# AT BREAK

### Structured Mode Syntax

```
[AT] BREAK [(r)] [OF] operand1 [/n/]
  statement ...
END-BREAK
```

### Reporting Mode Syntax

```
[AT] BREAK [(r)] [OF] operand1 [/n/]
  { statement                    }
  { DO statement... DOEND        }
```

This chapter covers the following topics:

- Function
- Syntax Description
- Multiple Break Levels
- Examples

For an explanation of the symbols used in the syntax diagram, see *Syntax Symbols*.

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

Belongs to Function Group: *Database Access and Update*

---

## Function

The AT BREAK statement is used to cause the execution of one or more statements whenever a change in value of a control field occurs. It is used in conjunction with automatic break processing and is available with the following statements: FIND, READ, HISTOGRAM, SORT, READ WORK FILE.

The automatic break processing works as follows: Immediately after a record was read by the processing loop, the control field is checked. If a value change is detected in comparison to the previous record, the statements included in the AT BREAK statement block are executed. This does not apply to the very first record in the processing loop. In addition, when the processing loop is terminated (as reading of records is complete or due to an ESCAPE BOTTOM statement), a final execution of the statements in the AT BREAK statement block is triggered.

For further information, see *Automatic Break Processing* in the *Programming Guide*.

An AT BREAK statement block is only executed if the object which contains the statement is active at the time when the break condition occurs.

It is possible to initiate a new processing loop within an AT BREAK condition. This loop must also be closed within the same AT BREAK condition.

This statement is non-procedural (that is, its execution depends on an event, not on where in a program it is located).

Natural system functions may be used in conjunction with an AT BREAK statement, see *Natural System Functions for Use in Processing Loops* in the *System Functions* documentation and *Example of System Functions with AT BREAK Statement* in the *Programming Guide*.

For further information, see also the section *AT BREAK Statement* in the *Programming Guide*. It covers topics such as:

- *Control Break Based on a Database Field*
- *Control Break Based on a User-Defined Variable*

# Syntax Description

Operand Definition Table:

| Operand | Possible Structure | | | | | Possible Formats | | | | | | | | | | | | Referencing Permitted | Dynamic Definition |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *operand1* | | S | | | | A | U | N | P | I | F | B | D | T | L | | | | yes | no |

Syntax Element Description:

| Syntax Element | Description |
|---|---|
| (*r*) | **Reference Notation:**<br>By default, the final AT BREAK condition (for loop termination) is always related to the outermost active processing loop initiated with a FIND, READ, READ WORK FILE, HISTOGRAM or SORT statement.<br><br>With the notation (*r*) you can relate the final break condition of an AT BREAK statement to another specific currently open processing loop (that is, the loop in which the AT BREAK statement is located or any outer loop).<br><br>**Example:**<br><br>`...`<br>`READ ...`<br>`  FIND ...`<br>`    FIND ...`<br>`      AT BREAK ...`<br>`        FIND ...`<br>`        END-FIND`<br>`      END-BREAK`<br>`    END-FIND`<br>`  END-FIND`<br>`END-READ`<br>`...`<br><br>In this example, the final AT BREAK condition is related to the READ loop initiated in line 0120. It would be possible to have it related to one of the FIND loops initiated in line 0130 and 0140, but not to the one initiated in line 0160.<br><br>If (*r*) is specified for a break hierarchy, it must be specified with the first AT BREAK statement and applies also to all AT BREAK statements which follow. |
| *operand1* | **Control Field:**<br>The field used as the break control field is usually a database field. If a user-defined variable is used, it must be initialized prior to the evaluation of automatic break processing (see BEFORE BREAK PROCESSING statement). A specific occurrence of an array can also be used as a control field. |
| /*n*/ | **Notation /*n*/:**<br>The notation /*n*/ may be used to indicate that only the first *n* positions (counting from left to right) of the control field are to be checked for a change in value. This notation can only be used with operands of format A, B, N or P.<br><br>A control break occurs when the value of the control field changes, or when all records in the processing loop for which the AT BREAK statement applies have been processed. |
| *statement* | **Statement(s) to be Executed at Break Condition:**<br>In place of *statement*, you must supply one or several suitable statements, depending on the situation. For an example of a statement, see *Examples* below. |
| END-BREAK | **End of AT BREAK Statement:**<br>The Natural reserved word END-BREAK must be used to end the AT BREAK statement. |

# Multiple Break Levels

Multiple AT BREAK statements may be specified within a processing loop within the same program module. If multiple BREAK statements are specified for the same processing loop, they form a hierarchy of break levels independent of whether they are specified consecutively or interspersed within other statements. The first AT BREAK statement represents the lowest control break level, and each additional AT BREAK statement represents the next higher control break level.

Every processing loop in a loop hierarchy may have its own break hierarchy attached.

**Example:**

| Structured Mode: | Reporting Mode: |
|---|---|
| `FIND ...`<br>`  AT BREAK`<br>`  ...`<br>`  END-BREAK`<br>`  AT BREAK`<br>`  ...`<br>`  END-BREAK`<br>`  AT BREAK`<br>`  ...`<br>`  END-BREAK`<br>`END-FIND`<br>`...` | `FIND ...`<br>`  AT BREAK`<br>`  DO`<br>`  ...`<br>`  DOEND`<br>`  AT BREAK`<br>`  DO`<br>`  ...`<br>`  DOEND`<br>`...` |

A change in the value of a control field in a break level causes break processing to be activated for that break level and all lower break levels, regardless of the values of the control fields for the lower break levels.

For easier program maintenance, it is recommended to specify multiple breaks consecutively.

See also *Example 3* below and the section *Multiple Control Break Levels* in the *Programming Guide*.

# Examples

This section covers the following topics:

- Example 1 - AT BREAK
- Example 2 - AT BREAK Using /n/ Notation
- Example 3 - AT BREAK with Multiple Break Levels

For further examples of AT BREAK, see *Natural System Functions for Use in Processing Loops*, Examples ATBEX3 and ATBEX4.

## Example 1 - AT BREAK

```
** Example 'ATBEX1S': AT BREAK (structured mode)
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 CITY
```

```
  2 COUNTRY
  2 NAME
END-DEFINE
*
LIMIT 10
READ EMPLOY-VIEW BY CITY
AT BREAK OF CITY
    SKIP 1
  END-BREAK
  DISPLAY NOTITLE CITY (IS=ON) COUNTRY (IS=ON) NAME
END-READ
*
END
```

### Output of Program ATBEX1S:

```
        CITY         COUNTRY         NAME
-------------------- ------- --------------------

AIKEN                USA     SENKO

AIX EN OTHE          F       GODEFROY

AJACCIO                      CANALE

ALBERTSLUND          DK      PLOUG

ALBUQUERQUE          USA     HAMMOND
                             ROLLING
                             FREEMAN
                             LINCOLN

ALFRETON             UK      GOLDBERG

ALICANTE             E       GOMEZ
```

Equivalent reporting-mode example: ATBEX1R.

## Example 2 - AT BREAK Using /*n*/ Notation

```
** Example 'ATBEX2': AT BREAK (with /n/ notation)
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 DEPT
  2 NAME
END-DEFINE
*
LIMIT 10
READ EMPLOY-VIEW BY DEPT STARTING FROM 'A'
AT BREAK OF DEPT /4/
    SKIP 1
  END-BREAK
  DISPLAY NOTITLE DEPT NAME
END-READ
*
END
```

### Output of Program ATBEX2:

```
DEPARTMENT         NAME
   CODE
---------- --------------------

ADMA01     JENSEN
ADMA01     PETERSEN
ADMA01     MORTENSEN
ADMA01     MADSEN
ADMA01     BUHL
ADMA02     HERMANSEN
ADMA02     PLOUG
ADMA02     HANSEN

COMP01     HEURTEBISE
COMP01     TANCHOU
```

## Example 3 - AT BREAK with Multiple Break Levels

```
** Example 'ATBEX5S': AT BREAK (multiple break levels) (structured mode)
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 CITY
  2 DEPT
  2 NAME
  2 LEAVE-DUE
1 #LEAVE-DUE-L (N4)
END-DEFINE
*
LIMIT 5
FIND EMPLOY-VIEW WITH CITY = 'PHILADELPHIA' OR = 'PITTSBURGH'
                 SORTED BY CITY DEPT
  MOVE LEAVE-DUE TO #LEAVE-DUE-L
  DISPLAY CITY (IS=ON) DEPT (IS=ON) NAME #LEAVE-DUE-L
  /*
 AT BREAK OF DEPT
    WRITE NOTITLE /
          T*DEPT OLD(DEPT) T*#LEAVE-DUE-L SUM(#LEAVE-DUE-L) /
  END-BREAK
  AT BREAK OF CITY
    WRITE NOTITLE
          T*CITY OLD(CITY) T*#LEAVE-DUE-L SUM(#LEAVE-DUE-L) //
  END-BREAK
END-FIND
*
END
```

### Output of Program ATBEX5:

```
        CITY         DEPARTMENT         NAME         #LEAVE-DUE-L
                        CODE
-------------------- ---------- -------------------- ------------

PHILADELPHIA         MGMT30     WOLF-TERROINE              11
                                MACKARNESS                 27

                     MGMT30                                38

                     TECH10     BUSH                       39
                                NETTLEFOLDS                24
```

```
                    TECH10                          63

PHILADELPHIA                                       101

PITTSBURGH          MGMT10     FLETCHER             34

                    MGMT10                          34

PITTSBURGH                                          34
```

Equivalent reporting-mode example: ATBEX5R.