tableBASE

Release Notes

Release 6.0.2

Document Number: TBM011-R6.0.2v1.0

Publication History

Maintenance Release 6.0.1 - May 2004

Service Pack 2 R6.0.1v1.1 - November 2004

tableBASE Release 6.0.2v1.0 - May 2005

Technical Support Hotline: (613) 523-5588

Data Kinetics Ltd.
202 - 2460 Lancaster Road
Ottawa, ON
Canada K1B 4S5

Telephone: 613-523-5500
1-800-267-0730 (toll free in the US and Canada)

Facsimile: (613) 523-5533

Email: tableBASE@dkl.com

http://www.dkl.com

Table of Contents

6 - tablesONLINE 27

7 - Notes and Cautions 29

8 - Documentation Updates 35

Preface

The Release Notes provide a high level view of the tableBASE Version 6 which includes Release 6.0, 6.0.1 and 6.0.2.

Conventions Used in this Guide

This guide uses conventions to differentiate code and typed commands, and to display the names of parameters.

Convention	Description
`code examples and commands`	Appear in a fixed font.
MAXNMTAB	Names of parameters appear in upper case simply for ease of reading; actual case used is upper or lower or a mixture.
Version	Following the IBM standards, version refers to a program that has significant new code or new functionality. Version is a more general term than release. For example, Version 6 includes Release 6.1 and 6.2, and is equivalent to Release 6.x.
Release	Following the IBM standards, release refers to a specific program. For example, Release 6.0 is a term that is used to identify the first release of Version 6, and does not necessarily include other software releases published under Version 6, such as Release 6.1.

The term VTS-TSR is introduced with this version of tableBASE. Tables in tableBASE are either opened for private use in a local TSR, or, for shared use in a VTS-TSR. The latter is an optional component provided with the installation of the VTS Agent. The term TSR when not qualified refers to either a local or a VTS-TSR. VTS is an optional component of tableBASE.

Glossary

The terms defined in the glossary will help you better understand the content of this document.

Data Table	The raw data. Each Data Table has a table definition (DT-BLOCK) that is used to generate the Index for the Data Table. Also referred to as a base or primary table.
Index	An Index is defined for each Data Table. A Data Table Index is generated dynamically when a table is opened based on the information in the table definition (DT-BLOCK).
Alternate Index	An Alternate Index is an Index that may be defined for a Data Table. The Alternate Index has an Alternate Index definition (ALT-DEFINITION) that defines the key, organization, and search order. Each Data Table may have as many Alternate Indexes as desired or none.
Delivered defaults	Refer to the defaults that are delivered with the product.
Installation defaults	Refers to the defaults set at installation time, which may or may not be the same as the delivered defaults.

In addition, tablesONLINE/CICS uses these terms:

View	A View provides the field, edit, and display attributes for a Data Table with its Index. Referred to in previous releases as a Field Definition Table (FDT).
Alternate Index View	An Alternate Index View is identical to a View but applies to a Data Table with its Alternate Index.

1 Introduction

tableBASE Release 6.0.2 provides enhancements, fixes of known issues and support for the Library Bridge product (tableBASE 5.B).

Version 6 offers significant enhancements and improvements, such as multi-tasking capabilities, that also require additional consideration during implementation. A section is included in the Release Notes that covers considerations that should be reviewed carefully by those moving to this new release (see “Cautions” on page 32).

For users already on Release 6.0.1, the information in this guide includes sections indicating the changes in Release 6.0.2. For users moving from Version 4 or 5 of tableBASE, or who are new to tableBASE, please refer to all sections of the document to fully understand all the information that applies to all release levels of Version 6.

tableBASE Version 6 operates with z/OS and OS/390, and is compatible with CICS TS, IMS TM, DB2 and batch.

Highlights of tableBASE Version 6 all Release Levels

- tableBASE Version 6 provides a fully re-entrant engine that extends tableBASE performance benefits to multi-tasking applications.
- Virtual Table Share (VTS-TSR) provides read and write access to shared tables from any region.
- tableBASE uses Data Spaces for local TSR and VTS-TSR, allowing for an increased size up to 2 G.
- Support for DB2 stored procedures managed either by DB2 or by Workload Manager (WLM) is introduced in this release.
- OTE functionality is supported in CICS Transaction Server 2.2 through 3.1. Support of OTE reduces the number of TCB switches and consequently the CPU usage.
- A single unified API (Application Programming Interface) that supports all prior-release stubs, for all environments, with a single link-edit include library.

- The internal locking of objects within tableBASE has been greatly improved. For the user this means enhanced performance.
- Many code paths have been shortened to provide faster, more efficient access to tables. This is particularly true of access to VTS-TSR tables where the code path is now the same as that for the local TSR.
- Enhanced performance for initial table access and for table access without the table's handle (also known as the pivot). A more efficient algorithm is used to significantly improve the time needed to locate a table. The table's handle (pivot) is preserved in the command area for subsequent accesses to the same table (see "Table Access Optimization" on page 29).
- Improved internal tableBASE locking ensures the fastest speed for table access and reduces contention.
- Improved API simplifies tableBASE access for C++ and Java applications — the Version 5 API required the user to specify a variable number of parameters for calls to tableBASE. The variable number of parameters depends on the command specified. For C++ and Java this syntax was cumbersome. With Version 6, four parameters may be specified on every call. The user enters null pointers for the parameters that are not required for the specified command.
- New capabilities have been added to the EXPORT function available in the TBEXEC utility to allow exporting by table rather than just by library. The IMPORT command is unchanged and continues to work on all tables specified in the input file.
- Tightened memory management supports 24x7 use of tableBASE.
- Improved library conversion process makes the move from Version 5 libraries to the new Version 6 format easier.
- Improved open table recovery in error situations is automated. This feature requires the PC Server to be operational.

Additional Enhancements in Version 6 all Release Levels

Performance Enhancements

- A new memory-model provides more efficient memory management of the TSR.
- MVS paging of TSR pages is reduced. This should result in reduced overhead in the tableBASE regions and less paging by the operating system.
- tableBASE libraries can have custom blocksizes. If the tables are large or have a big row size, a custom blocksize allows for fewer I/Os to read and write tables. Libraries with different blocksizes may be used in the same tableBASE Library List.
- Library directory caching improves time to load tables into memory or to store to disk, particularly when multiple tables are being opened, which is the case during the initial loading of a TSR.
- Two new methods are used to put CICS transactions into wait state if the transaction elects to wait while opening for write a table that is already open for write in another

region. These methods do not affect the QR TCB and they do not use excessive CPU as did the technique used in prior releases.

- Almost all Version 6 modules are re-entrant, making them candidates for LPA.
- Batch applications can be built using multitasking and tableBASE will handle each task as a separate entity with its own execution state.

Operational Enhancements

- The tableBASE Root and Nucleus modules are above the line, as is the tableBASE CICS Resource Manager.
- Additional TBOPT parameters have been added to provide runtime overrides for most of the installation options.
- Version 6 I/O routines now tolerate open failures and I/O errors that may have previously resulted in region failures.
- Improved diagnostics are provided with the use of error subcodes in the extended command area.
- Individual libraries may be set to read-only access.
- The Version 6 stub is re-entrant, allowing you to build fully re-entrant application load modules.
- A library of any size may be expanded.
- All API stubs are shown in CICS EDF tracing. In previous releases, TBCALLC, TBASEV, TBCALLV were not captured by the EDF trace.
- If a CICS task is put into a wait for a table exclusively held—opened for write—by another region, a CEMT INQUIRE task will show TBLBASE in the Suspend Value field.
- STGPROT may be used with TBASEV and TBCALLV.
- The re-entrant API stub allows for re-entrant applications that can be used with the CICS RENTPGM feature.
- Support for CICS storage protection keys is enhanced. Almost the entire tableBASE execution path is in the key set by the installation's CICS EXECKEY definitions, affording the protection of this CICS feature.
- The CICS TBOPT file may now be QSAM (including DD *) or, as it was in previous releases, VSAM.
- More flexibility in specifying TBOPT input, more options, and the ability to list all options have been added.
- The TBOPTV functionality has been integrated into TBOPT, allowing for a single source of runtime parameter input. TBOPTV is still maintained for backwards compatibility.
- The allocation of VTS-TSRs is done in a way that does not effect your site’s MVS MAXCAD parameter.

- The default data-sensitive processing date rolls over at midnight.

Administration and Maintenance Enhancements

- A common naming structure for all load modules has been implemented for easy identification. It is DK1xxxxx. tablesONLINE, an optional component of tableBASE, does not yet follow this convention.
- Ongoing maintenance is reduced with the introduction of a single load library that contains all Version 6 tableBASE modules.
- The Version 6 install process is simplified. The process is the same for all interfaces (batch, CICS, IMS, and VTS-TSR) and there is a uniform treatment of the install options.
- A library diagnosis utility is provided that allows for detection of errors in tableBASE libraries.

New in tableBASE 6.0.2

- The fix for the data integrity issue in 6.0.1 when the Status Switch WAIT set to Y (see Chapter 3 of the tableBASE Programming Guide) in your tableBASE environment. Release 6.0.1 causes the waiting application to access the previous generation of the table, thereby potentially losing the changes of the current generation.
- The library conversion and diagnostic tools of 6.0 and 6.0.1 (DK1TLBFX, DK15CNV1 and DK15CNV2) have been replaced by two new utilities: DK1TCNV and DK1TLCHK.
- Improved error detection in the table update and store functionality.

Moving to 6.0.2 from 6.0.1

- Release 6.0.2 is a complete replacement for all executable modules from Release 6.0.1.
- Release 6.0.2 tableBASE libraries are compatible with all Version 6 libraries (Version 6 libraries, however, are not compatible with Version 5 libraries).

Exceptions to Compatibility with Previous Versions

tableBASE 6.0 is compatible with Versions 4 and 5 with the following exceptions:

- All tableBASE libraries must be converted to Version 6 or Library Bridge format before use with tableBASE Version 6.
- With the TSR now resident in Data Spaces and up to 2 GB in size, there is no longer the need for complex memory-saving methods. As a result, Version 6 no longer supports paged tables and ROLL-IN/ROLL-OUT/TBTSLIB. The utility program DK1TCNV integrates the conversion of paged tables to memory-resident tables in the

library conversion process. Please refer to the tableBASE Installation Guide for more information.

- In Version 6, MULTOPNX=Y.
- The VTS Agent no longer runs the CMD file during initialization.
- With Version 6, the interface to the user exits—formerly called system exits—has been formalized to provide a well-defined interface to user exit code which should be stable over subsequent releases. However, it is not compatible with that of prior releases. Please refer to the tableBASE Administration Guide for more information.
- The Refresh functionality used with the optional VTS component has been modified and will require changes in the applications that use this functionality. However, with the upgrade of VTS to read/write functionality, the Refresh functionality may no longer be needed.

Changes of Note

The following changes should be noted:

- The XX command no longer closes paged tables that have been updated because Version 6 does not use paged tables.
- TBACC and TBINDX programs are still available in Version 6, however they have not been modified. These programs continue to access main memory as they did in previous releases.
- With previous releases of tableBASE, the use of the TBCALLV was promoted as a way to access VTS with the shortest path. This is no longer the case with Version 6. A user of TBCALLV offers no advantages in Version 6 and may provide slower performance.
- All ISPF applications and DB2 stored procedures must be linked AMODE(31), and RMODE(ANY).

Maintenance Methodology

Version 6 is no longer maintained by ZAPs.

All levels of Version 6 track maintenance by load module level. A LISTLVL CLIST and job are provided that allow you to list the maintenance level of every tableBASE load module in your library. This information allows DKL support staff to determine whether specific fixes are included in your tableBASE software.

The internal level of Release 6.0.2, when displayed by the LISTLVL CLIST is V60100004.

Future Releases

Release 6.1 is under development. If you have enhancement requests for future tableBASE releases please let us know.

Please contact the tableBASE product manager at

- tablebase@dkl.com or
- 1-800-267-0730.

2
Base Product Changes

The following information applies to all release levels of Version 6.

Naming Protocol

Version 6 features the new tableBASE naming protocol.

All tableBASE executables begin with DK1 for easy identification, a prefix that has been reserved for exclusive use with IBM.

Aliases are retained so that no changes are required to your existing applications.

Multitasking

tableBASE Version 6 is a fully re-entrant engine that extends tableBASE performance benefits to multitasking applications. With a fully re-entrant tableBASE, multiple tasks can open, read, update, and store tables in all TSRs (local or shared).

Note: Multitasking applications require the use of the Version 6 stub TBLBASE and the Program Call Server.

The Program Call Server (PC Server) is new in Version 6. The PC Server provides the functions needed for tableBASE multitasking, accessing a VTS-TSR, and compatibility with DB2 stored procedures. For more information please see the tableBASE Administration Guide.

tableBASE supports batch multitasking where a primary task attaches one or more subtasks. tableBASE is compatible with CICS multitasking through the use of quasi-re-entrant TCBs (QR-TCB) and CICS 2.2 OTE which uses both QR-TCB and the attaching of sub-tasks. tableBASE is also compatible with DB2 stored procedures managed either by DB2 or WLM.

tableBASE can be tailored to perform optimally in your environment by setting execution-time parameters and switches for both batch and online environments. In a batch environment only, the Multitasking parameter can be set to Y to allow multiple

subtasks within a region to access tableBASE concurrently. The normal batch default is N. The default in a DB2 Stored Procedure Address Space is Y.

Note: For a full list of execution-time parameters see the tableBASE Installation Guide.

Above The Line

All Version 6 tableBASE modules can run above the line. However, a small amount of below-the-line memory is required for each active task and for each open library. Those modules that are required to support any RMODE(24) applications will continue to run below the line.

Note: Release 4 and 5 stubs operate only in Rmode(24).

Unified tableBASE Stub

The tableBASE API in Version 6 can be used in CICS, batch, IMS, and VTS-TSR operating environments. tableBASE programs that use this stub can be ported from one environment to another without requiring any stub relinking. Of course, programs must continue to conform to the requirements of each execution environment.

The Version 6 stub contains all the old API entry names such as TBASEC, TBCALLI, TBASEV. All of these old entry points can be replaced by link-editing with the Version 6 stub. However, we strongly recommend using the Version 6 stub itself to gain all the benefits of Version 6.

The Version 6 stub, TBLBASE, is distinct from the stub of the same name used in Version 5. The TBLBASE stubs of previous releases were held in physically separate libraries — one for each operating environment. With Version 6, there is only one TBLBASE for all operating environments. In Version 6, TBLBASE also has a shorter instruction path.

If your application dynamically calls the tableBASE stubs, it can immediately take advantage of the faster path by using the Version 6 tableBASE load library in your STEPLIB concatenation (but subject to the restriction that you are running tableBASE Version 6 and the libraries are converted to the Version 6 format).

The Version 6 stub is re-entrant, allowing you to build fully re-entrant application load modules.

Of course, your stubs for previous releases will continue to work. However, the Version 6 stub is needed to experience the full benefits of Version 6, and we strongly recommend using the Version 6 stub.

Note: The Version 6 stub is not backwards compatible with previous tableBASE releases. If the stubs of previous releases are replaced with the Version 6 stub, the application can no longer be used in Version 4 or 5.

During your initial testing of Version 6, it is possible to have both Version 5 and 6 application libraries where corresponding modules differ only by the tableBASE stub, Version 5 or 6, that is linked-in.

It is possible to run programs, such as various CICS transactions with different stubs from Version 4, 5 and/or 6, under Version 6 in a single CICS region.

Memory Model Modifications

In Version 6, the memory model has been modified to utilize segmented memory for more efficient memory management. Segmented memory means the space allocated for data rows are no longer contiguous and therefore the rows no longer need to be moved to accommodate updates. Instead, an Index is used for each table to point to the rows. This allows for more efficient use of the local TSR and VTS-TSRs.

Previous to this release, memory for each table had to be contiguous. This presented a number of issues. For example, space was not released until a table was closed. In Version 6, when an entire segment becomes empty the space is freed for reuse.

The data will now be maintained as efficiently as possible as a result of using segmented memory. When there is insufficient memory available to load data together, tableBASE will use whatever space is available — even if it is not contiguous.

This change to the memory model also means that all tables are now stored internally as Pointer tables. In previous Releases, tableBASE allowed two type of tables: Pointer and True tables. True tables were characterized by not having an Index and by being stored in contiguous memory. In 6, the concept of True tables still exists, however they are treated within tableBASE as Pointer tables as all memory is now in segmented memory that requires Indexes. The True table Indexes will be transparent to the application program.

Dataspaces

tableBASE now uses Dataspaces for the local TSR and VTS-TSR. This allows the size of a TSR to be increased up to 2 GB. The use of Dataspaces also means that TSRs are better-protected against accidental overwrites by faulty application programs, and that no space is taken up for the local TSR from the region's above-the-line virtual memory.

Note: Maximum table size is limited by the maximum 2 G size minus the system overhead.

Dataspaces used for the VTS-TSRs do not affect the MAXCAD parameter of the MVS operating system. This prevents a MAXCAD issue which could have potentially occurred in Version 4 and Version 5 in which the allocation of VTS Dataspaces competed with other jobs in the system causing the MAXCAD limit to be exceeded and, as a result, intermittent failures in VTS initialization.

Paged Tables

Version 6 loads the table into the Data Space and allows the operating system to handle paging, rather than having tableBASE paging individual blocks to and from a tableBASE library. The library conversion process converts existing paged tables to non-paged tables as it converts the libraries to Version 6 format.

Indexes

All tableBASE tables have Indexes in Version 6. The designation of a table as True (Index = T) is only supported for compatibility with previous releases.

All Indexes are dynamically generated when the tables are opened.

Controlling the Number of Tables in a TSR

tableBASE requires some space in a TSR to maintain the open table directory. In previous releases, expansion of this directory was disruptive to the tableBASE response time. In this release, the directory limit is set during the tableBASE initialization and is dependent on the TSR size. The size of the open table directory can be reduced by specifying the tableBASE execution-time parameter MAXNMTAB=N. For more information refer to the tableBASE Installation Guide.

Note: For a VTS-TSR region, VTSNMTAB is retained for compatibility.

Error Subcodes

Better diagnostics are provided with the use of error subcodes in the extended command area. New applications can make use of this value to pinpoint a reason for failure. If a task has the Abend Switch set ON, a message with the error code and any related subcode is written to the job's JES log. The TBDRIVER utility and the TBDR CICS transaction both display the error subcode.

User Exits

In previous documentation, User Exits were referred to as System Exits. System Exits are reserved for use by DKL and User Exits refer to exits created by tableBASE users.

User Exits that were available in previous versions of tableBASE have been enhanced and renamed. In addition, a new User Exit has been added. This exit gets control after each tableBASE command is processed.

Exits are dynamically loaded at tableBASE start up if the tableBASE execution-time parameter USEREXITS=Y is specified in TBOPT.

Note: The interface to the User Exits in tableBASE Version 6 is not compatible with that of prior releases. The interface has been formalized to provide a well-defined interface which should be stable over subsequent releases.

For detailed information on user exits, please see the tableBASE Administration Guide.

TBCALLC

The operation of TBCALLC may change for some users.

It is now implemented consistently with TBASEC and TBLBASE. That means all calls to TBCALLC will be considered a single thread.

In prior releases, if the TBCALLC user zeroed the TMA Pointer after the first call (or presented a second TMA-AREA with a zeroed TMA Pointer), tableBASE would create a new thread within the transaction.

In Version 6, only the most recent LIB-LIST will be in effect, even if the TMA was zeroed or a second TMA-AREA was presented. TBCALLC calls are supported through CEDF in Version 6.

Alternate Indexes

There have been two changes of note with respect to Alternate Indexes.

In previous releases, it was possible to create an Alternate Index with the same name as the Data Table (even though accessing such a table resulted in an endless loop). However, in Version 6 you are no longer able to create an Alternate Index with the same name as the Data Table. An attempt to do so now results in error code 19.

Note: This problem may occur in Version 6, if a Version 5 table with such a naming problem is imported into Release 6.0.1. Ensure that all libraries that are imported into Release 6.0.1 do not contain Alternate Indexes with the same name as the Data Table.

If an application opened a Data Table using an Alternate Index in Release 4.x, any attempt to open the Data Table directly or open another Alternate Index resulted in error code 84. In Release 5.x this restriction was lifted but an indicator (MULTOPNX) could be set to N to enforce the restriction for compatibility with Release 4.x. However, in Version 6 this restriction is no longer enforceable. The MULTOPNX option is set to Y and cannot be changed.

Generation Number

The absolute generation number of the table being accessed is returned to the user's extended command area, and can be used to verify that the same generation of the table is being used from transaction to transaction.

Changes to tableBASE API Commands

Release Table Command (RL)

The command that changes a table's status from open for write to open for read (RL) has been changed. In previous releases, if RL was applied to an Alternate Index, the status of the Index and all Alternate Indexes were changed to open for read. An RL command applied to an Alternate Index now demotes the Alternate Index to open for read, but leaves the Data Table as open for write.

List Open Tables Command (LT)

There have been formatting and data changes to the LT command and the display drivers, such as TBDRIVER. Figure 2-1 and Figure 2-2 show the differences between the information displayed for Release 5.1, and Version 6.

- I/O has become L/V (local or VTS-TSR). This flag now indicates where the table resides.
- ROLL INS and ROLL OUT have been removed as there is no longer a roll-out library (TSTSLIB).
- ROWS and RWS-BF-EXP have been added. ROWS indicates the current number of data rows in the table. RWS-BF-EXP indicates the total number of data rows that can be added to this table before the data row space or Index space must be expanded.
- BASE TBL has changed to BASE/VTS. If this LT entry represents an Alternate Index the name of the Data Table appears in this field. If the entry represents a linked table, the name of the VTS-TSR appears in this field. For more information on linked tables, please see "Linked Tables" on page 26.

Note: When in a VTS, the display driver shows the VTS name in the top left-hand corner.

If any errors or inconsistencies are found in the inputs to the LT command, -1 is returned in the LIST-BLOCK LIST-TOTAL field in Version 6. Prior releases returned with no indication and no output data.

```
                  ----- HIGH WATER MARKS -----          - CURRENT SPACE -
     TSR            OPEN         TSR          TSR          OPEN         TSR
   ENTRIES         TABLES       USAGE        SIZE         TABLES       USAGE
      34         7,279,224   7,213,560   7,340,032      7,278,176   7,212,512

 REF    NAME    W/R I/O IA?       CALLS       SPACE ROLL INS ROLL OUTS DATA TBL
 ------------------------------------------------------------------------------
  16  TBSYSCP    R   I   N           59         696         0         0
  17  TBOLPROF   R   I   N            6      14,856         0         0
  18  DD......   W   I   N          200      18,128         0         0
  19  TBOLLIBX   R   I   N            4         712         0         0 TBOLLIBR
  20  DV......   W   I   N          182      19,296         0         0
  21  BIG        W   I   N            1   6,658,240         0         0
  22  PRIKEY     W   I   Y           66      15,880         0         0
  23  SP......   W   I   N           12         776         0         0
  24  TBASEMSG   R   I   N           10      11,664         0         0
  25  PRIKEY     R   I   N           96       2,304         0         0
  26  TBDRTUTR   R   I   N           66      67,368         1         1
  27  pRIKEY04   R   I   N        1,344       2,304         0         0
  28  EC......   W   I   N          887       2,208         0         0
  29  TBSYSALI   W   I   N            1         776         0         0
  30  DATKEY     R   I   N            1       2,376         0         0 PRIKEY
```

Figure 2-1: Release 5.1 Information for the LT Command

```
                    ----- HIGH WATER MARKS -----          - CURRENT SPACE -
       TSR           OPEN         TSR          TSR          OPEN         TSR
   ENTRIES          TABLES       USAGE         SIZE         TABLES       USAGE
        17             636K      1,288K       5,120K           636K      1,288K

   REF    NAME    W/R L/V IA?      CALLS       SPACE    ROWS RWS-BF-EXP BASE/VTS
   ------------------------------------------------------------------------------
-
     1 TBOLUSAP  R   L   N            5       8,192      35       252 ........
     2 TBOLACT   R   L   N            4      12,288      13        29 ........
     3 TBSYSTMP  W   L   Y            4      61,440       0        10 ........
     4 TBSYSID   W   L   N           15      61,440       0        10 TBSYSTMP
     5 TBDRTUTR  R   L   N           90      77,824     632       790 ........
     6 TBASEMSG  R   L   N            3      16,384      67       109 ........
     7 EXAMHELP  R   V                1           0       0         0 VTS:DK1O
     8 TBOLPFKS  R   L   N            3       8,192      24       161 ........
     9 TBOLDESC  R   L   N           20      94,208     410       504 ........
    10 TBOLMENU  R   L   N           20      20,480      68        90 ........
    11 TBOLCMDS  R   L   N            4      12,288      65        81 ........
    12 TBOLMSGS  R   L   N            4      94,208     615       767 ........
    13 TBOLHELP  R   L   N            2      12,288      50        70 ........
    14 TBOLLIBR  R   L   N            2       8,192      15        77 ........
```

Figure 2-2: Version 6 Information for the LT Command

Banner Retrieval Command (BN)

The BN command returns the customer information, as before. In Version 6, when the command is called with an optional second data area parameter, BN returns the tableBASE version (16 bytes) 'tableBASE V601'.

Note: The internal level of Release 6.0.2, when displayed by the LISTLVL CLIST is V60100004.

Change Table Definition Command (CD)

When the CD command specifies an Alternate Index rather than a Data Table, the following restrictions are now enforced:

- The Table Index field cannot be changed. It must be P (Pointer table).
- The row size cannot be changed. The Data Table's definition determines the row size.
- The expansion factor and upper and lower densities cannot be changed.
- The read and write passwords cannot be changed. The Data Table's passwords must be used to open an Alternate Index.

In prior releases, attempts to change these fields with the CD command may have resulted in RC = 0 even though no change was made. Now, these attempts will result in RC = 85 (The command is invalid for an Alternate Index).

Divert Table Command (DV)

In prior releases the DV command did not check whether the target DDNAME was allocated. In Version 6, tableBASE checks to see that the target DDNAME is allocated and return an error code 40-1 if it is not.

Fetch Next by key (FN)

This is a new command that has been added in Version 6 that allows you to search for a row that is greater than or equal to the key if the table is in ascending order, or for a row that is less than or equal to the key if the table is in descending order.

Get Next Table Name Command (NX)

The tableBASE Programming Guide for previous tableBASE releases stated that the LIB-SPACE parameter is ignored unless the COMMAND-AREA table name is blank or contains low values. In fact, tableBASE checked only for low values. Version 6 uses the LIB-SPACE parameter if present, independent of the command-area table name.

Previous tableBASE Programming Guides also stated that if the DDNAME parameter is not provided, the first entry in the LIB-LIST is used. However, if the DDNAME parameter is blanks or is null, then the first LIB-LIST entry is also used. Version 6 also operates in this manner.

Open For Write (OW)

In Version 6, a security hole was repaired that was introduced in Version 5. In previous releases, an open for write with a LOCK-LATCH used on an Alternate Index when the Data Table is already open for read, promotes the open table from read to write. This behavior is correct. However, previous releases also allowed you to update the table either directly or through the Alternate Index and then store it without the LOCK-LATCH. In Version 6, the LOCK-LATCH is required.

3 Libraries

This chapter of the Release Notes provides information on the Libraries used in all release levels of Version 6.

Note: For VTS-TSRs, JCL DD statements for libraries must be set to DISP=SHR otherwise enqueues will not be used. If DISP=OLD, they will not be enqueued across the MVS. Furthermore, dataset names must be permanent, not temporary.

z/OS 1.5 with Enhanced Data Integrity

With the release of z/OS 1.5, an option called "Enhanced Data Integrity" can be enabled in SYS1.PARMLIB. This option is designed to protect physical sequential (DSORG=PS) datasets allocated with DISP=SHR from being concurrently opened for update by multiple users.

This can affect tableBASE users, since tableBASE libraries can be allocated with DSORG=PS (even though we only document BDAM (DSORG=DA) and VSAM). TableBASE internally protects tableBASE libraries when allocated with DISP=SHR, so "Enhanced Data Integrity" should not be enabled for tableBASE libraries.

The system programmer can bypass this option for specific datasets. Ensure that tableBASE libraries allocated with DSORG=PS are listed in the exclude list in the IFGPSEDI member.

Specify Library as Read-Only

tableBASE Version 6 respects individual libraries that are designated as read-only. This can prevent accidental writes to the library. An attempt to write results in an error code returned to the caller or the calling task abends, depending on the abend switch setting.

In batch, tableBASE supports customers who wish to use LABEL=(,,,IN) on the DD statement to prevent updates to a tableBASE library and returns error code 61 subcode 11 if an attempt is made.

In CICS, tableBASE supports customers who use the RDO statements READ(YES), ADD(NO), and/or UPDATE(NO)—to prevent updates to a tableBASE library—and returns error code 61 subcode 11: The operation to write to a library was terminated, library read-only; if an attempt is made.

Library Expansion

Any library may be expanded in Version 6. Previous tableBASE versions had limitations that prevented a library from being expanded beyond certain boundary points, depending on the original size of the library. After expansion in Version 6, libraries cannot be converted to previous releases.

Note: Library expansion must be done using the corresponding version of TBEXEC. For example if a Version 6 library must be expanded, use the TBEXEC provided with tableBASE 6.0.2.

Library Directory Caching

The tableBASE library directory can now be cached in memory. This can dramatically reduce the time needed to open a large number of tables when a TSR is initially loaded.

Note: In an in-house test, load time for 40,000 tables from a tableBASE library into a local TSR was reduced from 11 hours to 7 minutes. This was an extreme stress test; your results may vary.

Caching for a library is controlled by coding OPTCD=C on the DD statement for a library. This feature is not supported in a CICS region.

The initial loading of a TSR may benefit from the use of the library directory caching feature, and subsequent refreshes may also benefit if more than one table from the library is being refreshed. However, if updates are being done to the library while the loading or refreshing is being performed, these benefits will be lost. It is also important to note that the library directory caching feature is enabled only at region initialization.

Note: Caching has significant benefit when a library is accessed heavily by a region and other regions are not frequently updating the library. However, caching can have negative performance consequences when multiple regions are frequently updating the library directory.

User-Selectable Blocksize for tableBASE Libraries

Each library can have an optimal blocksize selected at the time it is created.

tableBASE Version 6 features user-selectable blocksizes. Previously, blocksize was fixed at 3120 bytes. Now you can optimize the blocksize for your tables up to the size of a track (or 32,760 bytes, which ever is smaller). This increased flexibility allows you to decrease the time to load tables from the library to a TSR since fewer I/Os are required.

Note: Consider putting all of your large tables together in one library and increasing the blocksize.

Changing the blocksize of a library that has been converted from Version 5 format to 6 prohibits it from being converted back to a Version 5 library.

Libraries created in Version 6 can have any block size from 800 to 32760.

Description of tableBASE Library Versions

With the introduction of tableBASE Version 6 and Library Bridge, there are now a number of types of tableBASE libraries. The attributes of each are highlighted below:

Version 5 Libraries

- Created with tableBASE Version 5 (including Library Bridge tableBASE 5.B)
- Can be converted to Bridge or Version 6 libraries
- Can be accessed by tableBASE Version 5.x and Library Bridge
- 3120 Block Size, Version 5 directory structure, indexes are stored on the library, hash tables may be true or indexed, paged tables supported

Bridge Libraries

- Converted from Version 5 libraries via the Library Bridge conversion utility
- Can be converted back to Version 5 libraries
- Can be converted to Version 6 libraries
- Can be accessed by Library Bridge, and tableBASE Version 6.0.2 (or later)
- 3120 Block Size, V6 directory structure, indexes are stored on the library, no paged tables
- Can *NOT* be accessed by any tableBASE 5.x releases prior to Release 5.B

Note: All releases of tableBASE 6.0.x will continue to provide support for Bridge libraries. However, DKL cannot guarantee support in future releases of tableBASE beyond 6.0.x (e.g. tableBASE 6.1.x). Any modifications or

improvements required to the libraries will only be made to the Version 6 libraries.

Version 6 Transition Libraries

- Created originally by converting Version 5 tableBASE libraries using the job streams CVLB526B and CVLB526V delivered with tableBASE 6.0.1 that are now obsolete, see note below
- Converted from Version 5 libraries using the conversion utility DK1TCNV
- Can be converted back to Version 5 libraries if they have not been expanded
- 3120 Block Size, V6 directory structure, indexes are stored on the library, no paged tables
- Can only be accessed by tableBASE release 6.0.x (or later)

Version 6 Transition Libraries must be converted using the DK1TCNV utility to Version 6 libraries, or to BRIDGE libraries if so desired. The library and can be identified with the DK1TLCHK utility. For more information see “Appendix B: Converting tableBASE Libraries” on page 57 of the tableBASE Installation Guide.

Version 6 Libraries

- Defined with tableBASE Version 6 or are converted from Version 5, Bridge, or Version 6 Transition libraries with the conversion utility DK1TCNV.
- Cannot be converted back to any other type of library
- Can only be accessed by tableBASE Version 6
- No fixed Block Size (This feature does not apply to libraries converted from previous library versions to the Version 6 format.)
- V6 directory structure, no paged tables
- Provide enhanced performance unique to Version 6 libraries:
  - Tables have dynamic indexes that are build at open time, instead of having to read indexes from and write indexes to the library.
  - Libraries that are dedicated to a region may cache their directory information, eliminating many library accesses.
  - Libraries can be defined with BLOCKSIZEs optimized for your table characteristics and DASD devices.

tableBASE Version/Release	**Libraries it can access**
tableBASE 5.x	V5

tableBASE Version/Release	Libraries it can access
Library Bridge (tableBASE 5.B)	V5, V5 Bridge
tableBASE 6.0.2	V5 Bridge, V6Trans, V6

Table 3-1: Compatibility of Library Versions with tableBASE code

Note: The Library Version Identification utility, DK1TLCHK, is used to identify the version of a tableBASE library.

Converting a tableBASE Library

Note: tableBASE Version 6 libraries are not compatible with libraries from previous releases.

Converting libraries to the Version 6 format is an essential part of the migration to tableBASE 6.0.2. This can be done in one of two ways: use the utilities provided with tableBASE 6.0.2 to convert your libraries, or use the utilities provided along with the Library Bridge product.

Note: Please backup your libraries before proceeding with any conversion process.

Conversion Utilities

tableBASE 6.0.2 comes with utilities to allow you to convert your tableBASE libraries to the Version 6 format. Once your libraries are converted, and you've set up your environment (CICS, batch, IMS, etc.) to use tableBASE 6.0.2, you're ready to go.

Library Bridge

The main advantage of using the Library Bridge is to allow an extended conversion period as both Library Bridge and tableBASE 6.0.2 (and later) are able to read Bridge libraries.

Install Library Bridge and convert your Version 5 libraries to Bridge libraries. Bridge libraries can be read by both Library Bridge (tableBASE 5.B) and tableBASE 6.x. Install and set up your environment (CICS, batch, IMS, etc.) to use tableBASE 6.0.2 (or later). Gradually convert Bridge libraries to the Version 6 format, and you're done.

The Library Bridge can only be used with tableBASE 6.0.2 (or later), and is a leased product. Data Kinetics will lease the Library Bridge free of charge to our maintenance customers for a period of 9 months. After this period expires you may continue to lease this product at a monthly rate. For more details visit our web site or contact Customer Support.

All libraries in the Version 6 format have indexes, therefore when converting libraries from Version 5 include an index for each table in the size estimation. If the size estimation is too small (in other words the indexes are not included) then the library conversion can not be completed.

Note: The conversion utilities will also work for Version 4 libraries.

For information on converting your library from Version 5 to 6, please see the tableBASE Installation Guide. For inofrmation on using the Library Bridge to migrate your libraries, please consult the Library Bridge documentation.

Convertion Restrictions

Once a library has been converted to the Version 6 format it cannot be converted back. However, it is possible to convert Library Bridge and Version 6 Transition Libraries back to the Version 5 format provided that the V6TRANS library has not been expanded.

4
CICS

Major improvements to CICS processing are delivered by Version 6. For example, now that the local TSR and VTS-TSRs are in Data Spaces, CICS users will notice an increase in the amount of available memory in CICS regions.

tableBASE Version 6 is compatible with CICS/TS 1.3, 2.1, 2.2, 2.3 and 3.1.

tableBASE Termination

Version 6 CICS allows you to shut down tableBASE in a CICS region and subsequently restart it. When it restarts, it automatically forces new copies of the tableBASE modules to be loaded (on the assumption that the restart is to retreat to back-level versions). This allows you to reconfigure or change tableBASE without restarting a production CICS region.

Use the transaction `TBST TERM` to terminate tableBASE.

Use the transaction `TBST INIT` to restart tableBASE.

Use of CICS Storage Protection

All tableBASE modules are loaded by CICS. Since tableBASE 6 is now re-entrant, STGPROT=YES can be used with applications that use the TBASEV and TBCALLV stubs. To take advantage of this, you must relink your applications with the Version 6 stubs. In previous versions, the TBASEV and TBCALLV modules were MVS-loaded and could not be loaded to key-0 memory because they were not re-entrant. They were loaded into key-8 memory, which caused problems for tasks running in UserKey (9) in a STGPROT environment.

If you have the optional VTS component, you no longer need to include the load library containing the VTS modules in the STEPLIB concatenation with Version 6.

CICS Userkey/CICSkey

Version 6 of tableBASE honours the CICS Userkey/CICSkey conventions. Most of tableBASE executes in the designated EXEC key, thus providing the protection intended by the CICS multi-key feature.

Waiting CICS Tasks

Version 6 of tableBASE uses several strategies to handle CICS tasks that are allowed to wait while opening a table for write that is held exclusively by another region.

In previous releases, a simple delay-and-retry loop was used to handle this type of CICS task. This approach often resulted in degraded performance because the CICS region, running at a higher dispatch priority than batch, looped too fast to allow a batch job to complete its work on a held table so that the table could be released for the CICS users.

If the tableBASE PC Server is installed at the site, a very efficient algorithm is used that lets the CICS task wait for the availability of the held table without impacting the CICS QR (quasi re-entrant) task.

Note: The Program Call Server (PC Server) is new in Version 6. The PC Server provides the functions needed for tableBASE multitasking, accessing a VTS-TSR and is compatible with DB2 stored procedures.

If the tableBASE PC Server is not installed, then a second algorithm is used. This algorithm uses a proxy subtask to wait for the table to be released and the CICS task is suspended—again, with no effect on the QR task—until its proxy subtask notifies it that the table is available.

For more information on the PC Server see the tableBASE Administration Guide.

VTS.LOAD

In previous releases of tableBASE, TBCALLV and TBASEV (along with TBROOTV and TBNUCLV) were MVS-loaded into memory. For this reason, they had to be in a library that could be concatenated to the CICS region's STEPLIB. The special APF-authorized library VTS.LOAD was provided for this purpose.

In Version 6, all tableBASE modules required in a CICS region are CICS-loaded from the RPL concatenation. You no longer need the STEPLIB concatenation with the following qualification: if your applications had statically linked stubs TBASEV or TBCALLV, they must be relinked to get the Version 6 stub. Applications that dynamically called TBASEV or TBCALLV will automatically use the Version 6 stub.

5 VTS-TSR

Read/Write VTS-TSR

Version 6 introduces update capabilities to the VTS-TSR. This allows the complete removal of any tableBASE transaction affinity restrictions.

Tables opened for write in a VTS-TSR can be accessed and modified at the same time by multiple applications running in all tableBASE supported environments, such as batch, CICS, and IMS. The rule that ensures that a table can be opened for write in only one TSR (local or shared) at a time remains unchanged in this release.

In prior releases, VTS commands that attempted to update the VTS-TSR would have received a non-zero error code. With read and write now available in the VTS, those commands, like DK, IK, RK, DC, IC, RC, OW, and MT, among others, will now complete successfully.

Installation of the PC Server is required for access to a VTS-TSR. The PC Server is new in Version 6. It provides the functions needed for tableBASE multitasking, accessing a VTS-TSR, and is compatible with DB2 stored procedures. For more information please see the tableBASE Administration Guide.

Note: Version 6 of the VTS-TSR cannot be accessed by previous releases of tableBASE, nor can tableBASE 6 access previous versions of a VTS-TSR.

Considerations for Moving to Read/Write VTS-TSR

Significant reduction of virtual storage utilization is possible when accessing tables from a VTS-TSR. If your installation uses multiple IMS regions or multiple CICS regions that each require access to the same tables, without VTS the tables are replicated in the TSR local to each region. Since each IMS or CICS region can access tables from a common VTS-TSR, the storage required for local TSRs can be greatly reduced or eliminated altogether. If a local TSR is not required after moving all the tables into a VTS-TSR, the local TSR size can be reduced to the absolute minimum by setting the TSRSIZE=0. See the tableBASE Installation Guide, Appendix A “TSRSIZE - Table Space Region Size”on page 54.

If multiple users could be updating a table—whether online, in a VTS-TSR or other multitasking environment—be aware that a LOCK-LATCH is required to ensure exclusive access to a table.

Note: In a read/write VTS environment, it is important to ensure all tableBASE jobs operate at the same WLM priority level.

LOCK-LATCH

The LOCK-LATCH feature has been present in previous releases of tableBASE but it was limited to the TSR opened within a CICS environment. Version 6 expands this feature into a read and write VTS-TSR.

If required, an application may use a LOCK-LATCH to restrict table updates and closes to only those programs that provide the LOCK-LATCH. The scope of this restriction also extends to all Alternate Indexes associated with the locked table.

Batch applications that may now be operating in a multitasking environment may require a longer COMMAND-AREA to prevent miscellaneous characters from being interpreted as a LOCK-LATCH.

Linked Tables

A linked table is created when a user issues a command to open a table, and during the LIB-LIST search, the table is found to be already open in a VTS-TSR. Since the table is already open in a VTS-TSR that was part of the LIB-LIST, a link is created to that table. A linked table is also known as a remote table.

Note: The LIB-LIST provides the list and order of libraries to be searched when tables are opened. A VTS-TSR can be added to this list and treated as if it was a library using the ML command or the VTSFIRST/VTSLAST execution-time parameters of TBOPT.

In Version 6, a linked table is always treated as opened for read, even if the table in the VTS-TSR is opened for write. This will ensure backwards compatibility with Release 5.x when the VTS-TSR was read-only.

An attempt to open for write a linked table will fail with an error code 13 subcode 5.

A linked table is identified by the use of an LT or GD command.

By setting the VTSNAME in the TBPARM, you can avoid linking the table and access the table directly. For more information, see “Linked Tables and TB-PARM” on page 31.

6
tablesONLINE

This chapter highlights the changes for the Version 6.

tablesONLINE/CICS

New in Version 6

New features were added to tablesONLINE/CICS Version 6:

1. track which user has created or last modified a row is accomplished by defining an eight byte field within the row to receive the logged-on USER ID. For more information, consult "Define Fields in a View" in the Release 6.0.1 tablesONLINE/ CICS User Guide. When a table is being edited, any row that is created or modified will have the USER ID placed into this field.

2. enforce unique primary keys with an Alternate View

3. define the initial value, upper bound, and lower bound of a field

4. control simultaneous updates by multiple users to a single table

While using Alternate Views, there has always been the risk of having duplicate primary keys created when the Data Table is edited with the Alternate View. Now, the primary keys of the Data Table can be incorporated into the design of the Alternate Index View and indicate that you wish to enforce these primary keys in your Alternate View. Taking these steps prevents duplicate primary keys from being created. The default is still to allow duplicate keys. For more information, consult "Enforcing Unique Primary Keys with an Alternate Index" in the Release 6.0.2 tablesONLINE/CICS User Guide.

Before 6.0, only Action Codes could be used to restrict the range of the field input. This was often difficult and cumbersome if you only wanted to set the upper and lower bound. In Version 6 of tablesONLINE, you can set the initial value of a field as well as an upper and lower value. The initial value sets the default input of the field when the row is edited. The upper and lower values are used together to restrict the field input. For more

information, consult "Define Fields in a View" in the Release 6.0 tablesONLINE/CICS User Guide.

Multiple users can edit the same Data Table at the same time. Version 6 allows you to set options to allow multiple users to work in a View and/or and Alternate View on the same Data Table—warnings are provided to indicate to users that multiple user access is in effect.

Duplicate primary keys are not allowed when multiple users can edit a table. Enforcing unique primary keys in a multiple user environment ensures that all of the data can be properly referenced.

A great deal of work went into ensuring browse integrity in this new multiple-user environment. tablesONLINE tracks all of the users accessing a Data Table simultaneously and prevents users from editing the same row at the same time.

If you attempt to access a row that is being edited by another user, a message informs you that the row is already being edited, and also identifies the user and the application.

Although you won't be able to edit the row while another user is editing it, the row is always accessible to browse. For more information, consult "Multiple User Update" in the Release 6.0 tablesONLINE/CICS User Guide.

There have also been improvements to the tablesONLINE/CICS User's Guide. Take a look at the revised section on Description Tables. There are now 16 pages that provide detailed information on how to customize tablesONLINE.

Exits

In Release 6.0, in order to implement the browse integrity enhancement, the offset of the variable T-LCMD-NO-OF-ITEMS has been changed by 4 bytes. If your exits use this variable, you will need to recompile. Otherwise, no changes are needed. The change is flagged by F00017 in the margin of the EXITPARM copybook.

Note: TBOL/CICS requires that working storage be initialized to binary zeroes. Under LE, this is accomplished by ensuring that the run-time storage option for the CICS region is STORAGE=(00,...). Refer to IBMs LE Customization manual for guidance in creating a CEEUOPT or CEEROPT module.

tablesONLINE/ISPF

There have been no changes to the functionality of TBOL/ISPF with this release. The tableBASE driver (TBDRIVER) that you access through TBOL/ISPF is the Version 6 driver.

7 Notes and Cautions

Notes

Table Access Optimization

Table access can be optimized by using a separate command area for each table. Instead of having to search a TSR directory each time to determine where the table resides in memory, tableBASE uses a reserved field in the command area (table handle) pointing to the position of the table in the TSR. By keeping a separate command area for each table accessed, this table handle is used to quickly locate a previously opened table.

High-Speed Processing Without EDF

When using EDF to trace an application in CICS, all calls to any tableBASE entry point will be traced using the tableBASE CICS Resource Manager. If you stop using EDF with the transaction, tableBASE will bypass the Resource Manager and automatically revert to high-speed processing (TB-Turbo option in TB-PARM).

Date-Sensitive Processing

The default date used by date-sensitive processing rolls over at midnight. This will prove useful for 24x7 applications that rely on date-sensitive processing.

Access Register Mode

Version 6 has the following MVS restrictions applicable to AR mode (this applies only to applications that have been coded in Assembler):

1. tableBASE must be called in a primary mode. Failure to do this may result in unpredictable results.

2. tableBASE zeroes all access registers (A0-A15) on return to a caller. It is a responsibility of the caller to save and restore its access registers.

VSAM LSR Pools for tableBASE Libraries

tableBASE must be allowed to control its own I/O activity to ensure the integrity of library directories. If a VSAM library has been associated to an LSR pool, when the library is opened by tableBASE and found to be pooled, it is closed by tableBASE, removed from the pool, and reopened. The exception to this is if the library happens to have DISP=OLD, then it will be left in the LSR pool.

Note: You risk destroying your data if you are using a tool that dynamically modifies the VSAM LSR setting.

When a tableBASE library is set to DISP=SHR, there is an exposure when using third party software like MBO to dynamically turn on LSR processing (or mechanisms equivalent in effect). In these cases, tableBASE is not notified that LSR is turned on and cannot remove the library from the LSR pool. If more than one region can update the library, the library will be corrupted.

The library directory caching feature of Version 6 should provide equivalent or better results than LSR pooling.

Considerations for Working with Large Tables

With Version 6, there are new considerations for dealing with large tables in tableBASE. Both the local TSR and VTS-TSRs are now in Data Spaces. The maximum size of a Data Space is 2 G.

In addition you should consider the region size, the number of tasks running in the region, processing sequence, and the frequency and type of access activity. The choice of one search strategy over another can impact system performance.

What is large? For the purposes of the following recommendations, a table is considered very large when the total size is 50 M or larger, or the number of records is greater than one million. The maximum row size a table can have is 32 K, which is the maximum positive value in a half-word.

Large Table Recommendations

Binary or Hash search methods are recommended for tables with more than 300-400 rows.

High frequency of inserts with low expansion factor are not recommended.

Note: With the introduction of Version 6, all tables are now treated as Pointer tables and have an Index, even if they are listed as being True tables. The change is transparent to the user, but Pointer and True tables are now treated internally as Pointer tables.

Table Space Region

The TSR is in a Data Space in Release 6.0. Rollouts of tables using TBTSLIB are no longer supported with this release. The Data Space must be large enough to hold all tables that were paged tables in prior releases.

Binary Searches

With Version 6, all searches cache the first 4 bytes of the key. This greatly improves search performance. With a large table where these 4 bytes may be the same throughout all rows of the tables these benefits may not be realized. Better results can be achieved by using a key that varies within the first 4 bytes.

Open/Store Activity

The size of the table determines the load on the I/O system, possibly causing slowdowns in an online environment.

Creation Method

For a keyed table, creation in Sequential order can be much more efficient than in Random order.

If the table is already in Random order, consider loading the Data Table and then changing the definition to Sequential.

Expansion Factor

This controls the amount of additional memory that will be obtained to expand non-Hash tables. It should be set sufficiently large so all insertions between opens of the table will fit. For example, if the table has 100,000 rows, and a maximum of 10,000 rows are inserted with no deletions when it is opened, an expansion factor of 100 (10%) is sufficient.

tableBASE Libraries

A tableBASE library will never use secondary allocated space because it can never increase in size dynamically. It will only use the initially allocated space. This initial space allocation does not need to be contiguous.

Linked Tables and TB-PARM

If every tableBASE call in your applications is to VTS-TSR, then using the VTSNAME in the TB-SUBSYSTEM field in the TB-PARM to access VTS-TSR is more efficient than using the VTSFIRST, VTSLAST, or the ML command. Use of the VTSNAME in the TBPARM completely bypasses the use of the local TSR.

If you are using VTSFIRST, VTSLAST, or ML to add a VTS-TSR to the LIB-LIST, the transaction process will first go to the local TSR (through CICS, IMS or batch) to find a handle (pivot entry) for the table requested, and if it is the first request for that table, a dummy entry will be entered in the local TSR. The process then goes to the VTS-TSR to locate the table entry.

The table entry address in the VTS-TSR would then be used to update the initial dummy entry in the local TSR.

On subsequent passes, the local TSR would have a valid address as a Pointer to the VTS-TSR for that table. This process describes a linked table. Also see "Linked Tables" on page 26).

Note: Use of VTSFIRST, VTSLAST, or the ML to command to access a table in VTS-TSR (linked table) does not allow for update commands against the table. Updates of the table can only be achieved by the use of the VTSNAME in the TB-PARM to access the table.

XML and tableBASE

The ability to export COBOL copybooks can be used to assist in creating XML from your tableBASE data and in moving tableBASE data out of tableBASE using third party tools. For more details, please see the tableBASE Batch Utilities Guide.

Cautions

tableBASE and BMC CICS monitor (2031)

When using the BMC CICS monitor in a CICS environment tableBASE transactions appear to stop. Transactions going to tableBASE appear to enter tableBASE but do not appear to exit tableBASE. High CPU usage is also observed.

Solution

You can either disable and not use the BMC product or contact BMC to obtain the software fix.

Alternate Index Problem in Version 5

In previous releases, it was possible to create an Alternate Index with the same name as the Data Table (even though this table was consequently not usable). However, in Version 6 you are no longer able to create an Alternate Index with the same name as the Data Table. An attempt to do so now results in error code 19.

This problem may occur in Version 6, if a Version 5 table with such a loop is imported into Release 6.0.1.

Solution

Ensure that tableBASE libraries in Version 5 format that are to be converted to Version 6, do not contain Alternate Indexes that have the same name as the Data Table.

Error Importing Library (2043)

An invalid true hash table in Version 5 (such as one that has more populated rows than the row count value in the table definition indicates) will cause errors during the import process into a Version 6 release of tableBASE.

Solution

The TBEXEC EXPORT command is modified in Release 6.0.1 to return an error if a hash table has more rows than specified in the definition.

Note: Prior versions of TBEXEC will still incorrectly export an invalid hash table. The Version 5 library can be converted directly to Release 6 using the library conversion utility provided. A corrected version of TBEXEC for 5.1.0 is now available by request.

Enqueue Missing For Table Opened for Write (2008)

tableBASE issues an MVS enqueue for any table open for write in a TSR if the tableBASE library is allocated as DISP=SHR. If a tableBASE library is allocated as DISP=OLD the enqueue is not obtained.

If the table is opened in a VTS-TSR the table will not be protected by the enqueue once the job that opened the table has terminated. This could result in a table being opened for write in multiple TSRs.

Solution

If a job opens tables for write in a VTS-TSR ensure that the job allocates the tableBASE library as DISP=SHR.

TSR Out of Space While Creating a Table (2064)

When populating a table in a TSR with limited free space, error 92 (insufficient TSR size) may be returned prematurely. The index may contain up to 16 free slots when this condition occurs.

This can also occur when a table is reopened in a TSR with the RF, OR, and OW commands. The space required must be sufficient to accommodate the actual number of rows plus the table expansion factor.

Solution

When allocating a table ensure that the estimated number of rows is 16 greater than the maximum rows expected.

If tables can increase in size ensure that the TSR has sufficient space to accommodate the expansion.

Table Remains Locked in a TSR (2065)

If a job that accesses tableBASE is cancelled or abnormally terminates during tableBASE processing, a table may remain locked in the TSR. The subsequent attempts to access the table may result in loops.

Solution

Recycle the TSR (local region for a local TSR or the VTSAGENT for a VST-TSR).

Customer Anchor Table DKL slot invalid (1899)

When the PCSERVER initializes, and attempts to create or use an internal anchor, the PCSERVER sometimes erroneously puts out a message:
DK100844W Customer Anchor Table DKL slot invalid.

Solution

Please ignore message as it is for diagnosis only and, not for tableBASE operation.

Lock out of Batch Processing (2119)

The enabling of Implicit Open can cause CICS transactions to lock out batch processing if the transaction is accessing a table in the VTS that is not open and no tableBASE library for that region exists.

Solution

When no tableBASE library for the region exists, turn Implicit Open off.

8
Documentation Updates

This section lists additions to the DKL documentation that have recently been added. It is listed here, as well as in the appropriate Guides, to assist you in identifying new information.

Access to VTS-TSRs

In prior releases VTS-TSRs users were allowed only read access (updates were supported only through the refresh process run under the VTS address space). Four methods of access were supported:

1. using the TBOPT data set VTSNAME= parameter to specify the VTS-TSR and invoking tableBASE with the TBCALLV or TBASEV API.

2. using the VS command to specify the VTS-TSR and invoking tbaleBASE with any API.

3. specifying the VTS-TSR in the TBPARM data area passed as the first parameter on the call to tableBASE.

4. using the TBOPT data set VTSFIRST= or VTSLAST= parameters to specify a VTS-TSR and/or specifying a VTS name in the LIB-LIST of the ML command.

With Version 6 users are allowed read/write access to VTS-TSRs. Methods 1 through 3 (above) are supported as read/write interfaces—with Method 3 as the recommended update interface. Method 4 is supported as a read-only interface as in prior releases.

Commands which access the VTS-TSR and also access a tableBASE library will locate the library based on the LIB-LIST in effect in the local address space (not the VTSAGENT address space which created the VTS-TSR).

The refresh process is no longer run under the VTS address space. The refresh process can be emulated in Version 6 by using the TBDRIVER utility. The SELECT SUBSYSTEM (SS) command is used to specify the VTS-TSR for subsequent commands.

The REFRESH (RF) command provides the same functionality as it did in the REFRESH process in prior releases.

Determining TSR Size

In prior versions of tableBASE the area for tables was acquired from local virtual memory. (Only in VTS-TSR was a dataspace acquired.) If a TBOPT parameter specified TSRSIZE= then the table space was acquired in a single GETMAIN request. If it was not then GETMAIN requests were made for every table as it was opened. If TSR size was specified then a tableBASE rollout library allocated with DDNAME TBTSLIB was also required to provide overflow space for the TSR.

With Version 6 the area for tables is always provided by acquiring a data space. This means a TSRSIZE= is always used, either from the default setup at installation or the TBOPT data set for this job. See Appendix A in the Installation Guide.

CD

Return Value Error Code 92—In the event of receiving an Error Code 92 create more space in the VTS-TSR or local TSR. Retry the CD command.

Notes In the event of an ABEND when using the CD command review the table to confirm data integrity.