

Il Linguaggio di Interrogazione SQL

Sommario

- Introduzione
- Sintassi
- Interrogazioni su singole relazioni
- Interrogazioni su relazioni multiple
- Interrogazioni Innestate
- Operatori
- Conclusioni

SQL (Structured Query Language)

- Non solo un linguaggio di interrogazione:
 - DDL (Data Definition Language)
 - DML (Data Manipulation Language)
- Diffusione di SQL
 - Presente praticamente in tutti i DBMS relazionali
 - Ha fortemente influenzato anche i DBMS Orientati agli Oggetti (OQL)
- Standardizzazione di SQL

Interrogazioni SQL

Forma Base

SELECT attributi desiderati

FROM variabili tupla – variano sulle relazioni

WHERE condizione sulle variabili tupla

Schema di Esempio

Birre(nome,produttore)

Bar(nome,indirizzo,permesso)

Bevitori(nome,indirizzo,telefono)

Piace(bevitore,birra)

Vende(bar,birra,prezzo)

Frequenta(bevitore,bar)

Esempio

Quali birre sono prodotte dalla Peroni ?

Birre(nome, produttore)

```
SELECT nome
FROM Birre
WHERE produttore = 'Peroni';
```

nome

Nastro Azzurro

Sans Souci

Gran Riserva

Interrogazioni Singole Relazioni

SEMANTICA FORMALE

1. Parti dalla relazione della clausola FROM
2. Applica σ , utilizzando le condizioni della clausola WHERE
3. Applica π , utilizzando gli attributi della clausola SELECT

EQUIVALENTE SEMANTICA OPERAZIONALE

Consideriamo una *variabile-tupla* che varia su tutte le tuple della relazione. Per ogni tupla:

- Controlla se soddisfa la clausola WHERE
- Se la soddisfa, allora stampa gli attributi della clausola SELECT

'*' come lista di tutti gli attributi

Birre(nome, produttore)

```
SELECT *  
FROM Birre  
WHERE produttore = 'Peroni';
```

<u>nome</u>	<u>produttore</u>
Nastro Azzurro	Peroni
Sans Souci	Peroni
Gran Riserva	Peroni

Ridenominazione di attributi

Birre(nome, produttore)

```
SELECT nome AS birra
FROM Birre
WHERE produttore = 'Peroni';
```

birra

Nastro Azzurro

Sans Souci

Gran Riserva

Espressioni come valori nelle colonne

Vende(bar,birra,prezzo)

```
SELECT bar, birra,   prezzo/1936.27 AS prezzoInEuro
FROM Vende;
```

bar	birra	prezzoInEuro
Impero	Nastro Azzurro	1.5
Bruzio	Gran Riserva	1.8
...

- Nota: la clausola **WHERE** non é obbligatoria

Esempio

Trova il prezzo della Nastro Azzurro al bar Impero.

Vende(bar,birra,prezzo)

```
SELECT prezzo
```

```
FROM Vende;
```

```
WHERE bar = 'Impero' AND birra = 'Nastro Azzurro';
```

- Le condizioni nella clausola `WHERE` possono utilizzare operatori logici `AND`, `OR`, `NOT` e parentesi nel modo usuale.
- SQL é “case insensitive”.
Maiuscole/Minuscole sono rilevanti solo nelle stringhe (tra apici).

Caratteri Jolly

- ‘%’ rappresenta qualunque stringa
- ‘_’ rappresenta qualunque carattere (singolo)
- La condizione “attributo LIKE pattern” risulta vera se il valore dell’attributo corrisponde al pattern (secondo le regole di sopra)

Esempio

Nomi dei bevitori il cui telefono ha prefisso di quattro cifre ed inizia con “098”.

Bevitori(nome, indirizzo, telefono)

```
SELECT nome
```

```
FROM Bevitori
```

```
WHERE telefono LIKE '098_ - %';
```

- Nota: i ‘pattern’ devono essere posti tra apici, come le stringhe.

Interrogazioni su Relazioni Multiple

- Lista di relazioni nella clausola FROM
- Ambiguitá dovute ad omonimie tra attributi sono eliminabili con la notazione Relazione-Punto-Attributo

Esempio: Birre preferite dai frequentatori del bar Impero.

Piace(bevitore,birra) Frequenta(bevitore,bar)

```
SELECT  birra
FROM    Frequenta, Piace
WHERE   Frequenta.bevitore = 'Piace.Bevitore'
        AND bar = 'Impero';
```

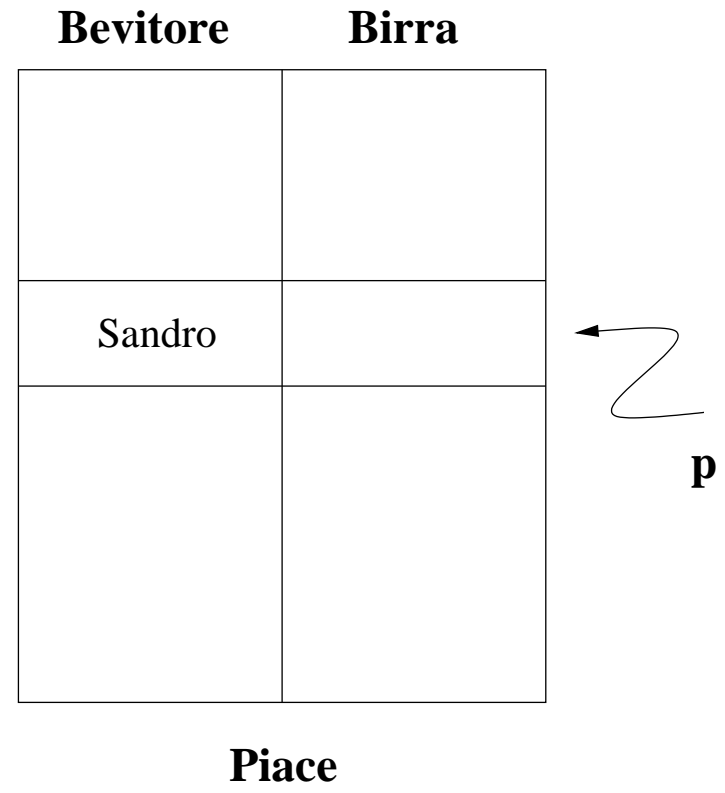
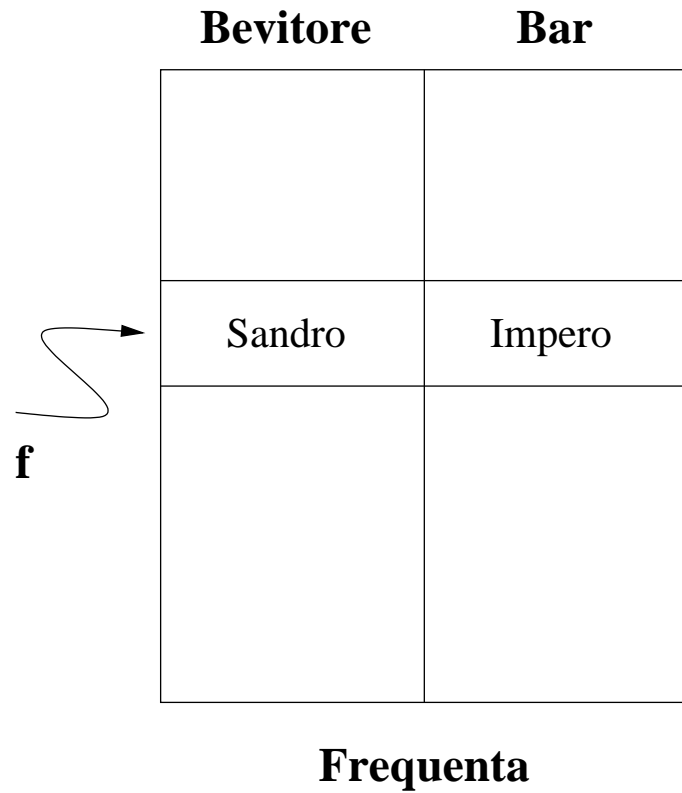
SEMANTICA FORMALE

Come per le interrogazioni su relazioni singole, ma partendo dal prodotto cartesiano di tutte le relazioni elencate nella clausola **FROM**

SEMANTICA OPERAZIONALE

Consideriamo una variabile-tupla per ogni relazione della clausola **FROM**.

- Immaginiamo che ognuna di queste variabili-tupla punti ad una tupla della sua relazione, in tutte le possibili combinazioni
- Se la condizione del **WHERE** é resa vera dal corrente assegnamento delle variabili-tupla alle tuple, allora stampiamo gli attributi della clausola **SELECT**



Variabili-tupla Esplicite

Talvolta dobbiamo far riferimento a piú copie della stessa relazione.

- Utilizziamo le *variabili-tupla* come sinonimi per le relazioni.

Esempio: (Coppie di) Birre prodotte dallo stesso produttore.

Birre(nome, produttore)

```
SELECT b1.nome, b2.nome
```

```
FROM Birre b1, Birre b2
```

```
WHERE b1.produttore = b2.produttore AND b1.nome < b2.nome
```

Nota: `b1.nome < b2.nome` evita la generazione di tuple (X,X)

nonche' la generazione della stessa coppia in ordine inverso.

Interrogazioni Innestate

Il risultato di un'interrogazione può essere utilizzato nella clausola **WHERE** di un'altra interrogazione.

Esempio: Bar che servono la Gran Riserva allo stesso prezzo a cui l'Impero serve la birra Devil.

Vende(bar,birra,prezzo)

```
SELECT bar
```

```
FROM Vende
```

```
WHERE birra = 'Gran Riserva' AND
```

```
    prezzo = ( SELECT prezzo
```

```
                FROM Vende
```

```
                WHERE bar='Impero' AND birra= 'Devil');
```

Note:

- *Regola di Visibilitá:* un attributo fá riferimento alla relazione innestata piú vicina ad esso.
- Le parentesi devono obbligatoriamente essere utilizzate per racchiudere l'interrogazione innestata.

Operatore IN

La condizione “Tupla IN Relazione” é vera se e solo se la tupla appartiene alla relazione

Esempio: Nomi e produttori delle birre che piacciono a Sandro.

Birre(nome,produttore) Piace(bevitore,birra)

```
SELECT *
FROM Birre
WHERE nome IN ( SELECT birra
                 FROM Piace
                 WHERE bevitore = 'Sandro');
```

Importanza di NOT IN

Permette di calcolare la differenza di due relazioni.

La condizione “Tupla NOT IN Relazione” é vera se e solo se la tupla non appartiene alla relazione.

Esempio: Bar frequentati da Sonia non frequentati da Eugenio.

```
SELECT bar
```

```
FROM Frequenta
```

```
WHERE bevitore = 'Sonia'
```

```
    AND bar NOT IN (    SELECT bar  
                        FROM Frequenta  
                        WHERE bevitore = 'Eugenio');
```

Esempio: Nomi e produttori delle birre che piacciono a Nicola ma non piacciono a Francesco.

Birre(nome,produttore) Piace(bevitore,birra)

```
SELECT *
FROM Birre
WHERE nome IN      ( SELECT birra
                     FROM Piace
                     WHERE bevitore = 'Nicola')
AND nome NOT IN ( SELECT birra
                  FROM Piace
                  WHERE bevitore = 'Francesco');
```

EXISTS

“EXISTS Relazione” é vera se e solo se la relazione é non vuota.

Esempio: Birre uniche ad essere fornite dal loro fornitore.

Birre(nome, produttore)

```
SELECT nome
```

```
FROM Birre b1
```

```
WHERE NOT EXISTS ( SELECT *  
                    FROM birre  
                    WHERE produttore = b1.prodotto  
                    AND nome <> b1.nome );
```

Quantificatori

Esistenziale: ANY. Universale: ALL.

Attenzione: Nella lingua inglese assumono significati simili.

In SQL la semantica é ben diversa:

Esempio: Birra (o birre) venduta al prezzo piú alto.

Vende(bar,birra,prezzo)

```
SELECT birre
```

```
FROM Vende
```

```
WHERE prezzo >= ALL ( SELECT prezzo  
                        FROM Vende);
```

Esercizio: Trovare le birre che non sono vendute al prezzo piú basso.

UNION, INTERSECT, EXCEPT

“Relazione UNION Relazione” genera l’unione delle due relazioni.

INTERSECT e EXCEPT si comportano analogamente generando, rispettivamente, intersezione e differenza.

Esempio

Determinare i bevitori e le birre tali che al bevitore piace la birra ed egli frequenta un bar che la serve.

Piace(bevitore,birra) Vende(bar,birra,prezzo)

Frequenta(bevitore,bar)

Piace

INTERSECT

(SELECT bevitore, birra

FROM Vende, Frequenta

WHERE Frequenta.bar = Vende.bar);

Conclusioni

- Cosa abbiamo appreso.
- Cenni sull'espressività.
- Cosa vedremo nella prossima lezione.