Sistemi Informativi T
10 febbraio 2011
Risoluzione

Tempo a disposizione: 2:30 ore

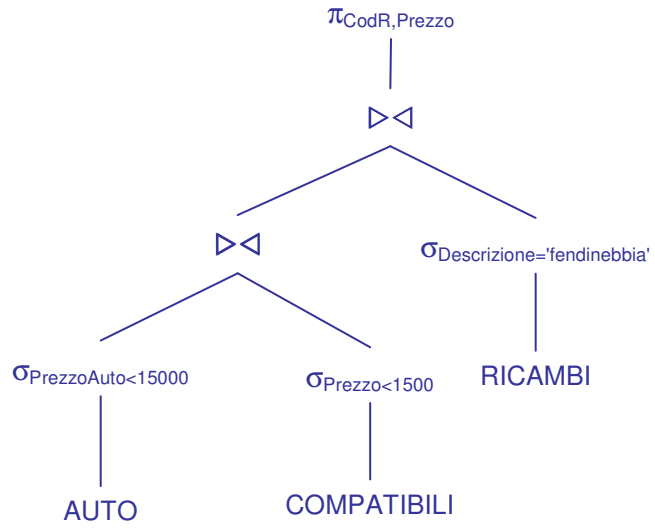
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

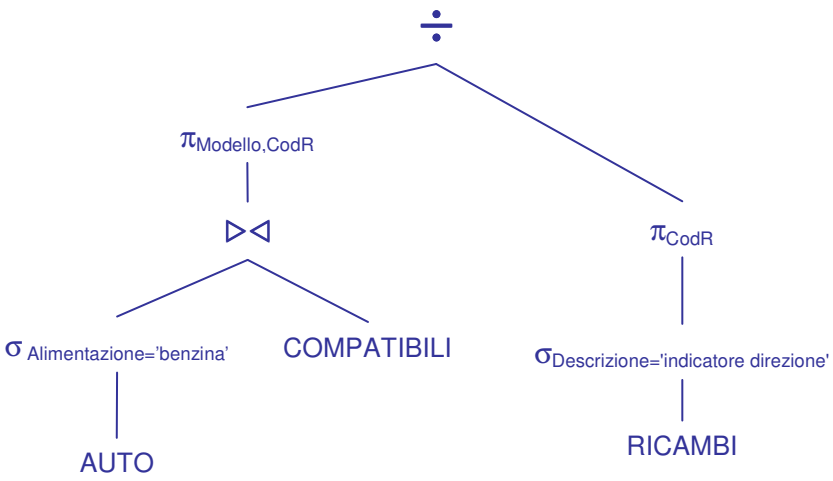
```
RICAMBI(CodR,Descrizione);
AUTO(Modello,Cilindrata,Alimentazione,PrezzoAuto);
COMPATIBILI(CodR,Modello,Prezzo),
CodR references RICAMBI, Modello references AUTO;
-- sia PrezzoAuto che Prezzo sono interi
-- per ipotesi ogni auto ha almeno un pezzo di ricambio
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) **[1 p.]** I pezzi di ricambio di tipo 'fendinebbia' che costano meno di 1500 € e che sono montabili su auto che costano meno di 15000 €



1.2) **[2 p.]** Le auto a benzina che possono montare qualsiasi ricambio di tipo 'indicatore direzione'



Sistemi Informativi T
10 febbraio 2011
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 auto a benzina che possono montare almeno due ricambi di tipo 'indicatore direzione'

```
SELECT   A.Modello
FROM     AUTO A, COMPATIBILI C, RICAMBI R
WHERE    A.Modello = C.Modello
AND      C.CodR = R.CodR
AND      A.Alimentazione = 'benzina'
AND      R.Descrizione = 'indicatore direzione'
GROUP BY A.Modello
HAVING   COUNT(*) > 1
```

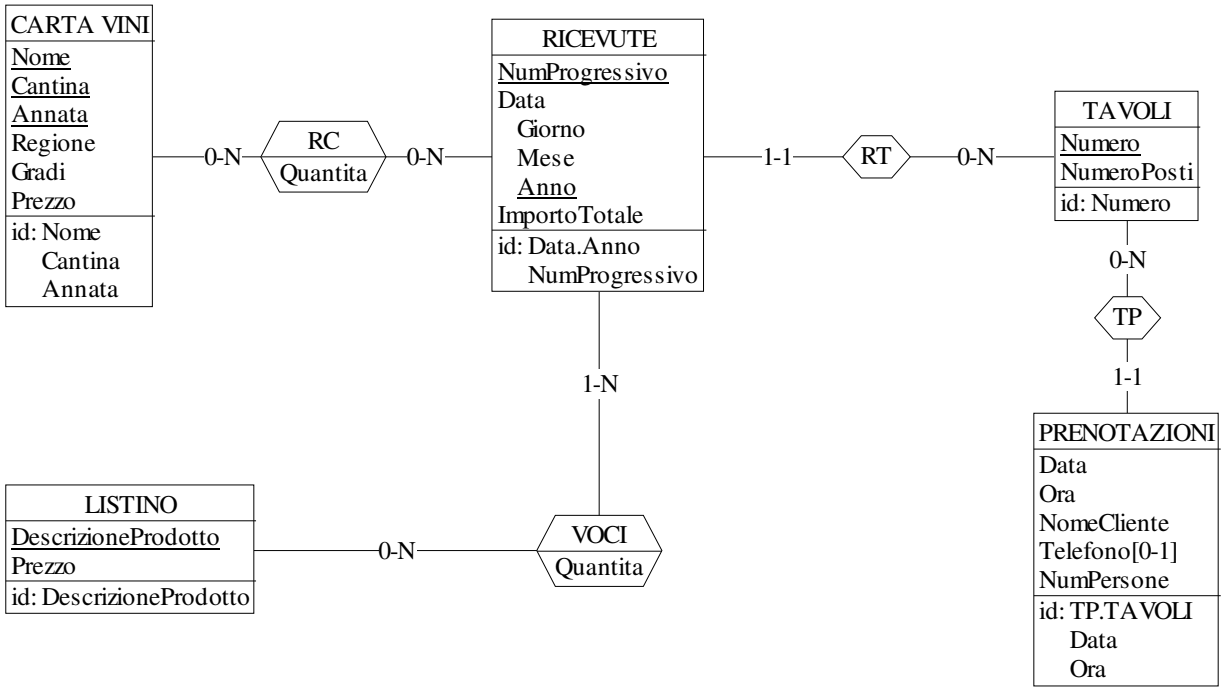
2.2) [3 p.] Per ogni modello di auto, il prezzo totale dei ricambi disponibili, scegliendo solo quello di prezzo minimo nel caso di ricambi dello stesso tipo

```
WITH PREZZIMIN(Modello,TipoRicambio,PrezzoMin) AS (
   SELECT C.Modello, R.Descrizione, MIN(C.Prezzo)
   FROM   COMPATIBILI C, RICAMBI R
   WHERE  C.CodR = R.CodR
   GROUP BY  C.Modello, R.Descrizione )
SELECT P.Modello, SUM(P.PrezzoMin) AS PrezzoTotale
FROM PREZZIMIN P
GROUP BY P.Modello
-- La Common Table Expression restituisce, per ogni modello e tipo di ricambio
-- disponibile, il prezzo minimo. A questo punto e' sufficiente sommare per
-- ogni modello tali prezzi
```

Sistemi Informativi T
10 febbraio 2011
Risoluzione

3) Progettazione concettuale (6 punti)

Il ristorante PappaBuona ha deciso di automatizzare la gestione delle ricevute e delle prenotazioni.
Una ricevuta, identificata univocamente da un numero progressivo per ogni anno, è caratterizzata da una data, un importo totale, il numero del tavolo e da una serie di voci. Ogni voce si riferisce a un elemento del listino prezzi (ovvero descrizione del prodotto e prezzo unitario) e ha anche una quantità (es. 2 tagliatelle al ragù, prezzo unitario 7.50 €). Opzionalmente, una ricevuta può anche contenere riferimenti alla carta dei vini, dove per ogni vino si riportano il nome, la cantina e l'annata (congiuntamente univoci), la regione di provenienza, la gradazione alcoolica e il prezzo.
Ogni prenotazione riporta il nome del cliente, un eventuale riferimento telefonico, il numero di persone e la data e l'orario previsto di arrivo. Ad ogni prenotazione viene associato un tavolo (ogni tavolo ha un numero identificativo e un numero di posti a sedere). Non possono esistere evidentemente due prenotazioni per lo stesso tavolo nello stesso giorno e alla stessa ora. Per esigenze di gestione della sala, è possibile che il tavolo assegnato (e che figura quindi in ricevuta) differisca da quello prenotato.

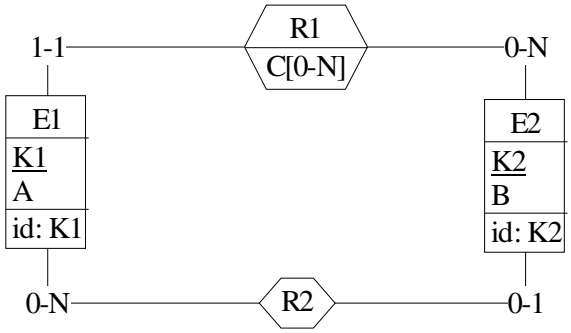


Commenti:

- L'esercizio non presenta difficoltà di rilievo
- Il vincolo relativo al numero di persone presenti in una prenotazione, che deve essere non superiore al numero di posti del tavolo riservato, non è modellabile
- Si noti che, poiché le ricevute non riportano nessun dato che identifichi i clienti, non esiste nessuna relazione diretta tra RICEVUTE e PRENOTAZIONI

Sistemi Informativi T
10 febbraio 2011
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) se una coppia di istanze (k1,k2) di E1 ed E2 è associata tramite R1, non può essere associata anche tramite R2, e viceversa;

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,
K2 INT NOT NULL); -- il vincolo di foreign key va dichiarato dopo aver definito E2

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

ALTER TABLE E1
ADD CONSTRAINT FKR1 FOREIGN KEY (K2) REFERENCES E2 ;

CREATE TABLE R1C(
K1 INT NOT NULL REFERENCES E1,
C INT NOT NULL,
PRIMARY KEY (K1,C)  );
```

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

```
-- Il vincolo del punto c) NON e' violabile in fase di inserimento, ma solo in fase di aggiornamento
CREATE TRIGGER UPDATE_E2
NO CASCADE BEFORE UPDATE ON E2
REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN (EXISTS (SELECT *
                    FROM E1
                    WHERE E1.K1 = N.K1
                    AND E1.K2 = N.K2 ))
SIGNAL SQLSTATE '70001' ('La coppia (K1,K2)  inserita e' gia' presente in E1!')@

CREATE TRIGGER UPDATE_E1
NO CASCADE BEFORE UPDATE ON E1
REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN (EXISTS (SELECT *
                    FROM E2
                    WHERE E2.K2 = N.K2
                    AND E2.K1 = N.K1 ))
SIGNAL SQLSTATE '70001' ('La coppia (K1,K2)  inserita e' gia' presente in E2!')@
```