Sistemi Informativi T
30 giugno 2022
Risoluzione

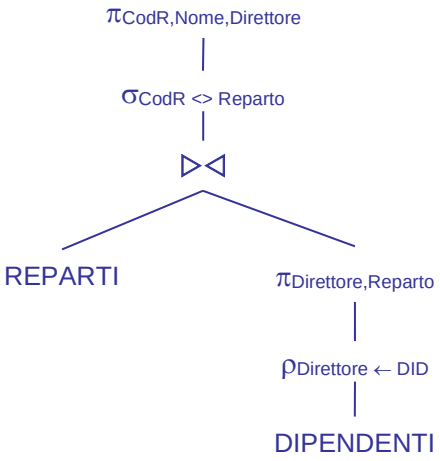
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

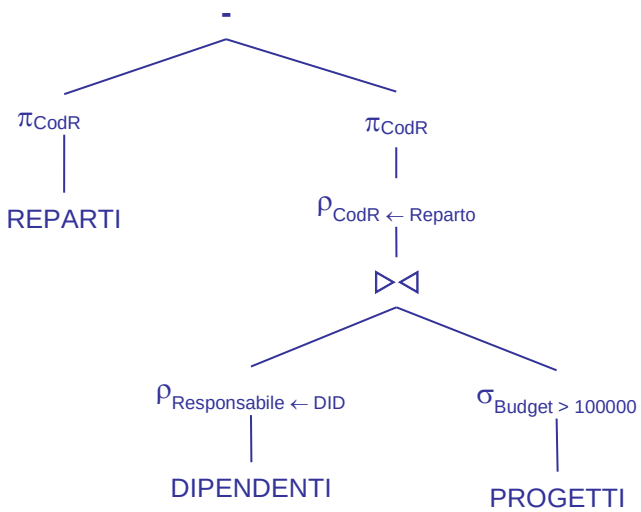
```
REPARTI(CodR,Nome,Direttore),
      Direttore REFERENCES DIPENDENTI;
DIPENDENTI(DID,Nome,Stipendio,Reparto),
      Reparto REFERENCES REPARTI;
PROGETTI(CodProg,Titolo,Budget,Responsabile),
      Responsabile REFERENCES DIPENDENTI;
-- Stipendio e Budget sono di tipo DEC(8,2).
-- Più reparti possono avere lo stesso direttore.
-- Un progetto afferisce al reparto del responsabile del progetto.
```

si esprimano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] I dati dei reparti che hanno come direttore un dipendente di un altro reparto



1.2) [2 p.] I codici dei reparti che non hanno progetti con Budget > 100000€



L'operando destro della differenza trova i reparti con almeno un progetto afferente con Budget > 100000€

Sistemi Informativi T
30 giugno 2022
Risoluzione

2) SQL (5 punti totali)

Con riferimento al DB dell'esercizio 1, si esprimano in SQL le seguenti interrogazioni:

2.1) [2 p.] Per ogni reparto la somma degli stipendi dei dipendenti, escludendo i dipendenti responsabili di progetto e/o direttori di reparto

```
SELECT   R.CodR, SUM(D.Stipendio) AS SommaStipendi
FROM     REPARTI R, DIPENDENTI D
WHERE    D.Reparto = R.CodR
AND      D.DID NOT IN ( SELECT P.Responsabile
                        FROM   PROGETTI P
                           UNION
                        SELECT R1.Direttore
                        FROM   REPARTI R1
                       )
GROUP BY R.CodR ;
```

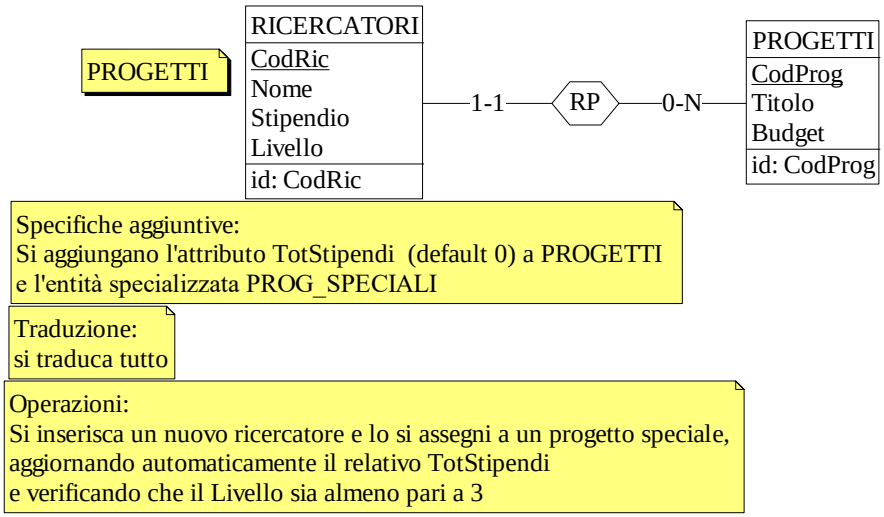
2.2) [3 p.] Il reparto che ha come direttore un dipendente di un altro reparto e per il quale la somma dei budget dei progetti ad esso afferenti è massima

```
WITH
ALTROREP(CodR) AS (
   SELECT R.CodR
   FROM   REPARTI R, DIPENDENTI D
   WHERE  R.Direttore = D.DID
   AND    D.Reparto <> R.CodR  ),
SUMBUDGET (CodR,TotBudget) AS (
   SELECT R.CodR, SUM(P.Budget)
   FROM   REPARTI R, DIPENDENTI D, PROGETTI P
   WHERE  D.Reparto = R.CodR
  AND     D.DID = P.Responsabile
  AND     R.CodR IN ( SELECT CodR
                      FROM   ALTROREP )
   GROUP BY R.CodR                          )
SELECT   S.*
FROM     SUMBUDGET S
WHERE    S.TotBudget = ( SELECT MAX(S1.TotBudget)
                         FROM   SUMBUDGET S1       );

-- La prima c.t.e. determina i reparti che hanno come direttore un
-- dipendente di un altro reparto; la seconda c.t.e. calcola la
-- somma dei budget per ognuno di questi reparti
```

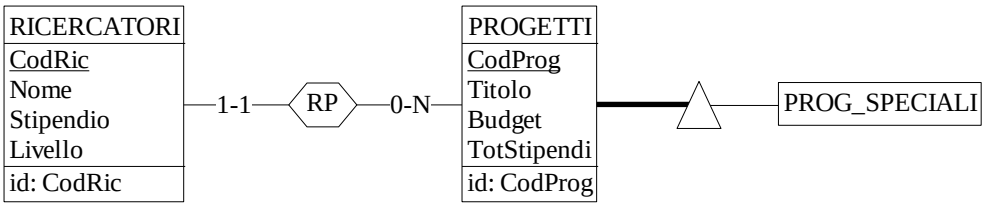
Sistemi Informativi T
30 giugno 2022
Risoluzione

3) Modifica di schema E/R e del DB (6 punti totali)

Dato il file ESE3.lun fornito, in cui è presente lo schema ESE3-input in figura:



3.1) [1 p.] Si modifichi ESE3-input secondo le Specifiche aggiuntive;



3.2) [1 p.] Si copi lo schema modificato in uno schema ESE3-tradotto. Mediante il comando Transform/Quick SQL, si traduca la parte di schema specificata, modificando lo script SQL in modo da essere compatibile con DB2 e permettere l'esecuzione del punto successivo, ed eventualmente aggiungendo quanto richiesto dalle Specifiche aggiuntive;
Si veda il relativo file .sql

3.3) [4 p.] Si scriva l'istruzione SQL che modifica il DB come da specifiche (usare valori a scelta) e si definiscano i trigger necessari.

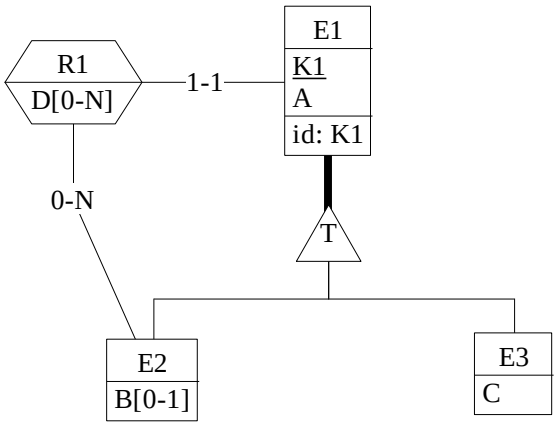
```
CREATE OR REPLACE TRIGGER CONTROLLA_LIVELLO
BEFORE INSERT ON RICERCATORI
REFERENCING NEW AS N
FOR EACH ROW
WHEN (N.Livello < 3 AND EXISTS
            ( SELECT *
              FROM   PROG_SPECIALI
              WHERE  CodProg = N.CodProg ) )
SIGNAL SQLSTATE '70001' ('Il livello deve essere almeno pari a 3!')

CREATE OR REPLACE TRIGGER AGGIORNA_TOT_STIPENDI
AFTER INSERT ON RICERCATORI
REFERENCING NEW AS N
FOR EACH ROW
UPDATE PROGETTI
SET      TotStipendi = TotStipendi + N.Stipendio
WHERE  CodProg = N.CodProg;

INSERT INTO RICERCATORI VALUES
     (:codRic, :nome, :stipendio, :livello, :codprog);
```

Sistemi Informativi T
30 giugno 2022
Risoluzione

4) Progettazione logica (6 punti totali)

Dato lo schema concettuale in figura e considerando che:

a) le entità E1, E2 ed E3 vengono tradotte insieme;
b) l'associazione R1 non viene tradotta separatamente;
c) le istanze di E1 che appartengono sia ad E2 che ad E3 non hanno nessun valore di D;



4.1) [3 p.] Si progettino gli opportuni schemi relazionali e si definiscano tali schemi mediante uno script SQL compatibile con DB2

```
-- il tipo degli attributi non è necessariamente INT

CREATE TABLE E1 (
K1          INT NOT NULL PRIMARY KEY,
A           INT NOT NULL,
K1R1        INT NOT NULL REFERENCES E1,
TIPO2       SMALLINT NOT NULL CHECK (TIPO2 IN (1,2)), -- TIPO2 = 2 se appartiene anche a E2
TIPO3       SMALLINT NOT NULL CHECK (TIPO3 IN (1,3)), -- TIPO3 = 3 se appartiene anche a E3
B           INT,
C           INT,
CONSTRAINT GERARCHIA CHECK (TIPO2 = 2 OR TIPO3 = 3),
CONSTRAINT E2 CHECK ((TIPO2 = 1 AND B IS NULL) OR (TIPO2 = 2)),
CONSTRAINT E3 CHECK ((TIPO3 = 1 AND C IS NULL) OR (TIPO3 = 3 AND C IS NOT NULL)) );

CREATE TABLE R1D (
K1          INT NOT NULL REFERENCES E1,
D           INT NOT NULL,
PRIMARY KEY (K1,D)      );
```

4.2) [3 p.] Per i vincoli non esprimibili a livello di schema si predispongano opportuni trigger che evitino **inserimenti di singole tuple non corrette**

```
CREATE TRIGGER K1R1_REFERENCES_E2
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( NOT EXISTS ( SELECT *
                    FROM   E1
                    WHERE N.K1R1 = E1.K1
                    AND    E1.TIPO2 = 2 ) )
SIGNAL SQLSTATE '70001' ('La foreign key K1R1 deve referenziare un"istanza di E2! ');

-- Il vincolo al punto c) può essere violato solo inserendo in R1D
CREATE TRIGGER PUNTO_C
BEFORE INSERT ON R1D
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS (   SELECT *
                 FROM   E1
                 WHERE N.K1 = E1.K1
                 AND    E1.TIPO2 = 2
                 AND    E1.TIPO3 = 3 ) )
SIGNAL SQLSTATE '70002' ('La tupla di E1 appartiene sia a E2 che a E3! ');
```