SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2 (S.B.3.1.2)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli

Versione 2016-05-14

Indice

Queste slide sono composte dalle seguenti sottounità:

Indice (2)

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.1 (S.B.3.1.2.1)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Introduzione

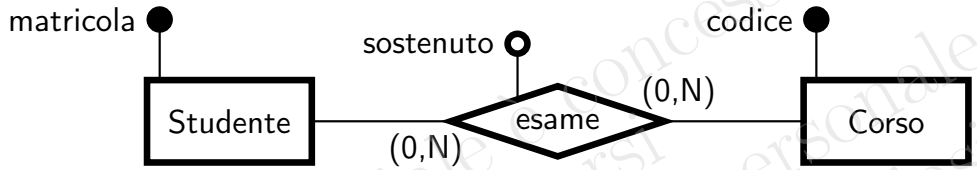
Versione 2016-05-14

Produzione dello schema relazionale da ER

Obiettivo: generare lo schema relazionale (con vincoli) per la base dati.

- Input: diagramma ER concettuale con entità, relationship, attributi, is-a, generalizzazioni, vincoli di identificazione, vincoli esterni
- Output: un insieme di relazioni con vincoli di integrità.

Esempio (un caso molto semplice)



Entità Studente

attributo	dominio
matricola	intero

Relationship esame

attributo	dominio
sostenuto	data

Entità Corso

attributo	dominio
codice	intero

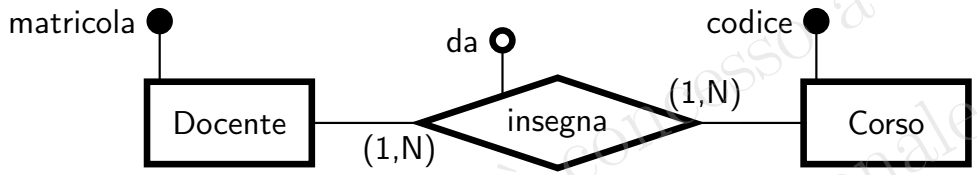
- Definizione delle seguenti relazioni con vincoli:

  Studente (matricola:integer)

  Corso (codice:integer)

  esame (studente:integer, corso:integer, sostenuto:date)
  Vincolo foreign key: studente references Studente(matricola)
  Vincolo foreign key: corso references Corso(codice)

Esempio (un caso leggermente più complesso)



Entità Docente

attributo	dominio
matricola	intero

Relationship insegna

attributo	dominio
da	data

Entità Corso

attributo	dominio
codice	intero

- Definizione delle seguenti relazioni con vincoli:

  Docente (matricola:integer)
  Vincolo inclusione: matricola $\subseteq$ insegna(docente)

  Corso (codice:integer)
  Vincolo inclusione: codice $\subseteq$ insegna(corso)

  insegna (docente:integer, corso:integer, da:date)
  Vincolo foreign key: docente references Docente(matricola)
  Vincolo foreign key: corso references Corso(codice)

Una Metodologia Robusta e Scalabile

Il linguaggio ER offre molti costrutti e i diagrammi possono essere complicati.

- Come gestire attributi multivalore?
- Come gestire relazioni is-a?
- Come gestire generalizzazioni complete e non complete?
- ...

Scrivere direttamente lo schema relazionale a partire dall'ER concettuale comporta alto rischio di errore...

Una Metodologia Robusta e Scalabile (2)

. . . Scrivere direttamente lo schema relazionale a partire dall'ER concettuale comporta alto rischio di errore.

Una metodologia robusta e scalabile:

1. Ristrutturazione dell'ER : a partire dall'ER concettuale produciamo un nuovo diagramma ER (ristrutturato). Il nuovo diagramma:
   - è (sostanzialmente) equivalente all'originale: i modelli della formula FOL relativa all'ER originale sono in corrispondenza 1–1 con i modelli della formula FOL relativa all'ER ristrutturato
   - contiene solo i costrutti più semplici: entità, relationship, attributi di domini SQL base e con molteplicità massima pari ad 1
   - tiene conto di alcuni requisiti di performance
2. Traduzione diretta: produciamo lo schema relazionale della base dati a partire dall'ER ristrutturato e tenendo conto di ulteriori requisiti di performance.

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2 (S.B.3.1.2.2)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER

Versione 2016-05-14

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.1 (S.B.3.1.2.2.1)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Attributi Multivalore

Versione 2016-05-14

Eliminazione di Attributi Multivalore

Obiettivo

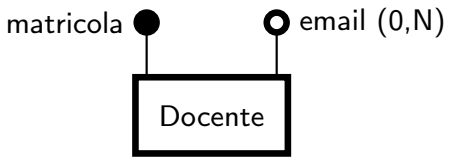
Ristrutturare il diagramma ER in uno (sostanzialmente) equivalente che non contenga attributi multivalore (che non sono supportati dai DBMS).

Equivalenza sostanziale: le formule FOL dei due diagrammi ER possono essere definite su vocabolari diversi, quindi non possiamo parlare di "equivalenza logica".

Due formule FOL sono sostantialmente equivalenti se esiste una funzione biunivoca (corrispondenza 1-1) tra i modelli dell'una e quelli dell'altra.

Eliminazione di Attributi Multivalore di Entità

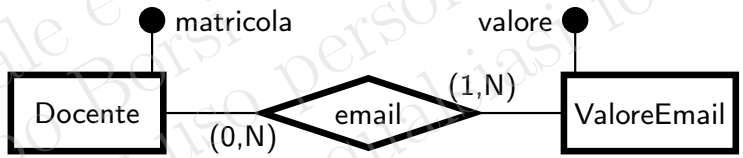
Ristrutturiamo il diagramma definendo una nuova entità e una nuova relationship per rappresentare i singoli valori dell'attributo e legarli alle istanze dell'entità originaria che hanno quei valori per l'attributo.



Entità Docente

attributo	dominio
matricola	intero
email	stringa

$\Longrightarrow$



Entità Docente

attributo	dominio
matricola	intero

Entità ValoreEmail

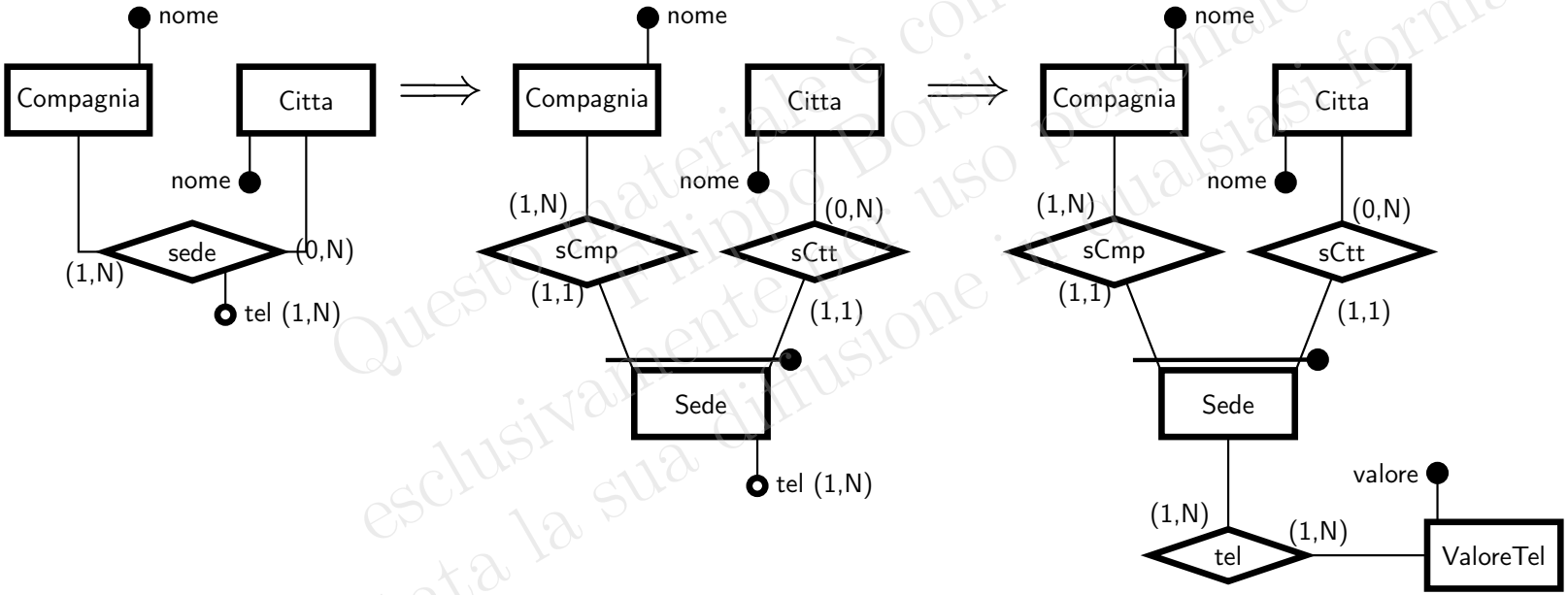
attributo	dominio
valore	stringa

Si notino i vincoli di cardinalità della relationship email:

- (0,N) per il ruolo Docente: lo stesso vincolo dell'attributo multivalore
- (1,N) per il ruolo ValoreEmail: per mantenere l'equivalenza tra i diagrammi, ogni istanza della nuova entità ValoreEmail dovrà essere coinvolta in almeno un'istanza della relationship

Eliminazione di Attributi Multivalore di Relationship

Stessa idea, ma prima dobbiamo trasformare la relationship in entità, mantenendone la semantica.

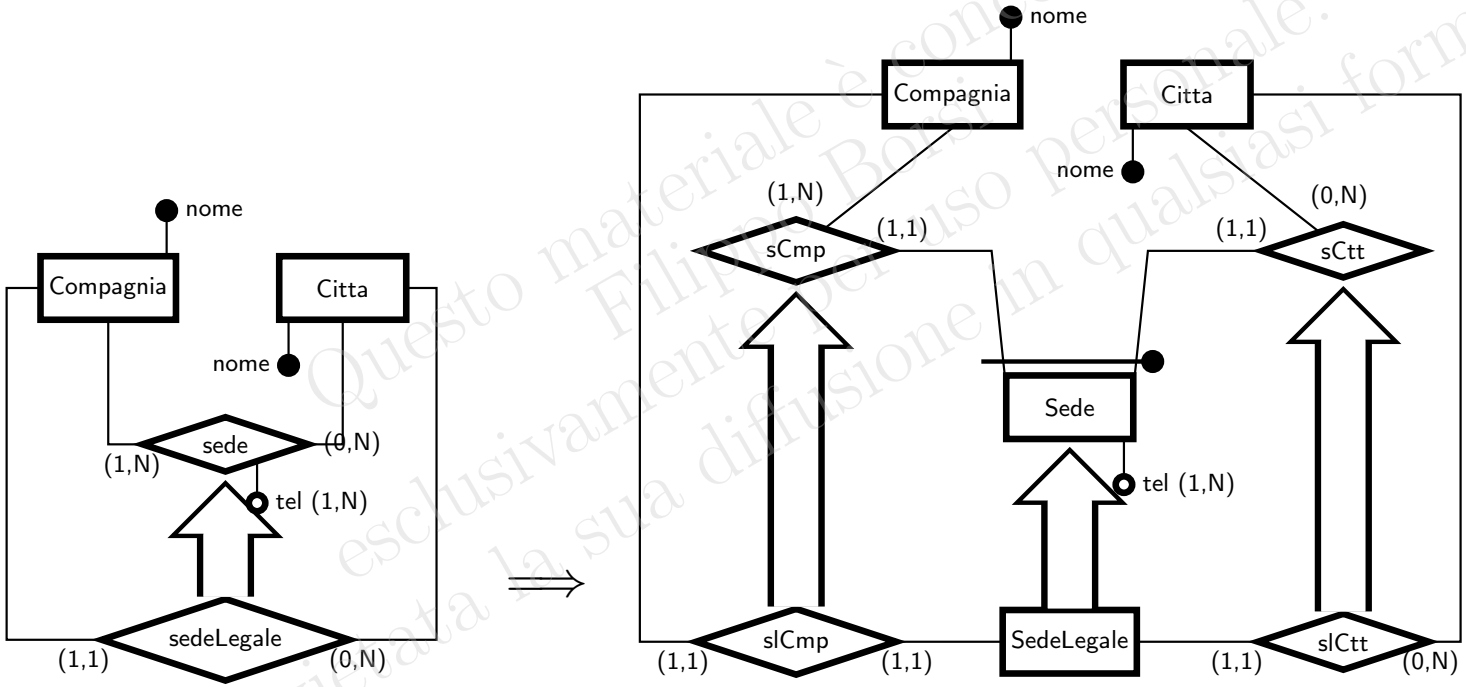


La relationship Sede è stata trasformata in entità mantenendone la semantica, grazie al nuovo vincolo di identificazione.

La trasformazione di relationship in entità può comportare ristrutturazioni a cascata se la relationship è coinvolta in relazioni is-a...

Eliminazione di Attributi Multivalore di Relationship (2)

. . . La trasformazione di relationship in entità può comportare ristrutturazioni a cascata se la relationship è coinvolta in relazioni is-a:



si prosegue come al caso precedente,
creando l'entità ValoreTel

Eliminazione di Attributi Multivalore

Al termine del passo di eliminazione degli attributi multivalore

tutti gli attributi del diagramma ristrutturato hanno

vincoli di cardinalità (0,1) oppure (1,1)

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.2 (S.B.3.1.2.2.2)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Domini

Versione 2016-05-14

Corrispondenza dei Domini Concettuali in Domini Supportati dal DBMS

Obiettivo

Ristrutturare il diagramma ER in uno (sostanzialmente) equivalente che contenga solo attributi di domini supportati dal DBMS.

Corrispondenza dei Domini Concettuali in Domini Supportati dal DBMS (2)

Nello schema concettuale abbiamo utilizzato diversi domini:

- domini base: intero, reale, stringa, data, ora, dataora, booleano, etc.
- domini base specializzati: intero > 0, $[x, y]$ (intervallo di interi), etc.
- domini enumerativi: {M,F}, {Africa, America, Asia, Europa, Oceania}, etc.
- domini di tipo record definiti dall'analista: Indirizzo, etc.

Dobbiamo decidere come rappresentare i rispettivi valori nella base dati.

Nota: grazie al passo di eliminazione di attributi multivalore, le molteplicità di tutti gli attributi sono (0,1) oppure (1,1).

Domini base

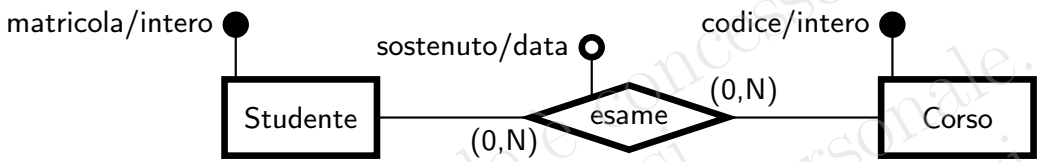
I domini base di solito hanno un corrispondente esatto in SQL
$\Longrightarrow$ usiamo quello (cfr. manuale SQL e del DBMS in uso).

dominio concettuale	dominio SQL
intero	integer, smallint, etc.
reale	real, decimal, numeric, float, etc.
data	date, etc.
ora	time, etc.
dataora	timestamp, etc.

Ristrutturiamo il diagramma ER e le specifiche dei dati sostituendo ad ogni dominio concettuale di tipo base il corrispondente dominio SQL che abbiamo scelto.

Esempio

Diagramma ER concettuale



Entità Studente

attributo	dominio
matricola	intero

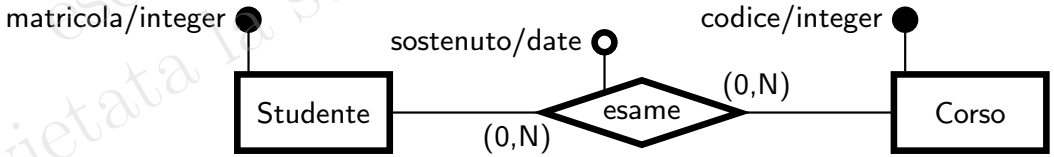
Relationship esame

attributo	dominio
sostenuto	data

Entità Corso

attributo	dominio
codice	intero

Diagramma ER ristrutturato



Entità Studente

attributo	dominio
matricola	integer

Relationship esame

attributo	dominio
sostenuto	date

Entità Corso

attributo	dominio
codice	integer

Domini Specializzati

Il dominio SQL più vicino ad un dominio base specializzato di solito è più esteso, e permette di rappresentare valori che non dovrebbero essere ammessi.

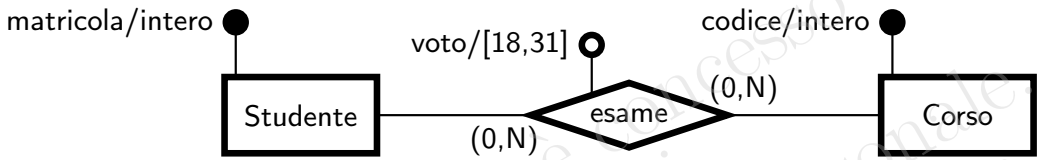
Definiamo un dominio utente per ognuno di essi: in SQL questo può essere fatto con il comando **create domain**.

dominio concettuale	comando SQL per definire dominio utente
intero > 0	`create domain InteroPos as integer check (value > 0)`
reale ≤ 0	`create domain RealeNonPos as real check (value <= 0)`
$[0, 10]$	`create domain Intero_0_10 as integer check (value >= 0 and value <= 10)`

Ristrutturiamo il diagramma ER e le specifiche dei dati sostituendo ad ogni dominio concettuale di tipo specializzato il corrispondente dominio SQL che abbiamo definito.

Esempio

Diagramma ER concettuale



Entità Studente

attributo	dominio
matricola	intero

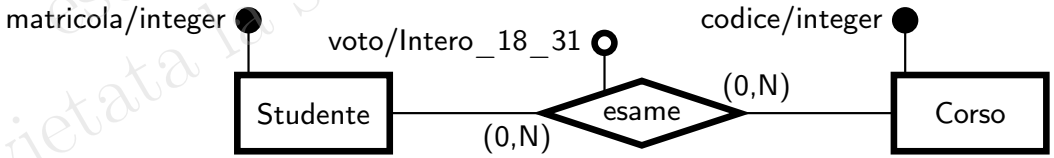
Relationship esame

attributo	dominio
voto	[18,31]

Entità Corso

attributo	dominio
codice	intero

Diagramma ER ristrutturato



Entità Studente

attributo	dominio
matricola	integer

Relationship esame

attributo	dominio
voto	Intero_18_31

Entità Corso

attributo	dominio
codice	integer

Domini Enumerativi: Approccio 1

I domini enumerativi vengono fatti corrispondere a domini utente SQL di tipo enum.

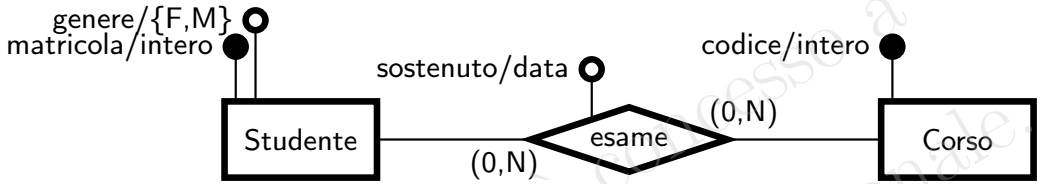
dominio concettuale	comando SQL per definire dominio utente
{M,F}	`create type Sesso as enum ('F', 'M')`
{ Africa, America, ...}	`create type Continente as enum ('Africa', 'America', ...)`

Attenzione: alcuni DBMS potrebbero offrire costrutti non standard, anche con limitazioni.

Ristrutturiamo il diagramma ER e le specifiche dei dati sostituendo ad ogni dominio concettuale di tipo specializzato il corrispondente dominio SQL che abbiamo definito.

Esempio

Diagramma ER concettuale



Entità Studente

attributo	dominio
matricola	intero
genere	{F,M}

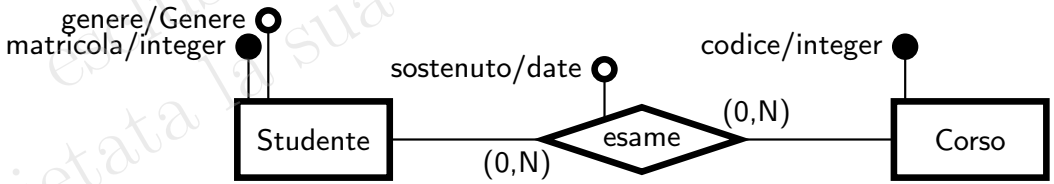
Relationship esame

attributo	dominio
sostenuto	data

Entità Corso

attributo	dominio
codice	intero

Diagramma ER ristrutturato



Entità Studente

attributo	dominio
matricola	integer
genere	Genere

Relationship esame

attributo	dominio
sostenuto	date

Entità Corso

attributo	dominio
codice	integer

Domini Enumerativi: Approccio 2

Cosa fare quando il DBMS in uso non supporta i domini enumerativi?

Possiamo ristrutturare il diagramma ER e definire il dominio come una nuova entità. Le istanze di questa entità sono vincolate ad essere solo quelle che rappresentano i valori legali del dominio.

Vincolo:

$$\forall c, n \ \text{Continente}(c) \wedge \text{nome}(c, n) \ \rightarrow$$
$$c = \text{'Af'} \vee c = \text{'Am'} \vee \cdots \vee c = \text{'O'}$$

Approccio 2 è vantaggioso anche quando il numero di valori del dominio è particolarmente alto.

Domini Composti: Approccio 1

I domini composti sono più problematici.

SQL:1999 permette all'utente di definire tipi strutturati con il costrutto **create** type, ma questa funzionalità non è ad oggi supportata da tutti i DBMS commerciali.

Esempio: dominio Indirizzo di tipo record con i campi:

- via : stringa
- civico : intero
- citta : stringa

```
create type Indirizzo as (
    via varchar(100), civico integer,
    citta varchar(100)
)
```

Attenzione: molti DBMS offrono costrutti non standard e/o con limitazioni.

Domini Composti: Approccio 1 (2)

Molti DBMS offrono costrutti non standard e/o con limitazioni.

Ad esempio, PostgreSQL non permette di usare clausole **check** in **create type**.

In caso di necessità, una possibilità è definire i domini dei campi con vincoli usando **create domain**.

Esempio: dominio Indirizzo di tipo record con i campi:

- via : stringa
- civico : intero > 0
- citta : stringa

```
create domain InteroPos as integer check (value > 0);
create type Indirizzo as (
  via varchar(100), civico InteroPos,
  citta varchar(100)
);
```

Ristrutturiamo il diagramma ER e le specifiche dei dati sostituendo ad ogni dominio concettuale composto il corrispondente dominio SQL che abbiamo definito.

Esempio

Diagramma ER concettuale

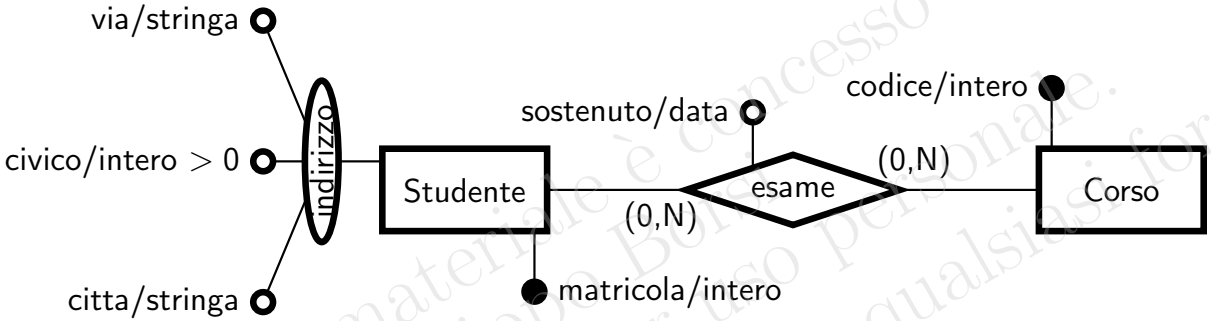
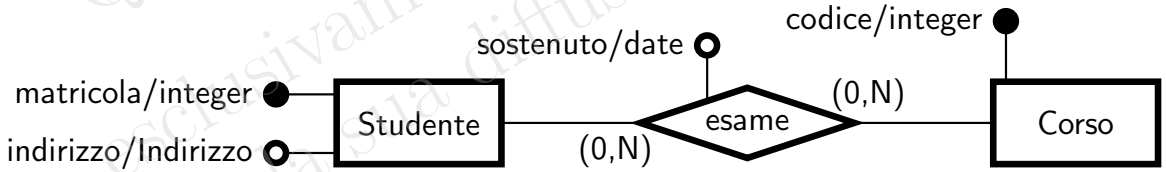


Diagramma ER ristrutturato



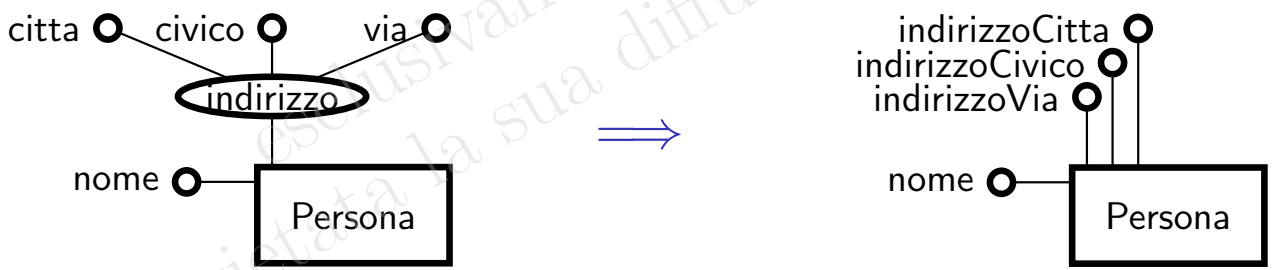
Entità Studente

attributo	dominio
matricola	integer
indirizzo	Indirizzo

Domini Composti: Approccio 2

Cosa fare quando non riusciamo a definire un tipo composto mediante **create** type?

Possiamo ristrutturare il diagramma ER e ridefinire ogni attributo che utilizza il dominio composto come un insieme di attributi di domini semplici, uno per ogni campo del dominio.

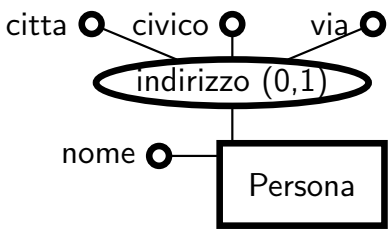


Bisogna però essere attenti ad eventuali vincoli...

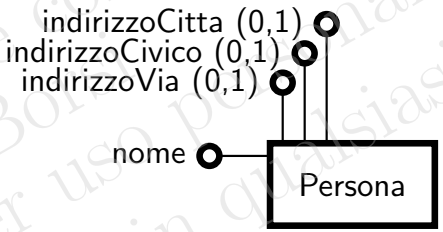
Domini Composti: Approccio 2 (2)

. . . Bisogna però essere attenti ad eventuali vincoli.

Esempio: l'attributo indirizzo è opzionale



$\Longrightarrow$



Vincolo:
per ogni istanza dell'entità Persona: o tutti gli attributi 'indirizzo. . . ' sono definiti o sono tutti non definiti:

$$\forall p\ \text{Persona}(p) \rightarrow$$

$$\left[\begin{array}{l} \exists v, c, t\ \text{indirizzoVia}(p, v) \wedge \\ \qquad \text{indirizzoCivico}(p, c) \wedge \text{indirizzoCitta}(p, t) \end{array}\right]$$

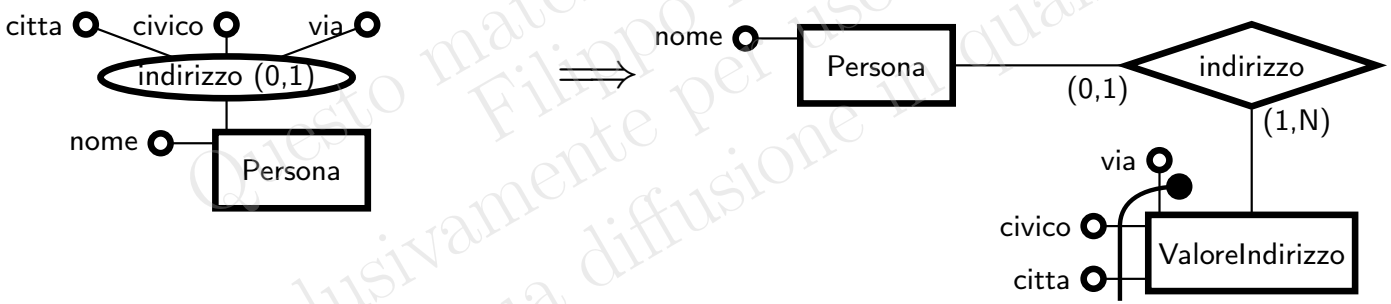
$$\vee$$

$$\left[\begin{array}{l} \forall v, c, t\ \neg\text{indirizzoVia}(p, v) \wedge \\ \qquad \neg\text{indirizzoCivico}(p, c) \wedge \neg\text{indirizzoCitta}(p, t) \end{array}\right]$$

Domini Composti: Approccio 3

Cosa fare quando non riusciamo a definire un tipo composto mediante **create** type?

Possiamo ristrutturare il diagramma ER e definire una nuova entità. Le istanze di questa entità rappresentano i valori del dominio.



Questo approccio può essere particolarmente vantaggioso quando **create** type non è utilizzabile e il vincolo di cardinalità dell'attributo è (0,1) (e non (1,1)), in quanto evita i vincoli esterni richiesti dall'Approccio 2 dovuti all'opzionalità.

Se l'attributo appartiene ad una relationship, dobbiamo prima trasformarla in entità (con possibile necessità di ristrutturazioni a cascata se la relationship è coinvolta in relazioni is-a).

Vantaggi e Svantaggi

Principali vantaggi e svantaggi di diversi approcci:

Alternativa	Vantaggi	Svantaggi
1. Creaz. tipo strutturato	▶ Realizzazione molto vicina all'ER	▶ Spesso non supportato pienamente ▶ Non utilizzabile per tipi complessi (ad es. con vincoli)
2. Sostituzione con attributi semplici	▶ Pienamente portabile	▶ Perdita di struttura ▶ Necessità di vincoli complessi per gestire opzionalità
3. Sostituzione con entità	▶ Pienamente portabile ▶ Vantaggiosa se il vincolo è (0,1) (e non (1,1)) e l'alternativa 1 non è applicabile	▶ Parziale perdita di struttura

Corrispondenza dei Domini Concettuali in Domini Supportati dal DBMS

Al termine del passo di definizione della corrispondenza dei domini concettuali in domini supportati dal DBMS

tutti gli attributi del diagramma ristrutturato sono semplici, hanno un dominio supportato dal DBMS

ed hanno vincoli di cardinalità (0,1) oppure (1,1)

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.3 (S.B.3.1.2.2.3)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Relazioni IS-A e Generalizzazioni tra Entità

Versione 2016-05-14

Eliminazione delle Relazioni IS-A tra Entità e delle Generalizzazioni

Obiettivo

Ristrutturare il diagramma ER in uno (sostanzialmente) equivalente che non contenga relazioni is-a tra entità né generalizzazioni (che non sono supportate dal DBMS).

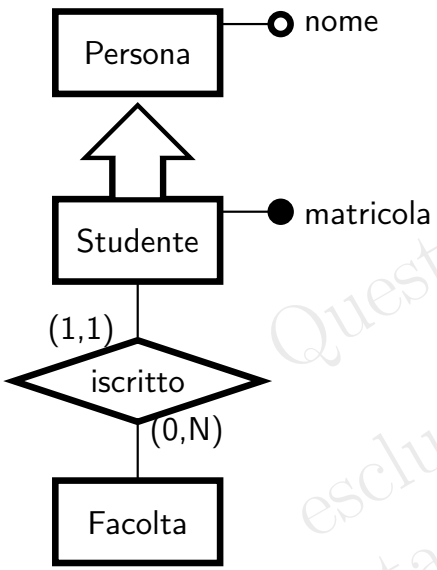
Eliminazione delle Relazioni IS-A tra Entità e delle Generalizzazioni (2)

Tre possibili approcci:

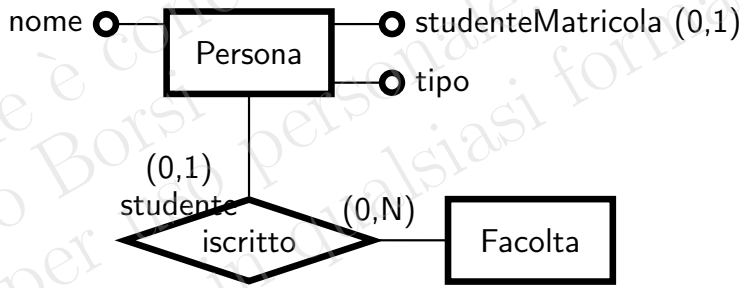
1. Fusione di tutte le entità coinvolte in una sola entità
2. Divisione delle entità coinvolte in entità disgiunte
3. Sostituzione delle relazioni is-a e delle generalizzazioni con relationship.

In generalizzazioni complesse può essere utile combinare i tre approcci.

Fusione delle entità coinvolte in una sola entità



$\Longrightarrow$



Nuovo attributo tipo : $\{Studente, Persona\}$

Vincoli:

$\forall p$ Persona(p) $\rightarrow$

$[\text{tipo}(p, Studente) \leftrightarrow \exists m\ \text{studenteMatricola}(p, m)] \wedge$

$[\text{tipo}(p, Studente) \leftrightarrow \exists f\ \text{iscritto}(f, p)]$

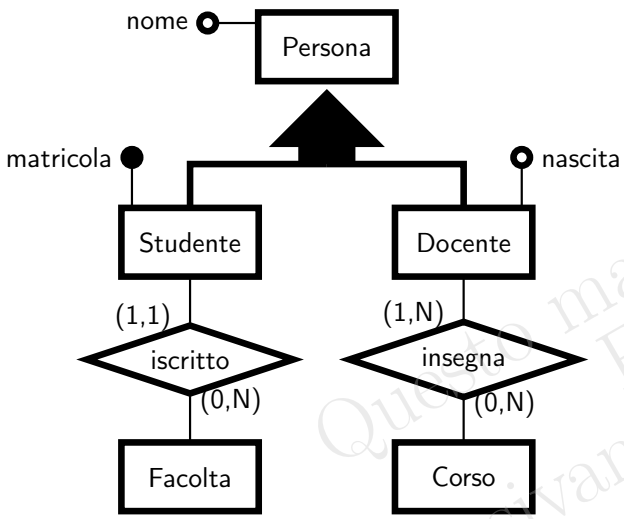
$\wedge$

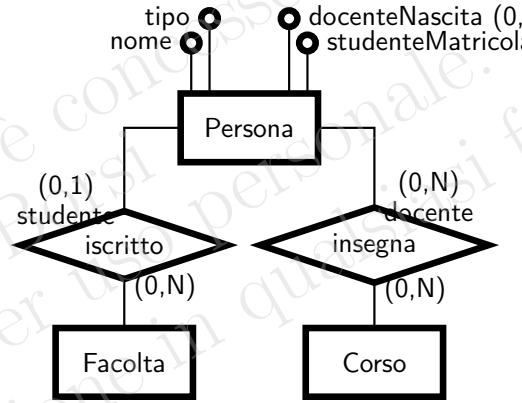
$\forall p, p', m$ Persona$(p) \wedge$ Persona$(p') \wedge$

studenteMatricola$(p, m) \wedge$ studenteMatricola(p', m) $\rightarrow$

$p = p'$

Fusione delle entità coinvolte in una sola entità (2)



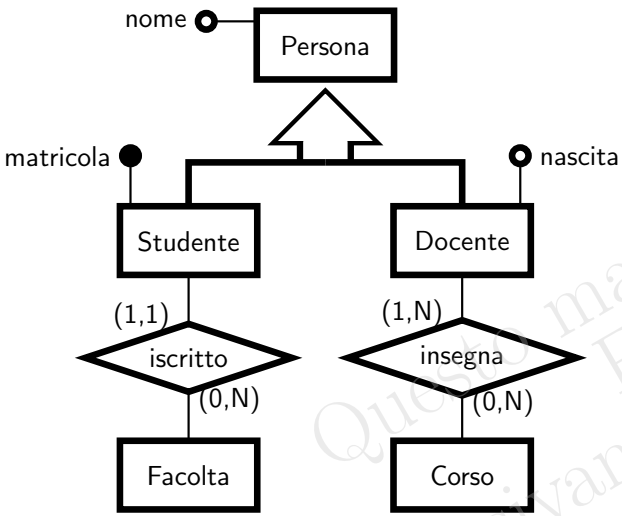


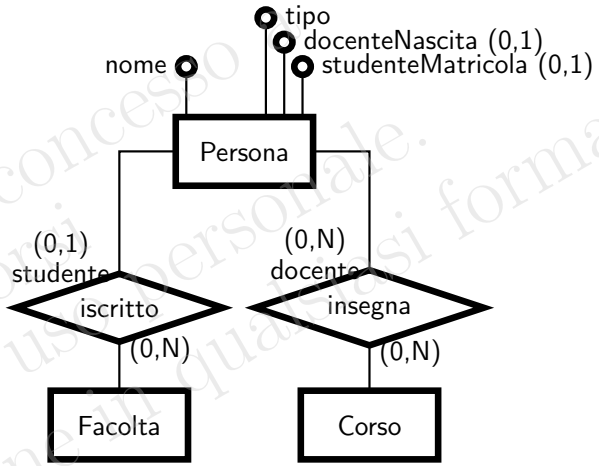
Nuovo attributo tipo : $\{Studente, Docente\}$

Vincoli:

$$\forall p\ \text{Persona}(p) \rightarrow$$
$$[\text{tipo}(p, Studente) \leftrightarrow \exists m\ \text{studenteMatricola}(p, m)] \wedge$$
$$[\text{tipo}(p, Studente) \leftrightarrow \exists f\ \text{iscritto}(f, p)] \wedge$$
$$[\text{tipo}(p, Docente) \leftrightarrow \exists n\ \text{docenteNascita}(p, n)] \wedge$$
$$[\text{tipo}(p, Docente) \leftrightarrow \exists c\ \text{insegna}(c, p)]$$
$$\wedge$$
$$\forall p, p', m\ [\text{Persona}(p) \wedge \text{Persona}(p') \wedge \text{studenteMatricola}(p, m) \wedge \text{studenteMatricola}(p', m)] \rightarrow p = p'$$

Fusione delle entità coinvolte in una sola entità (3)



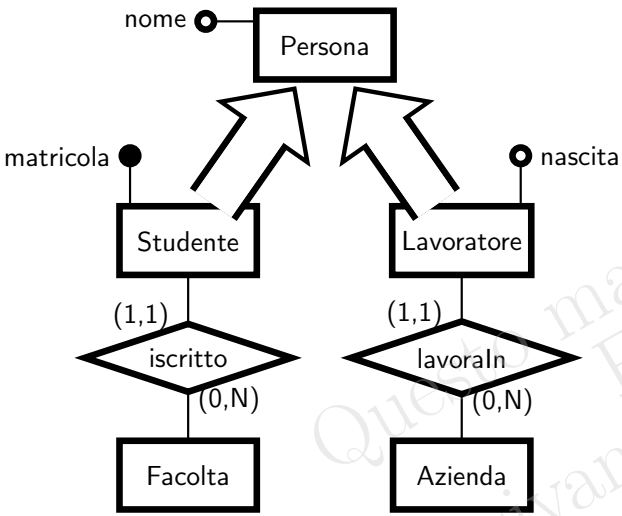


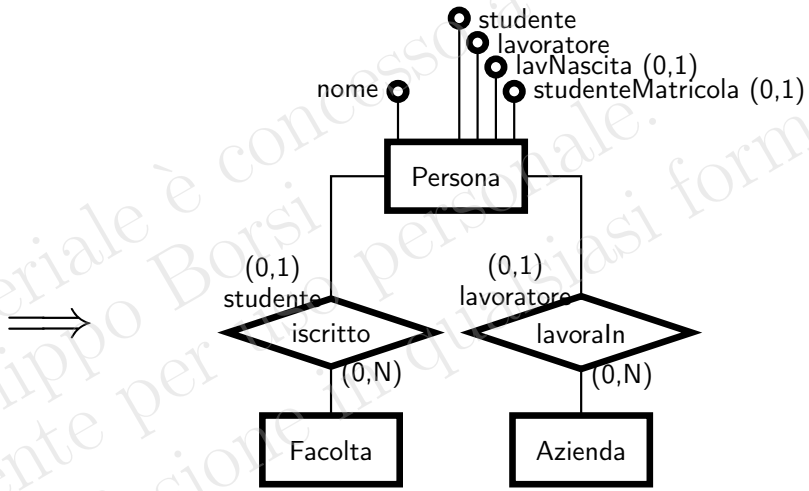
Attributo tipo : $\{Persona, Studente, Docente\}$

Vincoli:

$$\forall p\ \text{Persona}(p) \rightarrow$$
$$[\text{tipo}(p, Studente) \leftrightarrow \exists m\ \text{studenteMatricola}(p, m)] \wedge$$
$$[\text{tipo}(p, Studente) \leftrightarrow \exists f\ \text{iscritto}(f, p)] \wedge$$
$$[\text{tipo}(p, Docente) \leftrightarrow \exists n\ \text{docenteNascita}(p, n)] \wedge$$
$$[\text{tipo}(p, Docente) \leftrightarrow \exists c\ \text{insegna}(c, p)]$$
$$\wedge$$
$$\forall p, p', m\ [\text{Persona}(p) \wedge \text{Persona}(p') \wedge \text{studenteMatricola}(p, m) \wedge \text{studenteMatricola}(p', m)] \rightarrow p = p'$$

Fusione delle entità coinvolte in una sola entità (4)



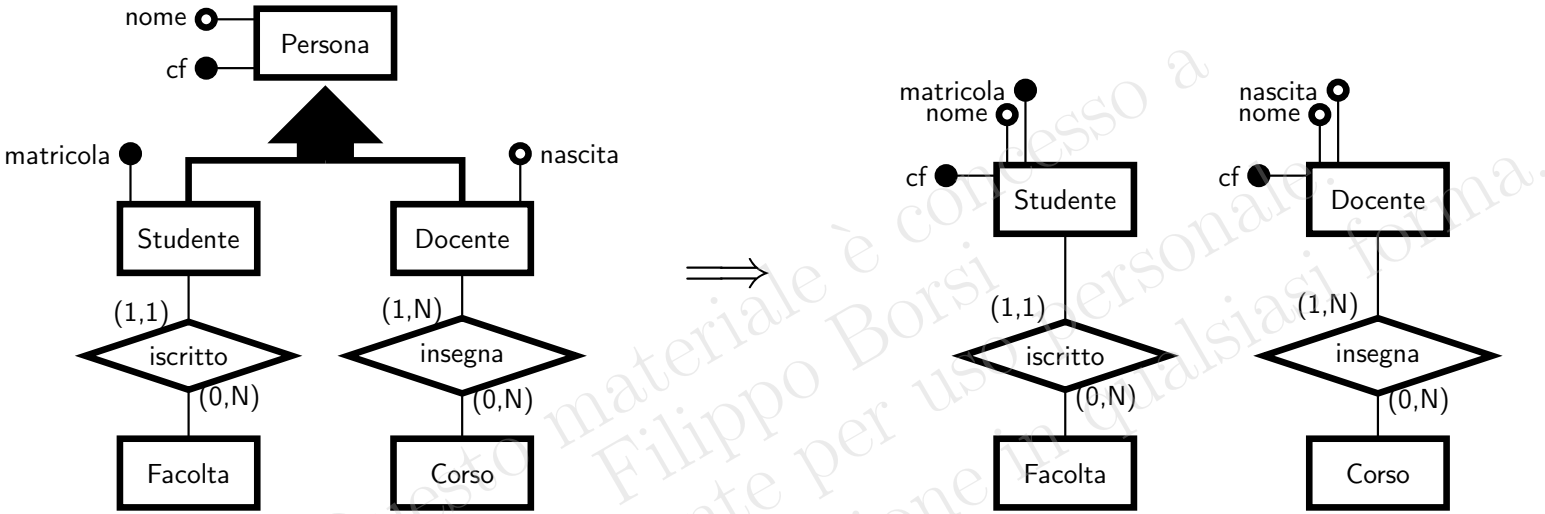


Nuovi attributi booleani: studente e lavoratore

Vincoli:

$$\forall p\ \text{Persona}(p) \rightarrow$$
$$[\text{studente}(p, true) \leftrightarrow \exists m\ \text{studenteMatricola}(p, m)] \wedge$$
$$[\text{studente}(p, true) \leftrightarrow \exists f\ \text{iscritto}(f, p)] \wedge$$
$$[\text{lavoratore}(p, true) \leftrightarrow \exists n\ \text{lavNascita}(p, n)] \wedge$$
$$[\text{lavoratore}(p, true) \leftrightarrow \exists c\ \text{lavoraIn}(c, p)]$$
$$\wedge$$
$$\forall p, p', m\ [\text{Persona}(p) \wedge \text{Persona}(p') \wedge \text{studenteMatricola}(p, m) \wedge \text{studenteMatricola}(p', m)] \rightarrow p = p'$$

Divisione delle entità in entità disgiunte



Attributi dell'entità base nome e cf sono ridefiniti per ogni entità figlia

Vincolo:

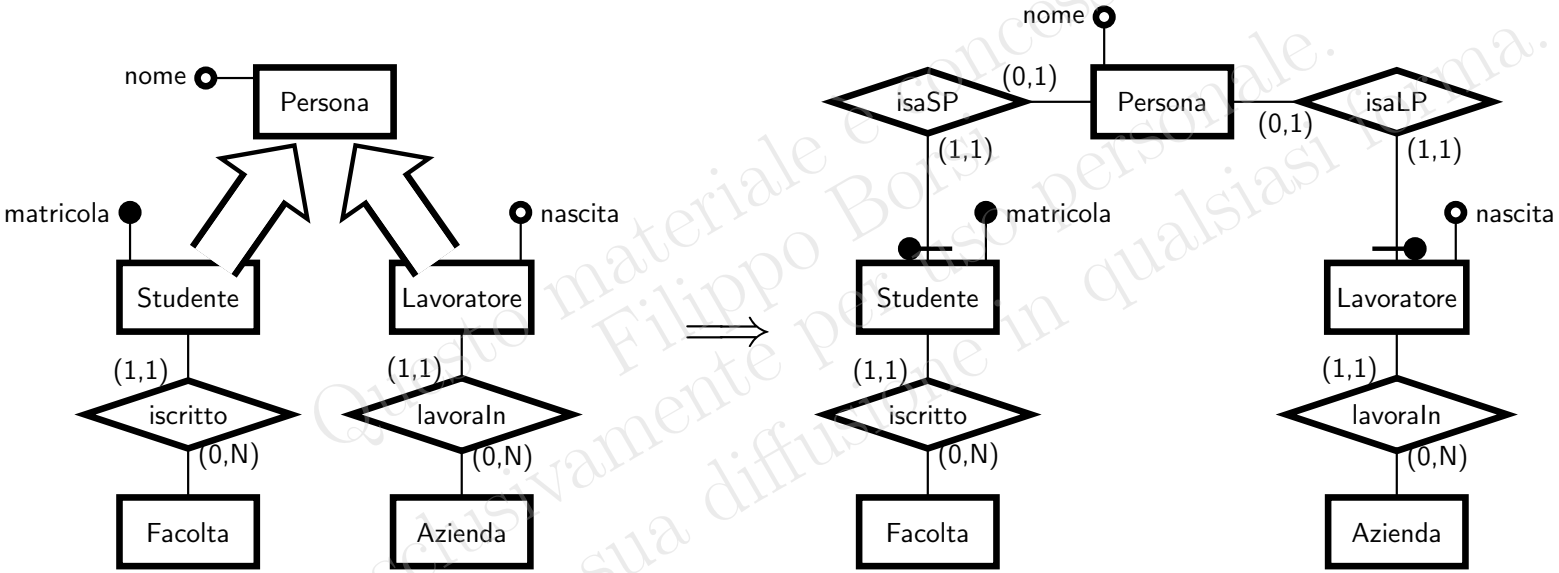
$\forall s, d, cf_s, cf_d$

$$\text{Studente}(s) \wedge \text{Docente}(d) \wedge \text{cf}(s, cf_s) \wedge \text{cf}(d, cf_d) \rightarrow$$
$$cf_s \neq cf_d$$

L'approccio è applicabile solo a generalizzazioni complete

. . . ma è possibile ristrutturare le altre per renderle complete

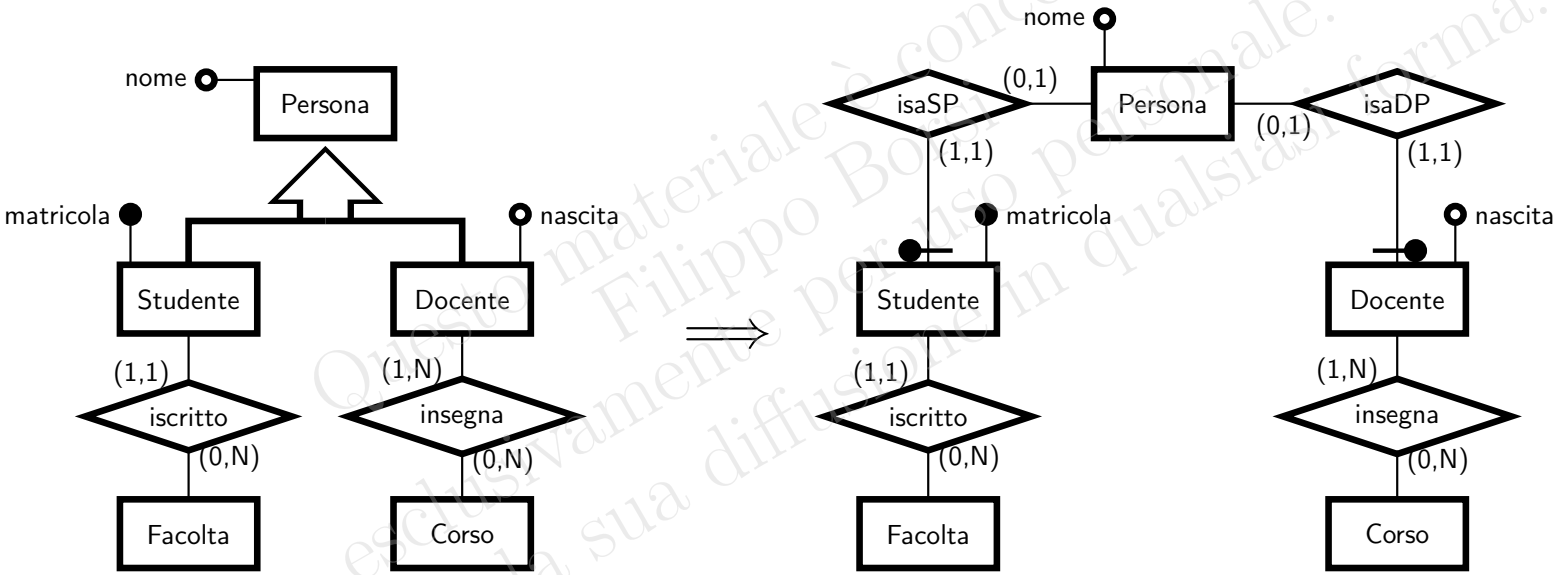
Sostituzione di is-a con relationship



Partizionamento delle istanze: ogni istanza dell'entità Studente (risp. Lavoratore) dell'ER concettuale viene partizionata in due istanze dell'ER ristrutturato:

- una istanza dell'entità Persona
- una istanza dell'entità Studente (risp. Lavoratore)

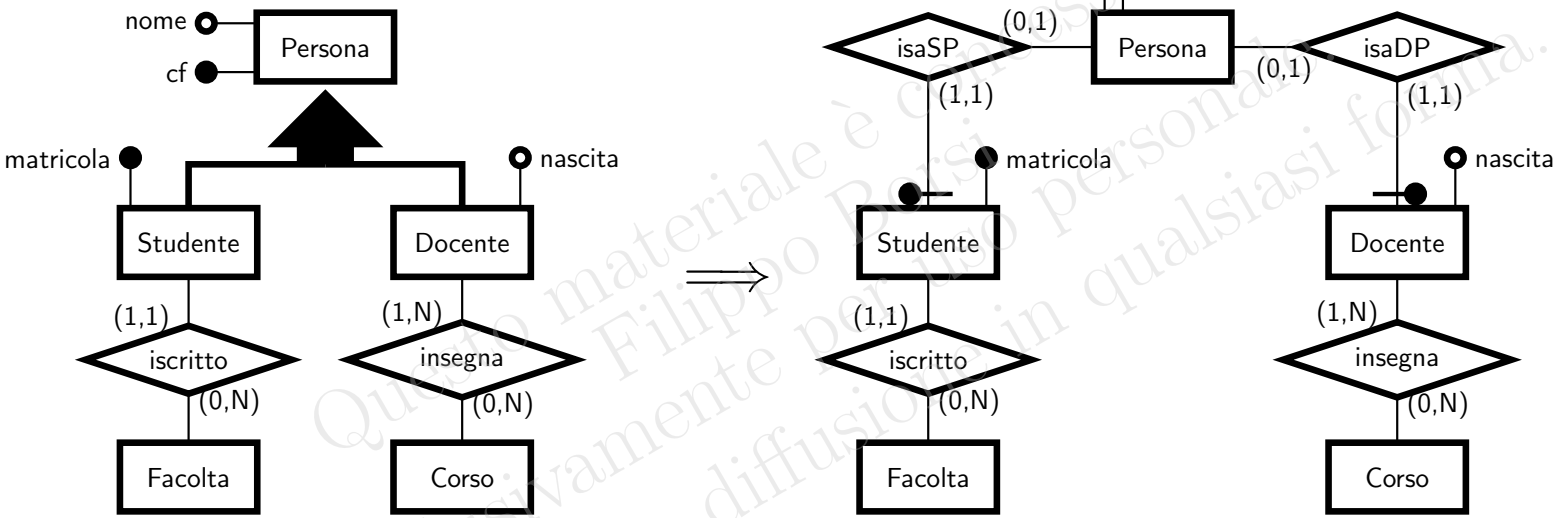
Sostituzione di is-a con relationship (2)



Vincolo (disgiunzione):

$$\forall p\ \text{Persona}(p) \rightarrow [\exists s\ \text{isaSP}(s,p)] \rightarrow [\nexists d\ \text{isaDP}(d,p)]$$

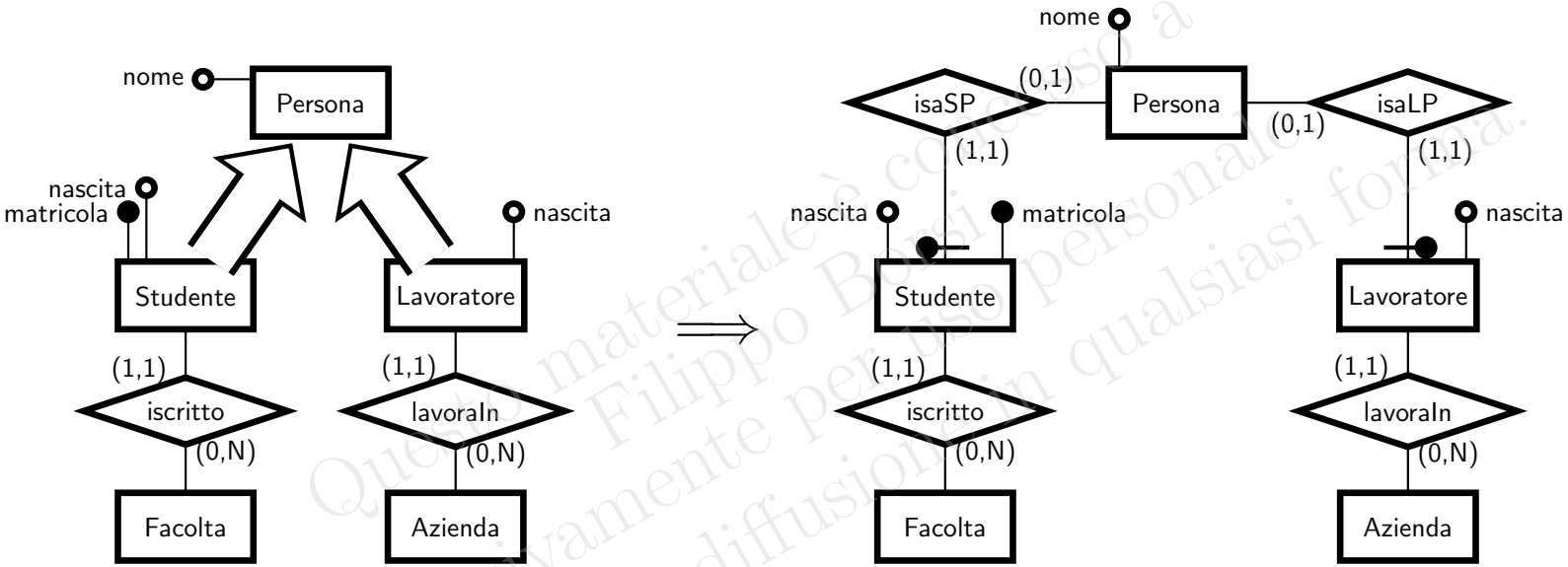
Sostituzione di is-a con relationship (3)



Vincolo (disgiunzione e completezza):

$$\forall p\ \text{Persona}(p) \rightarrow$$
$$[\exists s\ \text{isaSP}(s,p)] \rightarrow [\nexists d\ \text{isaDP}(d,p)]$$
$$\wedge$$
$$[\exists s\ \text{isaSP}(s,p)] \vee [\exists d\ \text{isaDP}(d,p)]$$

Sostituzione di is-a con relationship (4)



L'attributo comune alle due sotto-entità rappresenta la stessa relazione: istanze comuni a Studente e Lavoratore dell'ER concettuale dovranno avere un solo valore per l'attributo nascita.

Aggiungiamo il seguente vincolo nell'ER ristrutturato:

$$\forall p, s, l, n, n' \ \text{Persona}(p) \wedge \text{Studente}(s) \wedge \text{Lavoratore}(l) \wedge \text{isaSP}(s, p) \wedge \text{isaLP}(l, p) \wedge \text{nascita}(s, n) \wedge \text{nascita}(l, n') \rightarrow n = n'$$

Vantaggi e Svantaggi

Principali vantaggi e svantaggi di diversi approcci:

Alternativa	Vantaggi	Svantaggi
1. Fusione in un'unica entità	▶ Maggiore efficienza per le funzioni che non fanno molta distinzione tra le istanze delle diverse entità coinvolte	▶ Perdita di struttura ▶ L'entità raccoglie tutti gli attributi (con vincolo di card. (0,...)) delle sotto-entità (spreco di memoria) ▶ L'entità raccoglie tutti i ruoli (con vincolo di card. (0,...)) delle sotto-entità in relationship ▶ L'opzionalità di attributi e relationship provenienti dalle sotto-entità è soggetta a vincoli per rispettare la semantica dell'ER concettuale

Vantaggi e Svantaggi (2)

Alternativa	Vantaggi	Svantaggi
2. Divisione in entità disgiunte	▶ Maggiore efficienza per le funzioni che accedono ad istanze di sotto-entità note ▶ Risparmio di memoria rispetto ad alternativa 1 (niente aggiunta di valori nulli) ▶ Maggiore efficienza rispetto ad alternativa 3	▶ Perdita di struttura ▶ Utilizzabile solo per generalizzazioni complete ▶ Necessità di vincoli per gestire vincoli di identificazione su proprietà della entità base

Vantaggi e Svantaggi (3)

Alternativa	Vantaggi	Svantaggi
3. Sostituzione di is-a con relationship	▶ Mantiene la struttura il più possibile ▶ Facile gestione di non completezza e di non disgiunzione ▶ Risparmio di memoria rispetto ad alternativa 1 (no valori nulli) ▶ Maggiore efficienza rispetto ad alternativa 1 per funzioni che accedono ad istanze di sotto-entità note ▶ Facile gestione di vincoli di identificazione su proprietà della entità base	▶ Necessità di vincoli in caso di completezza o disgiunzione ▶ Minore efficienza rispetto ad alternativa 2 per le funzioni che accedono ad attributi ereditati di istanze di sotto-entità note (bisogna accedere anche all'entità base)

Eliminazione delle Relazioni IS-A tra Entità e delle Generalizzazioni

Al termine del passo di eliminazione delle relazioni is-a tra entità e delle generalizzazioni, tutte le entità del diagramma ristrutturato definiscono insiemi di istanze disgiunte a coppie

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.4 (S.B.3.1.2.2.4)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Relazioni IS-A tra Relationship

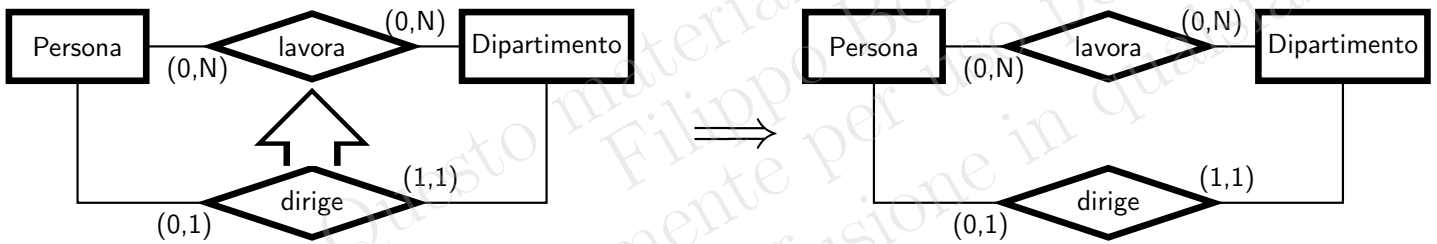
Versione 2016-05-14

Eliminazione delle Relazioni IS-A tra Relationship

Obiettivo

Ristrutturare il diagramma ER in uno (sostanzialmente) equivalente che non contenga relazioni is-a tra relationship (che non sono supportate dal DBMS).

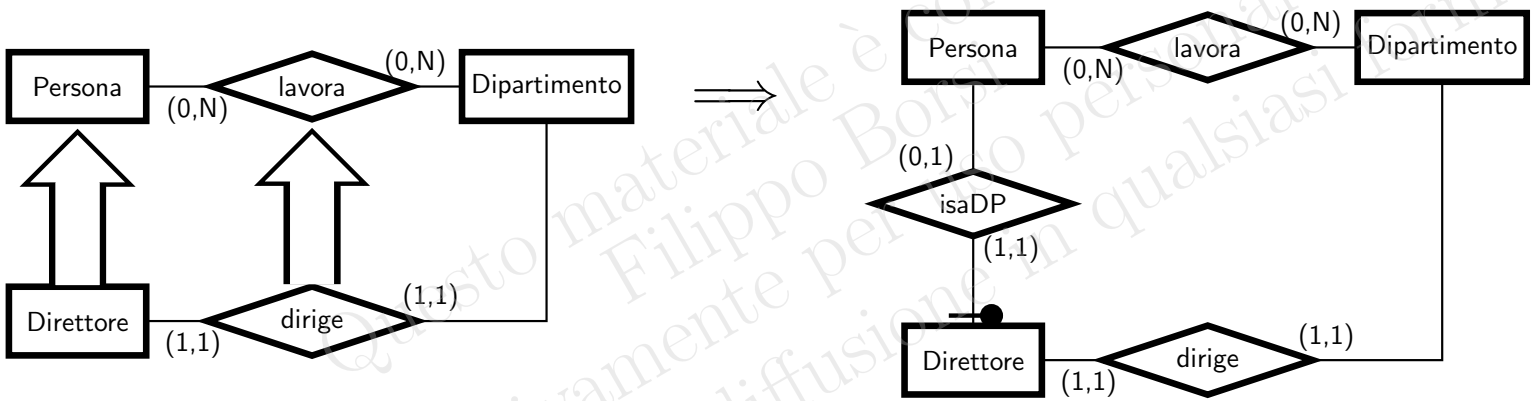
Eliminazione di is-a tra relationship



Vincolo:

$$\forall p, d \ \text{dirige}(p, d) \ \rightarrow \ \text{lavora}(p, d)$$

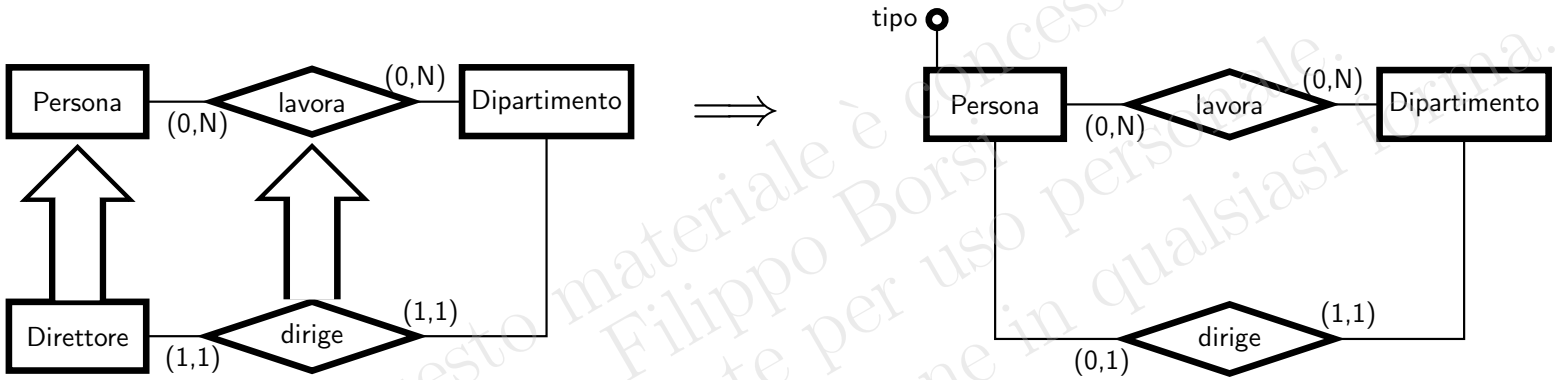
Eliminazione di is-a tra relationship (2)



Vincolo:

$$\forall p, dir, dip \ \text{dirige}(dir, dip) \wedge \text{isaDP}(dir, p) \rightarrow \text{lavora}(p, dip)$$

Eliminazione di is-a tra relationship (3)



Vincolo:

$$\forall p\ \text{Persona}(p) \rightarrow [\text{tipo}(p, \textit{Direttore}) \leftrightarrow \exists \textit{dip}\ \text{dirige}(p, \textit{dip})]$$

$$\wedge$$

$$\forall p, \textit{dip}\ \text{dirige}(p, \textit{dip}) \rightarrow \text{lavora}(p, \textit{dip})$$

Eliminazione delle Relazioni IS-A tra Relationship

Al termine di questo passo

non esistono relazioni is-a tra relationship

nel diagramma ER ristrutturato

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.5 (S.B.3.1.2.2.5)

Versione 2016-05-14

Definizione di identificatori per ogni Entità

Obiettivo

Ristrutturare il diagramma ER in uno (sostanzialmente) equivalente in cui ogni entità abbia almeno un identificatore.

Definizione di identificatori per ogni entità

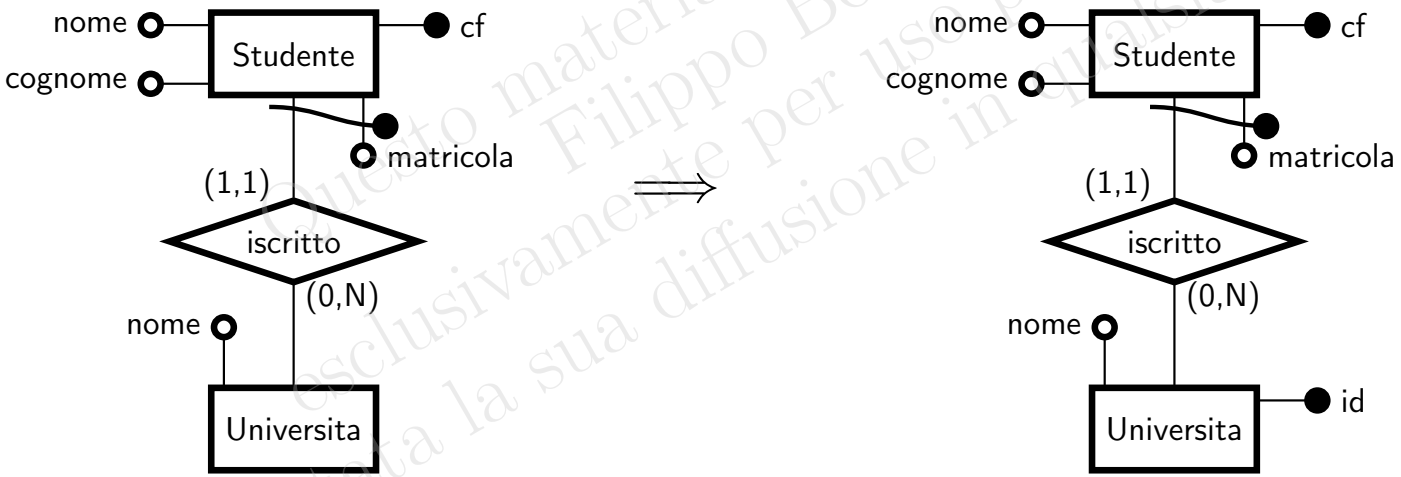
Il diagramma ER concettuale può già definire identificatori per alcune entità.

In questa fase dobbiamo:

1. assicurarci che ogni entità abbia almeno un identificatore
2. definire un identificatore primario per ogni entità, tenendo conto di requisiti di performance
3. rompere eventuali cicli nel grafo degli identificatori primari.

Definizione di identificatori per ogni entità (2)

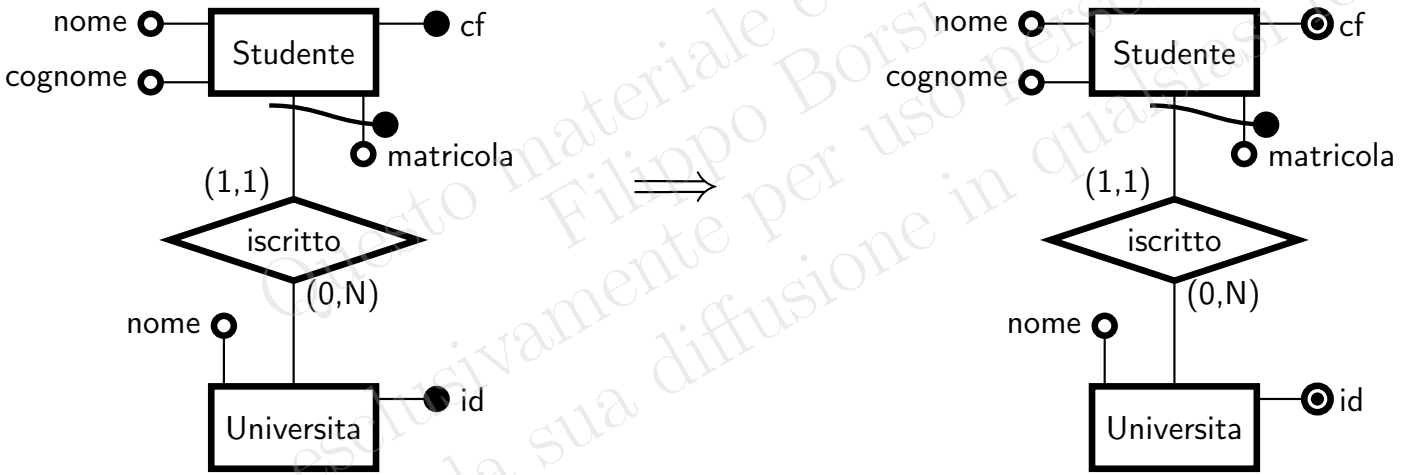
1. Ogni entità deve avere almeno un identificatore



In caso una entità del diagramma ER concettuale non sia provvista di alcun identificatore, viene aggiunto un nuovo attributo id : integer che funge da identificatore artificiale.

Definizione di identificatori per ogni entità (3)

2. Ogni entità deve avere un identificatore primario

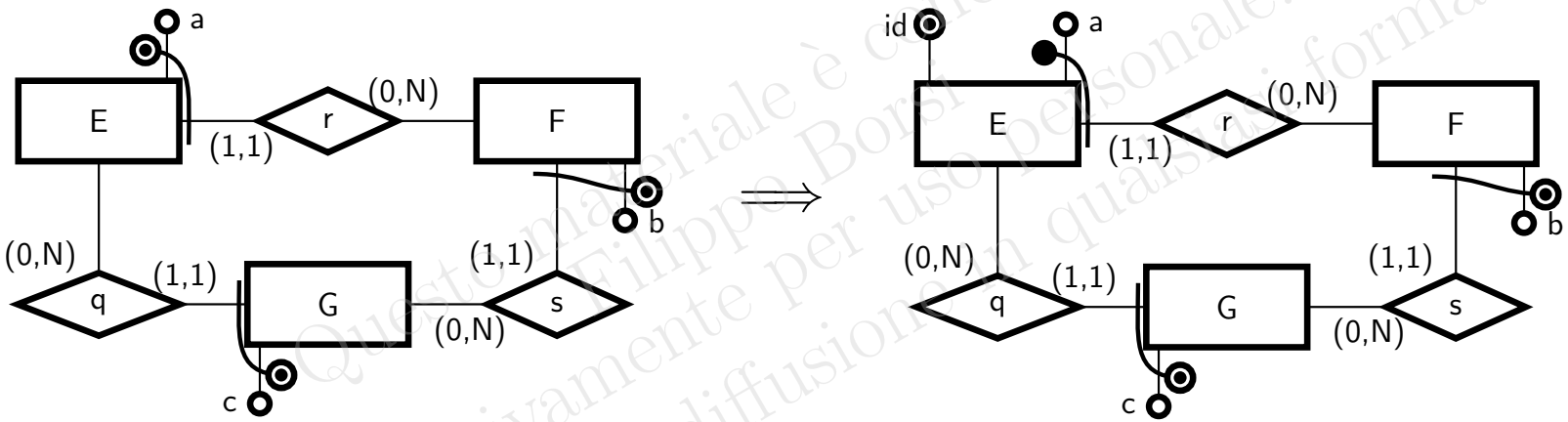


Criteri per la scelta dell'identificatore primario:

- Preferire attributi utilizzati nelle funzioni più importanti
- Preferire identificatori interni e semplici
- Considerare l'aggiunta di un identificatore artificiale se necessario.

Definizione di identificatori per ogni entità (4)

3. Gli identificatori primari esterni non devono formare cicli



Ciclo nel grafo avente come nodi le entità ed un arco $E_1 \to E_2$ sse E_1 partecipa ad una relationship presente nell'identificatore primario di E_2 (le istanze di E_1 contribuiscono ad identificare quelle di E_2).

Ogni ciclo di identificazione esterna è rotto scegliendo diversamente l'identificatore principale di una delle entità
(anche introducendo un nuovo attributo id).

Definizione di identificatori per ogni Entità

Al termine di questo passo

tutte le entità hanno un identificatore primario e

non esistono cicli nel grafo degli identificatori primari

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.6 (S.B.3.1.2.2.6)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Stima dei Costi delle Operazioni

Versione 2016-05-14

Stima dei volumi dei dati

È utile avere informazioni circa:

- Il numero atteso di istanze per ogni entità e relationship
- La frequenza attesa media dell'esecuzione di ogni funzione
- La tipologia di ogni funzione: interattiva o batch
- I requisiti di performance richiesti.
- ...

Alcune di queste informazioni possono essere presenti nella specifica dei requisiti (i cosiddetti requisiti non funzionali, che includono requisiti su affidabilità, privatezza, sicurezza, robustezza, usabilità, ...).

In tal caso, tali informazioni vengono riportate nello schema concettuale.

Un modello di costo delle operazioni

Durante il progetto di una base dati relazionale sono di estremo interesse informazioni su:

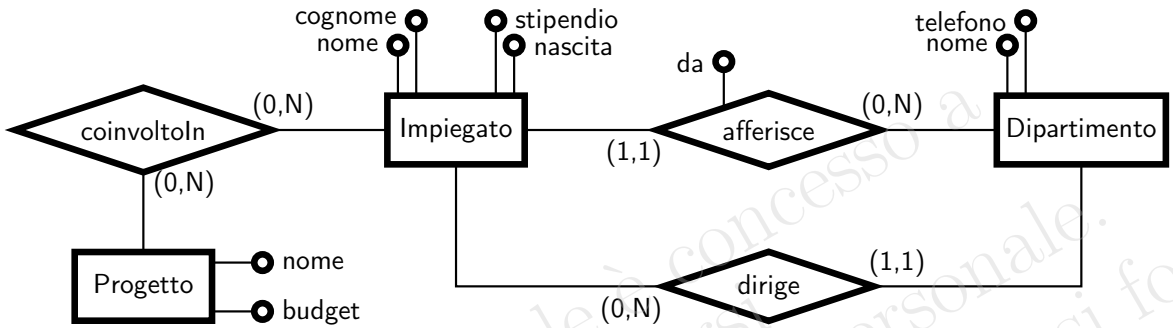
- Volume dei dati: numero atteso di istanze per ogni entità e relationship
- Frequenza delle operazioni: frequenza attesa media dell'esecuzione di ogni funzione
- Tipologia delle operazioni: interattiva o batch

In caso lo schema concettuale non abbia tali informazioni, è opportuno stimarle durante la fase di progettazione.

Sulla base di queste stime è possibile avere una misura quantitativa delle performance attese di ogni scelta di progetto vista in precedenza.

Un semplice modello di costo: Per ogni operazione di use-case, valutiamo il numero di istanze di ogni entità e relationship accedute in lettura ed in scrittura.

Esempio



Specifica use-case **Rapporti**

infoImp(i : Impiegato) : (Impiegato, Dipartimento, reale ≥ 0)

precondizioni: nessuna

postcondizioni:

Modifica del Livello Estensionale nessuna
Valore di Ritorno Siano:
– d il dipartimento di afferenza di i, ovvero tale per cui afferisce$(i, d) =$ true
– $P = \{(p, b) \mid \text{Progetto}(p) \wedge \text{coinvoltoIn}(i, p) \wedge \text{budget}(p, b)\}$ l'insieme dei progetti in cui i è coinvolto insieme con il loro budget
– $B = \sum_{(p,b)\in P}(b)$ il budget complessivo dei progetti di i.
result $= (i, d, B)$.

End

Esempio

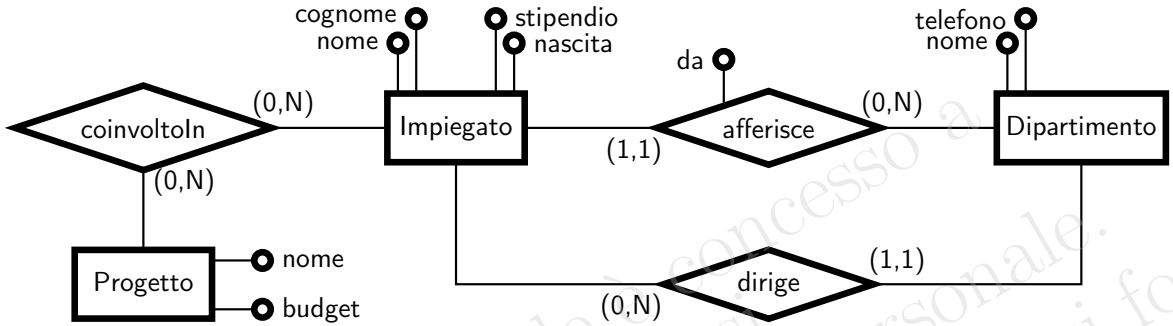


Tabella volumi:

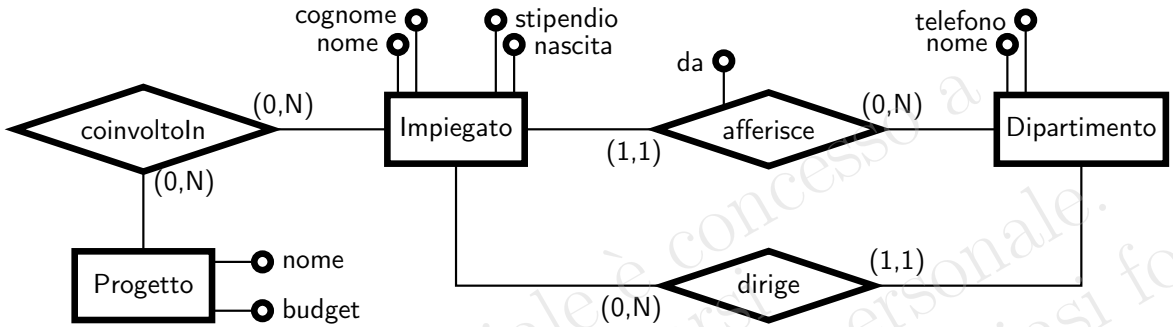
Entità/relationship	Numero atteso di istanze
Impiegato	100
Progetto	500
coinvoltoIn	300

Schema dell'operazione infoImp():

1. accesso (lettura) all'istanza (i, d) della relationship afferisce
2. accesso (lettura) all'istanza d dell'entità Dipartimento
3. accesso (lettura) a (mediamente) $300/100 = 3$ istanze della relationship coinvoltoIn
4. accesso (lettura) a (mediamente) 3 istanze dell'entità Progetto

$\Longrightarrow$ $1 + 1 + 3 + 3 = 8$ accessi in lettura.

Esempio



Specifica use-case strumentiSupervisore

assegnaProgetto(i: Impiegato, p: Progetto)

precondizioni: $\neg$coinvoltoIn(i, p)

postcondizioni:

Modifica del Livello Estensionale dei Dati:

Elementi del dominio di interpretazione: invariati
Variazioni nelle ennuple di predicati: coinvoltoIn(i, p).

Valore di Ritorno: nessuno

End

Esempio

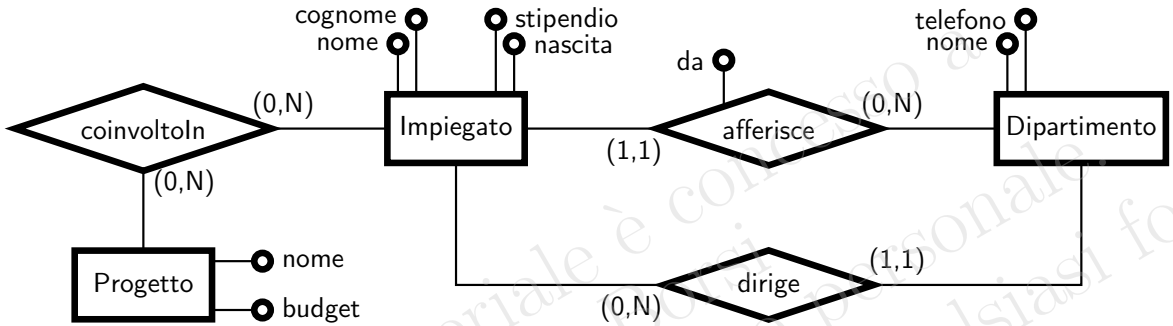


Tabella volumi:

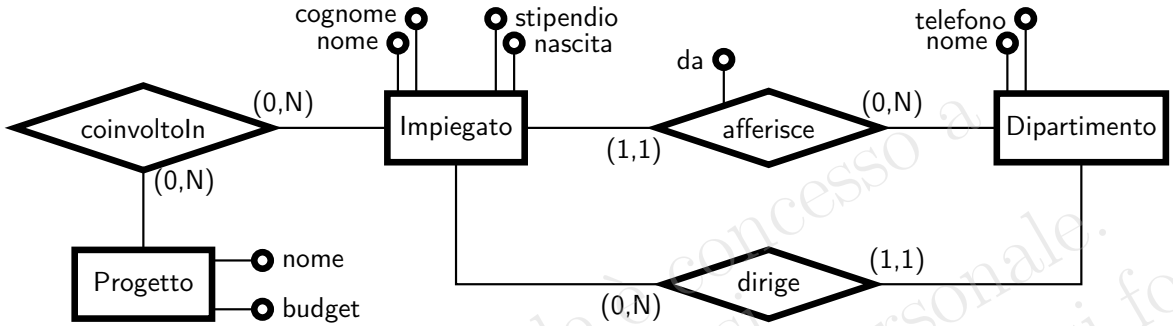
Entità/relationship	Numero atteso di istanze
Impiegato	100
Progetto	500
coinvoltoIn	300

Schema dell'operazione assegnaProgetto():

1. accesso (scrittura) alla relationship coinvoltoIn per la scrittura dell'istanza (i, p)

$\Longrightarrow$ 1 accesso in scrittura

Esempio



Costi e frequenza delle operazioni:

Operazione	Accessi Lettura	Accessi Scrittura	Esecuzioni al giorno
infoImp()	8	0	1000
assegnaProgetto()	0	1	5

Stimando il costo di un accesso in scrittura come il doppio di un accesso in lettura abbiamo:

Operazione	Accessi al giorno
infoImp()	8000
assegnaProgetto()	10
Costo totale	**8010**

Ristrutturazione e Vincoli di Performance

Il semplice modello di costo visto può essere utilizzato per decidere l'alternativa migliore durante i diversi passi della ristrutturazione del diagramma ER:

- come ristrutturare ogni generalizzazione?
- come gestire gli attributi multivalore?
- come gestire gli attributi composti?
- aggiungere un identificatore artificiale primario ad una certa entità che ha già identificatori?
- ...

Linee guida: durante l'analisi dei costi tenere in debita considerazione:

- la fisiologica imprecisione delle stime sulla frequenza delle operazioni e sui volumi dei dati
- cosa succede se le stime sono sotto-dimensionate!

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

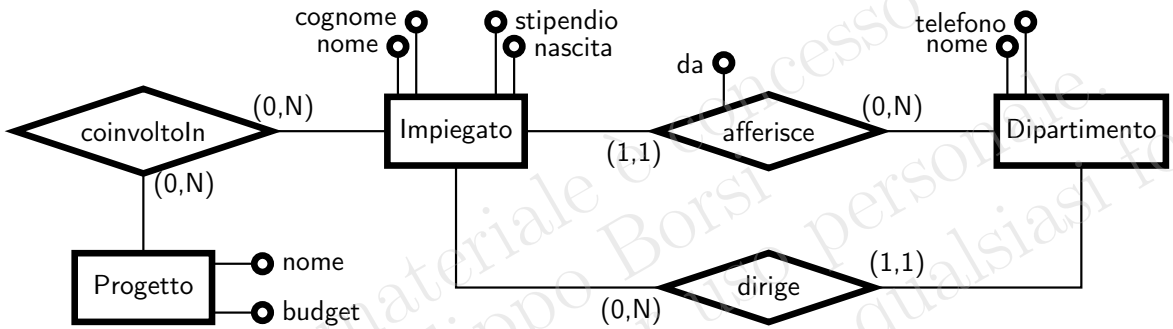
Slides B.3.1.2.2.7 (S.B.3.1.2.2.7)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Aggiunta Controllata di Ridondanza

Versione 2016-05-14

Aggiunta Controllata di Ridondanza

Esempio



Due operazioni di use-case:

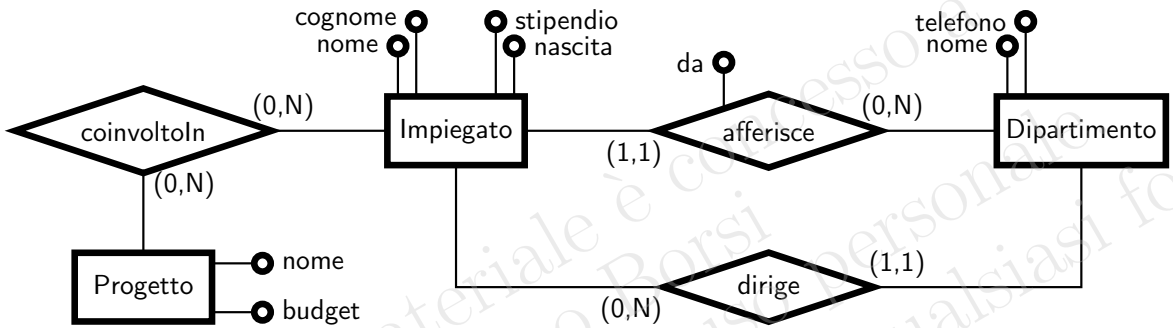
- infoImp(*i*:Impiegato) che restituisce il dipartimento dell'impiegato *i* e il budget totale dei progetti in cui è coinvolto
- assegnaProgetto(*i*:Impiegato, *p*:Progetto) che assegna l'impiegato *i* al progetto *p*.

Potremmo pensare di aggiungere l'attributo ridondante totBudget all'entità Impiegato per velocizzare l'operazione infoImp().

È una buona scelta?

Aggiunta Controllata di Ridondanza

Esempio



- Vantaggi: il costo dell'operazione infoImp() diventa di $1 + 1 = 2$ accessi in lettura invece di 8
- Svantaggi: per ogni impiegato, dobbiamo mantenere il valore dell'attributo totBudget uguale alla somma dei budget dei progetti in cui è coinvolto

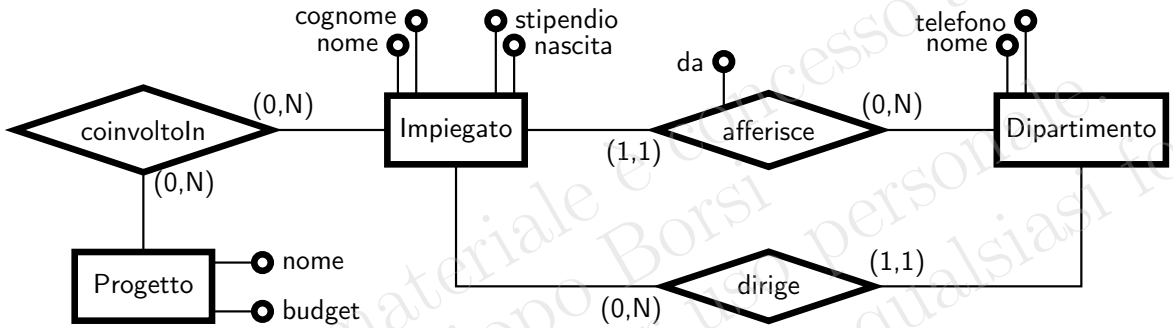
  $\Longrightarrow$ L'operazione assegnaProgetto() diventa più costosa:

  1. accesso (scrittura) alla relationship coinvoltoIn per la scrittura dell'istanza (i, p)
  2. accesso (scrittura) alla entità Impiegato per scrivere il nuovo valore di totBudget

  Il suo costo diventa di $1 + 1 = 2$ accessi in scrittura invece di 1

Aggiunta Controllata di Ridondanza

Esempio



operazione	esecuzioni giorno	senza ridondanza lettura	senza ridondanza scrittura	con ridondanza lettura	con ridondanza scrittura
infoImp()	1000	8	0	2	0
assegnaProgetto()	5	0	1	0	2

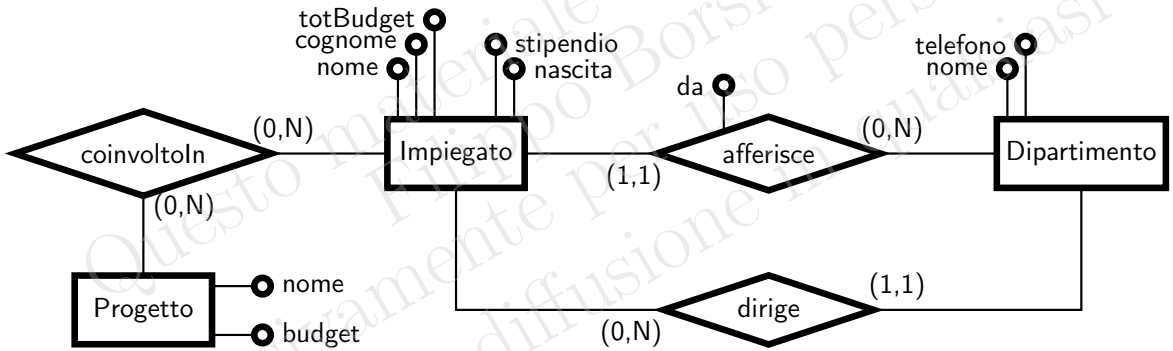
operazione	senza ridondanza lettura	senza ridondanza scrittura	senza ridondanza totale	con ridondanza lettura	con ridondanza scrittura	con ridondanza totale
infoImp()	8000	0	8000	2000	0	2000
assegnaProgetto()	0	5	10	0	10	20
totale accessi al giorno	8000	5	**8010**	2000	10	**2020**

$\Longrightarrow$ aggiungiamo la ridondanza

Aggiunta Controllata di Ridondanza

Esempio

. . . Aggiungiamo la ridondanza



e il vincolo esterno:

$$\forall i, t \ \text{Impiegato}(i) \wedge \text{totBudget}(i, t) \rightarrow t = \sum_{(p,b) \in B_i} (b)$$

dove $B_i = \{(p, b) | \text{Progetto}(p) \wedge \text{coinvolto}(i, p) \wedge \text{budget}(p, b)\}$.

Aggiunta Controllata di Ridondanza: Linee Guida

- Usare molta parsimonia: aggiungere ridondanza solo dove i benefici sono davvero rilevanti!
- Gestire la ridondanza mediante vincoli esterni
- Documentare le ragioni della scelta di introdurre la ridondanza
- Durante l'analisi dei costi, tenere in debita considerazione la fisiologica imprecisione delle stime sulla frequenza delle operazioni e sui volumi dei dati
- Tenere in debita considerazione cosa succede se le stime sono sotto-dimensionate!

Aggiunta Controllata di Ridondanza

Al termine di questo passo
eventuali decisioni di introdurre ridondanza per motivi di performance
sono state prese in modo esplicito, motivato e documentato,
e i necessari vincoli esterni sono stati definiti per mantenere consistente
la semantica del diagramma

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.2.8 (S.B.3.1.2.2.8)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Ristrutturazione del Diagramma ER
Vincoli Esterni e Specifiche di Use-Case

Versione 2016-05-14

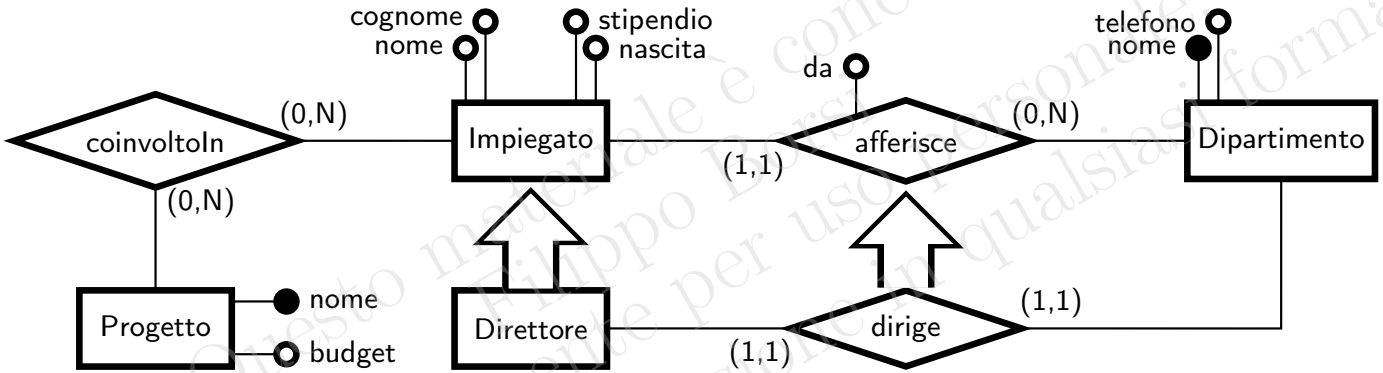
Vincoli Esterni e Operazioni di Use-Case

Obiettivo

Ristrutturare i vincoli esterni e le specifiche delle operazioni di use-case in termini del diagramma ER ristrutturato.

Esempio

Schema concettuale

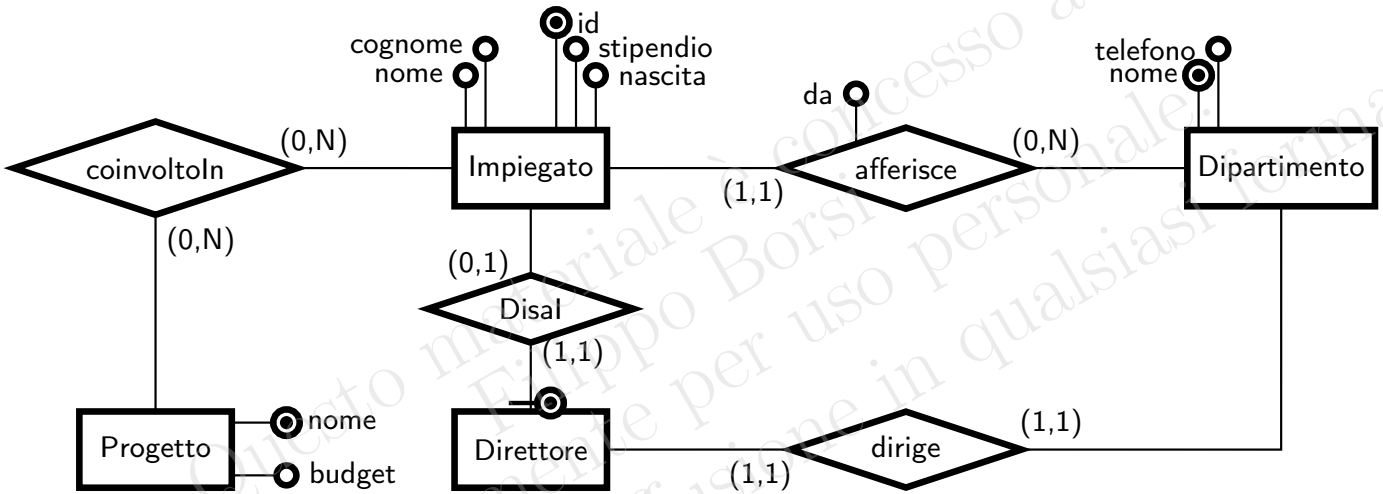


Vincolo [V.Direttore.afferenza]:

$$\forall \text{dir,dip, annoAff, annoCorr}$$
$$[\text{Direttore(dir)} \wedge \text{afferisce(dir, dip)} \wedge \text{da(dir, dip, annoAff)} \wedge$$
$$\text{anno(adesso, annoCorr)}] \rightarrow \text{annoCorr} - \text{annoAff} \geq 5$$

Esempio

Diagramma ER e vincolo esterno ristrutturati



Vincolo [V.dirige.isa]:
$\forall dir, imp, dip$ dirige$(dir, dip) \wedge$ DisaI$(dir, imp) \rightarrow$ afferisce(imp, dip)

Vincolo [V.Direttore.afferenza]:

$\forall$dir, imp,dip, annoAff, annoCorr

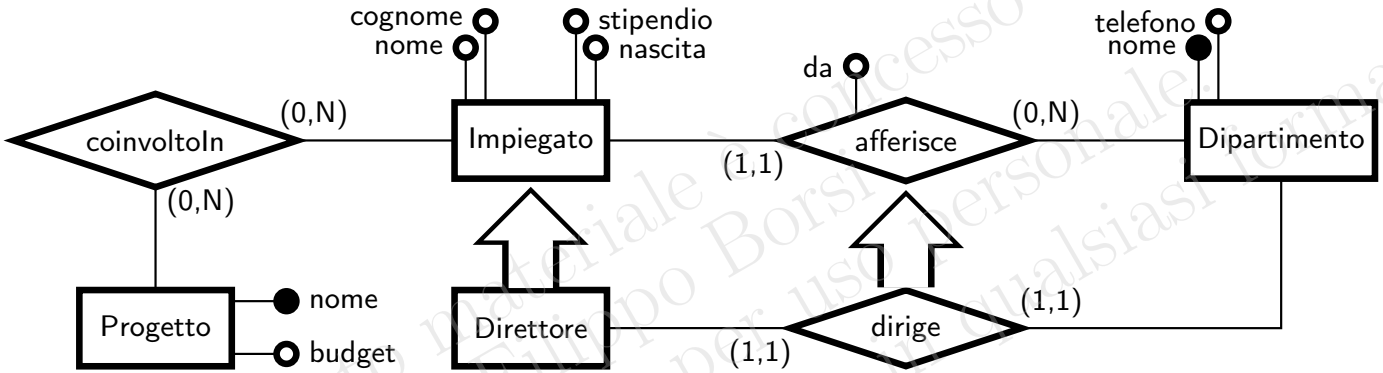
[Direttore(dir) $\wedge$ Impiegato(imp) $\wedge$ DisaI(dir, imp)$\wedge$

afferisce(imp, dip) $\wedge$ da(imp, dip, annoAff)$\wedge$

anno(adesso, annoCorr)] $\rightarrow$ annoCorr $-$ annoAff ≥ 5

Esempio

Schema concettuale



assumiDirettore(n:stringa, c:stringa, s:reale≥ 0, nasc:intero> 0, dip:Dipart.) : Direttore
precondizioni: nessuna

postcondizioni:
Modifica del Livello Estensionale dei Dati:

Elementi del dominio di interpretazione: un nuovo elemento α

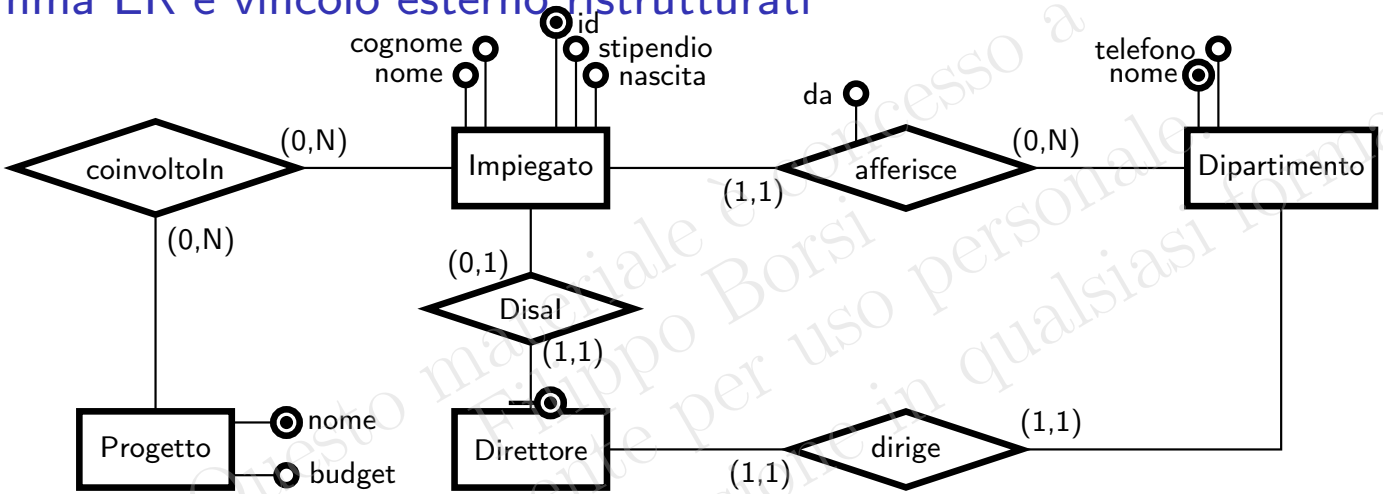
Ennuple di predicati che non esistono più: dirige(δ, *dip*), dove δ è l'unico elemento del dominio di interpretazione che, nel livello estensionale di partenza, rendeva vero dirige(δ, *dip*).

Nuove ennuple di predicati: Direttore(α), Impiegato(α), nome(α, *n*), cognome(α, *c*), stipendio(α, *s*), nascita(α, *nasc*), afferisce(α, *dip*), dirige(α, *dip*)

Valore di Ritorno: *result* $= \alpha$

Esempio

Diagramma ER e vincolo esterno ristrutturati



assumiDirettore(n:varchar(100), c:varchar(100), s:RealePos, nasc:IntPos, dip:Dipart.) : Direttore

precondizioni: nessuna

postcondizioni:

Modifica del Livello Estensionale dei Dati:

Elementi del dominio di interpretazione: nuovi elementi α, β

Ennuple di predicati che non esistono più: dirige(δ, *dip*), dove δ è l'unico elemento del dominio di interpretazione che, nel livello estensionale di partenza, rendeva vero dirige(δ, *dip*).

Nuove ennuple di predicati: Direttore(α), Impiegato(β), Disal(α, β), nome(β, n), cognome(β, c), stipendio(β, s), nascita(β, *nasc*), afferisce(β, *dip*), dirige(α, *dip*)

Valore di Ritorno: *result* $= \alpha$

Vincoli Esterni e Operazioni di Use-Case

Al termine di questo passo
tutti i vincoli esterni e la specifica delle operazioni di use-case
sono stati ristrutturati in termini del diagramma ER ristrutturato.
Lo schema ristrutturato del progetto è consistente e
(sostanzialmente) equivalente allo schema concettuale.

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3 (S.B.3.1.2.3)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato

Versione 2016-05-14

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.1 (S.B.3.1.2.3.1)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Introduzione

Versione 2016-05-14

Traduzione Diretta dell'ER Ristrutturato

Dopo aver ristrutturato il diagramma ER, i vincoli esterni e le specifiche di use-case, siamo pronti per produrre lo schema relazionale della base dati e i vincoli.

- Input: diagramma ER ristrutturato più vincoli esterni in FOL
- Output: schema relazionale della base dati più vincoli supportati dal DBMS.

Lo schema relazionale ottenuto può essere ulteriormente raffinato per tener conto di vincoli di performance

Schema Generale

La traduzione diretta del diagramma ER ristrutturato in schema relazionale segue il seguente schema generale:

- Le entità si traducono in relazioni
- Le relationship si traducono in relazioni e vincoli di chiave esterna, oppure si accorpano con una delle entità coinvolte
- I vincoli di cardinalità si traducono in vincoli di chiave, vincoli di chiave esterna, o in vincoli esterni (tipicamente vincoli di inclusione).

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

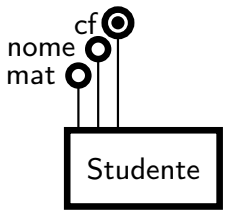
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

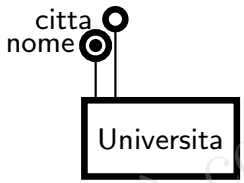
Slides B.3.1.2.3.2 (S.B.3.1.2.3.2)

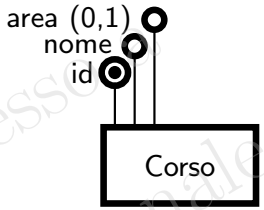
Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Entità

Versione 2016-05-14

Traduzione di entità







Entità Studente

attributo	dominio
mat	PosInt
nome	varchar(100)
cf	char(16)

Entità Universita

attributo	dominio
nome	varchar(100)
citta	varchar(100)

Entità Corso

attributo	dominio
id	PosInt
nome	varchar(100)
area	varchar(100)

- Definizione del dominio PosInt come intero > 0
- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), cognome:varchar(100), mat:PosInt)
  - **Universita** (nome:varchar(100), citta:varchar(100))
  - **Corso** (id:PosInt, nome:varchar(100), area*:varchar(100))

(*): può assumere valori **NULL**

Creazione di Tabelle Relazionali

Le relazioni così definite possono essere successivamente create nella base dati, durante la Fase di Realizzazione, mediante costrutti SQL **create table**.

Esempio:

Corso (id:PosInt, nome:varchar(100), area*:varchar(100))

```
create table Corso (
   id PosInt not null,
   nome varchar(100) not null,
   area varchar(100),
   primary key (id)
)
```

Nota: l'attributo area (contrassegnato con '*') può assumere valori **NULL**

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.3 (S.B.3.1.2.3.3)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship Binarie

Versione 2016-05-14

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

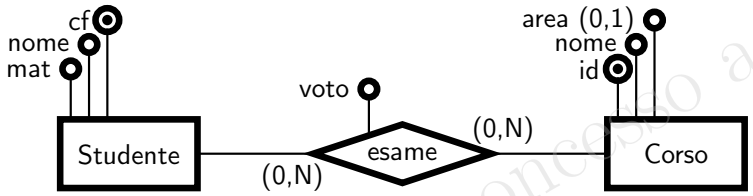
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.3.1 (S.B.3.1.2.3.3.1)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship Binarie
Relationship (0,N) – (0,N)

Versione 2016-05-14

Traduzione di Relationship (0,N) – (0,N)



Relationship esame

attributo	dominio
voto	Voto

- Definizione del dominio Voto come intero tra 18 e 30
- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), . . .)
  - **Corso** (id:PosInt, . . .)
  - **esame** (studente:char(16), corso:PosInt, voto:Voto)
    Vincolo foreign key: studente references Studente(cf)
    Vincolo foreign key: corso references Corso(id)

Creazione di Tabelle Relazionali

Le relazioni così definite possono essere successivamente create nella base dati, durante la Fase di Realizzazione, mediante costrutti SQL **create table**.

Esempio:

esame (studente:char(16), corso:PosInt, voto:Voto)

Vincolo foreign key: studente references Studente(cf)
Vincolo foreign key: corso references Corso(id)

```
create table esame (
   studente char(16) not null,
   corso PosInt not null,
   voto Voto not null,
   primary key (studente, corso),
   foreign key studente references Studente(cf),
   foreign key corso references Corso(id)
)
```

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

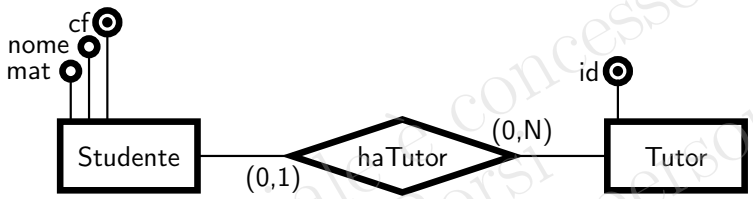
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.3.2 (S.B.3.1.2.3.3.2)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship Binarie
Relationship (0,1) – (0,N)

Versione 2016-05-14

Traduzione di relationship (0,1) – (0,N)



- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), . . .)
  - **Tutor** (id:PosInt, . . .)
  - **haTutor** (studente:char(16), tutor:PosInt)
    Vincolo foreign key: studente references Studente(cf)
    Vincolo foreign key: tutor references Tutor(id)

Nota: vincolo (. . . , 1) $\Longrightarrow$ la chiave di haTutor è formata solo dall'attributo studente

Creazione di Tabelle Relazionali

La relazione

haTutor (studente:char(16), tutor:PosInt)

Vincolo foreign key: studente references Studente(cf)
Vincolo foreign key: tutor references Tutor(id)

può essere creata nella base dati, durante la Fase di Realizzazione, mediante il costrutto SQL:

```
create table haTutor (
   studente char(16) not null,
   tutor PosInt not null,
   primary key (studente),
   foreign key studente references Studente(cf),
   foreign key tutor references Tutor(id)
)
```

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

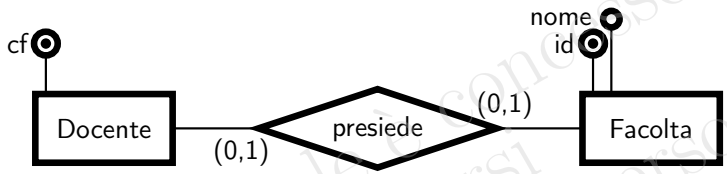
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.3.3 (S.B.3.1.2.3.3.3)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship Binarie
Relationship (0,1) – (0,1)

Versione 2016-05-14

Traduzione di relationship (0,1) – (0,1)



- Definizione delle seguenti relazioni con vincoli:
  - **Docente** (cf:char(16), . . .)
  - **Facolta** (id:integer, . . .)
  - **presiede** (docente:char(16), facolta:integer)
    Vincolo foreign key: docente references Docente(cf)
    Vincolo foreign key: facolta references Facolta(id)
    Vincolo chiave: facolta

Nota: vincoli (. . . , 1) $\Longrightarrow$ l'attributo docente e l'attributo facolta formano due chiavi di presiede. Una viene scelta come primaria

Creazione di Tabelle Relazionali

La relazione

presiede (docente:char(16), facolta:integer)

Vincolo foreign key: docente references Docente(cf)
Vincolo foreign key: facolta references Facolta(id)
Vincolo chiave: facolta

può essere creata nella base dati, durante la Fase di Realizzazione, mediante il costrutto SQL:

```
create table presiede (
   docente char(16) not null,
   facolta integer not null,
   primary key (docente),
   foreign key docente references Docente(cf),
   foreign key facolta references Facolta(id),
   unique facolta
)
```

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

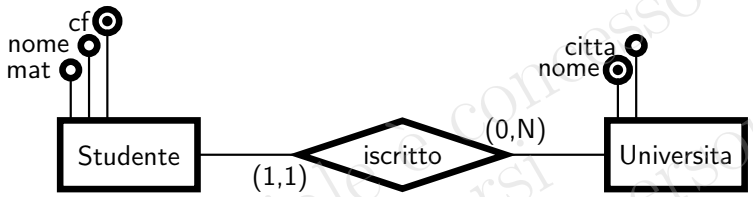
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.3.4 (S.B.3.1.2.3.3.4)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship Binarie
Relationship (1,1) – (0,N)

Versione 2016-05-14

Traduzione di relationship (1,1) – (0,N)



- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), . . .)
    Vincolo inclusione: cf $\subseteq$ iscritto(studente)
  - **Universita** (nome:varchar(100), . . .)
  - **iscritto** (studente:char(16), universita:varchar(100))
    Vincolo foreign key: studente references Studente(cf)
    Vincolo foreign key: universita references Universita(nome)

Nota: vincolo (1,. . .) $\Longrightarrow$ vincolo di inclusione di Studente(cf) in iscritto

Creazione di Tabelle Relazionali

La relazione per l'entità Studente:

Studente (cf:char(16), ...)

Vincolo inclusione: cf $\subseteq$ iscritto(studente)

può essere creata nella base dati, durante la Fase di Realizzazione, osservando che il vincolo di inclusione è in realtà un vincolo di foreign key, essendo l'attributo studente chiave primaria della relazione iscritto:

iscritto (studente:char(16), universita:varchar(100))

La relazione può quindi essere definita mediante il costrutto SQL:

```
create table Studente (
  cf char(16) not null, ...,
  primary key (cf),
  foreign key cf references iscritto(studente)
)
```

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

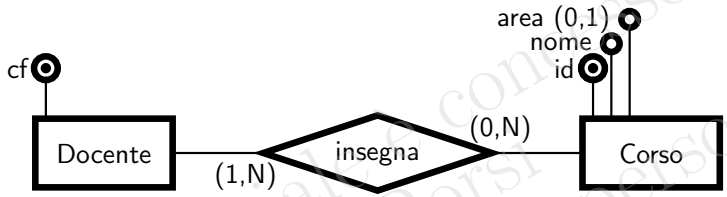
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.3.5 (S.B.3.1.2.3.3.5)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship Binarie
Relationship (1,N) – (0,N)

Versione 2016-05-14

Traduzione di relationship (1,N) – (0,N)



- Definizione delle seguenti relazioni con vincoli:
  - **Docente** (cf:char(16), . . .)
    Vincolo inclusione: cf $\subseteq$ insegna(docente)
  - **Corso** (id:PosInt, . . .)
  - **insegna** (docente:char(16), corso:PosInt)
    Vincolo foreign key: docente references Docente(cf)
    Vincolo foreign key: corso references Corso(id)

Nota: vincolo (1,. . .) $\Longrightarrow$ vincolo di inclusione di Docente(cf) in insegna

Creazione di Tabelle Relazionali

Il vincolo di inclusione per l'entità Docente:

Docente (cf:char(16), ...)

Vincolo inclusione: cf $\subseteq$ insegna(docente)

non può essere implementato mediante un vincolo di foreign key, perché l'attributo docente della relazione insegna:

insegna (docente:char(16), corso:PosInt)

non è chiave primaria.

La relazione Docente va definita, nella Fase di Realizzazione, ignorando il vincolo di inclusione:

```
create table Docente (
  cf char(16) not null, ...,
  primary key (cf)
)
```

Il vincolo di inclusione andrà definito come vincolo esterno.

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

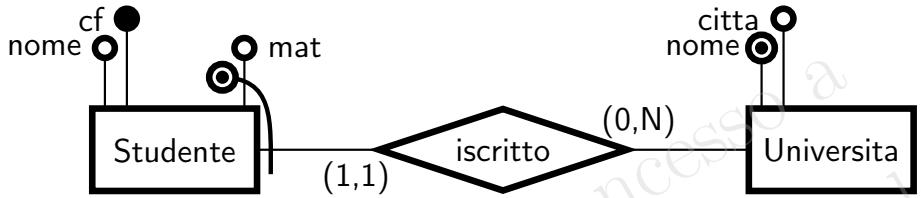
Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

Slides B.3.1.2.3.4 (S.B.3.1.2.3.4)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Accorpamento di Relationship in Entità

Versione 2016-05-14

Traduzione di Entità con Identificatori Primari Esterni



- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), nome:varchar(100), mat:PosInt, iscritto:varchar(100))
    Vincolo foreign key: iscritto references Universita(nome)
    Vincolo chiave: cf
  - **Universita** (nome:varchar(100), . . .)

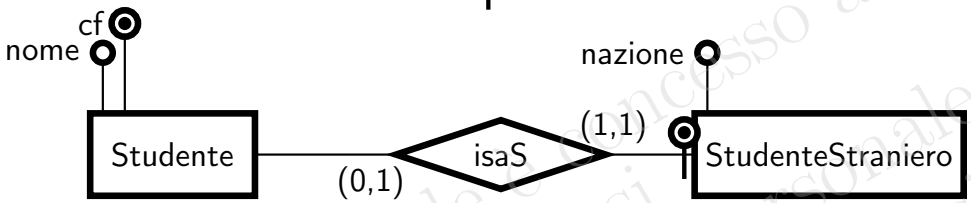
Nota: per esprimere il vincolo di identificazione primario $\{mat, iscritto\}$ dobbiamo avere entrambi i dati nella stessa relazione.

$\Longrightarrow$ La relazione per la relationship iscritto viene accorpata alla relazione per l'entità Studente.

L'aver eliminato cicli tra identificatori primari esterni garantisce che è sempre possibile procedere agli accorpamenti.

Traduzione di Entità con Identificatori Primari Esterni (2)

Caso tipico: traduzione di relationship derivanti da ristrutturazione di is-a



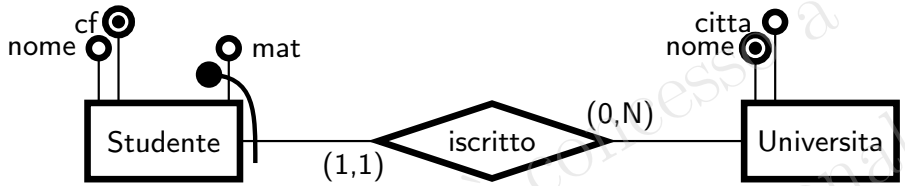
- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), . . .)
  - **StudenteStraniero** (studente:char(16), nazione:varchar(100))
    Vincolo foreign key: studente references Studente(cf)

Nota: per esprimere il vincolo di identificazione primario {*isaS*} dobbiamo avere entrambi i dati nella stessa relazione

$\Longrightarrow$ la relazione per la relationship isaS viene accorpata alla relazione per l'entità StudenteStraniero.

L'aver eliminato cicli tra identificatori primari esterni garantisce che è sempre possibile procedere agli accorpamenti.

Entità con Identificatori Esterni Non Primari

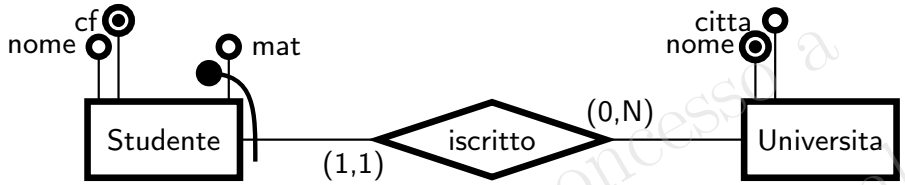


- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), nome:varchar(100), mat:PosInt, iscritto:varchar(100))
    Vincolo foreign key: iscritto references Universita(nome)
    Vincolo chiave: (mat, iscritto)
  - **Universita** (nome:varchar(100), . . .)

Nota: per esprimere il vincolo di identificazione $\{mat, iscritto\}$ è comodo avere entrambi i dati nella stessa relazione.

$\Longrightarrow$ la relazione per la relationship iscritto può essere accorpata alla relazione per l'entità Studente, eliminando eventuali cicli di identificazione esterna (che abbiamo già eliminato solo per gli identificatori primari).

Entità con Identificatori Esterni Non Primari



In caso non si voglia procedere all'accorpamento, il vincolo di identificazione $\{mat, iscritto\}$ dovrà essere ristrutturato come vincolo esterno e gestito separatamente:

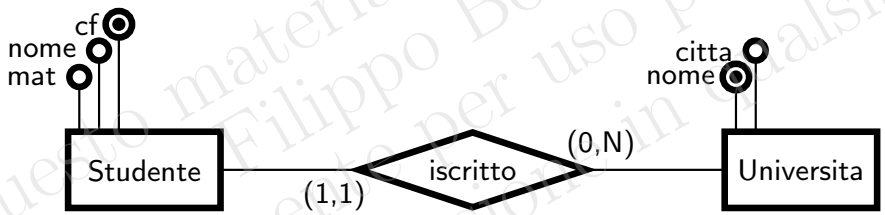
- Definizione delle seguenti relazioni con vincoli:
  - **Studente** (cf:char(16), nome:varchar(100), mat:PosInt)
    Vincolo inclusione: cf $\subseteq$ iscritto(studente)
  - **Universita** (nome:varchar(100), . . .)
  - **iscritto** (studente:char(16), universita:varchar(100))
    Vincolo foreign key: studente references Studente(cf)
    Vincolo foreign key: universita references Universita(nome)
- Definizione del vincolo esterno V.Studente.identificazione:

$$\forall s, s', u, m\ [\text{Studente}(s) \wedge \text{Studente}(s') \wedge \text{mat}(s, m) \wedge \text{mat}(s', m) \wedge \text{Universita}(u) \wedge \text{iscritto}(s, u) \wedge \text{iscritto}(s', u)] \rightarrow s = s'$$

Ulteriori Accorpamenti

In linea di principio, è possibile procedere all'accorpamento di una relationship r in una entità E ogni volta che il vincolo di molteplicità massima di E in r è 1.

Esempio

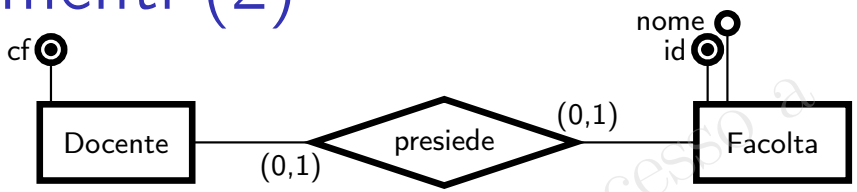


Possiamo accorpare la relationship iscritto in Studente.

- **Studente** (cf:char(16), . . . , iscritto:varchar(100))
  Vincolo foreign key: iscritto references Universita(nome)
- **Universita** (nome:varchar(100), . . .)

Ulteriori Accorpamenti (2)

Esempio



Possiamo accorpare la relationship presiede in Facolta:

- **Docente** (cf:char(16))
- **Facolta** (id:integer, nome:varchar(100), preside*:char(16))
  Vincolo foreign key: preside references Docente(cf)
  Vincolo chiave: preside

oppure possiamo accorpare la relationship presiede in Docente:

- **Docente** (cf:char(16), facoltaPresieduta*:integer)
  Vincolo foreign key: facoltaPresieduta references Facolta(id)
  Vincolo chiave: facoltaPresieduta
- **Facolta** (id:integer, nome:varchar(100))

Nota: gli attributi contrassegnati da '*' possono avere valori **NULL**.

Ulteriori Accorpamenti (3)

Come decidere se procedere ad accorpamenti non obbligatori?

Utilizzare il modello di stima dei costi delle operazioni di use-case per ottimizzare le performance complessive, dati:

- la stima dei volumi
- la natura (interattiva o batch) delle singole operazioni
- la frequenza relativa delle operazioni.

In generale, accorpare una relationship r nell'entità E:

- Pro: Evita di effettuare il join con la tabella per r in caso si voglia risalire alle istanze dell'altra entità collegata ad r.
- Contro: Rende ogni ennupla di E più grande. Quindi il caricamento da memoria di massa di tanti record di E è meno efficiente (una pagina caricata da disco conterrà un minor numero di record).

Una scelta ottimale richiede di valutare pro e contro dell'accorpamento caso per caso.

SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ing. dell'Informazione, Informatica e Statistica
Laurea in Informatica

Basi di Dati, Modulo 2

Prof. Toni Mancini, Prof. Federico Mari
Dipartimento di Informatica
http://tmancini.di.uniroma1.it
http://mari.di.uniroma1.it

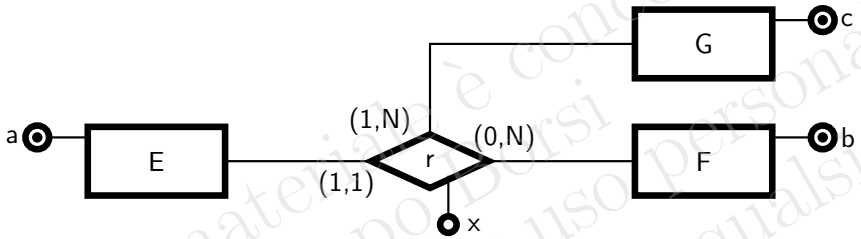
Slides B.3.1.2.3.5 (S.B.3.1.2.3.5)

Basi di Dati Relazionali
La Fase di Progettazione
Progettazione della Base Dati
Produzione dello Schema Relazionale con Vincoli
Traduzione Diretta del Diagramma ER Ristrutturato
Traduzione di Relationship non Binarie

Versione 2016-05-14

Traduzione di Relationship *n*-arie

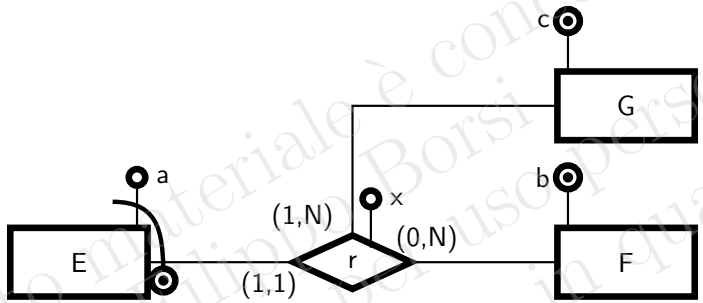
Generalizzazione diretta dell'approccio visto. Esempio:



- Definizione delle seguenti relazioni con vincoli:
  - **E** (a:varchar(100), . . .)
    Vincolo foreign key: a references r(e)
  - **F** (b:varchar(100), . . .)
  - **G** (c:varchar(100), . . .)
    Vincolo inclusione: c $\subseteq$ r(g)
  - **r** (e:varchar(100), f:varchar(100), g:varchar(100), x:varchar(100))
    Vincolo foreign key: e references E(a)
    Vincolo foreign key: f references F(b)
    Vincolo foreign key: g references G(c)

Accorpamento di Relationship *n*-arie

Generalizzazione diretta dell'approccio visto. Esempio:



- Definizione delle seguenti relazioni con vincoli:
  - **E** (a:varchar(100), f:varchar(100), g:varchar(100), rx:varchar(100))
    Vincolo foreign key: f references F(b)
    Vincolo foreign key: g references G(c)
  - **F** (b:varchar(100), . . .)
  - **G** (c:varchar(100), . . .)
    Vincolo inclusione: c $\subseteq$ E(g)