

Query Language e i Raggruppamenti

Operatori Aggregati e Clausole SQL

Informatica - IIS Fermi Sacconi Ceci

Ascoli Piceno

Tutti i diritti riservati



- Operatori Aggregati
- Clausola GROUP BY
- Clausola HAVING
- Limitazione dei risultati

Estensione di SQL

Rispetto all'algebra relazionale, la più importante estensione che introduce SQL è quella degli **operatori aggregati**.

Caratteristiche principali:

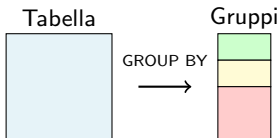
- Aggregano più righe della tabella
- Compiono operazioni a livello di *relazione* e non di tupla
- Restituiscono un **singolo valore aggregato**
- Non selezionano un sottoinsieme di righe, ma calcolano valori

Gli Operatori Aggregati

Funzionalità

Le operazioni di aggregazione consentono di:

- **Raggruppare** le tuple
- Effettuare **calcoli specifici** (somme, conteggi, statistiche)
- **Dividere** la tabella in sottoinsiemi
- Raggruppare tuple con **stessi valori** per attributi specifici



Gli Operatori Aggregati

Differenza fondamentale

Gli operatori di aggregazione, a differenza degli operatori matematici che agiscono soltanto sulla tupla in corso di elaborazione:

Possono essere applicati a più tuple della tabella e si caratterizzano per:

- Restituire un **valore unico**
- Operare su un **gruppo di valori**
- Lavorare sui valori che formano una **colonna**



Gli Operatori Aggregati

Principali operatori di aggregazione

AVG(campo) Calcola la **media aritmetica**

COUNT(expr|*) **Conta** le righe

MAX(expr) Calcola il valore **massimo**

MIN(expr) Calcola il valore **minimo**

SUM(campo) Calcola la **somma totale**

STDDEV(campo) Calcola la **deviazione standard**

SELECT AVG(prezzo) → 234.56

Gli Operatori Aggregati

Clausola GROUP BY

Specifica su quali campi effettuare i raggruppamenti.

Importante

Gli operatori aggregati possono comparire solo dopo le clausole:

- SELECT
- HAVING

```
SELECT categoria, COUNT(*) AS numero  
FROM prodotti  
GROUP BY categoria;
```

La Clausola HAVING

Funzionamento

La clausola HAVING permette di inserire un vincolo sui dati risultanti dall'operazione di raggruppamento GROUP BY.

Differenza con WHERE:

- WHERE opera sui **campi del database** (prima del raggruppamento)
- HAVING opera sui **campi risultanti dai raggruppamenti** (dopo)

```
SELECT reparto, AVG(stipendio) AS media
FROM impiegati
GROUP BY reparto
HAVING AVG(stipendio) > 30000;
```


Operatore COUNT

Sintassi

COUNT(espressione | *)

```
SELECT COUNT(*) AS totale_auto  
FROM auto;
```

totale_auto
12

Nota: COUNT(*) conta tutte le righe, inclusi i duplicati.

Operatore COUNT

Risultato

Il risultato è **12**: vengono contati tutti i record che compongono la tabella e che non sono NULL, a prescindere dal loro contenuto.

Importante

COUNT è l'unico operatore di aggregazione che considera i valori nulli ai fini del calcolo.

Per contare valori distinti:

```
SELECT COUNT(DISTINCT marca) AS marche_diverse  
FROM auto;
```

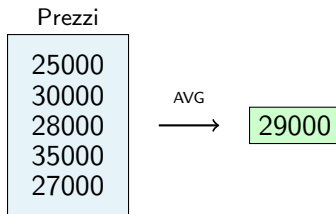
marche_diverse
8

Operatore AVG

Media Aritmetica

Calcola la media dei valori di una colonna numerica.

```
SELECT AVG(prezzo) AS prezzo_medio  
FROM auto;
```



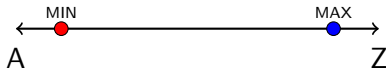
Operatori MIN e MAX

Funzionamento

MAX e MIN restituiscono i valori più alti e più bassi contenuti all'interno di una colonna.

Comportamento per tipo di dato:

- **Campi numerici:** restituisce il valore numerico
- **Campi testuali:** individua il campo che, in ordine alfabetico, è ultimo (MAX) o primo (MIN)
- Ordinamento secondo la tabella dei codici **ASCII**
- Ordinamento **lessicografico**



Operatori MIN e MAX

Espressioni matematiche

È possibile usare come argomento degli operatori MIN e MAX anche espressioni matematiche.

```
SELECT MAX(prezzo * 1.22) AS prezzo_max_iva  
FROM auto;
```

```
SELECT MIN(prezzo - sconto) AS prezzo_min_scontato  
FROM auto;
```

Prezzo \longrightarrow * 1.22 \longrightarrow MAX()

Operatore MIN e MAX

```
SELECT MIN(stipendio) AS stipendio_minimo  
FROM impiegati;
```

Nome	Stipendio
Mario Rossi	28000
Laura Bianchi	32000
Giuseppe Verdi	25000
Anna Neri	35000

stipendio_minimo
25000

Operatore MIN e MAX

```
SELECT MAX(stipendio) AS stipendio_massimo  
FROM impiegati;
```

Nome	Stipendio
Mario Rossi	28000
Laura Bianchi	32000
Giuseppe Verdi	25000
Anna Neri	35000

stipendio_massimo
35000

Operatore MIN e MAX

```
SELECT MAX(cognome) AS cognome_max  
FROM impiegati;
```

Cognome
Bianchi
Neri
Rossi
Verdi

cognome_max
Verdi

Ordinamento alfabetico: B | N | R | V

Operatore MIN e MAX

```
SELECT MIN(cognome) AS cognome_min  
FROM impiegati;
```

Cognome
Bianchi
Neri
Rossi
Verdi

cognome_min
Bianchi

Ordinamento alfabetico: B | N | R | V

Operatore MIN e MAX

Espressioni complesse

Vediamo come applicare operatori aritmetici nella query. Per conoscere la differenza tra stipendio minimo e massimo degli impiegati, rapportata in percentuale:

```
SELECT  
    (MAX(stipendio) - MIN(stipendio)) /  
    MAX(stipendio) * 100 AS diff_percentuale  
FROM impiegati;
```

diff_percentuale
28.57

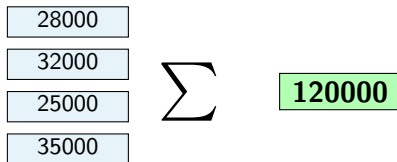
La differenza tra lo stipendio massimo e minimo è del 28.57%

Operatore SUM

Somma totale

Calcola la somma di tutti i valori di una colonna numerica.

```
SELECT SUM(stipendio) AS totale_stipendi  
FROM impiegati;
```



Operatore SUM

```
SELECT
    SUM(importo) AS totale_vendite,
    SUM(quantita) AS totale_pezzi
FROM vendite
WHERE anno = 2024;
```

totale_vendite	totale_pezzi
158750.50	1243

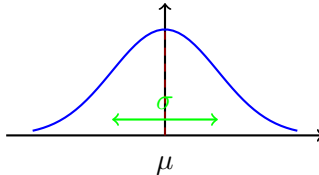
Nota

SUM ignora i valori NULL nel calcolo.

Operatori AVG e STDDEV

```
SELECT
  AVG(stipendio) AS media_stipendio,
  STDDEV(stipendio) AS deviazione_standard
FROM impiegati;
```

media_stipendio	deviazione_standard
30000.00	4082.48



La Clausola GROUP BY

Funzione

La clausola GROUP BY serve per raggruppare ed elaborare in modo uniforme diverse righe che nella tabella origine hanno valori uguali in una determinata colonna.

Sintassi:

```
SELECT colonna1, funzione_aggregata(colonna2)
FROM tabella
WHERE condizione
GROUP BY colonna1;
```

Gruppo A

Gruppo B

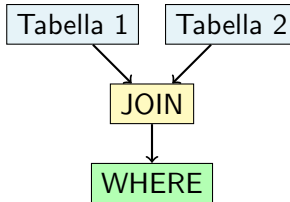
Gruppo C

Ordine di esecuzione

Tenendo conto delle considerazioni che seguono, facciamo chiarezza sulle priorità durante l'esecuzione di una selezione con raggruppamento.

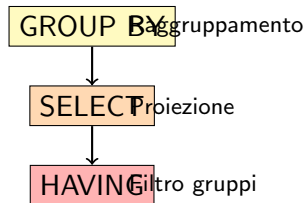
Fasi di elaborazione:

- 1 Esecuzione della JOIN, se presente
- 2 Selezione sulle tuple con la condizione dopo WHERE



Fasi di elaborazione (continua):

- ③ Raggruppamento in base ai campi specificati dopo GROUP BY
- ④ Lista dei valori specificati dopo la SELECT (target list)
- ⑤ Selezione delle tuple che soddisfano la condizione dopo HAVING



La Clausola GROUP BY

```
SELECT reparto, COUNT(*) AS numero_impiegati
FROM impiegati
GROUP BY reparto;
```

Tabella originale:

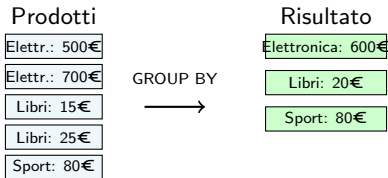
Nome	Reparto
Mario	Vendite
Laura	IT
Giuseppe	Vendite
Anna	IT
Carlo	Marketing

Risultato:

reparto	numero_impiegati
Vendite	2
IT	2
Marketing	1

La Clausola GROUP BY

```
SELECT categoria, AVG(prezzo) AS prezzo_medio
FROM prodotti
GROUP BY categoria;
```



La Clausola GROUP BY

```
SELECT
    anno ,
    mese ,
    SUM(vendite) AS totale_vendite
FROM vendite_mensili
GROUP BY anno , mese
ORDER BY anno , mese;
```

anno	mese	totale_vendite
2023	1	45000
2023	2	52000
2024	1	48000
2024	2	55000

Raggruppamento su **più colonne**

La Clausola GROUP BY

```
SELECT
    citta,
    COUNT(*) AS numero_clienti,
    SUM(ordini) AS totale_ordini
FROM clienti
GROUP BY citta;
```

citta	numero_clienti	totale_ordini
Roma	15	234
Milano	22	387
Napoli	8	145
Torino	12	198

La Clausola GROUP BY

```
SELECT
    marca ,
    modello ,
    COUNT(*) AS unita_vendute ,
    AVG(prezzo) AS prezzo_medio
FROM auto_vendute
GROUP BY marca, modello
ORDER BY unita_vendute DESC;
```

marca	modello	unità	prezzo
Fiat	Panda	45	12500
Volkswagen	Golf	38	22000
Ford	Fiesta	32	15000

La Clausola GROUP BY

```
SELECT
    YEAR(data_vendita) AS anno,
    MONTH(data_vendita) AS mese,
    COUNT(*) AS numero_transazioni,
    SUM(importo) AS fatturato
FROM transazioni
GROUP BY YEAR(data_vendita), MONTH(data_vendita);
```

anno	mese	transazioni	fatturato
2024	10	152	45200.00
2024	11	178	52800.00
2024	12	201	68500.00

Raggruppamento con **funzioni** nelle colonne

La Clausola HAVING

```
SELECT reparto , AVG(stipendio) AS media_stipendio
FROM impiegati
GROUP BY reparto
HAVING AVG(stipendio) > 30000;
```

Tutti i gruppi:

reparto	media_stipendio
Vendite	28000
IT	35000
Marketing	27000
Produzione	32000

Risultato con HAVING:

reparto	media_stipendio
IT	35000
Produzione	32000

La Clausola HAVING

```
SELECT categoria, COUNT(*) AS numero_prodotti
FROM prodotti
GROUP BY categoria
HAVING COUNT(*) >= 5;
```

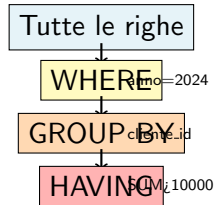
categoria	numero_prodotti
Elettronica	12
Abbigliamento	8
Casa	6

Nota

HAVING filtra i gruppi **dopo** l'aggregazione, mentre WHERE filtra le righe **prima**.

La Clausola HAVING

```
SELECT
    cliente_id,
    SUM(importo) AS totale_acquisti
FROM ordini
WHERE anno = 2024
GROUP BY cliente_id
HAVING SUM(importo) > 10000;
```



La Clausola HAVING

```
SELECT
    citta,
    COUNT(*) AS numero_ordini,
    AVG(importo) AS importo_medio
FROM ordini
GROUP BY citta
HAVING COUNT(*) > 10 AND AVG(importo) > 500;
```

citta	numero_ordini	importo_medio
Milano	25	675.50
Roma	18	820.30

Condizioni **multiple** in HAVING

La Clausola HAVING

```
SELECT
    marca,
    COUNT(*) AS modelli,
    MIN(prezzo) AS prezzo_min,
    MAX(prezzo) AS prezzo_max
FROM automobili
GROUP BY marca
HAVING MAX(prezzo) - MIN(prezzo) > 20000;
```

marca	modelli	prezzo_min	prezzo_max
BMW	8	35000	85000
Mercedes	6	40000	95000

HAVING con **espressioni** tra aggregazioni

Limitazione delle Tuple Risultato

Clausola LIMIT

Permette di limitare il numero di righe restituite dalla query.

Sintassi:

```
SELECT colonne  
FROM tabella  
WHERE condizione  
ORDER BY colonna  
LIMIT numero;
```

Utilizzi comuni:

- Visualizzare solo i primi N risultati
- Implementare la paginazione
- Ottimizzare le performance delle query

Limitazione delle Tuple Risultato

```
SELECT nome, cognome, stipendio
FROM impiegati
ORDER BY stipendio DESC
LIMIT 5;
```

Top 5 stipendi più alti:

nome	cognome	stipendio
Anna	Neri	45000
Mario	Rossi	42000
Laura	Bianchi	38000
Giuseppe	Verdi	35000
Carlo	Gialli	33000

Restituisce solo le prime 5 righe ordinate

Limitazione delle Tuple Risultato

Paginazione con OFFSET

OFFSET permette di saltare un numero specificato di righe.

```
SELECT nome , prezzo  
FROM prodotti  
ORDER BY prezzo DESC  
LIMIT 10 OFFSET 20;
```



Skip 20



Get 10



Altri...

Utile per la paginazione: pagina 3 = LIMIT 10 OFFSET 20

Limitazione delle Tuple Risultato

```
SELECT
    prodotto ,
    vendite ,
    fatturato
FROM statistiche_vendite
ORDER BY fatturato DESC
LIMIT 3;
```

Top 3 prodotti per fatturato:

prodotto	vendite	fatturato
Laptop Pro	234	234000 €
Smartphone X	456	182400 €
Tablet Plus	198	99000 €

Best Practice

Usare sempre ORDER BY con LIMIT per risultati consistenti.

Limitazione delle Tuple Risultato

Vantaggi di LIMIT

- **Performance:** riduce il carico sul database
- **Memoria:** trasferisce meno dati
- **Usabilità:** migliora l'esperienza utente
- **Testing:** utile per testare query su grandi dataset

```
-- Query di test su tabella grande  
SELECT * FROM log_eventi  
WHERE data >= '2024-01-01'  
LIMIT 100;
```

