Sistemi Informativi T
31 gennaio 2012
Risoluzione

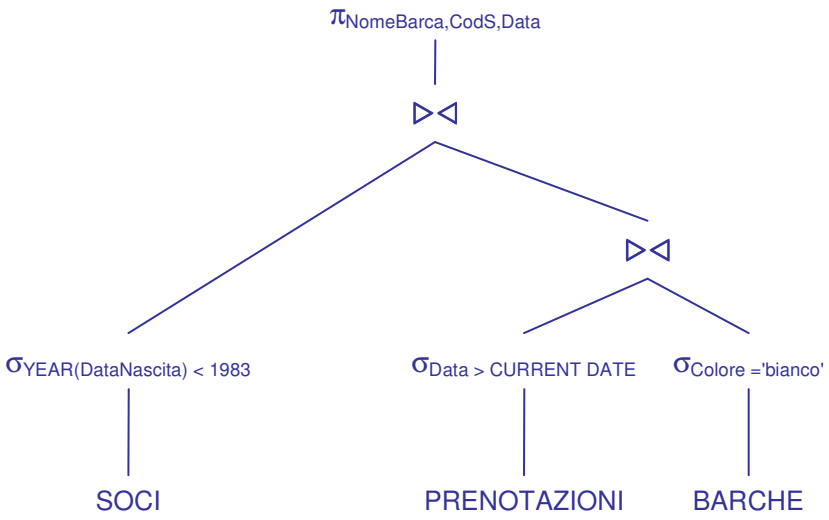
Tempo a disposizione: 2:30 ore

1) Algebra relazionale (3 punti totali):

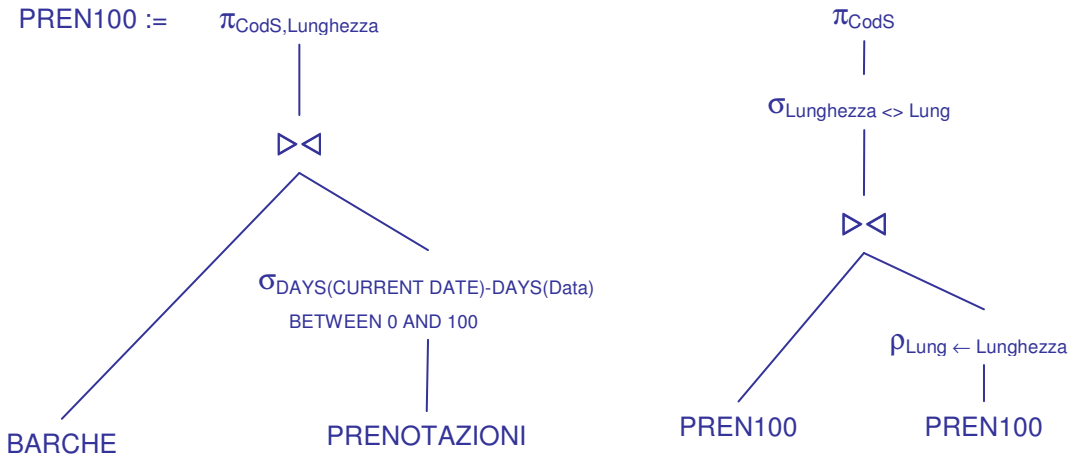
Date le seguenti relazioni:

```
BARCHE(NomeBarca,Colore,Lunghezza);
-- Lunghezza e' in formato Dec(4,2)
SOCI(CodS,Nome,Cognome,DataNascita);
PRENOTAZIONI(NomeBarca,CodS,Data),
NomeBarca REFERENCES BARCHE, CodS REFERENCES SOCI;
-- Data si riferisce alla data in cui si vuole usare una barca,
-- NON alla data in cui e' stata eseguita la prenotazione
```

1.1) [1 p.] I dati delle prenotazioni future fatte da soci nati prima del 1983 per barche di colore bianco



1.2) [2 p.] I soci che hanno noleggiato almeno due barche di diversa lunghezza negli ultimi 100 giorni



La vista PREN100 restituisce le prenotazioni degli ultimi 100 giorni. Il self-join viene eseguito sul solo CodS. Si noti che non basta specificare DAYS(CURRENT DATE) - DAYS(Data) <= 100, perché questo è soddisfatto anche da prenotazioni per date future

Sistemi Informativi T
31 gennaio 2012
Risoluzione

2) SQL (5 punti totali)

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

2.1) [2 p.] Per ogni anno, il numero di barche diverse noleggiate dai soci

```
SELECT   YEAR(P.Data) AS Anno, P.CodS, COUNT(DISTINCT NomeBarca) AS NumBarche
FROM     PRENOTAZIONI P
GROUP BY YEAR(P.Data), P.CodS
```

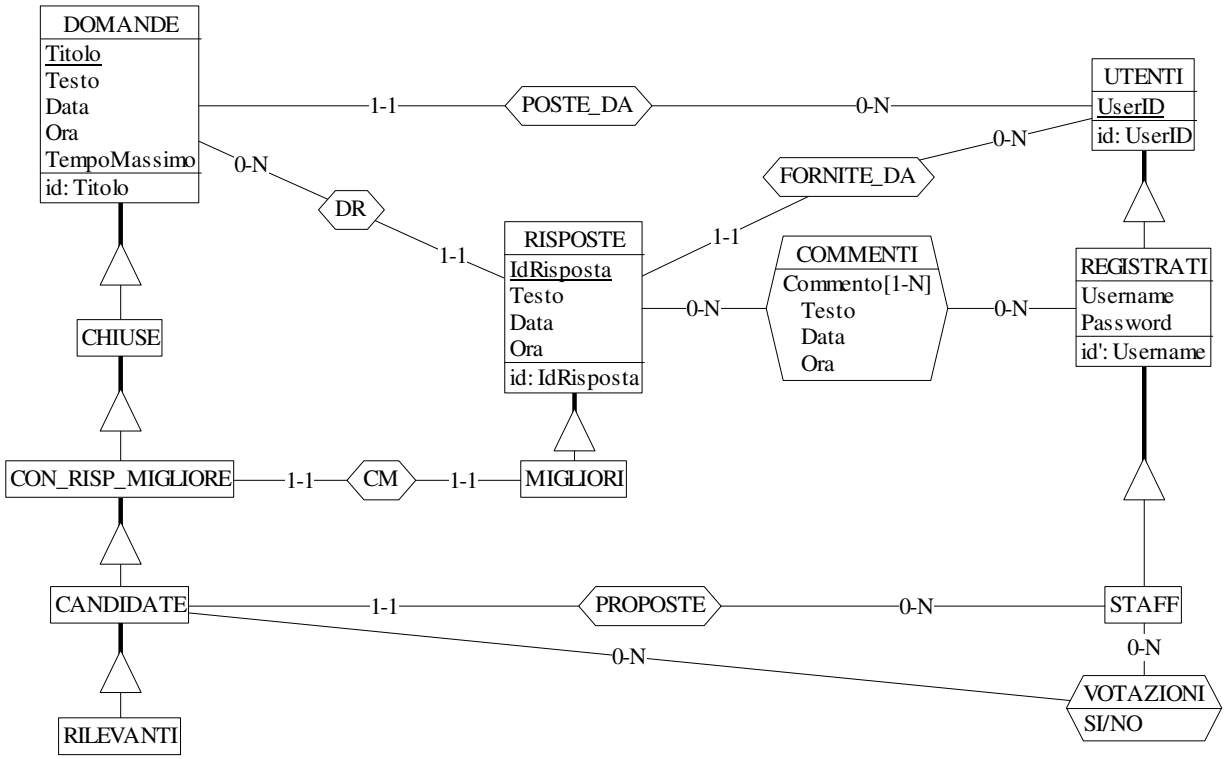
2.2) [3 p.] I soci che non hanno prenotato in giorni consecutivi due barche diverse della stessa lunghezza

```
SELECT S.CodS
FROM SOCI S
   EXCEPT
SELECT P1.CodS
FROM PRENOTAZIONI P1, PRENOTAZIONI P2, BARCHE B1, BARCHE B2
WHERE P1.CodS = P2.CodS
AND P1.NomeBarca = B1.NomeBarca
AND P2.NomeBarca = B2.NomeBarca
AND DAYS(P2.Data) - DAYS(P1.Data) = 1
AND B1.Lunghezza = B2.Lunghezza
AND B1.NomeBarca <> B2.NomeBarca
```

Sistemi Informativi T
31 gennaio 2012
Risoluzione

3) Progettazione concettuale (6 punti)

Il sito TheBestAnswer (TBA) permette a chiunque, registrato o meno, di porre domande di qualsiasi natura (Come si pulisce il caffé dai tappeti? Chi è nato prima, l'uovo o la gallina?...). Ad ogni domanda, caratterizzata da un titolo, il testo, data e ora, ognuno può rispondere entro un tempo massimo stabilito da chi formula la domanda (dopodiché la domanda è "chiusa"). Chi ha formulato la domanda può quindi scegliere, se lo ritiene opportuno, la risposta migliore. A differenza degli utenti occasionali, un utente registrato ha la possibilità di fare commenti (uno o più) sulle risposte date. Per riposte e commenti è importante mantenere informazioni anche su data e ora.
Alcune domande chiuse per cui è stata scelta la risposta migliore, ritenute particolarmente interessanti, vengono catalogate come "rilevanti". Tale catalogazione è proposta da un membro dello staff di TBA, e viene approvata a maggioranza (è importante memorizzare chi dello staff ha votato e come: a favore o meno).



Commenti:

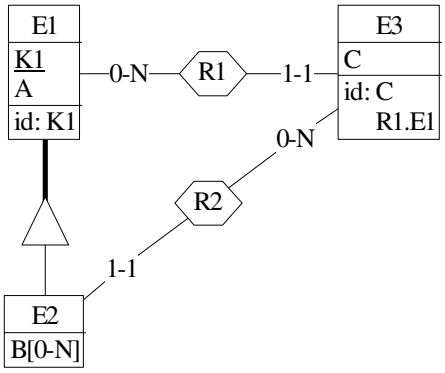
- La gerarchia radicata in DOMANDE gestisce i vari "stati" di una domanda.
- L'associazione CM è ridondante, in quanto se una risposta è scelta come migliore lo è necessariamente per la domanda cui è associata tramite DR. A livello di schema questo vincolo tuttavia non è esprimibile.
- La soluzione proposta modella i membri dello STAFF come un subset degli utenti REGISTRATI. In alternativa sarebbe stato lecito anche modellare queste due entità in mutua esclusione tra loro.
- L'identificatore di UTENTI è qui un generico UserID, che potrebbe anche coincidere con lo Username per gli utenti registrati e uguale all'indirizzo IP per i non registrati.

Sistemi Informativi T
31 gennaio 2012
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) le entità E1 ed E2 vengono tradotte assieme;
d) un'istanza di E2 non è mai associata, tramite R1 e R2, a istanze di E2 che hanno almeno un valore B > 10;



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 (0,1)),        -- 1: istanza anche di E2
K1R2 INT,
CR2 INT,
CONSTRAINT E2 CHECK
     ( (TIPO2 = 1 AND K1R2 IS NOT NULL AND CR2 IS NOT NULL) OR
       (TIPO2 = 0 AND K1R2 IS NULL AND CR2 IS NULL) )                    );

CREATE TABLE E2B
K1 INT NOT NULL REFERENCES E1
B INT NOT NULL,
PRIMARY KEY (K1,B)        );

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

ALTER TABLE E1
ADD COSTRAINT FKR2 FOREIGN KEY (K1R2,CR2) REFERENCES E3(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

```
-- Quando si inserisce una tupla in E2B bisogna verificare che il valore di K1 referenzi un'istanza di E2
CREATE TRIGGER E2B_REFERENZIA_E2
NO CASCADE BEFORE INSERT ON E2
REFERENCING NEW AS N
FOR EACH ROW
WHEN (NOT EXISTS ( SELECT *
                   FROM E1
                   WHERE E1.K1 = N.K1
                    AND E1.TIPO2 = 1) )
SIGNAL SQLSTATE '70001' ('La tupla inserita deve referenziare una istanza di E2!')@

-- Per garantire il rispetto del vincolo di cui al punto d) è necessario impostare il seguente trigger:
CREATE TRIGGER PUNTO_D
NO CASCADE BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS (SELECT *
              FROM E2B
              WHERE E2B.K1 = N.K1R2          -- K1R2 e' definito solo per istanze di E2
              AND E2B.B > 10 ))              -- B e' definito solo per istanze di E2
SIGNAL SQLSTATE '70002' ('La tupla inserita referenzia una tupla di E2 con B>10!')@
```