

p\_id\_utilizator NUMBER

);

PROCEDURE afiseaza\_jocuri (

p\_id\_developer NUMBER

);

FUNCTION calculeaza\_numar\_iteme (

p\_id\_utilizator NUMBER,

p\_valoare NUMBER

) RETURN NUMBER;

FUNCTION afisare\_jocuri\_natale (

p\_id\_utilizator NUMBER

) RETURN jocuri\_natale\_table;

END steam\_pack;

/

CREATE OR REPLACE PACKAGE BODY steam\_pack AS

PROCEDURE afiseaza\_iteme\_in\_inventar (

p\_id\_utilizator NUMBER

) IS

v\_exista\_utilizator NUMBER;

BEGIN

SELECT

COUNT(\*)

INTO v\_exista\_utilizator

FROM

utilizator

WHERE

id\_utilizator = p\_id\_utilizator;

IF v\_exista\_utilizator = 0 THEN

raise\_application\_error(-20001, 'Utilizatorul nu exista.');

END IF;

FOR rec IN (

SELECT

i.id\_item,

i.nume\_item,

i.pret\_item

FROM

inventar\_item ii

JOIN inventar inv ON ii.id\_inventar = inv.id\_inventar

JOIN item i ON ii.id\_item = i.id\_item

WHERE

inv.id\_utilizator = p\_id\_utilizator

) LOOP

IF rec.pret\_item IS NOT NULL THEN

dbms\_output.put\_line('Item ID: '

|| rec.id\_item

|| ', Nume Item: '

|| rec.nume\_item

|| ', Pret: '

|| rec.pret\_item);

ELSE

dbms\_output.put\_line('Item ID: '

|| rec.id\_item

|| ', Nume Item: '

|| rec.nume\_item);

END IF;

END LOOP;

IF SQL%rowcount = 0 THEN

dbms\_output.put\_line('Utilizatorul nu are iteme in inventar.');

END IF;

END afiseaza\_iteme\_in\_inventar;

PROCEDURE afiseaza\_jocuri (

p\_id\_developer NUMBER

) IS

v\_joc joc\_table;

v\_index PLS\_INTEGER := 0;

BEGIN

FOR rec IN (

SELECT

id\_joc,

id\_developer,

nume\_joc,

pret\_joc,

data\_lansare\_joc,

varsta\_recomandata,

rating

FROM

joc

WHERE

id\_developer = p\_id\_developer

) LOOP

v\_index := v\_index + 1;

v\_joc(v\_index).id\_joc := rec.id\_joc;

v\_joc(v\_index).id\_developer := rec.id\_developer;

v\_joc(v\_index).nume\_joc := rec.nume\_joc;

v\_joc(v\_index).pret\_joc := rec.pret\_joc;

v\_joc(v\_index).data\_lansare\_joc := rec.data\_lansare\_joc;

v\_joc(v\_index).varsta\_recomandata := rec.varsta\_recomandata;

v\_joc(v\_index).rating := rec.rating;

END LOOP;

IF v\_index > 0 THEN

FOR i IN 1..v\_index LOOP

dbms\_output.put\_line('ID Joc: '

|| v\_joc(i).id\_joc

|| ', Nume Joc: '

|| v\_joc(i).nume\_joc

|| ', Pret: '

|| v\_joc(i).pret\_joc

|| ', Data Lansare: '

|| to\_char(v\_joc(i).data\_lansare\_joc, 'YYYY-MM-DD')

|| ', Varsta Recomandata: '

|| v\_joc(i).varsta\_recomandata

|| ', Rating: '

|| v\_joc(i).rating);

END LOOP;

ELSE

dbms\_output.put\_line('Nu exista jocuri inregistrate pentru dezvoltatorul cu ID: ' || p\_id\_developer);

END IF;

END afiseaza\_jocuri;

FUNCTION calculeaza\_numar\_iteme (

p\_id\_utilizator NUMBER,

p\_valoare NUMBER

) RETURN NUMBER IS

v\_numar\_iteme NUMBER;

BEGIN

SELECT

COUNT(\*)

INTO v\_numar\_iteme

FROM

inventar\_item ii

JOIN inventar inv ON ii.id\_inventar = inv.id\_inventar

JOIN item i ON ii.id\_item = i.id\_item

WHERE

inv.id\_utilizator = p\_id\_utilizator

AND i.pret\_item > p\_valoare;

RETURN v\_numar\_iteme;

END calculeaza\_numar\_iteme;

FUNCTION afisare\_jocuri\_natale (

p\_id\_utilizator NUMBER

) RETURN jocuri\_natale\_table IS

v\_jocuri jocuri\_natale\_table;

v\_index PLS\_INTEGER := 0;

v\_id\_tara NUMBER;

v\_nume\_tara VARCHAR2(50);

BEGIN

SELECT

id\_tara

INTO v\_id\_tara

FROM

utilizator

WHERE

id\_utilizator = p\_id\_utilizator;

SELECT

nume\_tara

INTO v\_nume\_tara

FROM

tara

WHERE

id\_tara = v\_id\_tara;

FOR rec IN (

SELECT

j.id\_joc,

j.nume\_joc,

d.nume\_developer

FROM

joc j

JOIN developer d ON j.id\_developer = d.id\_developer

JOIN locatie loc ON d.id\_locatie = loc.id\_locatie

WHERE

loc.id\_tara = v\_id\_tara

) LOOP

v\_index := v\_index + 1;

v\_jocuri(v\_index).id\_joc := rec.id\_joc;

v\_jocuri(v\_index).nume\_joc := rec.nume\_joc;

v\_jocuri(v\_index).nume\_developer := rec.nume\_developer;

v\_jocuri(v\_index).tara := v\_nume\_tara;

END LOOP;

RETURN v\_jocuri;

END afisare\_jocuri\_natale;

END steam\_pack;

/

DECLARE

v\_jocuri steam\_pack.jocuri\_natale\_table;

BEGIN

dbms\_output.put\_line('Toate itemele utilizatorului: ');

steam\_pack.afiseaza\_iteme\_in\_inventar(1); -- id user

dbms\_output.put\_line('Toate jocurile dezvoltatorului: ');

steam\_pack.afiseaza\_jocuri(6); -- id dezvoltator

dbms\_output.put\_line('Numarul de iteme cu pret mai mare de 50 din inventarul utilizatorului: '

|| steam\_pack.calculeaza\_numar\_iteme(1, 50)); -- id user

v\_jocuri := steam\_pack.afisare\_jocuri\_natale(1); -- id user

dbms\_output.put\_line('Toate jocurile dezvoltate in tara utilizatorului: ');

FOR i IN 1..v\_jocuri.count LOOP

dbms\_output.put\_line('ID Joc: '

|| v\_jocuri(i).id\_joc

|| ', Nume Joc: '

|| v\_jocuri(i).nume\_joc

|| ', Nume Developer: '

|| v\_jocuri(i).nume\_developer

|| ', Tara: '

|| v\_jocuri(i).tara);

END LOOP;

END;

/

