Sistemi Informativi T
24 giugno 2026
Risoluzione

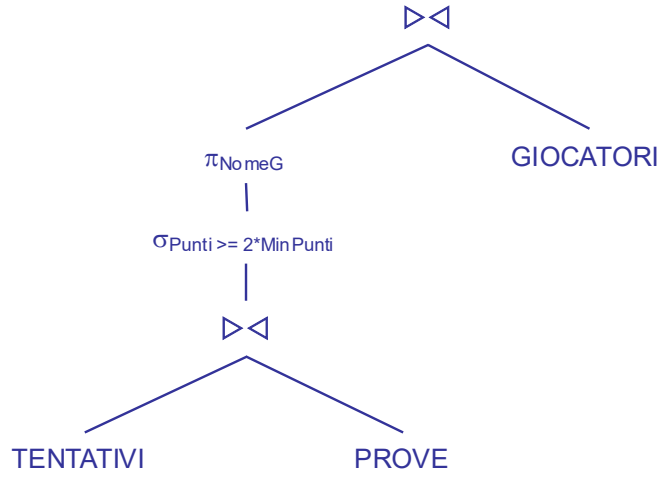
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

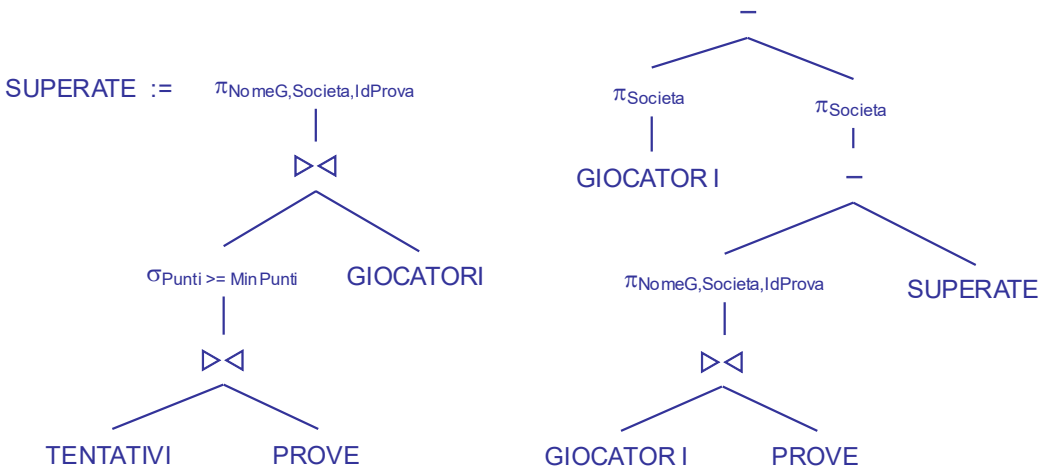
```
GIOCATORI(NomeG,Societa);
PROVE(IdProva,MinPunti);
TENTATIVI(NomeG,IdProva,Data,Punti),
      NomeG references GIOCATORI,
      IdProva REFERENCES PROVE;
-- MinPunti e Punti sono di tipo DEC(6,2).
-- Ogni volta che Punti >= MinPunti la relativa prova si intende
-- superata.
```

si esprimano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] I dati dei giocatori che hanno superato almeno una prova con un punteggio almeno pari al doppio del minimo richiesto



1.2) [2 p.] Le società in cui tutti i giocatori hanno superato almeno una volta tutte le prove



-- La vista considera solo i tentativi che hanno avuto successo e la prima differenza, in cui l'operando
-- sinistro è un prodotto Cartesiano, serve a trovare le società in cui almeno un giocatore non ha
-- superato tutte le prove.
-- La divisione si può usare per trovare i giocatori che hanno superato tutte le prove, ma non le società
-- (perché ogni società ha giocatori diversi).

Sistemi Informativi T
24 giugno 2026
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 prova il numero di giocatori che l'hanno superata al primo tentativo (il caso in cui tale numero è zero non va considerato)

```
SELECT      P.IDPROVA, COUNT(T.NOMEG) AS NUM_GIOCATORI
FROM        PROVE P, TENTATIVI T
WHERE       P.IDPROVA = T.IDPROVA
AND         T.PUNTI >= P.MINPUNTI
AND         NOT EXISTS (   SELECT  *
                           FROM    TENTATIVI T1
                           WHERE   T1.IDPROVA = T.IDPROVA
                           AND     T1.NOMEG = T.NOMEG
                           AND     T1.DATA < T.DATA )
GROUP BY P.IDPROVA ;

-- …
```

2.2) [3 p.] Per ogni prova il giocatore che l'ha superata il maggior numero di volte, riportando anche il totale dei punti ottenuti (il totale include anche i punti minori del punteggio minimo richiesto)

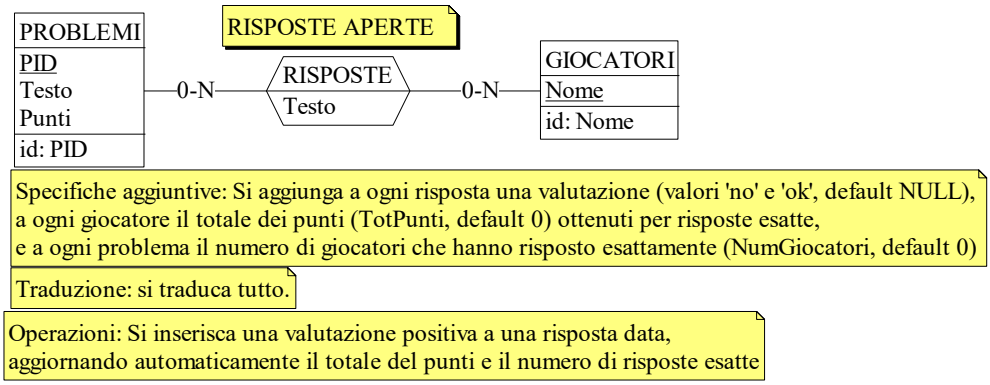
```
WITH SUPERATE(IDPROVA,NOMEG,NUMVOLTE) AS (
     SELECT  P.IDPROVA, T.NOMEG, COUNT(*)
     FROM    PROVE P, TENTATIVI T
     WHERE   P.IDPROVA = T.IDPROVA
     AND     T.PUNTI >= P.MINPUNTI
     GROUP BY P.IDPROVA, T.NOMEG  )
SELECT      S.IDPROVA, S.NOMEG, S.NUMVOLTE, SUM(T.PUNTI) AS TOTPUNTI
FROM        SUPERATE S, TENTATIVI T
WHERE       (S.IDPROVA,S.NOMEG) = (T.IDPROVA,T.NOMEG)
AND         S.NUMVOLTE  = ( SELECT MAX(S1.NUMVOLTE)
                            FROM    SUPERATE S1
                            WHERE S1.IDPROVA = S.IDPROVA )
GROUP BY S.IDPROVA, S.NOMEG, S.NUMVOLTE ;

-- La c.t.e. calcola per ogni giocatore e prova il numero di tentativi che hanno avuto successo
```

Sistemi Informativi T
24 giugno 2026
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.

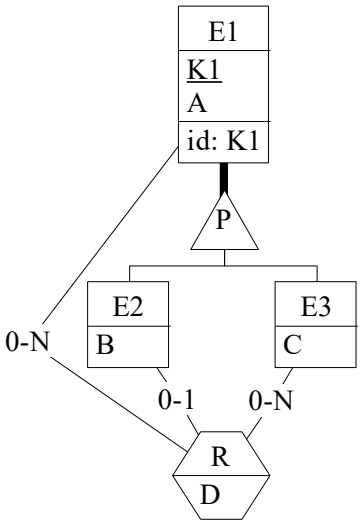
```
-- possibili anche due trigger distinti
CREATE OR REPLACE TRIGGER AGGIORNA_PUNTI_E_GIOCATORI
AFTER UPDATE OF Valutazione ON RISPOSTE
REFERENCING OLD AS O NEW AS N
FOR EACH ROW
WHEN (O.Valutazione IS NULL AND N.Valutazione = 'ok')
BEGIN ATOMIC
UPDATE GIOCATORI
SET       TotPunti = TotPunti + (  SELECT Punti
                                   FROM    PROBLEMI
                                   WHERE PID = O.PID )
WHERE  Nome = O.Nome      ;
--
UPDATE PROBLEMI
SET       NumGiocatori = NumGiocatori +1
WHERE  PID = O.PID ;
END

UPDATE     RISPOSTE
SET           Valutazione = 'ok'
WHERE      Nome = :nome
AND           PID = :pid;
```

Sistemi Informativi T
24 giugno 2026
Risoluzione

4) Progettazione logica (6 punti totali)

Dato lo schema concettuale in figura e considerando che:



a) la gerarchia viene tradotta operando un collasso verso il basso;
b) l'associazione R viene tradotta separatamente;
c) le istanze di E2 ed E3 associate tramite R hanno sempre B+C < 10;

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 E2 (
K1      INT NOT NULL PRIMARY KEY,
A       INT NOT NULL,
B       INT NOT NULL ) ;

CREATE TABLE E3 (
K1      INT NOT NULL PRIMARY KEY,
A       INT NOT NULL,
C       INT NOT NULL ) ;,

CREATE TABLE R (
K1E2   INT NOT NULL REFERENCES E2 PRIMARY KEY,
K1E3   INT NOT NULL REFERENCES E3,
D       INT NOT NULL,
K1E1E2 INT REFERENCES E2,                    -- esattamente una delle 2 foreign key è non nulla
K1E1E3 INT REFERENCES E3,
CONSTRAINT E1 CHECK ((K1E1E2 IS NOT NULL AND K1E1E3 IS NULL)
                     OR  (K1E1E2 IS NULL AND K1E1E3 IS NOT NULL))          );
```

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 garantiscono la mutua esclusione di E2 ed E3
CREATE OR REPLACE TRIGGER E2_NOT_E3
BEFORE INSERT ON E2
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS ( SELECT * FROM E3 WHERE N.K1 = E3.K1 )   )
SIGNAL SQLSTATE '70001' ('La tupla è già presente in E3!');

CREATE OR REPLACE TRIGGER E3_NOT_E2
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS ( SELECT * FROM E2 WHERE N.K1 = E2.K1 )   )
SIGNAL SQLSTATE '70002' ('La tupla è già presente in E2!');

CREATE OR REPLACE TRIGGER PUNTO_C
BEFORE INSERT ON R                           -- unico caso in cui il vincolo può essere violato
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( EXISTS ( SELECT *
                FROM    E2, E3
                WHERE N.K1E2 = E2.K1
                AND     N.K1E3 = E3.K1
                AND     E2.B + E3.C >= 10 )  )
SIGNAL SQLSTATE '70003' ('La tupla inserita viola il vincolo al punto c)!');
```