Extracting Information & Manipulating Data

Select Statement, Returning only Distinct Rows, Using Aliases, Filtering Results using WHERE Clause, Logical Operations and Operator Precedence, NOT operator, BETWEEN Operator, LIKE Operator, IN Operator, Ordering Results with ORDER BY

Understanding SQL Arithmetic, basic Math operations, ABS() function, POWER() function, SQRT() function, RAND() function, CEILING() function, FLOOR() function, ROUND() function, SUBSTRING() function, Case Conversion Functions, REVERSE() function, TRIM() function, LENGTH() function, SOUNDEX() function, DIFFERENCE() function, DATE() function

Quick Summary -

Select Statement:

Used to retrieve data from one or more database tables
Syntax: SELECT column1, column2, ... FROM table_name

Returning only Distinct Rows:

Used to retrieve only unique/distinct values from a column
Syntax: SELECT DISTINCT column_name FROM table_name

Using Aliases:

Used to rename a table or column in the output of a query
Syntax: SELECT column_name AS alias_name FROM table_name

Filtering Results using WHERE Clause:

Used to retrieve rows that meet a specific condition
Syntax: SELECT column1, column2, ... FROM table_name WHERE condition

Logical Operations and Operator Precedence:

Used to filter results using multiple conditions and operators
Operators: AND, OR, NOT
Precedence: NOT, AND, OR
NOT operator:

Used to retrieve rows that do not match a condition
Syntax: SELECT column1, column2, ... FROM table_name WHERE NOT condition

BETWEEN Operator:

Used to retrieve rows that fall within a specific range
Syntax: SELECT column1, column2, ... FROM table_name WHERE column_name BETWEEN value1 AND

value2

LIKE Operator:

Used to retrieve rows that match a specific pattern
Syntax: SELECT column1, column2, ... FROM table_name WHERE column_name LIKE pattern

IN Operator:

Used to retrieve rows that match one of several values
Syntax: SELECT column1, column2, ... FROM table_name WHERE column_name IN (value1, value2, ...)

Ordering Results with ORDER BY:

Used to sort the output of a query in ascending or descending order
Syntax: SELECT column1, column2, ... FROM table_name ORDER BY column_name ASC/DESC

SQL Arithmetic and Math Functions:

Used to perform mathematical operations on data in the database

Functions: ABS(), POWER(), SQRT(), RAND(), CEILING(), FLOOR(), ROUND(), SUBSTRING(), Case Conversion Functions, REVERSE(), TRIM(), LENGTH(), SOUNDEX(), DIFFERENCE(), DATE()
Syntax: SELECT function(column_name) FROM table_name
Note: All the questions must be answered with the help of examples.

Questions from Each Concepts:

1. **Write a SQL query that retrieves all columns from a table named "employees".**

```
SELECT * FROM employees;
```

2. **Returning only Distinct Rows:**

Write a SQL query that retrieves only distinct values from a column named "product_name" in a table named "products".

```
SELECT DISTINCT product_name FROM products;
```

3. **Using Aliases:**

Write a SQL query that retrieves the "order_id" and "order_date" columns from a table named "orders", but renames "order_id" to "id" and "order_date" to "date".

```
SELECT order_id AS id, order_date AS date FROM orders;
```

4. Filtering Results using WHERE Clause:

Write a SQL query that retrieves all columns from a table named "customers" where the "city" column equals "New York".

```
SELECT * FROM customers WHERE city = 'New York';
```

5. Logical Operations and Operator Precedence:

Write a SQL query that retrieves all columns from a table named "orders" where the "order_date" is after January 1st, 2022 and the "total_price" is greater than $100.

```
SELECT * FROM orders WHERE order_date > '2022-01-01' AND total_price > 100;
```

6. NOT operator:

Write a SQL query that retrieves all columns from a table named "products" where the "product_name" does not contain the word "discount".

```
SELECT * FROM products WHERE product_name NOT LIKE '%discount%';
```

7. BETWEEN Operator:

Write a SQL query that retrieves all columns from a table named "orders" where the "order_date" is between January 1st, 2022 and June 30th, 2022.

```
SELECT * FROM orders WHERE order_date BETWEEN '2022-01-01' AND '2022-06-30';
```

8. LIKE Operator:

Write a SQL query that retrieves all columns from a table named "employees" where the "last_name" starts with the letter "S".

```
SELECT * FROM employees WHERE last_name LIKE 'S%';
```

9. IN Operator:

Write a SQL query that retrieves all columns from a table named "orders" where the "customer_id" is either 1001, 1002, or 1003.

```
SELECT * FROM orders WHERE customer_id IN (1001, 1002, 1003);
```

10. Ordering Results with ORDER BY:

Write a SQL query that retrieves all columns from a table named "products" and orders the results by the "unit_price" column in descending order.

```
SELECT * FROM products ORDER BY unit_price DESC;
```

—----------

SQL Arithmetic:

1. Write a SQL query that calculates the sum of the "quantity" column and the average of the "price" column in a table named "sales".

Solution:

```
SELECT SUM(quantity) as total_quantity, AVG(price) as avg_price FROM sales;
```

Basic Math Operations:

2. Write a SQL query that multiplies the "price" column by 2 in a table named "products".

Solution:

```
SELECT product_name, price * 2 as new_price FROM products;
```

ABS() Function:

3. Write a SQL query that retrieves the absolute value of the "discount" column in a table named "sales".

Solution:

```
SELECT ABS(discount) as absolute_discount FROM sales;
```

POWER() Function:

4. Write a SQL query that calculates the value of 2 raised to the power of 5.

Solution:

```
SELECT POWER(2, 5) as result;
```

SQRT() Function:

5. Write a SQL query that calculates the square root of the "total" column in a table named "orders".

Solution:

```
SELECT SQRT(total) as square_root_total FROM orders;
```

RAND() Function:

6. Write a SQL query that retrieves a random number between 1 and 10.

Solution:

```
SELECT FLOOR(RAND() * 10) + 1 as random_number;
```

CEILING() Function:

7. Write a SQL query that rounds up the value of the "price" column in a table named "products".

Solution:

```
SELECT product_name, CEILING(price) as rounded_price FROM
```

products; **FLOOR() Function:**

8. Write a SQL query that rounds down the value of the "total" column in a table named "orders".

Solution:

```
SELECT FLOOR(total) as rounded_total FROM orders;
```

ROUND() Function:

9. Write a SQL query that rounds the value of the "price" column to two decimal places in a table named "products".

Solution:

```
SELECT product_name, ROUND(price, 2) as rounded_price FROM products;
```

SUBSTRING() Function:

10. Write a SQL query that retrieves the first three characters of the "product_name" column in a table named "products".

Solution:

SELECT SUBSTRING(product_name, 1, 3) as first_three_chars FROM products;

Case Conversion Functions:

11. Write a SQL query that retrieves the "product_name" column in all uppercase letters from a table named "products".

Solution:

SELECT UPPER(product_name) as uppercase_name FROM

products; **REVERSE() Function:**

12. Write a SQL query that retrieves the "product_name" column in reverse order from a table named "products".

Solution:

SELECT REVERSE(product_name) as reversed_name FROM products;

TRIM() Function:

13. Write a SQL query that removes any leading or trailing spaces from the "product_name" column in a table named "products".

Solution:

SELECT TRIM(product_name) as trimmed_name FROM products;
LENGTH() Function:

14. Write a SQL query that retrieves the length of the "product_name" column in a table named "products".

Solution:

SELECT product_name, LENGTH(product_name) as name_length FROM products;

SOUNDEX() Function:

15. Write a SQL query that retrieves the SOUNDEX code for the "last_name" column in a table named "customers".

Solution:

```
SELECT last_name, SOUNDEX(last_name) as soundex_code FROM customers;
```

DIFFERENCE() Function:

16. Write a SQL query that retrieves the difference value between the SOUNDEX codes of the "last_name" column and "first_name" column in a table named "customers".

Solution:

```
SELECT last_name, first_name, DIFFERENCE(last_name, first_name) as soundex_difference FROM customers;
```

Note: The DATE() function in SQL is used to extract the date part from a datetime or timestamp expression. It returns a value in the format 'YYYY-MM-DD'.

```
SELECT DATE('2022-05-03 14:30:00') as date_only;
```

Output:

```
+------------+
| date_only |
+------------+
| 2022-05-03 |
+------------+
```

SCENARIO BASED CHALLENGE QUESTIONS: SELF PROBLEM-SOLVING TASK

Scenario 1:

You are a marketing analyst for a startup company, and you have been tasked with analyzing the company's website traffic data. You have been given a table named "website_traffic" which contains the following columns:

- **date: the date of the website visit**
- **page_visited: the name of the webpage visited**
- **user_id: the unique identifier for the user who visited the webpage**
- **device_type: the type of device used for the website visit (mobile, desktop, tablet)**

Your task is to write SQL queries to answer the following questions:

1. What is the total number of website visits for the past month? 2. How many unique users visited the website in the past month? 3. What was the most visited page on the website in the past month? 4. What percentage of website visits were made using a mobile device in the past month?
5. How many website visits were made by users who visited the website for the first time in the past month?

ANSWERS FOR SCENARIO:

```
CREATE DATABASE mod_4;
USE mod_4;

CREATE TABLE website_traffic (
date DATE,
page_visited VARCHAR(50),
user_id INT,
device_type VARCHAR(10)
);

-- Modified values to be inserted
INSERT INTO website_traffic (date, page_visited, user_id, device_type)
VALUES
('2023-04-01', 'homepage', 1001, 'desktop'),
('2023-04-01', 'products', 1002, 'mobile'),
('2023-04-02', 'about', 1003, 'tablet'),
('2023-04-02', 'homepage', 1004, 'desktop'),
('2023-04-03', 'contact', 1005, 'mobile'),
('2023-04-03', 'homepage', 1006, 'tablet'),
('2023-04-04', 'products', 1007, 'desktop'),
('2023-04-04', 'about', 1008, 'mobile'),
('2023-04-05', 'homepage', 1009, 'tablet'),
('2023-04-05', 'contact', 1010, 'desktop');

-- Query1. What is the total number of website visits for the past month?
SELECT * FROM website_traffic
WHERE date>=now() - INTERVAL 1 MONTH;

-- Query2. How many unique users visited the website in the past month?
SELECT COUNT(DISTINCT user_id) FROM website_traffic
WHERE date>=now() - INTERVAL 1 MONTH;

-- Query3. What was the most visited page on the website in the past month?
SELECT page_visited, COUNT(*) as visits
FROM website_traffic
WHERE date >= now() - INTERVAL 1 MONTH
GROUP BY page_visited
```

```
ORDER BY visits DESC
LIMIT 1;

-- Query4. What percentage of website visits were made using a mobile device in the past
     month?
SELECT COUNT(*) / (SELECT COUNT(*) FROM website_traffic)*100 FROM website_traffic
WHERE device_type='mobile'
AND
date >= now() - INTERVAL 1 MONTH;

-- Query5. How many website visits were made by users who visited the website for the
     first time in the past month?

-- Inserted new values to have duplicate ids

-- INSERT INTO website_traffic (date, page_visited, user_id, device_type)
-- VALUES ('2023-04-05', 'about', 1001, 'mobile'),
-- ('2023-04-05', 'homepage', 1010, 'desktop');

-- Gives Count
SELECT COUNT(DISTINCT user_id) FROM website_traffic
WHERE date>=now() - INTERVAL 1 MONTH;

-- Gives Data
SELECT * FROM website_traffic
WHERE date>=now() - INTERVAL 1 MONTH;
```

—---------- All the Best !! —---------