Dipendenze funzionali e normalizzazione per database relazionali

Lucia Ferrari lucia02.ferrari@edu.unife.it

Introduzione

Introduzione

Abbiamo bisogno di metodi per misurare la bontà di un modello relazionale.

Due livelli di bontà:

1. Logico: interpretazione dello schema relazionale e del significato degli attributi da parte dell'UTENTE.
2. Implementativo: come le tuple delle relazioni sono memorizzate e aggiornate.

Linee guida informali

Linee guida informali

1. La semantica degli attributi deve essere chiara, evitando ambiguità
2. Ridurre la ridondanza di informazioni nelle tuple
3. Ridurre il più possibile la presenza di valori NULL nelle tuple
4. Togliere la presenza di tuple spurie (= tuple con informazioni non corrette)

1) semantica chiara degli attributi nelle relazioni

Linee guida:
- Dare dei nomi corretti agli attributi, non: uno, due, tre, quattro... - Disegnare uno schema relazionale in modo che sia facile spiegarne il significato.
- Non combinare più entità e associazioni in una singola relazione, altrimenti diventa complicato capirne il significato.

In generale

Memorizzare join naturali tra relazioni crea più anomalie:

- Anomalie di aggiornamento: devo aggiornare più tuple che contengono lo stesso valore
- Anomalie di inserimento: non riesco ad inserire l'informazione perchè non è completa
- Anomalie di eliminazione: elimino anche informazioni importanti

Linee guida: lo schema relazionale deve essere creato in modo che non siano presenti anomali, se questo non è possibile devono essere specificate chiaramente per fare in modo che gli aggiornamento del database operino correttamente.

2) Problemi relativi alla ridondanza

Considerando la seguente relazione:

Impiegato	Salario	Progetto	Budget	Funzione
Brown	20	Mars	2	Technician
Green	35	Jupiter	15	Designer
Green	35	Venus	15	Designer
Hoskins	55	Venus	15	Manager
Hoskins	55	Jupiter	15	Consultant
Hoskins	55	Mars	2	Consultant
Moore	48	Mars	2	Manager
Moore	48	Venus	15	Designer
Kemp	48	Venus	15	Designer
Kemp	48	Jupiter	15	Manager

La chiave è creata dagli attributi Impiegato e Progetto.

Anomalie della relazione precedente:

- Il valore del salario di ogni impiegato è ripetuto in più tuple causando RIDONDANZA.
- Se il salario di un impiegato cambia è necessario modificare il valore in tutte le tuple corrispondenti (anomalie di modifica).
- Se un impiegato smette di lavorare sui progetti ma non lascia l'azienda tutte le tuple delle sue informazioni sono perse (compreso nome e salario), causando un anomalia di eliminazione.
- Se abbiamo le informazioni di un impiegato questo non può essere inserito finchè non gli è assegnato un progetto (anomalia di inserimento).

La soluzione ai problemi è realizzare due diverse relazioni, una riguardante le informazioni dell'impiegato e una riguardante le informazioni sui progetti (ed eventualmente una terza relazione per determinare quale impiegato lavora su quale progetto).

3) Valori NULL nelle tuple

In alcune relazioni possono essere raggruppati molteplici attributi anche se non sono applicati a tutte le tuple della relazione (es: nel caso delle ereditarietà). Le tuple che non hanno un valore nell'attributo memorizzano un NULL.

Linee guida: per quanto possibile è meglio evitare di inserire attributi nella relazione che presentano frequentemente valori NULL. Se i NULL sono inevitabili bisogna fare in modo che siano applicati sono in casi eccezionali e non nella maggioranza delle tuple per evitare sprechi di memoria.

Generazione di tuple spurie

Le tuple spurie sono delle tuple che rappresentano delle informazioni NON valide. Ad esempio se il database relazionale ha un design sbagliato si possono generare degli errori durante le operazioni di JOIN.

Linee guida: creare schemi relazionali in modo che possa essere eseguito il JOIN su attributi che siano effettivamente collegati tra loro (primary key con foreign key), in questo c'è la garanzia che non vengono create tuple spurie.

Teoria formale: Dipendenze Funzionali

Dipendenze funzionali

Per sviluppare la teoria formale supponiamo che il database sia rappresentato da un singolo schema relazionale $R = \{A_1, \ldots, A_n\}$.

Una dipendenza funzionale (FD), denotata da $X \rightarrow Y$, tra due insiemi non vuoti di attributi X, $Y \subseteq R$ indica che per ogni tupla t_1 t_2 in ognuna delle istanze della possibili istanze $r(R)$, vale il seguente vincolo:

$$\text{se } t_1[X] = t_2[X], \text{ allora } t_1[Y] = t_2[Y].$$

Si dice che X determina funzionalmente Y se il valore di X determina specificamente il valore di Y.
Si indica con: $X \rightarrow Y$

Osserviamo che:

- Se X è una chiave candidata allora $X \rightarrow Y$ è valida per ogni insieme non vuoto di attributi Y di R
- Se $X \rightarrow Y$ in R, non vuol dire che necessariamente che $Y \rightarrow X$.

Una FD è una proprietà del database

Il principale uso delle dipendenze funzionali è descrivere ulteriormente lo schema relazionale, specificando i vincoli sui suoi attributi che deve mantenere in ogni momento.

Es:

Impiegato	Salario	Progetto	Budget	Funzione
Brown	20	Mars	2	Technician
Green	35	Jupiter	15	Designer
Green	35	Venus	15	Designer
Hoskins	55	Venus	15	Manager
Hoskins	55	Mars	2	Consultant
Moore	48	Mars	2	Manager
Moore	48	Venus	15	Designer
Kemp	48	Venus	15	Designer
Kemp	48	Jupiter	15	Manager

Le dipendenze funzionali sono:

$Impiegato \rightarrow Salario$

$Progetto \rightarrow Budget$

$Impiegato, Progetto \rightarrow Funzione$

esempio: se $t_1[Impiegato] = t_2[Impiegato]$, allora $t_1[Salario] = t_2[Salario]$.

Ovvero, per le tuple t6 e t7 -> Moore = Moore quindi 48 = 48.

Non vale sempre il viceversa: infatti alle tuple t7 e t8: 48 = 48 ma Moore != Kemp

Le dipendenze funzionali possono essere mostrate graficamente sugli schemi relazionali tramite frecce dirette

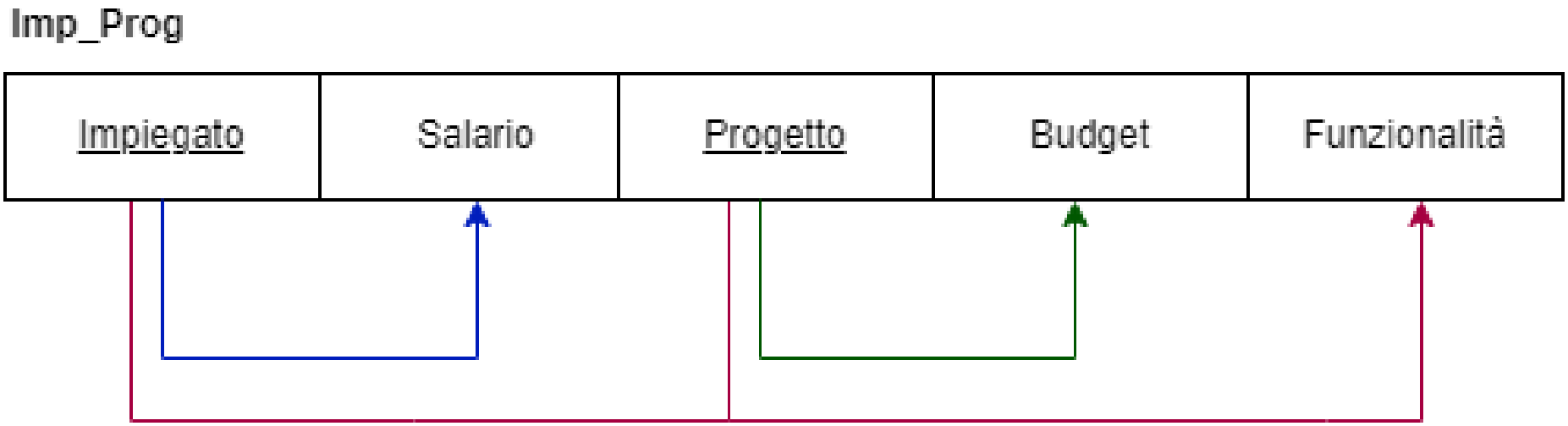
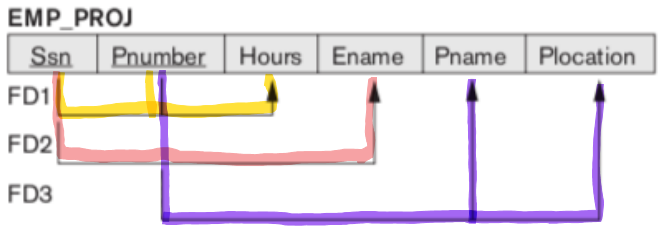


Figure 1: Esempio con FD

Altro esempio di FD:

Considerando la seguente relazione:



Possiamo derivare che le dipendenze funzionali sono:

$$
\begin{aligned}
Ssn &\rightarrow Ename \\
Pnumber &\rightarrow \{Pname, Plocation\} \\
\{Ssn, Pnumber\} &\rightarrow Hours
\end{aligned}
$$

Regole di inferenza per le dipendenze funzionali

Dato un insieme di dipendenze funzionali possiamo inferirne altre che risultano valide quando le prime lo sono.
Data una relazione $R = \{A_1, \ldots, A_n\}$ e dati X, Y e Z come insiemi non vuoti di attributi (e $X, Y, Z \subseteq R$), abbiamo i seguenti assiomi (chiamati anche regole di inferenza di Armstrong):

1. Riflessività Se $Y \subseteq X$, allora $X \to Y$.
   Es: se X = {Ssn, nome, cognome} e Y = { nome} allora X $\to$ Y
2. Aumentazione Se $X \to Y$, allora $X \cup Z \to Y \cup Z$.
   Es: se X = {ssn} e Y = {nome} allora {ssn} $\cup$ {eta} $\to$ {nome} $\cup$ {eta}
3. Transitività se $X \to Y$ e $Y \to Z$, allora $X \to Z$.
   Es: se X = {matricola} e Y = {cognome} e Z = {email} con {matricola } $\to$ {cognome} e {cognome} $\to$ {email} allora {matricola} $\to$ {email}

Queste 3 regole sono corrette e complete per le dipendenze funzionali.

Possono essere ricavate anche altre regole di inferenza:

- Unione Se $X \to Y$ e $X \to Z$, allora $X \to Y \cup Z$.
- Decomposizione Se $X \to Y \cup Z$, allora $X \to Y$ e $X \to Z$.
- Pseudotransitività Se $X \to Y$ e $W \cup Y \to Z$, allora $W \cup X \to Z$.

Queste 3 regole possono essere ricavate dall'insieme precedente grazie alla proprietà di completezza.

Per un insieme F di dipendenze funzionali la chiusura di F, indicata da F^+, è l'insieme di tutte le dipendenze funzionali che possono essere inferite da F.

Un insieme F di dipendenze funzionali copre un altro insieme G di dipendenze funzionali se $G^+ \subseteq F^+$.
Quindi F e G sono equivalenti se $F^+ \subseteq G^+$ e $G^+ \subseteq F^+$.

Normalizzazione

Normalizzazione

Assumiamo che un insieme di FD sia dato per ogni relazione e che ogni relazione abbia una chiave primaria. Il processo di normalizzazione consiste nell'analizzare gli schemi relazionali basandosi sulle FD e sulla chiave primaria per:
1) Minimizzare la ridondanza delle informazioni.
2) Minimizzare le anomalie di: inserimento, eliminazione, modifica.

Uno schema relazionale che non rispetta le condizioni della forma normale è scomposto in schemi relazionali minori che contengono un sottoinsieme degli attributi iniziali e che rispettano la forma normale.

La forma normale di una relazione si riferisce alla più alta condizione di normalità raggiunta (es: 1NF, 2NF, 3NF, ...) e indica il grado di normalizzazione.

Denormalizzazione è un processo che consiste nel memorizzare le più alte forme di normalizzazione tramite un join, ovvero otteniamo la forma di normale più bassa.

Il processo di normalizzazione attraverso la scomposizione della relazione deve rispettare le seguenti proprietà:

1. Proprietà di join non addittiva: garantisce che non vengano generate tuple spurie dopo la decomposizione.
2. Proprietà di preservazione delle dipendenze: le dipendenze funzionali vengono rispettate anche nelle relazioni scomposte.

Piccolo ripasso

Una superchiave di uno schema relazionale $R = \{A_1, \ldots, A_n\}$ è un insieme non vuoto di attributi $S \subseteq R$ tale che non possono esistere due tuple t_1 e t_2 in $r(R)$ per cui $t_1[S] = t_2[S]$.

Esempio: Se S = {matricola, eta} nella tabella studente non possiamo avere: t1[matricola, eta] = (120, 21) e t2[matricola, studente] = (120,21)

Una chiave K è una superchiave minimale (minimo numero di attributi necessario affinché ogni tupla sia distinta). Possono esserci più chiavi candidate ma ne scegliamo solo una come chiave primaria.

Un attributo di uno schema relazionale R è chiamato attributo primo di R se è membro di una qualche chiave candidata di R (altrimenti è detto non primo).

Prima forma normale 1NF

Prima forma normale 1NF:
Uno schema relazionale R è nella sua prima forma normale 1NF se:
- i domini dei suoi attributi consentono solo valori ATOMICI.
- esiste una chiave primaria.

Ovvero la prima forma normale impedisce la presenza di attributi multivalore (passando da un modello ER al modello relazionale otteniamo già la prima forma normale)

Esempio violazione 1NF

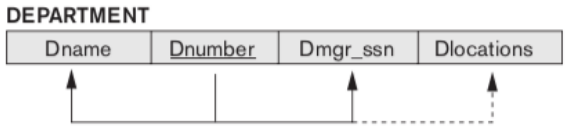


Figure 2: Schema relazionale che non è in 1NF.

Dipendenze funzionali: $Dnumber \rightarrow Dname$, $Dnumber \rightarrow Dmgr\_ssn$, $Dnumber \rightarrow Dlocations$

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

Figure 3: Un esempio delle istanze di $DEPARTMENT$.

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

Figure 4: versione in 1NF della stessa relazione (MA aggiunta di ridondanza).

Esempio II: violazione 1NF

Consideriamo la relazione Dipendente(ID, nome, cognome, telefono, competenze), con competenze attributo multivalore.

ID	Nome	Cognome	telefono	competenze
1	Mario	Rossi	333-1111111	Java, SQL
2	Laura	Bianchi	333-2222222	Python, JavaScript
3	Carlo	Verdi	333-3333333	C++, Ruby

Raggiungiamo la 1NF scomponendo in più tabelle:

ID	Nome	Cognome	telefono
1	Mario	Rossi	333-1111111
2	Laura	Bianchi	333-2222222
3	Carlo	Verdi	333-3333333

ID	Competenze
1	Java
1	SQL
2	Python
2	JavaScript
3	C++
3	Ruby

Seconda forma normale 2NF

Seconda forma normale 2NF:
Proprietà: Una FD $X \rightarrow Y$ è una dipendenza funzionale PIENA se la rimozione di un qualunque attributo A da X significa che la dipendenza non è più valida. Ovvero $X - \{A\} \not\rightarrow Y$.

Uno schema relazionale R è in seconda forma normale (2NF) se:
- è in 1NF
- ogni attributo non primo A in R dipende da una dipendenza funzionale PIENA della chiave primaria di R

(Ovvero ogni attributo non primo deve dipendere da TUTTA la chiave della relazione)

Se una relazione ha un unico attributo chiave non c'è bisogno di eseguire il test della seconda forma normale (chiaramente la dipendenza è sempre piena perchè l'attributo chiave è solo uno).

Esempio violazione della 2NF e correzione

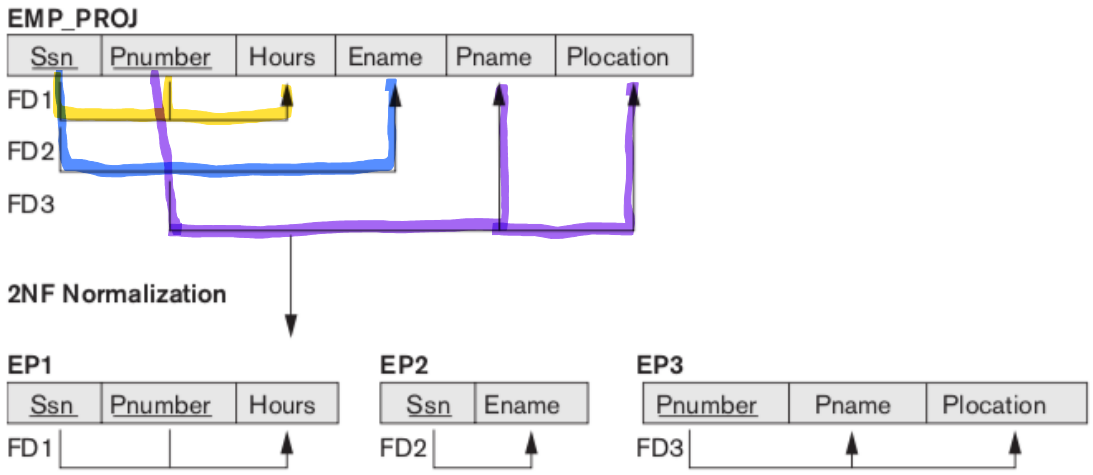


Figure 5: Normalizzazione di Emp_Prog in relazioni che rispettano 2NF: ogni attributo non primo dipende dall'intera chiave

Dipendenze funzionali:

$$\begin{aligned} Ssn &\rightarrow Ename \\ Pnumber &\rightarrow \{Pname, Plocation\} \\ \{Ssn, Pnumber\} &\rightarrow Hours \end{aligned}$$

Esempio II: violazione della 2NF e correzione

Il modello relazionale: Ordine (n_ordine, data, id_cliente, indirizzo_cliente, nome_prodotto) ha le seguenti dipendenze funzionali:
n_ordine -> data
id_cliente -> indirizzo_cliente
n_ordine, id_cliente -> nome_prodotto
Non si trova in 2NF

Trasformazione in 2NF scomponendo la relazione in più relazioni in modo che ogni attributo non primo abbia una dipendenza piena dalla chiave:
1) Ordine(n_ordine, id_cliente, nome_prodotto).
2) Info(n_ordine, data).
3) Cliente(id_cliente, indirizzo_cliente).

Terza forma normale 3NF

Terza forma normale 3NF:
Proprietà dipendenza transitiva: una FD $X \rightarrow Y$ di uno schema relazionale R è una dipendenza transitiva se esiste un insieme di attributi Z in R che non è ne una chiave candidata ne un sottoinsieme di una chiave di R e se è vale $X \rightarrow Z$ e $Z \rightarrow Y$.

Uno schema relazionale R è in 3NF se:
- è in 2NF
- nessun attributo non primo di R dipende transitivamente dalla chiave primaria

Esempio violazione 3NF e soluzione

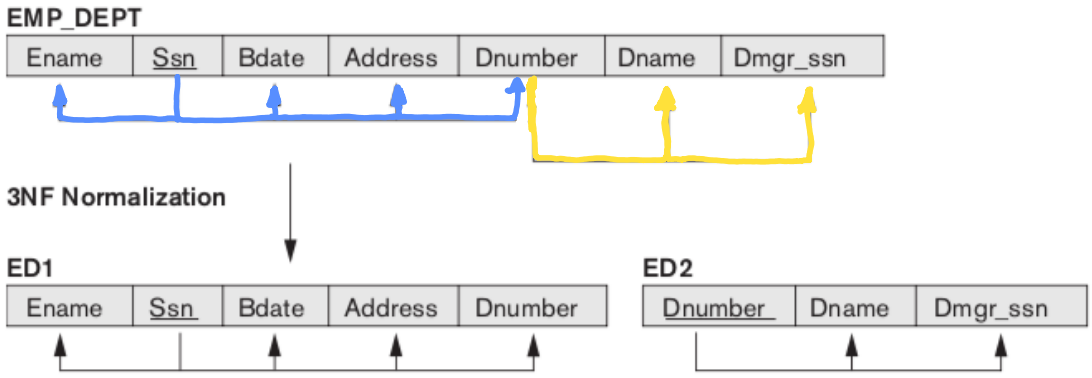


Figure 6: Normalizzazione in 3NF

Nel primo caso abbiamo che Dname e Dmrg_ssn dipendono transitivamente dalla chiave Ssn (tramite Y = DNumber che non è primo)

Esempio II: violazione 3NF e soluzione

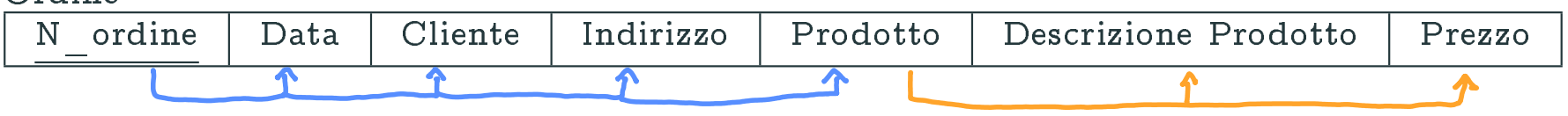
Consideriamo il seguente modello relazionale:

Ordine (n_ordine, data, cliente, indirizzo, prodotto, descrizione_prodotto, prezzo).

Le dipendenze funzionali sono le seguenti:

numero_ordine -> data, n_ordine -> cliente, n_ordine -> indirizzo, prodotto -> descrizione_prodotto, prodotto -> prezzo

Ordine



Violazione 3NF - soluzione:

Ordine

Prodotto

Riassunto forme e strategie

- 1NF
  - Test: Le relazioni non devono avere attributi multivalore/composti
  - Strategia: creare nuove relazioni per ogni attributo multivalore (o aumentare la ridondanza come in Fig.4) e scomporre gli attributi composti in attributi semplici.
- 2NF
  - Test: Per ogni relazione in cui la chiave primaria è composta da più attributi, tutti gli attributi non primi devono avere una dipendenza piena da essa.
  - Strategia: scomponi la relazione e crea una nuova relazione per ogni chiave parziale e i gli attributi dipendenti da essa. Mantieni anche una relazione con la chiave primaria iniziale e gli attributi che dipendono pienamente da essa.
- 3NF
  - Test : Nella relazione non deve esserci nessuna dipendenza transitiva di un attributo non primo rispetto alla chiave primaria.
  - Strategia: scomponi la relazione creandone di nuove con gli attributi non primi Y (che diventano chiavi) e gli attributi da loro determinati.

Definizioni generali della seconda e terza forma normale

Quindi come detto in precedenza i possibili problemi sono:

- Un sottoinsieme di chiavi di R determina un attributo non primo (non è rispettata la 2NF)
- Un attributo non primo determina un altro attributo non primo (non è rispettata la 3NF)

Una definizione alternativa per la 3NF è la seguente, uno schema relazionale R si trova in 3NF se per ogni attributo non primo valgono le seguenti condizioni:

- Ha una dipendenza funzionale piena per ogni chiave di R
- è dipendente non transitivamente da ogni chiave di R

Forma normale di Boyce-Codd

Forma normale di Boyce-Codd

Osservazione: una FD $X \to Y$ è definita triviale se $Y \subseteq X$.
Uno schema relazionale R è nella forma di Boyce-Codd se per ogni FD non triviale $X \to A$ in R, vale che X è una superchiave di R.

BCNF è più ristretta rispetto alla 3NF

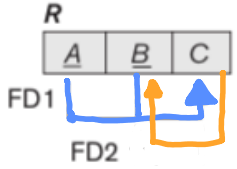


Figure 7: Relazione che è in 3NF ma non in BCNF

Le dipendenze funzionali sono:
- $A, B \to C$
- $C \to B$

La relazione non è in BCNF a causa di $C \to B$ dato che C non è una superchiave (ma è in 3NF dato che ogni attributo non primo ha una dipendenza non transitiva e piena).

Ogni forma normale contiene le precedenti:

- Ogni 2NF è in 1NF
- Ogni 3NF è in 2NF
- Ogni BCNF è in 3NF

Esistono anche altre forme normali maggiori che non guardiamo.

Alcuni esempi

Esempio di 2NF

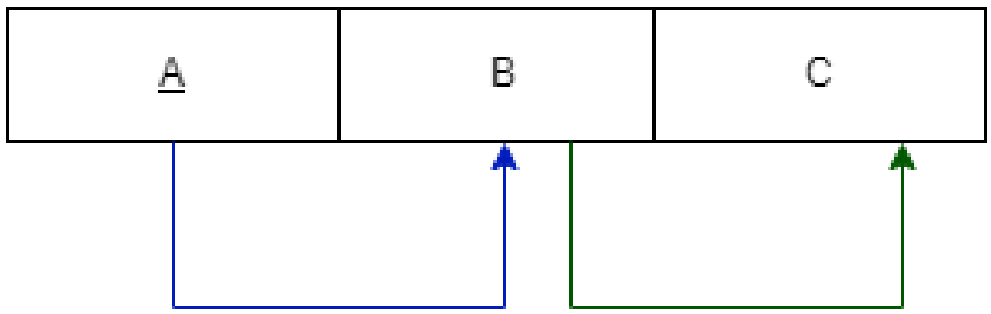


Figure 8: Relazione che è in 2NF ma non in 3NF o BCNF

- è in 2NF: perché sia B che C hanno hanno una dipendenza piena della chiave (anche se transitivamente)
- NON è in 3NF: esiste una dipendenza transitiva su un attributo non primo
- NON è in BCNF: a causa della dipendenza $B \rightarrow C$, perché B non è superchiave

Esempio di normalizzazione

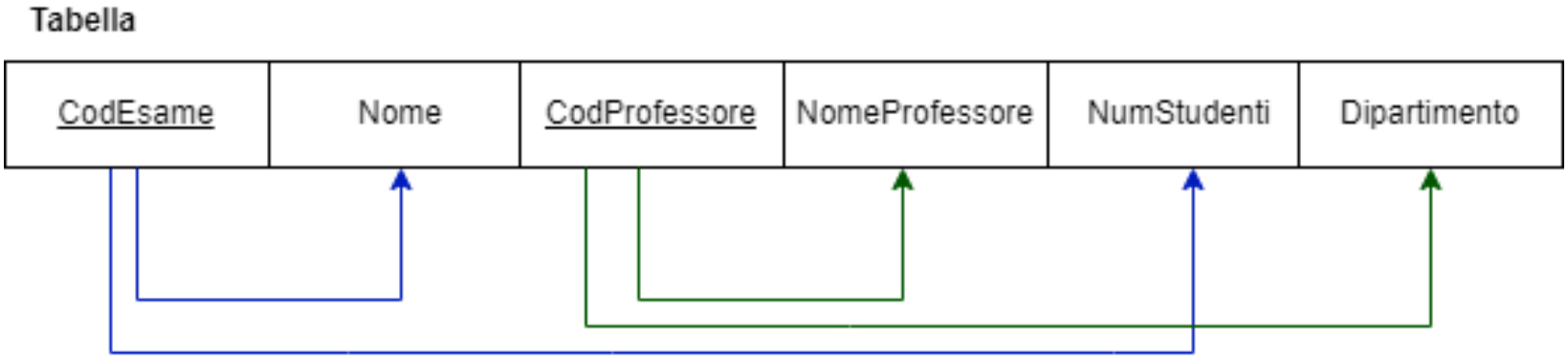


Figure 9: Relazione non normalizzata (violazione 2NF)

La relazione non è normalizzata, abbiamo degli attributi non primi che non hanno una dipendenza piena dalla chiave.
Strategia di soluzione: scomponiamo le tabelle in modo che ogni attributo non primo abbia una dipendenza piena dalle chiavi e ci assicuriamo di tenere una tabella con la chiave intera (ed eventuali attributi che dipendono da essa)

Normalizzazione in 2NF (e risulta essere anche in 3NF e BCNF)

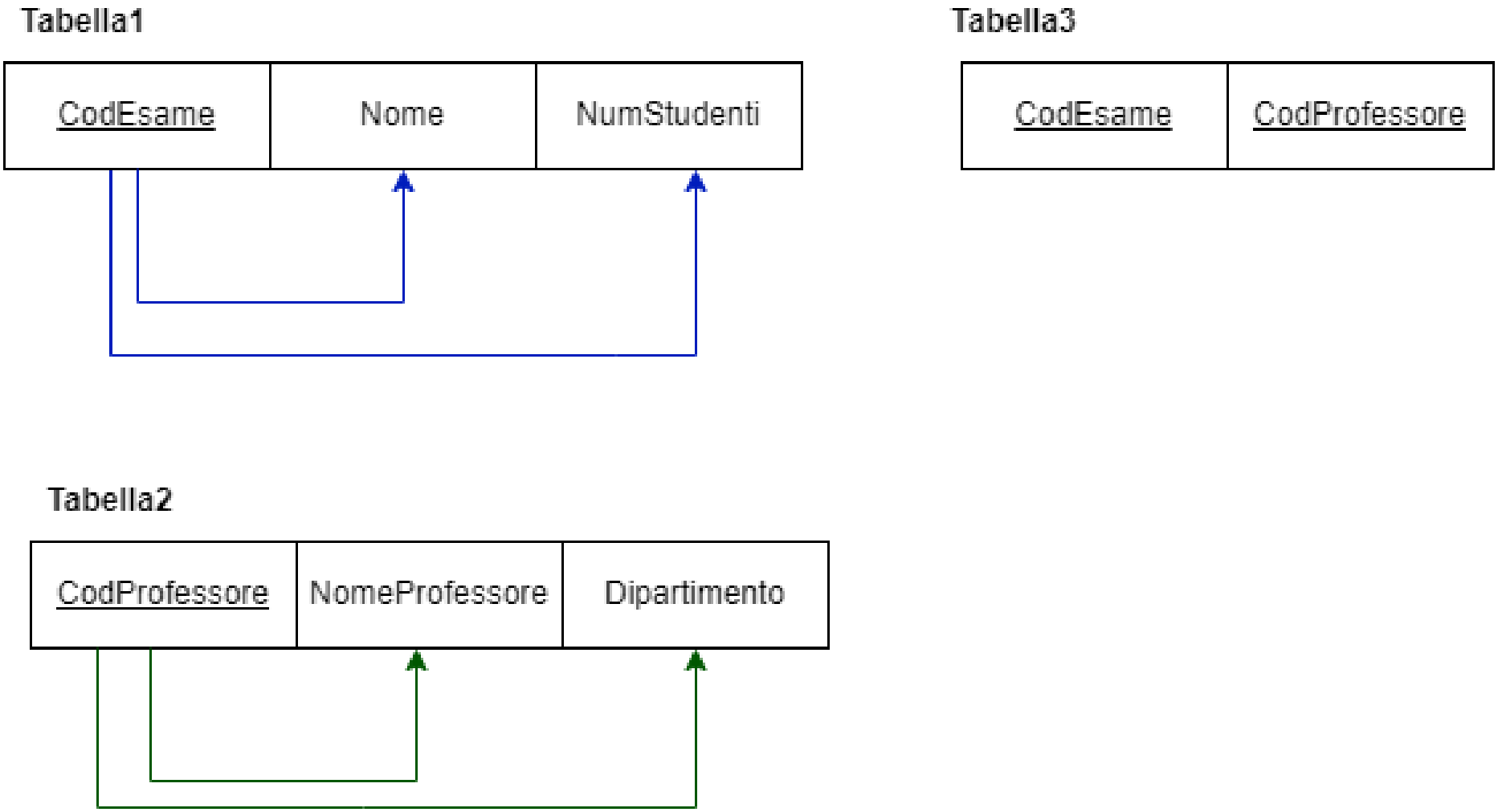


Figure 10: Relazione normalizzata

Esempio ZOO

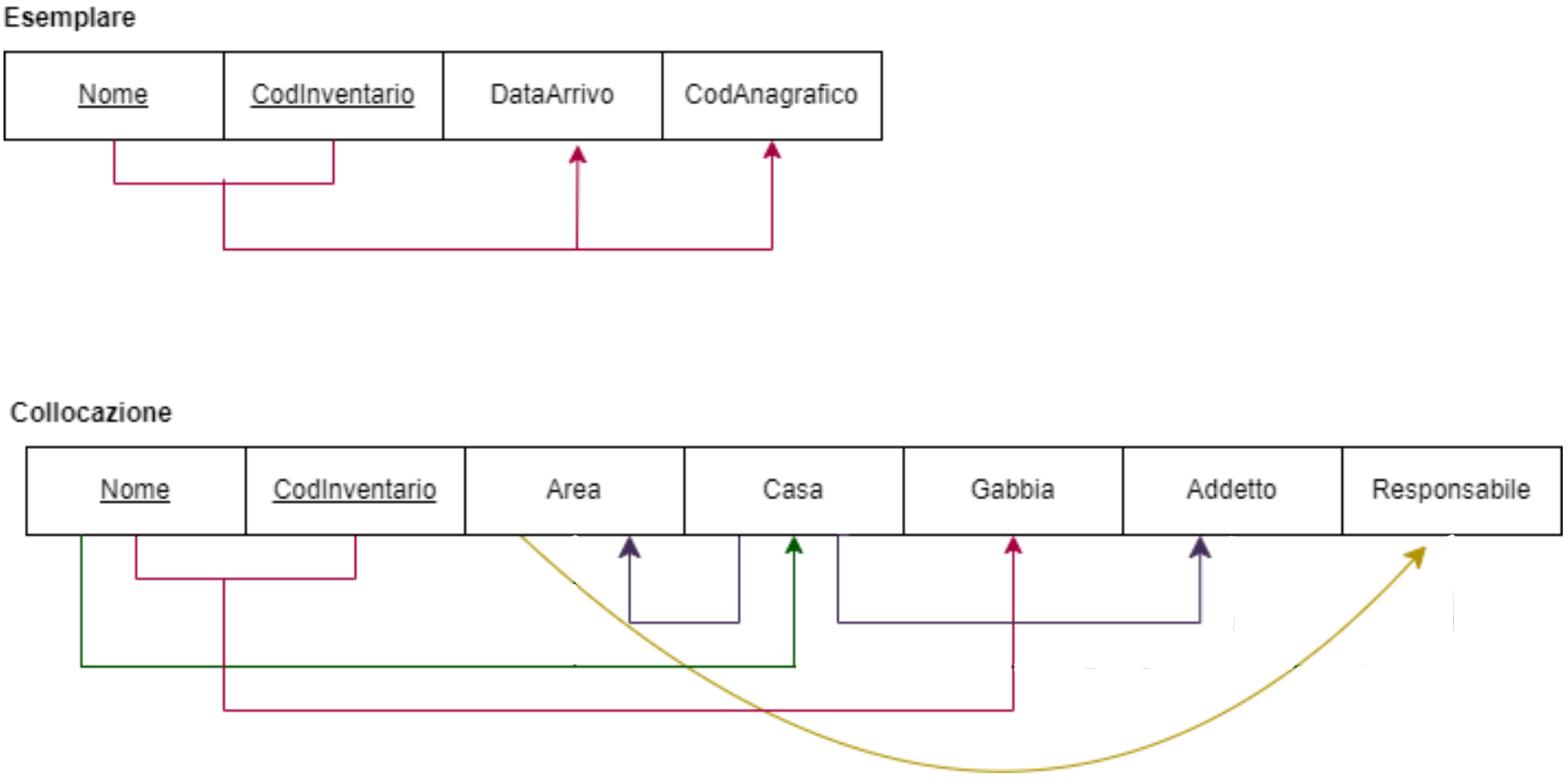
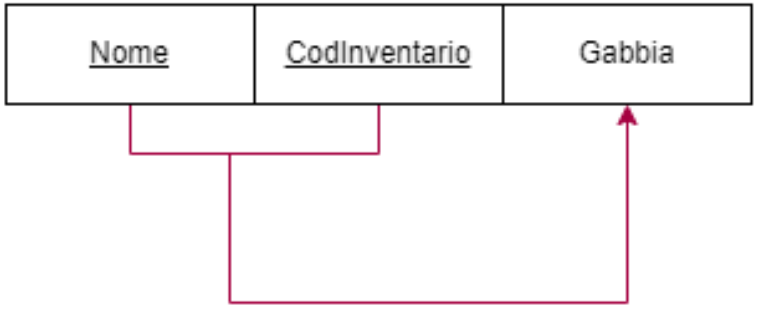


Figure 11: Relazione non normalizzata

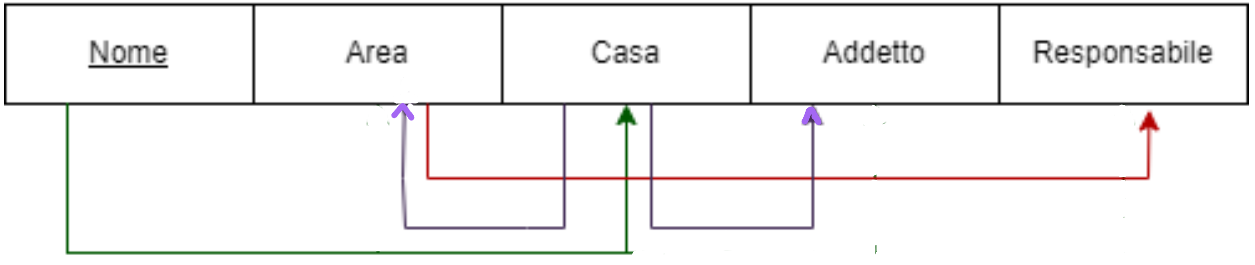
Esempio ZOO in 2NF

Esemplare

Collocazione 1



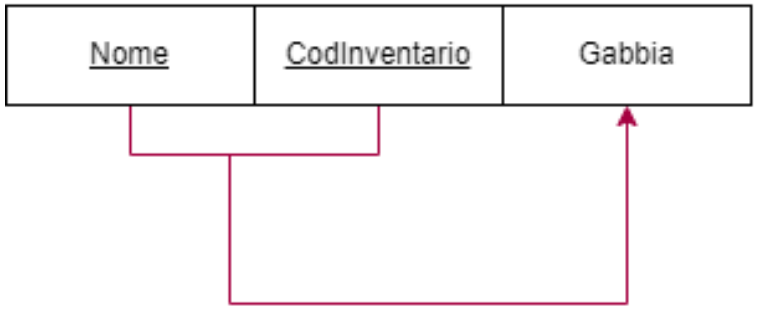
Collocazione 2



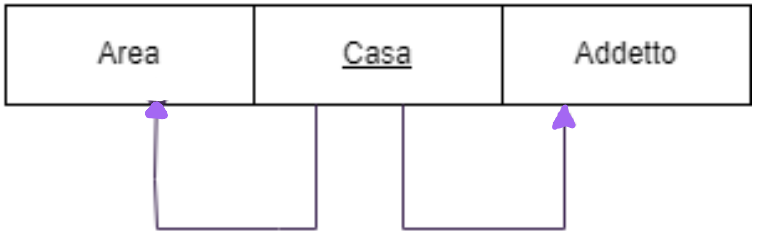
Esempio ZOO in 3NF

Esemplare

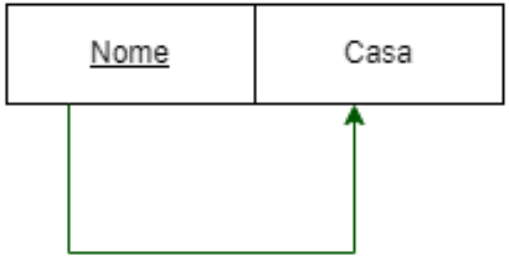
Collocazione 1



Info-Casa



Collocazione 2



Info-area