**The SQL EXISTS Operator**

The EXISTS operator is used to test for the existence of any record in a subquery.

The EXISTS operator returns true if the subquery returns one or more records.

**EXISTS Syntax**

SELECT *column\_name(s)*  
FROM *table\_name*  
WHERE EXISTS  
(SELECT *column\_name* FROM *table\_name* WHERE *condition*);

**Demo Database**

Below is a selection from the "Products" table in the Northwind sample database:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 | 22 |
| 5 | Chef Anton's Gumbo Mix | 2 | 2 | 36 boxes | 21.35 |

And a selection from the "Suppliers" table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| SupplierID | SupplierName | ContactName | Address | City | PostalCode | Country |
| 1 | Exotic Liquid | Charlotte Cooper | 49 Gilbert St. | London | EC1 4SD | UK |
| 2 | New Orleans Cajun Delights | Shelley Burke | P.O. Box 78934 | New Orleans | 70117 | USA |
| 3 | Grandma Kelly's Homestead | Regina Murphy | 707 Oxford Rd. | Ann Arbor | 48104 | USA |
| 4 | Tokyo Traders | Yoshi Nagase | 9-8 Sekimai Musashino-shi | Tokyo | 100 | Japan |

**SQL EXISTS Examples**

The following SQL statement returns TRUE and lists the suppliers with a product price less than 20:

**Example**

SELECT SupplierName  
FROM Suppliers  
WHERE EXISTS (SELECT ProductName FROM Products WHERE Products.SupplierID = Suppliers.supplierID AND Price < 20);

The following SQL statement returns TRUE and lists the suppliers with a product price equal to 22:

**Example**

SELECT SupplierName  
FROM Suppliers  
WHERE EXISTS (SELECT ProductName FROM Products WHERE Products.SupplierID = Suppliers.supplierID AND Price = 22);

**The SQL ANY and ALL Operators**

The ANY and ALL operators are used with a WHERE or HAVING clause.

The ANY operator returns true if **any** of the subquery values meet the condition.

The ALL operator returns true if **all** of the subquery values meet the condition.

**ANY Syntax**

SELECT *column\_name(s)*  
FROM *table\_name*  
WHERE *column\_name operator* ANY  
(SELECT *column\_name* FROM *table\_name* WHERE *condition*);

**ALL Syntax**

SELECT *column\_name(s)*  
FROM *table\_name*  
WHERE *column\_name operator* ALL  
(SELECT *column\_name* FROM *table\_name* WHERE *condition*);

**Note:** The *operator* must be a standard comparison operator (=, <>, !=, >, >=, <, or <=).

**Demo Database**

Below is a selection from the "Products" table in the Northwind sample database:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 | 22 |
| 5 | Chef Anton's Gumbo Mix | 2 | 2 | 36 boxes | 21.35 |

And 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 |

**SQL ANY Examples**

The ANY operator returns TRUE if any of the subquery values meet the condition.

The following SQL statement returns TRUE and lists the product names if it finds ANY records in the OrderDetails table that quantity = 10:

**Example**

SELECT ProductName  
FROM Products  
WHERE ProductID = ANY (SELECT ProductID FROM OrderDetails WHERE Quantity = 10);

The following SQL statement returns TRUE and lists the product names if it finds ANY records in the OrderDetails table that quantity > 99:

**Example**

SELECT ProductName  
FROM Products  
WHERE ProductID = ANY (SELECT ProductID FROM OrderDetails WHERE Quantity > 99);

**SQL ALL Example**

The ALL operator returns TRUE if all of the subquery values meet the condition.

The following SQL statement returns TRUE and lists the product names if ALL the records in the OrderDetails table has quantity = 10 (so, this example will return FALSE, because not ALL records in the OrderDetails table has quantity = 10):

**Example**

SELECT ProductName  
FROM Products  
WHERE ProductID = ALL (SELECT ProductID FROM OrderDetails WHERE Quantity = 10);

## The SQL CASE Statement

The CASE statement goes through conditions and returns a value when the first condition is met (like an IF-THEN-ELSE statement). So, once a condition is true, it will stop reading and return the result. If no conditions are true, it returns the value in the ELSE clause.

If there is no ELSE part and no conditions are true, it returns NULL.

## CASE Syntax

CASE  
    WHEN condition1 THEN result1  
    WHEN condition2 THEN result2  
    WHEN conditionN THEN resultN  
    ELSE result  
END;

## Demo Database

Below is a selection from the "OrderDetails" table in the Northwind sample database:

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

## SQL CASE Examples

The following SQL goes through conditions and returns a value when the first condition is met:

### Example

SELECT OrderID, Quantity,  
CASE  
    WHEN Quantity > 30 THEN "The quantity is greater than 30"  
    WHEN Quantity = 30 THEN "The quantity is 30"  
    ELSE "The quantity is under 30"  
END AS QuantityText  
FROM OrderDetails;

The following SQL will order the customers by City. However, if City is NULL, then order by Country:

### Example

SELECT CustomerName, City, Country  
FROM Customers  
ORDER BY  
(CASE  
    WHEN City IS NULL THEN Country  
    ELSE City  
END);

## SQL IFNULL(), ISNULL(), COALESCE(), and NVL() Functions

Look at the following "Products" table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P\_Id | ProductName | UnitPrice | UnitsInStock | UnitsOnOrder |
| 1 | Jarlsberg | 10.45 | 16 | 15 |
| 2 | Mascarpone | 32.56 | 23 |  |
| 3 | Gorgonzola | 15.67 | 9 | 20 |

Suppose that the "UnitsOnOrder" column is optional, and may contain NULL values.

Look at the following SELECT statement:

SELECT ProductName, UnitPrice \* (UnitsInStock + UnitsOnOrder)  
FROM Products;

In the example above, if any of the "UnitsOnOrder" values are NULL, the result will be NULL.

## Solutions

**MySQL**

The MySQL [IFNULL()](https://www.w3schools.com/sql/func_mysql_ifnull.asp) function lets you return an alternative value if an expression is NULL:

SELECT ProductName, UnitPrice \* (UnitsInStock + IFNULL(UnitsOnOrder, 0))  
FROM Products;

or we can use the [COALESCE()](https://www.w3schools.com/sql/func_mysql_coalesce.asp) function, like this:

SELECT ProductName, UnitPrice \* (UnitsInStock + COALESCE(UnitsOnOrder, 0))  
FROM Products;

**SQL Server**

The SQL Server [ISNULL()](https://www.w3schools.com/sql/func_sqlserver_isnull.asp) function lets you return an alternative value when an expression is NULL:

SELECT ProductName, UnitPrice \* (UnitsInStock + ISNULL(UnitsOnOrder, 0))  
FROM Products;

**MS Access**

The MS Access [IsNull()](https://www.w3schools.com/sql/func_msaccess_isnull.asp) function returns TRUE (-1) if the expression is a null value, otherwise FALSE (0):

SELECT ProductName, UnitPrice \* (UnitsInStock + IIF(IsNull(UnitsOnOrder), 0, UnitsOnOrder))  
FROM Products;

**Oracle**

The Oracle NVL() function achieves the same result:

SELECT ProductName, UnitPrice \* (UnitsInStock + NVL(UnitsOnOrder, 0))  
FROM Products;