Operators

Arithmetic Operators

Following table shows all the arithmetic operators supported by PL/SQL. Let us assume **variable A** holds 10 and **variable B** holds 5, then −

[Show Examples](https://www.tutorialspoint.com/plsql/plsql_arithmetic_operators.htm)

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| + | Adds two operands | A + B will give 15 |
| - | Subtracts second operand from the first | A - B will give 5 |
| \* | Multiplies both operands | A \* B will give 50 |
| / | Divides numerator by de-numerator | A / B will give 2 |
| \*\* | Exponentiation operator, raises one operand to the power of other | A \*\* B will give 100000 |

Relational Operators

Relational operators compare two expressions or values and return a Boolean result. Following table shows all the relational operators supported by PL/SQL. Let us assume **variable A** holds 10 and **variable B** holds 20, then −

[Show Examples](https://www.tutorialspoint.com/plsql/plsql_relational_operators.htm)

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| = | Checks if the values of two operands are equal or not, if yes then condition becomes true. | (A = B) is not true. |
| !=  <>  ~= | Checks if the values of two operands are equal or not, if values are not equal then condition becomes true. | (A != B) is true. |
| > | Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true. | (A > B) is not true. |
| < | Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true. | (A < B) is true. |
| >= | Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true. | (A >= B) is not true. |
| <= | Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true. | (A <= B) is true |

Comparison Operators

Comparison operators are used for comparing one expression to another. The result is always either **TRUE, FALSE** or **NULL**.

[Show Examples](https://www.tutorialspoint.com/plsql/plsql_comparison_operators.htm)

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| LIKE | The LIKE operator compares a character, string, or CLOB value to a pattern and returns TRUE if the value matches the pattern and FALSE if it does not. | If 'Zara Ali' like 'Z% A\_i' returns a Boolean true, whereas, 'Nuha Ali' like 'Z% A\_i' returns a Boolean false. |
| BETWEEN | The BETWEEN operator tests whether a value lies in a specified range. x BETWEEN a AND b means that x >= a and x <= b. | If x = 10 then, x between 5 and 20 returns true, x between 5 and 10 returns true, but x between 11 and 20 returns false. |
| IN | The IN operator tests set membership. x IN (set) means that x is equal to any member of set. | If x = 'm' then, x in ('a', 'b', 'c') returns Boolean false but x in ('m', 'n', 'o') returns Boolean true. |
| IS NULL | The IS NULL operator returns the BOOLEAN value TRUE if its operand is NULL or FALSE if it is not NULL. Comparisons involving NULL values always yield NULL. | If x = 'm', then 'x is null' returns Boolean false. |

Logical Operators

Following table shows the Logical operators supported by PL/SQL. All these operators work on Boolean operands and produce Boolean results. Let us assume **variable A** holds true and **variable B** holds false, then −

[Show Examples](https://www.tutorialspoint.com/plsql/plsql_logical_operators.htm)

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Examples** |
| and | Called the logical AND operator. If both the operands are true then condition becomes true. | (A and B) is false. |
| or | Called the logical OR Operator. If any of the two operands is true then condition becomes true. | (A or B) is true. |
| not | Called the logical NOT Operator. Used to reverse the logical state of its operand. If a condition is true then Logical NOT operator will make it false. | not (A and B) is true. |

PL/SQL Operator Precedence

Operator precedence determines the grouping of terms in an expression. This affects how an expression is evaluated. Certain operators have higher precedence than others; for example, the multiplication operator has higher precedence than the addition operator.

For example, **x = 7 + 3 \* 2**; here, **x** is assigned **13**, not 20 because operator \* has higher precedence than +, so it first gets multiplied with **3\*2** and then adds into **7**.

Here, operators with the highest precedence appear at the top of the table, those with the lowest appear at the bottom. Within an expression, higher precedence operators will be evaluated first.

The precedence of operators goes as follows: =, <, >, <=, >=, <>, !=, ~=, ^=, IS NULL, LIKE, BETWEEN, IN.

[Show Examples](https://www.tutorialspoint.com/plsql/plsql_operators_precedence.htm)

|  |  |
| --- | --- |
| **Operator** | **Operation** |
| \*\* | exponentiation |
| +, - | identity, negation |
| \*, / | multiplication, division |
| +, -, || | addition, subtraction, concatenation |
| comparison |  |
| NOT | logical negation |
| AND | conjunction |
| OR | inclusion |