Proiect –Baze de Date

Gestionarea unui cinematograf

Cerințele 14-16

Bălăiță Cosmin – Neculai

Grupa 141

Cuprins

[14. Crearea unei vizualizări complex 2](#_Toc135746077)

[15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n.. 2](#_Toc135746078)

[16. Optimizarea unei cereri 3](#_Toc135746079)

# 14. Crearea unei vizualizări complex

CREATE VIEW Vizualizare AS

SELECT f.ID\_FILM, f.TITLU, r.NUME AS NUME\_REGIZOR, a.NUME AS NUME\_ACTOR

FROM FILM f

JOIN REGIZOR r ON f.ID\_REGIZOR = r.ID\_REGIZOR

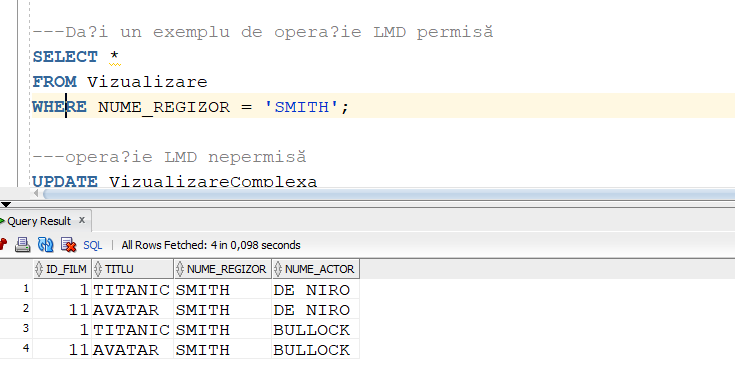
JOIN ACTOR a ON r.id\_regizor = a.id\_regizor;

--- Dați un exemplu de operație LMD permisă

SELECT \*

FROM Vizualizare

WHERE NUME\_REGIZOR = 'SMITH';

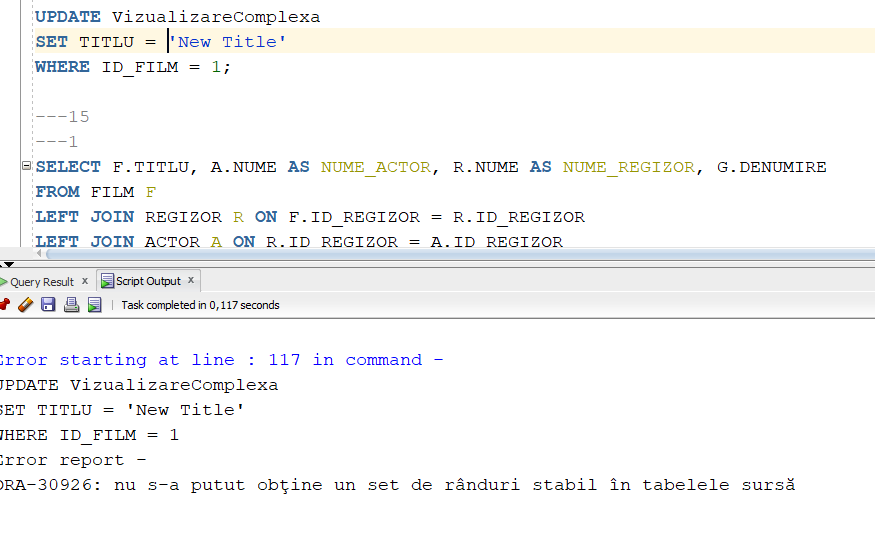


--- operație LMD nepermisă

UPDATE Vizualizare

SET TITLU = 'New Title'

WHERE ID\_FILM = 1;



Operația UPDATE nu este permisă asupra unei vizualizări complexe deoarece o vizualizare complexă poate include coloane calculate sau combinate din mai multe tabele de bază. Modificarea unei astfel de vedere ar putea duce la o actualizare incorectă sau ambiguă a datelor în tabelele de bază.

# 15. Formulați în limbaj natural și implementați în SQL: o cerere ce utilizează operația outer-join pe minimum 4 tabele, o cerere ce utilizează operația division și o cerere care implementează analiza top-n..

Limbaj natural: Selectați titlul filmului, numele actorului, numele regizorului și denumirea genului pentru toate înregistrările din tabela FILM. Inclusiv, dacă există, afișați numele regizorului și actorului, chiar dacă nu există o potrivire în tabela REGIZOR și respectiv tabela ACTOR. Asigurați-vă că afișați și denumirea genului, indiferent dacă există o potrivire în tabela GEN.

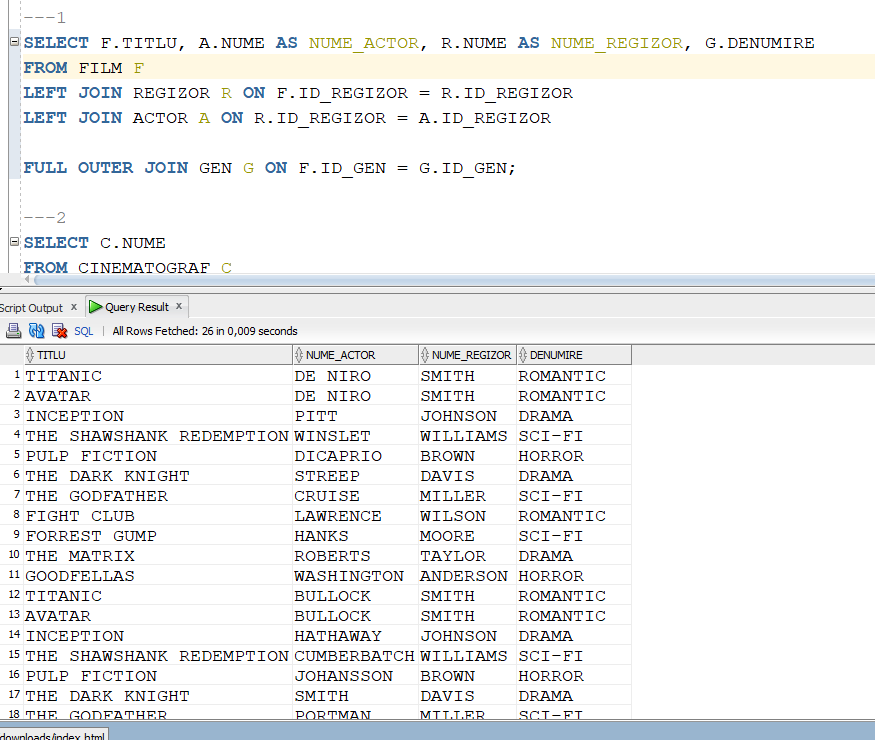
SELECT F.TITLU, A.NUME AS NUME\_ACTOR, R.NUME AS NUME\_REGIZOR, G.DENUMIRE

FROM FILM F

LEFT JOIN REGIZOR R ON F.ID\_REGIZOR = R.ID\_REGIZOR

LEFT JOIN ACTOR A ON R.ID\_REGIZOR = A.ID\_REGIZOR

FULL OUTER JOIN GEN G ON F.ID\_GEN = G.ID\_GEN;



Limbaj Natural: Selectați numele cinematografelor în care toți angajații sunt prezenți. Adică, selectați numele cinematografelor pentru care nu există niciun angajat care să lipsească."

SELECT C.NUME

FROM CINEMATOGRAF C

WHERE NOT EXISTS (

SELECT A.ID\_ANGAJAT

FROM ANGAJAT A

WHERE A.ID\_CINEMATOGRAF = C.ID\_CINEMATOGRAF

AND A.ID\_ANGAJAT NOT IN (

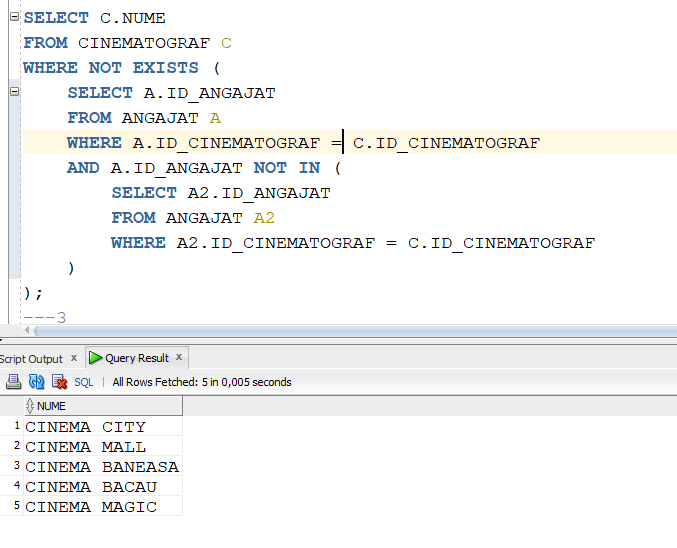
SELECT A2.ID\_ANGAJAT

FROM ANGAJAT A2

WHERE A2.ID\_CINEMATOGRAF = C.ID\_CINEMATOGRAF

)

);



Limbaj natural: Doresc sa obtine primele 2 filme cu rating cel mai mare

SELECT F.ID\_Film, F.Titlu, R.VALOARE

FROM Film F

JOIN RATING R ON F.ID\_RATING=R.ID\_RATING

ORDER BY r.valoare DESC

FETCH FIRST 2 ROWS ONLY;

# 16. Optimizarea unei cereri

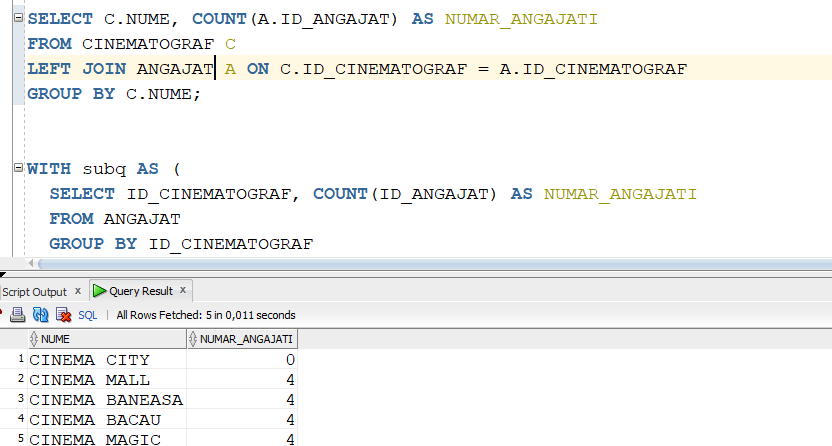
Cerere inițială:

SELECT C.NUME, COUNT(A.ID\_ANGAJAT) AS NUMAR\_ANGAJATI

FROM CINEMATOGRAF C

LEFT JOIN ANGAJAT A ON C.ID\_CINEMATOGRAF = A.ID\_CINEMATOGRAF

GROUP BY C.NUME;



Expresia algebrică corespunzătoare acestei cereri este:

γ C.NUME, COUNT(A.ID\_ANGAJAT) → NUMAR\_ANGAJATI (C ⋈ A)

Arborele algebric corespunzător acestei cereri este:

γ C.NUME, COUNT(A.ID\_ANGAJAT) → NUMAR\_ANGAJATI

|

⋈

/ \

C A

| |

ID\_C ID\_C

NUME ID\_ANGAJAT

|

NUME

În acest exemplu, o optimizare potențială ar fi adăugarea unui index pe coloana ID\_CINEMATOGRAF din ambele tabele CINEMATOGRAF și ANGAJAT, pentru a accelera operațiile de join.

Cerere optimizată:

WITH subq AS (

SELECT ID\_CINEMATOGRAF, COUNT(ID\_ANGAJAT) AS NUMAR\_ANGAJATI

FROM ANGAJAT

GROUP BY ID\_CINEMATOGRAF

)

SELECT C.NUME, COALESCE(subq.NUMAR\_ANGAJATI, 0) AS NUMAR\_ANGAJATI

FROM CINEMATOGRAF C

LEFT JOIN subq ON C.ID\_CINEMATOGRAF = subq.ID\_CINEMATOGRAF;

