

Atzeni, Ceri, Paraboschi, Torlone

Basi di dati

McGraw-Hill, 1996-2002

Capitolo 6:

**Progettazione di basi di dati:
Metodologie e modelli**

24/3/2003

**modificato per il corso di
Laboratorio di Sistemi Informativi
Università di Chieti-Pescara**

Progettazione di basi di dati

A partire dai requisiti, consente di definire **struttura, caratteristiche e contenuto** della base di dati.

Progettazione di basi di dati

- È una delle attività del processo di sviluppo dei sistemi informativi
- va quindi inquadrata in un contesto più generale:
- **il ciclo di vita dei sistemi informativi:**
 - Insieme e sequenzializzazione delle attività svolte da analisti, progettisti, utenti, nello sviluppo e nell'uso dei sistemi informativi
 - attività iterativa, quindi **ciclo**



Fasi (tecniche) del ciclo di vita

- **Studio di fattibilità: definizione costi e priorità**
- **Raccolta e analisi dei requisiti: studio delle proprietà del sistema**
- **Progettazione: di dati e funzioni**
- **Realizzazione**
- **Validazione e collaudo: sperimentazione**
- **Funzionamento: il sistema diventa operativo**

La progettazione di un sistema informativo riguarda due aspetti:

- ▶ progettazione dei dati
- progettazione delle applicazioni

Ma:

- ▶ i dati hanno un ruolo centrale
 - i dati sono più stabili

- Per garantire prodotti di buona qualità è opportuno seguire una
 - **metodologia di progetto**, con:
 - articolazione delle attività in fasi
 - criteri di scelta
 - modelli di rappresentazione
 - generalità e facilità d'uso



Requisiti della base di dati

**Progettazione
concettuale**

Schema concettuale

**Progettazione
logica**

Schema logico

**Progettazione
fisica**

Schema fisico

Progettazione concettuale

Rappresentare le specifiche **informali** della realtà di interesse tramite una descrizione **formale** completa ma indipendente dal sistema usato per rappresentare i dati.

Progettazione logica

Tradurre lo schema concettuale secondo il **modello logico** del DBMS. Nel caso di sistema relazionale, lo schema logico che si ottiene è l'insieme delle tabelle e dei vincoli relativi.

Questa fase è indipendente dal DBMS scelto, ma dipende dal modello logico dei dati.

Progettazione fisica

Completare lo schema logico con la specifica dei parametri fisici di memorizzazione (ad esempio con l'elenco degli indici da utilizzare).

Questa fase dipende dal DBMS scelto.

Schemi e Modelli dei Dati

Ognuno degli tre schemi prodotti, è basato su un sottostante **modello dei dati**. Abbiamo un modello **concettuale**, un modello **logico** e un modello **fisico**.

Modello dei dati

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

Schemi e istanze

- In ogni livello di astrazione (concettuale, logico, fisico) esistono:
 - lo **schema**, sostanzialmente invariante nel tempo, che ne descrive la struttura (aspetto intensionale)
 - nel modello relazionale, le intestazioni delle tabelle
 - l'**istanza**, i valori attuali, che possono cambiare anche molto rapidamente (aspetto estensionale)
 - nel modello relazionale, il “corpo” di ciascuna tabella

Due tipi (principali) di modelli

- **modelli logici**: utilizzati nei DBMS esistenti per l'organizzazione dei dati
 - utilizzati dai programmi
 - indipendenti dalle strutture fisicheesempi: **relazionale**, reticolare, gerarchico, a oggetti
- **modelli concettuali**: permettono di rappresentare i dati in modo indipendente da ogni sistema
 - cercano di descrivere i concetti del mondo reale
 - sono utilizzati nelle fasi preliminari di progettazioneil più noto è il modello **Entity-Relationship**

Modelli concettuali, perché?

- Proviamo a modellare una applicazione definendo direttamente lo schema logico della base di dati:
 - da dove cominciamo?
 - rischiamo di perderci subito nei dettagli
 - dobbiamo pensare subito a come correlare le varie tabelle (chiavi etc.)
 - i modelli logici sono rigidi

Modelli concettuali, perché?

- **servono per ragionare sulla realtà di interesse, indipendentemente dagli aspetti realizzativi**
- **permettono di rappresentare le classi di dati di interesse e le loro correlazioni**
- **prevedono efficaci rappresentazioni grafiche (utili anche per documentazione e comunicazione)**

Modello Entity-Relationship (Entità-Relazione)

- Il più diffuso modello concettuale
 - Ne esistono molte versioni,
 - (più o meno) diverse l'una dall'altra

I costrutti del modello E-R

- Entità
- Relationship
- Attributo
- Identificatore
- Generalizzazione
-

Entità

- **Classe di oggetti (fatti, persone, cose) della applicazione di interesse con proprietà comuni e con esistenza “autonoma”**
- **Esempi:**
 - **impiegato, città, conto corrente, ordine, fattura**

Relationship

- **Legame logico fra due o più entità, rilevante nell'applicazione di interesse**
- **Esempi:**
 - **Residenza (fra persona e città)**
 - **Esame (fra studente e corso)**

Uno schema E-R, graficamente



Entità

- **Classe di oggetti (fatti, persone, cose) della applicazione di interesse con proprietà comuni e con esistenza “autonoma”**
- **Esempi:**
 - **impiegato, città, conto corrente, ordine, fattura**

Entità: schema e istanza

- **Entità:**
 - classe di oggetti, persone, ... "omogenei"
- **Occorrenza** (o istanza) di entità:
 - elemento della classe (l'oggetto, la persona, ..., non i dati)
- nello schema concettuale rappresentiamo le **entità**, non le singole istanze ("astrazione")

Rappresentazione grafica di entità

Impiegato

Dipartimento

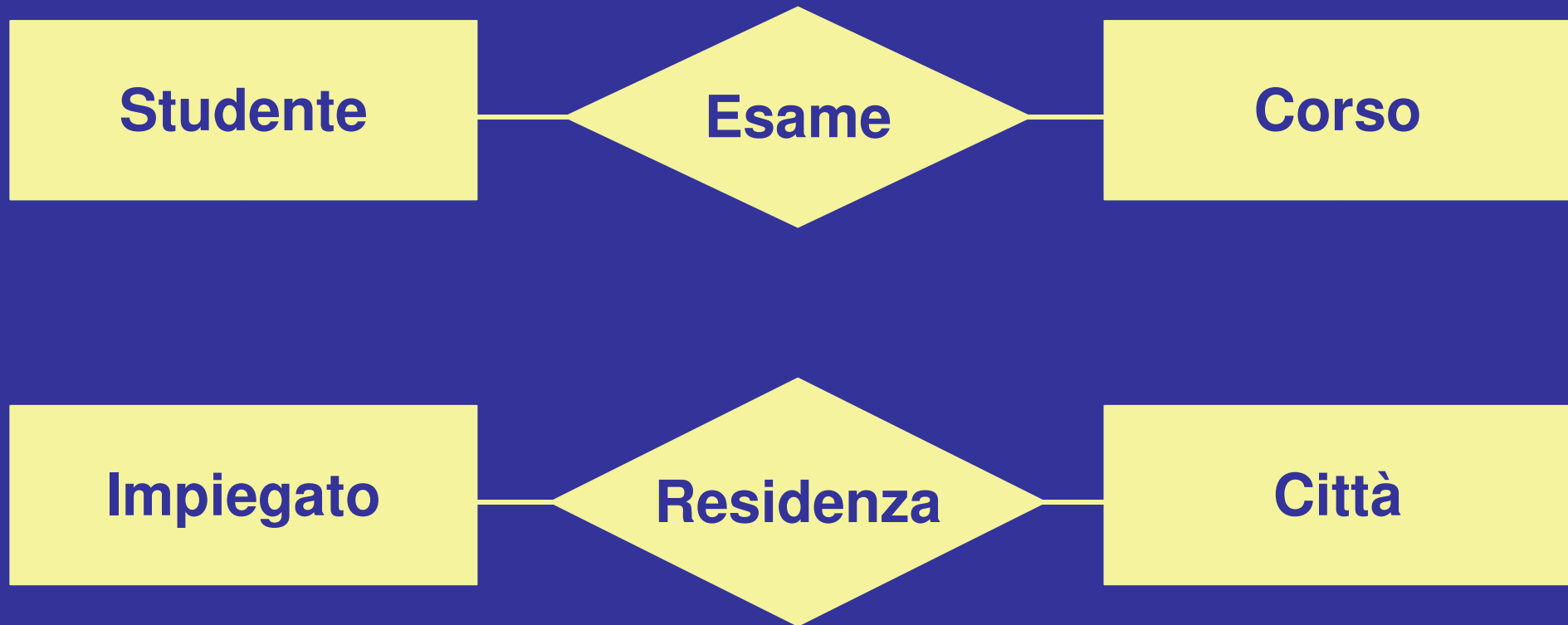
Città

Vendita

Relationship

- **Legame logico fra due o più entità, rilevante nell'applicazione di interesse**
- **Esempi:**
 - **Residenza (fra persona e città)**
 - **Esame (fra studente e corso)**
- **Chiamata anche:**
 - **relazione, correlazione, associazione**

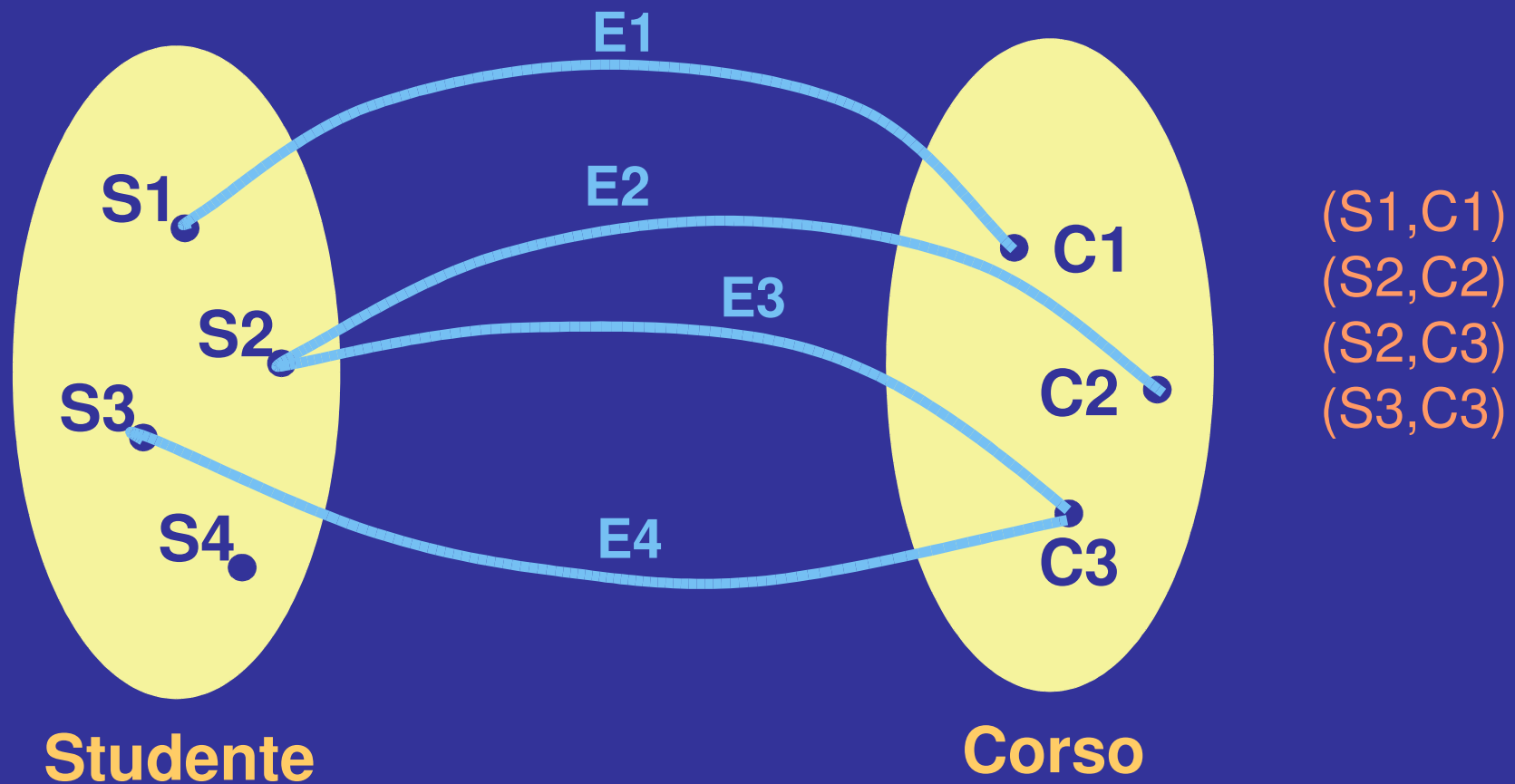
Rappresentazione grafica di relationship



Relationship, occorrenze

- Una occorrenza di una relationship binaria è coppia di occorrenze di entità, una per ciascuna entità coinvolta
- Una occorrenza di una relationship n-aria è una n-upla di occorrenze di entità, una per ciascuna entità coinvolta
- Nell'ambito di una relationship non ci possono essere occorrenze (coppie, ennuple) ripetute

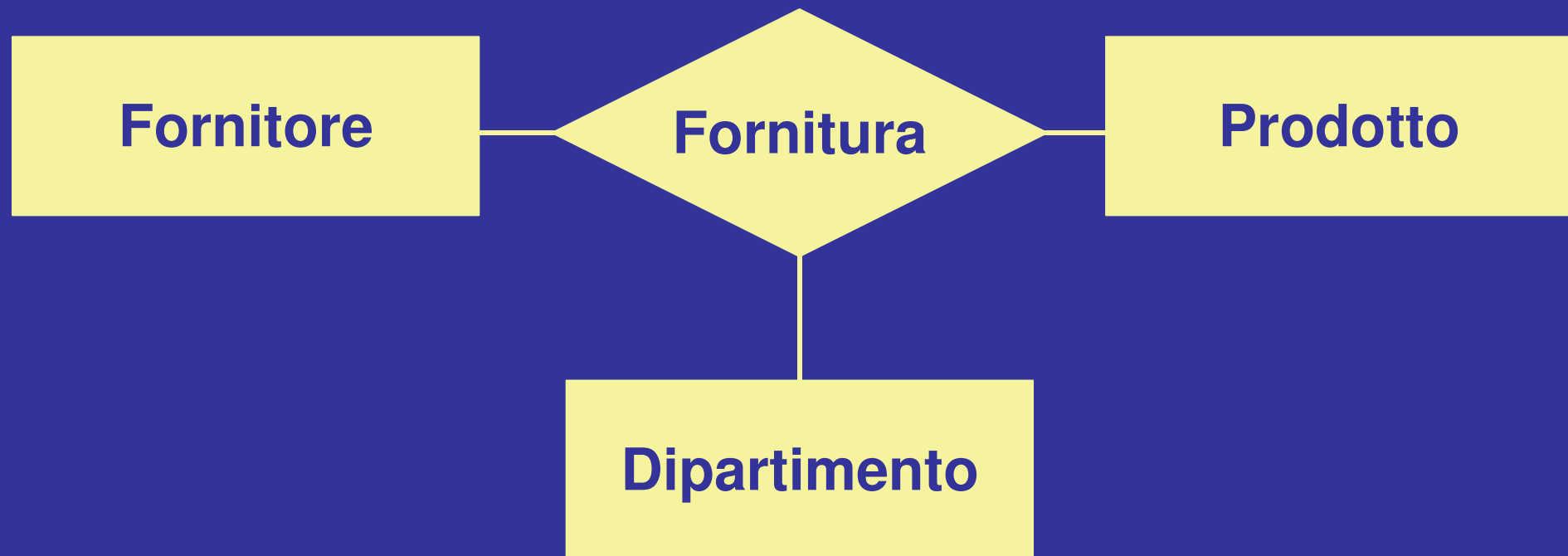
Esempi di occorrenze



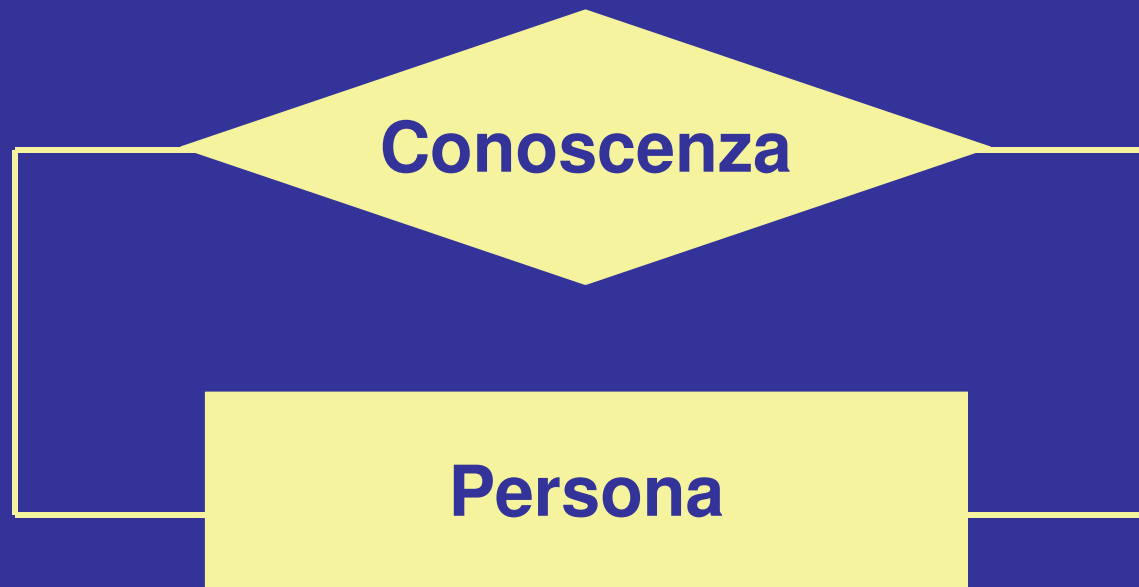
Due relationship sulle stesse entità



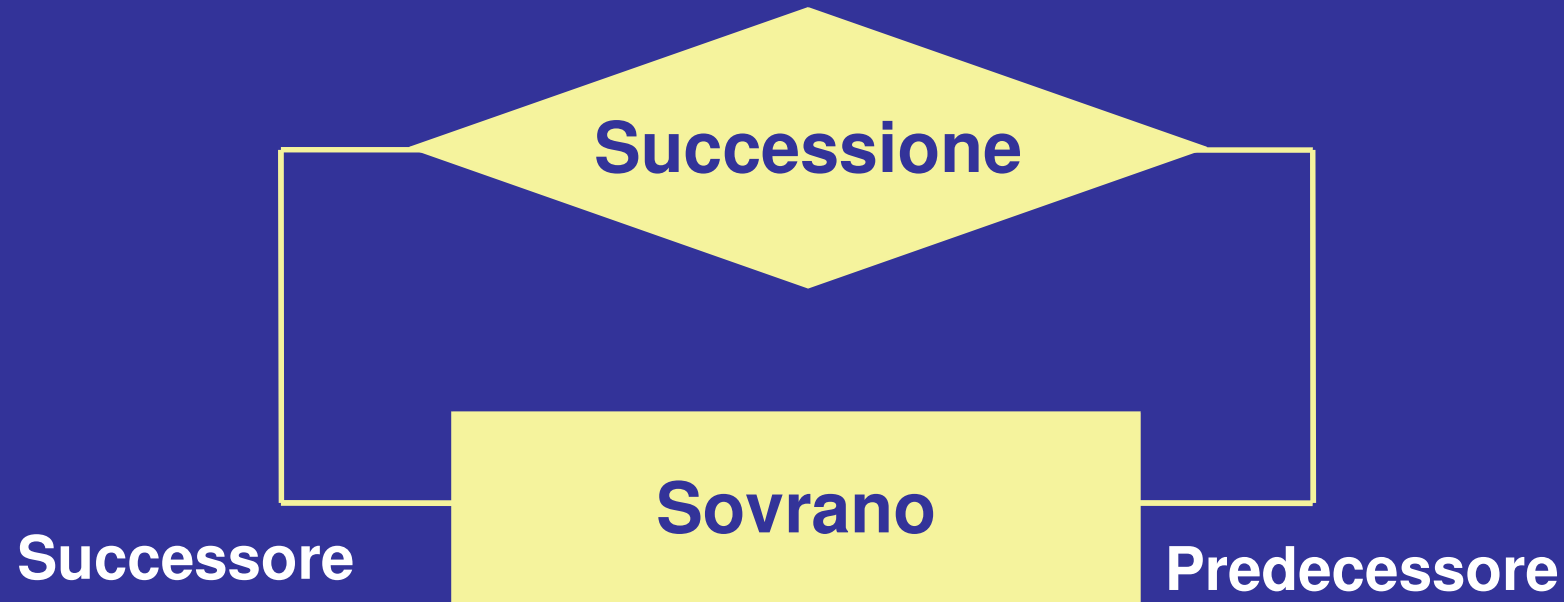
Relationship n-aria



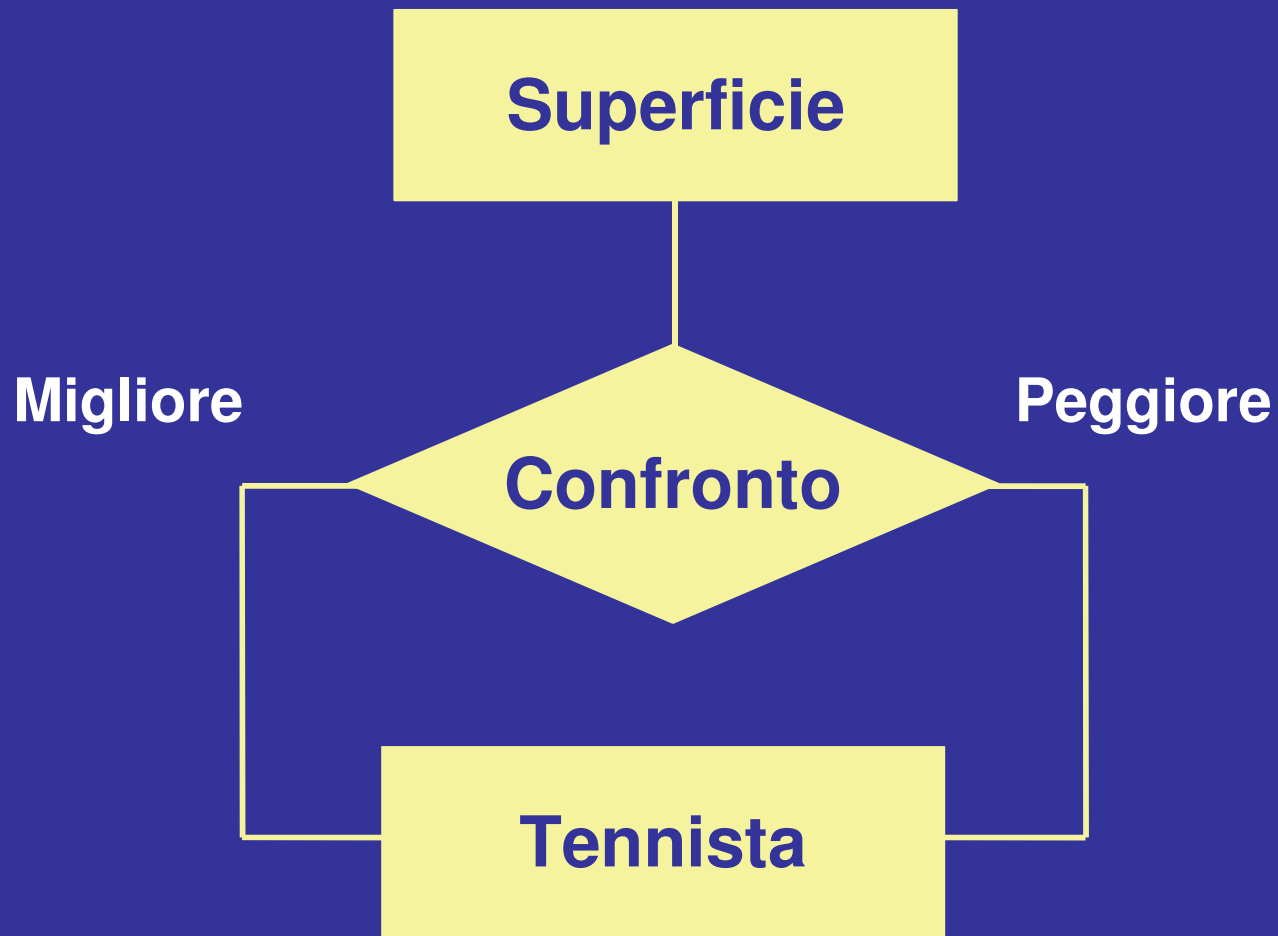
Relationship ricorsiva: coinvolge “due volte” la stessa entità



Relationship ricorsiva con “ruoli”



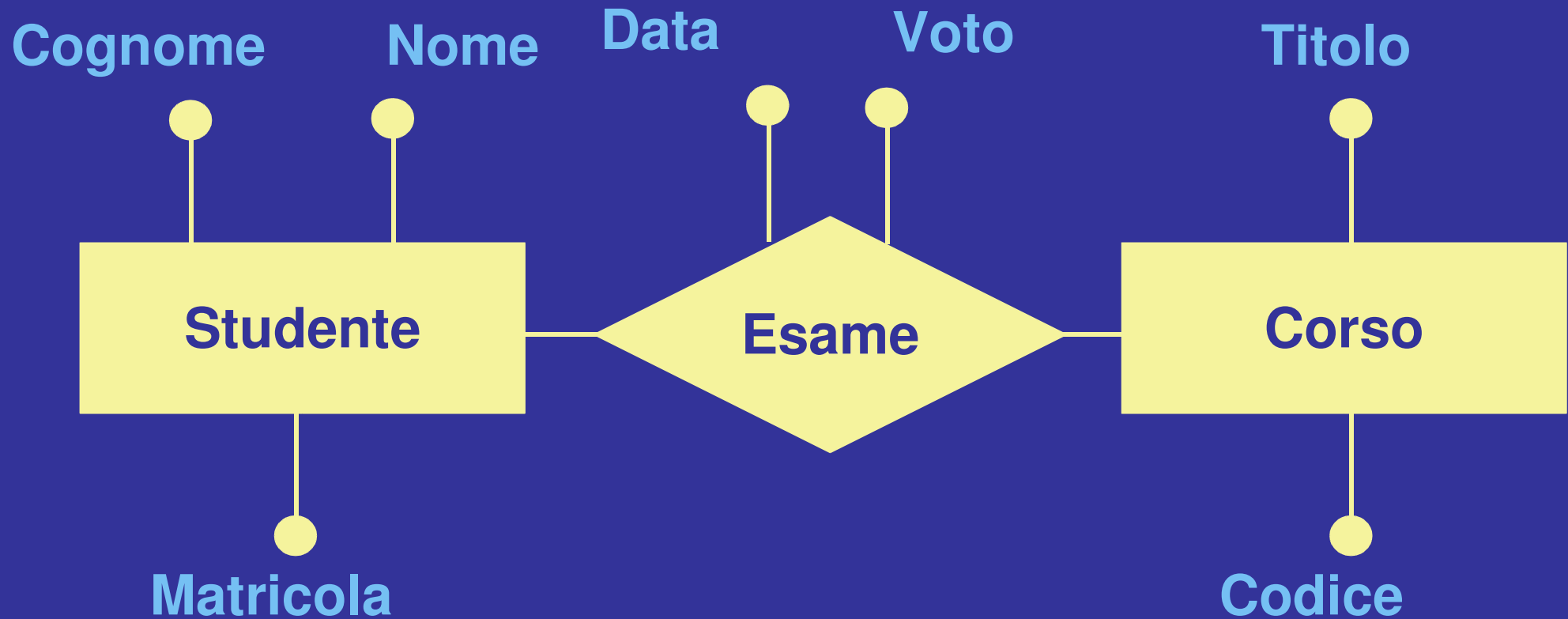
Relationship ternaria ricorsiva



Attributo

- **Proprietà elementare di un'entità o di una relationship, di interesse ai fini dell'applicazione**
- **Associa ad ogni occorrenza di entità o relationship un valore appartenente a un insieme detto **dominio** dell'attributo**

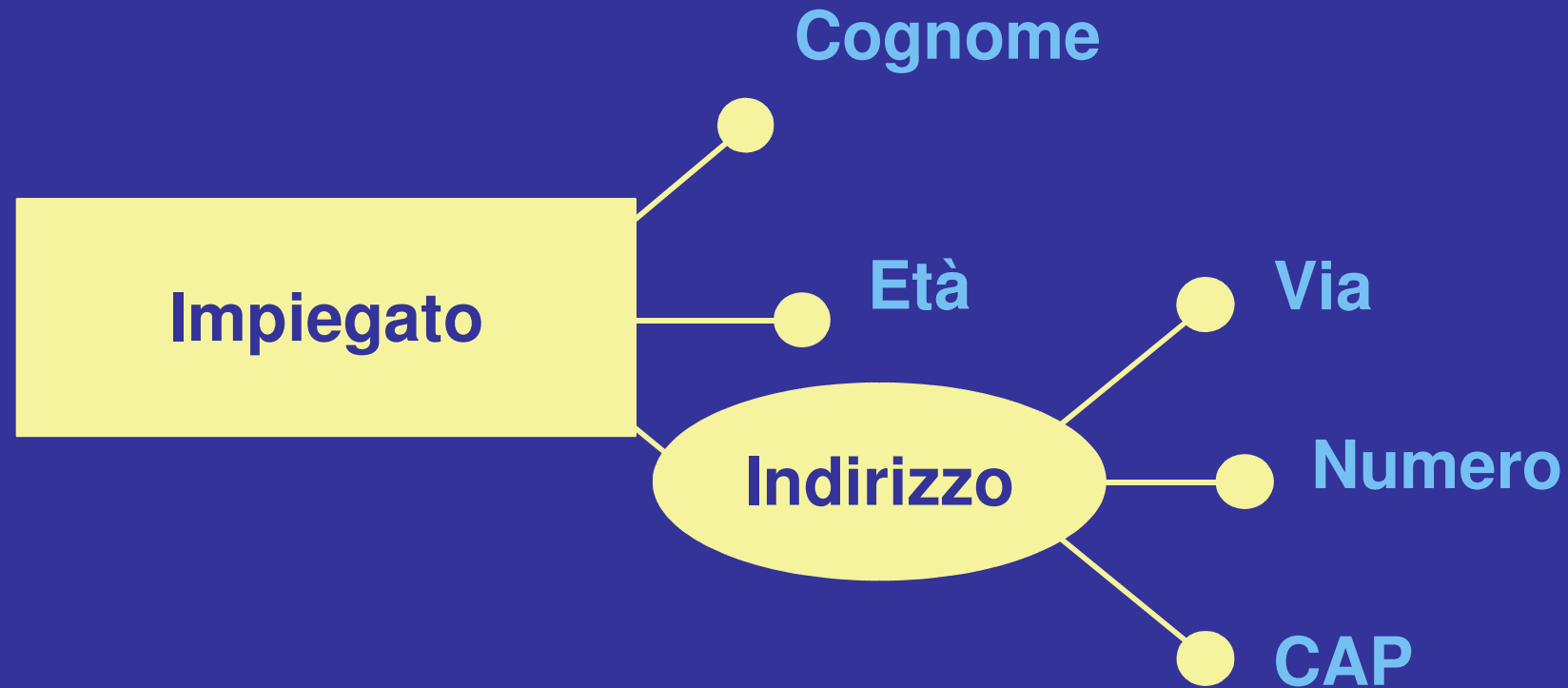
Attributi, rappresentazione grafica



Attributi composti

- Raggruppano attributi di una medesima entità o relationship che presentano affinità nel loro significato o uso
- Esempio:
 - **Via, Numero civico e CAP** formano un **Indirizzo**

Rappresentazione grafica



Cognome

Telefono



Altri costrutti del modello E-R

- **Cardinalità**
 - di relationship
 - di attributo
- **Identificatore**
 - interno
 - esterno
- **Generalizzazione**

Cardinalità di relationship

- Coppia di valori associati a ogni entità che partecipa a una relationship
- specificano il numero minimo e massimo di occorrenze delle relationship cui ciascuna occorrenza di una entità può partecipare

Esempio di cardinalità



- per semplicità usiamo solo tre simboli:
- 0 e 1 per la cardinalità minima:
 - 0 = “partecipazione **opzionale**”
 - 1 = “partecipazione **obbligatoria**”
- 1 e “N” per la massima:
 - “N” non pone alcun limite

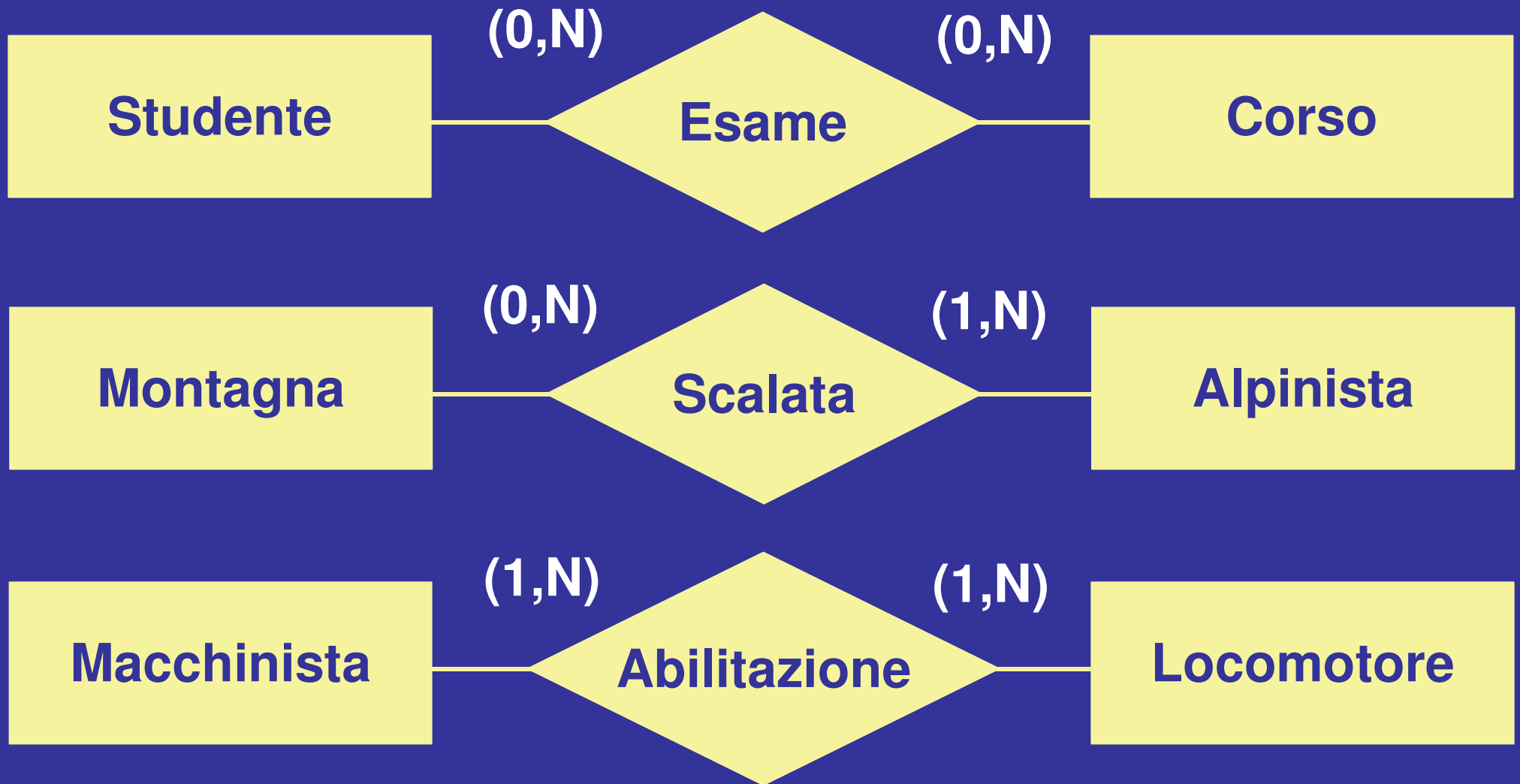
Cardinalità di Residenza



Tipi di relationship

- Con riferimento alle cardinalità **massime**, abbiamo relationship:
 - **uno a uno**
 - **uno a molti**
 - **molti a molti**

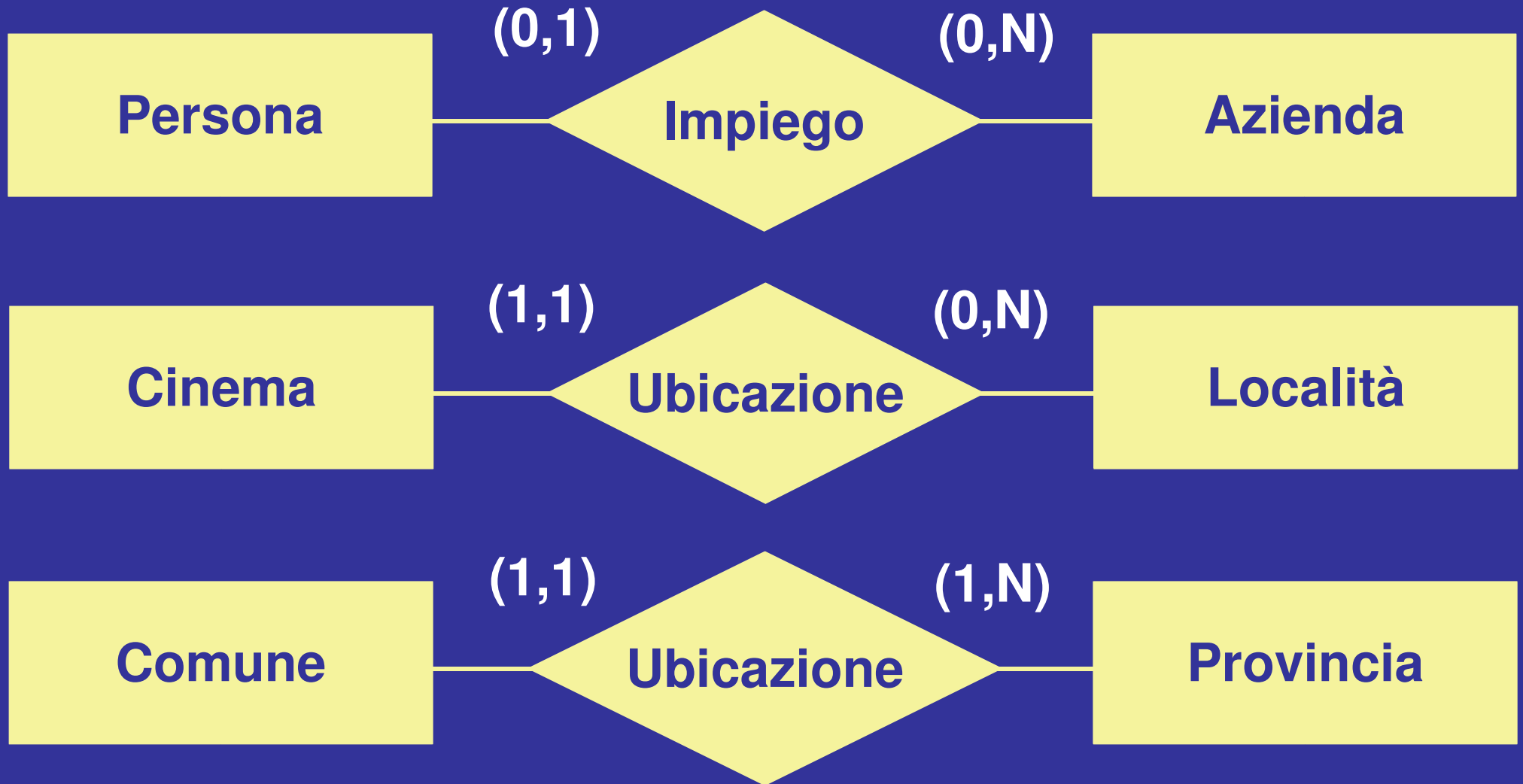
Relationship “molti a molti”



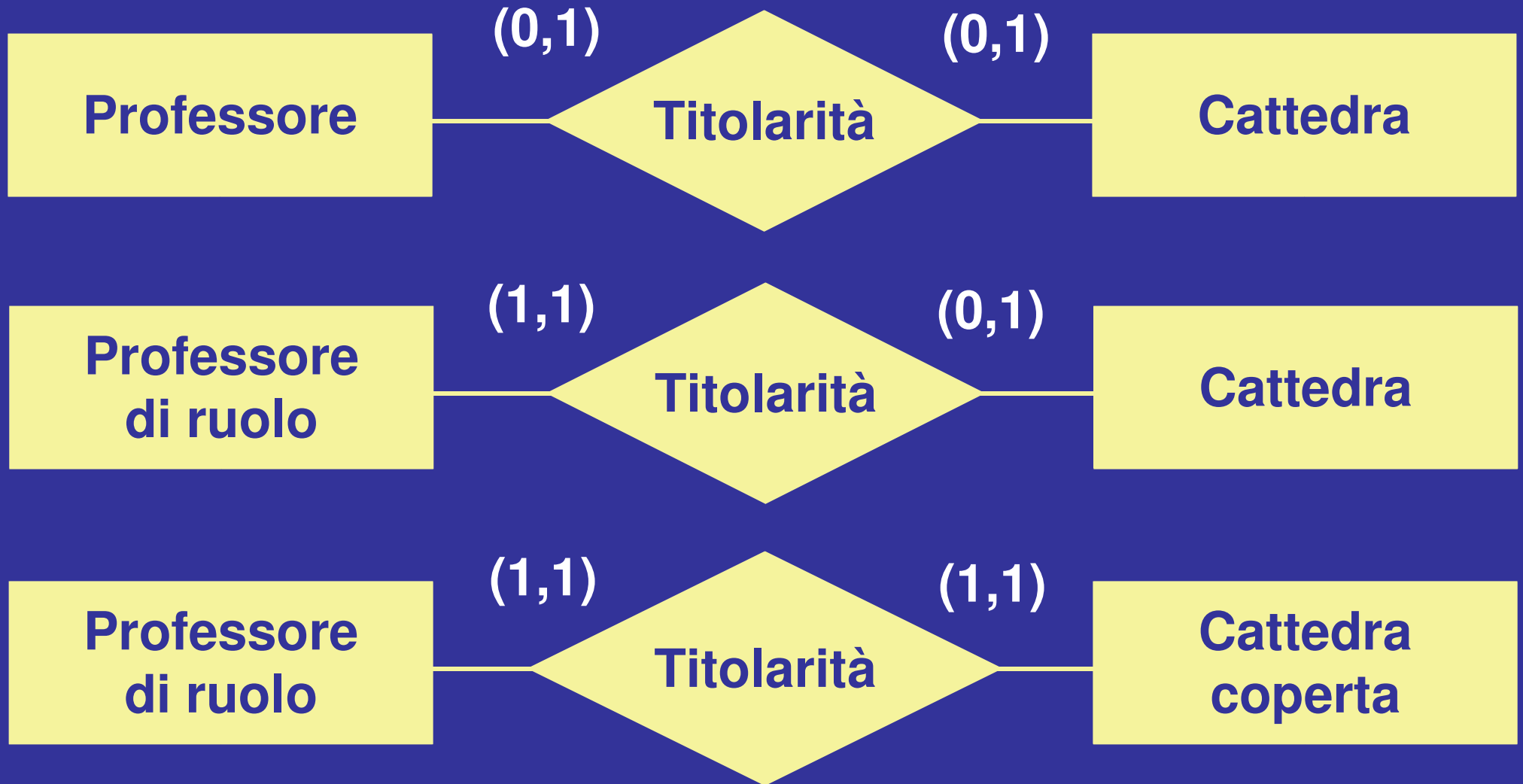
Due avvertenze

- **Attenzione al "verso" nelle relationship uno a molti**
- **le relationship obbligatorie-obbligatorie sono molto rare**

Relationship “uno a molti”



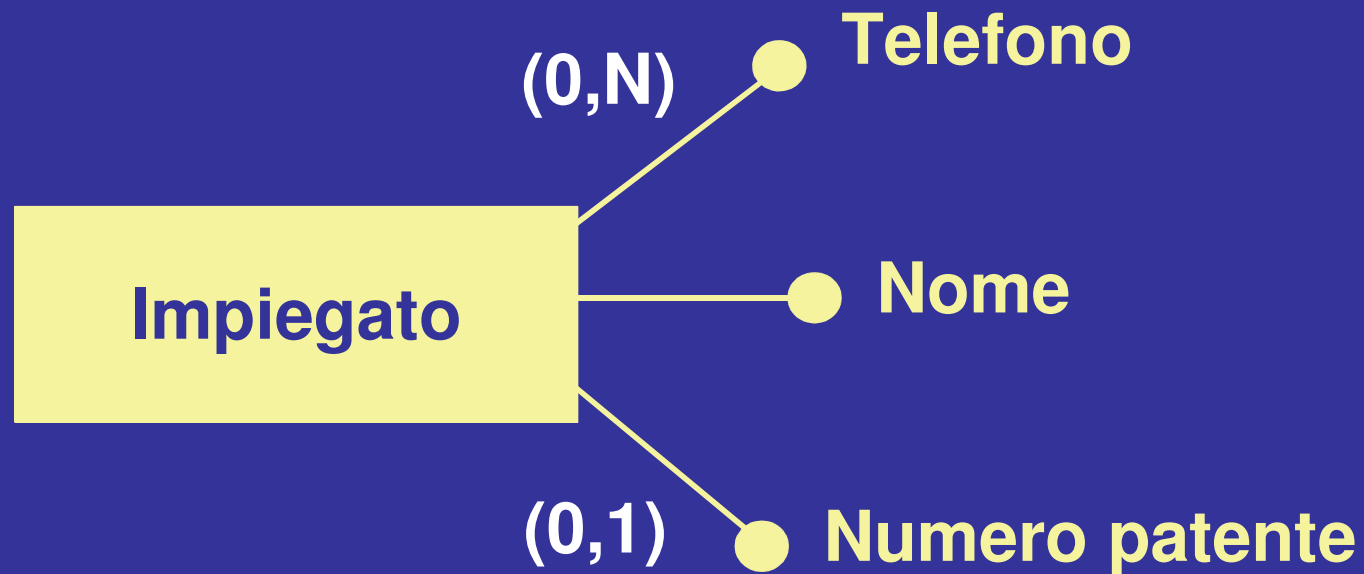
Relationship “uno a uno”



Cardinalità di attributi

- E' possibile associare delle cardinalità anche agli attributi, con due scopi:
 - indicare opzionalità ("informazione incompleta")
 - indicare attributi multivalore

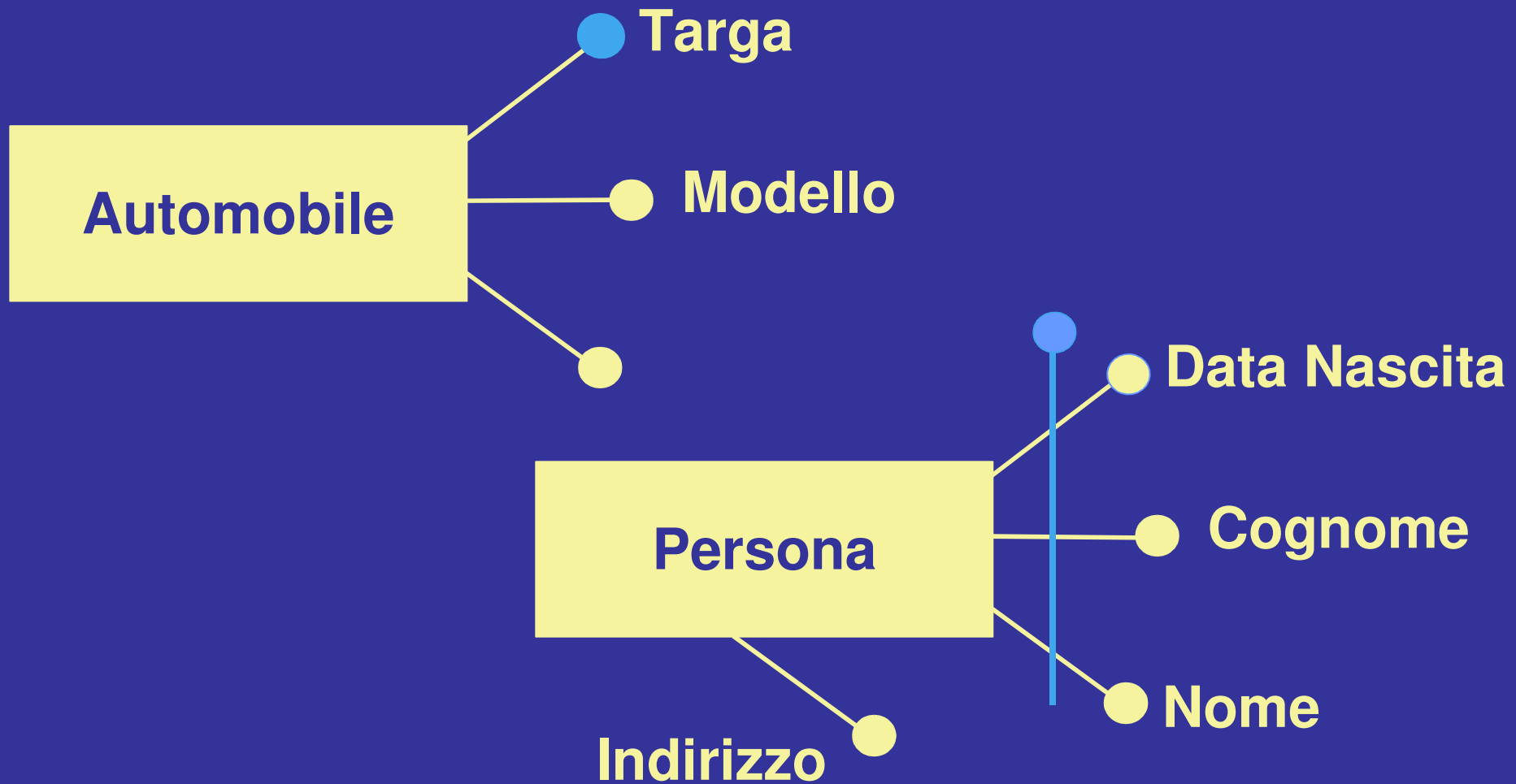
Rappresentazione grafica



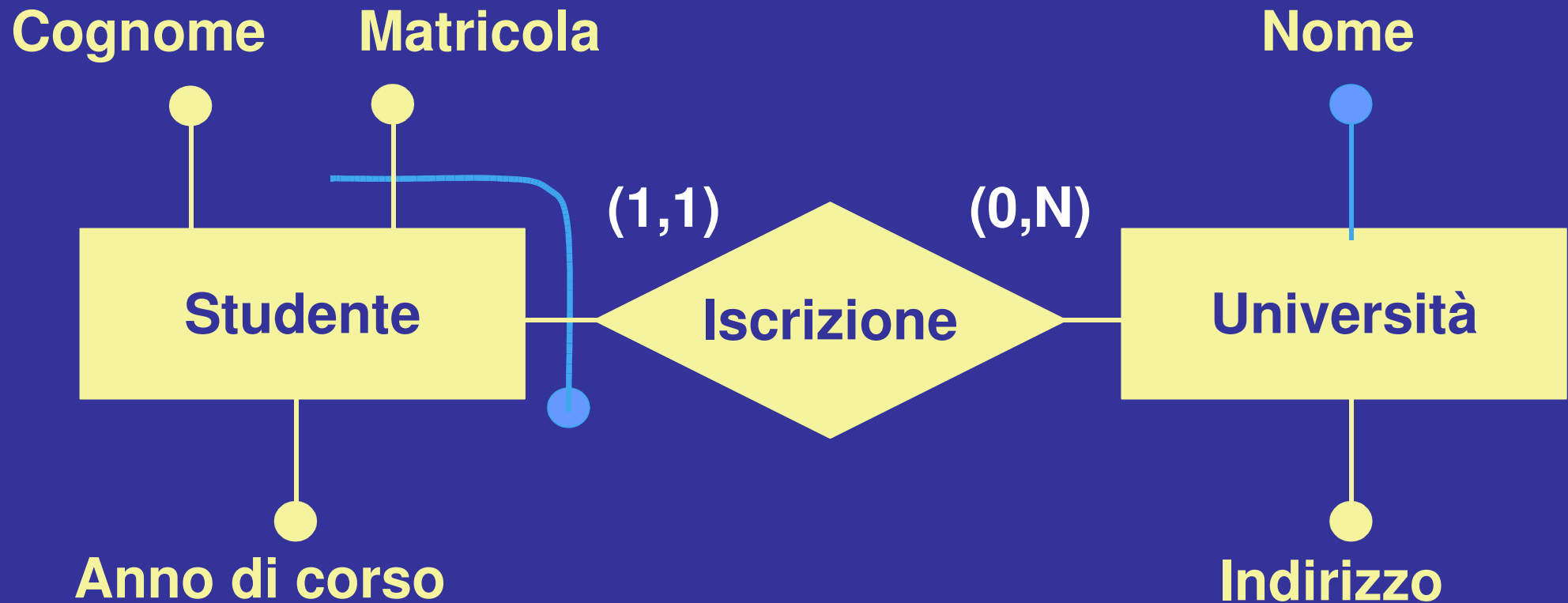
Identificatore di una entità

- “strumento” per l’identificazione univoca delle occorrenze di un’entità
- costituito da:
 - attributi dell’entità
 - **identificatore interno**
 - (attributi +) entità esterne attraverso relationship
 - **identificatore esterno**

Identificatori interni



Identificatore esterno



Alcune osservazioni

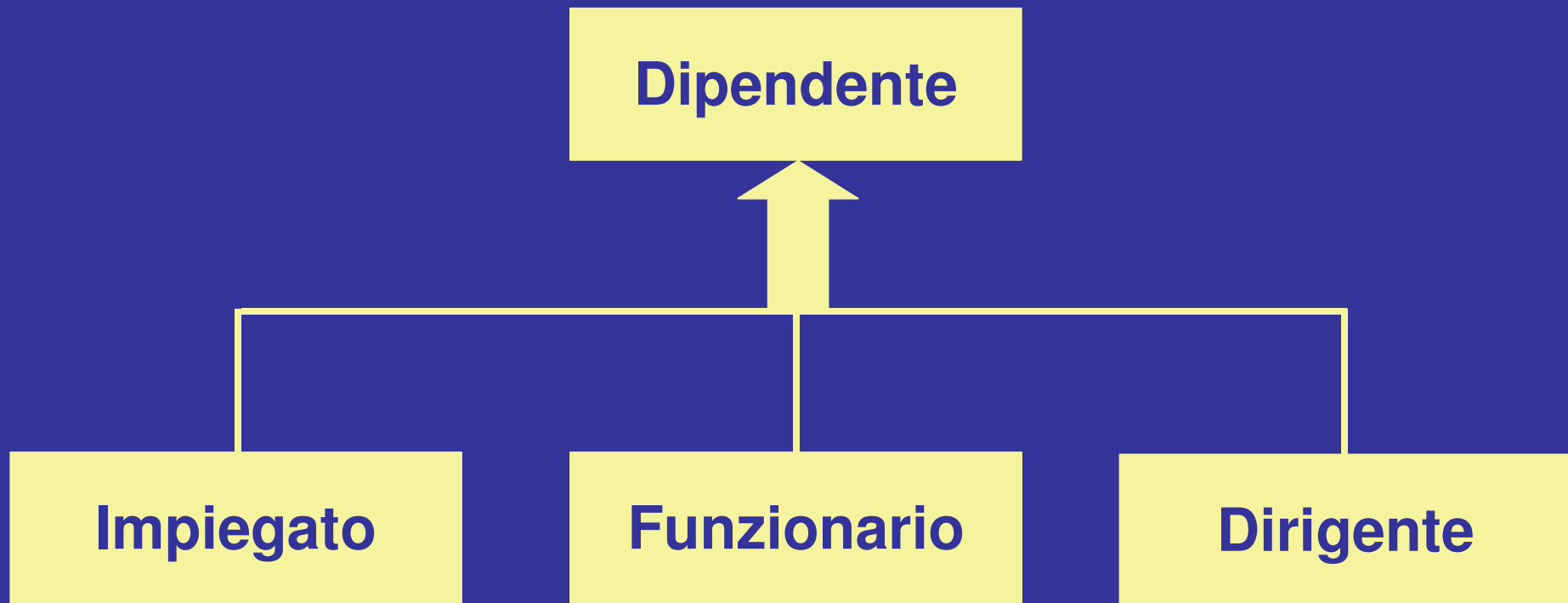
- ogni entità deve possedere almeno un identificatore, ma può averne in generale più di uno
- una identificazione esterna è possibile solo attraverso una relationship a cui l'entità da identificare partecipa con cardinalità (1,1)



Generalizzazione

- mette in relazione una o più entità E_1, E_2, \dots, E_n con una entità E , che le comprende come casi particolari
 - E è **generalizzazione** di E_1, E_2, \dots, E_n
 - E_1, E_2, \dots, E_n sono **specializzazioni** (o sottotipi) di E

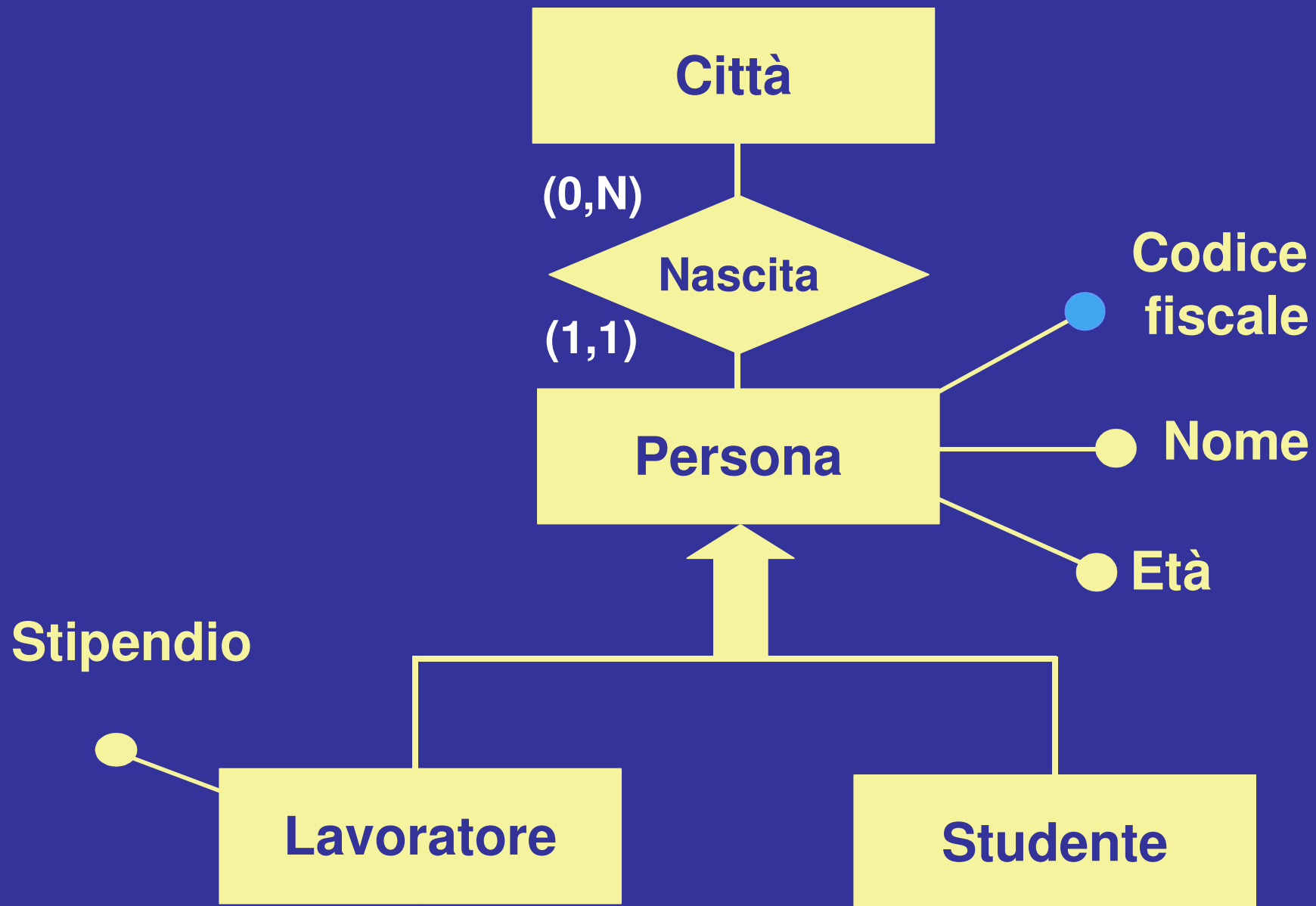
Rappresentazione grafica



Proprietà delle generalizzazioni

Se E (genitore) è generalizzazione di E_1, E_2, \dots, E_n (figlie):

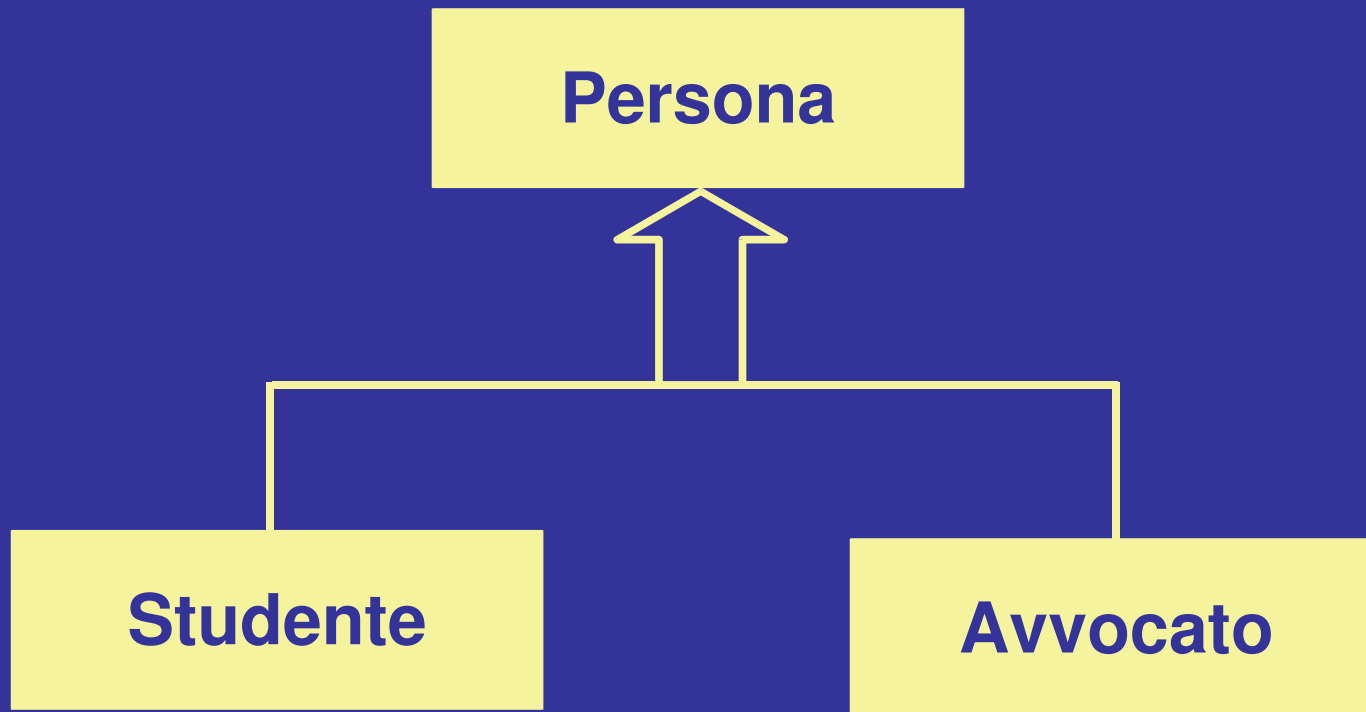
- ogni proprietà di E è significativa per E_1, E_2, \dots, E_n (**ereditarietà**)
- ogni occorrenza di E_1, E_2, \dots, E_n è occorrenza anche di E



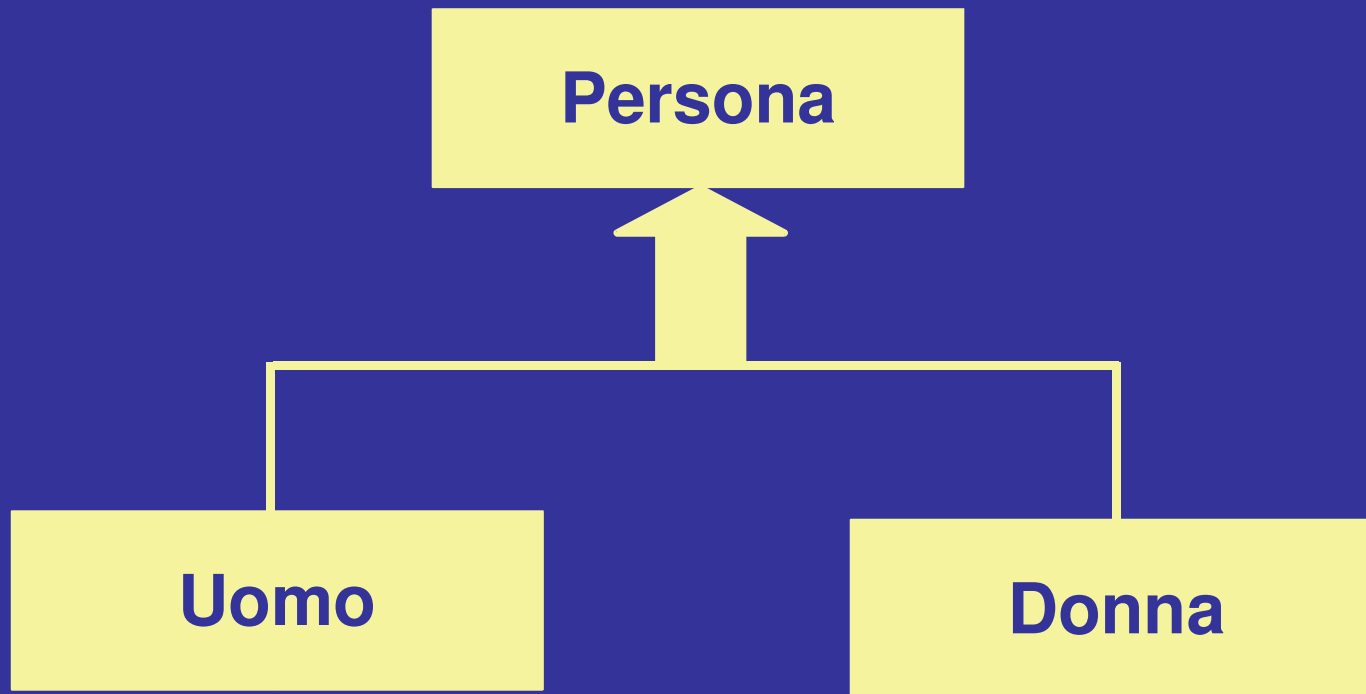
Tipi di generalizzazioni

- **totale** se ogni occorrenza dell'entità genitore è occorrenza di almeno una delle entità figlie, altrimenti è **parziale**
- **esclusiva** se ogni occorrenza dell'entità genitore è occorrenza di al più una delle entità figlie, altrimenti è **sovrapposta**
- consideriamo (senza perdita di generalità) solo generalizzazioni esclusive e distinguiamo fra totali e parziali

Generalizzazione Parziale



Generalizzazione Totale

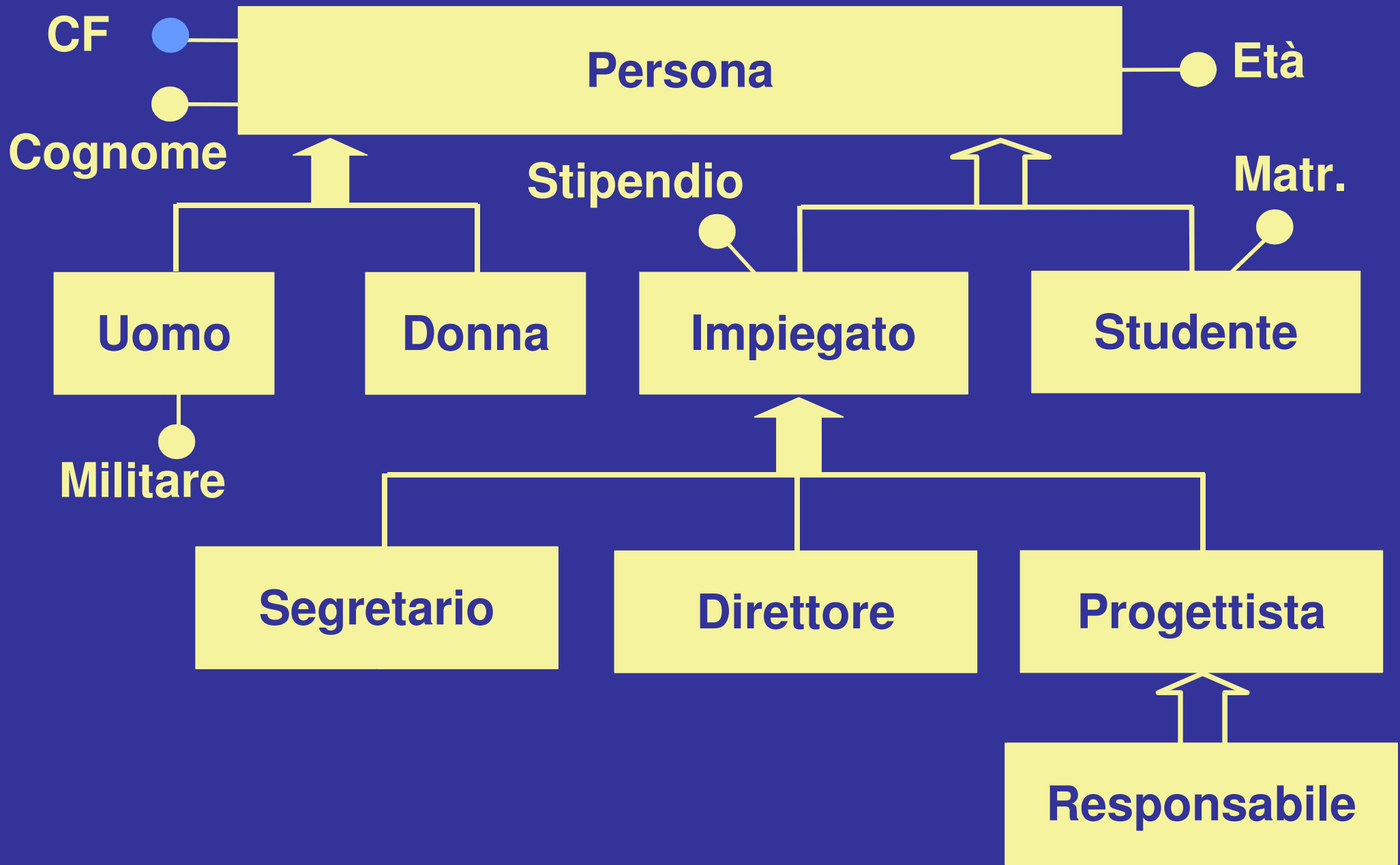


Altre proprietà

- possono esistere gerarchie a più livelli e multiple generalizzazioni allo stesso livello
- un'entità può essere inclusa in più gerarchie, come genitore e/o come figlia
- se una generalizzazione ha solo un'entità figlia si parla di **sottoinsieme**

Esercizio

- Le persone hanno CF, cognome ed età; gli uomini anche la posizione militare; gli impiegati hanno lo stipendio e possono essere segretari, direttori o progettisti (un progettista può essere anche responsabile di progetto); gli studenti (che non possono essere impiegati) un numero di matricola; esistono persone che non sono né impiegati né studenti (ma i dettagli non ci interessano)



Applicazioni degli schemi ER

I diagrammi ER possono essere usati in vari modi:

- come ausilio alla progettazione dei dati**
- a scopo documentativo, in quanto facilmente comprensibili a chi non è esperto di basi di dati**
- per descrivere la struttura di una base di dati già esistente**

Insufficienza degli schemi ER

Gli schemi ER non sono quasi mai sufficienti per rappresentare tutte le informazioni che si desidera.

- talvolta è necessario tralasciare qualche attributo per rendere più chiaro lo schema
- lo schema indica il nome degli attributi ma non i valori che questi possono assumere (dominio)
- alcuni vincoli tra i dati non possono essere espressi

Le Regole Aziendali

Occorre accompagnare il diagramma ER con ulteriore documentazione. Le informazioni supplementari allo schema ER sono spesso chiamate “**business rules**” (regole aziendali).

Le Regole Aziendali /2

Una regola aziendale può essere:

- la **descrizione di un concetto** rilevante di una applicazione (definizione precisa di una entità o di un attributo,...)

“una istanza della entità impiegato identifica un qualunque impiegato dell'azienda, a qualunque livello livello”

Le Regole Aziendali /3

- un **vincolo di integrità** sui dati, sia esso la documentazione di un vincolo espresso nello schema ER che un vincolo nuovo non esprimibile nello schema.

“un impiegato deve avere uno stipendio maggiore del direttore del dipartimento a cui afferisce”

- una **derivazione**, ovvero una regola matematica per ottenere alcuni attributi da altri attributi dello schema

“l'attributo Costo di ottiene sommando all'attributo Costo Netto l'attributo Tasse”

Cognome

(0,1)

(1,1)

Telefono

(1,N)



Direzione

Dipartimento

(0,1)

(0,N)

Afferenza

(1,1)

Nome

Codice

(0,N)



(1,N)

(0,1)

Data



(1,N)



Budget

Nome

Via

Indirizzo

Città

CAP

Dizionario dei dati (entità)

Le regole aziendali di tipo descrittivo vengono di solito raccolte nel “dizionario dei dati”.

Entità	Descrizione	Attributi	Identificatore
Impiegato	Dipendente dell'azienda	Codice, Cognome, Stipendio	Codice
Progetto	Progetti aziendali	Nome, Budget	Nome
Dipartimento	Struttura aziendale	Nome, Telefono	Nome, Sede
Sede	Sede dell'azienda	Città, Indirizzo	Città

Dizionario dei dati (relationship)

Relazioni	Descrizione	Componenti	Attributi
Direzione	Direzione di un dipartimento	Impiegato, Dipartimento	
Afferenza	Afferenza a un dipartimento	Impiegato, Dipartimento	Data
Partecipazione	Partecipazione a un progetto	Impiegato, Progetto	
Composizione	Composizione dell'azienda	Dipartimento, Sede	

Vincoli

Vincoli di integrità sui dati

1. Il direttore di un dipartimento deve afferire a tale dipartimento
2. Un impiegato non deve avere uno stipendio maggiore del direttore del dipartimento al quale afferisce
3. Un dipartimento con sede a Roma deve essere diretto da un impiegato con più di dieci anni di anzianità
4. Un impiegato che non afferisce a nessun dipartimento non deve partecipare a nessun un progetto

Derivazioni

Regole di Derivazione

1. Il budget di un progetto si ottiene moltiplicando per tre la somma degli stipendi degli impiegati che vi partecipano