```
--  Q1) Running Total
SELECT
    user_id,
    order_date,
    amount,
    SUM(amount) OVER(PARTITION BY user_id ORDER BY order_date ROWS BETWEEN
UNBOUNDED PRECEDING AND CURRENT ROW) AS running_total
FROM orders
ORDER BY user_id, order_date;

-- Q2) First / Last Order
SELECT
    u.user_id,
    MIN(o.order_date) AS first_order_date,
    MAX(o.order_date) AS last_order_date
FROM users u
LEFT JOIN orders o ON o.user_id= u.user_id
GROUP BY u.user_id
ORDER BY u.user_id;
-- !! Siparişi olmayanlar için null görmek istediğimdizden LEFT JOIN


-- Q3) Rank vs Dense Rank

WITH user_spend AS(
    SELECT
        u.user_id,
        COALESCE(SUM(o.amount), 0) AS total_spent
    FROM users u
    LEFT JOIN orders o ON o.user_id = u.user_id
    GROUP BY u.user_id
)

SELECT
    user_id,
    total_spent,
    RANK() OVER (ORDER BY total_spent DESC) AS rank_spent,
    DENSE_RANK() OVER (ORDER BY total_spent DESC) as dense_rank_spent
FROM user_spend
ORDER BY total_spent DESC, user_id;


-- Q4) Churn (Last 30 days) - Include NULL
WITH last_orders AS (
    SELECT
        u.user_id,
        u.full_name,
        MAX(o.order_date) AS last_order_date
    FROM users u
    LEFT JOIN orders o ON o.user_id = u.user_id
    GROUP BY u.user_id, u.full_name
)
```

```sql
SELECT
    user_id,
    full_name,
    last_order_date,
    CASE
        WHEN last_order_date IS NULL THEN NULL
        ELSE (CURRENT_DATE - last_order_date)
    END AS days_since_last_order
FROM last_orders
WHERE last_order_date IS NULL
    OR last_order_date < (CURRENT_DATE - INTERVAL '30 days')
ORDER BY user_id;
```