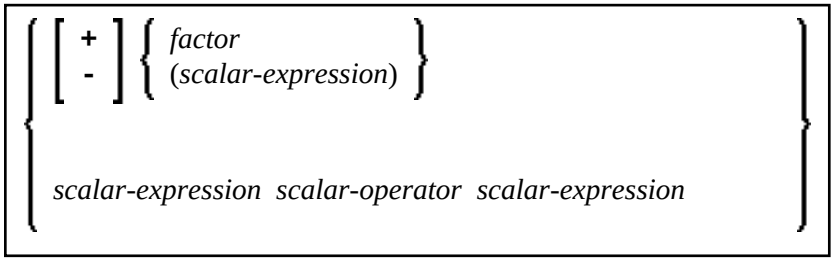
Scalar Expressions



This chapter covers the following topics:

- Scalar Expression
- Scalar Operator
- Factor

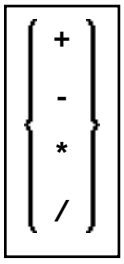
Scalar Expression

A *scalar-expression* consists of a factor or other scalar expressions including scalar operators.

Concerning reference priority, scalar expressions behave as follows:

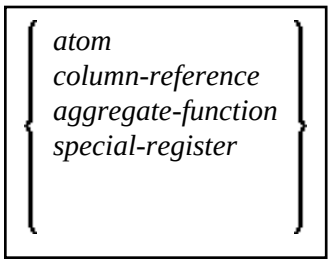
- When a non-qualified variable name is specified in a scalar expression, the first approach is to resolve the variable name as column name of the referenced table.
- If no column with the specified name is available in the referenced table, Natural tries to resolve this variable as a Natural user-defined variable (host variable).

Scalar Operator



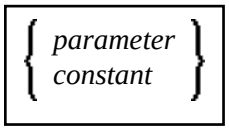
A *scalar-operator* can be any of the operators listed above; the minus (-) and slash (/) operators must be separated by at least one blank from preceding operators.

Factor



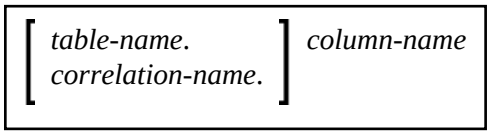
A *factor* can consist of one of the items listed in the above diagram and described in the text below.

Atom



An *atom* can be either a *parameter* or a *constant* (as described in the section *Basic Syntactical Items*).

Column Reference

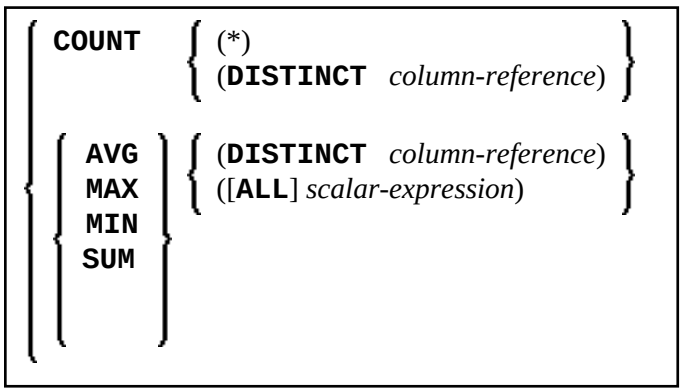


A *column-reference* is a column name optionally qualified by either a *table-name* or a *correlation-name* (see also the section *Basic Syntactical Items*). Qualified names are often clearer than unqualified names and sometimes they are essential.

Note:
A table name in this context must not be qualified explicitly with an authorization identifier. Use a correlation name instead if you need a qualified table name.

If a column is referenced by a *table-name* or *correlation-name*, it must be contained in the corresponding table. If neither a *table-name* nor a *correlation-name* is specified, the respective column must be in one of the tables specified in the FROM clause (see *Table Expression*).

Aggregate Function



SQL provides a number of special functions to enhance its basic retrieval power. The so-called SQL aggregate functions currently available and supported by Natural are:

AVG	gives the average of the values in a column
COUNT	gives the number of values in a column
MAX	gives the highest value in a column
MIN	gives the lowest value in a column
SUM	gives the sum of the values in a column

Apart from COUNT(*), each of these functions operates on the collection of scalar values in an argument (that is, a single column or a *scalar-expression*) and produces a scalar value as its result.

Example:

```
DEFINE DATA LOCAL
1 AVGAGE   (I2)
END-DEFINE
...
SELECT AVG (AGE)
  INTO AVGAGE
  FROM SQL-PERSONNEL
  ...
```

In general, the argument can optionally be preceded by the keyword DISTINCT to eliminate redundant duplicate values before the function is applied.

If DISTINCT is specified, the argument must be the name of a single column; if DISTINCT is omitted, the argument can consist of a general *scalar-expression*.

DISTINCT is not allowed with the special function COUNT(*), which is provided to count all rows without eliminating any duplicates.

Special Register

USER

A reference to a *special-register* returns a scalar value.