**Case statement syntax:**

select x,y,z, case when x+y>z and y+z>z and x+z>y

then 'Yes' else 'No' end as 'triangle' from Triangle;

**The SQL UNION Operator**

The UNION operator is used to combine the result-set of two or more SELECT statements.

* Every SELECT statement within UNION must have the same number of columns
* The columns must also have similar data types
* The columns in every SELECT statement must also be in the same order

**SELECT *column\_name(s)* FROM *table1*  
UNION  
SELECT *column\_name(s)* FROM *table2*;**

SELECT 'Customer' AS Type, ContactName, City, Country  
FROM Customers  
UNION  
SELECT 'Supplier', ContactName, City, Country  
FROM Suppliers;

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.

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

SELECT SupplierName

FROM Suppliers

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

**COPY**

**SELECT \* INTO CustomersBackup2017  
FROM Customers;**

**The following SQL statement creates a backup copy of Customers:**

Tip: SELECT INTO can also be used to create a new, empty table using the schema of another. Just add a WHERE clause that causes the query to return no data:

SELECT \* INTO *newtable*  
FROM *oldtable*  
WHERE 1 = 0;

**Insert Into Command**

INSERT INTO Customers (CustomerName, City, Country)  
SELECT SupplierName, City, Country FROM Suppliers;

INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)  
SELECT SupplierName, ContactName, Address, City, PostalCode, Country FROM Suppliers;

**Null Values Handling**

**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;

**What is a Stored Procedure?**

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it.

You can also pass parameters to a stored procedure, so that the stored procedure can act based on the parameter value(s) that is passed.

CREATE PROCEDURE SelectAllCustomers  
AS  
SELECT \* FROM Customers  
GO;

EXEC SelectAllCustomers;

**Parameteric:**

CREATE PROCEDURE SelectAllCustomers @City nvarchar(30)  
AS  
SELECT \* FROM Customers WHERE City = @City  
GO;

EXEC SelectAllCustomers @City = 'London';

CREATE PROCEDURE SelectAllCustomers @City nvarchar(30), @PostalCode nvarchar(10)  
AS  
SELECT \* FROM Customers WHERE City = @City AND PostalCode = @PostalCode  
GO;

EXEC SelectAllCustomers @City = 'London', @PostalCode = 'WA1 1DP';