













A. Use EXTRACT(MONTH FROM order_date)




Limit to 1000 rows

```
1  -- A: Use EXTRACT(MONTH FROM order_date)
2  • SELECT
3      order_id,
4      order_date,
5      EXTRACT(MONTH FROM order_date) AS month
6  FROM
7      online_sales
8  LIMIT 10;
9
10 -- B: GROUP BY year and month
11 • SELECT
12     EXTRACT(YEAR FROM order_date) AS year,
13     EXTRACT(MONTH FROM order_date) AS month,
14     COUNT(*) AS order_count
15 FROM
16     online sales
```











Result Grid

 Filter Rows:

Export:  Wrap

	order_id	order_date	month
▶	1	2024-05-10	5
	2	2024-12-31	12
	3	2024-11-10	11
	4	2022-05-02	5
	5	2023-04-12	4
	6	2022-11-27	11
	7	2022-03-29	3
	8	2024-05-21	5
	9	2022-05-11	5
	10	2024-02-09	2



B. GROUP BY year and month





Limit to 1000 rows

```
7      online_sales
8  LIMIT 10;
9
10 -- B: GROUP BY year and month
11 • SELECT
12     EXTRACT(YEAR FROM order_date) AS year,
13     EXTRACT(MONTH FROM order_date) AS month,
14     COUNT(*) AS order_count
15 FROM
16     online_sales
17 GROUP BY
18     EXTRACT(YEAR FROM order_date),
19     EXTRACT(MONTH FROM order_date);
20
21 -- C: SUM(amount) for revenue
22 • SELECT
```

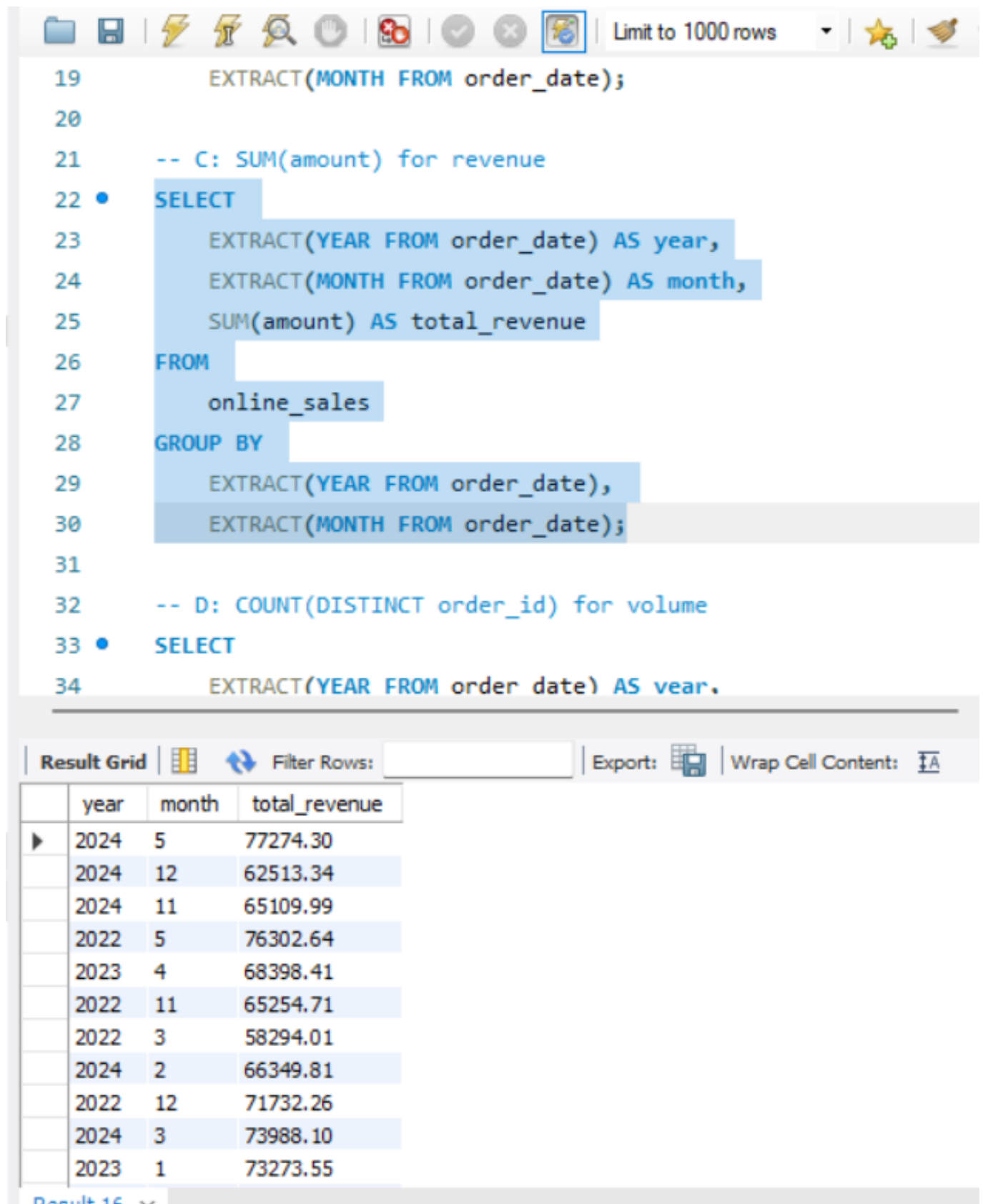
Result Grid

 Filter Rows:

Export:  

	year	month	order_count
▶	2024	5	143
	2024	12	136
	2024	11	131
	2022	5	153
	2023	4	135
	2022	11	130
	2022	3	120
	2024	2	126

C. SUM(amount) for revenue



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The SQL editor contains the following code:

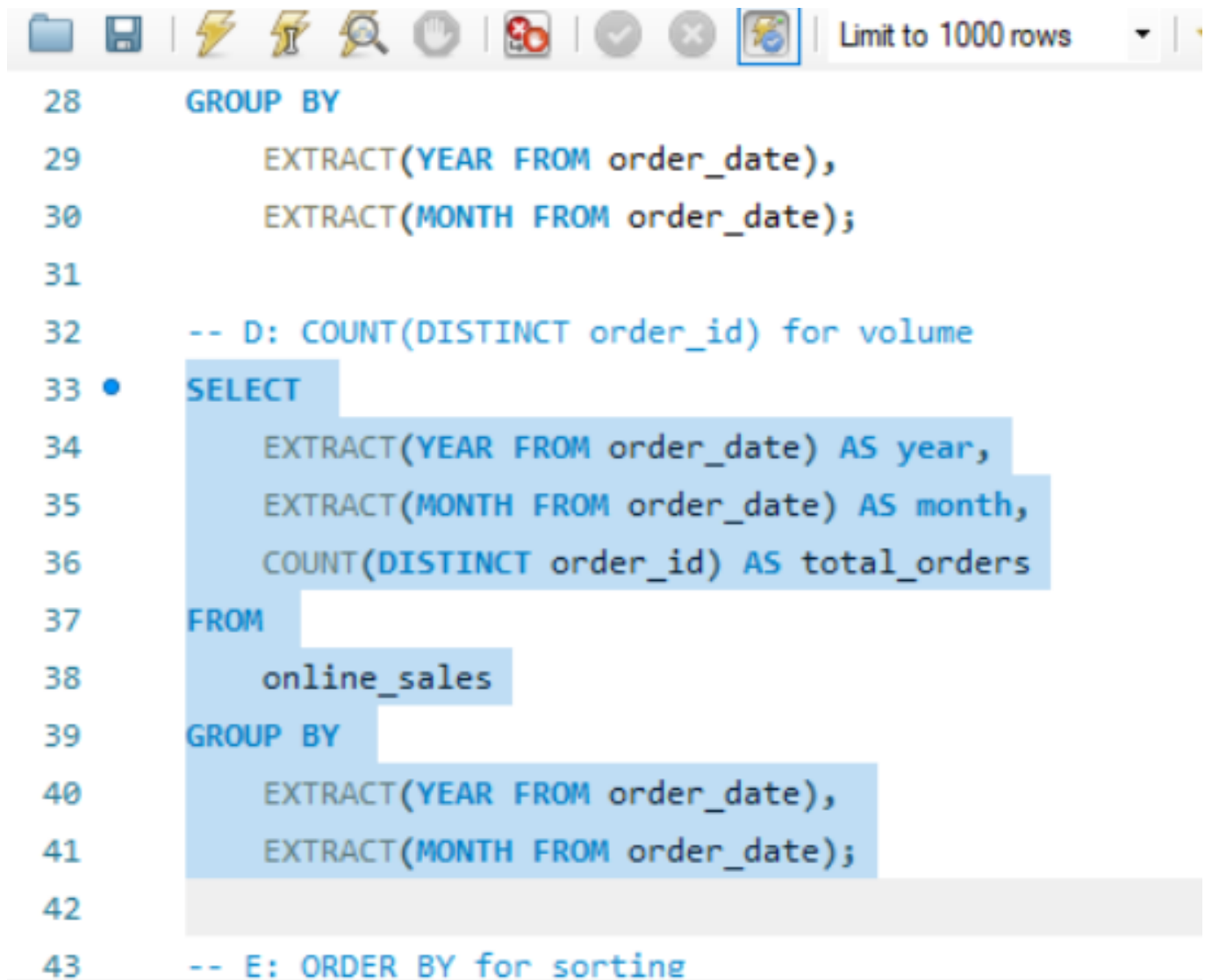
```
19      EXTRACT(MONTH FROM order_date);
20
21      -- C: SUM(amount) for revenue
22      • SELECT
23          EXTRACT(YEAR FROM order_date) AS year,
24          EXTRACT(MONTH FROM order_date) AS month,
25          SUM(amount) AS total_revenue
26      FROM
27          online_sales
28      GROUP BY
29          EXTRACT(YEAR FROM order_date),
30          EXTRACT(MONTH FROM order_date);
31
32      -- D: COUNT(DISTINCT order_id) for volume
33      • SELECT
34          EXTRACT(YEAR FROM order_date) AS year,
```

Below the editor is the 'Result Grid' section, which includes a 'Filter Rows' input field and buttons for 'Export' and 'Wrap Cell Content'. The results are displayed in a table with 4 columns: year, month, and total_revenue. The table shows 12 rows of data, with the first row highlighted.

	year	month	total_revenue
▶	2024	5	77274.30
	2024	12	62513.34
	2024	11	65109.99
	2022	5	76302.64
	2023	4	68398.41
	2022	11	65254.71
	2022	3	58294.01
	2024	2	66349.81
	2022	12	71732.26
	2024	3	73988.10
	2023	1	73273.55

Result 16

D. COUNT(DISTINCT order_id) for volume



The screenshot shows a SQL editor with a toolbar at the top containing icons for file operations, execution, and search. The query text is as follows:

```
28 GROUP BY
29     EXTRACT(YEAR FROM order_date),
30     EXTRACT(MONTH FROM order_date);
31
32 -- D: COUNT(DISTINCT order_id) for volume
33 • SELECT
34     EXTRACT(YEAR FROM order_date) AS year,
35     EXTRACT(MONTH FROM order_date) AS month,
36     COUNT(DISTINCT order_id) AS total_orders
37 FROM
38     online_sales
39 GROUP BY
40     EXTRACT(YEAR FROM order_date),
41     EXTRACT(MONTH FROM order_date);
42
43 -- E: ORDER BY for sorting
```

Below the query editor is the 'Result Grid' section, which includes a 'Filter Rows' input field and an 'Export' button. The grid displays the results of the query, showing columns for year, month, and total_orders.

	year	month	total_orders
▶	2022	1	153
	2022	2	118
	2022	3	120
	2022	4	131
	2022	5	153
	2022	6	131
	2022	7	155
	2022	8	130
	2022	9	131
	2022	10	129
	2022	11	130

At the bottom left, there is a tab labeled 'Result 17' with a close button (X).

E. ORDER BY for sorting

```
43  -- E: ORDER BY for sorting
44  •  SELECT
45      EXTRACT(YEAR FROM order_date) AS year,
46      EXTRACT(MONTH FROM order_date) AS month,
47      SUM(amount) AS total_revenue,
48      COUNT(DISTINCT order_id) AS total_orders
49  FROM
50      online_sales
51  GROUP BY
52      EXTRACT(YEAR FROM order_date),
53      EXTRACT(MONTH FROM order_date)
54  ORDER BY
55      year ASC,
56      month ASC;
57
58  -- F: Limit results to a specific time period (e.g.. 2023)
```

Result Grid		Filter Rows:		Export:	Wrap Cell Content:
	year	month	total_revenue	total_orders	
▶	2022	1	77426.00	153	
	2022	2	56152.82	118	
	2022	3	58294.01	120	
	2022	4	70322.26	131	
	2022	5	76302.64	153	
	2022	6	66360.61	131	
	2022	7	77127.49	155	
	2022	8	70485.89	130	
	2022	9	70379.54	131	
	2022	10	65375.31	129	
	2022	11	65254.71	130	

Result 18 ×

F. Limit results to a specific time period (e.g., 2023)

```
56      MONTH FROM order_date) AS month,
57
58      -- F: Limit results to a specific time period (e.g., 2023)
59  •  SELECT
60      EXTRACT(YEAR FROM order_date) AS year,
61      EXTRACT(MONTH FROM order_date) AS month,
62      SUM(amount) AS total_revenue,
63      COUNT(DISTINCT order_id) AS total_orders
64  FROM
65      online_sales
66  WHERE
67      order_date BETWEEN '2023-01-01' AND '2023-12-31'
68  GROUP BY
69      EXTRACT(YEAR FROM order_date),
70      EXTRACT(MONTH FROM order_date)
71  ORDER BY
```

result Grid | Filter Rows: | Export: | Wrap Cell Content:

year	month	total_revenue	total_orders
2023	1	73273.55	150
2023	2	63086.55	130
2023	3	74145.41	137
2023	4	68398.41	135
2023	5	75518.06	149
2023	6	63188.02	127
2023	7	67939.82	140
2023	8	70257.12	135
2023	9	70744.95	143
2023	10	72394.92	147
2023	11	71619.32	134

result 19