**Lab Exercises**

**Departments Info by ID**

SELECT

    department\_id,

    count(\*) AS employee\_count

FROM

    employees

GROUP BY

    department\_id

ORDER BY

    department\_id

;

**Departments Info by Salary**

SELECT

    department\_id,

    count(salary) AS employee\_count

FROM

    employees

GROUP BY

    department\_id

ORDER BY

    department\_id

;

**Sum Salaries per Department**

SELECT

    department\_id,

    sum(salary) AS "total\_salaries"

FROM

    employees

GROUP BY

    department\_id

ORDER BY

    department\_id

;

**Maximum Salary**

SELECT

    department\_id,

    max(salary) AS "max\_salary"

FROM

    employees

GROUP BY

    department\_id

ORDER BY

    department\_id

;

**Minimum Salary**

SELECT

    department\_id,

    min(salary) AS "min\_salary"

FROM

    employees

GROUP BY

    department\_id

ORDER BY

    department\_id

;

**Average Salary**

SELECT

    department\_id,

    avg(salary) AS "avg\_salary"

FROM

    employees

GROUP BY

    department\_id

ORDER BY

    department\_id

;

**Filter Total Salaries**

SELECT

    department\_id,

    sum(salary) AS "Total Salary"

FROM

    employees

GROUP BY

    department\_id

HAVING

    sum(salary) < 4200

ORDER BY

    department\_id

;

**Department Names**

SELECT

    id,

    first\_name,

    last\_name,

    TRUNC(salary, 2),

    department\_id,

    CASE

        WHEN department\_id = 1 THEN 'Management'

        WHEN department\_id = 2 THEN 'Kitchen Staff'

        WHEN department\_id = 3 THEN 'Service Staff'

        ELSE 'Other'

    END AS "department\_name"

FROM

    employees

;

**Homework Exercises**

**GROUP BY Deposit Interest**

SELECT

    deposit\_group,

    SUM(deposit\_interest) AS deposit\_interest

FROM

    wizard\_deposits

GROUP BY

    deposit\_group

ORDER BY

    SUM(deposit\_interest) DESC

;

**LIMIT the Magic Wand Creator**

SELECT

    magic\_wand\_creator,

    MIN(magic\_wand\_size) AS "minimum\_wand\_size"

FROM

    wizard\_deposits

GROUP BY

    magic\_wand\_creator

ORDER BY

    minimum\_wand\_size

LIMIT 5

;

**Bank Profitability**

SELECT

    deposit\_group,

    is\_deposit\_expired,

    FLOOR(AVG(deposit\_interest)) AS deposit\_interest

FROM

    wizard\_deposits

WHERE

    deposit\_start\_date > '1985-01-01'

GROUP BY

    is\_deposit\_expired,

    deposit\_group

ORDER BY

    deposit\_group DESC,

    is\_deposit\_expired

;

**Notes with Dumbledore**

SELECT

    last\_name,

    COUNT(notes) AS "notes\_with\_dumbledore"

FROM

    wizard\_deposits

WHERE

    notes LIKE '%Dumbledore%'

GROUP BY

    last\_name

;

**Wizard View**

CREATE VIEW view\_wizard\_deposits\_with\_expiration\_date\_before\_1983\_08\_17 AS

SELECT

    CONCAT(first\_name, ' ', last\_name) AS "wizard\_name",

    deposit\_start\_date AS "start\_date",

    deposit\_expiration\_date AS "expiration\_date",

    deposit\_amount AS "amount"

FROM

    wizard\_deposits

WHERE

    deposit\_expiration\_date <= '1983-08-17'

GROUP BY

    wizard\_name,

    start\_date,

    expiration\_date,

    amount

ORDER BY

    expiration\_date ASC

;

**Filter Max Deposit**

SELECT

    magic\_wand\_creator,

    MAX(deposit\_amount) AS "max\_deposit\_amount"

FROM

    wizard\_deposits

GROUP BY

    magic\_wand\_creator

HAVING

    MAX(deposit\_amount) NOT BETWEEN 20000 AND 40000

ORDER BY

    max\_deposit\_amount DESC LIMIT 3

;

**Age Group**

SELECT

    CASE

        WHEN age BETWEEN 0 and 10 THEN '[0-10]'

        WHEN age BETWEEN 11 and 20 THEN '[11-20]'

        WHEN age BETWEEN 21 and 30 THEN '[21-30]'

        WHEN age BETWEEN 31 and 40 THEN '[31-40]'

        WHEN age BETWEEN 41 and 50 THEN '[41-50]'

        WHEN age BETWEEN 51 and 60 THEN '[51-60]'

        WHEN age >= 61 THEN '[61+]'

    END AS "age\_group",

    COUNT(\*)

FROM

    wizard\_deposits

GROUP BY

    age\_group

ORDER BY

    age\_group

;

**SUM the Employees**

SELECT

    COUNT(CASE WHEN department\_id = 1 THEN 1 END) AS "Engineering",

    COUNT(CASE WHEN department\_id = 2 THEN 2 END) AS "Tool Design",

    COUNT(CASE WHEN department\_id = 3 THEN 3 END) AS "Sales",

    COUNT(CASE WHEN department\_id = 4 THEN 4 END) AS "Marketing",

    COUNT(CASE WHEN department\_id = 5 THEN 5 END) AS "Purchasing",

    COUNT(CASE WHEN department\_id = 6 THEN 6 END) AS "Research and Development",

    COUNT(CASE WHEN department\_id = 7 THEN 7 END) AS "Production"

FROM

    employees

;

**Update Employees’ Data**

UPDATE

    employees

SET

    salary = CASE

        WHEN hire\_date < '2015-01-16' THEN salary + 2500

        WHEN hire\_date < '2020-03-04' THEN salary + 1500

        ELSE salary

    END,

    job\_title = CASE

        WHEN hire\_date < '2015-01-16' THEN CONCAT('Senior', ' ', job\_title)

        WHEN hire\_date < '2020-03-04' THEN CONCAT('Mid-', job\_title)

        ELSE job\_title

    END

;

**Categorizes Salary**

SELECT

    job\_title,

    CASE

        WHEN AVG(salary) > 45800 THEN 'Good'

        WHEN AVG(salary) BETWEEN 27500 AND 45800 THEN 'Medium'

        WHEN AVG(salary) < 27500 THEN 'Need Improvement'

    END AS "category"

FROM

    employees

GROUP BY

    job\_title

ORDER BY

    category,

    job\_title

;

**WHERE Project Status**

SELECT

    project\_name,

    CASE

        WHEN start\_date IS NULL AND end\_date IS NULL THEN 'Ready for development'

        WHEN start\_date IS NOT NULL AND end\_date IS NULL THEN 'In Progress'

        ELSE 'Done'

    END

FROM

    projects

WHERE

    project\_name LIKE '%Mountain%'

;

**HAVING Salary Level**

SELECT

    department\_id,

    COUNT(department\_id) AS "num\_employees",

    CASE

        WHEN AVG(salary) > 50000 THEN 'Above average'

        WHEN AVG(salary) <= 50000 THEN 'Below average'

    END AS "salary\_level"

FROM

    employees

GROUP BY

    department\_id

HAVING

    AVG(salary) > 30000

ORDER BY

    department\_id

;

**Nested CASE Conditions**

CREATE VIEW view\_performance\_rating AS

    SELECT

        first\_name,

        last\_name,

        job\_title,

        salary,

        department\_id,

        CASE

            WHEN salary >= 25000 THEN

                CASE

                    WHEN job\_title LIKE 'Senior%' THEN 'High-performing Senior'

                    ELSE 'High-performing Employee'

                END

            ELSE 'Average-performing'

        END AS "performance\_rating"

FROM

    employees

;