PROGETTAZIONE DI BASI DI DATI: METODOLOGIE E MODELLI

Progettazione di basi di dati

- è una delle attività del processo di sviluppo dei sistemi informatici
- va quindi inquadrata in un contesto più generale: il *ciclo di vita dei sistemi informatici*:
  - Insieme e sequenzializzazione delle attività svolte da analisti, progettisti, utenti, nello sviluppo e nell'utilizzazione dei sistemi informatici.
  - Si tratta di una attività iterativa (perciò "ciclo")

Fasi del ciclo di vita

una possibile articolazione

- Attività preliminari
- Analisi
- Progettazione
- Realizzazione
- Installazione, transizione, gestione, manutenzione

Al termine (e non solo) di ciascuna fase, deve essere svolta una attività di verifica (validazione) della qualità dei prodotti.

Anche in caso di affidamento all'esterno delle attività di sviluppo, buona parte delle fasi del ciclo di vita rimangono di interesse.

Analisi

Attività volta alla individuazione dei requisiti dell'applicazione, in tutti i dettagli significativi. Le informazioni vengono acquisite attraverso

- interazione con gli utenti (a diversi livelli)
- studio delle realizzazioni esistenti
- studio della normativa

Progettazione

Attività volta alla individuazione delle modalità secondo cui l'applicazione risponderà ai requisiti. In questa fase vengono definiti

- i dati di ingresso e uscita e la loro organizzazione
- l'architettura hardware e software
- la organizzazione dei moduli software

Analisi e progettazione

Analisi definizione di un problema;
"che cosa"
esame delle procedure aziendali
scopo dell'analisi è la **modellazione**, la costruzione di un modello preciso, ma astratto rispetto alle realizzazioni

Progettazione risoluzione di un problema;
"come"
automatizzazione delle procedure

- prodotti fondamentali della fase di analisi sono le rappresentazioni del sistema secondo opportuni formalismi
- poiché in un sistema abbiamo processi (funzioni) e dati, l'analisi deve riguardare entrambi gli aspetti
  - analisi dei dati
  - analisi delle funzioni
- l'analisi delle funzioni può apparire più "naturale" perché fa riferimento alle attività che vengono svolte
- l'analisi dei dati è però essenziale per studiare i flussi informativi (soprattutto interprocesso) ed è per certi aspetti più semplice perché i dati sono più stabili e più facilmente analizzabili in dettaglio

- lo studio di metodologie complete per lo sviluppo di sistemi informativi (e informatici) è oltre gli obiettivi di un corso di basi di dati
- il ruolo centrale dei dati (e quindi delle basi di dati) giustifica uno studio autonomo della progettazione di basi di dati

Metodologia

- articolazione delle attività in fasi, con input e prodotti
- strategie per i vari passi e criteri di scelta
- modelli per la rappresentazione dei prodotti delle varie fasi

Proprietà:

- generalità (rispetto alle applicazioni e ai sistemi)
- qualità del prodotto (correttezza, completezza, efficienza, ...)
- facilità d'uso

Fasi del processo di progettazione di basi di dati

Analisi dei requisiti e progettazione concettuale:
raccolta ed esame delle specifiche;
costruzione di uno **schema concettuale** (descrizione dei dati in un modello astratto) e di un glossario

Progettazione logica: definizione dello **schema logico** della base di dati (descrizione utilizzata dagli utenti e dai programmi)

Progettazione fisica: definizione dei parametri realizzativi di livello più basso

La progettazione di basi di dati segue l'articolazione in livelli dei sistemi, aggiungendo anzi un livello più astratto

- il prodotto fondamentale della prima fase (analisi e progettazione concettuale) è lo **schema concettuale** (con la documentazione associata), cioè uno schema dei dati secondo un modello **concettuale**
- i modelli concettuali vengono spesso utilizzati fin dall'inizio dell'atività di analisi e sono molto utili anche per la comunicazione fra specialisti e non specialisti (analisti e committenti)
- vediamo in dettaglio un modello concettuale

Modello dei dati

- costrutti utilizzati per organizzare i dati di interesse
- componente fondamentale: **meccanismi di strutturazione** (o **costruttori di tipo**)
- ad esempio, il modello relazionale prevede il costruttore **relazione**, che permette di definire insiemi di record omogenei

Le relazioni possono essere rappresentate per mezzo di tabelle

docenza

Corso	*Docente*
Basi di dati	Rossi
Impianti	Neri
Linguaggi	Verdi
...	...

manifesto

CdL	*Materia*	*Anno*
II	Basi di dati	5
II	Impianti	5
II	Linguaggi	4
IE	Linguaggi	5
IE	Impianti	5
...	...	...

In ogni base di dati esistono:

- lo **schema**, sostanzialmente invariante nel tempo, che ne descrive la struttura (aspetto **intensionale**);
  nell'esempio, le intestazioni delle tabelle
- l'**istanza**, costituita dai valori attuali, che possono cambiare molto e molto rapidamente (aspetto **estensionale**);

Due tipi (principali) di modelli:

- **modelli logici**: utilizzati nei DBMS esistenti per l'organizzazione dei dati; ad essi fanno riferimento i programmi; sono indipendenti dalle strutture fisiche;

- **modelli concettuali**: peremettono di rappresentare i dati in modo indipendente da ogni sistema, cercando di descrivere i concetti del mondo reale; sono utilizzati nelle fasi preliminari di progettazione;
  il più noto è il modello **Entity-Relationship**

- i modelli concettuali vengono spesso utilizzati fin dall'inizio dell'attività di analisi e sono molto utili anche per la comunicazione fra specialisti e non specialisti (analisti e committenti)

- i modelli concettuali sono stati introdotti per ovviare al limitato potere espressivo e alla "rigidità" dei modelli logici

- in un modello concettuale (come in ogni modello dei dati), i dati possono essere descritti a due livelli:

  estensionale, relativo ai dati veri e propri, fortemente variabile nel tempo

  intensionale, o dello **schema** relativo alla struttura dei dati, molto più stabile

- in sede di progettazione, interessa (solo) lo schema, in quanto astrazione del livello estensionale

- lo schema concettuale è una rappresentazione delle classi dei dati di interesse e delle loro correlazioni, in modo indipendente da ogni aspetto realizzativo (poi, la realizzazione sarà basata sullo schema concettuale, ma non viceversa)

- si utilizzano rappresentazioni grafiche degli schemi concettuali (che hanno un significato ben preciso, non solo "scatole e frecce")

Il modello E-R (Entity-Relationship)

- il più diffuso modello concettuale
- attenzione: ne esistono molte versioni, (più o meno) diverse l'una dall'altra

I costrutti del modello E-R

- entità
- relationship
- attributi
- identificatori
- generalizzazioni e sottoinsiemi

entità: classe di oggetti (fatti, persone) del frammento di interesse del mondo reale, con proprietà omogenee e con esistenza "autonoma"; es. persone, città, aziende, fatture, ordini

relationship: legame logico fra due o più entità, di interesse per l'applicazione; classe di fatti di interesse

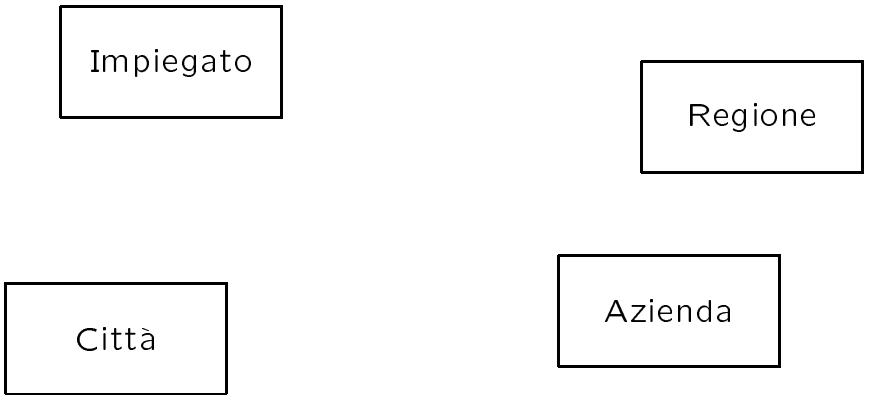
entità: la classe di oggetti omogenei (e il loro tipo)

occorrenza (o **istanza**) **di entità**: ciascun elemento della classe

- nello schema concettuale rappresentiamo le entità, non le singole istanze ("astrazione")

- ogni entità ha un **nome** che la identifica univocamente nello schema
  - nomi espressivi
  - opportune convenzioni (ad esempio, sempre singolare)

Entità: rappresentazione grafica



relationship legame logico fra due o più entità, di interesse per l'applicazione; classe di fatti di interesse

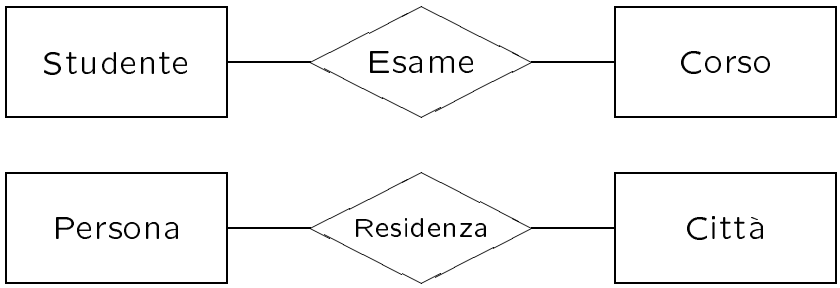
- spesso non tradotto per evitare la confusione con relazione (traduzione di *relation*)
- anche tradotto con **correlazione** o **associazione**

occorrenza (o **istanza**) **di relationship**:
n-upla (coppia, terna, ...) di istanze di entità, una per ciascuna entità coinvolta

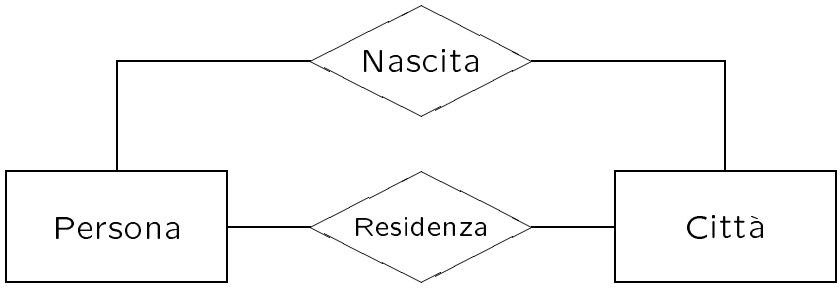
relationship: insieme di n-uple omogenee per

- tipo (entità di provenienza)
- significato (rispetto all'applicazione)

Relationship: rappresentazione grafica



- due entità possono essere coinvolte in più relationship

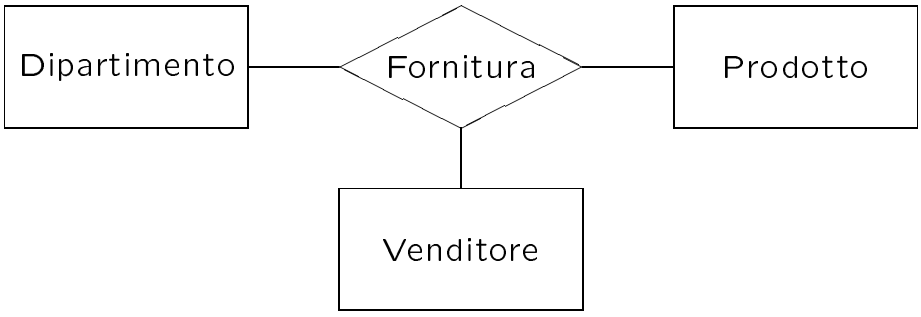


un dettaglio formale:

- le istanze di una relationship formano un **insieme**: non ci possono essere elementi ripetuti
- si tratta di una **relazione** nel senso matematico del termine: (sottoinsieme del prodotto cartesiano degli insiemi di istanze coinvolte)

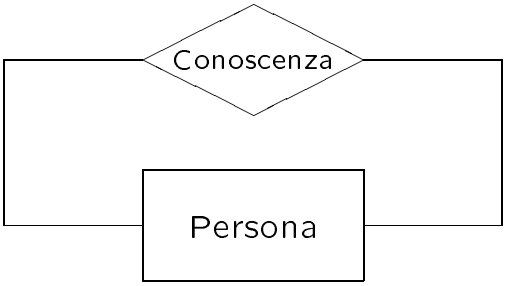
?

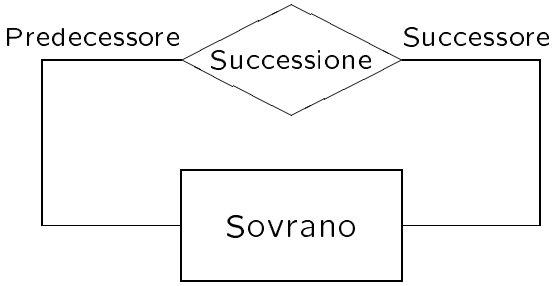
- le relationship possono coinvolgere più di due entità



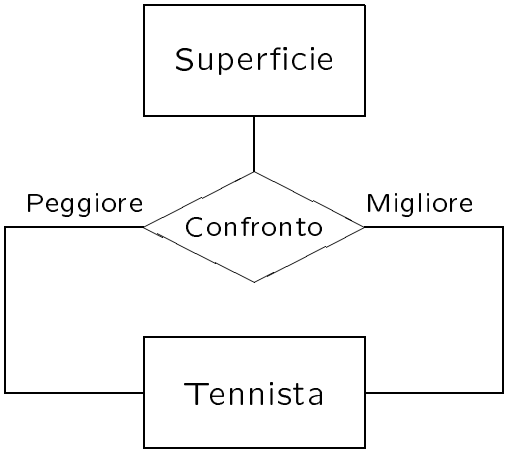
A	fornisce	stampanti	al	Dip	Personale
B	"	fotocopiatrici	"	"	Personale
A	"	calcolatori	"	"	Ricerca
C	"	calcolatori	"	"	Personale

- una relationship può coinvolgere "due volte" la stessa entità (relationship **ricorsiva**)
- può essere necessario distinguere "ruoli"





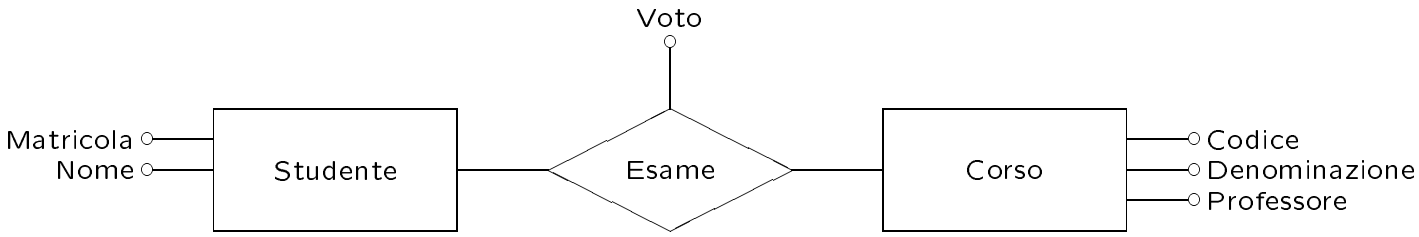
- possono esistere relationship ternarie ricorsive



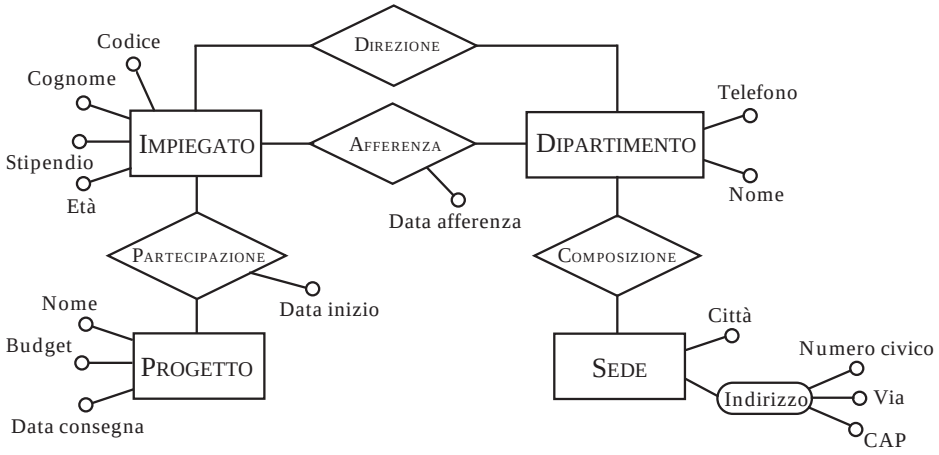
Migliore	Peggiore	Superficie
Sanchez	Navratilova	Terra
Navratilova	Sanchez	Erba
Graf	Sanchez	Erba

attributo: proprietà elementare di un'entità o una relationship, di interesse ai fini dell'applicazione

- un attributo associa un valore a ciascuna occorrenza dell'entità (o relationship) su cui è definito



Uno schema E-R



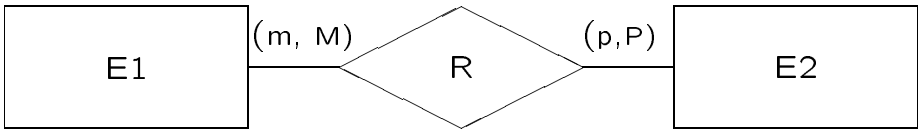
Altri costrutti e dettagli del modello:

- cardinalità delle relationship
- identificatori
- gerarchie

Cardinalità delle relationship

- permettono di specificare il numero minimo e massimo di occorrenze della relationship cui ciascuna occorrenza di una entità deve/può partecipare
- per semplicità usiamo:
  - 0 e 1 per la cardinalità minima:
    0 = "è opzionale"; 1 = "è obbligatoria"
  - 1 e "N" per la massima:
    "N" non pone alcun limite
- con riferimento alle cardinalità massime, abbiamo relationship: uno a uno, uno a molti, molti a molti

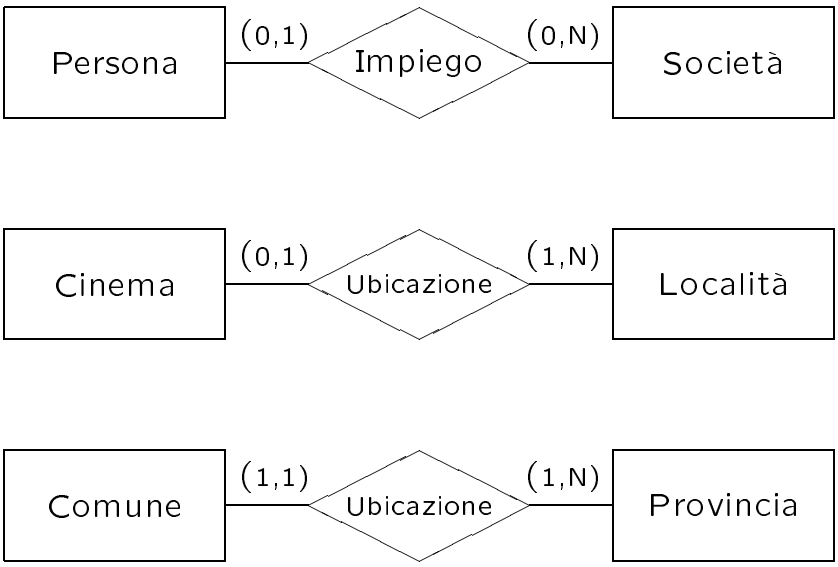
- le cardinalità vanno specificate per ciascuna entità che partecipa alla relationship



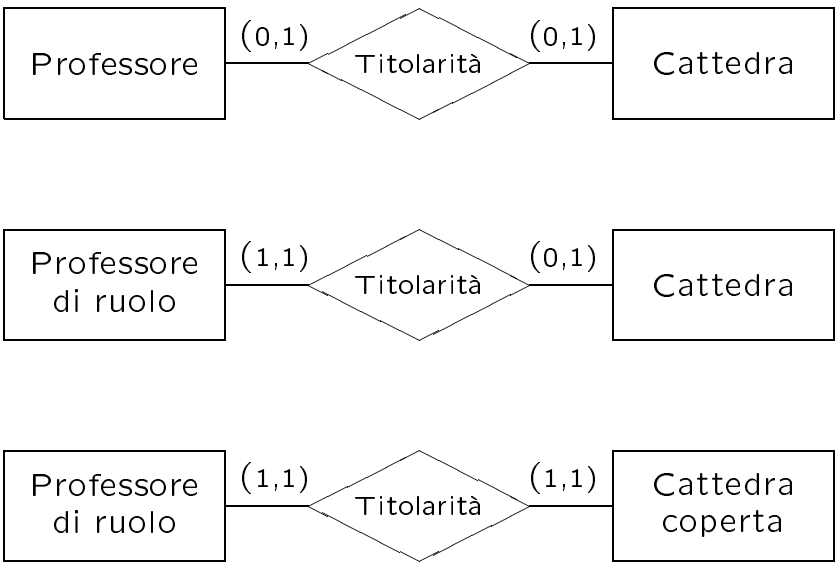
- ogni occorrenza di E1 partecipa ad almeno m e al più M occorrenze di R
- ogni occorrenza di E2 partecipa ad almeno p e al più P occorrenze di R

Relationship "molti a molti"

Relationship “uno a molti”



Relationship “uno a uno”



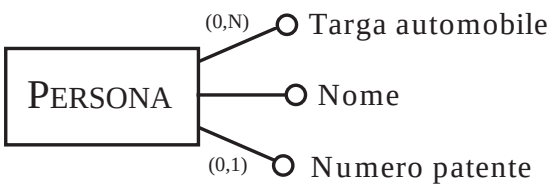
Due avvertenze:

- attenzione al "verso" nelle relationship uno a molti
- le relationship obbligatorie-obbligatorie spesso hanno poco senso

Cardinalità di attributi

Possono essere usate per:

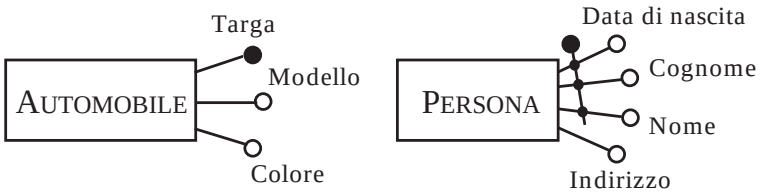
- indicare opzionalità (molto utili, corrispondono ai valori nulli)
- per indicare attributi multivalore (non sempre utilizzati)



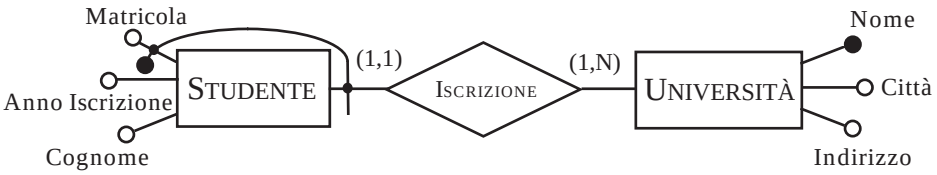
Identificatore di una entità

- "strumento" per l'identificazione univoca delle occorrenze
- è costituito da
  - attributi (uno o più) dell'entità: identificatore **interno**
  - (da attributi e) entità esterne (attraverso relationship): identificatore **esterno**
- ogni entità deve avere (almeno) un identificatore

Identificatori interni

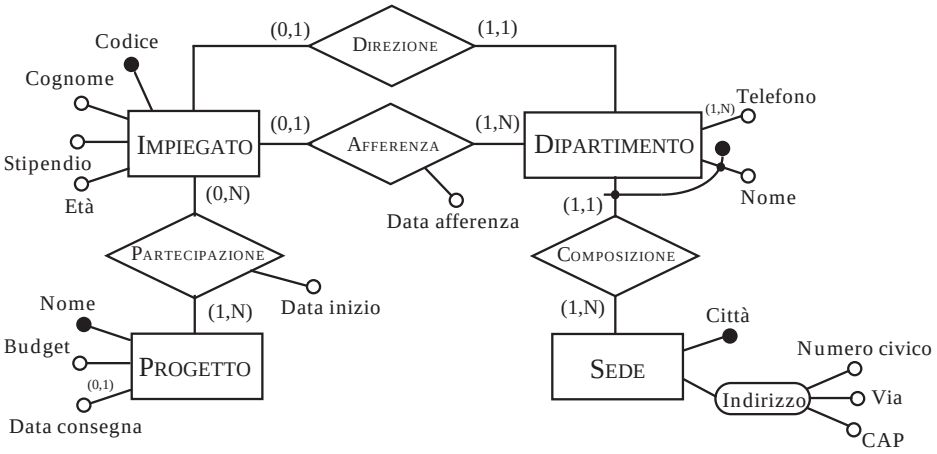


Identificatore esterno



- perchè non parliamo degli identificatori delle relationship?

Uno schema con attributi e identificatori



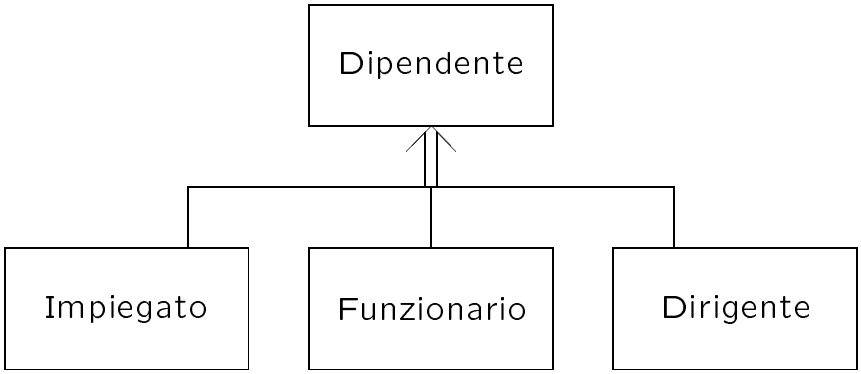
Generalizzazione

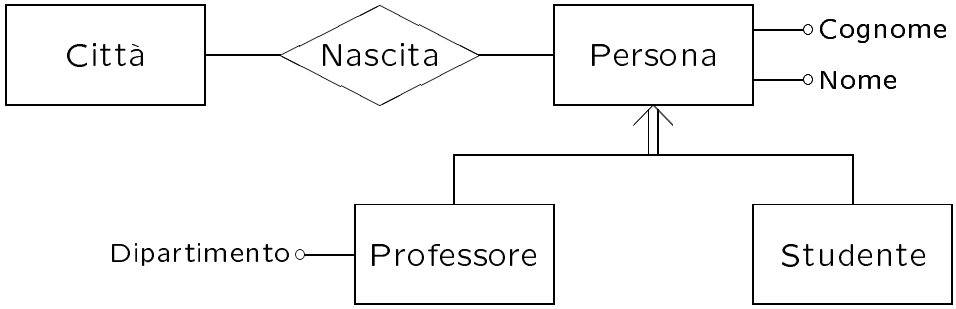
- uomo e donna sono i due casi particolari di persona
- dipendente è un concetto che generalizza i concetti di funzionario, impiegato, dirigente

Generalizzazione

- mette in relazione una o più entità $\mathbf{E}_1$, $\mathbf{E}_2$, ..., $\mathbf{E}_n$ con un'altra entità **E**, di esse più generale, nel senso che le comprende come caso particolare
- ogni proprietà di **E** è significativa per $\mathbf{E}_1$, $\mathbf{E}_2$, ..., $\mathbf{E}_n$
- ogni occorrenza di $\mathbf{E}_i$ è occorrenza anche di **E**
- ogni occorrenza di **E** è occorrenza di al più una $\mathbf{E}_i$

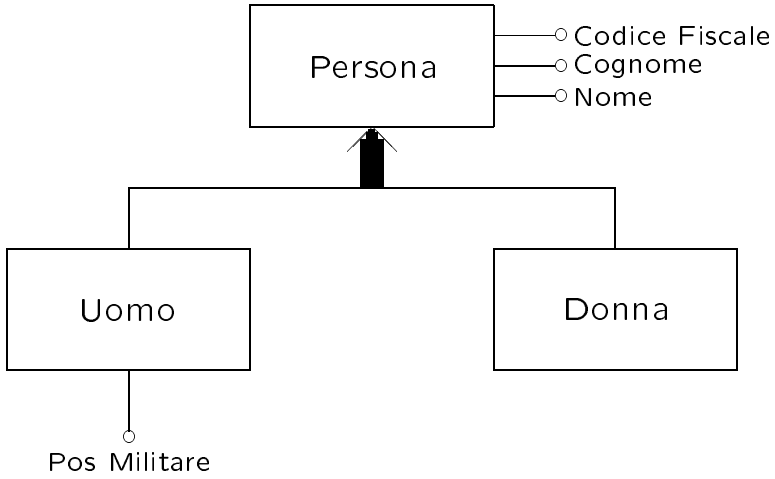
- $\mathbf{E}_1$, $\mathbf{E}_2$, ..., $\mathbf{E}_n$ sono **specializzazioni** (**sottotipi**, **sottoinsiemi**) di **E**
- le proprietà (attributi, relationship, altre generalizzazioni) di **E** vengono **ereditate** da $\mathbf{E}_1$, $\mathbf{E}_2$, ..., $\mathbf{E}_n$ (e non rappresentate esplicitamente)



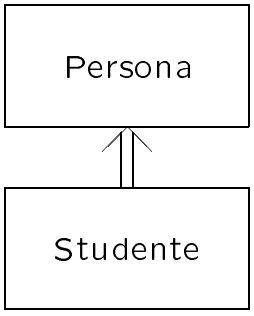


- ogni professore è una persona
- ogni studente è una persona
- nessun professore è studente e viceversa
- esistono persone che non sono né studenti né professori
- ogni studente ha nome, cognome e città di nascita (ereditati)

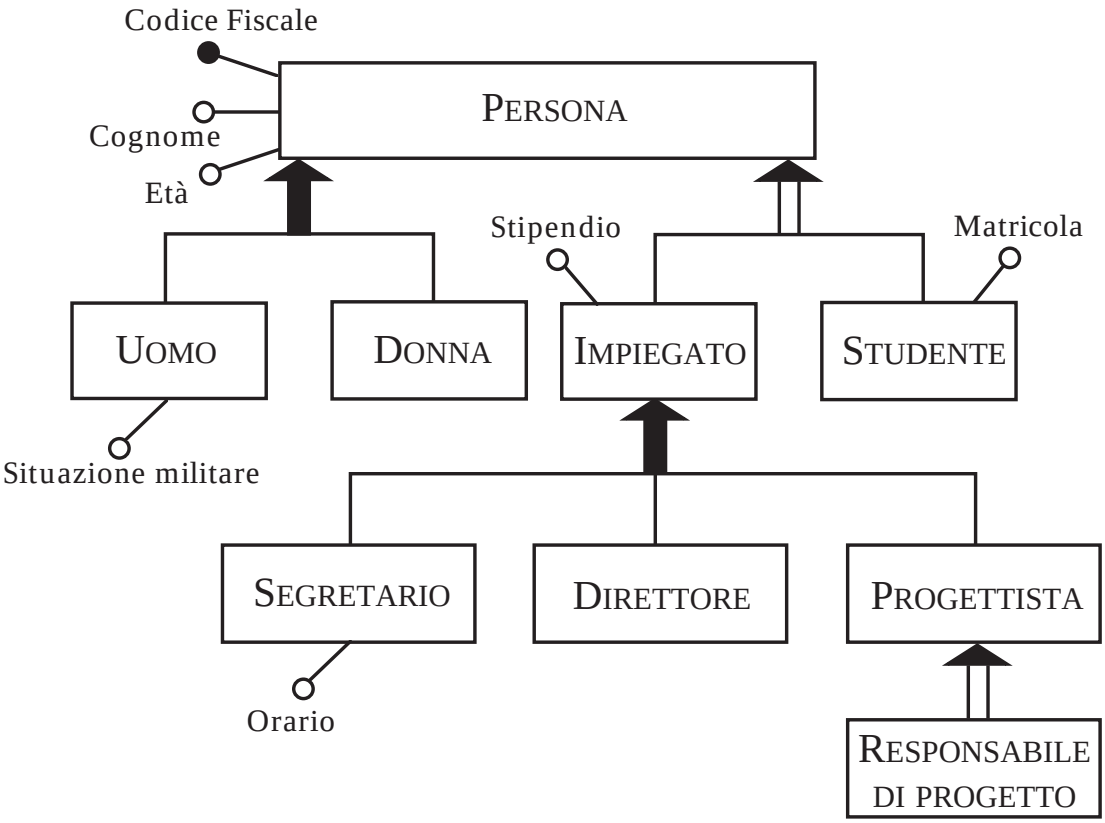
- una generalizzazione è **completa** se ogni occorrenza del "genitore" è occorrenza di uno dei figli



- nel caso di generalizzazione con un solo figlio si parla di **sottoinsieme** (o **gerarchia is-a**)
- ovviamente, non si può avere completezza



- possono esistere gerarchie a più livelli e multiple gerarchie allo stesso livello
- un'entità può essere inclusa in più gerarchie, come genitore e/o come figlio
- alcune configurazioni non hanno senso
- il genitore di una gerarchia completa può non avere identificatore (purché ce l'abbiano tutte le entità figlie)



Documentazione associata

- dizionario dei dati (entità, relationship, attributi, gerarchie)
- vincoli non esprimibili

Entità	Descrizione	Attributi	Identificatore
Impiegato	Impiegato che lavora nella azienda	Codice, Cognome, Stipendio, Età	Codice
Progetto	Progetti ai quali lavorano gli impiegati della azienda	Nome, Budget, Data consegna	Nome
Dipartimento	Dipartimenti delle sedi della azienda	Telefono, Nome	Nome, Sede
Sede	Sede della azienda in una certa città	Città, Indirizzo	Città

Relationship	Descrizione	Entità Coinvolte	Attributi
Direzione	Associa un dipartimento al suo direttore	Impiegato, Dipartimento	
Afferenza	Associa un impiegato al suo dipartimento	Impiegato, Dipartimento	Data afferenza
Partecipazione	Associa agli impiegati i progetti sui quali lavorano	Impiegato, Progetto	Data inizio
Composizione	Associa una sede ai dipartimenti di cui è composta	Dipartimento, Sede	

Vincoli di integrità sui dati
(1) Un impiegato può essere direttore solo del dipartimento a cui afferisce.
(2) Un impiegato non può avere uno stipendio maggiore del direttore del dipartimento al quale afferisce.
(3) Un impiegato non può partecipare a un progetto se non afferisce a nessun dipartimento.
(4) Il budget di un progetto deve essere superiore alla somma degli stipendi degli impiegati che vi partecipano.