



# Basi di dati

**Maurizio Lenzerini**

***Dipartimento di Informatica e Sistemistica “Antonio Ruberti”  
Università di Roma “La Sapienza”***

Anno Accademico 2017/2018

Maurizio Lenzerini

<http://www.dis.uniroma1.it/~lenzerin/home/?q=node/44>



# 5. La progettazione logica

## 5.1 introduzione alla progettazione logica

1. **introduzione alla progettazione logica**
2. ristrutturazione dello schema ER
3. traduzione diretta nel modello relazionale
4. ristrutturazione dello schema logico



# Fasi della progettazione

↓  
**Requisiti**

**Progettazione  
concettuale**

**“COSA”**

↓  
**Schema concettuale**

**Progettazione  
logica**

↓  
**Schema logico**

**“COME”**

**Progettazione  
fisica / tuning**

↓  
**Schema fisico**



# Obiettivo della progettazione logica

**Tradurre** lo schema concettuale (espresso nel modello ER con vincoli) in uno schema logico che rappresenti gli stessi dati:

- utilizzando il **modello logico** del DBMS scelto  
(nel nostro caso, il modello relazionale con vincoli)
- in maniera **corretta**, **completa** ed **efficiente**



# Dati di ingresso e uscita

## Ingresso:

- schema concettuale (diagramma ER e dizionario dei dati)
- informazioni sul carico applicativo

## Uscita:

- schema logico (nel modello relazionale)
- vincoli aggiuntivi
- documentazione associata



# Non si tratta di una pura e semplice traduzione

Motivi:

- Alcuni aspetti dello schema ER possono non essere direttamente rappresentabili nel modello relazionale. È quindi opportuno ristrutturare lo schema ER in modo da renderlo traducibile in modo diretto.
- È necessario porre attenzione alle **prestazioni**.
- Adottiamo un semplice **modello di costo** che permette di fornire una valutazione approssimata delle prestazioni della base di dati in funzione di un certo carico applicativo.
- Le scelte durante in fase di progettazione logica devono essere effettate con l'obiettivo di ottimizzare le prestazioni.



# Carico applicativo

Consideriamo degli “indicatori” dei parametri che regolano le prestazioni:

- **tempo di esecuzione** delle operazioni di principale interesse:  
numero di istanze (di entità e relazioni) mediamente accedute durante l'esecuzione dell'operazione (**accessi**)
- **spazio di memoria** necessario per memorizzare i dati di interesse

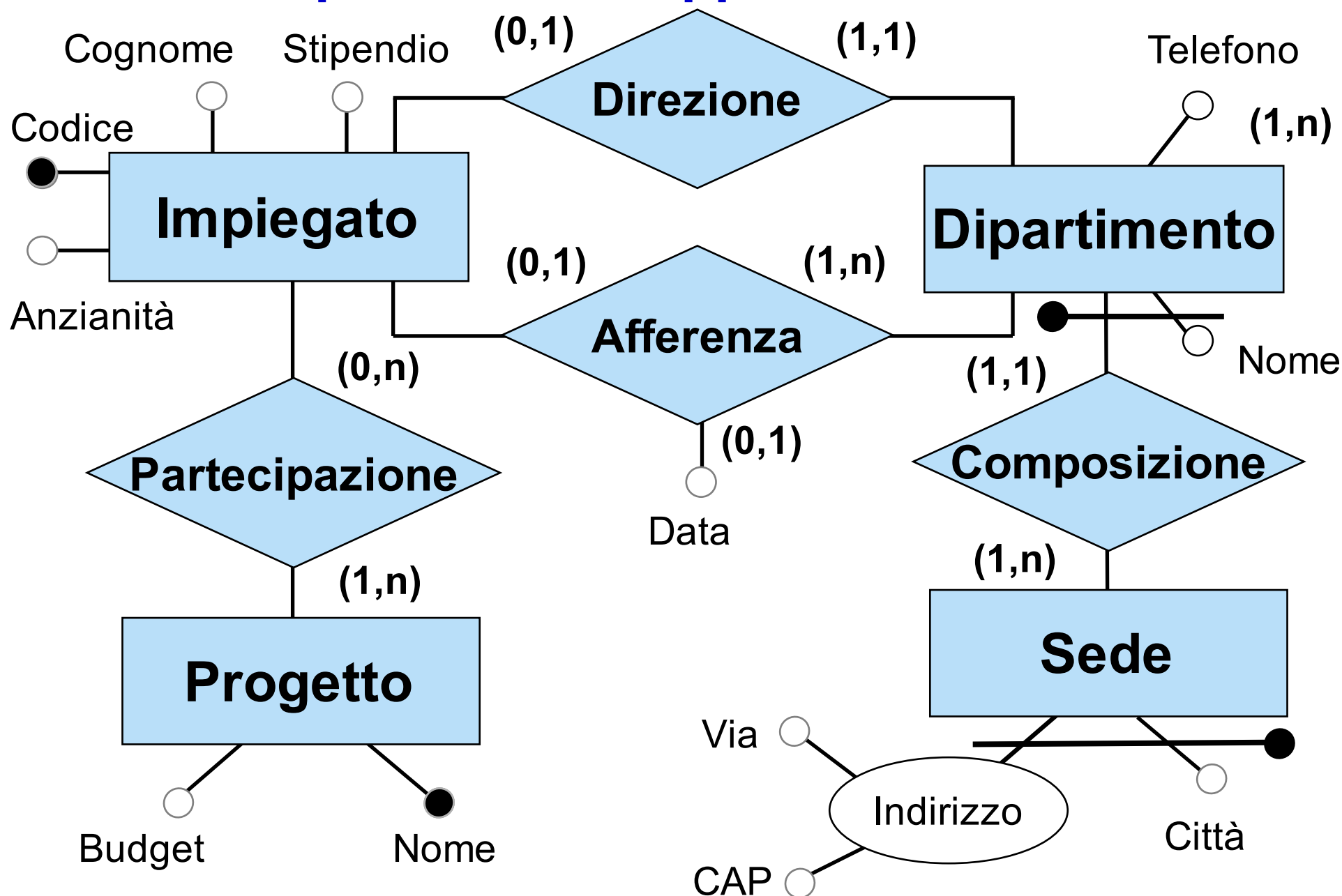
Per valutare questi parametri bisogna conoscere (oltre allo schema):

- **volume dei dati:**
  - numero di istanze previste di entità e relazioni
  - dimensione di ciascun attributo
- **caratteristiche delle operazioni:**
  - tipo: interattiva o batch
  - frequenza: numero medio di esecuzioni in un certo periodo
  - dati coinvolti

Si noti che la valutazione sarà necessariamente approssimata, in quanto le prestazioni effettive della base di dati dipendono anche da parametri fisici, difficilmente prevedibili in questa fase (DBMS utilizzato, indici, ...).



## Esempio di carico applicativo: schema







## Esempio di carico applicativo: operazioni

Supponiamo che le operazioni di interesse siano:

1. Assegna un impiegato ad un progetto.
2. Trova tutti i dati di un impiegato, del dipartimento nel quale lavora e dei progetti ai quali partecipa.
3. Trova i dati di tutti gli impiegati di un certo dipartimento.
4. Per ogni sede, trova i suoi dipartimenti con il cognome del direttore e l'elenco degli impiegati del dipartimento.



# Tabella dei volumi e tavola delle operazioni

## Tabella dei volumi

Concetto	Costrutto	Volume
Sede	Entità	10
Dipartimento	Entità	80
Impiegato	Entità	200
Progetto	Entità	500
Composizione	Relazione	80
Afferenza	Relazione	190
Direzione	Relazione	80
Partecipazione	Relazione	600

## Tabella delle operazioni

Op.	Tipo	Frequenza
1	Interattiva	50 / giorno
2	Interattiva	100 / giorno
3	Interattiva	10 / giorno
4	Batch	2 / sett.



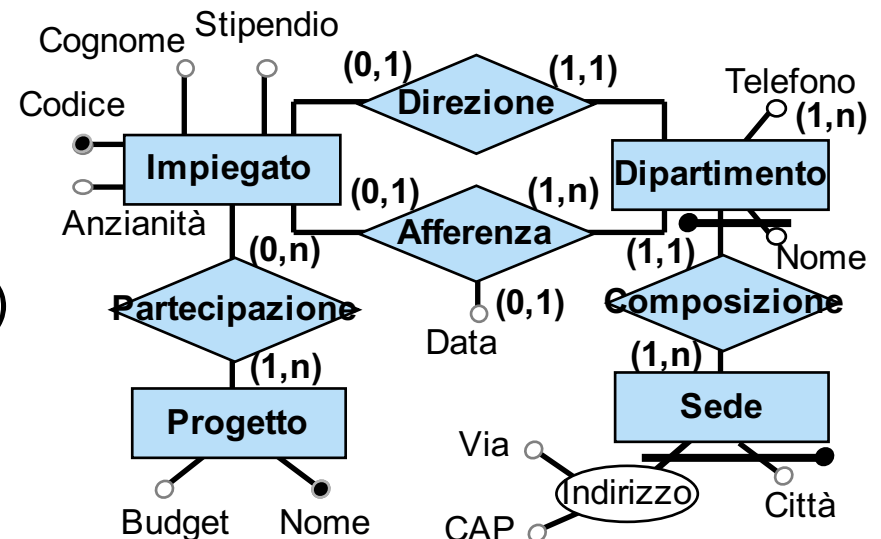
## Tabella dei volumi

Si noti che i valori relativi al numero di istanze di entità e relazioni nella tabella dei volumi sono influenzati:

- dalle cardinalità nello schema
- dal numero medio di volte che le istanze delle entità partecipano alle relazioni

### Esempio:

- $\text{vol}(\text{Composizione}) = \text{vol}(\text{Dipartimento})$
- $\text{vol}(\text{Direzione}) = \text{vol}(\text{Dipartimento})$
- $\text{vol}(\text{Afferenza}) \leq \text{vol}(\text{Impiegato})$
- se ogni impiegato partecipa in media a 3 progetti:  
 $\text{vol}(\text{Partecipazione}) \approx 3 \cdot \text{vol}(\text{Impiegato})$

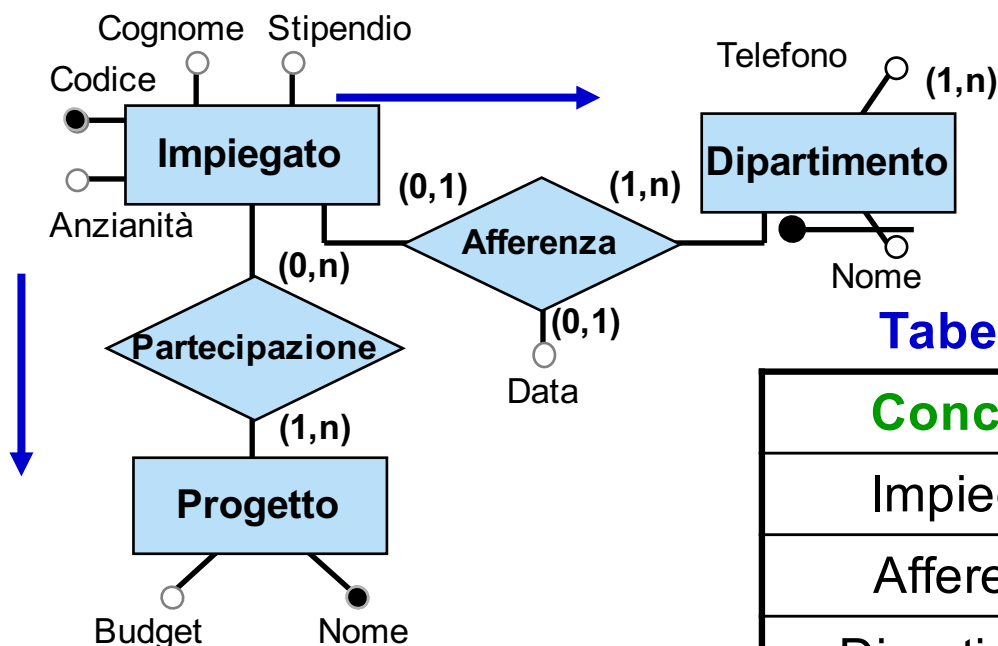




# Valutazione di costo

Per valutare il costo di un'operazione, si costruisce una **tabella degli accessi** basata su uno schema di navigazione associato all'operazione.

**Esempio:** trova tutti i dati di un impiegato, del dipartimento nel quale lavora e dei progetti ai quali partecipa (operazione 2).



**Tabella degli accessi dell'operazione 2**

Concetto	Costrutto	Accessi	Tipo
Impiegato	Entità	1	L
Afferenza	Relazione	1	L
Dipartimento	Entità	1	L
Partecipazione	Relazione	3	L
Progetto	Entità	3	L



## Valutazione dei costi e traduzione

- La traduzione dello schema concettuale in uno schema relazionale è guidata dall'**obiettivo di ottimizzare le prestazioni**.
- La valutazione delle prestazioni può essere effettuata adottando il modello di costo appena visto.
- Alcune scelte del processo di traduzione sono di fatto fisse e dettate dalla struttura dello schema ER (e quindi dalle scelte effettuate in fase di progettazione concettuale). In determinati casi invece il progettista deve effettuare delle scelte volte a ottimizzare le prestazioni.
- Vediamo una metodologia di progettazione, articolata in diverse fasi, nella quale i momenti in cui il progettista deve effettuare scelte progettuali sono chiaramente delimitati.



# Fasi della progettazione logica

## 1. Ristrutturazione dello schema ER:

- eliminazione (mantenendone la semantica) dei costrutti non direttamente traducibili nel modello relazionale
- scelta degli identificatori principali delle entità e delle relazioni

## 2. Traduzione diretta dello schema ER ristrutturato nel modello relazionale:

- la traduzione è **diretta**, nel senso che può essere eseguita automaticamente, non richiedendo scelte da parte del progettista
- Il risultato della traduzione diretta tiene conto delle scelte fatte in fase di progettazione concettuale
- lo schema relazionale prodotto **non** contiene ridondanze, se non quelle volute

## 3. Ristrutturazione dello schema relazionale:

- richiede delle scelte da parte del progettista, tenendo conto dell'ottimizzazione delle prestazioni (carico applicativo)



# 5. La progettazione logica

## 5.2 ristrutturazione dello schema ER

1. introduzione alla progettazione logica
- 2. ristrutturazione dello schema ER**
3. traduzione diretta nel modello relazionale
4. ristrutturazione dello schema logico



# Ristrutturazione dello schema ER

## Motivazioni:

- **semplificare** la successiva fase di traduzione nel modello relazionale eliminando quei costrutti non direttamente traducibili (mantenendone però la semantica)
- tenere conto di aspetti relativi all'**efficienza**

## Osservazione:

- uno schema ER **ristrutturato** è uno schema ER **degradato** opportunamente (cioè “monco” di alcuni costrutti) allo scopo di avvicinarsi al modello relazionale





## **Attività della ristrutturazione dello schema ER**

0. preparazione iniziale dello schema ER
1. analisi delle ridondanze
2. eliminazione degli attributi multivalore
3. eliminazione degli attributi composti
4. eliminazione delle ISA e delle generalizzazioni
5. scelta degli identificatori principali di entità e relazioni
6. specifica degli ulteriori vincoli esterni
7. riformulazione delle operazioni e delle specifiche sul carico applicativo in termini dello schema ristrutturato



## Ristrutturazione – attività 0: preparazione iniziale dello schema ER

- Lo scopo di questa fase è fare in modo che il progettista abbia il quadro completo di quanto rappresentato nello schema, in modo che nessun aspetto dello schema concettuale venga perso nella fase di progettazione logico.
- In questa fase viene quindi ridisegnato il diagramma dello schema ER aggiungendo al diagramma tutti i vincoli **rilevanti** che, essendo implicati da quelli presenti nello schema, non erano stati indicati esplicitamente nello schema concettuale.
- In particolare:
  - Per ogni entità E nello schema ER, i vincoli di identificazione rilevanti (anche quelli implicati da altri vincoli) per l'entità E vengono riportati nello schema ER
  - Per ogni relazione R nello schema ER, tutti i vincoli di identificazione rilevanti (anche quelli derivati da altri) per la relazione R, vengono riportati nello schema ER. Tipicamente, quello implicito (ovvero formato da tutti i ruoli della relazione) non viene riportato.

È importante osservare che nello schema ristrutturato non è necessario riportare tutti i vincoli implicati. È necessario però riportare tutti quelli che si reputano rilevanti, ovvero che non si vuole rischiare di ignorare per errore.



## Ristrutturazione – attività 1: analisi delle ridondanze

- Una **ridondanza** (estensionale) in uno schema ER è una informazione significativa, ma derivabile da altre.
- Sappiamo che le ridondanze, se presenti, sono documentate nello schema ER (ovvero espresse attraverso **vincoli**).
- In questa fase si decide se eliminare le ridondanze eventualmente presenti o mantenerle, e se introdurne delle nuove.
- **Vantaggi** nel mantenere una ridondanza:
  - potenziale maggiore efficienza nella esecuzione delle interrogazioni
- **Svantaggi** nel mantenere una ridondanza:
  - gestione dei vincoli aggiuntivi
  - appesantimento degli aggiornamenti
  - maggiore occupazione di spazio



## Analisi delle ridondanze

Abbiamo visto che le forme di ridondanza estensionale in uno schema ER sono date da:

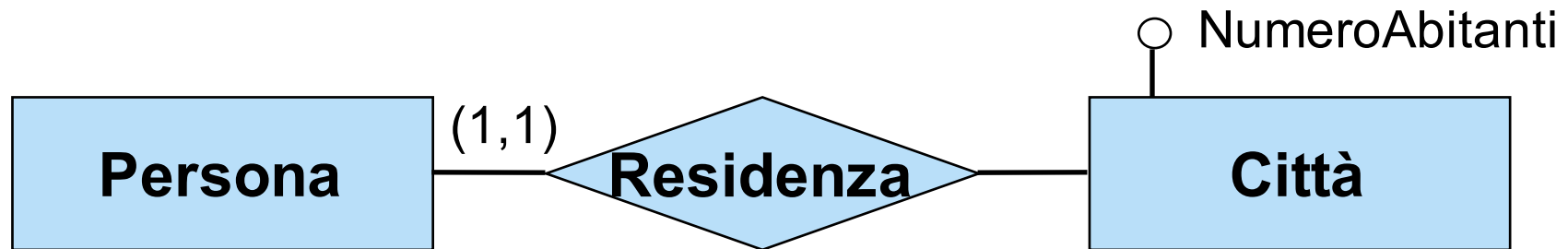
- attributi derivabili:
  - da altri attributi della stessa entità (o relazione)
  - da attributi di altre entità (o relazioni)
- relazioni derivabili dalla composizione di altre relazioni in presenza di cicli

Per ciascuna ridondanza bisogna valutare, in funzione del carico applicativo previsto (aggiornamenti, interrogazioni, occupazione di spazio) se è opportuno mantenerla oppure eliminarla.

Se si sceglie di mantenere la ridondanza, questa deve essere **documentata** (attraverso opportuni vincoli).



## Esempio di analisi delle ridondanze



### Tabella dei volumi

Concetto	Costrutto	Volume
Città	Entità	200
Persona	Entità	1.000.000
Residenza	Relazione	1.000.000

**Operazione 1:** memorizza una nuova persona con la relativa città di residenza (500 volte al giorno)

**Operazione 2:** stampa tutti i dati di una città, incluso il numero di abitanti (2 volte al giorno)



# Valutazione dei costi in assenza di ridondanza

Tabella degli accessi operazione 1

Concetto	Costrutto	Accessi	Tipo
Persona	Entità	1	S
Residenza	Relazione	1	S

Tabella degli accessi operazione 2

Concetto	Costrutto	Accessi	Tipo
Città	Entità	1	L
Residenza	Relazione	5000	L

## Costi:

- operazione 1: 1000 accessi in scrittura al giorno
- operazione 2: 10000 accessi in lettura al giorno

Contiamo doppi gli accessi in scrittura: **totale di 12000 accessi al giorno**



# Valutazione dei costi in presenza di ridondanza

## Tabella degli accessi operazione 1

Concetto	Costrutto	Accessi	Tipo
Persona	Entità	1	S
Residenza	Relazione	1	S
Città	Entità	1	S

(scrittura dati su persona)

(associazione città di res.)

(aggiornamento num. ab.)

## Tabella degli accessi operazione 2

Concetto	Costrutto	Accessi	Tipo
Città	Entità	1	L

### Costi:

- operazione 1: 1500 accessi in scrittura
- operazione 2: 2 accessi in lettura al giorno

Contiamo doppi gli accessi in scrittura: **totale di 3002 accessi al giorno**

**Conclusione: in questo esempio, manteniamo la ridondanza.**



## Ristrutturazione – attività 2: eliminazione di attributi multivalore

- Un **attributo multivalore** (ovvero un attributo con cardinalità massima maggiore di 1) non può essere tradotto direttamente nel modello relazionale senza introdurre delle ridondanze inaccettabili nelle relazioni ottenute.
- Dobbiamo quindi eliminare tutti gli attributi multivalore.
- L'eliminazione di un **attributo multivalore di un'entità** si effettua trasformando l'attributo in una relazione binaria, ed introducendo un'opportuna entità per il dominio (cfr. la parte sulla progettazione concettuale, esercizio 19a).
- L'eliminazione di un **attributo multivalore di una relazione** può essere realizzata mediante la preventiva trasformazione della relazione in un'entità (cfr. la parte sulla progettazione concettuale, esercizio 19c e 19d).

Nota: se trasformiamo una relazione R in un'entità, ed R era in ISA o in una gerarchia con altre relazioni, allora anche queste devono essere trasformate in entità (cfr. la parte sulla progettazione concettuale, esercizio 18c con ISA tra relazioni).

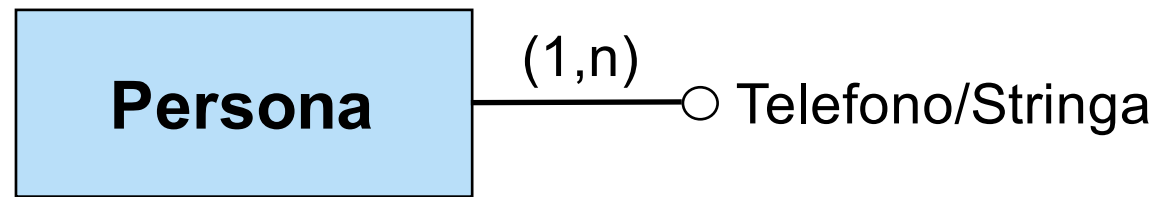




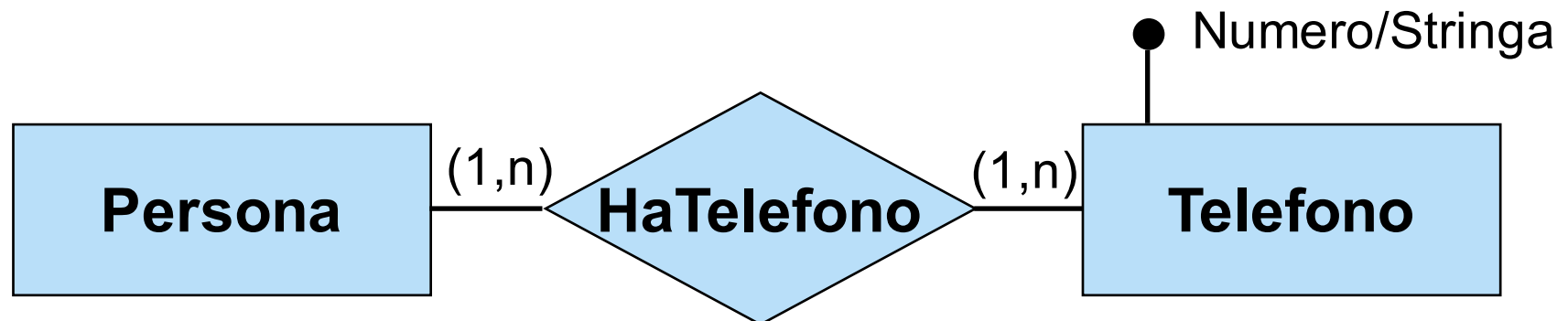
# Eliminazione di attributi multivalore di entità

Si trasforma l'attributo multivalore dell'entità in una relazione e il corrispondente dominio in entità.

*Esempio:*



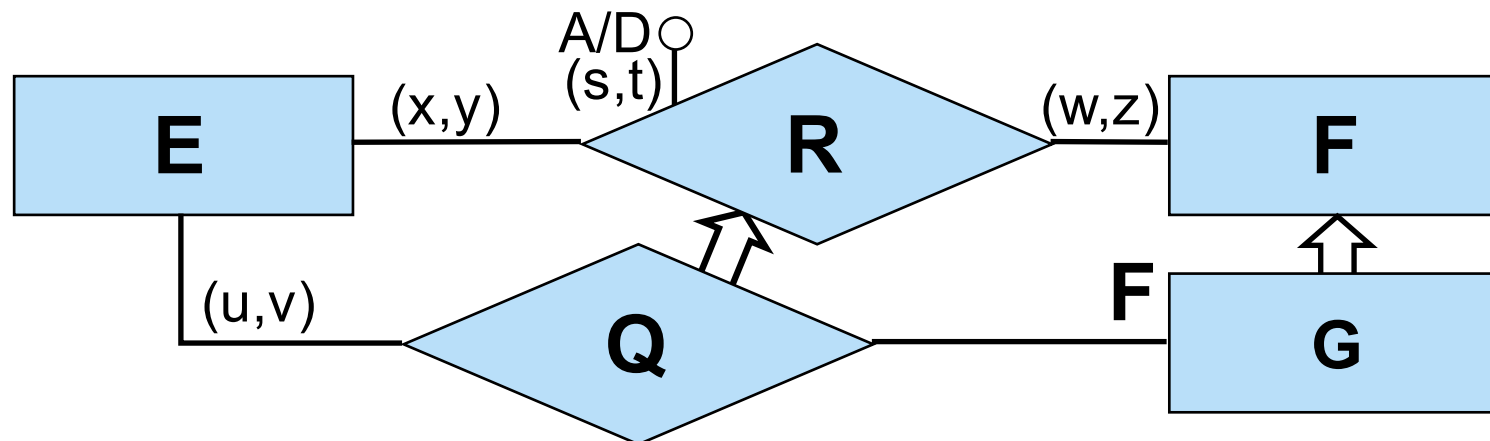
Si trasforma in:



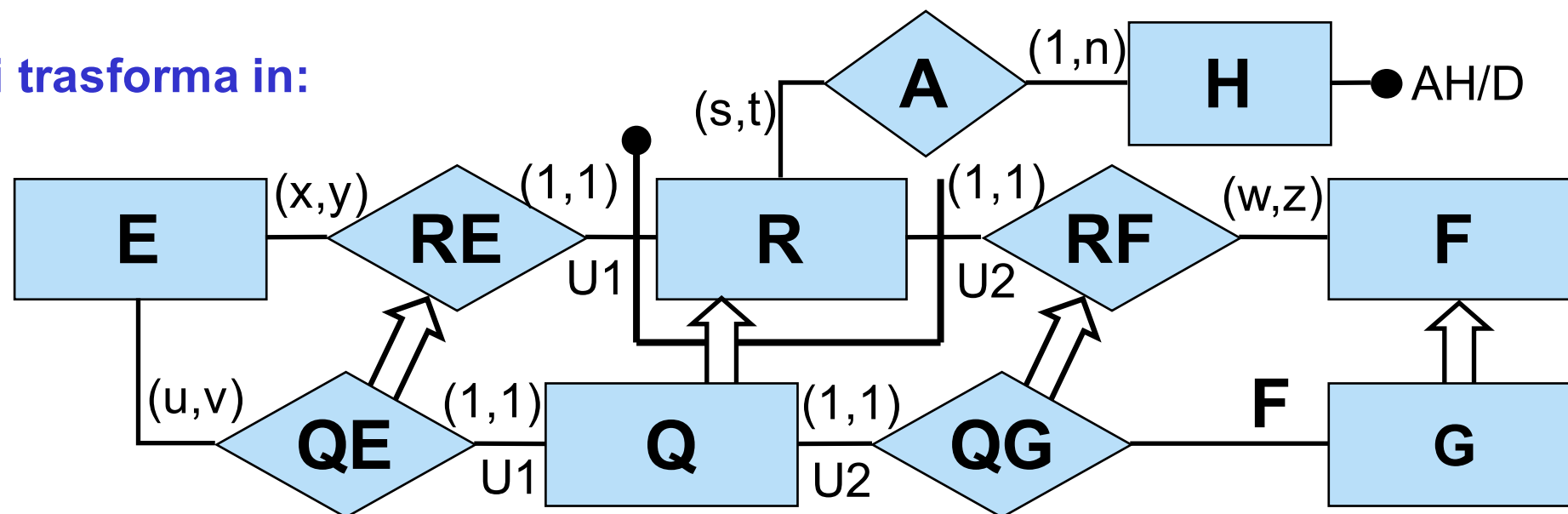


# Eliminazione di attributi multivalore di relazioni

Qualunque sia la cardinalità (s,t) dell'attributo si può trasformare la relazione R in entità e l'attributo multivalore di R in una relazione. Anche eventuali relazioni in ISA con R devono essere trasformate in entità.



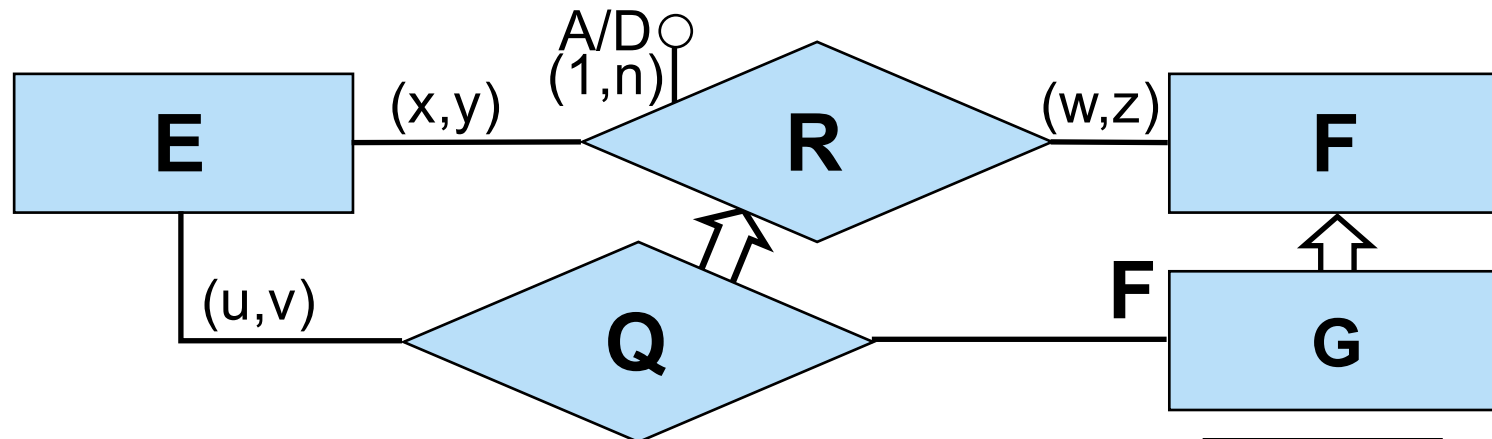
Si trasforma in:



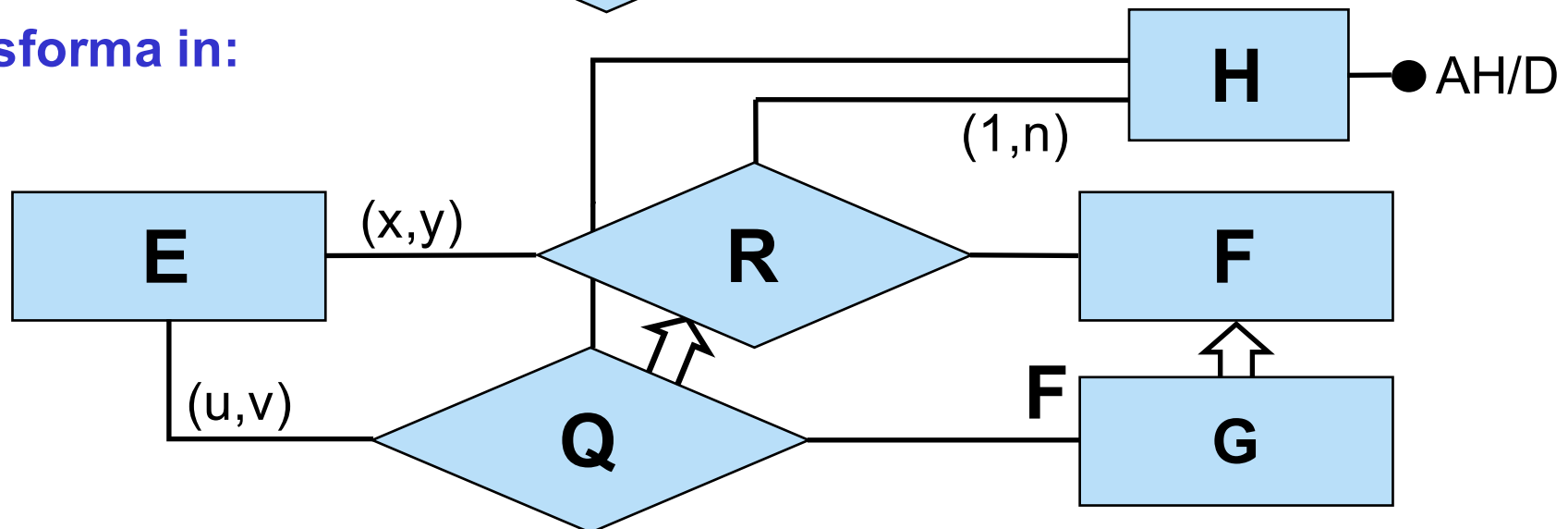


# Eliminazione di attributi multivalore di relazioni

Se la cardinalità (s,t) dell'attributo è (1,n), allora si può effettuare una diversa trasformazione: si trasforma l'attributo multivalore di R in un'entità che viene connessa alla relazione R (che quindi aumenta la arietà). Anche eventuali relazioni in ISA con R devono essere connesse alla nuova entità.



Si trasforma in:





## Ristrutturazione – attività 3: eliminazione di attributi composti

Un **attributo composto** di un'entità (o di una relazione) a questo punto ha cardinalità  $(1,1)$  oppure  $(0,1)$ .

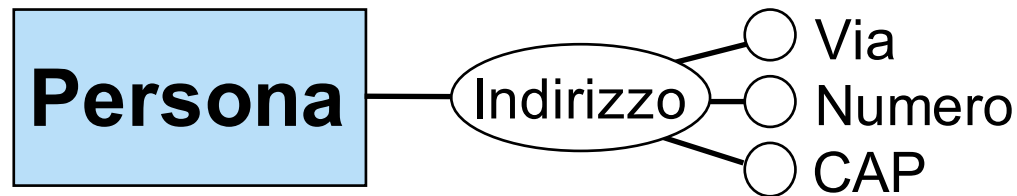
- Se l'attributo ha cardinalità  $(1,1)$ , si associano direttamente gli attributi componenti all'entità (o alla relazione).
  - Se l'attributo ha cardinalità  $(0,1)$  si può
    - procedere come per gli attributi  $(1,1)$ , ma con l'avvertenza che rappresentare l'opzionalità richiede un vincolo esterno
    - oppure trasformare l'attributo composto in una relazione binaria introducendo una nuova entità, come fatto per gli attributi multivalore (cfr. parte 4 esercizio 19b), mantenendo la cardinalità  $(0,1)$  sulla relazione.
- Ovviamente, se l'attributo composto è di una relazione, è necessario trasformare tale relazione in un'entità.



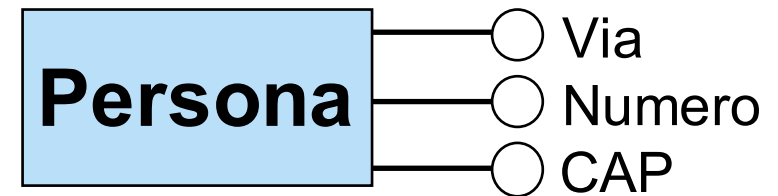
# Eliminazione di attributi composti: alternativa 1

Si associano direttamente gli attributi componenti all'entità (o alla relazione) a cui è associato l'attributo:

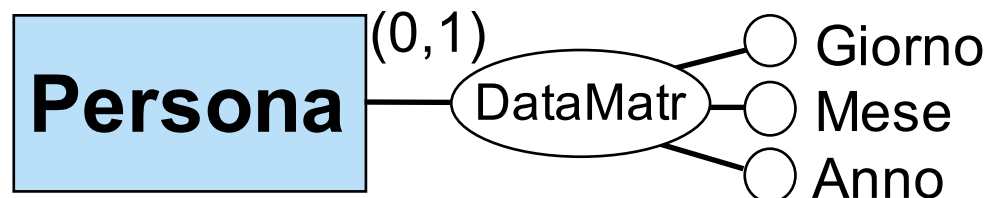
*Esempio:*



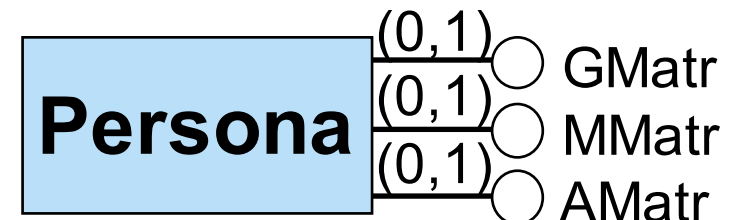
viene trasformato in:



*Esempio:*



viene trasformato in:



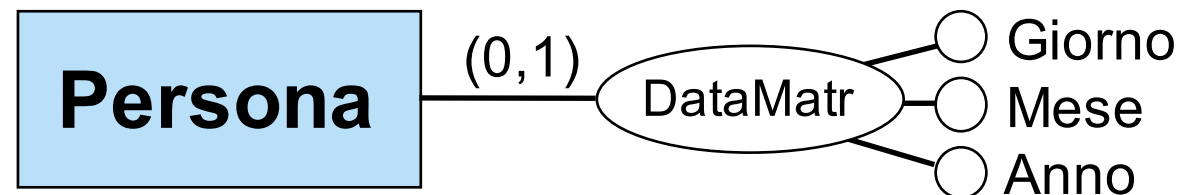
... con il vincolo esterno: *per ogni istanza di Persona, ciascun attributo tra GMatr, MMatr e AMatr è definito se e solo se lo sono anche gli altri due.*



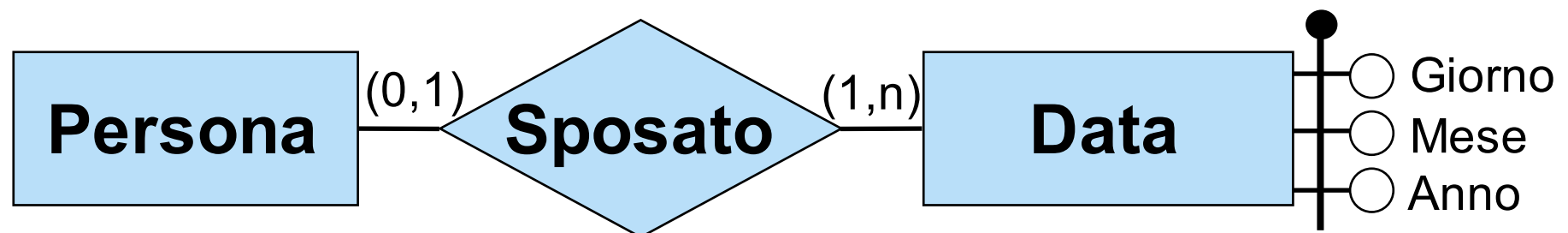
## Eliminazione di attributi composti: alternativa 2

Si trasforma l'attributo composto in una relazione binaria e si introduce una nuova entità.

*Esempio:*



Viene trasformato in:

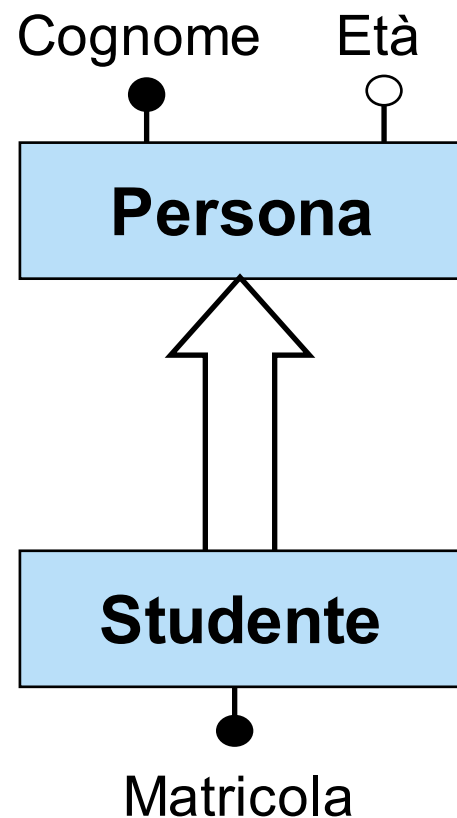




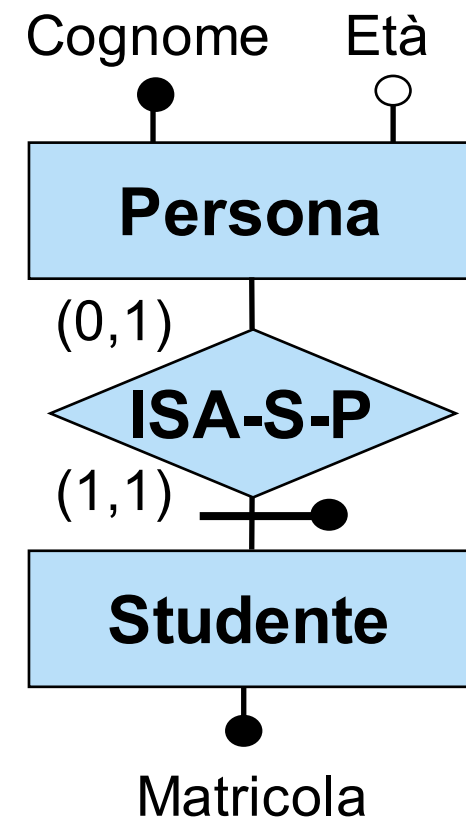
## Ristrutturazione – attività 4: eliminazione di ISA tra entità

Una relazione **E ISA F** tra due entità **E** ed **F** viene sostituita da una nuova relazione binaria **ISA-E-F** tra **E** ed **F** a cui **E** partecipa con cardinalità (1,1) e **F** con cardinalità (0,1). Agli eventuali identificatori di **E** viene aggiunto un identificatore esterno dato dalla partecipazione ad **ISA-E-F**.

*Esempio:*

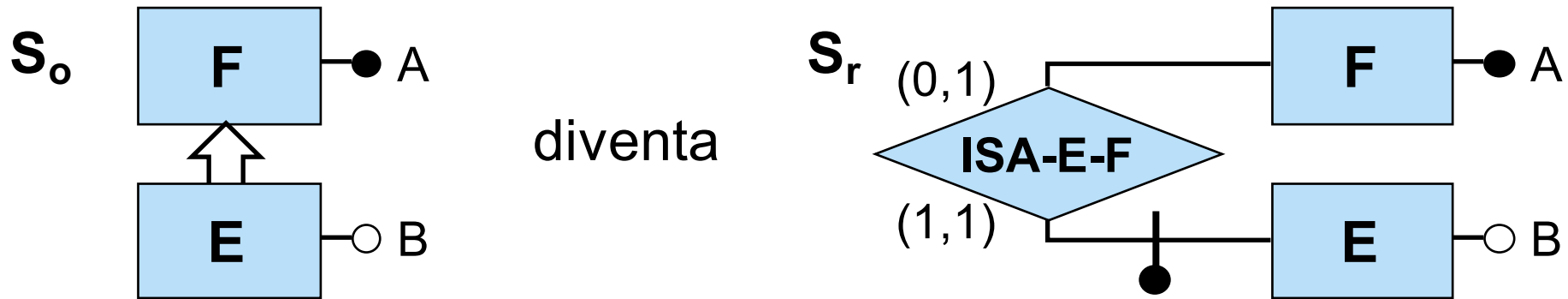


diventa





# Eliminazione di ISA tra entità: livello estensionale



## Istanza I di $S_o$ :

$istanze(I, F) = \{ f_1, f_2, f_3 \}$

$istanze(I, E) = \{ f_1, f_2 \}$

$istanze(I, A) = \{ (f_1, a_1), (f_2, a_2), (f_3, a_3) \}$

$istanze(I, B) = \{ (f_1, b_1), (f_2, b_2) \}$

## Istanza J di $S_r$ :

$istanze(J, F) = \{ f_1, f_2, f_3 \}$

$istanze(J, E) = \{ e_1, e_2 \}$

$istanze(J, A) = \{ (f_1, a_1), (f_2, a_2), (f_3, a_3) \}$

$istanze(J, B) = \{ (e_1, b_1), (e_2, b_2) \}$

$istanze(J, ISA-E-F) = \{ (E:e_1, F:f_1), (E:e_2, F:f_2) \}$

Si noti che nello schema  $S_r$  risultante dall'eliminazione delle ISA da  $S_o$ , **tutte le entità sono disgiunte a coppie**, e quindi non hanno più istanze in comune (su questo torneremo fra breve).

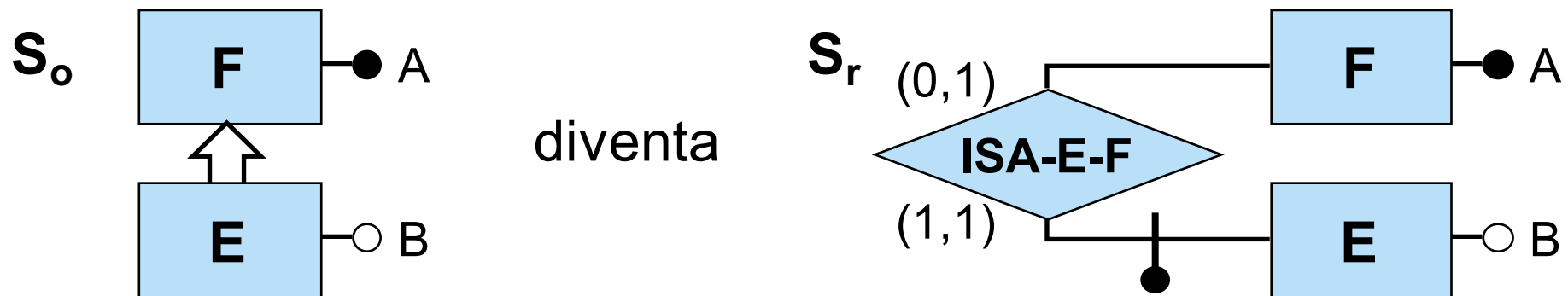




## Eliminazione di ISA tra entità: corrispondenza tra istanze

Esiste una stretta corrispondenza tra le istanze di uno schema  $S_o$  e le istanze dello schema  $S_r$  ottenuto da  $S_o$  eliminando le ISA tra entità.

*Mostriamo questa proprietà tramite un esempio:*

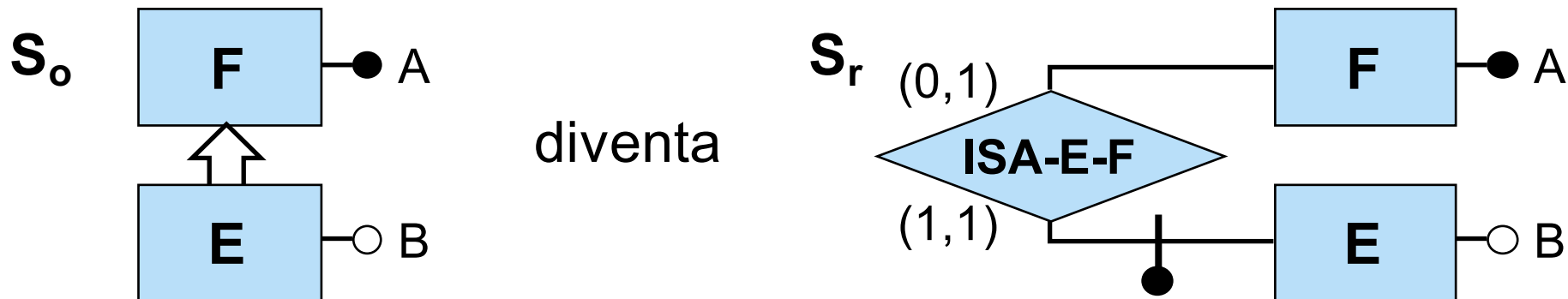


In particolare, mostriamo che esistono due funzioni  $g$  ed  $h$  tali che:

- $g$  è una funzione totale da  $istanze(S_o)$  a  $istanze(S_r)$
- $h$  è una funzione totale da  $istanze(S_r)$  a  $istanze(S_o)$
- per ogni istanza  $l_o$  di  $S_o$ , si ha che  $h(g(l_o)) = l_o$



## Corrispondenza tra istanze – funzione g



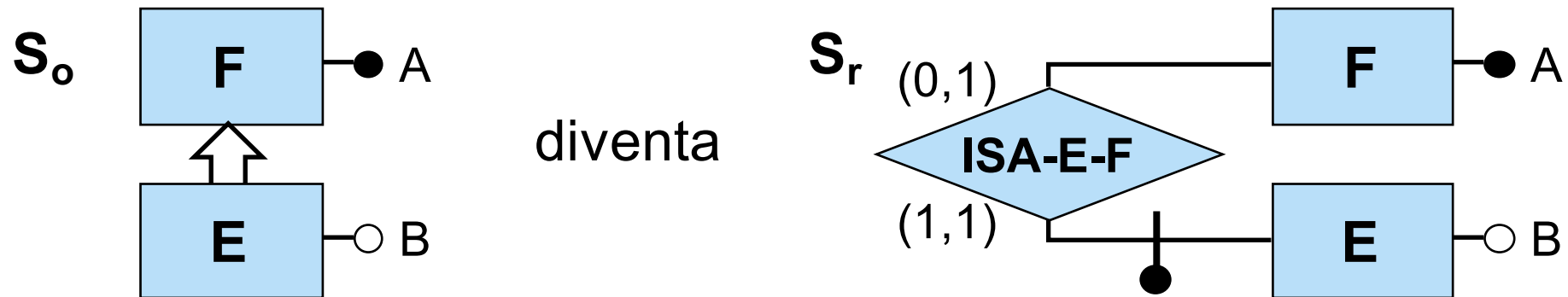
Definiamo la funzione  $g: \text{istanze}(S_o) \rightarrow \text{istanze}(S_r)$  in modo che, ad un'istanza  $I_o$  di  $S_o$ ,  $g$  assegni un'istanza  $I_r$  di  $S_r$  definita nel seguente modo:

- $\text{istanze}(I_r, F) = \text{istanze}(I_o, F)$
- $\text{istanze}(I_r, A) = \text{istanze}(I_o, A)$
- per definire  $\text{istanze}(I_r, E)$ , introduciamo in  $I_r$ , per ogni  $x \in \text{istanze}(I_o, E)$ , un nuovo oggetto  $g_E(x)$ , e definiamo  
 $\text{istanze}(I_r, E) = \{ g_E(x) \mid x \in \text{istanze}(I_o, E) \}$
- $\text{istanze}(I_r, \text{ISA-E-F}) = \{ (g_E(x), x) \mid x \in \text{istanze}(I_o, E) \}$
- $\text{istanze}(I_r, B) = \{ (g_E(x), b) \mid (x, b) \in \text{istanze}(I_o, B) \}$

È facile verificare che  $I_r$  così definita è effettivamente un'istanza di  $S_r$ .



## Corrispondenza tra istanze – funzione h



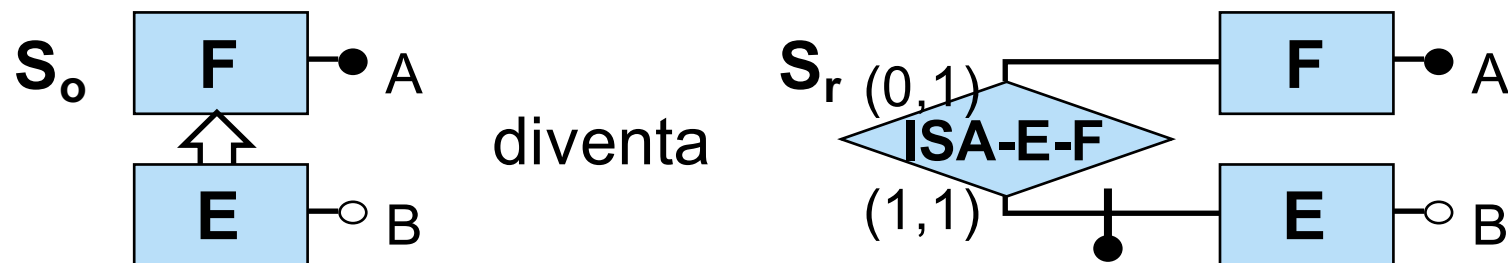
Definiamo la funzione  $h$ :  $istanze(S_r) \rightarrow istanze(S_o)$  in modo che, ad un'istanza  $I_r$  di  $S_r$ ,  $h$  assegni l'istanza  $I_o$  di  $S_o$  definita nel seguente modo:

- $istanze(I_o, F) = istanze(I_r, F)$
- $istanze(I_o, A) = istanze(I_r, A)$
- $istanze(I_o, E) = \{ x \in istanze(I_r, F) \mid \text{esiste un } y \in istanze(I_r, E) \text{ con } (y, x) \in istanze(I_r, ISA-E-F) \}$
- $istanze(I_o, B) = \{ (x, b) \mid x \in istanze(I_o, E), (y, x) \in istanze(I_r, ISA-E-F) \text{ e } (y, b) \in istanze(I_r, E) \}$

È facile verificare che  $I_o$  così definita è effettivamente un'istanza di  $S_o$ , e che inoltre  $h(g(I_o)) = I_o$ .



## Osservazione sullo schema risultante



### Istanza I di $S_o$ :

$istanze(I, F) = \{ f_1, f_2, f_3 \}$   
 $istanze(I, E) = \{ f_1, f_2 \}$   
 $istanze(I, A) = \{ (f_1, a_1), (f_2, a_2), (f_3, a_3) \}$   
 $istanze(I, B) = \{ (f_1, b_1), (f_2, b_2) \}$

### Istanza J di $S_r$ :

$istanze(J, F) = \{ f_1, f_2, f_3 \}$   
 $istanze(J, E) = \{ e_1, e_2 \}$   
 $istanze(J, A) = \{ (f_1, a_1), (f_2, a_2), (f_3, a_3) \}$   
 $istanze(J, B) = \{ (e_1, b_1), (e_2, b_2) \}$   
 $istanze(J, ISA-E-F) = \{ (E:e_1, F:f_1), (E:e_2, F:f_2) \}$

Come osservato prima, nello schema  $S_r$  risultante dall'eliminazione delle ISA da  $S_o$ , **tutte le entità sono disgiunte a coppie**.

Come si concilia questa osservazione con il fatto che nello schema originario ciò non era vero? La risposta sta nelle funzioni g ed h illustrate precedentemente. Tramite queste funzioni è possibile infatti stabilire se due qualunque istanze di entità nello schema risultante corrispondono ad un'unica istanza di entità nello schema originario (ad es.,  $e_1$  ed  $f_1$  nella istanza di  $S_r$  corrispondono entrambi ad  $f_1$  nella istanza di  $S_o$ ).



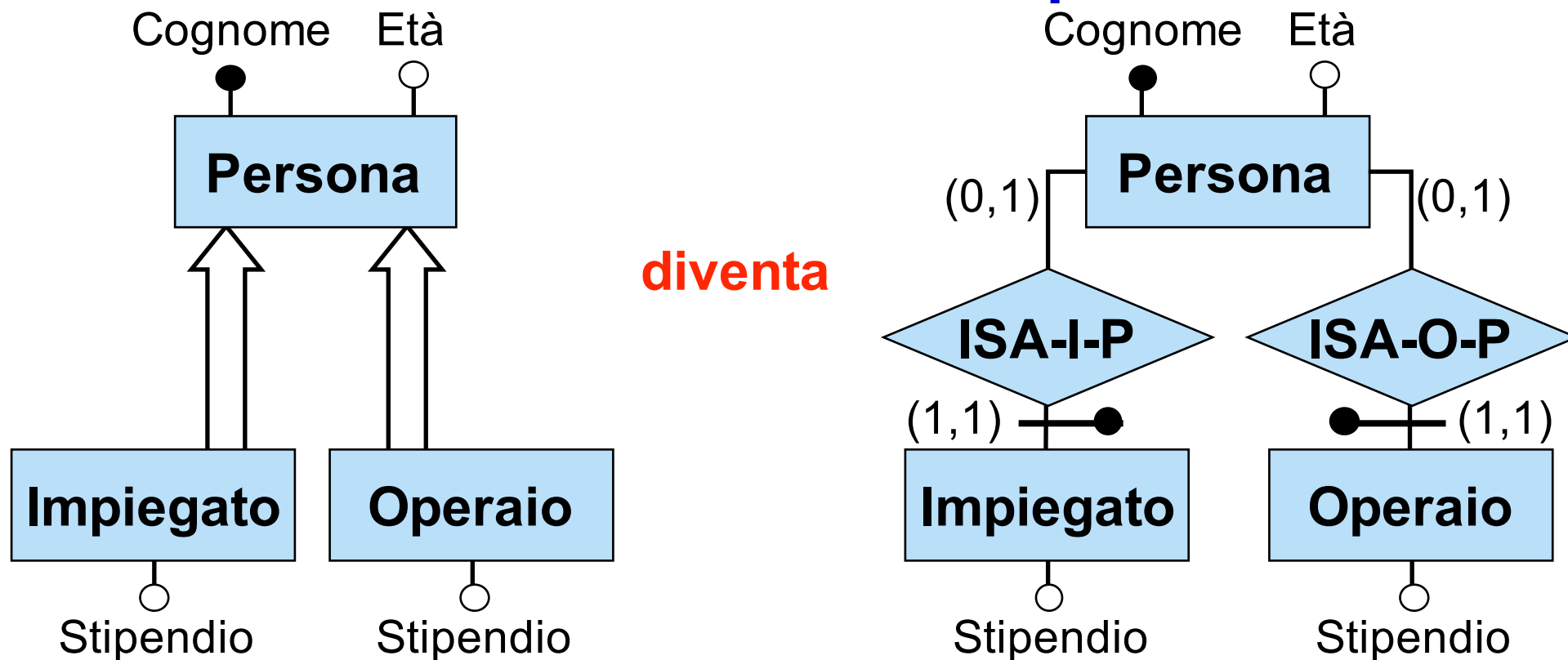
## Attributi in comune nella eliminazione di ISA tra entità

Se due entità non disgiunte nello schema originario hanno un attributo  $A$  in comune, applicando la trasformazione per eliminare la relazione ISA, nelle due corrispondenti entità disgiunte dello schema risultante troviamo due attributi di nome  $A$ , che di fatto rappresentano due funzioni (o relazioni, se  $A$  è multivalore) diverse.

Al fine di mantenere la semantica originaria, dobbiamo quindi imporre che le due funzioni, quando applicate a due oggetti  $e$ ,  $f$  dello schema risultante che corrispondono allo stesso oggetto nello schema di partenza, assegnino ad  $e$  ed  $f$  lo stesso valore. Ciò viene fatto con un **vincolo esterno sullo schema ristrutturato**.



## Attributi in comune nella eliminazione di ISA tra entità: esempio



diventa

con il **vincolo esterno** nello schema risultante:

per ogni istanza  $I$  dello schema, per ogni  $p \in \text{istanze}(I, \text{Persona})$ , se esistono  $i \in \text{istanze}(I, \text{Impiegato})$ , e  $o \in \text{istanze}(I, \text{Operaio})$  tali che  $(i, p) \in \text{istanze}(I, \text{ISA-I-P})$  e  $(o, p) \in \text{istanze}(I, \text{ISA-O-P})$ , allora esiste  $s$  tale che  $\langle i, s \rangle \in \text{Istanze}(I, \text{Stipendio})$  e  $\langle o, s \rangle \in \text{Istanze}(I, \text{Stipendio})$

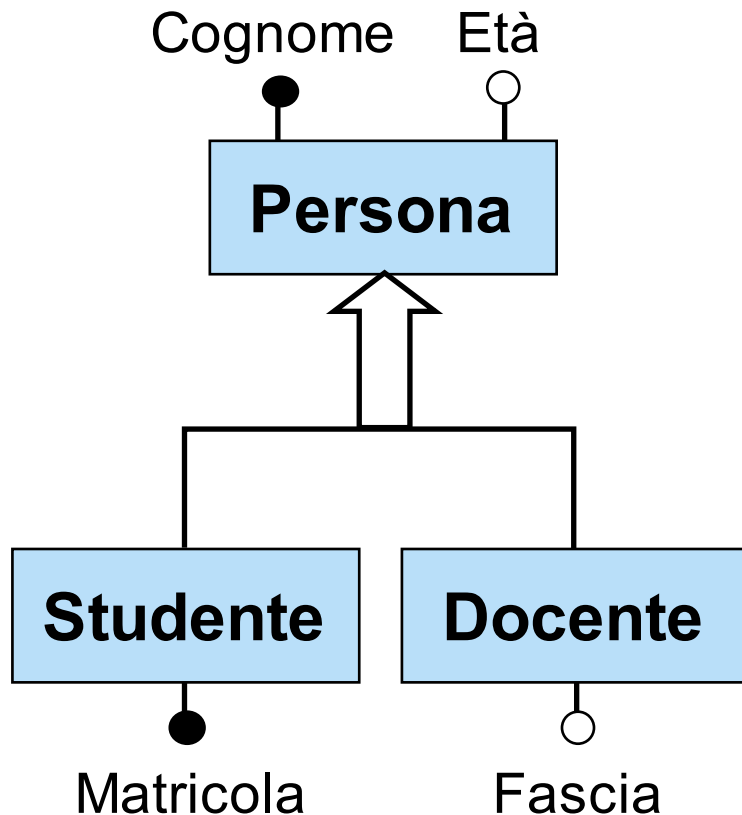


## Ristrutturazione – attività 4: eliminazione di generalizzazioni tra entità

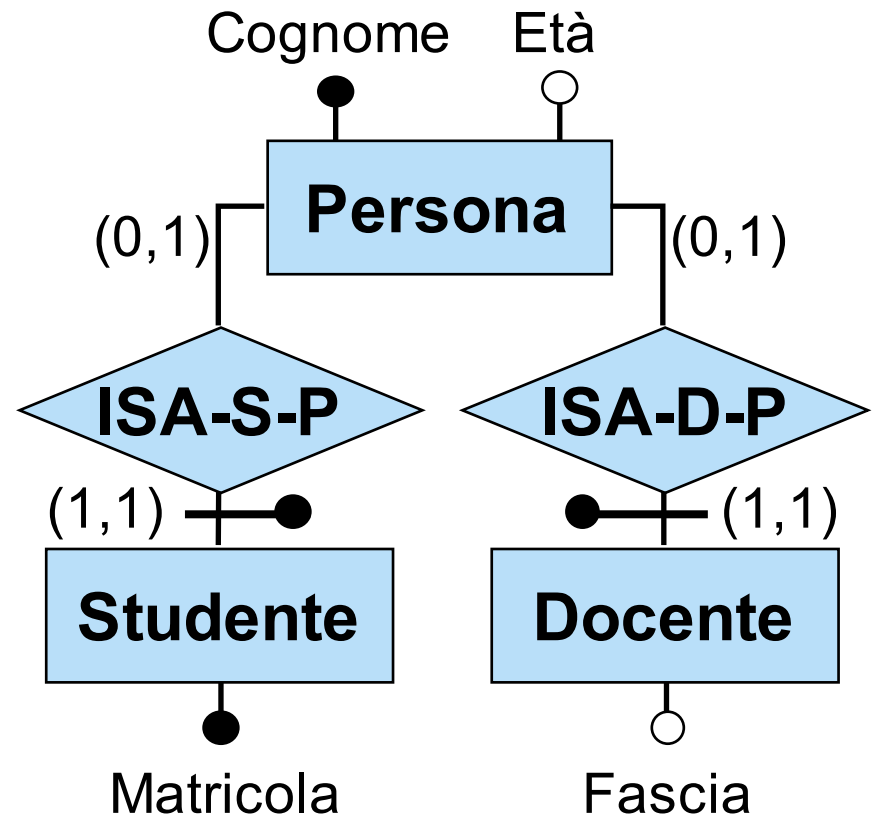
- Una generalizzazione tra una entità padre **F** e le sottoentità **E<sub>1</sub>, E<sub>2</sub>, ..., E<sub>n</sub>**, viene trattata come n relazioni **E<sub>1</sub> ISA F, ..., E<sub>n</sub> ISA F**, introducendo n relazioni binarie **ISA-E<sub>1</sub>-F, ..., ISA-E<sub>n</sub>-F**.
- Per tenere conto delle proprietà delle generalizzazioni si aggiungono opportuni vincoli esterni, detti **vincoli di generalizzazione**:
  - il fatto che per ogni istanza **I** dello schema di partenza valga la proprietà  $istanze(I, E_i) \cap istanze(I, E_k) = \emptyset$ , per ogni  $1 \leq i, k \leq n, i \neq k$  viene colto dal vincolo:  
*per ogni coppia (E<sub>i</sub>, E<sub>j</sub>) di entità figlie, ogni istanza di F partecipa al più ad una delle relazioni ISA-E<sub>i</sub>-F, ISA-E<sub>j</sub>-F*
  - se la **generalizzazione** è **completa**, vale l'ulteriore proprietà  $istanze(I, E_1) \cup \dots \cup istanze(I, E_n) = istanze(I, F)$  nello schema di partenza e quindi il vincolo di generalizzazione diventa:  
*ogni istanza di F partecipa esattamente ad una delle relazioni ISA-E<sub>1</sub>-F, ..., ISA-E<sub>n</sub>-F*



# Eliminazione di generalizzazioni tra entità: esempio



diventa



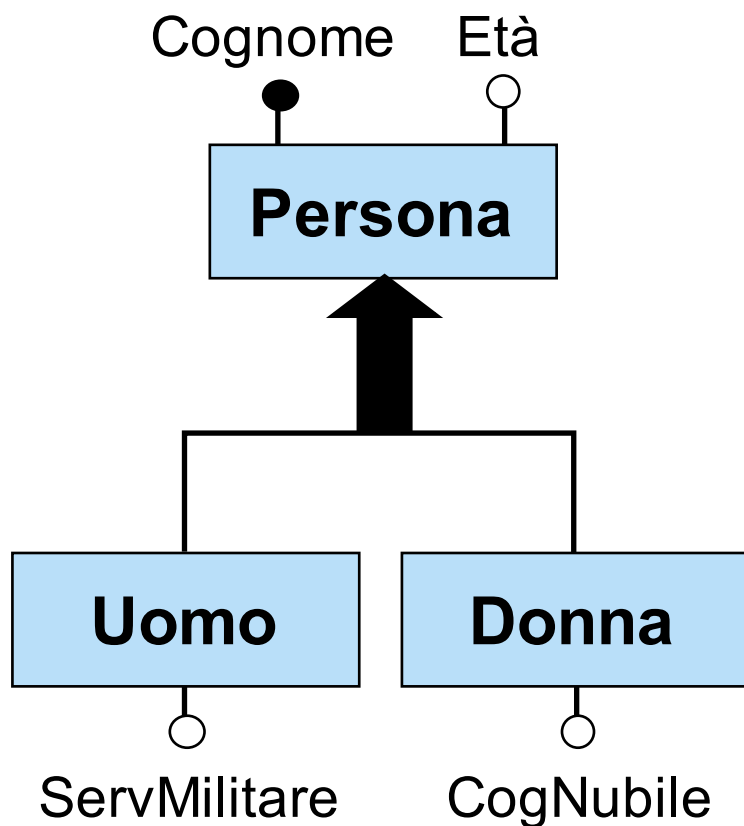
## Vincoli di generalizzazione:

nessuna istanza di Persona partecipa sia a ISA-S-P  
sia a ISA-D-P

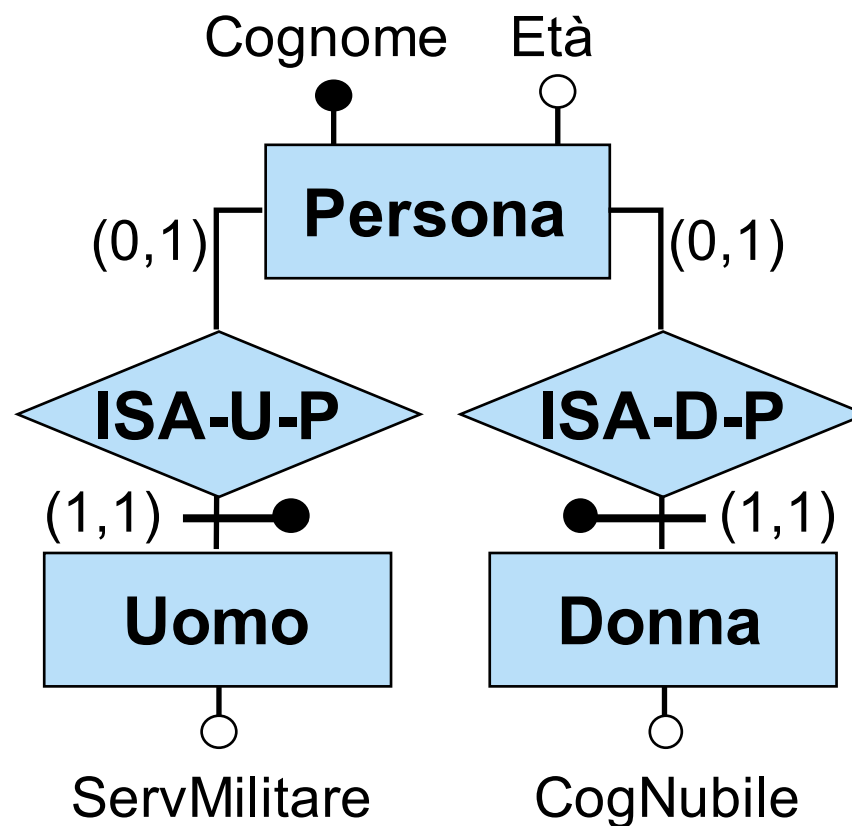




## Eliminazione di generalizzazioni complete: esempio



diventa



### Vincoli di generalizzazione:

ogni istanza di Persona partecipa ad ISA-U-P oppure ad ISA-D-P, ma non ad entrambi

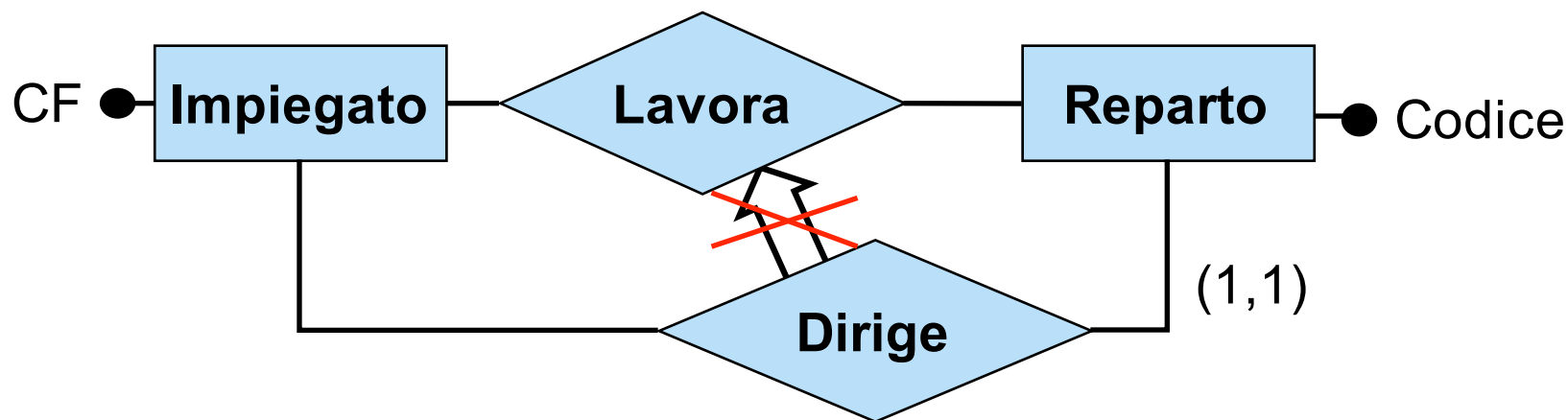


## Ristrutturazione – attività 4: eliminazione di ISA e generalizzazioni tra relazioni

Le **relazioni ISA e le generalizzazioni tra relazioni** vengono eliminate dallo schema e vengono espresse tramite opportuni vincoli esterni.

Nel caso in cui le relazioni in ISA (o nella generalizzazione) insistano su esattamente le **stesse entità per tutti i ruoli**, è immediato esprimere il vincolo esterno.

*Esempio:*



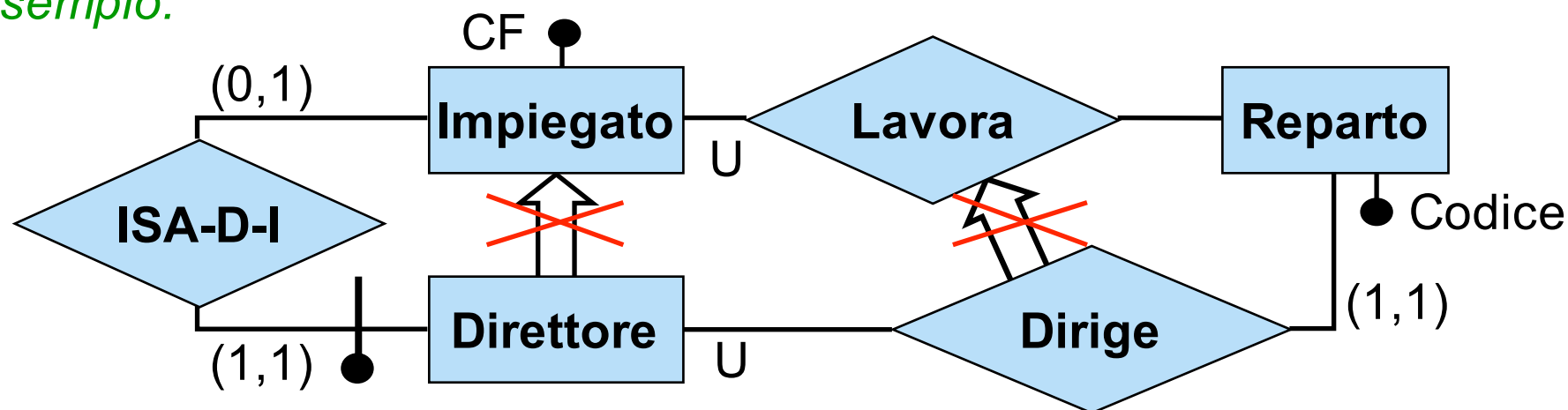
**Vincolo esterno:** ogni istanza di **Dirige** è anche un'istanza di **Lavora**.



# Eliminazione di ISA e generalizzazioni tra relazioni

Nel caso in cui le relazioni in ISA (o nella generalizzazione) insistano su **entità diverse in qualche ruolo**, nell'esprimere il vincolo esterno bisogna tenere conto che nello schema ristrutturato entità diverse sono tra loro disgiunte.

*Esempio:*



**Vincolo esterno:** In ogni istanza  $I$  dello schema, per ogni istanza  $(U:d, Reparto:r)$  di *Dirige* in  $I$ , detta  $i$  l'istanza di *Impiegato* tale che  $(Direttore:d, Impiegato:i)$  è un'istanza di *ISA-D-I* in  $I$  (si noti che  $i$  esiste sempre ed è unica), si ha che  $(U:i, Reparto:r)$  è un'istanza di *Lavora* in  $I$ .



## Ristrutturazione – attività 5: scelta degli identificatori principali

1. Scelta dell'identificatore principale di ogni **entità**
2. Scelta dell'identificatore principale di ogni **relazione**

Ogni identificatore principale di entità o relazione deve essere **essenziale**. Ricordiamo che un identificatore di  $W$  (entità o relazione) si dice **essenziale** se ogni insieme ottenuto da  $W$  togliendo un elemento non è più un identificatore per  $W$ .



## Ristrutturazione – attività 5: scelta degli identificatori principali delle entità

Per **ogni entità** è necessario:

- individuare almeno un identificatore (se una entità non ne ha nessuno, occorre concordare con gli esperti del dominio un nuovo attributo – tipicamente un codice – che possa agire da identificatore)
- scegliere tra gli identificatori dell'entità un **identificatore principale**.

**Criteri per la scelta** dell'identificatore principale di entità:

- essenzialità (ogni identificatore principale deve essere essenziale)
- semplicità (cioè formato da pochi elementi)
- utilizzo nelle operazioni più frequenti o importanti
- preferenza per gli identificatori interni
- tra quelli identificatori esterni, privilegiare quelli sulle relazioni provenienti dalla eliminazione della relazione ISA
- se per un'entità nessuno degli identificatori soddisfa tali requisiti, è possibile introdurre un ulteriore attributo dell'entità (un **codice**, i cui valori sono speciali ed hanno l'unico scopo di identificare le istanze dell'entità).

In una entità con più identificatori, quello principale viene indicato nella documentazione associata allo schema ristrutturato. Nel diagramma dello schema ristrutturato, in presenza di più identificatori per un'entità, **denoteremo quello principale con un cerchio aggiuntivo intorno al pallino dell'identificatore**.



## Cicli di identificazione esterna su entità

Nella scelta degli identificatori principali delle entità è necessario fare attenzione a non introdurre **cicli tra gli identificatori principali esterni**.

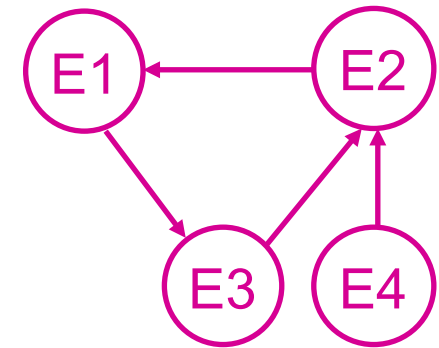
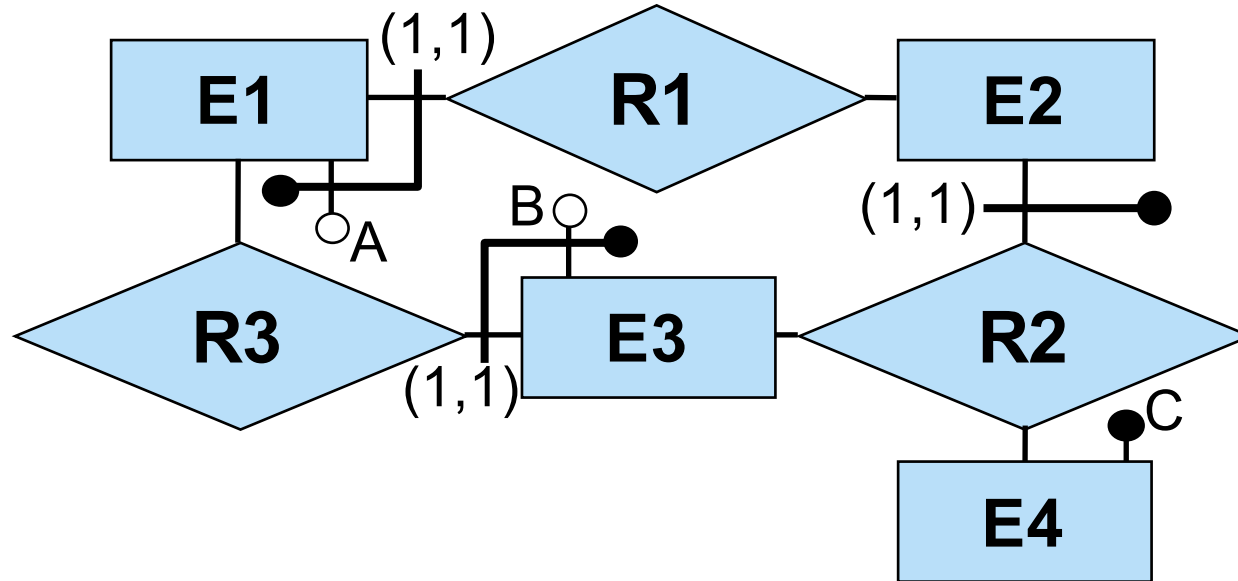
Definiamo il **grafo degli identificatori (principali) esterni** nel seguente modo:

- ad ogni entità del diagramma corrisponde un nodo
- c'è un arco dall'entità E all'entità F se e solo se E partecipa ad una relazione che è parte dell'identificatore (o che è identificatore) principale esterno di F.

Si ha un ciclo di identificazione esterna quando il grafo degli identificatori principali esterni contiene un ciclo.

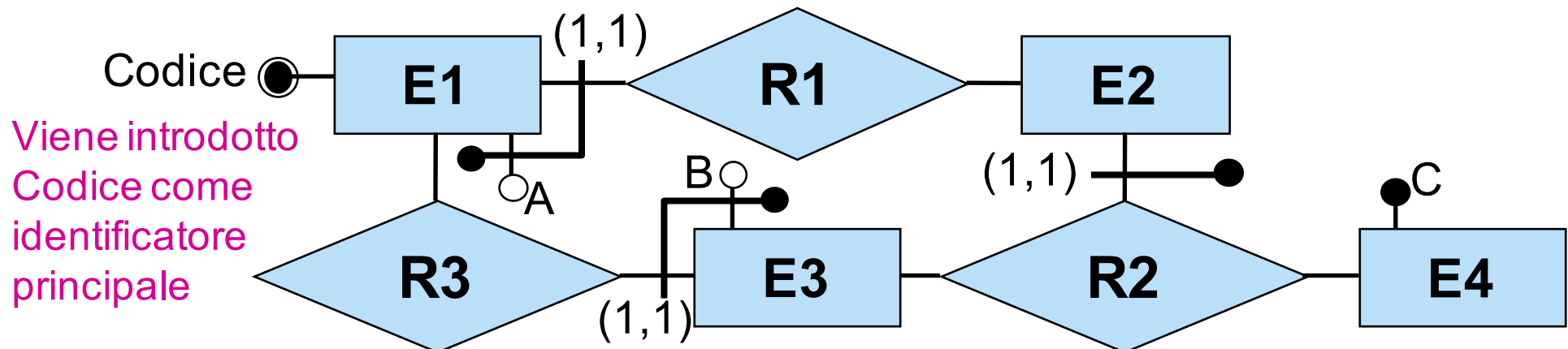


## Cicli di identificazione esterna



grafo degli identificatori principali esterni

È necessario **spezzare i cicli di identificazione esterna** scegliendo per almeno una entità nel ciclo un identificatore principale diverso. Se non ci sono alternative, è necessario introdurre un opportuno codice.

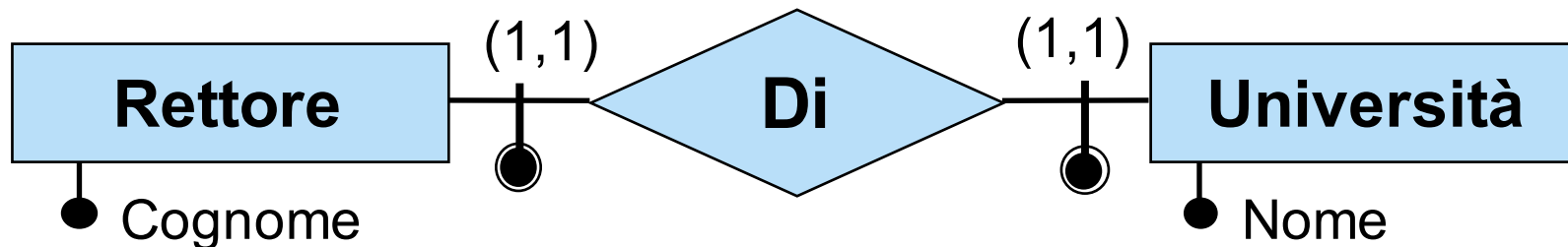




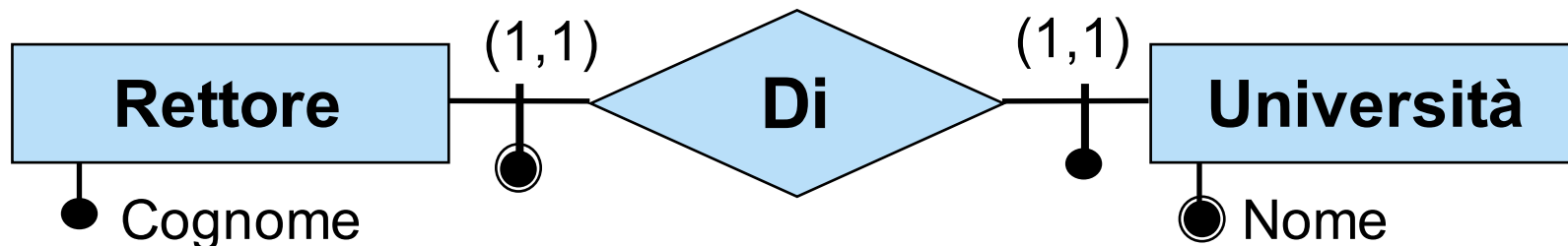
## Cicli di identificazione esterna: esempio

Un caso significativo di ciclo di identificazione esterna è dato da due entità che si identificano a vicenda.

*Esempio:*



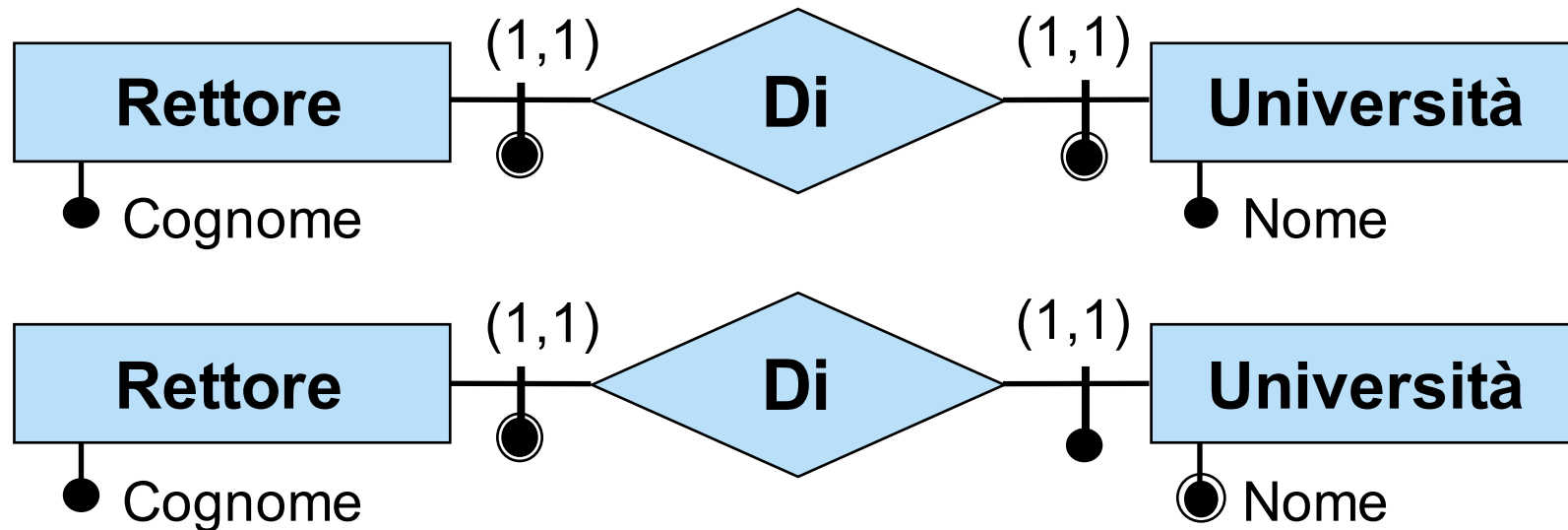
Abbiamo un ciclo di identificazione esterna, che deve essere spezzato. Una possibilità è la seguente:







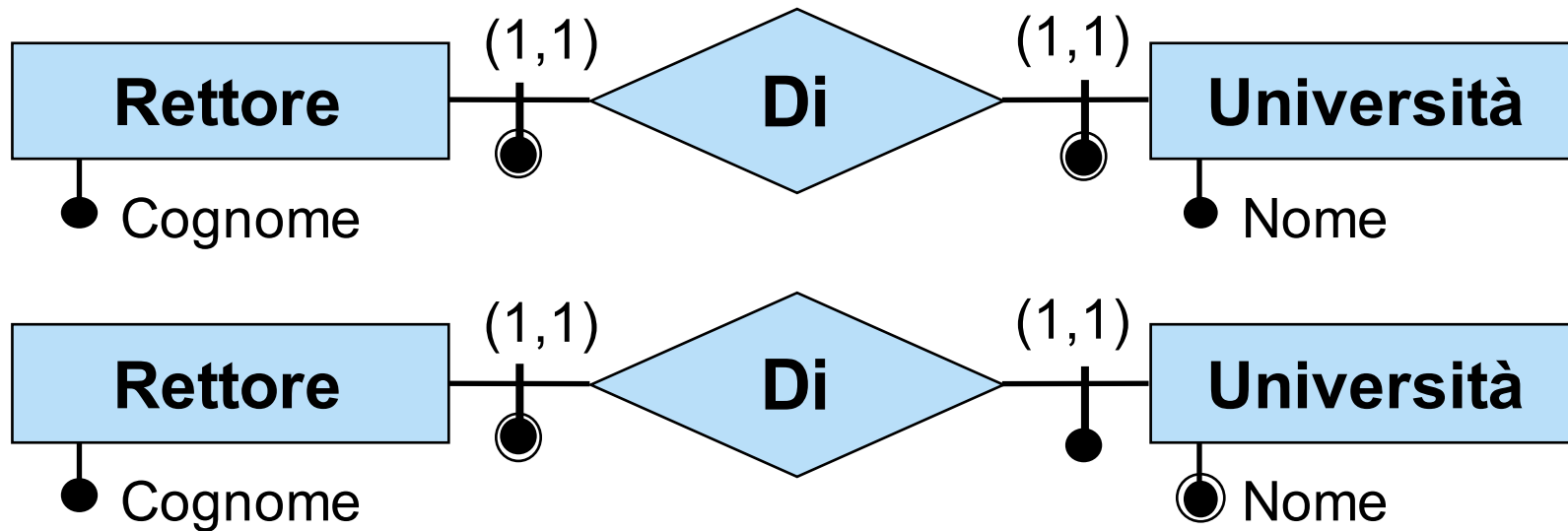
## Esercizio 2: cicli di identificazione esterna



- Che differenza c'è tra le istanze dei due schemi?
- Perché è necessario spezzare i cicli di identificazione esterna?



## Esercizio 2: soluzione



*I due schemi hanno le stesse istanze, in quanto la scelta degli identificatori principali non ha portato all'introduzione di nuovi attributi.*

*I cicli di identificazione esterna non rappresentano alcun problema per quanto riguarda lo schema concettuale.*

*È però necessario spezzarli perché renderebbero impossibile la traduzione nel modello relazionale.* Vedremo infatti che, se un'entità *E* ha un identificatore principale esterno su una relazione *R*, nello schema relazionale prodotto, ad *E* corrisponderà una relazione in cui viene accorpata la relazione *R*, con tutti gli identificatori delle entità che partecipano ad *R*. In presenza di cicli di identificazione esterna questo accorpamento non è fattibile.



## Ristrutturazione – attività 5: scelta degli identificatori principali delle relazioni

Per **ogni relazione che non partecipa ad un identificatore principale di entità** è necessario scegliere, tra i suoi identificatori, l'**identificatore principale**.

**Criteri per la scelta** dell'identificatore principale di relazione:

- essenzialità (ogni identificatore principale deve essere essenziale)
- semplicità (cioè formato da pochi elementi),
- utilizzo nelle operazioni più frequenti o importanti,
- sulle relazioni provenienti dalla eliminazione della relazione ISA, privilegiare quello sul ruolo corrispondente alla entità figlia.

In una relazione con più identificatori, quello principale viene indicato nella documentazione associata allo schema ristrutturato. Nel diagramma dello schema ristrutturato, in presenza di più identificatori per una relazione, **denoteremo quello principale con un cerchio aggiuntivo intorno al pallino dell'identificatore**.



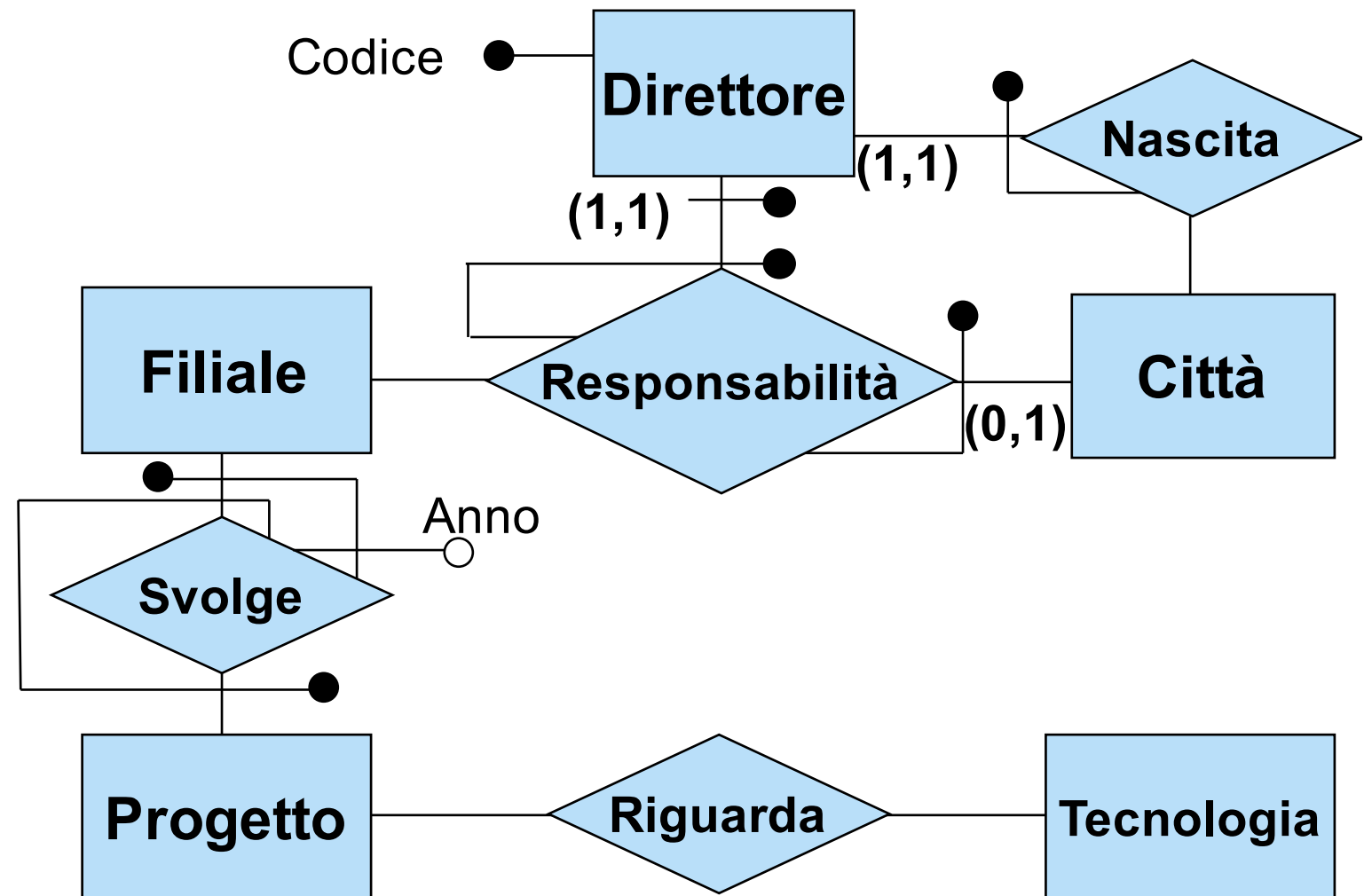
# Scelta degli identificatori principali delle relazioni

1. Se una relazione R partecipa ad un identificatore principale esterno dell'entità E, allora **non occorre scegliere l'identificatore principale di R**, perché di fatto le sue istanze sono in corrispondenza biunivoca con le istanze di E, e quindi l'identificatore di E può essere visto già come un identificatore per le istanze di R.
2. Se una relazione R non ha alcun identificatore esplicitamente indicato nel diagramma dello schema concettuale, e quindi ha solo l'identificatore implicito (ovvero quello formato da tutti i ruoli di R), allora **tale identificatore implicito diventa l'identificatore principale della relazione R** (ed in questo caso non lo indichiamo neppure nel diagramma dello schema ristrutturato)
3. Se una relazione non partecipa ad alcun identificatore principale esterno di entità, ed ha almeno un identificatore indicato esplicitamente, allora se ne deve scegliere uno come identificatore principale, al limite scegliendo quello implicito, **che in questo caso andrà indicato eccezionalmente nel diagramma dello schema ristrutturato**.



## Esempio: identificatori principali delle relazioni

Consideriamo il seguente schema ristrutturato e supponiamo che le operazioni usino principalmente il codice per accedere ai direttori ed usino principalmente le città per accedere a Responsabilità.

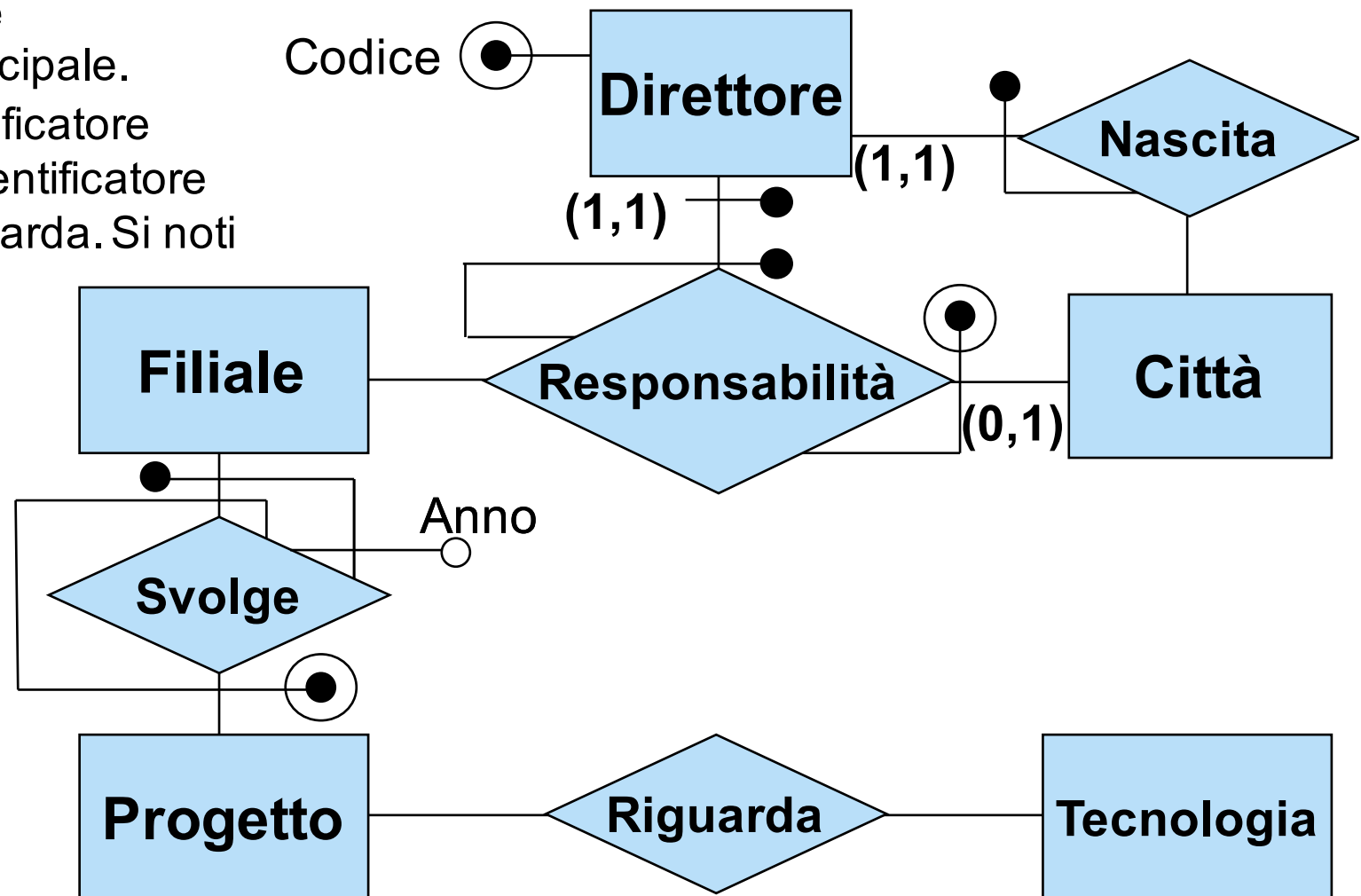




## Esempio: identificatori principali delle relazioni

Scegliamo gli identificatori principali Codice per Direttore e Città per Responsabilità. Scegliamo l'identificatore implicito come identificatore principale di Svolge; esso viene indicato nel diagramma dello schema ristrutturato perché altrimenti lasceremmo indicato il solo identificatore (Anno, Filiale) che quindi sarebbe interpretato come Identificatore principale.

Scegliamo l'identificatore implicito come identificatore principale di Riguarda. Si noti che nel caso della relazione Riguarda non serve indicare l'identificatore nel diagramma dello schema ristrutturato, perché non ci sono identificatori espliciti per tale relazione.





## Ristrutturazione – attività 6: specifica degli ulteriori vincoli esterni

- È necessario riformulare tutti i vincoli esterni dello schema originario in termini dello schema ristrutturato.
  - mettere in evidenza anche i vincoli impliciti (ovvero che sono conseguenza di altri vincoli), se non è ancora stato fatto

Es. vincoli di identificazione esterna per relazioni (1,1) – (1,1) oppure (1,1) – (0,1)
- Si devono aggiungere i vincoli derivanti dalla ristrutturazione:
  - vincoli derivanti da attributi composti opzionali
  - vincoli per due entità che erano in ISA con una stessa entità padre e che hanno attributi in comune
  - vincoli di generalizzazione (disgiuntezza, completezza)
  - vincoli dovuti agli identificatori non principali (se non sono più rappresentati nello schema)



## **Ristrutturazione – attività 7: riformulazione di operazioni e carico applicativo**

- È necessario riformulare le operazioni e i relativi schemi di navigazione in termini dello schema ristrutturato.
- È necessario riformulare le specifiche sul carico applicativo in termini dello schema ristrutturato.



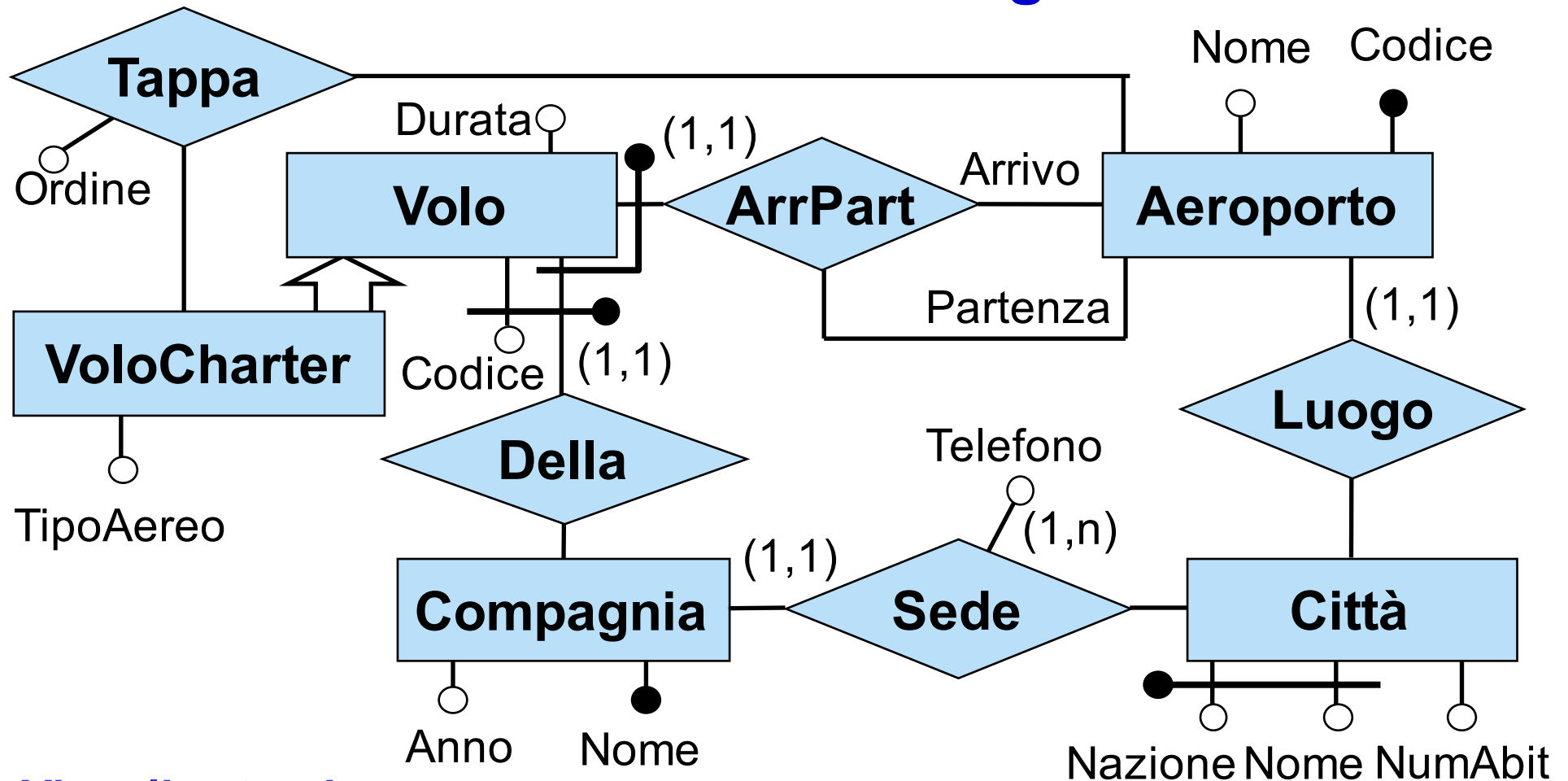


## Riassunto sulla ristrutturazione

0. Preparazione iniziale dello schema ER
1. Analisi delle ridondanze (si tiene conto dell'efficienza)
2. Eliminazione degli attributi multivalore
3. Eliminazione degli attributi composti (eventuale vincolo (0,1) diventa un vincolo esterno sugli attributi componenti)
4. Eliminazione delle ISA e delle generalizzazioni
  - vincoli per stesso attributo di entità figlie della stessa entità padre
  - vincoli di generalizzazione (disgiuntezza e completezza)
  - si noti che tutte le entità diventano disgiunte
5. Scelta degli identificatori principali di entità e relazioni
  - tutte le entità devono avere un identificatore (eventualmente, introdurre codice)
  - eliminazione di cicli di identificatori principali esterni
6. Specifica degli ulteriori vincoli esterni
  - vincoli derivanti dalla ristrutturazione
  - riformulazione dei vincoli esterni dello schema originario
7. Riformulazione delle operazioni e delle specifiche sul carico applicativo in termini dello schema ristrutturato



## Esercizio 3: ristrutturare il seguente schema

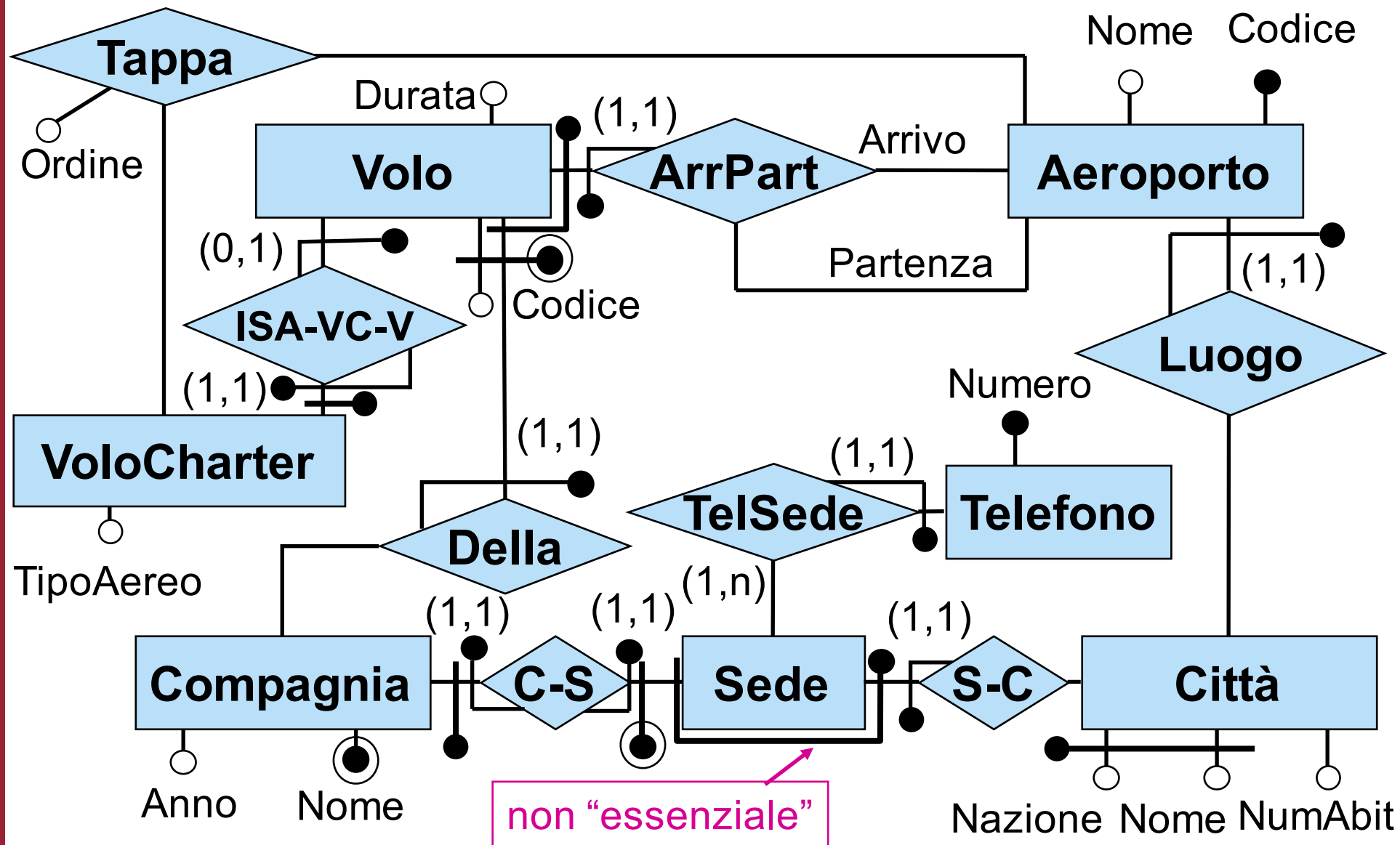


### Vincoli esterni:

- 1) per ogni  $v$  in **VoloCharter**, se  $(v, a_1), \dots, (v, a_n)$  sono tutte le coppie in **Tappa** alle quali partecipa  $v$ , e se  $o_1, \dots, o_n$  sono i valori assegnati a tali coppie dall'attributo *Ordine*, allora per  $i=1, \dots, n$  esiste un  $o_j$  tale che  $o_j=i$ .
- 2) Un telefono è di una sola sede.



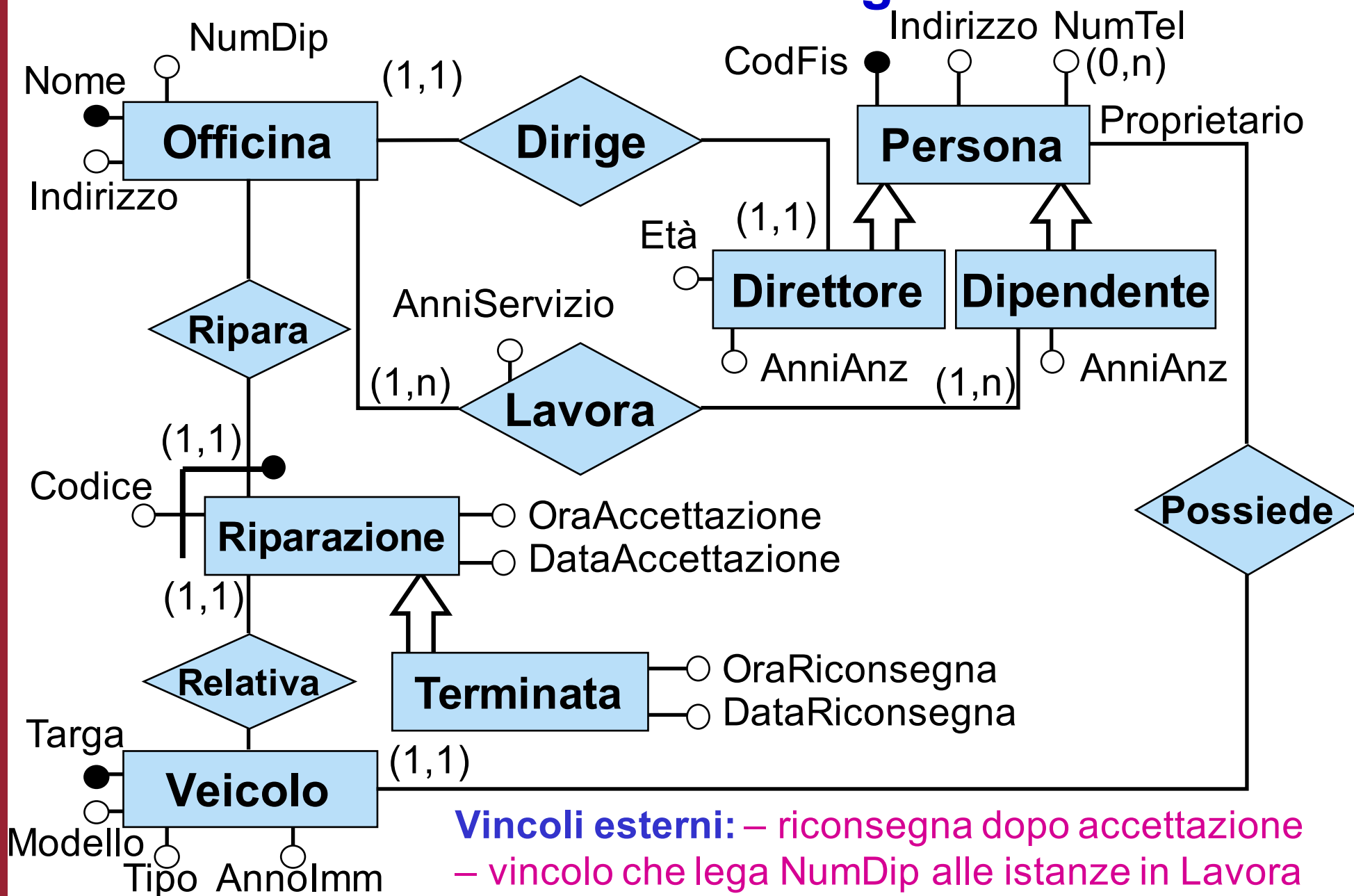
## Esercizio 3: soluzione



**Vincolo esterno: 1)** vincolo su Ordine in Tappa (2 è diventato interno allo schema)

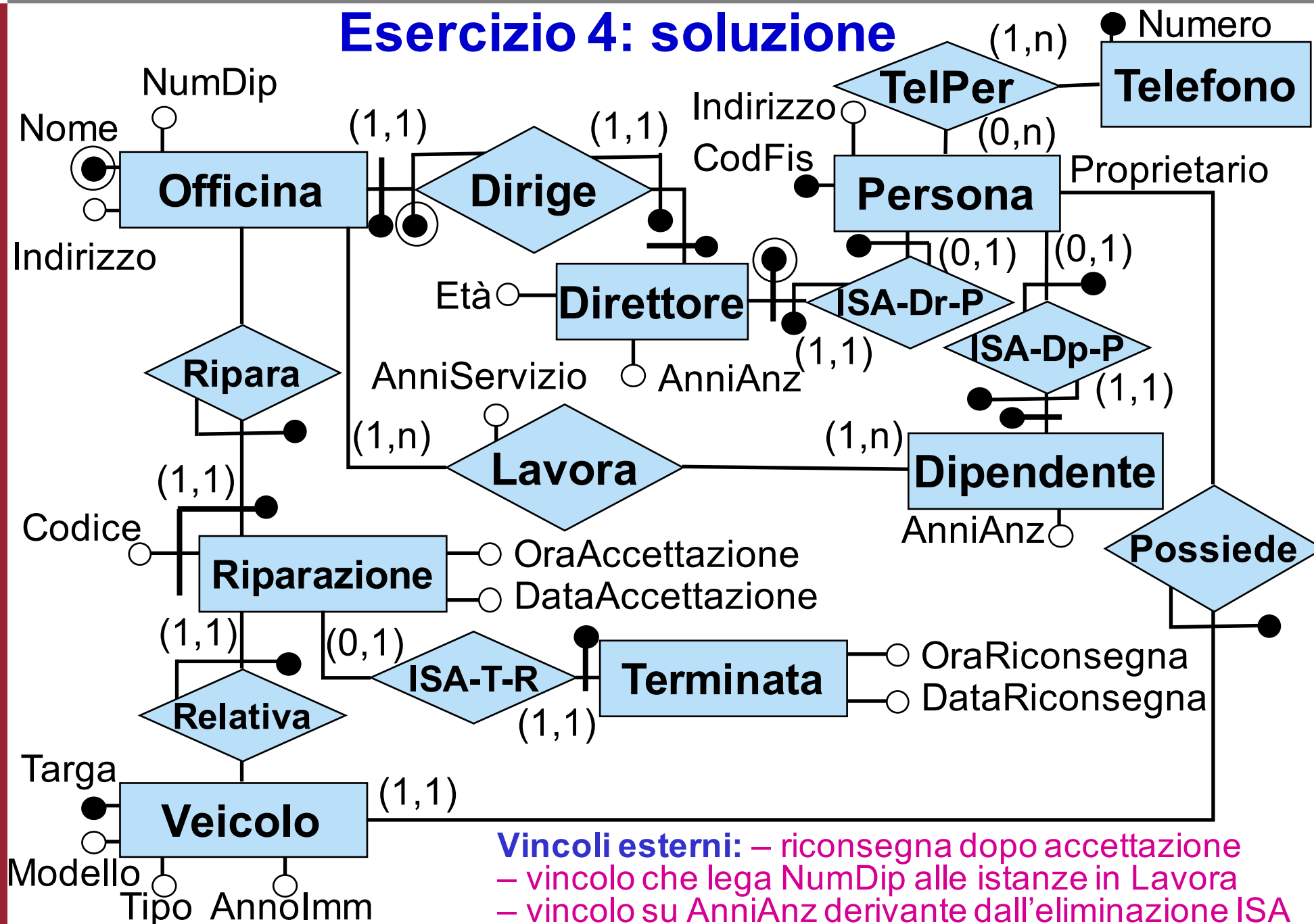


## Esercizio 4: ristrutturare il seguente schema





## Esercizio 4: soluzione





# 5. La progettazione logica

## 5.3 traduzione diretta nel modello relazionale

1. introduzione alla progettazione logica
2. ristrutturazione dello schema ER
- 3. traduzione diretta nel modello relazionale**
4. ristrutturazione dello schema logico



## Proprietà dello schema ristrutturato

La fase di ristrutturazione ha prodotto uno schema ER ristrutturato con le seguenti proprietà:

- preserva la semantica dello schema originale. Intuitivamente, esiste una funzione che associa ad ogni istanza dello schema originale un'opportuna istanza dello schema ristrutturato e viceversa
- può contenere delle ridondanze, ma sono volute per motivi di efficienza e sono comunque documentate
- non contiene attributi multivalore
- non contiene attributi composti
- non contiene ISA o generalizzazione (né tra entità, né tra relazioni); quindi tutte le entità sono disgiunte a coppie
- tutte le entità e tutte le relazioni hanno un unico identificatore principale

Lo schema ristrutturato è il punto di partenza per la traduzione nel modello relazionale.



## Traduzione diretta

- La traduzione diretta ha lo scopo di tradurre lo schema ER ristrutturato (con vincoli) in uno schema relazionale con vincoli che rappresenti le stesse informazioni
- Viene eseguita seguendo un metodo automatico, cioè che non richiede e di effettuare scelte, in quanto si basa sulle scelte fatte in fase di ristrutturazione
- Produce uno schema logico di massima, che può essere accettabile, ma che può richiedere successive ristrutturazioni, in particolare nella fase di ristrutturazione dello schema logico
- Consiste delle seguenti **attività**:
  - (1) traduzione delle **entità** in relazioni dello schema logico, con relativi vincoli
  - (2) traduzione delle **relazioni** dello schema ER in relazioni dello schema logico, con relativi vincoli
  - (3) traduzione dei **vincoli esterni** in vincoli dello schema logico
  - (4) riformulazione di **operazioni** e **specifiche** sul carico applicativo in termini dello schema logico

Come al solito, per distinguere tra le due accezioni di relazione, useremo il termine ER-relazione per denotare le relazioni dello schema ER





## Notazione

- Nella fase di traduzione diretta ed in quella di ristrutturazione dello schema logico, esprimeremo gli schemi relazionali mediante una notazione che prevede di descrivere le relazioni con nome e attributi, ed i vincoli ad esse associati in forma testuale. Illustriamo la notazione per mezzo di un esempio.

- Esempio:

chiave primaria      attributi non null

Partecipa(Cognome, DataN, Progetto, OreSett, Iva\*)

foreign key: Partecipa[Cognome, DataN]  $\subseteq$  Impiegato[Cognome, DataN]  
inclusione: Partecipa[OreSett]  $\subseteq$  Orario[Ore]  
chiave: Progetto

l'asterisco indica che l'attributo può assumere valore nullo

- Ovviamente, questa notazione si può tradurre senza difficoltà in termini di “create table” in SQL. Si lascia allo studente la verifica di come effettuare la traduzione in SQL



# (1) Traduzione di entità: regole generali

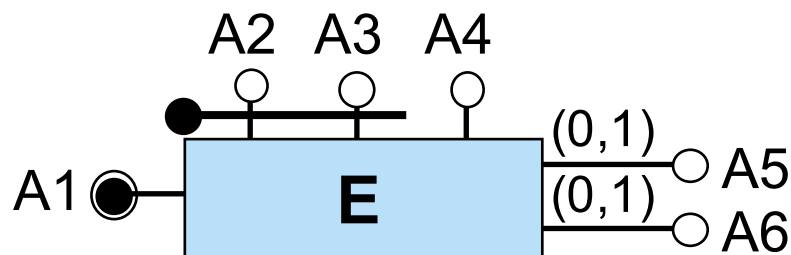
- Ogni entità **E** dello schema ER viene tradotta in una relazione **R<sub>E</sub>** dello schema relazionale
- Gli **attributi** della relazione **R<sub>E</sub>** sono:
  - gli attributi dell'entità **E** (tutti not null, tranne quelli con cardinalità (0,1))
  - gli attributi derivanti dall'accorpamento di ER-relazioni in **R<sub>E</sub>** – per ogni ER-relazione **Q** accorpata in **R<sub>E</sub>** vengono aggiunti ad **R<sub>E</sub>** come attributi:
    - le chiavi primarie delle relazioni che corrispondono alle altre entità che partecipano a **Q**
    - gli eventuali attributi della ER-relazione **Q**
- Una ER-relazione **Q** viene **accorpata** in **R<sub>E</sub>** quando uno o più ruoli di **Q** partecipano all'identificatore principale (esterno) di **E** (si noti che in questo caso **E** partecipa a **Q** con cardinalità (1,1), ed inoltre **E** è l'unica entità per cui un ruolo di **Q** partecipa all'identificatore principale esterno, altrimenti ci sarebbe un ciclo di identificazione principale esterna)
- La **chiave primaria** di **R<sub>E</sub>** è determinata in base all'identificatore principale di **E** (se interno, formato dagli attributi di **R<sub>E</sub>**, altrimenti formato da proprietà derivanti dall'identificazione esterna)
- Gli altri vincoli di identificazione di **E** verranno tipicamente tradotti in **vincoli di chiave** (non primaria) su **R<sub>E</sub>**
- A seconda dei casi, possono essere necessari ulteriori vincoli



# (1) Traduzione di entità senza accorpamento

Consideriamo per ora **un'entità per cui non si effettua accorpamento di relazioni** (in particolare, un'entità che non ha identificatori esterni).

- L'entità si traduce in una **relazione** dello schema relazionale
- Gli **attributi** della relazione corrispondente all'entità sono quelli dell'entità;
  - se un attributo è opzionale diventa un attributo della relazione che può assumere valore nullo (nella nostra notazione per lo schema logico, tali attributi sono indicati con \*)
  - altrimenti l'attributo non può assumere valore nullo
- L'identificatore principale dell'entità si traduce nella **chiave primaria** della relazione
- Gli altri identificatori interni si traducono in **chiavi** della relazione
- Ricordarsi dei vincoli esterni per identificatori opzionali correlati (derivanti da attributi composti opzionali)



*Vincoli esterni:*

- *per ogni istanza di E, A5 è definito se e solo se lo è anche A6*

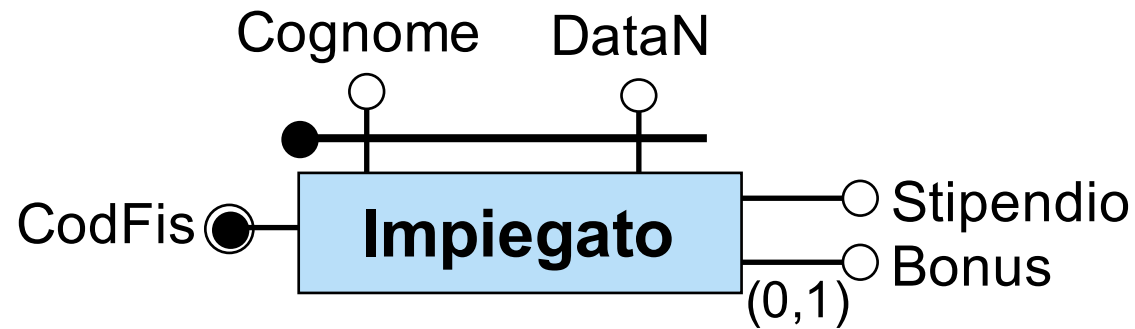
$E(\underline{A1}, A2, A3, A4, A5^*, A6^*)$

chiave: **A2, A3**

vincolo: **A5 è NULL se e solo se A6 è NULL**



# (1) Traduzione di entità senza accorpamento: esempio



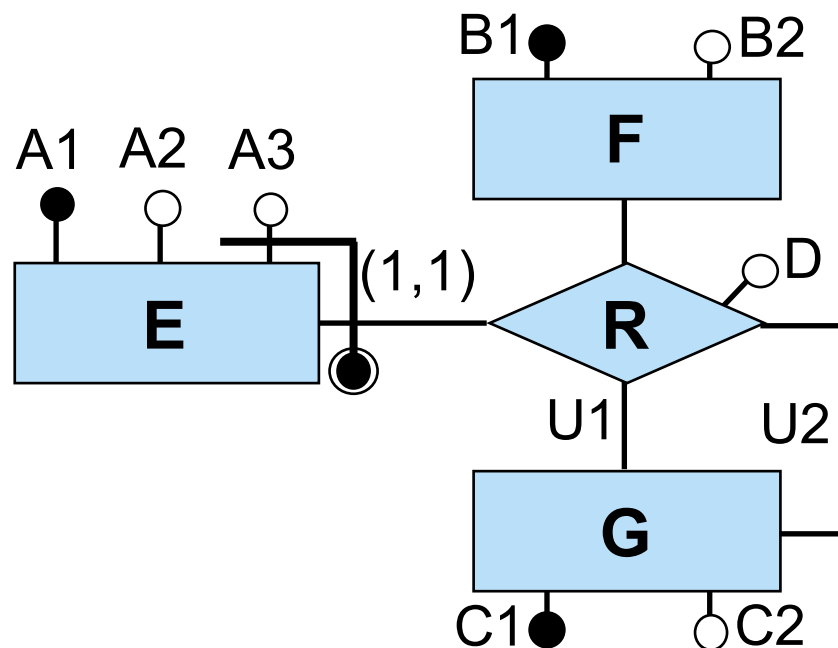
Impiegato(CodFis, Cognome, DataN, Stipendio, Bonus\*)  
chiave: Cognome, DataN



# (1) Traduzione di entità con accorpamento: caso 1

Consideriamo il caso in cui una ER-relazione **R** è **parte dell'identificatore principale esterno di un'entità E con ruolo U**, e in tutti gli altri ruoli di **R** la cardinalità è (0,n).

- La ER-relazione **R** viene **accorpata** nell'entità **E**. Questo significa che tutti gli attributi della ER-relazione e le chiavi primarie delle altre entità partecipanti diventano attributi della relazione che corrisponde all'entità **E**. Tali chiavi primarie contribuiscono (insieme agli eventuali altri attributi che formano l'identificatore principale di E) alla chiave primaria della relazione RE.
- Si noti che eventuali altri identificatori dell'entità in cui la relazione è stata accorpata si traducono in vincoli di chiave sull'entità.



$E(A1, A2, \underline{A3}, \underline{F}, \underline{U1}, \underline{U2}, D)$

foreign key:  $E[F] \subseteq F[B1]$

foreign key:  $E[U1] \subseteq G[C1]$

foreign key:  $E[U2] \subseteq G[C1]$

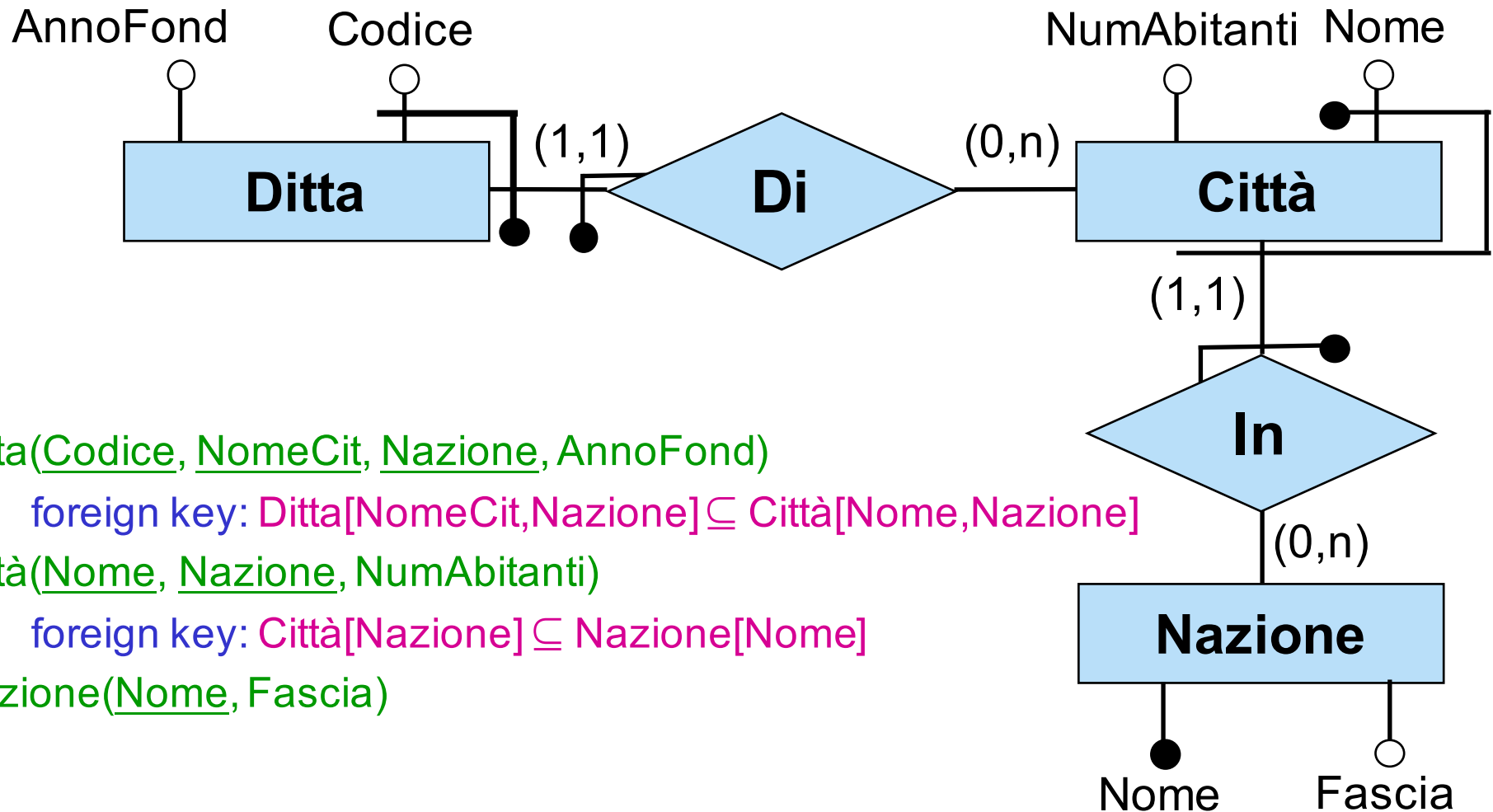
chiave:  $A1$

$F(\underline{B1}, B2)$

$G(\underline{C1}, C2)$



## (1) Traduzione di entità con accorpamento - caso 1: esempio



**Ditta**(Codice, NomeCit, Nazione, AnnoFond)

foreign key:  $Ditta[NomeCit, Nazione] \subseteq Città[Nome, Nazione]$

**Città**(Nome, Nazione, NumAbitanti)

foreign key:  $Città[Nazione] \subseteq Nazione[Nome]$

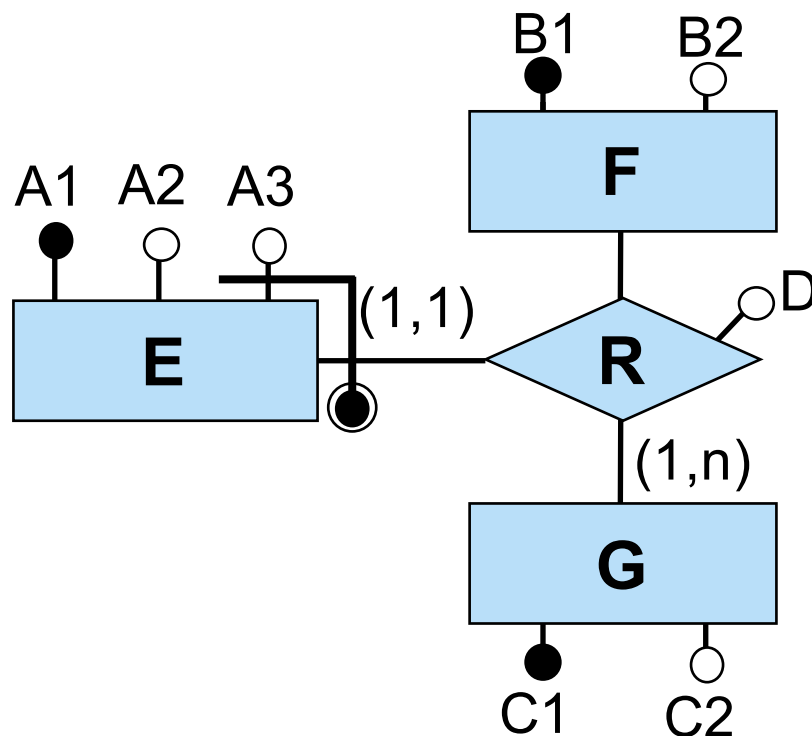
**Nazione**(Nome, Fascia)



# (1) Traduzione di entità con accorpamento: caso 2

Consideriamo il caso in cui una ER-relazione **R** è **parte dell'identificatore principale esterno di un'entità E con ruolo U**, ed in alcuni degli altri ruoli di **R** la cardinalità è (1,n).

- La ER-relazione **R** viene **accorpata** nell'entità **E** in modo analogo al caso 1
- Ma in questo caso, per ogni ruolo **U** con cardinalità (1,n), si aggiunge un vincolo di inclusione tra la chiave primaria dell'entità corrispondente al ruolo **U** e l'attributo (o gli attributi, se la chiave è composta) di **E** corrispondente ad **U**



$E(A1, A2, \underline{A3}, \underline{F}, \underline{G}, D)$

foreign key:  $E[F] \subseteq F[B1]$

foreign key:  $E[G] \subseteq G[C1]$

chiave:  $A1$

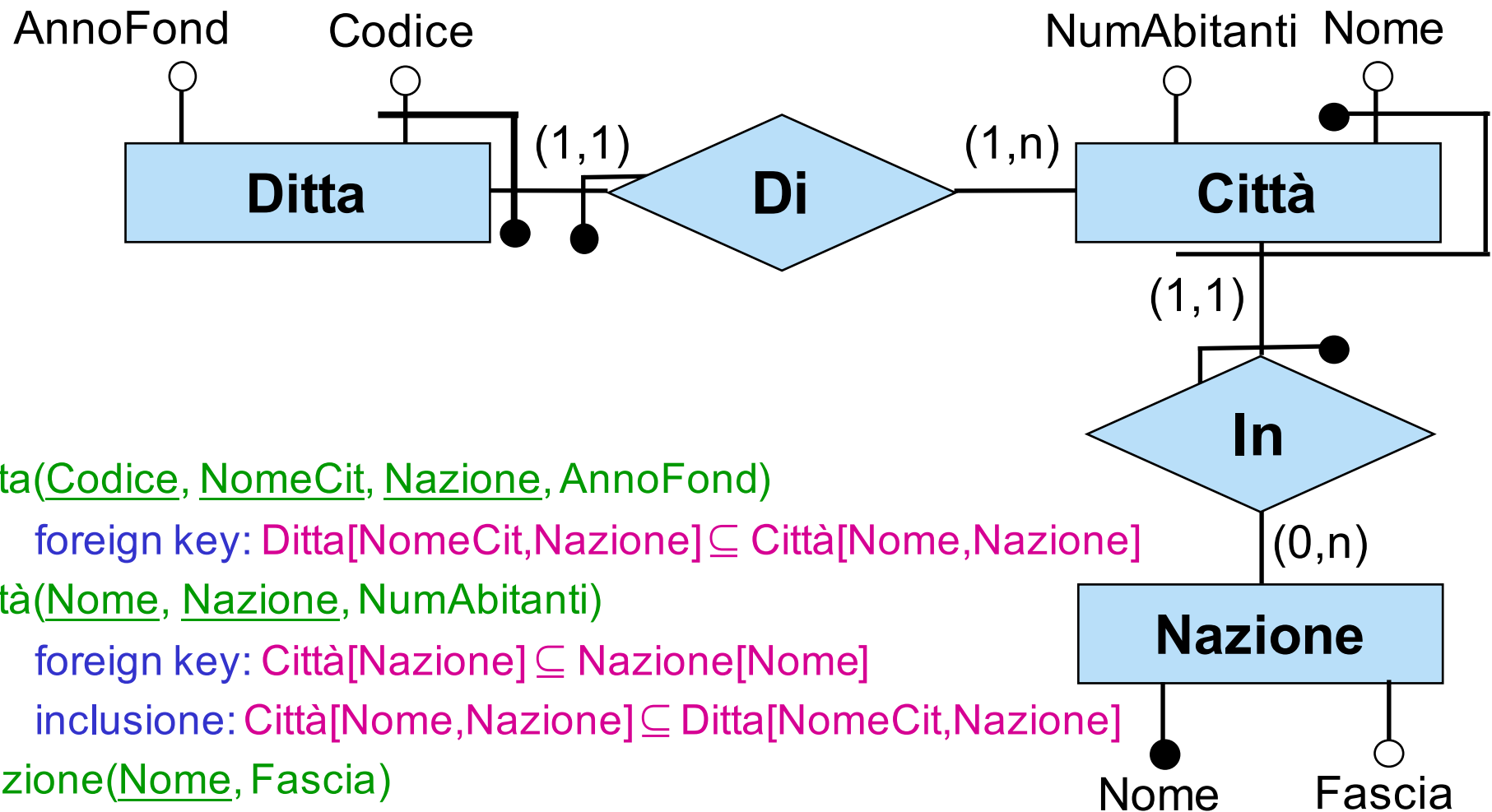
$F(\underline{B1}, B2)$

$G(\underline{C1}, C2)$

inclusione:  $G[C1] \subseteq E[G]$



## (1) Traduzione di entità con accorpamento - caso 2: esempio



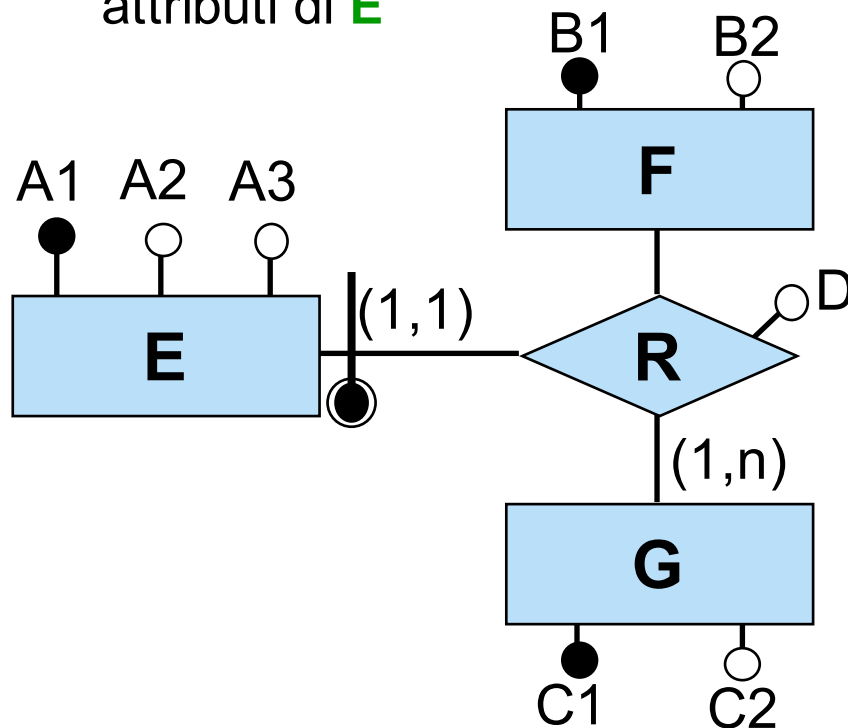




# (1) Traduzione di entità con accorpamento: caso 3

Consideriamo il caso in cui una ER-relazione **R** è **identificatore principale esterno** (non “parte” di identificatore) **di un’entità E con ruolo U**, ed in alcuni degli altri ruoli di **R** la cardinalità è (1,n).

- La ER-relazione **R** viene **accorpata** nell’entità **E** ed opportuni vincoli di inclusione riflettono la cardinalità minima pari ad 1, in modo analogo ai casi 1 e 2
- La differenza è che in questo caso la chiave primaria di **E** non comprende attributi di **E**



$E(A1, A2, A3, \underline{F}, \underline{G}, D)$

foreign key:  $E[F] \subseteq F[B1]$

foreign key:  $E[G] \subseteq G[C1]$

chiave: **A1**

$F(\underline{B1}, B2)$

$G(\underline{C1}, C2)$

inclusione:  $G[C1] \subseteq E[G]$



# (1) Traduzione di entità con accorpamento: caso 4

Consideriamo il caso in cui una ER-relazione **R** è **identificatore principale esterno di un'entità E con ruolo U**, e in alcuni degli altri ruoli di **R** la cardinalità massima è 1. Si noti che in questo caso **R** è l'identificatore principale (non “parte” di identificatore). La ER-relazione **R** viene **accorpata** nell'entità **E** in modo analogo ai casi precedenti, ma in questo caso la chiave primaria della relazione che rappresenta **E** deve essere scelta tra le chiavi primarie delle altre entità coinvolte in **R** i cui ruoli hanno cardinalità massima

Supponiamo di scegliere la chiave primaria di **F** come parte di chiave primaria di **E** (**U1** diventa quindi chiave non primaria di **E**):

**E**(**A1**, **A2**, **A3**, **E**, **U1**, **U2**, **P**)

foreign key:  $E[F] \subseteq F[B1]$

foreign key:  $E[U1] \subseteq G[C1]$

foreign key:  $E[U2] \subseteq G[C2]$

chiave: **A1**

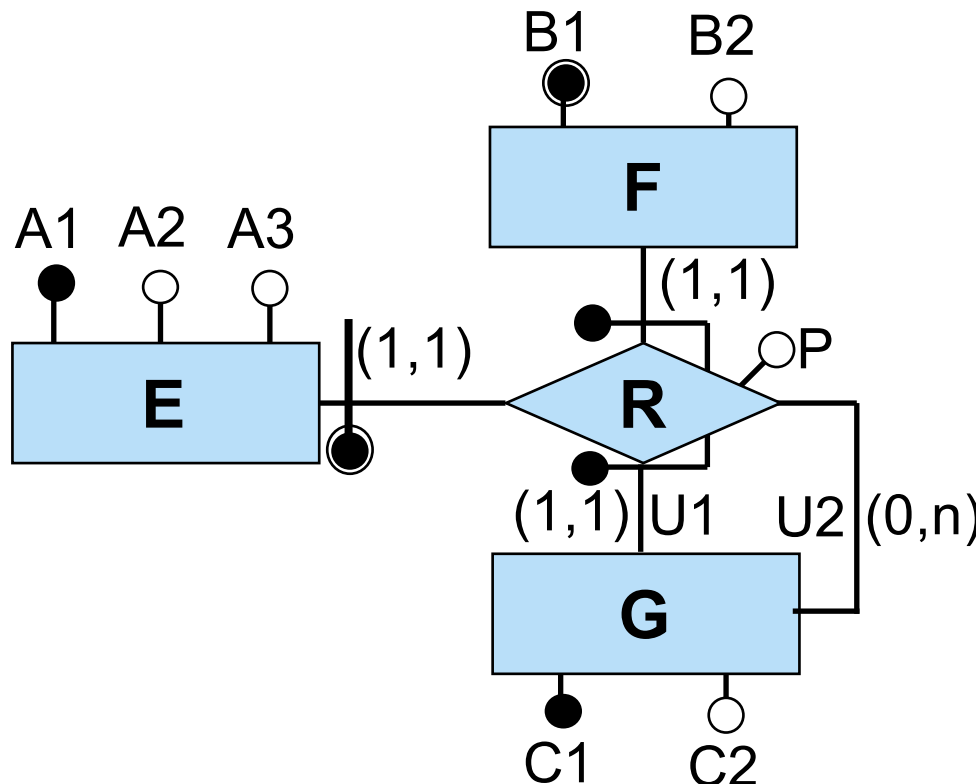
chiave: **U1**

**F**(**B1**, **B2**)

foreign key:  $F[B1] \subseteq E[F]$

**G**(**C1**, **C2**)

foreign key:  $G[C1] \subseteq E[U1]$

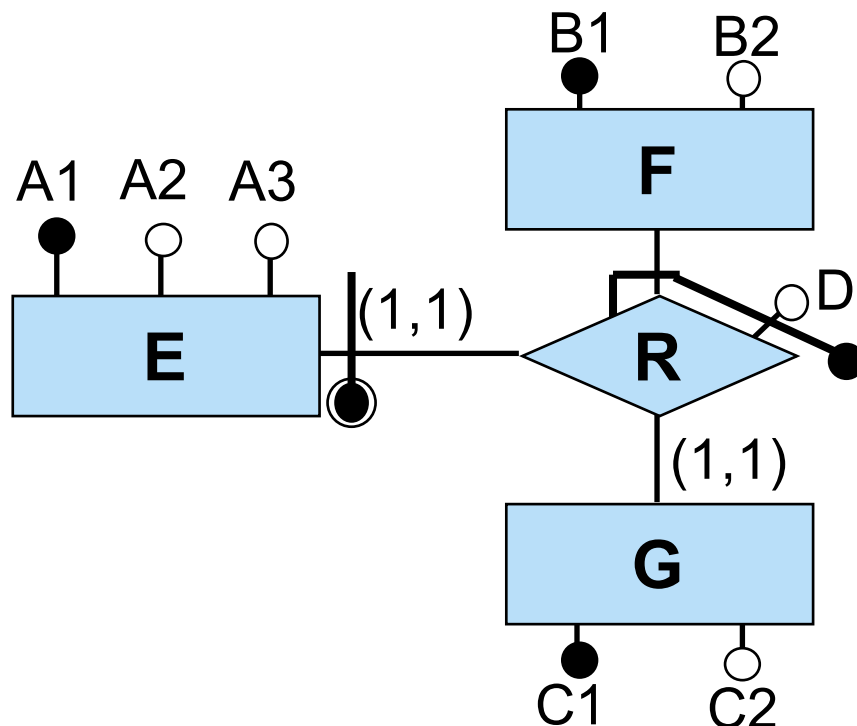




# (1) Traduzione di entità con accorpamento: caso 5

Consideriamo il caso in cui una ER-relazione **R** è **identificatore principale esterno** (non “parte” di identificatore) **di un’entità E con ruolo U**, in alcuni degli altri ruoli di **R** la cardinalità è (1,1), ed **R** ha un identificatore.

- La ER-relazione **R** viene **accorpata** nell’entità **E**, in modo analogo ai casi 1, 2 e 3. Scegliamo **G** come chiave primaria di **E**,
- La differenza è che in questo caso la tabella **E** ha una chiave in più, che proviene dall’identificatore di **R**



$E(A1, A2, A3, F, \underline{G}, D)$

foreign key:  $E[F] \subseteq F[B1]$

foreign key:  $E[G] \subseteq G[C1]$

chiave:  $A1$

chiave:  $F, D$

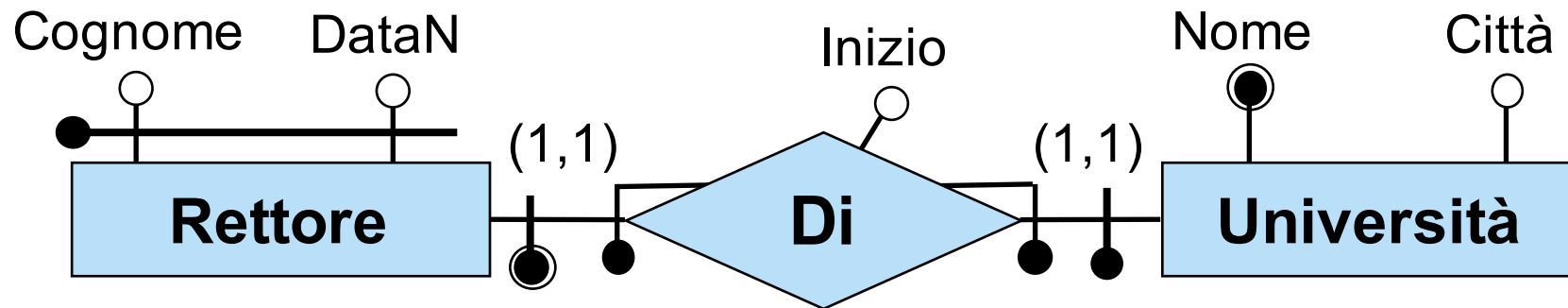
$F(\underline{B1}, B2)$

$G(\underline{C1}, C2)$

inclusione:  $G[C1] \subseteq E[G]$



# (1) Traduzione di entità con accorpamento – caso 4: esempio



**Rettore**(Cognome, DataN, Inizio, Università)

foreign key: Rettore[Università]  $\subseteq$  Università[Nome]

chiave: Cognome, DataN

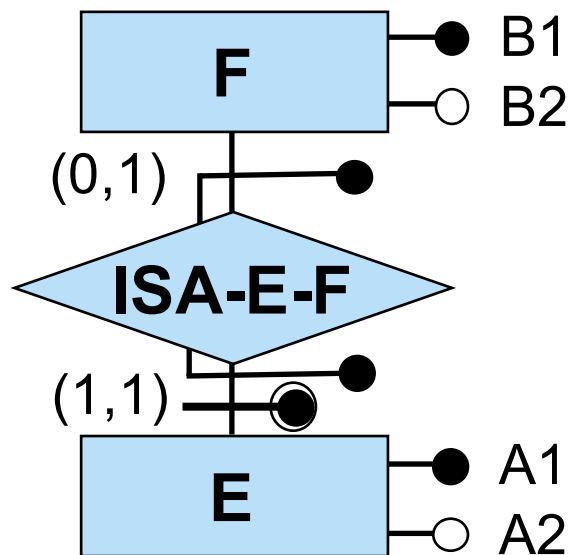
**Università**(Nome, Città)

foreign key: Università[Nome]  $\subseteq$  Rettore[Università]



# (1) Accorpamento di relazione derivante da ISA o generalizzazione

Un caso importante di ER-relazione che è identificatore principale esterno di un'entità è quello derivante dalla ristrutturazione di una relazione ISA o di una generalizzazione nello schema ER originale.



$E(\underline{A1}, A2, \underline{B1})$

foreign key:  $E[B1] \subseteq F[B1]$

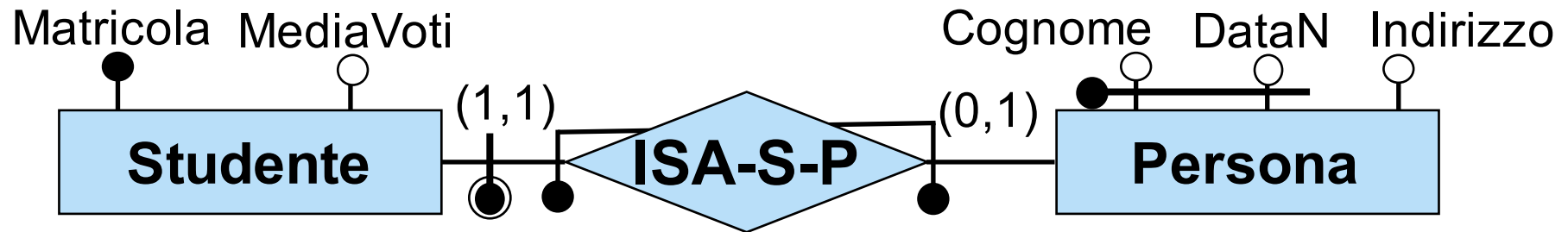
chiave:  $A1$

$F(\underline{B1}, B2)$

Si noti come la traduzione della parte di schema ER che si ottiene dalla ristrutturazione di **E ISA F** corrisponda ad aggiungere agli attributi della tabella **E** la chiave primaria della tabella **F**, ed a rendere tali attributi anche chiave primaria della tabella **E**. Il vincolo derivante dall'ISA dello schema ER originario diventa quindi un vincolo di foreign key dello schema logico.



## (1) Accorpamento di relazione derivante da ISA: esempio



Studente(Cognome, DataN, Matricola, MediaVoti)

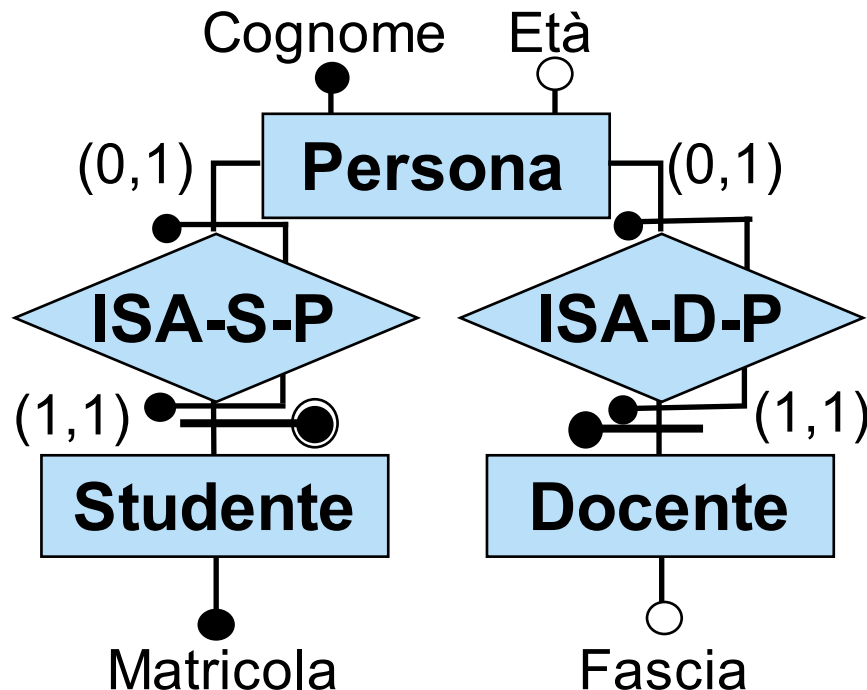
foreign key:  $\text{Studente}[\text{Cognome}, \text{DataN}] \subseteq \text{Persona}[\text{Cognome}, \text{DataN}]$

chiave: Matricola

Persona(Cognome, DataN, Indirizzo)



## (1) Accorpamento di relazione derivante da generalizzazione non completa: esempio



Persona(Cognome, Età)

Studente(Cognome, Matricola)

foreign key:  $\text{Studente}[\text{Cognome}] \subseteq \text{Persona}[\text{Cognome}]$

chiave: Matricola

Docente(Cognome, Fascia)

foreign key:  $\text{Docente}[\text{Cognome}] \subseteq \text{Persona}[\text{Cognome}]$

**Vincolo di generalizzazione:**

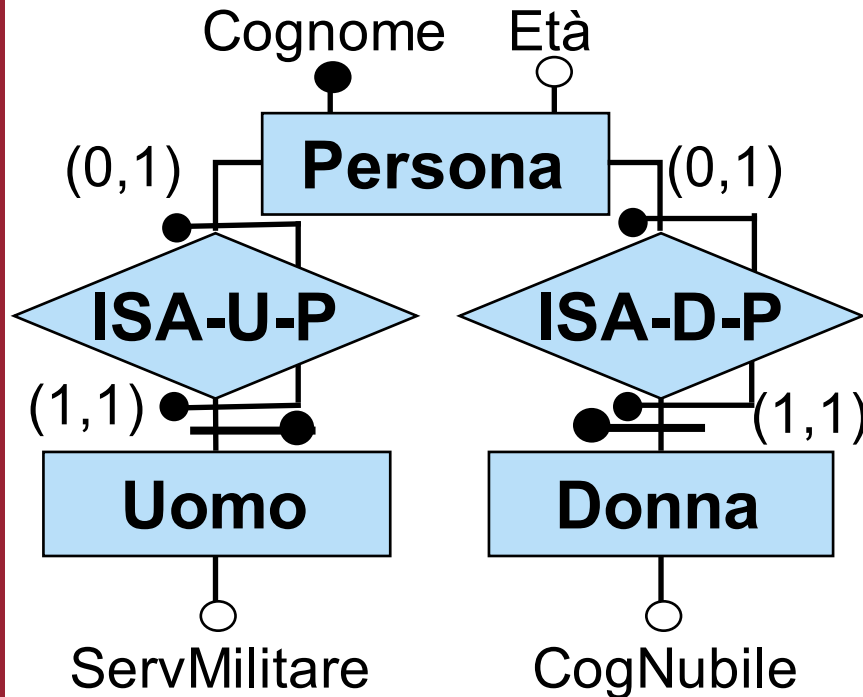
nessuna istanza di Persona  
partecipa sia a ISA-S-P sia a  
ISA-D-P

**Il vincolo di generalizzazione  
diventa sullo schema logico:**

$\text{Studente}[\text{Cognome}] \cap$   
 $\text{Docente}[\text{Cognome}] = \emptyset$



# (1) Accorpamento di relazione derivante da generalizzazione completa: esempio



## Vincolo di generalizzazione:

ogni istanza di Persona  
partecipa ad ISA-U-P oppure ad  
ISA-D-P, ma non ad entrambi

## Il vincolo di generalizzazione diventa sullo schema logico:

$Uomo[Cognome] \cap Donna[Cognome] = \emptyset$

$Persona[Cognome] \subseteq$

$Uomo[Cognome] \cup Donna[Cognome]$

$Persona(\underline{Cognome}, Et\grave{a})$

$Uomo(\underline{Cognome}, ServMilitare)$

foreign key:  $Uomo[Cognome] \subseteq Persona[Cognome]$

$Donna(\underline{Cognome}, CogNubile)$

foreign key:  $Donna[Cognome] \subseteq Persona[Cognome]$





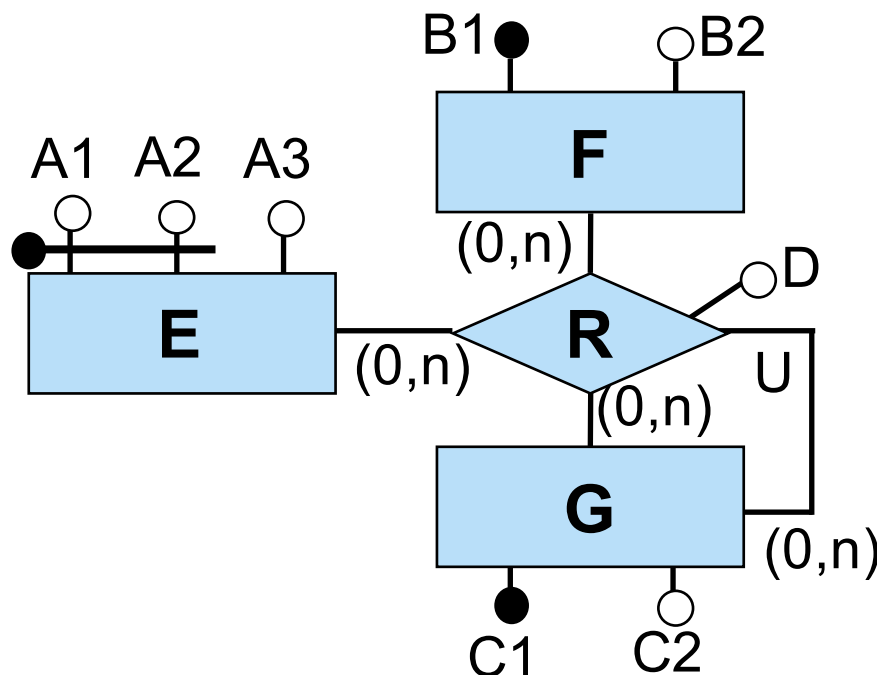
## (2) Traduzione di ER-relazioni: regole generali

- Ogni ER-relazione **Q** dello schema ER che non è stata accorpata al passo precedente viene tradotta in una relazione **R<sub>Q</sub>** dello schema relazionale.
- Gli **attributi** della relazione **R<sub>Q</sub>** sono:
  - gli attributi della ER-relazione **Q**
  - le chiavi primarie delle entità che partecipano alla ER-relazione **Q**
- La scelta della **chiave primaria** di **R<sub>Q</sub>** è basata sull'identificatore principale di **R<sub>Q</sub>**
- Gli altri identificatori essenziali di **R<sub>Q</sub>** danno luogo a **vincoli di chiave** (non primaria) su **R<sub>Q</sub>**
- Le tipizzazioni delle componenti di **Q**, per ogni ruolo, con le entità partecipanti divengono in **R<sub>Q</sub>** vincoli di **foreign key** verso le relazioni che corrispondono alle entità partecipanti
- A seconda dei casi, possono essere necessari ulteriori vincoli



## (2) Traduzione di relazione – caso 1

- Consideriamo il caso di ER-relazione **R** che **non è stata accorpata ad alcuna entità, ed in cui l'unico vincolo di identificazione sia quello implicito**.
- La ER-relazione si traduce in una relazione
- Gli **attributi** della relazione sono quelli della ER-relazione, più le chiavi primarie delle relazioni corrispondenti alle entità partecipanti (per ogni ruolo)
- La **chiave primaria** è determinata dall'identificatore principale della ER-relazione. Ovviamente, poiché l'identificatore principale è quello implicito, la chiave primaria della relazione sarà data dalle chiavi primarie delle entità partecipanti
- Si definiscono vincoli di foreign key dalla relazione verso le entità partecipanti per dar conto dei vincoli di tipizzazione nello schema ER



$E(\underline{A1}, \underline{A2}, A3)$

$F(\underline{B1}, B2)$

$G(\underline{C1}, C2)$

$R(\underline{EA1}, \underline{EA2}, \underline{F}, \underline{G}, \underline{U}, D)$

foreign key:  $R[EA1, EA2] \subseteq E[A1, A2]$

foreign key:  $R[F] \subseteq F[B1]$

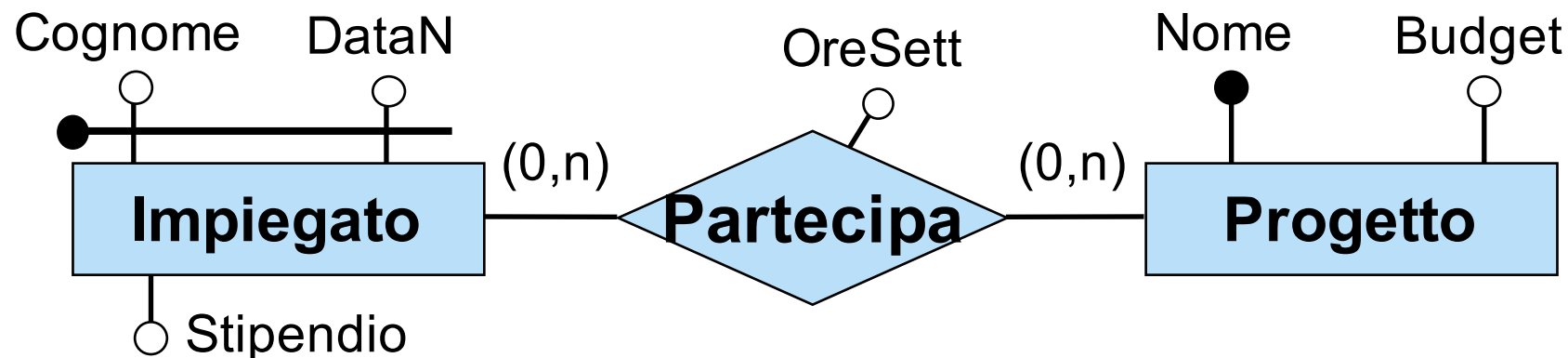
foreign key:  $R[G] \subseteq G[C1]$

foreign key:  $R[U] \subseteq G[C1]$



## (2) Traduzione di relazione – caso 1: esempio

Nello scegliere per una relazione il nome di un attributo che rappresenta la chiave primaria di un'entità che partecipa alla relazione, può essere opportuno utilizzare il nome del ruolo con cui l'entità partecipa alla relazione (invece del nome che l'attributo ha per l'entità).



Impiegato(Cognome, DataN, Stipendio)

Progetto(Nome, Budget)

Partecipa(Cognome, DataN, Progetto, OreSett)

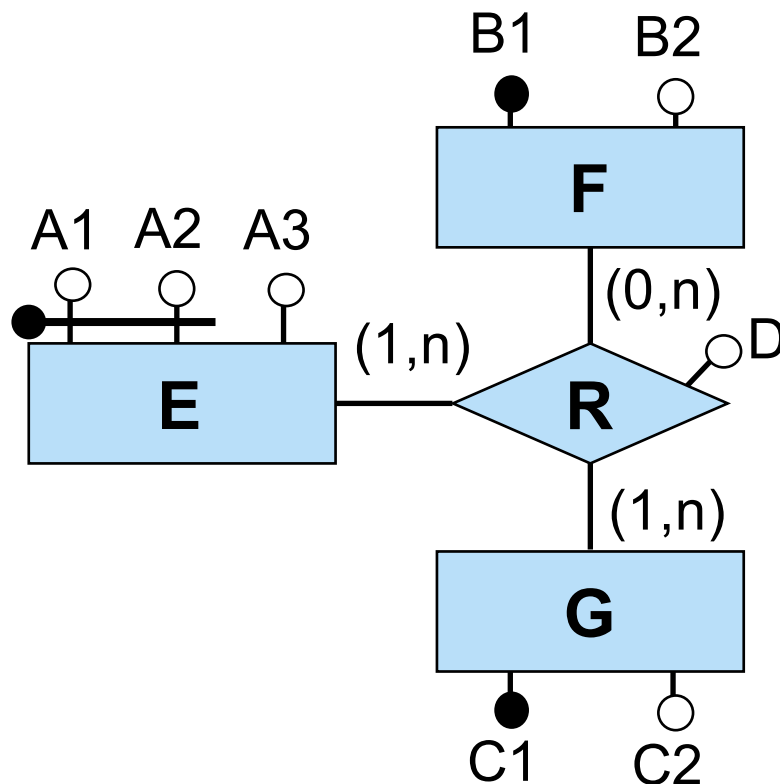
foreign key: Partecipa[Cognome,DataN]  $\subseteq$  Impiegato[Cognome,DataN]

foreign key: Partecipa[Progetto]  $\subseteq$  Progetto[Nome]



## (2) Traduzione di relazione – caso 1 con cardinalità minima pari ad 1

- Consideriamo ancora il caso di ER-relazione **R** che **non è stata accorpata ad alcuna entità, ed in cui l'unico vincolo di identificazione sia quello implicito**.
- Un vincolo di cardinalità minima 1 per la partecipazione di un'entità (in un ruolo) alla relazione si traduce in un **vincolo di inclusione** dall'entità verso l'attributo (o gli attributi) corrispondenti a quel ruolo nella relazione.
- Il vincolo di inclusione non è in generale di foreign key (si vedano in seguito i casi in cui l'inclusione diventa in realtà una foreign key).



$E(\underline{A1}, \underline{A2}, A3)$

inclusione:  $E[A1, A2] \subseteq R[A1, A2]$

$F(\underline{B1}, B2)$

$G(\underline{C1}, C2)$

inclusione:  $G[C1] \subseteq R[C1]$

$R(\underline{A1}, \underline{A2}, \underline{B1}, \underline{C1}, D)$

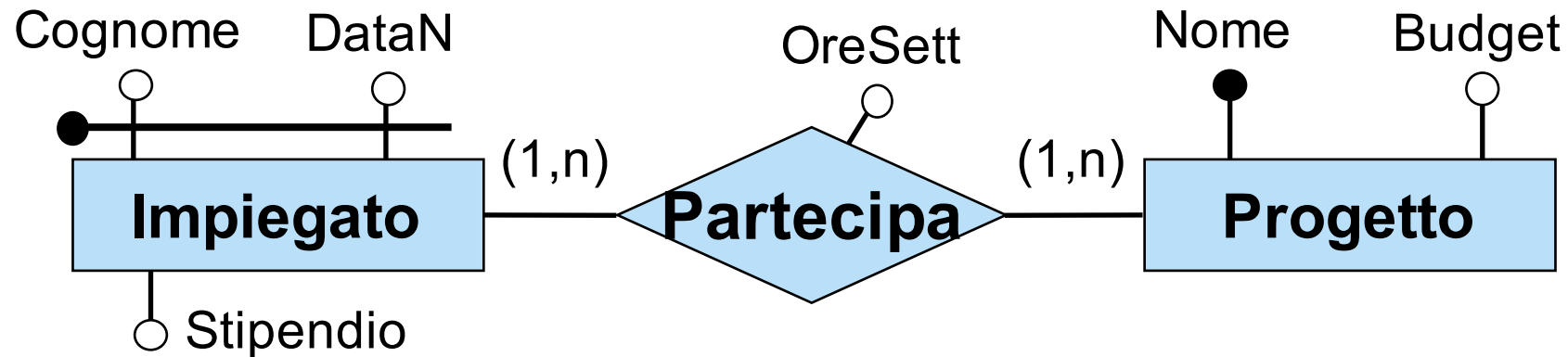
foreign key:  $R[A1, A2] \subseteq E[A1, A2]$

foreign key:  $R[B1] \subseteq F[B1]$

foreign key:  $R[C1] \subseteq G[C1]$



## (2) Traduzione di relazione – caso 1 con cardinalità minima pari ad 1: esempio



Impiegato(Cognome, DataN, Stipendio)

inclusione:  $\text{Impiegato}[\text{Cognome}, \text{DataN}] \subseteq \text{Partecipa}[\text{Cognome}, \text{DataN}]$

Progetto(Nome, Budget)

inclusione:  $\text{Progetto}[\text{Nome}] \subseteq \text{Partecipa}[\text{Progetto}]$

Partecipa(Cognome, DataN, Progetto, OreSett)

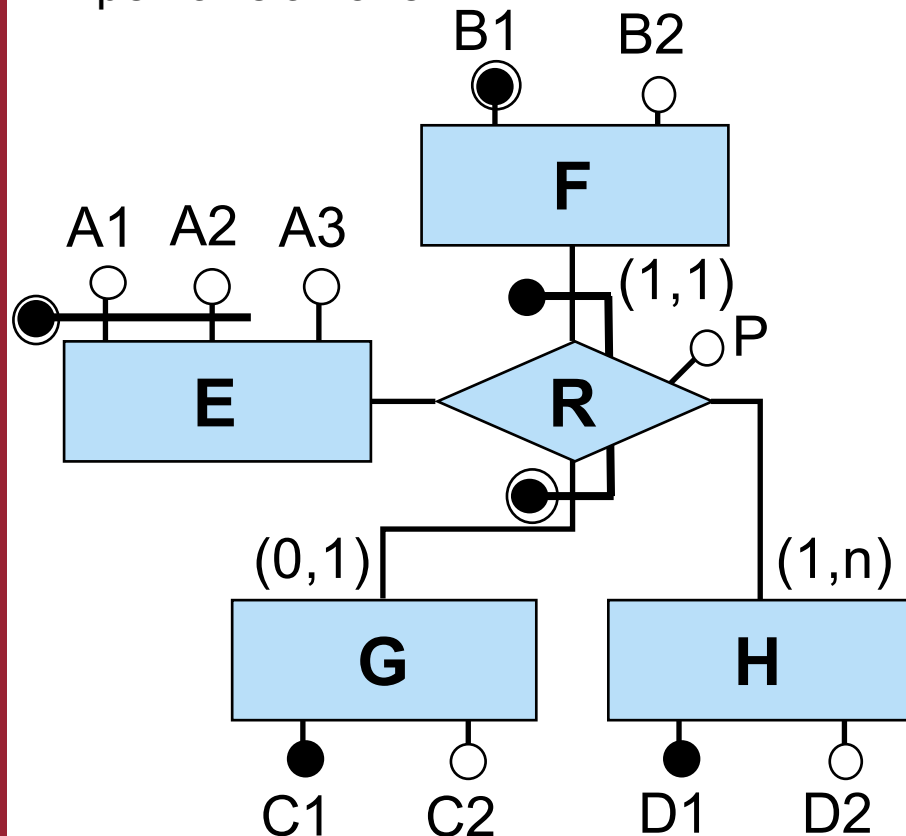
foreign key:  $\text{Partecipa}[\text{Cognome}, \text{DataN}] \subseteq \text{Impiegato}[\text{Cognome}, \text{DataN}]$

foreign key:  $\text{Partecipa}[\text{Progetto}] \subseteq \text{Progetto}[\text{Nome}]$



## (2) Traduzione di relazione – caso 2

- Consideriamo il caso di ER-relazione **R** che **non è stata accorpata ad alcuna entità, ed in cui il vincolo di identificazione principale non sia quello implicito**.
- L'identificatore principale della ER-relazione determina la chiave primaria della relazione. Se l'identificatore principale della relazione è costituito da una sola entità e se tale l'entità ha anche cardinalità minima 1, il vincolo di inclusione corrispondente diventa in realtà un vincolo di foreign key.
- Gli identificatori di relazione che non sono principali si traducono in vincoli di chiave per la relazione.



$E(\underline{A1}, \underline{A2}, A3)$

$F(\underline{B1}, B2)$

foreign key:  $F[B1] \subseteq R[F]$

$G(\underline{C1}, C2)$

$H(\underline{D1}, D2)$

inclusione:  $H[D1] \subseteq R[H]$

$R(EA1, EA2, F, \underline{G}, H, P)$

foreign key:  $R[EA1, EA2] \subseteq E[A1, A2]$

foreign key:  $R[F] \subseteq F[B1]$

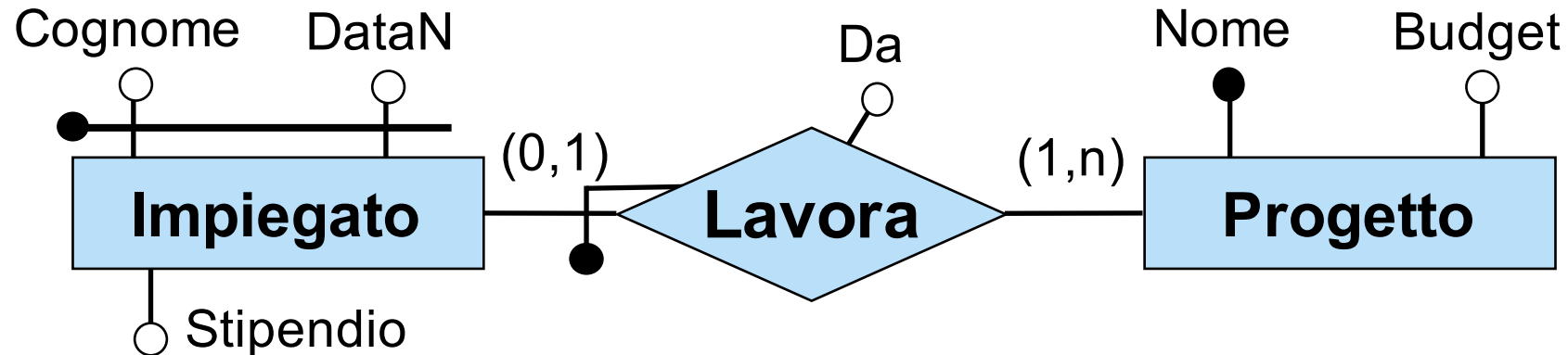
foreign key:  $R[G] \subseteq G[C1]$

foreign key:  $R[H] \subseteq H[D1]$

chiave: **F**



## (2) Traduzione di relazione – caso 2: esempio



Impiegato(Cognome, DataN, Stipendio)

Progetto(Nome, Budget)

inclusione: Progetto[Nome]  $\subseteq$  Lavora[Progetto]

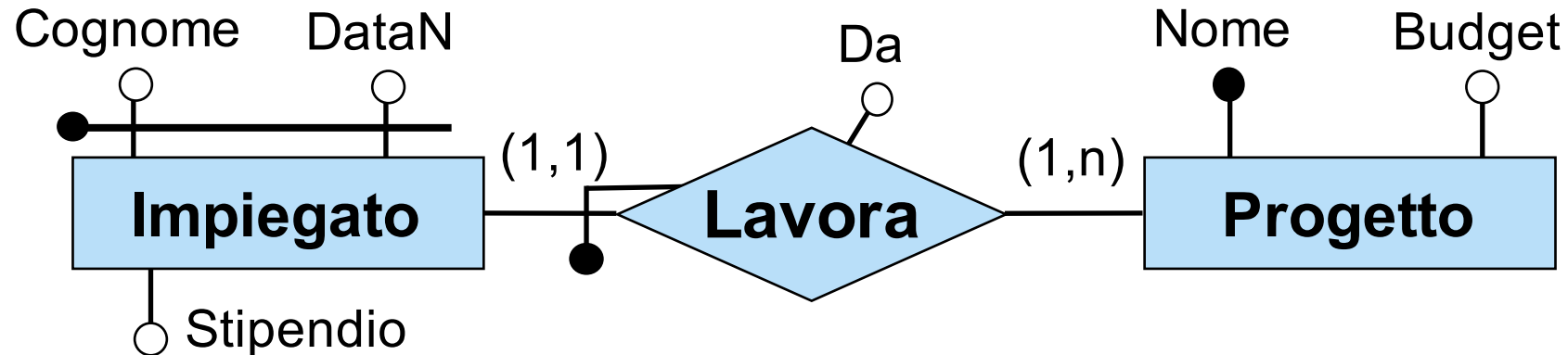
Lavora(Cognome, DataN, Progetto, Da)

foreign key: Lavora[Cognome,DataN]  $\subseteq$  Impiegato[Cognome,DataN]

foreign key: Lavora[Progetto]  $\subseteq$  Progetto[Nome]



## (2) Traduzione di relazione – caso 2 con cardinalità minima pari da 1: esempio



**Impiegato**(Cognome, DataN, Stipendio)

foreign key:  $\text{Impiegato}[\text{Cognome}, \text{DataN}] \subseteq \text{Lavora}[\text{Cognome}, \text{DataN}]$

**Progetto**(Nome, Budget)

inclusione:  $\text{Progetto}[\text{Nome}] \subseteq \text{Lavora}[\text{Progetto}]$

**Lavora**(Cognome, DataN, Progetto, Da)

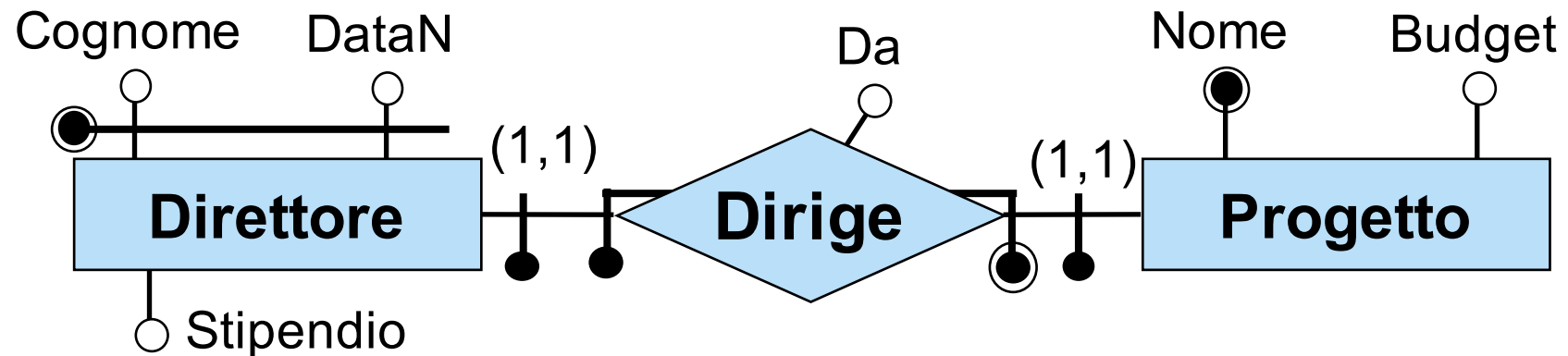
foreign key:  $\text{Lavora}[\text{Cognome}, \text{DataN}] \subseteq \text{Impiegato}[\text{Cognome}, \text{DataN}]$

foreign key:  $\text{Lavora}[\text{Progetto}] \subseteq \text{Progetto}[\text{Nome}]$





## (2) Traduzione di relazione – caso 2: altro esempio



Direttore(Cognome, DataN, Stipendio)

foreign key: Direttore[Cognome,DataN]  $\subseteq$  Dirige[Cognome,DataN]

Progetto(Nome, Budget)

foreign key: Progetto[Nome]  $\subseteq$  Dirige[Progetto]

Dirige(Cognome, DataN, Progetto, Da)

chiave: Cognome, DataN

foreign key: Dirige[Cognome,DataN]  $\subseteq$  Direttore[Cognome,DataN]

foreign key: Dirige[Progetto]  $\subseteq$  Progetto[Nome]



```

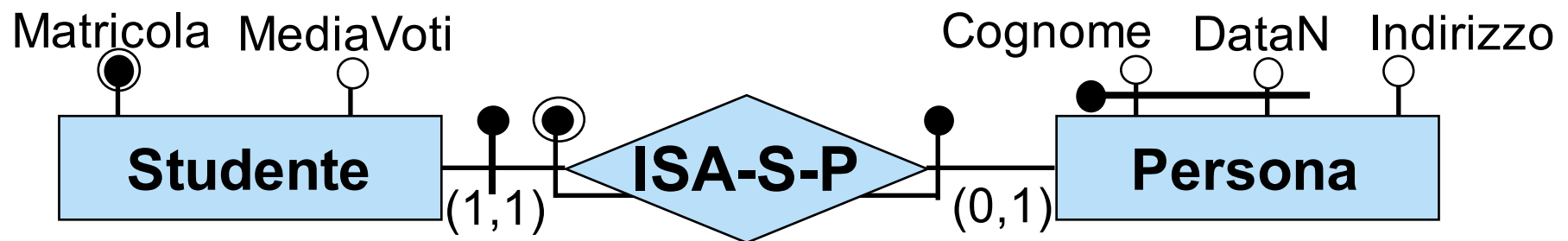
graph TD
    A1((A1)) --> E[E]
    A2((A2)) --> E
    A3((A3)) --> E
    E --> R{R}
    B1((B1)) --> F[F]
    B2((B2)) --> F
    F -- "(1,1)" --> F
    R --> G[G]
    R --> H[H]
    G --> C1((C1))
    G --> C2((C2))
    H -- "(1,n)" --> H
    H --> D1((D1))
    H --> D2((D2))
    style A1 fill:#000,stroke:#000,stroke-width:2px
    style B1 fill:#000,stroke:#000,stroke-width:2px
    style C1 fill:#000,stroke:#000,stroke-width:2px
    style D1 fill:#000,stroke:#000,stroke-width:2px
    style E fill:#add8e6,stroke:#000,stroke-width:1px
    style F fill:#add8e6,stroke:#000,stroke-width:1px
    style G fill:#add8e6,stroke:#000,stroke-width:1px
    style H fill:#add8e6,stroke:#000,stroke-width:1px
    style R fill:#add8e6,stroke:#000,stroke-width:1px
    style A2 fill:#fff,stroke:#000,stroke-width:1px
    style A3 fill:#fff,stroke:#000,stroke-width:1px
    style B2 fill:#fff,stroke:#000,stroke-width:1px
    style C2 fill:#fff,stroke:#000,stroke-width:1px
    style D2 fill:#fff,stroke:#000,stroke-width:1px
    style P((P)) fill:#fff,stroke:#000,stroke-width:1px
  
```

chiave: F



## (2) Traduzione di relazione derivante da ISA: esempio

Una relazione derivante da ISA che non è stata accorpata (perché l'identificatore principale dell'entità che era figlia nello schema ER originario non è quello esterno) si traduce tenendo conto dell'identificatore principale della relazione (che in questo caso sappiamo deve essere indicato). Consideriamo il seguente esempio:



**ISA-S-P**(Matricola, Cognome, DataN)

foreign key: **ISA-S-P**[Cognome, DataN]  $\subseteq$  **Persona**[Cognome, DataN]

foreign key: **ISA-S-P**[Matricola]  $\subseteq$  **Studente**[Matricola]

chiave: Cognome, DataN

**Studente**(Matricola, MediaVoti)

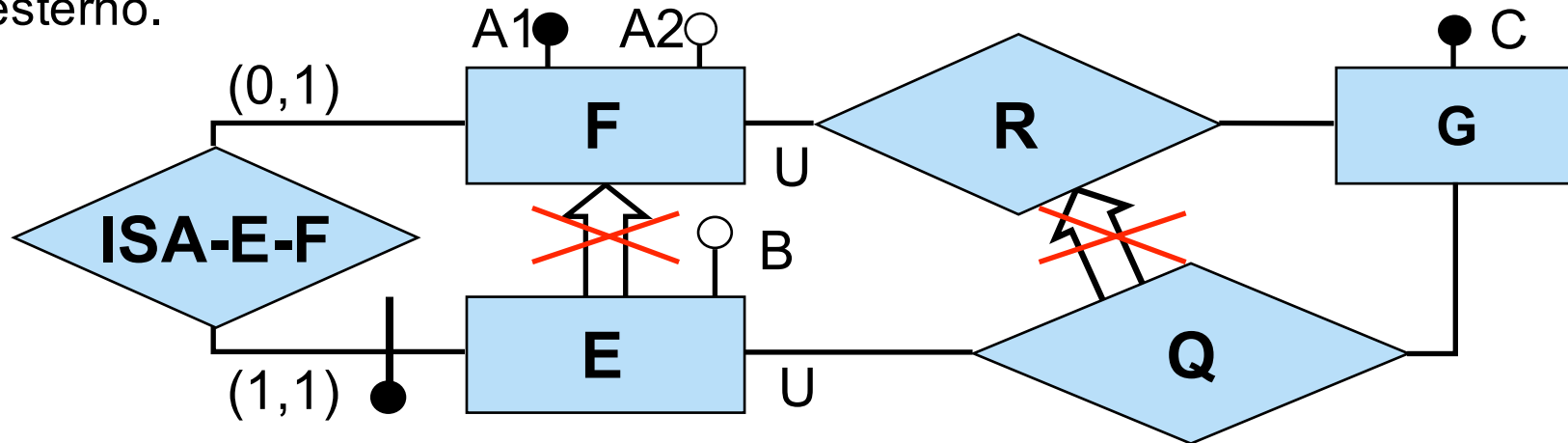
foreign key: **Studente**[Matricola]  $\subseteq$  **ISA-S-P**[Matricola]

**Persona**(Cognome, DataN, Indirizzo)



## (2) Traduzione di vincoli derivanti da ISA tra relazioni

Si ricordi che la ristrutturazione di una ISA tra relazioni ha prodotto un vincolo esterno.



**Vincolo esterno:** per ogni istanza  $(e,g)$  di  $Q$ , sia  $f$  l'istanza di  $F$  tale che  $(e,f)$  è un'istanza di  $ISA-E-F$  (si noti che  $f$  esiste sempre ed è unica). Allora  $(f,g)$  deve essere un'istanza di  $R$ .

**Traduzione:** il vincolo esterno diventa un vincolo di **foreign key**

$E(\underline{A1}, B)$  foreign key:  $E[A1] \subseteq F[A1]$

$F(\underline{A1}, A2)$

$G(\underline{C})$

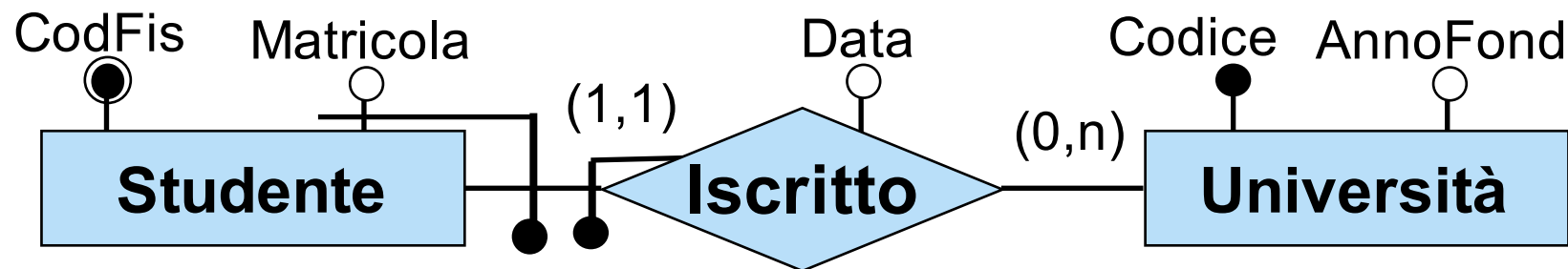
$R(\underline{A1}, \underline{C})$  foreign key:  $R[A1] \subseteq F[A1]$  foreign key:  $R[C] \subseteq G[C]$

$Q(\underline{A1}, \underline{C})$  foreign key:  $Q[A1] \subseteq E[A1]$  foreign key:  $Q[A1, C] \subseteq R[A1, C]$



## (1) Traduzione di entità con identificatore esterno non principale

Ora che sappiamo come si traducono le relazioni, possiamo tornare brevemente alla traduzione delle entità per osservare che **un vincolo di identificazione non principale esterna per una entità E diventa un vincolo esterno (cosiddetto “di join”) che riguarda sia l’entità E sia la relazione che partecipa all’identificazione.**



**Iscritto**(Studente, Università, Data)

foreign key:  $\text{Iscritto}[\text{Studente}] \subseteq \text{Studente}[\text{CodFis}]$

foreign key:  $\text{Iscritto}[\text{Università}] \subseteq \text{Università}[\text{Codice}]$

**Studente**(CodFis, Matricola)

foreign key:  $\text{Studente}[\text{CodFis}] \subseteq \text{Iscritto}[\text{Studente}]$

**Università**(Codice, AnnoFond)

**Vincolo:** nel join tra **Studente** e **Iscritto** sugli attributi **CodFis** e **Studente**, gli attributi **Matricola** e **Università** formano una chiave



### (3) Traduzione di vincoli

Questi sono i vincoli da considerare nella traduzione:

- Vincoli **not null** per gli attributi obbligatori
- Vincoli di interdipendenza di valori nulli (provenienti da attributi composti opzionali), formulati come vincoli di tupla
- Vincoli di **chiave** (primarie e non)
- Vincoli di **foreign key** che provengono
  - dalla tipizzazione di relazioni (incluse quelle che sono state accorpate in entità)
  - dai vincoli esterni derivanti dall'ISA di relazioni
- Vincoli **di generalizzazione**, formulati come vincoli insiemistici
- Vincoli di cardinalità:
  - La partecipazione obbligatoria (cardinalità minima 1) diventa **vincolo di inclusione** o **foreign key** dalla relazione che corrisponde all'entità a quella che corrisponde alla ER-relazione
  - La funzionalità (cardinalità massima 1) diventa **vincolo di chiave** sulla relazione che corrisponde alla ER-relazione
  - Gli altri vincoli di cardinalità diventano vincoli esterni
- Gli altri vincoli esterni vanno opportunamente tradotti



## **(4) Riformulazione di operazioni e carico applicativo**

- Le operazioni e le informazioni sul carico applicativo sono state espresse all'inizio della progettazione logica sulla base dello schema concettuale, e poi modificate per renderle coerenti con lo schema concettuale ristrutturato. È ora necessario riformulare le operazioni e le informazioni sul carico applicativo in modo che siano coerenti con lo schema logico
- La riformulazione viene condotta semplicemente tenendo presente come le entità e le relazioni dello schema Entità-Relazione ristrutturato sono state tradotte nello schema relazionale



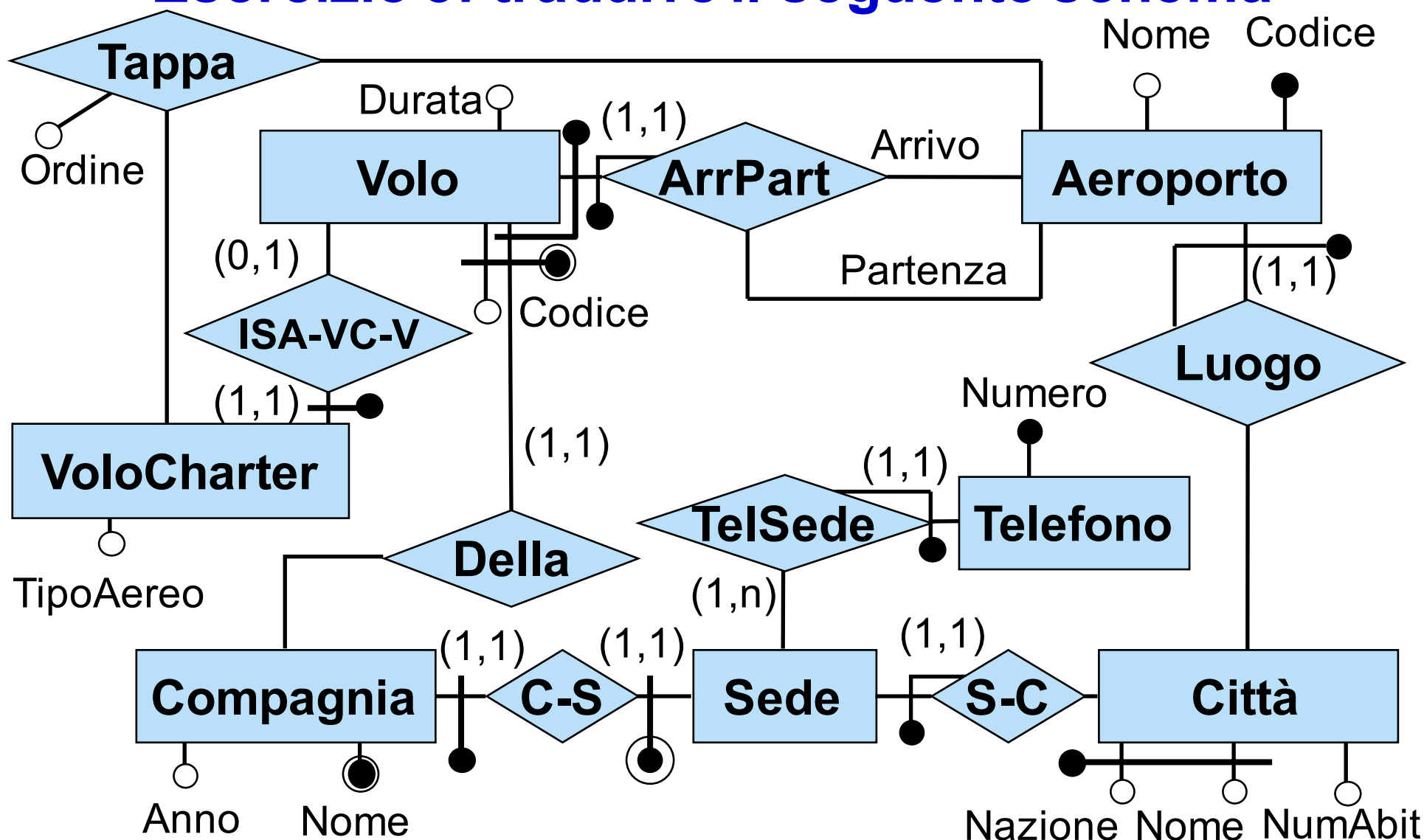
# Riassunto sulla traduzione diretta

1. Traduzione di ogni entità in una relazione, con i seguenti attributi:
  - gli attributi dell'entità stessa
  - gli attributi delle relazioni che partecipano all'identificazione principale esterna dell'entità, insieme alle chiavi primarie, opportunamente nominate (possibilmente con ruolo), delle entità connesse a tali relazioni (si noti che in questo caso, per l'assenza di cicli sull'identificazione principale esterna, la relazione non può avere altre entità per le quali la relazione è parte di identificatore esterno)
2. Traduzione di ogni ER-relazione (non accorpata al punto 1) in relazione, con opportuna chiave primaria, e con i seguenti attributi:
  - gli identificatori principali delle entità partecipanti (con opportuno nome)
  - gli attributi della ER-relazione
3. Traduzione di vincoli
  - not null per gli attributi obbligatori
  - chiavi (primarie e non)
  - foreign key che provengono dall'accorpamento (vedi punto 1), da tipizzazione di relazioni, da ISA di relazioni
  - vincoli di generalizzazione
  - vincoli di cardinalità (parte obbligatoria diventa vincolo di inclusione, parte di funzionalità diventa vincolo di chiave)
  - altri vincoli esterni (in particolare i vincoli di join da entità con identificatore esterno non principale)
4. Riformulazione di operazioni e specifiche sul carico applicativo in termini dello schema logico





## Esercizio 5: tradurre il seguente schema



**Vincolo esterno:** per ogni  $v$  in **VoloCharter**, se  $(v, a_1), \dots, (v, a_n)$  sono tutte le coppie in **Tappa** alle quali partecipa  $v$ , e se  $o_1, \dots, o_n$  sono i valori assegnati a tali coppie dall'attributo *Ordine*, allora per  $i=1, \dots, n$  esiste un  $o_j$  tale che  $o_j=i$ .



## Esercizio 5: soluzione (parte 1)

Volo(Codice, Comp, Durata)

foreign key: Volo[Comp]  $\subseteq$  Compagnia[Nome]

foreign key: Volo[Codice, Comp]  $\subseteq$  ArrPart[Codice, Comp]

ArrPart(Codice, Comp, Arrivo, Partenza)

foreign key: ArrPart[Arrivo]  $\subseteq$  Aeroporto[Codice]

foreign key: ArrPart[Partenza]  $\subseteq$  Aeroporto[Codice]

foreign key: ArrPart[Codice, Comp]  $\subseteq$  Volo[Codice, Comp]

chiave: Comp, Arrivo, Partenza ←

VoloCharter(Codice, Comp, TipoAereo)

foreign key:

VoloCharter[Codice, Comp]  $\subseteq$  Volo[Codice, Comp]

Aeroporto(Codice, Nome)

foreign key: Aeroporto[Codice]  $\subseteq$  LuogoAeroporto[Aeroporto]

LuogoAeroporto(Aeroporto, NomeCittà, NazCittà)

foreign key: LuogoAeroporto[Aeroporto]  $\subseteq$  Aeroporto[Codice]

foreign key: LuogoAeroporto[NomeCittà, NazCittà]  $\subseteq$   
Città[Nome, Nazione]

Città(Nome, Nazione, NumAbitanti)

Nota: viene dall'identificatore esterno non principale di Volo. In questo caso non è necessario ricorrere ad un vincolo sul join perché tutti gli attributi rilevanti si trovano nella relazione ArrPart



## Esercizio 5: soluzione (parte 2)

Compagnia(Nome, AnnoFond)

foreign key: Compagnia[Nome]  $\subseteq$  Sede[Comp]

Sede(Comp)

foreign key: Sede[Comp]  $\subseteq$  Compagnia[Nome]

foreign key: Sede[Comp]  $\subseteq$  CittàSede[Comp]

inclusione: Sede[Comp]  $\subseteq$  TelefonoComp[Comp]

CittàSede(Comp, NomeCittà, NazCittà)

foreign key: CittàSede[Comp]  $\subseteq$  Sede[Comp]

foreign key: CittàSede[NomeCittà, NazCittà]  $\subseteq$  Città[Nome, Nazione]

TelefonoComp(Numero, Comp)

foreign key: TelefonoComp[Comp]  $\subseteq$  Sede[Comp]

foreign key: TelefonoComp[Numero]  $\subseteq$  Telefono[Numero]

Telefono(Numero)

foreign key: Telefono[Numero]  $\subseteq$  TelefonoComp[Numero]

Tappa(CodVoloCharter, Comp, Aeroporto, Ordine)

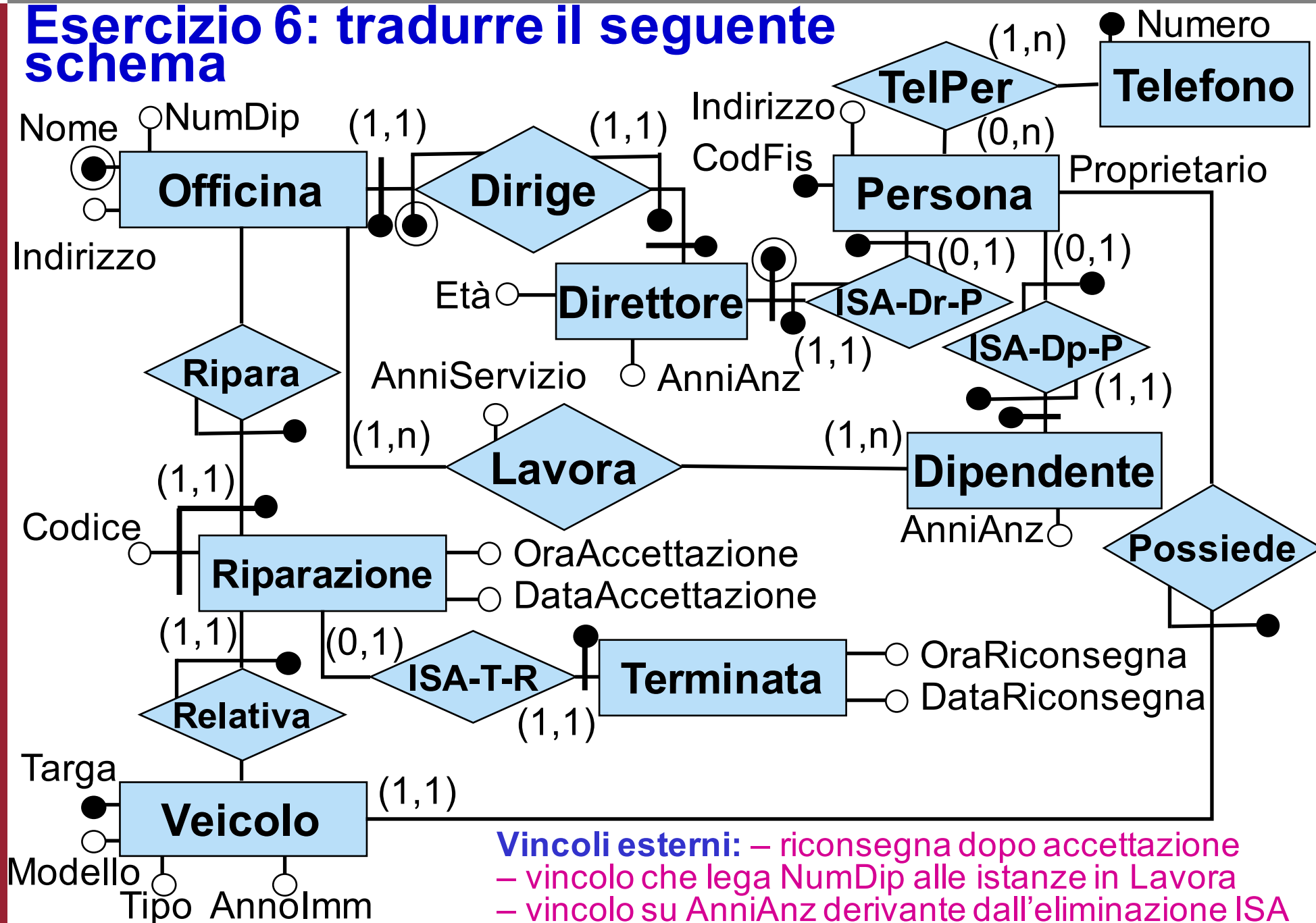
foreign key: Tappa[CodVoloCharter, Comp]  $\subseteq$  VoloCharter[Codice, Comp]

foreign key: Tappa[Aeroporto]  $\subseteq$  Aeroporto[Codice]

**Vincolo esterno:** per ogni  $v$  in Tappa[CodVoloCharter], se  $o_1, \dots, o_n$  sono i valori che compaiono nell'attributo Ordine nelle tuple di Tappa che hanno  $v$  come valore nell'attributo CodVoloCharter, allora per  $i=1, \dots, n$  esiste uno ed un solo  $o_j$  tale che  $o_j=i$ .



## Esercizio 6: tradurre il seguente schema





## Esercizio 6: soluzione (parte 1)

Officina(Nome, NumDip, Indirizzo)

foreign key: Officina[Nome]  $\subseteq$  Dirige[Officina]

inclusione: Officina[Nome]  $\subseteq$  Lavora[Officina]

Persona(CodFis, Indirizzo)

Direttore(CodFis, Età, AnniAnz)

foreign key: Direttore[CodFis]  $\subseteq$  Persona[CodFis]

foreign key: Direttore[CodFis]  $\subseteq$  Dirige[Direttore]

Dipendente(CodFis, AnniAnz)

foreign key: Dipendente[CodFis]  $\subseteq$  Persona[CodFis]

inclusione: Dipendente[CodFis]  $\subseteq$  Lavora[Dipendente]

Dirige(Officina, Direttore)

foreign key: Dirige[Officina]  $\subseteq$  Officina[Nome]

foreign key: Dirige[Direttore]  $\subseteq$  Direttore[CodFis]

chiave: Direttore

Lavora(Officina, Dipendente, AnniServizio)

foreign key: Lavora[Officina]  $\subseteq$  Officina[Nome]

foreign key: Lavora[Dipendente]  $\subseteq$  Dipendente[CodFis]

TelPer(CodFis, Telefono)

foreign key: TelPer[CodFis]  $\subseteq$  Persona[CodFis]

foreign key: TelPer[Telefono]  $\subseteq$  Telefono[Numero]



## Esercizio 6: soluzione (parte 2)

Telefono(Numero)

inclusione: Telefono[Numero]  $\subseteq$  TelPer[Telefono]

Veicolo(Targa, Modello, Tipo, AnnoImm)

foreign key : Veicolo[Targa]  $\subseteq$  Possiede[Veicolo]

Possiede(Veicolo, Proprietario)

foreign key: Possiede[Veicolo]  $\subseteq$  Veicolo[Targa]

foreign key: Possiede[Proprietario]  $\subseteq$  Persona[CodFis]

Riparazione(Codice, Officina, OraAcc, DataAcc)

foreign key: Riparazione[Officina]  $\subseteq$  Officina[Nome]

foreign key: Riparazione[Codice,Officina]  $\subseteq$  Relativa[Codice,Officina]

Relativa(Codice, Officina, Veicolo)

foreign key: Relativa[Codice,Officina]  $\subseteq$  Riparazione[Codice,Officina]

foreign key: Relativa[Veicolo]  $\subseteq$  Veicolo[Targa]

Terminata(Codice, Officina, OraRic, DataRic)

foreign key: Terminata[Codice,Officina]  $\subseteq$  Riparazione[Codice,Officina]

### Vincoli esterni:

- riconsegna dopo accettazione
- vincolo che lega Officina[NumDip] alle istanze in Lavora
- vincolo su AnniAnz di Direttore e Dipendente derivante dall'eliminazione ISA



## 5. La progettazione logica

### 5.4 ristrutturazione dello schema logico

1. introduzione alla progettazione logica
2. ristrutturazione dello schema ER
3. traduzione diretta nel modello relazionale
- 4. ristrutturazione dello schema logico**



## Cosa sappiamo dopo la traduzione

- Abbiamo rispettato la modularizzazione concettuale
- Si possono presentare potenziali problemi di efficienza rispetto allo spazio
  - valori nulli (solo quelli dovuti ad attributi opzionali)
  - ci possono essere due relazioni  $R_1$  e  $R_2$  con chiavi  $K_1$  e  $K_2$  tali che valga sia  $R_1[K_1] \subseteq R_2[K_2]$  sia  $R_2[K_2] \subseteq R_1[K_1]$
  - ridondanze lasciate
- Si possono presentare potenziali problemi di efficienza rispetto al tempo di esecuzione delle query e degli update
  - ridondanze lasciate
  - numero e struttura delle relazioni
- Ci devono essere dei buoni motivi legati all'**efficienza** per cambiare le scelte fatte. Ma se queste ragioni ci sono, si tratta adesso di ristrutturare lo schema in modo che la base di dati sia più efficiente rispetto alle operazioni dell'applicazione.





## Modello di costo

- Una relazione occupa un certo numero di pagine di memoria secondaria
- Il numero di accessi alle pagine della memoria secondaria domina sull'elaborazione in memoria centrale
- Un accesso a memoria secondaria avviene ad una pagina intera
- Il numero complessivo di pagine accedute allo scopo di effettuare un'operazione dipende in generale:
  - dal tipo di operazione (lettura o scrittura)
  - dal numero di tuple delle relazioni coinvolte (influisce sul numero di pagine delle relazioni coinvolte)
  - dal numero di tuple per pagina di memoria secondaria (influisce sul numero di pagine delle relazioni coinvolte)



## Modello di costo: esempi

Per una relazione **R** denotiamo con

- **$N_P(R)$**  il numero di pagine in memoria secondaria occupate da **R**
- **$N_{TP}(R)$**  il numero di tuple per ogni pagina di **R**

Stima **di massima** del costo delle operazioni:

- la selezione su una relazione **R** ha costo pari a  **$N_P(R)$**
- la proiezione su una relazione **R** ha costo pari a  **$N_P(R)$**
- il join di **R** con **Q** si basa su un doppio ciclo
  - se non ci sono indici, allora il costo è  **$N_P(R) \times N_P(Q)$** :  
per ogni pagina **p1** di **R**, per ogni pagina **p2** di **Q**  
per ogni tupla **r** di **R** in **p1**, per ogni tupla **q** di **Q** in **p2**  
se **r** e **q** sono in join, metti la tupla nel risultato
  - se **Q** ha un indice molto selettivo sull'attributo di join,  
allora il costo è  **$N_P(R) + N_P(R) \times N_{TP}(R)$**



# Criteri generali per individuare potenziali problemi

- relazione con
  - tante tuple → la relazione occupa molte pagine
  - tanti attributi → una pagina contiene poche tuple ed il numero di pagine aumenta
- attributi con tanti valori nulli
  - spreco di spazio → una pagina contiene poche tuple
- la proiezione è costosa (quando una pagina contiene poche tuple)
- il join è costoso (quasi sempre)
- la verifica di vincoli è costosa

## Conclusione:

per una relazione  $R$  rilevante (con tante tuple) occorre tentare di tenere basso  $N_P(R)$ , al limite aumentando  $N_{TP}(R)$ , perché, a parità di altre condizioni, poche tuple in ogni pagina significa un numero maggiore di pagine. Aumentare  $N_{TP}(R)$ , può significare spezzare (**decomporre**) le relazioni. D'altra parte, per favorire l'efficienza dei join potremmo essere indotti ad **accorpare** le relazioni.



# Ristrutturazioni dello schema logico

- **Decomposizione**

- Verticale (sempre sulla chiave)
  - per facilitare l'accesso (con selezioni e proiezioni)
  - per normalizzazione (ignoreremo questo aspetto)
- Orizzontale
  - per facilitare l'accesso (con selezioni)
- Mista
  - per evitare valori nulli

- **Accorpamento**

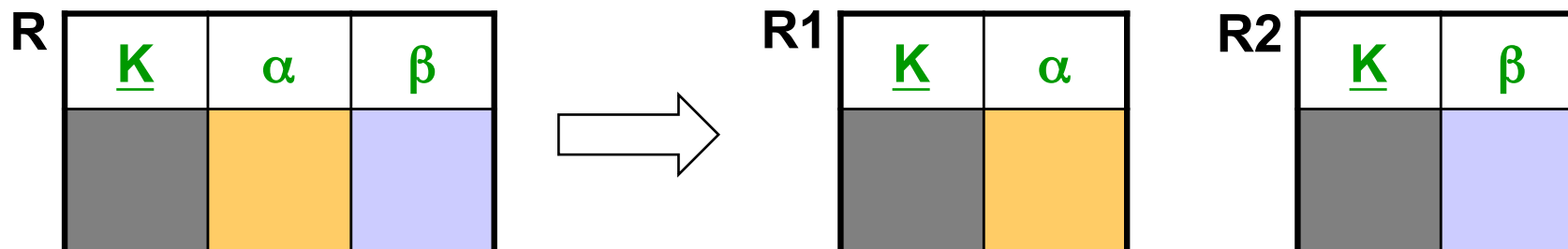
- per facilitare l'accesso (evita join)
- per eliminare relazioni inutili
- per eliminare attributi inutili

Le relazioni dello schema originario possono essere ricostruite attraverso la definizione di opportune viste (view)

Nota: le ristrutturazioni si applicano in presenza di determinati attributi che formano una chiave di relazione. Sulle slide indicheremo tali attributi con **K**, intendendo che **K** è una chiave (composta da uno o più attributi) della relazione corrispondente



# Decomposizione verticale per facilitare l'accesso



## Vincoli dello schema ristrutturato:

- foreign key:  $R1[K] \subseteq R2[K]$   
foreign key:  $R2[K] \subseteq R1[K]$
  - vincoli di inclusione da e verso  $R$  si riformulano su  $R1$  (o  $R2$ )
  - tutti gli altri vincoli che coinvolgono  $R$  vanno riformulati
- 
- Si applica quando gli accessi ad  $R$  avvengono prevalentemente in modo separato sugli attributi  $\alpha$  rispetto agli attributi  $\beta$
  - $N_p(R1)$  e  $N_p(R2)$  sono più bassi rispetto a  $N_p(R)$
  - La relazione  $R$  può essere ricostruita attraverso una vista che calcola il join tra  $R1$  ed  $R2$  su  $K$  (ma ovviamente l'esigenza di ricostruire il join deve essere rara)



## Decomposizione verticale: esempio

Immobile(Codice, Particella, Zona, Indirizzo, Città, Mq, NumVani, Valore)

foreign key: Immobile[Città]  $\subseteq$  Città [Cod]

Agenzia(CodiceAg, Immobile)

foreign key: Agenzia[Immobile]  $\subseteq$  Immobile[Codice]

Supponiamo che ai dati catastali degli immobili (particella, zona, indirizzo, città) si acceda prevalentemente in modo separato rispetto ai dati commerciali (metri quadri, numero di vani, valore). Appliciamo quindi la decomposizione verticale, ed otteniamo:

ImmobileCatasto(Codice, Particella, Zona, Indirizzo, Città)

foreign key: ImmobileCatasto[Città]  $\subseteq$  Città [Cod]

foreign key: ImmobileCatasto[Codice]  $\subseteq$  ImmobileComm [Codice]

ImmobileComm(Codice, Mq, NumVani, Valore)

foreign key: ImmobileComm[Codice]  $\subseteq$  ImmobileCatasto[Codice]

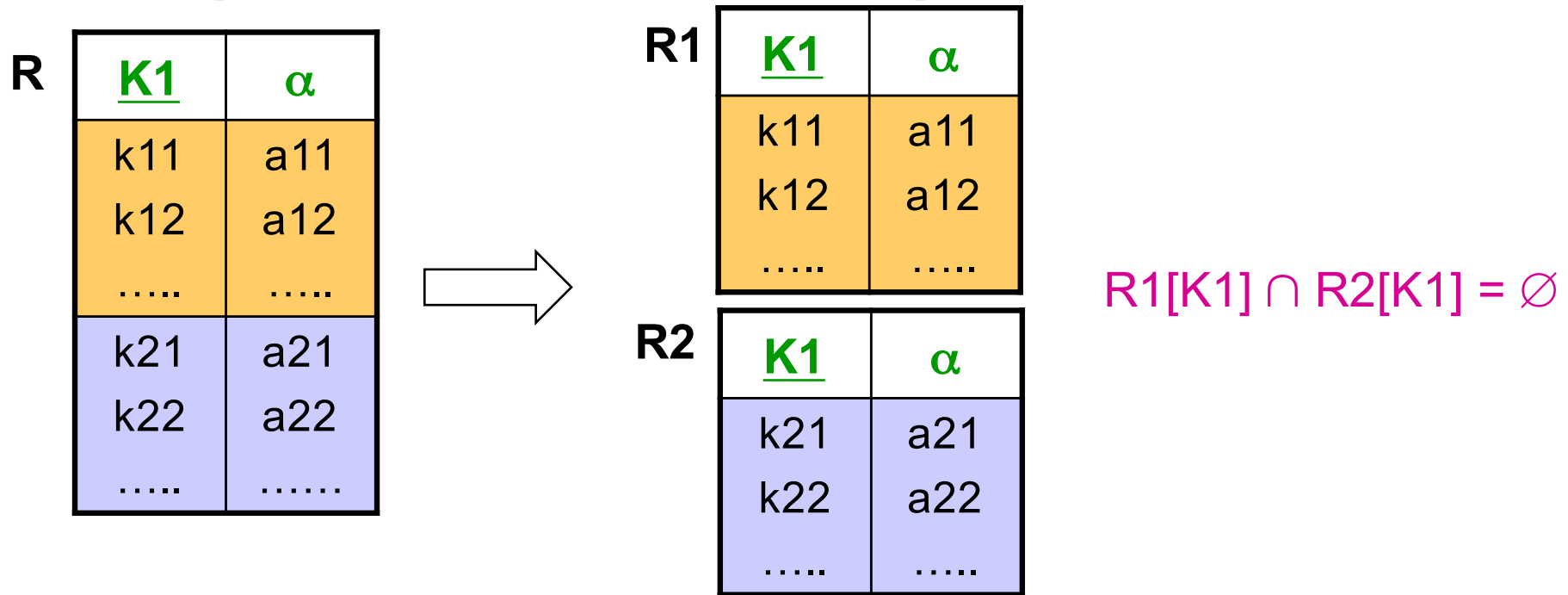
Agenzia(CodiceAg, Immobile)

foreign key: Agenzia[Immobile]  $\subseteq$  ImmobileCatasto[Codice]

Si noti come la relazione **Immobile** originaria si possa sempre ricostruire mediante l'equi-join tra **ImmobileCatasto** e **ImmobileComm** sull'attributo **Codice**



# Decomposizione orizzontale per facilitare l'accesso



## Ulteriori vincoli dello schema ristrutturato:

- vincoli di inclusione da **R** diventano vincoli di inclusione da **R1** e da **R2**
- vincoli di inclusione a **R** diventano vincoli di inclusione a **R1**  $\cup$  **R2**
- tutti gli altri vincoli che coinvolgono **R** vanno riformulati
- Si applica quando gli accessi alle tuple di **R** di una delle due “fasce” avvengono separatamente dagli accessi alle tuple dell'altra fascia
- **N<sub>P</sub>(R1)** e **N<sub>P</sub>(R2)** sono più bassi rispetto a **N<sub>P</sub>(R)**
- **R** può essere ricostruita attraverso una vista che calcola l'unione di **R1** ed **R2**



# Decomposizione orizzontale: esempio

Telefonata(Codice, OrarioInizio, OrarioFine, UtenzaInvio, UtenzaDestinazione)

foreign key: Telefonata[UtenzaInvio]  $\subseteq$  Utenza[Cod]

foreign key: Telefonata[UtenzaDestinazione]  $\subseteq$  Utenza[Cod]

Centrale(Codice, Telefonata)

foreign key: Centrale[Telefonata]  $\subseteq$  Telefonata[Codice]

Supponiamo che alle telefonate si acceda per fasce orarie (giorno, sera, notte) determinate sul tempo di inizio. Applichiamo quindi la decomposizione orizzontale (opportuni vincoli detteranno le regole per i valori corretti del campo OrarioInizio delle tabelle e realizzeranno quindi i vincoli di disgiuntezza), ed otteniamo:

TelefonataGiorno(Codice, OrarioInizio, OrarioFine, UtenzaInvio, UtenzaDestinazione)

foreign key: TelefonataGiorno[UtenzaInvio]  $\subseteq$  Utenza[Cod]

foreign key: TelefonataGiorno[UtenzaDestinazione]  $\subseteq$  Utenza[Cod]

TelefonataSera(Codice, OrarioInizio, OrarioFine, UtenzaInvio, UtenzaDestinazione)

foreign key: TelefonataSera[UtenzaInvio]  $\subseteq$  Utenza[Cod]

foreign key: TelefonataSera[UtenzaDestinazione]  $\subseteq$  Utenza[Cod]

TelefonataNotte(Codice, OrarioInizio, OrarioFine, UtenzaInvio, UtenzaDestinazione)

foreign key: TelefonataNotte[UtenzaInvio]  $\subseteq$  Utenza[Cod]

foreign key: TelefonataNotte[UtenzaDestinazione]  $\subseteq$  Utenza[Cod]

Centrale(Codice, Telefonata)

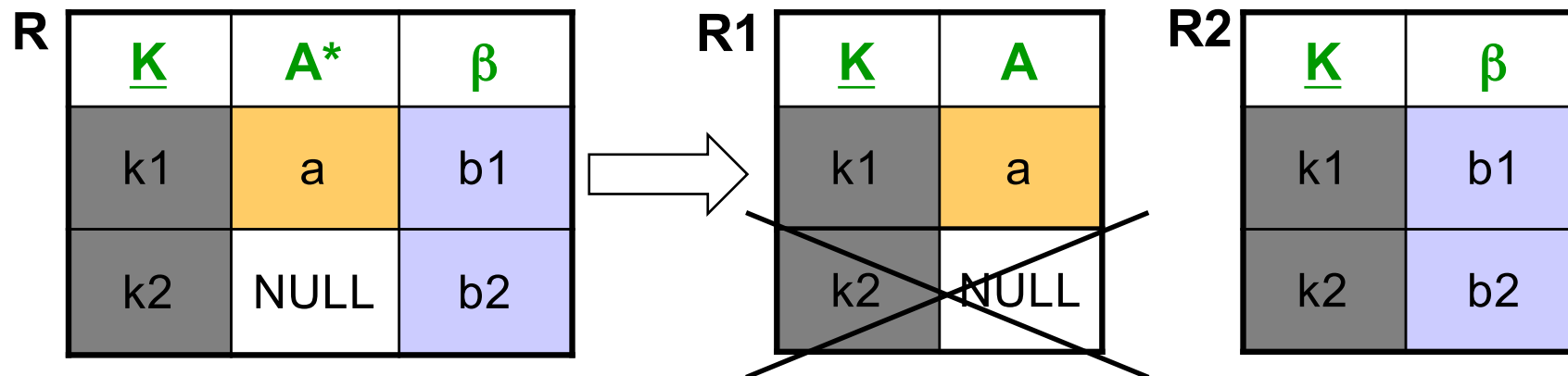
Vincolo esterno:

Centrale[Telefonata]  $\subseteq$  TelefonataGiorno[Codice]  $\cup$  TelefonataSera[Codice]  $\cup$  TelefonataNotte[Codice]





## Decomposizione mista per evitare valori nulli



**A** è l'attributo su cui si vogliono evitare valori nulli, e  $\beta$  denota un insieme di attributi. È una decomposizione verticale seguita da una orizzontale su **R1**, seguita a sua volta dall'eliminazione della parte di **R1** con valori nulli in **A**.

### Vincoli dello schema ristrutturato:

- foreign key:  $R1[K] \subseteq R2[K]$
- vincoli di inclusione da e verso **R** diventano inclusioni da e verso **R2**
- tutti gli altri vincoli che coinvolgono **R** vanno riformulati
- Si applica quando si prevede che la relazione abbia tanti valori nulli in **A**
- $N_p(R1)$  e  $N_p(R2)$  sono più bassi rispetto a  $N_p(R)$
- **R** può essere ricostruita attraverso una vista che calcola il join esterno tra **R1** ed **R2** su **K**.



## Decomposizione mista: esempio

Persona(CodFiscale, CittàNascita, NumTel\*, DataMatrimonio\*)

foreign key: Persona[CittàNascita]  $\subseteq$  Città[Cod]

Città(Cod, Sindaco)

foreign key: Città[Sindaco]  $\subseteq$  Persona[CodFiscale]

Supponiamo di non volere valori nulli. Applichiamo quindi due volte in cascata la decomposizione mista, ed otteniamo:

Persona(CodFiscale, CittàNascita)

foreign key: Persona[CittàNascita]  $\subseteq$  Città[Cod]

PersonaConTelefono(CodFiscale, NumTel)

foreign key: PersonaConTelefono[CodFiscale]  $\subseteq$  Persona[CodFiscale]

PersonaSposata(CodFiscale, DataMatrimonio)

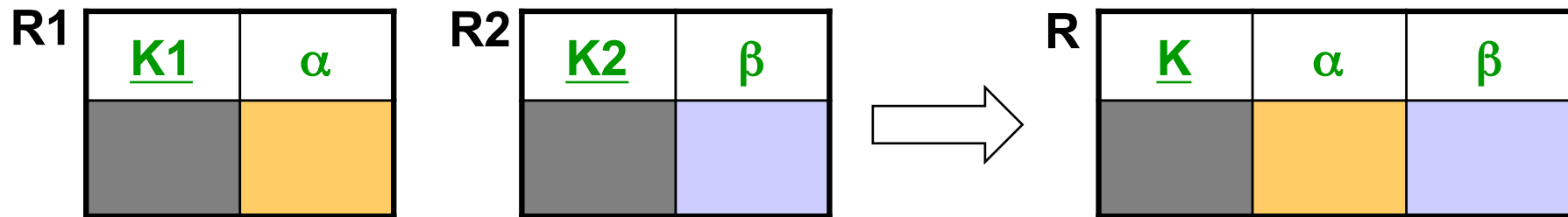
foreign key: PersonaSposata[CodFiscale]  $\subseteq$  Persona[CodFiscale]

Città(Cod, Sindaco)

foreign key: Città[Sindaco]  $\subseteq$  Persona[CodFiscale]



# Accorpamento per facilitare l'accesso: caso 1



foreign key:  $R1[K1] \subseteq R2[K2]$

foreign key:  $R2[K2] \subseteq R1[K1]$

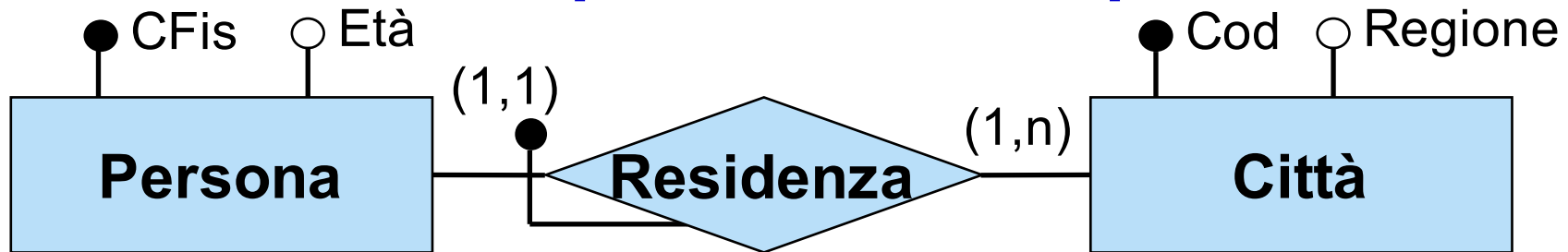
Si noti che **K1** e **K2** sono chiavi, anche non primarie. Si noti anche che se **beta** manca, **R** coincide con **R1**, e l'effetto è quello di eliminare **R2**.

## Vincoli dello schema ristrutturato:

- tutti i vincoli che coinvolgono **R1** o **R2** vanno riformulati su **R**
- Si applica per facilitare gli accessi a **R1** e **R2** quando questi avvengono prevalentemente insieme e richiedono di calcolare il join tra **R1** e **R2** con **K1=K2**; in altre parole si applica per evitare il join tra **R1** e **R2**
- Le relazioni **R1** e **R2** possono essere ricostruite attraverso due viste che calcolano rispettivamente le proiezioni di **R** su **(K, alpha)** e su **(K, beta)**
- Quando non c'è **beta**, serve ad eliminare una relazione inutile (cioè **R2**), che peraltro può essere ricostruita attraverso la proiezione di **R** su **K**



## Accorpamento: esempio 1



Persona(CFis, Età)

foreign key: Persona[CFis]  $\subseteq$  Residenza[Persona]

Residenza(Persona, Città)

foreign key: Residenza[Persona]  $\subseteq$  Persona[CFis]

foreign key: Residenza[Città]  $\subseteq$  Città[Cod]

Città(Cod, Regione)

inclusione: Città[Cod]  $\subseteq$  Residenza[Città]

Supponiamo che quando si accede ad una persona si acceda spesso anche alla sua città di residenza. Applichiamo quindi l'accorpamento, ed otteniamo:

Persona(CFis, Età, Città)

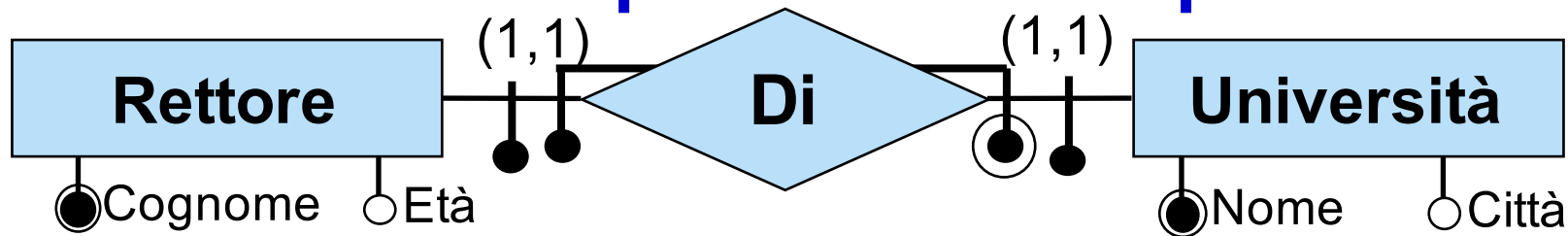
foreign key: Persona[Città]  $\subseteq$  Città[Cod]

Città(Cod, Regione)

inclusione: Città[Cod]  $\subseteq$  Persona[Città]



## Accorpamento: esempio 2



Rettore(Cognome, Età)

foreign key:  $\text{Rettore}[\text{Cognome}] \subseteq \text{Di}[\text{Rettore}]$

Di(Rettore, Università)

foreign key:  $\text{Di}[\text{Rettore}] \subseteq \text{Rettore}[\text{Cognome}]$

foreign key:  $\text{Di}[\text{Università}] \subseteq \text{Università}[\text{Nome}]$

chiave: **Rettore**

Università(Nome, Città)

foreign key:  $\text{Università}[\text{Nome}] \subseteq \text{Di}[\text{Università}]$

Supponiamo che quando si accede ad una università si acceda anche al suo rettore. Appliciamo l'accorpamento di **Università** e **Di**, ed otteniamo

Rettore(Cognome, Età)

foreign key:  $\text{Rettore}[\text{Cognome}] \subseteq \text{Università}[\text{Rettore}]$

Università(Nome, Città, Rettore)

foreign key:  $\text{Università}[\text{Rettore}] \subseteq \text{Rettore}[\text{Cognome}]$

chiave: **Rettore**

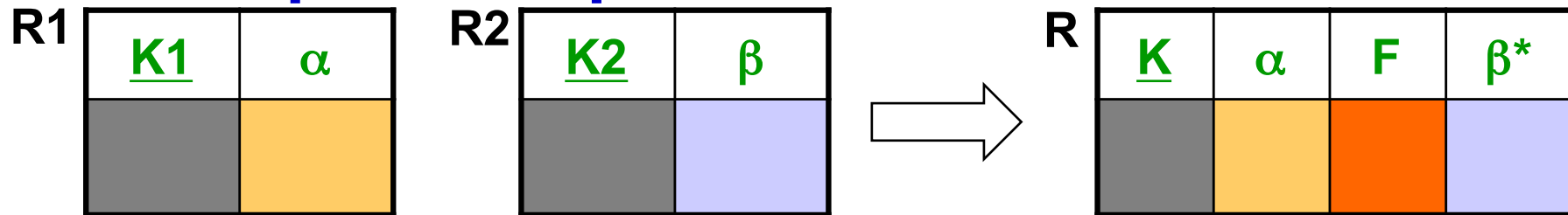
Supponiamo che quando si accede ad un rettore si acceda anche ai dati relativi alla sua università. Appliciamo un ulteriore accorpamento ed otteniamo

UniversitàRettore(NomeUniversità, CittàUniversità, CognomeRettore, EtàRettore)

chiave: **CognomeRettore**



## Accorpamento per facilitare l'accesso: caso 2



foreign key:  $R2[K2] \subseteq R1[K1]$

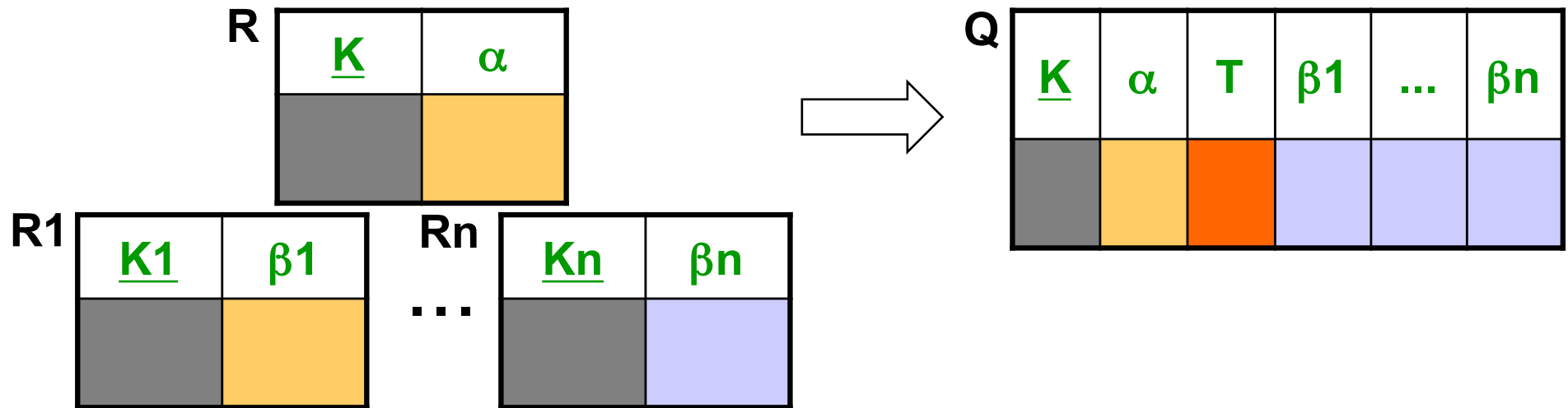
Si noti che **K1** e **K2** sono chiavi, anche non primarie. Questo caso è una variante del caso precedente, in cui vale solo una foreign key da **R2** a **R1**. L'attributo **F** (flag) è booleano, e vale true nelle tuple corrispondenti a tuple di **R2**, false nelle altre. Spesso si esegue quando  **$\beta$**  manca.

### Vincoli dello schema ristrutturato:

- vincolo che dice che se **F** è false, allora  **$\beta$**  è NULL
- tutti i vincoli che coinvolgono **R1** o **R2** vanno riformulati su **R**
- Si applica per facilitare gli accessi a **R1** e **R2** quando questi avvengono prevalentemente insieme e richiedono di calcolare il join tra **R1** e **R2** con **K1=K2**; in altre parole si applica per evitare il join tra **R1** e **R2**
- Le relazioni **R1** e **R2** possono essere ricostruite attraverso due viste che calcolano rispettivamente la proiezione di **R** su **(K,  $\alpha$ )** e la proiezione su **(K,  $\beta$ )** solo per quelle tuple per cui **F** è true.
- Quando non c'è  **$\beta$** , serve ad eliminare una relazione (cioè **R2**), che peraltro può essere ricostruita mediante l'attributo **F**.



## Accorpamento per facilitare l'accesso: caso 3



foreign key:  $R1[K1] \subseteq R[K]$  ... foreign key:  $Rn[Kn] \subseteq R[K]$

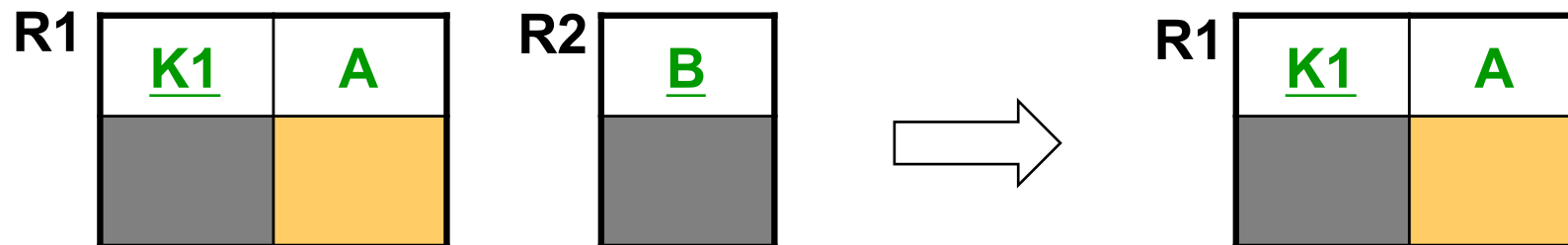
Questo caso può essere visto come “accorpamento orizzontale”, l'inverso della decomposizione orizzontale. L'attributo **T** (tipo) è enumerato e può assumere i valori 1...n corrispondenti a **R1**, ..., **Rn**. Esso vale “i” nelle tuple corrispondenti a tuple di **Ri**.

### Vincoli dello schema ristrutturato:

- vincolo che mette in relazione il valore di **T** con le tuple di **R1**, ..., **Rn**
- tutti i vincoli che coinvolgono **R1** o **R2** vanno riformulati su **R**
- Si applica quando le varie **Ri** sono disgiunte a coppie e serve a facilitare gli accessi a **Ri** e **R** quando questi avvengono prevalentemente insieme e richiedono di calcolare il join tra **R** e **Ri** con  $K=Ki$ ; in altre parole si applica per evitare il join tra **R** e le varie **Ri**.
- Le relazioni **Ri** possono essere ricostruite attraverso viste.



## Accorpamento per eliminare relazioni inutili



foreign key:  $R1[A] \subseteq R2[B]$

inclusione:  $R2[B] \subseteq R1[A]$

Si noti che **B** è chiave in **R2**, ed **A** può esserlo o no in **R1** (nella figura è mostrato il caso in cui non lo sia, ma la regola vale anche nel caso in cui **A** sia chiave o parte di chiave e questo corrisponde all'accorpamento - caso 2).

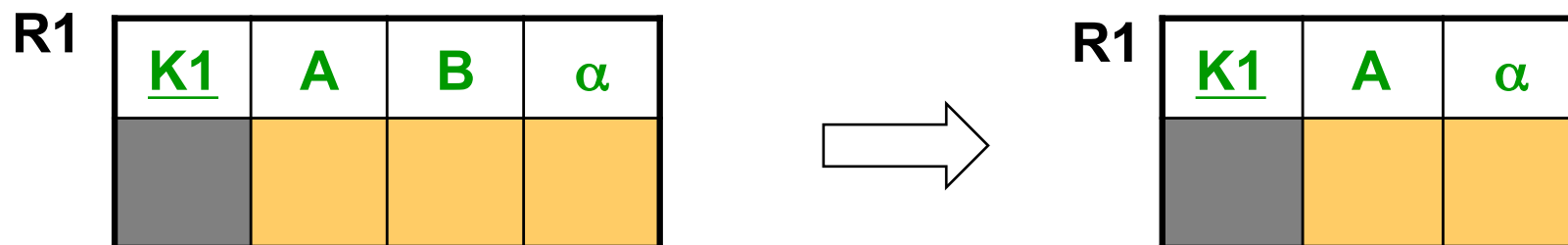
### Vincoli dello schema ristrutturato:

- tutti i vincoli che coinvolgono **R2** vanno riformulati su **R1**
- Si applica per eliminare la relazione **R2** che è inutile, visto che tutti i valori nell'unico suo attributo si ritrovano nell'attributo **A** della relazione **R1**
- La relazione **R2** può essere ricostruita attraverso la proiezione della relazione **R1** sull'attributo **A**





# Accorpamento per eliminare attributi inutili



Vincolo di tupla:

per ogni tupla  $t$  di R1,  $t[A]=t[B]$

Si noti che **A,B** sono attributi qualunque, oppure insiemi di attributi (in questo caso,  $t[A]$  e  $t[B]$  saranno tuple, e non singoli valori).

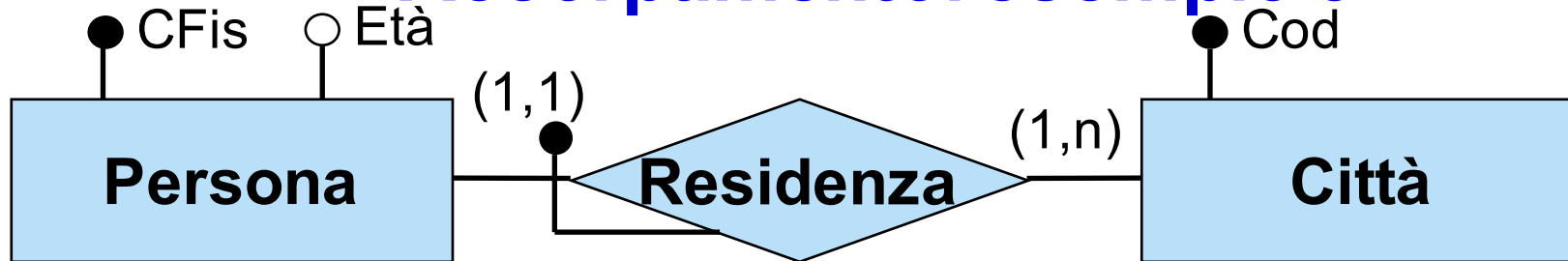
## Vincoli dello schema ristrutturato:

– tutti i vincoli che coinvolgono **B** vanno riformulati su **A**

- Si applica per eliminare attributi inutili
- La relazione originale può essere ricostruita attraverso un'opportuna vista, che per ogni tupla della relazione “duplica” il valore di **A** in un nuovo attributo **B**.



## Accorpamento: esempio 3



Persona(CFis, Età)

foreign key:  $\text{Persona}[\text{CFis}] \subseteq \text{Residenza}[\text{Persona}]$

Residenza(Persona, Città)

foreign key:  $\text{Residenza}[\text{Persona}] \subseteq \text{Persona}[\text{CFis}]$

foreign key:  $\text{Residenza}[\text{Città}] \subseteq \text{Città}[\text{Cod}]$

Città(Cod)

inclusione:  $\text{Città}[\text{Cod}] \subseteq \text{Residenza}[\text{Città}]$

Supponiamo che quando si accede alla persone si acceda spesso anche alla sua città di residenza. Applichiamo quindi l'accorpamento, ed otteniamo:

Persona(CFis, Età, Città)

foreign key:  $\text{Persona}[\text{Città}] \subseteq \text{Città}[\text{Cod}]$

Città(Cod)

inclusione:  $\text{Città}[\text{Cod}] \subseteq \text{Persona}[\text{Città}]$

Applichiamo quindi di nuovo l'accorpamento per eliminare la relazione inutile (Città), ed otteniamo:

Persona(CFis, Età, Città)



# Traduzione in SQL

Ogni relazione viene definita tramite la “create table”, nel modo ovvio. In particolare, i vincoli vengono tradotti secondo queste indicazioni generali:

- attributi obbligatori: **not null**
- chiave primaria: **primary key**
- chiave: **unique**
- foreign key: **foreign key**
- interdipendenza valori nulli:

**check ((A is null and B is null) or  
(A is not null and B is not null))**

- inclusione: **check (A in (select B from R))**
- disgiunzione: **check (A not in (select B from R))**
- cardinalità (i,j):

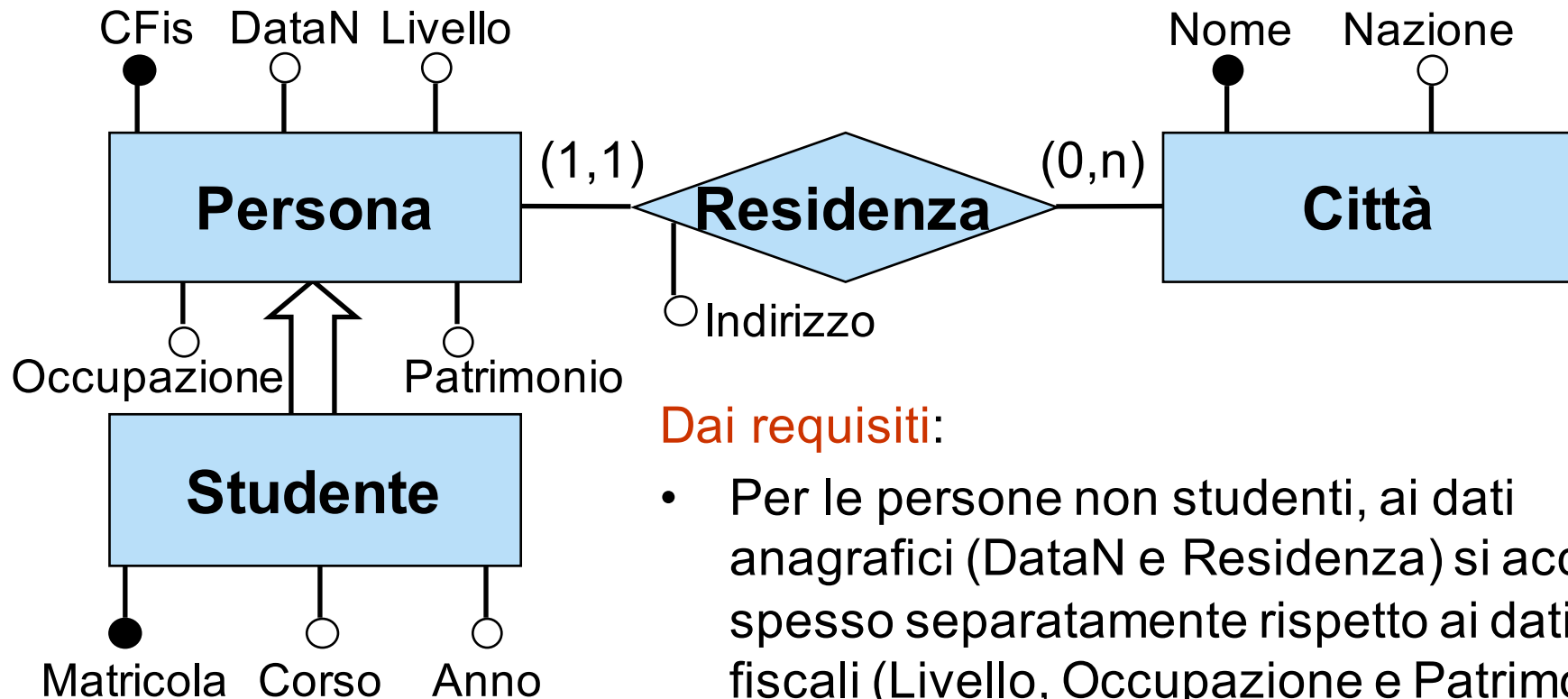
**check ((i ≤ (select count(\*) from R where ...)) and  
(j ≥ (select count(\*) from R where ...)))**

- vincoli esterni: **assertion** o controllati a livello di applicazione



# Esercizio 7: effettuare la progettazione logica

Schema concettuale:

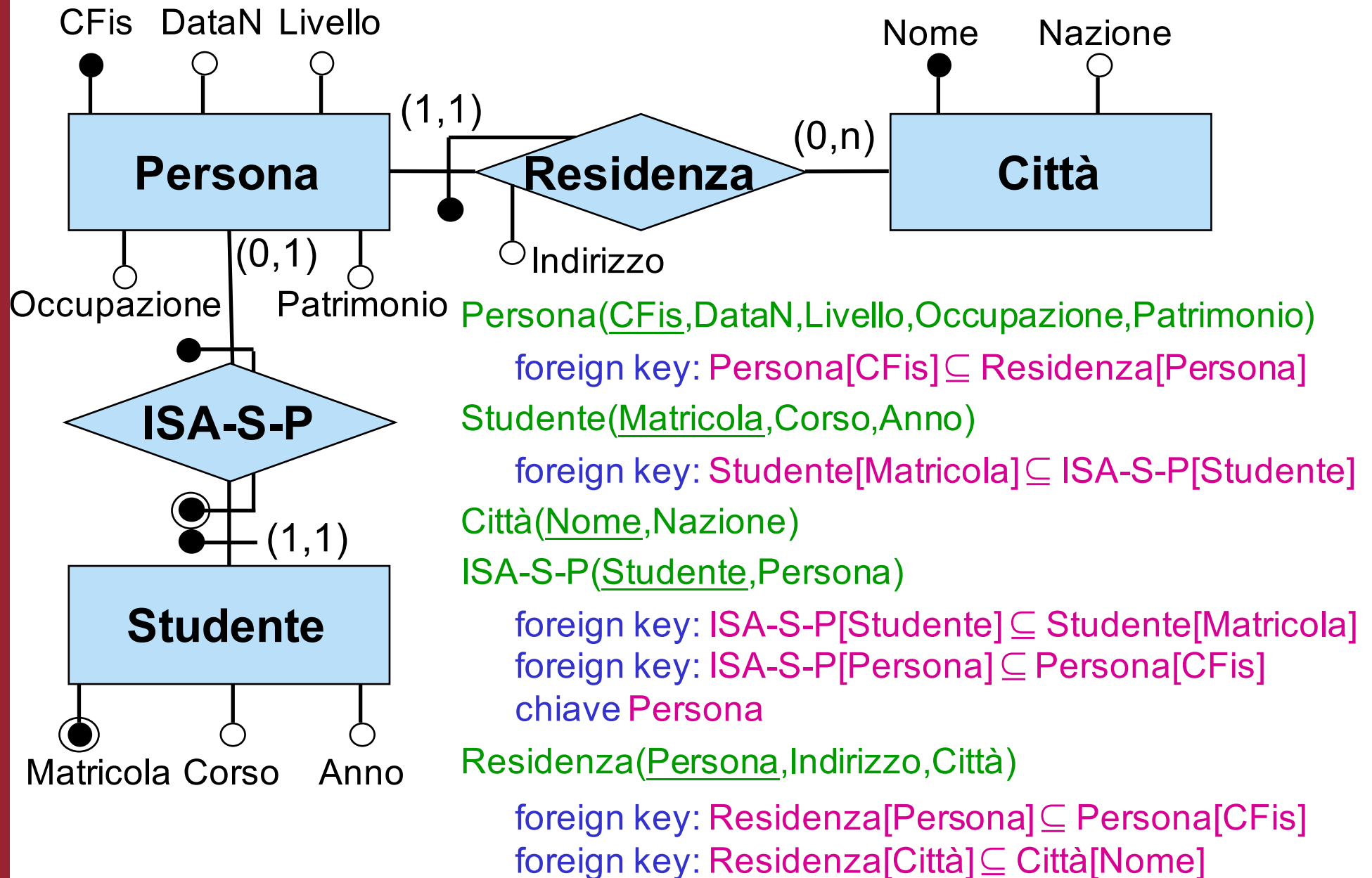


Dai requisiti:

- Per le persone non studenti, ai dati anagrafici (DataN e Residenza) si accede spesso separatamente rispetto ai dati fiscali (Livello, Occupazione e Patrimonio)
- Agli studenti si accede prevalentemente per matricola, e si accede spesso ai dati universitari (Corso, Anno) insieme alla data di nascita e ai dati fiscali (Livello, Occupazione e Patrimonio)



## Esercizio 7: ristrutturazione e traduzione diretta





## Esercizio 7: ristrutturazioni dello schema logico

Per le persone non studenti valgono criteri di accesso diversi rispetto alle persone che sono studenti, anche relativamente ai dati riguardanti la residenza:

- **decomposizione orizzontale** di Persona in PersonaNonStudente e PersonaStudente
- **decomposizione orizzontale** di Residenza in ResidenzaNonStudente e ResidenzaStudente

PersonaNonStudente(CFis, DataN, Livello, Occupazione, Patrimonio)

foreign key: PersonaNonStudente[CFis]  $\subseteq$  ResidenzaNonStudente[Persona]

PersonaStudente(CFis, DataN, Livello, Occupazione, Patrimonio)

foreign key: PersonaStudente[CFis]  $\subseteq$  ISA-S-P[Persona]

foreign key: PersonaStudente[CFis]  $\subseteq$  ResidenzaStudente[Studente]

Studente(Matricola, Corso, Anno)

foreign key: Studente[Matricola]  $\subseteq$  ISA-S-P[Studente]

Città(Nome, Nazione)

ISA-S-P(Studente, Persona)

foreign key: ISA-S-P[Studente]  $\subseteq$  Studente[Matricola]

foreign key: ISA-S-P[Persona]  $\subseteq$  PersonaStudente[CFis]

chiave: Persona

ResidenzaNonStudente(Persona, Indirizzo, Città)

foreign key: ResidenzaNonStudente[Persona]  $\subseteq$  PersonaNonStudente[CFis]

foreign key: ResidenzaNonStudente[Città]  $\subseteq$  Città[Nome]

ResidenzaStudente(Studente, Indirizzo, Città)

foreign key: ResidenzaStudente[Studente]  $\subseteq$  PersonaStudente[CFis]

foreign key: ResidenzaStudente[Città]  $\subseteq$  Città[Nome]

Vincolo: PersonaNonStudente[CFis]  $\cap$  PersonaStudente[Cfis] =  $\emptyset$



Per le persone non studenti, ai dati anagrafici (DataN e Residenza) si accede spesso separatamente rispetto ai dati fiscali (Livello, Occupazione e Patrimonio):

- **decomposizione verticale** di PersonaNonStudente in PersonaDatiAnagrafe e PersonaDatiFisco

PersonaDatiAnagrafe(CFis,DataN)

foreign key: PersonaDatiAnagrafe[CFis]  $\subseteq$  PersonaDatiFisco[CFis]

foreign key: PersonaDatiAnagrafe[CFis]  $\subseteq$  ResidenzaNonStudente[Persona]

PersonaDatiFisco(CFis,Livello,Occupazione,Patrimonio)

foreign key: PersonaDatiFisco[CFis]  $\subseteq$  PersonaDatiAnagrafe[CFis]

PersonaStudente(CFis,DataN,Livello,Occupazione,Patrimonio)

foreign key: PersonaStudente[CFis]  $\subseteq$  ISA-S-P[Persona]

foreign key: PersonaStudente[CFis]  $\subseteq$  ResidenzaStudente[Studente]

Studente(Matricola,Corso,Anno)

foreign key: Studente[Matricola]  $\subseteq$  ISA-S-P[Studente]

Città(Nome,Nazione)

ISA-S-P(Studente,Persona)

foreign key: ISA-S-P[Studente]  $\subseteq$  Studente[Matricola]

foreign key: ISA-S-P[Persona]  $\subseteq$  PersonaStudente[CFis]    chiave: Persona

ResidenzaNonStudente(Persona,Indirizzo,Città)

foreign key: Residenza[Persona]  $\subseteq$  PersonaDatiAnagrafe[CFis]

foreign key: Residenza[Città]  $\subseteq$  Città[Nome]

ResidenzaStudente(Studente,Indirizzo,Città)

foreign key: ResidenzaStudente[Studente]  $\subseteq$  PersonaStudente[CFis]

foreign key: Residenza[Città]  $\subseteq$  Città[Nome]

Vincolo: PersonaDatiAnagrafe[CFis]  $\cap$  PersonaStudente[CFis] =  $\emptyset$



Per le persone non studenti, ai dati anagrafici (DataN e Residenza) si accede spesso separatamente rispetto ai dati fiscali (Livello, Occupazione e Patrimonio):

- **accorpamento** tra PersonaDatiAnagrafe e ResidenzaNonStudente

PersonaDatiAnagrafe(CFis,DataN,Indirizzo,Città)

foreign key: PersonaDatiAnagrafe[CFis]  $\subseteq$  PersonaDatiFisco[CFis]

foreign key: PersonaDatiAnagrafe[Città]  $\subseteq$  Città[Nome]

PersonaDatiFisco(CFis,Livello,Occupazione,Patrimonio)

foreign key: PersonaDatiFisco[CFis]  $\subseteq$  PersonaDatiAnagrafe[CFis]

PersonaStudente(CFis,DataN,Livello,Occupazione,Patrimonio)

foreign key: PersonaStudente[CFis]  $\subseteq$  ISA-S-P[Persona]

foreign key: PersonaStudente[CFis]  $\subseteq$  ResidenzaStudente[Studente]

Studente(Matricola,Corso,Anno)

foreign key: Studente[Matricola]  $\subseteq$  ISA-S-P[Studente]

Città(Nome,Nazione)

ISA-S-P(Studente,Persona)

foreign key: ISA-S-P[Studente]  $\subseteq$  Studente[Matricola]

foreign key: ISA-S-P[Persona]  $\subseteq$  PersonaStudente[CFis] chiave: Persona

ResidenzaStudente(Studente,Indirizzo,Città)

foreign key: ResidenzaStudente[Studente]  $\subseteq$  PersonaStudente[CFis]

foreign key: Residenza[Città]  $\subseteq$  Città[Nome]

Vincolo: PersonaDatiAnagrafe[CFis]  $\cap$  PersonaStudente[CFis] =  $\emptyset$





Agli studenti si accede prevalentemente per matricola, e si accede spesso ai dati universitari (Corso, Anno) insieme alla data di nascita e ai dati fiscali (Livello, Occupazione e Patrimonio):

- **accorpamento** tra Studente e ISA-S-P, e poi tra la relazione così ottenuta e PersonaStudente

PersonaDatiAnagrafe(CFis,DataN,Indirizzo,Città)

foreign key: PersonaDatiAnagrafe[CFis]  $\subseteq$  PersonaDatiFisco[CFis]

foreign key: PersonaDatiAnagrafe[Città]  $\subseteq$  Città[Nome]

PersonaDatiFisco(CFis,Livello,Occupazione,Patrimonio)

foreign key: PersonaDatiFisco[CFis]  $\subseteq$  PersonaDatiAnagrafe[CFis]

Studente(Matricola,Corso,Anno,CFis,DataN,Livello,Occupazione,Patrimonio)

foreign key: Studente[CFis]  $\subseteq$  ResidenzaStudente[Studente]

chiave: CFis

Città(Nome,Nazione)

ResidenzaStudente(Studente,Indirizzo,Città)

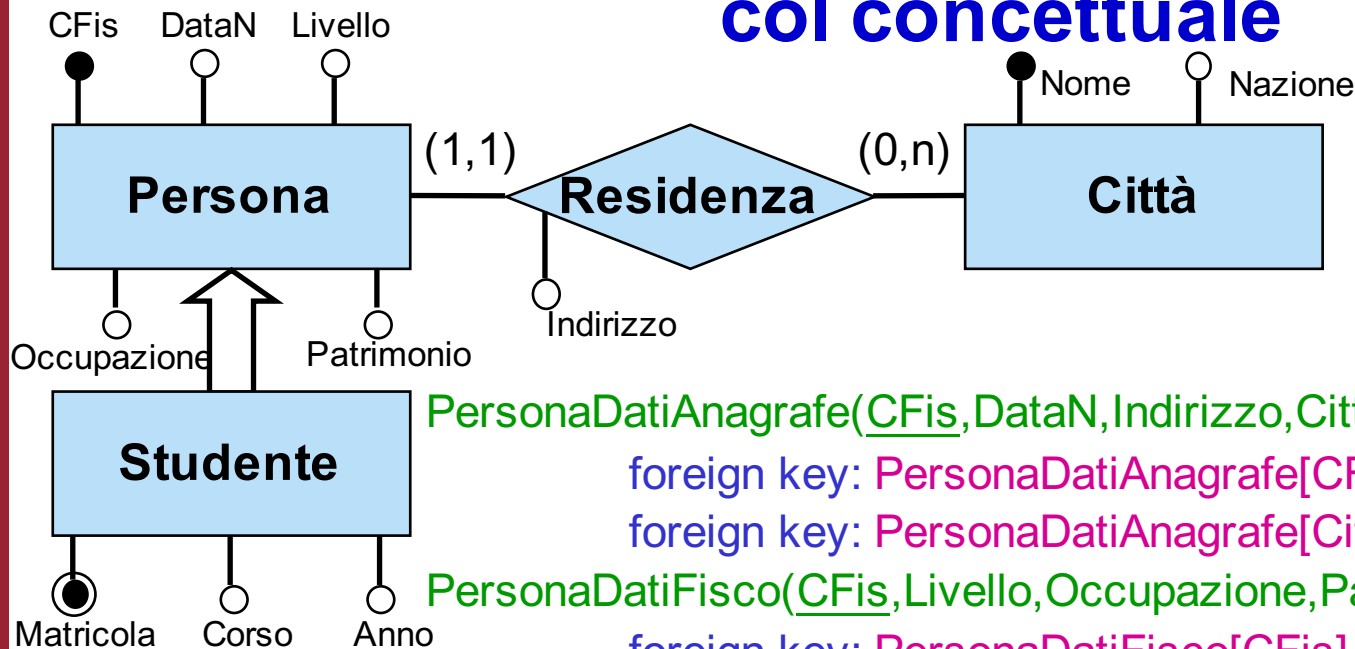
foreign key: ResidenzaStudente[Studente]  $\subseteq$  Studente[CFis]

foreign key: Residenza[Città]  $\subseteq$  Città[Nome]

Vincolo: PersonaDatiAnagrafe[CFis]  $\cap$  Studente[CFis] =  $\emptyset$



## Esercizio 7: schema logico e confronto col concettuale



**PersonaDatiAnagrafe**(CFis, DataN, Indirizzo, Città)

foreign key: **PersonaDatiAnagrafe**[CFis]  $\subseteq$  **PersonaDatiFisco**[CFis]

foreign key: **PersonaDatiAnagrafe**[Città]  $\subseteq$  **Città**[Nome]

**PersonaDatiFisco**(CFis, Livello, Occupazione, Patrimonio)

foreign key: **PersonaDatiFisco**[CFis]  $\subseteq$  **PersonaDatiAnagrafe**[CFis]

**Studente**(Matricola, Corso, Anno, CFis, DataN, Livello, Occupazione, Patrimonio)

foreign key: **Studente**[CFis]  $\subseteq$  **ResidenzaStudente**[Studente]

chiave: **Cfis**

**ResidenzaStudente**(Studente, Indirizzo, Città)

foreign key: **ResidenzaStudente**[Studente]  $\subseteq$  **Studente**[CFis]

foreign key: **Residenza**[Città]  $\subseteq$  **Città**[Nome]

Vincolo: **PersonaDatiAnagrafe**[CFis]  $\cap$  **Studente**[Cfis] =  $\emptyset$

**Città**(Nome, Nazione)

Si lascia come esercizio la definizione delle viste per ricostruire le relazioni dello schema logico originario



## Esercizio 8: ristrutturare il seguente schema ...

Volo(Codice, Comp, Durata)

foreign key: Volo[Comp]  $\subseteq$  Compagnia[Nome]

foreign key: Volo[Codice, Comp]  $\subseteq$  ArrPart[Codice, Comp]

ArrPart(Codice, Comp, Arrivo, Partenza)

foreign key: ArrPart[Arrivo]  $\subseteq$  Aeroporto[Codice]

foreign key: ArrPart[Partenza]  $\subseteq$  Aeroporto[Codice]

foreign key: ArrPart[Codice, Comp]  $\subseteq$  Volo[Codice, Comp]

chiave: Comp, Arrivo, Partenza

VoloCharter(Codice, Comp, TipoAereo)

foreign key: VoloCharter[Codice, Comp]  $\subseteq$  Volo[Codice, Comp]

Aeroporto(Codice, Nome)

foreign key: Aeroporto[Codice]  $\subseteq$  LuogoAeroporto[Aeroporto]

LuogoAeroporto(Aeroporto, NomeCittà, NazCittà)

foreign key: LuogoAeroporto[Aeroporto]  $\subseteq$  Aeroporto[Codice]

foreign key: LuogoAeroporto[NomeCittà, NazCittà]  $\subseteq$   
Città[Nome, Nazione]

Città(Nome, Nazione, NumAbitanti)

Compagnia(Nome, AnnoFond)

foreign key: Compagnia[Nome]  $\subseteq$  Sede[Comp]



## Esercizio 8: ristrutturare il seguente schema...(2)

### Sede(Comp)

foreign key: Sede[Comp]  $\subseteq$  Compagnia[Nome]

foreign key: Sede[Comp]  $\subseteq$  CittàSede[Comp]

inclusione: Sede[Comp]  $\subseteq$  TelefonoComp[Comp]

### CittàSede(Comp, NomeCittà, NazCittà)

foreign key: CittàSede[Comp]  $\subseteq$  Sede[Comp]

foreign key: CittàSede[NomeCittà, NazCittà]  $\subseteq$  Città[Nome, Nazione]

### TelefonoComp(Numero, Comp)

foreign key: Telefono[Comp]  $\subseteq$  Sede[Comp]

foreign key: TelefonoComp[Numero]  $\subseteq$  Telefono[Numero]

### Telefono(Numero)

foreign key: Telefono[Numero]  $\subseteq$  TelefonoComp[Numero]

### Tappa(CodVoloCharter, Comp, Aeroporto, Ordine)

foreign key: Tappa[CodVoloCharter, Comp]  $\subseteq$   
VoloCharter[Codice, Comp]

foreign key: Tappa[Aeroporto]  $\subseteq$  Aeroporto[Codice]

**Vincolo esterno:** vincolo su  
Ordine in Tappa



## ... tenendo conto delle seguenti specifiche

- Si accede spesso all'insieme dei voli charter separatamente dagli altri voli, sia per quanto riguarda la durata, sia per quanto riguarda gli aeroporti di arrivo e partenza
- Quando si accede ad un volo (charter o no) si vuole spesso conoscere tutte le proprietà di tale volo (durata, tipo aereo se volo charter, aeroporto di arrivo e aeroporto di partenza)
- Quando si accede ad una compagnia aerea si accede anche ai dati relativi alla sua sede



## Esercizio 8: soluzione – ristrutturazioni

- Si accede spesso all'insieme dei voli charter separatamente dagli altri voli, sia per quanto riguarda la durata, sia per quanto riguarda gli aeroporti di arrivo e partenza:
  - **decomposizione orizzontale** di Volo in
    - VoloNonCharter
    - DatiVoloCharter
  - **decomposizione orizzontale** di ArrPart in
    - ArrPartVoloNonCharter
    - ArrPartVoloCharter
- Quando si accede ad un volo (charter o no) si vuole spesso conoscere tutte le proprietà di tale volo (durata, tipo aereo se volo charter, aeroporto di arrivo e aeroporto di partenza):
  - **accorpamento** di DatiVoloCharter e VoloCharter e, successivamente, della relazione risultante con ArrPartVoloCharter
  - **accorpamento** di VoloNonCharter e ArrPartVoloNonCharter
- Quando si accede alla compagnia si accede anche ai dati relativi alla sua sede:
  - **accorpamento** di Compagnia e CittàSede
- Si eliminano le relazioni inutili Sede e Telefono mediante **accorpamento**



## Esercizio 8: soluzione – schema ristrutturato

**VoloNonCharter**(Codice, Comp, Durata, Arrivo, Partenza)

foreign key: **VoloNonCharter**[Comp]  $\subseteq$  **Compagnia**[Nome]

foreign key: **VoloNonCharter**[Arrivo]  $\subseteq$  **Aeroporto**[Codice]

foreign key: **VoloNonCharter**[Partenza]  $\subseteq$  **Aeroporto**[Codice]

chiave: Comp, Arrivo, Partenza

**VoloCharter**(Codice, Comp, TipoAereo, Durata, Arrivo, Partenza)

foreign key: **VoloCharter**[Comp]  $\subseteq$  **Compagnia**[Nome]

foreign key: **VoloCharter**[Arrivo]  $\subseteq$  **Aeroporto**[Codice]

foreign key: **VoloCharter**[Partenza]  $\subseteq$  **Aeroporto**[Codice]

chiave: Comp, Arrivo, Partenza

**Aeroporto**(Codice, Nome)

foreign key: **Aeroporto**[Codice]  $\subseteq$  **LuogoAeroporto**[Aeroporto]

**LuogoAeroporto**(Aeroporto, NomeCittà, NazCittà)

foreign key: **LuogoAeroporto**[Aeroporto]  $\subseteq$  **Aeroporto**[Codice]

foreign key: **LuogoAeroporto**[NomeCittà, NazCittà]  $\subseteq$  **Città**[Nome, Nazione]

**Città**(Nome, Nazione, NumAbitanti)

**Compagnia**(Nome, AnnoFond, NomeCittà, NazCittà)

foreign key: **Compagnia**[NomeCittà, NazCittà]  $\subseteq$  **Città**[Nome, Nazione]

inclusione: **Compagnia**[Nome]  $\subseteq$  **TelefonoComp**[Comp]

**TelefonoCom**(Numero, Comp)

foreign key: **TelefonoComp**[Comp]  $\subseteq$  **Compagnia**[Nome]

**Tappa**(CodVoloCharter, Comp, Aeroporto, Ordine)

foreign key: **Tappa**[CodVoloCharter, Comp]  $\subseteq$  **VoloCharter**[Codice, Comp]

foreign key: **Tappa**[Aeroporto]  $\subseteq$  **Aeroporto**[Codice]



## Esercizio 8: soluzione – vincoli e viste

### Vincoli:

- VoloNonCharter e VoloCharter sono disgiunti:  
 $\text{VoloNonCharter}[\text{Codice}, \text{Comp}] \cap \text{VoloCharter}[\text{Codice}, \text{Comp}] = \emptyset$   
 $\text{VoloNonCharter}[\text{Comp}, \text{Arrivo}, \text{Partenza}] \cap \text{VoloCharter}[\text{Comp}, \text{Arrivo}, \text{Partenza}] = \emptyset$
- vincolo su Ordine in Tappa

### Viste per ricostruire le relazioni dello schema originario:

view Volo =  $\text{PROJ}_{\text{Codice}, \text{Comp}, \text{Durata}}(\text{VoloNonCharter}) \cup \text{PROJ}_{\text{Codice}, \text{Comp}, \text{Durata}}(\text{VoloCharter})$

view ArrPart =  $\text{PROJ}_{\text{Codice}, \text{Comp}, \text{Arrivo}, \text{Partenza}}(\text{VoloNonCharter}) \cup \text{PROJ}_{\text{Codice}, \text{Comp}, \text{Arrivo}, \text{Partenza}}(\text{VoloCharter})$

view Sede =  $\text{PROJ}_{\text{Nome}}(\text{Compagnia})$

view CompagniaOriginaria =  $\text{PROJ}_{\text{Nome}, \text{AnnoFond}}(\text{Compagnia})$

view CittàSede =  $\text{PROJ}_{\text{Nome}, \text{NomeCittà}, \text{NazCittà}}(\text{Compagnia})$

view Telefono =  $\text{PROJ}_{\text{Numero}}(\text{TelefonoComp})$





## Esercizio 9: ristrutturare il seguente schema ...

Officina(Nome, NumDip, Indirizzo)

foreign key: Officina[Nome]  $\subseteq$  Dirige[Officina]

inclusione: Officina[Nome]  $\subseteq$  Lavora[Officina]

Persona(CodFis, Indirizzo)

Direttore(CodFis, Età, AnniAnz)

foreign key: Direttore[CodFis]  $\subseteq$  Persona[CodFis]

foreign key: Direttore[CodFis]  $\subseteq$  Dirige[Direttore]

Dipendente(CodFis, AnniAnz)

foreign key: Dipendente[CodFis]  $\subseteq$  Persona[CodFis]

inclusione: Dipendente[CodFis]  $\subseteq$  Lavora[Dipendente]

Dirige(Officina, Direttore)

foreign key: Dirige[Officina]  $\subseteq$  Officina[Nome]

foreign key: Dirige[Direttore]  $\subseteq$  Direttore[CodFis]

chiave: Direttore

Lavora(Officina, Dipendente, AnniServizio)

foreign key: Lavora[Officina]  $\subseteq$  Officina[Nome]

foreign key: Lavora[Dipendente]  $\subseteq$  Dipendente[CodFis]

TelPer(CodFis, Telefono)

foreign key: TelPer[CodFis]  $\subseteq$  Persona[CodFis]

foreign key: TelPer[Telefono]  $\subseteq$  Telefono[Numero]



## Esercizio 9: ristrutturare il seguente schema...(parte 2)

Telefono(Numero)

inclusione: Telefono[Numero]  $\subseteq$  TelPer[Telefono]

Veicolo(Targa, Modello, Tipo, AnnoImm)

foreign key : Veicolo[Targa]  $\subseteq$  Possiede[Veicolo]

Possiede(Veicolo, Proprietario)

foreign key: Possiede[Veicolo]  $\subseteq$  Veicolo[Targa]

foreign key: Possiede[Proprietario]  $\subseteq$  Persona[CodFis]

Riparazione(Codice, Officina, OraAcc, DataAcc)

inclusione: Riparazione[Officina]  $\subseteq$  Officina[Nome]

foreign key: Riparazione[Codice,Officina]  $\subseteq$  Relativa[Codice,Officina]

Relativa(Codice, Officina, Veicolo)

foreign key: Relativa[Codice,Officina]  $\subseteq$  Riparazione[Codice,Officina]

foreign key: Relativa[Veicolo]  $\subseteq$  Veicolo[Targa]

Terminata(Codice, Officina, OraRic, DataRic)

foreign key: Terminata[Codice,Officina]  $\subseteq$  Riparazione[Codice,Officina]

### Vincoli esterni:

- riconsegna dopo accettazione
- vincolo che lega Officina[NumDip] alle istanze in Lavora
- vincolo su AnniAnz di Direttore e Dipendente derivante dall'eliminazione ISA



## ... tenendo conto delle seguenti specifiche

- Quando si accede ai direttori, interessano anche tutti i dati relativi all'officina che dirigono e viceversa, quando si accede alle officine, interessano anche i dati relativi all'età e alla anzianità del loro direttore
- Alle persone dipendenti si accede spesso separatamente rispetto alle persone direttori
- Quando si accede ai dipendenti interessano anche i dati relativi all'indirizzo



## Esercizio 9: soluzione – ristrutturazioni

- Quando si accede ai direttori, interessano anche tutti i dati relativi all'officina che dirigono e viceversa quando si accede alle officine, interessano anche tutti i dati relativi al loro direttore:
  - **accorpamento** di Direttore e Dirige e, successivamente, della relazione risultante con Officina
- Ai dipendenti si accede spesso separatamente rispetto ai direttori:
  - **decomposizione orizzontale** di Persona in dipendenti e non
- Quando si accede ai dipendenti interessano anche i loro dati anagrafici:
  - **accorpamento** delle relazioni Dipendente e PersonaDipendente
- Si elimina la relazione inutile Telefono mediante **accorpamento**



## Esercizio 9: soluzione – schema ristrutturato

OfficinaDirettore(Nome, NumDip, Indirizzo, Direttore, EtaDir, AnniAnzDir)

chiave: Direttore

inclusione: Officina[Nome]  $\subseteq$  Lavora[Officina]

PersonaNonDip(CodFis, Indirizzo)

Dipendente(CodFis, AnniAnz, Indirizzo)

inclusione: Dipendente[CodFis]  $\subseteq$  Lavora[Dipendente]

Lavora(Officina, Dipendente, AnniServizio)

foreign key: Lavora[Officina]  $\subseteq$  OfficinaDirettore[Nome]

foreign key: Lavora[Dipendente]  $\subseteq$  Dipendente[CodFis]

TelPer(CodFis, Telefono)

Veicolo(Targa, Modello, Tipo, AnnoImm)

Possiede(Veicolo, Proprietario)

foreign key: Possiede[Veicolo]  $\subseteq$  Veicolo[Targa]

Riparazione(Codice, Officina, OraAcc, DataAcc,)

inclusione: Riparazione[Officina]  $\subseteq$  OfficinaDirettore[Nome]

foreign key: Riparazione[Codice, Officina]  $\subseteq$  Relativa[Codice, Officina]

Relativa(Codice, Officina, Veicolo)

foreign key: Relativa[Codice, Officina]  $\subseteq$  Riparazione[Codice, Officina]

foreign key: Relativa[Veicolo]  $\subseteq$  Veicolo[Targa]

Terminata(Codice, Officina, OraRic, DataRic)

foreign key: Terminata[Codice, Officina]  $\subseteq$  Riparazione[Codice, Officina]



## Esercizio 9: soluzione – vincoli e viste

### Vincoli:

- PersonaNonDip e Dipendente sono disgiunti:
  - $\text{PersonaNonDip}[\text{CodFis}] \cap \text{Dipendente}[\text{CodFis}] = \emptyset$
- Vincoli risultanti dal vincolo di foreign key verso Persona
  - $\text{Officina}[\text{Direttore}] \subseteq \text{PersonaNonDip}[\text{CodFis}] \cup \text{Dipendente}[\text{CodFis}]$
  - $\text{Veicolo}[\text{Proprietario}] \subseteq \text{PersonaNonDip}[\text{CodFis}] \cup \text{Dipendente}[\text{CodFis}]$
  - $\text{TelPer}[\text{CodFis}] \subseteq \text{PersonaNonDip}[\text{CodFis}] \cup \text{Dipendente}[\text{CodFis}]$
- Vincoli esterni:
  - riconsegna dopo accettazione
  - vincolo che lega Officina[NumDip] alle istanze in Lavora
  - vincolo su Officina[AnniAnzDir] e Dipendente[AnniAnz] derivante dall'eliminazione ISA

**Viste** per ricostruire le relazioni dello schema originario:

view Persona = PersonaNonDip  $\cup$  PROJ<sub>CodFis, Indirizzo</sub>(Dipendente)

view OfficinaOrig = PROJ<sub>Nome, NumDip, Indirizzo</sub>(Officina)

view Direttore = PROJ<sub>Direttore, EtaDir, AnniAnzDir</sub>(Officina)

view Dirige = PROJ<sub>Nome, Direttore</sub>(Officina)

view Telefono = PROJ<sub>Telefono</sub>(TelPer)