Prontuario di base di SQL

LearnSQL.it

SQL, o *Structured Query Language*, è un linguaggio per comunicare con i database. Viene utilizzato per selezionare dati specifici e creare report complessi. Oggi SQL è diventato un linguaggio universale per i dati, usato praticamente in tutte le tecnologie che elaborano dati.

DATI D'ESEMPIO

PAESI

id	nome	popolazione	area
1	Francia	66600000	640680
2	Germania	80700000	357000
...	...	...	...

CITTA

id	nome	id_paese	popolazione	classificazione
1	Parigi	1	2243000	5
2	Berlino	2	3460000	3
...	...	...	...	...

QUERY SU UNA SINGOLA TABELLA

Recuperare tutte le colonne dalla tabella `paesi`:

```
SELECT *
FROM paesi;
```

Visualizzare le colonne `id` e `nome` dalla tabella `citta`:

```
SELECT id, nome
FROM citta;
```

Ottenere i nomi delle città ordinati in base alla colonna `classificazione` in ordine ascendente:

```
SELECT nome
FROM citta
ORDER BY classificazione [ASC];
```

Selezionare i nomi delle città ordinati in base alla colonna `classificazione` in ordine discendente:

```
SELECT nome
FROM citta
ORDER BY classificazione DESC;
```

ALIAS

COLONNE

```
SELECT nome AS nome_citta
FROM citta;
```

TABELLE

```
SELECT pa.nome, ci.nome
FROM citta AS ci
JOIN paesi AS pa
  ON ci.id_paese = pa.id;
```

RISULTATI DI FILTRAGGIO

OPERATORI DI CONFRONTO

Selezionare i nomi delle città con classificazione maggiore di 3:

```
SELECT nome
FROM citta
WHERE classificazione > 3;
```

Ottenere i nomi delle città che non sono né Berlino né Madrid:

```
SELECT nome
FROM citta
WHERE nome != 'Berlino'
  AND nome != 'Madrid';
```

OPERATORI DI TESTO

Ricerca di nomi di città che iniziano con una "P" o finiscono con una "s":

```
SELECT nome
FROM citta
WHERE nome LIKE 'P%'
  OR nome LIKE '%s';
```

Ricerca di nomi di città che iniziano con qualsiasi lettera seguita da "ublino" (come Dublino in Irlanda o Lublino in Polonia):

```
SELECT nome
FROM citta
WHERE nome LIKE '_ublino';
```

ALTRI OPERATORI

Selezionare i nomi delle città con una popolazione compresa tra 500.000 e 5 milioni di abitanti:

```
SELECT nome
FROM citta
WHERE popolazione BETWEEN 500000 AND 5000000;
```

Visualizzare i nomi delle città che non hanno un valore di classificazione:

```
SELECT nome
FROM citta
WHERE classificazione IS NOT NULL;
```

Ottenere i nomi delle città situate in Paesi il cui identificatore è 1, 4, 7 o 8:

```
SELECT nome
FROM citta
WHERE id_paese IN (1, 4, 7, 8);
```

QUERY SU PIÙ TABELLE

INNER JOIN

`JOIN` (o con il nome completo **`INNER JOIN`**) restituisce le righe con valori che corrispondono in entrambe le tabelle.

```
SELECT citta.nome, paesi.nome
FROM citta
[INNER] JOIN paesi
  ON citta.id_paese = paesi.id;
```

CITTA			PAESI	
id	nome	id_paese	id	nome
1	Parigi	1	1	Francia
2	Berlino	2	2	Germania
3	Varsavia	4	3	Islanda

LEFT JOIN

`LEFT JOIN` restituisce tutte le righe della tabella di sinistra con le righe corrispondenti della tabella di destra. Se non ci sono righe corrispondenti, i valori restituiti dalla seconda tabella saranno **`NULL`**.

```
SELECT citta.nome, paesi.nome
FROM citta
LEFT JOIN paesi
  ON citta.id_paese = paesi.id;
```

CITTA			PAESI	
id	nome	id_paese	id	nome
1	Parigi	1	1	Francia
2	Berlino	2	2	Germania
3	Varsavia	4	NULL	NULL

RIGHT JOIN

`RIGHT JOIN` restituisce tutte le righe della tabella di destra con le righe corrispondenti della tabella di sinistra. Se non ci sono righe corrispondenti, i valori restituiti dalla seconda tabella saranno **`NULL`**.

```
SELECT citta.nome, paesi.nome
FROM citta
RIGHT JOIN paesi
  ON citta.id_paese = paesi.id;
```

CITTA			PAESI	
id	nome	id_paese	id	nome
1	Parigi	1	1	Francia
2	Berlino	2	2	Germania
NULL	NULL	NULL	3	Islanda

FULL JOIN

`FULL JOIN` (o con il nome completo **`FULL OUTER JOIN`**) restituisce tutte le righe di entrambe le tabelle - se non c'è una riga corrispondente nella seconda tabella, vengono restituiti i valori **`NULL`**.

```
SELECT citta.nome, paesi.nome
FROM citta
FULL [OUTER] JOIN paesi
  ON citta.id_paese = paesi.id;
```

CITTA			PAESI	
id	nome	id_paese	id	nome
1	Parigi	1	1	Francia
2	Berlino	2	2	Germania
3	Varsavia	4	NULL	NULL
NULL	NULL	NULL	3	Islanda

CROSS JOIN

`CROSS JOIN` restituisce tutte le possibili combinazioni di righe delle due tabelle. Sono disponibili due sintassi.

```
SELECT citta.nome, paesi.nome
FROM citta
CROSS JOIN paesi;
```

```
SELECT citta.nome, paesi.nome
FROM citta, paesi;
```

CITTA			PAESI	
id	nome	id_paese	id	nome
1	Parigi	1	1	Francia
1	Parigi	1	2	Germania
2	Berlino	2	1	Francia
2	Berlino	2	2	Germania

NATURAL JOIN

`NATURAL JOIN` collega le tabelle in base a tutte le colonne con lo stesso nome.

```
SELECT citta.nome, paesi.nome
FROM citta
NATURAL JOIN paesi;
```

CITTA			PAESI	
id_paese	id	nome	nome	id
6	6	San Marino	San Marino	6
7	7	Città del Vaticano	Città del Vaticano	7
5	9	Grecia	Grecia	9
10	11	Monaco	Monaco	10

`NATURAL JOIN` utilizza queste colonne per abbinare le righe: **`citta.id`**, **`citta.nome`**, **`paesi.id`**, **`paesi.nome`**.
`NATURAL JOIN` è usato molto raramente nella pratica.

Prontuario di base di SQL

LearnSQL.it

AGGREGAZIONE E RAGGRUPPAMENTO

GROUP BY **raggruppa** le righe con gli stessi valori nelle colonne specificate. Genera sommatorie (aggregati) per ogni combinazione unica di valori.

CITTA		
id	**nome**	**id_paese**
1	Parigi	1
101	Marsiglia	1
102	Lione	1
2	Berlino	2
103	Amburgo	2
104	Monaco	2
3	Varsavia	4
105	Cracovia	4

→

CITTA	
id_paese	**conteggio**
1	3
2	3
4	2

FUNZIONI DI AGGREGAZIONE

- `avg(expr)` – valore medio delle righe del gruppo
- `count(expr)` – numero di valori per le righe del gruppo
- `max(expr)` – valore massimo del gruppo
- `min(expr)` – valore minimo del gruppo
- `sum(expr)` – somma dei valori del gruppo

ESEMPIO DI QUERY

Trovare il numero di città:

```
SELECT COUNT(*)
FROM citta;
```

Trovare il numero di città con un punteggio non nullo:

```
SELECT COUNT(classificazione)
FROM citta;
```

Trovare il numero di valori distinti per i Paesi:

```
SELECT COUNT(DISTINCT id_paese)
FROM citta;
```

Trovare i Paesi con la popolazione più piccola e più grande:

```
SELECT MIN(popolazione), MAX(popolazione)
FROM paesi;
```

Determinare la popolazione totale delle città nei rispettivi Paesi:

```
SELECT id_paese, SUM(popolazione)
FROM citta
GROUP BY id_paese;
```

Determinare la classificazione media delle città nei rispettivi Paesi, se la media è superiore a 3,0:

```
SELECT id_paese, AVG(classificazione)
FROM citta
GROUP BY id_paese
HAVING AVG(classificazione) > 3.0;
```

SUBQUERY

Una subquery è una query annidata in un'altra query o in un'altra subquery. Esistono diversi tipi di sottoquery.

VALORE SINGOLO

La subquery più semplice restituisce esattamente una colonna e una riga. Può essere utilizzata con gli operatori di confronto =, <, <=, > o >=.
Questa query viene utilizzata per trovare le città che hanno la stessa classificazione di Parigi:

```
SELECT nome
FROM citta
WHERE classificazione = (
  SELECT classificazione
  FROM citta
  WHERE nome = 'Parigi'
);
```

VALORI MULTIPLI

Una sottoquery può anche restituire più colonne o più righe. Queste sottoquery possono essere utilizzate con gli operatori `IN`, `EXISTS`, `ALL` o `ANY`.
Questa query trova le città dei Paesi con più di 20 milioni di abitanti:

```
SELECT nome
FROM citta
WHERE id_paese IN (
  SELECT id_paese
  FROM paesi
  WHERE popolazione > 20000000
);
```

SUBQUERY CORRELATA

Una sottoquery correlata si riferisce alle tabelle inserite nella query esterna. Una sottoquery correlata dipende dalla query esterna. Non può essere eseguita indipendentemente dalla query esterna.
Questa query cerca le città la cui popolazione è superiore alla popolazione media del Paese:

```
SELECT *
FROM citta citta_principale
WHERE popolazione > (
  SELECT AVG(popolazione)
  FROM citta citta_media
  WHERE citta_media.id_paese = citta_principale.id_paese
);
```

Questa query cerca i Paesi con almeno una città:

```
SELECT nome
FROM paesi
WHERE EXISTS (
  SELECT *
  FROM citta
  WHERE id_paese = paesi.id
);
```

OPERAZIONI DI SET

Le operazioni di set vengono utilizzate per combinare i risultati di due o più query in un unico risultato. Le query combinate devono restituire lo stesso numero di colonne e tipi di dati compatibili. I nomi delle colonne corrispondenti possono essere diversi.

CICLISMO		
id	**nome**	**paese**
1	YK	DE
2	ZG	DE
3	WT	PL
...	...	...

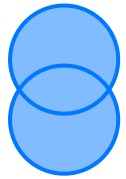
PATTINAGGIO		
id	**nome**	**paese**
1	YK	DE
2	DF	DE
3	AK	PL
...	...	...

UNION

`UNION` combina i risultati di due set di dati ed esclude i duplicati. **`UNION ALL`** non esclude le righe duplicate.

Questa query visualizza i ciclisti e i pattinatori tedeschi:

```
SELECT nome
FROM ciclismo
WHERE paese = 'DE'
UNION / UNION ALL
SELECT nome
FROM pattinaggio
WHERE paese = 'DE';
```

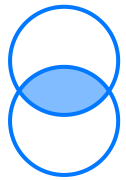


INTERSECT

`INTERSECT` restituisce solo le righe che compaiono in entrambi gli insiemi di risultati.

Questa ricerca mostra i ciclisti tedeschi che sono anche pattinatori tedeschi allo stesso tempo:

```
SELECT nome
FROM ciclismo
WHERE paese = 'DE'
INTERSECT
SELECT nome
FROM pattinaggio
WHERE paese = 'DE';
```



EXCEPT

`EXCEPT` restituisce solo le righe che compaiono nel primo insieme di risultati, ma non nel secondo.

Questa query visualizza i ciclisti tedeschi, a meno che non siano anche pattinatori tedeschi allo stesso tempo:

```
SELECT nome
FROM ciclismo
WHERE paese = 'DE'
EXCEPT / MINUS
SELECT nome
FROM pattinaggio
WHERE paese = 'DE';
```