

Capitolo 6

SQL PER L'USO INTERATTIVO DI BASI DI DATI

Soluzione degli esercizi

1. La seguente espressione restituisce true se i valori di A sono tutti diversi, false altrimenti.

```
SELECT    COUNT(*) = COUNT(DISTINCT A)
FROM      R;
```

Si noti che per stabilire se un insieme di attributi {B, C} determina l'attributo A, si controlla se la seguente interrogazione ritorna una tabella vuota:

```
SELECT    B, C, COUNT(DISTINCT A)
FROM      R
GROUP BY  B, C
HAVING    COUNT(DISTINCT A) > 1;
```

2. Vedi testo.
3. Vedi testo.
4. Di solito un'interrogazione può essere espressa in SQL in modi diversi. Solo in qualche caso si mostrano più soluzioni.

```
(a) SELECT    Nome
FROM          Studenti, Esami
WHERE         Matricola = MatricolaStudente AND Provincia = 'Pisa'
AND          Materia = 'Programmazione' AND Voto = 30;

(b) SELECT    Nome
FROM          Studenti, Esami
WHERE         MatricolaStudente = Matricola
GROUP BY     Nome
HAVING       COUNT(*) = 5;
oppure:
SELECT       Nome
FROM         Studenti
WHERE        5 = ( SELECT COUNT(*)
                  FROM   Esami
                  WHERE  MatricolaStudente = Matricola);
```

Questa soluzione è meno efficiente della precedente perché la sottoselect viene calcolata per ogni record della tabella Studenti.

- ```
(c) SELECT Matricola, Nome, COUNT(*) AS EsamiSuperati,
 MAX(Voto), MIN(Voto), AVG(Voto)
FROM Studenti, Esami
WHERE Matricola = MatricolaStudente AND Provincia = 'Pisa'
GROUP BY Matricola, Nome;

(d) SELECT Materia, COUNT(*) AS NumeroEsami,
 MAX(Voto), MIN(Voto), AVG(Voto)
FROM Esami
WHERE NumAppello = 1
GROUP BY Materia;
```

5. Lo schema relazionale è mostrato in Figura 6.1. Solo in alcuni casi si mostrano più soluzioni per le interrogazioni, fra quelle possibili. Per poter provare queste ed altre interrogazioni nel sistema JRS, i comandi per la definizione della base di dati e per inserire dei dati sono disponibili nel file <http://fondamentidibasedidati.it/AziendaPerJRS.zip>. Tutte le interrogazioni assu-

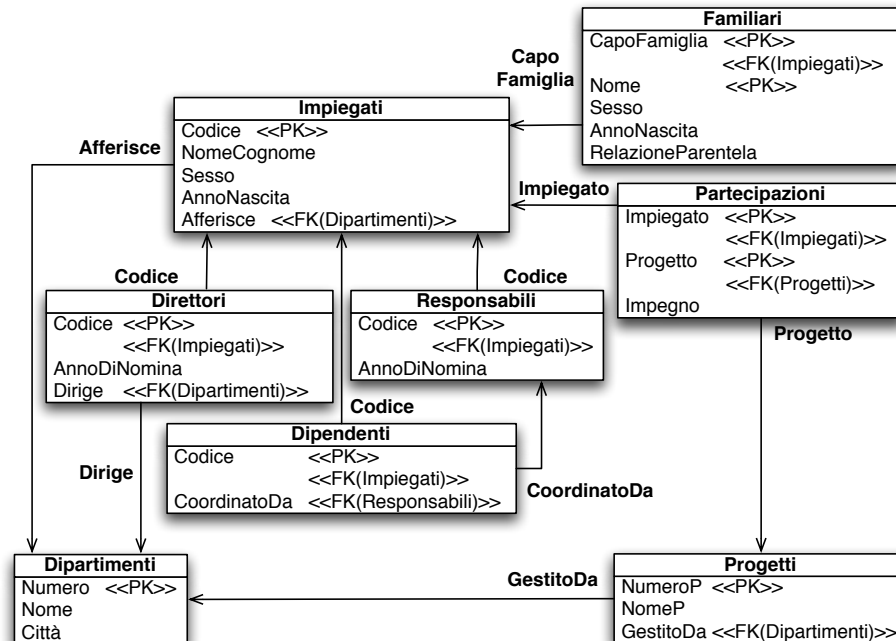


Figura 6.1. Gestione di dati aziendali: progetto relazionale

mono l'esistenza delle seguenti viste:

```
CREATE VIEW DatiDirettori
(Codice, NomeCognome, Sesso, AnnoNascita, Afferisce,
Dirige, AnnoNomina)
AS
SELECT i.Codice, i.NomeCognome, i.Sesso, i.AnnoNascita,
i.Afferisce, d.Dirige, d.AnnoDiNomina
FROM Impiegati i, Direttori d
```

```

WHERE i.Codice = d.Codice;

CREATE VIEW DatiResponsabili
 (Codice, NomeCognome, Sesso, AnnoNascita, Afferisce,
 AnnoNomina)
 AS
SELECT i.Codice, i.NomeCognome, i.Sesso, i.AnnoNascita,
 i.Afferisce, r.AnnoDiNomina
FROM Impiegati i, Responsabili r
WHERE i.Codice = r.Codice;

CREATE VIEW DatiDipendenti
 (Codice, NomeCognome, Sesso, AnnoNascita, Afferisce,
 CoordinatoDa)
 AS
SELECT i.Codice, i.NomeCognome, i.Sesso, i.AnnoNascita,
 i.Afferisce, d.CoordinatoDa
FROM Impiegati i, Dipendenti d
WHERE i.Codice = d.Codice;

(a) SELECT Nome, AnnoNascita
 FROM Familiari
 WHERE CapoFamiglia = 350 and RelazioneParentela = 'figlio';
(b) SELECT i.NomeCognome, i.Codice, d.Nome
 FROM Impiegati i, Dipartimenti d
 WHERE i.Afferisce = d.Numero;
(c) SELECT i.NomeCognome, f.Nome, f.AnnoNascita
 FROM Impiegati i, Familiari f
 WHERE f.CapoFamiglia = i.Codice
 AND f.RelazioneParentela = 'figlio' AND f.Sesso = 'm';
(d) SELECT p.NumeroP, p.NomeP, d.Nome, dd.NomeCognome
 FROM Progetti p, Dipartimenti d, DatiDirettori dd
 WHERE p.GestitoDa = d.Numero AND dd.Dirige = d.Numero
 AND d.Citta = 'Pisa';
(e) SELECT DISTINCT d.Nome
 FROM Dipartimenti d, Impiegati i, Familiari f
 WHERE i.Afferisce = d.Numero AND f.CapoFamiglia = i.Codice;
(f) SELECT COUNT(*)
 FROM Dipartimenti d, Impiegati i
 WHERE d.Nome = 'Informatica' AND d.Numero = i.Afferisce;
(g) SELECT p.NomeP, p.NumeroP, COUNT(*) AS NumerolImpiegati
 FROM Progetti p, Impiegati i, Partecipazioni pa
 WHERE p.NumeroP = pa.Progetto AND pa.Impiegato = i.Codice
 GROUP BY p.NumeroP, p.NomeP
 HAVING COUNT(*) > 2;
(h) SELECT d.Nome, COUNT(*) AS NumerolImpiegati,
 AVG(i.AnnoNascita) AS MediaAnnoNascImpiegati
 FROM Dipartimenti d, Impiegati i
 WHERE d.Numero=i.Afferisce
 GROUP BY d.Numero, d.Nome;
(i) SELECT dr.NomeCognome, ddi.NomeCognome
 FROM DatiResponsabili dr, DatiDipendenti ddi
 WHERE ddi.CoordinatoDa = dr.Codice;
(j) SELECT i.NomeCognome, d.Nome
 FROM Impiegati i, Dipartimenti d
 WHERE i.Afferisce = d.Numero;
(k) Da fare.

```

```
(l) SELECT p.NomeP, i.Codice, i.NomeCognome
FROM Progetti p, Partecipazioni pa, Impiegati i, DatiDirettori dd,
 Dipartimenti d
WHERE p.NumeroP = pa.Progetto AND pa.Impiegato = i.Codice
 AND p.GestitoDa = d.Numero AND dd.Dirige = d.Numero
 AND (i.NomeCognome = 'Pablo Rossi' OR
 dd.NomeCognome = 'Pablo Rossi');
```

*oppure:*

```
SELECT p.NomeP, i.Codice AS c, i.NomeCognome AS n
FROM Progetti p, Partecipazioni pa, Impiegati i
WHERE p.NumeroP = pa.Progetto AND pa.Impiegato = i.Codice
 AND i.NomeCognome = 'Pablo Rossi'
```

```
UNION
SELECT p.NomeP, dd.Codice AS c, dd.NomeCognome AS n
FROM Progetti p, DatiDirettori dd, Dipartimenti d
WHERE p.GestitoDa = d.Numero AND dd.Dirige = d.Numero
 AND dd.NomeCognome = 'Pablo Rossi';
```

(m) Da fare.

(n) Da fare.

(o) La soluzione con il quantificatore universale è:

```
SELECT i.NomeCognome
FROM Impiegati i
WHERE FOR ALL x IN Partecipazioni WHERE x.Impiegato = 300:
 (FOR SOME y IN Partecipazioni WHERE y.Impiegato = i.Codice:
 x.Progetto = y.Progetto);
```

con la doppia negazione si ottiene:

```
SELECT i.NomeCognome
FROM Impiegati i
WHERE NOT FOR SOME x IN Partecipazioni WHERE x.Impiegato = 300:
 (NOT FOR SOME y IN Partecipazioni WHERE y.Impiegato = i.Codice:
 x.Progetto = y.Progetto);
```

che in SQL diventa:

```
SELECT i.NomeCognome
FROM Impiegati i
WHERE NOT EXISTS
 (SELECT *
 FROM Partecipazioni x
 WHERE x.Impiegato =300
 AND NOT EXISTS
 (SELECT *
 FROM Partecipazioni y
 WHERE y.Impiegato = i.Codice
 AND x.Progetto = y.Progetto));
```

```
(p) CREATE VIEW DipConAlmenoDuelmp AS
SELECT Numero, Nome
FROM Impiegati, Dipartimenti
WHERE Afferisce = Numero
GROUP BY Numero, Nome
HAVING COUNT(*) > 2;

SELECT Nome, COUNT(*) AS NImp
FROM Impiegati, DipConAlmenoDuelmp
WHERE Afferisce = Numero AND AnnoNascita > 1950
```

GROUP BY Numero, Nome;

Il risultato non contiene i dipartimenti con più di due impiegati, tutti nati prima del 1950, che si trovano con l'interrogazione:

```
SELECT *
FROM DipConAlmenoDuelmp
WHERE FOR ALL x IN Impiegati WHERE x.Afferisce = Numero:
 x.AnnoNascita <= 1950;
```

con la doppia negazione si ottiene:

```
SELECT *
FROM DipConAlmenoDuelmp
WHERE NOT FOR SOME x IN Impiegati WHERE x.Afferisce = Numero:
 x.AnnoNascita > 1950;
```

che in SQL diventa:

```
SELECT Nome, 0 AS NImp
FROM DipConAlmenoDuelmp
WHERE NOT EXISTS (SELECT *
 FROM Impiegati x
 WHERE x.Afferisce = Numero
 AND x.AnnoNascita > 1950);
```

Quindi la soluzione completa è la seguente:

```
SELECT Nome, COUNT(*) AS NImp
FROM Impiegati, DipConAlmenoDuelmp
WHERE Afferisce = Numero AND AnnoNascita > 1950
GROUP BY Numero, Nome
UNION
SELECT Nome, 0 AS NImp
FROM DipConAlmenoDuelmp
WHERE NOT EXISTS (SELECT *
 FROM Impiegati x
 WHERE x.Afferisce = Numero
 AND x.AnnoNascita > 1950);
```