

16-11-2022

Importazione dati

```
mysql> LOAD DATA INFILE 'WARE.TXT'
-> INTO TABLE WAREHOUSE
-> FIELDS TERMINATED BY '\t'
-> LINES TERMINATED BY '\r\n';
Query OK, 3 rows affected (0.00 sec)
Records: 3 Deleted: 0 Skipped: 0 Warnings: 0
mysql> SELECT *
-> FROM WAREHOUSE;
```

WAREHOUSE	LOCATION
1	East
2	West
3	North

3 rows in set (0.00 sec)

Data in the WAREHOUSE table

FIGURE 8-4 Importing text data into the WAREHOUSE table

- L'istruzione `LOAD DATA` legge un file e carica i dati in tabella.
- `Fields terminated by`: i campi sono terminati da...
 - `\r` cambia riga ma non va all'inizio del testo
 - `\t` è la tabulazione
- ogni riga termina con `/r` `/n` cioè "a capo" e questi carattere variano in base al sistema operativo. Quindi i simboli di Windows non saranno uguali a quelli di Mac o Linux.

```
LOAD DATA [LOW_PRIORITY | CONCURRENT] [LOCAL] INFILE 'file_name.txt'
[REPLACE | IGNORE]
INTO TABLE tbl_name
[FIELDS
    [TERMINATED BY '\t']
    [[OPTIONALLY] ENCLOSED BY '']
    [ESCAPED BY '\\']
]
[LINES
    [STARTING BY '']
    [TERMINATED BY '\n']
]
[IGNORE number LINES]
[(col_name,...)]
```

- `LOAD DATA INFILE` legge il file txt dal server.

- `LOAD DATA LOCAL INFILE` legge il file txt dal locale.

```
mysql> LOAD DATA INFILE '/tmp/test.txt'
-> INTO TABLE test LINES STARTING BY
'yyy';
```

- Quindi un file contenente
`yyy"Row",1`
`blablabla yyy"Row",2`
- Può essere letto e verrà caricato come `("row",1), ("row",2)`

Explain

Questa istruzione serve per vedere come il database esegue query.

- Solitamente il database esegue un processo di ottimizzazione di quello che si scrive.
- Quindi la **query di input si modifica** e si crea una **query equivalente ottimizzata**.
- Essa è fatta in base alle regole viste e in base agli indici creati, alle chiavi del database, chiavi esterne e altri fattori

Quanto affermato sopra implica che quello che si scrive a livello di codice non è la query che effettivamente viene eseguita.

Il **QUERY PLAN** è quello che mostra il procedimento dell'esecuzione delle query del database

In mysql si usa **EXPLAIN** che consente di capire le performance di una query e mostra quali indici effettivamente la query sta usando.

```
mysql> explain select * from event where year(event.date) < '2003';
+-----+-----+-----+-----+-----+-----+-----+
| table | type | possible_keys | key | key_len | ref | rows | Extra |
+-----+-----+-----+-----+-----+-----+-----+
| event | ALL | NULL | NULL | NULL | NULL | 6 | Using where |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

```
mysql> explain select * from event where event.date < '2003-01-01';
+-----+-----+-----+-----+-----+-----+-----+
| table | type | possible_keys | key | key_len | ref | rows | Extra |
+-----+-----+-----+-----+-----+-----+-----+
| event | ALL | Index_2 | NULL | NULL | NULL | 6 | Using where |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

- La query soprastante è più lenta perchè non si sta usando nessun indice. In questo caso si scorre tutta la tabella e vede se la condizione è vera.
- Nel caso della query sottostante si usano gli indici e la la ricerca risulta più veloce.

PROGETTAZIONE

Per progettare una base di dati serve la **DOCUMENTAZIONE** (*anche per l'approvazione del cliente stesso*).

Si hanno diversi tipi di progettazione:

- Progettazione **CONCETTUALE** : si ha il passaggio idee cliente → idee strutturate in maniera chiara
- Progettazione **LOGICA**: schemi derivati da sopra, analizzo performance, produzione traduzione nel modello relazionale (se scelto) o modello basi di dati che si vuole usare
- Progettazione **FISICA**: realizzazione DB.

Tutto ciò è supportato dalla **Normalizzazione** del database, cioè dalla mancanza di ridondanze ecc...)

I passi da seguire sono i seguenti:

- **raccolta e analisi dei requisiti**, tramite interazione con cliente e con chi userà il sistema. In questa fase si definiscono le proprietà e le funzioni del sistema;
- **Progettazione**;
- **Implementazione**
- **Validazione**: questo passo può richiedere diverse interazioni con cliente finché si ottiene prodotto stabile
- **Funzionamento e Manutenzione**

METODOLOGIE DI PROGETTAZIONE

- Dal principio dell'ingegneria: separa il maniera netta le decisioni relative a cosa rappresentare in una base di dati (prima fase), da quelle relative a come farlo (seconda fase).

Distinguiamo:

- la progettazione **concettuale**;
- la progettazione **logica**;
- la progettazione **fisica**.

Progettazione Concettuale

Scopo della progettazione concettuale è:

- **tradurre** la descrizione informale della realtà, risultato dell'analisi dei requisiti del DB (*tipicamente sotto forma di documenti e moduli di vario genere*), in uno **schema formale**

e **completo** che dovrà essere **indipendente** dai criteri di rappresentazione del DBMS usato.

- La descrizione formale fa riferimento ad un modello concettuale, cioè un insieme di concetti e notazioni standard adatti alla rappresentazione del dominio applicativo.

L'obiettivo è **schematizzare** le richieste del cliente in una forma a *metà strada* fra Database e la realtà stessa.

Lo schema concettuale **NON** tiene conto della parte di **implementazione del database** a livello di codice.

Progettazione Logica

- Consiste nella **traduzione dello schema concettuale** in termini di un determinato **modello logico** (*ad esempio il modello relazionale*) di dati usato dal DBMS che si intende utilizzare. **Il risultato è lo schema logico**
- **Include anche l'ottimizzazione** della rappresentazione in funzione delle operazioni eseguite (es. **normalizzazione**).

L'obiettivo è avere lo schema logico del database ottimizzato *in base alle operazioni che si faranno su di esso*. (ottimizzazione dati in funzione delle operazioni).

Si prendono le scelte relative alla rappresentazione. (*quali tabelle, quali chiavi?*)

Progettazione Fisica

- Si **completa lo schema logico** (*effettiva implementazione del DB*) con la specifica dei parametri fisici di memorizzazione dei dati. Si produce lo schema fisico che fa riferimento ad un certo modello fisico dei dati che dipende dal DBMS scelto
- In questo caso si è fortemente dipendenti dal DB utilizzato.
Quindi in queste 3 fasi la dipendenza al DB non conta(fase1) poi conta poco (fase 2) poi conta sempre (fase 3)

PROGETTAZIONE CONCETTUALE

In questa fase (ANALISI DEI REQUISITI) si deve:

- **individuare le proprietà e le funzionalità** del sistema;
- **produrre una descrizione dei dati** coinvolti e delle **operazioni** su di essi;
- **individuare** (*in linea di massima, quindi approssimativa*) i **requisiti software** ed **hardware** del sistema.

Inoltre si deve produrre (OUTPUT DELL'ANALISI):

- **studio di fattibilità** che stimi:
 - i **costi** in termini di budget, di impegno del personale;
 - le **inefficienze temporanee** dovute al cambio di sistema e di modalità di lavoro;

- i **benefici** in termini di *riduzione dei tempi* di lavoro o migliore efficienza dei processi aziendali;
- **piano di sviluppo** del sistema con priorità e **tempi di realizzazione**.

Procedure per l'analisi

- Per ogni settore aziendale in esame si procede con i seguenti passi:
 - si analizza il sistema informativo esistente, si intervistano i responsabili del settore;
 - si produce una prima versione dei requisiti in linguaggio naturale, raggruppando **frasi descrittive** relative a categorie diverse di **dati** e di **operazioni**.
- si analizzano le descrizioni per eliminare ambiguità provocate da:
 - pluralismo di percezione
 - incompletezze di descrizione
- **ambiguità del tipo**:
 - Omonimie
 - Sinonimie
 - conflitti di descrizione
 - similitudini

Vedo quali sono i bisogni e il sistema attualmente presente. Queste info vengono schematizzate togliendo ambiguità sopra descritte.

- si ricontrollano insieme ai responsabili di settore **le frasi** relative alle varie categorie di dati e alle operazioni che li coinvolgono
- si costruisce a partire dalle frasi verificate un **glossario di termini**;
 - Il glossario tipicamente per ogni termine contiene: la descrizione, l'elenco dei sinonimi e l'elenco dei termini collegati.
- si verifica la **completezza**
 - tutti gli aspetti importanti sono stati considerati
- si verifica la **consistenza** delle specifiche vedendo se:
 - tutti i termini sono stati definiti;
 - tutti i termini compaiono in delle operazioni;
 - le operazioni fanno riferimento a termini definiti.

il GLOSSARIO serve per comprendere meglio quello che il cliente chiede.

Esempio: “frasi descrittive di un magazzino”

- il magazzino è composto da scaffali
- i fornitori forniscono prodotti
- i clienti ordinano prodotti
- gli scaffali contengono prodotti
- gli operai sono addetti agli scaffali

TERMINE	DESCRIZIONE	SINONIMI	LEGAME
fornitore	p. iva, denom., indirizzo, num. tel.		prodotto
cliente	p. iva, denom., indirizzo, num. tel.	acquirente	prodotto
prodotto	codice, nome, genere....	articolo voce	fornitore scaffale cliente
Scaffale	supporto numerato	ripiano (incertezza)	operaio prodotto
operaio	dati anagrafici, matricola, qualifica	addetto magazziniere	scaffale

Si prendono i requisiti e cerco di togliere i sinonimi per semplificare.

Il modello Entità-Relazione

- Il modello **Entità-Relazione** (E-R) è un **modello concettuale di dati** che contiene alcuni costrutti atti a descrivere la realtà in maniera semplice, indipendente dalla organizzazione dei dati nel computer.
- I costrutti sono:
 - Entità
 - Associazioni (Relazioni)
 - Proprietà (Attributi)
 - Cardinalità
 - Identificatori
 - Generalizzazioni
 - Sottoinsiemi

I **costrutti** descrivono le operazioni. Una schematizzazione cerca di modellare la realtà in forma schematica.

$E - R$ serve per rappresentare **CONOSCENZA ASTRATTA** e **CONOSCENZA CONCRETA**:

- **CONOSCENZA ASTRATTA**
 - entità
 - associazione
 - Proprietà
- **CONOSCENZA CONCRETA**
 - istanza di entità
 - istanza di associazione
 - proprietà delle istanze

Concetti nello schema E-R:

- **Entità** sono classi di oggetti (*cose, persone*) che hanno proprietà comuni ai fini dell'applicazione di interesse che si intende modellare. (cose, persone, docenti, corsi, automobili, scaffali, prodotti ecc..)
- **Istanza di Entità**: cosa (*oggetto, persona*) che esiste di per sé nel dominio applicativo, della quale si vogliono registrare fatti specifici e che può essere chiaramente identificata in modo da poterla distinguere dalle altre.

- | | |
|------------------|-----------------------------|
| • Entità: | • Istanza di Entità: |
| – Docente | – il docente Ferro |
| – Corso | – il corso Basi di Dati |
| – Automobile | – l'auto AB111XY |
| – Studente | – lo studente 567345 |
| – Volo | – il volo AZ3313 |
| – Percorso | – il percorso CT-PA |

- Le **Associazioni** (o **Relazioni**) rappresentano legami logici fra due o più entità dell'applicazione.
 - Possono esserci più relazioni fra le stesse entità (*associazioni ricorsive*).
- **Istanza di Associazione**: fatto che descrive un'azione o una situazione e che stabilisce legami tra istanze di entità (associa, mette in relazione)

Esempi

- Associazione:
 - Insegna
 - Appartiene
 - Ordina
 - Lavora
 - Viaggia sulla tratta
- Istanza di associazione:
 - Ferro insegna Basi di dati
 - Sicilia appartiene all'Italia
 - La ditta Rossi ordina PC
 - Bianchi lavora al magazzino 4
 - il TIR 542 viaggia sulla tratta CT-PA

- Dette anche **attributi delle istanze**: sono *fatti* che **descrivono le caratteristiche delle istanze** di entità e le caratteristiche delle istanze di associazione
- Le proprietà delle istanze assumono **VALORI**.

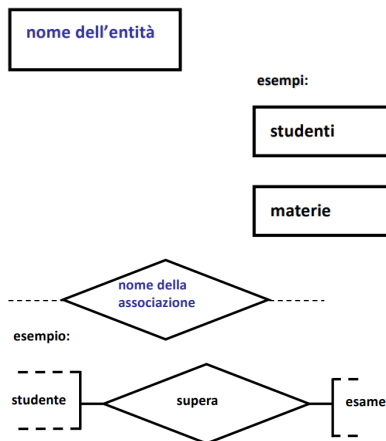
Proprietà Istanze delle Entità e delle Associazioni

- Proprietà di istanze di entità:
 - Ferro ha nome Alfredo
 - Il recapito della ditta Rossi è via Etnea 22
 - Formazione Analitica 1 si tiene al primo anno

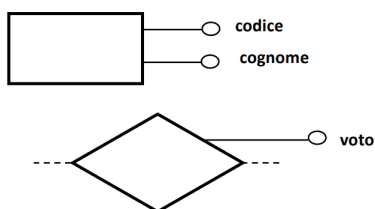
Esempi:

- Proprietà di istanze di associazione
 - Ferro insegna basi di dati *nell'anno 1995*
 - Bianchi ha lavorato *3 ore* al magazzino 4
 - La ditta Rossi ordina 15 PC
 - Pappalardo supera Formazione Analitica 1 con 27
- Il modello E-R **usa simboli grafici** per favorire l'immediatezza della comprensione
- Gli schemi E-R sono **schemi grafici**;
- Si possono **aggiungere frasi di commento** per rappresentare le **caratteristiche non modellabili**, detti anche **vincoli non esprimibili** (*ad esempio, vincoli complessi*).

Rappresentazione entità e associazioni:



Notazioni E-R: proprietà e schema scheletro

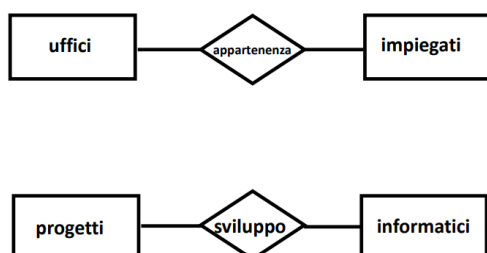


Notazioni E-R: schemi scheletro

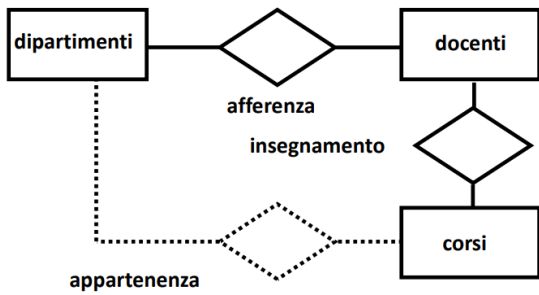
- Gli **schemi scheletro** descrivono una prima struttura di massima dello schema senza indicazioni sul TIPO delle entità e delle associazioni
- Descrivono in generale i collegamenti tra le entità di interesse e le associazioni che le legano

Lo schema scheletro è la **più semplice struttura**(o *schema*) che serve per indicare il **punto di partenza della progettazione**.

Schemi scheletro (esempi)

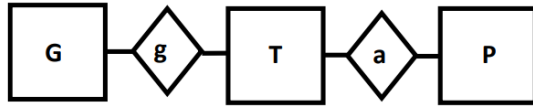


A volte capitano **ridondanze**, cioè **cicli di associazioni**.

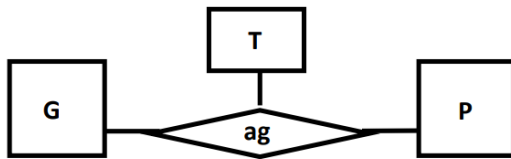


i Guidatori guidano TIR,

i TIR sono assegnati a Percorsi

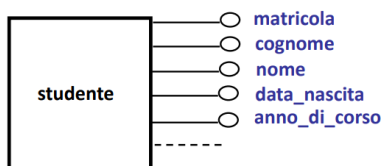


i Guidatori guidano TIR su Percorsi

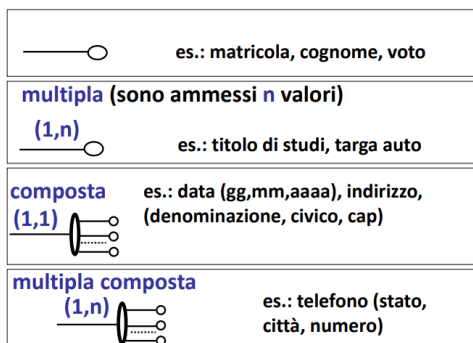


- Gli schemi scheletro **descrivono in generale i collegamenti tra le entità e le associazioni**;
- Le entità e le associazioni devono essere descritte attraverso l'aggregazione di proprietà.

Identificazione delle proprietà



Proprietà



Il simbolo **(n,m)** esprime la **cardinalità** minima e massima della proprietà

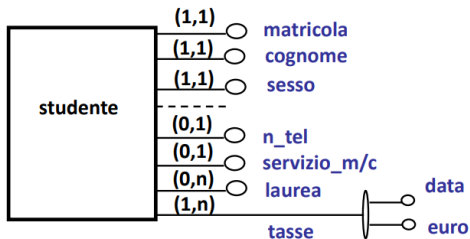
cardinalità (0,m) è una **proprietà opzionale** e può assumere al massimo **m** valori

Esempio:

Proprietà **opzionale** (è ammessa la "non esistenza del valore")

$(0,n)$ $(0,1)$ es.: tel., qualifica, targa

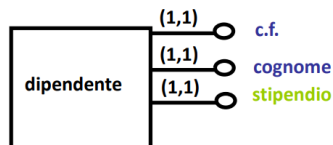
Esempio di specifica delle proprietà



(1, 1) posso anche omettere la proprietà e verrà considerata semplicemente 1.

Proprietà calcolate

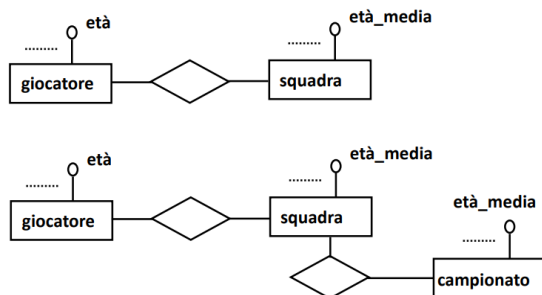
Proprietà **calcolata (derivata)**: il valore è calcolato con un algoritmo, in alternativa la proprietà è **esplicita**.



NB: la regola di calcolo va espressa a parte, usando un qualche linguaggio di specifica:
stipendio = salario_giornaliero*presenze

Proprietà di sintesi

- valori medi, max e min ecc.




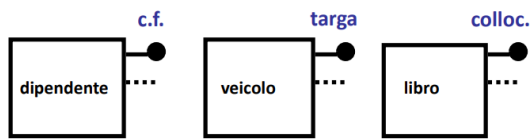
Le proprietà che **identificano le istanze delle varie entità** sono dette **PROPRIETA' CHIAVE**:

Una proprietà (attributo) **chiave** identifica in modo univoco la **singola istanza** di entità

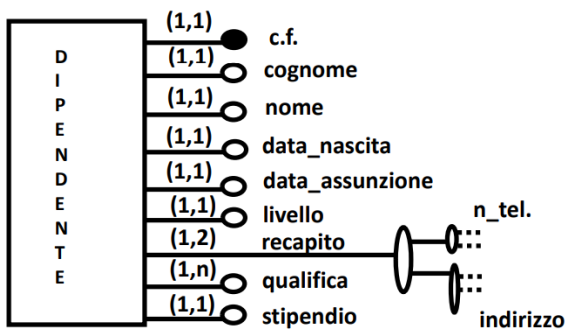
(o di associazione):

- obbligatoria, unica, esplicita
- può essere composta
- in generale non è modificabile

Simbolo = 

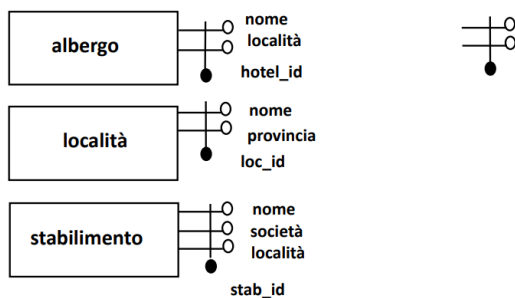


In questo caso un'entità può avere più chiavi.

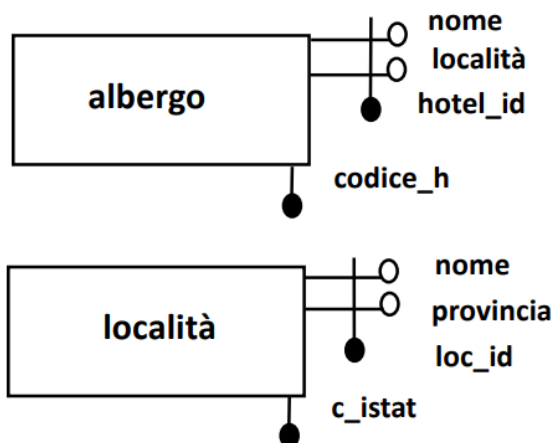


La chiave con più attributi si indica con chiave composta:

La chiave di un'entità può essere composta



Chiave alternativa: si possono avere più chiavi nelle stesse entità. Solamente in seguito si sceglieranno le chiavi primarie primarie:



In questo caso la chiave alternativa è `codice_h`.

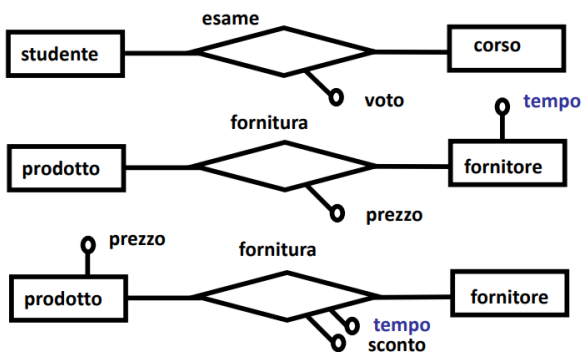
Criteri di scelta della chiave

Questa scelta è condizionata da fattori esterni:

- Esempio: per gli studenti: codice fiscale o matricola?
 - **velocità di digitazione** (*matricola*)
 - **controllo validità** (*codice fiscale*)

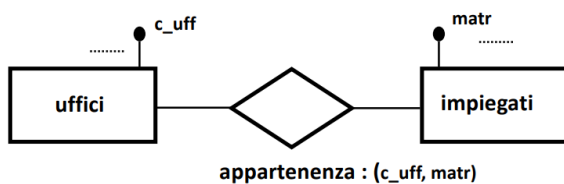
Spesso si fa uso di **valori progressivi assegnati automaticamente** dal sistema garantendo l'unicità (Ricordate `AUTO_INCREMENT`)

Proprietà delle associazioni



- Nelle associazioni **NON** si definiscono chiavi.
- L'associazione **EREDITA** le chiavi dalle entità correlate e in questo modo si definiscono le **CHIAVI ESTERNE**.
- Quindi le chiavi esterne le trovo nelle associazioni.
- Si indica **senza mettere pallini pieni** sulle associazioni

Non si possono definire chiavi nelle associazioni

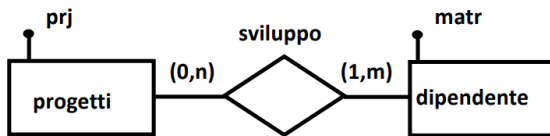


Cardinalità delle associazioni

Per cardinalità si intende il numero di volte che una data istanza di entità deve o può partecipare alla associazione:

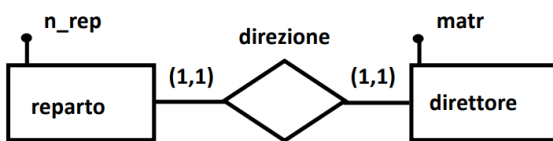
1. (1,1) : **obbligatoria**, una sola volta
2. (1,n) : **obbligatoria**, almeno una volta
3. (0,1) : **opzionale**, una sola volta
4. (0,n) : **opzionale**, n volte

Associazioni N:M



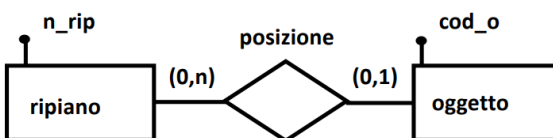
- **Con vincolo:** un dipendente **deve partecipare ad almeno** un progetto $(1, m)$
- **Senza vincolo:** ad un progetto **possono partecipare dipendenti**, ma può esistere anche un progetto senza dipendenti $(0,n)$
- Chiamata anche: **ASSOCIAZIONE MOLTI A MOLTI**

Associazioni 1:1



- **Con doppio vincolo:**
 - un reparto deve avere un direttore ed il direttore è uno solo $(1, 1)$
 - un direttore deve dirigere uno ed un solo reparto $(1, 1)$
- Il vincolo diventerebbe $(0, 1)$ se avessimo la generica entità "impiegato" al posto dell'entità "direttore" (alcuni impiegati non dirigono reparti)

Associazioni N:1



- **Con vincolo:** un oggetto può stare su un ripiano, ma su un solo ripiano $(0, 1)$
- **Senza vincolo:** su un ripiano possono stare n oggetti
- È possibile vincolare la partecipazione, per esempio ponendo $n = 5$, non più di 5 oggetti per ripiano $(0, 5)$
- Chiamata anche: **ASSOCIAZIONE MOLTI A UNO.**