Basic SQL Syntax:

1. **SELECT Statement:**
   - The `SELECT` statement is used to retrieve data from one or more tables.

     ```sql
     SELECT column1, column2
     FROM table_name;
     ```

     Example 1:

     ```sql
     SELECT first_name, last_name
     FROM employees;
     ```

     Example 2:

     ```sql
     SELECT product_name, price
     FROM products;
     ```

2. **FROM Clause:**
   - The `FROM` clause specifies the table from which the data should be retrieved.

     ```sql
     SELECT column1, column2
     FROM table_name;
     ```

     Example 1:

     ```sql
     SELECT customer_name, order_date
     FROM orders;
     ```

     Example 2:

     ```sql
     SELECT book_title, author
     FROM books;
     ```

```

3. **WHERE Clause:**
   - The `WHERE` clause is used to filter the rows based on a specified condition.

   ```sql
    SELECT column1, column2
   FROM table_name
   WHERE condition;
   ```

   Example 1:

   ```sql
   SELECT product_name, price
   FROM products
   WHERE category = 'Electronics';
   ```

   Example 2:

   ```sql
   SELECT employee_name, salary
   FROM employees
   WHERE department = 'IT';
   ```

4. **ORDER BY Clause:**
   - The `ORDER BY` clause is used to sort the result set in ascending or descending order.

   ```sql
   SELECT column1, column2
   FROM table_name
   ORDER BY column1 [ASC|DESC];
   ```

   Example 1:

   ```sql
   SELECT product_name, price
   FROM products
   ORDER BY price DESC;
   ```
   ```

Example 2:

```sql
SELECT employee_name, hire_date
FROM employees
ORDER BY hire_date ASC;
```

Certainly! Let's continue with more detailed examples for each component:

5. **SELECT Statement (Aggregate Functions):**
   - The `SELECT` statement can also use aggregate functions like `SUM`, `AVG`, `COUNT`, etc.

```sql
SELECT AVG(column1) AS average_value, COUNT(column2) AS count_value
FROM table_name
WHERE condition;
```

Example 1:

```sql
SELECT AVG(price) AS average_price, COUNT(product_id) AS product_count
FROM products
WHERE category = 'Electronics';
```

Example 2:

```sql
SELECT MAX(salary) AS max_salary, MIN(salary) AS min_salary
FROM employees
WHERE department = 'HR';
```

6. **FROM Clause (Multiple Tables - JOIN):**
   - The `FROM` clause can involve multiple tables using the `JOIN` operation.

```sql
SELECT column1, column2
FROM table1
JOIN table2 ON table1.column_name = table2.column_name;
```

Example 1:

```sql
SELECT orders.order_id, customers.customer_name
FROM orders
JOIN customers ON orders.customer_id = customers.customer_id;
```

Example 2:

```sql
SELECT employees.employee_name, departments.department_name
FROM employees
JOIN departments ON employees.department_id = departments.department_id;
```

7. **WHERE Clause (Logical Operators):**
   - The `WHERE` clause can use logical operators like `AND`, `OR`, and `NOT` for more complex conditions.

```sql
SELECT column1, column2
FROM table_name
WHERE condition1 AND condition2;
```

Example 1:

```sql
SELECT product_name, price
FROM products
WHERE category = 'Electronics' AND price > 500;
```

Example 2:

```sql
SELECT employee_name, salary
FROM employees
WHERE department = 'IT' OR department = 'Engineering';
```

8. **ORDER BY Clause (Multiple Columns):**

- The `ORDER BY` clause can sort the result set based on multiple columns.

  ```sql
  SELECT column1, column2
  FROM table_name
  ORDER BY column1, column2 [ASC|DESC];
  ```

  Example 1:

  ```sql
  SELECT customer_name, order_date, total_amount
  FROM orders
  ORDER BY order_date DESC, total_amount DESC;
  ```

  Example 2:

  ```sql
  SELECT product_name, price, stock_quantity
  FROM products
  ORDER BY stock_quantity ASC, price DESC;
  ```