

CONTINUAZIONE OPERATORI AGGREGAZIONE:

/// **GROUP BY** \Rightarrow Suddividere il **Risultato in GRUPPI** \rightarrow RAGGRUPPAMENTO
Che Condividono lo **Stesso Valore di**
un Attributo

ESEMPIO: tutti gli Studenti di Verona

```
SELECT  
FROM  
[WHERE...]  
GROUP BY  
[HAVING...]
```

dato: **STUDENTE** (matricola, nome, Cognome, Indirizzo, città, Media)

/// **Contare Tutti gli Studenti** `SELECT count(*) FROM STUDENTE`
 \neq
`count(città)` perché NON so
se città sia NOT NULL o
meno
e a sua
Volta
 \neq
`count(DISTINCT città)`

/// **Contare tutti gli Studenti Per Ogni Città:**

```
SELECT città, count(*)  
FROM Studente
```

Date le RIGHE della Tab Studenti, si
Creano GRUPPI Per Ogni Città e Solamente

GROUP BY città

Poi Viene fatto il COUNT V GRUPPO

Con GROUP BY, Nella Select Posso Indicare Solo Attributi che Compaiono Nella Clausola GROUP BY + operatori di AGGREGAZIONE visto che il Risultato avrà 1 Riga per Ogni GRUPPO

Con CLAUSOLA WHERE:

```
SELECT Studente  
FROM ①  
WHERE ② matricola <> 'VROO1'  
GROUP BY ③ città
```

- ① determino Righe d'Interesse
- ② Raggruppo \Rightarrow GROUP BY
- ③ Filtra gli Attributi
- ④

WHERE mi Permette di Selezionare le RIGHE sul Quale Calcolare il Risultato.

GROUP BY \Rightarrow Raggruppare le Righe Restituite dal WHERE

Voglio Una Selezione Anche Sui Gruppi \Rightarrow HAVING

/// SOLO CITTÀ CON 10 STUDENTI:

↪ con Aggiunta:
/// Attributi GROUP BY
/// Operatori

```
SELECT città, COUNT(*) ④ Restituisco  
FROM Studente ① Seleziono Righe di Pertenza per il RISULTATO  
GROUP BY città ② Calcolo i Gruppi  
HAVING COUNT(*)  $\geq$  10 ③ Seleziono i Gruppi
```

SOTTOLINEARE DIFFERENZA TRA HAVING e WHERE

```
SELECT Lower(città), avg(media)::DECIMAL(5,2)
FROM Studente
WHERE
GROUP BY Lower(città)
HAVING avg(media) > 23.5
```

VISUALIZZAZIONE

ho Raggruppato in
modo CASE-INSENSITIVE

'Verona'
'VERONA'
'verona' } stesso Gruppo

Deve Esserci COERENZA
anche Nella SELECT.

Se NON Avessi messo Lower(città)
ci Sarebbero ECCEZIONI → ERRORE
visto che Avrei Colonne diverse MA
Nello Stesso Gruppo

Risultato del TIPO ⇒

CITTÀ	MEDIA
Verona	24.00
Padova	23.50

Non Aspettato... Ma il

CASTING può IMBROGLIARE infatti

Padova poteva Avere 23.50g e Quindi facendo CAST
a 2 cifre tornerà 23.50

OPERATORE DI JOIN:

/// METODO BASE:

SELECT *
FROM a, b, c

Mi porta ad un PRODOTTO CARTESIANO
che poi filtri con un WHERE

/// UTILIZZO CLAUSOLA DI JOIN:

SELECT *
FROM t₁ TIPO JOIN t₂ ON

Tipi di JOIN:

/// Inner \Leftrightarrow con il WHERE

/// Full

/// Right

/// Left

} NON OTTENIBILI ALTRIMENTI

INNER-JOIN: θ -Join dell'Algebra Relazionale

Combina ciascuna Riga di t₁ con ogni Riga di t₂ che Soddisfano la Condizione di JOIN

IMPIEGATO (matricola, Nome, Cognome, nomeReparto)

REPARTO (nome, indirizzo, tel, città)

SELECT I.nome, I.cognome, I.nomeReparto, R.tel

FROM Impiegato AS I, REPARTO AS R

WHERE I.nomeReparto = R.nome

oppure:

```
FROM Impiegato JOIN REPARTO ON I.nomeReparto = R.nome
```

Voglio tutti gli Impiegati e per Quelli Assegnati ad un Reparto Anche il Numero di Telefono

```
SELECT I.nome, I.cognome, I.nomeReparto, R.tel  
FROM Impiegato I LEFT JOIN Reparto R ON I.nomeRep = R.nome
```



Esegue un **INNER-JOIN** e per Ciascuna Riga di T_{1} (impiegato) che non ha un CORRISPONDENTE in T_{2} (REPARTO) aggiunge la RIGA r assegnando a NULL gli Attributi di T_{2}

Quindi \forall Impiegati che non hanno un Reparto Assegnato aggiungiamo al Risultato una Riga per quell' Impiegato con $tel = NULL$

C'è Anche il DUALE ovvero il **RIGHT [OUTER] JOIN** dove andrò a Tenere le Rigue di T_{2} che non hanno un CORRISPONDENTE dove tutti Attr. di $T_{1} = NULL$

T_1 **FULL OUTER** JOIN T_2 dove Calcola

① INNER	} Includo Tutti
② LEFT	
③ RIGHT	

NB!

→ diverso da Prodotto CARTESIANO visto che prima Combino SOLO ciò che ha corrispondenza, altrimenti posso

Usare il CROSS-JOIN

ESEMPIO:

T_1	A_1	A_2
	1	a
	2	b
	3	c

T_2	A_3	A_4
	a	10
	a	14
	b	18
	f	20

SELECT * FROM $T_1, T_2 \Rightarrow$ TUTTE COMBINAZIONI

SELECT * FROM T_1 JOIN T_2 ON $t_1.A_2 = t_2.A_3 \Rightarrow$ ESCLUDERÒ L'ULTIMA RIGA DI ENTRAMBE LE TABELLE

SELECT * FROM T_1 LEFT-JOIN T_2 ON $t_1.A_2 = t_2.A_3 \Rightarrow$ ALLORA AVRÒ TUTTE LE RIGHE DI T_1 MA PER L'ULTIMA RIGA AVRÒ (A_3, A_4) CHE SONO = NULL

SELECT * FROM T_1 RIGHT-JOIN T_2 ON $t_1.A_2 = t_2.A_3 \Rightarrow$ ALLORA AVRÒ TUTTE LE RIGHE DI T_2 MA PER L'ULTIMA RIGA AVRÒ (A_1, A_2) CHE SONO = NULL

SELECT * FROM T_1 FULL-JOIN T_2 ON $t_1.A_2 = t_2.A_3 \Rightarrow$ ALLORA AVRÒ TUTTE LE RIGHE DI T_1 MA PER L'ULTIMA RIGA AVRÒ (A_3, A_4) CHE SONO = NULL MA ANCHE TUTTE LE RIGHE DI T_2 MA L'ULTIMA NON AVENDO CORRISPONDENZA HA ATTRIBUTI $(A_1, A_2) = \text{NULL}$

BASE DI DATI: