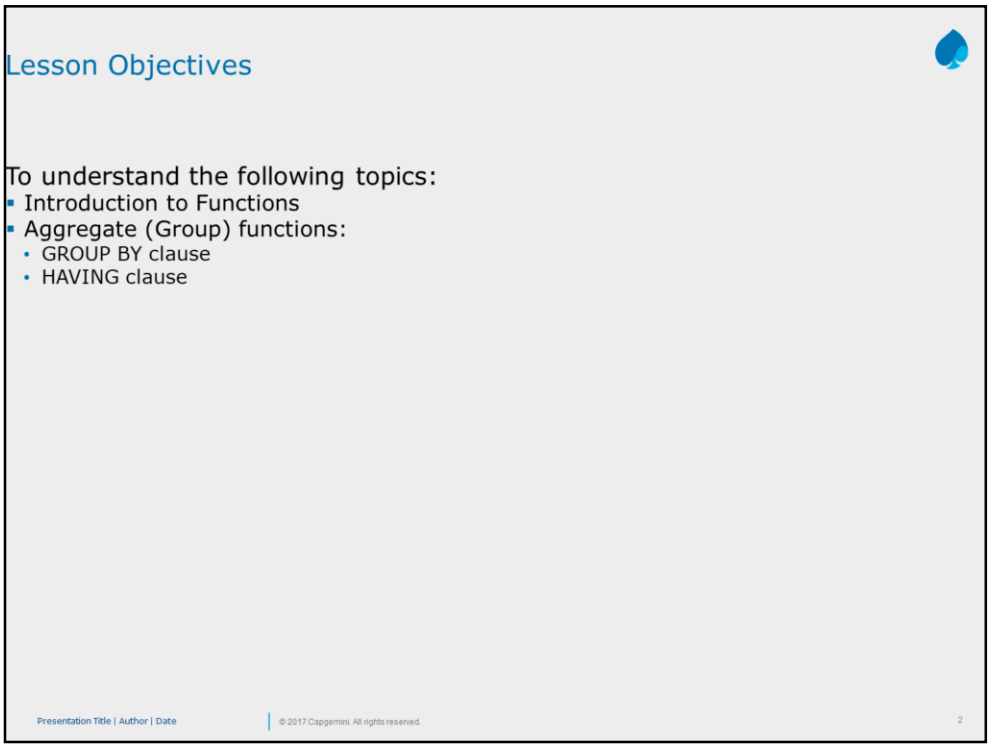
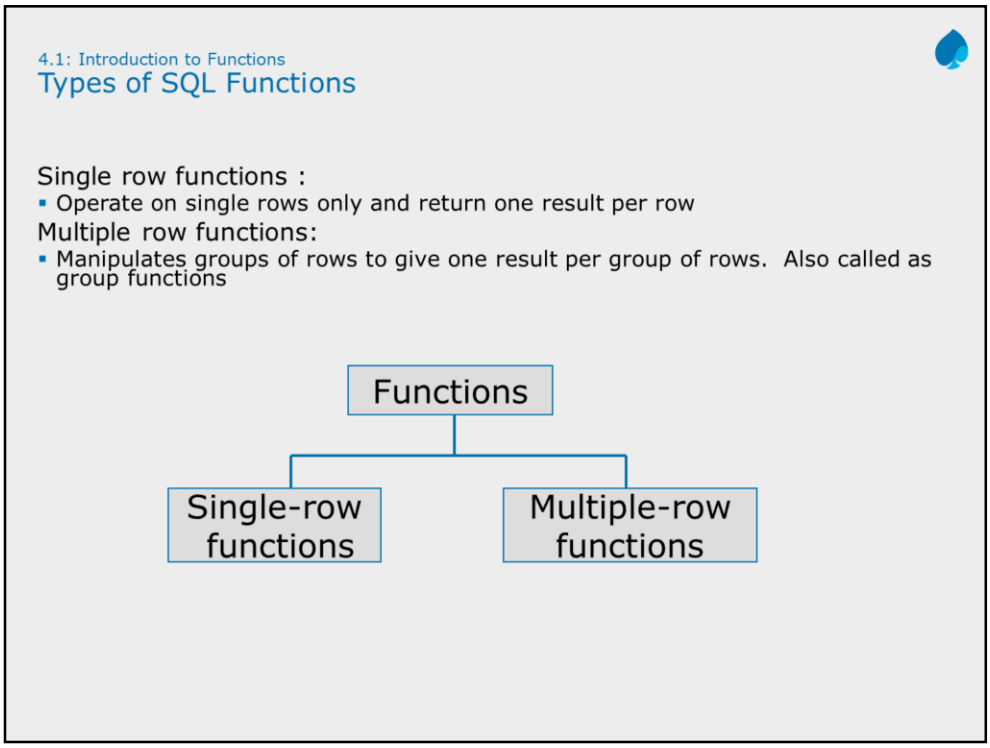
Instructor Notes:

Instructor Notes:

None



Instructor Notes:

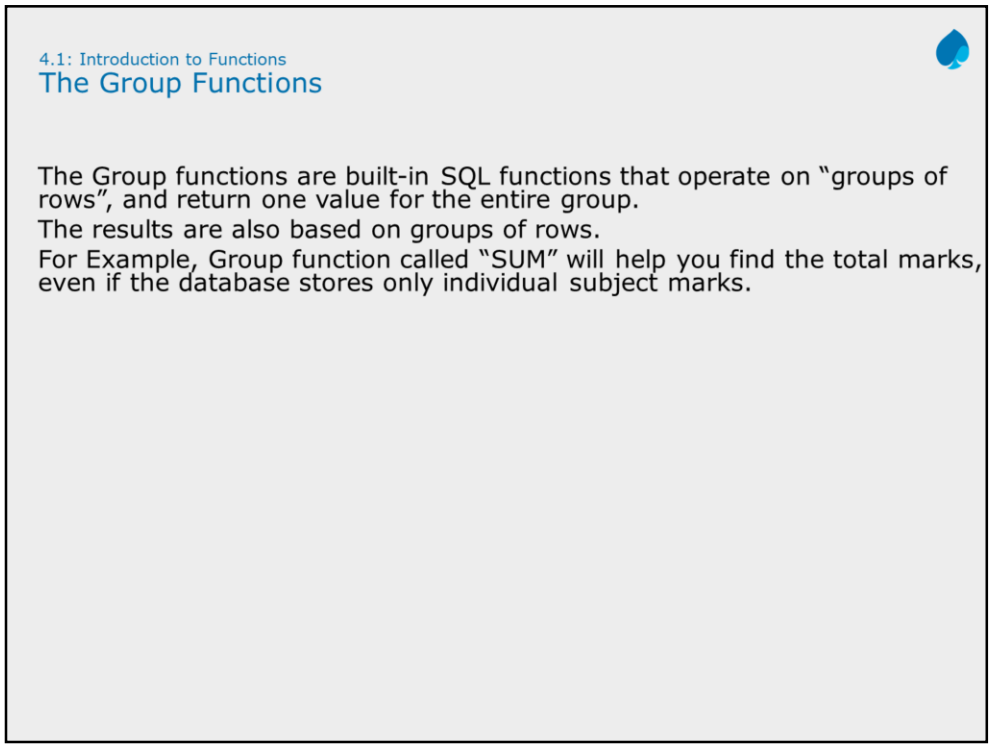


Functions:

- Functions can be used to manipulate data values in a variety of ways.
  - Functions help in making the basic query block more powerful.
  - Functions may be used to perform calculations on data, alter data formats for display, convert data types, etc.
- Functions are similar to operators in that they manipulate data items and return a result.
- Functions differ from operators in the format in which they appear with their arguments. This format allows them to operate on zero, one, two, or more arguments:
  - function(argument, argument, ...)
- There are two types of functions: SQL functions and User-defined functions.
  1. **SQL functions**: SQL functions are built into most DBMS and are available for use in various appropriate SQL statements. SQL functions are further categorized as:
     a) **Single-Row functions**: Single-row functions return a single result row for every row of a queried Table or View.
     For example: ABS, ROUND etc.

Instructor Notes:

None

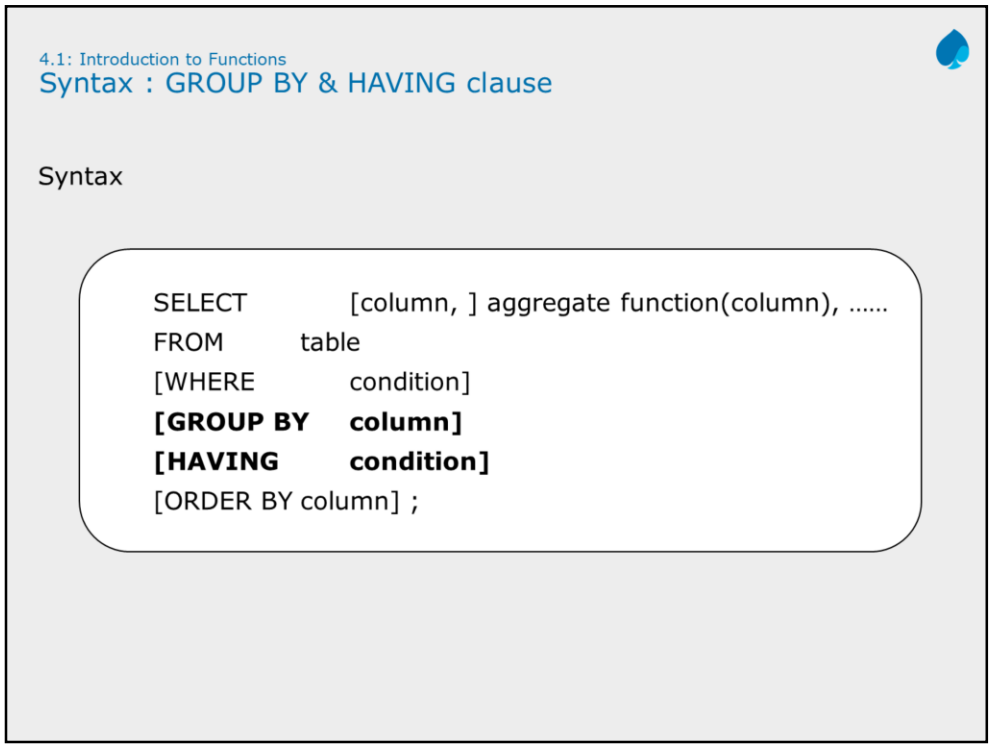


Aggregate (Group) Functions:

- SQL provides a set of "built-in" functions for producing a single value for an entire group. These functions are called as "Set functions" or "Aggregate (Group) functions".
- These functions can work on a "normal result table" or a "grouped result table".
  - If the result is not grouped, then the aggregate will be taken for the whole result table.

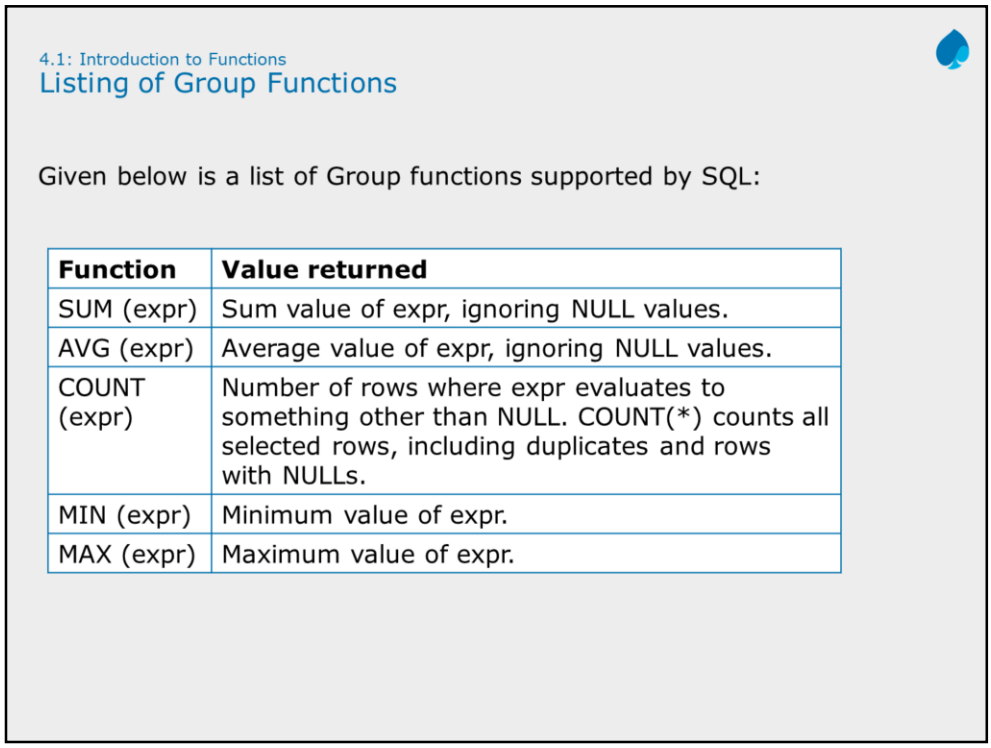
Instructor Notes:

None



Instructor Notes:

None



4.1: Introduction to Functions

Listing of Group Functions

Given below is a list of Group functions supported by SQL:

Function	Value returned
SUM (expr)	Sum value of expr, ignoring NULL values.
AVG (expr)	Average value of expr, ignoring NULL values.
COUNT (expr)	Number of rows where expr evaluates to something other than NULL. COUNT(*) counts all selected rows, including duplicates and rows with NULLs.
MIN (expr)	Minimum value of expr.
MAX (expr)	Maximum value of expr.

Aggregate (Group) Functions supported by SQL:

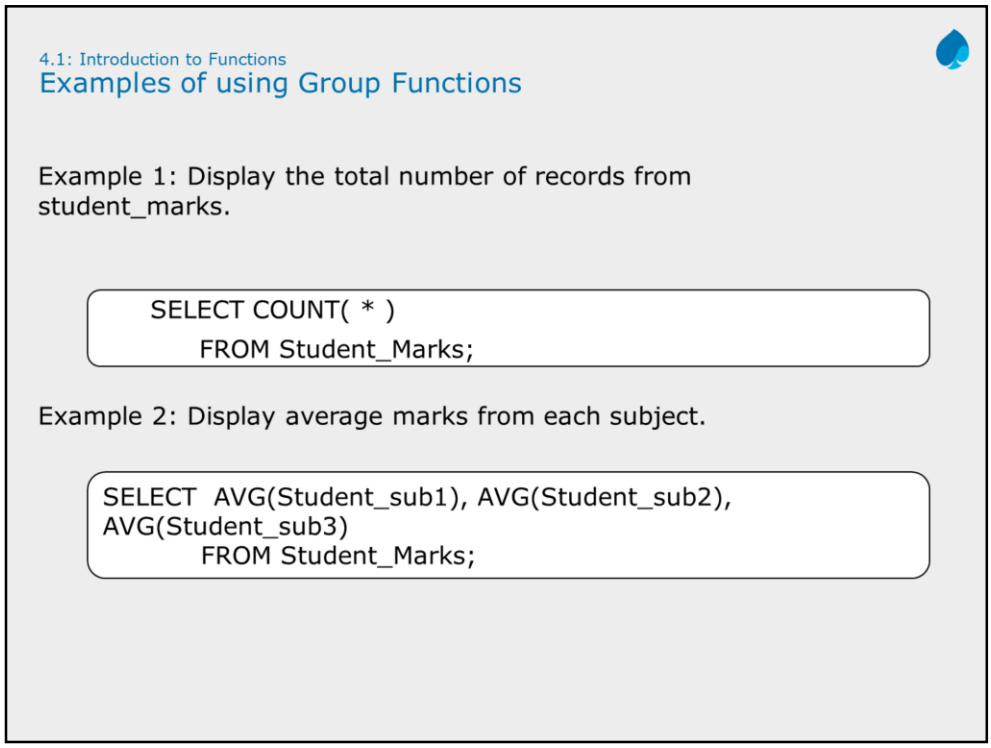
- All the above functions operate on a number of rows (for example, an entire table), and are therefore known as “Group (or Aggregate) functions”.
- A Group function can be used on a subset of the rows in a table by using the WHERE clause.
- The Aggregate functions ignore NULL values in the column.
  - To include NULL values, NVL function can be used with Aggregate functions.

Note :

- **Count(*)**: Returns the number of rows in the table, including duplicates and those with NULLs.
- **Count(<Expression>)**: Returns the number of rows where expression is NOT NULL.

Instructor Notes:

None

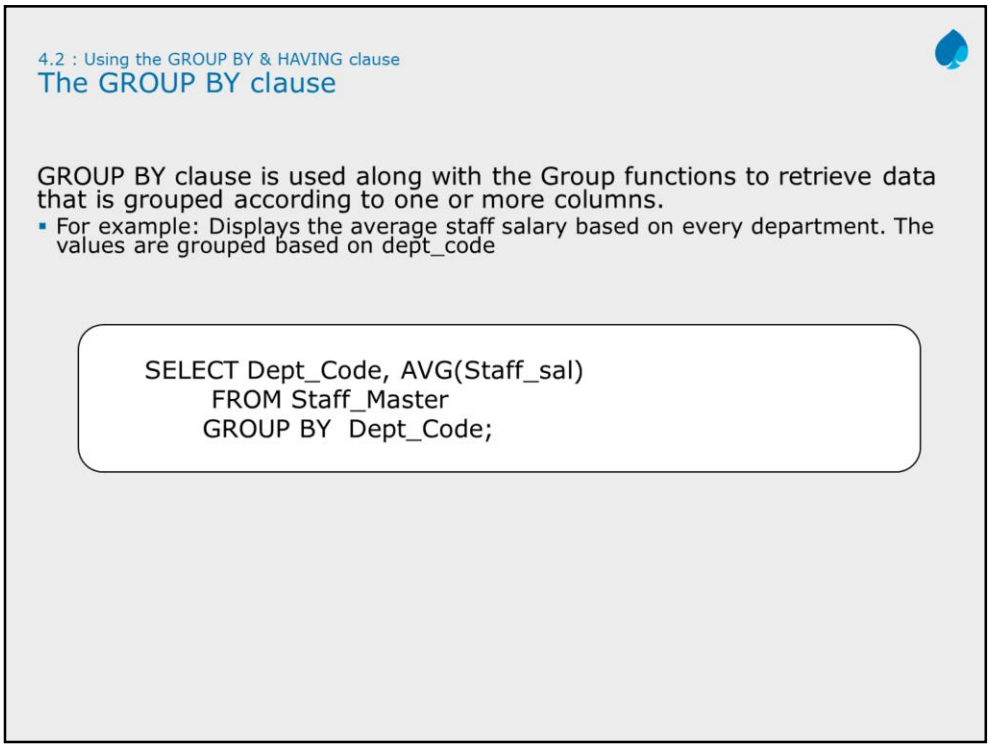


Note:

- The first query returns the value, which counts the number of rows fetched from the student_marks table. All the rows in thetable are treated as one group.Group Functions operate on sets of rows to give one result per group. Can be used on the whole table or certain set of rows

Instructor Notes:

None

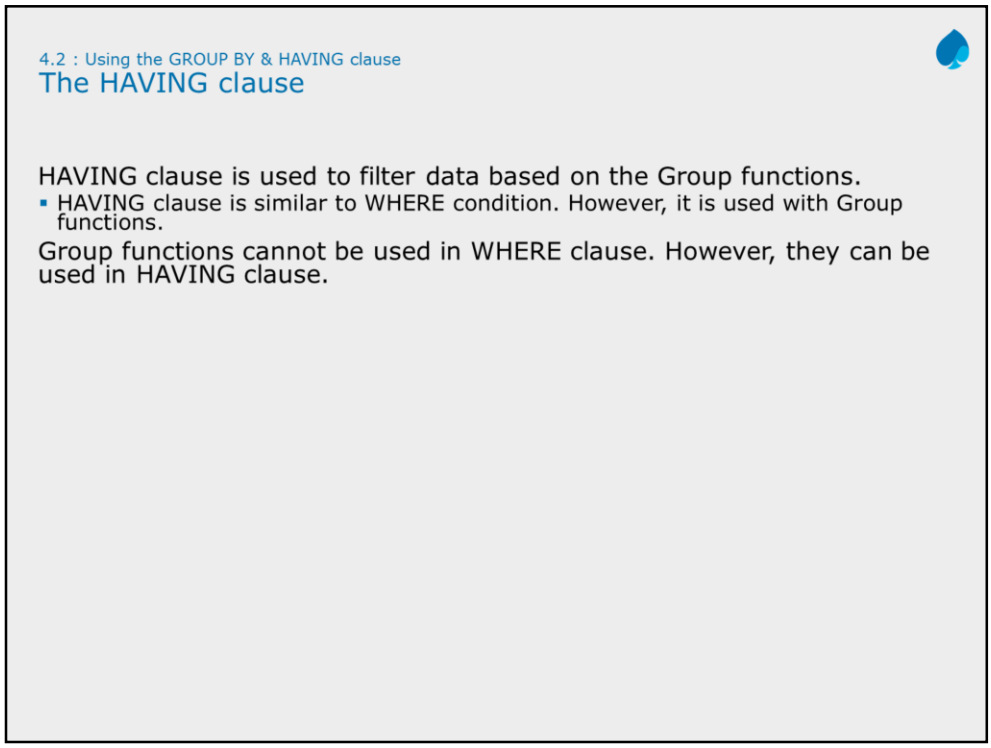


Usage of GROUP BY and HAVING clauses:

- All the SELECT statements we have used until now have acted on data as if the data is in a "single group". But the rows of data in some of the tables can be thought of as being part of "different groups".
  For example: The staff_master table contains staff information allocated to various departments identified by dept_code. If we wish to find the minimum salary of each group of employees in respective department, then none of the clauses that we have seen until now are of any use.
- The GROUP BY clause is used to group the result table derived from earlier FROM and WHERE clauses. Further, a HAVING clause is used to apply search condition on these groups.
  - When a GROUP BY clause is used, each row of the resulting table will represent a group having same values in the column(s) used for grouping.
  - Subsequently, the HAVING clause acts on the resulting grouped table to remove the row that does not satisfy the criteria in the HAVING search condition

Instructor Notes:

None



The HAVING Clause:

- A HAVING clause is of the form:

  HAVING <search condition>

- The HAVING search condition applies to "each group". It can be:
  - formed using various predicates like between, in, like, null, comparison, etc
  - combined with Boolean operators like AND, OR, NOT
- Since the search condition is for a "grouped table". The predicates should be:
  - on a column by which grouping is done.
  - on a set function (Aggregate function) on other columns.
- The aggregate functions can be used in HAVING clause. However, they cannot be used in the WHERE clause.
- When WHERE, GROUP BY, and HAVING clauses are used together in a SELECT statement:
  1. The WHERE clause is processed first in order.
  2. Subsequently, the rows that are returned after the WHERE clause is executed are grouped based on the GROUP BY clause.
  3. Finally, any conditions, on the Group functions in the HAVING clause, are applied to the grouped rows before the final output is displayed.

Instructor Notes:

None

Order of Clause Evaluation:
When included in the same SELECT statement, evaluated in order of:
•WHERE
•GROUP BY
•HAVING

Nesting Functions:
•Inner functions are resolved first

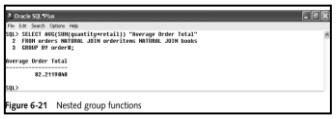


Figure 6-21 Nested group functions

4.2 : Using the GROUP BY & HAVING clause

Examples – GROUP BY and HAVING clause

For example: Display all department numbers having more than five employees.

```
SELECT Department_Code, Count(*)
        FROM Staff_Master
        GROUP BY Department_Code
        HAVING Count(*)> 5;
```

The HAVING clause (contd.):

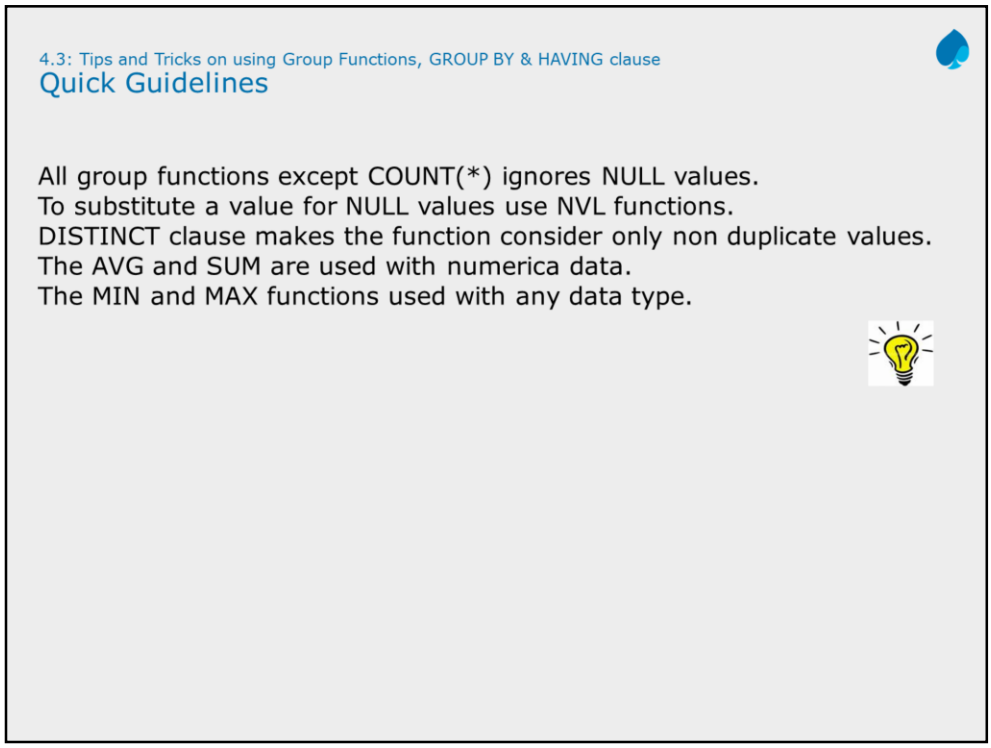
- To find out Average, Maximum, Minimum salary of departments, where average salary is greater than 2000.

```
SELECT dept_code,AVG(staff_sal),MIN(staff_sal),
MAX(staff_sal)
FROM staff_master
GROUP BY dept_code HAVING AVG(staff_sal) >
2000;
```

- To find out average salary of all staff members who belong to department 10 or 20 and their average salary is greater than 10000

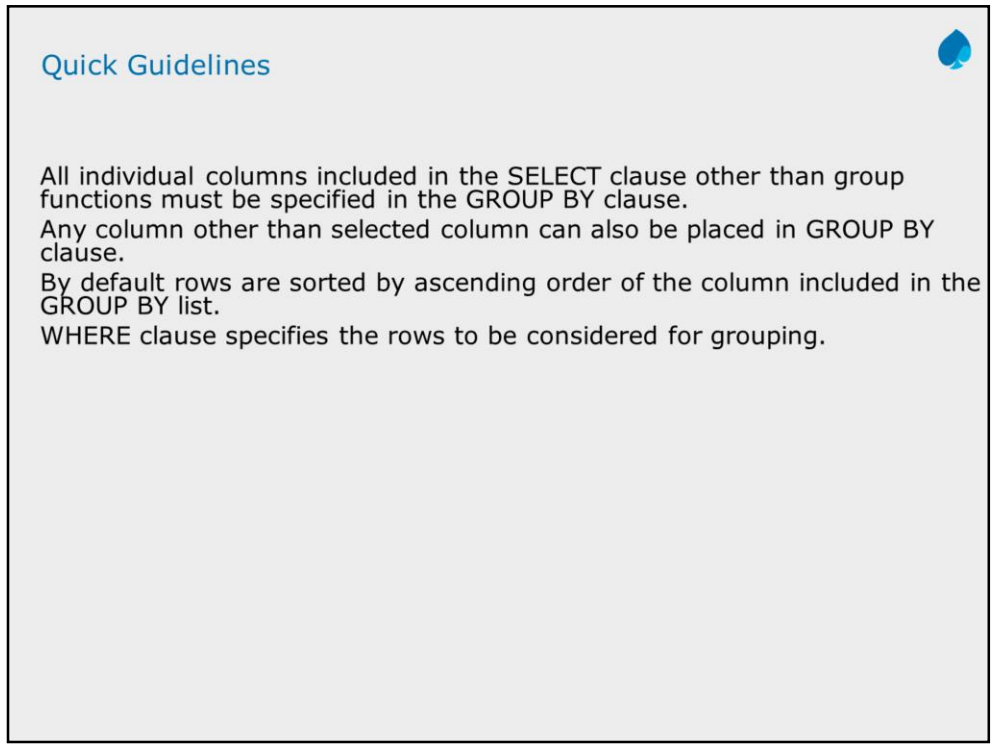
```
SELECT design_code,dept_code,avg(staff_sal)
FROM staff_master where dept_code in(10,20)
GROUP BY  design_code,dept_code
HAVING avg(staff_sal) >10000;
```

Instructor Notes:



NVL function is covered in the next lesson

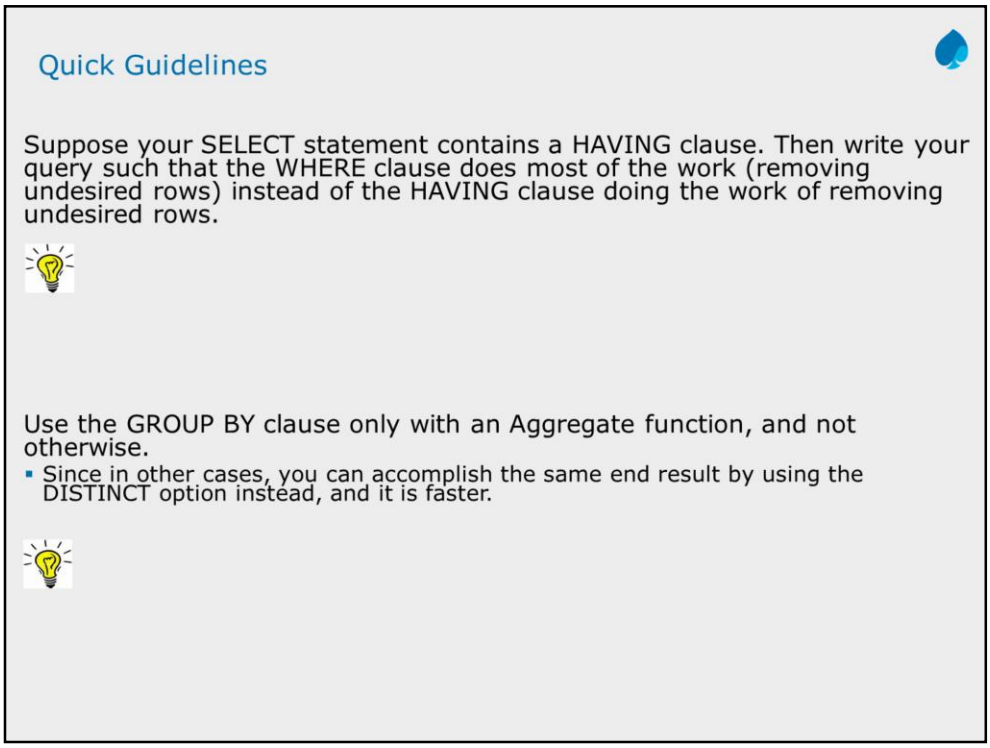
Instructor Notes:



NVL function is covered in the next lesson

Instructor Notes:

None



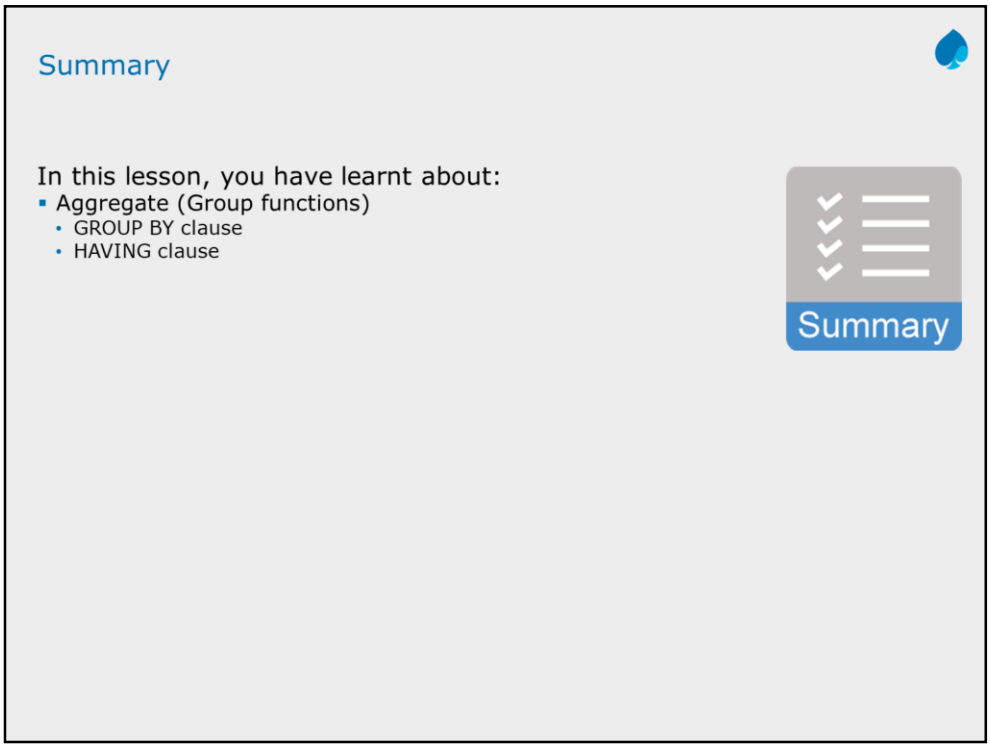
Tips and Tricks:

- By appropriately using the WHERE clause, you can eliminate unnecessary rows before they reach the GROUP BY and HAVING clause. Thus saving some unnecessary work, and boosting performance.
  For example: In a SELECT statement with WHERE, GROUP BY, and HAVING clauses, the query executes in the following sequence.
  - First, the WHERE clause is used to select the appropriate rows that need to be grouped.
  - Next, the GROUP BY clause divides the rows into sets of grouped rows, and then aggregates their values.
  - And last, the HAVING clause then eliminates undesired aggregated groups.
    - If the WHERE clause is used to eliminate as many of the undesired rows as possible, then the GROUP BY and the HAVING clauses will have to do less work. Thus boosting the overall performance of the query.

contd.

Instructor Notes:

None



Instructor Notes:

Answers for Review Questions:

Question 1:
Answers: Option 2, 3

Review – Questions

Question 1: Identify the various group functions from the list given below:

- Option 1: maximum
- Option 2: sum
- Option 3: count
- Option 4: minimum

Instructor Notes:

Answers for Review Questions:

Question 2:
Answer: True

Question 3:
Answer: True

Review – Questions

Question 2: The AVG function ignores NULL values in the column.
- True / False

Question 3: Count(*) returns the number of rows in the table, including duplicates and those with NULLs.
- True / False