Tudorache Alexandru-Theodor

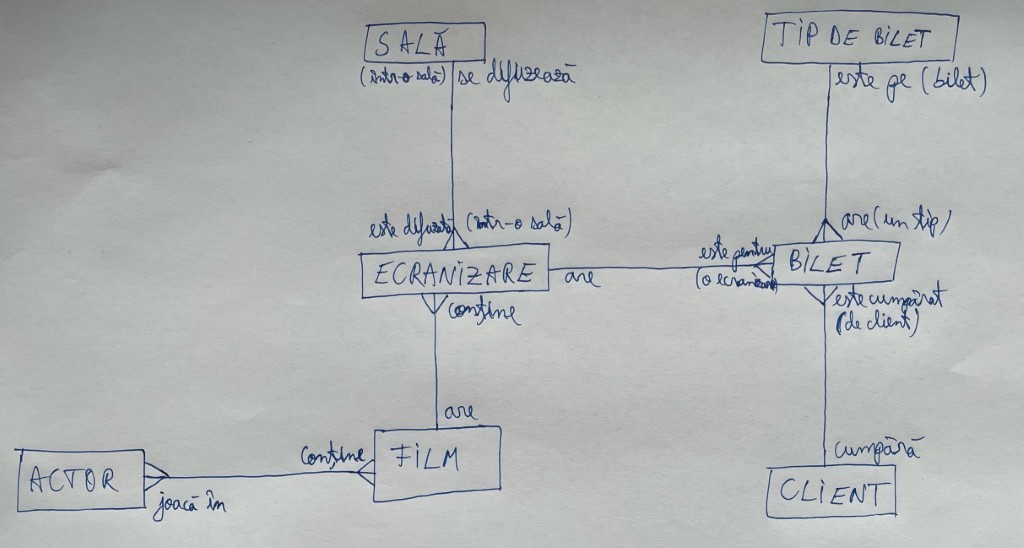
Grupa 242

**Proiect SGBD**

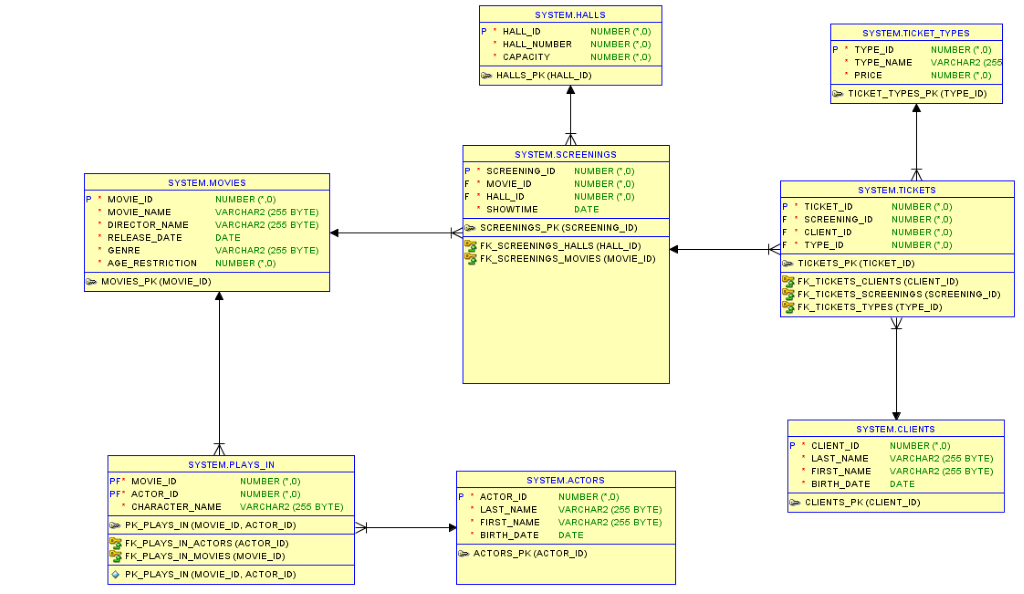
-Cinema Database-

1. Baza de date este a unui cinema cu o singură locație (nu un lanț de cinematografe precum Cinema City), dar cu săli multiple. Cu ajutorul acestei baze de date clienții își pot cumpăra bilete la anumite ecranizări ale filmelor pe care vor să le vizioneze, biletele având diferite tipuri pentru a putea determina prețul acestora. De asemenea, în baza de date se află și actorii care joacă în filmele care sunt ecranizate pentru a putea fi afișați alături de filmul/filmele în care joacă.

2.

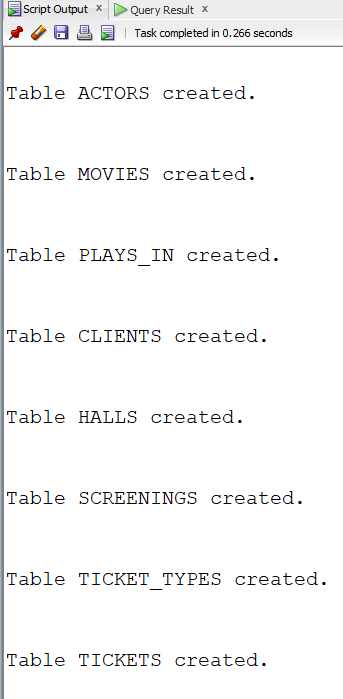


3.



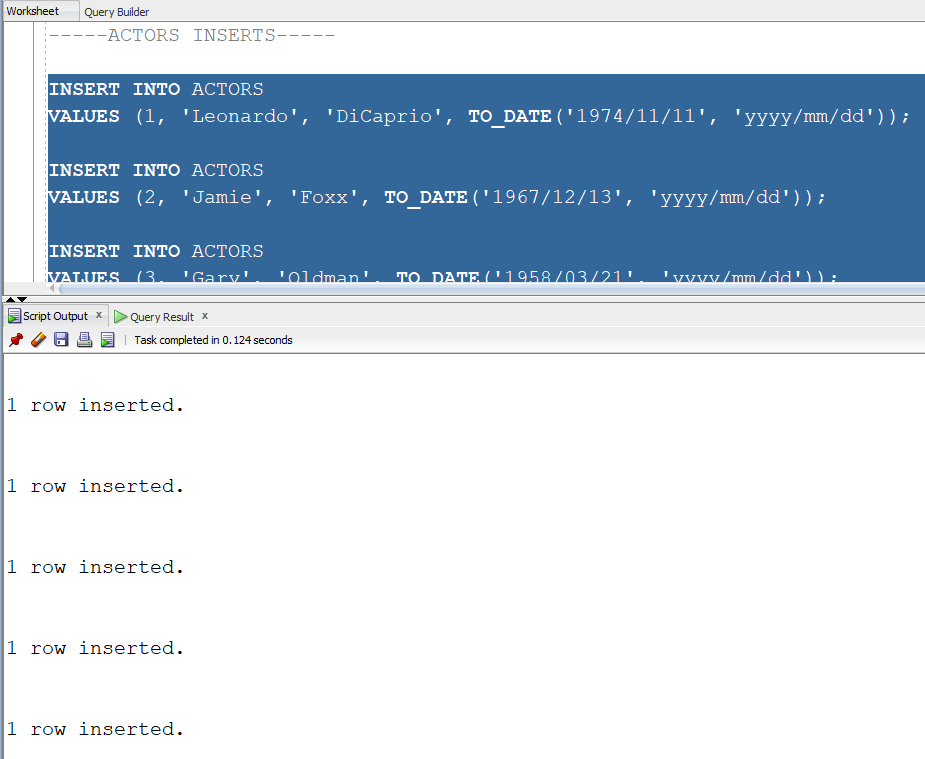
4.

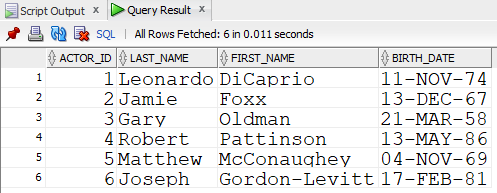
|  |
| --- |
| **CREATE** **TABLE** ACTORS (   actor\_id INT PRIMARY **KEY**,   last\_name varchar(255) **NOT** NULL,   first\_name varchar(255) **NOT** NULL,   birth\_date DATE **NOT** NULL  );    **CREATE** **TABLE** MOVIES (   movie\_id INT PRIMARY **KEY**,   movie\_name varchar(255) **NOT** NULL,   director\_name varchar(255) **NOT** NULL,   release\_date DATE **NOT** NULL,   genre varchar(255) **NOT** NULL,   age\_restriction int **NOT** NULL  );    **CREATE** **TABLE** PLAYS\_IN (   movie\_id INT **NOT** NULL,   actor\_id INT **NOT** NULL,   character\_name varchar(255) **NOT** NULL,   **CONSTRAINT** FK\_PLAYS\_IN\_MOVIES FOREIGN **KEY** (movie\_id) **REFERENCES** MOVIES(movie\_id),   **CONSTRAINT** FK\_PLAYS\_IN\_ACTORS FOREIGN **KEY** (actor\_id) **REFERENCES** ACTORS(actor\_id),   **CONSTRAINT** PK\_PLAYS\_IN PRIMARY **KEY** (movie\_id, actor\_id)  );    **CREATE** **TABLE** CLIENTS (   client\_id INT PRIMARY **KEY**,   last\_name varchar(255) **NOT** NULL,   first\_name varchar(255) **NOT** NULL,   birth\_date DATE **NOT** NULL  );    **CREATE** **TABLE** HALLS (   hall\_id INT PRIMARY **KEY**,   hall\_number INT **NOT** NULL,   **capacity** INT **NOT** NULL  );    **CREATE** **TABLE** SCREENINGS (   screening\_id INT PRIMARY **KEY**,   movie\_id INT **NOT** NULL,   hall\_id INT **NOT** NULL,   showtime DATE **NOT** NULL,   **CONSTRAINT** FK\_SCREENINGS\_MOVIES FOREIGN **KEY** (movie\_id) **REFERENCES** MOVIES(movie\_id),   **CONSTRAINT** FK\_SCREENINGS\_HALLS FOREIGN **KEY** (hall\_id) **REFERENCES** HALLS(hall\_id)  );    **CREATE** **TABLE** TICKET\_TYPES (   type\_id INT PRIMARY **KEY**,   type\_name VARCHAR(255) **NOT** NULL,   price INT **NOT** NULL  );    **CREATE** **TABLE** TICKETS (   ticket\_id INT PRIMARY **KEY**,   screening\_id INT **NOT** NULL,   client\_id INT **NOT** NULL,   type\_id INT **NOT** NULL,   **CONSTRAINT** FK\_TICKETS\_SCREENINGS FOREIGN **KEY** (screening\_id) **REFERENCES** SCREENINGS(screening\_id),   **CONSTRAINT** FK\_TICKETS\_CLIENTS FOREIGN **KEY** (client\_id) **REFERENCES** CLIENTS(client\_id),   **CONSTRAINT** FK\_TICKETS\_TYPES FOREIGN **KEY** (type\_id) **REFERENCES** TICKET\_TYPES(type\_id)  ); |



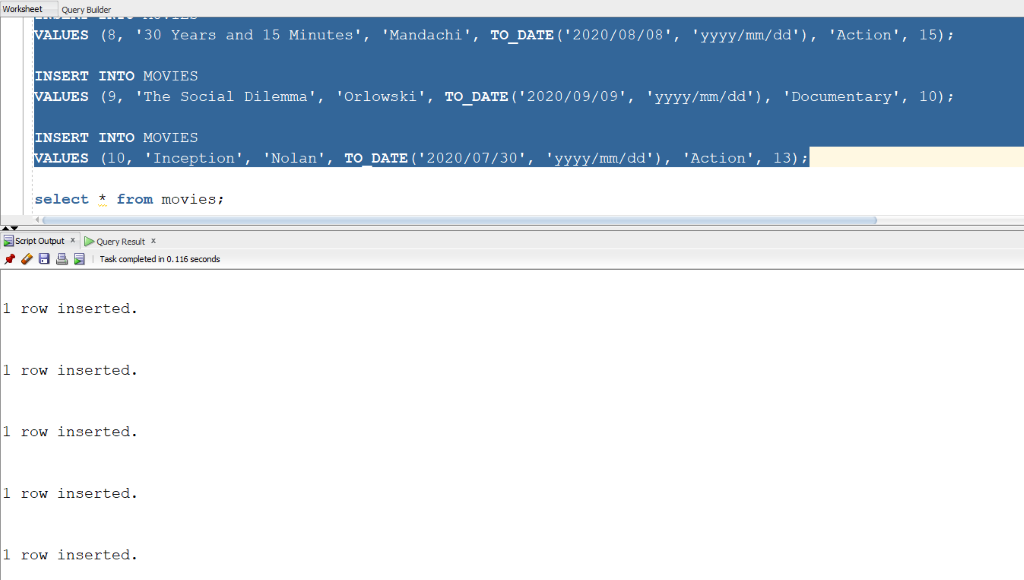
5.

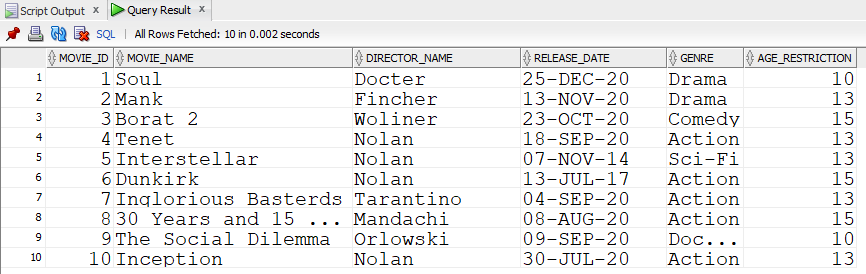
|  |
| --- |
| -----ACTORS INSERTS-----    **INSERT** **INTO** ACTORS  **VALUES** (1, 'Leonardo', 'DiCaprio', **TO\_DATE**('1974/11/11', 'yyyy/mm/dd'));    **INSERT** **INTO** ACTORS  **VALUES** (2, 'Jamie', 'Foxx', **TO\_DATE**('1967/12/13', 'yyyy/mm/dd'));    **INSERT** **INTO** ACTORS  **VALUES** (3, 'Gary', 'Oldman', **TO\_DATE**('1958/03/21', 'yyyy/mm/dd'));    **INSERT** **INTO** ACTORS  **VALUES** (4, 'Robert', 'Pattinson', **TO\_DATE**('1986/05/13', 'yyyy/mm/dd'));    **INSERT** **INTO** ACTORS  **VALUES** (5, 'Matthew', 'McConaughey', **TO\_DATE**('1969/11/04', 'yyyy/mm/dd'));    **INSERT** **INTO** ACTORS  **VALUES** (6, 'Joseph', 'Gordon-Levitt', **TO\_DATE**('1981/02/17', 'yyyy/mm/dd'));    **select** \* **from** actors; |



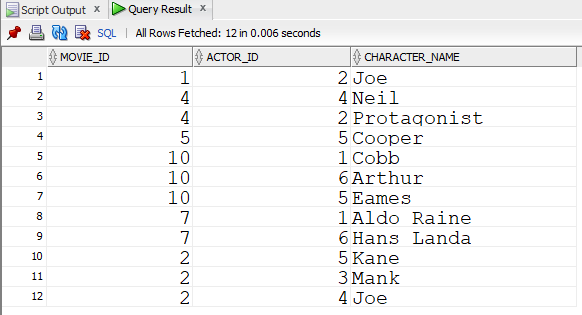
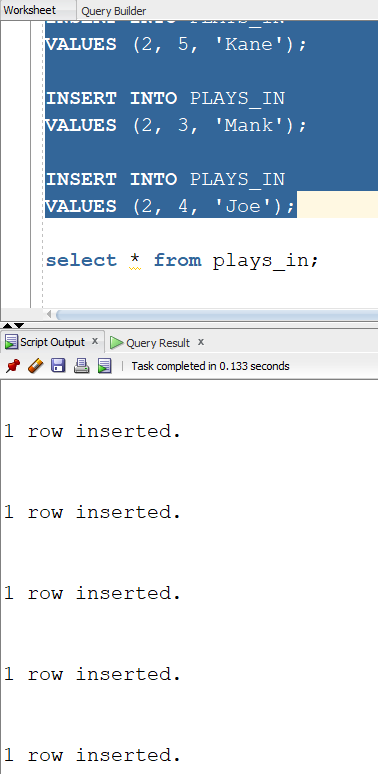


|  |
| --- |
| -----MOVIES INSERTS-----    **INSERT** **INTO** MOVIES  **VALUES** (1, 'Soul', 'Docter', **TO\_DATE**('2020/12/25', 'yyyy/mm/dd'), 'Drama', 10);    **INSERT** **INTO** MOVIES  **VALUES** (2, 'Mank', 'Fincher', **TO\_DATE**('2020/11/13', 'yyyy/mm/dd'), 'Drama', 13);    **INSERT** **INTO** MOVIES  **VALUES** (3, 'Borat 2', 'Woliner', **TO\_DATE**('2020/10/23', 'yyyy/mm/dd'), 'Comedy', 15);    **INSERT** **INTO** MOVIES  **VALUES** (4, 'Tenet', 'Nolan', **TO\_DATE**('2020/09/18', 'yyyy/mm/dd'), 'Action', 13);    **INSERT** **INTO** MOVIES  **VALUES** (5, 'Interstellar', 'Nolan', **TO\_DATE**('2014/11/07', 'yyyy/mm/dd'), 'Sci-Fi', 13);    **INSERT** **INTO** MOVIES  **VALUES** (6, 'Dunkirk', 'Nolan', **TO\_DATE**('2017/07/13', 'yyyy/mm/dd'), 'Action', 15);    **INSERT** **INTO** MOVIES  **VALUES** (7, 'Inglorious Basterds', 'Tarantino', **TO\_DATE**('2020/09/04', 'yyyy/mm/dd'), 'Action', 13);    **INSERT** **INTO** MOVIES  **VALUES** (8, '30 Years and 15 Minutes', 'Mandachi', **TO\_DATE**('2020/08/08', 'yyyy/mm/dd'), 'Action', 15);    **INSERT** **INTO** MOVIES  **VALUES** (9, 'The Social Dilemma', 'Orlowski', **TO\_DATE**('2020/09/09', 'yyyy/mm/dd'), 'Documentary', 10);    **INSERT** **INTO** MOVIES  **VALUES** (10, 'Inception', 'Nolan', **TO\_DATE**('2020/07/30', 'yyyy/mm/dd'), 'Action', 13);    **select** \* **from** movies; |



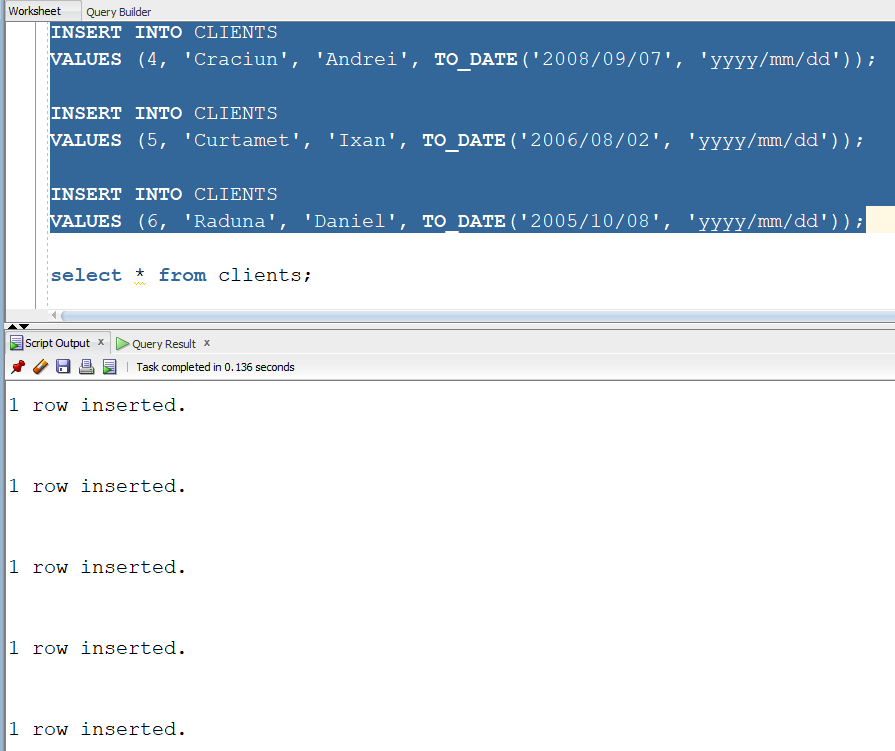


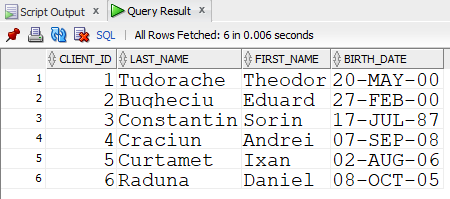
|  |
| --- |
| -----PLAYS IN INSERTS-----    **INSERT** **INTO** PLAYS\_IN  **VALUES** (1, 2, 'Joe');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (4, 4, 'Neil');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (4, 2, 'Protagonist');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (5, 5, 'Cooper');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (10, 1, 'Cobb');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (10, 6, 'Arthur');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (10, 5, 'Eames');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (7, 1, 'Aldo Raine');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (7, 6, 'Hans Landa');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (2, 5, 'Kane');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (2, 3, 'Mank');    **INSERT** **INTO** PLAYS\_IN  **VALUES** (2, 4, 'Joe');    **select** \* **from** plays\_in; |



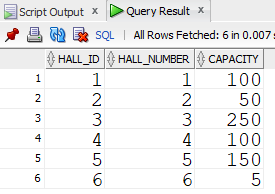
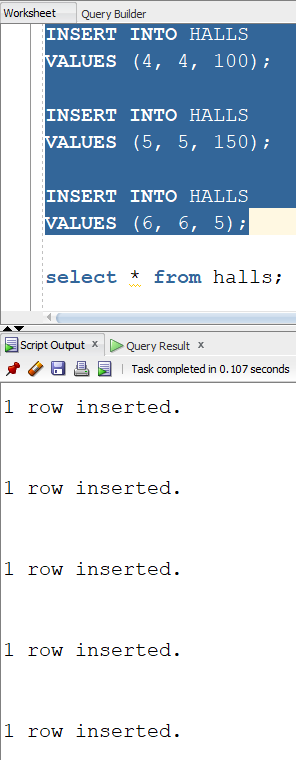
|  |
| --- |
| -----CLIENTS INSERTS-----    **INSERT** **INTO** CLIENTS  **VALUES** (1, 'Tudorache', 'Theodor', **TO\_DATE**('2000/05/20', 'yyyy/mm/dd'));    **INSERT** **INTO** CLIENTS  **VALUES** (2, 'Bugheciu', 'Eduard', **TO\_DATE**('2000/02/27', 'yyyy/mm/dd'));    **INSERT** **INTO** CLIENTS  **VALUES** (3, 'Constantin', 'Sorin', **TO\_DATE**('1987/07/17', 'yyyy/mm/dd'));    **INSERT** **INTO** CLIENTS  **VALUES** (4, 'Craciun', 'Andrei', **TO\_DATE**('2008/09/07', 'yyyy/mm/dd'));    **INSERT** **INTO** CLIENTS  **VALUES** (5, 'Curtamet', 'Ixan', **TO\_DATE**('2006/08/02', 'yyyy/mm/dd'));    **INSERT** **INTO** CLIENTS  **VALUES** (6, 'Raduna', 'Daniel', **TO\_DATE**('2005/10/08', 'yyyy/mm/dd')); |

select \* from clients;

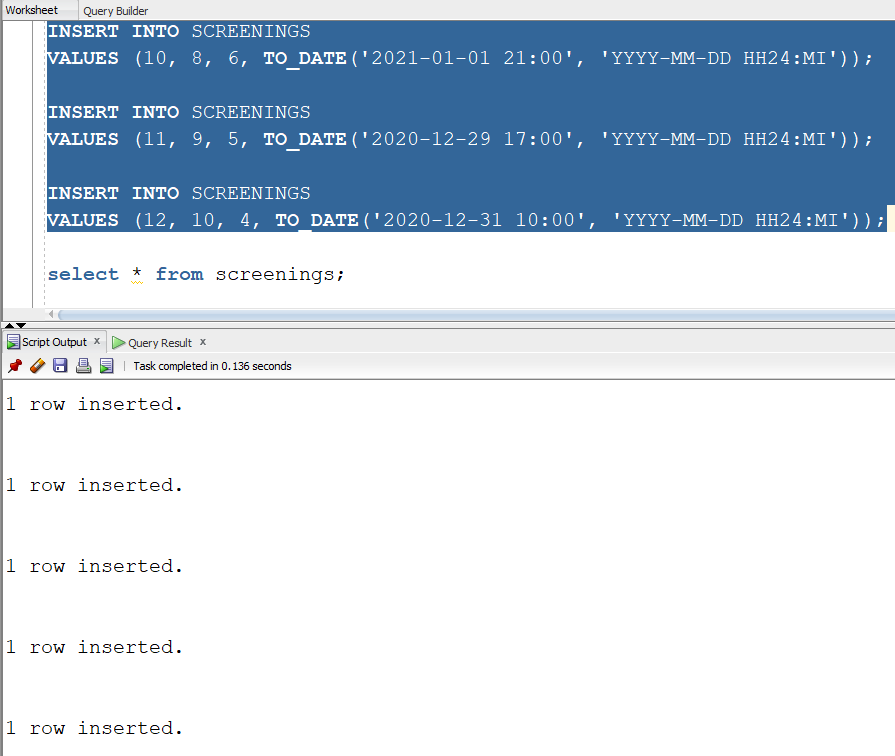


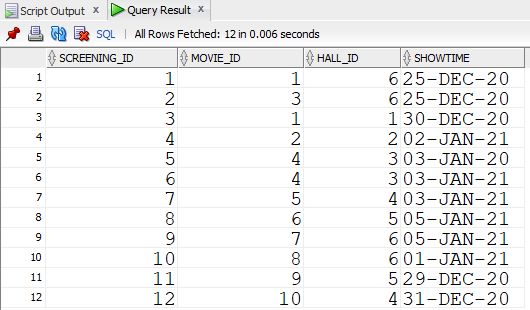


|  |
| --- |
| -----HALLS INSERTS-----    **INSERT** **INTO** HALLS  **VALUES** (1, 1, 100);    **INSERT** **INTO** HALLS  **VALUES** (2, 2, 50);    **INSERT** **INTO** HALLS  **VALUES** (3, 3, 250);    **INSERT** **INTO** HALLS  **VALUES** (4, 4, 100);    **INSERT** **INTO** HALLS  **VALUES** (5, 5, 150);    **INSERT** **INTO** HALLS  **VALUES** (6, 6, 5);    **select** \* **from** halls; |

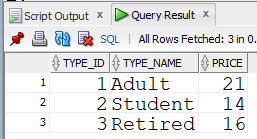
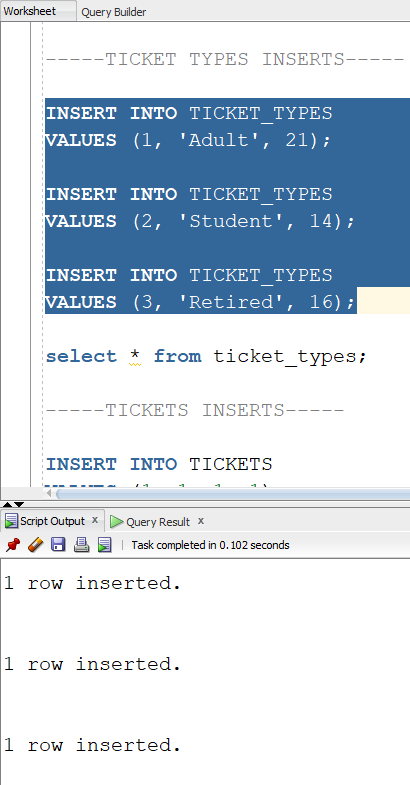


|  |
| --- |
| -----SCREENINGS INSERTS-----    **INSERT** **INTO** SCREENINGS  **VALUES** (1, 1, 6, **TO\_DATE**('2020-12-25 13:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (2, 3, 6, **TO\_DATE**('2020-12-25 17:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (3, 1, 1, **TO\_DATE**('2020-12-30 21:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (4, 2, 2, **TO\_DATE**('2021-01-02 20:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (5, 4, 3, **TO\_DATE**('2020-01-03 17:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (6, 4, 3, **TO\_DATE**('2021-01-03 17:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (7, 5, 4, **TO\_DATE**('2021-01-03 13:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (8, 6, 5, **TO\_DATE**('2021-01-05 11:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (9, 7, 6, **TO\_DATE**('2021-01-05 20:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (10, 8, 6, **TO\_DATE**('2021-01-01 21:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (11, 9, 5, **TO\_DATE**('2020-12-29 17:00', 'YYYY-MM-DD HH24:MI'));    **INSERT** **INTO** SCREENINGS  **VALUES** (12, 10, 4, **TO\_DATE**('2020-12-31 10:00', 'YYYY-MM-DD HH24:MI'));    **select** \* **from** screenings; |

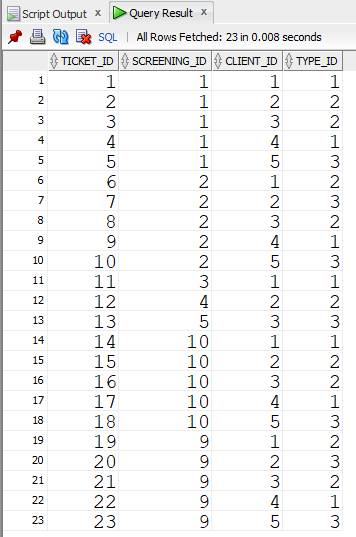
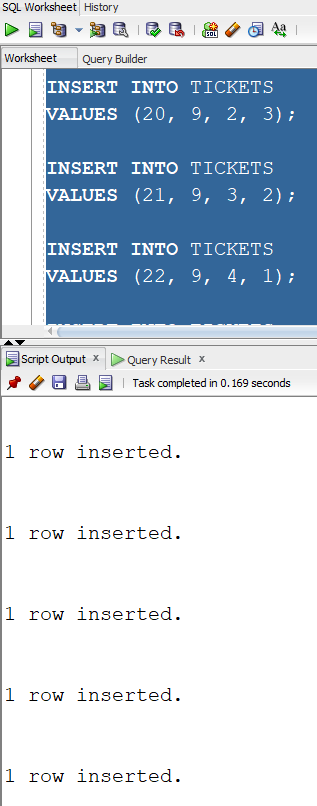




|  |
| --- |
| -----TICKET TYPES INSERTS-----    **INSERT** **INTO** TICKET\_TYPES  **VALUES** (1, 'Adult', 21);    **INSERT** **INTO** TICKET\_TYPES  **VALUES** (2, 'Student', 14);    **INSERT** **INTO** TICKET\_TYPES  **VALUES** (3, 'Retired', 16);    **select** \* **from** ticket\_types; |

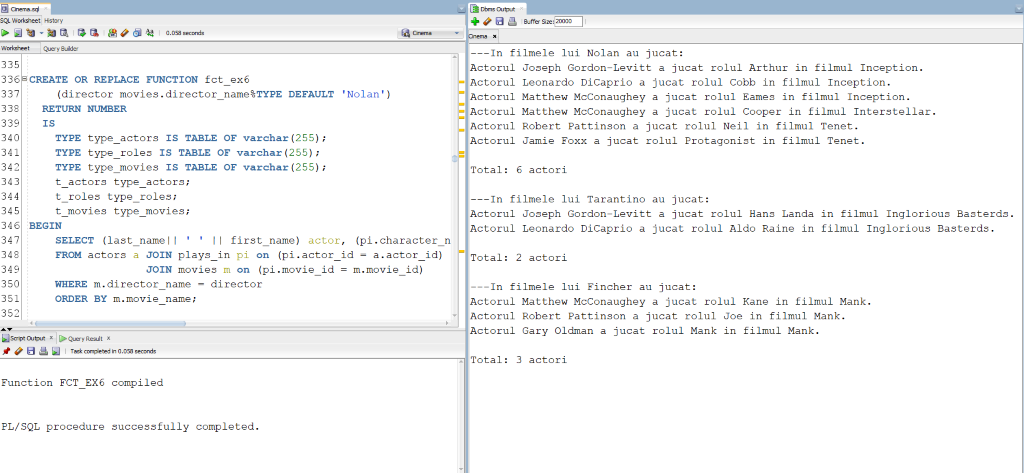


|  |
| --- |
| -----TICKETS INSERTS-----    **INSERT** **INTO** TICKETS  **VALUES** (1, 1, 1, 1);    **INSERT** **INTO** TICKETS  **VALUES** (2, 1, 2, 2);    **INSERT** **INTO** TICKETS  **VALUES** (3, 1, 3, 2);    **INSERT** **INTO** TICKETS  **VALUES** (4, 1, 4, 1);    **INSERT** **INTO** TICKETS  **VALUES** (5, 1, 5, 3);    **INSERT** **INTO** TICKETS  **VALUES** (6, 2, 1, 2);    **INSERT** **INTO** TICKETS  **VALUES** (7, 2, 2, 3);    **INSERT** **INTO** TICKETS  **VALUES** (8, 2, 3, 2);    **INSERT** **INTO** TICKETS  **VALUES** (9, 2, 4, 1);    **INSERT** **INTO** TICKETS  **VALUES** (10, 2, 5, 3);    **INSERT** **INTO** TICKETS  **VALUES** (11, 3, 1, 1);    **INSERT** **INTO** TICKETS  **VALUES** (12, 4, 2, 2);    **INSERT** **INTO** TICKETS  **VALUES** (13, 5, 3, 3);    **INSERT** **INTO** TICKETS  **VALUES** (14, 10, 1, 1);    **INSERT** **INTO** TICKETS  **VALUES** (15, 10, 2, 2);    **INSERT** **INTO** TICKETS  **VALUES** (16, 10, 3, 2);    **INSERT** **INTO** TICKETS  **VALUES** (17, 10, 4, 1);    **INSERT** **INTO** TICKETS  **VALUES** (18, 10, 5, 3);    **INSERT** **INTO** TICKETS  **VALUES** (19, 9, 1, 2);    **INSERT** **INTO** TICKETS  **VALUES** (20, 9, 2, 3);    **INSERT** **INTO** TICKETS  **VALUES** (21, 9, 3, 2);    **INSERT** **INTO** TICKETS  **VALUES** (22, 9, 4, 1);    **INSERT** **INTO** TICKETS  **VALUES** (23, 9, 5, 3);    **select** \* **from** tickets; |



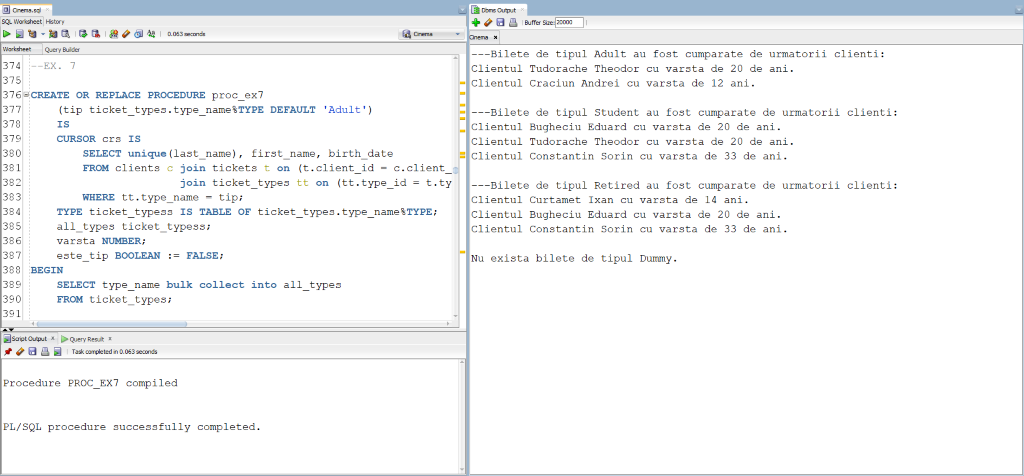
6. Să se definească o funcție care să afișeze numele actorilor, dar și personajul lor, care au jucat în filmele regizate de un regizor cu nume dat că parametru(Parametrul default va fi 'Nolan'). Acest subprogram trebuie să returneze numărul acestor actori.

|  |
| --- |
| **CREATE** **OR** REPLACE **FUNCTION** **fct\_ex6**   (director movies.director\_name%**TYPE** **DEFAULT** 'Nolan')   **RETURN** **NUMBER**   **IS**   **TYPE** **type\_actors** **IS** **TABLE** **OF** **varchar**(255);   **TYPE** type\_roles **IS** TABLE **OF** varchar(255);   **TYPE** type\_movies **IS** TABLE **OF** varchar(255);   t\_actors type\_actors;   t\_roles type\_roles;   t\_movies type\_movies;  **BEGIN**   **SELECT** (last\_name|| ' ' || first\_name) actor, (pi.character\_name) personaj, (m.movie\_name) BULK COLLECT **INTO** t\_actors, t\_roles, t\_movies   **FROM** actors a **JOIN** plays\_in pi **on** (pi.actor\_id = a.actor\_id)   **JOIN** movies m **on** (pi.movie\_id = m.movie\_id)   **WHERE** m.director\_name = director   **ORDER** **BY** m.movie\_name;     DBMS\_OUTPUT.PUT\_LINE('---In filmele lui ' || director || ' au jucat:');     **FOR** I **in** t\_actors.first..t\_actors.last **LOOP**   DBMS\_OUTPUT.PUT\_LINE('Actorul ' || t\_actors(i) || ' a jucat rolul ' || t\_roles(i) || ' in filmul ' || t\_movies(i) || '.');   **END** **LOOP**;   DBMS\_OUTPUT.NEW\_LINE;   RETURN t\_actors.count;  **END** fct\_ex6;  /    --APELARE:    **BEGIN**   DBMS\_OUTPUT.PUT\_LINE('Total: ' || fct\_ex6() || ' actori');   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Total: ' || fct\_ex6('Tarantino') || ' actori');   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Total: ' || fct\_ex6('Fincher') || ' actori');  **END**;  / |



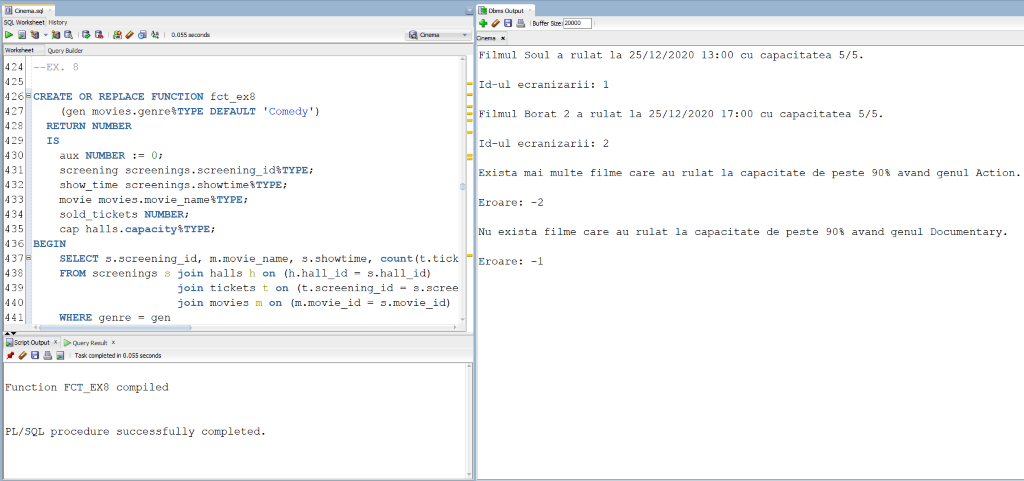
7. Să se definească o procedură care să afișeze(o singură dată) numele și vârstă tuturor clienților care au cumpărat bilete de tipul dat ca parametru(Parametrul default va fi 'Adult').

|  |
| --- |
| **CREATE** **OR** REPLACE **PROCEDURE** **proc\_ex7**   (tip ticket\_types.type\_name%**TYPE** **DEFAULT** 'Adult')   **IS**   **CURSOR** **crs** **IS**   **SELECT** **unique**(last\_name), **first\_name**, **birth\_date**   **FROM** **clients** **c** **join** **tickets** **t** **on** (t.client\_id = c.client\_id)   **join** **ticket\_types** **tt** **on** (tt.type\_id = t.type\_id)   **WHERE** **tt**.**type\_name** = **tip**;   **TYPE** ticket\_typess **IS** TABLE **OF** ticket\_types.type\_name%**TYPE**;   all\_types ticket\_typess;   varsta NUMBER;   este\_tip BOOLEAN := **FALSE**;  **BEGIN**   **SELECT** type\_name bulk collect **into** all\_types   **FROM** ticket\_types;     **FOR** i **IN** all\_types.first..all\_types.last **LOOP**   **IF** (all\_types(i) = tip) **THEN**   este\_tip := **TRUE**;   **END** **IF**;   **END** **LOOP**;   **IF** (este\_tip = **TRUE**) **THEN**   DBMS\_OUTPUT.PUT\_LINE('---Bilete de tipul ' || tip || ' au fost cumparate de urmatorii clienti: ');   **ELSE**   DBMS\_OUTPUT.PUT\_LINE ('Nu exista bilete de tipul ' || tip || '.');   **END** **IF**;   **FOR** i **in** crs **LOOP**   varsta := TRUNC(MONTHS\_BETWEEN(sysdate, i.birth\_date) / 12);   **IF** (varsta >= 20) **THEN**   DBMS\_OUTPUT.PUT\_LINE('Clientul ' || i.last\_name || ' ' || i.first\_name || ' cu varsta de ' || TRUNC(MONTHS\_BETWEEN(sysdate, i.birth\_date) / 12) || ' de ani.');   **ELSE**   DBMS\_OUTPUT.PUT\_LINE('Clientul ' || i.last\_name || ' ' || i.first\_name || ' cu varsta de ' || TRUNC(MONTHS\_BETWEEN(sysdate, i.birth\_date) / 12) || ' ani.');   **END** **IF**;   **END** **LOOP**;   DBMS\_OUTPUT.NEW\_LINE;  **END** proc\_ex7;  /    --APELARE:    **BEGIN**   proc\_ex7();   proc\_ex7('Student');   proc\_ex7('Retired');   proc\_ex7('Dummy'); -- nu se va afisa nimic  **END**;  / |



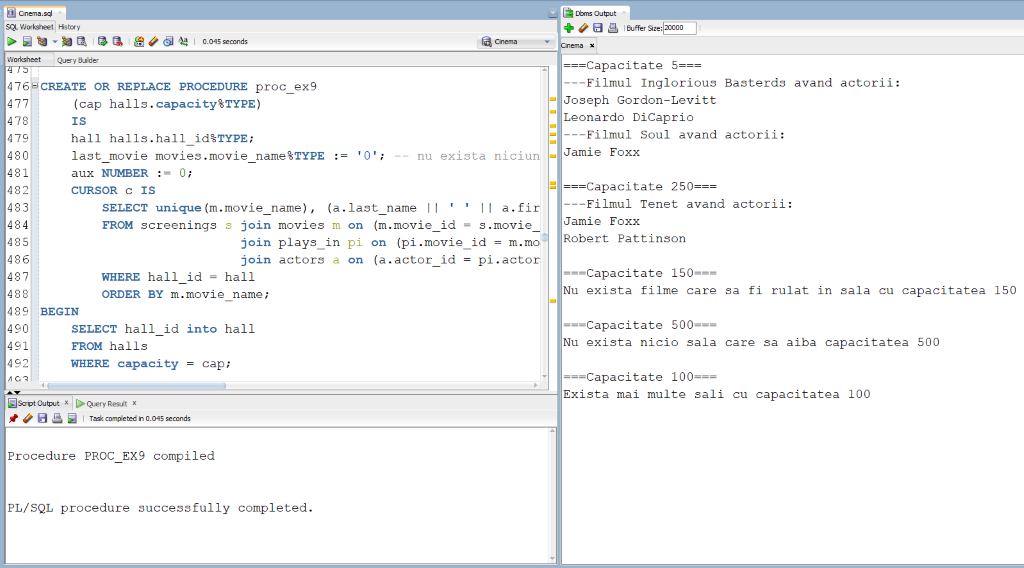
8. Să se definească o funcție care să afișeze următoarele detalii: (numele filmului; dată și ora la care a fost ecranizat) despre filmul care a rulat la o capacitate de peste 90% din sala în care au fost ecranizate, având genul dat ca parametru. Subprogramul trebuie să returneze id-ul ecranizării respective.

|  |
| --- |
| **CREATE** **OR** REPLACE **FUNCTION** **fct\_ex8**   (gen movies.genre%**TYPE** **DEFAULT** 'Comedy')   **RETURN** **NUMBER**   **IS**   **aux** **NUMBER** := 0;   screening screenings.screening\_id%**TYPE**;   show\_time screenings.showtime%**TYPE**;   movie movies.movie\_name%**TYPE**;   sold\_tickets NUMBER;   cap halls.capacity%**TYPE**;  **BEGIN**   **SELECT** s.screening\_id, m.movie\_name, s.showtime, count(t.ticket\_id) , h.capacity **INTO** screening, movie, show\_time,sold\_tickets, cap   **FROM** screenings s **join** halls h **on** (h.hall\_id = s.hall\_id)   **join** tickets t **on** (t.screening\_id = s.screening\_id)   **join** movies m **on** (m.movie\_id = s.movie\_id)   **WHERE** genre = gen   **GROUP** **BY** s.screening\_id, m.movie\_name, s.showtime ,h.capacity   HAVING COUNT(t.ticket\_id) >= h.capacity \* (9/10);   DBMS\_OUTPUT.PUT\_LINE('Filmul ' || movie || ' a rulat la ' || TO\_CHAR(show\_time, 'dd/mm/yyyy HH24:MI') || ' cu capacitatea ' || sold\_tickets || '/' || cap || '.');   DBMS\_OUTPUT.NEW\_LINE;   RETURN screening;   EXCEPTION   WHEN NO\_DATA\_FOUND **THEN**   DBMS\_OUTPUT.PUT\_LINE('Nu exista filme care au rulat la capacitate de peste 90% avand genul ' || gen || '.');   DBMS\_OUTPUT.NEW\_LINE;   return -1;   WHEN TOO\_MANY\_ROWS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Exista mai multe filme care au rulat la capacitate de peste 90% avand genul ' || gen || '.');   DBMS\_OUTPUT.NEW\_LINE;   return -2;   WHEN OTHERS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');   return -3;  **END** fct\_ex8;  /    --APELARE:    **BEGIN**   DBMS\_OUTPUT.PUT\_LINE('Id-ul ecranizarii: ' || fct\_ex8('Drama'));   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Id-ul ecranizarii: ' || fct\_ex8('Comedy'));   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Eroare: ' || fct\_ex8('Action')); -- mai mult de 1 film   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Eroare: ' || fct\_ex8('Documentary')); -- niciun film   DBMS\_OUTPUT.NEW\_LINE;  **END**;  / |



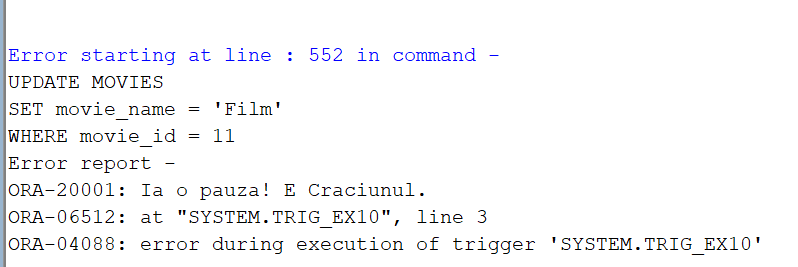
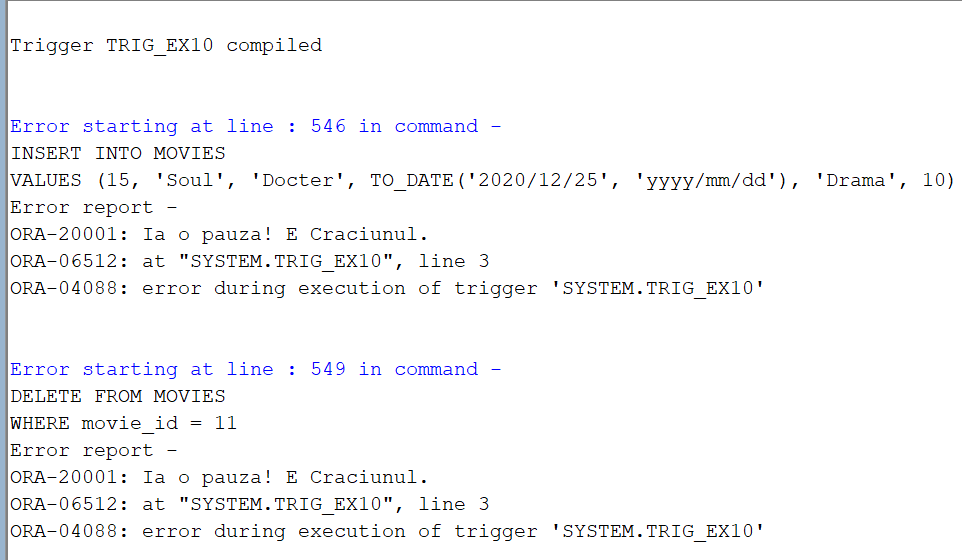
9. Să se definească o procedură care să afișeze numele filmelor(o singură dată) și numele actorilor care au jucat în ele, care au rulat în sala cu capacitate dată ca parametru.

|  |
| --- |
| **CREATE** **OR** REPLACE **PROCEDURE** **proc\_ex9**   (cap halls.capacity%**TYPE**)   **IS**   **hall** **halls**.**hall\_id**%**TYPE**;   last\_movie movies.movie\_name%**TYPE** := '0'; -- nu exista niciun film **in** baza de date cu numele '0'   aux NUMBER := 0;   CURSOR c **IS**   **SELECT** unique(m.movie\_name), (a.last\_name || ' ' || a.first\_name) nume   **FROM** screenings s **join** movies m **on** (m.movie\_id = s.movie\_id)   **join** plays\_in pi **on** (pi.movie\_id = m.movie\_id)   **join** actors a **on** (a.actor\_id = pi.actor\_id)   **WHERE** hall\_id = hall   **ORDER** **BY** m.movie\_name;  **BEGIN**   **SELECT** hall\_id **into** hall   **FROM** halls   **WHERE** capacity = cap;     **FOR** i **in** c **LOOP**   **IF** (i.movie\_name != last\_movie) **THEN**   DBMS\_OUTPUT.PUT\_LINE ('---Filmul ' || i.movie\_name || ' avand actorii: ');   **END** **IF**;   DBMS\_OUTPUT.PUT\_LINE (i.nume);   last\_movie := i.movie\_name;   aux := aux + 1;   **END** **LOOP**;   **IF** (aux = 0) **THEN**   DBMS\_OUTPUT.PUT\_LINE('Nu exista filme care sa fi rulat in sala cu capacitatea ' || cap);   **END** **IF**;   EXCEPTION   WHEN TOO\_MANY\_ROWS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Exista mai multe sali cu capacitatea ' || cap);   WHEN NO\_DATA\_FOUND **THEN**   DBMS\_OUTPUT.PUT\_LINE('Nu exista nicio sala care sa aiba capacitatea ' || cap);   WHEN OTHERS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Alta eroare');  **END** proc\_ex9;  /    --APELARE:    **BEGIN**   DBMS\_OUTPUT.PUT\_LINE('===Capacitate 5===');   proc\_ex9(5); -- mai multe filme   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('===Capacitate 250===');   proc\_ex9(250); -- un singur film   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('===Capacitate 150===');   proc\_ex9(150); -- niciun film   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('===Capacitate 500===');   proc\_ex9(500); -- nu exista capacitatea   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('===Capacitate 100===');   proc\_ex9(100); -- mai multe sali cu capacitatea respectiva  **END**;  / |



10. Să se definească un declanșator care să nu permită nimănui să lucreze în ziua de Crăciun cu comenzile UPDATE, INSERT, DELETE pe tabela filmelor. (Declanșatorul se va testa cu ziua în care ne aflăm, în loc de 25/12.)

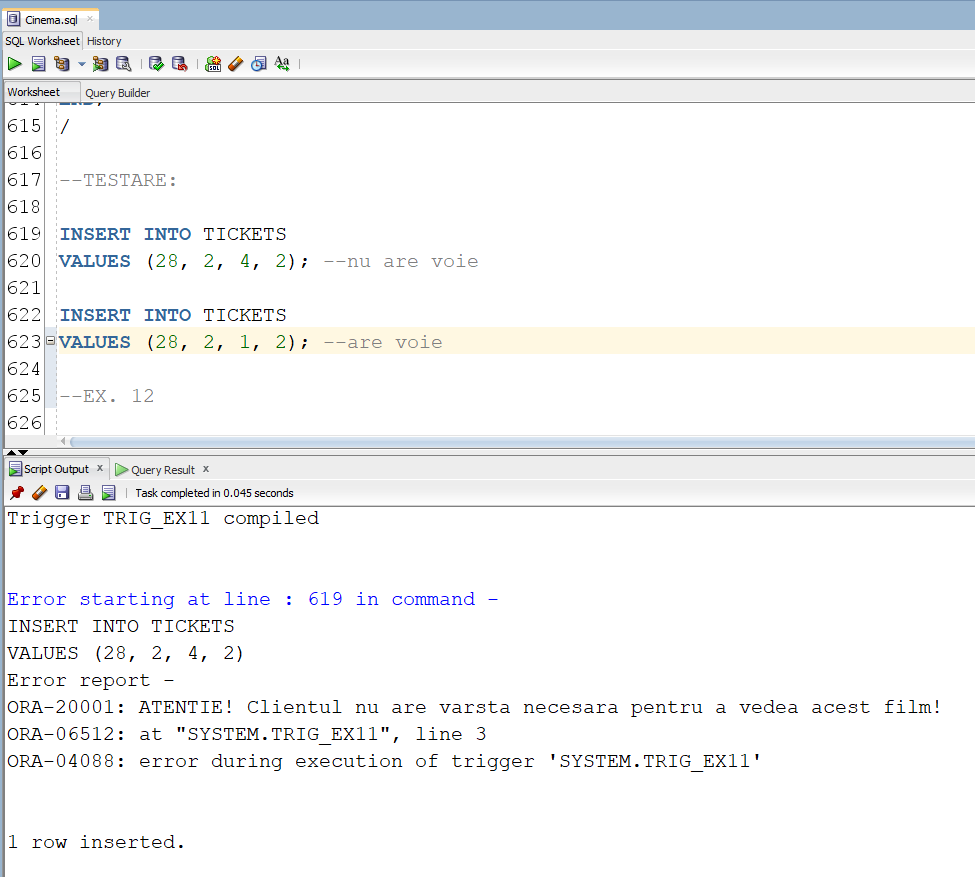
|  |
| --- |
| **CREATE** **OR** **REPLACE** **TRIGGER** trig\_ex10   **BEFORE** **INSERT** **OR** **UPDATE** **OR** **DELETE** **ON** movies  **BEGIN**   **IF** (TO\_CHAR(**SYSDATE**,'DD/MM') = '25/12') **THEN**   RAISE\_APPLICATION\_ERROR(-20001,'Ia o pauza! E Craciunul.');   **END** **IF**;  **END**;  /    --TESTARE:    **INSERT** **INTO** MOVIES  **VALUES** (15, 'Soul', 'Docter', **TO\_DATE**('2020/12/25', 'yyyy/mm/dd'), 'Drama', 10);    **DELETE** **FROM** MOVIES  **WHERE** movie\_id = 11;    **UPDATE** MOVIES  **SET** movie\_name = 'Film'  **WHERE** movie\_id = 11; |



11. Să se definească un declanșator care să nu permită(să afișeze o eroare) inserarea în tabela biletelor a unor valori corespunzătoare unui client care nu are vârsta necesară pentru a putea vedea o anumită ecranizare a unui film.

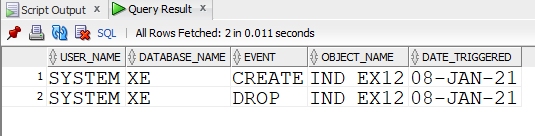
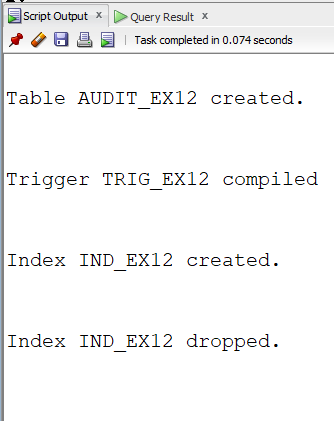
--functie auxiliara pentru declansator

|  |
| --- |
| CREATE **OR** REPLACE **FUNCTION** are\_voie\_ex11   (screening tickets.screening\_id%TYPE, id\_client clients.client\_id%TYPE)   **RETURN** BOOLEAN   **IS**   CURSOR crs **IS**   **SELECT** m.age\_restriction   **FROM** screenings s **join** movies m **on** (m.movie\_id = s.movie\_id)   **WHERE** s.screening\_id = screening;   CURSOR crs2 **IS**   **SELECT** TRUNC(MONTHS\_BETWEEN(sysdate, birth\_date) / 12)   **FROM** clients   **WHERE** client\_id = id\_client;   age number;   age\_r movies.age\_restriction%TYPE;  BEGIN   OPEN crs;   FETCH crs **INTO** age\_r;   CLOSE crs;   OPEN crs2;   FETCH crs2 **INTO** age;   CLOSE crs2;   **IF** (age\_r <= age) **THEN**   **RETURN** TRUE;   **ELSE**   **RETURN** FALSE;   **END** **IF**;  **END** are\_voie\_ex11;  /  CREATE **OR** REPLACE TRIGGER trig\_ex11   BEFORE INSERT **ON** tickets   **FOR** **EACH** ROW  BEGIN   **IF** (are\_voie\_ex11(:**NEW**.screening\_id, :**NEW**.client\_id) = FALSE) **THEN**   RAISE\_APPLICATION\_ERROR(-20001,'ATENTIE! Clientul nu are varsta necesara pentru a vedea acest film!');   **END** **IF**;  **END**;  /    --TESTARE:    INSERT **INTO** TICKETS  VALUES (28, 2, 4, 2); -- nu are voie    INSERT **INTO** TICKETS  VALUES (28, 2, 1, 2); -- are voie |



12. Să se definească un declanșator care să introducă date într-un tabel de tip audit după ce utilizatorul a folosit o comandă LDD.

|  |
| --- |
| **CREATE** **TABLE** audit\_ex12 (   user\_name VARCHAR2(30),   database\_name VARCHAR2(50),   **event** VARCHAR2(20),   object\_name VARCHAR2(30),   date\_triggered DATE  );    **CREATE** **OR** **REPLACE** **TRIGGER** trig\_ex12   **AFTER** **CREATE** **OR** **DROP** **OR** **ALTER** **ON** **SCHEMA**  **BEGIN**   **INSERT** **INTO** audit\_ex12   **VALUES** (SYS.LOGIN\_USER,   SYS.DATABASE\_NAME,   SYS.SYSEVENT,   SYS.DICTIONARY\_OBJ\_NAME,   **SYSDATE**);  **END**;  /    --TESTARE:    **CREATE** **INDEX** ind\_ex12  **ON** movies(movie\_name);    **DROP** **INDEX** ind\_ex12;    **SELECT** \* **FROM** audit\_ex12; |



13.

|  |
| --- |
| **CREATE** **OR** REPLACE PACKAGE package\_ex13 **AS**   **FUNCTION** **pack\_fct\_ex6**(director movies.director\_name%**TYPE** **DEFAULT** 'Nolan')   **RETURN** **NUMBER**;   **PROCEDURE** **pack\_proc\_ex7**(tip ticket\_types.type\_name%**TYPE** **DEFAULT** 'Adult');   **FUNCTION** **pack\_fct\_ex8**(gen movies.genre%**TYPE** **DEFAULT** 'Comedy')   **RETURN** **NUMBER**;   **PROCEDURE** **pack\_proc\_ex9**(cap halls.capacity%**TYPE**);   **FUNCTION** **pack\_are\_voie\_ex11**(screening tickets.screening\_id%**TYPE**, id\_client clients.client\_id%**TYPE**)   **RETURN** **BOOLEAN**;  **END** package\_ex13;  /    **CREATE** **OR** REPLACE PACKAGE BODY package\_ex13 **AS**   **FUNCTION** **pack\_fct\_ex6**   (director movies.director\_name%**TYPE** **DEFAULT** 'Nolan')   **RETURN** **NUMBER**   **IS**   **TYPE** **type\_actors** **IS** **TABLE** **OF** **varchar**(255);   **TYPE** type\_roles **IS** TABLE **OF** varchar(255);   **TYPE** type\_movies **IS** TABLE **OF** varchar(255);   t\_actors type\_actors;   t\_roles type\_roles;   t\_movies type\_movies;   **BEGIN**   **SELECT** (last\_name|| ' ' || first\_name) actor, (pi.character\_name) personaj, (m.movie\_name) BULK COLLECT **INTO** t\_actors, t\_roles, t\_movies   **FROM** actors a **JOIN** plays\_in pi **on** (pi.actor\_id = a.actor\_id)   **JOIN** movies m **on** (pi.movie\_id = m.movie\_id)   **WHERE** m.director\_name = director   **ORDER** **BY** m.movie\_name;     DBMS\_OUTPUT.PUT\_LINE('---In filmele lui ' || director || ' au jucat:');     **FOR** I **in** t\_actors.first..t\_actors.last **LOOP**   DBMS\_OUTPUT.PUT\_LINE('Actorul ' || t\_actors(i) || ' a jucat rolul ' || t\_roles(i) || ' in filmul ' || t\_movies(i) || '.');   **END** **LOOP**;   DBMS\_OUTPUT.NEW\_LINE;   RETURN t\_actors.count;   **END** pack\_fct\_ex6;       **PROCEDURE** **pack\_proc\_ex7**   (tip ticket\_types.type\_name%**TYPE** **DEFAULT** 'Adult')   **IS**   **CURSOR** **crs** **IS**   **SELECT** **unique**(last\_name), **first\_name**, **birth\_date**   **FROM** **clients** **c** **join** **tickets** **t** **on** (t.client\_id = c.client\_id)   **join** **ticket\_types** **tt** **on** (tt.type\_id = t.type\_id)   **WHERE** **tt**.**type\_name** = **tip**;   **TYPE** ticket\_typess **IS** TABLE **OF** ticket\_types.type\_name%**TYPE**;   all\_types ticket\_typess;   varsta NUMBER;   este\_tip BOOLEAN := **FALSE**;   **BEGIN**   **SELECT** type\_name bulk collect **into** all\_types   **FROM** ticket\_types;     **FOR** i **IN** all\_types.first..all\_types.last **LOOP**   **IF** (all\_types(i) = tip) **THEN**   este\_tip := **TRUE**;   **END** **IF**;   **END** **LOOP**;   **IF** (este\_tip = **TRUE**) **THEN**   DBMS\_OUTPUT.PUT\_LINE('---Bilete de tipul ' || tip || ' au fost cumparate de urmatorii clienti: ');   **ELSE**   DBMS\_OUTPUT.PUT\_LINE ('Nu exista bilete de tipul ' || tip || '.');   **END** **IF**;   **FOR** i **in** crs **LOOP**   varsta := TRUNC(MONTHS\_BETWEEN(sysdate, i.birth\_date) / 12);   **IF** (varsta >= 20) **THEN**   DBMS\_OUTPUT.PUT\_LINE('Clientul ' || i.last\_name || ' ' || i.first\_name || ' cu varsta de ' || TRUNC(MONTHS\_BETWEEN(sysdate, i.birth\_date) / 12) || ' de ani.');   **ELSE**   DBMS\_OUTPUT.PUT\_LINE('Clientul ' || i.last\_name || ' ' || i.first\_name || ' cu varsta de ' || TRUNC(MONTHS\_BETWEEN(sysdate, i.birth\_date) / 12) || ' ani.');   **END** **IF**;   **END** **LOOP**;   DBMS\_OUTPUT.NEW\_LINE;   **END** pack\_proc\_ex7;       **FUNCTION** **pack\_fct\_ex8**   (gen movies.genre%**TYPE** **DEFAULT** 'Comedy')   **RETURN** **NUMBER**   **IS**   **aux** **NUMBER** := 0;   screening screenings.screening\_id%**TYPE**;   show\_time screenings.showtime%**TYPE**;   movie movies.movie\_name%**TYPE**;   sold\_tickets NUMBER;   cap halls.capacity%**TYPE**;   **BEGIN**   **SELECT** s.screening\_id, m.movie\_name, s.showtime, count(t.ticket\_id) , h.capacity **INTO** screening, movie, show\_time,sold\_tickets, cap   **FROM** screenings s **join** halls h **on** (h.hall\_id = s.hall\_id)   **join** tickets t **on** (t.screening\_id = s.screening\_id)   **join** movies m **on** (m.movie\_id = s.movie\_id)   **WHERE** genre = gen   **GROUP** **BY** s.screening\_id, m.movie\_name, s.showtime ,h.capacity   HAVING COUNT(t.ticket\_id) >= h.capacity \* (9/10);   DBMS\_OUTPUT.PUT\_LINE('Filmul ' || movie || ' a rulat la ' || TO\_CHAR(show\_time, 'dd/mm/yyyy HH24:MI') || ' cu capacitatea ' || sold\_tickets || '/' || cap || '.');   DBMS\_OUTPUT.NEW\_LINE;   RETURN screening;   EXCEPTION   WHEN NO\_DATA\_FOUND **THEN**   DBMS\_OUTPUT.PUT\_LINE('Nu exista filme care au rulat la capacitate de peste 90% avand genul ' || gen);   DBMS\_OUTPUT.NEW\_LINE;   return -1;   WHEN TOO\_MANY\_ROWS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Exista mai multe filme care au rulat la capacitate de peste 90% avand genul ' || gen);   DBMS\_OUTPUT.NEW\_LINE;   return -2;   WHEN OTHERS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Alta eroare!');   return -3;   **END** pack\_fct\_ex8;     **PROCEDURE** **pack\_proc\_ex9**   (cap halls.capacity%**TYPE**)   **IS**   **hall** **halls**.**hall\_id**%**TYPE**;   last\_movie movies.movie\_name%**TYPE** := '0'; -- nu exista niciun film **in** baza de date cu numele '0'   aux NUMBER := 0;   CURSOR c **IS**   **SELECT** unique(m.movie\_name), (a.last\_name || ' ' || a.first\_name) nume   **FROM** screenings s **join** movies m **on** (m.movie\_id = s.movie\_id)   **join** plays\_in pi **on** (pi.movie\_id = m.movie\_id)   **join** actors a **on** (a.actor\_id = pi.actor\_id)   **WHERE** hall\_id = hall   **ORDER** **BY** m.movie\_name;   **BEGIN**   **SELECT** hall\_id **into** hall   **FROM** halls   **WHERE** capacity = cap;     **FOR** i **in** c **LOOP**   **IF** (i.movie\_name != last\_movie) **THEN**   DBMS\_OUTPUT.PUT\_LINE ('---Filmul ' || i.movie\_name || ' avand actorii: ');   **END** **IF**;   DBMS\_OUTPUT.PUT\_LINE (i.nume);   last\_movie := i.movie\_name;   aux := aux + 1;   **END** **LOOP**;   **IF** (aux = 0) **THEN**   DBMS\_OUTPUT.PUT\_LINE('Nu exista filme care sa fi rulat in sala cu capacitatea ' || cap);   **END** **IF**;   EXCEPTION   WHEN TOO\_MANY\_ROWS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Exista mai multe sali cu capacitatea ' || cap);   WHEN NO\_DATA\_FOUND **THEN**   DBMS\_OUTPUT.PUT\_LINE('Nu exista nicio sala care sa aiba capacitatea ' || cap);   WHEN OTHERS **THEN**   DBMS\_OUTPUT.PUT\_LINE('Alta eroare');   **END** pack\_proc\_ex9;       **FUNCTION** **pack\_are\_voie\_ex11**   (screening tickets.screening\_id%**TYPE**, id\_client clients.client\_id%**TYPE**)   **RETURN** **BOOLEAN**   **IS**   **CURSOR** **crs** **IS**   **SELECT** **m**.**age\_restriction**   **FROM** **screenings** **s** **join** **movies** **m** **on** (m.movie\_id = s.movie\_id)   **WHERE** **s**.**screening\_id** = **screening**;   CURSOR crs2 **IS**   **SELECT** TRUNC(MONTHS\_BETWEEN(sysdate, birth\_date) / 12)   **FROM** clients   **WHERE** client\_id = id\_client;   age number;   age\_r movies.age\_restriction%**TYPE**;   **BEGIN**   OPEN crs;   FETCH crs **INTO** age\_r;   CLOSE crs;   OPEN crs2;   FETCH crs2 **INTO** age;   CLOSE crs2;   **IF** (age\_r <= age) **THEN**   RETURN **TRUE**;   **ELSE**   RETURN **FALSE**;   **END** **IF**;   **END** pack\_are\_voie\_ex11;  **END** package\_ex13;  /    --APELARE:    **BEGIN**   DBMS\_OUTPUT.PUT\_LINE('===6===');   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Total: ' || package\_ex13.pack\_fct\_ex6() || ' actori');   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Total: ' || package\_ex13.pack\_fct\_ex6('Tarantino') || ' actori');   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE('Total: ' || package\_ex13.pack\_fct\_ex6('Fincher') || ' actori');   DBMS\_OUTPUT.NEW\_LINE;     DBMS\_OUTPUT.PUT\_LINE('===7===');   DBMS\_OUTPUT.NEW\_LINE;   package\_ex13.pack\_proc\_ex7();   package\_ex13.pack\_proc\_ex7('Student');   package\_ex13.pack\_proc\_ex7('Retired');   package\_ex13.pack\_proc\_ex7('Dummy');     DBMS\_OUTPUT.PUT\_LINE('===8===');   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE(package\_ex13.pack\_fct\_ex8('Drama'));   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE(package\_ex13.pack\_fct\_ex8('Comedy'));   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE(package\_ex13.pack\_fct\_ex8('Action')); -- mai mult de 1 film   DBMS\_OUTPUT.NEW\_LINE;   DBMS\_OUTPUT.PUT\_LINE(package\_ex13.pack\_fct\_ex8('Documentary')); -- niciun film   DBMS\_OUTPUT.NEW\_LINE;     DBMS\_OUTPUT.PUT\_LINE('===9===');   DBMS\_OUTPUT.NEW\_LINE;   package\_ex13.pack\_proc\_ex9(5); -- mai multe filme   DBMS\_OUTPUT.NEW\_LINE;   package\_ex13.pack\_proc\_ex9(250); -- un singur film   DBMS\_OUTPUT.NEW\_LINE;   package\_ex13.pack\_proc\_ex9(150); -- niciun film   DBMS\_OUTPUT.NEW\_LINE;   package\_ex13.pack\_proc\_ex9(500); -- nu exista capacitatea   DBMS\_OUTPUT.NEW\_LINE;   package\_ex13.pack\_proc\_ex9(100); -- mai multe sali cu capacitatea respectiva   DBMS\_OUTPUT.NEW\_LINE;     DBMS\_OUTPUT.PUT\_LINE('===11===');   DBMS\_OUTPUT.NEW\_LINE;   **IF** (package\_ex13.pack\_are\_voie\_ex11(2, 4) = **TRUE**) **THEN** --client cu varsta 12 vrea sa mearga la film cu restrictie de 15   DBMS\_OUTPUT.PUT\_LINE('Allowed');   **ELSE**   DBMS\_OUTPUT.PUT\_LINE('Not allowed');   **END** **IF**;   **IF** (package\_ex13.pack\_are\_voie\_ex11(2, 1) = **TRUE**) **THEN** --client cu varsta 20 vrea sa mearga la film cu restrictie de 15   DBMS\_OUTPUT.PUT\_LINE('Allowed');   **ELSE**   DBMS\_OUTPUT.PUT\_LINE('Not allowed');   **END** **IF**;  **END**;  / |

