SQL Learnings

Topics

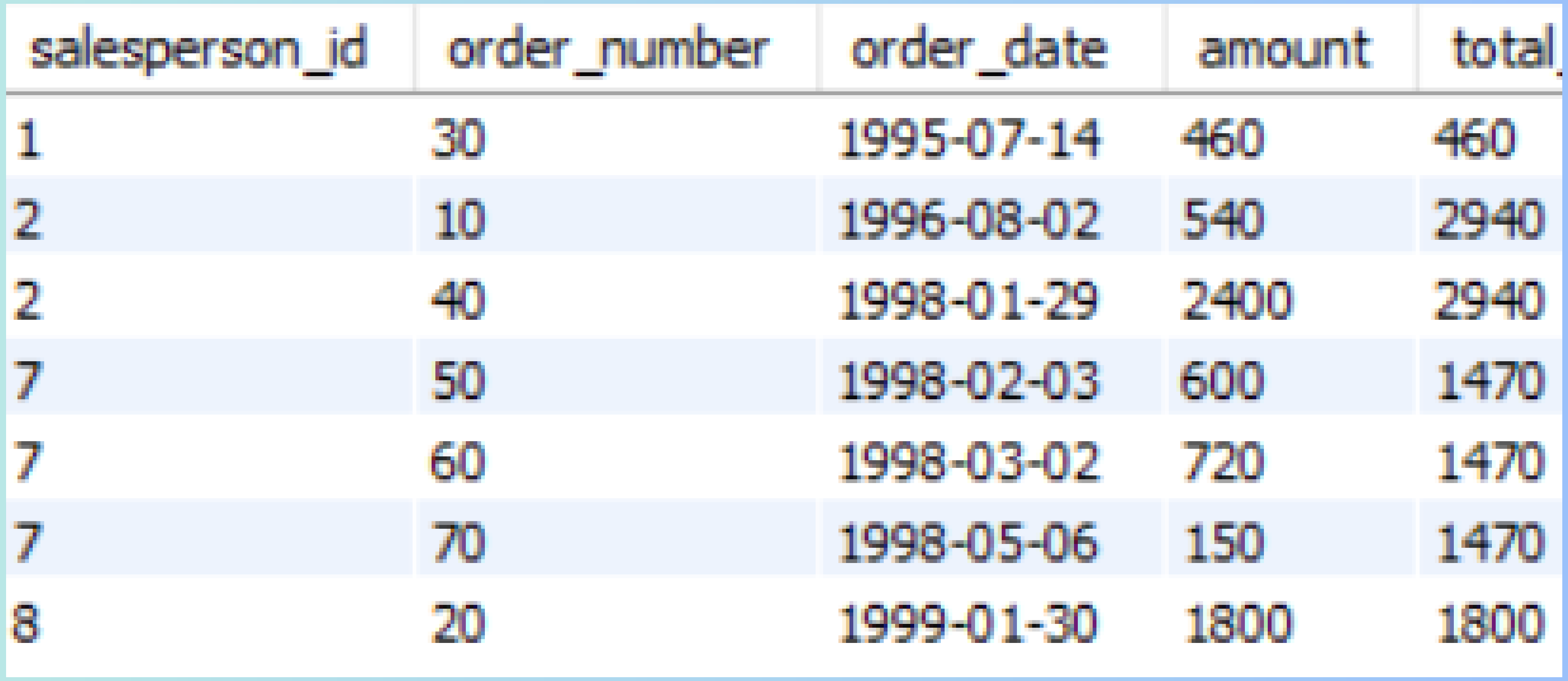
- Partition By
- Group By
- Partition By with Group By
- Examples

Partition By

The **PARTITION BY** clause is used within window functions to divide the result set into partitions and perform calculations on each partition separately.

For example, all rows with salesperson_id 1 form one partition, all rows with salesperson_id 2 form another, and so on.

```
SELECT salesperson_id, order_number, order_date, amount,
SUM(amount) OVER (PARTITION BY salesperson_id) AS total_sale
FROM int_orders;
```



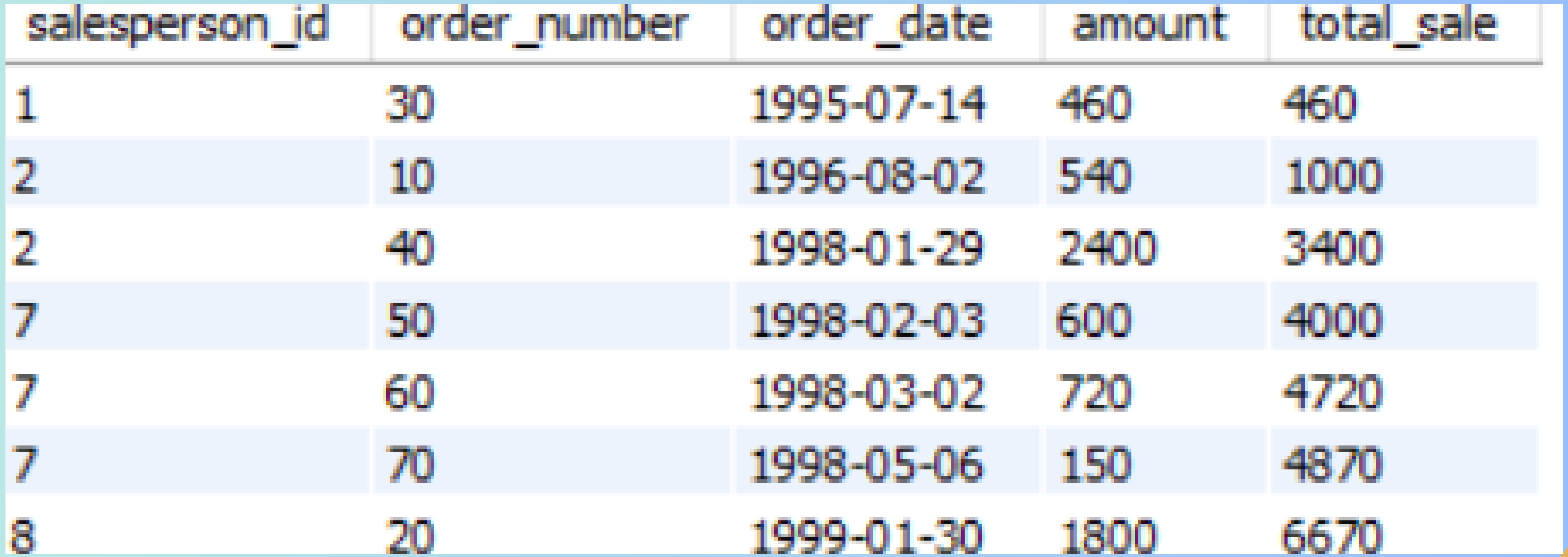
salesperson_id	order_number	order_date	amount	total_
1	30	1995-07-14	460	460
2	10	1996-08-02	540	2940
2	40	1998-01-29	2400	2940
7	50	1998-02-03	600	1470
7	60	1998-03-02	720	1470
7	70	1998-05-06	150	1470
8	20	1999-01-30	1800	1800

Order By

The **ORDER BY** clause within the OVER() window function determines the order in which the window function SUM(amount) is applied.

**For example `ORDER BY** order_date` is calculating a running total of the `amount` column.
This running total adds up the amounts in the order that the orders were made, based on their `order_date`.

```
SELECT salesperson_id, order_number, order_date, amount,
SUM(amount) OVER (ORDER BY order_date) AS total_sale
FROM int_orders;
```

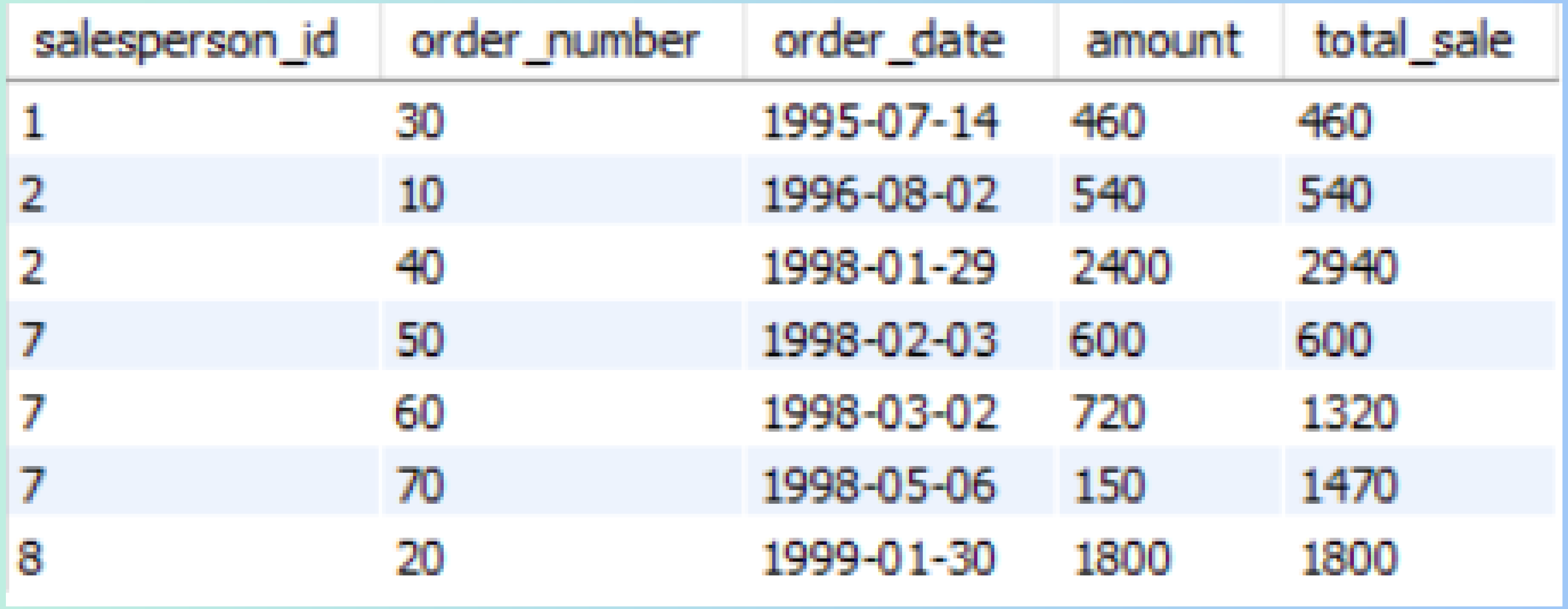


salesperson_id	order_number	order_date	amount	total_sale
1	30	1995-07-14	460	460
2	10	1996-08-02	540	1000
2	40	1998-01-29	2400	3400
7	50	1998-02-03	600	4000
7	60	1998-03-02	720	4720
7	70	1998-05-06	150	4870
8	20	1999-01-30	1800	6670

Partition By With Order By

PARTITION BY salesperson_id organizes the data into separate groups for each salesperson, and within each group, **ORDER BY** order_date arranges the rows in specified sequence based on when the orders were placed. .

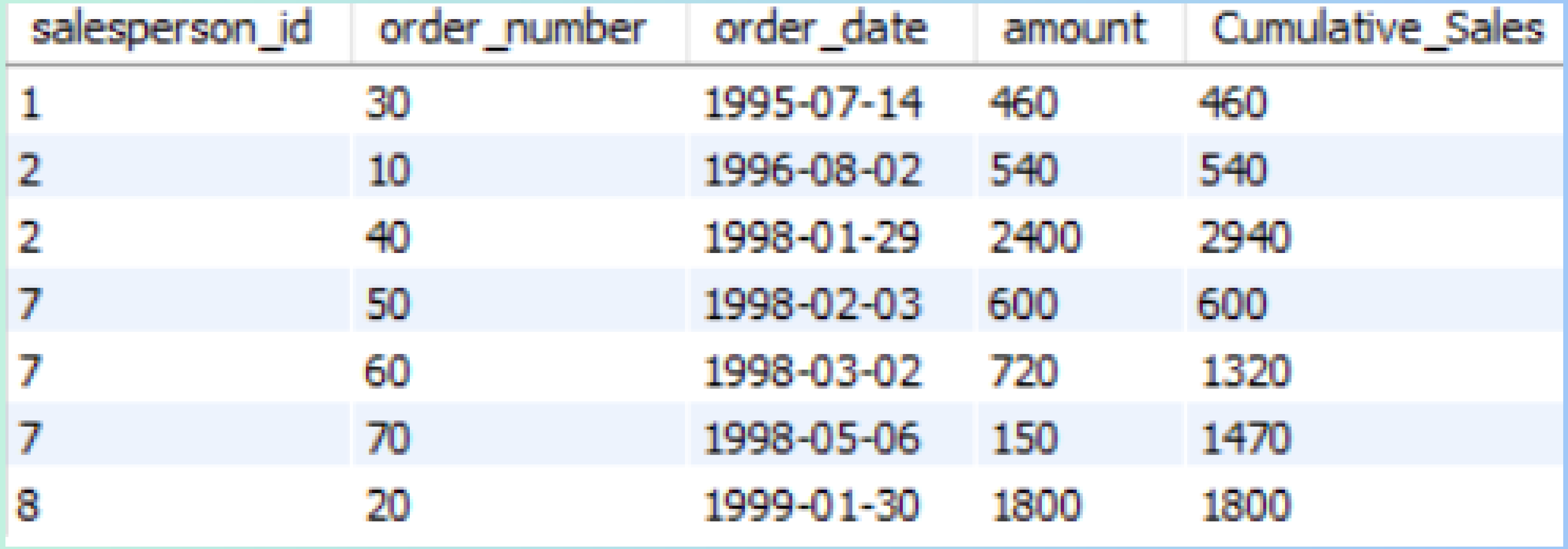
```
SELECT salesperson_id, order_number, order_date, amount,
SUM(amount) OVER (PARTITION BY salesperson_id ORDER BY order_date) AS total_sale
FROM int_orders;
```



salesperson_id	order_number	order_date	amount	total_sale
1	30	1995-07-14	460	460
2	10	1996-08-02	540	540
2	40	1998-01-29	2400	2940
7	50	1998-02-03	600	600
7	60	1998-03-02	720	1320
7	70	1998-05-06	150	1470
8	20	1999-01-30	1800	1800

What is the 3-month moving average of order amounts for each salesperson?

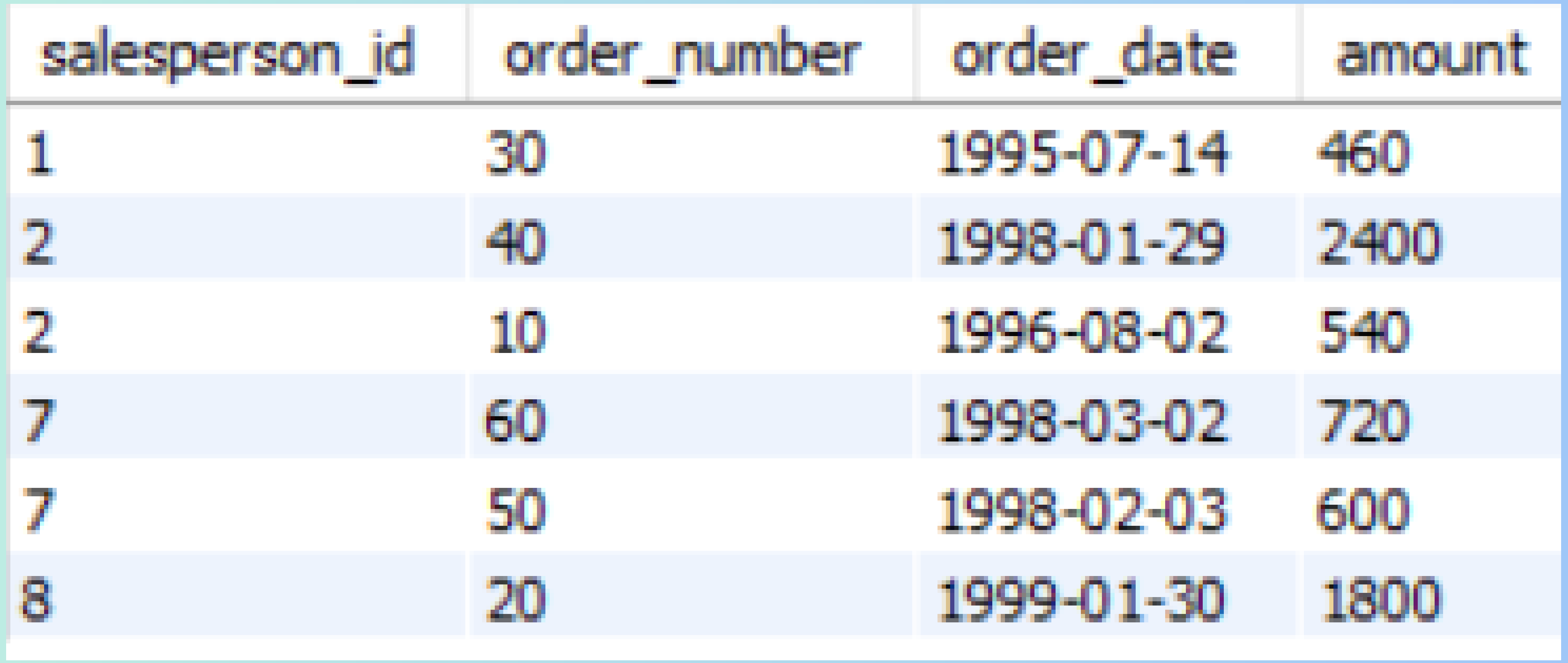
```
SELECT salesperson_id,order_number, order_date,amount,
AVG(amount) OVER (PARTITION BY salesperson_id ORDER BY order_date ROWS
BETWEEN 2 PRECEDING AND CURRENT ROW) AS moving_avg
FROM orders;
```



salesperson_id	order_number	order_date	amount	Cumulative_Sales
1	30	1995-07-14	460	460
2	10	1996-08-02	540	540
2	40	1998-01-29	2400	2940
7	50	1998-02-03	600	600
7	60	1998-03-02	720	1320
7	70	1998-05-06	150	1470
8	20	1999-01-30	1800	1800

Find the Top 2 Orders by Amount for Each Salesperson

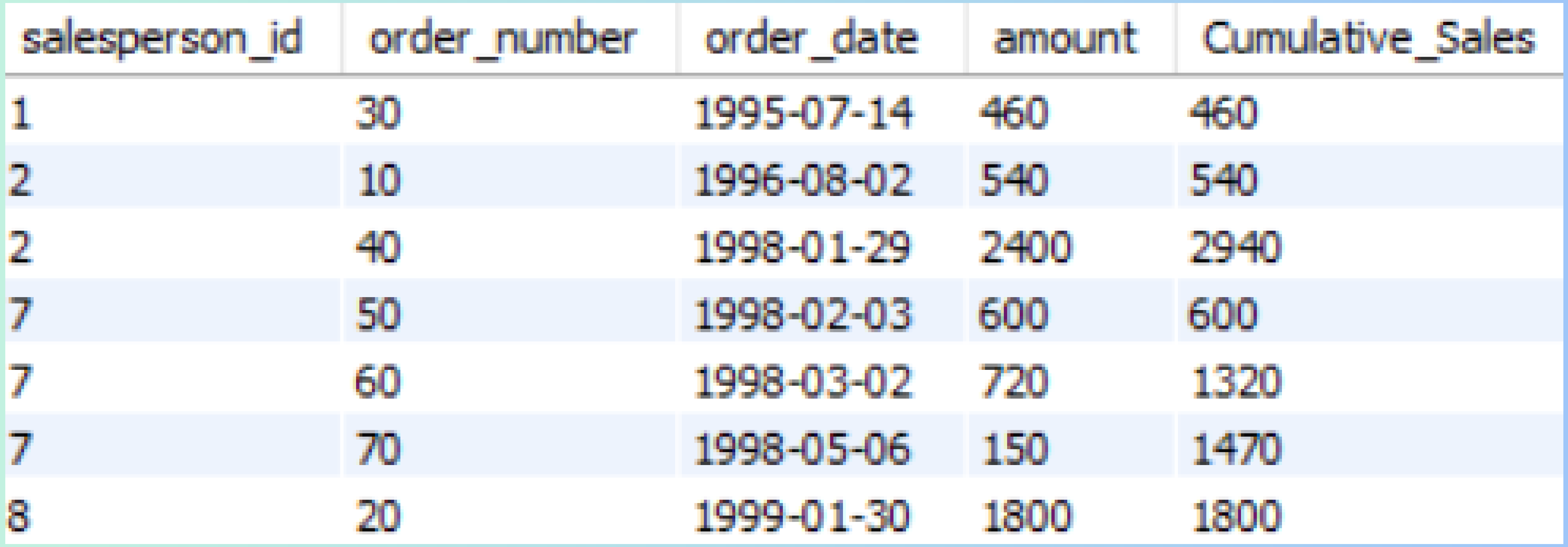
```
with rank_order as (select salesperson_id,  order_number, order_date, amount,
ROW_NUMBER() OVER (PARTITION BY salesperson_id ORDER BY amount DESC) as
row_num from int_orders)
select salesperson_id, order_number, order_date, amount
from rank_order
where row_num <= 2;
```



salesperson_id	order_number	order_date	amount
1	30	1995-07-14	460
2	40	1998-01-29	2400
2	10	1996-08-02	540
7	60	1998-03-02	720
7	50	1998-02-03	600
8	20	1999-01-30	1800

Calculate Cumulative Sales for Each Salesperson

```
select *,
    SUM(amount) OVER (PARTITION BY salesperson_id ORDER BY
order_date) as cumulative_sales
from int_orders;
```



salesperson_id	order_number	order_date	amount	Cumulative_Sales
1	30	1995-07-14	460	460
2	10	1996-08-02	540	540
2	40	1998-01-29	2400	2940
7	50	1998-02-03	600	600
7	60	1998-03-02	720	1320
7	70	1998-05-06	150	1470
8	20	1999-01-30	1800	1800

Thank You

Found Useful, Feel Free to Repost

Kanchan Prajapati