

Sistemi Informativi T
27 gennaio 2010
Risoluzione

Tempo a disposizione: 2:30 ore

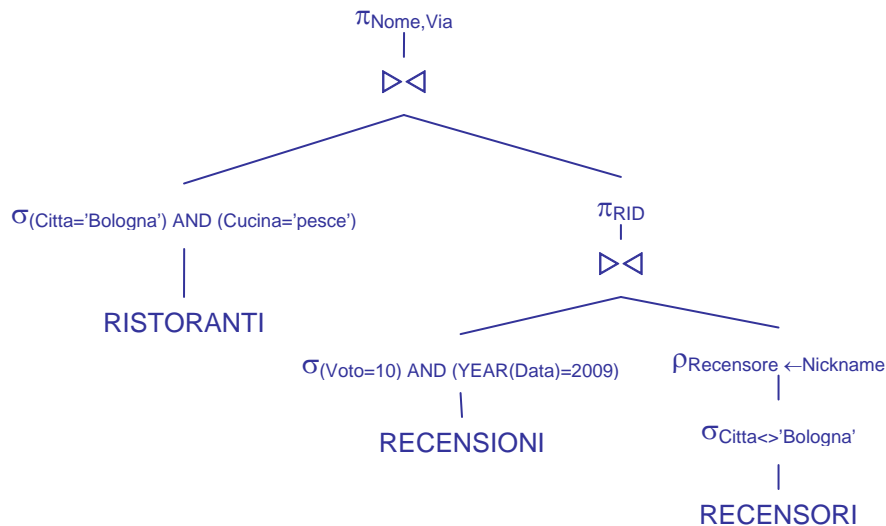
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

```
RISTORANTI(RID, Nome, Cucina, Via, Citta);  
-- Cucina: tipo cucina (regionale, pesce, carne, ecc.)  
RECENSIONI(RCID, RID, Recensore, Data, Voto, Npersone, Prezzo),  
  RID references RISTORANTI, Recensore references RECENSORI;  
-- Prezzo: prezzo totale pagato per Npersone  
-- Voto: di tipo INT tra 1 e 10  
RECENSORI(Nickname, Citta, NumRecensioni);  
-- NumRecensioni: num. totale di recensioni scritte
```

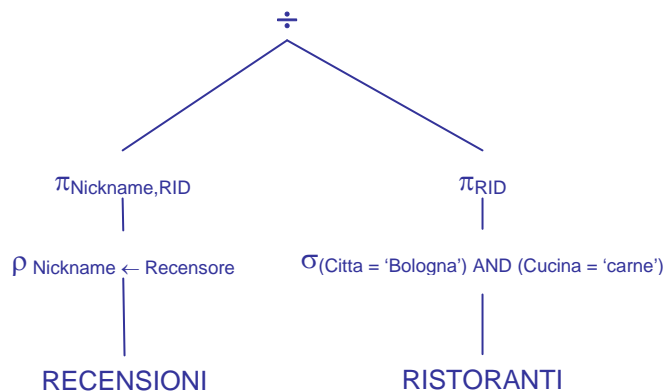
si scrivano in algebra relazionale le seguenti interrogazioni:

- 1.1) [1 p.]** Nome e via dei ristoranti di pesce di Bologna che nel 2009 hanno ricevuto almeno un 10 da un recensore non di Bologna



E' importante fare in modo che il join con RISTORANTI sia eseguito solo su RID. A tale scopo o si proietta, oppure si deve ridenominare l'attributo Citta (di RISTORANTI o di RECENSORI)

- 1.2) [2 p.]** I nickname dei recensori che hanno recensito tutti i ristoranti di carne di Bologna



La ridenominazione non è strettamente necessaria

Sistemi Informativi T
27 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.] Nome, via e voto medio (in formato XX.YY) dei ristoranti di pesce che hanno avuto almeno 3 recensioni da recensori di città diverse dalla loro, ordinati per voto medio decrescente e, in caso di parità, per nome

```
SELECT RI.Nome, RI.Via,  
       AVG(CAST(Voto AS DECIMAL(4,2))) AS VotoMedio  
FROM RISTORANTI RI, RECENSIONI RC, RECENSORI RE  
WHERE RI.RID = RC.RID  
AND RC.Recensore = RE.Nickname  
AND RI.Cucina = 'pesce'  
AND RI.Citta <> RE.Citta  
GROUP BY RI.RID, RI.Nome, RI.Via  
-- il GROUP BY va fatto anche su RID per evitare problemi di omonimia  
HAVING COUNT(*) >=3  
ORDER BY VotoMedio DESC, RI.Nome
```

Nella soluzione proposta la media viene calcolata considerando **solo** i voti di recensori di altre città. Un'interpretazione alternativa consiste nel calcolare la media considerando **tutti** i voti ricevuti:

```
SELECT RI.Nome, RI.Via, AVG(CAST(Voto AS DECIMAL(4,2))) AS VotoMedio  
FROM RISTORANTI RI, RECENSIONI RC  
WHERE RI.RID = RC.RID  
AND RI.Cucina = 'pesce'  
AND (SELECT COUNT(*) -- numero di recensori di altre citta'  
      FROM RECENSIONI RC1, RECENSORI RE  
      WHERE RI.RID = RC1.RID  
      AND RC1.Recensore = RE.Nickname  
      AND RI.Citta <> RE.Citta) >=3  
GROUP BY RI.RID, RI.Nome, RI.Via  
-- il GROUP BY va fatto anche su RID per evitare problemi di omonimia  
ORDER BY VotoMedio DESC, RI.Nome
```

Entrambe le soluzioni possono ritenersi corrette, data la parziale ambiguità della specifica

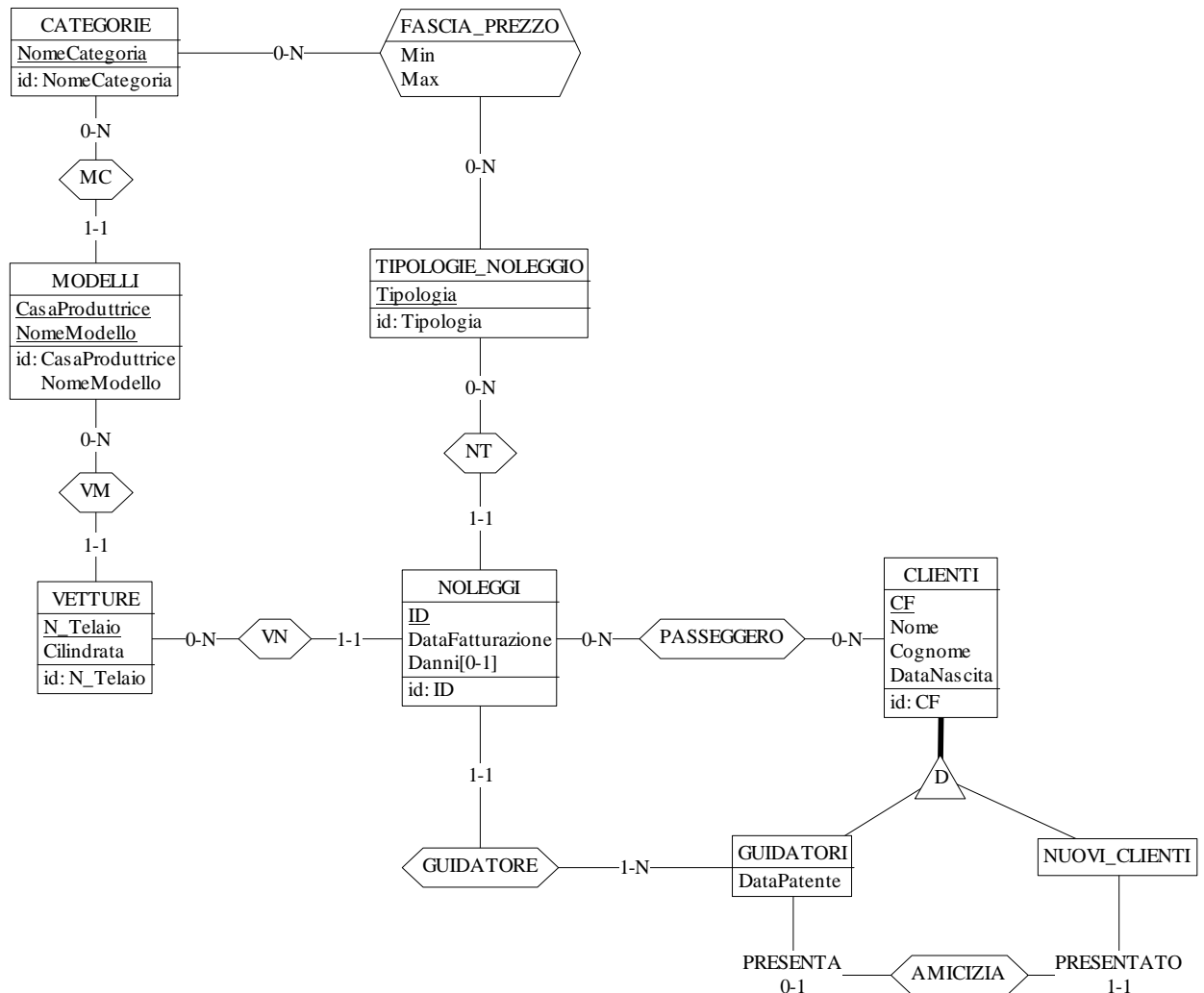
2.2) [3 p.] Il prezzo massimo e minimo pagato per persona nei ristoranti di Bologna, escludendo le recensioni di recensori con meno di 5 recensioni scritte

```
SELECT RI.RID, RI.Nome,  
       MAX(CAST(Prezzo AS DECIMAL(4,2))/Npersone) AS PrezzoMassimo,  
       MIN(CAST(Prezzo AS DECIMAL(4,2))/Npersone) AS PrezzoMinimo  
FROM RISTORANTI RI, RECENSIONI RC  
WHERE RI.RID = RC.RID  
AND RC.Recensore IN( SELECT Nickname  
                     FROM RECENSORI  
                     WHERE NumRecensioni >= 5)  
AND RI.Citta = 'Bologna'  
GROUP BY RI.RID, RI.Nome  
-- il GROUP BY va fatto anche su RID per evitare problemi di omonimia
```

Sistemi Informativi T
27 gennaio 2010
Risoluzione

3) Progettazione concettuale (6 punti)

La ditta di rental car SIVA desidera informatizzare il proprio database per la gestione dei noleggi. Di ogni vettura nel proprio parco auto desidera memorizzare il numero di telaio univoco, la cilindrata, la casa produttrice e il modello. Ogni modello di auto appartiene ad una specifica categoria d'auto (es.: citycar, economy, premium, minivan, ...). SIVA mette a disposizione diverse tipologie di noleggio: giornaliero, mezza giornata, settimanale, ecc.. La fascia di prezzo di un'auto dipende dalla categoria dell'auto e dalla tipologia di noleggio. Per ogni noleggio effettuato, identificato da un codice univoco, si desidera memorizzare l'auto noleggiata, la tipologia di noleggio, la data della fatturazione del noleggio e se l'auto è stata danneggiata durante il noleggio. Inoltre ad ogni noleggio sono associati i dati del guidatore e degli eventuali passeggeri (si vuole poter distinguere il guidatore dai passeggeri). Per ciascuno di questi clienti si desidera memorizzare il codice fiscale, il nome, il cognome e la data di nascita. Per gli utenti guidatori si desidera memorizzare anche la data in cui hanno conseguito la licenza di guida. Per allargare il proprio bacino di clienti la SIVA ha lanciato una campagna promozionale per cui ogni cliente guidatore ne può presentare un altro come suo amico per godere di particolari offerte speciali.

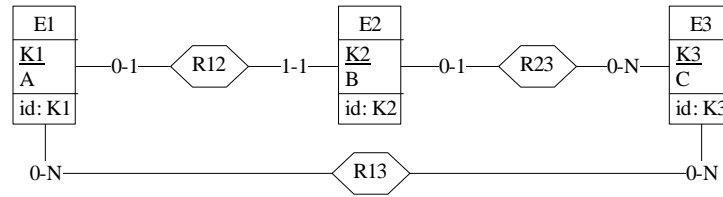


Commenti

- La soluzione proposta modella passeggeri, guidatori e potenziali nuovi clienti (entità NUOVI_CLIENTI) come una gerarchia unica. In alternativa i nuovi clienti potevano modellarsi separatamente, non essendo per questi specificati gli attributi di interesse

4) Progettazione logica (6 punti totali)

Dato lo schema concettuale in figura



e considerando che:

- tutti gli attributi sono di tipo INT;
- le associazioni R12 e R23 non vengono tradotte separatamente;
- le entità E1 ed E2 vengono tradotte assieme;
- un'istanza di E1 non è mai associata, tramite R13, a istanze 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 E3(
  K3 INT NOT NULL PRIMARY KEY,
  C INT NOT NULL );

CREATE TABLE E12(
  K1 INT NOT NULL PRIMARY KEY,
  A INT NOT NULL,
  SEL2 SMALLINT CHECK (SEL2 IN (1,2)), -- SEL2 = 2 : la tupla rappresenta anche un'istanza di E2
  K2 INT, -- in DB2 una chiave non ammette valori nulli, quindi non si può definire K2 UNIQUE...
  B INT,
  K3 INT REFERENCES E3,
  CONSTRAINT E2_DEFINED CHECK ((SEL2 = 2 AND K2 IS NOT NULL AND B IS NOT NULL)
    OR (SEL2 = 1 AND K2 IS NULL AND B IS NULL AND K3 IS NULL)) );

CREATE TABLE R13(
  K1 INT NOT NULL REFERENCES E12,
  K3 INT NOT NULL REFERENCES E3,
  PRIMARY KEY (K1,K3) );
```

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_R13
NO CASCADE BEFORE INSERT ON R13
REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN ( EXISTS ( SELECT *
                FROM E12, E3
                WHERE E12.K1 = N.K1 AND E3.K3 = N.K3 AND E12.A > E3.C ))
SIGNAL SQLSTATE '70001' ('Inserimento non permesso')@

CREATE TRIGGER INS_E12
NO CASCADE BEFORE INSERT ON E12
REFERENCING NEW AS N
FOR EACH ROW MODE DB2SQL
WHEN ( EXISTS ( SELECT *
                FROM E12
                WHERE E12.K2 = N.K2 ))
SIGNAL SQLSTATE '70002' ('I valori di K2 non possono essere duplicati')@
```