

# Basi di Dati

La progettazione delle basi di dati

# LA PROGETTAZIONE DELLE BASI DI DATI

Metodologie e modelli per il progetto

---

Progettazione concettuale 1<sup>^</sup> parte

# Progetto della base di dati

- Si inserisce nel:
  - Ciclo di vita del sistema informativo
- Consiste nella:
  - Raccolta ed analisi dei requisiti
  - Progettazione
  - Implementazione
  - Validazione e collaudo
  - Funzionamento e manutenzione

# Progettazione

- *la progettazione dei dati:*
  - individua l'organizzazione e la struttura della base di dati
- *la progettazione delle applicazioni:*
  - schematizza le *operazioni* sui dati e progetta il software applicativo
  - Si concentra su come si comporta il sistema

# Metodologie di progettazione

- Distinguiamo:
  - la progettazione **concettuale**;
  - la progettazione **logica**;
  - la progettazione **fisica**.

# La progettazione concettuale

- Scopo della progettazione concettuale è tradurre la descrizione informale della realtà, risultato dell'analisi dei requisiti del DB (tipicamente sottoforma di documenti e moduli di vario genere), in uno schema formale e completo che dovrà essere indipendente dai criteri di rappresentazione del DBMS usato.
- La descrizione formale fa riferimento ad un modello concettuale, cioè un insieme di concetti e notazioni standard adatti alla rappresentazione del dominio applicativo.

# Progettazione Logica

- Consiste nella traduzione dello schema concettuale in termini di un determinato **modello logico** (ad esempio il modello relazionale) di dati usato dal DBMS che si intende utilizzare. Il risultato e' lo **schema logico**.
- Include anche l'**ottimizzazione** della rappresentazione in funzione delle operazioni eseguite (es. **normalizzazione**).

# Progettazione Fisica

- Si completa lo schema logico con la **specificazione dei parametri fisici** di memorizzazione dei dati. Si produce lo **schema fisico** che fa riferimento ad un **certo modello fisico** dei dati che dipende dal DBMS scelto.

# Metodologie di progettazione

- la progettazione concettuale
- la progettazione logica
- la progettazione fisica

# L'analisi dei requisiti

- Deve:
  - individuare le **proprietà** e le **funzionalità** del sistema;
  - produrre una descrizione dei **dati** coinvolti e delle **operazioni** su di essi;
  - individuare (in linea di massima) i requisiti **software** ed **hardware** del sistema.

# Output dell'analisi

- **Studio di fattibilità che stimi:**
  - i costi in termini di budget, di impegno del personale;
  - le inefficienze temporanee dovute al cambio di sistema e di modalità di lavoro;
  - i benefici in termini di riduzione dei tempi di lavoro o migliore efficienza dei processi aziendali;
- **piano di sviluppo del sistema con priorità e tempi di realizzazione.**

# Procedure per l'analisi

- Per ogni settore aziendale in esame si procede con i seguenti passi:
  - si analizza il sistema informativo esistente, si intervistano i responsabili del settore;
  - si produce una prima versione dei requisiti in linguaggio naturale, raggruppando **frasi descrittive** relative a categorie diverse di **dati e di operazioni**.

# Procedure per l'analisi

- si analizzano le descrizioni per eliminare ambiguità provocate da:
  - pluralismo di percezione
  - incompletezze di descrizione
- ambiguità del tipo:
  - Omonimie
  - Sinonimie
  - conflitti di descrizione
  - similitudini

# Procedure per l'analisi

- si ricontrollano insieme ai responsabili di settore **le frasi** relative alle varie categorie di dati e alle operazioni che li coinvolgono
- si costruisce a partire dalle frasi verificate un **glossario di termini**;
  - Il glossario tipicamente per ogni termine contiene: la descrizione, l'elenco dei sinonimi e l'elenco dei termini collegati.
- si verifica la **completezza**
  - tutti gli aspetti importanti sono stati considerati
- si verifica la **consistenza** delle specifiche vedendo se:
  - tutti i termini sono stati definiti;
  - tutti i termini compaiono in delle operazioni;
  - le operazioni fanno riferimento a termini definiti.

# Esempio: "frasi descrittive di un magazzino"

- il magazzino è composto da scaffali
- i fornitori forniscono prodotti
- i clienti ordinano prodotti
- gli scaffali contengono prodotti
- gli operai sono addetti agli scaffali

# Glossario corrispondente

<b>TERMINE</b>	<b>DESCRIZIONE</b>	<b>SINONIMI</b>	<b>LEGAME</b>
<b>fornitore</b>	<b>p. iva, denom., indirizzo, num. tel.</b>		<b>prodotto</b>
<b>cliente</b>	<b>p. iva, denom., indirizzo, num. tel.</b>	<b>acquirente</b>	<b>prodotto</b>
<b>prodotto</b>	<b>codice, nome, genere....</b>	<b>articolo voce</b>	<b>fornitore scaffale cliente</b>
<b>scaffale</b>	<b>supporto numerato</b>	<b>ripiano (incertezza)</b>	<b>operaio prodotto</b>
<b>operaio</b>	<b>dati anagrafici, matricola, qualifica</b>	<b>addetto magazziniere</b>	<b>scaffale</b>

# Il modello Entità-Relazione

- Il modello **Entità-Relazione (E-R)** è un **modello concettuale di dati** che contiene alcuni costrutti atti a descrivere la realtà in maniera semplice, indipendente dalla organizzazione dei dati nel computer.
- I costrutti sono:
  - **Entità**
  - **Associazioni (Relazioni)**
  - **Proprietà (Attributi)**
  - **Cardinalità**
  - **Identificatori**
  - **Generalizzazioni**
  - **Sottoinsiemi**

# Costrutti fondamentali

- **CONOSCENZA ASTRATTA**
  - entità
  - associazione
  - Proprietà
- **CONOSCENZA CONCRETA**
  - istanza di entità
  - istanza di associazione
  - proprietà delle istanze

# Entità - Istanza di Entità

- **Entità** sono classi di oggetti (cose, persone) che hanno proprietà comuni ai fini dell'applicazione di interesse che si intende modellare.
- **Istanza di Entità:** cosa (oggetto, persona) che esiste di per sé nel dominio applicativo, della quale si vogliono registrare fatti specifici e che può essere chiaramente identificata in modo da poterla distinguere dalle altre

# Esempi

- Entità:
  - Docente
  - Corso
  - Automobile
  - Studente
  - Volo
  - Percorso
- Istanza di Entità:
  - il docente Ferro
  - il corso Basi di Dati
  - l'auto AB111XY
  - lo studente 567345
  - il volo AZ3313
  - il percorso CT-PA

# Associazione - istanza di associazione

- Le **Associazioni** (o **Relazioni**) rappresentano legami logici fra due o più entità dell'applicazione.
  - Possono esserci più relazioni fra le stesse entità (associazioni ricorsive).
- **Istanza di Associazione:** fatto che descrive un'azione o una situazione e che stabilisce legami tra istanze di entità (associa, mette in relazione)

# Esempi

- Associazione:
  - Insegna
  - Appartiene
  - Ordina
  - Lavora
  - Viaggia sulla tratta
- Istanza di associazione:
  - Ferro insegna Basi di dati 2
  - Sicilia appartiene all'Italia
  - La ditta Rossi ordina PC
  - Bianchi lavora al magazzino 4
  - il TIR 542 viaggia sulla tratta CT-PA

# Proprietà delle istanze

- Dette anche attributi delle istanze: sono fatti che descrivono le caratteristiche delle istanze di entità e le caratteristiche delle istanze di associazione
- Le proprietà delle istanze assumono VALORI

# Esempi di proprietà delle istanze

- Proprietà di istanze di entità:
- Ferro ha nome Alfredo
- Il recapito della ditta Rossi è via Etnea 22
- Formazione Analitica 1 si tiene al primo anno

# Esempi di proprietà delle istanze

- Proprietà di istanze di associazione
- Ferro insegna basi di dati dall'anno 1995
- Bianchi ha lavorato 3 ore al magazzino 4
- La ditta Rossi ordina 15 PC
- Pappalardo supera Formazione Analitica 1 con 27

# Modello E-R

- Il modello E-R usa simboli grafici per favorire l'immediatezza della comprensione;
- Gli schemi E-R sono schemi grafici;
- Si possono aggiungere frasi di commento per rappresentare le caratteristiche non modellabili (ad esempio, vincoli complessi).

# Notazioni E-R: entità

**nome dell'entità**

**esempi:**

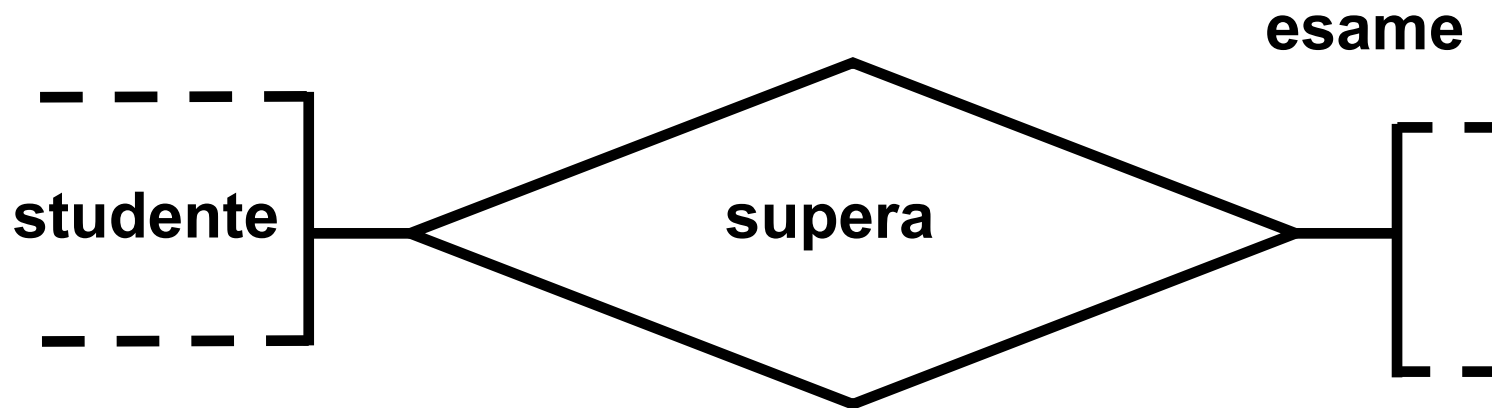
**studenti**

**materie**

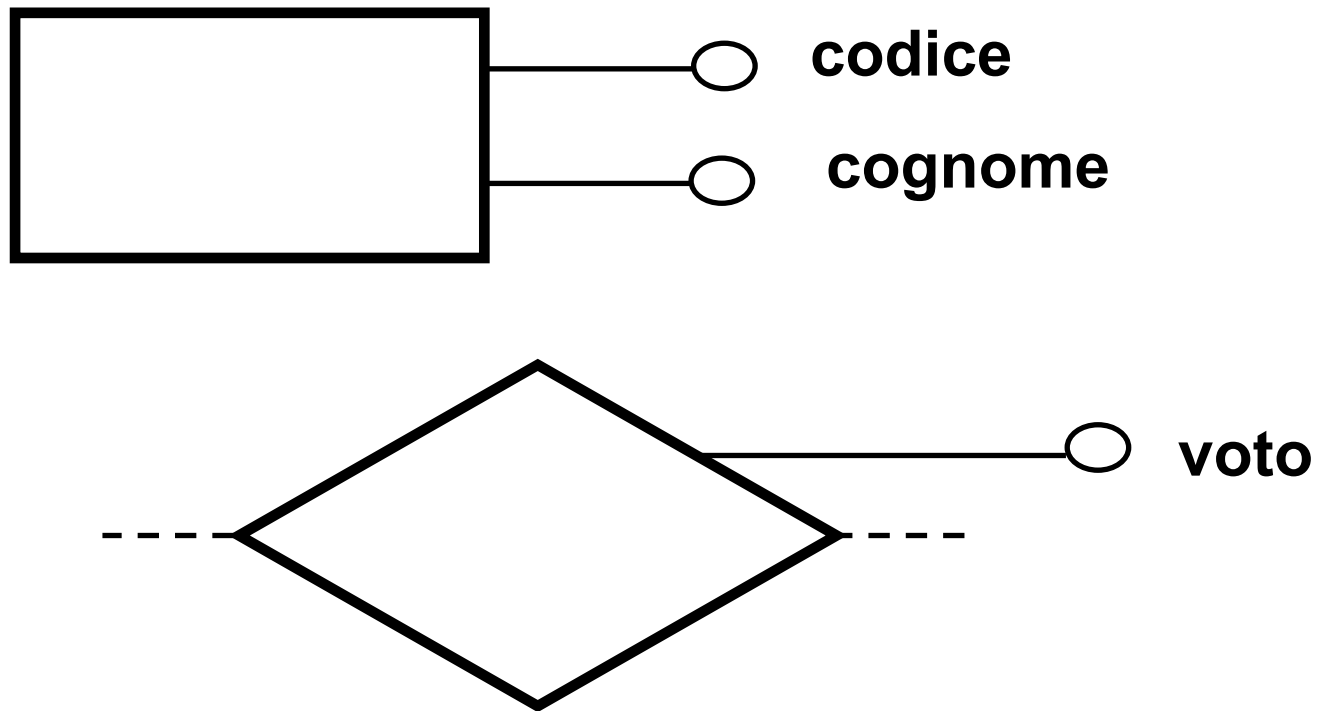
# Notazioni E-R: associazioni



**esempio:**



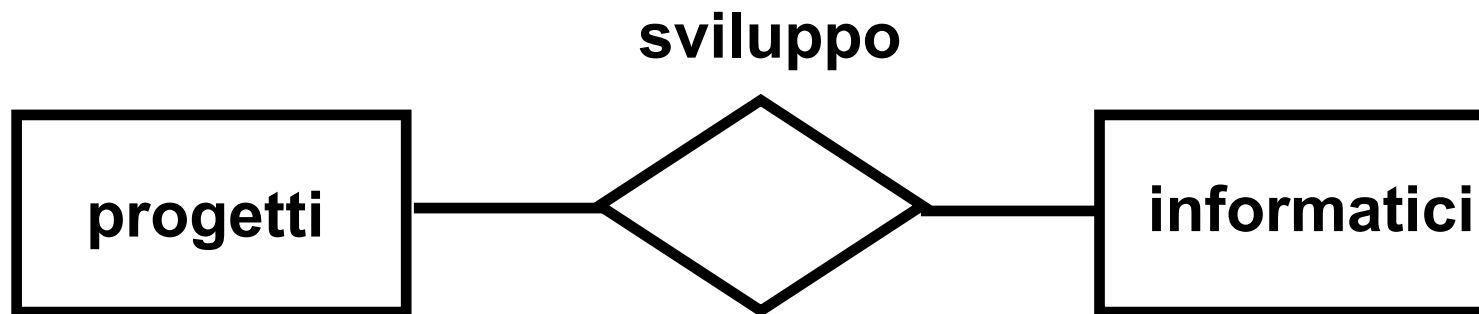
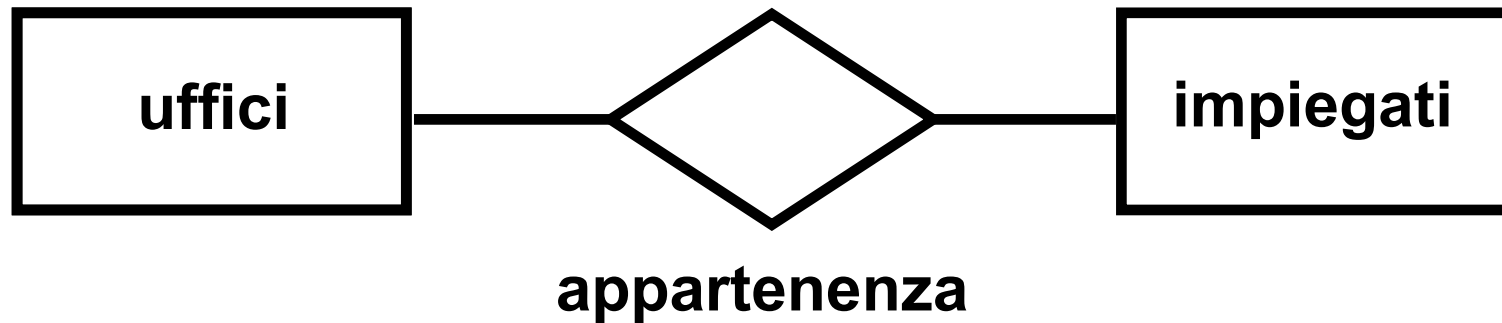
# Notazioni E-R: proprietà

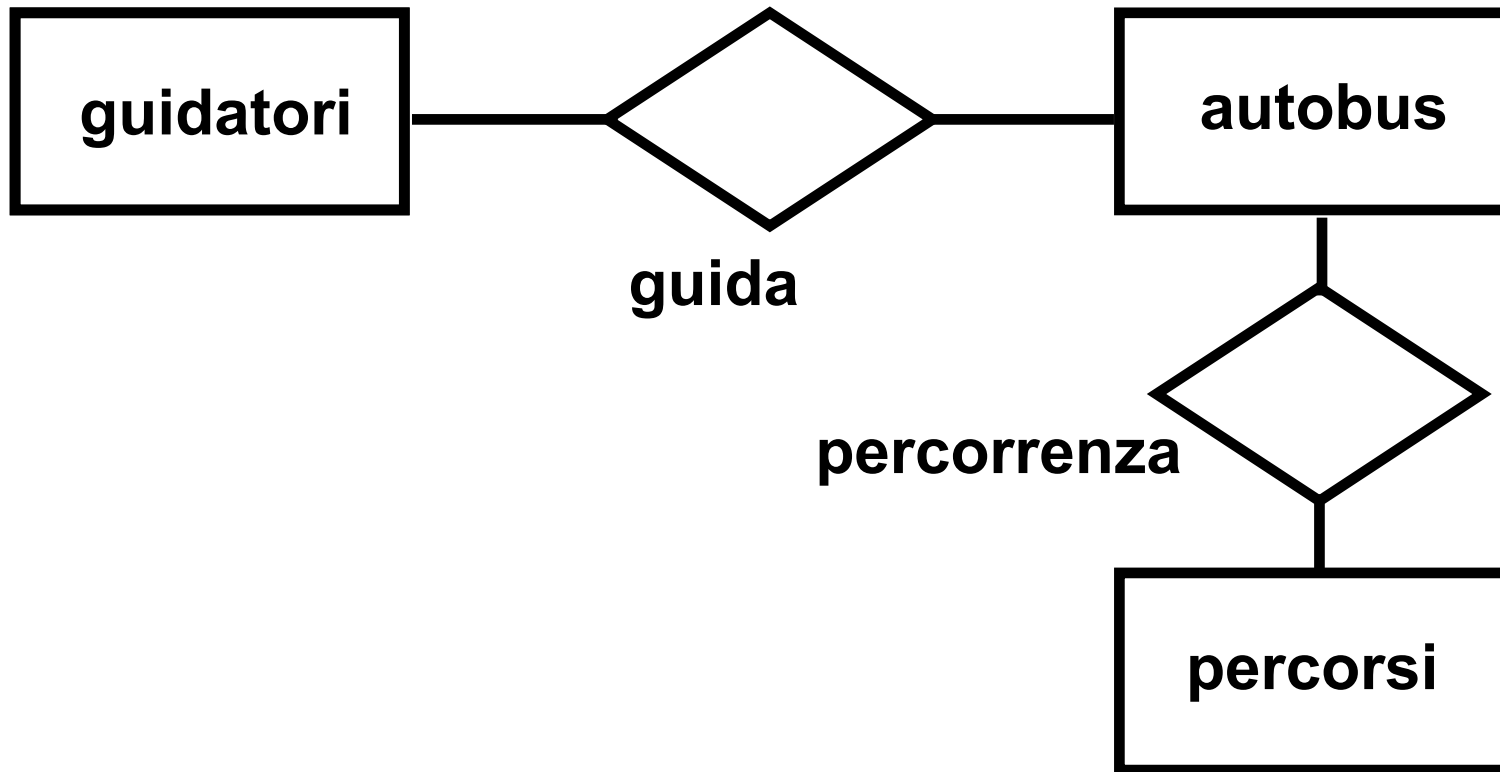


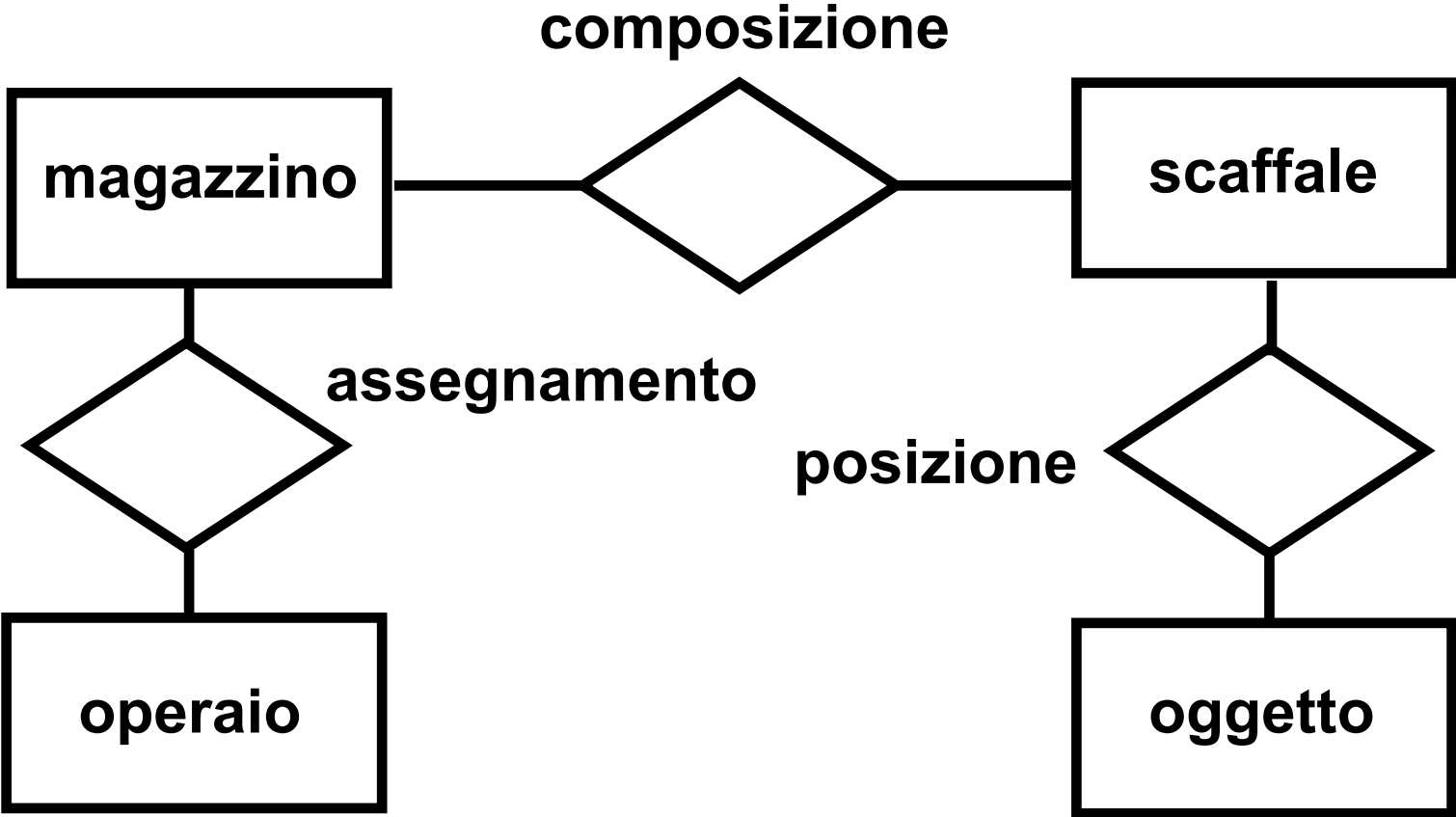
# Notazioni E-R: schemi scheletro

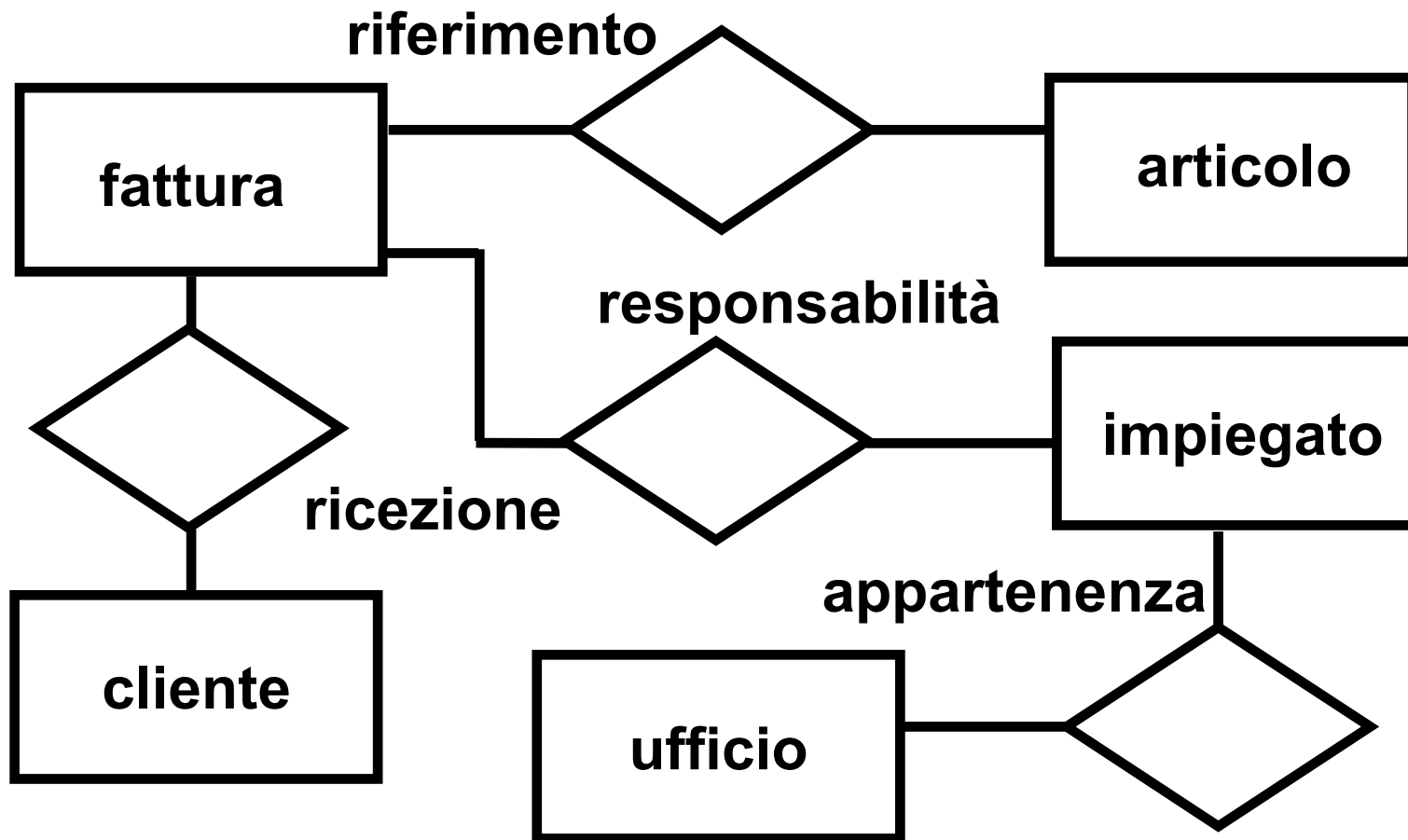
- **Gli schemi scheletro** descrivono una prima struttura di massima dello schema senza indicazioni sul TIPO delle entità e delle associazioni
- Descrivono in generale i collegamenti tra le entità di interesse e le associazioni che le legano

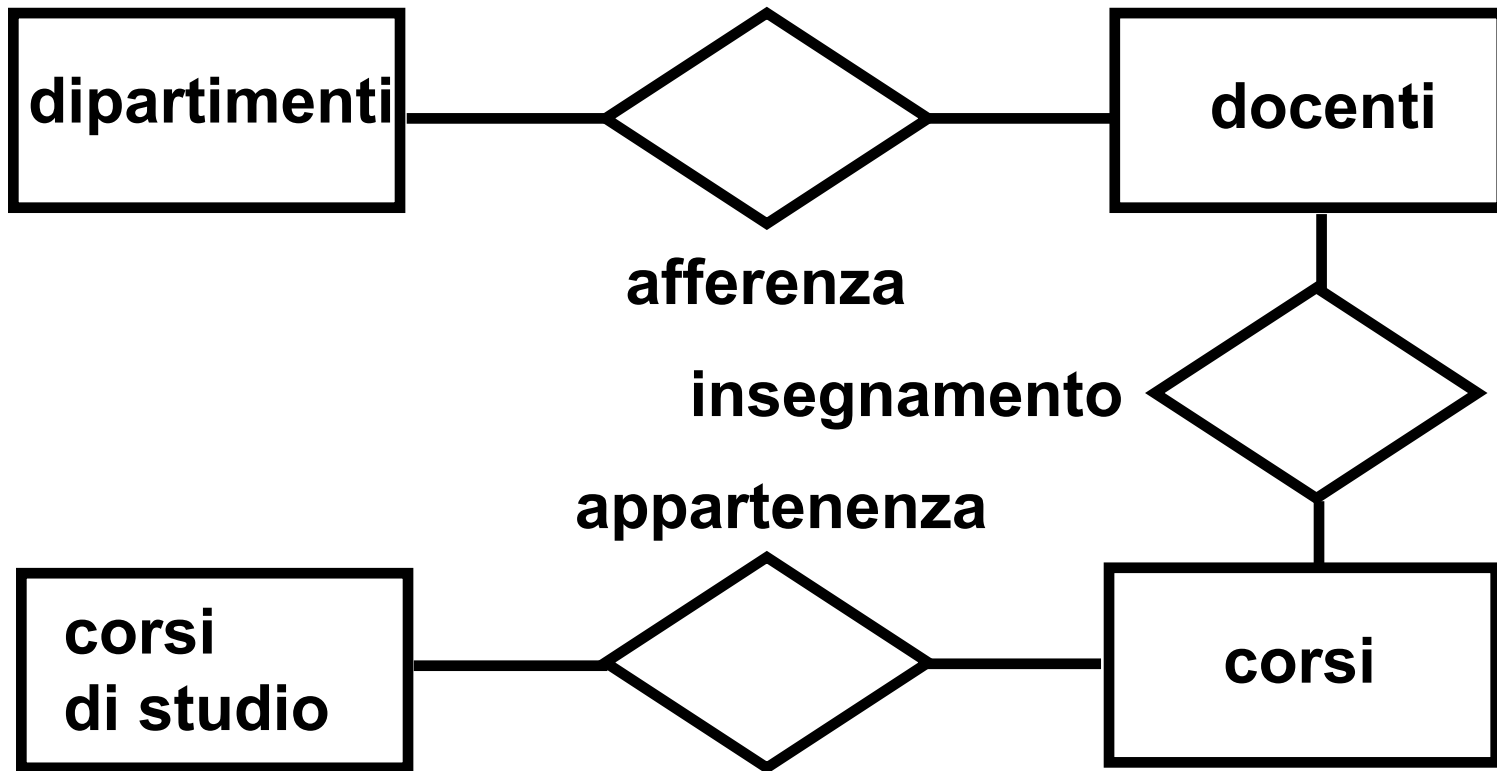
# Schemi scheletro (esempi)



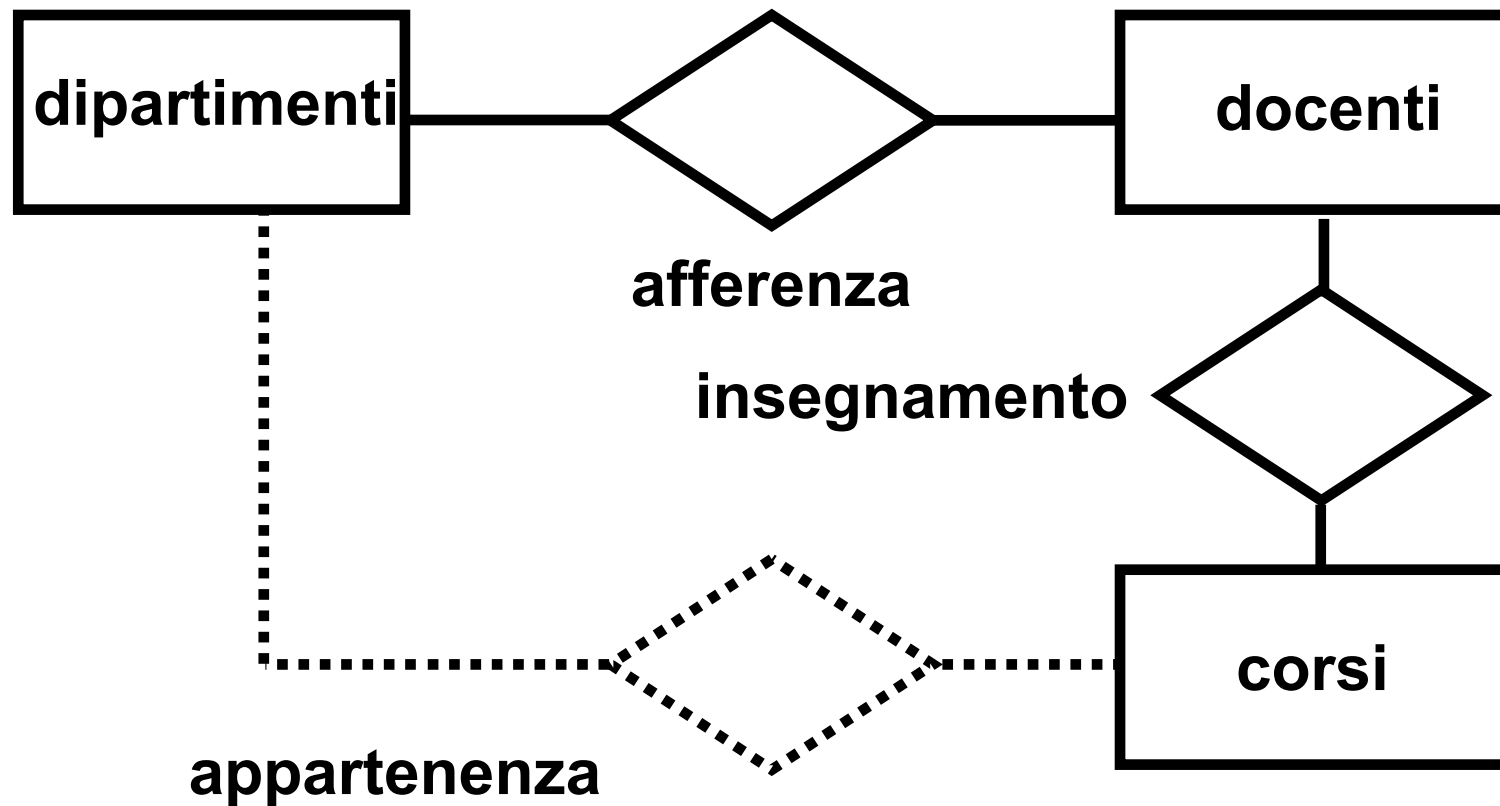




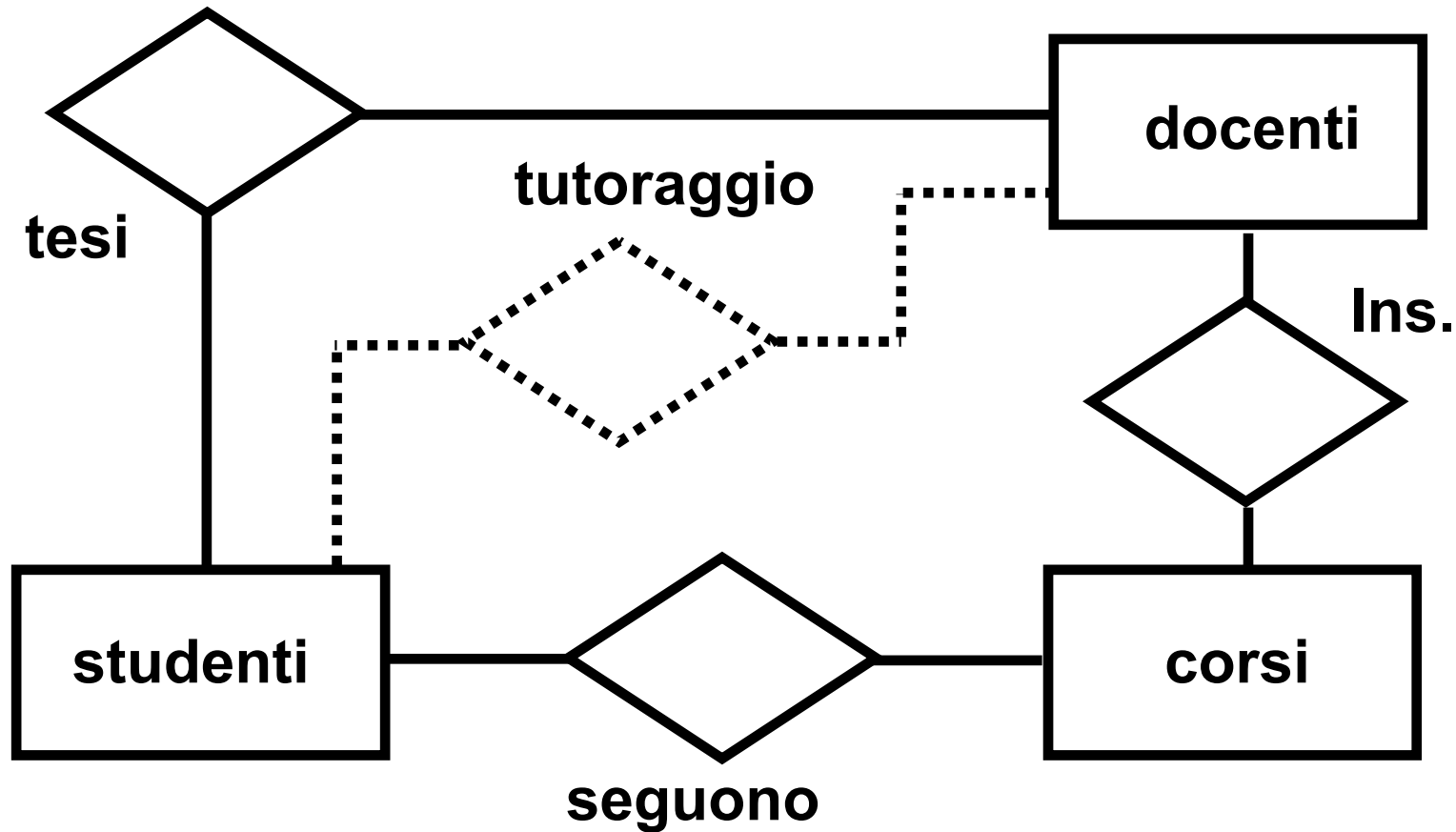




# Associazioni ridondanti



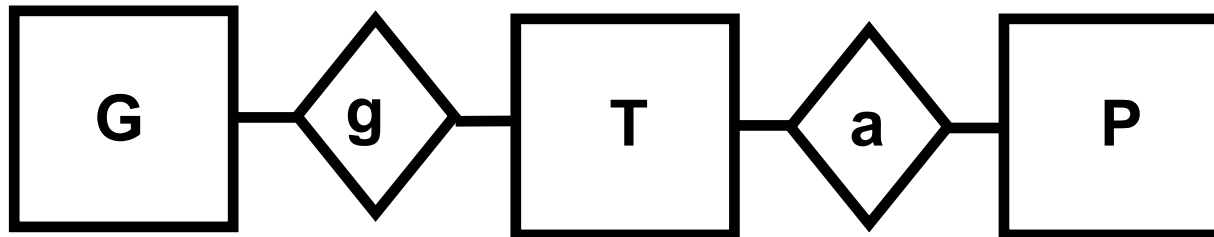
# Associazioni ridondanti?



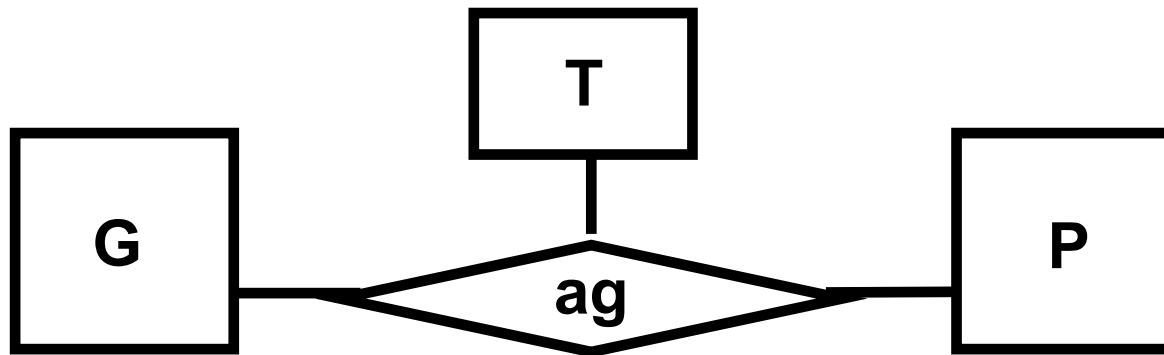
# Risoluzione delle ambiguità di modellazione

**i Guidatori guidano TIR,**

**i TIR sono assegnati a Percorsi**



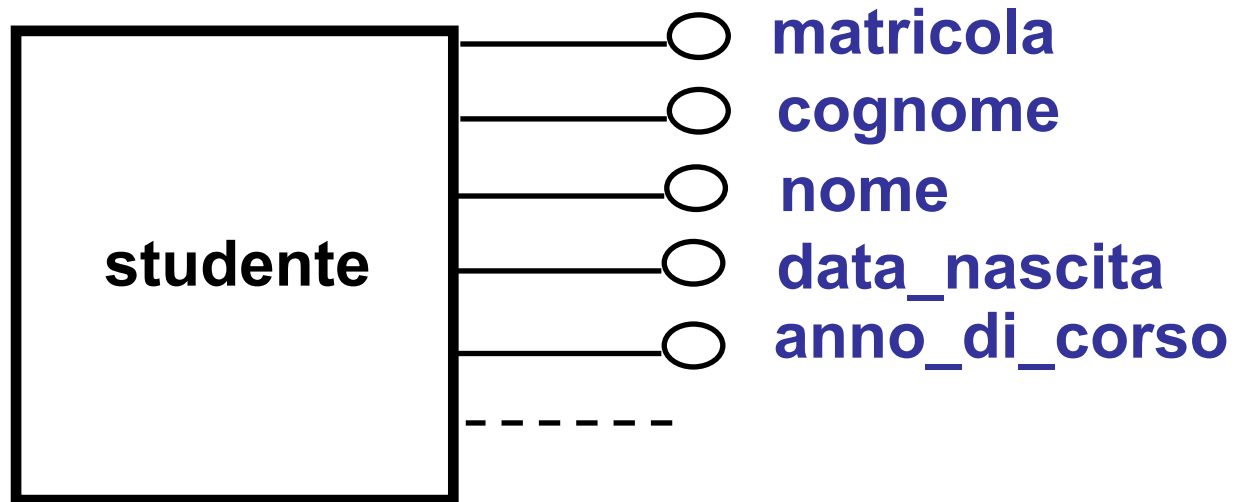
**i Guidatori guidano TIR su Percorsi**



# Le proprietà

- Gli schemi scheletro descrivono in generale i collegamenti tra le entità e le associazioni;
- Le entità e le associazioni devono essere descritte attraverso l'aggregazione di proprietà;

# Identificazione delle proprietà



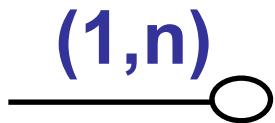
# Proprietà

scalare (semplice, ad un sol valore)



es.: matricola, cognome, voto

multipla (sono ammessi  $n$  valori)



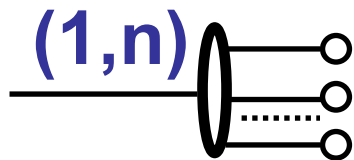
es.: titolo di studi, targa auto

composta

$(1,1)$

es.: data (gg,mm,aaaa), indirizzo,  
(denominazione, civico, cap)

multipla composta



es.: telefono (stato,  
città, numero)

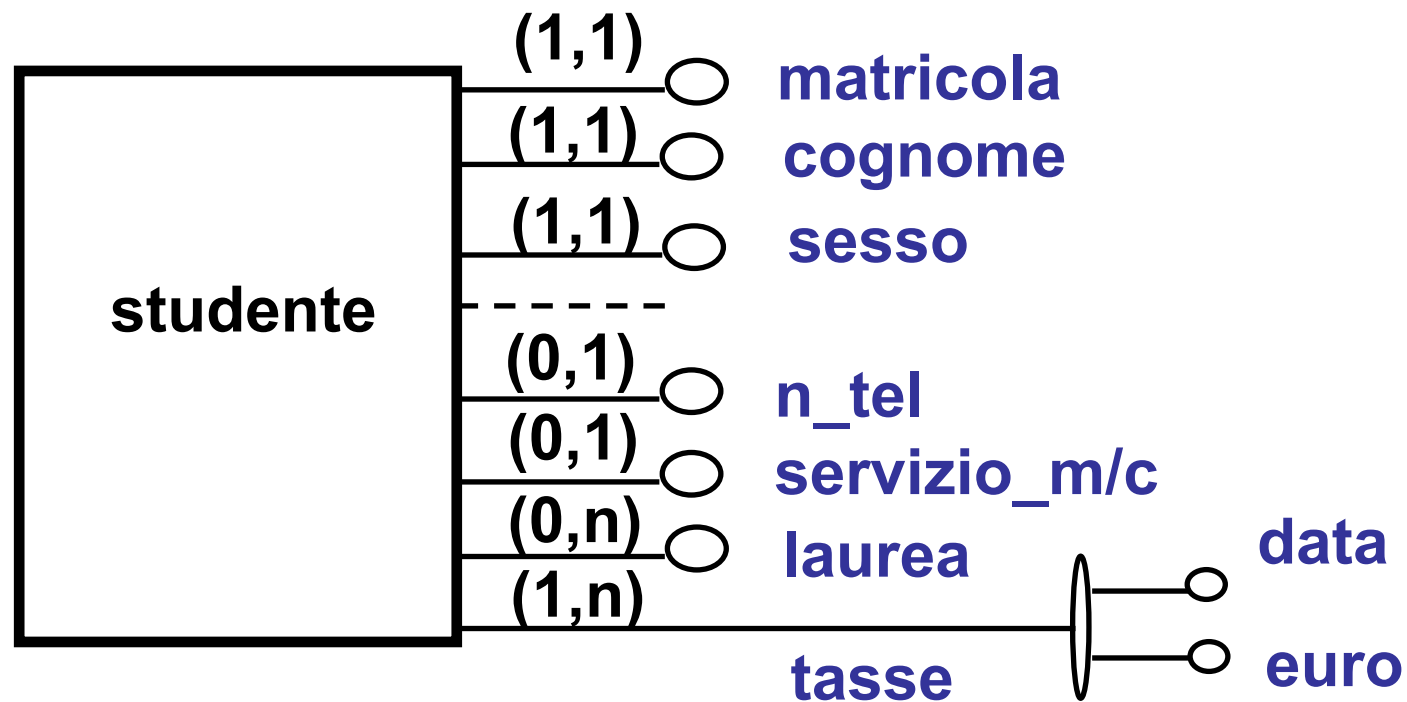
Il simbolo  $(n,m)$  esprime la **cardinalità** minima e massima della proprietà

# Proprietà opzionali

**Proprietà opzionale** (è ammessa la  
“non esistenza del valore”)

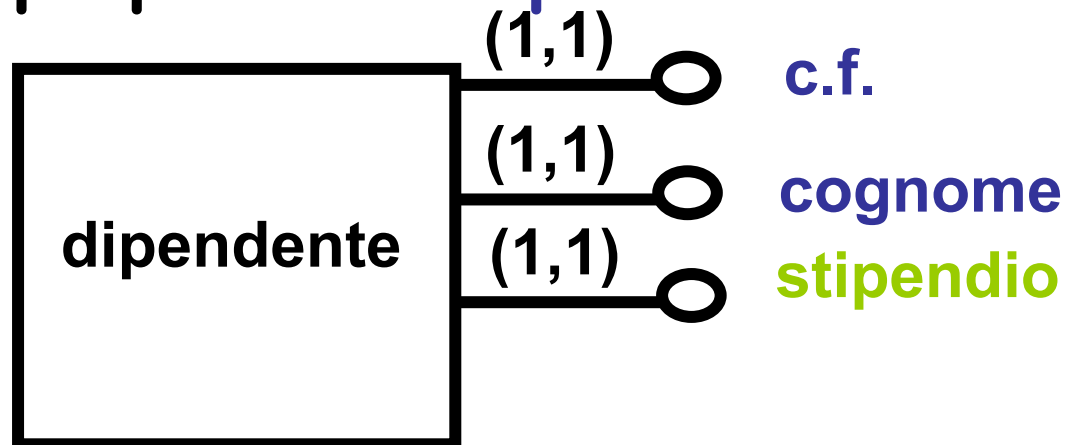
$(0,n)$    $(0,1)$   es.: tel., qualifica, targa

# Esempio di specifica delle proprietà



# Proprietà calcolate

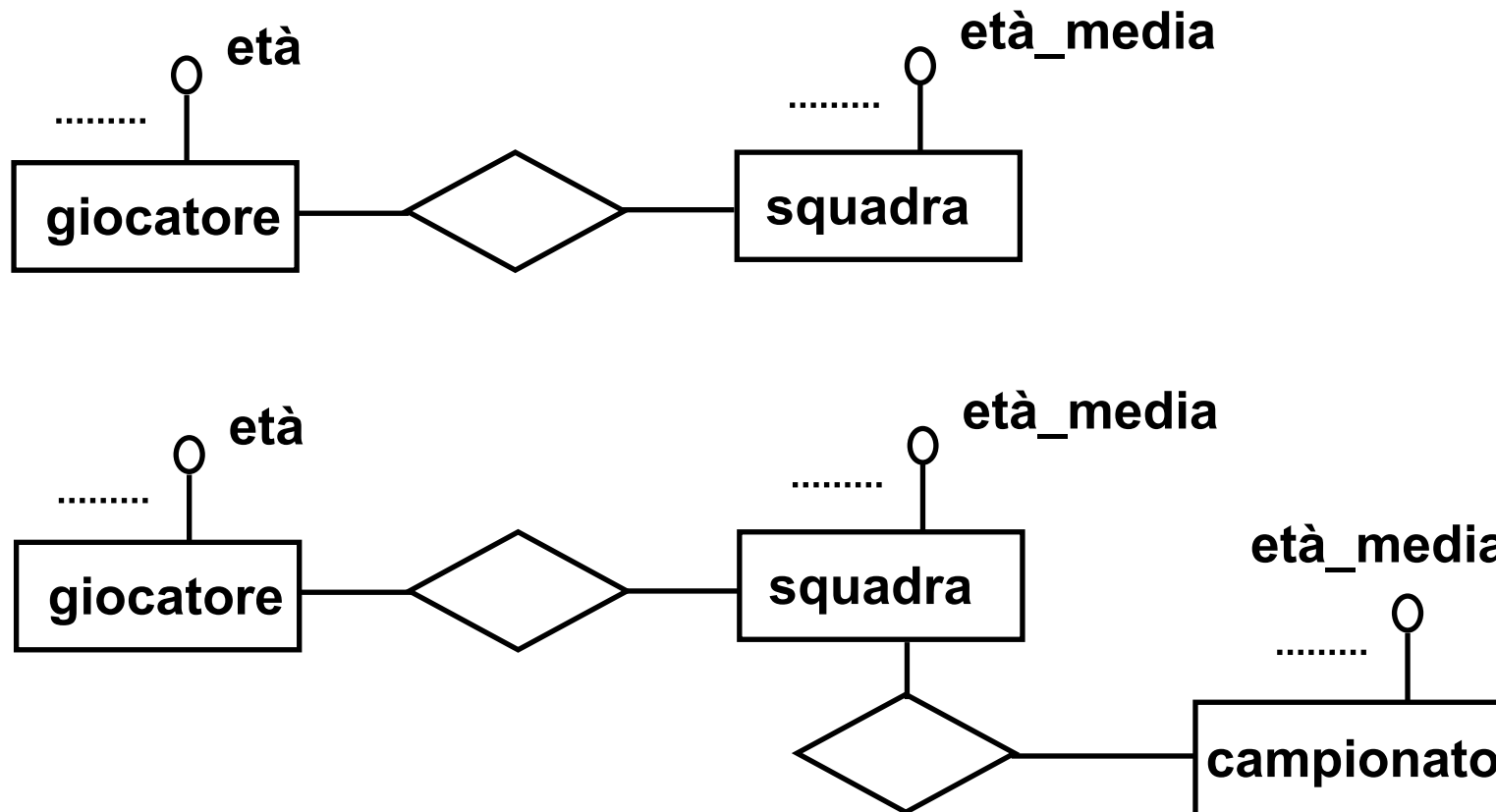
Proprietà **calcolata (derivata)**: il valore è calcolato con un algoritmo, in alternativa la proprietà è **esplicita**.



NB: la regola di calcolo va espressa a parte, usando un qualche linguaggio di specifica:  
`stipendio = salario_giornaliero*presenze`

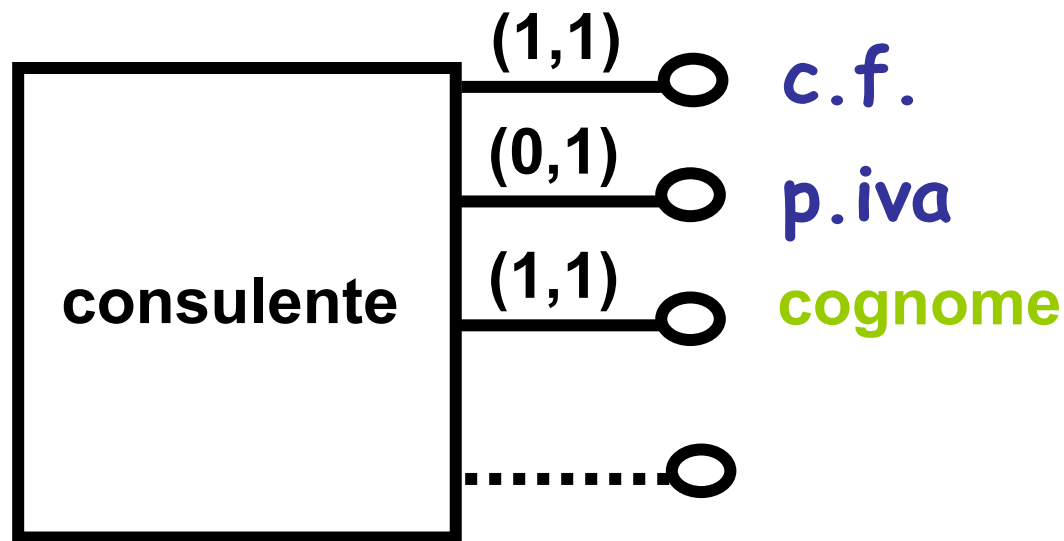
# Proprietà di sintesi

- valori medi , max e min ecc.



# Unicità dei valori

Proprietà **unica**: tutte le istanze della classe hanno valore diverso, in alternativa la proprietà è **non unica**:

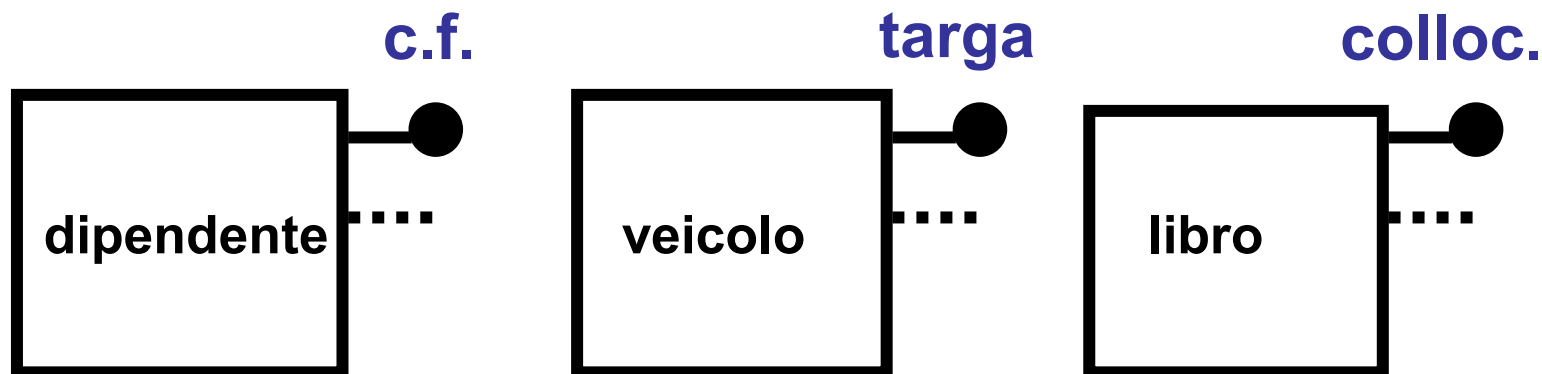


# Proprietà chiave

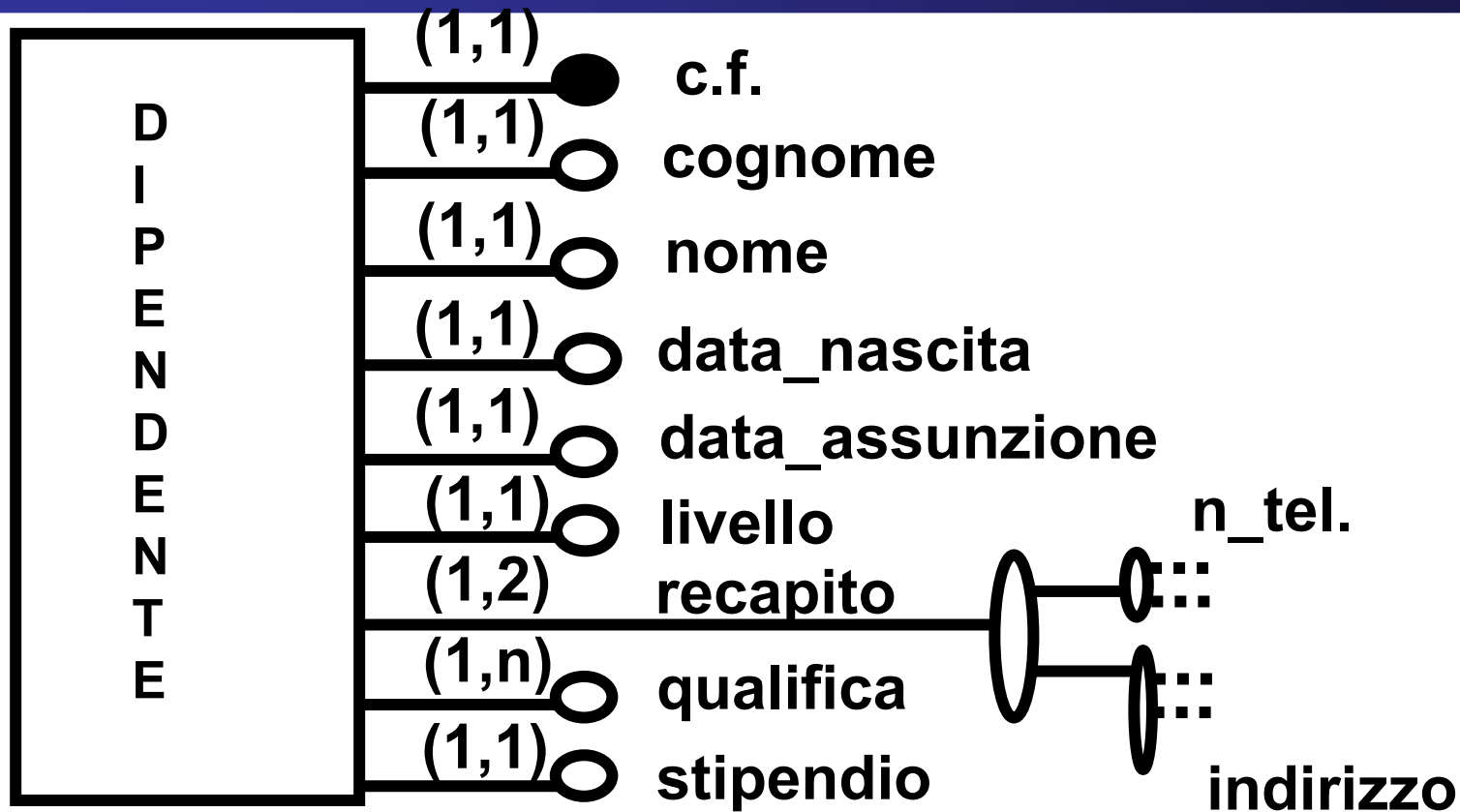
Una proprietà (attributo) **chiave** identifica in modo univoco la **singola istanza** di entità (o di associazione):

- obbligatoria, unica, esplicita
- può essere composta
- in generale non è modificabile

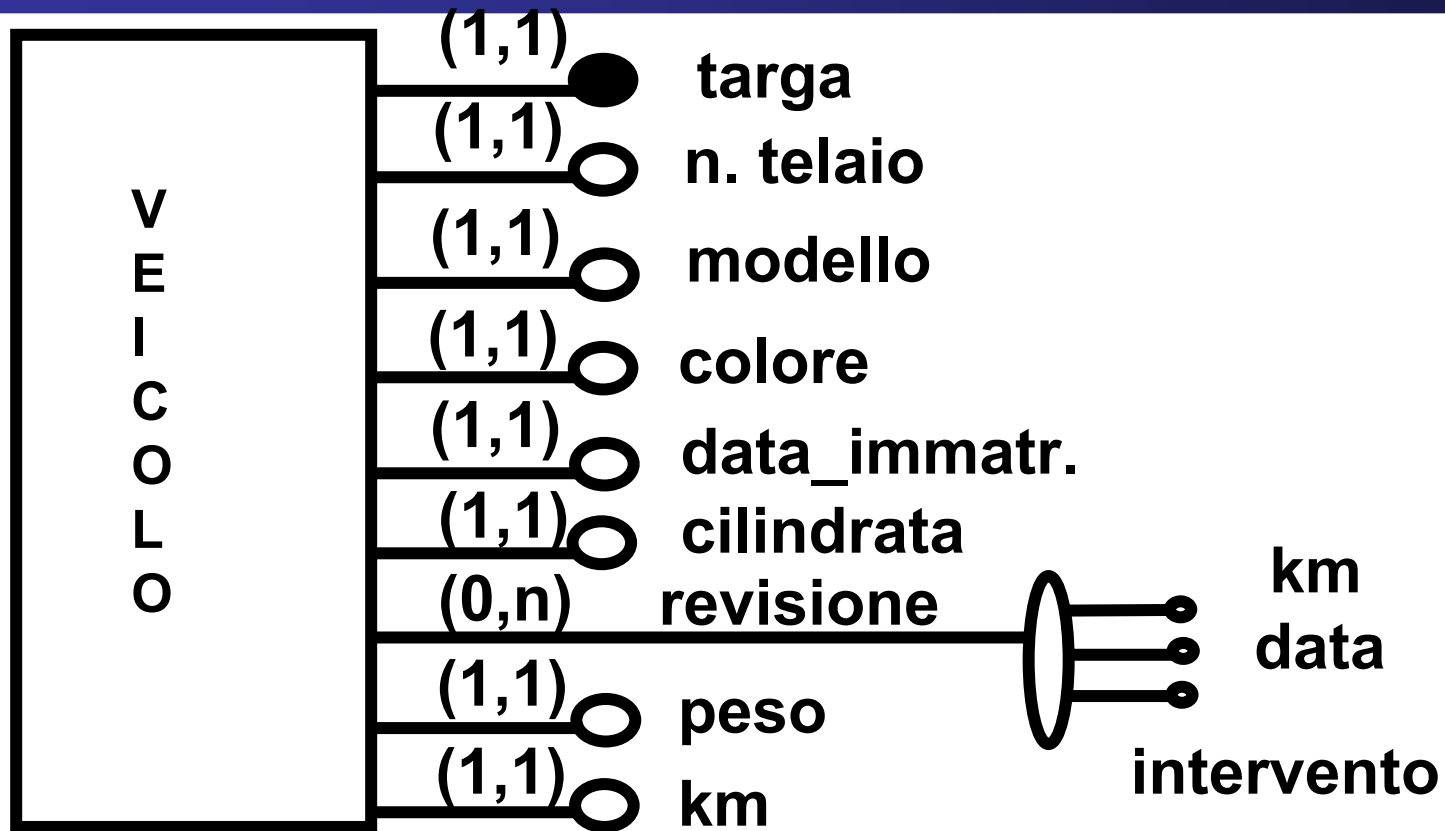
Simbolo = —●



# Esempio

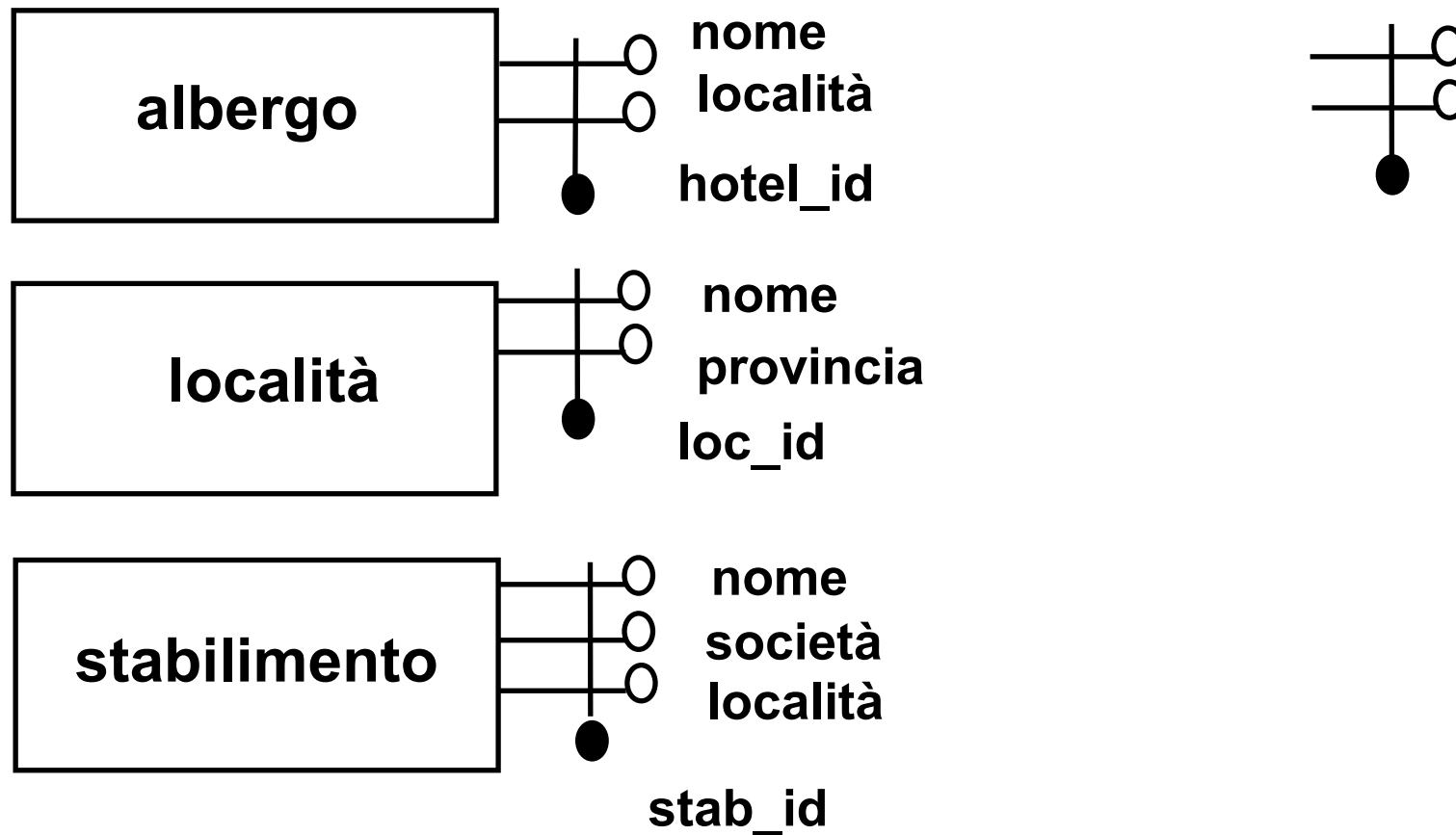


# Esempio



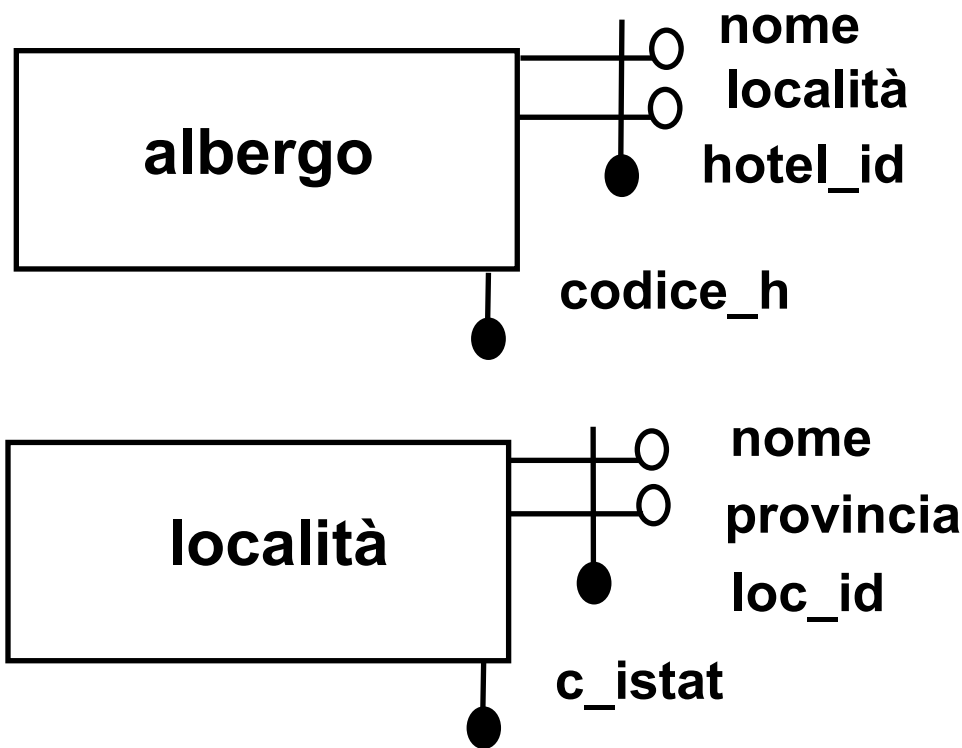
# Chiave composta

La chiave di un'entità può essere composta



# Chiavi alternative

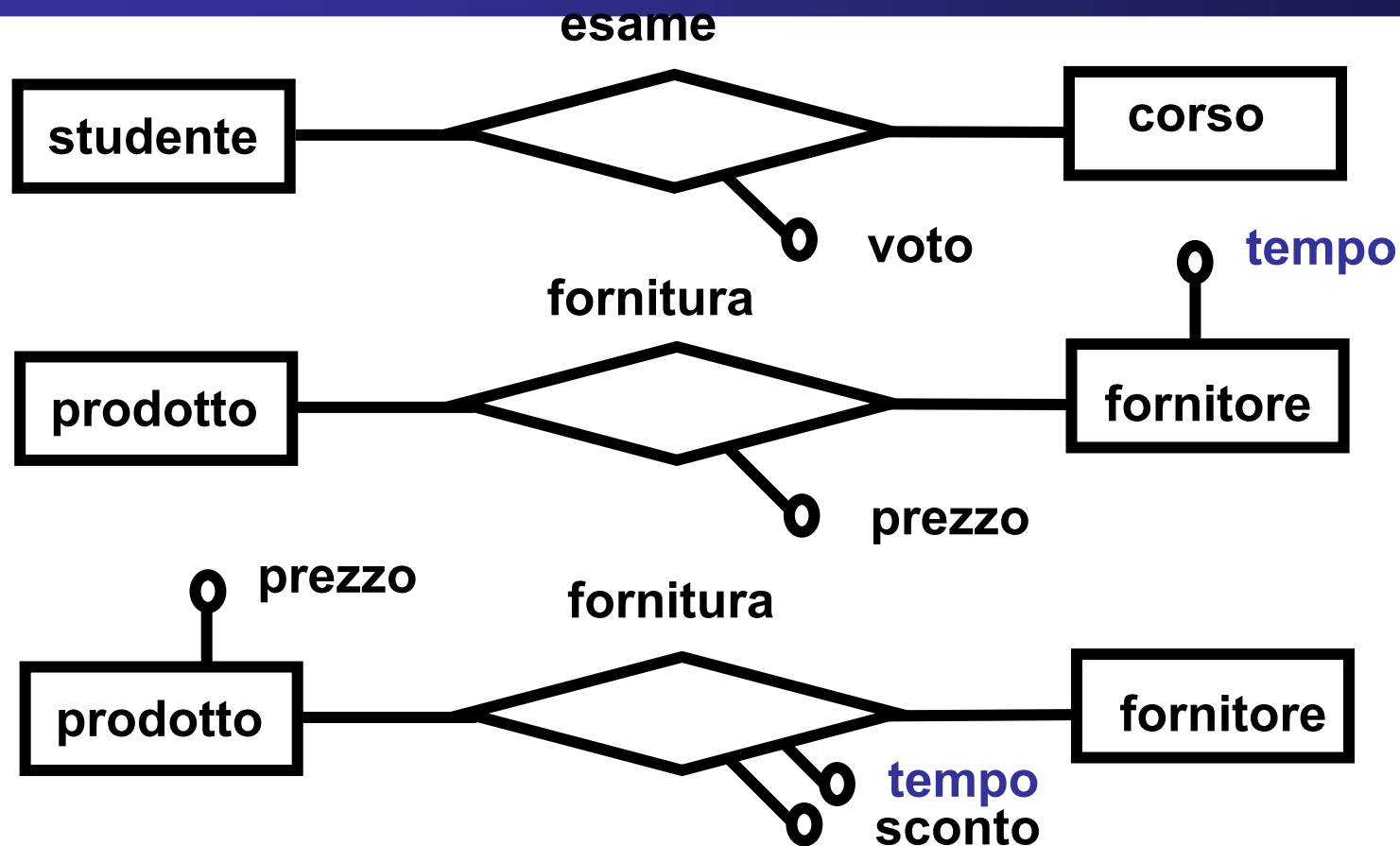
Chiavi **alternative** per una stessa entità



# Criteri di scelta della chiave

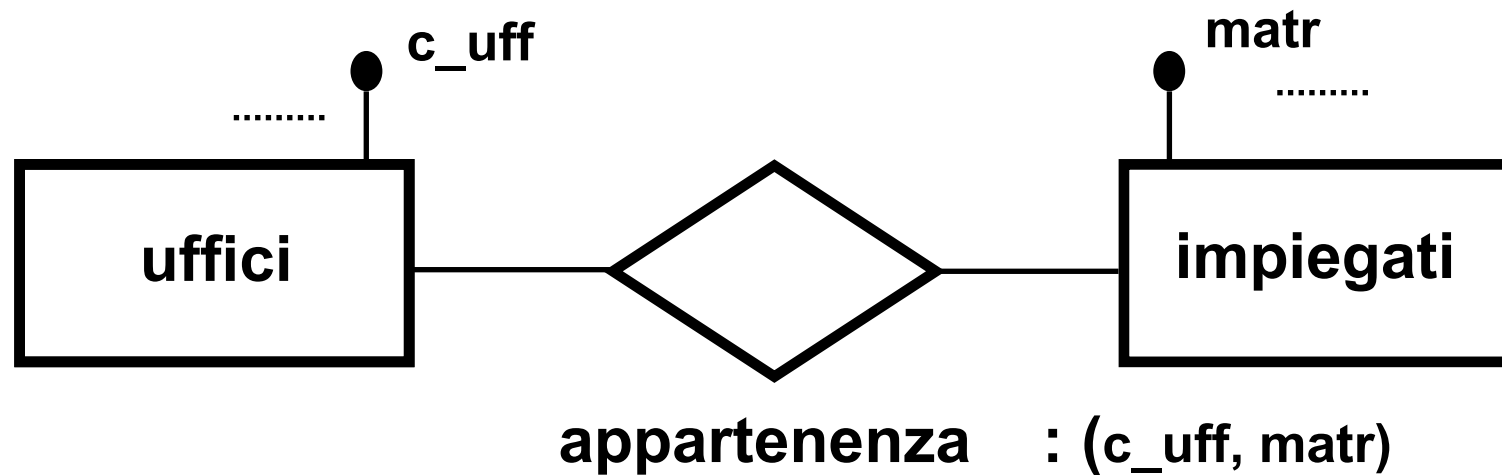
- Scelta condizionata da fattori esterni
  - es.: per gli studenti: codice fiscale o matricola?
  - velocità di digitazione (matricola)
  - controllo validità (codice fiscale)
- Spesso si fa uso di valori progressivi assegnati automaticamente dal sistema garantendo l'unicità (AUTO\_INCREMENT/SEQUENCE)

# Proprietà delle associazioni

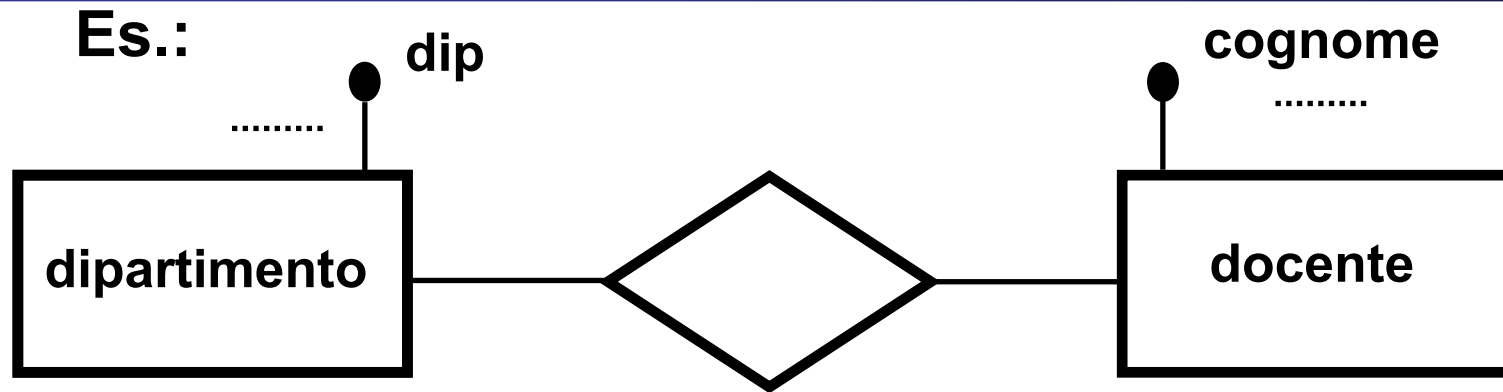


# La chiave delle associazioni

La chiave è sempre composta ed è definita come l'insieme delle chiavi delle entità partecipanti, es.:



# Esempio

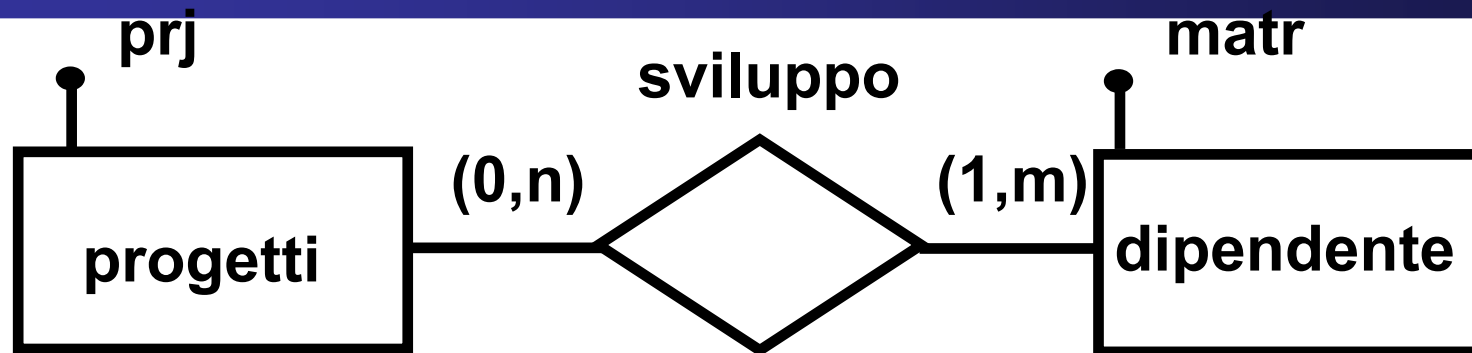


Afferenza : (dip, cognome)

# Cardinalità delle associazioni

- per **cardinalità** si intende il numero di volte che una data istanza di entità deve o può partecipare alla associazione
  - (1,1) : obbligatoria, una sola volta
  - (1,n) : obbligatoria, almeno una volta
  - (0,1) : opzionale, una sola volta
  - (0,n) : opzionale, n volte

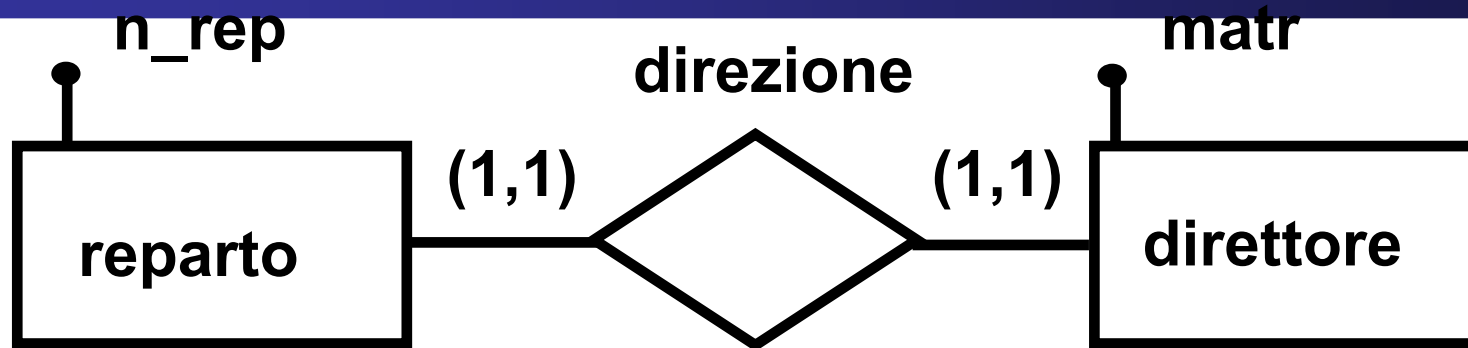
# Associazioni N:M



- **Con vincolo:** un dipendente **deve** partecipare ad almeno un progetto (1,m)
- **Senza vincolo:** ad un progetto **possono** partecipare dipendenti, ma può esistere anche un progetto senza dipendenti (0,n)

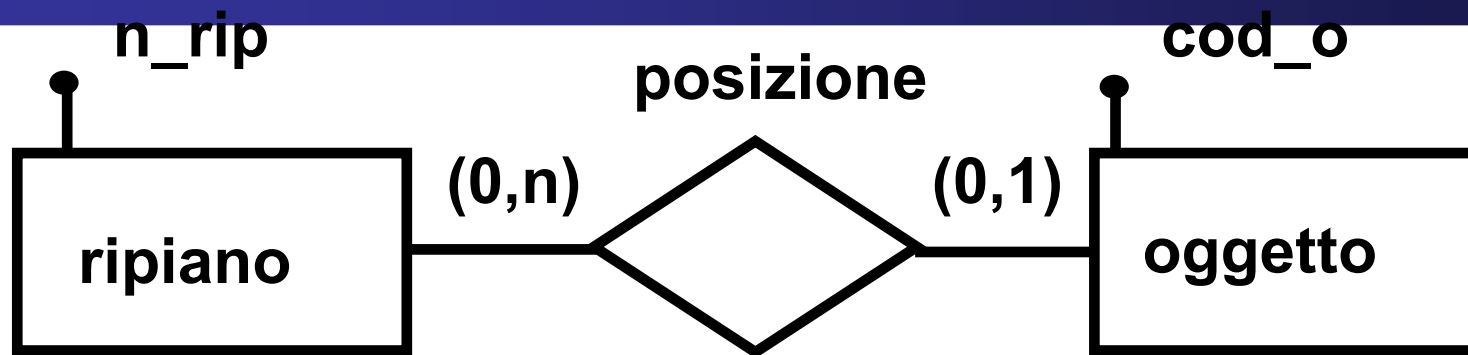
Chiamata anche: **associazione molti a molti**

# Associazioni 1:1



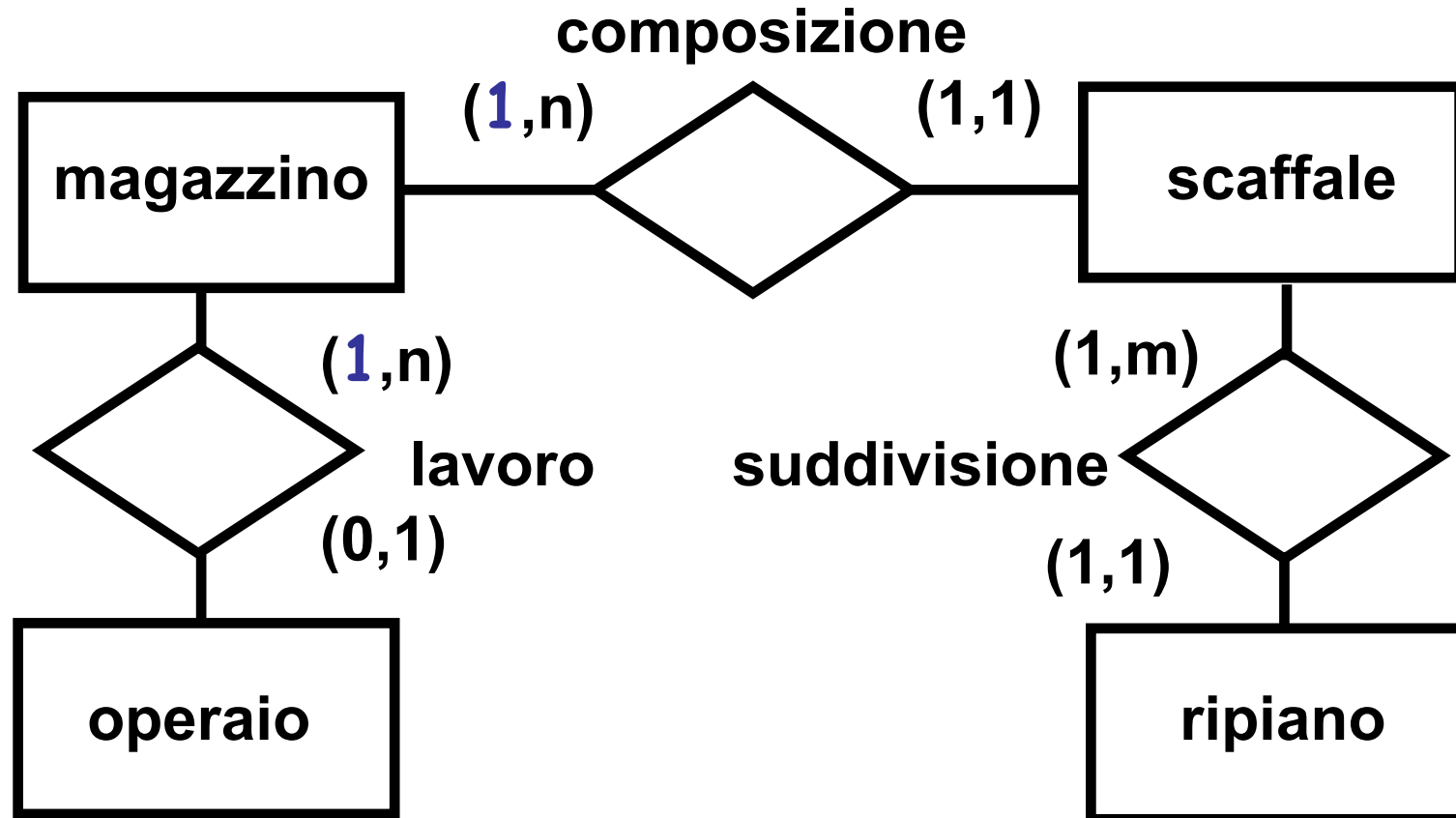
- **Con doppio vincolo:**
  - un reparto **deve** avere un direttore ed il direttore è **uno solo** (1,1)
  - un direttore **deve** dirigere uno ed **un solo** reparto (1,1)
- Il vincolo diventerebbe (0,1) se avessimo la generica entità "impiegato" al posto dell'entità "direttore" (alcuni impiegati non dirigono reparti)

# Associazioni N:1

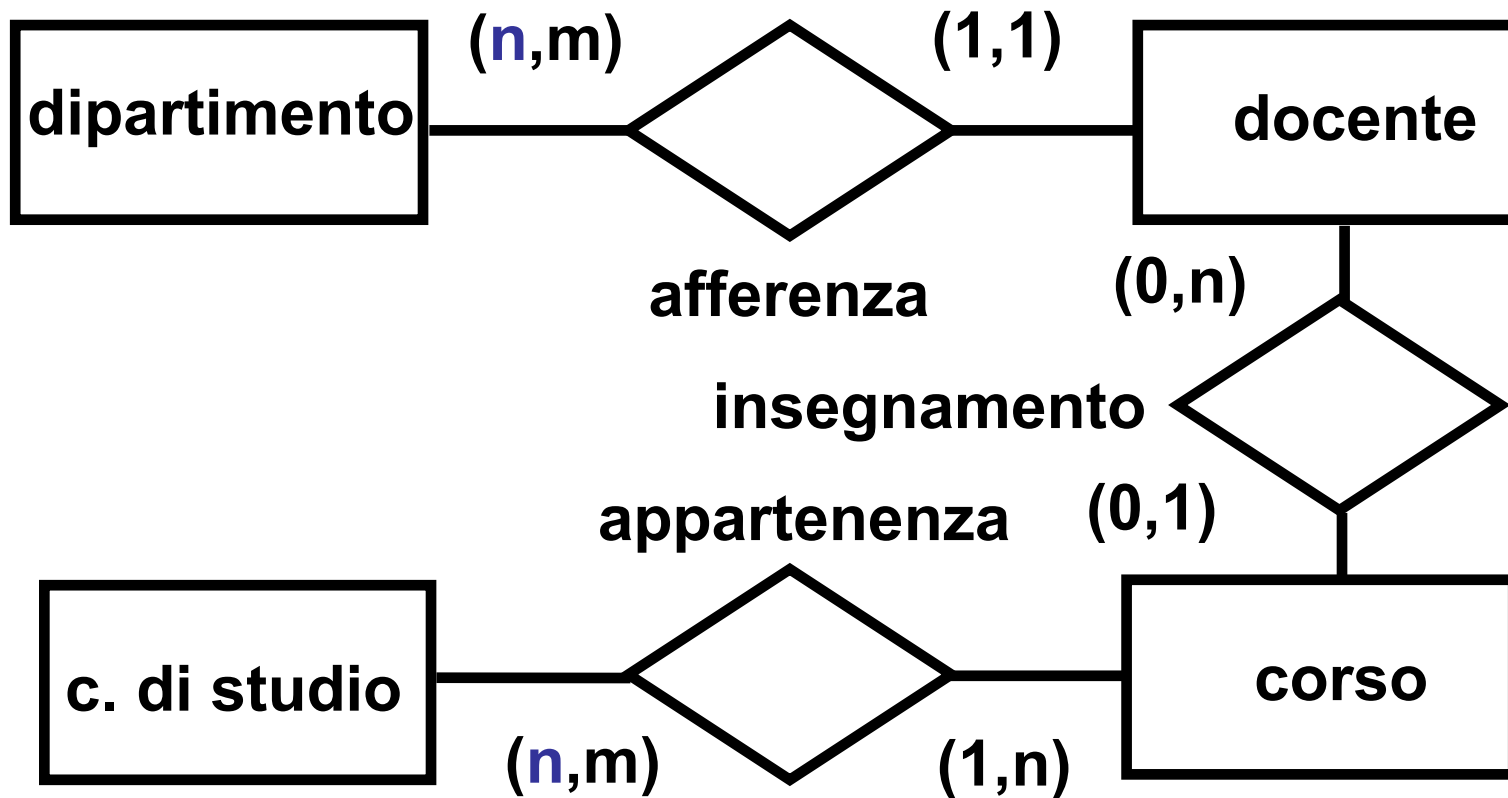


- **Con vincolo:** un oggetto **può** stare su un ripiano, ma su **un solo** ripiano **(0,1)**
- **Senza vincolo:** su un ripiano **possono** stare n oggetti
- È possibile vincolare la partecipazione, per esempio ponendo  $n = 5$ , **non più di 5** oggetti per ripiano **(0,5)**
- Chiamata anche: associazione molti a uno

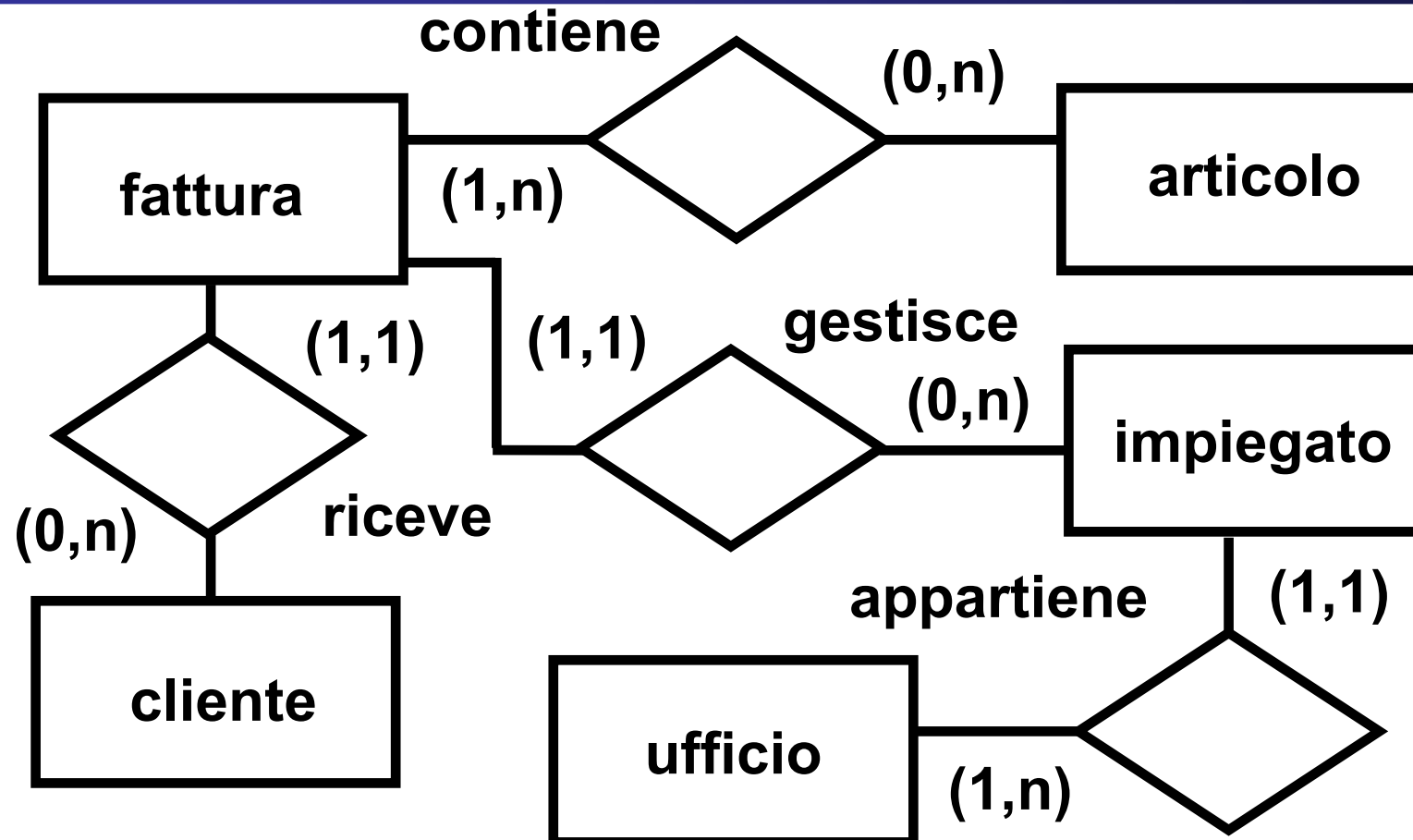
# Gestione magazzino



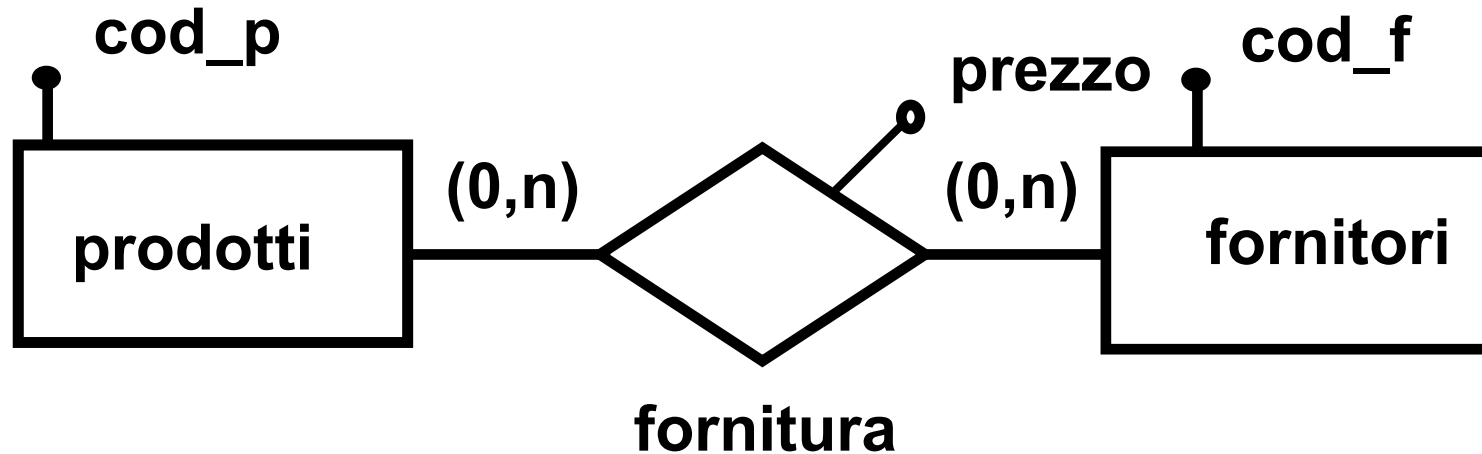
# Base dati universitaria



# Contabilità

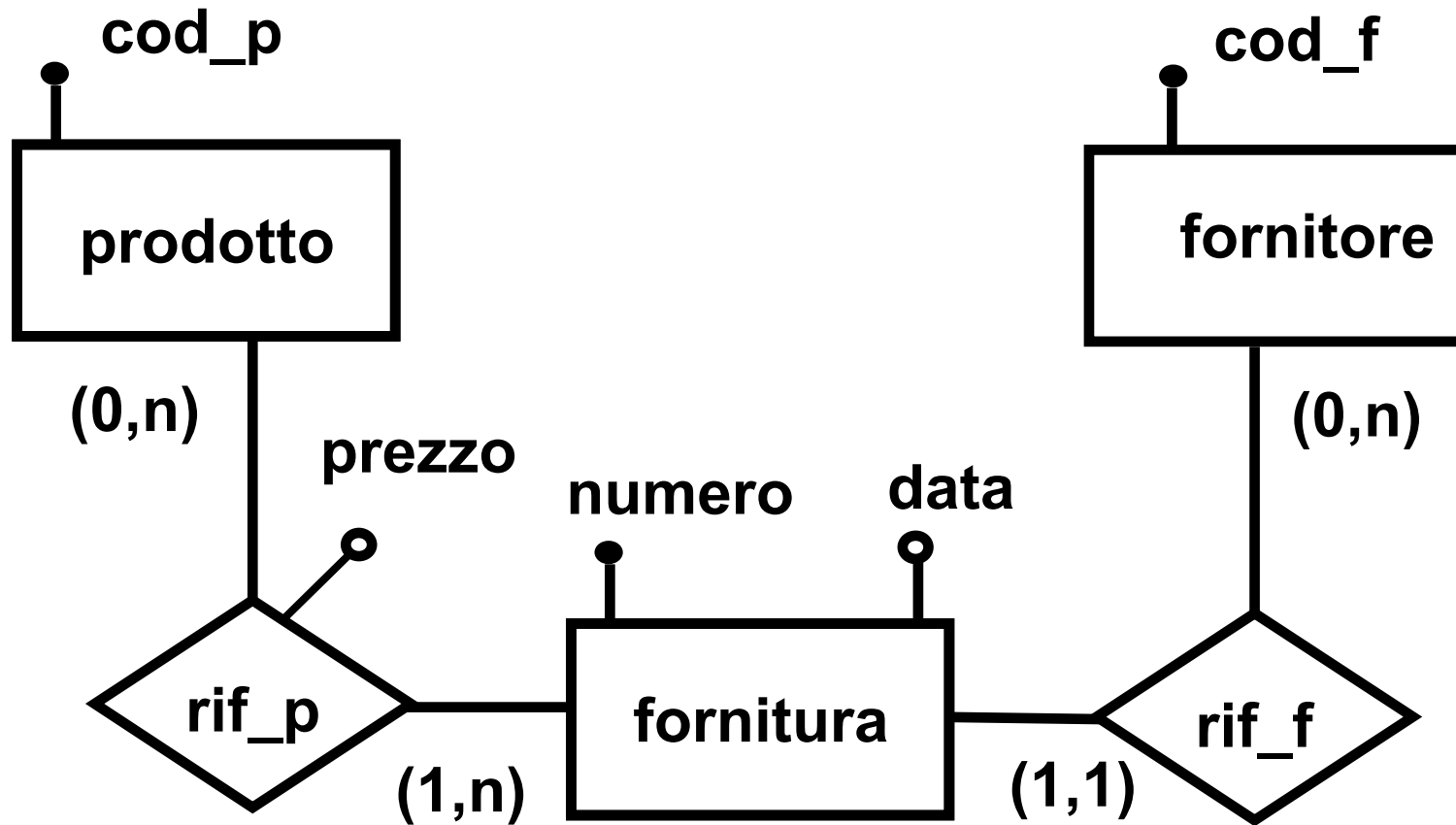


# Fornitura

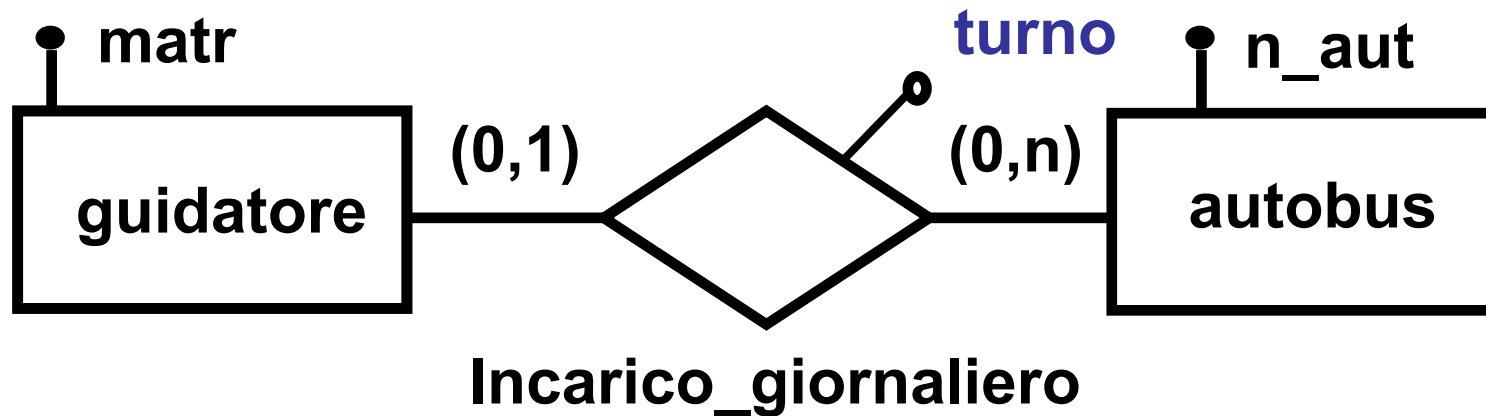


**NOTA:** questo modello non consente a un fornitore di fornire più volte lo stesso prodotto a seguito di ordini diversi perché si avrebbe una violazione dell'unicità delle chiavi

# Fornitura / 2



# Trasporti

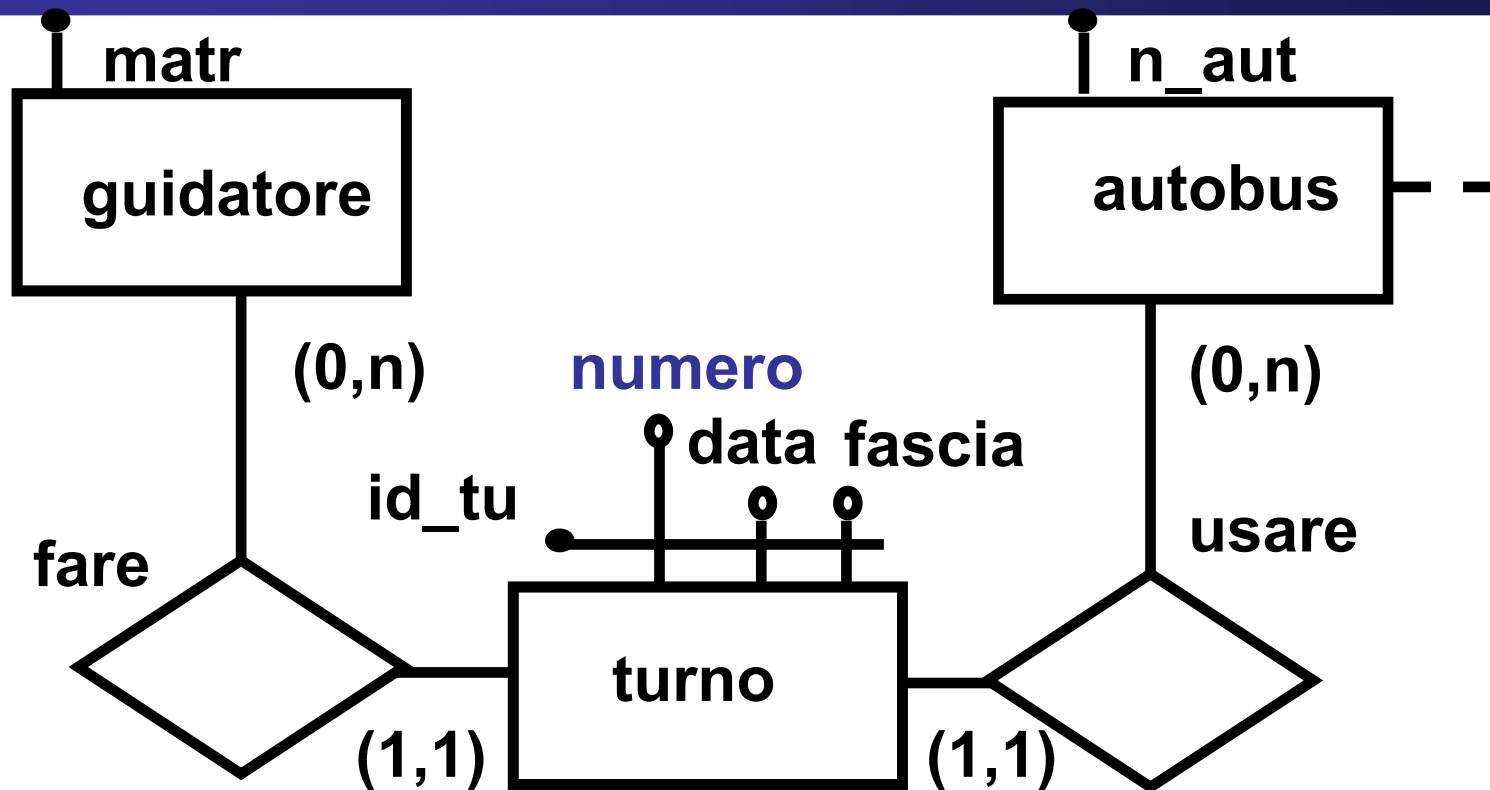


- **Attenzione all'interpretazione:**

**questo è il programma giornaliero, in cui ogni guidatore è assegnato a un solo bus e fa un solo turno**

**non rappresenta l'archivio dei turni svolti da un guidatore!**

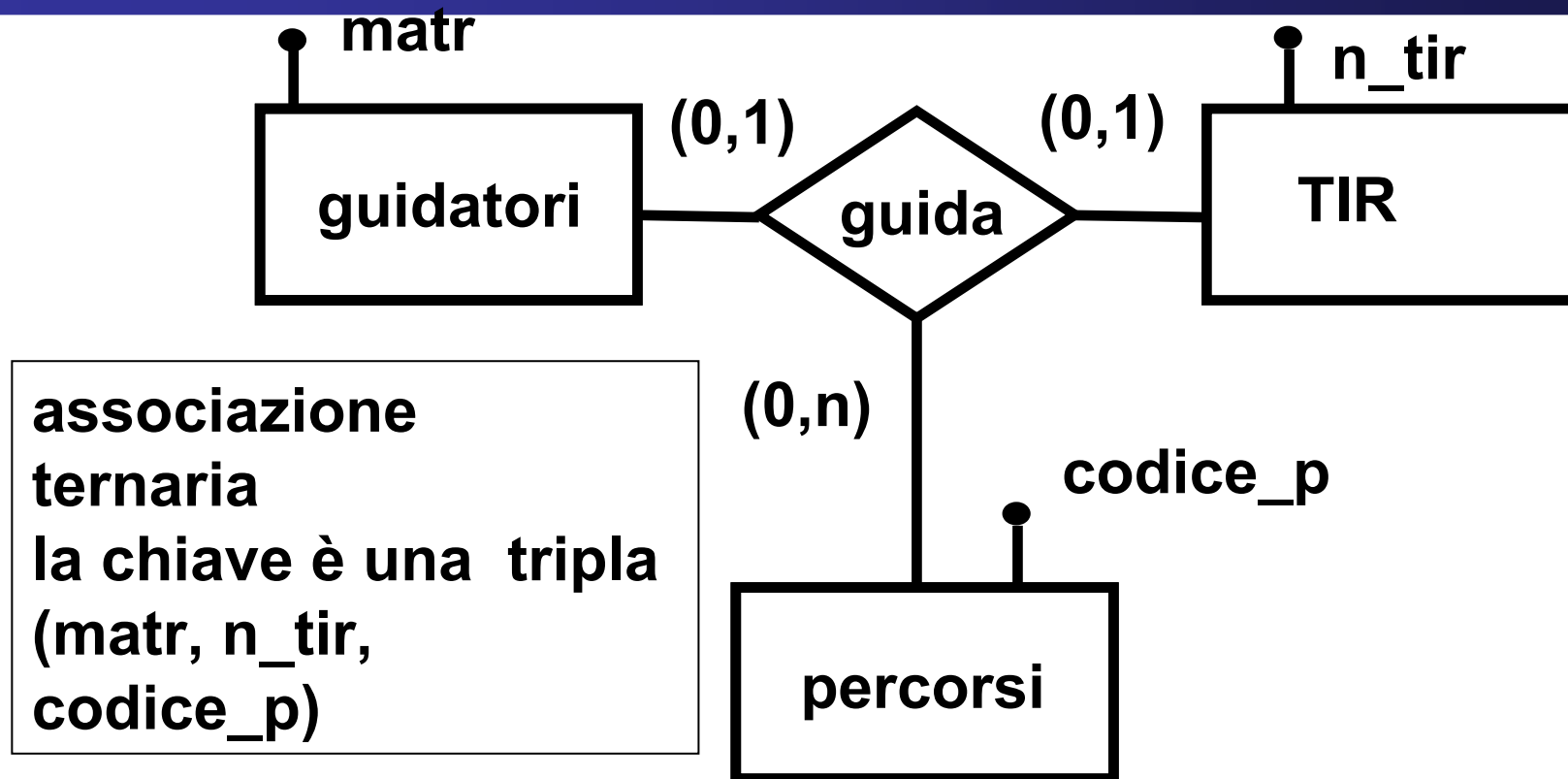
# Trasporti / 2



## Archivio assegnamenti:

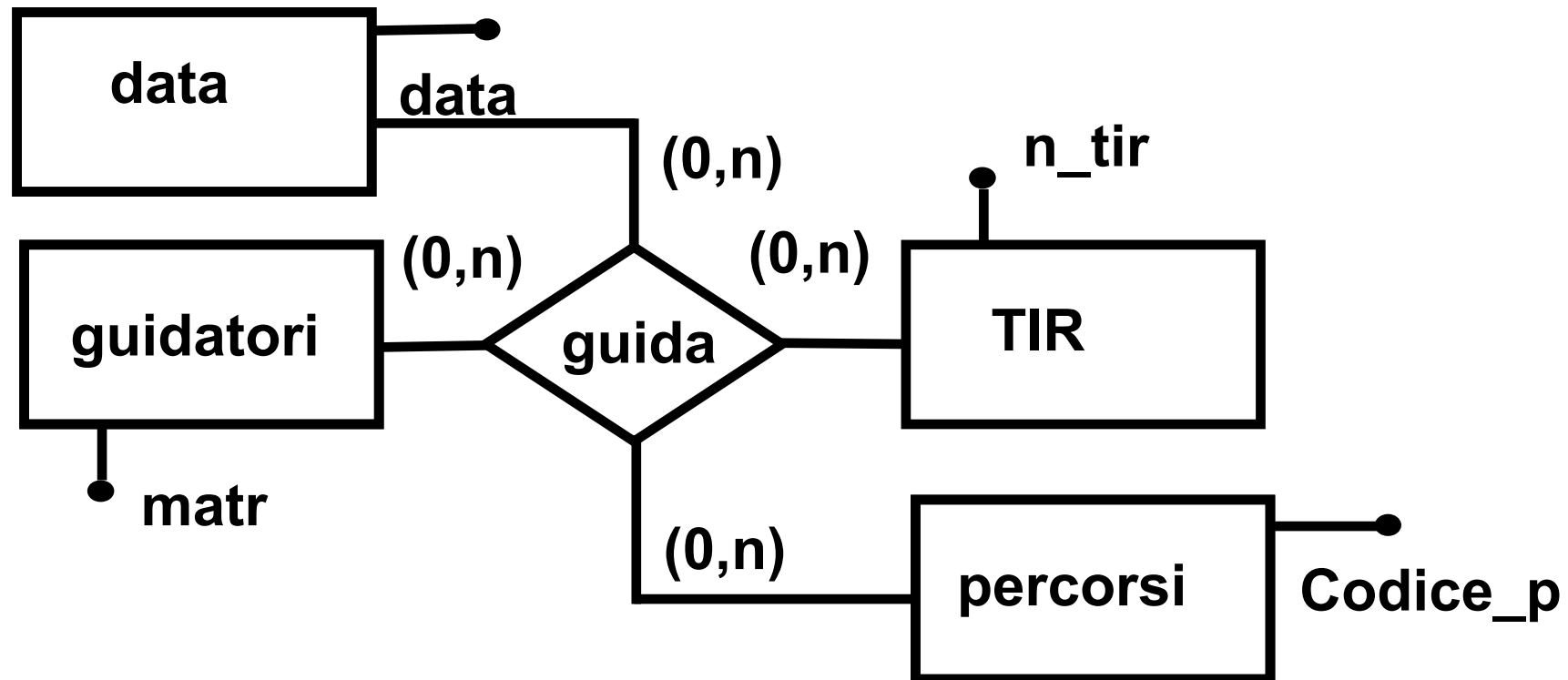
ci possono essere più turni nella stessa data e fascia oraria, svolti da guidatori diversi su bus diversi

# Trasporti bis / 1



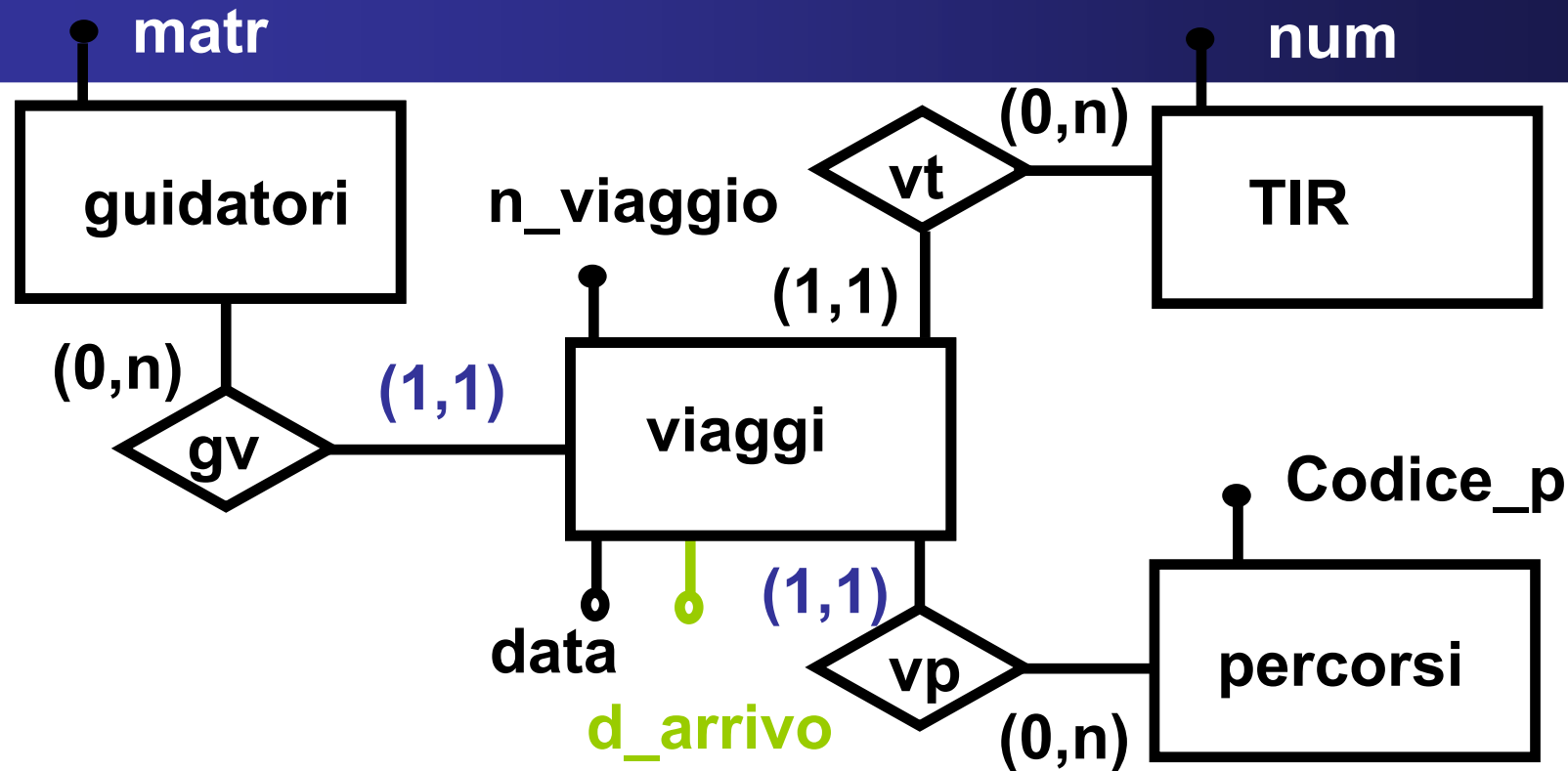
**NOTA:** descrive la situazione di un solo giorno, perché un guidatore non può condurre lo stesso TIR sullo stesso percorso più di una volta

# Trasporti bis / 2



- un guidatore può condurre lo stesso TIR sullo stesso percorso più di una volta, ma in date diverse
- questa rappresentazione nasconde la presenza di un'entità

# Trasporti bis / 3



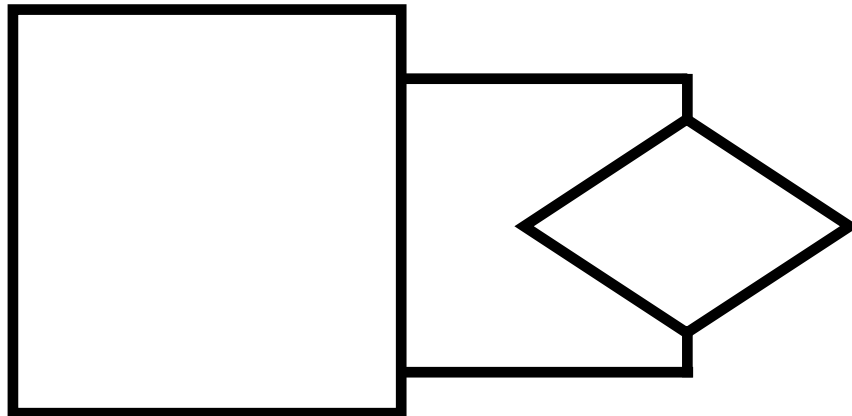
- Potremmo anche mettere  $(1,n)$  al posto di  $(1,1)$ , per esempio per rappresentare viaggi con più di un guidatore

# Commento

- Le soluzioni due e tre non consentono di definire sullo schema l'unicità della terna (`matr,n_tir,codice_p`) in ogni snapshot giornaliera, in questo caso si può:
  - aggiungere un foglio di specifica con la soluzione uno
  - usare gli "identificatori esterni" come vedremo in seguito.

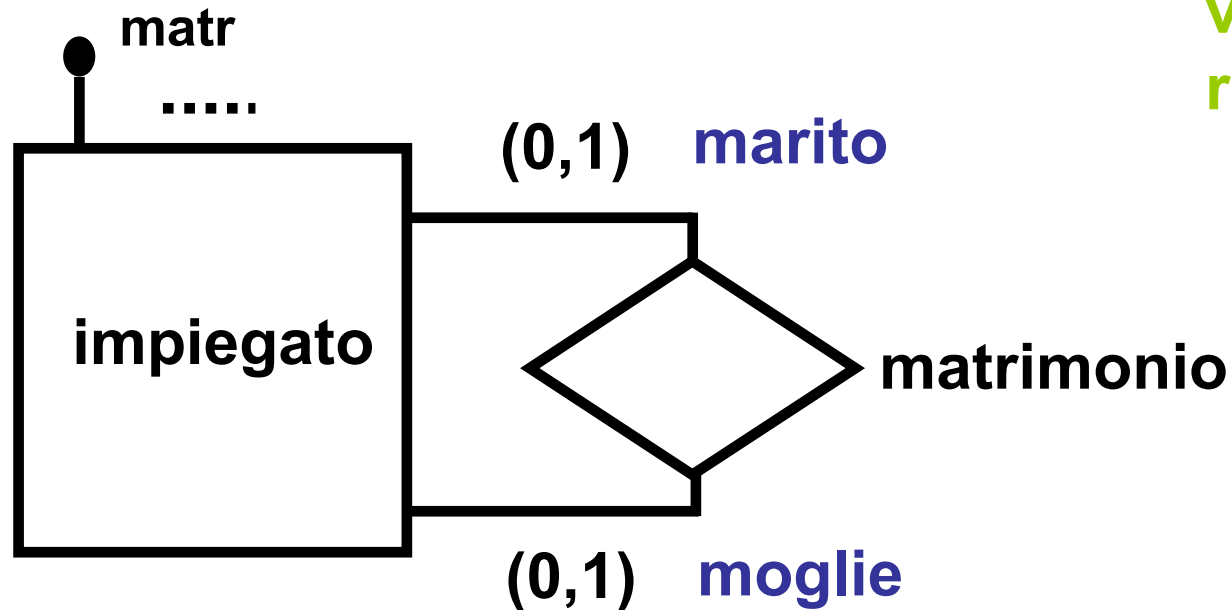
# Le auto-associazioni

- Associazioni aventi come partecipanti istanze provenienti dalla stessa entità (chiamate anche unarie o ad anello):



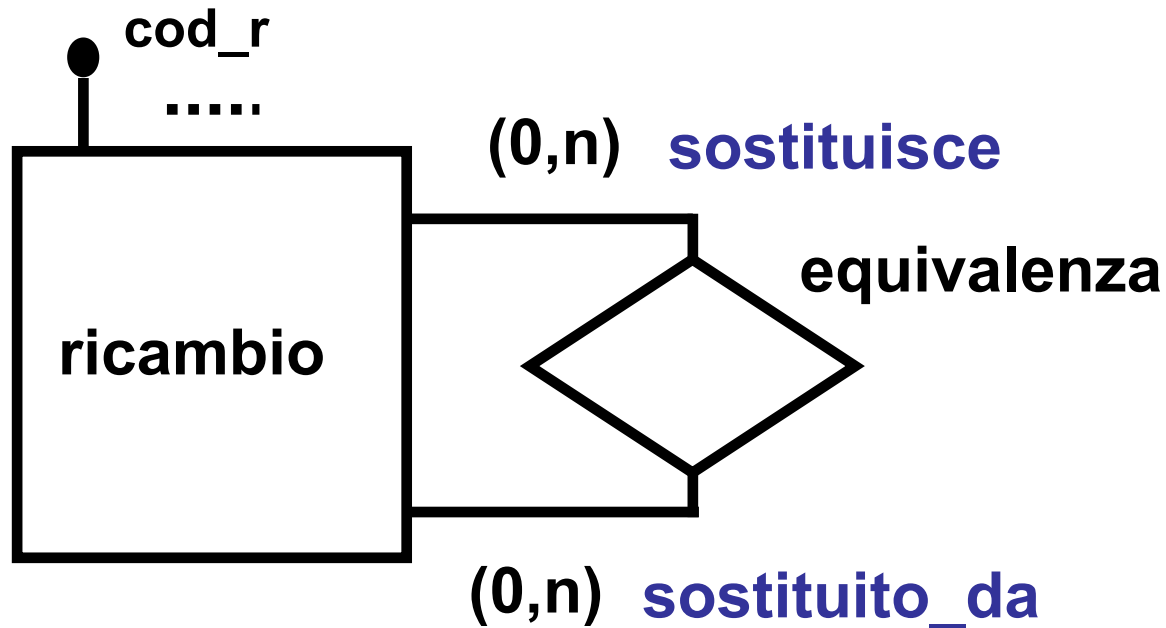
# Auto-associazioni 1:1

bnc, nre  
vrd, gll  
rss, vli



Sul ogni ramo può essere riportato il "ruolo" del partecipante all'associazione

# Auto-associazioni n:m



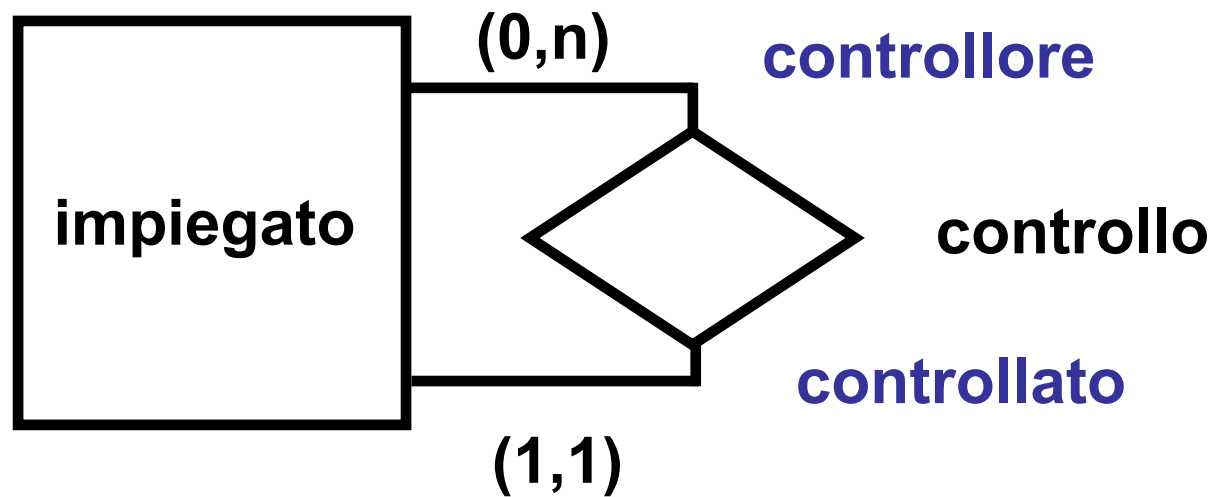
NB: può essere interpretata  
come **bi-** o **mono-**direzionale

p22	m89
k45	s56

p22	m89
k45	s56
m89	p22
s56	k45

# Auto-associazioni ricorsive

esempio: gerarchia (1:n)



grafo diretto aciclico (albero)

# LA PROGETTAZIONE DELLE BASI DI DATI

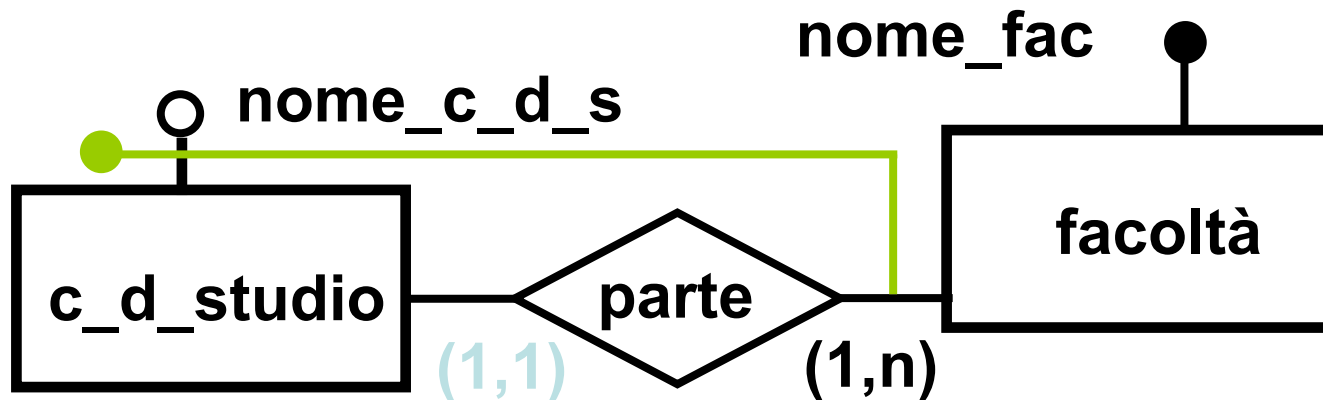
Progettazione concettuale 2<sup>^</sup> parte

—

Progettazione logica 1<sup>^</sup> parte

# Identificazione esterna

In alcuni casi una entità può essere identificata da altre ad essa collegate



Nell'esempio i corsi di studio sono identificati da un nome proprio e da quello della facoltà che li eroga, ad esempio:

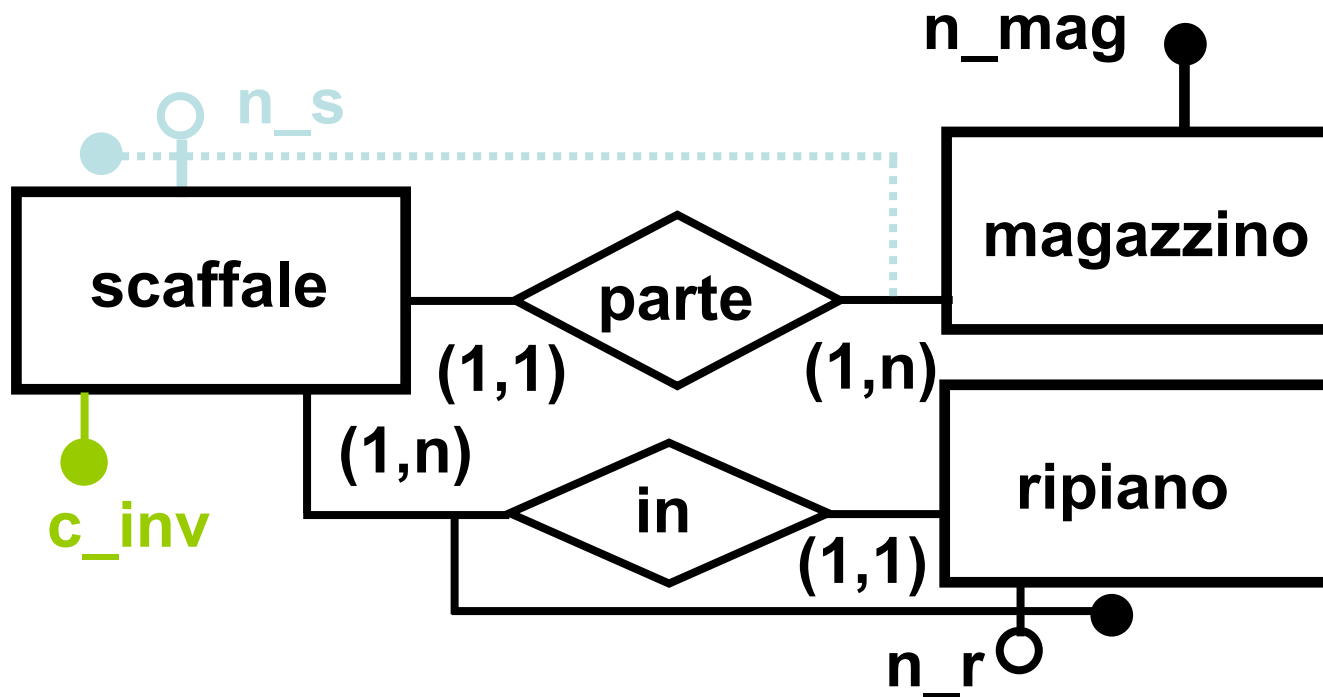
**laurea in Informatica della facoltà di Scienze MM. FF. NN. di Catania**

# Regole da rispettare

- le identificazioni esterne avvengono sempre tramite **associazioni binarie** in cui **l'entità da identificare partecipa con cardinalità (1,1)**
- una identificazione esterna può coinvolgere una entità che a sua volta è identificata esternamente a patto che non si creino cicli di identificazione
- una identificazione esterna può coinvolgere più entità purché legate da associazioni binarie in cui l'entità da identificare partecipa con cardinalità (1,1)

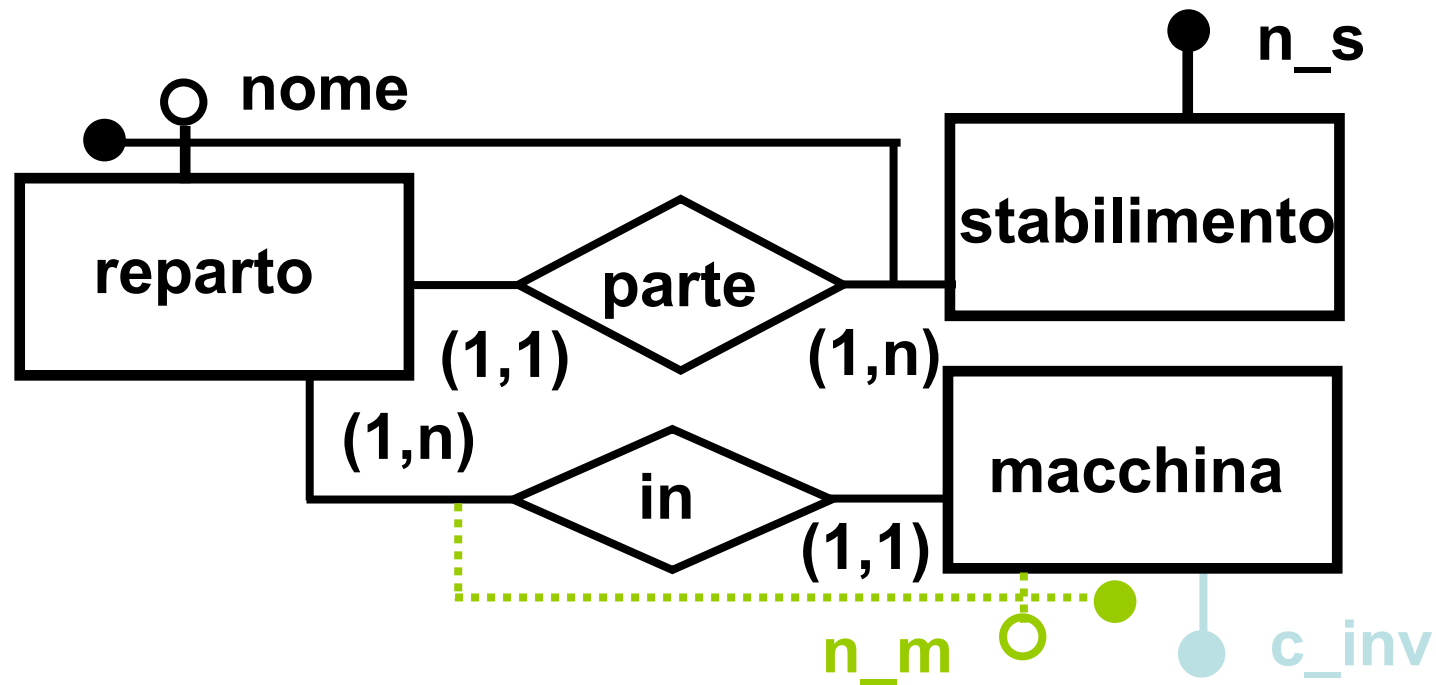
# Chiavi alternative

entità con chiavi alternative: interno ed esterna



# Scelta delle chiavi

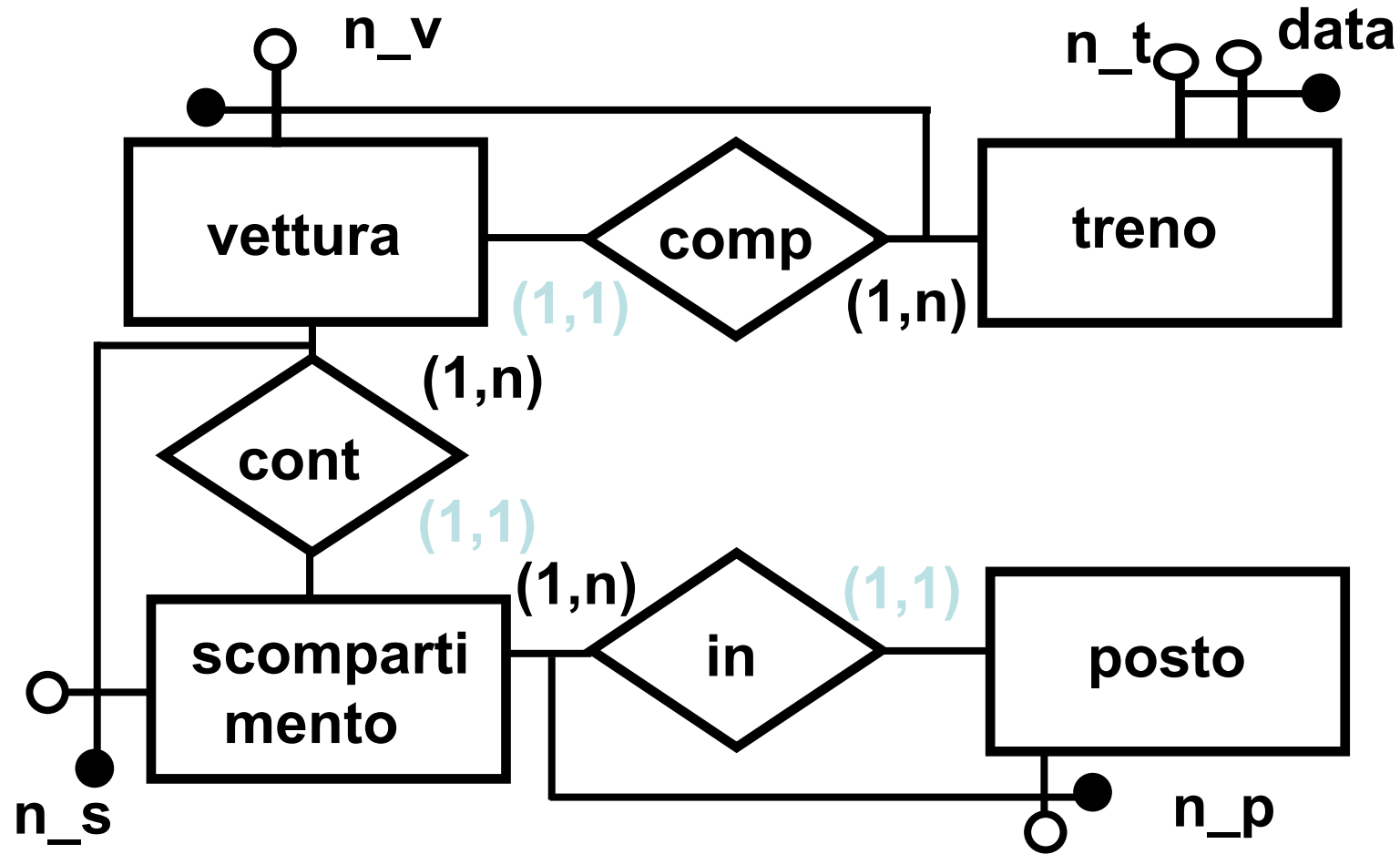
entità con chiavi alternative: interno ed esterna



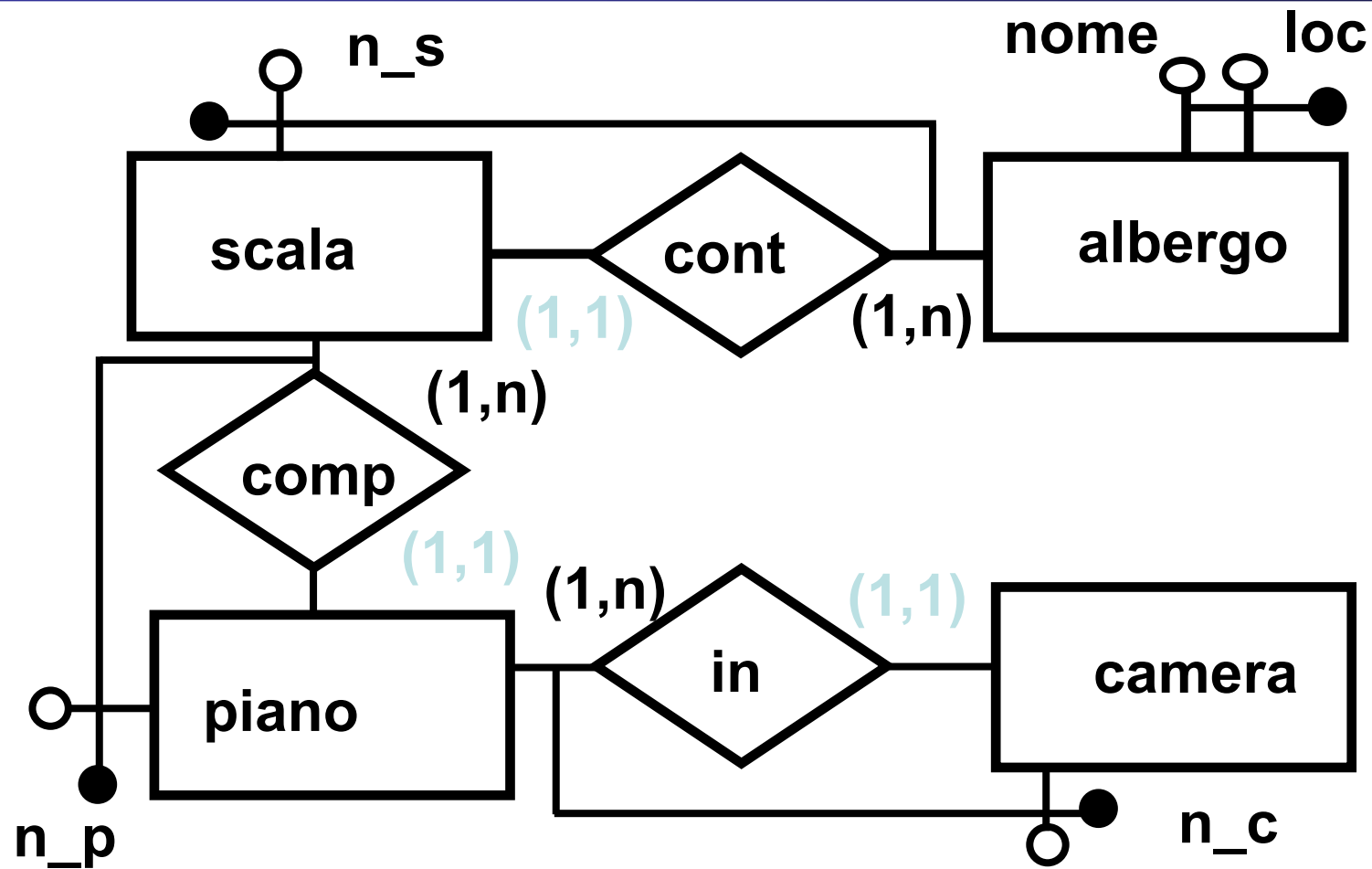
# Esempio: composizione treni

- i treni sono identificati da un codice e da una data, sono composti da vetture che contengono i posti da prenotare
- le vetture sono numerate, i posti sono numerati nello stesso modo all'interno di ogni vettura
- (potremmo tenere conto anche degli scompartimenti interni alle vetture)

# Schema composizione treni



# Esempio: camere d'albergo



# Requisiti di modellazione

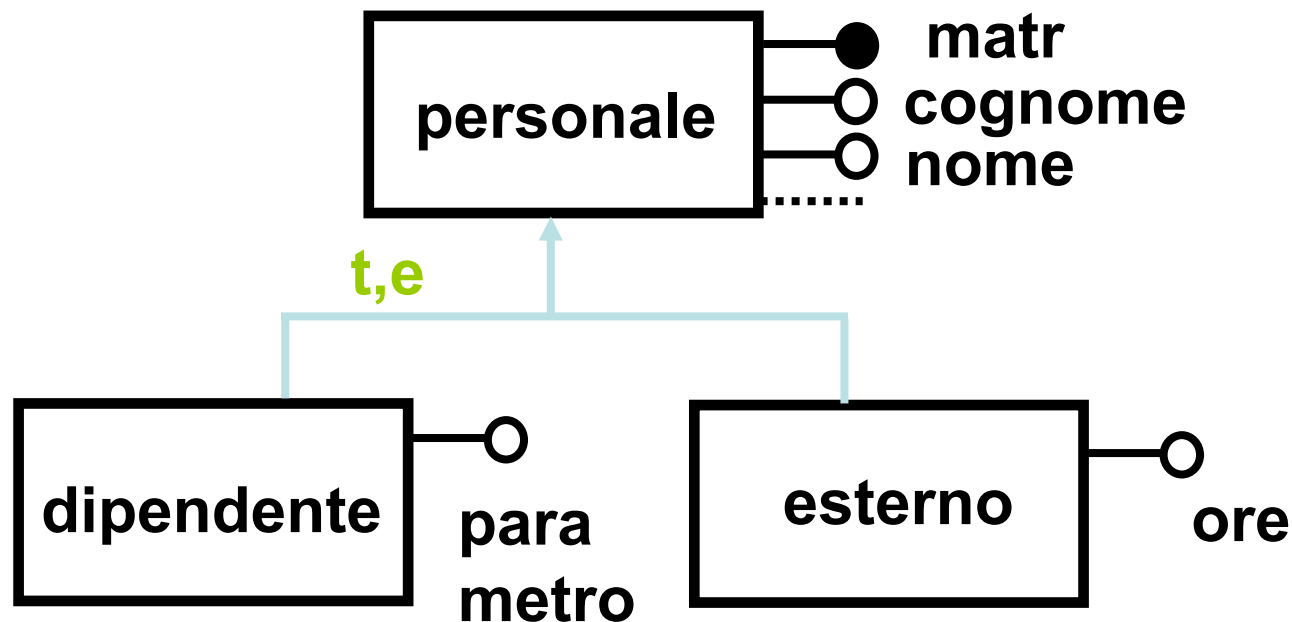
- spesso nella analisi di un settore aziendale può risultare che più entità risultino simili o casi particolari l'una dell'altra, derivanti da "viste" diverse da parte dell'utenza
- emerge quindi la necessità di evidenziare sottoclassi di alcune classi
- si definisce pertanto **gerarchia di specializzazione** il legame logico che esiste tra classi e sottoclassi

# Le gerarchie

- Definizione: la **gerarchia concettuale** è il legame logico tra un'entità padre  $E$  ed alcune entità figlie  $E_1 E_2 \dots E_n$  dove:
  - $E$  è la **generalizzazione** di  $E_1 E_2 \dots E_n$
  - $E_1 E_2 \dots E_n$  sono **specializzazioni** di  $E$
  - una istanza di  $E_k$  è anche istanza di  $E$  (e di tutte la sue generalizzazioni)
  - una istanza di  $E$  **può** essere una istanza di  $E_k$
  - **NOTA:** nel caso in cui  $k=1$  allora  $E_1$  è un sottoinsieme di  $E$

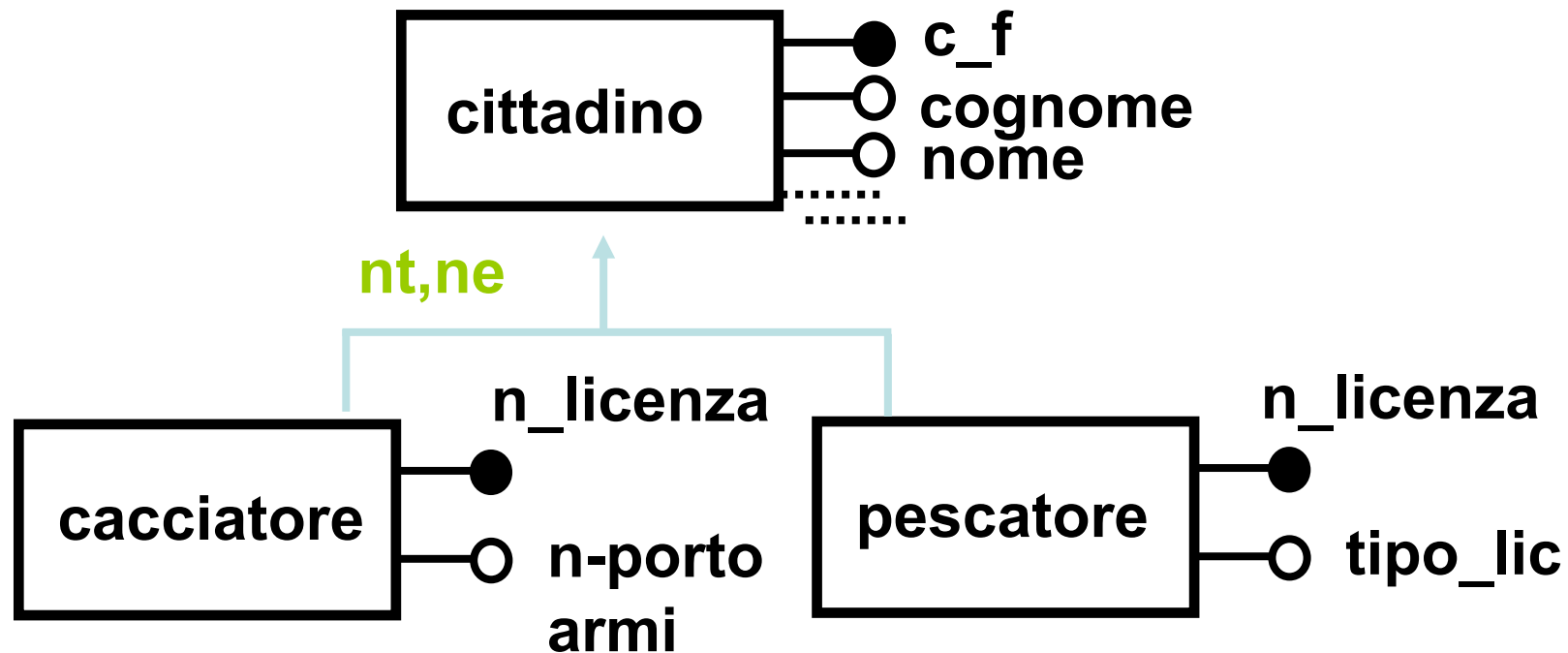
# Classificazione del personale

un'azienda si avvale dell'opera di professionisti esterni, quindi il suo personale si suddivide in esterni e dipendenti:



# Anagrafe comunale

un comune gestisce l'anagrafe ed i servizi per i suoi cittadini alcuni di questi richiedono I dati relativi alla licenza di pesca e/o di caccia:



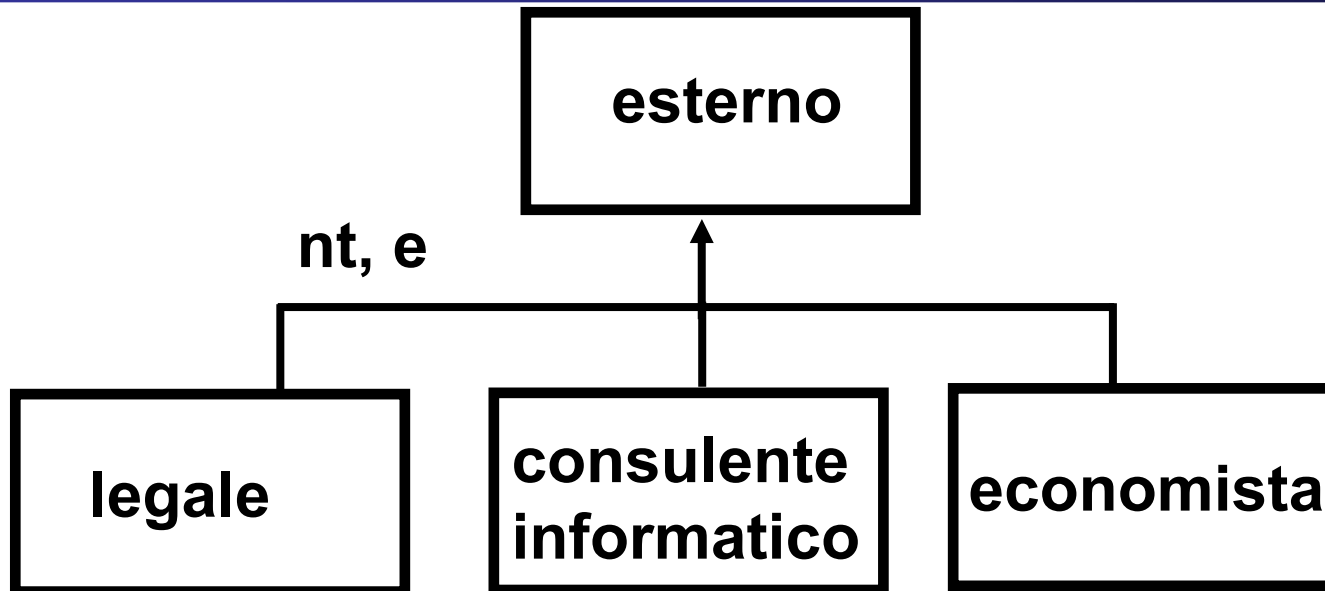
# Tipi di gerarchie: totalità

- **t** sta per **totale**: ogni istanza dell'entità padre deve far parte di una delle entità figlie
  - nell'esempio il personale si divide (completamente) in esterni e dipendenti
- **nt** sta per **non totale**: le istanze dell'entità padre possono far parte di una delle entità figlie
  - nell'esempio i pescatori sono un sottoinsieme dei cittadini

# Tipi di gerarchie: esclusività

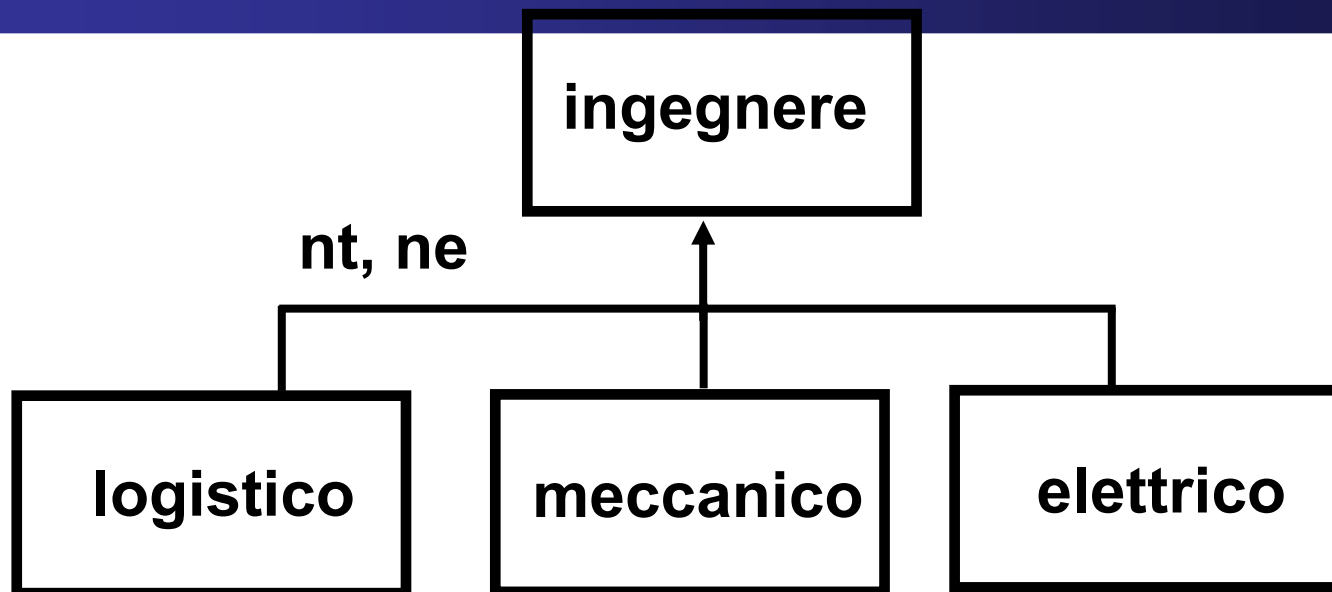
- **e sta per esclusiva:** ogni istanza dell'entità padre deve far parte di una sola delle entità figlie
  - esempio: una istanza di personale non può sia essere sia dipendente che esterno
- **ne sta per non esclusiva:** ogni istanza dell'entità padre può far parte di una o più entità figlie
  - esempio: un cittadino può essere sia pescatore che cacciatore

# Mansioni esterne



***nt*** : **possono** esistere esterni generici che non sono né legali, né ingegneri, né economisti ma non interessa stabilire una **sottoclasse** ad hoc

# Tipi di ingegnere



***ne*** : **possono** esistere ingegneri con competenze meccaniche, elettriche, e logistiche  
le tre qualifiche **non si escludono**

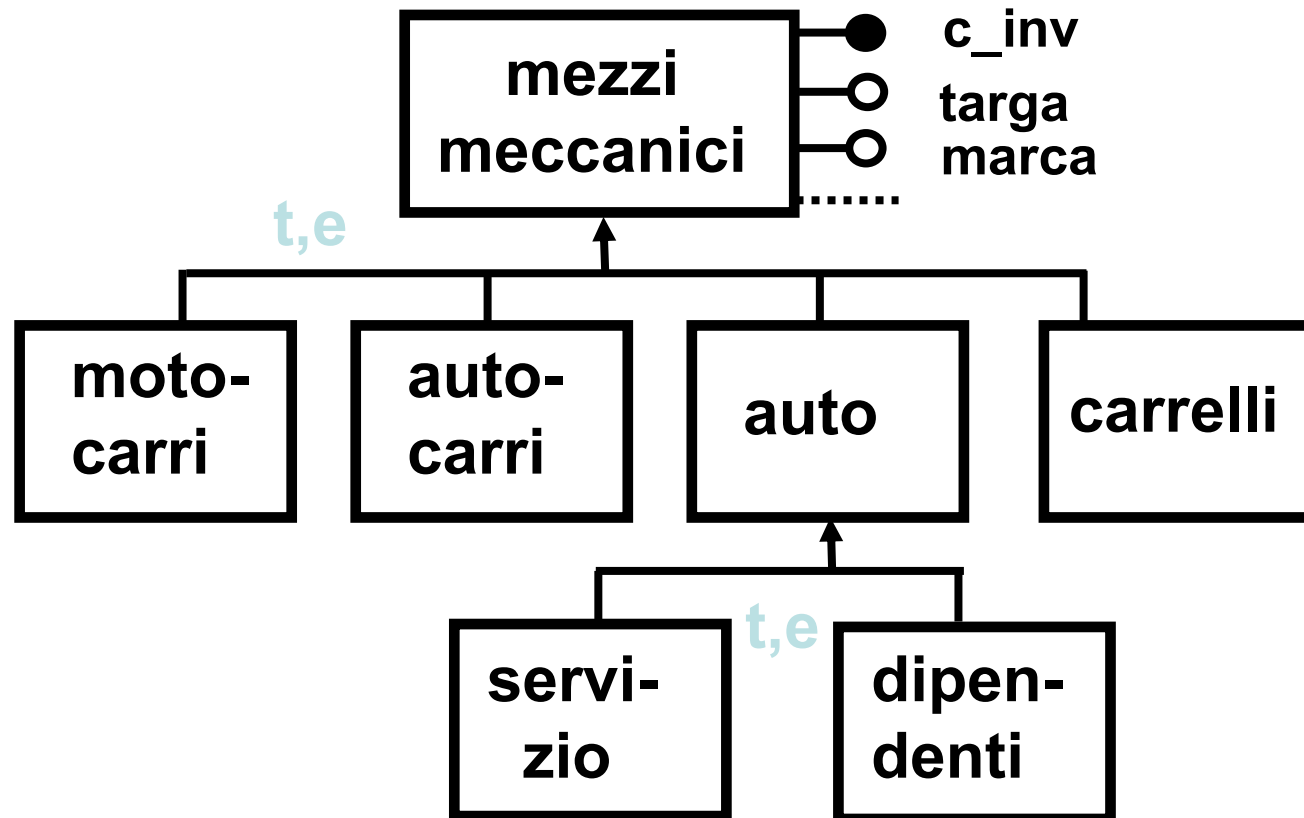
# Ereditarietà delle proprietà

- le *proprietà* dell'entità padre non devono essere replicate sull'entità figlia in quanto questa le *eredita* cioè:
- le proprietà dell'entità padre fanno parte del *tipo* dell'entità figlia
- non è vero il viceversa
  - il tipo di **personale** è: (**matricola, cognome, nome, indirizzo, data\_nascita**)

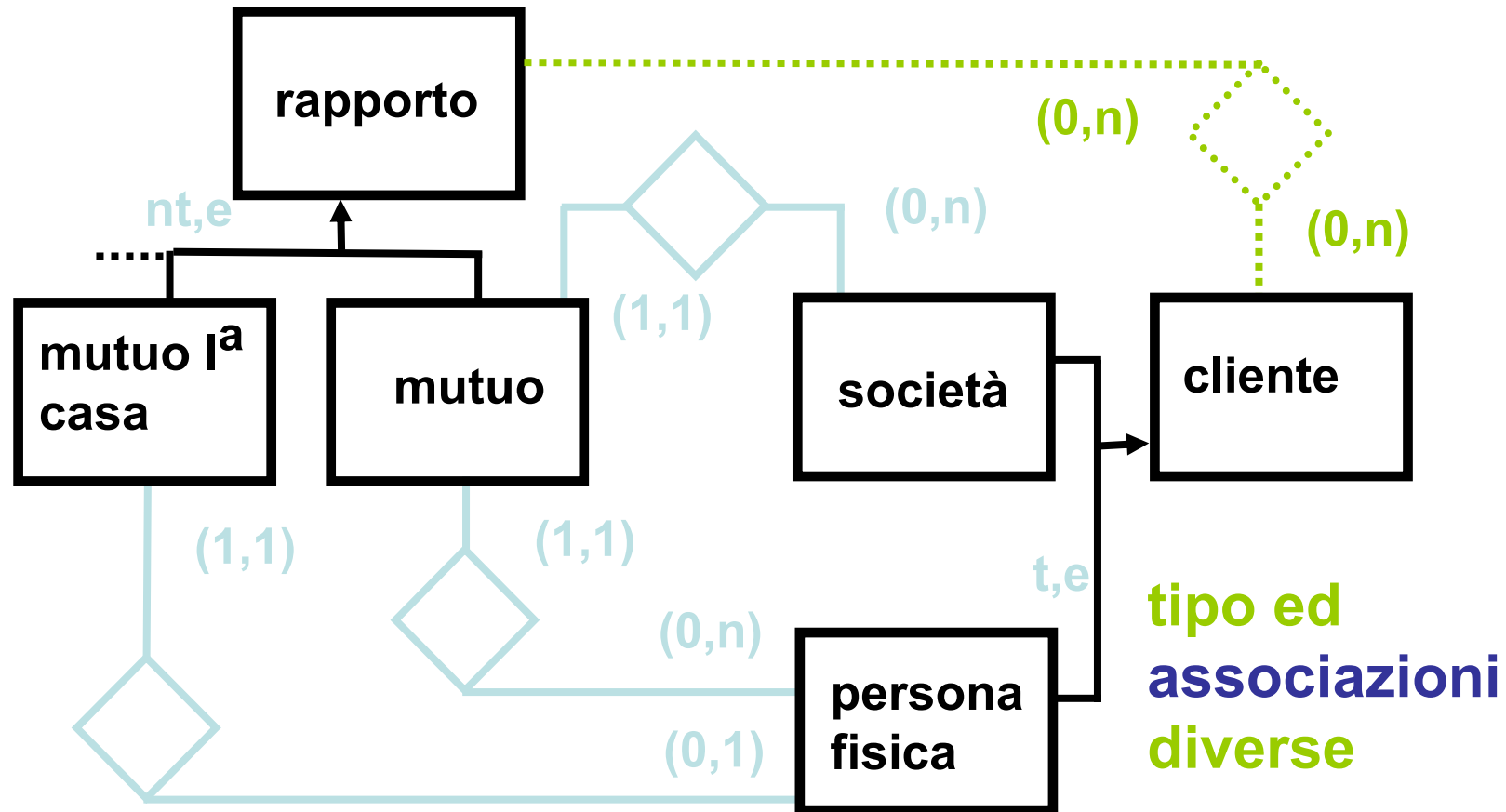
# Ereditarietà

- il tipo di **dipendente** è: (matricola, cognome, nome, indirizzo, data\_nascita, **parametro**)
- il tipo di **esterno** è: (matricola, cognome, nome, indirizzo, data\_nascita, **ore**)
- dipendente ed esterno hanno lo stesso tipo se considerati come personale
- **NB:** le gerarchie concettuali sono anche denominate **gerarchie ISA**
  - dipendente è un (**is a**) personale
  - esterno è un (**is a**) personale

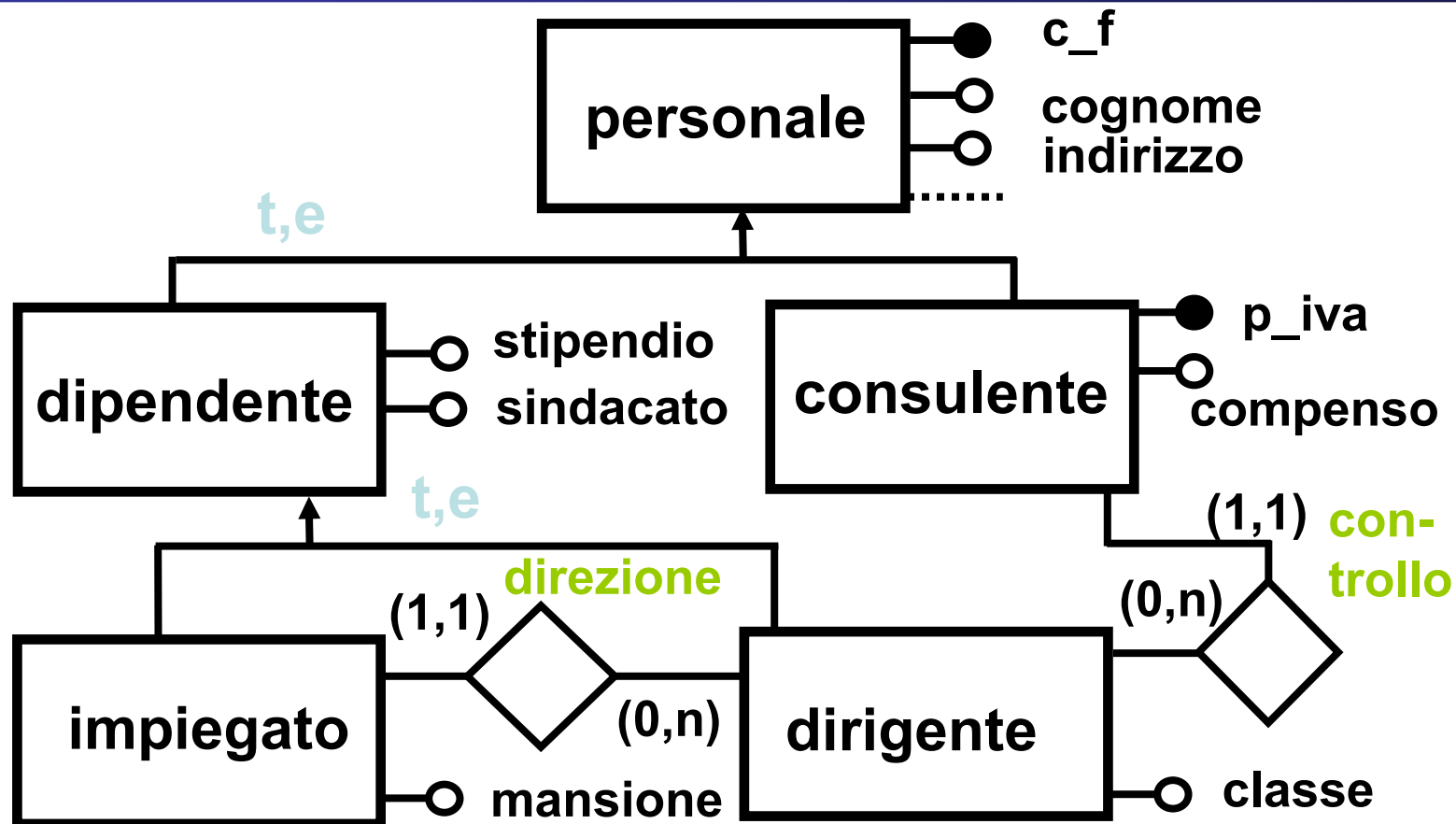
# Parco mezzi meccanici



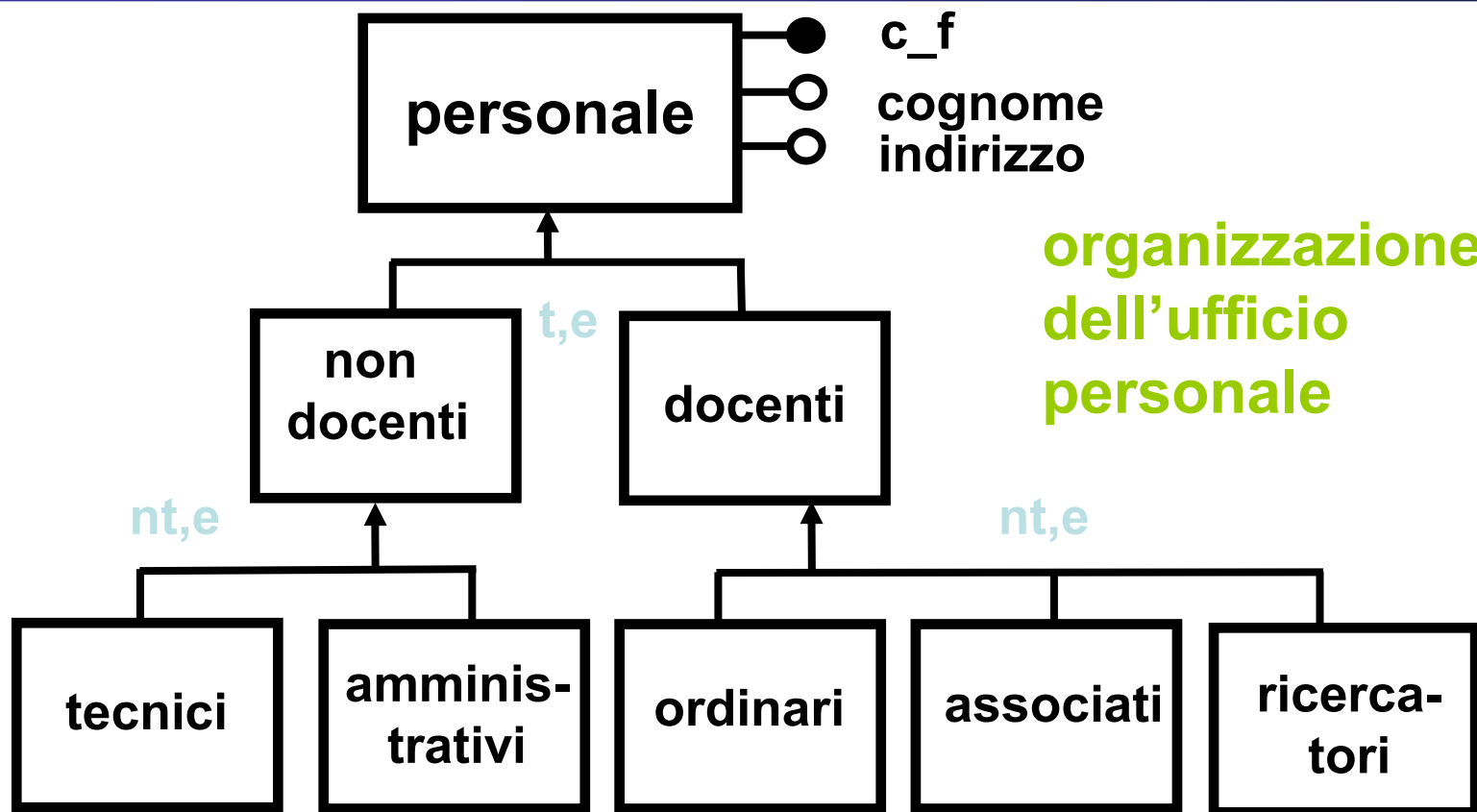
# Anagrafe bancaria



# Anagrafe aziendale



# Università



# Incertezze e ridondanze

- Dalle frasi di specifica possono emergere due situazioni
- carenze di specifica:
  - schemi incongruenti
- eccesso di specifica:
  - schemi ridondanti (e contraddittori)

# Incertezze negli schemi

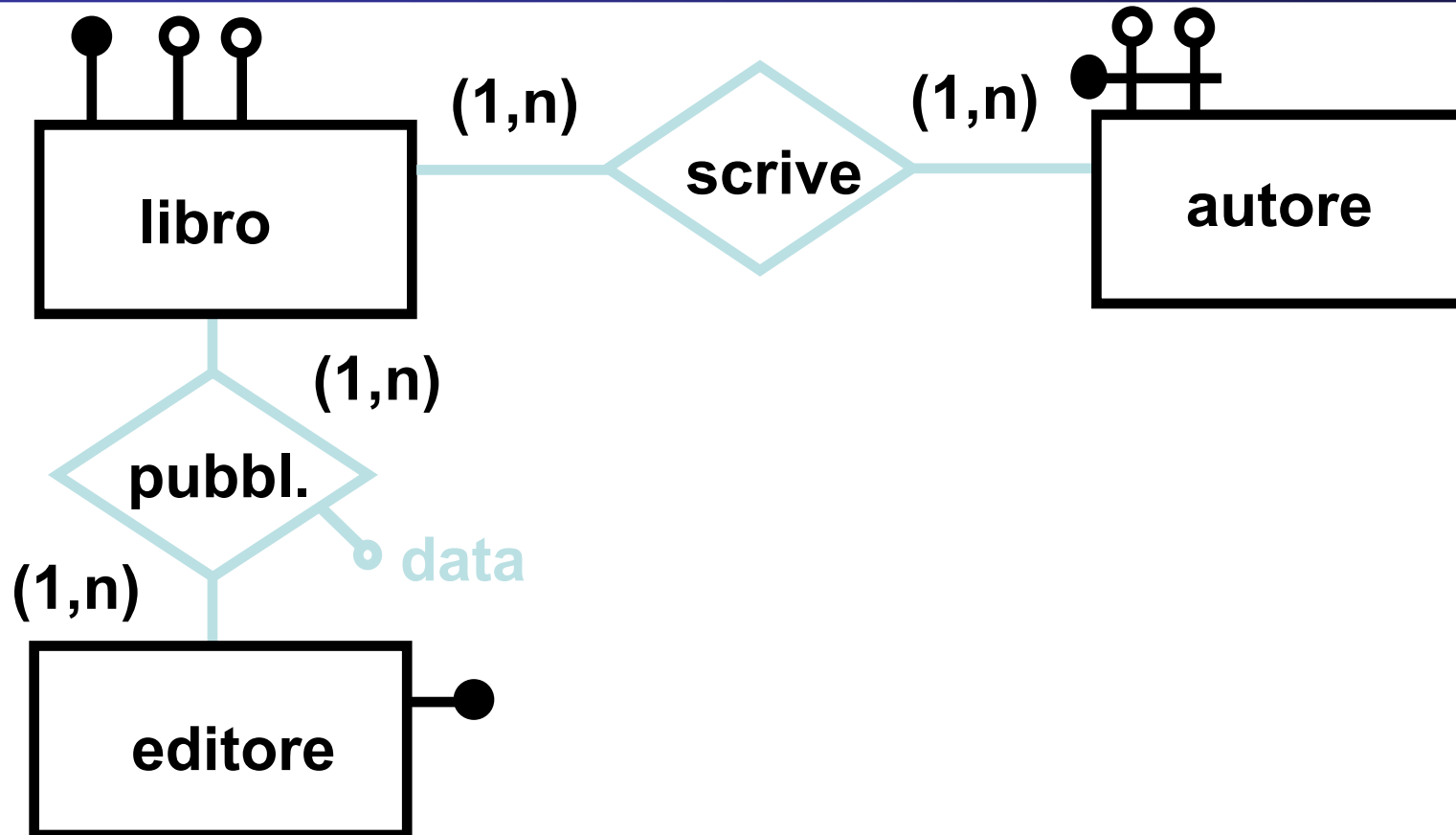
- vediamo adesso di costruire uno schema partendo dalle frasi di specifica
- le frasi conterranno delle incertezze che dovranno via, via essere chiarite apportando di volta, in volta le opportune modifiche allo schema
- l'esempio riguarda la gestione di una biblioteca

# Biblioteca: specifiche

- Le frasi:
  - i libri sono identificati da un codice, hanno un titolo ed un numero di pagine
  - dei libri interessa conoscere gli autori
  - dei libri interessa conoscere l'editore
  - gli autori sono identificati dal nome e dal cognome
  - gli editori sono identificati da una denominazione
  - gli autori possono aver scritto più libri
  - un libro può aver avuto più edizioni in date diverse e/o da parte di editori diversi
  - autori ed editori sono da considerarsi entità perché di essi interessano anche altre caratteristiche che qui non sono riportate per semplicità

# Biblioteca: schema 1

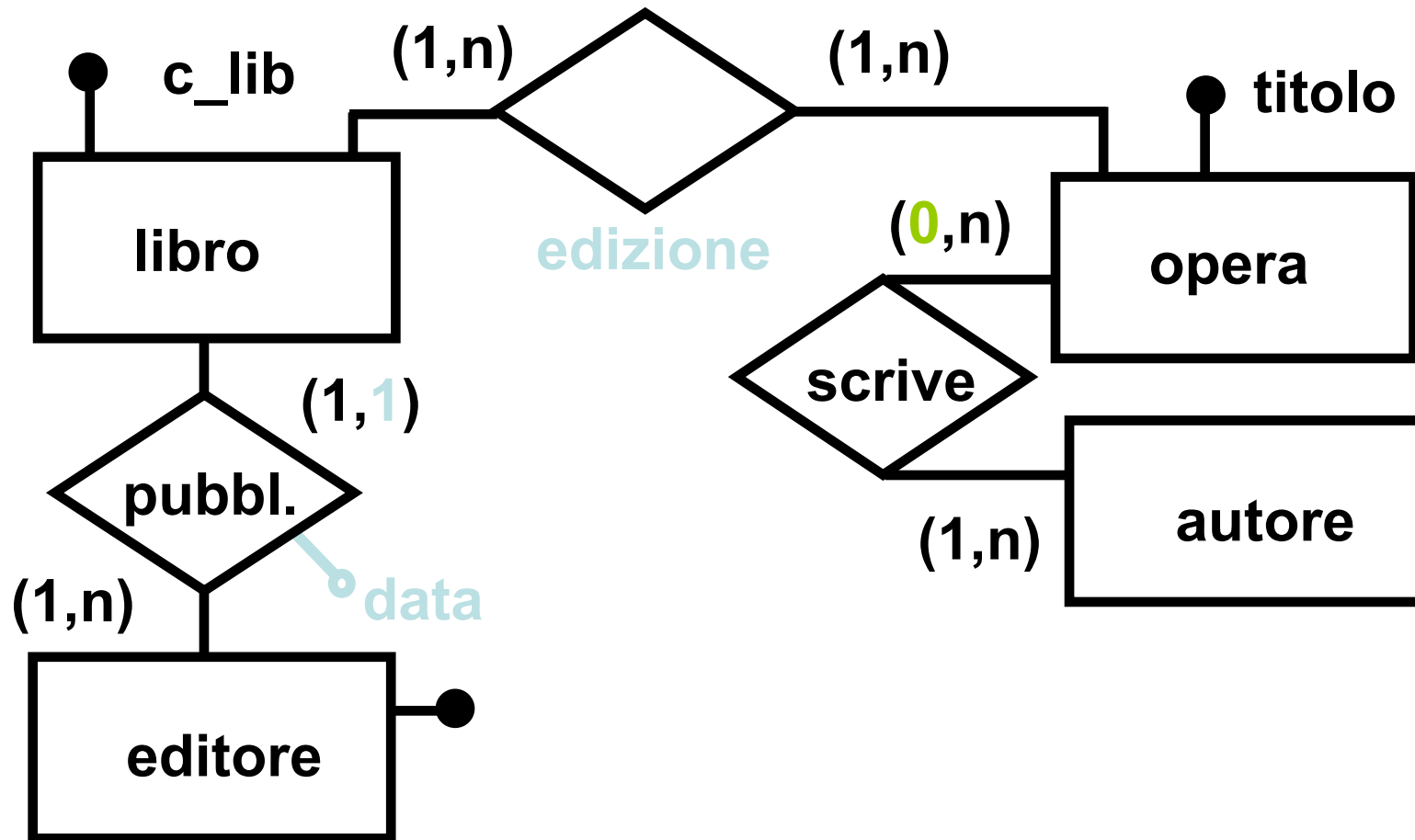
c\_lib



# Errori

- il termine libro è ambiguo: edizione o opera?
  - un editore non può fare la seconda edizione di un'opera
  - non si vede il rapporto tra autore e edizioni di un'opera

# Opera ed edizione



# Documentazione di schemi E-R

- E' sempre bene corredare un diagramma E-R con una documentazione spesso non esprimibile nel diagramma. Due sono gli elementi di questa documentazione:
  - Un **dizionario** delle entità e relazioni con relativi attributi.
  - Un elenco di **vincoli d'integrità** dei dati non esprimibili nel diagramma.

# Vincoli di integrità sulle proprietà

- Al loro ingresso nel database i valori devono essere controllati sulla base di vincoli definiti in sede di analisi;
- Non sempre i valori delle proprietà possono evolvere liberamente ma sono vincolati da regole;

# Vincoli statici

- Verifiche all'interno di intervalli:  
18<età<65, 500<peso<2000
- Presenza di valori in elenchi:  
colore in (rosso, verde, bianco, nero...),

# Vincoli statici multiproprietà

- Il vincolo su una proprietà può essere dipendente da valori di altre proprietà:
  - - se il modello è "et2" i colori disponibili sono (avorio,blu, grigio,..)
  - - se livello è 7 allora stipendio è tra 1.5 e 2.5 mil.
  - - se scaffale è di tipo "a" allora carico <100 kg
  - - se gara è "slalom speciale", sesso "M" e categoria "internazionale" il dislivello è tra 180 e 220 m, il numero di porte è libero
  - - se gara è "slalom" e categoria "cuccioli" il dislivello è < 100 m e il numero di porte è <30

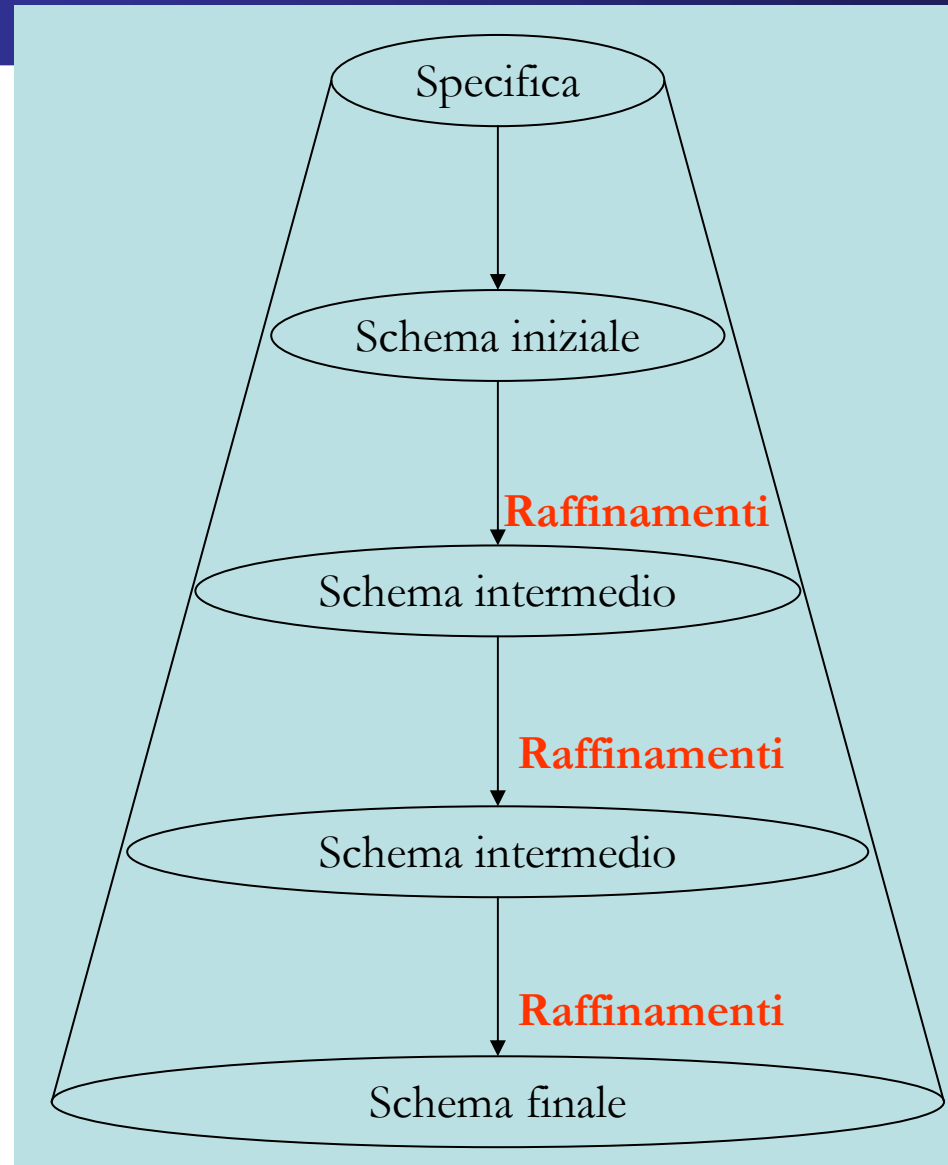
# Vincoli dinamici

- Il controllo statico può non essere sufficiente, un nuovo valore può essere valido staticamente ma può violare la regola della sua evoluzione:
  - stipendio precedente < stipendio successivo
  - età precedente < età successiva
- Sono necessarie le regole sugli eventi che fanno cambiare stato

# Strategie di progetto

- Top-Down
- Bottom-Up
- Inside-Out
- Mista

# Strategia Top-Down



# Primitive di Trasformazione TOP-DOWN

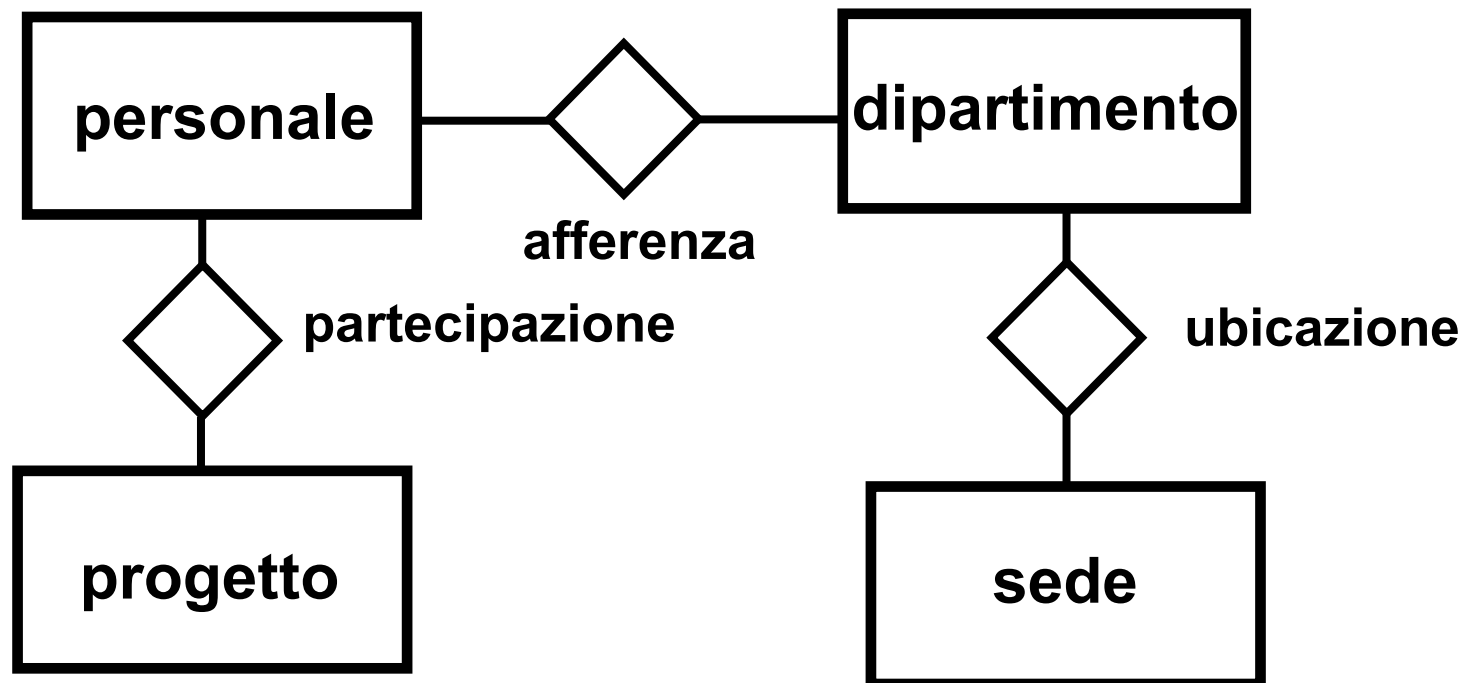
- Le primitive di trasformazione top-down sono regole che operano su un singolo concetto dello schema e lo trasformano in una struttura più complessa che descrive il concetto con maggiore dettaglio

# Regole

- **Trasformazione**
  - **T1:** si applica quando un'entità descrive *due concetti diversi* legati fra di loro.
  - **T2:** Un'entità è composta da **sotto-entità distinte**.
  - **T3:** Una relazione in realtà descrive due relazioni diverse tra le stesse entità.
  - **T4:** Una **relazione** descrive un concetto con **esistenza autonoma**. In questo caso essa va sostituita con un' entità.
  - **T5:** Si applica per aggiungere **attributi ad entità**.
  - **T6:** Si applica per aggiungere **attributi a relazioni**.

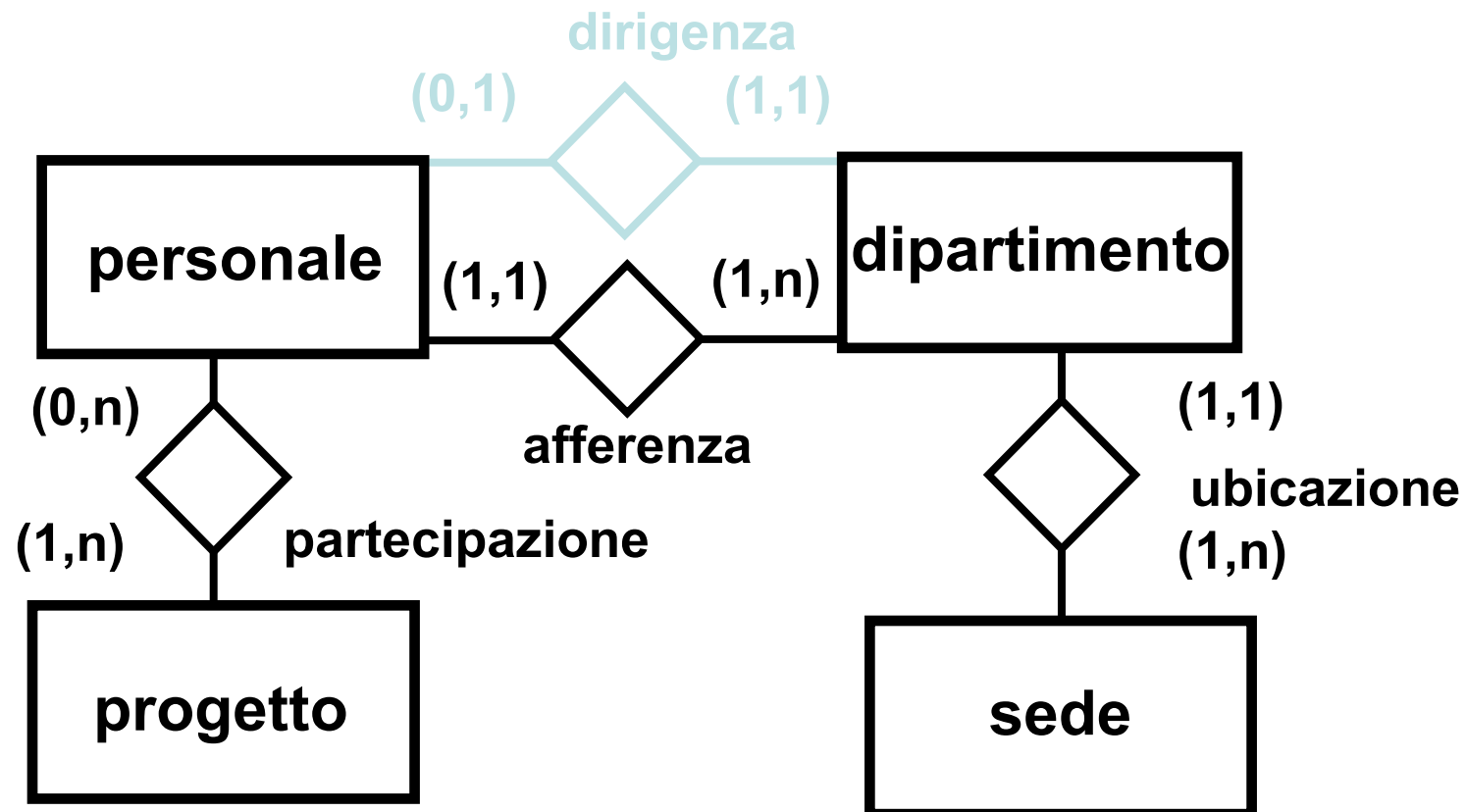
# Esempio

schema iniziale

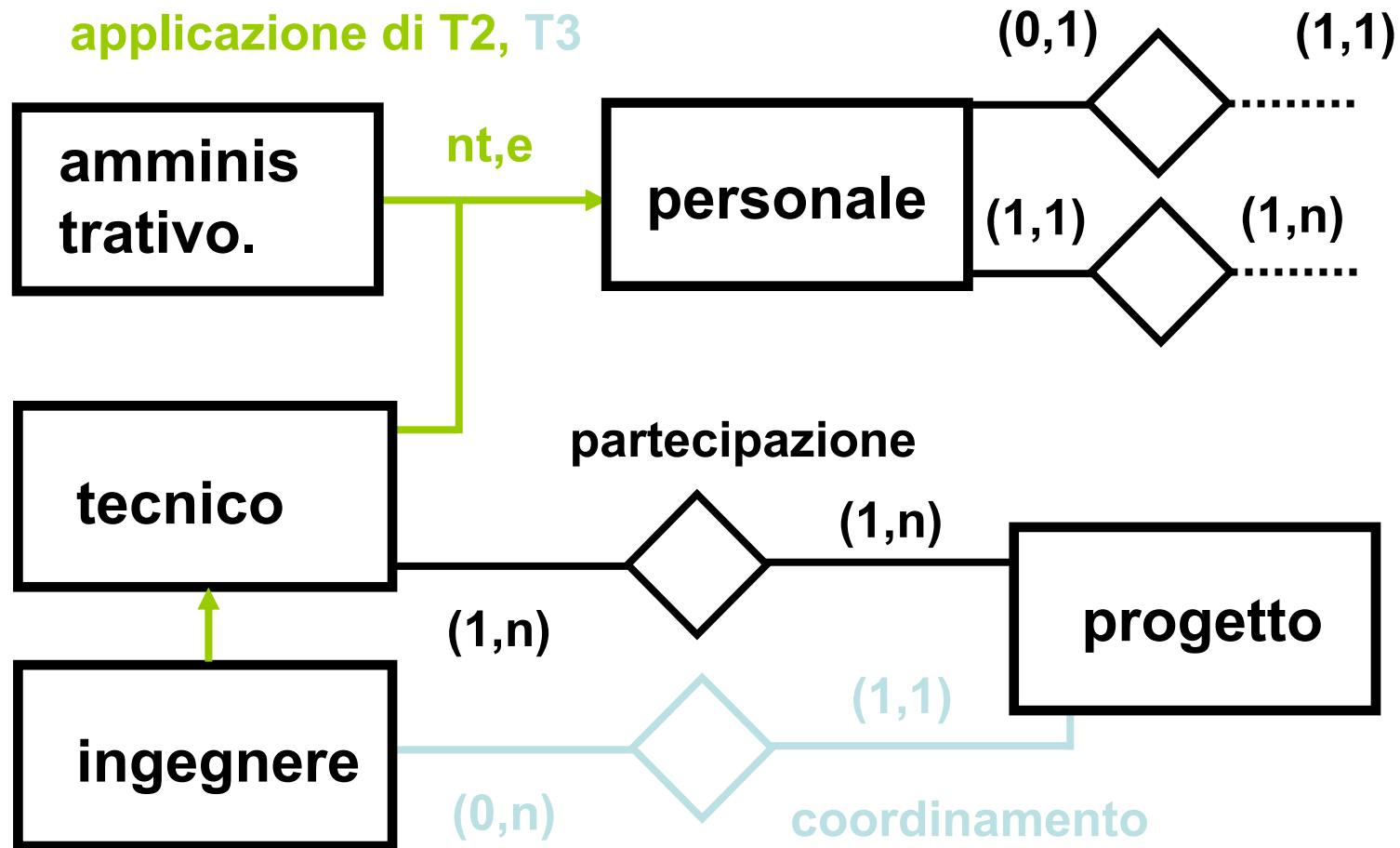


# passo 2

applicazione di T3

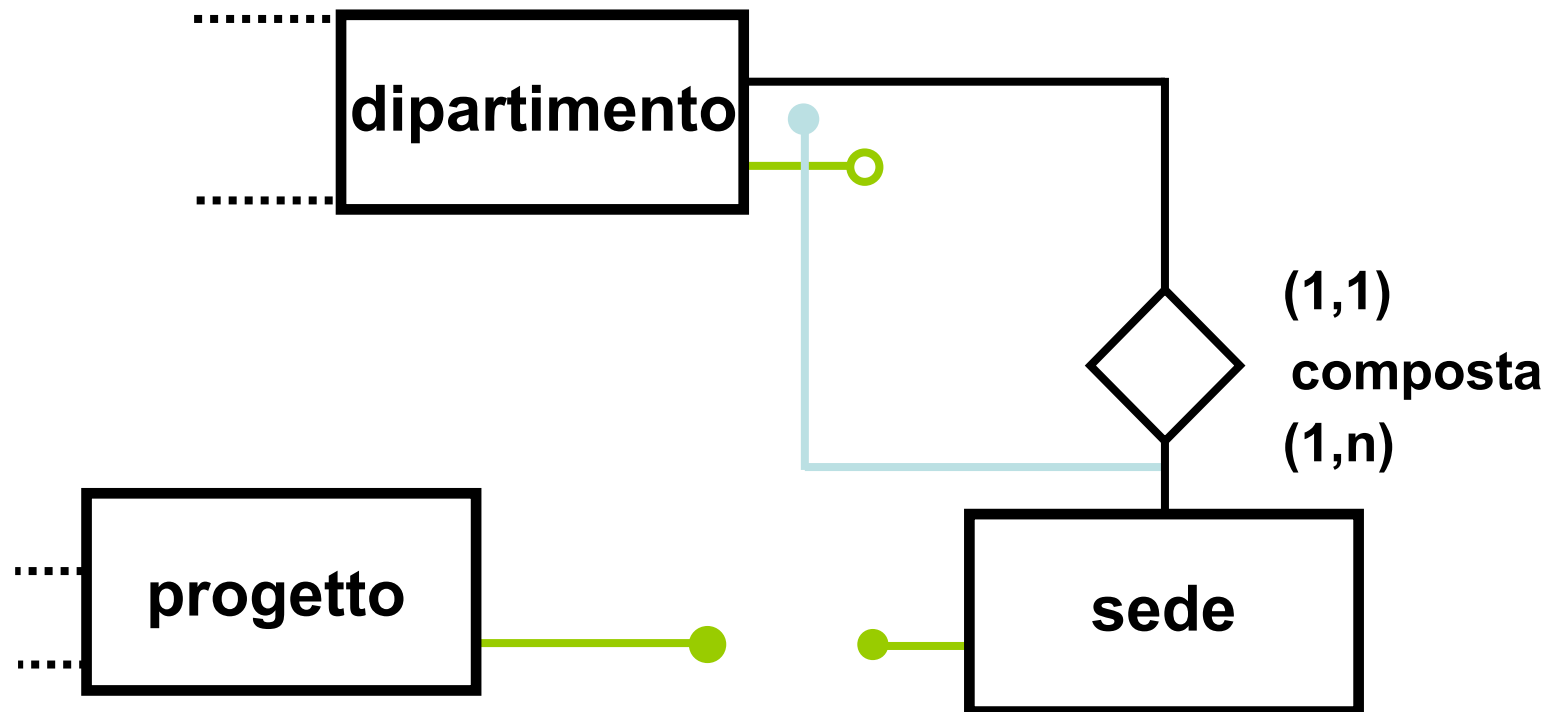


# passo 3



# passo 4

## applicazione di T5



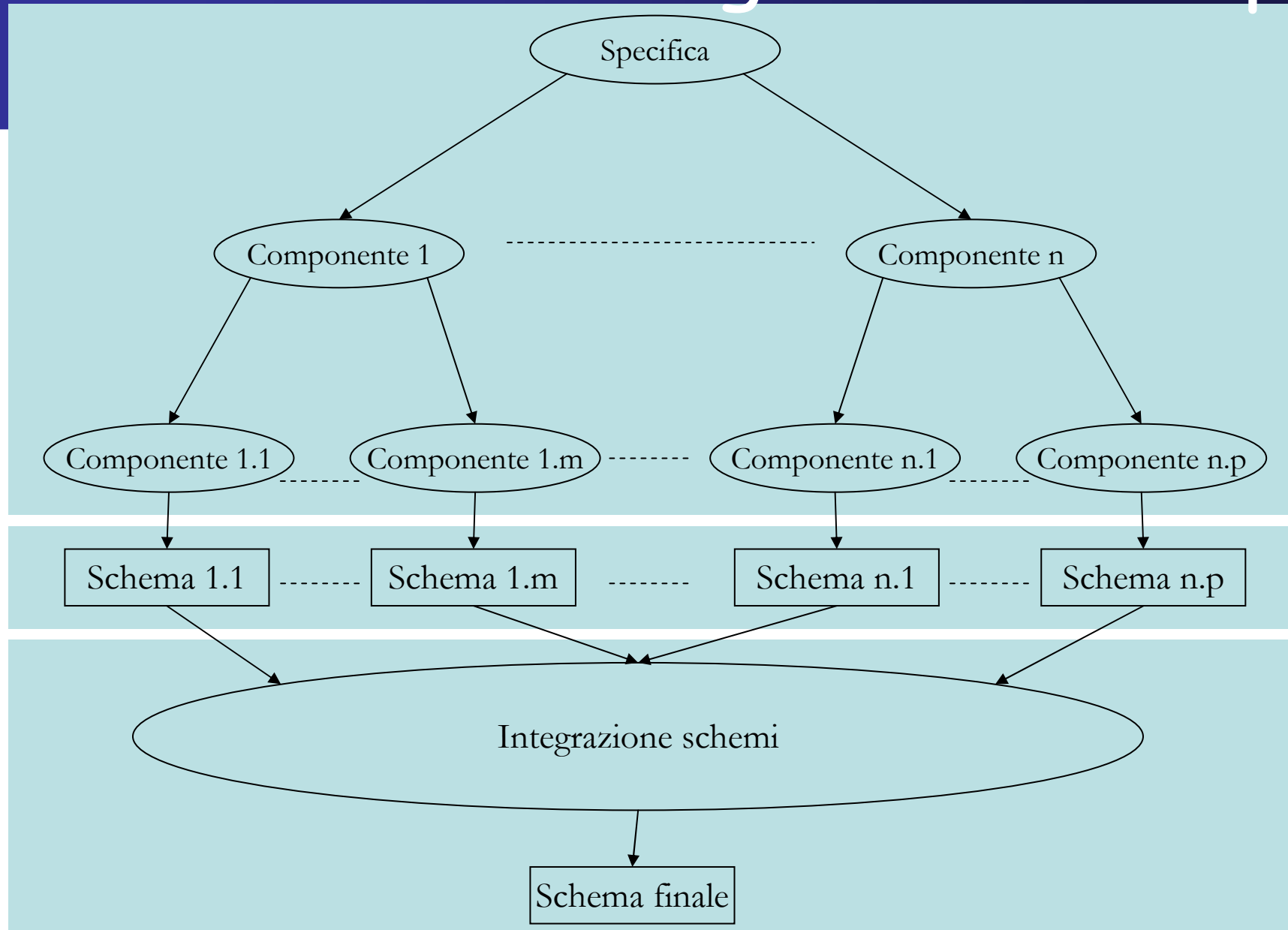
# Valutazione

- vantaggi:
  - il progettista descrive inizialmente lo schema trascurando i dettagli
  - precisa lo schema gradualmente
- problema:
  - non va bene per applicazioni complesse perché è difficile avere una visione globale precisa iniziale di tutte le componenti del sistema

# Strategia bottom-up

- le specifiche nascono suddivise per sottoprogetti descrittivi frammenti limitati della realtà da schematizzare
- si sviluppano i sottoschemi separati
- si fondono i sottoschemi per ottenere lo schema finale

# Strategia bottom-up



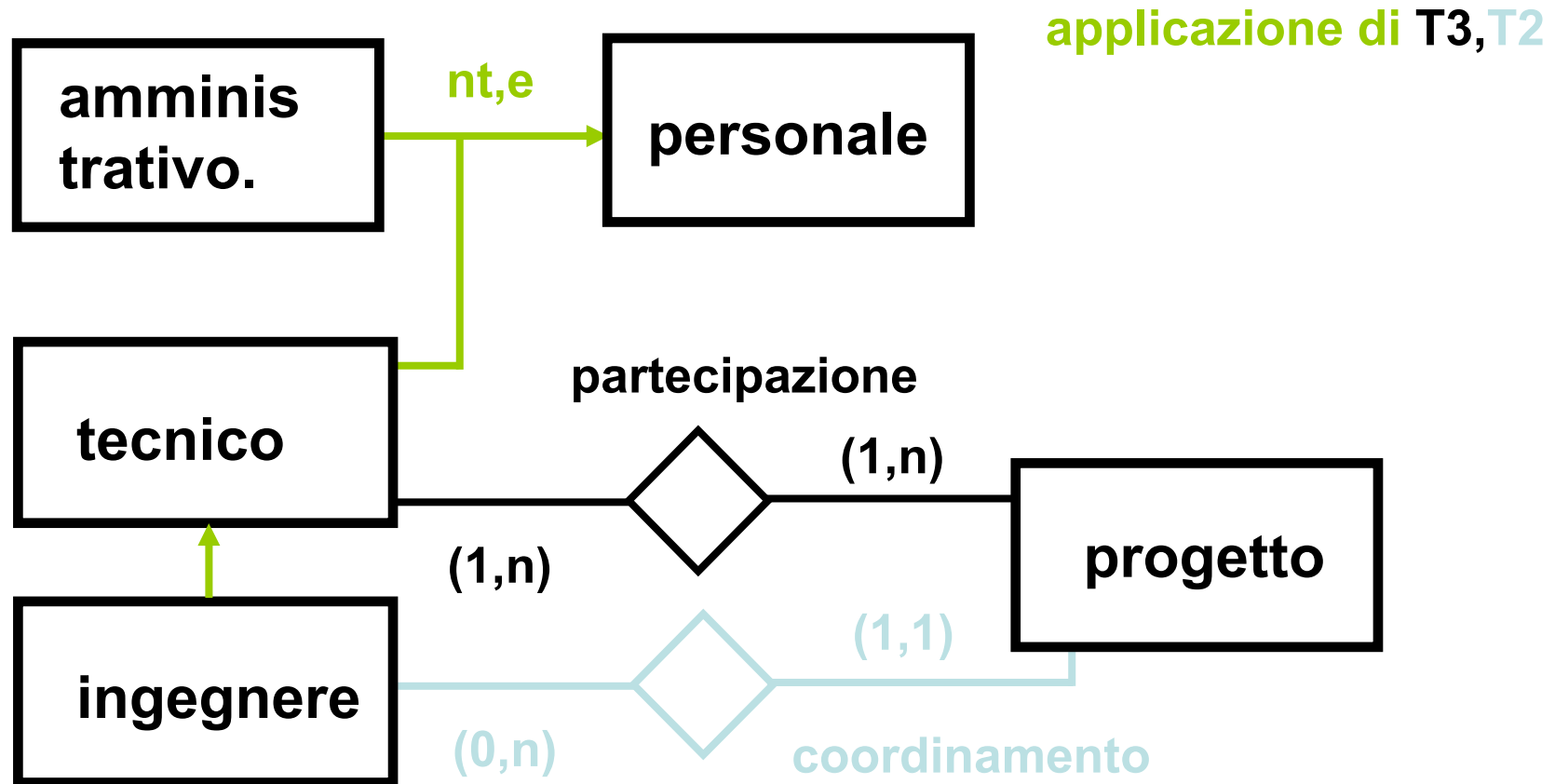
# Primitive di trasformazione

## Bottom-Up

- **Trasformazione**

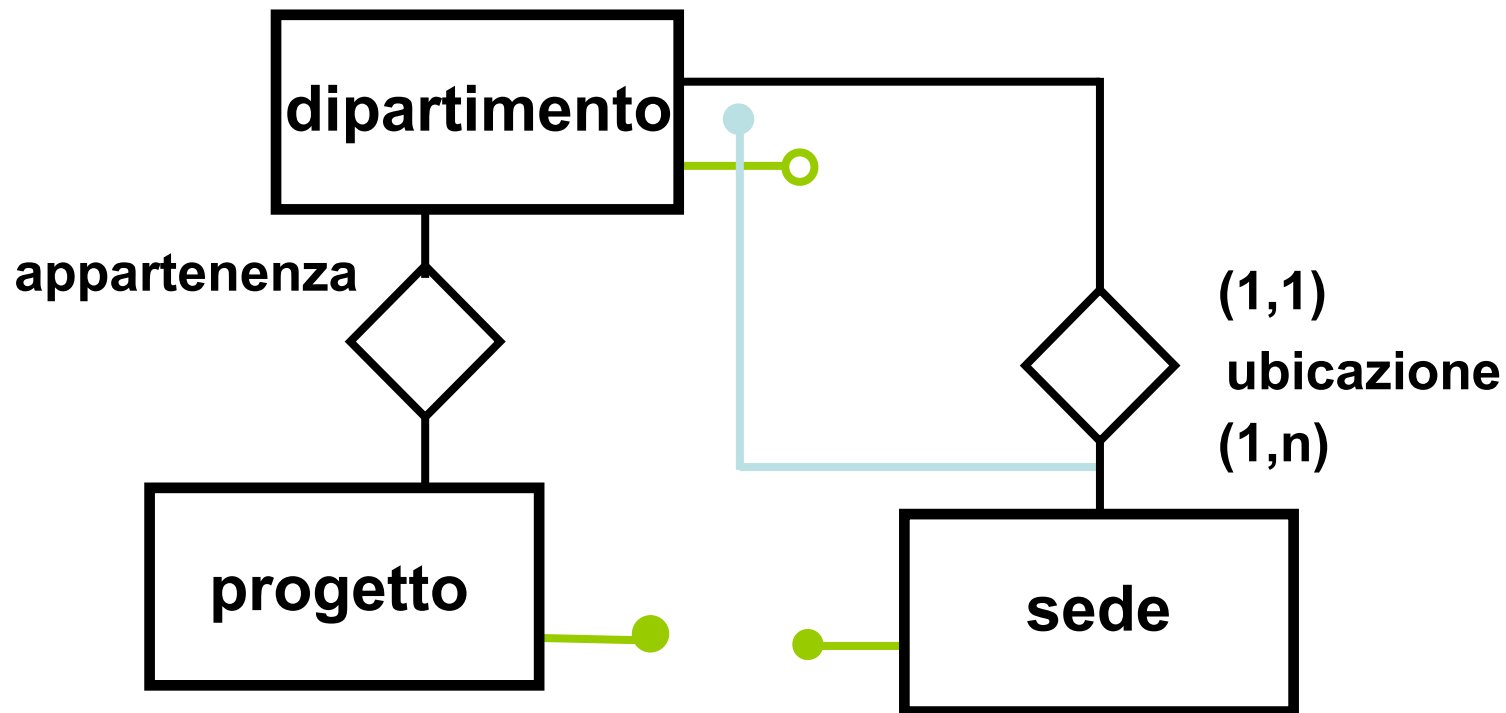
- **T1**: si individua nella specifica una classe di oggetti con proprietà comuni e si introduce un'entità corrispondente.
- **T2**: si individua nella specifica un legame logico fra entità e si introduce una **associazione** fra esse.
- **T3**: si individua una **generalizzazione** fra entità.
- **T4**: a partire da una serie di **attributi** si individua un'entità che li aggrega.
- **T5**: a partire da una serie di **attributi** si individua una relazione che li aggrega.

# Sviluppo bottom-up: schema 1

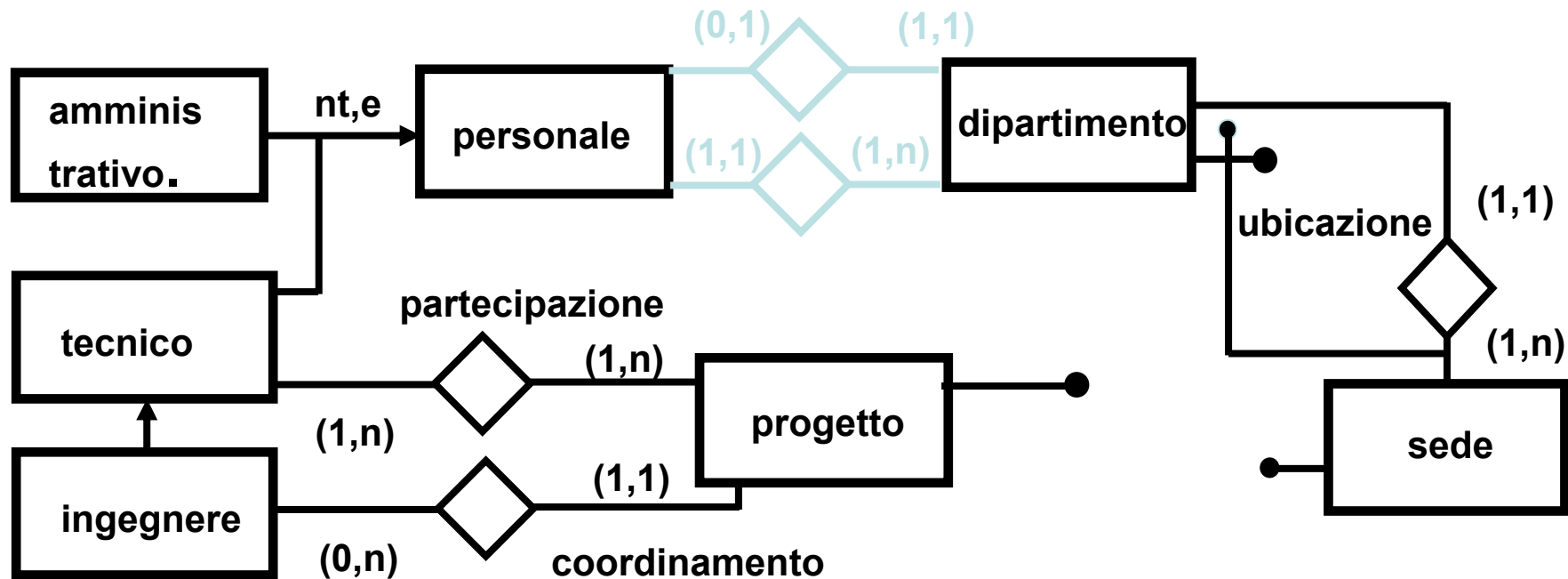


# Sviluppo bottom-up: schema 2

applicazione di T4



# Sviluppo bottom-up: schema integrato



**NB:** l'associazione **appartenenza** è ridondante se un progetto appartiene sempre al dipartimento a cui afferisce il suo coordinatore

# Vantaggi e Svantaggi della Strategia Bottom-Up

- Si adatta bene ad una progettazione di gruppo in cui , diversi progettisti possono sviluppare parti disgiunte che possono essere assemblate successivamente.
- L'integrazione di sistemi concettualmente diversi comporta notevoli difficoltà.

# Ulteriori strategie

- **inside-out:** è una variante della bottom-up, si sviluppano schemi parziali in aggiunta a sottoschemi già definiti precedentemente e separatamente. Inizialmente si sviluppano alcuni concetti e poi si estendono a macchia d'olio.
- **strategia mista:** cerca di combinare i vantaggi top-down e bottom-up: il progettista divide i requisiti in componenti separate (come nel bottom-up) ma, allo stesso tempo, definisce uno *schema scheletro*, contenente, a livello astratto, i concetti principali dell'applicazione. Questo fornisce una visione unitaria, anche se astratta, dell'intero progetto e può *guidare* le fasi di integrazione dei sottoschemi

# Vantaggi e Svantaggi dell'Inside-Out

- Ha il vantaggio di non richiedere passi di integrazione.
- D'altra parte è necessario, di volta in volta, esaminare tutte le specifiche per individuare concetti non ancora rappresentati e descriverli nel dettaglio.

# Vantaggi della strategia mista

- è la più flessibile perché permette di suddividere i problemi in sottoproblemi (bottom-up) e di procedere per raffinamenti successivi (top-down).
- è applicabile nei casi pratici in cui quando si inizia la progettazione non sono ancora disponibili tutti i dati e, dei vari dati, non abbiamo descrizioni a livello diverso di dettaglio.

# Metodologia Generale

- **Analisi requisiti**
  - Costruire glossario dei termini
  - Analizzare I requisiti ed eliminare ambiguità
  - Raggruppare i requisiti in insiemi omogenei
- **Passo base**
  - Individuare I concetti più rilevanti e rappresentarli in uno schema scheletro

# Metodologia Generale

- Passo di decomposizione
  - //Da effettuare se opportuno o necessario
  - Effettuare una decomposizione dei requisiti con riferimento ai concetti presenti nello schema scheletro

# Metodologia Generale

- Passo iterativo
  - //da ripetere a tutti i sottoschemi (se presenti) finché ogni specifica è stata rappresentata
  - Raffinare I concetti presenti sulla base delle loro specifiche
  - Aggiungere nuovi concetti allo schema per descrivere specifiche non ancora descritte

# Metodologia Generale

- Passo di integrazione
  - //da effettuare se sono presenti diversi sottoschemi
  - Integrare I vari sottoschemi in uno schema generale facendo riferimento allo schema scheletro
- Analisi di qualità

# Metodologia Generale

- Se i passi di Decomposizione e Integrazione e il passo iterativo in cui si aggiungono nuovi concetti non vengono effettuati si ottiene una strategia TOP DOWN
- Se il passo base non viene effettuata e nel passo iterativo vengono aggiungono solo nuovi concetti allora si ottiene una strategia BOTTOM-UP

# Qualità di uno Schema Concettuale

- Viene giudicata in base a delle proprietà che lo schema deve possedere:
  - Correttezza
  - Completezza
  - Leggibilità
  - Minimalità

# Correttezza e Completezza

- **Correttezza:** se si utilizzano propriamente i costrutti. Gli errori possono essere **sintattici** : uso non ammesso dei costrutti (ad esempio generalizzazione fra relazioni) o **semantici** : uso che non rispetta il loro significato ( si usa una relazione per descrivere che un'entità è generalizzazione di un'altra).
- **Completezza:** *tutti i dati di interesse sono rappresentati e tutte le operazioni possono essere eseguite a partire dai concetti dello schema*

# Leggibilità

- Uno schema è **leggibile** quando rappresenta i requisiti in maniera naturale e facilmente comprensibile. Alcune regole:
  - *disporre al centro* i costrutti con più legami
  - usare linee perpendicolari cercando di minimizzare le intersezioni.
  - Disporre i padri di generalizzazioni sopra i figli
- Verificare con gli utenti la leggibilità

# Minimalità

- Uno schema è **minimale** quando tutte le specifiche sono rappresentate una sola volta. Non devono contenere *ridondanze* ovvero concetti deducibili da altri oppure cicli di relazioni e generalizzazioni.
- Una ridondanza a volte può nascere da una scelta precisa di progettazione

# Progettazione logica

## 1<sup>^</sup> parte

# Progettazione Logica

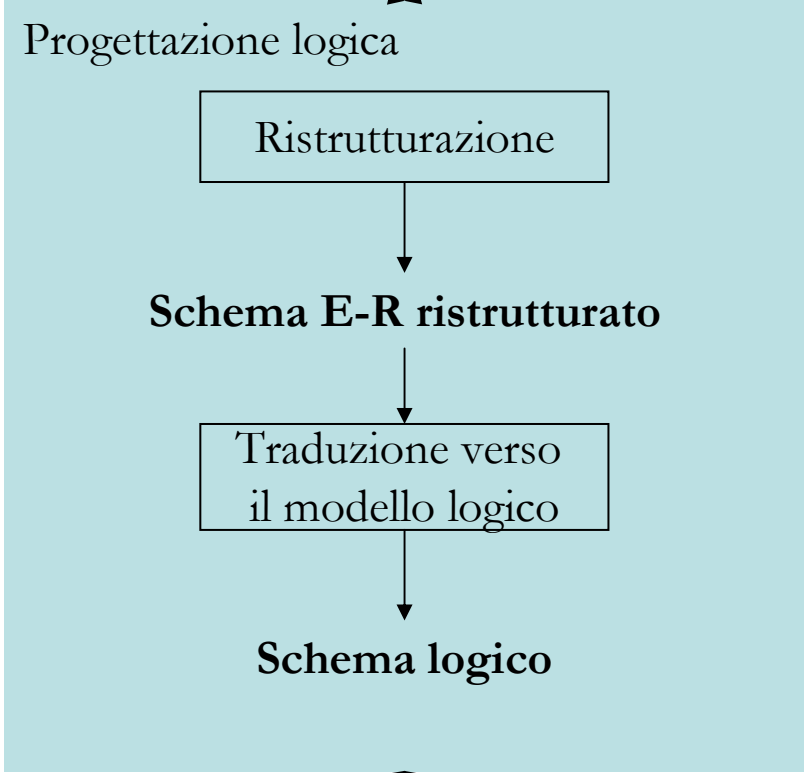
- Obiettivo della **Progettazione Logica** e' quello di costruire uno schema logico ,in un determinato *modello* (ad es. *relazionale*), che descriva in maniera *corretta ed efficiente* tutte le informazioni contenute nello schema E-R prodotto dalla progettazione concettuale.
- Non si tratta di una *semplice traduzione*

# Fasi della Progettazione Logica

- **Ristrutturazione dello schema E-R:**
  - e' una fase indipendente dal modello logico e si basa su criteri di *ottimizzazione* dello schema e di successiva *semplificazione*.
- **Traduzione verso il Modello Logico:**
  - fa riferimento ad un modello logico (ad es. relazionale) e puo' includere ulteriore *ottimizzazione* che si basa sul modello logico stesso (es. normalizzazione).

# Input ed output della prima fase

- **Input:**
  - *Schema Concettuale E-R iniziale, Carico Applicativo* previsto (in termini di dimensione dei dati e caratteristica delle operazioni)
- **Output :**
  - *Schema E-R ristrutturato* che rappresenta i dati e tiene conto degli aspetti realizzativi



Vincoli di integrità

Schema logico

Documentazione

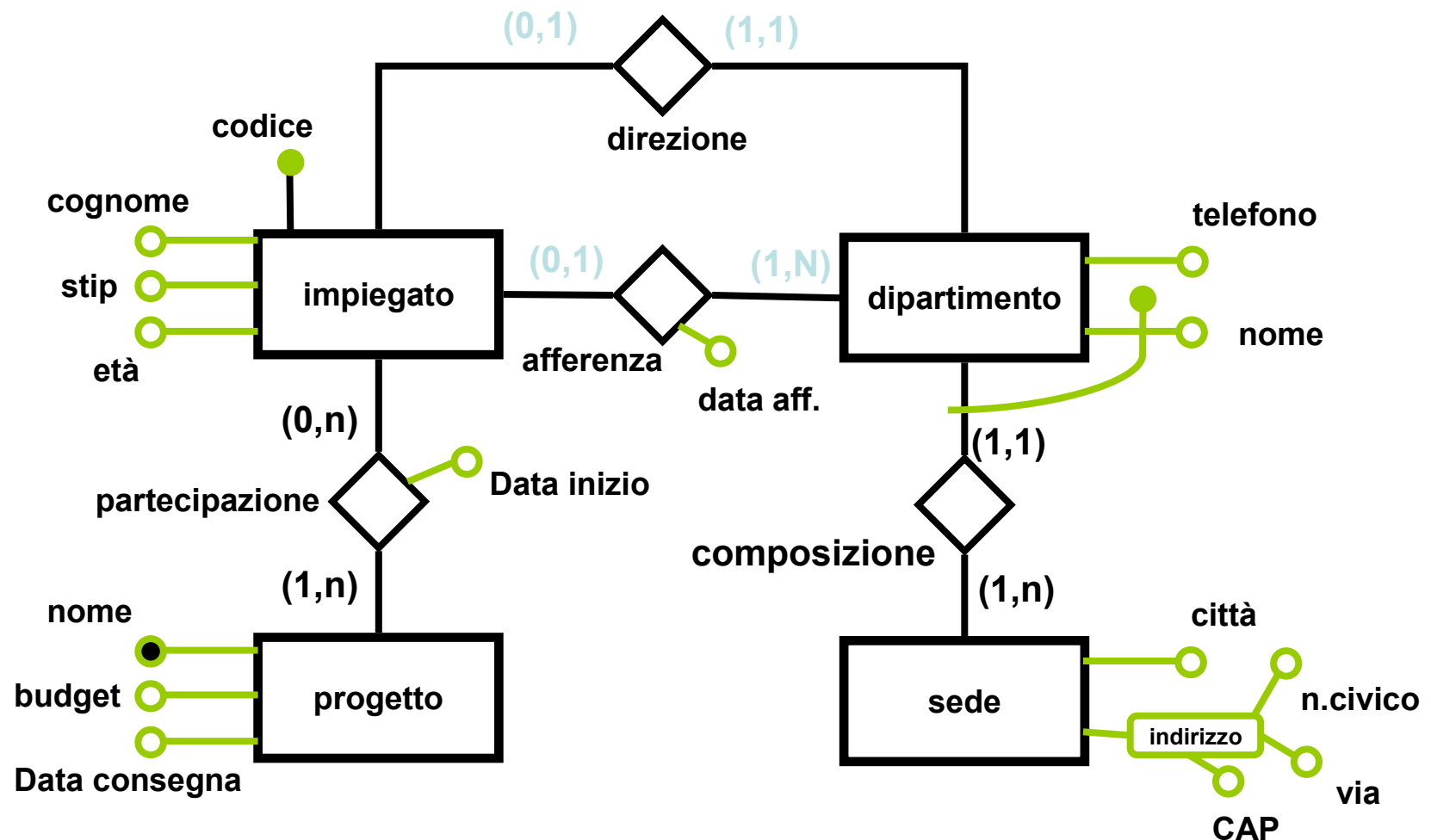
# Analisi delle prestazioni su schemi E-R

- *Indici di prestazione* per la valutazione di schemi E-R sono due:
  - **Costo di un'operazione:** in termini di numero di occorrenze di entità ed associazioni che mediamente vanno visitate per rispondere a quella operazione sulla base di dati (talvolta sarà necessario raffinare questo criterio)
  - **Occupazione di memoria:** viene valutata in termini dello spazio di memoria (misurato in byte) necessario per memorizzare i dati del sistema.

# Volumi e Caratteristiche dei dati

- Per studiare questi due parametri abbiamo bisogno di conoscere:
- **Volume dei dati:**
  - a) numero (medio) di occorrenze di ogni entita' ed associazione
  - b) dimensioni di ciascun attributo
- **Caratteristiche delle operazioni:**
  - a) tipo di operazione (interattiva o batch)
  - b) frequenza (esecuzioni/tempo)
  - c) dati coinvolti (entita' e o associazioni)

# Esempio di analisi: ditta con sedi in città diverse



# Operazioni dell'esempio

- **Operazione 1:** assegna un impiegato ad un progetto
- **Operazione 2:** trova i dati di un impiegato, del dipartimento nel quale lavora e dei progetti in cui e' coinvolto
- **Operazione 3:** trova i dati di tutti gli impiegati di un certo dipartimento
- **Operazione 4:** per ogni sede, trova i dipartimenti con il cognome del direttore e l'elenco degli impiegati.

## TABELLA DEI VOLUMI

Concetto	Tipo	Volume
Sede	<b>E</b>	<b>10</b>
Dipartimento	<b>E</b>	<b>80</b>
Impiegato	<b>E</b>	<b>2000</b>
Progetto	<b>E</b>	<b>500</b>
Composizione	<b>R</b>	<b>80</b>
Afferenza	<b>R</b>	<b>1900</b>
Direzione	<b>R</b>	<b>80</b>
Partecipazione	<b>R</b>	<b>6000</b>

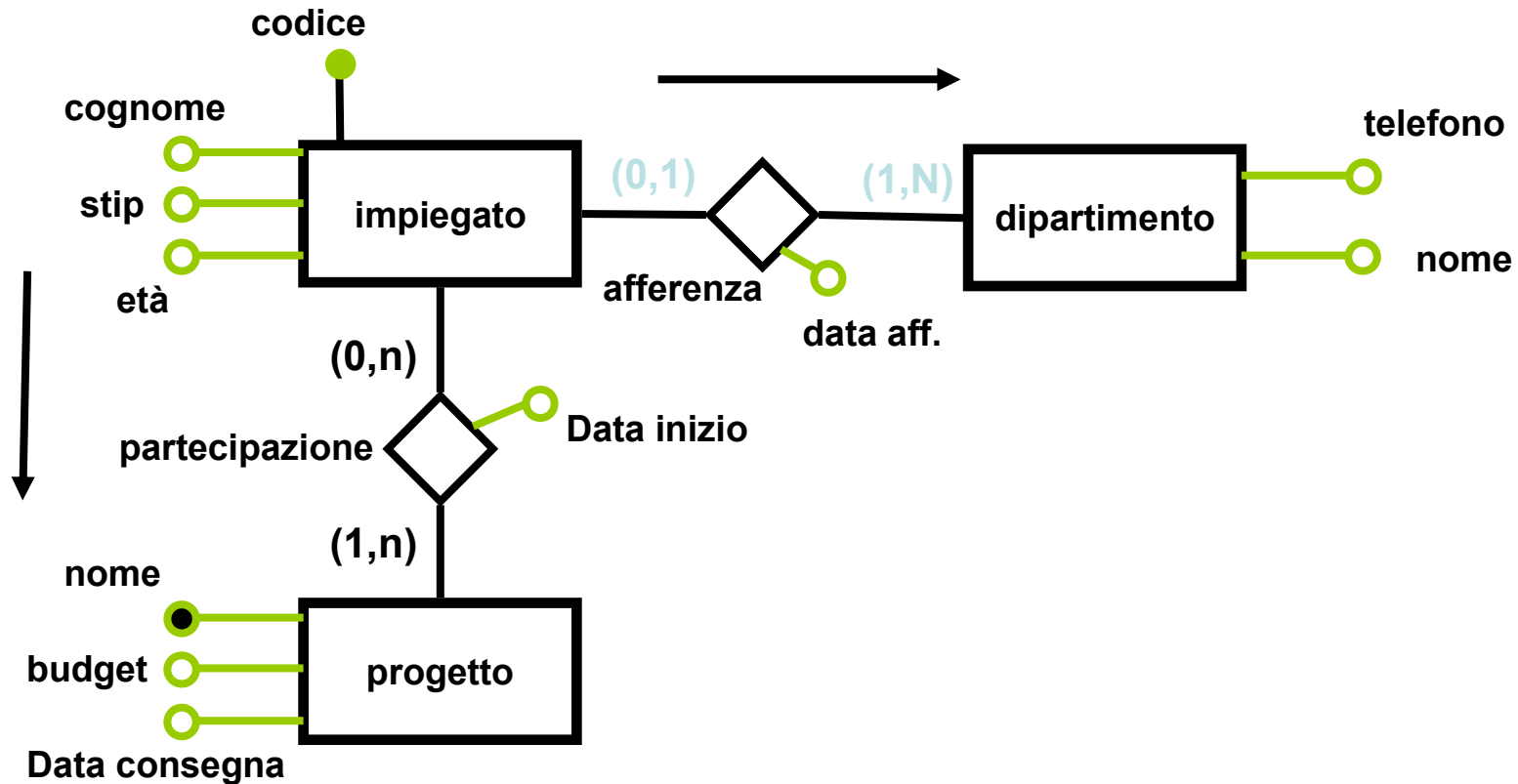
## TABELLA DELLE OPERAZIONI

Concetto	Tipo	Frequenza
Op. 1	<b>I</b>	<b>50 al giorno</b>
Op. 2	<b>I</b>	<b>100 al giorno</b>
Op. 3	<b>I</b>	<b>10 al giorno</b>
Op. 4	<b>B</b>	<b>2 a settimana</b>

# Stima dei costi

- Avendo a disposizione questi dati è possibile stimare i costi di ogni operazione contando il numero di accessi alle occorrenze di entità e relazioni necessario per eseguire l'operazione.
- Prendiamo per esempio *Operazione 2: trova i dati di un impiegato, del dipartimento nel quale lavora e dei progetti in cui e' coinvolto* e facciamo riferimento allo schema di operazione

# Esempio dell' operazione 2



# Stima del costo dell'operazione 2

- Dobbiamo accedere ad:
  - un'occorrenza di *Impiegato* e di *Afferenza* e quindi di *Dipartimento*;
  - Successivamente, per avere i dati dei progetti a cui lavora, dobbiamo accedere (in media) a tre occorrenze di *Partecipazione* e quindi a tre entità *Progetto*.
  - Tutto viene riassunto nella tavola degli accessi

# Tavola degli accessi

CONCETTO	COSTRUTTO	ACCESSI	TIPO
Impiegato	<b>Entita'</b>	1	L
Afferenza	<b>Relazione</b>	1	L
Dipartimento	<b>Entita'</b>	1	L
Partecipazione	<b>Relazione</b>	3	L
Progetto	<b>Entita'</b>	3	L

L lettura, S scrittura. In genere la scrittura e' piu' onorosa che la lettura

# Ristrutturazione di schemi E-R

- **Analisi delle Ridondanze:** si decide se eliminare o no eventuali ridondanze.
- **Eliminazione delle Generalizzazioni:** tutte le generalizzazioni vengono analizzate e sostituite da altro.
- **Partizionamento/Accorpamento di entita' ed associazioni:** si decide se partizionare concetti in piu' parti o viceversa accorpare.
- **Scelta degli identificatori primari:** si sceglie un identificatore per quelle entita' che ne hanno piu' di uno

**Schema E-R**

**Carico  
Applicativo**

Ristrutturazione dello schema E-R

Analisi delle  
ridondanze

Eliminazione delle  
generalizzazioni

Partizionamento/ Accorpamento  
di Entita e associazioni

Scelta degli identificatori  
principali

**Schema  
E-R Ristrutturato**

# Ristrutturazione di schemi E-R

- **Analisi delle Ridondanze:** si decide se eliminare o no eventuali ridondanze.
- **Eliminazione delle Generalizzazioni:** tutte le generalizzazioni vengono analizzate e sostituite da altro.
- **Partizionamento/Accorpamento di entità ed associazioni:** si decide se partizionare concetti in più parti o viceversa accorpare.
- **Scelta degli identificatori primari:** si sceglie un identificatore per quelle entità che ne hanno più di uno

# Analisi delle Ridondanze

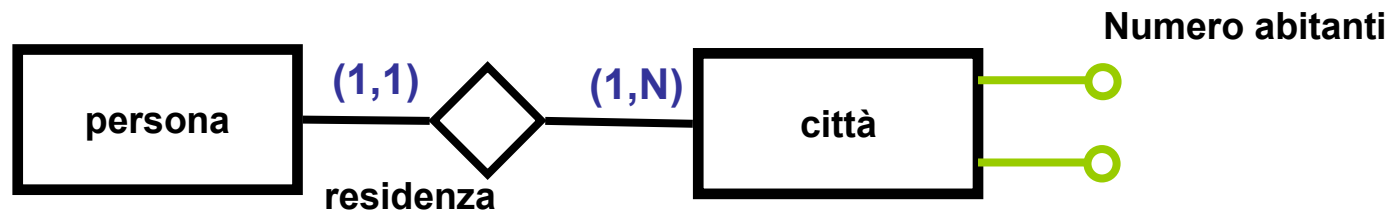
- *Attributi derivabili da altri attributi della stessa entità (fattura: importo lordo)*
- *Attributi derivabili da attributi di altre entità (o associazioni) (Acquisto: Importo totale da Prezzo )*
- *Attributi derivabili da operazioni di conteggio (Città: Numero abitanti contando il numero di Residenza )*
- *Associazioni derivabili dalla composizione di altre associazioni in presenza di cicli. (Docenza da Frequenza ed Insegnamento). Tuttavia i cicli non necessariamente generano ridondanze.*

# Dato derivabile

- *Vantaggi*: riduce gli accessi per calcolare il dato derivato.
- *Svantaggi*: occupazione di memoria e necessita' di effettuare operazioni aggiuntive per mantenere il dato aggiornato.
- **Decisione: mantenere o eliminare?**
  - Basta confrontare i costi di esecuzione delle operazioni sull'oggetto

# Esempio

- Consideriamo l'esempio Città-Persona per l'anagrafica di una regione.
  - *Operazione 1*: memorizza una persona nuova con la relativa città.
  - *Operazione 2*: stampa tutti i dati di una città (incluso il numero di abitanti).
- Valutiamo gli indici di prestazione per l'attributo Numero Abitanti



Concetto	Tipo	Volume
Città	E	200
Persona	E	1000000
Residenza	R	1000000

Operazione	Tipo	Frequenza
Op. 1	I	500 al giorno
Op. 2	I	2 al giorno

# Valutazione in presenza della ridondanza

- Assumendo che il numero di abitanti richieda 4 byte il dato richiede  $4 \times 200 = 800$  byte.
  - Operazione 1 richiede un accesso in scrittura a Persona uno in scrittura a Residenza ed uno in lettura ed uno in scrittura (per incrementare il numero di abitanti) a Città ripetuto 500 volte si hanno 1500 accessi in scrittura e 500 in lettura.
  - L'operazione 2 richiede un solo accesso in lettura a Città 2 volte al giorno.
  - Supponendo che la scrittura ha un costo doppio rispetto ad una lettura si hanno 3500 accessi al giorno in presenza della ridondanza.

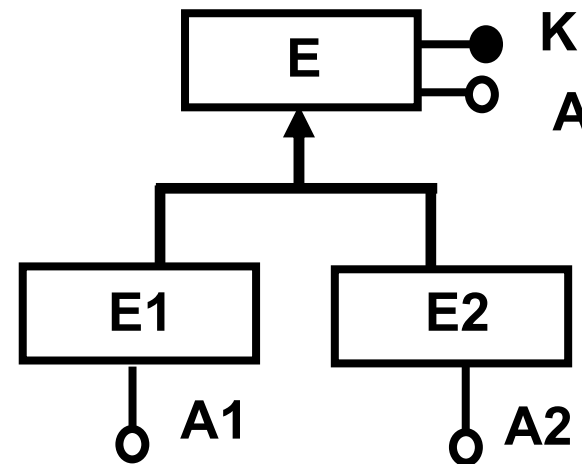
# Valutazione in assenza della ridondanza

- Per l'operazione 1, un accesso in scrittura a Persona ed uno in scrittura a Residenza per un totale di 1000 accessi in scrittura al giorno.
- Per l'operazione 2 abbiamo bisogno di un accesso in lettura a Città (possiamo trascurare) e 5000 accessi in lettura a Residenza in media (persone/città) per un totale di 10.000 accessi in lettura al giorno.
- Il totale e' di 12000 accessi in lettura al giorno. Quindi 8500 in più rispetto al caso di ridondanza contro meno di un solo Kilobyte di memoria in più.
- D'altra parte se l'operazione 2 fosse stata richiesta solo 1 volta ogni 4 settimane avremmo avuto  $3500 \times 24 = 84000$  accessi ogni 4 settimane con ridondanza contro 58000 in assenza.

# Eliminazione delle gerarchie

**il modello relazionale non rappresenta le gerarchie, le gerarchie sono sostituite da entità e associazioni:**

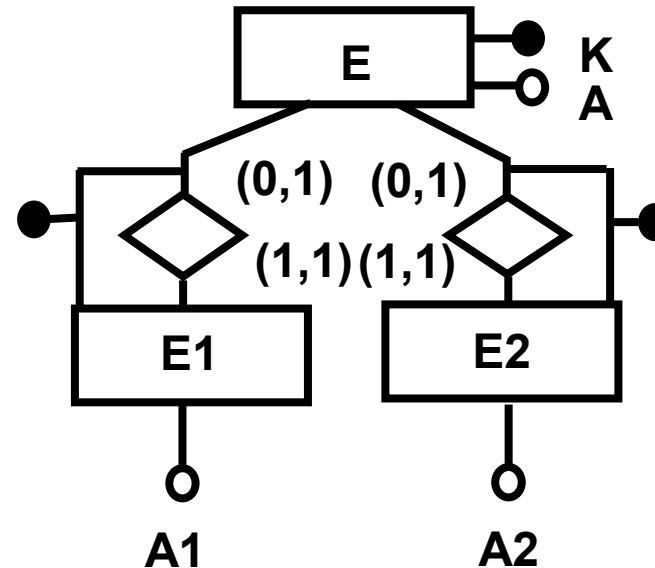
- 1) **mantenimento delle entità con associazioni**
- 2) **collasso verso l'alto**
- 3) **collasso verso il basso**



**l'applicabilità e la convenienza delle soluzioni dipendono dalle proprietà di copertura e dalle operazioni previste**

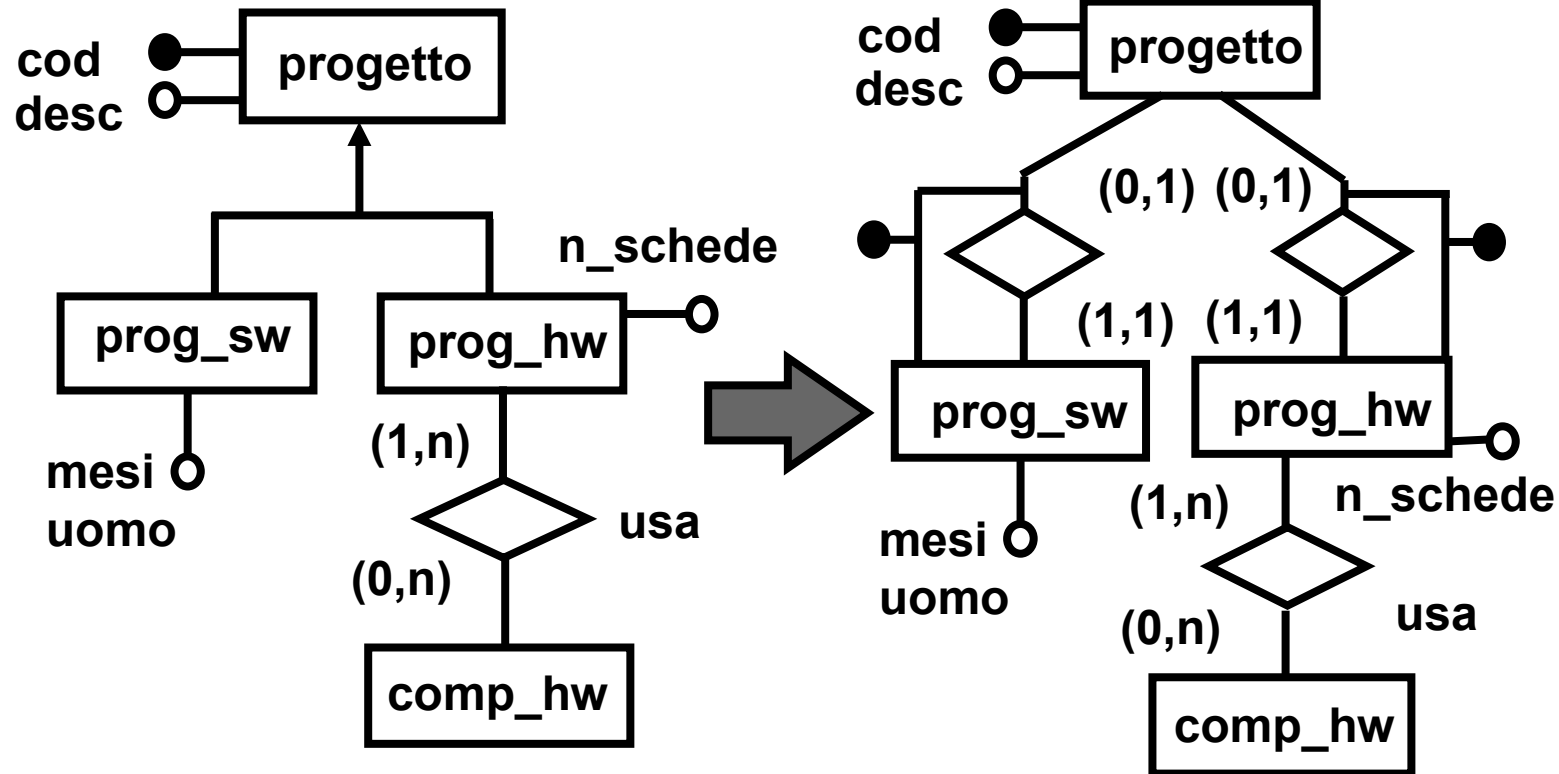
# mantenimento delle entità

- **tutte** le entità vengono mantenute
- le entità **figlie** sono in associazione con l'entità padre
- le entità figlie sono **identificate esternamente** tramite l'associazione



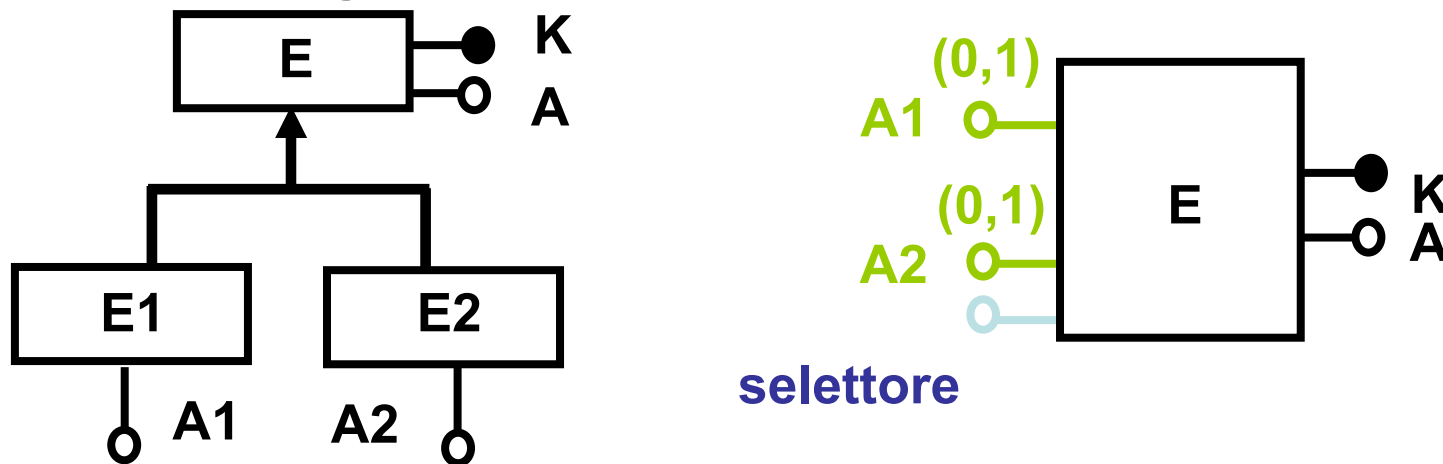
questa soluzione è sempre possibile,  
indipendentemente dalla copertura

# mantenimento entità - es.:



# eliminazione delle gerarchie

- Il **collasso verso l'alto** riunisce tutte le entità figlie nell'entità padre

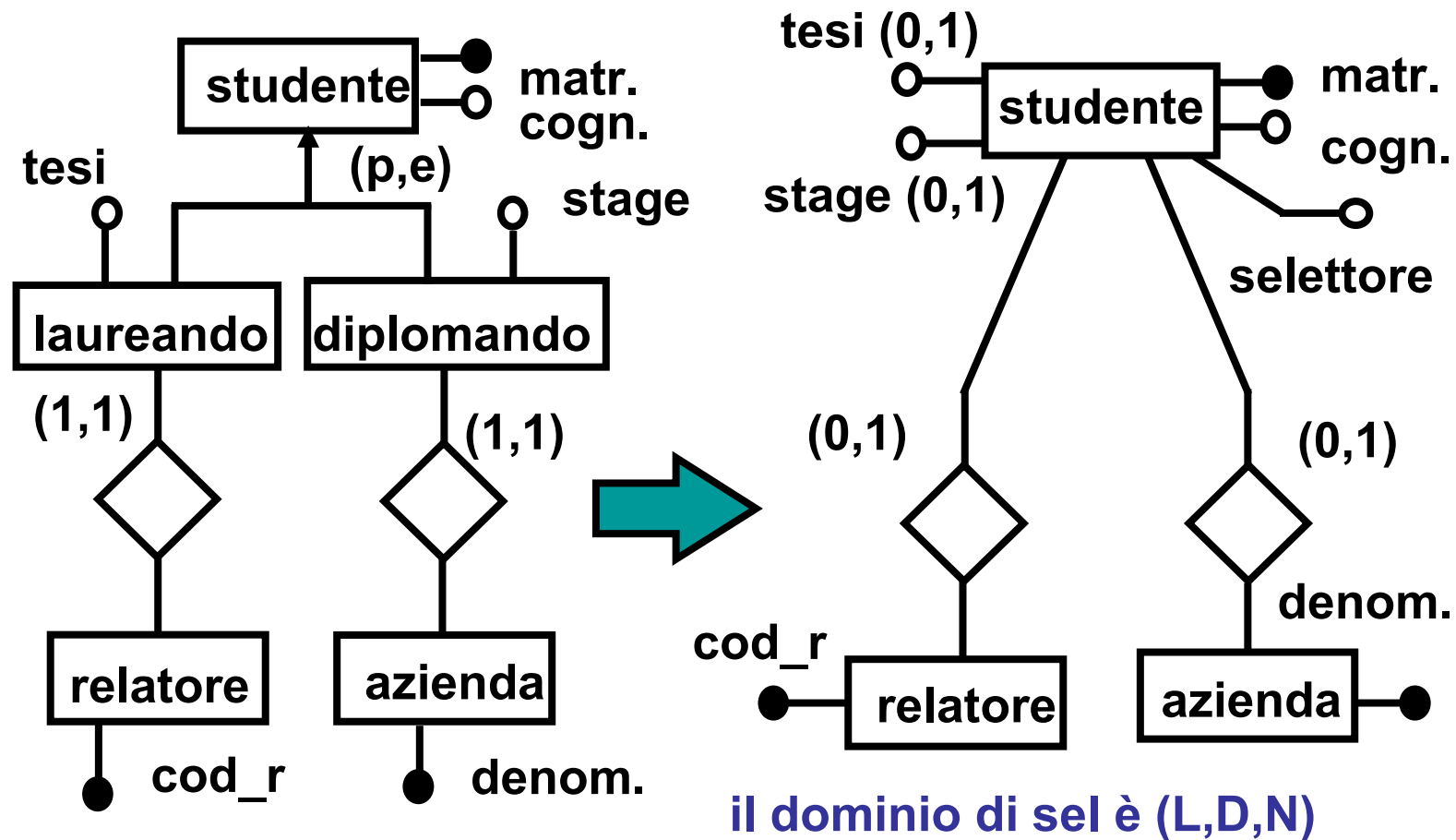


**selettore** è un attributo che specifica se una istanza di E appartiene a una delle sottoentità

# ISA: collasso verso l'alto

- Il collasso verso l'alto favorisce operazioni che consultano insieme gli attributi dell'entità padre e quelli di una entità figlia:
  - in questo caso si accede a una sola entità, anziché a due attraverso una associazione
- gli attributi obbligatori per le entità figlie divengono opzionali per il padre
  - si avrà una certa percentuale di valori nulli

# ISA: collasso verso l'alto



# ISA: collasso verso l'alto

studente(123, rossi)  
studente(218, bianchi)  
studente(312, verdi)

laureando(123, DFD)

diplomando(312, ST)



(selettore)



studente(123,rossi, L, DFD, NULL)  
studente(218,bianchi, N, NULL, NULL)  
studente(312,verdi, D, NULL, ST)

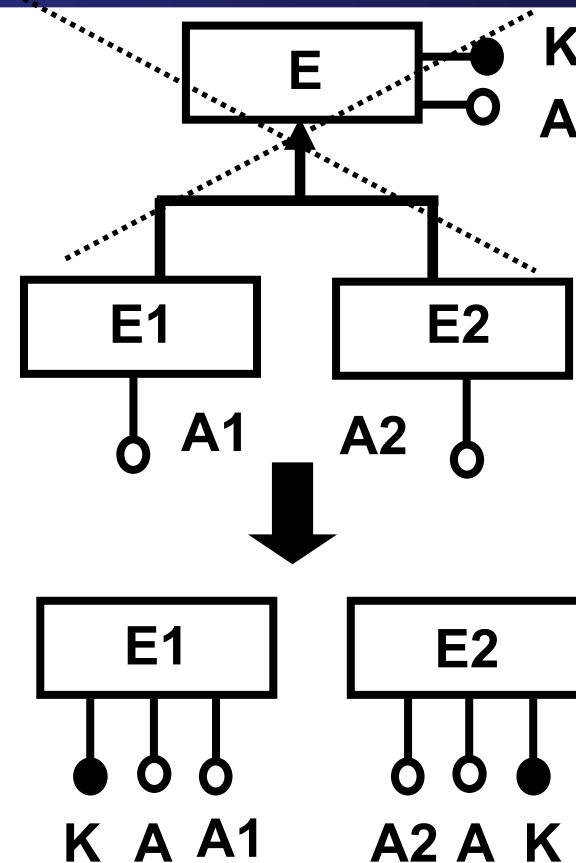
# ISA: collasso verso il basso

- Collasso verso il basso:
- si elimina l'entità padre trasferendone gli attributi su tutte le entità figlie
  - una associazione del padre è replicata, tante volte quante sono le entità figlie
  - la soluzione è interessante in presenza di molti attributi di specializzazione (con il collasso verso l'alto si avrebbe un eccesso di valori nulli)
  - favorisce le operazioni in cui si accede separatamente alle entità figlie

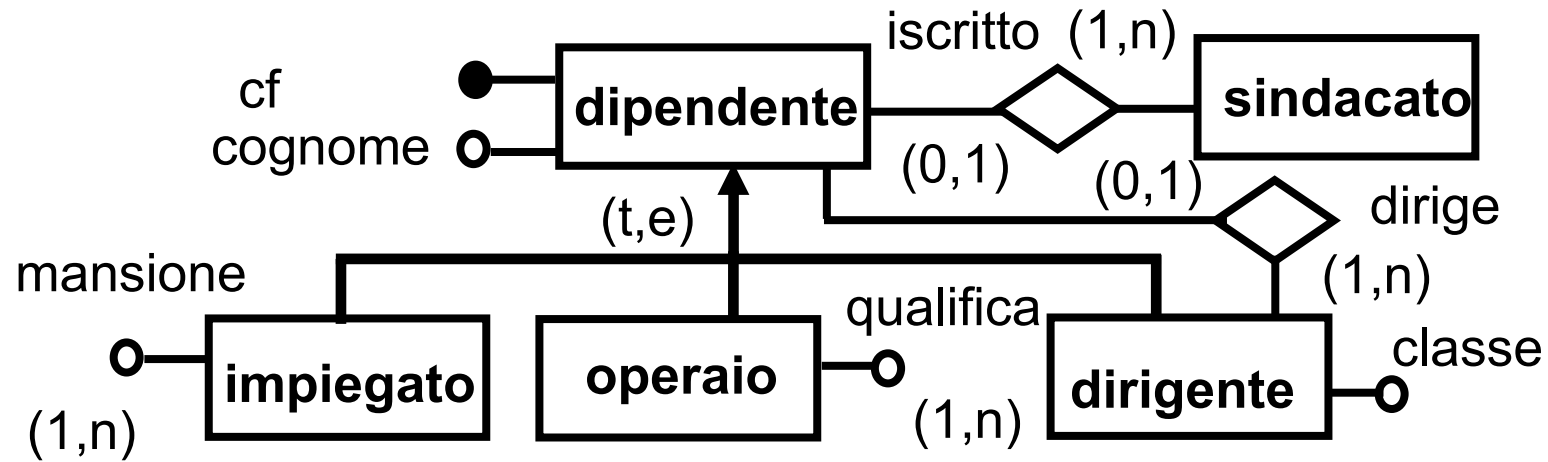
# ISA: collasso verso il basso

limiti di applicabilità:

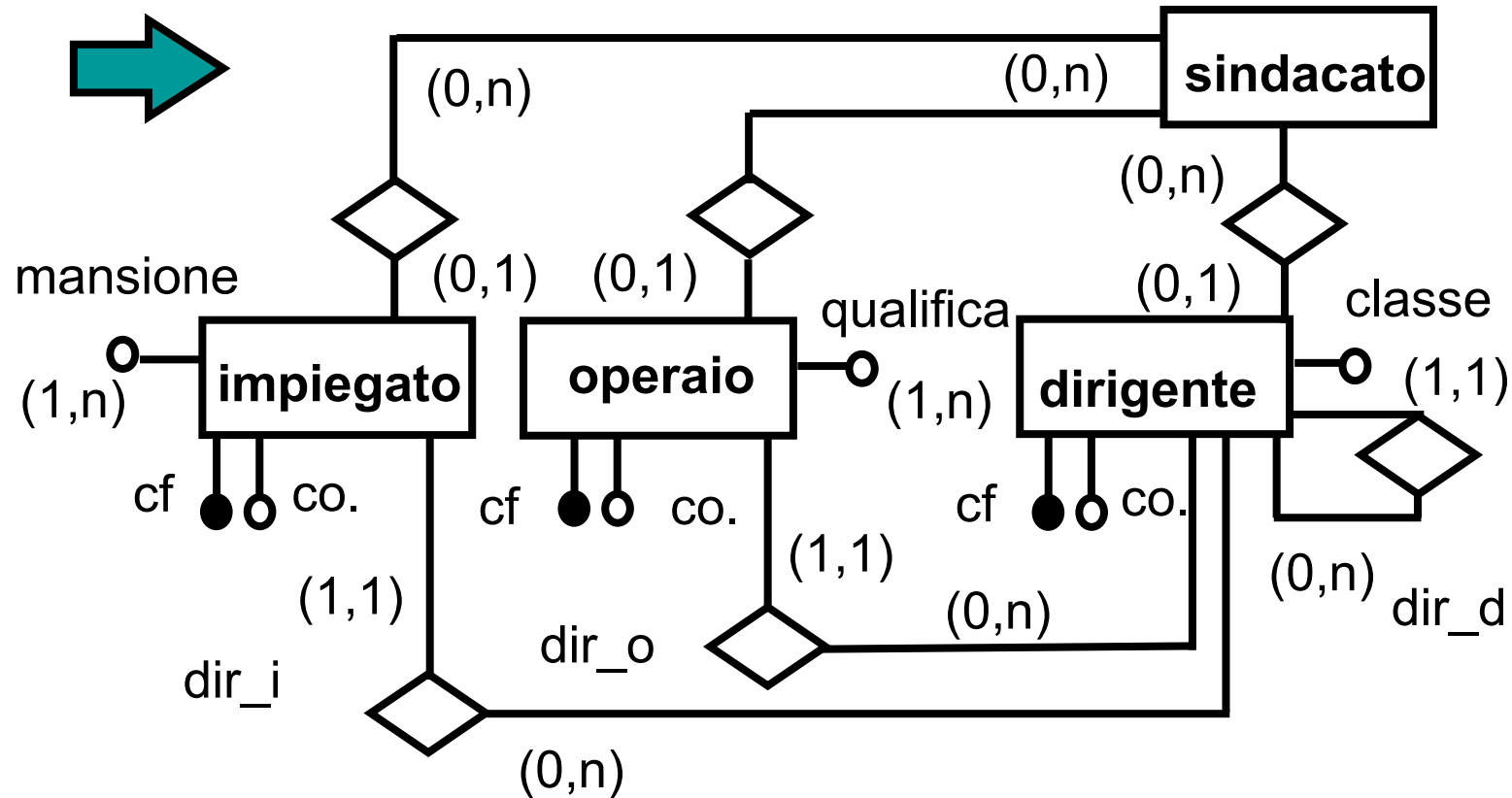
- se la copertura non è totale **non si può fare:**  
**dove mettere gli E che non sono né E1, né E2 ?**
- se la copertura non è esclusiva **introduce ridondanza:** **per una istanza presente sia in E1 che in E2 si rappresentano due volte gli attributi di E**



# collasso verso il basso: es.



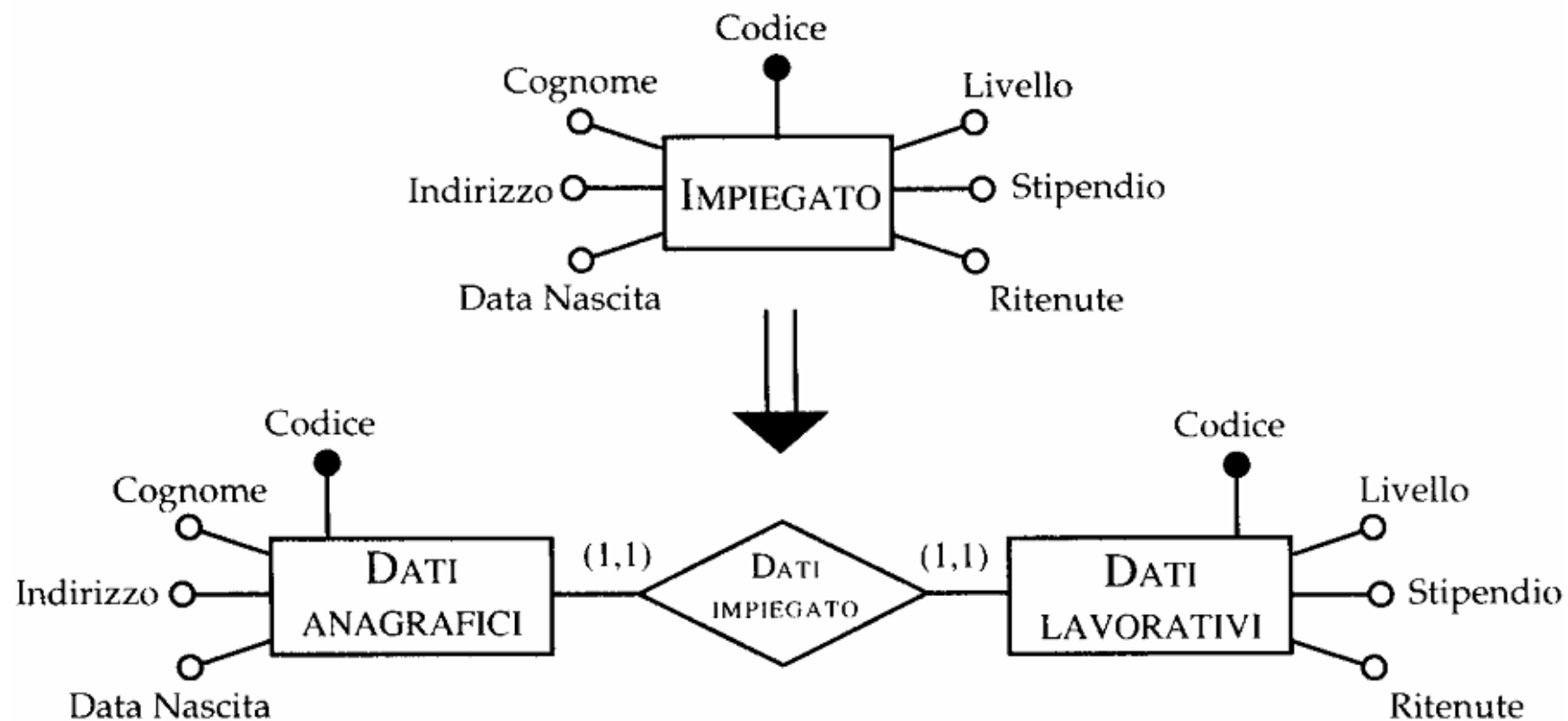
# collasso verso il basso: es.



# Partizionamento/Accorpamento

- Il principio generale e' il seguente: gli accessi si riducono
  - separando attributi di uno stesso concetto che vengono acceduti da operazioni diverse
  - raggruppando attributi di concetti diversi che vengono acceduti dalle medesime operazioni

# Partizionamento di entità

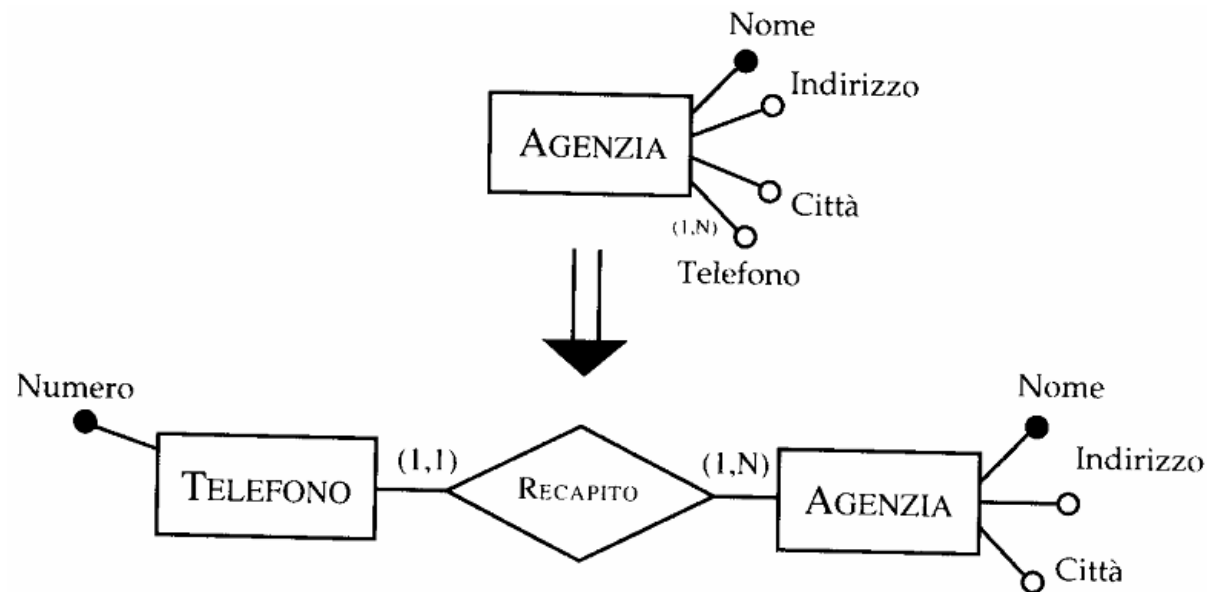


# Partizionamento verticale ed orizzontale

- Nell'esempio precedente vengono create due entità e gli attributi vengono divisi: *partizionamento verticale*
- Se invece si suddivide in due entità con gli stessi attributi (ad esempio Analista e Venditore) con operazioni distinte sulle due si ha il *partizionamento orizzontale*

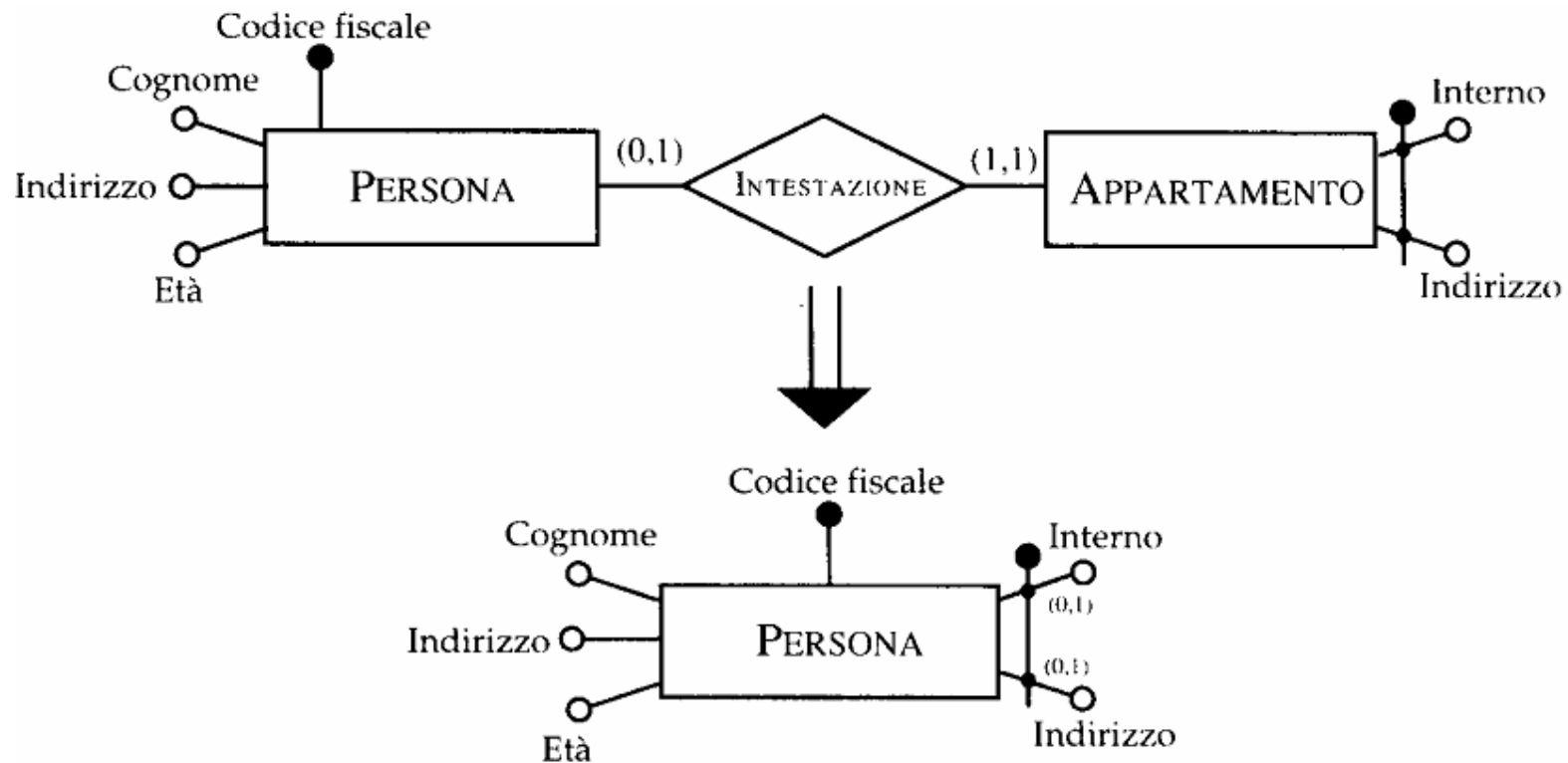
# Eliminazione di attributi multivalore

- Il modello relazionale non li supporta (anche se alcuni sistemi moderni li ammettono)



# Accorpamento di entità

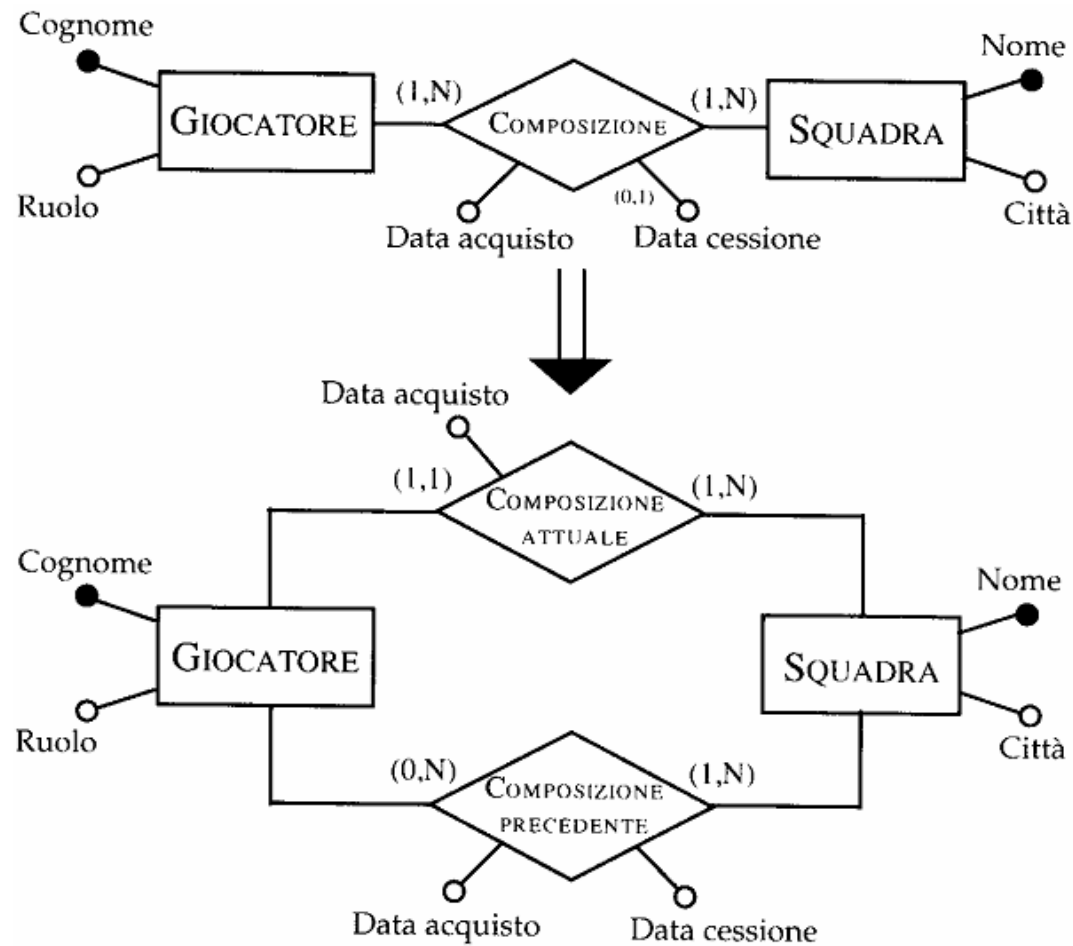
- E' l'operazione inversa del partizionamento



# Quando si fa un accorpamento

- L'accorpamento precedente è giustificato se le operazioni più frequenti su Persona richiedono sempre i dati relativi all'appartamento e quindi vogliamo risparmiare gli accessi alla relazione che li lega. Normalmente gli accorpamenti si fanno su relazioni uno ad uno, raramente su uno a molti mai su molti a molti.

# Partizionamento/Accorpamento di associazioni

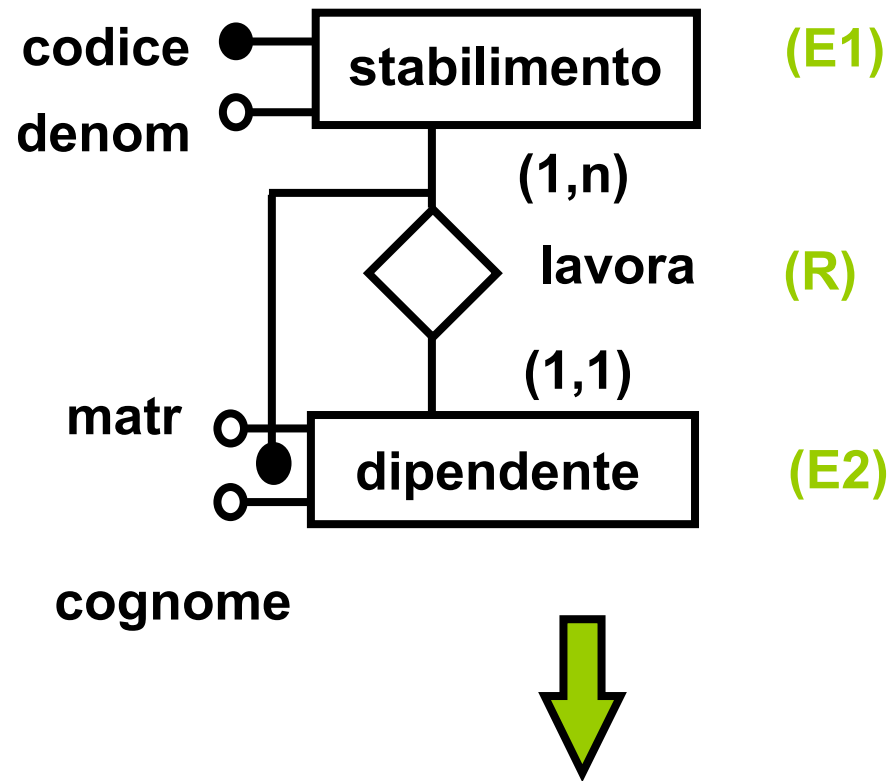


# Scelta della chiave primaria

- È necessario che tra i diversi identificatori di una entità venga designata una chiave primaria:
  - per la chiave primaria occorrerà, infatti, che il DBMS sia provvisto di strumenti per garantire l'unicità dei valori
- criteri euristici di scelta:
  - primo: scegliere la chiave che è usata più frequentemente per accedere all'entità
  - secondo: si preferiscono chiavi semplici a chiavi composte, interne anziché esterne

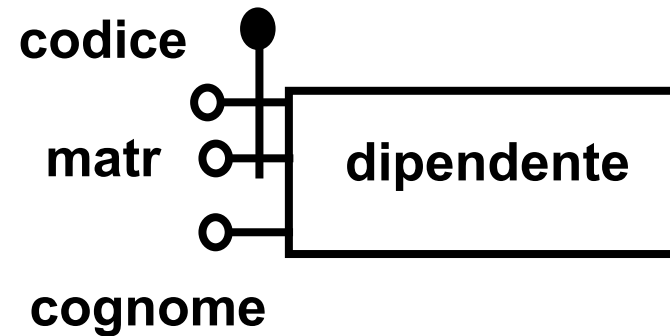
# identificatori esterni

- una componente di identificazione esterna di una entità E2 da una entità E1 attraverso una associazione R comporta il **trasporto della chiave primaria di E1 su E2**



# identificatori esterni

- in questo modo l'associazione è rappresentata attraverso la chiave, e può essere eliminata
- la chiave trasportata è chiave esterna
- in presenza di più identificazioni in cascata, è necessario iniziare la propagazione dall'entità che non ha identificazioni esterne

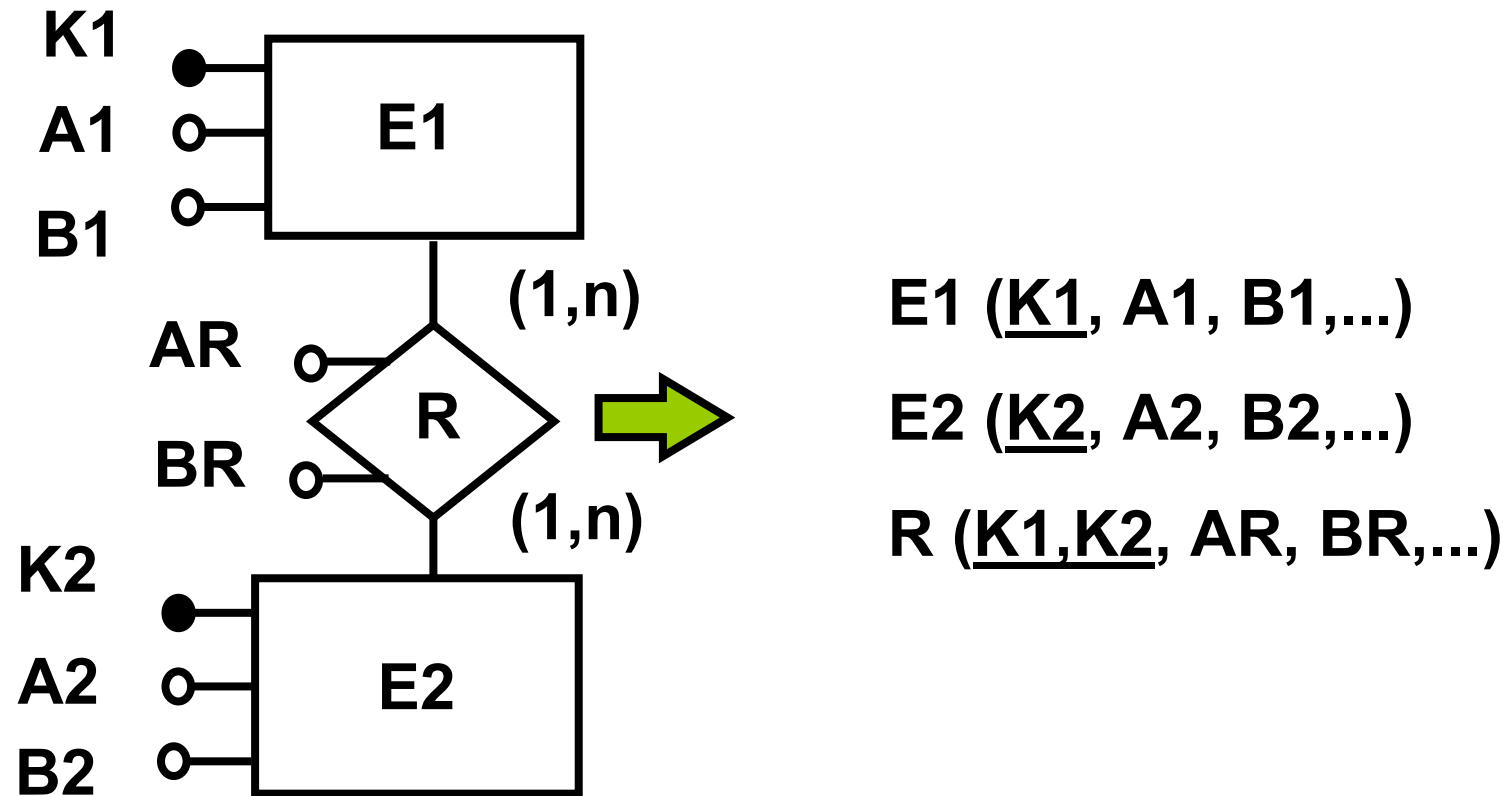


# Traduzione canonica

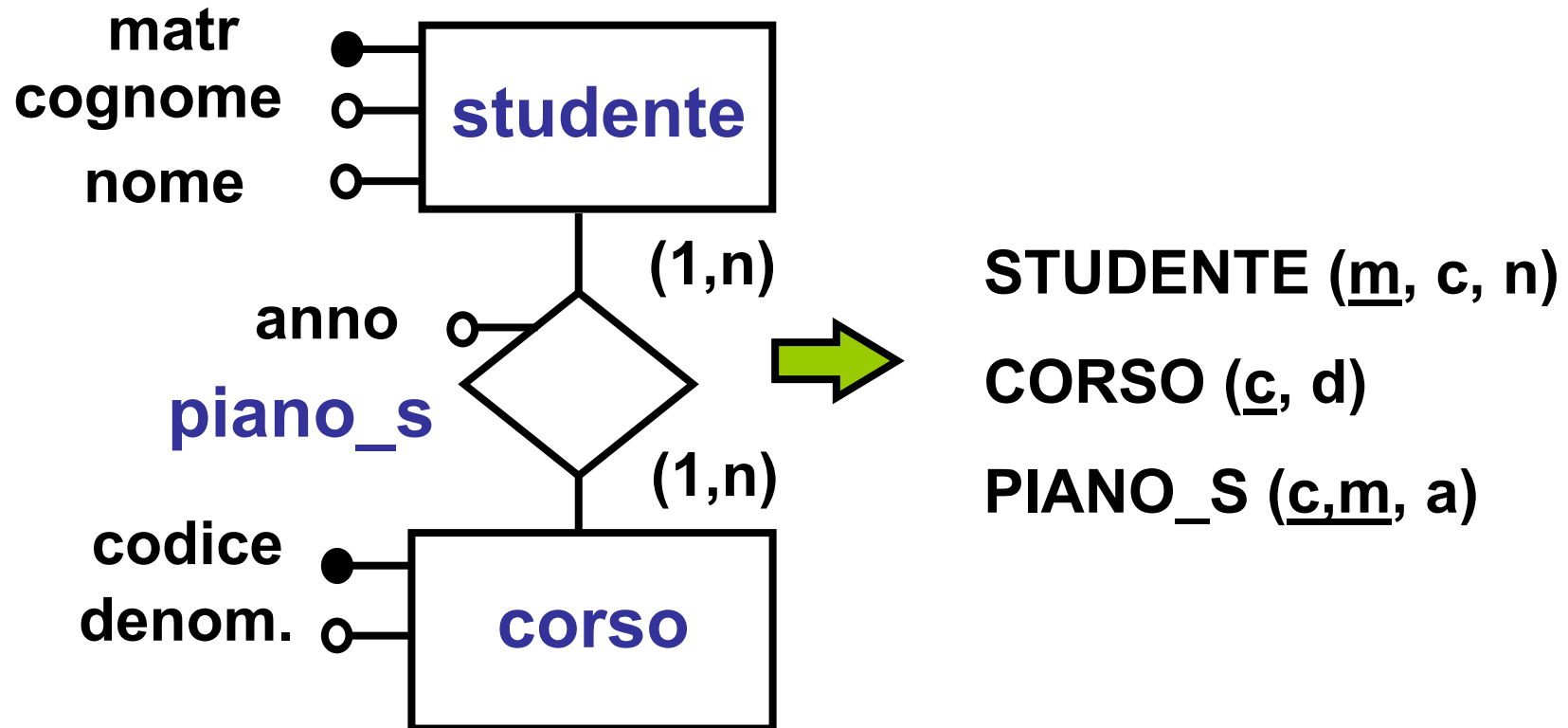
## Entità ed Associazioni molti a molti

- ogni entità è tradotta con una relazione con gli stessi attributi
  - la chiave è la chiave (o identificatore) dell'entità stessa
- ogni associazione è tradotta con una relazione con gli stessi attributi, cui si aggiungono gli identificatori di tutte le entità che essa collega (già visto)
  - la chiave è composta dalle chiavi delle entità collegate

# Traduzione canonica Entità ed Associazioni molti a molti



# traduzione canonica: es.



# traduzione canonica (standard): es.

```
CREATE TABLE STUDENTE (MATR... NOT NULL,  
..., NOME... , PRIMARY KEY (MATR));
```

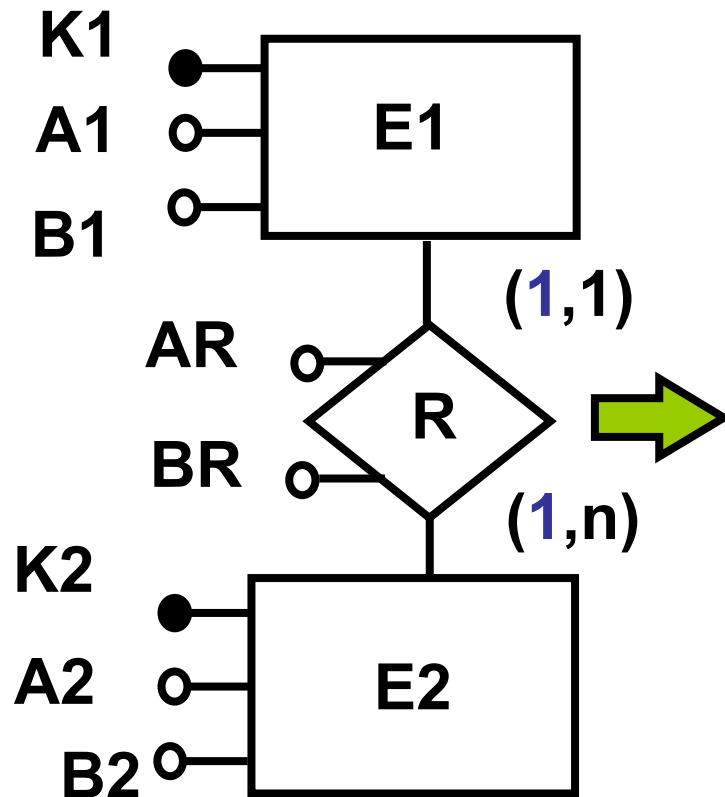
```
CREATE TABLE CORSO (CODICE... NOT NULL,  
DENOM ... , PRIMARY KEY (CODICE));
```

```
CREATE TABLE PIANO_ST (MATR... NOT NULL,  
CODICE... NOT NULL, ANNO...  
PRIMARY KEY (MATR, CODICE),  
FOREIGN KEY (MATR) REFERENCES STUDENTE  
FOREIGN KEY (CODICE) REFERENCES CORSO);
```

# altre traduzioni

- La traduzione canonica è sempre possibile ed è l'unica possibilità per le associazioni  $N$  a  $M$
- Altre forme di traduzione delle associazioni sono possibili per altri casi di cardinalità (1 a 1, 1 a  $N$ )
- Le altre forme di traduzione fondono in una stessa relazione entità e associazioni

# Associazione binaria 1 a N



- traduzione standard:

E1 (K1, A1, B1)

E2 (K2, A2, B2)

R (K1,K2, AR, BR)

# associazione binaria 1 a N

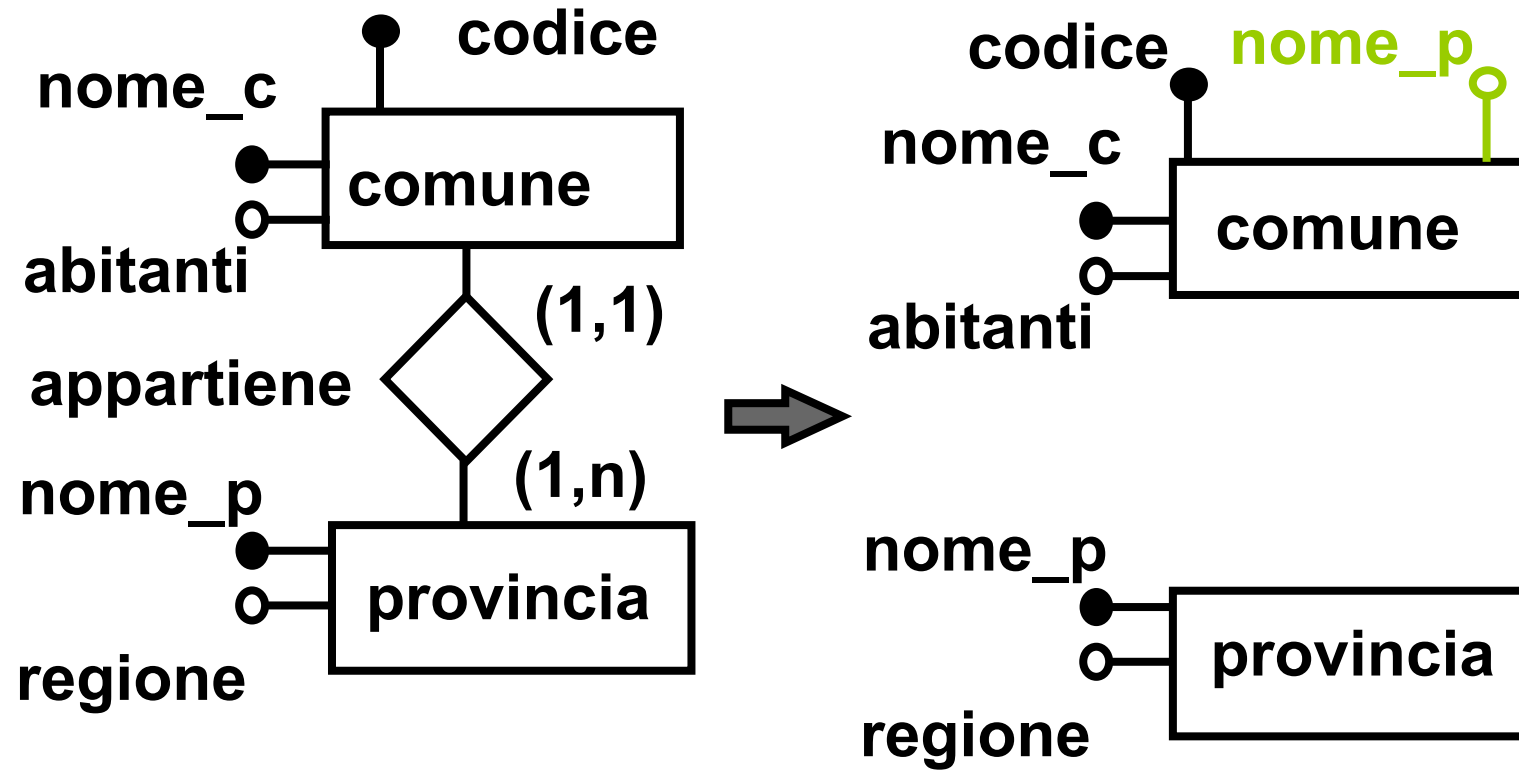
- Se E1 partecipa con cardinalità (1,1) può essere fusa con l'associazione, ottenendo una soluzione a due relazioni:

$E1(\underline{K1}, A1, B1, K2, AR, BR)$

$E2(\underline{K2}, A2, B2)$

- Se E1 partecipa con cardinalità (0,1) la soluzione a due relazioni ha valori nulli in K2, AR, BR per le istanze di E1 che non partecipano all'associazione

# ass. binaria 1 a N es.



(senza attributi sull'associazione)



ass. binaria 1 a N es.

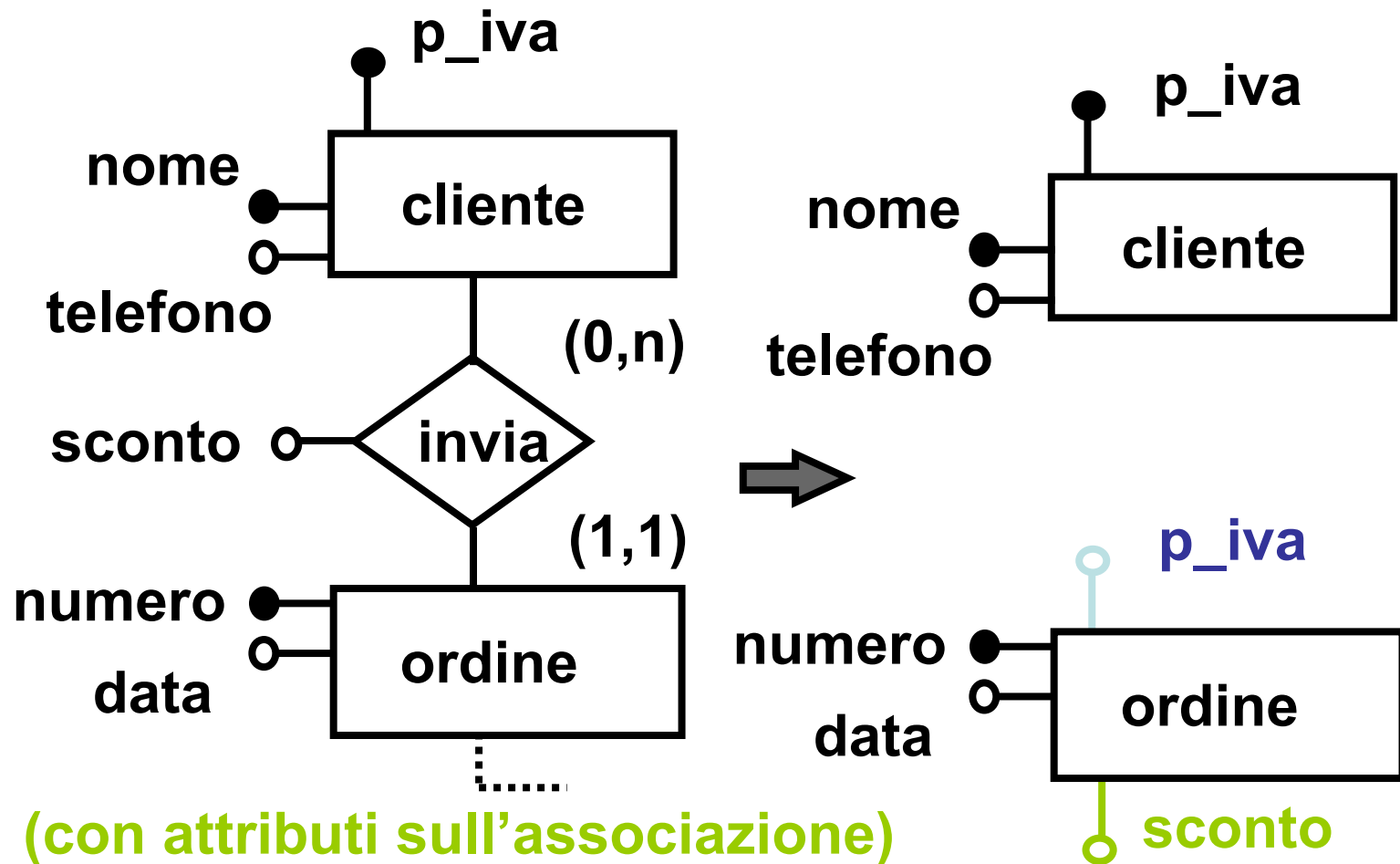
**CREATE TABLE PROVINCIA**

**(NOME\_P ... NOT NULL,  
REGIONE ... PRIMARY KEY (NOME\_P));**

**CREATE TABLE COMUNE**

**(CODICE ... NOT NULL, NOME\_C ...  
ABITANTI ..., NOME\_P ... NOT NULL  
PRIMARY KEY (CODICE)  
FOREIGN KEY NOME\_P  
REFERENCES PROVINCIA);**

# ass. binaria 1 a N es.



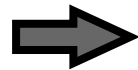
# ass. binaria 1 a N es.

traduzione con due relazioni:

```
CREATE TABLE CLIENTE (P_IVA..... NOT NULL,  
NOME ...,TELEFONO ..., PRIMARY KEY (P_IVA));
```

```
CREATE TABLE ORDINE (NUMERO ... NOT NULL,  
DATA ... P_IVA ... NOT NULL, SCONTO ...,  
PRIMARY KEY (NUMERO)  
FOREIGN KEY P_IVA REFERENCES CLIENTE);
```

con tre relazioni:



# ass. binaria 1 a N es.

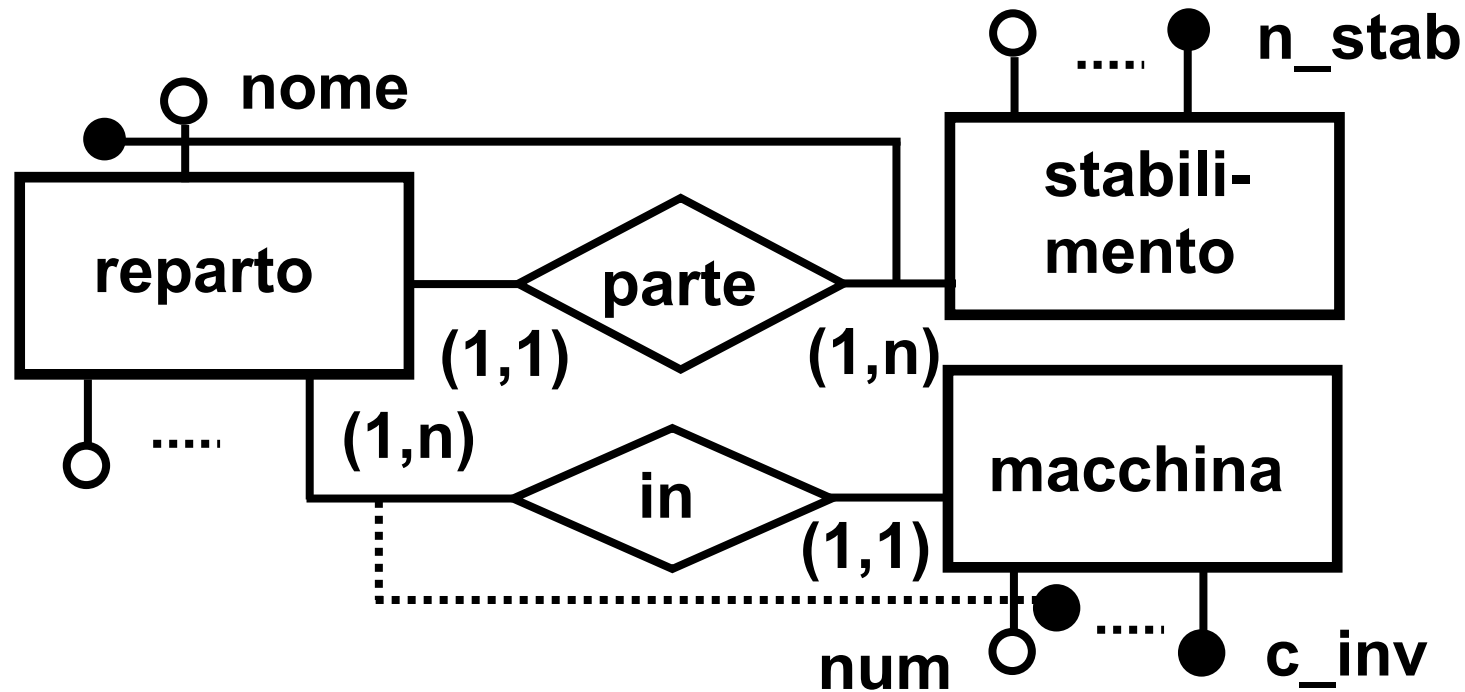
```
CREATE TABLE CLIENTE (P_IVA..... NOT NULL,  
NOME ...,TELEFONO ..., PRIMARY KEY (P_IVA));
```

```
CREATE TABLE ORDINE (NUMERO ... NOT NULL,  
DATA ... PRIMARY KEY (NUMERO));
```

```
CREATE TABLE SCRIVE  
(P_IVA ... NOT NULL, NUMERO ... NOT NULL,  
SCONTO ..., PRIMARY KEY (NUMERO)  
FOREIGN KEY P_IVA REFERENCES CLIENTE  
FOREIGN KEY NUMERO REFERENCES  
ORDINE);
```

# ass. binaria 1 a N es.

Con identificazione esterna



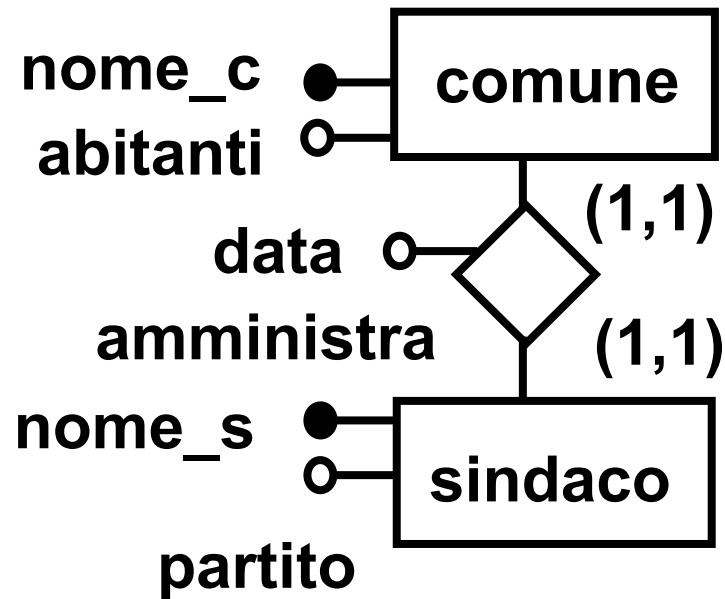
ass. binaria 1 a N es.

**STABILIMENTO (N\_STAB .....);**

**REPARTO (NOME, N\_STAB, .....);**

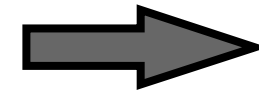
**MACCHINA (NUM, NOME, N\_STAB);**

# Associazione binaria 1 a 1



- traduzione con una relazione:

**E12 (K1, A1, B1,  
K2, A2, B2,  
AR, BR)**



# associazione binaria 1 a 1

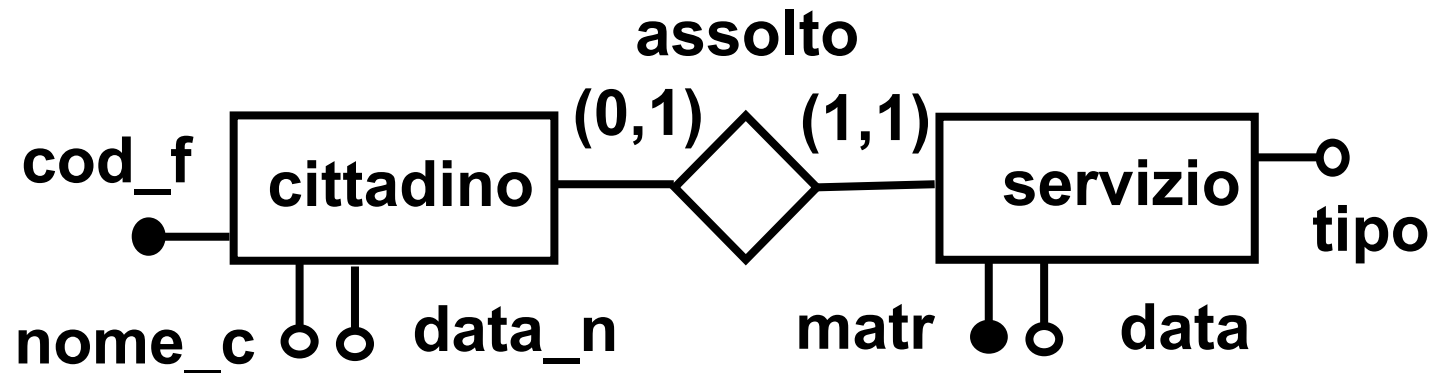
```
CREATE TABLE AMMINISTRAZIONE  
(NOME_C ... NOT NULL,  ABITANTI ...,  
  NOME_S ... NOT NULL UNIQUE,  
  INDIRIZZO ...,  DATA  
PRIMARY KEY (NOME_C));
```

se le cardinalità minime sono entrambe 1 la chiave può essere indifferentemente K1 o K2 si sceglierà quella più significativa

# associazione binaria 1 a 1

- se la cardinalità di E2 è 0,1 e quella di E1 è 1,1 allora la **chiave sarà K2** ; E2 è l'entità con maggior numero di istanze alcune della quali non si associano, ci saranno quindi valori nulli in corrispondenza di K1, K1 in questo caso non potrebbe essere scelta
- se la cardinalità è 0,1 da entrambe le parti allora le relazioni saranno due per **l'impossibilità di assegnare la chiave** all'unica relazione a causa della presenza di valori nulli sia su K1 che su K2

# associazione binaria 1 a 1



**CITTADINO (COD\_F, NOME\_C, INDIRIZZO, DATA\_N, MATR, DATA, TIPO);**

# associazione binaria 1 a 1

- Traduzione con due relazioni
  - l'associazione può essere **compattata** con l'entità che partecipa **obbligatoriamente** (una delle due se la partecipazione è obbligatoria per entrambe) la discussione sulla chiave è analoga al caso di traduzione con una relazione

E1 (K1, A1, B1,...)

E2 (K2, A2, B2,... **K1, AR, BR**)

# associazione binaria 1 a 1

- **Traduzione con tre relazioni**
  - la chiave della relazione che traduce l'associazione può essere indifferentemente K1 o K2, non ci sono problemi di valori nulli

E1 (K1, A1, B1,...)

E2 (K2, A2, B2,...)

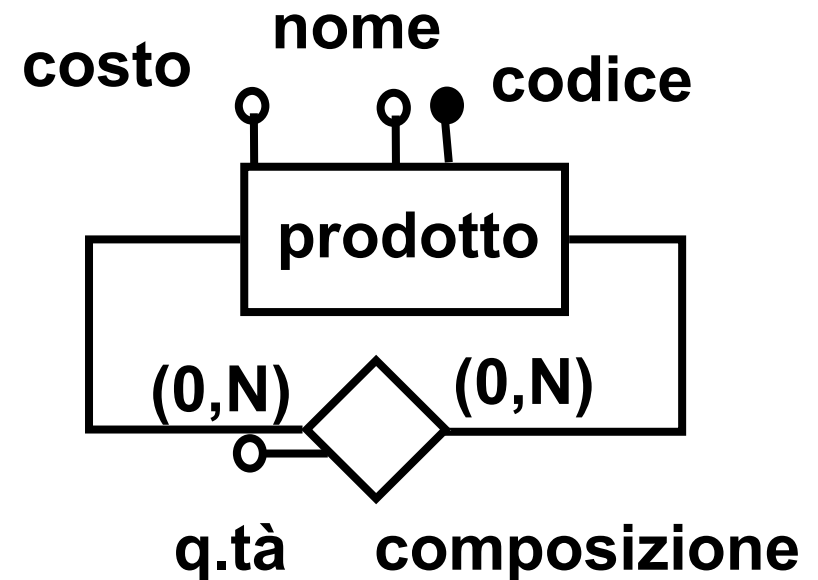
R (K1, K2, AR, BR,...)

# Necessità della ridenominazione nel caso di relazioni ricorsive

Prodotto(Codice, Nome, Costo)

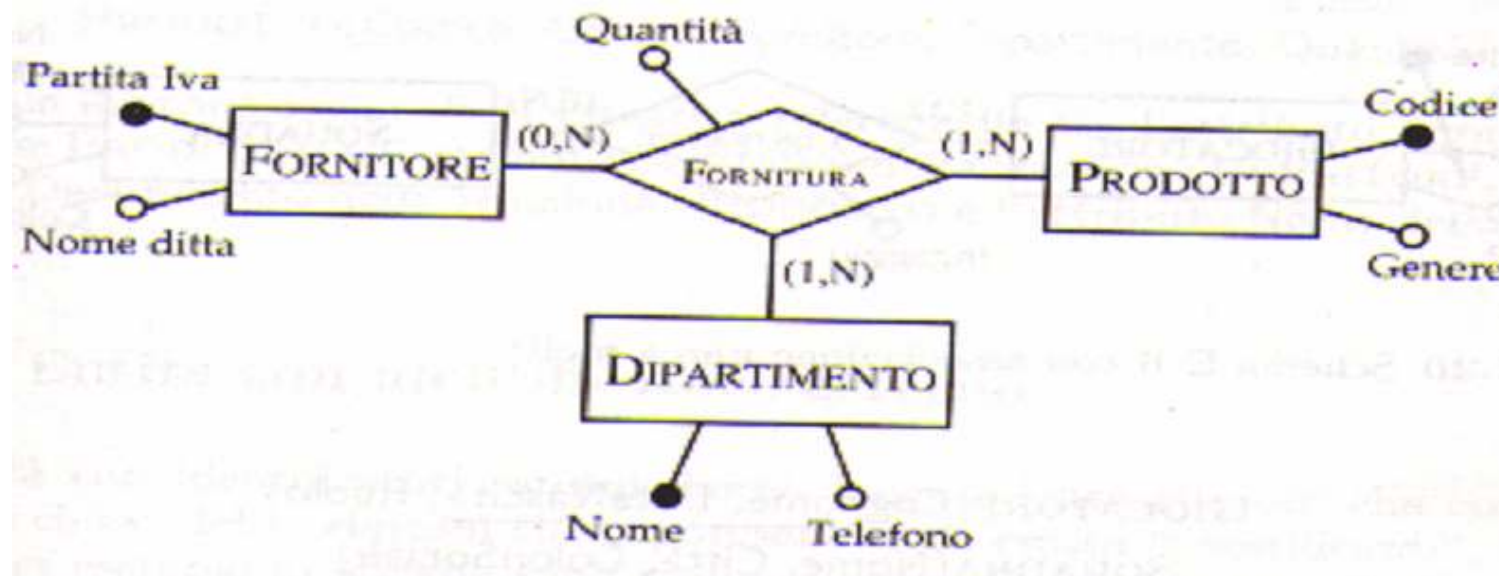
Composizione(Composto, Componente, Quantita')

- Esiste un vincolo referenziale tra Composto, Componente e l'attributo Codice di Prodotto.



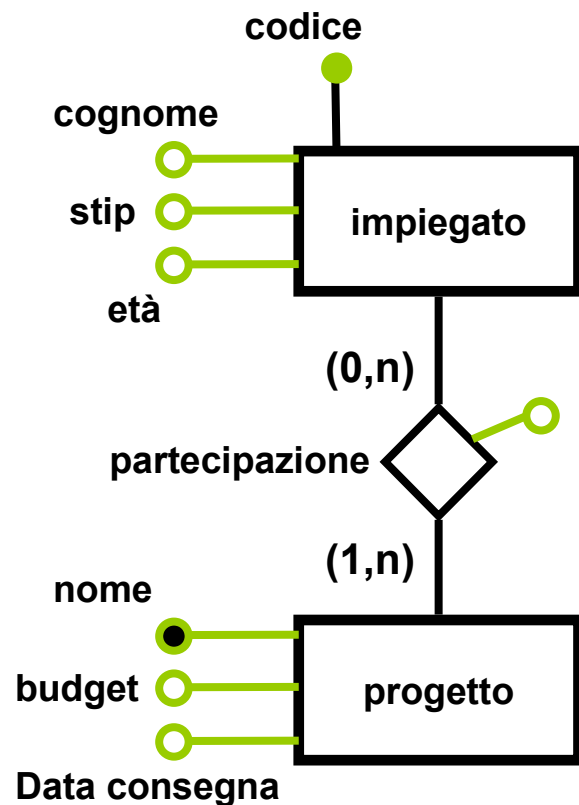
# Associazioni con più entità

- Fornitore(PartitaIVA, NomeDitta)
- Prodotto(Codice, Genere),
- Dipartimento(Nome, Telefono)
- Fornitura(Fornitore, Prodotto, Dipartimento, Quantità)



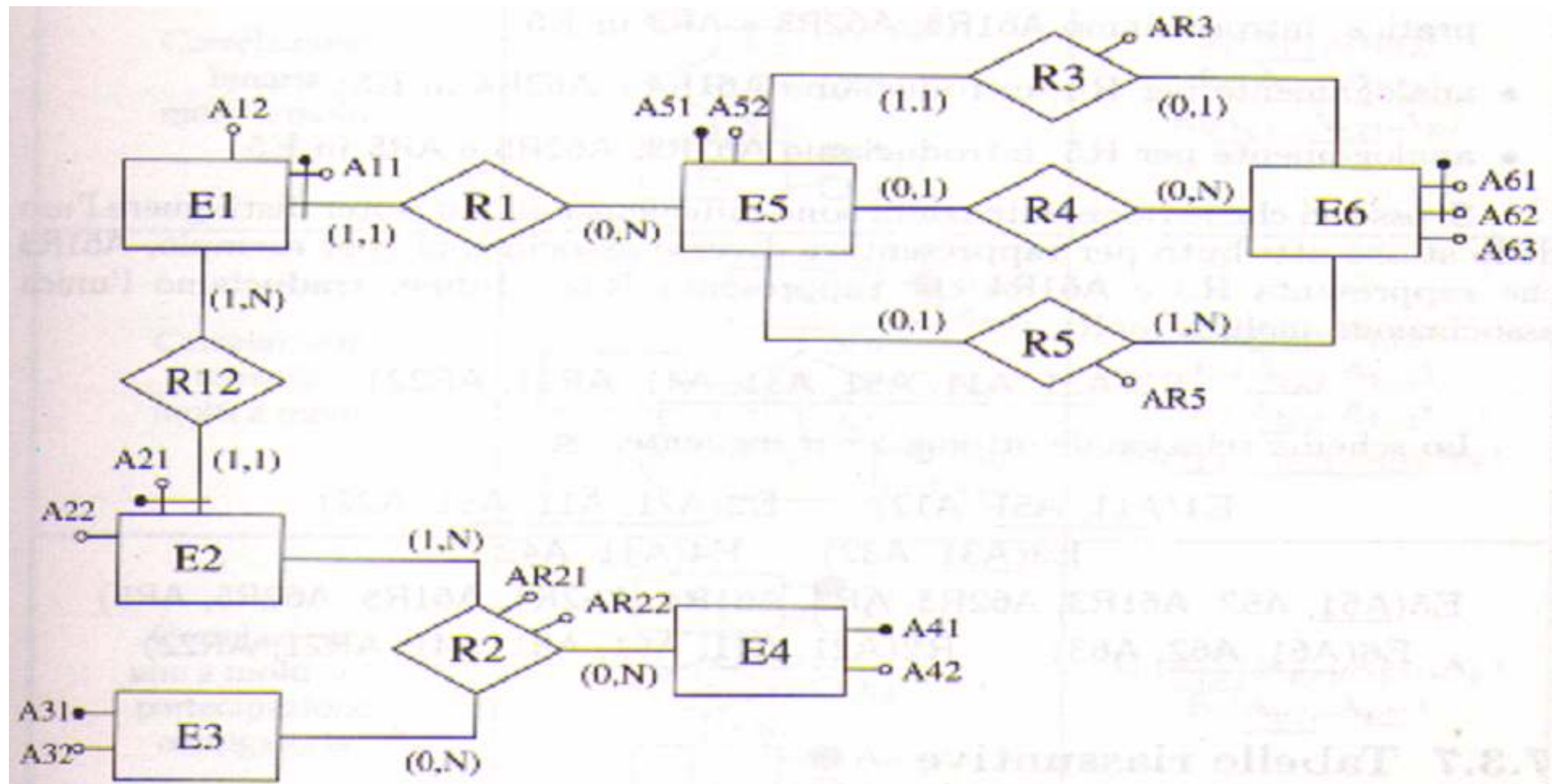
# Rappresentazione Grafica delle Traduzioni

- Permette di evidenziare i vincoli referenziali



Impiegato(Matricola, Cognome, Stipendio)  
Progetto(Codice, Nome, Budget)  
Partecipazione(Matricola, Codice, Data inizio)

# Traduzione di schemi Complessi



# Traduzione delle entita'

- *Con identificatore interno :*  
E3(A31,A32) E4(A41,A42)  
E5(A51,A52) E6(A61,A62,A63)
- *Con identificatore esterno :*  
E1(A11,A51,A12)  
E2(A21,A11,A51,A22)

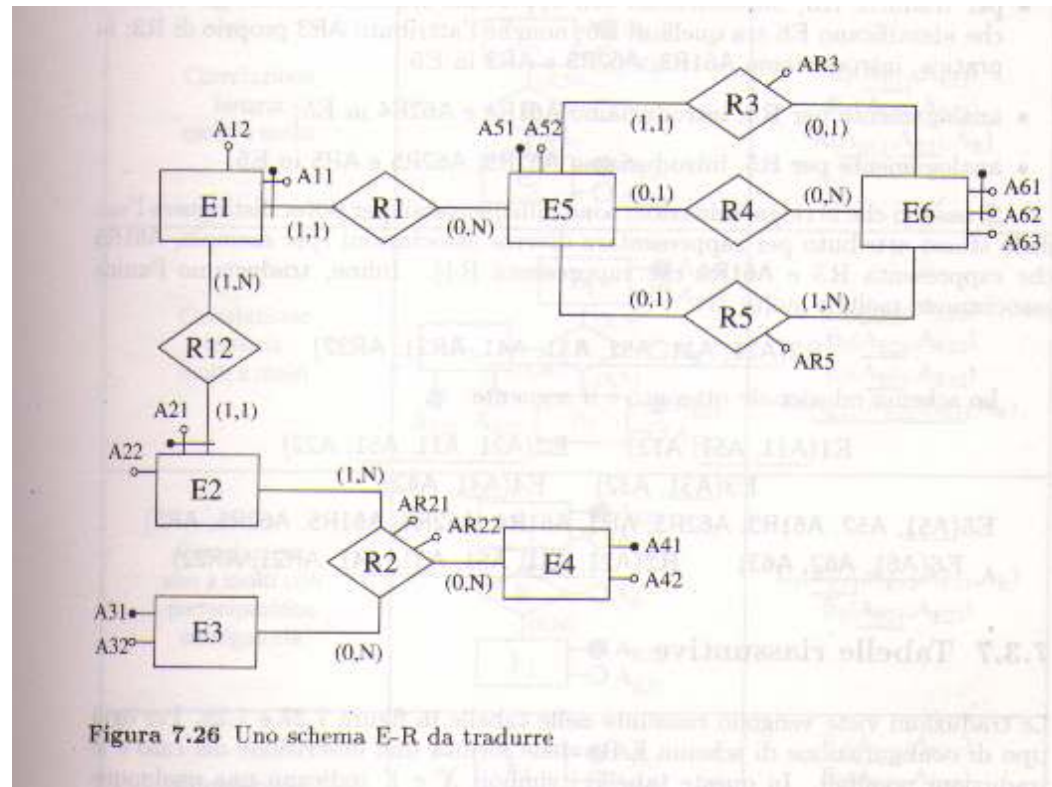


Figura 7.26 Uno schema E-R da tradurre

# Traduzioni delle Associazioni

- Le relazioni R1 ed R12 sono state tradotte con l'identificazione esterna di E1 ed E2.
- Per tradurre R3 scarichiamo su E5 con opportuni rinominamenti gli attributi che individuano E6 nonché l'attributo AR3 di R3
- Facciamo lo stesso per R4 ed R5 scaricando tutto su E5 (rinominando R3 come A61R3 ed R4 con A61R4)
- Infine l'unica associazione R2 molti a molti viene tradotta R2(A21,A11,A51,A31,A41,AR21,AR22)

# Schema Finale

E1(A11,A51,A12)

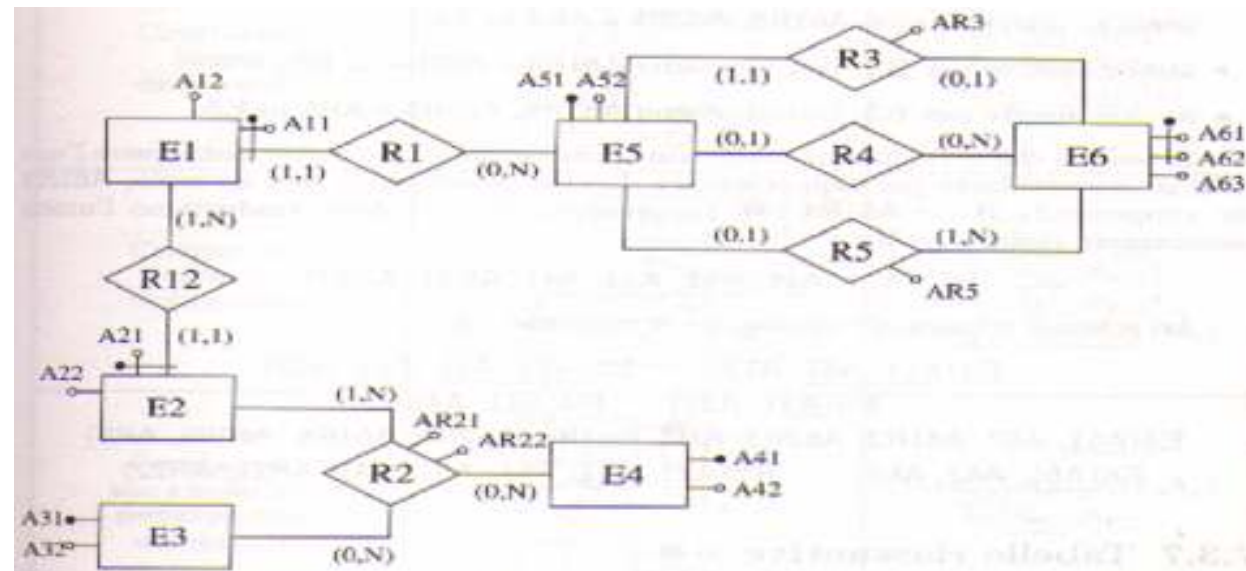
E2(A21,A11,A51,A22)

E3(A31,A32)

E4(A41,A42)

E5(A51,A52,A61R3,A62R3,AR3,A61R4, A62R4,A61R5,A62R5,AR5)

E6(A61,A62,A63) R2(A21,A11,A51,A31,A41,AR21,AR22)



# Progetto di basi di dati

Laboratorio di diagnosi mediche

La progettazione delle basi di dati

# Descrizione e specifiche

Si vuole realizzare il progetto della base di dati di laboratorio di diagnosi medica, partendo da un insieme di requisiti. Le fasi da svolgere vanno dall'analisi dei requisiti, alle varie fasi dell'analisi fino all'implementazione delle operazioni previste. Durante il progetto è necessario produrre un insieme di documenti, che costituiscono appunto la *documentazione* del progetto:

- Analisi dei requisiti;
- Lo schema concettuale, tramite il modello E-R, presentato a diversi gradi di raffinamento;
- Una descrizione delle operazioni previste e le relative tavole di carico;
- Lo schema ottenuto per ristrutturazione dalla prima fase della progettazione logica. Lo schema logico finale.
- Un listato delle interrogazioni e delle istruzioni (aggiornamenti, inserimenti, cancellazioni) SQL relative alle operazioni previste;
- Contenuto di test della base di dati e nella stampa dei risultati delle interrogazioni su tali dati.

# Specifiche sui dati

Si vuole progettare il sistema informativo di un laboratorio di diagnosi medica. Diversi tipi di persone sono coinvolte nel laboratorio: medici, assistenti, pazienti.

Per i pazienti, rappresentiamo alcuni dati anagrafici, quali il nome, il cognome, l'età, l'indirizzo, il telefono ed il codice fiscale (che li identifica).

Per i medici e gli assistenti, oltre ai dati anagrafici abbiamo un codice interno che li identifica. I clienti del laboratorio (circa 100000) hanno bisogno di visite mediche e/o analisi che vanno riservate in anticipo, fissando data e ora.

La storia delle analisi e delle visite degli ultimi 12 mesi deve essere memorizzata nel sistema. Le prestazioni offerte dal laboratorio appartengono a varie tipologie, identificate da un codice e caratterizzate da una descrizione.

Ogni tipo di prestazione ha un costo che dipende dal tipo di paziente.

# Specifiche sui dati

Ogni dottore (150) può effettuare solo determinati tipi di analisi e visite. Ogni assistente (circa 300) può effettuare solo determinati tipi di analisi.

Le analisi (circa 200 al giorno) e le visite (circa 100 al giorno) sono effettuate in apposite stanze. Ogni prestazione offerta ha un esito, caratterizzata da una descrizione, una data ed un prezzo. L'esito di ogni analisi va approvato con il nome di un dottore.

Gli esiti delle analisi e delle visite devono essere memorizzate in una cartella del paziente, che registra la storia delle ultime 30 visite e/o analisi. Di ogni cartella va memorizzata la data di apertura.

Le prestazioni possono essere effettuate o come esito di altre prestazioni o indipendentemente.

Per gli assistenti, che sono dipendenti dal laboratorio, vogliamo rappresentare il loro livello e lo stipendio. Per i dottori, che sono considerati consulenti del laboratorio, rappresentiamo un valore percentuale per il calcolo delle parcelle, la specializzazione, l'ente di appartenenza e la disponibilità settimanale.

Il laboratorio rilascia delle fatture per gli esiti di analisi e visite. Una fattura può riferirsi a diverse prestazioni di uno stesso cliente.