Capitolo 9 (Normalizzazione)

Normalizzazione di database relazionali

Non sempre dalla fase di progettazione si ottiene uno schema privo di difetti:

- Possono sorgere dei problemi nella fase di mapping da E-R a relazionale
- Si può ottenere uno schema non ottimale progettando direttamente uno schema logico saltando la fase di analisi concettuale

Tali difetti possono portare ad anomalie nella base di dati

Misure informali di qualità

Esistono delle misure informali di qualità per il disegno di schemi di relazione:

1. Semantica degli attributi
2. Riduzione dei valori ridondanti nelle ennuple
3. Riduzione dei valori nulli nelle ennuple
4. Non consentire ennuple spurie

Tali misure non sempre sono indipendenti tra loro

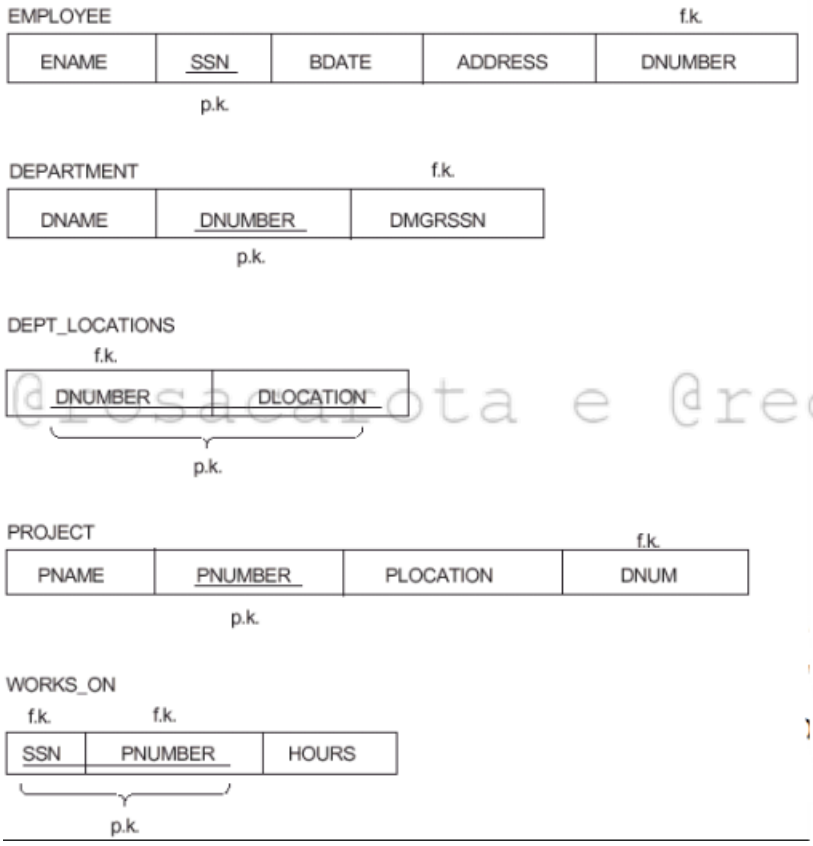
Semantica degli attributi di una relazione

Quando si raggruppano gli attributi in uno schema relazionale, assumiamo he essi abbiano associato un significato

Il significato, o **semantica**, specifica come interpretare i valori degli attributi di una relazione:

- Più è semplice spiegare la semantica ella relazione, migliore è il disegno dello schema di relazione

Esempio:



Versione semplificata dello schema del database Company

Semantica degli attributi: spiegazione significato

Significato dello schema del database:

- ENAME, SSN, BDATE, ADDRESS → dati dell'impiegato
- DNUMBER → dipartimento per cui lavora
- DNUMBER (foreign key) → relazione implicita tra employee e department
- DEPARTMENT → un'entità dipartimento
- PROJECT → un'entità progetto
- f.k DMGRSSN (di DEPARTMENT) → correla il dipartimento all'impiegato che è suo manager
- f.k. DNUM (di PROJECT) → correla un progetto al dipartimento che lo controlla
- Ogni ennupla in DEPT_LOCATIONS contiene un numero di dipartimento (DNUMBER) e una delle sedi del dipartimento (DLOCATIONS)
- Ogni ennupla in WORKS_ON contiene l'SSN (Codice Fiscale) di un impiegato, il numero di progetto PNUMBER di uno dei progetti su cui lavora ed il numero di ore settimanali HOURS
- DEPT_LOCATIONS → rappresenta un attributo multivalore di DEPARTMENT
- WORKS_ON → rappresenta una relazione di cardinalità M:N tra EMPLOYY e PROJECT

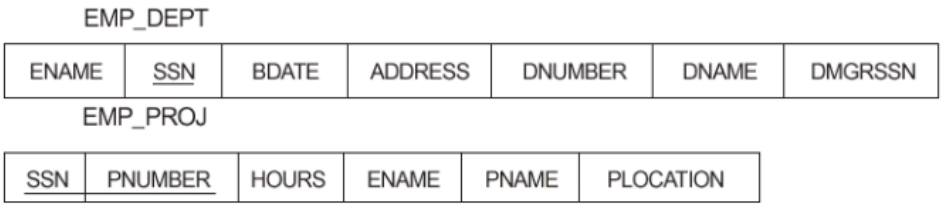
Tutte le relazioni possono quindi essere considerate "buone" **se hanno una semantica chiara**

Linea guida 1

Disegnare uno schema di relazione del quale sia facile spiegarne il significato

- Non combinare attributi da entità e relazioni differenti in una singola tabella

Esempio, consideriamo le due tabelle:



- EMPT_DEPT → ogni entità rappresenta un impiegato, ma include informazioni sul dipartimento in cui lavora
- EMP_PROJ → ogni ennupla correla un impiegato ad un progetto, ma richiede anche il nome dell'impiegato ENAME ed il nome e la locazione del progetto, PNAME e PLOCATION

Anche questi schemi di relazione hanno una semantica chiara, però entrambi contravvengono la linea guida 1, contenendo attributi di entità distinte

Riduzione dei valori ridondanti nelle ennuple

Un obiettivo nel disegno dello schema è quello di minimizzare lo spazio di memoria occupato dalle relazioni base

Oltre all'occupazione di più spazio, le tabelle EMP_DEPT e EMP_PROJ presentano anche il problema delle "anomalie di

aggiornamento” (**update anomalies**)

Anomalie di aggiornamento

Le anomalie di aggiornamento possono sorgere durante la gestione di un database relazionale progettato in modo non corretto

Si dividono in:

- Anomalie di Inserimento (**Insertion Anomalies**)
- Anomalie di Cancellazione (**Deletion Anomalies**)
- Anomalie di Modifica (**Modification Anomalies**)

Anomalie di Inserimento

Possono essere di due tipi

Ad es., riferendosi allo schema EMP_DEPT:

EMP_DEPT

ENAME	SSN	BDATE	ADDRESS	DNUMBER	DNAME	DMGRSSN

1. Per inserire una nuova ennupla di IMPIEGATO, dobbiamo richiedere anche i valori degli attributi del dipartimento per cui lavora

   Inoltre tali valori devono essere consistenti

   - Esempio: per inserire un nuovo impiegato ed assegnarlo al dipartimento 5, dobbiamo inserire i valori degli attributi DNAME e DMGRSSN consistenti con gli stessi valori di altre ennuple in EMP_DEPT per il dipartimento 5

2. È difficile inserire un nuovo dipartimento nella relazione EMP_DEPT, se questo non ha ancora impiegati. L’unico modo è di inserirlo ponendo a null gli attributi per l’impiegato, ma ciò crea un problema poiché SSN è la chiave primaria. Inoltre, quando si inserisce la ennupla per il primo impiegato del dipartimento, non serve più la ennupla con i valori null

Anomalie di Cancellazione

Questo problema è correlato al problema delle anomalie di inserimento:

- Se cancelliamo da EMP_DEPT la ennupla relativa all’ultimo impiegato che lavora per un particolate dipartimento, l’informazione sul dipartimento viene persa nel database

Anomalie di Modifica

Se cambiano valore di uno degli attributi di un particolare dipartimento, dobbiamo aggiornare le ennuple di tutti gli impiegati che lavorano per quel dipartimento. Altrimenti il database diventa inconsistente

Linea guida 2

Disegnare gli schemi di relazione di base in modo che non possano verificarsi anomalie di inserimento, cancellazione o modifica

Talvolta le linee guida possono essere violate per scopi di efficienza. Una soluzione migliore potrebbe essere quella di **definire delle viste**

Pertanto, se si decide di mantenere un’anomalia, bisogna fare in modo che i programmi che aggiornano il database operino correttamente

Riduzione dei valori NULL

Se nel disegno di uno schema di database raggruppiamo molti attributi in una relazione, può capitare che alcuni degli attributi non riguardano tutte le ennuple

Pertanto, possiamo avere molti valori NULL

Oltre allo spreco di spazio, ciò crea problemi con le funzioni di aggregazione COUNT e SUM

Interpretazioni di NULL

Il valore NULL può avere diverse interpretazioni:

- L’attributo non si applica alla ennupla
- Il valore dell’attributo per la ennupla non è noto
- Il valore dell’attributo è noto, ma assente, cioè non è stato ancora registrato

Linea guida 3

Evitare, per quanto possibile, di porre in una tabella attributi i cui valori possono essere null

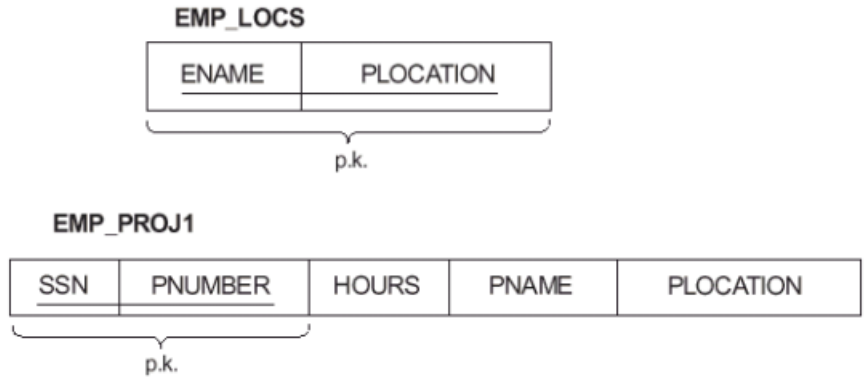
Se i valori null sono inevitabili, assicurarsi che essi ricorrano in casi eccezionali e non per la maggioranza delle ennuple

Esempio:

- Se solo il 10% degli impiegati ha un ufficio individuale, non ha senso includere l'attributo OFFICE_NUMBER nella tabella EMPLOYEE; piuttosto è più corretto creare la tabella: EMP_OFFICES(ESSN, OFFICE_NUMBER)

Tuple spurie

Consideriamo i due schemi, alternativi a EMP_PROJ:



Le ennuple rappresentano:

- EMP_LOCS: l'impiegato di nome ENAME lavora per qualche progetto la cui sede è PLOCATION
- EMP_PROJ1: l'impiegato con codice fiscale SSN lavora HOURS ore alla settimana sul progetto avente numero PNUMBER, nome PNAME e sede PLOCATION

La suddivisione appena vista non corrisponde ad un buon disegno di database in quanto, volendo riottenere le informazioni di

EMP_PROJ mediante un natural join sui due schemi, si ottengono **tuple spurie**, rappresentanti informazioni errate

I valori presenti nelle due tabelle:

EMP_LOCS

ENAME	PLOCATION
Smith, John B.	Bellaire
Smith, John B.	Sugarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Sugarland
Wong, Franklin T.	Sugarland
Wong, Franklin T.	Houston
Wong, Franklin T.	Stafford

EMP_PROJ1

SSN	PNUMBER	HOURS	PNAME	PLOCATION
123456789	1	32.5	Product X	Bellaire
123456789	2	7.5	Product Y	Sugarland
666884444	3	40.0	Product Z	Houston
453453453	1	20.0	Product X	Bellaire
453453453	2	20.0	Product Y	Sugarland
333445555	2	10.0	Product Y	Sugarland
333445555	3	10.0	Product Z	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston

Risultato del natural join: le tuple spurie sono contrassegnate da un ‘*’

SSN	PNUMBER	HOURS	PNAME	PLOCATION	
123456789	1	32.5	ProductX	Bellaire	Smith,John B.
* 123456789	1	32.5	ProductX	Bellaire	English,Joyce A.
123456789	2	7.5	ProductY	Sugarland	Smith,John B.
* 123456789	2	7.5	ProductY	Sugarland	English,Joyce A.
* 123456789	2	7.5	ProductY	Sugarland	Wong,Franklin T.
666884444	3	40.0	ProductZ	Houston	Narayan,Ramesh K.
* 666884444	3	40.0	ProductZ	Houston	Wong,Franklin T.
* 453453453	1	20.0	ProductX	Bellaire	Smith,John B.
453453453	1	20.0	ProductX	Bellaire	English,Joyce A.
* 453453453	2	20.0	ProductY	Sugarland	Smith,John B.
453453453	2	20.0	ProductY	Sugarland	English,Joyce A.
* 453453453	2	20.0	ProductY	Sugarland	Wong,Franklin T.
* 333445555	2	10.0	ProductY	Sugarland	Smith,John B.
* 333445555	2	10.0	ProductY	Sugarland	English,Joyce A.
333445555	2	10.0	ProductY	Sugarland	Wong,Franklin T.
* 333445555	3	10.0	ProductZ	Houston	Narayan,Ramesh K.
333445555	3	10.0	ProductZ	Houston	Wong,Franklin T.
333445555	10	10.0	Computerization	Stafford	Wong,Franklin T.
* 333445555	20	10.0	Reorganization	Houston	Narayan,Ramesh K.
333445555	20	10.0	Reorganization	Houston	Wong,Franklin T.

Tale problema sorge perché PLOCATION è l'attributo che correla EMP_LOCS e EMP_PROJ1, ma non è né chiave primaria né chiave esterna in nessuna delle due relazioni

Linea guida 4

Progettare gli schemi di relazione in modo da poter effettuare JOIN con condizioni di uguaglianza su attributi che sono o chiave primaria o chiave esterna, in modo da non generare tuple spurie

Sommario delle linee guida

Abbiamo mostrato in modo informale che una cattiva progettazione dello schema di un database può portare ad una serie di problemi:

- Anomalie che implicano un maggiore sforzo nell'inserimento e nella modifica di una tabella, e che possono portare a perdita accidentali di dati durante la cancellazione
- Spreco di spazio a causa dei valori null
- Generazione di tuple spurie o non valide durante operazioni di join

Dipendenze funzionali

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

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

Supponiamo che lo schema di database relazionale abbia n attributi $A_1, A_2, ..., A_n$ e che l'interno database sia descritto da uno schema di relazione universale $R = \{A_1, A_2, ..., A_n\}$

Una dipendenza funzionale, denotata da $X \rightarrow Y$, tra due insiemi di attributi X e Y che sono sottoinsiemi di R, specifica un vincolo sulle possibili ennuple che possono formare una istanza di relazione r di R

Il vincolo stabilisce che se $X \rightarrow Y$, allora $\forall\, t_1, t_2$ in r tali che $t_1[X] = t_2[X]$ deve valere $t_1[Y] = t_2[Y]$

- Ciò significa che i valori della componente Y di una tupla di r **dipendono da** (o **sono determinati da**) i valori della componente X

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:

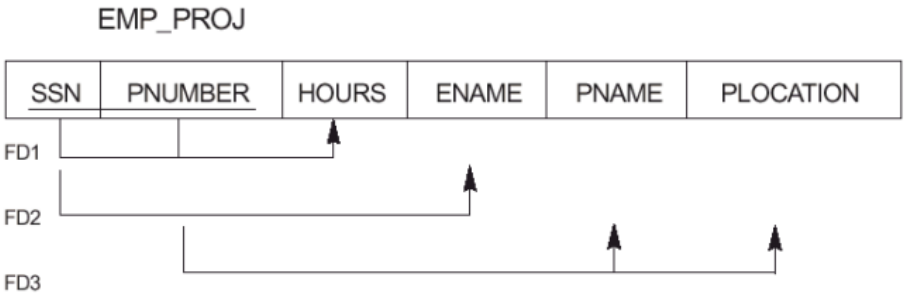
- Y è funzionalmente dipendente da X
- X è la parte sinistra della FD
- Y è la parte destra della FD

Si noti che:

- Un'osservazione riguarda il vincolo di chiave:
  - Se prendiamo una chiave K di una relazione r, si può facilmente verificare che esiste una dipendenza funzionale tra K e ogni altro attributo dello schema di r
  - Questo perché, per definizione stessa di vincolo di chiave, non possono esistere due tuple con gli stessi valori su K e quindi una dipendenza funzionale che ha K al primo membro sarà sempre soddisfatta
  - *Esisterà sempre una dipendenza funzionale tra una chiave di una relazione e tutti gli attributi dello schema della relazione (esclusi quelli della chiave stessa)*
- Il fatto che $X \to Y$ in R non implica nulla circa $Y \to X$ in R

Esempio:

- La dipendenza funzionale è una proprietà della semantica degli attributi



- FD1: $\{SSN, PNUMBER\} \rightarrow HOURS$
- FD2: $SSN \rightarrow ENAME$
- FD3: $PNUMBER \rightarrow \{PNAME, PLOCATION\}$

Dipendenza funzionale dalla semantica degli attributi

Una dipendenza funzionale è una proprietà dello schema di relazione (intensione) R e non di un particolare stato di relazione (estensione legale r di R

- Le **estensioni legali** o stati di relazione legali sono delle estensioni di relazione $r(R)$ che soddisfano i vincoli di FD

Una FD non può essere inferita in modo automatico da un'istanza di relazione r, ma deve essere definita esplicitamente da chi conosce il significato degli attributi

Esempio:

TEACH

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

In teoria si potrebbe pensare che Teacher determina funzionalmente Course e che Course determina funzionalmente Text

Ci sono però delle tuple che non seguono queste FD, e quindi giungiamo alla conclusione che:

- $TEACHER \nrightarrow COURSE$
- $COURSE \nrightarrow TEXT$

Regole di inferenza per FD

Sia F l'insieme delle dipendenze funzionali di uno schema di relazione R. Il disegnatore dello schema specifica le dipendenze funzionali **semanticamente ovvie**

In tutte le istanze di relazione legali, però, valgono numerose altre dipendenze funzionali: queste possono essere **inferite** (o dedotte) da F

L'insieme di tali dipendenze è detto **chiusura di $\mathbf{F}$** ed è denotato da $\mathbf{F}^+$

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

  $\}$

- Possiamo inferire:
  - $SSN \rightarrow \{DNAME, DMGRSSN\}$
  - $SSN \rightarrow SSN$
  - $DNUMBER \rightarrow DNAME$
  - …

Notazioni:

- $F \models X \rightarrow Y$ denota che la FD $X \rightarrow Y$ è inferita da F
- $\{X, Y\} \rightarrow Z$ è abbreviato in $XY \rightarrow Z$
- $\{X, Y, Z\} \rightarrow \{U, V\}$ è abbreviato in $XYZ \rightarrow UV$

Regole di inferenza fondamentali

Regole di inferenza per le dipendenze funzionali:

- (IR1) - Regola riflessiva

    - Se $Y \subseteq X$ allora $X \to Y$
- (IR2) - Regola incrementale
    - $\{X \to Y\} \vDash XZ \to YZ$
- (IR3) - Regola transitiva
    - $\{X \to Y, Y \to Z\} \vDash X \to Z$
- (IR4) - Regola di proiezione o decomposizione (notare $Z \subseteq YZ$, e $Y \subseteq YZ$)
    - $\{X \to YZ\} \vDash X \to Z$
    - $\{X \to YZ\} \vDash X \to Y$
- (IR5) - Regola di unione (o additiva)
    - $\{X \to Y, X \to Z\} \vDash X \to YZ$
- (IR6) - Regola pseudo-transitiva
    - $\{X \to Y, WY \to Z\} \vDash WX \to Z$

approf cap algoritmi.pdf

Regole di inferenza di Armstrong

È stato provato che (IR1) - (IR3) (le prime tre regole) sono regole di inferenza **corrette** e **complete** (Armstrong 1974)

- Corretto: dato un insieme F di dipendenze funzionali su R, ogni dipendenza inferita da F usando (IR1) - (IR3) è vera in ogni stato che soddisfa le dipendenze in F
- Completo: la chiusura di F (l'insieme di tutte le possibili dipendenze inferibili da F) può essere determinata usando

solo (IR1) - (IR3) **(regole di inferenza di Armstrong)**

Regole di inferenza nella progettazione di database

Tipicamente, il progettista del database 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 che appare come parte sinistra di qualche dipendenza funzionale in F
2. Usare le regole di inferenza per determinare l'insieme di tutti gli attributi dipendenti da X

Normalizzazione dei dati

La **normalizzazione dei dati** può essere vista come un processo che consente di decomporre schemi di relazione non ottimali in schemi più piccoli, tali da garantire la mancanza di anomalie di aggiornamento **(update anomalies)**

Forme normali

Le forme normali forniscono:

- Un metodo formale per analizzare schemi di relazione, basato sulle chiavi e sulle dipendenze funzionali tra gli attributi
- Una serie di test da condurre sugli schemi di relazione individuali

- Se un test fallisce, la relazione che viola il test deve essere decomposta in relazioni che individualmente superano il test

Esistono varie forme normali

Le più importanti sono la prima, la seconda e la terza forma normale (1NF, 2NF, 3NF)

Esiste poi la forma normale di Boyce-Codd (BCNF), definita sulla base di 3NF

1NF, 2NF, 3NF e BCNF sono tutte definite considerando solo vincoli di dipendenza funzionale e di chiave

Richiami

Ricordiamo che una **superchiave** di uno schema di relazione $R = \{A_1, A_2, ..., A_n\}$ è un insieme di attributi $S \subseteq R$ tali che non esistono due tuple t_1 e t_2, in qualche stato di r di R per cui $t_1[S] = t_2[S]$

Inoltre, una **chiave** k è una superchiave con la proprietà di minimalità. Cioè la rimozione da k di un qualche attributo fa perdere a k la proprietà di superchiave

Tutte le chiavi "minimali" sono dette **chiavi candidate**. Una tra esse viene scelta arbitrariamente ed è detta **chiave primaria**

Un attributo di uno schema di relazione R che ricorre in qualche chiave candidata di R è detto attributo **primo** di R

Esempio:

EMPLOYEE

ENAME	SSN	BDATE	ADDRESS	DNUMBER

- $\{SSN\}$ è una chiave
- $\{SSN\}, \{SSN, ENAME\}, \{SSN, ENAME, BDATE\}$ etc. sono **superchiavi**
- $\{SSN\}$ è l'unica **chiave candidata**
- $\{SSN\}$ è la **chiave primaria**

Prima forma normale (1NF)

> 💡 La 1NF è stata definita per non consentire attributi multivalore, composti e le loro combinazioni
>
> Gli unici valori consentiti da 1NF sono valori **atomici** (o **indivisibili**)

Esempio:

DEPARTMENT

DNAME	DNUMBER	DMGRSSN	DLOCATIONS
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

L’idea è di rimuovere DLOCATIONS che viola la 1NF e porlo in una relazione separate insieme con la chiave primaria

La 1NF proibisce anche attributi composti **(relazioni annidate)**

EMP_PROJ

SSN	ENAME	PROJS	
		PNUMBER	HOURS

EMP_PROJ

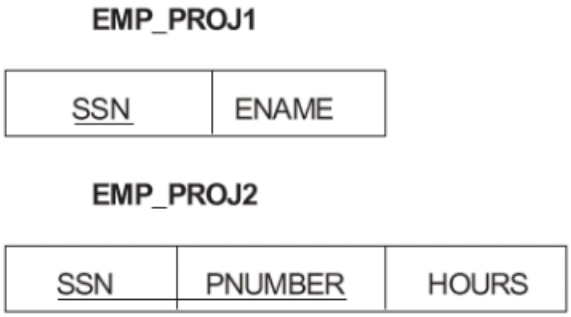
SSN	ENAME	PNUMBER	HOURS
123456789	Smith,John B.	1	32.5
		2	7.5
666884444	Narayan,Ramesh K.	3	40.0
453453453	English,Joyce A.	1	20.0
		2	20.0
333445555	Wong,Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0

SSN è chiave primaria

PNUMBER è chiave primaria parziale della relazione annidata

Per normalizzare:

- Spostare gli attributi della relazione annidata in una nuova relazione e propagare la chiave primaria in essa



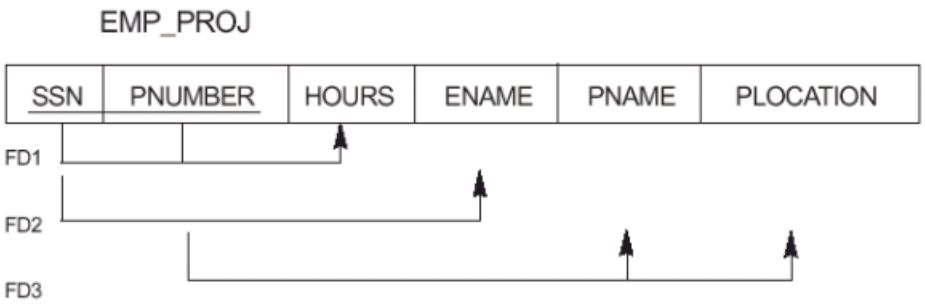
Seconda forma normale (2NF)

È basata sul concetto di **dipendenza funzionale piena**

Definizione:

- Una **dipendenza funzionale** $X \to Y$ è **piena** se la rimozione di qualche attributo A da X implica che la dipendenza non vale più, cioè:
  - $\forall A \in X, (X - \{A\}) \not\to Y$
- Una dipendenza funzionale $X \to Y$ è **parziale** se qualche attributo $A \in X$ può essere rimosso e la dipendenza vale ancora, cioè:
  - $\exists A \in X, (X - \{A\}) \to Y$

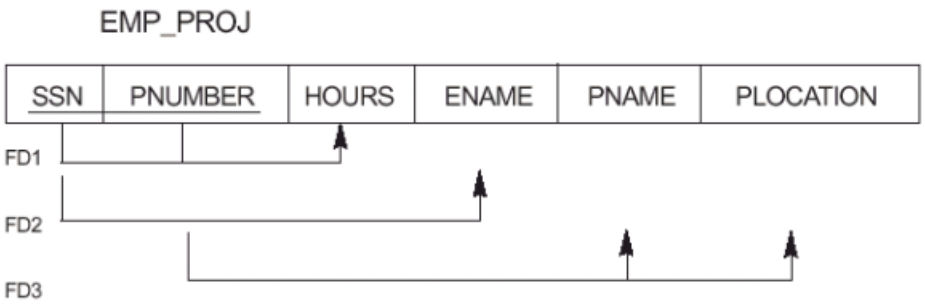
Esempio dipendenze piene:



- $\{SSN, PNUMBER\} \rightarrow HOURS$: **piena**, infatti:
  - $SSN \nrightarrow HOURS$
  - $PNUMBER \nrightarrow HOURS$
- $\{SSN, PNUMBER\} \rightarrow ENAME$: **parziale**, infatti:
  - $SSN \rightarrow ENAME$

> 💡 Uno schema di relazione R è in 2NF se ogni attributo non primo A in R ha una **dipendenza funzionale piena** dalla chiave primaria di R

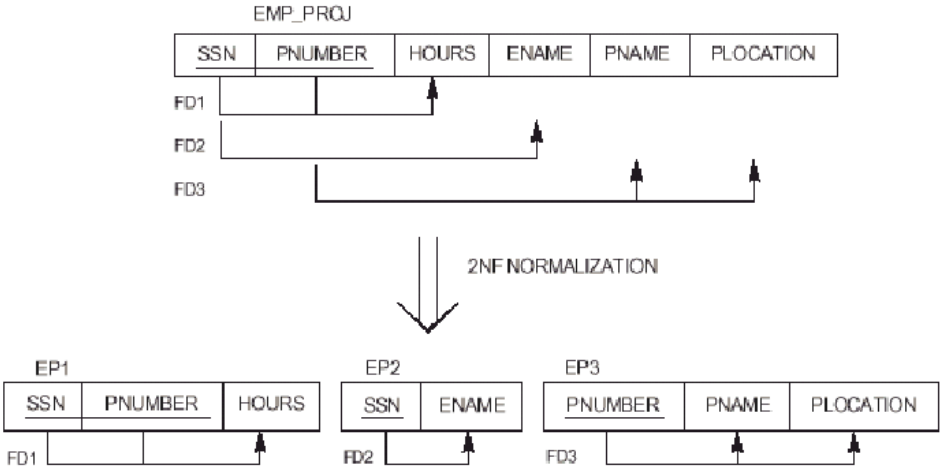
Esempio:



- EMP_PROJ è in 1NF ma non in 2NF. Infatti gli attributi non primi PNAME e PLOCATION violano 2NF in virtù di FD3. ENAME viola 2NF in virtù di FD2
- Le dipendenze funzionali FD2 e FD3 rendono ENAME, PNAME, PLOCATION parzialmente dipendenti dalla chiave primaria $\{SSN, PNUMBER\}$

Se uno schema di relazione non è in 2NF, può essere ulteriormente normalizzato in un certo numero di tabelle in 2NF, in cui gli attributi non primi sono associati solo con la parte di chiave primaria su cui sono funzionalmente dipendenti in modo pieno

Normalizzazione in 2NF della relazione EMP_PROJ:

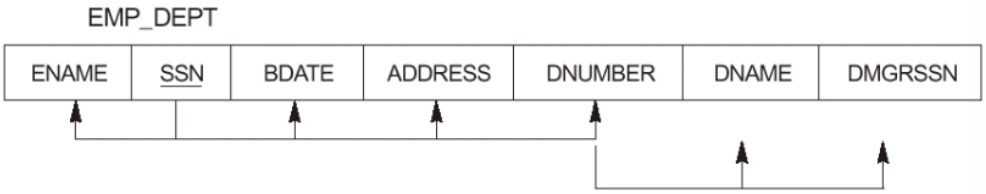


Terza forma normale (3NF)

È basata sul concetto di **dipendenza transitiva**

Una dipendenza funzionale $X \rightarrow Y$ in uno schema R è una **dipendenza transitiva** se esiste un insieme di attributi Z che non è sottoinsieme di alcuna chiave di R e valgono $X \rightarrow Z$ e $Z \rightarrow Y$

Esempio:



- $SSN \rightarrow DMGRSSN$ è transitiva attraverso $DNUMBER$, infatti:
  - $SSN \rightarrow DNUMBER$
  - $DNUMBER \rightarrow DMGRSSN$
  - $DNUMBER$ non è sottoinsieme di alcuna chiave di EMP_DEPT

💡 Secondo la definizione di Codd: uno schema di relazione R è in 3NF se è in 2NF e nessun attributo non-primo di R è transitivamente dipendente dalla chiave primaria

Definizione generale di 2NF e 3NF

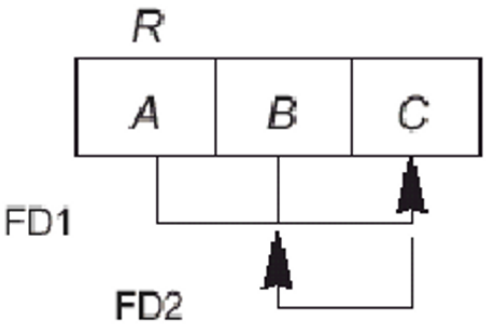
La definizione generale di 2NF e 3NF non riguarda solo la chiave primaria ma tutte le chiavi candidate

💡 Uno schema di relazione R è in 2NF se ogni attributo non-primo A in R non è parzialmente dipendente da alcuna chiave di R, né primaria né candidata

💡 Uno schema di relazione R è in 3NF se, ogni volta che vale in R una dipendenza funzionale $X \rightarrow A$ o X è una super-chiave di R, oppure A è un attributo primo di R

Forma normale di Boycee-Codd (BCNF)

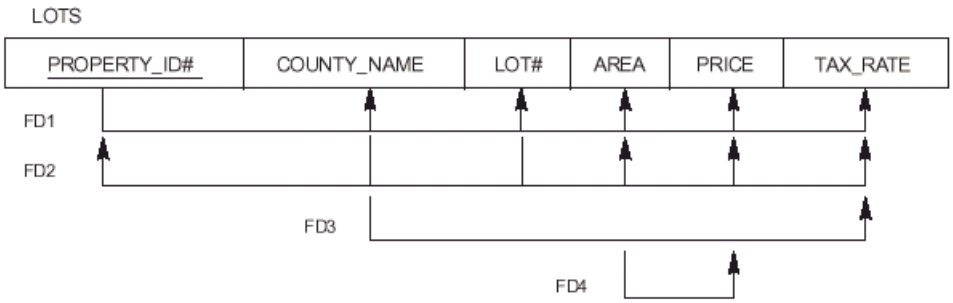
💡 Uno schema R è in BCNF se ogniqualvolta vale in R una dipendenza funzionale $X \to A$, allora X è una super-chiave di R



Se $\{A, B\}$ è la chiave e C è non primo, R è in 3NF, ma non in BCNF ($C \to B$ ma C non è super-chiave di R)

Forme normali: esempi

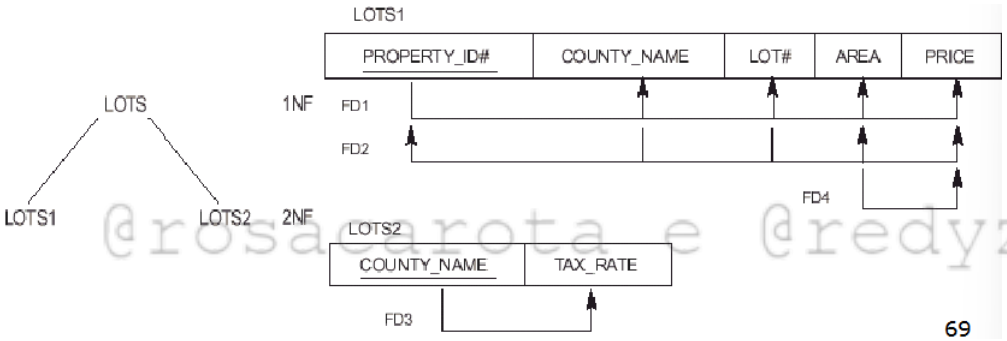
Lo schema di relazione LOTS descrive lotti di terreno in vendita in varie contee di uno stato:



- Chiavi candidate: $PROPERTY\_ID\#$ e $\{COUNTRY\_NAME, LOT\#\}$
- Chiave primaria: $PROPERTY\_ID\#$
- Inoltre valgono:
  - FD3 $COUNTRY\_NAME \rightarrow TAX\_RATE$
  - FD4 $AREA \rightarrow PRICE$

LOTS viola la definizione generale di 2NF perché TAX_RATE è parzialmente dipendente dalla chiave candidata $\{COUNTRY\_NAME, LOT\#\}$ in virtù di FD3

Normalizziamo decomponendo LOTS in LOTS1 e LOTS2:

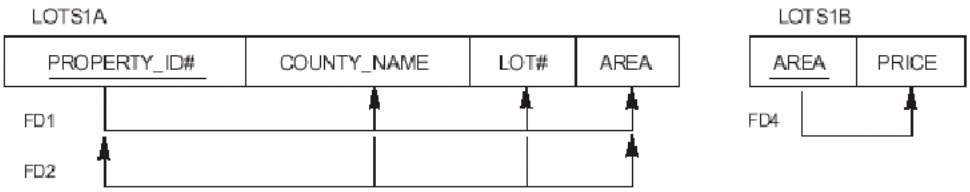


Secondo la definizione generale di 3NF, LOTS2 è in 3NF, mentre FD4 in LOTS1 viola la 3NF

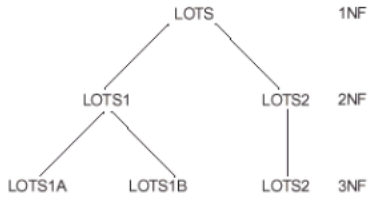
Infatti $AREA \rightarrow PRICE$, ma:

- $AREA$ non è super-chiave di LOTS1
- $PRICE$ non è attributo primo

Per normalizzare decomponiamo LOTS1 rimuovendo $PRICE$ e ponendolo insieme ad $AREA$ in un altro schema di relazione



Decomposizione di Lots1 in 3NF



Sommario delle decomposizioni

Supponiamo che ci siano migliaia di lotti, appartenenti a due sole contee:

- Mario Country con aree: 0,5 - 0,6 - 0,7 - 0,8 - 0,9 - 1,0
- Libery Country con aree: 1,1 - 1,2 - … - 1,9 - 2,0