Sure, SQL (Structured Query Language) uses various operators to perform operations on data in a database. Here are two examples for each of the commonly used SQL operators:

1. Arithmetic Operators:

Example 1: Addition
```sql
SELECT salary + bonus AS total_income
FROM employee;
```

Example 2: Division
```sql
SELECT revenue / number_of_customers AS average_revenue_per_customer
FROM sales_data;
```

2. Comparison Operators:

Example 1: Equal to
```sql
SELECT *
FROM products
WHERE category = 'Electronics';
```

Example 2: Not equal to
```sql
SELECT *
FROM employees
WHERE department_id <> 3;
```

3. Logical Operators:

Example 1: AND
```sql
SELECT *
FROM orders
WHERE order_date >= '2023-01-01'
  AND order_date < '2023-02-01';
```

Example 2: OR

```sql
SELECT *
FROM customers
WHERE country = 'USA'
   OR country = 'Canada';
```

4. LIKE Operator (Pattern Matching):

Example 1: Basic pattern matching
```sql
SELECT *
FROM products
WHERE product_name LIKE 'Laptop%' ;
```

Example 2: Case-insensitive pattern matching
```sql
SELECT *
FROM employees
WHERE last_name ILIKE 'smith%';
```

5. IN Operator:

Example 1: Matching values in a list
```sql
SELECT *
FROM customers
WHERE country IN ('USA', 'Canada', 'Mexico');
```

Example 2: Subquery with IN
```sql
SELECT *
FROM orders
WHERE customer_id IN (SELECT customer_id FROM preferred_customers);
```

6. IS NULL Operator:

Example 1: Finding null values
```sql
SELECT *
```

```
FROM employees
WHERE manager_id IS NULL;
```

Example 2: Finding non-null values
```sql
SELECT *
FROM products
WHERE expiration_date IS NOT NULL;
```

These are just a few examples, and the use of operators can vary based on the specific requirements of your SQL queries.

Certainly! Let's explore a few more SQL operators:

7. BETWEEN Operator:

Example 1: Range of values
```sql
SELECT *
FROM orders
WHERE order_date BETWEEN '2023-01-01' AND '2023-01-31';
```

Example 2: Numeric range
```sql
SELECT *
FROM products
WHERE price BETWEEN 50 AND 100;
```

8. ORDER BY Clause:

Example 1: Sorting in ascending order
```sql
SELECT product_name, price
FROM products
ORDER BY price ASC;
```

Example 2: Sorting in descending order
```sql
```

```sql
SELECT customer_name, total_purchase
FROM customers
ORDER BY total_purchase DESC;
```

9. GROUP BY Clause:

Example 1: Grouping by category
```sql
SELECT category, COUNT(*) AS product_count
FROM products
GROUP BY category;
```

Example 2: Grouping with aggregate function
```sql
SELECT department_id, AVG(salary) AS avg_salary
FROM employees
GROUP BY department_id;
```

10. HAVING Clause:

Example 1: Filtering groups by total sales
```sql
SELECT customer_id, SUM(order_total) AS total_sales
FROM orders
GROUP BY customer_id
HAVING total_sales > 1000;
```

Example 2: Filtering groups by count
```sql
SELECT category, COUNT(*) AS product_count
FROM products
GROUP BY category
HAVING product_count > 5;
```

11. JOIN Operator:

Example 1: INNER JOIN
```sql
SELECT orders.order_id, customers.customer_name
```

```
FROM orders
INNER JOIN customers ON orders.customer_id = customers.customer_id;
```

Example 2: LEFT JOIN
```sql
SELECT employees.employee_id, employees.first_name, departments.department_name
FROM employees
LEFT JOIN departments ON employees.department_id = departments.department_id;
```

These examples cover additional SQL operators and clauses commonly used in queries. The exact usage may vary depending on the specific database system you are working with (e.g., MySQL, PostgreSQL, SQL Server).