Introduzione ai sistemi di Basi di Dati

c.vallati@iet.unipi.it

Sommario

- Gestione dei Dati nei sistemi informativi
- Basi di dati e DBMS
- Approccio tradizionale: File System
- I vantaggi del DBMS: Il caso Ford
- Modelli di dati
- Utenti di un DBMS
- Progettazione di una base di dati
- Architettura generale di un DBMS

Gestione dei dati

- Dato, definizione: '*In informatica, la singola informazione codificabile o codificata*'
- L'evoluzione della tecnologia permette oggi di raccogliere una mole di dati, continuamente e in maniera pervasiva, e.g. smart-watch, cellulare
- Questo ha solo portato alle estreme conseguenze una necessità, come la *corretta e efficiente gestione dati*, che è sempre stata fondamentale per la buona realizzazione di qualsiasi sistema informativo

Sistema Informativo

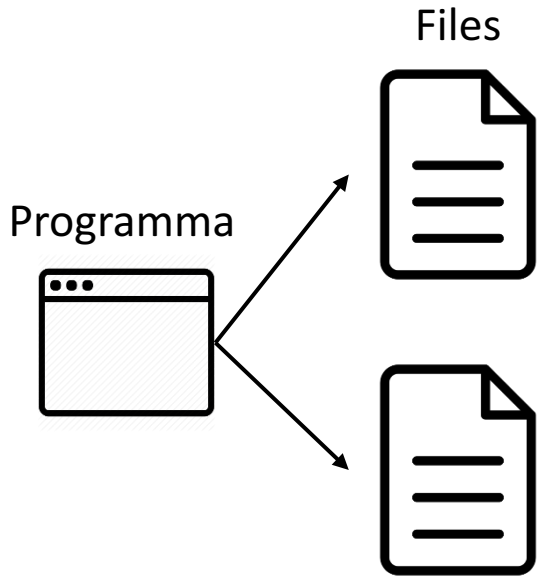
- Il Sistema Informativo è un sistema atto a organizzare e gestire i dati in maniera tale da permettere la loro facile elaborazione per ottenere informazioni
- I sistemi informativi venivano realizzati anche prima dell'invenzione e diffusione dei calcolatori elettronici, e.g. archivi delle banche e servizi anagrafici
- Per la porzione automatizzata del sistema informativo, al giorno d'oggi viene usato il termine Sistema Informatico, termine usato oggi per contraddistinguere tutti i sistemi informativi

Database Management System - DBMS

- All'interno del sistema informativo, la collezione dei dati è chiamata ***Base di Dati*** o ***Database***
- Compito della base di dati è non solo di memorizzare i dati ma di rappresentare le relazioni tra di essi
- All'interno del sistema informativo il software atto specificatamente a gestire i dati è detto ***Sistema di Gestione della basi di dati*** o ***Database Management System***
- Tradizionalmente adottato nei sistemi informativi di grandi dimensioni (solitamente composti da più programmi), oggi adottato anche da sistemi più semplici

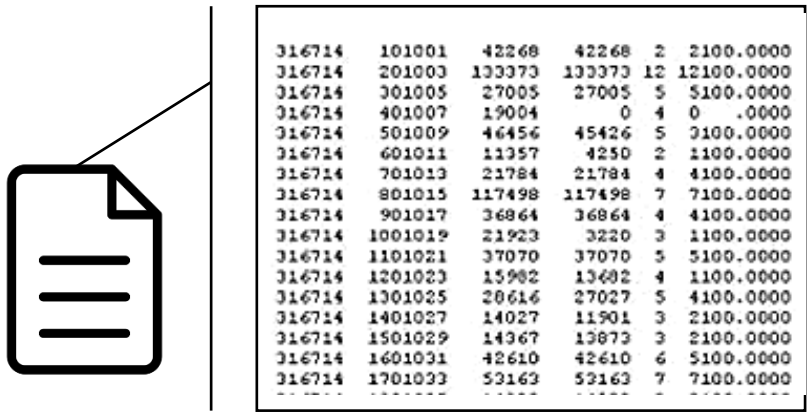
Archivio basato su Files

- L'approccio classico usato dal/dai programma/i che compongono il sistema informativo per la gestione delle informazioni è un **archivio basato su files**
- Ogni programma ha accesso al **file system** gestito dal sistema operativo per creare uno o più files (archivi)

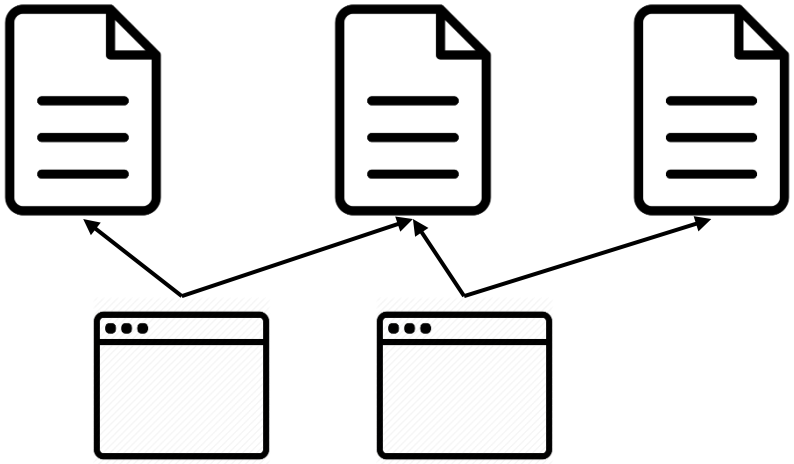


Archivio basato su File

- Ogni file è un insieme di registrazioni (record) all'interno dei quali sono memorizzati i dati elementari (attributi e campi)
- Condivisione di dati tra più programmi può essere fatto tramite l'uso di file condivisi



316714	101001	42268	42268	2	2100.0000
316714	201003	133373	133373	12	12100.0000
316714	301005	27005	27005	5	5100.0000
316714	401007	19004	0	4	0 .0000
316714	501009	46456	45426	5	9100.0000
316714	601011	11357	4250	2	1100.0000
316714	701013	21784	21784	4	4100.0000
316714	801015	117498	117498	7	7100.0000
316714	901017	36864	36864	4	4100.0000
316714	1001019	21923	3220	3	1100.0000
316714	1101021	37070	37070	5	5100.0000
316714	1201023	15982	13682	4	1100.0000
316714	1301025	28616	27027	5	4100.0000
316714	1401027	14027	11901	3	2100.0000
316714	1501029	14367	13873	3	2100.0000
316714	1601031	42610	42610	6	5100.0000
316714	1701033	53163	53163	7	7100.0000

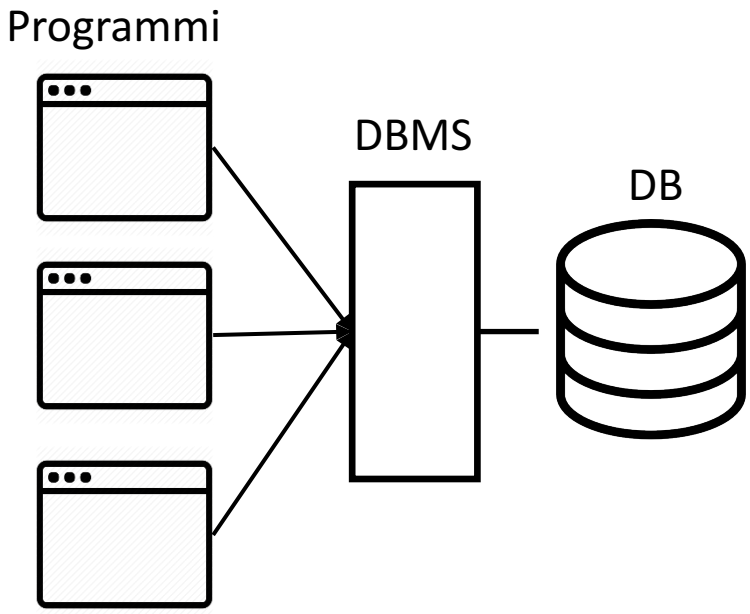


Uso file - Svantaggi

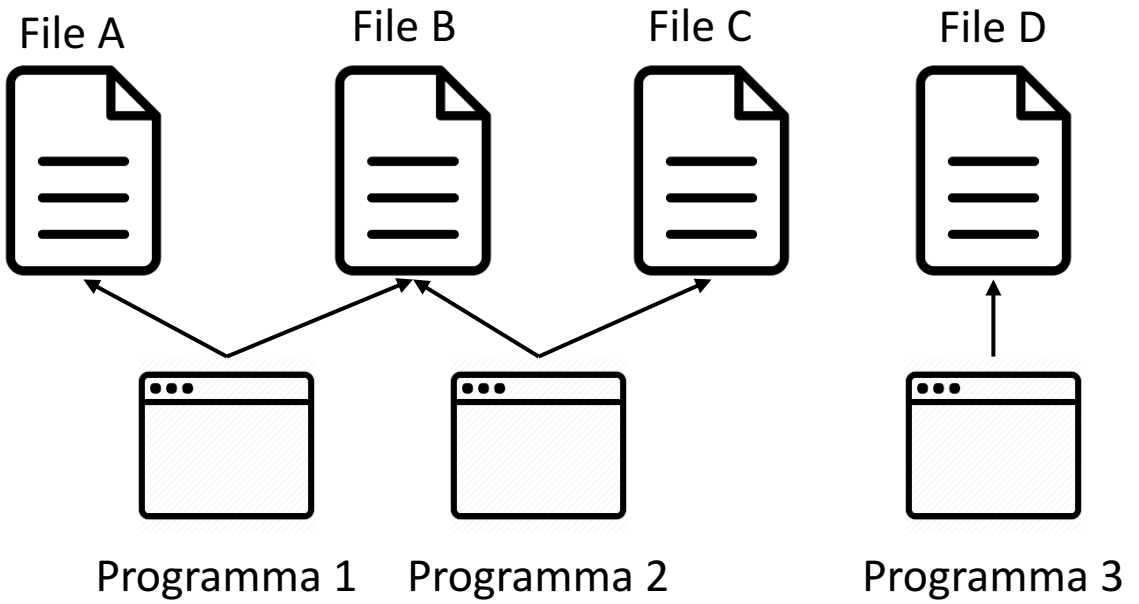
- I file possono avere diversi **formati incompatibili** tra di loro, i programmi si devono adeguare a diverse convenzioni anche a distanza di parecchio tempo. Questo rende la **condivisione** dei dati attraverso applicazioni differenti **difficoltosa**
- I dati se non memorizzati su file condivisi sono **replicati** con spreco di risorse di memorizzazione e possibili problemi legati a inconsistenze
- L'accesso a file in condivisione porta a dover gestire la **concorrenza con soluzioni ad-hoc** (specialmente se due o più programmi vogliono modificarne il contenuto)

Approccio basato su DBMS

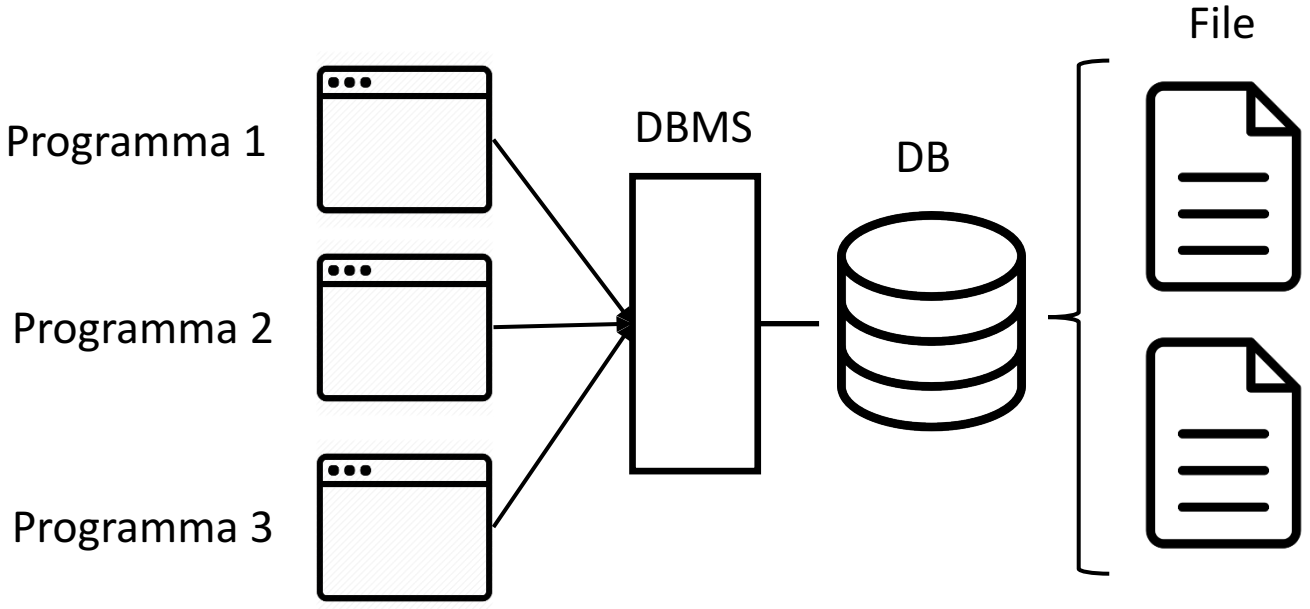
- L'approccio basato su DBMS invece va oltre l'uso di file locali gestiti dalle singole applicazioni tramite l'adozione di un *sistema di gestione dei dati* che risulta **indipendente** dalle applicazioni e **specializzato** in tale funzione
- I dati non sono gestiti dalle singole applicazioni ma da un DBMS che offre **un'interfaccia comune** a tutte le applicazioni
- Si interpone fra le applicazioni e la *memoria di massa*
- I dati non appartengono ad una singola applicazione, ma esse vi accedono attraverso il DBMS



Approccio basato su file



Approccio basato su DBMS



DBMS - Vantaggi

- Le basi di dati sono **condivise**: le applicazioni e gli utenti a dati comuni evitando la realizzazione ad-hoc di soluzioni come la condivisione di file tra applicazioni diverse. Questo aiuta a:
  - Ridurre la **ridondanza**: una base di dati centralizzata permette di ridurre la replica della stessa informazione che si avrebbe se le diverse applicazioni gestissero i dati tramite file locali
  - Ridurre l'**inconsistenza**: l'eliminazione della presenza di varie copie dello stesso dato elimina la possibilità di inconsistenze, la gestione attraverso una componente specializzata permette di introdurre controlli sui dati per garantirne la consistenza

DBMS - Vantaggi

- DBMS sono componenti software specializzati nel gestire grandi quantità di dati e implementano *procedure basate sulle best-practices* (solitamente non implementate nelle soluzioni basate su file) per la gestione di:
  - **Efficacia e efficienza**: le tecniche di memorizzazione adottate permettono di migliorare le prestazioni di memorizzazione e accesso alle informazioni (che altrimenti dovrebbero essere implementate in ogni programma)
  - **Affidabilità**: tecniche di salvaguardia e verifica dell'integrità dei dati in caso di malfunzionamenti hardware e software (crash recovery) sono solitamente implementate
  - **Concorrenza**: i sistemi DBMS implementano delle metodologie per garantire un accesso concorrente ai dati minimizzandone l'impatto sulle prestazioni di accesso (esempio limitando i tempi di attesa in seguito alla mutua esclusione su un dato)
  - **Privatezza**: tecniche di sicurezza per garantire accesso ristretto sono implementate in modo da garantire a ciascun utente accesso solo al sottoinsieme dei dati a cui è autorizzato

DBMS - Vantaggi

- L'utilizzo di una componente per la gestione dei dati di diverse applicazioni permette di migliorare il processo di realizzazione delle applicazioni:
  - **Riduzione del tempo di sviluppo**: invece di implementare le funzionalità di gestione dei dati ogni applicazione si appoggia su quelle fornite dal DBMS
  - **Semplificazione e standardizzazione dello sviluppo**: il processo di realizzazione delle applicazioni viene semplificato dato che la memorizzazione e la gestione dei dati è demandato ad una componente con la quale l'applicazione interagisce tramite un'interfaccia standard

DBMS nei processi aziendali

- I diversi settori in cui si articola una grande organizzazione possono trarre beneficio dalla gestione **integrata** e **condivisa** dell'informazione offerta dal DBMS
- Quest'ultima in particolare può essere sfruttata per *reingegnerizzare i processi aziendali* sfruttando il coordinamento prima assente tra diverse applicazioni
- Esempio: *il caso Ford*

Il caso Ford

Nei primi anni '80, la Ford cercava di ridurre le proprie spese amministrative, e uno dei settori in cui appariva possibile tagliare i costi era quello della **contabilità fornitori** che in quel tempo ammontava a circa 500 persone.

Il processo iniziava con l'invio da parte dell'ufficio approvvigionamenti di un *ordine d'acquisto* al fornitore, con relativa copia per la contabilità; quando il fornitore spediva la merce e questa arrivava all'azienda, un impiegato del ricevimento merci riempiva un *modulo con la descrizione degli articoli* e lo mandava alla contabilità fornitori. Infine il fornitore inviava la *fattura*. La contabilità fornitori operava quindi con tre documenti che si riferivano alla stessa partita di merce: ordine d'acquisto, modulo di ricevimento, fattura. *Se non si riscontravano discordanze, un impiegato disponeva il pagamento.*

Tuttavia il processo conteneva numerose anomalie. Infatti gli impiegati passavano la maggior parte del tempo a risolvere quei pochi casi in cui i documenti - ordine d'acquisto, modulo di ricevimento e fattura - erano **divergenti**. A volte occorrevano settimane intere e molti sforzi per venirne a capo.

Il caso Ford

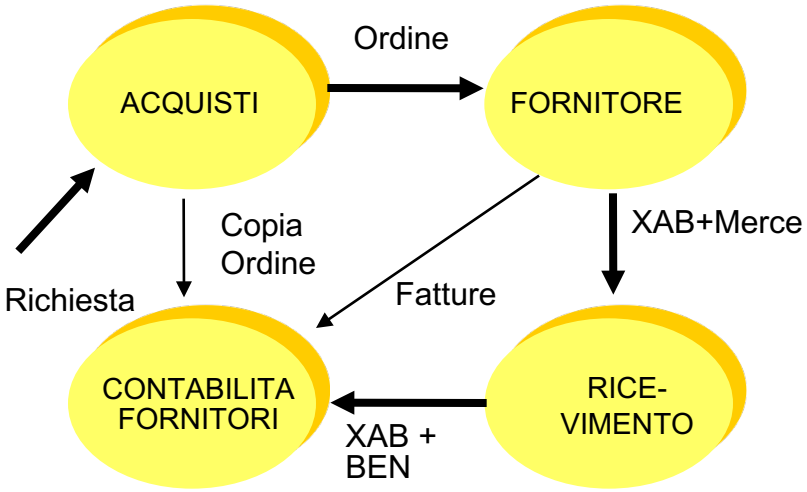
Utilizzando i computer per **automatizzare** *alcune* funzioni, il management era convinto di tagliare del 20 per cento il numero dei dipendenti del reparto, riducendolo a 400 unità.

Ford aveva comprato il 25 per cento delle azioni di Mazda. Mazda riusciva a gestire il pagamento delle fatture dei fornitori con uno staff di 5 persone. Il divario di personale - 500 persone in Ford contro 5 in Mazda - era troppo grande per potere essere giustificabile solo dalle diverse dimensioni.

La riduzione del 20 per cento del personale non avrebbe messo Ford in parità con Mazda. *Ford si vide costretta a ripensare l'intero processo* cui il reparto di contabilità dei fornitori era coinvolto.

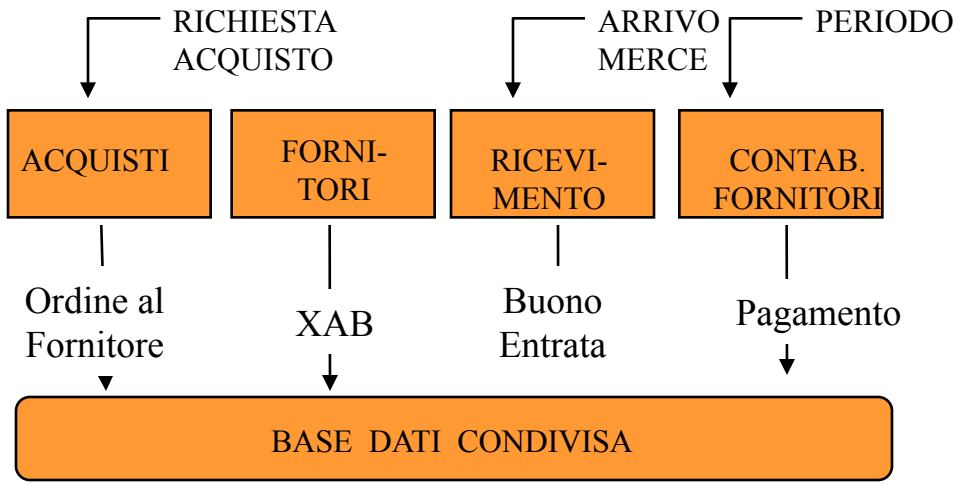
Il caso Ford - Processo Aziendale

AS IS



- *3 documenti da accoppiare:*
  - Fattura
  - Nota di accompagnamento (XAB) e Buono Entrata (BEN)
  - Ordine al Fornitore (ORFOR)
- *20% ordini e consegne impegnano lo 80% del tempo*

TO BE



- *No flussi cartacei*
- *Pagamento su avanzamento programma*
- *Riduzione di leadtime per effetto della riduzione dei tempi di appuntamento, dei polmoni ed altri*

DBMS – Prodotti commerciali

- I DBMS non sono software ad-hoc ma sono solitamente prodotti commerciali
- Le aziende produttrici solitamente coinvolgono decine di sviluppatori con progetti che durano svariati anni al fine di garantire la solidità del software richiesta dagli utenti finali
- *Esempi commerciali*: Oracle, IBM DB2, Microsoft SQL Server
- In aggiunta esistono DBMS open-source che vengono sviluppati e gestiti da una comunità di sviluppatori e sono liberamente accessibili
- Questi DBMS pur avendo molte delle funzionalità offerte dai prodotti commerciali non sono adatti a gestire grandi quantità di dati, ma possono invece essere usati in progetti più piccoli
- *Esempi open-source*: MySQL, MariaDB, Firebird SQL, PostgreSQL

Modello dei Dati

- I DBMS non sono progettati per gestire un unico caso d'uso, al contrario sono software in grado di gestire dati eterogenei
- Al fine di creare e gestire la corrispondente base di dati uno **schema** dei dati deve essere fornito al DBMS
- Lo schema viene costruito secondo un modello di dati ben definito. Un **Modello di Dati** è una collezione di costrutti usati per descrivere lo schema dei dati, le loro relazioni e i vincoli di consistenza che devono essere applicati sugli stessi
- Tramite questo schema dei dati si fornisce al DBMS una rappresentazione dei dati, in modo tale da permettere l'organizzazione della gestione

Modello dei Dati Relazionale

- Esistono diverse tipologie di modelli logici definiti nel tempo, e.g. *Modello Gerarchico, Modello a Oggetti, Modello Reticolare*
- Il modello attualmente più diffuso è il **Modello Relazionale**
- Il modello relazionale descrive lo schema di una base di dati attraverso la specifica delle *relazioni* che i dati stessi hanno tra di loro
- Il costrutto base del modello relazionale è la relazione o tabella relazionale che può essere pensata come un *insieme di record*
- Nel modello relazionale ciascuna relazione viene identificata da un nome
- I campi della relazione vengono identificati attraverso un nome e il tipo

Docenza

Nome Corso	Docente
Impianti	Rossi
Informatica	Verdi

Schemi e Istanze

- La struttura di un database, descritta tramite il modello dei dati è detta **schema**
  - Lo schema è frutto del processo di progettazione della base di dati e raramente cambia durante la vita del DBMS
- L'**istanza** di un database invece è l'insieme delle informazioni contenute in un database in un certo istante
  - L'istanza cambia continuamente nel tempo
  - L'inserimento/modifica/cancellazione dei dati non cambia comunque la struttura generale

Astrazioni sui dati

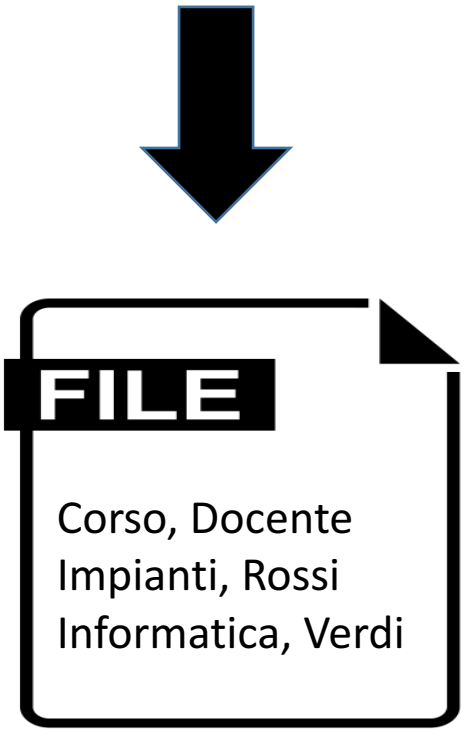
- Altro vantaggio dei DBMS è il loro supporto su *astrazioni sui dati*
- La descrizione dello schema di una base di dati attraverso il modello relazionale ad esempio fornisce un'astrazione sui dati rispetto a come, ad esempio, questi verranno memorizzati realmente su file
- Tramite diversi livelli di astrazioni, il DBMS maschera dettagli implementativi, al fine di semplificare l'interazione di diversi tipi di utente (alcuni anche non esperti) con il sistema

Livelli di astrazione

- I sistemi DBMS sono caratterizzati da tre livelli di astrazione: Fisico, Logico e delle Viste. Per ciascun livello di astrazione esiste uno schema
- Livello Logico:
  - È il livello di astrazione rappresentato dal modello dei dati (e.g. dal modello relazionale). Associato al livello logico abbiamo lo **schema logico** che fornisce una descrizione dell'intera base di dati per mezzo del modello logico adottato dal DBMS. *Sono tralasciati tutti i dettagli implementativi*
- Livello Fisico:
  - È il livello più basso di astrazione, quello interno del database. Lo **schema interno** associato alla rappresentazione dei dati al livello fisico descrive come i dati rappresentati nello schema logico sono memorizzati nelle strutture fisiche di memorizzazione.

Docenza

Corso	Docente
Impianti	Rossi
Informatica	Verdi



Livelli di astrazione

- Livello delle viste:
  - È il livello di astrazione più alto che permette di esporre agli utenti finali delle visioni (**viste**) parziali o successivamente elaborate del livello logico. Vengono definite quando non tutti gli utenti hanno necessità (o diritto) di conoscere tutta la struttura completa logica del database ma solo parti di effettivo interesse. Lo **schema esterno** costituisce la descrizione della porzione della base di dati di interesse
  - Le viste possono essere usate per regolare meglio il controllo degli accessi o per calcolare dinamicamente nuovi dati a partire da quelli memorizzati nel DB evitando ridondanza

Vista Docente

Nome	Media
Rossi	28.7
Verdi	25

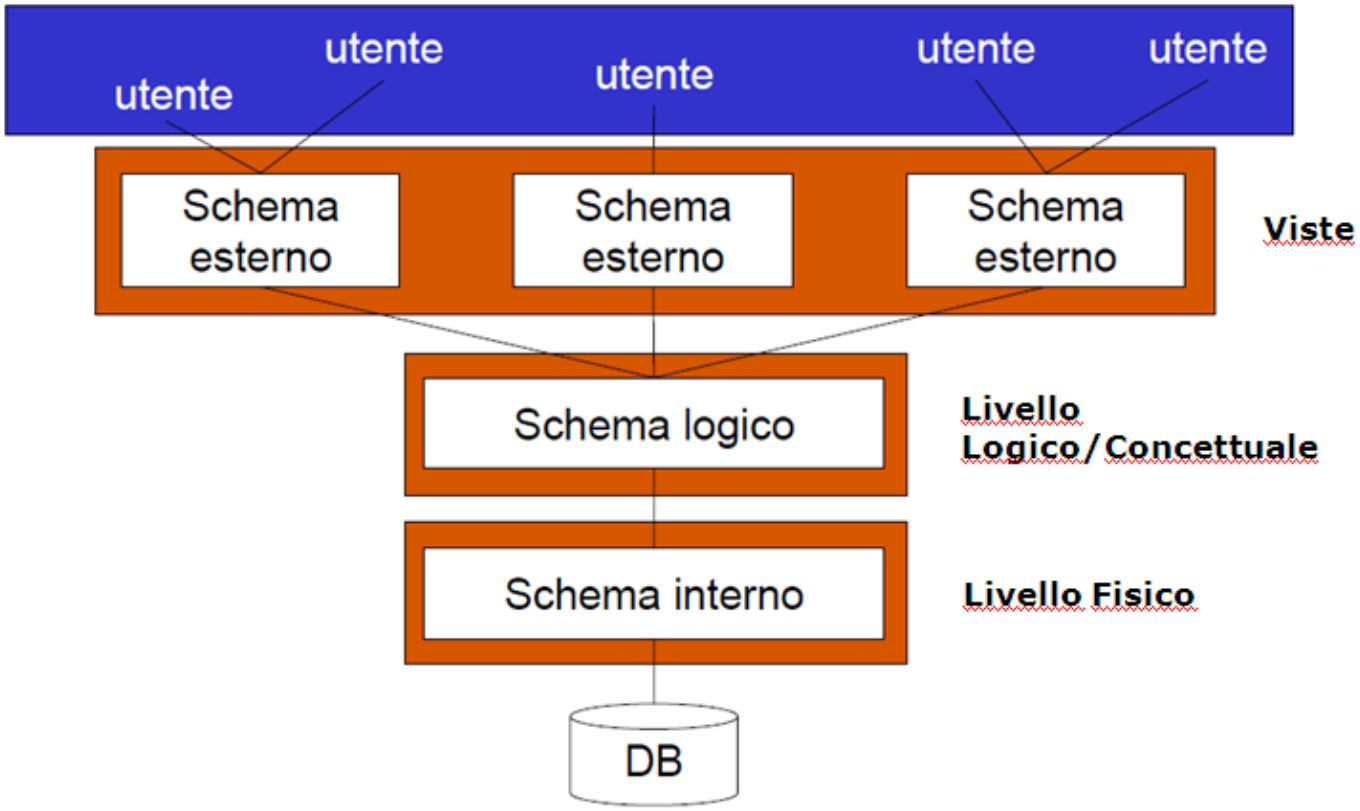
Vista Amministrativo

Nome	Indirizzo
Rossi	Via Battisti
Verdi	Corso Italia

Studenti

Nome	Media	Indirizzo
Rossi	28.7	Via Battisti
Verdi	25	Corso Italia

Livelli di astrazione



Indipendenza dei dati

- I livelli di astrazione implementati nell'architettura dal DBMS permettono di garantire l'*indipendenza dei dati*
- Gli utenti e i programmatori che utilizzano una base di dati interagiscono con essa indipendentemente dai suoi dettagli implementativi
- In particolare abbiamo:
  - Indipendenza Fisica: gli utenti interagiscono con il DBMS indipendentemente dalla struttura fisica usata per memorizzare i dati. Questo permette di cambiare la struttura di memorizzazione fisica in maniera trasparente rispetto agli utenti o i programmi
  - Indipendenza Logica: gli utenti interagiscono con le viste in maniera indipendente dal livello logico. Questo permette di aggiungere nuove viste per accomodare i requisiti di nuovi utenti senza modificare lo schema logico. In aggiunta è possibile cambiare lo schema logico mantenendo inalterate le viste

Utenti di un DBMS

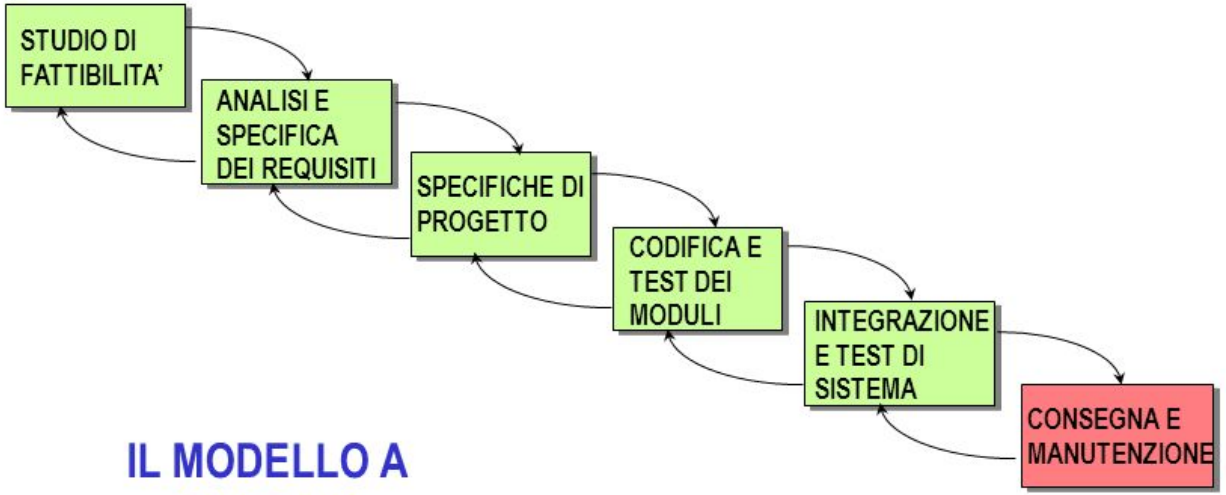
- Gli utenti di un DB possono essere suddivisi in diverse tipologie, in base al ruolo e alle informazioni a cui vogliono accedere
- A ciascuna tipologia di utenti viene associate autorizzazioni distinte:
  - Data Base Administrator:
    - Hanno il compito di progettare la base di dati e definirne lo schema e i vincoli di integrità sui dati
    - Hanno il compito di definire le politiche di accesso ai vari utenti e specificare i loro permessi
  - Progettisti e programmatori di applicazioni:
    - Sono gli utenti che realizzano i programmi e le applicazioni che accedono alla base di dati
    - Essi usano il linguaggio di manipolazione dei dati per accedere al DBMS, possono aver accesso allo schema logico direttamente o ad una vista per accedere ai dati.
  - Utenti Finali:
    - Sono gli utenti (o terminalisti) che utilizzano la base di dati indirettamente attraverso i programmi

Linguaggi

- Il DBMS mette a disposizione diversi linguaggi per effettuare operazioni di vario tipo sulla base di dati
- Questi linguaggi si distinguono in due categorie in base allo scopo:
  - Linguaggi di definizione dei dati o Data Definition Language (DDL), utilizzati dai Data Base Administrators per definire gli schemi (logici, fisici e esterni) e le autorizzazioni per l'accesso ai vari utenti
  - Linguaggi di manipolazione dei dati o Data Manipulation Language (DML), utilizzati dai programmatori e dagli utenti finali per interrogare o aggiornare un'istanza della base di dati
- L'accesso ai dati può essere effettuato tramite delle interfacce oppure direttamente nei vari programmi scritti in diversi linguaggi utilizzando apposite librerie
- In entrambi i casi le operazioni da svolgere sono descritte tramite DDL o DML
- Il linguaggio SQL (il linguaggio che copre questo corso) è un linguaggio completo che include funzionalità DDL e DML allo stesso tempo

Progettazione

- Come tutti i sistemi software, lo sviluppo di un sistema informativo complesso passa attraverso diverse fasi di progettazione e sviluppo



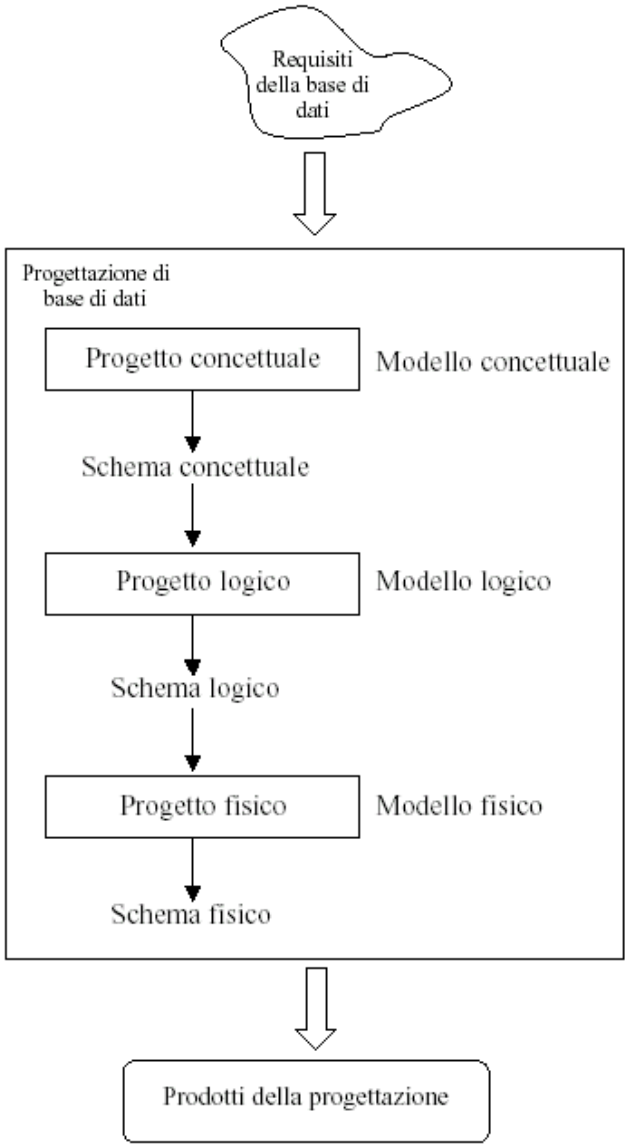
IL MODELLO A CASCATA

DBMS - Progettazione

- All'interno di questo workflow si inserisce anche la progettazione della base di dati
- A differenza del software, la base di dati non va progettata e realizzata ex-novo (lo sviluppo di un DBMS richiede anni di sviluppo e elevate competenze) ma solitamente si acquista
- *Progettare una base di dati significa definirne i vari schemi per definire come i dati reali verranno memorizzati in essa*

Progettazione

- La progettazione della base di dati si inserisce nel workflow della progettazione del software integrandosi in ogni fase passo dopo passo
- Si parte con un'analisi dei requisiti in cui si individuano i requisiti che la base di dati dovrà avere rispetto al sistema informatico di cui ne farà parte (descrizione informale di quello che viene chiesto alla base di dati)
- A partire dai requisiti si effettua una progettazione in tre fasi: una prima in cui si prende una decisione su *cosa* rappresentare nella base di dati, altre due fasi che seguono per decidere *come* farlo



Progettazione concettuale

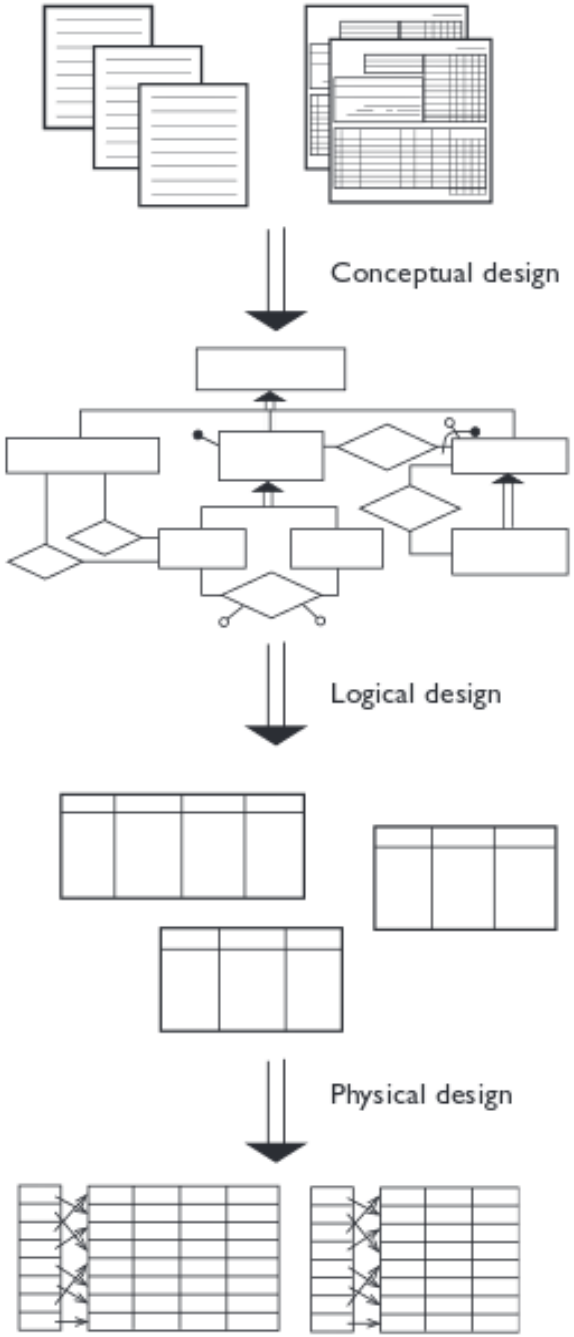
- Lo scopo di questa prima fase è collezionare le *specifiche della realtà di interesse* in termini di una descrizione formale completa, *ma indipendente dai criteri di rappresentazione utilizzati nel DBMS*
- Il prodotto di questa fase è uno *schema concettuale* dei dati costruito secondo un *modello concettuale*
- In altre parole in questa fase si descrive la realtà che la base di dati dovrà rappresentare, il suo *contenuto informativo*
- Uno dei modelli più adottati è il modello concettuale *entità – relazione* (o semplicemente modello ER)

Progettazione Logica

- A partire dallo schema concettuale avviene la progettazione logica della base di dati, che consiste nella traduzione dello schema concettuale nel modello di rappresentazione dei dati adottato dal sistema di gestione della base di dati a disposizione
- Il prodotto di questa fase è lo *schema logico* secondo un *modello logico* dei dati adottato dalla base di dati adottata
- In altre parole in questa fase si progetta lo schema della base di dati a partire dalla descrizione concettuale della realtà, in maniera indipendente da come poi i dati verranno fisicamente memorizzati
- Uno dei modelli più adottati è il *modello relazionale*

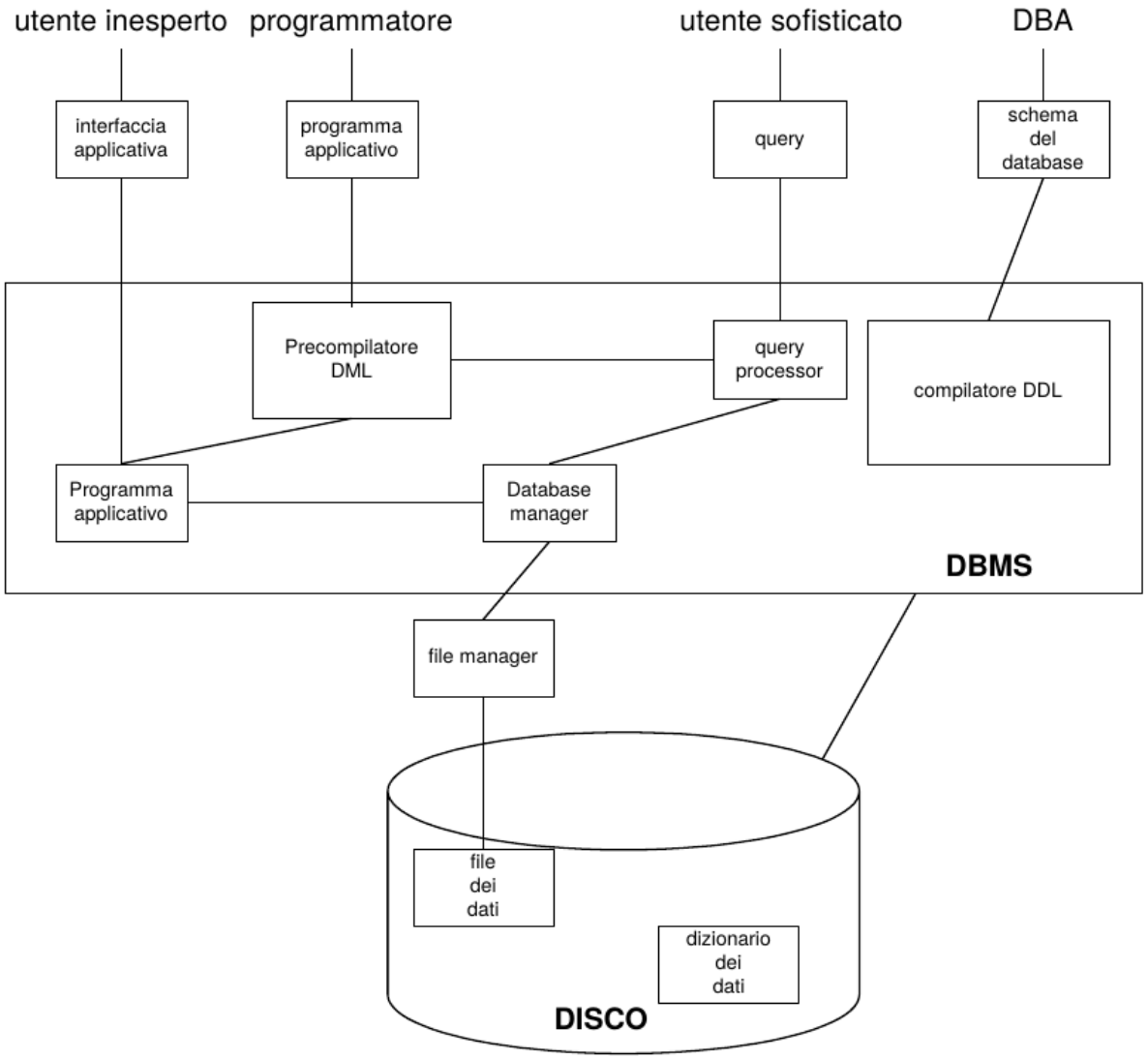
Progettazione Fisica

- Fase di progettazione tecnologica in cui lo schema logico viene completato con la specifica dei parametri fisici di memorizzazione dei dati
- Il prodotto di questa fase è lo *schema fisico dei dati*
- In altre parole in questa fase si definisce come i dati poi verranno fisicamente memorizzati (organizzazione dei file e degli indici)
- Questa fase strettamente dipendente dal DBMS adottato è solitamente nascosta al progettista


Conceptual design
Logical design
Physical design

Architettura di un DBMS

Il DBMS è un software complesso composto da diversi moduli



Architettura

- Il **Database Manager** è il core del sistema
- Gestisce la sicurezza, la concorrenza, l'imposizione dei vincoli di integrità e le funzionalità di backup provvedendo al recupero delle situazioni di errore
- Interagisce con il file manager che gestisce la memorizzazione fisica dei dati

Architettura

- Il **File Manager** gestisce l'allocazione dello spazio su disco e le strutture dati usate per rappresentare l'informazione memorizzata sul disco
- Il **Query Processor** traduce le istruzioni del DML in istruzioni a basso livello che il Database Manager è in grado di capire. Si occupa inoltre delle strategie di ottimizzazione, ossia traduce le interrogazioni dell'utente/programma in interrogazioni equivalenti ma di più rapida esecuzione
- Il **Precompilatore** e **Compilatore DDL** converte gli statement DDL in un insieme di tabelle contenenti meta-dati ossia "dati sui dati"

Architettura

- Al livello fisico abbiamo:
  - Il **Data Dictionary** conserva le informazioni sulla struttura del database
  - Gli **Indici** sono delle strutture ausiliarie utilizzate per accelerare il reperimento delle informazioni

Interfaccia

- L’interfaccia esposta dal DBMS verso applicazioni permette ad utenti esperti e programmi di interagire con il DBMS usando linguaggi DML e DDL
- Tale interfaccia solitamente è un prompt (o console) a riga di comando all’interno del quale possono essere digitati i comando (dagli utenti esperti o dall’amministratore) nel linguaggio
- Al fine di facilitare la creazione di sistemi complessi in cui il DBMS e le applicazioni sono installate ed eseguite su sistemi diversi, tali interfacce possono essere interrogate anche in remoto attraverso protocolli di comunicazione, permettendo di interrogare il DBMS anche attraverso una rete locale o Internet