La Normalizzazione

Materiale adattato dal libro Albano et al e
dal libro Atzeni-et al., Basi di dati

Parte I

TEORIA RELAZIONALE: INTRODUZIONE

- Due metodi per produrre uno schema relazionale:
  a) Partire da un buon schema a oggetti e tradurlo
  b) Partire da uno schema relazionale fatto da altri e modificarlo o completarlo
- Teoria della progettazione relazionale: studia cosa sono le "anomalie" e come eliminarle (normalizzazione).
- È particolarmente utile se si usa il metodo (b). È moderatamente utile anche quando si usa il metodo (a).

UNA TABELLA

N Inv	Stanza	Resp	Oggetto	Produttore	Descrizione
1012	256	Ghelli	Mac Mini	Apple	Personal Comp
1015	312	Albano	Dell XPS M1330	Dell	Notebook 2 GHZ
1034	256	Ghelli	Dell XPS M1330	Dell	Notebook 2GB
1112	288	Leoni	Mac Mini 2	Apple	Personal Comp

È fatta male? Perché? Come si può correggere?

SCHEMI CON ANOMALIE

- Esempio:
  - StudentiEdEsami(Matricola, Nome, Provincia,AnnoNascita, Materia, Voto)
- Anomalie:
  - Ridondanze
  - Potenziali inconsistenze
  - Anomalie nelle inserzioni
  - Anomalie nelle eliminazioni

SCHEMI CON ANOMALIE

- Esempio:
  - StudentiEdEsami(Matricola, Nome, Provincia,AnnoNascita, Materia, Voto)
- Anomalie:
  - Ridondanze
  - Potenziali inconsistenze
  - Anomalie nelle inserzioni/eliminazioni

Va bene?

- Soluzione: dividiamo lo schema in due tabelle.
  - Studenti (Matricola, **Nome**, Provincia, AnnoNascita)
  - Esami (**Nome**, Materia, Voto)

SCHEMI CON ANOMALIE

- Esempio:
  - StudentiEdEsami(Matricola, Nome, Provincia,AnnoNascita, Materia, Voto)
- Anomalie:
  - Ridondanze
  - Potenziali inconsistenze
  - Anomalie nelle inserzioni/eliminazioni
- Soluzione: dividiamo lo schema in due tabelle.
  - Studenti (**Matricola**, Nome, Provincia, AnnoNascita)
  - Esami (**Matricola**, Materia, Voto)

Va bene?

SCHEMI CON ANOMALIE

- Esempio:
  - StudentiEdEsami(Matricola, Nome, Provincia,AnnoNascita, Materia, Voto)
  - Quali sono le relazioni fra i diversi campi?

OBIETTIVI

- Nozione base: dipendenze funzionali
- Obiettivi della teoria:
  - **Equivalenza** di schemi: in che misura si può dire che uno schema rappresenta un altro
  - **Qualità** degli schemi (forme normali)
  - **Trasformazione** degli schemi (normalizzazione di schemi)
- Ipotesi dello **schema di relazione universale**:
  - Tutti i fatti sono descritti da attributi di un'unica relazione (**relazione universale**), cioè gli attributi hanno un significato globale.
- Definizione: Lo schema di relazione universale U di una base di dati relazionale ha come attributi l'unione degli attributi di tutte le relazioni della base di dati.

Obiettivi: Forme normali

- Una forma normale è una proprietà di una base di dati relazionale che ne garantisce la "qualità", cioè l'assenza di determinati difetti
- Quando una relazione non è normalizzata:
  - presenta ridondanze,
  - si presta a comportamenti poco desiderabili durante gli aggiornamenti
- La normalizzazione è una procedura che permette di trasformare schemi non normalizzati in schemi che soddisfano una forma normale

Perché questi fenomeni indesiderabili? - Parte I

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Ridondanza

Lo stipendio di ciascun impiegato è ripetuto in tutte le ennuple relative.

Questo perché lo stipendio dipende solo dall'Impiegato.

Il costo del bilancio per ogni progetto è ripetuto.

Anomalia di aggiornamento

Se lo stipendio di un impiegato varia, è necessario andarne a modificare il valore in diverse ennuple

Perché questi fenomeni indesiderabili? - Parte II

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Posso consentire valori null per il progetto?

Anomalia di cancellazione

Se un impiegato interrompe la partecipazione a tutti i progetti, dobbiamo cancellare tutte le ennuple in cui appare, e in questo modo l'impiegato non sarà più presente nel database

Anomalia di inserimento

Un nuovo impiegato non può essere inserito finché non gli viene assegnato un progetto

Linee Guida per una corretta progettazione – Parte I

Semantica degli attributi

- Si progetti ogni schema relazionale in modo tale che sia semplice spiegarne il significato. Non si uniscano attributi provenienti da più tipi di classi e tipi di associazione in una unica relazione.

Ridondanza

- Si progettino gli schemi relazionale in modo che nelle relazioni non siano presenti anomalie di inserimento, cancellazione o modifica. Se sono presenti delle anomalie (che si vuole mantenere), le si rilevi chiaramente e ci si assicuri che i programmi che aggiornano la base di dati operino correttamente

Linee Guida per una corretta progettazione – Parte II

Valori Nulli

- Per quanto possibile, si eviti di porre in relazione di base attributi i cui valori possono essere (frequentemente) nulli. Se è inevitabile, ci si assicuri che essi si presentino solo in casi eccezionali e che non riguardino una maggioranza di tuple nella relazione

Tuple spurie

- Si progettino schemi di relazione in modo tale che essi possano essere riuniti tramite JOIN con condizioni di uguaglianza su attributi che sono o chiavi primarie o chiavi esterne in modo da garantire che non vengano generate tuple spurie. Non si abbiano relazioni che contengono attributi di «accoppiamento» diversi dalle combinazioni chiave esterna-chiave primaria.

DIPENDENZE FUNZIONALI

- Per formalizzare la nozione di schema senza anomalie, occorre una descrizione formale della semantica dei fatti rappresentati in uno schema relazionale.

Confronto con il committente!!!

- **Istanza valida di R: è una nozione semantica**, che dipende da ciò che sappiamo del dominio del discorso (non estensionale, non deducibile da alcune istanze dello schema)

- Nozione fondamentale: **dipendenza funzionale**

Dipendenza funzionale (informale)

- Istanza valida r su R(T)
- Siano X e Y due sottoinsiemi non vuoti di T
- esiste in r una dipendenza funzionale da X a Y se, per ogni coppia di ennuple t_1 e t_2 di r con gli stessi valori su X, risulta che t_1 e t_2 hanno gli stessi valori anche su Y
- La dipendenza funzionale da X a Y si denota con $X \rightarrow Y$

Esempio:

Persone (CodiceFiscale, Cognome, Nome, DataNascita)

CodiceFiscale $\rightarrow$ Cognome, Nome

Dipendenza funzionale vs chiave – Parte I

- Istanza valida r su R(T) - Siano X e Y due sottoinsiemi non vuoti di T
- esiste in r una dipendenza funzionale da X a Y ($X \rightarrow Y$) se, per ogni coppia di ennuple t_1 e t_2 di r con gli stessi valori su X, risulta che t_1 e t_2 hanno gli stessi valori anche su Y

Esempio:

StudentiEdEsami(Matricola, Nome, Provincia,AnnoNascita, Materia, Voto)

Matricola → Nome, Provincia, AnnoNascita

Matricola è chiave?

DIPENDENZE FUNZIONALI (formalmente)

- Dato uno schema R(T) e X, Y $\subseteq$ T, una **dipendenza funzionale** (DF) fra gli attributi X e Y, è un vincolo su R sulle istanze della relazione, espresso nella forma

$$X \rightarrow Y,$$

i.e. X determina funzionalmente Y o Y è determinato da X, se per ogni istanza valida r di R un valore di X determina in modo univoco un valore di Y:

$\forall$ r istanza valida di R.

$\forall$ t1, t2 $\in$ r. se t1[X] = t2[X] allora t1[Y] = t2[Y]

- In altre parole: un'istanza r di R(T) soddisfa la dipendenza X $\rightarrow$ Y, (o X $\rightarrow$ Y vale in r), se per ogni coppia di ennuple t1 e t2 di r, se t1[X] = t2[X] allora t1[Y] = t2[Y]

Esempio: dipendenze funzionale

Esiste la DF
Dipartimento→Indirizzo?

∀ r istanza valida di R.

∀ t1, t2∈r. se t1[X] = t2[X] allora t1[Y] = t2[Y]

CodCorso	Titolo	CFU	Anno	Semestre	Codice Docente	Dipartimento	Indirizzo
1	Basi di Dati	6	2022	I	A1	Informatica	Via Roma
2	Basi di Dati	6	2023	II	A4	Informatica	Via Roma
3	Algebra	12	2022	I	A1	Informatica	Via Roma
4	Algebra	12	2023	I	A4	Informatica	Via Roma
5	Basi di Dati	6	2021	I	A1	Informatica	Via Roma
6	Basi di Dati	6	2020	I	A1	Informatica	Via Roma
7	Algebra	12	2023	II	A4	Matematica	Via Bianchi

Esiste la DF
titolo →Semestre?

Esiste la DF
Titolo,Anno →Semestre?

Esempio

- $\forall$ r istanza valida di R.

  $\forall$ t1, t2 $\in$ r. se t1[X] = t2[X] allora t1[Y] = t2[Y]

- Questa tabella soddisfa la dipendenza funzionale

Matricola → Cognome

Matricola	Cognome
1	Rossi
2	Verdi
3	Rossi
4	Viola

DIPENDENZE FUNZIONALI vs Chiave – Esempio

- $\forall$ r istanza valida di R.
  $\forall$ t1, t2 $\in$ r. se t1[X] = t2[X] allora t1[Y] = t2[Y]

Esempio:

StudentiEdEsami(Matricola, Nome, Provincia,AnnoNascita, Materia, Voto)

- Matricola è una chiave?
- Materia è una chiave?
- Matricola → Nome, Provincia, AnnoNascita
- Esempio:

Studenti (Matricola, Nome, Provincia, AnnoNascita)

DIPENDENZE FUNZIONALI

- Dato uno schema R(T) e X, Y $\subseteq$ T, una dipendenza funzionale (DF) è un vincolo su R del tipo $X \rightarrow Y$,

□ $\forall$ r istanza valida di R.

$\forall$ t1, t2 $\in$ r. se t1[X] = t2[X] allora t1[Y] = t2[Y]

Si dice che

- un'**istanza r_0 di R *soddisfa* la DF $X \rightarrow Y$** (r_0 |= $X \rightarrow Y$) se
  - la proprietà vale per r_0: $\forall$ t1, t2 $\in$ r. se t1[X] = t2[X] allora t1[Y] = t2[Y]
- e che un'istanza r_0 di R *soddisfa* un insieme F di DF
  - se per ogni $X \rightarrow Y \in F$, vale r_0 |= $X \rightarrow Y$:

    r_0 |= $X \rightarrow Y$ sse $\forall$ t1, t2 $\in$ r_0. se t1[X] = t2[X] allora t1[Y] = t2[Y]

Esempio

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Abbiamo usato un'unica relazione per rappresentare informazioni eterogenee

- gli impiegati con i relativi stipendi (Impiegato → Stipendio)
- i progetti con i relativi bilanci (Progetto → Bilancio)

Anomale?

- le partecipazioni degli impiegati ai progetti con le relative funzioni (Impiegato Progetto → Funzione).

Anomala?

UNA TABELLA

N Inv	Stanza	Resp	Oggetto	Produttore	Descrizione
1012	256	Ghelli	Mac Mini	Apple	Personal Comp
1015	312	Albano	Dell XPS M1330	Dell	Notebook 2 GHZ
1034	256	Ghelli	Dell XPS M1330	Dell	Notebook 2GB
1112	288	Leoni	Mac Mini 2	Apple	Personal Comp

Che domande
potremmo fare
al committente?

ESEMPIO – Parte I

- DotazioniLibri(CodiceLibro, NomeNegozio, IndNegozio, Titolo, Quantità)

- DF:

  { CodiceLibro → Titolo

  NomeNegozio → IndNegozio

  CodiceLibro, NomeNegozio → IndNegozio, Titolo, Quantità }

Dipendenze funzionali atomiche – Asimmetria – Esempio Parte II

Ogni dipendenza funzionale $X \rightarrow A_1 A_2 \ldots A_n$, si può scomporre nelle dipendenze funzionali $X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_n$

Le dipendenze funzionali del tipo $X \rightarrow A$ si chiamano dipendenze funzionali atomiche.

- DotazioniLibri(CodiceLibro, NomeNegozio, IndNegozio, Titolo, Quantità)

  CodiceLibro, NomeNegozio → IndNegozio, Titolo, Quantità

  ⇓

  CodiceLibro, NomeNegozio → IndNegozio,

  CodiceLibro, NomeNegozio → Titolo

  CodiceLibro, NomeNegozio → Quantità

Esempio Parte III - Dipendenza funzionali e ridondanza

DotazioniLibri(CodiceLibro, NomeNegozio, IndNegozio, Titolo, Quantità)

- DF:

{ CodiceLibro $\rightarrow$ Titolo

NomeNegozio $\rightarrow$ IndNegozio

CodiceLibro, NomeNegozio $\rightarrow$ IndNegozio, Titolo, Quantità }

$\Downarrow$

{ CodiceLibro $\rightarrow$ Titolo

NomeNegozio $\rightarrow$ IndNegozio

CodiceLibro, NomeNegozio $\rightarrow$ Quantità }

Altro esempio - Parte I

Articoli (Kit, Componente, Tipo, QuantComp, PrezzoComp, Fornitore, PrezzoTot)

Kit	Componente	Tipo	QuantComp	PrezzoComp	Fornitore	PrezzoTot
Libreria	Legno	Noce	50	10	A	4400
Libreria	Bulloni	Acciaio	200	1	B	4400
Libreria	Vetro	Cristallo	3	50	C	4400
Scaffale	Legno	Mogano	37	15	A	555
PC	Bulloni	Acciaio	25	1	B	700
PC	Tastiera	A3000	3	30	D	700
PC	Mouse	B2000	5	45	D	700
Scrivania	Legno	Noce	10	8	B	500
Scrivania	Maniglie	Rame	10	24	B	500
Tavolo	Legno	Noce	4	10	A	600
	…		…			…

PrezzoTot è il Prezzo di vendita
Assumiamo che:
Il tipo si riferisca ad una sola componente

Chiave:
Kit, Tipo

Sono chiavi:
- Quantcomp, PrezzoComp
- Tipo, PrezzoTot

???

Quali sono le dipendenze funzionali?

Altro esempio - Parte II

Assumiamo che:
Il tipo si riferisca ad una sola componente

Ridondanze:
- PrezzoTot è ripetuto in ogni tupla che si riferisce allo stesso kit
- PrezzoComp è ripetuto in ogni tupla che ha lo stesso valore di Tipo e Fornitore
- Componente è ripetuto in ogni tupla che ha lo stesso Tipo

Kit	Componente	Tipo	QuantComp	PrezzoComp	Fornitore	PrezzoTot
Libreria	Legno	Noce	50	10	A	4400
Libreria	Bulloni	Acciaio	200	1	B	4400
Libreria	Vetro	Cristallo	3	50	C	4400
Scaffale	Legno	Mogano	37	15	A	555
PC	Bulloni	Acciaio	25	1	B	700
PC	Tastiera	A3000	3	30	D	700
PC	Mouse	B2000	5	45	D	700
Scrivania	Legno	Noce	10	8	B	500
Scrivania	Maniglie	Rame	10	24	B	500
Tavolo	Legno	Noce	4	10	A	600
	…		…			…

Quali sono le dipendenze funzionali?

- Tipo → Componente
- Kit → PrezzoTot
- Kit,Tipo → PrezzoComponente, QuantComp, Fornitore

Altro esempio – Parte III - Decomposizione

- Una decomposizione della relazione che non presenti ridondaze e senza perdita di informazione

- Dipendenze funzionali
  - Tipo → Componente
  - Kit → PrezzoTot
  - Kit,Tipo → PrezzoComponente, QuantComp, Fornitore

Kit	Componente	Tipo	QuantComp	PrezzoComp	Fornitore	PrezzoTot

Kit	Tipo	QuantComp	Fornitore

Tipo	PrezzoComp	Fornitore

Kit	PrezzoTot

Tipo	Componente

Altro esempio - Dipendenza funzionali banali

<u>Impiegato</u>	Stipendio	<u>Progetto</u>	Bilancio	Funzione

La dipendenza funzionale del tipo

Impiegato Progetto $\rightarrow$ Progetto

è sempre valida, per cui si tratta di una DF "banale"

X $\rightarrow$ *A* è non banale se *A* non è contenuta in X

Siamo interessati alle dipendenze funzionali non banali.

ESPRIMERE LE DIPENDENZE FUNZIONALI

- Consideriamo: NomeNegozio → IndNegozio
- Espressione diretta ($P \Rightarrow Q$):
  - Se in due righe il NomeNegozio è uguale, anche l'IndNegozio è uguale:
    - $\text{NomeNegozio}_{=} \Rightarrow \text{IndNegozio}_{=}$
- Per contrapposizione ($\neg Q \Rightarrow \neg P$):
  - Se l'IndNegozio è diverso allora il NomeNegozio è diverso:
    - $\text{IndNegozio}_{\neq} \Rightarrow \text{NomeNegozio}_{\neq}$
- Per assurdo:
  - Non possono esserci due nuple in DotazioniLibri con NomeNegozio uguale e IndNegozio diverso:
    - Not $(\text{NomeNegozio}_{=} \wedge \text{IndNegozio}_{\neq})$
    - $\text{NomeNegozio}_{=} \wedge \text{IndNegozio}_{\neq} \Rightarrow \text{False}$

MANIPOLAZIONE DI CLAUSOLE

- Sono <u>equivalenti</u>:
  - $NomeNegozio_{=} \Rightarrow IndNegozio_{=}$
  - $IndNegozio_{\neq} \Rightarrow NomeNegozio_{\neq}$
  - $NomeNegozio_{=} \wedge IndNegozio_{\neq} \Rightarrow False$

Importante!

- In generale:
  - $A \Rightarrow B \quad \Leftrightarrow \quad A \wedge \neg B \Rightarrow False \quad \Leftrightarrow \quad \neg B \Rightarrow \neg A$
- Più in generale, in ogni clausola $A \wedge B \Rightarrow E \vee F$ posso spostare le sottoformule da un lato all'altro, negandole
- Quindi sono equivalenti:
  - $NomeNegozio_{=} \wedge CodiceLibro_{=} \Rightarrow Quantità_{=}$
  - $NomeNegozio_{=} \wedge CodiceLibro_{=} \wedge Quantità_{\neq} \Rightarrow False$
  - $CodiceLibro_{=} \wedge Quantità_{\neq} \Rightarrow NomeNegozio_{\neq}$
  - $NomeNegozio_{=} \wedge Quantità_{\neq} \Rightarrow CodiceLibro_{\neq}$

ESEMPIO CdL

- Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine)
- In un dato momento, un docente si trova al più in un'aula
- Non è possibile che due docenti diversi siano nella stessa aula contemporaneamente
- Se due lezioni si svolgono su due piani diversi appartengono a due corsi di laurea *diversi*
- Se due lezioni *diverse* si svolgono lo stesso giorno per la stessa materia, appartengono a due CDL diversi (lezioni diverse: not($CodAula_{=} \wedge$ and $NomeAula_{=} \wedge$ …))

ESEMPIO CdL – Prima domanda

- Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine)
- In un dato momento, un docente si trova al più in un'aula.
- Due domande:
  - Che vuole dire momento?
  - Che tipo di implicazione vi aspettate?

Soluzione ESEMPIO CdL - 1

- Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine)
- In un dato momento, un docente si trova al più in un'aula.

$$\text{NOT}\ ([\text{Giorno, OraInizio}]_= \land \text{Docente}_= \Rightarrow \text{Aula}_{\neq}) \Rightarrow \text{FALSE}$$

$$\text{NOT}\ ([\text{Giorno, OraFine}]_= \land \text{Docente}_= \Rightarrow \text{Aula}_{\neq}) \Rightarrow \text{FALSE}$$

$$A \land B \land C_{\neq} \Rightarrow \text{FALSE}$$

$$\Downarrow$$

$$A \land B \Rightarrow C_=$$

$$\Downarrow$$

$$[\text{Giorno, OraInizio}]_= \land \text{Docente}_= \Rightarrow \text{Aula}_=$$

$$[\text{Giorno, OraFine}]_= \land \text{Docente}_= \Rightarrow \text{Aula}_=$$

Soluzione ESEMPIO CdL - 2

- Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine)
- Non è possibile che due docenti diversi siano nella stessa aula contemporaneamente

Non posso avere:

$[\text{Giorno, OraInizio}]_= \wedge \text{Aula}_= \wedge \text{Docente}_{\neq} \Rightarrow \text{FALSE}$

$[\text{Giorno, OraFine}]_= \wedge \text{Aula}_= \wedge \text{Docente}_{\neq} \Rightarrow \text{FALSE}$

$\Downarrow$

$[\text{Giorno, OraInizio}]_= \wedge \text{Aula}_= \Rightarrow \text{Docente}_=$

$[\text{Giorno, OraFine}]_= \wedge \text{Aula}_= \Rightarrow \text{Docente}_=$

Soluzione ESEMPIO CdL - 3

- Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine)
- Se due lezioni si svolgono su due piani diversi appartengono a due corsi di laurea diversi

- $Piano_{\neq} \Rightarrow CdL_{\neq}$
- Che equivale a
- $CdL_{=} \Rightarrow Piano_{=}$

Soluzione ESEMPIO CdL - 4

- Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine)
- Se due lezioni *diverse* si svolgono lo stesso giorno per la stessa materia, appartengono a due CDL diversi (lezioni diverse: not($CodAula_= \wedge$ and $NomeAula_= \wedge$ …))
- $Materia_= \wedge Giorno_= \Rightarrow CdL_{\neq}$???
- $Lezioni_{\neq} \wedge Materia_= \wedge Giorno_= \Rightarrow CdL_{\neq}$ cioè

  $Materia_= \wedge Giorno_= \wedge CdL_= \Rightarrow Lezioni_=$

Lezioni non è un attributo

- $Materia_= \wedge Giorno_= \wedge CdL_= \Rightarrow$ CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, OraInizio, OraFine
- $Materia_= \wedge Giorno_= \wedge CdL_= \Rightarrow$ CodAula, NomeAula, Piano, Posti, Docente, OraInizio, OraFine

Riepilogo

- Qualità degli schemi relazionali
- Anomalie
- Dipendenze funzionali $X \rightarrow Y$
  - $r \models X \rightarrow Y$ se $\forall$ t1, t2 $\in$ r. se t1[X] = t2[X] allora t1[Y] = t2[Y]
  - r soddisfa un insieme F di DF ($r \models F$)
  - $F \models X \rightarrow Y$

Attenzione: una dipendenza funzionale è un vincolo.

Non vi possono essere gradi di libertà!

PARTE II

DIPENDENZE FUNZIONALI

- Notazione:
  - **R <T, F>** denota uno <u>schema</u> con attributi T e dipendenze funzionali F.
- *Le DF sono una proprietà semantica, cioè dipendono dai fatti rappresentati e non da come gli attributi sono combinati in schemi di relazione.*
- Si parla di DF **completa** quando $X \rightarrow Y$ e per ogni $W \subset X$, non vale $W \rightarrow Y$.
- Se X è una superchiave, allora X determina ogni altro attributo della relazione: $X \rightarrow T$
- Se X è una chiave, allora $X \rightarrow T$ è una DF completa

PROPRIETÀ DELLE DF

Da un insieme F di DF, in generale altre DF sono 'implicate' da F.

Esempio:

Matricola $\rightarrow$ CodFisc

CodFisc $\rightarrow$ Cognome

Matricola $\rightarrow$ Cognome

Implicazione logica

- Dipendenze implicate (definizione):

  Sia F un insieme di DF sullo schema R, diremo che

  F implica logicamente X $\rightarrow$ Y (F |= X $\rightarrow$ Y,),

  se ogni istanza r di R che soddisfa F soddisfa anche X $\rightarrow$ Y.

- Dipendenze banali:

  implicate dal vuoto, es. {} |= X -> X

ESEMPIO

- Sia r un'istanza di R<T, F>, con $F = \{X \rightarrow Y, X \rightarrow Z\}$ e $X, Y, Z \subseteq T$.

  Sia $X' \subseteq X$. Altre DF sono soddisfatte da r,

- ad es.
  - $X \rightarrow X'$ (DF *banale*) e
  - $X \rightarrow YZ$, infatti

    $t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$

    $t_1[X] = t_2[X] \Rightarrow t_1[Z] = t_2[Z]$

    $t_1[X] = t_2[X] \Rightarrow t_1[YZ] = t_2[YZ]$

  Pertanto $\{X \rightarrow Y, X \rightarrow Z\} \models X \rightarrow YZ$

- Altro esempio: $\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$
- $\models$ denota l'implicazione logica

REGOLE DI INFERENZA

- Come derivare DF implicate logicamente da F?
  - usando un insieme di regole di inferenza.

- "Assiomi" (sono in realtà regole di inferenza) di Armstrong:
  - Se $Y \subseteq X$, allora $X \rightarrow Y$ (Riflessività R)
  - Se $X \rightarrow Y$, $Z \subseteq T$, allora $XZ \rightarrow YZ$ (Arricchimento A)
  - Se $X \rightarrow Y$, $Y \rightarrow Z$, allora $X \rightarrow Z$ (Transitività T)

DERIVAZIONE

Definizione

Sia F un insieme di DF, diremo che $X \rightarrow Y$ *sia derivabile da F* (F |- $X \rightarrow Y$),

se $X \rightarrow Y$ può essere inferito da F usando gli assiomi di Armstrong.

- Si dimostra che valgono anche le regole:
  - $\{X \rightarrow Y, X \rightarrow Z\}$ |- $X \rightarrow YZ$ (unione **U**)
  - $Z \subseteq Y$ $\{X \rightarrow Y\}$ |- $X \rightarrow Z$ (decomposizione **D**)
- L'unione: $\{X \rightarrow Y, X \rightarrow Z\}$ |- $X \rightarrow YZ$ (unione **U**)

1.	$X \rightarrow Y$	(per ipotesi)
2.	$X \rightarrow XY$	(per arricchimento da 1)
3.	$X \rightarrow Z$	(per ipotesi)
4.	$XY \rightarrow YZ$	(per arricchimento da 3)
5.	$X \rightarrow YZ$	(per transitività da 2, 4)

Se $Y \subseteq X$, allora $X \rightarrow Y$ (Riflessività R)

Se $X \rightarrow Y$, $Z \subseteq T$, allora $XZ \rightarrow YZ$ (Arricchimento A)

Se $X \rightarrow Y$, $Y \rightarrow Z$, allora $X \rightarrow Z$ (Transitività T)

DERIVAZIONE

Definizione

- Sia F un insieme di DF, diremo che $X \rightarrow Y$ *sia derivabile* da F (F |- $X \rightarrow Y$), se $X \rightarrow Y$ può essere inferito da F usando gli assiomi di Armstrong.
- Una derivazione di f da F è una sequenza finita $f_1, \ldots, f_m$ di dipendenze, dove $f_m = f$ e ogni f_i è un elemento di F oppure è ottenuta dalle precedenti dipendenze $f_1, \ldots, f_{i-1}$ della derivazione usando una regola di inferenza.

Si dimostra che valgono anche le regole:

- $\{X \rightarrow Y, X \rightarrow Z\}$ |- $X \rightarrow YZ$ (unione **U**)
- $Z \subseteq Y$ $\{X \rightarrow Y\}$ |- $X \rightarrow Z$ (decomposizione **D**)

- Da **Unione** e **Decomposizione** si ricava che se $Y = A_1A_2 \ldots A_n$ allora
  - $X \rightarrow Y \Leftrightarrow \{X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_n\}$

ESEMPIO

- R(A B C D)
- $F = \{A \to B, BC \to D\}$
- AC è una superchiave? Ovvero $AC \to ABCD$?

1. $A \to B$	ipotesi 1
2. $AC \to BC$	da 1 per **Arr** (C)
3. $BC \to D$	ipotesi 2
4. $BC \to BCD$	da 3 per **Arr** (BC)
5. $AC \to BCD$	da 2+4 per **Trans**
6. $AC \to ABCD$	da 5 per **Arr** (A)

Se $Y \subseteq X$, allora $X \to Y$ (Riflessività R)

Se $X \to Y$, $Z \subseteq T$, allora $XZ \to YZ$ (Arricchimento Arr)

Se $X \to Y$, $Y \to Z$, allora $X \to Z$ (Transitività Trans)

CORRETTEZZA E COMPLETEZZA DEGLI ASSIOMI DI ARMSTRONG

- **Teorema:** Gli assiomi di Armstrong sono corretti e completi.
- Attraverso gli assiomi di Armstrong, si può mostrare l'equivalenza della nozione di implicazione logica (|=) e di quella di derivazione (|-): se una dipendenza è derivabile con gli assiomi di Armstrong allora è anche implicata logicamente (correttezza degli assiomi), e viceversa se una dipendenza è implicata logicamente allora è anche derivabile dagli assiomi (completezza degli assiomi).
- Correttezza degli assiomi:

$$\forall f, \qquad F \models f \;\;\Leftarrow\;\; F \vdash f$$

- Completezza degli assiomi:

$$\forall f, \qquad F \models f \;\;\Rightarrow\;\; F \vdash f$$

CHIUSURA DI UN INSIEME F

- **Definizione** Dato un insieme F di DF, la chiusura di F, denotata con F^+, è:

  $F^+ = \{ X \rightarrow Y \mid F \mid\!- X \rightarrow Y\}$

- Un problema che si presenta spesso è quello di decidere se una dipendenza funzionale appartiene a F^+ (problema dell'implicazione);
  - la sua risoluzione con l'algoritmo banale (di generare F^+ applicando ad F ripetutamente gli assiomi di Armstrong) ha una complessità esponenziale rispetto al numero di attributi dello schema.

CHIUSURA DI UN INSIEME F

- **Definizione** Dato un insieme F di DF, la chiusura di F, denotata con F^+, è:

  $$F^+ = \{ X \rightarrow Y \mid F \vdash X \rightarrow Y\}$$

- **Definizione** Dato R<T, F>, e $X \subseteq T$, la *chiusura* di X rispetto ad F, denotata con X_F^+, (o X^+, se F è chiaro dal contesto) è

  $$X_F^+ = \{A_i \in T \mid F \vdash X \rightarrow A_i\}.$$

CHIUSURA DI UN INSIEME F

- **Definizione** Dato un insieme F di DF, la chiusura di F, denotata con F^+, è:

  $$F^+ = \{ X \rightarrow Y \mid F \mid\!- X \rightarrow Y\}$$

- **Definizione** Dato R<T, F>, e $X \subseteq T$, la *chiusura* di X rispetto ad F, denotata con X_F^+, (o X^+, se F è chiaro dal contesto) è

  $$X_F^+ = \{A_i \in T \mid F \mid\!- X \rightarrow A_i\}.$$

- **Problema dell'implicazione**: controllare se una DF $V \rightarrow W \in F^+$

  Un algoritmo efficiente per risolvere il problema dell'implicazione senza calcolare la chiusura di F scaturisce dal seguente teorema.

  Teorema/osservazione $F \mid\!- X \rightarrow Y \Leftrightarrow Y \subseteq X_F^+$.

Algoritmo per calcolare X_F^+ - Idea

Sia X un insieme di attributi e F un insieme di dipendenze. Vogliamo calcolare X_F^+

1. Inizializziamo X^+ con l'insieme X
2. Se fra le dipendenze di F c'è una dipendenza $Y \rightarrow A$ con $Y \subseteq X^+$ allora si inserisce A in X^+, ossia $X^+ = X^+ \cup \{A\}$
3. Si ripete il passo 2 fino a quando non ci sono altri attributi da aggiungere a X^+
4. Si dà in output $X_F^+ = X^+$

CHIUSURA LENTA

- Un semplice algoritmo per calcolare X^+ (ne esiste uno migliore di complessità di tempo $O(ap)$) è
- **Algoritmo** CHIUSURA LENTA

1. Si dà in output $X_F^+ = X^+$

input $R\langle T, F\rangle$ $X \subseteq T$

output X^+

begin

$X^+ = X$ Inizializziamo X^+ con l'insieme X

while (X^+ cambia) **do**

fino a quando non ci sono altri attributi da aggiungere a X^+

for $W \rightarrow V$ **in** F **with** $W \subseteq X^+$ **and** $V \vee X^+$

do $X^+ = X^+ \cup V$

Se fra le dipendenze di F c'è una dipendenza $W \rightarrow V$ con $W \subseteq X^+$ allora si inserisce V in X^+, ossia $X^+ = X^+ \cup \{V\}$

end

Terminazione dell'algoritmo

L'algoritmo termina perché ad ogni passo viene aggiunto un nuovo attributo a X^+. Essendo gli attributi in numero finito, a un certo punto l'algoritmo deve fermarsi

Per dimostrare la correttezza, si dimostra che $X_F^+ = X^+$

(per induzione)

ESEMPIO

- $F = \{DB \rightarrow E, B \rightarrow C, A \rightarrow B\}$, trovare $(AD)^+$:

- Vogliamo conoscere gli attributi che sono determinati funzionalmente da un insieme di dipendenze A e D.

$X^+ = AD$

$X^+ = ADB$

$X^+ = ADBE$

$X^+ = ADBEC$

Se fra le dipendenze di F c'è una dipendenza $Y \rightarrow A$ con $Y \subseteq X^+$ allora si inserisce A in X^+, ossia $X^+ = X^+ \cup \{A\}$

CHIAVI E ATTRIBUTI PRIMI

- **Definizione** Dato lo schema R<T, F>, diremo che un insieme di attributi $W \subseteq T$ è una chiave candidata di R, se
  - $W \rightarrow T \in F^+$ (W superchiave)
  - $\forall\ V \subset W,\ V \rightarrow T \notin F^+$ (se $V \subset W$, V non superchiave)
- *Attributo primo* : attributo che appartiene ad almeno una chiave

ESEMPIO – superchiave?

- $F = \{DB \rightarrow E,\ B \rightarrow C,\ A \rightarrow B\}$, trovare $(AD)^+$:

  $X^+ = AD$

  $X^+ = ADB$

  $X^+ = ADBE$

  $X^+ = ADBEC$

- AD è superchiave?
  - Si poiché contiene tutti gli attributi
- A è superchiave?
  - $A \rightarrow B$, $A \rightarrow BC$, si ferma $\rightarrow$ non è superchiave
- ABD è superchiave?
  - $(ABD)^+$ è analoga a $(AD)^+$, perché ABD è più grande di AD, quindi è superchiave
- ABC è superchiave?
  - ABC stesso, quindi non è superchiave

E per verificare se è una chiave?

Esempio: chiave?

<u>Impiegato</u>	Stipendio	<u>Progetto</u>	Bilancio	Funzione

F={Impiegato → Stipendio, Progetto → Bilancio,

Impiegato Progetto → Funzione}

$\{Impiegato\}^+ = \{Impiegato, Stipendio\}$

$\{Progetto\}^+ = \{Progetto, Bilancio\}$

$\{Impiegato, Progetto\}^+ = \{Impiegato, Progetto, Stipendio, Bilancio, Funzione\}$

K=(Impiegato Progetto) è chiave. Infatti genera tutto l'insieme U e nessuno dei suoi sottoinsiemi di attributi lo genera

CHIAVI E ATTRIBUTI PRIMI

- **Definizione** Dato lo schema $R<T, F>$, diremo che un insieme di attributi $W \subseteq T$ è una chiave candidata di R, se
  - $W \rightarrow T \in F^+$ (W superchiave)
  - $\forall\, V \subset W,\ V \rightarrow T \notin F^+$ (se $V \subset W$, V non superchiave)
- *Attributo primo* : attributo che appartiene ad almeno una chiave

- Complessità
  - Il problema di trovare tutte le chiavi di una relazione richiede un algoritmo di complessità esponenziale nel caso peggiore
  - Il problema di controllare se un attributo è primo è NP-completo

Proprietà interessanti per trovare tutte le chiavi

L'algoritmo per trovare tutte le chiavi si basa su due proprietà:

1. Se un attributo A di T **non** appare a destra di alcuna dipendenza in F, allora A **appartiene** ad ogni chiave di R
2. Se un attributo A di T **appare a destra** di qualche dipendenza in F, ma **non appare a sinistra** di alcuna dipendenza non banale, allora **A non appartiene ad alcuna chiave**.

Domande:

- Posso sfruttare queste due proprietà per trovare una chiave?
- Quale è il punto di partenza?

TROVARE <u>TUTTE</u> LE CHIAVI di R

- Sia F = $\{C \rightarrow D, CF \rightarrow B, D \rightarrow C, F \rightarrow E\}$
- Ogni chiave deve contenere AF; le chiavi sono in $AF \cdot P(BCDE) = AF^{BCDE}$ (nel testo: AF::(BCDE))
- $AF^+ = AFE$; ogni chiave in $AF^{BCD} - \{AF\}$
- Candidati: $AF^{BCD} - \{AF\} = AFB^{CD} + AFC^{D} + AFD$

AF		$AF^+ = AFE$	BCDE-AFE = BCD
B		$AFB^+ = AFBE$	CD-AFBE = CD
	~~C~~	no: AFC chiave	
	~~D~~	no: AFD chiave	
C		$AFC^+ = AFCDBE$	**AFC chiave**
D		$AFD^+ = AFDCEB$	**AFD chiave**

PARTE III

Copertura Canonica

COPERTURA DI INSIEMI DI DF

- **Definizione:** Due insiemi di DF, F e G, sullo schema R sono equivalenti,

$$F \equiv G, \text{ sse } F^+ = G^+.$$

- Se $F \equiv G$, allora F è una copertura di G (e G una copertura di F).

Esempio:

studenti(matricola, CF, Cognome, Nome, Anno)

COPERTURA DI INSIEMI DI DF – Parte I – attributo estraneo

- **Definizione** Sia F un insieme di DF:

  1. Data una $X \rightarrow Y \in F$, si dice che X contiene un attributo *estraneo* A_i sse $(X - \{A_i\}) \rightarrow Y \in F^+$, cioè $F \mid\!- (X - \{A_i\}) \rightarrow Y$

(Come facciamo a stabilire che in una DF del tipo $AX \rightarrow B$ l'attributo A è estraneo? Per verificare se A è estraneo calcoliamo X^+ e verifichiamo se include B, ovvero se basta X a determinare B)

- Esempio:

  Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, Ora)

  - se vale
    - Docente, Giorno, Ora -> CodAula
    - Docente, Giorno -> Ora
  - allora
    - Docente, Giorno -> CodAula
    - (quindi) nella prima dipendenza Ora è attributo estraneo

Attributo estraneo - Esempio

$$F = \{AB \rightarrow C, A \rightarrow B\}$$

- In $AB \rightarrow C$, l'attributo B è estraneo?
- Calcoliamo A^+ e verifichiamo se include C, ovvero se basta X a determinare C

$A^+ = A$

$A^+ = AB$ poiché $A \rightarrow B$ e $A \subseteq A^+$

$A^+ = ABC$ poiché $AB \rightarrow C$ e $AB \subseteq A^+$

C dipende solo da A, ovvero in $AB \rightarrow C$ l'attributo B è estraneo (perché a sua volta dipendente da A: $A \rightarrow B$)

A^+ include l'attributo C

L'insieme di DF può essere riscritto come: $F' = \{A \rightarrow C, A \rightarrow B\}$

Nota che

$(AB)^+ = AB$

$(AB)^+ = ABC$ poiché $AB \rightarrow C$ e $AB \subseteq A^+$

COPERTURA DI INSIEMI DI DF – Parte II - ridondante

- **Definizione** Sia F un insieme di DF:

  2. $X \rightarrow Y$ è una dipendenza *ridondante* sse $(F - \{X \rightarrow Y\})^+ = F^+$,

     Equivalentemente $F - \{X \rightarrow Y\} \vdash X \rightarrow Y$

(Come facciamo a stabilire che una DF del tipo $X \rightarrow A$ è ridondante? La eliminiamo da F, calcoliamo X^+ e verifichiamo se include *A*, ovvero se con le DF che restano riusciamo ancora a dimostrare che X determina *A*)

- Esempio:

  Orari(CodAula, NomeAula, Piano, Posti, Materia, CDL, Docente, Giorno, Ora)

  - se vale
    - Docente, Giorno, Ora -> CodAula
    - CodAula -> NomeAula
  - è inutile avere anche
    - Docente, Giorno, Ora -> NomeAula

DF ridondanti - Esempio

- $F = \{B \rightarrow C, B \rightarrow A, C \rightarrow A\}$

- $B \rightarrow A$ è ridondante
- Poiché possiamo dedurla da $B \rightarrow C$ e $C \rightarrow A$

COPERTURA DI INSIEMI DI DF – Parte III

- **Definizione** Sia F un insieme di DF:

1. Data una $X \rightarrow Y \in F$, si dice che X contiene un *attributo estraneo* A_i sse $(X - \{A_i\}) \rightarrow Y \in F^+$, cioè $F \mid\!- (X - \{A_i\}) \rightarrow Y$

2. $X \rightarrow Y$ è una *dipendenza ridondante* sse

   $(F - \{X \rightarrow Y\})^+ = F^+$,

   Equivalentemente $F - \{X \rightarrow Y\} \mid\!- X \rightarrow Y$

- Altro Esempio: studenti(matricola, CF, Cognome, Nome, Anno)

  Matricola → cognome

  Matricola, **cognome** → nome (cognome è estraneo)

  è equivalente a

  Matricola → cognome

  Matricola → nome

Esempio

$$F_1=\{A \to B,\ AB \to C,\ A \to C\}$$

$$F_2=\{A \to B,\ AB \to C\}$$

$$F_3=\{A \to B,\ A \to C\}$$

- In F_1 vi è ridondanza? Presenta attributi estranei?
- In F_2 vi è ridondanza? Presenta attributi estranei?
- In F_3 vi è ridondanza? Presenta attributi estranei?

Esempio - Soluzione

$$F_3=\{A \rightarrow B,\ A \rightarrow C\}$$

- F_3 non presenta attributi estranei

$$F_2=\{A \rightarrow B,\ AB \rightarrow C\}$$

- F_2 non è ridondante ma presenta un attributo estraneo, perché B può essere eliminato dal primo membro della seconda dipendenza ($A^+=A$; $A^+=AB$; $A^+=ABC$) (quindi equivale a F_3)

$$F_1=\{A \rightarrow B,\ AB \rightarrow C,\ A \rightarrow C\}$$

- F_1 è ridondante, perché $\{A \rightarrow B,\ AB \rightarrow C\}$ implica $A \rightarrow C$ (quindi equivale a F_2)

COPERTURA DI INSIEMI DI DF – Parte IV

- **Definizione** Sia F un insieme di DF:

1. Data una $X \rightarrow Y \in F$, si dice che X contiene un attributo *estraneo* A_i sse $(X - \{A_i\}) \rightarrow Y \in F^+$, cioè $F \mid\!- (X - \{A_i\}) \rightarrow Y$
2. $X \rightarrow Y$ è una dipendenza *ridondante* sse

   $(F - \{X \rightarrow Y\})^+ = F^+$,

   Equivalentemente

$$F - \{X \rightarrow Y\} \mid\!- X \rightarrow Y$$

F è detta una *copertura canonica* sse

- la parte destra di ogni DF in F è un attributo;
- non esistono attributi estranei;
- nessuna dipendenza in F è ridondante.

ESISTENZA DELLA COPERTURA CANONICA

- *Teorema*

  Per ogni insieme di dipendenze F esiste una copertura canonica.

- Algoritmo per calcolare una copertura canonica:
  - Trasformare le dipendenze nella forma $X \rightarrow A$
  - Eliminare gli attributi estranei
  - Eliminare le dipendenze ridondanti

Si sostituisce l'insieme dato con quello equivalente che ha tutti i secondi membri costituiti da singoli attributi (dipendenze atomiche)

Per ogni dipendenza si verifica se esistono attributi eliminabili dal primo membro.
Data una $X \rightarrow Y \in F$, si dice che X contiene un attributo *estraneo* A_i sse $(X - \{A_i\}) \rightarrow Y \in F^+$, cioè $F \mid\!- (X - \{A_i\}) \rightarrow Y$

$X \rightarrow Y$ è una dipendenza *ridondante* sse $(F - \{X \rightarrow Y\})^+ = F^+$, Equivalentemente $F - \{X \rightarrow Y\} \mid\!- X \rightarrow Y$

Esempio ESISTENZA DELLA COPERTURA CANONICA – Parte I

Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

Consideriamo il seguente insieme di dipendenze funzionali

$\{M \rightarrow RSDG, MS \rightarrow CD, G \rightarrow R, D \rightarrow S, S \rightarrow D, MPD \rightarrow AM\}$

Esempio ESISTENZA DELLA COPERTURA CANONICA – Parte II

Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

$\{M \rightarrow RSDG, MS \rightarrow CD, G \rightarrow R, D \rightarrow S, S \rightarrow D, MPD \rightarrow AM\}$

1. Creiamo le dipendenze funzionali atomiche

$\{M \rightarrow R, M \rightarrow S, M \rightarrow D, M \rightarrow G, MS \rightarrow C, MS \rightarrow D, G \rightarrow R, D \rightarrow S, S \rightarrow D, MPD \rightarrow A, MPD \rightarrow M\}$

2. Eliminare gli attributi estranei:

- è possibile eliminare S dal primo membro di $M\cancel{S} \rightarrow C$ e $M\cancel{S} \rightarrow D$ perché $M \rightarrow S$ (si ottiene da $M \rightarrow D$ e $D \rightarrow S$)
- È possibile eliminare D dal primo membro di $MP\cancel{D} \rightarrow A$ e $MP\cancel{D} \rightarrow M$ poiché $M \rightarrow D$

$\{M \rightarrow R, M \rightarrow S, M \rightarrow D, M \rightarrow G, M \rightarrow C, M \rightarrow D, G \rightarrow R, D \rightarrow S, S \rightarrow D, MP \rightarrow A, MP \rightarrow M\}$

Esempio ESISTENZA DELLA COPERTURA CANONICA – Parte III

Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

$\{M \to R, M \to S, M \to D, M \to G, MS \to C, MS \to D, G \to R, D \to S, S \to D, MPD \to A, MPD \to M\}$

Eliminazione degli attributi estranei

$\{M \to R, M \to S, M \to D, M \to G, M \to C, M \to D, G \to R, D \to S, S \to D, MP \to A, MP \to M\}$

3. <u>Si trova l'insieme di dipendenza funzionali non ridondante</u>: eliminiamo le dipendenze ottenibili da altre:

$M \to R$ (deriva da $M \to G$ e $G \to R$)

$M \to S$ (deriva da $M \to D$ e $D \to S$)

$M \to D$ (Perché già $M \to D$)

$MP \to M$ (Perché M compare a primo membro)

Esempio ESISTENZA DELLA COPERTURA CANONICA – Parte IV

Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

$\{M \rightarrow RSDG, MS \rightarrow CD, G \rightarrow R, D \rightarrow S, S \rightarrow D, MPD \rightarrow AM\}$

Eliminazione degli attributi estranei

$\{M \rightarrow R, M \rightarrow S, M \rightarrow D, M \rightarrow G, MS \rightarrow C, MS \rightarrow D, G \rightarrow R,$
$D \rightarrow S, S \rightarrow D, MPD \rightarrow A, MPD \rightarrow M\}$

Eliminazione di: $M \rightarrow R$, $M \rightarrow S$, $M \rightarrow D$, $MP \rightarrow M$

$\{M \rightarrow R, M \rightarrow S, M \rightarrow D, M \rightarrow G, M \rightarrow C, M \rightarrow D, G \rightarrow R,$
$D \rightarrow S, S \rightarrow D, MP \rightarrow A, MP \rightarrow M\}$

$\{M \rightarrow D, M \rightarrow G, M \rightarrow C, G \rightarrow R, D \rightarrow S, S \rightarrow D, MP \rightarrow A\}$

Riassunto

- Qualità degli schemi relazionali e anomalie
- Dipendenze funzionali $X \rightarrow Y$

  $r \models X \rightarrow Y$ se $\forall$ t1, t2 $\in$ r.

  se t1[X] = t2[X] allora t1[Y] = t2[Y]

- Chiusura di un insieme di dipendenze

  $F^+ = \{ X \rightarrow Y \mid F \vdash X \rightarrow Y \}$

- Chiave e attributi primi
- Copertura canonica (attributi estranei, dipendenze ridondanti)

Trovare una qualsiasi chiave

- $T = \{A, B, C, D, E, F\}$
- $F = \{C \rightarrow D, CF \rightarrow B, D \rightarrow C, F \rightarrow E\}$

Quale proprietà possiamo usare?

Attributi estranei?

Dipendenza funzionale banale?

Proprietà sugli attributi e dipendenze funzionali?

RIEPILOGO

Normalizzazione dei dati

- Le **ridondanze** sui dati possono essere di due tipi:

> **Ridondanza concettuale** → non ci sono duplicazioni dello stesso dato, ma sono memorizzate informazioni che possono essere ricavate da altre già contenute nel DB.

> **Ridondanza logica** → esistono duplicazioni sui dati, che possono generare anomalie nelle operazioni sui dati …

Normalizzazione dei dati

- Le dipendenze funzionali sono definite **a livello di schema** e non a livello di istanza!

Matricola	Cognome	Corso	Voto
1244	Rossi	Basi di Dati	18
1567	Bianchi	Programmazione	22
1898	Bianchi	Analisi I	20
2040	Verdi	Programmazione	22
2121	Verdi	Basi di Dati	18
2678	Bruni	Analisi I	20

- Dipendenza funzionale **Corso → Voto**? NO!

Normalizzazione dei dati

- Le dipendenze funzionali hanno sempre **un verso**!

<u>Matricola</u>	Studente	<u>Corso</u>	Docente	Voto
1244	Rossi	Basi di Dati	Ghelli	18
1567	Bianchi	Programmazione	Messina	22
1898	Bianchi	Analisi I	Palermo	20
2040	Verdi	Programmazione	Messina	22
2121	Verdi	Basi di Dati	Ghelli	18
2678	Bruni	Analisi I	Palermo	20

- Dipendenza funzionale **Corso → Docente** ? Può essere, occorre considerare le specifiche del sistema …

Le dipendenze funzionali hanno sempre **un verso**!

Matricola	Studente	Corso	Docente	Voto
1244	Rossi	Basi di Dati	Roma	18
1567	Bianchi	Programmazione	Messina	22
1898	Bianchi	Analisi I	Palermo	20
2040	Verdi	Programmazione	Messina	22
2121	Verdi	Basi di Dati	Roma	18
2678	Bruni	Analisi I	Palermo	20
4354	Bruni	Architetture	Roma	28

- Corso → Docente? **OK**   Docente → Corso? **NO**!

Normalizzazione dei dati

- Le dipendenze funzionali sono una **generalizzazione del vincolo di chiave (e di <u>superchiave</u>)**.

- Data una tabella con schema R(X), con superchiave K.
  - Esiste un vincolo di dipendenza funzionale tra K e qualsiasi attributo dello schema r.

$$K \to X_1, \quad X_1 \subseteq X$$

Riepilogo: ESISTENZA DELLA COPERTURA CANONICA

- *Teorema*

  Per ogni insieme di dipendenze F esiste una copertura canonica.

- Algoritmo per calcolare una copertura canonica:
  - Trasformare le dipendenze nella forma $X \rightarrow A$
  - Eliminare gli attributi estranei
  - Eliminare le dipendenze ridondanti

Si sostituisce l'insieme dato con quello equivalente che ha tutti i secondi membri costituiti da singoli attributi (dipendenze atomiche)

Per ogni dipendenza si verifica se esistono attributi eliminabili dal primo membro.
Data una $X \rightarrow Y \in F$, si dice che X contiene un attributo *estraneo* A_i sse $(X - \{A_i\}) \rightarrow Y \in F^+$, cioè $F \mid\!- (X - \{A_i\}) \rightarrow Y$

$X \rightarrow Y$ è una dipendenza *ridondante* sse $(F - \{X \rightarrow Y\})^+ = F^+$,
Equivalentemente $F - \{X \rightarrow Y\} \mid\!- X \rightarrow Y$

Decomposizione di Schemi

IN GENERALE, PER ELIMINARE ANOMALIE DA UNO SCHEMA OCCORRE DECOMPORLO IN SCHEMI PIÙ PICCOLI "EQUIVALENTI"

Esempio di decomposizione

L'intuizione è che si devono "estrarre" gli attributi che sono determinati da attributi non chiave ovvero "creare uno schema per ogni funzione"

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20	Marte	2	tecnico
Verdi	35	Giove	15	progettista
Verdi	35	Venere	15	progettista
Neri	55	Venere	15	direttore
Neri	55	Giove	15	consulente
Neri	55	Marte	2	consulente
Mori	48	Marte	2	direttore
Mori	48	Venere	15	progettista
Bianchi	48	Venere	15	progettista
Bianchi	48	Giove	15	direttore

Impiegato → Stipendio

Impiegato	Stipendio
Rossi	20
Verdi	35
Neri	55
Mori	48
Bianchi	48

Progetto → Bilancio

Progetto	Bilancio
Marte	2
Giove	15
Venere	15

Impiegato Progetto → Funzione

Impiegato	Progetto	Funzione
Rossi	Marte	tecnico
Verdi	Giove	progettista
Verdi	Venere	progettista
Neri	Venere	direttore
Neri	Giove	consulente
Neri	Marte	consulente
Mori	Marte	direttore
Mori	Venere	progettista
Bianchi	Venere	progettista
Bianchi	Giove	direttore

Non sempre è così facile - Esempio

La soluzione non è tuttavia sempre così semplice, bisogna fare anche altre considerazioni; ad esempio, operando come prima:

<u>Impiegato</u>	<u>Progetto</u>	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

Ammette le due dipendenze funzionali

Impiegato → Sede

Progetto → Sede

Decomponiamo sulla base delle dipendenze

Impiegato	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

Impiegato → Sede
Progetto → Sede

Ci sono anomalie?

Impiegato → Sede

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

Progetto → Sede

Progetto	Sede
Marte	Roma
Giove	Milano
Saturno	Milano
Venere	Milano

Proviamo a ricostruire (mediante join)

T_1

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

$\bowtie$

Progetto	Sede
Marte	Roma
Giove	Milano
Saturno	Milano
Venere	Milano

T_2

Decomposizione con **perdite**:
si generaro delle tuple spurie dopo il join.

T_1 join T_2

Impiegato	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Saturno	Milano
Neri	Venere	Milano
Verdi	Saturno	Milano
Neri	Giove	Milano

Queste due tuple sono in più (spurie)

Impiegato	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

Diversa dalla relazione di partenza!

DECOMPOSIZIONE DI SCHEMI

- **Definizione** Dato uno schema R(T),
  $\rho = \{R_1(T_1), \ldots, R_k(T_k)\}$ è una decomposizione di R sse $T_1 \cup \ldots \cup T_k = T$:

- StudentiEdEsami(Matricola, Nome, Provincia, AnnoNascita, Materia, Voto)

$$\Downarrow$$

- {Studenti(Matricola, Nome, Provincia,AnnoNascita),
  Esami(Matricola, Materia, Voto)}
- {Studenti(Matricola, Nome, Provincia,AnnoNascita),
  Esami(Nome, Materia, Voto)}
- {Studenti(Matricola, Nome, Provincia,AnnoNascita),
  Esami(Materia, Voto)}

DECOMPOSIZIONE DI SCHEMI: Conservazione dei dati e delle dipendenze

- In generale, per eliminare anomalie da uno schema occorre decomporlo in schemi più piccoli "equivalenti"

- **Definizione** Dato uno schema $R(T)$,
  $\rho = \{R_1(T_1), \ldots, R_k(T_k)\}$ è una decomposizione di R *sse* $\cup T_i = T$:

- Due proprietà desiderabili di una decomposizione:
  - conservazione dei dati (*nozione semantica*)
  - conservazione delle dipendenze

DECOMPOSIZIONE DI SCHEMI – Conservazione dei dati

- Decomposizioni che preservano i dati:
- *Definizione*: $\rho = \{R_1(T_1), \ldots, R_k(T_k)\}$ è una decomposizione di uno schema R(T) che preserva i dati sse per ogni *istanza valida* r di R:

$$r = (\pi_{T1}\, r) \vee (\pi_{T2}\, r) \vee \ldots \vee (\pi_{Tk}\, r)$$

- Dalla definizione di *giunzione naturale* scaturisce il seguente risultato:
- *Teorema*: Se $\rho = \{R_1(T_1), \ldots, R_k(T_k)\}$ è una decomposizione di R(T), allora per ogni istanza r di R:

$$r \subseteq (\pi_{T1}\, r) \vee (\pi_{T2}\, r) \vee \ldots \vee (\pi_{Tk}\, r)$$

DECOMPOSIZIONE DI SCHEMI – Con Perdita di informazione

StudentiEdEsami(Matricola, Nome, Provincia, AnnoNascita, Materia, Voto)

⇩

{Studenti(Matricola, Nome, Provincia,AnnoNascita),

Esami(Nome, Materia, Voto)}

Cosa succede quando si fa la giunzione?

Nessuna tupla si perde, ma…?

Questa decomposizione crea tuple spurie: ci sono n-uple in più.

Si pensi al caso di due persone con lo stesso nome che hanno sostenuto esami diversi, cosa succede dopo la giunzione?

<u>Perdita di informazione</u>!

DECOMPOSIZIONE DI SCHEMI – Con Perdita di informazione

StudentiEdEsami(Matricola, Nome, Provincia, AnnoNascita, Materia, Voto)

⇓

{Studenti(Matricola, Nome, Provincia,AnnoNascita),

Esami(Materia, Voto)}

Si perde l'informazione sullo studente.

Perdita di informazione!

DECOMPOSIZIONE DI SCHEMI – Senza Perdita di informazione

StudentiEdEsami(Matricola, Nome, Provincia, AnnoNascita, Materia, Voto)

⇓

{Studenti(Matricola, Nome, Provincia, AnnoNascita),
Esami(Matricola, Materia, Voto)}

Perché non perdo informazione con questa decomposizione?

La chiave è l'unico modo per avere una decomposizione senza perdita di informazione.

ESEMPIO DI DECOMPOSIZIONE – non preserva i dati

- Sia r qui sotto un'istanza valida di R(ABC):

r=

A	B	C
a1	b	c1
a2	b	c2

Allora la decomposizione {R(AB), R(BC)}:

π_{T1} r =

A	B
a1	b
a2	b

π_{T2} r =

B	C
b	c1
b	c2

non preserva i dati, infatti $\pi_{T1}\, r \vee \pi_{T2}\, r \supseteq r$

r=

A	B	C
a1	b	c1
a1	b	c2
a2	b	c1
a2	b	c2

Decomposizione senza perdita

Uno schema R(X) si **decompone senza Perdita dei dati** negli schemi R1(X1) ed R2(X2) se, per ogni possibile istanza r di R(X), il join naturale delle proiezioni di r su X1 ed X2 produce la tabella di partenza. (cioè non contiene *ennuple spurie*)

$$\rho_{X1}(r) \bowtie \rho_{X2}(r) = r$$

Decomposizione senza perdita

$$\rho_{X1}(r) \bowtie \rho_{X2}(r) = r$$

La decomposizione senza perdita è garantita se l'insieme degli attributi comuni alle due relazioni ($X_1 \cap X_2$) è chiave per almeno una delle due relazioni decomposte.

Ad esempio, Sede=(Progetto, Sede)∩(Impiegato, Sede) non è chiave per nessuna delle due relazioni

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

Progetto	Sede
Marte	Roma
Giove	Milano
Saturno	Milano
Venere	Milano

(decomposizione con perdita)

Senza perdita

Sia r una relazione su un insieme di attributi X e siano X_1 e X_2 due sottoinsiemi di X la cui unione sia pari a X stesso;

Inoltre, sia X_0 l'intersezione di X_1 e X_2, allora:

- r si decompone senza perdita su X_1 e X_2 se soddisfa la dipendenza funzionale $X_0 \rightarrow X_1$ oppure la dipendenza funzionale $X_0 \rightarrow X_2$

Motivazione

Teorema (non formale):

Se l'insieme degli attributi comuni alle due relazioni ($X_1 \cap X_2$) è chiave per almeno una delle due relazioni decomposte allora la decomposizione è senza perdita

Dimostrazione (non formale)

- Supponiamo r sia una relazione sugli attributi ABC e consideriamo le sue proiezioni r_1 su AB e r_2 su AC.
- Supponiamo che r soddisfi la dipendenza funzionale $A \rightarrow C$. Allora A è chiave per r su AC e quindi non ci sono in tale proiezione due tuple diverse sugli stessi valori di A.

Motivazione

Il join costruisce tuple a partire dalle tuple nelle due proiezioni

- Sia t=(a,b,c) una tupla del join di r_1 e r_2.
- Mostriamo che appartiene ad r (cioè non è spuria).
  - t è ottenuta mediante join da t_1=(a,b) di r_1 e t_2=(a,c) su r_2.
  - Allora per la definizione di proiezione, esistono due tuple in r, t'_1= (a,b,*) e t'_2= (a,*,c) (dove * sta per un valore non noto).
  - Poiché $A \rightarrow C$, allora esiste un solo valore in C associato al valore a. Dato che (a,c) compare nella proiezione, questo valore è proprio c.
  - Ma allora nella tupla t'_1 il valore incognito deve essere proprio c. Quindi (a,b,c) appartiene a r.

DECOMPOSIZIONI BINARIE

- **Teorema**

- Sia R<T, F> uno schema di relazione, la decomposizione

  $\rho = \{R_1(T_1), R_2(T_2)\}$ preserva i dati sse

  - $T_1 \cap T_2 \rightarrow T_1 \in F^+$ oppure $T_1 \cap T_2 \rightarrow T_2 \in F^+$.

- Esistono criteri anche per decomposizioni in più di due schemi.

Decomposizione senza perdita

- Anche se una decomposizione è senza perdite, può comunque presentare dei **problemi di conservazione delle dipendenze** …
- Ad esempio, Impiegato=(Impiegato, Sede)∩(Impiegato, Progetto)

Impiegato	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Venere	Milano
Neri	Saturno	Milano

IMPIEGATO→ SEDE

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

Impiegato	Progetto
Rossi	Marte
Verdi	Giove
Verdi	Venere
Neri	Venere
Neri	Saturno

- Con questa decomposizione, non ho tuple spurie …

Proviamo a decomporre senza perdita

Impiegato	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

Impiegato → Sede

$\boxed{\cancel{\text{Progetto} \rightarrow \text{Sede}}}$

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

Impiegato	Progetto
Rossi	Marte
Verdi	Giove
Verdi	Venere
Neri	Saturno
Neri	Venere

In questa decomposizione: trascuriamo la seconda dipendenza funzionale

Un altro problema – Parte I

- Supponiamo di voler inserire una nuova ennupla che specifica la partecipazione dell'impiegato Neri (che opera a Milano) al progetto Marte (lo schema non lo impedisce)

Insert into ImpProg value (Neri, Marte)

Qualcosa di «strano»?

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

ImpProg

Impiegato	Progetto
Rossi	Marte
Verdi	Giove
Verdi	Venere
Neri	Saturno
Neri	Venere
Neri	Marte

Impiegato → Sede

Progetto → Sede

Domanda: Che accade se aggiungo l'impiegato Neri al progetto Marte?

Un altro problema - Parte II

IMPIEGATO→ SEDE

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

$\bowtie$

Impiegato	Progetto
Rossi	Marte
Verdi	Giove
Verdi	Venere
Neri	Venere
Neri	Saturno
Neri	Marte

Viene violata la seconda dipendenza funzionale (che per il momento avevamo tenuto in sospeso)

Progetto → Sede

Una decomposizione conserva le dipendenze se ciascuna delle dipendenze funzionali dello schema originario coinvolge attributi che compaiono tutti insieme in uno degli schemi decomposti

=

Impiegato	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Venere	Milano
Neri	Saturno	Milano
Neri	Venere	Milano
Neri	Marte	Milano

Nell'esempio considerato
Progetto → Sede
non è conservata

Esempio di query di verifica

- Se una DF non si preserva diventa più complicato capire quali sono le modifiche del DB che non violano la FD stessa
- In generale si devono prima eseguire query SQL di verifica (o, meglio, fare uso di trigger)
- Bisogna verificare che il progetto (Marte) sia presso la stessa sede dell'impiegato (Neri). A tal fine bisogna trovare un impiegato che lavora al progetto Marte

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

Impiegato	Progetto
Rossi	Marte
Verdi	Giove
Verdi	Venere
Neri	Venere
Neri	Saturno
Neri	Marte

```
SELECT * -- OK se restituisce una tupla
FROM Impiegati I
WHERE
    I.Impiegato = 'Neri' AND I.Sede IN (
        SELECT I1.Sede
        FROM Impiegati I1, ImpProg IP
        WHERE I1.Impiegato = IP.Impiegato
            AND IP.Progetto = `Marte')
```

Qualità delle decomposizioni

Una decomposizione:

- deve essere senza perdita, per garantire la ricostruzione delle informazioni originarie
- dovrebbe preservare le dipendenze, per semplificare il mantenimento dei vincoli di integrità originari

Nell'esempio, questo suggerisce di inserire anche:

- Va sempre eseguita una query, ma più semplice

Progetti

Progetto	Sede
Marte	Roma
Giove	Milano
Venere	Milano
Saturno	Milano

Impiegato	Sede
Rossi	Roma
Verdi	Milano
Neri	Milano

Impiegato	Progetto
Rossi	Marte
Verdi	Giove
Verdi	Venere
Neri	Venere
Neri	Saturno
Neri	Marte

```
SELECT * -- OK se restituisce una tupla
FROM Impiegati I, Progetti P
WHERE
        I.Impiegato = 'Neri' AND
        P.Progetto = `Marte' AND
        I.Sede = P.Sede
```

Una decomposizione conserva le dipendenze se ciascuna delle dipendenze funzionali dello schema originario coinvolge attributi che compaiono tutti insieme in uno degli schemi decomposti

Nell'esempio considerato
Progetto → Sede
non è conservata

ESEMPIO – Preservare le dipendenze?

- Telefoni(Prefisso, Numero, Località, Abbonato, Via)

{Pref Num $\rightarrow$ Loc Abb Via, Loc $\rightarrow$ Pref} cioè:

(Prefisso, Numero) è chiave

{P N $\rightarrow$ L A V, L $\rightarrow$ P}

- Si consideri la decomposizione:

  ρ = {Tel <{N, L, A, V}, F1 >,

  Pref<{L, P}, F2 >} con

  F1 = {LN $\rightarrow$ A V} e F2 ={L $\rightarrow$ P}

Numero	Località	Abbonato	Via
5348	Padova	Gino	Pascoli
5348	Vigonza	Ignazio	Silone
2344	Venezia	Lino	Leopardi
2122	Padova	Ciro	Pavese

Prefisso	Località
49	Padova
49	Vigonza
41	Venezia

- Preserva dati ma non le dipendenze: PN $\rightarrow$ L non è deducibile da F1 e F2.
- Esistono istanze valide della decomposizione che non sono proiezione di una istanza valida della relazione originale

PROIEZIONE DELLE DIPENDENZE

- **Definizione**
- Dato lo schema R<T, F>, e $T_1 \subseteq T$, la **proiezione di F su T_1** è

$$\pi_{T1}(F) = \{X \to Y \in F^+ \mid X\,Y \subseteq T_1\}$$

- **Esempio**

Sia R(A, B, C) e F={$A \to B$, $B \to C$, $C \to A$}.

$\pi_{AB}(F) \equiv \{A \to B, B \to A\}$

$\pi_{AC}(F) \equiv \{A \to C, C \to A\}$

Potrebbe sembrare che la decomposizione ρ = {R1(A, B), R2(A,C)} non preservi le dipendenze perché B e C non appaiono insieme in uno schema della decomposizione, invece da $B \to A$ e $A \to C$ si ha $B \to C$

- Algoritmo banale per il calcolo di $\pi_{T1}(F)$:

for each $Y \subseteq T_1$ **do** ($Z := Y^+$; output $Y \to Z \cap T_1$)

PRESERVAZIONE DELLE DIPENDENZE

- **Definizione** Dato lo schema R<T, F>, la decomposizione $\rho = \{R_1, \ldots, R_n\}$ preserva le dipendenze sse l'unione delle dipendenze in $\pi_{T_i}(F)$ è una copertura di F.

- **Proposizione** Dato lo schema R<T, F>, il problema di stabilire se la decomposizione $\rho = \{R_1, \ldots, R_n\}$ preserva le dipendenze ha complessità di tempo polinomiale.

- Un **teorema importante**:

  Sia $\rho = \{R_i\langle T_i, F_i\rangle\}$ una decomposizione di R<T, F> che preserva le dipendenze e tale che T_j, per qualche j, è una superchiave per R<T, F>.

  Allora ρ preserva i dati.

ESEMPIO (ridondanza?)

- Telefoni(Prefisso, Numero, Località, Abbonato, Via)

$$F=\{P\ N \rightarrow L\ A\ V,\ L \rightarrow P\}$$

- Si consideri la decomposizione:

  ρ = {Tel<{N, L, A, V}, F1>, Pref<{L, P}, F2>} con

  - F1 = {LN $\rightarrow$ A V}
  - F2 = {L $\rightarrow$ P}

- Preserva dati ma non le dipendenze: PN $\rightarrow$ L non è deducibile da F1 e F2.

Localita	Numero	Abbonato	Via
Pisa	506070	Rossi	Via Roma
Calci	506070	Verdi	Lungarno

Localita	Prefisso
Pisa	050
Calci	050

Qualità delle decomposizioni

- Una decomposizione dovrebbe sempre soddisfare le seguenti proprietà:

  - la decomposizione senza perdita, che garantisce la ricostruzione delle informazioni originarie senza generazione di tuple spurie

  - la conservazione delle dipendenze, che garantisce il mantenimento dei vincoli di integrità (di dipendenza funzionale) originari

Forme normali

Forme normali

- Una forma normale è una proprietà di una base di dati relazionale che ne garantisce la "qualità", cioè l'assenza di determinati difetti

- Quando una relazione non è normalizzata:
  - presenta ridondanze,
  - si presta a comportamenti poco desiderabili durante gli aggiornamenti

FORME NORMALI

- **1FN**:
  - Impone una restrizione sul tipo di una relazione: ogni attributo ha un tipo elementare.

- **2FN**, **3FN** e **FNBC**:
  - Impongono restrizioni sulle dipendenze.
  - FNBC **(**Boyce-Codd**)** è la più naturale e la più restrittiva.

Forma normale di Boyce e Codd (BCNF)

Una relazione r è in forma normale di Boyce e Codd (BCNF) se, per ogni dipendenza funzionale (non banale) $X \rightarrow Y$ definita su di essa, X contiene una chiave K di r (è una superchiave)

La forma normale richiede che i concetti in una relazione siano omogenei (solo proprietà direttamente associate alla chiave)

La relazione sugli Impiegati

Impiegato	Stipendio	Progetto	Bilancio	Funzione

non è in forma normale di Boyce and Codd perché esiste la dipendenza funzionale

Impiegato → Stipendio   (Impiegato non è (super)chiave per la relazione)

FORME NORMALI di Boyce-Codd

- **FNBC**:
  - Intuizione: se esiste in R una dipendenza $X \rightarrow A$ non banale ed X non è chiave, allora X modella l'identità di un'entità diversa da quelle modellate dall'intera R

  - Ad esempio, in

    StudentiEdEsami(Matricola, Nome, Provincia, AnnoNascita, Materia, Voto)

    - Matricola -> Nome e Matricola non è (super)chiave.
      - il Nome dipende dalla Matricola che *non* è chiave.

FNBC

- **Definizione**
- $R<T, F>$ è in BCNF $\Leftrightarrow$

  per ogni $X \rightarrow A \in F^+$, con $A \notin X$ (non banale), X è una superchiave.

- **Teorema** $R<T, F>$ è in BCNF $\Leftrightarrow$ per ogni $X \rightarrow A \in F$ non banale,

  X è una superchiave.

- **Esempi**:
  - Docenti(CodiceFiscale, Nome, Dipartimento, Indirizzo) $\{CF \rightarrow N\ D;\ D \rightarrow I\}$
  - Impiegati(Codice, Qualifica, NomeFiglio) $\{C \rightarrow Q\}$

L'ALGORITMO DI ANALISI

- R<T,F> è decomposta in: $R_1(X, Y)$ e $R_2(X, Z)$ e su di esse si ripete il procedimento; esponenziale.
- input: R<T, F>, con F copertura canonica
- output: decomposizione in BCNF che preserva i dati

$\rho = \{R<T, F>\}$

while esiste in ρ una $R_i<T_i, F_i>$ non in BCNF per la DF $X \rightarrow A$

do

$T_a = X^+$

$F_a = \pi_{Ta}(F_i)$

$T_b = T_i - X^+ + X$ ← *attenzione: errore nel vecchio libro*

$F_b = \pi_{Tb}(F_i)$

$\rho = \rho - R_i + \{ R_a<T_a, F_a>, \; R_b< T_b, F_b > \}$

(R_a ed R_b sono nomi nuovi)

end

Osservazione

$T_a = X^+$

$T_b = T_i - X^+ + X$

Perché aggiungiamo X?

PROPRIETA' DELL'ALGORITMO

- Preserva i dati, ma non necessariamente le dipendenze
- Esempi di decomposizioni senza perdita di dipendenze:
  - Docenti(CodiceFiscale, Nome, Dipartimento, Indirizzo), {$CF \rightarrow N\ D$; $D \rightarrow I$}

$(CF)^+ = CF\ N\ D\ I$ è chiave

$(D)^+ = D\ I$ non è chiave

Decompongono

- R1(D,I); R2(CF,N, **D**)
- $F1 = \{ D \rightarrow I \}$ $F2 = \{ CF \rightarrow N\ D \}$

PROPRIETA' DELL'ALGORITMO

- Preserva i dati, ma non necessariamente le dipendenze
- Esempi di decomposizioni senza perdita di dipendenze:

  - Impiegati(Codice, Qualifica, NomeFiglio) $\{C \rightarrow Q\}$
    - R1(C, Q); R2(C, NF)
    - $F1 = \{ C \rightarrow Q \}$ $F2=\{ \}$

Dato che non perdo dipendenze funzionali, posso fare la proiezione approssimata su F

Esempio

- Telefoni (Prefisso, Numero, Località, Abbonato, Via)

$$F = \{ P\ N \rightarrow L\ A\ V, \quad L \rightarrow P \}$$

- Ad esempio:
  - Pisa → 050
  - Milano → 02

Preserva Dati e Dipendenze?

PROPRIETA' DELL'ALGORITMO (cont.)

- Decomposizione con perdita di dipendenze:

- Telefoni(Prefisso, Numero, Località, Abbonato, Via),

  {P N → L A V, L → P}

  - R1(L, P); R2(L, N, A, V)
  - Preserva dati ma non le dipendenze: PN → L non è deducibile da F1 e F2.

- Cosa vuole dire "non preserva le dipendenze"?
  - R1 = {(Pisa, 050); (Calci, 050)}
  - R2 = {(Pisa, 506070, Rossi, Piave),
    (Calci,506070, Bianchi, Isonzo)}

Posso inserire due numeri telefonici (senza prefisso) con comuni differenti che hanno lo stesso prefisso?

Esercizio I

Si consideri il seguente schema relazionale R< ABCDE, F= { $CE \rightarrow A$, $D \rightarrow E$, $CB \rightarrow E$, $CE \rightarrow B$ } > Applicare l'algoritmo di **analisi** e dire se dati e dipendenze sono stati preservati.

Esercizio II

Si consideri il seguente schema relazionale R< ABCDE, { AC→D, BD→A, BD→E, A→B } >. Applicare l'algoritmo di **analisi** e dire se dati e dipendenze sono stati preservati.

Esercizio I

Si consideri il seguente schema relazionale R< ABCDE, F= { CE→A, D→E, CB→E, CE→B } > Applicare l'algoritmo di **analisi** e dire se dati e dipendenze sono stati preservati.

Consideriamo CE→A. CE^+=CEAB (<u>CE non è chiave</u>), per cui decomponiamo:

R1(CEAB) (gli attributi di CE^+), R2(CED)

(In R2 tutti gli altri attribuiti (D) e la chiave esterna (CE))

Proiettiamo le dipendenze (<u>approssimate</u> su F):

R1 < CEAB, { CE → A, CB → E, CE → B } > (Proiezione in F)

R2 < CED, { D→E } > (Proiezione in F)

CE^+=CEAB e CB^+=CBEA, per cui R1 è in BCNF

D^+=DE, per cui R2 va ancora decomposta: R2 < CED, { D→E } > -> R3 , R4

La decomposizione è quindi: { R1(CBEA), R3(DE), R4(DC) }.

La decomposizione preserva dati e dipendenze ed è in questo caso è la stessa prodotta dall'algoritmo di sintesi (che vedremo dopo).

Esercizio II

Si consideri il seguente schema relazionale R< ABCDE, { $AC \to D$, $BD \to A$, $BD \to E$, $A \to B$ } >. Applicare l'algoritmo di **analisi** e dire se dati e dipendenze sono stati preservati.

Consideriamo $AC \to D$. AC^+=ACDBE (AC è chiave)

Consideriamo **BD**$\to$A. BD^+=BDAE (BD non è chiave), per cui decomponiamo:

R1(BDAE) (gli attributi di BD^+),

R2(BDC) (tutti gli altri attribuiti (C) e la chiave esterna BD)

Proiettiamo le dipendenze (approssimate su F):

R1 < BDAE, { $BD \to A$, $BD \to E$, $A \to B$ } > (Proiezione approx. in F)

R2 < BDC, { } > (Proiezione approx in F)

Esercizio II – Soluzione Parte II

Si consideri il seguente schema relazionale

$R\langle ABCDE, \{ AC \rightarrow D, BD \rightarrow A, BD \rightarrow E, A \rightarrow B \} \rangle$.

....

Proiettiamo le dipendenze (approssimate su F):

$R1 \langle BDAE, \{ BD \rightarrow A, BD \rightarrow E, A \rightarrow B \} \rangle$ (Proiezione in F)

$R2 \langle BDC, \{ \} \rangle$ (Proiezione in F)

La dipendenza $AC \rightarrow D$ si perde!

In R1: $BD^+=BDAE$ (BD è chiave), ma $A \rightarrow B$ viola la BCNF ($A^+=AB$), per cui proiettando su R1 ($BD \rightarrow A$, $BD \rightarrow E$ si perdono) abbiamo:

$R3 \langle AB, \{A \rightarrow B\}, R4(ADE, \{ \} \rangle$

La decomposizione è quindi { R2(BDC, { }), R3(AB, { A→B }), R4(ADE, { }) } che preserva i dati ma non le dipendenze.

Riepilogo

- Boyce-Codd Normal Form
  - Anomalia:
    - dipendenza $X \rightarrow A$ non banale, con X non superchiave,
    - X identità di un'entità con attributi X+ diversa da quelle modellate dall'intera R
    - Es:

StudentiEdEsami(Matricola, Nome, Prov, AnnoNascita, Materia, Voto)

    - con Matricola -> Nome, Prov, AnnoNascita

- Schema in BCNF = schema privo di anomalie

- Algoritmo di normalizzazione

Una relazione non normalizzata

Data una relazione non in FNBC, **è sempre possibile** ottenere una **decomposizione in FNBC**?

Dirigente	<u>Progetto</u>	<u>Sede</u>
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Marte	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

- Progetto Sede → Dirigente: ogni progetto ha più dirigenti che ne sono responsabili, ma in sedi diverse, e ogni dirigente può essere responsabile di più progetti; però per ogni sede, un progetto ha un solo responsabile
- Dirigente → Sede: ogni dirigente opera presso una sede
- Dirigente → Sede è una dipendenza funzionale ma Dirigente non è una (super)chiave. Quindi la relazione non è in BCNF.

La decomposizione è problematica

Dirigente	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Marte	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

Progetto Sede → Dirigente
Dirigente → Sede

- Progetto Sede → Dirigente coinvolge tutti gli attributi e quindi nessuna decomposizione può preservare tale dipendenza.
- quindi in alcuni casi la BCNF "non è raggiungibile"
- Occorre ricorrere a una forma normale indebolita

Algoritmi per la normalizzazione

- Quando si hanno diverse DF è difficile ragionare sullo schema, ed è quindi altrettanto difficile operare manualmente buone decomposizioni

- La terza forma normale (3NF) è un target di normalizzazione che consente di ottenere automaticamente:
  - decomposizioni senza perdita
  - decomposizioni che preservano tutte le dipendenze

- Una relazione r è in terza forma normale (3NF) se, per ogni FD (non banale) $X \rightarrow Y$ definita su r, è verificata almeno una delle seguenti condizioni:
  - X contiene una chiave K di r (come nella BCNF)
  - Oppure ogni attributo in Y è contenuto in almeno una chiave K di r

Una relazione in terza forma normale

Dirigente	Progetto	Sede
Rossi	Marte	Roma
Verdi	Giove	Milano
Verdi	Marte	Milano
Neri	Saturno	Milano
Neri	Venere	Milano

Progetto Sede → Dirigente

Dirigente → Sede

Nella prima dipendenza funzionale il primo membro della dipendenza (Progetto, Sede) è una chiave, nella seconda il secondo membro (Sede) è contenuto in una chiave. Quindi la relazione è in terza forma normale

(SVANTAGGI) La 3FN è meno restrittiva della FNBC

- **Tollera alcune ridondanze** ed anomalie sui dati.
  - Es. per ogni occorrenza di un dirigente viene ripetuta la sua sede
- **Certifica meno lo qualità** dello schema ottenuto.

(VANTAGGI) La 3FN è sempre ottenibile, qualsiasi sia lo schema di partenza.

- COME? **Algoritmo di normalizzazione** in TFN!

TERZA FORMA NORMALE

- **Definizione:** R<T, F> è in 3FN se per ogni $X \rightarrow A \in F^+$, con $A \notin X$, X è una superchiave o A è primo.
- Nota:
  - La 3FN ammette una dipendenza non banale e non-da-chiave se gli attributi a destra sono primi;
  - la BCNF non ammette mai nessuna dipendenza non banale e non-da-chiave.
- **Teorema:**

  R<T, F> è in 3FN se per ogni $X \rightarrow A \in F$ non banale,
  allora X è una superchiave oppure A è primo.

ESEMPI (Non sono in 3FN)

- **Definizione:** R<T, F> è in 3FN se per ogni $X \rightarrow A \in F^+$, con $A \notin X$, X è una superchiave o A è primo.

- Non sono in 3FN (e quindi, neppure in BCNF)
  - Docenti(CodiceFiscale, Nome, Dipartimento, Indirizzo)
    - { CF → N D; D → I }
  - Impiegati(Codice, Qualifica, NomeFiglio)
    - { C → Q }

ESEMPI (in 3FN, ma non in BCNF)

- Sono in 3FN, ma non in BCNF:
  - Telefoni(Prefisso, Numero, Località, Abbonato, Via)
    - F = {P N $\rightarrow$ L A V,   L $\rightarrow$ P}
    - Chiavi = {PN, LN}
  - Esami(Matricola, Telefono, Materia, Voto)
    - Matricola Materia $\rightarrow$ Voto
    - Matricola $\rightarrow$ Telefono
    - Telefono $\rightarrow$ Matricola
    - Chiavi:
      - Matricola Materia,
      - Telefono Materia

L'ALGORITMO DI SINTESI: VERSIONE BASE

Input: Un insieme R di attributi e un insieme F di dipendenze su R.

Output: Una decomposizione $\rho = \{S_i\}_{i=1..n}$ di R tale che preservi dati e dipendenze e ogni S_i sia in 3NF, rispetto alle proiezioni di F su S_i.

begin

Passo 1. Trova una copertura canonica G di F e poni $\rho = \{\ \}$.

Passo 2. Sostituisci in G ogni insieme $X \rightarrow A_1, \dots, X \rightarrow A_h$ di dipendenze con lo stesso determinante, con la dipendenza $X \rightarrow A_1 \cdots A_h$.

Passo 3. Per ogni dipendenza $X \rightarrow Y$ in G, metti uno schema con attributi XY in ρ.

Passo 4. Elimina ogni schema di ρ contenuto in un altro schema di ρ.

Passo 5. Se la decomposizione non contiene alcuno schema i cui attributi costituiscano una superchiave per R, aggiungi ad essa lo schema con attributi W, con W una chiave di R.

end

L'ALGORITMO DI SINTESI: VERSIONE BASE

- Dati R di attributi, ed un insieme di dipendenze F, **l'algoritmo di sintesi di schemi in terza forma normale** procede come segue:

Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

$\{M \rightarrow RSDG, MS \rightarrow CD, G \rightarrow R, D \rightarrow S, S \rightarrow D, MPD \rightarrow AM\}$

STEP 1. Costruire una copertura canonica G di F.

$F=\{M \rightarrow RSDG, MS \rightarrow CD, G \rightarrow R, D \rightarrow S, S \rightarrow D, MPD \rightarrow AM\}$

Vedi slide copertura canonica

$\mathbf{G=\{M \rightarrow D, M \rightarrow G, M \rightarrow C, G \rightarrow R, D \rightarrow S, S \rightarrow D, MP \rightarrow A\}}$

L'ALGORITMO DI SINTESI: VERSIONE BASE

$F=\{M \to RSDG, MS \to CD, G \to R, D \to S, S \to D, MPD \to AM\}$

$G=\{M \to D, M \to G, M \to C, G \to R, D \to S, S \to D, MP \to A\}$

- **STEP 2a. Decomporre G nei sottoinsiemi $G^{(1)}$, $G^{(2)}$, …, $G^{(n)}$,** tali che ad ogni sottoinsieme appartengano dipendenze con gli stessi lati sinistri (facoltativo)
- $G^{(1)}=\{M \to D, M \to G, M \to C\}$
- $G^{(2)}=\{G \to R\}$
- $G^{(3)}=\{D \to S\}$
- $G^{(4)}=\{S \to D\}$
- $G^{(5)}=\{MP \to A\}$

L'ALGORITMO DI SINTESI: VERSIONE BASE

$G=\{M\rightarrow D,\ M\rightarrow G,\ M\rightarrow C,\ G\rightarrow R,\ D\rightarrow S,\ S\rightarrow D,\ MP\rightarrow A\}$

- **STEP 2b** sostituisci in G ogni insieme $X \rightarrow A_1,\ldots, X\rightarrow A_h$ di dipendenze con lo stesso determinante, con la dipendenza $X \rightarrow A_1\ldots A_h$.

$G^{(1)}=\{M\rightarrow D,\ M\rightarrow G,\ M\rightarrow C\}$ $\Rightarrow$ $G^{(1)}=\{M\rightarrow DGC\}$

$G^{(2)}=\{G\rightarrow R\}$ $\Rightarrow$ $G^{(2)}=\{G\rightarrow R\}$

$G^{(3)}=\{D\rightarrow S\}$ $\Rightarrow$ $G^{(3)}=\{D\rightarrow S\}$

$G^{(4)}=\{S\rightarrow D\}$ $\Rightarrow$ $G^{(4)}=\{S\rightarrow D\}$

$G^{(5)}=\{MP\rightarrow A\}$ $\Rightarrow$ $G^{(5)}=\{MP\rightarrow A\}$

L'ALGORITMO DI SINTESI: VERSIONE BASE

- **STEP 3. Trasformare ciascun $G^{(i)}$ in una relazione $R^{(i)}$** con gli attributi contenuti in ciascuna dipendenza.
- Il lato sinistro diventa la chiave della relazione.
- Step 2 — Step 3
- $G^{(1)}=\{M \rightarrow DGC\}$: — $R^{(1)}(\underline{M}DGC)$
- $G^{(2)}=\{G \rightarrow R\}$: — $R^{(2)}(\underline{G}R)$
- $G^{(3)}=\{D \rightarrow S\}$: — $R^{(3)}(\underline{SD})$
- $G^{(4)}=\{S \rightarrow D\}$: — $R^{(4)}(\underline{SD})$
- $G^{(5)}=\{MP \rightarrow A\}$: — $R^{(5)}(\underline{MP}A)$

L'ALGORITMO DI SINTESI: VERSIONE BASE

- **STEP 4.** Si eliminano schemi contenuti in altri.

(se allo step precedente **due** o più lati sinistri delle dipendenze si implicano a vicenda, si fondono i relativi insiemi.)

Step 2	Step 3	Step 4
$G^{(1)}=\{M\to DGC\}$:	$R^{(1)}(\underline{M}DGC)$	$R^{(1)}(\underline{M}DGC)$
$G^{(2)}=\{G\to R\}$:	$R^{(2)}(\underline{G}R)$	$R^{(2)}(\underline{G}R)$
$G^{(3)}=\{D\to S\}$:	$R^{(3)}(\underline{D}\underline{S})$	$R^{(3)}(\underline{D}\underline{S})$
$G^{(4)}=\{S\to D\}$:	$R^{(4)}(\underline{S}\underline{D})$	
$G^{(5)}=\{MP\to A\}$:	$R^{(5)}(\underline{MP}A)$	$R^{(5)}(\underline{MP}A)$

$G^{(3)}=\{D\to S\}$
$G^{(4)}=\{S\to D\}$

$G^{(3)}=\{D\to S,\ S\to D\}$

L'ALGORITMO DI SINTESI: VERSIONE BASE

- **STEP 5**. Se <u>nessuna</u> relazione $R^{(i)}$ così ottenuta contiene una (super)chiave K di R(U), **inserire una nuova relazione $R^{(n+1)}$** contenente gli attributi della chiave.

- Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

$$F=\{M \rightarrow RSDG,\ MS \rightarrow CD,\ G \rightarrow R,\ D \rightarrow S,\ S \rightarrow D,\ MPD \rightarrow AM\}$$

$$G=\{M \rightarrow D,\ M \rightarrow G,\ M \rightarrow C,\ G \rightarrow R,\ D \rightarrow S,\ S \rightarrow D,\ MP \rightarrow A\}$$

- la chiave è costituita da: (MP).
- Dallo step 4: $R^{(1)}$(MDGC) $R^{(2)}$(GR) $R^{(3)}$(SD) **$R^{(5)}$(MPA)**
- **$R^{(5)}$(MPA) contiene la chiave** → non c'è necessità di aggiungere altre relazioni!

L'ALGORITMO DI SINTESI: VERSIONE BASE

- In conclusione, data la relazione: **R(MGCRDSPA)**, con dipendenze funzionali:

Impiegato (Matricola, Cognome, Grado, Retribuzione, Dipartimento, Supervisore, Progetto, Anzianità)

F= {M→RSDG, MS → CD, G→R, D→S, S→D, MPD→AM}

G={M→D, M→G, M →C, G→R, D→S, S→D, MP→A}

La sua decomposizione in 3FN è la seguente:

- **$R^{(1)}$(MDGC)** **$R^{(2)}$(GR)** **$R^{(3)}$(SD)** **$R^{(4)}$(MPA)**

Impiegati
Codice Stipendi: seq num NomeFigli: seq string

LE DF NON BASTANO: DIPENDENZE MULTIVALORE

- Impiegati(Codice, StoriaStipendio, NomeFiglio)

c1	s1	n1
c1	s1	n2
c1	s2	n1
c1	s2	n2

Verdi	1000	Giorgio
Verdi	1000	Anna
Verdi	1400	Giorgio
Verdi	1400	Anna
Rossi	1900	Gino

- La coesistenza di due proprietà multivalore INDIPENDENTI, fa sì che per ogni impiegato esistono tante ennuple quante sono le possibili coppie di valori di Qualifica e NomeFiglio.

Impiegati
Codice Qualifiche: seq num NomeFigli: seq string

Impiegati
Codice Posizioni: seq (Qualifica, NomeDirigente)

Tre casi di relazioni con proprietà multivalori. Si possono risolvere usando le decomposizioni?

Le DF non bastano: DF multivalore – Esempio 1

- Impiegati(Codice, StoriaStipendio, NomeFiglio)

c1	s1	n1
c1	s1	n2
c1	s2	n1
c1	s2	n2

Cogn	StoriaStip	Figlio
Verdi	1000	Giorgio
Verdi	1000	Anna
Verdi	1400	Giorgio
Verdi	1400	Anna
Rossi	1900	Gino

- La coesistenza di due proprietà multivalore **indipendenti**, fa sì che per ogni impiegato esistano tante ennuple quante sono le possibili coppie di valori di Stipendio e NomeFiglio.

Impiegati
Codice
Stipendi: seq num
NomeFigli: seq string

Le DF non bastano: DF multivalore – Esempio 1

- Impiegati(Codice, StoriaStipendio, NomeFiglio)

c1	s1	n1
c1	s1	n2
c1	s2	n1
c1	s2	n2

Verdi	1000	Giorgio
Verdi	1000	Anna
Verdi	1400	Giorgio
Verdi	1400	Anna
Rossi	1900	Gino

- Decomponendo lo schema in due sottoschemi in modo da modellare separatamente le proprietà multivalori indipendenti, si avrebbe una base di dati priva di anomalie:
- StipendiImpiegati(Codice, StoriaStipendio)
- FigliImpiegati(Codice, NomeFiglio)

LE DF NON BASTANO: DIPENDENZE MULTIVALORE

- Altro esempi:

Impiegati

Codice
Qualifiche: seq num
NomeFigli: seq string

Impiegati

Codice
Posizioni: seq (Qualifica, NomeDirigente)

Si possono risolvere usando le decomposizioni?