



# INTRODUZIONE AI DATABASE RELAZIONALI

Base dati e tecniche di utilizzo in ambito scientifico – per gli studenti

*Opus Facere – INAF Bologna*

[nicastro@iasfbo.inaf.it](mailto:nicastro@iasfbo.inaf.it)

<http://ross.iasfbo.inaf.it/opusfacere/>

# Il percorso

1. capire quali sono le peculiarità dei "**database**" (nel suo significato più ampio) rispetto alla gestione dei dati come "file" (vedi ad es. i fogli Excel),
2. capire ed usare un set minimo di comandi **SQL**, in particolare per inserire nel database dei dati (**INSERT**) e leggerli (**SELECT**),
3. capire ed usare delle **query** di **SELECT** per accedere e "**visualizzare**" (tabelle e grafici) dei dati scientifici, ad. es. astronomici (il catalogo di stelle più brillanti e quello scaricabile dal sito del progetto SDSS).

# Gli strumenti

1. Un database server: MySQL



2. Un Web server (con PHP)



3. Librerie JavaScript



4. Dati scientifici: cataloghi astronomici



# Più dati per tutti

In informatica, la definizione base di **dati** è:

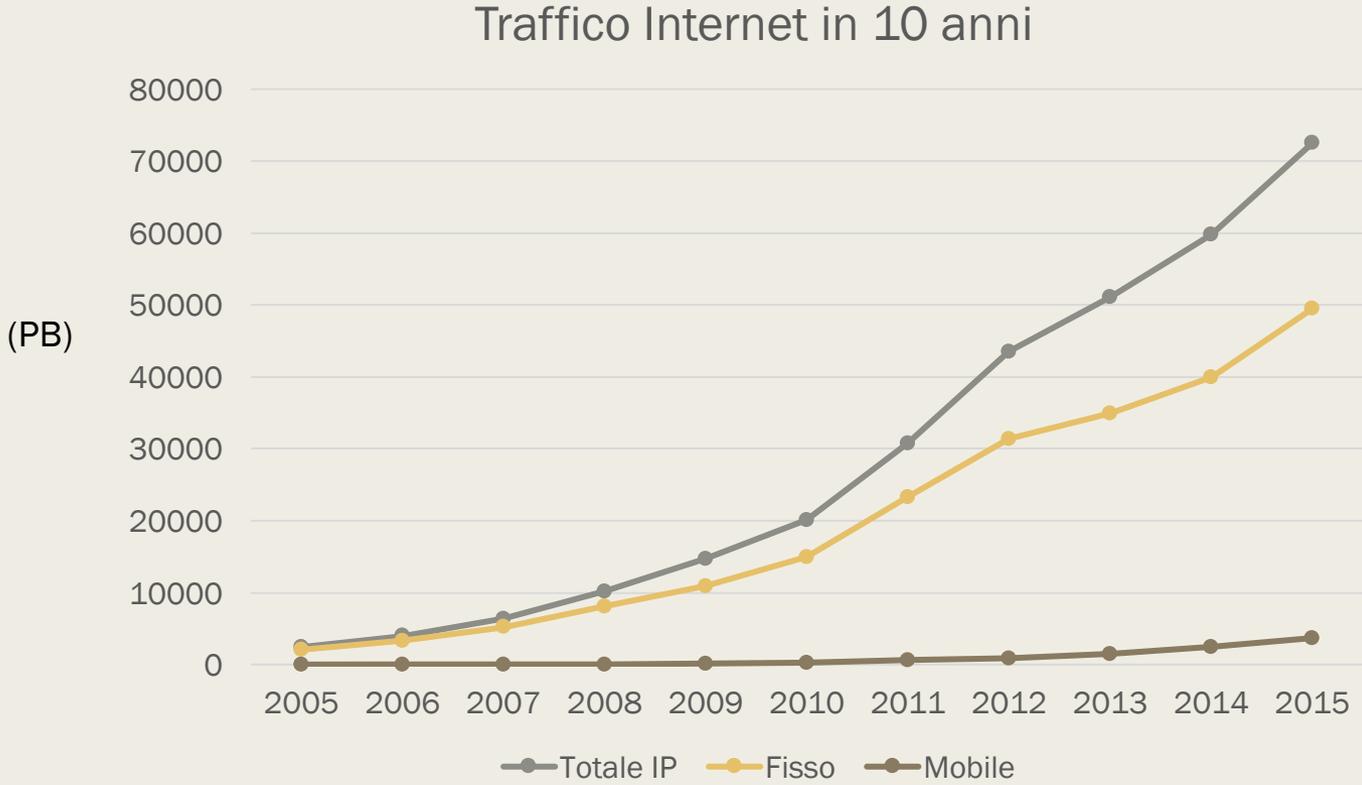
“**una sequenza di byte**”, generalmente organizzata per “**file**”.

Unità	Equivalente in byte
1 byte (B)	8 bit
1 kilobyte (KB)	1,024 bytes
1 megabyte (MB)	1,048,576 bytes
1 gigabyte (GB)	1,073,741,824 bytes
1 terabyte (TB)	1,099,511,627,776 bytes
1 petabyte (PB)	1,125,899,906,842,624 bytes
1 exabyte (EB)	1,024 PB
1 zettabyte (ZB)	1,024 EB
1 yottabyte (YB)	1,024 ZB

# Più dati per tutti

Anno	Traffico IP (PB/month)	Traffico Internet Fisso (PB/mese)	Traffico Internet mobile (PB/mese)
2005	2,426	2,055	0.9
2006	3,992	3,339	4
2007	6,430	5,219	15
2008	10,174	8,140	33
2009	14,686	10,942	91
2010	20,151	14,955	237
2011	30,734	23,288	597
2012	43,570	31,339	885
2013	51,168	34,952	1,480
2014	59,848	39,909	2,514
2015	72,521	49,494	3,685

# Più dati per tutti



Fonte Cisco Systems + Wikipedia



( [Video](#) )

# Cartelle e file



## File Testuali

- Possono contenere solo “**testo**”.
- Non hanno bisogno di un gestore di documenti (es. Word) ma basta un “**text editor**”.
- Possono usare codifiche diverse, es. ASCII, UTF-8, ecc.
- Un errore nel contenuto non ne pregiudica la leggibilità e si “nota” al momento dell’apertura / visualizzazione. Per questo motivo ad es. Microsoft a partire da Office 2007 usa formato testo di tipo XML – “compresso”.

## File Binari

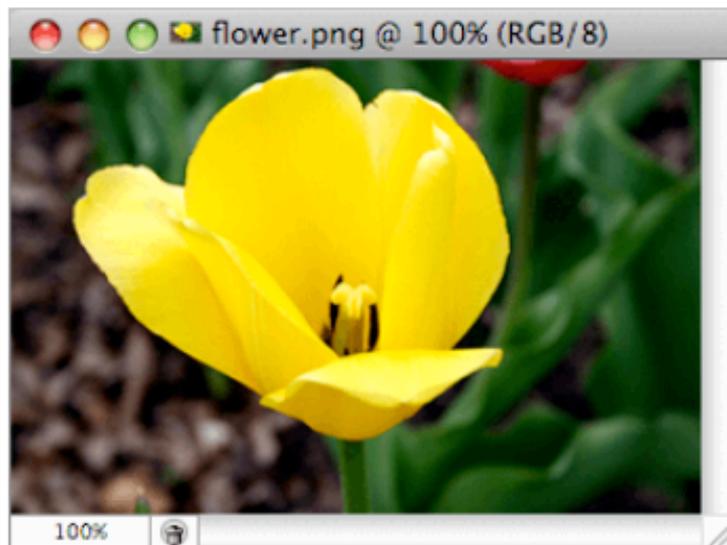
- Contengono una sequenza (qualsiasi) di bytes.
- Possono contenere varie tipologie di dati, ed. es. testo + immagini + video + audio.
- Possono essere “visualizzati” solo da applicativi che sanno interpretarne il contenuto.
- Un errore anche solo di 1 bit può rendere il file inutilizzabile.

Se un carattere della codifica ASCII usa 1 byte, quanti sono i caratteri possibili?

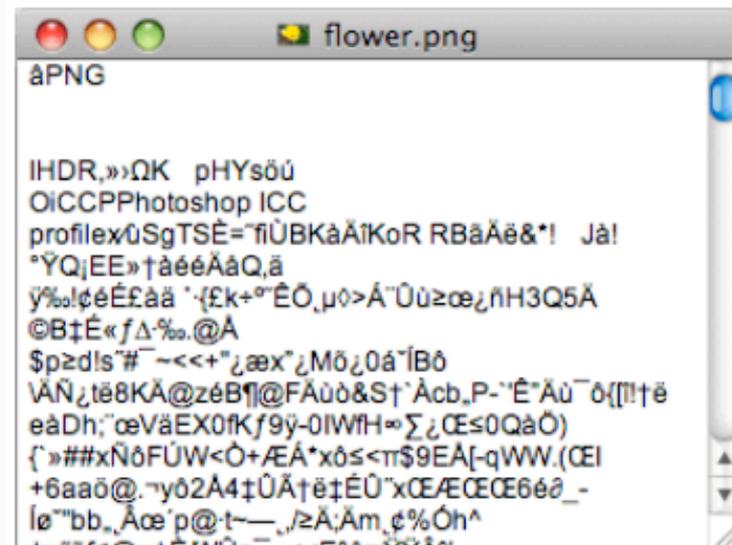
# Cartelle e file

Cosa “vedrei” se trattassi un file binario come testo, ad es. un’immagine PNG?

Visualizzatore di immagine



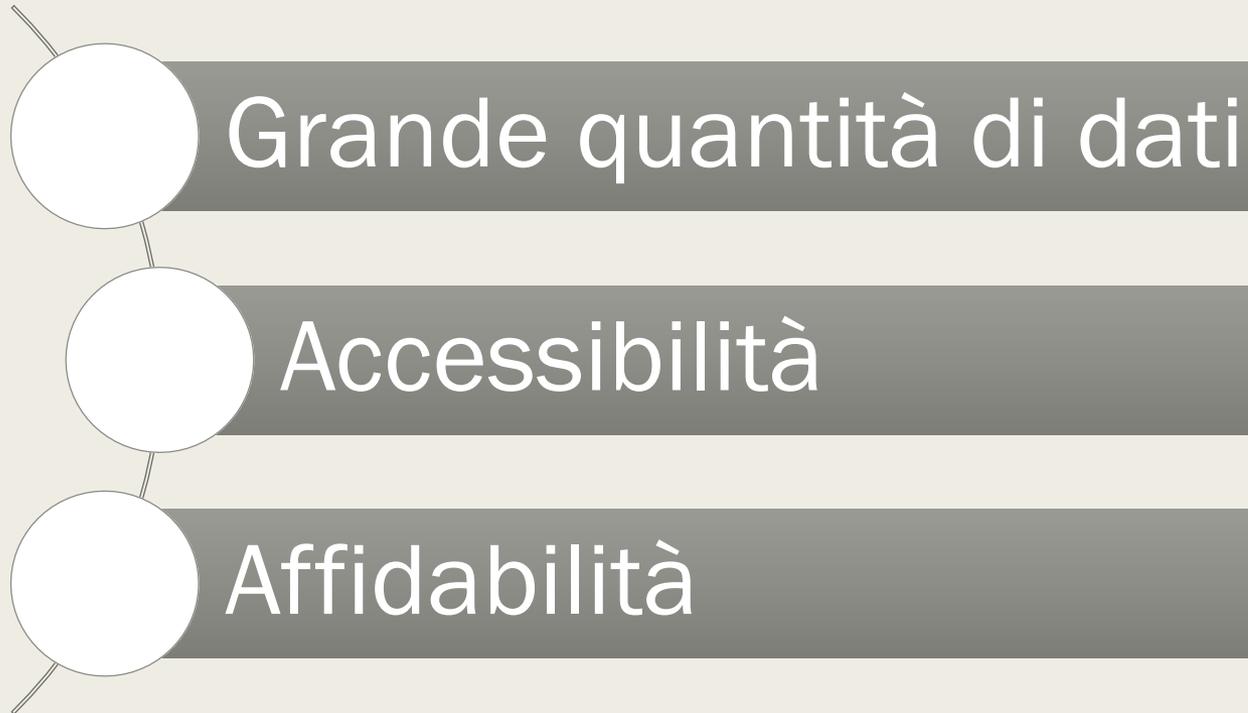
Visualizzatore di testo



Il visualizzatore cerca di interpretare con uno dei possibili “set di caratteri”

# Database: quando

Praticamente tutte le **applicazioni WEB** e gli applicativi che gestiscono **dati variabili nel tempo**, usano un sistema database!  
**Il numero di sistemi attivi è quasi impossibile da stimare.**



In ambito scientifico la sfida maggiore è costituita dalla “varietà” di dati da gestire oltre che dalla quantità.

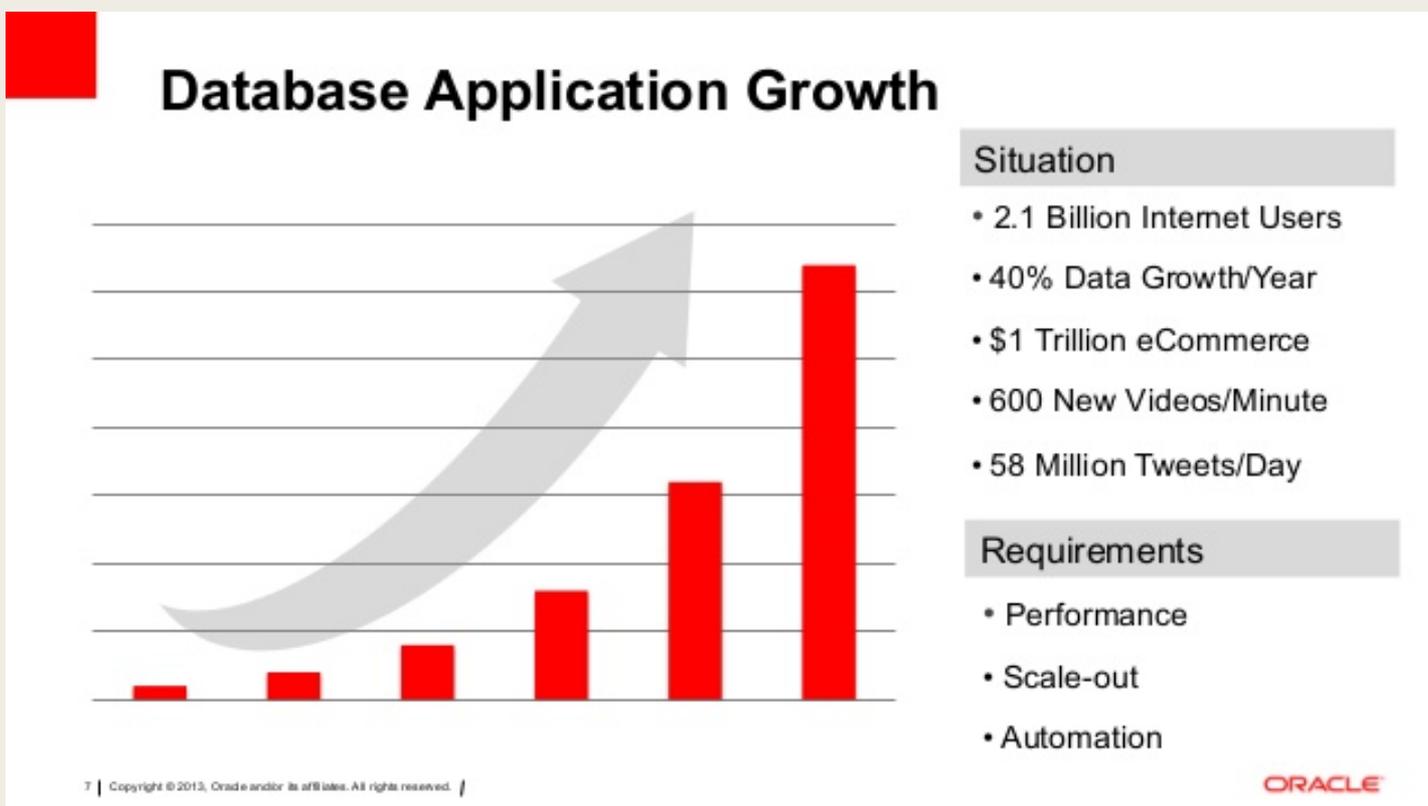
# Database: come

- Sul proprio PC, apparato
- Da terminale o via WEB
- Da applicativo

# Database: chi

*Praticamente chiunque*: Azienda, Ufficio, Ente, Scuola, Università, ecc.

La quantità di dati cresce e nuove metodologie di gestione dei dati vengono sviluppate anche per i “sistemi database”.





# Proposte di utilizzo

Gli esempi a cui abbiamo pensato saranno di tipo:



# Dati: cosa sono

- **Informazione:** notizia, o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni e modi d'essere.
- **Dato:** elementi di informazione costituiti da simboli che devono essere elaborati.

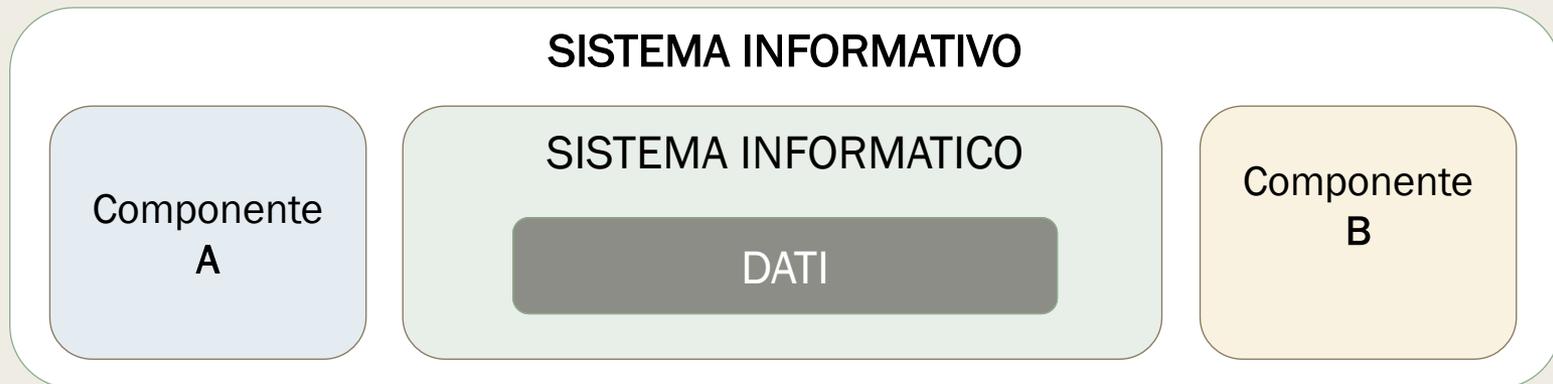
Un **Sistema Informativo (SI)** è una componente di una organizzazione\* il cui scopo è quello di **gestire le informazioni** utili ad i fini dell'organizzazione stessa.

L'esistenza di un **Sistema Informativo** è indipendente dalla sua automatizzazione.

\* Azienda, Ufficio, Ente, Scuola, Università, ecc.

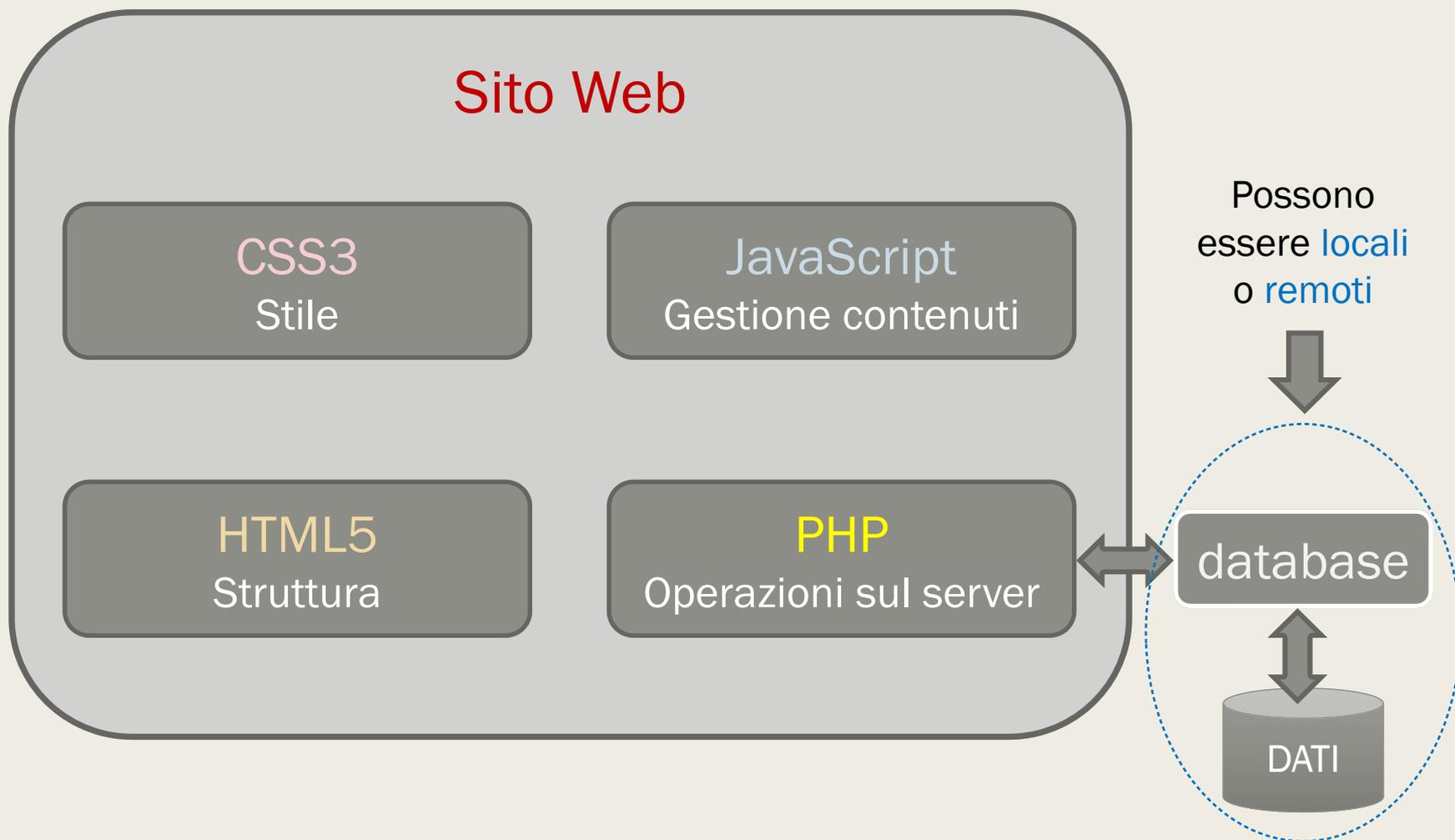
# Sistema informativo

- La porzione **automatizzata** di un sistema informativo prende il nome di **Sistema Informatico**.
- All'interno di un sistema informatico, le informazioni sono rappresentate da **dati** ...



# Il ruolo del database per il web

I componenti base di un sito web moderno sono tipicamente **4**



# Le tabelle

E' la struttura di riferimento per i database

TABELLA  
SCHEMA

COLONNE  
CAMPI

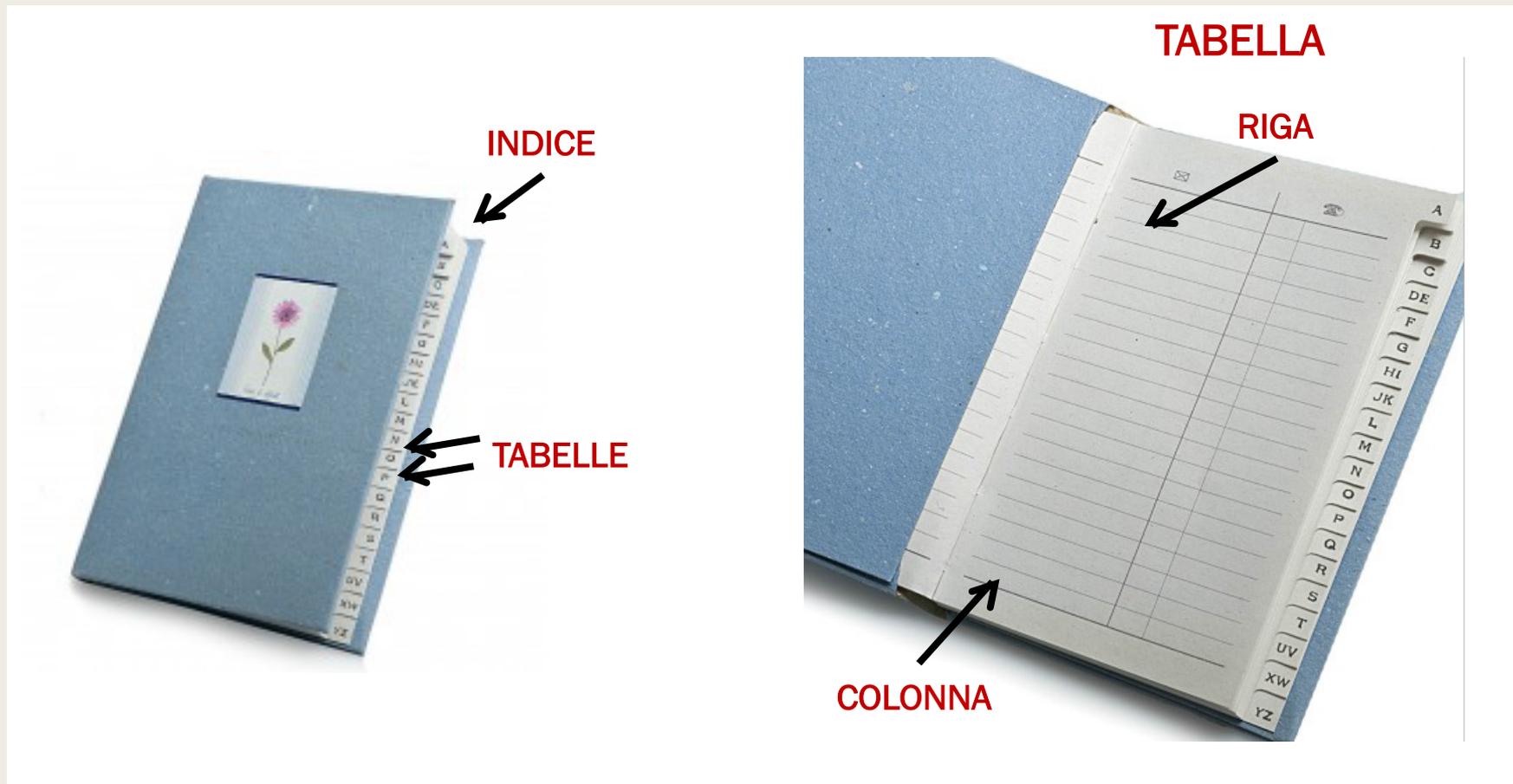
Codice (intero)	Nome (carattere)	N_Ore (reale)	Semestre (intero)	Crediti (intero)
010	Basi di Dati	72.5	1	9
022	Algoritmi	90.3	1	12
...	...	...	...	...

METADATI

RIGHE  
ISTANZE

- Ad es. chiamiamo la tabella sopra **“corsi”**
- I più comuni tipi di dati sono: intero, reale (float), carattere (stringa), data (timestamp).

# Un database d'altri tempi



# Dati: gestione

Gran parte dei sistemi informatici hanno necessità di gestire **dati in maniera persistente**.

**Persistente** → Dati memorizzati su memoria secondaria, es. Hard Disk



Come anticipato, gli APPROCCI di GESTIONE possono essere diversi:

- **Convenzionale** (basato su *files*)
- **Strutturato** (basato su *software di gestione dei dati*)

# Dati: gestione con file

Gran parte dei sistemi informatici hanno necessità di gestire **dati in maniera persistente**.

- Nessuna chiara distinzione tra **dati ed applicazioni**.
- L'applicazione contiene al suo interno la **logica di gestione e memorizzazione** dei dati stessi (es. formato dei dati).
- Il **Sistema Operativo** offre i componenti di base per l'accesso ai files ed i meccanismi di sicurezza del **file-system**.

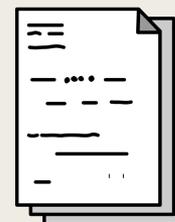
APPLICAZIONE



Operazioni di **Lettura/Scrittura**  
su file mediante supporto del  
Sistema Operativo



FILES



# Dati: gestione con file

## PROBLEMA 1: Gestione di grandi quantità di dati?

Qualche esempio “estremo” (ma neanche tanto):

AMAZON



Oltre 200 Terabyte di dati (istantanei)  
244 Milioni di clienti iscritti  
1.1 Miliardi di ordini all'anno

AT&T



> 400 Terabyte di dati  
> 400 Milioni di clienti  
> 2 Trilioni di record relativi a chiamate

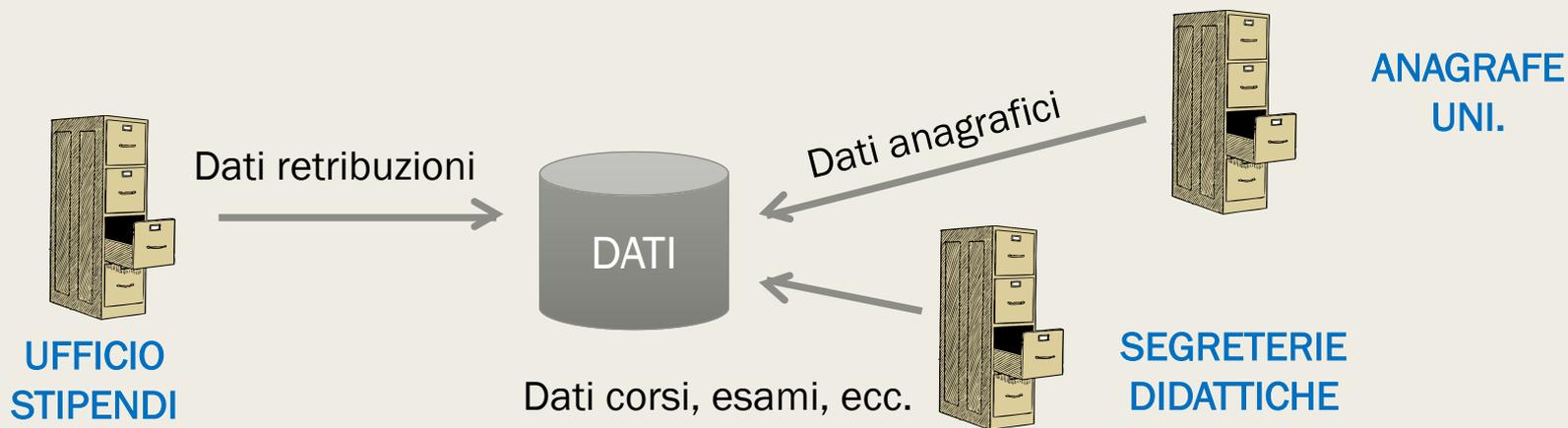
Ovvi problemi di **scalabilità** ed **efficienza** ...

# Dati: gestione con file

## PROBLEMA 2: Condivisione ed accesso concorrente?

In molti scenari pratici, i dati devono essere a disposizione di una moltitudine di utenti/applicazioni per accessi concorrenti.

Es. Dati del personale strutturato universitario



# Dati: gestione con file

## PROBLEMA 2: Condivisione ed accesso concorrente?

### LIMITAZIONI

- Accesso a file condivisi avviene attraverso le politiche di accesso del file-system → *Lock* a livello di file, **bassa granularità di concorrenza**, prestazioni limitate!
- Applicazioni diverse devono conoscere l'esatta collocazione e formato dei dati → **Aggiornamento del formato dei dati?**
- In alternativa: replica dei dati presso i vari sistemi/utenti che ne fanno utilizzo → **Consistenza delle repliche?**

# Dati: gestione

Gran parte dei sistemi informatici hanno necessità di gestire **dati in maniera persistente**.

**Persistente** → Dati memorizzati su memoria secondaria, es. Hard Disk



Come anticipato, gli APPROCCI di GESTIONE possono essere diversi:

- **Convenzionale** (basato su *files*)
- **Strutturato** (basato su *software di gestione dei dati*)



# Introduzione ai Database Management System (**DBMS**)

Un **DBMS** è un sistema software che è in grado di gestire collezioni di dati *grandi, condivise e persistenti*, in maniera efficiente e sicura.

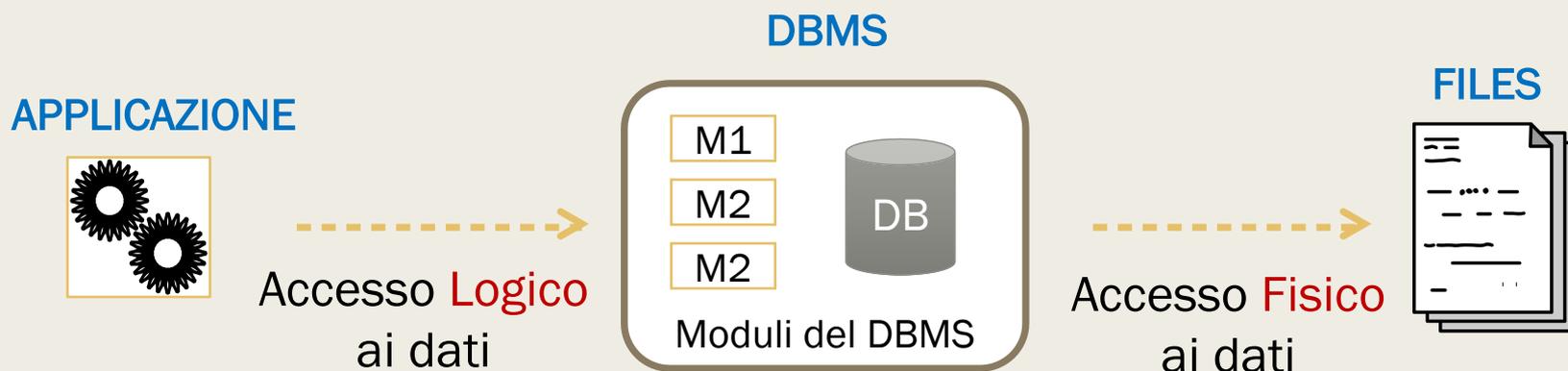
## (ALCUNE) FUNZIONALITA':

- Creazione di una base di dati e memorizzazione su memoria secondaria
- Accesso in lettura/scrittura ad i dati
- Condivisione di dati tra diversi utenti/applicazioni
- Protezione dei dati da accessi non autorizzati
- Reliability dei dati in caso di guasti (hardware/software)
- ...

**Un database è una collezione di dati gestita da un DBMS!**

# DBMS: caratteristiche

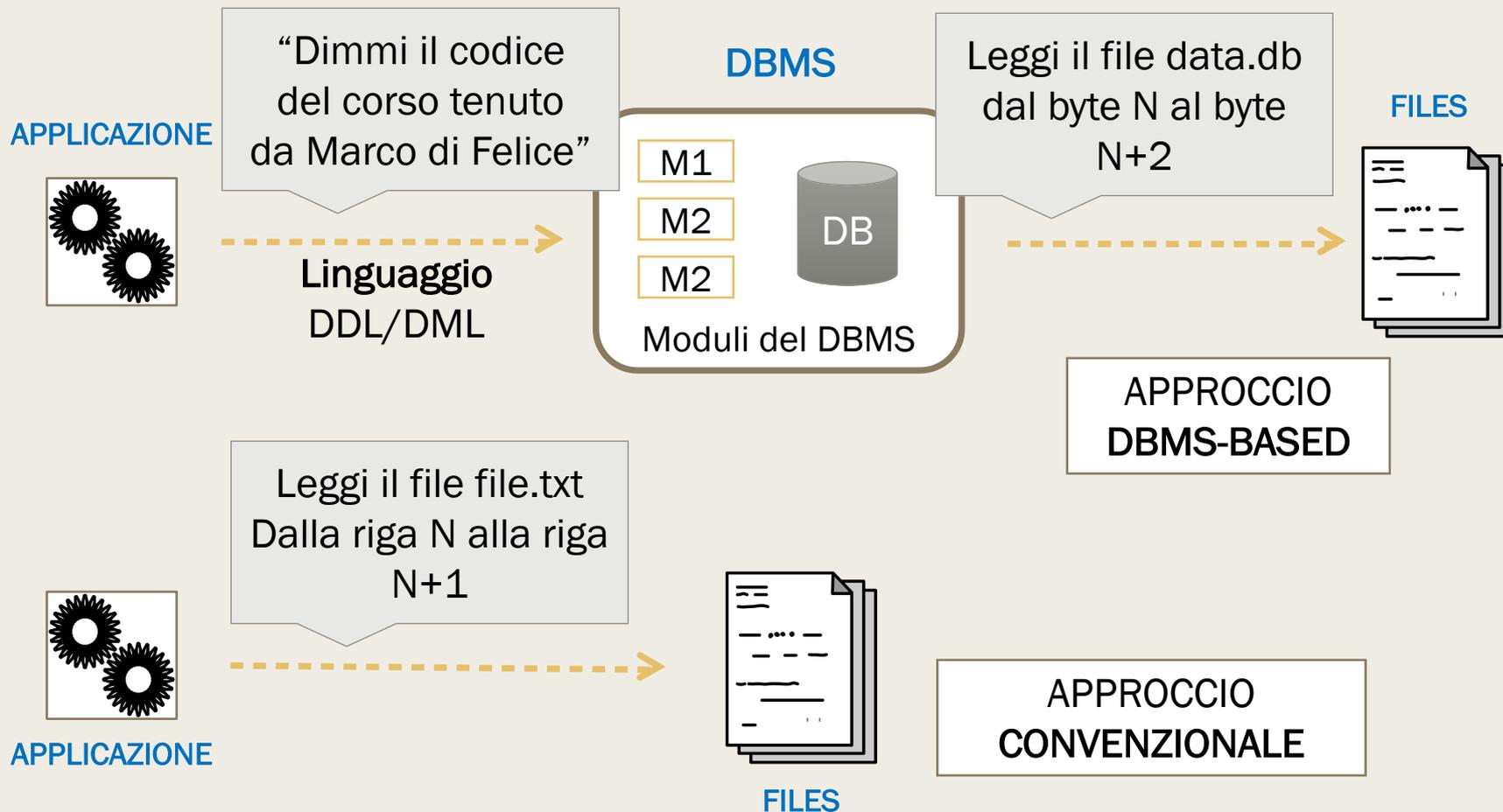
## Separazione Dati – Applicazioni con DBMS



- Tramite i DBMS, è possibile implementare un **paradigma di separazione di dati ed applicazioni ...**
- Le applicazioni non necessitano di conoscere la struttura fisica dei dati (es. come e dove sono memorizzati su disco) ma **solo la struttura logica** (cosa rappresentano).

# DBMS: caratteristiche

## Separazione Dati – Applicazioni con DBMS

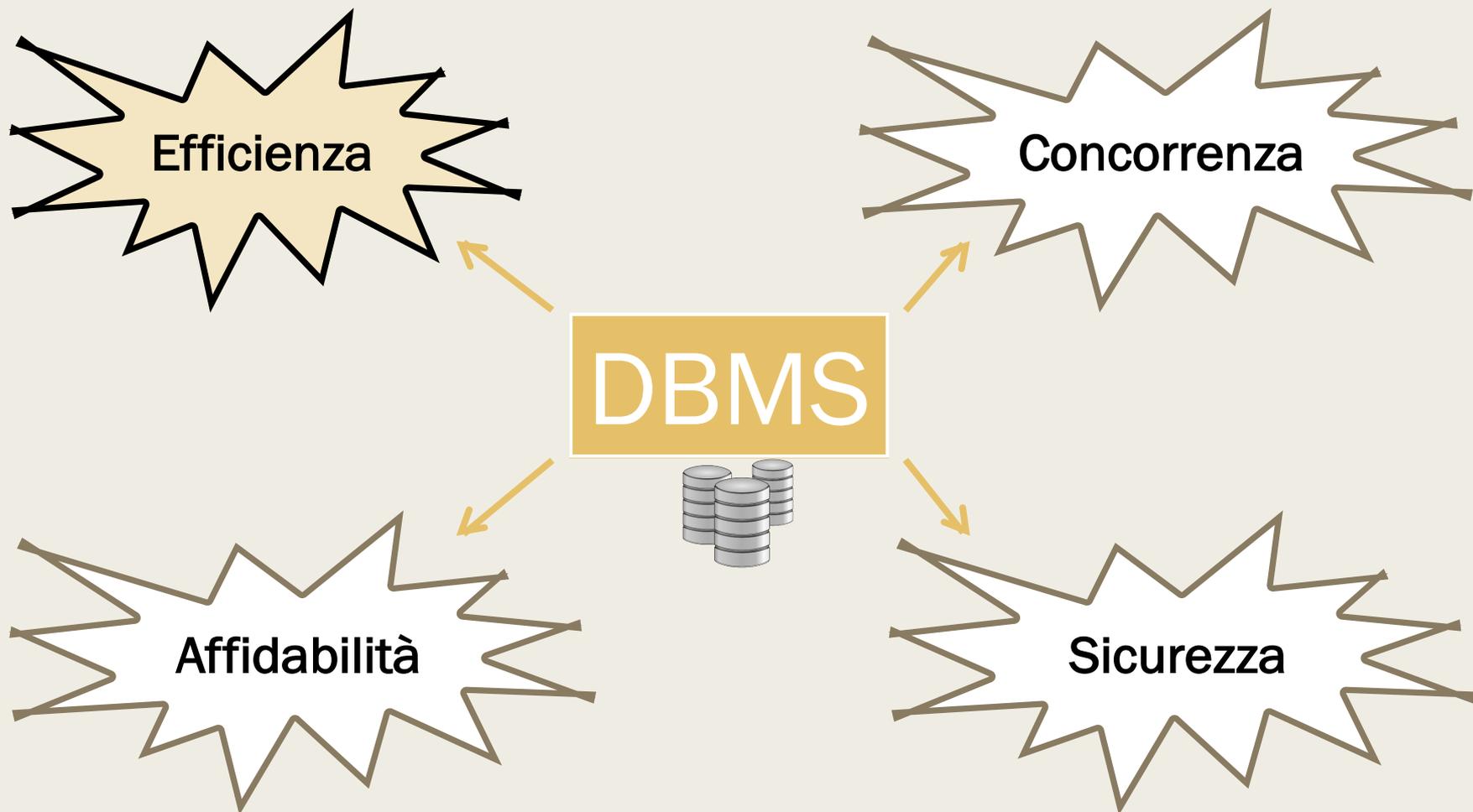


# DBMS: componenti

- I DBMS sono **software complessi** (e tipicamente costosi)
- L'architettura di un DBMS include tipicamente **molte componenti con ruoli diversi**.



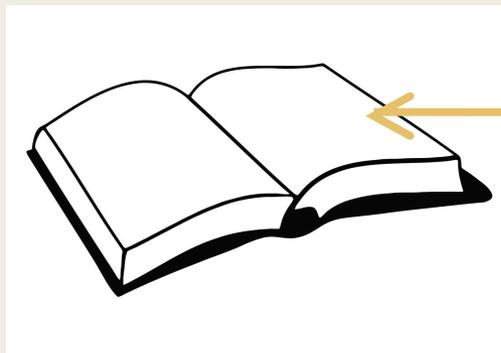
# DBMS: componenti



# DBMS: efficienza

**Indice** → struttura che contiene **informazioni sulla posizione di memorizzazione delle tuple** (righe, sezioni) sulla base del valore del campo **chiave**.

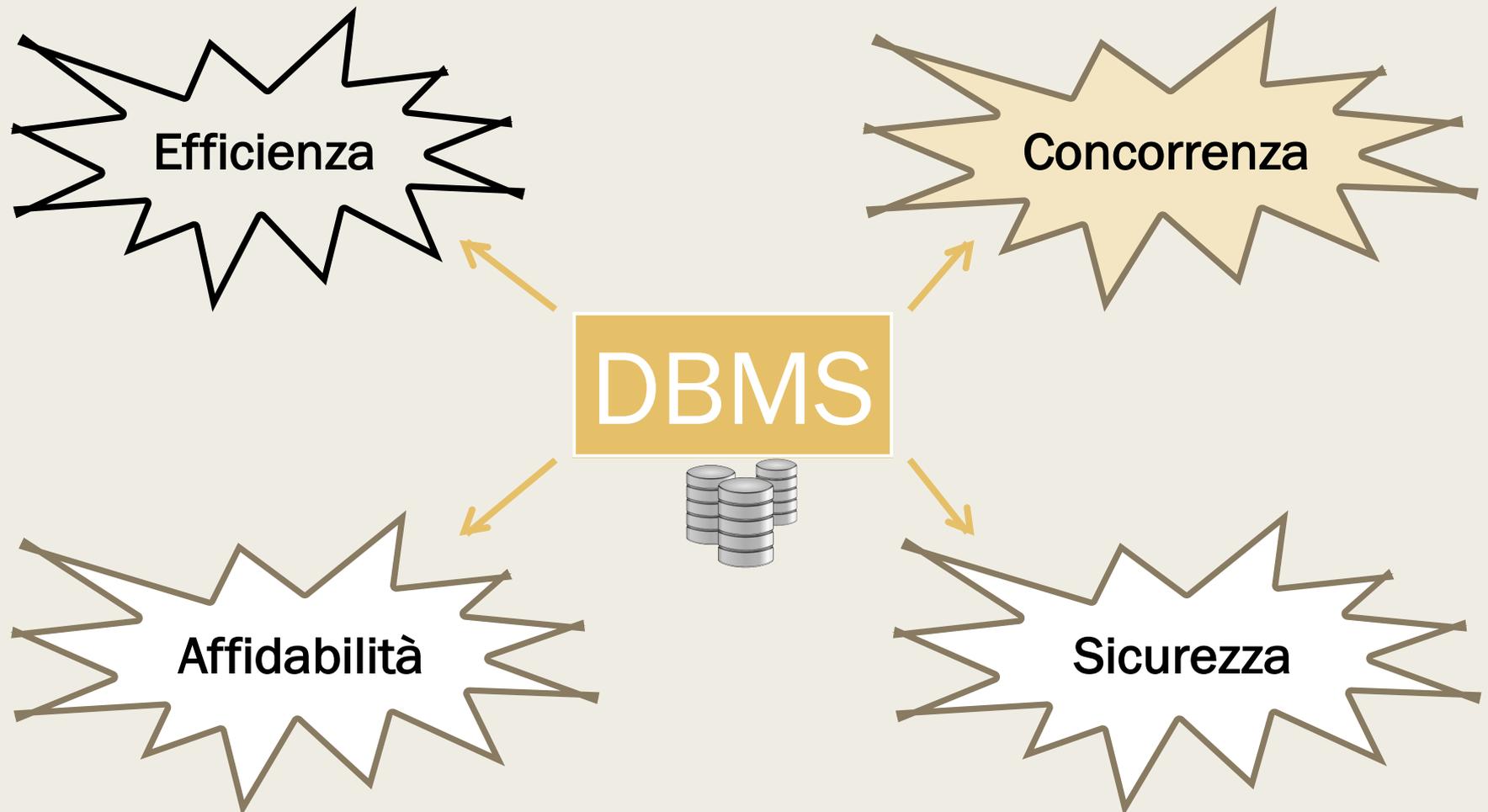
D. A che serve un indice?



ACCESSO DIRETTO

Indice	pag.
Introduzione	1
Capitolo 1	20
<b>Capitolo 2</b>	<b>40</b>
Capitolo 3	60
Conclusioni	65

# DBMS: componenti



# DBMS: concorrenza – 1

- In molti sistemi è fondamentale **gestire operazioni concorrenti** di accesso ai dati ...
- Al tempo stesso, un DBMS deve garantire il fatto che **accessi da parte di applicazioni diverse non interferiscano tra loro**, lasciando il sistema in uno **stato inconsistente** ...

Es. Sistema informativo dei conti bancari

- 2 richieste da gestire al **medesimo tempo t**:
  - Prelievo di **100** euro dal conto **X**
  - Prelievo di **80** euro dal conto **X**
- Saldo del conto X al tempo **t**: **120** euro

# DBMS: concorrenza – 2

## ESEMPIO di ESECUZIONE (non corretta!)

### OP1

Leggi X  
Calcola X – 100  
Scrivi X

### OP2

Leggi X  
Calcola X – 80  
Scrivi X

### Schedula:

OP1: Leggi X  
OP2: Leggi X  
OP1: Calcola X – 100  
OP2: Calcola X – 80  
OP1: Scrivi X  
OP2: Scrivi X

### Valore X:

120  
120  
120  
120  
20  
**40 (????)**

Per prevenire tali situazioni, i DBMS implementano **algoritmi di controllo della concorrenza** che **operazioni sui dati (transazioni) eseguite in concorrenza producano lo stesso risultato di un'esecuzione seriale.**

# DBMS: concorrenza – 3

**Lock Manager** → componente del DBMS responsabile di **gestire i lock alle risorse del DB**, e di **rispondere alle richieste delle transazioni**.

OP1

**Lock(x)**

Leggi X

Calcola X - 100

Scrivi X

**Unlock(x)**

OP2

**Lock(x)**

Leggi X

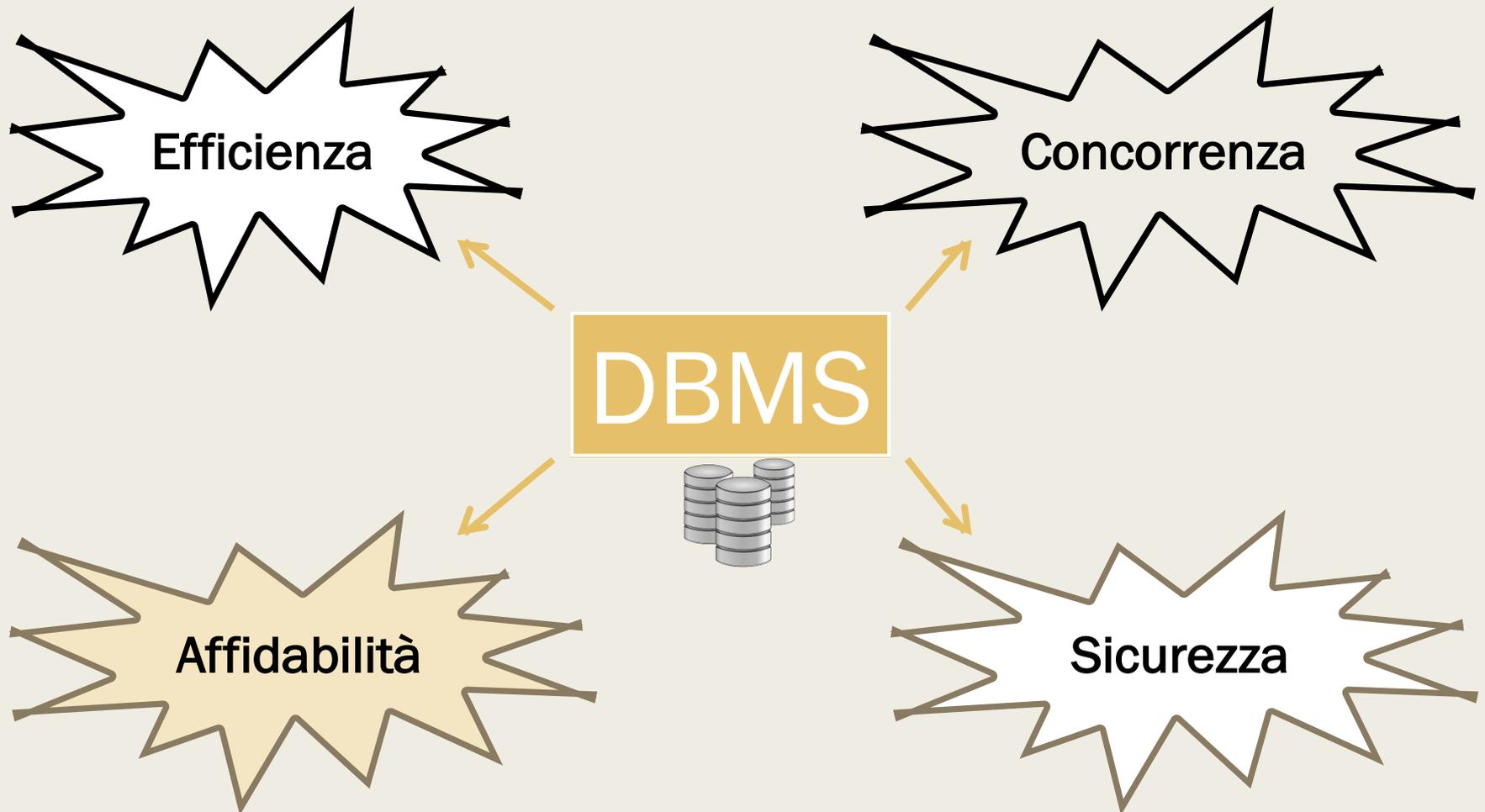
Calcola X - 80

Scrivi X

**Unlock(x)**

- Utilizzo di **lock** in lettura/scrittura per accesso a risorse condivise (dati).
- Algoritmi per gestire ordine di acquisizione dei lock.

# DBMS: componenti



# DBMS: affidabilità; roll-back - 1

- Alcune operazioni sui dati sono **particolarmente delicate**, e devono essere gestite in maniera opportuna, secondo la regola del **tutto o niente**.

Es. Trasferimento di denaro (100 €) dal conto X al conto Y.

Op1:  $X = X - 100$

**CRASH!**

Op2:  $Y = Y + 100$



## DBMS: affidabilità; *roll-back* - 2

- Alcune operazioni sui dati sono **particolarmente delicate**, e devono essere gestite in maniera opportuna, secondo la regola del **tutto o niente**.



- Per questo, i DBMS devono fornire appositi strumenti per **annullare operazioni non completate** e fare **roll-back** dello stato del sistema ...

# DBMS: affidabilità; persistenza – 1

In molti casi i DBMS mettono a disposizione appositi **strumenti ed algoritmi per garantire la persistenza dei dati** anche in presenza di **malfunzionamenti hardware/software**.

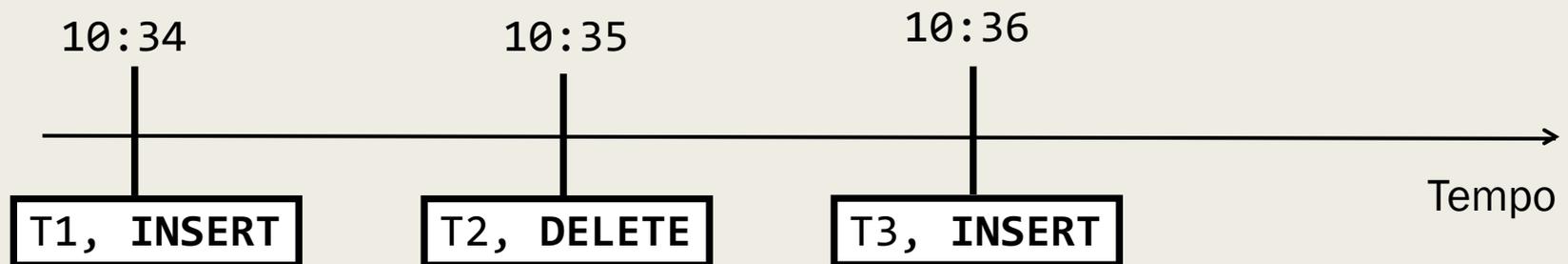
Il controllore di affidabilità utilizza dei **log**, nel quale sono indicate tutte le **operazioni svolte dal DBMS**.

- Algoritmi ad-hoc (es. algoritmo di **ripresa a caldo / a freddo**) per ripristinare lo stato dei dati a partire dai log del DBMS.

## DBMS: affidabilità; persistenza – 2

Il controllore di affidabilità utilizza un **log**, nel quale sono indicate tutte le **operazioni svolte dal DBMS**.

LOG, struttura logica



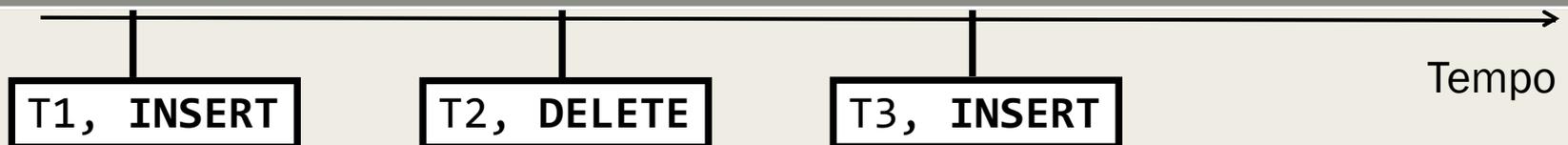
Tramite il log, è possibile fare il **do / undo** delle operazioni...

## DBMS: affidabilità; persistenza – 3

Il controllore di affidabilità utilizza un **log**, nel quale sono indicate tutte le **operazioni svolte dal DBMS**.

LOG, struttura logica

D. Dove/come memorizzare il file di log?



Tramite il log, è possibile fare il **do / undo** delle operazioni...

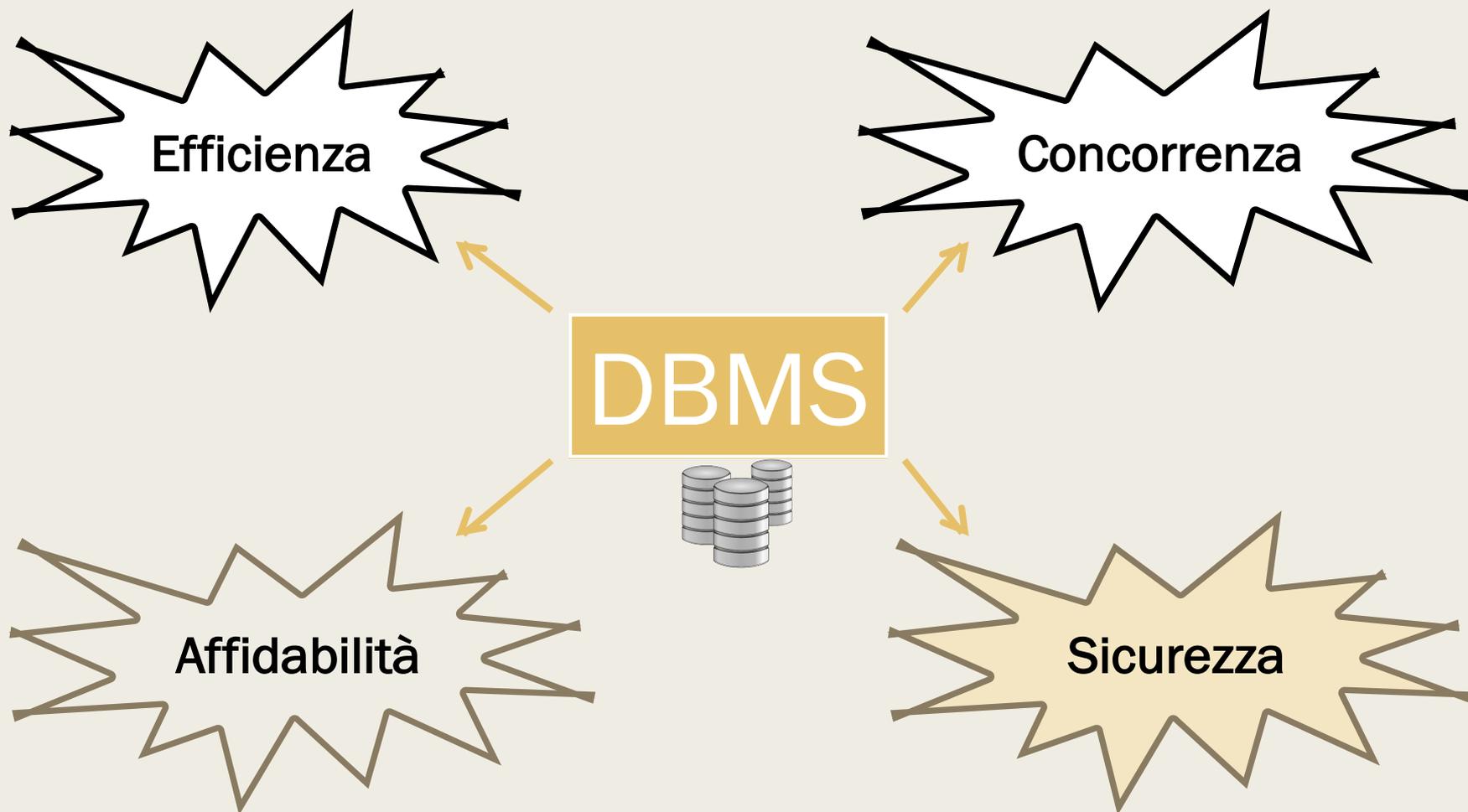
# DBMS: affidabilità; persistenza – 4

**Transazione** → insieme di operazioni (lettura/scrittura) eseguite su un DB dal DBMS.

Proprietà **ACID** di un sistema transazionale

- **Atomicità** → La transazione deve essere eseguita con la regola del “**tutto o niente**”.
- **Consistenza** → La transazione deve lasciare il DB in uno stato **consistente**, vincoli di integrità sui dati non devono essere violati.
- **Isolamento** → L'esecuzione di una transazione deve essere **indipendente** dalle altre.
- **Persistenza** → L'effetto di una transazione conclusa con successo non deve essere perso.

# DBMS: componenti



# DBMS: sicurezza; multi-utenza

La maggior parte dei DBMS implementa **politiche di controllo degli accessi** ad i dati mediante **sistemi di permessi**:

- Quali **operazioni sono consentite** all'utente X?
- Quali **dati appartengono** all'utente X?



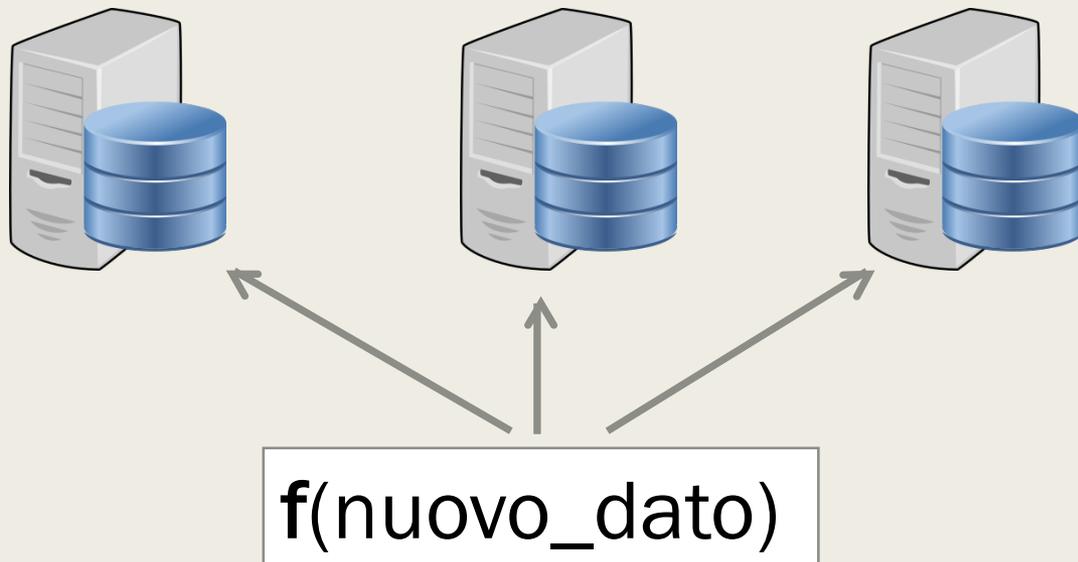
Utente	OPERAZIONE	DATO	PERMESSO
Utente X	Lettura	Stipendio di X	Consentito
Utente X	Lettura	Stipendio di Y	Consentito
Utente Y	Scrittura	Stipendio di Y	Negato

# DBMS: sicurezza; scalabilità – 1

## Scalabilità orizzontale

Possibilità di gestire grandi moli di dati aumentando il numero di nodi usati per lo *storage*

→ **database distribuito.**



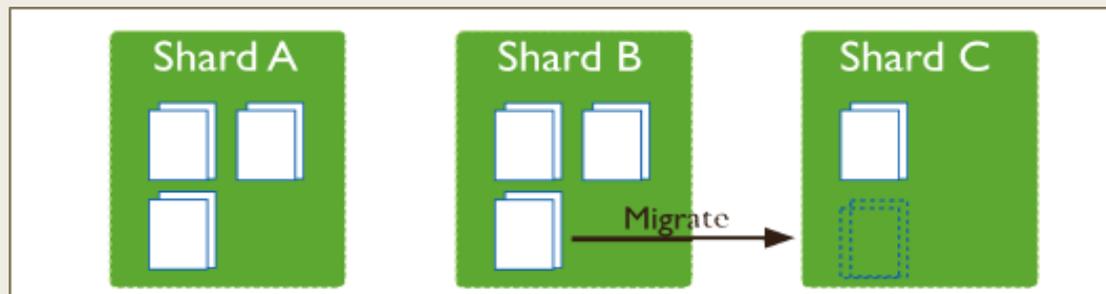
La funzione di **sharding** determina la politica di distribuzione dei dati tra i nodi

# DBMS: sicurezza; scalabilità – 2

## Scalabilità orizzontale

Ulteriori funzionalità di un database distribuito.

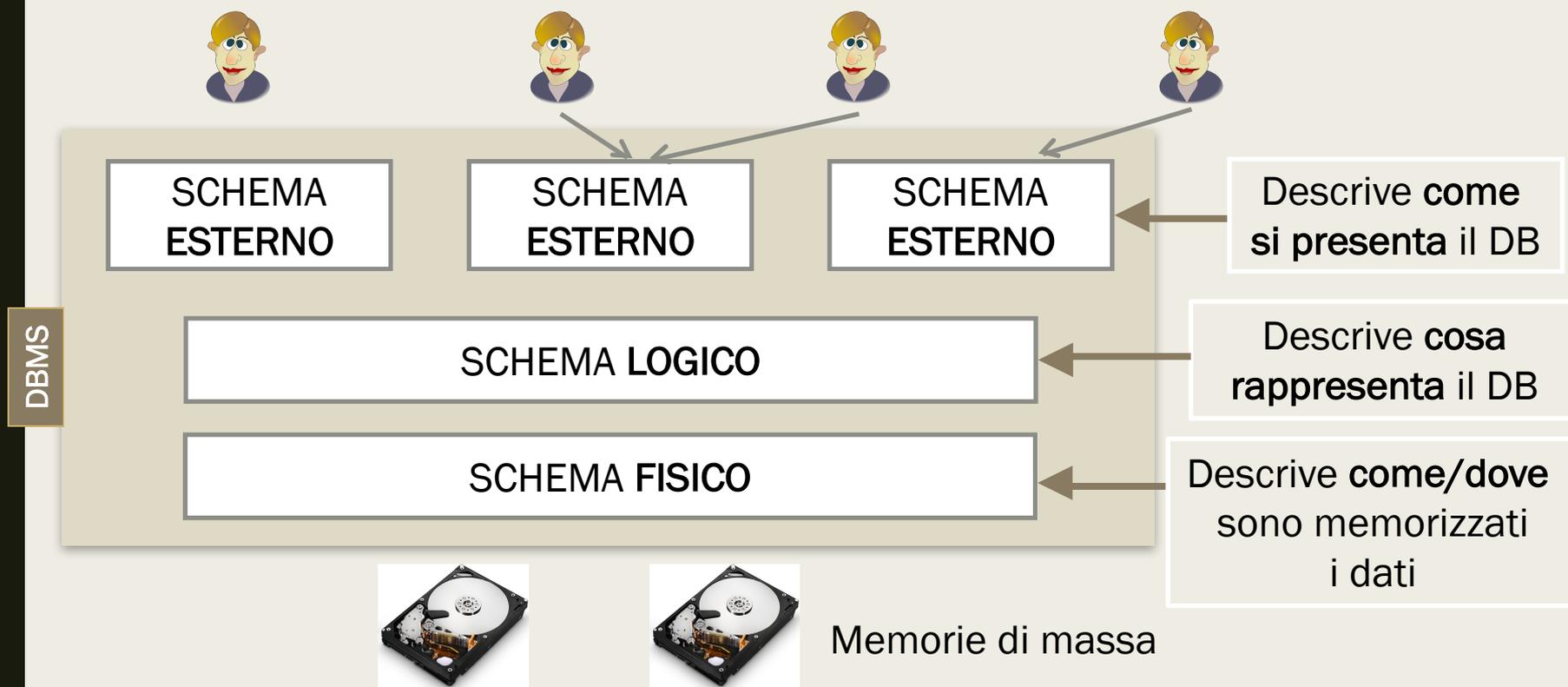
1. Meccanismi di **load-balancing**
2. Meccanismi di **gestione delle repliche** dati



**PROBLEMA.** Come gestire la **consistenza** delle repliche dati in presenza di partizionamenti della rete e perdita di messaggi? **CAP Theorem** ...

# DBMS: architettura a 3 livelli

In pratica, un DBMS può essere visto come una **architettura software** a 3 livelli ...



# DBMS: livello logico - 1

I DBMS forniscono un approccio **strutturato** ai dati.



???

In un DBMS, i dati sono organizzati secondo un **modello logico**, che definisce i concetti rappresentati, le associazioni dei dati, i vincoli che questi devono rispettare.

In pratica, l'utente/applicazione interagisce con i dati del DBMS sulla base del modello logico.

## DBMS: livello logico – 2

- Sono stati proposti **diversi modelli logici**
- I DBMS possono differire sulla base del modello logico dei dati che supportano:
- Modello **Relazionale** (di fatto, il più usato)
  - Modello Gerarchico
  - Modello Reticolare
  - Modello ad Oggetti
  - Approcci NoSQL (diversi)

# DBMS: livello logico – 3

## Esempio: Modello Relazionale

Base di dati che gestisce le informazioni relative alla programmazione didattica di un Corso di Laurea: elenco corsi, con numero ore, semestre, crediti, nome e codice identificativo di ciascun corso.

Codice	Nome	Num.Ore	Semestre	Crediti
010	Basi di Dati	72	1	9
001	Algoritmi	90	1	12

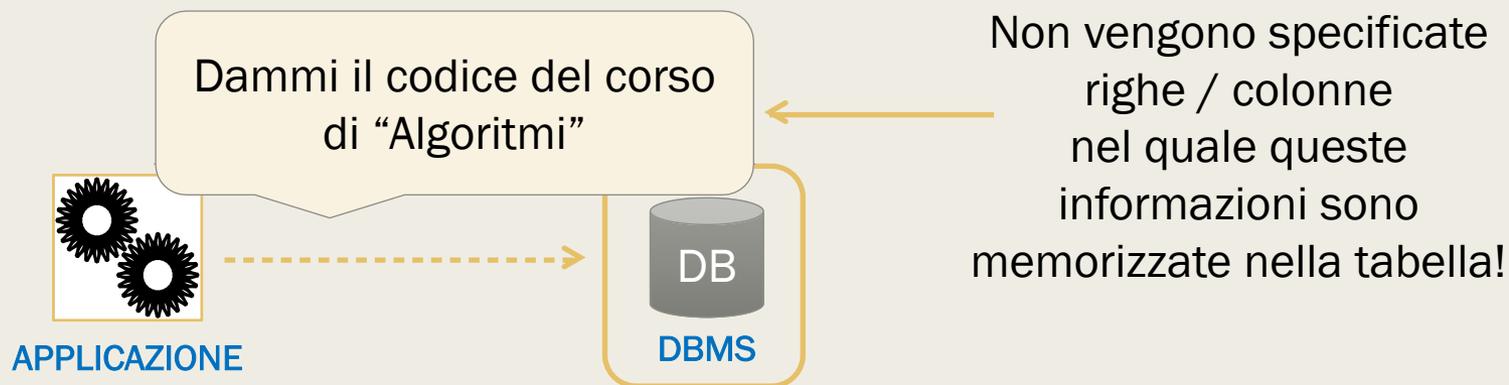
CAMPI, COLONNE  
SCHEMA, RELAZIONE, TABELLA  
ISTANZE, RIGHE

Nel modello relazionale, i dati sono organizzati in **tabelle** ...  
**vedremo meglio in seguito!**

# DBMS: livello logico – 4

## Indipendenza **Modello Logico** – **Modello Fisico**

- L'organizzazione logica dei dati **non dipende** dalle strutture dati usate per l'effettiva memorizzazione su disco!
- In pratica, **le applicazioni accedono al DBMS specificando concetti logici** del modello dei dati, piuttosto che i dettagli relativi alla loro memorizzazione.



# DBMS: livello esterno

Il livello esterno consente di avere **viste** personalizzate della base di dati **da parte di diversi utenti / applicazioni!**

Esempio: base di dati condivisa tra diversi uffici di una stessa organizzazione. Solo alcuni uffici possono accedere a tutto il contenuto del DB!

Codice	Nome	Cognome	Data Nascita	Livello	Stipendio
001	Marco	Rossi	10/10/1970	1	24000
002	Michele	Bianchi	13/12/1977	1	32000

Vista Ufficio Anagrafe

Vista Ufficio Stipendi

# DBMS: interazione – 1

Come possono utenti ed applicazioni interagire con un DBMS?

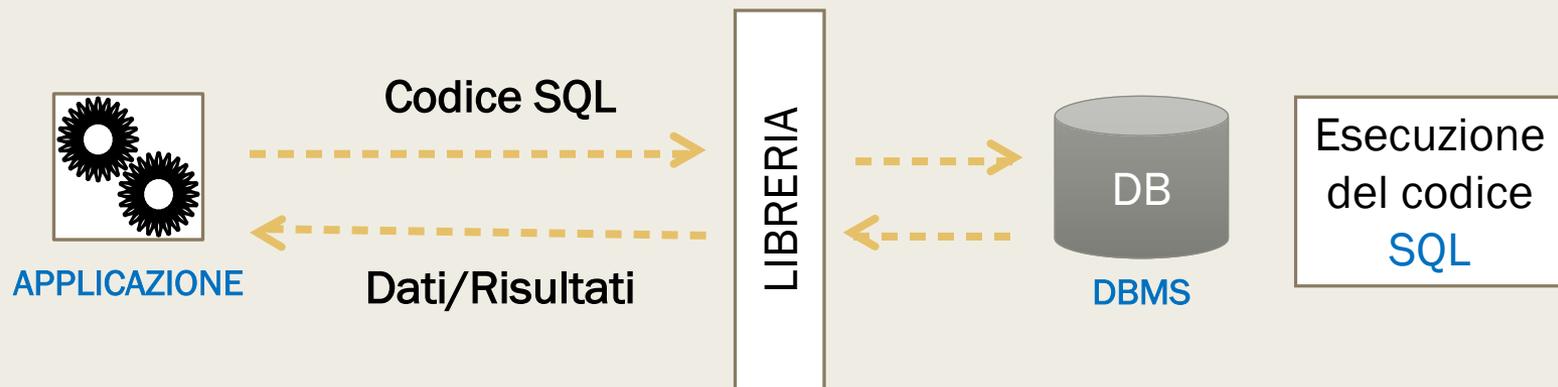
Quasi tutti i DBMS mettono a disposizione dei linguaggi:

- Definizione dello schema logico → Linguaggio **DDL**
- Manipolazione delle istanze → Linguaggio **DML**
- Linguaggi **orientati ai dati**, molto diversi da linguaggi di programmazione “tradizionali” (es. C / C++ / Java / ecc.)!
- Noi vedremo il linguaggio (DDL + DML) **SQL**

## DBMS: interazione - 2

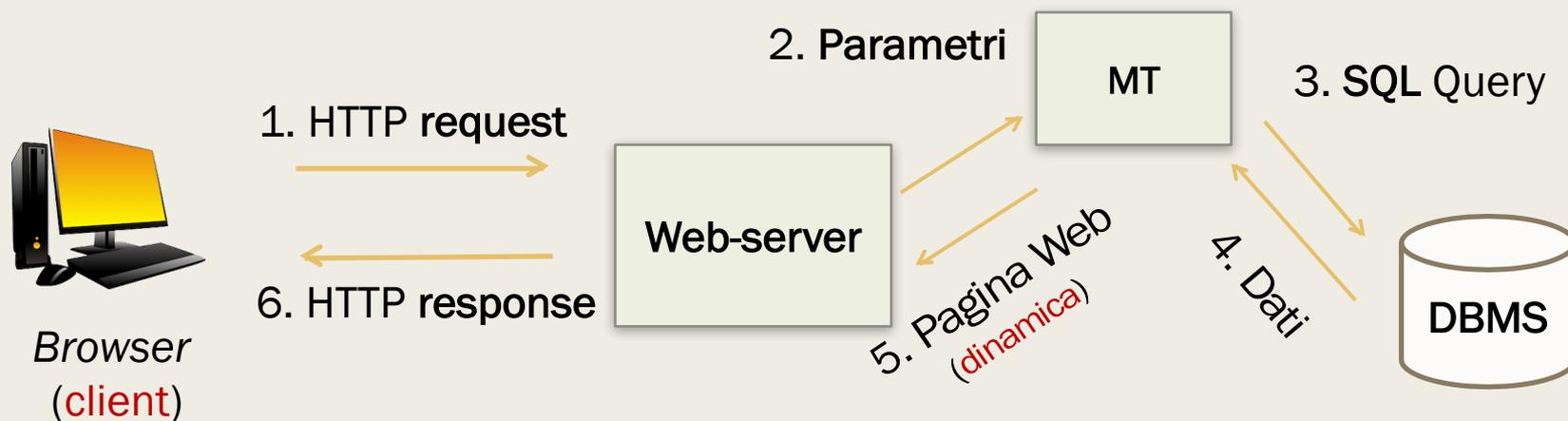
Le applicazioni che si interfacciano con un DBMS:

- integrano **codice SQL** all'interno del loro codice
- utilizzano opportune **librerie** (fornite dal DBMS) per gestire la connessione al DBMS.



# DBMS: interazione – 3

Un esempio di modello integrato DMBS/App molto in voga: **Web Information System (WIS)**



➤ Esempio: Architettura **AMP** (Apache + MySQL + PHP)

# DBMS: riassumendo - 1

## Quando **usare** un DBMS in un progetto SW?

- Necessità di gestire **grandi volumi di dati**
- Necessità di costruire **sistemi data-centric** con molte operazioni di accesso ai dati
- Necessità di **condividere dati**, fornendo l'accesso a diversi sistemi SW/applicazioni
- Necessità di garantire la **persistenza dei dati** anche a fronte di possibili guasti e malfunzionamenti HW/SW
- Necessità di implementare **meccanismi di sicurezza** per l'accesso ad i dati in un ambiente multi-utente

## DBMS: riassumendo – 2

### Quando **NON usare** un DBMS in un progetto SW?

- **Prestazioni** → In alcuni sistemi con richieste di efficienza sull'elaborazione (es. **real-time**), l'*overhead* computazionale introdotto dal DBMS può essere eccessivo ...
- **Costo** → Spese per l'acquisto di DBMS, formazione del personale, amministrazione del DB, ecc.
- **Complessità** → Applicazioni/sistemi di dimensioni ridotte, *single-user* e con pochi dati da gestire ...

# DBMS: quale usare?

- Fino ad ora abbiamo parlato in generale delle caratteristiche dei DBMS ...
- ... ma i **DBMS** sono tutti uguali? **NO!**

**Differenze sostanziali, ad esempio in termini di:**

- Modello logico supportato (relazionale? → RDBMS)
- Linguaggio DDL/DML (SQL-2? SQL-3? varianti?)
- Algoritmi di indicizzazione (es. R+ tree?)
- Supporto alla transazioni (es. proprietà ACID?)
- Gestione della concorrenza
- ...

# DBMS: i più usati *open source*

## TOP 5 – Ottobre 2017

Posiz.	Sistema	Punteggio	Posiz. generale
1.	MySQL	1299	2.
2.	PostgreSQL	373	4.
3.	MongoDB	329	5.
4.	Cassandra	125	8.
5.	Redis	122	9.

Fonte DB-Engines.com

# DBMS: tendenze

- Oggi, una delle nuove linee evolutive dei DBMS è rappresentata dall'approccio **NoSQL**.
- **Idea di base:** superare la rigidità del modello relazionale nella definizione dello schema, consentendo una più facile espansione del DB in termini di dati, e di computazione distribuita.
- Molti approcci sotto la definizione NoSQL:  
Es. **Apache Cassandra, Apache CouchDB, ...**

# Vediamo cosa andremo a fare

<http://ross.iasfbo.inaf.it/opusfacere/>

- [index.html](#)
  - Pagina di query + visione tabella
- [plot.html](#) e [histo.html](#)
  - Pagina di query + visione grafico “scatter plot” e istogramma
- [cielo.html](#)
  - Pagina di query + visione sulla sfera celeste
- [stile.css](#)
  - File di stile
- [ajax\\_query.js](#)
  - Funzioni di trasferimento dei dati dal DB server alla pagina web
- [query\\_bsc.php](#)
  - Programma per effettuare la **query locale** sul DB server

[Altra visualizzatore del catalogo di stelle brillanti \(BSC\)](#)