SQL Aliases

An **alias** in SQL is a temporary name given to a table or column for the purpose of a specific query. Aliases make queries easier to read and write, especially when working with complex queries, multiple tables, or when you need to rename columns for clarity in the result set.

Types of Aliases

1. **Column Aliases**
   A column alias renames a column in the result set of a query. It is useful when you want to display a more meaningful or readable name for a column.
2. **Table Aliases**
   A table alias gives a table a temporary name for the duration of a query. It is especially useful in `JOIN` operations or when working with multiple instances of the same table.

Syntax of Aliases

- **Column Alias Syntax**

```sql
SELECT column_name AS alias_name FROM table_name;
```

`AS` is optional but often used for readability. You can also omit `AS`:

```sql
SELECT column_name alias_name FROM table_name;
```

- **Table Alias Syntax**

```sql
SELECT column_name FROM table_name AS alias_name;
```

Like with column aliases, `AS` is optional for table aliases.

1. Column Alias Example

Renaming the column `Salary` to `EmployeeSalary` for clarity in the result set:

```sql
SELECT Salary AS EmployeeSalary
FROM Employees;
```

2. Table Alias Example

Renaming the table `Employees` to `E` for use in a `JOIN` operation:

```sql
SELECT E.Name, E.Salary
FROM Employees AS E;
```

(Here, `E` is used as a shortcut for the `Employees` table in the query.)

3. Using Aliases in Joins

Using table aliases to simplify a query with a `JOIN`:

```sql
SELECT E.Name, D.DepartmentName
FROM Employees AS E
INNER JOIN Departments AS D
ON E.DepartmentID = D.DepartmentID;
```

(Here, `E` is the alias for `Employees`, and `D` is the alias for `Departments`.)

4. Column Alias in Aggregation

Renaming the result of an aggregate function:

```sql
SELECT AVG(Salary) AS AverageSalary
FROM Employees;
```

(Here, `AVG(Salary)` is given the alias `AverageSalary`.)

Key Points About Aliases

- Aliases are temporary and only exist for the duration of the query.
- **Column aliases** help improve clarity in the result set, especially for computed values or functions.
- **Table aliases** are most useful when dealing with multiple tables, particularly in `JOIN` operations, to simplify the query and avoid long table names.
- Aliases can be used with `GROUP BY`, `ORDER BY`, and `HAVING` clauses.

Limitations

- **No Direct Modifications**: You cannot use an alias in the `WHERE` clause, since the alias is created after the `WHERE` clause is processed.

Example (invalid):

sql Copy Edit

```sql
SELECT Salary AS EmployeeSalary
FROM Employees
WHERE EmployeeSalary > 50000;  -- Invalid, alias cannot be used in WHERE
```

Instead, you would use the original column name:

sql Copy Edit

```sql
SELECT Salary AS EmployeeSalary
FROM Employees
WHERE Salary > 50000;  -- Valid
```