

Informatica

Le basi di dati

Thomas Barbera
thomas.barbera@unimib.it

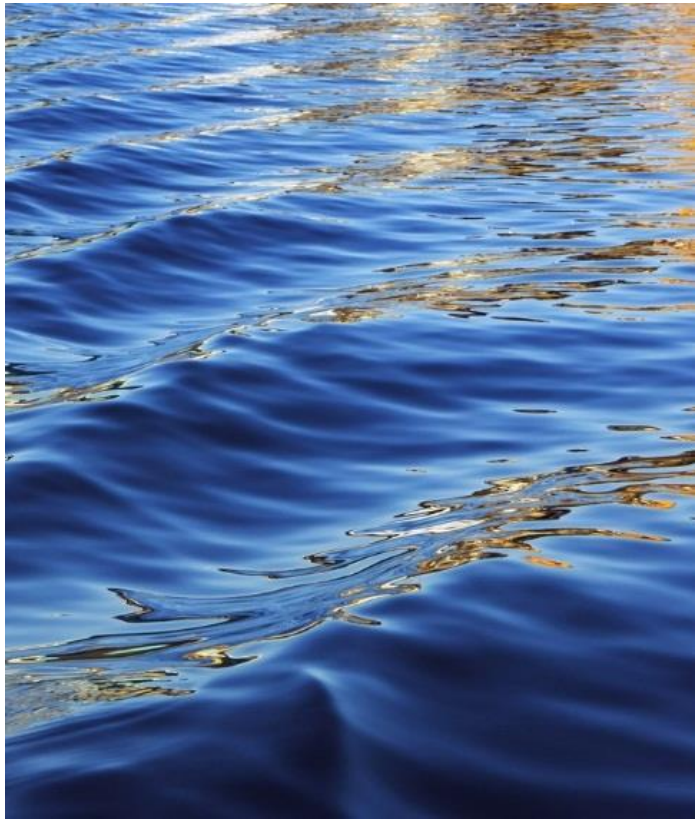
Riadattamento delle slides di Marco Viviani e Simone Zini







Seconda parte:
Il linguaggio SQL



Introduzione

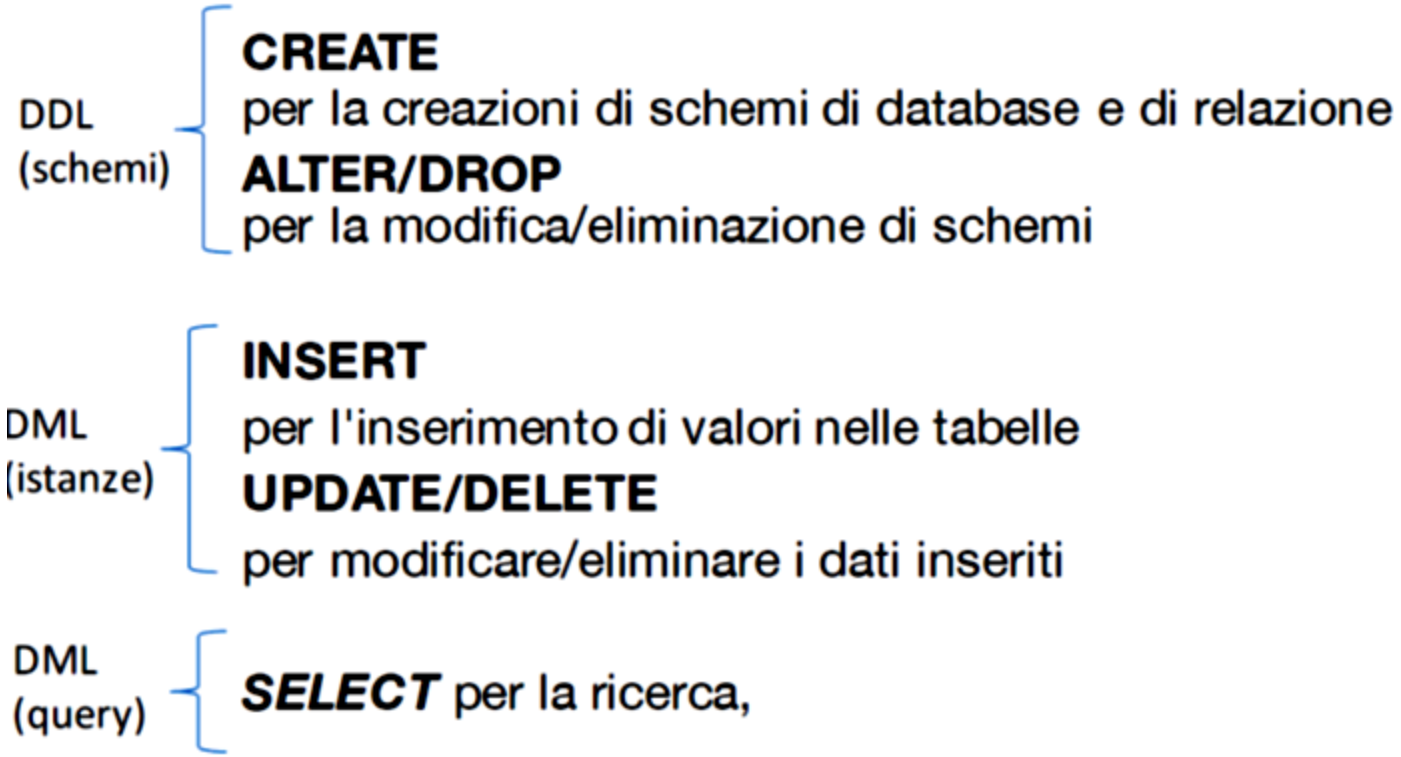
- **SQL**: *Structured Query Language*

- Linguaggio con varie funzionalità:
  - DDL – Data Definition Language
  - DQL – Data Query Language
  - DML – Data Manipulation Language
  - DCL – Data Control Language

- Permette dunque di:
  - Definire lo **schema** di un database e di una relazione
  - **Inserire valori** nelle relazioni
  - **Interrogare** il database (estrarre relazioni e valori)

Structured Query Language (SQL)

- I «verbi» utilizzati sono:



Notazione della sintassi

- Notazione della sintassi SQL che utilizzeremo per descrivere i comandi SQL
  - **TERMINI SPECIFI DEL LINGUAGGIO** in maiuscolo
  - **Variabili** in minuscolo
  - **[termini opzionali, non obbligatori]** tra parentesi quadre

- Esempio: estrarre da una tabella alcuni attributi che rispettino una certa condizione
  - **SELECT** attributi
  - **FROM** tabella
  - **[WHERE condizioni]**

Istruzioni SQL

- L'istruzione **CREATE DATABASE** crea un database con un dato nome
- Esempio, creazione del database World:

CREATE DATABASE World;

- L'istruzione **USE** comunica che le tabelle che verranno create verranno inserite in quel database

USE World;

Istruzioni SQL

- Istruzione **CREATE TABLE**:
  - definisce uno schema di relazione e ne crea un'istanza vuota specifica **attributi**, **domini** e **vincoli**

- Esempio:

```
CREATE TABLE Country(
Name VARCHAR(35) PRIMARY KEY,
Code VARCHAR(4),
Capital VARCHAR(35),
Province VARCHAR(35),
Area FLOAT CHECK (Area >= 0),
Population INT CHECK (Population >= 0)
);
```

COUNTRY

Name	Code	Capital	Province	Area	Population

Istruzioni SQL

```
CREATE TABLE Country(
Name VARCHAR(35) PRIMARY KEY,
Code VARCHAR(4),
Capital VARCHAR(35),
Province VARCHAR(35),
Area FLOAT CHECK (Area >= 0),
Population INT CHECK (Population >= 0)
);
```

Nome tabella

COUNTRY

Name	Code	Capital	Province	Area	Population

Istruzioni SQL

```
CREATE TABLE Country(
Name VARCHAR(35) PRIMARY KEY,
Code VARCHAR(4),
Capital VARCHAR(35),
Province VARCHAR(35),
Area FLOAT CHECK (Area >= 0),
Population INT CHECK (Population >= 0)
);
```

Nomi attributi

COUNTRY

Name	Code	Capital	Province	Area	Population

Istruzioni SQL

```
CREATE TABLE Country(
Name VARCHAR(35) PRIMARY KEY,
Code VARCHAR(4),
Capital VARCHAR(35),
Province VARCHAR(35),
Area FLOAT CHECK (Area >= 0),
Population INT CHECK (Population >= 0)
);
```

Vincolo di chiave primaria

COUNTRY

Name	Code	Capital	Province	Area	Population

Istruzioni SQL

```
CREATE TABLE Country(
Name VARCHAR(35) PRIMARY KEY,
Code VARCHAR(4),
Capital VARCHAR(35),
Province VARCHAR(35),
Area FLOAT CHECK (Area >= 0),
Population INT CHECK (Population >= 0)
);
```

Vincoli di integrità

COUNTRY

Name	Code	Capital	Province	Area	Population

Domini elementari

- **Carattere**: singoli caratteri o stringhe, anche di lunghezza variabile
- **Numerici**, esatti e approssimati
- **Data**, **ora**, **intervalli di tempo**
- Introdotti in SQL dal 1999:
  - **Boolean**
  - **BLOB**, **CLOB** (binary/character large object): per grandi immagini e testi

Domini definiti dall'utente

- Istruzione **CREATE DOMAIN**:
  - definisce un dominio (semplice), utilizzabile in definizioni di relazioni, anche con vincoli e valori di default
  - quando non specificato diversamente, il valore di default è NULL

- Esempio:

```
CREATE DOMAIN Voto
AS SMALLINT DEFAULT NULL
CHECK (value >=18 AND value <= 30) ;
```

Unique e chiave primaria

- **UNIQUE:** garantisce che tutti i valori di una colonna siano **differenti**
- **NOT NULL:** impone che un dato attributo **non possa assumere il valore nullo**. In genere l'attributo viene associato ad un valore di default, che viene assegnato automaticamente all'atto della creazione di una nuova tupla.
- **PRIMARY KEY:** garantisce che tutti i valori di una colonna siano differenti e non nulli
- Una tabella può avere solo una *primary key*, ma più attributi *unique*

Chiavi su più attributi

- Ad esempio nello **schema seguente**, la tabella Acquisti ha una chiave primaria formata da due attributi

Clienti(user, pwd, nome, cognome, indirizzo, nascita, sesso)
Ordini(numero, data_ora, idArticolo, idCliente)
Acquisti(idOrdine, idProdotto, quantità)
Prodotti(codiceArticolo, categoria, modello, colore, taglia, prezzo, quantità)

;

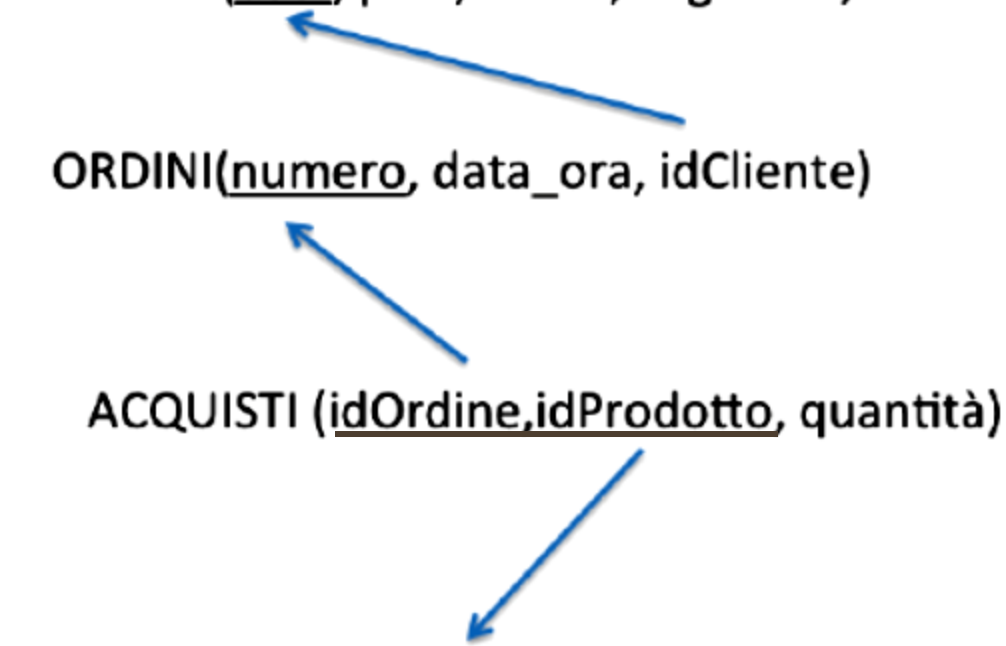
Chiavi su più attributi

CLIENTI(user, pwd, nome, cognome, indirizzo, nascita, sesso)

ORDINI(numero, data_ora, idCliente)

ACQUISTI (idOrdine,idProdotto, quantità)

PRODOTTI(codiceArticolo, categoria, modello, colore, taglia, prezzo,quantitàDisponibile)



```
CREATE TABLE acquisti (
idOrdine INT REFERENCES ordini,
idProdotto INT REFERENCES prodotti,
quantità INT NOT NULL,
PRIMARY KEY(idOrdine, idProdotto))
```

Creazione automatica di indici

PRODOTTI

```
CREATE TABLE prodotti (
codiceArticolo INT PRIMARY KEY AUTO_INCREMENT,
categoria VARCHAR(20),
modello VARCHAR(20),
colore VARCHAR(20),
taglia CHAR check(taglia in ('S','M','L','U')),
prezzo NUMERIC(20,2),
quantita INT NOT NULL check(quantita >=0) );
```

Il vincolo AUTO_INCREMENT spefica che se si omette il codiceAritcolo questo sarà crauto automaticamente in modo incrementale

Inserimento dati in una tabella

- Esempio

```
INSERT INTO
country
VALUES
('Albania','AL','Tirane','Albania',28750,3249136);
```

Name	Code	Capital	Province	Area	Population
'Albania’	'AL’	'Tirane’	'Albania'	28750	3249136

Dalla creazione di un DB all'inserimento dei valori

```
CREATE DATABASE recipes_database;

USE recipes_database;

CREATE TABLE recipes (
  recipe_id INT NOT NULL,
  recipe_name VARCHAR(30) NOT NULL,
  PRIMARY KEY (recipe_id),
  UNIQUE (recipe_name)
);
```

```
INSERT INTO recipes
    (recipe_id, recipe_name)
VALUES
    (1,"Tacos"),
    (2,"Tomato Soup"),
    (3,"Grilled Cheese");
```

Interrogazione di un database

- Le interrogazioni o **query di selezione**, sono la funzionalità principale di SQL.
- Tutte le query hanno un struttura **SELECT-FROM-**[**WHERE**]
- Tutte le query iniziano quindi con la clausola **SELECT**, anche chiamata ***target list***

Struttura base di una query

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

- clausola SELECT → **PROIEZIONE**, scelgo le colonne
- clausola FROM → Relazione/i da cui voglio estrarre le informazioni
- clausola WHERE → **SELEZIONE**, scelgo le righe

Proiezione senza selezione

SELECT attributi

FROM tabelle

ESEMPIO di query sulla relazione
CLIENTI(user, pwd, nome, cognome, indirizzo, nascita, sesso)

SELECT nascita
FROM clienti

La query ritorna la data di nascita di tutti i clienti

Clausola WHERE

Clausola WHERE: è opzionale (può essere omessa), filtra le tuple in base ad una condizione specifica. Espressione logica di predicati

1. operatori **logici** (**AND, OR, NOT**)

2. Operatori di **comparazione** (o confronto)

=	Uguale
<> *oppure* **!=**	Diverso
<	Minore di
>	Maggiore di
<=	Minore o uguale di
>=	Maggiore o uguale di

3. Operatori sulle **stringhe** (**LIKE**)

4. Operatori **matematici**

+
-
*****
/

5. Parole chiave di **confronto** (**BETWEEN, IN**)

Esempio di utilizzo della clausola WHERE

luogo	data	mercalli	magnitudo	Cod_terremoto
Aquila	2025	10	5,9	1
Christchurch	2021	8	6	2
Messina	2018	11	7	3

Tabella (relazione) terremoti

```
SELECT luogo, data, mercalli
FROM terremoti
WHERE magnitudo < = 6;
```



Estraggo dalla tabella *terremoti* gli attributi *luogo, data, mercalli* degi terremoti che non hanno superato la magnitudo 6

Risultato della selezione:

Risposta?

Selezione senza proiezione (*)

luogo	data	mercalli	magnitudo	Cod_terremoto
Aquila	2025	10	5,9	1
Christchurch	2021	8	6	2
Messina	2018	11	7	3

Tabella (relazione) terremoti

```
SELECT *
FROM terremoti
WHERE magnitudo < = 6;
```

L'asterisco * indica che voglio estrarre TUTTI gli attributi della tabella terremoti, non soltanto luogo data e scala Mercalli

↓

La condizione WHERE non è obbligatoria. Posso anche voler visualizzare (proiettare) i dati senza selezionarne delle specifiche caratteristiche.

Risultato della selezione:

Risposta?

Selezionare stringhe di caratteri

LIKE si usa per selezionare stringhe di caratteri, che devono essere espresse tra due apici ' '

Nelle stringhe, _ indica un carattere qualsiasi, mentre % un numero qualsiasi di caratteri

es. voglio selezionare le tuple che nel campo località iniziano per M e finiscono per a

Selezionare stringhe di caratteri

```
SELECT luogo, data
FROM terremoti
WHERE mercalli > = 9
and localita  LIKE 'M%a';
```

luogo	data
Messina	2018



'M%a' significa: prima lettera M, numero qualsiasi di lettere, lettera finale a (es. Messina, Monza, Molfetta...)

Altro esempio: **'M_l%'** significa: prima lettera M, seconda qualsiasi, terza l, e poi numero imprecisato di lettere (es. Milano, Milazzo, Molfetta...)

Selezione di campi nulli

Anche i campi nulli possono essere selezionati

Es. voglio ottenere nomi, cognomi e residenza degli studenti che non frequentano più l'università

```
SELECT cognome, nome, residenza
FROM studenti
WHERE Scuola_attualm_frequentata IS NULL;
```

cod fiscale	nome	cognome	residenza	anni	titolo_ studio	luogo_ nascita	scuola_attualm _frequentata
MTTBNC	Matteo	Bianchi	Milano	24	Dipl	Milano	NULL
LRARSS	Lara	Rossi	Sesto S.G.	27	Laurea	Milano	NULL
LCAVRD	Luca	Verdi	Bologna	24	Dipl	Bologna	Università
NNARSS	Anna	Rossi	Torino	23	Dipl	Settimo T.se	Università

studenti

Risultato selezione

Selezione di campi nulli

Anche i campi nulli possono essere selezionati

Es. voglio ottenere nomi, cognomi e residenza degli studenti che non frequentano più l'università

```
SELECT cognome, nome, residenza
FROM studenti
WHERE Scuola_attualm_frequentata IS NULL;
```

N.B. NON POSSO SCRIVERE attributo = NULL ma devo scrivere IS NULL

cod fiscale	nome	cognome	residenza	anni	titolo_ studio	luogo_ nascita	scuola_attualm _frequentata
MTTBNC	Matteo	Bianchi	Milano	24	Dipl	Milano	NULL
LRARSS	Lara	Rossi	Sesto S.G.	27	Laurea	Milano	NULL
LCAVRD	Luca	Verdi	Bologna	24	Dipl	Bologna	Università
NNARSS	Anna	Rossi	Torino	23	Dipl	Settimo T.se	Università

studenti

Risultato selezione

Esercizio

- Data la tabella precedente:

  Studenti(cod_fiscale, nome, cognome, residenza, anni, titolo_studio, luogo_nascita, scuola_attualm_frequentata)

  Scrivere il codice SQL che restituisca **nome**, **cognome**, **residenza** e **anni** di coloro che frequentano l'università

  SELECT nome, cognome, residenza, anni
  FROM studenti
  WHERE scuola_attualm_frequentata = 'Università';

Selezionare valori disinti

Specificando **DISTINCT** nella selezione è possibile eliminare dai risultati le righe duplicate, evitando la ridondanza dei dati. Se non viene esplicitato, SELECT fornisce tutte le tuple in selezione, anche se ci sono ripetizioni

```
SELECT DISTINCT cognome, nome, residenza
FROM studenti
WHERE Scuola_attualm_frequentata IS NULL ;
```

Esempio: caso di omonimia!

nome	cognome	residenza
Matteo	Bianchi	Milano
Lara	Rossi	Sesto S.G.
Matteo	Bianchi	Milano

Senza uso di DISTINCT

nome	cognome	residenza
Matteo	Bianchi	Milano
Lara	Rossi	Sesto S.G.

Con uso di DISTINCT

Selezionare valori in un intervallo

L'operatore **BETWEEN** permette di esprimere condizioni di appartenenza ad un intervallo

Rilievi_piezometrici

piezometro	Data rilievo	Livello piezometrico	Testa Pozzo	contaminazione	Contaminante
P1	13/6/03	199	212	SI	0,04
P2	13/6/03	200	209	NO	0
P3	13/6/03	203	210	SI	0,25
P4	13/6/03	186	199	SI	0,32

```
SELECT piezometro
FROM Rilievi_piezomatrici
WHERE contaminante BETWEEN 0,1 AND 0,3;
```

Risposta?

Ordinare i risultati

- La clausola **ORDER BY**, che compare in coda all'interrogazione, ordina le righe del risultato

- Sintassi:

  ORDER BY AttributoOrdinamento [ASC | DESC]
  {, AttributoOrdinamento [ASC | DESC] };

- Le condizioni di ordinamento vengono valutate in ordine
  - a pari valore del primo attributo, si considera l'ordinamento sul secondo, e così via

Ordinare i risultati

studenti

cognome	nome	anni	residenza	matricola	anno iscrizione
Bianchi	Anna	26	Milano	234567	2011
Rossi	Luca	23	Bologna	876987	2011
Verdi	Ada	24	Torino	234654	2012

ORDER BY serve per ordinare i risultati in base a uno o più campi.

```
SELECT cognome, nome, residenza
FROM studenti
WHERE anni < = 25
ORDER BY cognome;
```

cognome	nome	residenza
Rossi	Luca	Bologna
Verdi	Ada	Torino

Ordine alfabetico

Esercizio

- Data la tabella

studenti

cognome	nome	anni	residenza	matricola	anno iscrizione
Bianchi	Anna	26	Milano	234567	2011
Rossi	Luca	23	Bologna	876987	2011
Verdi	Ada	24	Torino	234654	2012

- Selezionare cognome, nome, anni e ordinare per cognome (crescente) e età (decrescente)

```
SELECT cognome, nome, anni
FROM studenti
ORDER BY cognome ASC, anni DESC;
```

Operazioni in SQL

La SELECT list può contenere non solo attributi, ma anche espressioni

Rilievi_piezometrici

piezometro	Data rilievo	Livello_ piezometrico	Testa_ Pozzo	contami nazione	Contamin ante
P1	13/6/03	199	212	SI	0,04
P2	13/6/03	200	209	NO	0
P3	13/6/03	203	210	SI	0,25
P4	13/6/03	186	199	SI	0,32

```
SELECT piezometro,
        (Testa_Pozzo – Livello_piezometrico)
FROM Rilievi_piezometrici;
```

piezometro	
P1	13
P2	9
P3	7
P4	13



Risultato in colonna senza nome

Concatenare condizioni con la clausola WHERE

La condizione, espressa da WHERE, può anche essere complessa e comprendere più condizioni:

luogo	data	mercalli	magnitudo	Cod_terremoto
Aquila	2025	10	5,9	1
Christchurch	2021	8	6	2
Messina	2018	11	7	3

Tabella (relazione) terremoti

```
SELECT luogo, data
FROM terremoti
WHERE mercalli > = 9
and magnitudo > = 7;
```

] Devono essere vere entrambe contemporaneamente

Risposta?

N.B. Per indicare una stringa, devo utilizzare gli apici ‘ ‘
Es. luogo = ‘Messina’

Esercizio (operatore OR)

Caso 1: voglio effettuare una perforazione per installare un pozzo. L'uso del suolo dell'area di studio deve consentirlo: eviterò le aree boscate, la roccia affiorante, gli alvei fluviali… scelgo solo zone adibite a prato, terreni agricoli e pascoli

Area mq	uso	proprietà
24578	prato	Rossi
3200	terreno_agricolo	Verdi
2457800	urbanizzato	NULL
3400989	industriale	Bianchi
35600329	pascolo	Brambilla

Tabella uso_suolo



Area mq	uso	proprietà
24578	prato	Rossi
3200	terreno_agricolo	Verdi
35600329	pascolo	Brambilla

Risultato della selezione

L'operatore OR esprime le possibili **alternative** che voglio accettare.

```
SELECT*
FROM uso_suolo
WHERE uso='prato' OR uso='pascolo' OR uso='terreno_agricolo';
```

Esercizio (operatore AND)

Caso 2: voglio prelevare dei campioni di roccia sul granito, su di un versante esposto a sud, per fare dei test geomeccanici. Voglio individuare le zone adatte.

Formazione	superficie mq	esposizione
Calcare oolitico	2389700	S
Granito	56700	NE
Dolomia principale	3245	SO
Deposito alluvionale	789003200	NE
Deposito glaciale	3298900	SE
Granito	450098	S

Tabella formazioni

```
SELECT*
FROM formazioni
WHERE formazione='granito' AND esposizione='S';
```



Formazione	superficie mq	esposizione
Granito	450098	S

Risultato della selezione

L'operatore AND esprime le condizioni che devono essere **CONTEMPORANEAMENTE** vere

Esercizio (operatore NOT)

L'operatore NOT esprime le condizioni che **non devono** essere vere

Caso 3: voglio prelevare dei campioni di roccia per fare dei test geomeccanici. Voglio evitare di campionare in zone coperte di detrito di vario tipo

Formazione	superficie mq	esposizione
Calcare oolitico	2389700	S
Granito	56700	NE
Dolomia principale	3245	SO
Deposito alluvionale	789003200	NE
Deposito glaciale	3298900	SE
Granito	450098	S

Tabella formazioni

```
SELECT*
FROM formazioni
WHERE NOT
(formazione='deposito_alluvionale'
OR formazione='deposito_glaciale');
```

Gli operatori logici possono essere combinati!

Formazione	superficie mq	esposizione
Calcare oolitico	2389700	S
Granito	56700	NE
Dolomia principale	3245	SO
Granito	450098	S

Risultato della selezione

Specificare l'attributo di una tabella

L'attributo di una tabella specifica può essere indicato come

nometabella.nomeattributo

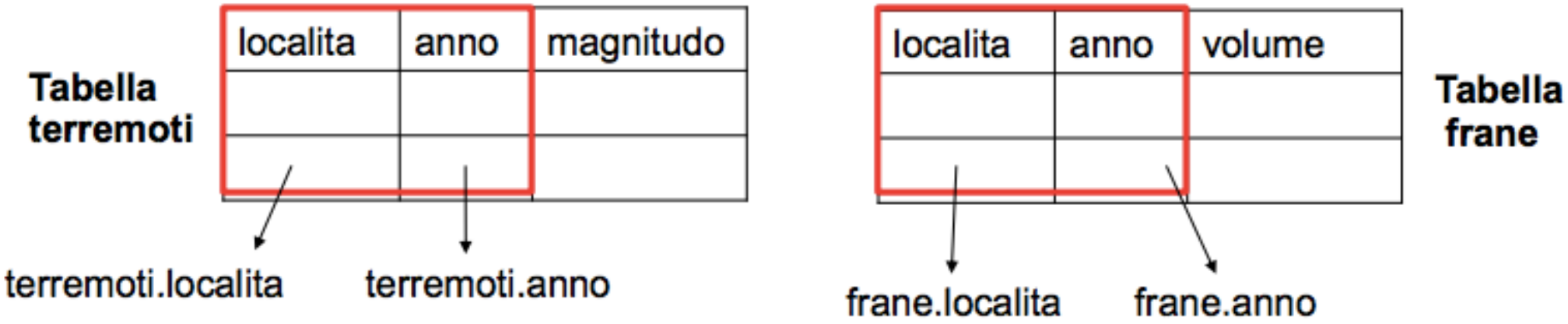
luogo	data	mercalli	magnitudo	Cod_terremoto
Aquila	2025	10	5,9	1
Christchurch	2021	8	6	2
Messina	2018	11	7	3

Tabella (relazione) terremoti

L'attributo data della tabella terremotii, può essere indicato come **terremoti.data**

Specificare l'attributo di una tabella

Questa notazione è indispensabile quando ho più tabelle con attributi con lo stesso nome



Ridenominazione di attributi e/o tabelle

```
SELECT nome, reddito
FROM persone
WHERE eta < 30;
```




```
SELECT p.nome AS name, p.reddito AS income
FROM persone AS p
WHERE p.eta < 30;
```

NB: una volta rinominata, la tabella non può essere riconosciuta (e quindi specificata) con il suo vecchio nome all'interno della query



Selezione dati da più tabelle

- È necessario indicare **più relazioni (più tabelle)** nella clausola FROM
- È possibile effettuare una operazione di **JOIN** tra più tabelle
- Il predicato JOIN esprime una relazione che deve essere verificata dalle tuple che sono il risultato della query
- È possibile definire JOIN:
  - In forma abbreviata
  - In forma esplicita

JOIN (forma abbreviata)

Voglio vedere se esiste una relazione tra magnitudo di terremoti avvenuti in un certo luogo e volume delle frane avvenute nello stesso luogo, nello stesso anno

localita	anno	magnitudo
Christchurch	2021	6
Messina	2018	7

Tabella terremoti

localita	anno	volume
Christchurch	2021	2000
Pola	2017	30,000,000

Tabella frane

Il predicato AS serve a rinominare nomi di tabella o attributi

```
SELECT magnitudo, volume
FROM terremoti AS T, frane AS F
WHERE T.località=F.località
and T.anno=F.anno ;
```

Posso lasciare il nome semplice, non c'è possibilità di confusione

Per comodità, chiamo **T** la tabella terremoti, e **F** la tabella frane

Devo usare la notazione completa, altrimenti si crea confusione tra tabelle

Risultato della selezione

magnitudo	volume
6	2000

SELECT con JOIN esplicito, sintassi

SELECT column-names
FROM table-name1 JOIN table-name2
ON column-name1 = column-name2
WHERE condition



Formato generale

SELECT magnitudo, volume
FROM terremoti AS T JOIN frane AS F
ON T.località = F.località AND
T.Anno = F.anno
WHERE … (eventuali condizion aggiuntive)

```
SELECT magnitudo, volume
FROM terremoti AS T, frane AS F
WHERE T.località=F.località
and T.anno=F.anno
```

Risorse utili per SQL

- http://joshualande.com/create-tables-sql
- https://www.w3schools.com/sql/default.asp
- Microsoft Access (licenza studenti)
- MySQL Workbench: https://www.mysql.com/it/products/workbench/
  - Slide installazione e utilizzo