Sistemi Informativi T
12 giugno 2015
Risoluzione

Tempo a disposizione: 2:30 ore

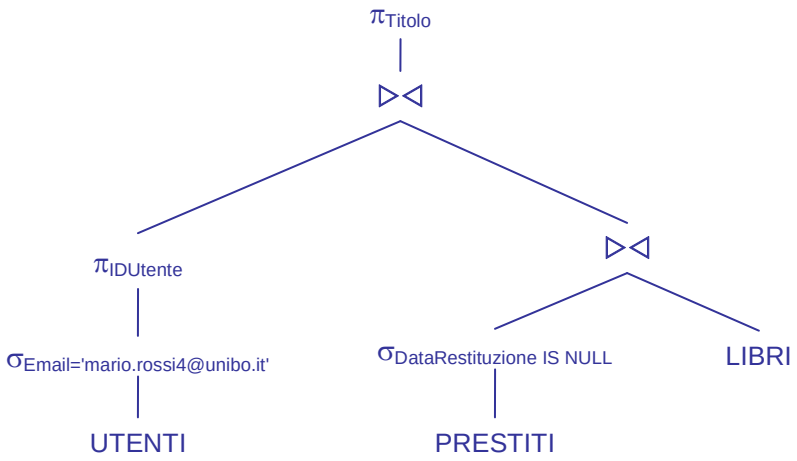
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

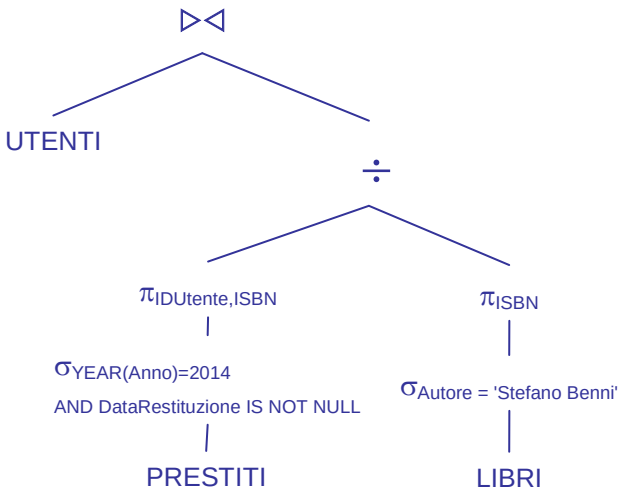
```
LIBRI(ISBN,Titolo,Autore);
UTENTI(IDUtente,Nome,Cognome,Email);
PRESTITI(ISBN,IDUtente,Data,DataRestituzione*),
ISBN references LIBRI, IDUtente references UTENTI;
-- l'asterisco indica la possibilità di valori nulli
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] I titoli dei libri attualmente in prestito all'utente con email 'mario.rossi4@unibo.it'



1.2) [2 p.] I dati degli utenti che nel 2014 hanno preso in prestito (e restituito) tutti i libri presenti in biblioteca scritti da Stefano Benni



Sistemi Informativi T
12 giugno 2015
Risoluzione

2) SQL (5 punti totali)

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

2.1) [2 p.] I dati degli utenti che nel 2014 hanno preso in prestito (e restituito) tutti i libri presenti in biblioteca scritti da Stefano Benni

```
SELECT   U.IDUtente,U.Nome,U.Cognome,U.Email
FROM     UTENTI U, PRESTITI P, LIBRI L
WHERE    U.IDUtente = P.IDUtente
AND      P.ISBN = L.ISBN
AND      YEAR(P.Data) = 2014
AND      P.DataRestituzione IS NOT NULL
AND      L.Autore = 'Stefano Benni'
GROUP BY U.IDUtente,U.Nome,U.Cognome,U.Email
HAVING   COUNT(DISTINCT L.ISBN) = (SELECT COUNT(*)
                                   FROM   LIBRI
                                   WHERE  AUTORE = 'Stefano Benni')
-- Come nella query successiva, COUNT(DISTINCT ...) evita di contare più
-- volte lo stesso libro
```

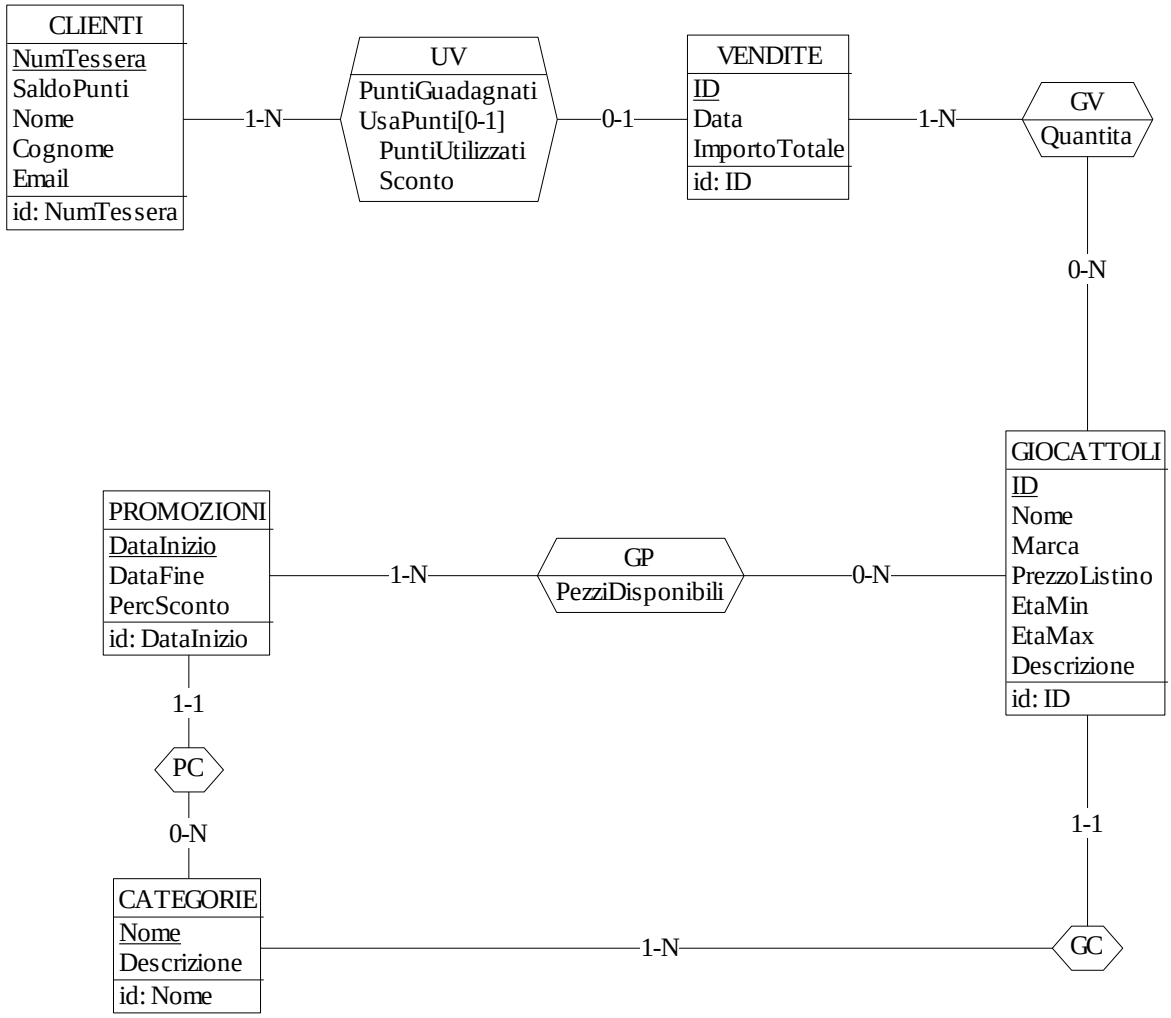
2.2) [3 p.] Per ogni utente, l'autore più letto (un libro preso in prestito più volte dallo stesso utente viene conteggiato una sola volta)

```
WITH NUMLETTI (Utente,Autore,NLetti) AS (
   SELECT   P.IDUtente, L.Autore, COUNT(DISTINCT L.ISBN)
   FROM     PRESTITI P, LIBRI L
   WHERE    P.ISBN = L.ISBN
   GROUP BY P.IDUtente, L.Autore       )

SELECT   N1.*
FROM     NUMLETTI N1
WHERE    N1.NLetti = ( SELECT    MAX(N2.NLetti)
                       FROM      NUMLETTI N2
                       WHERE     N2.Utente = N1.Utente )

-- La c.t.e. calcola, per ogni utente, il numero di libri (diversi)
-- che l'utente ha letto di ciascun autore
```

Sistemi Informativi T
12 giugno 2015
Risoluzione

3) Progettazione concettuale (6 punti)

GioCaGiò (GCG) è un grande negozio di giocattoli il cui catalogo include oltre 30000 prodotti diversi, ognuno caratterizzato da un nome, una marca, il prezzo di listino, la fascia di età consigliata, una descrizione e la categoria di appartenenza (ogni categoria ha a sua volta una propria descrizione). Periodicamente GCG organizza vendite promozionali in cui molti prodotti vengono venduti a presso scontato: ogni vendita, che di norma dura almeno una settimana e riguarda sempre una sola categoria, applica lo stesso sconto a tutti i giocattoli in promozione (per ognuno dei quali però il numero di pezzi disponibili è limitato e stabilito in anticipo).
Gli acquisti eseguiti dai clienti tesserati comportano l'attribuzione di "punti GCG", che possono essere utilizzati per usufruire di ulteriori sconti. Oltre alle vendite normali, il sistema della GCG tiene conto dei punti erogati ai clienti o da questi utilizzati, e nel secondo caso dello sconto in Euro praticato. Ovviamente ogni acquisto riguarda uno o più prodotti, ognuno in una certa quantità.



Commenti:

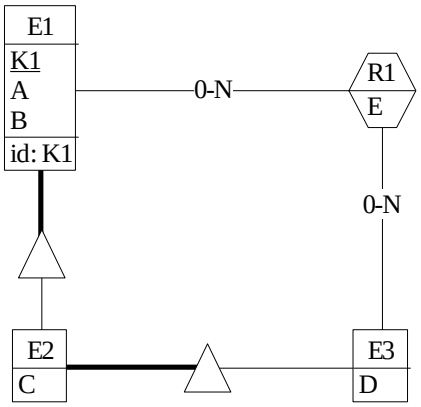
- L'esercizio non presenta particolari difficoltà. L'unico punto degno di nota riguarda la parte delle specifiche asserente "[...] punti erogati ai clienti o da questi utilizzati, e nel secondo caso dello sconto in Euro praticato", in cui "o" non va inteso in senso esclusivo (acquistando dei giocattoli si guadagnano punti e se ne possono anche utilizzare per ottenere uno sconto).
- Il vincolo che tutti i giocattoli relativi a una data vendita promozionale facciano parte della stessa categoria (in particolare, quella individuata dall'associazione PC) non è esprimibile in E/R.

Sistemi Informativi T
12 giugno 2015
Risoluzione

- **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 E2 ed E3 vengono tradotte insieme e separatamente da E1;
c) l'attributo E di R1 ha un valore sempre maggiore della somma dei valori di A e D dell'istanza di E3 associata;



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,
B INT NOT NULL     );
-- Volendo si può anche inserire un selettore per E2, ad es:
-- TIPO2 SMALLINT NOT NULL CHECK (TIPO2 IN (1,2))
-- In questo caso è necessario anche prevedere un trigger che ad ogni inserimento in E23 verifichi che la
-- tupla referenziata in E1 abbia TIPO2 = 2

CREATE TABLE E23 (
K1 INT NOT NULL PRIMARY KEY REFERENCES E1,
C INT NOT NULL,
TIPO3 SMALLINT NOT NULL CHECK (TIPO3 IN (2,3)),          -- 3: istanza anche di E3
D INT,
CONSTRAINT E3 CHECK (
      (TIPO3 = 2 AND D IS NULL) OR (TIPO3 = 3 AND D IS NOT NULL) )                    );

CREATE TABLE R1 (
K1E1 INT NOT NULL REFERENCES E1,
K1E3 INT NOT NULL REFERENCES E23,
E INT NOT NULL,
PRIMARY KEY (K1E1,K1E3)                              );
```

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

```
-- Trigger che garantisce che una tupla di R1 referenzi una tupla di  E3
CREATE TRIGGER R1_E3
BEFORE INSERT ON R1
REFERENCING NEW AS N
FOR EACH ROW
WHEN (NOT EXISTS  ( SELECT * FROM E23
                              WHERE    N.K1E3 = E23.K1
                              AND         E23.TIPO3 = 3 )       )
SIGNAL SQLSTATE '70001' ('La tupla deve referenziare una tupla di E3!');

-- Trigger che garantisce il rispetto del vincolo di cui al punto c)
CREATE TRIGGER PUNTO_C
BEFORE INSERT ON R1
REFERENCING NEW AS N
FOR EACH ROW
WHEN (N.E <= (SELECT E23.D + E1.A FROM E23, E1
                     WHERE N.K1E3 = E23.K1
                     AND  E23.K1 = E1.K1           ) )
SIGNAL SQLSTATE '70002' ('Il valore di E e" troppo basso!');
```