

Prima prova di verifica del 15/12/2010

1. (Obbligatorio) Si considerino le relazioni $S(D :string, E :int)$ e $R(A :string, B :string, C :int)$:

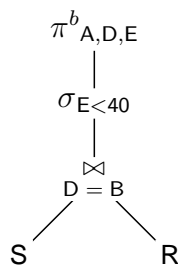
<i>D</i>	<i>E</i>
b1	10
b2	40
b3	30

<i>A</i>	<i>B</i>	<i>C</i>
a1	b1	10
a1	b2	20
a2	b2	20
a1	b3	30
a2	b3	30

Si diano (a) l'albero logico della seguente interrogazione, (b) il tipo del risultato e (c) il valore del risultato:

```
SELECT A, D, E
FROM   S, R
WHERE  D = B AND E < 40;
```

Albero logico



Tipo risultato (multinsieme): $\{(A :string, D :string, E :int)\}$

<i>A</i>	<i>D</i>	<i>E</i>
a1	b1	10
a1	b3	30
a2	b3	30

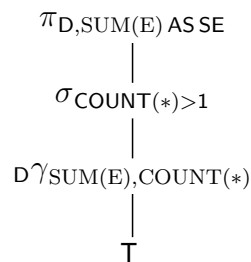
Sia T il risultato della precedente interrogazione. Si diano (a) l'albero logico della seguente interrogazione, (b) il tipo del risultato e (c) il valore del risultato:

```

SELECT    D, SUM(E) AS SE
FROM      T
GROUP BY  D
HAVING    COUNT(*) > 1;

```

Albero logico



Tipo risultato: $\{(D : \text{string}, SE : \text{int})\}$

<i>D</i>	<i>SE</i>
b3	60

2. Si vuole utilizzare una base di dati per gestire le seguenti informazioni sui servizi turistici offerti dalle città italiane.

Di una città interessano il nome, provincia, regione, prefisso telefonico e numero di asterischi (da zero a due) che ne sottolineano l'importanza turistica. Se è stata classificata dall'UNESCO come Patrimonio dell'Umanità, interessa da quale anno e i motivi della scelta (nomi dei siti di eccezionale importanza da un punto di vista culturale o naturale).

Di ogni città interessa conoscere da quali aeroporti è servita e a quale distanza si trovano. Per ogni aeroporto interessano inoltre il nome, l'indirizzo e la sigla.

Di ogni città interessano anche i monumenti e musei da visitare. Per ciascuno di questi interessano indirizzo, recapiti telefonici, orario, giorni e periodi di chiusura.

Dei servizi interessano il codice, nome, indirizzo e recapito telefonico, in quale zona della città si trovano (centro, periferia o dintorni) e, se accettano pagamenti con carta di credito, le sigle di quelle accettate. Un servizio può essere un albergo o un ristorante. Di un albergo interessa la categoria e il numero delle stanze. Di un ristorante interessa il tipo di cucina e il numero di coperti.

- (a) Si definisca lo schema concettuale della base di dati (Figura 1).

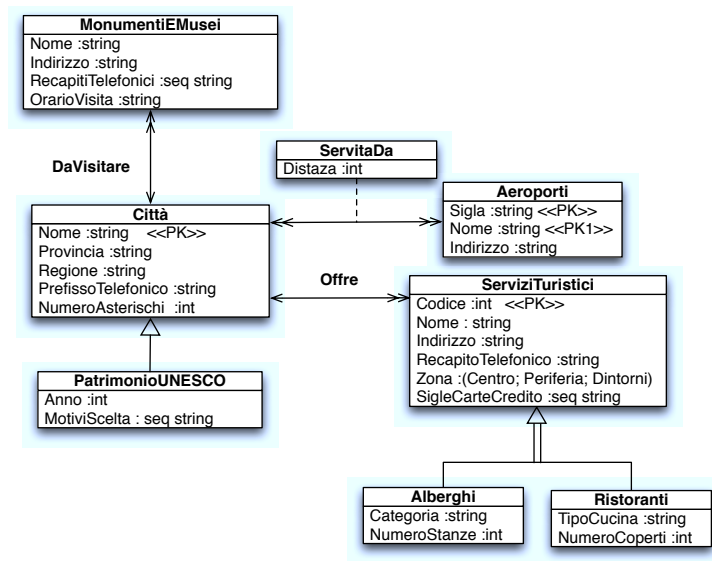


Figura 1: Schema concettuale

(b) Si traduca lo schema concettuale in uno schema relazionale grafico, definendo gli attributi delle chiavi primarie e delle chiavi esterne (Figura 2).

3. Si consideri il seguente schema relazionale:

Dipartimenti (IdDip, Nome, Città)

Impiegati (IdImp, Nome, AnnoNascita, Sesso, Stipendio, IdDip*)

ImpiegatiProgetti (IdImp*, IdProg*)

Progetti(IdProg, Nome, DataInizio, DataFine, Budget)

(a) Si rappresenti graficamente lo schema relazionale della base di dati.

(b) Si scrivano le interrogazioni SQL che restituiscono le seguenti informazioni:

i. Per ogni dipartimento in cui lavora qualche impiegato nato dopo il 1980, si restituisca l'IdDip, il nome e la città.

```

SELECT DISTINCT d.IdDip, d.Nome, d.Città
FROM Dipartimenti d, Impiegati i
WHERE d.IdDip = i.IdDip
AND i.AnnoNascita > 1980;
  
```

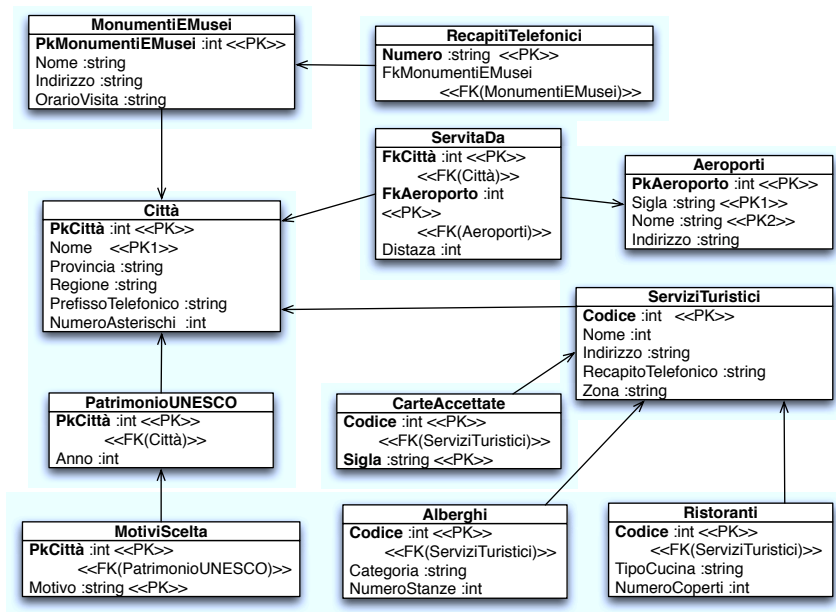


Figura 2: Schema Logico

- ii. Per ogni dipartimento in cui lavorano solo impiegati nati dopo il 1980, si restituisca l'IdDip, il nome e la città.¹

```

SELECT d.IdDip, d.Nome, d.Città
FROM Dipartimenti d
WHERE FOR ALL i IN Impiegati WHERE d.IdDip = i.IdDip
      : i.AnnoNascita > 1980;
  
```

```

SELECT d.IdDip, d.Nome, d.Città
FROM Dipartimenti d
WHERE NOT EXISTS
  (SELECT *
   FROM Impiegati i
   WHERE d.IdDip = i.IdDip
        AND NOT (i.AnnoNascita > 1980));
  
```

Altra soluzione:

```

SELECT d.IdDip, d.Nome, d.Città
FROM Dipartimenti d, Impiegati i
  
```

¹Si assume che non esistano dipartimenti senza impiegati

```

WHERE    d.IdDip = i.IdDip
GROUP BY d.IdDip, d.Nome, d.Città
HAVING   MIN(i.AnnoNascita) > 1980;

```

- iii. Per ogni dipartimento in cui lavorano almeno dieci impiegati, si restituisca il nome del dipartimento, il numero di impiegati, lo stipendio minimo e lo stipendio massimo.

```

SELECT   d.Nome, COUNT(*) AS NumImp, MIN(Stipendio), MAX(Stipendio)
FROM     Dipartimenti d, Impiegati i
WHERE    d.IdDip = i.IdDip
GROUP BY d.IdDip, d.Nome
HAVING   COUNT(*) > 9;

```

- iv. Per tutte le coppie di progetti che condividono almeno un impiegato, restituire i nomi dei due progetti, senza ripetizioni.

```

SELECT p1.IdProg, p2.IdProg
FROM   Progetti p1, Progetti p2
WHERE  p1.IdProg > p2.IdProg AND
       FOR SOME IP1 IN ImpiegatiProgetti, IP2 IN ImpiegatiProgetti
         WHERE p1.IdProg = IP1.IdProg AND p2.IdProg = IP2.IdProg:
         : IP1.IdImp = IP2.IdImp;

```

```

SELECT p1.IdProg, p2.IdProg
FROM   Progetti p1, Progetti p2
WHERE  p1.IdProg > p2.IdProg AND
       EXISTS(
         SELECT *
         FROM   ImpiegatiProgetti IP1, ImpiegatiProgetti IP2
         WHERE  p1.IdProg = IP1.IdProg AND p2.IdProg = IP2.IdProg AND
                IP1.IdImp = IP2.IdImp);

```

Altra soluzione:

```

SELECT p1.IdProg, p2.IdProg
FROM   Progetti p1, Progetti p2, ImpiegatiProgetti ip1, ImpiegatiProgetti ip2
WHERE  p1.IdProg > p2.IdProg AND
       p1.IdProg = ip1.IdProg AND p2.IdProg = ip2.IdProg AND
       ip1.IdImp = ip2.IdImp;

```

- v. (Opzionale) Per quei progetti in cui lavorano almeno dieci impiegati di sesso femminile, si riporti il nome del progetto ed il numero totale di impiegati.

```

SELECT      p.Nome, COUNT(*) AS NumImpiegati
FROM        Impiegati i, ImpiegatiProgetti ip, Progetti p
WHERE       i.IdImp = ip.IdImp AND p.IdProg = ip.IdProg AND
           10 <= ( SELECT COUNT(*)
                   FROM    Impiegati i2, ImpiegatiProgetti ip2
                   WHERE   i2.IdImp = ip2.IdImp AND ip2.IdProg = p.IdProg
                   AND i2.Sesso = 'F' )
GROUP BY    p.IdProg, p.Nome;

```

- vi. (Opzionale) Restituire il nome e l'Id di quei dipartimenti in cui tutti gli impiegati lavorano in qualche progetto la cui DataFine sia minore di 31/12/2010.

```

SELECT      d.Nome, d.IdDip
FROM        Dipartimenti d
WHERE       FOR ALL i IN Impiegati WHERE i.IdDip = d.IdDip
           : (FOR SOME ip IN ImpiegatiProgetti, p IN Progetti
              WHERE ip.IdProg = p.IdProg AND i.IdImp = ip.IdImp
              : p.DataFine < 31/12/2010;

```

```

SELECT      d.Nome, d.IdDip
FROM        Dipartimenti d
WHERE       NOT EXISTS(
           SELECT *
           FROM    Impiegati i,
           WHERE   i.IdDip = d.IdDip AND
           NOT EXISTS(
           SELECT *
           FROM    ImpiegatiProgetti ip, Progetti p
           WHERE   ip.IdProg = p.IdProg AND i.IdImp = ip.IdImp
           AND p.DataFine < 31/12/2010));

```