Sistemi Informativi T
19 settembre 2013
Risoluzione

Tempo a disposizione: 2:30 ore

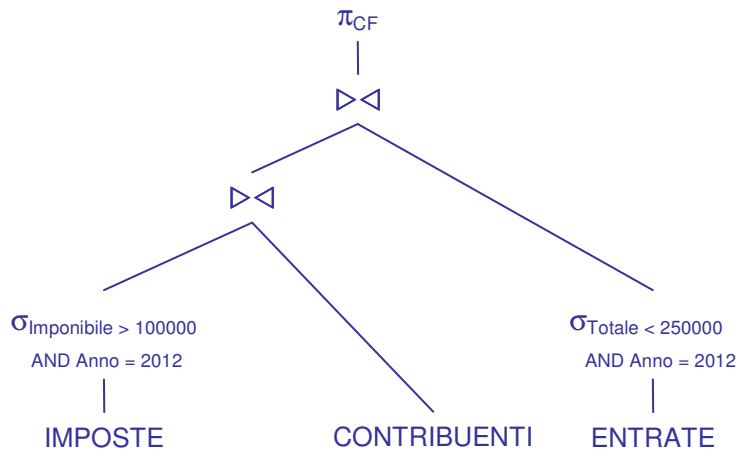
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

```
CONTRIBUENTI(CF,Comune);
ENTRATE(Anno,Comune,Totale);
IMPOSTE(Anno,CF,Imponibile),
      CF REFERENCES CONTRIBUENTI;
-- ogni anno i codici fiscali (CF) presenti in IMPOSTE sono un
-- sottoinsieme di quelli in CONTRIBUENTI
```

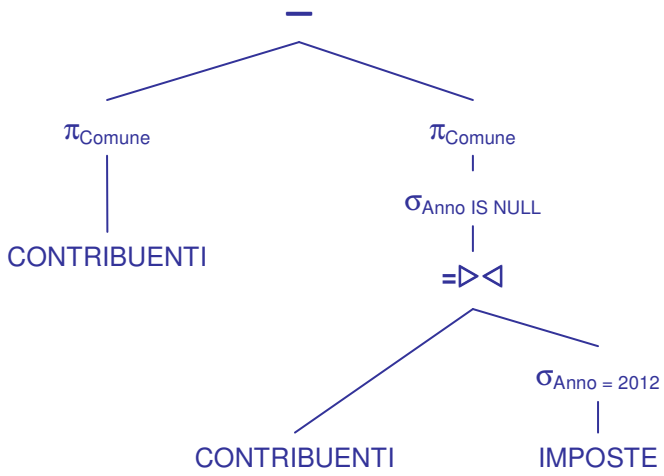
si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] I codici fiscali dei contribuenti con un imponibile maggiore di 100000 € nel 2012 e residenti in comuni che nel 2012 hanno avuto entrate inferiori a 250000 €



La condizione Anno = 2012 applicata a ENTRATE si può omettere, in quanto anni diversi vengono comunque eliminati dal join

1.2) [2 p.] I comuni che nel 2012 hanno avuto tutti i loro contribuenti registrati in IMPOSTE



L'operando destro della differenza calcola i comuni che nel 2012 hanno almeno un contribuente non presente in IMPOSTE

Sistemi Informativi T
19 settembre 2013
Risoluzione

SQL (5 punti totali)

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

2.1) [2 p.] I comuni che nel 2012 hanno avuto tutti i loro contribuenti registrati in IMPOSTE

```
SELECT DISTINCT C.Comune
FROM   CONTRIBUENTI C
WHERE  NOT EXISTS ( SELECT *
                    FROM   CONTRIBUENTI C1
                    WHERE  C1.CF NOT IN ( SELECT I.CF
                                          FROM   IMPOSTE I
                                          WHERE  I.Anno = 2012 ) )
```

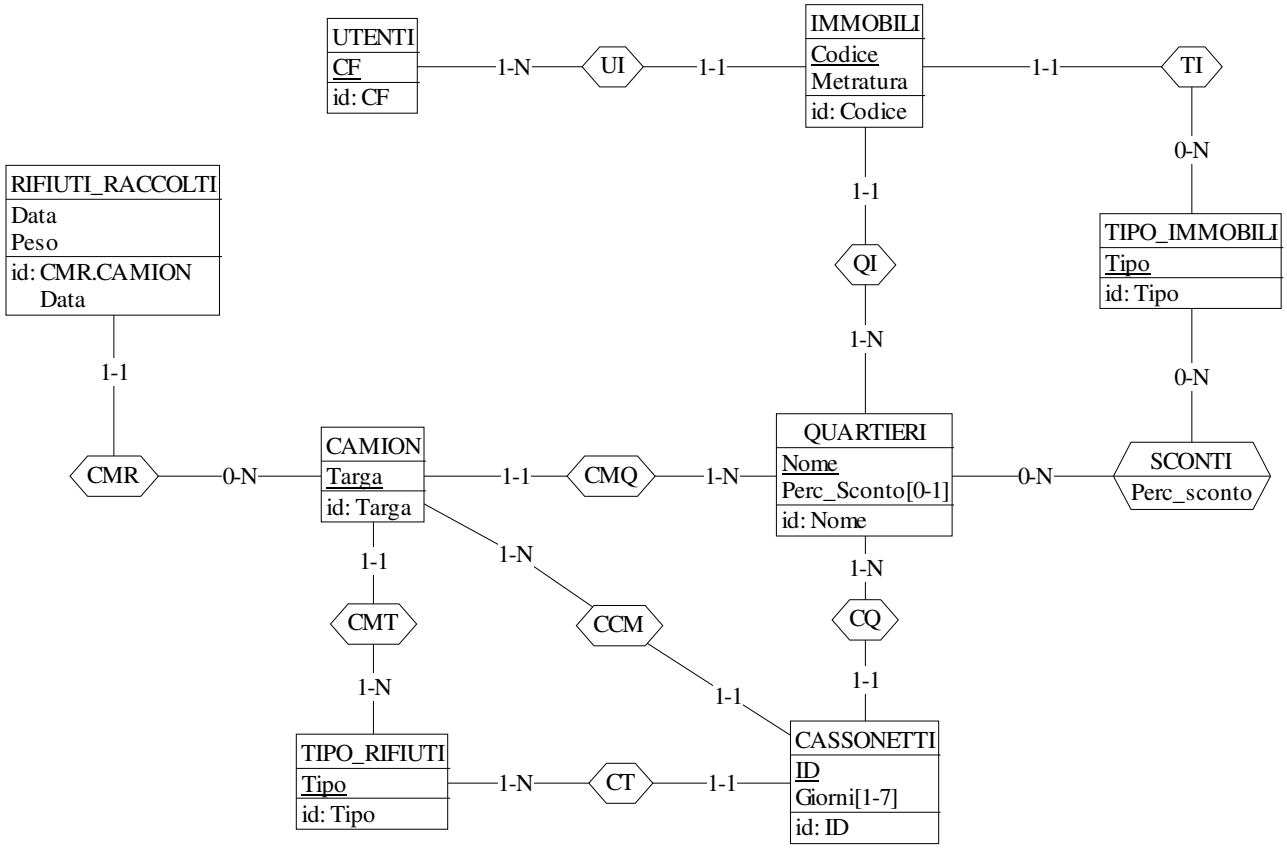
2.2) [3 p.] I comuni che per almeno tre volte hanno incassato in un anno meno dell'anno precedente

```
SELECT E.Comune
FROM   ENTRATE E, ENTRATE EPREC        -- EPREC per l'anno precedente
WHERE  E.Comune = EPREC.Comune
AND    E.Anno = EPREC.Anno + 1
AND    E.Totale < EPREC.Totale
GROUP BY E.Comune
HAVING COUNT(*) >= 3
```

Sistemi Informativi T
19 settembre 2013
Risoluzione

3) Progettazione concettuale (6 punti)

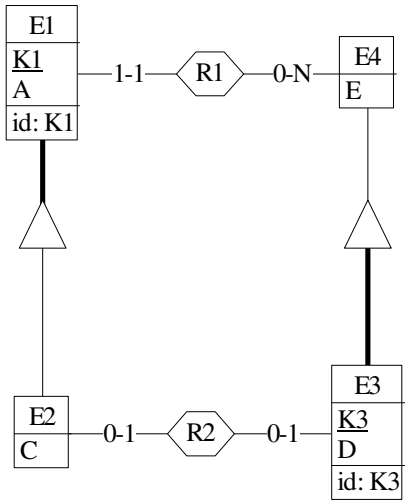
Il Sistema di Raccolta Integrata dei RIfiuti (RIRI) monitora costantemente la produzione di rifiuti riciclabili sul territorio comunale. Ogni cassonetto, destinato a un tipo particolare di rifiuti (vetro, carta, ecc.), viene svuotato una o più volte alla settimana, in giorni prestabiliti, e sempre dallo stesso camion (ogni camion è destinato a un solo tipo di rifiuti e opera su cassonetti di uno stesso quartiere). Ogni volta che un camion termina il suo giro di svuotamento cassonetti i rifiuti raccolti vengono pesati.
In base al volume di rifiuti riciclati viene stabilita per ogni quartiere una percentuale base di sconto sulle tasse per la raccolta rifiuti. Ogni utente registrato nel sistema RIRI è titolare di una o più utenze, per ognuna delle quali, ai fini del calcolo delle tasse, sono noti il quartiere di appartenenza, la tipologia dell'immobile relativo (casa di residenza, casa di proprietà affittata, ecc.) e la sua metratura. La percentuale di sconto applicata ad ogni immobile dipende dalla percentuale base calcolata per il relativo quartiere e dalla tipologia dell'immobile stesso.



Commenti:

- Le associazioni CT e CQ sono ridondanti (composizione, rispettivamente, di CCM e CMT e di CCM e CMQ), ma sono inserite in soluzione sia per chiarezza sia perché indotte dalle specifiche (è in fase di progettazione logica che va considerato se mantenerle o meno).
- L'associazione SCONTI modella al meglio il vincolo che gli sconti applicati agli immobili dipendono solo dallo sconto applicato al quartiere e dal tipo di immobile. La modellazione è leggermente imprecisa, in quanto l'identificatore di QUARTIERI non è Perc_Sconto (e quindi va garantito che due quartieri con la stessa Perc_Sconto abbiano, per ogni tipo di immobile, lo stesso sconto). L'alternativa di creare un'entità con gli sconti dei quartieri, e collegare questa a SCONTI, risolverebbe il problema, ma sembra artificiosa.
- Non è rappresentabile il vincolo che una data in cui un camion raccoglie i rifiuti corrisponde a un giorno della settimana in cui almeno uno dei cassonetti gestiti dal camion viene svuotato.

Sistemi Informativi T
19 settembre 2013
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) l'associazione R1 non viene tradotta separatamente;
c) l'associazione R2 viene tradotta inglobandola in E3;
d) le entità E2 ed E4 non vengono tradotte separatamente;

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,
K3 INT NOT NULL,
TIPO2 SMALLINT NOT NULL CHECK (TIPO2 IN (1,2)),          -- 2: istanza anche di E2
C INT,
CONSTRAINT E2 CHECK ((TIPO2 = 1 AND C IS NULL) OR (TIPO2 = 2 AND C IS NOT NULL))      );

CREATE TABLE E3 (
K3 INT NOT NULL PRIMARY KEY,
D   INT NOT NULL,
K1 INT REFERENCES E1,
TIPO4 SMALLINT NOT NULL CHECK (TIPO4 IN (3,4)),          -- 4: istanza anche di E4
E INT,
CONSTRAINT E4 CHECK ((TIPO4 = 3 AND E IS NULL) OR (TIPO4 = 4 AND E IS NOT NULL))      );

ALTER TABLE E1
ADD CONSTRAINT FKR1 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 ';')

```
CREATE TRIGGER R1_FK
BEFORE INSERT ON E1
REFERENCING NEW AS N
FOR EACH ROW
WHEN (NOT EXISTS ( SELECT * FROM    E3
                           WHERE N.K3 = E3.K3
                           AND     E3.TIPO4 = 4          )
SIGNAL SQLSTATE '70001' ('La tupla inserita non e'' associata a una istanza di E4!');

CREATE TRIGGER R2_FK
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN (N.K1 IS NOT NULL
        AND NOT EXISTS ( SELECT * FROM    E1
                                 WHERE N.K1 = E1.K1
                                 AND     E1.TIPO2 = 2          )
SIGNAL SQLSTATE '70002' ('La tupla inserita non e'' associata a una istanza di E2!');

CREATE TRIGGER K1_UNIQUE
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS ( SELECT * FROM    E3
                       WHERE N.K1 = E3.K1   )
SIGNAL SQLSTATE '70003' ('K1 non ammette valori duplicati!');
```