Sistemi Informativi T
18 settembre 2017
Risoluzione

Tempo a disposizione: 2:30 ore

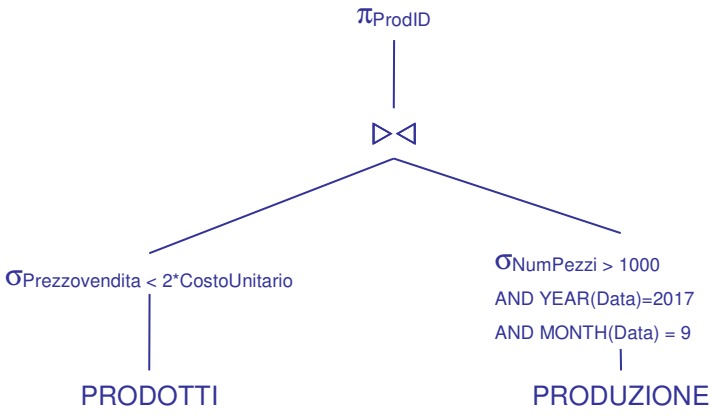
1) Algebra relazionale (3 punti totali)

Date le seguenti relazioni:

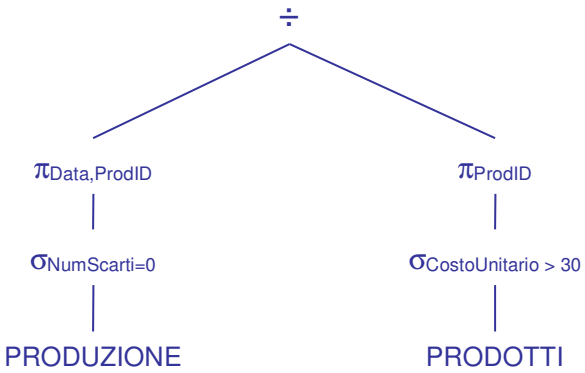
```
PRODOTTI(ProdID,CostoUnitario,PrezzoVendita);
PRODUZIONE(ProdID,Data,NumPezzi,NumScarti),
ProdID REFERENCES PRODOTTI;
-- NumPezzi e NumScarti sono interi (INT) non negativi
-- CostoUnitario e Prezzo Vendita sono di tipo DEC(7,2)
-- È sempre CostoUnitario < Prezzo Vendita
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) **[1 p.]** I codici dei prodotti per cui il prezzo di vendita è minore del doppio del costo unitario e per cui, in almeno un giorno di Settembre 2017, sono stati prodotti più di 1000 pezzi



1.2) **[2 p.]** Le date in cui sono stati in produzione tutti i prodotti di costo unitario maggiore di 30€ e per nessuno vi sono stati scarti di produzione (NumScarti = 0)



Il dividendo contiene, per ogni data, i codici dei prodotti che in quella data non hanno avuto scarti, mentre il divisore è dato dai prodotti di costo > 30€. Quindi la divisione restituisce una data se e solo il dividendo contiene tutti i prodotti nel divisore, come richiesto

Sistemi Informativi T
18 settembre 2017
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 prodotto, il numero di giorni in cui i pezzi scartati sono stati più del 5% di quelli prodotti

```
SELECT   ProdID, COUNT(*) AS NumGiorni
FROM     PRODUZIONE
WHERE    NumScarti > 0.05*NumPezzi
GROUP BY ProdID
```

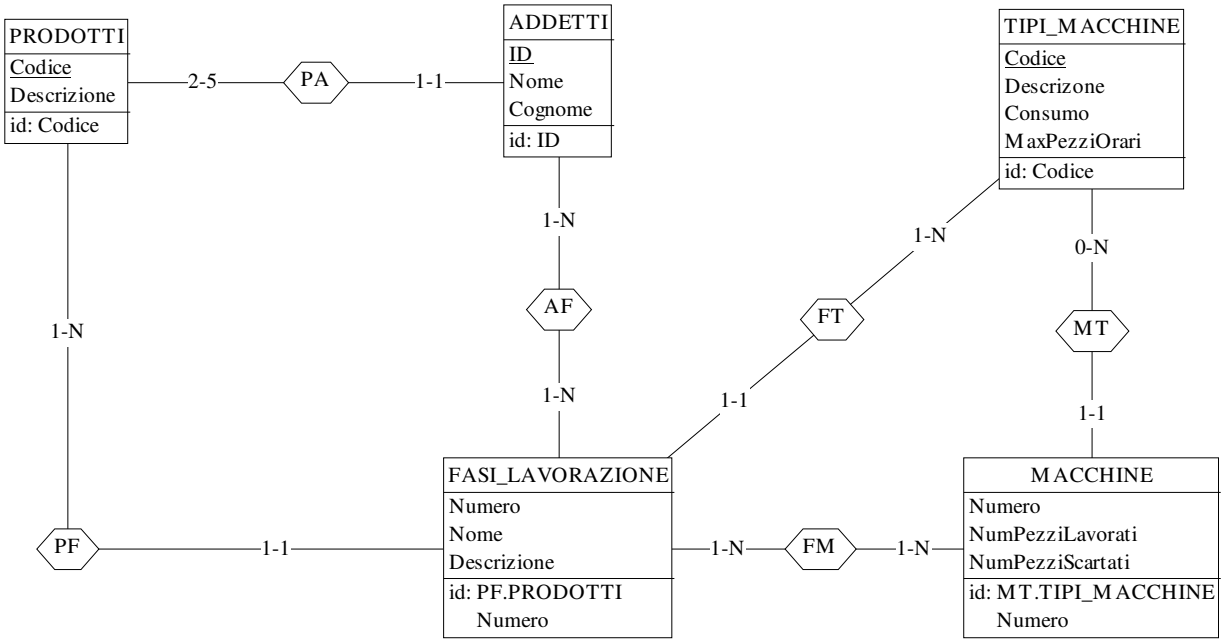
2.2) [3 p.] Per ogni mese, il giorno in cui la somma degli scarti di produzione è stata minima

```
WITH SUMSCARTI (Data,TotScarti) AS (
   SELECT Data,SUM(NumScarti)
   FROM   PRODUZIONE
   GROUP BY  Data

SELECT YEAR(Data),MONTH(Data),DAY(Data)
FROM   SUMSCARTI S
WHERE  S.TotScarti = ( SELECT MIN(S1.TotScarti)
                       FROM   SUMSCARTI S1
                     WHERE  (YEAR(S1.Data),MONTH(S1.Data)) =
                            (YEAR(S.Data),MONTH(S.Data))  )


-- La c.t.e. calcola per ogni data il totale degli scarti.
-- Si noti che "per ogni mese" non va inteso come "per ogni numero di mese"
```

Sistemi Informativi T
18 settembre 2017
Risoluzione

3) Progettazione concettuale (6 punti)

IND5.0 è una fabbrica all'avanguardia, in cui la produzione di tutti i prodotti avviene in maniera completamente automatizzata. Per ogni prodotto è prevista una specifica linea di produzione, costituita da diverse fasi di lavorazione in sequenza (fase 1, fase 2, ecc.), ad ognuna delle quali è anche dato un nome per distinguerla più chiaramente. Ogni fase della linea di produzione di un prodotto prevede l'utilizzo di una o più macchine dello stesso tipo, per ognuna delle quali è noto il consumo di energia elettrica e il massimo numero di pezzi lavorabili in un'ora (macchine dello stesso tipo hanno ovviamente le stesse caratteristiche). Per ogni macchina si tiene traccia del numero complessivo di pezzi lavorati e di pezzi scartati perché difettosi.
Tutta la produzione viene costantemente monitorata da addetti organizzati in squadre formate da un minimo di 2 a un massimo di 5 persone (ogni addetto fa parte di una e una sola squadra): una squadra monitora un singolo prodotto, e viceversa, mentre ogni fase è monitorata da uno o più addetti di quella squadra, e viceversa.



Commenti:

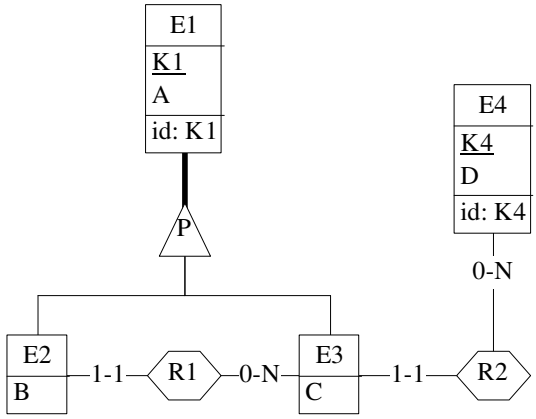
- L'esercizio non presenta particolari difficoltà.
- Nella soluzione proposte le sole cose degne di nota sono:
  - Non è presente nessuna entità per le squadre, che sarebbero in corrispondenza 1-1 con i PRODOTTI, in quanto non vi sono proprietà specifiche da rappresentare, e l'associazione PA tra PRODOTTI e ADDETTI è sufficiente
  - E' introdotta un'associazione ridondante FT tra le fasi e i tipi di macchine al fine di meglio evidenziare che in una fase di lavorazione di un prodotto si usa solo un tipo
- Non sono rappresentabili i vincoli che le fasi monitorate da un addetto si riferiscono tutte allo stesso prodotto cui l'addetto è associato tramite PA e che le macchine effettivamente usate in una fase sono dello stesso tipo, come specificato in FT

Sistemi Informativi T
18 settembre 2017
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 entità E1, E2 ed E3 vengono tradotte assieme;
c) le associazioni R1 e R2 non vengono tradotte separatamente;
d) un'istanza di E3 con C > 10 non può essere associata a un'istanza di E4 con D < 20;



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 E4 (
K4 INT NOT NULL PRIMARY KEY,
D INT NOT NULL                          );

CREATE TABLE E1 (
K1 INT NOT NULL PRIMARY KEY,
A INT NOT NULL,
TIPO23 SMALLINT NOT NULL CHECK (TIPO23 IN (2,3)),      -- 2: istanza di E2; 3: istanza di E3
B INT,
K1R1 INT NOT NULL REFERENCES E1,
C INT,
K4R2 INT REFERENCES E4,
CONSTRAINT E2E3 CHECK ((TIPO23 = 2 AND B IS NOT NULL AND K1R1 IS NOT NULL
                                    AND C IS NULL AND K4R2 IS NULL) OR
                       (TIPO23 = 3 AND B IS NULL AND K1R1 IS NULL
                        AND C IS NOT NULL AND K4R2 IS NOTNULL))                    );
```

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 R1 referenzi solo tuple di E3

CREATE TRIGGER R1_E3
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN (N.TIPO23 = 2 AND NOT EXISTS ( SELECT * FROM E1
                                    WHERE N.K1R1 = E1.K1
                                    AND   E1.TIPO23 = 3      ) )
SIGNAL SQLSTATE '70001' ('La tupla inserita deve referenziare una  tupla di E3!');

-- Trigger che garantisce il rispetto del vincolo di cui al punto d)

CREATE TRIGGER PUNTO_D
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN (N.C > 10 AND EXISTS ( SELECT * FROM E4
                            WHERE N.K4R2 = E4.K4
                            AND   E4.D < 20          ) )
SIGNAL SQLSTATE '70002' ('La tupla inserita ha C>10 e referenzia una  tupla di E4 con D<20!');
```