😷

Capitolo 8 (Progettazione logica)

Requisiti della base di dati

> Progettazione concettuale (schema concettuale)
>
> Progettazione logica (schema logico)
>
> Progettazione fisica (schema fisico)

Obiettivo della progettazione logica

“Tradurre” lo schema concettuale in uno schema logico che rappresenti gli stessi dati nel formato di un modello intermedio **(modello logico)**, ad es. il modello relazionale

Dati di ingresso e uscita:

- **Ingresso:**
  - Schema concettuale
  - Informazioni sul carico applicativo
  - Modello logico
- **Uscita:**
  - Schema logico (memorizzabile tramite il DBMS)
  - Documentazione associata

Traduzione ER-Relazionale

Non si tratta di una semplice trascrizione tra i due modelli

Alcuni aspetti dello schema concettuale non sono direttamente rappresentabili nello schema logico

In questa fase è opportuno anche valutare le prestazioni

Sarà necessaria una ristrutturazione, anche perché, non è detto che i costrutti dello schema concettuale siano presenti come come costrutti nel modello logico scelto

Ristrutturazione schema ER

Motivazioni:

- Semplificare la traduzione
- “Ottimizzare” le prestazioni

Per ottimizzare il risultato abbiamo bisogno di analizzare le prestazioni a questo livello

Le prestazioni non sono valutabili con precisione su di uno schema concettuale

Parametri per valutare le prestazioni:

- **Costo di una operazione**
  - Valutato in termini di numero di occorrenze previste per entità e associazioni che mediamente vanno visitate per rispondere a operazioni sula base di dati
- **Occupazione di memoria**
  - Valutato in termini di spazio della memoria (misurato per esempio in byte) necessario per memorizzare i dati

descritti dallo schema

Principio di Pareto (80:20)

Regola empirica secondo la quale un sistema dedica l’80% delle sue risorse per elaborare il 20% delle operazioni più frequenti

Sfruttando questo principio calcoliamo gli accessi totali per il 20% di operazioni più frequenti

In questo modo si potranno stabilire le prestazioni soltanto di $1/5$ delle operazioni (appunto quelle più frequenti), questa stima calcolata rappresenterà l’80% delle risorse di elaborazione del sistema

Tavole di carico

Per stimare le prestazioni sviluppiamo tre tipi di tavole:

- **Tavola volumi** - contenente una stima delle occorrenze per entità ed associazioni
- **Tavola operazioni** - riporta tipo e frequenza per il 20% di operazioni più frequenti
- **Tavole accessi** - numero di accessi in lettura e scrittura su entità ed associazioni per il 20% di operazioni più frequenti

Rappresentazioni grafiche:

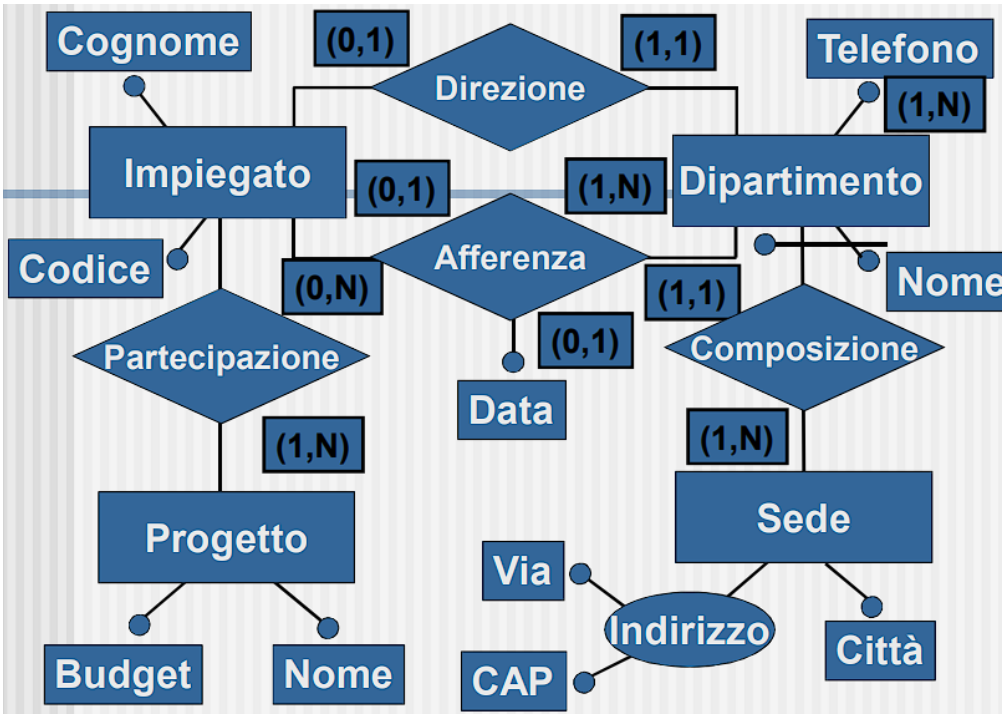


Tavola dei volumi:

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

Volume rappresenta il numero di occorrenze per quel costrutto

Il Tipo può essere E, ovvero entità oppure R, cioè relazione

Tavola delle operazioni:

Operazione	Tipo	Frequenza
Operazione 1	I	1 volta/giorno
Operazione 2	B	1 volta/mese

I: Operazione interattiva

B: Operazione batch

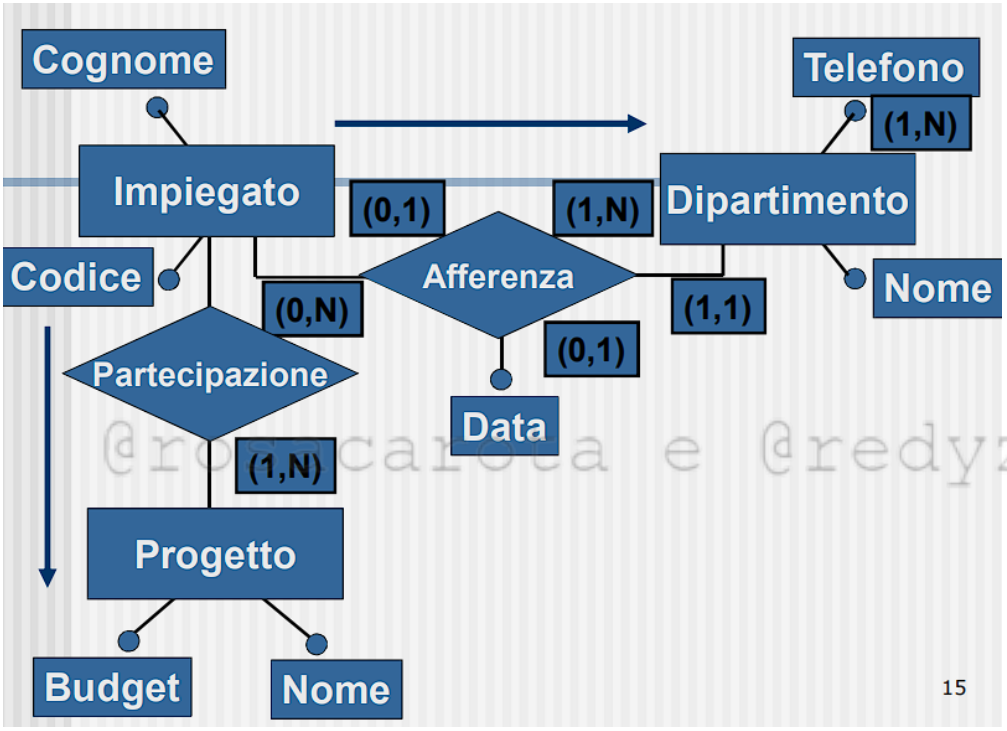
In questo caso sono state previste 10 operazioni, quindi la stima si concentra sul 20% (2) con maggiore frequenza

Esempio di valutazione di costo:

- Operazione frequente:
  - Trova tutti i dati di un impiegato, del dipartimento nel quale lavora e dei progetti ai quali partecipa

Si costruisce una tavola degli accessi basata su di uno schema di navigazione

Estraggo solo la porzione di schema concettuale interessata:



Costruisco la tavola degli accessi:

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

La tavole degli accessi descrive:

- Il concetto (entità e relazioni)
- Il costrutto dello schema da attraversare per effettuare l’operazione
- Il numero di accessi per quel costrutto
- Il tipo di accesso (in questo caso L = lettura)

In questo caso, per ogni impiegato svolgo 9 accessi in lettura

La distinzione tra accesso in lettura o scrittura va fatta, perché, generalmente, le operazioni di scrittura sono più onerose di quelle in lettura (in quanto devono essere eseguite in modo esclusivo e possono richiedere l’aggiornamento di indici)

Attività della ristrutturazione

- Analisi delle ridondanze
  - Si decide se eliminare o mantenere eventuali ridondanze presenti nello schema
- Eliminazione delle generalizzazioni

    - Tutte le generalizzazioni presenti nello schema vengono analizzate e sostituite da altri costrutti
- Partizionamento/accorpamento di entità e associazioni
    - Si decide se è opportuno partizionare concetti dello schema (entità e/o associazioni) in più concetti o, viceversa, accorpare concetti separati in un unico concetto
- Scelta degli identificatori primari
    - Si seleziona un identificatore per quelle entità che ne hanno più di uno

Analisi delle ridondanze

Una ridondanza in uno schema ER è una informazione significativa ma derivabile da altre (attraverso una serie di operazioni)

Esistono vari tipi di ridondanze:

- Attributi derivabili, occorrenza per occorrenza, da altri attributi della stessa entità (o associazione)
- Attributi derivabili da attributi di altre entità (o associazioni)
- Attributi derivabili da operazioni di conteggio di occorrenze
- Associazioni derivabili dalla composizione di altre associazioni in presenza di cicli

In questa fase di decide se eliminare le ridondanze eventualmente presenti o di mantenerle, in base al loro impatto sul numero di accessi per il 20% di operazioni più frequenti (una ridondanza potrebbe velocizzare alcune operazioni)

Ridondanze

Vantaggi:

- Semplificazione delle interrogazioni

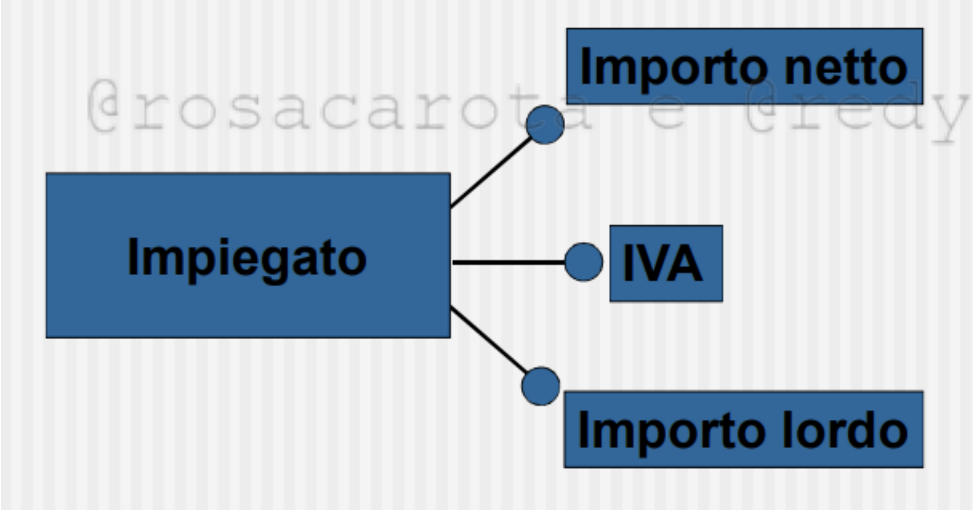
Svantaggi:

- Appesantimento degli aggiornamenti
- Maggiore occupazione di spazio

Forme di ridondanza in uno schema ER:

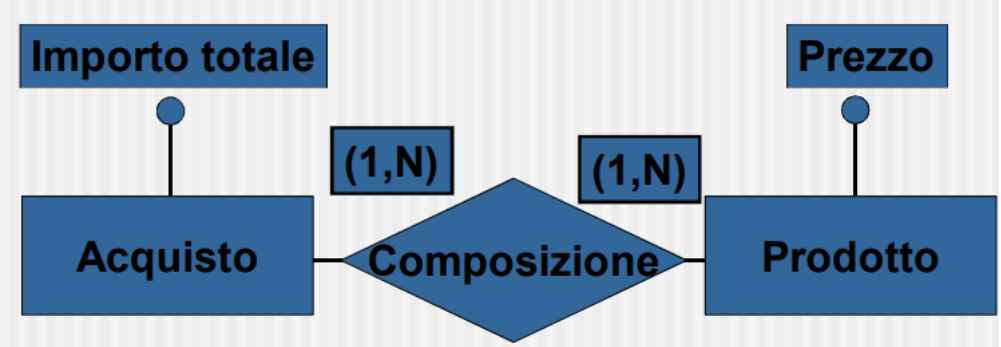
- Attributi derivabili:
  - Da altri attributi della stessa entità (o associazione)
  - Da attributi di altre entità (o associazioni)
- Associazioni derivabili dalla composizione di altre associazioni in presenza di cicli

Attributo derivabile:



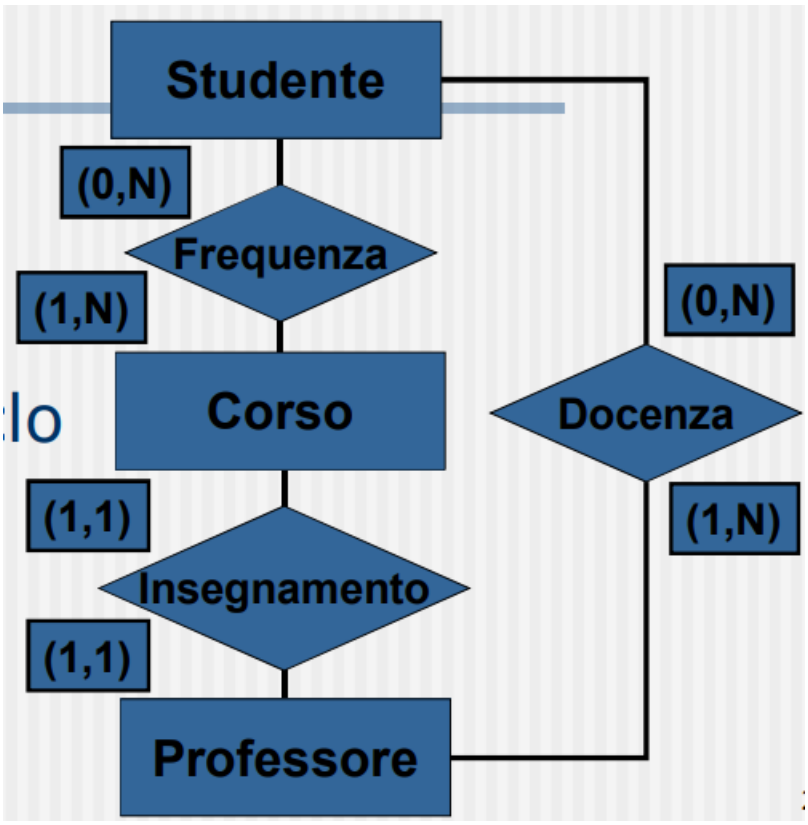
Si può ricavare l’importo lordo dall’importo netto e l’IVA

Attributo derivabile da altra entità:



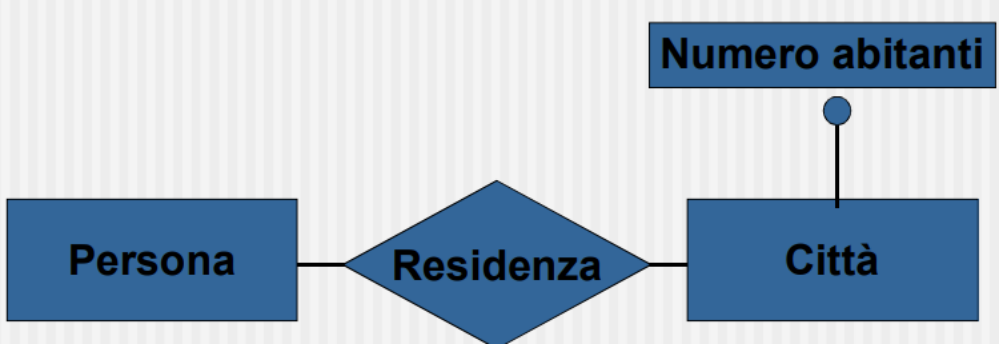
L’importo totale è ricavabile dalla somma del prezzo dei vari prodotti

Ridondanza dovuta a ciclo:



L’associazione docenza è ridondante perché ricavabile dalle altre strutture

Analisi di una ridondanza



Ipotesi di tavola dei volumi:

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

Inoltre, se una città può avere fino a milioni di abitanti, occorrono circa 3 byte per città per memorizzare il dato ridondante, totale 600 byte

Ipotesi di tavola operazioni:

Concetto	Tipo	Volume
Operazione 1	I	500 volte/giorno
Operazione 2	B	2 volte/giorno

Operazione 1: memorizza una nuova persona e relativa città di residenza

Operazione 2: stampa i dati di una città (incluso il numero di abitanti)

La prima operazione subisce effetti negativi dalla ridondanza, ad ogni inserimento devo modificare il totale degli abitanti

La seconda operazione beneficia della ridondanza, non devo calcolare il totale degli abitanti

Tavolo accessi (in presenza di ridondanza):

Operazione 1

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

Operazione 2

Concetto	Costrutto	Accessi	Tipo
Città	Entità	1	L

Tavole accessi (in assenza di ridondanza):

Operazione 1

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

Operazione 2

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

Numero totale accessi (in presenza di ridondanza):

- Costi:
  - Operazione 1: 1500 accessi in scrittura e 500 accessi in lettura al giorno
  - Operazione 2: 2 accessi in lettura
- Contiamo doppi gli accessi in scrittura
- Totale di 3502 accessi al giorno e 600 byte per il dato ridondante

Numero totale accessi (in assenza di ridondanza):

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

- Contando doppi gli accessi in scrittura si hanno 12000 accessi al giorno

Eliminazione delle generalizzazioni

Il modello relazionale non può rappresentare direttamente le generalizzazioni

Entità e associazioni sono invece direttamente rappresentabili (abbastanza per le associazioni)

Si eliminano perciò le gerarchie, sostituendole con entità e associazioni

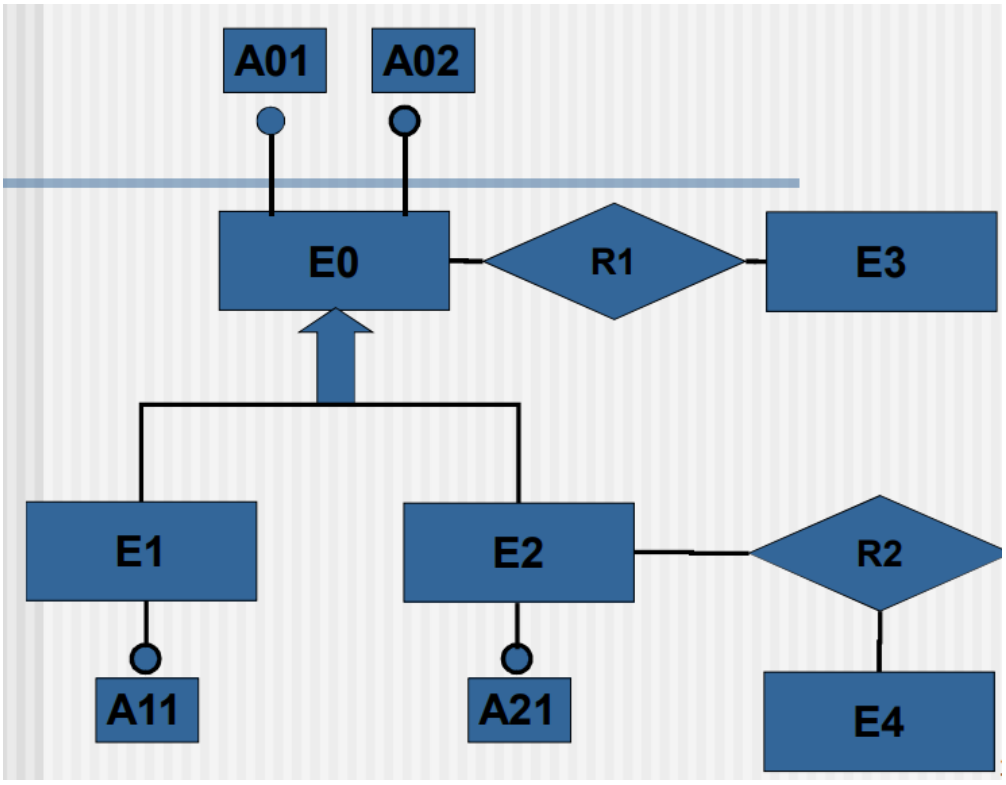
Tre possibilità:

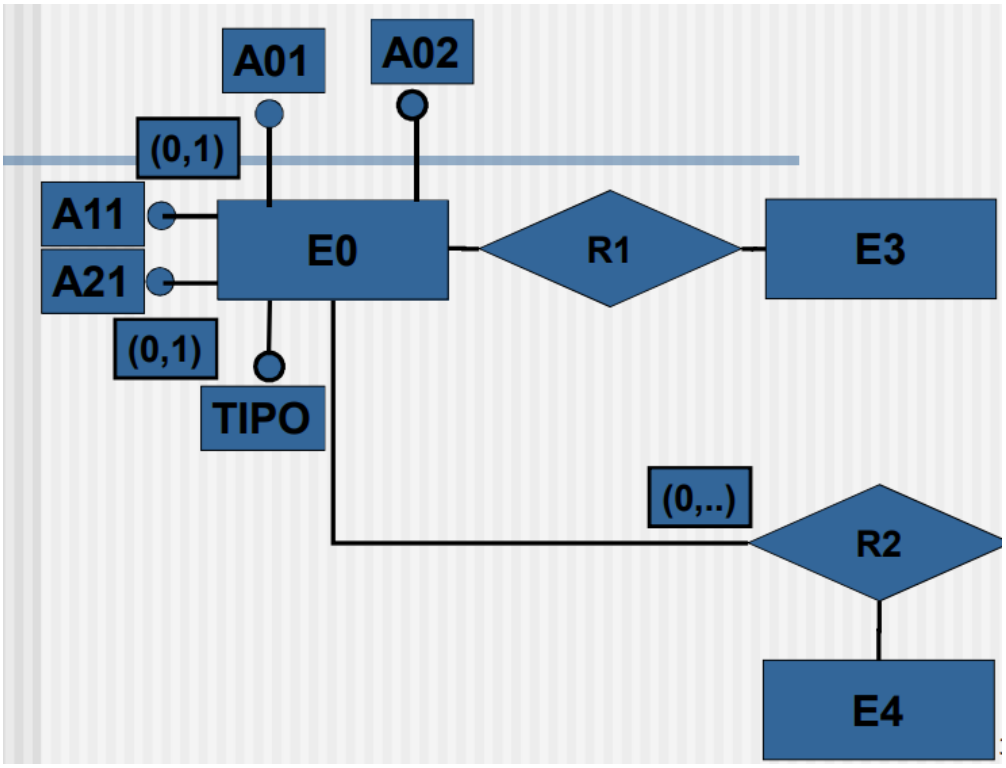
- Accorpamento delle figlie della generalizzazione nel genitore
  - Le entità E_1 ed E_2 vengono eliminate e le loro proprietà (attributi e partecipazioni ad associazioni e generalizzazioni) vengono aggiunte all'entità genitore E_0. A tale entità viene aggiunto un ulteriore attributo che serve a distinguere il "tipo" di una occorrenza di E_0, cioè se tale occorrenza apparteneva a E_1, a E_2, o, nel caso di generalizzazione non totale, a nessuna di esse
  - Si osservi che gli attributi A_{11} e A_{21} possono assumere valori nulli (perché non significativi) per alcune occorrenze di E_0, e che la relazione R_2, avrà, in ogni caso, una cardinalità minima pari a 0 sull'entità E_0 (perché le occorrenze di E_2 sono solo un sottoinsieme delle occorrenze di E_0)
- Accorpamento del genitore della generalizzazione nelle figlie
  - L'entità genitore E_0 viene eliminata e, per la proprietà dell'ereditarietà, i suoi attributi, il suo identificatore e le relazioni a cui tale entità partecipava, vengono

aggiunti alle entità figlie E_1 ed E_2. Le relazioni R_{11} e R_{12} rappresentano rispettivamente la restrizione della relazione R_1 sulle occorrenze delle entità E_1 ed E_2

- Sostituzione della generalizzazione con associazioni
  - La generalizzazione si trasforma in due associazioni uno a uno che legano rispettivamente l'entità genitore con le entità figlie E_1 ed E_2. Non ci sono trasferimenti di attributi o associazioni e le entità E_1 ed E_2 sono identificate esternamente dall'entità E_0. Nello schema ottenuto vanno aggiunti però dei vincoli: ogni occorrenza di E_0 non può partecipare contemporaneamente a R_{G1} e R_{G2}; inoltre, se la generalizzazione è totale, ogni occorrenza di E_0 deve partecipare o a un'occorrenza di R_{G1} oppure a un'occorrenza di R_{G2}

Accorpamento delle figlie della generalizzazione nel genitore:

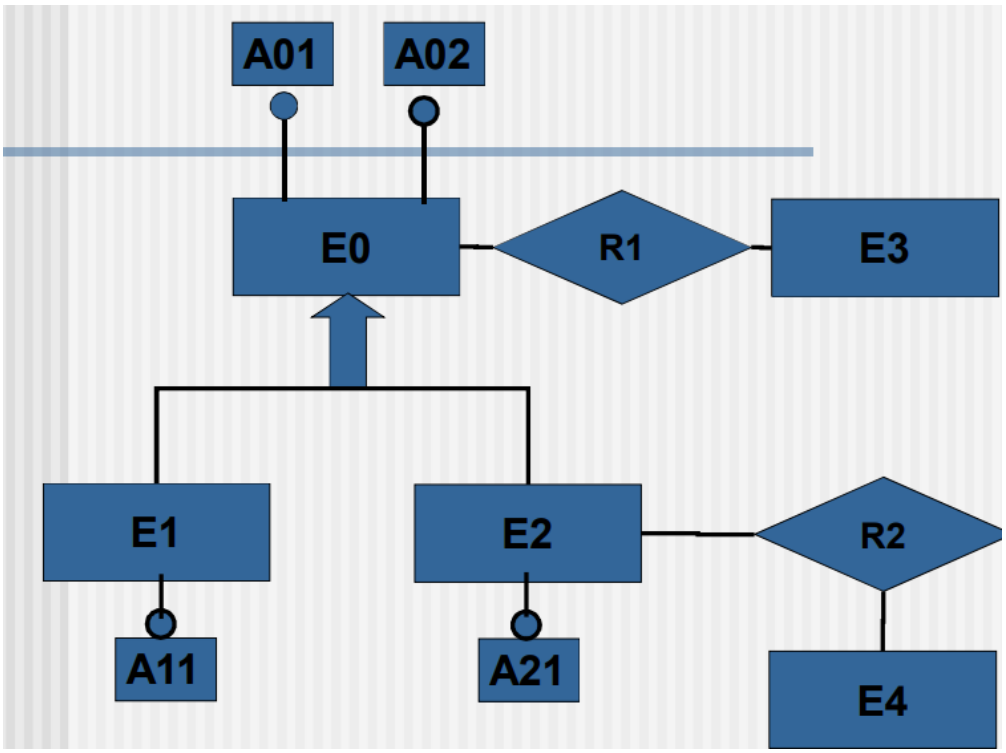


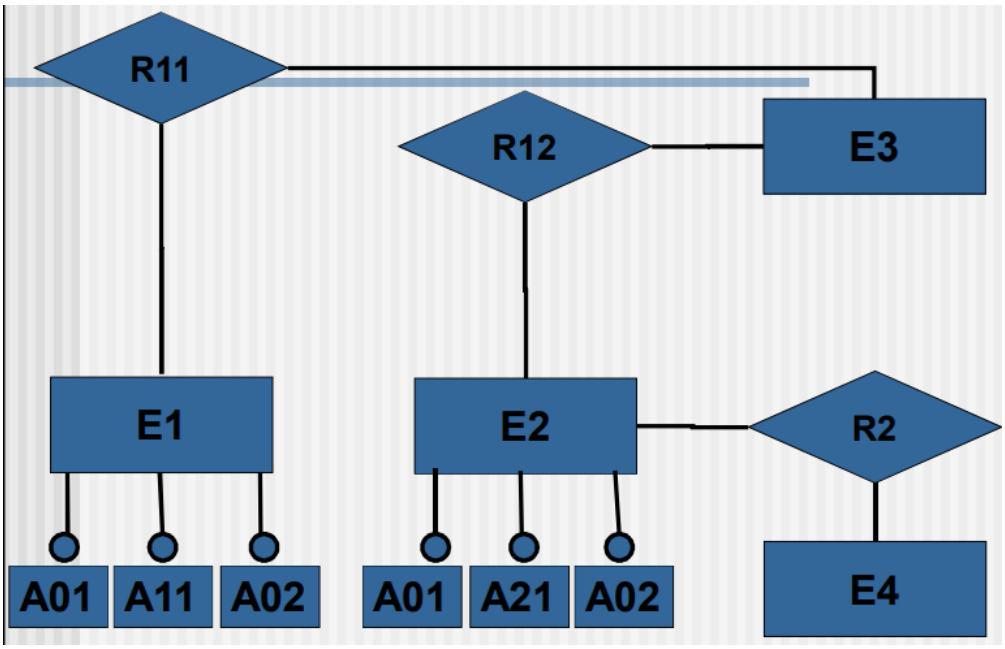


Portando A11 e A21 in E0 è stato necessario impostare il valore su opzionale (0,1), poiché dipendente dal tipo (E1 o E2), attributo aggiunto per identificare il tipo di E0

Motivo analogo per il quale la partecipazione nell'associazione R2 con E4 è diventata opzionale (0,…)

Accorpamento del genitore della generalizzazione nelle figlie:

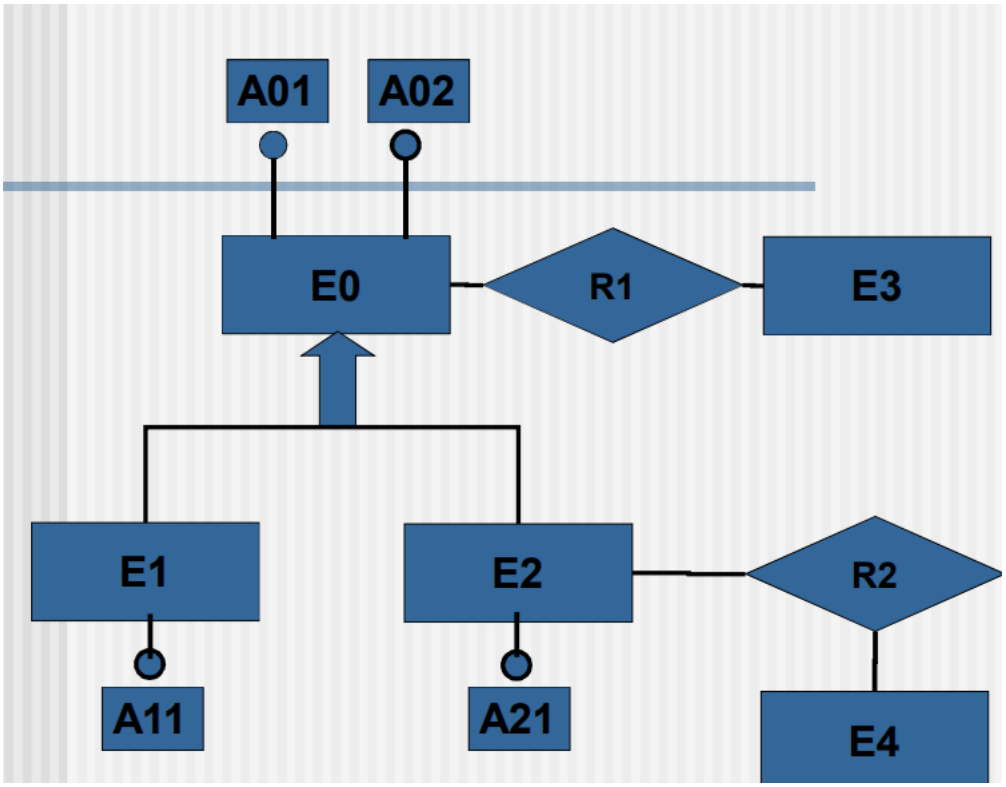


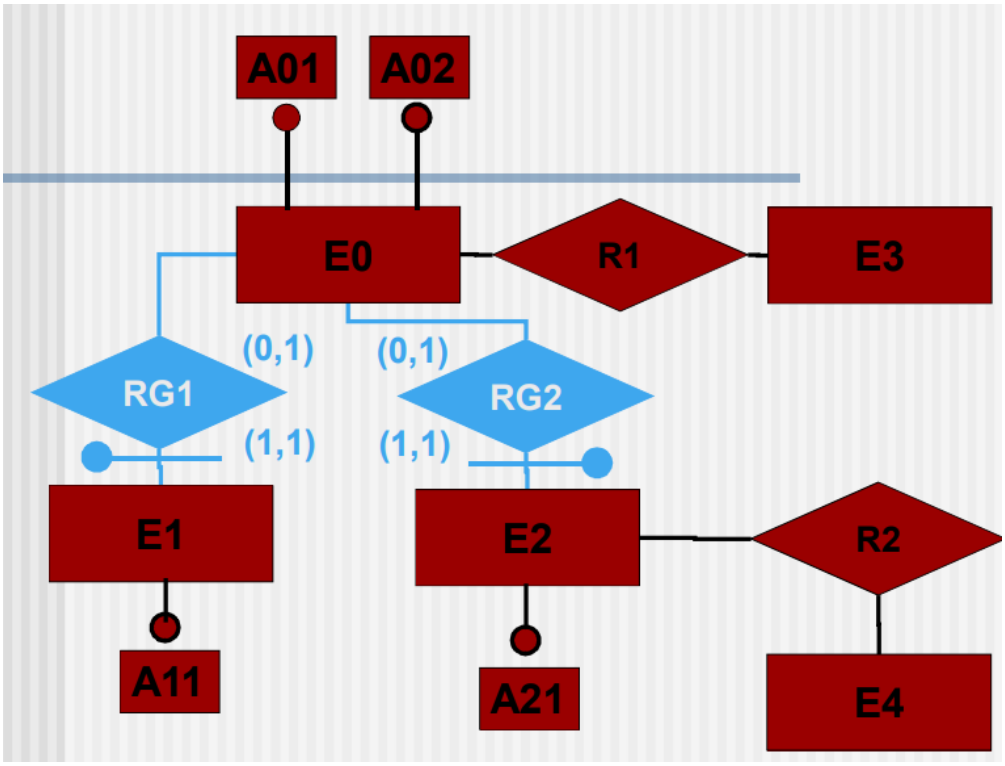


Duplichiamo gli attributi A01 e A02 del genitore nelle figlie, in più sdoppiamo l'associazione R1 in R11 e R12 per ciascuna delle entità figlie

Se la generalizzazione non è totale, questa soluzione porterebbe a perdite di informazioni

Sostituzione della generalizzazione con associazioni:





Scelte progettuali

(però non basato solo sul numero degli accessi)

Si possono anche seguire alcune semplici regole generali

Criteri di scelta

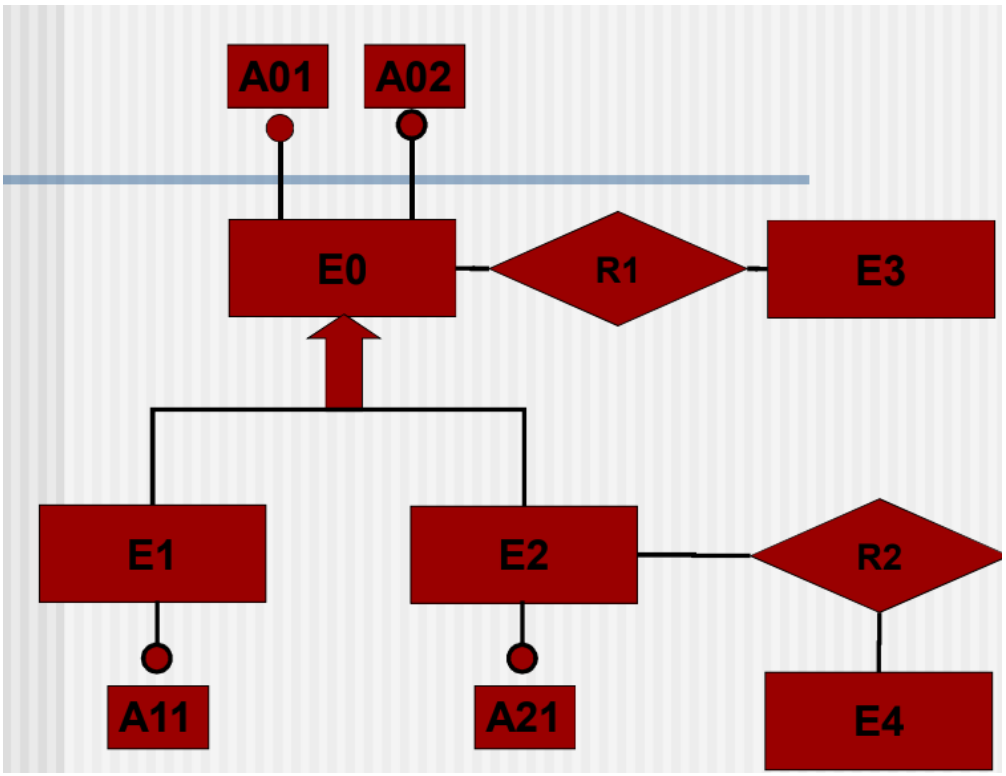
La scelta fra le alternative si può fare con metodo simile a quello visto per l'analisi delle ridondanze, considerando vantaggi e svantaggi di ognuna delle scelte possibili

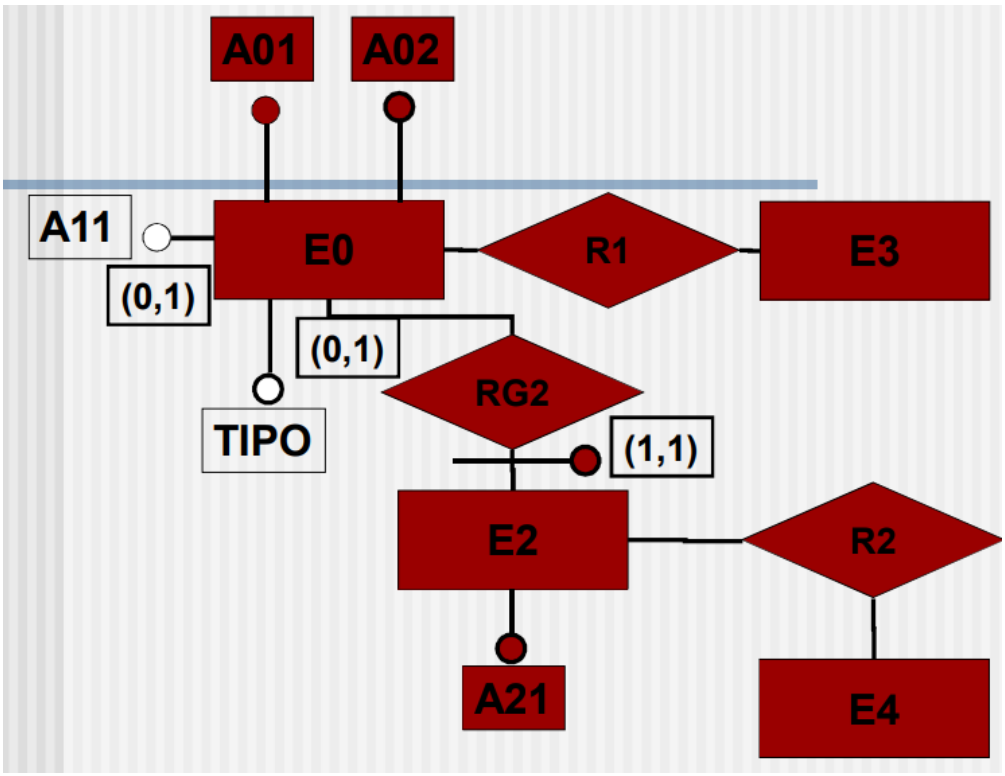
Tuttavia è possibile comunque stabilire alcune regole di carattere generale:

- La prima alternativa è conveniente quando le operazioni non fanno molta distinzione tra le occorrenze e tra gli attributi di E_0, E_1 ed E_2. In questo caso infatti, anche se abbiamo uno spreco di memoria per la presenza di valori nulli, la scelta ci assicura un numero minore di accessi rispetto alle altre nelle quali le occorrenze e gli attributi sono distribuiti tra le varie entità
- La seconda alternativa è *possibile solo se la generalizzazione è totale*, altrimenti le occorrenze di E_0 che non sono occorrenze né di E_1 né di E_2 non sarebbero rappresentate. Conviene quando ci sono operazioni che si riferiscono solo a occorrenze di E_1 oppure di E_2, e dunque fanno distinzioni tra tali entità. In questo caso abbiamo un risparmio di memoria rispetto alla prima scelta, perché, in linea di principio, gli attributi non assumono mai valori nulli
- La terza alternativa è conveniente quando la generalizzazione non è totale (sebbene ciò non sia necessario) e ci sono operazioni che si riferiscono solo a occorrenze di E_1 (E_2) oppure di E_0, e dunque fanno delle distinzioni tra entità figlia ed entità genitore. In questo caso abbiamo un risparmio di memoria rispetto alla prima scelta, per l'assenza di valori nulli, ma c'è un incremento degli accessi per mantenere la consistenza delle occorrenze rispetto ai vincoli introdotti

Sono anche possibili soluzioni "ibride", soprattutto in gerarchie a più livelli

Esempio:





Partizionamento/accorpamento di entità e associazioni

Ristrutturazioni effettuate per rendere più efficienti le operazioni in base ad un semplice principio

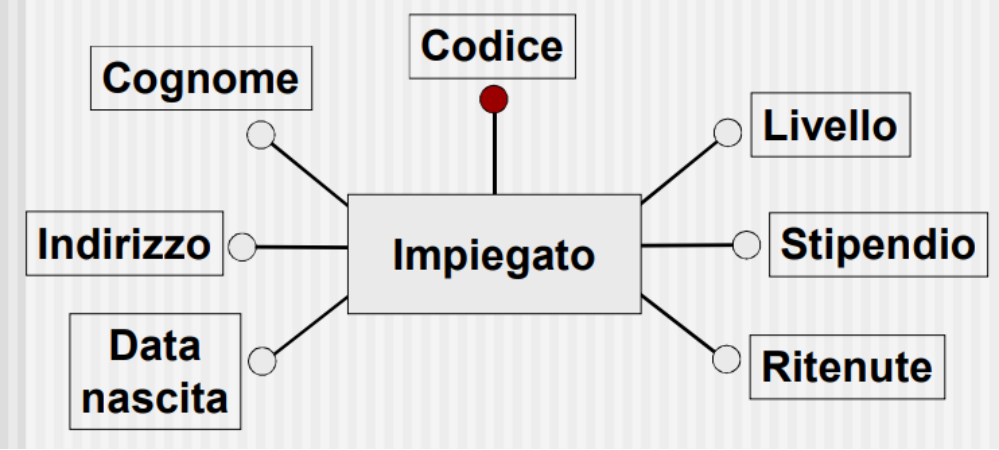
Gli accessi si riducono:

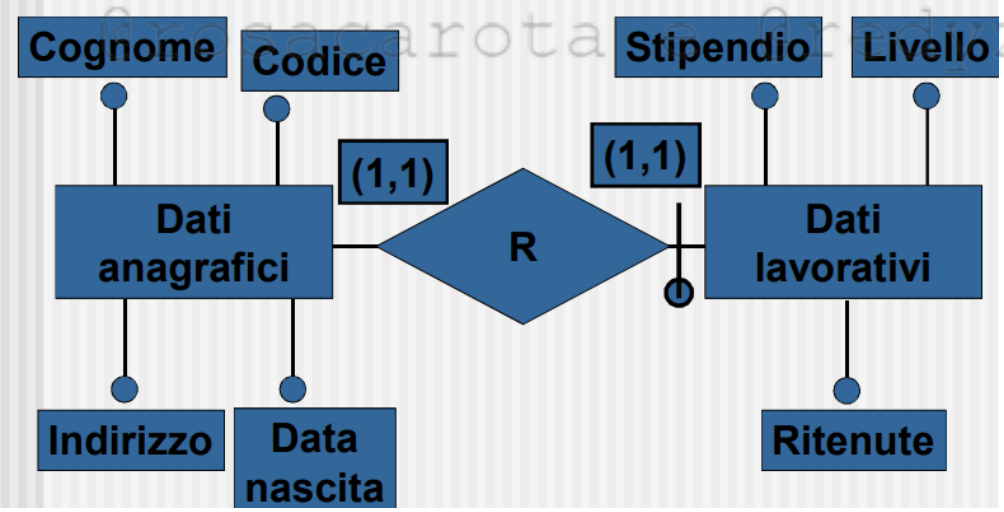
- Separando attributi di un concetto che vengono acceduti separatamente
- Raggruppando attributi di concetti diversi acceduti insieme

Casi principali:

- Partizionamento verticale di entità
- Partizionamento orizzontale di associazioni
- Eliminazione di attributi multivalore
- Accorpamento di entità/associazioni

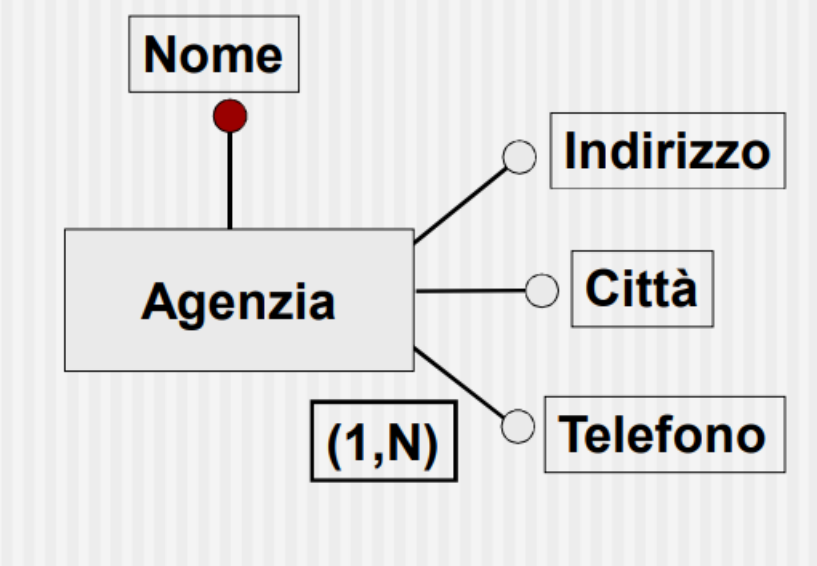
Esempio 1 (partizionamento verticale di entità):

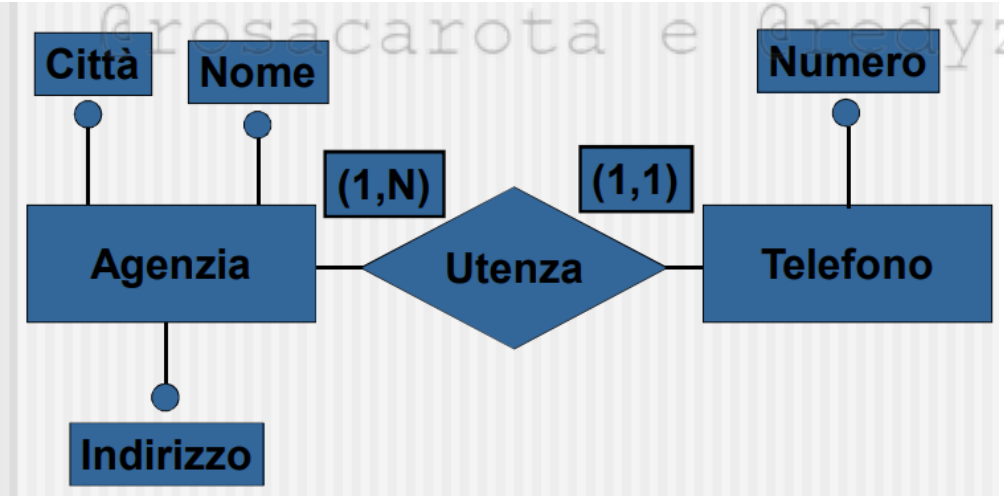




La partecipazione è obbligatoria

Esempio 2 (eliminazione attributo multivalore):



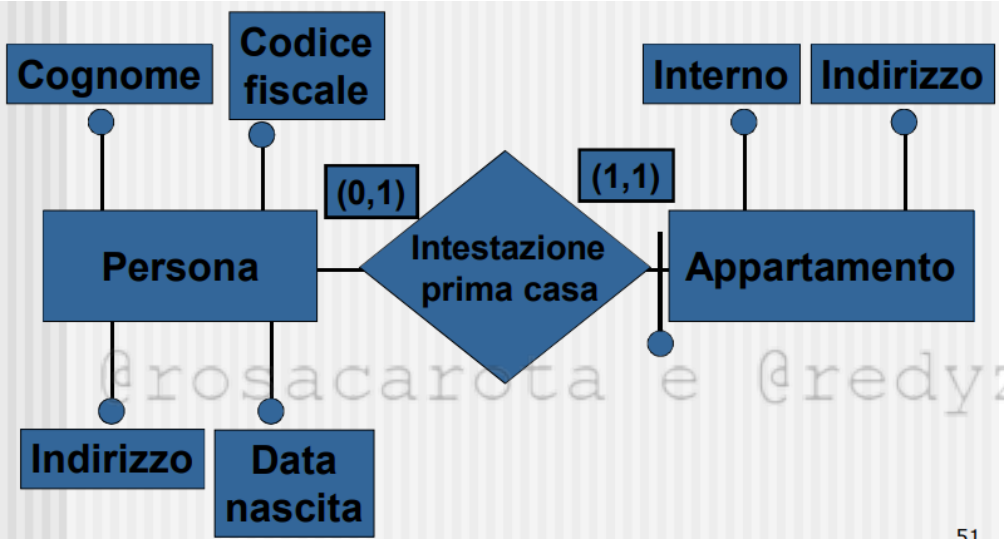


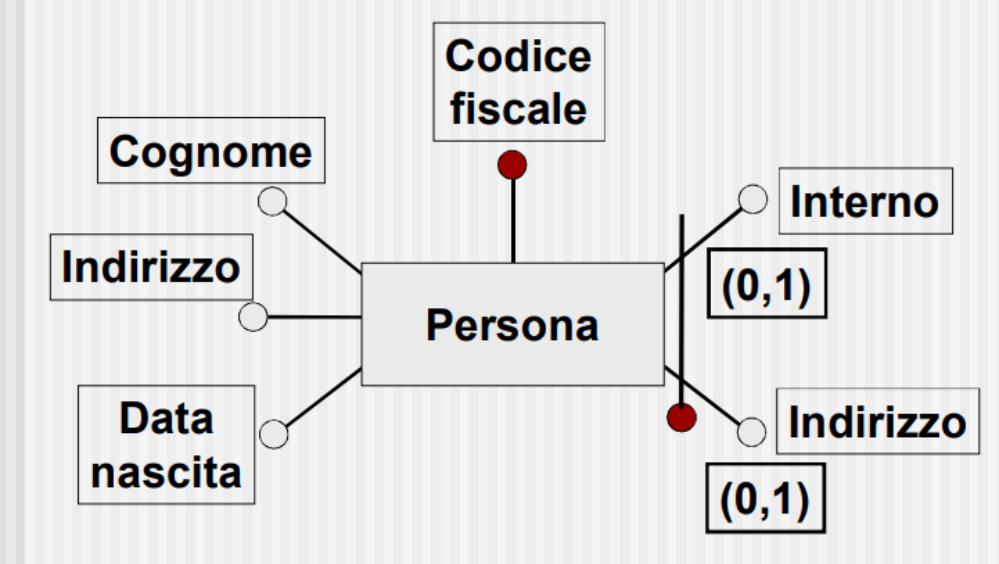
Questa ristrutturazione è necessaria poiché, come per le generalizzazioni, il modello relazionale non permette di rappresentare in maniera diretta questo tipo di attributo

I numeri di telefono diventano occorrenze di un'entità Telefono, essi saranno ottenibili tramite l'associazione Utenza

Se l'attributo fosse stato anche opzionale, allora la cardinalità minima per l'entità Agenzia sarebbe stata pari a 0

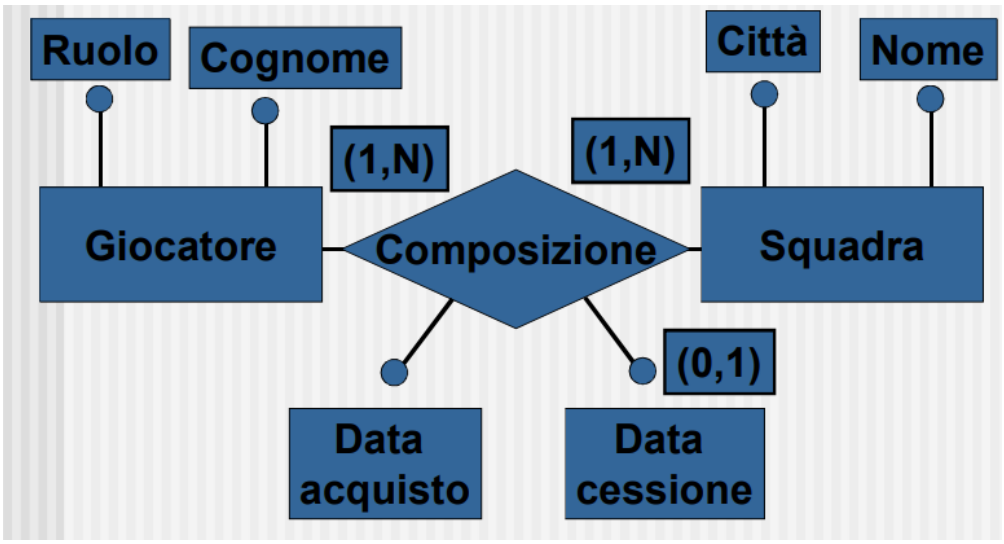
Esempio 3 (accorpamento):

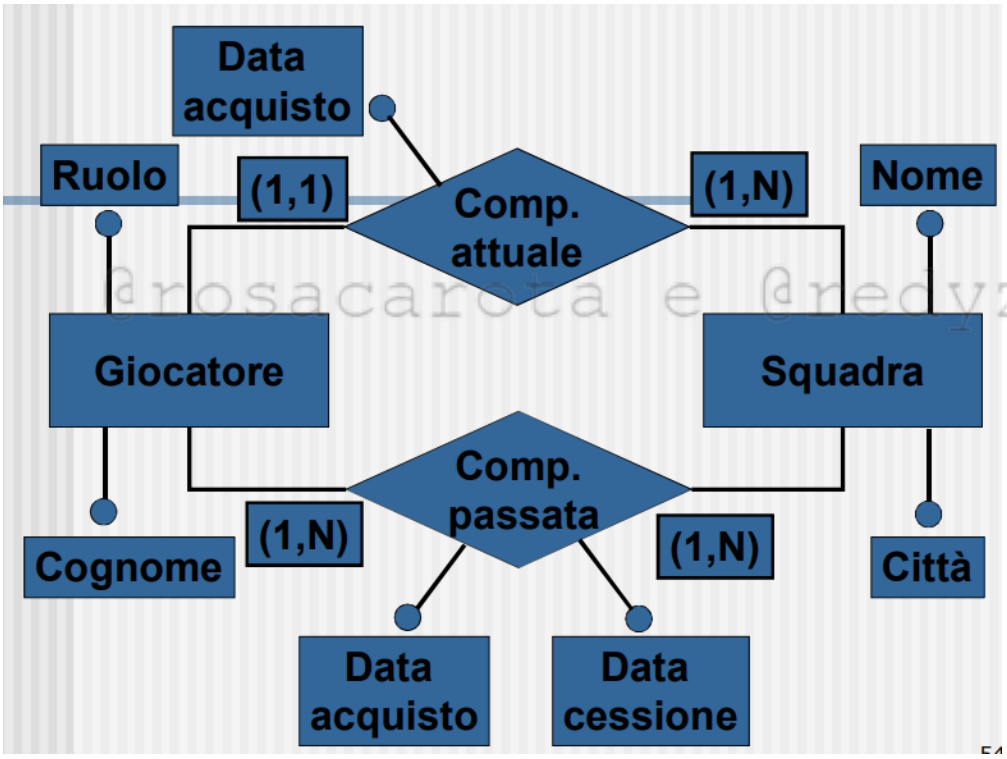




Errore (l’altro identificatore primario deve avere valore obbligatorio)

Esempio 4 (partizionamento orizzontale di associazioni):





Scelta degli identificatori primari

Operazione indispensabile per la traduzione nel modello relazionale

I sistemi di gestione di basi di dati richiedono di specificare una *chiave primaria* sulla quale vengono costruite automaticamente delle strutture ausiliarie, dette *indici*, per il reperimento efficiente dei dati

Quindi, nei casi in cui esistono entità per le quali sono stati specificati più identificatori, bisogna decidere quale di questi identificatori verrà utilizzato come chiave primaria

Criteri:

- Assenza di opzionalità
  - Gli attributi con valori nulli non possono costruire identificatori principali. Tali attributi infatti non garantiscono l'accesso a tutte le occorrenze dell'entità corrispondente
- Semplicità
  - Un identificatore composto da uno o da pochi attributi è da preferire a identificatori costituiti da molti attributi. Questo infatti garantisce che gli indici siano di dimensioni ridotte, permettendo un risparmio di memoria nella realizzazione di legami logici tra le relazioni e facilitando le operazioni di join
- Utilizzo nelle operazioni più frequenti o importanti
  - Permette di eseguire le operazioni in maniera più efficiente, poiché è possibile trarre vantaggio dagli indici creati automaticamente dal DBMS

Se nessuno degli identificatori soddisfa i requisiti visti si introducono nuovi attributi (codici) contenenti valori speciali

generati per questo scopo

Traduzione verso il modello relazionale

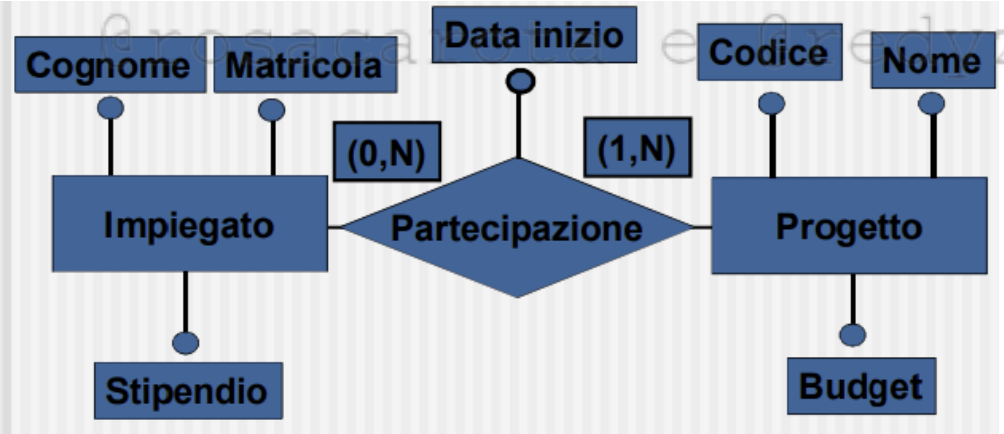
Idea di base:

- Le entità diventano relazioni sugli stessi attributi
- Le associazioni (ovvero le relazioni ER) diventano relazioni sugli identificatori delle entità coinvolte (più gli attributi propri)
- Per queste ultime è importante esaminare le informazioni di cardinalità

Associazioni molti a molti

Con vincoli di integrità referenziale fra:

- Matricola in Partecipazione è la chiave di Impiegato
- Codice in Partecipazione è la chiave di Progetto



La traduzione naturale nel modello relazionale prevede:

- Per ogni entità, una relazione con lo stesso nome avente per attributi i medesimi attributi dell’entità e per chiave il suo identificatore
- Per l’associazione, una relazione con lo stesso nome avente per attributi gli attributi dell’associazione e gli identificatori delle entità coinvolte; tali identificatori formano la chiave della relazione

Se gli attributi originali di entità o associazioni sono opzionali, i corrispondenti attributi di relazione possono assumere valori nulli

Lo schema ottenuto è il seguente:

Impiegato (Matricola, Cognome, Stipendio)

Progetto(Codice, Nome, Budget)

Partecipazione(Matricola, Codice, DataInizio)

Partecipazione conterrà la chiave di Impiegato e di Progetto

- Introduciamo una Join Table
  - Ha le relazioni di entrambe le entità che partecipano all’associazione
  - Ci saranno N ennuple con la stessa matricola di impiegato ma codice di progetto differente

Se l’associazione non è N a N, possiamo fare a meno di introdurre la Join Table

Si preferisce chiamare le chiavi esterne come il nome della tabella da cui deriviamo la chiave primaria

Nomi più espressivi per gli attributi della chiave della relazione:

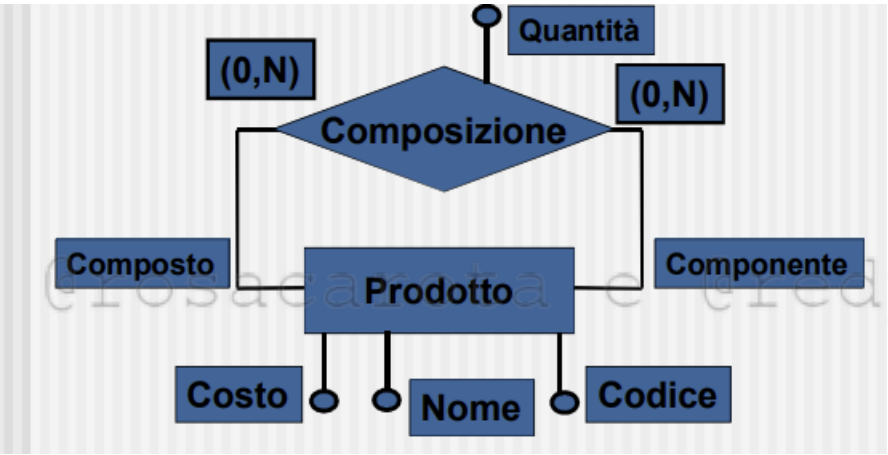
Impiegato(Matricola, Cognome, Stipendio)

Progetto(Codice, Nome, Budget)

Partecipazione(Matricola, Codice, DataInizio)

Partecipazione(**Impiegato**, **Progetto**, DataInizio)

Associazioni ricorsive



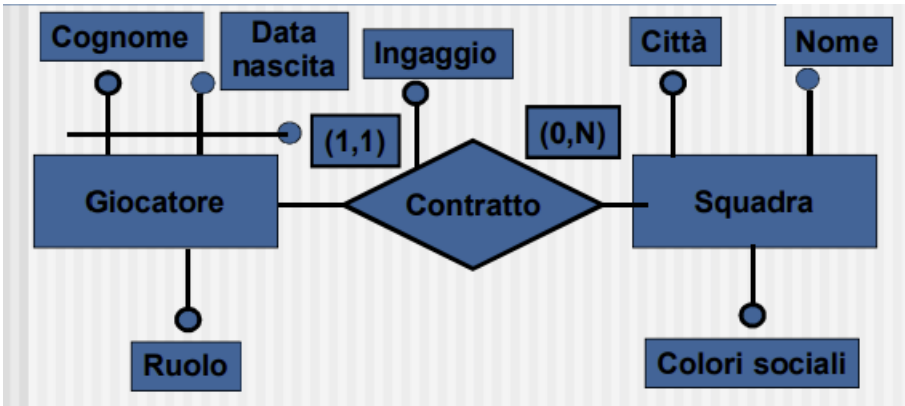
Prodotto(Codice, Nome, Costo)

Composizione(Composto, Componente, Quantità)

Ogni occorrenza di composizione punta ad un prodotto composto e uno componente (sarebbe sempre “codice” ma con nomi diversi)

Sia *Composto* che *Componente* contengono quindi codici di prodotti

Associazioni uno a molti



Seguendo la regola vista per le associazioni molti a molti la traduzione sarebbe la seguente:

Giocatore(Cognome, DataNascita, Ruolo)

Squadra(Nome, Città, ColoriSociali)

Contratto(CognomeGiocatore, DataNascG, Squadra, Ingaggio)

Si noti che nella relazione *Contratto*, la chiave è costituita solo dall'identificatore di *Giocatore* perché le cardinalità dell'associazione ci dicono che ogni giocatore ha un contratto con una sola squadra

A questo punto le relazioni *Giocatore* e *Contratto* hanno la stessa chiave ed è allora possibile fonderle in un'unica relazione (perché esiste una corrispondenza biunivoca tra le rispettive occorrenze)

Si preferisce quindi la seguente soluzione:

Giocatore(Cognome, DataNascita, Ruolo, Squadra, Ingaggio)

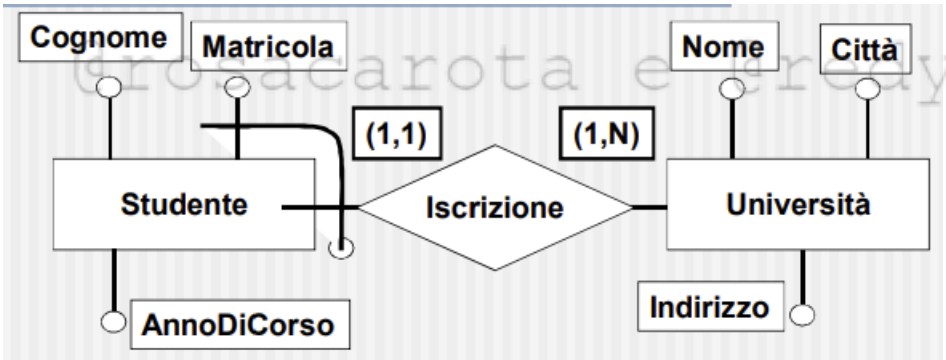
Squadra(Nome, Città, ColoriSociali)

Con vincolo di integrità referenziale fra *Squadra* in *Giocatore* e l'attributo *Nome* della relazione *Squadra*

Se la cardinalità minima dell'entità *Giocatore* fosse stata pari 0, allora *Squadra* e *Ingaggio* in *Giocatore* devono ammettere valori nulli

Entità con identificazione esterna

Le entità con identificatori esterni danno luogo a relazioni con chiavi che includono gli identificatori delle entità "identificanti"



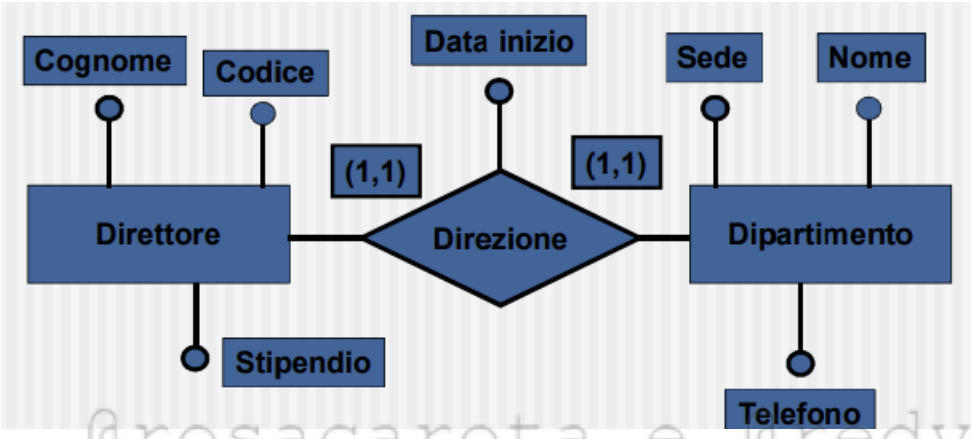
Lo schema relazionale corrispondente è il seguente:

Studente(Matricola, Università, Cognome, AnnoDiCorso)

Università(Nome, Città, Indirizzo)

Con vincolo di integrità referenziale tra l'attributo *Università* della relazione *Studente* e l'attributo *Nome* della relazione *Università*

Relazioni uno a uno



Per questo tipo di associazioni abbiamo due possibilità simmetriche e ugualmente valide:

Direttore(Codice, Cognome, Stipendio, DipartimentoDiretto, InizioDirezione)

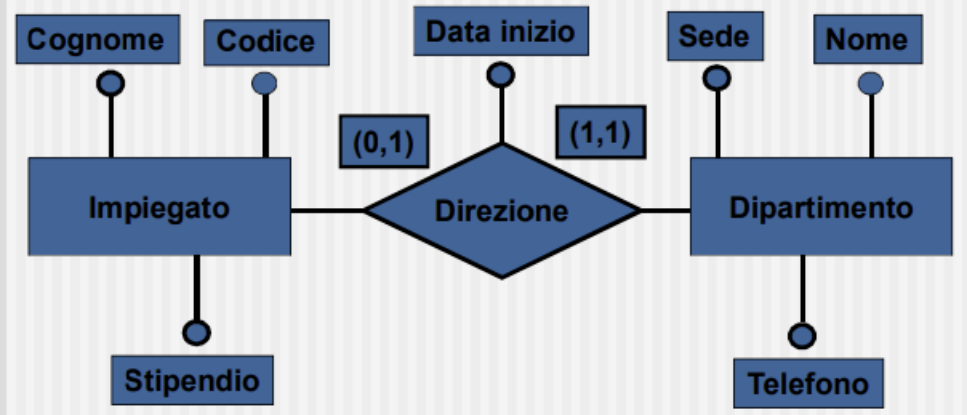
Dipartimento(Nome, Telefono, Sede)

Oppure:

Direttore(Codice, Cognome, Stipendio)

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

Una possibilità privilegiata:



Si consideri ora il caso di associazione uno a uno con partecipazione opzionale per una sola entità, in questo caso abbiamo una soluzione preferibile rispetto alle altre:

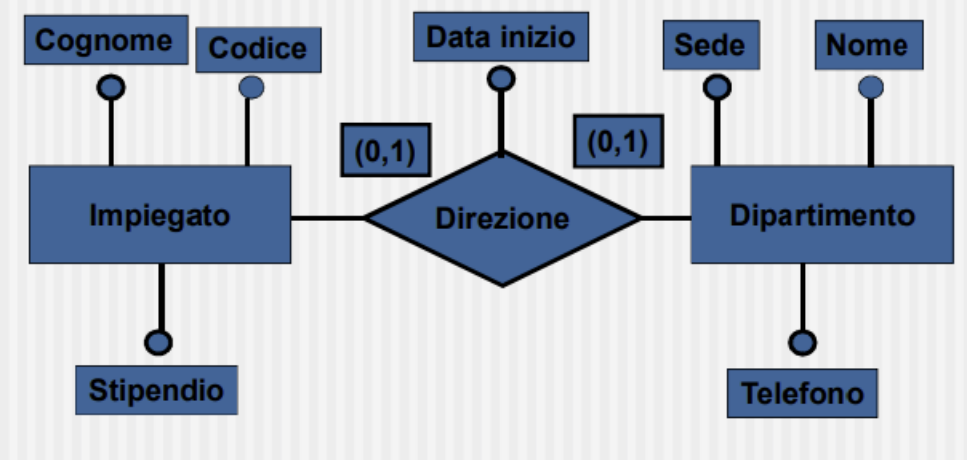
Impiegato(Codice, Cognome, Stipendio)

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

Con vincolo di integrità referenziale, senza valori nulli

Se avessi messo il Nome del Dipartimento e la Data inizio in Impiegato avrei dovuto ammettere la possibilità di valori nulli, si preferisce quindi l’altra soluzione

Un altro caso:

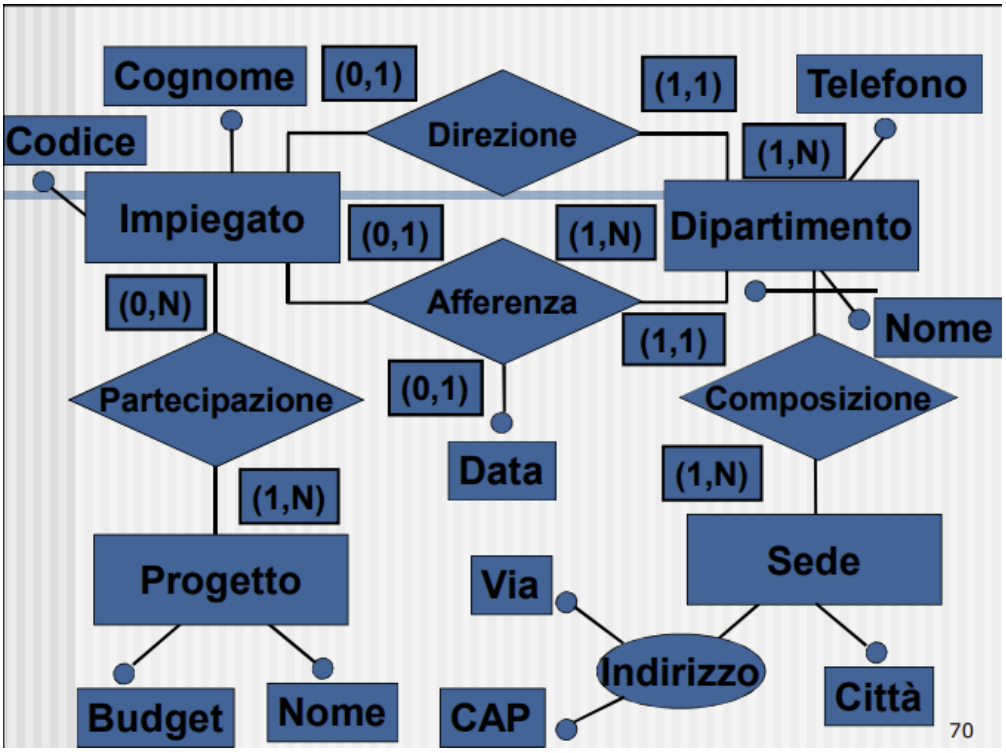


Considerando invece il caso di associazione uno a uno con partecipazione opzionale per entrambe le entità

Nel seguente caso, essendo la partecipazione opzionale per entrambi, si stabilisce in base al numero di occorrenze dove fondere l'entità, nel seguente caso sarebbe opportuno metterla in Dipartimento cosi da avere meno valori nulli (vincolo di integrità referenziale di Impiegato in Dipartimento con la chiave primaria di Impiegato)

È anche possibile utilizzare una Join Table per evitare di introdurre valori nulli, tutta via questa soluzione porta ad un aumento della complessità sulla base di dati ed è da prendere in considerazione solo se il numero di occorrenze dell'associazione è molto basso rispetto alle occorrenze che partecipano all'associazione

Uno esempio completo:



Schema finale:

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

Dipartimento(Nome, Città, Telefono, Direttore)

Sede(Città, Via, CAP)

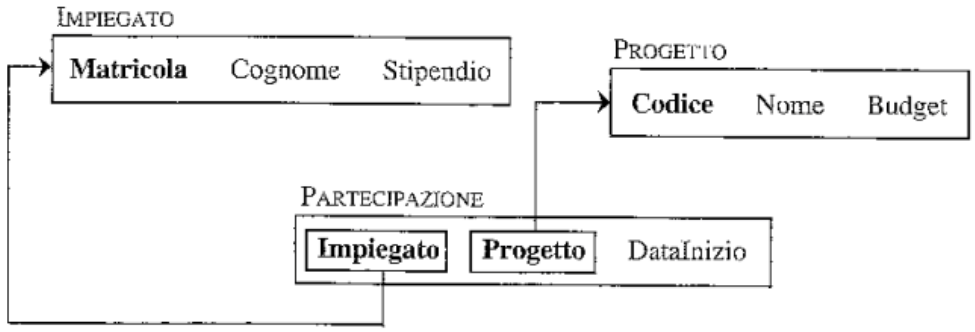
Progetto(Nome, Budget)

Partecipazione(Impiegato, Progetto)

Mapping logico

Una rappresentazione grafica di supporto è il mapping logico, nel quale:

- Le chiavi delle relazioni sono rappresentate in grassetto, le frecce indicano vincoli di integrità referenziale e la presenza di asterischi sui nomi di attributo indica la possibilità di avere valori nulli



Risulta utile per individuare immediatamente i **cammini di join**, ovvero le operazioni di join necessarie per ricostruire l'informazione rappresentata dalle associazioni originarie

- Nell'esempio, le informazioni sui progetti ai quali gli impiegati partecipano, attraverso il join tra *Impiegato*, *Partecipazione* e *Progetto*

Strumenti di supporto

Esistono sul mercato prodotti CASE che forniscono un supporto a tutte le fasi della progettazione di basi di dati

Essi usano notazioni UML-like