



ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Il Linguaggio SQL DML/DQL

Basi di Dati

Corso di Laurea in Informatica per il Management

Alma Mater Studiorum - Università di Bologna

Prof. Marco Di Felice

Dipartimento di Informatica – Scienza e Ingegneria

marco.difelice3@unibo.it

Il Linguaggio SQL

- **LINGUAGGI** supportati dai **RDBMS**

3. SQL (*Structured Query Language*)

Diverse **versioni** del linguaggio:

- SQL-86 → Costrutti base
- SQL-89 → (**SQL1**) Integrità referenziale
- SQL-92 (**SQL2**) → SQL Interattivo, sistema tipi
- SQL:1999 (**SQL3**) → Modello ad oggetti
- SQL:2003 (**SQL3**) → Nuove parti: SQL/JRT, SQL/XML
- SQL:2006 (**SQL3**) → Estensione di SQL/XML
- SQL:2008 (**SQL3**) → Lievi aggiunte
-

<http://troels.arvin.dk/db/rdbms/>

Il Linguaggio SQL

Due **componenti** principali:

- **DDL** (*Data Definition Language*)
Contiene i costrutti necessari per la creazione e modifica dello **schema** della base di dati.



- **DML/DQL** (*Data Manipulation/Query Language*)
Contiene i costrutti per le interrogazioni e di inserimento, eliminazione e modifica di **dati**.

SQL: DML/DQL

Esempio di **interrogazione (query)** → Recuperare nome e cognome dello studente con numero di matricola pari a 4678...

STUDENTI

<u>Matricola</u>	Nome	Cognome	DataNascita
4566	Marco	Rossi	3/5/1989
4678	Michele	Bianchi	2/5/1989
4900	Antonio	Rossi	14/3/1990



Nome	Cognome
Michele	Bianchi

SQL: DML/DQL

Le operazioni di **interrogazione** vengono implementate dal costrutto di select.

```
select  Attributo1, ... AttributoM  
from    Tabella1, ... ,TabellaN  
where   Condizione
```

SEMANTICA: Effettua il **prodotto cartesiano** delle Tabella₁, .., Tabella_N. Da queste, **estrai le righe** che rispettano la **Condizione**. Di quest'ultime, preleva solo le colonne corrispondenti a: **Attributo₁, ...,Attributo_M**.

SQL: DML/DQL

Nel caso di una sola tabella:

select $\text{Attributo}_i, \text{Attributo}_j, \dots \text{Attributo}_m$
from Tabella
where Condizione

STEP 1: Si selezionano le ennuple della tabella che soddisfano la condizione ...

TABELLA	
	Attributo_1 ... Attributo_i ... Attributo_j ... Attributo_m
e_1	
e_2	
e_3	
e_4	

SQL: DML/DQL

Nel caso di una sola tabella:

select $\text{Attributo}_i, \text{Attributo}_j, \dots \text{Attributo}_m$
from Tabella
where Condizione

STEP 2: Si selezionano le colonne/attributi specificati dalla SELECT ...

	TABELLA					
	Attributo ₁	...	Attributo _i	...	Attributo _j	Attributo _m
e ₁						
e ₂						
e ₃						
e ₄						

SQL: DML/DQL

Nel caso di una sola tabella:

select $\text{Attributo}_i, \text{Attributo}_j, \dots \text{Attributo}_m$
from Tabella
where Condizione

STEP 3: Si costruisce la tabella risultato ...



Numero di colonne definito dalla clausola **SELECT**



Numero di righe
definito dalla
clausola **WHERE**



Attributo_1	Attributo_i	Attributo_m

SQL: DML/DQL

Esempio 1. Selezionare i nomi degli impiegati che lavorano nell'ufficio A.

IMPIEGATI						
Codice	Nome	Cognome	Ufficio	Stipendio		
123	Marco	Marchi	A	15000		Nome
125	Michele	Monti	B	18000		Marco
134	Antonio	Verdi	A	25000		Antonio
156	Giorgio	Rossi	A	32000		Giorgio

SELECT NOME FROM IMPIEGATI WHERE (UFFICIO="A")

SQL: DML/DQL

Esempio 2. Selezionare i nomi degli impiegati che guadagnano più di 20000 euro annui.

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	15000
125	Michele	Monti	B	18000
134	Antonio	Verdi	A	25000
156	Giorgio	Rossi	A	32000




Nome
Antonio
Giorgio

SELECT NOME FROM IMPIEGATI WHERE (STIPENDIO>20000)

SQL: DML/DQL

Esempio 3. Selezionare nomi e cognomi degli impiegati che lavorano nell'ufficio B e guadagnano più di 20000 euro annui.

IMPIEGATI						
Codice	Nome	Cognome	Ufficio	Stipendio		
123	Marco	Marchi	A	15000		
125	Michele	Monti	B	18000		
134	Antonio	Verdi	A	25000		
156	Giorgio	Rossi	A	32000		



Nome	Cognome
------	---------

```
SELECT NOME, COGNOME FROM IMPIEGATI WHERE ((STIPENDIO > 20000) AND (UFFICIO = "B"))
```

SQL: DML/DQL

La clausola **where** specifica quali **righe della/e tabella/e** devono **comparire nel risultato finale**.

La condizione della clausola può contenere un'espressione booleana, o una combinazione di espressioni mediante gli operatori **and**, **or**, **not**.

```
SELECT CODICE  
FROM IMPIEGATI  
WHERE NOT((NOME="Marco") AND (UFFICIO="A"))
```

SQL: DML/DQL

Nella clausola **where**, è possibile fare **confronti tra stringhe** usando l'operatore **like** e l'utilizzo di **wildcard**:

_ → carattere arbitrario

% → sequenza di caratteri arbitraria.

In questo modo, è possibile trovare tutte le stringhe che **rispettano un certo pattern**. Es: selezionare il codice di tutti gli impiegati il cui nome inizi per 'M', abbia una 'r' come terzo carattere, e termini per 'o'.

```
SELECT CODICE  
FROM IMPIEGATI  
WHERE (NOME LIKE 'M_R%O')
```

SQL: DML/DQL

Nella clausola **where**, l'operatore **between** consente di verificare **l'appartenenza ad un certo insieme di valori**.

Es. Trovare i codici degli impiegati il cui stipendio sia compreso tra i 24000 ed i 34000 euro annui.

IMPIEGATI						
Codice	Nome	Cognome	Ufficio	Stipendio		
123	Marco	Marchi	A	15000		Nome
125	Michele	Monti	B	18000		Antonio
134	Antonio	Verdi	A	25000		Giorgio
156	Giorgio	Rossi	A	32000		

SELECT NOME FROM IMPIEGATI WHERE (STIPENDIO BETWEEN 24000 AND 34000)

SQL: DML/DQL

D. Cosa accade nella clausola **where** in caso di valori **NULL**...
Vengono inclusi nel risultato finale? **NO!**

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	15000
125	Michele	Monti	B	18000
134	Antonio	Verdi	A	NULL
156	Giorgio	Rossi	A	32000



Nome
Giorgio

SELECT NOME FROM IMPIEGATI WHERE (STIPENDIO > 20000)

SQL: DML/DQL

In generale, SQL utilizza una **logica a tre valori**: **true (T)**, **false (F)**, **unknown (U)**. Esistono gli operatori **IS NULL** ed **IS NOT NULL**.

IMPIEGATI						
Codice	Nome	Cognome	Ufficio	Stipendio		
123	Marco	Marchi	A	15000		Nome
125	Michele	Monti	B	18000		Antonio
134	Antonio	Verdi	A	NULL		Giorgio
156	Giorgio	Rossi	A	32000		

**SELECT NOME FROM IMPIEGATI WHERE ((STIPENDIO > 20000)
OR (STIPENDIO IS NULL))**

SQL: DML/DQL

La clausola **select** specifica quali **colonne delle righe selezionate devono comparire nel risultato finale**.

L'asterisco (*) indica tutte le colonne della tabella.

```
SELECT * FROM IMPIEGATI  
WHERE (NOME="Marco") AND (UFFICIO="A")
```



IMPIEGATI				
Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	15000

SQL: DML/DQL

E' possibile **ridenominare** le colonne del risultato di una query attraverso il costrutto as.

```
SELECT NOME as Name, Cognome as LastName  
FROM IMPIEGATI  
WHERE (NOME="Marco")
```

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	C	36000




Name	LastName
Marco	Marchi
Marco	Bianchi

SQL: DML/DQL

E' possibile usare *espressioni aritmetiche* (semplici) sui valori degli attributi di una **SELECT**.

```
SELECT NOME as Name, Stipendio/12 as SalaryM
FROM IMPIEGATI
WHERE (NOME="Marco")
```

IMPIEGATI						
Codice	Nome	Cognome	Ufficio	Stipendio		
123	Marco	Marchi	A	12000		
145	Marco	Bianchi	B	24000		
167	Lucia	Di Lucia	C	36000		
					Name	SalaryM
					Marco	1000
					Marco	2000

SQL: DML/DQL

La clausola **from** specifica la **lista delle tabelle cui si deve accedere** (nel caso #tabelle>1, si effettua il **prodotto cartesiano** delle stesse).

E' possibile specificare degli alias per i nomi delle tabelle, mediante il costrutto **as**:

```
SELECT CODICE  
FROM IMPIEGATI AS I  
WHERE (I.NOME="MICHELE")
```

SQL: DML/DQL

Vediamo come funziona la **SELECT su più tabelle**.

Es. Selezionare il numero di telefono dell'impiegato con codice 145

IMPIEGATI				
Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000

SEDI	
UffNum	Telefono
A	2034333
B	2035434

SQL: DML/DQL

```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI, SEDI  
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

... COSA FA QUESTA QUERY??

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000

SEDI

UffNum	Telefono
A	2034333
B	2035434

SQL: DML/DQL

```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI, SEDI  
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP 1. Si effettua il **prodotto
cartesiano** delle due tabelle ...

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

SQL: DML/DQL

SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)

STEP 2. Si selezionano le righe con
valori comuni nelle tue tabelle ...

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

SQL: DML/DQL

SELECT TELEFONO AS TEL
FROM IMPIEGATI, SEDI
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)

STEP 3. Si selezionano le righe
relative all'impiegato 145 ...

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

SQL: DML/DQL

```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI, SEDI  
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```

STEP 4. Si seleziona la **colonna**
dell'attributo Telefono ...

Codice	Nome	Cognome	Ufficio	Stipendio	UffNum	Telefono
123	Marco	Marchi	A	12000	A	2034333
145	Marco	Bianchi	B	24000	A	2034333
167	Lucia	Di Lucia	A	36000	A	2034333
187	Giorgio	Rossi	B	12000	A	2034333
123	Marco	Marchi	A	12000	B	2035434
145	Marco	Bianchi	B	24000	B	2035434
167	Lucia	Di Lucia	A	36000	B	2035434
187	Giorgio	Rossi	B	12000	B	2035434

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

SQL: DML/DQL

```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI, SEDI  
WHERE (UFFICIO=UFFNUM) AND (CODICE=145)
```



TEL
2035434

STEP 5. Si costruisce il risultato finale ...

SQL: DML/DQL

D. Che accade se le tabelle della clausola from hanno attributi con nomi uguali?

```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI, SEDI  
WHERE (UFFICIO=UFFICIO) AND (CODICE=145)
```

???? ERRORE!!!

IMPIEGATI

Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000

SEDI

Ufficio	Telefono
A	2034333
B	2035434

SQL: DML/DQL

In questi casi, si può utilizzare la notazione **NomeTabella.NomeAttributo** per far riferimento ad un attributo in maniera non ambigua.


```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI, SEDI  
WHERE (IMPIEGATI.UFFICIO=SEDI.UFFICIO) AND (CODICE=145)
```

```
SELECT TELEFONO AS TEL  
FROM IMPIEGATI AS I, SEDI AS S  
WHERE (I.UFFICIO=S.UFFICIO) AND (CODICE=145)
```

SQL: DML/DQL

ATTENZIONE: Il risultato di una query SQL **potrebbe avere righe duplicate!**

```
SELECT NOME AS NAME  
FROM IMPIEGATI AS I  
WHERE (STIPENDIO < 25000)
```

IMPIEGATI						
Codice	Nome	Cognome	Ufficio	Stipendio		
123	Marco	Marchi	A	12000		Name
145	Marco	Bianchi	B	24000		Marco
167	Lucia	Di Lucia	C	36000		Marco

SQL: DML/DQL

Il costrutto `distinct` (nella `select`) consente di **rimuovere i duplicati** nel risultato.

Il costrutto `all` (nella `select`) **NON** rimuove i duplicati (comportamento di default).

```
SELECT DISTINCT NOME AS NAME  
FROM IMPIEGATI AS I  
WHERE (STIPENDIO >20000)
```



Name
Marco

SQL: DML/DQL

Nella clausola **where** possono comparire **più istanze della stessa tabella mediante il meccanismo degli alias ...**

Es. Selezionare *i nomi dei nonni di Matteo Bianchi*.

GENITORI			
Nome	Cognome	NomeGen	Cognome Gen
Matteo	Bianchi	Michele	Bianchi
Michele	Bianchi	Gianni	Bianchi
Matteo	Bianchi	Lucia	Rossi
Lucia	Rossi	Sara	Rossi
Nicola	Verdi	Simone	Verdi

SQL: DML/DQL

```
SELECT NOME, COGNOME  
FROM GENITORI, GENITORI  
WHERE (GENITORI.NOME=GENITORI.NOMEGEN) ...
```

???? ERRORE!!!



GENITORI			
Nome	Cognome	NomeGen	CognomeGen
Matteo	Bianchi	Michele	Bianchi
Michele	Bianchi	Gianni	Bianchi
Matteo	Bianchi	Lucia	Rossi
Lucia	Rossi	Sara	Rossi
Nicola	Verdi	Simone	Verdi

SQL: DML/DQL

```
SELECT G2.NOMEGEN, G2.COGNOMEGEN
FROM GENITORI AS G1, GENITORI AS G2
WHERE (G1.NOMEGEN=G2.NOME) AND (G1.COGNOMEGEN=G2.COGNOME)
AND (G1.NOME="MATTEO") AND (G1.COGNOME="BIANCHI")
```

GENITORI			
Nome	Cognome	NomeGen	CognomeGen
Matteo	Bianchi	Michele	Bianchi
Michele	Bianchi	Gianni	Bianchi
Matteo	Bianchi	Lucia	Rossi
Lucia	Rossi	Sara	Rossi
Nicola	Verdi	Simone	Verdi

SQL: DML/DQL

Il costrutto **order by** consente di **ordinare le righe** del risultato di un'interrogazione in base al valore di un attributo specificato.

order by Attributo₁ [asc|desc], ..., Attributo_N
[asc|desc]

```
SELECT *  
FROM IMPIEGATI  
WHERE (UFFICIO="A")  
ORDER BY STIPENDIO
```


← Deve comparire sempre
dopo la clausola where!

SQL: DML/DQL

Supponiamo di voler scrivere una **query** per contare il numero di Impiegati che lavorano nell'ufficio A.

Problema: La SELECT vista fin qui opera **a livello di tuple**, e non **a livello di colonne** ..

IMPIEGATI				
Codice	Nome	Cognome	Ufficio	Stipendio
123	Marco	Marchi	A	12000
145	Marco	Bianchi	B	24000
167	Lucia	Di Lucia	A	36000
187	Giorgio	Rossi	B	12000



Da questa colonna
dovremmo estrarre un
solo valore!

SQL: DML/DQL

Gli operatori **aggregati** si applicano a gruppi di tuple (e non tupla per tupla), e **producono come risultato un solo valore.**

Vengono in genere inseriti nella `select`, e valutati DOPO la clausola `where` e `from`.

`count (* | [distinct|all] Lista Attributi)`

* → si applica su tutti gli attributi, in pratica **conta il numero di righe**

SQL: DML/DQL

Gli operatori **aggregati** si applicano a gruppi di tuple (e non tupla per tupla), e **producono come risultato un solo valore.**

- **sum** (Lista Attributi)
- **avg** (Lista Attributi)
- **min** (Lista Attributi)
- **max** (Lista Attributi)

SQL: DML/DQL

Sintassi Generale:

SELECT **OP**(Attributo)
FROM ListaTabelle
WHERE Condizione

sum
max
min
avg
count

count(*)

STEP 0: Si considerano le tabelle indicate nella clausola **FROM**

T ₁		
...
...

T ₂		
...
...

T _N		
...
...

SQL: DML/DQL

Sintassi Generale:

SELECT **OP**(Attributo)

FROM ListaTabelle

WHERE Condizione

sum
max
min
avg
count

count(*)

STEP 1: Si effettua il **prodotto cartesiano** delle tabelle.

...								

SQL: DML/DQL

Sintassi Generale:

SELECT **OP**(Attributo)
FROM ListaTabelle
WHERE Condizione

sum
max
min
avg
count

count(*)

STEP 2: Si selezionano le righe che soddisfano la condizione del **WHERE**.

...								

SQL: DML/DQL

Sintassi Generale:

SELECT OP(Attributo)

FROM ListaTabelle

WHERE Condizione

sum
max
min
avg
count

count(*)

STEP 3: Si considera l'Attributo della **SELECT** e si applica l'operatore aggregato su tutti i valori della colonna.

...								

SQL: DML/DQL

Sintassi Generale:

SELECT OP(Attributo)

FROM ListaTabelle

WHERE Condizione

sum
max
min
avg
count

count(*)

STEP 4: Dalla colonna si calcola **un solo valore** come risultato della query



Se non si usa l'operatore
AS, la colonna risultato
non ha un nome...

SQL: DML/DQL

Es. Contare il numero di strutturati che lavorano nel *Dipartimento di Fisica*.

STRUTTURATI

<u>Codice</u>	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

```
SELECT COUNT(*) AS CONTATORE  
FROM STRUTTURATI  
WHERE (DIPARTIMENTO="FISICA")
```



Contatore
3

SQL: DML/DQL

Es. Contare la somma complessiva degli stipendi degli strutturati *del dipartimento di Fisica*.

STRUTTURATI					
<u>Codice</u>	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

```
SELECT SUM(STIPENDIO) AS TOTALE  
FROM STRUTTURATI  
WHERE (DIPARTIMENTO="FISICA")
```



Totale
70000

SQL: DML/DQL

Es. Determinare il valore dello *stipendio più alto* tra i professori associati.

STRUTTURATI

<u>Codice</u>	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	50000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

```
SELECT MAX(STIPENDIO) AS MAXSTIPENDIO  
FROM STRUTTURATI  
WHERE (TIPO="ASSOCIATO")
```



MaxStipendio
50000

SQL: DML/DQL

Es. Estrarre codice e stipendio del professore associato che ha lo stipendio più alto ...

```
SELECT CODICE, MAX(STIPENDIO)
FROM STRUTTURATI
WHERE (TIPO="ASSOCIATO")
```

ERRORE!



L'operatore aggregato **restituisce un solo valore**, mentre la prima parte della SELECT **restituisce un valore per ogni tupla selezionata!!!**

COME FARE? Con **interrogazioni annidate** (vedi prossimi blocchi ...)

SQL: DML/DQL

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento.

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento.

STRUTTURATI					
Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	RISULTATO	Micheli	Associato	Fisica	20000
125	Dipartimento		Numero		Fisica
126	Chimica		1		30000
127	Fisica		3		32000
129	Informatica		2		15000
				Fisica	20000

SQL: DML/DQL

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento.

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
--------	------	---------	------	--------------	-----------

Soluzione 1:

```
SELECT COUNT(*) AS NUMERO  
FROM STRUTTURATI
```

NON FA QUANTO RICHIESTO!!!



Numero

6

SQL: DML/DQL

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento.

STRUTTURATI					
Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio

Soluzione 2:

```
SELECT COUNT(*) AS NUMERO, DIPARTIMENTO  
FROM STRUTTURATI
```

QUERY ERRATA!!!

SQL: DML/DQL

Es. Contare il numero di strutturati (ricercatori + professori) di ciascun dipartimento.

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
--------	------	---------	------	--------------	-----------

Soluzione 3:

```
SELECT COUNT(*) AS NUMERO  
FROM STRUTTURATI  
WHERE (Dipartimento='Fisica')
```

Come faccio a generalizzare questa query?

Numero

3

SQL: DML/DQL

Operatori di query visti fin qui:

- **SELECT ATTRIBUTI FROM WHERE** → Valuta i valori di ciascuna riga **in isolamento**.
- **SELECT OP(ATTRIBUTI) FROM WHERE** → Valuta i valori delle righe corrispondenti alle colonne della SELECT **in modo aggregato**.

D. Esiste la possibilità di **combinare i due approcci**?

SQL: DML/DQL

Operatori di query visti fin qui:

- **SELECT ATTRIBUTI FROM WHERE** → Valuta i valori di ciascuna riga **in isolamento**.
- **SE** Estrarre informazioni aggregate da gruppi di righe... i valori delle righe corrispondenti alle colonne della **SELECT in modo aggregato**.

D. Esiste la Possibilità di **combinare i due approcci**?

SQL: DML/DQL

L'operatore di **raggruppamento** consente di dividere la tabella in **gruppi**, ognuno caratterizzata da un valore comune dell'attributo specificato nell'operatore.

```
SELECT ListaAttributi1  
FROM ListaTabelle  
WHERE Condizione  
GROUPBY ListaAttributi2
```

ListaAttributi1 deve essere un sottoinsieme di ListaAttributi2, puo' contenere operatori aggregati!

Ogni gruppo produce **una sola riga** nel risultato finale!

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO
```



DIP	Numero
Chimica	1
Fisica	3
Informatica	2

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO
```

STRUTTURATI

STEP 1: Partizionamento della tabella

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO
```

STEP 1: Partizionamento della tabella

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
129	Michele	Bianchi	Associato	Fisica	20000

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000

BASI DI DATI

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO
FROM STRUTTURATI
GROUPBY DIPARTIMENTO
```

STEP 2: Si applica la select su ciascun gruppo

Codice	Nome	Cognome	Dipartimento	Numero	Stipendio
123	Marco	M	Chimica	1	20000

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
124	Michele	M	ASSOCIATO	Fisica	20000
125	Lucia	D	ASSOCIATO		30000
129	Michele	Bianchi	ASSOCIATO		20000

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
126	Dario	R	ASSOCIATO	Informatica	32000
127	Mario	R	ASSOCIATO		15000

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP, COUNT(*) AS NUMERO  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO
```

STEP 3: Si costruisce il risultato finale



Dip	Numero
Chimica	1
Fisica	3
Informatica	2

SQL: DML/DQL

Es. Calcolare, per ogni dipartimento, lo stipendio medio degli strutturati.

STRUTTURATI

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP, AVG(STIPENDIO) AS  
STIPENDIOMEDIO  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO
```



Dip	StipendioMedio
Chimica	20000
Fisica	23333
Informatica	23500

SQL: DML/DQL

Attenzione! Nella SELECT possono comparire solo un **sottoinsieme degli attributi** della clausola GROUPBY oppure operatori aggregati.

STRUTTURATI

Come faccio ad estrarre 1 sola riga??

Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Fisica	20000
124	Michele	Micheli	Associato	Fisica	20000

```
SELECT NOME, COUNT(*) AS NUMERO  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO
```

ERRORE!

SQL: DML/DQL

E' possibile **filtrare** i gruppi in base a determinate condizioni, attraverso il costrutto **having**.

```
SELECT ListaAttributi1
```

```
...
```

```
GROUPBY ListaAttributi2
```

```
HAVING Condizione
```

- clausola **where** → valutata riga per riga.
- clausola **having** → valutata **su ciascun gruppo**, contiene operatori aggregati o condizioni su **ListaAttributi2**.

SQL: DML/DQL

Sintassi Generale:

```
SELECT ListaAttributi1  
FROM ListaTabelle  
WHERE Condizione  
GROUPBY ListaAttributi2  
HAVING Condizione
```

STEP 0: Prodotto cartesiano delle tabelle
+ Estrazione delle righe che rispettano
la condizione della clausola WHERE

...								

BASI DI DATI

PROF. MARCO DI FELICE – CORSO DI LAUREA IN INFORMATICA PER IL MANAGEMENT

SQL: DML/DQL

Sintassi Generale:

```
SELECT ListaAttributi1  
FROM ListaTabelle  
WHERE Condizione  
GROUPBY ListaAttributi2  
HAVING Condizione
```

STEP 1: Partizionamento della tabella

...								
...								

SQL: DML/DQL

Sintassi Generale:

```
SELECT ListaAttributi1  
FROM ListaTabelle  
WHERE Condizione  
GROUPBY ListaAttributi2  
HAVING Condizione
```

STEP 2: Selezione dei gruppi

...								

SQL: DML/DQL

Sintassi Generale:

```
SELECT ListaAttributi1  
FROM ListaTabelle  
WHERE Condizione  
GROUPBY ListaAttributi2  
HAVING Condizione
```



STEP 3: Selezione dei valori delle colonne o esecuzione degli operatori aggregati su ciascuno dei gruppi, e composizione della tabella finale.

SQL: DML/DQL

Es. Estrarre il nome dei dipartimenti che hanno **almeno due strutturati** nel suo organico.

STRUTTURATI					
Codice	Nome	Cognome	Tipo	Dipartimento	Stipendio
123	Marco	Marchi	Associato	Chimica	20000
124	Michele	Micheli	Associato	Fisica	20000
125	Lucia	Di Lucia	Ordinario	Fisica	30000
126	Dario	Rossi	Ordinario	Informatica	32000
127	Mario	Rossi	Ricercatore	Informatica	15000
129	Michele	Bianchi	Associato	Fisica	20000

SQL: DML/DQL

```
SELECT DIPARTIMENTO AS DIP  
FROM STRUTTURATI  
GROUPBY DIPARTIMENTO  
HAVING COUNT(*) >= 2
```



DIP
Fisica
Informatica

SQL: DML/DQL

Costrutto `select` nella sua forma più generale.

`SELECT` ListaAttributi

`FROM` ListaTabelle

`WHERE` Condizione

`LIMIT` Number

`GROUPBY` AttributiRaggruppamento

`HAVING` CondizioniGruppi

`ORDERBY` ListaAttributiOrdinamento

SQL: DML/DQL

In SQL, è possibile effettuare **operazioni insiemistiche** tra tabelle o in generale tra risultati di SELECT:

- UNION [ALL]
- INTERSECT [ALL]
- EXCEPT [ALL]

Gli attributi della SELECT devono avere **tipi di dato compatibili** e (possibilmente) gli stessi nomi.

SQL: DML/DQL

Es. Estrarre nome e cognome di tutto il personale universitario (strutturati + tecnici).

STRUTTURATI

Codice	Nome	Cognome	Ruolo
123	Marco	Marchi	Associato
124	Michele	Micheli	Ordinario
125	Lucia	Di Lucia	Ricercatore
126	Dario	Rossi	Ordinario
127	Mario	Rossi	Ordinario
129	Michele	Bianchi	Associato

TECNICI

Codice	Nome	Cognome	Livello
445	Michele	Marini	5
356	Daniele	Marini	6
154	Giovanna	Bianchi	5
156	Lucia	Di Lucia	4

SQL: DML/DQL

Es. Estrarre nome e cognome di tutto il personale universitario (strutturati + tecnici).

```
SELECT NOME, COGNOME  
FROM STRUTTURATI  
UNION  
SELECT NOME, COGNOME  
FROM TECNICI
```

Nome	Cognome
Marco	Marchi
Michele	Micheli
Lucia	Di Lucia
Dario	Rossi
Mario	Rossi
Michele	Bianchi
Michele	Marini
...	...

SQL: DML/DQL

Es. Estrarre nome e cognome degli strutturati che hanno degli omonimi che lavorano come tecnici.

```
SELECT NOME, COGNOME  
FROM STRUTTURATI
```

INTERSECT

```
SELECT NOME, COGNOME  
FROM TECNICI
```

Nome	Cognome
Lucia	Di Lucia

SQL: DML/DQL

Es. Estrarre nome e cognome degli strutturati che NON hanno degli omonimi che lavorano come tecnici ...

```
SELECT NOME, COGNOME  
FROM STRUTTURATI  
EXCEPT  
SELECT NOME, COGNOME  
FROM TECNICI
```

Nome	Cognome
Marco	Marchi
Michele	Micheli
Dario	Rossi
Mario	Rossi
Michele	Bianchi
Michele	Marini

SQL: DML/DQL

Attenzione. Gli attributi delle SELECT nelle due tabelle devono avere **tipi compatibili** ...

```
SELECT RUOLO  
FROM STRUTTURATI  
UNION  
SELECT LIVELLO  
FROM TECNICI
```

ERRORE!

STRUTTURATO.Ruolo e' una **stringa**
TECNICI.Livello e' un **intero**.

SQL: DML/DQL

Oltre ad i comandi di interrogazione, la parte DML definisce anche le operazioni per la **modifica dell'istanza** della base di dati.

- **insert** → inserisce una o più righe.
- **delete** → cancella una o più righe.
- **update** → aggiorna un attributo o più.

SQL: DML/DQL

E' possibile **inserire una riga** **esplicitando i valori degli attributi** oppure estraendo le righe da altre tabelle del database.

- **insert** into NomeTabella [ListaAttributi]
values (ListaValori)

```
INSERT INTO IMPIEGATI(Codice, Nome, Cognome,  
Ufficio) values ('8','Vittorio','Rossi','A')
```

SQL: DML/DQL

E' possibile **inserire una riga** **esplicitando i valori degli attributi** oppure estraendo le righe da altre tabelle del database.

- **insert** into NomeTabella [ListaAttributi]
values (ListaValori)

```
INSERT INTO IMPIEGATI(Codice, Nome, Cognome)  
values('8', 'Vittorio', 'Rossi')
```

Ufficio → non
specificato, NULL o default

SQL: DML/DQL

E' possibile **inserire una riga** esplicitando i valori degli attributi oppure **estraendo le righe da altre tabelle del database.**

- **insert** into NomeTabella SQLSelect

```
INSERT INTO IMPIEGATI  
(Codice,Nome,Cognome,Ufficio) VALUES (  
    SELECT *  
    FROM IMPIEGATICOMUNE)
```

SQL: DML/DQL

E' possibile **cancellare** tutte le righe che soddisfano una condizione (cancella tutto se non specificata).

- **delete** from Tabella where Condizione

```
DELETE FROM IMPIEGATI
```

```
DELETE FROM IMPIEGATI WHERE (UFFICIO="A")
```

```
DELETE FROM TABELLA WHERE NOME IN (  
    SELECT NOME FROM IMPIEGATICOMUNE)
```

SQL: DML/DQL

E' possibile **aggiornare** il contenuto di uno o più attributi di una tabella che rispettano una certa condizione.

- **update** NomeTabella
set attributo = expr|SELECT|null|default
[where Condizione]

UPDATE IMPIEGATI
SET NOME="Mario"
WHERE (CODICE=5)

SQL: DML/DQL

E' possibile **aggiornare** il contenuto di **uno o più attributi di una tabella** che rispettano una certa **condizione**.

- UPDATE IMPIEGATI SET NOME='MARCO' WHERE (CODICE=5)
- UPDATE IMPIEGATI SET NOME=(SELECT NOME FROM IMPIEGATICOMUNE WHERE CODICE=5) WHERE (CODICE=5)

SQL: DML/DQL

E' possibile implementare il **join** tra tabelle in **due modi** distinti (ma equivalenti nel risultato):

- Inserendo le condizioni del JOIN direttamente nella clausola del WHERE
- Attraverso l'utilizzo dell'operatore di **inner JOIN** nella clausola FROM

```
SELECT ListaAttributi  
FROM Tabella JOIN Tabella ON CondizioneJoin  
[WHERE Condizione]  
...
```


SQL: DML/DQL

Esempio di query con utilizzo dell'**INNER join**.

GUIDATORI

<u>NrPatente</u>	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

```
SELECT Modello
FROM GUIDATORI, VEICOLI
WHERE (GUIDATORI.NrPatente=
      VEICOLI.NrPatente) AND
      (Nome="Sara")
```

VEICOLI

<u>Targa</u>	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

```
SELECT Modello
FROM GUIDATORI JOIN VEICOLI
ON GUIDATORI.NrPatente
=VEICOLI.NrPatente
WHERE (Nome="Sara")
```

SQL: DML/DQL

Esistono altre **tre** varianti (poco usate) dell'operatore di JOIN

- **LEFT join** → risultato dell'inner join + righe della tabella di sinistra che non hanno un corrispettivo a destra (completate con valori NULL)

```
SELECT ListaAttributi  
FROM Tabella LEFT JOIN Tabella ON CondizioneJoin  
[WHERE Condizione]  
...
```

SQL: DML/DQL

Esempio di query con utilizzo del **LEFT join**.

GUIDATORI

<u>NrPatente</u>	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

<u>Targa</u>	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

```
SELECT Modello  
FROM GUIDATORI LEFT JOIN VEICOLI ON  
GUIDATORI.NrPatente =VEICOLI.NrPatente
```

SQL: DML/DQL

Esempio di query con utilizzo del **LEFT** join.

GUIDATORI

<u>NrPatente</u>	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

<u>Targa</u>	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4567	Yaris	5687876

NrPatente	Nome	Cognome	Targa	Modello	NrPatente
1243242	Sara	Bianchi	BO2121	Panda	1243242
1243242	Sara	Bianchi	BO4567	Panda	1243242
2656565	Michele	Rossi	NULL	NULL	NULL

SQL: DML/DQL

Esistono altre **tre** varianti (poco usate) dell'operatore di JOIN

- **RIGHT join** → risultato dell'inner join + righe della **tabella di destra** che non hanno un corrispettivo a destra (completate con valori NULL)

```
SELECT ListaAttributi  
FROM Tabella RIGHT JOIN Tabella ON CondizioneJoin  
[WHERE Condizione]  
...
```

SQL: DML/DQL

Esempio di query con utilizzo del **RIGHT join**.

GUIDATORI

<u>NrPatente</u>	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

<u>Targa</u>	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

NrPatente	Nome	Cognome	Targa	Modello	NrPatente
1243242	Sara	Bianchi	BO2121	Panda	1243242
1243242	Sara	Bianchi	BO4567	Panda	1243242
NULL	NULL	NULL	BO4896	Yaris	5687876

SQL: DML/DQL

Esistono altre **tre** varianti (poco usate) dell'operatore di JOIN

- **FULL join** → risultato dell'inner join + righe della tabella di sinistra/destra che non hanno un corrispettivo a destra/sinistra (completate con valori NULL)

```
SELECT ListaAttributi  
FROM Tabella FULL JOIN Tabella ON CondizioneJoin  
[WHERE Condizione]  
...
```

SQL: DML/DQL

Esempio di query con utilizzo del **FULL** join.

GUIDATORI

<u>NrPatente</u>	Nome	Cognome
1243242	Sara	Bianchi
2656565	Michele	Rossi

VEICOLI

<u>Targa</u>	Modello	NrPatente
BO2121	Panda	1243242
BO4567	Punto	1243242
BO4896	Yaris	5687876

NrPatente	Nome	Cognome	Targa	Modello	NrPatente
1243242	Sara	Bianchi	BO2121	Panda	1243242
1243242	Sara	Bianchi	BO4567	Panda	1243242
NULL	NULL	NULL	BO4896	Yaris	5687876
2656565	Michele	Rossi	NULL	NULL	NULL