Sistemi Informativi T
20 gennaio 2025
Risoluzione

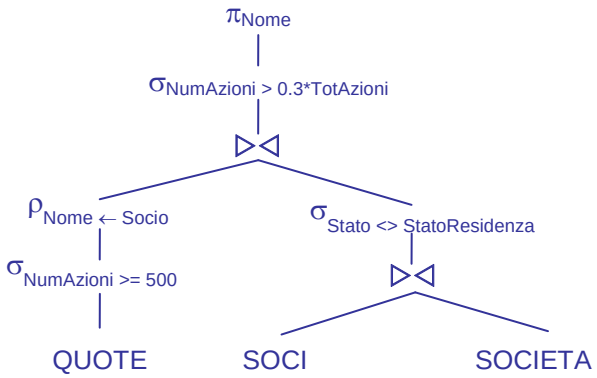
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

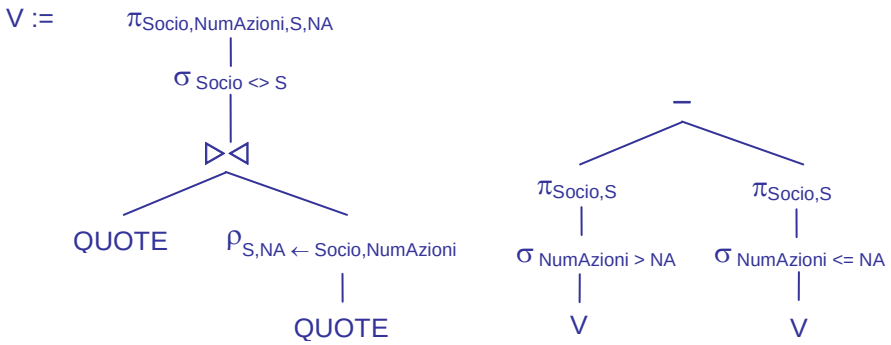
```
SOCIETA(IDSoc,TotAzioni,Stato);
SOCI(Nome,StatoResidenza);
QUOTE(IDSoc,Socio,NumAzioni),
      IDSoc references SOCIETA,
      Socio references SOCI;
-- TotAzioni e NumAzioni sono di tipo INT.
-- Per ogni società, la somma delle azioni possedute dai soci
-- (NumAzioni) è pari al valore di TotAzioni.
```

si esprimano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] I nomi dei soci che possiedono almeno 500 azioni di una società in uno stato diverso da quello in cui risiedono, e queste rappresentano più del 30% delle azioni della società



1.2) [2 p.] Le coppie di (nomi di) soci per cui, nelle società in cui hanno entrambi azioni, il primo ha sempre più azioni del secondo



La vista V contiene le coppie di soci che hanno almeno una società in comune.
L'operando sinistro della differenza seleziona tra queste coppie quelle per cui, in almeno una società, il primo (Socio) ha più azioni del secondo (S), e l'operando destro quelle in cui in almeno una società il primo NON ha più azioni del secondo (e quindi la coppia va eliminata).
E' ovviamente necessario proiettare prima di eseguire la differenza.

Sistemi Informativi T
20 gennaio 2025
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 socio, il numero di società in cui è socio singolo, includendo anche i soci che non sono mai soci singoli e ordinando il risultato per numero decrescente di società

```
SELECT   Q.Socio, COUNT(*) AS NUM_SOCIETA
FROM     SOCIETA SA, QUOTE Q
WHERE    Q.IDSoc = SA.IDSoc
AND      Q.NumAzioni = SA.TotAzioni          -- socio singolo
GROUP BY Q.Socio
 UNION
SELECT   S.Nome AS Socio, 0 AS NUM_SOCIETA
FROM     SOCI S
WHERE    S.Nome NOT IN ( SELECT  Q.Socio     -- soci mai singoli
                         FROM    SOCIETA SA, Quote Q
                         WHERE   Q.IDSoc = SA.IDSoc
                         AND     Q.NumAzioni = SA.TotAzioni )
ORDER BY NUM_SOCIETA DESC;
```

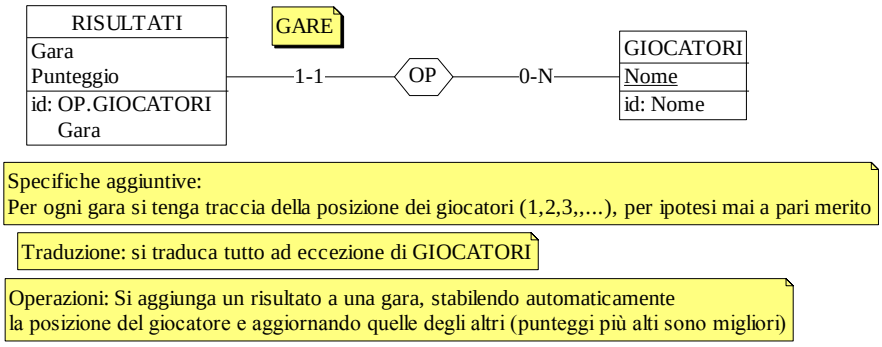
2.2) [3 p.] I dati delle società con più di un socio in cui la maggioranza delle azioni (> 50%) è posseduta da soci dello stesso stato della società

```
WITH
MAGG_INTERNA (IDSoc,TotAzioni,Stato) AS (
   SELECT SA.IDSoc, SA.TotAzioni, SA.Stato
   FROM   SOCIETA SA, SOCI S, QUOTE Q
   WHERE  SA.IDSoc = Q.IDSoc
   AND    S.Nome = Q.Socio
   AND    SA.Stato = S.StatoResidenza
   GROUP BY SA.IDSoc, SA.TotAzioni, SA.Stato
   HAVING SUM(Q.NumAzioni) > 0.5*SA.TotAzioni )
SELECT   M.*
FROM     MAGG_INTERNA M
WHERE    M.IDSoc IN ( SELECT Q.IDSoc
                      FROM   Quote Q
                      GROUP BY Q.IDSoc
                      HAVING COUNT(*) > 1 );

-- La c.t.e. determina le società, con 1 o più soci, a "maggioranza interna",
-- ovvero in cui più del 50% delle azioni sono possedute da residenti dello
-- stesso stato della società
```

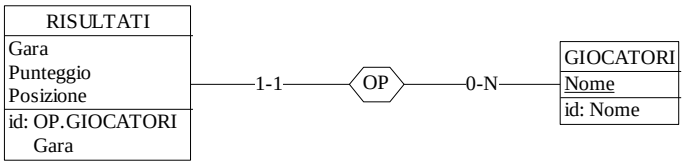
Sistemi Informativi T
20 gennaio 2025
Risoluzione

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 copi lo schema ESE3-input in uno schema ESE3-modificato e si modifichi quest'ultimo 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

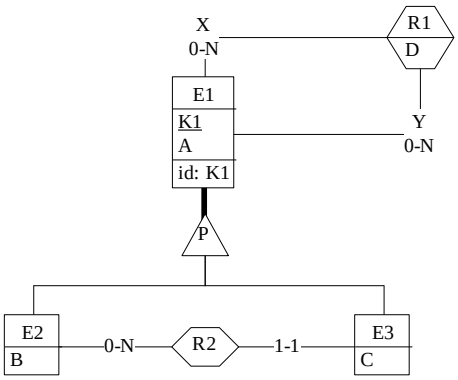
3.3) [4 p.] Si scriva l'istruzione SQL che modifica il DB come da specifiche (usare valori a scelta) e si definiscano i trigger necessari.

```
CREATE OR REPLACE TRIGGER NUOVO_RISULTATO
BEFORE INSERT ON RISULTATI
REFERENCING NEW AS N
FOR EACH ROW
SET  N.Posizione = 1 + ( SELECT COALESCE (MAX(R.Posizione),0)
                         FROM   Risultati R
                         WHERE R.Gara = N.Gara
                         AND      R.Punteggio > N.Punteggio ) ;
-- Se attualmente primo MAX(R.Posizione) è NULL

CREATE OR REPLACE TRIGGER AGGIORNA_POSIZIONI
AFTER INSERT ON RISULTATI
REFERENCING NEW AS N
FOR EACH ROW
UPDATE RISULTATI
SET        Posizione = Posizione + 1
WHERE  Gara = N.Gara
AND       Punteggio < N.Punteggio;


INSERT INTO RISULTATI(Gara,Nome,Punteggio)
VALUES (:gara,:nome giocatore,:punteggio) ;
```

Sistemi Informativi T
20 gennaio 2025
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 assieme;
b) l'associazione R2 non viene tradotta separatamente;
c) un'istanza di E2 che partecipa a R2 non può partecipare a R1 con il ruolo Y;

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,                    );
TIPO 23     SMALLINT NOT NULL CHECK (TIPO23 IN (2,3)),
B           INT,
C           INT,
K1R2        INT REFERENCES E1,
CHECK ((TIPO23 = 2 AND B IS NOT NULL AND C IS NULL AND K1R2 IS NULL) OR
          (TIPO23 = 3 AND B IS NULL AND C IS NOT NULL AND K1R2 IS NOT NULL))        );

CREATE TABLE R1 (
K1X         INT NOT NULL REFERENCES E1,
K1Y         INT NOT NULL REFERENCES E1,
D           INT NOT NULL,
PRIMARY KEY (K1X,K1Y)                              );
```

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**

```
-- Trigger che garantisce che K1R2 referenzi un'istanza di E2
CREATE OR REPLACE TRIGGER R2_E2
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( EXISTS ( SELECT * FROM E1
                WHERE N.K1R2 = E1.K1
                AND     E2.TIPO23 = 3  )   )
SIGNAL SQLSTATE '70001' ('La tupla referenzia una tupla che non appartiene a E2!');

-- Il vincolo al punto c) può essere violato inserendo in R1 oppure in E1
CREATE TRIGGER PUNTO_C_R1
BEFORE INSERT ON R1
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( EXISTS (    SELECT * FROM E1
                   WHERE N.K1Y = E1.K1R2 )       )
SIGNAL SQLSTATE '70002' ('La tupla inserita in R1 non rispetta il vincolo del punto c)! ');

CREATE TRIGGER PUNTO_C_E1
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( EXISTS (    SELECT * FROM R1
                   WHERE N.K1R2 = R1.K1Y )       )
SIGNAL SQLSTATE '70002' ('La tupla inserita in E1 non rispetta il vincolo del punto c)! ');
```