Informatica per le Scienze Umane - Corso di Laurea in Lettere
Verifica in itinere del 20.11.2008 - Docente: Mirco Nanni

Soluzioni

Si consideri la seguente base di dati relazionale che descrive un insieme di eventi, luoghi in cui essi avvengono e personaggi che vi prendono parte.

TABLE Luoghi (codice: integer PRIMARY KEY, stato: varchar(50), regione: varchar(50), città: varchar(50))	TABLE Personaggi (codice: integer PRIMARY KEY, nome: varchar(50), annoNascita: integer, coniuge: integer REFERENCES Personaggi(codice), luogoNascita: integer REFERENCES Luoghi(codice))
TABLE Eventi (codice: char(10) PRIMARY KEY, descrizione: varchar(100), luogo: integer REFERENCES Luoghi(codice), tipologia: varchar(50), anno: integer)	TABLE PersonaggiEventi (evento: char(10) REFERENCES Eventi(codice), personaggio: integer REFERENCES Personaggi(codice), ruolo: varchar(100))

L'attributo “Personaggi.coniuge” indica il coniuge di ogni personaggio, e contiene NULL in caso esso non sia coniugato. Inoltre “Eventi.tipologia” descrive la tipologia in cui ricade ogni evento, ad esempio “battaglia”, “discorso”, “missione di pace”, ecc.

---

1. Elencare gli eventi che hanno avuto luogo a Pisa tra il 1939 e il 1945. **(5 punti)**

SELECT Eventi.descrizione
FROM Eventi JOIN Luoghi ON Eventi.luogo = Luoghi.codice
WHERE Luoghi.città = “Pisa” AND Eventi.anno ≥ 1939 AND Eventi.anno ≤ 1945

2. Elencare gli eventi che coinvolgono solo personaggi nati in Italia. **(7 punti)**

```
SELECT Eventi.descrizione
FROM Eventi JOIN PersonaggiEventi ON Eventi.codice = PersonaggiEventi.luogo

EXCEPT

SELECT Eventi.descrizione
FROM Eventi JOIN PersonaggiEventi ON Eventi.codice = PersonaggiEventi.luogo
        JOIN Personaggi ON PersonaggiEventi.personaggio = Personaggi.codice
        JOIN Luoghi ON Personaggi.luogoNascita = Luoghi.codice
WHERE Luoghi.stato ≠ "Italia"
```

Nota: la JOIN della prima sotto-interrogazione serve a selezionare solo gli eventi a cui sia associato almeno un personaggio, del quale però non ci interessa l'identità. Soluzioni alternative, che assumano non ci siano tali eventi (per le quali, quindi, la JOIN non occorre) o che ricostruiscano l'identità dei personaggi e il loro luogo di nascita (con conseguenti due JOIN aggiuntive), sono comunque da ritenersi corrette.

3. Elencare i personaggi che sono coinvolti in eventi accaduti nella loro città natale. **(6 punti)**

```
SELECT Personaggi.nome
FROM Luoghi JOIN Personaggi ON Luoghi.codice = Personaggi.luogoNascita
        JOIN PersonaggiEventi ON Personaggi.codice = PersonaggiEventi.personaggio
        JOIN Eventi ON PersonaggiEventi.evento = Eventi.codice
        JOIN Luoghi AS LuoghiEventi ON Eventi.luogo = LuoghiEventi.codice
WHERE Luoghi.città = LuoghiEventi.città
```

Nota: eventuali condizioni più complesse, che controllano anche "stato" e "regione", o semplicemente "codice", sono ugualmente corrette.

4. Elencare gli stati a cui non sono associati eventi di tipo "battaglia" dopo il 1950. **(6 punti)**

```
SELECT Luoghi.stato
FROM Luoghi

EXCEPT

SELECT Luoghi.stato
FROM Luoghi JOIN Eventi ON Luoghi.codice = Eventi.luogo
WHERE Eventi.tipologia = "battaglia"  AND  Eventi.anno > 1950
```

5. Elencare le coppie di personaggi coniugati, entrambi partigiani di una qualche battaglia (ovvero: coinvolti in eventi di tipo "battaglia" col ruolo di "partigiano"). Nota: la battaglia in questione non deve essere necessariamente la stessa per i due personaggi. **(2 punti)**

```
SELECT Personaggi.nome, Personaggi2.nome
FROM Eventi JOIN PersonaggiEventi ON Eventi.codice = PersonaggiEventi.evento
        JOIN Personaggi ON PersonaggiEventi.personaggio = Personaggi.codice
        JOIN Personaggi AS Pers2 ON Personaggi.coniuge = Pers2.codice
        JOIN PersonaggiEventi AS PersEventi2 ON Pers2.codice = PersEventi2.personaggio
        JOIN Eventi AS Eventi2 ON PersEventi2.evento = Eventi2.codice
WHERE Eventi.tipologia = "battaglia" AND PersonaggiEventi.ruolo = "partigiano"  AND
        Eventi2.tipologia = "battaglia" AND PersEventi2.ruolo = "partigiano"
```

6. Si estenda la base di dati (aggiungendo nuove tabelle e/o modificando quelle esistenti) in modo da poter associare ad ogni personaggio i suoi eventuali figli (o figlie), anche loro intesi come personaggi descritti nella base di dati. **(6 punti)**

a. Soluzione più compatta: modificare la tabella Personaggi, aggiungendo un attributo che indichi il padre del personaggio, ed uno che ne indichi la madre. Infatti, mentre un personaggio puo' avere un numero qualunque di figli, esso ha solo un padre ed una madre (limitandoci a quelli biologici):

```
TABLE Personaggi (
        codice: integer PRIMARY KEY,
        nome: varchar(50),
        annoNascita: integer,
        coniuge: integer REFERENCES Personaggi(codice),
        luogoNascita: integer REFERENCES Luoghi(codice)
        padre: integer REFERENCES Personaggi(codice),
        madre: integer REFERENCES Personaggi(codice)
)
```

b. Soluzione più complessa: aggiungere una tabella che descrive l'associazione figli ↔ genitori:

```
TABLE GenitoriFigli (
        genitore: integer REFERENCES Personaggi(codice),
        figlio: integer REFERENCES Personaggi(codice),
        PRIMARY KEY (genitore, figlio)  )
```

c. Ibrido delle precedenti: tabella aggiuntiva che associa padre e madre ad ogni figlio:

```
TABLE GenitoriFigli (
        figlio: integer  PRIMARY KEY     REFERENCES Personaggi(codice),
        padre: integer REFERENCES Personaggi(codice),
        madre: integer REFERENCES Personaggi(codice)  )
```