"At the time, Nixon was normalizing relations with China. I figured that if he could normalize relations then so could I." E.F. Codd [1]

Teoria della Normalizzazione

Raffaella Gentilini

[1] www.dba-oracle.com/oracle_tips_codd_obit.htm

Obbiettivi

Sviluppare una metodologia che permetta di:

1. Decidere se uno schema di relazione e' uno schema di relazione ben definito
2. Qualora uno schma di relazione R non soddisfi i criteri di bonta', decomporlo in $\{R_1, \ldots, R_n\}$, dove :
   - ogni R_i sia uno schema di relazione ben definito
   - non vi sia perdita di informazione

Il nostro approccio e' basato sui concetti di:

- dipendenze funzionali

Dipendenze Funzionali

Dipendenze Funzionali

- Generalizzazione del concetto di chiave
- Esprimono vincoli sulla ammissibilita' delle istanze di relazione
- Stabiliscono vincoli di dipendenza tra attributi:
    - I valori di alcuni attributi determinino i valori di altri attributi nelle tuple

Dipendenze Funzionali

Definition (Dipendenze Funzionali)

Dato lo schema di relazione R sull'insieme di attributi X, si considerino $\alpha \subseteq X$ e $\beta \subseteq X$.
La dipendenza funzionale $\alpha \to \beta$ vale su R

$$\Updownarrow$$

Per ogni istanza di r di R:

- Ogni coppia di ennuple t_1, t_2 di r avente gli stessi valori per gli attributi in α, ha gli stessi valori per gli attributi in β.

Formalmente:

Dipendenze Funzionali

Definition (Dipendenze Funzionali)

Dato lo schema di relazione R sull'insieme di attributi X, si considerino $\alpha \subseteq X$ e $\beta \subseteq X$.
La dipendenza funzionale $\alpha \rightarrow \beta$ vale su R

$$\Updownarrow$$

Per ogni istanza di r di R:

- Ogni coppia di ennuple t_1, t_2 di r avente gli stessi valori per gli attributi in α, ha gli stessi valori per gli attributi in β.

Formalmente:

$$\alpha \rightarrow \beta \text{ vale su R}$$
$$\Updownarrow$$
$$\forall r \text{ istanza di } R, \forall t_1, t_2 \in r : t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$$

Example

Si consideri la seguente istanza di R dello schema $R(A, B)$:

A	B
3	4
1	5
3	7

- Si osservi che $A \rightarrow B$ non vale

Dipendenze Funzionali e Chiavi

- K e' superchiave per $R(X)$ sse $K \rightarrow X$.
- K e' chiave candidata per $R(X)$ sse:
  - $K \rightarrow X$
  - non esiste $K' \subset K$ tale che $K' \rightarrow X$
- dipendenze funzionali permettono di esprimere vincoli non esprimibili tramite nozione di chiave:

Vendita(nomeCliente, codiceMerce, nomeProduttore, costo)

Dato lo schema sopra, desideriamo valgano le dipendenze:

- codiceMerce $\rightarrow$ costo
- codiceMerce $\rightarrow$ produttore

ma non desideriamo che valga:

- codiceMerce $\rightarrow$ nomeCliente

Chiusura Insieme Dipendenze Funzionali

- Dato un insieme F di dipendenze funzionali, vi possono essere altre dipendenze funzionali logicamente implicate da F.
  - Ad esempio, se valgono $A \rightarrow B$ e $B \rightarrow C$, possimao infereire che vale $A \rightarrow C$
- L'insieme di dipendenze funzionali logicamente implicate da F, denotato F^+, e' detto chiusura di F
- Possiamo determinare F^+ applicando gli assiomi di Armstrong
- Assiomi Armstrong sono insieme regole inferenza corretto (generano solo DF valide) e completo (generano tutte le DF in F^+)

Chiusura Insieme Dipendenze Funzionali

- Dato un insieme F di dipendenze funzionali, vi possono essere altre dipendenze funzionali logicamente implicate da F.
  - Ad esempio, se valgono $A \rightarrow B$ e $B \rightarrow C$, possimao infereire che vale $A \rightarrow C$
- L'insieme di dipendenze funzionali logicamente implicate da F, denotato F^+, e' detto chiusura di F
- Possiamo determinare F^+ applicando gli assiomi di Armstrong
- Assiomi Armstrong sono insieme regole inferenza corretto (generano solo DF valide) e completo (generano tutte le DF in F^+)

Assiomi di Armstrong

1. se $\beta \subseteq \alpha$, allora $\alpha \rightarrow \beta$ (riflessivita')
2. se $\alpha \rightarrow \beta$, allora $\gamma\alpha \rightarrow \gamma\beta$ (arrichimento)
3. se $\alpha \rightarrow \beta$ e $\beta \rightarrow \gamma$, allora $\alpha \rightarrow \gamma$ (transitivita')

Example

- $R = (A, B, C, G, H, I)$
  $F = \{A \to B, A \to C, CG \to H, CG \to I, B \to H\}$
- Alcuni membri di F^+ sono:
  - $A \to H$
  - $AG \to I$
  - $CG \to HI$

Example

- $R = (A, B, C, G, H, I)$
  $F = \{A \to B, A \to C, CG \to H, CG \to I, B \to H\}$
  G G
- Alcuni membri di F^+ sono:
  - $A \to H$
    - per transitivita' da $A \to B$ e $B \to H$
  - $AG \to I$
  - $CG \to HI$

Example

- $R = (A, B, C, G, H, I)$
  $F = \{A \to B, A \to C, CG \to H, CG \to I, B \to H\}$
- Alcuni membri di F^+ sono:
  - $A \to H$
    - per transitivita' da $A \to B$ e $B \to H$
  - $AG \to I$
    - arricchendo $A \to C$ con G e poi utilizzando la transitivita' con $CG \to I$
  - $CG \to HI$
    - arricchimento di $CG \to I$ con CG per ottenere $CG \to CGI$, arricchimento di $CG \to H$ con I per ottenere $CGI \to HI$, ed infine applicazione della transitivita'.

Calcolo di F^+

Algoritmo per il calcolo della chiusura di un insieme di dipendenze funzionali F

$F^+ \leftarrow F$
repeat
  for each dipendenza funzionale $f \in F^+$ **do**
    applica riflessivita' ed arrichimento ad f
    aggiungi ad F^+ le dipendenze ottenute
  end for
  for each coppia di dipendenze funzionali $f_1, f_2 \in F^+$ **do**
    if f_1 ed f_2 possono essere combinate usando la transitivita' **then**
      aggiungi ad F^+ le dipendenze ottenute
    end if
  end for
until F^+ non cambia

Calcolo di F^+

Possiamo velocizzare/semplificare il calcolo della chiusura di F^+ utilizzando ulteriori regole di inferenza:

- Unione Se valgono $\alpha \rightarrow \beta$ e $\alpha \rightarrow \gamma$, allora vale $\alpha \rightarrow \beta\gamma$
- Decomposizione Se vale $\alpha \rightarrow \beta\gamma$, allora valgono $\alpha \rightarrow \beta$ ed $\alpha \rightarrow \gamma$
- Pseudotransitivita' Se valgono $\alpha \rightarrow \gamma$ e $\gamma\beta \rightarrow \delta$, allora vale anche $\alpha\beta \rightarrow \delta$

Calcolo di F^+

Possiamo velocizzare/semplificare il calcolo della chiusura di F^+ utilizzando ulteriori regole di inferenza:

- Unione Se valgono $\alpha \to \beta$ e $\alpha \to \gamma$, allora vale $\alpha \to \beta\gamma$
- Decomposizione Se vale $\alpha \to \beta\gamma$, allora valgono $\alpha \to \beta$ ed $\alpha \to \gamma$
- Pseudotransitivita' Se valgono $\alpha \to \gamma$ e $\gamma\beta \to \delta$, allora vale anche $\alpha\beta \to \delta$

Esercizio: Ricavare le precedenti regole a partire dagli assiomi di Armstrong.

Chiusura di un Insieme di Attributi

Dato un insieme di attributi α, la chiusura di α rispetto ad F (denotato α^+) e' l'insieme di attributi determinati funzionalmente da attributi in α utilizzando le dipendenze in F.

Avremo che:

- $\alpha\beta \in F$ sse $\beta \subseteq \alpha^+$

Calcolo di α^+ rispetto ad F

$\alpha^+ \leftarrow \alpha$
while ci sono cambiamenti in α^+ **do**
    for each $\beta \rightarrow \gamma \in F$ **do**
        if $\beta \subseteq \alpha^+$ **then**
            $\alpha^+ \leftarrow \alpha^+ \cup \gamma$
        end if
    end for
end while

Example

- $R(X) = (A, B, C, G, H, I)$, ovvero $X = ABCGHI$
- $F = \{A \rightarrow B, A \rightarrow C, CG \rightarrow H, B \rightarrow H\}$
- Calcolo di CG^+ rispetto ad F:
  1. $AG^+ \leftarrow AG$
  2. $AG^+ \leftarrow ABCG$ (da $A \rightarrow C$ e $A \rightarrow B$)
  3. $AG^+ \leftarrow ABCGH$ (da $CG \rightarrow H$ e $CG \subseteq ABCG$)
  4. $AG^+ \leftarrow ABCGHI$ (da $CG \rightarrow I$ e $CG \subseteq ABCGH$)

Usare la chiusura di attributi ...

Viene sfruttata in diversi contesti:

- per verificare se un insieme di attributi e' una superchiave.
  - $\alpha \subseteq X$ e' superchiave per R sse α^+ contiene tutti gli attributi di $R(X)$.
- per verificare se vale una dipendenza funzionale.
  - per verificare se vale $\alpha \rightarrow \beta$ (ovvero se $\alpha \rightarrow \beta$ appartiene ad F^+) basta verificare se $\beta \subseteq \alpha^+$.
- calcolo della chiusura di F.
  - per ogni $\gamma \subseteq X$, si calcola la chiusura γ^+ e per ogni $Y \subseteq \gamma^+$ si genera la DF $\gamma \rightarrow Y$.

Copertura Minimale

- Un insieme F di dipendenze funzionale puo' contenere dipendenze ridondanti, ovvero che possono essere ottenute dalle altre dipendenze di F
  - Esempio: $A \rightarrow C$ e' ridondante in $\{A \rightarrow B, B \rightarrow C, A \rightarrow C\}$
- Anche degliattributi di una dipendenza funzionale potrebbero essere ridondanti:
  - A destra: $\{A \rightarrow B, B \rightarrow C, A \rightarrow CD\}$ puo' essere semplificata in $\{A \rightarrow B, B \rightarrow C, A \rightarrow D\}$
  - A sinistra: $\{A \rightarrow B, B \rightarrow C, AC \rightarrow D\}$ puo' essere semplificata in $\{A \rightarrow B, B \rightarrow C, A \rightarrow D\}$
- Intuitivamente, una copertura minimale di F e' un insieme minimale di dipendenze funzionali equivalenti ad F, privo di dipendenze e attributi ridondanti.

Copertura Minimale

Piu' formalmente, un insieme F di dipendenze funzionali e' minimale sse:

1. Ogni dipendenza funzionale in F ha come parte destra un solo attributo
2. Non e' possibile sostituire una dipendenza funzionale $\alpha \rightarrow A$ di F con una dipendenza funzionale $\beta \rightarrow A$ dove $\beta \subset \alpha$, ed avere ancora un insieme di dipendenze funzionali equivalente ad F.
3. Non e' possibile rimuovere una dipendenza funzionale da F e avere ancora un insieme di dipendenze funzionali equivalente ad F.

Una copertura minimale di un insieme di dipendenze funzionali F e' un insieme minimale di dipendenze funzionali E equivalente ad F.

Copertura Minimale

Calcolo di una copertura minimale E per un insieme di DF F.

1. Si imposti $E := F$
2. Si sostituisca ogni DF $X \rightarrow A_1 \ldots A_n$ in E con le n DF $X \rightarrow A_1, \ldots, X \rightarrow A_n$.
3. Per ogni DF $X \rightarrow A$ in E, per ogni attributo B in X:

   Se B e' ridondante nella DF $X \rightarrow A$, ovvero se E e' equivalente a $(E \setminus \{X \rightarrow A\}) \cup \{(X \setminus \{B\} \rightarrow A\}$, allora si sostituisca $X \rightarrow A$ con $X \setminus \{B\} \rightarrow A$ in E
4. Per ogni DF rimanente $X \rightarrow A$: Se $E \setminus \{X \rightarrow A\}$ e' equivalente ad E, allora si rimuova $X \rightarrow A$ da E.

Copertura Minimale

Come verificare ridondanza attributi?

- Sia F un insieme di DF. Consideriamo la DF $X \to Y$ in F e l'attributo $B \in X$.
- Per verificare se $B \in X$ e' ridondante:
  - Calcoliamo la chiusura $(X \setminus \{B\})^+$ rispetto ad F
  - Verifichiamo se $(X \setminus \{B\})^+$ contiene Y
  - Se si', allora B e' ridondante (e puo' essere eliminato).

Example

- $R = (A, B, C)$
- $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$
- Dopo l'esecuzione del passo (2) dell'algoritmo si ha:

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

- Eseguiamo il passo (3). A e' ridondante in $AB \rightarrow C$?
  - Verifichiamo se la chiusura di B rispetto ad E contiene C
    - Si: Infatti $B^+ = \{B, C\}$. Dunque E diventa $E = \{A \rightarrow B, A \rightarrow C, B \rightarrow C\}$
- Eseguiamo il passo (4):
  - $A \rightarrow C$ e' implicata logicamente da $A \rightarrow B, B \rightarrow C$ (per transitivita'). Dunque E diventa $E = \{A \rightarrow B, B \rightarrow C\}$.
- Una copertura canonica (o minimale) e':

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

Forme Normali basate su Dipendenze Funzionali

- 1NF: Prima Forma Normale
- 2NF: Seconda Forma normale
- 3NF: Terza Forma Normale
- BCNF: Forma Normale di Boyce-Codd

Normalizzare uno schema di relazione R

=

Decomporre (opportunamente) R in schemi che siano in forma normale

Normalizzare sfruttando le Dipendenze Funzionali

Decomponendo uno schema di relazione R sfruttando un insieme di dipendenze funzionali F in un insieme di schemi $R_1 \ldots R_n$ vogliamo:

- Minimizzare la ridondanza
- Decomposizione Lossless-join: Senza perdita di informazione
- Conservare le dipendenze: Se F_i e' l'insieme delle dipendenze di F^+ che includono solo attributi di R_i, allora:
  - La decomposizione dovrebbe essere dependency preserving, cioe' $(F_1 \cup \cdots \cup F_n)^+ = F^+$
  - altrimenti il controllo delle violazioni delle dipendenze funzionali (dello schema originario) comporterebbe la computazione esplicita di operazioni di join.

Example

- $R = (A, B, C)$, $F = \{A \to B, B \to C\}$
  - puo' essere decomposto in due modi diversi:

1. $R_1 = (A, B)$, $R_2 = (B, C)$
   - decomposizione senza perdite
   - conserva le dipendenze
2. $R_1 = (A, B)$, $R_2 = A, C)$
   - decomposizione senza perdite
   - non conserva le dipendenze (non posso controllare se viene violato il vincolo $B \to C$ senza calcolare $R_1 \bowtie R_2$)

Verificare la Conservazione delle Dipendenze

- Per verificare se la dipendenza $\alpha \rightarrow \beta$ e' preservata in una decomposizione di R in $R_1 \ldots R_n$, applichiamo il seguente test (le chiusure di attributi sono fatte rispetto ad F):

result $\leftarrow \alpha$
while result cambia **do**
  for each R_i nella decomposizione **do**
    $t = (result \cap R_i)^+ \cap R_i$
    $result \leftarrow result \cup t$
  end for
end while

- Se $result \supseteq \beta$, allora la DF $\alpha \rightarrow \beta$ e' preservata.
- Applicheremo il test su tutte le dipendenze di F
- Questa procedura e' polinomiale, mentre la computazione di F^+ e $(F_1 \cup \cdots \cup F_n)^+$ richiede un tempo esponenziale.

Boyce-Codd Normal Form (BCNF)

Definizione: Boyce-Codd Normal Form

Uno schema di relazione $R(X)$ e' in BCNF rispetto ad un insieme F di dipendenze funzionali, se per ogni dipendenza in F^+ della forma $\alpha \to \beta$, $\alpha, \beta \subseteq X$, almeno una delle seguenti condizioni e' soddisfatta:

- $\alpha \to \beta$ e' banale (ovvero $\beta \subseteq \alpha$)
- α e' superchiave di $R(X)$

Example

- $R(X) = (A, B, C)$, $F = \{A \rightarrow B, B \rightarrow C\}$
  - A e' chiave
- R non e' in BCNF
- Decomposizione: $R_1 = (A, B), R_2 = (B, C)$
  - R_1 e R_2 sono in BCNF
  - la decomposizione e' senza perdite
  - e preserva le dipendenze

Algoritmo per la decomposizione in BCNF

$result \leftarrow \{R\}$;
$done \leftarrow false$
while *not done* **do**
  if $\exists S \in result$ non in BCNF **then**
    si determini una DF $\alpha \rightarrow \beta$ su S che violi BCNF
    $result \leftarrow (result \setminus S) \cup \{(S \setminus \beta)\} \cup \{(\alpha\beta)\}$
  else
    $done \leftarrow true$
  end if
end while

Test per BCNF

- Per verificare se DF non banale$\alpha \rightarrow \beta$ causa violazione della BCNF:
  - computare α^+ (la chiusura di α), e verificare se include tutti gli attributi di R, cioe' se α^+ e' superchiave di R
- Test semplificato: Per verificare se uno schema R e' in BCNF, e' sufficiente verificare solo che le DF in F non violano la BCNF (invece di controllare tutte le dipendenze di F^+). Infatti:
  - se nessuna delle DF in F causa una violazione della BCNF, allora nessuna delle DF in F^+ causa una violazione della BCNF
- Tuttavia, utilizzare solo F e' scorretto quando si effettua il test su una relazione della decomposizione di R.
  - Ad esempio, consideriamo $R(A, B, C, D)$ ed $F = \{A \rightarrow B, B \rightarrow C\}$
    - decomponiamo R in $R_1(A, B)$ e $R_2(A, C, D)$
    - nessuna delle DF in F contiene solo attributi in (A, C, D), tuttavia la DF $A \rightarrow C \in F^+$ mostra che R_2 non e' in BCNF

Test per BCNF

Per verificare se uno schema R_i di una decomposizione di R e' in BCNF si opera come segue:

- o verificare se R_i e' in BCNF rispetto alla restrizione di F^+ su R_i (cioe' tutte le dipendenze funzionali in R^+ che contengono solo attributi di R_i
- oppure effettuare sull'insieme di DF F il seguente test:
  - per ogni insieme di attributi $\alpha \subseteq R_i$, verificare che α^+ o non includa attributi di $R_i \setminus \alpha$, oppure includa tutti gli attributi di R_i
  - se la condizione sopra e' violata da qualche $\alpha \rightarrow \beta \in F$, si dimostra che la DF $\alpha \rightarrow (\alpha^+ \setminus \alpha) \cap R_i$ certifica che R_i viola BCNF.
  - Le dipendenze di questo tipo saranno usate per decomporre ulteriormente R_i

BCNF e conservazione delle dipendenze

Non e' sempre possibile ottenere una BCNF che conservi le dipendenze:

Example

- $R = (J, K, L)$, $F = \{JK \rightarrow L, L \rightarrow K\}$
- due chiavi candidate: JK e JL
- R non e' in BCNF
- ogni possibile decomposizione di R non preserva $JK \rightarrow L$.

Terza Forma Normale: Motivazioni

- Ci sono casi in cui:
    - BCNF non preserva le dipendenze, mentre e' necessario avere una procedura efficiente per mantenere le DF
- Soluzione: Definire una forma normale piu' debole (vedremo ora la terza forma normale – 3NF).
    - ammettere della ridondanza (con i conseguenti svantaggi; vedremo esempio) ma
    - garantire che le DF possano essere controllate sulle relazioni decomposte, senza alcun join.
- Proprieta': Esiste sempre una decomposizione in 3NF che conserva le dipendenze.

Terza Forma Normale (3NF)

Definizione: Terza Forma Normale

Uno schema di relazione $R(X)$ e' in terza forma normale rispetto ad un insieme F di dipendenze funzionali, se per ogni dipendenza in F^+ della forma $\alpha \rightarrow \beta$, $\alpha, \beta \subseteq X$, almeno una delle seguenti condizioni e' soddisfatta:

- $\alpha \rightarrow \beta$ e' banale (ovvero $\beta \subseteq \alpha$)
- α e' superchiave di $R(X)$
- ogni attributo A in $\beta \setminus \alpha$ e' contenuto in una chiave candidata di R

- Una relazione in BCNF e' anche in 3NF
- La terza condizione e' il rilassamento della BCNF che assicura la conservazione delle dipendenze.

3NF: Esempio

Example

- $R = (J, K, L)$, $F = \{JK \rightarrow L, L \rightarrow K\}$
- due chiavi candidate: JK e JL
- R e' in 3NF
  - $JK \rightarrow L$: JK e' superchiave
  - $L \rightarrow K$: K e' contenuta in una chiave candidata
- La decomposizione in BCNF ha i due schemi (JL), (LK)
  - verificare il rispetto della DF $JK \rightarrow L$ richiederebbe un join
- nello schema c'e' ridondanza

Algoritmo di Decomposizione in 3NF

1. Sia G una copertura canonica di F
2. Per ogni parte sinistra X in una DF in G:
   - si definisca schema D con attributi $\{X \cup \{A_1\} \cup \cdots \cup \{A_k\}\}$, dove $X \rightarrow A_1, \ldots, X \rightarrow A_k$ sono le sole dipendenze di G con X come parte sinistra
   - X sara' la chiave dello schema
3. Se nessuno degli schemi di relazione in D contiene una chiave di R, si definisca un ulteriore schema di relazione D contenente attributi che formano una chiave di R
4. Si eliminino le relazioni ridondanti (i.e. proiezioni di altre relazioni)

Algoritmo di Decomposizione in 3NF

Si dimostra che l'algoritmo visto e' tale che:

- e' corretto
- ogni schema R_i e' in NF
- la decomposizione conserva le dipendenze ed e' senza perdite.

Decomposizione in 3NF: Esempio

Example

- $R(nomeDitta, nomeCliente, nomeImp, numUff)$
- $nomeImp \rightarrow nomeDitta\ numUff$
  $nomeCliente\ nomeDitta \rightarrow nomeImp$
- Il passo 2 inserisce i seguenti schemi nella decomposizione:
  - $S(nomeimpiegato, nomeDitta, numUff)$
  - $T(nomeCliente, nomeDitta, nomeImp)$
- Poiche' T contiene una chiave candidata per R abbiamo finito

Comparazione di BCNF e 3NF

- Per ogni dato schema e' sempre possibile calcolare una 3NF:
  - senza perdite
  - che conserva le dipendenze
- Per ogni dato schema e' sempre possibile calcolare una BCNF
  - senza perdite
  - potrebbe non preservare tutte le dipendenze

Comparazione di BCNF e 3NF

- Esempio di problemi dovuti alla ridondanza ammessa dalla 3NF
  - $R = (J, K, L)$
  - $F = \{JK \rightarrow L, L \rightarrow K\}$

J	**L**	**K**
j_1	l_1	k_1
j_2	l_1	k_1
j_3	l_1	k_1
null	l_2	k_2

Uno schema in 3NF ma non in BCNF comporta:

- ripetizione di informazione (ad esempio, la coppia di dati l_1, k_1)
- impiego di valori nulli (ad esempio, per rappresentare la correlazione tra l_2 e k_2 quando non ci sono corrispondenti valori per J.