Sistemi Informativi T
23 gennaio 2017
Risoluzione

Tempo a disposizione: 2:30 ore

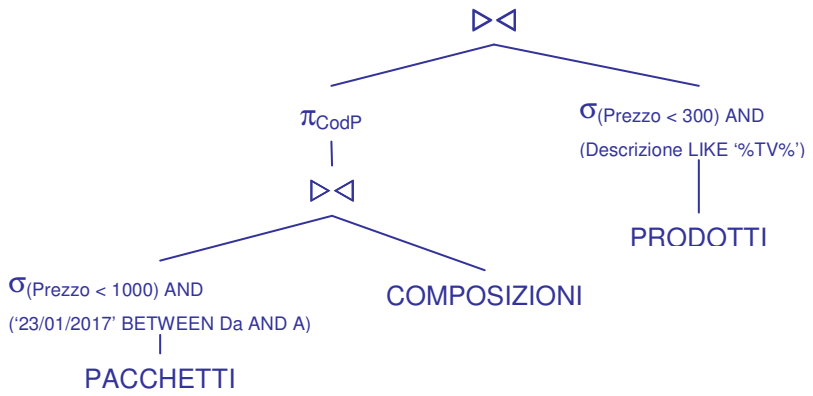
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

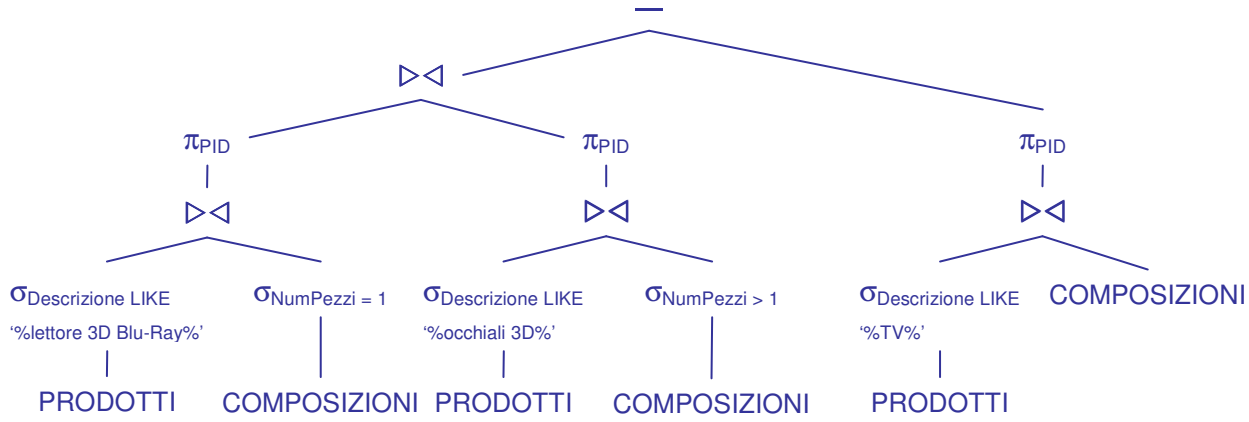
```
PRODOTTI(CodP,Descrizione,Prezzo);
PACCHETTI(PID,Prezzo,Da,A);
COMPOSIZIONI(PID,CodP,NumPezzi),
      PID REFERENCES PACCHETTI,
      CodP REFERENCES PRODOTTI;
--
-- Ogni pacchetto (o "bundle") consiste di uno o più prodotti,
-- ciascuno in quantità specificata da NumPezzi.
-- Il Prezzo di un pacchetto è sempre inferiore a quello che si
-- pagherebbe comprando i vari prodotti separatamente.
-- Da e A rappresentano il periodo temporale in cui un certo pacchetto
-- è disponibile.
-- Tutti i prezzi sono in formato DEC(6,2).
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) **[1 p.]** I dati completi dei prodotti con prezzo minore di 300€ nella cui descrizione compare il termine 'TV' e che sono inclusi in un pacchetto di prezzo minore di 1000€ disponibile il 23/01/2017



1.2) **[2 p.]** I codici dei pacchetti che non includono una 'TV' e che hanno un solo 'lettore 3D Blu-Ray' e almeno 2 paia di 'occhiali 3D'



Sistemi Informativi T
23 gennaio 2017
Risoluzione

SQL (5 punti totali)

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

2.1) **[2 p.]** I codici dei prodotti il cui prezzo è sempre almeno il 50% del prezzo di ogni pacchetto in cui sono inclusi (non vanno considerati eventuali prodotti non inclusi in nessun pacchetto)

```
SELECT   P.CodP
FROM     PRODOTTI P
WHERE    NOT EXISTS( SELECT *
                     FROM  PACCHETTI B, COMPOSIZIONI C
                     WHERE B.PID = C.PID
                     AND   C.CodP = P.CodP
                     AND   P.Prezzo < 0.5*B.Prezzo)
AND      EXISTS (    SELECT *
                     FROM  COMPOSIZIONI C
                     WHERE C.CodP = P.CodP )

-- La seconda subquery elimina i prodotti che non sono inclusi in
-- nessun pacchetto
```

2.2) **[3 p.]** Il pacchetto, con relativo prezzo, per cui il risparmio percentuale, calcolato rispetto all'acquisto dei prodotti che lo compongono, è massimo

```
WITH RISPARMI (PID, Prezzo, RisparmioPerc) AS (
   SELECT B.PID,B.Prezzo, (1 - B.Prezzo/SUM(P.Prezzo*C.NumPezzi))*100
   FROM  PRODOTTI P, PACCHETTI B, COMPOSIZIONI C
   WHERE P.CodP = C.CodP
   AND   C.PID = B.PID
   GROUP BY    B.PID,B.Prezzo    )

SELECT   R.*
FROM     RISPARMI R
WHERE    R.RisparmioPerc = (     SELECT MAX(R1.RisparmioPerc)
                                 FROM   RISPARMI R1 )

-- La c.t.e. calcola, per ogni pacchetto, il risparmio relativo.
-- Ovviamente si può evitare di moltiplicare per 100, in particolare se
-- il risparmio non viene prodotto in output
```

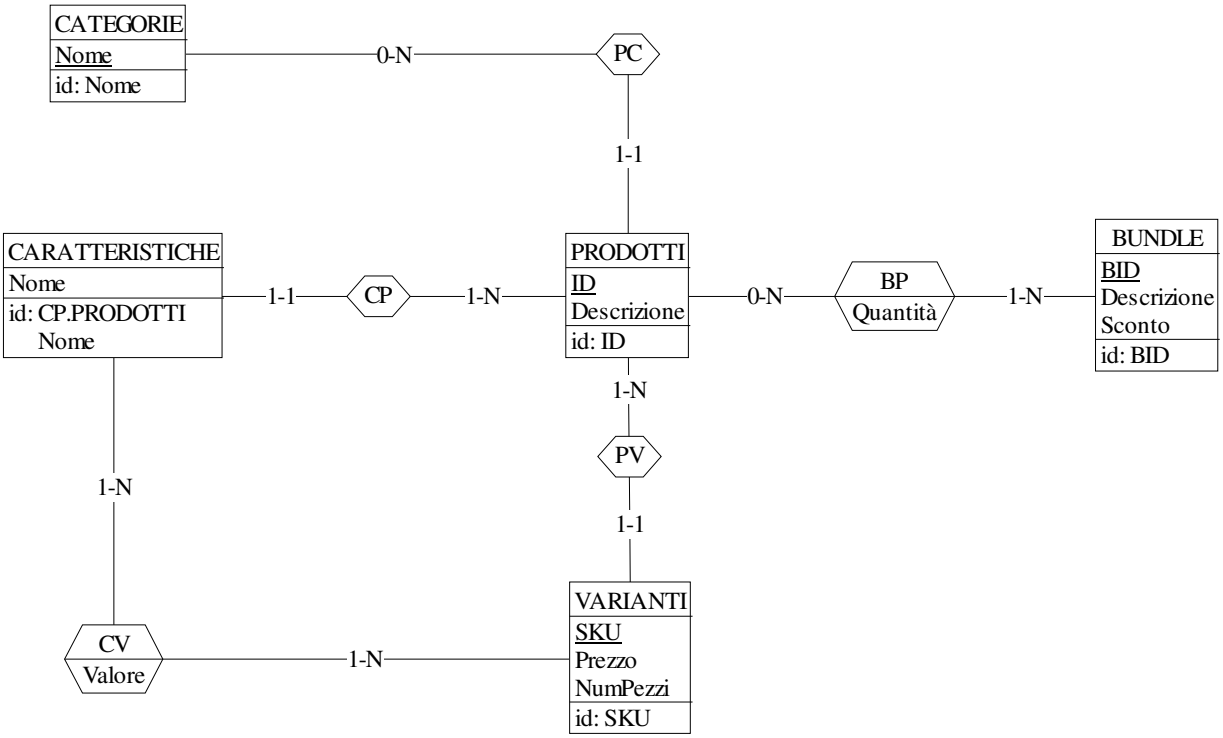
Sistemi Informativi T
23 gennaio 2017
Risoluzione

3) Progettazione concettuale (6 punti)

Il sito MZN.com vende prodotti di diverse categorie (le categorie sono predefinite dal sistema). Ogni prodotto ha una descrizione ed è disponibile in una o più varianti, ognuna caratterizzata da un identificatore univoco su MZN detto SKU*. Varianti di uno stesso prodotto possono differire per il prezzo e per la disponibilità in magazzino (numero di pezzi), oltre che per altre caratteristiche (ad es. colore, peso, taglia, ecc.). Per ogni prodotto sono definite una o più caratteristiche che differenziano le sue varianti, e ogni variante ha quindi dei valori specifici per queste caratteristiche.

MZN offre anche la possibilità di acquistare dei "bundle", ovvero combinazioni di prodotti. Ogni bundle consiste di uno o più prodotti, ognuno previsto in una certa quantità. Il prezzo del bundle è calcolato dinamicamente dal sistema applicando una percentuale di sconto (variabile da bundle a bundle) al prezzo che si otterrebbe acquistando separatamente i vari prodotti, e scegliendo per ognuno di questi una specifica variante.

* SKU = Stock Keeping Unit, standard comunemente usato per gestire l'inventario di magazzino dei prodotti

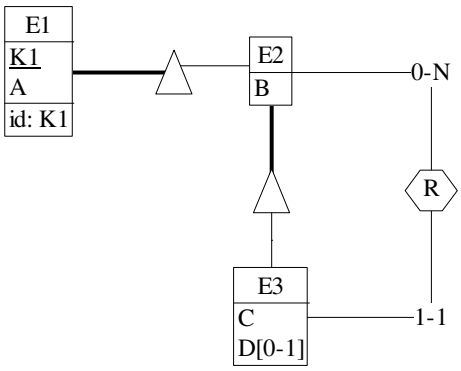


Commenti:

- La particolarità dell'esercizio sta nella necessità di gestire proprietà (CARATTERISTICHE) e valori per le stesse.
- Il vincolo che una variante possa avere dei valori per le sole caratteristiche definite per il relativo prodotto non è esprimibile in E/R.
- Le cardinalità minime possono in diversi casi essere poste a 0 per motivi legati alla dinamica del DB (inserimento di dati non transazionale)
- Per l'entità CARATTERISTICHE è possibile anche optare (solo) per un'identificazione interna e prevedere specifiche sui valori ammissibili (si veda anche la soluzione del 18/01/2016)

Sistemi Informativi T
23 gennaio 2017
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) le entità E1 ed E2 vengono tradotte assieme, e separatamente da E3;
c) l'associazione R non viene tradotta separatamente;

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

```
CREATE TABLE E1 (
K1  INT NOT NULL PRIMARY KEY,
A   INT NOT NULL,
TIPO2 SMALLINT NOT NULL CHECK (TIPO2 IN (1,2)),        -- 2 se istanza anche di E2
B   INT,
CONSTRAINT E2 CHECK (  (TIPO2 = 1 AND B IS NULL) OR (TIPO2 = 2 AND B IS NOT NULL) ) );

CREATE TABLE E3 (
K1  INT NOT NULL PRIMARY KEY REFERENCES E1,
C   INT NOT NULL,
D   INT,
K1R INT NOT NULL REFERENCES E1);
```

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 e usando se necessario il simbolo '@' per terminare gli statement SQL (altrimenti ';')

```
-- Trigger che garantisce che una tupla di E3 referenzi, tramite R, una tupla di E2

CREATE TRIGGER R_E2
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( NOT EXISTS (       SELECT * FROM E1
                          WHERE N.K1R = E1.K1
                          AND E1.TIPO2 = 2                ) )
SIGNAL SQLSTATE '70001' ('La tupla deve referenziare una tupla di E2!');

-- Trigger che garantisce che una tupla di E3 sia anche istanza di E2, e non solo di E1

CREATE TRIGGER PUNTO_D
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN ( NOT EXISTS (       SELECT * FROM E1
                          WHERE N.K1 = E1.K1
                          AND E1.TIPO2 = 2                ) )
SIGNAL SQLSTATE '70002' ('La tupla inserita deve essere un''istanza di E2!');

-- Se in E1 si inserisce anche un selettore per E3 (TIPO3, che vale 3 se l'istanza è anche di E3)
-- allora bisogna verificare nel trigger che sia TIPO3 = 3 anziché TIPO2 = 2.
-- Vanno inoltre opportunamente vincolati in E1 i valori di TIPO2 e TIPO3. In alternativa si può anche
-- definire un unico selettore

-- NB: volendo, i due trigger si possono riunire in uno solo (ponendo in OR le condizioni), anche se così facendo
-- non risulta poi immediato capire quale vincolo è stato violato
```