# AT END OF PAGE

### Structured Mode Syntax

```
[AT] END [OF] PAGE [(rep)]
  statement ...
END-ENDPAGE
```

### Reporting Mode Syntax

```
[AT] END [OF] PAGE [(rep)]
{ statement                  }
{ DO statement ... DOEND     }
```

This chapter covers the following topics:

- Function
- Syntax Description
- Example

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

Related Statements: AT TOP OF PAGE | CLOSE PRINTER | DEFINE PRINTER | DISPLAY | EJECT | FORMAT | NEWPAGE | PRINT | SKIP | SUSPEND IDENTICAL SUPPRESS | WRITE | WRITE TITLE | WRITE TRAILER

Belongs to Function Group: *Creation of Output Reports*

---

## Function

The `AT END OF PAGE` statement is used to specify processing that is to be performed when an end-of-page condition is detected (see the session parameter `PS` in the *Parameter Reference*). An end-of-page condition may also occur as a result of a `SKIP` or `NEWPAGE` statement, but not as a result of an `EJECT` or `INPUT` statement.

See also the following sections in the *Programming Guide*:

- *Controlling Data Output*
- *Report Specification - (rep) Notation*
- *Layout of an Output Page*

- *AT END OF PAGE Statement*

## Processing

An `AT END OF PAGE` statement block is only executed if the object which contains the statement block is active at the time when the end-of-page condition occurs.

An `AT END OF PAGE` statement must not be placed within an inline subroutine.

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

## Logical Page Size

The end-of-page check is performed after the processing of a `DISPLAY` or `WRITE` statement is completed. Therefore, if a `DISPLAY` or `WRITE` statement produces multiple lines of output, overflow of the physical page may occur before an end-of-page condition is detected.

A logical page size (session parameter `PS`) which is less than the physical page size must be specified to ensure that information printed by an `AT END OF PAGE` statement appears on the same physical page as the title.

## Last-Page Handling

Within a main program, an end-of-page condition is activated when the execution of the main program terminates via `ESCAPE`, `STOP` or `END`.

Within a subroutine, an end-of-page condition is not activated when the execution of the subroutine terminates via `ESCAPE-ROUTINE`, `RETURN` or `END-SUBROUTINE`.

## System Functions

Natural system functions may be used in conjunction with an `AT END OF PAGE` statement as described in the section *Using System Functions in Processing Loops* in the *System Functions* documentation.

If a system function is to be used within an `AT END OF PAGE` statement block, the `GIVE SYSTEM FUNCTIONS` clause must be specified in the corresponding `DISPLAY` statement.

## INPUT Statement with AT END OF PAGE

If an `INPUT` statement is specified within an `AT END OF PAGE` statement block, no new page operation is performed. The page size (session parameter `PS`) must be reduced to a value that allows the lines created by the `INPUT` statement to appear on the same physical page.

See also:

- *Split Screen Feature* of `INPUT` Statement
- *Example 2 - AT END OF PAGE with INPUT Statement*

# Syntax Description

| | |
|---|---|
| **(*rep*)** | **Report Specification:**<br>The notation `(rep)` may be used to specify the identification of the report for which the `AT END OF PAGE` statement is applicable. A value in the range 0 - 31 or a logical name which has been assigned using the DEFINE PRINTER statement may be specified.<br><br>If `(rep)` is not specified, the `AT END OF PAGE` statement will apply to the first report (Report 0).<br><br>For information on how to control the format of an output report created with Natural, see *Controlling Data Output* in the *Programming Guide*. |
| **END-ENDPAGE** | The Natural reserved word `END-ENDPAGE` must be used to end the `AT END OF PAGE` statement. |

# Example

- Example 1 - AT END OF PAGE
- Example 2 - AT END OF PAGE with INPUT Statement

## Example 1 - AT END OF PAGE

```
** Example 'AEPEX1S': AT END OF PAGE (structured mode)
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 NAME
  2 JOB-TITLE
  2 SALARY    (1)
  2 CURR-CODE (1)
END-DEFINE
*
FORMAT PS=10
LIMIT 10
READ EMPLOY-VIEW BY PERSONNEL-ID FROM '20017000'
  DISPLAY NOTITLE GIVE SYSTEM FUNCTIONS
          NAME JOB-TITLE 'SALARY' SALARY(1) CURR-CODE (1)
  /*

 AT END OF PAGE
    WRITE / 28T 'AVERAGE SALARY: ...' AVER(SALARY(1)) CURR-CODE (1)
  END-ENDPAGE


END-READ
*
END
```

See also *Natural System Functions for Use in Processing Loops*.

### Output of Program AEPEX1S:

```
        NAME                 CURRENT            SALARY   CURRENCY
                            POSITION                       CODE
------------------- ----------------------- ---------- --------

CREMER              ANALYST                       34000 USD
MARKUSH             TRAINEE                       22000 USD
GEE                 MANAGER                       39500 USD
KUNEY               DBA                           40200 USD
NEEDHAM             PROGRAMMER                    32500 USD
JACKSON             PROGRAMMER                    33000 USD

                          AVERAGE SALARY: ...     33533 USD
```

Equivalent reporting-mode example: AEPEX1R.

## Example 2 - AT END OF PAGE with INPUT Statement

```
** Example 'AEPEX2': AT END OF PAGE (with INPUT)
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 NAME
  2 FIRST-NAME
  2 POST-CODE
  2 CITY
*
1 #START-NAME (A20)
END-DEFINE
*
FORMAT PS=21
*
REPEAT
  READ (15) EMPLOY-VIEW BY NAME = #START-NAME
    DISPLAY NOTITLE NAME FIRST-NAME POST-CODE CITY
  END-READ
  NEWPAGE
  /*
 AT END OF PAGE
    MOVE NAME TO #START-NAME
    INPUT / '-' (79)
          / 10T 'Reposition to name ==>'
                #START-NAME (AD=MI) '(''.'' to exit)'
    IF #START-NAME = '.'
      STOP
    END-IF
  END-ENDPAGE
  /*
END-REPEAT
END
```

### Output of Program AEPEX2S:

```
        NAME             FIRST-NAME        POSTAL           CITY
                                          ADDRESS
------------------- -------------------- ---------- --------------------

ABELLAN             KEPA                 28014      MADRID
ACHIESON            ROBERT               DE3 4TR    DERBY
ADAM                SIMONE               89300      JOIGNY
```

```
ADKINSON            JEFF                11201      BROOKLYN
ADKINSON            PHYLLIS             90211      BEVERLEY HILLS
ADKINSON            HAZEL               20760      GAITHERSBURG
ADKINSON            DAVID               27514      CHAPEL HILL
ADKINSON            CHARLIE             21730      LEXINGTON
ADKINSON            MARTHA              17010      FRAMINGHAM
ADKINSON            TIMMIE              17300      BEDFORD
ADKINSON            BOB                 66044      LAWRENCE
AECKERLE            SUSANNE             7000       STUTTGART
AFANASSIEV          PHILIP              39401      HATTIESBURG
AFANASSIEV          ROSE                60201      EVANSTON
AHL                 FLEMMING            2300       SUNDBY

--------------------------------------------------------------------------------
         Reposition to name ==> AHL                ('.' to exit)
```