



## Il linguaggio SQL: raggruppamenti

---

Sistemi Informativi L-A

Home Page del corso:

<http://www-db.deis.unibo.it/courses/SIL-A/>

Versione elettronica: [SQLb-gruppi.pdf](#)

Sistemi Informativi L-A



## Informazioni di sintesi

---

- Quanto sinora visto permette di estrarre dal DB informazioni che si riferiscono a **singole tuple** (eventualmente ottenute mediante operazioni di join)  
Esempio: **il ruolo dell'impiegato 'E001', il responsabile della sede 'S02', ecc.**
- In molti casi è viceversa utile ottenere dal DB informazioni (di sintesi) che caratterizzano **"gruppi" di tuple**  
Esempio: **il numero di programmatori della sede 'S01', la media degli stipendi a Bologna, ecc.**
- A tale scopo SQL mette a disposizione due strumenti di base:
  - **Funzioni aggregate**
  - **Clausola GROUP BY**



## DB di riferimento per gli esempi

### Imp

CodImp	Nome	Sede	Ruolo	Stipendio
E001	Rossi	S01	Analista	2000
E002	Verdi	S02	Sistemista	1500
E003	Bianchi	S01	Programmatore	1000
E004	Gialli	S03	Programmatore	1000
E005	Neri	S02	Analista	2500
E006	Grigi	S01	Sistemista	1100
E007	Violetti	S01	Programmatore	1000
E008	Aranci	S02	Programmatore	1200

### Sedi

Sede	Responsabile	Citta
S01	Biondi	Milano
S02	Mori	Bologna
S03	Fulvi	Milano

### Prog

CodProg	Citta
P01	Milano
P01	Bologna
P02	Bologna

SQL - Raggruppamenti

Sistemi Informativi L-A

3



## Funzioni aggregate (1)

- Lo standard SQL mette a disposizione una serie di **funzioni aggregate** (o "di colonna"):
  - MIN** minimo
  - MAX** massimo
  - SUM** somma
  - AVG** media aritmetica
  - STDEV** deviazione standard
  - VARIANCE** varianza
  - COUNT** contatore

```
SELECT SUM(Stipendio) AS TotStipS01
FROM Imp
WHERE Sede = 'S01'
```

TotStipS01
5100

SQL - Raggruppamenti

Sistemi Informativi L-A

4



## Funzioni aggregate (2)

- L'argomento di una funzione aggregata è una qualunque espressione che può figurare nella SELECT list (ma **non un'altra funzione aggregata!**)

```
SELECT    SUM(Stipendio*12) AS TotStipAnnuiS01
FROM      Imp
WHERE     Sede = 'S01'
```

TotStipAnnuiS01
61200

- Tutte le funzioni, ad eccezione di **COUNT**, ignorano i valori nulli
- Il risultato è **NULL** se tutti i valori sono **NULL**
- L'opzione **DISTINCT** considera solo i valori distinti

```
SELECT    SUM(DISTINCT Stipendio)
FROM      Imp
WHERE     Sede = 'S01'
```

4100

SQL - Raggruppamenti

Sistemi Informativi L-A

5



## COUNT e valori nulli

- La forma **COUNT(\*)** conta le tuple del risultato; viceversa, specificando una colonna, si omettono quelle con valore nullo in tale colonna

**Imp**

CodImp	Sede	...	Stipendio
E001	S01		2000
E002	S02		1500
E003	S01		1000
E004	S03		NULL
E005	S02		2500
E006	S01		NULL
E007	S01		1000
E008	S02		1200

```
SELECT    COUNT(*) AS NumImpS01
FROM      Imp
WHERE     Sede = 'S01'
```

NumImpS01
4

```
SELECT    COUNT(Stipendio)
          AS NumStipS01
FROM      Imp
WHERE     Sede = 'S01'
```

NumStipS01
3

SQL - Raggruppamenti

Sistemi Informativi L-A

6



## Funzioni aggregate e tipo del risultato

- Per alcune funzioni aggregate, al fine di ottenere il risultato desiderato, è necessario operare un *casting* dell'argomento

Imp

...	Stipendio
	2000
	1500
	1000
	1000
	2500
	1100
	1000
	1200

```
SELECT  AVG(Stipendio) AS AvgStip
FROM    Imp      -- valore esatto 1412.5
```

AvgStip
1412

```
SELECT  AVG(CAST(Stipendio AS Decimal(6,2)))
        AS AvgStip
FROM    Imp
```

AvgStip
1412.50

SQL - Raggruppamenti

Sistemi Informativi L-A

7



## Clausola SELECT e funzioni aggregate

- Se si usano funzioni aggregate, la SELECT list non può includere altri elementi che non siano a loro volta funzioni aggregate

```
SELECT  Nome, MIN(Stipendio)
FROM    Imp
```

non va bene!

(viceversa, `SELECT MIN(Stipendio), MAX(Stipendio) ..` è corretto)

- Il motivo è che una funzione aggregata restituisce un singolo valore, mentre il riferimento a una colonna è in generale un insieme di valori (eventualmente ripetuti)
- Nel caso specifico (chi sono gli impiegati con stipendio minimo?) è necessario ricorrere a un'altra soluzione, che vedremo più avanti

SQL - Raggruppamenti

Sistemi Informativi L-A

8

## Funzioni aggregate e raggruppamento

- I valori di sintesi calcolati dalle funzioni aggregate si riferiscono a **tutte** le tuple che soddisfano le condizioni della clausola WHERE
- In molti casi è viceversa opportuno fornire tali valori per **gruppi omogenei di tuple** (es: impiegati di una stessa sede)
- La clausola **GROUP BY** serve a definire tali gruppi, specificando una o più **colonne (di raggruppamento)** sulla base della/e quale/i le tuple sono raggruppate per valori uguali

```
SELECT Sede, COUNT(*) AS NumProg
FROM Imp
WHERE Ruolo = 'Programmatore'
GROUP BY Sede
```

Sede	NumProg
S01	2
S03	1
S02	1

- La SELECT list può includere le colonne di raggruppamento, ma non altre!

SQL - Raggruppamenti

Sistemi Informativi L-A

9

## Come si ragiona con il GROUP BY

- Le tuple che soddisfano la clausola WHERE...

CodImp	Nome	Sede	Ruolo	Stipendio
E003	Bianchi	S01	Programmatore	1000
E004	Gialli	S03	Programmatore	1000
E007	Violetti	S01	Programmatore	1000
E008	Aranci	S02	Programmatore	1200

- ...sono raggruppate per valori uguali della/e colonna/e presenti nella clausola GROUP BY...

CodImp	Nome	Sede	Ruolo	Stipendio
E003	Bianchi	S01	Programmatore	1000
E007	Violetti	S01	Programmatore	1000
E004	Gialli	S03	Programmatore	1000
E008	Aranci	S02	Programmatore	1200

- ...e infine a ciascun gruppo si applica la funzione aggregata

Sede	NumProg
S01	2
S03	1
S02	1

SQL - Raggruppamenti

Sistemi Informativi L-A

10



## GROUP BY: esempi

- 1) Per ogni ruolo, lo stipendio medio nelle sedi di Milano

```
SELECT I.Ruolo, AVG(I.Stipendio) AS AvgStip
FROM   Imp I JOIN Sedi S ON (I.Sede = S.Sede)
WHERE  S.Citta = 'Milano'
GROUP BY I.Ruolo
```

Ruolo	AvgStip
Analista	2000
Sistemista	1100
Programmatore	1000

- 2) Per ogni sede di Milano, lo stipendio medio

```
SELECT I.Sede, AVG(I.Stipendio) AS AvgStip
FROM   Imp I JOIN Sedi S ON (I.Sede = S.Sede)
WHERE  S.Citta = 'Milano'
GROUP BY I.Sede
```

Sede	AvgStip
S01	1275
S03	1000

- 3) Per ogni ruolo e sede di Milano, lo stipendio medio

```
SELECT I.Sede, I.Ruolo, AVG(I.Stipendio)
FROM   Imp I JOIN Sedi S ON (I.Sede = S.Sede)
WHERE  S.Citta = 'Milano'
GROUP BY I.Sede, I.Ruolo
```

Ruolo	Sede	AvgStip
Analista	S01	2000
Sistemista	S01	1100
Programmatore	S01	1000
Programmatore	S03	1000

SQL - Raggruppamenti

Sistemi Informativi L-A

11



## Raggruppamento e proiezione

- Quando la **SELECT list include solo le colonne di raggruppamento**, il tutto è equivalente a ciò che si otterrebbe omettendo il **GROUP BY** e rimuovendo i duplicati con l'opzione **DISTINCT**

```
SELECT Sede
FROM   Imp
GROUP BY Sede
```

Sede
S01
S02
S03

equivale pertanto a

```
SELECT DISTINCT Sede
FROM   Imp
```

SQL - Raggruppamenti

Sistemi Informativi L-A

12



## Condizioni sui gruppi

- Oltre a poter formare dei gruppi, è anche possibile **selezionare dei gruppi sulla base di loro proprietà "complessive"**

```
SELECT  Sede, COUNT(*) AS NumImp
FROM    Imp
GROUP BY Sede
HAVING  COUNT(*) > 2
```

Sede	NumImp
S01	4
S02	3

- La clausola **HAVING** ha per i gruppi una funzione simile a quella che la clausola WHERE ha per le tuple (**attenzione a non confonderle!**)



## Tipi di condizioni sui gruppi

- Nella clausola HAVING si possono avere due tipi di predicati:
  - Predicati che fanno uso di **funzioni aggregate** (es. **COUNT(\*) > 2**)
  - Predicati che si riferiscono alle **colonne di raggruppamento**
    - Questi ultimi si possono anche inserire nella clausola WHERE

```
SELECT  Sede, COUNT(*) AS NumImp
FROM    Imp
GROUP BY Sede
HAVING  Sede <> 'S01'
```

equivale a

```
SELECT  Sede, COUNT(*) AS NumImp
FROM    Imp
WHERE    Sede <> 'S01'
GROUP BY Sede
```

Sede	NumImp
S02	3
S03	1

## Un esempio completo

*Per ogni sede di Bologna in cui il numero di impiegati è almeno 3, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per sede*

```
SELECT    I.Sede, AVG(Stipendio) AS AvgStipendio
FROM      Imp I, Sedi S
WHERE     I.Sede = S.Sede
          AND S.Citta = 'Bologna'
GROUP BY  I.Sede
HAVING    COUNT(*) >= 3
ORDER BY  AvgStipendio DESC, Sede
```



L'ordine delle clausole è **sempre** come nell'esempio  
Si ricordi che **il GROUP BY non implica alcun ordinamento del risultato**

## Riassumiamo:

- Le **funzioni aggregate** di SQL permettono di ottenere informazioni di sintesi sulle tuple che soddisfano la clausola WHERE
- Mediante la **clausola GROUP BY** è possibile suddividere tali tuple in gruppi, per ognuno dei quali si possono quindi calcolare informazioni di sintesi
- Se le informazioni non sono richieste per tutti i gruppi, si ricorre alla **clausola HAVING**, che permette di esprimere condizioni a livello di gruppo (anziché di singola tupla)