Basi di Dati

Federico Matteoni

A.A. 2019/20

Indice

Introduzione

Obiettivi del corso Modelli dei dati, linguaggi e sistemi per lo sviluppo di applicazioni che prevedono l'uso di grandi quantità di dati permanenti organizzati in **basi di dati**.

Testo di Riferimento *Fondamenti di Basi di Dati*, A. Albano, G. Ghelli e R. Orsini, Zanichelli. Scaricabile liberamente da `fondamentidibasididati.it`

Terminologia

Base di dati: tecnologia di base, gestione delle attività quotidiane dell'organizzazione e **tema di questo corso**

Data Warehouse, Data Lake, Big Data, Data Science: termini che hanno a che vedere con l'**analisi dei dati** e che non rientrano nei temi trattati nel corso.

Capitolo 1

Costruzione di una base di dati

Cos'è una base di dati? Una **base di dati** è un **insieme organizzato di dati** usati per il supporto allo svolgimento di un'attività (di un ente, azienda, ufficio, persona...)

Qualche esempio

Materie

Titolo	Codice	Syllabus
Basi di Dati	AA024	Progettazione e interrogazione...
Reti di Calcolatori	AA019	Realizzazione e uso di reti, protocollo TCP...

Corsi

Materia	AA	Semestre	Titolare
AA024	2007	1	Albano
AA024	2007	1	Ghelli
AA019	2007	1	Brogi

1.1 Elementi

1.1.1 Figure Coinvolte

Committente

Dirigente

Operatore

Fornitore

Direttore del progetto

Analista

Progettista del DB

Programmatore di applicazioni che usano il DB

Manutenzione e messa a punto del DB – **Gestione del DBMS**

Amministratore del DBMS

1.1.2 Sistemi Informativi

Definizione Un **sistema informativo** di un'organizzazione è una **combinazione di risorse, umane e materiali, e di procedure** organizzate per raccolta, archiviazione, elaborazione e scambio **delle informazioni** necessarie alle attività:

Operative (informazioni di servizio)

Programmazione e **controllo** (informazioni di gestione)

Pianificazione strategica (informazioni di governo)

Esempi di sistemi informativi Un comune

Gestione servizi demografici (anagrafe, stato civile, servizio elettorale e vaccinale) e della rete viaria

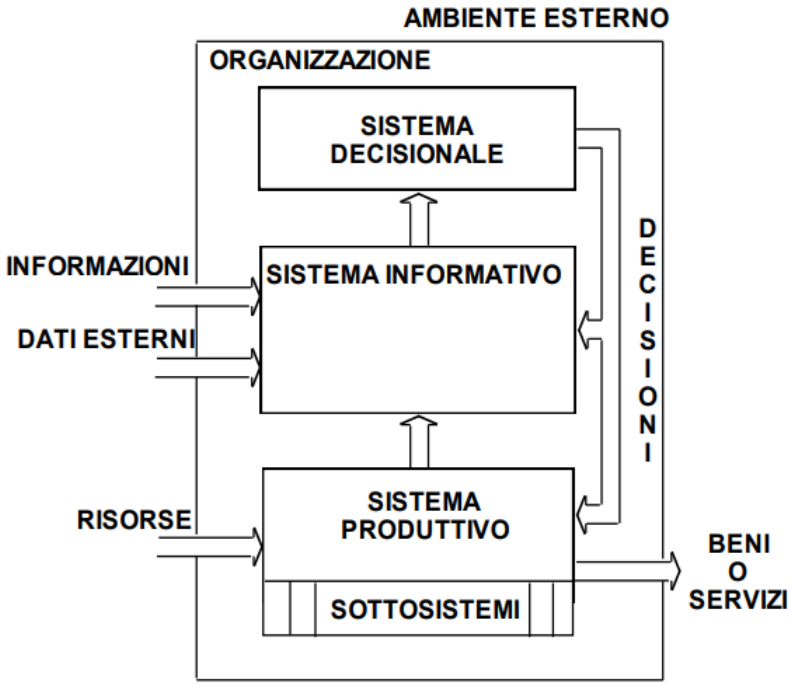
Gestione attività finanziaria secondo la normativa vigente

Gestione del personale per il calcolo della retribuzione in base al tipo di normativa contrattuale

Gestione dei servizi amministrativi e sanitari delle USL

Gestione della cartografia generale e tematica del territorio

Sistema informativo nelle organizzazioni



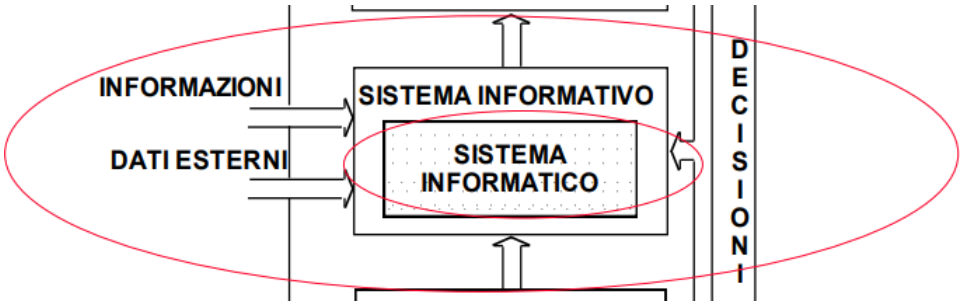
1.1.3 Sistemi Informatici

Sistema Informativo Automatizzato Quella parte del sistema informativo in cui le informazioni sono raccolte, elaborate, archiviate e scambiate usando un **sistema informatico**.

Sistema Informatico Insieme delle tecnologie informatiche e della comunicazione (**ICT**, Information and Communication Technologies) a supporto delle attività di un'organizzazione.

Terminologia

Sistema informativo ⟶ Sistema informativo automatizzato
Sistema informativo automatizzato ⟶ Sistema informatico



1.1.4 Classificazione dei sistemi informatici

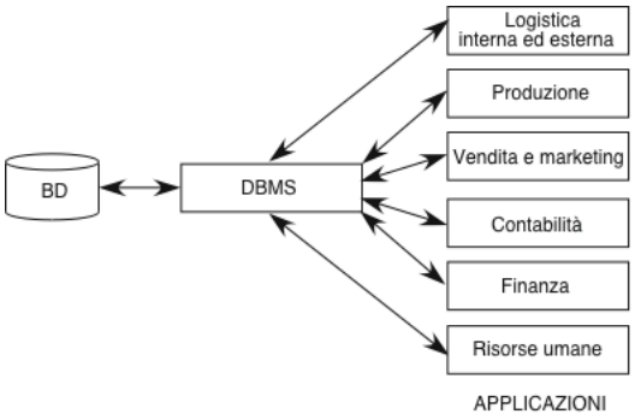
Sistemi Informatici Operativi $\longrightarrow$ Sistemi Informatici Direzionali

Sistemi Informatici Operativi I dati sono organizzati di DB. Le applicazioni si usano per svolgere le classiche attività strutturate e ripetitive dell'azione nelle aree amministrativa e finanziaria: vendite, risorse umane, produzione...

Alcune sigle:

DP Data Processing
EDP Electronic Data Processing

TPS Transaction Processing Systems



DBMS Le caratteristiche del DB sono **garantite da un sistema per la gestione della base di dati** (**DBMS**, Data Base Management System) che ha il controllo dei dati e li rende accessibili agli utenti autorizzati.

OLTP **On-Line Transaction Processing**, modo d'uso principale dei DBMS. Tradizionale elaborazione di transazioni, che realizzano processi operativi per il funzionamento di organizzazioni:

Operazioni predefinite e relativamente semplici

Ogni operazione coinvolge *pochi* dati

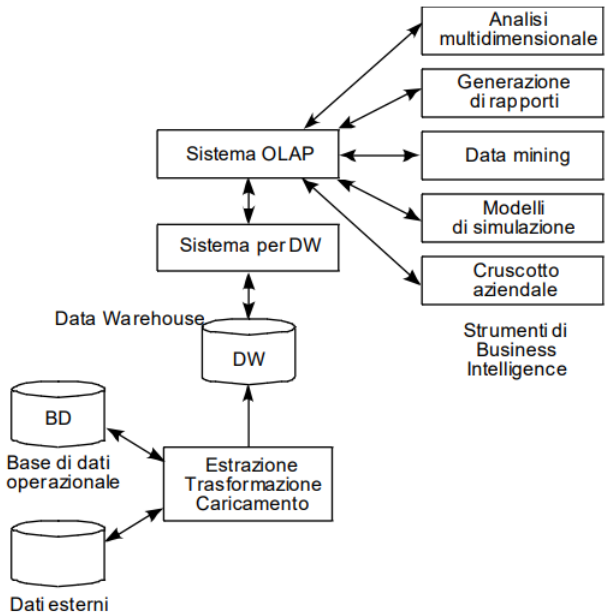
Dati di dettaglio, aggiornati

Sistemi Informatici Direzionali I dati sono organizzati in data warehouse (DW) e gestiti ad un opportuno sistema. Le applicazioni, dette di **business intelligence**, sono strumenti di supporto ai processi di controllo delle prestazioni aziendali e di decisione manageriale. Terminologia:

MIS Management Information Systems

DSS Decision Support Systems, data-based o model-based

EIS Executive Information System



OLAP **On-Line Analytical Processing** modo d'uso principale dei DW. Analisi dei dati di supporto alle decisioni:

Operazioni complesse e casuali

Ogni operazione può coinvolgere *molti* dati

Dati aggregati, storici, anche non attualissimi

Differenze tra OLTP e OLAP

	OLTP	OLAP
Scopi	Supporto operatività	Supporto decisioni
Utenti	Molti, esecutivi	Pochi, dirigenti e analisti
Dati	Analitici, relazionali	Sintetici, multidimensionali
Usi	Noti a priori	Poco prevedibili
Quantità di dati per attività	Bassa (decine)	Alta (milioni)
Orientamento	Applicazione	Soggetto
Aggiornamenti	Frequenti	Rari
Visione dei dati	Corrente	Storica
Ottimizzati per	Transazioni	Analisi

1.1.5 Requisiti per l'Analisi dei Dati

Aggregati Non interessa **un** dato, ma la **somma**, la **media**, il **minimo/massimo** di una misura...

Multidimensionale Interessa **incrociare le informazioni**, per analizzarle da punti di vista diversi e valutare i risultati del business per intervenire sui problemi critici o per cogliere nuove opportunità

Diversi livelli di dettaglio Per esempio, una volta scoperto un calo delle vendite in un determinato periodo in una specifica regione, si passa ad un'analisi dettagliata nell'area di interesse per cercare di scoprirne le cause (dimensioni con **gerarchie**)

1.1.6 Big Data

Ampio Big data è un termine ampio riferito a situazioni in cui l'approccio "schema-first" tipico di DB e DW risulta troppo restrittivo o troppo lento.

3 V Volume, Varietà, Velocità

I Big Data sono in genere associati a sistemi NoSQL, machine learning e approcci Data Lake.

1.2 DBMS

Un **DBMS** è un sistema (**software**) in grado di **gestire collezioni di dati** che siano, tra le altre cose:

Grandi

Persistenti, con un periodo di vita indipendente dalle singole esecuzioni dei programmi che le utilizzano

Condivise, usate da applicazioni diverse

garantendo **affidabilità** (resistenza a malfunzionamenti hardware e software-recovery) e **privacy** (con una disciplina e un controllo degli accessi).
Come ogni altro software, un DBMS deve essere **efficiente** (usare al meglio le risorse di spazio e tempo del sistema) ed **efficace** (rendere produttive le attività degli utilizzatori).
Un DBMS offre opportuni linguaggi per:

Definire lo schema di un DB, che va definito prima di creare dati

Scegliere le strutture dati per la memorizzazione

Memorizzare i dati **rispettando i vincoli** definiti nello schema

Recuperare e modificare i dati, interattivamente (**query language**, linguaggio di interrogazione) o da programmi

1.2.1 Dati

I dati permanenti contenuti in un DB sono divisi in due categorie:

Metadati
Descrivono datti sullo schema dei dati, utenti autorizzati, applicazioni, parametri quantitativi...
I metadati sono descritti da uno schema usando il modello dei dati usato dal DBMS e sono interrogabili con le stesse modalità previste dai dati

Dati
Rappresentazioni di certi fatti conformi alle definizioni dello schema. Hanno le seguenti caratteristiche:

Organizzati in **insiemi strutturati e omogenei**, fra i quali sono definite delle **relazioni**. La struttura dei dati e le relazioni sono **descritte nello schema** usando i meccanismi di astrazione del modello dei dati del DBMS.

Sono **molti**, sia in assoluto che rispetto ai metadati, e non possono essere gestiti in memoria temporanea

Sono **accessibili mediante transazioni**, **unità di lavoro atomiche** che **non possono avere effetti parziali**

Sono **protetti** sia **da accesso da parte di utenti non autorizzati**, sia **da corruzione dovuta a malfunzionamenti** hardware o software

Sono **utilizzabili contemporaneamente da utenti diversi**

Il **modello relazionale dei dati** è il più diffuso fra i DBMS commerciali. Il **meccanismo di astrazione** fondamentale è la **relazione** (**tabella**), sostanzialmente un insieme di record dai campi elementari.
Lo schema di una relazione ne definisce il nome e ne descrive la struttura dei possibili elementi della relazione (insieme di attributi con il loro tipo)

Esempio

Definizione del DB

```
create database EsempioEsame
```

Definizione schema

```
create table Esami(Materia char(5), Candidato char(8), Voto int, Lode char(1),
Data char(6))
```

Inserzione dati

```
insert into Esami values ('BDSI1', '080709', 30 'S', '070900')
```

Interrogazione

```
select Candidato from Esami where Materia = "BDSI1" and Voto = 30
       > Candidato
       > 080709
```

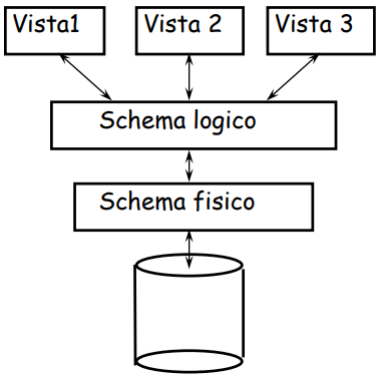
1.2.2 DDL

Data Definition Language Linguaggio per la definizione della base di dati.
Utile distinguere tre diversi livelli di descrizione dei dati (**schemi**):

Livello di **vista logica**

Livello **logico**

Livello **fisico**



Livello Logico Descrive la struttura degli insiemi di dati e delle relazioni fra loro, secondo un erto modello dei dati, senza nessun riferimento alla loro organizzazione fisica nella memoria permanente.
Esempi:

```
Studenti(Matricola char(8), Nome char(20), Login char(8), Anno int, Reddito float)
Corsi(IdeC char(8), Titolo char(20), Credito int)
Esami(Matricola char(8), IdeC char(8), Voto int)
```

Livello Fisico Descrive come vanno organizzati fisicamente i dati nelle memorie permanenti e quali strutture dati ausiliarie prevedere per facilitarne l'uso (schema fisico o interno).
Esempi: relazioni Studenti e Esami organizzate in modo seriale, Corsi organizzata sequenziale con indice, indice su Matricola.

Vista Logica Descrive come deve **apparire la struttura** del DB ad una certa applicazione (**schema esterno** o **vista**). Esempio:

```
InfCorsi (IdeC char(8), Titolo char(20), NumEsami int)
```

Nell'organizzazione di una banca, lo **schema logico** conterrà tutte le tabelle e i dati relativi ai conti correnti, ma anche al personale. Lo schema logico conserva **tutte le informazioni** della banca. Nello **schema esterno** ogni correntista potrà **accedere solo ad alcune informazioni** di suo interesse: quelle del proprio conto corrente.

Indipendenza L'approccio con tre livelli è stato proposto per garantire le proprietà di indipendenza logica e fisica dei dati, fra gli obiettivi più importanti dei DBMS.

Indipendenza fisica: i programmi applicativi non devono essere modificati in seguito a modifiche dell'organizzazione fisica dei dati

Indipendenza logica: i programmi applicativi non devono essere modificati in seguito a modifiche dello schema logico

1.2.3 DML

Data Manipulation Language Linguaggio per l'uso dei dati.
Un DBMS deve prevedere più modalità d'uso per soddisfare esigenze di diverse categorie d'utenti: GUI per accedere ai dati, linguaggio di interrogazione per i non programmatori, linguaggio di programmazione per chi sviluppa le applicazioni, linguaggio di sviluppo per le interfacce delle applicazioni.
Linguaggi vari e interfacce diverse:

Linguaggi testuali interattivi, SQL

Comandi (come quelli del linguaggi interattivo) immersi in un linguaggio ospite, come il C

Comandi (come quelli del linguaggi interattivo) immersi in un linguaggio ad hoc (come PL/SQL) con anche altre funzionalità (come grafici e stampe strutturate)

Interfacce amichevoli

1.2.4 Schemi e Istanze

Schema Descrive la **struttura dei dati**, sostanzialmente invariante nel tempo: le "classi", intestazione delle tabelle

Istanza **Valori attuali** dei dati che possono cambiare anche molto rapidamente: gli "oggetti", il corpo di ciascuna tabella

1.2.5 Meccanismi per il controllo dei dati

Caratteristica molto importante dei DBMS è il tipo di meccanismi usati per garantire le seguenti proprietà

Integrità: mantenimento delle proprietà specificate nello schema

Sicurezza: protezione da usi non autorizzati

Affidabilità: protezione da malfunzionamenti e interferenze dovute all'accesso concorrente di più utenti

1.2.6 Transazioni

Definizione Una **transazione** è una **sequenza di azioni di lettura/scrittura in memoria permanente e di elaborazione dati in memoria temporanea**, con le seguenti proprietà:

Atomicità: le transazioni che terminano prematuramente (**aborted transactions**) sono **trattate dal sistema come se non fossero mai iniziate**. Eventuali effetti sul DB sono **annullati**.

Serializzabilità: esecuzioni concorrenti di più transazioni danno come effetto quello di una esecuzione seriale

Persistenza: le **modifiche** sul DB di una transazione terminata normalmente sono **permanenti**, cioè **non alterabili da malfunzionamenti**

1.3 Progettazione

Progettare Progettare un DB significa **progettare la struttura dei dati e delle applicazioni**. La progettazione dei dati è l'attività più importante e per progettare al meglio i dati è necessario che essi siano un **modello fedele del dominio** in esame. Per questo ora parleremo della **modellazione**.

1.3.1 Modellazione

Definizione Un **modello astratto** è la **rappresentazione formale di idee e conoscenze relative ad un fenomeno**.
Aspetti di un modello:

Il **modello** è la **rappresentazione di certi fatti**.

La **rappresentazione** è **data con un linguaggio formale**.

Il **modello** è **il risultato di un processo di interpretazione**, guidato dalle idee e conoscenze possedute dal soggetto che interpreta.

La stessa realtà può utilmente essere modellata in modi diversi e a diversi livelli di astrazione.

Metodologia di progetto Per garantire prodotti di buona qualità è opportuno seguire una metodologia di progetto, con:

Articolazione delle attività in fasi (**decomposizione**)

Criteri di scelta (**strategie**)

Modelli da rappresentare

Generalità rispetto al problema in esame e agli strumenti a disposizione

Qualità del prodotto

Facilità d'uso

Progettazione della base di dati Suddivisa nelle seguenti fasi:

1. **Analisi** dei requisiti
2. Progettazione **concettuale**
3. Progettazione **logica**
4. Progettazione **fisica**

Ciascuna fase è incentrata sulla modellazione, che discuteremo quindi con riferimento alla problematica della progettazione del DB.

Modello dei dati Insieme di costrutti utilizzati per organizzare i dati di interesse e descriverne la dinamica. Il componente fondamentale è l'insieme dei **meccanismi di strutturazione** (o **costruttori di tipo**). Come nei linguaggi di programmazione, esistono meccanismi che permettono di definire nuovi tipi, così **ogni modello dei dati prevede alcuni costruttori**: per esempio, il **modello relazionale prevede il costruttore *relazione***, che permette di definire insiemi di record omogenei.

1.3.2 Aspetti del problema

Aspetto ontologico

Quale conoscenza del dominio del discorso si rappresenta? Ontologico cioè studio di ciò che si suppone esista nell'universo del discorso e che sia quindi necessario modellare. Cosa si modella:

Conoscenza concreta: i fatti

Conoscenza astratta: la struttura e i vincoli sulla conoscenza concreta

Conoscenza procedurale, comunicazioni: le operazioni base, le operazioni degli utenti, come si comunicherà con il sistema informatico

Ci concentreremo sulla conoscenza concreta e astratta.

Aspetto logico

Con quali meccanismi di astrazione si modella? Il modello dei dati a oggetti.

Modello dei dati Insieme dei meccanismi di astrazione per descrivere la struttura della conoscenza concreta.
Schema: descrizione della **struttura della conoscenza concreta** e dei **vincoli di integrità** usando un particolare modello dei dati.

Useremo come notazione grafica una **variante** dei diagrammi a oggetti (o diagrammi Entità-Relazione, diagrammi ER). Nozioni fondamentali: oggetto, tipo di oggetto, classe, ereditarietà, gerarchia fra tipi e gerarchia fra classi.

Aspetto linguistico

Con quale linguaggio formale si definisce un modello?

Aspetto pragmatico

Come si procede per costruire un modello? Metodologia da seguire nel processo di modellazione, cioè l'insieme di regole finalizzate alla costruzione del modello informatico.

1.3.3 Conoscenza concreta

La conoscenza concreta riguarda i fatti specifici che si vogliono rappresentare:

Entità, sono **ciò di cui interessa rappresentare alcuni fatti o proprietà**. Ad esempio: una descrizione bibliografica di un libro, un libro, un documento, un prestito, un utente della biblioteca...
Le **proprietà** sono **fatti che interessano solo in quanto descrivono caratteristiche di determinate entità**. Ad esempio un indirizzo interessa perché è l'indirizzo di un utente. Hanno delle classificazioni:

Primitiva/strutturata
Obbligatoria/opzionale
Univoca/multivalore
Costante/variabile
Calcolata/non calcolata

Una proprietà è una coppia attributo-valore di un certo tipo. Ogni entità appartiene ad un **tipo** che ne specifica la natura. Ogni proprietà ha associato un **dominio**, l'insieme dei possibili valori.
Una proprietà è **atomica** se il suo valore non è scomponibile, altrimenti è **strutturata**. Inoltre è **univoca** se ha valore unico, altrimenti è **multivalore**, e **totale** (obbligatoria) se ogni entità dell'universo in esame ha per essa un valore specificato, altrimenti è detta **parziale** (opzionale)

Certi fatti possono essere interpretati come proprietà in certi contesti e come entità in altri. Ad esempio, una `DescrizioniBibliografiche` con attributi `autori`, `titolo`, `editore`..., **oppure** un `Autori` con attributi `nome`, `nazionalità`...e `Editori` con `nome`, `indirizzo`...

Collezioni variabili nel tempo di entità omogenee. Ad esempio, la collezione di tutti gli utenti della biblioteca.

Associazioni fra entità

1.3.4 Modellazione ad oggetti

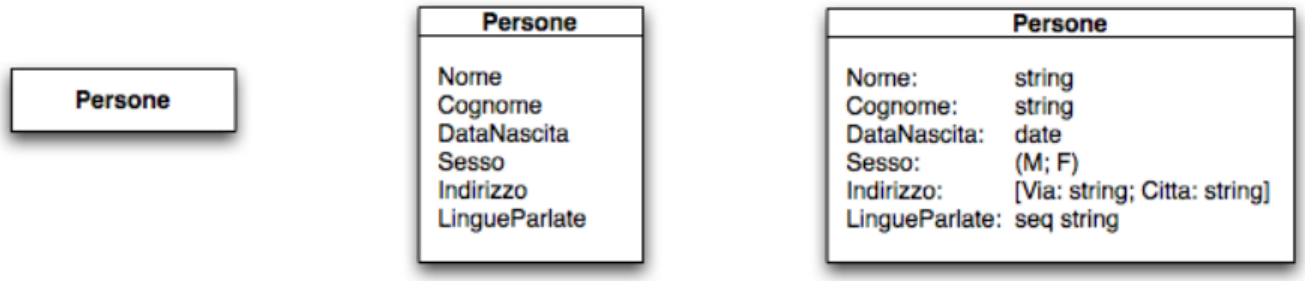
Oggetti Ad ogni entità del dominio corrisponde un oggetto del modello. Un **oggetto** è un'**entità software con stato, comportamento ed identità** che modella un'entità dell'universo.

Stato modellato da un insieme di costanti o variabili con valori di qualsiasi complessità

Comportamento dato da un insieme di procedure locali chiamate **metodi**, che modellano le operazioni di base che riguardano l'oggetto e le proprietà derivabili da altre.

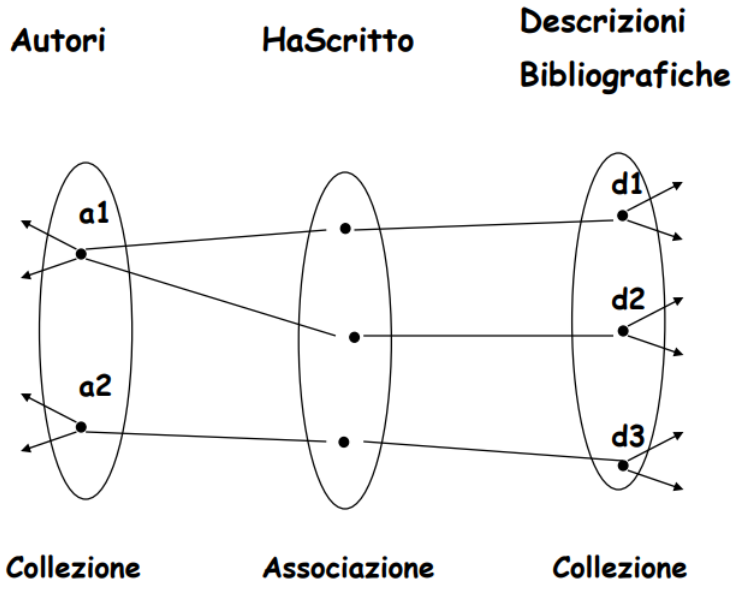
Un oggetto può rispondere a dai **messaggi**, restituendo valori memorizzati nello stato o calcolati con una procedura locale.

Classe **Insieme di oggetti dello stesso tipo**, modificabile con operatori per includere o estrarre elementi dall'insieme. Può essere specificata a diversi livelli.



Tipo oggetto Il primo passo nella costruzione di un modello consiste nella classificazione delle entità del dominio con la definizione dei tipi degli oggetti che la rappresentano.
Un **tipo oggetto definisce l'insieme dei messaggi** (**interfaccia**) **a cui può rispondere un insieme di possibili oggetti**. I nomi dei messaggi sono detti anche attributi degli oggetti.
Nei diagrammi ER i tipi oggetti non si rappresentano, perché l'attenzione è sulle collezioni e sulle associazioni. Tuttavia, la rappresentazione grafica di una collezione indica anche gli attributi del tipo oggetto associato.

Associazioni Un'**istanza di associazione** è un **fatto che correla due o più entità**, stabilendo un legame logico fra loro. Ad esempio, l'utente Tizio ha in prestito una copia della Divina Commedia.
Un'associazione `R(X, Y)` fra due collezioni di entità `X` e `Y` è un **insieme di istanze di associazione** tra elementi di `X` e di `Y` che **varia in generale nel tempo**.
Il prodotto cartesiano $X \times Y$ è il dominio dell'associazione.
Un esempio:



Un associazione è **caratterizzata da due proprietà strutturali**: **molteplicità** e **totalità**.

Vincolo di univocità Un'associazione `R(X, Y)` è **univoca rispetto a** `X` se $\forall$`x` $\in$ `X` $\exists$ al più un elemento di `Y` associato ad `x`.
Se non vale questo vincolo, l'associazione è **multivalore rispetto ad** `X`.

Cardinalità dell'associazione:

`R(X, Y)` è `1:N` se è multivalore su `X` ed univoca su `Y`

`R(X, Y)` è `N:1` se è univoca su `X` e multivalore su `Y`

`R(X, Y)` è `N:M` se è multivalore su `X` e multivalore su `Y`

`R(X, Y)` è `1:1` se è univoca su `X` ed univoca su `Y`

Qualche esempio:

`Frequenta(Studenti, Corsi)` ha cardinalità `N:M`

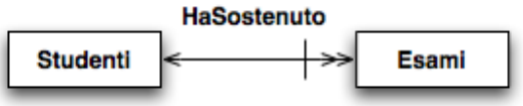
`Insegna(Professori, Corsi)` ha cardinalità `1:N`

`SuperatoDa(Esami, Studenti)` ha cardinalità `N:1`

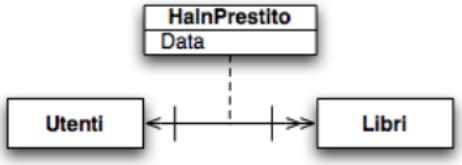
`Dirige(Professori, Dipartimenti)` ha cardinalità `1:1`

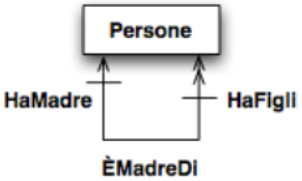
Vincolo di totalità Un associazione `R(X, Y)` è **totale** (o surgettiva) su `X` se $\forall$ `x` $\in$ `X` $\exists$ almeno un elemento di `Y` associato ad `x`.
Se non vale questo vincolo, l'associazione è **parziale rispetto a** `X`.
Ad esempio, `Insegna(Professori, Corsi)` è totale su `Corsi` perché non può esistere un corso senza il corrispondente docente.

Rappresentazione Un'associazione si rappresenta con una linea che collega le classi che rappresentano le due collezioni. La linea è etichettata con il nome dell'associazione, di solito scelto utilizzando un predicato.
L'univocità dell'associazione rispetto ad una classe si rappresenta disegnando una freccia singola sulla linea che esce dalla classe ed entra nella destinazione. L'assenza di tale vincolo è indicata da una freccia doppia.
Similmente, la parzialità è rappresentata da un taglio vicino alla freccia, mentre la totalità è rappresentata dall'assenza del taglio.



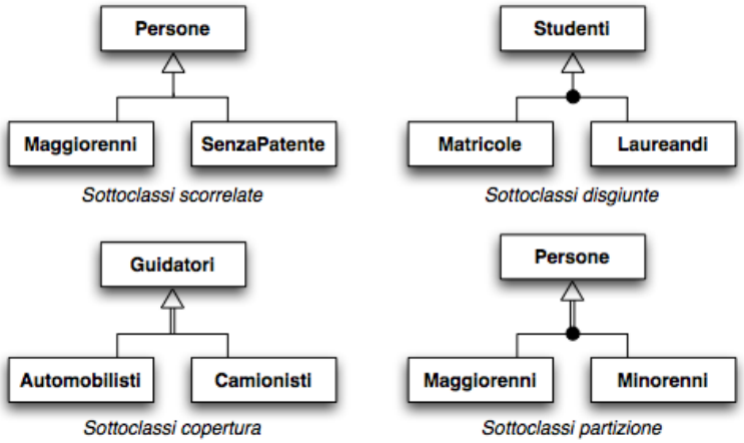
Ogni esame riguarda uno ed un solo studente
Parzialità e assenza di univocità sugli esami
superati da uno studente.





Possono avere proprietà ed essere ricorsive.

1.3.5 Sottoclassi



Vincoli

Vincolo intensionale: C sottoclasse di C' ⇒ tipo degli elementi di C è sottotipo del tipo degli elementi di C'

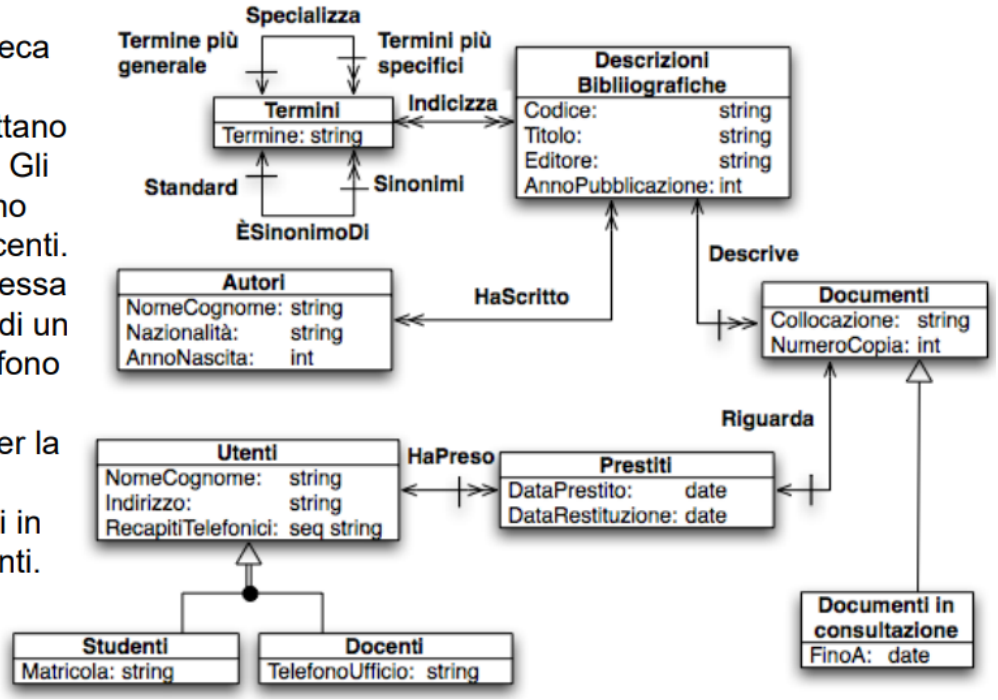
Vincolo estensionale: C sottoclasse di C' ⇒ gli elementi di C sono un sottoinsieme degli elementi di C'

Disgiunzione: ogni coppia di sottoclassi è disgiunta, priva di elementi comuni (pallino nero) **sottoclassi disgiunte**)

Copertura: l'unione degli elementi delle sottoclassi coincide con l'insieme degli elementi della superclasse (freccia con doppia asta) (**sottoclassi copertura**)

1.3.6 Un esempio elaborato

Gli utenti della biblioteca vengono sospesi dal servizio se non rispettano le regole del prestito. Gli utenti regolari possono essere studenti o docenti. Di uno studente interessa anche la matricola e di un docente anche il telefono dell'ufficio.
Alcune opere sono per la sola consultazione e possono essere presi in prestito solo da docenti.



1.3.7 Conoscenza astratta

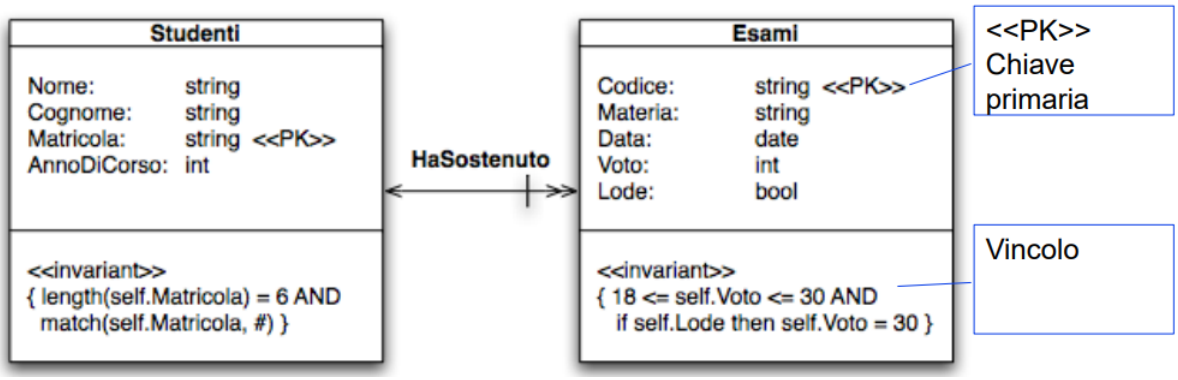
La conoscenza astratta riguarda i **fatti generali che descrivono**

la **struttura della conoscenza concreta**, come collezioni, tipi entità, associazioni...

le **restrizioni sui valori** possibili della conoscenza concreta e sui modi in cui essi possono evolvere nel tempo (**vincoli d'integrità**, statici e dinamici)

le **regole per derivare fatti nuovi** da altri noti

Vincoli Possono essere descritti in **modo dichiarativo** (da preferire), con formule di calcolo dei predicati, oppure mediante controlli da eseguire nelle operazioni.



1.4 Costruzione

1. Analisi dei requisiti → specifica dei requisiti, schemi di settore
2. Progettazione
   - Progettazione **concettuale** (→ schema concettuale), **logica** (→ schema logico), **fisica** (→ schema fisico) dei dati
   - Progettazione delle applicazioni
3. Realizzazione

Spesso consideriamo l'analisi dei requisiti come parte della progettazione.

1.4.1 Analisi dei requisiti

Analizza il sistema esistente e **raccoglie requisiti informali**. Dopodiché **elimina le ambiguità** e la disuniformità, **raggruppando frasi relative a diverse categorie** di dati, vincoli e operazioni.
Costruisce un **glossario**, **disegna lo schema** di settore, **specifica le operazioni** e **verifica la coerenza tra operazioni e dati**.

Documentazione descrittiva In generale, il linguaggio naturale è pieno di ambiguità e fraintendimenti, che bisogna evitare per quanto possibile. Come prima approssimazione si può seguire queste regole:

Studiare e comprendere il sistema informativo ed i bisogni informativi di tutti i settori dell'organizzazione

Scegliere il corretto **livello di astrazione**

Standardizzare la scrittura delle frasi

Suddividere le frasi articolate

Separare le frasi sui **dati** da quelle sulle **funzioni**

Organizzare i concetti e i termini Regole generali

Eliminare le ambiguità, le imprecisioni e la disuniformità: individuare omonimi e sinonimi e unificare i termini

Riorganizzare le frasi per **concetti**, ovvero ottenendo diverse categorie di dati, vincoli e operazioni

Costruire un **glossario** dei termini

Disegnare lo schema

Specificare le operazioni

Verificare la coerenza fra le operazioni e i dati

1.5 Modello Relazionale

Origini Proposto da E. F. Codd nel 1970 per favorire l'indipendenza dei dati, disponibile in DBMS reali dal 1981 (non è facile implementare l'indipendenza con efficienza e affidabilità). Si basa sul **concetto matematico di relazione** con una variante, naturalmente rappresentata come **tabella**.

1.5.1 Relazione matematica

Dalla teoria degli insiemi Dati n insiemi anche non distinti $D_1, \ldots, D_n$.
Il **prodotto cartesiano** $D_1 \times \ldots \times D_n$ è l'insieme di tutte le n-uple $(d_1, \ldots, d_n)$ tali che $d_1 \in D_1, \ldots, d_n \in D_n$

Una **relazione matematica** su $D_1, \ldots, D_n$ **è un sottoinsieme di** $D_1 \times \ldots \times D_n$, con $D_1, \ldots, D_n$ detti **domini della relazione**.

Un esempio Dati $D_1 = \{a, b\}$ e $D_2 = \{x, y, z\}$.
Il prodotto cartesiano è l'insieme $D_1 \times D_2 = \{(a, x), (a, y), (a, z), (b, x), (b, y), (b, z)\}$
Una relazione r potrebbe essere $r \subset D_1 \times D_2 = \{(a, x), (a, z), (b, y)\}$

Proprietà Una relazione matematica è un insieme di n-uple ordinate $(d_1, \ldots, d_n)$ tali che $d_1 \in D_1, \ldots, d_n \in D_n$.
Osservazioni: una relazione è un insieme, quindi

non c'è ordinamento fra le n-uple

le n-uple sono distinte

ciascuna n-upla è ordinata, cioè l'i-esimo valore proviene dall'i-esimo dominio

Tabelle Una tabella rappresenta una relazione se:

I valori di ogni colonna sono fra loro omogenei

Le righe sono diverse fra loro

Le intestazioni delle colonne sono diverse fra loro

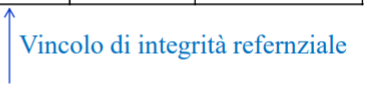
In una tabella che rappresenta una relazione **l'ordinamento** tra le righe e l'ordinamento tra le colonne **è irrilevante**

1.5.2 Valori

Il modello relazionale è basato sui valori Ciò significa che i riferimenti fra dati in relazioni diverse sono rappresentati per mezzo di valori dei domini che compaiono nelle n-uple.

Studenti

Nome	Matricola	Provincia	AnnoNascita
Isaia	071523	PI	1982
Rossi	067459	LU	1984
Bianchi	079856	LI	1983
Bonini	075649	PI	1984



Esami

Materia	Candidato*	Data	Voto
BD	071523	12/01/06	28
BD	067459	15/09/06	30
FP	079856	25/10/06	30
BD	075649	27/06/06	25
LMM	071523	10/10/06	18

Vantaggi

Indipendenza delle strutture fisiche che possono cambiare dinamicamente, che potremmo avere anche con puntatori di alto livello. La rappresentazione logica dei dati (che è costituita dai soli valori) non fa riferimento a quella fisica.

Si **rappresenta solo ciò che è rilevante** dal punto di vista dell'applicazione

I **dati** sono **portabili** più facilmente da un sistema all'altro

I **puntatori** sono **direzionali**

1.5.3 Meccanismi

Definizione I meccanismi per definire una base di dati con il modello relazionale sono l'**ennupla** e la **relazione**.

Tipo ennupla Un tipo ennupla T è un insieme finito di coppie ⟨attributo, tipo elementare⟩
Se T è un tipo ennupla, R(T) è lo schema della relazione R, quindi **lo schema di un DB è l'insieme di schemi di relazione $\mathbf{R}_i(\mathbf{T}_i)$**. Un'**istanza** di uno schema R(T) è un insieme finito di ennuple di tipo T.

Informazione incompleta Per rappresentare un'informazione incompleta (es.: l'assenza del secondo nome) **non bisogna usare elementi del dominio** come lo 0, stringa vuota, "99"....
Questo perché potrebbero non esistere valori "non utilizzati", e se esistono potrebbero diventare significativi. Inoltre, in fase di utilizzo (nei programmi) bisognerebbe tenere conto ogni volta del "significato" di questi valori.

Il **valore nullo** denota l'**assenza di un valore del dominio e non è un valore del dominio**.
Quindi t[A] per ogni attributo A è un valore del dominio dom(A) oppure è il valore nullo NULL.
Si possono (e **devono**) imporre restrizioni sulla presenza di valori sulli.

Vincoli d'Integrità Esistono istanze di DB che, nonostante siano sintatticamente corrette, non rappresentano informazioni possibili per l'applicazione e che quindi **generano informazioni prive di significato**. Ad esempio un voto di 32, o due studenti con la stessa matricola.
Uno **schema relazionale è costituito da un insieme di schemi di relazione e un insieme di vincoli d'integrità** sui possibili valori delle estensioni delle relazioni.
Un **vincolo d'integrità** è una **proprietà che deve essere soddisfatta dalle istanze che rappresentano informazioni corrette** per l'applicazione.

Vincoli Intrarelazionali: sono vincoli che **devono essere rispettati dai valori contenuti nella relazione considerata**. Vincoli **sui valori** (o **di dominio**), vincoli **di ennupla**.

Vincoli Interrelazionali: sono vincoli che **devono essere rispettati da valori contenuti in relazioni diverse**.

Chiave Informalmente, una **chiave** è un **insieme di attributi che identificano le ennuple di una relazione**. Formalmente, **un insieme K di attributi è *superchiave* per** r **se** r ***non*** **contiene due ennuple distinte** t_1 **e** t_2 **con** $t_1[K] = t_2[K]$.
K è chiave per r **se è superchiave minimale per** r, cioè se non contiene altre superchiavi.
Un esempio classe è la matricola: è superchiave ed è un solo attributo, quindi è minimale.

Una relazione non può contenere ennuple distinte ma con valori uguali. Ogni relazione ha **sicuramente** come superchiave l'insieme di tutti gli attributi su cui è definita, quindi ogni relazione ha (almeno) una chiave.
L'esistenza delle chiavi garantisce l'accesso di ciascun dato della base di dati e permettono di correlare i dati in relazioni diverse.
Un valore nullo in una chiave non permette di identificare le ennuple o realizzare i riferimenti con altre relazioni. Una **chiave primaria** è una chiave su cui non sono ammessi valori nulli: si denota sottolineando il nome dell'attributo es. matricola.

Integrità referenziale Le informazioni in relazioni diverse sono correlate attraverso valori comuni. In particolare, **vengono prese in considerazione i valori delle chiavi primarie**, quindi le **correlazioni devono essere coerenti**.
Un **vincolo di integrità referenziale** (foreign key) **fra gli attributi X di una relazione** $\mathbf{R_1}$ **e un'altra relazione** $\mathbf{R_2}$ **impone ai valori su X in** $\mathbf{R_1}$ **di comparire come valori della chiave primaria di** $\mathbf{R_2}$.