Order of Execution in SQL

An SQL query comprises various clauses like SELECT, FROM, WHERE, GROUPBY, HAVING, and ORDERBY clauses. Each clause has a specific role in the query. Let's understand each of them briefly.

When you write any query, your query is processed in the following steps:

- Getting Data (FROM/JOIN)
- Row Filter (WHERE)
- Grouping (GROUP BY)
- Group Filter (HAVING)
- Return Expression (SELECT)
- Order & Paging (ORDER BY & LIMIT/OFFSET)

Clause	Function
FROM / JOIN	When you write any query, SQL starts by identifying the tables for the data retrieval and how they are connected.
WHERE	It acts as a filter; it filters the record based on the conditions specified by the users.
GROUP BY	The filtered data is grouped based on the specified condition.
HAVING	It is similar to the WHERE clause but applied after grouping the data.
SELECT	The clause selects the columns to be included in the final result.
DISTINCT	Remove the duplicate rows from the result. Once you apply this clause, you are only left with distinct records.

ORDER BY	It sorts the results (increasing/decreasing/A->Z/Z->A) based on the specified condition.
LIMIT / OFFSET	It determines the number of records to return and from where to start.

You have clearly understood the theoretical aspect of the order of execution in SQL until now. Let's take an example to better understand the concept.

Let's consider a simple dataset with two tables: Customers and Orders.

- The Customers table has 5 columns: customer_id, first_name, last_name, age, and country.
- Orders Table has 4 columns: order_id, item, amount, customer_id

Customers Table

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Orders Table

order_id	item	amount	customer_id

1	Keyboard	400	4
2	Mouse	300	4
3	Monitor	12000	3
4	Keyboard	400	1
5	Mousepad	250	2

Problem Statement: Find the amount spent by each customer in the USA.

Explain

```
SELECT Customers.first_name, Customers.last_name, SUM(Orders.Amount) as Amount
FROM Customers
JOIN Orders ON Customers.customer_id = Orders.customer_id
WHERE Customers.country = 'USA'
GROUP BY Customers.first_name, Customers.last_name
ORDER BY Amount DESC;
```

Copy code

Output

first_name	last_name	Amount
John	Doe	400
Robert	Luna	250

Explanation

- FROM and JOIN: We start by identifying the ‘Customers‘ and ‘Orders‘ tables and joining them on ‘customer_id‘.
- WHERE: It will filter the record to include only those where ‘country‘ = ‘USA‘.

- GROUP BY: Group the remaining entries (after filtering by WHERE clause) by ‘first_name‘ and ‘last_name‘.
- SELECT: SELECT the ‘first_name‘, ‘last_name‘, and the sum of ‘Amount‘ for each group.
- ORDER BY: Finally, the result is sorted by ‘Amount‘ in descending order.

Now, let’s take an example and reshuffle the order of execution in sql.

Case-1: Let you want to filter the record based on the ‘Amount’ using the WHERE clause.

Explain

```
SELECT Customers.first_name, Customers.last_name, SUM(Orders.Amount) as Amount
FROM Customers
JOIN Orders ON Customers.customer_id = Orders.customer_id
WHERE Orders.Amount >300
GROUP BY Customers.first_name, Customers.last_name
ORDER BY Amount DESC;
```

Copy code

Output

first_name	last_name	Amount
David	Robinson	12000
John	Doe	400
John	Reinhardt	400

Case-2: Filter the record based on the ‘Amount’ using the HAVING clause.

Explain

```
SELECT Customers.first_name, Customers.last_name, SUM(Orders.Amount) as Amount
```

```
FROM Customers
JOIN Orders ON Customers.customer_id = Orders.customer_id
GROUP BY Customers.first_name, Customers.last_name
HAVING Amount > 300
ORDER BY Amount DESC;
```

Copy code

Output

first_name	last_name	Amount
David	Robinson	12000
John	Reinhardt	700
John	Doe	400

Now, let's see what happened in both cases:

Since the WHERE clause is processed before the SELECT clause in the Order of Execution, so, in the first case, SQL won't recognize the Amount and will give the error. It just filters out the record of the customer who purchased orders greater than 300.

However, the best way to filter the aggregate function is to use the HAVING clause. Since the HAVING clause is processed after the GROUP BY clause. So, in the second case, the HAVING clause filters the group to include only those where the total Amount is greater than 300.