

Regole per vincoli di integrità

Possono essere espresse sotto forma di **asserzioni**, cioè di affermazioni che devono sempre essere verificate nella base di dati, e che siano:

- **atomiche**: per chiarezza, non possono essere decomposte in frasi che contengono altre asserzioni
- **dichiarative**: devono enunciare la proprietà non il modo per soddisfarla
 - la reazione ad una possibile violazione è un problema realizzativo, non pertinente alla rappresentazione concettuale

Assertzioni per vincoli di integrità

Possibile struttura:

- *<concetto>* **deve/non deve** *<espressione su concetti>*

I concetti citati possono corrispondere a oggetti citati nello schema o a oggetti derivabili da essi (es., *direttore di dipartimento* derivato dalla relazione *direzione* fra *impiegato* e *dipartimento*)

Esempio

- un dipartimento con sede a Roma **deve** essere diretto da un impiegato con più di dieci anni di afferenza al dipartimento
- un impiegato **non deve** avere uno stipendio maggiore del direttore di dipartimento al quale afferisce

Regole per derivazioni

Possono essere espresse specificando le operazioni che permettono di ottenere il concetto derivato.

Possibile struttura:

- *<concetto> si ottiene <operazione su concetti>*

I concetti citati possono corrispondere a oggetti citati nello schema o a oggetti derivabili da essi (es., *direttore di dipartimento*)

Esempio

- il budget di un progetto **si ottiene** moltiplicando per tre la somma degli stipendi degli impiegati che vi partecipano

Implementazione di regole

Nella traduzione dello schema concettuale in una base di dati, le regole aziendali non descrittive (asserzioni per integrità e derivazioni) devono **essere codificate** per garantire la consistenza dei dati rispetto alle proprietà stabilite dalle regole.

Diversi approcci:

- clausole del linguaggio **SQL**
- regole attive (**trigger**)
- opportune **procedure** scritte in qualche linguaggio di programmazione

Documentazione di schemi E-R

Le regole aziendali forniscono documentazione di supporto al sistema E-R.

Deve essere prodotta documentazione per:

- descrizione di concetti
- regole di vincolo
- regole di derivazione

Descrizione di concetti

Può essere prodotta facendo uso di un dizionario dei dati, composto da due tabelle:

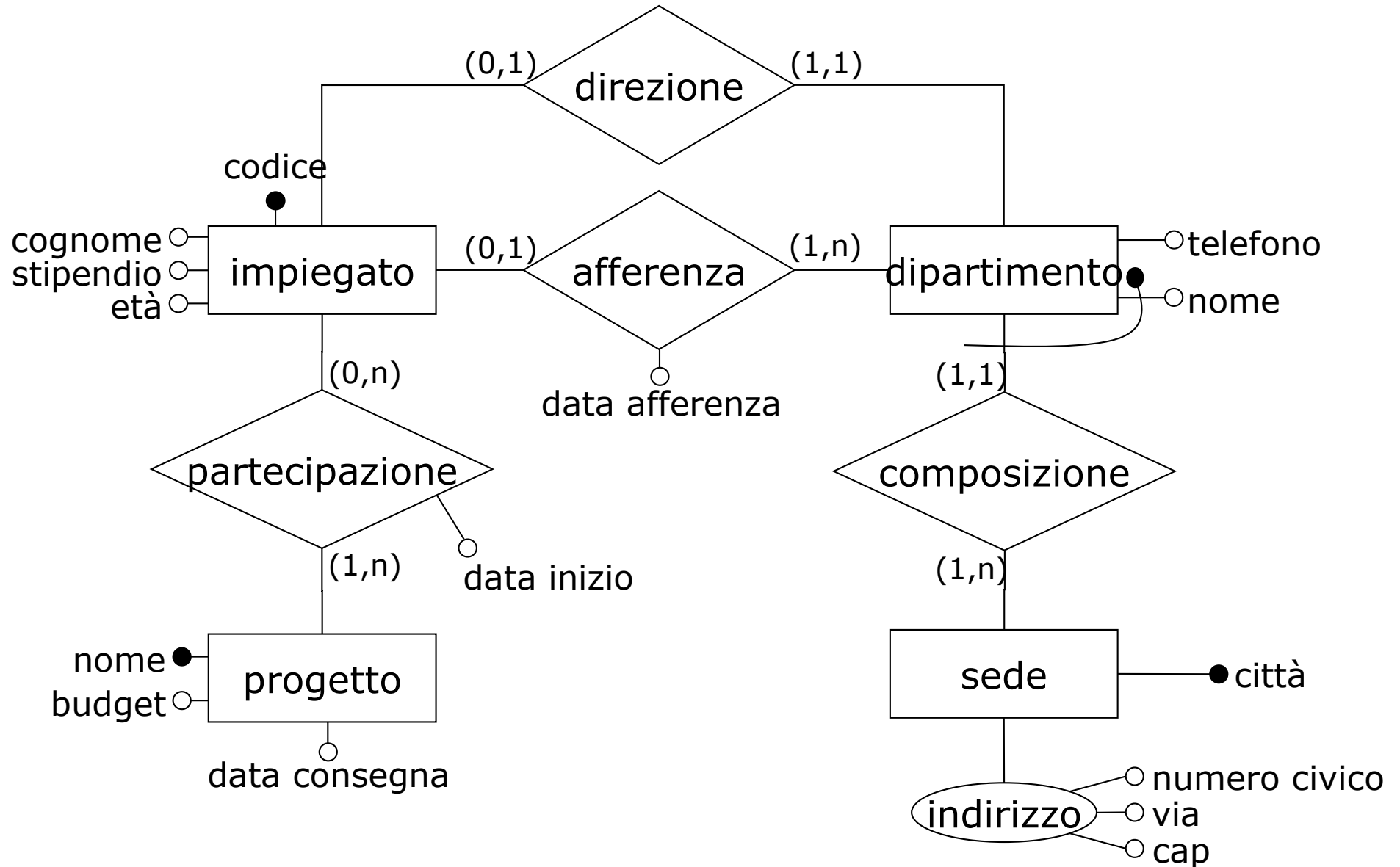
- **descrizione delle entità**, con attributi:
 - nome,
 - definizione informale,
 - elenco degli attributi (con eventuale descrizione)
 - identificatore
- **descrizione delle relazioni**, con attributi:
 - nome,
 - definizione informale,
 - elenco degli attributi (con eventuale descrizione),
 - elenco delle entità coinvolte insieme con la loro cardinalità di partecipazione

Documentazione di vincoli e derivazioni

Può essere prodotta facendo uso di due tabelle:

- **descrizione dei vincoli di integrità**: elenca le regole che esprimono vincoli
- **descrizione delle derivazioni**: elenca le regole che esprimono concetti derivati

Documentazione di schemi E-R: esempio (1)



Documentazione di schemi E-R: esempio (2)

Nome Entità	Descrizione	Attributi	Identificatore
impiegato	Impiegato che lavora nella azienda.	codice, cognome, stipendio, età	codice
progetto	Progetti aziendali sui quali lavorano gli impiegati.	nome, budget, data consegna	nome
dipartimento	Dipartimenti delle sedi dell'azienda.	telefono, nome	nome, sede
sede	Sede dell'azienda in una certa città.	città, indirizzo (numero, via, CAP)	città

Documentazione di schemi E-R: esempio (3)

Nome Relazione	Descrizione	Entità Coinvolte	Attributi
direzione	Associa un dipartimento al suo direttore.	impiegato (0,1) dipartimento (1,1)	
afferenza	Associa un impiegato al suo dipartimento.	impiegato (0,1) dipartimento (1,n)	data afferenza
partecipazione	Associa agli impiegati i progetti sui quali lavorano.	impiegato (0,1) progetto (1,n)	data inizio
composizione	Associa una sede ai dipartimenti di cui è composta.	dipartimento (1,1) sede (1,n)	

Documentazione di schemi E-R: esempio (4)

Regole di Vincolo

- (RV1) Il direttore di un dipartimento deve afferire a tale dipartimento.
- (RV2) Un impiegato non deve avere uno stipendio maggiore del direttore del dipartimento al quale afferisce.
- (RV3) Un dipartimento con sede a Roma deve essere diretto da un impiegato con più di dieci anni di anzianità.
- (RV4) Un impiegato che non afferisce a alcun dipartimento non deve partecipare a alcun progetto.

Regole di Derivazione

- (RD1) Il budget di un progetto si ottiene moltiplicando per 3 la somma degli stipendi degli impiegati che vi partecipano.

Strategie di progetto

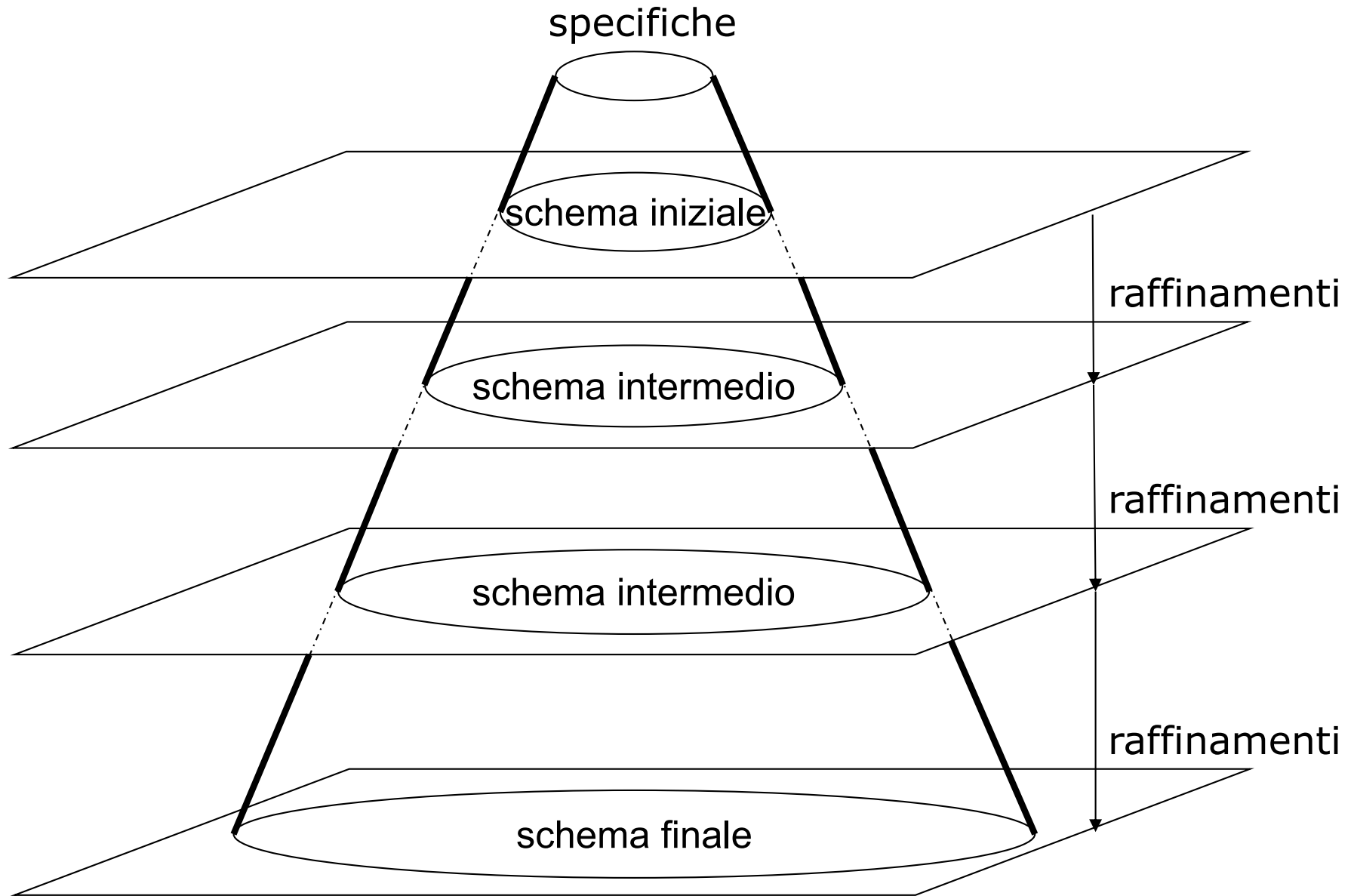
Lo sviluppo di uno schema concettuale a partire dalle sue specifiche può utilizzare diverse strategie (come tutti i progetti di ingegnerizzazione):

- **Top-down**
- Bottom-up
- Inside-out
- Mista

Top-down

- A partire dalle specifiche si produce uno schema iniziale con pochi concetti molto astratti.
- Lo schema viene raffinato aumentando il dettaglio dei vari concetti:
 - i raffinamenti prevedono l'uso di **trasformazioni elementari** (**primitive**) che operano sul singolo concetto per descriverlo con maggior dettaglio.
- Per raffinamenti successivi si arriva allo schema finale.

Strategia top-down



Primitive di trasformazione top-down

- **T1:** da entità a relazione tra entità
- **T2:** da entità a generalizzazione
- **T3:** da relazione a insieme di relazioni
- **T4:** da relazione a entità con relazioni
- **T5:** introduzione di attributi su entità
- **T6:** introduzione di attributi su relazioni

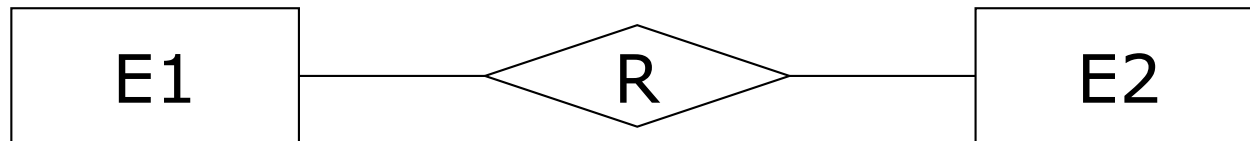
Da entità a relazione tra entità

Una entità viene sostituita da due entità collegate tra loro da una relazione.

Da:



A:

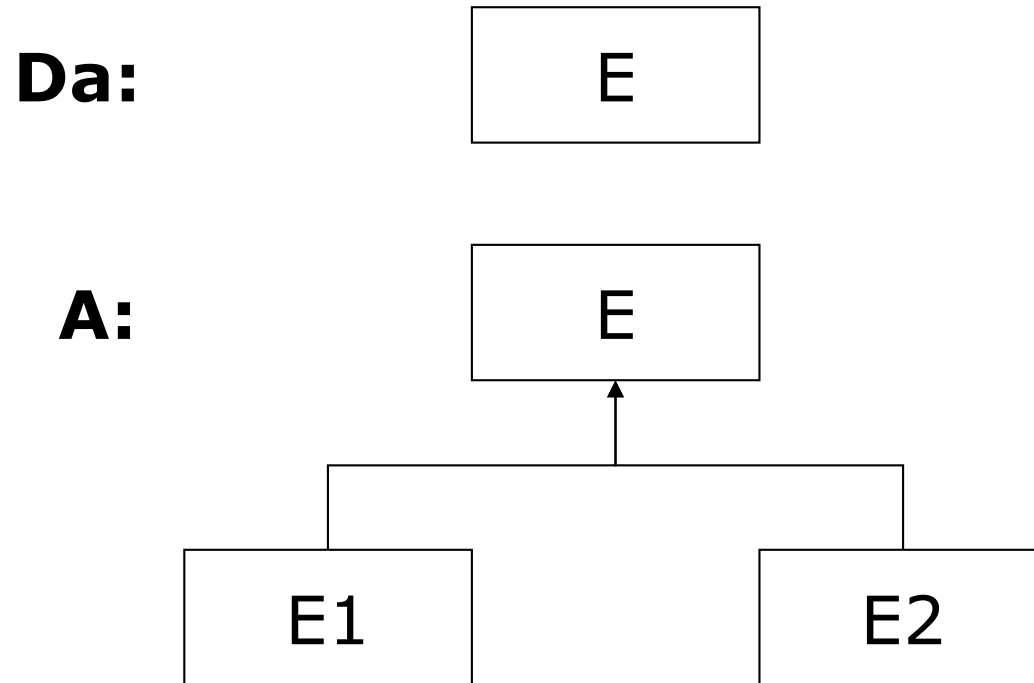


Esempio

Entità *corso* può essere sostituita da due entità: *tipo-corso* (che descrive il corso) e *edizione-corso* (relativa alla erogazione in uno specifico A.A.), collegate dalla relazione *tipologia*

Da entità a generalizzazione

Una entità è scomposta in sotto-entità distinte.



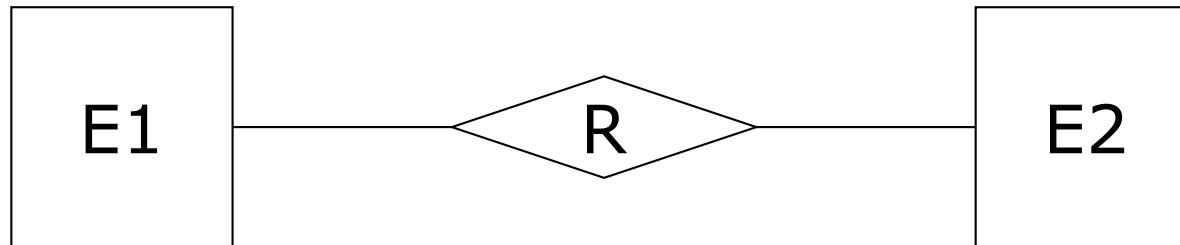
Esempio

Entità *impiegato* può essere scomposta in:
full-time e *part-time*.

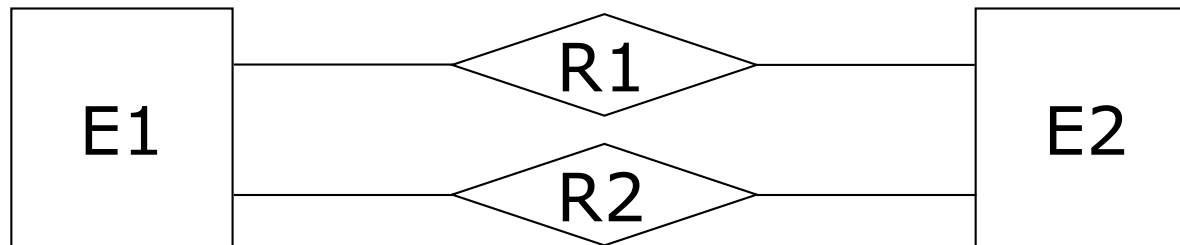
Da relazione a insiemi di relazioni

Una relazione viene sostituita da due relazioni.

Da:



A:

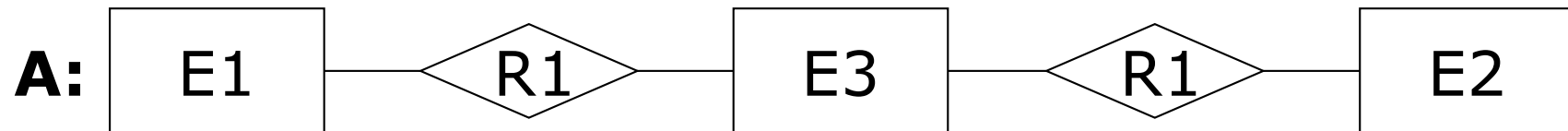
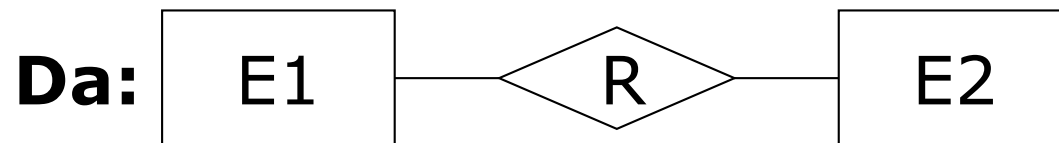


Esempio

Relazione *docenza* fra *docente* e *corso* può essere sostituita dalle relazioni: *docenza-corrente* (nell' A.A. attuale) e *docenza-passata*.

Da relazione a entità con relazioni

Una relazione viene sostituita con una entità (costituisce oggetto con esistenza autonoma)



Esempio

Relazione *contratto* fra entità *consulente* e *azienda* può essere sostituita da una entità *contratto* in relazione con *consulente* e con *azienda*.

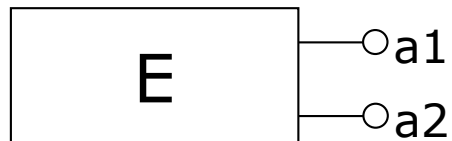
Introduzione di attributi su entità

Vengono aggiunti attributi ad una entità

Da:



A:



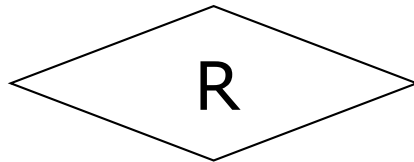
Esempio

Entità *studente*, possono essere aggiunti attributi *matricola, nome, cognome*.

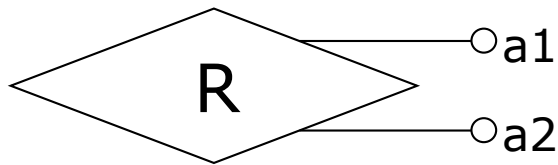
Introduzione di attributi su relazioni

Vengono aggiunti attributi ad una relazione

Da:



A:



Esempio

Relazione *esame*, possono essere aggiunti attributi *data*, *voto*.

Top-down: vantaggi/svantaggi

Tutti gli aspetti presenti nello schema finale sono presenti, in linea di principio, a ogni livello di raffinamento.

Vantaggi:

- il progettista può descrivere inizialmente tutte le specifiche dei dati, trascurando i dettagli, per poi entrare nel merito di un concetto alla volta

Svantaggi:

- è possibile solo quando si possiede sin dall'inizio una visione globale e astratta di tutte le componenti del sistema, generalmente non è così per applicazioni complesse

Strategie di progetto

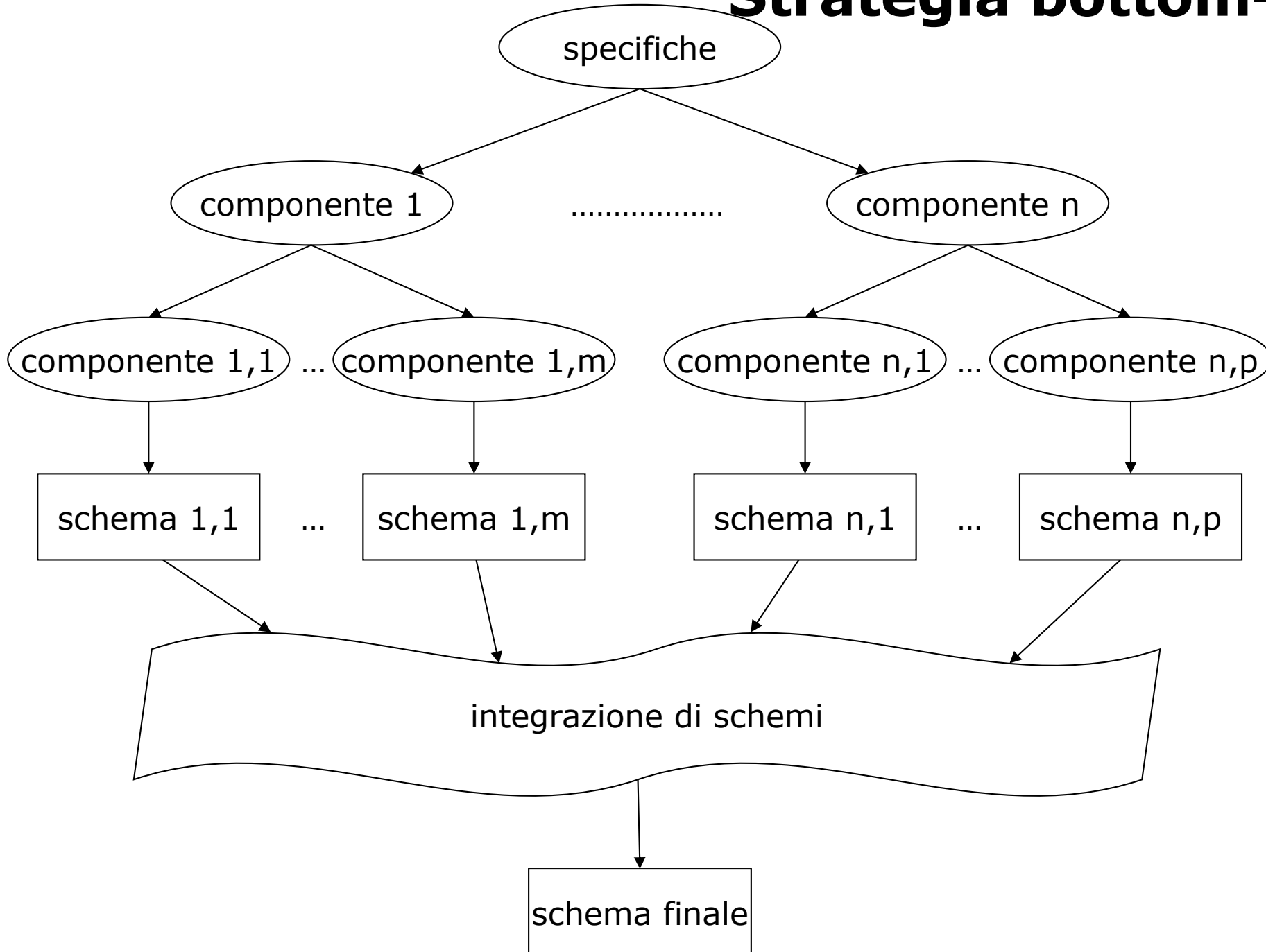
Lo sviluppo di uno schema concettuale a partire dalle sue specifiche può utilizzare diverse strategie (come tutti i progetti di ingegnerizzazione):

- Top-down
- **Bottom-up**
- **Inside-out**
- **Mista**

Bottom-up

- Le specifiche iniziali sono suddivise in sotto-componenti via via sempre più piccole fino a quando queste descrivono un frammento elementare dell' applicazione
- Le varie componenti vengono rappresentate tramite schemi concettuali:
 - prevede l' uso di **trasformazioni elementari** (**primitive**) che introducono nuovi concetti
- I vari sotto-schemi vengono fusi fino a giungere, attraverso completa integrazione delle componenti, allo schema concettuale finale

Strategia bottom-up



Primitive di trasformazione bottom-up

- **T1:** generazione di entità
- **T2:** generazione di relazione
- **T3:** generazione di generalizzazione
- **T4:** aggregazione di attributi su entità
- **T5:** aggregazione di attributi su relazione

Generazione di entità

Si individua, nelle specifiche, una classe di oggetti con proprietà comuni.

Da:

A:

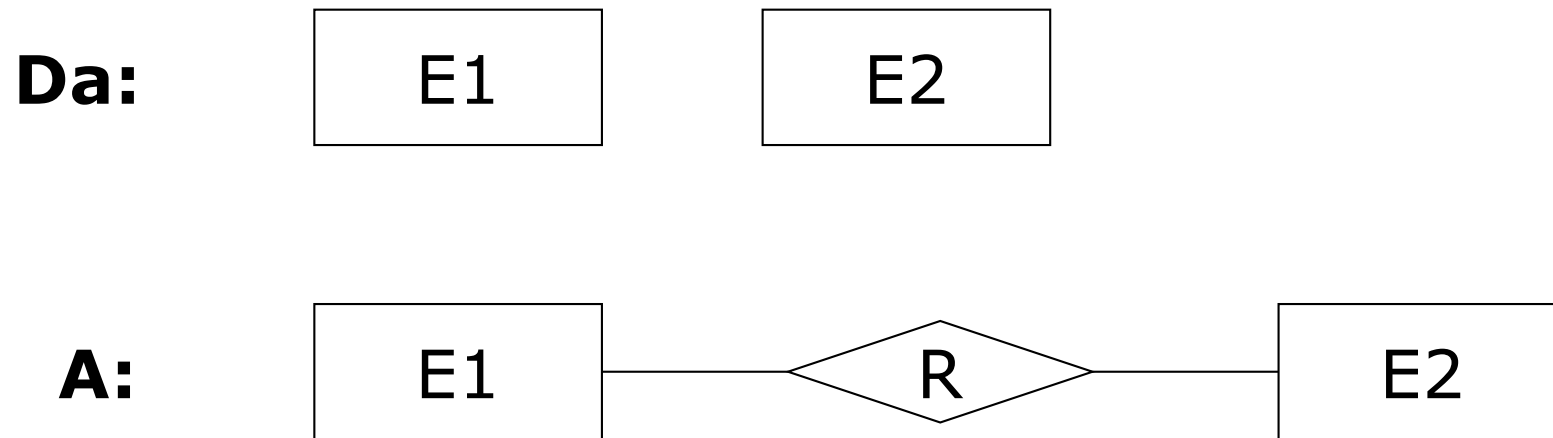


Esempio

Individuiamo nelle specifiche, l'entità *docente*.

Generazione di relazione

Si individua, nelle specifiche, un legame logico tra entità.

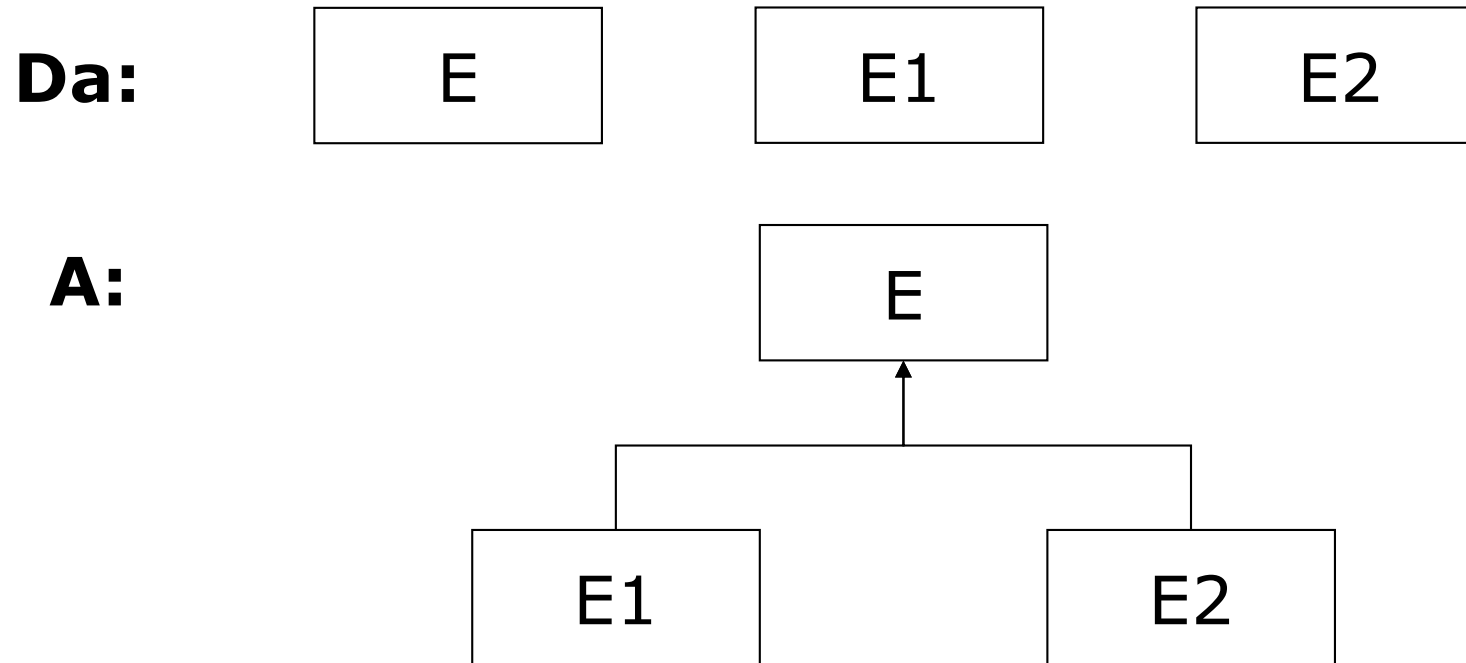


Esempio

Introduciamo la relazione *tenere* fra entità *docente* e *corso*.

Generazione di generalizzazione

Si individua, nelle specifiche, un legame di generalizzazione fra entità



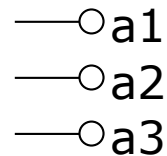
Esempio

Individuiamo che *personale* è una generalizzazione di: *dipendente* e *consulente*.

Aggregazione di attributi su entità

A partire da un insieme di attributi si individua una entità che li aggrega

Da:



A:



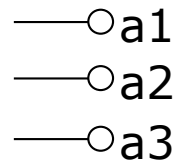
Esempio

Attributi *matricola*, *nome*, *cognome* vengono raccolti come entità *studente*.

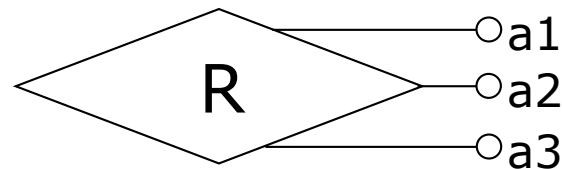
Aggregazione di attributi su relazioni

A partire da un insieme di attributi si individua una relazione che li aggrega

Da:



A:



Esempio

Attributi *voto*, *data* vengono raccolti come relazione *esame*

Bottom-up : vantaggi/svantaggi

Le primitive introducono nello schema nuovi concetti per descrivere aspetti della realtà di interesse non ancora rappresentati

Vantaggi:

- si adatta alla decomposizione del problema in componenti più semplici, il cui progetto può essere affidato a progettisti diversi:
 - sviluppi di applicazioni complesse e in collaborazione

Svantaggi:

- richiede operazioni di integrazione (spesso difficoltose) di schemi concettuali diversi
- i singoli progettisti hanno una visione parziale

Inside-out (1)

Caso particolare della strategia bottom-up

- Si individuano inizialmente alcuni concetti e poi si procede, a partire da questi, a “macchia d’olio”:
 - si rappresentano i concetti più vicini a quelli iniziali
 - si procede attraverso una navigazione delle specifiche

Inside-out (2)

Vantaggi:

- non richiede passi di integrazione (rispetto al bottom-up)

Svantaggi:

- richiede di esaminare di volta in volta tutte le specifiche per individuare i concetti non ancora rappresentati e descriverli nel dettaglio (cosa non sempre possibile)
- non permette di procedere per livelli di astrazione

Combina le strategie top-down e bottom-up

- Si suddividono i requisiti in **componenti separate** (come nel **bottom-up**)
- Si definisce uno **schema scheletro** contenente, a livello astratto, i concetti principali dell'applicazione (come nel **top-down**)

Vantaggi:

- unisce i vantaggi delle strategie top-down e bottom-up:
 - la divisione in sotto-componenti permette di affrontare applicazioni complesse
 - lo schema scheletro fornisce una visione unitaria e favorisce le fasi di integrazione degli schemi sviluppati separatamente

È l'unica effettivamente utilizzabile in tutti i casi pratici di una certa complessità (dove è spesso necessario cominciare la progettazione quando non tutti i dati sono disponibili)

Qualità di uno schema concettuale

Nello sviluppo di uno schema concettuale vanno garantite alcune proprietà generali:

- correttezza
- completezza
- leggibilità
- minimalità

Correttezza

Lo schema concettuale deve **utilizzare propriamente** i costrutti messi a disposizione

- **correttezza sintattica**: i costrutti devono essere utilizzati solo secondo l'uso previsto.
 - Es., non è corretto definire una generalizzazione fra relazioni.
- **correttezza semantica**: l'uso dei costrutti deve rispettare la loro definizione.
 - Es., non è corretto utilizzare una relazione per definire che una entità è generalizzazione di un'altra.

Completezza

Lo schema concettuale deve rappresentare tutti i dati di interesse e permettere di eseguire tutte le operazioni a partire dai dati espressi nello schema:

- tutte le specifiche sui dati devono essere rappresentate (da un qualche concetto) nello schema
- tutti i concetti coinvolti in una operazione presente nelle specifiche devono essere raggiungibili “navigando” attraverso lo schema

Leggibilità

Lo schema concettuale deve rappresentare i requisiti in **maniera naturale e facilmente comprensibile**

- rendere lo schema autoesplicativo scegliendo opportunamente i nomi dei concetti
- mantenere il più possibile semplice e pulito il diagramma
 - disporre i costrutti scegliendo come elementi centrali quelli con più relazioni
 - tracciare solo linee perpendicolari e minimizzare le intersezioni
 - disporre le entità che sono padri di generalizzazioni sopra le loro figlie

Minimalità

Lo schema concettuale dovrebbe rappresentare ogni specifica sui dati **una sola volta (**non dovrebbero esserci ridondanze**).**

- Esistono ridondanze quando un concetto può essere derivato da altri:
 - tipicamente una ridondanza corrisponde ad un ciclo nello schema (da una entità E, percorrendo relazioni e entità, riesco a ritornare ad E) in cui è possibile rimuovere una relazione **senza perdere informazione**.

Non sempre le ridondanze sono indesiderate, a volte possono voler essere mantenute per facilitare l'accesso ai dati (devono però essere individuate).

Strumenti CASE (1)

Esistono diversi pacchetti applicativi per il progetto e lo sviluppo di basi di dati (strumenti CASE) che offrono generalmente:

- **interfaccia grafica** per manipolare schemi E-R
- **dizionario dei dati** che memorizza informazioni sui vari concetti dello schema
- **strumenti integrati** che **eseguono**, in modo automatico o tramite interazione con l'utente, **compiti della progettazione**
 - layout, analisi di qualità, produzione automatica del codice per DBMS

Strumenti CASE (2)

Le funzionalità offerte variano fra i prodotti:

- alcuni sistemi sono direttamente integrabili con i DBMS
- alcuni sistemi forniscono anche supporto all'analisi dei requisiti
- alcuni sistemi mettono a disposizione librerie di progetti

Non esistono standardizzazioni né sulle notazioni né sul modello concettuale di riferimento

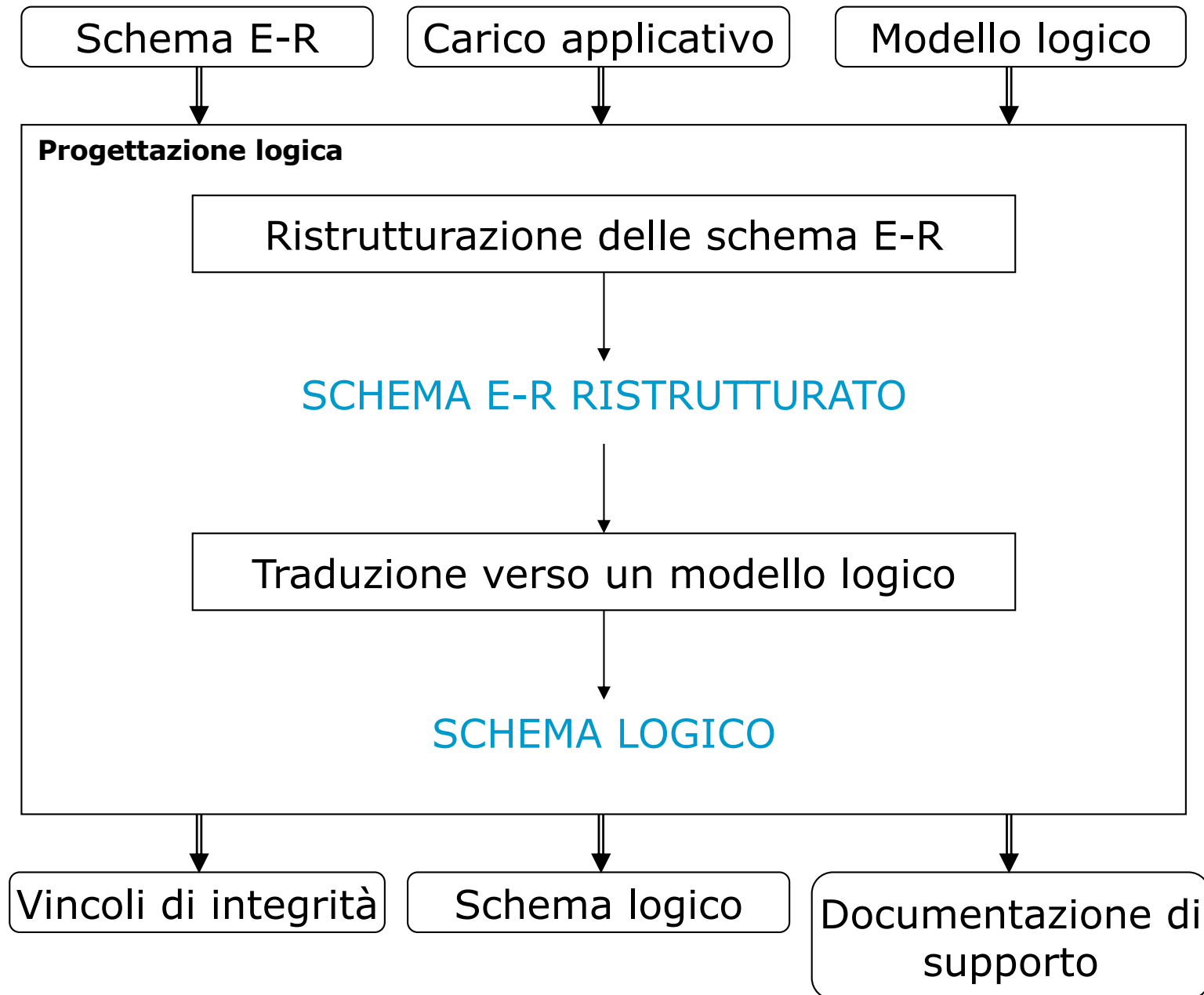
Progettazione logica (1)

Traduce lo schema concettuale in uno schema logico in grado di descrivere in maniera corretta ed efficiente tutte le informazioni prodotte dalla progettazione concettuale (schema E-R + vincoli)

Articolata in due fasi:

- ristrutturazione dello schema E-R
- traduzione verso il modello logico

Progettazione logica (2)



Progettazione logica: fasi

Ristrutturazione dello schema E-R

- ottimizzazione e semplificazione dello schema
 - per rendere più efficienti le operazioni previste
 - per tradurre concetti non rappresentabili nel modello logico
- è indipendente dal modello logico scelto

Traduzione verso il modello logico

- fa riferimento allo specifico modello logico scelto
- può includere una ulteriore ottimizzazione basata sulle caratteristiche del modello logico

Ristrutturazione di schemi E-R

Composta da diverse fasi:

- analisi delle ridondanze
- eliminazione delle gerarchie
- partizionamento/accorpamento di entità e relazioni
- scelta degli identificatori primari

Analisi delle ridondanze

Attributi **derivabili** da:

- altri attributi della stessa entità o relazione
- altri attributi di altre entità o relazioni
- operazioni di conteggio di occorrenze

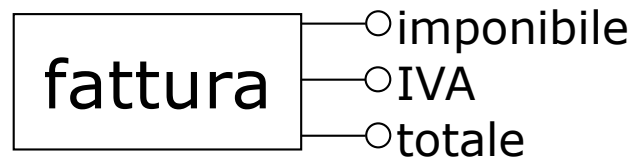
Relazioni **derivabili** da:

- composizione di altre relazioni in presenza di cicli

Ridondanze: esempio (1)

Attributi derivabili da altri della stessa entità o relazione:

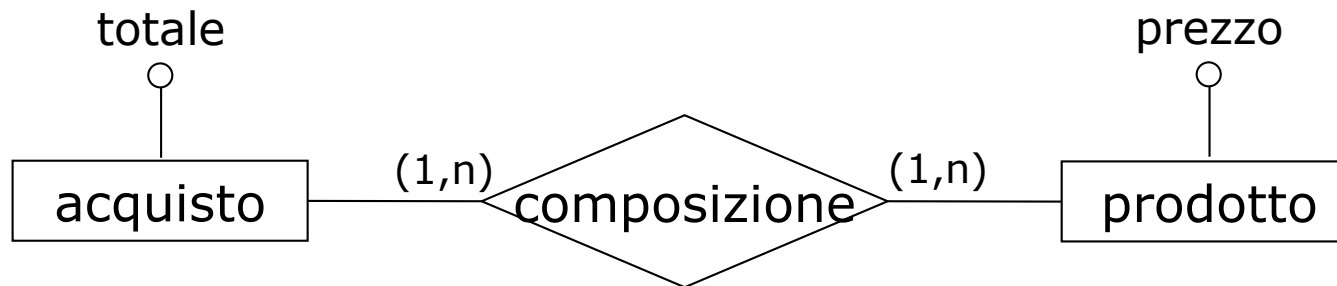
- *totale = imponibile + IVA*



Ridondanze: esempio (2)

Attributi derivabili da attributi di altre entità o relazioni:

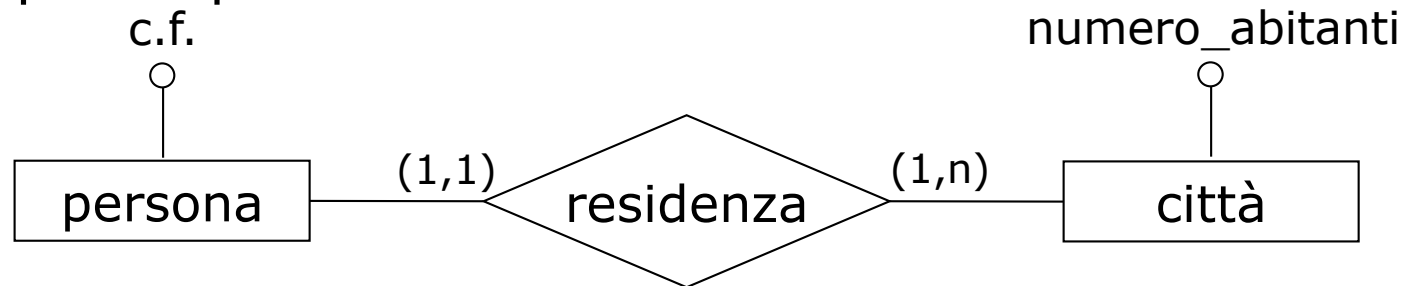
- *totale* di *acquisto* può essere derivato da *prezzo* di *prodotto* (attraverso *composizione*)



Ridondanze: esempio (3)

Attributi derivabili da operazioni di conteggio di occorrenze:

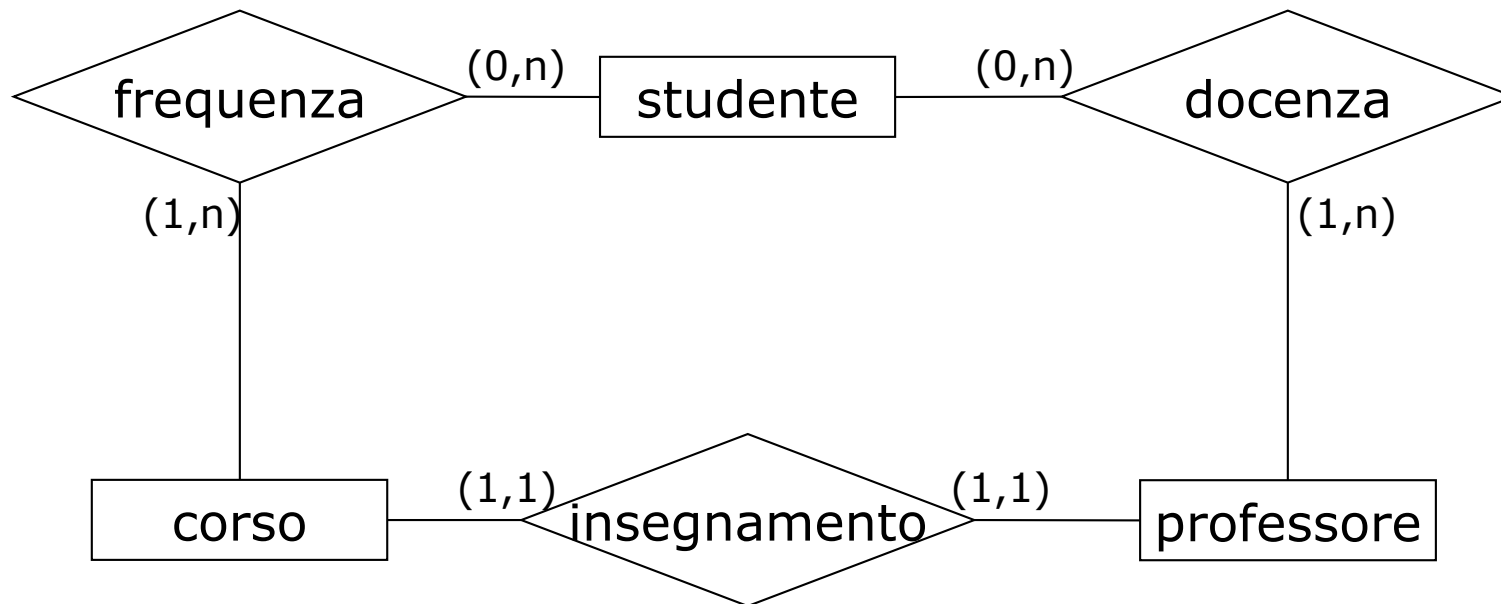
- *numero_abitanti* di una *città* può essere derivato contando le occorrenze di *residenza* a cui la *città* partecipa



Ridondanze: esempio (4)

Relazioni derivabili dalla composizione di altre relazioni in presenza di cicli:

- *docenza* può essere derivata da *frequenza* e *insegnamento*



Ridondanze

Vantaggi

- riduzione degli accessi e operazioni necessari per computare il dato derivato

Svantaggi

- maggiore occupazione di memoria (spesso non rilevante)
- necessita di effettuare operazioni aggiuntive per mantenere il dato derivato aggiornato

La decisione di mantenere o eliminare una ridondanza dipende dai **costi delle operazioni e dallo **spazio di memoria** occupato con o senza ridondanze (specificati dal carico applicativo)**

Ristrutturazione di schemi E-R

Composta da diverse fasi:

- analisi delle ridondanze
- **eliminazione delle gerarchie**
- partizionamento/accorpamento di entità e relazioni
- scelta degli identificatori primari

Eliminazione delle gerarchie

Le gerarchie devono essere eliminate perché non rappresentabili direttamente nei DBMS.

Tre approcci:

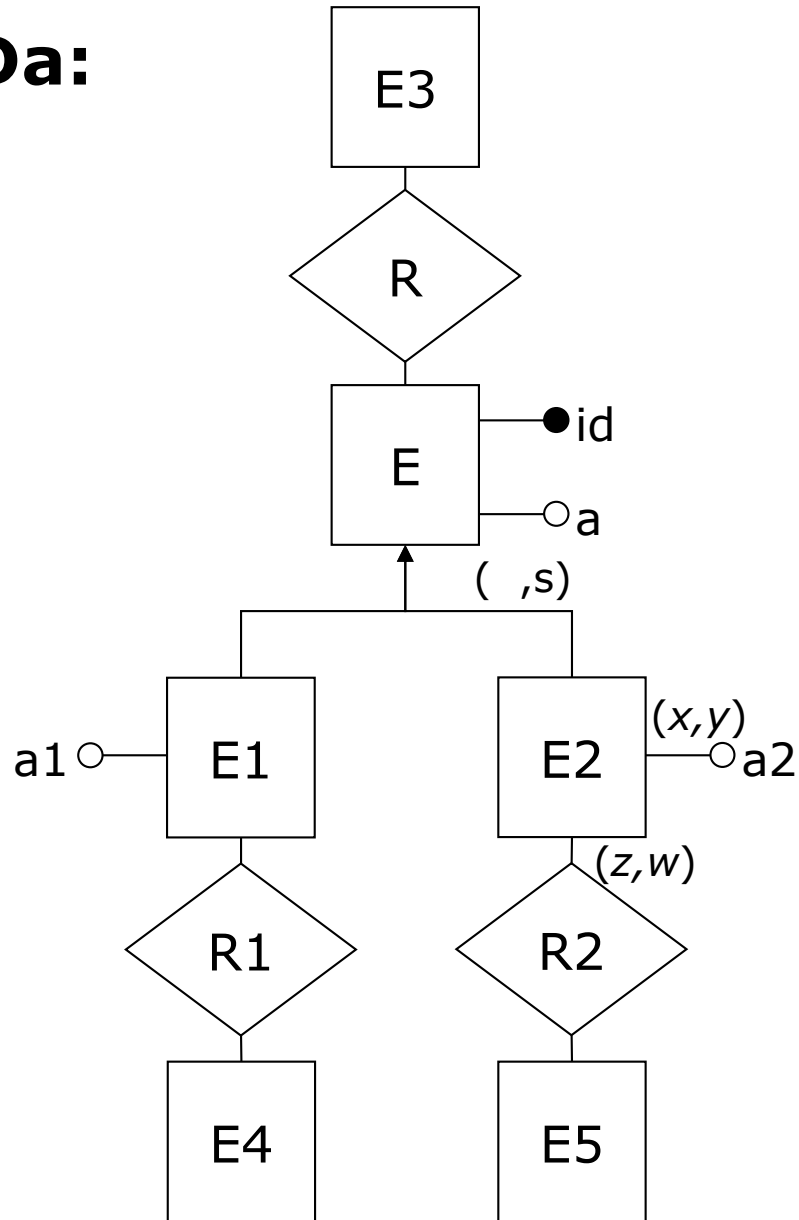
- **accorpamento delle figlie nel padre**
 - collasso verso l'alto
- **accorpamento del padre nelle figlie**
 - collasso verso il basso
- **mantenimento delle entità**

Accorpamento delle figlie nel padre (1)

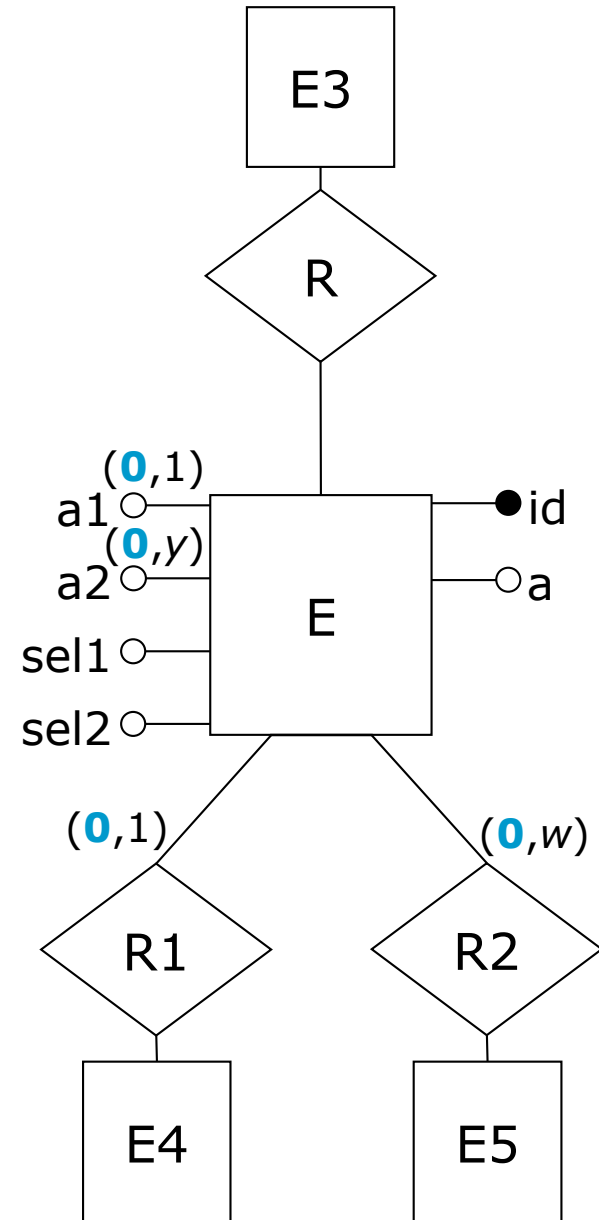
- Le entità **figlie** vengono **eliminate**
- Gli **attributi** e le **relazioni** delle entità figlie vengono **trasferiti al padre**, diventando **opzionali** (la cardinalità minima sia degli attributi sia delle relazioni dalla parte dell'entità padre diventa 0)
- Uno o più attributi (**selettori**) sono aggiunti al padre per indicare per ogni istanza a quale/i fra le N figlie l'istanza appartiene:
 - esclusiva: 1 selettore
 - totale: dominio del selettore ha N possibili valori
 - parziale: dominio del selettore ha N+1 possibili valori
 - sovrapposto: N selettori con dominio Vero/Falso
 - totale: almeno un selettore avrà valore Vero
 - parziale: i selettori potrebbero tutti avere valore Falso

Accorpamento delle figlie nel padre (2)

Da:



A:



Accorpamento delle figlie nel padre (3)

Vantaggi

- conveniente quando le operazioni non fanno molta distinzione tra le occorrenze e gli attributi del padre o delle figlie
 - si accede a una sola entità anziché a due o più attraverso la relazione

Svantaggi

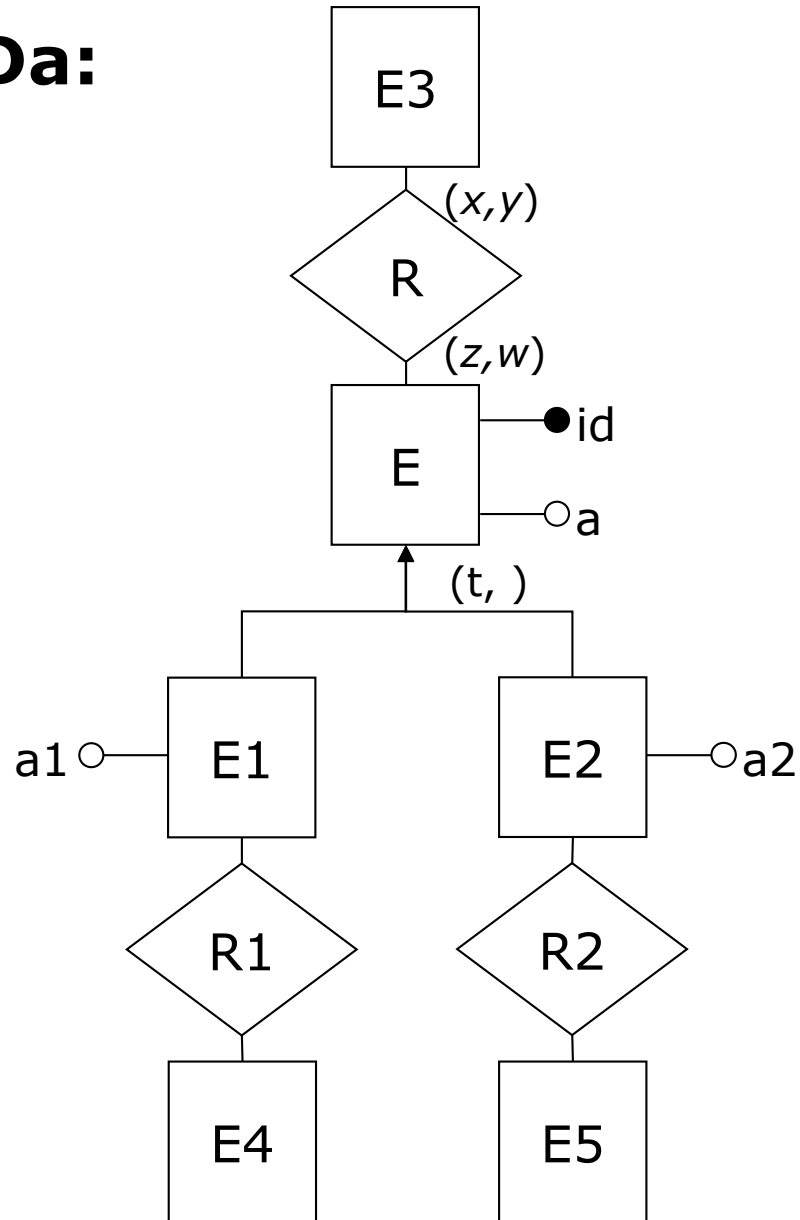
- gli attributi/relazioni obbligatori per le figlie diventano opzionali per il padre
 - si avranno dei valori nulli

Accorpamento del padre nelle figlie (1)

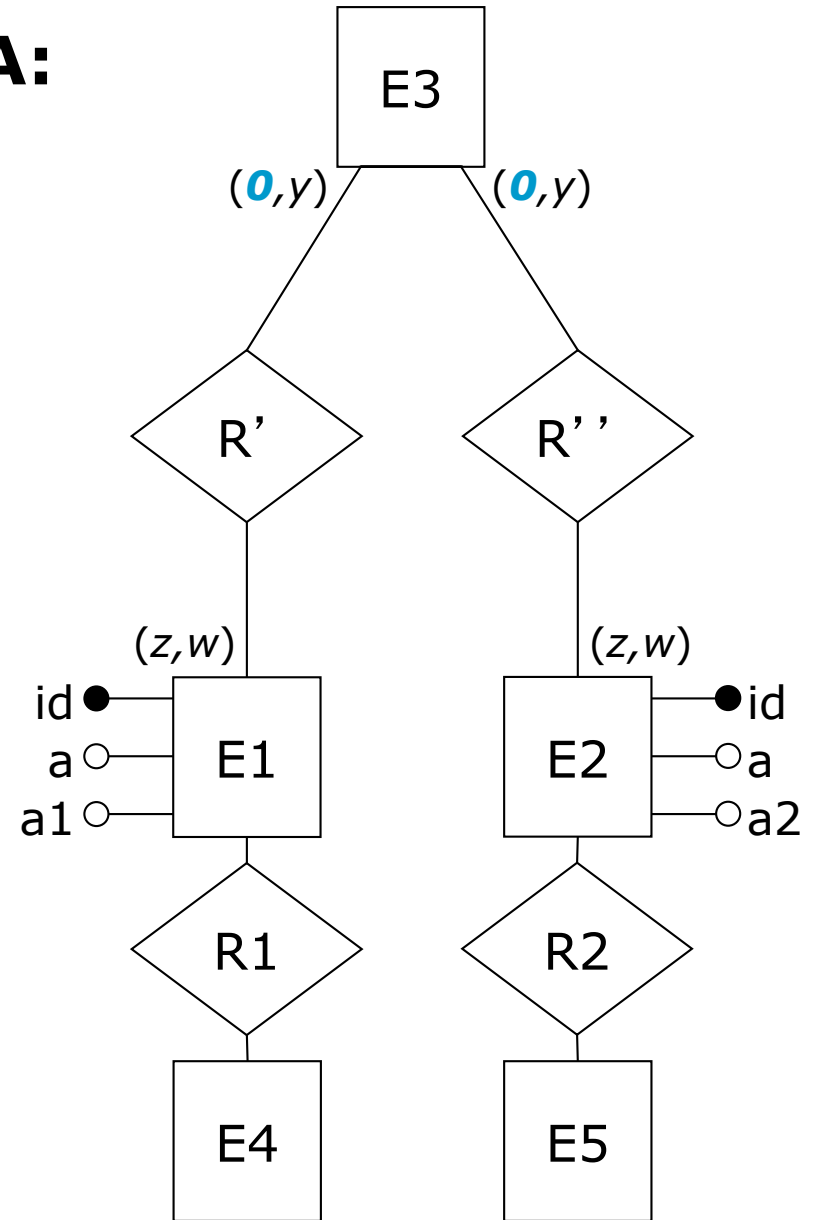
- L'entità **padre** viene **eliminata**
- Gli **attributi** e le **relazioni** del padre vengono trasferiti a ciascuna delle figlie
 - le entità che erano in relazione con il padre sono in relazione **opzionale** con le figlie (la cardinalità minima delle relazioni ereditate diventa 0 dalla parte delle entità che erano in relazione con il padre)
- Se la gerarchia è parziale non si può fare
 - a meno di aggiungere una entità figlia aggiuntiva (che raggruppa tutte le istanze che non appartengono ad alcuna figlia)

Accorpamento del padre nelle figlie (2)

Da:



A:



Accorpamento del padre nelle figlie (3)

Vantaggi

- conveniente per le operazioni che accedono separatamente alle figlie
- conveniente in presenza di molti attributi di specializzazione
 - collasso verso l'alto porterebbe a molti valori nulli

Svantaggi

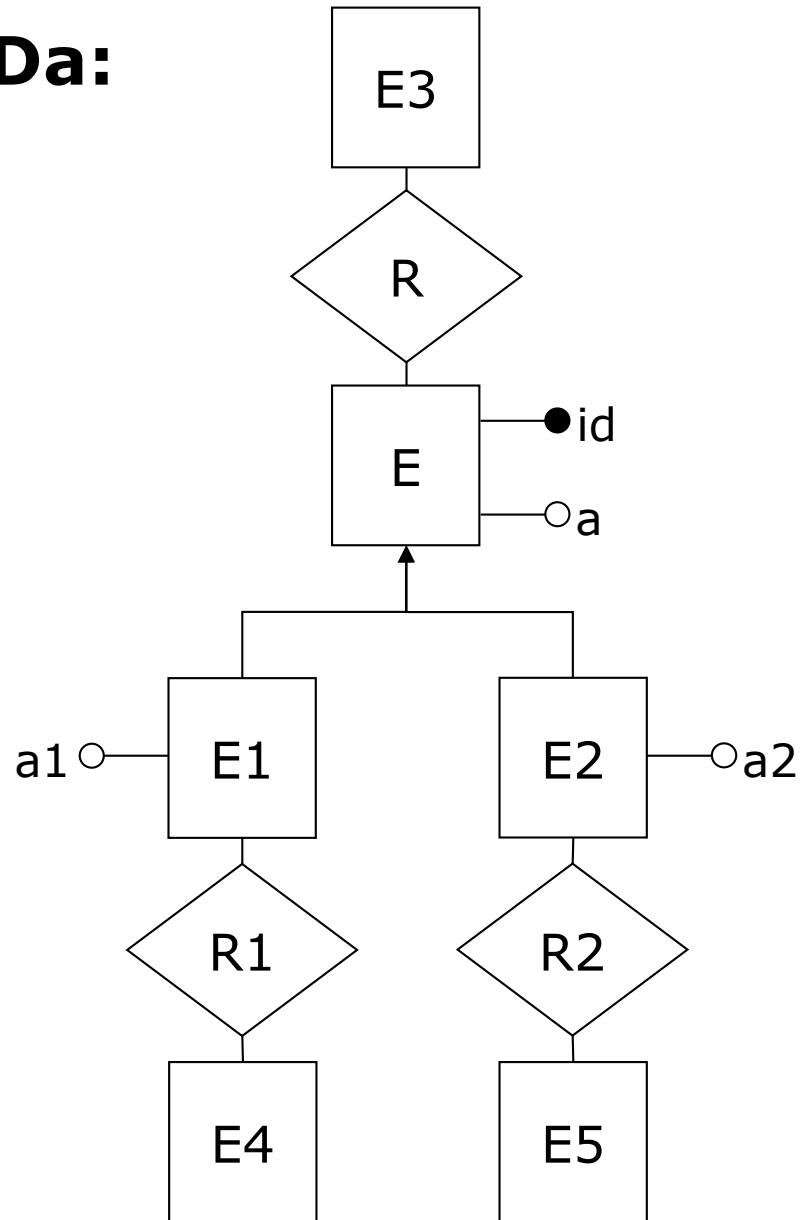
- se la gerarchia è parziale non si può fare
 - a meno di aggiungere una entità figlia aggiuntiva
- se la gerarchia è sovrapposta introduce ridonanze
 - istanze appartengono a più figlie con attributi comuni replicati

Mantenimento delle entità (1)

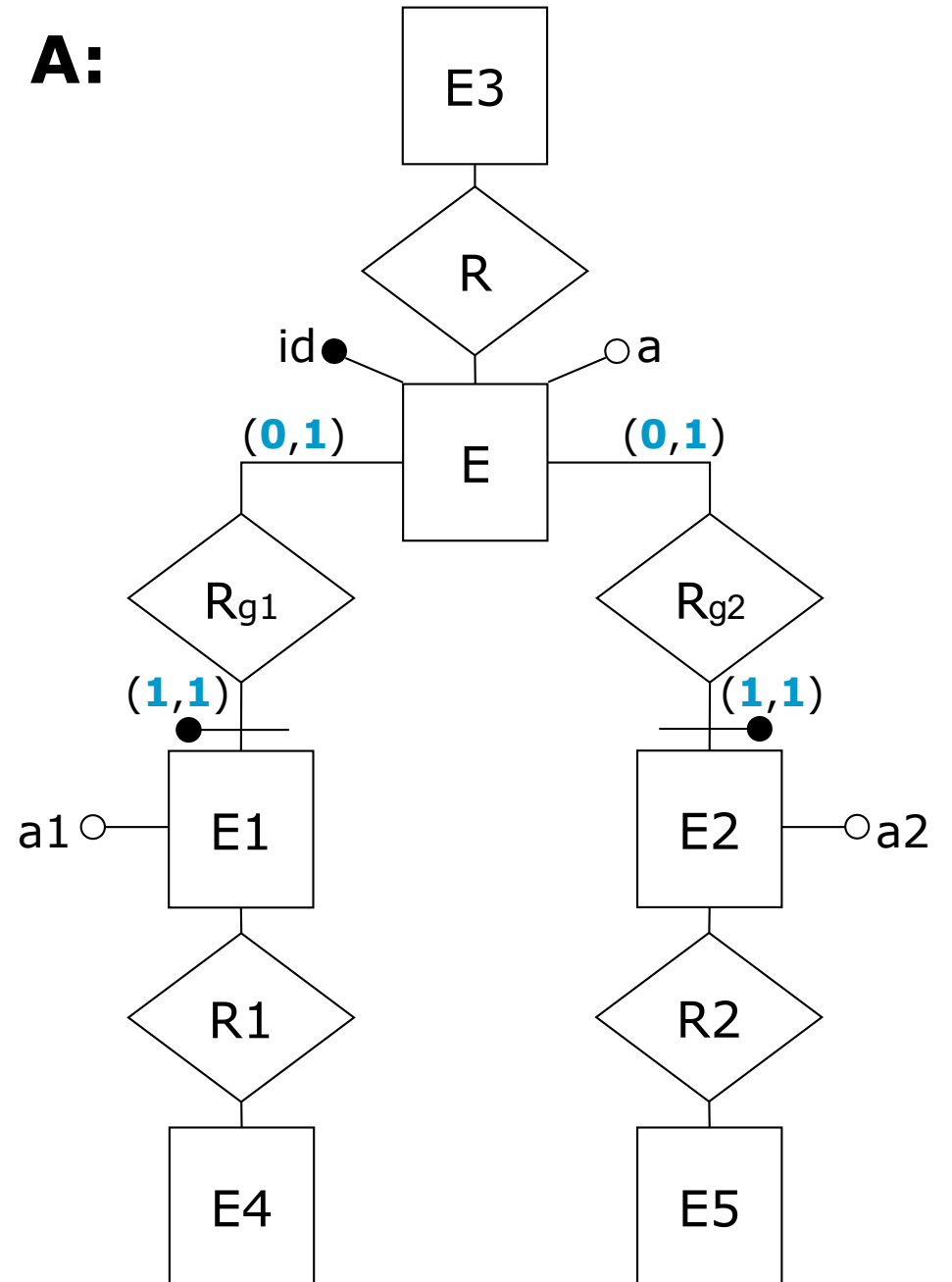
- Il legame gerarchico viene **sostituito con relazioni** fra il padre e ciascuna delle figlie
- Le entità **figlie** sono **identificate esternamente** tramite la relazione con il padre (entità **deboli**)
- Nello schema bisogna aggiungere vincoli che derivano dalla gerarchia
 - esclusiva: ogni istanza del padre può partecipare alla relazione con una sola delle figlie
 - totale: ogni istanza del padre deve partecipare alla relazione con almeno una delle figlie

Mantenimento delle entità (2)

Da:



A:



Mantenimento delle entità (3)

Vantaggi

- è sempre possibile (a prescindere dalla copertura)
- conveniente in presenza di operazioni che si riferiscono solo alle occorrenze (e agli attributi) delle figlie

Svantaggi

- le operazioni che accedono agli attributi di padre e figlie richiedono più accessi
- necessità di mantenere i vincoli introdotti

Eliminazione delle gerarchie (1)

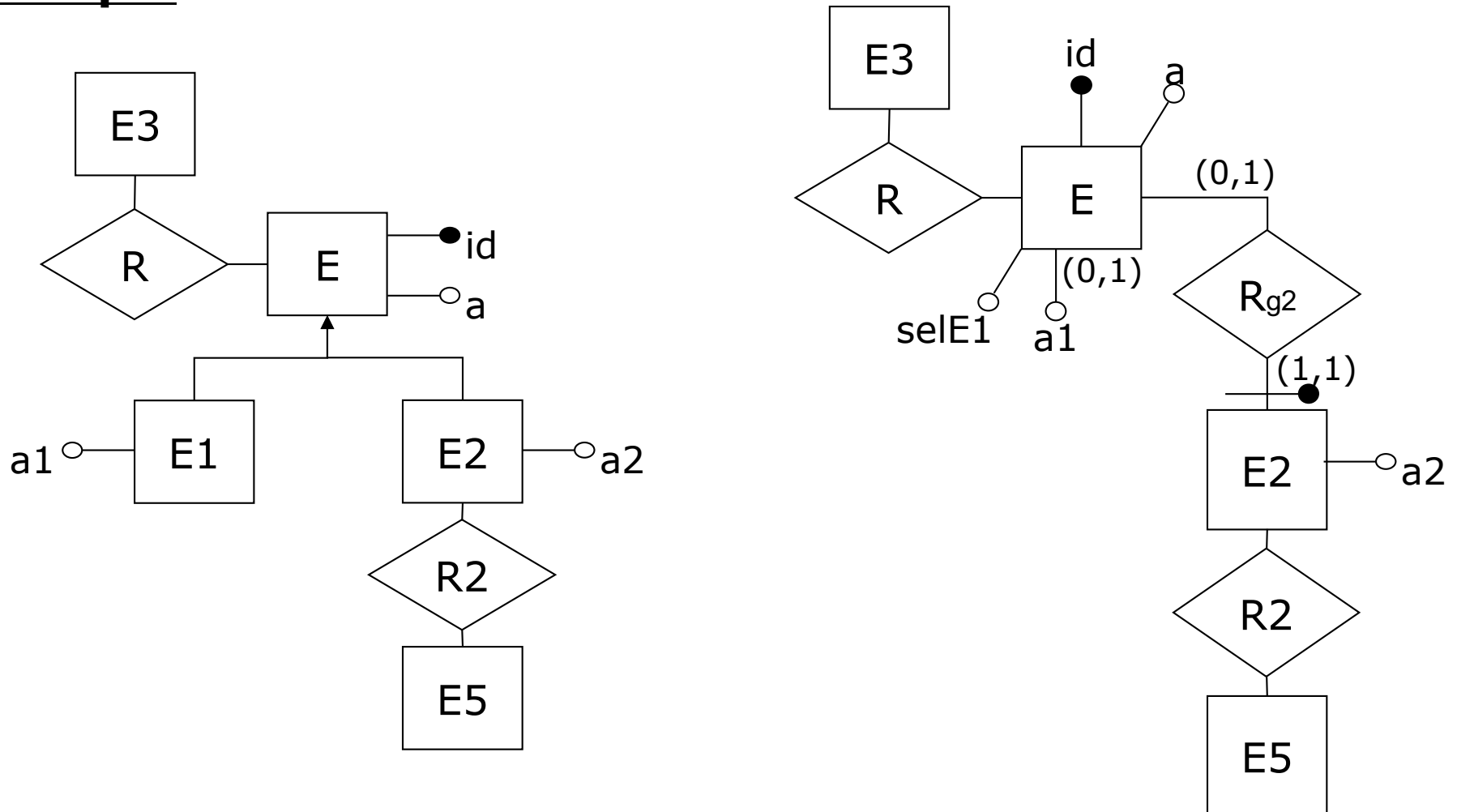
La scelta fra le varie alternative dipende da valutazioni sull'occupazione di memoria e sul costo delle operazioni nei diversi casi

- il semplice conteggio degli accessi e delle occorrenze non è sufficiente
- è necessario considerare anche la dimensione dei domini e la quantità di dati recuperabili con una operazione di accesso a memoria secondaria
 - es.: mantenimento delle entità richiede più accessi per le operazioni. Però ha il vantaggio di generare entità con pochi attributi (che si traducono in relazioni di piccole dimensioni \Rightarrow maggior numero di tuple recuperabili con un solo accesso fisico).

Eliminazione delle gerarchie (2)

Sono possibili ristrutturazioni ibride (che combinano più di una alternativa)

Esempio



Ristrutturazione di schemi E-R

Composta da diverse fasi:

- analisi delle ridondanze
- eliminazione delle gerarchie
- partizionamento/accorpamento di entità e relazioni
- scelta degli identificatori primari

Partizionamento/accorpamento

Per **aumentare efficienza** delle operazioni:

- partizionamento di entità/relazioni per
 - dividere attributi di una stessa entità/relazione che sono accedute da operazioni diverse
- accorpamento di entità/relazioni per
 - unire in una stessa entità/relazione attributi che sono acceduti spesso insieme

Per **tradurre concetti non supportati** dal modello logico:

- eliminazione degli attributi multivalore
- eliminazione degli attributi composti

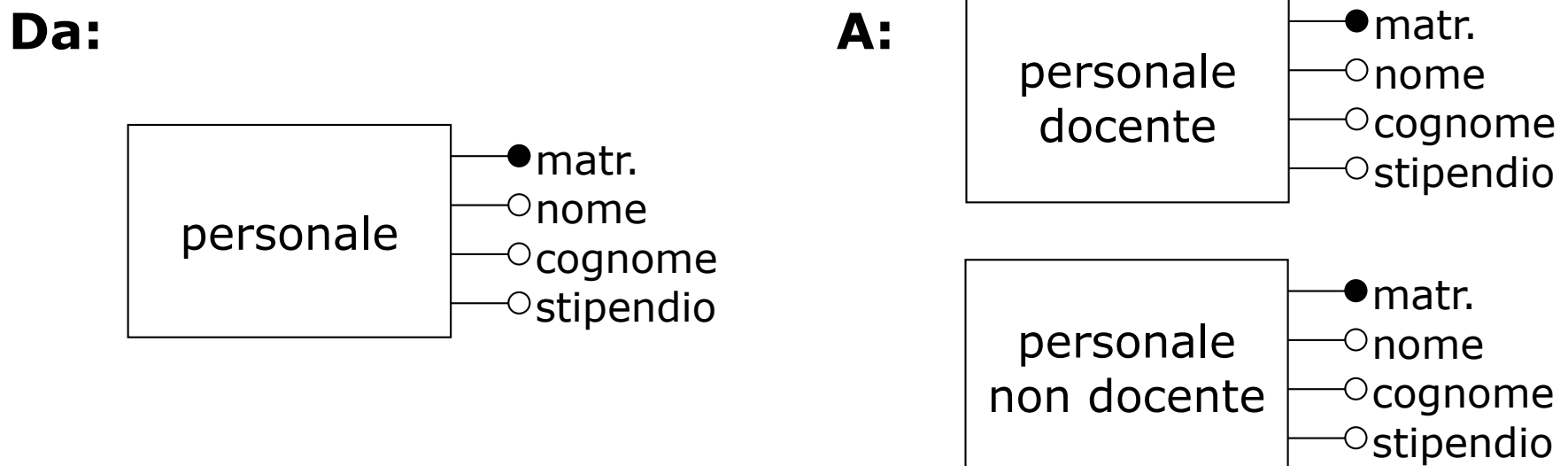
Partizionamento di entità (1)

Partizionamento orizzontale

- Una entità **E** viene **sostituita** con due entità **E1** e **E2** che **si dividono le istanze di E** (equivale a sostituire E con sue figlie di generalizzazione)



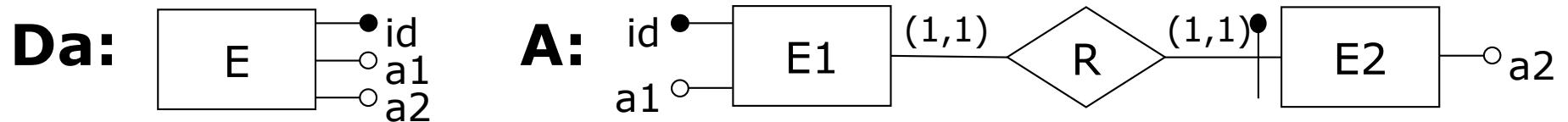
Esempio



Partizionamento di entità (2)

Partizionamento verticale

- Una entità **E** viene **sostituita con** due entità **E1** e **E2** che **si dividono gli attributi/relazioni di E** e che sono in relazione. Una delle entità è identificata esternamente da quella che mantiene l'identificatore



Partizionamento di entità (3)

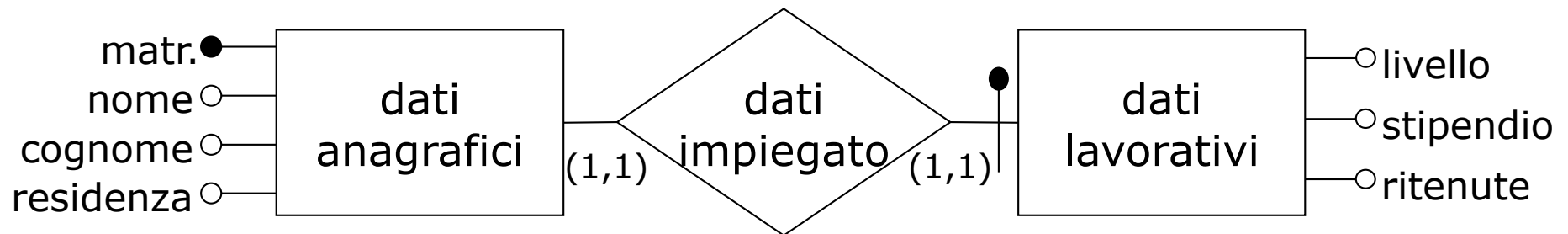
Partizionamento verticale

Esempio

Da:



A:



Accorpamento di entità

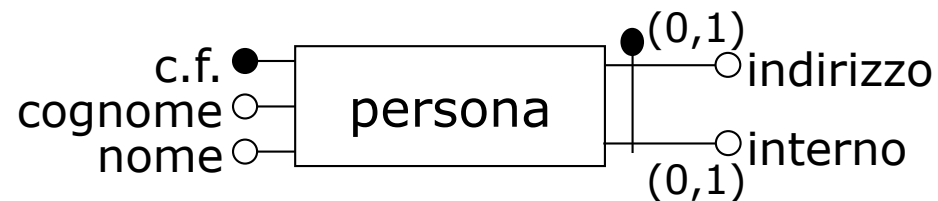
Due entità **E1** e **E2** legate da una relazione (1:1) o (1:n) vengono **sostituite con** una entità **E** che ha gli attributi e le relazioni di entrambe

Esempio

Da:



A:



Partizionamento/accorpamento di relazioni

Partizionamenti/accorpamenti possono essere fatti anche su relazioni

Partizionamento

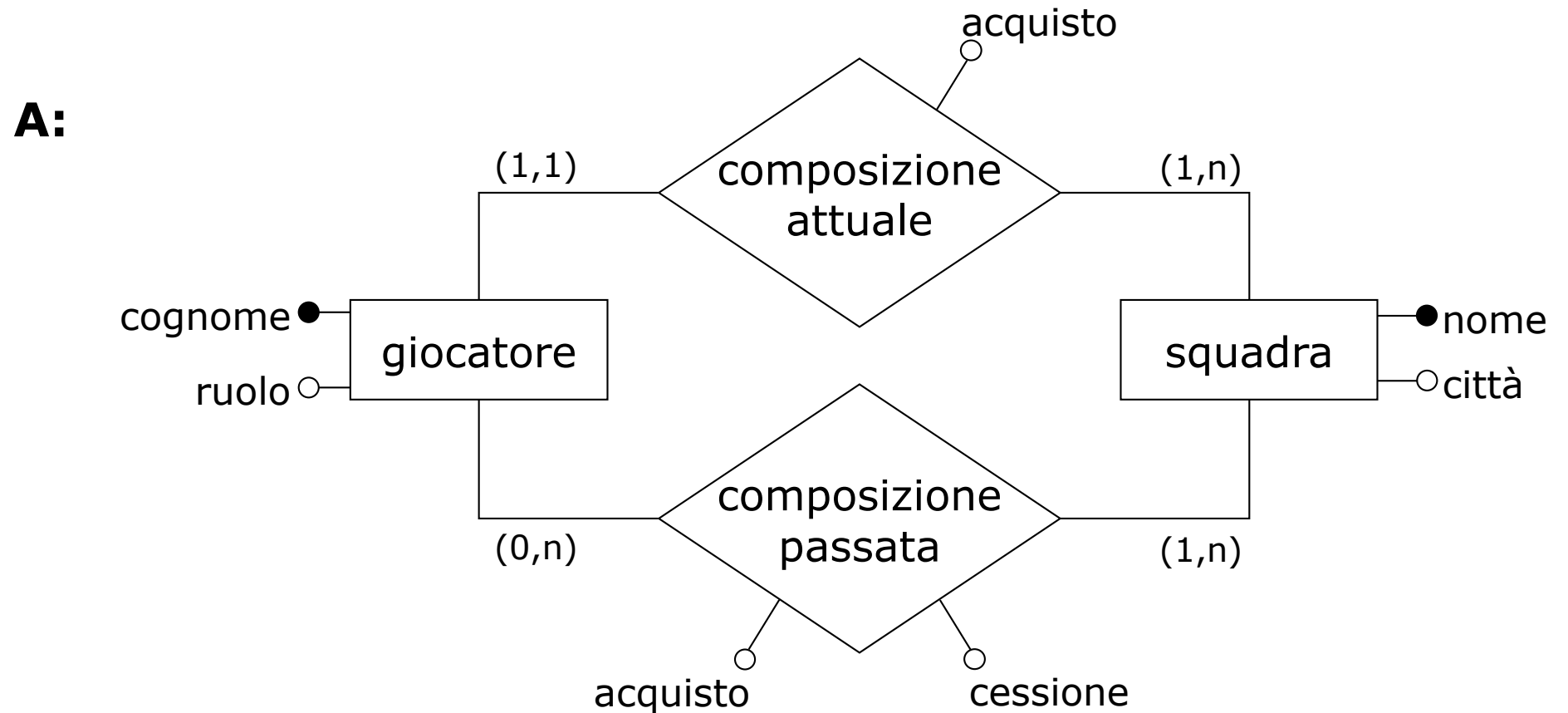
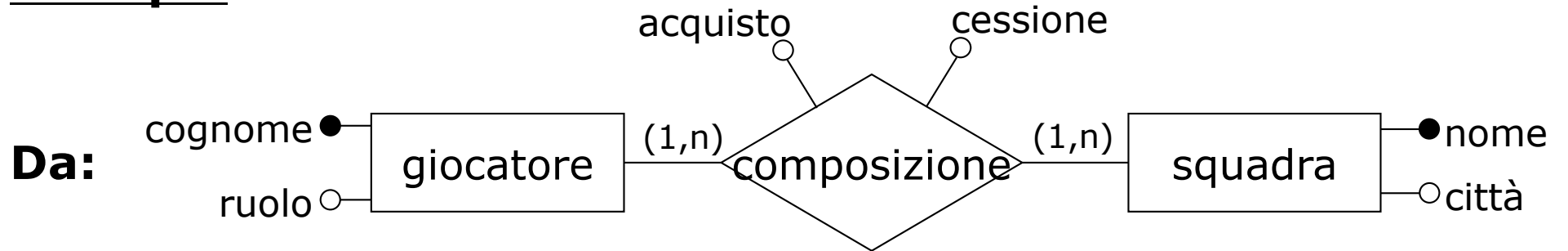
- una relazione **R** viene **sostituita con** due relazioni **R1** e **R2 fra le stesse entità** per separare le occorrenze di **R** accedute sempre separatamente

Accorpamento

- due relazioni **R1** e **R2** tra le stesse entità, che si riferiscono a due aspetti del medesimo concetto e le cui occorrenze siano sempre accedute contemporaneamente, vengono **sostituite con** una sola relazione **R**

Partizionamento di relazioni

Esempio

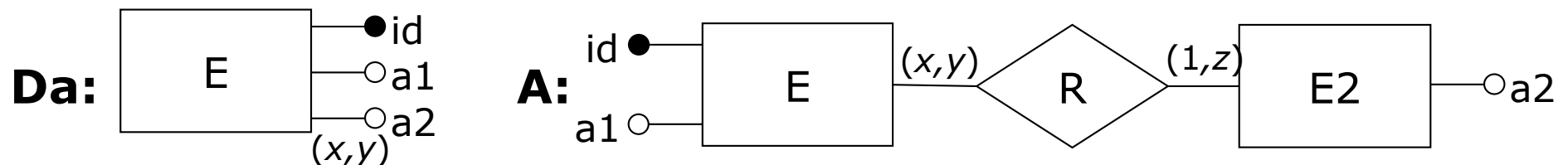


Eliminazione di attributi multivalore (1)

Attributi multivalore non sono supportati dal modello relazionale

- si crea una **nuova entità** che contiene l'attributo multivalore e che sia **collegata all'entità originale** da una relazione
 - cardinalità della relazione dipende dalla cardinalità originale e dal fatto che i valori dell'attributo possano comparire più di una volta

Esempio



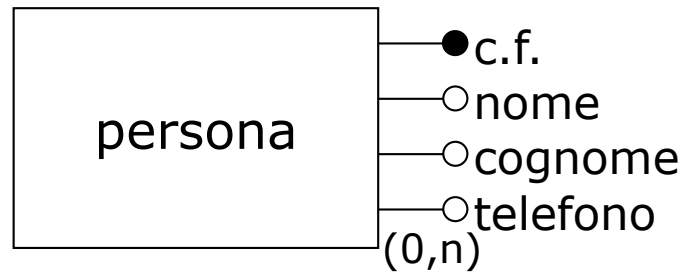
$y > 1$

$z = 1$ se il valore non può essere ripetuto, n altrimenti

Eliminazione di attributi multivalore (2)

Esempio

Da:



A:



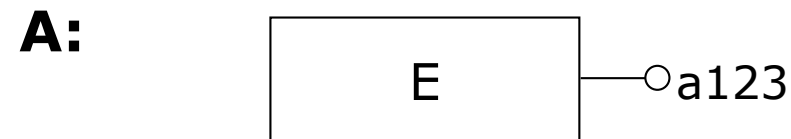
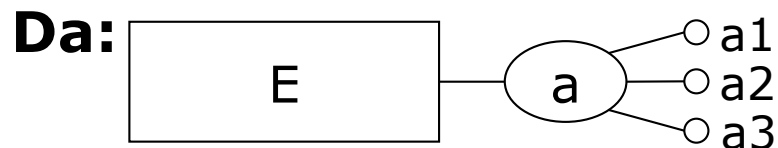
Eliminazione di attributi composti

Attributi composti non sono supportati dal modello relazionale.

Due soluzioni:

- si **elimina la suddivisione** in componenti e si considera il composto come un attributo semplice
- si **elimina l'attributo composto** e si considerano le sue componenti come attributi semplici

Esempio



oppure:



Scelta degli identificatori primari (1)

Per ogni entità deve essere selezionato un identificatore primario

- se l'entità ha più identificatori (**chiavi candidate**) bisogna decidere quale verrà utilizzato come primario (**chiave primaria**)
- gli identificatori che non sono selezionati come primari potranno costituire, in sede di progettazione fisica, indici secondari

Scelta degli identificatori primari (2)

È preferibile scegliere come primario un identificatore:

- utilizzato in molte operazioni per accedere all'entità
- con pochi attributi
- con solo attributi interni (in contrasto a esterni)

Non possono essere scelti come primari identificatori:

- con attributi che possono avere valori nulli

Traduzione verso il modello logico

Traduce lo schema concettuale ristrutturato in uno schema logico

- fa riferimento allo specifico modello logico scelto
- può includere una ulteriore ottimizzazione basata sulle caratteristiche del modello logico

Utilizziamo come modello logico il modello relazionale

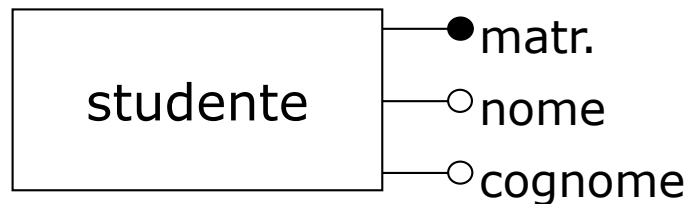
- per distinguere la relazione del modello E-R dalla relazione del modello relazionale, ci riferiamo a quest'ultima come **tabella**

Traduzione di entità (1)

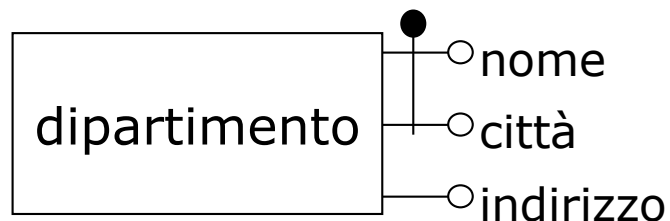
Ogni entità forte E è tradotta in una tabella R con

- **attributi**: gli attributi di E
- **chiave primaria**: l'identificatore primario di E
- **vincoli**: es., non nullità su attributi obbligatori

Esempi



STUDENTE(matr, nome, cognome)



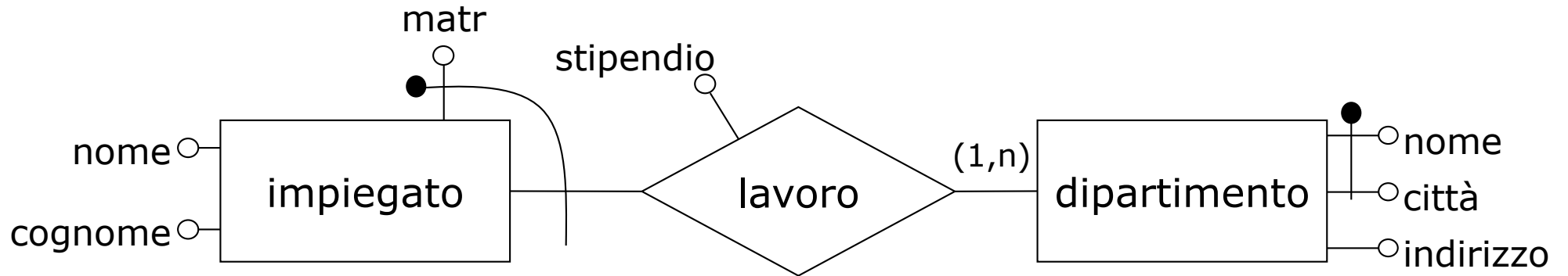
DIPARTIMENTO(nome, città, indirizzo)

Traduzione di entità (2)

Ogni entità debole E identificata esternamente da E' è tradotta in una tabella R con

- **attributi**: gli attributi di E più l' identificatore di E' più gli attributi della relazione che identifica E esternamente
- **chiave primaria**: l' identificatore primario di E
 - attributi interni ad E unitamente all' identificatore di E'
- **vincoli**:
 - **integrità referenziale** sull' identificatore di E'
 - **altri vincoli**: es., non nullità su attributi obbligatori

Traduzione di entità: esempio (1)



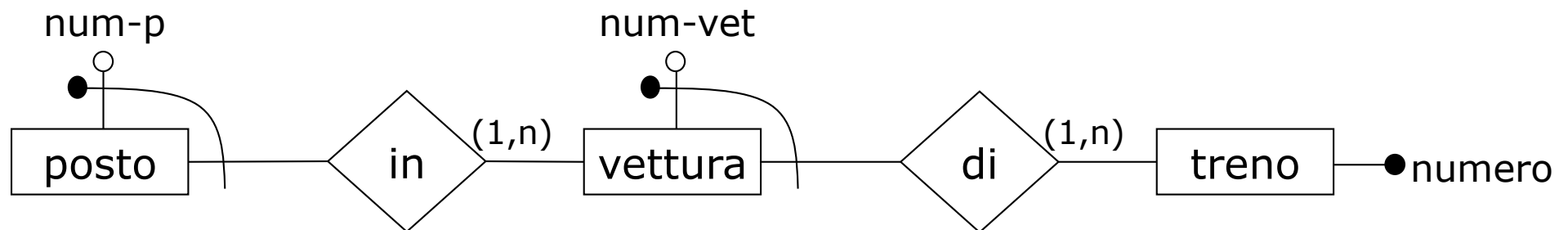
DIPARTIMENTO(nome,città,indirizzo)

IMPIEGATO(matr,nome-dip,città-dip,nome,cognome,stipendio)

Traduzione di entità: esempio (2)

Per catene di entità deboli il processo di traduzione deve seguire il percorso a partire dall'entità forte

Esempio



TRENO(numero)

VETTURA(num-vet, num-treno)

POSTO(num-p, num-vet, num-treno)

Traduzione di relazioni: standard (1)

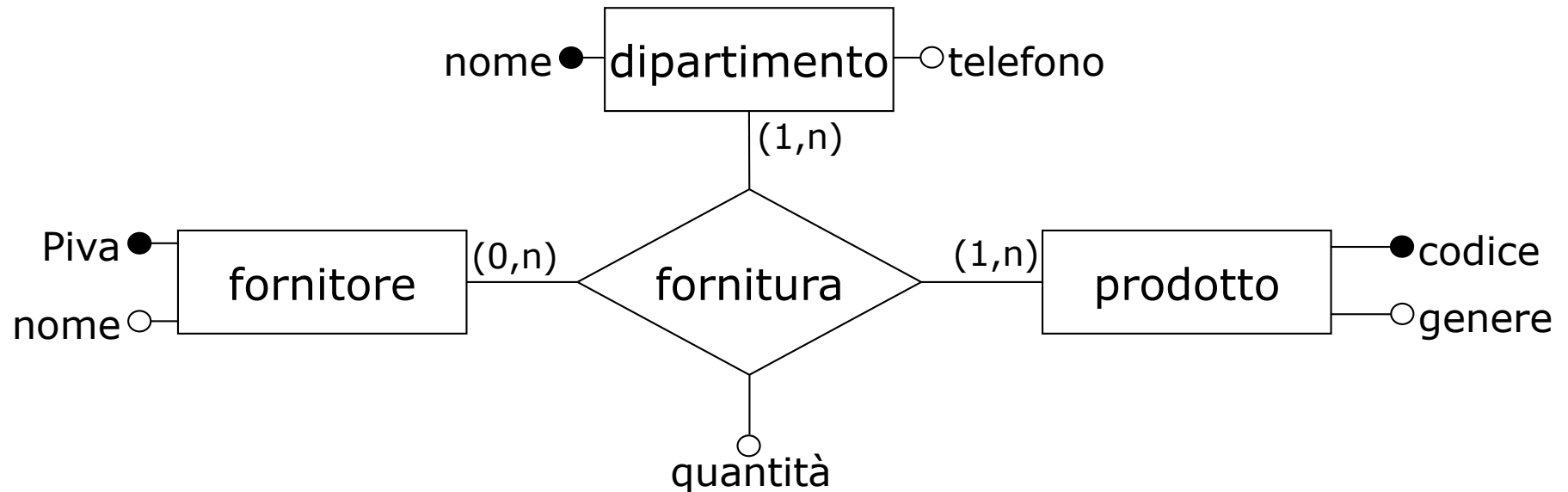
Ogni relazione R fra le entità E_1, E_2, \dots, E_n è tradotta in una tabella R con

- **attributi**: gli attributi di R più gli identificatori di E_1, E_2, \dots, E_n
- **chiave primaria**: l'unione K degli identificatori di E_1, E_2, \dots, E_n è superchiave; la chiave K o un suo sottoinsieme
 - dipende dalla cardinalità della relazione o da eventuali dipendenze
- **vincoli**:
 - **integrità referenziale** sugli identificatori di E_1, E_2, \dots, E_n
 - **altri vincoli**: es., non nullità su attributi obbligatori

NOTA: Le relazioni che legano entità deboli a forti non sono tradotte

Traduzione di relazioni: standard (2)

Esempio



DIPARTIMENTO(nome,telefono)

FORNITORE(Piva,nome)

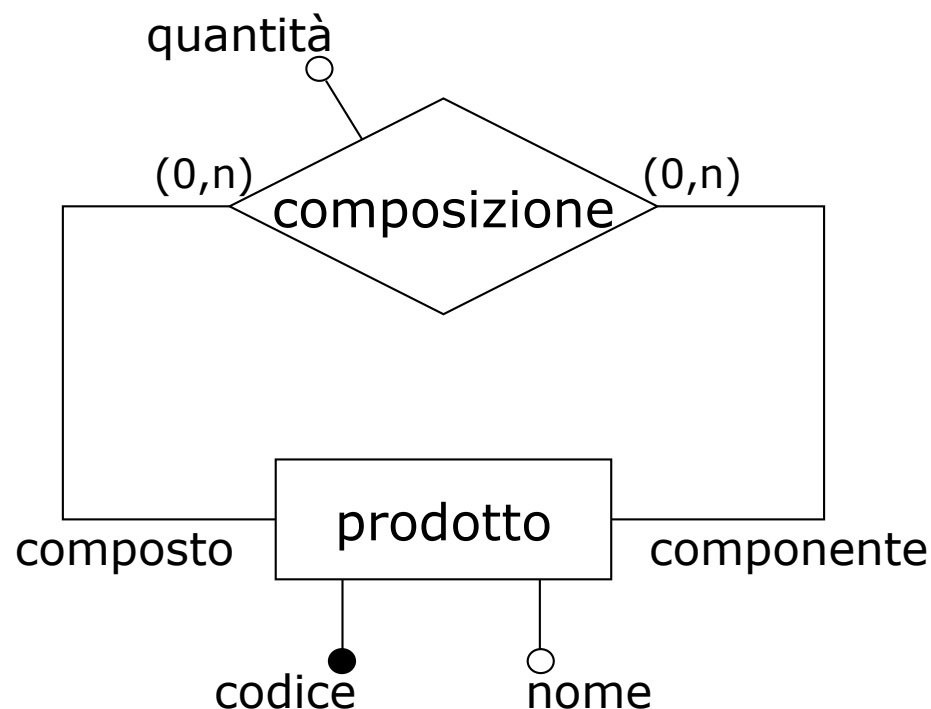
PRODOTTO(codice,genere)

FORNITURA(nome-dip,Piva-forn,cod-prodotto,quantità)

Traduzione di relazioni: standard (3)

Nel caso di **autorelazioni** la tabella che descrive la relazione contiene due volte l' identificatore dell' entità, che deve quindi essere necessariamente ridenominata

Esempio



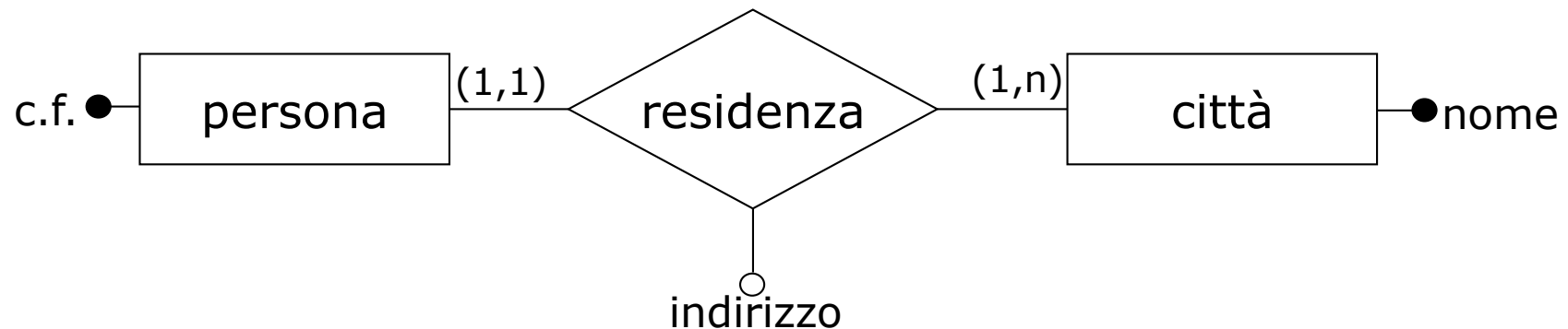
PRODOTTO(codice, nome)

COMPOSIZIONE(composto, componente, quantità)

Traduzione di relazioni: standard (4)

Nel caso di relazioni (1:n) la chiave è l'identificatore dell'entità che partecipa con cardinalità massima 1

Esempio



PERSONA(c.f.)

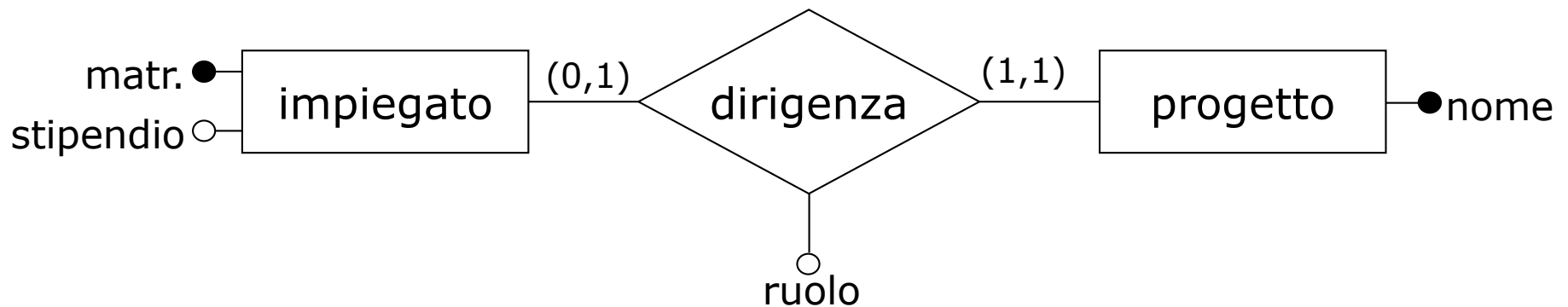
CITTÀ(nome)

RESIDENZA(c.f., città, indirizzo)

Traduzione di relazioni: standard (5)

Nel caso di relazioni (1:1) la chiave è l'identificatore di una delle entità

Esempio



IMPIEGATO(matr., stipendio)

PROGETTO(nome)

DIRIGENZA(matr., nome-prog, ruolo)

oppure

DIRIGENZA(matr., nome-prog, ruolo)

Traduzione di relazioni

- La traduzione standard è sempre possibile ed è l' unica possibile nel caso di relazioni (n:m)

Per relazioni (1:1) o (1:n) sono possibili traduzioni alternative che fondono in una stessa tabella la relazione e una o entrambe le entità

- **Vantaggi:**

- minor numero di tabelle
- minor numero di join per navigare la relazione

- **Svantaggi:**

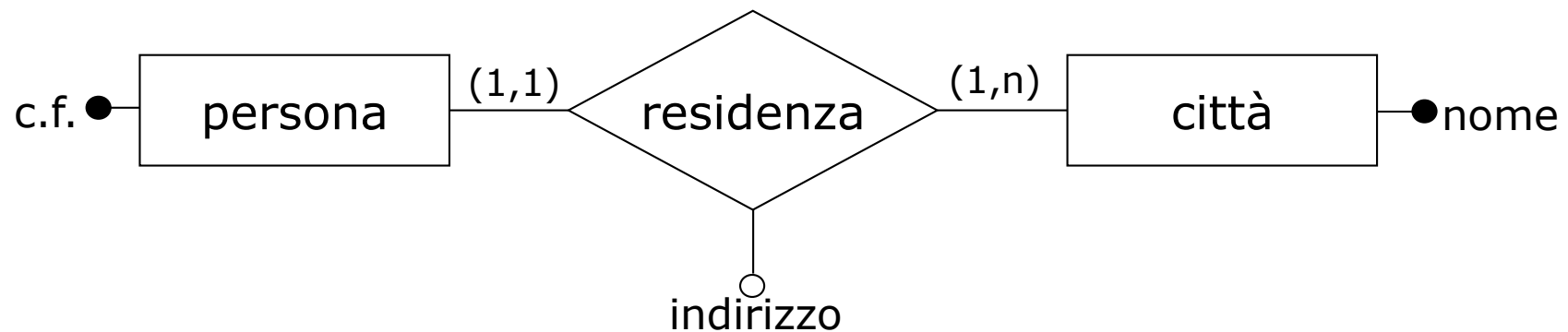
- penalizzate le operazioni che consultano soltanto gli attributi di una entità che è stata fusa

Traduzione di relazioni (1:n)

La relazione può essere **fusa** nella entità che partecipa con **cardinalità massima 1**

- se la cardinalità minima è 0 gli attributi inglobati dalla relazione assumeranno valori nulli per le istanze che non partecipano alla relazione

Esempio



PERSONA(c.f.)

CITTÀ(nome)

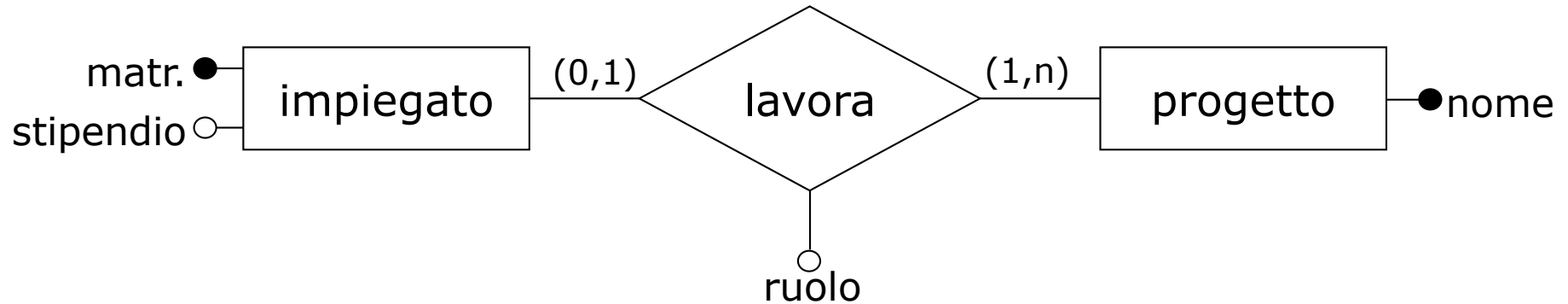
RESIDENZA(c.f.,città,indirizzo)

oppure

PERSONA(c.f.,città,indirizzo)

CITTÀ(nome)

Traduzione di relazioni (1:n): esempio



IMPIEGATO(matr., stipendio)

PROGETTO(nome)

LAVORA(matr., nome-progetto, ruolo)

oppure

IMPIEGATO(matr., stipendio, nome-progetto*, ruolo*)

PROGETTO(nome)

* = può assumere valori nulli

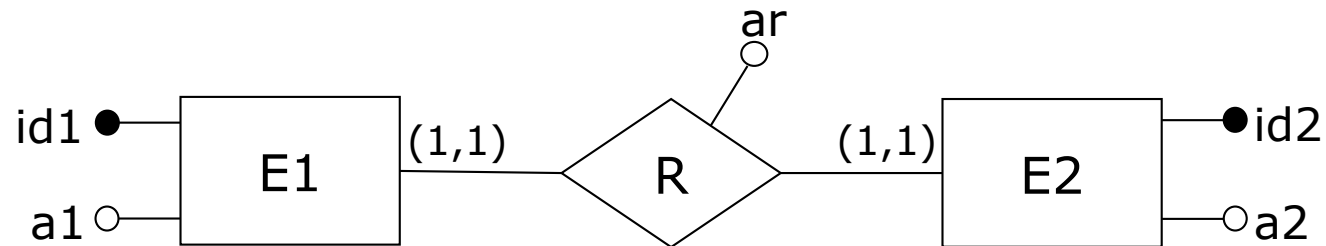
Traduzione di relazioni (1:1) (1)

La relazione può essere **fusa**, distinguiamo tre casi a seconda delle **cardinalità minime** con cui le entità **E1** e **E2** partecipano alla relazione

- **entrambe 1**: (1,1) e (1,1)
- **una 1 e l'altra 0**: (0,1) e (1,1)
- **entrambe 0**: (0,1) e (0,1)

Traduzione di relazioni (1:1) (2)

Entrambe le entità partecipano con cardinalità (1,1)



- standard:

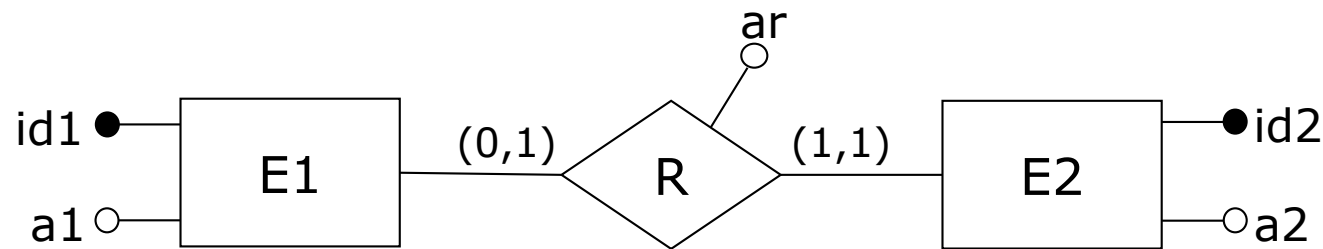
- $E1(\underline{id1}, a1) \quad E2(\underline{id2}, a2)$
 $R(\underline{id1}, id2, ar)$ **oppure** $R(id1, \underline{id2}, ar)$

- alternative:

- $E1(\underline{id1}, a1, id2, ar) \quad E2(\underline{id2}, a2)$
- $E1(\underline{id1}, a1) \quad E2(\underline{id2}, a2, id1, ar)$
- $E12(\underline{id1}, a1, id2, a2, ar)$
- $E21(\underline{id2}, a2, id1, a1, ar)$

Traduzione di relazioni (1:1) (3)

Una delle entità partecipa con cardinalità (0,1)



- standard:

- $E1(\underline{id1}, a1) \quad E2(\underline{id2}, a2)$
 $R(\underline{id1}, id2, ar)$ **oppure** $R(id1, \underline{id2}, ar)$

- alternative:

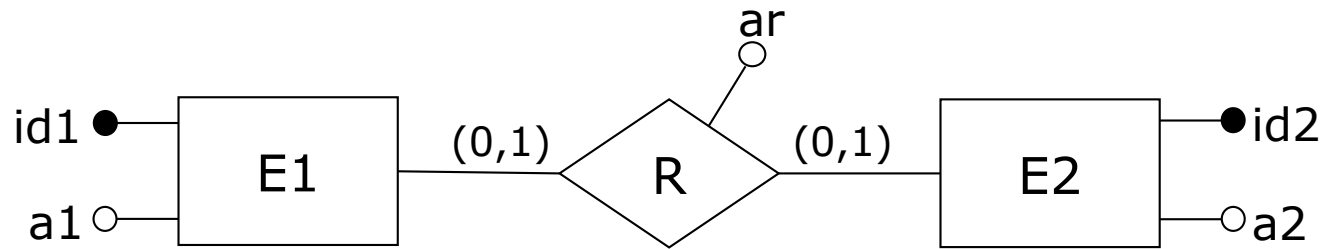
- $E1(\underline{id1}, a1, id2^*, ar^*) \quad E2(\underline{id2}, a2)$
- $E1(\underline{id1}, a1) \quad E2(\underline{id2}, a2, id1, ar)$
- $E12(\underline{id1}, a1, id2^*, a2^*, ar^*)$

* = può assumere valori nulli

Attenzione: $E21(\underline{id2}, a2, id1, a1, ar)$ NON VALIDA

Traduzione di relazioni (1:1) (4)

Entrambe le entità partecipano con cardinalità (0,1)



- standard:

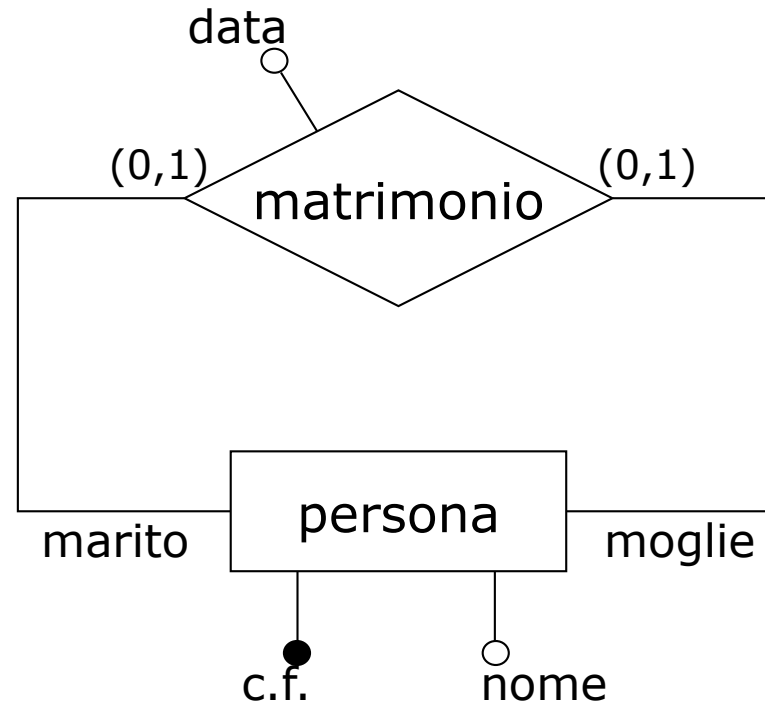
- $E1(\underline{id1}, a1) \quad E2(\underline{id2}, a2)$
 $R(\underline{id1}, \underline{id2}, ar)$ **oppure** $R(id1, id2, ar)$

- alternative:

- $E1(\underline{id1}, a1, id2^*, ar^*) \quad E2(\underline{id2}, a2)$
- $E1(\underline{id1}, a1) \quad E2(\underline{id2}, a2, id1^*, ar^*)$

* = può assumere valori nulli

Traduzione di relazioni (1:1) (5)



- standard:

- PERSONA(c.f.,nome)

- MATR(c.f.lui,c.f.lei,data) **oppure** MATR(c.f.lui,c.f.lei,data)

- alternativa:

- PERSONA(c.f.,nome,c.f.altro*,data*)

* = può assumere valori nulli

Documentazione di schemi logici (1)

Risultati della progettazione logica:

- schema logico
- documentazione associata
 - ereditata e adattata dallo schema concettuale
 - introdotta per descrivere i nuovi vincoli (es., integrità referenziale)

Lo schema e i vincoli di integrità referenziale possono essere rappresentati con notazione grafica

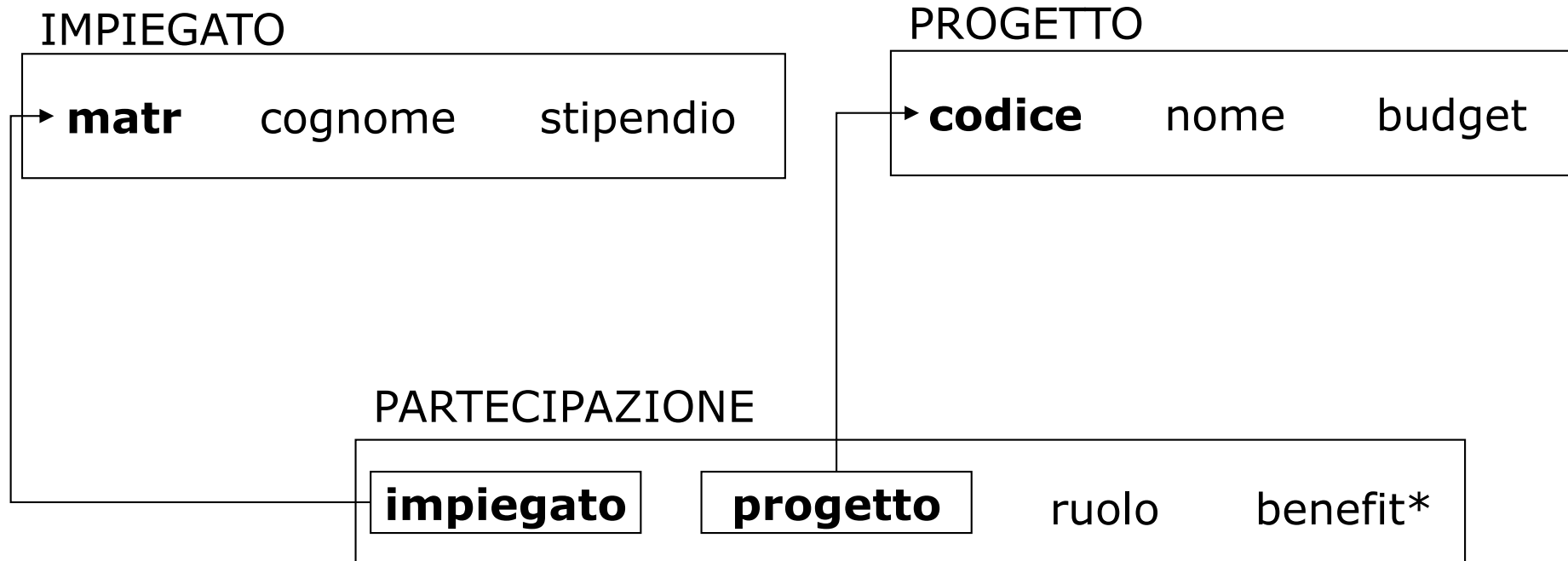
Documentazione di schemi logici (2)

Possibile notazione grafica:

- schemi di relazione in rettangoli
- chiavi in grassetto
- possibilità di valori nulli con asterisco
- integrità referenziale con frecce (con partenza evidenziata da un rettangolo e con eventuale etichetta)

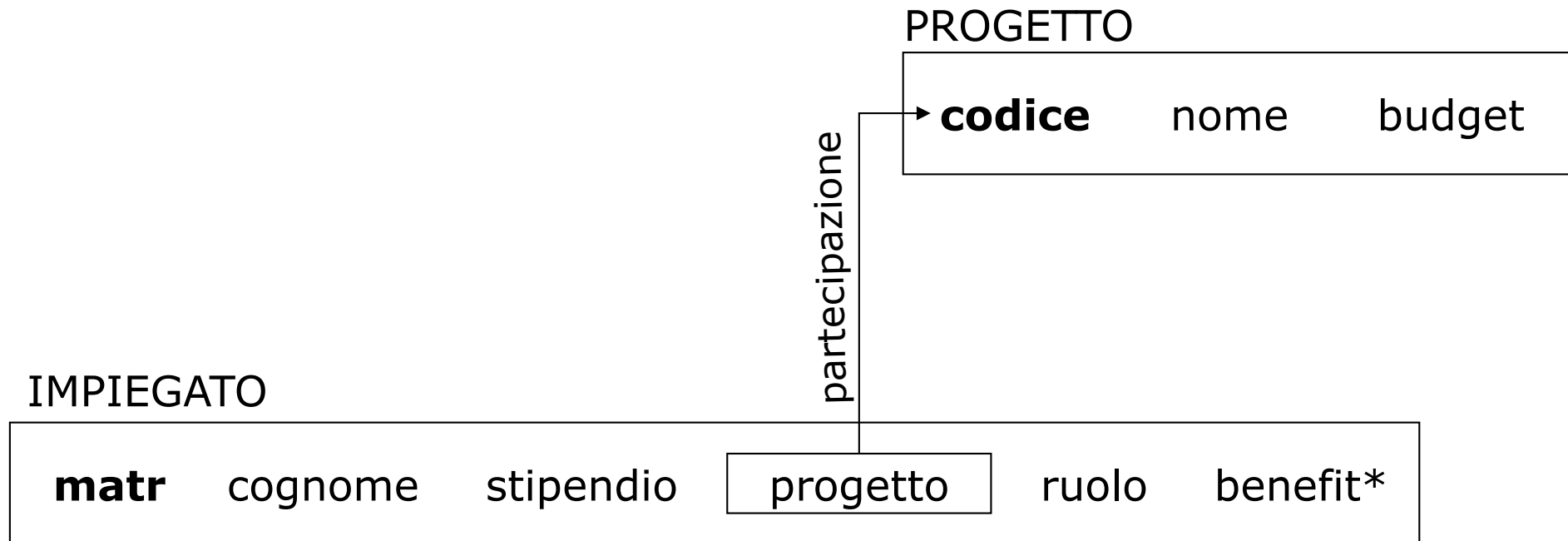
Documentazione di schemi logici (3)

Esempio



Documentazione di schemi logici (4)

Esempio



Verifica di schemi

Lo schema risultante dalla progettazione logica deve essere verificato rispetto a criteri di qualità

- **forme normali** definiscono le proprietà che uno schema deve soddisfare
 - assenza di ridondanze e anomalie di comportamento
- se la progettazione logica è stata fatta bene lo schema sarà in forma normale, altrimenti
 - processo di **normalizzazione**

Strumenti CASE per progettazione logica

La fase di progettazione logica viene generalmente supportata da strumenti CASE

- traduzione automatica da schema ristrutturato a modello relazionale
- la fase di ristrutturazione deve essere invece fatta manualmente
 - i sistemi automatici non sono in grado di valutare le diverse alternative

Progettazione fisica (1)

Ingresso

- schema logico
- caratteristiche del DBMS scelto
- previsioni sul carico applicativo

Uscita

- **schema fisico** della base di dati
 - definizioni di relazioni con DDL
 - **strutture fisiche** di accesso con relativi parametri
 - particolare importanza ha l'individuazione degli **indici**

Progettazione fisica (2)

Indici:

- definiti su attributi si traducono in speciali strutture di accesso che garantiscono **accesso diretto**
- **ottimizzano accesso** agli attributi su cui sono definiti

Individuazione degli indici:

- spesso fatta in modo empirico con una fase di **regolazione (tuning)** che può introdurre nuovi indici per migliorare le prestazioni

Progettazione – Esercizi

Esercitazione 1

Requisiti (1)

L'associazione **Ed È Subito Sera** (EESS) organizzatrice di eventi culturali legati alla poesia contemporanea vuole realizzare una applicazione base di dati.

Scopo primario della associazione EESS è quello di organizzare eventi culturali in tutta Italia. Tipicamente, gli eventi sono caratterizzati da uno slogan, si svolgono in qualche città italiana, hanno una certa durata e hanno una quota di partecipazione.

Gli eventi possono essere sponsorizzati da enti di cui si conosce il nome, una breve descrizione della loro attività, l'indirizzo e l'importo versato per la sponsorizzazione. Si noti che uno stesso ente può sponsorizzare più eventi e che l'importo versato può variare.

Requisiti (2)

Gli eventi sono in generale costituiti da una o più esibizioni eseguite da uno o più artisti (per ognuno dei quali si conosce nome e cognome, nome del manager, numero di telefono di contatto) e sono caratterizzati dalla tipologia e dalla durata. Si noti che la stessa esibizione può aver luogo in più eventi.

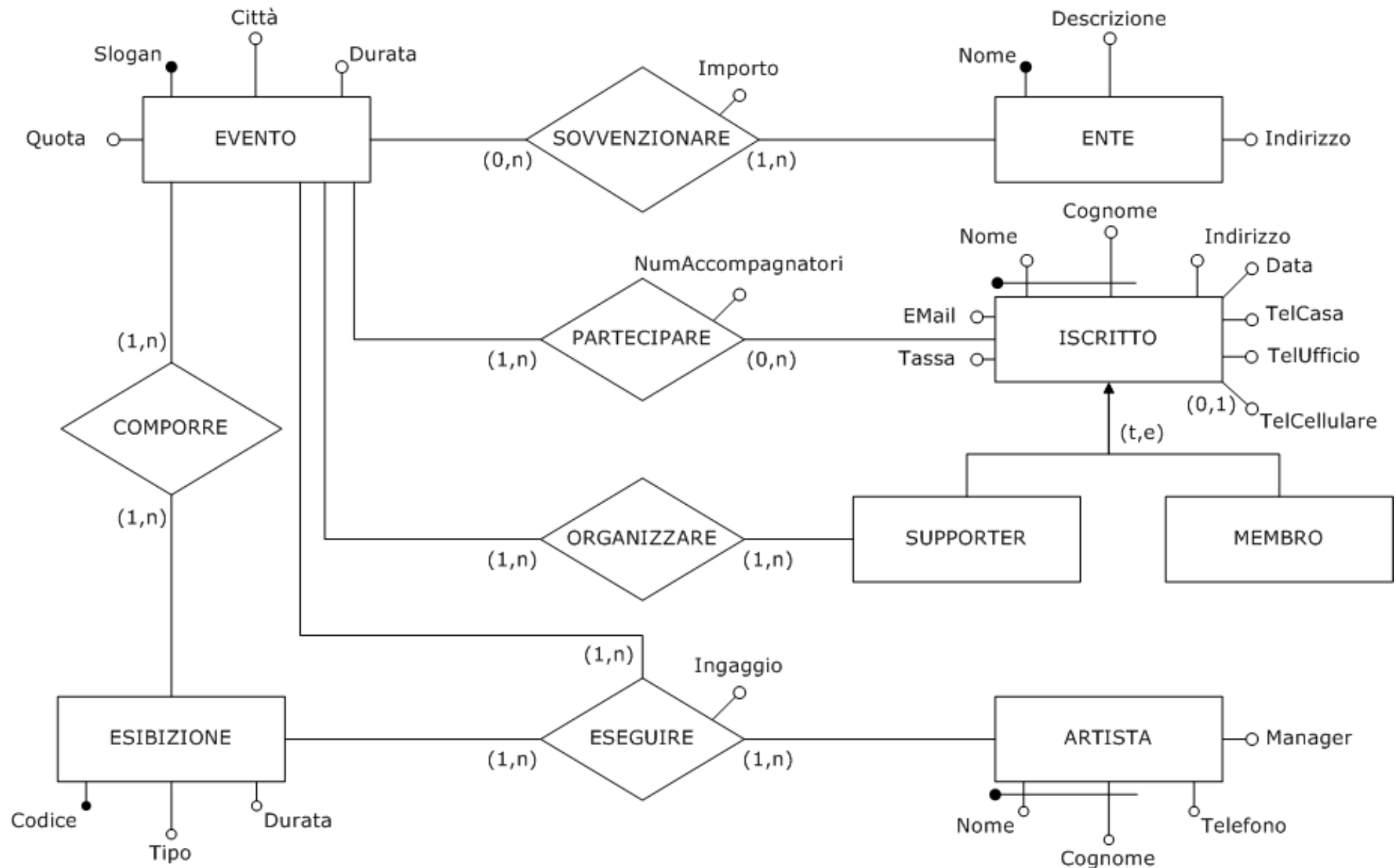
EESS deve inoltre tenere traccia dell'importo versato per l'ingaggio degli artisti, importo che varia a seconda della esibizione a cui l'artista partecipa e dall'evento in cui si svolgerà l'esibizione stessa.

Requisiti (3)

Per gli iscritti alla associazione EESS si conoscono nome, cognome, indirizzo, numeri di telefono (casa, ufficio ed eventualmente cellulare), indirizzo di e-mail, data di iscrizione e tassa di iscrizione.

Gli iscritti si suddividono in due classi: i membri e i supporter. Questi ultimi partecipano alla organizzazione degli eventi che l'associazione pubblicizza. Tutti gli iscritti possono partecipare gratuitamente agli eventi mentre gli eventuali accompagnatori (è necessario tenere traccia solo del numero di persone che accompagnano gli iscritti alla associazione) devono versare una quota di partecipazione.

Schema E-R



Schema logico

Relazione

Attributi

EVENTO	<u>Slogan</u>, Città, Durata, Quota
ESIBIZIONE	<u>Codice</u>, Tipo, Durata
ARTISTA	<u>Nome</u>, <u>Cognome</u>, Manager, Telefono
ENTE	<u>Nome</u>, Descrizione, Indirizzo
ISCRITTO	<u>Nome</u>, <u>Cognome</u>, Indirizzo, Email, TelCasa, TelUfficio, TelCellulare, Data, Tassa, Selettore
COMPORRE	<u>SloganEvento</u>, <u>CodiceEsibizione</u>
ESEGUIRE	<u>SloganEvento</u>, <u>CodiceEsibizione</u>, <u>NomeArtista</u>, <u>CognomeArtista</u>, Ingaggio
SOVVENZIONARE	<u>SloganEvento</u>, <u>NomeEnte</u>, Importo
PARTECIPARE	<u>Nomelscritto</u>, <u>Cognomelscritto</u>, <u>SloganEvento</u>, NumAccompagnatori
ORGANIZZARE	<u>Nomelscritto</u>, <u>Cognomelscritto</u>, <u>SloganEvento</u>

Vincoli di integrità referenziale

Chiave Esterna

Referenzia

COMPORRE.SloganEvento	EVENTO.Slogan
COMPORRE.CodiceEsibizione	ESIBIZIONE.Codice
ESEGUIRE.SloganEvento	EVENTO.Slogan
ESEGUIRE.CodiceEsibizione	ESIBIZIONE.Codice
ESEGUIRE.(NomeArtista,CognomeArtista)	ARTISTA.(Nome, Cognome)
SOVVENZIONARE.SloganEvento	EVENTO.Slogan
SOVVENZIONARE.NomeEnte	ENTE.Nome
PARTECIPARE.(Nomelscritto, Cognomelscritto)	ISCRITTO.(Nome, Cognome)
PARTECIPARE.SloganEvento	EVENTO.Slogan
ORGANIZZARE.(Nomelscritto, Cognomelscritto)	ISCRITTO.(Nome, Cognome)
ORGANIZZARE.SloganEvento	EVENTO.Slogan

Esercitazione 2

Requisiti (1)

La ditta **Sempre Vivo** (SV), specializzata nella realizzazione di manichini, busti vetrina, torsi ed accessori, vuole realizzare una applicazione base di dati.

I prodotti SV sono principalmente manichini ed accessori. Il catalogo dell'azienda riporta, per ciascun prodotto, un codice, il materiale di cui è composto, le dimensioni ed il prezzo. Per gli accessori vengono inoltre riportati i colori disponibili e, nel caso di accessori venduti ma non prodotti da SV, il nome del produttore.

SV è inoltre proprietaria di un certo numero di negozi dislocati su tutto il territorio nazionale. Di ciascun negozio si conosce il nome, le informazioni di contatto (che comprendono il nome del gestore, l'indirizzo ed i numeri di telefono) ed i prodotti venduti.

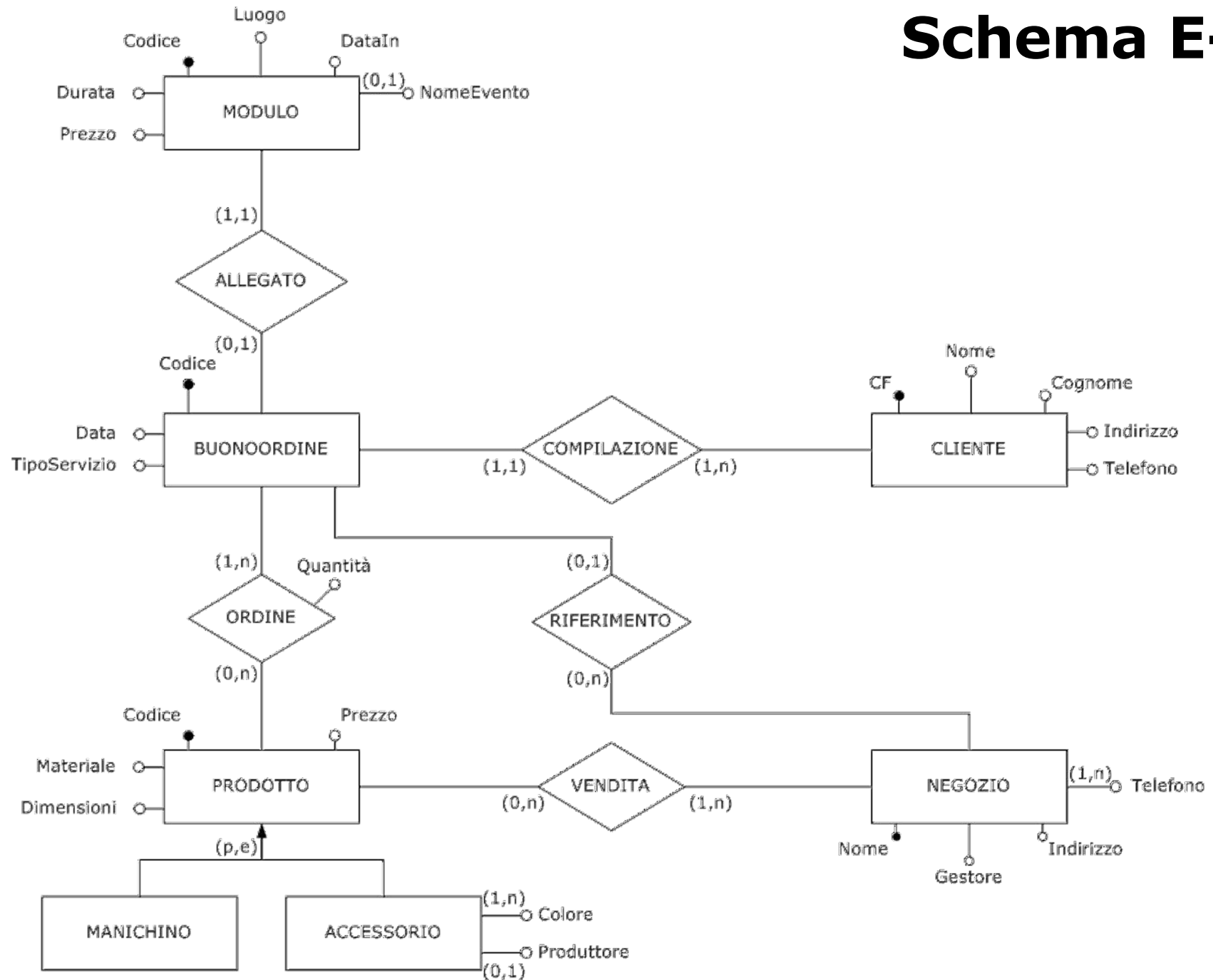
Requisiti (2)

L'azienda oltre a vendere i propri prodotti tramite i negozi fornisce anche direttamente un servizio di noleggio e allestimento per fiere, manifestazioni e vetrine.

I clienti dell'azienda o dei negozi (caratterizzati dalle usuali informazioni anagrafiche) devono compilare un buono d'ordine sul quale viene riportata la data di compilazione, l'eventuale codice del negozio presso cui viene compilato il modulo, il tipo di servizio richiesto ed i prodotti richiesti con la relativa quantità.

Ad ogni buono d'ordine può essere associato un modulo sul quale si indicano il luogo presso il quale devono essere portati i prodotti, la data di inizio del servizio e la sua durata, il prezzo totale del servizio ed eventualmente il nome dell'evento per cui si richiede il servizio.

Schema E-R



Schema logico

Relazione

Attributi

PRODOTTO	<u>Codice</u> , Materiale, Dimensioni, Prezzo, Produttore, Selettore
NEGOZIO	<u>Nome</u> , Gestore, Indirizzo
TELEFONO	<u>NumTel</u> , NomeNegozio
CLIENTE	<u>CF</u> , Nome, Cognome, Indirizzo, Telefono
BUONOORDINE	<u>Codice</u> , Data, TipoServizio, CFCliente, NomeNegozio
MODULO	<u>Codice</u> , Luogo, DataIn, Durata, Prezzo, NomeEvento, CodiceBuono
COLORE	<u>CodiceColore</u> , <u>CodiceProdotto</u>
VENDITA	<u>CodiceProdotto</u> , <u>NomeNegozio</u>
ORDINE	<u>CodiceBuono</u> , <u>CodiceProdotto</u> , Quantità

Vincoli di integrità referenziale

Chiave Esterna

Referenzia

COLORE.CodiceProdotto	PRODOTTO.Codice
TELEFONO.NomeNegozio	NEGOZIO.Nome
VENDITA.CodiceProdotto	PRODOTTO.Codice
VENDITA.NomeNegozio	NEGOZIO.Nome
BUONOORDINE.CFCliente	CLIENTE.CF
BUONOORDINE.NomeNegozio	NEGOZIO.Nome
MODULO.CodiceBuono	BUONOORDINE.Codice
ORDINE.CodiceBuono	BUONOORDINE.Codice
ORDINE.CodiceProdotto	PRODOTTO.Codice

Esercitazione 3

Requisiti (1)

I proprietari della cascina [La Piccola Luna](#) hanno deciso di progettare una base di dati in grado di gestire efficientemente tutte le informazioni relative alla cascina ed allo svolgimento della loro attività.

La cascina dispone di più appezzamenti di terreno ognuno dei quali è suddiviso in più lotti. Ciascun appezzamento è caratterizzato da un nome, dal tipo di terreno e da una posizione geografica. Per i lotti, identificati da un codice univoco nell'ambito del relativo appezzamento, si conosce la dimensione, la data in cui si è effettuata l'ultima semina all'interno del lotto ed il numero medio di mesi che devono trascorrere prima di poter effettuare il raccolto.

Requisiti (2)

Ogni lotto viene sempre utilizzato per la coltivazione di un ben preciso prodotto per il quale è noto il nome scientifico, la lista di trattamenti a cui il prodotto deve essere sottoposto ed il costo (per chilo) delle relative sementi.

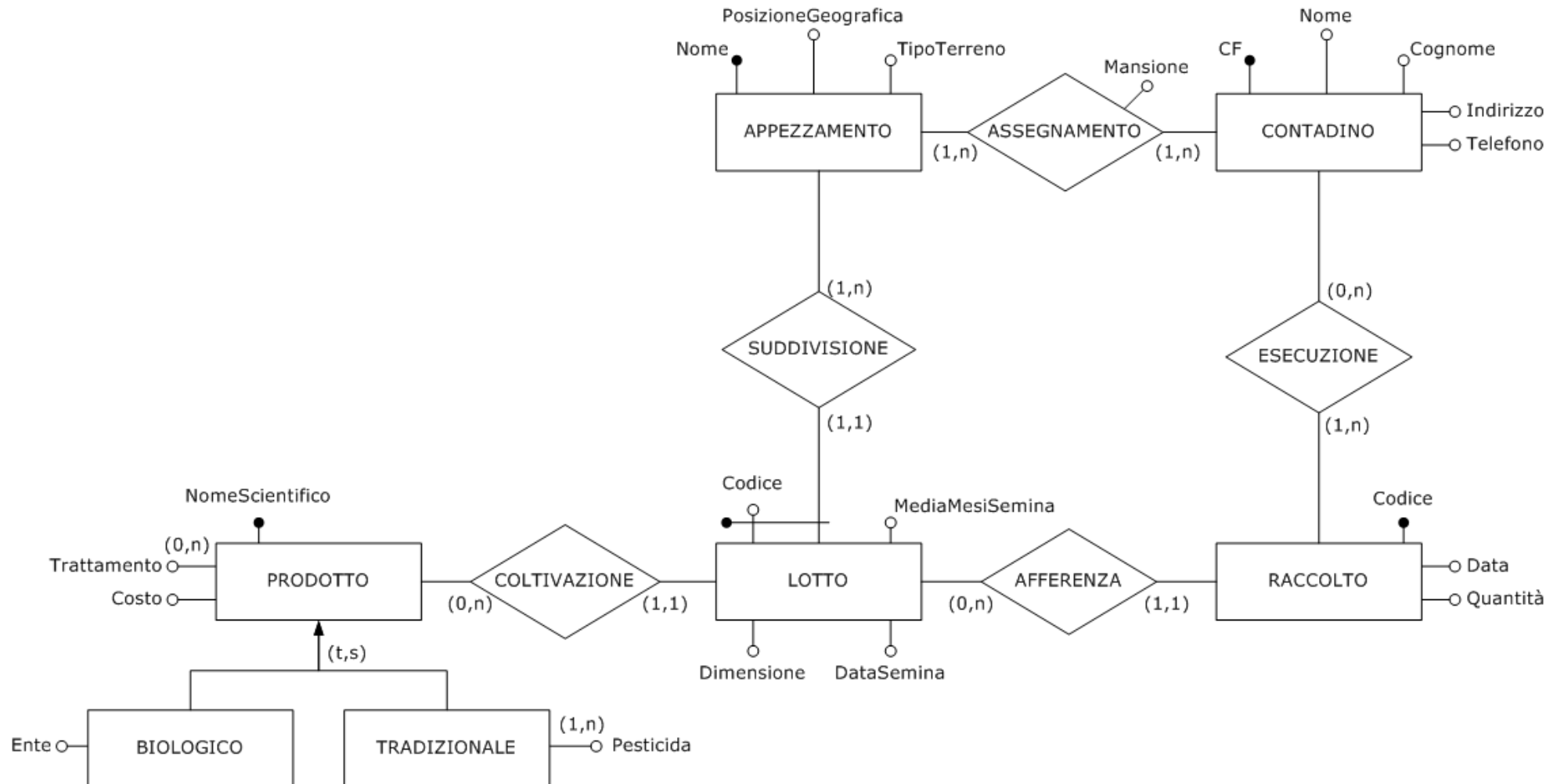
I prodotti coltivati presso la cascina si suddividono in due categorie principali: i prodotti biologici, per i quali non è ammesso l'uso di pesticidi ed i prodotti coltivati con tecniche tradizionali. Si osservi che queste due tipologie non sono mutuamente esclusive (per uno stesso prodotto è possibile la presenza contemporanea di coltivazioni biologiche e non). Per i prodotti biologici si deve tener traccia del nome dell'ente che attesterà la qualità del prodotto, mentre per quelli coltivati con tecniche tradizionali si deve specificare la lista dei pesticidi che possono essere utilizzati.

Requisiti (3)

I contadini che lavorano nella cascina (per i quali si conoscono le usuali informazioni anagrafiche come nome, cognome, indirizzo e telefono) possono essere assegnati ad uno o più appezzamenti di terreno dove possono svolgere mansioni diverse.

Si vogliono inoltre rilevare tutte le informazioni relative ai singoli raccolti. Ogni raccolto è caratterizzato da una data, dalla quantità di prodotto raccolto, dai contadini che hanno effettuato il raccolto e il lotto coinvolto.

Schema E-R



Schema logico

Relazione	Attributi
APPEZZAMENTO	<u>Nome</u> , TipoTerreno, PosizioneGeografica
CONTADINO	<u>CF</u> , Nome, Cognome, Indirizzo, Telefono
LOTTO	<u>NomeAppezzamento</u> , <u>Codice</u> , Dimensione, DataSemina, MediaMesiSemina, NomeProdotto
PRODOTTO	<u>NomeScientifico</u> , Costo, Ente, Biologico,Tradizionale
RACCOLTO	<u>Codice</u> , Data, Quantità, NomeAppezzamento, CodiceLotto
ASSEGNAZIONE	<u>NomeAppezzamento</u> , <u>CFContadino</u> , Mansione
ESECUZIONE	<u>CFContadino</u> , <u>CodiceRaccolto</u>
TRATTAMENTO	<u>NomeProdotto</u> , <u>CodiceTrattamento</u>
PESTICIDA	<u>NomeProdotto</u> , <u>CodicePesticida</u>

Vincoli di integrità referenziale

Chiave Esterna

Referenzia

LOTTO.NomeAppezzamento	APPEZZAMENTO.Nome
LOTTO.NomeProdotto	PRODOTTO.NomeScientifico
RACCOLTO.(NomeAppezzamento, CodiceLotto)	LOTTO.(NomeAppezzamento, Codice)
ASSEGNAAMENTO.NomeAppezzamento	APPEZZAMENTO.Nome
ASSEGNAAMENTO.CFContadino	CONTADINO.CF
ESECUZIONE.CFContadino	CONTADINO.CF
ESECUZIONE.CodiceRaccolto	RACCOLTO.Codice
TRATTAMENTO.NomeProdotto	PRODOTTO.NomeScientifico
PESTICIDA.NomeProdotto	PRODOTTO.NomeScientifico

A series of thin, light brown lines forming an abstract, overlapping geometric pattern in the top-left corner of the page. The lines intersect to create various polygonal shapes, some of which are partially cut off by the edge of the frame.

VINCENZO CALABRÒ

LinkedIn [vincenzocalabro](#)

www.vincenzocalabro.it