

Progettazione logica

↓
Requisiti della base di dati

**Progettazione
concettuale**

↓
Schema concettuale

**Progettazione
logica**

↓
Schema logico

**Progettazione
fisica**

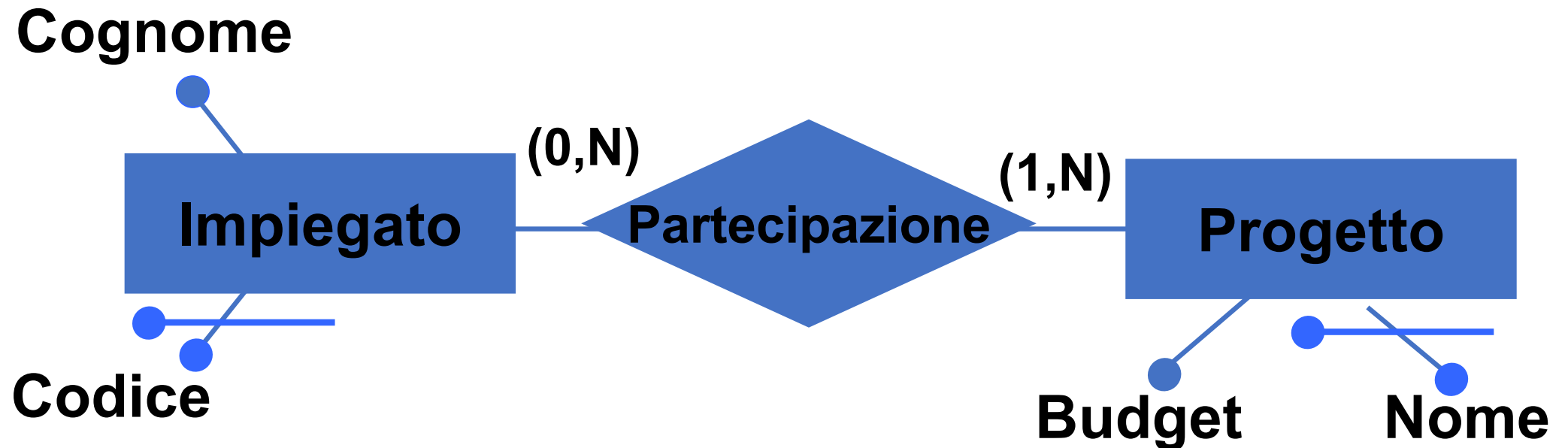
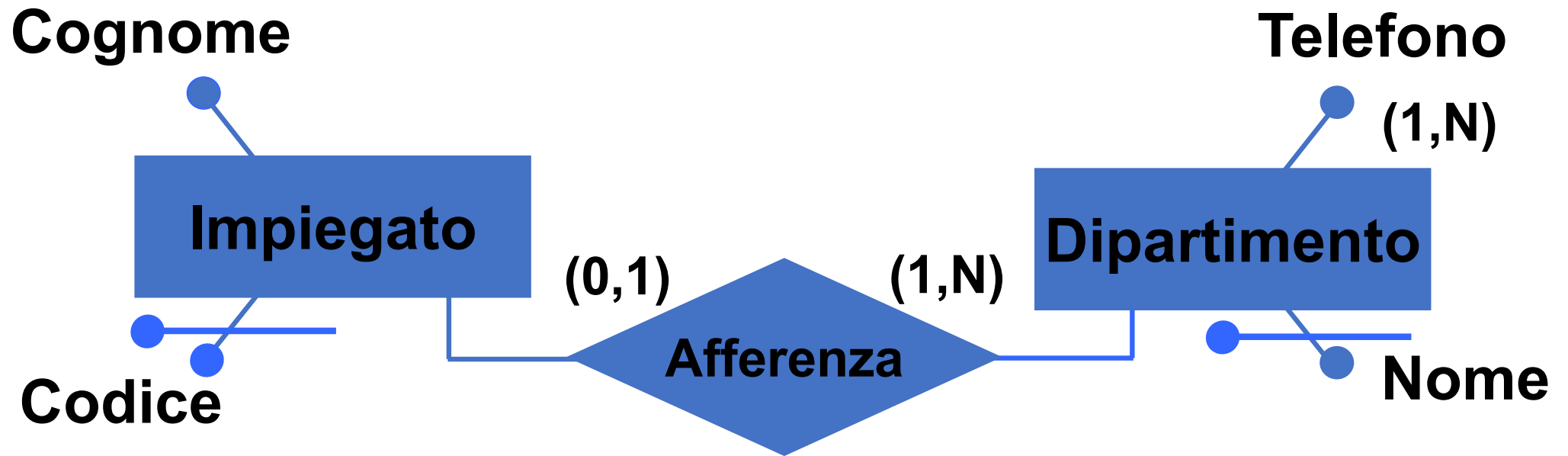
↓
Schema fisico

Obiettivo della progettazione logica

- "tradurre" lo schema concettuale in uno schema logico che rappresenti gli stessi dati in maniera corretta ed efficiente
 - Molti aspetti sono semplici e immediati
 - Altri richiedono attenzione (anche alle prestazioni)

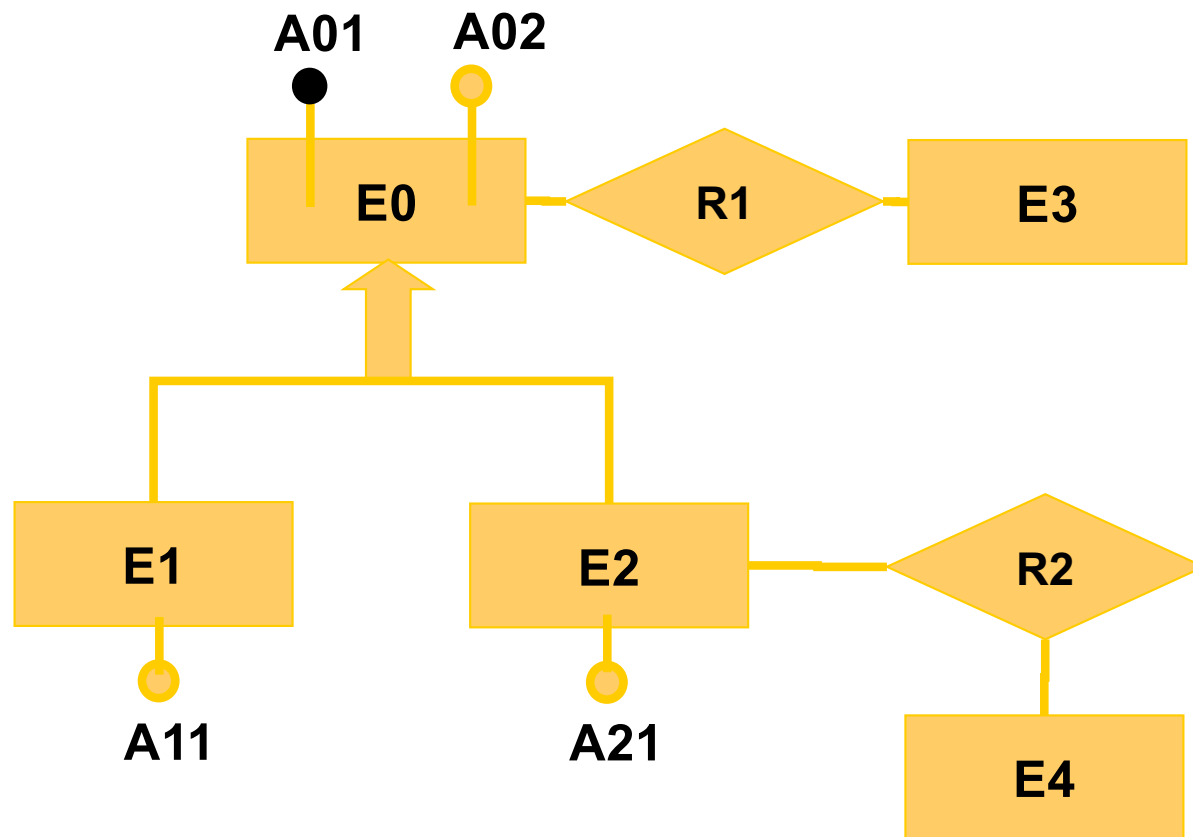
Esempio semplice ...

- ... alla lavagna traduzione
 - due entità e una relationship molti a molti
 - due entità e una relationship uno a molti



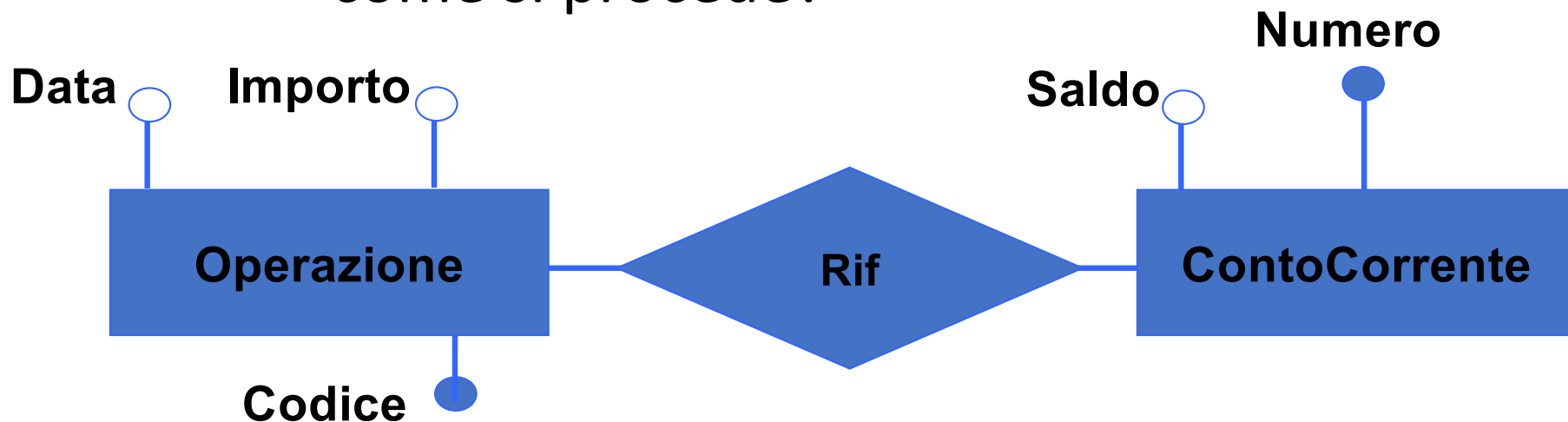
Aspetto complesso

- Per le generalizzazioni, non esiste una traduzione immediata e naturale



Aspetto più specifico

- Ridondanze, sono utili o dannose?
 - concettualmente, ne vogliamo poche, ma qualcuna sì
 - qui il ragionamento è diverso, perché?
 - come si procede?

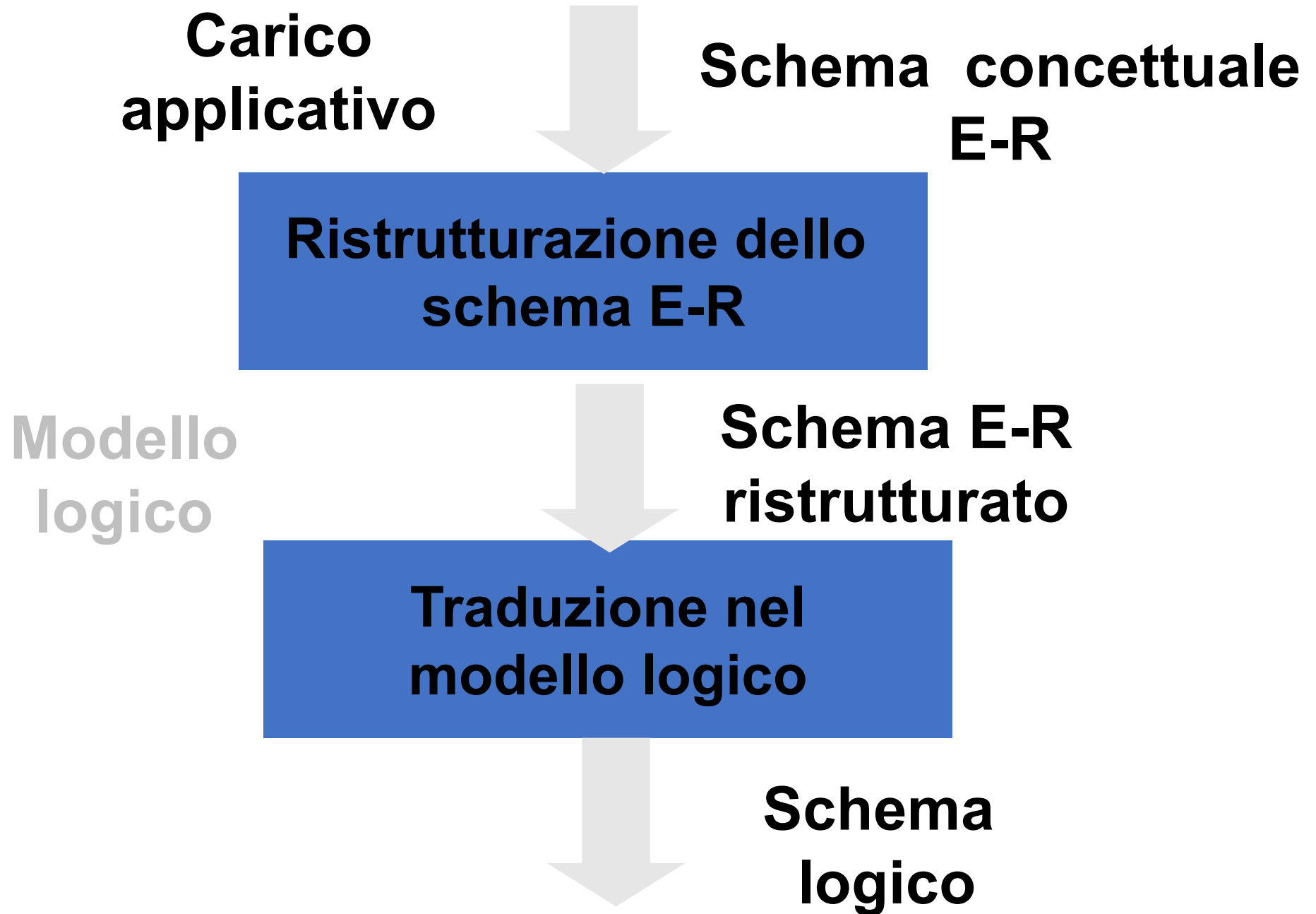


Intuitivamente

- La ridondanza conviene se ...
 - "gli aggiornamenti costano meno delle interrogazioni"
- Dobbiamo conoscere le operazioni!

Input e output della progettazione logica

- **Ingresso:**
 - schema concettuale (e documentazione associata)
 - informazioni sul carico applicativo (le operazioni)
 - modello logico (per noi è fisso, relazionale)
- **Uscita:**
 - schema logico (e documentazione associata)



Ristrutturazione schema E-R

- Motivazioni:
 - semplificare la traduzione
 - "ottimizzare" le prestazioni
- Osservazione:
 - uno schema E-R ristrutturato non è (più) uno schema concettuale nel senso stretto del termine

Prestazioni?

- Per ottimizzare il risultato abbiamo bisogno di analizzare le prestazioni a questo livello
- Ma:
 - le prestazioni non sono valutabili con precisione su uno schema concettuale!

Prestazioni, approssimate

- Consideriamo:
 - “indicatori”
 - parametri che fornisco indizi sulle prestazioni
- spazio:
 - numero di occorrenze previste
- tempo:
 - numero di occorrenze (di entità e relationship) visitate durante un' operazione

Attività della ristrutturazione

- Analisi delle ridondanze
- Eliminazione delle generalizzazioni
- Partizionamento/accorpamento di entità e relationship
- Scelta degli identificatori primari

Analisi delle ridondanze

- Una ridondanza in uno schema concettuale è una informazione significativa ma derivabile da altre
- in questa fase si decide se eliminare le ridondanze eventualmente presenti o mantenerle (o anche di introdurne di nuove)

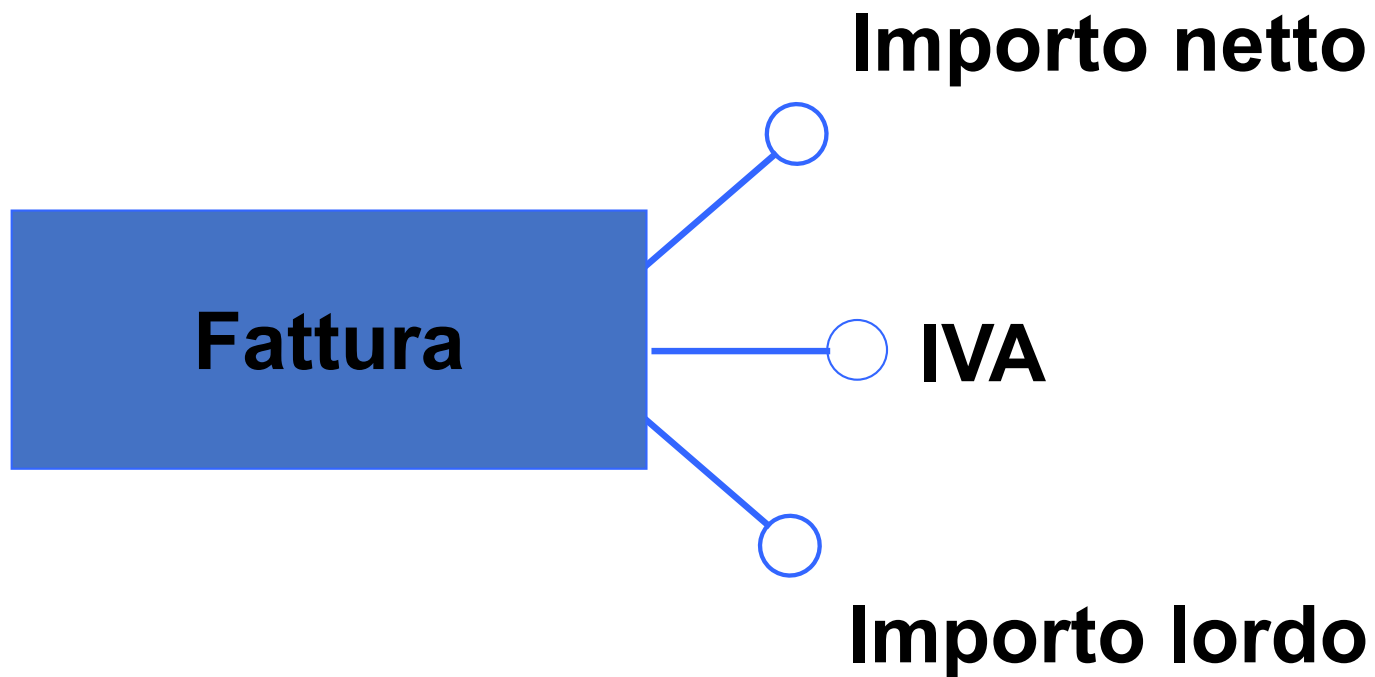
Ridondanze

- Vantaggi
 - semplificazione delle interrogazioni
- Svantaggi
 - appesantimento degli aggiornamenti
 - maggiore occupazione di spazio (trascuriamo)

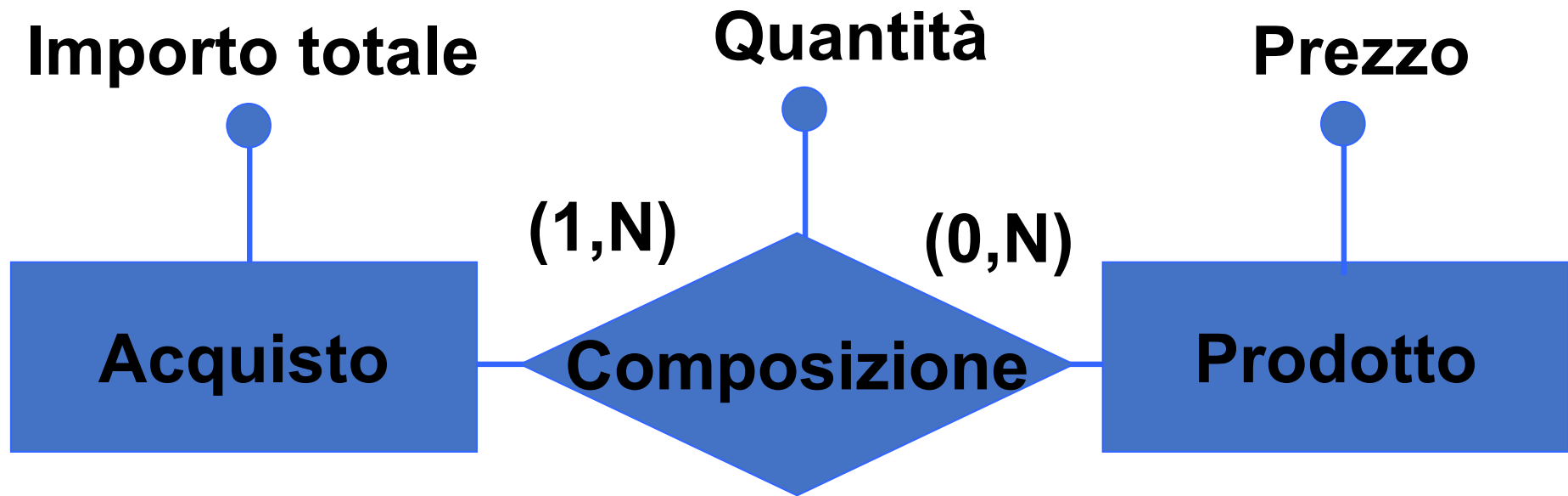
Forme di ridondanza in uno schema E-R

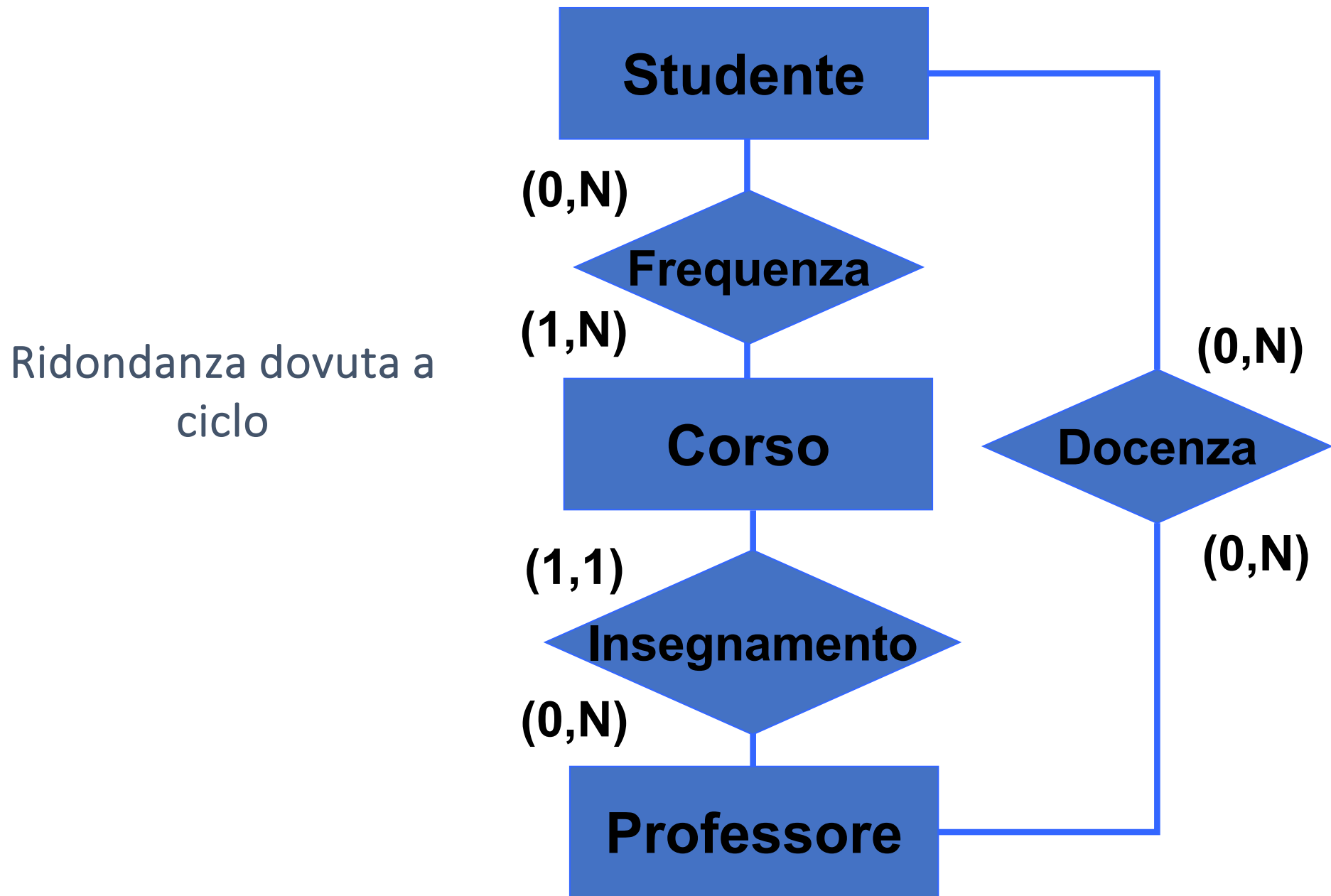
- attributi derivabili:
 - da altri attributi della stessa entità (o relationship)
 - da attributi di altre entità (o dalla partecipazione a relationship)
- relationship derivabili dalla composizione di altre (più in generale: cicli di relationship)

Attributo derivabile



Attributo derivabile da altra entità

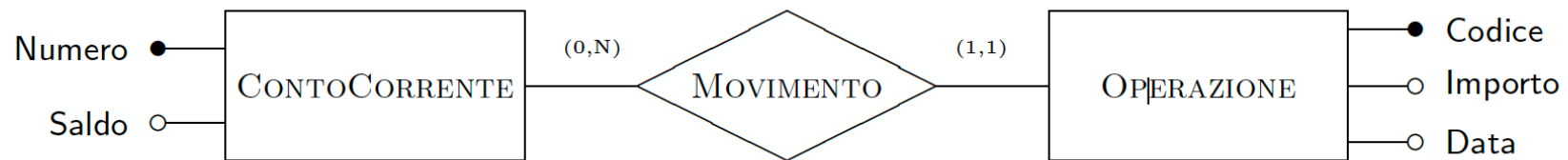




Analisi di una ridondanza

- **Esercizio 5 del 14 novembre 2001**

Domanda 5 (15%) Si consideri lo schema concettuale seguente, nel quale l'attributo **Saldo** di una occorrenza di **CONTOCORRENTE** è ottenuto come somma dei valori dell'attributo **Importo** per le occorrenze di **OPERAZIONE** ad essa correlate tramite la relationship **MOVIMENTO**.



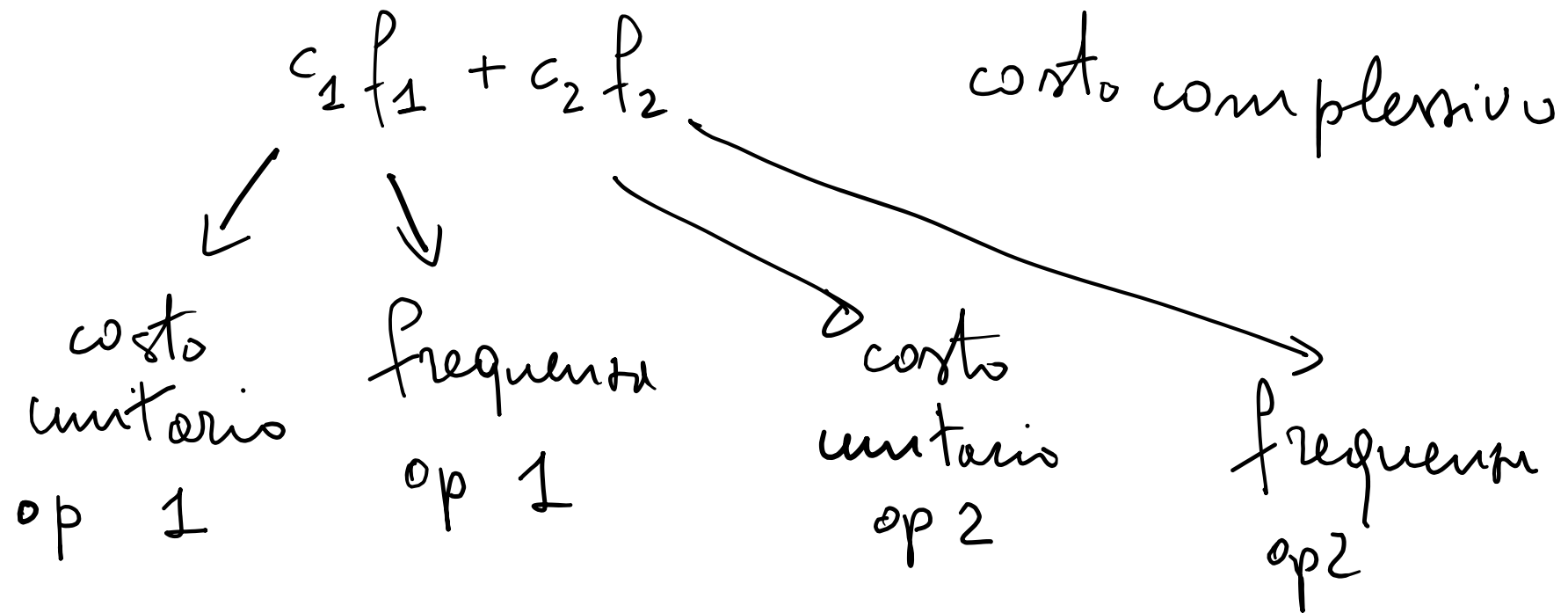
Valutare se convenga o meno mantenere la ridondanza, tenendo conto del fatto che le cardinalità delle due entità sono $L_{CC} = 2.000$ e $L_{OP} = 20.000$ e che le operazioni più importanti sono:

OP₁ scrittura di un movimento, con frequenza $f_1 = 10$

OP₂ lettura del saldo, con con frequenza $f_2 = 1000$

(alla lavagna -- possibile soluzione sul sito;

sul sito si adotta una convenzione leggermente diversa per i costi)



$$\begin{array}{c}
 c_{1R} f_1 + c_{2R} f_2 \\
 \swarrow \\
 \text{costo op1} \quad c_{1S} f_2 + c_{2S} f_2 \\
 \text{con} \\
 \text{rid} \\
 \downarrow \\
 \text{costo op1} \\
 \text{senza rid}
 \end{array}$$

COSTO TOTALE
CON RIDONDANZA

COSTO TOTALE
SENZA RIDONDANZA

	OP1	OP2
CON RIDONDANZA	1W 1R 1W 5	1R 1
SENZA RIDONDANZA	1W 2	10R 10

ammettendo che le scritture
costino "2" e le letture "1"

$$c_1 p_1 + c_2 p_2$$

$$5 \times 10 + 1 \times 1000 = 1050$$

$$2 \times 10 + 10 \times 1000 = 10020$$

CONVIENE LA
RIDONDANZA

Attività della ristrutturazione

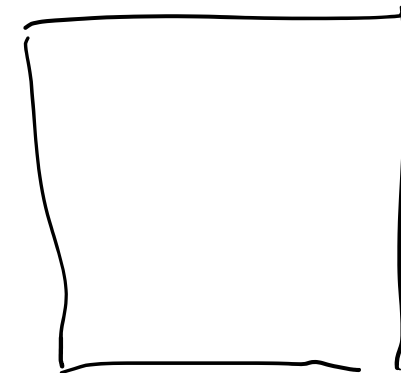
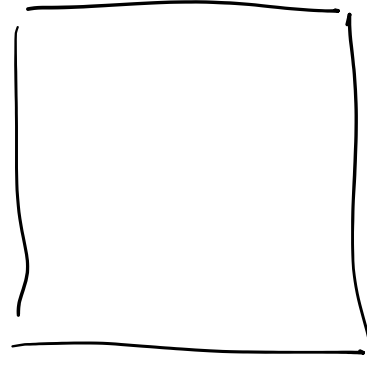
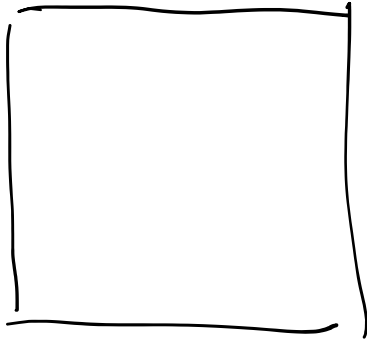
- Analisi delle ridondanze
- Eliminazione delle generalizzazioni
- Partizionamento/accorpamento di entità e relationship
- Scelta degli identificatori primari

Approccio comune

(per generalizzazioni e partizionamento)

- Intuitivamente:
 - aggregare ciò che viene acceduto insieme
 - separare ciò che viene acceduto separatamente
- Motivazione:
 - gli accessi sono ai blocchi e si risparmia se i dati visitati insieme sono concentrati in “pochi” blocchi
- Intuizione alla lavagna, proiezione su pochi attributi di una relazione su moltissimi attributi

blochi dimensione fissa (4K)



MATR	COGN	NOME

MATR	COG

MATR	NOME

MATR 2 B

COGN 50 B

NOME 50 B

1000 tuple

blocco 4K

3 attributi

in un blocco

40 tuple

la relazione

occupa

25 blocchi

2+2 attributi

in un blocco

80 tuple

ogni relazione

occupa

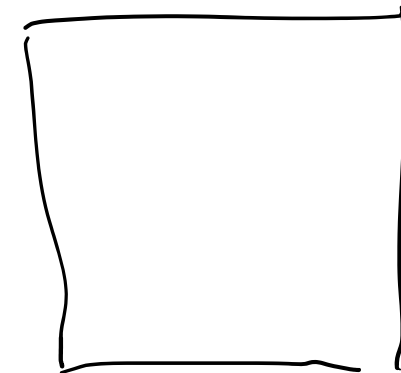
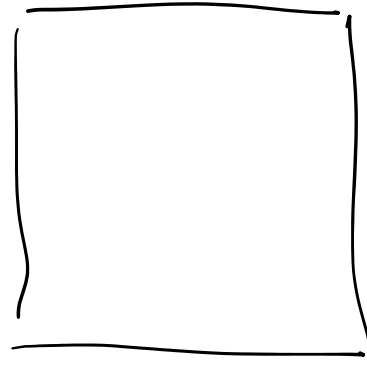
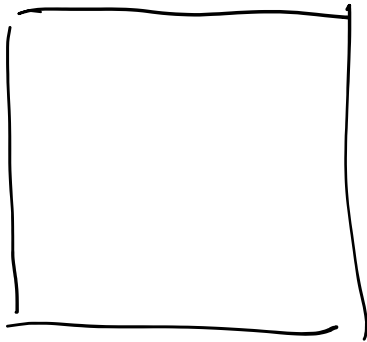
13 blocchi

totale

26 blocchi

trascuriamo MATR

blochi dimensione fissa (4K)



MATR	COGN	NOME

accanto
contextuali
a COGN
e NOME

MATR	COG	MATR	NOME

accanto
separati
e
COGN
e
NOME

la implementazione

- con una relazione conviene se accedo insieme a COG e NOME (ad esempio, COG e NOME di una persona)
- con due relazioni conviene se accedo separatamente a NOME e COG (ad esempio COG di tutte le persone)

Attività della ristrutturazione

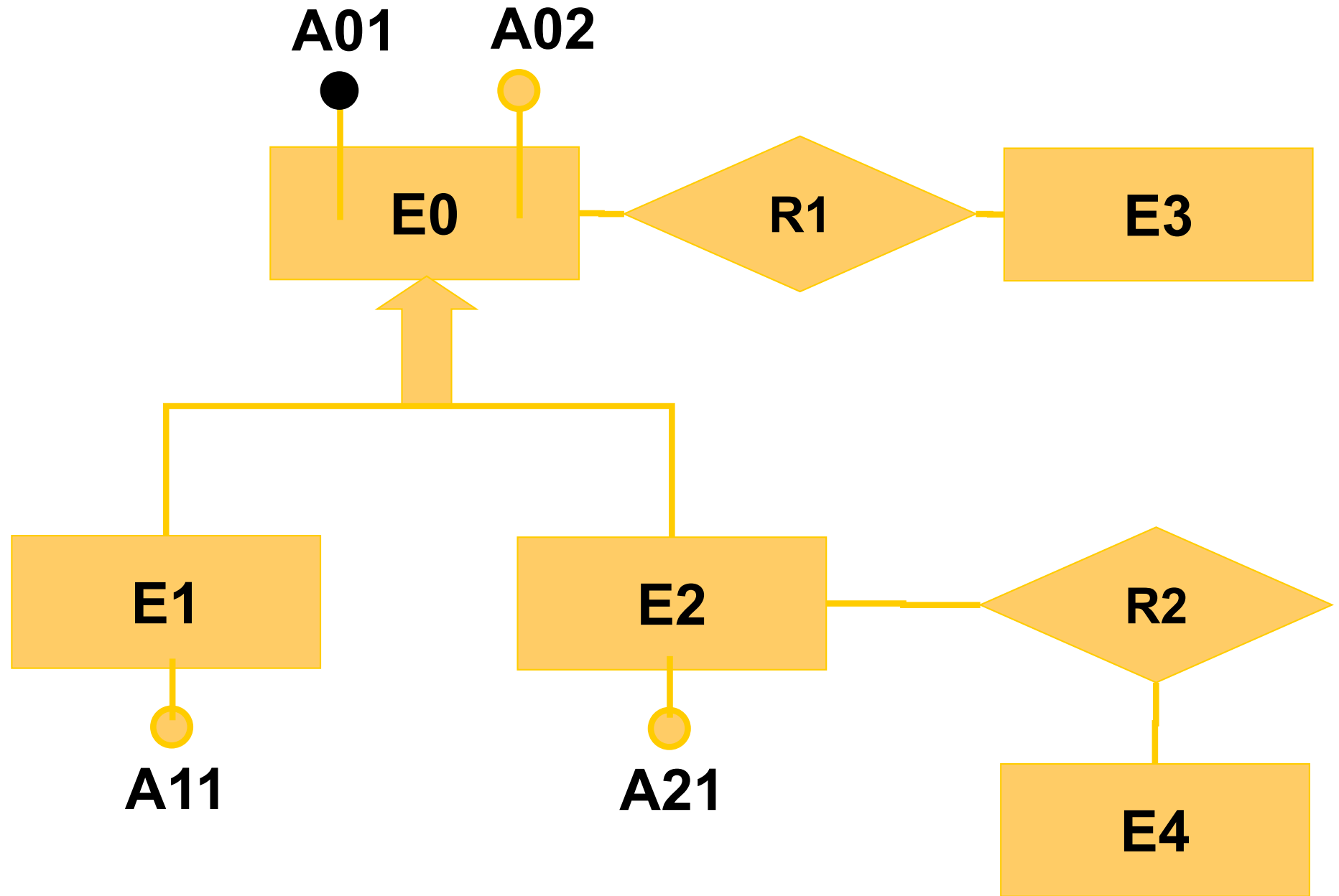
- Analisi delle ridondanze
- Eliminazione delle generalizzazioni
- Partizionamento/accorpamento di entità e relationship
- Scelta degli identificatori primari

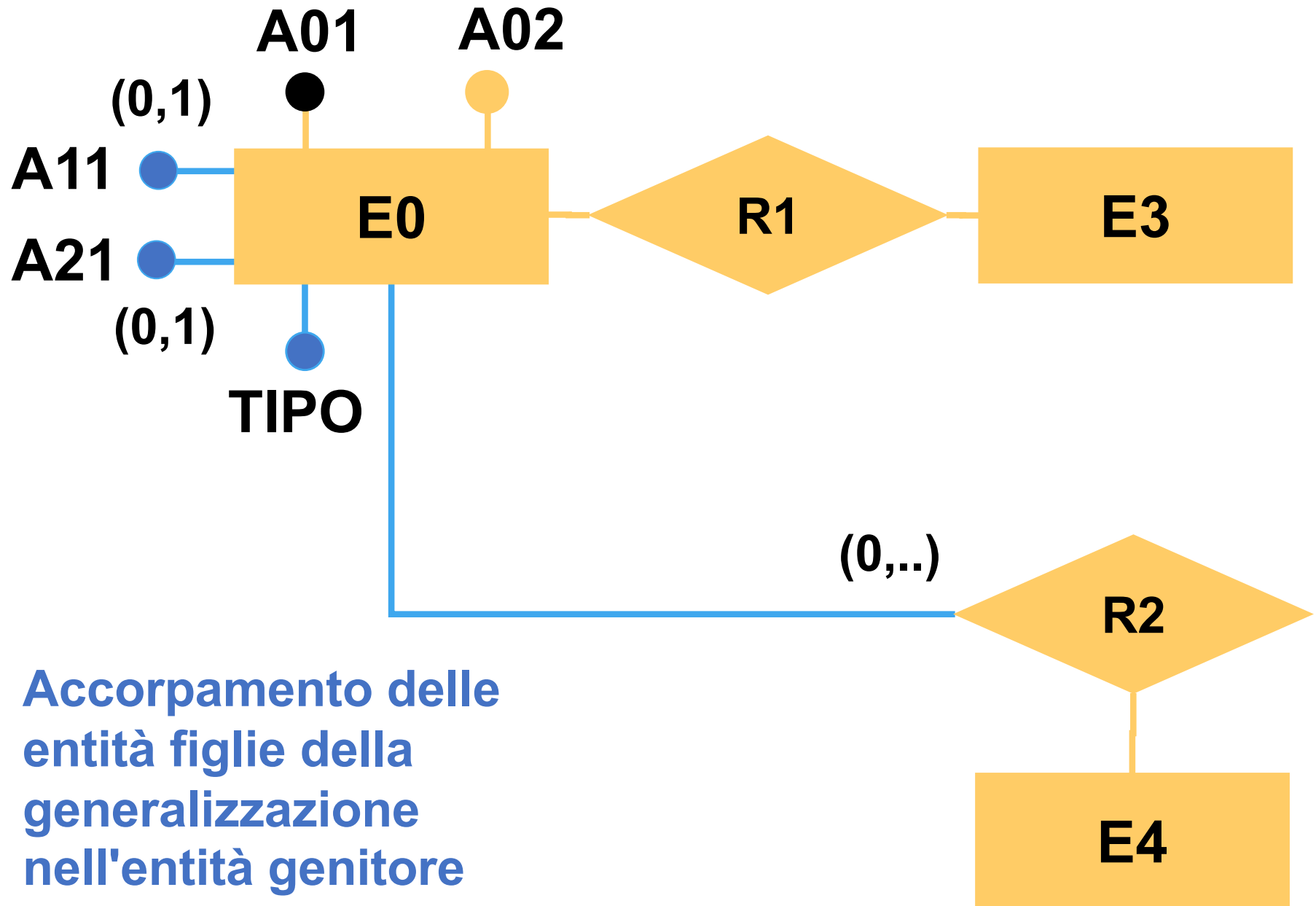
Eliminazione delle generalizzazioni

- il modello relazionale non può rappresentare direttamente le generalizzazioni
- entità e relationship sono invece direttamente rappresentabili
- si eliminano perciò le generalizzazioni, sostituendole con entità e relationship

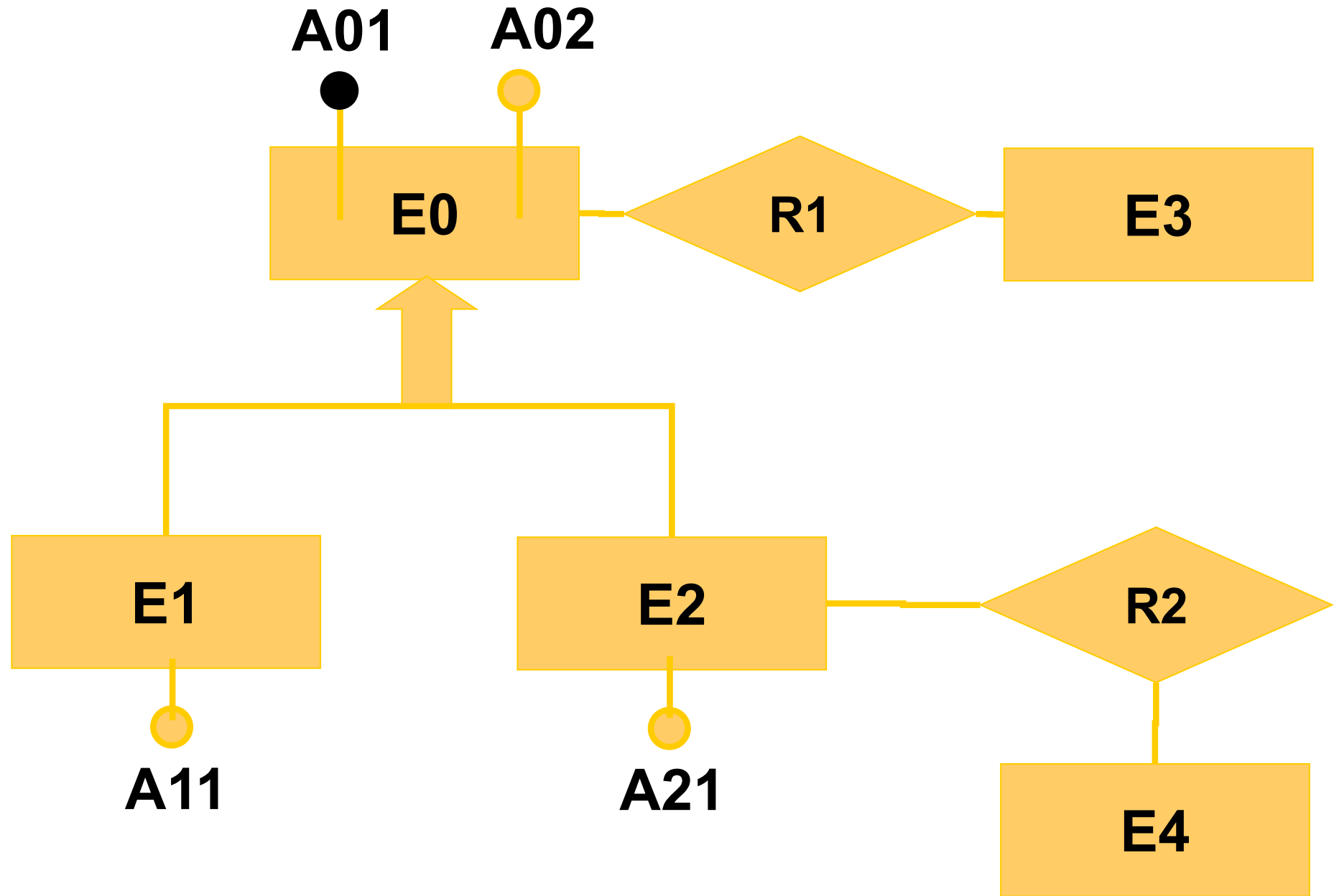
Tre possibilità

1. accorpamento delle entità figlie della generalizzazione nell'entità genitore
2. accorpamento dell'entità genitore della generalizzazione nelle entità figlie
3. sostituzione della generalizzazione con relationship

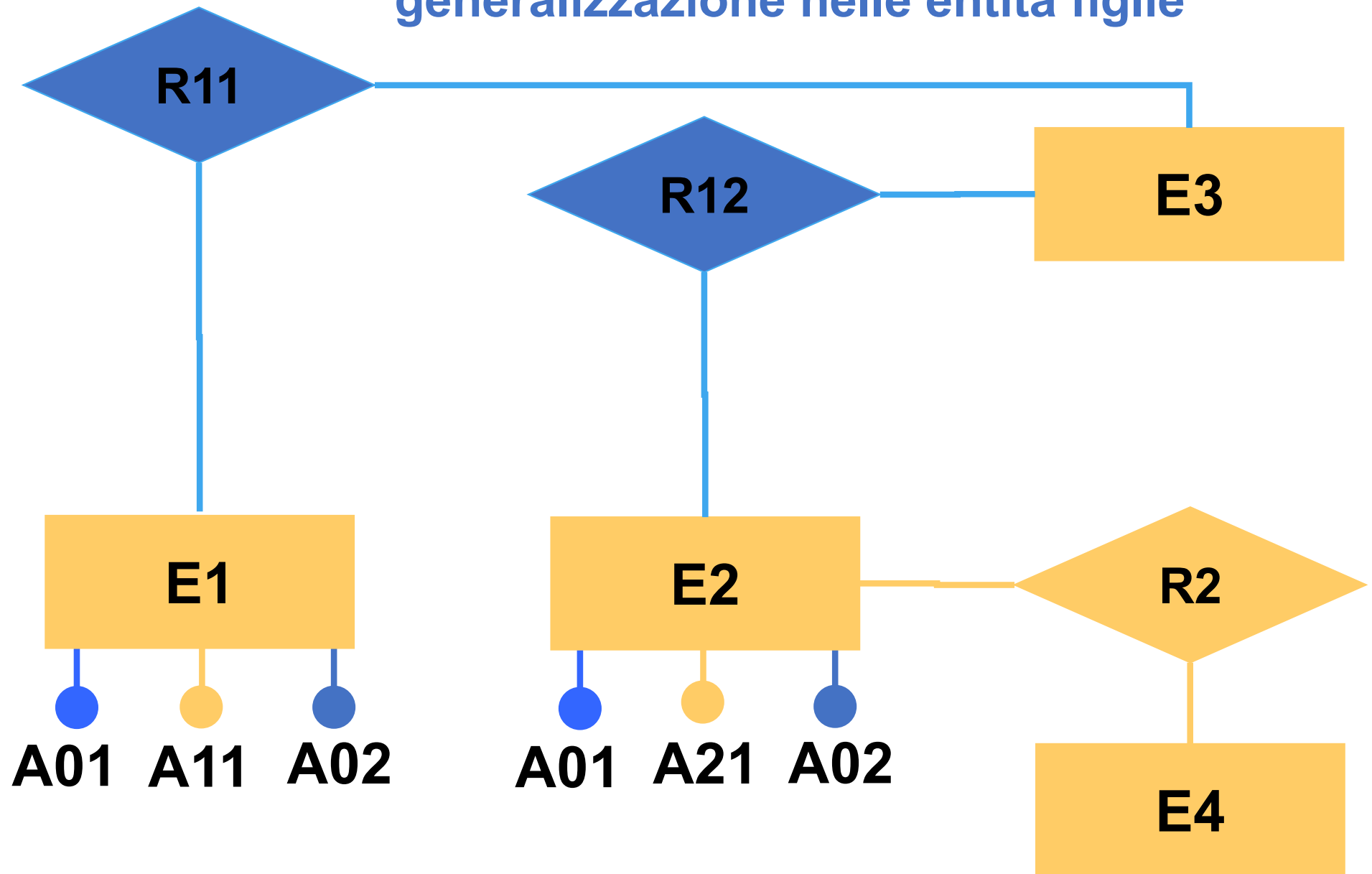


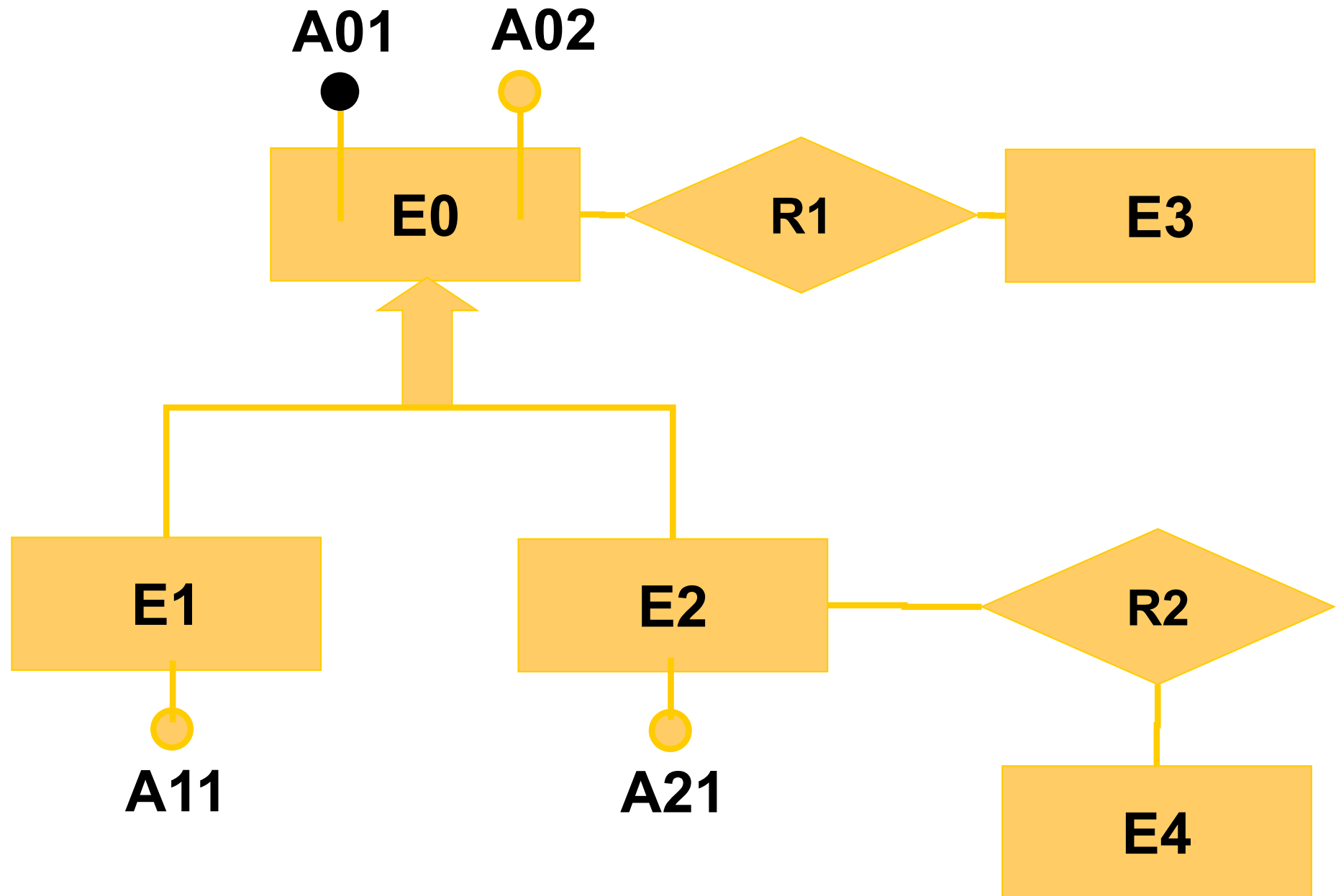


1. Accorpamento delle entità figlie della generalizzazione nell'entità genitore

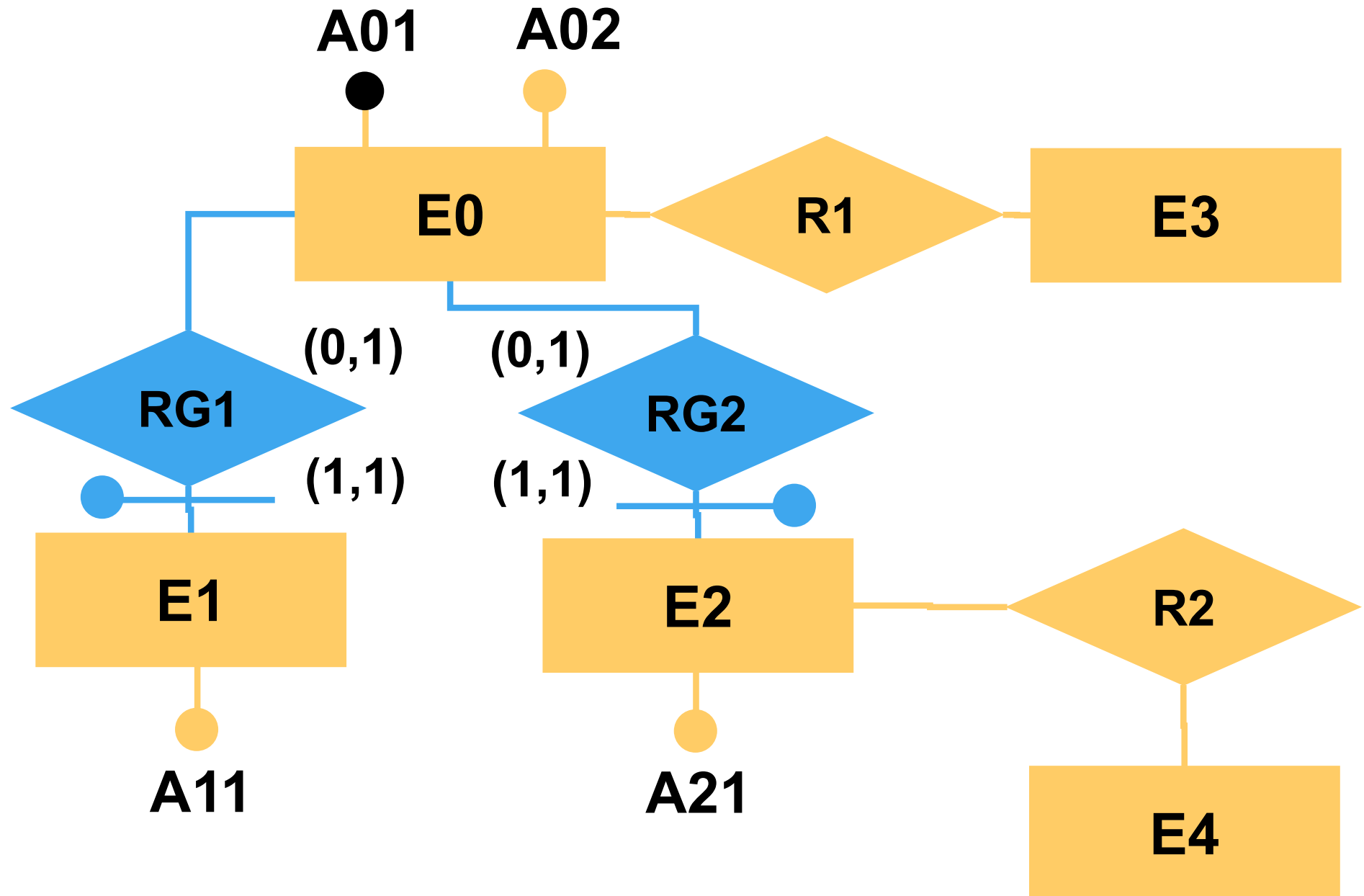


2. accorpamento dell'entità genitore della generalizzazione nelle entità figlie



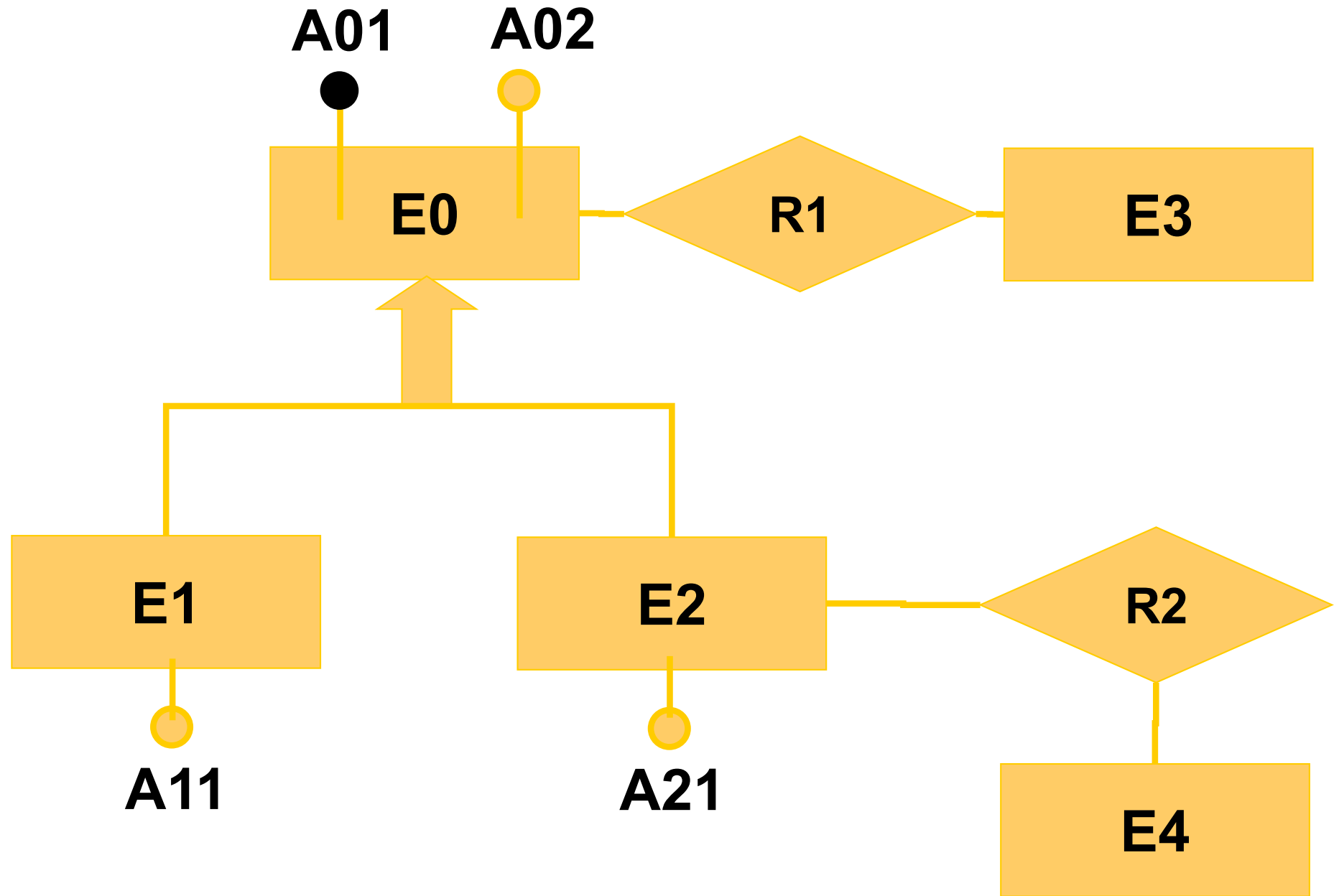


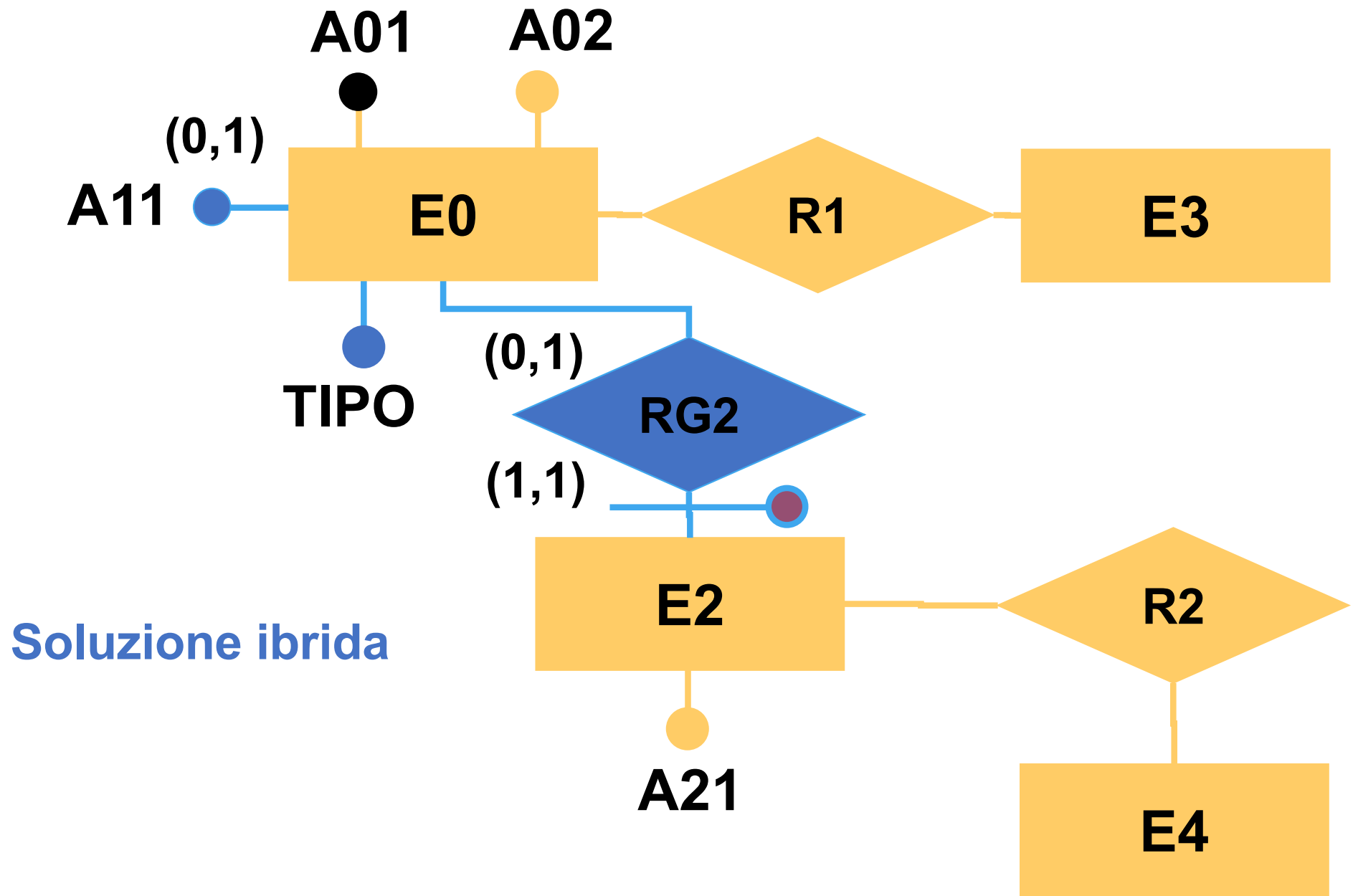
3. sostituzione della generalizzazione con relationship



- la scelta fra le alternative si può fare seguendo l'idea di mantenere insieme ciò che viene visitato insieme e separare ciò che viene acceduto separatamente
- è possibile seguire alcune semplici regole generali

1. conviene se gli accessi all'entità genitore e alle entità figlie sono contestuali
 2. conviene se gli accessi alle entità figlie sono separati gli uni dagli altri
 3. conviene se gli accessi alle entità figlie sono separati dagli accessi all'entità genitore
- sono anche possibili soluzioni “ibride”, soprattutto in gerarchie a più livelli





Attività della ristrutturazione

- Analisi delle ridondanze
- Eliminazione delle generalizzazioni
- Partizionamento/accorpamento di entità e relationship
- Scelta degli identificatori primari

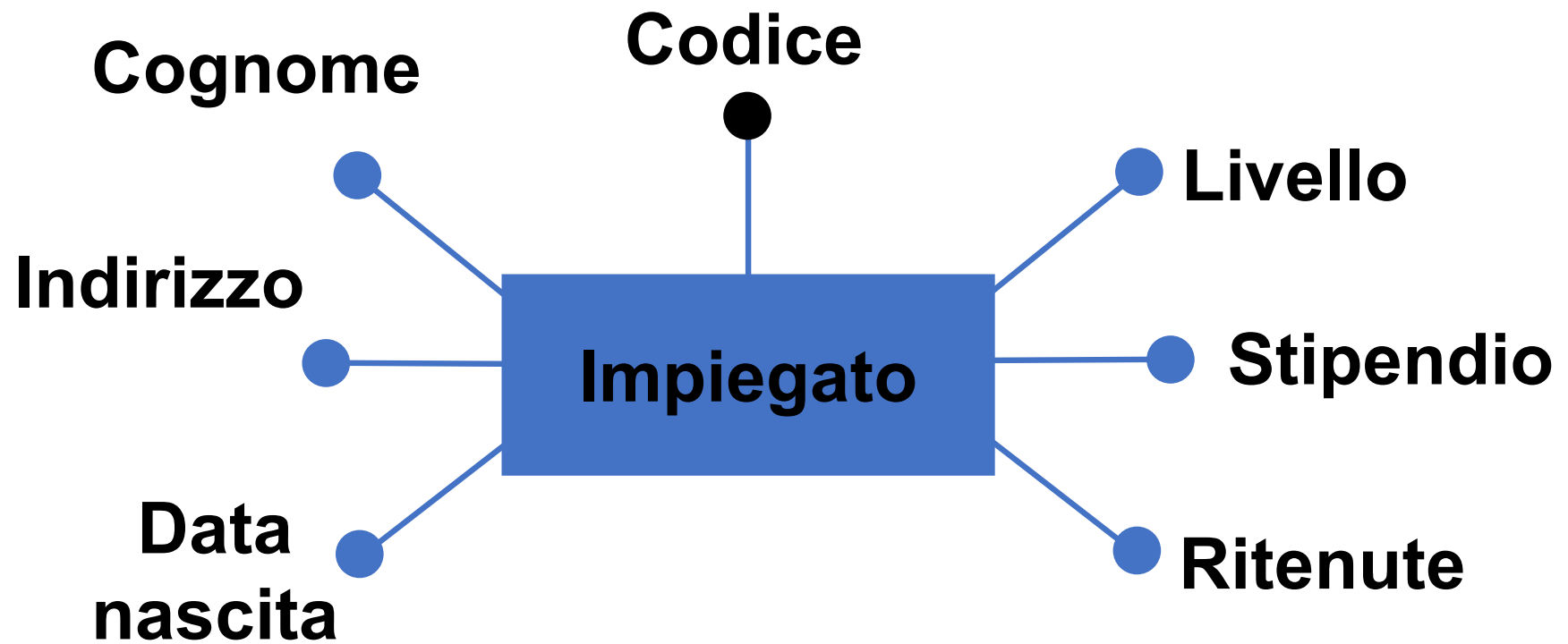
- Ristrutturazioni effettuate per rendere più efficienti le operazioni in base a un semplice principio, lo stesso utilizzato per l'eliminazione delle gerarchie
- Gli accessi si riducono:
 - separando attributi di un concetto che vengono acceduti separatamente
 - raggruppando attributi di concetti diversi acceduti insieme

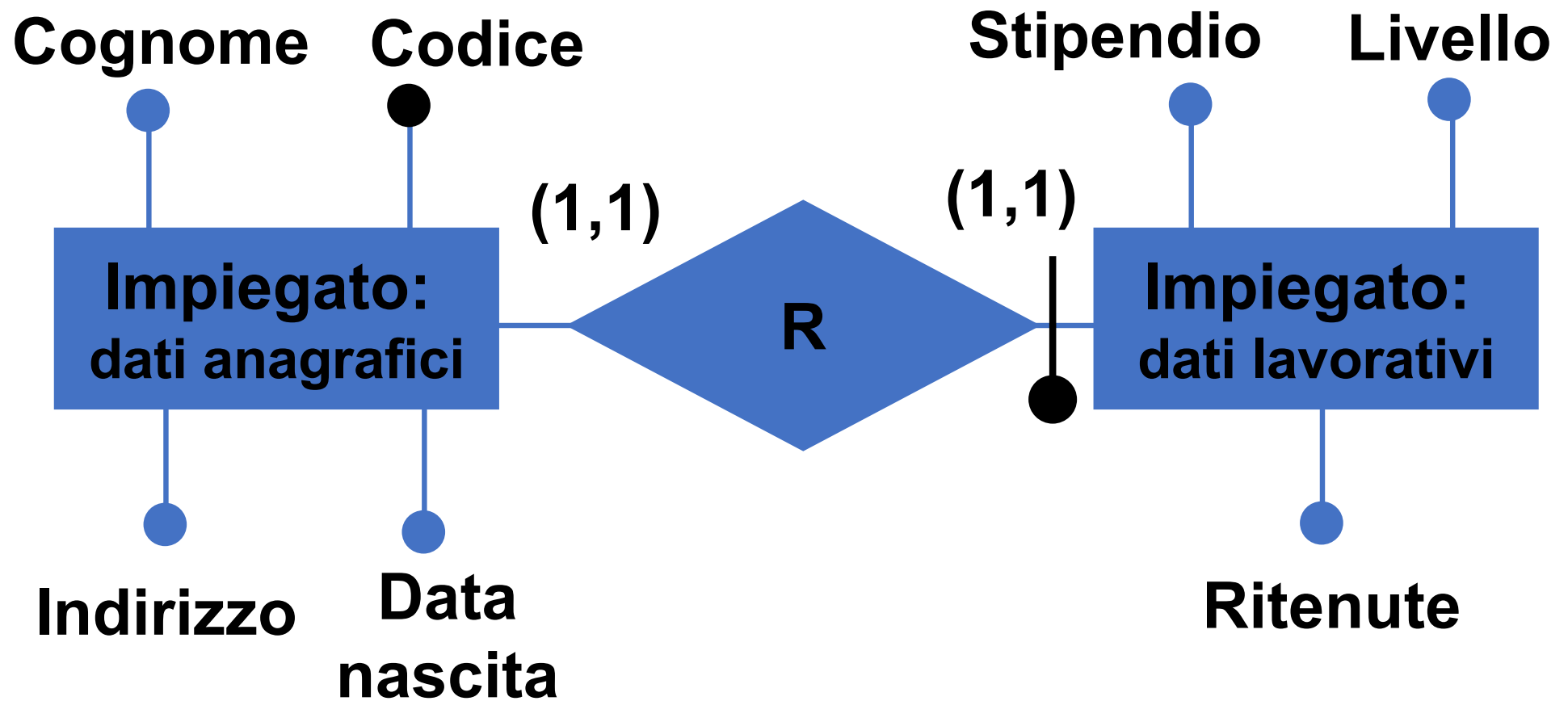
Partizionamento/accorpamento, casi principali

- partizionamento verticale di entità
- accorpamento di entità/ relationship
- eliminazione di attributi multivalore
- partizionamento orizzontale di relationship

Partizionamento/accorpamento, casi principali

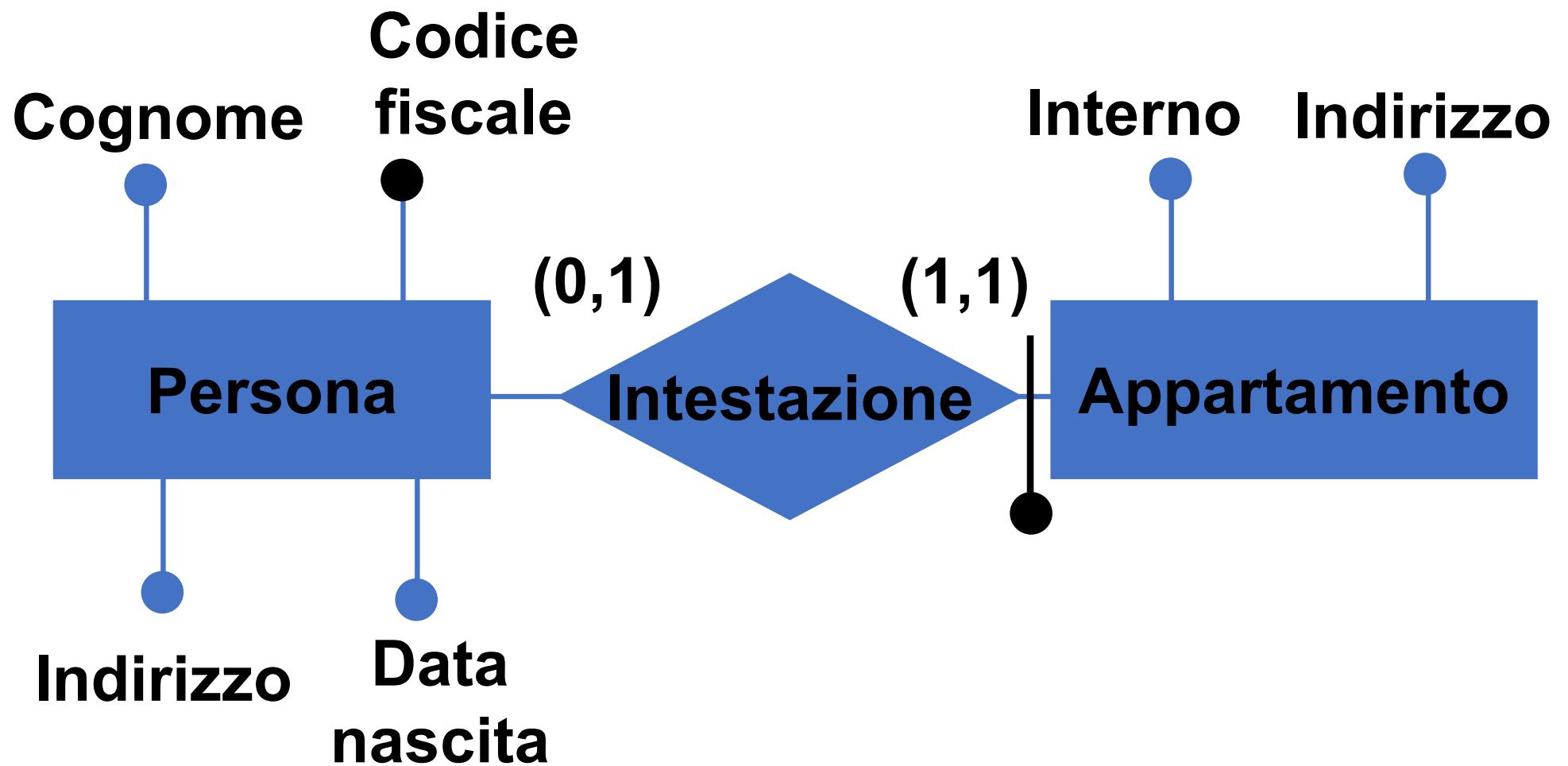
- partizionamento verticale di entità
- accorpamento di entità/ relationship
- eliminazione di attributi multivalore
- partizionamento orizzontale di relationship

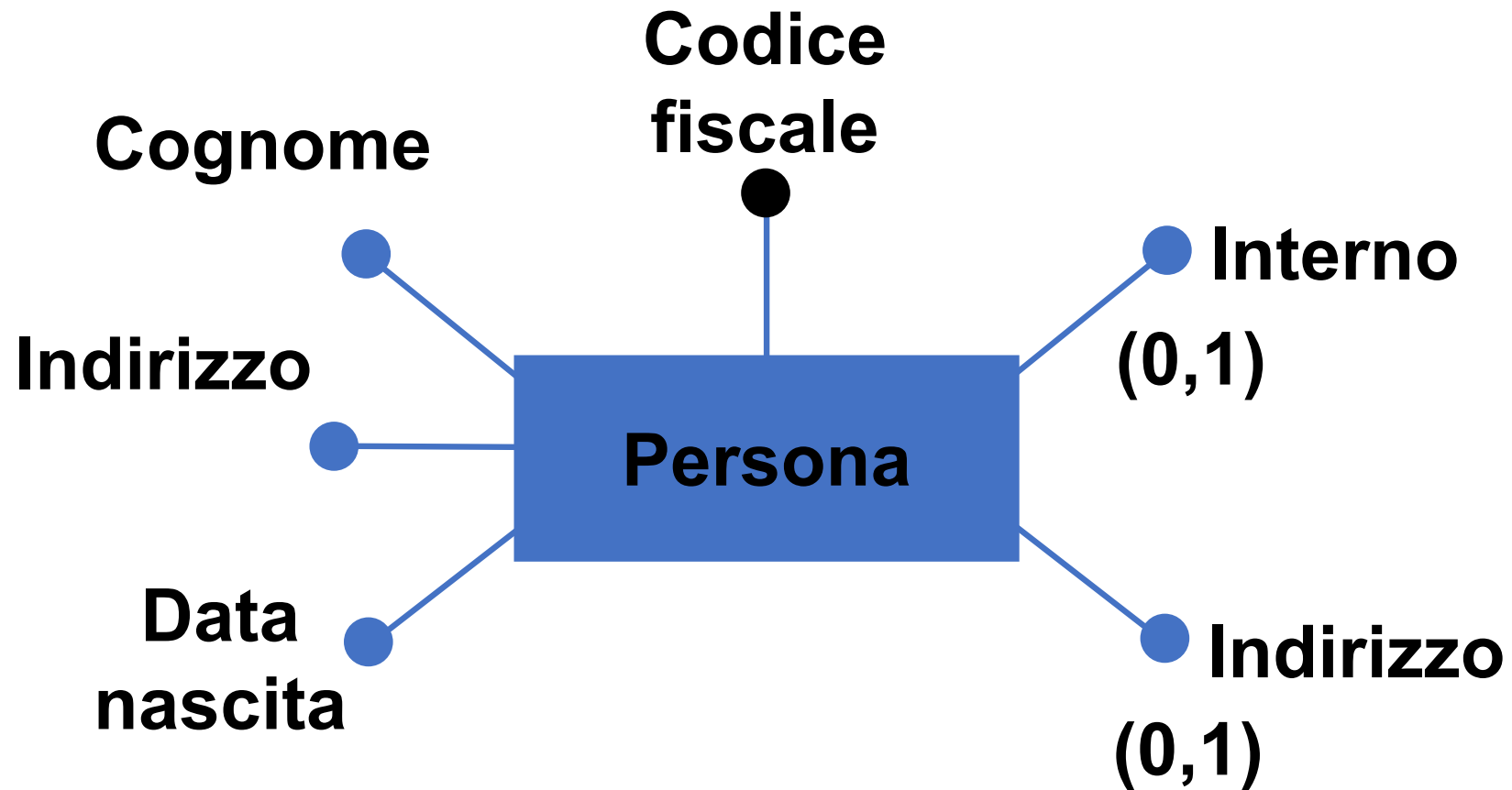




Partizionamento/accorpamento, casi principali

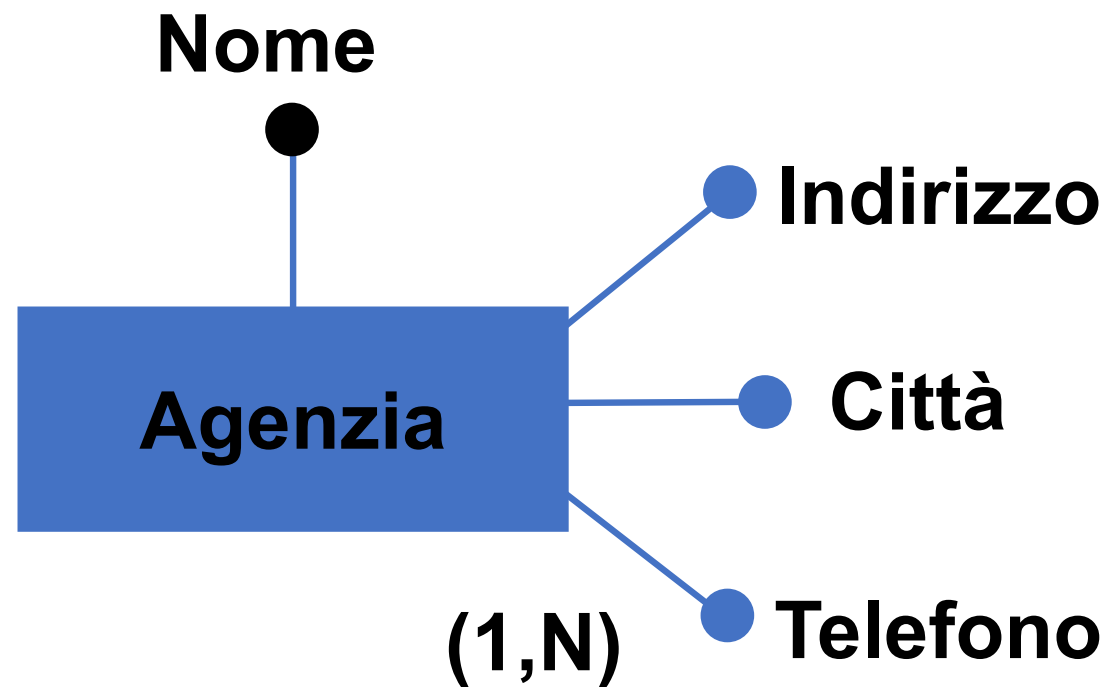
- partizionamento verticale di entità
- accorpamento di entità/ relationship
- eliminazione di attributi multivalore
- partizionamento orizzontale di relationship

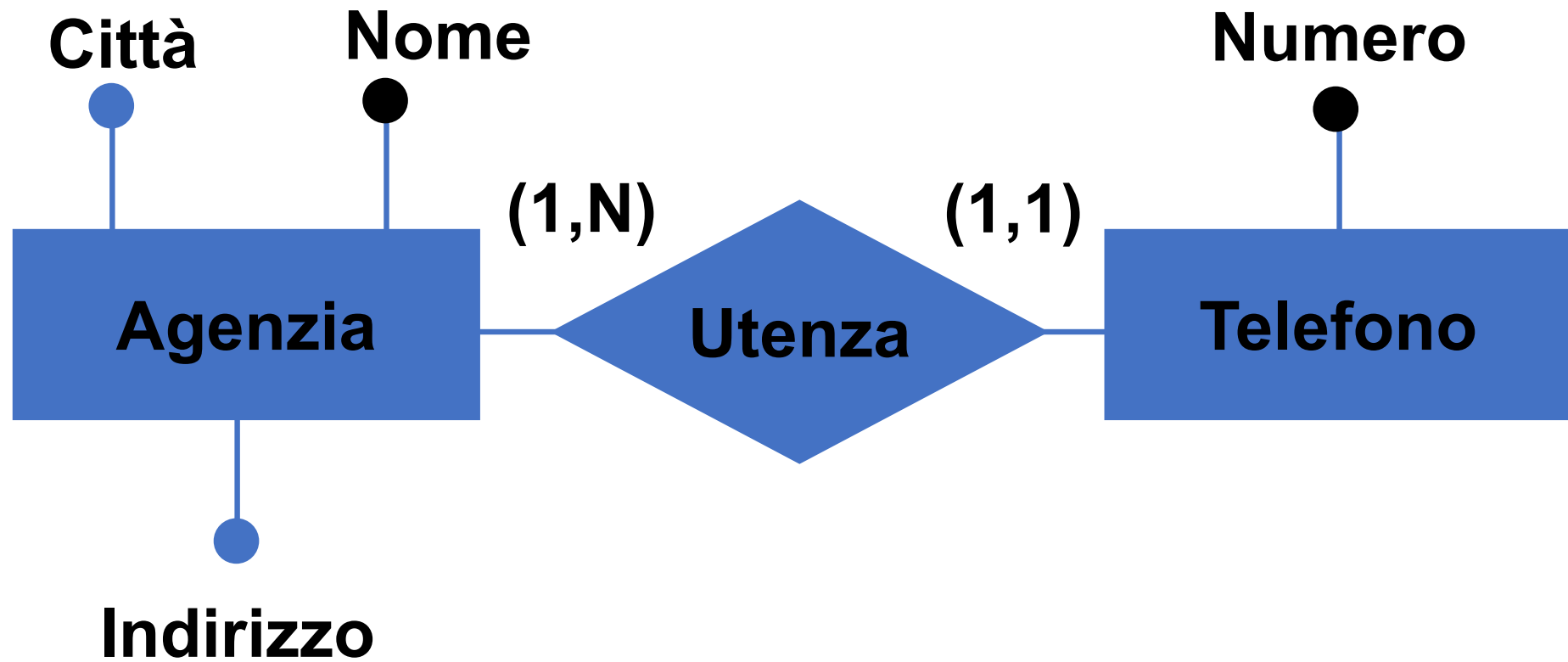


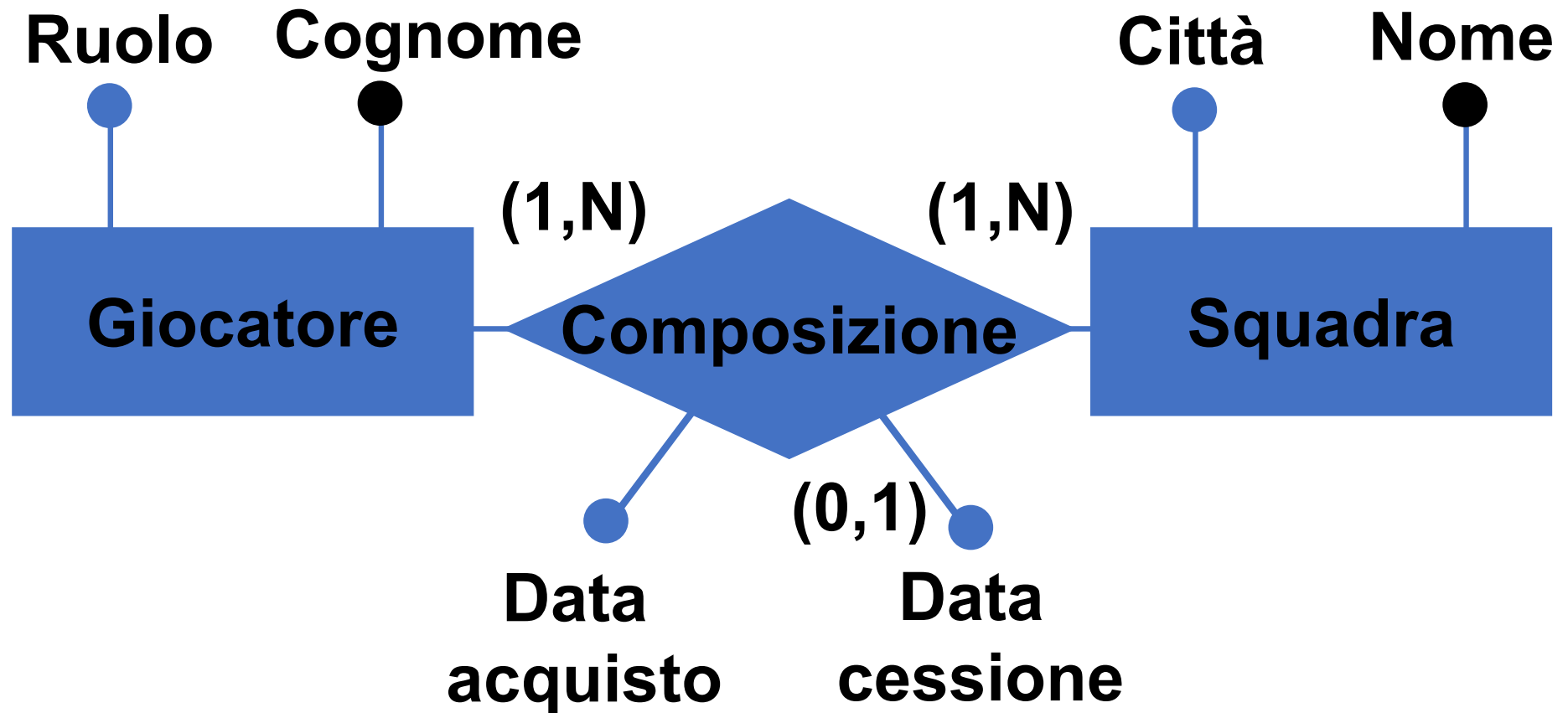


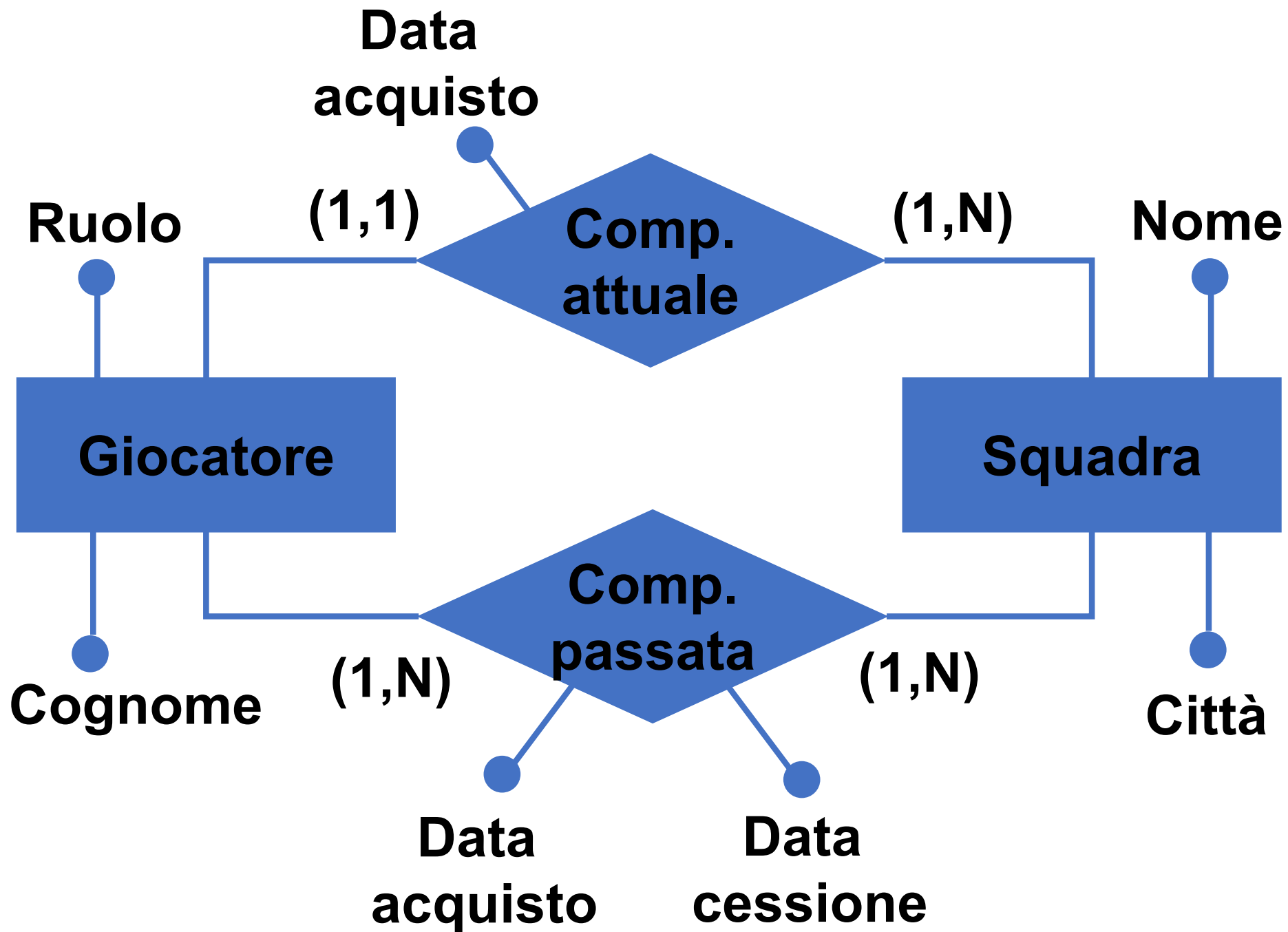
Partizionamento/accorpamento, casi principali

- partizionamento verticale di entità
- accorpamento di entità/ relationship
- eliminazione di attributi multivalore
- partizionamento orizzontale di relationship









Attività della ristrutturazione

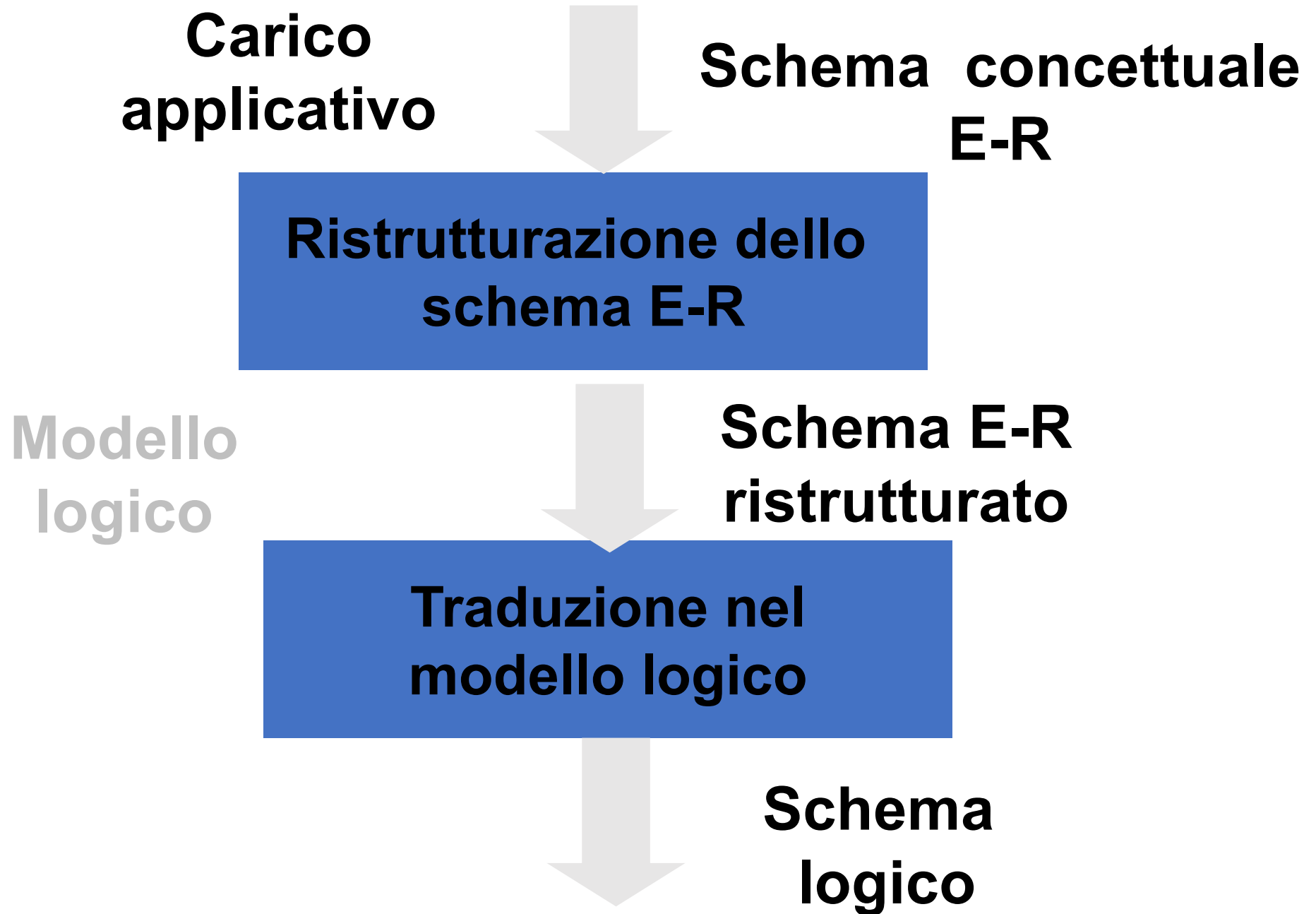
- Analisi delle ridondanze
- Eliminazione delle generalizzazioni
- Partizionamento/accorpamento di entità e relationship
- Scelta degli identificatori principali

Scelta degli identificatori principali

- operazione indispensabile per la traduzione nel modello relazionale
- Criteri
 - assenza di opzionalità
 - semplicità
 - utilizzo nelle operazioni più frequenti o importanti

Se nessuno degli identificatori soddisfa i requisiti visti?

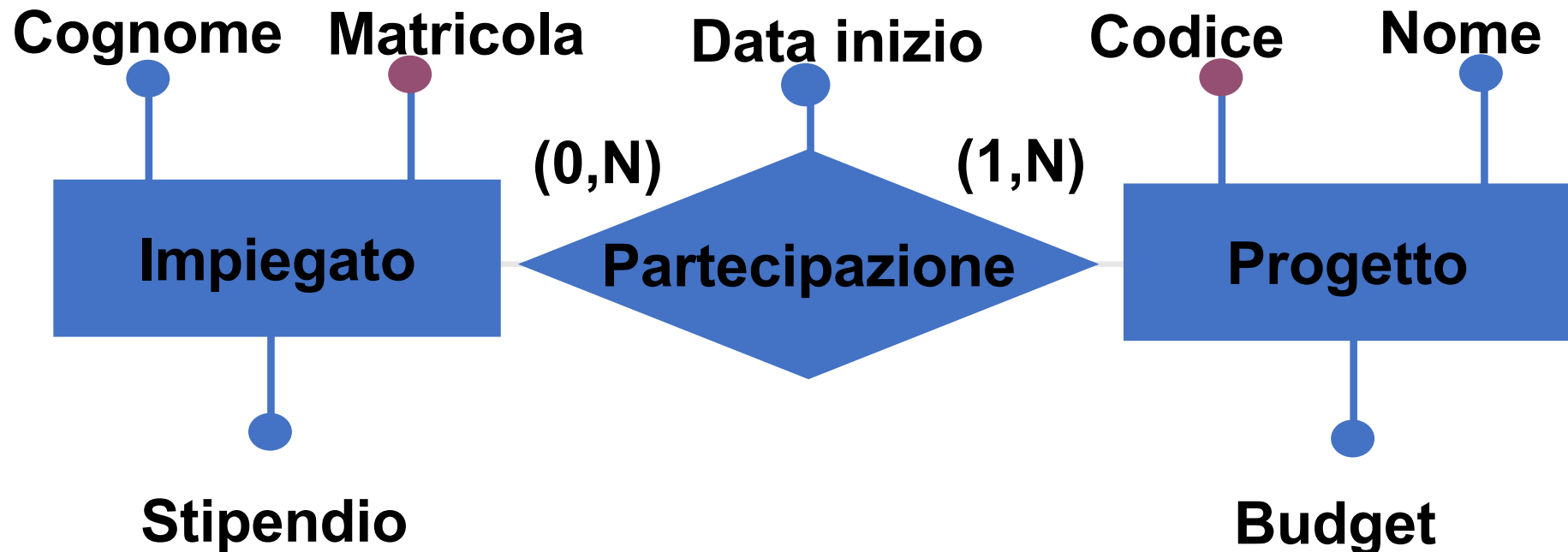
Si introducono nuovi attributi (codici**) contenenti
valori speciali generati appositamente per
questo scopo**



Traduzione verso il modello relazionale

- **idea di base:**
 - le entità diventano relazioni sugli stessi attributi
 - le relationship diventano relazioni sugli identificatori delle entità coinvolte (più gli attributi propri)

Entità e relationship molti a molti



Impiegato(Matricola, Cognome, Stipendio)

Progetto(Codice, Nome, Budget)

Partecipazione(Matricola, Codice, DataInizio)

Entità e relationship molti a molti

Impiegato(Matricola, Cognome, Stipendio)

Progetto(Codice, Nome, Budget)

Partecipazione(Matricola, Codice, DataInizio)

- con vincoli di integrità referenziale fra
 - **Matricola** in **Partecipazione** e (la chiave di) **Impiegato**
 - **Codice** in **Partecipazione** e (la chiave di) **Progetto**

**Nomi più espressivi per gli attributi
della chiave della relazione che
rappresenta la relationship**

Impiegato(Matricola, Cognome, Stipendio)

Progetto(Codice, Nome, Budget)

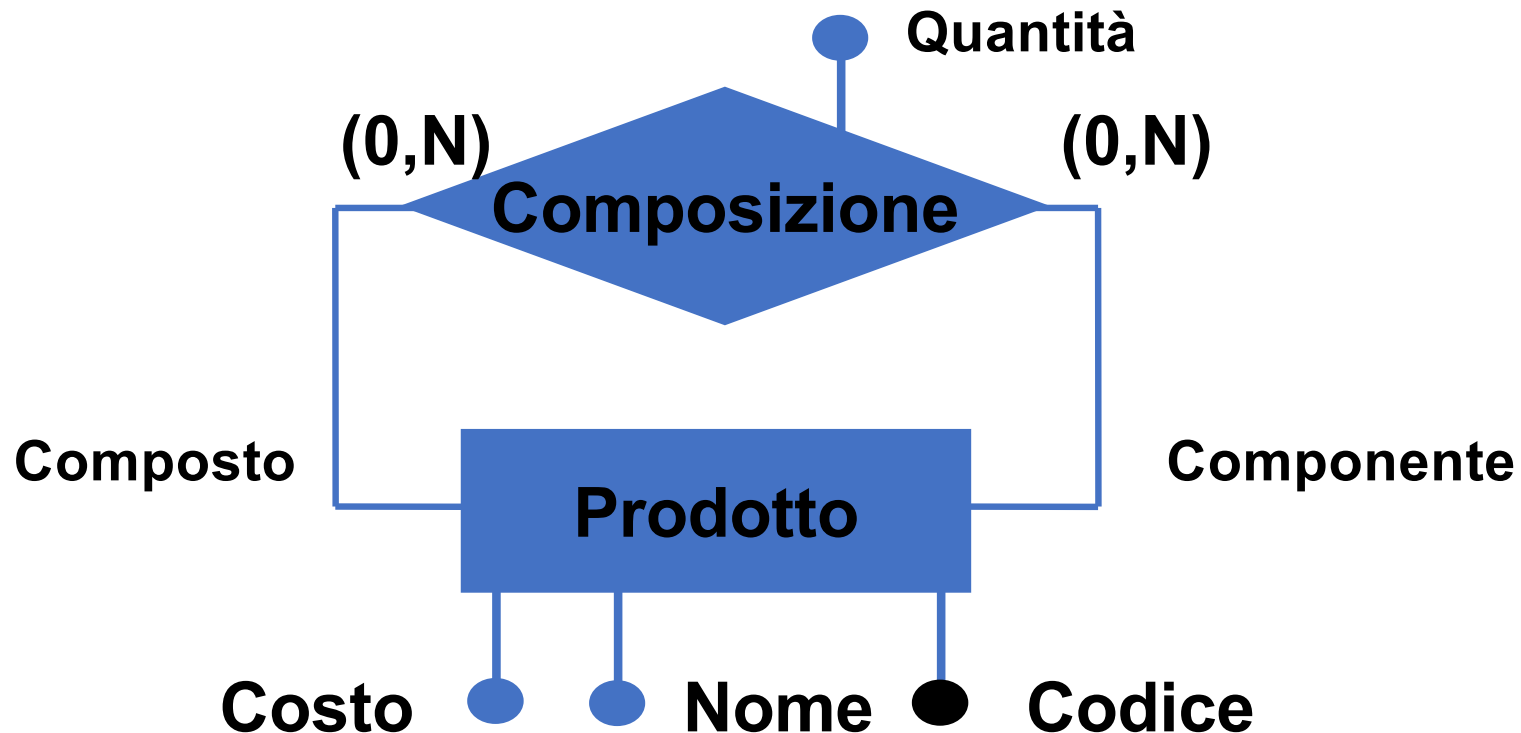
Partecipazione(Matricola, Codice, DataInizio)

Partecipazione(Impiegato, Progetto, DataInizio)

Nota

- La traduzione non riesce a tener conto delle cardinalità minime delle relationship molti a molti (se non con vincoli di CHECK complessi e poco usati)

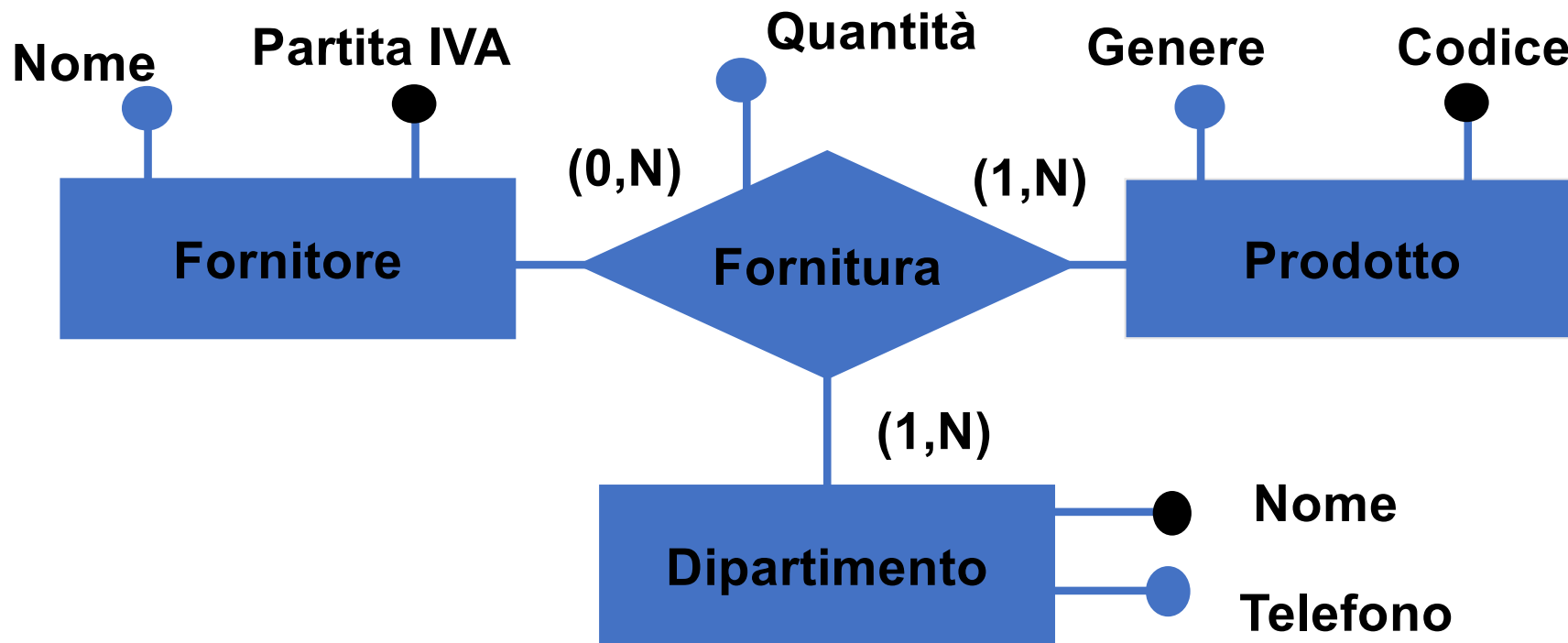
Relationship ricorsive



Prodotto(Codice, Nome, Costo)

Composizione(Composto, Componente, Quantità)

Relationship n-arie



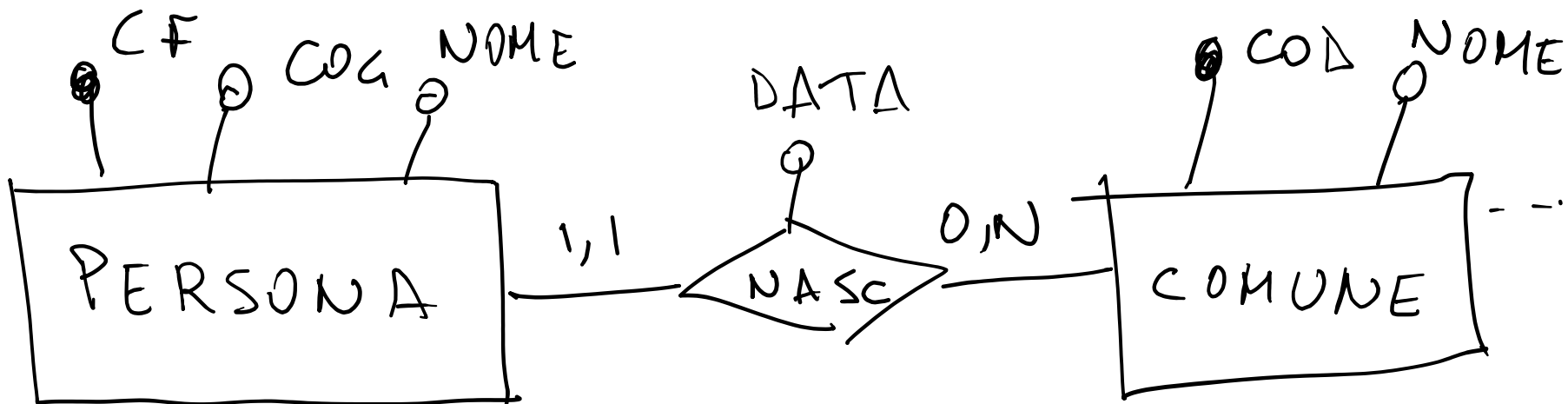
Fornitore(PartitaIVA, Nome)

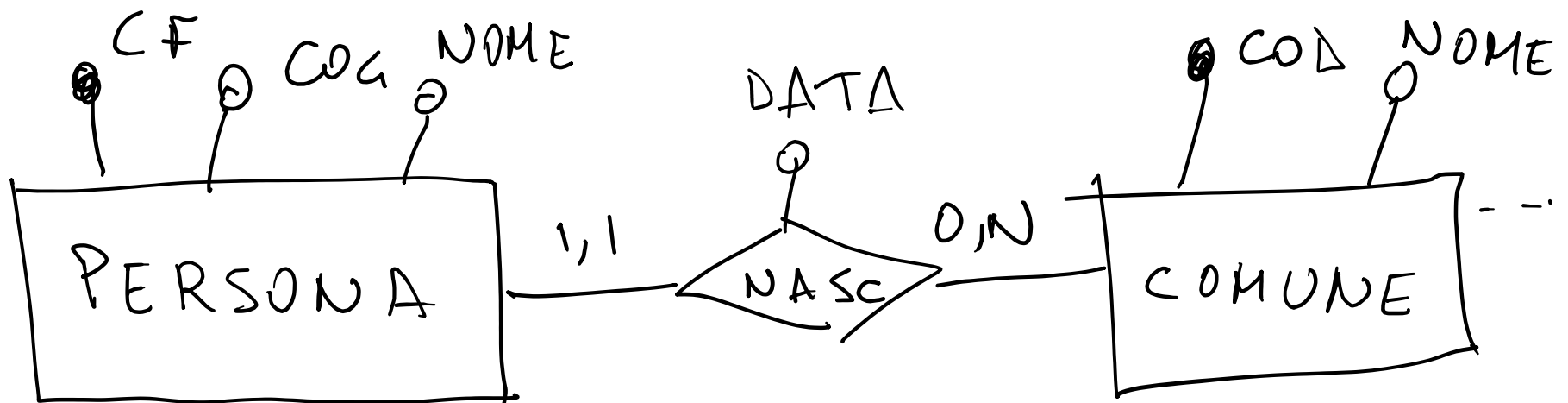
Prodotto(Codice, Genere)

Dipartimento(Nome, Telefono)

Fornitura(Fornitore, Prodotto, Dipartimento, Quantità)

Relationship uno a molti

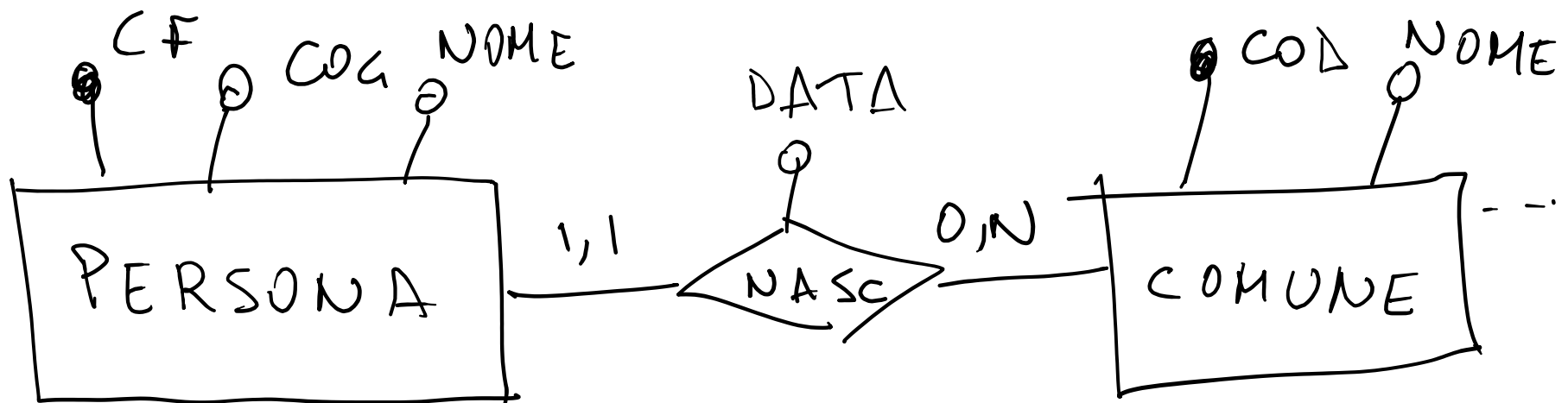




PERSONA (CF, COG, NOME)

COMUNE (COD, NOME, ...)

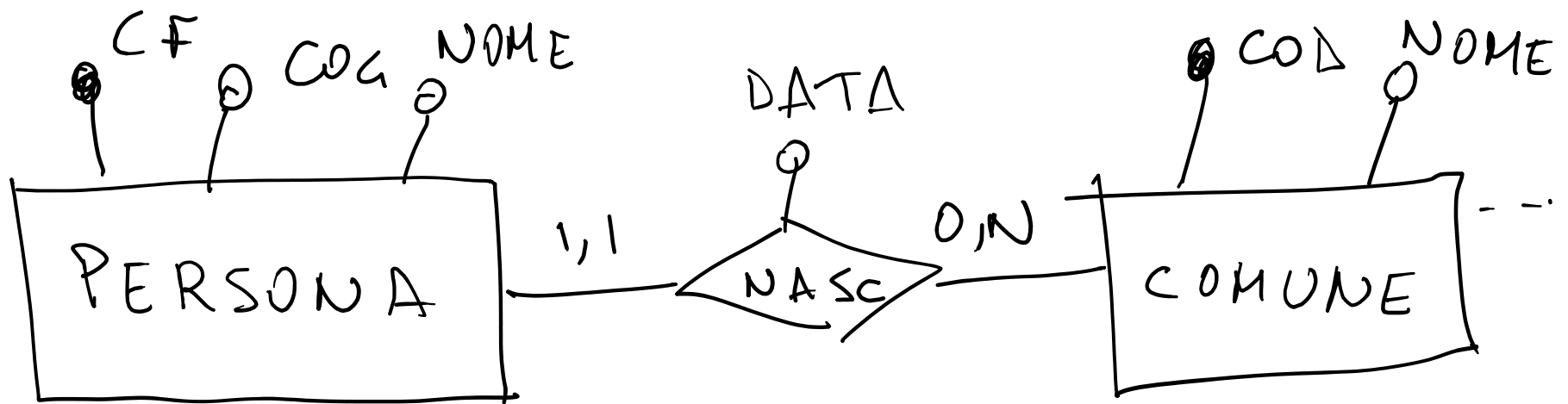
NASCITA (PERSONA, COMUNE, DATA)



PERSONA (CF, COG, NOME)

COMUNE (COD, NOME, ...)

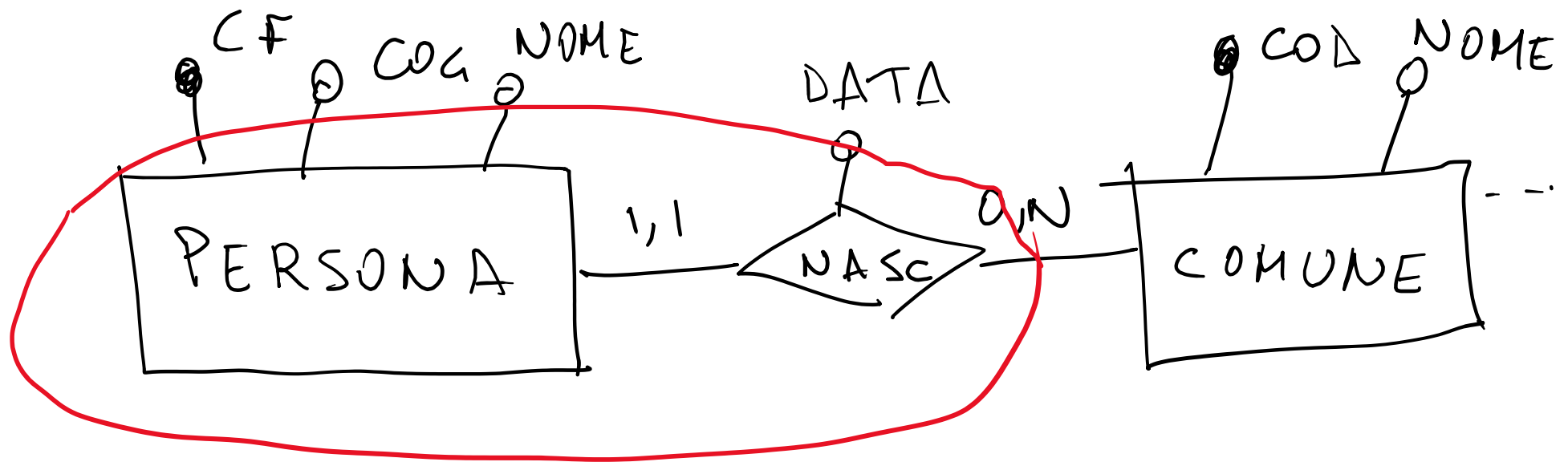
NASCITA (PERSONA, COMUNE, DATA)



PERSONA (CF, COG, NOME)

COMUNE (COD, NOME, ...)

NASCITA (PERSONA, COMUNE, DATA)
CF



PERSONA (CF, COG, NOME, COMUNE, DATA)

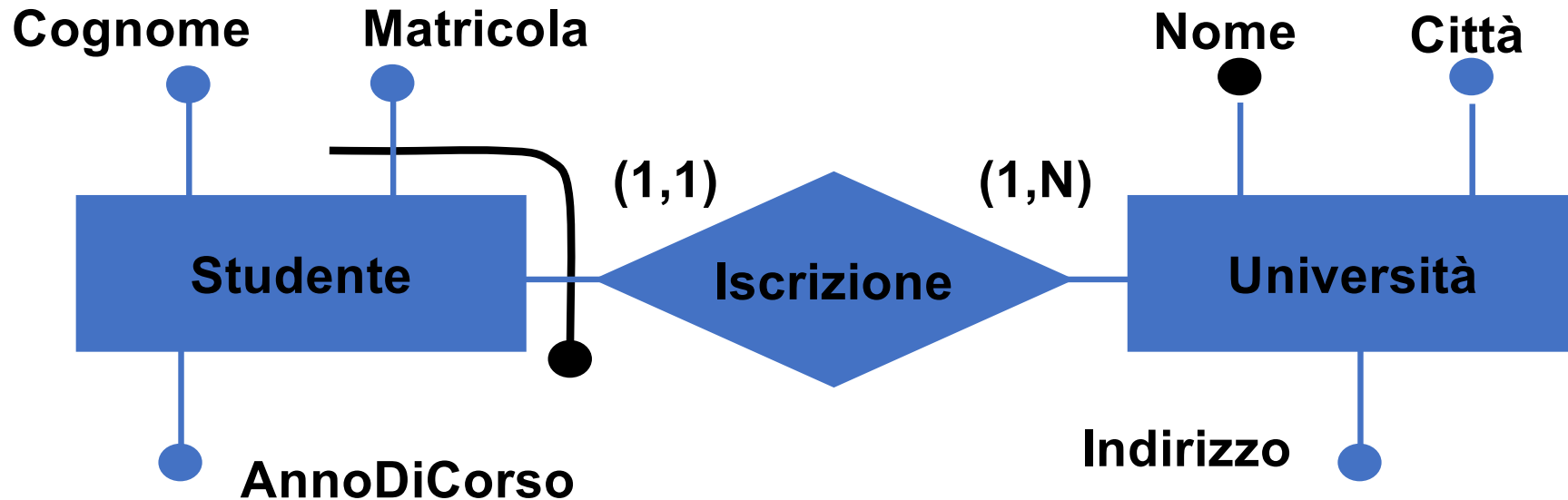
COMUNE (COD, NOME, ...)

~~NASCITA (PERSONA, COMUNE, DATA)~~
 CF

Nota

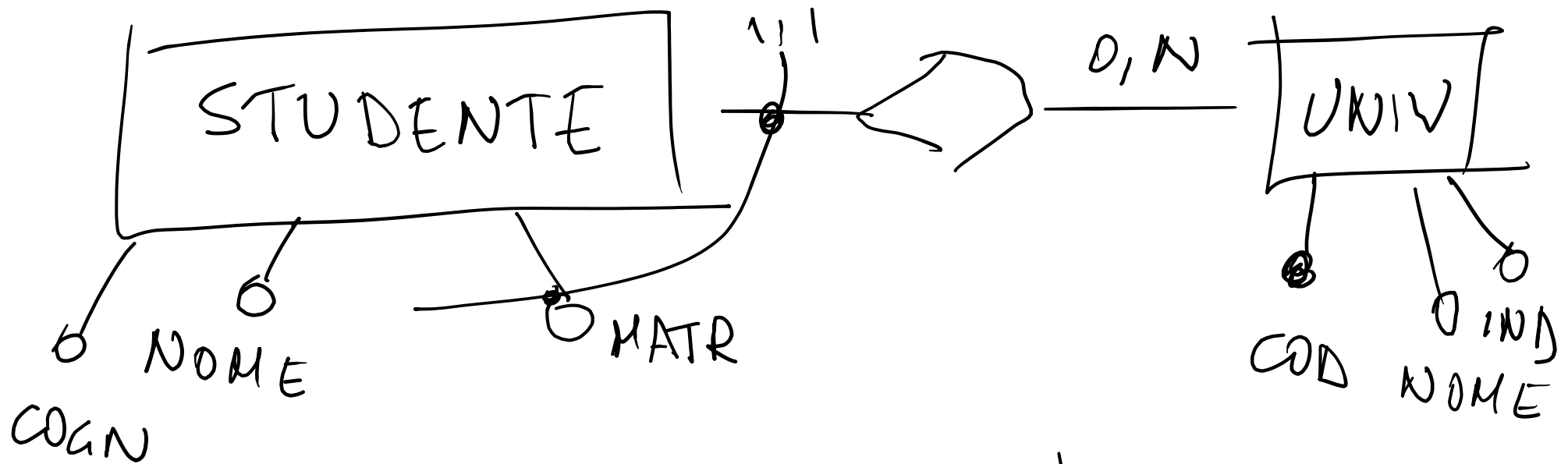
- La traduzione riesce a rappresentare efficacemente la cardinalità minima della partecipazione che ha 1 come cardinalità massima:
 - 0 : valore nullo ammesso
 - 1 : valore nullo non ammesso

Entità con identificazione esterna



Studente(Matricola, Università, Cognome, AnnoDiCorso)
Università(Nome, Città, Indirizzo)

- con vincolo ...

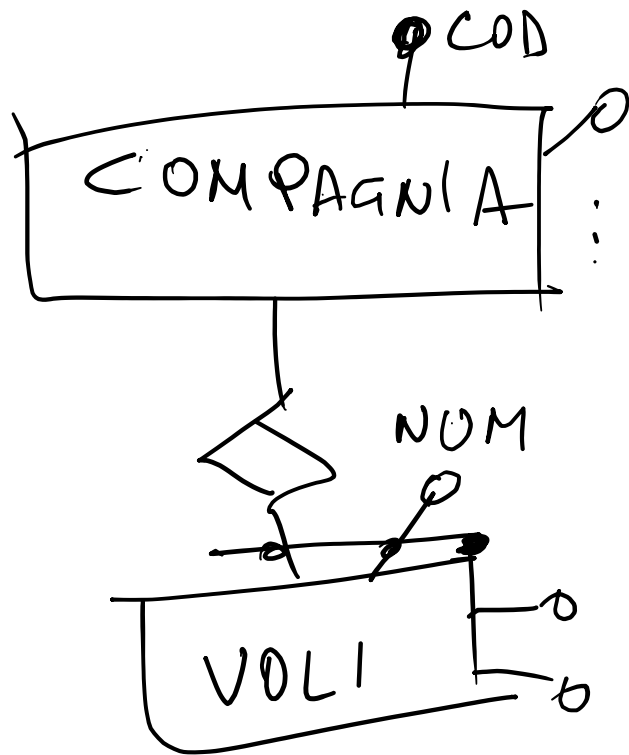


UNIV (COD, NOME, IND)

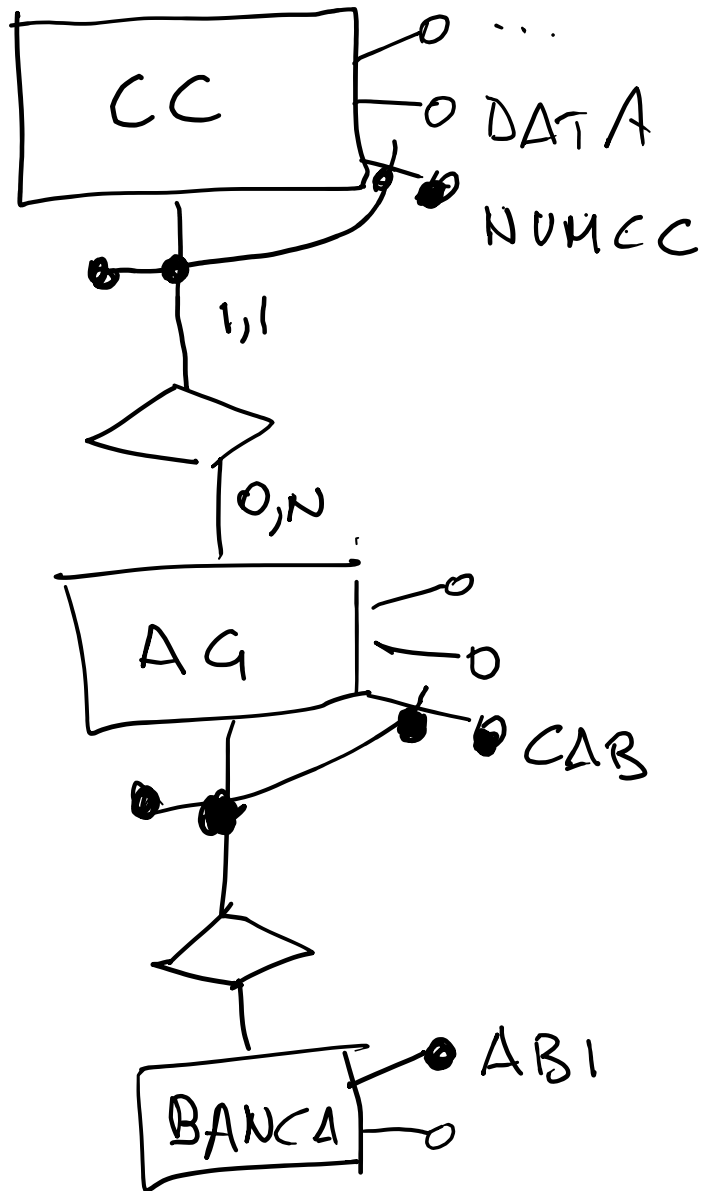
STUDENTE (MATR, UNIV, NOME, COGNOME)

Esempi

- Voli e coordinate bancarie, vediamo alla lavagna



$VOLI(\underbrace{COMP}, \underbrace{NUM}, \dots)$



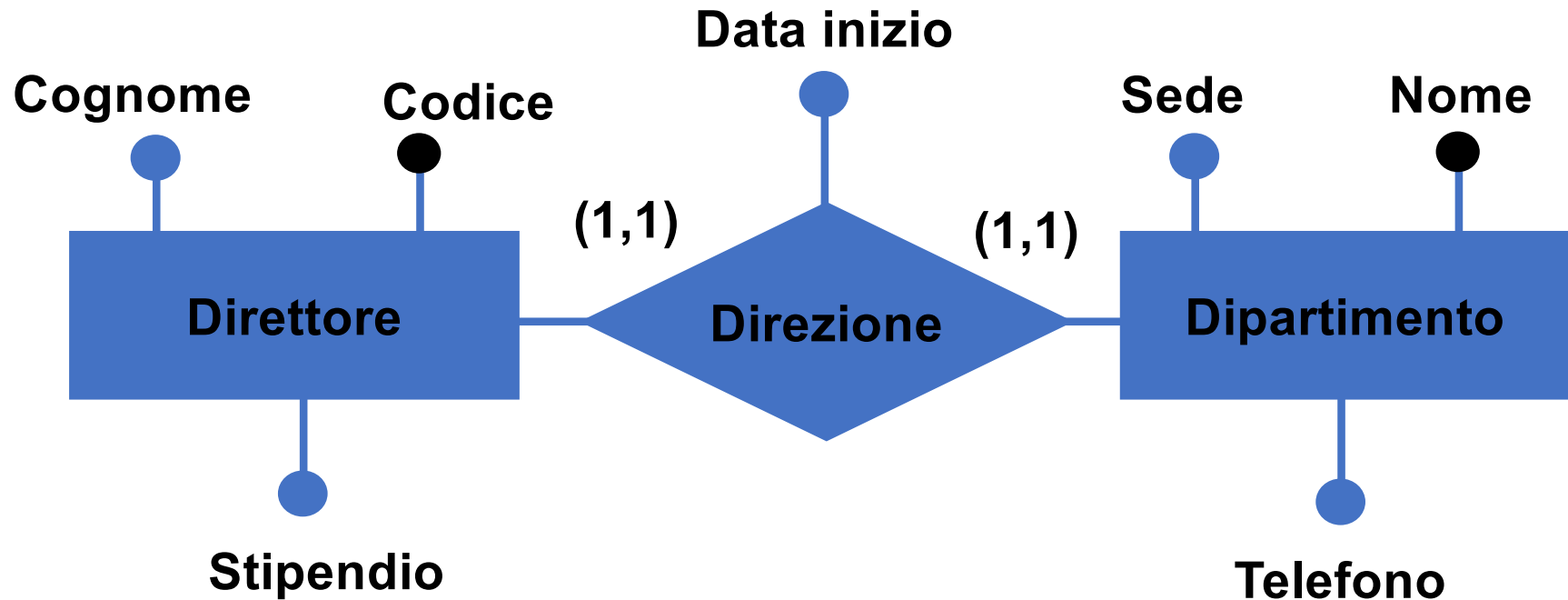
$CC(\underline{ABI}, \underline{CAB}, \underline{NUMCC}, \dots)$



$AGENZIA(\underline{ABI}, \underline{CAB}, \dots)$

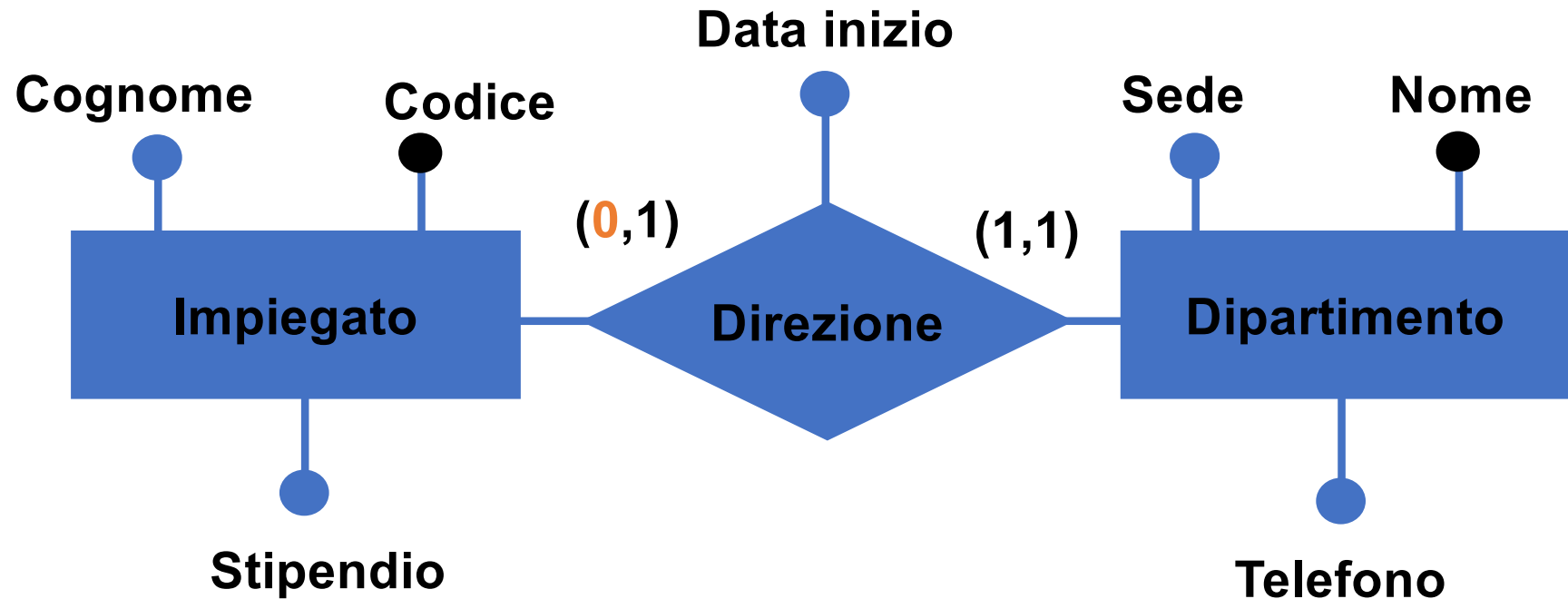
$BANCA(\underline{ABI}, \dots)$

Relationship uno a uno



- **varie possibilità:**
 - **fondere da una parte o dall'altra**
 - **fondere tutto?**

Una possibilità privilegiata

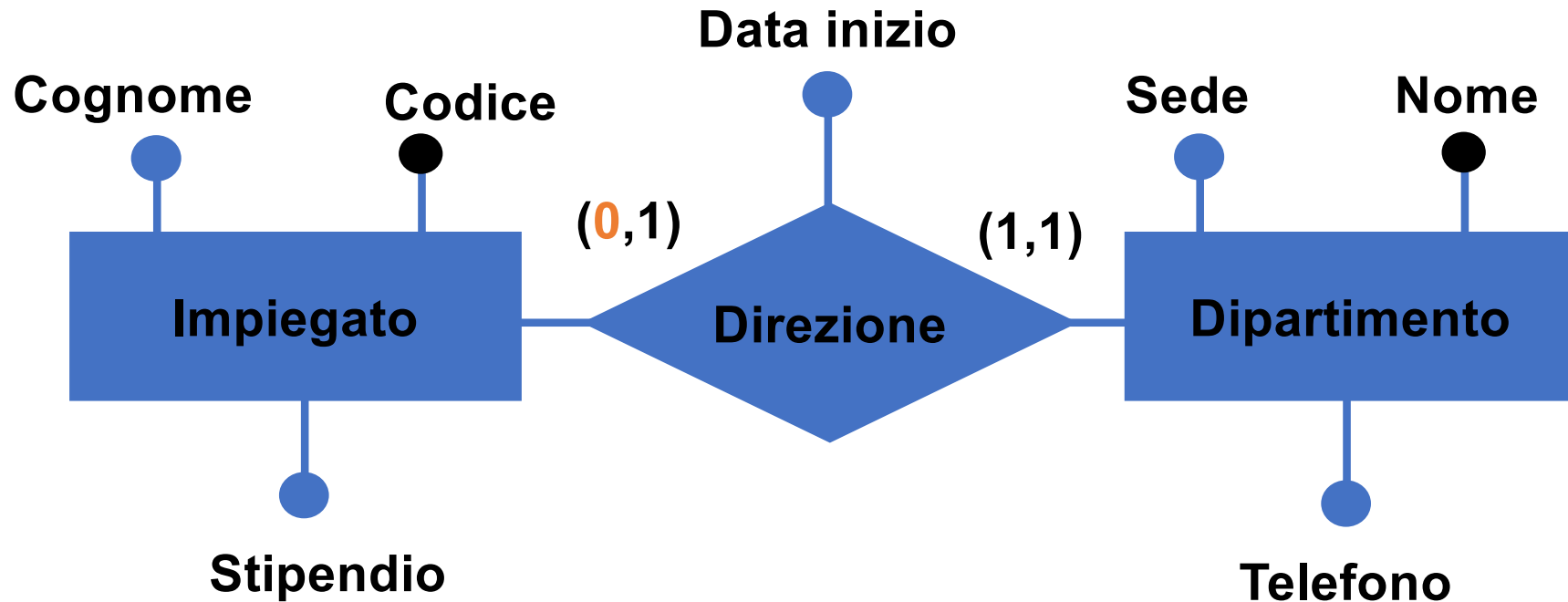


Impiegato (Codice, Cognome, Stipendio)

Dipartimento (Nome, Sede, Telefono, Direttore, DataInizioD)

- **con vincolo di integrità referenziale, senza valori nulli**

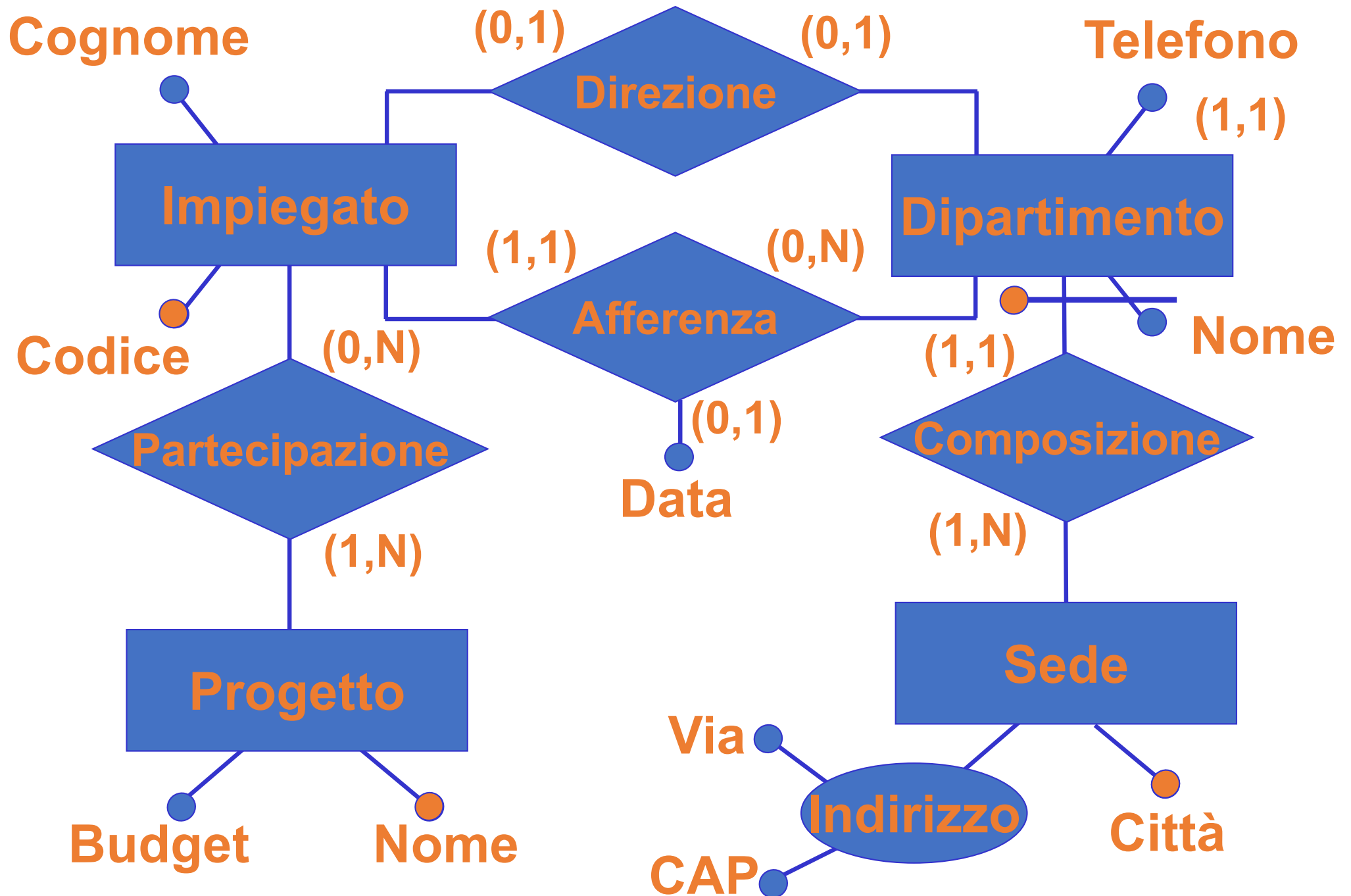
Alternativa, non preferibile

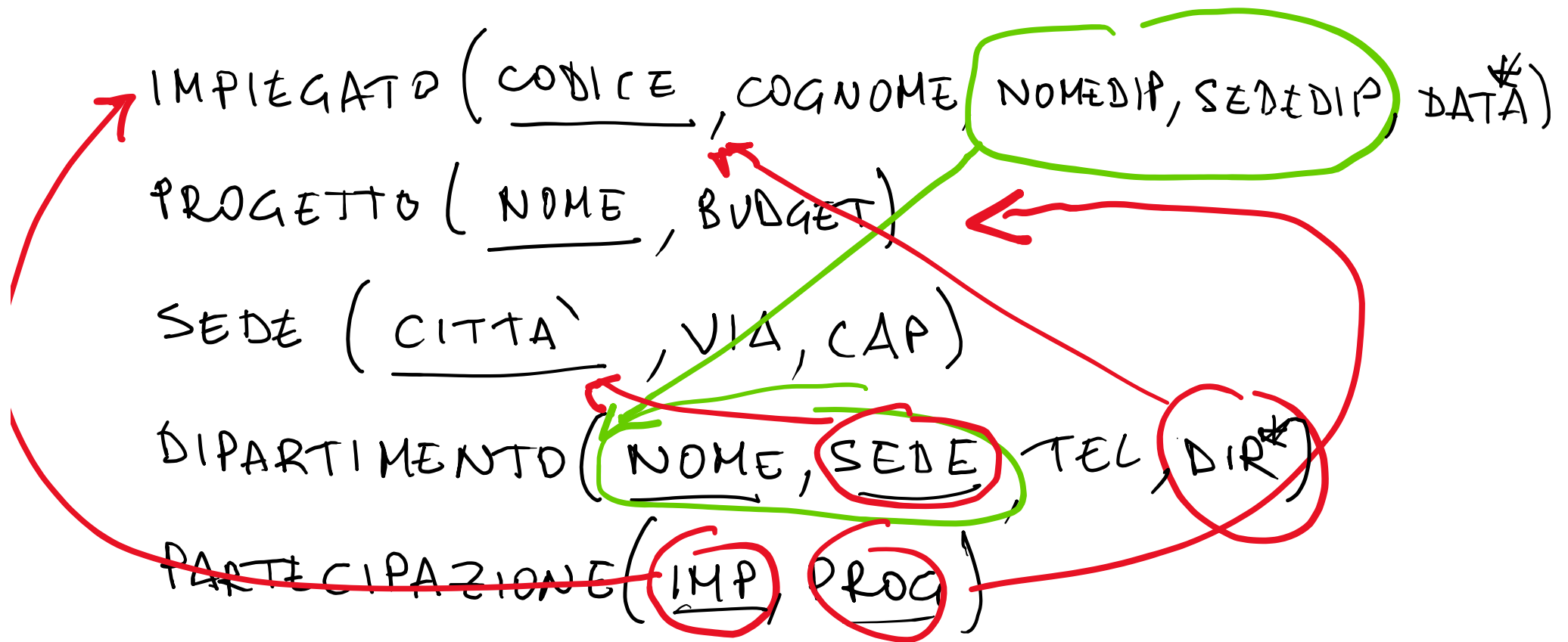


Impiegato (Codice, Cognome, Stipendio, Dip, DataInizioD)

Dipartimento (Nome, Sede, Telefono)

- con vincolo di integrità referenziale e valore nullo ammesso su Dip e DataInizioD





SEDE (CITTÀ, VIA, CAP)

PROGETTO(NOME, BUDGET)

IMPIEGATO(COD, COG, NOMEDIP, SEDE, DATA ASS^{*})

DIPARTIMENTO(NOME, SEDE, TELEFONO, DIRETTORE^{*})

PARTICIPAZIONE(IMP, PROG)

Schema finale

**Impiegato(Codice, Cognome,
Dipartimento, Sede, Data*)**

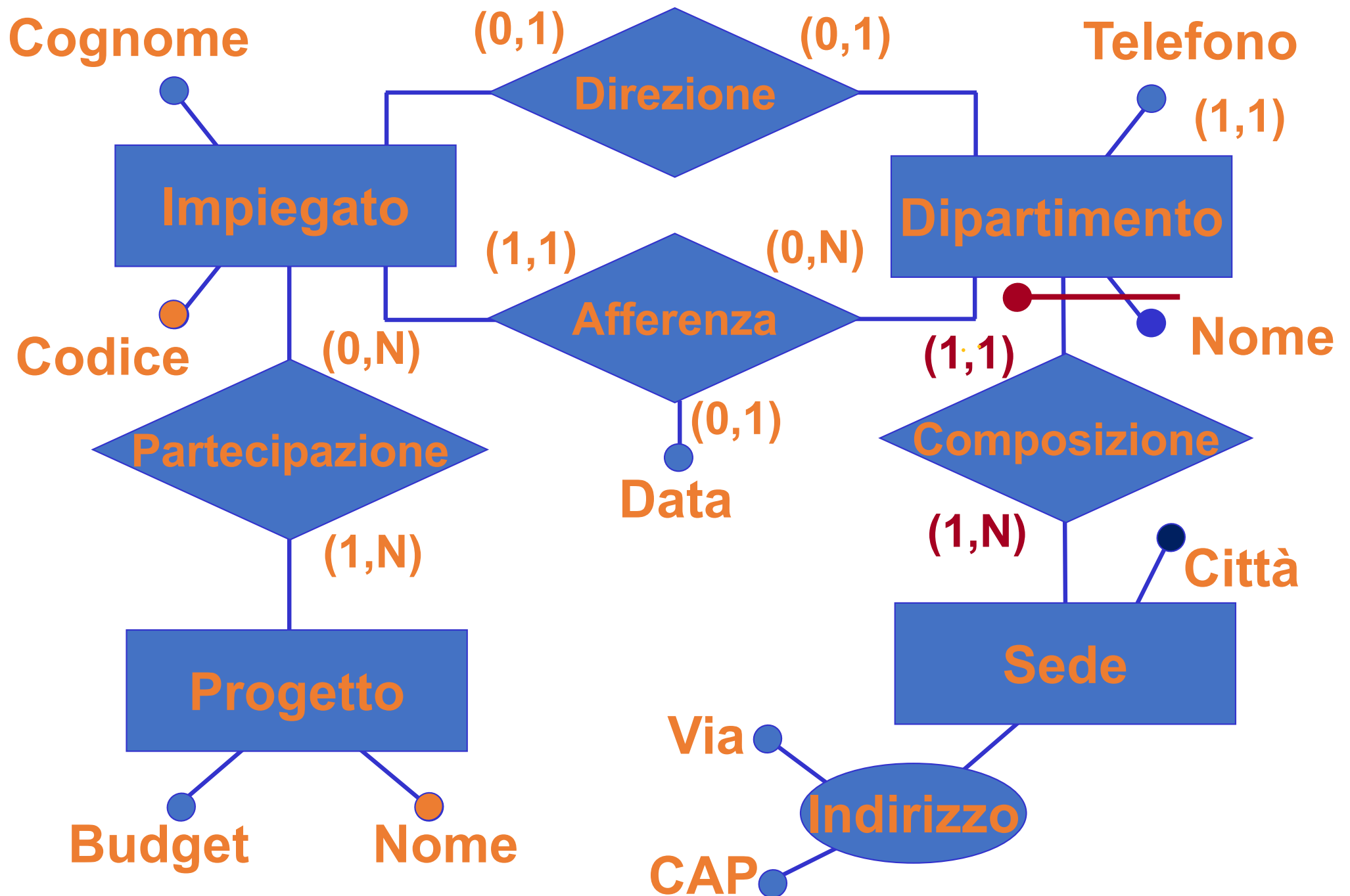
Dipartimento(Nome, Città, Telefono, Direttore*)
Sede(Città, Via, CAP)

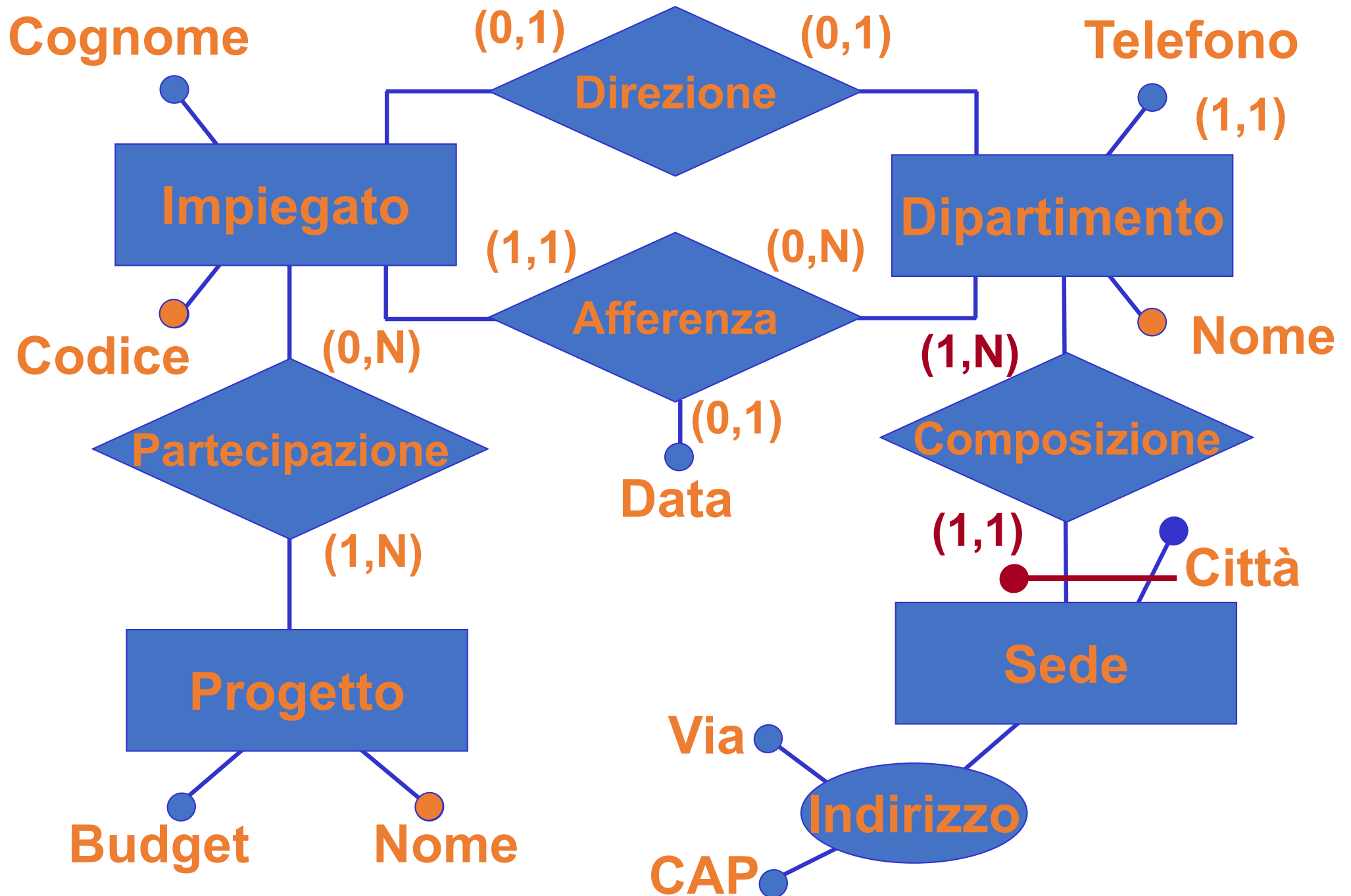
Progetto(Nome, Budget)

Partecipazione(Impiegato, Progetto)

Attenzione

- Differenze apparentemente piccole in cardinalità e identificatori possono cambiare di molto il significato ...





Schema finale

**Impiegato(Codice, Cognome,
Dipartimento, Data*)**

Dipartimento(Nome, Telefono, Direttore*)

Sede(Dipartimento, Città, Via, CAP)

Progetto(Nome, Budget)

Partecipazione(Impiegato, Progetto)