Sistemi Informativi T
16 settembre 2019
Risoluzione

Tempo a disposizione: 2:30 ore

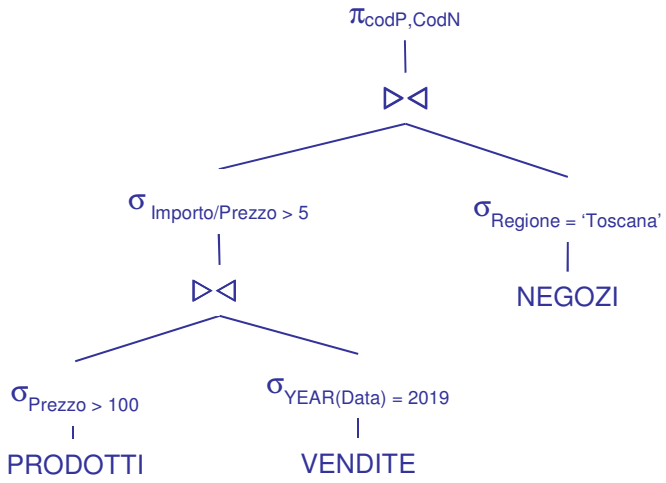
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

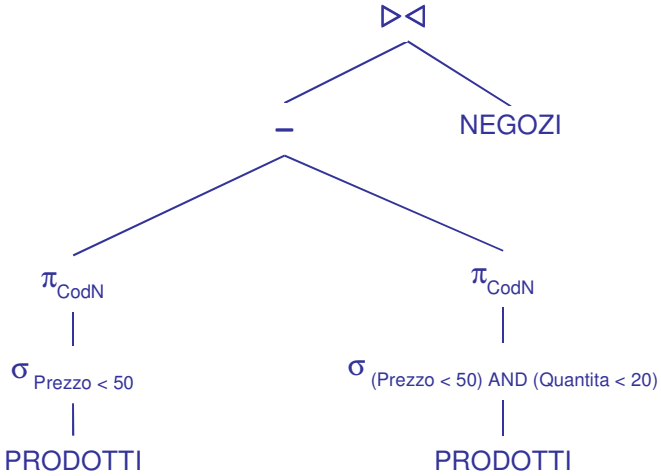
```
NEGOZI(CodN,Via,Comune,Regione);
PRODOTTI(CodP,CodN,Prezzo,Quantita),
      CodN REFERENCES NEGOZI;
VENDITE(CodP,CodN,Data,Importo),
      (CodP,CodN) REFERENCES PRODOTTI;
-- Importo e Prezzo sono di tipo DEC(8,2)
-- Quantita è un intero non negativo (= disponibilità del prodotto)
-- Importo rappresenta l'incasso totale per le vendite di un dato
-- prodotto in un dato negozio in una certa data
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) **[1 p.]** I codici dei prodotti e dei negozi in Toscana per cui il prezzo di vendita del prodotto in quel negozio è maggiore di 100€ e il negozio, in almeno un giorno del 2019, ha venduto più di 5 pezzi di quel prodotto



1.2) **[2 p.]** I dati dei negozi che hanno tutti i prodotti da loro venduti e di costo inferiore a 50€ disponibili in quantità pari a 20 o più



Il predicato Prezzo<50 nell'operando sinistro della differenza serve a non fornire come risultato anche quei negozi che non hanno prodotti che costano meno di 50€

Sistemi Informativi T
16 settembre 2019
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, l'incasso totale nel 2019 per i prodotti venduti solo da quel negozio nella propria regione

```
SELECT   N.CodN, SUM(V.Importo) AS Totale
FROM     NEGOZI N, VENDITE V
WHERE    N.CodN = V.CodN
AND      YEAR(V.Data) = 2019
AND      V.CodP NOT IN (   SELECT P1.CodP
                           FROM  NEGOZI N1, PRODOTTI P1
                           WHERE N1.CodN = P1.CodN
                           AND   N1.CodN <> N.CodN
                           AND   N1.Regione = N.Regione    )
GROUP BY N.CodN;

-- la subquery restituisce tutti i codici dei prodotti venduti da qualche
-- altro negozio (N1.CodN <> N.CodN) nella stessa regione di quello per cui
-- si valuta il blocco esterno, che quindi non vanno considerati nel calcolo
-- dell'incasso totale
```

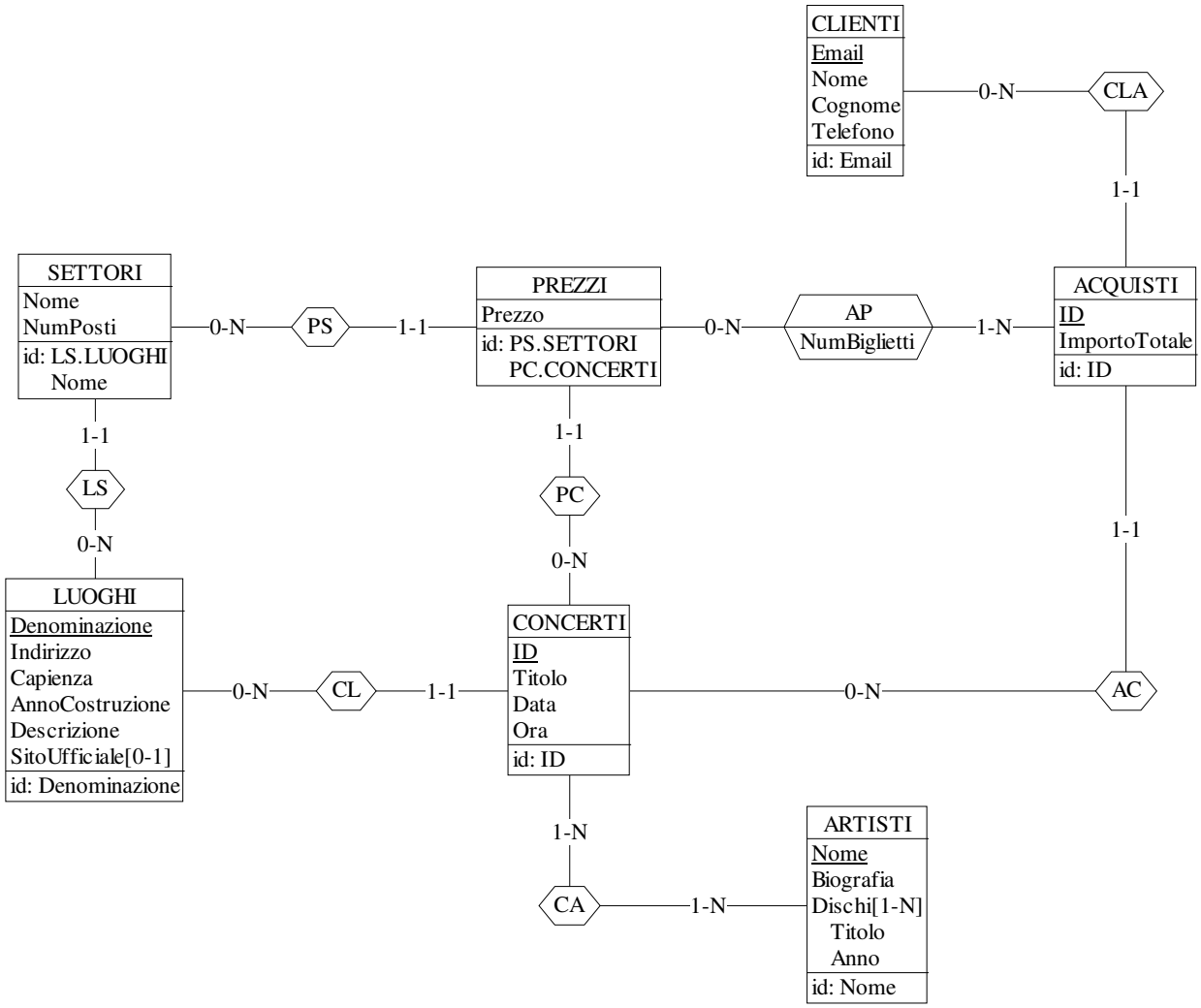
2.2) [3 p.] Per ogni prodotto, la regione in cui la differenza di prezzo tra due negozi che vendono quel prodotto è massima

```
WITH
   MAX_DIFF (CodP,Regione,DiffMassima) AS (
   SELECT   P.CodP, N.Regione, MAX(P.Prezzo) - MIN(P.Prezzo)
   FROM     NEGOZI N, PRODOTTI P
   WHERE    N.CodN = P.CodN
   GROUP BY P.CodP, N.Regione
   HAVING   COUNT(*) > 1
          )

SELECT   M.CodP, M.Regione
FROM     MAX_DIFF M
WHERE    M.DiffMassima = ( SELECT MAX(M1.DiffMassima)
                           FROM   MAX_DIFF M1
                           WHERE  M1.CodP = M.CodP  );

-- La c.t.e. calcola la differenza massima di prezzo di ogni prodotto in ogni
-- regione in cui quel prodotto è venduto.
-- Si noti che la clausola HAVING evita, nel caso di 1 solo negozio che vende
-- un certo prodotto, di restituire il valore 0
-- Possibile, ma inutilmente più complicato, confrontare a parità di prodotto
-- coppie di negozi in una stessa regione, quindi calcolare il MAX delle
-- differenze di prezzo ottenute
```

Sistemi Informativi T
16 settembre 2019
Risoluzione

3) Progettazione concettuale (6 punti)

La società MusicLive (ML) organizza e gestisce la vendita di biglietti per concerti sul territorio nazionale. Ogni concerto ha un identificatore univoco, un titolo ufficiale con cui viene propagandato, uno o più artisti che si esibiscono, e ovviamente informazioni su data, ora e luogo dell'evento. Per permettere di avere maggiori informazioni, ML mantiene anche dettagli sugli artisti (biografia e l'elenco dei principali dischi prodotti, con titolo e anno di uscita) e sui luoghi dei concerti (indirizzo, capienza, anno di costruzione, una breve descrizione e, se presente, un link al sito ufficiale del luogo).
I prezzi dei biglietti variano da un concerto all'altro, e dipendono dal settore scelto (ogni luogo ha i propri specifici settori, ognuno caratterizzato dal numero massimo di posti utilizzabili). ML mantiene traccia di tutti gli acquisti, registrando nome, cognome ed email dell'acquirente (quest'ultima identifica univocamente il cliente sul sito di ML), numero di biglietti acquistati per ogni settore, e importo totale pagato.

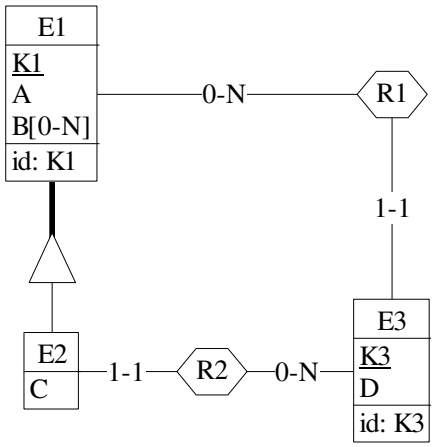


Commenti:

- L'associazione AC, che è ridondante e può quindi essere omessa, evidenzia che un acquisto si riferisce a un singolo concerto. Tuttavia, il vincolo che i biglietti di un acquisto si riferiscano tutti allo stesso concerto non è esprimibile in E/R.
- L'entità PREZZI è ottenuta per reificazione, dovendo essere riferita dall'associazione AP.

Sistemi Informativi T
16 settembre 2019
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 ed E2 vengono tradotte insieme;
c) le associazioni R1 e R2 non vengono tradotte separatamente;
d) Un'istanza di E3 non è mai associata, tramite R1, a un'istanza di E1 che ha un valore di B maggiore di 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,
SEL SMALLINT NOT NULL CHECK (SEL IN (1,2)),          -- 2: istanza anche di E2
C   INT,
K3  INT,
CONSTRAINT E2 CHECK (  (SEL = 1 AND C IS NULL AND K3 IS NULL) OR
                       (SEL = 2 AND C IS NOT NULL AND K3 IS NOT NULL) )  );

CREATE TABLE E1B (
K1  INT NOT NULL REFERENCES E1,
B   INT NOT NULL,
PRIMARY KEY (K1,B)        );

CREATE TABLE E3 (
K3  INT NOT NULL PRIMARY KEY,
D   INT NOT NULL,
K1  INT NOT NULL REFERENCES E1          );

ALTER TABLE E1
ADD CONSTRAINT FKR2 FOREIGN KEY (K3) REFERENCES E3;
```

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 il rispetto del vincolo al punto d)

CREATE TRIGGER PUNTO_D
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS (     SELECT *
                   FROM    E1B
                   WHERE N.K1 = E1B.K1
                   AND     E1B.B > 10        ) )
SIGNAL SQLSTATE '70001' ('La tupla è associata a una tupla di E1 con un valore di B > 10!');

-- Si noti che è inutile scrivere la clausola WHERE con 2 predicati di join, N.K1 = E1.K1 AND
-- E1.K1 = E1B.K1, data la presenza in E1B del vincolo di foreign key.
```