

Oracle 11g: SQL

Group Functions

Objectives

- Differentiate between single-row and multiple-row functions
- Use the SUM and AVG functions for numeric calculations
- Use the COUNT function to return the number of records containing non-NULL values
- Use COUNT(*) to include records containing NULL values
- Use the MIN and MAX functions with nonnumeric fields

Objectives (continued)

- Determine when to use the GROUP BY clause to group data
- Identify when the HAVING clause should be used
- List the order of precedence for evaluating WHERE, GROUP BY, and HAVING clauses
- State the maximum depth for nesting group functions
- Nest a group function inside of a single-row function

Objectives (continued)

- Calculate the standard deviation and variance of a set of data, using the STDDEV and VARIANCE functions
- Use composite columns and concatenated groupings in grouping operations

Group Functions

- Return one result per group of rows processed
- Are also called multiple-row and aggregate functions
- All group functions ignore NULL values except COUNT(*)
- Use DISTINCT to suppress duplicate values

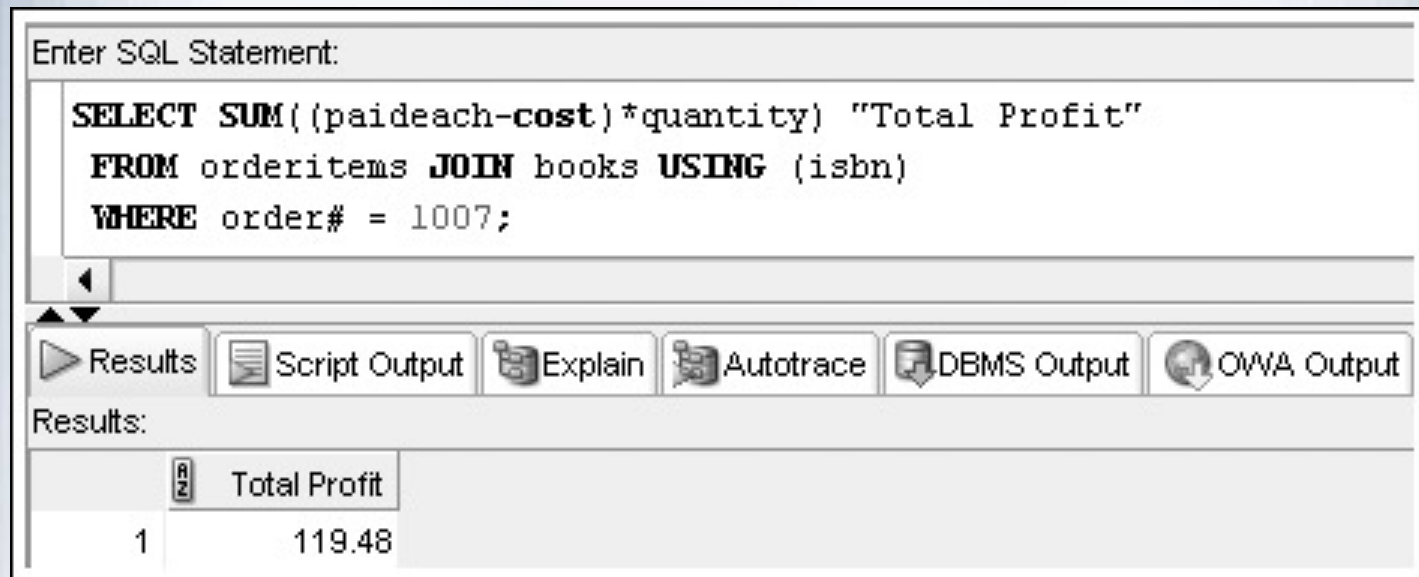
Added Clauses

```
SELECT *|columnname, columnname...  
FROM tablename  
[WHERE condition]  
[GROUP BY columnname, columnname...]  
[HAVING group condition];
```



SUM Function

- Calculates total amount stored in a numeric column for a group of rows



The screenshot shows the Oracle SQL Developer interface. At the top, there is a text area labeled "Enter SQL Statement:" containing the following SQL query:

```
SELECT SUM((paideach-cost)*quantity) "Total Profit"  
FROM orderitems JOIN books USING (isbn)  
WHERE order# = 1007;
```

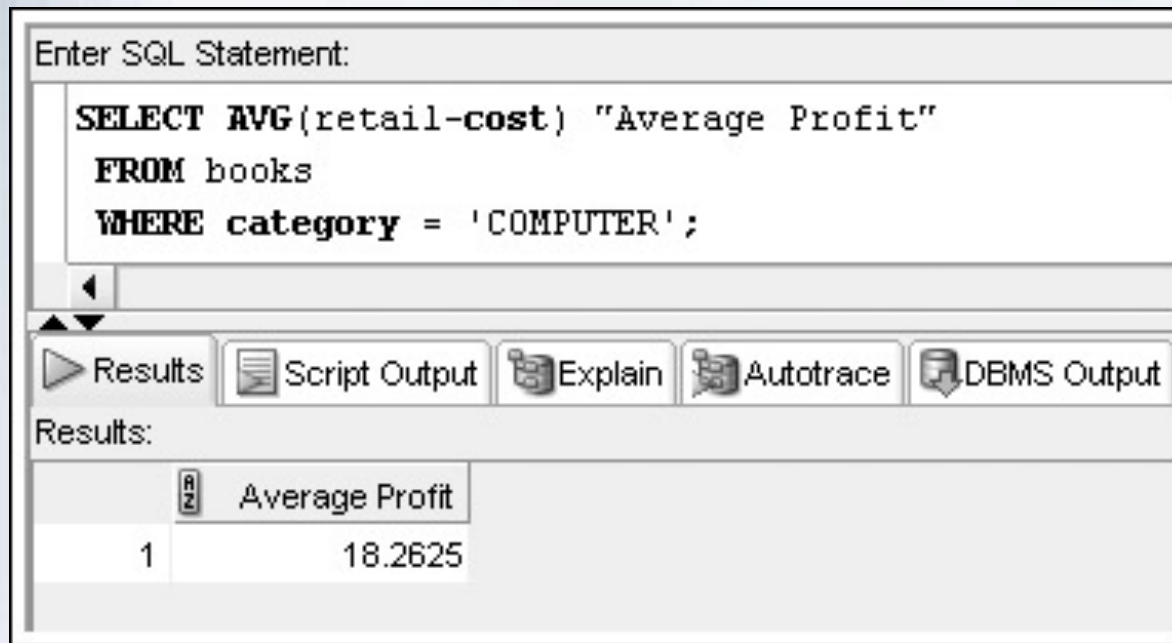
Below the text area is a toolbar with several icons: a play button for "Results", a document icon for "Script Output", a magnifying glass for "Explain", a document with a cursor for "Autotrace", a document with a download icon for "DBMS Output", and a globe for "OWA Output".

Below the toolbar, the "Results:" section displays a table with the following data:

	Total Profit
1	119.48

AVG Function

- Calculates the average of numeric values in a specified column



The screenshot shows a database interface with a text area for entering SQL statements. The entered query is: `SELECT AVG(etail-cost) "Average Profit" FROM books WHERE category = 'COMPUTER';`. Below the text area are buttons for "Results", "Script Output", "Explain", "Autotrace", and "DBMS Output". The "Results" button is active, and the results are displayed in a table below. The table has one column, "Average Profit", and one row with the value 18.2625.

```
Enter SQL Statement:
```

```
SELECT AVG(etail-cost) "Average Profit"
FROM books
WHERE category = 'COMPUTER';
```

Results:

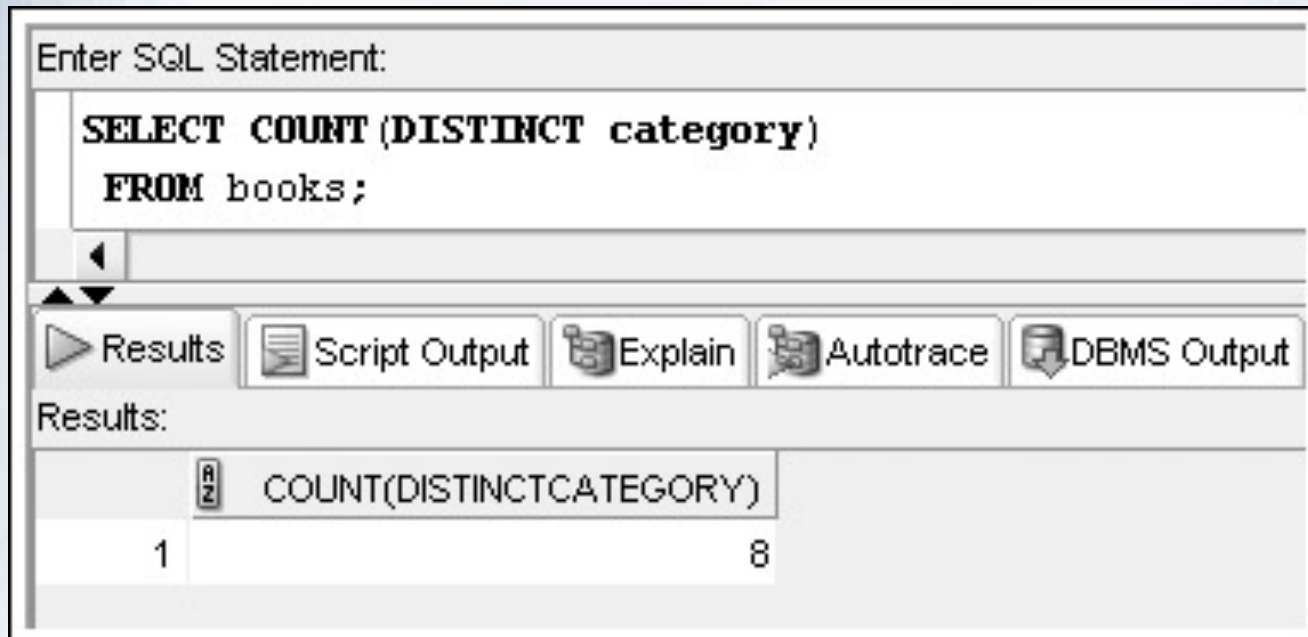
	Average Profit
1	18.2625

COUNT Function

- Two purposes
 - Count non-NULL values
 - Count total records, including those with NULL values

COUNT Function – Non-NULL Values

- Include column name in argument to count number of occurrences

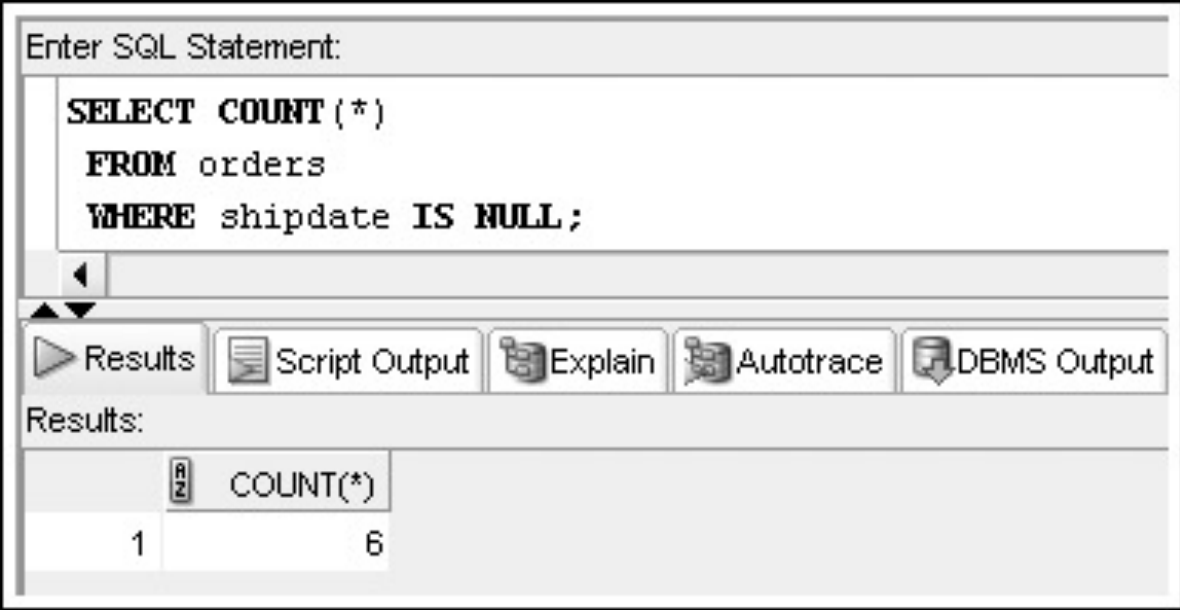


The screenshot shows a web-based SQL interface. At the top, there is a text area labeled "Enter SQL Statement:" containing the query: `SELECT COUNT (DISTINCT category)
FROM books;`. Below the text area is a row of five buttons: "Results" (with a play icon), "Script Output" (with a document icon), "Explain" (with a book icon), "Autotrace" (with a book icon), and "DBMS Output" (with a document icon). The "Results" button is selected. Below the buttons, the word "Results:" is displayed. Underneath, a table shows the query result. The table has one column header, "COUNT(DISTINCTCATEGORY)", and one data row with the value "8".

COUNT(DISTINCTCATEGORY)
8

COUNT Function – NULL Values

- Include asterisk in argument to count number of rows



The screenshot shows a web-based SQL interface. At the top, there is a text area labeled "Enter SQL Statement:" containing the following SQL query:

```
SELECT COUNT (*)  
FROM orders  
WHERE shipdate IS NULL;
```

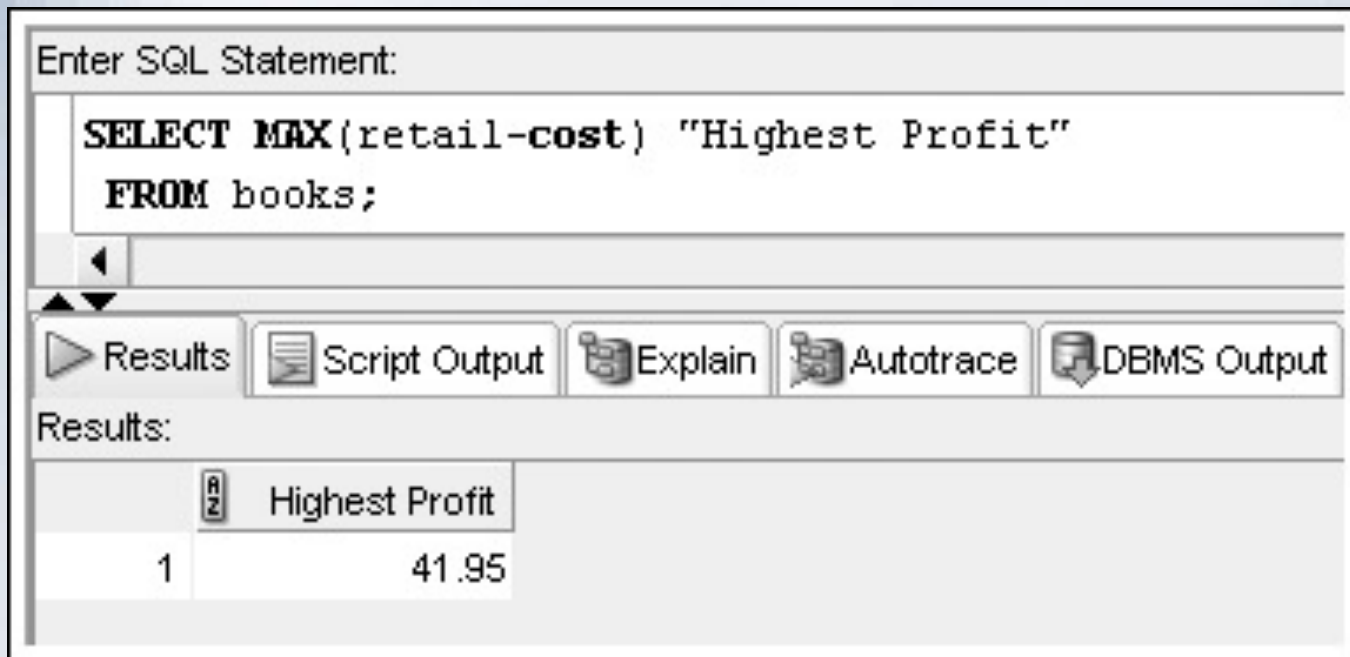
Below the text area is a toolbar with five buttons: "Results" (with a play icon), "Script Output" (with a document icon), "Explain" (with a book icon), "Autotrace" (with a book icon), and "DBMS Output" (with a document icon). The "Results" button is selected.

Below the toolbar, the word "Results:" is displayed. Underneath, a table shows the query results:

	COUNT(*)
1	6

MAX Function

- Returns largest value

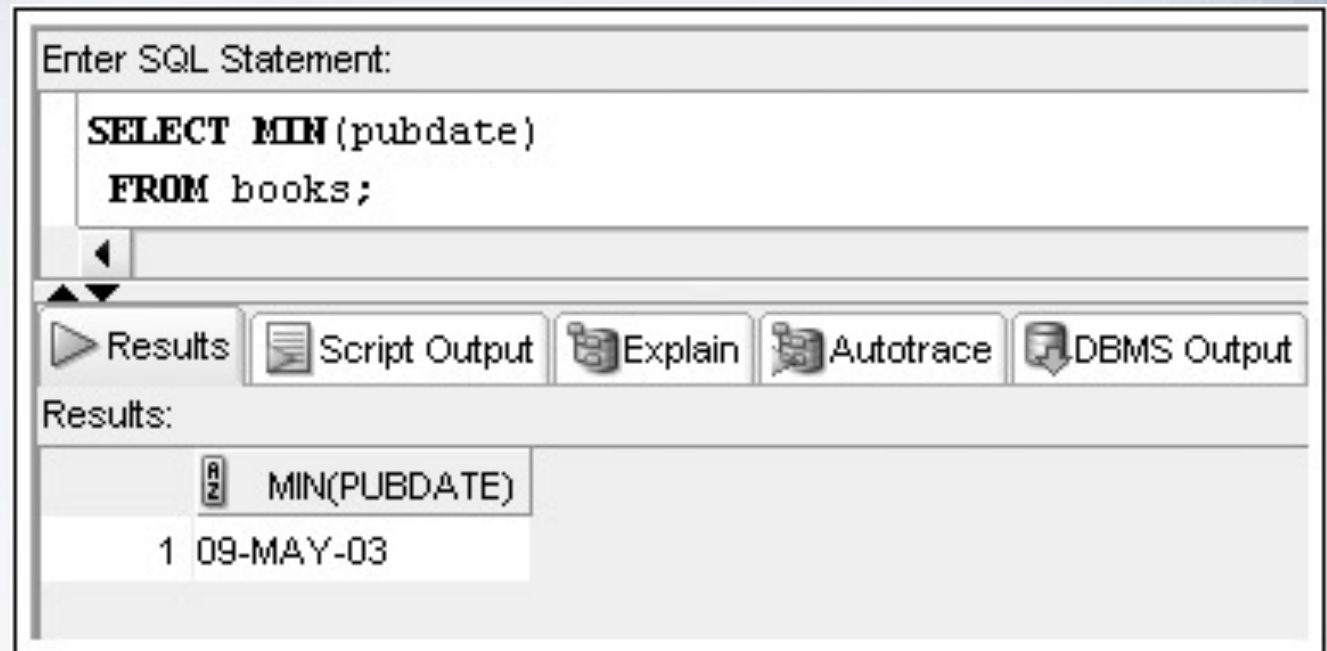


The screenshot shows a web-based SQL interface. At the top, there is a text area labeled "Enter SQL Statement:" containing the query: `SELECT MAX(retail-cost) "Highest Profit" FROM books;`. Below the text area is a toolbar with five buttons: "Results" (with a play icon), "Script Output" (with a document icon), "Explain" (with a database icon), "Autotrace" (with a database icon), and "DBMS Output" (with a database icon). The "Results" button is selected. Below the toolbar, the word "Results:" is displayed. A table shows the query results with one row. The table has two columns: an index column and a column labeled "Highest Profit". The first row has the value "1" in the index column and "41.95" in the "Highest Profit" column.

	Highest Profit
1	41.95

MIN Function

- Returns the smallest value



The screenshot shows a web-based SQL interface. At the top, there is a text area labeled "Enter SQL Statement:" containing the query: `SELECT MIN(pubdate)
FROM books;`. Below the text area is a row of five buttons: "Results" (with a play icon), "Script Output" (with a document icon), "Explain" (with a book icon), "Autotrace" (with a book icon), and "DBMS Output" (with a document icon). The "Results" button is selected. Below the buttons, the word "Results:" is displayed. Underneath, a table shows the query result. The table has one column header "MIN(PUBDATE)" and one data row with the value "09-MAY-03".

	MIN(PUBDATE)
1	09-MAY-03

Datatypes

- The COUNT, MIN, and MAX functions can be used on values with character, numeric, and date datatypes

Grouping Data

- GROUP BY clause
 - Used to group data
 - Must be used for any individual column in the SELECT clause with a group function
 - Cannot reference column aliases

GROUP BY Example

Enter SQL Statement:

```
SELECT category, TO_CHAR(AVG( retail-cost), '999.99') "Profit"  
FROM books  
GROUP BY category;
```

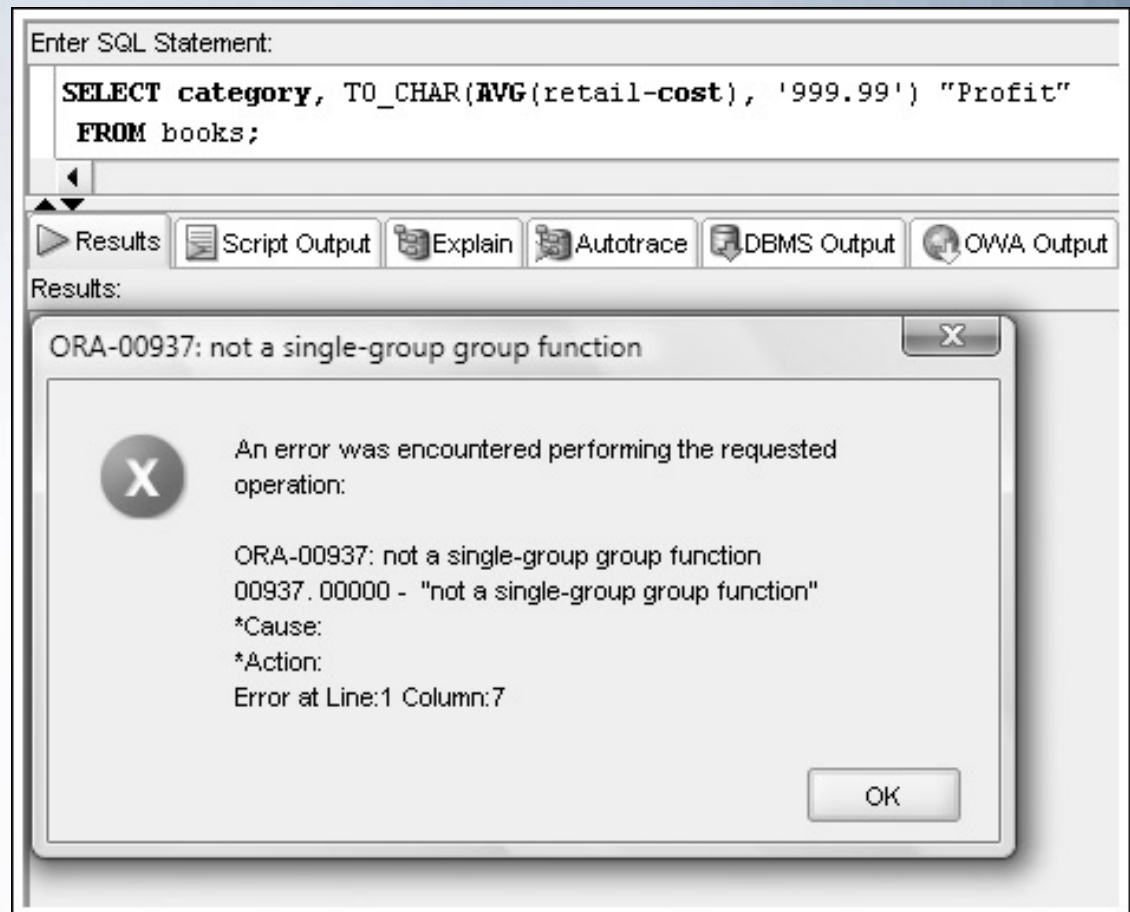
Results Script Output Explain Autotrace DBMS Output OWA Output

Results:

	CATEGORY	Profit
1	COMPUTER	18.26
2	COOKING	8.60
3	CHILDREN	12.89
4	LITERATURE	18.10
5	BUSINESS	16.55
6	FITNESS	12.20
7	FAMILY LIFE	24.88
8	SELF HELP	12.10

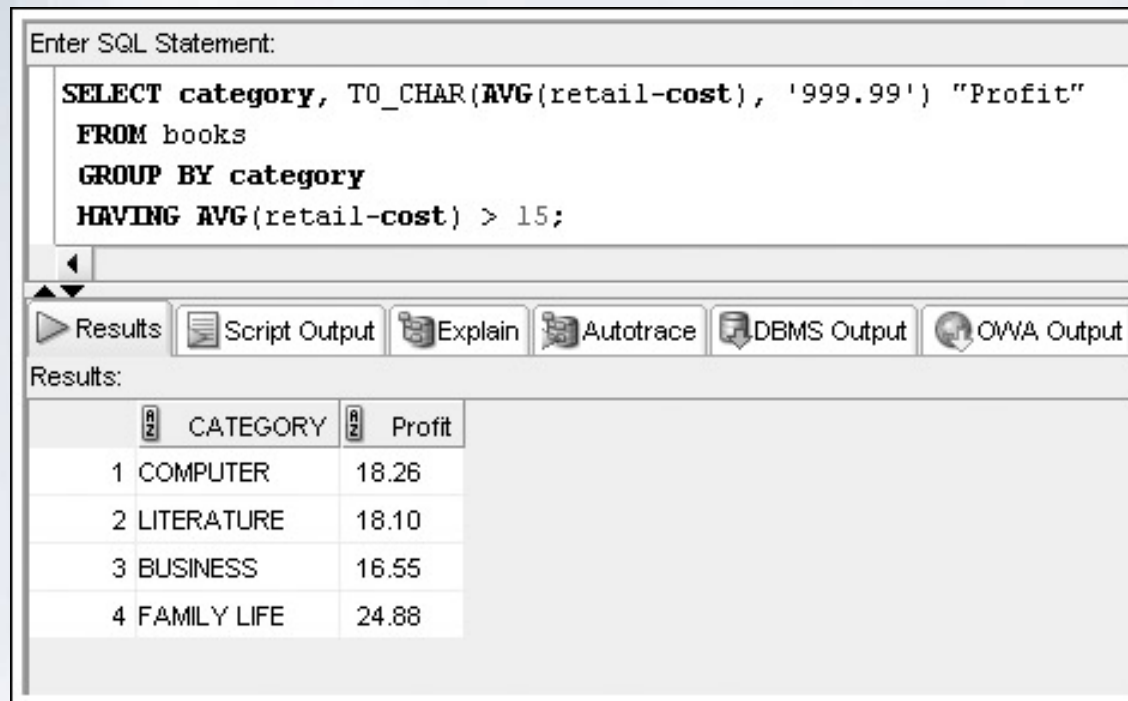
Common Error

- A common error is missing a GROUP BY clause for nonaggregated columns in the SELECT clause



Restricting Aggregated Output

- HAVING clause serves as the WHERE clause for grouped data



The screenshot shows a SQL*Plus window with the following components:

- Enter SQL Statement:** A text area containing the SQL query:

```
SELECT category, TO_CHAR(AVG( retail-cost), '999.99') "Profit"
FROM books
GROUP BY category
HAVING AVG( retail-cost) > 15;
```
- Results:** A row of buttons: Results (selected), Script Output, Explain, Autotrace, DBMS Output, and OWA Output.
- Results:** A table displaying the query results.

	CATEGORY	Profit
1	COMPUTER	18.26
2	LITERATURE	18.10
3	BUSINESS	16.55
4	FAMILY LIFE	24.88






Restricting Aggregated Output (continued)



- When included in the same SELECT statement, the clauses are evaluated in the order of:
 - WHERE
 - GROUP BY
 - HAVING

Restricting Aggregated Output (continued)

Enter SQL Statement:

```
SELECT category, TO_CHAR(AVG( retail-cost), '999.99') "Profit"  
FROM books  
WHERE pubdate > '01-JAN-05'  
GROUP BY category  
HAVING AVG( retail-cost) > 15;
```

Results:     

	 CATEGORY	 Profit
1	COMPUTER	16.17
2	LITERATURE	18.10

Nesting Functions

- Inner function is resolved first
- Maximum nesting depth: 2

[illegible]

Statistical Group Functions

- Based on normal distribution
- Includes:
 - STDDEV
 - VARIANCE

STDDEV Function

Enter SQL Statement:

```
SELECT category, COUNT(*), TO_CHAR(AVG( retail-cost ),'999.99') "Avg",  
        TO_CHAR(STDDEV( retail-cost ),'999.9999') "Stddev"  
FROM books  
GROUP BY category;
```

Results Script Output Explain Autotrace DBMS Output OWA Output

Results:

	 CATEGORY	 COUNT(*)	 Avg	 Stddev
1	COMPUTER	4	18.26	11.2267
2	COOKING	2	8.60	1.6263
3	CHILDREN	2	12.89	13.0956
4	LITERATURE	1	18.10	.0000
5	BUSINESS	1	16.55	.0000
6	FITNESS	1	12.20	.0000
7	FAMILY LIFE	2	24.88	24.1477
8	SELF HELP	1	12.10	.0000

VARIANCE Function

- Determines data dispersion within a group

Enter SQL Statement:

```
SELECT category, TO_CHAR(VARIANCE(retail-cost), '999.99') "Var",
      MIN(retail-cost) "Min", MAX(retail-cost) "Max"
FROM books
GROUP BY category;
```

Results Script Output Explain Autotrace DBMS Output OWA Output

Results:

	CATEGORY	Var	Min	Max
1	COMPUTER	126.04	3.2	28.7
2	COOKING	2.65	7.45	9.75
3	CHILDREN	171.50	3.63	22.15
4	LITERATURE	.00	18.1	18.1
5	BUSINESS	.00	16.55	16.55
6	FITNESS	.00	12.2	12.2
7	FAMILY LIFE	583.11	7.8	41.95
8	SELF HELP	.00	12.1	12.1

Summary

- The AVG, SUM, STDDEV, and VARIANCE functions are used only with numeric fields
- The COUNT, MAX, and MIN functions can be applied to any datatype
- The AVG, SUM, MAX, MIN, STDDEV, and VARIANCE functions all ignore NULL values
- By default, the AVG, SUM, MAX, MIN, COUNT, STDDEV, and VARIANCE functions include duplicate values

Summary (continued)

- The GROUP BY clause is used to divide table data into groups
- If a SELECT clause contains both an individual field name and a group function, the field name must also be included in a GROUP BY clause
- The HAVING clause is used to restrict groups in a group function
- Group functions can be nested to a depth of only two. The inner function is always performed first, using the specified grouping. The results of the inner function are used as input for the outer function.

Summary (continued)

- The STDDEV and VARIANCE functions are used to perform statistical analyses on a set of data