END TRANSACTION

END [**OF**] **TRANSACTION** [*operand1* **...**]

This chapter covers the following topics:

- Function
- Restriction
- Syntax Description
- Databases Affected
- Database-Specific Considerations
- Examples

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

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

Belongs to Function Group: *Database Access and Update*

Function

The END TRANSACTION statement is used to indicate the end of a logical transaction. A logical transaction is the smallest logical unit of work (as defined by the user) which must be performed in its entirety to ensure that the information contained in the database is logically consistent.

Successful execution of an END TRANSACTION statement ensures that all updates performed during the transaction have been or will be physically applied to the database regardless of subsequent user, Natural, database or operating system interruption. Updates performed within a transaction for which the END TRANSACTION statement has not been successfully completed will be backed out automatically.

The END TRANSACTION statement also results in the release of all records placed in hold status during the transaction.

The END TRANSACTION statement can be executed based upon a logical condition.

For further information, see the section *Database Update - Transaction Processing* (in the *Programming Guide*).

Restriction

This statement cannot be used with Entire System Server.

Syntax Description

Operand Definition Table:

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

Syntax Element Description:

Syntax Element	Description
operand1	**Storage of Transaction Data:** For a transaction applied to an Adabas database, you may also use this statement to store transaction-related information. These transaction data must not exceed 2000 bytes. They may be read with a `GET TRANSACTION DATA` statement. The transaction data are written to the database specified with the profile parameter `ETDB`. If the `ETDB` parameter is not specified, the transaction data are written to the database specified with the profile parameter `UDB`, except on mainframe computers: here, they are written to the database where the Natural Security system file (`FSEC`) is located (if `FSEC` is not specified, it is considered to be identical to the Natural system file, `FNAT`; if Natural Security is not installed, the transaction data are written to the database where `FNAT` is located).

Databases Affected

An `END TRANSACTION` statement *without* transaction data (that is, without *operand1*) will only be executed if a database transaction under control of Natural has taken place. Depending on the setting of the Natural profile parameter `ET`, the statement will be executed only for the database affected by the transaction (`ET=OFF`), or for all databases that have been referenced since the last execution of a `BACKOUT TRANSACTION` or `END TRANSACTION` statement (`ET=ON`).

An `END TRANSACTION` statement *with* transaction data (that is, with specifying *operand1*) will always be executed and the transaction data be stored in a database as described in the following section. It depends on the setting of the `ET` parameter (see above) for which other databases the `END TRANSACTION` statement will be executed.

Database-Specific Considerations

SQL Databases	As most SQL databases close all cursors when a logical unit of work ends, an `END TRANSACTION` statement must not be placed within a database modification loop; instead, it has to be placed after such a loop.
XML Databases	An `END TRANSACTION` statement must not be placed within a database modification loop; instead, it has to be placed after such a loop.

Examples

- Example 1 - END TRANSACTION
- Example 2 - END TRANSACTION with ET Data

Example 1 - END TRANSACTION

```
** Example 'ETREX1': END TRANSACTION
**
** CAUTION: Executing this example will modify the database records!
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 CITY
  2 COUNTRY
END-DEFINE
*
FIND EMPLOY-VIEW WITH CITY = 'BOSTON'
  ASSIGN COUNTRY = 'USA'
  UPDATE
  END TRANSACTION
  /*
  AT END OF DATA
    WRITE NOTITLE *NUMBER 'RECORDS UPDATED'
  END-ENDDATA
  /*
END-FIND
END
```

Output of Program ETREX1:

```
7 RECORDS UPDATED
```

Example 2 - END TRANSACTION with ET Data

```
** Example 'ETREX2': END TRANSACTION (with ET data)
**
** CAUTION: Executing this example will modify the database records!
************************************************************************
DEFINE DATA LOCAL
1 EMPLOY-VIEW VIEW OF EMPLOYEES
  2 PERSONNEL-ID
  2 NAME
  2 FIRST-NAME
  2 CITY
*
1 #PERS-NR (A8) INIT <' '>
```

```
END-DEFINE
*
REPEAT
  INPUT 'ENTER PERSONNEL NUMBER TO BE UPDATED:' #PERS-NR
  IF #PERS-NR = ' '
    ESCAPE BOTTOM
  END-IF
  /*
  FIND EMPLOY-VIEW PERSONNEL-ID = #PERS-NR
    INPUT (AD=M)    NAME / FIRST-NAME / CITY
    UPDATE
    END TRANSACTION #PERS-NR
  END-FIND
  /*
END-REPEAT
END
```

Output of Program ETREX2:

```
ENTER PERSONNEL NUMBER TO BE UPDATED: 20027800
```

After entering and confirming the personnel number:

```
NAME LAWLER
FIRST-NAME SUNNY
CITY MILWAUKEE
```