

# CAP2\_MODELLO\_RELAZIONALE

## Table of contents

- [Modelli logici dei dati](#)
  1. [Modello relazionale](#)
    1. [Relazione matematica](#)
    2. [Tabelle e relazioni](#)
- [Vincoli d'integrità](#)
  1. [Intrarelazionali](#)
    1. [Vincoli di  \$n\$ -upla](#)
    2. [Vincoli su valori \(o dominio\)](#)
    3. [Chiave](#)
      1. [Chiave primaria](#)
  2. [Interrelazionali](#)
    1. [Integrità referenziale \(chiave esterna, foreign key\)](#)

## Modelli logici dei dati

3 modelli logici tradizionali:

- gerarchico, con puntatori come un albero
- reticolare, un grafo
- **relazionale**, che è ancora oggi il più utilizzato, basato sui valori della realtà che noi vogliamo modellare

## Modello relazionale

---

Creato ('70) per favorire l'indipendenza dei dati rispetto la rappresentazione e ha impiegato tempo per essere adottato ('80).

Si dice relazionale perché legato al concetto matematico (non strettamente), le relazioni hanno rappresentazione tramite tabelle:

- relazione matematica
- relazione secondo modello relazionale dei dati
- **relationship**, due entità hanno un qualche collegamento, termine di riferito agli schemi ER (la chiameremo 'associazione' per evitare confusione)

L'utilità della relazione per valori è nella facilità dei collegamenti logici, rispetto a quella dei puntatori dove la confusione è facile comparire.

### Schema di relazione

un nome  $R$  con un insieme di attributi  $A_n$ :

$$R(A_1, \dots, A_n)$$

## Schema di base di dati

insieme/lista di schemi di relazione:

$$R = \{R_1(X_1), \dots, R_k(X_k)\}$$

**Ennupla**  $\rightarrow$   $n$ -upla

su insieme di attributi  $X$ , è una funzione che associa a ciascun attributo  $A$  in  $X$  un valore nel dominio di  $A$ , e  $t[A]$  denota il valore di  $t$  su  $A$ .

## Base di dati

insieme di relazioni:

$$r = \{r_1, \dots, r_n\}$$

### ≡ Esempio relazione su unico attributo

matricola	/
6554	/
3456	/

### ≡ Esempio di struttura nidificata

Le strutture nidificate non sono consentite nel modello ER

numero	data	totale	quantità	descrizione
1235	12/10/2002	39,20	3	coperti
			2	antipasti
			3	primi

vengono piuttosto separate in 2 tabelle

numero	data	totale
...	...	...

numero	quantità	descrizione
...	...	...

## Relazione matematica

$$D_1 = \{a, b\}$$

$$D_2 = \{x, y, z\}$$

Il prodotto cartesiano sarebbe:

$$D_1 * D_2$$

Una relazione:

$$r \subseteq D_1 * D_2$$

### ≡ Esempio di tabella con nome "Partite"

`partite ⊆ string * string * int * int`

casa	fuori	reticasa	retifuori
Juve	Lazio	3	1
Lazio	Milan	2	0
Juve	Roma	0	2
Roma	Milan	0	1

La  $n$ -upla della tabella non la pensiamo come valore massimo  $\infty$ .

Alcune proprietà:

- non c'è ordinamento tra le  $n$ -uple
- le  $n$ -uple sono distinte
- ciascuna  $n$ -upla è ordinata
- la struttura è posizionale

## Tabelle e relazioni

Siccome la struttura posizionale non ci è comoda, associamo un nome unico alla tabella (*attributo*) che ne descrive il "ruolo" ('casa', 'fuori', 'reticasa', 'retifuori').

Una *tabella* rappresenta una relazione (nel modello logico relazionale teorico) dove:

- le righe sono diverse tra loro
- le intestazioni delle colonne sono diverse tra loro
- i valori di ogni colonna sono tra loro omogenei, sono valori del dominio (un numero non è una stringa)

## Vincoli d'integrità

Un *vincolo d'integrità* deve essere una proprietà di tutte le basi di dati, che deve essere rispettata. La base di dati viene presa per il suo intero e verificato che il vincolo restituisca VERO, ovvero sia corretta.

Il compito del DBMS è quello di fare controlli in maniera più o meno efficiente, perché controllare tutto il DB è lento.

Utilità:

1. niente spazzatura nella base di dati, ho dati di qualità più alta
2. sono effettivamente utili per il DBMS per eseguire interrogazioni in maniera efficiente
3. utili nella progettazione

I vincoli corrispondono a proprietà del mondo reale modellato dalla base di dati. A uno schema associamo un insieme di vincoli e consideriamo *corrette* le istanze che soddisfano tutti i vincoli.

## Intrarelazionali

Il vincolo riguarda **una sola** tabella/relazione e mi è sufficiente per verificare la veridicità del DB. I due vincoli non sono molto separati per quanto teoria, nei DBMS non c'è molta distinzione.

### Vincoli di $n$ -upla

Controllo ogni singola  $n$ -upla. Indipendente una dalle altre.

Ci dobbiamo immaginare tutti i valori possibili per il nostro dato: situazioni temporali, situazioni indefinite devono avere un comportamento da noi voluto.

☰ Controllo se il voto è maggiore o uguale a 18 e sotto il 30.

```
(Voto ≥ 18) AND (Voto ≤ 30)
(Voto = 30) AND NOT (Lode = "e lode")
```

### Vincoli su valori (o dominio)

Controllo il valore.

La distinzione fra vincoli di  $n$ -upla e di dominio e' sempre piu' sottile nei DBMS recenti, quindi d'ora in avanti tutti i vincoli che prenderemo in considerazione saranno di  $n$ -upla.

## Chiave

Una **chiave** possiamo identificarla come un insieme di attributi per singola tabella/relazione, univoca, identificanti le  $n$ -uple di una relazione.

Chiamiamo questo insieme di attributi  $K$ .

Si chiama **superchiave** per  $r$  se  $r$  non contiene due  $n$ -uple distinte  $t_1$  e  $t_2$  con  $t_1[K] = t_2[K]$

☰ Esempio di chiave

<u>Matricola</u>	Congome	Nome	Corso	Nascita
------------------	---------	------	-------	---------

- Non esistono due persone con lo stesso numero **Matricola**, quindi questa sarà la nostra chiave.
- **Congome**, **Nome**, **Nascita** potrebbe essere una chiave fintanto che non esista una persona che ha tutti e quanti gli stessi valori:
  - è superchiave
  - minimale

L'esistenza delle chiavi garantisce l'accessibilità a ciascun dato della base di dati; le chiavi permettono di correlare i dati in relazioni diverse (modello relazionale basato su valori).

Nel caso di valori **NULL**, impedisce di usare chiavi, quindi da ricordare che una chiave non può avere questo valore.

### Chiave primaria

Sulla quale non sono MAI ammessi valori nulli, su nessun attributo componente la **chiave primaria** possiamo consentire il valore nullo.

La sottolineatura identifica questa chiave.

Se piu' attributi sono sottolineati, insieme formano una chiave.

### ≡ Esempio

La `Matricola` e il `CodiceFiscale` possono fare chiave, pero' sara' primaria soltanto `Matricola` siccome uno studente potrebbe venire dall'estero e non avere il `CodiceFiscale`.

## Interrelazionali

Guardiamo *diverse* tabelle per verificare la veridicità.

### Integrità referenziale (chiave esterna, foreign key)

Quel vincolo che serve per dire che da *questa* tabella, scrivo un valore contenente in *un'altra* tabella.

- informazioni in relazioni diverse sono correlate attraverso valori comuni
- in particolare, valori delle chiavi (primarie), usiamo quasi sempre quelle
- le correlazioni debbono essere "coerenti"

La dicitura  $f_k$  identifica questa chiave.

### ≡ Esempio

`Infrazioni`

Codice	Data	Vigile	Prov	Numero
--------	------	--------	------	--------

`Vigili`

Matricola	Cognome	Nome
-----------	---------	------

Il valore dell'attributo `Vigile` in tabella `Infrazioni`, deve essere un valore contenuto in tabella `Vigili`. Quindi c'è un vincolo di chiave esterna che lega `Vigile` → `Matricola`.

Un vincolo di **integrità referenziale (foreign key)** fra gli attributi  $X$  di una relazione  $R_1$  e un'altra relazione  $R_2$  impone ai valori su  $X$  in  $R_1$  di comparire come valori della chiave primaria di  $R_2$ .