



UNIVERSITÀ DEGLI STUDI DI SALERNO

Atzeni, Ceri, Fraternali, Paraboschi, Torlone  
 Basi di dati *Quinta edizione*  
 McGraw-Hill Education, 2018

1

## BASI DI DATI

Concetti Base SQL-DML - seconda parte



**Matteo Gaeta**

Full Professor – Senior Member IEEE

*La matematica non conosce razze o confini geografici; per la matematica, il mondo culturale è una singola nazione. (David Hilbert)*

## BASI DI DATI

2

Concetti base – SQL

ISTRUZIONE SELECT - VERSIONE BASE

### ➤ Interrogazione:

- ✓ **SELECT** ListaAttributi
- FROM** ListaTabelle
- [ **WHERE** Condizione ]

- clausola **SELECT** (chiamata *target list*)
- clausola **FROM** (*DA*)
- clausola **WHERE** (*DOVE*)

CLAUSOLE ISTRUZIONE SELECT

- ❖ Le interrogazioni SQL, selezionano, tra le righe del prodotto cartesiano delle tabelle indicate nella clausola FROM quelle che soddisfano le condizioni espresse nell'argomento della clausola WHERE

La matematica è linguaggio [...] più logica.  
(R. P. Feynman)

## BASI DI DATI

3

Concetti base – SQL ISTRUZIONE SELECT - VERSIONE ESTESA

### ➤ Sintassi più completa ... Warning all'applicazione della clausola Having

#### ● Sintassi (quasi) generale

- **Select** <elenco attributi>
- From** <elenco tabelle>
- [**Where** <condizione>]
- [**Group by** <attributi di raggruppamento>]
- [**Having** <condizione di raggruppamento>]
- [**Order by** <elenco attributi>]



La più alta categoria dell'intelletto immaginativo è sempre eminentemente matematica.  
(E. A. Poe)

## BASI DI DATI

4

Concetti base – SQL

TABELLE DI RIFERIMENTO

### ➤ SELEZIONE E PROIEZIONE

**Una Query:** elencare il Nome e Reddito delle persone con meno di trenta anni

✓ **PROJ**<sub>Nome, Reddito</sub>(**SEL**<sub>Eta < 30</sub>(**Persone**))

❖ **SELECT** Nome, Reddito  
**FROM** Persone  
**WHERE** Eta < 30

Maternità	Madre	Figlio
	Luisa	Maria
	Luisa	Luigi
	Anna	Olga
	Anna	Filippo
	Maria	Andre
	Maria	Aldo

Paternità	Padre	Figlio
	Sergio	Franco
	Luigi	Olga
	Luigi	Filippo
	Franco	Andre
	Franco	Aldo

Persone		
Nome	Età	Reddito
Andrea	27	21
Aldo	25	15
Maria	55	42
Anna	50	35
Filippo	26	30
Luigi	50	40
Franco	60	20
Olga	30	41
Sergio	85	35
Luisa	75	87

## BASI DI DATI

5

### Concetti base – SQL

### ISTRUZIONE SELECT

#### ➤ SELECT con le abbreviazioni

Una Query: elencare nome e reddito delle persone con meno di trenta anni

➤ **SELECT** Nome, Reddito  
**FROM** Persone  
**WHERE** eta < 30

In alternativa con le abbreviazioni

➤ **SELECT P.Nome as** Nome, **P.Reddito as** Reddito  
**FROM** Persone **as P**  
**WHERE** P.Eta < 30

#### ➤ SELECT selezione senza proiezioni

Una Query: elencare le informazioni delle persone con meno di trenta anni

✓ **SEL**<sub>Eta < 30</sub> (**Persone**)

➤ **SELECT** \*  
**FROM** Persone  
**WHERE** Eta < 30

## BASI DI DATI

6

### Concetti base – SQL

### ISTRUZIONE SELECT

#### ➤ SELECT con le abbreviazioni

Data la Relazione  $R(A,B)$

➤ **SELECT** \*  
**FROM** R

Equivale intuitivamente a:

➤ **SELECT X.A as** A, **X.B as** B  
**FROM** R **as X**  
**WHERE** TRUE

#### ➤ Select proiezione senza selezione

Una Query: Nome e Reddito di tutte le persone

✓ **PROJ**<sub>Nome, Reddito</sub> (**Persone**)

➤ **SELECT** Nome, Reddito  
**FROM** Persone

## BASI DI DATI

7

### Concetti base – SQL

### ISTRUZIONE SELECT

#### ➤ SELECT e ordinamento risultato

Una Query: Nome e reddito delle persone con meno di trenta anni in ordine alfabetico

```
➤ SELECT Nome, Reddito
FROM Persone
WHERE eta < 30
ORDER BY Nome;
```

❖ Per default l'ordine è «Ascending», salvo diverse impostazioni.

#### ➤ SELECT e ordinamento ASC o DESC

Una Query: Nome e reddito delle persone con meno di trenta anni ordinati “ascending” sul nome (in ordine crescente alfabetico) e “descending” sul Reddito (in ordine decrescente)

```
➤ SELECT Nome, Reddito
FROM Persone
WHERE Eta < 30
ORDER BY Nome ASC, Reddito DESC;
```

## BASI DI DATI

8

### Concetti base – SQL

### ISTRUZIONE SELECT

#### ➤ SELECT senza ordinamento

Una Query: Nome e reddito delle persone con meno di trenta anni

```
➤ SELECT Nome, Reddito
FROM Persone
WHERE eta < 30
```

#### Persone

Nome	Reddito
Andrea	21
Aldo	15
Filippo	30

#### ➤ SELECT e ordinamento ASC

Analoga Query con ordinamento

```
➤ SELECT Nome, Reddito
FROM Persone
WHERE Eta < 30
ORDER BY Nome;
```

#### Persone

Nome	Reddito
Aldo	15
Andrea	21
Filippo	30

## BASI DI DATI

9

### Concetti base – SQL

### ISTRUZIONE SELECT

- **SELECT con espressioni nella target list ovvero nella clausola Select**

**Una Query:** Elencare il Reddito semestrale delle persone di nome «Luigi»

- **SELECT** Reddito/2 as RedditoSemestrale  
**FROM** Persone  
**WHERE** Nome = 'Luigi'

- **Select con condizione complessa**

**Una Query:** Elencare tutte le informazioni delle persone che hanno un reddito maggiore di 25 ed una età < di 30 oppure maggiore di 60.

- **SELECT** \*  
**FROM** Persone  
**WHERE** Reddito > 25 and (Eta < 30 or Eta > 60)

## BASI DI DATI

10

### Concetti base – SQL

### SELECT E OPERATORE LIKE

- **SELECT, uso di operatore LIKE e del carattere speciale percentuale «%»**

**Una Query:** elencare nome e reddito delle persone che hanno un nome di qualsiasi lunghezza che inizia per "Ro" e finisce in "a". Ad esempio, Rosa, Rosita, Rosetta, ecc.

- **SELECT** Nome, Reddito  
**FROM** Persone  
**WHERE** Nome **LIKE** 'Ro%a'

- **SELECT, uso di operatore LIKE e del carattere speciale trattino sotto «\_»**

**Una Query:** elencare nome e reddito delle persone che hanno un nome di quattro lettere che inizia per "Ro" e finisce in "a". Ad esempio, Rosa, Roma, ecc.

- **SELECT** Nome, Reddito
- **FROM** Persone
- **WHERE** Nome **LIKE** 'Ro\_a'

## BASI DI DATI

11

### Concetti base – SQL

### SELECT E OPERATORE LIKE

- **SELECT**, uso di operatore **LIKE** e dei caratteri speciali «%» e «\_»

**Una Query:** elencare le persone che hanno un nome che inizia per 'A' e ha una 'd' come terza lettera

- **SELECT** Nome, Reddito  
**FROM** Persone  
**WHERE** Nome **LIKE** 'A\_d%'

- **SELECT**, uso di operatore **LIKE** e dei caratteri speciali «%» e «\_»

**Una Query:** elencare le persone che hanno un nome che termina con 'a' e ha una 'e' in seconda posizione

- **SELECT** Nome, Reddito  
**FROM** Persone  
**WHERE** Nome **LIKE** '\_a%a'

## BASI DI DATI

12

### Concetti base – SQL

### SELECT E PROIEZIONE

- **Attenzione alle ennuple duplicate come output delle Select**

- **SELECT** Cognome, Filiale  
**FROM** Impiegati

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma
Rossi	Roma

- **SELECT** e uso del **DISTINCT** per evitare ennuple duplicate

- **SELECT DISTINCT** Cognome, Filiale  
**FROM** Impiegati

Cognome	Filiale
Neri	Napoli
Neri	Milano
Rossi	Roma

## BASI DI DATI

Concetti base – SQL

13

### GESTIONE DEI VALORI NULLI

❖ **Elencare gli impiegati la cui età è o potrebbe essere maggiore di 40**

✓ **SEL** `Età > 40 OR Età IS NULL` (Impiegati)

➤ **SELECT \***

**FROM Impiegati**

**WHERE Età > 40 or Età is NULL**

#### ❖ Significato di NULL

Ad oggi sono circa 15 le interpretazioni fornite per il valore NULL, molto convergono sulle seguenti:

Valore **sconosciuto**: un valore che esiste ma non è conosciuto.

Valore **non esistente**: un valore che non esiste.

Gli SQLman non considerano il NULL un valore ma una sorta di "marcatore". Per essi non è una valore perché, se tale, dovrebbe comportarsi come tutti gli altri valori che si possono assegnare ad un tipo dato all'interno del DB. La specifica ANSI SQL-92 afferma che un NULL deve essere uguale per tutti i tipi di dati, in modo che tutti i NULL siano gestiti in modo coerente. Nel linguaggio corrente si sente comunque spesso parlare di **valore NULL**.

## BASI DI DATI

Concetti base – SQL

14

### GESTIONE DEI VALORI NULLI

❖ **Un valore NULL non è un valore, quindi non è > di, < di o = ad altri valori.**

❖ **Un null non equivale ad un altro null.**

❖ **Non è valido (anzi è insignificante) verificare se una colonna è = NULL, <NULL, <= NULL, > NULL o > = NULL**

#### TABELLA

CodiceID	Nome	Telefono	CodNazione	Info
001	Andrea	+39 340	1	1000
002	Nicola	+39 335	3	NULL
003	Laura	NULL	4	1500
004	Francesca	NULL	NULL	200

➤ **Select From Tabella Where Info = NULL;** -- **nessun record**

➤ **Select From Tabella Where Info IS NULL;** -- **record CodiceID 002**

➤ **Select From Tabella Where Info >= 800;** -- **records CodiceID 002 e 003**

➤ **Select From Tabella Where Info < 800;** -- **record CodiceID 004**

## BASI DI DATI

15

Concetti base – SQL

OPERATORI AGGREGATI

- Una Query si dice aggregante se contiene almeno un operatore di aggregazione.
- Nelle espressioni della target list possiamo avere anche espressioni che calcolano valori a partire da insiemi di ennuple:
  - ✓ Conteggio;
  - ✓ Minimo;
  - ✓ Massimo;
  - ✓ Media;
  - ✓ Totale
- sintassi base (semplificata):
  - Funzione ( [ DISTINCT ] \* )*
  - Funzione ( [ DISTINCT ] Attributo )*

## BASI DI DATI

16

Concetti base – SQL

OPERATORE COUNT

- Il numero di figli di Franco

```
select count(*) as NumFigliDiFranco
from Paternita
where Padre = 'Franco'
```

Paternità	Padre	Figlio	NumFigliDiFranco
	Sergio	Franco	2
	Luigi	Olga	
	Luigi	Filippo	
	Franco	Andrea	
	Franco	Aldo	

- l'operatore aggregato (count) viene applicato al risultato della query

```
select *
from Paternita
where Padre = 'Franco'
```



Specifica che count restituisce il numero di valori univoci non Null

## BASI DI DATI

17

Concetti base – SQL

OPERATORE COUNT

- La funzione **COUNT(\*)** restituisce il numero di elementi di un gruppo, inclusi valori NULL e duplicati.
- **COUNT(ALL expression)** valuta expression per ogni riga in un gruppo e restituisce il numero di valori non Null.
- **COUNT(DISTINCT expression)** valuta expression per ogni riga in un gruppo e restituisce il numero di valori univoci non Null.
  - ✓ **ALL** - Applica la funzione di aggregazione a tutti i valori. ALL funge da valore predefinito.
  - ✓ **DISTINCT** - Specifica che COUNT restituisce il numero di valori distinti NOT NULL

## BASI DI DATI

18

Concetti base – SQL

OPERATORI AGGREGATI

- Altri operatori aggreganti sono:
  - ✓ Minimo - **MAX**;
  - ✓ Massimo - **MIN**;
  - ✓ Media - **AVG**;
  - ✓ Totale - **SUM**

- *Media dei redditi dei figli di Franco*

```
SELECT AVG(reddito)
FROM persone JOIN paternita ON nome=figlio
WHERE padre='Franco'
```

## BASI DI DATI

19

### Concetti base – SQL

### COUNT E VALORI NULL

Query Editor Query History

```

1 DROP TABLE IF EXISTS Persone;
2
3 Create Table Persone(
4     Nome Varchar(20),
5     Eta Integer,
6     Reddito Integer Default null
7 );
8 insert into persone values ('Andrea', 27, 21);
9 insert into persone values ('Aldo', 25, NULL);
10 insert into persone values ('MARia', 55, 21);
11 insert into persone values ('Anna', 50, 35);
12 Select *
13 From Persone;

```

Data Output Explain Messages Notifications

	nome character varying (20)	eta integer	reddito integer
1	Andrea	27	21
2	Aldo	25	[null]
3	MARia	55	21
4	Anna	50	35

➤ Osserviamo il comportamento del Count in presenza di valori NULL

➤ *select count(\*) from persone*

❖ 4

➤ *select count(reddito) from persone*

❖ 3

➤ *select count(distinct reddito) from persone*

❖ 2

## BASI DI DATI

20

### Concetti base – SQL

### OPERATORI AGGREGATI E VALORI NULL

Funzione	Risultati		
	Insieme vuoto	Insieme o colonna tutta a null	Altri casi
COUNT(*)	0	Numero totale delle righe	Numero totale delle righe
COUNT(Campo)	0	0	Numero di righe dove Campo non è NULL
MAX, MIN	NULL	NULL	Il valore minimo o massimo trovato nella colonna
SUM	NULL	NULL	Somma dei valori non NULL della colonna
AVG	NULL	NULL	Media dei valori non NULL della colonna. Questo vale $SUM(Campo) / COUNT(Campo)$ . <sup>a</sup>

## BASI DI DATI

Concetti base – SQL

21

### OPERATORI AGGREGATI E TARGET LIST

- ❖ La Target List di una SELECT che usa operatori aggregati deve essere «OMOGENEA». Ad esempio *la Target List «nome, max(reddito)» NON E' OMOGENEA ... di chi è il nome in proiezione?*
  - *SELECT nome, MAX(reddito)*  
*FROM persone*
- ❖ *Il nome della persona con il reddito massimo va realizzata ad esempio con una query nidificata molto semplice.*
  - *SELECT \**  
*FROM persone*  
*WHERE reddito = (SELECT MAX(reddito)*  
*FROM persone)*
- ❖ *Esempio di Target List Omogenea*  
*SELECT MIN (eta), AVG(reddito)*  
*FROM persone*

## BASI DI DATI

Concetti base – SQL

22

### OPERATORI AGGREGATI E RAGGRUPPAMENTI

- *Le funzioni possono essere applicate a partizioni delle Relazioni utilizzando la Clausola GROUP BY listaAttributi*
- *Elencare il numero di figli di ciascun padre*
- *select Padre, count(\*) AS NumFigli*  
*from Paternita*  
*group by Padre*

*In pratica viene effettuata la Query senza Group By e poi si raggruppa e si applica l'operatore per ciascun gruppo*

Paternita

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

Padre	NumFigli
Sergio	1
Luigi	2
Franco	2

## BASI DI DATI

Concetti base – SQL

23

### HAVING

- *Having si usa quando la condizione da verificare si riferisce alla partizione*
- *I padri i cui figli sotto i 30 anni hanno un reddito medio maggiore di 20*

```
SELECT padre, AVG(f.reddito)
  FROM persone f JOIN paternita ON figlio = nome
 WHERE eta < 30
GROUP BY padre
HAVING AVG(f.reddito) > 20
```

## BASI DI DATI

Concetti base – SQL

24

### HAVING

- *Having si usa quando la condizione da verificare si riferisce alla partizione*
- *Elencare i padri i cui figli hanno un reddito medio maggiore di 25; mostrare padre e reddito medio dei figli*

```
SELECT padre, AVG(f.reddito)
  FROM persone f JOIN paternita ON figlio = nome
GROUP BY padre
HAVING avg(f.reddito) > 25
```

# BASI DI DATI

25

## Concetti base – SQL

### GROUP BY E VALORI NULL

A	B
1	11
2	11
3	null
4	null

Tabella R

**select B, count (\*)  
from R group by B**

B	count (*)
11	2
null	2

**select A, count (\*)  
from R group by A**

A	count (*)
1	1
2	1
3	1
4	1

**select A, count (B)  
from R group by A**

A	count (B)
1	1
2	1
3	0
4	0

# BASI DI DATI

26

## Materiale utilizzato e bibliografia

➤ **Le slide utilizzate dai docenti per le attività frontali sono in gran parte riconducibili e riprese dalle slide originali (con alcuni spunti parziali ripresi dai libri indicati) realizzate da:**

- ✓ autori del libro *Basi di Dati* (Atzeni e altri) testo di riferimento del corso *Basi di Dati* e sono reperibili su internet su molteplici link oltre che laddove indicato dagli stessi autori del libro;
- ✓ Prof.ssa Tiziana Catarci e dal dott. Ing. Francesco Leotta – corso di *Basi di Dati* dell'Università degli Studi La Sapienza di Roma al seguente link ed altri: <http://www.dis.uniroma1.it/~catarci/basidatGEST.html> (molto Interessanti anche le lezioni su YouTube).
- ✓ Prof. Luca Allulli e Umberto Nanni, *Libro Fondamenti di basi di dati*, editore HOEPLI (testo di facile lettura ed efficace).

➤ **Diverse slide su specifici argomenti utilizzate dai docenti per le attività frontali sono anche in parte riconducibili e riprese dalle slide originali – facilmente reperibili e accessibili su internet - realizzate da:**

Prof.ssa Roberta Aiello – corso *Basi di Dati* dell'Università di Salerno

Prof. Dario Maio - corso *Basi di Dati* dell'Università di Bologna al seguente link ed altri: <http://bias.csr.unibo.it/maio>

Prof. Marco Di Felice - corso *Basi di Dati* dell'Università di Bologna al seguente link ed altri: <http://www.cs.unibo.it/difelice/dbsi/>

Prof. Marco Maggini e prof. Franco Scarselli - corso *Basi di Dati* dell'Università di Siena ai seguenti link ed altri: [http://staff.icar.cnr.it/pontieri/didattica/LabSI/lezioni/\\_preliminari-DB1%20\(Maggini\).pdf](http://staff.icar.cnr.it/pontieri/didattica/LabSI/lezioni/_preliminari-DB1%20(Maggini).pdf)

Prof.ssa Raffaella Gentilini - corso *Basi di Dati* dell'Università di Perugia al seguente link ed altri: <http://www.dmi.unipg.it/raffaella.gentilini/BD.htm>

Prof. Enrico Giunchiglia - corso *Basi di Dati* dell'Università di Genova al seguente link ed altri: <http://www.star.dist.unige.it/~enrico/BasiDiDati/>

Prof. Maurizio Lenzerini - corso *Basi di Dati* dell'Università degli Studi La Sapienza di Roma al seguente link ed altri: <http://didattica.info.altervista.org/Quinta/Database2.pdf>

➤ The PostgreSQL Global Development Group - PostgreSQL nn.xx Documentation

➤ PostgreSQL (appendice - scaricabile dal sito del libro (area studenti) e [www.postgresql.org](http://www.postgresql.org))