Normalization

La "forma normale" è una proprietà di un database relazionale che ne garantisce la qualità. Quando una relazione non è in forma normale:

- presenta ridondanze;
- può avere comportamenti indesiderati durante gli aggiornamenti;
  La **normalizzazione** deve essere usata come una tecnica di verifica per testare il risultato del design della base di dati, non è una metodo per il design del database.
  Esempio:

Impiegato	Stipendio	Progetto	Bilancio	Funzione
Rossi	20 000	Marte	2000	Tecnico
Verdi	35 000	Giove	15 000	Progettista
Verdi	35 000	Venere	15 000	Progettista
Neri	55 000	Venere	15 000	Direttore
Neri	55 000	Giove	15 000	Consulente
Neri	55 000	Marte	2000	Consulente
Mori	48 000	Marte	2000	Direttore
Mori	48 000	Venere	15 000	Progettista
Bianchi	48 000	Venere	15 000	Progettista
Bianchi	48 000	Giove	15 000	Direttore

  le tuple di questa relazione rispettano queste proprietà:
- lo stipendio di ogni impiegato è unico ed è funzione del solo impiegato, indipendentemente dai progetti cui partecipa;
- il bilancio di ciascun progetto è unico e dipende dal solo progetto, indipendentemente dagli impiegati che partecipano.
  Anomalie:
- **Ridondanza**: il valore dello stipendio di ciascun impiegato è ripetuto in tutte le tuple;
- **Anomalia di aggiornamento**: se lo stipendio di un impiegato varia, è necessario andarne a modificare il valore in tutte le tuple corrispondenti, comporta la necessità di più modifiche contemporaneamente;
- **Anomalia di cancellazione**: se un impiegato interrompe la partecipazione a tutti i progetti senza lasciare l'azienda, tutte le tuple corrispondenti vengono eliminate e non è possibile conservare traccia del suo nome e del suo stipendio;
- **Anomalia di inserimento**: se si hanno informazioni su un nuovo impiegato, non è possibile inserire finché non viene assegnato ad un progetto.
  Queste anomalie sono causate dal fatto che è stata utilizzata una sola relazione per rappresentare relazioni che riuniscono concetti fra loro disomogenei.

Dipendenze Funzionali

Sono un particolare vincolo di integrità che descrive legami di tipo funzionale tra gli attributi di una relazione. Formalizzato:

> **Def**
>
> Data una relazione r su uno schema $R(X)$ e due sottoinsiemi non vuoti Y e Z di X, esiste su r una dipendenza funzionale tra Y e Z se, per ogni coppia di tuple t_1 e t_2 di r aventi gli stessi valori sugli

attributi Y, risulta che t_1 e t_2 hanno gli stessi valori anche sugli attributi di Z.
Una dipendenza funzionale tra Y e Z viene generalmente indicata con $Y \rightarrow Z$.
Viene associata ad uno schema: una relazione su quello schema verrà considerata corretta se soddisfa tale dipendenza funzionale.

Nel nostro esempio: *Impiegato* → *Stipendio*
Progetto → *Bilancio*
Impiegato Progetto → *Funzione*

Osservazioni:

se l'insieme Z è composto dagli attributi $A_1, A_2, ..., A_k$, allora una relazione soddisfa $Y \rightarrow Z \Leftrightarrow$ soddisfa tutte le k dipendenze $Y \rightarrow A_1$, $Y \rightarrow A_2$, ..., $Y \rightarrow A_k$.

Diremo che una dipendenza funzionale $Y \rightarrow A$ è ***non banale*** se A non compare tra gli attributi di Y.
Impiegato Progetto → *Funzione* è una dipendenza *banale*.

Osservazione sulle dipendenze funzionali e il legame con la chiave:

Se prendiamo una chiave K di una relazione r, si verifica facilmente che esiste una dipendenza funzionale tra K e ogni altro attributo dello schema di r; infatti, per definizione stessa di vincolo di chiave, non possono esistere due tuple con gli stessi valori su K.

Nel nostro esempio:

- le prime due dipendenze funzionali non sono chiavi e causano anomalie;
- la terza dipendenza funzionale (*Impiegato Progetto*) è una chiave e non causa anomalie;

Forma normale di Boyce-Codd

Def

Una relazione r è in *forma normale di Boyce-Codd* se per ogni dipendenza funzionale (non banale) $X \rightarrow A$ definita su di essa, X contiene una chiave K di r, cioè X è superchiave per r.
Anomalie e ridondanze non si presentano per relazioni in forma normale di Boyce-Codd, perchè i concetti indipendenti sono separati, uno per relazione.

Decomposizione in BCNF

Data una relazione che non soddisfa la BCNF è possibile, in molti casi, sostituirla con due o più relazioni normalizzate attraverso un processo detto di ***normalizzazione***: se una relazione rappresenta più concetti indipendenti, allora va decomposta in relazioni più, piccole, una per ogni concetto.

Impiegato	Stipendio
Rossi	20 000
Verdi	35 000
Neri	55 000
Mori	48 000
Bianchi	48 000

Progetto	Bilancio
Marte	2000
Giove	15 000
Venere	15 000

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

Nell'esempio vengono costruite delle nuove relazioni in modo che a ciascuna dipendenza corrisponda una diversa relazione la cui chiave è proprio il primo membro della dipendenza stessa.
In molti casi, la decomposizione può essere effettuata producendo tante relazioni quante sono le dipendenze funzionali definite; in generale, le dipendenze possono avere una struttura complessa: può non essere necessario, o possibile, basare la decomposizione su tutte le dipendenze e può essere difficile individuare quelle su cui si deve basare la decomposizione.

Employee	Project	Office
Jones	Mars	Rome
Smith	Jupiter	Milan
Smith	Venus	Milan
White	Saturn	Milan
White	Venus	Milan

Employee → Office

Employee	Office
Jones	Rome
Smith	Milan
White	Milan

Project → Office

Project	Office
Mars	Rome
Jupiter	Milan
Venus	Milan
Saturn	Milan

Employee	Office
Jones	Rome
Smith	Milan
White	Milan

⋈

Office	Project
Rome	Mars
Milan	Jupiter
Milan	Venus
Milan	Saturn

=

Employee	Office	Project
Jones	Rome	Mars
Smith	Milan	Jupiter
Smith	Milan	Venus
Smith	**Milan**	**Saturn**
White	**Milan**	**Jupiter**
White	Milan	Saturn
White	Milan	Venus

≠

Employee	Office	Project
Jones	Rome	Mars
Smith	Milan	Jupiter
Smith	Milan	Venus
White	Milan	Saturn
White	Milan	Venus

DIFFERENT FROM THE ORIGINAL RELATION!

Decomposizione senza perdita

Def

Caso Generale: data una relazione r su un insieme di attributi X, se X_1 e X_2 sono due sottoinsiemi di X la cui unione sia pari a X stesso, allora il join delle due relazioni ottenute per proiezione da r su X_1 e X_2, rispettivamente, è una relazione che contiene tutte le tuple di r, più eventualmente altre che possiamo chiamare "spurie". Diciamo che r si *decompone senza perdita* su X_1 e X_2 se il join delle due proiezioni è uguale a r stessa (cioè non contiene tuple spurie).

Condizione per la decomposizione senza perdita:

Sia r una relazione su X e siano X_1 e X_2 sottoinsiemi di X tali che $X_1 \cup X_2 = X$;
inoltre, sia $X_0 = X_1 \cap X_2$;
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 $X_0 \rightarrow X_2$.
In altre parole, r si decompone senza perdita su due relazioni se l'insieme degli attributi comuni alle due relazione è chiave per almeno una delle relazioni composte.

Conservazione delle dipendenze

Per garantire che tutte le dipendenze funzionali dello schema originale siano ancora rispettate nella decomposizione è necessario che: ogni dipendenza funzionale dello schema originale coinvolga attributi che compaiono tutti in almeno uno degli schemi della decomposizione; questo assicura che i vincoli e le relazioni tra i dati originali rimangano validi anche dopo la decomposizione.
Esempio:

- schema originale: **R**(A, B, C)
- dipendenza funzionale: A → B, ovvero il valore dell'attributo A determina univocamente il valore dell'attributo B
- decomposizione:
  1. $\mathbf{R}_1$(A, B)
  2. $\mathbf{R}_2$(A, C)
- verifica:
  - dopo la decomposizione lo schema $\mathbf{R}_1$(A, B) conserva la dipendenza A → B perché entrambi gli attributi A e B sono inclusi in $\mathbf{R}_1$.
  ⇒ la decomposizione conserva le dipendenze funzionali dello schema originale.

Qualità delle decomposizioni

- La **decomposizione senza perdita** garantisce che le informazioni nella relazione originaria siano ricostruibili con precisione;
- la **conservazione delle dipendenze** garantisce che le relazioni decomposte hanno la stessa capacità della relazione originaria di rappresentare i vincoli di integrità e quindi di rilevare aggiornamenti illeciti: a ogni aggiornamento lecito sulla relazione originaria corrisponde un aggiornamento lecito sulle relazioni decomposte.

Terza forma normale

- **limitazioni BCNF**:

Chief	Project	Office
Smith	Mars	Rome
Johnson	Jupiter	Milan
Johnson	Mars	Milan
White	Saturn	Milan
White	Venus	Milan

Project Office → Chief

Chief → Office

la relazione non è in BCNF perché il primo membro della dipendenza Chief → Office non è superchiave. Inoltre non è possibile decomporre bene questa relazione in quanto la dipendenza **Project Office → Chief** coinvolge tutti gli attributi.
⇒ Talvolta la forma normale di Boyce-Codd non è raggiungibile.

Terza Forma Normale:

Diciamo che una relazione *r* è in *terza forma normale* se, per ogni dipendenza funzionale (non banale) $X \rightarrow A$ definita su di essa, almeno una delle seguenti condizioni è verificata:

- X contiene una chiave K di r;
- A appartiene ad almeno una chiave di r.

- la BCNF è più forte della 3NF (la 3NF ammette relazioni con anomalie);
- 3NF può essere sempre raggiunta;
- Se una relazione ha una sola chiave, è in BCNF se e solo se è in 3NF.
  Infatti, la dipendenza **Project Office → Chief** ha come primo membro una chiave della relazione, mentre **Chief → Office**, pur non contenendo una chiave al primo membro, ha un unico attributo a secondo membro che fa parte della chiave **Project → Office**.

Decomposizione in terza forma normale

Una relazione che non soddisfa la terza forma normale si decompone in relazioni ottenute per proiezione sugli attributi corrispondenti alle dipendenze funzionali, con l'accortezza di mantenere sempre una relazione che contiene una chiave della relazione originaria.

1. identificare le dipendenze funzionali nello schema
2. decomporre lo schema originale in più schemi, in modo che ogni schema soddisfi i requisiti della 3NF;
3. assicurarsi che la decomposizione conservi le dipendenze e che sia possibile ricostruire lo schema originale usando i sottoschemi (proprietà di lossless join).

Esempio:
R(A, B, C)
dipendenze funzionali sono:

4. A → B
5. B → C

dove A è la chiave primaria;

- la dipendenza B → C crea una dipendenza transitiva:
  - A → B e B → C ⇒ A → C, ma C dipende indirettamente da A tramite B
  difatti questo schema non è 3NF:
- creiamo due schemi:
  - R_1 (A, B): per rappresentare A → B
  - R_2 (B, C): per rappresentare B → C
- verifica:
  - schema R_1 (A, B): È in 3NF, perché B dipende direttamente dalla chiave primaria A;
  - Schema R_2 (B, C): È in 3NF, perché C dipende direttamente dalla chiave primaria B.
    Unendo R_1 e R_2, possiamo ricostruire lo schema originale senza perdita di dati.

Teoria delle dipendenze

Data una relazione e un insieme di dipendenze funzionali su di essa, generare una decomposizione di tale relazione che contenga solo le relazioni in forma normale che soddisfi le proprietà della decomposizione di cui abbiamo già parlato:

- decomposizione senza perdita;
- preservazione delle dipendenze.

> **Def**
> Diciamo che un insieme di dipendenze funzionali *F implica* un'altra dipendenza *f* se ogni relazione che soddisfa tutte le dipendenze in *F* soddisfa anche *f*.

Esempio: A → B, B → C ⇒ A → C

> **Chiusura di un insieme di attributi**
>
> Siano dati uno schema di relazione R(U) e un insieme di dipendenze funzionali F definite sugli attributi in U. Sia X un insieme di attributi contenuti in U (cioè $X \subseteq U$); la *chiusura* di X rispetto a F, indicata con X_F^+, è l'insieme degli attributi che dipendono funzionalmente da X (esplicitamente o implicitamente):
> $X_F^+ = \{A \mid A \in U \text{ e } F \text{ implica } X \to A\}$
> Se vogliamo vedere se X → A è implicata da F, basta vedere se A appartiene a X_F^+, a patto di saper calcolare X_F^+.

Esiste un algoritmo per il calcolo di X_F^+:
Input: un insieme X di attributi e un insieme F di dipendenze.
Output: un insieme X_P di attributi.

1. Inizializziamo X_P con l'insieme di input X.
2. Esaminiamo le dipendenze in F; se esiste una dipendenza $Y \rightarrow A$ con $Y \subseteq X_P$ e $A \notin X_P$ allora aggiungiamo A a X_P.
3. Ripetiamo il passo 2 fino al momento in cui non vi sono ulteriori attributi che possono essere aggiunti a X_P.

   Il concetto di chiusura X_F^+ è utile anche per formalizzare il legame fra il concetto di dipendenza funzionale e quello di chiave:

   > **Def**
   > un insieme di attributi K è chiave per uno schema di relazione R(U) su cui è definito un insieme di dipendenze funzionali F se F implica $K \rightarrow U$. Di conseguenza, l'algoritmo mostrato può essere utilizzato per verificare se un insieme è chiave.

Un insieme F è:

- *non ridondante* se non esiste dipendenza $f \in F$ tale che $F - \{f\}$ implica f;
- *ridotto* se è non ridondante e non esiste un insieme F' equivalente a F ottenuto eliminando attributi dai primi membri di una o più dipendenze di F.
  Esempi:
- $F_1 = \{A \rightarrow B, AB \rightarrow C, A \rightarrow C\}$
- $F_2 = \{A \rightarrow B, AB \rightarrow C\}$
- $F_3 = \{A \rightarrow B, A \rightarrow C\}$
  $\Rightarrow F_1$ è ridondante, perché $\{A \rightarrow B, AB \rightarrow C\}$ implica $A \rightarrow C$; F_1 è equivalente a F_2;
  $\Rightarrow F_2$ è non ridondante ma non è ridotto, perché B può essere eliminato dal primo membro della seconda dipendenza: F_2 è equivalente a F_3;
  $\Rightarrow F_3$ è ridotto.
  Due insiemi di dipendenze funzionali F_1 e F_2 sono equivalenti se F_1 implica ogni dipendenza in F_2 e viceversa.
  Se due insiemi sono equivalenti diciamo che sono uno copertura del altro.

Calcolare la copertura minima

Per trovare una copertura minima non ridondante esaminiamo ripetutamente le dipendenze dell'insieme dato, eliminando quelle implicate da altre, si procede in tre passi:

1. sostituiamo l'insieme dato con quello equivalente che ha tutti i secondi membri costituiti da singoli attributi;
2. eliminiamo le dipendenze ridondanti;
3. per ogni dipendenza verifichiamo se esistono attributi eliminabili dal primo membro: se F è l'insieme corrente, per ogni dipendenza $Y \rightarrow A \in F$, verifichiamo se esiste $Y \subseteq X$ tale che F è equivalente a $F - \{X \rightarrow A\} \cup \{Y \rightarrow A\}$.

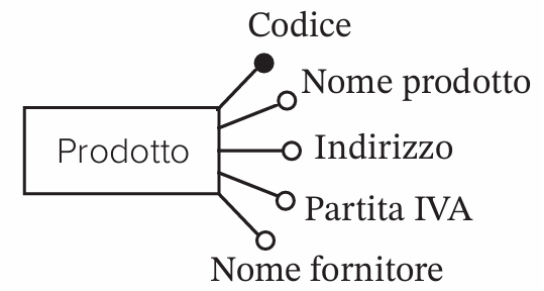
Algoritmo di sintesi per la terza forma normale

Def

Uno schema di relazione R(U) con l'insieme di dipendenze F è in *terza forma normale* se, per ogni dipendenza funzionale (non banale) $X \rightarrow A \in F$, almeno una delle seguenti condizioni è verificata:

- X contiene una chiave K di *r*: cioè $X_F^+ = U$;
- A è contenuto in almeno una chiave di *r*: esiste un insieme di attributi $K \subseteq U$ tale che $K_F^+ = U$ e $(K - A)_F^+ \subset U$.

Verifiche di normalizzazione sulle entità



- Partita IVA → NomeFornitore, Indirizzo
  Tutti gli attributi dipendono funzionalmente da Codice, ovvero l'identificatore di Prodotto.
  ⇒ l'entità viola la terza forma normale perché la dipendenza *Partita IVA → NomeFornitore, Indirizzo* ha un primo membro che non contiene l'identificatore e un secondo membro composto da attributi che non fanno parte della chiave.
  Decomposizione: