# Descrizione dei principi relativi alla normalizzazione dei database

## Sommario

In questo articolo viene spiegata la terminologia relativa alla normalizzazione dei database per gli utenti inesperti. La conoscenza di questa terminologia può rivelarsi utile nel discutere la struttura di un database relazionale.   
  
**Definizione di normalizzazione**

Il termine normalizzazione indica il processo di organizzazione dei dati in un database. Tale processo comprende la creazione di tabelle e la definizione di relazioni tra di esse sulla base di regole progettate in modo da proteggere i dati e rendere il database più flessibile mediante l'eliminazione della ridondanza e delle dipendenze incoerenti.   
  
La presenza di dati ridondanti comporta uno spreco di spazio su disco nonché problemi di manutenzione. Se è necessario modificare dati presenti in più posizioni, la modifica deve essere effettuata ovunque seguendo le stesse modalità. Ad esempio, è più agevole implementare la modifica relativa all'indirizzo di un cliente se i dati relativi sono memorizzati solo nella tabella Clienti.   
  
Che cos'è una "dipendenza incoerente"? Mentre è intuitivo per un utente cercare nella tabella Clienti l'indirizzo di un cliente specifico, può non avere senso cercare in tale tabella le informazioni relative allo stipendio del dipendente che si occupa di quel cliente. Le informazioni sullo stipendio sono correlate al dipendente, ovvero sono dipendenti da quest'ultimo, pertanto devono essere spostate nella tabella Dipendenti. Le dipendenze incoerenti possono rendere difficile l'accesso ai dati, in quanto il percorso per la ricerca dei dati può risultare mancante o danneggiato.   
  
Per la normalizzazione dei database è necessario seguire alcune regole, ciascuna delle quali viene definita "forma normale". Se si osserva la prima regola, il database viene considerato nella "prima forma normale". Se si osservano le prime tre regole, il database viene considerato nella "terza forma normale". Sebbene siano possibili altri livelli di normalizzazione, la terza forma normale è considerata il livello massimo necessario per la maggior parte delle applicazioni.   
  
Come per molte regole e specifiche formali, gli scenari reali non consentono sempre la conformità totale. In generale poiché la normalizzazione richiede l'uso di tabelle aggiuntive, viene considerata troppo dispendiosa da alcuni clienti. Se si decide di violare una delle prime tre regole di normalizzazione, assicurarsi che l'applicazione sia in grado di prevenire i problemi che possono derivarne, quali la ridondanza dei dati e le dipendenze incoerenti.   
  
Nelle descrizioni che seguono sono inclusi alcuni esempi.

### Prima forma normale

**Definizione**: Si dice che una base dati è in 1NF (prima forma normale) se vale la seguente relazione per ogni relazione contenuta nella base dati. Una relazione è in 1NF se e solo se:

1. non presenta gruppi di attributi che si ripetono (ossia ciascun attributo è definito su un dominio con valori atomici).
2. esiste una chiave primaria (ossia esiste un insieme di attributi, che identifica in modo univoco ogni tupla della relazione)

Non utilizzare campi multipli in una singola tabella per memorizzare dati simili. Ad esempio, per controllare una voce di inventario proveniente da due possibili origini, è possibile creare un record di inventario contenente i campi per Codice fornitore 1 e Codice fornitore 2.   
  
Che cosa succede quando si aggiunge un terzo fornitore? La soluzione non consiste nell'aggiungere un campo. Sono infatti necessarie modifiche al programma e alla tabella che tuttavia non permettono di inserire agevolmente un numero dinamico di fornitori. È invece opportuno inserire tutte le informazioni sui fornitori in un'altra tabella denominata Fornitori e quindi collegare l'inventario ai fornitori tramite una chiave basata sul numero di voce oppure i fornitori all'inventario tramite una chiave basata sul codice fornitore.

**Esempi:**

* Il seguente esempio viola la 1NF, perché pur esistendo una chiave primaria ({Matricola,Materia}), l'attributo Voto non è definito su un dominio con valori atomici:

|  |  |  |  |
| --- | --- | --- | --- |
| Matricola | Studente | Materia | SemestreVoto |
| 0000-000-01 | Pietro | Basi di Dati | 1 sem, B; 2 sem, F |
| 0000-000-02 | Pietro | Basi di Dati | 1 sem, A; 2 sem, A |
| 0000-000-03 | Sara | Basi di Dati | 1 sem, B; 2 sem, A |

È necessario ristrutturare la relazione come segue:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Matricola | Studente | Materia | Semestre | Voto |
| 0000-000-01 | Pietro | Basi di Dati | 1 | B |
| 0000-000-01 | Pietro | Basi di Dati | 2 | F |
| 0000-000-02 | Pietro | Basi di Dati | 1 | A |
| 0000-000-02 | Pietro | Basi di Dati | 2 | A |
| 0000-000-03 | Sara | Basi di Dati | 1 | B |
| 0000-000-03 | Sara | Basi di Dati | 2 | A |

* Il seguente esempio viola la 1NF, perché pur non presentando gruppi di attributi che si ripetono, manca una chiave primaria, rendendo impossibile distinguere studentesse e studenti con lo stesso nome:

|  |  |  |  |
| --- | --- | --- | --- |
| Studente | Materia | Semestre | Voto |
| Pietro | Basi di Dati | 1 | A |
| Pietro | Basi di Dati | 2 | A |
| Pietro | Basi di Dati | 1 | A |
| Pietro | Basi di Dati | 2 | A |
| Sara | Basi di Dati | 1 | B |
| Sara | Basi di Dati | 2 | A |

È necessario ristrutturare la relazione come segue:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Matricola | Studente | Materia | Semestre | Voto |
| nnnn-nnn-nn | Pietro | Basi di Dati | 1 | B |
| … | Pietro | Basi di Dati | 2 | F |
| … | Pietro | Basi di Dati | 1 | A |
| … | Pietro | Basi di Dati | 2 | A |
| … | Sara | Basi di Dati | 1 | B |
| … | Sara | Basi di Dati | 2 | A |

### Seconda forma normale

**Definizione**: Una base dati è invece in 2NF (seconda forma normale) quando è in 1NF e per ogni tabella tutti i campi non chiave dipendono funzionalmente dall'intera chiave composta e non da una parte di essa.

**Esempi:**

* Come esempio supponiamo di avere una tabella con gli esami sostenuti dagli studenti universitari. I campi di interesse potrebbero quindi essere i seguenti:

"Codice corso di laurea"

"Codice esame"

"Matricola studente"

"Voto conseguito"

"Data superamento"

La tabella avrà quindi la seguente intestazione:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id\_corso\_laurea** | **id\_esame** | **id\_studente** | voto | data |

La chiave candidata (tale terminologia è riservata alle superchiavi minimali, anche dette semplicemente chiavi) è rappresentata dalla tripla evidenziata in grassetto, ossia da:

"Codice corso di laurea"

"Codice esame"

"Matricola studente"

Essa infatti risulta essere l'insieme di chiavi minimale per poter identificare in modo univoco le tuple (i record) della tabella.

I campi "Voto conseguito" e "Data superamento", invece, sono campi non chiave, e fanno riferimento all'intera superchiave.

Difatti, se dipendessero solo da:

"Codice corso di laurea" si perderebbero le informazioni relative allo studente e all'esame superato

"Codice esame" si perderebbero le informazioni relative allo studente ed al corso di laurea a cui l'esame appartiene

"Matricola studente" si perderebbero le informazioni relative all'esame superato e al corso di laurea a cui lo studente è iscritto.

"Codice corso di laurea", "Codice esame" si perderebbero le informazioni relative allo studente che ha superato l'esame

"Codice corso di laurea", "Matricola studente" si perderebbero le informazioni relative all'esame superato

"Matricola studente", "Codice esame" si perderebbero le informazioni relative al Corso di Laurea di appartenenza.

### Terza forma normale

Definizione: Una base dati è in 3NF (terza forma normale) se è in 2NF e tutti gli attributi non-chiave dipendono dalla chiave soltanto, ossia non esistono attributi che dipendono da altri attributi non-chiave.

I valori di un record non compresi nella relativa chiave non appartengono alla tabella. In generale, ogni volta che il contenuto di un gruppo di campi può essere applicabile a più record della tabella, provare a inserire tali campi in una tabella separata.   
  
Ad esempio, in una tabella Colloqui di assunzione è possibile includere il nome e l'indirizzo dell'università dei candidati. Per i gruppi di distribuzione è tuttavia necessario disporre di un elenco completo di università. Se le informazioni sull'università sono memorizzate nella tabella Candidati, non sarà possibile ottenere l'elenco delle università non associate ai candidati correnti. Creare una tabella Università separata e collegarla alla tabella Candidati utilizzando una chiave basata sul codice università.   
  
ECCEZIONE: l'adozione della terza forma normale non è sempre pratica anche se teoricamente auspicabile. Se si dispone di una tabella Clienti e si desidera eliminare tutte le possibili dipendenze tra campi, è necessario creare tabelle separate per città, CAP, rappresentanti, classi di clienti e gli eventuali altri fattori che possono essere duplicati in più record. Nella teoria la normalizzazione è sempre auspicabile, tuttavia l'utilizzo di un numero elevato di tabelle di dimensioni limitate può determinare una riduzione del livello delle prestazioni oppure richiedere capacità di memoria e di apertura dei file superiori a quelle disponibili.   
  
Può risultare più appropriato applicare la terza forma normale solo ai dati soggetti a frequenti modifiche. Se sussistono dei campi dipendenti, progettare l'applicazione in modo da richiedere la verifica di tutti i campi correlati in caso di modifica di un campo.

**Esempi:**

Supponiamo di avere una base dati che associ il codice fiscale dell'iscritto al corso frequentato ed all'insegnante di riferimento. Si supponga che il nostro DB abbia un'unica chiave primaria ("Codice Fiscale") e sia così strutturato:

|  |  |  |
| --- | --- | --- |
| Codice Fiscale | Codice Corso | Insegnante |
| LBRRSS79Y12T344A | BB01 | Marco |
| GNNBNCT84A11L611B | BB01 | Marco |
| LBRMNN79E64A112A | BB01 | Marco |
| GLSTMT59U66P109B | AE02 | Federica |

Il nostro database non è certamente 3NF in quanto il campo "insegnante" non dipende dalla chiave primaria ma dal campo "Codice Corso" (che non è chiave). Per normalizzare il nostro DB in 3NF dovremo scomporlo in due tabelle:

|  |  |
| --- | --- |
| Codice Fiscale | Codice Corso |
| LBRRSS79Y12T344A | BB01 |
| GNNBNCT84A11L611B | BB01 |
| LBRMNN79E64A112A | BB01 |
| GLSTMT59U66P109B | AE02 |

e

|  |  |
| --- | --- |
| Codice Corso | Insegnante |
| BB01 | Marco |
| AE02 | Federica |

### Altre forme di normalizzazione

Sono inoltre disponibili una quarta forma normale, denominata Boyce Codd Normal Form (BCNF), e una quinta forma anche se vengono raramente prese in considerazione nella progettazione pratica. Il mancato utilizzo di queste regole può non consentire una perfetta progettazione del database, senza tuttavia influire negativamente sulla funzionalità.

### Normalizzazione di una tabella di esempio

La procedura descritta di seguito illustra il processo di normalizzazione di una tabella Students fittizia.

**Tabella non normalizzata:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Student#** | **Advisor** | **Adv-Room** | **Class1** | **Class2** | **Class3** |
| 1022 | Jones | 412 | 101-07 | 143-01 | 159-02 |
| 4123 | Smith | 216 | 201-01 | 211-02 | 214-01 |

1. **Prima forma normale: nessun gruppo ripetuto**  
   Le tabelle dovrebbero presentare solo due dimensioni. Poiché a un singolo studente sono associate più classi, è necessario elencare tali classi in una tabella diversa. I campi Class1, Class2 e Class3 nel record sopra riportato sono dunque indice di problemi di progettazione.   
     
   Nei fogli di calcolo viene spesso utilizzata la terza dimensione, ma non nelle tabelle. Un'altra prospettiva da cui osservare questo problema è l'utilizzo di una relazione uno-a-molti. In questo caso non inserire entrambi i lati della relazione nella stessa tabella, ma creare un'altra tabella utilizzando la prima forma normale ed eliminando il gruppo ripetuto (Class#), come illustrato di seguito:

|  |  |  |  |
| --- | --- | --- | --- |
| **Student#** | **Advisor** | **Adv-Room** | **Class#** |
| 1022 | Jones | 412 | 101-07 |
| 1022 | Jones | 412 | 143-01 |
| 1022 | Jones | 412 | 159-02 |
| 4123 | Smith | 216 | 201-01 |
| 4123 | Smith | 216 | 211-02 |
| 4123 | Smith | 216 | 214-01 |

1. **Seconda forma normale: eliminazione di dati ridondanti**  
   Considerare i valori multipli di Class# per ciascun valore Student# nella tabella sopra riportata. Class# non è funzionalmente dipendente da Student# (chiave primaria), pertanto questa relazione non è compresa nella seconda forma normale.  
     
   Le due tabelle che seguono illustrano la seconda forma normale:   
     
   Students:

|  |  |  |
| --- | --- | --- |
| **Student#** | **Advisor** | **Adv-Room** |
| 1022 | Jones | 412 |
| 4123 | Smith | 216 |

Registration:

|  |  |
| --- | --- |
| **Student#** | **Class#** |
| 1022 | 101-07 |
| 1022 | 143-01 |
| 1022 | 159-02 |
| 4123 | 201-01 |
| 4123 | 211-02 |
| 4123 | 214-01 |

1. **Terza forma normale: eliminazione di dati non dipendenti da chiave**  
   Nell'ultimo esempio, Adv-Room (il numero dell'ufficio del tutor) non è funzionalmente dipendente dall'attributo Advisor. Per risolvere il problema è possibile spostare tale attributo dalla tabella Students alla tabella Faculty, come illustrato di seguito:  
     
   Students:

|  |  |
| --- | --- |
| **Student#** | **Advisor** |
| 1022 | Jones |
| 4123 | Smith |

Faculty:

|  |  |
| --- | --- |
| **Name** | **Room** |
| Jones | 412 |
| Smith | 216 |