### AGGREGATE FUNCTIONs AND JOINs

**Aggregating with Grouping**

**GROUP BY Clause**

There are some cases where we may need to apply the aggregate function not only to a selected group of rows. In the previous parts, we found the total amount of salaries of male and female employees separately.  What if want to show them in the same result table grouped by their gender. In such cases, we use GROUP BY statement.

The GROUP BY clause groups the rows into summary rows. It returns one value for each group and is typically used with aggregate functions (COUNT, MAX, MIN, SUM, AVG).

The syntax is:

SELECT column\_1, aggregate\_function(column\_2)

FROM table\_name

GROUP BY column\_1;

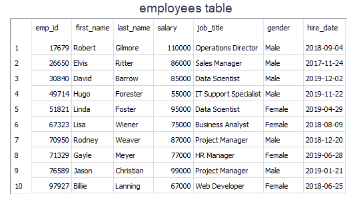
In case a WHERE clause is used, the GROUP BY clause should come after the WHERE clause. We specify the column or a list of comma-separated columns right after the GROUP BY.  Any nonaggregate expression in the SELECT must be included in the GROUP BY clause.

We've mentioned that the GROUP BY clause is often used with aggregate functions. Now, let's see how it is.

**💡Tips:**

* GROUP BY returns only one result per group of data.
* GROUP BY Clause always follows the WHERE Clause.
* GROUP BY Clause always precedes the ORDER BY.

**GROUP BY with COUNT Function**



The following statement returns the number of employees per gender.

query:

SELECT gender, COUNT(gender)

FROM employees

GROUP BY gender;

output:

gender COUNT(gender)

---------- -------------

Female 4

Male 6

**☝ Important:** The GROUP BY clause groups results before calling the aggregate function. This allows you to apply aggregate function to groups than the entire query.

Now, let's write another query. What is the number of employees working as a data scientist? Break down it by gender.  
  
query:

SELECT gender, COUNT(job\_title)

FROM employees

WHERE job\_title = 'Data Scientist'

GROUP BY gender;

output:

gender COUNT(job\_title)

---------- ----------------

Female 1

Male 1

**💡Tips:**

* WHERE clause operates on the data before the aggregation.
* WHERE clause happens before the GROUP BY clause.
* Only the rows that meet the conditions in the WHERE clause are grouped.

### GROUP BY with MIN&MAX Functions

Let's find the minimum salaries of each gender group using the MIN function.  
  
query:

SELECT gender, MIN(salary) AS min\_salary

FROM employees

GROUP BY gender;

output:

gender min\_salary

---------- ----------

Female 67000

Male 55000

Similarly, we can find the maximum salaries of each group using the MAX function.  
  
query:

SELECT gender, MAX(salary) AS max\_salary

FROM employees

GROUP BY gender;

output:

gender max\_salary

---------- ----------

Female 95000

Male 110000

You may also use the ORDER BY clause to sort the salaries in descending or ascending order.  The ORDER BY follows GROUP BY. For instance, sort the maximum salaries for each group in descending order.  
  
query:

SELECT gender,

MAX(salary) AS max\_salary

FROM employees

GROUP BY gender

ORDER BY max\_salary DESC;

output:

gender max\_salary

---------- ----------

Male 110000

Female 95000

**💡 Tip:** When sorting, you can use the alias with ORDER BY clause as we did in the query above (ORDER BY **max\_salary**). You may also use the original aggregate function as ORDER BY **MAX(salary**)

### GROUP BY with SUM&AVG Functions

In the previous part, we have learned how to use MIN and MAX functions with the GROUP BY clause. In this part, we will continue with SUM and AVG functions. Let's go with the same scenario. In this case, we will calculate the total salaries of each group (gender).

query:

SELECT gender, SUM(salary) AS total\_salary

FROM employees

GROUP BY gender;

output:

gender total\_salary

---------- ------------

Female 314000

Male 522000

Similarly, we can find the average salaries of each group using the AVG function.  
  
query:

SELECT gender, AVG(salary) AS average\_salary

FROM employees

GROUP BY gender;

output:

gender average\_salary

---------- --------------

Female 78500.0

Male 87000.0

You may also use the ORDER BY clause to sort the salaries in descending or ascending order. For instance, sort the total salaries for each group in descending order.  
  
query:

SELECT gender,

SUM(salary) AS total\_salary

FROM employees

GROUP BY gender

ORDER BY total\_salary DESC;

output:

---------- ------------

Male 522000

Female 314000