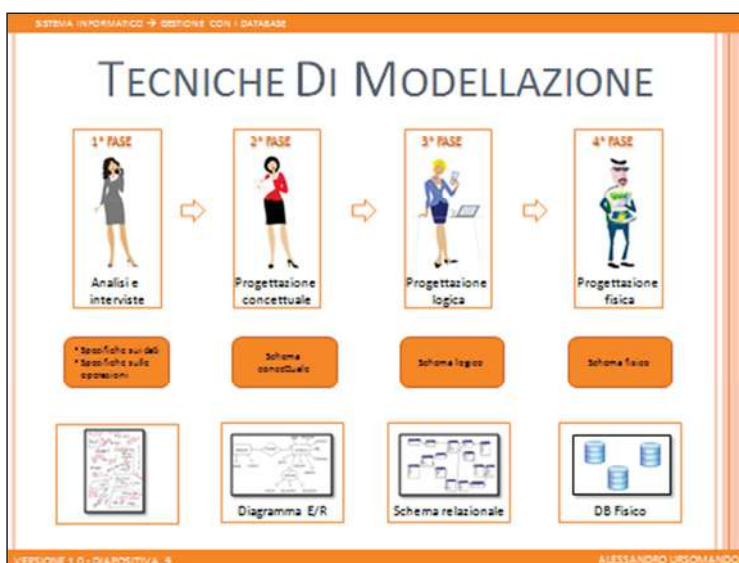


# SCHEMA FISICO

## Introduzione

## REALIZZAZIONE DI UN DB




Fasi di modellazione di un db:


- Analisi e interviste
- Progettazione concettuale (ER)
- Progettazione logica (schema relazionale)
- Progettazione fisica (db in MySQL)**


Come interagiamo con MySQL?

Come convertiamo lo schema relazionale per MySQL?

# USO DI MYSQL

Tutte le **relazioni** dello **schema relazionale** diventeranno delle **tabelle** del nostro database. 

Con MySql potremo **creare** il **database**, i suoi **utenti**, le sue **tabelle**; potremo **popolare** le tabelle; potremo **selezionare** dati e molto altro. 

MySql (come altri DBMS) permette di gestire il database sia con **strumenti visuali** che con l'esecuzione di **script**. 



# STRUMENTI VISUALI

Il nostro strumento visuale sarà phpmyadmin (già presente in xampp): interfaccia grafica tipo webapp.



PHPMYADMIN

localhost/phpmyadmin 

# REALIZZAZIONE DI SCRIPT

MySQL (come ogni altro DBMS) ha il proprio **dialetto** SQL. 


I comandi per la definizione dello schema fisico (**metadati**) fanno parte di un sottolinguaggio che si chiama **Data Definition Language**.


DDL


I comandi per la manipolazione dei dati fanno parte di un sottolinguaggio che si chiama **Data Manipulation Language**.

DML

# CONFIGURAZIONE

L'unica configurazione indispensabile per la realizzazione di script è relativa alla codifica dei caratteri. 

Posto che già sappiamo come creare dei file codificati in **UTF-8** vediamo come configurare MySQL in modo tale da fargli acquisire e mantenere dati e metadati in **UTF-8**. 

Aprire il file: `xampp\mysql\bin\my.ini` 

Aggiungere nella sezione `[mysqld]` le seguenti righe di codice:

```
default-character-set = utf8
character-set-server = utf8
collation-server = utf8_general_ci
```

Aggiungere nella sezione `[mysql]` la seguente riga di codice:

```
default-character-set = utf8
```

# SCHEMA FISICO

Comandi MYSQL per la gestione degli utenti

## CREAZIONE DI UN UTENTE

La creazione di un utente avviene in due passi:  
prima creo l'utente e poi gli attribuisco dei privilegi.



La gestione dei privilegi può essere molto complicata,  
per i nostri scopi è sufficiente vedere come garantire  
tutti i privilegi su tutte le tabelle di tutti i database.



```
create user <nomeUtente>@<nomeHost> identified by <password>;  
grant all privileges on *.*  
to <nomeUtente>@<nomeHost> identified by <password>;
```

```
create user alessandro@localhost identified by pippo;  
grant all privileges on *.*  
to alessandro@localhost identified by pippo;
```

# CANCELLAZIONE DI UN UTENTE

La cancellazione di un utente è molto banale.



```
drop user <nomeUtente>
```


```
drop user alessandro
```


## SCHEMA FISICO

Comandi MYSQL per definire la struttura del db (i metadati)

# CREAZIONE E CANCELLAZIONE DI DB


```
drop database <nomeDB>;  
create database <nomeDB>;  
use <nomeDB>;
```


I primi **comandi per la definizione** dei dati (DDL) sono **drop** e **create**. 


La prima volta che si esegue lo script la drop darà errore. Si può ovviare inserendo **"if exists"** prima del nome del database.. 

# CREAZIONE E CANCELLAZIONE DI DB

```
drop database if exists <nomeDB>;  
create database <nomeDB>;  
use <nomeDB>;
```

I primi **comandi per la definizione** dei dati (DDL) sono **drop** e **create**. 

La prima volta che si esegue lo script la drop darà errore. Si può ovviare inserendo **"if exists"** prima del nome del database.. 

Con il comando **use** evitiamo di dover specificare in ogni istruzione successiva il nome del db. 

# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
    <nomeCampo> <tipo>,
    ...
    <nomeCampo> <tipo>
) engine=innodb;
```

Specificare il **tipo**  
per ciascun campo è obbligatorio.



Tipo	Descrizione		Dimensione massima/formato
Bigint	intero lunghissimo	INTERI	da $-2^{63}$ a $2^{63}-1$
Integer	intero lungo		da $-2^{31}$ a $2^{31}-1$
Smallint	intero		da -32768 a 32767
Tinyint	intero ridotto		da -128 a +127
Double	reale a doppia precisione	REALI	da $\pm 2.225 \cdot 10^{-308}$ a $\pm 1.798 \cdot 10^{308}$
Float	reale a singola precisione		da $\pm 1.176 \cdot 10^{-38}$ a $\pm 3.403 \cdot 10^{38}$
Decimal	decimale memorizzato come stringa	DATA	da $\pm 2.225 \cdot 10^{-308}$ a $\pm 1.798 \cdot 10^{308}$
Date	data in formato US (aaaa-mm-gg)		dal '1000-01-01' a '9999-12-31'
Time	orario		formato HH:MM:SS
Datetime	data con ora		aaaa-mm-gg hh:mm:ss
Year	anno	STRINGA	formato AAAA
Character	stringa a lunghezza fissa		da 0 a 255 caratteri
Varchar	stringa a lunghezza variabile		da 0 a 255 caratteri
Text	campo testo a lunghezza fissa		da 0 a 65535 caratteri
Mediumtext	campo testo (memo)	OGGETTI	16 MB di caratteri -1
Blob	immagini jpeg, bmp, gif		fino 64 KB
Medium blob	(Binary Large Object), oppure file binary in esadecimale o di testo		fino a 16 MB
Long blob			circa 4 GB

# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
    <nomeCampo> <tipo>,
    ...
    <nomeCampo> <tipo>
) engine=innodb;
```

Specificare il **tipo** per ciascun campo è obbligatorio.



**Innodb** è il tipo di motore MySql che permette di specificare anche i vincoli di integrità tra tabelle



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
  <nomeCampo> <tipo>(<lunghezza>),
  <nomeCampo> unsigned <tipo>,
  <nomeCampo> <tipo> not null,
  <nomeCampo> <tipo> default <valore>,
  <nomeCampo> <tipo> auto_increment,
  ...
  <nomeCampo> <tipo>,
  index <nomeIndice> (<nomeCampo>),
  primary key (<nomeCampo>),
  foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
    on delete [ cascade | set null | set default | restrict ]
    on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Per alcuni tipi viene specificata anche la **dimensione**.



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
  <nomeCampo> <tipo>(<lunghezza>),
  <nomeCampo> unsigned <tipo>,
  <nomeCampo> <tipo> not null,
  <nomeCampo> <tipo> default <valore>,
  <nomeCampo> <tipo> auto_increment,
  ...
  <nomeCampo> <tipo>,
  index <nomeIndice> (<nomeCampo>),
  primary key (<nomeCampo>),
  foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
    on delete [ cascade | set null | set default | restrict ]
    on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Per i **tipi numerici** si può specificare la clausola **unsigned** (senza segno) che consente di usare solo i valori positivi, raddoppiando la capacità rappresentativa





# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
  <nomeCampo> <tipo>(<lunghezza>),
  <nomeCampo> unsigned <tipo>,
  <nomeCampo> <tipo> not null,
  <nomeCampo> <tipo> default <valore>,
  <nomeCampo> <tipo> auto_increment,
  ...
  <nomeCampo> <tipo>,
  index <nomeIndice> (<nomeCampo>),
  primary key (<nomeCampo>),
  foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
    on delete [ cascade | set null | set default | restrict ]
    on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Un campo **non opzionale** riporta la clausola **not null**.



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
  <nomeCampo> <tipo>(<lunghezza>),
  <nomeCampo> unsigned <tipo>,
  <nomeCampo> <tipo> not null,
  <nomeCampo> <tipo> default <valore>,
  <nomeCampo> <tipo> auto_increment,
  ...
  <nomeCampo> <tipo>,
  index <nomeIndice> (<nomeCampo>),
  primary key (<nomeCampo>),
  foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
    on delete [ cascade | set null | set default | restrict ]
    on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

È possibile specificare un valore di **default**.



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;

create table <nomeTabella> (
    <nomeCampo> <tipo>(<lunghezza>),
    <nomeCampo> unsigned <tipo>,
    <nomeCampo> <tipo> not null,
    <nomeCampo> <tipo> default <valore>,
    <nomeCampo> <tipo> auto_increment,
    ...
    <nomeCampo> <tipo>,
    index <nomeIndice> (<nomeCampo>),
    primary key (<nomeCampo>),
    foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
        on delete [ cascade | set null | set default | restrict ]
        on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Lo specificatore  
**auto\_increment**

viene usato per le chiavi primarie  
fittizie (di tipo int).

L'utente ha facoltà di inserire un  
valore ma qualora non lo facesse,  
il DBMS si occupa di calcolare ed  
impostare un valore diverso dai  
precedenti.



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;

create table <nomeTabella> (
    <nomeCampo> <tipo>(<lunghezza>),
    <nomeCampo> unsigned <tipo>,
    <nomeCampo> <tipo> not null,
    <nomeCampo> <tipo> default <valore>,
    <nomeCampo> <tipo> auto_increment,
    ...
    <nomeCampo> <tipo>,
    index <nomeIndice> (<nomeCampo>),
    primary key (<nomeCampo>),
    foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
        on delete [ cascade | set null | set default | restrict ]
        on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Dopo avere specificato tutti i campi,  
possiamo indicare indici e chiavi.



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
  <nomeCampo> <tipo>(<lunghezza>),
  <nomeCampo> unsigned <tipo>,
  <nomeCampo> <tipo> not null,
  <nomeCampo> <tipo> default <valore>,
  <nomeCampo> <tipo> auto_increment,
  ...
  <nomeCampo> <tipo>,
  index <nomeIndice> (<nomeCampo>),
  primary key (<nomeCampo>),
  foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
    on delete [ cascade | set null | set default | restrict ]
    on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Per una chiave esterna dobbiamo indicare la **tabella** e il **campo** a cui fa riferimento.



# CREAZIONE DI TABELLE

```
drop database if exists <nomeDB>;
create database <nomeDB>;
use <nomeDB>;
```

```
create table <nomeTabella> (
  <nomeCampo> <tipo>(<lunghezza>),
  <nomeCampo> unsigned <tipo>,
  <nomeCampo> <tipo> not null,
  <nomeCampo> <tipo> default <valore>,
  <nomeCampo> <tipo> auto_increment,
  ...
  <nomeCampo> <tipo>,
  index <nomeIndice> (<nomeCampo>),
  primary key (<nomeCampo>),
  foreign key (<nomeCampo>) references <nomeAltraTabella> (<nomeCampo>)
    on delete [ cascade | set null | set default | restrict ]
    on update [ cascade | set null | set default | restrict ]
) engine=innodb;
```

Per una chiave esterna dobbiamo indicare la **tabella** e il **campo** a cui fa riferimento.



È invece facoltativo indicare il comportamento in caso di **cancellazione** o **modifica** del valore a cui si fa riferimento.



# CANCELLAZIONE E MODIFICA DI TABELLE

```
drop table <nomeTabella>;
alter table <nomeTabellaVecchio> rename <nomeTabellaNuovo>;
alter table <nomeTabella> drop <nomeCampo>;
alter table <nomeTabella> change <nomeCampoVecchio> <nomeCampoNuovo> <tipo>;
alter table <nomeTabella> add <nomeCampo> <tipo>;
```

# CANCELLAZIONE E MODIFICA DI TABELLE


```
drop table <nomeTabella>;
alter table <nomeTabellaVecchio> rename <nomeTabellaNuovo>;
alter table <nomeTabella> drop <nomeCampo>;
alter table <nomeTabella> change <nomeCampoVecchio> <nomeCampoNuovo> <tipo>;
alter table <nomeTabella> add <nomeCampo> <tipo>;
```

# CANCELLAZIONE E MODIFICA DI TABELLE

```
drop table <nomeTabella>;  
alter table <nomeTabellaVecchio> rename <nomeTabellaNuovo>;  
alter table <nomeTabella> drop <nomeCampo>;  
alter table <nomeTabella> change <nomeCampoVecchio> <nomeCampoNuovo> <tipo>;  
alter table <nomeTabella> add <nomeCampo> <tipo>;
```

# CANCELLAZIONE E MODIFICA DI TABELLE

```
drop table <nomeTabella>;  
alter table <nomeTabellaVecchio> rename <nomeTabellaNuovo>;  
alter table <nomeTabella> drop <nomeCampo>;  
alter table <nomeTabella> change <nomeCampoVecchio> <nomeCampoNuovo> <tipo>;  
alter table <nomeTabella> add <nomeCampo> <tipo>;
```

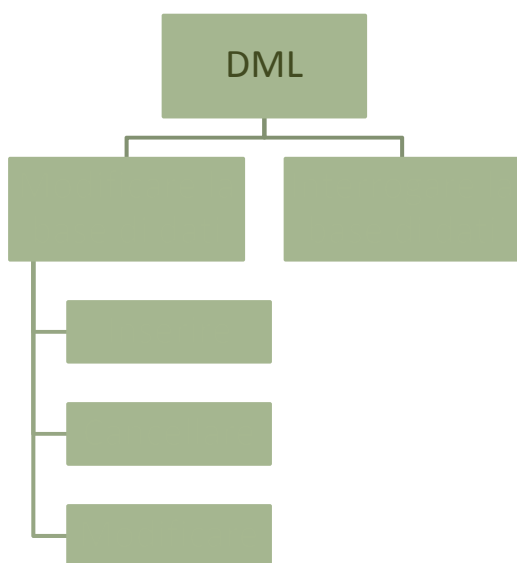
 Nel cambiare o aggiungere un campo alla tabella sono possibili tutti gli scenari visti all'atto della creazione:

- specifica della **taglia**
- specifica di **unsigned**
- specifica del **not null**
- specifica del valore di **default**
- specifica dell'**auto\_increment**
- specifica di chiave **primaria** o **esterna**

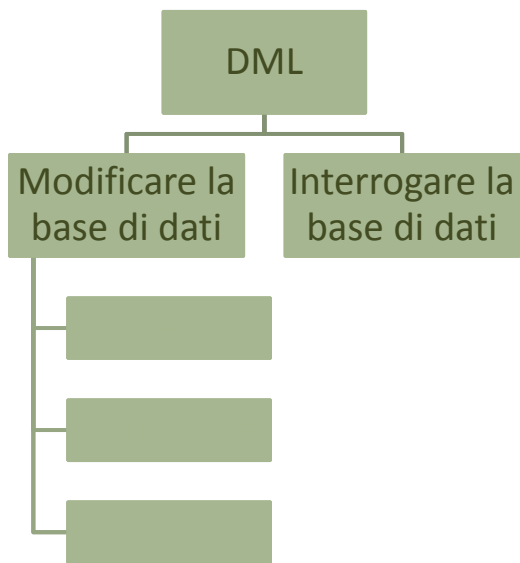
# SCHEMA FISICO

Comandi MYSQL per manipolare i dati: introduzione

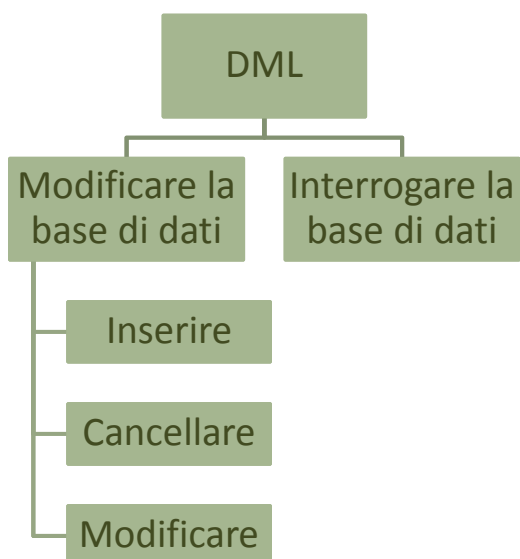
## INTRODUZIONE



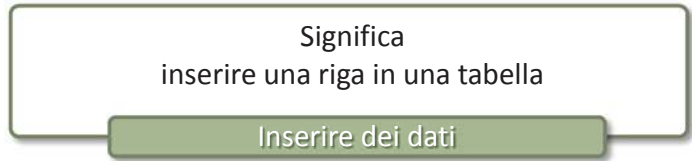
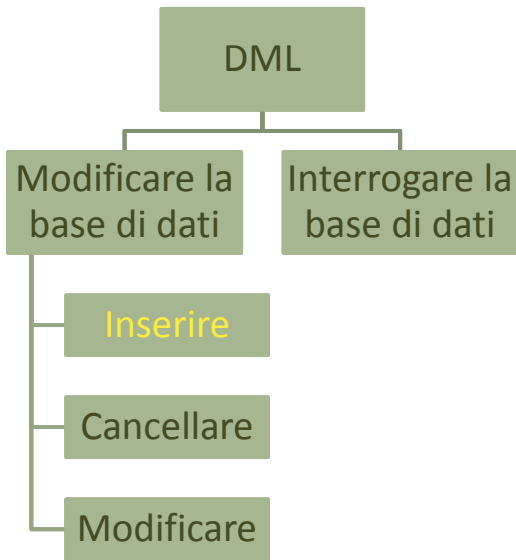
# INTRODUZIONE



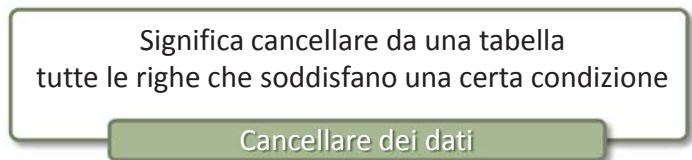
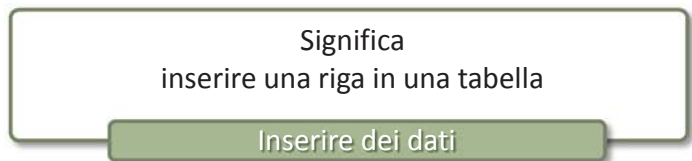
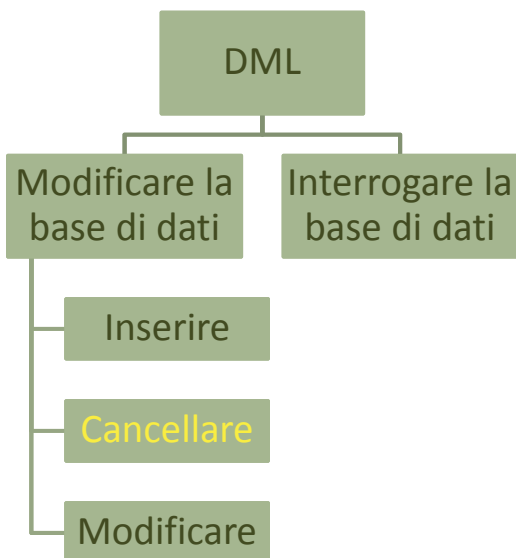
# INTRODUZIONE



# INTRODUZIONE

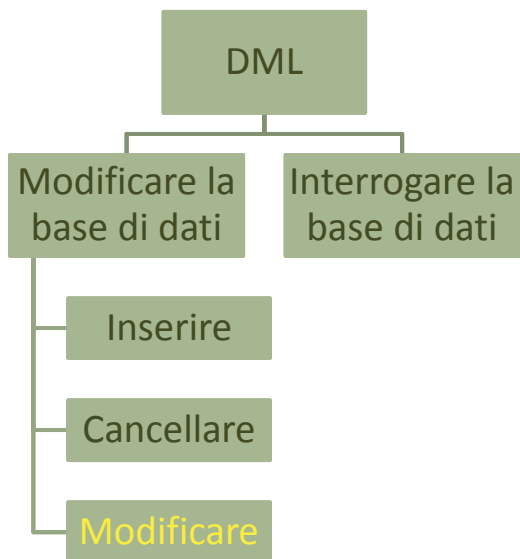


# INTRODUZIONE





# INTRODUZIONE



Significa inserire una riga in una tabella

Inserire dei dati

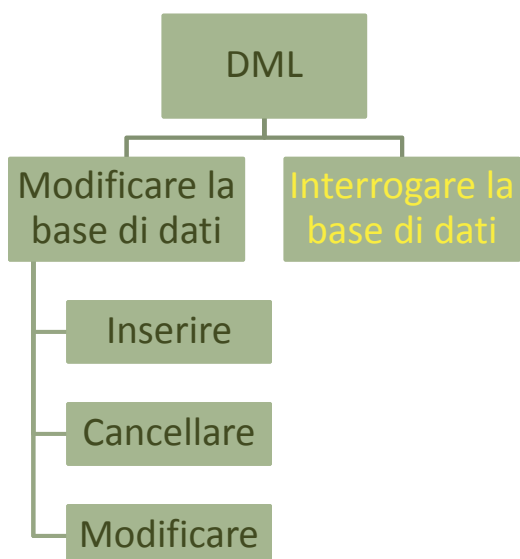
Significa cancellare da una tabella tutte le righe che soddisfano una certa condizione

Cancellare dei dati

Significa modificare in una certa tabella tutte le righe che soddisfano una certa condizione

Modificare dei dati

# INTRODUZIONE



Significa inserire una riga in una tabella

Inserire dei dati

Significa cancellare da una tabella tutte le righe che soddisfano una certa condizione

Cancellare dei dati

Significa modificare in una certa tabella tutte le righe che soddisfano una certa condizione

Modificare dei dati

Significa selezionare tutti i dati che soddisfano una certa condizione

Selezionare dei dati

# INTRODUZIONE

Vediamo subito come **inserire** i dati

Poi passiamo a vedere come **selezionare** i dati

Torniamo poi indietro per vedere come **cancellare e modificare** dei dati selezionati

Significa  
inserire una riga in una tabella

Inserire dei dati

Significa cancellare da una tabella  
tutte le righe che soddisfano una certa condizione

Cancellare dei dati

Significa modificare in una certa tabella  
tutte le righe che soddisfano una certa condizione

Modificare dei dati

Significa selezionare tutti i dati  
che soddisfano una certa condizione

Selezionare dei dati


## SCHEMA FISICO


Comandi MYSQL per manipolare i dati:  
modificare i dati - inserimento

# POPOLARE LE TABELLE


L'istruzione per **inserire** i dati in una tabella è molto semplice 

```
insert into <nomeTabella> (<nomeCampo1>, <nomeCampo2>, ... ,<nomeCampoN>)  
value (<valore1>, <valore2>, ... ,<valoreN>);
```


L'elenco dei campi può essere omissso   
se si indica un valore per **tutti** i campi della tabella nell'ordine in cui sono stati inseriti.

- È possibile **evitare** di specificare un valore 
- per i campi per i quali è stato specificato un **valore di default**
  - per i campi per i quali è stato specificato il costrutto **auto-increment**

# POPOLARE LE TABELLE

L'istruzione per **inserire** i dati in una tabella è molto semplice 

```
insert into <nomeTabella> (<nomeCampo1>, <nomeCampo2>, ... ,<nomeCampoN>)  
values  
(<valore1>, <valore2>, ... ,<valoreN>),  
(<valore1>, <valore2>, ... ,<valoreN>),  
...  
(<valore1>, <valore2>, ... ,<valoreN>);
```

Se vogliamo inserire più tuple modifichiamo **value** in **values**. 

# SCHEMA FISICO

Comandi MYSQL per manipolare i dati:  
interrogare la base di dati (struttura della query)

## Proiezione (taglio verticale)

Posso scegliere  
solo alcune **colonne**.



```
select <nomeCampo1>, <nomeCampo2>, ... <nomeCampoN>  
from <nomeTabella>;
```

```
select sigla, nome  
from Partiti;
```

Partiti				
id	sigla	nome	fondazione	scioglimento
325	PLI	Partito Liberale Italiano	08/10/1922	06/02/1994
326	DC	Democrazia Cristiana	15/12/1942	29/01/1994
329	PSDI	Partito Social Democratico Italiano	11/01/1947	01/01/1998
380	PSI	Partito Socialista Italiano	14/08/1892	12/11/1994
489	U	Ulivo	01/01/1995	01/01/1997
501	DS	Democratici di Sinistra	01/01/1998	

# Proiezione (taglio verticale)

Posso scegliere solo alcune **colonne**.



```
select <nomeCampo1>, <nomeCampo2>, ... <nomeCampoN>
from <nomeTabella>;
```

```
select sigla, nome
from Partiti;
```

sigla	nome
PLI	Partito Liberale Italiano
DC	Democrazia Cristiana
PSDI	Partito Social Democratico Italiano
PSI	Partito Socialista Italiano
U	Ulivo
DS	Democratici di Sinistra

Partiti				
id	sigla	nome	fondazione	scioglimento
325	PLI	Partito Liberale Italiano	08/10/1922	06/02/1994
326	DC	Democrazia Cristiana	15/12/1942	29/01/1994
329	PSDI	Partito Social Democratico Italiano	11/01/1947	01/01/1998
380	PSI	Partito Socialista Italiano	14/08/1892	12/11/1994
489	U	Ulivo	01/01/1995	01/01/1997
501	DS	Democratici di Sinistra	01/01/1998	

# Selezione (taglio orizzontale)

Posso scegliere solo alcune **righe**.



```
select <nomeCampo1>, <nomeCampo2>, ... <nomeCampoN>
from <nomeTabella>
where (<cond>);
```

```
select *
from Presidenti
where (percentuale > 70);
```

Presidenti					
id	nominativo	percentuale	dal	al	idPartiti
122	Enrico De Nicola	72,8	01/07/1946	12/05/1948	325
125	Luigi Einaudi	59,4	12/05/1948	11/05/1955	325
131	Giovanni Gronchi	74,5	11/05/1955	11/05/1962	326
132	Antonio Segni	52,6	11/05/1962	06/12/1964	326
135	Giuseppe Saragat	68,9	29/12/1964	29/12/1971	329
136	Giovanni Leone	52	29/12/1971	15/06/1978	326
145	Alessandro Pertini	83,6	09/07/1978	29/06/1985	380
155	Francesco Cossiga	75,4	03/07/1985	28/04/1992	326
156	Oscar Luigi Scalfaro	66,3	28/05/1992	15/05/1999	326
187	Carlo Azeglio Ciampi	71,4	18/05/1999	15/05/2006	null
221	Giorgio Napolitano	54,8	15/05/2006	14/01/2015	501
238	Sergio Mattarella	65,9	3/02/2015		null

# Selezione (taglio orizzontale)

Posso scegliere solo alcune **righe**.



```
select <nomeCampo1>, <nomeCampo2>, ... <nomeCampoN>
from <nomeTabella>
where (<cond>);
```

```
select *
from Presidenti
where (percentuale > 70);
```

Presidenti					
id	nominativo	percentuale	dal	al	idPartiti
122	Enrico De Nicola	72,8	01/07/1946	12/05/1948	325
125	Luigi Einaudi	59,4	12/05/1948	11/05/1955	325
131	Giovanni Gronchi	74,5	11/05/1955	11/05/1962	326
132	Antonio Segni	52,6	11/05/1962	06/12/1964	326
135	Giuseppe Saragat	68,9	29/12/1964	29/12/1971	329
136	Giovanni Leone	52	29/12/1971	15/06/1978	326
145	Alessandro Pertini	83,6	09/07/1978	29/06/1985	380
155	Francesco Cossiga	75,4	03/07/1985	28/04/1992	326
156	Oscar Luigi Scalfaro	66,3	28/05/1992	15/05/1999	326
187	Carlo Azeglio Ciampi	71,4	18/05/1999	15/05/2006	null
221	Giorgio Napolitano	54,8	15/05/2006		501
238	Sergio Mattarella	65,9	3/02/2015		null

# Selezione (taglio orizzontale)

id	nominativo	percentuale	dal	al	idPartiti
122	Enrico De Nicola	72,8	01/07/1946	12/05/1948	325
131	Giovanni Gronchi	74,5	11/05/1955	11/05/1962	326
145	Alessandro Pertini	83,6	09/07/1978	29/06/1985	380
155	Francesco Cossiga	75,4	03/07/1985	28/04/1992	326
187	Carlo Azeglio Ciampi	71,4	18/05/1999	15/05/2006	null

```
select *
from Presidenti
where (percentuale > 70);
```

Presidenti					
id	nominativo	percentuale	dal	al	idPartiti
122	Enrico De Nicola	72,8	01/07/1946	12/05/1948	325
125	Luigi Einaudi	59,4	12/05/1948	11/05/1955	325
131	Giovanni Gronchi	74,5	11/05/1955	11/05/1962	326
132	Antonio Segni	52,6	11/05/1962	06/12/1964	326
135	Giuseppe Saragat	68,9	29/12/1964	29/12/1971	329
136	Giovanni Leone	52	29/12/1971	15/06/1978	326
145	Alessandro Pertini	83,6	09/07/1978	29/06/1985	380
155	Francesco Cossiga	75,4	03/07/1985	28/04/1992	326
156	Oscar Luigi Scalfaro	66,3	28/05/1992	15/05/1999	326
187	Carlo Azeglio Ciampi	71,4	18/05/1999	15/05/2006	null
221	Giorgio Napolitano	54,8	15/05/2006		501
238	Sergio Mattarella	65,9	3/02/2015		null

# Prodotto Cartesiano

Il prodotto cartesiano di due tabelle è la combinazione di **ogni** riga della prima per **ogni** riga della seconda.

PRODOTTO CARTESIANO

Il prodotto Cartesiano è un risultato **intermedio** sul quale fare una operazione di selezione.

Partiti		
id	sigla	nome
325	PLI	Partito Liberale Italiano
326	DC	Democrazia Cristiana
380	PSI	Partito Socialista Italiano
501	DS	Democratici di Sinistra

Presidenti		
id	nominativo	idPartiti
155	Francesco Cossiga	326
156	Oscar Luigi Scalfaro	326
187	Carlo Azeglio Ciampi	null
221	Giorgio Napolitano	501

```
select Partiti.nome, Presidenti.nominativo
from Partiti, Presidenti
where (Partiti.id = Presidenti.idPartiti);
```

# Prodotto Cartesiano

Partiti X Presidenti					
id	sigla	nome	id	nominativo	idPartiti
325	PLI	Partito Liberale Italiano	155	Francesco Cossiga	326
325	PLI	Partito Liberale Italiano	156	Oscar Luigi Scalfaro	326
325	PLI	Partito Liberale Italiano	187	Carlo Azeglio Ciampi	null
325	PLI	Partito Liberale Italiano	221	Giorgio Napolitano	501
326	DC	Democrazia Cristiana	155	Francesco Cossiga	326
326	DC	Democrazia Cristiana	156	Oscar Luigi Scalfaro	326
326	DC	Democrazia Cristiana	187	Carlo Azeglio Ciampi	null
326	DC	Democrazia Cristiana	221	Giorgio Napolitano	501
380	PSI	Partito Socialista Italiano	155	Francesco Cossiga	326
380	PSI	Partito Socialista Italiano	156	Oscar Luigi Scalfaro	326
380	PSI	Partito Socialista Italiano	187	Carlo Azeglio Ciampi	null
380	PSI	Partito Socialista Italiano	221	Giorgio Napolitano	501
501	DS	Democratici di Sinistra	155	Francesco Cossiga	326
501	DS	Democratici di Sinistra	156	Oscar Luigi Scalfaro	326
501	DS	Democratici di Sinistra	187	Carlo Azeglio Ciampi	null
501	DS	Democratici di Sinistra	221	Giorgio Napolitano	501

Partiti		
id	sigla	nome
325	PLI	Partito Liberale Italiano
326	DC	Democrazia Cristiana
380	PSI	Partito Socialista Italiano
501	DS	Democratici di Sinistra

Presidenti		
id	nominativo	idPartiti
155	Francesco Cossiga	326
156	Oscar Luigi Scalfaro	326
187	Carlo Azeglio Ciampi	null
221	Giorgio Napolitano	501

```
select Partiti.nome, Presidenti.nominativo
from Partiti, Presidenti
where (Partiti.id = Presidenti.idPartiti);
```

# Prodotto Cartesiano

Partiti X Presidenti					
id	sigla	nome	id	nominativo	idPartiti
325	PLI	Partito Liberale Italiano	155	Francesco Cossiga	326
325	PLI	Partito Liberale Italiano	156	Oscar Luigi Scalfaro	326
325	PLI	Partito Liberale Italiano	187	Carlo Azeglio Ciampi	null
325	PLI	Partito Liberale Italiano	221	Giorgio Napolitano	501
326	DC	Democrazia Cristiana	155	Francesco Cossiga	326
326	DC	Democrazia Cristiana	156	Oscar Luigi Scalfaro	326
326	DC	Democrazia Cristiana	187	Carlo Azeglio Ciampi	null
326	DC	Democrazia Cristiana	221	Giorgio Napolitano	501
380	PSI	Partito Socialista Italiano	155	Francesco Cossiga	326
380	PSI	Partito Socialista Italiano	156	Oscar Luigi Scalfaro	326
380	PSI	Partito Socialista Italiano	187	Carlo Azeglio Ciampi	null
380	PSI	Partito Socialista Italiano	221	Giorgio Napolitano	501
501	DS	Democratici di Sinistra	155	Francesco Cossiga	326
501	DS	Democratici di Sinistra	156	Oscar Luigi Scalfaro	326
501	DS	Democratici di Sinistra	187	Carlo Azeglio Ciampi	null
501	DS	Democratici di Sinistra	221	Giorgio Napolitano	501

Partiti		
id	sigla	nome
325	PLI	Partito Liberale Italiano
326	DC	Democrazia Cristiana
380	PSI	Partito Socialista Italiano
501	DS	Democratici di Sinistra

Presidenti		
id	nominativo	idPartiti
155	Francesco Cossiga	326
156	Oscar Luigi Scalfaro	326
187	Carlo Azeglio Ciampi	null
221	Giorgio Napolitano	501

```
select Partiti.nome, Presidenti.nominativo
  from Partiti, Presidenti
 where (Partiti.id = Presidenti.idPartiti);
```

# Prodotto Cartesiano

Partiti X Presidenti					
id	sigla	nome	id	nominativo	idPartiti
325	PLI	Partito Liberale Italiano	155	Francesco Cossiga	326
325	PLI	Partito Liberale Italiano	156	Oscar Luigi Scalfaro	326
325	PLI	Partito Liberale Italiano	187	Carlo Azeglio Ciampi	null
325	PLI	Partito Liberale Italiano	221	Giorgio Napolitano	501
326	DC	Democrazia Cristiana	155	Francesco Cossiga	326
326	DC	Democrazia Cristiana	156	Oscar Luigi Scalfaro	326
326	DC	Democrazia Cristiana	187	Carlo Azeglio Ciampi	null
326	DC	Democrazia Cristiana	221	Giorgio Napolitano	501
380	PSI	Partito Socialista Italiano	155	Francesco Cossiga	326
380	PSI	Partito Socialista Italiano	156	Oscar Luigi Scalfaro	326
380	PSI	Partito Socialista Italiano	187	Carlo Azeglio Ciampi	null
380	PSI	Partito Socialista Italiano	221	Giorgio Napolitano	501
501	DS	Democratici di Sinistra	155	Francesco Cossiga	326
501	DS	Democratici di Sinistra	156	Oscar Luigi Scalfaro	326
501	DS	Democratici di Sinistra	187	Carlo Azeglio Ciampi	null
501	DS	Democratici di Sinistra	221	Giorgio Napolitano	501

nome	nominativo
Democrazia Cristiana	Francesco Cossiga
Democrazia Cristiana	Oscar Luigi Scalfaro
Democratici di Sinistra	Giorgio Napolitano

```
select Partiti.nome, Presidenti.nominativo
  from Partiti, Presidenti
 where (Partiti.id = Presidenti.idPartiti);
```



# Inner Join

Una selezione come questa viene detta prodotto cartesiano condizionato (**join**).

Nello specifico questa è una **equi-join** (simbolo di uguaglianza).

**MySql** accetta anche un altro tipo di sintassi per una **join**.

```
select Partiti.nome, Presidenti.nominativo
  from Partiti join Presidenti on Partiti.id = Presidenti.idPartiti;
```

Questa sintassi ci consente - se i due campi hanno **lo stesso nome** - la **natural join**.

```
select Partiti.nome, Presidenti.nominativo
  from Partiti natural join Presidenti;
```

```
select Partiti.nome, Presidenti.nominativo
  from Partiti, Presidenti
  where (Partiti.id = Presidenti.idPartiti);
```

# Inner Join e Outer Join

**Equi-join** (condizione di uguaglianza), **join** (altra condizione) e **natural join** (stesso nome su entrambe le tabelle) sono tutte esempi di **inner join**. Ovvero il risultato è composto **solo** dalle righe che soddisfano la condizione.

INNER JOIN

Nel caso in cui volessimo **tutte** le righe della **prima** tabella (anche quelle che non soddisfano la condizione) dovremmo realizzare una **left join**. Altrimenti una **right join**. In entrambi i casi parliamo di **outer join**.

OUTER JOIN

# Left Join

Partiti		
id	sigla	nome
325	PLI	Partito Liberale Italiano
326	DC	Democrazia Cristiana
380	PSI	Partito Socialista Italiano
501	DS	Democratici di Sinistra

Presidenti		
id	nominativo	idPartiti
155	Francesco Cossiga	326
156	Oscar Luigi Scalfaro	326
187	Carlo Azeglio Ciampi	null
221	Giorgio Napolitano	501

Supponiamo di volere l'elenco di tutti i partiti con accanto tutti i nomi di presidenti di quel partito compresi i partiti che non sono mai stati rappresentati da alcun presidente.



nome	nominativo
Partito Liberale Italiano	
Democrazia Cristiana	Francesco Cossiga
Democrazia Cristiana	Oscar Luigi Scalfaro
Partito Socialista Italiano	
Democratici di Sinistra	Giorgio Napolitano

```
select Partiti.nome, Presidenti.nominativo
from Partiti left join Presidenti on Partiti.id = Presidenti.idPartiti;
```

# Right Join

Partiti		
id	sigla	nome
325	PLI	Partito Liberale Italiano
326	DC	Democrazia Cristiana
380	PSI	Partito Socialista Italiano
501	DS	Democratici di Sinistra

Presidenti		
id	nominativo	idPartiti
155	Francesco Cossiga	326
156	Oscar Luigi Scalfaro	326
187	Carlo Azeglio Ciampi	null
221	Giorgio Napolitano	501

Supponiamo adesso di volere l'elenco di tutti i presidenti e per ciascuno di essi il nome del partito di riferimento quando c'è.



nome	nominativo
Democrazia Cristiana	Francesco Cossiga
Democrazia Cristiana	Oscar Luigi Scalfaro
	Carlo Azeglio Ciampi
Democratici di Sinistra	Giorgio Napolitano

```
select Partiti.nome, Presidenti.nominativo
from Partiti right join Presidenti on Partiti.id = Presidenti.idPartiti;
```

# IL PREDICATO AS

In certi casi può essere utile **rinominare** una colonna o una tabella: basta far seguire il nome della colonna o il nome della tabella dal costrutto **as** seguito dal nuovo nome.

AS

```
select Partiti.nome as Partito, Presidenti.nominativo as Presidente
from Partiti join Presidenti on Partiti.id = Presidenti.idPartiti;
```

```
select x.nome, y.nominativo
from Partiti as x join Presidenti as y on x.id = y.idPartiti;
```

# SELF JOIN

Quando nello schema ER c'è una relazione **unaria** abbiamo una tabella come quella qui di seguito riportata, qui ha senso parlare di **self-join**.



Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

```
select d1.cognome, d1.nome, d2.cognome as dirigente
from Dipendenti as d1 join Dipendenti as d2 on d1.idDirigente = d2.id;
```

cognome	nome	dirigente
Rossi	Lorenzo	Russo
Russo	Gabriele	Marino
Ferrari	Mattia	Marino
Esposito	Riccardo	Marino
Bianchi	Davide	Russo
Romano	Luca	Russo
Colombo	Marco	Marino
Ricci	Simone	Marino
Greco	Giuseppe	Russo

# SELF JOIN

Quando nello schema ER c'è una relazione **unaria** abbiamo una tabella come quella qui di seguito riportata, qui ha senso parlare di **self-join**.

Anche una **self join** può essere **left** o **right**.

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

```
select d1.cognome, d1.nome, d2.cognome as dirigente
from Dipendenti as d1 left join Dipendenti as d2 on d1.idDirigente = d2.id;
```

cognome	nome	dirigente
Rossi	Lorenzo	Russo
Russo	Gabriele	Marino
Ferrari	Mattia	Marino
Esposito	Riccardo	Marino
Bianchi	Davide	Russo
Romano	Luca	Russo
Colombo	Marco	Marino
Ricci	Simone	Marino
Marino	Leonardo	
Greco	Giuseppe	Russo

## SCHEMA FISICO

Comandi MYSQL per manipolare i dati:  
interrogare la base di dati (seconda parte)

# LA CLAUSOLA ORDER BY

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

```
select *
  from Dipendenti
 order by Cognome;
```

Posso specificare **asc** (ascendente) e **desc** (discendente) dopo ogni nome di colonna.



id	cognome	nome	idDirigente
4690	Bianchi	Davide	4556
4712	Colombo	Marco	4789
4689	Esposito	Riccardo	4789
4558	Ferrari	Mattia	4789
4853	Greco	Giuseppe	4556
4789	Marino	Leonardo	
4745	Ricci	Simone	4789
4691	Romano	Luca	4556
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789

# IL PREDICATO AS

In certi casi può essere utile **rinominare** una colonna o una tabella: basta far seguire il nome della colonna o il nome della tabella dal costrutto **as** seguito dal nuovo nome.

AS

```
select Partiti.nome as Partito, Presidenti.nominativo as Presidente
  from Partiti join Presidenti on Partiti.id = Presidenti.idPartiti;
```

```
select x.sigla, y.nominativo
  from Partiti as x join Presidenti as y on x.id = y.idPartiti;
```

# IL PREDICATO IS NULL

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

```
select *
  from Dipendenti
 where idDirigente is null;
```

id	cognome	nome	idDirigente
4789	Marino	Leonardo	

# IL PREDICATO IS NOT NULL

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

```
select *
  from Dipendenti
 where idDirigente is not null;
```

id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4853	Greco	Giuseppe	4556

# IL PREDICATO DISTINCT

In certi casi può essere utile descrivere se il risultato atteso deve ammettere o meno i duplicati.

**DISTINCT**

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

```
select distinct localita
from Alunni;
```

localita
Civezzano
Pergine V.
Zivignago
Canezza
Novaledo

# IL PREDICATO IN

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

```
select *
from Alunni
where localita in ('Civezzano', 'Novaledo');
```

id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN

# IL PREDICATO NOT IN

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

```
select *
  from Alunni
 where localita not in ('Pergine V.' , 'Zivignago' , 'Canezza');
```

id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN

# IL PREDICATO BETWEEN

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

```
select *
  from Alunni
 where civico between 100 and 1000;
```

id	cognome	nome	via	civico	localita	prov
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN



# IL PREDICATO LIKE

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

```
select *
  from Alunni
 where via like 'Vi%';
```

id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN

# UNIONE

ParentiDiLui						
id	cognome	nome	via	civico	localita	prov
4521	Ferrari	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Ferrari	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Ferrari	Riccardo	Via Roma	489	Civezzano	TN
4690	Ferrari	Davide	Piazza Vesuvio	47	Pergine V.	TN

ParentiDiLei						
id	cognome	nome	via	civico	localita	prov
221	Marino	Luca	Corso Magellano	78	Civezzano	TN
246	Marino	Marco	Vicolo Corto	12	Pergine V.	TN
245	Marino	Simone	Via Marco Polo	2	Civezzano	TN
248	Marino	Leonardo	Parco della Vittoria	489	Civezzano	TN
240	Marino	Giuseppe	Corso Magellano	47	Pergine V.	TN

```
select cognome, nome
  from ParentiDiLui
union
select cognome, nome
  from ParentiDiLei;
```

cognome	nome
Ferrari	Lorenzo
Ferrari	Gabriele
Ferrari	Mattia
Ferrari	Riccardo
Ferrari	Davide
Marino	Luca
Marino	Marco
Marino	Simone
Marino	Leonardo
Marino	Giuseppe



# SCHEMA FISICO

Comandi MYSQL per manipolare i dati:  
interrogare la base di dati (terza parte)

## FUNZIONI DI AGGREGAZIONE

Ad ogni selezione posso  
abbinare dei valori calcolati.

- sum
- avg
- min
- max
- count

FUNZIONI DI AGGREGAZIONE

# SUM

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select sum (percentuale) as valoreInutile  
from Presidenti;
```

<b>valoreInutile</b>
----------------------

731.7
-------

# AVG

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select avg (percentuale) as media  
from Presidenti;
```

<b>media</b>
--------------

66.5
------

# MIN

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select min (percentuale) as minimo  
from Presidenti;
```

<b>minimo</b>
52

# MAX

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select max (percentuale) as massimo  
from Presidenti;
```

<b>massimo</b>
83.6

# COUNT

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select count(percentuale) as totale
from Presidenti;
```

totale
11

# FUNZIONI DI AGGREGAZIONE

Ad ogni selezione posso abbinare dei valori calcolati.

- sum
- avg
- min
- max
- count

FUNZIONI DI AGGREGAZIONE

Usando le funzioni di aggregazione può avere senso desiderare dei risultati per **gruppi** di righe.



# RAGGRUPPAMENTI

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select partito, COUNT(nominativo) as 'tot presidenti'
from Presidenti
group by partito;
```


partito	tot presidenti
Partito Liberale Italiano	2
Democrazia Cristiana	5
Partito Social Democratico Italiano	1
Partito Socialista Italiano	1
Ulivo	1
Democratici di Sinistra	1


# FUNZIONI DI AGGREGAZIONE

Ad ogni selezione posso abbinare dei valori calcolati.

- sum
- avg
- min
- max
- count

## FUNZIONI DI AGGREGAZIONE

Usando le funzioni di aggregazione può avere senso desiderare dei risultati per **gruppi** di righe. 

Una volta ottenuti i risultati può essere opportuno presentare solo quelli che rispettano determinate **condizioni**. 

# RAGGRUPPAMENTI

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select partito, COUNT(nominativo) as 'tot presidenti'
from Presidenti
group by partito
having COUNT(nominativo) > 1;
```

partito	tot presidenti
Partito Liberale Italiano	2
Democrazia Cristiana	5

# QUERY NIDIFICATE

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select avg (percentuale) as media
from Presidenti;
```

media
66.5

È possibile utilizzare il risultato di una query all'interno di un'altra query.



# QUERY NIDIFICATE

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select avg (percentuale) as media
from Presidenti;
```

<b>media</b>
--------------

66.5
------

È possibile utilizzare il risultato di una query all'interno di un'altra query.



# QUERY NIDIFICATE

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
select avg (percentuale) as media
from Presidenti;
```

<b>media</b>
--------------

66.5
------

```
select *
from Presidenti
where (percentuale >
      (select avg (percentuale) as media
       from Presidenti)
      )
```

Quando la query nidificata restituisce un insieme di valori possiamo usare i costrutti:

- In
- Not in
- All
- Some (o any)





# SCHEMA FISICO

Comandi MYSQL per manipolare i dati:  
cancellare e modificare i dati

## QUERY DI AGGIORNAMENTO

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
update Presidenti
set partito = 'Indipendente (area Ulivo)'
where (id = 187);
```

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Indipendente (area Ulivo)
221	Giorgio Napolitano	54,8	Democratici di Sinistra

# QUERY DI ELIMINAZIONE

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

```
delete from Presidenti
where (partito = 'Democrazia Cristiana');
```

Presidenti			
id	nominativo	percentuale	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
145	Alessandro Pertini	83,6	Partito Socialista Italiano
187	Carlo Azeglio Ciampi	71,4	Indipendente (area Ulivo)
221	Giorgio Napolitano	54,8	Democratici di Sinistra

## SCHEMA FISICO

Comandi MYSQL per manipolare i dati:  
strumenti avanzati del DML

# INTERROGAZIONI AVANZATE

In alcuni casi gli strumenti che abbiamo mostrato non bastano. Mostriamo qui di seguito tre strumenti avanzati.

In certi casi è utile usare una variabile in una interrogazione.

VARIABILI

## USO DI VARIABILI

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

num	cognome	nome
1	Bianchi	Davide
2	Colombo	Marco
3	Esposito	Riccardo
4	Ferrari	Mattia
5	Greco	Giuseppe
6	Marino	Leonardo
7	Ricci	Simone
8	Romano	Luca
9	Rossi	Lorenzo
10	Russo	Gabriele

Se volessimo ordinare i dipendenti per cognome e creare un numero ordinale per ciascuna riga, dovremmo usare una variabile.

```
set @row := 0;
select @row := @row+1 as num, d.cognome, d.nome
from Dipendenti as d
order by d.cognome;
```

# INTERROGAZIONI AVANZATE

In alcuni casi gli strumenti che abbiamo mostrato non bastano. Mostreremo qui di seguito tre strumenti avanzati.

In certi casi è utile usare una variabile in una interrogazione.

VARIABILI

In certi casi è utile usare una tabella temporanea.

TABELLE TEMPORANEE

# TABELLE TEMPORANEE

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

num	cognome	nome
1	Bianchi	Davide
2	Colombo	Marco
3	Esposito	Riccardo
4	Ferrari	Mattia
5	Greco	Giuseppe
6	Marino	Leonardo
7	Ricci	Simone
8	Romano	Luca
9	Rossi	Lorenzo
10	Russo	Gabriele

Se volessimo solo i primi 5 risultati dovremmo aggiungere la **clausola where** alla query precedente, ma darebbe errore.

```
set @row := 0;
```

```
create temporary table temp
select @row := @row+1 as num, d.cognome, d.nome
from Dipendenti as d
order by d.cognome;
```

```
select *
from temp
where (num <= 5);
```

# TABELLE TEMPORANEE

Dipendenti			
id	cognome	nome	idDirigente
4521	Rossi	Lorenzo	4556
4556	Russo	Gabriele	4789
4558	Ferrari	Mattia	4789
4689	Esposito	Riccardo	4789
4690	Bianchi	Davide	4556
4691	Romano	Luca	4556
4712	Colombo	Marco	4789
4745	Ricci	Simone	4789
4789	Marino	Leonardo	
4853	Greco	Giuseppe	4556

num	cognome	nome
1	Bianchi	Davide
2	Colombo	Marco
3	Esposito	Riccardo
4	Ferrari	Mattia
5	Greco	Giuseppe
6	Marino	Leonardo
7	Ricci	Simone
8	Romano	Luca
9	Rossi	Lorenzo
10	Russo	Gabriele

Se volessimo solo i primi 5 risultati dovremmo aggiungere la **clausola where** alla query precedente, ma darebbe errore.

```
set @row := 0;
```

```
select *
  from
  (
    select @row := @row+1 as num,
           d.cognome, d.nome
    from Dipendenti as d
    order by d.cognome;
  )
 where (num <= 5);
```

Le tabelle temporanee sono un'alternativa all'uso di query annidate.

# INTERROGAZIONI AVANZATE

In alcuni casi gli strumenti che abbiamo mostrato non bastano. Mostreremo qui di seguito tre strumenti avanzati.

In certi casi è utile usare una variabile in una interrogazione.

VARIABILI

In certi casi è utile usare una tabella temporanea.

TABELLE TEMPORANEE

In certi casi sono utili delle funzioni che manipolano i dati.

FUNZIONI

# FUNZIONI

Presidenti			
id	nominativo	perc	partito
122	Enrico De Nicola	72,8	Partito Liberale Italiano
125	Luigi Einaudi	59,4	Partito Liberale Italiano
131	Giovanni Gronchi	74,5	Democrazia Cristiana
132	Antonio Segni	52,6	Democrazia Cristiana
135	Giuseppe Saragat	68,9	Partito Social Democratico Italiano
136	Giovanni Leone	52	Democrazia Cristiana
145	Alessandro Pertini	83,6	Partito Socialista Italiano
155	Francesco Cossiga	75,4	Democrazia Cristiana
156	Oscar Luigi Scalfaro	66,3	Democrazia Cristiana
187	Carlo Azeglio Ciampi	71,4	Ulivo
221	Giorgio Napolitano	54,8	Democratici di Sinistra

Supponiamo di voler presentare i dati di questa tabella con i valori percentuali arrotondati all'intero più vicino.

```
select nominativo, round(perc, 0) as perc
from Presidenti
```

nominativo	perc
Enrico De Nicola	73
Luigi Einaudi	59
Giovanni Gronchi	75
Antonio Segni	53
Giuseppe Saragat	69
Giovanni Leone	52
Alessandro Pertini	84
Francesco Cossiga	75
Oscar Luigi Scalfaro	66
Carlo Azeglio Ciampi	71
Giorgio Napolitano	55

# FUNZIONI

FUNZIONE	COSA RESTITUISCE
least(a, b, c, ..)	il parametro minore
greatest(a, b, c, ..)	il parametro maggiore
mod(a,b)	il resto di a diviso b
floor(a)	a arrotondato per difetto
ceiling(a)	a arrotondato per eccesso
round(a,b)	a arrotondato a b cifre decimali
exp(a)	l'esponenziale di a
log(a)	il logaritmo naturale di a
rand()	un num casuale tra 0 e 0.999

# FUNZIONI



FUNZIONE	COSA RESTITUISCE
concat(a,b,c, ..)	la concatenazione di a,b,c,..
trim(s)	la stringa data senza spazi
ltrim(s)	la stringa data senza spazi a sn
rtrim(s)	la stringa data senza spazi a ds
substring(s,x)	s meno i suoi x caratteri iniziali
mid(s,x,k)	x caratteri di s a partire dal k-esimo

# FUNZIONI



FUNZIONE	COSA RESTITUISCE
now()	data e ora corrente
year(d)	l'anno della data d
month(d)	il mese della data d
day(d)	il giorno della data d
hour(d)	l'ora della data d
min(d)	i minuti della data d
sec(d)	i secondi della data d
datediff(a,b)	i giorni che separano la data a dalla data b

# SCHEMA FISICO

DDL Avanzato: gli indici

## IL CONCETTO DI INDICE

Ospite						
id	nome	cognome	via	civico	numTel	idLocalità
12	Giorgio	Feola	Modena	82	3476644551	4
13	Beatrice	Feola	Roma	182	3201597532	6
14	Michela	Zancanella	Genova	4	3255666541	5
15	Stefano	De Santis	Machiavelli	40	3957698541	8
16	Raffaele	Cennamo	Milano	1	0823547858	8

Occupa					
id	dal	al	lettiAggiuntivi	idOspite	idStanza
24	10/08/2010	20/08/2010	1	14	20
25	15/08/2010	30/08/2010		16	23
26	29/07/2011	10/08/2011		16	22
27	12/08/2011	18/08/2011	1	14	17
28	20/08/2011	5/09/2011	2	15	17
29	28/07/2012	13/08/2012		16	21
30	1/08/2012	15/08/2012		12	22

Stanza					
id	numero	metratura	piano	tipologia	maxLettiAggiuntivi
17	100	20	1	singola	2
18	101	30	1	matrimoniale	1
19	102	30	1	matrimoniale	2
20	200	20	2	singola	2
21	201	30	2	matrimoniale	0
22	202	30	2	matrimoniale	0
23	300	60	3	matrimoniale	0

Facciamo un'osservazione importante su una query tipo.

Supponiamo di avere queste tre tabelle e di voler mostrare nome e cognome di tutti gli ospiti che nel corso del tempo hanno occupato una stanza di dimensione superiore ai 25 mq.



# IL CONCETTO DI INDICE

Ospite						
id	nome	cognome	via	civico	numTel	idLocalità
12	Giorgio	Feola	Modena	82	3476644551	4
13	Beatrice	Feola	Roma	182	3201597532	6
14	Michela	Zancanella	Genova	4	3255666541	5
15	Stefano	De Santis	Machiavelli	40	3957698541	8
16	Raffaele	Cennamo	Milano	1	0823547858	8

Occupa					
id	dal	al	lettiAggiuntivi	idOspite	idStanza
24	10/08/2010	20/08/2010	1	14	20
25	15/08/2010	30/08/2010		16	23
26	29/07/2011	10/08/2011		16	22
27	12/08/2011	18/08/2011	1	14	17
28	20/08/2011	5/09/2011	2	15	17
29	28/07/2012	13/08/2012		16	21
30	1/08/2012	15/08/2012		12	22

Stanza					
id	numero	metratura	piano	tipologia	maxLettiAggiuntivi
17	100	20	1	singola	2
18	101	30	1	matrimoniale	1
19	102	30	1	matrimoniale	2
20	200	20	2	singola	2
21	201	30	2	matrimoniale	0
22	202	30	2	matrimoniale	0
23	300	60	3	matrimoniale	0

```
select distinct h.cognome, h.nome
from Ospite as h, Occupa as k, Stanza as s
where (h.id = k.idOspite) and (s.id = k.idStanza) and (s.metratura > 25)
```

Facciamo un'osservazione importante su una query tipo.

Supponiamo di avere queste tre tabelle e di voler mostrare nome e cognome di tutti gli ospiti che nel corso del tempo hanno occupato una stanza di dimensione superiore ai 25 mq.

# IL CONCETTO DI INDICE

Osserviamo che molto spesso nelle query passiamo da una tabella all'altra confrontando **chiavi esterne** con **chiavi primarie**.

Lo scenario tipo è il seguente:  
siamo su una tupla, leggiamo il **valore** della **chiave secondaria** ed andiamo a cercare questo valore nella colonna **chiave primaria** di un'altra tabella.

Per rendere la ricerca più veloce il DBMS tiene le tavole **ordinate** secondo il valore della **chiave primaria**.  
In questo modo può usare la **ricerca dicotomica**.

Il concetto di **indice** ci viene in aiuto per velocizzare le ricerche su **altri campi** che riteniamo vengano usati spesso per le ricerche.

```
select distinct h.cognome, h.nome
from Ospite as h, Occupa as k, Stanza as s
where (h.id = k.idOspite) and (s.id = k.idStanza) and (s.metratura > 25)
```

# IL CONCETTO DI INDICE

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

In questa tabella ragionevolmente faremo spesso ricerche per **cognome**



Rendiamo quindi il campo **cognome** un **campo indice**



Il DBMS produrrà quindi una **tabella interna** ordinata per cognome abbinando ad ogni cognome il suo id: in questo modo con **due ricerche dicotomiche** raggiungiamo la tupla desiderata



# IL CONCETTO DI INDICE

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

Index cognome (tabella interna non visibile all'utente)	
cognome	id
Bianchi	4690
Colombo	4712
Esposito	4689
Ferrari	4558
Greco	4853
Marino	4789
Ricci	4745
Romano	4691
Rossi	4521
Russo	4556

Ovviamente **mantenere** una tabella indice ha un **costo** notevole in termini di tempo (ma anche di spazio occupato) per cui è importante scegliere con cura **quanti e quali** campi indicizzare.



# IL CONCETTO DI INDICE

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

Index cognome (tabella interna non visibile all'utente)	
cognome	id
Bianchi	4690
Colombo	4712
Esposito	4689
Ferrari	4558
Greco	4853
Marino	4789
Ricci	4745
Romano	4691
Rossi	4521
Russo	4556

Va detto che il DBMS non crea nessun indice in modo automatico. Sta quindi al progettista valutare se e quando indicizzare una **chiave esterna**.



# IL CONCETTO DI INDICE

Alunni						
id	cognome	nome	via	civico	localita	prov
4521	Rossi	Lorenzo	Viale Traiano	78	Civezzano	TN
4556	Russo	Gabriele	Viale Gran Sasso	12	Pergine V.	TN
4558	Ferrari	Mattia	Viale Monte Rosa	2	Civezzano	TN
4689	Esposito	Riccardo	Via Roma	489	Civezzano	TN
4690	Bianchi	Davide	Piazza Vesuvio	47	Pergine V.	TN
4691	Romano	Luca	Largo Colombo	59	Zivignago	TN
4712	Colombo	Marco	Corso Magellano	321	Canezza	TN
4745	Ricci	Simone	Vicolo Corto	94	Canezza	TN
4789	Marino	Leonardo	Via Marco Polo	42	Novaledo	TN
4853	Greco	Giuseppe	Parco della Vittoria	48	Pergine V.	TN

Index cognome (tabella interna non visibile all'utente)	
cognome	id
Bianchi	4690
Colombo	4712
Esposito	4689
Ferrari	4558
Greco	4853
Marino	4789
Ricci	4745
Romano	4691
Rossi	4521
Russo	4556

```
create index <nomeIndice>
on <nomeTabella> (<nomeCampo>)
```

```
drop index <nomeIndice>
on <nomeTabella>
```

