Introduzione al linguaggio SQL

Giorgio Bar – giorgio.bar@to.infn.it

Scuola di Specializzazione in Fisica Medica – Università di Torino

Anno accademico 2023/2024 (anno solare 2024/2025)

SQL: Structured Query Language

- Linguaggio di riferimento per le basi di dati relazionali
- Un linguaggio a livello di set.
  - Gli operatori operano su relazioni, il risultato è sempre una relazione
- Un linguaggio dichiarativo
  - Si specifica l'obiettivo e non il modo in cui ottenerlo, ad un livello di astrazione superiore rispetto ai linguaggi di programmazione procedurali come l'algebra relazionale.
- Le istruzioni del linguaggio SQL possono essere suddivise in
  - DDL (Data Definition Language)
  - DML (Data Manipulation Language)

Data Definition Language

- Definizione dello schema di una base di dati relazionale: creazione, modifica e cancellazione di tabelle e viste (tabelle virtuali il cui contenuto è ottenuto a partire da altre tabelle)
  - CREATE, ALTER, DROP TABLE / VIEW
- Definizione di strutture dati accessorie per recuperare in modo efficiente i dati.
  - CREATE, DROP INDEX
- Definizione dei privilegi di accesso alle risorse concessi utenti.
  - GRANT, REVOKE
- Definizione di transazioni (sequenza di operazioni di modifica che porta la base dati da uno stato consistente a un altro stato consistente)
  - COMMIT, ROLLBACK

Data Manipulation Language

- Interrogazione della base di dati per estrarre i dati di interesse
  - SELECT
- Inserimento di nuove informazioni
  - INSERT
- Aggiornamento di dati
  - UPDATE
- Cancellazione di dati
  - DELETE

Istruzione SELECT: fondamenti

Base dati di esempio

PRODOTTI (codP, NomeP, Colore, Peso, Magazzino)

CodP	NomeP	Colore	Peso	Magazzino
P1	Trion	Rosso	40	Torino
P2	Speed	Giallo	48	Milano
P3	Airtech	Blu	48	Roma

FORNITORI (codF, NomeF, Rating, Sede)

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano

ORDINI (codF, codP, Qta)

CodP	CodF	Qta
P1	F2	300
P2	F2	400
P2	F3	200

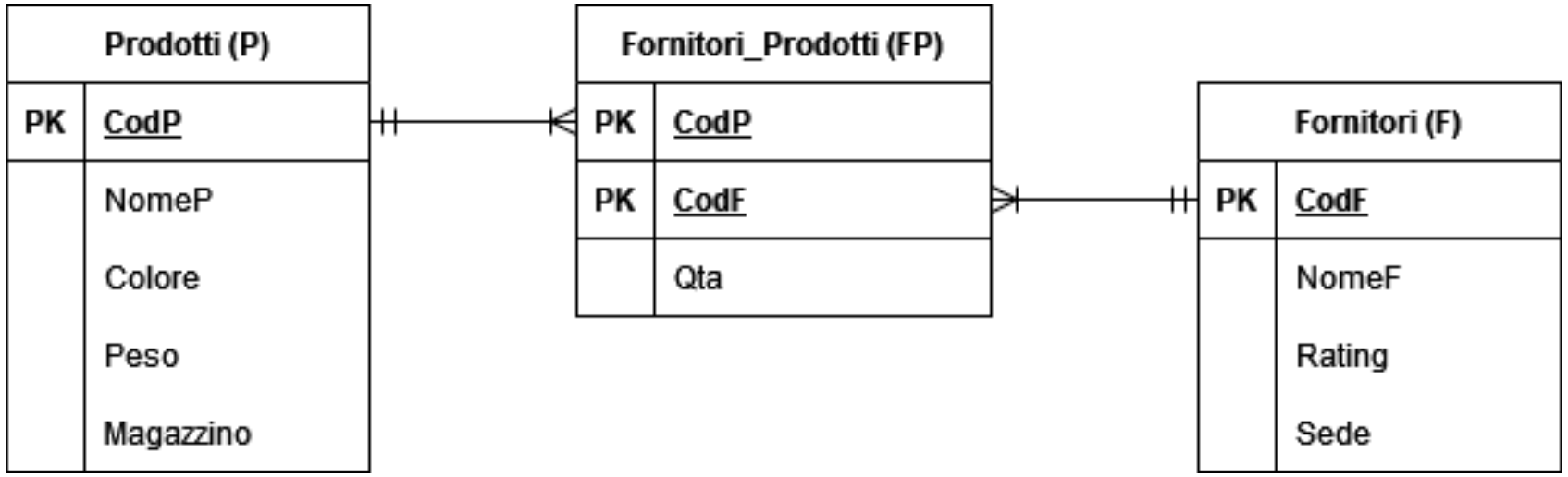
La tabella ORDINI mette in relazione i prodotti con i fornitori che li forniscono

Base dati di esempio

PRODOTTI (codP, NomeP, Colore, Peso, Magazzino)

FORNITORI (codF, NomeF, Rating, Sede)

ORDINI (codF, codP, Qta)



Struttura dell'istruzione SELECT

```
SELECT [DISTINCT] lista_attributi
FROM elenco_tabelle
[ WHERE condizioni_di_tupla ]
[ GROUP BY attributi_di_raggruppamento ]
[ HAVING condizioni_su_aggregati ]
[ ORDER BY attributi_di_ordinamento ];
```

SELECT base

```
SELECT lista_attributi
FROM elenco_tabelle;
```

- Corrisponde ad un'operazione di proiezione in algebra relazionale, *lista_attributi* definisce le colonne nel risultato dell'interrogazione. Ciascuna colonna può essere rinominata (clausola AS).
- *lista_attributi* può assumere il valore * che permette di recuperare tutti gli attributi della tabella specificata nella clausola FROM.

```
SELECT CodP, NomeP AS Nome_Prodotto
FROM PRODOTTI;
```

```
SELECT *
FROM PRODOTTI;
```

SELECT base

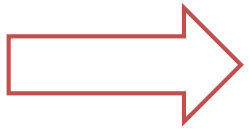
- Trovare il codice di tutti i prodotti

```
SELECT CodP
FROM PRODOTTI;
```

```
SELECT PRODOTTI.CodP
FROM PRODOTTI;
```

```
SELECT P.CodP
FROM PRODOTTI P;
```

CodP	NomeP	Colore	Peso	Magazzino
P1	Trion	Rosso	40	Torino
P2	Speed	Giallo	48	Milano
P3	Airtech	Blu	48	Roma
P4	Airtech	Blu	44	Torino
P5	Futura	Blu	40	Milano
P6	Zen	Rosso	42	Torino



CodP
P1
P2
P3
P4
P5
P6

SELECT base

- Trovare il codice dei prodotti forniti da almeno un fornitore

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT CodP
FROM ORDINI O;
```

CodP
P1
P2
P3
P4
P5
P6
P1
P2
P2
P3
P4
P5

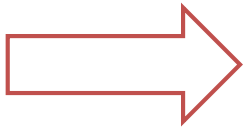
La clausola SELECT, a differenza della proiezione in algebra relazionale, non effettua la rimozione dei duplicati

SELECT DISTINCT

- Trovare il codice dei prodotti forniti da almeno un fornitore

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT DISTINCT CodP
FROM ORDINI O;
```



CodP
P1
P2
P3
P4
P5
P6

Rimozione dei valori duplicati

Struttura dell'istruzione SELECT

SELECT [DISTINCT] *lista_attributi*
FROM *elenco_tabelle*
[WHERE *condizioni_di_tupla*]
[GROUP BY *attributi_di_raggruppamento*]
[HAVING *condizioni_su_aggregati*]
[ORDER BY *attributi_di_ordinamento*];

Clausola WHERE

- Permette di esprimere condizioni di selezione che devono essere applicate singolarmente ad ogni tupla.
- Consiste in una espressione booleana ottenuta combinando predicati semplici di confronto tra attributi e costanti con operatori AND, OR e NOT.

Operator	Description
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
=	equal
<> or !=	not equal

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano
F4	Prometeo	2	Torino
F5	Ceo	3	Venezia

```
SELECT CodF
FROM FORNITORI
WHERE Sede = 'Milano';
```

https://www.postgresql.org/docs/10/functions-comparison.html

Clausola WHERE

- Trovare il codice dei fornitori di Milano

```
SELECT CodF
FROM FORNITORI
WHERE Sede = 'Milano';
```

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano
F4	Prometeo	2	Torino
F5	Ceo	3	Venezia

CodF
F2
F3

Clausola WHERE

- Trovare il codice e il rating dei fornitori che non hanno sede a Milano

```
SELECT CodF, Rating
FROM FORNITORI
WHERE Sede <> 'Milano';
```

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano
F4	Prometeo	2	Torino
F5	Ceo	3	Venezia

CodF	Rating
F1	2
F4	2
F5	3

Espressioni booleane

- Trovare il codice dei fornitori di Milano con un rating superiore a 2

```
SELECT CodF
FROM FORNITORI
WHERE Sede = 'Milano' AND Rating > 2;
```

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano
F4	Prometeo	2	Torino
F5	Ceo	3	Venezia

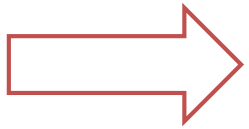
CodF
F3

Espressioni booleane

- Trovare il codice e il rating dei fornitori di Milano o di Torino

```
SELECT CodF, Rating
FROM FORNITORI
WHERE Sede = 'Milano' OR Sede = 'Torino';
```

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano
F4	Prometeo	2	Torino
F5	Ceo	3	Venezia



CodF	Rating
F1	2
F2	1
F3	3
F4	2

Gestione dei valori NULL

- Trovare il codice e il nome dei prodotti con peso minore di 44

```
SELECT CodP, NomeP, Peso
FROM PRODOTTI
WHERE Peso < 44;
```

Le tuple per cui il peso è NULL (valori non noti o non applicabili) non sono selezionate

CodP	NomeP	Colore	Peso	Magazzino
P1	Trion	Rosso	40	Torino
P2	Speed	Giallo	48	Milano
P3	Airtech	Blu	48	Roma
P4	Airtech	Blu	44	Torino
P5	Futura	Blu	-	Milano
P6	Zen	Rosso	42	Torino

CodP	NomeP	Peso
P1	Trion	40
P6	Zen	42

Gestione dei valori NULL

- In presenza di un valore NULL qualsiasi predicato di confronto è falso.
- Si utilizza l'operatore speciale IS [NOT] NULL
- Codice e nome dei prodotti con peso minore di 44 o il cui peso non è indicato:

CodP	NomeP	Colore	Peso	Magazzino
P1	Trion	Rosso	40	Torino
P2	Speed	Giallo	48	Milano
P3	Airtech	Blu	48	Roma
P4	Airtech	Blu	44	Torino
P5	Futura	Blu	-	Milano
P6	Zen	Rosso	42	Torino

```
SELECT CodP, NomeP, Peso
FROM PRODOTTI
WHERE Peso < 44 OR Peso IS NULL;
```

CodP	NomeP	Peso
P1	Trion	40
P5	Futura	-
P6	Zen	42

Struttura dell'istruzione SELECT

SELECT [DISTINCT] *lista_attributi*
FROM *elenco_tabelle*
[WHERE *condizioni_di_tupla*]
[GROUP BY *attributi_di_raggruppamento*]
[HAVING *condizioni_su_aggregati*]
[ORDER BY *attributi_di_ordinamento*];

Ordinamento del risultato

- Clausola ORDER BY

```
SELECT lista_attributi
FROM elenco_tabelle
ORDER BY nome_attributo [ASC | DESC]
{, nome_attributo [ASC | DESC]};
```

- L'ordinamento implicito è crescente (ASC)
- Gli attributi di ordinamento devono comparire nella clausola SELECT, anche implicitamente (come SELECT *)

https://www.postgresql.org/docs/13/queries-order.html

Ordinamento del risultato

- Trovare il codice dei prodotti e il loro peso, ordinando il risultato in ordine decrescente di peso

```
SELECT CodP, Peso
FROM PRODOTTI P
ORDER BY Peso DESC;
```

CodP	NomeP	Colore	Peso	Magazzino
P1	Trion	Rosso	40	Torino
P2	Speed	Giallo	48	Milano
P3	Airtech	Blu	48	Roma
P4	Airtech	Blu	44	Torino
P5	Futura	Blu	40	Milano
P6	Zen	Rosso	42	Torino

CodP	Peso
P2	48
P3	48
P4	44
P6	42
P1	40
P5	40

Ordinamento del risultato

- Trovare tutte le informazioni sui prodotti, ordinando il risultato in ordine crescente di nome e decrescente di peso

```
SELECT CodP, NomeP, Colore, Peso, Magazzino
FROM PRODOTTI P
ORDER BY NomeP, Peso DESC;
```

```
SELECT *
FROM PRODOTTI P
ORDER BY NomeP, Peso DESC;
```

Join

FORNITORI (codF, NomeF, Rating, Sede)

CodF	NomeF	Rating	Sede
F1	Atlante	2	Torino
F2	Oceano	1	Milano
F3	Crono	3	Milano

ORDINI (codF, codP, Qta)

CodP	CodF	Qta
P1	F2	300
P2	F2	400
P2	F3	200

CodF	NomeF	Rating	Sede	CodP	Qta
F2	Oceano	1	Milano	P1	300
F2	Oceano	1	Milano	P2	400
F3	Crono	3	Milano	P2	200

```
SELECT . . .
FROM FORNITORI F, ORDINI O
WHERE F.CodF = O.CodF;
```

Join

- Se l'argomento della clausola FROM è costituito da più di una tabella, le condizioni espresse dalla clausola WHERE vengono applicate sul prodotto cartesiano delle tabelle elencate.

- Un join (interno) può essere specificato indicando in modo esplicito le condizioni che esprimono il legame tra le tabelle.

```
SELECT F.CodF, NomeF, Rating, Sede, CodP, Qta
FROM FORNITORI F, ORDINI O
WHERE F.CodF = O.CodF;
```

Join

- Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF
FROM FORNITORI F, ORDINI O
WHERE F.CodF = O.CodF
AND CodP = 'P2';
```

prodotto cartesiano

Condizione di join

F.CodF	F.NomeF	F.Rating	F.Sede	O.CodF	O.CodP	O.Qta
F1	Atlante	2	Torino	F1	P2	200
F2	Oceano	1	Milano	F2	P2	400
F3	Crono	3	Milano	F3	P2	200

Join

- Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF
FROM FORNITORI F, ORDINI O
WHERE F.CodF = O.CodF
AND CodP = 'P2';
```

SELECT DISTINCT ?

F.CodF	F.NomeF	F.Rating	F.Sede	O.CodF	O.CodP	O.Qta
F1	Atlante	2	Torino	F1	P2	200
F2	Oceano	1	Milano	F2	P2	400
F3	Crono	3	Milano	F3	P2	200

Join

- Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF
FROM FORNITORI F, ORDINI O
WHERE F.CodF = O.CodF
AND CodP = 'P2';
```

SELECT DISTINCT ?

Nella chiave primaria della tabella ORDINI sono presenti sia CodP che CodF …

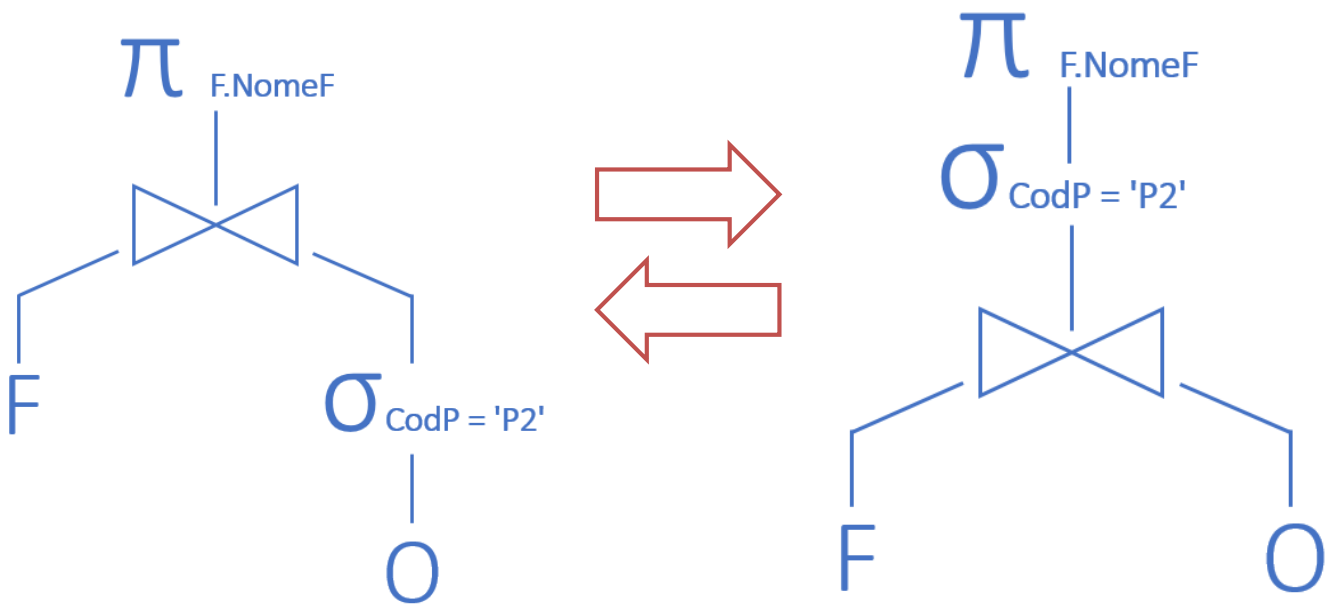
ORDINI (codF, codP, Qta)

F.CodF	F.NomeF	F.Rating	F.Sede	O.CodF	O.CodP	O.Qta
F1	Atlante	2	Torino	F1	P2	200
F2	Oceano	1	Milano	F2	P2	400
F3	Crono	3	Milano	F3	P2	200

CodP	CodF	Qta
P1	F2	300
P2	F2	400
P2	F3	200

SQL: un linguaggio dichiarativo

- In algebra relazionale si definisce l'ordine in cui sono applicati gli operatori, in SQL l'ordine migliore è scelto dall'ottimizzatore
- Il risultato e l'efficienza sono indipendenti dall'ordine delle tabelle nella clausola FROM e dall'ordine dei predicati nella clausola WHERE



Join

- Trovare il nome dei fornitori che forniscono almeno un prodotto rosso

```
SELECT NomeF
FROM FORNITORI F, ORDINI O, PRODOTTI P
WHERE F.CodF = O.CodF
      AND P.CodP = O.CodP
      AND Colore = 'Rosso';
```

SELECT DISTINCT 

ORDINI (codF, codP, Qta)

CodP	CodF	Qta
P1	F2	300
P2	F2	400
P2	F3	200

- Clausola FROM con N tabelle
- Almeno N-1 condizioni di join nella clausola WHERE

Join

- Trovare il nome dei fornitori che forniscono almeno un prodotto rosso

```
SELECT NomeF
FROM FORNITORI F, ORDINI O, PRODOTTI P
WHERE F.CodF = O.CodF
      AND P.CodP = O.CodP
      AND Colore = 'Rosso';
```

SELECT DISTINCT ?

Se lo stesso fornitore fornisce più di un prodotto rosso …

ORDINI (codF, codP, Qta)

CodP	CodF	Qta
P1	F2	300
P2	F2	400
P2	F3	200

- Clausola FROM con N tabelle
- Almeno N-1 condizioni di join nella clausola WHERE

Join: sintassi alternativa

- Permette di specificare ulteriori tipi di join
  - LEFT [OUTER], RIGHT [OUTER], FULL [OUTER]
- Permette di distinguere condizioni di join e condizioni di selezione sulle tuple

```
SELECT [DISTINCT] attributi_da_visualizzare
FROM tabella < INNER | LEFT [OUTER] | RIGHT [OUTER] | FULL [OUTER] >
JOIN tabella ON condizione_di_join
[ WHERE condizioni_di_tupla ];
```

- Una query LEFT [OUTER] JOIN fra le tabelle A e B include tutte le tuple della tabella A (left) e solo le tuple di B che soddisfano la condizione di join.
- Se per una tupla di A non ci sono tuple di B che soddisfano la condizione, nella tupla risultante le colonne di B assumeranno il valore NULL.

INNER Join

- Trovare il nome dei fornitori che forniscono almeno un prodotto rosso

```
SELECT NomeF
FROM PRODOTTI P INNER JOIN ORDINI O ON P.CodP = O.CodP
     INNER JOIN FORNITORI F ON F.CodF = O.CodF
WHERE P.Colore = 'Rosso';

SELECT NomeF
FROM FORNITORI F, ORDINI O, PRODOTTI P
WHERE F.CodF = O.CodF
      AND P.CodP = O.CodP
      AND Colore = 'Rosso';
```

OUTER Join

- Trovare il codice e il nome dei fornitori, insieme al codice dei relativi prodotti forniti, riportando anche i fornitori che non hanno forniture.

```
SELECT F.CodF, NomeF, CodP
FROM FORNITORI F LEFT OUTER JOIN ORDINI O
ON F.CodF = O.CodF;
```

F.CodF	NomeF	CodP
F1	Atlante	P1
…	…	…
F1	Atlante	P6
F2	Oceano	P1
F2	Oceano	P2
…	…	…
F5	Ceo	NULL