

LIMIT

LIMIT *n*

This chapter covers the following topics:

- Function
- Syntax Description
- Examples

Related Statements: ACCEPT/REJECT | AT BREAK | AT START OF DATA | AT END OF DATA | BACKOUT TRANSACTION | BEFORE BREAK PROCESSING | DELETE | END TRANSACTION | FIND | GET | GET SAME | GET TRANSACTION | HISTOGRAM | PASSW | PERFORM BREAK PROCESSING | READ | RETRY | STORE | UPDATE

Belongs to Function Group: *Database Access and Update*

Function

The **LIMIT** statement is used to limit the number of iterations of a processing loop initiated with a **FIND**, **READ**, or **HISTOGRAM** statement.

The limit remains in effect for all subsequent processing loops in the program until it is overridden by another **LIMIT** statement.

The **LIMIT** statement does not apply to individual statements in which a limit is explicitly specified (for example, **FIND** (*n*) . . .).

If the limit is reached, processing stops and a message is displayed; see also the session parameter **LE** which determines the reaction when the limit for the processing loop is exceeded.

If no **LIMIT** statement is specified, the default global limit defined with the Natural profile parameter **LT** during Natural installation will be used.

Record Counting

To determine whether a processing loop has reached the limit, each record read in the loop is counted against the limit. If the processing loop has reached the limit, the following will apply:

- A record that is rejected because of criteria specified in a **FIND** or **READ** statement **WHERE** clause is *not* counted against the limit.
- A record that is rejected as a result of an **ACCEPT/REJECT** statement is counted against the limit.

Syntax Description

Syntax Element	Description
LIMIT <i>n</i>	<p>Limit Specification:</p> <p>The limit <i>n</i> must be specified as a numeric constant in the range from 0 - 4294967295 (leading zeros are optional).</p> <p>The processing loop is not entered if the limit is set to zero.</p>

Examples

- Example 1 - LIMIT Statement
- Example 2 - LIMIT Statement (Valid for Two Database Loops)

Example 1 - LIMIT Statement

```

** Example 'LMTEX1': LIMIT
*****
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 NAME
  2 CITY
END-DEFINE
*
LIMIT 4
*
READ EMPLOY-VIEW BY NAME STARTING FROM 'BAKER'
  DISPLAY NOTITLE
    NAME PERSONNEL-ID CITY *COUNTER
END-READ
*
END

```

Output of Program LMTEX1:

```

          NAME          PERSONNEL          CITY          CNT
                   ID
-----
BAKER                20016700  OAK BROOK                1
BAKER                30008042  DERBY                    2
BALBIN               60000110  BARCELONA                3
BALL                 30021845  DERBY                    4

```

Example 2 - LIMIT Statement (Valid for Two Database Loops)

```

** Example 'LMTEX2': LIMIT (valid for two database loops)
*****
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 NAME
END-DEFINE

```

```

*
LIMIT 3
*
FIND EMPLOY-VIEW WITH NAME > 'A'
  READ EMPLOY-VIEW BY NAME STARTING FROM 'BAKER'
    DISPLAY NOTITLE 'CNT(0100)' *COUNTER(0100)
                  'CNT(0110)' *COUNTER(0110)
  END-READ
END-FIND
*
END

```

Output of Program LMTEX2:

CNT(0100)	CNT(0110)
1	1
1	2
1	3
2	1
2	2
2	3
3	1
3	2
3	3