



UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA



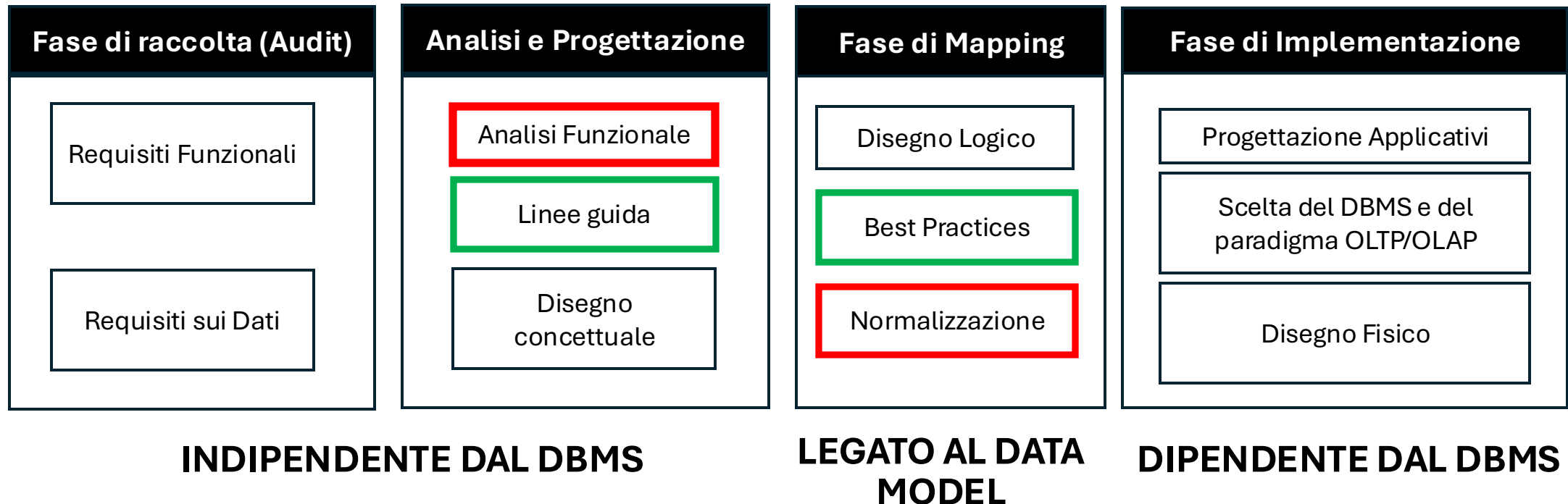
BASI DI DATI II

Progettare basi di dati di qualità: **regole di inferenza e dipendenze funzionali**

Anno 2024/2025
Prof. Genny Tortora
Dott. Luigi Di Biasi

Recap: Progettazione di un database generico

Dopo una visione informale, vediamo delle teorie e dei concetti formali per descrivere la “*bontà*” dei singoli schemi di relazione con maggiore precisione, usando le “*dipendenze funzionali*” e le “*forme normali*”.



Dalle buone pratiche alle regole «formali»

Possiamo considerare le buone pratiche di progettazione viste fino a questo punto come delle regole «dettate dall'esperienza» di altri.

Tuttavia, potremmo voler fare affidamento su indicatori di qualità ben definiti, con proprietà chiare e dimostrate, tali da garantirci la bontà del nostro lavoro quando utilizzati.

Dopo una visione informale, vediamo delle teorie e dei concetti formali per descrivere la “bontà” dei singoli schemi di relazione con maggiore precisione, usando le “dipendenze funzionali” e le “forme normali”.

Qui entrano in gioco le **dipendenze funzionali (FD)**, un concetto **che ci aiuta a descrivere in modo preciso le relazioni tra gli attributi di una tabella.**

Le FD ci aiutano non solo verificare se uno schema è progettato correttamente, ma anche a identificare potenziali problemi, **come la ridondanza o l'anomalia di aggiornamento.**

Dipendenza Funzionale

Immagiamo di essere nella nostra pizzeria preferita.

Immaginiamo di aver ordinato una pizza margherita e, come per magia, ogni volta che diciamo "**pizza margherita**", il cameriere ti porta esattamente quella (stessi ingredienti, stessa quantità di ingredienti, stessa forma → identificata da Y), senza variazioni.

Non importa quante volte ordiniamo dicendo «pizza margherita», il risultato sarà sempre lo stesso: pizza margherita (Y).

In un database **succede una cosa simile** con le dipendenze funzionali.

Se abbiamo una tabella con colonne come **CodicePizza** e **DescrizionePizza**, possiamo dire che **DescrizionePizza (Y)** dipende funzionalmente da **CodicePizza (X)** perché, **dato un certo CodicePizza**, otterremo sempre lo stesso **DescrizionePizza**.

Dipendenza Funzionale

Una dipendenza funzionale (FD) è un **vincolo** tra due insiemi di attributi del database.

{CodicePizza} -> {DescrizionePizza}

Supponiamo che **uno schema relazionale (R)** abbia **n** attributi **A_1, A_2, \dots, A_n** .

$$R = \{A_1, A_2, \dots, A_n\}.$$

Una dipendenza funzionale **tra due insiemi di attributi X e Y** è **denotata da $X \rightarrow Y$** .

X e Y sono sottoinsiemi di R!

$X \rightarrow Y$ specifica un vincolo sulle possibili tuple (*righe*) che possono formare una istanza di relazione **r** di R (*tabella*) .

Dipendenza Funzionale

Una dipendenza funzionale (FD) è un **vincolo** tra due insiemi di attributi del database.

{CodicePizza} -> {DescrizionePizza}

(Info sulla notazione) Ricordiamo che dato un insieme di attributi $X=\{X_1, X_2, \dots X_n\}$, una tupla $t_i[X]$ appartenente all'istanza di relazione r è l'insieme dei valori assunti da $\{X_1, X_2, \dots X_n\}$.

(Esempio) Se $R = \{\text{Numero Ordine}, \text{CodicePizza}, \text{DescrizionePizza}\}$, $X=\{\text{CodicePizza}\}$ e $Y=\{\text{DescrizionePizza}\}$ allora $t_1[X]=\{0\}$, $t_1[Y]=\{\text{Margherita}\}$, $t_1[XY] = \{0, \text{Margherita}\}$ e così via.

#IDO	CodicePizza	DescrizionePizza
1	0	Margherita
1	1	Capricciosa
1	0	Margherita
2	2	Diavola
2	0	Margherita

Dipendenza Funzionale




Una dipendenza funzionale (FD) è un **vincolo** tra due insiemi di attributi del database.

{CodicePizza} -> {DescrizionePizza}

L'esistenza di un vincolo di dipendenza funzionale $(X \rightarrow Y)$ implica che:

Se abbiamo una generica coppia di tuple (t_1, t_2) appartenenti a **r** e (AND) **se** entrambe le tuple hanno lo stesso valore per l'insieme di attributi **X** ($t_1[X] = t_2[X]$) allora **deve valere che**

$$t_1[Y] = t_2[Y].$$

#IDO	CodicePizza	DescrizionePizza
1	0 	Margherita
1	1	Capricciosa
1	0 	Margherita
2	2	Diavola
2	0 	Margherita

Dipendenza Funzionale

Una dipendenza funzionale (FD) è un **vincolo** tra due insiemi di attributi del database.

{CodicePizza} -> {DescrizionePizza}

Alternativamente, i valori della componente X di una tupla **determinano univocamente** (o **funzionalmente**) i valori della componente Y.

Cioè esiste una dipendenza funzionale da X a Y:

Y è funzionalmente dipendente da X

X è la **parte sinistra** della FD

Y è la **parte destra** della FD

#IDO	CodicePizza	DescrizionePizza
1	0	Margherita
1	1	Capricciosa
1	0	Margherita
2	2	Diavola
2	0	Margherita

Dipendenza Funzionale (Importante)

Una **chiave candidata** di una relazione R è un **insieme minimo** di attributi X che soddisfa due proprietà:

- **Unicità:** Non possono esistere due tuple con lo stesso valore di X;
- **Minimalità:** Nessun sottoinsieme proprio di X ha la proprietà di unicità.

CP	DescrizionePizza
0	Margherita
1	Capricciosa
2	Diavola

Chiaramente CP è la nostra chiave candidata. Se $X=\{PC\}$ è una **chiave candidata**, significa che ogni tupla della relazione è **univocamente identificata** dal valore di X (è una dipendenza funzionale!).

Di conseguenza, questo implica che **qualsiasi altro attributo della relazione** dipende funzionalmente da X.

Dipendenza Funzionale (Importante)

La dipendenza funzionale è **una proprietà della semantica degli attributi**.

Consideriamo nuovamente lo schema EMP_PROJ non propriamente ottimale.

EMP_PROJ

<u>SSN</u>	<u>PNUMBER</u>	HOURS	ENAME	PNAME	PLOCATION
------------	----------------	-------	-------	-------	-----------

In questo schema esistono almeno 3 dipendenze funzionali (da ora identificate come FD). Quali sono?

SSN = Codice Fiscale;

PNUMBER = ID Progetto;

HOURS = Ore allocate sul progetto;

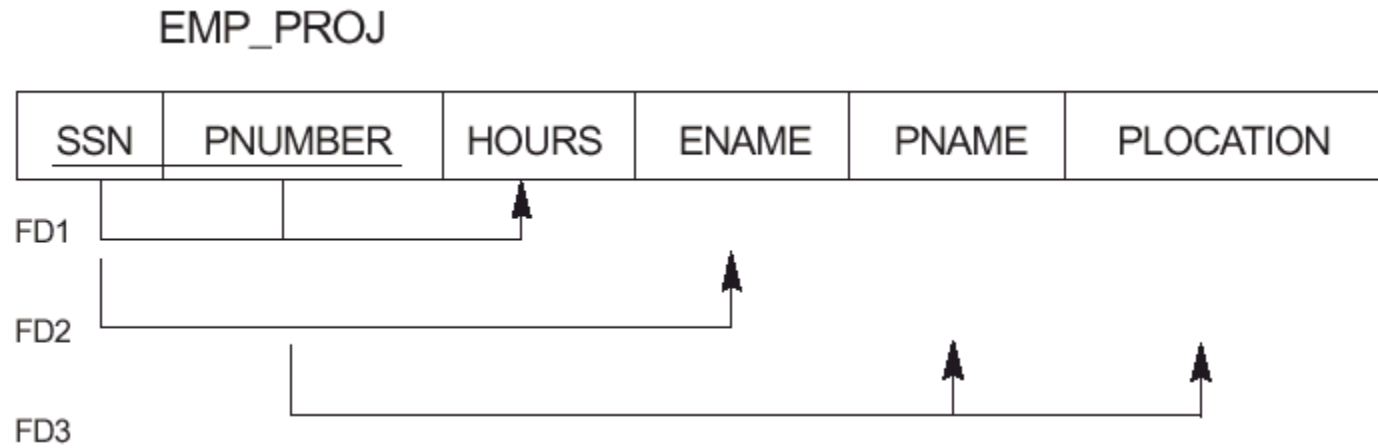
ENAME = Nome dell'impiegato;

PNAME = Nome del progetto;

PLOCATION = Dove si svolge il progetto?

Dipendenza Funzionale (**Importante**)

La dipendenza funzionale è una **proprietà della semantica** degli attributi.



Dipendenza Funzionale (**Importante**)

Poiché **una dipendenza funzionale è una proprietà della semantica degli attributi:**

- **È anche una proprietà dello schema di relazione R (è una proprietà di R, deve valere per ogni istanza di R!)**
- Dato che è una proprietà di R (intensione) e non di un particolare stato di relazione (estensione r di R) allora, deve valere per ogni istanza di r!

R = {ID Ordine, CodicePizza, DescrizionePizza}

FD = {CodicePizza} -> {DescrizionePizza}

Se vendiamo il nostro gestionale per pizzerie basato sullo schema R (intensione) a 10 pizzerie distinte (estensione), la FD varrà sempre, per ogni pizzeria poiché FD è una proprietà di R e non delle istanze (i distinti gestionali delle pizzerie).

Dipendenze Funzionali (**Importante**)

Attenzione: Le **estensioni valide** o **stati di relazione legali** sono tutte quelle estensioni di relazione $r(R)$ che soddisfano i vincoli di FD. Se una qualche estensione viola FD (ad esempio, se in uno dei gestionali viene rimosso il vincolo di chiave ID Ordine, Codice Pizza) allora tale istanza non è legale.

Importantissimo: Le FD **non possono essere inferite** automaticamente da un'estensione di relazione r , **ma devono essere definite esplicitamente** da chi conosce la semantica degli attributi.

Chiaramente, esistono dei tool per cercare dipendenze funzionali (immaginatela come la ricerca di una correlazione multi-attributo). Questi tool rientrano nell'ambito del **Data Profiling**.

Per i curiosi: <https://github.com/HPI-Information-Systems/Metanome>

Dipendenze Funzionali

Importantissimo: Una FD **non può essere inferita** automaticamente da un'estensione di relazione **r**.

TEACH		
TEACHER	COURSE	TEXT
Smith	Data Structures	Bartram
Smith	Data Management	Al-Nour
Hall	Compilers	Hoffman
Brown	Data Structures	Augenthaler

E' presente una FD?

Regole di inferenza

Torniamo di nuovo nella nostra pizzeria preferita.

Eravamo nella situazione in cui come per magia, ogni volta che diciamo "pizza margherita", il cameriere ci porta esattamente quella (**stessi ingredienti, stessa quantità di ingredienti, stessa forma**), **senza variazioni**.

Ora, supponiamo che il cameriere **conosca anche alcune regole** su come combinare le informazioni:

- Se una pizza è bianca (senza pomodoro), allora **ha sempre** una base con mozzarella;
- Se una pizza ha mozzarella e gorgonzola, allora **è** una "4 formaggi".

Supponiamo che un cliente ordini una pizza bianca con mozzarella e gorgonzola.

Il cameriere può dedurre automaticamente che:

- ***siccome è bianca*** → deve avere la base con mozzarella;
- ***siccome ha mozzarella*** e gorgonzola → è una 4 formaggi.

Regole di inferenza

«Regole»

- Se una pizza è bianca (senza pomodoro), allora **ha sempre** una base con mozzarella;
- Se una pizza ha mozzarella e gorgonzola, allora **è** una "4 formaggi".

«**Evento**» Un cliente ordina una pizza bianca con mozzarella e gorgonzola.

«Inferenza»

- **siccome è bianca** → deve avere la base con mozzarella;
- **siccome ha mozzarella** e gorgonzola → è una 4 formaggi.

Questo è esattamente il concetto delle regole di inferenza nel mondo delle dipendenze funzionali nei database!

Usiamo alcune regole di base per derivare nuove informazioni, proprio come il cameriere deduce che la pizza è una 4 formaggi senza bisogno che il cliente lo dica esplicitamente.

Dipendenze funzionali e Regole di inferenza

Le **dipendenze funzionali** ci dicono **quali informazioni sono collegate tra loro**.

Le **regole di inferenza** ci permettono di **usare queste connessioni per scoprire nuove informazioni**.

Attenzione a non confondere l'uso delle regole di inferenza con il data mining!

Regole di inferenza

Consideriamo $F = \{\rightarrow\}$ l'insieme delle dipendenze funzionali di uno schema di relazione R .

In un mondo ideale, **il progettista dello schema specifica TUTTE le dipendenze funzionali.**

Tuttavia, nella realtà, il progettista specifica solo le FD semanticamente «ovvie».

Di conseguenza, possiamo ritrovarci in una situazione nella quale nelle le istanze di relazione legali, **potrebbero valere numerose altre dipendenze funzionali.**

L'insieme delle FD non «ovvie» possono essere **inferite** (o dedotte) da F utilizzando **le regole di inferenza.**

L'insieme di tali dipendenze (dedotte) è detto **chiusura di F** , ed è denotato da F^+ .

Regole di inferenza

Esistono **tre regole di inferenza fondamentali**, dette «Regole di Armstrong» oppure «Assiomi di Armstrong».

Tali regole vengono dette **corrette e complete** perché **garantiscono che possiamo dedurre esattamente tutte e sole le dipendenze funzionali valide** di un dato schema relazionale.

Correttezza: qualsiasi dipendenza funzionale derivata attraverso una regola di Armstrong è effettivamente valida in tutte le possibili istanze dello schema relazionale.

Completezza: Le regole di Armstrong ci permettono di dedurre tutte le dipendenze funzionali valide basandoci solo sull'insieme iniziale di dipendenze. In altre parole, la chiusura di F (F^+ , l'insieme di tutte le possibili dipendenze inferibili da F) è calcolabile usando solo le regole di Armstrong.

Ergo: Ci bastano queste tre regole per inferire tutte le FD e stare tranquilli che siano sempre valide in tutte le istanze di r ... tuttavia è buona norma conoscere anche le altre!

Importante: Una FD $X \rightarrow Y$ è inferita da un insieme di dipendenze F su R se $X \rightarrow Y$ vale in ogni stato di relazione r che è estensione legale di R .

Notazioni

$F \models X \rightarrow Y$ denota che la FD $X \rightarrow Y$ è inferita da F .

Notazioni:

$\{X,Y\} \rightarrow Z$ è abbreviato in $XY \rightarrow Z$

$\{X,Y,Z\} \rightarrow \{U,V\}$ è abbreviato in $XYZ \rightarrow UV$

Regola Riflessiva (di Armstrong)

Regola Riflessiva (IR1) (Reflexive rule)

Se $Y \supseteq X$ allora $X \rightarrow Y$: Un insieme di attributi determina sempre un suo sottoinsieme.

Matricola	Nome	Cognome	Data Nascita
0	Tony	F	17 maggio 1991
1	Achille	Lauro	11 luglio 1990
2	Clara	Boh	25 ottobre 1999

Se consideriamo l'insieme {Matricola, Nome}, possiamo applicare la regola riflessiva:

- {Matricola, Nome} \rightarrow {Nome} perché Nome è già incluso nel lato sinistro.
- {Matricola, Nome} \rightarrow {Matricola} perché Matricola è già incluso nel lato sinistro.

Augmentation Rule (di Armstrong)

Regola dell'aumento (IR2) (Augmentation rule)

$X \rightarrow Y \models XZ \rightarrow YZ$: Se una dipendenza funzionale è valida tra X e Y , allora aggiungere lo stesso insieme di attributi Z a entrambi i lati della dipendenza la rende ancora valida.

Matricola	Nome	Cognome	Data Nascita
0	Tony	F	17 maggio 1991
1	Achille	Lauro	11 luglio 1990
2	Clara	Boh	25 ottobre 1999

Supponiamo di avere $\text{Matricola} \rightarrow \text{Nome, Cognome}$.

Allora sarà vero anche che $\text{Matricola, DataNascita} \rightarrow \text{Nome, Cognome, DataNascita}$

Regola Transitiva (di Armstrong)

Regola della transitività (IR3) (Transitivity rule)

$\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$: Se una dipendenza funzionale è valida tra X e Y e se una dipendenza funzionale è valida tra Y e Z , allora è valida anche la dipendenza funzionale tra X e Z !

Cd Fiscale	Nome	Cognome	Data Nascita	Età
0	Tony	F	17 maggio 1991	33
1	Achille	Lauro	11 luglio 1990	34
2	Clara	Boh	25 ottobre 1999	25

Cd Fiscale \rightarrow Data Nascita

Data Nascita \rightarrow Età allora: Cd Fiscale \rightarrow Età

Regole di inferenza (esercizio)

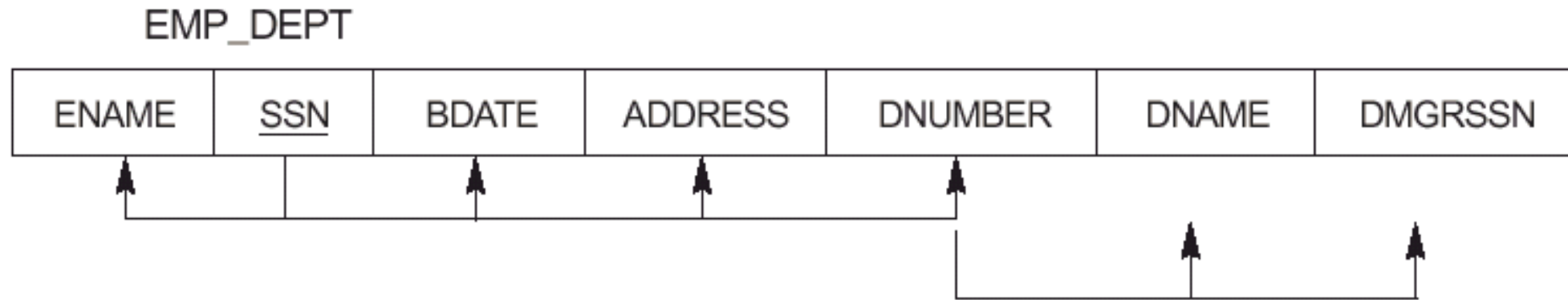
Consideriamo nuovamente lo schema EMP_PROJ non propriamente ottimale. Il progettista (ormai licenziato) di questo schema ci fornisce anche l'elenco delle FD «ovvie». **Quali altre FD possiamo inferire a partire da F?**



$F = \{$
 $SSN \rightarrow \{ENAME, BDATE, ADDRESS, DNUMBER\},$
 $DNUMBER \rightarrow \{DNAME, DMGRSSN\}$
 $\}$

Regole di inferenza (esercizio)

Consideriamo nuovamente lo schema EMP_PROJ non propriamente ottimale. Il progettista (ormai licenziato) di questo schema ci fornisce anche l'elenco delle FD «ovvie». **Quali altre FD possiamo inferire a partire da F?**



F = {

SSN \rightarrow {ENAME, BDATE, ADDRESS, DNUMBER},
DNUMBER \rightarrow {DNAME, DMGRSSN}

}



SSN \rightarrow {DNAME, DMGRSSN}
SSN \rightarrow SSN
DNUMBER \rightarrow DNAME

Regola della decomposizione

Regola Riflessiva (IR4) (Decomposition rule)

$\{X \rightarrow YZ\} \models X \rightarrow Z$: Se esiste una dipendenza funzionale del tipo $X \rightarrow YZ$ allora esistono anche:

- $X \rightarrow Y$;
- $X \rightarrow Z$

Cd Fiscale	Nome	Cognome	Data Nascita	Età
0	Tony	F	17 maggio 1991	33
1	Achille	Lauro	11 luglio 1990	34
2	Clara	Boh	25 ottobre 1999	25

Se esiste $\text{Matricola} \rightarrow \text{Nome, Cognome}$ allora esistono:

- $\text{Matricola} \rightarrow \text{Nome}$
- $\text{Matricola} \rightarrow \text{Cognome}$

Regola della decomposizione (importante)

Regola Riflessiva (IR4) (Decomposition rule)

$\{X \rightarrow YZ\} \models X \rightarrow Z$: a sinistra nella regola di decomposizione **non è necessario che ci sia per forza un unico attributo**.

La regola di decomposizione si applica anche se a sinistra c'è un insieme di attributi (piuttosto che un singolo attributo), purché la dipendenza funzionale sia valida.

Se è valida $\{\text{Matricola}, \text{Corso}\} \rightarrow \{\text{Nome}, \text{Cognome}\}$

Allora sono valide:

- $\{\text{Matricola}, \text{Corso}\} \rightarrow \text{Nome}$
- $\{\text{Matricola}, \text{Corso}\} \rightarrow \text{Cognome}$

Regola additiva (o dell'unione)

Regola additiva (IR5) (Union or additive rule)

$\{X \rightarrow Y, X \rightarrow Z\} \models X \rightarrow YZ$: La regola di unione (union rule) afferma che, se sappiamo che X determina sia Y che Z, allora X deve determinare anche l'insieme YZ (cioè Y e Z presi insieme come un'unica entità).

In altre parole, se abbiamo **due dipendenze funzionali separate**

$F = \{X \rightarrow Y, X \rightarrow Z\}$ allora le possiamo inferire che **$X \rightarrow YZ$**

Esempio, sappiamo che:

- **Matricola \rightarrow Nome** (La **Matricola** determina il **Nome**)
- **Matricola \rightarrow Cognome** (La Matricola determina il Cognome)

Allora è vero che **Matricola \rightarrow Nome, Cognome**

Regola pseudo-transitiva

Regola pseudotransitiva (IR6) (Pseudo-transitive rule)

$\{X \rightarrow Y, WY \rightarrow Z\} \models WX \rightarrow Z$: questa regola riguarda una combinazione di **transitività** e **aggiunta**.

Se abbiamo le seguenti che $X \rightarrow Y$ e che $WY \rightarrow Z$:

La prima dipendenza $X \rightarrow Y$ ci dice che X determina Y. Se X determina Y, possiamo sostituire Y con X in altre dipendenze che coinvolgono Y. In particolare, nella seconda dipendenza $WY \rightarrow Z$ possiamo sostituire Y con X perché $X \rightarrow Y$

La seconda parte della dipendenza $WY \rightarrow Z$ ci dice che WY determina Z. Se WY determina Z, e sappiamo che $X \rightarrow Y$, possiamo "aggiungere" X a W e concludere che WX determina Z. In altre parole, X (che determina Y) e W **insieme** determinano Z.

Regola pseudo-transitiva (esempio)

Regola pseudotransitiva (IR6) (Pseudo-transitive rule)

$$\{X \rightarrow Y, WY \rightarrow Z\} \models WX \rightarrow Z$$

Supponiamo di avere:

- **Matricola (X) \rightarrow Nome (Y)** (La **Matricola** determina il **Nome**);
- **Corso (W), Nome (Y) \rightarrow Voto (Z)** (Il Corso e il Nome insieme determinano il Voto)

Allora sarà vero che:

- Corso (**W**), Matricola (X), \rightarrow Voto (Z)

Regole di Basi di Dati 2

R1: Il segreto dell'insegnamento è quello di mostrare di conoscere da tutta una vita quello che hai appena imparato questa mattina.
(Anonimo)

R2: I migliori maestri sono quelli che ti indicano dove guardare, ma non ti dicono cosa vedere. (Alexandra K. Trenfor)

Pausa

Dimostrazioni delle IR

R3: Le dimostrazioni delle IR1, IR2, IR3, IR4, IR5 e IR6 **fanno parte del corso e potrebbero essere domande all'orale.**

Data la R1, vi vengono assegnate come compiti per casa!

Data la R2, verranno pubblicate sulla piattaforma (forse).

Il nuovo testamento del progettista relazionale

Misure di bontà informali

1. Fai in modo di avere una **semantica chiara degli attributi**;
2. **Riduci** il numero di valori **ridondanti** nelle tuple;
3. **Riduci** il numero di valori **null** nelle tuple;
4. **Non consentire tuple spurie**;

Linee guida estese!

5. Disegna uno schema di relazione facile da spiegare;
6. Disegna gli schemi di relazione in modo che non possano accadere insertion, deletion o modification anomalies.
7. Progetta gli schemi in modo da poter eseguire JOIN senza generare tuple spurie.

Bontà «Formali»

8. Definisci le dipendenze funzionali semanticamente «ovvie»
9. Calcola tutte le FD inferibili!
10. Normalizza lo schema (assicurandoti che sia dependencies preserving)!

IX° comandamento: «Calcola tutte le FD inferibili!»

Tipicamente, il progettista del db **prima specifica le dipendenze funzionali F a partire dalla semantica degli attributi** e poi usa le regole di inferenza di Armstrong per inferire dipendenze funzionali aggiuntive.

Metodo:

1. determinare ogni insieme di attributi X che appare come parte sinistra di qualche dipendenza funzionale in F.
2. usare le regole di Armstrong per determinare l'insieme di tutti gli attributi dipendenti da X.

IX° comandamento: «Calcola tutte le FD inferibili!»

Tipicamente, il progettista del db **prima specifica le dipendenze funzionali F a partire dalla semantica degli attributi** e poi usa le regole di inferenza di Armstrong per inferire dipendenze funzionali aggiuntive.

L'obiettivo è di ottenere l'insieme completo di tutte le dipendenze funzionali implicate da quelle iniziali (usando le regole di inferenza di Armstrong), in modo da garantire che il modello del database sia coerente e che tutte le relazioni tra gli attributi siano correttamente descritte.

IX° comandamento: «Calcola tutte le FD inferibili!»

Metodo: Identificare gli insiemi di attributi che appaiono come parte sinistra delle dipendenze funzionali (ovvero, **determinare ogni insieme di attributi X che appare come parte sinistra di qualche dipendenza funzionale in F.**)

Applicare le regole di Armstrong per inferire le dipendenze funzionali aggiuntive.

Continuare a inferire tutte le dipendenze funzionali che possono essere dedotte fino a ottenere un set completo di dipendenze per il database.

Quando ci fermiamo? Il processo termina quando non è più possibile inferire nuove dipendenze funzionali, e quindi l'insieme delle dipendenze funzionali è completo (**quando non possiamo applicarne IR**).

IX° comandamento: «Calcola tutte le FD inferibili!»

$X^+ := X;$

repeat

$\text{old}X^+ := X^+;$

for each functional dependency $Y \rightarrow Z$ in F **do**

if $X^+ \supseteq Y$

then $X^+ := X^+ \cup Z;$

until ($\text{old}X^+ = X^+$);

IX° comandamento: «Calcola tutte le FD inferibili!»

Inizializzazione:

L'algoritmo inizia con $X^+ := X$. In altre parole, la chiusura di X inizialmente è semplicemente l'insieme X stesso.

Condizione di stop: Viene eseguito un ciclo che continua fino a quando la chiusura di X non cambia più. Questo avviene quando X^+ non aumenta più, il che significa che abbiamo trovato tutti gli attributi che dipendono da X tramite le dipendenze funzionali.

IX° comandamento: «Calcola tutte le FD inferibili!»

Ciclo iterativo:

Per ogni dipendenza funzionale $Y \rightarrow Z$ in F :

- **se gli attributi Y sono contenuti nella chiusura corrente X ALLORA aggiungiamo gli attributi Z a X^+**

Esercizio (fast)

Supponiamo di avere le seguenti dipendenze funzionali

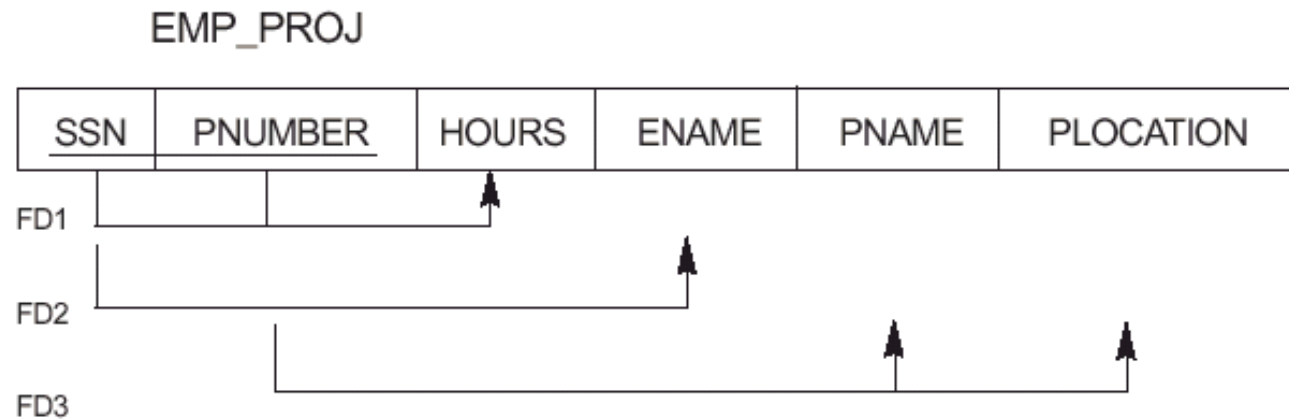
$F: = \{A \rightarrow B; B \rightarrow C; A \rightarrow C\}$ e $X = \{A\}$

Quale è la chiusura di X ?

IX° comandamento: «Calcola tutte le FD inferibili!»

Esercizio (simulazione prova d'esame)

Dato lo schema di relazione EMP_PROJ



Sappiamo che

$F = \{$
 $\{SSN, PNUMBER\} \rightarrow HOURS,$
 $SSN \rightarrow ENAME,$
 $PNUMBER \rightarrow \{PNAME, PLOCATION\}$
 $\}$

Calcolare la chiusura di F

Equivalenza di insieme di FD

Definizione:

Un insieme F di dipendenze funzionali **copre** un altro insieme E di dipendenze funzionali, se ogni DF in E è presente anche in F^+ , cioè se ogni dipendenza in E può essere inferita a partire da F .

Definizione:

Due insiemi E ed F di dipendenze funzionali sono **equivalenti** se $E^+ = F^+$; ossia E è equivalente ad F se sussistono entrambe le condizioni: E **copre** F e F **copre** E .

Si può determinare se F **copre** E calcolando X^+ *rispetto* a F per ogni $DF X \rightarrow Y$ in E , e quindi verificando se questo X^+ comprende gli attributi presenti in Y .