

Normalizzazione

Motivazione

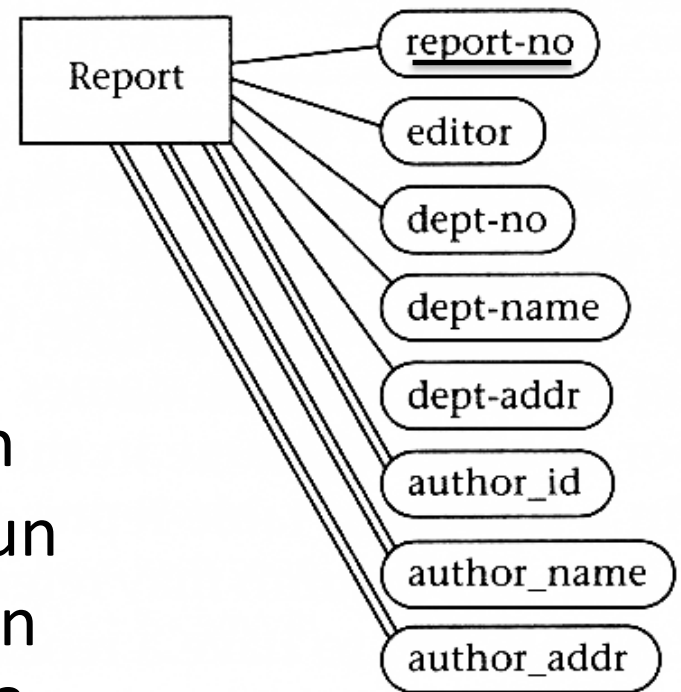
- Lo schema logico di una base dati, rappresentato attraverso il modello relazionale può presentare talvolta delle anomalie o criticità in termini di prestazioni, integrità e manutenzione.
- Come caso limite si può considerare quello di una base dati rappresentata attraverso un'unica relazione (tabella):
 - Tale soluzione è frequente tra coloro che considerano un database qualcosa di equivalente ad un foglio EXCEL

Motivazione

- Tipicamente, questa rappresentazione è associata ad alcune **anomalie** di funzionamento:
 - Elevata ridondanza dei dati, con conseguente **inefficienza** dei tempi di ricerca
 - **Complessità** delle operazioni di aggiornamento
 - Pericolo di **perdita** completa di informazioni in caso di cancellazioni di alcune righe della tabella
- Vediamo un esempio ...

Motivazione

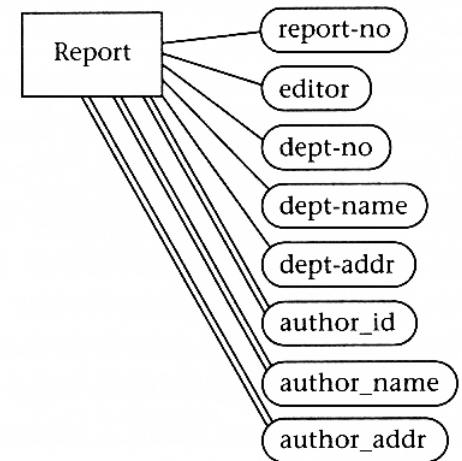
- Si consideri un contesto nel quale vogliamo rappresentare i report prodotti all'interno di alcuni dipartimenti.
- In un dipartimento esiste una persona che svolge le funzioni di editor e cura la produzione dei report del dipartimento
- Ogni report è identificato da un codice univoco ed è scritto da un numero di autori variabile da un minimo di 1 ad un massimo di 3



Motivazione

- Un'errata traduzione prevederebbe la seguente tabella:

REPORT (report-no, editor,
dept-no, dept-name, dept-addr,
author_id1, author_name1, author_addr1,
author_id2, author_name2, author_addr2,
author_id3, author_name3, author_addr3)



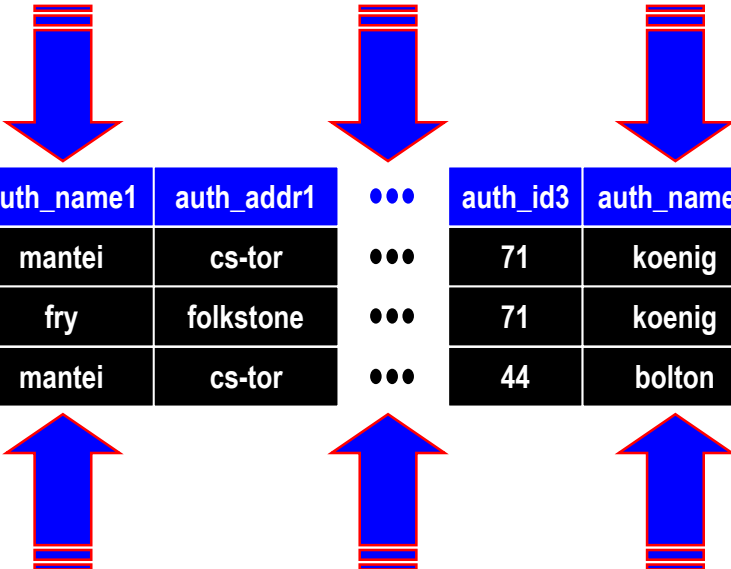
<u>report_no</u>	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev

Questa soluzione è caratterizzata da
alcuni aspetti tutt'altro che vantaggiosi

Motivazione

- La ricerca dei report scritti da un autore (per esempio “mantei”) richiede l’intera scansione del contenuto di tre colonne della tabella

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev



Motivazione

- I dati di alcuni autori e dipartimenti sono memorizzati in forma ridondante.

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev

Motivazione

- L'aggiornamento dell'indirizzo di un autore può richiedere la modifica di diverse righe della tabella (**anomalia di aggiornamento**)

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev

Motivazione

- Assunto che si crei un nuovo dipartimento con relativo editor, i dati su editor e dipartimento non possono essere inseriti sino a che non viene prodotto un report. Infatti report_no è chiave primaria della relazione e quindi non potrebbe avere valore nullo (**anomalia di inserimento**)

report_no	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev
NULL	cippi	18	research	argus 5	NULL	NULL	NULL	...	NULL	NULL	NULL

Motivazione

- Infine, la sola cancellazione di un report (per esempio il report n. 5789) può portare alla perdita di tutti i dati di un suo autore, per esempio *fry* se questi aveva scritto solo il report 5789, (**anomalia di cancellazione**)

report_no	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev

...come anche i dati del dipartimento se ad essere rimosso è l'unico report di quel dipart.

Motivazione

- È possibile stabilire dei criteri in base ai quali valutare il grado di immunità di una base dati rispetto a queste criticità?
- È possibile definire dei criteri per trasformare una base dati in modo da rappresentare le stesse informazioni ma essere immune a queste criticità?
- La risposta a queste domande è data dalle **forme normali** e dal processo di **normalizzazione** di una base dati
 - Prima forma normale (NF)
 - Seconda NF
 - Terza NF
 - Boyce-Codd NF
- Tanto maggiore è il livello della forma normale e tanto maggiore è l'immunità alle criticità

Normalizzazione

- Il processo attraverso il quale uno schema logico può essere trasformato per essere conforme ai requisiti delle forme normali è detto **normalizzazione**.
- Tipicamente, l'obiettivo minimo che ci si prefigge è quello di rendere una base dati in terza forma normale (3NF)
 - Meglio ancora se si riesce ad arrivare alla forma di Boyce-Codd
- Tuttavia, per comprendere le implicazioni della terza forma normale è necessario partire dalle forme normali meno restrittive: la prima e la seconda...

Prima forma normale

- **Def:** Una tabella si dice in **prima forma normale** (1NF) se e solo se ciascun attributo è definito su un dominio con valori atomici, e non c'è più di una colonna riferita ad uno stesso attributo (non ci sono attributi aggregati o attributi multivalore).
- Di fatto, la prima forma normale è considerata parte integrante della definizione formale di relazione nel modello relazionale

Prima forma normale

- Nello schema sotto riportato ci sono delle colonne dedicate alla memorizzazione di un medesimo tipo di attributo: codice, nome ed indirizzo dell'autore.

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	auth_id1	auth_name1	auth_addr1	...	auth_id3	auth_name3	auth_addr3
4216	woolf	15	design	argus 1	53	mantei	cs-tor	...	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone	...	71	koenig	mathrev
6316	woolf	15	design	argus 1	53	mantei	cs-tor	...	44	bolton	mathrev

- Pertanto, questa tabella non è in prima forma normale.

Prima forma normale

- Per portare la tabella in prima forma normale bisogna:
 - sostituire gli attributi aggregati con un unico attributo
 - riportare le informazioni sui diversi valori degli attributi in righe diverse della tabella.

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
4216	woolf	15	design	argus 1	53	mantei	cs-tor
4216	woolf	15	design	argus 1	44	bolton	mathrev
4216	woolf	15	design	argus 1	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone
5789	koenig	27	analysis	argus 2	38	umar	prise
5789	koenig	27	analysis	argus 2	71	koenig	mathrev

Forme normali

- Le successive forme normali (seconda, terza, ...) mirano a distribuire i dati su più relazioni
- Tale distribuzione ha come obiettivo quello di semplificare i **legami funzionali** presenti tra i diversi attributi di una relazione:
 - Fare in modo che gli attributi di una relazione abbiano legami solo con la chiave della relazione
 - Allo stesso tempo, lo schema finale deve preservare tutti i legami funzionali dello schema iniziale
- Vediamo meglio cosa sono questi legami funzionali...

Dipendenze funzionali

- Per esempio, nella relazione Report esiste un legame tra gli attributi `author_id`, `author_name` e `author_addr`
 - Righe diverse con lo stesso valore di `author_id` hanno anche valori uguali su `author_name` e `author_addr`: questi ultimi **dipendono funzionalmente** da `author_id`

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
4216	woolf	15	design	argus 1	53	mantei	cs-tor
4216	woolf	15	design	argus 1	44	bolton	mathrev
4216	woolf	15	design	argus 1	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone
5789	koenig	27	analysis	argus 2	38	umar	prise
5789	koenig	27	analysis	argus 2	71	koenig	mathrev

Dipendenze funzionali

- Questi legami tra attributi sono descritti attraverso il concetto di **dipendenza funzionale**
- **Def:** Data una relazione $R(X)$ e due insiemi di attributi A e B non vuoti e sottoinsiemi di X , si dice che **B è funzionalmente dipendente da A** se il fatto che due generiche tuple abbiano valori uguali sull'insieme di attributi A implica che debbano avere valori uguali anche su B .
- Tale dipendenza viene rappresentata con la seguente simbologia: **$A \rightarrow B$**

Dipendenze funzionali

- In altre parole, la dipendenza funzionale $A \rightarrow B$ è vera se il verificarsi di $t_1[A]=t_2[A]$ implica che $t_1[B]=t_2[B]$
 - Se $A \rightarrow B$ **non può succedere** che due tuple abbiano valori uguali su A e diversi su B
 - Se $A \rightarrow B$ **può** comunque **succedere** che due tuple abbiano valori diversi su A ed uguali su B

A	B	C
Gino	100	novembre
Gino	100	settembre
Carlo	150	giugno
Alfio	100	agosto
Gino	100	agosto
Manfredi	150	giugno

$A \rightarrow B ?$

$B \rightarrow A ?$

$A \rightarrow C ?$

$B \rightarrow C ?$

$C \rightarrow B ?$

Dipendenze funzionali

- Come nel caso del vincolo di chiave, la dipendenza funzionale esprime un **legame** logico che deve essere verificato **a livello di schema** piuttosto che di istanza
 - La dipendenza funzionale tra due attributi esprime un legame che è espressione di un vincolo valido nel contesto rappresentato dalla base dati

Dipendenze funzionali

- Per la tabella sotto mostrata valgono le seguenti dipendenze funzionali:
 - report_no → editor, dept_no
 - dept_no → dept_name, dept_addr
 - author_id → author_name, author_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
4216	woolf	15	design	argus 1	53	mantei	cs-tor
4216	woolf	15	design	argus 1	44	bolton	mathrev
4216	woolf	15	design	argus 1	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone
5789	koenig	27	analysis	argus 2	38	umar	prise
5789	koenig	27	analysis	argus 2	71	koenig	mathrev

Dipendenze funzionali

- **Def:** Data una relazione $R(X)$ e due insiemi di attributi A e B non vuoti e sottoinsiemi di X , la dipendenza funzionale $A \rightarrow B$ si dice **non banale** se $A \cap B = \emptyset$
- Una dipendenza funzionale banale $A \rightarrow B$ può essere trasformata in una non banale se esiste un sottoinsieme non vuoto di attributi di B che non è incluso in A
- Per esempio, se A , B e C sono tre insiemi di attributi, la dipendenza funzionale banale $AC \rightarrow BC$ può essere trasformata in $AC \rightarrow B$ che è non banale se $AC \cap B = \emptyset$
- **Attenzione:** sarebbe **errato** dire che se $AC \rightarrow BC$ allora $A \rightarrow B$

Dipendenze funzionali

- Per la definizione di chiave, data una relazione $R(X)$ con chiave K esiste una dipendenza funzionale tra K ed un qualunque sottoinsieme degli attributi di X
 - Infatti non è possibile trovare due tuple distinte che abbiano ugual valore in K ...
- In questo senso si è soliti dire che **il vincolo di dipendenza funzionale generalizza il vincolo di chiave**
- Una dipendenza funzionale $A \rightarrow B$ su uno schema $R(X)$ degenera nel vincolo di chiave se $A \cup B = X$.
In questo caso A è superchiave per $R(X)$
 - Si dimostra per assurdo ...

Dipendenze funzionali

- Uno schema di relazione $R(X)$ su cui è definito un insieme F di dipendenze funzionali (df) è denotato con $\langle R(X), F \rangle$ e può soddisfare anche altre df oltre a quelle in F
- Esempio: $\langle R(A,B,C), F = \{A \rightarrow B, B \rightarrow C\} \rangle \dots$

è facile dimostrare che dalle due df in F segue che $A \rightarrow C$.

In questi casi si dice che $A \rightarrow C$ è **implicata logicamente** dalle df in F

$$\{A \rightarrow B, B \rightarrow C\} \Rightarrow A \rightarrow C$$

Dipendenze Funzionali

- **Def:** Dato $\langle R(X), F \rangle$ l'insieme F^+ di tutte le df implicate logicamente da quelle in F è detto **chiusura** di F

$$F^+ = \{ A \rightarrow B \mid F \Rightarrow A \rightarrow B \}$$

- La chiusura di F gode delle seguenti proprietà:
 - **Contenimento** $F \subseteq F^+$
 - **Idempotenza** $F^+ = (F^+)^+$
 - **Monotonicità** $F \subseteq G \Rightarrow F^+ \subseteq G^+$

Dipendenze Funzionali

- Per conoscere la chiusura F^+ di un dato insieme F di df bisogna calcolare tutte le dipendenze funzionali logicamente implicate dalle df in F
- Questo calcolo si basa su semplici regole di inferenza note come **Assiomi di Armstrong**

Riflessività (r) se $X \supseteq Y \Rightarrow X \rightarrow Y$

Estensione (e) se $X \rightarrow Y \Rightarrow XZ \rightarrow YZ$

Transitività (t) se $X \rightarrow Y$ e $Y \rightarrow Z \Rightarrow X \rightarrow Z$

Dipendenze Funzionali

- Per semplificare il processo di derivazione sono inoltre applicabili le seguenti regole (tutte derivabili dagli assiomi di Armstrong)

Unione (u) se $X \rightarrow Y$ e $X \rightarrow Z \Rightarrow X \rightarrow YZ$

Pseudo-transitiva (p) se $X \rightarrow Y$ e $WY \rightarrow Z \Rightarrow WX \rightarrow Z$

Decomposizione (d) se $X \rightarrow Z$ e $Z \supseteq Y \Rightarrow X \rightarrow Y$

Dipendenze Funzionali

- Dato $\langle R(X), F \rangle$, la **chiusura** Y^+ **di un insieme di attributi** $Y \subseteq X$ rispetto ad F è l'insieme di tutti gli attributi di cui si può dimostrare la dipendenza funzionale da Y usando le df in F^+
- Esempio: dato lo schema $\langle R(A,B,C,D,E), F = \{A \rightarrow BD, BC \rightarrow D, D \rightarrow E\} \rangle$ dimostrare che la chiusura $(AC)^+$ di AC è l'insieme $ABCDE$
 1. Usando (e) $A \rightarrow BD \Rightarrow AC \rightarrow ABCD$
 2. Sempre usando (e) $D \rightarrow E \Rightarrow ABCD \rightarrow ABCDE$
 3. Pertanto, usando (t) sui due precedenti risultati si ricava che $AC \rightarrow ABCDE$ cioè $(AC)^+ = ABCDE$

Dipendenze Funzionali

- **Def:** Dato $\langle R(X), F \rangle$, una dipendenza f si dice **ridondante** rispetto ad F se $f \in F^+$, vale a dire se f è implicata logicamente da F

$$F \Rightarrow f$$

- **Def:** Due insiemi di df F e G definiti sullo stesso schema $R(X)$ si dicono **equivalenti** se e solo se

$$F^+ = G^+$$

Dipendenze Funzionali

- Problema: dati due insiemi di df F e G come fare a dimostrare la loro equivalenza?
 - Come si fa a sapere se F^+ e G^+ sono uguali ?
- Calcolare esplicitamente F^+ e G^+ potrebbe essere una procedura lunga e complicata ...
è necessario ricorrere ad una strategia alternativa

Dipendenze Funzionali

- In realtà basta dimostrare che tutte le df in F sono implicate da quelle in G (e quindi $F^+ \subseteq G^+$) e viceversa tutte le df in G sono implicate da quelle in F (e quindi $G^+ \subseteq F^+$). Da cui $F^+ = G^+$
- **Prop:** Condizione necessaria e sufficiente affinché due insiemi di df siano equivalenti è che tutte le df del primo insieme siano logicamente implicate da quelle del secondo e viceversa

Dipendenze Funzionali

- Esempio: dimostrare che i due insiemi F e G sono equivalenti

$R(A,B,C)$, $F=\{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$, $G=\{A \rightarrow B, B \rightarrow A, A \rightarrow C, C \rightarrow A\}$

Si considerano solo le df che non appartengono a $F \cap G$

Dipendenze Funzionali

- **Def:** Dato $\langle R(X), F \rangle$, sia $AX \rightarrow Y$ una df appartenente ad F . Si dice che A è **estraneo** in $AX \rightarrow Y$ rispetto ad F se $F^+ = G^+$ dove G è l'insieme ottenuto sostituendo in F la dipendenza $AX \rightarrow Y$ con $X \rightarrow Y$

$$G = F - \{AX \rightarrow Y\} + \{X \rightarrow Y\}$$

Dipendenze Funzionali

- Esempio: dimostrare che nel seguente schema sia A che B sono estranei (non contemporaneamente) nella prima df
 $\langle R(A,B,C), F=\{AB \rightarrow C, B \rightarrow A, A \rightarrow B\} \rangle$

Dipendenze Funzionali

- **Def:** Dato $\langle R(X), F \rangle$, si dice che un insieme di df G costituisce una **copertura minima** di F se valgono le seguenti proprietà:
 - $F^+ = G^+$
 - Ogni **dipendenza** in G è **semplice**, ovvero del tipo $A \rightarrow Y$ con Y attributo singolo
 - Nessuna df in G contiene attributi estranei
 - Nessuna df in G è ridondante (rispetto alle altre presenti in G)

Dipendenze Funzionali

- Dato $\langle R(X), F \rangle$, l'algoritmo per calcolare la copertura minima di F opera nei seguenti passi:
 1. Trasforma ogni df di F in una df semplice
 2. Per ogni df rimuove gli attributi estranei
 3. Elimina ogni df ridondante

Dipendenze Funzionali

- Esempio, trovare la copertura minima per il seguente schema:
 $\langle R(A,B,C), F=\{AB \rightarrow C, B \rightarrow A, C \rightarrow A\} \rangle$

Esercizi

- Trovare le coperture minime per i seguenti insiemi
 - $F = \{AC \rightarrow B, D \rightarrow E, B \rightarrow E, BE \rightarrow D, A \rightarrow D, E \rightarrow C\}$
 - $F = \{ABC \rightarrow DE, C \rightarrow D, BC \rightarrow A, D \rightarrow A\}$
 - $F = \{AB \rightarrow D, C \rightarrow E, DE \rightarrow A, C \rightarrow D\}$
 - $F = \{AB \rightarrow D, C \rightarrow EA, CE \rightarrow DA, B \rightarrow C\}$

Dipendenza completa

- La definizione di Seconda Forma Normale si basa sui concetti di dipendenza completa ed attributo primo
- **Def:** Data una dipendenza funzionale $A \rightarrow B$, si dice che B **dipende completamente** da A se per ogni sottoinsieme $Y \subset A$ la dipendenza $Y \rightarrow B$ è falsa (vale a dire se $A \rightarrow B$ non contiene attributi estranei)
- **Def:** Data una dipendenza funzionale $A \rightarrow B$, si dice che B **dipende parzialmente** da A se non ne dipende completamente

Attributo primo

- **Def:** Data una relazione $R(X)$ ed A un generico sottoinsieme di X , A è detto **attributo primo** (prime attribute) se è contenuto in una qualche chiave di $R(X)$

Seconda forma normale

- **Def:** Una tabella si dice in **seconda forma normale** (2NF) se e solo se è in 1NF e ciascun attributo non primo è completamente dipendente dalla chiave primaria

Seconda forma normale

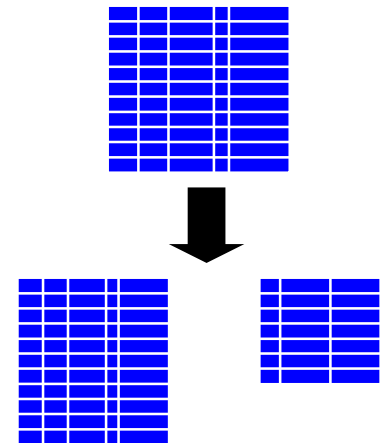
- Consideriamo la seguente tabella:
 - La chiave è (report_no, author_id) e le df valide sono:
 - report_no→editor, dept_no
 - dept_no→dept_name, dept_addr
 - author_id→author_name, author_addr
- La tabella è in seconda forma normale?

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
4216	woolf	15	design	argus 1	53	mantei	cs-tor
4216	woolf	15	design	argus 1	44	bolton	mathrev
4216	woolf	15	design	argus 1	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone
5789	koenig	27	analysis	argus 2	38	umar	prise
5789	koenig	27	analysis	argus 2	71	koenig	mathrev

Decomposizioni

- Il processo di normalizzazione di una base dati (e quindi di ciascuna delle sue relazioni) si basa sulla **decomposizione delle relazioni** che non sono in forma normale
- Il processo di decomposizione è un processo di distribuzione dei dati rappresentati in una relazione su due o più relazioni
- Ai fini della normalizzazione si cerca di decomporre le relazioni in nuove relazioni che soddisfino i vincoli delle forme normali



Decomposizioni

- **Def:** Una **decomposizione** di una relazione $R(X)$ è un insieme di relazioni R_1, \dots, R_n ottenute proiettando $R(X)$ su un insieme di attributi A_1, \dots, A_n tali che:

$$A_1 \cup \dots \cup A_n = X$$

Decomposizioni

- Data la relazione ...

Studente(matricola, nome, cognome, telefono)

- ...quali tra quelle che seguono sono decomposizioni di **Studente**?

R1(matricola), R2(cognome, telefono)

R1(matricola, nome), R2(cognome, telefono)

R1(matricola, nome, cognome), R2(cognome, telefono)

Decomposizioni

- Tuttavia, non tutte le decomposizioni di una relazione $R(X)$ sono in grado di rappresentare in modo equivalente i dati contenuti in $R(X)$:
 - Dipende dalla scelta degli attributi A_1, \dots, A_n su cui si effettua la decomposizione

Decomposizioni

- Tuttavia, non tutte le decomposizioni di una relazione $R(X)$ sono in grado di rappresentare in modo equivalente i dati contenuti in $R(X)$:
 - Decomporre la relazione

Studente(matricola, nome, cognome, telefono)

in **R1**(matricola, nome) **R2**(cognome, telefono)

porta alla perdita del legame logico tra la matricola di uno studente ed il suo cognome (e viceversa)

Decomposizioni

- Tuttavia, non tutte le decomposizioni di una relazione $R(X)$ sono in grado di rappresentare in modo equivalente i dati contenuti in $R(X)$:
 - Decomporre la relazione

Studiante(matricola, nome, cognome, telefono)

in **R1**(matricola, nome, cognome) **R2**(cognome, telefono)

può ugualmente portare ad una perdita di informazioni.
Infatti...

Decomposizioni

Matricola	Nome	Cognome	Telefono
M0001	Carlo	Bianchi	055210000
M0002	Giulio	Rossi	055230000
M0003	Anna	Bianchi	055240000
M0004	Paola	Tarli	055250000
M0005	Arturo	Pucci	055260000



R1

Matricola	Nome	Cognome
M0001	Carlo	Bianchi
M0002	Giulio	Rossi
M0003	Anna	Bianchi
M0004	Paola	Tarli
M0005	Arturo	Pucci

R2

Cognome	Telefono
Bianchi	055210000
Rossi	055230000
Bianchi	055240000
Tarli	055250000
Pucci	055260000

Che succede quando eseguo una query per ricostruire matricola, nome, cognome e telefono degli studenti?

Select R1.*, R2.telefono

From R1, R2 where R1.cognome=R2.cognome

Decomposizioni

Matricola	Nome	Cognome	Telefono
M0001	Carlo	Bianchi	055210000
M0002	Giulio	Rossi	055230000
M0003	Anna	Bianchi	055240000
M0004	Paola	Tarli	055250000
M0005	Arturo	Pucci	055260000

Matricola	Nome	Cognome	Telefono
M0001	Carlo	Bianchi	055210000
M0001	Carlo	Bianchi	055240000
M0002	Giulio	Rossi	055230000
M0003	Anna	Bianchi	055240000
M0003	Anna	Bianchi	055210000
M0004	Paola	Tarli	055250000
M0005	Arturo	Pucci	055260000

Compaiono tuple contenenti dati errati che non facevano parte della relazione iniziale

Decomposizioni

- In altri casi invece la relazione che si ottiene effettuando il JOIN è proprio la relazione originale.
- Per identificare queste ultime decomposizioni si introduce la seguente definizione
- **Def:** una decomposizione di $R(X)$ in $R_1(A)$ ed $R_2(B)$ si dice **lossless-join** se $R(X) = R_1(A) \text{ JOIN } R_2(B)$
- Una decomposizione non lossless-join è detta lossy-join o semplicemente lossy (decomposizione con perdita).

Decomposizioni

- Le uniche decomposizioni utilizzabili per pervenire alla trasformazione di uno schema in una forma normale sono le decomposizioni senza perdita: **lossless-join**
- E' quindi necessario un criterio per stabilire sulla base degli attributi su cui viene effettuata la decomposizione se questa sia o meno senza perdita.

Decomposizioni

- **Prop:** Data una relazione $R(X)$ siano A e B due sottoinsiemi di X tali che $A \cup B = X$ e sia $R_1(A)$ ed $R_2(B)$ una decomposizione di $R(X)$ su A, B ;
Sia inoltre $C = A \cap B$
Condizione sufficiente affinché la decomposizione sia senza perdita è che **C** sia **superchiave** per almeno una delle due relazioni $R_1(A)$ ed $R_2(B)$.
- In altri termini, deve valere almeno una delle due seguenti dipendenze funzionali:
 $C \rightarrow A$
 $C \rightarrow B$

Decomposizioni

- La precedente proposizione descrive un **criterio di sufficienza** affinché la decomposizione sia senza perdita.
- Il criterio non è tuttavia necessario:
 - esistono decomposizioni senza perdita per le quali non è vera nessuna delle due dipendenze funzionali $C \rightarrow A$ e $C \rightarrow B$.

Conservazione delle dipendenze

- Un'ulteriore proprietà delle decomposizioni descrive come queste agiscono relativamente ai vincoli di dipendenza funzionale definiti sulla relazione di partenza.
- Intuitivamente, si può considerare che se su una relazione $R(X)$ vale la dipendenza funzionale $A \rightarrow B$ e la relazione viene decomposta su due insiemi tali che uno non contiene A e l'altro non contiene B , allora la decomposizione non consente di preservare la dipendenza funzionale.

Conservazione delle dipendenze

- Dato uno schema $\langle R(X), F \rangle$ una decomposizione di $R(X)$ in $R_1(X_1)$ ed $R_2(X_2)$ induce una proiezione di F su due insiemi F_1 ed F_2 di dipendenze funzionali tali che:
 - F_1 contiene le df di F che coinvolgono solo attributi inclusi in X_1 .
 - F_2 contiene le df di F che coinvolgono solo attributi inclusi in X_2 .
- Tutte le dipendenze in F che coinvolgono attributi che non sono interamente contenuti né in X_1 né in X_2 vengono perse nella proiezione
- **Def:** si dice che la decomposizione di $R(X)$ in $R_1(X_1)$ ed $R_2(X_2)$ **preserva le dipendenze** in F se e solo se

$$F^+ = (F_1 \cup F_2)^+$$

Conservazione delle dipendenze

- **Nota:** una decomposizione può provocare la perdita di alcune dipendenze funzionali. Tuttavia questo non si traduce in una perdita di informazioni se le dipendenze funzionali che si perdono sono logicamente implicate da quelle che rimangono

Conservazione delle dipendenze

- Esempio: $\langle R(A,B,C), F=\{A \rightarrow B, B \rightarrow C, A \rightarrow C\} \rangle$
- La decomposizione di R in **R1(A,B)** ed **R2(B,C)** preserva le dipendenze di F?

$$F1 = \{A \rightarrow B\}$$

$$F2 = \{B \rightarrow C\}$$

La dipendenza $A \rightarrow C$ viene persa, tuttavia...

$$A \rightarrow C \subset \{F1 \cup F2\}^+ = \{A \rightarrow B, B \rightarrow C\}^+$$

Seconda forma normale

- **Def:** Una tabella si dice in **seconda forma normale** (2NF) se e solo se è in 1NF e ciascun attributo non primo è completamente dipendente dalla chiave primaria

Seconda forma normale

- Consideriamo la seguente tabella:
 - La chiave è (report_no, author_id) e le df valide sono:
 - report_no→editor, dept_no
 - dept_no→dept_name, dept_addr
 - author_id→author_name, author_addr
- Alcuni attributi dipendono solo da report_no, altri solo da author_id
 - Quindi, la tabella non è in seconda forma normale.

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
4216	woolf	15	design	argus 1	53	mantei	cs-tor
4216	woolf	15	design	argus 1	44	bolton	mathrev
4216	woolf	15	design	argus 1	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone
5789	koenig	27	analysis	argus 2	38	umar	prise
5789	koenig	27	analysis	argus 2	71	koenig	mathrev

Seconda forma normale

- I dati rappresentati nella tabella possono essere meglio rappresentati decomponendo la tabella in più tabelle che soddisfino le condizioni della seconda forma normale:
 - all'interno di ciascuna tabella la dipendenza dalla chiave primaria di ciascun attributo non chiave deve essere completa.
- La decomposizione dovrà essere senza perdita e dovrà preservare le dipendenze funzionali

Seconda forma normale

- Nell'esempio considerato valgono le seguenti dipendenze funzionali:
 - report_no → editor, dept_no
 - dept_no → dept_name, dept_addr
 - author_id → author_name, author_addr
- I membri destri delle prime due dipendenze dipendono da report_no
- Il membro destro della terza dipendenza dipende da author_id

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
4216	woolf	15	design	argus 1	53	mantei	cs-tor
4216	woolf	15	design	argus 1	44	bolton	mathrev
4216	woolf	15	design	argus 1	71	koenig	mathrev
5789	koenig	27	analysis	argus 2	26	fry	folkstone
5789	koenig	27	analysis	argus 2	38	umar	prise
5789	koenig	27	analysis	argus 2	71	koenig	mathrev

Seconda forma normale

- L'analisi delle dipendenze funzionali suggerisce di “togliere” dalla tabella iniziale due diversi gruppi di dati:
 - Gli attributi che dipendono da author_id
 - author_id, author_name, author_addr
 - Gli attributi che dipendono da report_no
 - report_no, editor, dept_no, dept_name, dept_addr



author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- Consideriamo il primo gruppo di attributi:
 - author_id, author_name, author_addr
- Se da soli costituissero lo schema di una relazione, questa rispetterebbe la II forma normale?

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- È possibile decomporre (senza perdita di informazioni) la tabella iniziale in due tabelle A e B, con B definita sugli attributi
 - author_id, author_name, author_addr ?

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- Come facciamo a sapere se la decomposizione è senza perdita?

A(author_id, report_no, editor, dept_no, dept_name, dept_addr)

B(author_id, author_name, author_addr)

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- Per il legame tra DF e chiave, author_id è chiave di B
- Quindi l'intersezione tra gli attributi dello schema decomposto è superchiave di una delle due relazioni

A(author_id, report_no, editor, dept_no, dept_name, dept_addr)

B(author_id, author_name, author_addr)

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- La tabella A non è ancora in 2NF: va decomposta in due tabelle C e D con D definita su
 - report_no, editor, dept_no, dept_name, dept_addr
- Come deve essere C ?

A(author_id, report_no, editor, dept_no, dept_name, dept_addr)

B(author_id, author_name, author_addr)

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- Come facciamo a sapere se la decomposizione è senza perdita?

C(author_id, report_no)

D(report_no, editor, dept_no, dept_name, dept_addr)

B(author_id, author_name, author_addr)

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- Per il legame tra DF e chiave, report_no è chiave di D
- Quindi l'intersezione tra gli attributi dello schema decomposto è superchiave di una delle due relazioni

C(author_id, report_no)

D(report_no, editor, dept_no, dept_name, dept_addr)

B(author_id, author_name, author_addr)

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- A questo punto tutte le tabelle sono in 2NF:
 - Verificare...

C(author_id, report_no)

D(report_no, editor, dept_no, dept_name, dept_addr)

B(author_id, author_name, author_addr)

author_id, author_name, author_addr

report_no, editor, dept_no, dept_name, dept_addr

Report

<u>report_no</u>	editor	dept_no	dept_name	dept_addr	<u>author_id</u>	author_name	author_addr
------------------	--------	---------	-----------	-----------	------------------	-------------	-------------

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Seconda forma normale

- Dipendenze funzionali:
- Report1 (D):
report_no → editor, dept_no
dept_no → dept_name, dept_addr
- Report2 (B):
author_id → author_name, author_addr
- Report3 (C):
nessuna dipendenza funzionale:
l'insieme di tutti gli attributi costituisce
la chiave primaria

Report 1

<u>report_no</u>	editor	dept_no	dept_name	dept_addr
4216	woolf	15	design	argus 1
5789	koenig	27	analysis	argus 2

Report 2

<u>author_id</u>	author_name	author_addr
53	mantei	cs-tor
44	bolton	mathrev
71	koenig	mathrev
26	fry	folkstone
38	umar	prise

Report 3

<u>report_no</u>	<u>author_id</u>
4216	53
4216	44
4216	71
5789	26
5789	38
5789	71

Seconda forma normale

- Le tabelle in 2NF rappresentano un significativo miglioramento rispetto a quelle 1NF, tuttavia risentono ancora di alcune inefficienze
 - la presenza di **dipendenze funzionali transitive** (ammesse dalla 2NF) è sintomatica del fatto che nella tabella vengono rappresentate due tipologie di informazioni diverse, una per ogni dipendenza funzionale.

Seconda forma normale

- Un miglioramento della seconda forma normale è costituito dalla terza forma normale
- La terza forma normale è più restrittiva della seconda e garantisce la minor occorrenza di anomalie

Seconda forma normale

- Nella tabella Report1 è presente una dipendenza funzionale transitiva tra alcuni attributi e la chiave primaria:
 $\text{report_no} \rightarrow \text{dept_no} \rightarrow \text{dept_name, dept_addr}$
- A causa di questo,
 - i dati su nome ed indirizzo di un dipartimento compaiono in modo ridondante nella tabella Report1 (anom. agg.)
 - rimuovendo un report dalla tabella Report1 si cancellano anche i dati del dipartimento associato al report in questione (anom. canc.)
 - non si può inserire un nuovo dipartimento se non è stato ancora prodotto un report (anom. inser.)

Report 1

<u>report_no</u>	editor	dept_no	dept_name	dept_addr
4216	woolf	15	design	argus 1
5789	koenig	27	analysis	argus 2

Report 2

<u>author_id</u>	author_name	author_addr
53	mantei	cs-tor
44	bolton	mathrev
71	koenig	mathrev
26	fry	folkstone
38	umar	prise

Report 3

<u>report_no</u>	<u>author_id</u>
4216	53
4216	44
4216	71
5789	26
5789	38
5789	71

Terza forma normale

- **Def:** Una tabella si dice in **terza forma normale** (3NF) se e solo se è in 1NF e per ogni dipendenza funzionale non banale $X \rightarrow A$ è vera una delle due condizioni seguenti:
 - X è una superchiave
 - A è attributo primo

Terza forma normale

- La tabella Report1 precedentemente definita non rispetta la condizione di 3NF
- Infatti, la dipendenza funzionale $\text{dept_no} \rightarrow \text{dept_name}, \text{dept_addr}$ non verifica nessuna delle due condizioni previste per la 3NF:
 - dept_no **NON E'** una superchiave
 - (dept_name, dept_addr) **NON E'** parte di una chiave

Report 1

<u>report_no</u>	editor	dept_no	dept_name	dept_addr
4216	woolf	15	design	argus 1
5789	koenig	27	analysis	argus 2

Terza forma normale

- La normalizzazione della tabella prevede la sua decomposizione in due tabelle di cui una definita sullo schema della dipendenza funzionale:

(dept_no, dept_name, dept_addr)

Report 1

<u>report_no</u>	editor	dept_no	dept_name	dept_addr
4216	woolf	15	design	argus 1
5789	koenig	27	analysis	argus 2

report_no → editor, dept_no

dept_no → dept_name, dept_addr

author_id → author_name, author_addr

Report 11

<u>report_no</u>	editor	dept_no
4216	woolf	15
5789	koenig	27

Report 12

<u>dept_no</u>	dept_name	dept_addr
15	design	argus 1
27	analysis	argus 2

Terza forma normale

- Per cui il risultato della normalizzazione è:

Report 11

<u>report_no</u>	editor	dept_no
4216	woolf	15
5789	koenig	27

Report 12

<u>dept_no</u>	dept_name	dept_addr
15	design	argus 1
27	analysis	argus 2

Report 2

<u>author_id</u>	author_name	author_addr
53	mantei	cs-tor
44	bolton	mathrev
71	koenig	mathrev
26	fry	folkstone
38	umar	prise

Report 3

<u>report_no</u>	<u>author_id</u>
4216	53
4216	44
4216	71
5789	26
5789	38
5789	71

report_no→editor, dept_no

dept_no→dept_name, dept_addr

author_id→author_name, author_addr

Terza forma normale

- Dimostrare che se una tabella è in III-NF allora è anche in II-NF
- **Def:** Una tabella si dice in terza forma normale (3NF) se e solo se è in 1NF e per ogni dipendenza funzionale non banale $X \rightarrow A$ tra due insiemi di attributi X ed A è vera una delle due condizioni seguenti:
 - X è una superchiave
 - A è attributo primo

Terza forma normale

- La terza forma normale elimina la maggior parte delle anomalie conosciute per le basi dati e rappresenta un livello qualitativo soddisfacente nella maggior parte dei casi
- Le rimanenti anomalie possono essere rimosse attraverso la trasformazione delle tabelle nelle forme normali successive alla terza
- La prima tra queste è la forma normale di Boyce-Codd che è considerata una forte variazione rispetto alla 3NF

Forma normale di Boyce-Codd

- **Def:** Una tabella si dice in **forma normale di Boyce-Codd** (BCNF) se e solo se è in 1NF e per ogni dipendenza funzionale non banale $X \rightarrow A$, X è una superchiave

Forma normale di Boyce-Codd

- Questa forma normale rappresenta una ulteriore restrizione rispetto alla 3NF in quanto non prevede l'esistenza di dipendenze funzionali in cui il termine destro sia semplicemente parte di una chiave:
 - Tutte le dipendenze funzionali devono avere come membro sinistro una superchiave
- Se una relazione è in BCNF ogni campo di ciascuna tupla registra un dato che **NON PUO'** essere dedotto dai valori degli altri campi non chiave delle tuple della relazione

Forma normale di Boyce-Codd

- Se una tabella verifica la forma normale di Boyce-Codd questa verifica anche la terza forma normale
- Invece non è detto che una tabella che verifichi la 3NF debba anche rispettare la Boyce-Codd
- In questi casi, si possono avere anomalie di cancellazione.
- Vediamo un esempio...

Forma normale di Boyce-Codd

Insegnamento(studente, corso, docente)

...memorizza le informazioni sui corsi seguiti dagli studenti e relativi docenti

- E' noto che:
 - Regola 1: Ciascun docente tiene un solo corso:
docente → corso
 - Regola 2: Un corso può essere tenuto da più docenti, ma per ogni studente, ciascun corso ha un solo docente:
studente, corso → docente
- Una chiave della relazione Insegnamento è (studente, corso)
 - Un'altra è (studente, docente)

Forma normale di Boyce-Codd

- A causa della df **docente** \rightarrow **corso** la tabella non è in BCNF, pur essendo in 3NF
 - corso è parte di una chiave
- Inoltre, a causa della df **studente, corso** \rightarrow **docente** non è possibile definire nessuna decomposizione che conservi le dipendenze

Insegnamento(studente, corso, docente)

docente \rightarrow **corso**

studente, corso \rightarrow **docente**

Forma normale di Boyce-Codd

- Questo significa che la relazione di partenza non può essere trasformata in un insieme di relazioni che soddisfino la forma normale di Boyce-Codd
- In generale:
 - è garantita la raggiungibilità della terza forma normale,
 - non è affatto garantita la raggiungibilità della forma normale di Boyce-Codd

Forme normali

- Esistono anche forme normali superiori alla BCNF, in particolare la 4NF e 5NF
 - si basano sul concetto di dipendenza multivalore (MVD)
- Per approfondimenti vedere Cap 11 Elmasri-Navathe

Procedura di normalizzazione

- Passi da seguire per trasformare uno schema $\langle R(X), F \rangle$ in 3NF
 1. **Copertura minima**: si determina una copertura minima G per l'insieme F . In base a questa si individua una chiave K e si determina la forma normale dello schema $R(X)$
 2. **Partizionamento**: si dividono le df di G in gruppi con lo stesso determinante (gli attributi a sx della df)
 3. **Fusione**: se gli attributi di una df in un gruppo G_1 sono **tutti** inclusi tra gli attributi che compaiono in un gruppo G_2 allora la df viene spostata da G_1 a G_2 (questo passo può portare a schemi in III-NF non trasformabili in BC-NF)
 4. **Sintesi schemi**: per ogni gruppo si costruisce uno schema usando tutti e soli gli attributi che compaiono in almeno una df del gruppo. Ogni schema ha come chiave i determinanti del gruppo individuati al punto 2
 5. **Decomposizione Lossless**: se **non esiste** uno schema che ha come chiave K quella individuata al passo 1, si aggiunge un nuovo schema di relazione con i soli attributi K

Esercizi normalizzazione

- Determinare la forma normale dei seguenti schemi (sono già cop.min.) trasformandoli almeno in III-NF
 - $\langle R(A,B,C,D,E) \ F=\{A \rightarrow B, D \rightarrow E, B \rightarrow D, E \rightarrow C\} \rangle$
chiave...
 - $\langle R(A,B,C,D,E) \ F=\{BC \rightarrow E, C \rightarrow D, D \rightarrow A\} \rangle$
chiave...
 - $\langle R(A,B,C,D,E) \ F=\{AB \rightarrow D, C \rightarrow E, DE \rightarrow A, C \rightarrow D\} \rangle$
chiave...
 - $\langle R(A,B,C,D,E) \ F=\{C \rightarrow E, C \rightarrow A, C \rightarrow D, B \rightarrow C\} \rangle$
chiave...
 - $\langle R(A,B,C,D,E) \ F=\{A \rightarrow B, D \rightarrow C, A \rightarrow E, BD \rightarrow A\} \rangle$
chiave...

Gestione schemi non trasformabili in BC-NF

- Riprendiamo l'esempio visto...

Insegnamento(studente, corso, docente)

- E' noto che:
 - Regola 1: Ciascun docente tiene un solo corso:
docente → corso
 - Regola 2: Un corso può essere tenuto da più docenti, ma per ogni studente, ciascun corso ha un solo docente:
studente, corso → docente
- Una chiave della relazione Insegnamento è (studente, corso)
 - Un'altra è (studente, docente)

Gestione schemi non trasformabili in BC-NF

- A causa della df **docente** \rightarrow **corso** la tabella non è in BCNF, pur essendo in 3NF
 - corso è parte di una chiave
- Inoltre, a causa della df **studente, corso** \rightarrow **docente** non è possibile definire nessuna decomposizione che conservi tutte le dipendenze

Insegnamento(studente, corso, docente)

docente \rightarrow **corso**

studente, corso \rightarrow **docente**

Gestione schemi non trasformabili in BC-NF

- La non raggiungibilità di BoyceCodd è associata ad anomalie:
 - **Di inserimento:** l'affidamento di un corso ad un docente non può essere registrato sino a che non c'è uno studente per il corso
 - **Di aggiornamento:** se cambia la docenza di un corso devo modificare tutte le tuple dove compare la coppia «docente, corso»
 - **Di cancellazione:** se vengono rimosse le righe degli studenti che seguono un corso, si perde l'informazione sul nome del docente incaricato del corso
- In presenza di schemi che siano in III-NF e non siano trasformabili in BC-NF sono possibili diversi scenari...

Insegnamento(studente, corso, docente)

docente → corso

studente, corso → docente

Gestione schemi non trasformabili in BC-NF

1. Affiancare alla tabella in III-NF una seconda tabella (definita sugli attributi che non rispettano la BC-NF) che introducendo una ridondanza elimini l'anomalia di cancellazione. Usare dei trigger per mantenere allineati i dati ridondanti
2. Decomporre lo schema in modo da raggiungere la BC-NF perdendo una dipendenza funzionale. Questa dovrà essere garantita attraverso l'aggiunta di un vincolo supplementare sulla base dati, per esempio attraverso lo strumento dei **constraint trigger**
3. Lasciare lo schema inalterato nella consapevolezza che sono possibili anomalie di cancellazione

Insegnamento(studente, corso, docente)

docente → corso

studente, corso → docente

Gestione schemi non trasformabili in BC-NF

- **Eliminare l'anomalia introducendo una ridondanza**
 - Si aggiunge un'altra tabella su cui docente è chiave:
Docenza(docente, corso)
- Si introducono dei trigger per mantenere allineati i dati ridondanti
- Tale soluzione è in generale da preferirsi rispetto alla soluzione che non crea ridondanza ma lascia l'anomalia di cancellazione

Insegnamento(studente, corso, docente)

docente → corso

studente, corso → docente

Gestione schemi non trasformabili in BC-NF

- Decomporre la tabella iniziale seguendo uno schema che non preserva le df
- Il primo schema della decomposizione è individuato dalla df tra docente e corso

Docenza(docente, corso)

Insegnamento(studente, corso, docente)

docente → corso

studente, corso → docente

Gestione schemi non trasformabili in BC-NF

- Decomporre la tabella iniziale seguendo uno schema che non preserva le df
- Il primo schema della decomposizione è individuato dalla df tra docente e corso

Docenza(docente, corso)

- Il secondo schema deve contenere *studente* per essere una decomposizione e *docente* per non avere perdita di dati

Studente(studente, docente)

Insegnamento(studente, corso, docente)

docente → corso

studente, corso → docente

Gestione schemi non trasformabili in BC-NF

- Nota: così facendo si perde la df **studente, corso → docente**
 - Nel vecchio schema **Insegnamento** non era possibile mettere due righe in cui uno stesso studente veniva abbinato allo stesso corso tenuto da docenti diversi. Questo perché (studente, corso) è PK
 - Nel nuovo schema posso invece aggiungere alla tabella **Studente** due righe in cui uno stesso studente viene abbinato a due docenti che insegnano lo stesso corso. Nessun vincolo PK o FK è violato.
 - Affinché la regola sia preservata è necessario che ad ogni riga inserita in **Studente** si verifichi il rispetto della regola in base al contenuto di **Docente**

Insegnamento(studente, corso, docente)
docente → corso
studente, corso → docente

Docenza(docente, corso)
Studente(studente, docente)

Gestione schemi non trasformabili in BC-NF

- Nota: così facendo si perde la df **studente, corso → docente**
 - Nel vecchio schema **Insegnamento** non era possibile mettere due righe in cui uno stesso studente veniva abbinato allo stesso corso tenuto da docenti diversi. Questo perché (studente,corso) è PK
 - Nel nuovo schema posso invece aggiungere alla tabella **Studente** due righe in cui uno stesso studente viene abbinato a due docenti che insegnano lo stesso corso. Nessun vincolo PK o FK è violato.
 - Affinché la regola sia preservata è necessario che ad ogni riga inserita in **Studente** si verifichi il rispetto della regola in base al contenuto di **Docente**

La verifica è fatta da un trigger attivato ad ogni INSERT su Studente

Insegnamento(studente, corso, docente)
docente → corso
studente, corso → docente

Docenza(docente, corso)
Studente(studente, docente)

Gestione schemi non trasformabili in BC-NF

```
create schema bcnf;  
create table bcnf.Docenza(docente char(20) primary key, corso char(20));  
create table bcnf.Studente(studente char(20),  
                             docente char(20) references bcnf.Docenza,  
                             primary key(studente, docente));
```

```
insert into bcnf.Docenza values ('Modica', 'analisi I'), ('Giaquinta', 'analisi I');
```

- Vogliamo impedire che nella tabella studente vengano aggiunte righe che associano ad un medesimo studente due docenti che insegnano lo stesso corso, per esempio:

```
insert into bcnf.Studente values ('Pala','Modica');
```

```
...
```

```
insert into bcnf.Studente values ('Pala','Giaquinta');
```

Gestione schemi non trasformabili in BC-NF

```
create schema bcnf;
```

```
create table bcnf.Docenza(docente char(20) primary key, corso char(20));
```

```
create table bcnf.Studente(studente char(20),  
                             docente char(20) references bcnf.Docenza,  
                             primary key(studente, docente));
```

```
insert into bcnf.Docenza values ('Modica', 'analisi I'), ('Giaquinta', 'analisi I');
```

- Vogliamo impedire che nella tabella studente vengano aggiunte righe che associano ad un medesimo studente due docenti che insegnano lo stesso corso
 - Bisogna definire una SELECT, da eseguire ad ogni inserimento o aggiornamento di riga nella tabella Studente, per calcolare il numero di docenti abbinati a **new.studente** che insegnano il corso abbinato a **new.docente**
 - Il sistema dovrà segnalare una violazione di integrità se tale numero risultasse > 1

Gestione schemi non trasformabili in BC-NF

docenza(docente, corso);

studente(studente, docente);

- Dato lo studente **S** ed il docente **D** individuare quanti risultano essere i docenti del corso tenuto da **D** per lo studente **S**

Gestione schemi non trasformabili in BC-NF

```
create schema bcnf;  
create table bcnf.Docenza(docente char(20) primary key, corso char(20));  
create table bcnf.Studente(studente char(20),  
                             docente char(20) references bcnf.Docenza,  
                             primary key(studente, docente));
```

```
insert into bcnf.Docenza values ('Modica', 'analisi I'), ('Giaquinta', 'analisi I');
```

Questa select va inserita in un trigger constraint che segnali una eccezione ogni volta che il risultato è > 1

```
select count(docente)  
from bcnf.studente NATURAL JOIN bcnf.docenza  
where studente = new.studente and corso = (  
    select corso from bcnf.docenza  
    where docente = new.docente )
```

Gestione schemi non trasformabili in BC-NF

```
create or replace function bcnf.check_consistency() returns trigger as $$  
begin  
  if ( select count(docente) from bcnf.studente NATURAL JOIN bcnf.docenza  
        where studente = new.studente and corso = (  
          select corso from bcnf.docenza  
          where docente = new.docente ) ) > 1  
  then raise exception 'Docenti multipli per lo stesso corso!!!';  
  end if;  
  return null;  
end;  
$$ language plpgsql;
```

```
create constraint trigger trigger_check_consistency  
after insert or update on bcnf.studente deferrable initially deferred  
for each row execute procedure bcnf.check_consistency();
```

Constraint triggers are expected to raise an exception when the constraints they implement are violated. Constraint triggers must be **AFTER ROW** triggers. They can be fired either at the end of the statement causing the triggering event, or at the end of the containing transaction (deferred).

Gestione schemi non trasformabili in BC-NF

```
create or replace function bcnf.check_consistency() returns trigger as $$
begin
    if ( select count(docente) from bcnf.studente
          where (studente=new.studente) and docente in (select d2.docente
                                                         from bcnf.docenza d1, bcnf.docenza d2
                                                         where d1.docente=new.docente and d1.corso=d2.corso) ) > 1
    then raise exception 'Docenti multipli per lo stesso corso!!!';
    end if;
    return null;
end;
$$ language plpgsql;
```

```
create constraint trigger trigger_check_consistency
after insert or update on bcnf.studente deferrable initially deferred
for each row execute procedure bcnf.check_consistency();
```

```
insert into bcnf.studente values ('Pala','Modica');
```

>> Query returned successfully: one row affected, 31 ms execution time.

```
insert into bcnf.studente values ('Pala','Giaquinta');
```

Esercizio

- Lo schema sotto riportato è impiegato per la gestione di un laboratorio
Computer(codice, marca, modello, fornitore)
Installazione(computer, software, descrSoftware, dataInstall)
- Tenuto conto che:
 - Esistono più computer della stessa marca e modello
 - I computer della stessa marca hanno lo stesso fornitore
 - Un software può essere installato su più computer
- Evidenziare le dipendenze funzionali presenti nello schema, verificare a quale forma normale sia conforme ciascuna delle due relazioni e trasformare lo schema almeno in terza forma normale

Esercizio

- Lo schema sotto riportato è impiegato per la gestione di un circo Zoo(codiceEsemplare, genere, gabbia, codAddetto, nomeAddetto, giornoPulizia, giornoSpettacolo, oraSpettacolo)
- Tenuto conto che:
 - Ogni gabbia è pulita da un solo addetto
 - Ogni gabbia è pulita sempre nello stesso giorno
 - Un addetto pulisce più gabbie
 - In ogni gabbia possono esserci più esemplari
 - Ad ogni spettacolo partecipano diversi esemplari
 - C'è un solo spettacolo al giorno
- Evidenziare le dipendenze funzionali presenti nello schema, verificare a quale forma normale sia conforme lo schema e trasformare lo schema almeno in terza forma normale

Esercizi

- Determinare la forma normale dei seguenti schemi di relazione ed eventualmente trasformare lo schema in modo che sia almeno in III-NF:
 - $R(\underline{A}, \underline{B}, C, D) \{B \rightarrow D\}$
 - $R(\underline{A}, \underline{B}, C, D) \{C \rightarrow A, C \rightarrow D\}$
 - $R(\underline{A}, \underline{B}, C, D) \{A \rightarrow C, B \rightarrow D\}$