GROUPBY

GROUP BY is a SQL clause used to group rows based on one or more columns in a table. The GROUP BY clause is used in conjunction with aggregate functions, such as SUM(), COUNT(), AVG(), MIN(), and MAX(), to perform calculations on the grouped rows.

Here is an example using the GROUP BY clause:

```
SELECT country, COUNT(*) FROM customers GROUP BY country;
```

This will group the customers table by the country column and return the count of customers for each country.

The GROUP BY clause creates a distinct group for each unique value in the specified column or columns, and then the aggregate function is applied to each group to calculate the summary statistic.

Here is another example using the GROUP BY clause with the AVG() function:

```
SELECT city, AVG(salary) FROM employees GROUP BY city;
```

This will group the employees table by the city column and return the average salary for each city.

The GROUP BY clause is useful when you want to summarize data by groups and perform calculations on those groups. It is commonly used in business intelligence and data analysis applications to summarize large amounts of data and make it more manageable for analysis.

HAVING

HAVING is a SQL clause used to filter the results of a GROUP BY query based on a condition involving an aggregate function. The HAVING clause is used in conjunction with the GROUP BY clause and follows it in the query.

Here is an example using the HAVING clause:

```
SELECT country, COUNT(*) FROM customers GROUP BY country HAVING COUNT(*) >
10;
```

This will group the customers table by the country column and return the count of customers for each country, but only for those countries where the count is greater than 10.

The HAVING clause allows you to filter the groups created by the GROUP BY clause based on the result of an aggregate function, such as COUNT(), SUM(), AVG(), MIN(), or MAX(). The HAVING clause is similar to the WHERE clause, but it operates on the groups created by the GROUP BY clause, rather than on individual rows.

Here is another example using the HAVING clause with the AVG() function:

```
SELECT city, AVG(salary) FROM employees GROUP BY city HAVING AVG(salary) > 50000;
```

This will group the employees table by the city column and return the average salary for each city, but only for those cities where the average salary is greater than 50,000.

The HAVING clause is useful when you want to filter the results of a GROUP BY query based on a summary statistic. It is commonly used in business intelligence and data analysis applications to focus on specific groups that meet certain criteria.

ORDER BY

ORDER BY is a SQL clause used to sort the results of a query by one or more columns in ascending or descending order. The ORDER BY clause is placed at the end of a SELECT statement and follows any WHERE, GROUP BY, and HAVING clauses.

Here is an example using the ORDER BY clause:

```
SELECT name, age FROM customers ORDER BY age DESC;
```

This will select the name and age columns from the customers table and sort the results by age in descending order.

The ORDER BY clause can sort the results of a query by one or more columns in ascending or descending order. By default, the ORDER BY clause sorts the results in ascending order, but you can specify the DESC keyword to sort in descending order.

Here is another example using the ORDER BY clause with two columns:

```
SELECT first_name, last_name, salary FROM employees ORDER BY last_name ASC, salary DESC;
```

This will select the first_name, last_name, and salary columns from the employees table and sort the results by last_name in ascending order, and then by salary in descending order.

The ORDER BY clause is useful when you want to sort the results of a query in a specific order. It is commonly used in business intelligence and data analysis applications to sort data in ascending or descending order based on a specific column or set of columns.