# Search Condition

```
{            [NOT] { predicate           }                  }
{                  { (search-condition)  }                  }
{ search-condition      { AND  }   search-condition          }
{                       { OR   }                             }
```

This chapter covers the following topics:

- Search Condition
- Predicate

## Search Condition

A *search-condition* can consist of a simple *predicate* or of multiple *search-conditions* combined with the Boolean operators AND, OR and NOT, and parentheses if required to indicate a desired order of evaluation.

### Example

```
DEFINE DATA LOCAL
01 NAME   (A20)
01 AGE    (I2)
END-DEFINE
...
SELECT *
  INTO NAME, AGE
  FROM SQL-PERSONNEL
  WHERE AGE = 32 AND NAME > 'K'
END-SELECT
...
```

## Predicate

```
{ scalar-expression comparison  { scalar-expression                                  }            }
{                               { subquery                                           }            }
{ scalar-expression [NOT] BETWEEN scalar-expression AND scalar-expression                         }
{ column-reference [NOT] LIKE   { atom                                               } [ESCAPE atom] }
{                               { special-register                                   }            }
{ column-reference IS [NOT] NULL                                                                  }
{ scale-expression [NOT] IN     { subquery                                           }            }
{                               { ( { atom             }                    )        }            }
{                               {   { special-register } , ...                       }            }
{ scalar-expression comparison  { ALL  }                       subquery                           }
                                { ANY  }
                                { SOME }
  EXISTS subquery
  XMLEXISTS (xquery-expression-constant     { BY REF  }    )
                                            { PASSING }
                                            xquery-argument,
                                            ...
```

A *predicate* specifies a condition that can be "true", "false" or "unknown".

In a *search-condition*, a *predicate* can consist of a simple or complex comparison operation or other kinds of conditions.

Example:

```
SELECT NAME, AGE
  INTO VIEW PERS
  FROM SQL-PERSONNEL
  WHERE AGE BETWEEN 20 AND 30
    0R AGE IN ( 32, 34, 36 )
    AND NAME LIKE ’%er’
    ...
```

**Note:**
The percent sign (%) may conflict with Natural terminal commands. If so, you must define a terminal command control character different from `%`.

The individual predicates are explained in the following topics (for further information on predicates, please refer to the relevant literature). According to the syntax above, they are called as follows:

- Comparison Predicate
- BETWEEN Predicate
- LIKE Predicate
- NULL Predicate

- IN Predicate
- Quantified Predicate
- EXISTS Predicate
- XMLEXISTS Predicate

## Comparison Predicate

```
scalar-expression  comparison  { scalar-expression }
                               { subquery          }
```

A comparison predicate compares two values.

See information on *`scalar-expression`*.

### Comparison

```
{  =  }
{  <  }
{  >  }
{  <= }
{  >= }
{  <> }
{ ¬ = }
{ ¬ > }
{ ¬ < }
```

*`comparison`* can be any of the following operators:

| | |
|---|---|
| = | equal to |
| < | less than |
| > | greater than |
| <= | less than or equal to |
| >= | greater than or equal to |
| <> | not equal to |
| ¬ = | not equal to |
| ¬ > | not greater than |
| ¬ < | not less than |

### Subquery

(*select-expression*)

A *subquery* is a *select-expression* that is nested inside another such expression.

Example:

```
DEFINE DATA LOCAL
1 #NAME    (A20)
1 #PERSNR  (I4)
END-DEFINE
...
SELECT NAME, PERSNR
  INTO #NAME, #PERSNR
  FROM SQL-PERSONNEL
  WHERE PERSNR IN
    ( SELECT PERSNR
        FROM SQL-AUTOMOBILES
        WHERE COLOR = 'black' )
        ...
END-SELECT
```

For further information, see *Select Expressions*.

## BETWEEN Predicate

*scalar-expression* [**NOT**] **BETWEEN** *scalar-expression* **AND** *scalar-expression*

A `BETWEEN` predicate compares a value with a range of values.

See information on *`scalar-expression`*.

## LIKE Predicate

*column-reference* [**NOT**] **LIKE** { *atom* | *special-register* } [**ESCAPE** *atom*]

A `LIKE` predicate searches for strings that have a certain pattern.

See information on *`column-reference`*, *`atom`* and *`special-register`*.

## NULL Predicate

*column-reference* **IS** [**NOT**] **NULL**

A `NULL` predicate tests for null values.

See information on *`column-reference`*.

## IN Predicate

$$\textit{scalar-expression}\ [\mathbf{NOT}]\ \mathbf{IN} \left\{ \begin{array}{l} \textit{subquery} \\ \mathbf{(} \left\{ \begin{array}{l} \textit{atom} \\ \textit{special-register} \end{array} \right\} \mathbf{, ...}\ \mathbf{)} \end{array} \right\}$$

An `IN` predicate compares a value with a collection of values.

If the compiler option `DB2ARRY` is set to `ON`, it is possible to specify a fixed Natural array or a fixed index range of an array as atom. The Natural SQL compiler will then decompose the array or fixed index range into a list of scalar host variables.

See information on *`scalar-expression`*, *`atom`* and *`special-register`*.

See information on *`subquery`*.

## Quantified Predicate

$$\textit{scalar-expression}\quad \textit{comparison} \left\{ \begin{array}{c} \mathbf{ALL} \\ \mathbf{ANY} \\ \mathbf{SOME} \end{array} \right\} \textit{subquery}$$

A quantified predicate compares a value with a collection of values.

See information on *`scalar-expression`*, *`comparison`*, and *`subquery`*.

## EXISTS Predicate

**`EXISTS`** *subquery*

An `EXISTS` predicate tests for the existence of certain rows.

The `EXISTS` predicate evaluates to true only if the result of evaluating the *`subquery`* is not empty; that is, if there exists at least one record (row) in the `FROM` table of the *`subquery`* satisfying the search condition of the `WHERE` clause of this *`subquery`*.

Example of `EXISTS`:

```
DEFINE DATA LOCAL
1 #NAME    (A20)
END-DEFINE
...
SELECT NAME
  INTO #NAME
  FROM SQL-PERSONNEL
  WHERE EXISTS
    ( SELECT *
      FROM SQL-EMPLOYEES
      WHERE PERSNR > 1000
        AND NAME < 'L' )
        ...
END-SELECT
...
```

See information on `subquery`.

## XMLEXISTS Predicate

**XMLEXISTS** (*xquery-expression-constant* [ **BY REF** | **PASSING** *xquery-argument*,... ] )

### *xquery-argument*

[ *xquery-context-item-expression* | *xquery-context-item-expression* **AS** *identifier* ]

The `XMLEXISTS` predicate tests whether an `XPATH` expression return a sequence of one or more items. For further details see the IBM *DB2 XML Guide*.