**1. String Functions**

**Exercise:** Write an SQL query to display the first 5 characters of the customer\_name field from a customers table. Additionally, convert the customer\_name to uppercase.

**Solution:**

SELECT

SUBSTRING(customer\_name, 1, 5) AS short\_name,

UPPER(customer\_name) AS upper\_name

FROM customers;

**2. Numeric Functions**

**Exercise:** Retrieve the square root of the price column from a products table. Round it to 2 decimal places.

**Solution:**

SELECT

product\_name,

ROUND(SQRT(price), 2) AS square\_root\_price

FROM products;

**3. Date Functions**

**Exercise:** Calculate the age of orders based on the order\_date column in the orders table. Assume today’s date is 2025-01-21.

**Solution:**

SELECT

order\_id,

DATEDIFF('2025-01-21', order\_date) AS days\_since\_order

FROM orders;

**4. Aggregate Functions**

**Exercise:** Find the total sales amount (price \* quantity) from the sales table.

**Solution:**

SELECT

SUM(price \* quantity) AS total\_sales

FROM sales;

**5. Generate Groups**

**Exercise:** Group employees by their department and find the average salary for each department from the employees table.

**Solution:**

SELECT

department,

AVG(salary) AS average\_salary

FROM employees

GROUP BY department;

**6. SQL Subqueries**

**Exercise:** Find the names of employees whose salary is above the average salary.

**Solution:**

SELECT

employee\_name

FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees);

**7. Correlated Subqueries**

**Exercise:** Retrieve all customers from the customers table who have placed more orders than the average number of orders per customer.

**Solution:**

SELECT

customer\_id, customer\_name

FROM customers c

WHERE (SELECT COUNT(\*) FROM orders o WHERE o.customer\_id = c.customer\_id) >

(SELECT AVG(order\_count) FROM (SELECT customer\_id, COUNT(\*) AS order\_count FROM orders GROUP BY customer\_id) AS avg\_orders);

**8. Non-Correlated Subqueries**

**Exercise:** Get the product names that belong to a category called "Electronics" using a subquery.

**Solution:**

SELECT

product\_name

FROM products

WHERE category\_id = (SELECT category\_id FROM categories WHERE category\_name = 'Electronics');

**9. Combining Aggregate Functions and Grouping**

**Exercise:** Find the maximum, minimum, and average salary for each job title from the employees table.

**Solution:**

SELECT

job\_title,

MAX(salary) AS max\_salary,

MIN(salary) AS min\_salary,

AVG(salary) AS avg\_salary

FROM employees

GROUP BY job\_title;

**10. Date Functions with Grouping**

**Exercise:** Group orders by the year they were placed and calculate the total sales for each year from the orders table.

**Solution:**

SELECT

YEAR(order\_date) AS order\_year,

SUM(price \* quantity) AS total\_sales

FROM orders

GROUP BY YEAR(order\_date);