Sistemi Informativi T
3 luglio 2020
Risoluzione

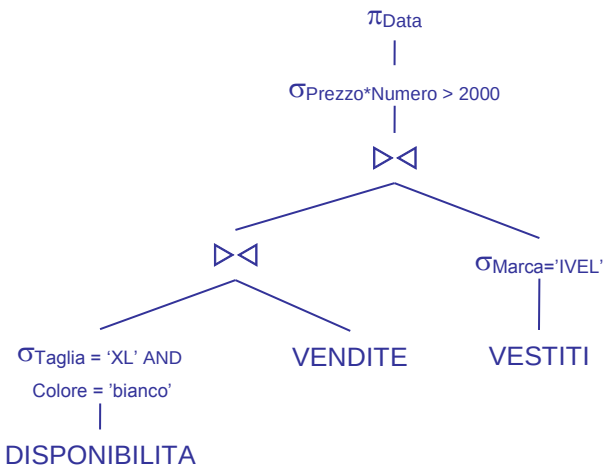
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

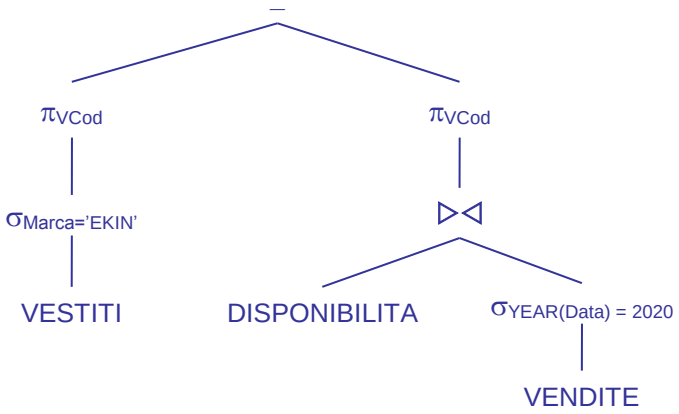
```
VESTITI(VCod,Marca,Descrizione,Prezzo);
DISPONIBILITA(TCID,VCod,Taglia,Colore,NumDisponibili),
     VCod REFERENCES VESTITI;
VENDITE(TCID,Data,Numero),
     TCID REFERENCES DISPONIBILITA;
--
-- Prezzo è di tipo DEC(6,2).
-- Taglia ha valori stringa ('XXS','XS','S','M','L',….).
-- NumDisponibili è un intero non negativo (0 = non disponibile)
-- Numero è un intero > 0 che indica il numero di vestiti
-- di un dato colore e una data taglia venduti in un dato giorno..
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] Le date in cui si sono incassati più di 2000€ per almeno un vestito di marca 'IVEL' di colore bianco e taglia 'XL'



1.2) [2 p.] I (codici di) vestiti della marca 'EKIN' che nel 2020 non hanno registrato nessuna vendita



L'operando destro della differenza è dato da tutti i codici di vestiti che nel 2020 hanno registrato almeno una vendita

Sistemi Informativi T
3 luglio 2020
Risoluzione

SQL (5 punti totali)

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

2.1) [2 p.] Per ogni vestito, la quantità complessivamente disponibile in magazzino, escludendo le combinazioni (Taglia,Colore) che nel 2020 non sono mai state vendute

```
SELECT   VS.VCod, SUM(D.NumDisponibili) AS TotDisponibili
FROM     VESTITI VS, DISPONIBILITA D
WHERE    VS.Vcod = D.VCod
AND      EXISTS ( SELECT   *
                  FROM     VENDITE VN
                  WHERE    YEAR(VN.DATA) = 2020
                  AND      VN.TCID = D.TCID )
GROUP BY VS.VCod;
```

2.2) [3 p.] Per ogni vestito, la combinazione (Taglia,Colore) che ha incassato di più nel 2019

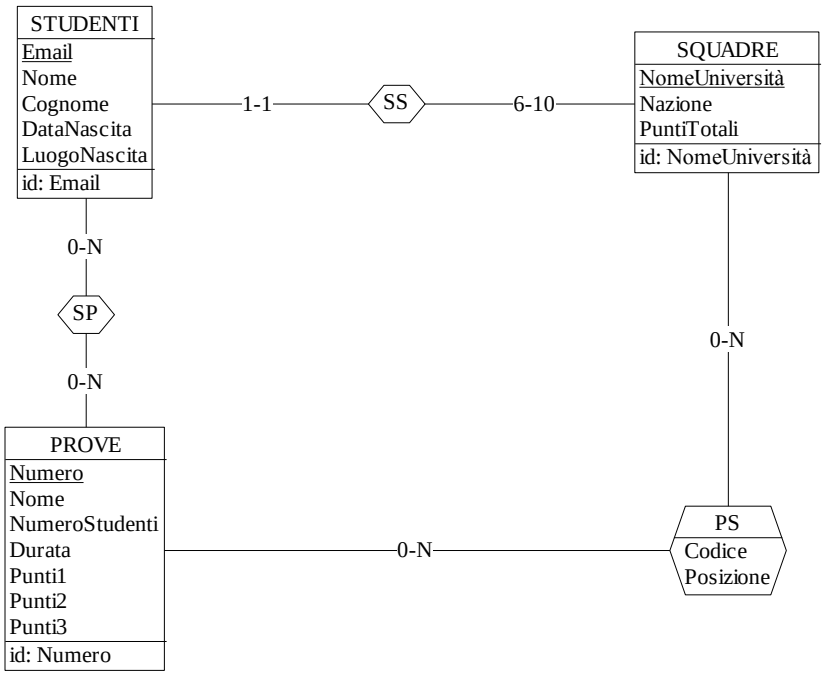
```
WITH TOT_INCASSI (VCod,Taglia,Colore,NumVendite) AS (
   SELECT D.VCod,D.Taglia,D.Colore,SUM(VN.Numero)
   FROM   DISPONIBILITA D, VENDITE VN
   WHERE  D.TCID = VN.TCID
   AND    YEAR(VN.Data) = 2019
   GROUP BY D.VCod,D.Taglia,D.Colore         )
SELECT T.*
FROM   TOT_INCASSI T
WHERE  T.NumVendite >= ALL ( SELECT T1.NumVendite
                             FROM   TOT_INCASSI T1
                             WHERE  T.VCod = T1.VCod    );

-- La c.t.e. calcola il numero di vendite nel 2019 per ogni combinazione
-- (Taglia,Colore) di ogni vestito.
-- Poiché il prezzo di un vestito non varia tra le diverse combinazioni,
-- questo è sufficiente per determinare la risposta (non c'è bisogno di
-- moltiplicare per il prezzo).
```

Sistemi Informativi T
3 luglio 2020
Risoluzione

3) Progettazione concettuale (6 punti)

Le Olimpiadi di Programmazione (OP) permettono l'iscrizione di squadre di studenti universitari da tutto il mondo. Ogni squadra, identificata dal nome della propria Università, si compone di un numero di studenti variabile, da 6 fino a un massimo di 10, per ognuno dei quali si riportano i principali dati anagrafici (nome, cognome, luogo e data di nascita, ed email identificativa). Le OP prevedono una serie di prove, ognuna caratterizzata da un tempo a disposizione e dai punteggi assegnati alle prime 3 squadre (ad es. la prova 5: "algoritmo di chiusura transitiva su un grafo orientato" ha una durata di 180 minuti, 10 punti per la prima squadra, 5 per la seconda e 2 per la terza). Per ogni prova viene anche dichiarato quanti studenti di ogni squadra possono svolgere quella prova, e quando la prova viene effettivamente svolta si registrano gli studenti che l'hanno svolta e si memorizza il codice consegnato.
Il DB delle OP fornisce inizialmente tutti i dettagli che caratterizzano le varie prove (nome, durata, punteggi, ecc.), quindi i dettagli delle squadre partecipanti, e man mano gli esiti delle diverse prove (chi l'ha svolta, la posizione ottenuta e il codice consegnato).



Commenti:

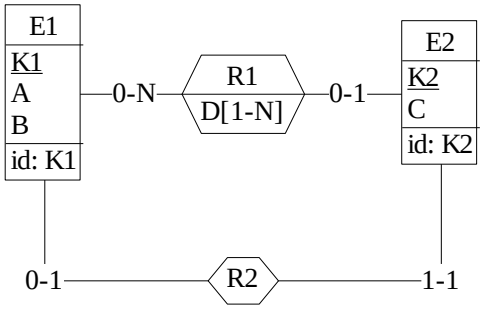
- L'esercizio, a dispetto di specifiche abbastanza articolate, è estremamente semplice. E' però importante impostare correttamente i vincoli di cardinalità minima (per permettere di inserire i dati come da specifiche) e attribuire gli attributi Codice e Posizione all'associazione PS tra SQUADRE e PROVE.
- L'attributo PuntiTotali è ridondante.
- Il vincolo che a una prova possano partecipare, per ogni squadra, un numero di studenti non superiore al massimo stabilito non è esprimibile in E/R.

Sistemi Informativi T
3 luglio 2020
Risoluzione

4) Progettazione logica (6 punti totali)

Dato lo schema concettuale in figura e considerando che:

a) tutti gli attributi sono di tipo INT;
b) nessuna associazione viene tradotta separatamente (in particolare, R2 viene tradotta con E2);
c) vale la dipendenza funzionale C → B, ovvero a valori uguali di C in due istanze di E2 corrispondono, tramite R2, due istanze di E1 con lo stesso valore di B;



4.1) [3 p.] Si progettino gli opportuni schemi relazionali e si definiscano tali schemi in DB2 (sul database SIT_STUD) mediante un file di script denominato SCHEMI.txt (o SCHEMI.sql)

```
CREATE TABLE E1 (
K1      INT NOT NULL PRIMARY KEY,
A       INT NOT NULL,
B       INT NOT NULL                    );

CREATE TABLE E2 (
K2      INT NOT NULL PRIMARY KEY,
C       INT NOT NULL,
K1R2    INT NOT NULL UNIQUE REFERENCES E1, -- UNIQUE: ogni istanza di E1 partecipa al max 1 volta
K1R1    INT REFERENCES E1               );

CREATE TABLE R1D (
K2      INT NOT NULL REFERENCES E2,
D       INT NOT NULL,
PRIMARY KEY (K2,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**, definiti in un file TRIGGER.txt (o TRIGGER.sql) e usando se necessario il simbolo '@' per terminare gli statement SQL (altrimenti ';')

```
-- Il seguente trigger garantisce che in R1D siano inserite solo tuple che riguardano un'istanza di E2 che partecipa
-- all'associazione R1
CREATE TRIGGER D
BEFORE INSERT ON R1D
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( EXISTS  (   SELECT *
                   FROM    E2
                   WHERE  N.K2 = E2.K2
                   AND     E2.K1R1 IS NULL    ) )
SIGNAL SQLSTATE '70001' ('Il valore di D inserito si riferisce a un''istanza di E2 che non partecipa a R1!');

-- Trigger che garantisce il rispetto del vincolo al punto c)
CREATE TRIGGER PUNTO_C
BEFORE INSERT ON E2
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS (     SELECT * FROM E2, E1 E1X, E1 E1Y
                   WHERE N.C = E2.C                          -- stesso valore di C…
                   AND     E2.K1R2 = E1X.K1
                   AND     N.K1R2 = E1Y.K1
                   AND     E1X.B <> E1Y.B) )                 -- …ma diverso valore di D
SIGNAL SQLSTATE '70002' ('La tupla inserita non rispetta la dipendenza funzionale da C ad B!');

-- L'inserimento di una tupla in E2 che partecipa a R1 deve prevedere l'inserimento di almeno una tupla in R1D.
-- E' quindi necessario definire una transazione.
```