HR Data Example SQL Code

==========================================================

Example SQL Query for Portfolio Demonstration

Purpose: Showcase UNION, JOIN, CASE, GROUP BY,

WHERE IN, LISTAGG, RANK, and ORDER BY

Database Theme: HR + Project database

==========================================================

----------------------------------------------------------

-- Step 1: Combine employees from two different systems

-- (HR upgraded, but legacy data still exists).

-- UNION ensures duplicate employees (if any) are collapsed.

----------------------------------------------------------

WITH all\_employees AS (

SELECT

employee\_id, employee\_name, title, department, start\_date, salary

FROM hr\_employees

UNION -- merge with legacy system employees

SELECT

employee\_id, employee\_name, title, department, start\_date, salary

FROM legacy\_employees

),

----------------------------------------------------------

-- Step 2: Join employees with project assignments

-- This enriches employee records with project data

-- such as project name, role, and start date.

----------------------------------------------------------

employee\_projects AS (

SELECT

e.employee\_id, e.employee\_name, e.title, e.department, e.start\_date, e.salary,

-- Project-related fields come from the 'projects' table

p.project\_name, p.project\_role, p.project\_start\_date

FROM all\_employees e

LEFT JOIN projects p ON e.employee\_id = p.employee\_id

-- LEFT JOIN ensures all employees appear,

-- even if they are not currently assigned to a project

)

----------------------------------------------------------

-- Step 3: Perform final analysis

-- Aggregate, classify, and rank employees by salary

-- while displaying project information.

----------------------------------------------------------

SELECT

ep.department, -- Group results by department

------------------------------------------------------

-- Create salary bands

------------------------------------------------------

CASE

WHEN ep.salary >= 100000 THEN 'High Earner'

WHEN ep.salary BETWEEN 60000 AND 99999 THEN 'Mid Earner'

ELSE 'Entry Level'

END AS salary\_band,

------------------------------------------------------

-- LISTAGG: Collect project names into a single field

-- (comma-separated), so we don’t get one row

-- per project but a consolidated list.

------------------------------------------------------

LISTAGG(ep.project\_name, ', ')

WITHIN GROUP (ORDER BY ep.project\_name)

AS project\_list,

------------------------------------------------------

-- Assign ranking within each department based on salary, highest salary = rank 1.

------------------------------------------------------

RANK() OVER (

PARTITION BY ep.department

ORDER BY ep.salary DESC

) AS salary\_rank,

------------------------------------------------------

-- Aggregations

------------------------------------------------------

COUNT(\*) AS employee\_count, -- how many employees in the group

AVG(ep.salary) AS avg\_salary -- average salary per group

FROM employee\_projects ep

----------------------------------------------------------

-- Filter results to a subset of departments only

----------------------------------------------------------

WHERE ep.department IN ('Engineering', 'Finance', 'HR')

----------------------------------------------------------

-- GROUP BY: Required because of CASE WHEN + aggregations

----------------------------------------------------------

GROUP BY

ep.department,

CASE

WHEN ep.salary >= 100000 THEN 'High Earner'

WHEN ep.salary BETWEEN 60000 AND 99999 THEN 'Mid Earner'

ELSE 'Entry Level'

END

----------------------------------------------------------

-- Departments first, then by highest average salary.

----------------------------------------------------------

ORDER BY ep.department, avg\_salary DESC;