In SQL, **joins** are used to combine rows from two or more tables based on a related column. Joins can be broadly classified into **equi-joins** and **non-equi-joins** based on the condition used in the join clause.

**Equi-Join**

An **equi-join** is a type of join where the condition used in the join clause is based solely on equality (=) between two columns.

**Key Characteristics:**

1. Combines rows from two or more tables where the values in the specified columns are equal.
2. Typically uses the ON or USING keyword in SQL.
3. Results in all matching rows from both tables based on the equality condition.
4. Can be used with different types of joins (e.g., INNER JOIN, LEFT JOIN, etc.).

**Example:**

Consider two tables: employees and departments.

**employees**

| **emp\_id** | **emp\_name** | **dept\_id** |
| --- | --- | --- |
| 1 | Alice | 101 |
| 2 | Bob | 102 |
| 3 | Charlie | 101 |

**departments**

| **dept\_id** | **dept\_name** |
| --- | --- |
| 101 | HR |
| 102 | Engineering |
| 103 | Finance |

**Query:**

SELECT employees.emp\_name, departments.dept\_name

FROM employees

JOIN departments

ON employees.dept\_id = departments.dept\_id;

**Result:**

| **emp\_name** | **dept\_name** |
| --- | --- |
| Alice | HR |
| Bob | Engineering |
| Charlie | HR |

**Non-Equi Join**

A **non-equi join** is a type of join where the join condition is based on a comparison operator other than equality, such as >, <, >=, <=, or <>.

**Key Characteristics:**

1. Combines rows based on a non-equality condition.
2. Commonly used when comparing ranges or performing complex joins.
3. Requires an explicit ON clause specifying the non-equal condition.

**Example:**

Consider two tables: sales and targets.

**sales**

| **sale\_id** | **amount** |
| --- | --- |
| 1 | 500 |
| 2 | 1200 |
| 3 | 800 |

**targets**

| **target\_id** | **min\_amount** | **max\_amount** |
| --- | --- | --- |
| 1 | 0 | 499 |
| 2 | 500 | 999 |
| 3 | 1000 | 1499 |

**Query:**

SELECT sales.sale\_id, targets.target\_id

FROM sales

JOIN targets

ON sales.amount BETWEEN targets.min\_amount AND targets.max\_amount;

**Result:**

| **sale\_id** | **target\_id** |
| --- | --- |
| 1 | 2 |
| 2 | 3 |
| 3 | 2 |

**Key Differences Between Equi and Non-Equi Joins**

| **Feature** | **Equi-Join** | **Non-Equi Join** |
| --- | --- | --- |
| Join Condition | Uses equality (=) operator. | Uses other comparison operators. |
| Typical Use Case | Matching exact values in columns. | Comparing ranges or complex logic. |
| Complexity | Relatively straightforward. | More complex conditions. |

Both types of joins are useful depending on the requirements of the query.