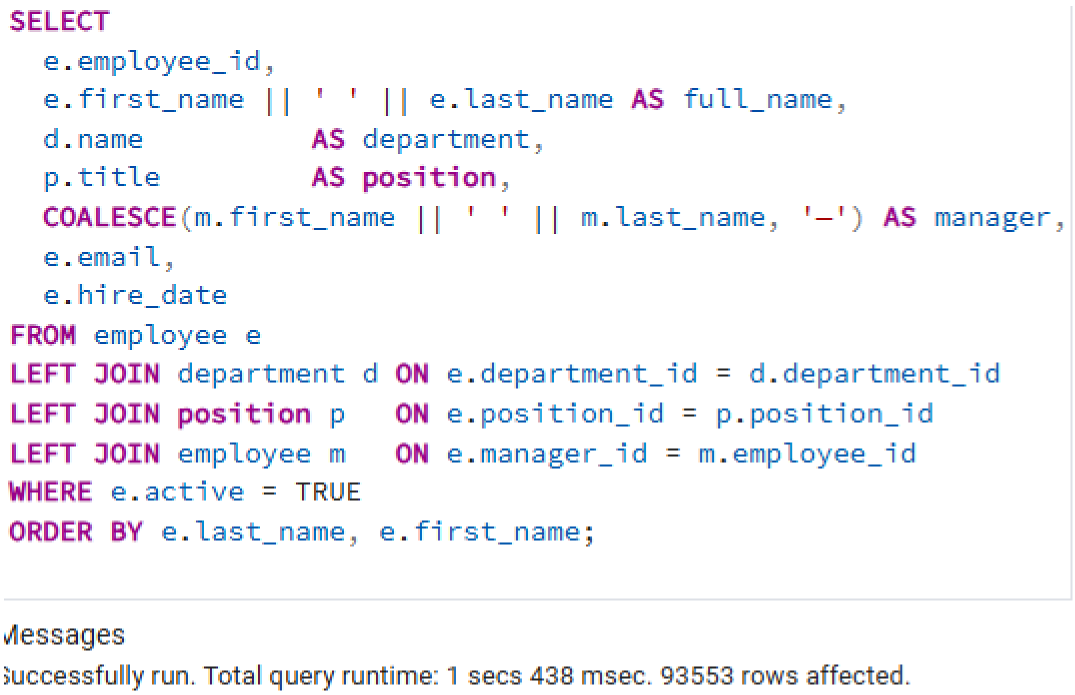
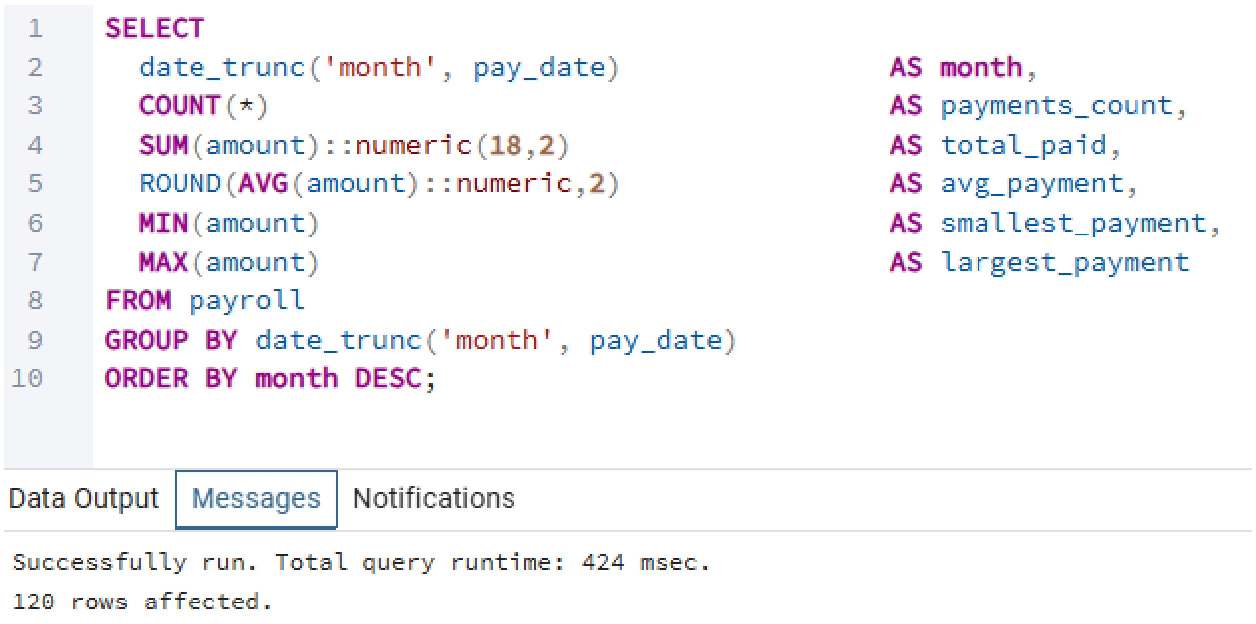
Q1



```
SELECT
  e.employee_id,
  e.first_name || ' ' || e.last_name AS full_name,
  d.name            AS department,
  p.title           AS position,
  COALESCE(m.first_name || ' ' || m.last_name, '—') AS manager,
  e.email,
  e.hire_date
FROM employee e
LEFT JOIN department d ON e.department_id = d.department_id
LEFT JOIN position p   ON e.position_id = p.position_id
LEFT JOIN employee m   ON e.manager_id = m.employee_id
WHERE e.active = TRUE
ORDER BY e.last_name, e.first_name;
```

```
Messages
Successfully run. Total query runtime: 1 secs 438 msec. 93553 rows affected.
```

Q2

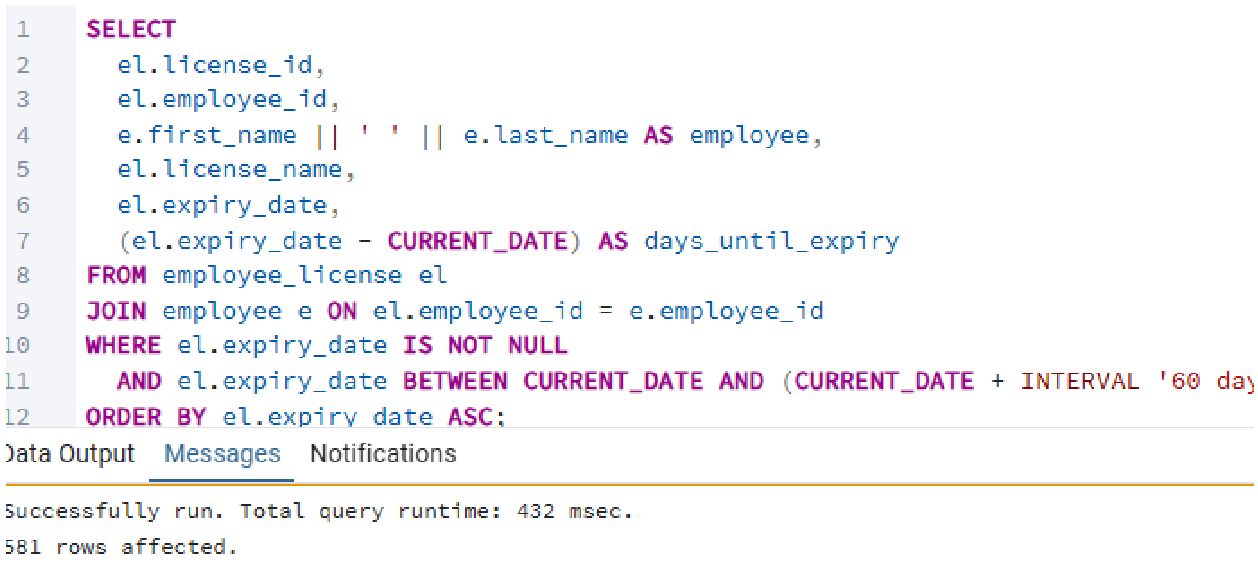


```
SELECT
  date_trunc('month', pay_date)                AS month,
  COUNT(*)                                     AS payments_count,
  SUM(amount)::numeric(18,2)                   AS total_paid,
  ROUND(AVG(amount)::numeric,2)                AS avg_payment,
  MIN(amount)                                  AS smallest_payment,
  MAX(amount)                                  AS largest_payment
FROM payroll
GROUP BY date_trunc('month', pay_date)
ORDER BY month DESC;
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 424 msec.
120 rows affected.
```

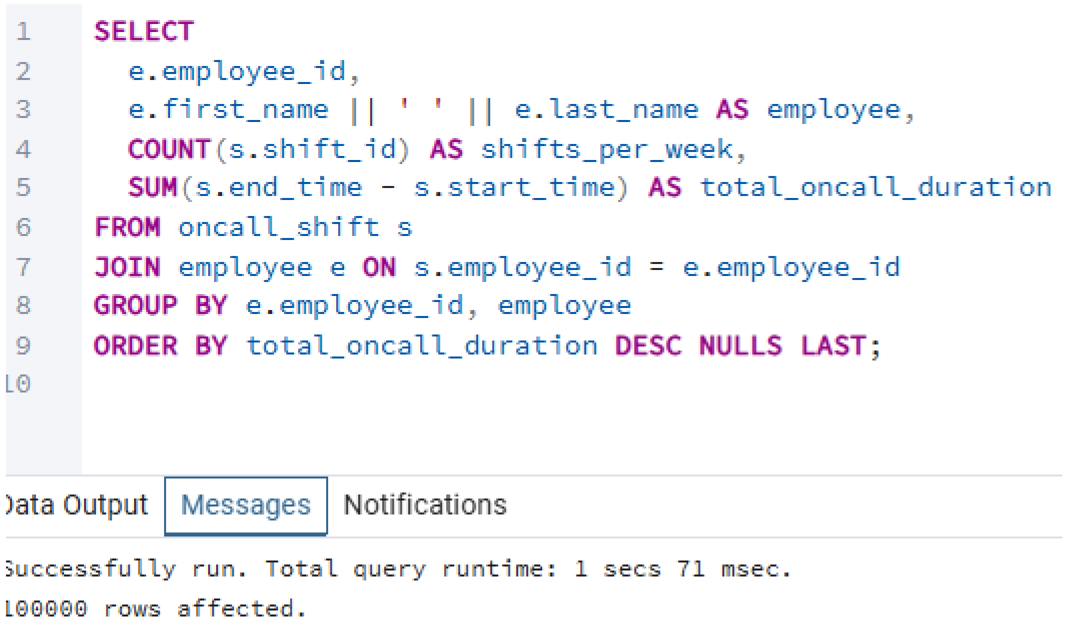
Q3



```sql
SELECT
  el.license_id,
  el.employee_id,
  e.first_name || ' ' || e.last_name AS employee,
  el.license_name,
  el.expiry_date,
  (el.expiry_date - CURRENT_DATE) AS days_until_expiry
FROM employee_license el
JOIN employee e ON el.employee_id = e.employee_id
WHERE el.expiry_date IS NOT NULL
  AND el.expiry_date BETWEEN CURRENT_DATE AND (CURRENT_DATE + INTERVAL '60 day
ORDER BY el.expiry_date ASC;
```

```
Successfully run. Total query runtime: 432 msec.
581 rows affected.
```

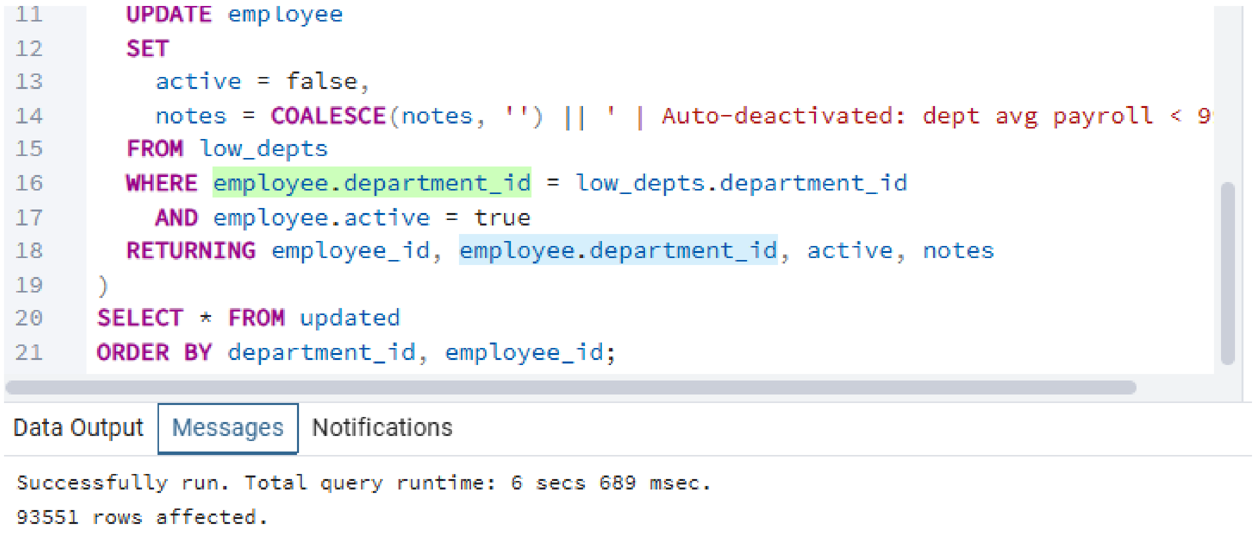
Q4



```sql
SELECT
  e.employee_id,
  e.first_name || ' ' || e.last_name AS employee,
  COUNT(s.shift_id) AS shifts_per_week,
  SUM(s.end_time - s.start_time) AS total_oncall_duration
FROM oncall_shift s
JOIN employee e ON s.employee_id = e.employee_id
GROUP BY e.employee_id, employee
ORDER BY total_oncall_duration DESC NULLS LAST;
```

```
Successfully run. Total query runtime: 1 secs 71 msec.
100000 rows affected.
```

Q5

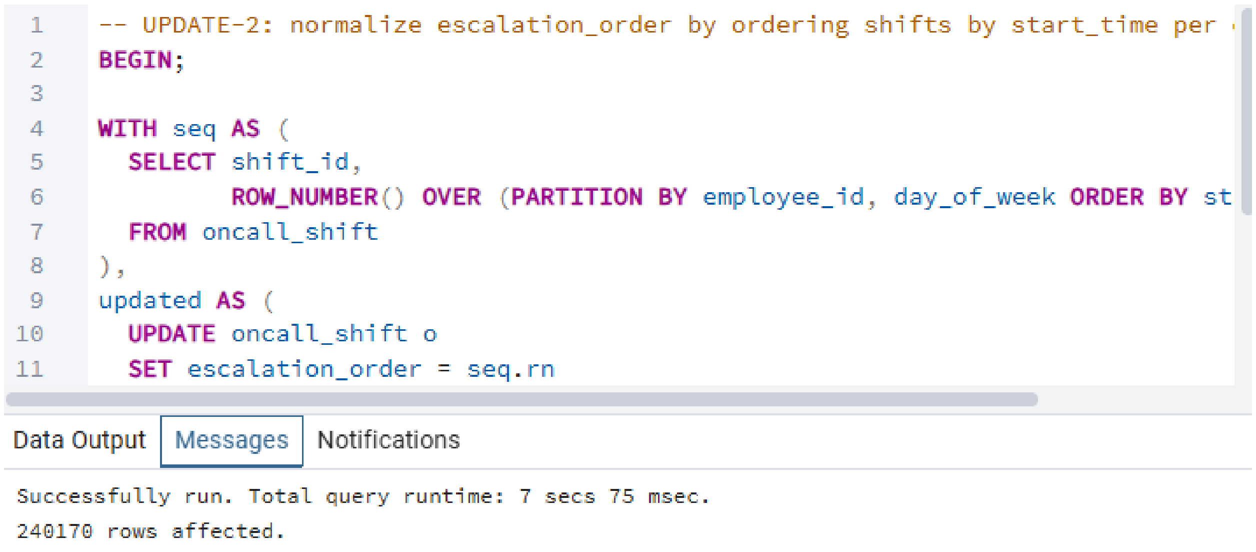


```
  UPDATE employee
  SET
    active = false,
    notes = COALESCE(notes, '') || ' | Auto-deactivated: dept avg payroll < 9
  FROM low_depts
  WHERE employee.department_id = low_depts.department_id
    AND employee.active = true
  RETURNING employee_id, employee.department_id, active, notes
)
SELECT * FROM updated
ORDER BY department_id, employee_id;
```

Data Output   Messages   Notifications

```
Successfully run. Total query runtime: 6 secs 689 msec.
93551 rows affected.
```

Q6

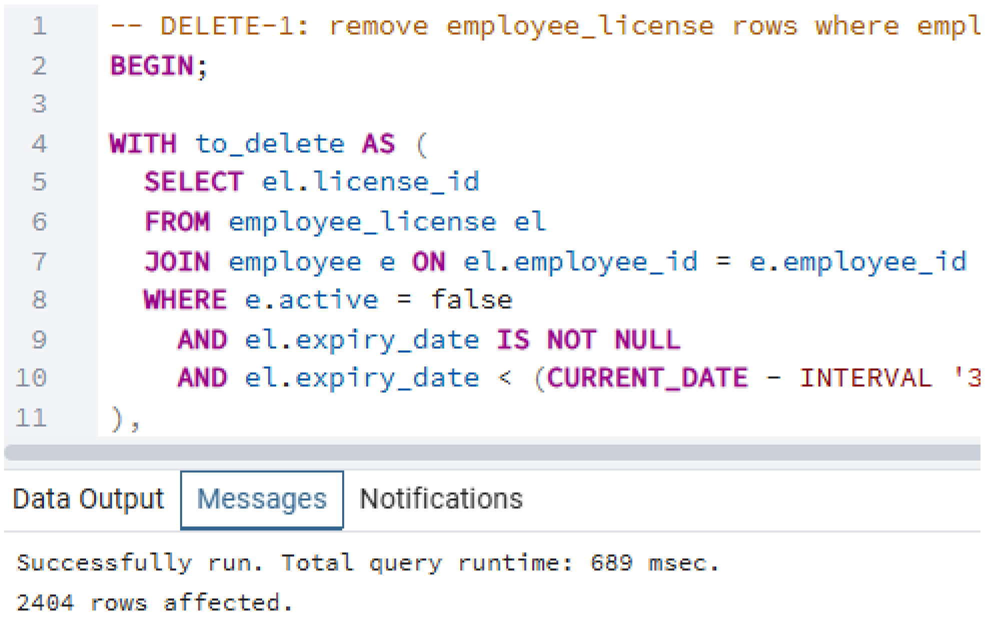


```
-- UPDATE-2: normalize escalation_order by ordering shifts by start_time per
BEGIN;

WITH seq AS (
  SELECT shift_id,
         ROW_NUMBER() OVER (PARTITION BY employee_id, day_of_week ORDER BY st
  FROM oncall_shift
),
updated AS (
  UPDATE oncall_shift o
  SET escalation_order = seq.rn
```

Data Output   Messages   Notifications

```
Successfully run. Total query runtime: 7 secs 75 msec.
240170 rows affected.
```

Q7

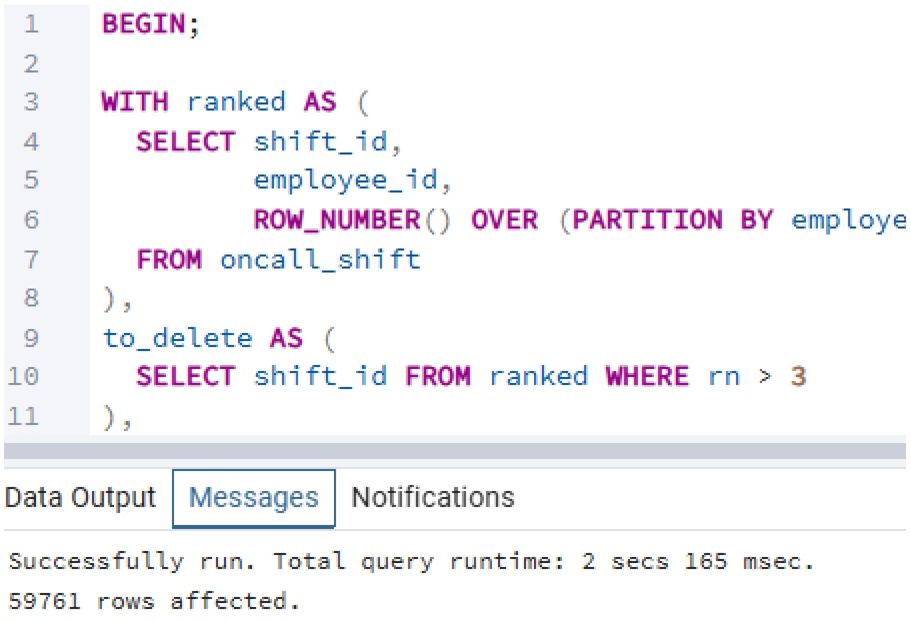


```
-- DELETE-1: remove employee_license rows where empl
BEGIN;

WITH to_delete AS (
  SELECT el.license_id
  FROM employee_license el
  JOIN employee e ON el.employee_id = e.employee_id
  WHERE e.active = false
    AND el.expiry_date IS NOT NULL
    AND el.expiry_date < (CURRENT_DATE - INTERVAL '3
),
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 689 msec.
2404 rows affected.
```

Q8



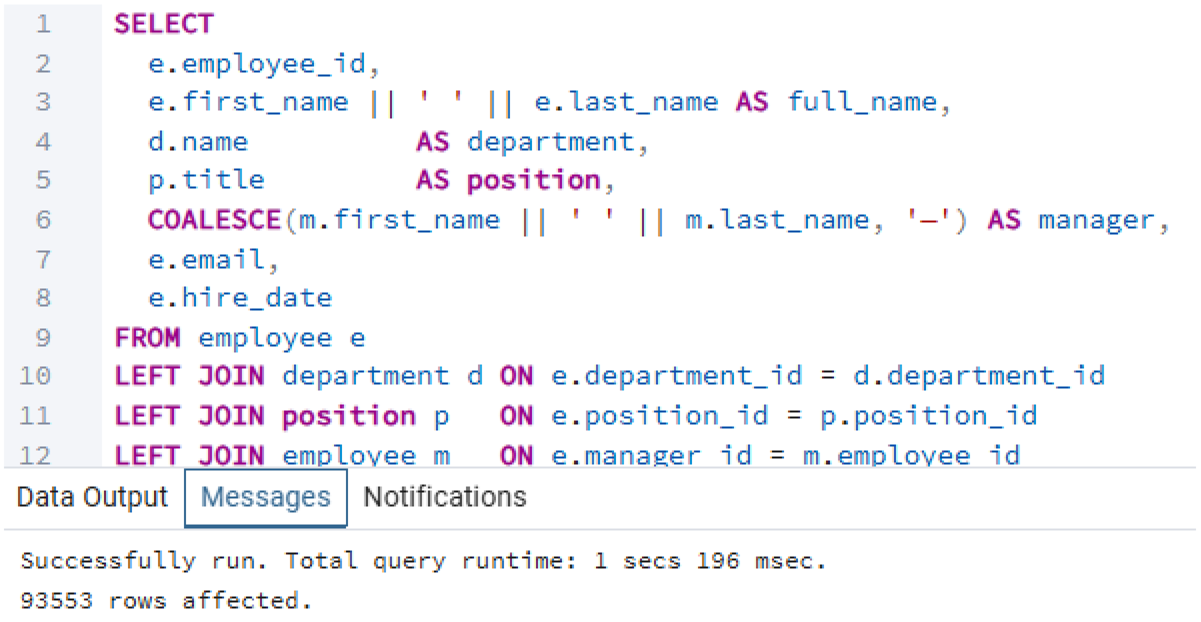
```
BEGIN;

WITH ranked AS (
  SELECT shift_id,
         employee_id,
         ROW_NUMBER() OVER (PARTITION BY employe
  FROM oncall_shift
),
to_delete AS (
  SELECT shift_id FROM ranked WHERE rn > 3
),
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 2 secs 165 msec.
59761 rows affected.
```

After Index (AI):

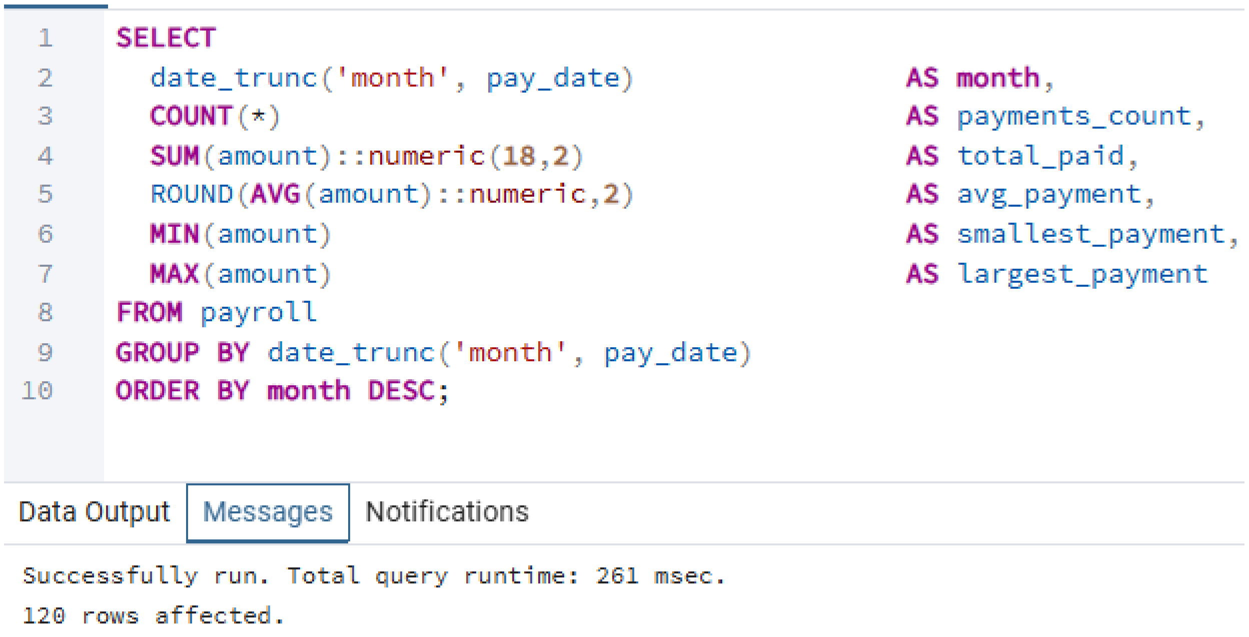
Q1



```
SELECT
  e.employee_id,
  e.first_name || ' ' || e.last_name AS full_name,
  d.name           AS department,
  p.title          AS position,
  COALESCE(m.first_name || ' ' || m.last_name, '-') AS manager,
  e.email,
  e.hire_date
FROM employee e
LEFT JOIN department d ON e.department_id = d.department_id
LEFT JOIN position p   ON e.position_id = p.position_id
LEFT JOIN employee m   ON e.manager_id = m.employee_id
```

```
Successfully run. Total query runtime: 1 secs 196 msec.
93553 rows affected.
```

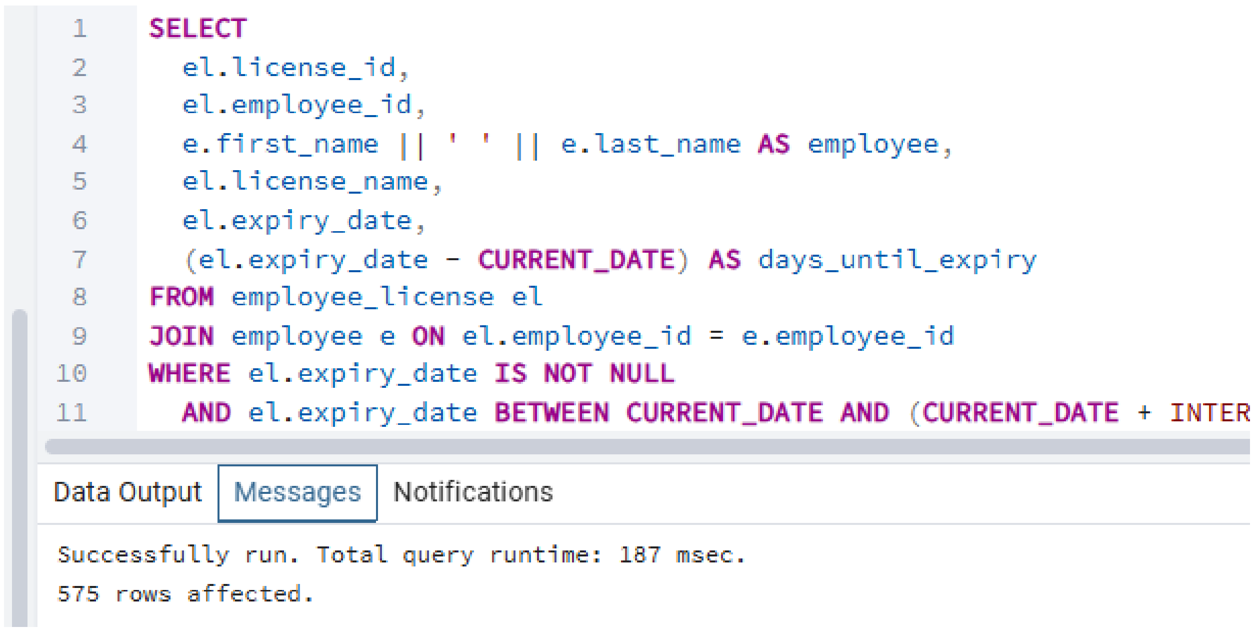
Q2



```
SELECT
  date_trunc('month', pay_date)                  AS month,
  COUNT(*)                                       AS payments_count,
  SUM(amount)::numeric(18,2)                     AS total_paid,
  ROUND(AVG(amount)::numeric,2)                  AS avg_payment,
  MIN(amount)                                    AS smallest_payment,
  MAX(amount)                                    AS largest_payment
FROM payroll
GROUP BY date_trunc('month', pay_date)
ORDER BY month DESC;
```

```
Successfully run. Total query runtime: 261 msec.
120 rows affected.
```

Q3

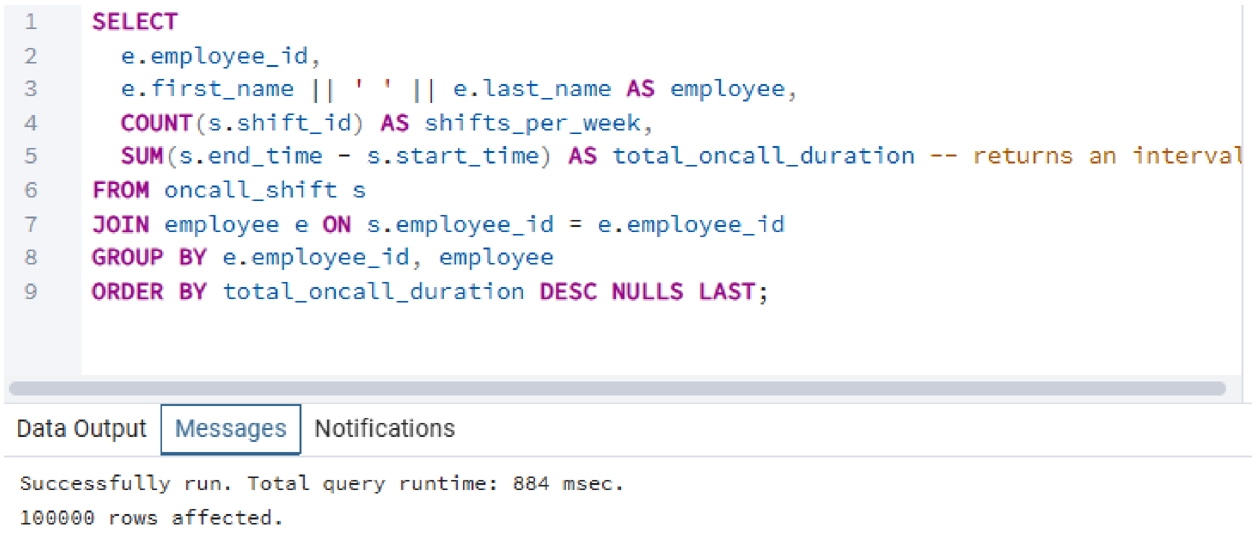


```sql
SELECT
  el.license_id,
  el.employee_id,
  e.first_name || ' ' || e.last_name AS employee,
  el.license_name,
  el.expiry_date,
  (el.expiry_date - CURRENT_DATE) AS days_until_expiry
FROM employee_license el
JOIN employee e ON el.employee_id = e.employee_id
WHERE el.expiry_date IS NOT NULL
  AND el.expiry_date BETWEEN CURRENT_DATE AND (CURRENT_DATE + INTER
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 187 msec.
575 rows affected.
```

Q4

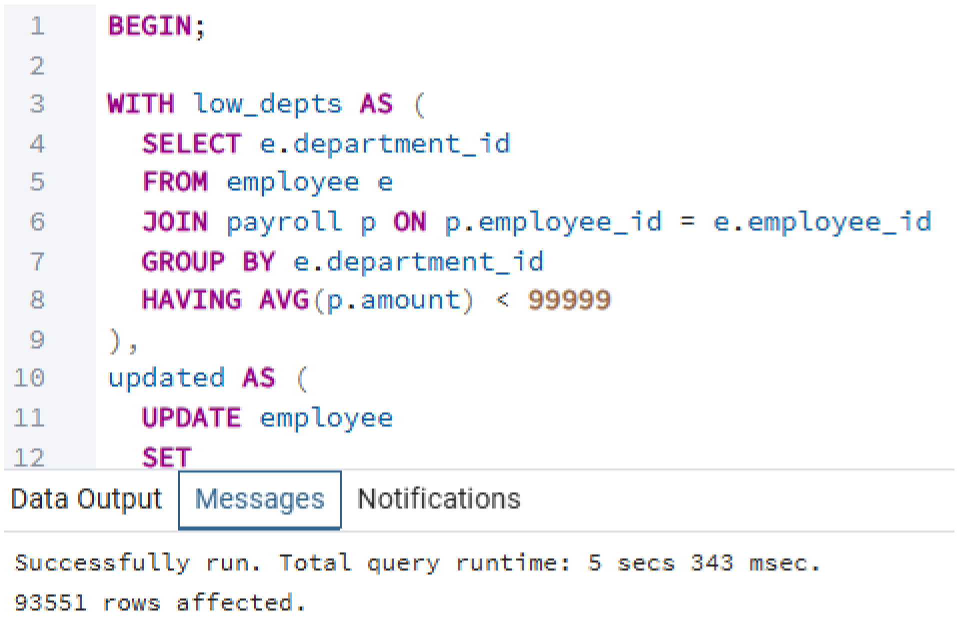


```sql
SELECT
  e.employee_id,
  e.first_name || ' ' || e.last_name AS employee,
  COUNT(s.shift_id) AS shifts_per_week,
  SUM(s.end_time - s.start_time) AS total_oncall_duration -- returns an interval
FROM oncall_shift s
JOIN employee e ON s.employee_id = e.employee_id
GROUP BY e.employee_id, employee
ORDER BY total_oncall_duration DESC NULLS LAST;
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 884 msec.
100000 rows affected.
```

Q5

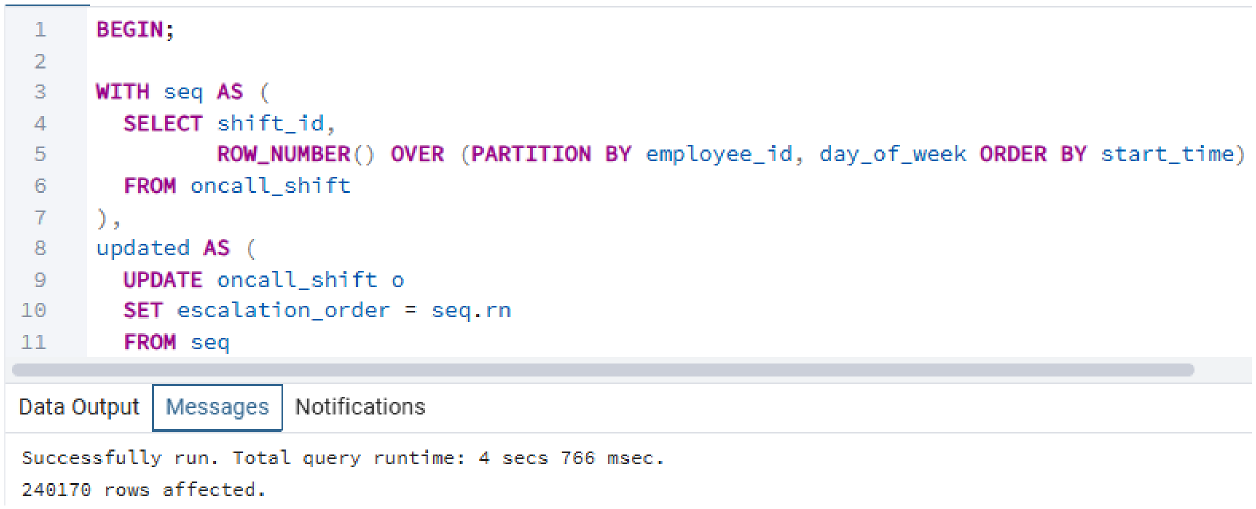


```
BEGIN;

WITH low_depts AS (
  SELECT e.department_id
  FROM employee e
  JOIN payroll p ON p.employee_id = e.employee_id
  GROUP BY e.department_id
  HAVING AVG(p.amount) < 99999
),
updated AS (
  UPDATE employee
  SET
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 5 secs 343 msec.
93551 rows affected.
```

Q6

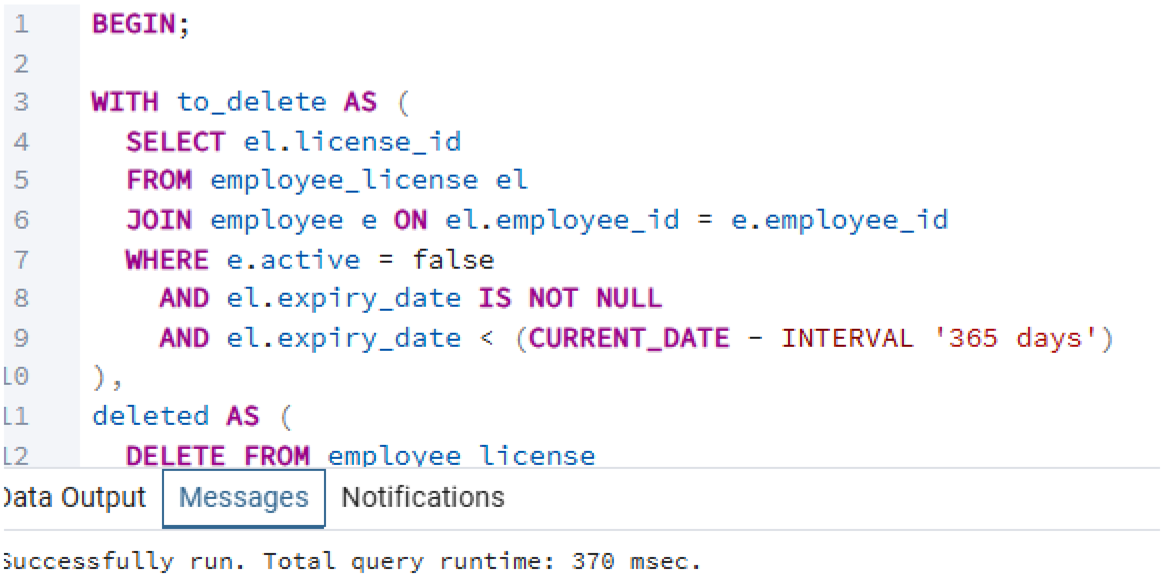


```
BEGIN;

WITH seq AS (
  SELECT shift_id,
         ROW_NUMBER() OVER (PARTITION BY employee_id, day_of_week ORDER BY start_time)
  FROM oncall_shift
),
updated AS (
  UPDATE oncall_shift o
  SET escalation_order = seq.rn
  FROM seq
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 4 secs 766 msec.
240170 rows affected.
```

Q7

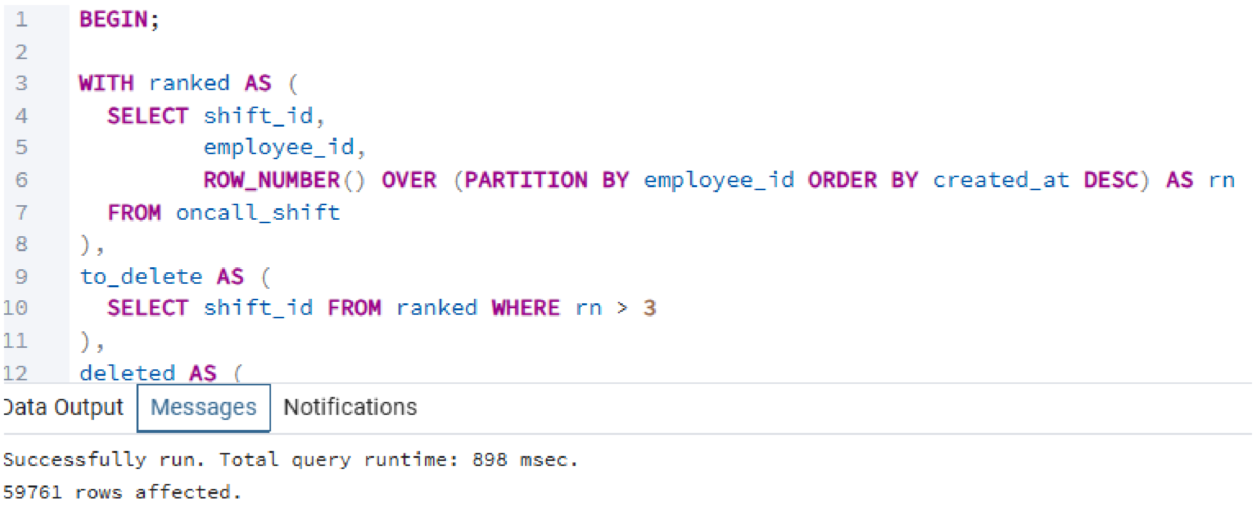


```
BEGIN;

WITH to_delete AS (
  SELECT el.license_id
  FROM employee_license el
  JOIN employee e ON el.employee_id = e.employee_id
  WHERE e.active = false
    AND el.expiry_date IS NOT NULL
    AND el.expiry_date < (CURRENT_DATE - INTERVAL '365 days')
),
deleted AS (
  DELETE FROM employee_license
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 370 msec.
```

Q8



```
BEGIN;

WITH ranked AS (
  SELECT shift_id,
         employee_id,
         ROW_NUMBER() OVER (PARTITION BY employee_id ORDER BY created_at DESC) AS rn
  FROM oncall_shift
),
to_delete AS (
  SELECT shift_id FROM ranked WHERE rn > 3
),
deleted AS (
```

Data Output | Messages | Notifications

```
Successfully run. Total query runtime: 898 msec.
59761 rows affected.
```