SQL Clause Execution Order

In PostgreSQL (and SQL in general), the **clause execution order** for a query is determined by the logical execution order of operations. The order might differ from the way the query is written syntactically. Here's the logical order of execution for SQL clauses:

Logical Execution Order of SQL Clauses:

1. **FROM**
   The table(s) or subquery specified is identified, and joins are performed if necessary.
2. **WHERE**
   Filters rows based on the condition provided in the clause.
3. **GROUP BY**
   Groups rows that have the same values in specified columns into aggregated rows.
4. **HAVING**
   Filters groups based on a condition applied to the aggregated data.
5. **SELECT**
   Evaluates the columns or expressions specified for output. Aliases defined here are applied at this stage.
6. **DISTINCT**
   Removes duplicate rows from the result set.
7. **ORDER BY**
   Sorts the result set according to the specified columns or expressions.
8. **LIMIT/FETCH**
   Limits the number of rows returned in the final result set.

Example Query with Logical Execution Order:

sql

```sql
SELECT DISTINCT department_id, AVG(salary) AS avg_salary
FROM employees
WHERE job_id = 'IT_PROG'
GROUP BY department_id
HAVING AVG(salary) > 5000
ORDER BY avg_salary DESC
LIMIT 5;
```

Logical Steps:

1. **FROM**: Identify the `` `employees` `` table.

2. **WHERE**: Filter rows where `job_id = 'IT_PROG'`.
3. **GROUP BY**: Group rows by `department_id`.
4. **HAVING**: Filter groups where `AVG(salary) > 5000`.
5. **SELECT**: Compute `department_id` and `AVG(salary)` for output.
6. **DISTINCT**: Remove duplicate rows (if any).
7. **ORDER BY**: Sort the result by `avg_salary` in descending order.
8. **LIMIT**: Return only the first 5 rows.

Key Points:

- `WHERE` works on rows before aggregation; `HAVING` works on aggregated groups.
- Aliases defined in the `SELECT` clause cannot be used in `WHERE`, but they can be used in `ORDER BY`.
- Always think in terms of the logical order, even if the SQL syntax looks different.

Would you like to explore an example for a better understanding? 😊