Sistemi Informativi T
19 settembre 2016
Risoluzione

Tempo a disposizione: 2:30 ore

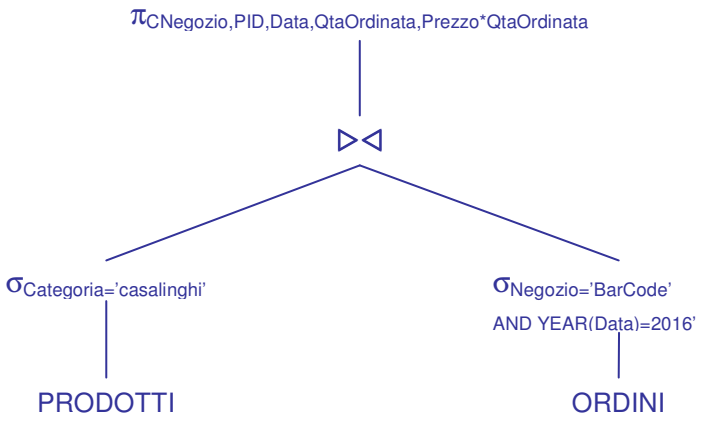
1) Algebra relazionale (3 punti totali)

Date le seguenti relazioni:

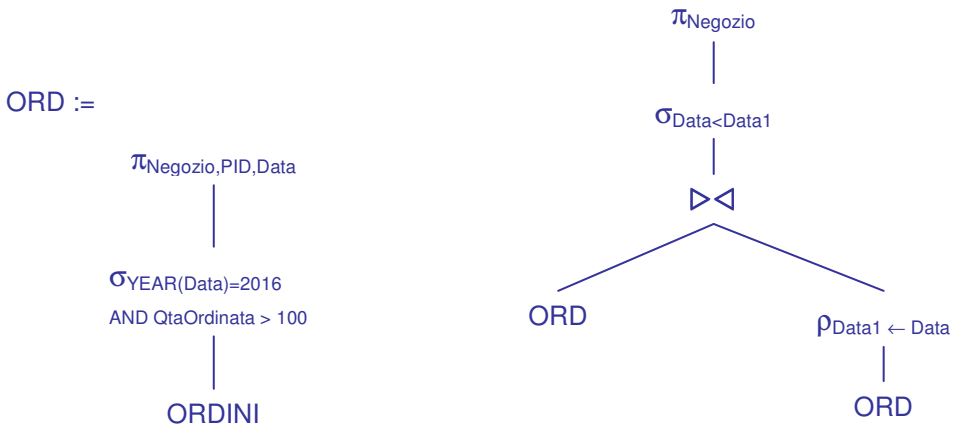
```
PRODOTTI(PID,Categoria,Prezzo);
ORDINI(Negozio,PID,Data,QtaOrdinata),
      PID REFERENCES PRODOTTI;
-- QtaOrdinata è di tipo INT
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) **[1 p.]** I dettagli degli ordini del negozio 'BarCode' nel 2016 per i prodotti di categoria 'casalinghi', aggiungendo a ogni ordine l'importo totale pagato (Prezzo*QtaOrdinata)



1.2) **[2 p.]** I negozi che nel 2016 hanno ordinato almeno 2 volte uno stesso prodotto, in entrambi i casi in quantità maggiore di 100



Il join garantisce che le tuple combinate si riferiscano allo stesso prodotto e allo stesso negozio; la condizione sulla data assicura che si tratti di due ordini distinti

Sistemi Informativi T
19 settembre 2016
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 negozio e ogni prodotto l'importo totale pagato

```
SELECT   O.Negozio, O.PID,
         SUM(P.Prezzo*O.QtaOrdinata) AS Importo_Totale
FROM     PRODOTTI P, ORDINI O
WHERE    P.PID = O.PID
GROUP BY O.Negozio, O.PID
```

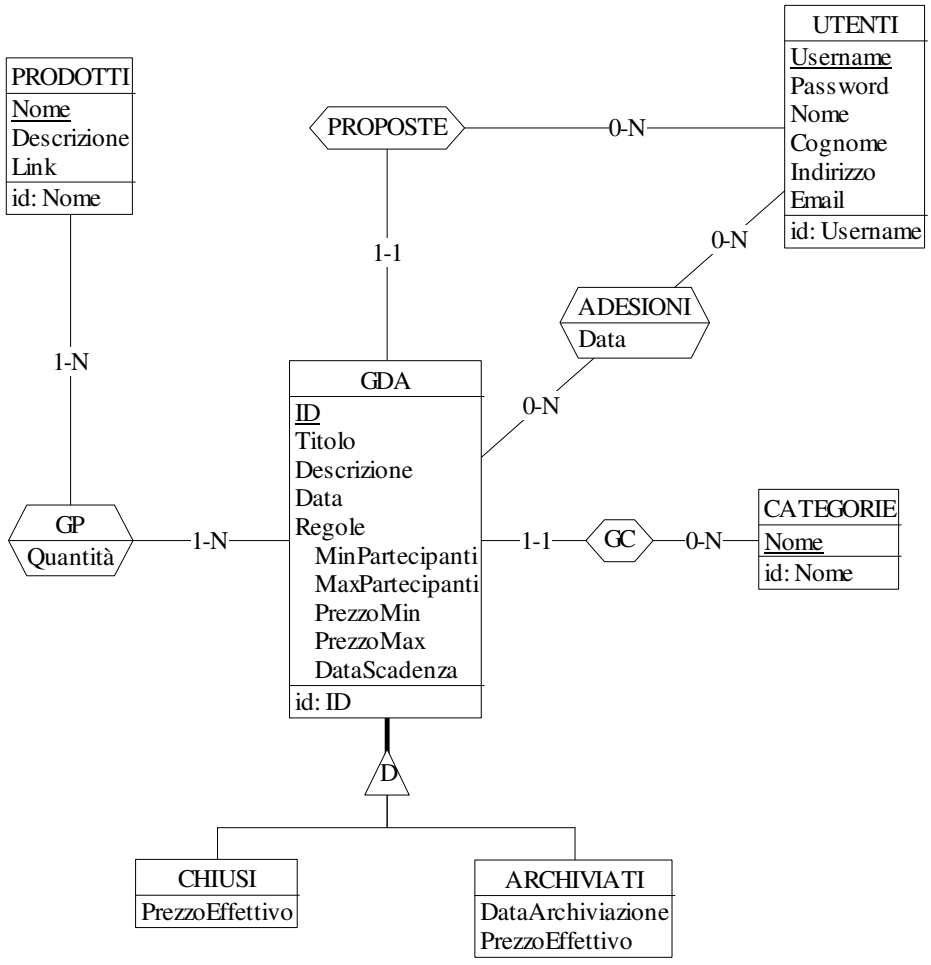
2.2) [3 p.] Per ogni negozio il prodotto che nel 2016 ha subito l'incremento maggiore di quantità complessivamente ordinata rispetto al 2015

```
WITH
TOTORDINI (Negozio,PID,Anno,Totale) AS (
   SELECT O.Negozio,O.Pid,YEAR(O.Data),SUM(O.QtaOrdinata)
   FROM   ORDINI O
   WHERE  YEAR(O.Data) IN (2015,2016)
   GROUP BY  O.Negozio,O.Pid,YEAR(O.Data) ),

INCREMENTI (Negozio,PID,Differenza) AS (
   SELECT T2016.Negozio,T2016.PID,T2016.Totale-T2015.Totale
   FROM   TOTORDINI T2016, TOTORDINI T2015
   WHERE  (T2016.Negozio,T2016.PID)= (T2015.Negozio,T2015.PID)
   AND    T2016.Anno = 2016
   AND    T2015.Anno = 2015      )

SELECT I1.*
FROM   INCREMENTI I1
WHERE  I1.Differenza > 0
AND    I1.Differenza = (   SELECT MAX(I2.Differenza)
                           FROM   INCREMENTI I2
                           WHERE  I1.Negozio = I2.Negozio )


-- La prima c.t.e. calcola per ogni negozio e ogni prodotto le quantità
-- totali ordinate nel 2015 e nel 2016.
-- La seconda c.t.e calcola la differenza tra le due quantità.
-- Il test I1.Differenza > 0 evita di restituire quei negozi in cui non
-- c'è stato nessun incremento di ordinativi
```

Sistemi Informativi T
19 settembre 2016
Risoluzione

3) Progettazione concettuale (6 punti)

GD4all è un sito web che ha lo scopo di semplificare la gestione di gruppi d'acquisto (GDA) di prodotti in rete. Ogni GDA viene attivato da un utente del sito e comprende un titolo, una descrizione, uno o più link ai prodotti che si vogliono acquistare e le regole specifiche di partecipazione al GDA (solo utenti registrati sul sito possono partecipare). In particolare, le regole includono un numero minimo e massimo di partecipanti, i prezzi minimi e massimi stimati di acquisto di un lotto (l'insieme dei prodotti che si vogliono acquistare) e una data di scadenza entro la quale aderire al GDA. Per ogni GDA viene stabilita una categoria (ad es. "alimentare", "viaggi", ecc.) scegliendola tra quelle previste dal sito.
Alla data di scadenza, se il numero minimo di partecipanti è stato raggiunto il GDA diventa "chiuso" e il responsabile comunica il prezzo effettivo di acquisto. Quando le procedure di pagamento dei lotti e di spedizione agli acquirenti sono concluse (queste fasi non avvengono sotto il controllo di GDA4all), il responsabile etichetta il GDA come "archiviato".
I GDA che non arrivano entro la data di scadenza a raccogliere un numero minimo di adesioni vengono rimossi dal sistema, che quindi non ne tiene più traccia.

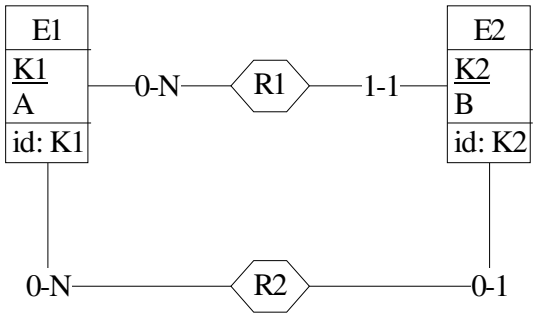


Commenti:

- L'esercizio non presenta difficoltà degne di nota.
- La gerarchia si basa sull'ipotesi che per i GDA archiviati si mantengano tutte le informazioni di partenza.

Sistemi Informativi T
19 settembre 2016
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) per le solo istanze di E2 che partecipano a R2, esiste una dipendenza funzionale da B ad A (tramite R2);

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 E2 (
K2 INT NOT NULL PRIMARY KEY,
B INT NOT NULL,
K1R1 INT NOT NULL REFERENCES E1,
K1R2 INT REFERENCES E1                  );
```

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 ';')

```
-- Per garantire il rispetto del vincolo di cui al punto d) è necessario impostare il seguente trigger:
CREATE TRIGGER DIP_FUN_B_A
BEFORE INSERT ON E2
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS  ( SELECT * FROM E2, E1 E1X, E1 E1Y
                WHERE   N.B = E2.B
                AND     E2.K1R2 = E1Y.K1
                AND     N.K1R2 = E1X.K1
                AND     E1Y.A <> E1X.A                  ) )
SIGNAL SQLSTATE '70001' ('La tupla inserita viola la dipendenza funzionale da B ad A!');
-- Si noti che E1 viene usata 2 volte, una per trovare il valore di A (E1Y.A) già associato al valore di B che
-- si sta inserendo,  l'altra per trovare il valore di A (E1X.A) associato alla nuova tupla inserita n E2
```