



Basi di Dati

Modellazione dei dati in UML

a.a 2020/2021

Prof.ssa G. Tortora

Modellazione dei dati in UML

- UML (Unified Modeling Language) è un linguaggio grafico per la modellazione di applicazioni software basate sulla programmazione orientata agli oggetti.
 - Permette di rappresentare (attraverso una serie di diagrammi) tutti gli aspetti di un applicazione software: dati, operazioni, processi e architetture.
- UML può essere utilizzato in alternativa al modello ER:
 - I *diagrammi delle classi* sono usati per descrivere le classi di oggetti di interesse e le relazioni che intercorrono tra di esse.
- *Cambia la rappresentazione diagrammatica ma non l'approccio alla progettazione.*

UML vs. ER/EER

- UML
 - modellazione di un'applicazione software
 - aspetti strutturali e comportamentali (dati, operazioni, processi e architetture)
 - formalismo ricco
 - diagramma delle classi, degli attori, di sequenza, di comunicazione, degli stati, ...
- ER/EER
 - modellazione di una base di dati
 - aspetti strutturali di un'applicazione
 - costrutti funzionali alla modellazione di basi di dati

UML vs. ER/EER (2)

- Principali differenze di UML rispetto ad ER
 - assenza di notazione standard per definire gli identificatori
 - possibilità di aggiungere note per commentare i diagrammi
 - possibilità di indicare il verso di navigazione di una associazione (non rilevante nella progettazione di una base di dati)

UML vs. ER/EER (3)

- Formalismi diversi
- Il diagramma delle classi di un'applicazione è diverso dallo schema ER della base di dati
- Il diagramma delle classi, anche se progettato per uso diverso, può essere adattato per la descrizione del progetto concettuale di una base di dati

UML: le origini

- Proposto a metà degli anni '90 quale **formalismo unificante** per la modellazione o.o. di applicazioni software, è stato standardizzato sotto l'egida dell'Object Management Group (OMG).
- UML offre una molteplicità di diagrammi, corredati da una descrizione testuale della loro semantica: **molteplicità di viste** della medesima applicazione.

UML: il modello dell'applicazione

- L'insieme dei diagrammi definisce il **modello dell'applicazione** (controparte dello schema ER):
 - UML è un metamodello per la descrizione di modelli di applicazioni software.
- Nella sua versione corrente UML prevede un certo numero di **diagrammi fondamentali**:
 - diagramma delle classi, degli oggetti, dei casi d'uso, di sequenza, di comunicazione, delle attività, degli stati, dei componenti e di distribuzione dei componenti.

UML: I diagrammi principali (1)

- Diagramma delle classi:
 - descrive le caratteristiche statiche e dinamiche delle componenti (**classi**) e delle loro relazioni (**associazioni**).
- Diagramma degli oggetti:
 - rappresentazione delle possibili istanze delle classi (**oggetti**) e dei loro collegamenti.
- Diagramma dei casi d'uso:
 - modalità di utilizzo del sistema da parte di persone/altri sistemi (**attori**) e interazioni tra sistema e attori.

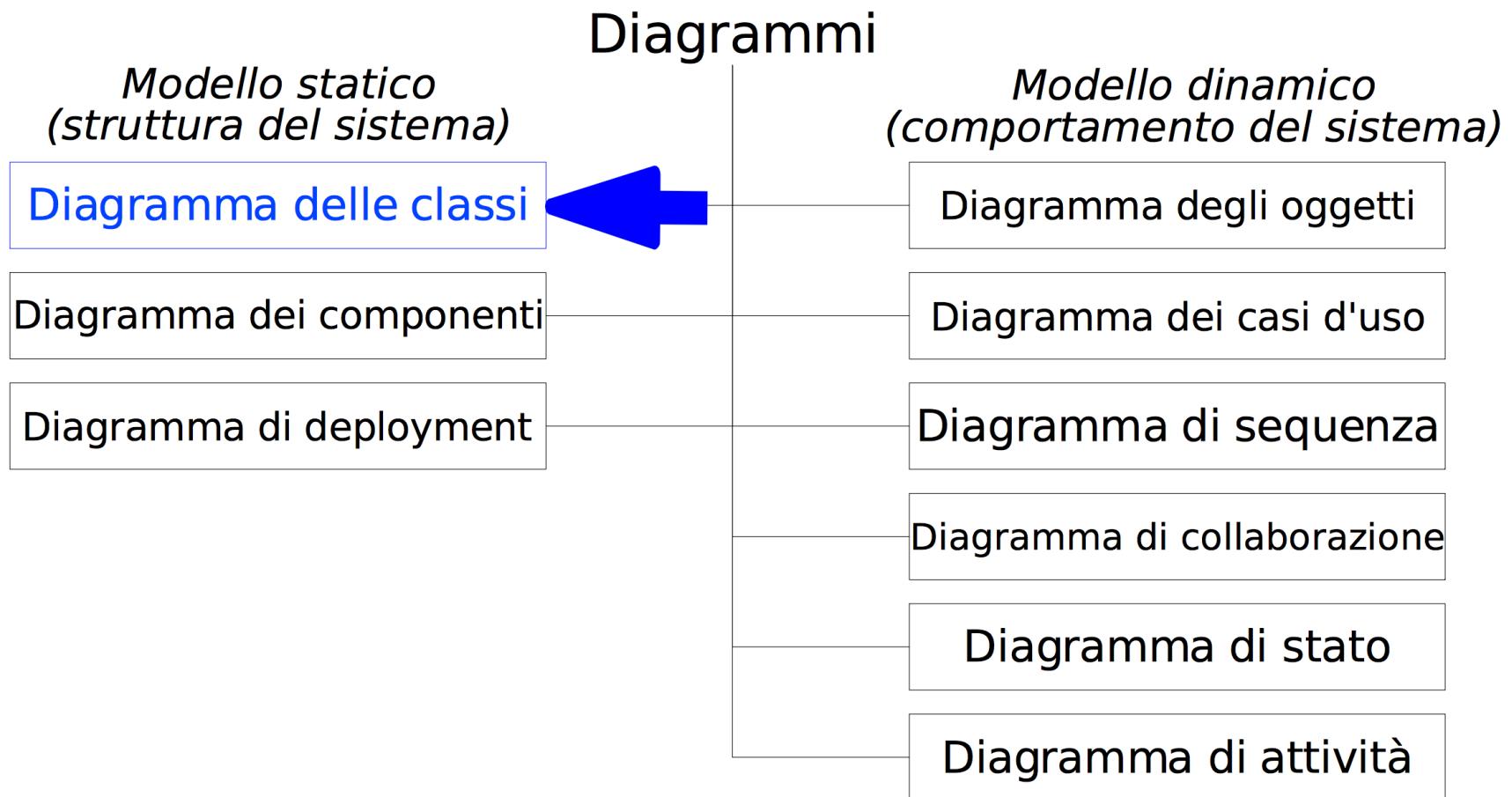
UML: I diagrammi principali (2)

- Diagramma di sequenza:
 - ordinamento temporale di messaggi (**invocazione di metodi**) scambiati tra i diversi oggetti dell'applicazione.
- Diagramma delle attività:
 - comportamento dinamico di un processo dell'applicazione in termini dei flussi di **attività** da svolgere.
- Diagramma degli stati:
 - descrive il ciclo di vita di un oggetto dell'applicazione attraverso gli **stati** che può assumere.

UML: I diagrammi principali (3)

- Diagramma di collaborazione (comunicazione):
 - descrive lo scambio di **messaggi** tra gli oggetti, ma utilizzando notazione e prospettiva diverse.
- Diagramma dei componenti:
 - descrive l'organizzazione delle **componenti** fisiche del sistema (file, moduli, ..) e le loro dipendenze.
- Diagramma di distribuzione dei componenti (deployment):
 - descrivere la **dislocazione** dei nodi hardware del sistema e le loro associazioni.

UML: I diagrammi principali (4)



Modellazione strutturale

- Modellazione strutturale:
 - Rappresenta una vista di un sistema software che pone l'accento sulla struttura degli oggetti (classi di appartenenza, relazioni, attributi, operazioni)
- Il **diagramma delle classi** descrive il tipo degli oggetti che compongono il sistema e le relazioni statiche esistenti tra loro

Rappresentazione di dati con i diagrammi delle classi

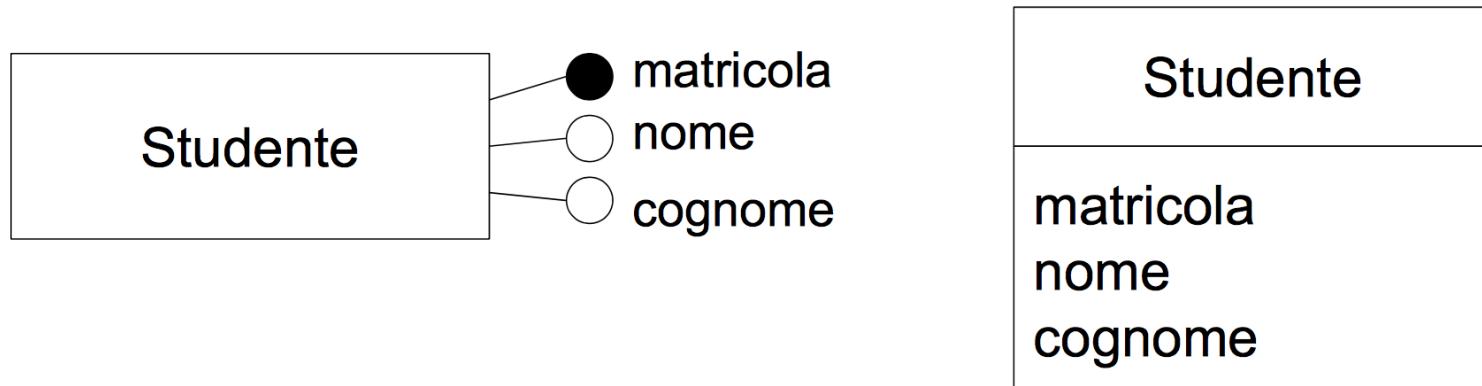
- **Classi:** Sono le componenti principali dei diagrammi delle classi.
 - Corrispondono alle entità del modello ER.
 - È possibile aggiungere i metodi (i.e., le operazioni ammissibili sugli oggetti della classe).
 - I dati vengono descritti *insieme* alle operazioni da svolgere su di essi.

Impiegato
Codice
Cognome
Stipendio
Età
<i>SetStipendio()</i>

Progetto
Nome
Budget
Data consegna

Attributi in ER e UML

- ER:
 - un attributo descrive una proprietà elementare di un'entità o di una relazione.
- UML:
 - un attributo rappresenta una proprietà elementare degli oggetti di una classe.



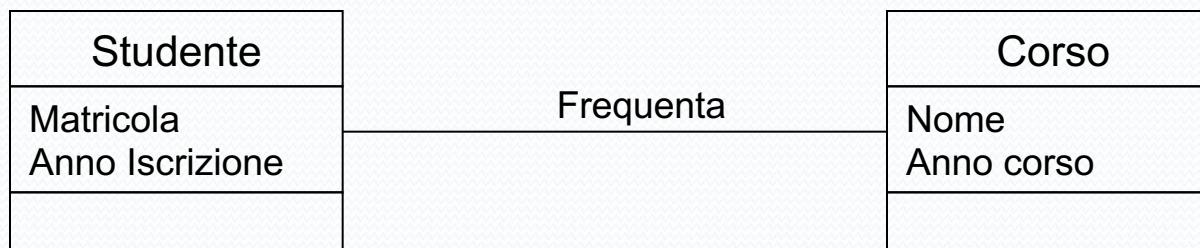
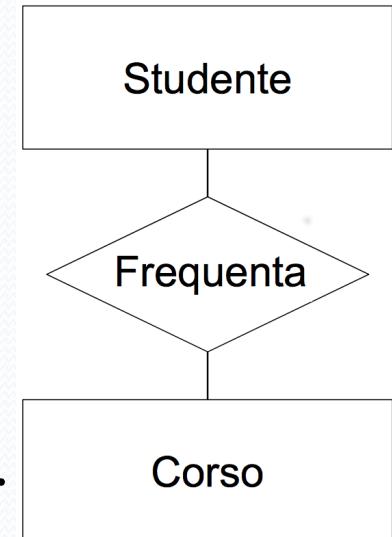
Le Classi

- Non è possibile definire attributi composti.
- È possibile associare agli attributi i rispettivi domini
 - Interi, reali, stringhe, etc.

Impiegato
String Codice
String Cognome
Float Stipendio
int Età
<i>SetStipendio()</i>

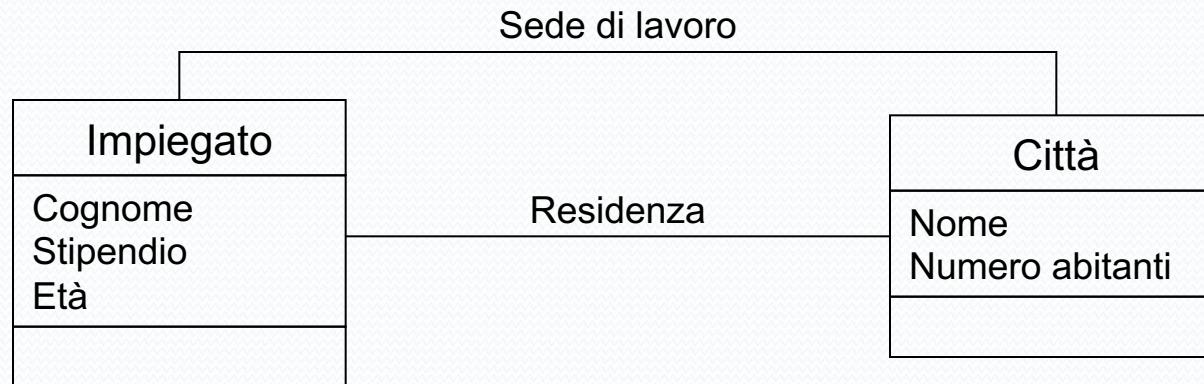
Le Associazioni

- **Associazioni:** Corrispondono alle relazioni del modello ER.
 - Le associazioni binarie si rappresentano con semplici linee che congiungono le classi coinvolte.
 - Il nome della relazione viene posto sulla linea.
 - *Non è obbligatorio:* infatti, in UML possono esistere relazioni senza nome.



Le Associazioni (2)

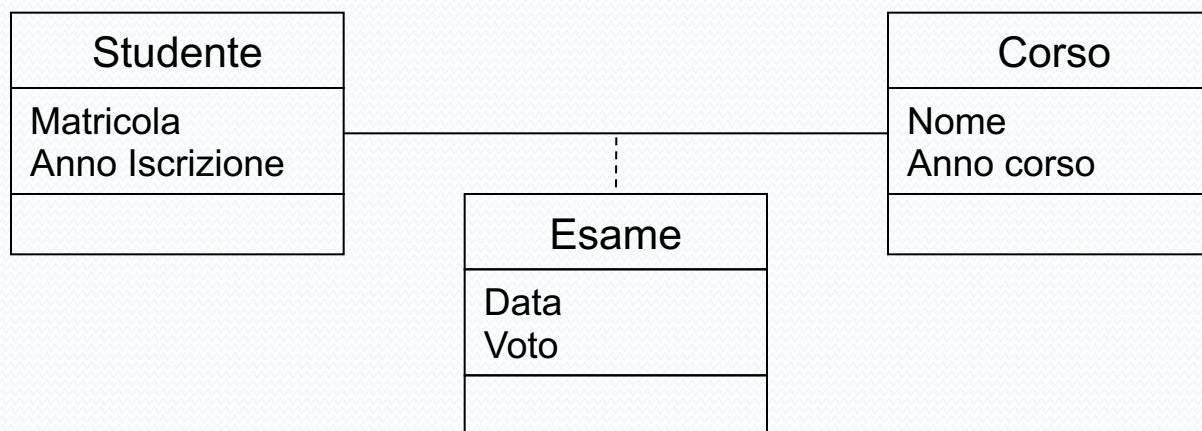
- Si possono definire più associazioni tra le medesime classi.



- È possibile associare *ruoli* alle classi coinvolte nelle associazioni.
- Non è possibile assegnare attributi alle associazioni.

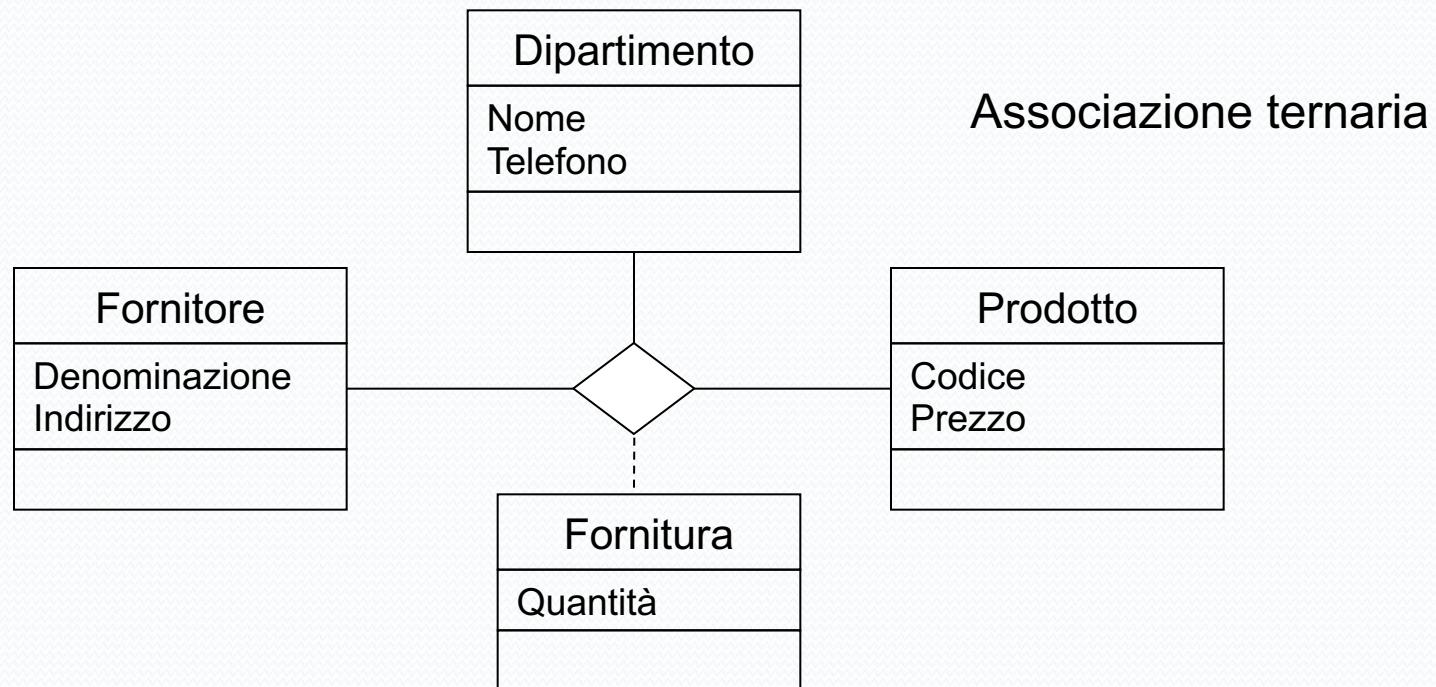
Le Associazioni (3)

- Per assegnare attributi ad una associazione si fa uso delle *classi di associazione* per descrivere le proprietà di una associazione.
 - Queste classi vengono collegate all'associazione da descrivere mediante una linea tratteggiata.



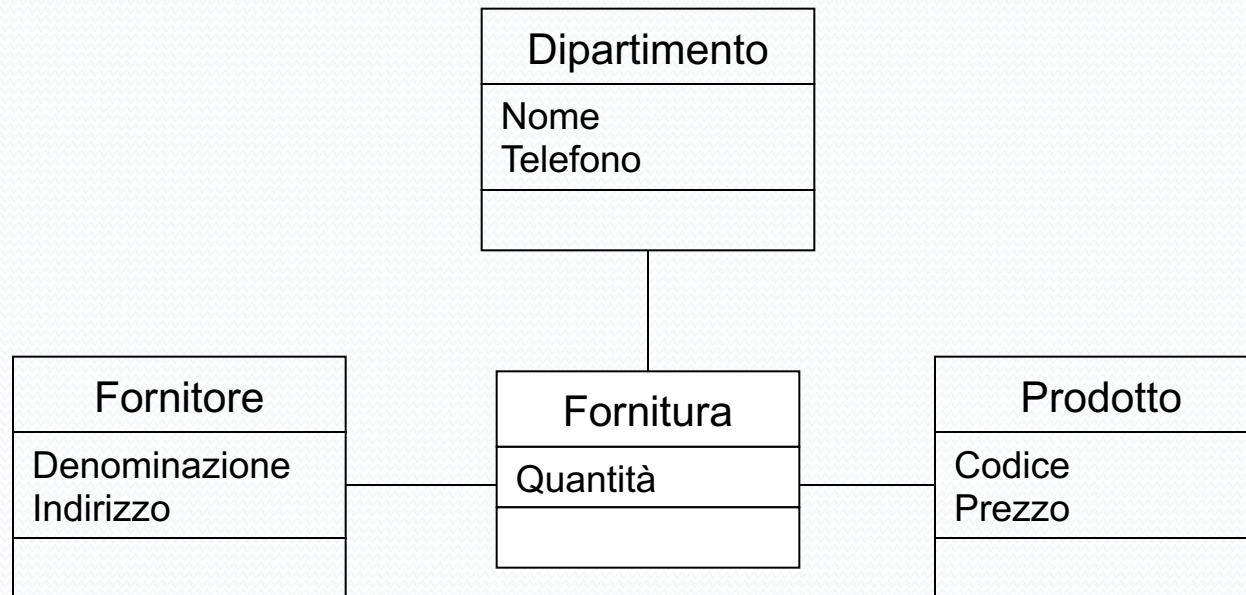
Associazioni n-arie

- L'associazione viene rappresentata da un *rombo* collegato con le classi che partecipano all'associazione.
 - Si può usare una *classe di associazione* per assegnare attributi all'associazione n-aria.



Associazioni n-arie (2)

- L'uso di associazioni n-arie è raro nel diagramma delle classi.
 - Si può applicare un processo di “*reificazione*”: ovvero trasformare l'associazione in una classe legata alle altre tramite associazioni binarie.



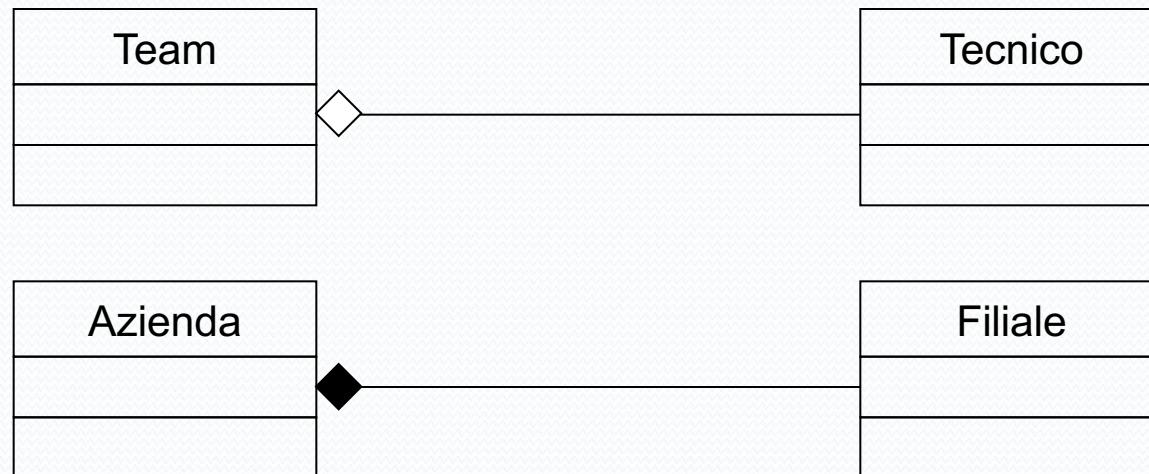
Proprietà delle associazioni

- Con una freccia è possibile indicare un verso privilegiato di *navigabilità* di un'associazione.
 - Si possono specificare *aggregazioni* di concetti: relazioni tra un oggetto composito e uno o più concetti che ne costituiscono una sua parte.
 - Sono indicate da linee avente un rombo dal lato del concetto “*aggregante*”.



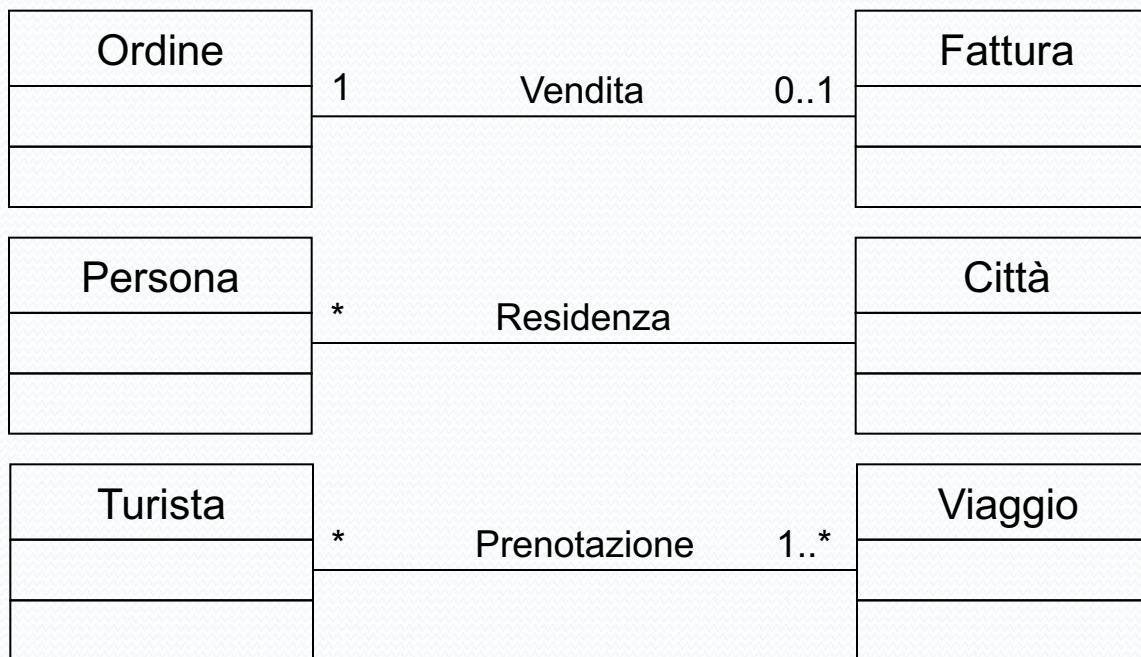
Proprietà delle associazioni (2)

- Il rombo bianco indica che un oggetto della classe “*parte*” può esistere senza dover appartenere a un oggetto della classe “*aggregante*”. (**Aggregazione**)
- Il rombo nero indica bianco indica che un oggetto della classe “*parte*” **non** può esistere senza appartenere a un oggetto della classe “*aggregante*”. (**Associazione**)



Associazioni con molteplicità

- La molteplicità di default è 1 (che denota la coppia di cardinalità 1..1).
 - Le cardinalità di default possono essere omesse.
- La cardinalità “*molti*” viene rappresentata dal simbolo * (dove si intende 0..*, ovvero (0, N) dello schema ER).
- La cardinalità di default per l’aggregazione è * (i.e., 0..*).



*Corrispondenza (invertita)
con le molteplicità dell'ER:*



Identifieri (interni)

- In UML non esiste una notazione per esprimere identifieri di classi.
- Si possono definire vincoli di integrità su associazioni e attributi specificandoli tra parentesi graffe (*vincolo utente*).
 - *Es.* $\{id\}$ indica che l'attributo è un identificatore.
 - *Es.* assegnare $\{id\}$ a più attributi indica un identificatore composto.

Automobile
Targa $\{id\}$
Modello
Colore

Persona
Data Nascita $\{id\}$
Cognome $\{id\}$
Nome $\{id\}$
Indirizzo

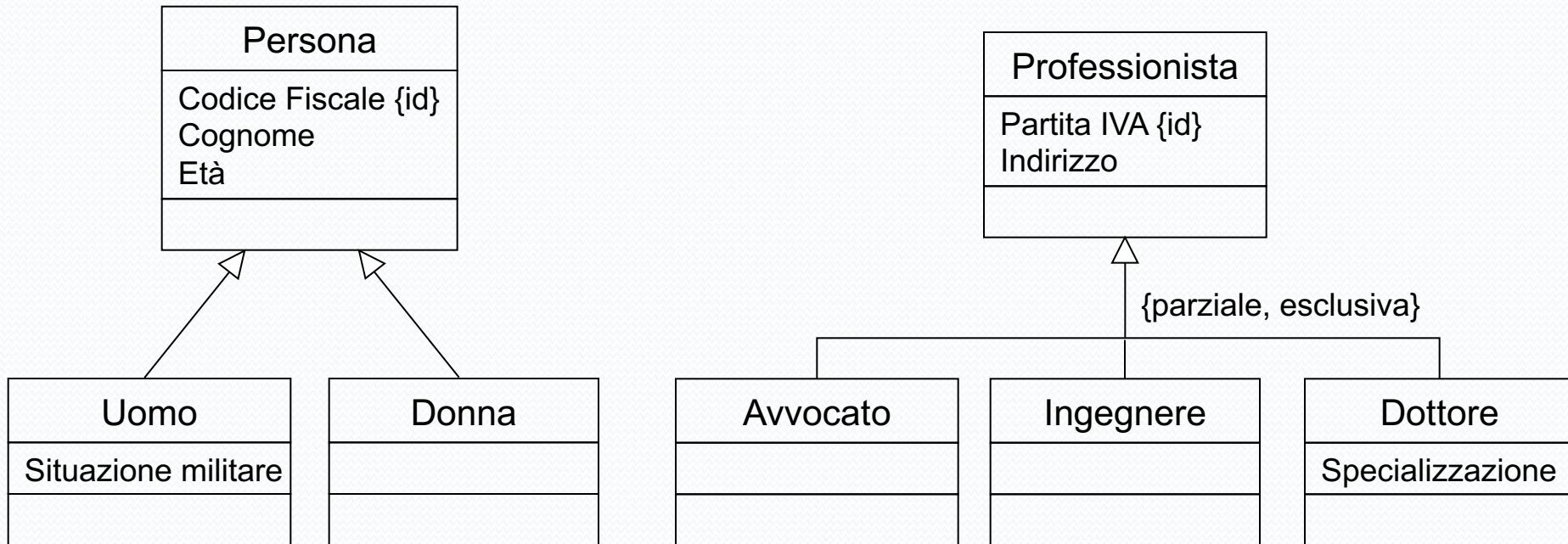
Identificatore esterno

- Per gli identificatori esterni si usa uno *stereotipo*.
 - Si usano per estendere i costrutti base dell'UML:
 - quando si vuole modellare un concetto che non può essere modellato con i costrutti di base.
 - Vengono indicati da un nome racchiuso tra i simboli << e >>.
 - *Es.* Lo stereotipo <<identificante>> insieme alla *Matricola* identifica univocamente uno *Studente* rispetto una particolare *Università*.



Generalizzazioni

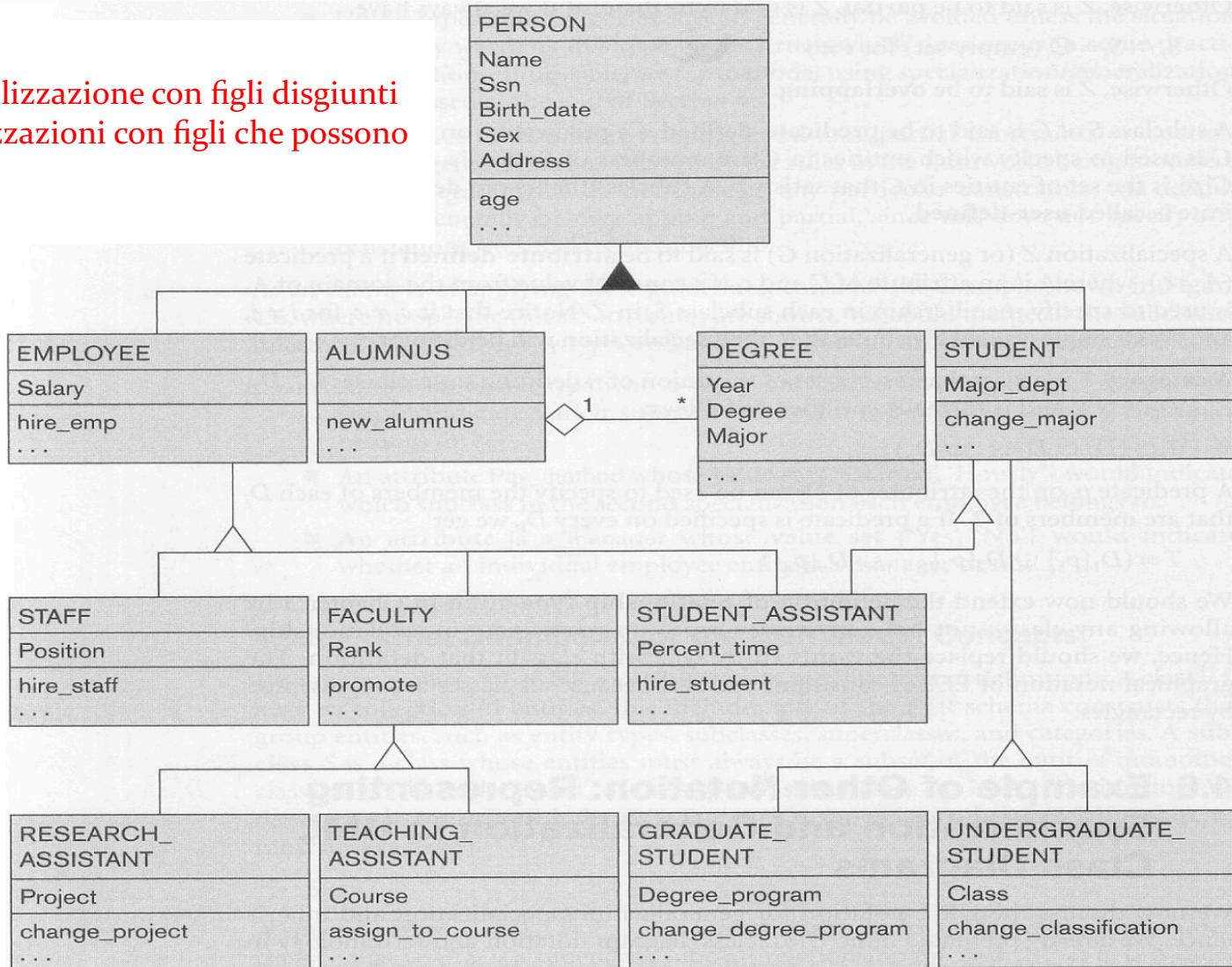
- Le generalizzazioni sono molto simili a quelle del modello ER.
- Eventuali proprietà possono essere rappresentate con vincoli racchiusi tra le parentesi { e }.



Una gerarchia di generalizzazioni

Triangolo bianco = generalizzazione con figli disgiunti

Triangolo nero = generalizzazioni con figli che possono sovrapporsi



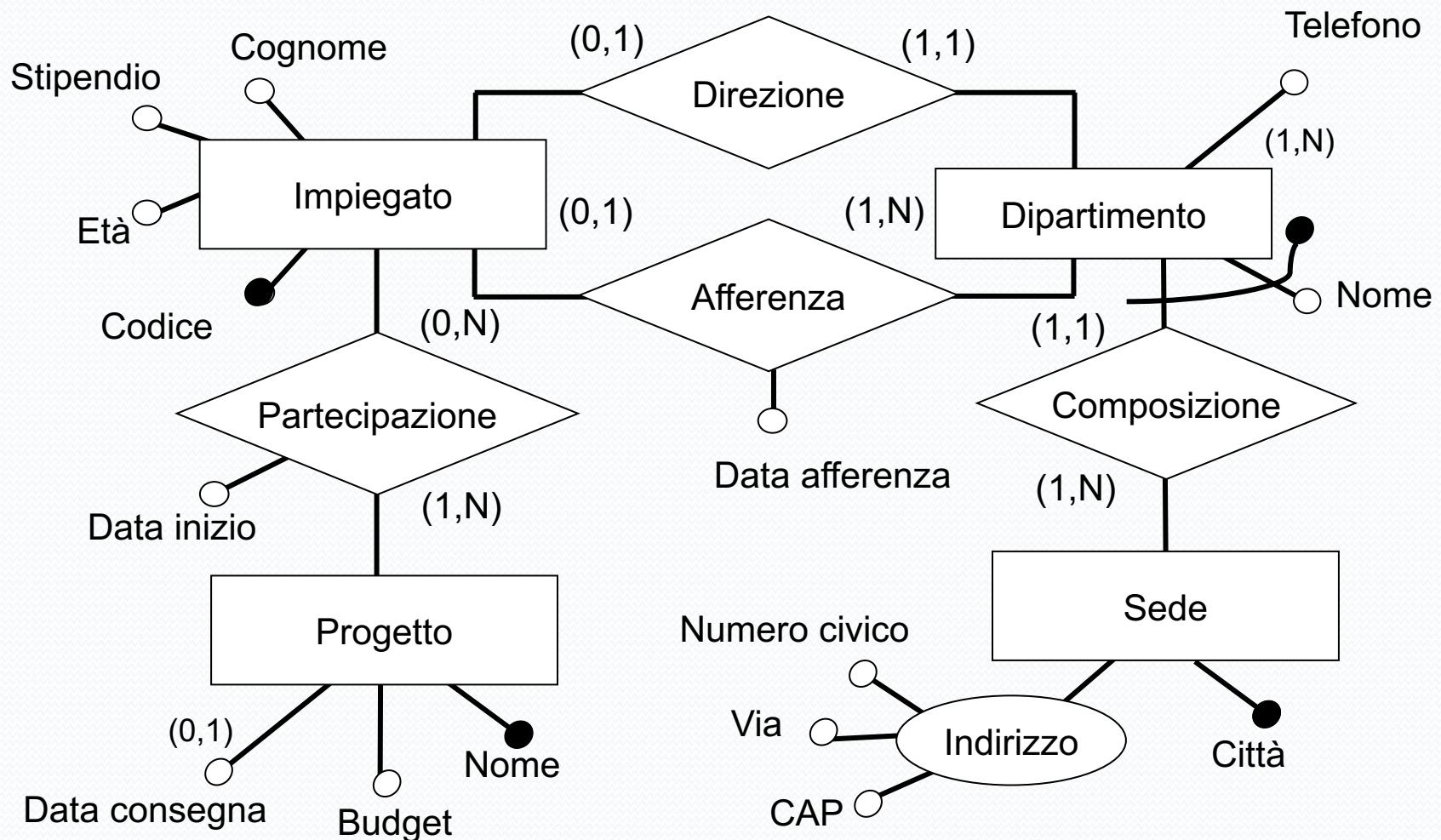
Le note

- Un diagramma delle classi può essere documentato con delle *note*.
 - Consistono in semplici commenti testuali.
 - Sono riportate sul diagramma in un rettangolo con l'angolo superiore destro ripiegato.
 - Una nota può essere associata ad un particolare elemento del diagramma attraverso una linea tratteggiata.

Esempio

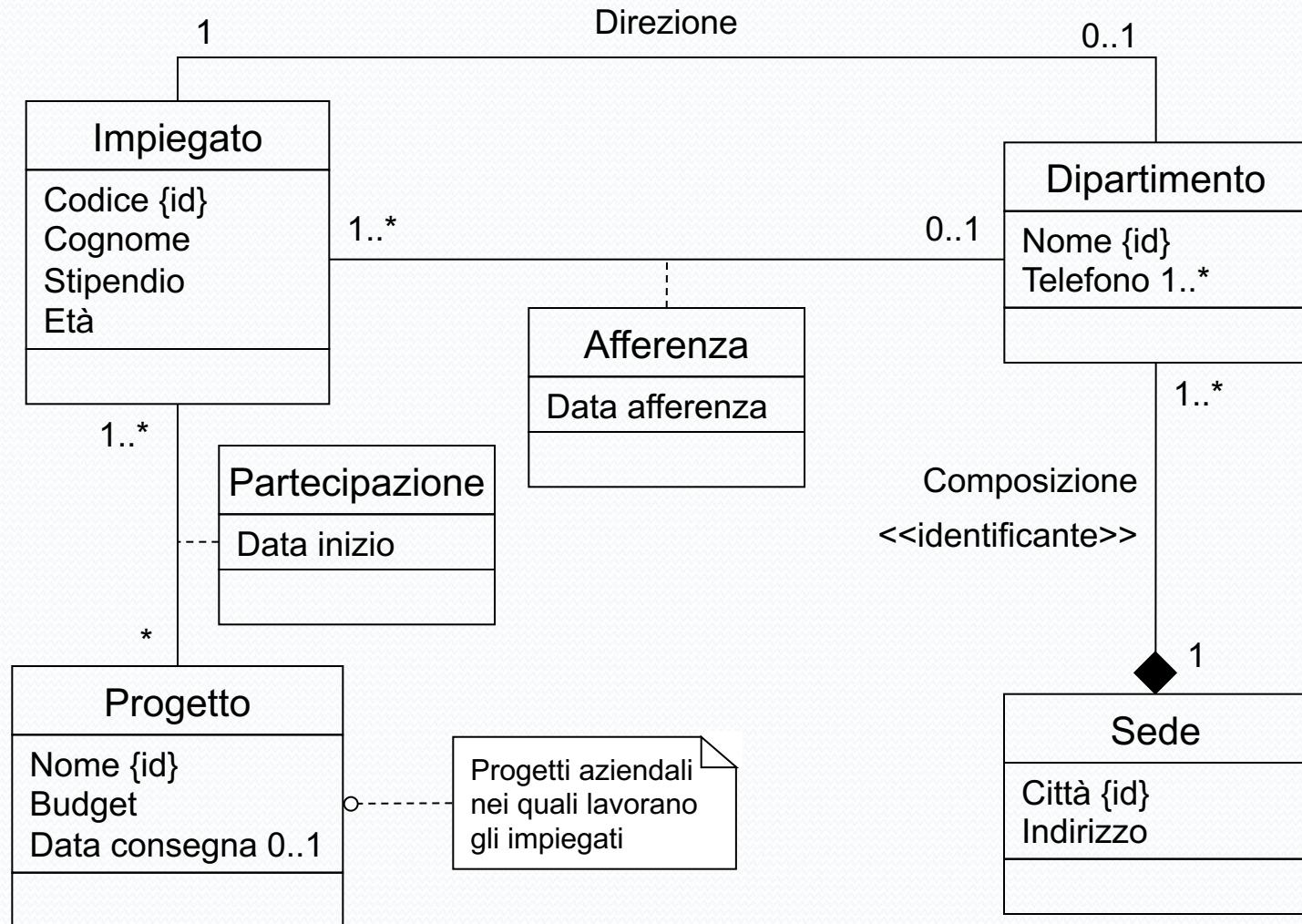
- *Modellare uno schema riguardanti le informazioni di carattere organizzativo relative ad una azienda con diverse sedi.*
 - Una **sede** dell'azienda è identificata dalla città ed è composta da una serie di dipartimenti che non possono essere definiti al di fuori della sede. Una sede ha anche un indirizzo.
 - Un **dipartimento** è identificato dal nome e dalla sede di appartenenza e possiede diversi numeri di telefono.
 - Ai dipartimenti afferiscono (a partire da una certa data) uno o più impiegati; e un impiegato li dirige.
 - Gli **impiegati** vengono rappresentati dal cognome, stipendio, età e un codice che serve per identificarli.
 - Gli impiegati lavorano su zero o più progetti a partire da una certa data.
 - Ogni **progetto** ha un nome, un budget e una data di consegna che può non essere specificata.
 - Una **nota** associata alla classe progetto ne descrive il significato.

Lo schema concettuale (notazione Atzeni)



Indirizzo è stato modellato come attributo composto: Via, Numero civico, CAP

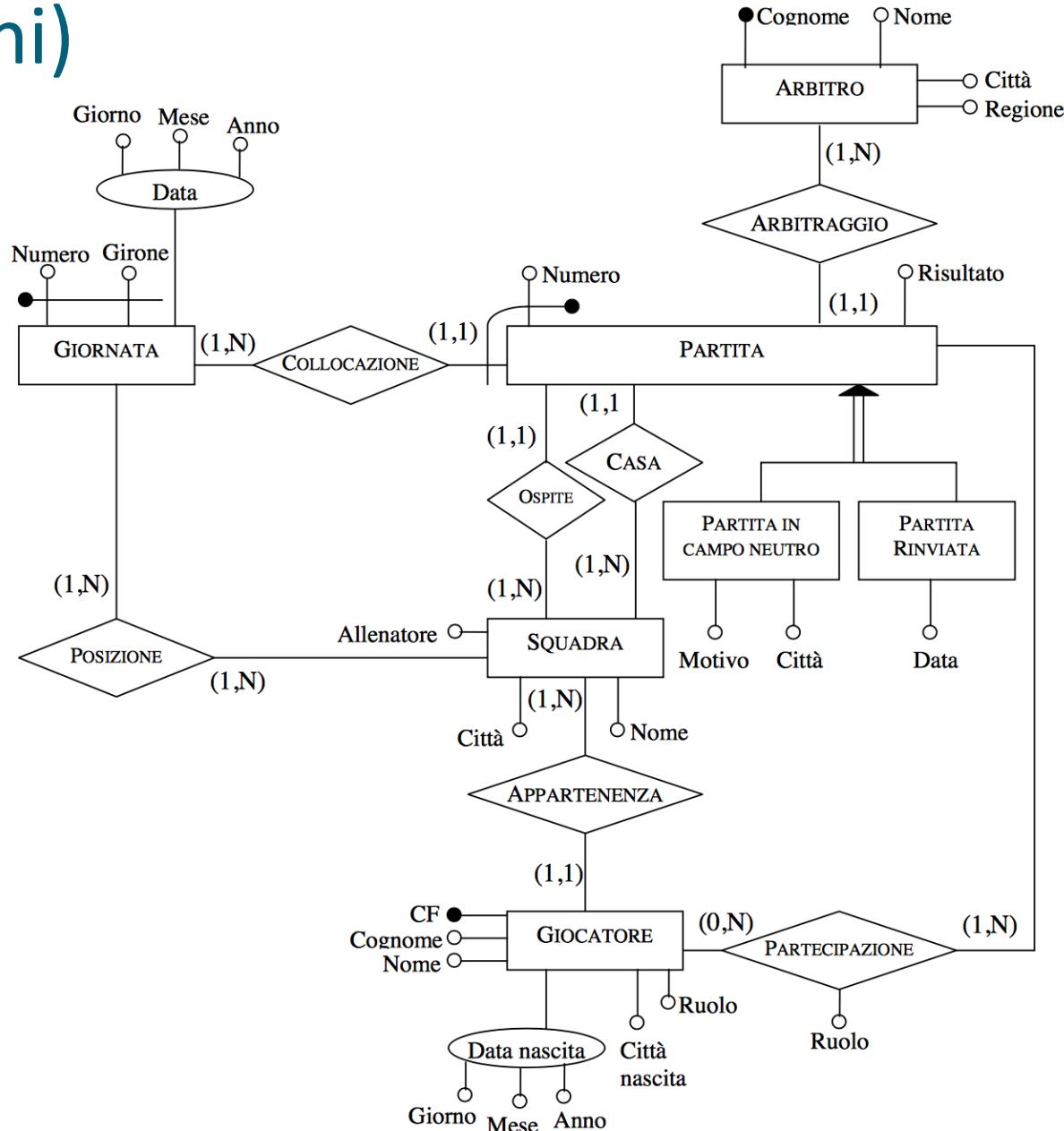
Lo schema concettuale in UML



Esempio 2

- *Modellare uno schema riguardanti le informazioni di un campionato di calcio:*
 - L'entità **SQUADRA** rappresenta tutte le squadre del campionato, indicando per ognuna di esse il nome, la città e il nome dell'allenatore.
 - L'entità **GIOCATORE** rappresenta i giocatori delle squadre: ogni giocatore ha un contratto con una sola squadra e ogni squadra ha più giocatori. I giocatori sono identificati dal loro Codice Fiscale e per ognuno di essi è indicato il nome, il cognome, il ruolo nella squadra, la città di nascita e la data di nascita.
 - Lo schema contiene anche informazioni sulle partite. Una **PARTITA** è identificata con un numero (che deve essere differente per tutte le partite dello stesso giorno) e con un riferimento al giorno (attraverso la relazione **COLLOCAZIONE** e l'entità giornata).
 - Le relazioni **CASA** e **OSPITE** rappresentano le due squadre che giocano la partita: per ogni partita è indicato il risultato e l'**ARBITRO**, con la relazione **ARBITRAGGIO** tra partita e arbitro; questa entità rappresenta tutti gli arbitri del campionato e per ognuno di essi è indicato il Nome, il Cognome, la Città e la Regione. Un arbitro è rappresentato solo se ha arbitrato almeno una partita.
 - Una partita può essere giocata su campo neutrale o può essere rinviata ad un'altra data (ma questi due eventi non sono ammessi contemporaneamente nello schema).
 - La relazione **Partecipazione** rappresenta il fatto che un giocatore abbia giocato in una partita, la sua posizione (che può essere diversa dalla sua solita). Lo schema non esprime la condizione che i giocatori che giocano una partita devono avere un contratto con una delle due squadre.
 - L'entità **GIORNATA** rappresenta la giornata del campionato. Sono identificate con Numero e Girone. La relazione **Posizione** dà il punteggio di ogni squadra in ogni giornata.

Lo schema concettuale (notazione Atzeni)



Lo schema concettuale in UML

