

Basi di dati

VR443470

novembre 2022

# Indice

<b>1</b>	<b>Introduzione</b>	<b>3</b>
1.1	Sistemi informativi, informazioni e dati . . . . .	3
1.2	Basi di dati e sistemi di gestione di basi di dati . . . . .	4
1.3	Linguaggi per basi di dati . . . . .	5
1.4	Modelli dei dati . . . . .	6
1.5	Astrazione (architettura) dei DBMS . . . . .	7
1.6	Indipendenza dei dati . . . . .	8
<b>2</b>	<b>Metodologie e modelli per il progetto</b>	<b>9</b>
2.1	Ciclo di vita dei sistemi informativi . . . . .	9
2.2	Metodologie di progettazione e basi di dati . . . . .	11
2.3	Il modello Entità-Relazione (E-R) . . . . .	12
2.4	I costrutti principali del modello . . . . .	13
2.4.1	Entità . . . . .	13
2.4.2	Relazioni (o associazioni) . . . . .	14
2.4.3	Attributi . . . . .	15
2.5	Altri costrutti del modello . . . . .	16
2.5.1	Cardinalità delle relazioni . . . . .	16
2.5.2	Cardinalità degli attributi . . . . .	17
2.5.3	Identificatori . . . . .	18
2.5.4	Generalizzazioni . . . . .	20

# 1 Introduzione

## 1.1 Sistemi informativi, informazioni e dati

Ogni organizzazione è dotata di un *sistema informativo*, che organizza e gestisce le informazioni necessarie per perseguire gli scopi dell'organizzazione stessa. Per indicare la **porzione automatizzata del sistema informativo** viene di solito utilizzato il termine *sistema informatico*.

Nei sistemi informatici le informazioni vengono rappresentate per mezzo di *dati*, che hanno bisogno di essere interpretati per fornire informazioni. Esiste una differenza sottile tra dato e informazioni. Solitamente i primi, se presi da soli, non hanno significato, ma, una volta interpretati e correlati opportunamente, essi forniscono informazioni, che consentono di arricchire la conoscenza:

**Informazione:** notizia, dato o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni, modi di essere;

**Dato:** ciò che è immediatamente presente alla conoscenza, prima di ogni elaborazione. In informatica, sono elementi di informazione costituiti da simboli che devono essere elaborati.

**[ESAME] Definizione base di dati:** Una *base di dati* è una collezione di dati, utilizzati per rappresentare con tecnologia informatica le informazioni di interesse per un sistema informativo.

## 1.2 Basi di dati e sistemi di gestione di basi di dati

Inizialmente, venne adottato un “approccio convenzionale” alla gestione dei dati. Esso **sfruttava** la presenza di archivi o **file per memorizzare e per ricercare dati**. Tuttavia, i metodi di accesso e condivisione erano semplici e banali. Infatti, erano presenti numerosi **problemi**:

- ✗ **Accesso sequenziale**: la scarsa efficienza nell’accesso ai dati su file rendeva lento l’accesso a tali informazioni;
- ✗ **Ridondanza**: i dati di interesse per più programmi sono replicati tante volte quanti sono i programmi che li utilizzano, con evidente ridondanza e possibilità di incoerenza;
- ✗ **Inconsistenza**: una diretta conseguenza della ridondanza. Con la presenza di più copie di un determinato dato, l’eventuale cambiamento di uno solo potrebbe portare a questo effetto;
- ✗ **Progettazione duplicata**: per ogni programma viene replicata la progettazione.

La **soluzione** è arrivata negli anni ’80 con l’avvento delle **basi di dati**. Quest’ultime gestiscono in modo integrato e flessibile le informazioni di interesse per diversi soggetti.

**[ESAME] Definizione DBMS**: Un *sistema di gestione di basi di dati* (in inglese *Data Base Management System*, **DBMS**) è un sistema software in grado di gestire collezioni di dati che siano:

- ✓ **Grandi**;
- ✓ **Condivise**;
- ✓ **Persistenti**.

assicurando allo stesso tempo:

- ★ **Affidabilità**;
- ★ **Privatezza**;
- ★ **Accesso efficiente**.

Il **vantaggio** di utilizzare un DBMS è stato evidenziato nella definizione. Quindi:

- ✓ **Maggiore astrazione** poiché le sue funzioni estendono il *file system*, fornendo la possibilità di accesso condiviso agli stessi dati da parte di più utenti e applicazioni;
- ✓ **Maggiore efficacia** poiché le operazioni di accesso ai dati si basano su un linguaggio di interrogazione.

### 1.3 Linguaggi per basi di dati

Su un DBMS è possibile specificare operazioni di vario tipo, ma principalmente si distinguono in due categorie:

- **Linguaggi di definizione dei dati** (*Data Definition Language*, abbreviato con **DDL**) utilizzati per definire gli schemi logici, esterni e fisici e le autorizzazioni per l'accesso;
- **Linguaggi di manipolazione dei dati** (*Data Manipulation Language*, abbreviato con **DML**) utilizzati per l'interrogazione e l'aggiornamento delle istanze di basi di dati:
  - *Linguaggio di interrogazione*, estrae informazioni da una base di dati (SQL, algebra relazionale);
  - *Linguaggio di manipolazione*, popola la base di dati, modifica il suo contenuto con aggiunte, cancellazioni e variazioni sui dati (SQL).

## 1.4 Modelli dei dati

**Definizione modello dei dati:** Un *modello dei dati* è un insieme di concetti utilizzati per organizzare i dati di interesse e descriverne la struttura in modo che essa risulti comprensibile ad un elaboratore.

Ogni modello dei dati fornisce **meccanismi di strutturazione**, analoghi ai **costruttori** di tipo dei linguaggi di programmazione (es: Java), che permettono di definire nuovi tipi sulla base di tipi predefiniti (elementari) e costruttori di tipo. Quindi, i *costruttori* consentono di:

- ☞ *Definire* le **strutture dati** che **conterranno le informazioni** della base di dati;
- ☞ *Specificare* le **proprietà** che **dovranno soddisfare le istanze** di informazione che saranno contenuto nelle strutture dati.

**Definizione schemi e istanze:** È molto importante distinguere gli **schemi** e le **istanze** dal concetto di modello dei dati:

- **Schema:** parte invariante nel tempo, è costituita dalle caratteristiche dei dati. In altre parole, è la descrizione della struttura e delle proprietà di una specifica base di dati fatta utilizzando i costrutti del modello dei dati;
- **Istanza o stato:** parte variabile nel tempo, è costituita dai valori effettivi. Quest'ultimi, in un certo istante, popolano le strutture dati della base di dati.

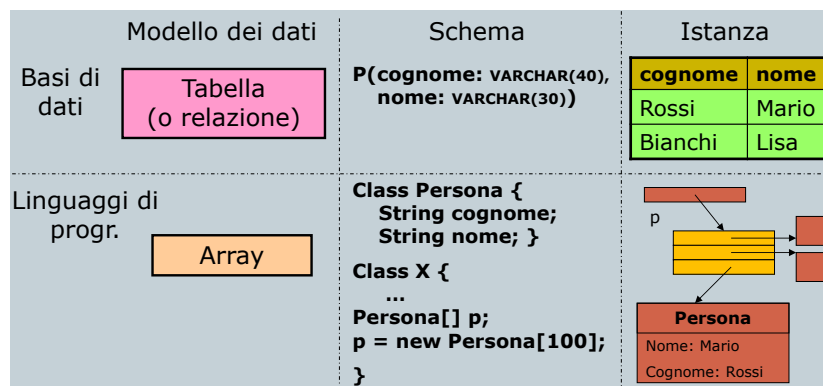


Figura 1: Esempio di modello di dati, schema e istanza.

## 1.5 Astrazione (architettura) dei DBMS

Esiste un'architettura standardizzata per i DBMS, la quale si caratterizza su tre livelli: **esterno**, **logico** e **interno**:

- ☆ *Schema logico*. È la rappresentazione della **struttura** e delle **proprietà** della **base di dati** definita attraverso i costrutti del modello dei dati del DBMS. In altre parole, descrive l'intera base di dati per mezzo del modello logico adottato dal DBMS (quindi relazione o ad oggetti).
- ★ *Schema interno*. È la rappresentazione della base di dati per mezzo delle **strutture fisiche di memorizzazione** (e.g. file sequenziale, file hash, ecc.).
- ☆ *Schema esterno*. Descrive una **porzione dello schema logico** di interesse per uno **specifico** utente o applicazione. Possono esistere più schemi esterni che consentono di avere punti di vista differenti senza cambiare la logica di base.

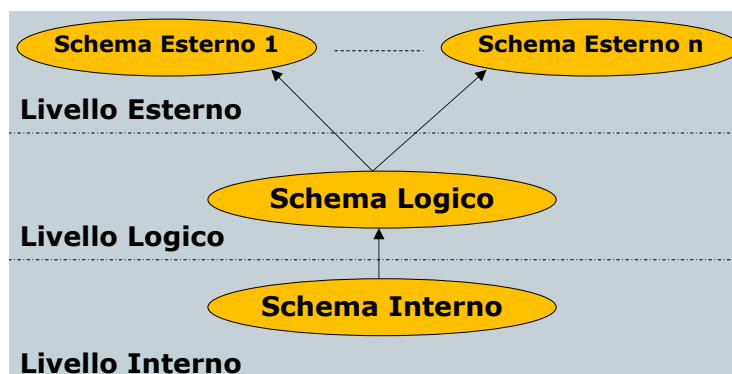


Figura 2: Architettura generale di un DBMS.

## 1.6 Indipendenza dei dati

L'architettura a livelli definita nel paragrafo 1.5 garantisce l' **indipendenza dei dati**, la **proprietà più importante** dei DBMS. L'**obbiettivo** è quello di poter fornire all'utente una base di dati in grado di interagire con un elevato livello di astrazione. Esistono due tipi di indipendenza:

- ☛ **Indipendenza fisica.** Lo schema logico della base di dati è completamente indipendente dallo schema interno. Quindi, l'interazione con il DBMS può essere effettuato in modo indipendente dalla struttura fisica dei dati.

**Vantaggio:** le modifiche **non** influiscono sullo schema logico, cioè sulle applicazioni che lo utilizzano.

- ☛ **Indipendenza logica.** Gli schemi esterni (definizione nel paragrafo: 1.5) della base di dati sono **indipendenti** dallo schema logico. Quindi, è possibile interagire con il livello esterno in modo indipendente dal livello logico.

**Vantaggio:**

- Aggiunta/Modifica** di uno schema esterno in base alle esigenze di un nuovo utente, senza modificare lo schema logico;
- Modifica** di uno schema logico mantenendo inalterate le strutture esterne.



## 2 Metodologie e modelli per il progetto

### 2.1 Ciclo di vita dei sistemi informativi

La progettazione di una base di dati costituisce solo una delle componenti del processo di sviluppo di un sistema informativo complesso e va quindi inquadrata in un contesto più ampio quello del **ciclo di vita** dei sistemi informativi:

- ☛ **Studio di fattibilità.** Definisce i costi delle varie alternative possibili e stabilisce le priorità di realizzazione delle varie componenti del sistema.
- ☛ **Raccolta e analisi dei requisiti.** Individua le proprietà e le funzionalità che il sistema informativo deve avere producendo una descrizione completa, ma generalmente informale.
- ☛ **Progettazione.** Si divide in due fasi:
  - **Progettazione dei dati.** Individua la struttura e l'organizzazione che i dati devono avere.
  - **Progettazione delle applicazioni.** Definizione delle caratteristiche dei programmi applicativi.
- ☛ **Implementazione (su un DBMS).** È la realizzazione del sistema informativo secondo la struttura e le caratteristiche fornite durante la fase di progettazione. In questa fase viene costruita e popola la base di dati.
- ☛ **Validazione e collaudo.** Verifica il corretto funzionamento e la qualità del sistema informativo.
- ☛ **Funzionamento.** Il sistema informativo diventa operativo ed esegue i compiti per i quali è stato progettato.

Spesso il processo **non** è strettamente sequenziale. Infatti, come si vede dalla seguente figura, durante l'esecuzione di una delle attività sopraelencate, è necessario rivedere decisioni prese nell'attività precedente.

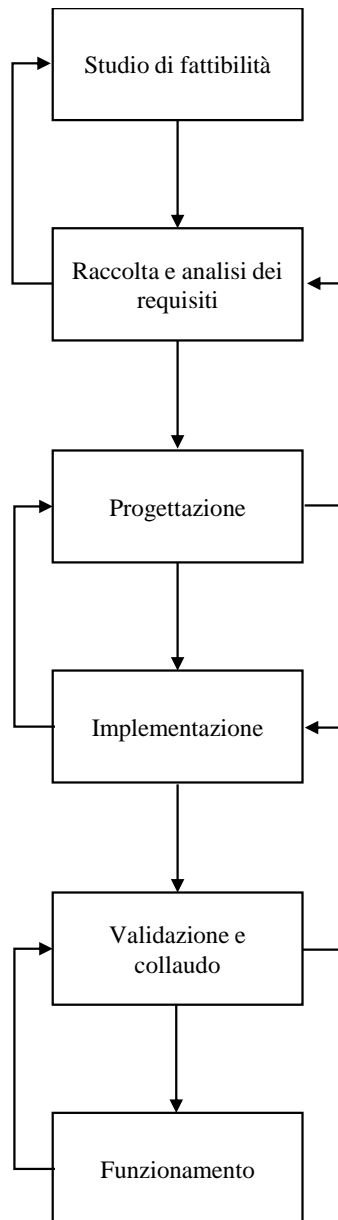


Figura 3: Ciclo di vita di un sistema informativo.

## 2.2 Metodologie di progettazione e basi di dati

Una **metodologia di progettazione** consiste in:

- ✓ **Decomposizione** dell'intera attività di progetto in passi successivi indipendenti tra loro.
- ✓ **Strategie** da seguire nei vari passi e **criteri** nel caso di alternative.
- ✓ **Modelli di riferimento** per descrivere i dati in ingresso e uscita delle varie fasi.

Le **proprietà** che una metodologia deve garantire sono:

- ★ **Generalità** rispetto alle applicazioni e ai sistemi in gioco;
- ★ **Qualità del prodotto** in termini di correttezza, completezza ed efficienza rispetto alle risorse impiegate;
- ★ **Facilità d'uso** delle strategie e dei modelli di riferimento.

Negli anni si è *consolidata una metodologia* di progetto che ha dato prova di soddisfare pienamente le proprietà descritte. Si basa sull'idea di separare le decisioni relative a “cosa” rappresentare in una base di dati (prima fase), da quelle relative a “come” farlo (seconda e terza fase):

### ☛ **Progettazione concettuale.**

**Obiettivo:** rappresentare le specifiche informali della realtà di interesse in termini di una descrizione formale e completa. La **rappresentazione deve essere indipendente** dai criteri di rappresentazione utilizzati nei sistemi di gestione di basi di dati.

**Prodotto di questa fase:** schema concettuale. È un documento formale che rappresenta il contenuto della base di dati in modo indipendente dall'implementazione (DBMS).

**Applicazione:** cercare di rappresentare il **contenuto informativo** della base di dati, senza preoccuparsi né della modalità con le quali queste informazioni verranno codificate in un sistema reale, né dell'efficienza dei programmi che faranno uso di queste informazioni.

### ☛ **Progettazione logica.**

**Obiettivo:** traduzione dello schema concettuale prodotto nella fase precedente, in termini del modello di rappresentazione dei dati adottato dal sistema di gestione di base di dati a disposizione.

**Prodotto di questa fase:** schema logico.

**Applicazione:** durante la traduzione, le scelte progettuali si devono basare anche su criteri di ottimizzazione delle operazioni da effettuare sui dati.

### ☛ **Progettazione fisica.**

**Obiettivo:** lo schema logico viene completato con la specifica dei parametri fisici di memorizzazione dei dati.

**Prodotto di questa fase:** schema fisico.

### 2.3 Il modello Entità-Relazione (E-R)

Il **modello Entità-Relazione** è un modello **concettuale** di dati, quindi utilizzato nella **progettazione concettuale**, e fornisce una serie di strutture, chiamati **costrutti**, atte a descrivere la realtà di interesse in una maniera facile da comprendere e che prescinde dai criteri di organizzazione dei dati nei calcolatori.

I costrutti vengono utilizzati per **definire schemi** che **descrivono l'organizzazione e la struttura delle occorrenze** (o **istanze**) **dei dati**, ovvero, dei valori assunti dai dati al variare del tempo.

Si possono riassumere le caratteristiche del modello Entità-Relazione:

- ☛ **Modello concettuale.** Utilizzato durante la progettazione concettuale (definizione al paragrafo 2.2) di una base di dati.
- ☛ **Strumenti formali.** Vengono messi a disposizione diversi strumenti per definire la **struttura** e le **proprietà** di una base di dati (*esempio i costrutti*).
- ☛ **Indipendente dalla tecnologia.** Essendo un modello astratto, l'obiettivo è quello di definire la struttura e le proprietà della base di dati (non di implementarla!).
- ☛ **Formale.** È facile da utilizzare nonostante non ammetta ambiguità.
- ☛ **Grafico.** La sintassi è prettamente grafica e questo aumenta anche la leggibilità.

## 2.4 I costrutti principali del modello

Si analizzano i principali costrutti di questo modello: entità (pagina 13), relazioni (pagina 14) e attributi (pagina 15).

### 2.4.1 Entità

**Definizione.** Rappresentano **classi di oggetti** (per esempio, fatti, cose, persone) **che hanno proprietà comuni ed esistenza “autonoma” ai fini dell’applicazione di interesse**. Per esempio, “città, dipartimento, impiegato, acquisto e vendita” sono entità di un’applicazione aziendale. Inoltre, **ogni entità ha un nome identificativo**, il quale deve essere **univoco**. In sintesi:

- Hanno **proprietà comuni**;
- Hanno **esistenza autonoma**;
- Hanno **identificazione univoca**.

**Sintassi grafica.**



Figura 4: Sintassi grafica dell’entità.

**Istanza (o occorrenza).** Un’istanza (o occorrenza) di un’entità è un **oggetto della classe che l’entità rappresenta**. Le città di Roma, Milano e Palermo sono esempi di occorrenze dell’entità “Città”.

**Attenzione!** L’istanza di un’entità **non è un valore** che identifica un oggetto (per esempio, il cognome dell’impiegato o il suo codice fiscale), **ma è l’oggetto stesso** (l’impiegato in “carne e ossa”). Quindi, un’istanza **ha un’esistenza indipendente dalle proprietà a esso associate**.

Un’istanza dell’entità  $E$  è un oggetto appartenente alla classe rappresentata da  $E$ . Si indica con  $I(E)$  l’insieme delle istanze di  $E$  che esistono nella base di dati in un certo istante.

### 2.4.2 Relazioni (o associazioni)

**Definizione.** Rappresentano **legami logici**, significativi per l'applicazione di interesse, **tra due o più entità**. Per esempio, “Residenza” è una relazione che sussiste tra le entità “Città” e “Impiegato”. Nello schema E-R, **ogni relazione ha un nome identificativo univoco**.

Le relazioni possono essere di **tipo**:

- **Ricorsive**, ovvero **relazioni tra un'entità e se stessa**. Per esempio, la relazione “Collega” sull'entità “Impiegato” connette coppie di impiegati che lavorano insieme.
- ***n*-arie**, ovvero **relazioni che coinvolgono più di due entità**. Per esempio, la relazione “Fornitura” tra le tre entità “Fornitore, Prodotto e Dipartimento” descrive il fatto che un fornitore rifornisce un dipartimento di un certo prodotto.

**Nota fondamentale:** per eseguire una relazione, le entità devono essere tutte piene o con almeno un dato all'interno.

**Sintassi grafica.** Una relazione  $R$  si rappresenta nello schema con un **rombo** a cui si collegano attraverso linee spezzate le entità coinvolte nella relazione. Il nome della relazione viene scritto a fianco del rombo.

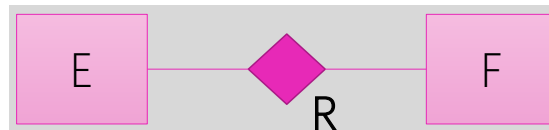


Figura 5: Sintassi grafica della relazione.

**Istanza (o occorrenza).** Un'istanza di relazione è un'ennupla costituita da **istanze di entità, una per ciascuna delle entità coinvolte**.

Data una relazione  $R$  tra  $n$  entità  $E_1, \dots, E_n$ , un'istanza della relazione  $R$  è una ennupla di istanze di entità:

$$(e_1, \dots, e_n) \text{ dove } e_i \in I(E_i), 1 \leq i \leq n$$

Infine, esiste una relazione importante. Data una relazione  $R$  tra  $n$  entità, vale sempre la seguente proprietà sull'insieme delle istanze di  $R(I(R))$ :

$$I(R) \subseteq I(E_1) \times \dots \times I(E_n)$$

### 2.4.3 Attributi

**Definizione.** Descrivono le **proprietà elementari** di entità o relazioni che sono di interesse ai fini dell'applicazione. Per esempio, “Cognome, Stipendio ed Età” sono possibili attributi dell'entità “Impiegato”.

Un attributo associa a ciascun istanza di entità (o relazione) **uno e un solo** valore appartenente a un insieme, chiamato **dominio**, che contiene i valori ammissibili per l'attributo. Per esempio, l'attributo “Cognome” dell'entità “Impiegato” può avere come dominio l'insieme delle stringhe di 20 caratteri.

L'attributo può essere visto come una **funzione che ha come dominio le istanze dell'entità** (o relazione) e come **codominio l'insieme dei valori ammissibili**:

$$f_a : I(E) \rightarrow D$$

- $a$  è un attributo dell'entità  $E$ ;
- $I(E)$  è l'insieme delle istanze di  $E$ ;
- $D$  è l'insieme dei valori ammissibili.

**Sintassi grafica.**

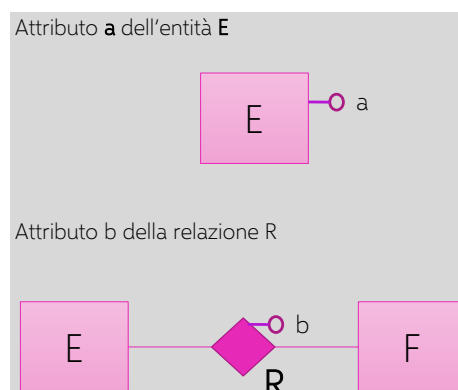


Figura 6: Sintassi grafica dell'attributo.

**Istanza (o occorrenza).** Dato un attributo  $a$  di un'entità  $E$  (o relazione  $R$ ), un'istanza di  $a$  è il valore  $v$  che esso assume su un'istanza di  $E$  (o istanza di  $R$ ).

Quindi, data un'istanza  $e$  dell'entità  $E$  (o relazione  $R$ ), l'istanza di un suo attributo  $a$  si ottiene dalla funzione  $f_a$  applicata a  $e$ :

$$\text{valore di } a \text{ su } e = f_a(e)$$

#### Attributi composti.

Questo tipo di attributo viene introdotto solo a fini didattici, ma l'obiettivo è quello di usare unicamente gli attributi normali. Talvolta potrebbe tornare comodo raggruppare **attributi di una medesima entità o relazione che presentano affinità nel loro significato o uso**: tale insieme prende il nome di **attributo composto**. Per esempio, gli attributi “Via, Numero civico e CAP” dell'entità “Persona” per formare l'attributo composto “Indirizzo”.

## 2.5 Altri costrutti del modello

I rimanenti costrutti del modello E-R sono le cardinalità delle relazioni e degli attributi e gli identificatori.

### 2.5.1 Cardinalità delle relazioni

**Definizione.** Le **cardinalità** vengono **specificate per ciascuna entità collegata ad una relazione** e descrivono il **numero minimo e massimo di occorrenze** di relazione a cui una occorrenza dell'entità può partecipare.

Più formalmente, data una relazione  $R$  i vincoli di cardinalità vengono specificati per ogni entità  $E_i$  coinvolta nella relazione  $R$  e specificano: il numero massimo e il numero minimo di istanze di  $R$  a cui un'istanza di  $E_i$  deve/può partecipare.

In parole povere, dicono quante volte, in una relazione tra entità, un'istanza di una di queste entità può essere legata a istanze delle altre entità coinvolte. **Per esempio**, in una relazione “Assegnamento” tra le entità “Impiegato” e “Incarico” si specifica per la prima entità una cardinalità minima pari a uno e una cardinalità massima pari a cinque. Quindi, un impiegato può partecipare a un minimo di una occorrenza e a un massimo di cinque occorrenze della relazione “Assegnamento”.

**N.B.** Specificando **zero come cardinalità minima**, si impone che un'occorrenza può apparire oppure no.

È possibile **assegnare un qualunque valore intero non negativo a una cardinalità di una relazione con l'unico vincolo che la cardinalità minima deve essere minore o uguale della cardinalità massima**.

Tuttavia, nella maggior parte dei casi, è sufficiente utilizzare solamente tre simboli: 0, 1 e  $N$  (molti).

#### ★ Cardinalità minima.

- **Zero.** La partecipazione dell'entità relativa è *opzionale*;
- **Uno.** La partecipazione dell'entità relativa è *obbligatoria*.

#### ★ Cardinalità massima.

- **Uno.** La partecipazione dell'entità relativa è come una funzione che associa a una occorrenza dell'entità una sola occorrenza (o nessuna) dell'altra entità che partecipa alla relazione;
- **N (molti).** Esiste un'associazione con un numero arbitrario di occorrenze dell'altra entità.

**Sintassi grafica.**

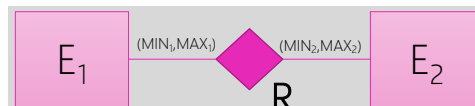


Figura 7: Sintassi grafica della cardinalità.



### 2.5.2 Cardinalità degli attributi

**Definizione.** Le **cardinalità degli attributi** è specificata per gli attributi di entità o relazione e hanno l'**obbiettivo** di **descrivere il numero minimo e massimo di valori dell'attributo associati a ogni occorrenza di entità o relazione.**

Solitamente, il valore di cardinalità pari  $(1, 1)$ , ma si possono avere vari casi:

- $(1, 1) \rightarrow$  L'attributo rappresenta una funzione che associa ad ogni occorrenza di entità un solo valore dell'attributo. Solitamente viene omesso e, come si vede nell'immagine 8, la "Persona" ha uno e un solo "Cognome";
- $(0, 1) \rightarrow$  L'attributo con cardinalità minima pari a zero vuol dire che è **opzionale** e la cardinalità massima pari a uno indica che nel **caso in cui esista**, questo valore è **unico**. Nell'esempio in figura 8, la persona può avere solo un numero di patente, ma potrebbe anche non avercela;
- $(1, N) \rightarrow$  L'attributo deve esistere, ma contiene più valori, quindi si dice che è **multivalore**;
- $(0, N) \rightarrow$  L'attributo è opzionale, ma se esiste può essere multivalore.

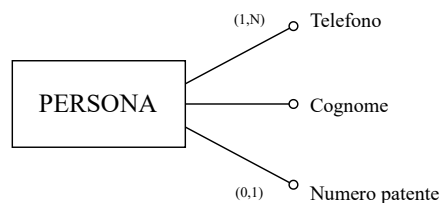


Figura 8: Esempio di cardinalità degli attributi.

### 2.5.3 Identificatori

**Definizione.** Vengono specificati per ciascuna entità di uno schema e descrivono i concetti (attributi e/o entità) dello schema che permettono di identificare in maniera univoca le occorrenze delle entità.

È assolutamente **vietato inserire uno o più identificatori all'interno di una relazione**. Quindi, quest'ultima non può avere identificatori interni!

**Per esempio**, un identificato interno per l'entità "Automobile" con attributi "Modello, Targa e Colore" è l'attributo "Targa", in quanto non possono esistere due automobili con la stessa targa e quindi due occorrenze dell'entità "Automobile" con gli stessi valori sull'attributo "Targa".

Un'entità  $E$  può essere identificata da altre entità solo se tali entità sono coinvolte in una relazione a cui  $E$  partecipa con cardinalità  $(1, 1)$ . Nei casi in cui l'identificazione di un'entità è ottenuta utilizzando altre entità si parla di **identificatore esterno**.

Per comprendere meglio si espone un **esempio**. Per identificare univocamente uno studente serve, oltre al numero di matricola, anche la relativa università. Quindi, un identificatore corretto per l'entità "Studente" in questo schema è costituito dall'attributo "Matricola" e dall'entità "Università".

Quindi, in generale:

- Un identificatore può **coinvolgere uno o più attributi**, ognuno dei quali deve avere cardinalità  $(1, 1)$ ;
- Un identificatore esterno può **coinvolgere una o più entità**, ognuna delle quali deve essere membro di una relazione alla quale l'entità da identificare partecipa con cardinalità  $(1, 1)$ ;
- Un identificatore esterno può **coinvolgere un'entità che è a sua volta identificata esternamente**, purché non vengano generati, in questa maniera, cicli di identificazione esterna;
- Ogni **entità deve avere almeno un identificatore** (interno o esterno), ma ne può avere in generale più di uno.

### Sintassi grafica.

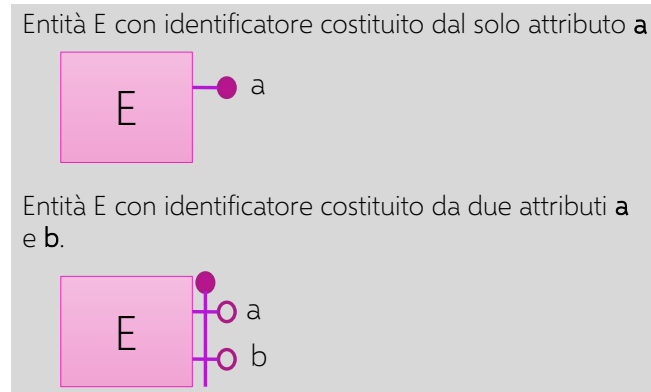


Figura 9: Sintassi grafica dell'identificatore.

#### 2.5.4 Generalizzazioni

**Definizione.** Sono i legami logici tra un'entità  $E$ , chiamata **entità genitore**, e una o più entità  $E_1, \dots, E_n$ , dette **entità figlie**, di cui  $E$  è più generale, nel senso che le comprende come caso particolare. Quindi, si dice che  $E$  è **generalizzazione** di  $E_1, \dots, E_n$  e che le entità  $E_1, \dots, E_n$  sono **specializzazioni** dell'entità  $E$ .

**Per esempio**, l'entità "Persona" è una generalizzazione delle entità "Uomo e Donna". Invece, "Professionista" è una generalizzazione delle entità "Ingegnere, Medico e Avvocato".

#### Proprietà.

- **Ogni occorrenza di un'entità figlia è anche un'occorrenza dell'entità genitore.** Per esempio, una occorrenza dell'entità "Avvocato" è anche una occorrenza dell'entità "Professionista".
- **Ogni proprietà dell'entità genitore** (come attributi, identificatori, relazioni e altre generalizzazioni) **è anche una proprietà delle entità figlie.** Per esempio, se l'entità "Persona" ha attributi "Cognome ed Età", anche le entità "Uomo" e "Donna" possiedono questi attributi.

**Classificazioni.** Le generalizzazioni possono essere classificate:

- **Totale.** Ogni occorrenza dell'entità genitore è una occorrenza di almeno una dell'entità figlie. Se non è così, la generalizzazione è **parziale**;
- **Esclusiva.** Ogni occorrenza dell'entità genitore è al massimo un'occorrenza di una delle entità figlie. Se non è così, la generalizzazione è **sovrapposta**.

In generale, una stessa entità può essere coinvolta in più generalizzazioni diverse. Possono esserci **generalizzazioni su più livelli**: in questo caso si parla di **gerarchia** di generalizzazioni. Infine, una **generalizzazione può avere una sola entità figlia**: in questo caso si parla di **sottoinsieme**.

**Sintassi grafica.** Non è difficile da comprendere, ma si presti attenzione a  $(x, y)$ . Esse indicano il **tipo di generalizzazione**:

- $[t, e] \rightarrow$  totale ed esclusiva;
- $[t, s] \rightarrow$  totale e sovrapposta;
- $[p, e] \rightarrow$  parziale ed esclusiva;
- $[p, s] \rightarrow$  parziale e sovrapposta.

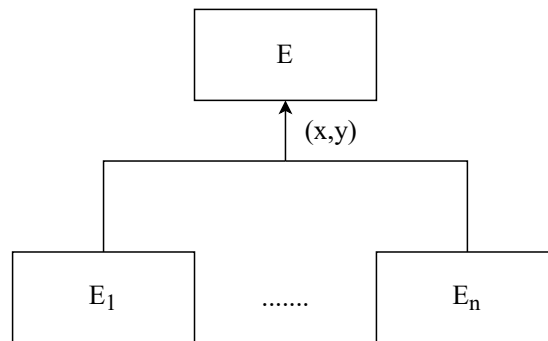


Figura 10: Sintassi grafica della generalizzazione.