The IF statement

The following illustrates the syntax of the IF statement:

IF boolean\_expression

BEGIN

{ statement\_block }

END

Code language: SQL (Structured Query Language) (sql)

In this syntax, if the Boolean\_expression evaluates to TRUE then the statement\_block in the [BEGIN...END](https://www.sqlservertutorial.net/sql-server-stored-procedures/sql-server-begin-end/) block is executed. Otherwise, the statement\_block is skipped and the control of the program is passed to the statement after the END keyword.

Note that if the Boolean expression contains a SELECT statement, you must enclose the SELECT statement in parentheses.

The following example first gets the sales amount from the sales.order\_items table in the [sample database](https://www.sqlservertutorial.net/sql-server-sample-database/) and then prints out a message if the sales amount is greater than 1 million.

BEGIN

DECLARE @sales INT;

SELECT

@sales = SUM(list\_price \* quantity)

FROM

sales.order\_items i

INNER JOIN sales.orders o ON o.order\_id = i.order\_id

WHERE

YEAR(order\_date) = 2018;

SELECT @sales;

IF @sales > 1000000

BEGIN

PRINT 'Great! The sales amount in 2018 is greater than 1,000,000';

END

END

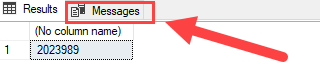
Code language: SQL (Structured Query Language) (sql)

The output of the code block is:

Great! The sales amount in 2018 is greater than 1,000,000

Code language: SQL (Structured Query Language) (sql)

Note that you have to click the **Messages** tab to see the above output message:



The IF ELSE statement

When the condition in the IF clause evaluates to FALSE and you want to execute another statement block, you can use the ELSE clause.

The following illustrates the IF ELSE statement:

IF Boolean\_expression

BEGIN

-- Statement block executes when the Boolean expression is TRUE

END

ELSE

BEGIN

-- Statement block executes when the Boolean expression is FALSE

END

Code language: SQL (Structured Query Language) (sql)

Each IF statement has a condition. If the condition evaluates to TRUE then the statement block in the IF clause is executed. If the condition is FALSE, then the code block in the ELSE clause is executed.

See the following example:

BEGIN

DECLARE @sales INT;

SELECT

@sales = SUM(list\_price \* quantity)

FROM

sales.order\_items i

INNER JOIN sales.orders o ON o.order\_id = i.order\_id

WHERE

YEAR(order\_date) = 2017;

SELECT @sales;

IF @sales > 10000000

BEGIN

PRINT 'Great! The sales amount in 2018 is greater than 10,000,000';

END

ELSE

BEGIN

PRINT 'Sales amount in 2017 did not reach 10,000,000';

END

END

Code language: SQL (Structured Query Language) (sql)

In this example:

First, the following statement sets the total sales in 2017 to the @sales variable:

SELECT

@sales = SUM(list\_price \* quantity)

FROM

sales.order\_items i

INNER JOIN sales.orders o ON o.order\_id = i.order\_id

WHERE

YEAR(order\_date) = 2017;

Code language: SQL (Structured Query Language) (sql)

Second, this statement returns the sales to the output:

SELECT @sales;

Code language: SQL (Structured Query Language) (sql)

Finally, the IF clause checks if the sales amount in 2017 is greater than 10 million. Because the sales amount is less than that, the statement block in the ELSE clause executes.

IF @sales > 10000000

BEGIN

PRINT 'Great! The sales amount in 2018 is greater than 10,000,000';

END

ELSE

BEGIN

PRINT 'Sales amount in 2017 did not reach 10,000,000';

END

Code language: SQL (Structured Query Language) (sql)

The following shows the output:

Sales amount did not reach 10,000,000

Code language: SQL (Structured Query Language) (sql)

Nested IF...ELSE

SQL Server allows you to nest an IF...ELSE statement within inside another IF...ELSE statement, see the following example:

BEGIN

DECLARE @x INT = 10,

@y INT = 20;

IF (@x > 0)

BEGIN

IF (@x < @y)

PRINT 'x > 0 and x < y';

ELSE

PRINT 'x > 0 and x >= y';

END

END

Code language: SQL (Structured Query Language) (sql)

In this example:

First, declare two variables @x and @y and set their values to 10 and 20 respectively:

DECLARE @x INT = 10,

@y INT = 20;

Code language: SQL (Structured Query Language) (sql)

Second, the output IF statement check if @x is greater than zero. Because @x is set to 10, the condition (@x > 10) is true. Therefore, the nested IF statement executes.

Finally, the nested IF statement check if @x is less than @y ( @x < @y). Because @y is set to 20,  the condition (@x < @y) evaluates to true. The PRINT 'x > 0 and x < y'; statement in the IF branch executes.

Here is the output:

x > 0 and x < y

It is a good practice to not nest an IF statement inside another statement because it makes the code difficult to read and hard to maintain.