## DBPUT

### INTRINSIC NUMBER 407

Adds new entries to a manual master or detail data set. The database must be opened in access mode 1, 3, or 4.

**OPENTURBO vs TurboIMAGE Difference**

100% - Except the path info is not updated, OpenTURBO doesn't not automatically adjust the current path linkage after the addition for continuing Mode 5 and 6 DBGET

**OPENTURBO Performance Enhancements**

Elimination of AUTOMATIC Dataset.

**OPENTURBO Additional Features**

N/A.

DBPUT is mapped to ORACLE INSERT:

1. You may only insert Manual and Detail tables.
2. The Automatic table is automatically maintained by OPENTURBO.
3. You may not insert Detail row that has no corresponding key in Manual table, which is controlled by PRIMARY and FOREIGN constraint.
4. All unique sequence number is maintained by OPENTURBO in chronological order automatically, which is managed by trigger, sequence, and index constraint.
5. All Detail paths are also maintained by OPENTURBO automatically via trigger, sequence and index constraint.

```
DBPUT:

    I1 and I2 -> 32-bit integer
    I4        -> 64-bit integer
    I8        ->

    J
    K
    K3, K5-255-> RAW

    E and R

    P
    U
    X
    Z         -> Last digit
              {, A-I => 0-9 Positive
              }, J-R => 0-9 Negative
```

**Syntax**

> DBPUT, base, dset, mode, status, list, buffer

**Parameters**

`base`             is the name of the array used as the `base` parameter when opening the database. The first element of the array must contain the base ID returned by DBOPEN. (Refer to `DBOPEN` for more information about base ID.)

`dset`             is the name of an array containing the left-justified name of the data set to which the entry is to be added, or is an integer referencing the data set by number. The data set name can be up to 16 characters long. If shorter, it must be terminated by a semicolon or a blank.

`mode`             must be an integer equal to 1.

> If your database is enabled for third-party indexing (TPI), refer to your vendor documentation for additional `DBPUT` mode information. The section on `DBUTIL` in chapter 8 of this book has a brief description of the TPI option.

`status` is the name of an array of 10 halfwords in which TurboIMAGE/XL returns status information about the procedure. If the procedure executes successfully, the status array contents are:

| Element | Contents |
|---------|----------|
| 1 | If the procedure succeeds, the return status is 0. Table 5-21. describes the contents of element 1 when the procedure does not succeed. |
| 2 | Length of logical entry in buffer array (in halfwords). |
| 3-4 | Word record number of new entry. |
| 5-6 | Word count of number of entries in chain. If master data set, chain is synonym chain. If detail data set, chain is current chain of new entry. |
| 7-8 | If master, word record address of predecessor on synonym chain. If detail, word record number of predecessor on current detail chain. |
| 9-10 | If detail, word record number of successor on current chain. If master, word zero. |

`list`             is the name of an array containing an ordered set of data item identifiers; names or numbers. The new entry contains values supplied in the buffer array for data items in the *list* array. Search or sort items defined for the entry must be included in the *list* array. Fields of unreferenced items are filled with binary zeros.

> The `list` array can contain a left-justified set of data item names, separated by commas and terminated by a semicolon or blank; no embedded blanks are allowed, and no name can appear more than once. For example:

ACCOUNT,LAST-NAME,CITY,STATE;

When referencing by number, the first halfword of the `list` array is an integer `n` that is followed by `n` single positive integers identifying unique data item numbers. Example: 4 1 10 3 16 lists for the four data item numbers 1, 10, 3, and 16.

The `list` specifies data items for which values are supplied in the buffer array, and is saved internally by TurboIMAGE/XL as the current list for the data set. The current list is unchanged until a different list is specified in a subsequent call to `DBGET,` `DBPUT`, or `DBUPDATE` for the same access path and data set.

Some special list constructs are allowed. These are described in Table 5-20. and illustrated in the programs in chapter 6. List processing is a relatively high overhead operation which can be shortened in subsequent calls by using the asterisk construct to specify that the current list is to be used. Be sure a current list exists before using the asterisk construct, or a null list is assumed.

`buffer` is the name of an array containing data item values to be added. The values must be in the same order as their data item identifiers in the list array. The number of halfwords for each value must correspond to the number required by its type; for example, I2 values must be 2 halfwords long.

## Table 5-20. Special `list` Parameter Constructs

| Construct | *list* Array Contents | Purpose |
|---|---|---|
| Empty | 0; or 0⊔ or ; or ⊔ (0 must be ASCII.) | Request no data transfer. |
| Empty Numeric | 0 (*n*, length of data item identifier list, is zero) | Request no data transfer. |
| Asterisk | *; or *⊔ | Request procedure to use previous *list* and apply it to same data set. This construct saves TurboIMAGE/XL processing time, especially if more than one or two items are involved. If "*" is used to define the list in the first call to `DBGET` and `DBPUT`, TurboIMAGE/XL treats it as a zero. |

| Construct | *list* Array Contents | Purpose |
|---|---|---|
| Commercial At-Sign | @; or @⊔ | Request procedure to use all data items of the data set in the order of their occurrence in the entry. |
| **NOTE:** | ⊔ indicates blank | |

## Discussion for Master Data Sets

When adding entries to master data sets, the following rules apply:

- The data set must be a manual master.

- The key item must be referenced in the list array and its value in the buffer array must be unique in relation to other entries in the master.

- Space must be available in the master set, or must be dynamically expandable to add an entry.

- If dynamic capacity expansion parameters are specified for the master data, when the master data set is almost full, expansion is done by the incremental amount. If the expansion is successful, the new record is added to the database. If the expansion is not successful, an error message is displayed, and the record is not added to the database. If there is insufficient disc space to expand the data set to the *full* incremental amount, DBPUT will perform a *partial* expansion up to the disc space available. DBPUT will terminate if there is no available group or account disc space even if there is enough system disc space. (The current capacity for a data set can be displayed by the SHOW CAPACITY command in DBUTIL or the output buffer from DBINFO modes 202 and 205.)

- The order of data item values in the new entry is determined by the set definition in the schema and not by the order of the items' occurrence in the list and buffer arrays.

- Values for data items not included in the list array are filled with binary zeros.

- The caller must have a lock on the data set or the database if the database is open in access mode 1.

- DBPUT to an indexed master triggers a similar operation to indexed master's B-Tree index file.

## Discussion for Detail Data Sets

When adding entries to detail data sets, the following rules apply:

- The data set must have free space for the entry.

- If the database is opened in access mode 1, the caller must have a lock covering the entry to be added.

- All search and sort items defined for the entry must be referenced in the list array.

- Each related manual master data set must contain a matching entry for the corresponding search item value. If any automatic master does not have a matching entry, it must have space to add one. This addition occurs automatically.

- The order of data item values in the new entry is determined by the set definition in the schema and not by the order of the items' occurrence in the list and buffer arrays.

- Values for data items not included in the list array are filled with binary zeros.

- The new entry is linked into one chain for each search item, or path, defined according to the search item value. It is linked to the end of chains having no sort items and into its sorted position according to the collating sequence of the sort item values in the chain. If two or more entries have the same sort item value, their position in the chain is determined by the values of the items following the sort item in the entry. The position of an entry on a sorted chain is determined by a backward search of the chain beginning at the last entry. The position is maintained by logical pointers rather than physical placement in the file.

- Proper Native Language collating sequence must be maintained for chain sorting.

- If dynamic capacity expansion is allowed for the detail data set, when the detail data set reaches the end of the current allocation (that is, data set free count is zero), expansion is requested by the incremental amount. If the expansion is successful, the new record is added to the database. If the expansion is not successful, an error message is displayed, and the record is not added to the database. If there is insufficient disc space to expand the data set to the *full* incremental amount, DBPUT will perform a *partial* expansion up to the disc space available. DBPUT will terminate if there is no available group or account disc space even if there is enough system disc space. (The current capacity for a data set can be displayed by the SHOW CAPACITY command in DBUTIL or the output buffer from DBINFO modes 202 and 205.) The record in which the new data entry is placed becomes the current record for the data set. The forward and backward pointers reflect the new entry's position. Refer to the description of status elements 7 through 10.

The record number of the new data entry is returned to status halfwords 3-4; and its forward and backward pointers are returned in status halfwords 7-8 and 9-10, respectively. If you intend to use these numbers for directed reads (see "Directed Access" in chapter 4), save them because subsequent TurboIMAGE/XL procedure calls can overwrite the status area.

The execution of a call to DBPUT could require extensive resources depending on the amount of chain maintenance required. For example, when an entry is added to a detail data set, the new entry must be linked to all other related entries with the same key values and to all of its related master entries. This operation could involve many blocks of data. TurboIMAGE/XL prevents data block access conflicts with all other users and ensures data integrity by applying a temporary lock on other processes until the call to DBPUT completes. The timing of this temporary lock can be controlled with the PREFETCH option of DBUTIL. Refer to "Coordinating Additions to a Database" in chapter 4 for considerations when enabling or disabling this option.

Performance may be impacted by the number of entries incremented when DBPUT is used to dynamically expand the detail data set. The number of disc extents used for the data set file may also impact the performance of TurboIMAGE/XL.

If the process is logging, a call to DBPUT causes a log record to be written with such information as the time, date, user identification number, and a copy of the new record to be added.

If DBPUT is called within a dynamic transaction, a log record is written after the physical transaction has been successfully completed. If the intrinsic cannot be completed, an error is returned. This error condition must be checked, and you must decide to use DBXUNDO, DBXEND, or continue with the

remainder of the dynamic transaction. DBXUNDO  will abort the entire dynamic transaction. DBXEND will terminate the dynamic transaction; the modifications completed thus far within the transaction will remain in the database.

## Discussion for OPENTURBO Specific

DBPUT:
I1 and I2          → 32-bit integer
I4                 → 64-bit integer
I8                 →

J
K
K3, K5-K25         → RAW

E and R

P
U
X
Z                  → Last digit
                   {, A-I => 0-9 Positive
                   }, J-R => 0-9 Negative

## Table 5-21. DBPUT Return Status Values

| File System, Memory Management, and Transaction Management Failures: | -1 | FOPEN failure. |
| --- | --- | --- |
| | -3 | FREADDIR failure. |
| | -4 | FREADLABEL failure. |
| | -5 | FWRITEDIR failure. |
| | -6 | FWRITELABEL failure. |
| | -167 | Cannot begin MPE XL XM transaction: XM error *nn*. |
| | -168 | Cannot attach *n* to MPE XL XM: file system error *nn*. |
| | -169 | Invalid mode for XM attach options. |
| | -175 | Cannot attach *n* to MPE XL XM: XM error *nn*. |
| | -176 | Cannot detach *n* from MPE XL XM: XM error *nn*. |
| | -178 | Cannot detach *n* from MPE XL XM: file system error *nn*. |
| | -199 | Cannot end MPE XL XM transaction: XM error *nn*. |
| | -209 | Invalid mode for XM detach options. |
| | -1025 | The XM transaction size (28 MB) limit has been reached, only DBXEND or DBXUNDO are allowe. |

| Calling Errors: | -11 | Bad database reference. |
|---|---|---|
| | -12 | No lock covers the data entry to be added. (Occurs only if database is open in access mode 1.) |
| | -14 | Illegal intrinsic in current access mode. |
| | -21 | Bad data set reference. |
| | -23 | Data set not writable. |
| | -24 | Operation not allowed on automatic master data set. |
| | -31 | Bad mode. |
| | -51 | Bad list length. |
| | -52 | Bad list or bad item. |
| | -53 | Missing search or sort item. |
| | -222 | Only DBXUNDO allowed when a dynamic transaction encounters an error. |
| Communications Errors: | -102 | DSWRITE failure. |
| | -106 | Remote 3000 data inconsistent. |
| | -107 | NS 3000 or DS 3000 system error. |
| Logging System Failures: | -111 | WRITELOG failure. |

| Exceptional Conditions: | -193 | DBU control block is full. |
| --- | --- | --- |
| | -196 | DBB control block is full. |
| | -212 | Database corruption detected. |
| | -264 | XM write procedure returned 1030 or 1040. |
| | -312 | Internal error encountered while reading database block. |
| | -314 | Error while obtaining path information for set. |
| | -322 | Error returned by process list. |
| | -332 | Error in QLOCK table operation. |
| | 16 | Data set full. |
| | | (In the following messages, *dataset#* and FSERR # will be replaced by the actual number.) |
| | | DBPUT cannot expand *dataset#*: dataset at maximum capacity. |
| | | DBPUT *dataset#* incomplete expansion: File system error #. |
| | | DBPUT cannot expand *dataset#*: Out of disc space (FSERR #). |
| | 18 | Broken chain; forward and backward pointers not consistent. |
| | 43 | Duplicate key item value. |
| | 62 | DBG control block is full. |
| | 63 | DBG disabled; potential damage; only DBCLOSE allowed. |
| | 1*nn* | Missing chain head for path number *nn*. |
| | 2*nn* | Full chain for path number *nn*. |
| | 3*nn* | Internal error. |
| | 4*nn* | Full synonym chain. |

Refer to Appendix A for more information about these conditions.