

Basi di dati — Homework 24 maggio 2006

Soluzioni

Domanda 1 (20%) Si consideri una base di dati sulle relazioni

- $R_1(\underline{A}, B, C)$
- $R_2(\underline{D}, \underline{E}, F)$

Scrivere interrogazioni in SQL equivalenti alle seguenti espressioni dell'algebra relazionale:

1. $\pi_{BC}(\sigma_{C>10}(R_1))$
2. $\pi_B(R_1 \bowtie_{C=D} \sigma_{F=2}(R_2))$

Possibile soluzione

1. SELECT DISTINCT B , C
FROM R1
WHERE C > 10
2. SELECT DISTINCT B
FROM R1, R2
WHERE C = D AND F = 2

Domanda 2 (20%) Con riferimento alla base di dati nella domanda 1 scrivere espressioni dell'algebra relazionale equivalenti alle seguenti interrogazioni SQL

1. SELECT DISTINCT A , B
FROM R1, R2
WHERE C = D AND E > 100
2. SELECT DISTINCT A , B
FROM R1 X1
WHERE NOT EXISTS
(SELECT *
FROM R1 Y1, R2
WHERE Y1.C = D AND X1.A = Y1.A AND F>10)

Possibile soluzione

1. $\pi_{AB}(R_1 \bowtie_{C=D} \sigma_{E>100}(R_2))$
2. $\pi_{AB}(R_1) - \pi_{AB}(R_1 \bowtie_{C=D} \sigma_{F>10}(R_2))$

Nota: le proiezioni si possono fare direttamente su AB perché A è chiave, e quindi ad ogni valore di A è associato un solo valore di B ; se così non fosse stato, le due proiezioni avrebbero dovuto essere su A e poi sul risultato della differenza si sarebbe dovuto eseguire un join con R_1 .

Domanda 3 (10%) Ancora con riferimento alla base di dati nella domanda 1, indicare, per ciascuna delle seguenti interrogazioni, se la parola chiave **DISTINCT** è necessaria

1. l'interrogazione 1 nella domanda 2
2. l'interrogazione 2 nella domanda 2
3. SELECT DISTINCT A , B
FROM R1, R2
WHERE B = D AND C = E
4. SELECT DISTINCT B , C
FROM R1, R2
WHERE B = D AND C = E

Possibile soluzione

1. SÌ
2. NO
3. NO
4. SÌ

Domanda 4 (30%) Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```
create table Studenti (  
    matricola numeric not null primary key,  
    cognome char(20) not null,  
    nome char(20) not null,  
    età numeric not null  
);  
create table Esami (  
    codiceCorso numeric not null,  
    studente numeric not null references Studenti(matricola),  
    data date not null,  
    voto numeric not null,  
    primary key (codiceCorso, studente, data));
```

Si supponga che vengano registrati anche gli esami non superati, con voti inferiori al 18.
Formulare in SQL

1. l'interrogazione che trova gli studenti che hanno riportato in almeno un esame un voto più alto di quello ottenuto, in tale esame, da Archimede Pitagorico
2. l'interrogazione che trova, per ciascuno studente, il numero di esami superati e la relativa media
3. l'interrogazione che trova lo studente con la media più alta
4. l'interrogazione che trova gli studenti che non hanno superato esami (proporre due soluzioni, una che utilizza l'operatore EXCEPT e l'altra che non lo utilizza)

Possibile soluzione

Può essere utile la vista:

```
create view esamiStudOK  
as select *  
from studenti join esami on matricola = studente  
where voto >= 18
```

1. Supponendo si tratti di esami superati:

```
select s.matricola, s.cognome, s.codiceCorso, s.voto, a.voto as votoArc  
from esamiStudOK s, esamiStudOK a  
where a.cognome='Pitagorico' and a.nome='Archimede'  
and s.codiceCorso = a.codiceCorso and s.voto > a.voto
```
2.

```
select matricola, cognome, nome, count(*) as numEsami, avg(voto) as media  
from esamiStudOK  
group by matricola, cognome, nome
```
3.

```
select matricola, cognome, nome, avg(voto) as media  
from esamiStudOK  
group by matricola, cognome, nome  
having avg(voto) >= all (select avg(voto) from esamiStud group by matricola)
```

oppure (alcuni sistemi non accettano la nidificazione nella having:

```
create view studMedia (studente, media)  
as select studente, avg(voto)  
from esami  
where voto >= 18  
group by studente
```

```
select matricola, cognome, nome, media  
from studenti, studMedia  
where matricola = studente  
and media >= all (select media from studMedia)
```

oppure ancora, sempre con la vista studMedia

```
select matricola, cognome, nome, media  
from studenti, studMedia
```

```

where matricola = studente
  and media = (select max(media) from studMedia)

```

```

4. select matricola , cognome, nome
   from studenti
   except
   select matricola , cognome, nome
   from esamiStudOK
   oppure
   select matricola , cognome, nome
   from studenti s
   where not exists (
     select *
     from esamistud e
     where s.matricola = e.matricola)

```

Domanda 5 (20%) Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```

create table impiegati (
  cf numeric not null primary key,
  titolo char(5),
  cognome char(20) not null,
  nome char(20) not null,
  stipendio numeric not null,
  dip numeric not null references dipartimenti(codice)
);
create table Dipartimenti (
  codice numeric not null primary key,
  nomeDip char(20) not null unique,
  indirizzo char(30) not null );

```

e il metodo mostrato sotto, che avrebbe lo scopo di stampare, per ciascun dipartimento, le informazioni sintetiche (nome e indirizzo) e, subito dopo, la lista degli impiegati.

Riguardo a tale metodo:

1. il codice SQL contiene un errore, a causa del quale la stampa può non avvenire correttamente; individuare tale errore e correggerlo
2. modificare il metodo in modo che stampi anche le informazioni sui dipartimenti che non hanno impiegati (che sono invece ignorati nella versione attuale)

```

static void stampaDati(Connection connection) throws SQLException {
  Statement statement = connection.createStatement();
  String query = "select * from dipartimenti JOIN impiegati ON codice = dip " ;
  ResultSet resultSet = statement.executeQuery(query);
  boolean primo = true ;
  int codice = 0;
  int codicePrec = 0;
  while (resultSet.next()){
    primo = false ;
    codice = resultSet.getInt("codice");
    if (primo || !(codice==codicePrec)){
      System.out.println("Dipartimento: " + codice +
        " Nome: " + resultSet.getString("NomeDip") +
        " Indirizzo: " + resultSet.getString("Indirizzo") +
        "\nImpiegati");
    }
    String titolo = resultSet.getString("titolo");
    // Nota bene (dalla documentazione di Java)
    // boolean wasNull(): reports whether the last column read had a value of SQL NULL.
    if ((resultSet.wasNull())){ titolo = " ";}
    System.out.println(" " + resultSet.getInt("cf") + " " + titolo +
      resultSet.getString("Nome") + " " + resultSet.getString("Cognome"));
  }
}

```

```
        codicePrec = codice ;  
    }  
}
```

Possibile soluzione

1. l'interrogazione restituisce il risultato secondo un ordine non prevedibile; poiché è necessario ottenere uno dopo l'altro tutti gli impiegati di uno stesso dipartimento, è necessario essere sicuri di generarli in ordine; l'interrogazione va sostituita con la seguente:
`select * from dipartimenti JOIN impiegati ON codice = dip ORDER BY dip`
2. il join deve diventare esterno (sinistro) e nella scansione debbono essere ignorate le ennuple per le quali le informazioni sull'impiegato sono assenti (valori nulli).