```sql
-- Q1) Aktif aboneliği olan müşterileri plan adı ve koltuk ile listele
SELECT
    c.customer_id,
    c.full_name,
    p.plan_name,
    s.seats,
    s.start_date
FROM subscriptions s
JOIN customers c ON c.customer_id = s.customer_id
JOIN plans p ON p.plan_id = s.plan_id
WHERE s.status = 'active'
ORDER BY s.start_date;

--Q2) Hiç aboneliği olmayan müşteriler var mı?
SELECT
    c.customer_id,
    c.full_name,
    c.email
FROM customers c
LEFT JOIN subscriptions s ON s.customer_id = c.customer_id
WHERE s.subscription_id IS NULL;

--Q3) Her plan için aktif abone sayısı, toplam seat, aylık plan fiyati * aktif
abone sayısı
SELECT
    p.plan_name,
    COUNT(*) FILTER (WHERE s.status = 'active') AS active_subscribers,
    COALESCE(SUM(s.seats) FILTER (WHERE s.status = 'active'), 0) AS
total_active_seats,
    ROUND(p.monthly_price * COUNT(*) FILTER (WHERE s.status = 'active'),2) AS mrr
FROM plans p
LEFT JOIN subscriptions s ON s.plan_id = p.plan_id
GROUP BY p.plan_name, p.monthly_price
ORDER BY mrr DESC;

--Q4) 2026-01 ayı için toplam billed amount ve paid amount
WITH jan_invoices AS (
    SELECT
        invoice_id, amount_due
    FROM invoices
    WHERE invoice_month = '2026-01-01'
        AND status <> 'void'
),
jan_payments AS (
    SELECT
        invoice_id,
        SUM(amount_paid) AS paid_amount
    FROM payments
    GROUP BY invoice_id
)

SELECT
```

```
    ROUND(SUM(ji.amount_due), 2) AS total_billed,
    ROUND(COALESCE(SUM(jp.paid_amount), 0), 2) AS total_paid
FROM jan_invoices ji
LEFT JOIN jan_payments jp ON jp.invoice_id = ji.invoice_id;

--Q5) Ödenmemiş faturaları müşteri adı ve gecikme tarihi ile getir
WITH as_of AS (
    SELECT
        DATE '2026-01-19' AS as_of_date
)

SELECT
    c.full_name,
    i.invoice_id,
    i.invoice_month,
    i.amount_due,
    i.due_date,
    (a.as_of_date - i.due_date) AS days_overdue
FROM invoices i
JOIN subscriptions s ON s.subscription_id = i.subscription_id
JOIN customers c ON c.customer_id = s.customer_id
CROSS JOIN as_of a
WHERE i.status = 'open'
ORDER BY days_overdue DESC;

--Q6) Her müşterinni toplam lifetime revenue'u ve sıralaması
WITH paid_invoices AS (
    SELECT
        i.invoice_id,
        i.subscription_id,
        i.amount_due
    FROM invoices i
    WHERE i.status = 'paid'
),
customer_paid AS (
    SELECT
        c.customer_id,
        c.full_name,
        SUM(pi.amount_due) AS lifetime_revenue
    FROM paid_invoices pi
    JOIN subscriptions s ON s.subscription_id = pi.subscription_id
    JOIN customers C on c.customer_id = s.customer_id
    GROUP BY c.customer_id, c.full_name
)

SELECT
    *,
    DENSE_RANK() OVER(ORDER BY lifetime_revenue DESC) AS revenue_rank
FROM customer_paid
ORDER BY lifetime_revenue DESC;

-- Q7) Son 30 günde en çok API çağıranları listele
WITH params AS (
  SELECT DATE '2026-01-19' AS as_of_date
```

```sql
),

last30 AS (
    SELECT
        ue.customer_id,
        SUM(ue.quantity) AS api_calls_30d
    FROM usage_events ue
    CROSS JOIN params p
    WHERE ue.event_type = 'api_call'
        AND ue.event_date >= (p.as_of_date - INTERVAL '30 days')
        AND ue.event_date <= p.as_of_date
    GROUP BY ue.customer_id
)
SELECT
    c.full_name,
    ı.api_calls_30d

FROM last30 ı
JOIN customers c ON c.customer_id = ı.customer_id
ORDER BY ı.api_calls_30d DESC
LIMIT 3;

-- Q8) Churn listesi canceled abonelikler için churn tarihi ve kaç gün önce churn
oldu
WITH as_of AS(
    SELECT
        DATE '2026-01-19' AS as_of_date
)
SELECT
    c.full_name,
    p.plan_name,
    s.end_date AS churn_date,
    (a.as_of_date - s.end_date) AS days_since_churn
FROM subscriptions s
JOIN customers c ON c.customer_id = s.customer_id
JOIN plans p ON p.plan_id = s.plan_id
CROSS JOIN as_of a
WHERE s.status = 'canceled'
    AND s.end_date IS NOT NULL
ORDER BY days_since_churn DESC;

-- Q9) Müşteri bazında son fatura durumunu getir
WITH inv AS (
    SELECT
        c.customer_id,
        c.full_name,
        i.invoice_id,
        i.invoice_month,
        i.status,
        ROW_NUMBER() OVER(PARTITION BY c.customer_id ORDER BY i.invoice_month
DESC, i.invoice_id DESC) AS rn
    FROM customers c
    JOIN subscriptions s ON s.customer_id = c.customer_id
    JOIN invoices i ON i.subscription_id = s.subscription_id
```

```sql
)
SELECT
    customer_id,
    full_name,
    invoice_id AS latest_invoice_id,
    invoice_month AS latest_invoice_month,
    status AS latest_invoice_status
FROM inv
WHERE rn = 1
ORDER BY customer_id;

--Q10) Risk segmentasyonu
WITH as_of AS(
    SELECT
        DATE '2026-01-19' AS as_of_date
),
usage30 AS(
    SELECT
        ue.customer_id,
        SUM(CASE WHEN ue.event_type = 'api_call' THEN ue.quantity ELSE 0 END) AS
api_calls_30d
    FROM usage_events ue
    CROSS JOIN as_of a
    WHERE ue.event_date >= (a.as_of_date - INTERVAL '30 days')
        AND ue.event_date <= a.as_of_date
    GROUP BY ue.customer_id
),
active_customers AS (
    SELECT
        DISTINCT customer_id
    FROM subscriptions
    WHERE status = 'active'
)

SELECT
    c.customer_id,
    c.full_name,
    COALESCE(u.api_calls_30d, 0) AS api_calls_30d,
    CASE
        WHEN COALESCE(u.api_calls_30d, 0) = 0 THEN 'At Risk'
        ELSE 'Healthy'
    END AS customer_health
FROM active_customers ac
JOIN customers c ON c.customer_id = ac.customer_id
LEFT JOIN usage30 u ON u.customer_id = c.customer_id
ORDER BY customer_health, api_calls_30d;
```