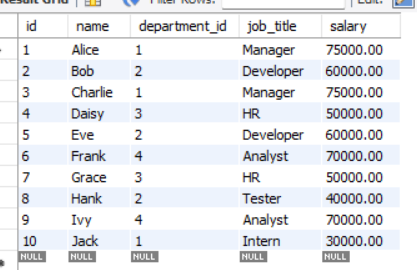
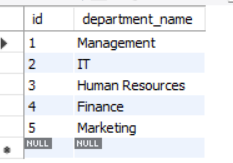
**SQL BASICS – TASK**

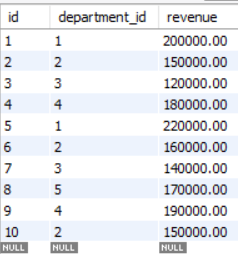
**EMPLOYEES TABLE**



**DEPARTMENTS TABLE**



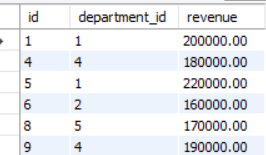
**SALES\_DATA TABLE**



1. **What is SQL, and why is it important in data analytics?**  
   SQL (Structured Query Language) is a programming language used to communicate with and manipulate relational databases. It is essential in data analytics for retrieving, managing, and analyzing data stored in databases to derive meaningful insights.

**QUERY EXAMPLE**

SELECT \* FROM sales\_data WHERE revenue > 150000;



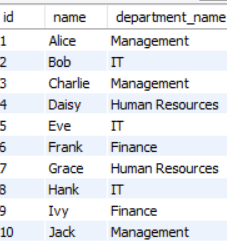
1. **Difference between INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN**

* **INNER JOIN:** Returns rows that have matching values in both tables.
* **LEFT JOIN:** Returns all rows from the left table and matching rows from the right table. If no match, NULL values are returned.
* **RIGHT JOIN:** Returns all rows from the right table and matching rows from the left table. If no match, NULL values are returned.
* **FULL OUTER JOIN:** Returns all rows when there is a match in either table. If there is no match, NULL values are returned.

**INNER JOIN**

select employees.id, employees.name, departments.department\_name

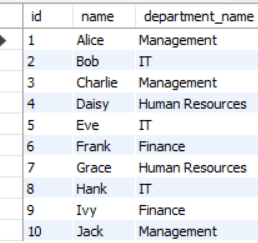
from employees inner join departments on employees.department\_id = departments.id;



**LEFT JOIN**

select employees.id, employees.name, departments.department\_name

from employees left join departments on employees.department\_id = departments.id;



**RIGHT JOIN**

select employees.id, employees.name, departments.department\_name

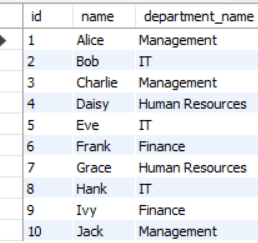
from employees right join departments on employees.department\_id = departments.id;



**FULL OUTER JOIN**

select employees.id, employees.name, departments.department\_name

from employees full outer join departments on employees.department\_id = departments.id;

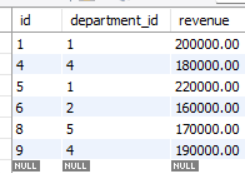


1. **Difference between WHERE and HAVING clauses**

* **WHERE:** Filters rows before any grouping is applied.
* **HAVING:** Filters rows after grouping is applied.

**WHERE Clause**

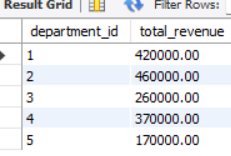
SELECT \* FROM sales\_data WHERE revenue > 150000;



**HAVING Clause**

SELECT department\_id, SUM(revenue) AS total\_revenue

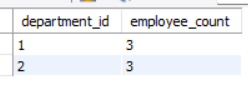
FROM sales\_data GROUP BY department\_id HAVING total\_revenue > 100000;



**4. Using GROUP BY and HAVING in a query**  
GROUP BY organizes data into groups based on one or more columns. HAVING filters these groups.

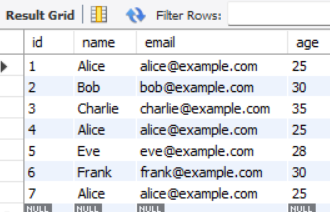
SELECT department\_id, COUNT(\*) AS employee\_count FROM employees

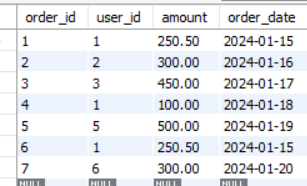
GROUP BY department\_id HAVING employee\_count > 2;



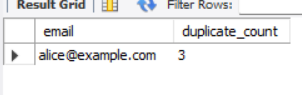
**5.Query to find duplicate records in a table**

Users table



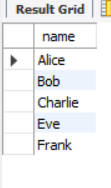


SELECT email, COUNT(\*) AS duplicate\_count FROM users GROUP BY email HAVING COUNT(\*) > 1;



**6.Retrieve unique values from a table using SQL**

SELECT DISTINCT name FROM users;



7. **Use of Aggregate Functions**

* **COUNT():** Counts the number of rows.
* **SUM():** Calculates the total of a numeric column.
* **AVG():** Calculates the average value.
* **MIN():** Finds the minimum value.
* **MAX():** Finds the maximum value.

SELECT

COUNT(\*) AS total\_orders,

SUM(amount) AS total\_revenue,

AVG(amount) AS average\_order\_amount,

MIN(amount) AS smallest\_order,

MAX(amount) AS largest\_order

FROM orders;

