Sistemi Informativi T
4 luglio 2013
Risoluzione

Tempo a disposizione: 2:30 ore

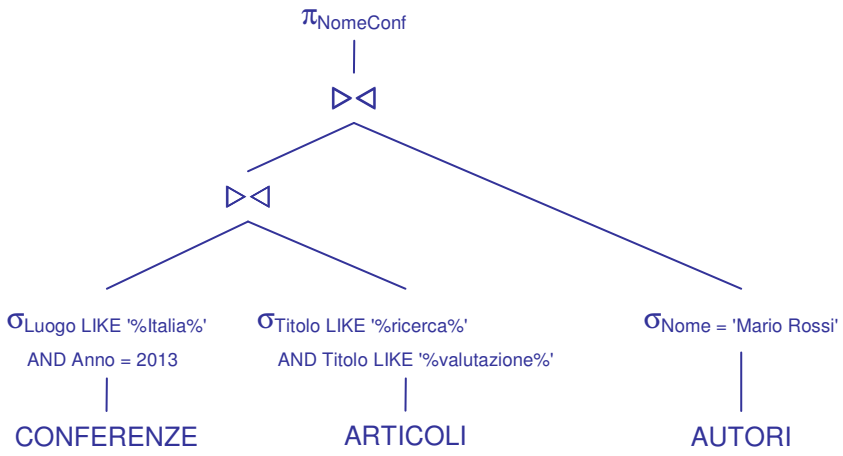
1) Algebra relazionale (3 punti totali):

Date le seguenti relazioni:

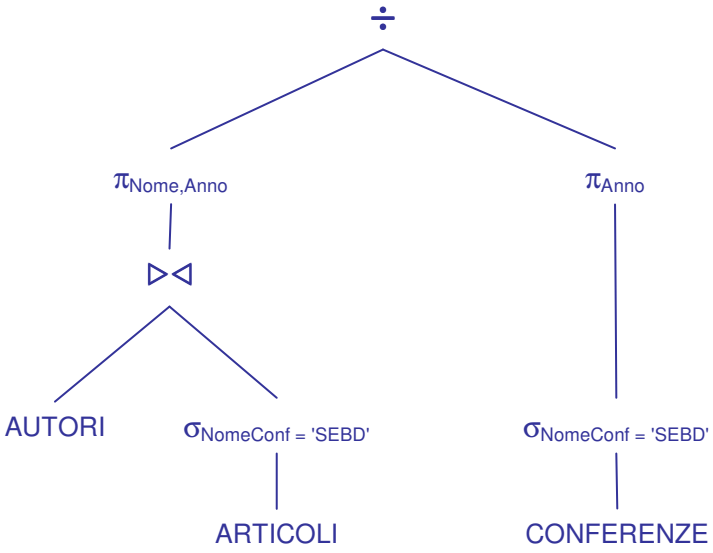
```
CONFERENZE(NomeConf,Anno,Luogo);
ARTICOLI(ArtID,Titolo,NomeConf,Anno),
NomeConf,Anno REFERENCES CONFERENZE;
AUTORI(ArtID,Nome),
ArtID REFERENCES ARTICOLI;
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] Le conferenze tenutesi in Italia nel 2013, in cui compare un articolo di Mario Rossi il cui titolo contiene, in ordine qualsiasi, le parole 'ricerca' e 'valutazione'



1.2) [2 p.] Gli autori che tutti gli anni hanno pubblicato almeno un articolo nella conferenza 'SEBD'



Si noti che la selezione è necessaria anche nel dividendo.

Sistemi Informativi T
4 luglio 2013
Risoluzione

SQL (5 punti totali)

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

2.1) [2 p.] Gli autori che tutti gli anni hanno pubblicato almeno un articolo nella conferenza 'SEBD'

```
SELECT AU.NOME
FROM  AUTORI AU, ARTICOLI AR
WHERE AU.ArtID = AR.ArtID
AND   AR.NomeConf = 'SEBD'
GROUP BY AU.NOME
HAVING COUNT(DISTINCT AR.Anno) = ( SELECT COUNT(*)
                                   FROM   CONFERENZE C
                                   WHERE  C.NomeConf = 'SEBD' )
```

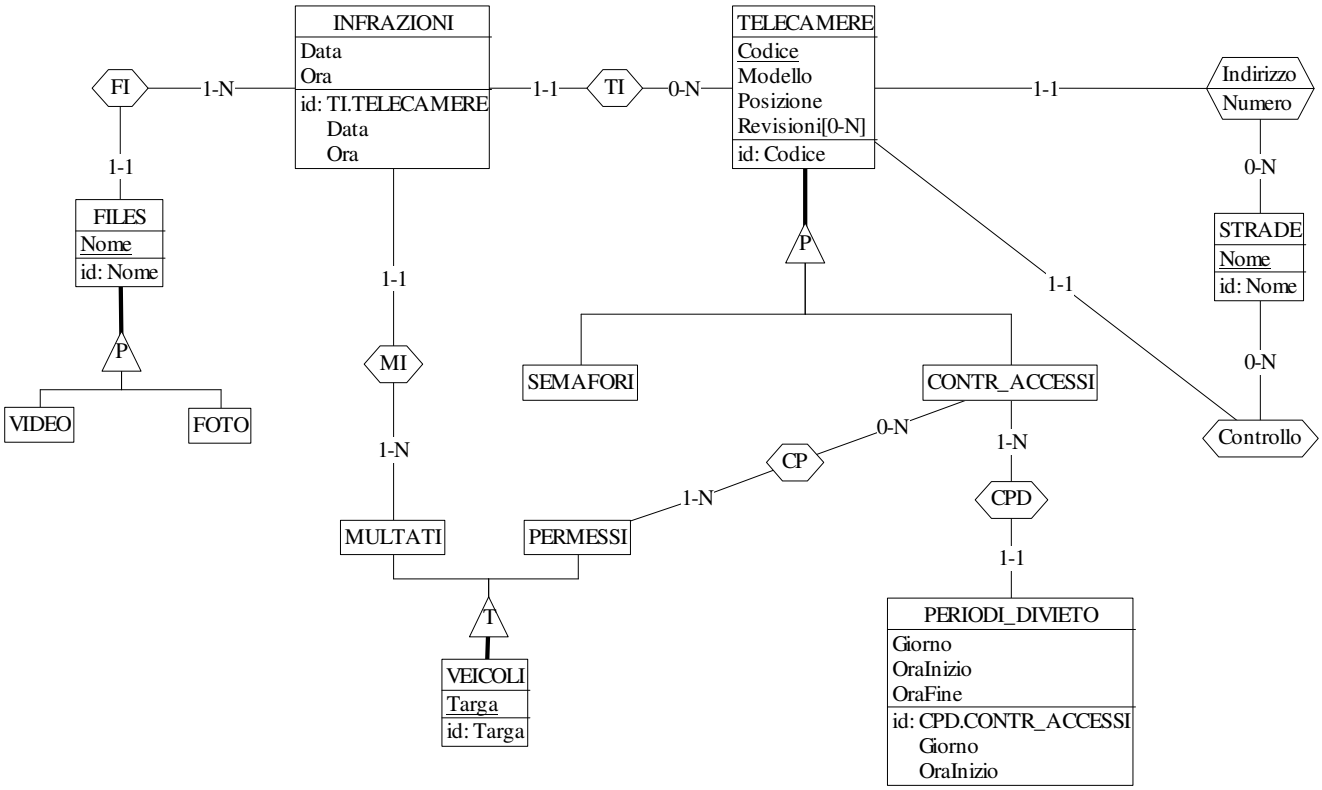
2.2) [3 p.] Per ogni conferenza italiana, l'autore che ha pubblicato nel maggior numero di anni

```
WITH   CONFAUTORI (CONF,AUTORE,NUMANNI) AS (
SELECT AR.NomeConf, AU.Nome, COUNT(DISTINCT AR.Anno)
FROM   AUTORI AU, ARTICOLI AR, CONFERENZE C
WHERE  AR.ArtID = AU.ArtID
AND    AR.NomeConf = C.NomeConf
AND    AR.Anno = C.Anno
AND    C.Luogo LIKE '%Italia%'
GROUP BY AR.NomeConf, AU.Nome )

SELECT CA.CONF, CA.AUTORE
FROM   CONFAUTORI CA
WHERE  CA.NUMANNI = ( SELECT MAX(CA1.NUMANNI)
                      FROM   CONFAUTORI CA1
                      WHERE  CA1.CONF = CA.CONF )

-- Nella common table expression si sfrutta la forma COUNT(DISTINCT ...)
-- per contare una volta sola gli anni in cui un dato autore ha
-- pubblicato piu' di 1 lavoro nella stessa conferenza
```

Sistemi Informativi T
4 luglio 2013
Risoluzione

3) Progettazione concettuale (6 punti)

Il sistema AlphaCentauri (AC) ha lo scopo di rilevare infrazioni al codice stradale, controllando mediante telecamere semafori e strade ad accesso limitato.
Di ogni telecamera, identificata da un codice univoco, è noto il modello, l'ubicazione (ovvero l'indirizzo e il posizionamento specifico, ad es. altezza da terra), la strada controllata, e tutte le date in cui la telecamera è stata revisionata. Tutti i nomi delle strade fanno riferimento allo stradario ufficiale del Comune.
Per le telecamere dedite al controllo degli accessi vengono specificati, per ogni giorno della settimana, i periodi (uno o più) di inizio e fine del divieto di accesso, oltre a un elenco di targhe esentate dal divieto.
Ogni infrazione rilevata viene registrata da AC memorizzando la data e l'ora di rilevamento, la targa del veicolo interessato e uno o più file di documentazione (video o foto).



Commenti:

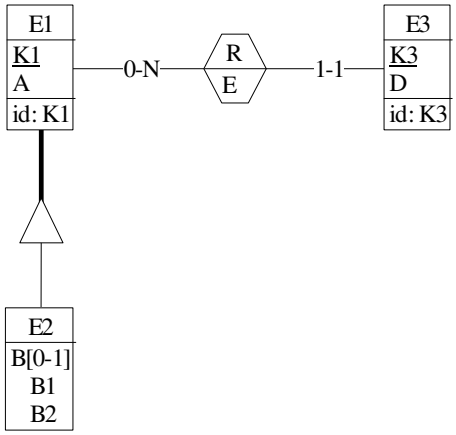
- Lo schema, solo apparentemente complesso, si caratterizza per la presenza di 3 gerarchie, di cui solo quella radicata in TELECAMERE è essenziale.
- Si noti la doppia associazione tra TELECAMERE e STRADE.
- Lo schema ER non esprime il vincolo che un veicolo non può essere multato per l'accesso a una strada per cui ha il relativo permesso.

Sistemi Informativi T
4 luglio 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 R non viene tradotta separatamente;
c) le entità E1 ed E2 non vengono tradotte separatamente;
d) un'istanza di E3 può essere associata tramite R a un'istanza di E2 solo se il valore di B è definito;



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,
TIPO2 SMALLINT NOT NULL CHECK (TIPO2 IN (0,1)),          -- 1: istanza anche di E2
B1 INT,
B2 INT,
CONSTRAINT E2 CHECK (TIPO2 = 0 AND B1 IS NULL AND B2 IS NULL),
CONSTRAINT B CHECK ((B1 IS NOT NULL AND B2 IS NOT NULL) OR
                    (B1 IS NULL AND B2 IS NULL))                    );

CREATE TABLE E3 (
K3 INT NOT NULL PRIMARY KEY,
D   INT NOT NULL,
K1R INT NOT NULL REFERENCES E1,
E   INT NOT NULL  );
```

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 definire il seguente trigger:

CREATE TRIGGER PUNTO_D
BEFORE INSERT ON E3
REFERENCING NEW AS N
FOR EACH ROW
WHEN (EXISTS ( SELECT *
               FROM    E1
               WHERE N.K1R = E1.K1
               AND     E1.TIPO2 = 1        -- Se TIPO2 = 0 il vincolo e' sempre rispettato
               AND     E1.B1 IS NULL)    )
SIGNAL SQLSTATE '70001' ('La tupla inserita e'' associata a una istanza di E2 con B nullo!');
```