Prontuario di Funzioni Finestra

LearnSQL.it

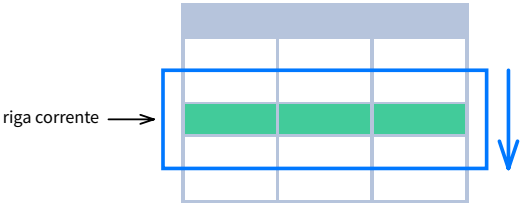
INDICE

Prontuario di Funzioni Finestra

LearnSQL.it

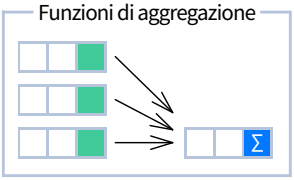
FUNZIONI FINESTRA

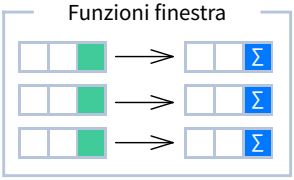
Le **funzioni finestra** calcolano il risultato in base a una finestra scorrevole (*frame*), un insieme di righe in qualche modo correlate alla riga corrente.



FUNZIONI DI AGGREGAZIONE VS. FUNZIONI FINESTRA

A differenza delle funzioni di aggregazione, le funzioni finestra non comprimono le righe.





Questo prontuario è stato preparato da **LearnSQL.it** nell'ambito del suo programma di formazione su SQL. Vedi gli altri cheat sheet di **SQL**.

Prontuario di Funzioni Finestra

LearnSQL.it

SINTASSI

```
SELECT citta, mese,
  SUM(venduto) OVER (
    PARTITION BY citta
    ORDER BY mese
    RANGE UNBOUNDED PRECEDING) totale
FROM vendite;
```

```
SELECT <colonna_1>, <colonna_2>,
  <funzione_finestra> OVER (
    PARTITION BY <...>
    ORDER BY <...>
    <window_frame>) <alias_colonna_finestra>
FROM <nome_tabella>;
```

Prontuario di Funzioni Finestra

LearnSQL.it

DEFINIZIONE DI FINESTRA CON NOME

```
SELECT paese, citta,
  RANK() OVER media_vendite_paese
FROM vendite
WHERE mese BETWEEN 1 AND 6
GROUP BY paese, citta
HAVING sum(venduto) > 10000
WINDOW media_vendite_paese AS (
  PARTITION BY paese
  ORDER BY avg(venduto) DESC)
ORDER BY paese, citta;
```

```
SELECT <colonna_1>, <colonna_2>,
  <funzione_finestra>() OVER <nome_finestra>
FROM <nome_tabella>
WHERE <...>
GROUP BY <...>
HAVING <...>
WINDOW <nome_finestra> AS (
  PARTITION BY <...>
  ORDER BY <...>
  <window_frame>)
ORDER BY <...>;
```

`PARTITION BY`, `ORDER BY` e la definizione della cornice della finestra (*window frame*) sono tutti opzionali.

Prontuario di Funzioni Finestra

LearnSQL.it

ORDINE LOGICO DELLE OPERAZIONI IN SQL

1. `FROM`, `JOIN`
2. `WHERE`
3. `GROUP BY`
4. Funzioni di aggregazione
5. `HAVING`
6. **Funzioni finestra**
7. `SELECT`
8. `DISTINCT`
9. `UNION`/`INTERSECT`/`EXCEPT`
10. `ORDER BY`
11. `OFFSET`
12. `LIMIT`/`FETCH`/`TOP`

È possibile utilizzare le funzioni finestra in `SELECT` e `ORDER BY`. Tuttavia, non è possibile inserire le funzioni finestra in nessun punto delle clausole `FROM`, `WHERE`, `GROUP BY` o `HAVING`.

Prontuario di Funzioni Finestra

LearnSQL.it

PARTITION BY

divide le righe in gruppi multipli, chiamati **partizioni**, a cui viene applicata la funzione finestra.

mese	citta	venduto
1	Roma	200
2	Parigi	500
1	Londra	100
1	Parigi	300
2	Roma	300
2	Londra	400
3	Roma	400

```
PARTITION BY citta
```

mese	citta	venduto	sum
1	Parigi	300	800
2	Parigi	500	800
1	Roma	200	900
2	Roma	300	900
3	Roma	400	900
1	Londra	100	500
2	Londra	400	500

Partizione predefinita: senza la clausola `PARTITION BY`, l'intero insieme di risultati è la partizione.

Prontuario di Funzioni Finestra

LearnSQL.it

ORDER BY

specifica l'ordine delle righe in ogni partizione a cui viene applicata la funzione finestra.

venduto	citta	mese
200	Roma	1
500	Parigi	2
100	Londra	1
300	Parigi	1
300	Roma	2
400	Londra	2
400	Roma	3

```
PARTITION BY citta
ORDER BY mese
```

venduto	citta	mese
300	Parigi	1
500	Parigi	2
200	Roma	1
300	Roma	2
400	Roma	3
100	Londra	1
400	Londra	2

ORDER BY predefinito: senza la clausola ORDER BY, l'ordine delle righe all'interno di ogni partizione è arbitrario.

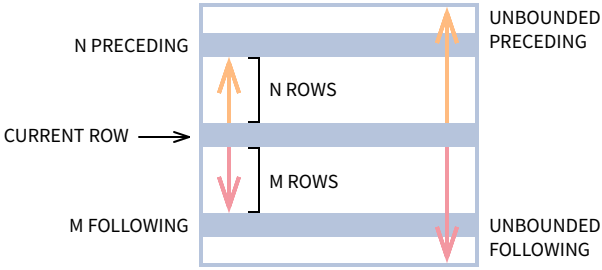
Prontuario di Funzioni Finestra

LearnSQL.it

CORNICE DELLA FINESTRA

Una **cornice** (*frame*) è un insieme di righe che sono in qualche modo collegate alla riga corrente. La cornice della finestra viene valutata separatamente all'interno di ogni partizione.

```
<ROWS | RANGE | GROUPS> BETWEEN limite_inferiore
AND limite_superiore
```



I limiti possono essere una qualsiasi delle cinque opzioni:

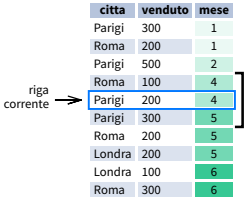
- `UNBOUNDED PRECEDING`
- `n PRECEDING`
- `CURRENT ROW`
- `n FOLLOWING`
- `UNBOUNDED FOLLOWING`

Il `limite_inferiore` deve essere PRECEDENTE al `limite_superiore`.

Prontuario di Funzioni Finestra

LearnSQL.it

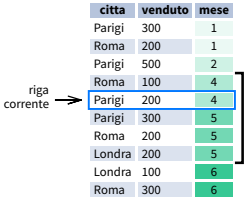
`ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING`



citta	venduto	mese
Parigi	300	1
Roma	200	1
Parigi	500	2
Roma	100	4
Parigi	200	4
Parigi	300	5
Roma	200	5
Londra	200	5
Londra	100	6
Roma	300	6

1 riga prima della riga corrente e 1 riga dopo la riga corrente

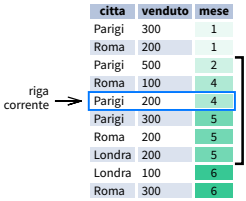
`RANGE BETWEEN 1 PRECEDING AND 1 FOLLOWING`



citta	venduto	mese
Parigi	300	1
Roma	200	1
Parigi	500	2
Roma	100	4
Parigi	200	4
Parigi	300	5
Roma	200	5
Londra	200	5
Londra	100	6
Roma	300	6

Valori nell'intervallo tra 3 e 5 ORDER BY deve contenere un'unica espressione

`GROUPS BETWEEN 1 PRECEDING AND 1 FOLLOWING`



citta	venduto	mese
Parigi	300	1
Roma	200	1
Parigi	500	2
Roma	100	4
Parigi	200	4
Parigi	300	5
Roma	200	5
Londra	200	5
Londra	100	6
Roma	300	6

1 gruppo prima della riga corrente e 1 gruppo dopo la riga corrente, indipendentemente dal valore

A partire dal 2024, GROUPS è supportato solo in PostgreSQL 11 e successivi.

Prontuario di Funzioni Finestra

LearnSQL.it

ABBREVIAZIONI

ABBREVIAZIONE	SIGNIFICATO
`UNBOUNDED PRECEDING`	`BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW`
`n PRECEDING`	`BETWEEN n PRECEDING AND CURRENT ROW`
`CURRENT ROW`	`BETWEEN CURRENT ROW AND CURRENT ROW`
`n FOLLOWING`	`BETWEEN CURRENT ROW AND n FOLLOWING`
`UNBOUNDED FOLLOWING`	`BETWEEN CURRENT ROW AND UNBOUNDED FOLLOWING`

CORNICE PREDEFINITA DELLA FINESTRA

Se viene specificato `ORDER BY`, la cornice è `RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW`.

Senza `ORDER BY`, la cornice è `ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING`.

Prontuario di Funzioni Finestra

LearnSQL.it

ELENCO DELLE FUNZIONI FINESTRA

Funzioni di aggregazione

- `avg()`
- `count()`
- `max()`
- `min()`
- `sum()`

Funzioni di ranking

- `row_number()`
- `rank()`
- `dense_rank()`

Funzioni di distribuzione

- `percent_rank()`
- `cume_dist()`

Funzioni analitiche

- `lead()`
- `lag()`
- `ntile()`
- `first_value()`
- `last_value()`
- `nth_value()`

Questo prontuario è stato preparato da **LearnSQL.it** nell'ambito del suo programma di formazione su SQL. Vedi gli altri cheat sheet di **SQL**.

Prontuario di Funzioni Finestra

LearnSQL.it

FUNZIONI DI AGGREGAZIONE

- **`avg(`**`expr`**`)`** – valore medio per le righe all'interno della finestra
- **`count(`**`expr`**`)`** – conteggio dei valori per le righe all'interno del riquadro della finestra
- **`max(`**`expr`**`)`** – valore massimo all'interno della finestra
- **`min(`**`expr`**`)`** – valore minimo all'interno della finestra
- **`sum(`**`expr`**`)`** – somma dei valori all'interno della finestra

ORDER BY e finestre: le funzioni di aggregazione non richiedono un `ORDER BY`. Accettano la definizione di finestra (`ROWS`, `RANGE`, `GROUPS`).

Prontuario di Funzioni Finestra

LearnSQL.it

FUNZIONI DI RANKING

- **`row_number()`** – numero unico per ogni riga all'interno della partizione, con numeri diversi per i valori uguali
- **`rank()`** – rango all'interno della partizione, con spazi vuoti e stesso rango per i valori uguali
- **`dense_rank()`** – rango all'interno della partizione, senza spazi vuoti e con lo stesso rango per i valori uguali

citta	prezzo	row_number	rank	dense_rank
		over(order by prezzo)		
Parigi	7	1	1	1
Roma	7	2	1	1
Londra	8.5	3	3	2
Berlino	8.5	4	3	2
Mosca	9	5	5	3
Madrid	10	6	6	4
Oslo	10	7	6	4

ORDER BY e Window Frame: `rank()` e `dense_rank()` richiedono `ORDER BY`, ma `row_number()` non richiede `ORDER BY`. Le funzioni di classificazione non accettano la definizione di cornice (`ROWS`, `RANGE`, `GROUPS`).

Prontuario di Funzioni Finestra

LearnSQL.it

FUNZIONI DI DISTRIBUZIONE

- **`percent_rank()`** – il percentile di una riga, un valore nell'intervallo `[0, 1]`: (rank-1) / (numero totale di righe - 1)
- **`cume_dist()`** – la distribuzione cumulativa di un valore all'interno di un gruppo di valori, cioè il numero di righe con valori inferiori o uguali al valore della riga corrente diviso per il numero totale di righe; un valore nell'intervallo `(0, 1]`

`percent_rank() OVER(ORDER BY venduto)`

citta	venduto	percent_rank	
Parigi	100	0	
Berlino	150	0.25	
Roma	200	0.5	*
Mosca	200	0.5	
Londra	300	1	

* senza questa riga il 50% dei valori è inferiore al valore di questa riga

`cume_dist() OVER(ORDER BY venduto)`

citta	venduto	cume_dist	
Parigi	100	0.2	
Berlino	150	0.4	
Roma	200	0.8	*
Mosca	200	0.8	
Londra	300	1	

* L'80% dei valori è minore o uguale a questo

ORDER BY e Window Frame: le funzioni di distribuzione richiedono `ORDER BY`. Non accettano la definizione di cornice (`ROWS`, `RANGE`, `GROUPS`).

Prontuario di Funzioni Finestra

LearnSQL.it

FUNZIONI ANALITICHE

- **`lead(`**`expr, offset, predefinito`**`)`** - il valore per l'*offset* delle righe successive alla corrente; *offset* e *predefinito* sono opzionali; valori predefiniti: *offset* = 1, *predefinito* = `NULL`

`lead(venduto) OVER(ORDER BY mese)`

order by mese ↓

mese	venduto	lead
1	500	300
2	300	400
3	400	100
4	100	500
5	500	NULL

`lead(venduto, 2, 0) OVER(ORDER BY mese)`

order by mese ↓

mese	venduto	lead
1	500	400
2	300	100
3	400	500
4	100	0
5	500	0

offset = 2 ↑

Prontuario di Funzioni Finestra

LearnSQL.it

- **`lag(`**`expr, offset, predefinito`**`)`** - il valore per l'*offset* delle righe prima di quella corrente; *offset* e *predefinito* sono opzionali; valori predefiniti: *offset* = 1, *predefinito* = `NULL`

`lag(venduto) OVER(ORDER BY mese)`

order by mese

mese	venduto	lag
1	500	NULL
2	300	500
3	400	300
4	100	400
5	500	100

`lag(venduto, 2, 0) OVER(ORDER BY mese)`

order by mese

mese	venduto	lag
1	500	0
2	300	0
3	400	500
4	100	300
5	500	400

offset = 2

Prontuario di Funzioni Finestra

LearnSQL.it

- **`ntile(n)`** – divide le righe all'interno di una partizione il più equamente possibile in *n* gruppi e assegna a ogni riga il numero del gruppo.

`ntile(3)`

citta	venduto		ntile
Roma	100	1	1
Parigi	100		1
Londra	200		1
Mosca	200	2	2
Berlino	200		2
Madrid	300		2
Oslo	300	3	3
Dublin	300		3

ORDER BY e Window Frame: `ntile()`, `lead()` e `lag()` richiedono un `ORDER BY`. Non accettano la definizione di cornice (`ROWS`, `RANGE`, `GROUPS`).

Prontuario di Funzioni Finestra

LearnSQL.it

- **`first_value(expr)`** – il valore della prima riga all'interno del riquadro della finestra
- **`last_value(expr)`** – il valore dell'ultima riga all'interno della cornice della finestra

```
first_value(venduto) OVER
(PARTITION BY citta ORDER BY mese)
```

citta	mese	venduto	first_value
Parigi	1	500	500
Parigi	2	300	500
Parigi	3	400	500
Roma	2	200	200
Roma	3	300	200
Roma	4	500	200

```
last_value(venduto) OVER
(PARTITION BY citta ORDER BY mese
RANGE BETWEEN UNBOUNDED PRECEDING
AND UNBOUNDED FOLLOWING)
```

citta	mese	venduto	last_value
Parigi	1	500	400
Parigi	2	300	400
Parigi	3	400	400
Roma	2	200	500
Roma	3	300	500
Roma	4	500	500

Nota: di solito si consiglia di utilizzare `RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING` con `last_value()`. Con la cornice predefinita per `ORDER BY`, `RANGE UNBOUNDED PRECEDING`, `last_value()` restituisce il valore della riga corrente.

Prontuario di Funzioni Finestra

LearnSQL.it

- **`nth_value(expr, n)`** – il valore per la riga *n-esima* all'interno del riquadro della finestra; *n* deve essere un numero intero

citta	mese	venduto	nth_value
Parigi	1	500	300
Parigi	2	300	300
Parigi	3	400	300
Roma	2	200	300
Roma	3	300	300
Roma	4	500	300
Roma	5	300	300
Londra	1	100	NULL

ORDER BY e Window Frame: `first_value()`, `last_value()` e `nth_value()` non richiedono un `ORDER BY`. Accettano la definizione di cornice (`ROWS`, `RANGE`, `GROUPS`).

Scopri tutto su LearnSQL.it