Day 2 — Advanced SQL Practice + Quiz (With Answers)

PART 1 — Day-2 Practice (With Answers)

Q1. Top 3 highest salaries (no LIMIT)
Answer:

```
SELECT name, salary FROM (
  SELECT name, salary,
    ROW_NUMBER() OVER (ORDER BY salary DESC) AS rn
  FROM employees
) t WHERE rn <= 3;
```

Q2. Second-highest salary per department
Answer:

```
SELECT department, name, salary FROM (
  SELECT department, name, salary,
    ROW_NUMBER() OVER (PARTITION BY department ORDER BY salary DESC) rn
  FROM employees
) t WHERE rn = 2;
```

Q3. Employees earning above department average
Answer:

```
SELECT e.name, e.department, e.salary FROM employees e
WHERE e.salary > (
  SELECT AVG(e2.salary) FROM employees e2
  WHERE e2.department = e.department
);
```

Q4. Running total of salaries
Answer:

```
SELECT name, salary,
  SUM(salary) OVER (ORDER BY salary) AS running_total
FROM employees;
```

PART 2 — Day-2 Advanced SQL Quiz (With Answers)

Q1. Difference between RANK() and DENSE_RANK()?
Answer: RANK() skips numbers on ties; DENSE_RANK() does not skip.

Q2. What is a correlated subquery?
Answer: A subquery that depends on the outer query and runs once per row.

Q3. Write SQL to find 3rd highest salary (no LIMIT).
Answer: Use ROW_NUMBER() and filter rn = 3.

Q4. What is an index and why is it used?
Answer: An index speeds up searches on WHERE/JOIN/ORDER BY columns.

Q5. Query to get first-rank salary per department.
Answer:

```
SELECT * FROM (
  SELECT *, ROW_NUMBER() OVER (PARTITION BY department ORDER BY salary DESC) rn
  FROM employees
) t WHERE rn = 1;
```

Q6. What does PARTITION BY do?
Answer: Divides rows into groups so window functions apply separately in each group.

Q7. Query to find employees with duplicate salaries.
Answer:

```
SELECT * FROM (
  SELECT *, COUNT(*) OVER (PARTITION BY salary) cnt
  FROM employees
) t WHERE cnt > 1;
```

Q8. Difference between GROUP BY and WINDOW functions?
Answer: GROUP BY collapses rows; window functions do not collapse rows.

Q9. Query to calculate moving average of salary.
Answer:

```
SELECT name, salary, AVG(salary) OVER (ORDER BY salary ROWS 2 PRECEDING) AS moving_avg
FROM employees;
```

Q10. What is a CTE?
Answer: A temporary result set defined using WITH to simplify queries.