

**RACE CONDITION** → si verifica quando più processi provano a modificare

una stessa locazione di memoria nello stesso momento  
ma ciò può portare a non ottenere un risultato corretto

la gestione degli accessi è data allo SCHEDULER

inoltre il parallelismo NON esiste in quanto esso è in realtà  
implementato tramite l'affezarsi / esclusione delle operazioni

### ↓ MUTUA ESCLUSIONE

La mutua esclusione viene implementata attraverso metodologie  
del tipo lock. In Java esistono i metodi synchronized che  
risultano essere molto buoni.

**Appunto Tabella hash** → Tale tabella va bene nel solo caso in cui il metodo di  
accesso cambi (avendo sia sempre e solo la chiave) altrimenti

se essa varia oppure si voglia fare una ricerca per range essa  
è altamente inefficiente

Dunque spesso vengono utilizzati più sistemi o dei sistemi  
con più alberi cresciuti ordinato secondo un criterio

**BASI DI DATI** → Sono definite di dati e cose di informazioni in quanto

esse sono dette

TRANSAZIONALI

e gestiscono in

automatico la

concorrenza

GAIA BERTOLINO

- L'informazione è uno/più dati con un significato

- il dato può rimanere lo stesso ma l'informazione no in  
quanto il dato è una descrizione codificata dell'informazione

- l'informazione è assunto come CONCETTO PRIMITIVO dunque non  
viene esattamente definito ma si parla di MISURA DELL'INFORMAZIONE  
ovvero di quanto viene offerto lo stato del sistema che lo riceve.

I sistemi di gestione sono detti DBMS (Database Management System)  
es. Oracle, DB2 (IBM), MySQL

Access ad esempio non è una base di dati ma un semplice file nel file  
system che non gestisce la concorrenza sui singoli dati ma solo su  
tutto il file.

# Progettazione

domenica 23 gennaio 2022

13:49

La fase preliminare è la scelta di quali **INFORMAZIONI** rappresentare nel database e ciò è preliminare alla rappresentazione dei dati.

Quando un cliente e progettista si incontrano per la prima volta viene effettuato uno **STUDIO DI FATTIBILITÀ** in quanto si raccogliono informazioni (preliminari) sulle richieste del committente e si verifica la fattibilità allo stato attuale.

Successivamente si passa all'**ANALISI DEI REQUISITI** e alla **RACCOLTA DI SPECIFICHE (FUNZIONALI E NON)** ovvero di operazioni e azioni che il sistema deve essere in grado di eseguire e specifiche software richieste per il sistema. Questo passaggio avviene tramite una serie di **INTERVISTE** col cliente.

Il passo successivo è la **PROGETTAZIONE CONCETTUALE** che è la parte che necessita l'esperto informatico. Qui si realizza un "elenco" delle informazioni necessarie per la richiesta ricevuta e i legami fra esse. Tale rappresentazione avviene tramite un **MODELLO ENTITÀ-RELAZIONE (ER)**.

Successivamente si passa alla **PROGETTAZIONE LOGICA /FISICA**; logica indica la scelta della struttura logica (es. lista concatenata, Tabella hash) che serve a rappresentare i dati ed è logica e non fisica in quanto ad una struttura logica possono corrispondere diverse rappresentazioni fisiche (es. una lista concatenata può essere rappresentata come una linked list, array list ecc.). La sintassi per la realizzazione del progetto è quella del **MODELLO RELAZIONALE** che utilizza tabelle.

In particolare, in questo esame è data come struttura la Tabella e dunque è necessario scegliere in che modo rappresentare i dati al suo interno.

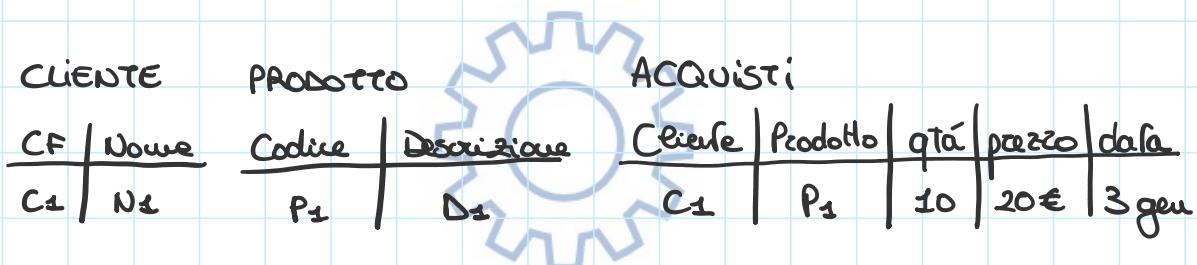
es. CLIENTI : CF, nome RELAZIONI  
 ACQUISTI : Clienti, prodotti, quantità, prezzi, data  
 PRODOTTI : codice, descrizione

Potrei creare un'unica tabella del tipo

CFcliente	Nomecliente	CodicePr	Descriz.	q'tà	prezzo	data
C1	N1	P1	D1	10	20€	3 gen
:						

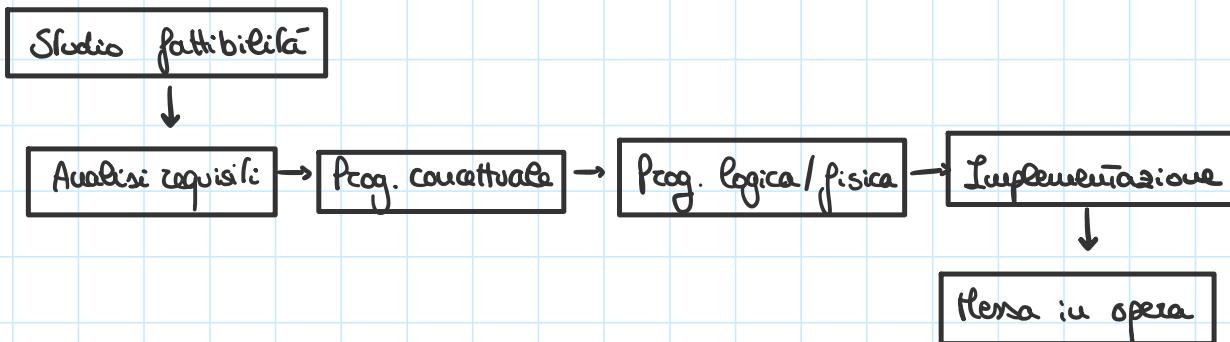
→ progettazione logica

Tuttavia questo non è l'unico modo ma potremmo creare tre tabelle diverse:



In particolare, la ridondanza dei dati NON influenza in alcun modo la nostra scelta perché occupa più memoria. Tuttavia, potrebbe causare un problema nel caso in cui si voglia modificare un campo.

Dunque i vari passaggi possono essere rappresentati tramite una waterfall (schema a cascata):



## MODELLO ENTITÀ - RELAZIONE

È dato l'esempio di un database di gestione per un Pomicino:

## MODELLO ENTITÀ - RELAZIONE

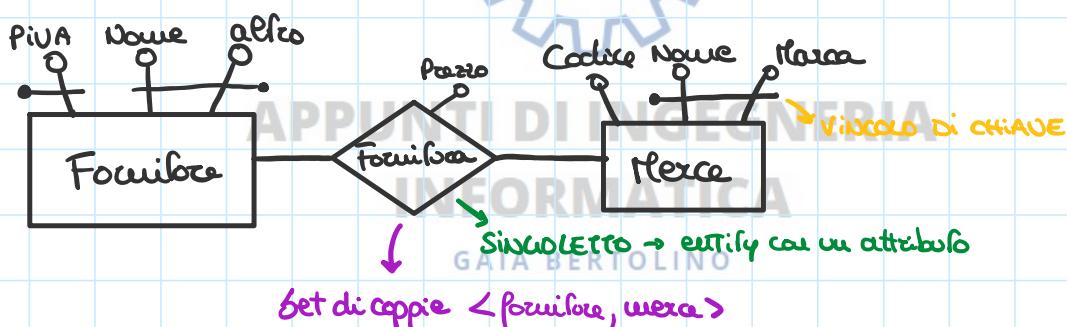
È dato l'esempio di un database di gestione per un fornitore:

- 1) Inserimento, modifica, Cancellazione, Ricerca di fornitori
  - 2) Caratteristiche dei fornitori (PiVA, nome, ...)
  - 3) Inserimento, modifica, Cancellazione, Ricerca di merce
  - 4) Caratteristiche della merce (codice, nome, ...)
  - 5) Inserimento, modifica, Cancellazione, Ricerca di fornitore
- ↳ Ricerche in base alla merce e al fornitore

→ questi sono i **REQUISITI**

Rappresento poi i concetti all'interno di rettangoli e posso rappresentare i suoi attributi attraverso piccole ramificazioni. Il rettangolo si chiama formalmente **ENTITÀ** o **ENTITY SET** → **INSIEME** (el. distinti)

Per quanto riguarda la fornitore, esse più che una entity a sé stante risulta essere una entity di "collegamento" che ha sia propri attributi sia derivanti dalle due entity merce e fornitore



In particolare, il concetto di relazione fa riferimento a quello matematico di sottoinsieme del prodotto cartesiano dei due insiemi messi in relazione.

Il concetto stesso di entity set significa che due righe devono differire per almeno un campo.

Dire che un campo è **IDENTIFICANTE** o **CHIAVE** se serve per univocamente determinare vari elementi di una tabella.

Ne consegue che due elementi non possono avere stessa chiave.

Tra i... — tra Tral... ... attributi indica che non sono identificanti

Il simbolo — che Taglia un attributo indica che essi rappresentano le chiavi. Inoltre, una chiave può essere caratterizzata da uno o più attributi.

→ È sbagliato dire che una chiave è un insieme di attributi identificatori per una entity in quanto è necessario che due elementi differiscono per un campo mentre, con un veicolo del genere, si sta definendo che essi debbano differire per tutti gli attributi della chiave.

Dunque una chiave deve essere composta da attributi e **minimale** rispetto all'identificazione ovvero non deve esistere un sottoinsieme della chiave in grado di identificare elementi dell'insieme.

→ NON deve esistere un sottoinsieme identificante della chiave ma la chiave deve essere il mininale.

L'insieme è detto mininale ma non minimo in quanto possono esservi più **chiavi candidate** con cardinalità diverse che sono tutte valide e comunque minimali ma non necessariamente minime. Inoltre, se sono presenti più chiavi esse si "escludono" a vicenda dal punto di vista della identificazione.

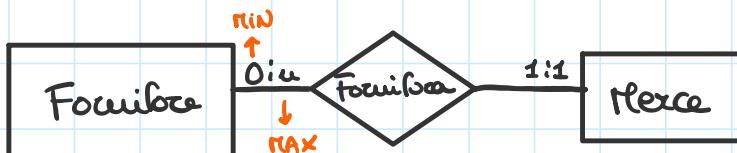
Per ogni entity deve essere indicata ALMENO una chiave e se non è possibile indicarne una (nemmeno considerando tutti gli attributi) allora vuol dire che manca un qualche attributo alla entity. Invece, per una relazione NON si deve specificare alcuna chiave; in caso contrario è sbagliato.

Nel modello E-R non tutte le specifiche possono essere indicate graficamente; in tal caso si ricorre ad una spiegazione tramite linguaggio naturale.

Quando si crea una relazione è possibile specificare la cardinalità

Quando si crea una relazione è possibile specificare la cardinalità degli elementi della entity all'entity collegata

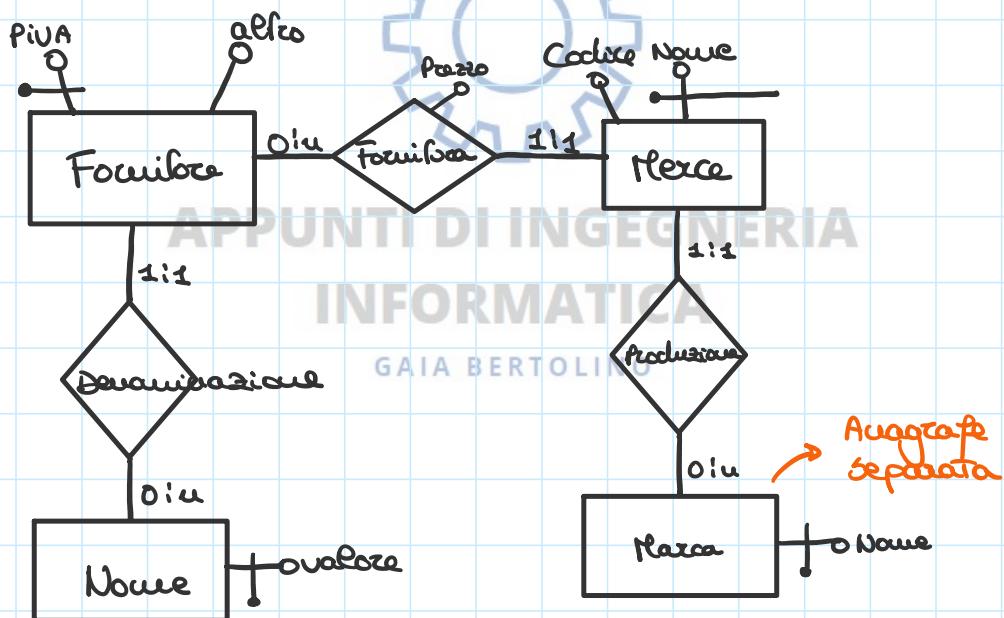
es.



Ricordando che le forniture sono coppie si può dire che:

- un fornitore è presente in numero di coppie del tipo fornitura che va da 0 a n
- una merce è presente in una sola coppia fornitoria

Tramite questi **VINCOLI DI CARDINALITÀ** è possibile trasformare un attributo come relazione. Dunque l'esempio precedente diventa



In questo caso non è influente "l'aumentare" ipotetico di riferimenti utilizzate.

L'utilizzo di un attributo è libero / generico mentre una entity permette di avere un "altro" definito e preciso di quale valore quel campo può avere e qual è la relazione fra due entity.

Ne consegue che se ad esempio inserisco la marca come attributo altro non avrò vincoli e dunque sarà a libero inserimento; invece, se la

non avrò vincoli e dunque sarà a libero inserimento; invece, se la marca è una entità allora avviene un "controllo incrociato" fra il valore del campo e l'entità adibita. Ne consegue una maggiore consistenza nei dati e l'impossibilità di scrivere un dato errato.

→ Dunque la scelta fra attributo ed entità è data dal caso specifico e dalla volontà di avere più o meno consistenza e correttezza.

I vincoli di cardinalità possono restituire le chiavi delle entità.

Rispetto al caso precedente si possono ottenere i seguenti casi :

1)



Un fornitore fornisce un numero qualsiasi di merci (da 0 a n)

2)



Un fornitore fornisce almeno una merce.

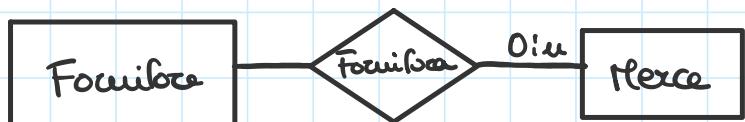
3)



Una merce è fornita da esattamente un solo fornitore.

Dunque una merce compare una sola volta.

4)

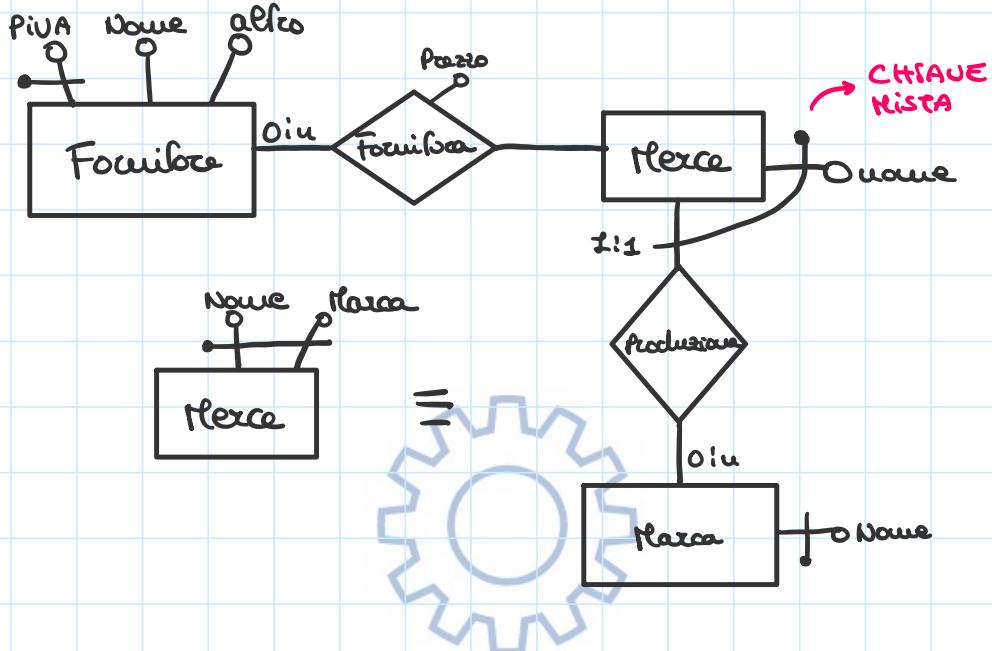


Una merce è fornita da zero o più fornitori.

Visto che una fornitura è una coppia merce - fornitore allora può comparire zero o più volte ma ciascuna volta con un nuovo fornitore poiché, anche se la fornitura ha altri attributi, essi non concorrono alla chiave che è data da

un'azion, essi non consentono di avere chiavi che sono un  
fornitore e merce.

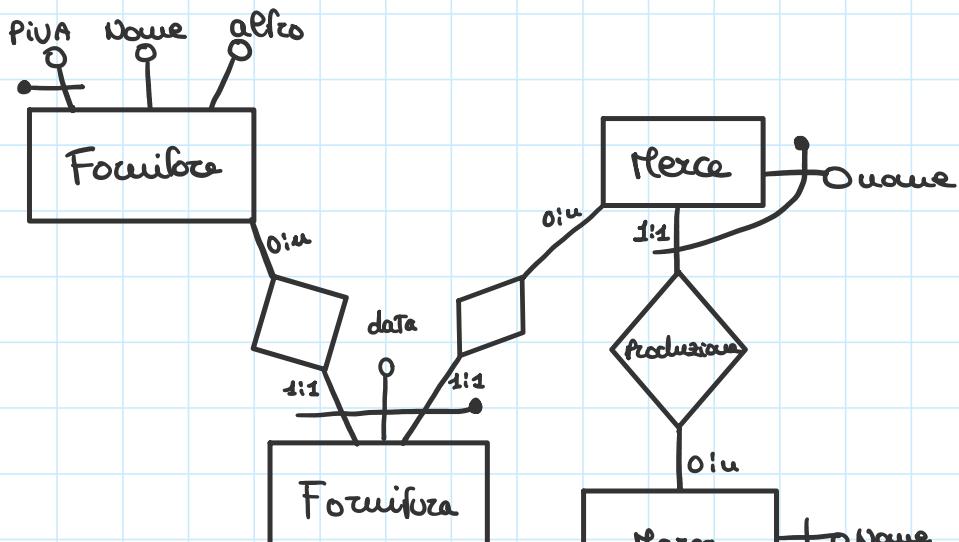
Per definire che una chiave è data da attributi e relazioni  
(detta **CHIAVE MISTA**) si utilizza un taglio che attraversa attributo  
e relazione.

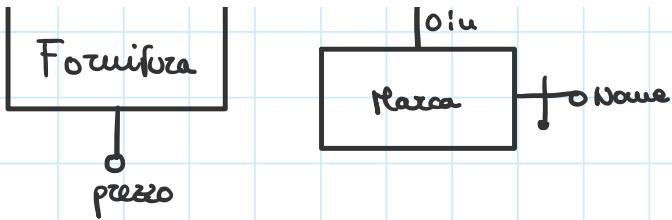


Ne deriva che il vincolo di cardinalità della relazione  
appartenente alla chiave deve essere del tipo **1:1**.

GAIA BERTOLINO

Per quanto riguarda l'esercizio, per introdurre il concetto per cui  
posso avere più forniture ma cronologicamente distinte lo schema  
a cascata diventa del tipo:



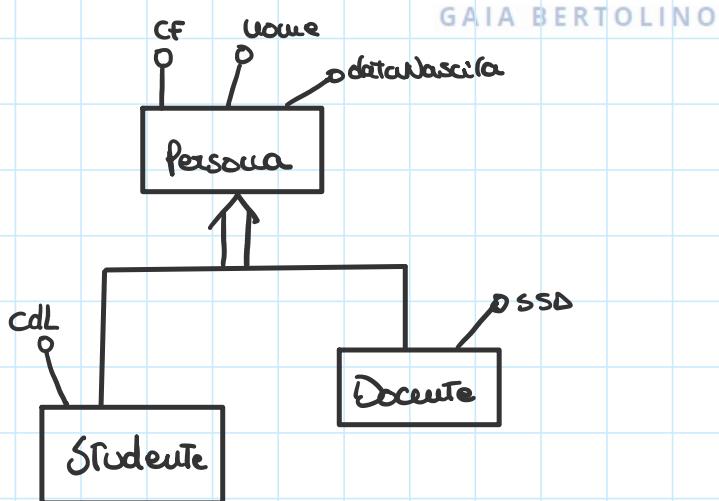


Una relazione 0:1 / 1:0 è SEMPRE traducibile in una entity del tipo 1:1 / 1:1. Infatti l'1:1 rappresenta la caratteristica della relazione trasformata di essere un insieme.

Attenzione: Le relazioni del modello ER è Bivariata. Dunque non esistono relazioni ternarie e nel caso sembra necessario si può trasformare in una entity set.

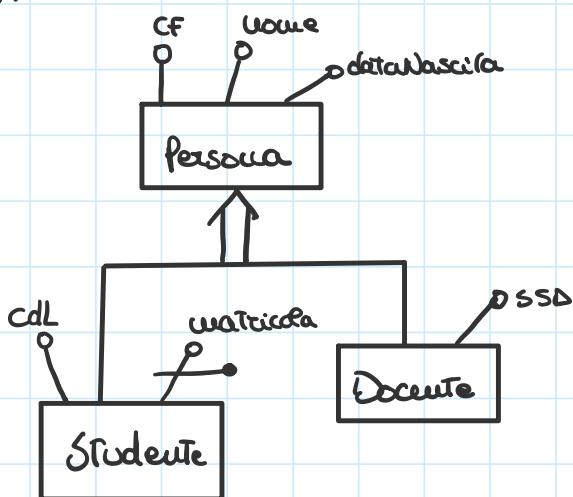
### Divisione in sottoinsiemi

Si vuole rappresentare gli insiemi studenti e docenti con relativi dati e per ogni professore si specifica l'insegnamento. Professori e studenti sono riconducibili ad un'unica entity persona. Nel modello E-R tale legame è rappresentato da una freccia



Una sottoentity non deve per forza avere un attributo o chiavi; infatti esse ereditano attributi e chiavi dall'entity principale. Tuttavia, sarebbe possibile introdurre una chiave come nell'esempio riportato qui un ad inserire una nuova attivita' controllata su quella già

Tuttavia, sarebbe possibile introdurre una chiave come nell'esempio seguente che va ad inserirsi come chiave candidata a quelle già presenti:



Attenzione: non capisrai mai di dove avere una chiave nella sottoentità

L'entità Person sarà simile al concetto di classe astratta della programmazione orientata agli oggetti.

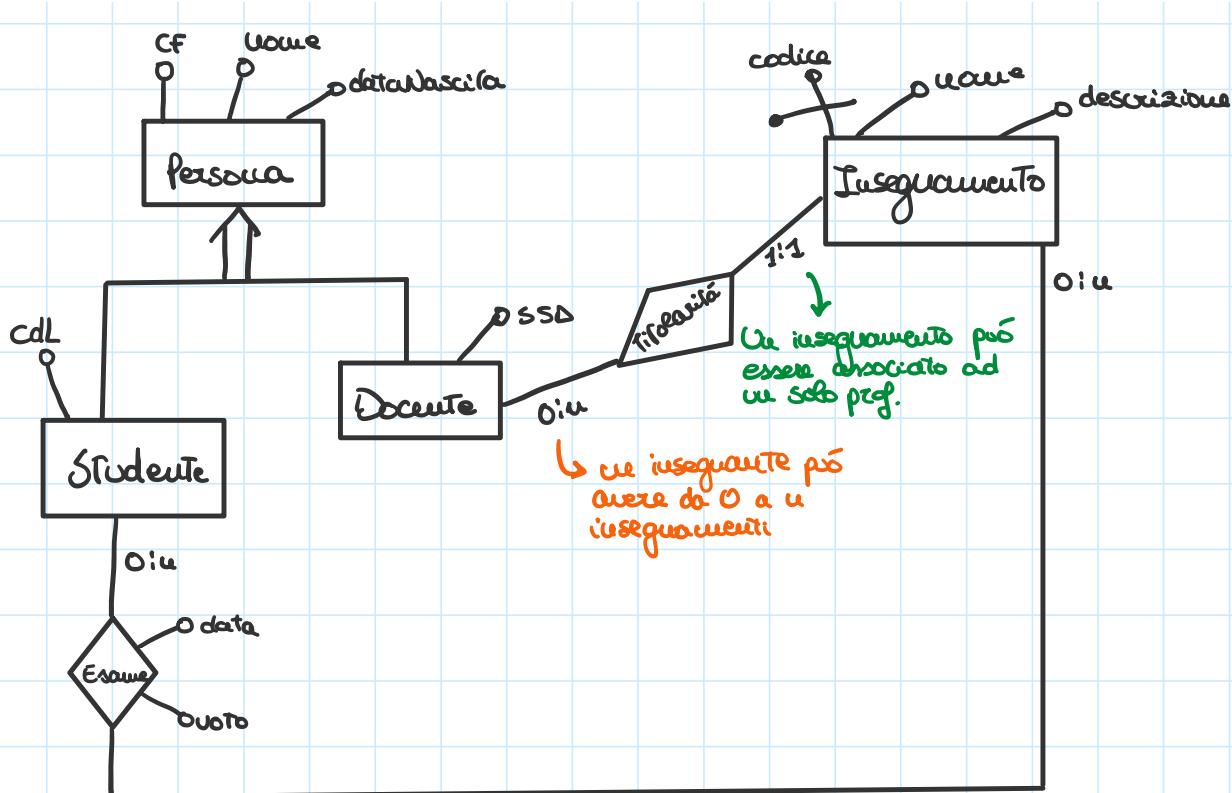
La relazione esistente fra sottoentità e Person è detta di

**GENERALIZZAZIONE** che può a sua volta essere

- Totale → ogni istanza dell'entità appartiene sicuramente ad una delle sottoentità
- partiale → delle istanze della entità possono far parte di sottoentità diverse da quelle rappresentate
- esclusiva → le sottoentità sono insiemi disgiunti
- inclusiva → le sottoentità sono insiemi che si sovrappongono

Se non si indica nulla allora la relazione è totale ed esclusiva e in questo caso si ha una partizione.

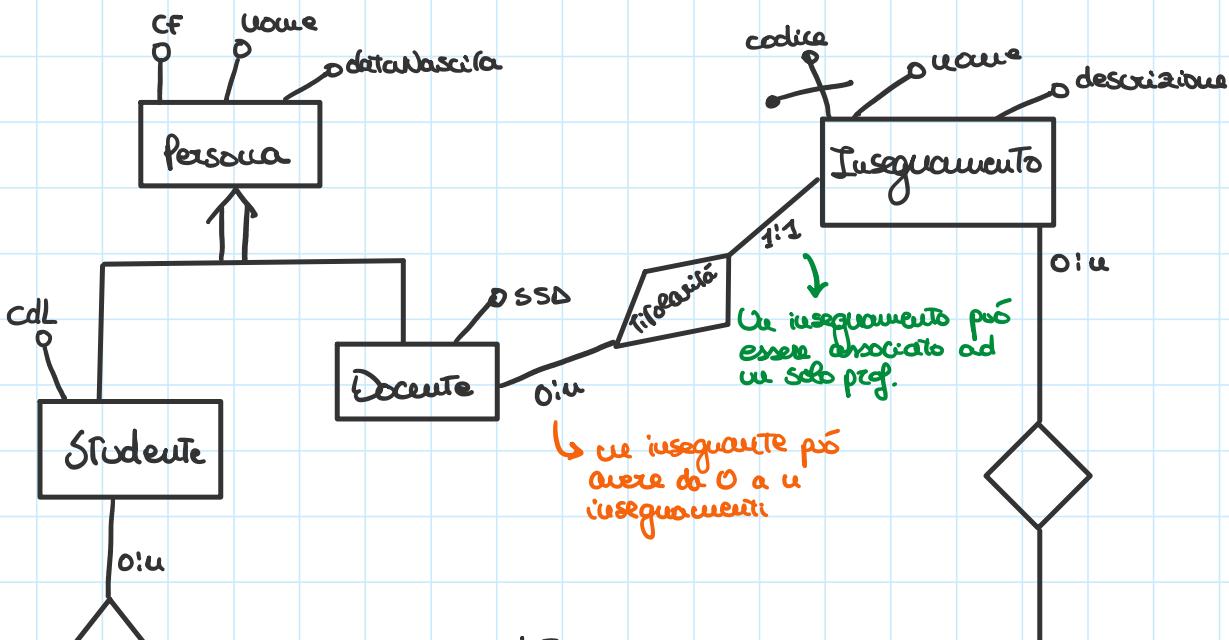
Arricchiamo il modello con gli insegnamenti e gli esami

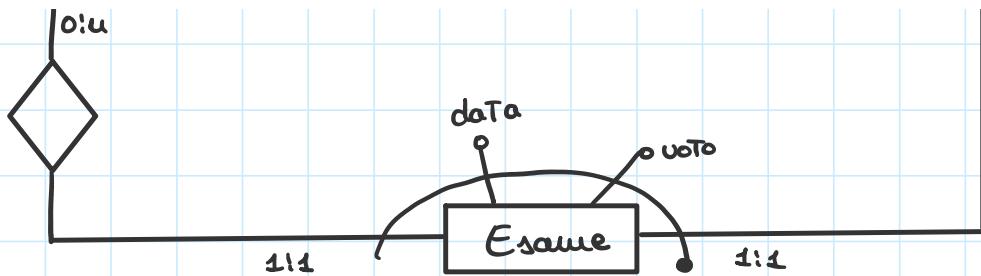


Tale tipo di relazione rappresenta il libertà degli esami.

Infatti, l'insieme esame è dato dalle coppie studente-insegnamento dove ad ogni studente può essere associato un qualsiasi insegnamento e ad ogni insegnamento è associato un qualsiasi studente ma ogni coppia è presente una sola volta.

Per invece registrare tutti i tentativi d'esame si può ricorrere all'uso di una entity :





Tuttavia in questo caso si verifica che non è possibile chiarire che non possono esistere più esami con voto maggiore o uguale a 18. Dunque tali specifiche rientrano fra quelle che vanno espresse a parte accanto lo schema progettuale.

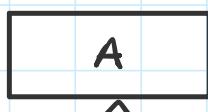
Per inserire una lista di caratteristiche NON si devono usare attributi, ma relazioni ed entity. Infatti gli attributi sono tipi semplici.

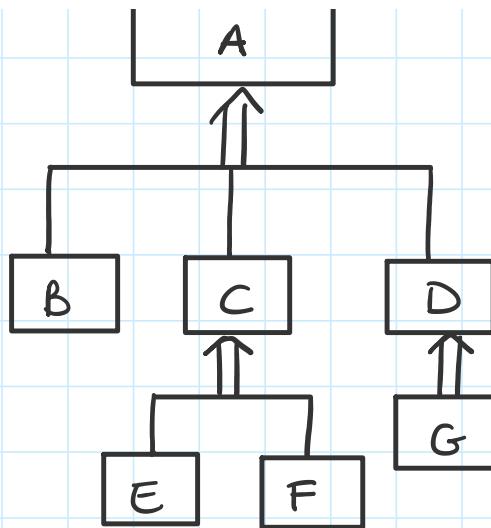
Utilizzare un attributo causa incertezza nei dati e rallenta la ricerca poiché anziché avere una entity separata con una anagrafe specificata esso potrà assumere un valore qualsiasi e sarà associato ad una entity esistente che andrà visibile tutta nel caso della ricerca.

GAIA BERTOLINO

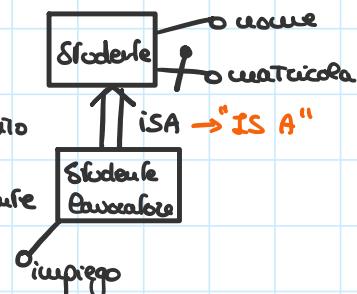
La scelta del posizionamento di un attributo fra entity e sottoentity è importante perché mettendolo e ripetendolo nelle sottoentity significa che essi hanno uno significato diverso; invece, se hanno significato uguali vanno inseriti nella entity seguendo il principio di massima sintesi.

La generalizzazione non è binaria e si perde totalità ed esclusività se provo a trovarla attraverso due relazioni. Inoltre, una generalizzazione può avere molte sottoentity:





Posso anche avere entity  
che non sono sottentity  
come ad esempio studente  
e studente lavoratore :



Tale tipo di relazione è  
SEMPRE parziale in quanto  
uno studente lavoratore  
è sicuramente uno studente  
ma non è valido il  
viceversa.

Inoltre, l'ISA non va mai tradotto con  
una relazione in quanto è completamente  
generico mentre la parziale indica  
specificazione.

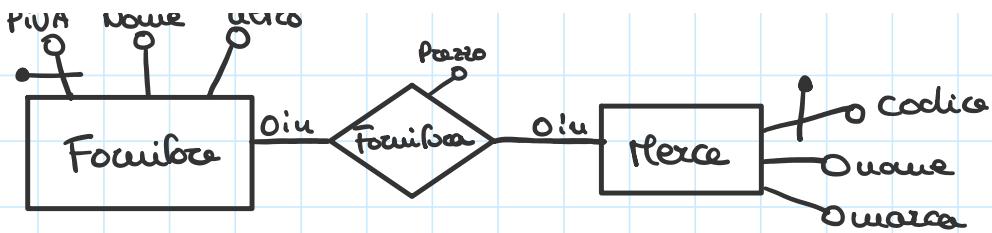
## PROGETTAZIONE LOGICA

I database relazionali sono detti RDBMS e sono detti relazionali  
in quanto usano le tabelle come strutture logiche. Una riga di una  
tavella è detta **TUPLA**.

Si può usare un approccio da concettuale a logico e viceversa ma noi useremo  
il primo.

Il primo passo consiste nel visualizzare il progetto concettuale !





Lo schema logico si realizza indicando la struttura della tabella ovvero nome tabella e nome colonne; essa non deve essere rappresentata graficamente ma attraverso la dicitura:

Nome Tabella (col1, col2, col3, ... coln)

Terminologicamente con **schema di tabella** si intende la tabella vuota di dati e con la sola riga di intestazione. Invece, per quanto riguarda l'**istanza di tabella** indica una tabella con all'interno dei dati.

Nel caso del progetto concettuale precedente si ha che, creando una sola tabella si potrebbe avere ridondanza o inconsistenza dei dati oltre che a volte rispettare le specifiche richieste tra cui le chiavi e l'insieme di relazioni.

GAIA BERTOLINO

**Attenzione!** una cella può avere valore null (ovviamente se esiste una relazione con vincolo di cardinalità 0:1)

Dunque anche nel caso di un progetto logico bisogna specificare dei vincoli di chiave; in questo caso potrebbe essere Piva e codice (se si usa una tabella unica).

Per schematizzare: nello schema Bisogna indicare nome, colonne e chiavi. In generale, la chiave è separata dal modello ER

Tuttavia nel caso corrente una singola tabella NON è corretta una

Tuttavia nel caso corrente una singola tabella non è corretta ma si creano tabelle separate per ogni entity diverso:

FORNITORE		
PIVA	NOME	CITTÀ
F <sub>1</sub>	N <sub>1</sub>	C <sub>1</sub>
F <sub>2</sub>	N <sub>2</sub>	C <sub>2</sub>

MERCI		
CODICE	NOME	MARCA
M <sub>1</sub>	N <sub>M1</sub>	γ
M <sub>2</sub>	N <sub>M2</sub>	β

### FORNITURA

FORNITORE	MERCE	PREZZO
→ F <sub>2</sub>	→ M <sub>1</sub>	
→ F <sub>1</sub>	→ M <sub>2</sub>	
→ F <sub>2</sub>	→ M <sub>2</sub>	

le chiavi sono circondate da un rettangolo

Sfessa merce, due fornitori diversi

Sfesso fornitore, due merci diverse

INSTANZA

## APPUNTI DI INGEGNERIA INFORMATICA

Fornitore (PIVA: VARCHAR, NOME: VARCHAR(20), CITTÀ: CHAR(15))

→ Tuttavia in questo esame esistono di indicare i tipi. Dunque:

Fornitore (PIVA, NOME, CITTÀ)

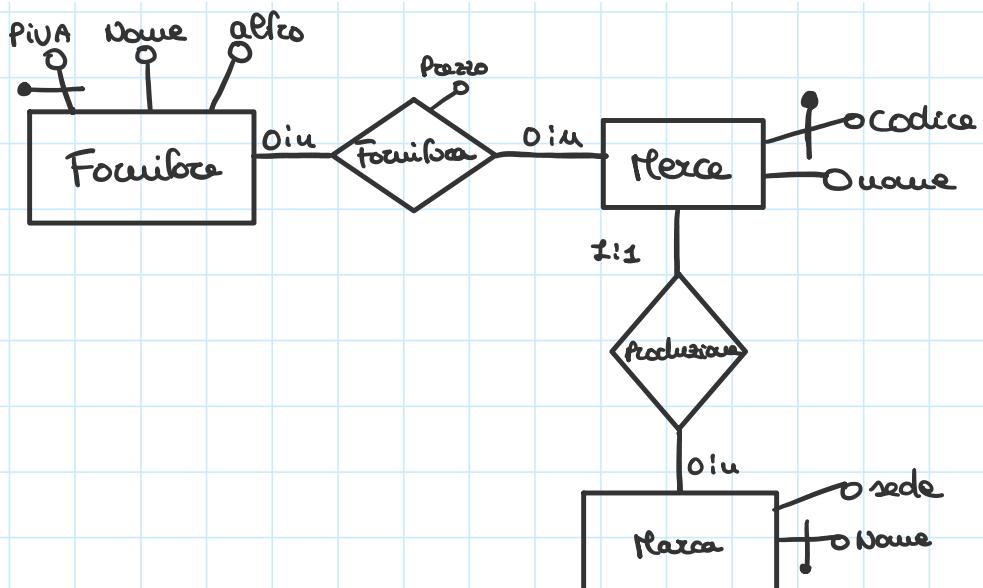
↳ gli attributi sottolineati identificano le chiavi

Merce (CODICE, NOME, MARCA)

Fornitura (FORNITORE, MERCE, PREZZO)

→ SCHEMA

Vi sono casi in cui tuttavia non è corretto tradurre tutte le entity in Tabelle. Ad esempio:



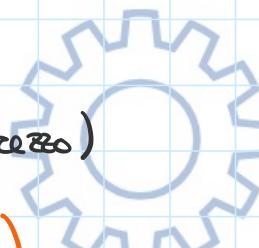
Fornitore (Piva, nome, città)

Marca (nome, sede)

Merce (codice, nome)

Fornitura (fornitore, merce, prezzo)

Produzione (merce, marca)



se avessimo scelto entrambe

tuttavia tale soluzione è stata creata rispettando il vincolo di 1:1  
perché una produzione può avere  
in quanto permette di avere  
merci senza mercie collegate

Fornitore (Piva, nome, città)

→ L'uso della parola NULL indica che l'attributo può essere nullo

Marca (nome, sede)

Merce (codice, nome, marca)

*correzione*

Fornitura (fornitore, merce, prezzo)

Produzione (merce, marca)

Se il vincolo fosse stato di 0:1 / 0:u sarebbe stato corretto  
l'esempio precedente anche se una soluzione simile sarebbe:

Fornitore (PiVA, nome, città)

Marca (nome, sede)

Merce (codice, nome, marca)

Fornitura (fornitore, merce, prezzo)

Tuttavia questa soluzione risulta essere peggiore nel caso in cui molti campi di marca siano null. Nel campo nome ci sono indicazioni in merito quindi la scelta con relativa spiegazione è affidata allo studente.

Inoltre, nei DBMS reali bisogna indicare anche la cosiddetta **chiave esterna** per definire che i campi di una tabella fanno riferimento ad un'altra tabella e dunque non possono essere diversi da essi.



es. Fornitura [fornitore]  $\xrightarrow{FK}$  Fornitore [PiVA]

Nel modello relazionale posso indicare le varie chiavi candidate nel seguente modo:

Fornitore (PiVA, nome, città)  
 $\xrightarrow{\text{UNIQUE}}$  **altra chiave candidata**  
 $\hookrightarrow$  **chiave primaria**

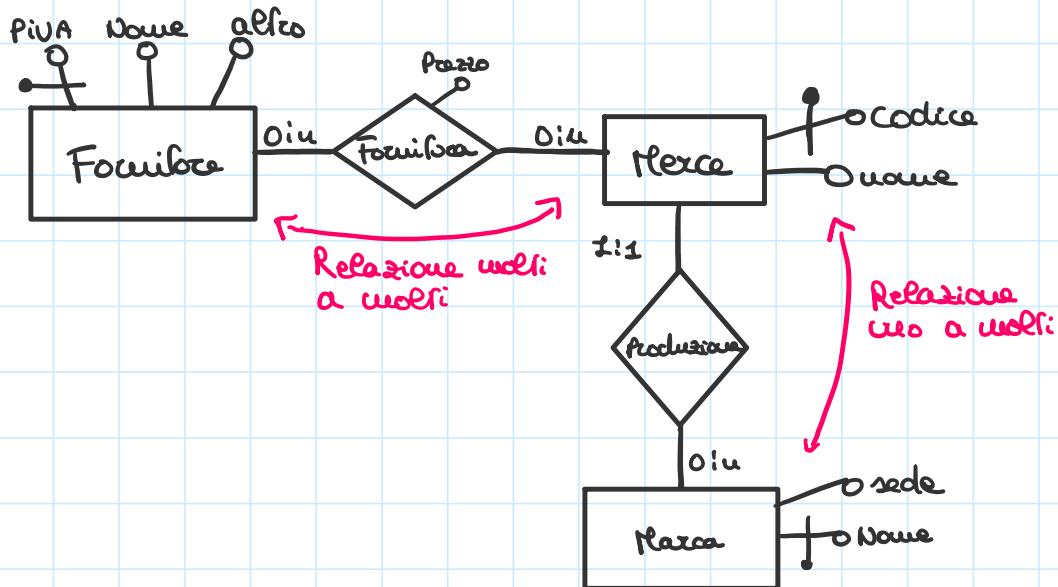
Negli esercizi studieremo solo la chiave primaria.

Una chiave esterna può riferirsi solo a chiavi primarie e mai a chiavi unique.

Regole generali di trasformazione da ER a R:

Trasformo ogni entity e ogni relazione molti a molti in delle tabelle mentre ogni relazione uno a molti lo introduco in una tabella e inserisco la chiave esterna

es. completo:



Fornitore (Piva, nome, città)

Marca (nome, sede)

Merce (codice, nome, marca)

Merce [marca]  $\leftarrow_{Fk}$  Marca [nome]  $\rightarrow$  CHIAVE ESTERNA

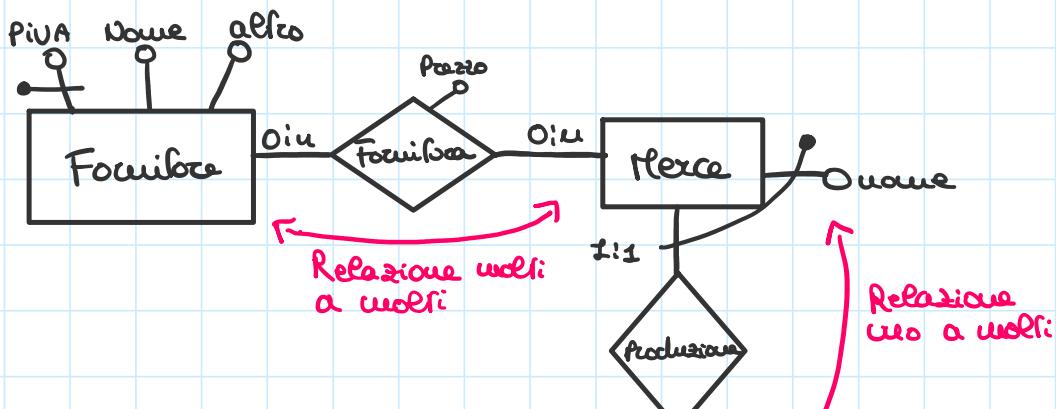
Il vincolo deve essere espresso  
nei confronti di un vincolo  
principale

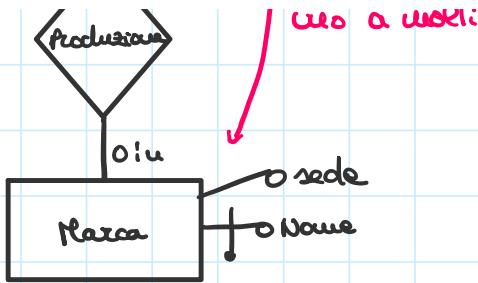
Fornitura (fornitore, merce, prezzo)

Fornitura [fornitore]  $\leftarrow_{Fk}$  Fornitore [Piva]

Fornitura [merce]  $\leftarrow_{Fk}$  Fornitura [codice]

Una chiave mista presente nel seguente schema si traduce  
nel seguente modo:





Fornitore (PiVA, Nome, Città)  
UNIQUE

Merce (Nome, sede)

Merce (Nome, marca)

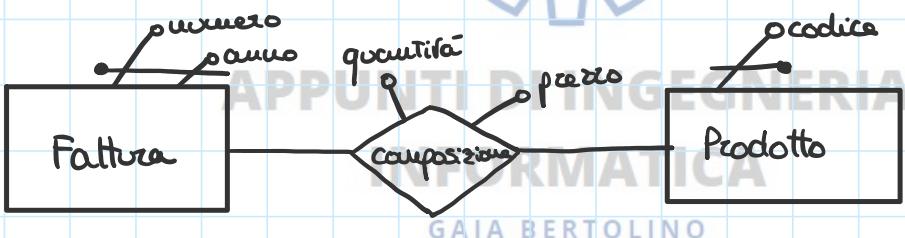
Merce [merce]  $\leftarrow_{FK}$  Merca [Nome]  $\rightarrow$  CHIAVE ESTERNA

Fornitura (fornitore, merce, prezzo)

Fornitura [fornitore]  $\leftarrow_{FK}$  Fornitore [PiVA]

Fornitura [merce]  $\leftarrow_{FK}$  ?

Nel caso di una **relazione di composizione** (avendo il caso seguente)  
si ha:



Fattura (numero, anno)

Prodotto (codice)

Composizione (fattura, prodotto, quantità, prezzo)

~~Composizione [fattura]  $\leftarrow_{FK}$  fattura [numero, anno]~~

Composizione (fattura-anno, fattura-numero, prodotto, quantità, prezzo)

~~Composizione [fattura-numero]  $\sqsubseteq_{FK}$  Fattura [numero]~~

~~Composizione [fattura-anno]  $\sqsubseteq_{FK}$  Fattura [anno]~~

- 1) sono semanticamente corrette  
2) non sono chiavi primarie

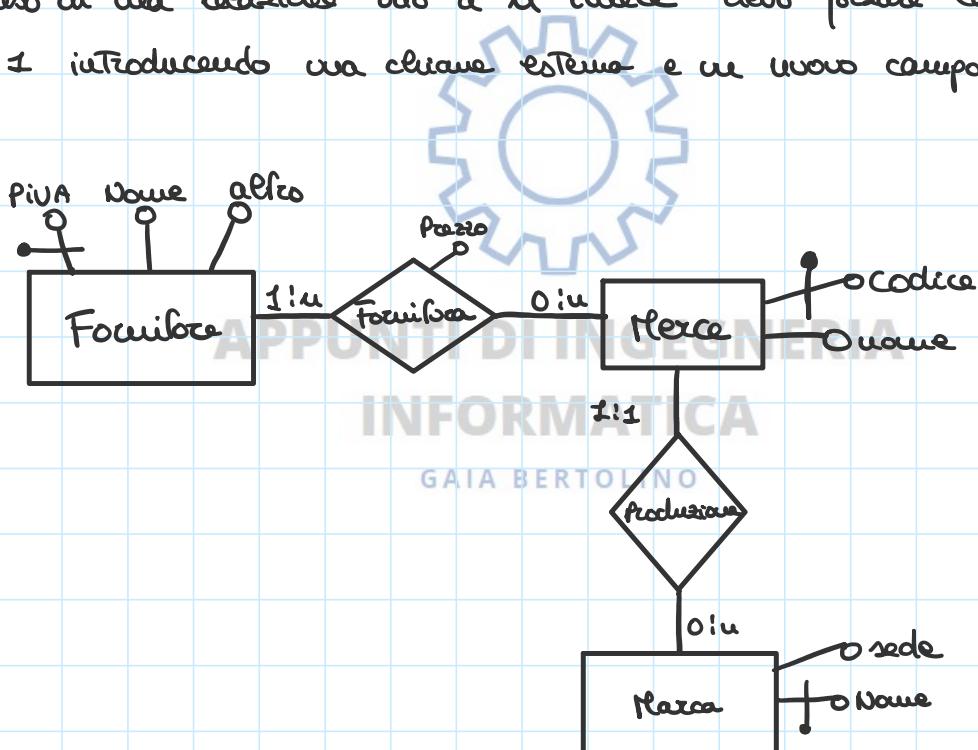


Composizione (fattura-anno, fattura-numero, prodotto, quantità, prezzo)

Composizione [fattura-anno, fattura-numero]  $\sqsubseteq_{FK}$  Fattura [anno, numero]

Inoltre, una chiave è molto importante in quanto prevede che quel campo dovrà essere compilato necessariamente (avendo una poterà essere NULL)

Nel caso di una riferzione uno a n invece devo forzare la presenza di quell'1 introducendo una chiave esterna e un nuovo campo. Ad esempio:



Fornitore (PiVA, nome, città, merci<sub>1</sub>, prezzo<sub>1</sub>)

Merce (codice, nome, marca)

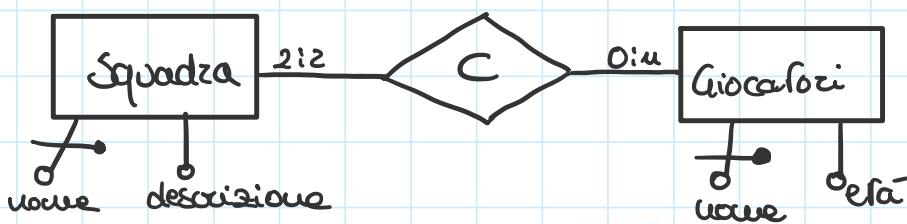
Fornitura (fornitore, merci, prezzo)

Fornitura [fornitore]  $\sqsubseteq_{FK}$  Fornitore [PiVA]

Fornitura [merci]  $\sqsubseteq_{FK}$  Merce [codice]

Tuttavia si può causare inconsistenza dunque si evita di introdurre quell' 1:u che necessita una soluzione del genere. Se invece capisci di doverlo realizzare, nel passare al modello logico allora si dice che si **ripara** il viuoto a 0:u e poi si rappresenta come tale nel modello relazionale.

Nel caso di relazioni numericamente friile esse si rappresentano come attributi:



Squadra (Nome, descrizione, gioc1, ruolo1, giocatore2, ruolo2)

Giocatore (Nome, Eta)

Squadra [gioc1]  $\sqsubseteq_{FK}$  Giocatore [Nome]

Squadra [gioc2]  $\sqsubseteq_{FK}$  Giocatore [Nome]

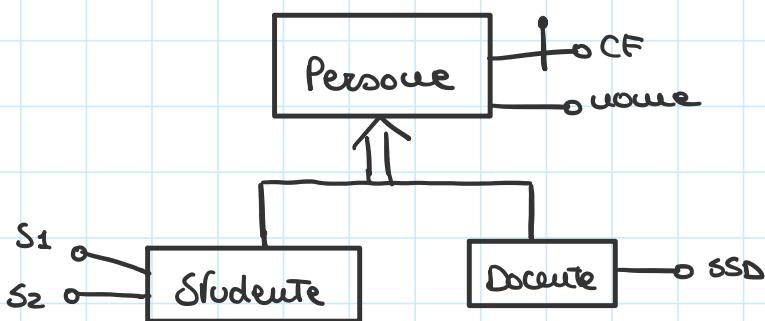
Tuttavia in questa soluzione si può verificare che un giocatore giochi in entrambi i ruoli. Tuttavia non è possibile effettuare la distinzione coi meccanismi di vuoto e dunque si esprime tale curcetto tramite linguaggio ufficiale.

Nel caso di cardinalità elevate (100:100) anziché operare come l'esempio precedente si **ripara** il valore a 0:u e dunque si realizza una tabella a parte.

Nel caso di vuoti del tipo 0:2 si pone il valore NULL sull'attributo a cui si riferisce.

Per quanto riguarda le subentity, il modello logico sarà del tipo:

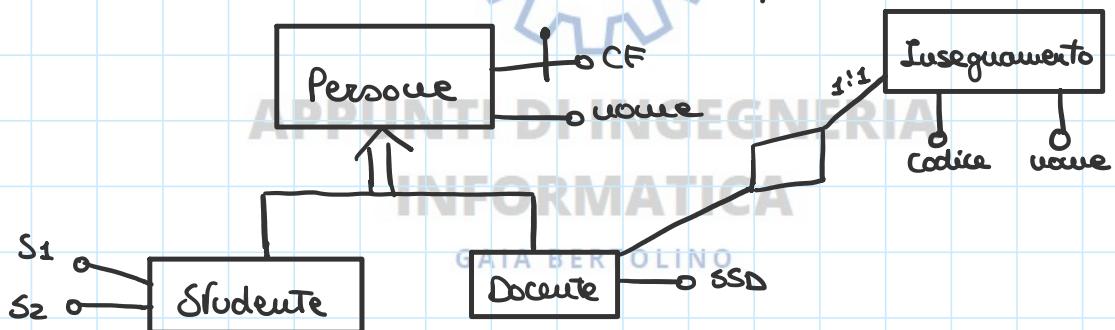
Per quanto riguarda le sottoentity, il modello logico sarà del tipo:



Personae (CF, nome, Stud / Doc, SSD, S<sub>1</sub>, S<sub>2</sub>)

Non essendo NULL otengo  
l'esclusività e la totalità

Se ad esempio avessi la situazione seguente cambierebbero i meccanismi in quanto l'esempio precedente va bene se non esistono ulteriori relazioni nelle sottoentity



Personae (CF, nome)

Studente (personae, S<sub>1</sub>, S<sub>2</sub>)

Studente [personae] ⊑ FK Personae [CF]

Docente (personae, SSD)

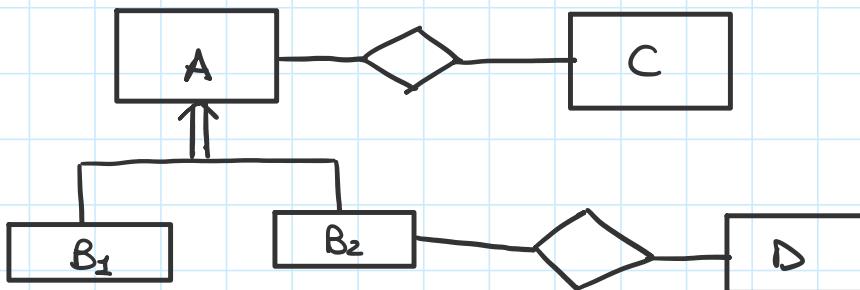
Insegnamento [docente] ⊑ Docente [personae]

Nella traduzione da E-R a relazioni della generalizzazione si applica un riflessamento a causa della insistenza della generalizzazione nella progettazione logica.

Essa può essere tradotta in tre modi:

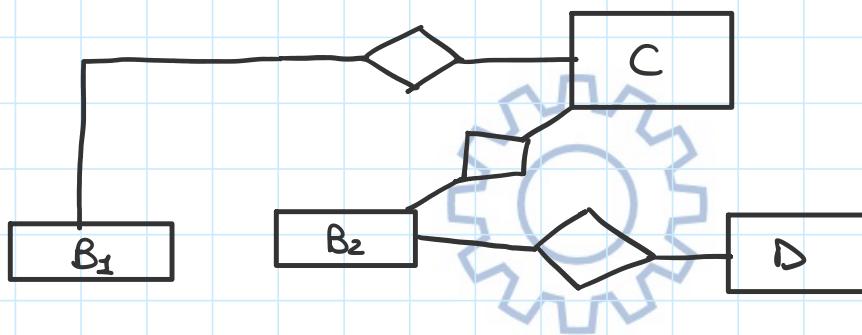
Esa può essere tradotta in tre modi :

Schema concettuale :

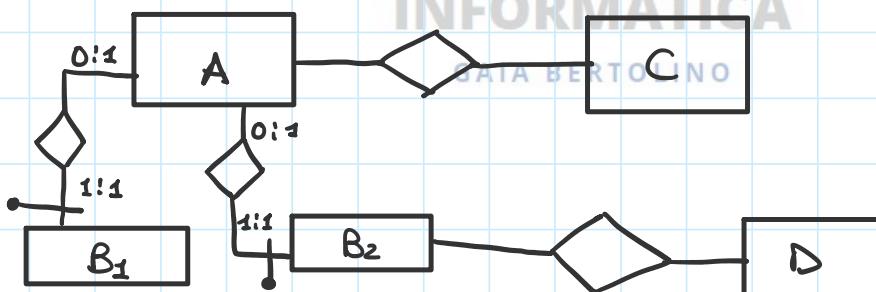


Schema logico :

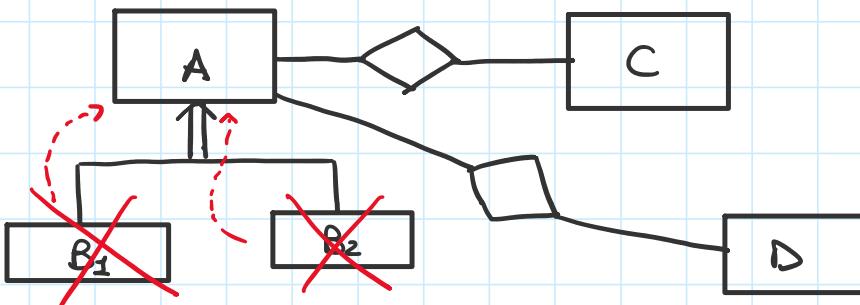
1)



2) In questo caso si perde di esclusività

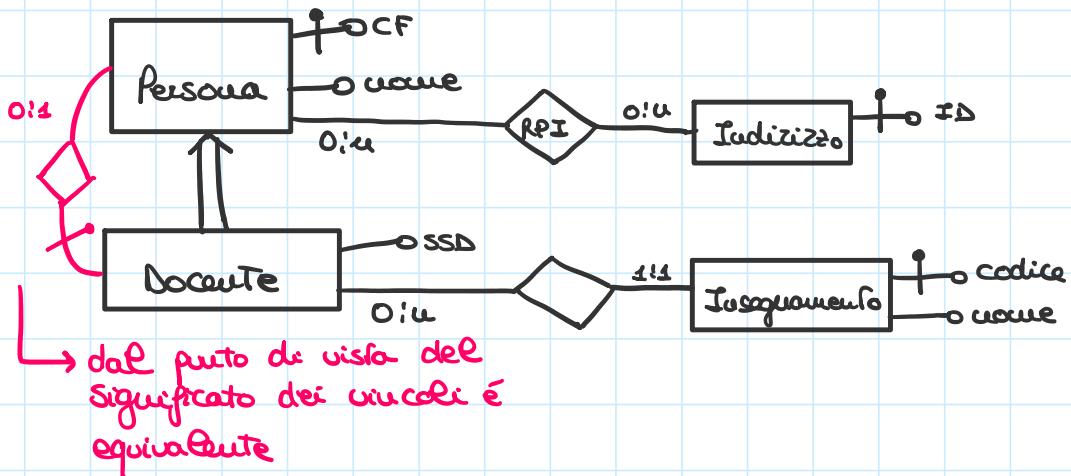


3)



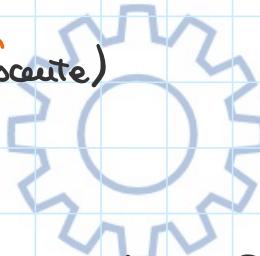
Invece la Traduzione di una ISA avviene nel seguente modo :

Schema concettuale:



### CHIAVI ESTERNE

Person (CF, nome)  
Docente (Person, SSD)  
Isegnaumento (codice, nome, docente)  
RPI (Person, Indirizzo)



Se tuttavia non ci fosse insegnamento si potrebbe scrivere  
Person (CF, nome, docente, SSD)

↳ BOOLEANO

ricordando però che docente non deve essere riferito  
con altro né avere troppi attributi.

# Esercitazione 1

lunedì 24 gennaio 2022 13:39



esercit0810

## Corso di Laurea in Ingegneria Informatica (DM 270) Esame di Basi di Dati 17 giugno 2019

### Esercizio 1: Progettazione concettuale e logica di una Base di Dati (40 minuti)

a) Si progetti una base di dati di supporto ad una società che si occupa di indagini demografiche. I **clienti** della società sono identificati dalla partita IVA e sono inoltre caratterizzati dalla denominazione, dalla sede, e dalla lista delle **indagini** che hanno commissionato. Ogni **indagine**, oltre che dal cliente committente, è caratterizzata da un numero progressivo, dalle date di inizio e di fine, dalla lista delle **domande** poste nell'intervista, e dalla lista delle **tipologie** di persone contattate per concedere l'intervista. Il numero progressivo serve ad identificare un'indagine rispetto alle altre commissionate dallo stesso cliente. → **classe** \*

Le **tipologie** di persone intervistate nel corso delle indagini sono identificate da un codice, e sono caratterizzate da una denominazione e da una descrizione. Ogni tipologia può essere contattata per 0..1..n persone di indagine, ed in ognuna di tali associazioni occorre indicare il numero di persone della tipologia che hanno concesso l'intervista.

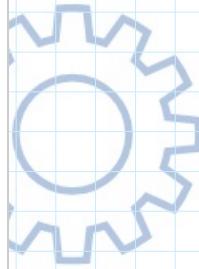
Le **domande** poste durante un'intervista sono caratterizzate da un numero progressivo, dal testo della domanda, e da tre possibili risposte. Una domanda può essere posta in una sola intervista e 1..1..n non possono esistere più domande nella stessa intervista aventi lo stesso numero progressivo. Nell'indicare l'appartenenza di una domanda ad un'intervista, deve essere possibile specificare quanti intervistati hanno dato la prima, la seconda, e la terza risposta. 0..1..1

A ciascuna indagine può essere associato al più un **documento** di sintesi, in cui vengono riassunti e commentati i risultati dell'indagine. Ogni documento di sintesi è caratterizzato dal titolo, dalla data in cui è stato completato e dall'**analizzatore** che lo ha redatto (l'analizzatore è una tipologia di dipendente, come appreso spiegato). Uno stesso documento di sintesi può riferirsi a più indagini. 0..1..n

Possono esistere documenti con lo stesso titolo, ma in tale caso differiscono per la data di completamento. → **classe**

I **dipendenti** della ditta sono identificati da un numero di matricola e sono anche caratterizzati dal nome, dal genere e dalla data di nascita. L'insieme dei dipendenti è partitionato in due classi: gli **intervistatori** e gli **analizzatori**. Gli intervistatori si occupano di effettuare le interviste, e per essi occorre memorizzare la lista delle indagini nell'ambito delle quali hanno effettuato le interviste. Si noti che un'indagine può essere effettuata da un numero qualunque di intervistatori. Gli analizzatori si occupano invece di redigere i documenti di sintesi, e ciascuno di essi è caratterizzato dalla lista di tali documenti da esso scritti. Si noti che ogni documento è scritto da un unico analizzatore.

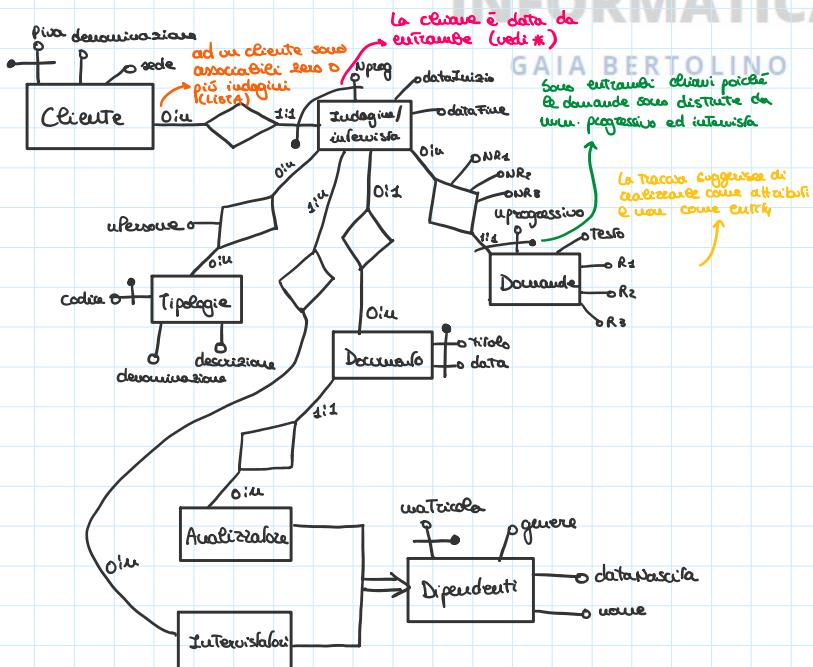
0..1..n



# APPUNTI DI INGEGNERIA INFORMATICA

GAIA BERTOLINO

## Progettazione concettuale



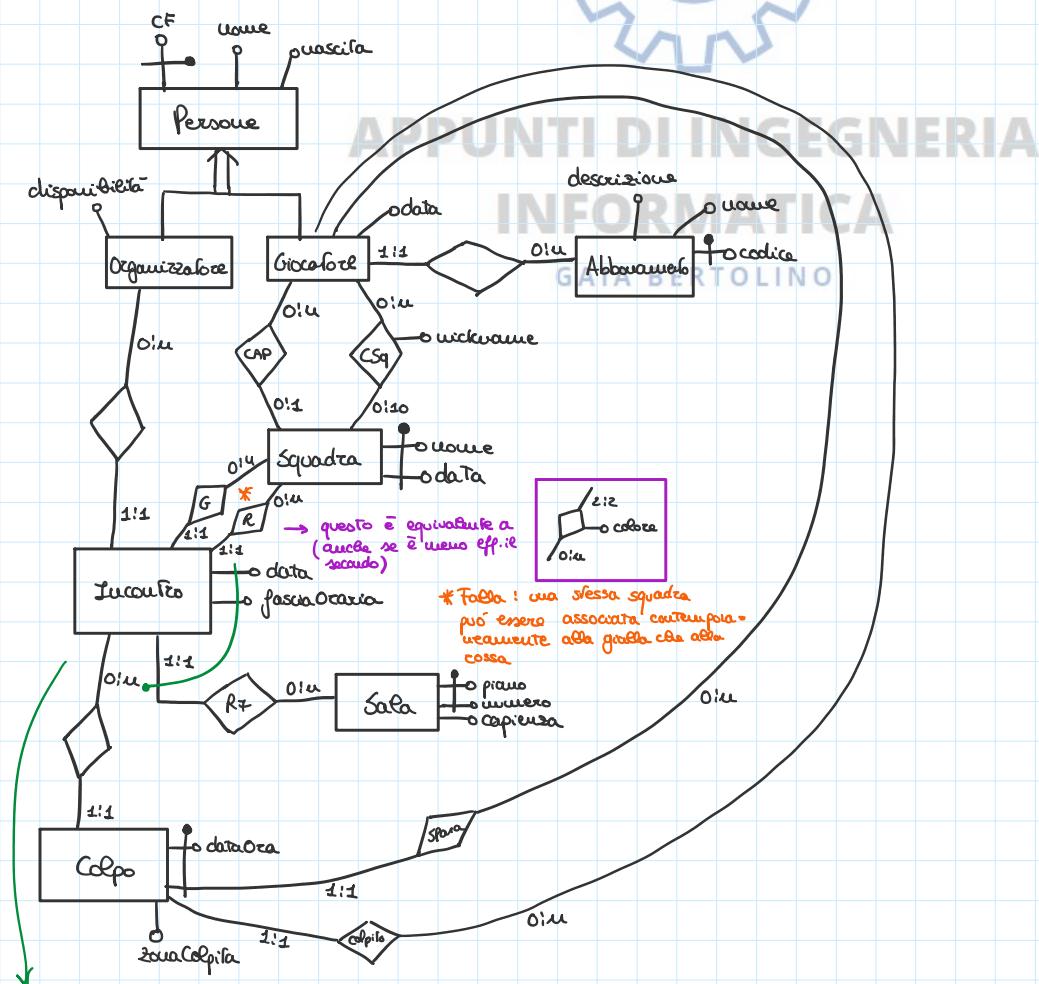
**Esercizio 1: Progettazione concettuale e logica di una Base di Dati (40 minuti)**

Si progetti una base di dati di supporto alla gestione di una sala giochi che organizza incontri di *LaserX*, un gioco in cui due squadre di giocatori muniti di pistole laser si affrontano e lo scopo di ciascun giocatore è quello di colpire più volte possibile i giocatori della squadra avversaria. Le *persone* coinvolte nello svolgimento degli incontri sono identificate dal codice fiscale e sono anche caratterizzate dal nome e dalla data di nascita. L'insieme delle persone è caratterizzato nelle due classi: *giocatore* i giocatori sono caratterizzati dal tipo di abbonamento che hanno sottoscritto con la sala giochi e dalla data di sottoscrizione. I tipi di abbonamento sono identificati dal codice e sono anche caratterizzati dal nome e da una descrizione. Uno stesso tipo di abbonamento può essere sottoscritto da un numero qualunque di *giocatori*. Gli organizzatori sono invece caratterizzati dall'informazione sulla loro disponibilità ad arbitrare incontri di *LaserX*. I giocatori si organizzano in *squadre*. Ogni squadra è caratterizzata dal nome, dalla data di fondazione, e dalla lista dei giocatori che la compongono (tale lista ha una cardinalità massima di 10). Non esistono squadre diverse aventi stesso nome e fondate nella stessa data. Inoltre, per ciascuna squadra può essere indicato il giocatore che ne è capitano. Si noti che un giocatore può partecipare ad un numero qualunque di squadre ed essere capitano di un numero qualunque di squadre. Nell'indicare l'afferenza di un giocatore ad una squadra, deve essere possibile specificare il nickname scelto dal giocatore nella *squadra* (un giocatore che partecipa a più squadre può scegliere nicknames diversi in tali squadre). Nell'organizzare un incontro di giocatori ad una squadra, a cui sono state assegnate le pistole di colore giallo, e da quella con le pistole di colore rosso. Ogni organizzatore può arbitrare un numero qualunque di incontri, ed ogni squadra può prendere parte ad un numero qualunque di incontri (sia come squadra rosa che gialla). Ciascuna sala è identificata da una coppia <piano, numero> ed è anche caratterizzata dalla capienza massima. Una stessa sala può ospitare un numero qualunque di incontri.

Per ogni incontro occorre inoltre tenere traccia della lista dei *colpi* di pistola andati a segno. Ogni colpo è caratterizzato dall'incontro nell'ambito del quale è avvenuto, dalla data/ora, dal giocatore che lo ha sparato, dal giocatore colpito, e dalla zona del corpo colpita. Nell'ambito dello stesso incontro, non possono esistere più colpi sparati contemporaneamente dallo stesso giocatore. Si noti che uno stesso giocatore può sparare un numero qualunque di colpi, ed essere colpito un numero qualunque di volte.

b) si definisca uno schema relazionale

## Progettazione concettuale



La clavare di incastro vuol essere specificata dalla traccia e dunque

specificata dalla traccia e dunque  
può essere chiesta oppure specificata  
a piacere.

Metto dunque data, fascia oraria e sala

### Progettazione logica

Sala (piano, numero, copertura)

\* essendo progettata una  
relazione 1:1 si ha che  
possiamo includere la relazione  
nei campi di riferimento. Dunque  
dovranno includere tutti i campi  
che sono chiave di sala

Incontro (data, fascia-oraria, sala-piano, sala-numero, squadraGruone, squadraGdata, squadraRgruone, squadraRdata, organizzatoreCF)

Incontro [sala-piano, sala-numero]  $\leq_{FK}$  Sala [piano, numero]

Incontro [squadraGruone, squadraGdata]  $\leq_{FK}$  Squadra [nuova, data]

Incontro [squadraRgruone, squadraRdata]  $\leq_{FK}$  Squadra [nuova, data]

Incontro [organizzatore]  $\leq$  Organizzatore [CF]

Squadra (nuova, data, <sup>\*NULL</sup>capitano,

Squadra [capitano]  $\leq_{FK}$  Giocatore [CF]

ComponentiSquadre (Giocatore, nuova\_squadra, fondazione\_squadra, nickname)

ComponentiSquadre [giocatore]  $\leq_{FK}$  Giocatori [CF]

ComponentiSquadre [nuova\_squadra, fondazione\_squadra]  $\leq_{FK}$  Squadra [nuova, data]

Abbraccamento (codice, nuova, descrizione)

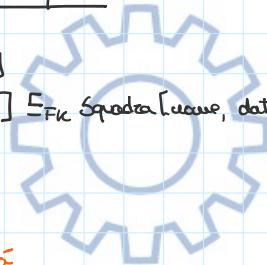
\* ciò esprime la relazione di O:1



La relazione 0:1:0 potrebbe essere  
interpretata come 1 o 0:1 che dunque  
richiedono le specificazioni NULL.

Tuttavia ciò comporta un abbassamento  
delle prestazioni del database.

Un'altra soluzione richiede il rafforzamento  
da 0:1:0 a 0:1:1, ottenendo dunque una  
1:1 a 0:



questa disposizione causa perdita di esclusività

Giocatore (CF, nuova, mascula, codice Abbraccamento)

Organizzatore (CF, nuova, mascula, disponibilità)

Giocatore [codice Abbraccamento]  $\leq_{FK}$  Abbraccamento [codice]

Colpo (dataOra, giocatoreSpara, giocatoreColpito, sociaColpita, dataIncontro, dataFasciaOraria, salaPiano, salaNumero)

Colpi [dataIncontro, dataFasciaOraria, salaPiano, salaNumero]  $\leq_{FK}$  Incontro [data, fascia-oraria, sala-piano, sala-numero]

Colpi [giocatoreColpito]  $\leq_{FK}$  Giocatore [CF]

Colpi [giocatoreSpara]  $\leq_{FK}$  Giocatore [CF]

## Interrogazione dei dati

mercoledì 26 gennaio 2022 13:57

### ALGEBRA RELAZIONALE

È un insieme di operatori che servono a manipolare il contenuto delle tabelle al fine di estrarre informazioni volute.

Il punto di partenza è una tabella (o detta anche **RELAZIONE**).

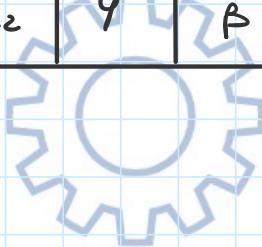
Sono date delle istruzioni di relazioni:

Fornitore

PiVA	Nome	Città
F <sub>1</sub>	A	CS
F <sub>2</sub>	B	RC
F <sub>3</sub>	C	CS

Marca

Codice	Nome	Marca
M <sub>1</sub>	X	d
M <sub>2</sub>	Y	B



### Operatore di selezione

$\sigma_{\text{tuple}}(\text{nome\_tabella})$

APPUNTI DI INGEGNERIA

INFORMATICA

GAIA BERTOLINO

È un operatore MONADICO ovvero prende un solo argomento e ha come pedice i valori delle tuple di interesse. Essa si comporta come un filo logico che analizza e filtra la relazione.

es.  $\sigma_{\text{Città}=\text{CS}}(\text{Fornitore})$  RISULTATO →

Fornitore

PiVA	Nome	Città
F <sub>1</sub>	A	CS
F <sub>3</sub>	C	CS

Le espressioni di selezione possono essere anche più complesse!

OR ↘

(città = 'CS' OR città = 'RC') AND nome = 'A'

$(CITTÀ = CS \vee PIVA \neq F1 \wedge NOME \neq C)$

AND

es.  $\delta(CITTÀ = RC \wedge NOME = A \wedge PIVA \neq 3)$

RISULTATO

Fornitore		
PIVA	NOME	CITTÀ
F1	A	CS
F2	B	RC

Gli operatori algebrici sono detti **RIENTRANTI** poiché dato un dominio restituisce un dominio ovvero data una tabella restituisce una tabella.

Ne consegue dunque la possibilità di innestare più selezioni:

es.  $\delta_{PIVA = F2} (\delta_{CITTÀ \neq RC} (Fornitore))$

EQUIVALENZA

$\delta_{PIVA = F2} (Fornitore)$   
CITTÀ  $\neq$  RC

## APPUNTI DI INGEGNERIA INFORMATICA

Le due operazioni risultano essere identiche anche nell'efficienza.

Ci interessa solo la correttezza e non l'efficienza che è demandata.

In ulteriore filigranno si può indicare quali delle colonne si vogliono visualizzare. Tale operatore è detto **PROIEZIONE**.

L'operatore sarà del tipo

$\Pi$  colonne (come Tabella)

es.  $\Pi_{NOME, CITTÀ} (Fornitore)$

RISULTATO

NOME	CITTÀ
A	Roma
B	Roma

Si dice che la proiezione **modifica l'axis** ovvero il numero di

Si dice che la proiezione **modifica l'area** ovvero il numero di colonne. Essa si distingue dalla **cardinalità** che è il numero delle tuple della tabella.

Tuttavia la proiezione ha anche la particolarità di modificare la cardinalità nel caso in cui si abbiano due o più tuple uguali.

es.

Fornitore

Piva	Nome	Città
F <sub>1</sub>	A	CS
F <sub>2</sub>	B	RC
F <sub>3</sub>	C	CS

$\Pi_{\text{Città}} (\text{Fornitore})$

↓ **RISULTATO**

Città
CS
RC

I due operatori sono comunitati!

es. Tabella con i nomi dei fornitori di RC

$\Pi_{\text{Nome}} [ d_{\text{Città}} = \text{RC} (\text{Fornitore}) ]$

↓ **RISULTATO**

Nome
RC

es. Marche delle merci il cui nome è X

$\Pi_{\text{marca}} [ d_{\text{Nome}} = \text{X} (\text{Merci}) ]$

Tuttavia l'ordine delle operazioni non è commutativo!

es.  $\Pi_{\text{marca}} [ d_{\text{Nome}} = \text{X} (\text{Merci}) ] \neq d_{\text{Nome} = \text{X}} [ \Pi_{\text{marca}} (\text{Merci}) ]$