

5 - Progettazione di Basi di Dati

Metodologie e modelli per il progetto

Ciclo di vita dei sistemi informatici

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.

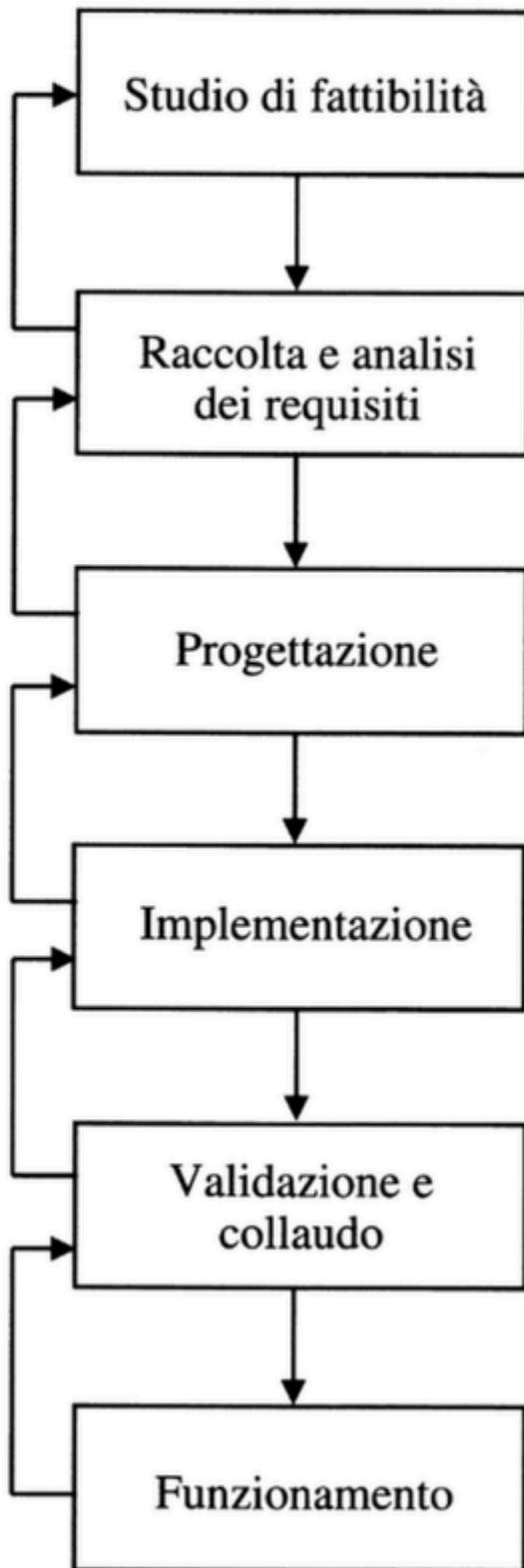
Tale ciclo di vita, generalmente, comprende:

- **Studio di fattibilità:** serve a definire se il progetto deve essere realizzato ed eventualmente i costi delle varie alternative possibili.
- **Raccolta e analisi dei requisiti:** consiste nel definire precisamente il problema da risolvere e quali proprietà e funzionalità il sistema deve avere. Occorre specificare l'ambiente di realizzazione sia hardware che software.
- **Progettazione:** in questa fase si concepisce la soluzione.

Si divide in:

- Progettazione dei dati: si individua la struttura e l'organizzazione che i dati dovranno avere
- Progettazione delle applicazioni: si definiscono le caratteristiche dei programmi applicativi
- **Implementazione:** consiste nella realizzazione del sistema informatico secondo la struttura e le caratteristiche definite nella fase di progettazione
- **Validazione e collaudo:** serve a verificare il corretto funzionamento e la qualità del sistema informativo. Il collaudo è condotto mediante prove su dati di test
- **Funzionamento:** in questa fase il sistema informativo diventa operativo ed esegue i compiti per i quali era stato originariamente progettato. Questa fase prevede anche

attività di gestione e manutenzione



Le fasi appena descritte non sono sequenziali, spesso durante l'esecuzione di una delle attività citate, bisogna rivedere decisioni prese nell'attività precedente, ottenendo un ciclo di operazioni. Le due fasi che saranno principalmente trattate durante questo corso sono la **fase di raccolta ed analisi dei requisiti** e la **fase di progettazione della base di dati**.

Metodologie di progettazione

La fase di progettazione di una base di dati è un'attività tanto complessa e delicata da essere considerata la più critica dell'intero ciclo, per questo motivo richiede l'applicazione di una vera e propria metodologia di progettazione e quest'ultima consiste in:

- Una **decomposizione** dell'intera attività di progetto in fasi successive indipendenti, con input e prodotti
- Una serie di **strategie** da seguire nei vari passi e alcuni criteri per la scelta in caso di alternative
- L'utilizzo di **modelli di riferimento** per descrivere i dati di ingresso e uscita delle varie fasi.

Le proprietà che una metodologia deve garantire sono principalmente:

- La **generalità** rispetto alle applicazioni e ai sistemi in gioco e quindi la possibilità di utilizzo indipendentemente dal problema e dagli strumenti a disposizione
- La **qualità** del prodotto in termini di correttezza, completezza ed efficienza
- La **facilità d'uso** sia delle strategie che dei modelli di riferimento.

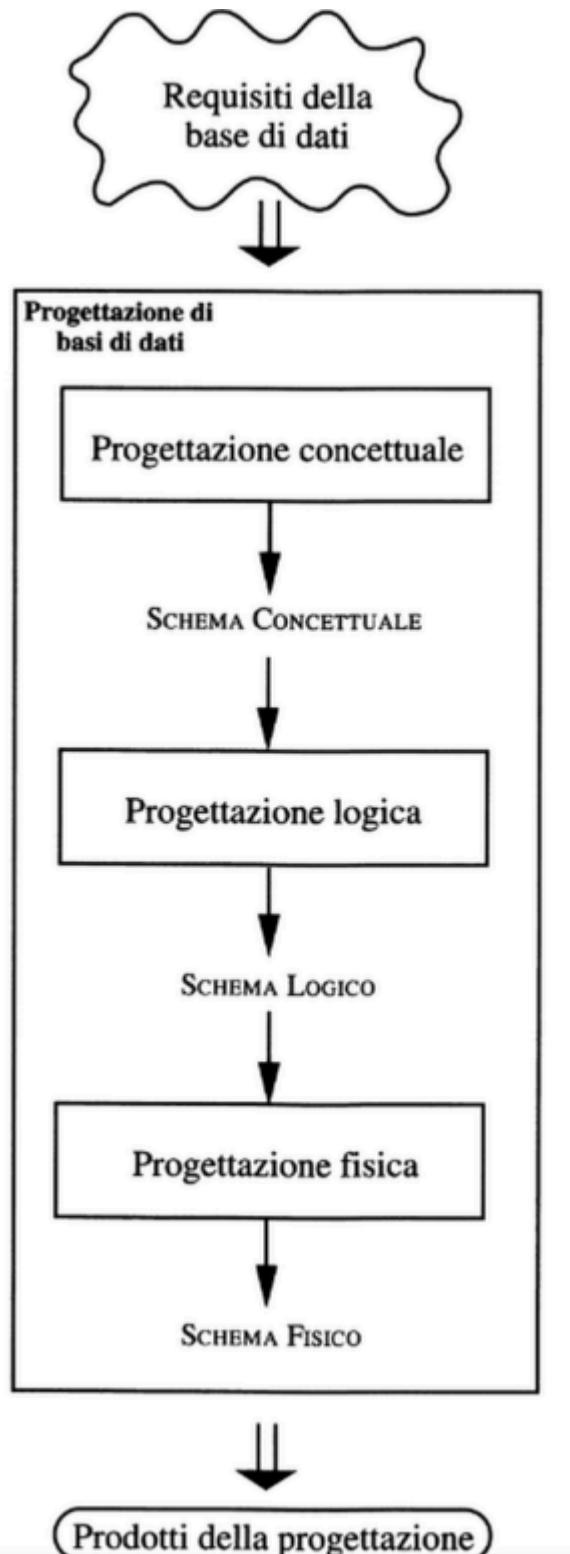
Nell'ambito delle basi di dati, si è consolidata negli anni una metodologia di progetto che ha dato prova di soddisfare pienamente le proprietà descritte.

Questa metodologia è costituita da tre fasi principali da effettuare in cascata e si fonda sul principio dell'astrazione, separando le decisioni relative a cosa rappresentare in una base di dati (prima fase) dalle decisioni relative al come farlo (seconda e terza fase):

- **Progettazione concettuale:** Traduce le specifiche dei requisiti in uno schema concettuale dei dati, dei vincoli e delle operazioni sui dati. Lo schema prodotto è descritto in modo formale e completo, inoltre fa riferimento a un modello concettuale dei dati che consente di descrivere l'organizzazione dei dati senza tener conto degli aspetti implementativi. Il modello concettuale utilizzato nell'ambito di questo corso è il **modello entità-relazione esteso**.
- **Progettazione logica:** Consiste nella traduzione dello schema concettuale in uno schema logico della base di dati che fa riferimento ad un modello logico dei dati. In questa fase le scelte progettuali si basano su criteri di ottimizzazione delle operazioni effettuabili sui dati.
La qualità dello schema logico può essere verificata con tecniche formali, ad esempio la normalizzazione per i DB basati su modello relazionale.
- **Progettazione fisica:** produce lo schema fisico che arricchisce lo schema logico con informazioni relative all'organizzazione fisica dei dati. Il modello fisico di riferimento dipende dallo specifico DBMS scelto.

Il risultato della progettazione di una base non sarà solo uno schema fisico, ma sarà costituito anche da uno schema concettuale ed uno schema logico. Lo schema concettuale fornisce infatti una rappresentazione della base di dati di alto livello che può essere utile a livello documentativo, mentre lo schema logico fornisce una descrizione concreta del

contenuto della base di dati che, prescindendo dagli aspetti implementativi, è il riferimento per le operazioni di interrogazione e aggiornamento.



Il modello Entità-Relazione

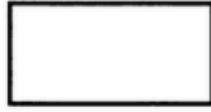
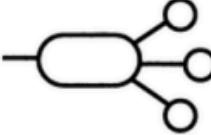
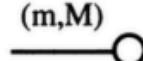
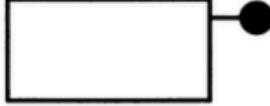
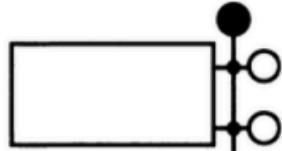
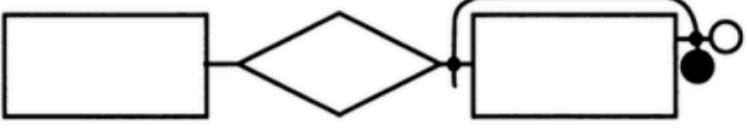
Il modello **Entità-Relazione (ER)** è un modello concettuale di dati e, come tale, fornisce una serie di strutture, dette **costrutti**, atte a descrivere la realtà di interesse.

Questi 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.

Tra i costrutti forniti dal modello E-R vi sono:

- **Entità**
- **Relazioni o Associazioni**
- **Attributi**
- **Identifieri**
- **Generalizzazioni e sottoinsiemi**

Alcune simbologie che vedremo:

| Costrutti | Rappresentazione grafica |
|--------------------------|---|
| Entità |  |
| Relazione |  |
| Attributo semplice |  |
| Attributo composto |  |
| Cardinalità di relazione |  |
| Cardinalità di attributo |  |
| Identificatore interno |   |
| Identificatore esterno |  |
| Generalizzazione |  |
| Sottoinsieme |  |

Costrutti principali del modello E-R

Entità

Rappresentano classi di oggetti che hanno proprietà comuni ed esistenza autonoma ai fini dell'applicazione di interesse (per esempio città, dipartimento, impiegato, acquisto, vendita sono esempi di entità di un'applicazione aziendale).

Una **occorrenza** di un'entità è un oggetto della classe che l'entità rappresenta, a tutti gli effetti le entità sono la descrizione intensionale (lo schema del database) del modello concettuale, le occorrenze sono la descrizione estensionale (l'istanza, l'insieme di dati).

Ogni entità ha un nome che la identifica univocamente nello schema. La scelta del nome deve soddisfare due criteri: essere espressivi ed essere al singolare

Relazioni (o associazioni)

Rappresentano legami logici, significativi per l'applicazione, tra due o più entità.

Una **occorrenza** è una n-upla (coppia nel caso di relazione binaria) costituita da occorrenze di entità, una per ciascuna delle entità coinvolte.

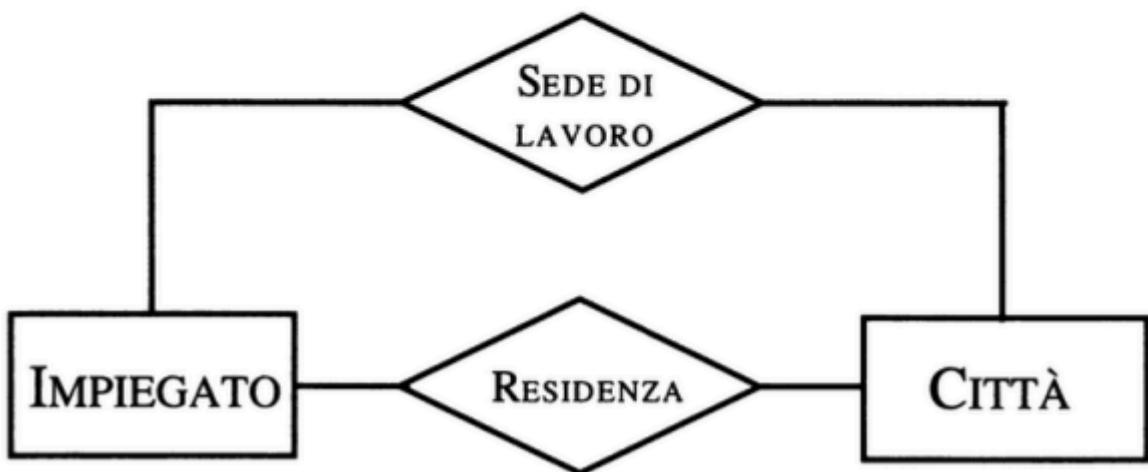
In uno schema E-R ogni relazione ha un nome che la identifica univocamente e viene rappresentata graficamente mediante un rombo con il nome della relazione all'interno e da linee che collegano la relazione con ciascuna delle sue componenti.

È importante osservare che due entità possono essere coinvolte in più relazioni e che l'insieme delle occorrenze di una relazione è a tutti gli effetti una relazione matematica tra le occorrenze delle entità coinvolte, ossia, è un sottoinsieme del loro prodotto cartesiano, implicando che tra le occorrenze di una relazione non ci possono essere ennuple ripetute

È anche possibile avere **relazioni ricorsive**, ovvero relazioni tra un entità e se stessa, e **relazioni n-arie**, cioè relazioni che coinvolgono più di due entità.

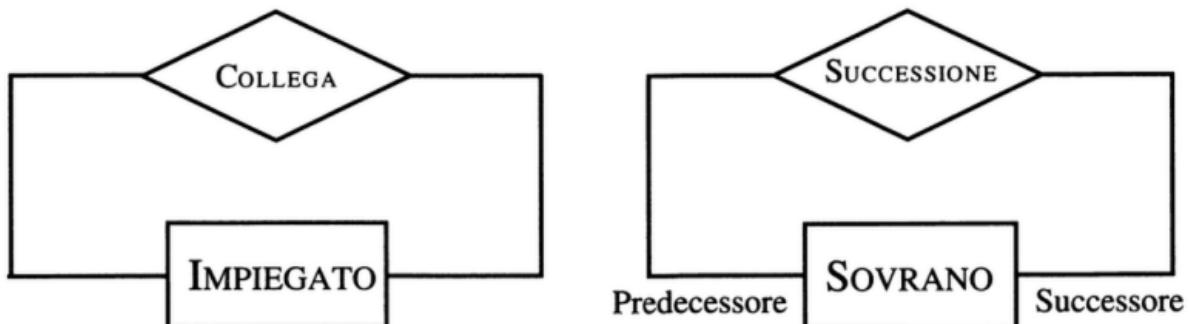
Infine, quando una relazione non è simmetrica può essere necessario distinguere i ruoli che l'entità coinvolta gioca nella relazione, ciò è ottenuto associando degli identificatori alle linee uscenti dalla relazione ricorsiva.

Esempio:

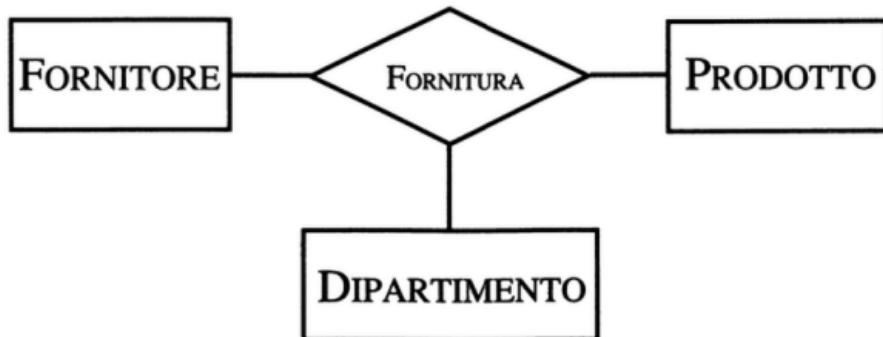


ESAME è un esempio di relazione tra le entità STUDENTE e CORSO

Esempio ricorsivo:



Esempio n-ario (ternario):



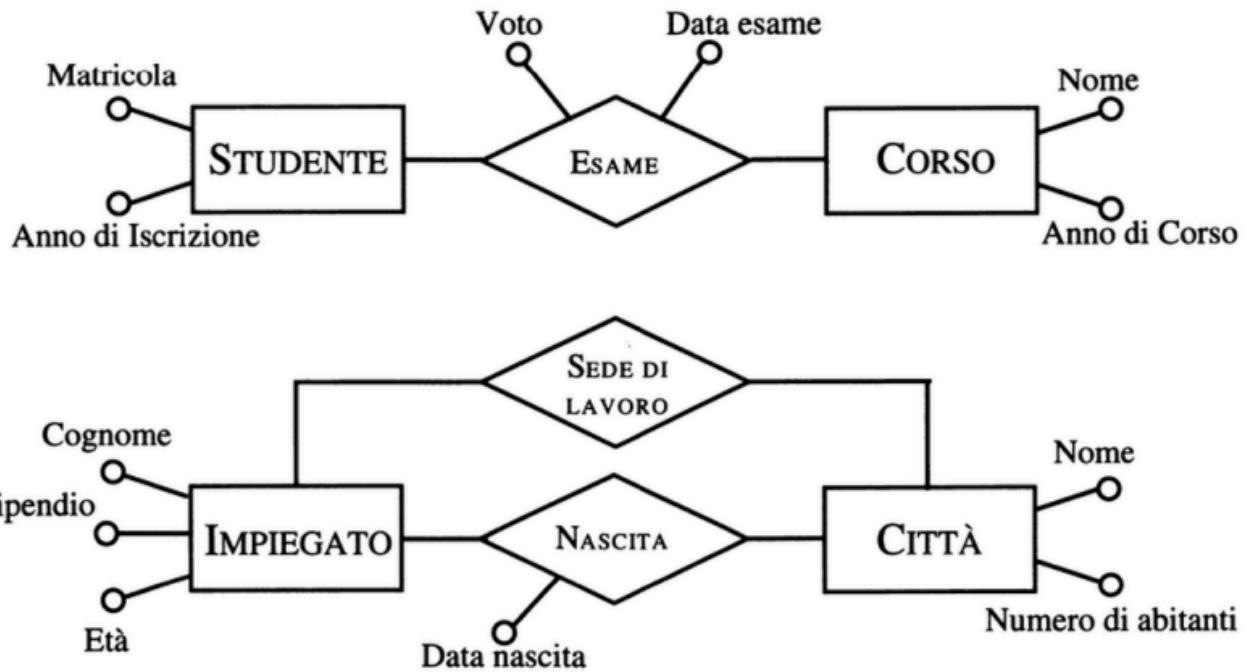
Attributi

Descrivono le proprietà elementari di entità o relazioni che sono di interesse ai fini dell'applicazione.

Un attributo associa a ciascuna occorrenza di entità (o di relazione) un valore appartenente a un dominio che contiene i valori ammissibili per l'attributo.

Gli attributi atomici possono essere raggruppati in un attributo composto.

Esempio schema E-R con relazioni, entità e attributi:



Altri costrutti del modello E-R

Cardinalità delle relazioni

Vengono specificate per ciascuna partecipazione di entità a una relazione e descrivono il numero minimo e massimo di occorrenze di relazione a cui una occorrenza dell'entità può (o deve) partecipare.



Nell'esempio in figura , ad un impiegato deve essere assegnato almeno un incarico, ma non più di cinque. Uno stesso incarico può non essere attribuito affatto, oppure può essere attribuito al più a cinquanta impiegati diversi.

Per la cardinalità minima (primo elemento nella coppia delle cardinalità) si usa:

- 0 per indicare che la partecipazione dell'entità alla relazione è **opzionale**
- 1 per indicare che la partecipazione dell'entità alla relazione è **obbligatoria**.

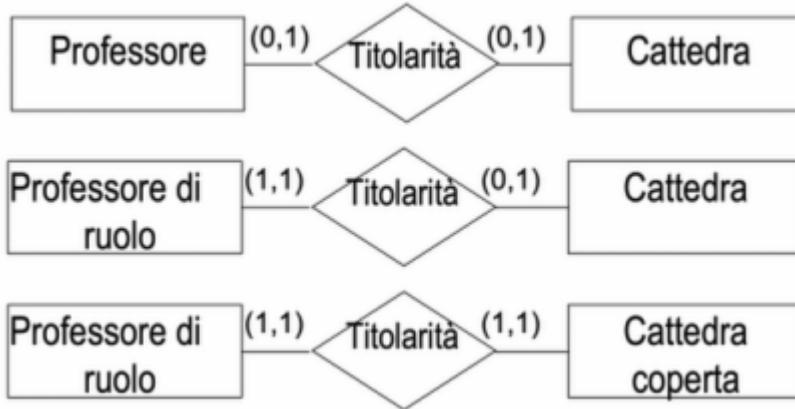
Per la cardinalità massima (secondo elemento nella coppia delle cardinalità) si usa:

- 1 per indicare che la relazione può essere espressa mediante una funzione che associa a una occorrenza dell'entità una sola occorrenza (o nessuna) dell'altra entità che partecipa alla relazione

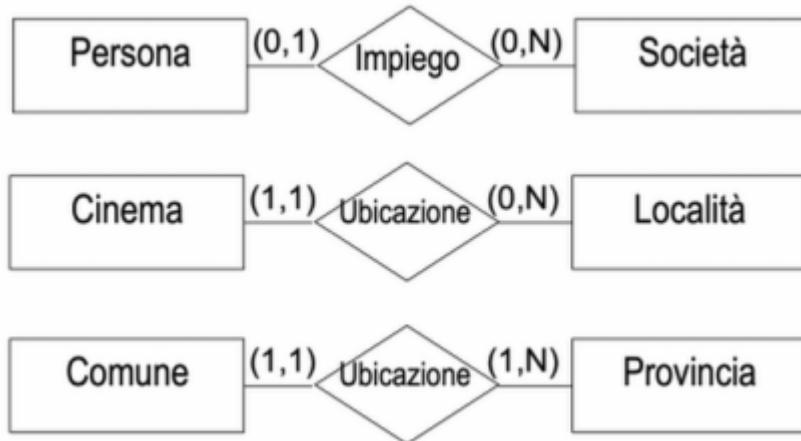
- N per indicare un'associazione con un numero arbitrario di occorrenze dell'altra entità

Con riferimento alle cardinalità massime, si classificano le relazioni binarie come:

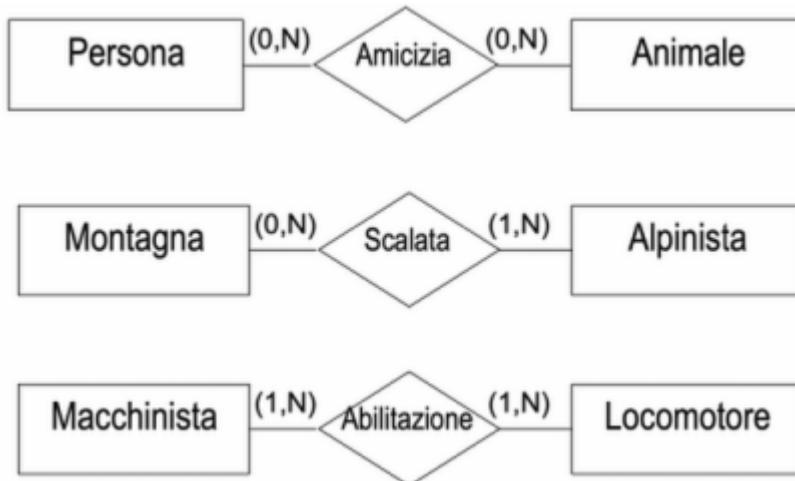
- **Uno-A-Uno**: relazioni aventi cardinalità massima pari a uno per entrambe le entità coinvolte



- **Uno-A-Molti**: relazioni aventi un'entità con cardinalità massima pari a uno e l'altra cardinalità massima pari ad N



- **Molti-A-Molti**: relazioni aventi cardinalità massima pari ad N per entrambe le entità coinvolte



Nelle relazioni n-arie, le entità coinvolte partecipano quasi sempre con cardinalità massima pari ad N . Se una entità partecipa ad una relazione n-aria con cardinalità massima pari a

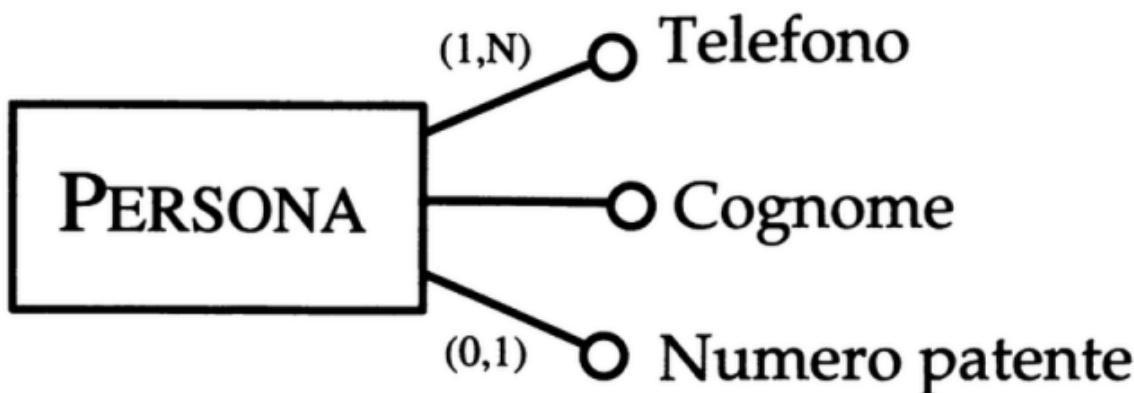
uno, significa che ogni sua occorrenza può essere legata ad un'unica entità delle altre entità coinvolte, si può allora legare direttamente tale entità con altre entità, mediante relazioni binarie di tipo uno-a-molti.

Cardinalità degli attributi

Possono essere specificate per attributi di entità o relazioni e descrivono il numero minimo e massimo di valori dell'attributo associati a ogni occorrenza di entità o relazione.

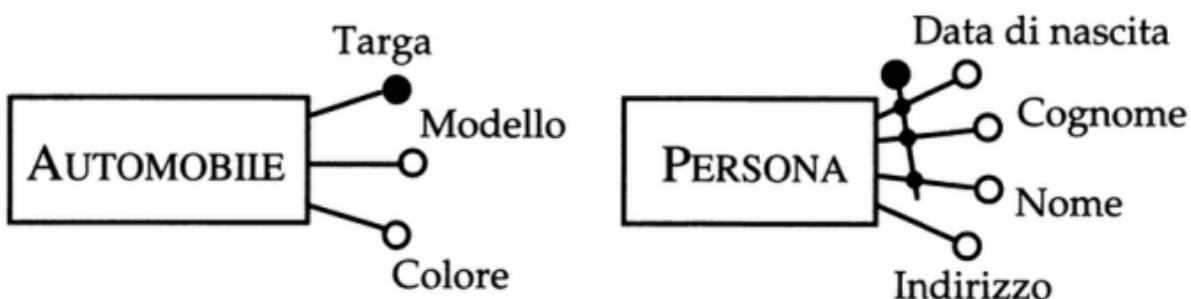
Se la cardinalità di un attributo è uno, può essere omessa e l'attributo rappresenta una funzione che associa ad ogni occorrenza di entità un solo valore dell'attributo.

Il valore di un certo attributo può essere però nullo, oppure possono esistere diversi valori di un certo attributo per una occorrenza di entità (multivalore), in quest'ultimo caso bisogna fare attenzione, poiché possono rappresentare situazioni talvolta modellabili con entità a sé, legate da relazioni uno-a-molti all'entità cui si riferiscono.



Identifieri delle entità

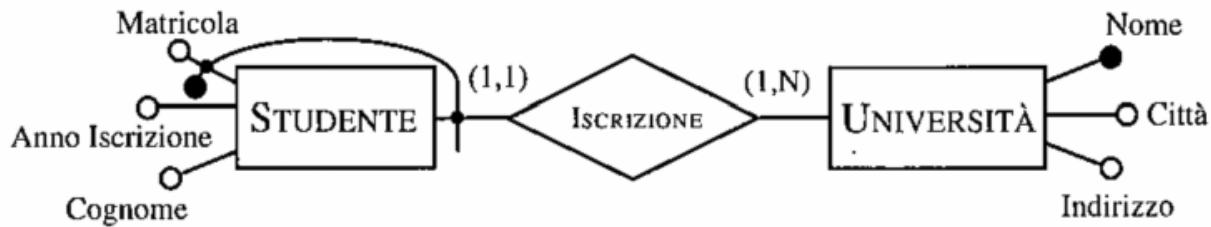
Vengono specificati per ciascuna entità di uno schema e sono lo strumento che permette di identificare univocamente una determinata occorrenza di una entità. In molti casi, uno o più attributi di una entità sono sufficienti ad individuare un identificatore, in questo caso si parla di **identificatore interno (o chiave)**.



Un identificatore interno può essere anche un insieme di attributi, come si vede nello schema PERSONA

Alcune volte però gli attributi di un'entità non sono sufficienti a identificare univocamente le sue occorrenze.

Guardiamo un esempio:



Si consideri per esempio l'entità STUDENTE nello schema in figura, l'attributo Matricola non può essere un identificatore interno poiché lo schema descrive studenti iscritti a varie università e due studenti iscritti a università diverse possono avere lo stesso numero di matricola.

In questo caso 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.

Va osservato che questa identificazione è resa possibile alla relazione uno a molti tra le entità UNIVERSITÀ e STUDENTE, che associa a ogni studente una e una sola università. Se questa relazione non esistesse, l'identificazione univoca attraverso un'altra entità non sarebbe possibile.

In generale 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), in casi come questi, ovvero nei casi in cui l'identificazione è ottenuta utilizzando altre entità si parla di **identificatore esterno**.

Generalizzazioni

Rappresentano legami logici tra un'entità E (detta **entità genitore**) e una o più entità $E_1 \dots E_n$ (dette **entità figlie**), di cui E è la più generale, nel senso che le comprende come caso particolare.

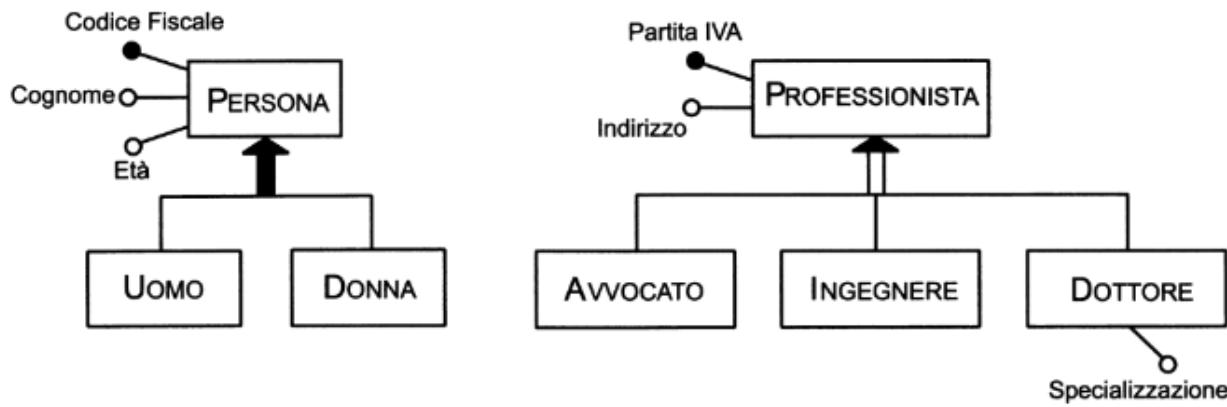
Le entità figlie $E_1 \dots E_n$ quindi sono **specializzazioni** (rispetto all'entità genitore), mentre quella genitore vi è una **generalizzazione** (rispetto all'entità dei figli).

Tra le entità coinvolte in una generalizzazione valgono le seguenti proprietà:

- Ogni occorrenza di un entità figlia è anche un occorrenza dell'entità genitore
- Ogni proprietà dell'entità genitore viene ereditata da tutte le entità figlie

Le generalizzazioni vengono rappresentate graficamente mediante delle frecce che congiungono le entità figlie con l'entità genitore.

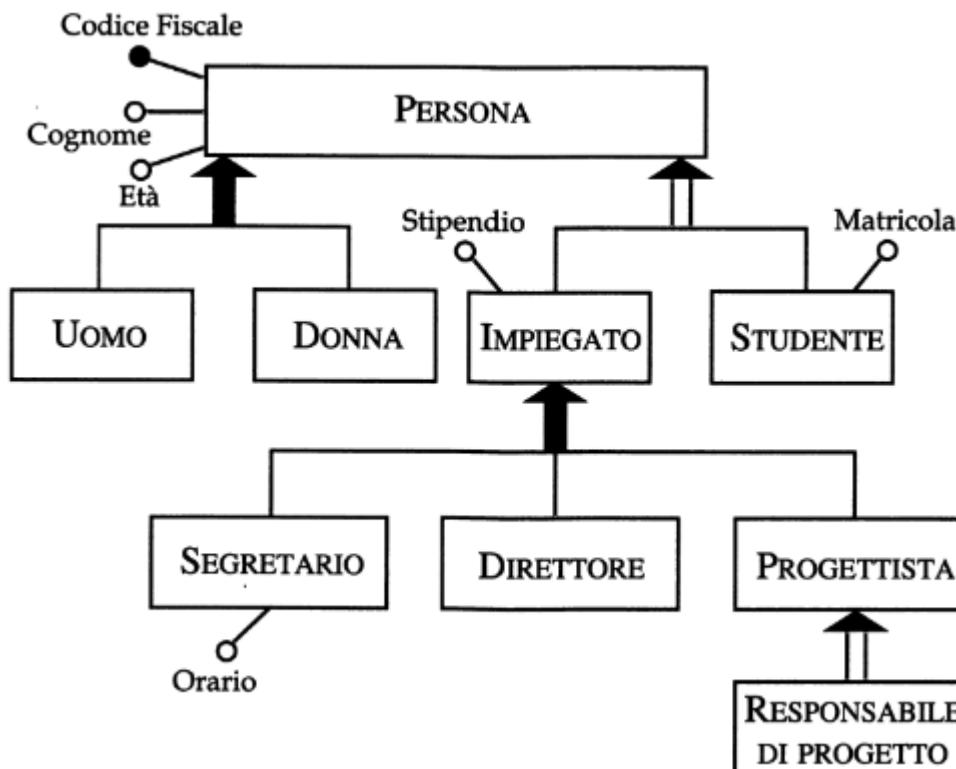
Per le entità figlie le proprietà eritate non vanno rappresentate esplicitamente.



Le generalizzazioni possono essere classificate sulla base di due proprietà tra loro ortogonali:

- Una generalizzazione è **totale** se ogni occorrenza dell'entità genitore è una occorrenza di almeno una delle entità figlie, altrimenti è **parziale**.
La generalizzazione totale si rappresenta graficamente con una freccia chiusa, mentre la generalizzazione parziale si rappresenta con una freccia aperta.
- Una generalizzazione è **esclusiva** se ogni occorrenza dell'entità genitore è al più un'occorrenza di una delle entità figlie, altrimenti è **sovraposta**.

Guardiamo questo schema:



La generalizzazione tra PERSONA, UOMO e DONNA in figura è totale, poiché gli uomini e le donne costituiscono tutte le persone (una persona deve essere o un uomo o una donna), e esclusiva, poiché una persona o è uomo o è donna (una persona non può essere sia uomo che donna).

La generalizzazione tra l'entità PROFESSIONISTA e le entità INGEGNERE e DOTTORI è invece parziale ed esclusiva, perché assumiamo che ciascun professionista abbia una sola

professione principale e che vi siano altre professioni oltre a queste tre.

Tra l'entità PERSONA e le entità STUDENTE e LAVORATORE esiste infine una generalizzazione parziale e sovrapposta, perché esistono studenti che sono anche lavoratori.

Quest'ultimo esempio ci suggerisce che le generalizzazioni sovrapposte possono essere facilmente trasformate in generalizzazioni esclusive aggiungendo una o più entità figlie per rappresentare i concetti che costituiscono le intersezioni delle entità che si sovrappongono nel caso degli studenti e dei lavoratori è sufficiente aggiungere l'entità STUDENTELAVORATORE per ottenere una generalizzazione esclusiva.

Una stessa entità può essere coinvolta in più generalizzazioni diverse e possono esserci inoltre generalizzazioni su più livelli, in questo caso si parla di **gerarchia di generalizzazioni**.

Nel caso in cui una generalizzazione ha una sola entità figlia si parla di **sottoinsieme**.

È necessario tener conto del fatto che le generalizzazioni non possiedono nomi, quindi per identificarle assumiamo che siano numerate.

Esistono infine altri vincoli sull'uso dei costrutti che non si possono esprimere sullo schema, per esempio il fatto che le gerarchie di generalizzazione non possono contenere cicli, oppure il fatto che una cardinalità minima non può essere maggiore della corrispondente cardinalità massima.

Documentazione di schemi E-R

Uno schema E-R non è quasi mai sufficiente, da solo, a rappresentare nel dettaglio tutti gli aspetti di un'applicazione.

Nel caso di schemi particolarmente complessi può accadere di non riuscire a rappresentare in maniera comprensibile ed esaustiva i vari concetti, dunque è buona norma corredare uno schema con una documentazione di supporto utile a facilitare l'interpretazione dello schema stesso e a descrivere vincoli non esprimibili nel modello E-R.

La documentazione di supporto è costituita da un dizionario dei dati e dai vincoli di integrità sui dati.

Il dizionario dei dati è composto dalle due tabelle:

1. Entità delle tabelle, che comprende:

- Nome
- Descrizione (informale)
- Attributi (con eventuali descrizioni associate)
- Identificatori

2. Relazioni delle tabelle, che comprende:

- Descrizione
- Entità coinvolte
- Attributi

Il dizionario dei dati è utile soprattutto quando lo schema è complesso e risulta pesante aggiungere allo schema tutti gli attributi di entità e relazioni.

Le regole che descrivono i vincoli di integrità possono essere espresse sotto forma di **asserzioni**, ovvero affermazioni che devono sempre essere verificate nella base di dati che si sta progettando.

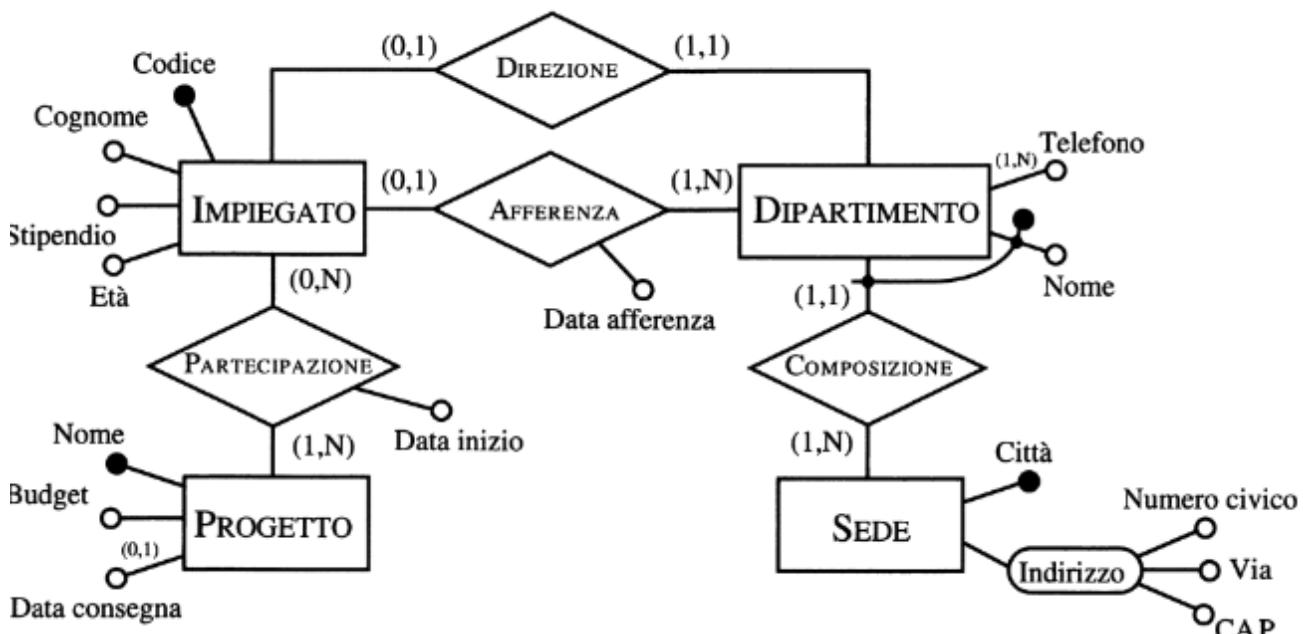
Le asserzioni vanno enunciate in maniera dichiarativa, in una forma quindi che non suggerisca un metodo per soddisfarle, per esempio notazioni del tipo "se allora " non sono adatte ad esprimere regole aziendali, quando queste documentano uno schema E-R.

Una struttura per enunciare regole aziendali sotto forma di asserzioni potrebbe essere invece la seguente:

<conetto> deve/non deve <espressione sui concetti>

dove i concetti citati corrispondono a concetti che compaiono nello schema E-R a cui si fa riferimento.

Prendiamo in esempio questo schema:



Da questo possiamo trarre il dizionario dei dati e i vincoli di integrità dei dati:

| Entità | Descrizione | Attributi | Identificatore |
|--------------|--|--------------------------------------|----------------|
| Impiegato | Impiegato che lavora nell'azienda. | Codice, Cognome, Stipendio, Età | Codice |
| Progetto | Progetti aziendali sui quali lavorano gli impiegati. | Nome, Budget, Data consegna | Nome |
| Dipartimento | Dipartimenti delle sedi dell'azienda. | Telefono, Nome | Nome, Sede |
| Sede | Sede dell'azienda in una certa città. | Città, Indirizzo (Numero, Via e CAP) | Città |

| Relazione | Descrizione | Entità Coinvolte | Attributi |
|----------------|---|-------------------------------------|----------------|
| Direzione | Associa un dipartimento al suo direttore. | Impiegato (0,1), Dipartimento (1,1) | |
| Afferenza | Associa un impiegato al suo dipartimento. | Impiegato (0,1), Dipartimento (1,N) | Data afferenza |
| Partecipazione | Associa agli impiegati i progetti sui quali lavorano. | Impiegato (0,N), Progetto (1,N) | Data inizio |
| Composizione | Associa una sede ai dipartimenti di cui è composta. | Dipartimento (1,1), Sede (1,N) | |

| Regole di vincolo |
|---|
| (RV1) Il direttore di un dipartimento deve afferire a tale dipartimento. |
| (RV2) Un impiegato non deve avere uno stipendio maggiore del direttore del dipartimento al quale afferisce. |
| (RV3) Un dipartimento con sede a Roma deve essere diretto da un impiegato con più di dieci anni di anzianità. |
| (RV4) Un impiegato che non afferisce a nessun dipartimento non deve partecipare a nessun progetto. |
| Regole di derivazione |
| (RD1) Il budget di un progetto si ottiene moltiplicando per 3 la somma degli stipendi degli impiegati che vi partecipano. |

Progettazione concettuale

La raccolta e l'analisi dei requisiti

Per **raccolta dei requisiti** si intende la completa individuazione sia dei problemi che il sistema da realizzare deve risolvere, sia delle caratteristiche che tale sistema dovrà assumere.

Per caratteristiche di sistema si intendono sia gli aspetti statici (dati) sia gli aspetti dinamici (operazioni sui suddetti dati).

I requisiti di un sistema provengono da fonti diverse, in genere sono:

- **Gli utenti**, attraverso interviste oppure documentazione scritta
- **La documentazione esistente**, come moduli, regolamenti interni, procedure aziendali, normative etc...
- **Eventuali realizzazioni pre-esistenti**, ovvero applicazione che si devono rimpiazzare o che devono interagire in qualche maniera con il sistema da realizzare.

I requisiti di solito vengono raccolti in linguaggio naturale e, per questo motivo, spesso ambigue e disorganizzate, quindi successivamente alla raccolta arriva la fase dell'**analisi dei requisiti**, la quale prevede il chiarimento e l'organizzazione delle specifiche dei dati.

Generalmente, si possono stabilire le seguenti regole di analisi dei requisiti:

- **Scegliere il corretto livello di astrazione**, per evitare di scegliere termini troppo astratti o troppo specifici.
- **Standardizzare la struttura delle frasi**, usando sempre lo stesso stile semantico (a costo di essere ripetitivi).
- **Evitare frasi contorte**, ossia avere delle definizioni semplici e chiare
- **Individuare omonimi e sinonimi ed unificare i termini**
- **Rendere esplicito il riferimento fra termini**, può succedere infatti che l'assenza di un contesto renda alcuni concetti ambigui.
- **Costruire un glossario dei termini**, utile per la comprensione e precisazione dei termini usati

Un esempio di glossario dei termini può essere questo:

| Termine | Descrizione | Sinonimi | Collegamenti |
|--------------|--|------------|-----------------------|
| Partecipante | Partecipante ai corsi. Può essere un dipendente o un professionista. | Studente | Corso, Datore |
| Docente | Docente dei corsi. Possono essere collaboratori esterni. | Insegnante | Corso |
| Corso | Corsi offerti. Possono avere varie edizioni. | Seminario | Docente, Partecipante |
| Datore | Datori di lavoro attuali e passati dei partecipanti ai corsi. | Posto | Partecipante |

Alcune frasi di esempio per la strutturazione dei requisti possono essere tali:

Frasi di carattere generale

Si vuole realizzare una base di dati per una società che eroga corsi, di cui vogliamo rappresentare i dati dei partecipanti ai corsi e dei docenti.

Frasi relative ai partecipanti

Per i partecipanti (circa 5000), identificati da un codice, rappresentiamo il codice fiscale, il cognome, l'età, il sesso, la città di nascita, i nomi dei loro attuali datori di lavoro e di quelli precedenti (insieme alle date di inizio e fine rapporto), le edizioni dei corsi che stanno attualmente frequentando e quelli che hanno frequentato in passato, con la relativa votazione finale in decimi.

Frasi relative ai datori di lavoro

Relativamente ai datori di lavoro presenti e passati dei partecipanti, rappresentiamo il nome, l'indirizzo e il numero di telefono.

Frasi relative ai corsi

Per i corsi (circa 200), rappresentiamo il titolo e il codice, le varie edizioni con date di inizio e fine e, per ogni edizione, rappresentiamo il numero di partecipanti e il giorno della settimana, le aule e le ore dove si sono tenute le lezioni.

Frasi relative a tipi specifici di partecipanti

Per i partecipanti che sono liberi professionisti, rappresentiamo l'area di interesse e, se lo possiedono, il titolo professionale. Per i partecipanti che sono dipendenti, rappresentiamo invece il loro livello e la posizione ricoperta.

Frasi relative ai docenti

Per i docenti (circa 300), rappresentiamo il cognome, l'età, la città di nascita, tutti i numeri di telefono, il titolo del corso che insegnano, di quelli che hanno insegnato in passato e di quelli che possono insegnare. I docenti possono essere dipendenti interni della società di formazione o collaboratori esterni.

Rappresentazione concettuale dei dati

Prima di affrontare le metodologie di progetto, cerchiamo di stabilire alcune buone pratiche generali per una corretta rappresentazione concettuale dei dati.

Criteri generali di rappresentazione

Dal momento che spesso non esiste una rappresentazione univoca di un insieme di specifiche, è utile stabilire delle regole concettuali basate sul modello E-R:

- Se un concetto ha proprietà significative e/o descrive classi di oggetti con esistenza autonoma, è opportuno **rappresentarlo con una entità**.

- Se un concetto ha una struttura semplice e non possiede proprietà rilevanti associate, è opportuno rappresentarlo con un **attributo** di un altro concetto a cui si riferisce.
- Se sono state individuate due (o più) entità e nei requisiti compare un concetto che le associa, questo concetto può essere rappresentato da una **relazione**.
- Se uno o più concetti risultano essere casi particolari di un altro, è opportuno rappresentarli facendo uso di una **generalizzazione**.

I criteri appena elencati hanno validità generale, sono cioè indipendenti dalla strategia di progettazione adottata

Pattern di progetto

Riferirsi al PDF [5.2 - Pattern di progetto.pdf](#), preso dal libro e presente negli Appunti Definitivi
(Essendo pieno di esempi vi è inutile riportarlo qui)

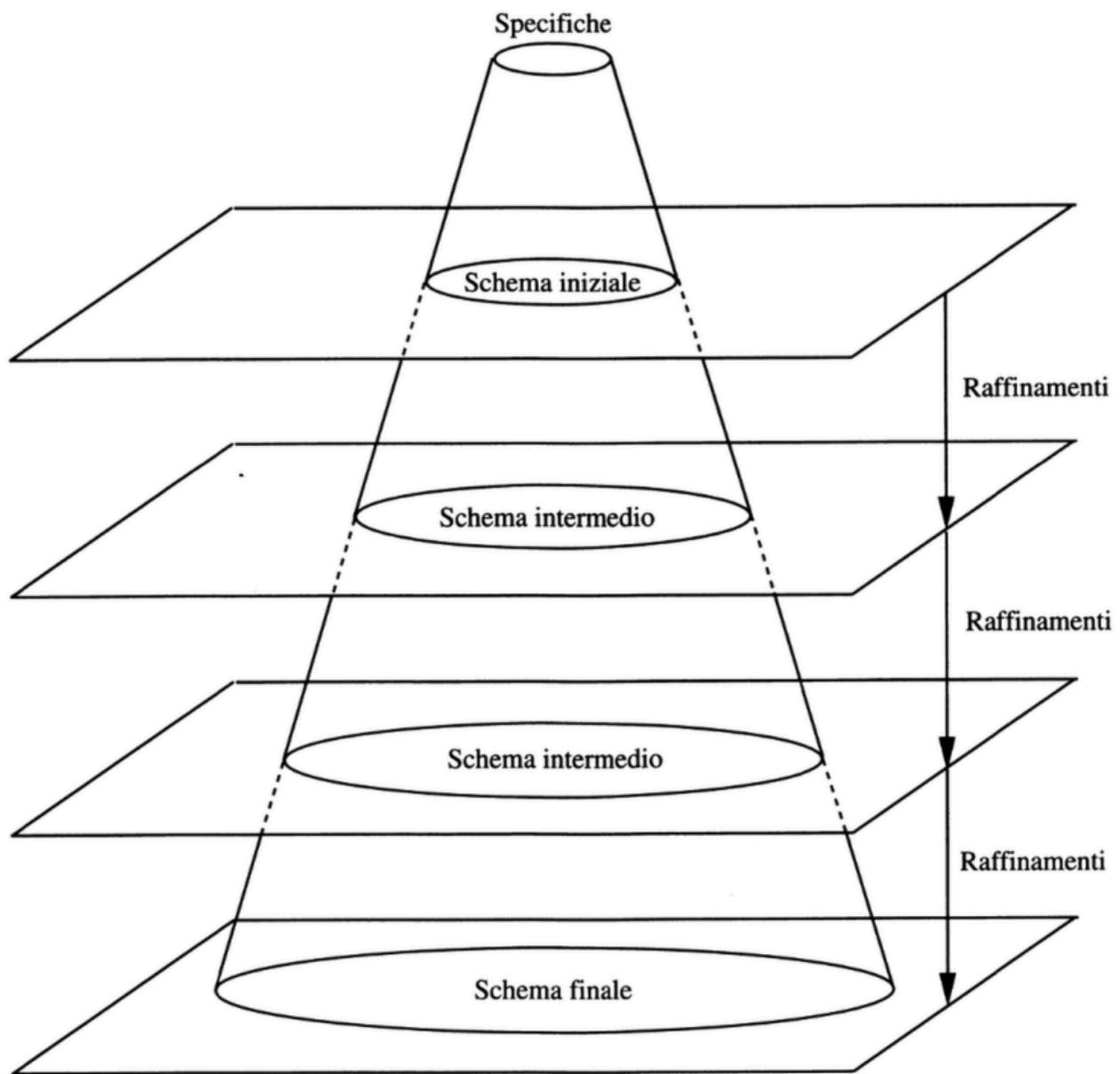
Strategie di progetto

Lo sviluppo di uno schema concettuale a partire dalle sue specifiche può essere considerato un processo di ingegnerizzazione e, come tale, è possibile utilizzare strategie viste già in altre discipline

Top-Down

Schema concettuale dove si parte da uno schema iniziale che descrive tutte le specifiche con pochi concetti molto astratto e in pian piano viene raffinato mediante opportune trasformazioni.

Nel passaggio da un raffinamento ad un altro lo schema viene modificato facendo uso di alcune trasformazioni elementari, chiamate **primitive di trasformazione top-down**.

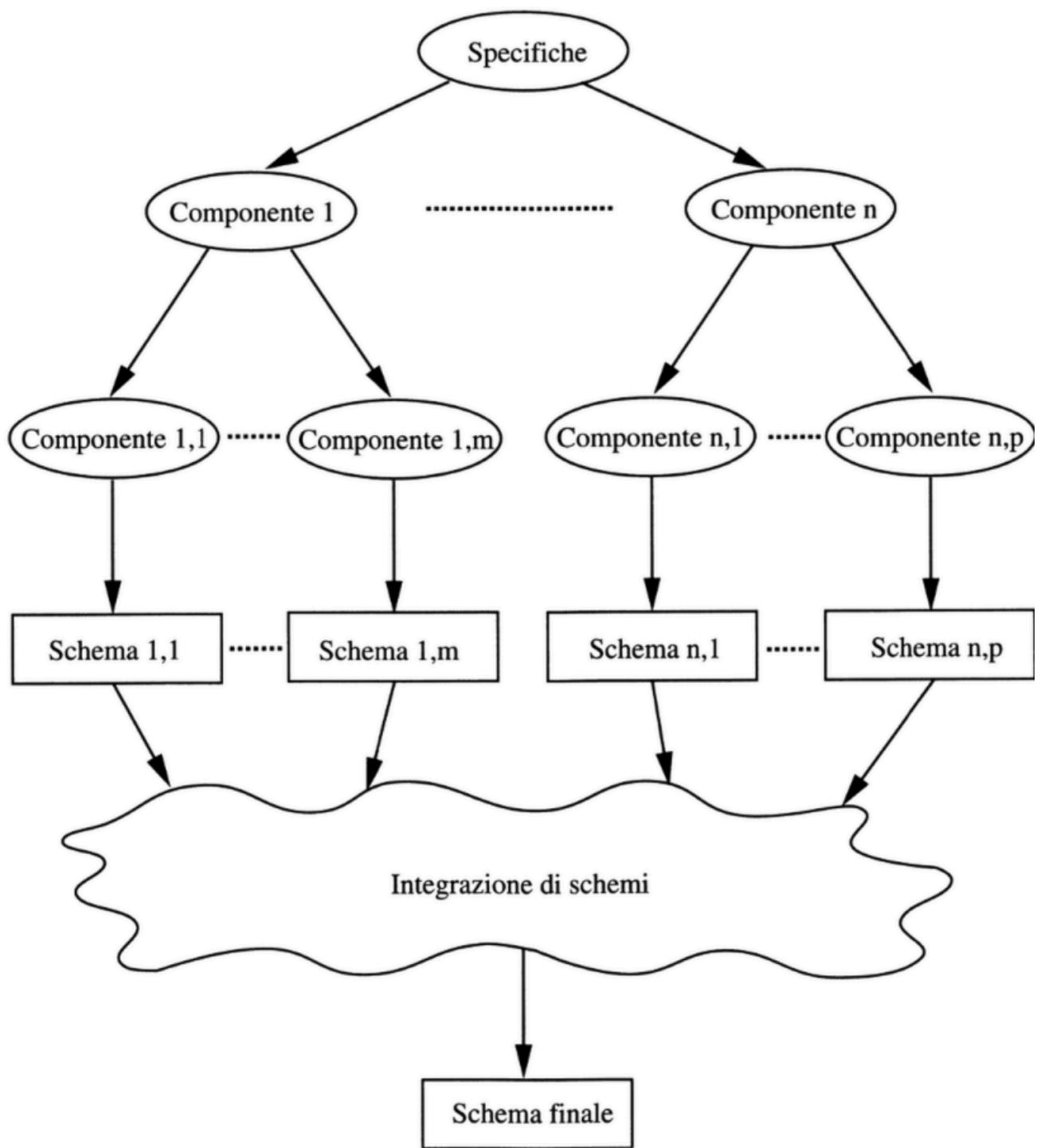


Il vantaggio di questa strategia è dalla parte del progettista, tale infatti può descrivere inizialmente tutte le specifiche dei dati trascurandone i dettagli, per poi entrare nel merito di un concetto per volta.

Tutto questo procedimento però vi è possibile solo se si ha una visione globale e astratta di tutte le componenti del sistema, molto difficile se quest'ultimo è complesso

Bottom-up

Questa strategia prevede l'inversione del metodo top-down, ossia si parte da componenti specifiche già dettagliate per poi astrarre per passi, rendendole descrittive di un frammento elementare della realtà di interesse, e infine riunirle in uno schema concettuale finale molto più generale, composto da semplici concetti.



Nello schema possiamo notare le varie fasi:

1. Decomposizione delle specifiche
2. Rappresentazione delle componenti di base
3. Integrazione in schemi elementari

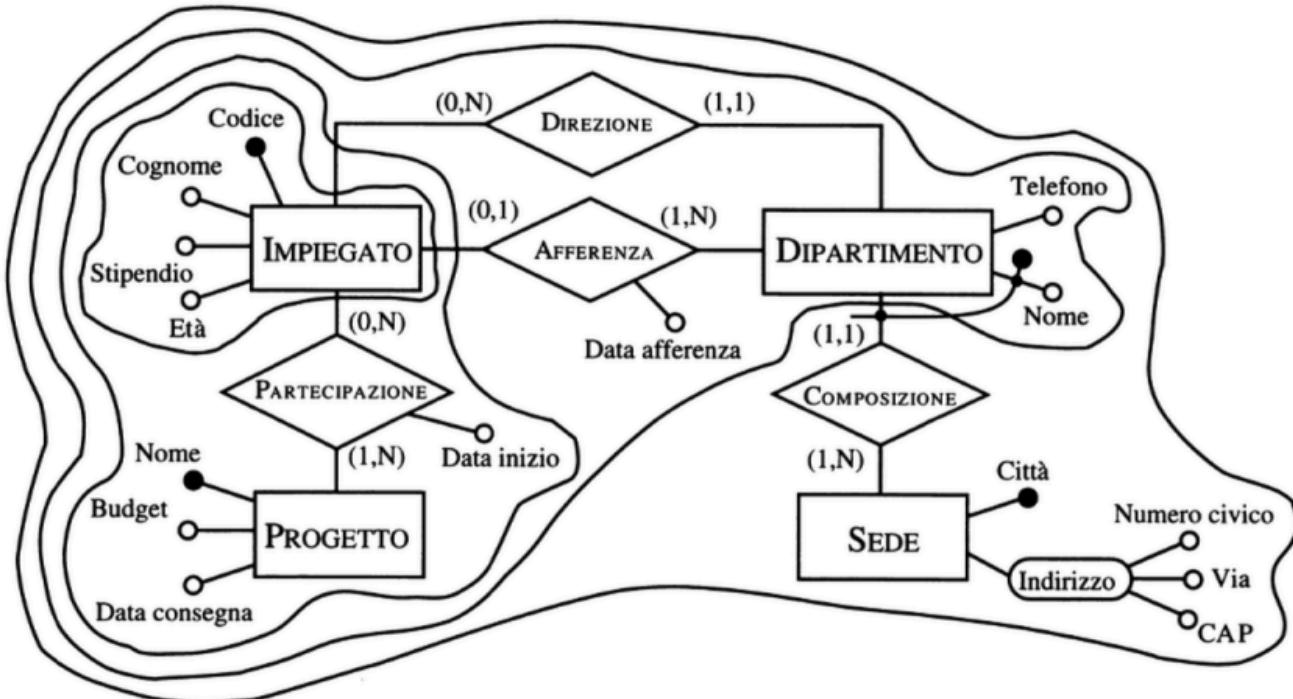
Le trasformazioni elementari che attuiamo vengono chiamate **primitive di trasformazione bottom-up**, che introducono in uno schema nuovi concetti non presenti precedentemente e in grado di descrivere aspetti della realtà di interesse non ancor stati rappresentati.

Il vantaggio di questa strategia è la stessa decomposizione di problemi in componenti più semplici e facilmente individuabili, rendendo possibile la collaborazione di progettisti diversi sullo stesso progetto. Uno svantaggio è la richiesta delle operazioni di integrazione degli

schemi concettuali diversi che, nel caso di schemi complessi, presentano quasi sempre grosse difficoltà

Inside-out

In questa strategia si rappresentano prima i concetti in relazione con i concetti iniziali per poi muoversi verso quelli più lontani usando una "navigazione" tra specifiche.



Come si può vedere dall'esempio proposto in figura, si hanno i concetti principali internamente, per poi espandersi verso l'esterno.

Questo tipo di strategia è una particolare strategia di bottom-up.

Questa strategia ha come vantaggio il non richiedere passi integrazione, a discapito di esaminare sempre tutte le specifiche di volta in volta per individuare concetti ancora non rappresentati e descriverne di nuovi nel dettaglio, rendendo quindi impossibile avere livelli di astrazione come in top-down.

Mista (o ibrida)

La strategia mista prevede la combinazione delle strategie top-down e bottom-up:

Il progettista suddivide i requisiti in componenti separate (come in bottom-up) e allo stesso tempo definisce uno **schema scheletro** contenente, a livello astratto, i concetti principali dell'applicazione. Questo schema fornisce una visione astratta dell'intero progetto, favorendo l'integrazione degli schemi sviluppati separatamente.

Per ogni concetto principale completo nello schema è possibile proseguire per raffinamenti successivi (top-down) o per concetti ancora non rappresentati (bottom-up).

Questo tipo di strategia è la più flessibile tra tutte, poiché si adatta a tutte le esigenze, per questo è molte volte l'unica ad essere adottabile in progetti di certa complessità.

Qualità di uno schema concettuale

Nella costruzione di uno schema concettuale vanno garantite alcune proprietà generali che uno schema di buona qualità deve possedere:

- **Correttezza:** Uno schema è corretto quando utilizza propriamente i costrutti messi a disposizione del modello concettuale di riferimento.
Gli errori possibili possono essere semantici o sintattici
- **Completezza:** Uno schema è completo quando rappresenta tutti i dati di interesse e quando tutte le operazioni possono essere eseguite a partire da concetti descritti nello schema.
Questa proprietà si può controllare verificando che tutte le specifiche sui dati siano rappresentate da qualche concetto presente nello schema che stiamo costruendo
- **Leggibilità:** Uno schema è leggibile quando rappresenta i requisiti in maniera naturale e facilmente comprensibile.
Per questa proprietà è necessario rendere lo schema autoesplicativo, per esempio con la scelta dei nomi da dare ai concetti.
Alcuni suggerimenti per renderlo più leggibili possono essere:
 - Riporre i costrutti su una griglia scegliendo come elementi centrali quelli con più legami
 - Tracciare solo linee perpendicolari e cercare di minimizzare le intersezioni
 - Disporre le entità che sono genitori di generalizzazioni sopra le relative entità **figlie**
- **Minimalità:** uno schema è minimale quando tutte le specifiche dei dati sono rappresentate una volta nello schema.
Quindi non vi è minimale quando ci sono ridondanze, ma quest'ultima potrebbe non essere indesiderata ma voluta per scelte progettuali desiderate

Metodologia generale

Descriviamo un metodo per la progettazione concettuale con strategia mista, da poter seguire:

- 1. Analisi dei requisiti**
 - Costruire un glossario
 - Analizzare i requisiti ed eliminare ambiguità
 - Raggruppare i requisiti in insiemi omogenei
- 2. Passo base**
 - Individuare i concetti più rilevanti e rappresentarli in uno schema scheletro
- 3. Passo di decomposizione (solo se necessario o appropriato)**
 - Effettuare una decomposizione dei requisiti con riferimento ai concetti presenti nello schema scheletro
- 4. Passo iterativo**
 - Raffinare i concetti presenti sulla base delle loro specifiche
 - Aggiungere nuovi concetti allo schema per descrivere specifiche non ancora descritte

5. Passo di integrazione

- Integrare vari sotto-schemi in uno schema generale facendo riferimento allo schema scheletro

6. Analisi di qualità

- Verificare la correttezza dello schema ed eventualmente ristrutturare lo schema
- Verificare la completezza dello schema ed eventualmente ristrutturare lo schema
- Verificare la minimalità, documentare le ridondanze ed eventualmente ristrutturare lo schema
- Verificare la leggibilità dello schema ed eventualmente ristrutturare lo schema

In questi passaggi è sempre utile per ogni passaggio la **documentazione degli schemi**

Esempi di progettazione concettuale

Vi consiglio di andare a vedere il PDF [5.1 - Esempio di progettazione concettuale.pdf](#), preso sempre dal libro e presente nella cartella degli Appunti Definitivi, insieme al PDF [5.4 - Esercitazione Analisi dei Requisiti e Progettazione Concettuale.pdf](#), preso dalla prof

Progettazione logica

L'obiettivo della progettazione logica è quello di costruire uno schema logico in grado di descrivere, in maniera corretta ed efficiente, tutte le informazioni contenute nello schema Entità-Relazione prodotto nella fase di progettazione concettuale,

Fasi della progettazione logica

Le attività principali della progettazione logica sono la riorganizzazione dello schema concettuale e la traduzione in un modello logico.

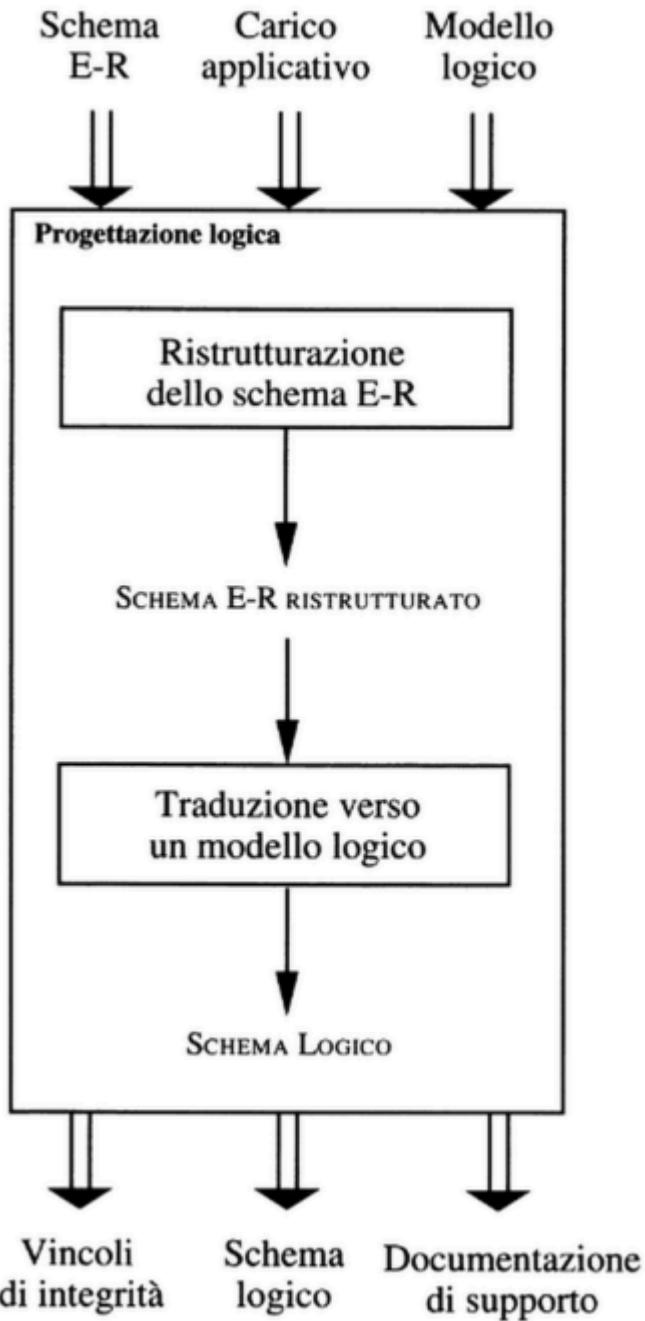
Questo processo prevede due fasi principali:

- **Ristrutturazione dello schema E-R:** fase indipendente dal modello logico scelto e si basa su criteri di ottimizzazione dello schema e di semplificazione della fase successiva.
- **Traduzione verso il modello logico:** fase che fa riferimento ad uno specifico modello logico e può includere un ulteriore ottimizzazione.

I dati di ingresso della prima fase sono lo schema concettuale prodotto nella fase precedente e il carico applicativo previsto, in termini di dimensione dei dati e caratteristiche delle operazioni. Il risultato che si ottiene è uno schema E-R che non è più strettamente concettuale, poiché tiene conto degli aspetti realizzativi. Questo schema e il modello logico scelto costituiscono i dati di input della seconda fase, che produce lo schema logico della base di dati.

Lo schema logico, i vincoli di integrità definiti su esso e la relativa documentazione,

costituiscono i prodotti finali della progettazione logica



Analisi delle prestazioni

Uno schema E-R può essere modificato per ottimizzare alcuni indici di prestazione del progetto, si parla di indici poiché non esiste una valutazione per le prestazioni della base di dati nella progettazione logica (non esistono semplicemente fattori possibili prevedibili).

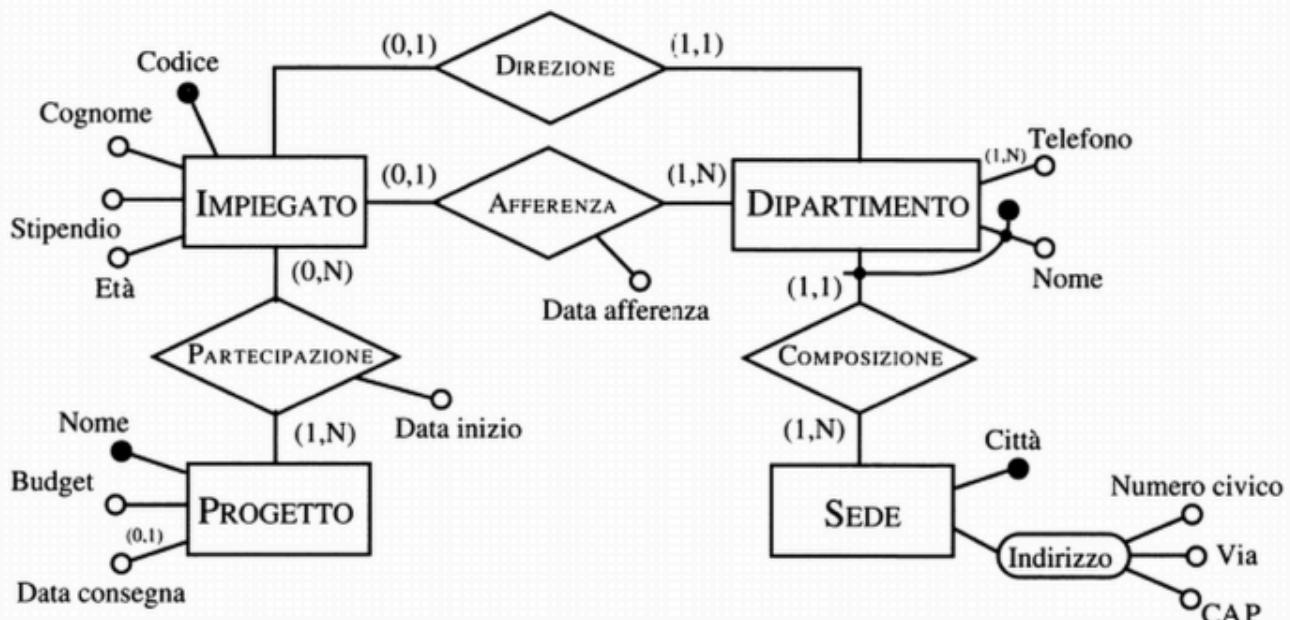
Due parametri che generalmente regolano le prestazioni di un sistema software sono:

- **Costo di un'operazione:** valutato in termini di numero medio di occorrenze di entità e associazioni da visitare per rispondere a una operazione sul DB
- **Occupazione di memoria:** valutata in termini di spazio di memoria necessario per memorizzare i dati descritti dallo schema.

Per studiare questi parametri abbiamo bisogno di conoscere, oltre allo schema, le seguenti informazioni:

- **Volume dei dati**, ossia:
 - Numero medio di occorrenze di ogni entità e associazione dello schema
 - Dimensioni di ciascun attributo (di entità o associazione)
- **Caratteristiche delle operazioni**, ossia:
 - Tipo dell'operazione (interattiva o batch)
 - Frequenza (numero medio di esecuzioni in un certo intervallo di tempo)
 - Dati coinvolti (entità e associazioni)

Per fare un esempio pratico, prendiamo uno schema:



Trattandosi di uno schema riguardante dati sul personale di un'azienda, le operazioni possibili potrebbero essere quelle che seguono:

1. Assegna un impiegato ad un progetto
2. Trova i dati di un impiegato, del dipartimento in quale lavora e dei progetti a cui partecipa
3. Trova i dati di tutti gli impiegati di un certo dipartimento
4. Per ogni sede, trova i suoi dipartimenti con il cognome del direttore e l'elenco degli impiegati del dipartimento

Sebbene un'analisi delle prestazioni che fa riferimento ad un numero ristretto di operazioni può sembrare riduttiva rispetto al reale carico della base di dati, va notato che le operazioni sulle basi di dati seguono la cosiddetta regola "ottanta-venti":

L'ottanta percento del carico è generato dal venti percento delle operazioni.

Questo ci permette di valutare adeguatamente il carico concentrando solo sulle operazioni previste

Il volume dei dati e le caratteristiche generali delle operazioni possono essere descritti facendo uso di tabelle, chiamate **tabelle dei volumi e delle operazioni**

Nella tavola dei volumi vengono riportati tutti i concetti dello schema (entità e associazioni), un simbolo per ogni concetto che distingua le entità (*E*) dalle relazioni (*R*) e il volume a

regime.

Nella tavola delle operazioni riportiamo, per ogni operazione la frequenza prevista (su un arco di tempo) e un simbolo che indichi se l'operazione è interattiva (*I*) o batch (*B*).

Nella tavola dei volumi, il numero delle occorrenze delle associazioni dipende da due parametri: il numero di occorrenze delle entità coinvolte nelle associazioni e il numero (medio) di partecipazioni di una occorrenza di entità alle occorrenze di associazioni. Il primo parametro è solitamente fornito dalla specifica dei requisiti, mentre il secondo parametro dipende a sua volta dalle cardinalità dell'associazione.

Tavola dei volumi

| Concetto | Tipo | Volume |
|----------------|------|--------|
| Sede | E | 10 |
| Dipartimento | E | 80 |
| Impiegato | E | 2000 |
| Progetto | E | 500 |
| Composizione | R | 80 |
| Afferenza | R | 1900 |
| Direzione | R | 80 |
| Partecipazione | R | 6000 |

Tavola delle operazioni

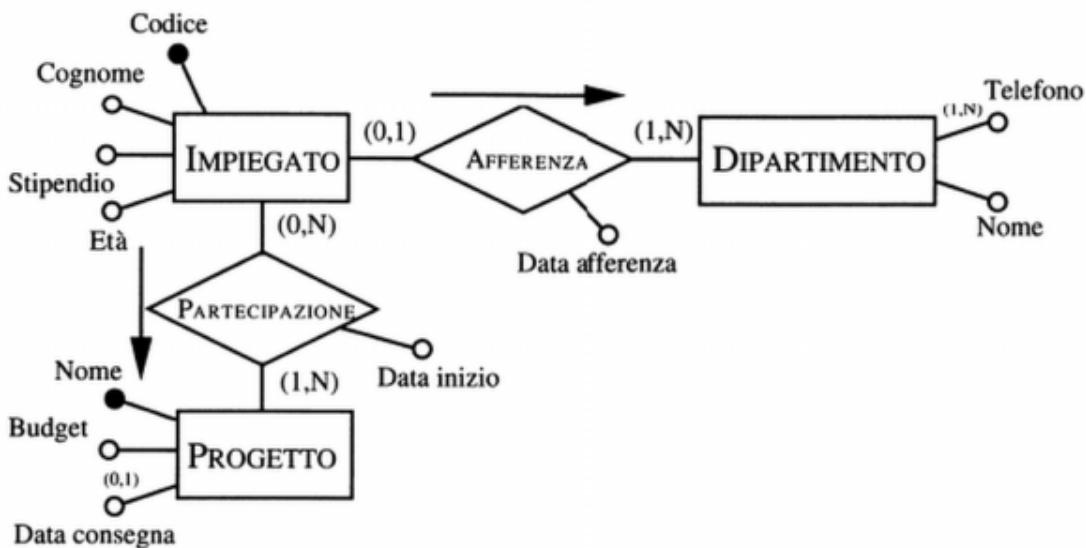
| Operazione | Tipo | Frequenza |
|------------|------|---------------|
| Op. 1 | I | 50 al giorno |
| Op. 2 | I | 100 al giorno |
| Op. 3 | I | 10 al giorno |
| Op. 4 | B | 2 a settimana |

Ad esempio, considerando lo schema E-R precedente e le sue tabelle, per calcolare il numero di occorrenze per l'associazione PARTECIPAZIONE si assume che un impiegato possa partecipare a 3 progetti contemporaneamente, ottenendo dunque $2000 \times 3 = 6000$ occorrenze.

Allo stesso modo, si sarebbe potuto assumere che ad uno stesso progetto possano partecipare 12 impiegati, ottenendo $500 \times 12 = 6000$ occorrenze della relazione partecipazione.

È assolutamente errato moltiplicare tra loro il numero di occorrenze delle due entità che partecipano all'associazione, poiché così facendo si considera che, ricorrendo all'esempio precedente, ogni impiegato possa partecipare solo ad un progetto.

Per ogni operazione, possiamo inoltre descrivere graficamente i dati coinvolti con uno schema di operazione che consiste nel frammento dello schema E-R interessato dall'operazione, sul quale viene disegnato il cammino logico da percorrere per accedere alle informazioni di interesse.



Preso in considerazione l'operazione 2, per eseguirla si deve: accedere a una occorrenza dell'entità IMPIEGATO per accedere poi ad una occorrenza dell'associazione AFFERENZA e, attraverso questa, a una occorrenza dell'entità DIPARTIMENTO. Successivamente, per conoscere i dati dei progetti ai quali lavora, dobbiamo accedere a tre occorrenze dell'associazione PARTECIPAZIONE e, attraverso queste, a tre occorrenze dell'entità progetto.

Tutto questo viene riassunto in una tavola degli accessi in cui viene anche indicato se l'accesso avviene in lettura (*L*) o scrittura (*S*). Evidenziare questa differenza è fondamentale poiché si assume che le operazioni di scrittura abbiano un costo pari al doppio di una operazione di lettura.

| Concetto | Costrutto | Accessi | Tipo |
|----------------|-----------|---------|------|
| Impiegato | Entità | 1 | L |
| Afferenza | Relazione | 1 | L |
| Dipartimento | Entità | 1 | L |
| Partecipazione | Relazione | 3 | L |
| Progetto | Entità | 3 | L |

Ristrutturazione di schemi E-R

La prima fase di ristrutturazione di uno schema Entità-Relazione si può suddividere in una serie di passi in sequenza:

- **Analisi delle ridondanze:** si decide se eliminare o mantenere eventuali ridondanze presenti nel E-R.
- **Eliminazione delle generalizzazioni:** tutte le generalizzazioni presenti nello schema sono analizzate e sostituite da altri costrutti. Nella progettazione concettuale utilizziamo costrutti assenti nella progettazione logica per rappresentare nel miglior modo possibile i dati.

- **Partizionamento/accorpamento di entità e associazioni:** si decide se partizionare concetti dello schema in più concetti o, viceversa, accorpare concetti in un unico concetto\
- **Scelta degli identificatori primari:** si seleziona un identificatore per le entità che ne hanno più di uno.



Analisi delle ridondanze

Per ridondanza, in uno schema concettuale, si intende la presenza di un dato che può essere derivato da altri. Le forme di ridondanza in uno schema E-R sono:

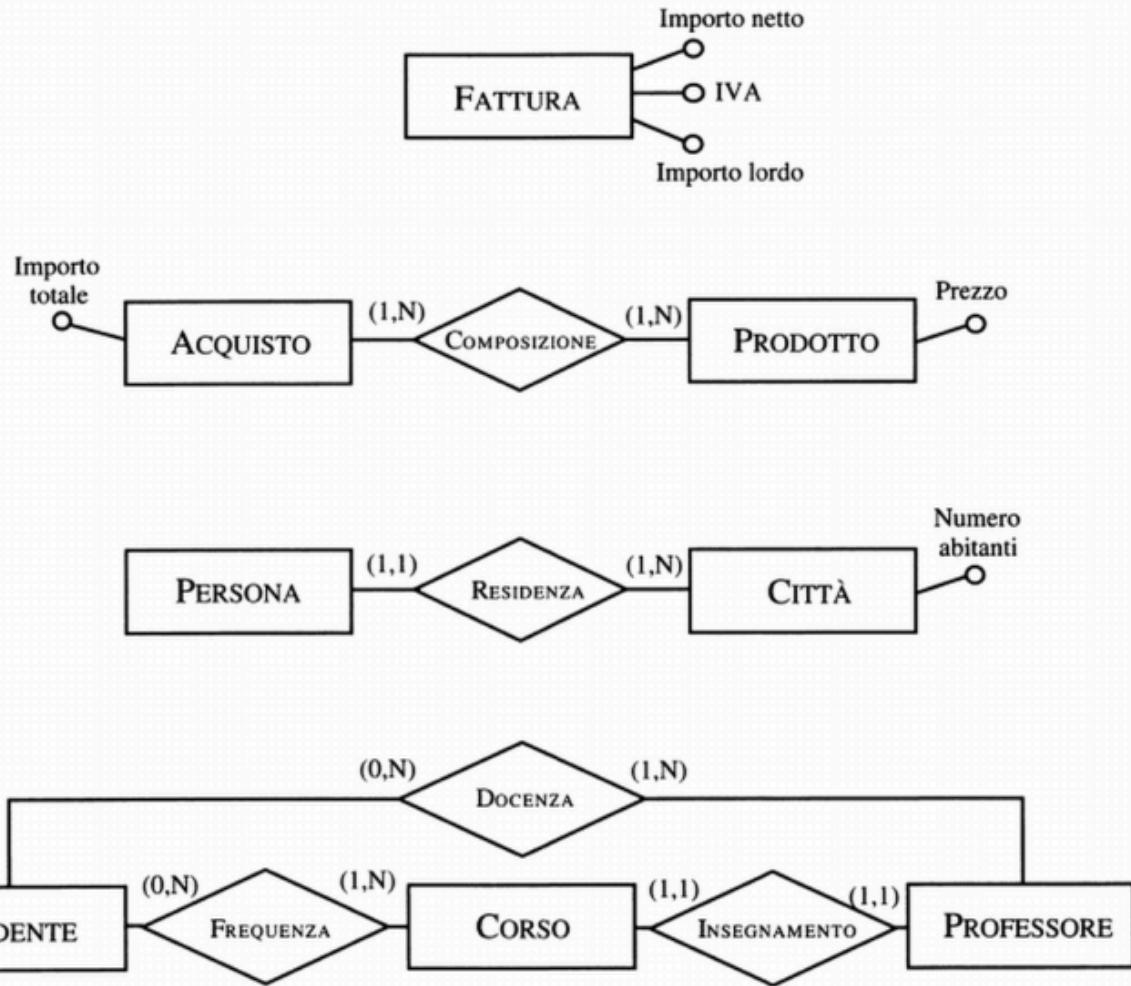
- Attributi derivabili da altri attributi della stessa entità (o associazione).
Per esempio nel primo schema in figura
- Attributi derivabili da attributi di altre entità (o associazioni), di solito attraverso funzioni di aggregazione (somma, media, conteggio...).
Se ne vedono due esempi nel secondo e terzo schema in figura. Nel secondo schema, l'attributo importo totale dell'entità acquisto si può derivare, attraverso l'associazione

composizione dall'attributo prezzo dell'entità prodotto, sommando i prezzi dei prodotti di cui un acquisto è composto

- Associazioni derivabili dalla composizione di altre associazioni in presenza di cicli.

Ad esempio nel quarto schema in figura l'associazione docenza tra studenti e professori può essere infatti derivata dalle associazioni frequenza e insegnamento. Va comunque precisato che la presenza di un ciclo non genera necessariamente ridondanze.

Nota bene: La presenza di cicli non genera necessariamente ridondanze.



La presenza di una ridondanza ha effetti positivi, semplificare le interrogazioni, ed effetti negativi, appesantisce gli aggiornamenti e comporta l'occupazione di più memoria. Per questo motivo, la decisione di mantenere o eliminare una ridondanza va presa in seguito ad una **analisi quantitativa** che confronti il costo di esecuzione delle operazioni che coinvolgono il dato ridondante e l'occupazione di memoria, sia in presenza e che in assenza di ridondanza.

Esempio di eliminazione di una ridondanza

Si consideri l'applicazione anagrafica di una regione rappresentata dal terzo schema nella figura precedente e si considerino le seguenti operazioni:

1. Memorizzare una nuova persona con la relativa città di residenza
2. Stampare tutti i dati di una città (incluso il numero di abitanti)

Supponiamo inoltre che per questa applicazione i dati di carico siano quelli riportati in figura:

Tavola dei volumi

| Concetto | Tipo | Volume |
|-----------|------|-----------|
| Città | E | 200 |
| Persona | E | 1 000 000 |
| Residenza | R | 1 000 000 |

Tavola delle operazioni

| Operazione | Tipos | Frequenza |
|------------|-------|---------------|
| Op. 1 | I | 500 al giorno |
| Op. 2 | I | 2 al giorno |

Individuato l'attributo Numero abitanti come dato ridondante, proviamo a valutare gli indici di prestazione in caso di presenza del dato ridondante.

Assumendo che il numero degli abitanti di una città richieda 4 byte, abbiamo che il dato ridondante richiede $4 \times 200 = 800$ byte di memoria aggiuntiva.

Passiamo ora alla stima del costo delle operazioni:

Per farlo si generano le tabelle degli accessi in figura, dalle quali si evince che l'operazione 1 ha un costo unitario pari a 7 in presenza di ridondanza e pari a 4 in assenza di ridondanza, l'operazione 2 ha costo unitario rispettivamente 1 e 5001.

A questo punto, tenendo conto delle frequenze, si ottiene che il sistema presenta:

1. $7 \times 500 + 1 \times 2 = 3502$ accessi totali in presenza di ridondanza
2. $4 \times 500 + 5001 \times 2 = 12002$ accessi assenti in presenza di ridondanza

Quindi, circa 8500 accessi giornalieri in più rispetto al caso di dato ridondante presente contro un risparmio di un solo kilobyte.

Possiamo dunque concludere che conviene, in questo caso, mantenere il dato ridondante.

Di seguito le tavole di accessi per caso

Tavole degli accessi in presenza di ridondanza

| Operazione 1 | | | |
|--------------|--------|------|------|
| Concetto | Costr. | Acc. | Tipo |
| Persona | E | 1 | S |
| Residenza | R | 1 | S |
| Città | E | 1 | L |
| Città | E | 1 | S |

Tavole degli accessi in assenza di ridondanza

| Operazione 1 | | | |
|--------------|--------|------|------|
| Concetto | Costr. | Acc. | Tipo |
| Persona | E | 1 | S |
| Residenza | R | 1 | S |

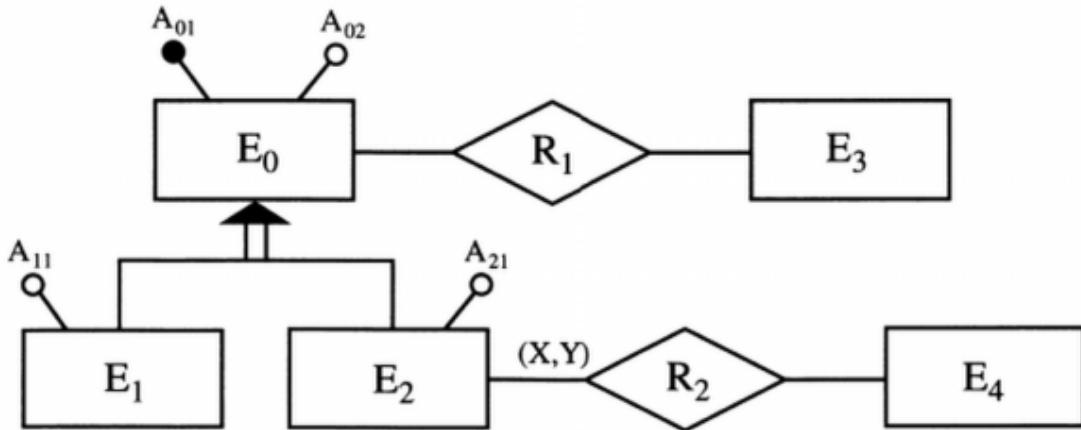
Operazione 2

| Concetto | Costr. | Acc. | Tipo |
|-----------|--------|------|------|
| Città | E | 1 | L |
| Residenza | R | 5000 | L |

Eliminazione delle generalizzazioni

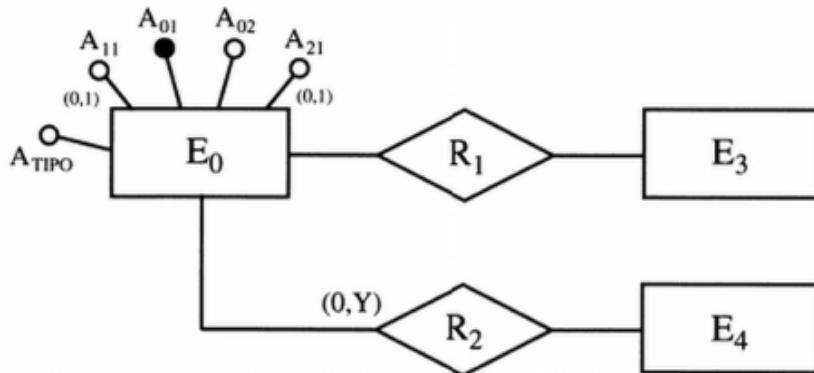
Dato che i sistemi tradizionali per la gestione dei dati non consentono di rappresentare direttamente una generalizzazione, risulta spesso necessario trasformare questo costrutto in altri costrutti del modello E-R per i quali esiste una implementazione naturale: le entità e le associazioni.

Prenderemo in esempio questo schema:



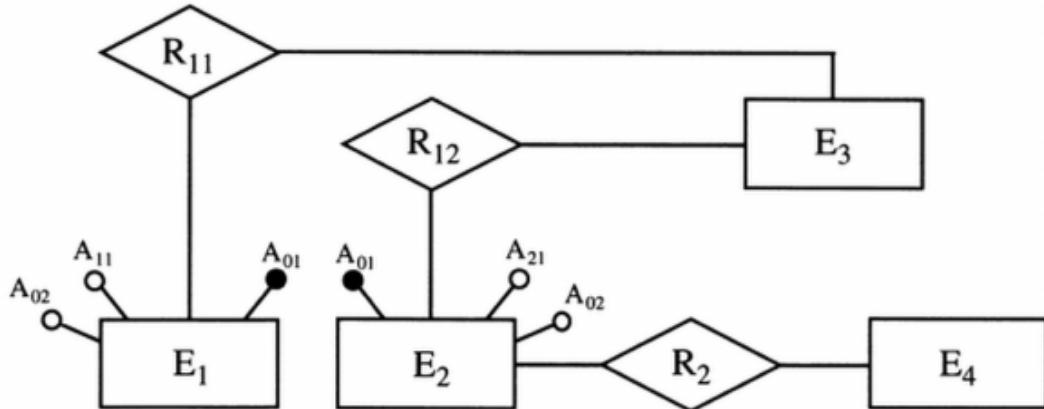
Per rappresentare una generalizzazione mediante entità e associazioni abbiamo essenzialmente tre alternative possibili:

- Accorpamento delle figlie nel genitore.** Le entità E_1 e E_2 sono eliminate e le loro proprietà (attributi, associazioni, generalizzazioni) sono aggiunte al padre E_0 . All'entità padre viene aggiunto un attributo per distinguere il tipo di occorrenza di E_0 , cioè se tale occorrenza apparteneva a E_1 o a E_2 o, nel caso di generalizzazione parziale, a nessuna delle due. In riferimento al primo schema, si osservi che gli attributi A_{11} e A_{12} possono assumere valori nulli per alcune occorrenze di E_0 e che la relazione R_2 avrà una cardinalità minima pari a 0 sull'entità E_0 .

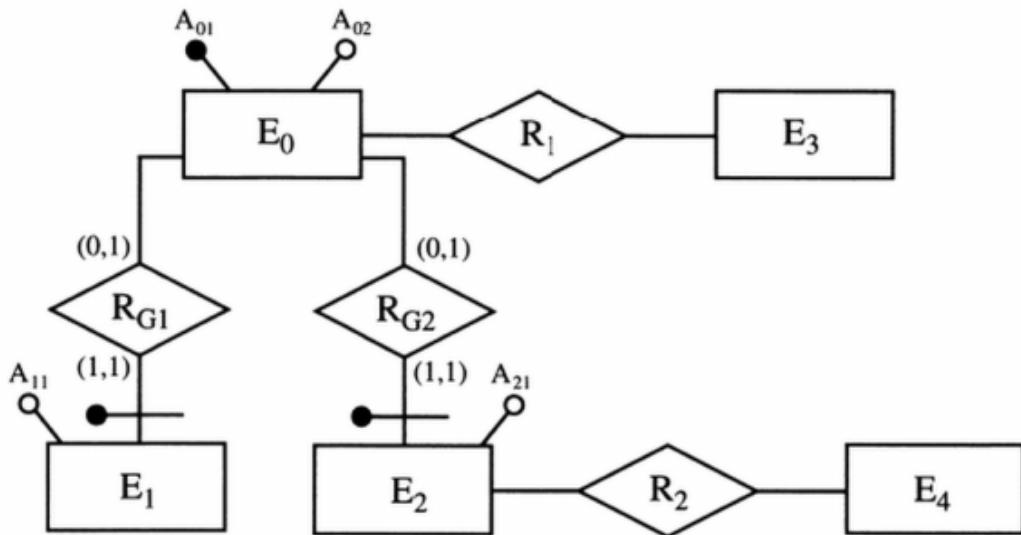


- Accorpamento del genitore nelle figlie.** Si elimina l'entità genitore E_0 . Per l'ereditarietà, gli attributi di E_0 , il suo identificatore e le relazioni cui partecipava, sono aggiunti alle figlie E_1 e E_2 . Le relazioni R_{11} ed R_{12} rappresentano la restrizione della relazione R_1 sulla occorrenza di E_1 ed E_2 . La cardinalità delle associazioni presenti non

vengono alterate.



3. **Sostituzione delle generalizzazioni con associazioni.** La generalizzazione si trasforma in due associazioni uno a uno che legano rispettivamente l'entità genitore con le entità figlie E_1 e E_2 . Non ci sono trasferimenti di attributi o associazioni e le entità E_1 ed E_2 sono identificate esternamente dall'entità E_0 . Nello schema ottenuto vanno aggiunti però dei vincoli: ogni occorrenza di E_0 non può partecipare contemporaneamente a R_{G1} e R_{G2} . Inoltre, se la generalizzazione è totale, ogni occorrenza di E_0 deve partecipare o a un'occorrenza di R_{G1} oppure ad un'occorrenza di R_{G2} .



La scelta fra le alternative si può effettuare dopo un'analisi quantitativa, analogamente all'analisi delle ridondanze, oppure in seguito ad un'analisi qualitativa che tenga conto di semplici regole generali:

- La prima conviene quando le operazioni non fanno molta distinzione fra occorrenze e attributi di E_0 , E_1 e E_2 . Pur generando uno spreco di memoria per la presenza di valori nulli, induce un minor numero di accessi
- La seconda è possibile solo se la generalizzazione è totale, altrimenti le occorrenze di E_0 che non sono occorrenze né di E_1 né di E_2 non sarebbero rappresentate. Questa alternativa è conveniente quando ci sono operazioni che si riferiscono solo a occorrenze di E_1 o di E_2 , e dunque fanno distinzione tra le entità figlie. In questo caso si ottiene un risparmio di memoria rispetto alla prima alternativa poiché in linea di principio non ci

sono valori nulli, ed una riduzione di accessi rispetto alla terza alternativa perché non si deve visitare E_0 per accedere ai alcuni attributi di E_1 ed E_2

- La terza alternativa conviene quando la generalizzazione non è totale e ci sono operatori che si riferiscono solo a E_1 o E_2 , oppure si riferiscono solo a E_0 , e dunque fanno distinzioni tra entità figlia ed entità genitore.

Le alternative presentate non sono le uniche ammesse, ma è possibile effettuare ristrutturazioni che sono combinazioni delle precedenti. Inoltre per la stessa generalizzazione è possibile applicare trasformazioni differenti sui diversi livelli

Partizionamento/Accorpamento di entità e/o associazioni

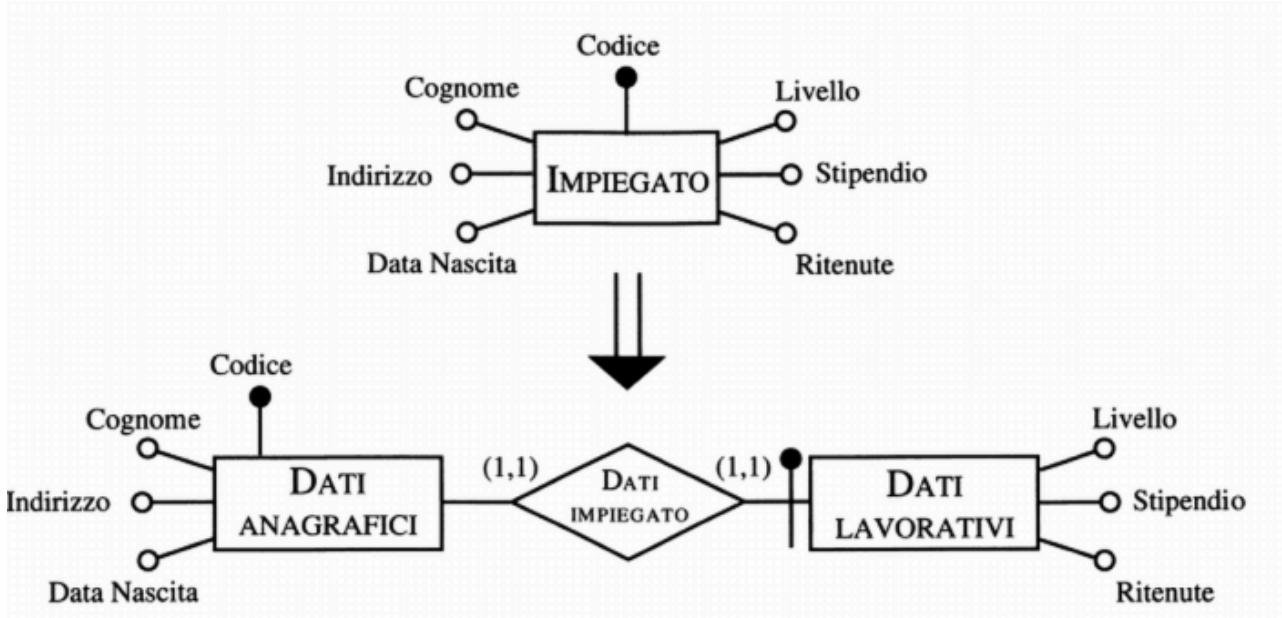
Entità e associazioni possono essere partizionati o accorpati per garantire maggiore efficienza delle operazioni in base a criteri simili a quelli usati per le generalizzazioni: Gli accessi si riducono separando attributi di uno stesso concetto che vengono acceduti da operazioni diverse e raggruppando attributi di concetti diversi che vengono acceduti dalle medesime operazioni.

Partizionamento (verticale e orizzontale) di entità

Si suddivide un concetto operando sui suoi attributi.

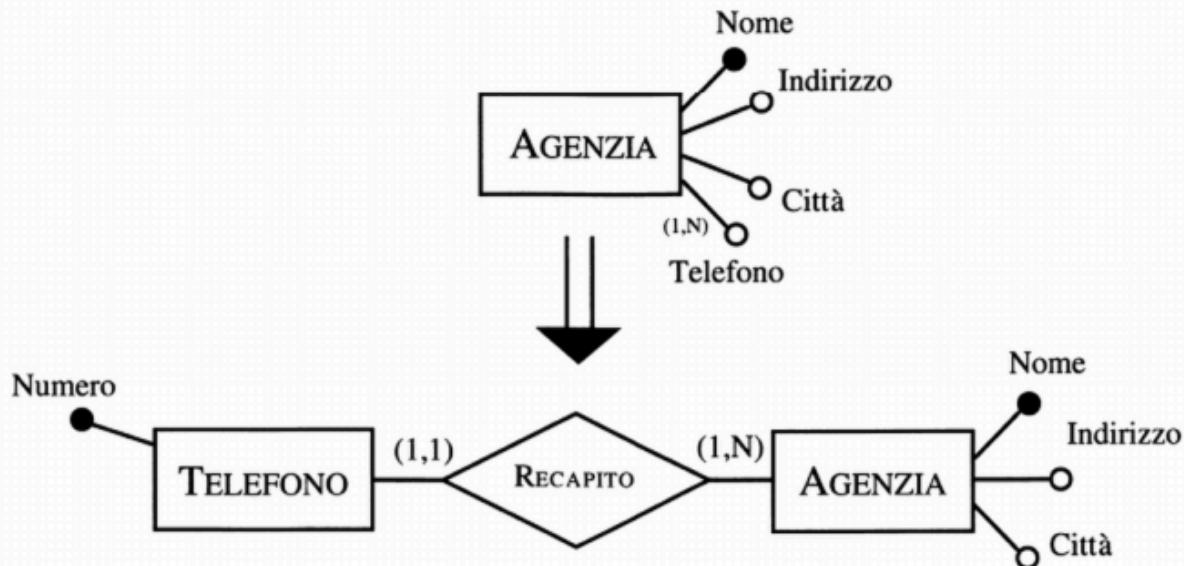
Questa ristrutturazione è conveniente se le operazioni che coinvolgono frequentemente l'entità originaria richiedono, ad esempio per un impiegato, o solo informazioni di carattere anagrafico o solo informazioni relative alla sua retribuzione. Un partizionamento di questo tipo è un esempio di partizionamento **verticale** di una entità. È possibile effettuare delle decomposizioni **orizzontali** nelle quali la suddivisione avviene sulle occorrenze dell'entità. Per esempio, per l'entità IMPIEGATO ci potrebbero essere alcune operazioni che riguardano soltanto gli analisti e altre che operano solo sui venditori. In questo caso però le entità ottenute (ANALISTA e VENDITORE) hanno gli stessi attributi dell'entità di partenza. Si osservi che una decomposizione orizzontale corrisponde all'introduzione di una generalizzazione a livello logico. L'effetto collaterale dei partizionamenti orizzontali è la duplicazione di tutte le associazioni a cui l'entità originaria partecipa. D'altra parte i

partizionamenti verticali ottimizzano il recupero delle informazioni.



Eliminazione di attributi multivalore

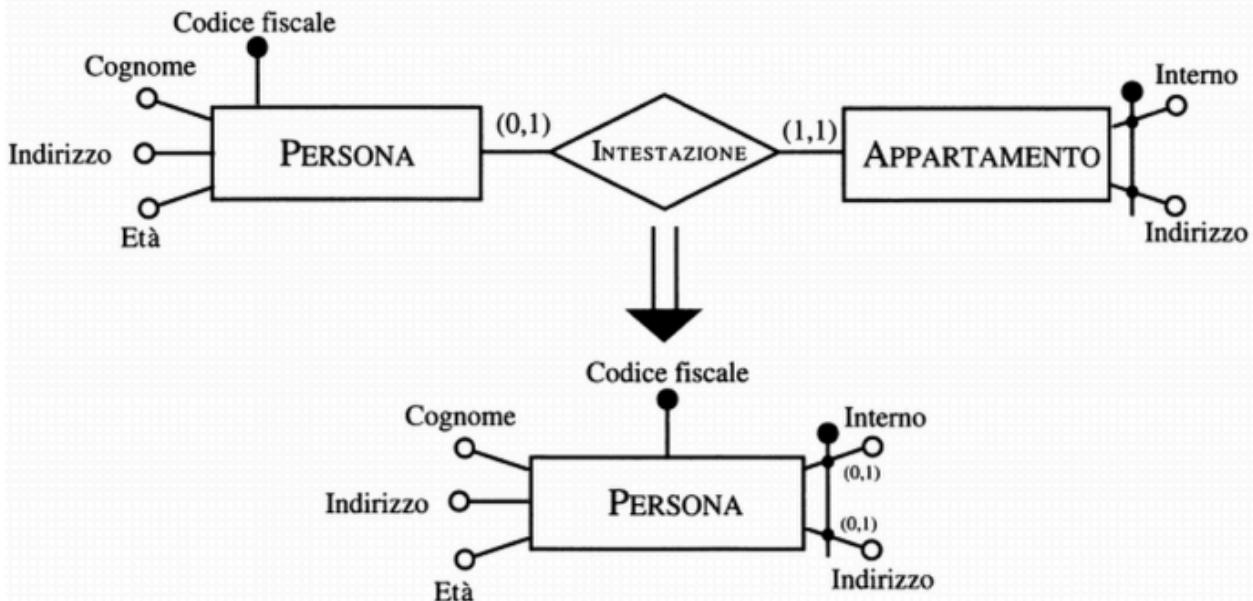
Questa ristrutturazione si rende necessaria perché, come per le generalizzazioni, il modello relazionale non permette di rappresentare questo tipo di attributo



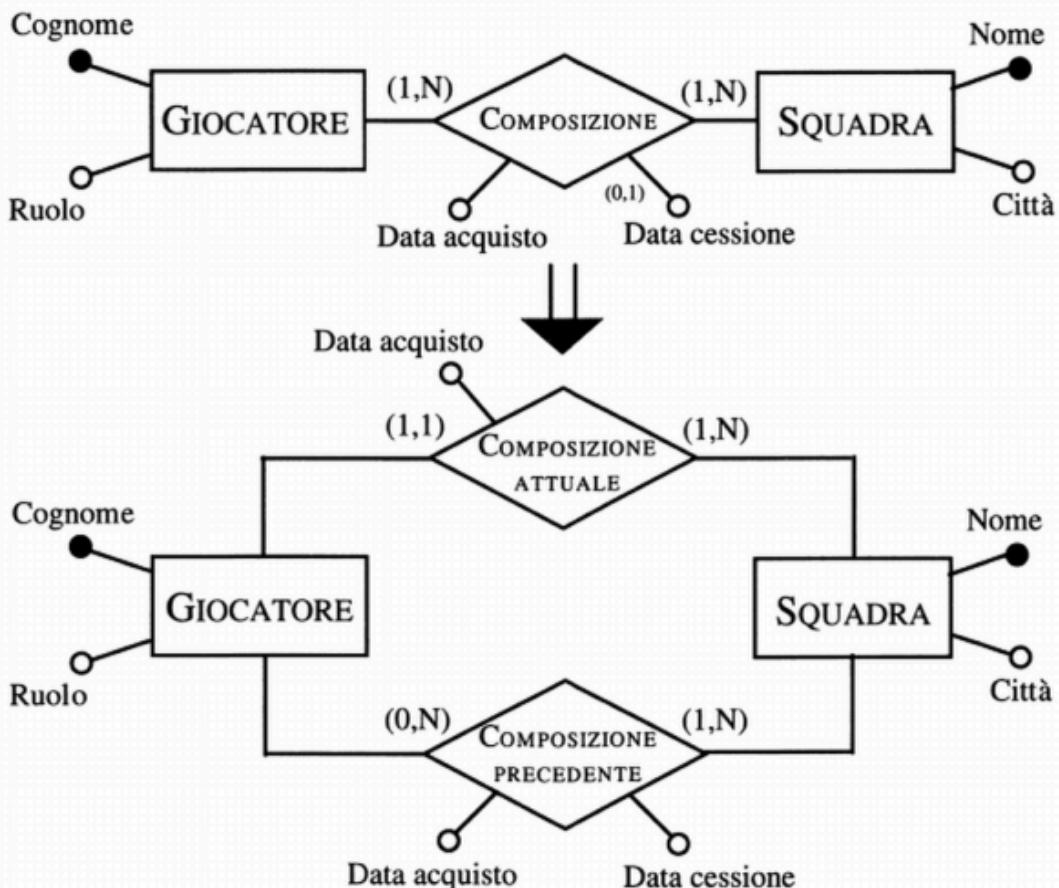
Come si vede in figura, l'entità **AGENZIA** avente l'attributo multivalore **TELEFONO** viene partizionata in due entità: una entità con lo stesso nome e gli stessi attributi a meno dell'attributo multivalore, e l'entità **TELEFONO**, con il solo attributo **Numero**, legata con una associazione uno a molti con l'entità **AGENZIA**. Se l'attributo multivalore fosse stato opzionale, allora la cardinalità minima dell'associazione sarebbe stata pari a zero.

Accorpamento di entità

È l'operazione inversa al partizionamento. Due entità connesse da relazione vengono accorpate in un'unica entità contenente gli attributi di entrambe. Si ottiene una riduzione degli accessi, ma con il rischio di uno spreco di memoria dovuto alla presenza di possibili valori nulli



Un ragionamento analogo si può estendere alle associazioni, come si vede nel prossimo esempio in cui vengono distinti i giocatori che compongono attualmente una squadra da quelli che ne facevano parte nel passato



Scelta degli identificatori principali

La scelta degli identificatori principali è indispensabile nelle traduzioni verso il modello relazionale, perché in questo modello le chiavi sono usate per stabilire legami tra dati di relazioni diverse. Quindi, nei casi in cui esistono entità per le quali sono stati specificati più

identificatori, bisogna decidere quale di questi identificatori verrà utilizzato come chiave primaria. I criteri per la scelta sono:

- Assenza di valori nulli, altrimenti non è garantito l'accesso a tutte le occorrenze dell'entità corrispondente.
- Semplicità: un identificatore composto da uno o pochi attributi è da preferire a identificatori costituiti da molti attributi.
- Utilizzo nelle operazioni più frequenti e/o importanti.
- Preferenza per gli identificatori interni

Traduzione verso il modello relazionale

La seconda fase della progettazione logica corrisponde a una traduzione tra modelli di dati diversi: a partire da uno schema E-R ristrutturato si costruisce uno schema logico equivalente, in grado cioè di rappresentare le medesime informazioni.

Facciamo riferimento a una versione semplificata del modello E-R, che non contiene generalizzazioni e attributi multivalue, e nella quale ogni entità ha un solo identificatore.

Affrontiamo il problema della traduzione caso per caso, iniziando dal caso più generale (quello di entità legate da associazioni molti a molti) che ci suggerisce l'idea generale su cui si basa la metodologia di traduzione.

Associazioni molti a molti ($N : N$)

Consideriamo lo schema in figura, la sua traduzione naturale nel modello relazionale prevede:

- Per ogni entità, una relazione con lo stesso nome avente per attributi i medesimi attributi dell'entità e per chiave il suo identificatore;
- Per l'associazione, una relazione con lo stesso nome avente per attributi gli attributi dell'associazione e gli identificatori delle entità coinvolte, tali identificatori formano la chiave della relazione.

Se gli attributi originali di entità o associazioni sono opzionali, i corrispondenti attributi di relazione possono assumere valori nulli.



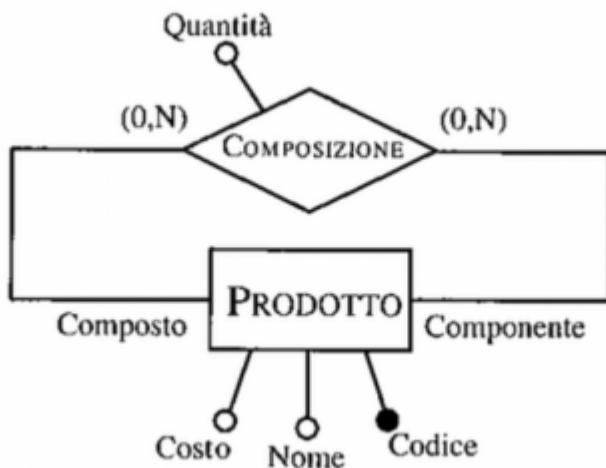
Lo schema relazionale che si ottiene è il seguente:

IMPIEGATO(Matricola, Cognome, Stipendio)PROGETTO(Codice, Nome, Budget)PARTECIPAZIONE(Matricola, Codice, DataInizio)

Vanno sempre specificati i **vincoli di integrità referenziale** presenti tra gli schemi creati. Nel caso in esempio esistono due vincoli tra gli attributi Matricola e Codice di PARTECIPAZIONE e gli omonimi attributi delle entità IMPIEGATO e PROGETTO. Per rendere più comprensibile il significato dello schema è conveniente effettuare alcune ridenominazioni, ad esempio:

PARTECIPAZIONE(Impiegato, Progetto, DataInizio)

La ridenominazione è essenziale in presenza di associazioni ricorsive

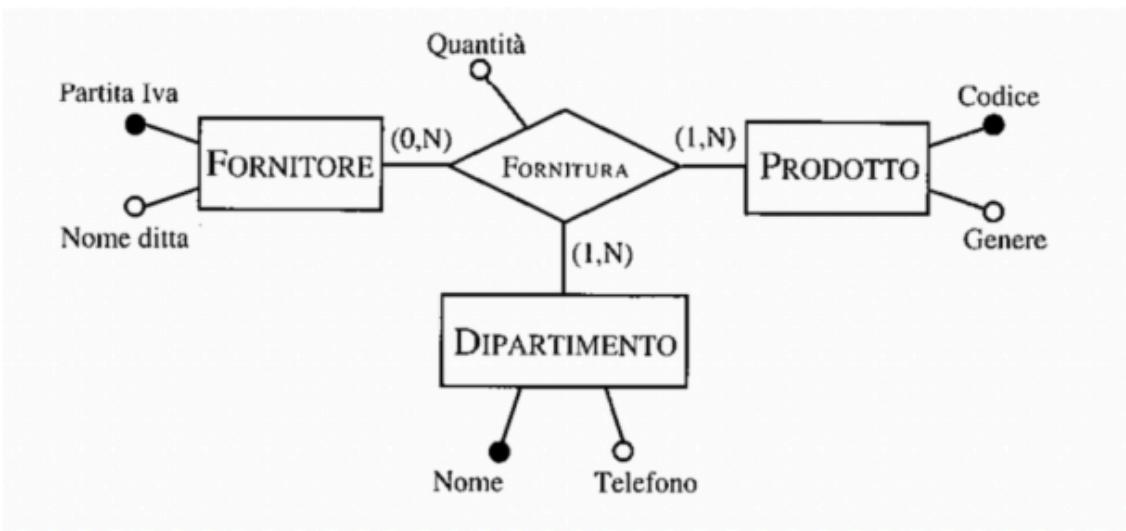


Questo schema si traduce nelle seguenti due relazioni:

PRODOTTO(Codice, Nome, Costo)COMPOSIZIONE(Composto, Componente, Quantità)

Esistono vincoli di integrità referenziale tra gli attributi Composto e Componente di COMPOSIZIONE e la chiave di PRODOTTO.

Le associazioni con più di due entità partecipanti si traducono in maniera analoga alle associazioni binarie.

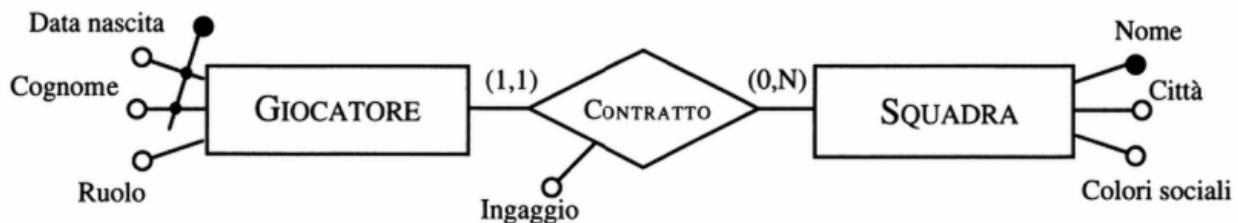


Lo schema in figura si traduce nelle seguenti tre relazioni:

$\text{FORNITORE}(\underline{\text{PartitaIVA}}, \underline{\text{NomeDitta}})$
 $\text{PRODOTTO}(\underline{\text{Codice}}, \underline{\text{Genere}})$ $\text{DIPARTIMENTO}(\underline{\text{Nome}}, \underline{\text{Telefono}})$
 $\text{FORNITURA}(\underline{\text{Fornitore}}, \underline{\text{Prodotto}}, \underline{\text{Dipartimento}}, \underline{\text{Quantità}})$

Associazioni uno a molti (1 : N)

Consideriamo lo schema in figura.



Secondo la regola vista per le associazioni molti a molti, la traduzione di questo schema dovrebbe essere la seguente:

$\text{GIOCATORE}(\underline{\text{Cognome}}, \underline{\text{DataNascita}}, \underline{\text{Ruolo}})$
 $\text{CONTRATTO}(\underline{\text{CognGiocatore}}, \underline{\text{DataNascG}}, \underline{\text{Squadra}}, \underline{\text{Ingaggio}})$
 $\text{SQUADRA}(\underline{\text{Nome}}, \underline{\text{Città}}, \underline{\text{ColoriSociali}})$

Ma la traduzione appena vista non è la più corretta, infatti essendo una relazione di tipo uno a molti, per identificare univocamente una squadra è sufficiente l'id di GIOCATORE.

Ottenendo in questo modo una nuova traduzione:

$\text{GIOCATORE}(\underline{\text{Cognome}}, \underline{\text{DataNascita}}, \underline{\text{Ruolo}})$
 $\text{CONTRATTO}(\underline{\text{CognGiocatore}}, \underline{\text{DataNascG}}, \underline{\text{Squadra}}, \underline{\text{Ingaggio}})$
 $\text{SQUADRA}(\underline{\text{Nome}}, \underline{\text{Città}}, \underline{\text{ColoriSociali}})$

Ancora una volta, notiamo che nella relazione CONTRATTO, la chiave è costituita solo dall'identificatore di GIOCATORE perché le cardinalità dell'associazione ci dicono che ogni giocatore ha un contratto con una sola squadra. A questo punto le relazioni GIOCATORE e CONTRATTO hanno la stessa chiave, quindi si possono fondere in un'unica relazione.

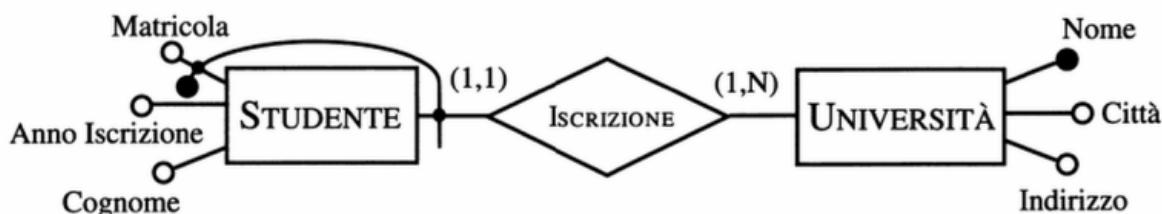
Si preferisce, dunque, la traduzione che segue, nella quale la relazione GIOCATORE rappresenta sia l'entità relativa sia l'associazione dello schema E-R originale:

GIOCATORE(Cognome, DataNascita, Ruolo, Squadra, Ingaggio)
 SQUADRA(Nome, Città, ColoriSociali)

con vincolo di integrità referenziale fra Squadra in GIOCATORE e la chiave di SQUADRA. Se la cardinalità minima dell'associazione è zero, allora SQUADRA e INGAGGIO in GIOCATORE devono ammettere valore nullo.

Entità con identificatore esterno

Le entità con identificatori esterni danno luogo a relazioni con chiavi che includono gli identificatori delle entità identificanti.



Per esempio, una traduzione per lo schema in figura potrebbe essere il seguente:

STUDENTE(Matricola, NomeUniversità, Cognome, AnnoIscrizione)
 UNIVERSITÀ(Nome, Città, Indirizzo)

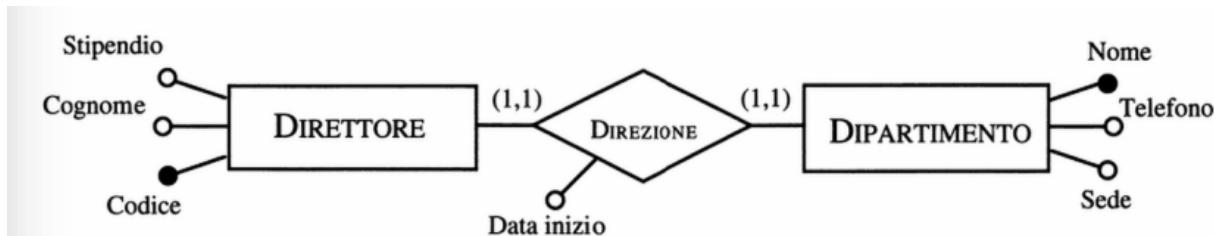
con vincolo di integrità referenziale tra l'attributo NomeUniversità della relazione STUDENTE e l'attributo Nome dell'entità UNIVERSITÀ.

Come si può vedere, rappresentando l'identificatore esterno si rappresenta direttamente anche l'associazione tra le due entità. Ricordiamo, infatti, che le entità identificate esternamente partecipano all'associazione sempre con cardinalità minima e massima pari a uno.

Associazioni uno a uno (1 : 1)

Per le associazioni uno a uno esistono diverse possibilità di traduzione sulla base delle cardinalità minime.

Cominciamo a vedere le associazioni uno a uno con partecipazione obbligatoria per entrambe le entità, come in figura:



In questo caso abbiamo due tipi di traduzioni completamente simmetriche:

DIRETTORE(Codice, Cognome, Stipendio, DipartimentoDiretto, InizioDirezione)
 DIPARTIMENTO(Nome, Telefono, Sede)

con vincolo di integrità referenziale tra l'attributo DipartimentoDiretto di DIRETTORE e Nome di DIPARTIMENTO.

Oppure

DIRETTORE(Codice, Cognome, Stipendio)

DIPARTIMENTO(Nome, Telefono, Sede, Direttore, InizioDirezione)

con vincolo di integrità referenziale tra l'attributo Direttore di DIPARTIMENTO e l'attributo Codice della relazione DIRETTORE.

Trattandosi di una relazione biunivoca, si potrebbe pensare di rappresentare tutti i concetti in un'unica relazione contenente tutti gli attributi in gioco. Questa alternativa è da escludere perché se ci fosse stato un valido motivo si sarebbe già ristrutturato lo schema E-R in modo da accoppare le due entità.

Consideriamo ora il caso di associazione uno a uno con **partecipazione opzionale** per una sola entità, come mostrato in figura.



In questo caso abbiamo una soluzione preferibile, ovvero la seguente:

IMPIEGATO(Codice, Cognome, Stipendio)

DIPARTIMENTO(Nome, Telefono, Sede, Direttore, InizioDirezione)

con vincolo di integrità referenziale tra l'attributo Direttore di DIPARTIMENTO e l'attributo Codice di IMPIEGATO.

Consideriamo infine il caso in cui entrambe le entità hanno partecipazione opzionale, come nel caso in figura, dove però possono esistere dipartimenti senza direttori, e quindi la cardinalità dell'entità DIPARTIMENTO diventa \$(0,1).

In questo caso esiste un ulteriore possibilità che prevede tre relazioni separate:

IMPIEGATO(Codice, Cognome, Stipendio)

DIPARTIMENTO(Nome, Telefono, Sede)

DIREZIONE(Direttore, Dipartimento, DataInizioDirezione)

con vincoli di integrità referenziale tra l'attributo Direttore di DIREZIONE e l'attributo Codice di IMPIEGATO e tra l'attributo Dipartimento di DIREZIONE e l'attributo Nome di DIPARTIMENTO.

Documentazione di schemi logici

Il risultato della progettazione logica è costituito dallo schema di una base di dati e la documentazione ad esso associata.

Buona parte della documentazione dello schema concettuale in ingresso alla fase di

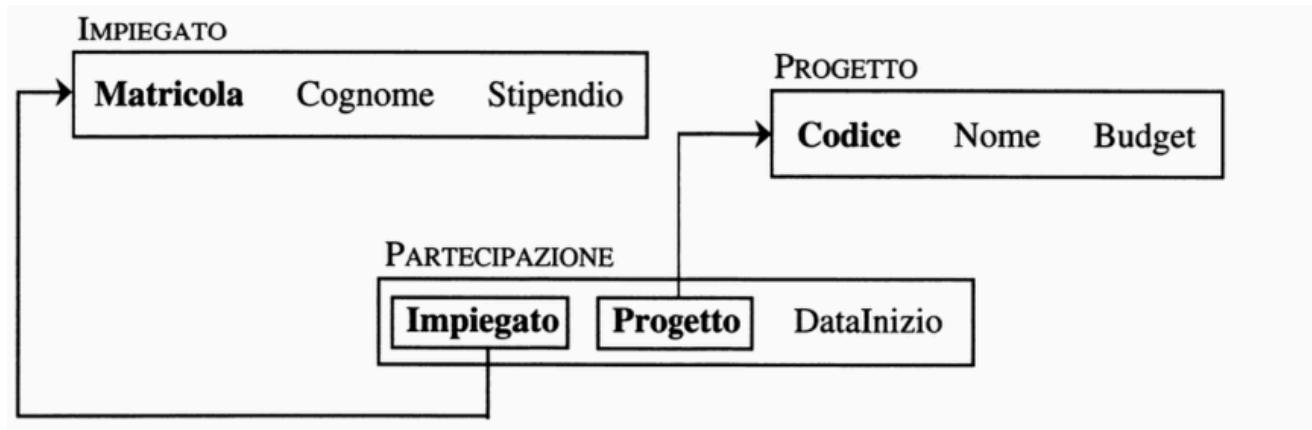
progettazione può essere ereditata dallo schema logico ottenuto come risultato di questa fase, in particolare se i nomi dei concetti dello schema E-R sono stati riutilizzati per costruire lo schema relazionale, le regole precedentemente definite possono essere utilizzate per documentare anche quest'ultimo.

A questa documentazione bisogna aggiungere ulteriore documentazione per descrivere vincoli di integrità referenziale introdotti dalla traduzione (in aggiunta ai vincoli individuati durante la progettazione concettuale), quindi a tal fine si adotta un semplice formalismo grafico.

Dato uno schema logico, i vincoli di integrità referenziale si documentano mediante un diagramma dove:

- Le chiavi delle relazioni sono rappresentate in grassetto
- Le frecce indicano vincoli di integrità referenziale
- Gli asterischi su attributi indicano possibili valori nulli

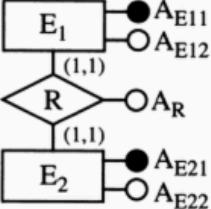
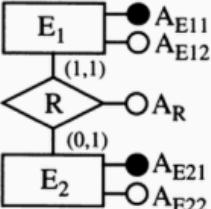
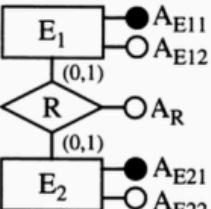
Prendendo in esempio lo schema logico sugli impiegati si otterrà questo:



È interessante osservare come, con questo tipo di rappresentazione, sia possibile rappresentare esplicitamente anche le associazioni dello schema Entità-Relazione di partenza alle quali, nello schema relazionale equivalente, non corrisponde nessuna relazione.

Extra

Tabella delle traduzioni dal modello E-R a quello relazionale

| Tipologia | Concetto iniziale | Risultati possibili |
|---|---|--|
| Associazione uno a uno con partecipazione obbligatoria per entrambe le entità |  | $E_1(A_{E11}, A_{E12}, A_{E21}, A_R)$ $E_2(A_{E21}, A_{E22})$ Oppure: $E_2(A_{E21}, A_{E22}, A_{E11}, A_R)$ $E_1(A_{E11}, A_{E12})$ |
| Associazione uno a uno con partecipazione opzionale per una entità |  | $E_1(A_{E11}, A_{E12}, A_{E21}, A_R)$ $E_2(A_{E21}, A_{E22})$ |
| Associazione uno a uno con partecipazione opzionale per entrambe le entità |  | $E_1(A_{E11}, A_{E12})$ $E_2(A_{E21}, A_{E22}, A_{E11}^*, A_R^*)$ Oppure: $E_1(A_{E11}, A_{E12}, A_{E21}^*, A_R^*)$ $E_2(A_{E21}, A_{E22})$ Oppure: $E_1(A_{E11}, A_{E12})$ $E_2(A_{E21}, A_{E22})$ $R(A_{E11}, A_{E21}, A_R)$ |

Esempio di progettazione logica

Guardare il PDF [5.3 - Esempio di Progettazione Logica.pdf](#), estratto dal libro e in più l'esercitazione data dalla prof [5.5 - Esercizi Progettazione Logica.pdf](#)

Normalizzazione

Esistono alcune proprietà, dette **forme normali**, che certificano la qualità dello schema di una base di dati relazionale tramite l'assenza di determinati difetti.

Quando una relazione non è normalizzata presenta ridondanze e si presta a comportamenti indesiderabili o anomali durante gli aggiornamenti.

Dunque, per **normalizzazione** si intende la procedura che permette di trasformare schemi non normalizzati in schemi che soddisfano una forma normale. È bene sottolineare, però, che la normalizzazione va utilizzata come tecnica di verifica dei risultati della progettazione di una base di dati, infatti una corretta applicazione di una metodologia di progettazione porta generalmente a schemi già normalizzati.

Facciamo un esempio con una tabella:

| IMPIEGATO | STIPENDIO | PROGETTO | BILANCIO | FUNZIONE |
|------------------|------------------|-----------------|-----------------|-----------------|
| Rossi | 20000 | Marte | 2000 | Tecnico |
| Verdi | 35000 | Giove | 15000 | Progettista |
| Verdi | 35000 | Venere | 15000 | Progettista |
| Neri | 55000 | Venere | 15000 | Direttore |
| Neri | 55000 | Giove | 15000 | Consulente |
| Neri | 55000 | Marte | 2000 | Consulente |
| Mori | 48000 | Marte | 2000 | Direttore |
| Mori | 48000 | Venere | 15000 | Progettista |
| Bianchi | 48000 | Venere | 15000 | Progettista |
| Bianchi | 48000 | Giove | 15000 | Direttore |

Si può notare facilmente che le tuple soddisfano le seguenti proprietà:

- Lo stipendio di ciascun impiegato è unico ed è funzione del solo impiegato, indipendentemente dai progetti a cui partecipa;
- Il bilancio di ciascun progetto è unico e dipende dal solo progetto, indipendentemente dagli impiegati che vi partecipano.

Questi fatti hanno alcune conseguenze sul contenuto della relazione e sulle operazioni che si possono effettuare su di essa, infatti si osserva che:

- Lo stipendio di ciascun impiegato è ripetuto in tutte le tuple relative a esso, generando **ridondanza** (se l'impiegato partecipasse a 20 progetti verrebbe ripetuto 20 volte)
- Se lo stipendio di un impiegato varia, è necessario modificare il valore in tutte le tuple corrispondenti, portando ad un **anomalia di aggiornamento**
- Se un impiegato interrompe la partecipazione a tutti i progetti senza lasciare l'azienda, non è possibile conservare traccia del suo nome e del suo stipendio (a meno di valori nulli sulla chiave), portando ad una **anomalia di cancellazione**
- Se si hanno informazioni su un nuovo impiegato, non è possibile inserirle finché questi non viene assegnato ad un progetto, portando ad una **anomalia di inserimento**

La motivazione a tutti questi inconvenienti deriva dal fatto che si sia utilizzata un'unica relazione per gestire dati e associazioni tra dati etereogeni, infatti nella relazione sono rappresentati:

- Gli impiegati con i relativi stipendi
- I progetti con i relativi bilanci
- Le partecipazioni degli impiegati ai progetti con le relative funzioni

Dipendenze funzionali

Per poter studiare in maniera sistematica i concetti introdotti formalmente nel paragrafo precedente è necessario fare uso delle **dipendenze funzionali**:

Si trattano di particolari vincoli di integrità per il modello relazionale che descrive legami di tipo funzionale tra gli attributi di una relazione.

Riconoscendo l'esempio precedente con la relazione in figura, abbiamo osservato che lo stipendio di ciascun impiegato è unico e quindi, ogni volta che in una tupla della relazione compare un certo impiegato, il valore del suo stipendio rimane sempre lo stesso.

Possiamo dunque dire che il valore dell'attributo *Impiegato* determina il valore dell'attributo *Stipendio* o, in maniera più precisa, che esiste una funzione che associa a ogni elemento del dominio dell'attributo *impiegato* un solo elemento del dominio dell'attributo *stipendio*.

Formalizzazione

Data una relazione r su uno schema $R(X)$ e due sottoinsiemi di attributi non vuoti Y e Z di X , diremo che esiste su r una dipendenza funzionale tra Y e Z se, per ogni coppia di tuple t_1 e t_2 di r aventi gli stessi valori sugli attributi Y , risulta che t_1 e t_2 hanno gli stessi valori anche sugli attributi Z .

Una dipendenza funzionale tra gli attributi Y e Z viene generalmente indicata con la notazione $Y \rightarrow Z$ e, come gli altri vincoli di integrità, viene associata ad uno schema: una relazione su quello schema verrà considerata corretta se soddisfa tale dipendenza funzionale

Osservazioni

- Se l'insieme Z è composto dagli attributi A_1, A_2, \dots, A_k , allora una relazione soddisfa $Y \rightarrow Z$ se e solo se soddisfa tutte le k dipendenze $Y \rightarrow A_1, Y \rightarrow A_2, \dots, Y \rightarrow A_k$. Di conseguenza, quando opportuno, possiamo, senza perdita di generalità, assumere che le dipendenze abbiano la forma $Y \rightarrow A$, in cui A è un singolo attributo.
- In base alla definizione data, possiamo notare che, nella nostra relazione, è verificata anche la dipendenza funzionale:

$$\text{Impiegato Progetto} \rightarrow \text{Progetto}$$

Questa è una dipendenza funzionale **banale** in quanto asserisce una proprietà ovvia di una relazione, infatti due tuple con gli stessi valori sulla coppia di attributi *Impiegato Progetto*, hanno ovviamente lo stesso valore sull'attributo *Progetto*, che è uno dei due. Diremo quindi che una dipendenza funzionale $Y \rightarrow A$ è **non banale** se A non compare tra gli attributi di Y .

- Se prendiamo una chiave K di una relazione r , si può facilmente verificare che esiste una dipendenza funzionale tra K e ogni altro attributo dello schema r . Questo perché, per definizione stessa di vincolo di chiave, non possono esistere due tuple con gli stessi valori su K e quindi una dipendenza funzionale che ha K al primo membro sarà sempre soddisfatta.

Con riferimento al nostro esempio abbiamo detto che gli attributi *Impiegato* e *Progetto* formano una chiave. Possiamo allora affermare che, per esempio, vale la dipendenza

funzionale Impiegato Progetto → Funzione. In particolare esisterà una dipendenza funzionale tra una chiave di una relazione e tutti gli attributi dello schema della relazione. Nel nostro caso abbiamo che:

Impiegato Progetto → Stipendio Bilancio Funzione

Possiamo quindi concludere dicendo che il vincolo di dipendenza funzionale **generalizza** il vincolo di chiave. Possiamo dire che una dipendenza funzionale $Y \rightarrow Z$ su uno schema $R(X)$ degenera nel vincolo di chiave se l'unione di Y e Z è pari a X . In tal caso, infatti, Y è (super)chiave per lo schema $R(X)$.

Forma normale di Boyce e Codd

Alla luce di quanto detto sulle dipendenze funzionali, l'idea fondamentale è che si possono introdurre delle proprietà dette **forme normali** che sono soddisfatte quando non ci sono anomalie.

Negli esempi precedenti notiamo che:

- Le dipendenze Impiegato → Stipendio e Progetto → Bilancio sono causa di anomalie
- La dipendenza Impiegato Progetto → Funzione non lo è

La differenza risiede nel fatto che Impiegato Progetto è una superchiave della relazione (più specificatamente è l'unica chiave).

Possiamo quindi concludere che le ridondanze e le anomalie sono causate dalle dipendenze funzionali $X \rightarrow A$ che permettono la presenza di più tuple fra loro uguali sugli attributi di X , ossia sulle dipendenze $X \rightarrow A$ tali che X non contiene una chiave.

Questo concetto è alla base della forma normale di Boyce e Codd (BCNF), secondo la quale: una relazione r è in forma normale di Boyce e Codd se per ogni dipendenza funzionale (non banale) $X \rightarrow Y$ definita su r , X è superchiave per r

Decomposizione in forma normale di Boyce e Codd

Data una relazione che non soddisfa la forma normale di Boyce e Codd è possibile, in molti casi, sostituirla con due o più relazioni normalizzate attraverso un processo detto di **normalizzazione**.

Questo processo si fonda su un semplice criterio:

Se una relazione rappresenta più concetti indipendenti, allora va decomposta in relazioni più piccole, una per ogni concetto.

Ora, riprendiamo in esempio sempre la tabella iniziale vista in [Normalizzazione](#), per mezzo di proiezioni sugli insiemi di attributi (rispettivamente corrispondenti ai tre concetti prima menzionati) eliminiamo le ridondanze.

Alla fine di questo processo otterremo le seguenti tabelle:

| Impiegato | Stipendio |
|-----------|-----------|
| Rossi | 20 000 |
| Verdi | 35 000 |
| Neri | 55 000 |
| Mori | 48 000 |
| Bianchi | 48 000 |

| Progetto | Bilancio |
|----------|----------|
| Marte | 2000 |
| Giove | 15 000 |
| Venere | 15 000 |

| Impiegato | Progetto | Funzione |
|-----------|----------|-------------|
| Rossi | Marte | tecnico |
| Verdi | Giove | progettista |
| Verdi | Venere | progettista |
| Neri | Venere | direttore |
| Neri | Giove | consulente |
| Neri | Marte | consulente |
| Mori | Marte | direttore |
| Mori | Venere | progettista |
| Bianchi | Venere | progettista |
| Bianchi | Giove | direttore |

Le tre relazioni ora sono in forma normale di Boyce e Codd.

Si osservi che abbiamo costruito le relazioni in modo che a ciascuna dipendenza corrisponda una diversa relazione la cui chiave è proprio il primo membro della dipendenza stessa.

In tal modo il soddisfacimento della forma normale di Boyce e Codd è garantito per la definizione stessa di tale forma normale.

Proprietà delle decomposizioni

Nell'esempio appena visto, la separazione delle dipendenze è stata facilitata dalla struttura delle dipendenze stesse, "naturalmente" separate ed indipendenti l'una dall'altra. Purtroppo questa procedura non è valida in generale, infatti basare le decomposizioni sulle dipendenze funzionali può essere non necessario o non possibile, inoltre individuare le dipendenze funzionali utili ai fini della decomposizione può essere difficile. Per questo è necessario individuare delle proprietà generalmente valide, che devono essere soddisfatte da una buona normalizzazione.

Tali proprietà sono le seguenti:

- Decomposizione senza perdita
- Conservazione delle dipendenze

Decomposizioni senza perdita

Andiamo ad esaminare questa relazione:

| IMPIEGATO | PROGETTO | SEDE |
|------------------|-----------------|-------------|
| Rossi | Marte | Roma |
| Verdi | Giove | Milano |
| Verdi | Venere | Milano |
| Neri | Saturno | Milano |
| Neri | Venere | Milano |

Tale relazione soddisfa le dipendenze funzionali:

$$\begin{aligned} \text{Impiegato} &\rightarrow \text{Sede} \\ \text{Progetto} &\rightarrow \text{Sede} \end{aligned}$$

che specificano il fatto che ciascun impiegato opera presso un'unica sede e che ciascun Progetto è sviluppato presso un'unica sede. Inoltre un impiegato può partecipare a più progetti, ma questi devono essere assegnati tutti alla sede cui afferisce.

Separando sulla base delle dipendenze funzionali, saremmo portati a decomporre la relazione in due parti:

- Una relazione sugli attributi Impiegato e Sede, in corrispondenza della dipendenza $\text{Impiegato} \rightarrow \text{Sede}$
- Una relazione sugli attributi Progetto e Sede, in corrispondenza della dipendenza $\text{Progetto} \rightarrow \text{Sede}$

Quindi l'istanza iniziale verrebbe decomposta per mezzo di proiezioni sugli attributi coinvolti, nelle due relazioni come segue:

| Impiegato | Sede | Progetto | Sede |
|------------------|-------------|-----------------|-------------|
| Rossi | Roma | Marte | Roma |
| Verdi | Milano | Giove | Milano |
| Neri | Milano | Saturno | Milano |
| | | Venere | Milano |

Dopo la decomposizione, la ricostruire delle informazioni di partenza, dunque la relazione originaria a partire dalle sue proiezioni, deve essere effettuata per mezzo di un'operazione di join naturale sull'attributo comune Sede, producendo la tabella seguente:

| Impiegato | Progetto | Sede |
|-----------|----------|--------|
| Rossi | Marte | Roma |
| Verdi | Giove | Milano |
| Verdi | Venere | Milano |
| Neri | Saturno | Milano |
| Neri | Venere | Milano |
| Verdi | Saturno | Milano |
| Neri | Giove | Milano |

la quale contiene tutte le tuple della relazione originaria più altre tuple **spurie** (le ultime due in questa tabella), infatti l'impiegato Verdi lavora a Milano e il Progetto Saturno ha sede a Milano, ma Verdi non lavora a tale Progetto.

In questo caso è impossibile ricostruire tutte e sole le informazioni della relazione originaria.

Affermiamo quindi che, data una relazione r su un insieme di attributi X , con X_1 e X_2 sottoinsiemi di X la cui unione sia pari a X stesso, si può decomporre senza perdita di dati sugli insiemi X_1 e X_2 se il join delle due proiezioni è uguale a r stessa (ossia non contiene **spurie**). È irrinunciabile che una decomposizione effettuata al fine di normalizzare sia senza perdita.

È possibile individuare una condizione che garantisce la decomposizione senza perdita di una relazione come segue:

Sia r una relazione su X e siano X_1 e X_2 sottoinsiemi di X tali che $X_1 \cup X_2 = X$; inoltre sia $X_0 = X_1 \cap X_2$; allora: r si decomponete senza perdita su X_1 e X_2 se soddisfa la dipendenza funzionale $X_0 \rightarrow X_1$ oppure la dipendenza funzionale $X_0 \rightarrow X_2$, oppure entrambe.

In altre parole, la decomposizione senza perdita è garantita se gli attributi comuni formano una chiave per almeno una delle relazioni decomposte. Ciò avviene se gli attributi comuni costituiscono il primo membro di almeno una delle dipendenze su cui si effettua la decomposizione. Nell'esempio, possiamo vedere che l'intersezione degli insiemi di attributi su cui abbiamo effettuato le due proiezioni è costituita dall'attributo Sede, che non è il primo membro di alcuna dipendenza funzionale, ovvero, non è chiave per nessuna delle due relazioni.

È opportuno notare come la condizione enunciata sia **sufficiente** ma non **necessaria** per la decomposizione senza perdita, esistono infatti istanze di relazione che non soddisfano nessuna delle due dipendenze ma al tempo stesso si decompongono senza perdita come si vede nella tabella di sopra.

La condizione in questione garantisce che tutte le istanze di relazione che soddisfano un dato insieme di dipendenze si decompongano senza perdita, rendendolo un risultato utilizzabile in pratica:

Ogniqualvolta che decomponiamo una relazione in due parti, se l'insieme degli attributi

comuni è chiave per una delle due relazioni allora possiamo essere certi che tutte le istanze della relazione si decompongono senza perdita.

Conservazione delle dipendenze

Tornando alla tabella di esempio precedentemente vista, possiamo rimuovere ancora anomalie utilizzando solo la dipendenza $\text{Impiegato} \rightarrow \text{Sede}$ per ottenere una decomposizione senza perdita (oppure volendo $\text{Progetto} \rightarrow \text{Sede}$, si otterrebbe comunque lo stesso risultato). Alla fine otteniamo due relazioni, una sugli attributi Impiegato e Sede e l'altra su Impiegato e Progetto , venendo rappresentata graficamente in questa maniera:

| Impiegato | Sede | Impiegato | Progetto |
|------------------|-------------|------------------|-----------------|
| Rossi | Roma | Rossi | Marte |
| Verdi | Milano | Verdi | Giove |
| Neri | Milano | Verdi | Venere |
| | | Neri | Saturno |
| | | Neri | Venere |

Il join di queste due relazioni produce effettivamente la relazione originaria, potendo affermare di aver ottenuto una decomposizione senza perdita.

Questa decomposizione, però, presenta un altro inconveniente:

Supponiamo di voler inserire una nuova tupla che specifica la partecipazione dell'impiegato Neri, che opera a Milano, al Progetto Marte, sulla relazione originaria un tale aggiornamento verrebbe immediatamente individuato come illecito, perché porterebbe ad una violazione della dipendenza funzionale $\text{Progetto} \rightarrow \text{Sede}$.

Sulle relazioni decomposte, al contrario, non è possibile rilevare alcuna violazione di dipendenze. Infatti, sulla relazione avente per attributi Impiegato e Progetto non è possibile definire alcuna dipendenza funzionale (quindi non possono esserci violazioni), mentre nella relazione su Impiegato e Sede la tupla con valori Neri e Milano soddisfa la dipendenza funzionale $\text{Impiegato} \rightarrow \text{Sede}$.

Ne evince che non è possibile fare alcuna verifica sulla dipendenza funzionale $\text{Progetto} \rightarrow \text{Sede}$ perché i due attributi Progetto e Sede sono stati separati in due relazioni diverse.

In generale, il teorema afferma che una decomposizione conserva le dipendenze se ciascuna delle dipendenze funzionali dello schema originario coinvolge attributi che compaiono tutti insieme in uno degli schemi decomposti.

Qualità delle decomposizioni

In sintesi, a partire da uno schema non normalizzato, se ne ottiene uno normalizzato valido, se la decomposizione è:

- **Senza perdite**, ovvero, garantisce la ricostruzione di tutte e sole le informazioni presenti nella relazione originaria a partire dalle informazioni rappresentate nelle relazioni decomposte
- **Conserva le dipendenze**, ovvero, garantisce il mantenimento dei vincoli di integrità originari e, quindi, permetta di rilevare aggiornamenti illeciti.

Di conseguenza considereremo accettabili solo le decomposizioni che soddisfano queste due proprietà.

Terza forma normale

Limitazioni della forma normale di Boyce e Codd

Nella maggior parte dei casi si può raggiungere l'obiettivo di una buona decomposizione in forma normale di Boyce e Codd, talvolta però questo non è possibile.

Possiamo vederlo tramite un esempio, consideriamo questa relazione

| <u>DIRIGENTE</u> | <u>PROGETTO</u> | <u>SEDE</u> |
|------------------|-----------------|-------------|
| Rossi | Marte | Roma |
| Verdi | Giove | Milano |
| Verdi | Marte | Milano |
| Neri | Saturno | Milano |
| Neri | Venere | Milano |

Su di essa possiamo supporre che siano definite le seguenti dipendenze:

- Dirigente → Sede: ogni dirigente opera presso una sede
- Progetto Sede → Dirigente: ogni Progetto ha più dirigenti che ne sono responsabili, ma in sedi diverse, e ogni dirigente può essere responsabile di più progetti; però, per ogni sede, un Progetto ha un solo responsabile

La relazione non è in forma normale di Boyce e Codd, perché:

- Il primo membro della dipendenza Dirigente → Sede non è superchiave
- La dipendenza Progetto Sede → Dirigente coinvolge tutti gli attributi e quindi nessuna decomposizione è in grado di conservarla.

L'esempio ci mostra quindi che esistono schemi che violano la forma normale di Boyce e Codd per i quali non esiste alcuna decomposizione che conservi le dipendenze.

Per trattare casi come questi, si ricorre ad una forma normale meno restrittiva, ossia la terza forma normale.

Definizione di terza forma normale

Una relazione r è in **terza forma normale** (3FN) se, per ogni dipendenza funzionale (non banale) $X \rightarrow A$ definita su di essa, almeno una delle seguenti condizioni è verificata:

- X contiene una chiave K di r
- A appartiene ad almeno una chiave di r

La terza forma normale è meno restrittiva rispetto alla BCNF, ma ha il vantaggio di essere sempre raggiungibile. Tornando all'esempio, possiamo verificare che la relazione soddisfa la 3NF, infatti la dipendenza Progetto Sede \rightarrow Dirigente ha come primo membro una chiave della relazione, mentre Dirigente \rightarrow Sede, pur non contenendo una chiave al primo membro ha un unico attributo a secondo membro che fa parte della chiave Progetto Sede.

Si osservi che la relazione presenta una forma di ridondanza: ogni volta che un dirigente compare in una tupla, viene ripetuta per esso la sede in cui opera. Questa ridondanza viene tollerata dalla 3NF perché non sarebbe possibile una decomposizione che la elimini e al tempo stesso conservi tutte le dipendenze.

Decomposizione in terza forma normale

Una relazione che non soddisfa la terza forma normale si decompone in relazioni ottenute per proiezione sugli attributi corrispondenti alle dipendenze funzionali (quindi si crea una relazione per ogni dipendenza funzionale) e, successivamente, si verifica che alla fine una relazione contenga una chiave della relazione originaria.

Consideriamo l'esempio della seguente relazione:

| IMPIEGATO | PROGETTO | STIPENDIO |
|------------------|-----------------|------------------|
| Rossi | Marte | 30000 |
| Verdi | Giove | 30000 |
| Verdi | Venere | 30000 |
| Neri | Saturno | 40000 |
| Neri | Venere | 40000 |

Su questa può essere riconosciuta la sola dipendenza funzionale Impiegato \rightarrow Stipendio. Questa relazione non è in BCNF in quanto nella dipendenza funzionale, Impiegato non è superchiave, non è in 3NF poiché Stipendio non è contenuto in almeno una chiave della relazione. In questo caso, se si effettuasse una decomposizione in una relazione sugli attributi Impiegato Stipendio e un'altra solo sull'attributo Progetto si violerebbe la proprietà di decomposizione senza perdita, proprio perché nessuna delle due relazioni contiene una chiave. Per garantire tale proprietà dobbiamo invece definire la seconda relazione sugli attributi Impiegato Progetto, che formano una chiave della relazione originaria, ottenendo in questo modo questa decomposizione:

| Impiegato | Progetto |
|-----------|----------|
| Rossi | Marte |
| Verdi | Giove |
| Verdi | Venere |
| Neri | Saturno |
| Neri | Venere |

| Impiegato | Stipendio |
|-----------|-----------|
| Rossi | 30 000 |
| Verdi | 30 000 |
| Neri | 40 000 |

Una decomposizione in terza forma normale produce, nella maggior parte dei casi, schemi in forma normale di Boyce-Codd. Si può dimostrare che se una relazione ha solo una chiave allora le due forme normali coincidono, cioè una relazione in 3NF è anche in BCNF.

Altre forme normali

Oltre alla terza forma normale esistono altre due forme normali:

1. La **prima forma normale** (1NF) stabilisce una condizione che sta alla base del modello relazione stesso: gli attributi delle relazioni sono definiti su valori atomici e non su valori complessi.
2. Una relazione è in **seconda forma normale** (2NF) se su di essa non sono definite dipendenze parziali, cioè dipendenze fra un sottoinsieme proprio della chiave e altri attributi.

Quindi le relazioni chiave di un solo attributo sono in seconda forma normale.

Una relazione in seconda forma normale è una variante debole della terza forma normale, facciamo un esempio prendendo questa relazione:

| Impiegato | Categoria | Stipendio |
|-----------|-----------|-----------|
| Neri | 3 | 30 000 |
| Verdi | 3 | 30 000 |
| Rossi | 4 | 50 000 |
| Mori | 4 | 50 000 |
| Bianchi | 5 | 72 000 |

Essa soddisfa le dipendenze $\text{Impiegato} \rightarrow \text{Categoria}$ e $\text{Categoria} \rightarrow \text{Stipendio}$, violando la terza forma normale poiché Categoria non è chiave. La seconda forma tollera la dipendenza tra Categoria e Stipendio perché Stipendio dipende comunque dall'intera chiave Impiegato.

La 2NF ammette **dipendenze funzionali transitive**, ossia dipendenze nella forma $K \rightarrow A$ dove K è la chiave ed esiste un altro insieme di attributi X , non chiave, con dipendenze $K \rightarrow X$ e $X \rightarrow A$.

Le dipendenze transitive non possono essere ammesse nella terza forma normale, in quanto

si potrebbe avere una dipendenza in cui il dominio non è superchiave ed il codominio non è contenuto in alcuna chiave della relazione di partenza.

È importante notare che tanto le dipendenze parziali quanto quelle transitive violano la terza forma normale così come noi l'abbiamo definita perché coinvolgono una dipendenza funzionale il cui primo membro non è super chiave

Normalizzazione e scelta degli attributi

Andando a riprendere la relazione d'esempio nel paragrafo [Limitazioni della forma normale di Boyce e Codd](#), possiamo arrivare alla conclusione che avremmo potuto descrivere l'applicazione di interesse in maniera più appropriata introducendo un ulteriore attributo, ossia Reparto, che partiziona le singole sedi sulla base dei responsabili.

Le dipendenze in questo caso possono essere così definite:

- Dirigente → Sede Reparto: ogni dirigente opera presso una sede e dirige un reparto
- Sede reparto → Dirigente: per ogni sede e reparto c'è un solo dirigente
- Progetto Sede → Reparto: per ogni sede, un progetto è assegnato a un solo reparto; la dipendenza funzionale Progetto Sede → Dirigente è quindi ricostruibile

| DIRIGENTE | PROGETTO | SEDE | REPARTO |
|------------------|-----------------|-------------|----------------|
| Rossi | Marte | Roma | 1 |
| Verdi | Giove | Milano | 1 |
| Verdi | Marte | Milano | 1 |
| Neri | Saturno | Milano | 2 |
| Neri | Venere | Milano | 2 |

Per questo schema esiste una buona decomposizione, ossia:

| Progetto | Sede | Reparto |
|-----------------|-------------|----------------|
| Marte | Roma | 1 |
| Giove | Milano | 1 |
| Marte | Milano | 1 |
| Saturno | Milano | 2 |
| Venere | Milano | 2 |

| Dirigente | Sede | Reparto |
|------------------|-------------|----------------|
| Rossi | Roma | 1 |
| Verdi | Milano | 1 |
| Neri | Milano | 2 |

- La decomposizione è senza perdita, gli attributi comuni Sede e Reparto formano una chiave per la prima relazione
- Le dipendenze sono conservate, per ciascuna dipendenza esiste una relazione decomposta che ne contiene tutti gli attributi
- Entrambe le relazioni sono in forma normale di Boyce e Codd: tutte le dipendenze hanno il primo membro costituito da una chiave

Possiamo quindi concludere che spesso la non raggiungibilità della forma normale di Boyce e Codd è dovuta a una analisi non sufficientemente accurata.

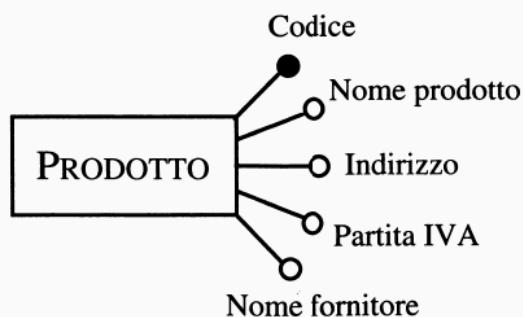
Progettazione di basi di dati e normalizzazione

La normalizzazione può essere utilizzata come base per operazioni di verifica di qualità degli schemi sia nella fase di progettazione concettuale (schemi ER) che logica (schema relazionale)

Verifiche di normalizzazioni su entità

Nel caso dello schema ER, è possibile considerare ciascuna entità e ciascuna associazione come una relazione, considerando le dipendenze funzionali che sussistono tra attributi dell'entità e verificare che ciascuna di esse abbia come primo membro l'identificatore (o lo contenga).

Prendiamo un esempio:



Possiamo notare che tra gli attributi di tale entità sussiste la dipendenza funzionale $\text{Partita IVA} \rightarrow \text{Nome fornitore}$ Indirizzo . Inoltre tutti gli attributi dipendono funzionalmente dall'attributo Codice , che costituisce l'identificatore dell'entità. Poiché Codice è l'unico identificatore, possiamo concludere che l'entità viola la terza forma normale, in quanto la dipendenza $\text{Partita IVA} \rightarrow \text{Nome fornitore}$ Indirizzo ha un primo membro che non contiene l'identificatore e un secondo membro composto da attributi che non fanno parte della chiave. In questi casi, la verifica di normalizzazione ci permette di segnalare il fatto che lo schema concettuale non è accurato e ci suggerisce di decomporre l'entità stessa. Dunque, rilevando la dipendenza funzionale, abbiamo capito che il concetto di fornitore è in effetti indipendente da quello di prodotto e ha proprietà associate (partita IVA, nome, indirizzo), quindi possiamo dire che è opportuno modellare il concetto di per mezzo di una entità chiamata **FORNITORE**, con identificatore costituito dall'attributo **Partita IVA** e ulteriori attributi **Nome fornitore** e **Indirizzo**.

Poiché nello schema originario gli attributi di prodotto e fornitore compaiono sulla stessa entità, è evidente che se li separiamo in due entità, è opportuno che che tali entità siano correlate da un'associazione binaria, ottenendo lo schema decomposto seguente:

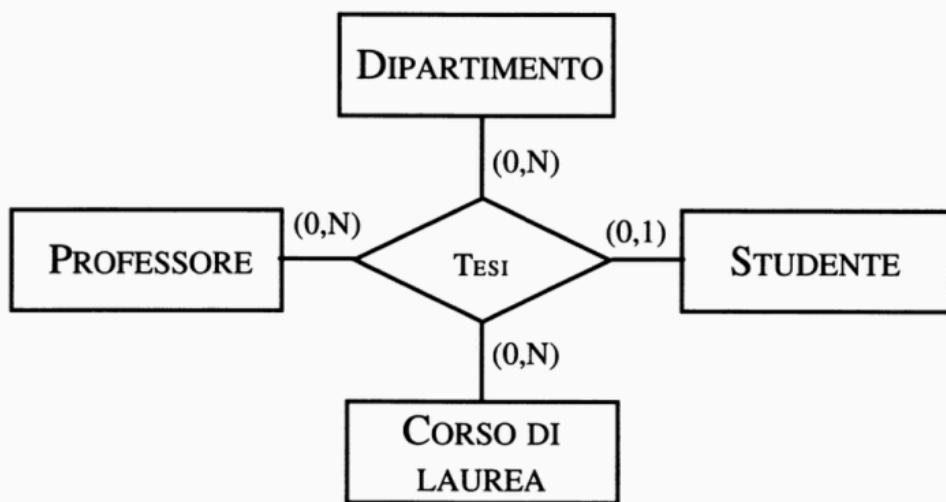


La partecipazione dell'entità PRODOTTO all'associazione deve avere cardinalità massima pari a 1.

Verifiche di normalizzazioni su associazioni

Per quanto riguarda le associazioni, il ragionamento è per certi aspetti più semplice, perché l'insieme delle occorrenze di ciascuna associazione è una relazione, e quindi è possibile applicare direttamente i concetti connessi con le forme normali, ma per altri più complesso, perché i domini su cui tale relazione è definita sono gli insiemi delle occorrenze delle entità coinvolte. Per verificare il soddisfacimento della 3NF, è necessario individuare le dipendenze funzionali che sussistono, nell'ambito dell'associazione in esame, fra le entità coinvolte.

La verifica di normalizzazione su associazioni è solo applicata a relazioni n-arie, come nell'esempio:



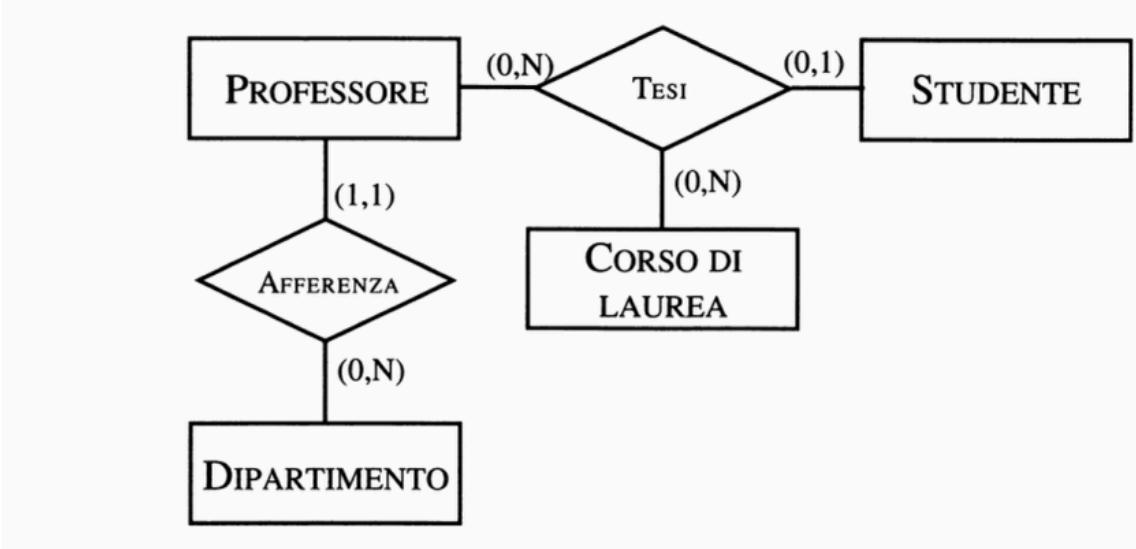
Esaminando tale associazione si conclude che sussistono le dipendenze funzionali

$$\begin{aligned}
 \text{STUDENTE} &\rightarrow \text{CORSO DI LAUREA} \\
 \text{STUDENTE} &\rightarrow \text{PROFESSORE} \\
 \text{PROFESSORE} &\rightarrow \text{DIPARTIMENTO}
 \end{aligned}$$

La chiave unica della relazione risulta essere **STUDENTE**, infatti dato uno studente sono univocamente individuati il corso di laurea, il professore e il dipartimento. La dipendenza **PROFESSORE** → **DIPARTIMENTO** causa la violazione della terza forma normale, infatti l'afferenza di un professore a un dipartimento è un concetto indipendente dall'esistenza di studenti che svolgono la tesi con il professore stesso.

Decomponendo la relazione, quindi separando le dipendenze funzionali con primi membri diversi, si ottengono due associazioni in terza forma normale e due associazioni anche in

forma normale di Boyce e Codd come si vede qui:



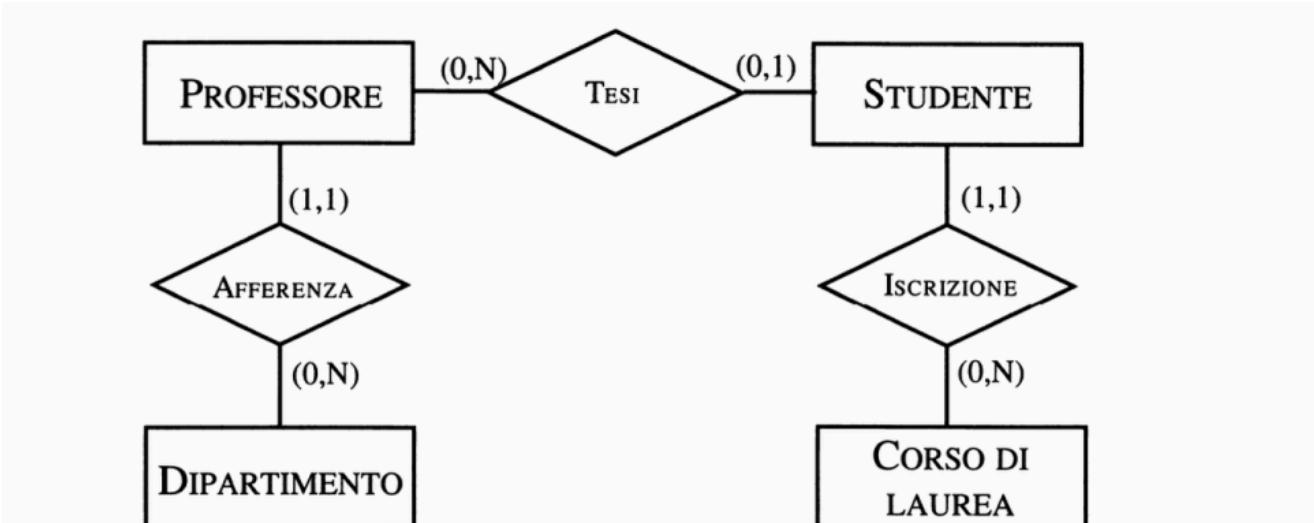
Ulteriori decomposizioni di associazioni

Sullo schema precedente possiamo fare alcune considerazioni aggiuntive che vanno al di là della teoria della normalizzazioni nel senso stretto ma rientrano nell'ambito dell'analisi e verifica degli schemi concettuali per mezzo di strumenti formali (nel nostro caso specifico, dipendenze funzionali).

L'associazione TESI è in terza forma normale, poiché la chiave è STUDENTE e le dipendenze che sussistono, ovvero

$$\begin{aligned} \text{STUDENTE} &\rightarrow \text{PROFESSORE} \\ \text{STUDENTE} &\rightarrow \text{CORSO DI LAUREA} \end{aligned}$$

hanno STUDENTE come primo membro, tuttavia le proprietà descritte dalle due dipendenze sono fra loro indipendenti, è evidente che non tutti gli studenti stanno svolgendo una tesi e quindi non hanno un relatore. Attraverso le dipendenze si nota che sarebbe opportuno decomporre ulteriormente l'associazione TESI ottenendo due associazioni per i due concetti, come in figura:



Siccome è molto raro incontrare delle associazioni con più di tre entità, possiamo affermare che è di solito opportuno decomporre un'associazione ternaria se su di essa è definita una

dipendenza funzionale il cui primo membro è costituito da un'entità e il secondo membro dalle altre due.

Tuttavia se le due entità nel secondo membro della dipendenza sono fra loro strettamente correlate, la decomposizione può risultare non conveniente.

Esercizi sulla normalizzazione

Li trovate sul PDF [5.6 - Esercizi Normalizzazione.pdf](#), presi dalla prof