Order of Execution in SQL

Complex or even sometimes simples SQL queries will make you scratch your head if you are not aware of or not able to connect with order of execution of sql statements.

Hierarchy of statements execution

Getting data (From, Join)

↓

Row Filter (where)

↓

Grouping (Group by)

↓

Group filter (Having)

↓

Return Expressions (Select)

↓

Order & Paging (order by / limit / offset)

If all these are used in a single query then order of execution of statement will be from top to bottom.

Now we'll take one query to understand the execution and few other cases that will give you an idea of why your query is not working.

DATE
PAGE

CITIZEN

Name	City_id
Rahul	3
Raj	2
Jemy	1
Sahil	3
Ram	1
Ankita	2
Suraj	3
Ramesh	1

CITY

City_id	City_name
1	Pune
2	Delhi
3	Bengaluru

We are provided with citizen and city table with city_id in common between two.

Let's say you are given with a task to show the city names where more than two citizens are living except they city Delhi and have to order your result alphabetically.

So query will looks something like

```
Select city.city_name as "City"
from citizen
inner join CITY
on citizen.city_id = city.city_id
where city.city_name != 'Delhi'
Groupby city.city_name
having count (*) >=2
order by city.city_name ASC
Limit 2.
```

DATE
PAGE

1) First step

getting data (From/Join)

Firstly it will check the from & join column and will form the base table

citizen

Name	City_id
Rahul	3
Raj	2
Jemy	1
Sahil	3
Ram	1
Ankita	2
Suraj	3
Ramesh	1

city

city_id	city_name
3	Bengaluru
2	Delhi
1	Pune
3	Bengaluru
1	Pune
2	Delhi
3	Bengaluru
1	Pune

↓

from citizen inner join city
on citizen.city_id = city.city_id

2) Second step

Row filtering (where)

Name	city_id	city_id	city_name
Rahul	3	3	Bangalore
Jemy	1	1	Pune
Sahil	3	3	Bangalore
Ram	1	1	Pune
Suraj	3	3	Bangalore
Ramesh	1	1	Pune

↓

where city.city_name != 'Delhi'

Now here you should be thinking what if ?

```
Select city.city_name as city
from citizen
inner join CITY
where city.city_name != 'Delhi'
on city.city_id = citizen.city_id
```

If you are not thinking about it. There could be two cases

Case 1 : You understood order of execution

Case 2 : You are not engaged

```
If case1 == True:
    print("Congrats. You got this")
Else
    print("Please be engaged")
```

See, its again related to order of execution whenever sql sees inner join or from statement it will prioritize them and execute it first

And if inner join has been already performed there is no sense of this where clause

3) Third Step

Grouping (Group by)

Name	City_id	city_id	City_name
Rahul	3	3	
Sahil	3	3	Bangalore
Suraj	3	3	
Jemy	1	1	
Ram	1	1	Pune
Ramesh	1	1	

Group by city.city_name.

4) Fourth step.

Group filter (having)

	Name	City_id	City_id	City_name
Count=3 (i.e >2)	Rahul	3	3	
	Sahil	3	3	Bangalore
	Suraj	3	3	
Count=3 (i.e >2)	Jemy	1	1	
	Ram	1	1	Pune
	Ramesh	1	1	

Having count(*) >= 2 (as there is no group with less than two rows no filtering will be done)

5) Fifth Step

Return Expression (select)

Now the table built till step 4 from that table we need to extract city_name and will assign a new name city to the column.

Note:- This alias will change the name in your output table/view. No change in original table.

City
Bangalore
Pune

Select city.city_name as "City"

6) Order (order by) and pagging (limit)

In this final step the output obtained by step 5 will be ordered and limited as per your requirements.

City
Bangalore
Pune

Order by city.city_name Asc
Limit 2

As here total no. of rows are two only & data is already sorted in Alphabetical order no change has been made in step 6.

DATE
PAGE

So, this is how a complete query executes at the back end.

Hope this has given you a broader idea on how SQL query works and you'll be ready to rock in SQL next time.

ALERT! Additional Gyan (knowledge) ahead!

I have seen a lot more people getting confused on aliases a lot.

Like why my aliases are not working when I have defined them.

Understand this although you are writing alias in the first line of your query these aliases are getting created in Step 5 (Return expression) so the steps above step 5 doesn't have any idea of the existence of these aliases. So your aliases name would be available for order & paging step (step 6) only.

Note!. MySQL and Postgrese got the smart enough and in these you have an facility to use alias name in Group by & Having clause also but for SQL server & Oracle its still the same and alias name will be available for order by only.

So if you are using alias name in where or group by clauses & its throwing an error this might be the reason.

```
1) SELECT CITY.CITY_NAME AS "NAMEOFCITY"
2) FROM CITIZEN
3) INNER JOIN CITY
4) ON CITY.CITY_ID = CITIZEN.CITY_ID
5) WHERE CITY.CITY_NAME != "DELHI"
6) GROUP BY NAMEOFCITY
7) HAVING COUNT(*) >= 2
8) ORDER BY NAMEOFCITY ASC
9) LIMIT 2;
```

This query will work fine for MySQL & Postgre but will through an error for Oracle & SQLserver because of line 6.

#Happy Learning :)