Unità didattica

A1 Il mondo delle basi di dati

Prerequisiti

- Ciclo di vita del software.
- Concetto di tipo di dato, di classe e di oggetto.

Obiettivi

- Acquisire la terminologia e i concetti introduttivi sulle basi di dati.
- Acquisire una definizione di "base di dati" che comprenda anche le regole e le autorizzazioni d'accesso.
- Acquisire le basi della metodologia per la progettazione di basi di dati.

Conoscenze da apprendere

- Conoscere la differenza tra sistema informativo e sistema informatico.
- Conoscere la differenza tra significato intensionale ed estensionale dei dati.
- Conoscere il concetto di schema e istanza di una base di dati.
- Conoscere il concetto di DDL e DML per un linguaggio per basi di dati.
- Conoscere la progettazione concettuale, logica e fisica.

Competenze da acquisire

- Saper distinguere tra schema e istanza.
- Saper distinguere tra metodi di progettazione per basi di dati, linguaggi per basi di dati e sistemi di gestione delle basi di dati.
- Saper individuare modelli, linguaggi e sistemi di gestione di una base di dati.
- Saper individuare le varie fasi della progettazione di una base di dati.

1 Introduzione

In ogni modello di organizzazione della vita dell'uomo, dal più complesso al più semplice, vengono trattate **informazioni**.

Queste informazioni sono patrimonio di ogni organizzazione e vengono considerate risorse preziosissime, grazie alle quali la stessa organizzazione può sopravvivere.

Una volta *individuate* e poi *raccolte*, tali informazioni devono necessariamente essere memorizzate, in modo che ogni individuo dell'organizzazione, purché abbia le necessarie autorizzazioni, possa facilmente:

- *recuperarle* in base a determinati criteri di ricerca;
- *aggiungerne* di nuove;
- *modificarle*, apportando nel tempo variazioni;
- *cancellare* quelle non più necessarie.

La necessità di raccogliere informazioni per utilizzi successivi era già sentita

molto prima dell'avvento dei computer, ma solo l'informatica ha permesso di realizzare gli strumenti idonei alla gestione di **grandi quantità** di informazioni.

Pensa a quante informazioni sono necessarie per gestire un ufficio anagrafico comunale, oppure la contabilità di una grossa azienda o per rendere efficiente un sistema di prenotazione aereo o ferroviario.

Teoria delle Basi di Dati

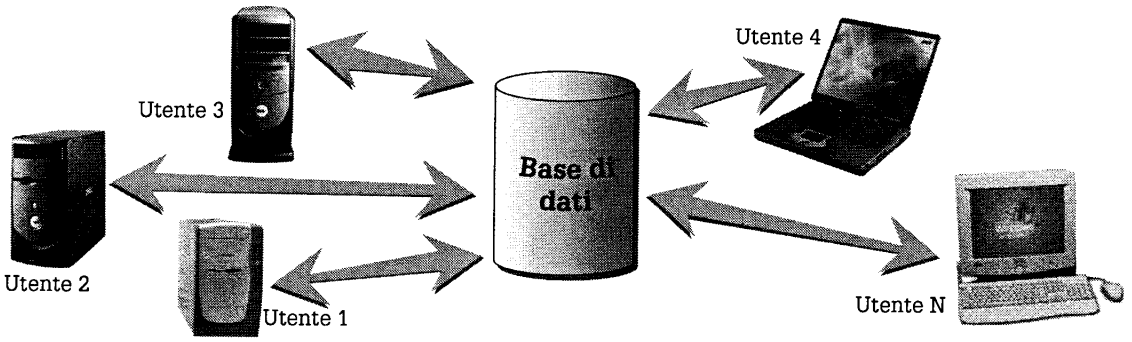
In campo informatico, la **Teoria delle Basi di Dati** studia come organizzare al meglio grandi quantità di informazioni, per poterle gestire in modo:

- *semplice*, in quanto le informazioni devono poter essere facilmente fruibili in applicazioni differenti e da parte di differenti utenti;
- *efficiente*, perché l'utilizzo delle risorse deve essere ottimizzato:
  - "in tempo" (efficiente utilizzo del microprocessore);
  - "in spazio" (efficiente utilizzo della memoria);
- *efficace*, nel senso che le informazioni devono essere rappresentative della realtà che si vuole analizzare (ad esempio: un'età non può essere negativa);
- *sicuro*, in quanto le operazioni sui dati sono permesse solo a soggetti identificabili e autorizzati.

Da quanto detto possiamo dare una prima definizione di base di dati:

> una **base di dati** (in inglese **database**; noi li utilizzeremo da ora in poi entrambi) può essere considerata come una raccolta di dati progettati in modo da poter essere fruiti in maniera ottimizzata da differenti applicazioni e utenti diversi.

La *Teoria delle Basi di Dati* si basa su *dati* più che su *informazioni* e opera nell'ambito di un **sistema informatico aziendale**; questi due aspetti sono chiariti nei prossimi due paragrafi.



✦ Figura A1.1 Interazione di più utenti con una base di dati

2 Le informazioni nei sistemi informativi e nei sistemi informatici

Sistema informativo

Le informazioni necessarie a un'organizzazione sono gestite da un **sistema informativo**.

> Un **sistema informativo** è un insieme organizzato di strumenti automatici, procedure manuali, risorse umane e materiali, norme organizzative, orientato alla *gestione delle informazioni rilevanti per un'organizzazione.*

Esempio

Consideriamo la seguente *organizzazione*: il team di vendita di una piccola ditta artigianale cui fa capo il signor Rossi.
Un esempio di *informazioni rilevanti per questa organizzazione consiste* nell'insieme dei numeri di telefono dei clienti

effettivi e dei potenziali clienti del signor Rossi (e del suo team).
Fanno parte del sistema informativo dell'organizzazione:

- un insieme di numeri, molto piccolo, tale che possa essere *tenuto a mente* dal signor Rossi;
- un altro insieme di numeri *trascritto su un'agenda* clienti. Le dimensioni e la struttura dell'agenda variano a seconda del numero di contatti;
- un *elenco telefonico* della città in cui il team opera per rintracciare i potenziali clienti;
- alcuni *documenti cartacei* informali quali biglietti da visita, ricevute di ristoranti e alberghi, depliant di aziende contattate, fatture, ordini.

Questo semplice sistema informativo utilizza *strumenti di memorizzazione prevalentemente cartacei* e procedure di consultazione e aggiornamento prevalentemente manuali.

Passiamo ora a dimensioni aziendali maggiori, considerando il seguente esempio:

Esempio

Analizziamo brevemente il sistema informativo gestito da una grossa organizzazione, che possiamo individuare in una compagnia di trasporti aerei. Il sistema informativo della compagnia deve essere in grado di fornire ai clienti un efficiente sistema di prenotazione dei posti sui voli, gestire il personale di terra e il personale di volo, gestire la manutenzione degli aerei, la contabilità, il magazzino dei pezzi di ricambio, il rifornimento degli aerei e così via.
Ovviamente, il flusso di informazioni da gestire (e quindi da memorizzare) è enorme e dalla sua gestione dipendono l'efficienza e la qualità del servizio offerto dalla compagnia.

La parte del sistema informativo di un'organizzazione che può essere automatizzata è chiamata *sistema informatico.*

Sistema informatico

Il **sistema informatico** (spesso chiamato anche sistema EDP: *Electronic Data Processing*) è quindi quel sottoinsieme del sistema informativo dedicato alla gestione automatica di informazioni, rappresentate mediante *dati digitali.*

Esempio

Nell'esempio che abbiamo riportato sopra, quello della piccola ditta artigianale, il *sistema informatico* può essere rappresentato da un'*agenda elettronica* nella quale memorizzare gran parte dei numeri di telefono dei clienti.

Esempio

Se consideriamo una *compagnia aerea*, il sistema informatico è costituito dagli *archivi elettronici* in cui sono memorizzati i dati di interesse relativi ai clienti, agli aerei, al personale, dalle componenti fisiche che costituiscono il supporto di memorizzazione, dalle procedure di interrogazione per la ricerca delle informazioni, dalle reti di comunicazioni tra i terminali degli operatori e così via.

La parte di *sistema informativo* che non fa parte del *sistema informatico* è costituita dagli operatori, dagli archivi cartacei, dalla rete di informazioni verbali, dalle tabelle di consultazione, dalle procedure organizzative aziendali. Nel seguito ci occuperemo esclusivamente di sistemi informativi basati sull'uso di sistemi informatici.

3 Dati e informazioni: schemi e istanze

La differenza tra dati e informazioni è stata già introdotta nel primo volume ma è opportuno riprenderla e chiarirla ulteriormente.

Il termine **dato** (dal latino *datum*) significa letteralmente *fatto*. Lo scopo dei dati è quello di *codificare* in vari modi i fatti (anche ipotetici) ritenuti importanti nell'ambito di una organizzazione.

Possiamo dire che un dato costituisce un'informazione per qualcuno se comporta un reale aumento di conoscenza.

Diremo allora che:

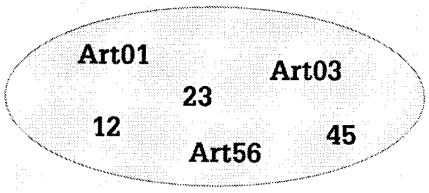
un'**informazione** è l'incremento di conoscenza che può essere acquisita (o inferita) dai dati.

Da questa considerazione consegue che i dati non possono essere ritenuti utili (non danno informazione) fino a quando non si fornisce una **chiave di interpretazione** (o semplicemente *interpretazione*) che consenta di comprendere il loro **significato** (o *semantica*), cioè i fatti che essi codificano. Vediamo un esempio di chiave di interpretazione:

Chiave di interpretazione

Esempio

I dati dell'insieme della figura seguente non forniscono alcuna informazione utile finché non si definisce cosa ciascun dato *rappresenta* e le eventuali *relazioni* esistenti tra i dati stessi.



✦ Figura A1.2
Un insieme di dati

Il dato 23, senza chiave di interpretazione che gli dia un significato, non costituisce informazione per nessuno. Lo stesso dato 23, invece, con la chiave di interpretazione "Articoli in magazzino" assume il significato di "quantità disponibile per un determinato articolo". Possiamo dare un significato al precedente insieme con la seguente forma tabellare:

Articoli in magazzino	
Codice articolo	Quantità

✦ Figura A1.3
Lo schema o intensione

Chiameremo **schema** o **intensione** la chiave di interpretazione dei dati, ovvero il *significato* (la semantica) che si attribuisce al dato per ricavare l'informazione da esso completata.
Chiameremo invece **istanza di uno schema** o **estensione** i valori assunti da uno schema in un *certo istante di tempo*.

Un'istanza dello schema precedente è quella composta dai seguenti tre articoli e dalle loro rispettive quantità:

Art01	**23**
Art03	**12**
Art56	**45**

✦ Figura A1.4
L'istanza di uno schema o estensione

Quindi:

parleremo di **significato intensionale** (o *schema*) riferendoci al contenuto informativo dei dati: la ricostruzione dei fatti (o *significato completo dei dati*) nasce dunque dall'*interpretazione dei dati.*
Parleremo di **significato estensionale** (o *istanze*) dei dati riferendoci ai valori che può assumere uno schema in un *certo istante.*

Lo schema *non varia nel tempo.*
L'istanza di uno schema *varia nel tempo* poiché possono essere eseguite modifiche, inserimenti, cancellazioni.
Assumendo che lo schema non può essere modificato, consideriamo la sua modifica come la definizione di un nuovo schema, ovvero la definizione di una nuova base di dati.

Variazioni nel tempo

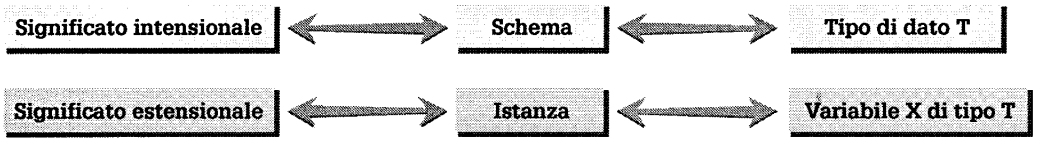
Riprendendo l'esempio precedente degli "Articoli in magazzino", allora abbiamo:

Esempio

schema (o contesto semantico):		Codice articolo	Quantità
istanza:	primo articolo	Art01	23
	secondo articolo	Art03	12
	terzo articolo	Art56	45

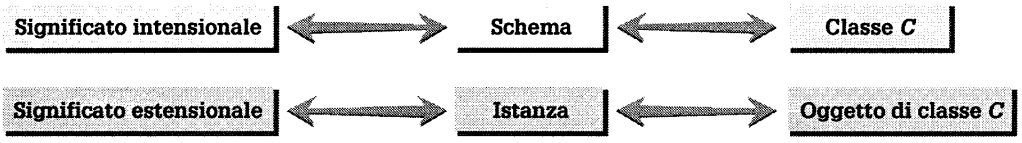
il significato intensionale del dato 23 è: "quantità in magazzino dell'articolo Art01"

Volendo fare un'analogia con un concetto già noto dei linguaggi di programmazione, possiamo dire che lo *schema* può essere equiparato alla *dichiarazione* di un tipo di dato, mentre un'*istanza* è un possibile *valore* di quel tipo.



✦ Figura A1.5 Analogia tra tipo di dato e schema e tra istanza e variabile

Oppure, nel paradigma a oggetti:



✦ Figura A1.6 Analogia tra schema e classe e tra istanza e oggetto

La distinzione tra intensione ed estensione non è limitata solo alla teoria delle basi di dati, ma pervade tutta l'informatica.

Linguaggi di programmazione "sporchi"

Nei *linguaggi di programmazione*, ad esempio, il significato intensionale di un dato è il tipo di dato. Sono considerati "sporchi" quei linguaggi in cui è possibile cambiare dinamicamente tipo a un dato (cioè attribuire interpretazioni diverse agli stessi dati).

Esempio

Consideriamo in un linguaggio di programmazione il dato intero 23. *Intero* è lo *schema*, *23* è l'*istanza dello schema "Intero"*. Se ora tale intero 23 può diventare (a seguito di una conversione di tipo) un reale, allora il dato 23, che prima aveva il significato (intensionale) di "numero intero", adesso ha il significato (intensionale) di "numero in virgola mobile". Ha quindi cambiato schema pur rimanendo con lo stesso significato estensionale.

Associazione tra insiemi di dati

Dal precedente esempio del magazzino notiamo infine che esiste una *associazione* implicita tra i due insiemi di dati: i *nomi degli articoli* e la loro *quantità in magazzino.*

Da un punto di vista pratico, quando si è in presenza di molti gruppi di dati con la stessa chiave di interpretazione, non è conveniente registrare l'interpretazione per ciascuno di essi. Nell'esempio precedente, dovendo elencare le quantità relative a diversi articoli, è stato spontaneo organizzare i dati in una tabella di coppie del tipo (X, Y) il cui significato intensionale è "l'articolo di codice X è presente in quantità Y in magazzino".

Tale interpretazione è comune a tutte le coppie e può quindi essere fornita una sola volta.

Separazione tra dati e interpretazione

La separazione tra i dati e la loro interpretazione, anche se non è tipica dei linguaggi naturali, si ritrova in molti ambiti della vita di tutti i giorni. Si pensi agli orari ferroviari, alle fatture commerciali e a tutte quelle rappresentazioni tabellari in cui viene fornita a parte la descrizione sul modo di interpretare i dati.

Chiameremo **categoria** una raccolta di gruppi di dati aventi la stessa chiave di interpretazione ovvero lo stesso schema.

Nell'esempio precedente avevamo la categoria "Articoli in magazzino". Di essa fa parte la seguente istanza ovvero particolari valori assunti dalla categoria.

✦ Figura A1.7
Categoria, schema e istanza

In una **base di dati** possiamo identificare:

- un insieme di **categorie;**
- un insieme di **regole**, che le categorie devono soddisfare;
- un insieme di **autorizzazioni**, che definiscono le operazioni consentite a determinati utenti.

Una **occorrenza** di una base di dati è l'insieme delle istanze delle categorie in un determinato istante di tempo.

Spesso per *base di dati* intenderemo una particolare occorrenza di una base di dati.

4 Metodi, linguaggi e sistemi per basi di dati

Realtà di interesse

Date le esigenze dell'utente relative a una determinata situazione, che chiameremo **realtà di interesse**, il nostro primo obiettivo sarà quello di effettuare un'analisi su tale realtà. A questo punto potremo progettare e realizzare una base di dati che possa soddisfare le esigenze dell'utente.

Diversi approcci alla realtà di interesse

Per raggiungere questi obiettivi possiamo seguire strade diverse in base al tipo di approccio che vogliamo adottare. Gli *approcci possibili* che considereremo sono:

- approccio puro *a oggetti*;
- approccio puro *relazionale*;
- approccio *misto*.

Nell'approccio puro a oggetti si segue un'analisi e una progettazione delle basi di dati orientata agli oggetti.

Nell'approccio puro relazionale si segue un'analisi e una progettazione "classica", utilizzando il modello logico relazionale.

L'approccio misto, invece, prevede un'analisi a oggetti, alla quale segue però un utilizzo di strumenti e modelli propri dell'approccio "classico". Le modalità operative di ciascun approccio saranno oggetto di studio delle successive unità didattiche di questo modulo.

Indipendentemente dagli approcci appena descritti, parlando di basi di dati si individuano i seguenti macroargomenti:

- **metodi di progettazione** per basi di dati;
- **linguaggi** per basi di dati;
- **sistemi di gestione** delle basi di dati.

Al momento ci basta sapere che:

- i **metodi** o **metodologie di progettazione** hanno l'obiettivo di definire lo schema di una base di dati, a partire dalle specifiche dell'utente;
- i **linguaggi** sono quelli disponibili per creare le basi di dati, per effettuare inter-

rogazioni e per modificare sia gli schemi sia le istanze di una base di dati;
- i **sistemi di gestione delle basi di dati** o **DBMS** (*Data Base Management System*), sono prodotti software che permettono di interagire con una base di dati, consentendo opportune operazioni agli utenti autorizzati, nel rispetto delle regole prestabilite. Le richieste degli utenti non devono violare alcun vincolo sui dati.

In seguito vedremo in dettaglio ognuno di questi argomenti, chiarendo gli aspetti più importanti. Occorre precisare che esistono altri approcci possibili per analizzare una realtà di interesse. Ricordiamo in particolare, per l'approccio *non a oggetti*, il **modello gerarchico** (che assunse grande rilevanza agli inizi degli anni Settanta) e il **modello reticolare**, utilizzato fino alla fine degli anni Ottanta, entrambi completamente rimpiazzati dal **modello relazionale**.

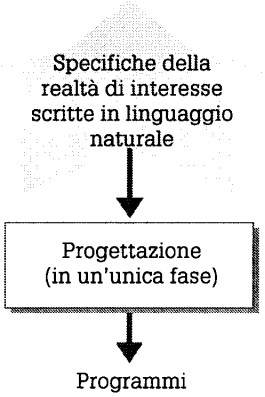
5 La progettazione di una base di dati

Metodologia di progettazione

La **metodologia di progettazione di una base di dati** è un insieme di:

- *attività* tra loro collegate;
- *prodotti* intermedi e finali di tali attività;
- *criteri di verifica* di qualità di tali fasi e prodotti, volti a realizzare una base di dati a partire da un insieme di **specifiche** che formalizzano le esigenze dell'utente.

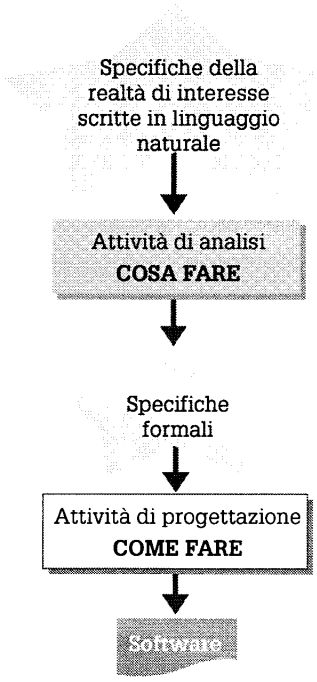
✦ Figura A1.8
La trasformazione diretta delle specifiche in programmi



In un primo momento della storia delle basi di dati avveniva quanto descritto nella *Fig. A1.8*, ovvero una *trasformazione delle specifiche* (scritte in linguaggio naturale) in programmi, attraverso una sola fase di progettazione.

Problemi legati alla scarsa documentazione della struttura del programma e alla mancanza di una visione d'insieme creavano difficoltà di modifica e mantenimento. Tali difficoltà sono state la spinta verso quanto descritto nella *Fig. A1.9*.

✦ Figura A1.9
Un primo approccio più metodologico



Come si può osservare, le *specifiche della realtà di interesse* vengono trasformate in **specifiche formali** dopo una prima fase di analisi. In questo caso "formali" si associa ad **astratte**, ovvero si descrivono in modo formale e astratto le tipologie di dati e le funzioni che opereranno su di essi.

Astratto, in questo caso, vuol dire indipendente dalla tecnologia e dal particolare linguaggio di programmazione che verrà utilizzato per generare i programmi.

Si dice anche che la fase di analisi è volta a determinare **cosa** il programma deve fare (indipendentemente dai linguaggi che verranno utilizzati) e che la fase di progetto è volta a determinare **come** il programma dovrà fare quanto stabilito, cioè in che modo il documento di specifiche formali viene trasformato in programma.

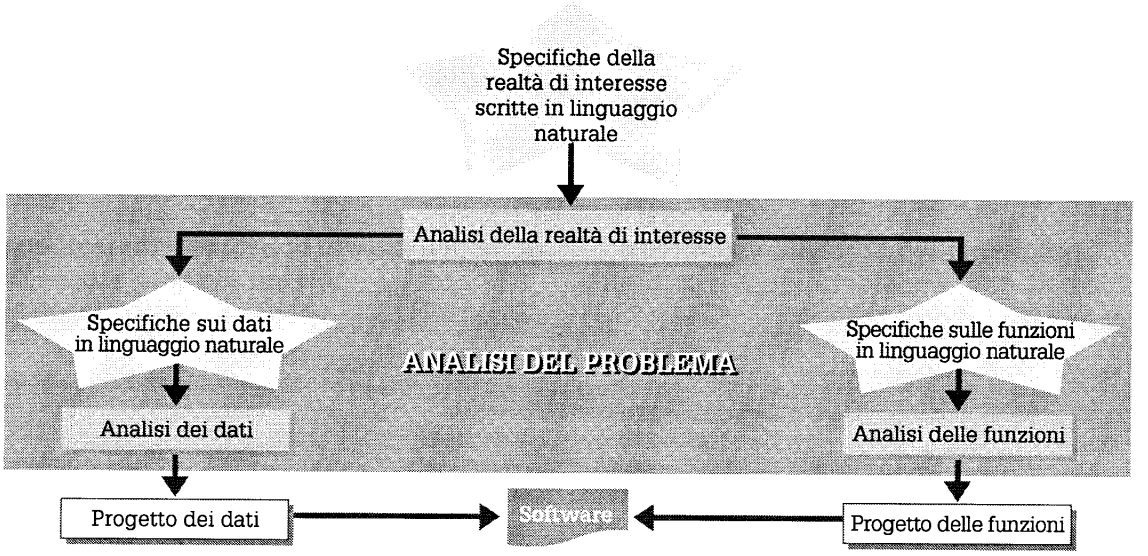
Questo approccio è ereditato da altre discipline ingegneristiche (basti pensare, ad esempio, alla costruzione di un edificio). Nella progettazione di un edificio infatti l'ingegnere progettista riassume sul progetto cartaceo le richieste del cliente. Utilizza poi i prospetti e le piante dei vari piani dell'edificio per dialogare con il cliente sull'aspetto che l'edificio dovrà assumere.

Da tale progetto si evincono inoltre i calcoli statici per il dimensionamento delle

strutture portanti. Tali calcoli specificano alla ditta costruttrice come dovrà procedere per la costruzione dell'edificio.

Nell'*analisi* e nella *progettazione* occorre considerare sia i dati sia le funzioni che agiscono sui dati.

La *Fig. A1.9* viene quindi modificata nel seguente modo:

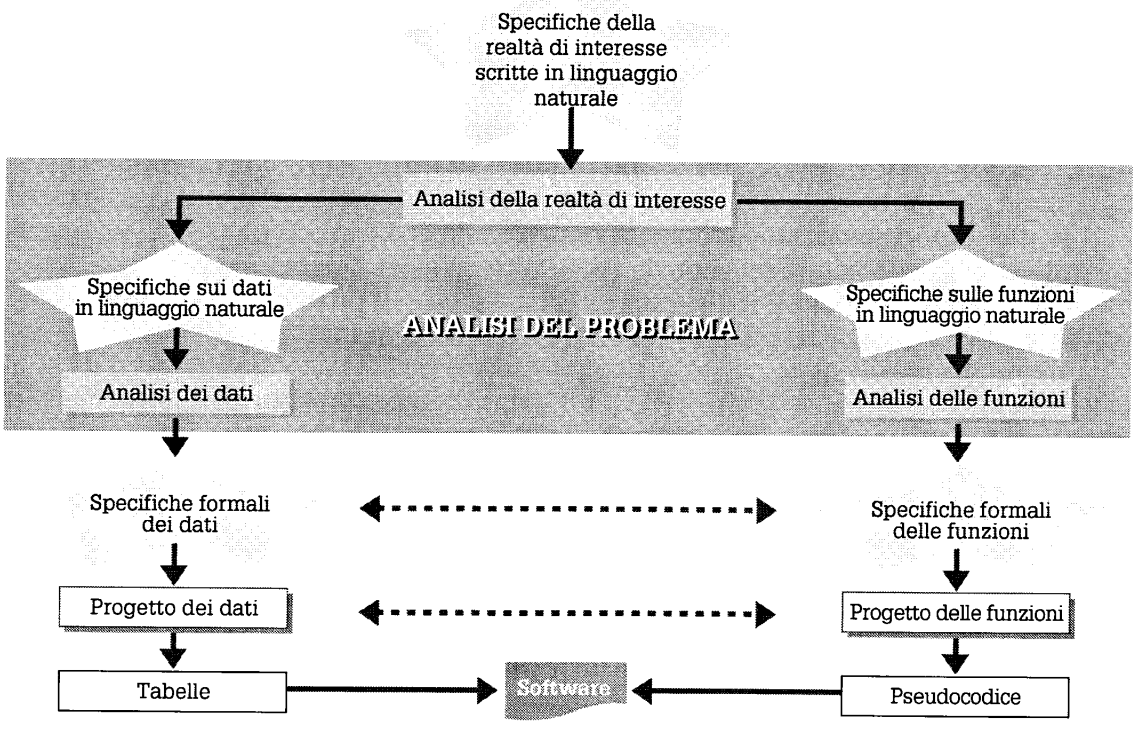


✦ Figura A1.10
Si considerano le funzioni oltre che i dati

La distinzione tra analisi e progettazione si era affermata più per le *funzioni* che per i *dati*. Quando si prendeva in considerazione un sistema di *gestione del personale*, si era più interessati a definire l'organizzazione delle funzioni che quella dei dati. Il calcolo dello stipendio, ad esempio, veniva definito specificando le funzioni:

stipendio netto = stipendio base + indennità speciali – ritenute

Solo successivamente si è cominciato a definire il formato dei dati su cui queste funzioni dovevano operare. Si assisteva quindi a un approccio asimmetrico tra dati e funzioni. Solo in seguito è emerso con chiarezza che la **risorsa centrale sono i dati**.



✦ Figura A1.11
Un approccio completo con funzioni e dati

Approccio a pari dignità

In un approccio di questo genere, detto **a pari dignità** proprio perché vengono attivate due fasi distinte, vi è una verifica di *completezza reciproca* (rappresentata dalla doppia freccia tratteggiata riportata nella figura precedente).

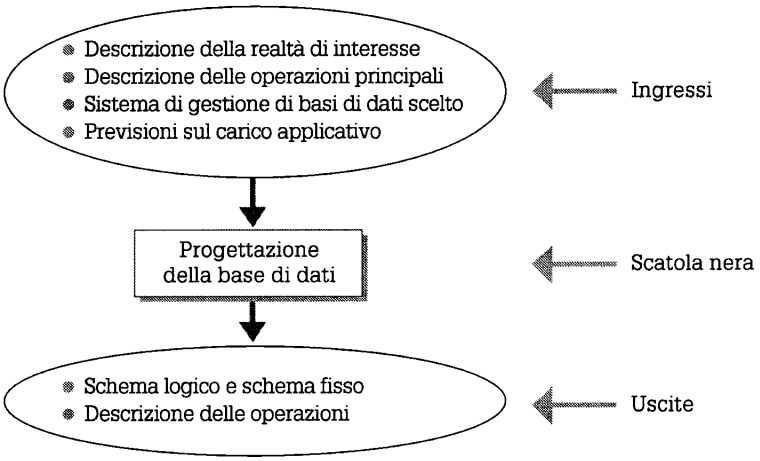
Alla fine della fase di analisi delle funzioni sarà cioè indispensabile verificare che:

- per ogni funzione definita nello schema delle funzioni siano rappresentate tutte le tipologie di dati da essa manipolate;
- per ogni aspetto riguardante i dati, all'interno delle specifiche formali dei dati, siano definite tutte le funzioni che su quell'aspetto operano.

5.1 La modellazione dei dati: progettazione concettuale, logica e fisica

Vediamo più in dettaglio gli **ingressi** e le **uscite** dell'attività di progettazione di una base di dati, considerando per il momento tale attività come un **sistema a scatola nera**.

✦ Figura A1.12
Gli ingressi e le uscite dell'attività di progettazione



L'insieme di attività che si devono svolgere in questa fase consta di tre distinte attività di progettazione.

Progettazione concettuale

1) **Progettazione concettuale**. Il suo scopo è *costruire* e *definire* una rappresentazione corretta e completa della realtà di interesse.
L'*input* di questa fase è il documento delle specifiche formali.
L'*output* è uno schema concettuale, cioè una rappresentazione astratta e il più possibile formale della realtà: un esempio di output, riferendoci solo ai dati (e non alle funzioni), è il diagramma ER che vedremo nella prossima unità.

Progettazione logica

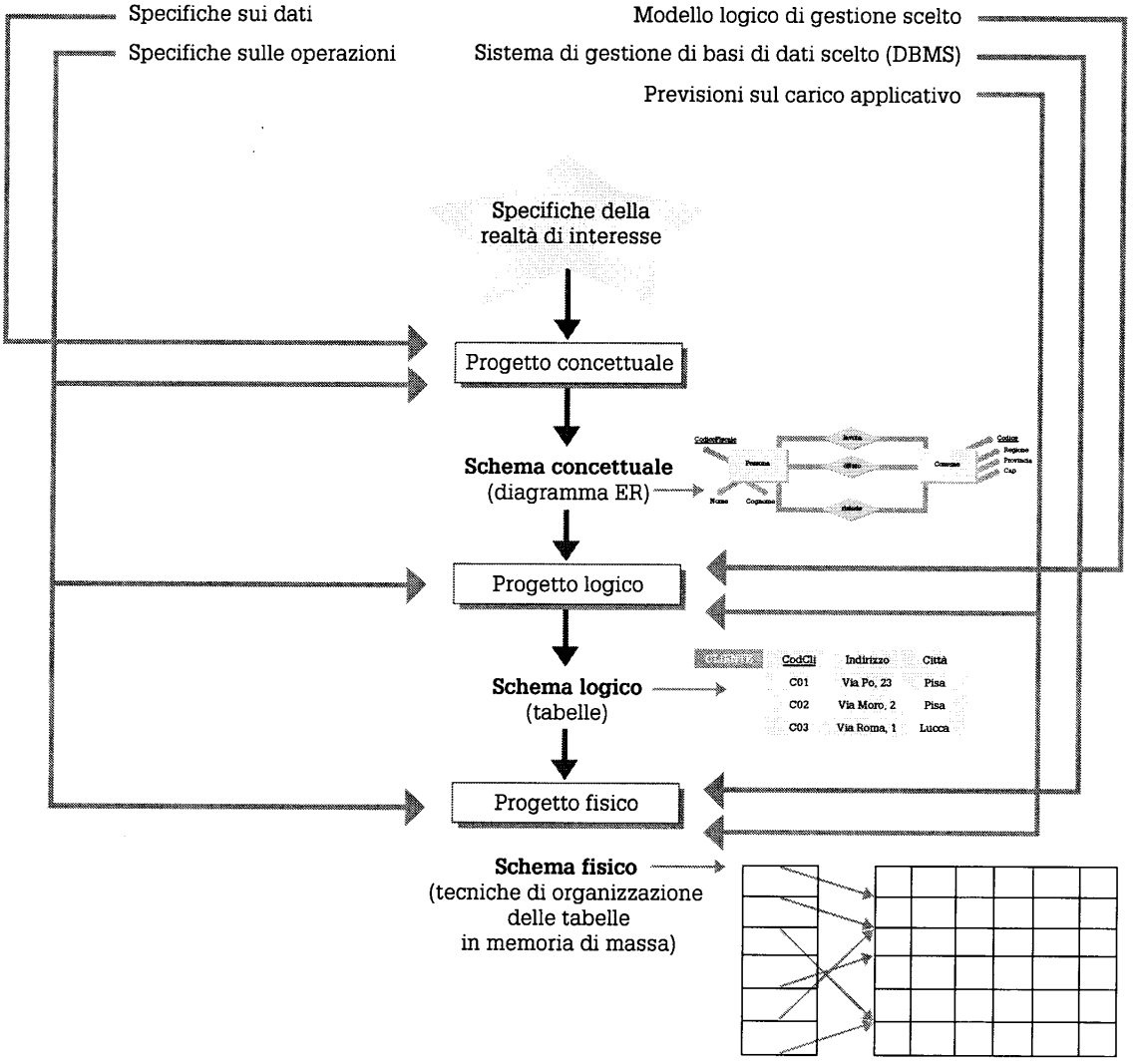
2) **Progettazione logica**. Il suo scopo è quello di *trasformare* questa rappresentazione ancora astratta e indipendente da un DBMS, in uno *schema logico* ovvero in una rappresentazione efficiente rispetto alle strutture di un DBMS: un esempio è una descrizione relazionale tramite tabelle del diagramma ER.
L'*input* di questa fase è il diagramma ER della fase di progettazione concettuale.
L'*output* di questa fase è uno schema logico riassumibile con relazioni rappresentate tramite tabelle.

Progettazione fisica

3) **Progettazione fisica**. Il suo scopo è quello di *implementare* lo schema logico definendo tutti gli aspetti fisici di memorizzazione e rappresentazione in memoria di massa.
L'*input* di questa fase sono le tabelle della fase di progettazione logica.
L'*output* di questa fase è l'implementazione in memoria di massa di tali tabelle.

Spesso si parla di *modello logico-fisico* intendendo questa come un'unica fase realizzativa.

Vediamo ora in quali delle tre fasi della progettazione intervengono gli ingressi visti nella rappresentazione *a scatola nera* della fase di progettazione di *Fig. A1.12*.



✦ Figura A1.13
Gli ingressi e le uscite nelle varie fasi di progettazione

Come possiamo vedere:

- le *specifiche sui dati* vanno come *input* nella prima fase, quella di *progettazione concettuale*;
- le *specifiche sulle operazioni* vanno in *input*, inizialmente nella fase di *progettazione concettuale* (per garantire quella che abbiamo chiamato la completezza reciproca delle specifiche delle operazioni rispetto alle specifiche sui dati) e, successivamente, nelle fasi di *progettazione logica* e *fisica*;
- per scegliere uno strumento efficiente dobbiamo sapere qual è il *carico applicativo* ovvero quali sono le operazioni sui dati e la frequenza con cui queste operazioni vengono eseguite;
- il *modello logico* svolge un ruolo solo nella fase di progettazione logica.

Il DBMS

Il **DBMS**, cioè il prodotto software che *implementa il modello logico*, verrà utilizzato nella progettazione fisica, fase in cui vengono appunto determinati tutti i parametri fisici di utilizzazione della base di dati.

6 I DBMS e il passaggio dagli archivi tradizionali ai database

Il **DBMS** è il software che offre la possibilità di costruire e gestire una base di dati su una memoria di massa, partendo da un progetto concettuale tradotto, poi, in un *modello logico*.

Rappresenta, quindi, un'interfaccia tra gli utenti della base di dati (con le applicazioni e gli archivi correlati) all'interno di un sistema di elaborazione.

Prima di addentrarci nell'analisi delle caratteristiche di un DBMS dobbiamo porci un quesito: perché utilizzare i database piuttosto che gli archivi? La riposta è semplice: le tecniche di gestione delle basi di dati nascono per superare i limiti posti dalle tradizionali tecniche di organizzazione degli archivi informatici.

Esempio

Supponiamo di voler registrare le informazioni relative agli ordini di accessori per auto fatti da alcuni clienti di un'impresa. In base alle più tradizionali (e poco efficienti) tecniche di gestione degli archivi, decidiamo di realizzare un *Archivio Clienti* come quello rappresentato di seguito:

✦ Figura A1.14
Archivio clienti

CLIENTI

CodCli	Cognome	Nome	Indirizzo	Città
010	Rossi	Paolo	Via Roma, 35	Milano
020	Verdi	Gianni	Corso Italia, 1/A	Roma
030	Neri	Piero	Viale Garibaldi, 3	Firenze

e un archivio che riporta le quantità ordinate dai clienti:

✦ Figura A1.15
Archivio ordini

ORDINI

CodCli	CodAcc	Descrizione	Prezzo	Quantità
010	M03	Batteria	100,00	3
010	M12	Radiatore	1200,00	1
010	M04	Antenna	25,00	3
020	M03	Batteria	100,00	2
020	M12	Radiatore	1200,00	1
030	M03	Batteria	100,00	2

Con tale tipo di organizzazione (decisamente poco efficiente) possono nascere problemi di varia natura tra loro collegati:

Ridondanza, incongruenza e inconsistenza dei dati

- *esistono dati ripetuti*: la descrizione dell'articolo ordinato viene ripetuta per ogni movimento (problema della **ridondanza** dei dati);
- a causa della ridondanza nascono problemi di **incongruenza** dei dati: se un dato di un articolo viene modificato, tale modifica dovrà essere apportata a tutti i record che movimentano quell'articolo (si pensi, ad esempio, al caso in cui il prezzo della batteria dovesse cambiare);
- a causa dell'incongruenza nascono problemi di **inconsistenza**: i dati diventano inaffidabili poiché si potrebbe anche non risalire al dato corretto (ritornando all'esempio della batteria, potremmo anche non risalire all'attuale prezzo di vendita).

Questi problemi nascono perché *i dati contenuti negli archivi non sono organizzati in modo integrato tra loro.* I problemi non terminano qui:

- nell'*organizzazione tradizionale degli archivi* i tradizionali linguaggi di programmazione richiedono che all'interno del programma vengano specificati gli archivi utilizzati e la struttura dei loro record. Ciò comporta notevoli disagi: la modifica di un record comporta la modifica di tutti i programmi che utilizzano quel record;
- l'*operazione di accesso ai dati* è strettamente correlata all'organizzazione assegnata agli archivi. Tale organizzazione vincola il programmatore, nel senso che potrà implementare le sole operazioni consentite per il tipo di organizzazione scelta.

La teoria delle basi di dati supera questi incresciosi problemi poiché un DBMS permette:

- indipendenza dalla *struttura fisica* dei dati;
- indipendenza dalla *struttura logica* dei dati.

L'**indipendenza fisica** dei dati consiste nella possibilità di modificare lo **schema fisico** (la struttura fisica) dei dati senza dover modificare i programmi applicativi che usano i dati, cioè l'*organizzazione logico-concettuale*.

Indipendenza fisica e logica dei dati

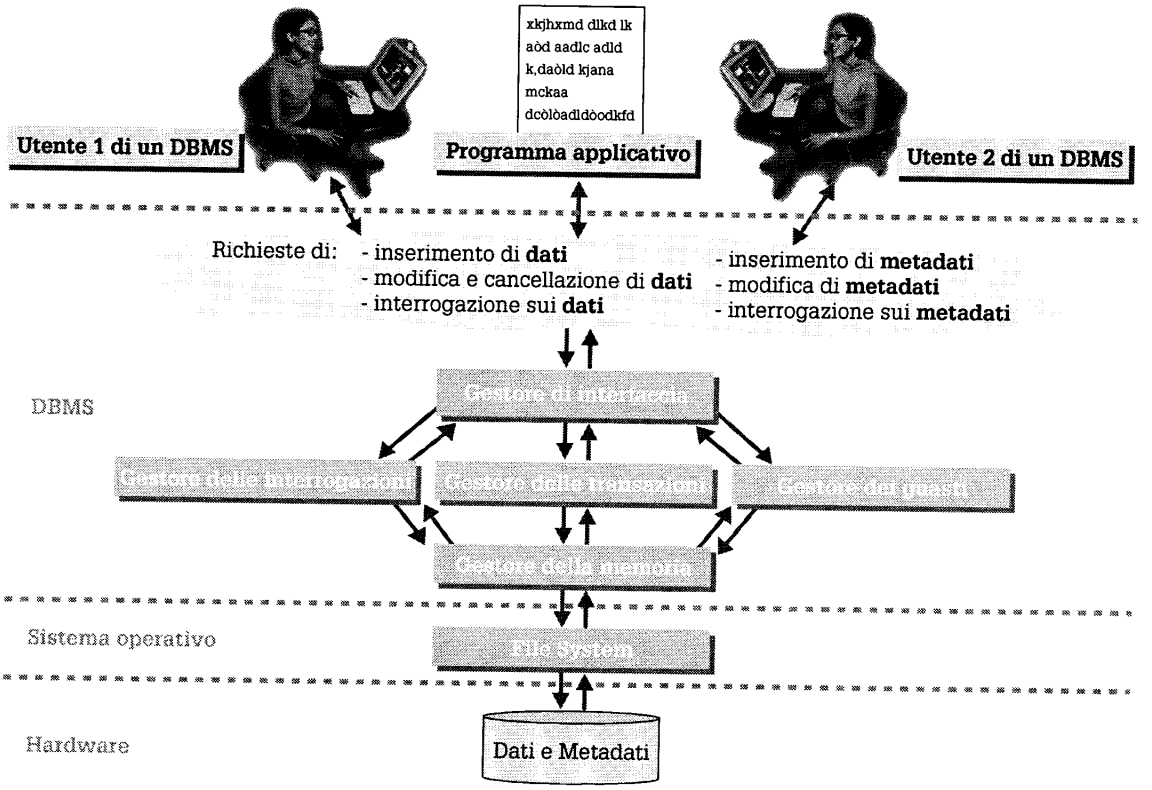
Esempio

Se abbiamo memorizzato la tabella relativa ai clienti su un file a indice sequenziale ordinato solo alfabeticamente, non dobbiamo essere costretti a interrogare i clienti solo in ordine alfabetico.

L'**indipendenza logica** dei dati consiste nella possibilità di modificare lo schema concettuale senza dover modificare il software applicativo.

7 Architettura di un DBMS

I più importanti DBMS presenti sul mercato (come *Microsoft Access*, *Oracle ecc.*), sono caratterizzati da un'architettura interna nella quale, anche se a grandi linee, è possibile riconoscere componenti comuni, che riportiamo nella figura seguente.



✦ Figura A1.16
L'architettura di un DBMS

Analizziamo dettagliatamente questi componenti.

Componenti comuni dei DBMS

- **Gestore di interfaccia.**
  È il componente di più alto livello, nel senso che interagisce direttamente con l'utente della base di dati, fornendo un'opportuna interfaccia grafica per formulare le richieste di operazione sui dati o sui metadati. Interagisce con il *gestore delle interrogazioni*.
- **Gestore delle interrogazioni.**
  Il suo compito è quello di eseguire una richiesta di interrogazione (o meglio di una qualsiasi operazione sui dati) proveniente dal gestore di interfaccia, trovando il modo ottimale di portare a termine l'operazione. Interagisce con il *gestore della memoria*.
- **Gestore delle transazioni.**
  Gestisce le modalità di accesso ai dati, compresa la concorrenza sui dati (ovvero

Log file

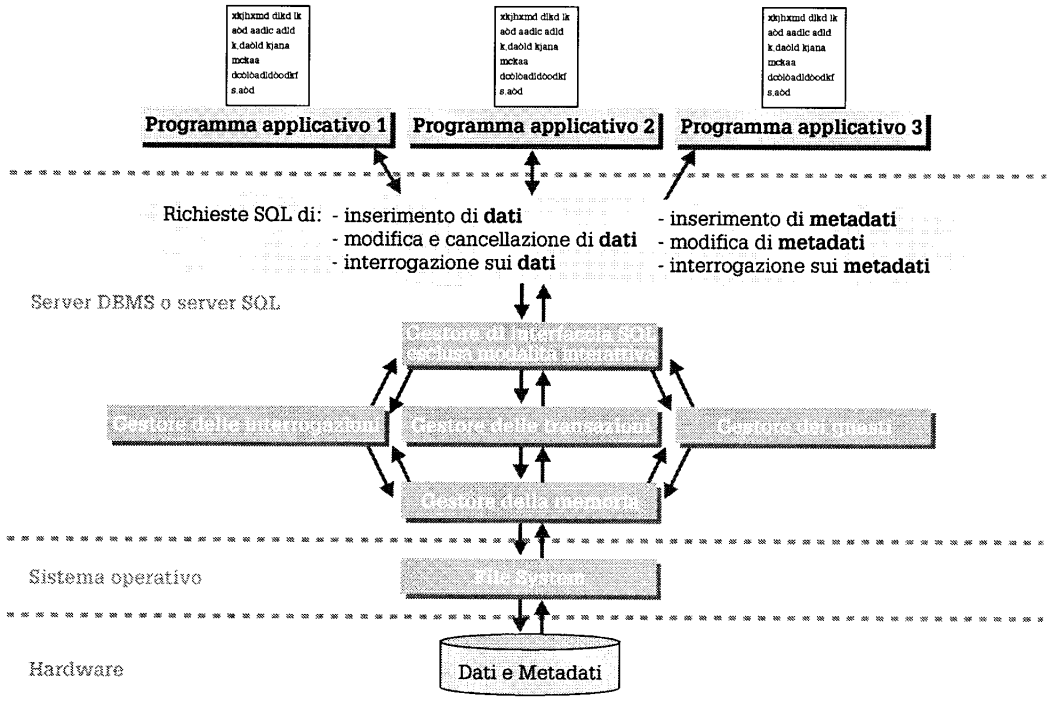
le richieste simultanee da parte di più utenti sugli stessi dati). Per far questo interagisce con il *gestore delle interrogazioni*, oltre che con il *gestore di interfaccia*. È responsabile della gestione dei **log file**, ovvero dei file nei quali vengono registrate tutte le operazioni effettuate sulla base di dati. Per questo motivo interagisce anche con il *gestore della memoria*.

- **Gestore della memoria**.
  È il componente che si trova a stretto contatto con il *file system* e i dispositivi fisici di memorizzazione. È interfacciato direttamente con tali dispositivi per accedere ai dati in essi memorizzati o per registrare modifiche sui dati (se richiesto dalle componenti dei livelli superiori).
- **Gestore dei guasti**.
  Si occupa di effettuare *copie periodiche* della base di dati, per poter ripristinare una situazione di emergenza, determinata da un evento disastroso.

8 Architettura client-server di un DBMS

L'**architettura client-server**, su cui si basa la maggior parte del software di ultima generazione, è anche alla base di tutti i principali prodotti DBMS presenti sul mercato. L'intero DBMS è in pratica un server, in quanto riceve richieste di servizi (richieste di operazioni sui dati) e risponde eseguendo tali operazioni, restituendo tabelle come risultato. Nell'architettura interna di un DBMS occorre però escludere la parte del *gestore di interfaccia* che riceve interrogazioni. Considereremo il seguente schema architetturale di un server DBMS (spesso anche chiamato server SQL), poiché supporremo che l'interazione tra server DBMS e applicazioni avvenga quasi esclusivamente tramite la parte di supporto per SQL della componente *gestore di interfaccia*.

✦ Figura A1.17
L'architettura di un server DBMS



9 I linguaggi

L'utente attiva le prestazioni di un DBMS utilizzando appropriati comandi, grazie ai quali si instaura una comunicazione con il sistema di elaborazione che gestisce il database. L'insieme di tali comandi costituisce un vero e proprio linguaggio, anzi costituisce un insieme di vari linguaggi. In particolare:

- il **DDL** (*Data Definition Language – Linguaggio di definizione dei dati*) è il linguaggio utilizzato per la descrizione dei dati, delle tabelle, delle interfacce utente e delle viste (sottoschemi). Rappresenta il mezzo con il quale l'utente costruisce la struttura fisica del database, partendo dallo schema logico;
- Il **DMCL** (*Device Media Control Language – Linguaggio di controllo dei supporti di memorizzazione dei dati*) è lo strumento che permette alla struttura fisica del database di far riferimento alle particolari unità di memoria di massa utilizzate dal sistema;
- Il **DML** (*Data Manipulation Language – Linguaggio per il trattamento dei dati*) è lo strumento attraverso il quale è possibile accedere al database per effettuare inserimenti,modifiche e cancellazioni;
- Il **DCL** (*Data Control Language – Linguaggio di controllo dei dati*) è il linguaggio utilizzato per stabilire i vincoli di integrità, oltre che per stabilire accessi e permessi;
- Il **QL** (*Query Language - Linguaggio di interrogazione*) è lo strumento interattivo che permette di interrogare il database al fine di ritrovare i dati relativi alla chiave di ricerca impostata dall'utente.

DDL, DMCL, DML, DCL e QL

Attualmente, la diffusione del modello relazionale (che esamineremo nelle prossime unità) ha favorito la nascita del **linguaggio per basi di dati**, ossia di un insieme di comandi che consentono la gestione globale di un database. Questo linguaggio, pertanto, raggruppa in sé le funzioni dei linguaggi DDL, DMCL, DML, DCL e QL.

10 Gli utenti di una base di dati

Individuiamo le seguenti **classi di utenza** di un database:

Classi di utenza di un database

- **utenti semplici**. Sono coloro che utilizzano il database per inserire, modificare e interrogare i dati in modo "protetto", seguendo i programmi applicativi creati dagli utenti programmatori. Spesso tali utenti sono anche chiamati *operatori*;
- **utenti avanzati**. Sono coloro che accedono alla base di dati per effettuare interrogazioni anche complesse, per le quali non è richiesto l'intervento dei programmatori per la scrittura di programmi *ad hoc*. Tali interrogazioni vengono effettuate con la conoscenza di un linguaggio di interrogazioni non procedurale come SQL. Tipicamente sono persone che fanno capo allo staff dirigenziale e di livello medio di un'organizzazione;
- **utenti programmatori**. Sono coloro che costruiscono specifiche applicazioni per permettere agli altri utenti un'interazione "controllata" sulla base di dati. Utilizzano il DML del DBMS. La distinzione tra utente e programmatore si sta sempre più attenuando a causa della interattività e facilità d'uso dei DBMS, con i quali è ormai possibile anche per un utente non esperto creare tabelle, manipolarle ed effettuare interrogazioni;
- **amministratore o DBA** (*DataBase Administrator*). È colui che progetta e si occupa della manutenzione della base di dati. Gestisce inoltre:
  - la definizione iniziale e la modifica dello schema della base di dati. È responsabile della *correzione*, dell'*ampliamento* e dell'*adeguamento* dello schema logico alle nuove esigenze degli utenti della base di dati;
  - la definizione iniziale e la modifica delle viste logiche sui dati. È responsabile della *protezione* e della *riservatezza* dei dati;
  - la definizione iniziale e la modifica delle politiche di accesso ai dati. È responsabile dell'*assegnazione dei privilegi di accesso* agli utenti della base di dati.
  - la definizione iniziale e la modifica delle politiche di salvataggio e ripristino dei dati. È responsabile dell'immediato *ripristino dei dati* a seguito di eventi disastrosi.

Conoscenze

VERO o FALSO?

	V	F
1. Un sistema informativo è un insieme organizzato di procedure automatiche.	☐	☐
2. EDP è sinonimo di sistema informativo.	☐	☐
3. Istanza e significato intenzionale sono sinonimi.	☐	☐
4. Un DBMS è un linguaggio di interrogazione per basi di dati.	☐	☐
5. Le specifiche formali sono il risultato dell'attività di progettazione.	☐	☐

QUESITI

1. Che differenza c'è tra sistema informativo e sistema informatico?
2. Cosa indichiamo con "realtà di interesse"?
3. Che differenza c'è tra dato e informazione?
4. Che differenza c'è tra schema e istanza?
5. Che cosa sono i DBMS?
6. Che cosa è una base di dati?
7. Quali sono i principali passi della fase di progettazione di una base di dati?
8. Che cosa contengono le specifiche della realtà di interesse?
9. In cosa consistono l'analisi dei dati e l'analisi delle funzioni?
10. Quali sono gli input della fase di progettazione concettuale?
11. Quali sono gli input della fase di progettazione logica?
12. Quali sono gli input della fase di progettazione fisica?
13. Quali sono gli output della fase di progettazione concettuale?
14. Quali sono gli output della fase di progettazione logica?
15. Quali sono gli output della fase di progettazione fisica?
16. Che cosa indica la sigla DDL?
17. Che cosa indica la sigla DML?
18. Perché si parla di specifiche formali?
19. Che differenza c'è tra utente programmatore e amministratore di una base di dati?
20. Perché si parla di indipendenza fisica e di indipendenza logica dei dati?

QUESITI A SCELTA MULTIPLA E A RISPOSTA SINGOLA

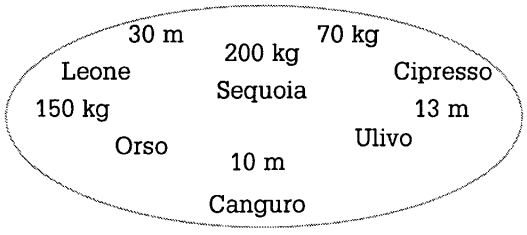
1. Qual è l'output della fase di progettazione concettuale?
   a. Il modello relazionale;
   b. il diagramma ER;
   c. il documento delle specifiche;
   d. le entità e associazioni.
2. Qual è l'output della fase di progettazione logica?
   a. Il modello ER;
   b. lo schema relazionale;
   c. il documento delle specifiche;
   d. le richieste degli utenti.
3. Il QL è:
   a. un linguaggio di definizione dei dati;
   b. un linguaggio di interrogazione;
   c. un linguaggio per il trattamento dei dati;
   d. nessuno dei precedenti.
4. L'amministratore di una base di dati è colui che:
   a. è responsabile della protezione e della riservatezza dei dati;
   b. è responsabile dell'implementazione tramite linguaggi di programmazione della base di dati;
   c. è responsabile della vendita della base di dati;
   d. nessuno dei precedenti.

Competenze

ESERCIZI

1. Quali dei seguenti oggetti possono essere considerati istanze e quali schemi?
   - il tipo Reale:
   - il valore 4,56:
   - Paolo Rossi:
   - una persona:
   - un indirizzo qualsiasi:
   - via Po, 45:

2. Date le seguenti istanze:



individuare:
- le categorie;
- per ogni categoria il relativo schema.