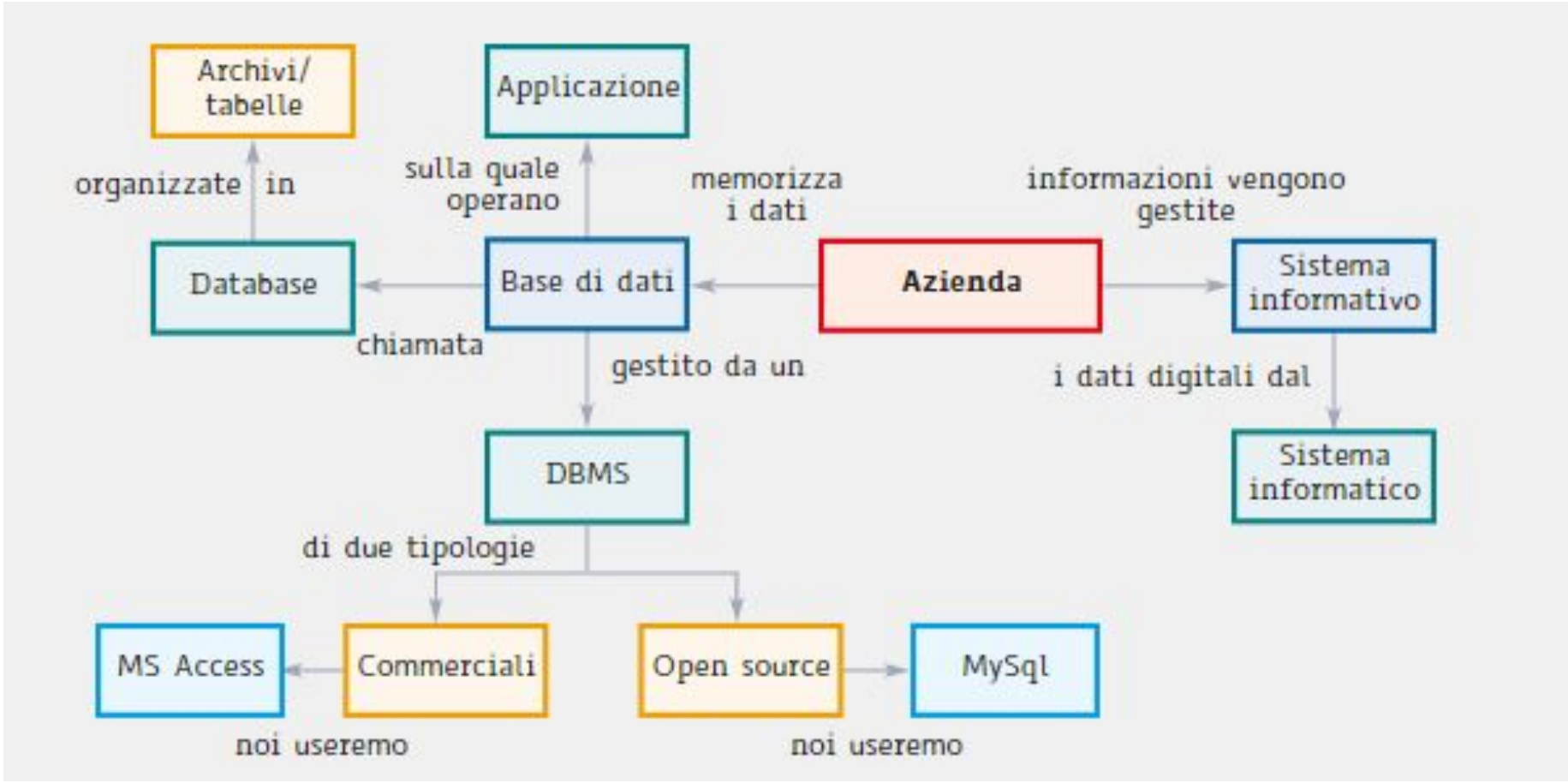
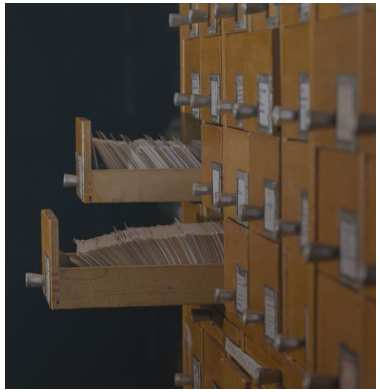
L’organizzazione degli archivi e le basi di dati

IIS Fermi Sacconi Ceci Ap
AS 2025/26 - Docente: Fedeli Massimo


Archivi/ tabelle
Applicazione
organizzate in
sulla quale operano
memorizza i dati
informazioni vengono gestite
Database
Base di dati
Azienda
Sistema informativo
chiamata
gestito da un
i dati digitali dal
DBMS
Sistema informatico
di due tipologie
MS Access
Commerciali
Open source
MySql
noi useremo
noi useremo

Gli archivi

- L'uso degli archivi deriva dalla necessità di **conservare** **dati** e **informazioni** in modo **permanente** perché potranno essere utili in momenti successivi.



Quali dati necessitano di essere memorizzati?

- Questo vale per dati di tipo personale, come quelli contenuti in una normale rubrica di contatti, ma anche per i documenti funzionali alla vita di uno **Stato**, di **un'azienda** o di un **ente**.

Chi ne ha bisogno?

- In un'**azienda** l'esecuzione delle normali attività amministrative e operative, così come la definizione e la scelta delle politiche commerciali, finanziarie e relative al personale, è strettamente legata all'**elaborazione di insiemi di dati** che sono raccolti e conservati in **archivi**.

Gli archivi

Archivi dei **movimenti contabili**, **archivi dei clienti** e dei **fornitori**, **archivi del personale**, archivi di **magazzino** e riguardanti la **produzione**: questi sono gli insiemi di dati tipici di un'azienda e il trattamento delle informazioni in essi contenute occupa una parte rilevante dell'attività delle aziende.

L'elaborazione di **grandi volumi di dati** che riguardano strumenti, attività e persone è di notevole importanza anche in tutti gli altri settori della società moderna.

Definizione di archivio

In generale, un **archivio** può essere definito come un **insieme organizzato di informazioni** caratterizzate da alcune **proprietà fondamentali**:

- **esiste un nesso logico tra di esse** (cioè sono in qualche modo inerenti a un medesimo argomento);
- sono rappresentate secondo un **formato** che ne rende possibile l'interpretazione;
- sono registrate con un **supporto** su cui è possibile scrivere e rileggere informazioni anche a distanza di tempo;
- sono organizzate in modo che siano **facilmente consultabili**.

Correlazione tra dati - esempio dell'archivio telefonico

L'**elenco telefonico** è un **archivio di dati** in cui le informazioni riguardano gli abbonati al telefono di una provincia.

Per ogni **abbonato** sono riportati nell'ordine: generalità, indirizzo, numero di telefono; tutte queste informazioni sono stampate su fogli di carta oppure sono **accessibili** tramite Internet.

Le **informazioni** vengono raccolte in questo archivio perché si riferiscono agli abbonati di una stessa provincia e, all'interno della provincia, di uno stesso **comune (nesso logico)**.

Gli archivi

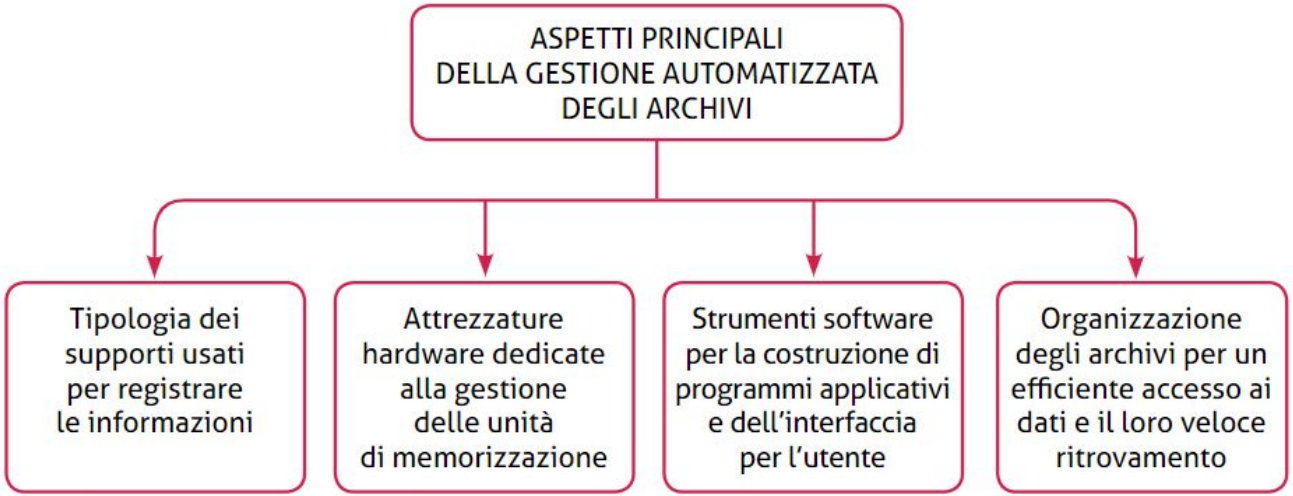
La **struttura** delle informazioni nelle **righe** (formato), ne rende facile la lettura e l'interpretazione da parte della persona che consulta l'elenco.

Il **supporto** è dal disco che contiene i dati.

Gli **abbonati** sono stampati seguendo l'ordine alfabetico dei cognomi, all'interno della suddivisione per comune, per permettere un veloce reperimento del numero di telefono che corrisponde alla persona cercata (**organizzazione dei dati**).

Gli archivi

Nella società moderna, con il crescere del volume delle informazioni e dell'importanza di **renderle disponibili in tempi rapidi**, la facile reperibilità dei dati è diventata una caratteristica fondamentale degli archivi e **l'uso di archivi gestiti da un computer è diventato la prassi** nella quasi totalità delle attività organizzate.

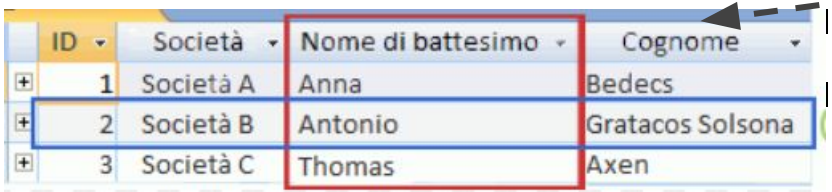


I record e i campi

In un **archivio** le informazioni, in genere, sono raggruppate secondo un'**unità logica**: nel caso dell'elenco telefonico i dati relativi a ogni **abbonato**, in un archivio notarile i dati contenuti nell' incartamento relativo a un cliente.

Questi insiemi di informazioni logicamente organizzate e riferite a un **unico soggetto** sono chiamati con il termine **record**;

- Le singole informazioni che compongono il **record** si chiamano **campi**;



ID	Società	Nome di battesimo	Cognome
1	Società A	Anna	Bedecs
2	Società B	Antonio	Gratacos Solsona
3	Società C	Thomas	Axen

L'elenco dei campi che lo compongono viene detto **tracciato del record**.

File

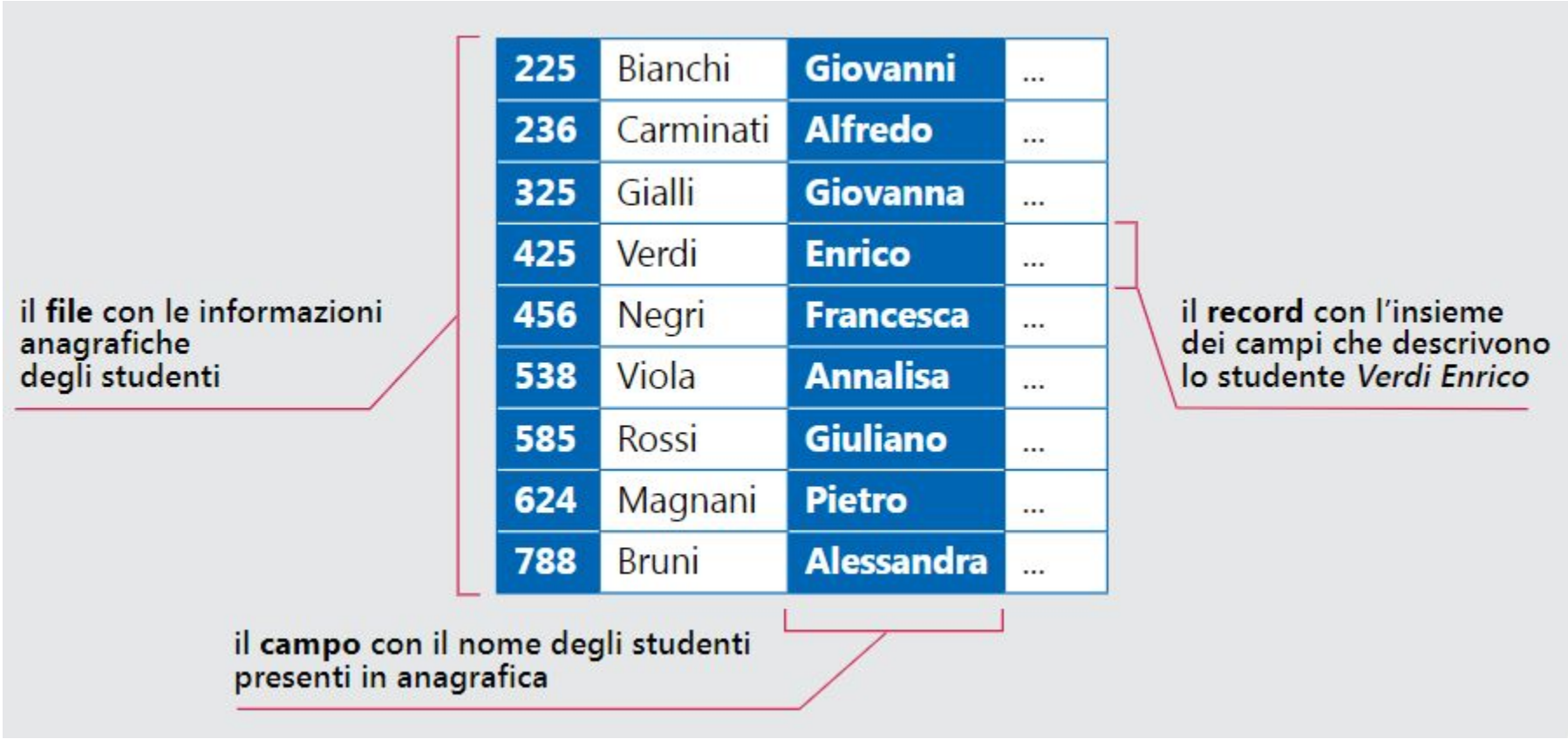
Si consideri un **archivio** con le informazioni **anagrafiche degli studenti**.

Una determinata **collezione di record** omogenei è detto **file**.
Un archivio si compone solitamente di più file.

Ogni **record** del file contiene le informazioni anagrafiche di un dato studente ed **è composto da un insieme di campi**.

Nel caso di un archivio anagrafico il **record** sarà composto, per esempio, da **campi** per rappresentare il numero di matricola, il cognome, il nome, la data e il luogo di nascita, l'indirizzo e il numero di telefono dello studente descritto nel record.

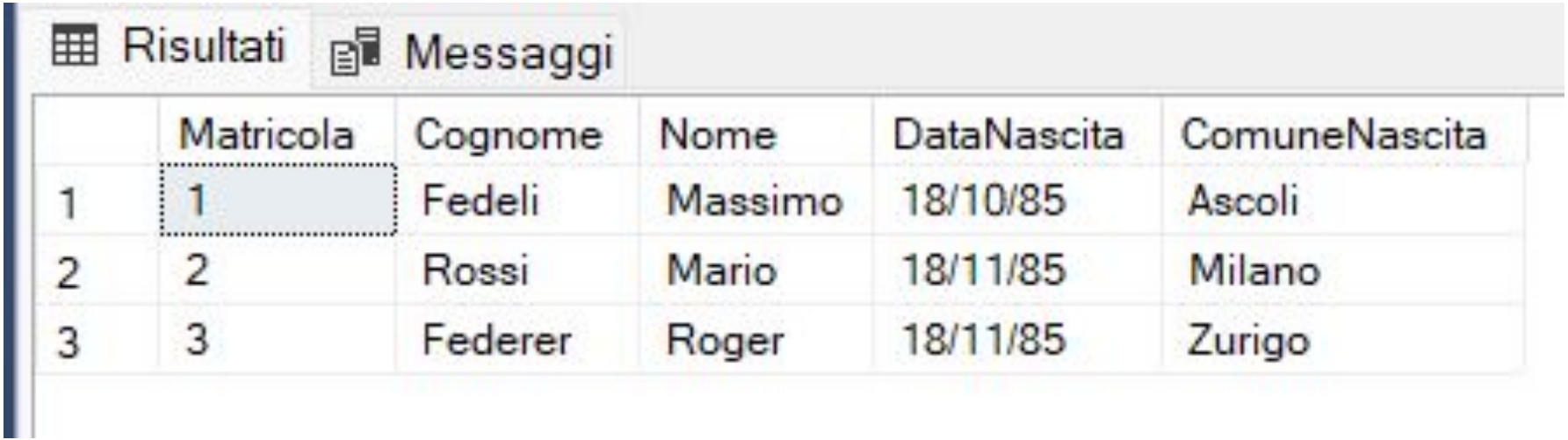
Esempio


il **file** con le informazioni anagrafiche degli studenti
225 Bianchi Giovanni ...
236 Carminati Alfredo ...
325 Gialli Giovanna ...
425 Verdi Enrico ...
456 Negri Francesca ...
538 Viola Annalisa ...
585 Rossi Giuliano ...
624 Magnani Pietro ...
788 Bruni Alessandra ...
il **record** con l'insieme dei campi che descrivono lo studente *Verdi Enrico*
il **campo** con il nome degli studenti presenti in anagrafica

Il file - definizione

Un **file** è una collezione di record, cioè di informazioni **logicamente omogenee** che descrivono i **singoli elementi** di una realtà considerata.

Ogni **record** è composto da un **insieme di campi** che contengono i valori assunti dalle caratteristiche scelte per descrivere la realtà.


Risultati Messaggi

	Matricola	Cognome	Nome	DataNascita	ComuneNascita
1	1	Fedeli	Massimo	18/10/85	Ascoli
2	2	Rossi	Mario	18/11/85	Milano
3	3	Federer	Roger	18/11/85	Zurigo

La creazione di un archivio

La creazione di un archivio

La creazione di un archivio richiede la **definizione preliminare** delle seguenti **specifiche**:

- il **nome dell'archivio**, che lo identifica e serve a ricordarne il contenuto; per esempio "archivio fornitori" oppure "archivio anagrafico";
- il **tracciato record**, in altre parole quali informazioni compongono il record;
- il **supporto da usare per archiviare i dati** (fogli di carta, dischi o nastri magnetici, dischi ottici);
- la **dimensione massima dell'archivio**: per esempio il numero massimo di scaffali occupati in un archivio cartaceo o di abbonati in un elenco telefonico;

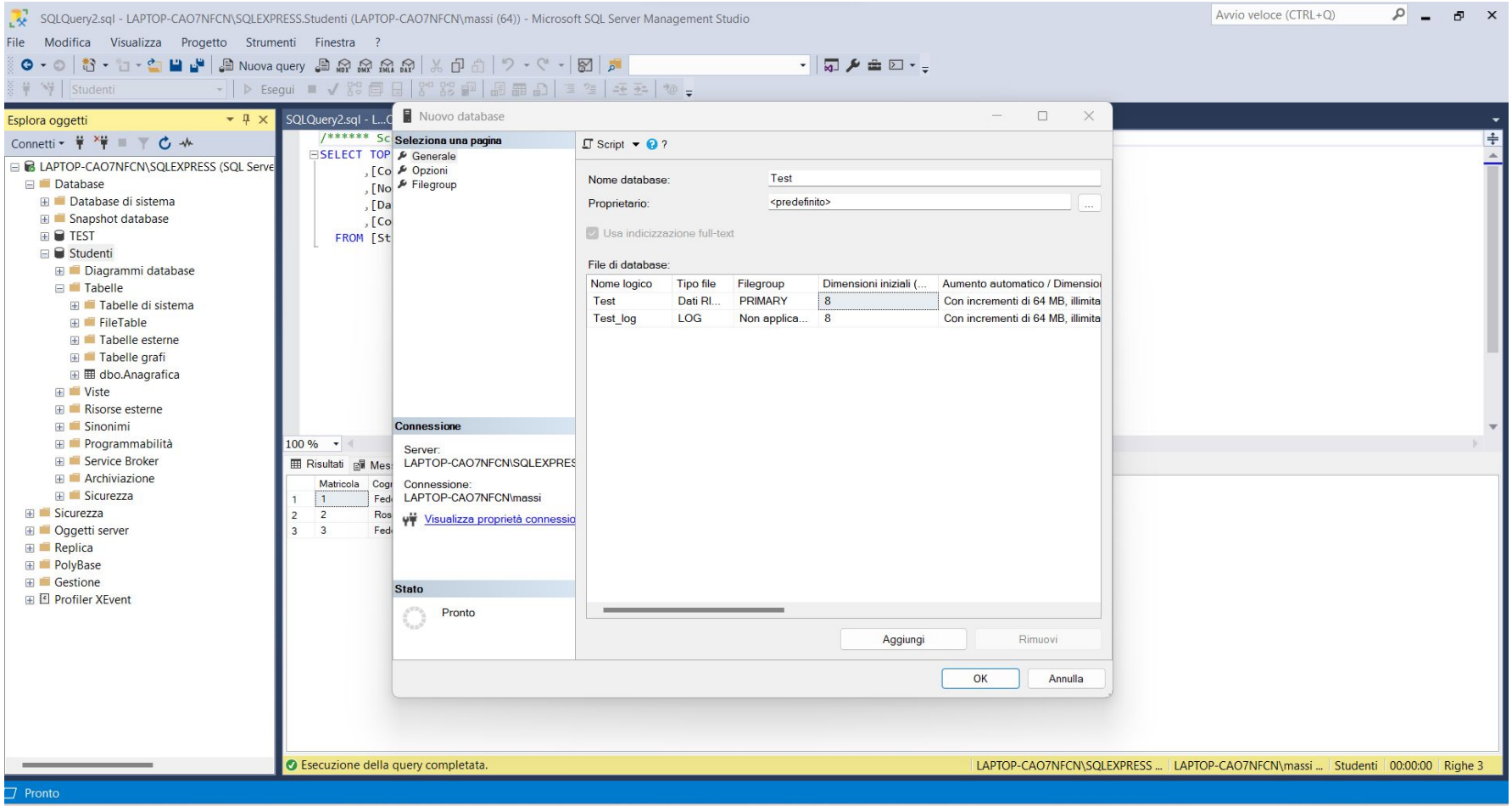
La creazione di un archivio

- il **modo con cui i dati sono strutturati** e collegati tra loro, cioè l'**organizzazione dell'archivio**: ci sono diverse possibilità di organizzazione e la scelta del **supporto di archiviazione** è spesso legata al tipo di organizzazione e alle modalità di consultazione previste.

(Legami tra le tabelle)

La decisione su queste specifiche fa parte dell'**analisi del problema** e deve precedere la **fase di realizzazione fisica dell'archivio**, che può consistere nella sistemazione delle informazioni già esistenti in un archivio ben organizzato, oppure nella predisposizione del supporto per la registrazione delle nuove informazioni che verranno successivamente inserite.

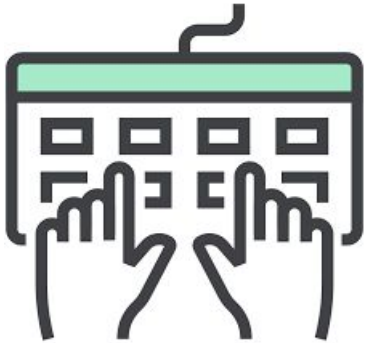
La creazione di un archivio



Interrogazione e modifica dei dati

Dopo aver creato l'archivio, su di esso si possono effettuare **operazioni di:**

- **manipolazione**, cioè inserimento di nuovi dati o variazione dei dati registrati;



- **interrogazione**, cioè reperimento all'interno dell'archivio delle informazioni necessarie

SQL

I file e le memorie di massa

La scelta del **tipo di supporto** in cui fisicamente sono contenute le informazioni è **strettamente legato alle diverse esigenze** di utilizzo delle informazioni stesse e quindi al modo in cui l'archivio viene **consultato**.

In **qualsiasi azienda moderna**, anche di piccole dimensioni, è necessario gestire una mole di dati tale da rendere **impensabile** o comunque molto onerosa sia la memorizzazione su **supporti tradizionali**.

I file e le memorie di massa

Per ragioni di **velocità nella ricerca** e nell'**elaborazione dei dati**, oltre che di spazio nella loro memorizzazione, si è passati da archivi registrati su supporti cartacei contenuti in armadi (schedari), adatti al reperimento manuale da parte dell'uomo, a supporti ideati per essere trattati in modo automatico dai computer.

I file e le memorie di massa

Gli archivi memorizzati su tali supporti vengono detti **file** perchè in inglese la parola archivio viene tradotta con **file**.

In informatica questo termine serve a indicare in modo generico qualsiasi informazione che può essere registrata sui supporti di memoria, come un **testo**, un **programma**, un **comando del sistema operativo** o un grafico.

I file e le memorie di massa

Nelle applicazioni con il computer, un **archivio** può contenere qualsiasi tipo di informazione, non solo di tipo testuale ma anche di tipo **multimediale**.

Studenti

Matricola	Cognome	Nome	DataNascita	Classe	Tel
0001	Rossi	Laura	15/04/2002	2A	320.5564332
0002	Verdi	Maria	12/08/2001	2A	333.9887001
0003	Bianco	Giorgio	06/01/2002	2A	349.5435672
0004	Neri	Luca	21/12/2001	2A	348.1267887

Prove

ID	Voto	Data	Materia
V001	10	24/03/2018	ITALIANO
V002	9	23/07/2017	MATEMATICA
V003	7	16/01/2018	FRANCESE
V004	9	20/11/2017	INGLESE

Nelle **procedure gestionali** comunque, nella maggior parte dei casi, gli archivi sono costituiti da insiemi di **record omogenei,** nel senso che ciascun archivio possiede un tracciato predefinito e uguale per tutti i record in esso contenuti, che si dicono **record logici** (file di record)

I file e le memorie di massa

I supporti per registrare i dati prendono il nome di **memorie di massa** perché possono **contenere notevoli quantità di dati**.

Si chiamano anche **memorie ausiliarie**, perché costituiscono un'estensione della memoria centrale di un **computer** e consentono, a differenza della memoria centrale che è una memoria volatile, la permanenza delle registrazioni nel tempo.

Tali apparecchiature sono esterne all'**unità centrale** e quindi sono unità **periferiche di memoria** (o semplicemente periferiche)

Le memorie di massa

Le **memorie di massa** sono caratterizzate da alcuni parametri usati abitualmente per illustrarne le prestazioni:

- il **tipo di accesso ai dati**, che può essere **diretto** (o random) come nei dischi oppure **sequenziale** come nei nastri;
- la **capacità**, in pratica la quantità di dati che il supporto é in grado di contenere; si misura in Megabyte, Gigabyte e Terabyte;
- il **tempo medio di accesso** misurato in millisecondi **(ms)**, costituito dal tempo medio necessario per ritrovare i dati e per trasferirli nell'unità centrale;
- la **velocità di trasferimento dei dati** dalla memoria di massa alla memoria centrale misurata in KB/s o MB/s, in altre parole il numero di byte trasferiti dal supporto alla memoria del computer in un secondo.

I file e le memorie di massa

Le **unità periferiche** che contengono i supporti di memorizzazione sono gestite da apparecchiature hardware, dette **drive**, che sono collegate al computer tramite opportune **schede di controllo** e che consentono la registrazione o l'accesso ai dati già registrati.

Il termine **driver** è invece usato in informatica per indicare il software di gestione della periferiche.

Trasferimento dati da e verso la memoria di massa

I bit e i codici binari

- I **dati** destinati a essere elaborati nei computer devono essere memorizzati in modo opportuno;
- I dispositivi di un calcolatore sono costituiti da **componenti**, in ognuno dei quali è possibile riconoscere con sicurezza **due stati distinti** (quali, per esempio, la conduzione oppure la non conduzione di corrente) ai quali sono associati per convenzione i valori 0 e 1, cioè quelli di una cifra binaria o **Binary digit**, da cui l'acronimo bit.

I blocchi

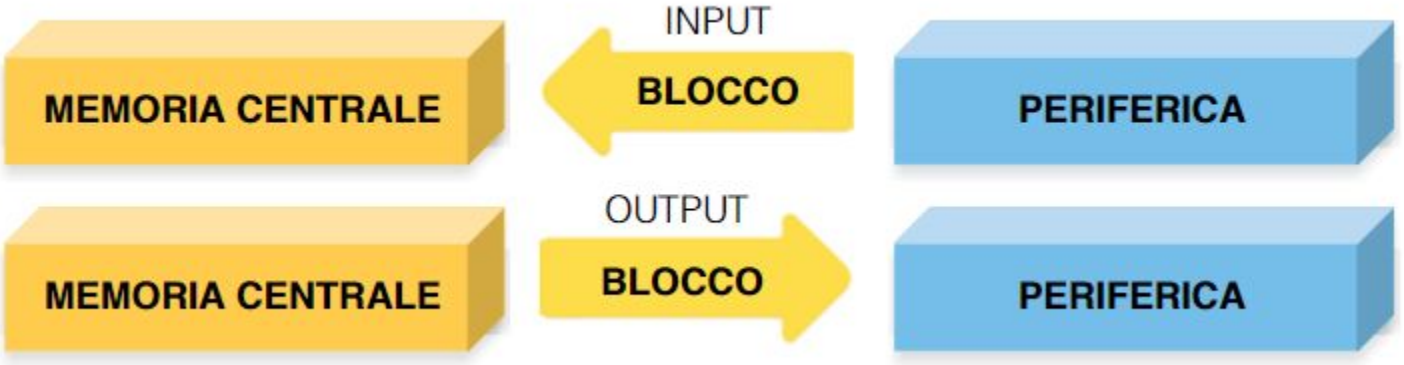
Ogni trasferimento di dati dalla **periferica** verso la **memoria centrale** e ogni trasferimento di dati dalla memoria centrale **verso la periferica** (operazione di output con l'uscita dei dati dalla memoria), riguarda non un singolo bit bensì un insieme di bit, detto **blocco**.

La **dimensione del blocco** varia da sistema a sistema e assume valori dell'ordine di alcuni kilobyte: **2KB**, **4KB**, **8KB** corrispondenti, rispettivamente, a 2048, 4096, 8192 byte, in quanto 1 KB corrisponde a 210 = 1024 byte.

I blocchi

- Nelle operazioni di **lettura** i dati sono copiati da disco in una zona di memoria centrale, detta **buffer di I/O**, per le successive elaborazioni.
- I dati inseriti o modificati nel buffer vengono ricopiati su disco con le operazioni di **scrittura**.
- Il **blocco** è l'**unità fisica** (o record fisico) di memorizzazione dei dati sulla memoria di massa e non deve essere confuso con il **record logico**: può accadere che un blocco contenga più record logici.

I blocchi



- In questo modo le operazioni di **lettura** e **scrittura** **su un file** non riguardano **singoli record logici**, ma gruppi di record: complessivamente diminuisce così il numero di accessi alla periferica, e quindi di operazioni lente rispetto agli accessi ai dati contenuti in memoria centrale.
- Si definisce **fattore di blocco** di un file il numero di record logici contenuti in un blocco.

I blocchi

- Se la **lunghezza del record logico** di un file è di 500 caratteri (e quindi 500 byte) e la dimensione del blocco sul disco è 2048 byte, il blocco può contenere 4 record, lasciando 48 byte inutilizzati.
- Il **fattore di blocco** risulta uguale a 4; un file con fattore di blocco uguale a 1 viene detto a **record sbloccati**.

I blocchi

- L'esempio precedente fa riferimento a un file i cui **record** hanno tutti la medesima lunghezza; questa è la scelta più frequente, poiché permette di semplificare le operazioni di lettura, di scrittura e di controllo.

- Oltre ai file con record a lunghezza fissa esistono file con **record a lunghezza variabile**: in essi è necessario specificare la posizione dove termina il singolo record (**end of record**).

- Ci sono fondamentalmente due modi per risolvere il problema: il numero di caratteri di ogni record è indicato in un campo di dimensione fissa all'interno del record stesso oppure, in alternativa, ogni record è seguito da una speciale sequenza per indicare la fine del medesimo.

I blocchi

Un esempio della seconda di queste tecniche consiste nell'impiego della **coppia di caratteri ASCII** di codice decimale 13 e 10 (**Carriage Return** e **Line Feed**), in sostanza una coppia di caratteri con il significato di "vai a capo" (usata originariamente dal sistema operativo DOS come marcatura di fine record)

Il filesystem

Il modulo del sistema operativo che svolge le funzioni di **gestore dei file** viene chiamato **file system**.

Esso è costituito dall'insieme delle **routine** (funzioni) che consentono all'**utente-programmatore** di usufruire degli archivi sulle memorie di massa, senza preoccuparsi dei dettagli delle operazioni di **input/output** (I/O) e facendo riferimento ai file solo con nomi simbolici.

Il filesystem

Il **file system** regola l'**organizzazione**, l'**assegnamento**, la **protezione** e il **ritrovamento** di insiemi di dati, cioè di **file**. In particolare esso svolge le seguenti funzioni:

- **tiene traccia dei file**, della loro posizione, del loro stato, usando particolari strutture dati dette file directory;
- decide secondo le richieste, le **protezioni** e i **diritti di accesso** (per esempio, lettura e scrittura oppure solo lettura) a quale programma assegnare un file o una parte di esso;
- assegna, cioè **rende disponibile, un file al programma** che lo ha richiesto;
- toglie l'uso di un file a un programma rendendolo disponibile ad altri programmi

Organizzazione degli archivi

Organizzazione sequenziale

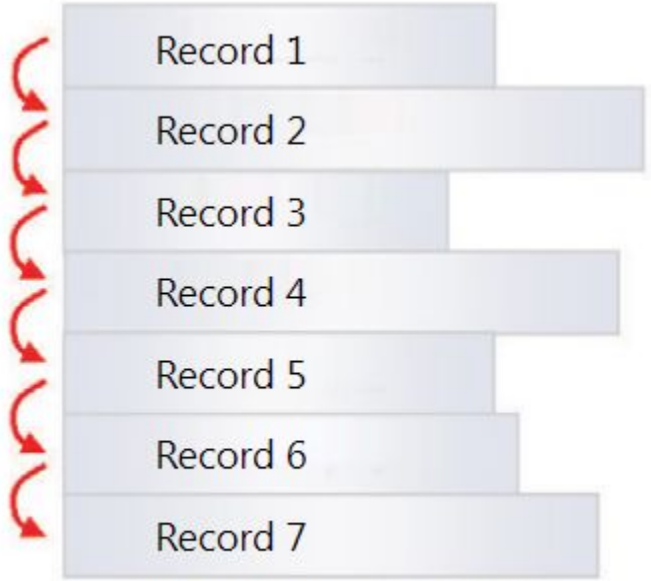
L’organizzazione sequenziale

L’**organizzazione sequenziale** consiste nel registrare i record **uno di seguito all’altro,** in modo sequenziale, intervallati da sequenze di caratteri che indicano la **fine del record**.

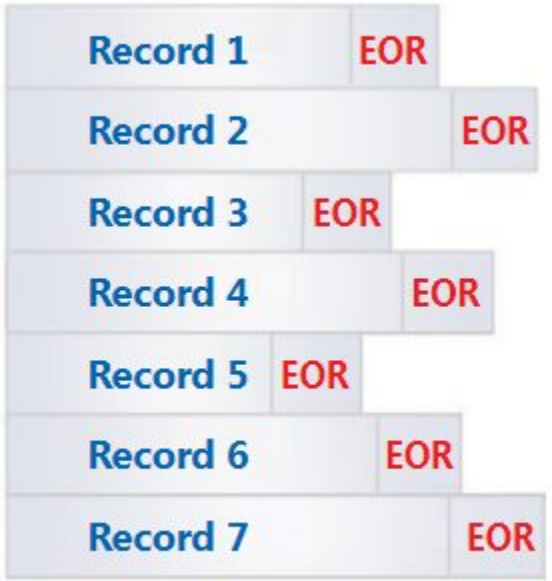
Essa consente l’uso di record a lunghezza variabile e ha come modello di file il pacco di schede o un file su nastro magnetico dove è possibile accedere a un record solo dopo aver visitato tutti i record che lo precedono.

Si parla in tal caso di **accesso sequenziale**

L’organizzazione sequenziale



Accesso sequenziale



Organizzazione sequenziale

L’organizzazione sequenziale

Questo tipo di organizzazione, **semplice da gestire**, consente l’uso di record aventi lunghezza diversa l’uno dall’altro.

Tuttavia è evidente che l’organizzazione sequenziale presenta dei **limiti in fase di ritrovamento dei dati** quando il numero dei record diventa elevato.

Risulta invece un’organizzazione efficace per file di piccole dimensioni, per esempio file di testo, e che comunque possono essere pensati come un **flusso di caratter**i intervallati da sequenze di caratteri che indicano la fine del record.

L’organizzazione sequenziale

Un file a **organizzazione sequenziale** può essere utilizzato solo per **scrivere** nuovi record, per **leggere** record o per **aggiungere** record **in coda a quelli già registrati**.

La scrittura di record a partire da una qualsiasi posizione, eseguita su un file già esistente, **provoca di norma la cancellazione** di tutti i record che lo seguono;

perciò **la riscrittura dei record non è permessa**.

Organizzazione ad accesso diretto

L’organizzazione ad accesso diretto

- I file con record di **lunghezza fissa**, cioè aventi tutti la stessa lunghezza, sono di uso comune nelle **applicazioni gestionali** (record del cliente, record dell’articolo di magazzino, record anagrafico dello studente): in questo caso si può calcolare la posizione del primo carattere di un qualsiasi record, nota la sua posizione nel file.

ESEMPIO

In un file con record di 240 caratteri, per leggere il 35esimo record ci si deve posizionare sul carattere di posto: (35 - 1) x 240 + 1 rispetto all’inizio del file, per poi trasferire da quella posizione 240 caratteri.

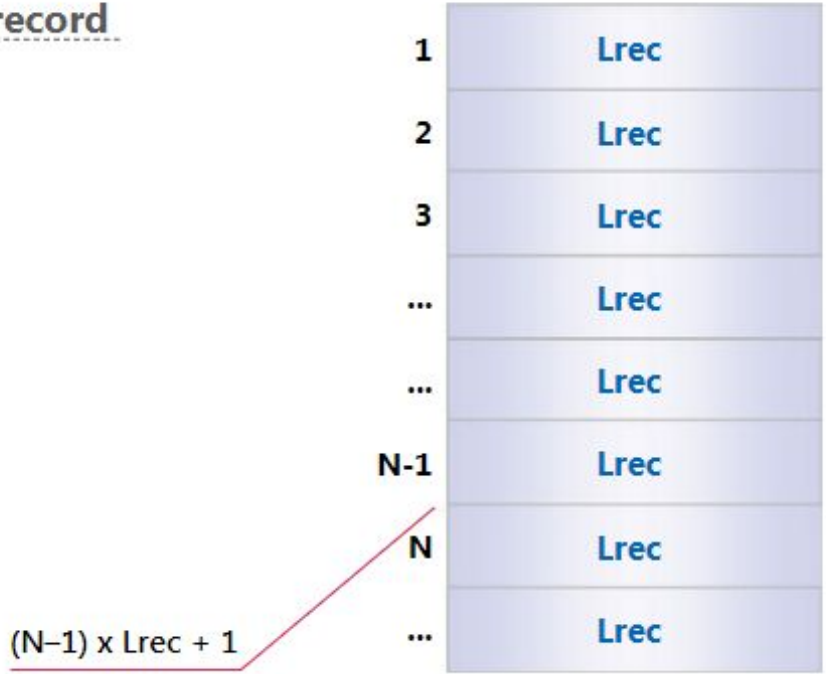
L’organizzazione ad accesso diretto

In generale, in un file con record di lunghezza *Lrec*, per leggere il record di posizione *N* ci si posiziona sul carattere di posto:

$$(N - 1) \times Lrec + 1$$

e da quella posizione si trasferiscono *Lrec* caratteri.

L’accesso diretto al record



L’organizzazione ad accesso diretto

Questo tipo di organizzazione si chiama **relativa** in quanto ci si riferisce ai record *relativamente* alla posizione che essi occupano nel file. Si parla anche di accesso **diretto** o **random** per evidenziare la possibilità di accedere al record che interessa senza leggere tutti quelli che lo precedono.

L’operazione di lettura con accesso diretto

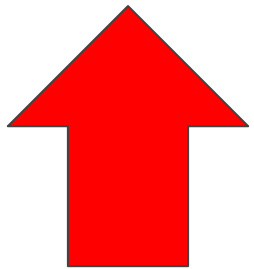
In un archivio a **organizzazione relativa** i record sono identificati dalla posizione che occupano nel file. Volendo accedere a un determinato record occorre specificare, con un valore numerico, la sua posizione in ogni operazione di lettura (**accesso diretto**).

Organizzazione ad indici

L'organizzazione ad indici (accesso associativo)

Nelle **applicazioni gestionali** non basta poter accedere a un record mediante la posizione che occupa in un file, ma bisogna riuscire a **identificare un record** in base a informazioni contenute nel record stesso (**accesso associativo**).

Si parla di accesso associativo quando in un archivio dei clienti si vogliono ritrovare le informazioni del cliente di nome *Giovanni Bianchi* senza conoscere la posizione occupata dal record nell'archivio.



Esempio Pratico: Archivio Clienti

Immagina di avere un database di 10.000 clienti.
Vuoi recuperare i dati di **Giovanni Bianchi**, ma non sai in quale posizione si trova il suo record nell'archivio.

Come Funziona l'Accesso Associativo?

Indice Creato: Un sistema crea un "indice" che associa ogni nome cliente alla posizione del suo record.
Esempio:

```
Indice:
- Mario Rossi → Posizione 145
- Giovanni Bianchi → Posizione 892
- Anna Verdi → Posizione 230
```

Ricerca Diretta: Inserisci "Giovanni Bianchi", l'indice ti restituisce istantaneamente la posizione **892**.
Il sistema accede direttamente al record senza scorrere tutto l'archivio.

Anche se devi "consultare" l'indice, lo fai su un dataset moltissimo più piccolo e con una struttura ottimizzata per la ricerca. Questo trasforma un'operazione lenta (scansione completa) in una quasi istantanea.

L’organizzazione ad indici

- L’**accesso associativo** si ottiene usando **file ad accesso diretto** e **metodi** per calcolare la posizione occupata dal record che contiene le informazioni cercate.
- Negli archivi con organizzazione ad indice i record sono identificati attraverso un elemento caratteristico (chiave) che di norma è una **variabile alfanumerica**, per esempio la **matricola del dipendente** di un’azienda, il **codice di un articolo**, il **codice del cliente**.

L’organizzazione ad indici

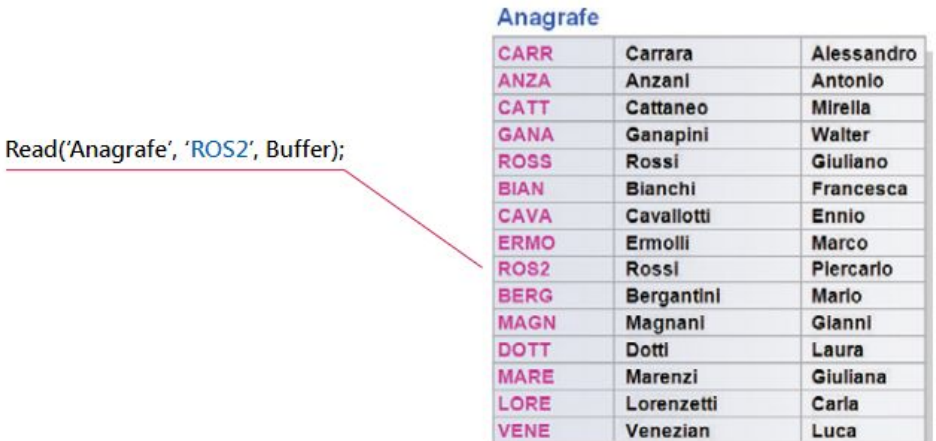
1. **La chiave (il "nome"):**
   Ogni record (foglio) ha un elemento unico:
   - Matricola del dipendente (`ID123`)
   - Codice articolo (`ART456`)
   - Codice cliente (`CLI789`)
2. **L'indice (l'"indirizzario"):**
   Il computer crea una tabella che associa ogni chiave alla posizione del record:

```
ID123 → Posizione 100 nel file
ART456 → Posizione 250 nel file
CLI789 → Posizione 300 nel file
```

3. **La ricerca:**
   Quando chiedi "trovami il cliente CLI789":
   - Il computer guarda l'indice → trova "CLI789 → Posizione 300".
   - Salta direttamente al record 300 → nessuna scansione totale!

L'organizzazione ad indici

- Organizzazione ad indici implica la possibilità di **leggere** e **scrivere** record in base al valore della chiave **identifica l'accesso a chiave**.
- I clienti sono identificati in modo univoco per mezzo di un **codice** e i record sono ordinati secondo l'ordine di arrivo in archivio.
- Si vogliono ritrovare i dati in base ai valori assunti dal codice evidenziato in colore.



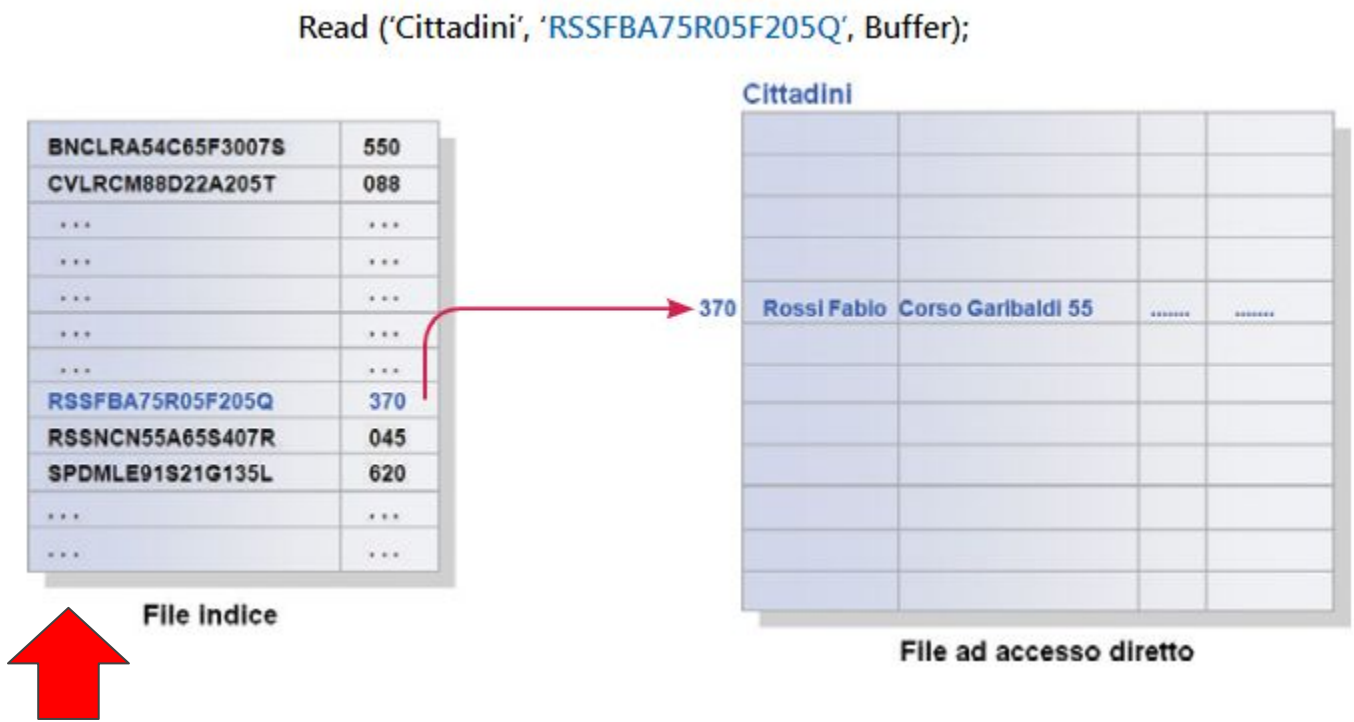
L'organizzazione ad indici

Per permettere l'**accesso a chiave** l'organizzazione dell'archivio deve essere simile a quella di un **libro dotato di indice analitico**: in tale indice sono elencate in ordine alfabetico le parole chiave, che richiamano gli argomenti e i concetti trattati nel testo, affiancate dalla pagina corrispondente del libro.

Il lettore cerca la parola nell'indice analitico secondo un **metodo di ricerca binaria (o ricerca dicotomica)**: essendo l'elenco ordinato, il numero di pagina funziona da puntatore della pagina dove viene trattato l'argomento.

Basta allora costruire un **indice ordinato** in base al valore della chiave e abbinare a ogni valore del codice un puntatore al record posto in un file ad accesso diretto.

L’organizzazione ad indici



Le chiavi sono ordinate in ordine alfabetico.

L'organizzazione ad indici

In un file con **organizzazione sequenziale a indici**, accanto alla zona con i record, memorizzati in un file ad accesso diretto, è gestita una **tabella delle chiavi o file indice** con l'**elenco ordinato delle chiavi.**

La **ricerca** di un record avviene con una **ricerca binaria** nella tabella delle chiavi per **recuperare la posizione del record** cercato nel file ad accesso diretto.

In questo modo si riesce ad accedere al record specificandone il valore della chiave (**accesso a chiave**).

L'organizzazione ad indici

È possibile costruire un **indice**

- sia sul campo che identifica univocamente un record, detto **chiave primaria**(per esempio, il numero di matricola di un dipendente)
- sia su **altri campi** anche non univoci (per esempio, il cognome del dipendente).

In questo caso, accanto all'indice della chiave primaria, devono essere costruiti gli indici per tutti gli altri campi chiave considerati,

in modo da consentire l'accesso a un record sia tramite la **chiave primaria**, sia tramite il **valore di altri campi**.

Le basi di dati

Le basi di dati

Con il termine **basi di dati** (o database) si indicano in informatica gli **archivi di dati**

- organizzati in **modo integrato**
- progettati con tecniche di **modellazione dei dati**
- **gestiti sulle memorie di massa** dei computer utilizzando appositi software

con l'obiettivo di **manipolare** e **trovare** in **modo efficiente i dati** memorizzati, superando i limiti presenti nelle organizzazioni tradizionali degli archivi.

Le basi di dati

Le **problematiche** relative alle basi di dati risalgono agli anni Sessanta e il passaggio dalla teoria all'applicazione sui computer con l'uso di software efficiente ha prodotto, nel corso degli anni successivi, strumenti software per **aumentare la produttività** nell'elaborazione di informazioni, soprattutto nei casi che riguardano la gestione di grandi quantità di dati.

La gestione delle basi di dati si è poi consolidata anche nelle applicazioni per i personal computer.

Le basi di dati

Quando si parla di **efficienza** e di **produttività** dell'organizzazione degli archivi si intende la possibilità di:

- **ritrovare facilmente** le informazioni desiderate, anche con diversi criteri di ricerca e per scopi diversi;
- **gestire i dati con una buona velocità** di elaborazione; garantire la sicurezza dei dati e l'integrità delle registrazioni.

Tutto questo diventa rilevante quando la gestione si riferisce a **una mole considerevole di dati**.

Devono anche essere offerte **misure di sicurezza** per impedire che il database venga danneggiato da interventi accidentali o non autorizzati, garantendo l'**integrità** dei dati e la **consistenza** del database.

Le basi di dati

- **Integrità** significa garantire che le operazioni effettuate sul database da utenti autorizzati non provochino una **perdita di consistenza ai dati.**

- **Consistenza** degli archivi significa assicurare che i dati in essi contenuti siano significativi e siano effettivamente utilizzabili nelle applicazioni.

Le basi di dati

Occorre anche sottolineare l'importanza che l'uso dei database assume nella gestione degli archivi di dati, favorendo l'utente del computer nel suo modo di vedere i dati e **liberandolo dagli aspetti riguardanti la collocazione fisica** delle registrazioni sui supporti degli archivi.

Viene così superata la visione tradizionale dell'**organizzazione degli archivi basata su file** che obbliga il programmatore ad accedere ai dati con comandi espliciti di apertura, chiusura, lettura, scrittura e posizionamento sui file contenenti i dati degli archivi da gestire.

I DBMS

Il **database**, è una **collezione di archivi (file) di dati ben organizzati** e ben **strutturati**, che sono gestiti in modo integrato e che costituiscono una base di lavoro per utenti diversi con programmi diversi.

I **prodotti software** per la gestione dei database sono indicati con il termine DBMS, acronimo di **DataBase Management System**.

I **DBMS** consentono all'utente di focalizzare l'attenzione solo sull'applicazione che utilizza i dati dell'archivio consentendone la gestione anche a utenti che nulla conoscono dell'hardware sottostante

I DBMS

In questo modo l'**utente del database** può concentrare la sua attenzione sul

- **progetto degli archivi**
- sulla **gestione e sul ritrovamento delle informazioni**

senza preoccuparsi del modo con il quale avviene l'**organizzazione fisica** dei dati sulle **memorie di massa**, compito che rimane a carico del software di gestione della base di dati.

I DBMS

Questo crea una distinzione tra:

- **utenti finali** che utilizzano le informazioni contenute nel database
- **amministratori del database** che si occupano della progettazione e della manutenzione degli archivi:

Questa distinzione corrisponde alla differenza tra

- **struttura concettuale** → modo attraverso il quale l'utente pensa all'organizzazione e al ritrovamento dei dati
- **struttura fisica dei dati** → tecniche utilizzate dal **sistema operativo** per registrare e leggere negli archivi.

Database vs DBMS

A questo proposito si tenga ben presente anche la differenza tra

- **Database** come insieme di dati
- **DBMS** come sistema per la gestione del database, così come nell'uso degli archivi tradizionali c'è una distinzione tra archivi di dati e file system.

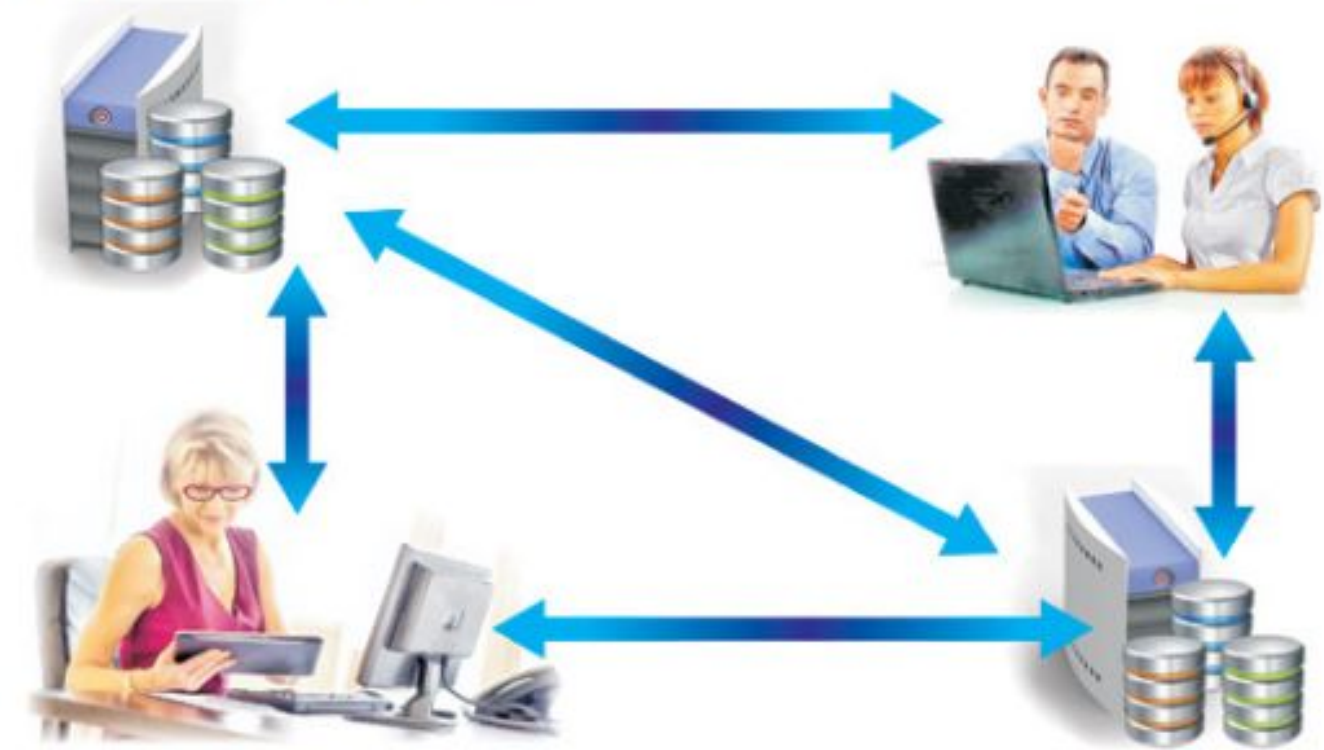
I DBMS - database distribuiti

Gli **utenti** della base di dati elaborano in modo locale gli archivi che hanno a disposizione nel proprio sistema e nello stesso tempo accedono in modo **remoto** a sistemi centrali attraverso le linee di comunicazione.

Gli archivi integrati che costituiscono la **base di dati aziendale** possono risiedere su un unico computer oppure possono essere distribuiti sulle memorie di massa di computer diversi, facenti parte di una rete aziendale, i cui nodi possono essere anche fisicamente lontani: in questo caso si parla di **database distribuiti**.

I DBMS

I database distribuiti



I DBMS

Con i **database distribuiti** è possibile gestire archivi di dimensioni limitate laddove vengono creati, facilitando il lavoro di **manutenzione** e di **controllo sulla sicurezza**, ma garantendo comunque la disponibilità dei dati aggiornati a tutti gli utenti del sistema informativo aziendale, qualunque sia la loro posizione geografica o il computer da essi usato per l'attività di elaborazione.

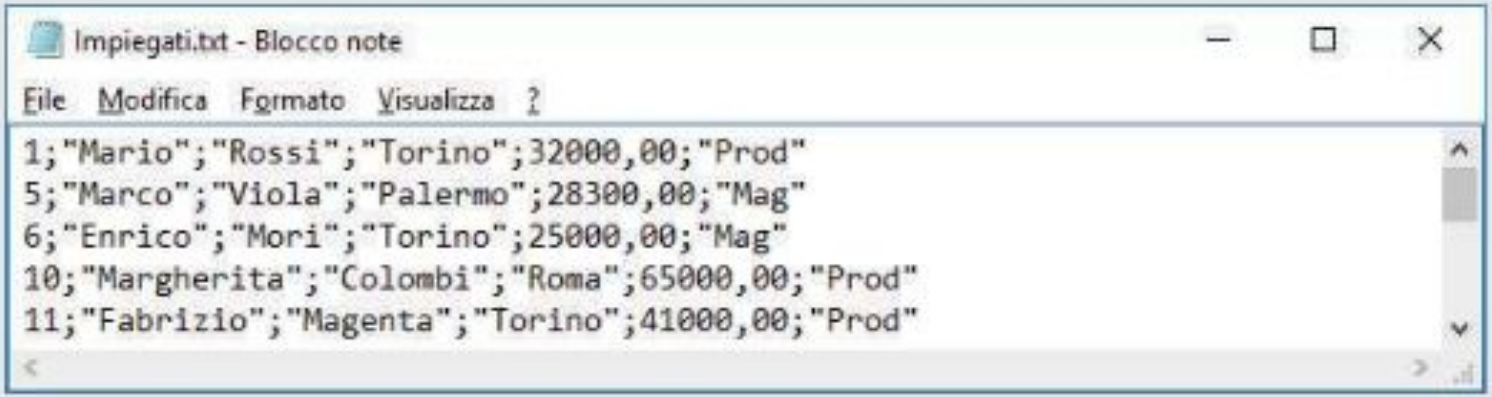
L'integrazione degli archivi è realizzata quindi a livello logico, nella visione che di essi hanno gli utenti finali, anche se i dati fisicamente possono risiedere su sistemi distanti.

Servizi offerti da un DBMS

Per esemplificare ed evidenziare i **servizi offerti** da un DBMS è utile confrontare le caratteristiche di un **file di dati** con quelle di un **archivio di database** con le stesse informazioni.

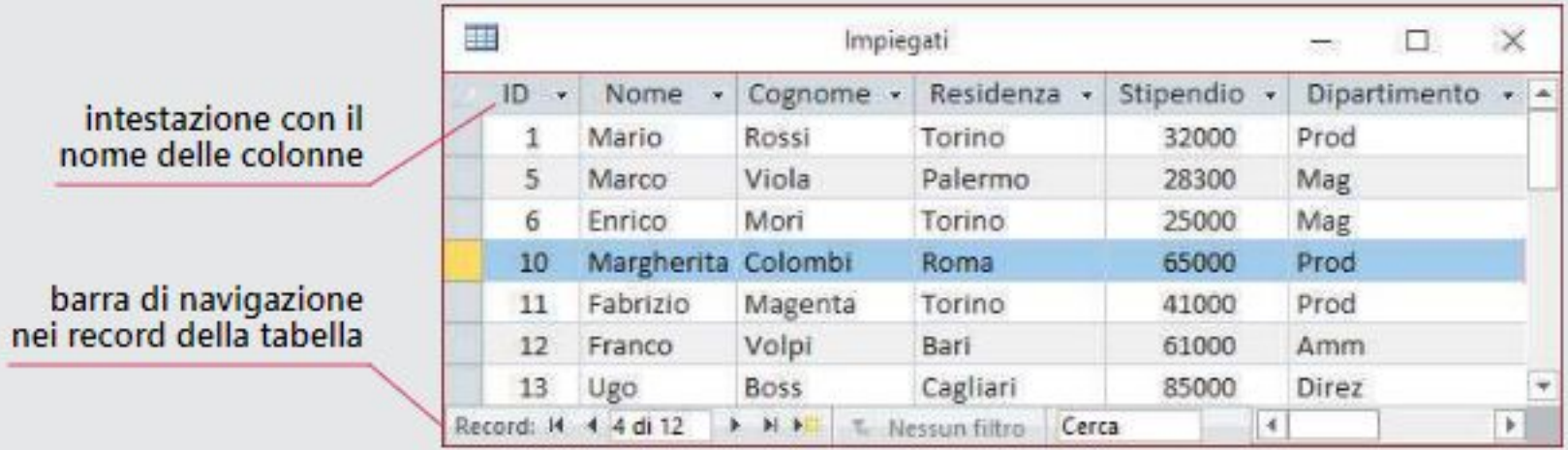
I DBMS

La figura seguente mostra i primi record del file *Impiegati.txt* che contiene alcune informazioni sui dipendenti di un'azienda. Il contenuto di *Impiegati.txt* è solo quello che si vede: un insieme di record in ognuno dei quali ci sono alcuni dati separati dal carattere punto e virgola (;). Il significato di tali dati può essere solo immaginato, ma non è esplicitamente dichiarato e memorizzato nel file.



I DBMS

La figura seguente mostra i primi record dei medesimi dati, memorizzati in un archivio di database che indicheremo come la tabella *Impiegati*. I dati sono mostrati in uno schema ben organizzato, composto da righe e colonne. Ogni colonna è etichettata con un nome, che permette di comprendere immediatamente quale sia il significato dei dati contenuti nell'archivio. Notiamo inoltre che la scritta esposta nella barra alla base della figura informa che la tabella contiene 12 record e che si sta operando sul quarto record di *Impiegati*: un operatore sta inserendo o modificando i campi dell'impiegato con *ID=10*.



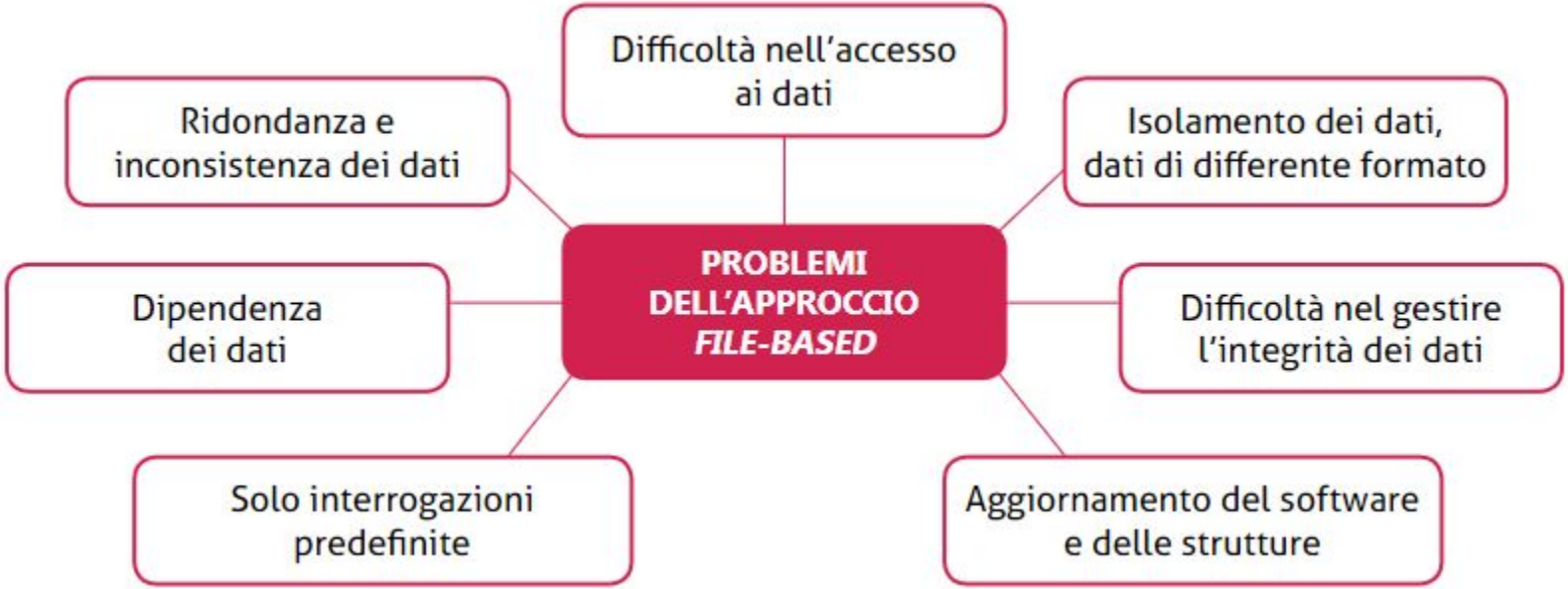
I DBMS

Questo avviene in forza dei **servizi offerti** dal **DBMS** che memorizza al proprio interno sia i dati che le loro caratteristiche.

Invece, nel caso del file **Impiegati.txt**, il nome dei campi e le loro caratteristiche sono presenti solo nella documentazione che accompagna il progetto e devono essere esplicitamente codificati dai programmatori in ogni modulo software che usa questi dati.

Le **tecniche di gestione delle basi di dati** nascono per risolvere i problemi e **superare i limiti dell'organizzazione tradizionale** degli archivi gestiti in modo non integrato.

Inconvenienti derivanti da archivi indipendenti su file



Il tradizionale approccio file-based, cioè basato su archivi indipendenti, favorisce alcuni inconvenienti.

Limiti dell'organizzazione tradizionale degli archivi

- **Ridondanza e inconsistenza dei dati**.

  La ridondanza nei dati è frequente nell'approccio basato su file indipendenti.

  La **duplicazione dei dati** richiede di inserirli più volte, con l'aumento dello spazio occupato dai dati, ma, soprattutto, la ridondanza può portare all' **incongruenza** nel caso in cui un dato sia aggiornato in un archivio e non in un altro, oppure siano presenti valori diversi per lo stesso dato.

Limiti dell'organizzazione tradizionale degli archivi

- L'**incongruenza** porta a sua volta all' **inconsistenza** dei dati, cioè i dati a disposizione non sono più affidabili, perché non si sa quale sia quello corretto.

- **Difficoltà nell'accesso ai dati**

L'accesso ai dati dipende dall'organizzazione scelta per gli archivi. Di conseguenza il programmatore deve accedere agli archivi con le modalità previste per l'organizzazione scelta e deve limitare le operazioni ammissibili sugli stessi.

Limiti dell'organizzazione tradizionale degli archivi

- **Isolamento dei dati, dati di differente formato**. I dati sono dispersi tra diversi file e sono rappresentati con differenti formati a causa, per esempio, dell'uso di differenti linguaggi nello sviluppo di diverse parti di un'applicazione: di conseguenza diventa difficile collegare i dati tra di loro e integrarli.

- **Dipendenza dai dati**. I programmi sono dipendenti dagli archivi che gestiscono, perché i linguaggi di programmazione richiedono di specificare, all'interno di ogni programma, quali sono gli archivi utilizzati e la struttura dei loro record

Limiti dell'organizzazione tradizionale degli archivi

- **Difficoltà nel gestire l'integrità dei dati**.

I vincoli di integrità si possono rappresentare solo scrivendo un apposito codice nei programmi che manipolano i dati.

Di conseguenza i vincoli di integrità non sono dichiarati esplicitamente ma sono impliciti, distribuiti e nascosti nel software e di difficile documentazione.

- **Interrogazioni predefinite**. È possibile accedere ai dati solo tramite un numero limitato di applicazioni sviluppate ad hoc. Per ogni altra esigenza informativa bisogna sviluppare nuove applicazioni

Limiti dell’organizzazione tradizionale degli archivi

- **Complessità nell’aggiornamento del software e delle strutture dati**

Ogni aggiornamento del software e ogni nuova applicazione deve tenere conto dei vincoli d’integrità inserendoli nei programmi mediante specifiche istruzioni.

Qualsiasi modifica alla struttura del record richiede la modifica di tutti i programmi che utilizzano quel record.

I modelli per il database

Il database è un **modello** della realtà considerata: i contenuti della base di dati rappresentano gli stati in cui si trova la realtà da modellare.

I cambiamenti che vengono apportati alla base di dati rappresentano gli eventi che avvengono nell'ambiente in cui opera l'azienda.

Per descrivere i dati, il loro significato, come sono correlati e i vincoli definiti su di essi si fa uso di **modelli dei dati**.

I modelli per il database

Essi sono classificabili secondo diversi livelli di generalità.

- **modello concettuale** (o modello a livello di oggetti)
- **modello logico** dei dati (o modello a livello di record)
- **modello fisico** dei dati, che si occupa delle modalità e delle tecniche usate nella registrazione sulle memorie di massa.

L'uso efficace dei dati organizzati in un database presuppone un attento lavoro di **progettazione iniziale**, che viene fatto con riferimento ai dati che si vogliono memorizzare e successivamente elaborare.

I modelli per il database

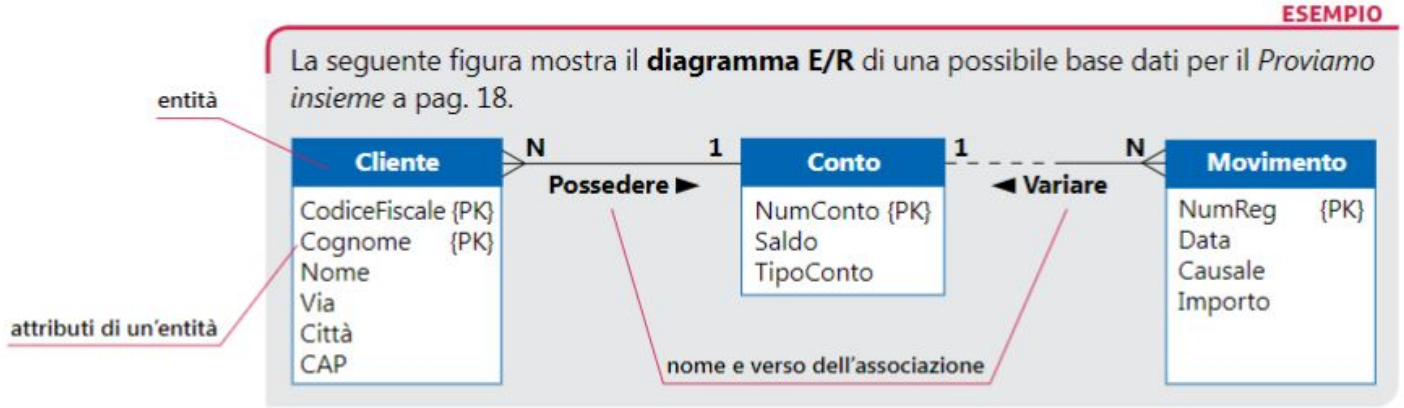
Il modello concettuale

Tra i numerosi modelli proposti per la progettazione concettuale, il più noto è il modello **Entità/Associazioni**, indicato come **modello E/R** dal termine inglese Entity/Relationship.

Nella costruzione del modello E/R di una realtà si **individuano gli oggetti che la compongono**, detti **entità**, poi gli **attributi**, che rappresentano le caratteristiche delle entità individuate, e infine le **associazioni**, che descrivono le correlazioni logiche tra entità.

Entità, attributi, associazioni sono rappresentati graficamente in un diagramma E/R.

Il modello concettuale



Nel modello E/R in figura si evidenziano tre **entità**:

- Cliente
- Conto
- Movimento

e due **associazioni**

Il modello concettuale

Associazioni:

- Cliente e Conto (indicata con Possedere)
- Movimento e Conto (di nome Variare).

L'entità **Cliente** è caratterizzata dagli **attributi**: CodiceFiscale, Cognome, Nome, Via, Città, CAP.

Gli **attributi** di Conto sono NumConto, Saldo e TipoConto; mentre gli attributi che caratterizzano Movimento sono NumReg (numero progressivo di registrazione), Data, Causale e Importo.

Gli attributi **CodiceFiscale** e **NumConto** sono le chiavi primarie, PK (primary key), delle entità Cliente e Conto.

La chiave di Movimento è NumReg

Il modello concettuale

Il modello **Entità/Associazioni** è anche un importante strumento di comunicazione.
Dalla lettura del modello Entità/Associazioni è possibile avere una precisa idea del problema analizzato ed esprimere la visione dell'analista con frasi espresse in linguaggio naturale.

Per esempio, le associazioni Possedere e Variare possono essere descritte con le seguenti frasi:

- ogni cliente deve possedere uno e un solo conto;
- ogni conto deve essere posseduto da uno o più clienti;
- ogni conto può essere variato da uno o più movimenti e ci possono essere conti non movimentati;
- ogni movimento deve variare uno e un solo conto

Il modello logico

A partire dallo schema concettuale Entità/Associazioni, un database può essere progettato e realizzato passando al modello logico, cioè alle strutture che organizzano i dati, in modo da consentire le operazioni di manipolazione e di interrogazione.

Nello sviluppo della teoria dei database, a partire dagli anni Sessanta, sono emersi tre diversi tipi di modelli a livello di record per le basi di dati: il **modello gerarchico**, il **modello reticolare** e il **modello relazionale**

Il modello logico

ESEMPIO

Per rappresentare il legame che intercorre tra *Cliente* e *Conto*, nel record che descrive l'anagrafica di ogni cliente è inserito, in coda ai dati, un puntatore che precisa la posizione fisica su disco del record correlato (quello relativo al conto di quel cliente).

È evidente che, dopo aver effettuato l'accesso all'anagrafica di un cliente, per trovare il record correlato è richiesta una limitata capacità elaborativa da parte del sistema ma, di contro, sono complesse le operazione di manutenzione, ristrutturazione e di esportazione dei dati.

Questo spiega anche il successo avuto da questi modelli di database nel corso degli anni Settanta e nella prima metà degli anni Ottanta, vista la limitata potenza dei sistemi allora disponibili.

Il modello logico

I modelli **gerarchico** e **reticolare** sono stati definiti attraverso un processo di astrazione da sistemi già implementati, mentre il modello relazionale è stato definito a livello teorico prima di qualsiasi implementazione sul computer.

Il modello relazionale nasce nel 1970, proposto da Edgar F. Codd, ricercatore IBM, come idea di un modello logico molto semplice e nello stesso tempo in grado di superare i limiti degli altri modelli utilizzati.

Il modello relazionale

Il **modello relazionale** si basa su alcuni concetti fondamentali tipicamente matematici e assegna grande importanza all'uso rigoroso del linguaggio matematico, con due obiettivi importanti:

- utilizzare un linguaggio conosciuto a livello universale, quale è il linguaggio matematico;
- eliminare i problemi di ambiguità nella terminologia e nella simbologia

Il modello relazionale è un modello basato sui valori. Le associazioni tra entità sono descritte solamente tramite i valori assunti da campi, nelle righe delle tabelle che modellano le entità stesse, senza fare uso di puntatori.

Il modello relazionale

Il database per i conti bancari del *Proviamo insieme* a pag. 18 si concretizza nelle tre tabelle in figura. Nella figura non sono tracciati archi che connettono le righe delle tabelle in quanto il modello relazionale è, come detto, un modello basato sui valori.

CLIENTI

Cognome	Via	Città	CAP	NumConto
Bianchi	Mazzini	Bergamo	24127	**9000**
Neri	Garibaldi	Napoli	80120	7000
Rossi	Rosmini	Milano	20125	**4500**
Galli	Cavour	Bari	70122	5100
Rosa	Crispi	Torino	10137	8000
Verdi	Dante	Roma	00128	5100
Moro	Colombo	Milano	20143	4310

i valori del campo *NumConto* permettono di affermare, per esempio, che il signor *Bianchi* ha un conto con saldo di *120.345* euro e che il *23 dicembre* ha effettuato un versamento di *2.500* euro

Il modello relazionale

CONTI

NumConto	Saldo
9000	120.345,00
7000	500,00
4500	3.500,00
5100	58.500,00
8000	6.320,00
4310	37,00

MOVIMENTI

NumConto	NumReg	Data	Causale	Importo
9000	1	23/12	Vers	2.500,00
7000	2	20/04	Prel	1.250,00
7000	3	27/05	Vers	550,00
–	–	–	–	–
4310	150	21/08	Prel	350,00
4310	151	21/08	Prel	536,00

I valori del campo *NumConto* di *Clienti* e di *Conti* permettono di stabilire l'associazione tra un certo cliente e il saldo del proprio conto. Analogamente i valori del campo *NumConto* di *Clienti* e di *Movimenti* permettono di stabilire l'associazione tra un cliente e i rispettivi movimenti.

L'assenza di puntatori semplifica le operazioni di manutenzione del database e di esportazione degli archivi.

Il modello relazionale

Si supponga che alla tabella Clienti si vogliano aggiungere due colonne, la prima con il codice fiscale del cliente e la seconda con un suo recapito telefonico.

La nuova tabella occuperà più spazio della precedente e andrà collocata in una diversa posizione su disco, ma non sarà necessario modificare i puntatori che descrivono l'associazione tra Clienti e Conti.

Il modello relazionale

Inoltre, volendo aggiungere al database la tabella Causali con i campi Causale e DescrizioneCausale, tutto quello che il DBMS deve fare per descrivere l'associazione tra Movimenti e Causali è inserire tra i metadati il fatto che il campo Causale di Movimenti è in corrispondenza con lo stesso campo di Causali, senza alcun riferimento alla collocazione della nuova tabella su disco e senza inserire puntatori per collegare una causale con la relativa descrizione.

Il costo di questa scelta risiede nella necessità di dover ricercare i record da abbinare ogni volta che ciò è necessario.

Il modello relazionale

Esempio.
Si supponga di formulare una richiesta per conoscere il saldo del conto di Rossi. Nel caso del modello relazionale il sistema deve cercare nella tabella Conti il record da abbinare a quello della tabella Clienti con le informazioni di Rossi.

L'abbinamento avviene esaminando la tabella Conti alla ricerca del record con il valore di 4500 nel campo NumConto e utilizzando un indice sul campo Cognome di Clienti. Nel modello gerarchico o reticolare invece l'accesso al record anagrafico di Rossi permette (al DBMS) di conoscere immediatamente la posizione sul disco del record dei conti collegato al record anagrafico di Rossi e di prelevare l'informazione richiesta

Il modello relazionale

La diffusione del modello relazionale è cresciuta nel tempo in coincidenza con la crescita della capacità delle memorie centrali e della potenza di calcolo dei processori, che hanno permesso di manipolare le tabelle con sempre maggior rapidità.

Invece i database costruiti sul modello gerarchico e reticolare, meno esigenti in termini di prestazioni dell'hardware, erano più adeguati alle limitate prestazioni delle macchine degli anni Settanta.

Il modello relazionale

Le operazioni sui database gerarchici e reticolari sono complesse, poiché agiscono su singoli record che vengono identificati specificando i percorsi per ritrovarli.

Si dice in tale caso che l'approccio adottato nell'elaborazione dei dati è di tipo procedurale, ovvero bisogna indicare al computer come deve fare per trovare i dati.

I sistemi relazionali adottano invece un approccio di tipo dichiarativo nell'elaborazione delle informazioni, in quanto viene specificato cosa si vuole trovare, mentre i percorsi per trovare i dati sono a carico del sistema

Il modello relazionale

Tra i modelli di database illustrati, il modello relazionale è il più diffuso nella quasi totalità dei prodotti commerciali DBMS, nei sistemi di elaborazione grandi, medi e piccoli.

Proprio questo modello ha determinato un’ampia diffusione dei programmi per la gestione di database anche per i personal computer,

Il modello fisico

Il **modello fisico** dei dati ha la funzione di descrivere il modo con il quale un dato modello logico è realizzato concretamente sulle memorie di massa del computer.

Nel caso del modello relazionale, il modello fisico precisa come sono realizzate le tabelle, il modo per implementare i vincoli sui dati che le compongono, come rappresentare le associazioni tra tabelle, come costruire gli indici sui campi di una tabella e così via.

Il modello fisico si occupa di problemi trattati dai progettisti di uno specifico DBMS. Esempi di prodotti DBMS che gestiscono i modelli fisici dei dati sono Oracle, DB2, MySQL, SQL Server, Access.

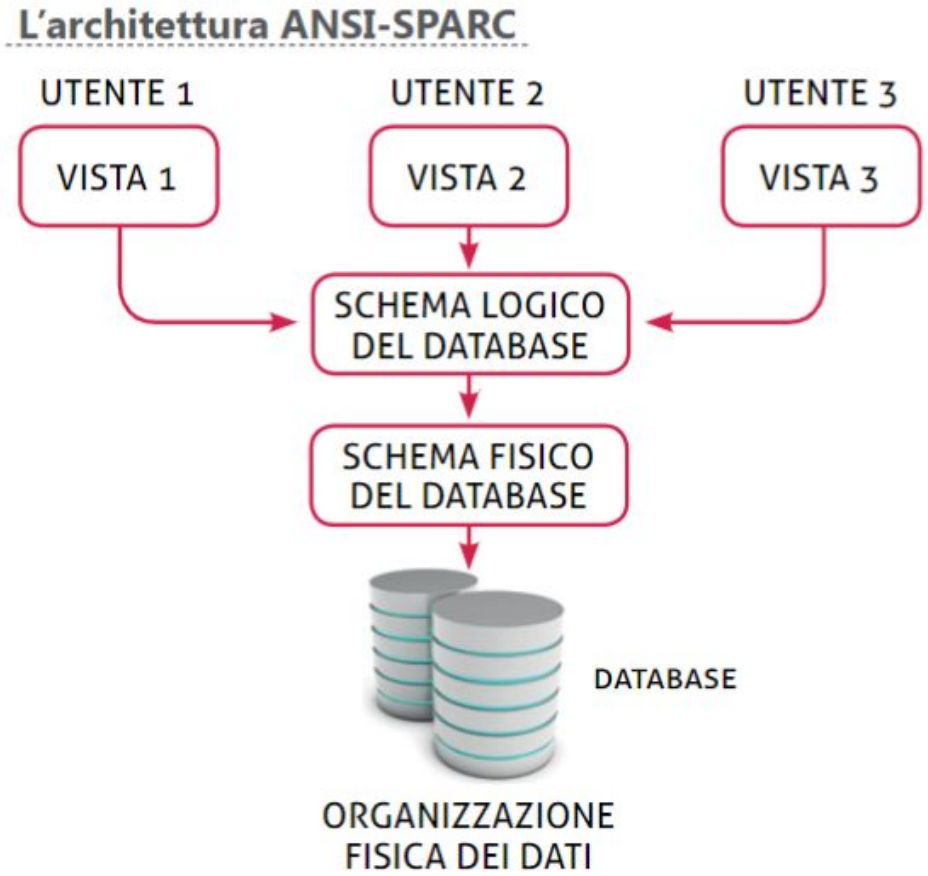
Architettura a tre livelli e indipendenza dei dati

I software DBMS seguono, di fatto, l'impostazione concettuale dell'architettura a tre livelli ANSI-SPARC, proposta nel 1975 da un comitato (Standard Planning And Requirements Committee) dell'ANSI (American National Standards Institute).

Secondo questa impostazione i dati sono descritti secondo tre differenti livelli di astrazione, mediante opportuni schemi. I tre livelli sono indicati come il livello esterno, il livello logico e il livello interno.Il livello esterno(viste)rappresenta la visione del database da parte dell'utente.

Descrive la parte del database che è importante per il singolo utente o per un gruppo di utenti.

Architettura a tre livelli e indipendenza dei dati



Architettura a tre livelli e indipendenza dei dati

In generale in un database ci sono tanti schemi esterni quante sono le classi di utente con differenti esigenze e visione dei dati.

Il livello logico rappresenta la visione complessiva del database dal punto di vista logico, indipendentemente dalle modalità e dalle tecniche di memorizzazione dei dati. Il livello logico descrive, con le convenzioni del modello di database scelto, entità, attributi, associazioni e vincoli. Il livello logico è la cerniera tra ciò che vedono gli utenti e quello che viene effettivamente memorizzato nel computer.

Architettura a tre livelli e indipendenza dei dati

Naturalmente c'è un solo schema logico di database.Il livello interno(fisico)riguarda la rappresentazione fisica del database nel computer e fornisce informazioni in merito alla realizzazione concreta del livello logico.

Il livello interno entra nel merito dei dettagli implementativi delle forme di memorizzazione.

Esempio.
Una tabella di un database relazionale può essere rappresentata con un file ad accesso diretto e con una struttura a indice sequenziale per rappresentare la sua chiave primaria; nel caso di un database gerarchico o reticolare, è di competenza del livello interno la realizzazione, mediante puntatori, degli archi che collegano i nodi.

Architettura a tre livelli e indipendenza dei dati

I LIVELLI DEL DATABASE

Esaminare il database con gli studenti e i dipartimenti di un'università

Per comprendere meglio il significato di livello esterno e come esso si relaziona con gli altri livelli, si consideri un semplice database relazionale composto dalle tabelle *Studenti* e *Dipartimenti*.

Le due tabelle contengono informazioni di tipo anagrafico su alcuni studenti universitari e informazioni sui dipartimenti che compongono l'università alla quale sono iscritti.

Studenti

Matricola	Nome	Cognome	Indirizzo	CodDip
2340	Nino	Verdi	Milano	Ing
2360	Enrico	Bianchi	Venezia	Eco
2361	Francesco	Spreafico	Bologna	Eco
2362	Anita	Rossigni	Ferrara	Let
2363	Clara	Riseri	Palermo	Let

Architettura a tre livelli e indipendenza dei dati

Dipartimenti

CodDip	NomeDipartimento	Indirizzo
Eco	Economia	Via DeGasperi
Giu	Giurisprudenza	Piazza Martini
Ing	Ingegneria	Via Carducci
Let	Lettere Moderne	Via Monte Rosa
Lin	Lingue Straniere	Via Garibaldi

Gli uffici dei diversi dipartimenti sono interessati alle informazioni anagrafiche dei soli studenti del dipartimento stesso. Di conseguenza la visione dei dati che hanno gli impiegati dei dipartimenti di Ingegneria ed Economia è differente. I primi desiderano conoscere i soli valori di *Matricola*, *Nome*, *Cognome* e *Indirizzo* dei record di *Studenti* per i quali il campo *CodDip* vale *Ing*, mentre i secondi desiderano le medesime informazioni, ma per record di *Studenti* che hanno il valore *Eco* nel campo *CodDip*.
Agli impiegati dei dipartimenti di Economia e Ingegneria vengono di conseguenza fornite differenti visioni esterne della tabella *Studenti*, indicate in figura con i nomi di *StudentiEconomia* e *StudentiIngegneria*.

Architettura a tre livelli e indipendenza dei dati

StudentiIngegneria

Matricola	Nome	Cognome	Indirizzo
2340	Nino	Verdi	Milano
2373	Giuseppe	Galeotti	Milano
2374	Fabio	Lorenzoni	Cagliari

StudentiEconomia

Matricola	Nome	Cognome	Indirizzo
2360	Enrico	Bianchi	Venezia
2361	Francesco	Spreafico	Bologna
2367	Gualtiero	Ceronte	Modena

Queste tabelle sono virtuali e prendono il nome di **viste logiche**; esse rappresentano un esempio di quello che si intende per **livello esterno** dei database. Naturalmente esisteranno anche viste logiche di nome *StudentiGiurisprudenza*, *StudentiLettere* e *StudentiLingue*.

Architettura a tre livelli e indipendenza dei dati

In questo semplice database:

- il livello esterno è composto dalle cinque differenti visioni dei dati che hanno gli impiegati dei cinque dipartimenti;
- il livello logico è costituito dalla coppia di tabelle Dipartimenti e Studenti;
- il livello interno descrive come sono memorizzate le tabelle Studenti e Dipartimenti e dipende dal DBMS scelto;
- al di sotto del livello interno ci sono i bit, o meglio i blocchi di byte, su disco che sono di competenza del sistema operativo.

Architettura a tre livelli e indipendenza dei dati

Il DBMS mantiene al proprio interno la descrizione delle viste logiche, ovvero degli schemi esterni, dello schema logico e dello schema interno del database. In base a queste informazioni, esso è in grado di attuare la corrispondenza (mapping) tra le viste logiche e lo schema logico del database e tra lo schema logico e il livello interno.

Si supponga per esempio che un impiegato di Ingegneria desideri accedere tramite un'apposita interrogazione alle informazioni anagrafi che di uno studente di nome Nino Verdi.

- Il DBMS traduce la richiesta nella ricerca dei record di Studenti con quel nome e cognome, ma con il campo CodDip di valore Ing, identifi cando il record di Studenti con Matricola = 2340.

Architettura a tre livelli e indipendenza dei dati

- In base alle informazioni sull'organizzazione degli archivi contenute nello schema interno del database, il DBMS riesce a localizzare il record corrispondente allo studente con quel valore di matricola. Il risultato di questa operazione è del tipo: la riga cercata si trova nel record di posto 25 del file ad accesso diretto.
- L'abbinamento tra numero di record e numero di blocco del file e la successiva ricerca su disco vengono attuati dal DBMS e dal sistema operativo con modalità che dipendono dal DBMS usato e dallo specifico sistema operativo.

Architettura a tre livelli e indipendenza dei dati

L’architettura a tre livelli ha il vantaggio di semplificare la visione del database da parte degli utenti, in quanto essi vedono solo le informazioni alle quali sono interessati, secondo la propria visione del mondo reale.

Inoltre l’architettura a tre livelli limita le possibilità di errore e permette di affrontare i problemi di riservatezza in modo puntuale.

Architettura a tre livelli e indipendenza dei dati

L’architettura a tre livelli dei database realizza meccanismi di astrazione dei dati e assicura la cosiddetta indipendenza dei dati.

Con questo termine si vuole indicare il fatto che i livelli superiori non sono influenzati, entro certi limiti, dai cambiamenti che avvengono nei livelli inferiori dell’architettura. Si identificano due livelli di indipendenza dei dati: l’indipendenza logica e l’indipendenza fisica.

Architettura a tre livelli e indipendenza dei dati

- Con **indipendenza logica dei dati** si fa riferimento alla capacità dello schema esterno di non essere influenzato dai cambiamenti apportati allo schema logico.

ESEMPIO

L'aggiunta di campi alla tabella *Studenti* non ha alcun effetto sulla vista logica *StudentiEconomia* (e sulle altre quattro); le applicazioni basate su queste tabelle virtuali, e non interessate a fare uso dei dati inseriti nei nuovi campi, non ne risentiranno e non sarà necessario modificarle.

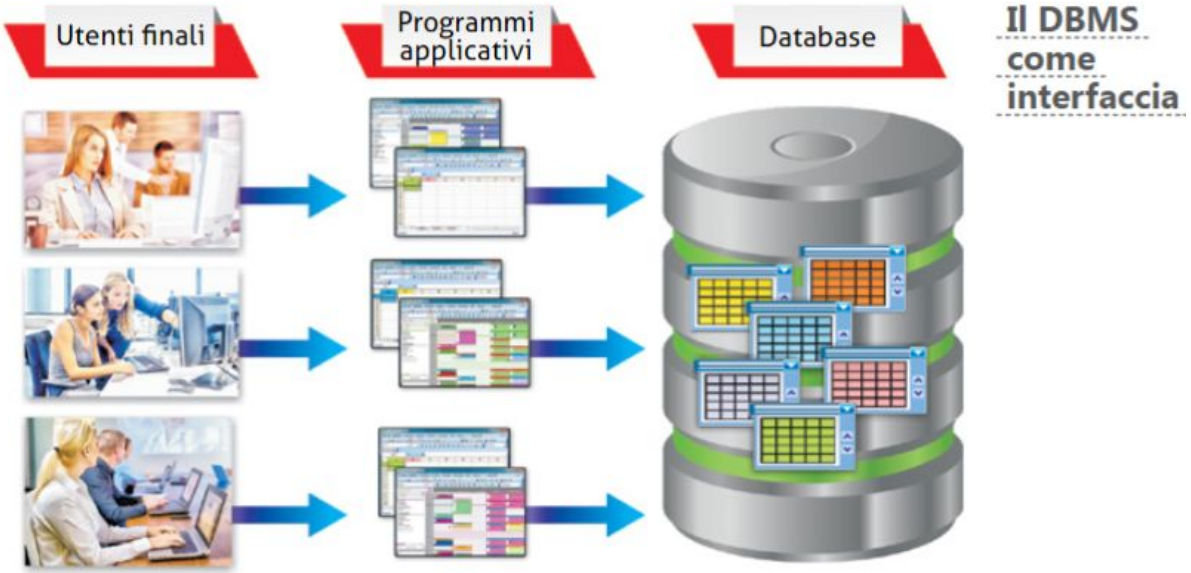
- Con **indipendenza fisica dei dati** si fa riferimento alla capacità dello schema logico di non essere influenzato da cambiamenti apportati allo schema interno dei dati.

ESEMPIO

Lo spostamento di una tabella da un dispositivo di memorizzazione a un altro, o la differente organizzazione degli archivi, non influenza lo schema logico e di conseguenza gli schemi esterni, rendendo il cambiamento trasparente per gli utenti.

La gestione del database

Il DBMS è il software che consente di costruire e gestire una base di dati, realizzandola nella pratica su memoria di massa, a partire da un progetto e da uno schema dei dati definiti a livello concettuale e tradotti poi in un modello logico dei dati



La gestione del database

Le funzioni che il DBMS, attraverso i suoi moduli software, è in grado di offrire agli utenti del database si possono raggruppare in sei categorie.

1. L'implementazione del modello logico sul sistema di elaborazione.

- Definizione dei dati e delle strutture dati derivate dallo schema logico (tipicamente le tabelle del modello relazionale), con produzione della documentazione sul modello.

La gestione del database

Definizione dei sottoschemi (viste logiche) che consentono agli utenti di accedere ai soli dati ai quali sono interessati in base alla proprie esigenze applicative.

Una vista logica è una tabella virtuale, mentre le tabelle del database sono indicate come tabelle primarie. Le viste logiche sono finestre dinamiche sulle tabelle del database, in quanto ogni modifica ai dati delle tabelle primarie si riflette sulle viste corrispondenti e viceversa.

La gestione del database

2. **La manipolazione e l'interrogazione sulla base di dati**

- Inserimento dei dati nel database e trattamento dei dati già registrati con operazioni di modifica o cancellazione.
- Interfaccia tra i programmi degli utenti (scritti con i tradizionali linguaggi di programmazione) e la base di dati, utilizzando le funzionalità del DBMS per migliorare l'organizzazione dei dati e le prestazioni dei programmi nelle operazioni di ritrovamento dei dati.
- Accesso ai dati del database attraverso interfacce grafiche e semplici comandi che facilitano l'utente non specialista.

La gestione del database

La gestione del database

3. Il controllo dell’integrità dei dati

- Integrità sui dati, in relazione ai valori che possono assumere e alle interdipendenze tra dati di differenti tabelle.
- Integrità definite dall’utente, cioè vincoli che sono specifici per un particolare database, come conseguenza di politiche commerciali dell’impresa oppure di norme legislative e fiscali.

La gestione del database

4. La sicurezza e la protezione

- Garanzia di sicurezza dei dati contro i danni causati da malfunzionamenti di componenti hardware o software o da interventi dolosi.
- Protezione dei dati da eventuali danneggiamenti per garantire l'integrità dei dati, offrendo anche la possibilità di attivare procedure di recovery in caso di perdita dei dati.
- Autorizzazione degli utenti che accedono alla base di dati e protezione dei dati dagli accessi non autorizzati.
- Controllo degli accessi al database in modo concorrente da parte di più utenti

La gestione del database

5. Il supporto alle transazioni

- Garanzia che tutte le operazioni che compongono la transazione siano completamente e correttamente eseguite oppure che non ne sia eseguita alcuna.

  Una **transazione** consiste in un insieme di operazioni di interrogazione o di modifica del database che devono essere eseguite come se fossero un’unica operazione.

La gestione del database

Un esempio di transazione è un’operazione di trasferimento di fondi tra conti correnti bancari.
L’importo trasferito deve essere tolto da un conto e aggiunto a un altro.

Entrambe le operazioni devono essere eseguite per poter affermare di avere effettuato il trasferimento e non è tollerabile che sia eseguita una sola delle due operazioni.

Il supporto alle transazioni deve garantirne il corretto completamento anche in caso di esecuzione concorrente

La gestione del database

6. L’ottimizzazione delle prestazioni

- Uso delle tecniche che ottimizzano l’occupazione della memoria di massa e i tempi di accesso ai dati registrati sui supporti di memorizzazione.
- Implementazione delle opportune strategie per ottimizzare le interrogazioni minimizzando i corrispondenti tempi di risposta o, più in generale, i costi dell’interrogazione.

La gestione del database - esempio

Le strategie messe in atto da un DBMS per ottimizzare il tempo di risposta nell'esecuzione di interrogazioni sulle tabelle *Studenti* e *Dipartimenti* (vedi pag. 28) possono essere le seguenti.
Si supponga di voler elencare gli studenti che, per frequentare le lezioni, si recano a un dato indirizzo di dipartimento. Per generare questo elenco il sistema deve combinare le due tabelle *Studenti* e *Dipartimenti* per formarne una sola, abbinando le righe di *Studenti* con quelle di *Dipartimenti* in tutti i modi possibili, scegliendo poi solo quelle combinazioni che hanno il medesimo valore nel campo *CodDip*. Della tabella ottenuta interessano le sole righe dove il campo *Indirizzo* di *Dipartimenti* ha il valore cercato e, di queste, bisogna considerare i valori di *Cognome*, *Nome* e *Matricola*.
Se nel database sono presenti i dati di 20.000 studenti e di 15 dipartimenti, per combinare le righe di *Studenti* con quelle di *Dipartimenti* il sistema deve costruire 20.000 × 15 = 300.000 abbinamenti anche se, di questi, solo 20.000 sono significativi (cioè quelli nei quali i valori dei campi *CodDip* di *Studenti* e *Dipartimenti* sono uguali). Tra essi il DBMS dovrà scegliere solo quelli con il valore cercato nel campo *Indirizzo* di *Dipartimenti*.
Un primo livello di ottimizzazione può essere attuato con la seguente strategia operativa: il sistema sceglie come prima cosa il dipartimento (o i dipartimenti) con l'indirizzo cercato e, solo successivamente, abbina gli studenti con i dati del dipartimento (o dei dipartimenti) così individuati.

La gestione del database - esempio

Quindi il numero di combinazioni tra righe delle due tabelle si riduce in modo significativo. Inoltre, come deve comportarsi il sistema per combinare in modo efficiente le righe di *Studenti* e *Dipartimenti* con un dato valore di *CodDip*? Scelto un dipartimento, per identificare gli studenti a esso associati il DBMS può procedere in due modi diversi: usare un indice costruito sul campo *CodDip* della tabella *Studenti*, per accedere rapidamente ai dati degli studenti di quel dipartimento, oppure scandire sequenzialmente l'intera tabella *Studenti*. Se gli studenti sono molte migliaia la strategia migliore è quella di usare un indice, mentre se gli studenti sono poche decine è più efficiente trasferire l'intera tabella in memoria centrale ed esaminare uno dopo l'altro i record che la compongono.

La gestione del database

Per poter attuare questo ulteriore livello di ottimizzazione, il DBMS deve disporre di informazioni sulla numerosità dei record nelle tabelle e sulla distribuzione dei dati di uno specifico campo o di un dato indice: questi dati informativi costituiscono le statistiche del database o dati statistici gestiti dal DBMS.

L'esempio mostra come le statistiche possano essere usate dal modulo di ottimizzazione delle interrogazioni per costruirle nel modo più effi ciente. Poiché la distribuzione dei dati in un campo o in un indice è un valore che cambia nel tempo (si pensi, per esempio, alla distribuzione degli studenti nei diversi dipartimenti), le statistiche sono un tipo di metadato dinamico gestito dal DBMS.

Altre funzioni del DBMS

I DBMS dispongono spesso di generatori di rapporti, cioè componenti specializzate nella produzione di report.

Si tratta, in pratica, di applicazioni progettate per presentare i dati estratti da un database in formati adatti alla consultazione da parte di una specifica categoria di utenti.

ESEMPIO

Il responsabile delle vendite di una catena di negozi potrebbe essere interessato a ricevere rapporti periodici sull'andamento delle vendite, con informazioni di sintesi sulle vendite nei singoli negozi, oppure sui differenti articoli commercializzati. La persona che ha il compito di preparare il rapporto si interfaccia con il generatore per scegliere la tabella o le tabelle che contengono i dati richiesti, definire il livello di aggregazione e i valori di sintesi desiderati, scegliere il layout della presentazione. A questo punto il sistema produce automaticamente il rapporto desiderato senza la necessità di sviluppare software specifico (vedi Capitolo 4).

La gestione del database

Un DBMS gestisce anche il dizionario dei dati (o catalogo del database), contenente informazioni su:

- nomi delle tabelle e delle colonne;
- associazioni;
- viste logiche;
- vincoli di integrità;
- utenti e proprietari;
- autorizzazioni degli accessi. Il dizionario contiene i metadati, cioè i dati che descrivono i dati contenuti nel database. Anche le informazioni del dizionario sono organizzate in modo relazionale, cioè come valori in tabelle. Gli utenti autorizzati possono quindi accedere alle informazioni del dizionario con le stesse modalità con le quali operano per ritrovare i dati nel database.

I linguaggi per i database

Le prestazioni del DBMS vengono attivate dall'utente usando appositi comandi, che costituiscono a tutti gli effetti un linguaggio attraverso il quale l'utente può comunicare con il sistema di elaborazione che gestisce il database.

I comandi che il DBMS mette a disposizione possono essere classificati nelle seguenti categorie di linguaggi.

- Linguaggio per la descrizione dei dati, delle tabelle e delle viste, delle associazioni tra tabelle, dei vincoli di integrità e dei controlli relativi alla sicurezza, detto DDL (Data Definition Language).

I linguaggi per i database

Il DDL rappresenta lo strumento attraverso il quale l'utente, facendo riferimento al proprio schema logico, ordina al DBMS la creazione della struttura fisica del database.

Il DDL possiede inoltre specifici comandi per definire i sottoschemi relativi alle applicazioni contenute nei programmi dei singoli utenti, oltre che per eliminare tabelle e viste già esistenti.

Linguaggio per il trattamento (o manipolazione) dei dati contenuti nel database, detto DML (Data Manipulation Language), che consente le usuali operazioni di accesso per inserimenti, modifiche o cancellazioni.

I linguaggi per i database

- Linguaggio per le interrogazioni alla base di dati, detto QL (Query Language), che consente il ritrovamento dei dati che interessano, sulla base dei criteri di ricerca richiesti dall'utente.

Il linguaggio denominato DDL comprende anche la possibilità di generare in modo automatico, a partire dalle tabelle e dai loro attributi, le maschere video che servono a facilitare l'utente nell'inserimento, nella modifica e nella consultazione dei dati, e i prospetti di output per la rappresentazione ordinata e facilmente leggibile dei dati estratti dalla base di dati

I linguaggi per database relazionali

Lo sviluppo e il raffinamento delle tecniche di gestione delle basi di dati hanno dato vita a linguaggi formati da comandi specifici, per consentire agli utenti un facile uso delle prestazioni del DBMS per basi di dati relazionali (detti RDBMS, cioè Relational DBMS).

Accanto alla possibilità di usare questi comandi, richiamandoli dall'interno di un programma scritto con i tradizionali linguaggi di programmazione (Cobol, C), è molto comune l'uso di linguaggi orientati alla gestione delle basi di dati, con caratteristiche di linguaggio a sé stanti.

I linguaggi per database relazionali

La diffusione del modello relazionale ha poi favorito l'uso prevalente di linguaggi non procedurali ma dichiarativi: in questo modo l'utente non ha la necessità di conoscere né le modalità con le quali le informazioni sono state fisicamente registrate, né i cammini per ritrovare le informazioni contenute nella base di dati.

I linguaggi per database relazionali

Si parla allora di linguaggio per basi di dati, intendendo un insieme completo di comandi che consente e facilita le operazioni di definizione del database, di manipolazione dei dati e di interrogazione da parte degli utenti: vengono cioè unificate in un unico linguaggio le funzioni dei linguaggi DDL, DML e QL.

Tipici linguaggi non procedurali per database relazionali sono il linguaggio **SQL (Structured Query Language, linguaggio strutturato di interrogazione)** e

QBE (Query By Example, inter-rogazioni per esempi)

I linguaggi per database relazionali

QBE permette di eseguire interrogazioni dichiarando, in un'apposita maschera, come si dovrà presentare la risposta alla query che si vuole realizzare. Nel descrivere l'output desiderato si possono effettuare calcoli partendo dai valori dei campi, oppure esprimere condizioni che devono essere verificate dai record estratti. Inoltre il QBE permette non solo di costruire interrogazioni, ma anche di modificare e variare i dati. Con il linguaggio SQL si può interrogare un database per estrarne informazioni.

Ma non solo: nonostante il nome, SQL è un linguaggio per database completo. Con SQL si possono infatti sia defi nire che manipolare dati. L'approccio dichiarativo del linguaggio è evidenziato dai seguenti esempi introduttivi.

I linguaggi per database relazionali

ESEMPIO

SQL usato come **DDL** (linguaggio di definizione dei dati) per creare lo schema di una tabella di nome *Dipartimenti*:

```
CREATE TABLE Dipartimenti (
CodDip            CHAR(10)   PRIMARY KEY,
NomeDipartimento  CHAR(30),
Indirizzo         CHAR(50),
);
```

ESEMPIO

SQL usato come **DML** (linguaggio di manipolazione dei dati) per inserire i dati del dipartimento di Medicina nella tabella *Dipartimenti*:

```
INSERT INTO Dipartimenti(CodDip, NomeDipartimento, Indirizzo)
VALUES ('Med', 'Medicina', 'Piazza Mentana');
```

I linguaggi per database relazionali

ESEMPIO

SQL usato come **QL** (linguaggio di interrogazione) per estrarre l'elenco degli studenti del dipartimento di Giurisprudenza; l'elenco contiene cognome, nome e numero di matricola:

```
SELECT Cognome, Nome, Matricola
FROM Studenti
WHERE CodDip = 'Giu';
```

Il comando SELECT cerca nella tabella *Studenti* le righe per le quali vale la condizione *CodDip* = *'Giu'* e, di queste righe, mostra i valori dei campi *Cognome*, *Nome* e *Matricola*.

I linguaggi per database relazionali

I linguaggi per database relazionali si basano sulla visione tabellare dei dati. I comandi del linguaggio relazionale operano inoltre su gruppi di righe o sull'intera tabella, anziché su una riga per volta: con una sola richiesta possono essere trattati o ritrovati molti record e non solo un record per volta, come avviene con i tradizionali linguaggi di programmazione.

Sono semplificate le operazioni per mettere in connessione tra loro tabelle diverse e per presentare sul video o stampare i risultati delle interrogazioni, ben impaginati, in modo da facilitarne la lettura e la comprensione.

I linguaggi per database relazionali

Nei software DBMS, soprattutto nelle implementazioni di prodotti su personal computer, sono disponibili interfacce utente, con le quali si può interagire usando la lingua nazionale, che permettono di selezionare i comandi del linguaggio attraverso menu, sottomenu e il puntamento a icone sul video, con le modalità caratteristiche di un'interfaccia grafica.

Sono inoltre disponibili messaggi di aiuto, che possono essere richiamati sul video in modo contestuale, cioè nel momento in cui serve una breve spiegazione sul tipo di operazione che si vuole attivare

Gli utenti

Un database viene utilizzato da persone diverse, per funzioni e per applicazioni diverse.

- 1. La responsabilità della gestione del database è affidata all'Amministratore della Base di Dati (**DBA, DataBase Administrator**), con i seguenti compiti:
- implementazione del modello logico del database nel sistema di elaborazione sui supporti fisici delle memorie di massa;
- gestione e trattamento dei dati (controllo di inserimenti, modifiche, cancellazioni);
- autorizzazione degli accessi;
- definizione delle viste per accessi parziali di utenti alla base di dati;
- controllo dei programmi applicativi che richiedono l'uso del database;

Gli utenti

- manutenzione del database nel tempo, in termini di efficienza e di ottimizzazione delle risorse;
- controllo sugli interventi di recupero, nel caso di cattivi funzionamenti, e sulle copie di salvataggio periodiche;
- controllo della disponibilità degli spazi su memoria di massa

Il DBA utilizza, per svolgere le sue funzioni, le prestazioni e i linguaggi del DBMS e collabora con le altre fi gure (sistemista, analista, responsabile dello sviluppo software, programmatore) nelle diverse fasi: progettazione del database, costruzione delle applicazioni, traduzione dello schema del database nel modello fisico da creare su memoria di massa, manutenzione dei dati nel tempo

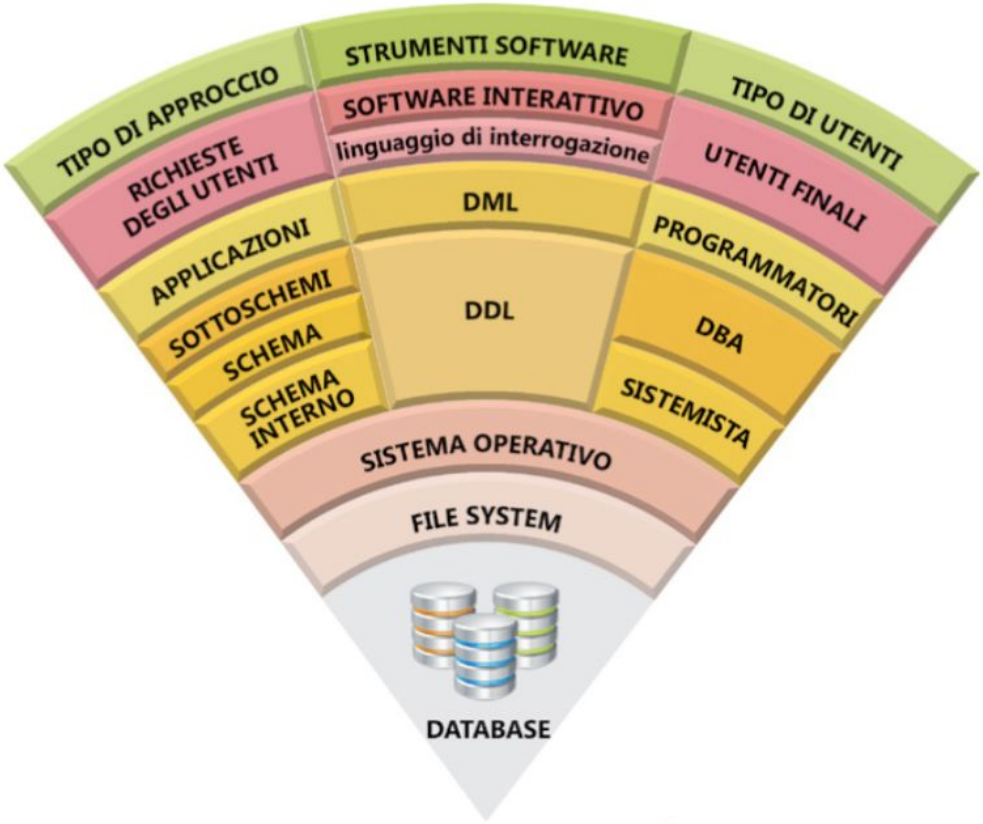
Gli utenti

2. I **programmatori**, che intendono utilizzare per le loro applicazioni i dati organizzati in un database, sfruttano un linguaggio DML, oppure comandi che sono un'estensione dei tradizionali linguaggi di programmazione, oppure un linguaggio specifico per basi di dati.

In ogni caso il programmatore deve agire sotto il controllo del gruppo di progetto e di produzione del software applicativo a cui partecipa anche l'Amministratore del database

Gli utenti

3. Gli utenti finali possono accedere alla base di dati attraverso i comandi di un linguaggio di interrogazione, di un linguaggio QBE, oppure (per utenti finali ancora meno esperti) attraverso interfacce software che presentano sul video finestre, menu e icone



Le transazioni

Una **transazione** consiste in un insieme di operazioni di interrogazione o di modifica del database che devono essere eseguite unitariamente, come se fossero un'unica operazione.

ESEMPIO

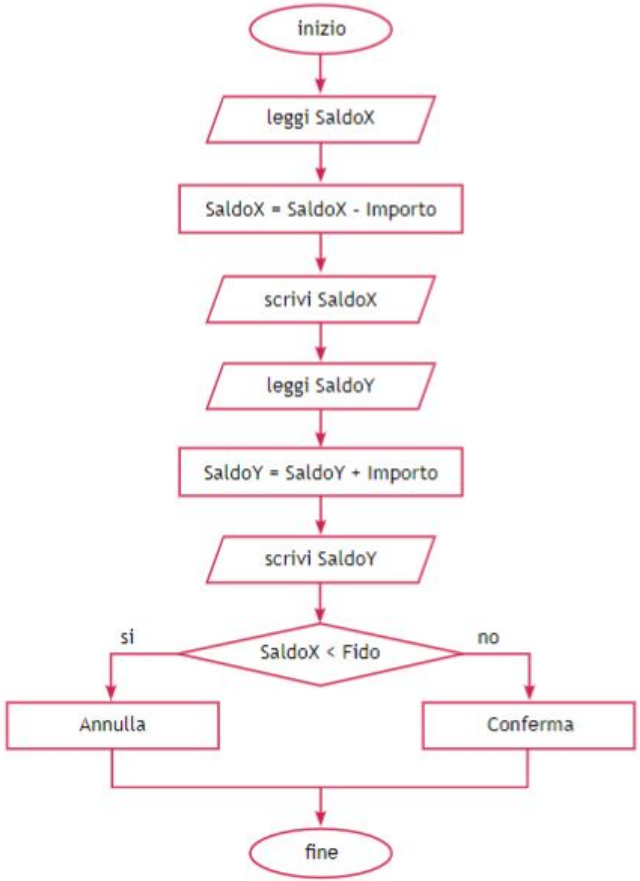
Sono transazioni: un'operazione di trasferimento di fondi da un conto a un altro, un prelievo a uno sportello Bancomat, un acquisto con carta di credito, la ricezione della merce in magazzino, la prenotazione di un volo, l'emissione del relativo biglietto.

Si consideri il trasferimento di fondi dal conto di X al conto di Y con l'utilizzo delle seguenti variabili:

- *SaldoX* e *SaldoY* indicano il valore del saldo dei conti di X e Y;
- *Importo* rappresenta l'entità del trasferimento;
- *Fido* è un valore negativo che rappresenta, cambiato di segno, il credito concesso su un conto.

Il seguente diagramma rappresenta il comportamento della transazione, il cui completamento è subordinato all'esistenza di un'adeguata copertura di fondi sul conto di X.

Le transazioni



La transazione può essere descritta attraverso un linguaggio di progetto (pseudocodifica).

Per la comprensione del codice si tenga presente che:

- il valore di Importo è acquisito in input prima dell'inizio della transazione;
- le istruzioni begin_transaction, end_transaction servono a delimitare l'insieme di azioni che compongono una transazione;

Le transazioni

- le istruzioni **read()** e **write()** indicano operazioni di accesso in lettura e, rispettivamente, in scrittura alla base di dati. Il database viene modificato solamente da istruzioni di scrittura sulla base di dati e quindi solo dalle istruzioni *write*();
- il comando **rollback** ha l'effetto di annullare tutte le modifiche alla base di dati apportate dall'inizio della transazione;
- il comando **commit** conferma tutte le modifiche apportate al database e termina la transazione.

```
begin_transaction;
    read (SaldoX);
    SaldoX = SaldoX - Importo;
    write (SaldoX);
    read (SaldoY);
    SaldoY = SaldoY + Importo;
    write (SaldoY);
    if( SaldoX < Fido ) then
        rollback;
    else
        commit;
    endif;
end_transaction;
```

rollback annulla le modifiche apportate al database

commit conferma le modifiche e termina la transazione

Le proprietà acide delle transazioni

L’esempio precedente mostra una situazione nella quale il programma, quando si accorge di non essere in grado di portare a buon fine la transazione, annulla il lavoro fatto sino a quel momento e pone fine alla transazione.

L’importo trasferito è tolto da un conto e aggiunto all’altro. Entrambe le operazioni devono essere eseguite per poter affermare di avere concluso il trasferimento e non è tollerabile che venga eseguita una sola delle due operazioni.

Le proprietà acide delle transazioni

Ne segue che il sistema di gestione delle transazioni deve intervenire per effettuare un rollback automatico nel caso che, per una qualsiasi ragione, una delle due operazioni di aggiornamento della base di dati non vada a buon fine.

Questo comportamento caratterizza un'importante proprietà delle transazioni e prende il nome di atomicità. Essa è una delle cosiddette proprietà acide delle transazioni, dall'acronimo inglese **ACID (Atomicity, Consistency, Isolation, Durability)**

Le proprietà acide delle transazioni

Le proprietà acide delle transazioni

- **Atomicità**: una transazione è un’entità atomica indivisibile. È compito del sistema di gestione della sicurezza garantire l’atomicità e riuscire a ripristinare la situazione preesistente quando necessario;
- **Consistenza**: le transazioni non devono violare i vincoli di integrità dei dati. La consistenza è gestita dal DBMS con procedure opportune;• Isolamento: gli effetti di una transazione devono essere indipendenti da quello di tutte le altre transazioni eseguite in concorrenza. L’isolamento è gestito dal controllore della concorrenza.

Le proprietà acide delle transazioni

- **Persistenza**: le informazioni in un database devono essere memorizzate in modo persistente, cioè per sempre. Gli effetti di una transazione eseguita con esito positivo (dopo l'esecuzione di un comando commit) devono essere memorizzati permanentemente nel database a cura del sistema di gestione della sicurezza e del ripristino dei dati;