# Fondamenti di Informatica

## CAP. 6

Accademia di belle Arti di Verona

Università degli Studi di Verona

A.A. 2022-2023

Docente - Vincenzo Giannotti

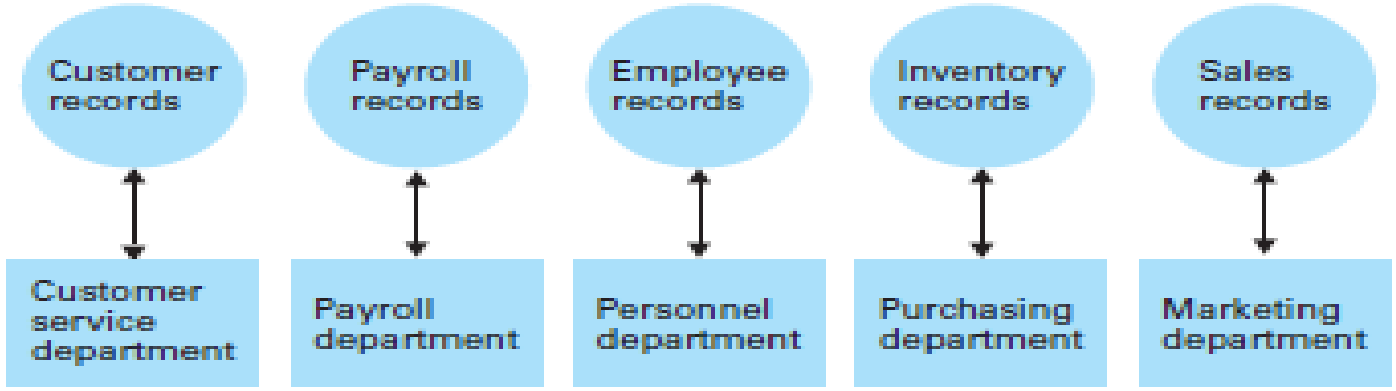
# CAPITOLO 6 – BASI DI DATI

# Basi di dati

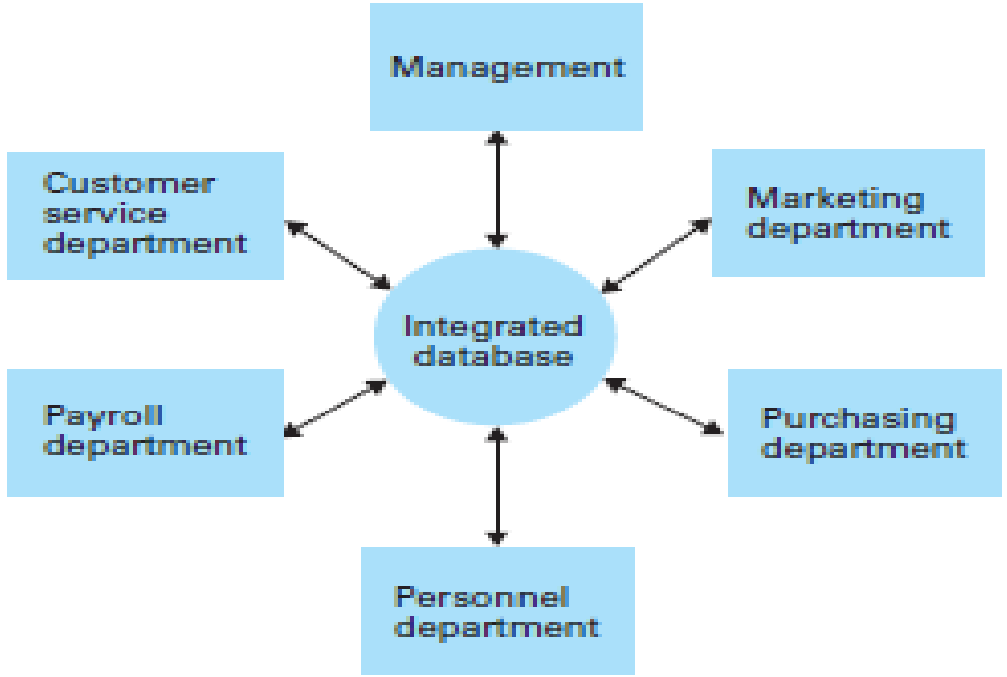
Il termine «Base di Dati» viene comunemente associato al termine inglese «**Database**», col quale ci si riferisce a una collezione di dati organizzati in maniera che la loro ricerca e il loro accesso possa avvenire in maniera efficiente secondo diverse modalità e utilizzando delle applicazioni dedicate.

Diversamente dal «**file**» che è un sistema di archiviazione delle informazioni di tipo mono-dimensionale, ossia in grado di rappresentarli secondo un unico punto di vista, il data base è un sistema multi-dimensionale che consente di rappresentare i dati in diversi modi e secondo diversi punti di vista.

a. File-oriented information system




b. Database-oriented information system




# Basi di Dati

Nella gestione dei dati per **File** (che spesso avviene nelle piccole organizzazioni) le analisi avvengono attraverso la comunicazione e lo scambio di informazioni tra le persone.

La gestione con **Data Base**, viceversa, consente di automatizzare molte attività di analisi dei dati, mettendo in relazione, per esempio, i risultati dei diversi dipartimenti aziendali e consentendo una condivisione profonda delle informazioni.

# Archiviare i dati in file o in database?

**Struttura**: I file hanno una struttura piatta, mentre i database hanno una struttura logica con tabelle, righe e colonne che relazionano i dati. Il database impone una struttura organizzata ai dati, mentre i file hanno una struttura più libera.

**Integrità dei dati**: I database applicano vincoli e relazioni che garantiscono l'integrità dei dati (nessun valore NULL, chiavi primarie univoche, etc.), mentre i file hanno una struttura più lasca e i dati possono essere corrotti o inconsistenti.

**Ridondanza**: I database eliminano la ridondanza dei dati grazie alle relazioni, mentre i file possono contenere dati ridondanti memorizzati in più punti.

**Interrogazioni**: È più facile recuperare e filtrare i dati da un database utilizzando interrogazioni SQL. I file devono essere esaminati manualmente riga per riga.

**Sicurezza**: I database offrono controlli di accesso granulari e funzioni di sicurezza a livello di riga e colonna. I file hanno controlli di accesso relativamente semplici a livello di file.

**Concorrenza**: Più utenti possono modificare contemporaneamente i dati in un database, in maniera controllata, mentre i file devono essere bloccati mentre li modifica un singolo utente per evitare la sovrascrittura.

**Aggiornamento**: È più facile aggiornare i dati in un database senza dover riscrivere completamente il file. Basta aggiornare le righe pertinenti.

**Backup**: I database offrono funzioni di backup integrate a livello di base di dati. I file richiedono Backup manuale.

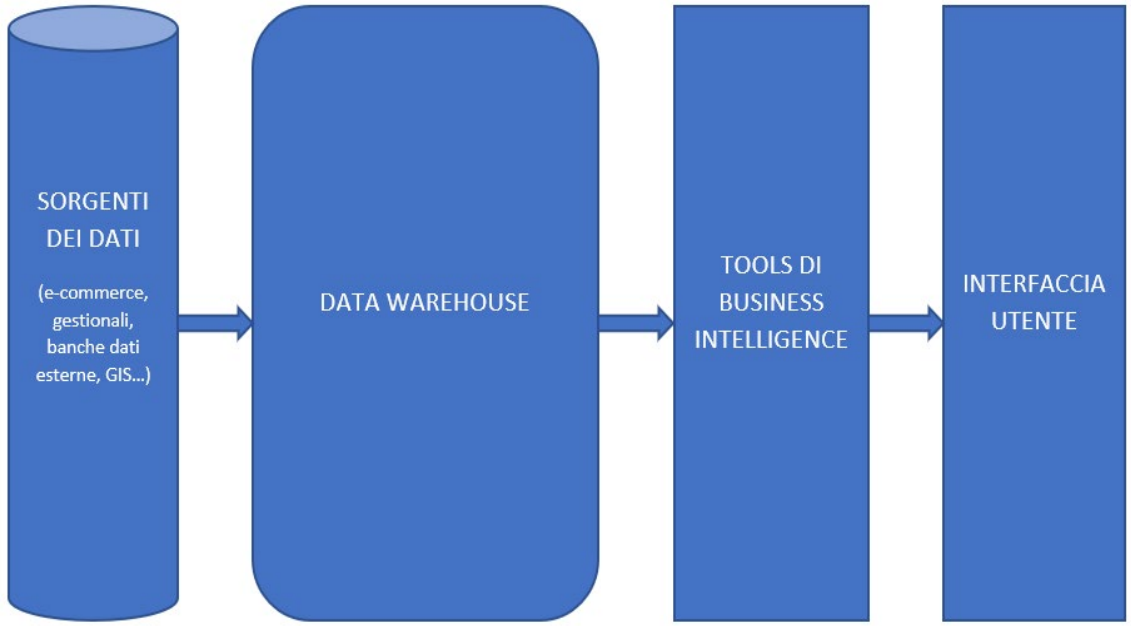
In conclusione, i database sono molto più strutturati e sofisticati dei semplici file. Mentre i file sono adatti per archiviare dati grezzi, i database sono la scelta migliore per archiviare e gestire dati relazionali complessi in modo organizzato.

# Basi di Dati

Col termine **Data Warehouse** (DW) si intende una aggregazione di dati strutturati, provenienti da fonti diverse, interne ed esterne al sistema informativo aziendale. Può essere visto come una grande banca dati nella quale si opera in sola lettura.

Una possibilità di applicazione del DW avviene nell'ambito dei processi di **Business Intelligence*** (BI) dove i dati sono analizzati per un uso strategico aziendale nei processi decisionali di impresa.

Una tecnica di analisi è quella del **Data Mining** dove l'esplorazione e l'analisi dei dati viene eseguita in maniera automatica o semiautomatica, su grandi quantità, con lo scopo di individuare pattern, schemi o regolarità significativi

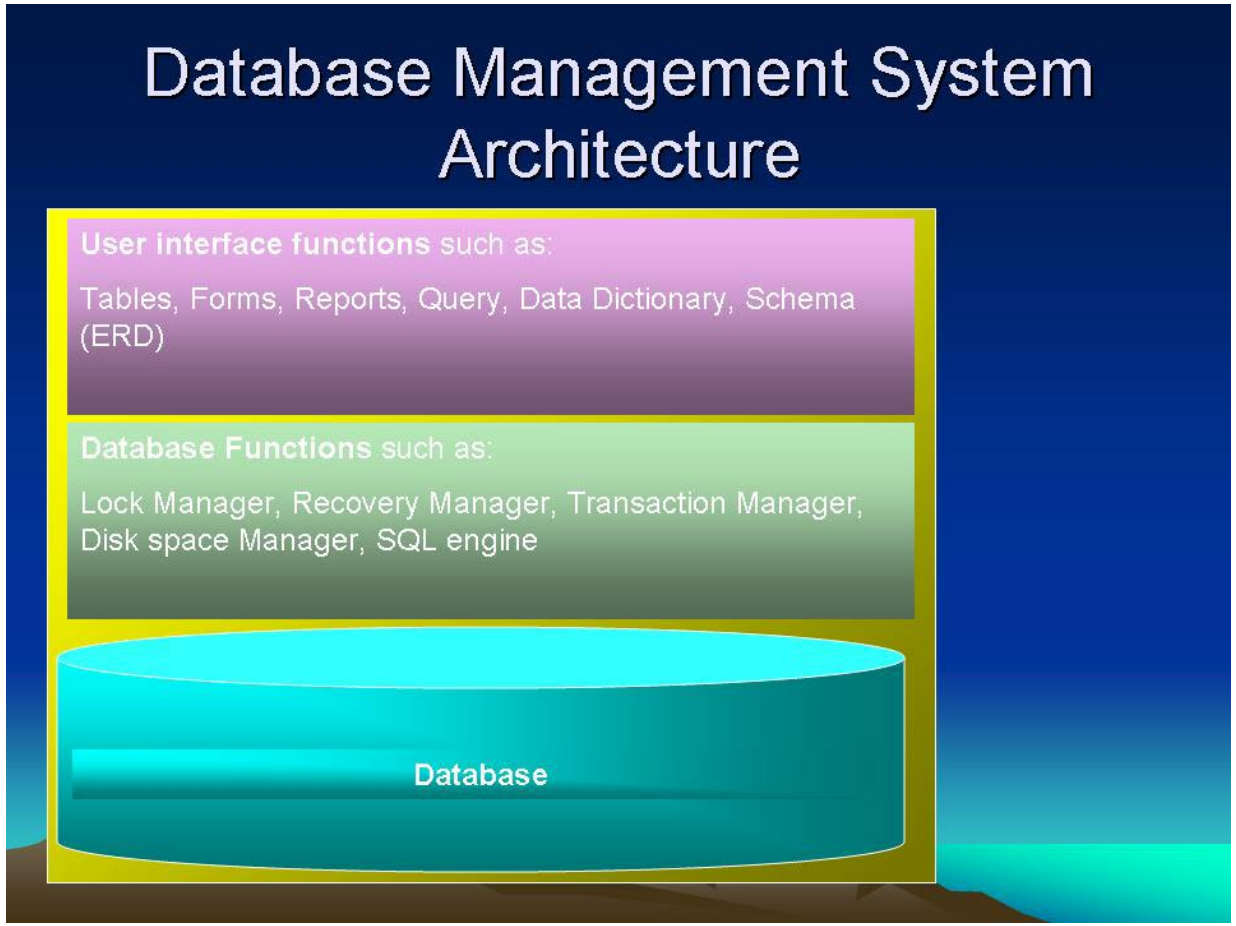



* In letteratura la *business intelligence* viene citata come il processo di "*trasformazione di dati e informazioni in conoscenza*".

# Il Data Base Management System (DBMS)

Come detto, un database è una collezione di dati organizzati in maniera che la loro ricerca e il loro accesso possa avvenire in maniera efficiente secondo diverse modalità.

Lo strumento che consente di effettuare queste operazioni è il «**DataBase Management System**» (DBMS), un sistema software progettato per consentire la creazione, la manipolazione e l'interrogazione di una collezione di dati strutturati. Spesso col termine «database» ci si riferisce sia al DBMS che alla collezione di dati da esso gestita.



# Il Data Base Management System (DBMS)

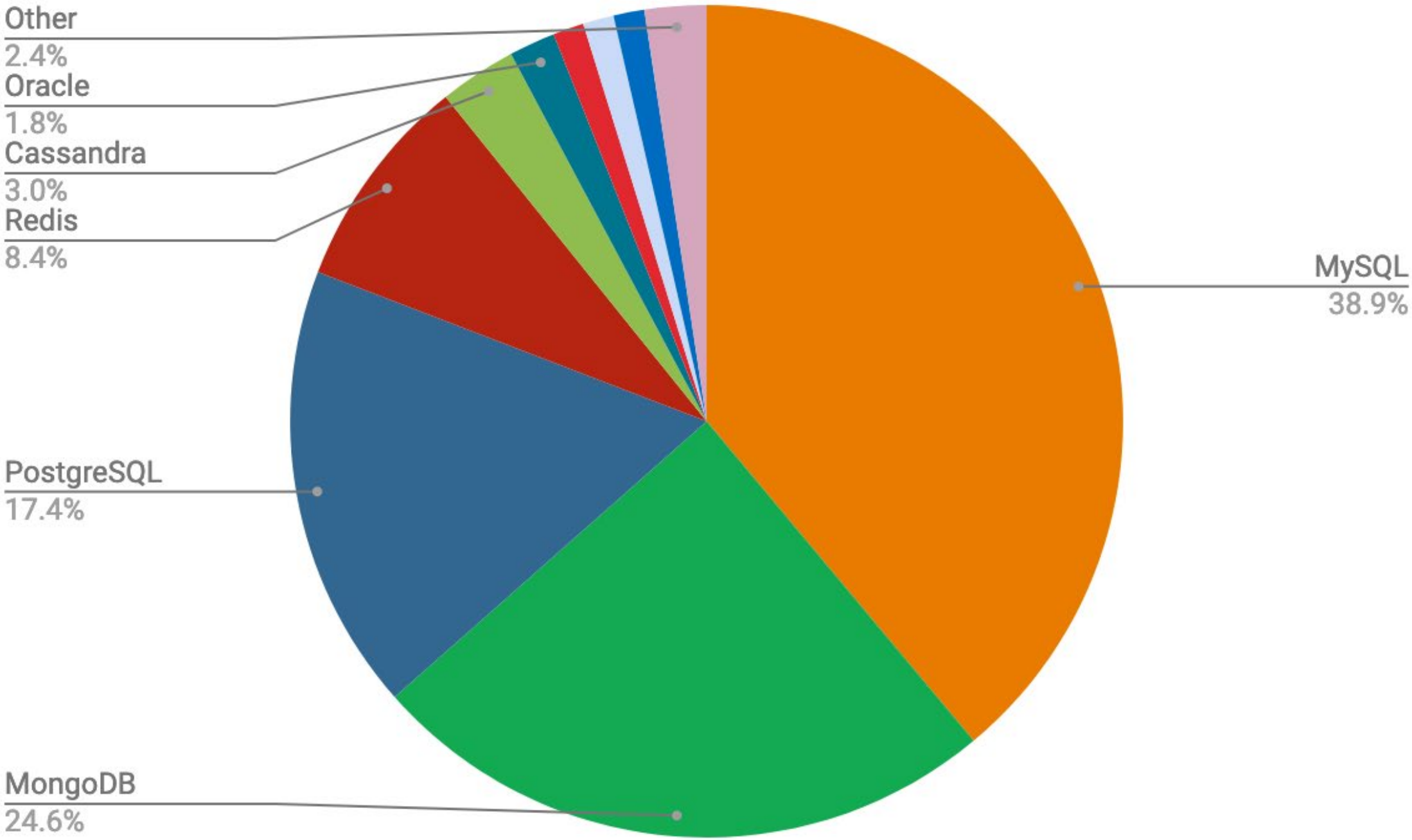
Il DBMS è dunque una applicazione software che normalmente possiede le seguenti caratteristiche di gestione dell'informazione:

- È in grado di gestire collezioni organizzate di dati di grandi dimensioni. Con l'affermazione «di grandi dimensioni» intendiamo collezioni di dimensioni molto maggiori di quelle che sono normalmente gestibili nella memoria centrale del computer
- È in grado di gestire informazioni «persistenti», le quali hanno un ciclo di vita del tutto indipendente dalle applicazioni che le utilizzano (che possono essere molte, anche contemporaneamente: pensiamo ai DB di Google)
- Poiché i dati persistenti esistono separatamente dalle applicazioni, non c'è bisogno di copiare i dati in più applicazioni. Esiste solo una copia affidabile dei dati. Questa si chiama «ridondanza ridotta».
- È capace di trattare informazioni «condivise» che possono quindi essere utilizzate e manipolate da applicazioni diverse.

# Il Data Base Management System (DBMS)

Inoltre il DBMS deve possedere le seguenti ulteriori caratteristiche di garanzia, strettamente connesse alla sicurezza e alle performance:

- È «**affidabile**». Ciò significa che è in grado di resistere ad eventuali malfunzionamenti del sistema sia dal punto di vista Hardware che Software.
- Garantisce la «**privacy**» dell'informazione, quando richiesta, con adeguati strumenti e procedure di controllo degli accessi e verifica delle identità.
- È «**efficiente**», ossia è in grado di ottimizzare l'utilizzazione delle risorse (in particolare di memorie a di tempi di CPU) che gli vengono messe a disposizione dal sistema operativo.
- È «**efficace**» e questo significa che riesce effettivamente ad aumentare la produttività di chi lo utilizza.


Other
2.4%
Oracle
1.8%
Cassandra
3.0%
Redis
8.4%
MySQL
38.9%
PostgreSQL
17.4%
MongoDB
24.6%

# Il Database Relazionale (SQL)

Nel **modello relazionale**, che è il modello di database più comune e fu introdotto nel 1970, tutto ruota attorno al concetto di **tabella**. Ne viene creata una per ogni tipo di informazione da trattare; per esempio, se vogliamo realizzare un archivio di libri: la tabella dei libri, quella degli autori, quella dei generi etc…

A sua volta ogni tabella è costituita da **colonne** (campi) che descrivono quel tipo di informazione: la tabella dei libri per esempio conterrà le colonne titolo, autore, genere, data di pubblicazione, editore etc…; a sua volta quella degli autori conterrà nome, cognome, data di nascita e di morte, luogo di nascita e così via.

Le **righe** della tabella (record) sono invece le istanze, ossia, per esempio, tutti i diversi libri che ho catalogato coi loro propri dati associati.

# Il Database Relazionale (SQL)

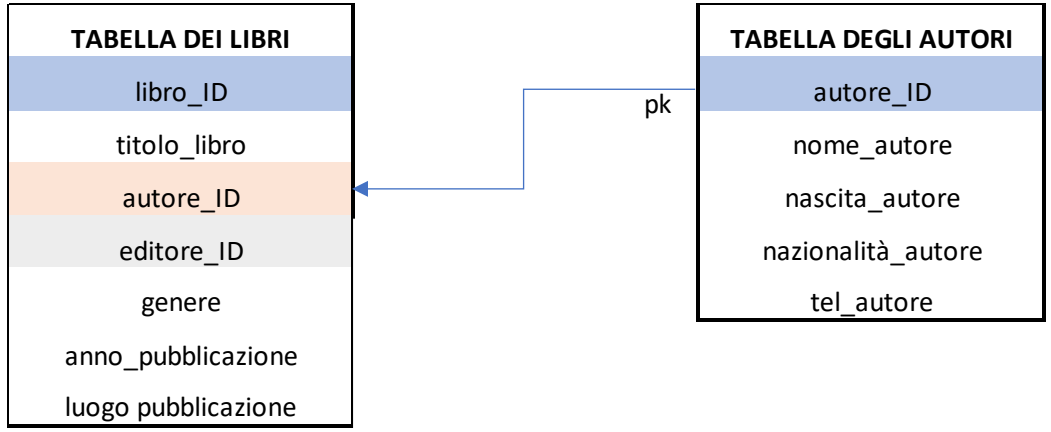
Nel database relazionale le tabelle possono essere messe in **relazione** tra loro. Questo significa che una riga della tabella 1 può far riferimento ad una riga della tabella 2.

Supponiamo che la prima tabella sia relativa alle informazioni sui libri. Ogni record include pertanto il titolo, l'autore, l'editore etc. ossia i diversi attributi di un dato libro. Assegniamo un ID univoco (una chiave) a ciascun autore.

Nella seconda tabella, relativa agli autori, ogni record include l'ID dell'autore, i suoi dati anagrafici etc.

Ora vediamo che queste due tabelle hanno un elemento in comune: la colonna autore_ID, ossia la chiave.

Grazie a questa chiave il software è in grado di creare una relazione tra le due tabelle, col vantaggio, tra gli altri, che quel determinato autore sarà scritto una volta sola nel database ma potrà essere richiamato n volte in associazione a diversi libri, cioè tutti quelli che lui ha scritto.



# Il Database Relazionale (SQL)

Queste operazioni di **join** consentono di combinare i dati da due o più tabelle in base alle chiavi utilizzate, creando una vista unica dei dati. Per ottenere queste viste si utilizza il linguaggio di interrogazione **SQL (Structured Query Language)** che consente di interrogare e manipolare i dati in modo flessibile ed efficiente, eseguendo query complesse tramite le quali vengono restituite solo le informazioni desiderate.

```
select titolo_libro
from tabella dei libri
where anno_pubblicazione < «anno»
order by «criterio»
```

# Il Database Non Relazionale (NoSQL)

È importante notare che i database relazionali non sono l'unica opzione disponibile per la gestione dei dati. Negli ultimi anni, sono emerse molte alternative ai database relazionali, come **i database NoSQL**, che utilizzano modelli di dati non relazionali. Questi modelli sono progettati per gestire grandi quantità di dati non strutturati, come i dati provenienti dai social media o dalle applicazioni mobili.

Nei database NoSQL la mancanza di strutturazione rigida dei contenuti, comporta alcuni vantaggi e alcuni svantaggi:

- Alcuni vantaggi rispetto ai database relazionali sono una maggiore scalabilità e flessibilità. Ad esempio, i database a documenti possono essere facilmente scalati aggiungendo nuovi server e distribuendo i dati tra di essi. Inoltre, i database NoSQL sono spesso utilizzati in ambienti cloud in cui la **scalabilità orizzontale** (aggiunta di nuovi server) è una necessità. Hanno normalmente **prestazioni migliori** data l'architettura più semplice e sono più flessibili poiché è più facile modificarne lo schema dati.
- I database NoSQL hanno anche alcuni svantaggi rispetto ai database relazionali. In primo luogo, i database NoSQL non supportano le transazioni **ACID** (Atomicità, Coerenza, Isolamento, Durabilità), che sono importanti per garantire l'integrità dei dati. In secondo luogo, l'assenza di uno schema fisso può rendere più difficile la query dei dati.

In conclusione quella NoSQL è una soluzione ideale quando abbiamo a che fare con dati scorrelati e che tendono ad evolvere nel tempo (come spesso accade nel Web).

Le transazioni **ACID** sono un concetto chiave per l'integrità dei dati in un database. ACID è l'acronimo di:

- A - **Atomicità**: Una transazione è un'unità di lavoro logica che deve essere eseguita completamente o non deve essere eseguita affatto. Non devono esserci stati parziali.
- C - **Consistency**: Una transazione può portare il database da uno stato consistente a un altro stato consistente. Non deve violare i vincoli di integrità.
- I - **Isolation**: Le transazioni concorrenti devono essere isolate le une dalle altre. Gli aggiornamenti di una transazione non devono interferire con altre transazioni in esecuzione contemporaneamente.
- D - **Durability**: Gli effetti di una transazione devono essere permanenti. Una volta che una transazione viene avviata, gli aggiornamenti devono persistere anche in caso di guasto.

In breve, le transazioni ACID sono fondamentali per gestire il database in modo coerente e affidabile. Alcuni esempi concreti:

- Un trasferimento bancario è una transazione. Deve essere atomica (tutta o nulla) e consistente (soldi sufficienti, soldi dedotti da un conto e aggiunti all'altro).
- Aggiornare l'inventario di prodotti è una transazione. Deve assicurare che l'inventario rimanga sempre consistente e positivo.
- Due utenti che modificano le informazioni su un ordine di acquisto devono essere isolati l'uno dall'altro. L’esecuzione di una transazione non deve influire sull'altra transazione in corso.
- Gli effetti di tutte le transazioni effettuate devono essere permanenti e non possono essere persi in caso di guasto del sistema.

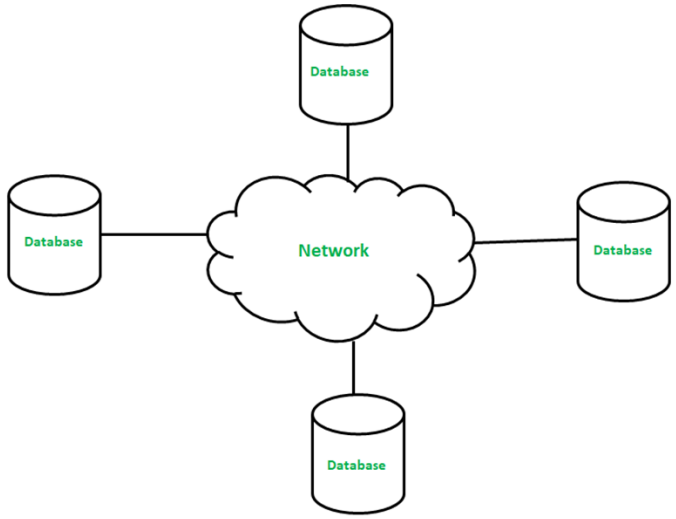
# Database distribuiti

Un **database distribuito** è un database che si trova sotto il controllo di un DBMS nel quale gli archivi di dati sono memorizzati su diversi computer anche fisicamente molto distanti tra loro ma **interconnessi in rete** così da lavorare insieme come se fossero una singola entità. Questi database sono progettati per gestire grandi quantità di dati e fornire una maggiore scalabilità, disponibilità e ridondanza rispetto ai database centralizzati.

Ci sono diversi tipi di architetture di database distribuiti che possono funzionare utilizzando la replica dei dati oppure la frammentazione dei dati. La **replica dei dati** consiste nella creazione di copie di dati in diverse macchine in modo che i dati siano disponibili in più punti della rete. Ciò garantisce una maggiore disponibilità e ridondanza dei dati, ma richiede una sincronizzazione costante tra le diverse copie. La **frammentazione dei dati**, d'altra parte, consiste nella suddivisione dei dati in parti più piccole e la distribuzione di queste parti tra diverse macchine. Ciò può migliorare le prestazioni e la scalabilità del database, poiché ogni macchina può elaborare solo una parte dei dati. Tuttavia, può essere difficile garantire la coerenza dei dati tra le diverse parti.

Un altro aspetto importante dei database distribuiti riguarda la **gestione della concorrenza**. Poiché diversi utenti possono accedere al database contemporaneamente, è necessario implementare un meccanismo di controllo della concorrenza per garantire che i dati siano gestiti in modo coerente. Ciò può comportare l'uso di tecniche particolari per garantire l'integrità dei dati.

# Vantaggi dei database distribuiti




Possono riflettere la **struttura organizzativa** con porzioni di database memorizzate all'interno dei servizi cui si riferiscono

Maggiore protezione dei **dati sensibili** che non risultano accentrati in un unico luogo

Miglioramento delle **prestazioni** accentrando i dati nei pressi del sito di maggiore domanda e distribuendo il carico di lavoro su più server, migliorando le prestazioni globali

Capacità di fornire una maggiore **disponibilità e ridondanza** dei dati. Poiché i dati sono archiviati su più nodi di rete, il fallimento di uno o più nodi non compromette l'intero sistema.

Capacità di **scalare orizzontalmente**. Ciò significa che è possibile aggiungere nuovi nodi di rete per gestire la crescente quantità di dati e di traffico senza dover sostituire l'hardware esistente. Ciò può essere un grande vantaggio per le aziende che si aspettano una crescita costante dei dati o del traffico.

# Svantaggi dei database distribuiti

- Maggiore **complessità** di implementazione e gestione del sistema che si riflette sul lavoro degli Amministratori che debbono garantire la massima trasparenza della natura distribuita del sistema e mantenere più sistemi eterogenei.
- Serve un maggior lavoro di **progettazione** del database. I database distribuiti richiedono una pianificazione e una progettazione attente per garantire che i dati siano correttamente distribuiti tra i nodi di rete e che la coerenza dei dati sia mantenuta.
- Minore **economicità** se riferita ai costi del lavoro in quanto servono competenze specialistiche del personale e per questo anche maggiore difficoltà a reperire il personale necessario per progettare e amministrare il sistema
- Maggiore complessità delle **procedure di sicurezza**; può essere necessario, per esempio, crittografare i collegamenti di rete tra i diversi siti remoti
- È più difficile mantenere l'**integrità** dei dati, per esempio da modifiche accidentali o dolose e garantire l’accesso concorrente

# Prossimo Capitolo – Sicurezza informatica

Nel prossimo capitolo vedremo i concetti basilari della sicurezza informatica e come possiamo anche noi, proteggere il nostro computer di casa da virus, malware e attacchi informatici.