**SQL Introduction – 8**

**DML (Data Manipulation Language)**

# SQL Functions

SQL has many built-in functions for performing calculations on data.

SQL Aggregate Functions

SQL aggregate functions return a single value, calculated from values in a column.

Useful aggregate functions:

* AVG() - Returns the average value
* COUNT() - Returns the number of rows
* FIRST() - Returns the first value
* LAST() - Returns the last value
* MAX() - Returns the largest value
* MIN() - Returns the smallest value
* SUM() - Returns the sum

## The AVG() Function

The AVG() function returns the average value of a numeric column.

### SQL AVG() Syntax

SELECT AVG(column\_name) FROM table\_name

## Demo Database

In this tutorial we will use the well-known Northwind sample database.

Below is a selection from the "Products" table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ProductID** | **ProductName** | **SupplierID** | **CategoryID** | **Unit** | **Price** |
| 1 | Chais | 1 | 1 | 10 boxes x 20 bags | 18 |
| 2 | Chang | 1 | 1 | 24 - 12 oz bottles | 19 |
| 3 | Aniseed Syrup | 1 | 2 | 12 - 550 ml bottles | 10 |
| 4 | Chef Anton's Cajun Seasoning | 2 | 2 | 48 - 6 oz jars | 21.35 |
| 5 | Chef Anton's Gumbo Mix | 2 | 2 | 36 boxes | 25 |

The following SQL statement gets the average value of the "Price" column from the "Products" table:

## Example

SELECT AVG(Price) AS PriceAverage FROM Products;

The following SQL statement selects the "ProductName" and "Price" records that have an above average price:

## Example

SELECT ProductName, Price FROM Products
WHERE Price>(SELECT AVG(Price) FROM Products);

## The COUNT() Function

The COUNT() function returns the number of rows that matches a specified criteria.

### SQL COUNT(column\_name) Syntax

The COUNT(column\_name) function returns the number of values (NULL values will not be counted) of the specified column:

SELECT COUNT(column\_name) FROM table\_name;

### SQL COUNT(\*) Syntax

The COUNT(\*) function returns the number of records in a table:

SELECT COUNT(\*) FROM table\_name;

### SQL COUNT(DISTINCT column\_name) Syntax

The COUNT(DISTINCT column\_name) function returns the number of distinct values of the specified column:

SELECT COUNT(DISTINCT column\_name) FROM table\_name;

**Note:** COUNT(DISTINCT) works with ORACLE and Microsoft SQL Server, but not with Microsoft Access.

## Demo Database

In this tutorial we will use the well-known Northwind sample database.

Below is a selection from the "Orders" table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderID** | **CustomerID** | **EmployeeID** | **OrderDate** | **ShipperID** |
| 10265 | 7 | 2 | 1996-07-25 | 1 |
| 10266 | 87 | 3 | 1996-07-26 | 3 |
| 10267 | 25 | 4 | 1996-07-29 | 1 |

The following SQL statement counts the number of orders from "CustomerID"=7 from the "Orders" table:

## Example

SELECT COUNT(CustomerID) AS OrdersFromCustomerID7 FROM Orders
WHERE CustomerID=7;

The following SQL statement counts the total number of orders in the "Orders" table:

## Example

SELECT COUNT(\*) AS NumberOfOrders FROM Orders;

The following SQL statement counts the number of unique customers in the "Orders" table:

## Example

SELECT COUNT(DISTINCT CustomerID) AS NumberOfCustomers FROM Orders;

## The FIRST() Function

The FIRST() function returns the first value of the selected column.

### SQL FIRST() Syntax

SELECT FIRST(column\_name) FROM table\_name;

**Note:** The FIRST() function is only supported in MS Access.

### MySQL Syntax

SELECT column\_name FROM table\_name
ORDER BY column\_name ASC
LIMIT 1;

### Example

SELECT CustomerName FROM Customers
ORDER BY CustomerID ASC
LIMIT 1;

## The LAST() Function

The LAST() function returns the last value of the selected column.

### SQL LAST() Syntax

SELECT LAST(column\_name) FROM table\_name;

**Note:** The LAST() function is only supported in MS Access.

### MySQL Syntax

SELECT column\_name FROM table\_name
ORDER BY column\_name DESC
LIMIT 1;

### Example

SELECT CustomerName FROM Customers
ORDER BY CustomerID DESC
LIMIT 1;

## The MAX() Function

The MAX() function returns the largest value of the selected column.

### SQL MAX() Syntax

SELECT MAX(column\_name) FROM table\_name;

The following SQL statement gets the largest value of the "Price" column from the "Products" table:

## Example

SELECT MAX(Price) AS HighestPrice FROM Products;

## The MIN() Function

The MIN() function returns the smallest value of the selected column.

### SQL MIN() Syntax

SELECT MIN(column\_name) FROM table\_name;

The following SQL statement gets the smallest value of the "Price" column from the "Products" table:

## Example

SELECT MIN(Price) AS SmallestOrderPrice FROM Products;

## The SUM() Function

The SUM() function returns the total sum of a numeric column.

### SQL SUM() Syntax

SELECT SUM(column\_name) FROM table\_name;

## Demo Database

In this tutorial we will use the well-known Northwind sample database.

Below is a selection from the "OrderDetails" table:

|  |  |  |  |
| --- | --- | --- | --- |
| **OrderDetailID** | **OrderID** | **ProductID** | **Quantity** |
| 1 | 10248 | 11 | 12 |
| 2 | 10248 | 42 | 10 |
| 3 | 10248 | 72 | 5 |
| 4 | 10249 | 14 | 9 |
| 5 | 10249 | 51 | 40 |

The following SQL statement finds the sum of all the "Quantity" fields for the "OrderDetails" table:

## Example

SELECT SUM(Quantity) AS TotalItemsOrdered FROM OrderDetails;

## The GROUP BY Statement

The GROUP BY statement is used in conjunction with the aggregate functions to group the result-set by one or more columns.

### SQL GROUP BY Syntax

SELECT column\_name, aggregate\_function(column\_name)
FROM table\_name
WHERE column\_name operator value
GROUP BY column\_name;

## Demo Database

In this tutorial we will use the well-known Northwind sample database.

Below is a selection from the "Orders" table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderID** | **CustomerID** | **EmployeeID** | **OrderDate** | **ShipperID** |
| 10248 | 90 | 5 | 1996-07-04 | 3 |
| 10249 | 81 | 6 | 1996-07-05 | 1 |
| 10250 | 34 | 4 | 1996-07-08 | 2 |

And a selection from the "Shippers" table:

|  |  |  |
| --- | --- | --- |
| **ShipperID** | **ShipperName** | **Phone** |
| 1 | Speedy Express | (503) 555-9831 |
| 2 | United Package | (503) 555-3199 |
| 3 | Federal Shipping | (503) 555-9931 |

And a selection from the "Employees" table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **EmployeeID** | **LastName** | **FirstName** | **BirthDate** | **Photo** | **Notes** |
| 1 | Davolio | Nancy | 1968-12-08 | EmpID1.pic | Education includes a BA.... |
| 2 | Fuller | Andrew | 1952-02-19 | EmpID2.pic | Andrew received his BTS.... |
| 3 | Leverling | Janet | 1963-08-30 | EmpID3.pic | Janet has a BS degree.... |

## SQL GROUP BY Example

Now we want to find the number of orders sent by each shipper.

The following SQL statement counts as orders grouped by shippers:

## Example

SELECT Shippers.ShipperName,COUNT(Orders.OrderID) AS NumberOfOrders FROM Orders
LEFT JOIN Shippers
ON Orders.ShipperID=Shippers.ShipperID
GROUP BY ShipperName;

## GROUP BY More Than One Column

We can also use the GROUP BY statement on more than one column, like this:

## Example

SELECT Shippers.ShipperName, Employees.LastName,
COUNT(Orders.OrderID) AS NumberOfOrders
FROM ((Orders
INNER JOIN Shippers
ON Orders.ShipperID=Shippers.ShipperID)
INNER JOIN Employees
ON Orders.EmployeeID=Employees.EmployeeID)
GROUP BY ShipperName,LastName;

## The HAVING Clause

The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

### SQL HAVING Syntax

SELECT column\_name, aggregate\_function(column\_name)
FROM table\_name
WHERE column\_name operator value
GROUP BY column\_name
HAVING aggregate\_function(column\_name) operator value;

## Demo Database

In this tutorial we will use the well-known Northwind sample database.

Below is a selection from the "Orders" table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OrderID** | **CustomerID** | **EmployeeID** | **OrderDate** | **ShipperID** |
| 10248 | 90 | 5 | 1996-07-04 | 3 |
| 10249 | 81 | 6 | 1996-07-05 | 1 |
| 10250 | 34 | 4 | 1996-07-08 | 2 |

And a selection from the "Employees" table:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **EmployeeID** | **LastName** | **FirstName** | **BirthDate** | **Photo** | **Notes** |
| 1 | Davolio | Nancy | 1968-12-08 | EmpID1.pic | Education includes a BA.... |
| 2 | Fuller | Andrew | 1952-02-19 | EmpID2.pic | Andrew received his BTS.... |
| 3 | Leverling | Janet | 1963-08-30 | EmpID3.pic | Janet has a BS degree.... |

## SQL HAVING Example

Now we want to find if any of the customers have a total order of less than 2000.

We use the following SQL statement:

The following SQL statement finds if any of the employees has registered more than 10 orders:

## Example

SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders FROM (Orders
INNER JOIN Employees
ON Orders.EmployeeID=Employees.EmployeeID)
GROUP BY LastName
HAVING COUNT(Orders.OrderID) > 10;

ow we want to find the if the employees "Davolio" or "Fuller" have more than 25 orders

We add an ordinary WHERE clause to the SQL statement:

## Example

SELECT Employees.LastName, COUNT(Orders.OrderID) AS NumberOfOrders FROM Orders
INNER JOIN Employees
ON Orders.EmployeeID=Employees.EmployeeID
WHERE LastName='Davolio' OR LastName='Fuller'
GROUP BY LastName
HAVING COUNT(Orders.OrderID) > 25;