Tema 6 - Laborator

Aelenei Alex

Cuprins

1 Introducere

Diagrama conceptuală și diagrama entitate-relație utilizate în această temă sunt mai jos. Datele de test folosite sunt cele utilizate în Tema 1 și Tema 2.

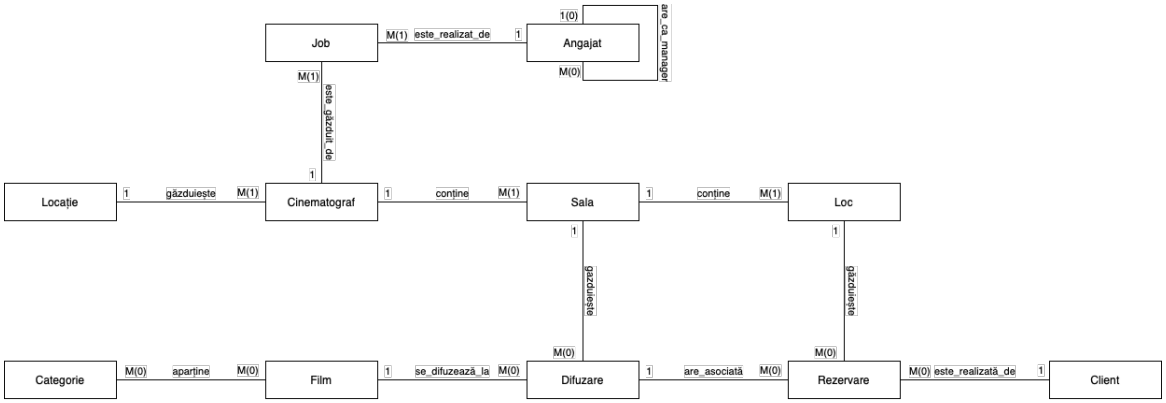

Figura 1: Diagrama conceptuală

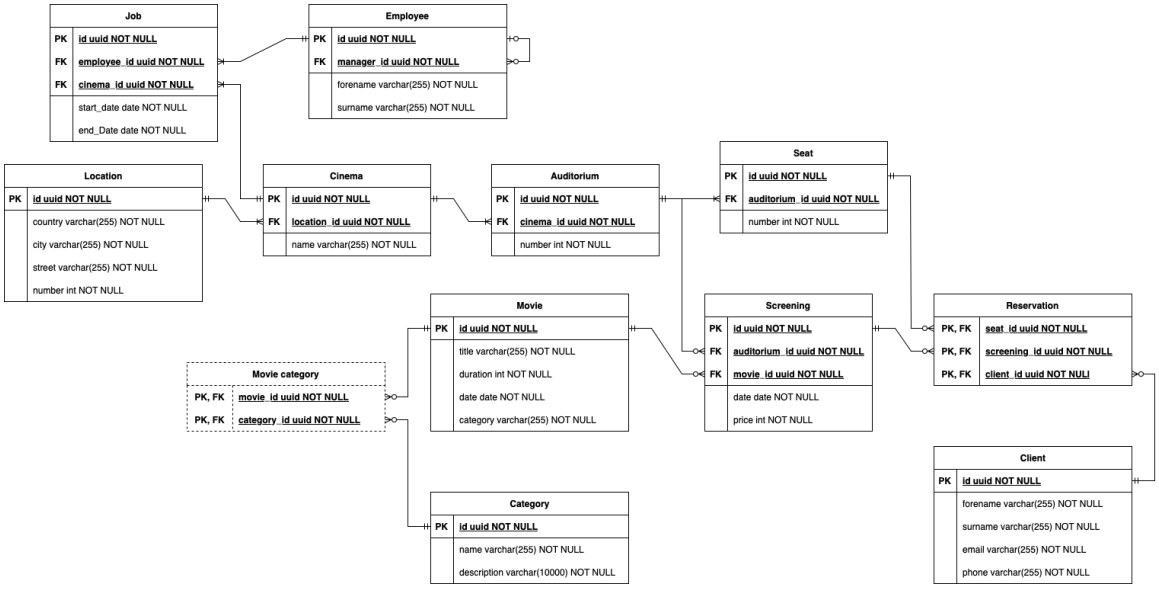

Figura 2: Diagrama entitate-relație

Aceste query-uri au fost rulate pe o instanță de Oracle Database 19, bazată pe imaginea oficială de la Oracle, dar cu un build pentru MacOS ARM local.

2 Exercițiul 3

Problema 3 din *Laborator5_PLSQL.pdf* ne cer să rezolvăm următoarea cerință: *Definiţi un pachet cu ajutorul căruia să se obţină salariul maxim înregistrat pentru salariaţii care lucrează într-un anumit oraş şi lista salariaţilor care au salariul mai mare sau egal decât acel maxim. Pachetul va conţine un cursor şi un subprogram funcţie.*. Adaptând cerința la schema utilizată, obținem următoarea cerință: *Definiți un pachet cu ajutorul căruia să se obțină, pentru o anumită perioadă de timp, venitul maxim generat de un film și filmul, dar și lista filmelor care au un număr de rezervări cel puțin egal cu o valoare data. Daca valoarea nu este specificata, ea va fi 0.*

În rezolvarea acestei cerințe, am optat să adaug două tipuri de date pentru a ajuta cu prelucrearea rezultatelor, dar și un cursor auxiliar, care returnează lista cu toate filmele și veniturile generate de ele în ordine descrescătoare. Acest cursor poate fi folosit și independent.

```
CREATE OR REPLACE PACKAGE MOVIE_STATISTICS AS
    TYPE MOVIE_ID_RESERVATIONS_COUNT IS RECORD (
        MOVIE_ID NUMBER(10),
        RESERVATIONS_COUNT NUMBER(10)
    );
    TYPE MOVIE_ID_REVENUE IS RECORD (
        MOVIE_ID NUMBER(10),
        REVENUE NUMBER(10)
    );
    CURSOR MOVIES_WITH_MINIMUM_RESERVATIONS(
        START_DATE DATE,
        END_DATE DATE,
        MINIMUM_RESERVATIONS NUMBER
    ) RETURN MOVIE_ID_RESERVATIONS_COUNT;
    CURSOR MOVIES_WITH_MAX_REVENUE(
        START_DATE DATE,
        END_DATE DATE
    ) RETURN MOVIE_ID_REVENUE;

    FUNCTION MOVIE_WITH_MAX_REVENUE(
        START_DATE DATE,
        END_DATE DATE
    ) RETURN MOVIE_ID_REVENUE;
END MOVIE_STATISTICS;
/

CREATE OR REPLACE PACKAGE BODY MOVIE_STATISTICS AS
    CURSOR MOVIES_WITH_MINIMUM_RESERVATIONS(
        START_DATE DATE,
        END_DATE DATE,
        MINIMUM_RESERVATIONS NUMBER
    ) RETURN MOVIE_ID_RESERVATIONS_COUNT IS
    SELECT
        *
    FROM
        (
            SELECT
                SR.MOVIE_ID,
                SUM(RESERVATIONS) RESERVATIONS_COUNT
            FROM
                (
                    SELECT
                        S.SCREENING_ID,
                        S.MOVIE_ID,
                        COUNT(CLIENT_ID) AS RESERVATIONS
                    FROM
                        SCREENING   S
                        INNER JOIN RESERVATION R
                        ON R.SCREENING_ID = S.SCREENING_ID
                    WHERE
                        "DATE" BETWEEN START_DATE AND END_DATE
                    GROUP BY
                        S.SCREENING_ID,
                        S.MOVIE_ID,
                        S.PRICE
                ) SR
            GROUP BY
                SR.MOVIE_ID
            ORDER BY
                RESERVATIONS_COUNT
        )
    WHERE
        RESERVATIONS_COUNT >= MINIMUM_RESERVATIONS;
    CURSOR MOVIES_WITH_MAX_REVENUE(
        START_DATE DATE,
        END_DATE DATE
    ) RETURN MOVIE_ID_REVENUE IS
    SELECT
        SR.MOVIE_ID,
        SUM(SR.PARTIAL_REVENUE) AS REVENUE
    FROM
        (
            SELECT
                S.SCREENING_ID,
                S.MOVIE_ID,
                COUNT(CLIENT_ID) * S.PRICE AS PARTIAL_REVENUE
```

```
77 |              FROM
78 |                  SCREENING   S
79 |                  INNER JOIN RESERVATION R
80 |                  ON R.SCREENING_ID = S.SCREENING_ID
81 |              WHERE
82 |                  "DATE" BETWEEN START_DATE AND END_DATE
83 |              GROUP BY
84 |                  S.SCREENING_ID,
85 |                  S.MOVIE_ID,
86 |                  S.PRICE
87 |          ) SR
88 |      GROUP BY
89 |          SR.MOVIE_ID
90 |      ORDER BY
91 |          REVENUE DESC;
92 |
93 |      FUNCTION MOVIE_WITH_MAX_REVENUE(
94 |          START_DATE DATE,
95 |          END_DATE DATE
96 |      ) RETURN MOVIE_ID_REVENUE IS
97 |          MAX_REVENUE_MOVIE MOVIE_ID_REVENUE;
98 |      BEGIN
99 |          FOR MOVIE_REVENUE IN MOVIES_WITH_MAX_REVENUE(START_DATE, END_DATE) LOOP
100|              MAX_REVENUE_MOVIE := MOVIE_REVENUE;
101|              EXIT;
102|          END LOOP;
103|
104|          RETURN MAX_REVENUE_MOVIE;
105|      END MOVIE_WITH_MAX_REVENUE;
106|  END MOVIE_STATISTICS;
107|  /
```

Listing 1: Query pentru definirea pachetului

```
01 |  SET SERVEROUTPUT ON
02 |
03 |  DECLARE
04 |      START_DATE             DATE := TO_DATE('2025/09/15', 'yyyy/mm/dd');
05 |      END_DATE               DATE := TO_DATE('2025/09/30', 'yyyy/mm/dd');
06 |      RESERVATIONS_COUNT     NUMBER(10) := 10;
07 |      MOVIE_WITH_MAX_REVENUE MOVIE_STATISTICS.MOVIE_ID_REVENUE;
08 |  BEGIN
09 |      FOR MOVIE_RESERVATIONS IN MOVIE_STATISTICS.MOVIES_WITH_MINIMUM_RESERVATIONS(
       START_DATE, END_DATE, RESERVATIONS_COUNT) LOOP
10 |          DBMS_OUTPUT.PUT_LINE('Filmul '
11 |                               || MOVIE_RESERVATIONS.MOVIE_ID
12 |                               || ' are in perioada '
13 |                               || TO_CHAR(START_DATE, 'dd.mm.yyyy')
14 |                                  || ' - '
15 |                                  || TO_CHAR(END_DATE, 'dd.mm.yyyy')
16 |                                     || ' '
17 |                                     || MOVIE_RESERVATIONS.RESERVATIONS_COUNT
18 |                                     || ' rezervari.');
19 |      END LOOP;
20 |
21 |      DBMS_OUTPUT.NEW_LINE;
22 |      MOVIE_WITH_MAX_REVENUE := MOVIE_STATISTICS.MOVIE_WITH_MAX_REVENUE(START_DATE,
       END_DATE);
23 |      DBMS_OUTPUT.PUT_LINE('Filmul '
24 |                           || MOVIE_WITH_MAX_REVENUE.MOVIE_ID
25 |                           || ' a generat in perioada '
26 |                           || TO_CHAR(START_DATE, 'dd.mm.yyyy')
27 |                              || ' - '
28 |                              || TO_CHAR(END_DATE, 'dd.mm.yyyy')
29 |                                 || ' cele mai multe venituri, si anume '
30 |                                 || MOVIE_WITH_MAX_REVENUE.REVENUE
31 |                                 || '.');
32 |  END;
```

Listing 2: Query pentru testarea pachetului

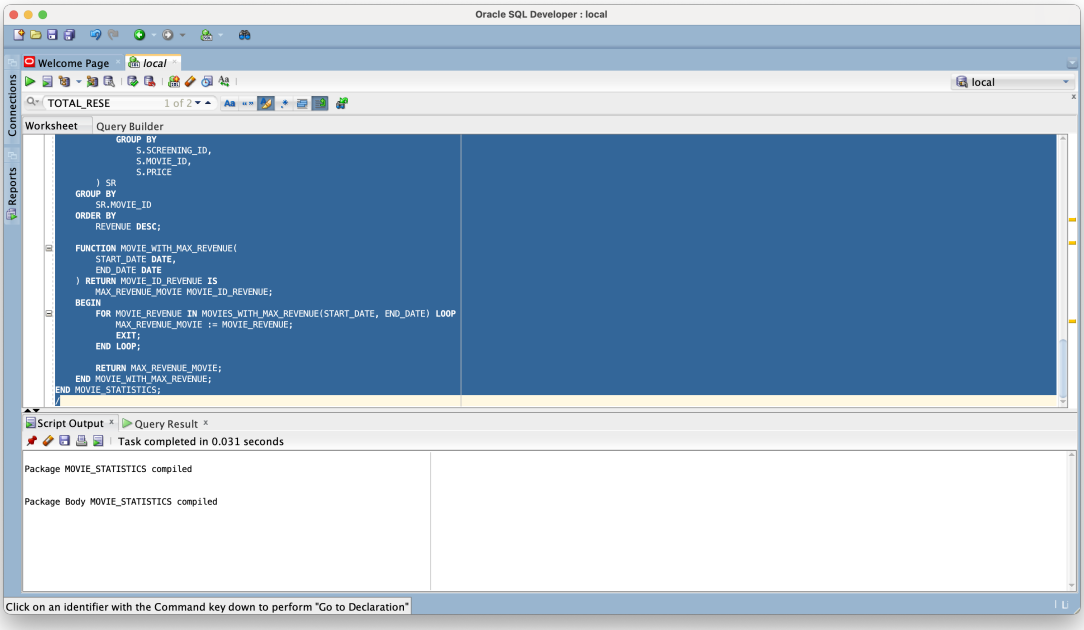

Figura 3: Rezultatul primului query

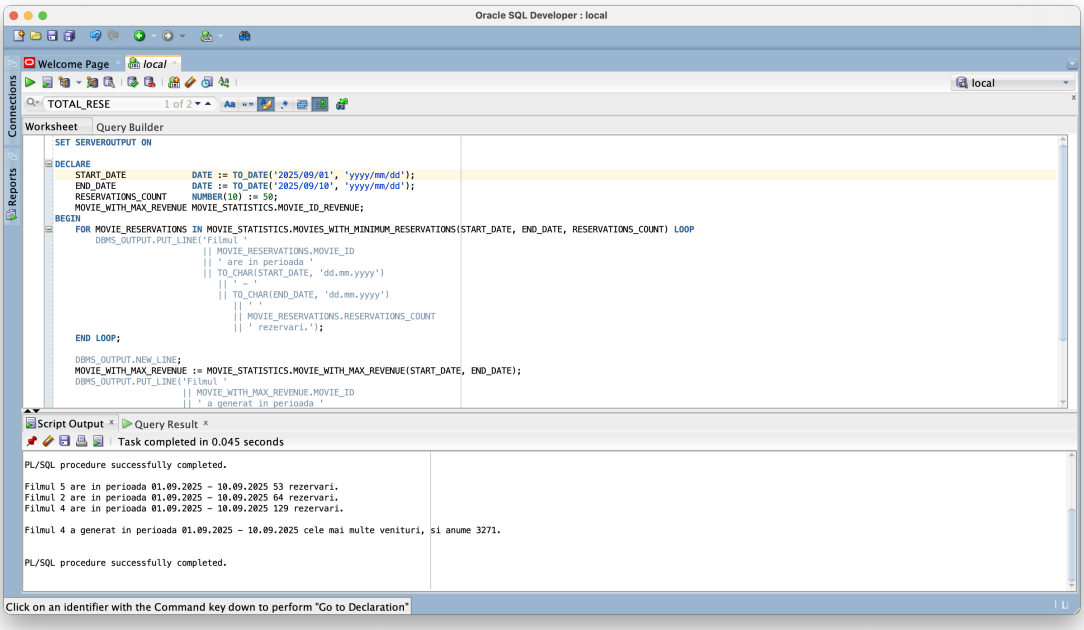

Figura 4: Rezultatul query-ului de testare, cu parametrii utilizați vizibili

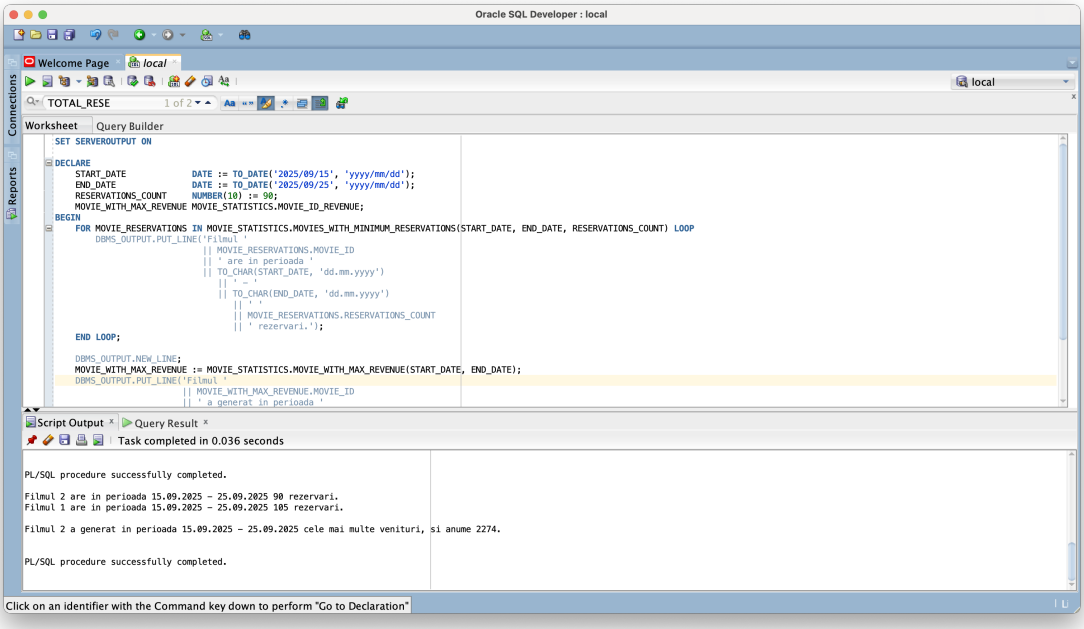

Figura 5: Rezultatul query-ului de testare, cu parametrii utilizați vizibili