Corso di Basi di Dati
Personale Universitario
Esercitazioni

Esercizi SQL 1

Maurizio Vincini (vincini@dsi.unimo.it)

SCHEMA DI RIFERIMENTO

S(Matr, Snome, Citta, Acorso)

C(CC, Cnome, CD, anno)
FK: CD **REFERENCES** D

D(CD, Cnome, Citta)

E(Matr, CC, Data, Voto)
FK: Matr **REFERENCES** S
FK: CC **REFERENCES** C

Esempio di riferimento per le interrogazioni

S

Matr	SNome	Città	ACorso
M1	Lucia Quaranta	SA	1
M2	Giacomo Tedesco	PA	2
M3	Carla Longo	MI	1
M4	Ugo Rossi	MO	1
M5	Valeria Neri	MO	2
M6	Giuseppe Verdi	BO	1
M7	Maria Rossi	null	1

C

CC	CNome	CD
C1	Fisica 1	D1
C2	Analisi Matematica 1	D2
C3	Fisica 2	D1
C4	Analisi Matematica 2	D3

D

CD	CNome	Città
D1	Paolo Rossi	MO
D2	Maria Pastore	BO
D3	Paola Caboni	FI

E

Matr	CC	Data	Voto
M1	C1	06-29-1995	24
M1	C2	08-09-1996	33
M1	C3	03-12-1996	30
M2	C1	06-29-1995	28
M2	C2	07-07-1996	24
M3	C2	07-07-1996	27
M3	C3	11-11-1996	25
M4	C3	11-11-1996	33
M6	C2	01-02-1996	28
M7	C1	06-29-1995	24
M7	C2	04-11-1996	26
M7	C3	06-23-1996	27

Predicati semplici

- **Operatori relazionali**: `<attr> <op-rel> <cost>`
  dove `<op-rel>` $\in \{=, <>, >, >=, <, <=\}$

  Es "Studenti del secondo anno di corso"

```
SELECT  *
FROM    S
WHERE   ACorso=2
```

  Es "Esami con voto compreso tra 24 e 28"

```
SELECT  *
FROM    E
WHERE   Voto >= 24
AND     Voto <= 8
```

- **Operatore di range**:
  `<attr>` **BETWEEN** `<cost1>` **AND** `<cost2>`

  Es "Esami con voto compreso tra 24 e 28"

```
SELECT  *
FROM    E
WHERE   Voto BETWEEN 4 AND 28
```

- **Operatore di set**:
  `<attr>` **IN** `(<cost1>, ... , <costN>)`

  Es "Esami con voto pari a 29, 30 oppure con lode (voto pari a 33)"

```
SELECT  *
FROM    E
WHERE   Voto IN (29,30,33)
```

- **Operatore di confronto stringhe**:
  `<attr>` **LIKE** `<stringa>`
  dove `<stringa>` può contenere i caratteri speciali _ (carattere arbitrario) e % (stringa arbitraria)

  Es "Studenti il cui nome inizia con A e termina con O"

```
SELECT  *
FROM    S
WHERE   SNome LIKE 'A%O'
```

- **Operatori quantificati**:
  `<attr> <op-rel>[`**ANY**|**ALL**`] (<cost1>,...,<costN>)`

  Es "Esami con voto pari a 29, 30 oppure con lode (voto pari a 33)"

```
SELECT  *
FROM    E
WHERE   Voto =ANY (29,30,33)
```

  Es "Esami con voto diverso da 29, 30 e 33"

```
SELECT  *
FROM    E
WHERE   Voto <>ALL (29,30,33)
```

  ♢ Confronto con l'insieme vuoto `()`:
  - `<attr> <op-rel>ANY ()` ha valore `false`
  - `<attr> <op-rel>ALL ()` ha valore `true`

- **Operatore di confronto con valori NULL**:
  `<attr> IS [NOT] NULL`

  Es "Studenti con l'attributo città non specificato"

```
SELECT  *
FROM    S
WHERE   Città IS NULL
```

♢ Ordinamento del risultato:

Es "Studenti di Modena ordinati in senso ascendente rispetto all'anno di corso"

```
SELECT    Matr,ACorso
FROM      S
WHERE     Città='MO'
ORDER BY  ACorso
```

- L'ordinamento deve essere fatto rispetto a uno o più elementi della `<lista-select>`: un tale elemento può essere indicato anche riportando la sua posizione nella `<lista-select>`.

Es "Esami del corso C1 ordinati in senso discendente rispetto al voto espresso in sessantesimi, e a parità di voto rispetto alla matricola"

```
SELECT    Matr,CC,(60*Voto)/30
FROM      E
WHERE     CC='C1'
ORDER BY  3 DESC, Matr
```

Prodotto Cartesiano e Join

♢ Il **prodotto cartesiano** di due o più relazioni si ottiene riportando le relazioni nella `<lista-from>` della clausola `FROM`, senza clausola `WHERE`.

♢ Il **join** viene espresso generalmente riportando nella clausola `FROM` le relazioni interessate e nella clausola `WHERE` le *condizioni di join*.

Es "Combinazioni di studenti e di docenti residenti nella stessa città"

```
SELECT  S.Matr,S.Città,D.CD
FROM    S,D
WHERE   S.Città=D.Città
```

S.Matr	S.Città	D.CD
M6	BO	D2
M3	MO	D1
M4	MO	DI
M5	MO	DI

♢ Con l'SQL92 è possibile esprimere le operazioni di join nella clausola FROM (INNER è il default):

<tabella1> [**INNER**|**LEFT**|**RIGHT**|**FULL**] **JOIN**

<tabella2> **ON** <condizione>

Es "Combinazioni di studenti e di docenti residenti nella stessa città"

```
SELECT  S.Matr,S.Città,D.CD
FROM    S join D on (S.Città=D.Città)
```

♢ Join con predicati locali

Es "Studenti residenti nella stessa città di residenza del docente D1"

```
SELECT  S.*
FROM    S,D
WHERE   S.Città = D.Città
AND     D.CD='D1'
```

oppure

```
SELECT  S.*
FROM    S JOIN D on (S.Città = D.Città)
WHERE   D.CD='D1'
```

S.Matr	S.Città	D.CD
M3	MO	D1
M4	MO	D1
M5	MO	D1

Outer-Join

♢ Oltre al join interno, con lo standard SQL92 sono stati introdotti gli operatori di outer join :

`<tabella1>` **LEFT JOIN** `<tabella2>` **ON** `<condizione>`
:
mantiene le tuple di `<tabella1>` per cui non esiste corrispondenza in `<tabella2>`

`<tabella1>` **RIGHT JOIN** `<tabella2>` **ON** `<condizione>`
:
mantiene le tuple di `<tabella2>` per cui non esiste corrispondenza in `<tabella1>`

`<tabella1>` **FULL JOIN** `<tabella2>` **ON** `<condizione>`
:
mantiene le tuple di `<tabella1>` e di `<tabella2>` per cui non esiste corrispondenza in `<tabella2>` e `<tabella1>` rispettivamente

♢ Le tuple senza corrispondenza vengono concatenate con tuple di valori `null`, di lunghezza opportuna.

Es "Tutti gli studenti con i relativi esami sostenuti inclusi gli studenti che non hanno sostenuto alcun esame"

```
SELECT  S.Matr,E.CC
FROM    S LEFT JOIN E on (S.Matr=E.Matr)
```

Es "Combinazioni di studenti e di docenti residenti nella stessa città inclusi gli studenti (docenti) che risiedono in una città che non ha corrispondenza nella relazione dei docenti (studenti)"

```
SELECT  S.Matr,S.Città,D.CD, D.Città
FROM    S FULL JOIN D on (S.Città=D.Città)
```

S.Matr	S.Città	D.CD	D.Città
M6	BO	D2	BO
M3	MO	D1	MO
M4	MO	D1	MO
M5	MO	D1	MO
M1	SA	null	null
M2	PA	null	null
M7	null	null	null
null	null	D3	FI

♢ Nella clausola FROM è possibile esprimere più di un'operazione di join.

Es "Per ogni esame con voto superiore a 24 riportare il nome dello studente e il codice del docente del corso"

```
SELECT   S.SNome,C.CD
FROM      (S JOIN E on (S.Matr=E.Matr))
           JOIN C on (E.CC=C.CC)
WHERE   Voto > 24
```

Es "Matricole degli studenti con i codici dei corsi dei relativi esami sostenuti, inclusi gli studenti che non hanno sostenuto alcun esame e i corsi per i quali non ci sono esami sostenuti"

```
SELECT  DISTINCT S.Matr,C.CC
FROM      (S LEFT JOIN E on (S.Matr=E.Matr))
           FULL JOIN C on (E.CC=C.CC)
```

S.Matr	C.CC
M1	C1
M1	C2
M1	C3
M2	C1
M2	C2
M3	C2
M3	C3
M4	C3
M5	null
M6	C2
M7	C1
M7	C2
M7	C3
null	C4

Self Join

♢ Nel join tra una tabella e se stessa occorre necessariamente utilizzare dei sinonimi (*alias*) per distinguere le diverse occorrenza della tabella.

Es "Coppie di studenti residenti nella stessa città"

```
SELECT  S1.Matr,S2.Matr
FROM    S S1, S S2
WHERE   S1.Città = S2.Città
AND     S1.Matr<S2.Matr
```

oppure

```
SELECT  S.*
FROM      S S1 Join S S2
          On (S1.Città = S2.Città)
WHERE   S1.Matr<S2.Matr
```

S1.Matr	S2.Matr
M3	M5
M3	M4
M4	M5

Es "Matricole degli studenti che hanno sostenuto almeno uno degli esami sostenuti dallo studente di nome 'Giuseppe Verdi"'

```
SELECT  E1.Matr
FROM    S, E E1, E E2
WHERE   E2.Matr =S.Matr
AND     E1.CC = E2.CC
AND     S.SNome='Giuseppe Verdi'
```

Interrogazioni innestate

♢ Una interrogazione viene detta *innestata* o *nidificata* se la sua condizione è formulata usando il risultato di un'altra interrogazione, chiamata *subquery*

Operatori quantificati:
`<attr> <op-rel>[`**ANY**`|`**ALL**`] <subquery>`

Operatore IN:
`<attr> [`**NOT**`] IN <subquery>`

Quantificatore esistenziale `EXISTS`:
`[`**NOT**`] EXISTS <subquery>`

♢ **Interrogazione innestata con operatori quantificati**

Es "Nome degli studenti che hanno sostenuto l'esame del corso C1"

```
SELECT  SNome
FROM    S
WHERE   Matr =ANY  (   SELECT   Matr
                       FROM     E
                       WHERE    CC='C1')
```

Es "Studenti con anno di corso più basso"

```
SELECT  *
FROM    S
WHERE   ACorso <= ALL  (   SELECT   ACorso
                           FROM     S)
```

♢ Interrogazioni innestate con l'operatore IN

Es "Nome degli studenti che hanno sostenuto l'esame del corso C1"

```
SELECT  SNome
FROM    E,S
WHERE   E.Matr=S.Matr
AND     E.CC = 'C1'
```

oppure

```
SELECT  SNome
FROM    S
WHERE   Matr IN  (   SELECT   Matr
                     FROM     E
                     WHERE    CC='C1')
```

La subquery restituisce l'insieme `(M1,M2,M7)` e pertanto la condizione dell'interrogazione innestata equivale a `Matr IN (M1,M2,M7)`.

Es "Nome degli studenti che hanno sostenuto l'esame di un corso del docente D1"

```
SELECT  SNome
FROM    S
WHERE   Matr IN
          (  SELECT  Matr
             FROM    E
             WHERE   CC IN
                       (  SELECT  CC
                          FROM    C
                          WHERE   CD='D1'))
```

Subquery correlate

Una subquery viene detta correlata se la sua condizione è formulata usando relazioni e/o sinonimi definite nella query esterna.

Es "Nome degli studenti che hanno sostenuto l'esame del corso C1"

```
SELECT  SNome
FROM    S
WHERE   'C1' IN  (   SELECT   CC
                     FROM     E
                     WHERE    E.Matr=S.Matr)
```

Per ogni tupla della relazione S della query esterna, (detta *tupla corrente*) si valuta la subquery che ha come risultato il codice degli esami sostenuti dallo studente corrente: se C1 è tra questi esami, il nome dello studente corrente viene riportato in uscita.

♢ Per una maggiore leggibilità è conveniente far uso di sinonimi

```
SELECT  S1.SNome
FROM    S S1
WHERE   'C1' IN  (   SELECT   E1.CC
                     FROM     E E1
                     WHERE    E1.Matr=S1.Matr)
```

♢ I sinonimi sono indispensabili quando una stessa relazione compare sia nella query esterna che nella subquery (analogalmente ai self join)

Es "Per ogni città, il nome degli studenti con anno di corso più alto"

```
SELECT  S1.Città,S1.SNome
FROM    S S1
WHERE   S1.ACorso >= ALL
          (  SELECT  S2.ACorso
             FROM    S S2
             WHERE   S1.Città=S2.Città)
```

♢ Per gli attributi non qualificati avviene la qualificazione automatica con il nome della relazione *più vicina*.

Ad esempio, la precedente interrogazione si può scrivere come:

```
SELECT  Città,SNome
FROM    S S1
WHERE   ACorso >= ALL
          (  SELECT  ACorso
             FROM    S
             WHERE   S1.Città=Città)
```