

# Introduzione alle basi di dati



Marco Botta

botta@di.unito.it

[www.di.unito.it/~botta/didattica/bioinfo.html](http://www.di.unito.it/~botta/didattica/bioinfo.html)

1

---

---

---

---

---

---

---

---

## Sistema Informativo

- Insieme di “strutture” in grado di acquisire, elaborare, trasmettere ed archiviare informazioni ad uso di un’organizzazione.
  - Risorse umane
  - Procedure manuali o automatizzate per il trattamento delle informazioni.



2

---

---

---

---

---

---

---

---

## Dato <> Informazione

- I dati sono registrazioni della descrizione di una qualsiasi caratteristica della realtà, su un supporto che ne garantisca la conservazione, la comprensibilità e la reperibilità.
- L’informazione produce variazioni nel patrimonio conoscitivo di un soggetto. Proviene dai dati, inseriti in un contesto interpretativo



3

---

---

---

---

---

---

---

---

## DBMS

- Obiettivo: gestione strutturata di dati, organizzati in modo omogeneo.
- Base di dati:
  - Collezione di dati organizzati in modo coerente (un insieme casuale di dati non è una base di dati !)
  - Modella alcuni aspetti del mondo reale
  - Costruito con funzionalità ben precise, note fin dall'inizio della progettazione.



4

---

---

---

---

---

---

---

---

## Basi di dati: esempi

- Rubrica telefonica personale
- anagrafe
- archivio di una biblioteca
- archivio centrale del ministero delle finanze
- segreteria studenti dell'università
- archivio laboratorio d'analisi
- ....



5

---

---

---

---

---

---

---

---

## Perché

- Organizzare e strutturare i dati
- Maggiore semplicità di utilizzo
- Facilità nel reperire le informazioni
- Facilità nell'estrarre i dati che servono
- Facilità nel mantenimento e aggiornamento delle informazioni



6

---

---

---

---

---

---

---

---

## Basi di dati: operazioni

- Definizione della base di dati
  - quali informazioni
  - quali relazioni tra le informazioni
- Manipolazione
  - inserimento di dati
  - cancellazione di dati
  - aggiornamento (update)
  - interrogazione
- Protezione e sicurezza dei dati



7

---

---

---

---

---

---

---

---

## Livelli di rappresentazione

- Livello fisico: come i dati sono memorizzati e organizzati su uno o più supporti di memoria secondaria
- Livello logico: come i dati sono organizzati secondo il modello logico adottato (relazionale, gerarchico, ad oggetti etc.)
- Livello concettuale: come i dati sono organizzati secondo uno schema concettuale
- Livello esterno: come i dati appaiono o vengono presentati all'utente



8

---

---

---

---

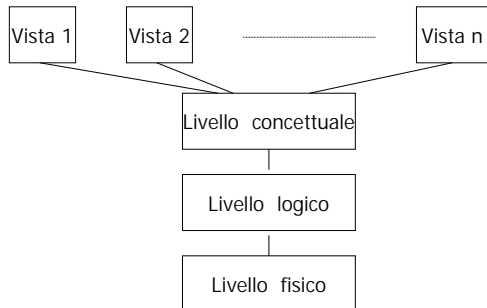
---

---

---

---

## Livelli di rappresentazione



9

---

---

---

---

---

---

---

---



## Modello relazionale

- Relazione su due insiemi A e B è un sottoinsieme del prodotto cartesiano  $A \times B$
- Esempio  
sposato\_con  $\subseteq$  Persone x Persone  
figlio\_di  $\subseteq$  Persone x Persone  
vive\_a  $\subseteq$  Persone x Città



13

---

---

---

---

---

---

---

---

## Esempio di relazione

- Persone = {Paolo, Luca, Mario}
- Città = {Torino, Roma}

### Persone x Città

Persone	Città
Paolo	Torino
Paolo	Roma
Luca	Torino
Luca	Roma
Mario	Torino
Mario	Roma

### Vive\_a

Persone	Città
Paolo	Torino
Luca	Torino
Mario	Roma



14

---

---

---

---

---

---

---

---

## Relazioni e Attributi

Attributi: nomi che specificano un ruolo in una relazione, esempi:

- sposato\_con(Marito, Moglie)
- figlio\_di(Figlio, Genitore)
- vive\_a(Nome, Città, Provincia)
- libro(N. Inv, Autore, Titolo, Anno\_edizione, Casa\_editrice, Collocazione)

- Gli attributi devono avere nomi differenti



15

---

---

---

---

---

---

---

---

## Terminologia

- **DOMINIO**: insieme di valori atomici
  - es. Nomi di persona, Nomi di città, Numeri interi
- **FORMATO**: rappresentazione degli elementi di un dominio
  - es. tre cifre decimali per i reali
- **SCHEMA DI RELAZIONE**  $R(A_1, \dots, A_n)$ 
  - R relazione
  - $A_i$  attributi



16

---

---

---

---

---

---

---

---

## Terminologia (2)

- **ISTANZA DI RELAZIONE**  
 $r(R) = \{ \langle v_1, v_2, \dots, v_n \rangle \}$

$t = \langle v_1, v_2, \dots, v_n \rangle$  : n-tupla

- I singoli valori  $v_i$  appartengono al dominio dell'attributo  $A_i$
- valore speciale "null" aggiunto a tutti i domini, per indicare assenza di valore
- Istanza di relazione = insieme non ordinato



17

---

---

---

---

---

---

---

---

## Terminologia (3)

- Istanza di relazione = insieme non ordinato di tuple
  - Non ci possono essere tuple ripetute
  - l'ordine delle tuple non conta
- Database = insieme di istanze di relazioni
- Terminologia alternativa:
  - tuple: *records*
  - attributi: *campi*



18

---

---

---

---

---

---

---

---

### La tabella rubrica

Rubrica(Cognome, Nome, Tel)

schema

nomi degli attributi

COGNOME	NOME	TEL.
Tanzi	Rosario	095 89 ...
Federici	Susanna	011 55 ...
De Bernardi	Silvio	06 44 ...

tuple

19

---

---

---

---

---

---

---

---

---

---

### Vincoli sulle Relazioni

- I valori contenuti nelle tabelle possono essere soggetti a vari tipi di vincoli che dipendono dalla 'realità' che si vuole rappresentate:
- **vincoli di dominio:** valori dei singoli attributi
- **vincoli di tupla:** valori di attributi correlati in una tupla
- **vincoli di integrità:** valori di attributi in tuple diverse (anche in relazioni diverse)

20

---

---

---

---

---

---

---

---

---

---

### Vincoli di dominio

vincoli sui valori dei singoli attributi (vincoli di dominio), es:

- dato ESAMI(Studente, Voto, Lode, Corso) deve essere  
 $Voto \geq 18 \text{ AND } Voto \leq 30$
- Vincoli sul valore di un attributo data

21

---

---

---

---

---

---

---

---

---

---

## Vincoli di tupla

vincoli sui valori di attributi correlati, es:

- Data la relazione ESAMI come prima deve essere

not (Lode = Yes) OR Voto = 30

equivalente a

Lode= No OR Voto = 30

- Data la relazione PAGAMENTI(Data,Importo,Ritenute,Netto) deve essere

Netto = Importo - Ritenute

22

---

---

---

---

---

---

---

---

## Esempio Vincolo (complesso)

Supponiamo che STUDENTE contenga anche gli attributi

- Borsa di Studio: valori {Yes, No}
- Reddito (della famiglia): euro
- Residente (nella città sede univ.): {Yes, No}

Uno studente ha diritto ad una borsa se

- ha una media  $\geq 27$  e un reddito  $\leq 20000$ , oppure

■ non è residente e ha una media  $\geq 25$  e un reddito  $\leq 25000$

23

---

---

---

---

---

---

---

---

## Relazioni tra tabelle

- In una base di dati relazionale le tabelle sono collegate tra loro tramite attributi comuni

■ Esempio:

- Il collegamento tra la relazione LIBRI e la relazione CASE\_EDITRICI può essere espressa mediante l' attributo "Numero di inventario".

28

---

---

---

---

---

---

---

---

## Funzionalità del DBMS relazionale

- Funzioni per
  - definizione della base di dati
  - inserimento / rimozione /aggiornamento di informazioni
    - deve soddisfare i vincoli!
  - Interrogazione



29

---

---

---

---

---

---

---

---

## Interrogazione

- Linguaggio SQL
- Produce come risultato una tabella
- Clausola Base
  - SELECT "Lista di attributi"  
FROM "Elenco relazioni"  
WHERE "Condizione"



30

---

---

---

---

---

---

---

---

## Esempi

- SELECT Autore, Titolo  
FROM Libri  
WHERE Casa\_Editrice = 'Feltrinelli'
- SELECT Autore, Titolo  
FROM Libri  
WHERE (Casa\_Editrice = Feltrinelli) and  
(Anno\_edizione = 1990)
- SELECT \*  
FROM Libri  
WHERE (Casa\_Editrice = Feltrinelli) or  
(Casa\_Editrice = 'Einaudi')



31

---

---

---

---

---

---

---

---

## Esempi: query parametriche

- "Trova tutti i libri presenti in biblioteca, dato il nome dell'autore

```
SELECT *  
FROM Libri  
WHERE Autore = [dimmi il nome dell'autore]
```

- In esecuzione:  
 > dimmi il nome dell' autore  
 Alessandro Manzoni
- Risultato:....



32

---

---

---

---

---

---

---

---

## Esempi: select in cascata

- LIBRI(N\_Inv, Autore, Titolo, Casa\_ed, Anno\_ed, Coll, Prezzo)  
 PRESTITI(N\_Inv, Cod\_Utente, Data\_prestito, Data\_restit)  
 UTENTI(Cod\_Utente, Nome, Indirizzo)

```
SELECT *  
FROM LIBRI  
WHERE (N_Inv IN  
  SELECT Num_Inv  
  FROM PRESTITI, UTENTI  
  WHERE (Prestiti.Cod_utente =  
    Utenti.Cod_Utente) and  
    (Utenti.Nome='Mario Rossi'))
```



33

---

---

---

---

---

---

---

---

## Select in cascata (2)

- SELECT INTERNA produce tabella
- SELECT ESTERNA usa tale tabella come condizione
- RISULTATO è solo la tabella prodotta dalla select esterna



34

---

---

---

---

---

---

---

---

### Select in cascata (3)

- Altri operatori:

- ALL
- EXISTS/NOT EXISTS

- Esempio

- SELECT \*  
FROM LIBRI  
WHERE Casa\_Editrice <> ALL  
(SELECT Casa\_Editrice  
FROM LIBRI  
WHERE Autore= 'Alessandro Manzoni')



35

---

---

---

---

---

---

---

---

### Operatori insiemistici

- UNION, INTERSECT, DIFFERENCE

- ESEMPIO

- SELECT Autore, Titolo  
FROM Libri  
WHERE Casa\_Editrice = 'Feltrinelli'

INTERSECT

- SELECT Autore, Titolo  
FROM Libri  
WHERE Casa\_Editrice = 'Einaudi'



36

---

---

---

---

---

---

---

---

### Funzioni aggregate

- COUNT, SUM, MAX, MIN, AVG

- ESEMPI:

- Contare i libri presenti in biblioteca editi da Feltrinelli.

```
SELECT Count(*)  
FROM Libri  
WHERE Casa_Editrice = 'Feltrinelli'
```



37

---

---

---

---

---

---

---

---

## Funzioni aggregate (2)

- Calcolare il costo totale dei libri presenti in biblioteca, scritti da Umberto Eco

```
SELECT Sum (Prezzo)
FROM Libri
WHERE Autore = 'Umberto Eco'
```

- Qual è il prezzo del più costoso libro presente in biblioteca?

```
SELECT Max(Prezzo)
FROM Libri
```



38

---

---

---

---

---

---

---

---

## Raggruppamento

- GROUP BY

- ESEMPI

- Per ogni autore, indicare il costo totale dei libri presenti in biblioteca

```
SELECT Autore, Sum (Prezzo)
FROM Libri
GROUP BY Autore
```



39

---

---

---

---

---

---

---

---

## Raggruppamento (2)

- ESEMPI

- Per ogni autore, contare i libri presenti in biblioteca editi da Laterza, ed indicarne il costo totale

```
SELECT Autore, Count(*), Sum (Prezzo)
FROM Libri
WHERE Casa_Ed = 'Laterza'
GROUP BY Autore
```



40

---

---

---

---

---

---

---

---

### Raggruppamento (3)

- La clausola HAVING consente di imporre una condizione sul risultato di una funzione aggregata
  - Per ogni autore di almeno 3 libri editi da Laterza presenti in biblioteca, indicarne il costo totale

```
SELECT Autore, Sum (Prezzo)
FROM Libri
WHERE Casa_Ed = 'Laterza'
GROUP BY Autore
HAVING Count(*) >=3
```



41

---

---

---

---

---

---

---

---

---

---

### Ordinamento dei risultati

- Si può chiedere che le tuple del risultato siano ordinate in base ai valori dei campi: ORDER BY
- Es. Restituire l'elenco dei libri in catalogo, secondo l'ordine alfabetico degli autori, per anno di edizione decrescente
  - SELECT (\*)
  - FROM LIBRI
  - ORDER BY Autore ASC, Anno\_ed DESC



42

---

---

---

---

---

---

---

---

---

---

### Valori Unici

- SQL restituisce una tabella che contiene tutte le righe che soddisfano una certa condizione, può contenere duplicati
- Per eliminare i duplicati si premette la parola chiave distinct
- Esempio elenca i libri per autore e titolo senza ripetizioni

```
SELECT DISTINCT libri.Autore, libri.Titolo
FROM libri
ORDER BY Autore DESC;
```



43

---

---

---

---

---

---

---

---

---

---

## Query con più tabelle - join

- Join: combinare le tuple di più tabelle i cui valori per attributi correlati soddisfano una condizione di confronto (caso più semplice: sono uguali)
- Il join di due relazioni è il sottoinsieme del loro prodotto cartesiano specificato dalla condizione di selezione



44

---

---

---

---

---

---

---

---

## Relazioni tra tabelle e Join

- Le relazioni tra tabelle sono espresse da valori comuni di attributi correlati
- Esempio Seleziona gli studenti e gli esami che hanno sostenuto con i rispettivi titoli

```
SELECT Studenti.Nome, Studenti.Cognome,  
       Corsi.Titolo, Corsi.Codice  
       Studenti.Matricola  
FROM Corsi, Studenti, Esami  
WHERE Corsi.Codice = Esami.Codice_Corso AND  
       Studenti.Matricola = Esami.Matricola  
ORDER BY Studenti.Cognome;
```



45

---

---

---

---

---

---

---

---

## Formulazione Alternativa (1)

```
SELECT Studenti.Nome,  
       Studenti.Cognome, Corsi.Titolo,  
       Corsi.Codice, Studenti.Matricola  
FROM Corsi, Studenti INNER JOIN Esami  
   ON Studenti.Matricola = Esami.Matricola  
WHERE  
   Corsi.Codice=Esami.Codice_Corso
```



46

---

---

---

---

---

---

---

---

### Formulazione Alternativa (2)

```
SELECT Studenti.Nome,  
       Studenti.Cognome, Corsi.Titolo,  
       Corsi.Codice, Studenti.Matricola  
FROM  
Studenti INNER JOIN (Corsi  
INNER JOIN Esami ON Corsi.Codice =  
Esami.Codice_Corso) ON  
Studenti.Matricola = Esami.Matricola
```



---

---

---

---

---

---

---

---

### Join e Aggregati esempi:

- Per ogni studente determina quanti esami ha sostenuto
- Per ogni corso (titolo) determina il numero di studenti che ne hanno sostenuto l'esame
- Per ogni studente determina la media dei voti
- Elenca gli studenti che hanno una media  $\geq 27$
- Elenca gli studenti che hanno sostenuto più di un esame



---

---

---

---

---

---

---

---

### Esempio

Per ogni studente determina quanti esami ha sostenuto  
(elenca nome,cognome matricola)

```
SELECT Studenti.Nome, Studenti.Cognome,  
       Studenti.Matricola, Count(*) AS  
       Esami_sostenuti  
FROM Studenti INNER JOIN Esami ON  
       Studenti.Matricola = Esami.Matricola  
GROUP BY Studenti.Nome, Studenti.Cognome,  
       Studenti.Matricola  
ORDER BY Studenti.Cognome;
```



---

---

---

---

---

---

---

---