INNER JOIN:
An INNER JOIN retrieves rows from both tables that satisfy the given condition. If a row in one table does not have a matching row in the other table, that row is excluded from the result set.

Example:
```sql
-- Assume we have two tables: 'employees' and 'departments'
SELECT employees.employee_id, employees.employee_name, departments.department_name
FROM employees
INNER JOIN departments ON employees.department_id = departments.department_id;
```

This SQL query selects employee information (ID and name) along with the corresponding department name using an INNER JOIN on the `department_id` column.

LEFT JOIN:
A LEFT JOIN retrieves all rows from the left table and the matching rows from the right table. If there is no match, NULL values are returned for columns from the right table.

Example:
```sql
-- Using the same tables as before
SELECT employees.employee_id, employees.employee_name, departments.department_name
FROM employees
LEFT JOIN departments ON employees.department_id = departments.department_id;
```

This query retrieves all employees, including those without a corresponding department. If an employee doesn't have a department, the `department_name` column will contain NULL.

RIGHT JOIN:
A RIGHT JOIN is similar to a LEFT JOIN but retrieves all rows from the right table and the matching rows from the left table. If there is no match, NULL values are returned for columns from the left table.

Example:
```sql
-- Using the same tables as before
SELECT employees.employee_id, employees.employee_name, departments.department_name
FROM employees
RIGHT JOIN departments ON employees.department_id = departments.department_id;
```

This query retrieves all departments, including those without any employees. If a department doesn't have any employees, the employee-related columns will contain NULL.

FULL OUTER JOIN:
A FULL OUTER JOIN retrieves all rows when there is a match in either the left or the right table. If there is no match, NULL values are returned for columns from the table without a match.

Example:
```sql
-- Using the same tables as before
SELECT employees.employee_id, employees.employee_name, departments.department_name
FROM employees
FULL OUTER JOIN departments ON employees.department_id = departments.department_id;
```

This query retrieves all employees and departments. If there is no match, the columns from the table without a match will contain NULL values.

Certainly! Let's continue with more detailed explanations and examples for each type of SQL join:

INNER JOIN:
An INNER JOIN returns only the rows where there is a match in both tables based on the specified condition in the ON clause.

Example 1:
```sql
-- Assuming 'orders' and 'customers' are two tables
SELECT orders.order_id, orders.order_date, customers.customer_name
FROM orders
INNER JOIN customers ON orders.customer_id = customers.customer_id;
```

This query retrieves information about orders (order ID and order date) along with the corresponding customer name using an INNER JOIN on the `customer_id` column.

Example 2:
```sql
-- Another example with different tables: 'products' and 'categories'
SELECT products.product_name, products.price, categories.category_name
FROM products
INNER JOIN categories ON products.category_id = categories.category_id;
```

This query fetches product details (product name and price) along with the corresponding category name using an INNER JOIN on the `category_id` column.

LEFT JOIN:
A LEFT JOIN returns all rows from the left table and the matching rows from the right table. If there is no match in the right table, NULL values are returned for columns from the right table.

Example 1:
```sql
-- Using 'students' and 'grades' tables
SELECT students.student_id, students.student_name, grades.grade_value
FROM students
LEFT JOIN grades ON students.student_id = grades.student_id;
```

This query retrieves information about students (student ID and student name) along with their corresponding grades using a LEFT JOIN on the `student_id` column.

Example 2:
```sql
-- Another example with different tables: 'employees' and 'projects'
SELECT employees.employee_id, employees.employee_name, projects.project_name
FROM employees
LEFT JOIN projects ON employees.employee_id = projects.employee_id;
```

This query fetches employee details (employee ID and employee name) along with the projects they are associated with using a LEFT JOIN on the `employee_id` column.

RIGHT JOIN:
A RIGHT JOIN returns all rows from the right table and the matching rows from the left table. If there is no match in the left table, NULL values are returned for columns from the left table.

Example 1:
```sql
-- Using 'customers' and 'orders' tables
SELECT customers.customer_id, customers.customer_name, orders.order_date
FROM customers
RIGHT JOIN orders ON customers.customer_id = orders.customer_id;
```

This query retrieves information about customers (customer ID and customer name) along with the dates of their orders using a RIGHT JOIN on the `customer_id` column.

Example 2:
```sql
-- Another example with different tables: 'departments' and 'employees'
SELECT departments.department_id, departments.department_name,
employees.employee_name
FROM departments
RIGHT JOIN employees ON departments.department_id = employees.department_id;
```

This query fetches department details (department ID and department name) along with the employees in those departments using a RIGHT JOIN on the `department_id` column.

FULL OUTER JOIN:
A FULL OUTER JOIN returns all rows when there is a match in either the left or the right table. If there is no match, NULL values are returned for columns from the table without a match.

Example 1:
```sql
-- Using 'students' and 'courses' tables
SELECT students.student_id, students.student_name, courses.course_name
FROM students
FULL OUTER JOIN courses ON students.student_id = courses.student_id;
```

This query retrieves information about students (student ID and student name) along with the courses they are enrolled in using a FULL OUTER JOIN on the `student_id` column.

Example 2:
```sql
-- Another example with different tables: 'employees' and 'skills'
SELECT employees.employee_id, employees.employee_name, skills.skill_name
FROM employees
FULL OUTER JOIN skills ON employees.employee_id = skills.employee_id;
```

This query fetches employee details (employee ID and employee name) along with the skills they possess using a FULL OUTER JOIN on the `employee_id` column.

These examples should provide a comprehensive understanding of how INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL OUTER JOIN work in SQL. Adjust the tables and columns according to your specific database schema and requirements.