Sistemi Informativi T
12 gennaio 2010
Risoluzione

Tempo a disposizione: 2:30 ore

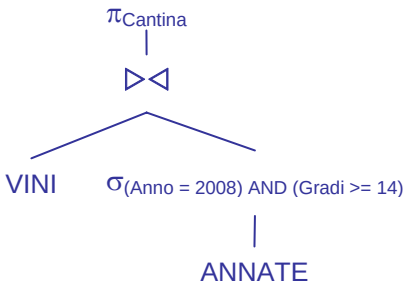
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

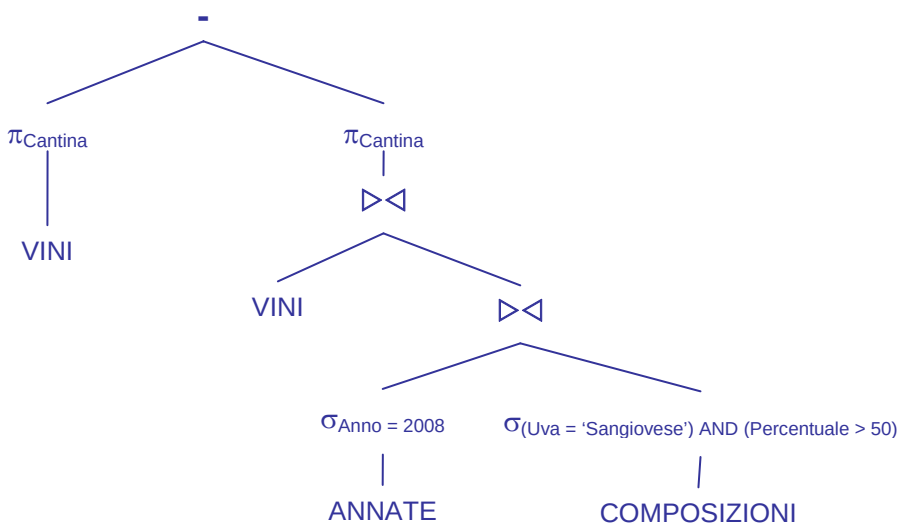
VINI(<u>VId</u>, Nome, Cantina);
COMPOSIZIONI(<u>VId</u>, <u>Uva</u>, Percentuale), VId references VINI;
ANNATE(<u>VId</u>, <u>Anno</u>, Gradi, NBottiglie), VId references VINI;

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] Le cantine che nel 2008 hanno prodotto almeno un vino con almeno 14 gradi



1.2) [2 p.] Le cantine che nel 2008 non hanno prodotto nessun vino nella cui composizione figuri l'uva Sangiovese in percentuali maggiori del 50% (50% corrisponde a Percentuale = 50)



Si noti che l'operando sinistro della differenza NON deve includere il join con ANNATE e la selezione Anno = 2008. Così facendo si eliminerebbero dal risultato le cantine che non hanno prodotto vini nel 2008, che invece devono essere restituite

Sistemi Informativi T
12 gennaio 2010
Risoluzione

2) **SQL (5 punti totali)**

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

2.1) [2 p.] Le uve che figurano in almeno un vino con più di 13 gradi e prodotto in un anno in meno di 1.000 bottiglie

```
SELECT DISTINCT Uva
FROM COMPOSIZIONI
WHERE VId IN     ( SELECT VId
                   FROM ANNATE
                   WHERE Gradi > 13 AND NBottiglie < 1000 )
```

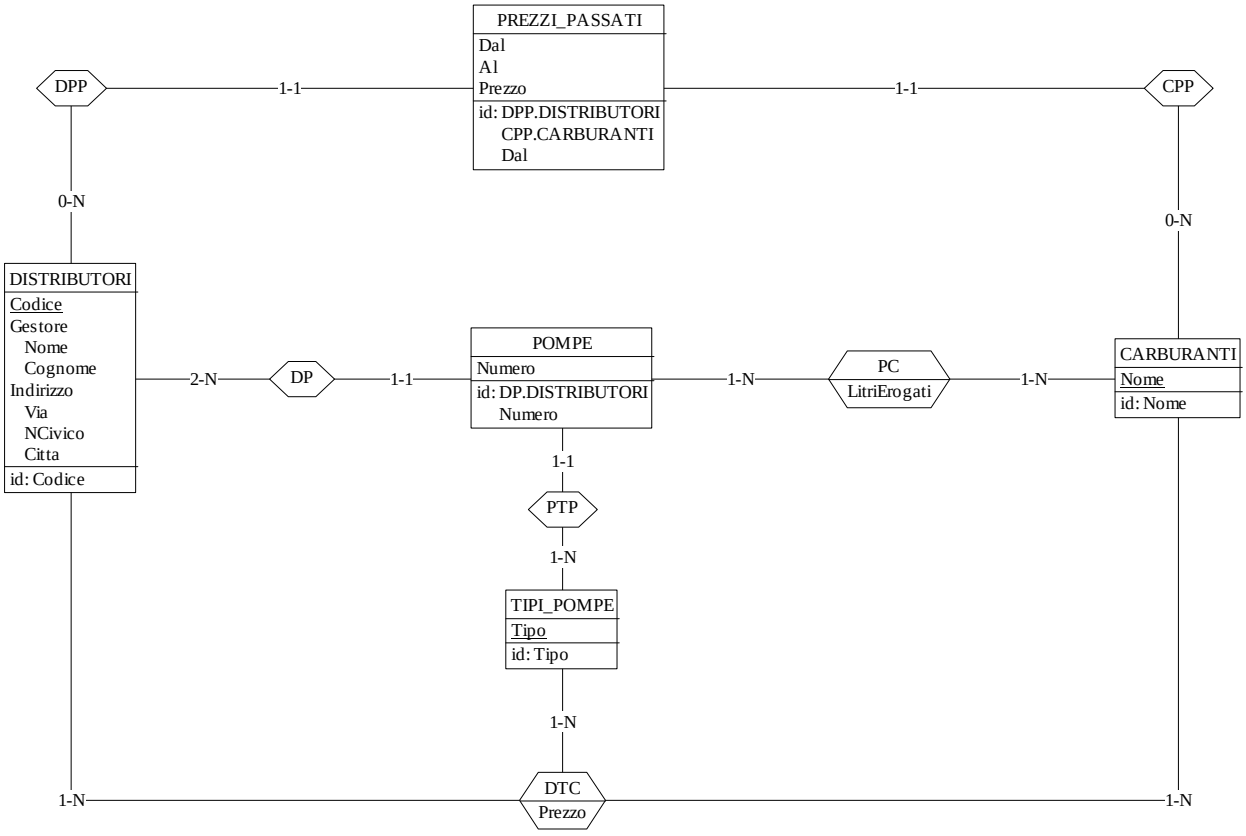
2.2) [3 p.] Per ogni cantina che complessivamente nel 2008 ha prodotto più di 100.000 bottiglie, l'elenco dei nomi dei vini in cui la percentuale di Sangiovese supera il 50%. Il risultato va ordinato per cantina, e quindi per percentuale decrescente di Sangiovese

```
SELECT V.Cantina,V.Nome, C.Percentuale
FROM VINI V JOIN COMPOSIZIONI C ON (V.Vid = C.VId)
WHERE C.Uva = 'Sangiovese' AND C.Percentuale > 50
    AND V.Cantina IN         ( SELECT V1.Cantina
                               FROM ANNATE A,VINI V1
                               WHERE A.Vid = V1.VId
                                   AND A.Anno = 2008
                               GROUP BY V1.Cantina
                               HAVING SUM(A.NBottiglie) > 100000 )
ORDER BY V.Cantina, C.Percentuale DESC
```

Sistemi Informativi T
12 gennaio 2010
Risoluzione

3) Progettazione concettuale (6 punti)

La catena di distributori JQ necessita di un sistema informativo che registri tutte le informazioni sulle vendite di carburanti. Ogni distributore (identificato da un codice e caratterizzato da un indirizzo e dal nome del gestore) ha diverse pompe (almeno 2), ognuna delle quali ha un numero che la distingue dalle altre dello stesso distributore ed è o di tipo self-service o con servizio di un addetto. Ogni pompa eroga uno o più tipi di carburante (Benzina, Gasolio, Gasolio Plus, ecc.). Il prezzo attuale di un dato carburante varia da un distributore all'altro, e dipende ovviamente anche dal tipo di pompa (self-service o con servizio). Il sistema della JQ deve tener traccia dei litri complessivi erogati da ciascuna pompa per ogni tipo di carburante. Per analisi di tipo statistico, il sistema mantiene anche, per ogni distributore e tipo di carburante, l'archivio dei prezzi medi passati (ad es. il distributore 025 ha venduto il Gasolio a 1.075 € dal 12/12/2009 al 23/12/2009).



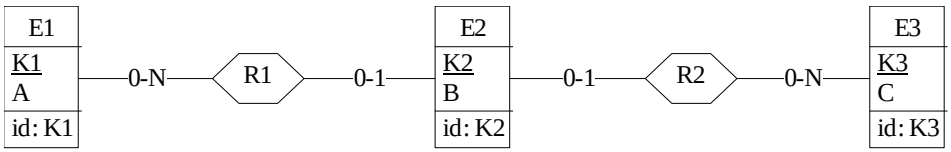
Commenti

- I tipi di pompe vengono rappresentati come entità al fine di modellare correttamente il vincolo sui prezzi correnti
- Modellando l'entità PREZZI_PASSATI come associazione si perderebbe l'univocità del prezzo in un dato periodo

Sistemi Informativi T
12 gennaio 2010
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 associazioni R1 e R2 non vengono tradotte separatamente;
c) un'istanza di E1 non è mai associata, tramite R1 e R2, a un'istanza di E3 con C = A;

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    );

CREATE TABLE E3(
K3 INT NOT NULL PRIMARY KEY,
C INT NOT NULL    );

CREATE TABLE E2(
K2 INT NOT NULL PRIMARY KEY,
B INT NOT NULL,
K1 INT REFERENCES E1,
K3 INT REFERENCES E3     );
```

4.2) [3 p.] Per i vincoli non esprimibili a livello di schema si predispongano opportuni **trigger che evitino inserimenti di tuple non corrette**, definiti in un file TRIGGER.txt e usando il simbolo '@' per terminare gli statement SQL

```
CREATE TRIGGER INS_E2
NO CASCADE BEFORE INSERT ON E2
REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN ( EXISTS (    SELECT *
                   FROM E1, E3
                   WHERE E1.K1 = N.K1 AND E3.K3 = N.K3 AND E1.A = E3.C ))
SIGNAL SQLSTATE '70001' ('Inserimento non permesso')@
```