# Algebra relazionale

L'algebra relazionale è un linguaggio formale utilizzato nel contesto dei database relazionali per manipolare e interrogare dati.

È stata introdotta da Edgar F. Codd, il quale ha definito una serie di operazioni algebriche che possono essere eseguite su insiemi di dati organizzati in tabelle (relazioni).

L'obiettivo principale è fornire un mezzo dichiarativo per specificare le operazioni da eseguire sui dati, senza specificare come effettivamente devono essere eseguite.

Le operazioni principali dell'algebra relazionale includono:

1. Selezione (σ): Estrae righe dalla tabella che soddisfano una determinata condizione.
2. Proiezione (π): Seleziona colonne specifiche dalla tabella.
3. Unione (∪): Combina due tabelle mantenendo tutte le righe, eliminando duplicati.
4. Intersezione (∩): Restituisce le righe presenti in entrambe le tabelle.
5. Differenza (-): Restituisce le righe presenti in una tabella ma non nell'altra.
6. Prodotto cartesiano (×): Combina tutte le righe di una tabella con tutte le righe di un'altra tabella.
7. Join (⨝): Combina le righe di due tabelle in base a una condizione specificata.
8. Division (÷): Trova tutte le tuple nella prima tabella che sono correlate a tutte le tuple nella seconda tabella.

Queste operazioni formano il nucleo dell'algebra relazionale e forniscono un modo rigoroso per manipolare e combinare dati all'interno di database relazionali.

L'utilizzo di un linguaggio dichiarativo come l'algebra relazionale consente agli utenti di concentrarsi sulla specifica di cosa desiderano ottenere, senza dover preoccuparsi dell'implementazione dettagliata delle operazioni.

Consideriamo due tabelle di esempio:

Tabella "Studenti":

| **ID** | **Nome** | **Corso** |
| --- | --- | --- |
| 1 | Mario | Mat |
| 2 | Anna | Fis |
| 3 | Carlo | Mat |
| 4 | Elena | Fis |

Tabella "Corsi":

| **Corso** | **Docente** |
| --- | --- |
| Mat | Rossi |
| Fis | Bianchi |
| Storia | Verdi |

Ora vediamo esempi di operazioni dell'algebra relazionale:

1. Selezione (σ):
   * Trova tutti gli studenti che seguono il corso di Matematica: [ \sigma\_{Corso='Mat'}(Studenti) ] Risultato:

| **ID** | **Nome** | **Corso** |
| --- | --- | --- |
| 1 | Mario | Mat |
| 3 | Carlo | Mat |

1. Proiezione (π):
   * Seleziona solo il nome degli studenti: [ \pi\_{Nome}(Studenti) ] Risultato:

| **Nome** |
| --- |
| Mario |
| Anna |
| Carlo |
| Elena |

1. Unione (∪):
   * Combina gli studenti di Matematica e Fisica: [ Studenti \cup Studenti\_{Fis} ] Risultato:

| **ID** | **Nome** | **Corso** |
| --- | --- | --- |
| 1 | Mario | Mat |
| 2 | Anna | Fis |
| 3 | Carlo | Mat |
| 4 | Elena | Fis |

1. Intersezione (∩):
   * Trova gli studenti che seguono entrambi i corsi di Matematica e Fisica: [ Studenti\_{Mat} \cap Studenti\_{Fis} ] Risultato:

| **ID** | **Nome** | **Corso** |
| --- | --- | --- |
| 3 | Carlo | Mat |

1. Differenza (-):
   * Trova gli studenti che seguono Matematica ma non Fisica: [ Studenti\_{Mat} - Studenti\_{Fis} ] Risultato:

| **ID** | **Nome** | **Corso** |
| --- | --- | --- |
| 1 | Mario | Mat |
| 3 | Carlo | Mat |

1. Prodotto Cartesiano (×):
   * Crea tutte le possibili coppie tra studenti e corsi: [ Studenti \times Corsi ] Risultato:

| **ID** | **Nome** | **Corso** | **Docente** |
| --- | --- | --- | --- |
| 1 | Mario | Mat | Rossi |
| 1 | Mario | Fis | Bianchi |
| 1 | Mario | Storia | Verdi |
| 2 | Anna | Mat | Rossi |
| 2 | Anna | Fis | Bianchi |
| 2 | Anna | Storia | Verdi |
| ... | ... | ... | ... |

1. Join (⨝):
   * Combina gli studenti e i corsi in base al corso: [ Studenti \bowtie\_{Studenti.Corso=Corsi.Corso} Corsi ] Risultato:

| **ID** | **Nome** | **Corso** | **Docente** |
| --- | --- | --- | --- |
| 1 | Mario | Mat | Rossi |
| 2 | Anna | Fis | Bianchi |
| 3 | Carlo | Mat | Rossi |
| 4 | Elena | Fis | Bianchi |

1. Division (÷):
   * Trova gli studenti che seguono tutti i corsi: [ Studenti \div Corsi ] Risultato:

| **ID** | **Nome** |
| --- | --- |
| 1 | Mario |
| 2 | Anna |