UNIVERSITA DEGLI STUDI DI FOGGIA

Dipartimento di Agraria

CdI in Ingegneria dei Sistemi Logistici per l'Agroalimentare

Corso integrato di Sistemi di Elaborazione

Modulo I

Prof. Crescenzio Gallo

crescenzio.gallo@unifg.it

La progettazione di una base di dati

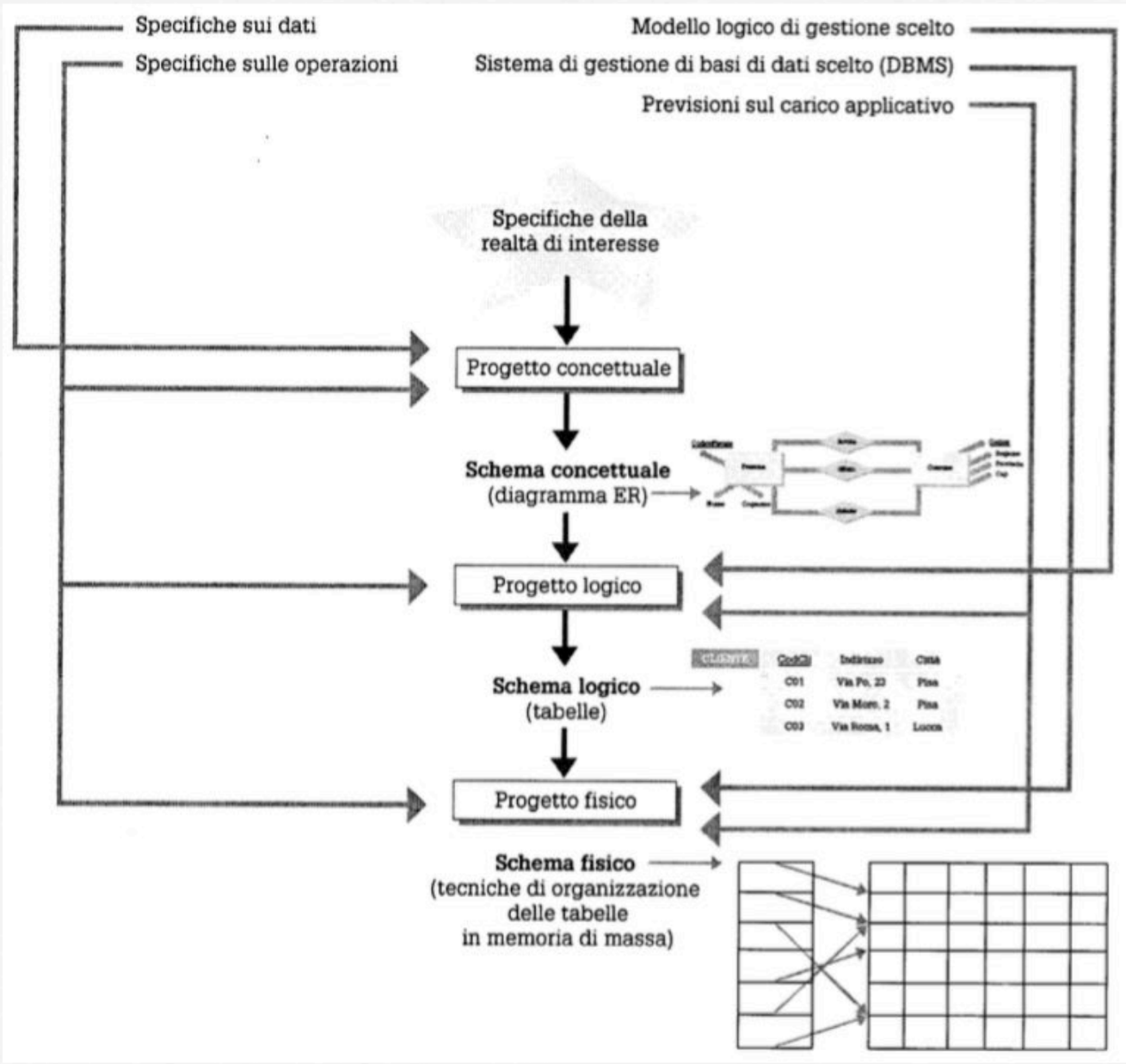
Progettazione di una base di dati

- In un sistema informativo la risorsa centrale sono i dati.
- Riveste quindi fondamentale importanza la progettazione della base di dati.
- La metodologia di progettazione di una base di dati si articola nelle seguenti fasi:
  1. progettazione **concettuale**
     - ➡ *diagrammi E/R (Entity/Relationship)*
  2. progettazione **logica**
  3. progettazione **fisica**

Progettazione di una base di dati

Requisiti

E/R



Modello relazionale

DBMS

Progettazione concettuale

Progettazione concettuale

- Lo scopo della progettazione concettuale è *costruire* e *definire* una rappresentazione corretta e completa della realtà di interesse.
- Essa produce uno schema concettuale (diagramma E/R), cioè una rappresentazione astratta e formale della realtà.

Progettazione concettuale

- La progettazione concettuale consiste nel riorganizzare tutti gli elementi che si hanno a disposizione per definire un modello astratto della base di dati, evitando i dettagli realizzativi.
- Assumono quindi un ruolo di primaria importanza le **astrazioni**.

Astrazioni

- L'**astrazione** è un procedimento mentale che permette di evidenziare alcune proprietà, ritenute significative, degli oggetti osservati, escludendone altre giudicate non rilevanti.
- Tramite l'astrazione si arriva a definire nuove *classi di oggetti*.

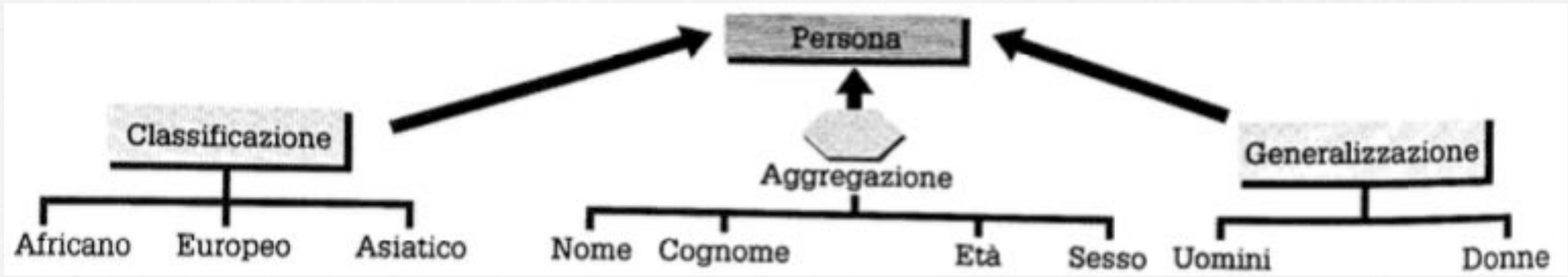
Astrazioni

Osserviamo i seguenti oggetti:

È possibile selezionare gli aspetti in comune (colore, numero di ruote, etc.) ed escludere gli aspetti che li differenziano, realizzando un processo di astrazione che porta al concetto di *automobile*.

Astrazioni

- Le **astrazioni** sono una modalità di *descrizione della realtà* comune a tutti i modelli, che possiamo utilizzare per progettare una base di dati.
- Nella progettazione di basi di dati vengono utilizzati principalmente tre procedimenti di astrazione:
  1. classificazione
  2. aggregazione
  3. generalizzazione

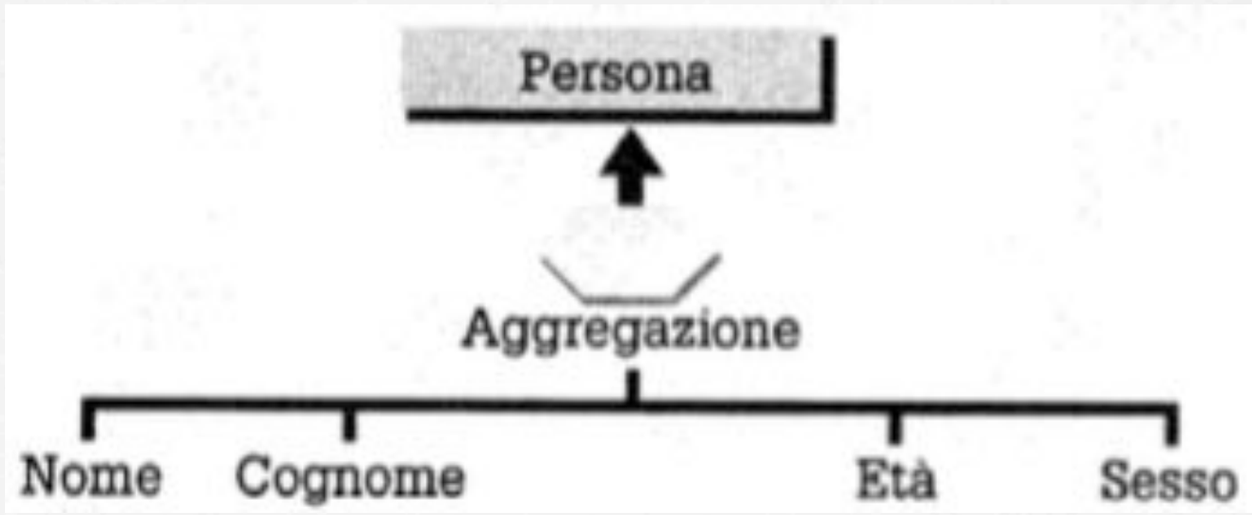


Astrazione per classificazione

- L'**astrazione per classificazione** permette di definire una classe a partire da un insieme di oggetti di cui si individuano proprietà comuni.
- Analizziamo le seguenti immagini → Se individuiamo le caratteristiche comuni ai soggetti presi in esame (respirano; hanno una testa, due braccia e due gambe; parlano; …) e scartiamo quelle che li differenziano, arriviamo a definire una classe Persona.

Astrazione per aggregazione

- L'**astrazione per aggregazione** unisce una o più classi componenti (o proprietà) per definire una nuova classe di arrivo.
- Consideriamo le proprietà informative *Nome*, *Cognome*, *Età*, *Sesso*. La classe di cui esse sono parti componenti è appunto la classe *Persona*.



Astrazione per generalizzazione

- L'**astrazione per generalizzazione** definisce una classe come unione di un insieme di classi, ognuna delle quali è completamente contenuta (parte, sottoinsieme) nella classe da definire.
- Le parti sono mutuamente esclusive.

Diagrammi E/R

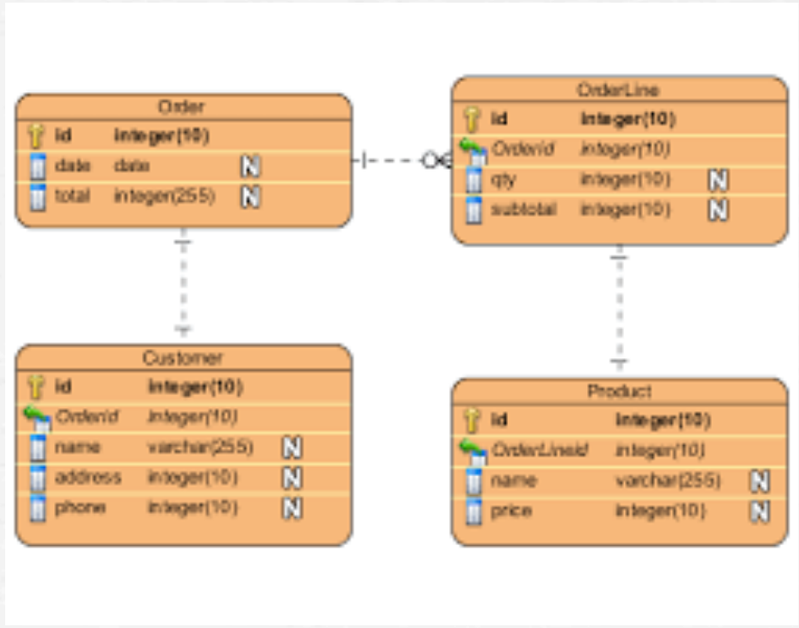
Diagramma E/R

- Il diagramma Entità/Relazioni(*) (in inglese Entity/Relationship), introdotto nel 1976 da P. Chen, è un modello grafico per la descrizione dei dati e delle loro relazioni.
- È uno *strumento* per l'analisi delle caratteristiche di una realtà indipendente dalle possibili applicazioni.
- Gli elementi di base sono: *entità, attributi, associazioni (relazioni), astrazioni, vincoli di integrità, identificatori (chiavi).*

(*) *Il termine "relazione" va inteso nel senso di "associazione" tra entità. Non va confuso con l'analogo del modello relazionale, dove "relazione" ha il significato matematico-insiemistico di insieme di n-uple.*

Entità, istanze, attributi

- Le **entità** corrispondono a classi di oggetti del mondo reale, definite specificandone le proprietà (attributi).
- Gli elementi di un'entità vengono chiamati **istanze** (o occorrenze), come per gli oggetti di una classe.
- Esempi di entità:
  - *persona;*
  - *automobile;*
  - *movimento contabile;*
  - *esame sostenuto da uno studente.*



Entità, istanze, attributi

- Gli **attributi** specificano le proprietà di un'entità.
- Possono essere:
  - *semplici;*
  - *composti (o aggregati) di altri attributi, a loro volta semplici o aggregati.*
- Un attributo è specificato da:
  - *un* **nome***;*
  - *un* **formato** *(indica il tipo di valori ammessi; ad es. intero, carattere, booleano, data);*
  - *una eventuale* **dimensione** *(ad es. n.ro massimo di cifre o caratteri);*
  - *un* **valore** *(l'insieme dei valori assunti da un attributo si chiama* **dominio** *dell'attributo);*
  - *una* **opzionalità** *(possibile mancanza di valore = valore nullo).*

Entità, istanze, attributi

Esempio

- Considerando l'entità *Persona* possiamo dire che:
  - *suoi possibili attributi semplici sono* Cognome *(formato:* Stringa*)*, Sesso *(formato:* Enum*)*, Età *(formato:* Intero*);*
  - *un possibile attributo composto è* Residenza, *aggregazione di "tipo indirizzo" (Via, Piazza, etc.), "indirizzo" e "località";*
  - *il dominio dell'attributo* Età *può arrivare ad un massimo (ad es. 120);*
  - *il dominio dell'attributo* Sesso *è* {uomo, donna}*;*
  - *il dominio di* Cognome *è rappresentato da tutte le possibili stringhe di massimo 30 caratteri.*

Attributi chiave

- Si indica con il termine **chiave** (o chiave candidata) l'insieme di uno o più attributi che identificano univocamente le istanze.

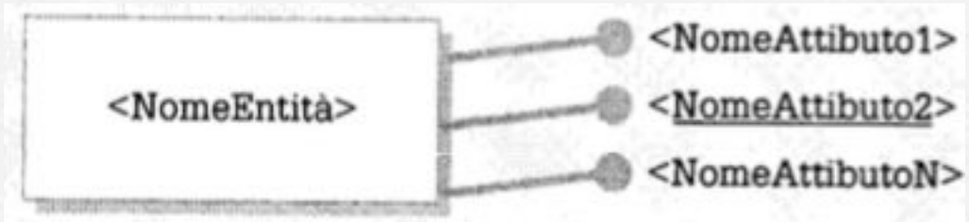
- Ad es. per l'entità Persona potremmo utilizzare come chiave:
  - *l'attributo* CodiceFiscale;
  - *l'attributo* NroProgressivo *(ad es. Matricola per gli studenti);*
  - *la combinazione degli attributi* Cognome, Nome, DataNascita *(qualora possibile).*

Chiave primaria

- Tra le *chiavi candidate*, quella con il minor numero di attributi prende il nome di **chiave primaria** (*primary key*).
- Nell'esempio precedente può essere chiave primaria: *CodiceFiscale* o *NroProgressivo*.
- È preferibile — per ragioni di efficienza — scegliere una chiave primaria intera, se disponibile (quindi meglio *NroProgressivo*); se non presente, di solito si aggiunge un attributo intero (incrementato automaticamente ad ogni nuova istanza, denominato tipicamente *Id* da "identificatore").

Rappresentazione grafica di entità e attributi

Per rappresentare graficamente le entità e i relativi attributi si utilizza la seguente notazione:



dove:

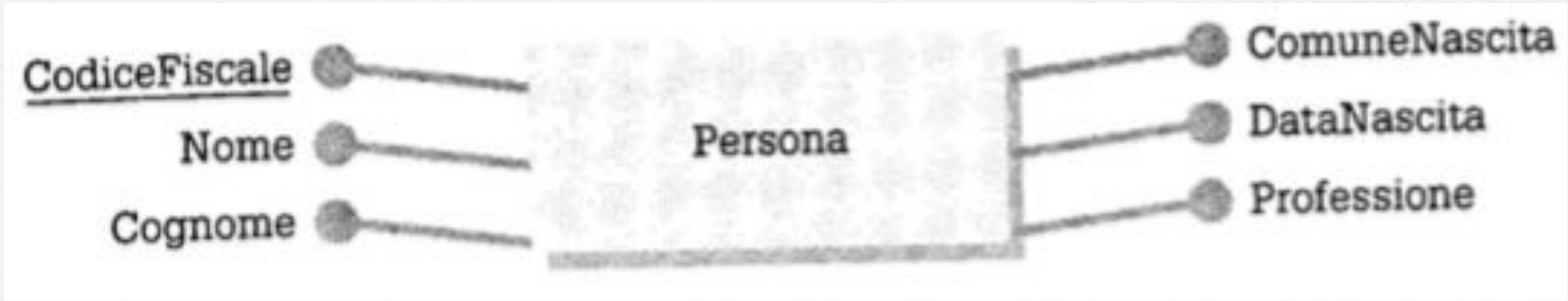
- per le *entità* si utilizza un rettangolo, contenente il nome dell'entità;
- per gli *attributi non chiave* una linea che termina con un cerchio e il nome dell'attributo;
- per gli *attributi chiave* si sottolinea il nome e/o colora il cerchio.

Rappresentazione grafica di entità e attributi

Per convenzione si utilizzano:

- *nomi al singolare* per le entità;
- *iniziali maiuscole* per (le parole costituenti) i nomi delle entità e degli attributi.

Una possibile rappresentazione dell'entità *Persona* è la seguente:



Le associazioni (relazioni tra entità)

- Un'**associazione** (in inglese *relationship*) è un legame esistente tra due o più entità.
- Così come le *entità* sono <u>classi di oggetti</u> del mondo reale, le *associazioni* sono <u>classi di fatti</u> che hanno proprietà omogenee e mettono in corrispondenza istanze di due o più entità.

Esempio

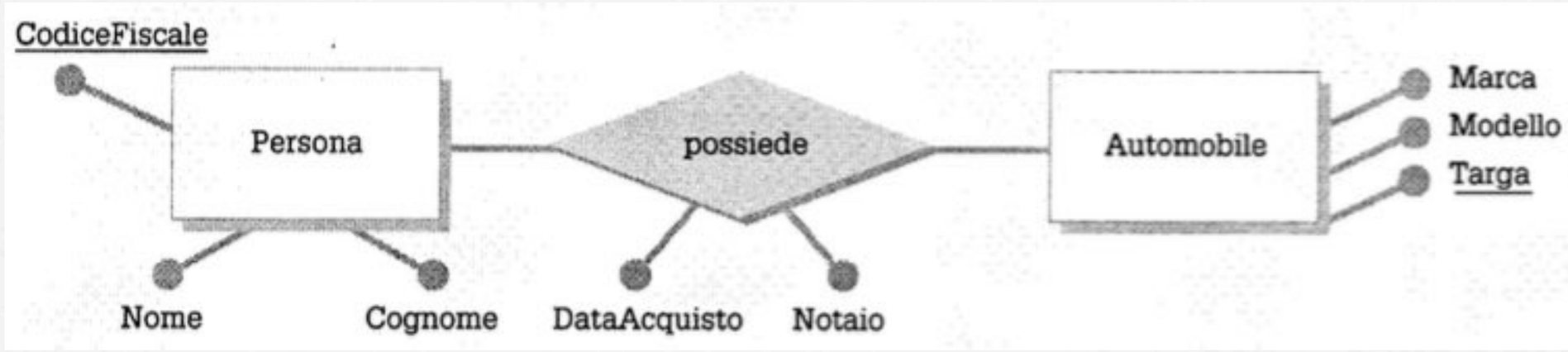
- *tra l'entità* Persona *e l'entità* Automobile *esiste un'associazione che descrive il* fatto *secondo il quale "una persona possiede una o più automobili" (e viceversa).*

Le associazioni (relazioni tra entità)

- Ogni associazione tra due entità possiede due **versi**.
- Si può ad es. dire che tra l'entità *Persona* e l'entità *Automobile* esiste l'associazione **possiede**; tra l'entità *Automobile* e l'entità *Persona* esiste l'associazione **possedutaDa**.
- Solitamente i **sostantivi** del mondo reale corrispondono alle *entità*, mentre i **verbi** corrispondono alle *associazioni*.
- Le associazioni si rappresentano graficamente con **rombi** collegati alle relative entità, con all'interno il nome dell'associazione (con *iniziale minuscola*).

Le associazioni (relazioni tra entità)

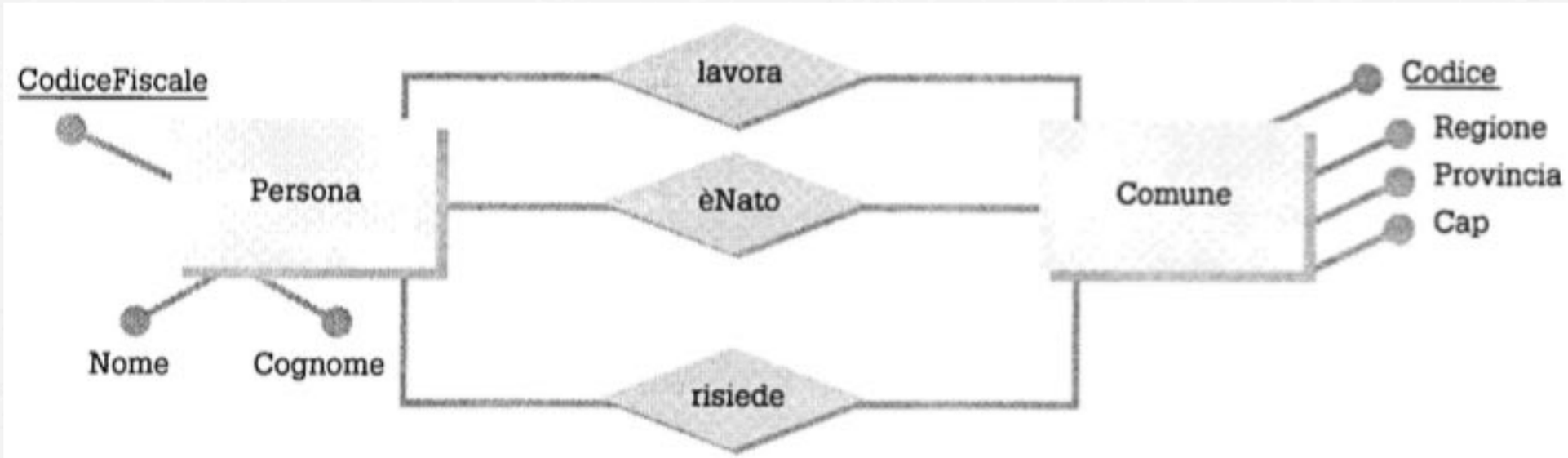
Ad es. l'associazione *possiede* tra le entità *Persona* e *Automobile* può essere rappresentata nel seguente modo:



Si osservi che gli attributi *DataAcquisto* e *Notaio* caratterizzano l'associazione e non le entità relazionate.

Associazioni multiple tra entità

Tra due entità possiamo definire più associazioni:



Le tre associazioni rappresentano i seguenti fatti:

- il comune dove le persone *lavorano*;
- il comune dove le persone sono *nate*;
- il comune dove le persone *risiedono*.

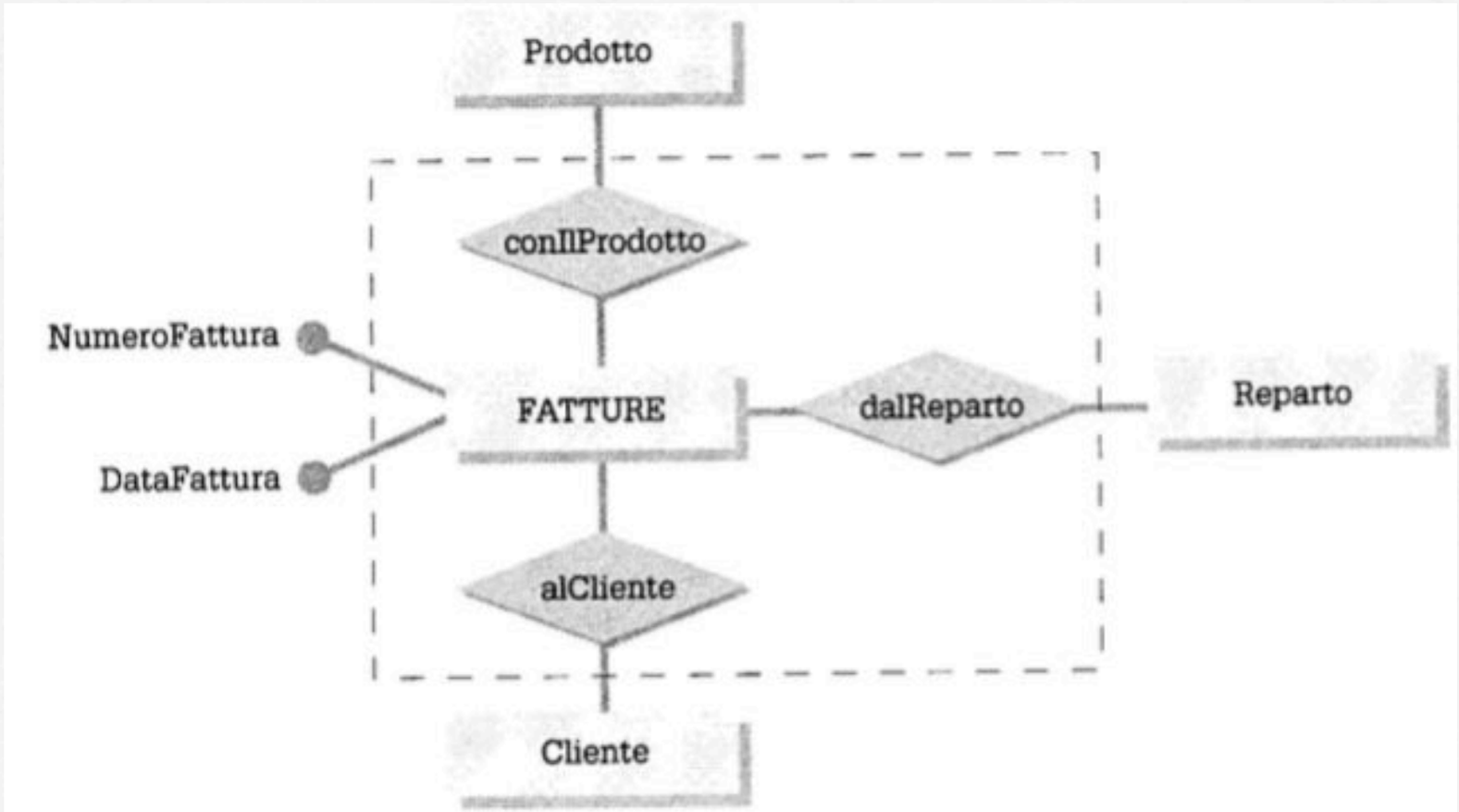
Associazioni multiple tra entità

Consideriamo la *realtà* di un supermercato, in cui si vuole rappresentare il *fatto* che ogni reparto possa emettere fatture ai clienti:

L'associazione ternaria *haFatturato* descrive appunto tale situazione.

Associazioni multiple tra entità

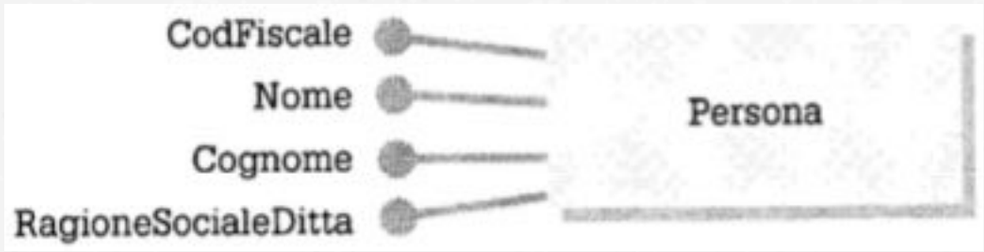
- Le *associazioni multiple* (tra più di due entità) possono (e devono) essere trasformate in *associazioni binarie* (tra due entità).
- La precedente associazione *haFatturato* può essere così trasformata:



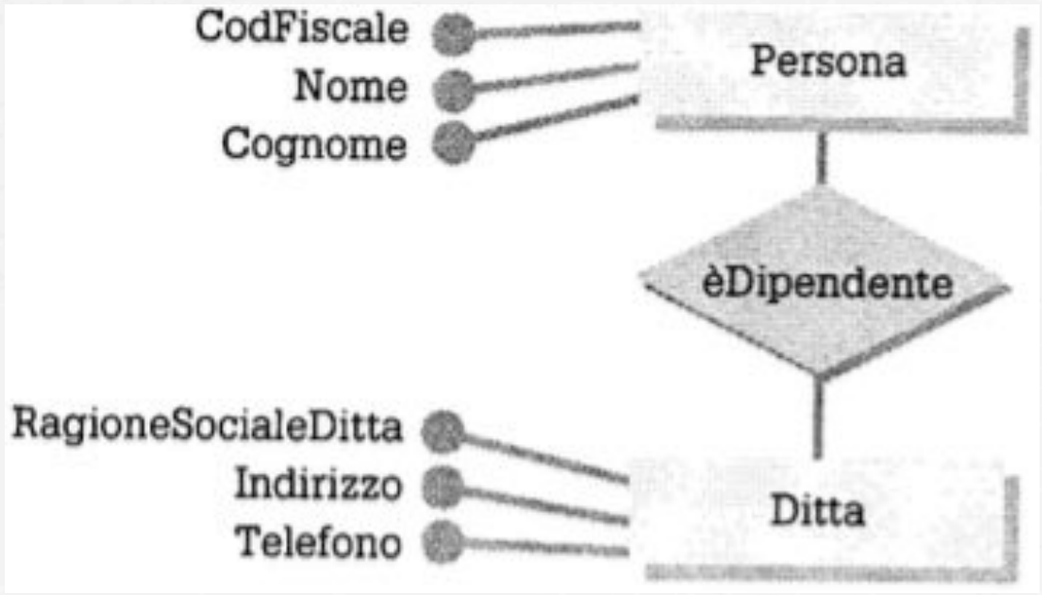
L'associazione ternaria *haFatturato* è diventata l'entità *Fatture* e sono state aggiunte tre associazioni necessarie per garantire i collegamenti preesistenti.

Trasferimento di attributi

Consideriamo l'entità *Persona*, contenente — tra gli altri — anche l'attributo *RagioneSocialeDitta*



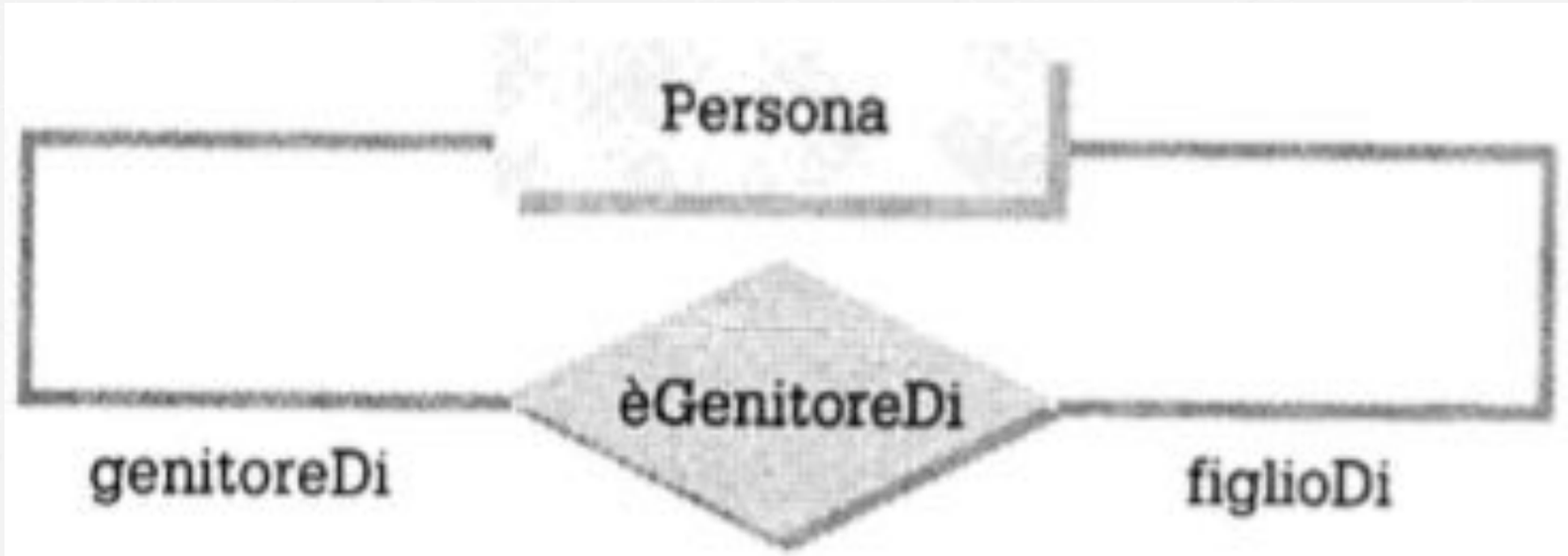
Se volessimo aggiungere anche gli altri attributi legati alla ditta avremmo una inutile ripetizione degli stessi per ogni persona che vi lavora.



Anche qui la soluzione è aggiungere una nuova entità, *Ditta*, con i relativi attributi.

Ruolo di un'entità

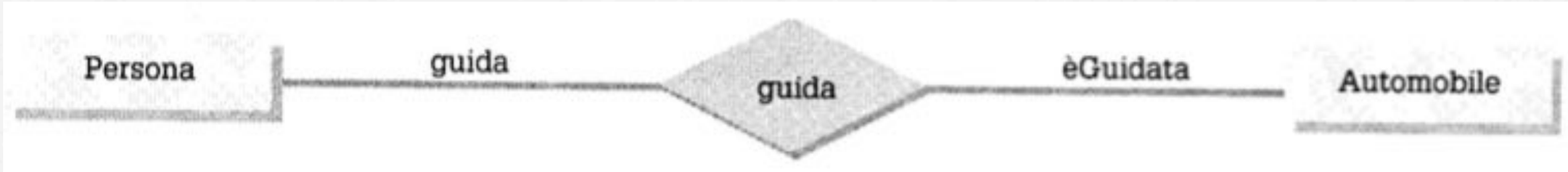
- Nell'esempio seguente l'entità *Persona* è relazionata su sé stessa, ma assume (a seconda del verso della relazione) **ruoli** diversi:



- *genitoreDi* e *figlioDi* sono i ruoli dell'entità *Persona* nell'ambito della relazione *èGenitoreDi*

Ruolo di un'entità / Relazione diretta e inversa

- Il concetto di **ruolo** può anche essere <u>esteso a qualsiasi relazione binaria</u>, come nel seguente esempio:



- Ciò deriva dalla (sempre possibile) doppia lettura del verbo (attivo e passivo) che caratterizza la relazione (*guida* = **relazione diretta**; *èGuidata* = **relazione inversa**).

Relazione totale e parziale

- Una relazione R tra due entità A e B è **totale** quando il legame tra le entità deve essere sempre presente (cioè ad ogni elemento di A deve corrispondere almeno un elemento di B).
- Altrimenti si dice **parziale**, e viene rappresentata tramite una linea tratteggiata.
- Può succedere che la relazione diretta sia totale mentre l'inversa è parziale, o viceversa. Ad esempio:

(una persona **può** *possedere un conto corrente, ma un conto corrente* **deve** *essere intestato ad una persona)*

Relazione totale e parziale

- In alternativa alla linea tratteggiata, si può utilizzare sempre la linea continua con un cerchio (che rappresenta uno 0: "anche nessuno") sul lato dell'entità "opzionale" (nel nostro esempio *ContoCorrente*).
- L'entità "obbligatoria" (*Persona*) è contrassegnata con una lineetta (che rappresenta un 1: "almeno uno").

(un conto corrente **deve** *essere intestato ad una sola persona; una persona* **può** *possedere uno o più conti correnti)*

Relazione totale e parziale

- Un formalismo alternativo (molto diffuso) per rappresentare la cardinalità di una relazione ed il vincolo minimo di istanze consiste nell'indicare, accanto all'entità interessata, tra parentesi tonde il numero minimo e massimo di istanze collegate tramite la relazione.

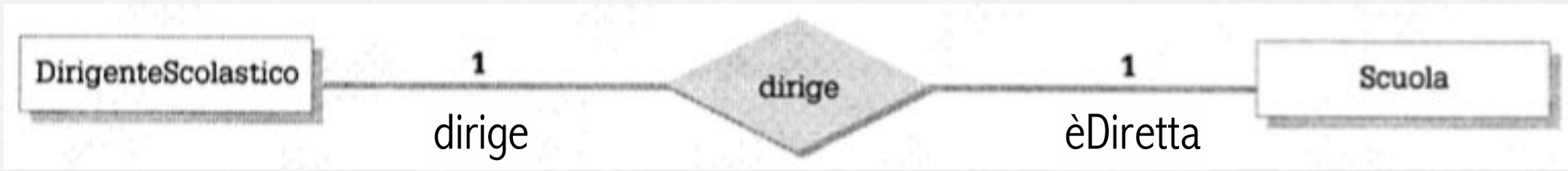
una persona **può** *essere titolare di nessuno, uno o più conti correnti*
un conto corrente **deve** *essere intestato almeno ad una persona*

Tipi di relazioni (1:1)

Una relazione binaria tra due entità *A* e *B* può essere di tipo:

- **uno a uno (1:1)**, quando ad un'istanza di *A* corrisponde una ed una sola istanza di *B* e viceversa.

Ad esempio:



(naturalmente non considerando i casi particolari di dirigenti super-impegnati!)

Tipi di relazioni (1:N)

Una relazione binaria tra due entità *A* e *B* può essere di tipo:

- **uno a molti (1:N)** o **semplice**, quando ad un'istanza di *A* possono corrispondere una o più istanze di *B* e ad ogni istanza di *B* deve corrispondere una sola istanza di *A*.

Ad esempio:

(una scuola può avere in organico più personale di segreteria; una persona di segreteria lavora in una ed una sola scuola)

Tipi di relazioni (N:N)

Una relazione binaria tra due entità *A* e *B* può essere di tipo:

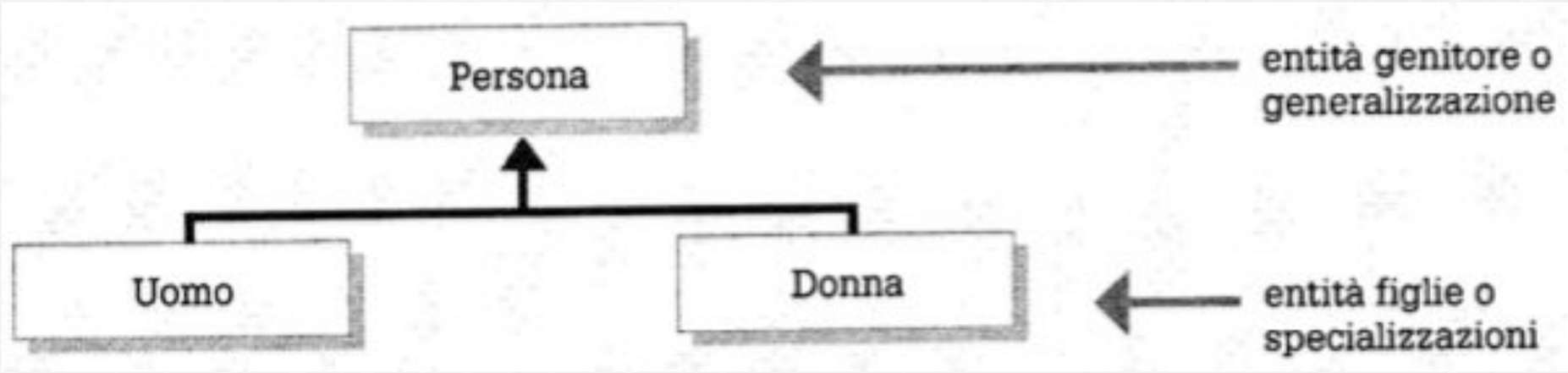
- **molti a molti (N:N)** o **complessa**, quando ad ogni istanza di *A* possono corrispondere una o più istanze di *B* e viceversa.

Ad esempio:

(una professore insegna a più studenti, laddove uno studente è seguito da più professori)

Gerarchie IS-A

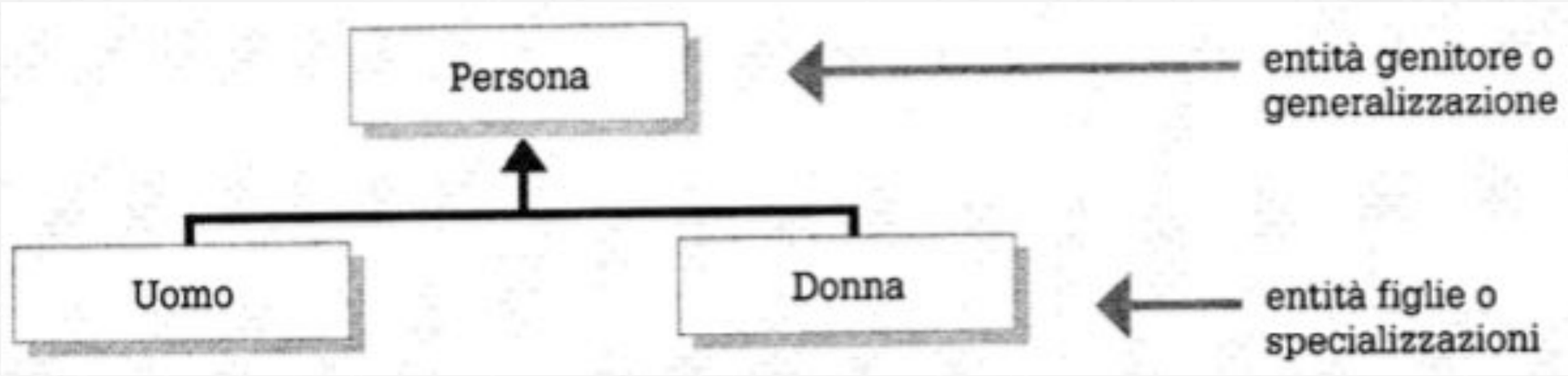
- In una **relazione per generalizzazione** l'entità astrazione è detta entità *padre* (o *genitore*), le entità "inferiori" sono dette entità *figlie* (o *specializzazioni*).
- La relazione per generalizzazione è anche detta associazione **IS-A** (dall'inglese "is a" cioè "è un").



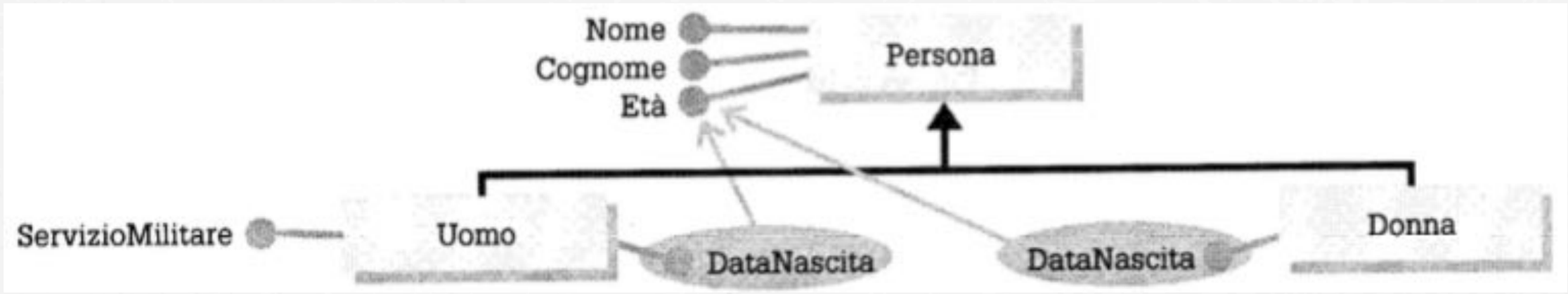
Diciamo che *Persona* è generalizzazione di *Uomo* e *Donna*.

Ereditarietà

- Le entità *figlie* hanno attributi che non possiede l'entità *genitore*.
- Tutte le proprietà dell'entità *genitore* (*attributi*, *associazioni*, *generalizzazioni* cui partecipa) sono però interamente posseduti dalle entità *figlie* (**ereditarietà**).



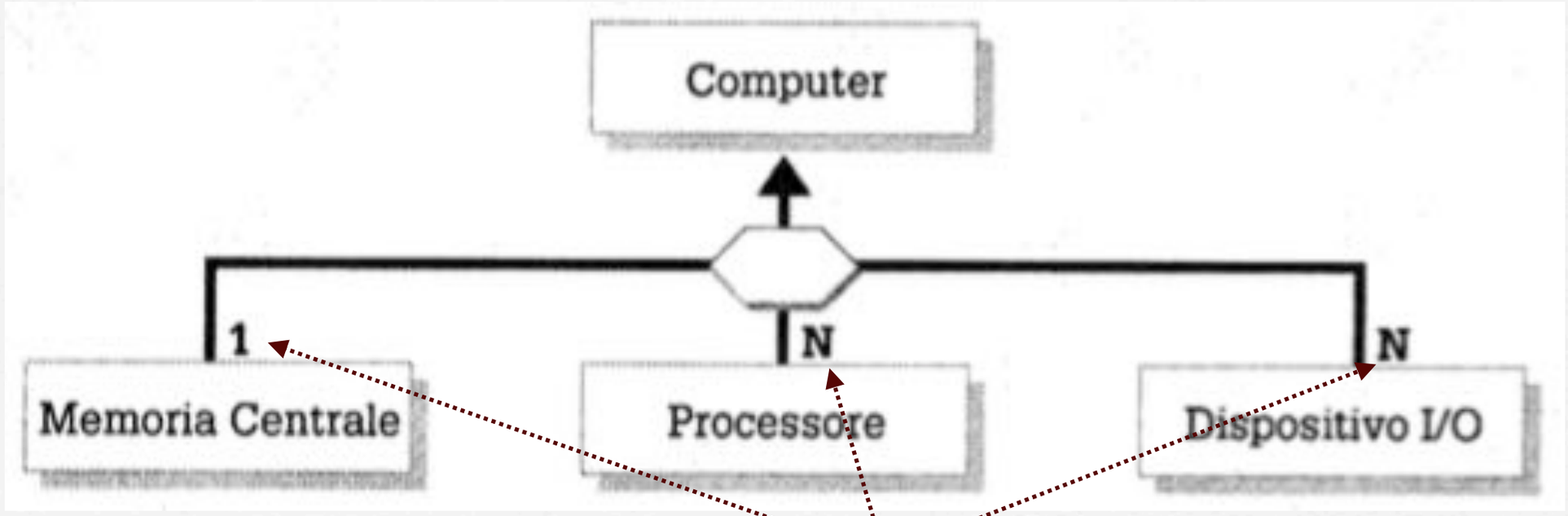
Ereditarietà: esempio



- L'attributo *Età* è definito nell'entità genitore *Persona* e viene ereditato da *Uomo* e *Donna*.
- Se però un attributo (ad es. *DataNascita*) è definito in tutte le entità figlie (*Uomo* e *Donna*, nell'esempio), esso deve essere considerato un attributo dell'entità superiore (*Persona* nell'esempio).

Relazioni Has-A

- In una **relazione per aggregazione** l'entità *composizione* (o *contenitore*) è formata dalle entità *componenti*, e viene anche chiamata associazione **HAS-A** ("has a" cioè "ha un").



Notiamo che è possibile indicare il **numero di elementi** *delle entità componenti che concorrono a formare un elemento dell'entità composta.*

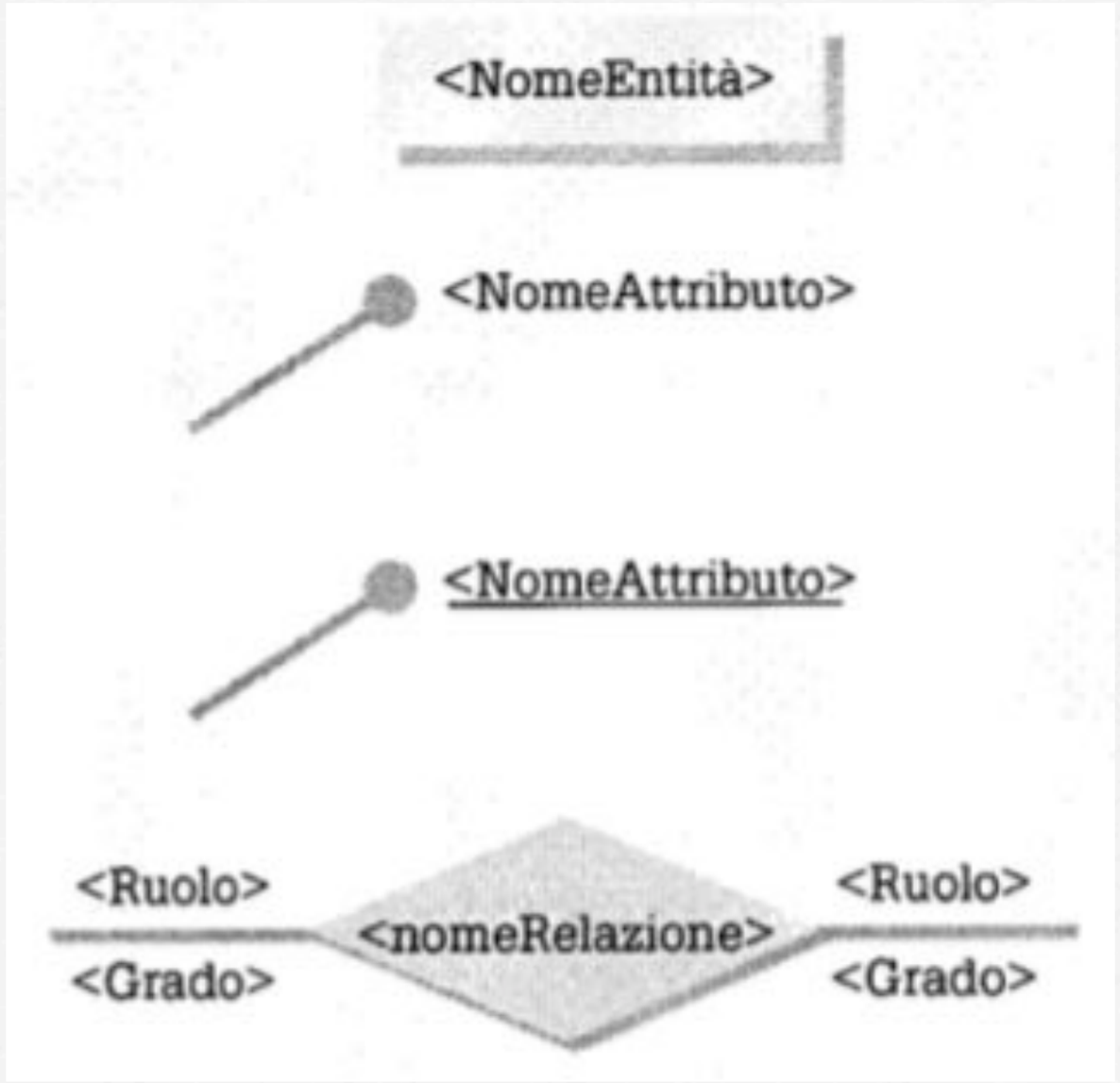
Simboli di un diagramma E/R

Entità

Attributo semplice

Attributo chiave

Associazioni (relazioni)

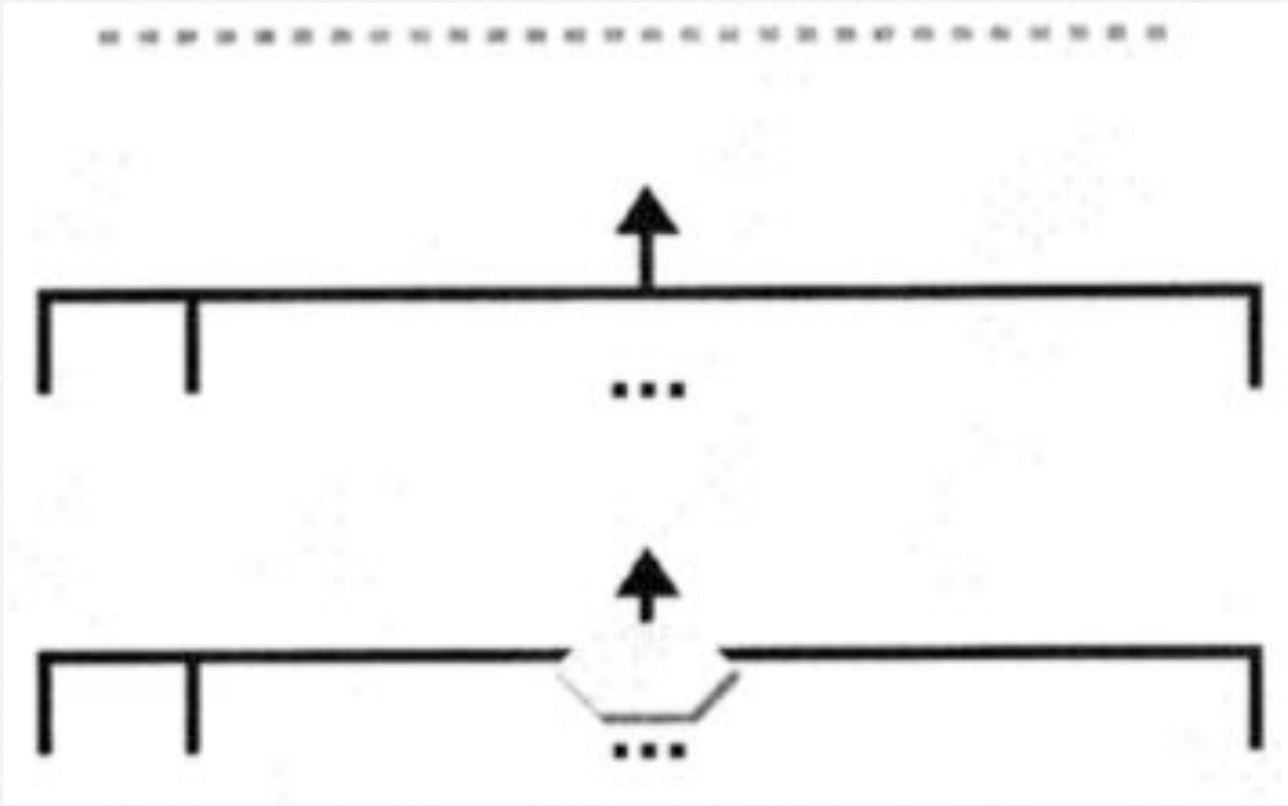


Simboli di un diagramma E/R

Parzialità di una relazione

Associazione per generalizzazione (IS-A)

Associazione per aggregazione (Has-A)



Vincoli di integrità

- Un **vincolo di integrità** è un'asserzione, ovvero un predicato che deve essere soddisfatto dalle istanze.
- Può essere di due tipi:

➡ *vincolo <u>implicito</u>*

- <u>vincolo di</u> **chiave primaria**
- vincolo **referenziale**

➡ *vincolo <u>esplicito</u>*

- (ad es. "0<*Età*<120").

Rappresentazione dei vincoli

- I vincoli impliciti di *chiave primaria* si rappresentano **sottolineando** i relativi attributi;
- i vincoli *referenziali* si rappresentano con le **linee continue** delle associazioni tra entità (obbligatorietà);
- i vincoli espliciti sono di solito espressi tramite uno **pseudolinguaggio**; ad esempio:
  - V1:[0<Età<120]
  - V2:[anno(Dipendente.DataAss)-anno(Dipendente.DataNasc)>16]

Si osservi che per riferirsi ad un attributo di un'entità, si fa utilizza la sintassi: <NomeEntità>.<NomeAttributo>

Esempio: concessionaria

Si vuole automatizzare la gestione delle attività di una concessionaria di automobili multi-marca.

Nel database vanno memorizzate le informazioni che consentono di:

- registrare le immatricolazioni di nuove automobili;
- registrare le informazioni che riguardano le riparazioni;
- elencare le automobili usate caratterizzate da un prezzo inferiore ad un valore fornito in input;
- elencare le riparazioni da effettuare per ogni auto usata;
- elencare le auto nuove e usate di ogni marca presente in concessionaria;
- elencare gli optional presenti su ogni automobile.

Esempio: concessionaria

Fasi di progettazione della base di dati

1. Definizione delle entità e degli attributi

- Dall'esempio si evince che vanno analizzate alcune delle funzioni svolte da una concessionaria.
- Sarà innanzitutto necessario raccogliere tutte le informazioni relative alle **automobili** (cilindrata, numero di cilindri e potenza in kW del **motore**, tipo e numero di telaio della **carrozzeria**, caratteristiche delle **ruote** e degli **optional**).

Esempio: concessionaria

Progettazione della base di dati

1. Definizione delle entità e degli attributi

- Registreremo inoltre le informazioni anagrafiche dei **proprietari** (supponendone uno solo per auto).
- Per la riparazione delle automobili sarà necessaria una lista delle **riparazioni**, individuando per ognuna di esse il livello di gravità.
- La lista sarà accompagnata anche da un preventivo di spesa.

Riassumiamo di seguito le entità individuate, con i relativi attributi.

Esempio: concessionaria

Progettazione della base di dati

1. Definizione delle entità e degli attributi

- **Proprietario** (CodFiscale, Nome, Cognome)
- **Automobile** (CodAuto, Marca Modello, Targa, Prezzo)
  - ➡ [ISA] **AutoNuova** (AnniGaranzia)
  - ➡ [ISA] **AutoUsata** (AnnoImmatr, KmPercorsi)
  - ❖ [HASA]**Motore** (CodMotore, TipoCarburante, Cilindrata)
  - ❖ [HASA] **Carrozzeria** (NumTelaio, Colore)
  - ❖ [HASA] **Ruota** (CodRuota, Larghezza, Diametro)
- **Riparazione** (CodRip, Tipo, Spesa, LivGravità)

Esempio: concessionaria

Progettazione della base di dati

1. Definizione delle entità e degli attributi

- Dalle entità e relativi attributi individuati si può opzionalmente ricavare il cosiddetto **Dizionario degli attributi**, che elenca una sola volta gli attributi (in ordine alfabetico) svincolati dall'entità di appartenenza. Ad esempio:

Attributo	Tipo	Lunghezza
Anno	INTERO	4
CodFiscale	STRINGA	16
Costo	DECIMALE	18,2
Id	INTERO	8

Esempio: concessionaria

Progettazione della base di dati

1. Definizione delle entità e degli attributi

- Gli attributi vengono di solito "raggruppati" per funzione.
- Per questo motivo abbiamo definito un generico attributo **Anno** poiché, successivamente, nella progettazione logica nell'entità Automobile si definirà l'attributo AnnoImmatr di tipo "Anno" (invece che genericamente INTERO).
- Analogamente l'attributo **Costo** del dizionario sarà utilizzato nella definizione di Prezzo (dell'entità Automobile) e Spesa (dell'entità Riparazione).

Esempio: concessionaria

Progettazione della base di dati

1. Definizione delle entità e degli attributi

- Infine l'attributo **Id** del dizionario rappresenta le chiavi primarie di tipo "contatore" (numero intero progressivo automatico), come nel caso degli attributi:
  - CodAuto (entità Automobile)
  - CodRip (entità Riparazione)

Esempio: concessionaria

Progettazione della base di dati

2. Definizione delle relazioni

- Dovendo trattare sia auto nuove che usate, scegliamo di utilizzare un'**associazione per generalizzazione** attraverso la quale memorizzare nell'entità genitore le informazioni comuni a tutte le automobili e nelle entità figlie solo quelle specifiche alle auto nuove e a quelle usate.
- Un proprietario può possedere più automobili; queste ultime sono l'aggregazione di motore, carrozzeria e ruote e la generalizzazione di auto nuova ed usata.

Esempio: concessionaria

Progettazione della base di dati

2. Definizione delle relazioni

- Un utile metodo per esplorare le possibili relazioni è dato dalla **matrice delle entità** che elenca le entità in riga e in colonna (con all'incrocio i nomi delle relazioni):

	Proprietario	Automobile	AutoNuova	AutoUsata	Optional	Riparazione
Proprietario		acquista				
Automobile						
Autonuova					èDotata	
AutoUsata						necessita
Optional						
Riparazione						

Esempio: concessionaria

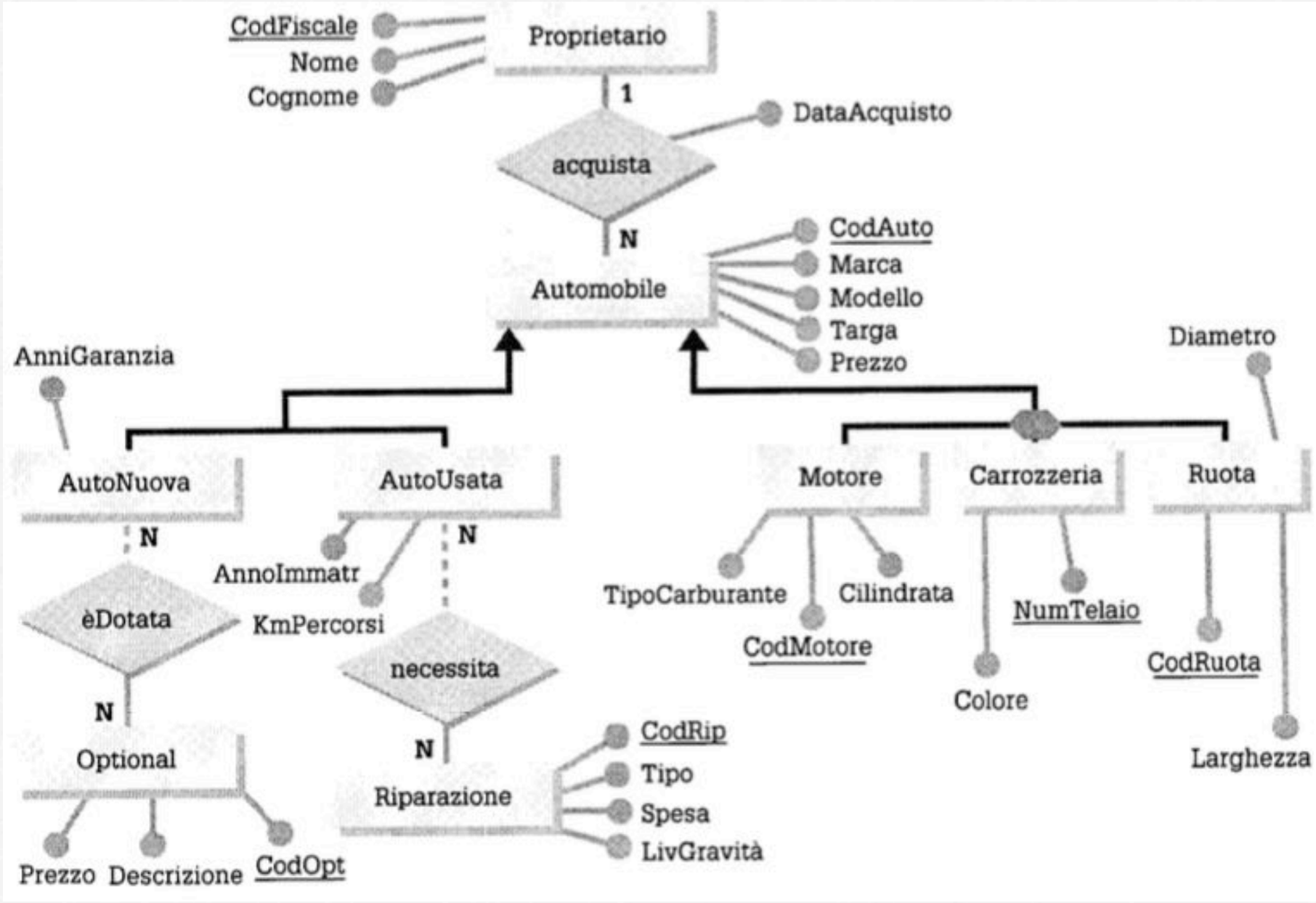


Diagramma E/R

Esempio: concessionaria

Progettazione della base di dati

3. Definizione dei vincoli

Vincoli di integrità (semantica)

- V1:[Automobile.Prezzo > 0]
- V2:[AutoUsata.AnnoImmatr > 01/01/1990]
- V3:[AutoUsata.KmPercorsi < 300000]
- V4:[SE(Riparazione.LivGravità ≥ 7) ALLORA Riparazione.Spesa > 1000]

L'ultimo vincolo esprime il fatto che "non devono esistere riparazioni che abbiano una spesa inferiore a 1000 Euro per una gravità uguale o superiore al livello 7".

Progettazione logica e fisica

Progettazione logica

- Il suo scopo è quello di *trasformare* lo schema astratto in uno *schema logico* ovvero in una rappresentazione efficiente rispetto alle strutture di un DBMS.
- Un esempio è la descrizione relazionale tramite tabelle relazionali di un diagramma E/R.

Progettazione fisica

- Il suo scopo è quello di *implementare* lo schema logico, definendo tutti gli aspetti fisici di memorizzazione e rappresentazione in memoria di massa e secondo il DBMS scelto.
- Si parla anche talvolta di progettazione *logico-fisica* intendendo questa come un'unica fase realizzativa.

Dal diagramma E/R allo schema relazionale

- Il passo successivo alla progettazione concettuale è la **progettazione logica** tramite il *modello relazionale*.

- Partendo dal diagramma E/R occorre quindi "mappare" le entità e le associazioni in tabelle — e relativi collegamenti — del modello relazionale.

Dal diagramma E/R allo schema relazionale

Lo **schema relazionale** si ricava dal **diagramma E/R** applicando alcune semplici *regole di derivazione* per rappresentare:

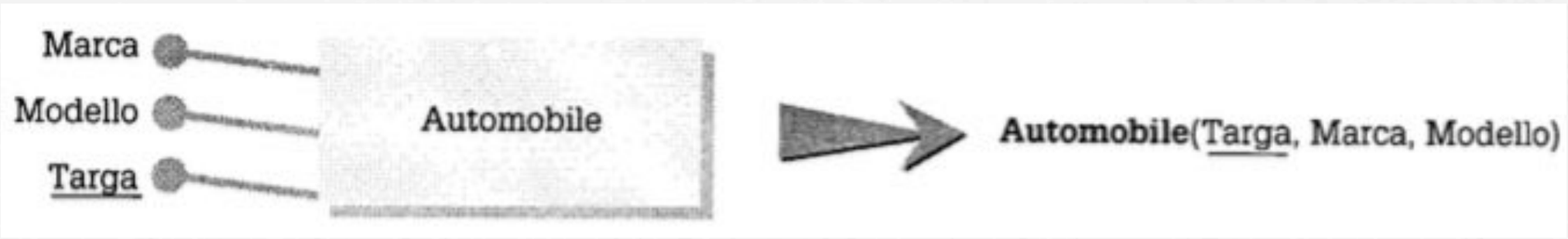
- entità e attributi;
- associazioni (relazioni), di tipo 1:1, 1:N, N:N;
- generalizzazioni;
- aggregazioni.

Esaminiamole in dettaglio.

Rappresentazione di entità e attributi

- Ogni *entità* diventa una tabella
- Ogni *attributo* dell'entità diventa un attributo (colonna) della tabella
- L'*attributo chiave* dell'entità diventa la chiave primaria della tabella

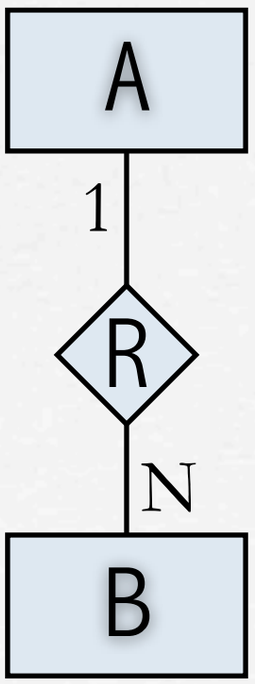
Ad Esempio:



Rappresentazione di associazioni

Date due entità A e B e un'associazione R di tipo 1:N da A (lato 1) a B (lato molti):

- l'entità A diventa una tabella T_A avente gli attributi di A;
- l'entità B diventa una tabella T_B avente gli attributi di B e la chiave K_A di T_A (chiave esterna in T_B).



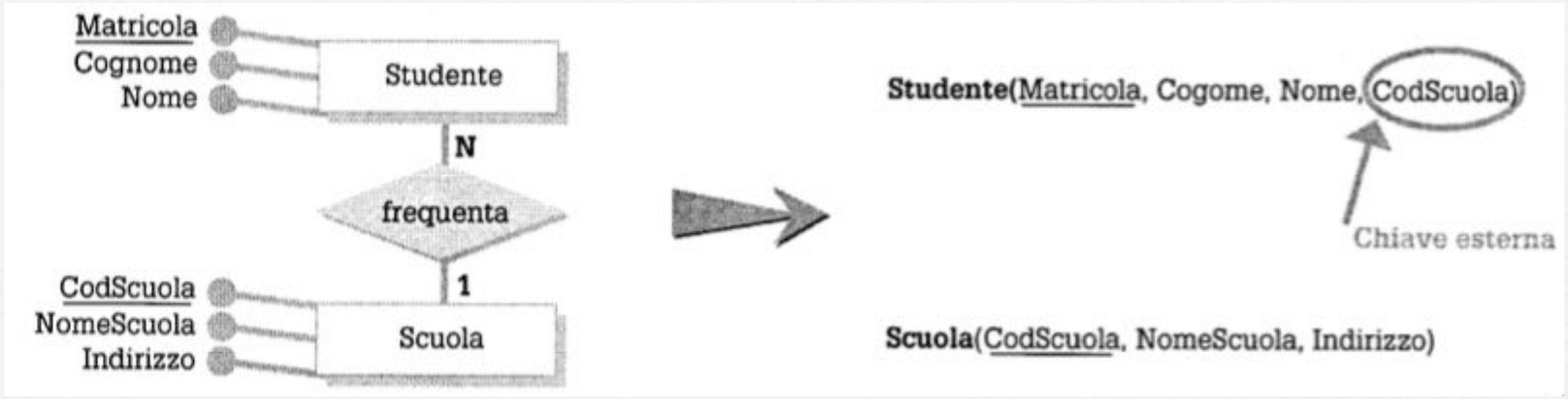
$T_A(<K_A>,<\text{attributi di A}>)$

$T_B(<K_B>,<\text{attributi di B}>,<K_A>)$

chiave esterna

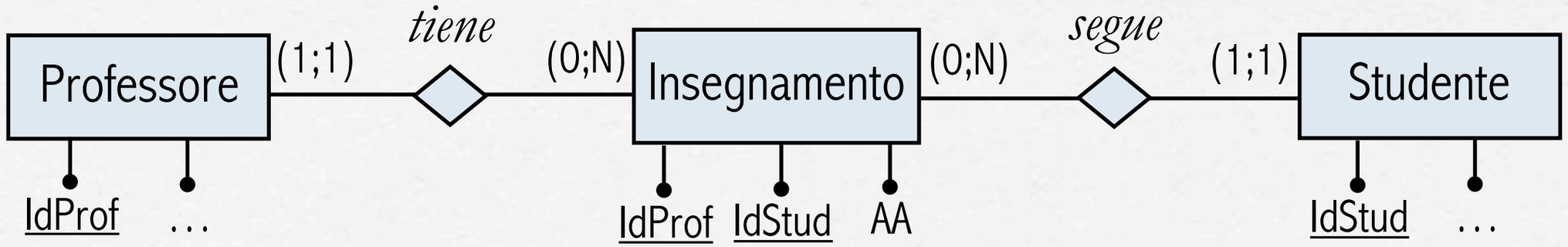
Rappresentazione di associazioni

Esempio



Associazioni molti a molti

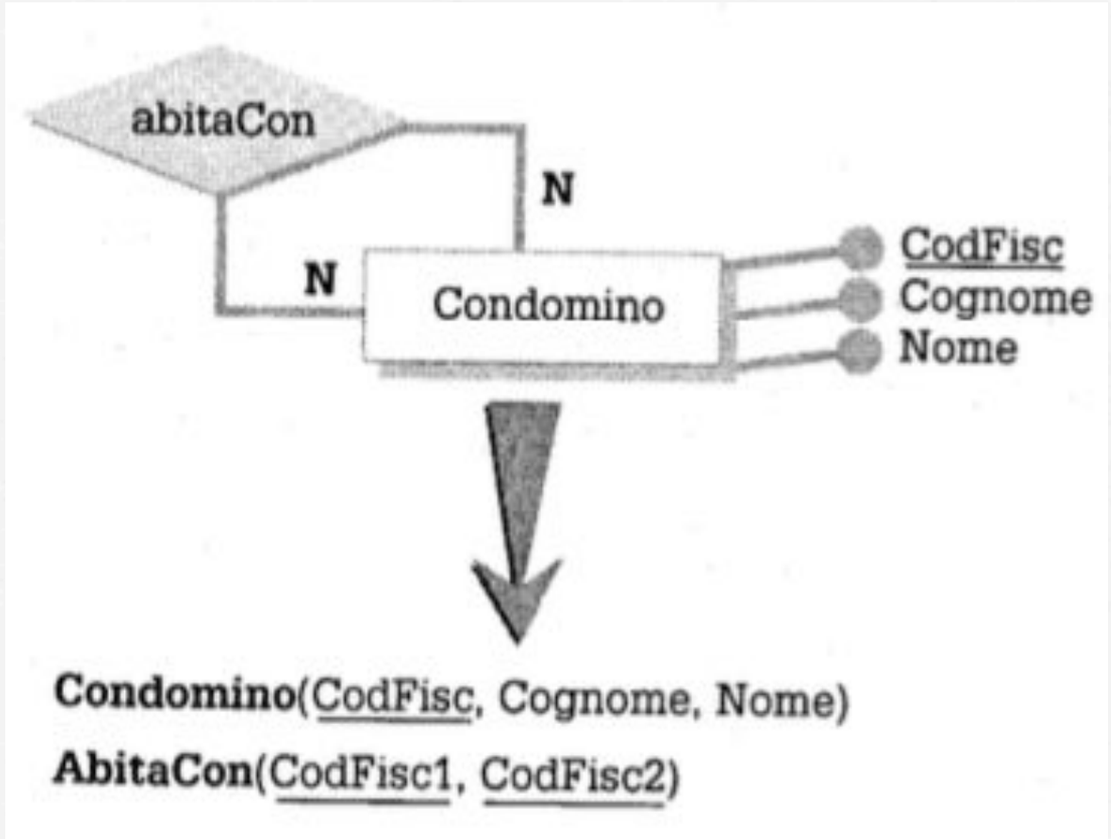
- Un'associazione **molti a molti (N:N)**, per essere implementata nel modello relazionale, deve essere suddivisa in due associazioni uno a molti, ciascuna avente le entità originarie sul lato "uno".
- L'associazione N:N originaria viene trasformata in un'entità che si trova sul lato "molti" delle due nuove associazioni, avente come attributi chiave almeno le chiavi delle entità iniziali, con eventuali altri attributi specifici.



Associazione "riflessiva"

- Un'associazione **su una stessa entità** viene implementata trasformando l'associazione in una nuova tabella.

Esempio:

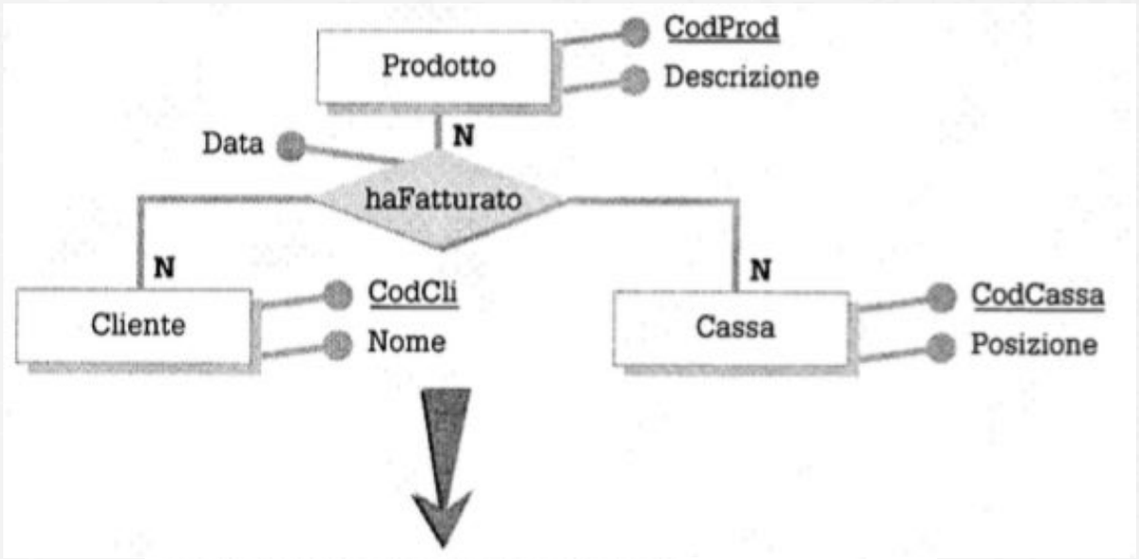


Associazioni multiple

- Un'associazione che collega **più di due entità** viene implementata trasformando l'associazione in una nuova opportuna tabella di collegamento.

N.B. Se una relazione possiede degli attributi, allora deve essere trasformata in un tabella nel modello relazionale.

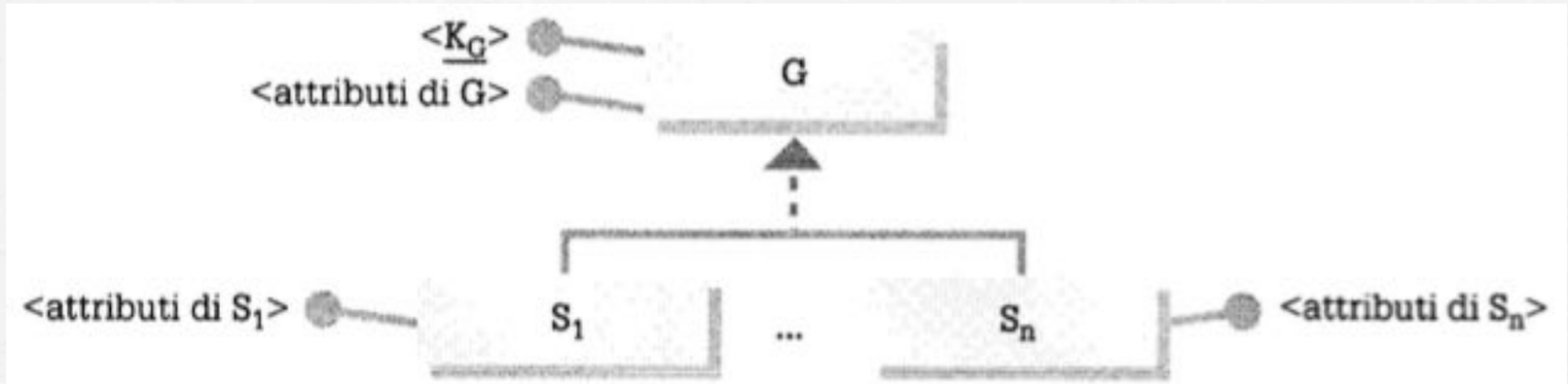
Esempio:



Prodotto(CodProd, Descrizione)
Cliente(CodCli, Nome)
Cassa(CodCassa, Posizione)
HaFatturato(CodProd, CodCli, CodCassa, Data)

Associazioni per generalizzazione

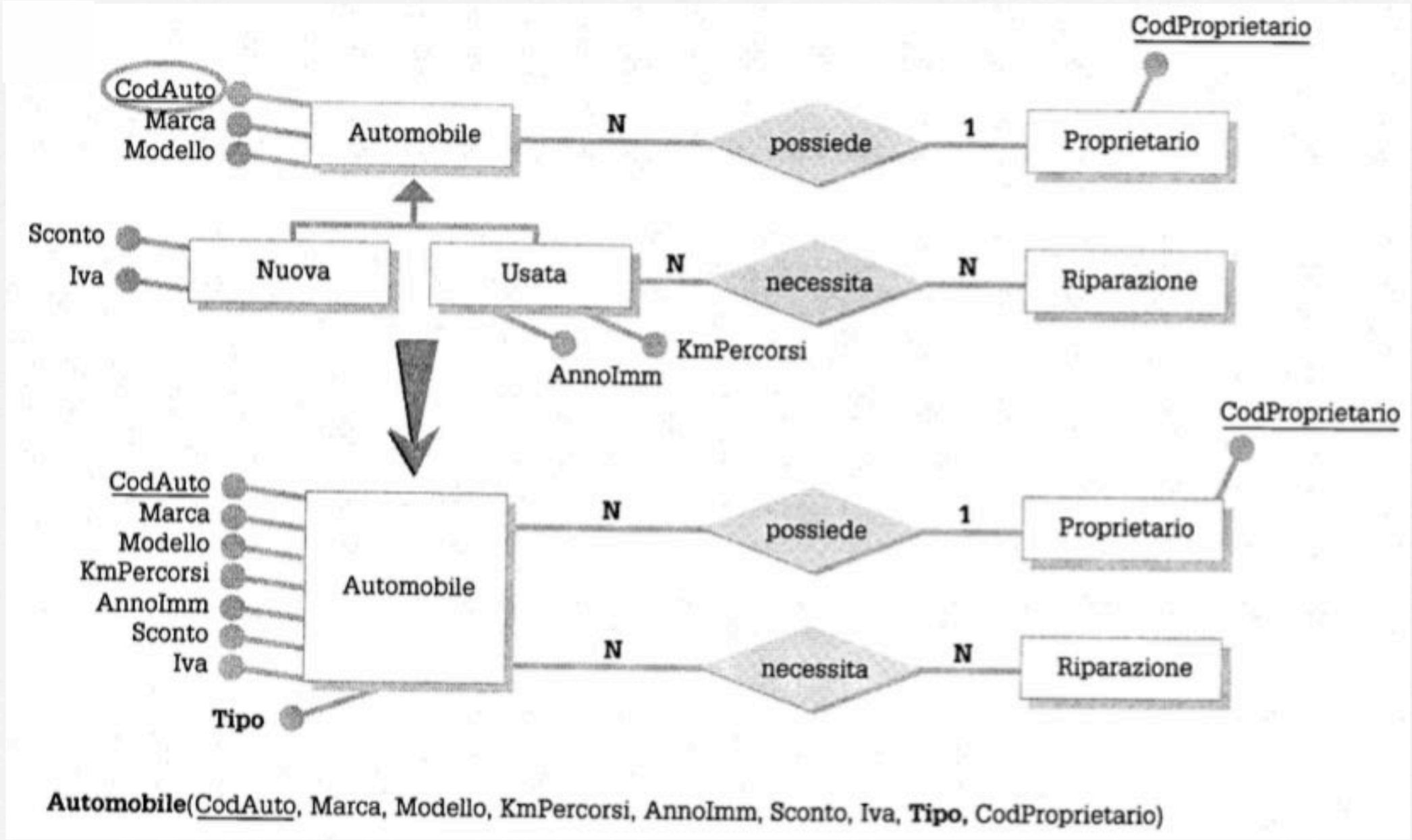
- Per rappresentare un'entità generalizzazione genitore *G* e le entità specializzazioni S_1, …, S_n:



è sufficiente accorpare (eliminandole) le entità S_1, …, S_n in *G*.

- Gli attributi delle entità eliminate (e le partecipazioni ad associazioni) vengono aggiunti all'entità *G*, alla quale viene anche aggiunto un attributo **Tipo** che serve a distinguere la provenienza (da S_1, …, S_n) di ogni n-upla.

Associazioni per generalizzazione

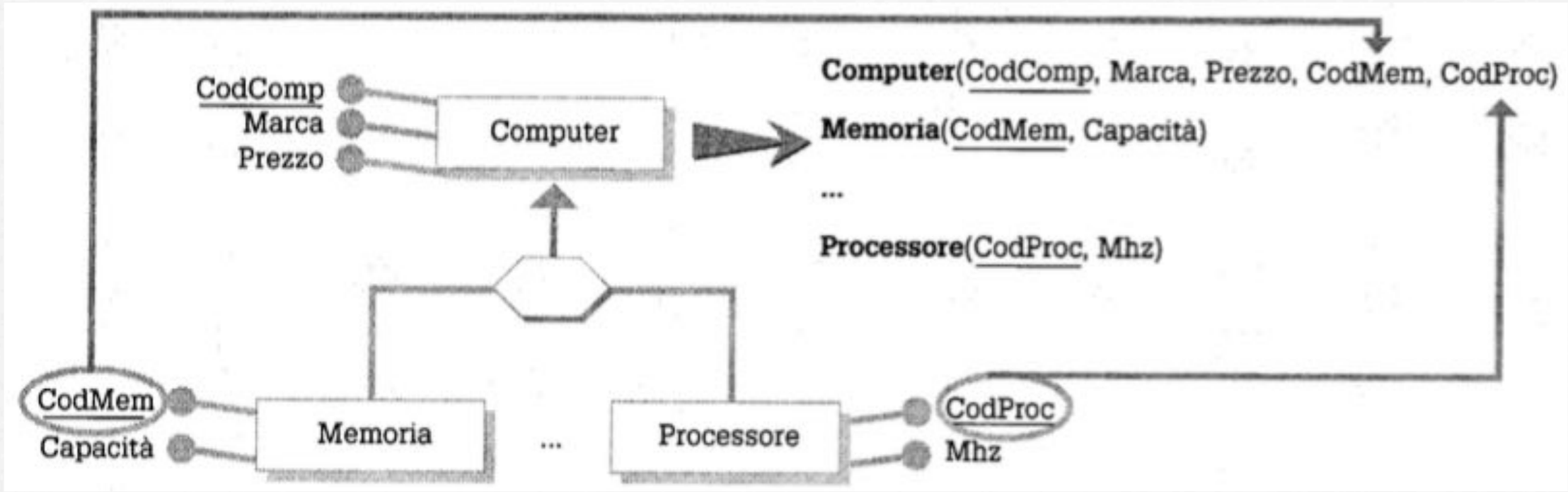


Automobile(CodAuto, Marca, Modello, KmPercorsi, AnnoImm, Sconto, Iva, **Tipo**, CodProprietario)

Associazioni di aggregazione

- Data un'associazione di aggregazione dell'entità A e le entità componenti C_1, …, C_n per rappresentare tale associazione nel modello relazionale occorre introdurre:
  - una tabella T_{C1} (… T_{Cn}) avente gli attributi di C_1 (…C_n);
  - una tabella T_A avente gli attributi chiave K_A di A, gli altri attributi di A e gli attributi chiave K_{C1} (… K_{Cn}) di C_1 (… C_n);
  - il vincolo di integrità referenziale tra le chiavi esterne in T_A e le corrispondenti chiavi primarie in T_{C1} … T_{Cn}.

Associazioni di aggregazione



Ogni chiave esterna presente nella tabella *Computer* deve essere chiave primaria delle corrispondenti tabelle collegate.

Se l'associazione è multipla (ad es. un computer può avere più processori) va creata una ulteriore tabella di collegamento (ad es. *ComputerProcessore (CodComp, CodProc)*.