

tableBASE

Installation Guide

Release 6.1.1



Copyright © 2015 DataKinetics Ltd.

Document Number: TBM010-R611v1.1

Publication Date: tableBASE Release 6.1.1 Version 1.0 - September 2012

tableBASE Release 6.1.1 Version 1.1 - April 2015

The guide is copyrighted and all rights are reserved. This document may not, in whole or in part, be copied, photocopied, reproduced, translated or reduced to any electronic medium or machine-readable form without the prior written consent of DataKinetics Ltd.

Information in this guide is subject to change without notice and does not represent a commitment on the part of the vendor. The software described in this guide is furnished under a license agreement and may be used or copied only in accordance with the terms of the agreement.

tableBASE, VTS, VTS Manager and tablesONLINE are registered trademarks of DataKinetics Ltd. The names of other products or companies may be trademarks or registered trademarks of their respective companies.

DataKinetics Technical Support: 1-613-523-5588

Email: tableBASE@dkl.com

DataKinetics Ltd.
240 - 50 Hines Road
Ottawa, ON
Canada K2K 2M5
<http://www.dkl.com>

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

Facsimile: (613) 523-5533

Table of Contents

Preface	9
Audience for this Guide	9
What is Covered in this Guide	9
What is new in Version 6	10
Naming protocol	10
Glossary	10
Conventions used in this guide	13
Additional tableBASE references	14
VTS Manager references	14
Customer Support	15
Training and Professional Services	15
About DataKinetics	15
1—Installation overview	17
Description of Installation Procedures	18
Installation procedure for Release 6.1.1 without VTS Manager	19
Installation procedure for Release 6.1.1 with VTS Manager	21
Installation procedure for VTS Manager on existing Release 6.1.1	23
2—Preparing for Installation	25
Step 1: Prepare for tableBASE installation	25
Step 2: Obtain installation files from installation media	28
Step 3: Verify recommended security considerations	30
3—Installing the tableBASE PC Server	31
Steps to install the tableBASE PC Server	31
Step 1—Move load modules to authorized load library	32
Step 2—Modify PROC and Install in PROCLIB	33
Step 3—Shutdown the tableBASE Servers for VTS Manager	34
Step 4—Start the tableBASE PC Server	35

4—Installing the tableBASE Servers for VTS Manager	37
Steps to install the tableBASE Servers for VTS Manager.....	38
Step 1—Move load modules to authorized library	39
Step 2—Configure the VTS Manager catalog	40
Step 3—Configure the compat VTS Group Manager catalog	41
Step 4—Modify PROCs and install in PROCLIB	42
Step 5—Shutdown tableBASE PC Server for Release 6.1.1	44
Step 6—Start the tableBASE Servers for VTS Manager	45
5—Installing tableBASE MVS batch	47
Steps to install tableBASE Batch.....	48
1. Modify default parameters (optional)	48
2. Copy tableBASE load modules	49
3. Run tableBASE interface above the line	49
4. Convert tableBASE libraries to Version 6 format	50
5. Point TBDR CLIST to newly installed libraries.....	51
6. Update binder procedure to use the Version 6 TBLBASE API	51
7. Convert BDAM table libraries to VSAM (optional)	52
8. Confirm successful install.....	52
6—Installing the VTS (Virtual Table Share) product	53
Steps to install the VTS product	53
Step 1—Modify default parameters (optional).....	54
Step 2—Copy tableBASE load modules (optional)	55
Step 3—Confirm successful VTS install	55
7—Installing the VTS Manager product	57
Steps to install VTS Manager	57
Step 1—Modify PROCs and install in PROCLIB	58
Step 2—Start the tableBASE Servers for VTS Manager	60
Step 3—Validate the VTS Manager installation	60
8—Installing the CICS Interface product	67
Difference between Release 5.x and Version 6 affecting CICS	67
Steps to install the CICS Interface product.....	67
Step 1—Modify default parameters (optional).....	67
Step 2—Copy tableBASE load modules (optional)	69
Step 3—Convert from BDAM table libraries to VSAM (optional)	69

Step 4—Create CICS definitions for tableBASE libraries	69
Step 5—Create CICS definitions for tableBASE programs and transactions	69
Step 6—Create CICS definitions for tableBASE journal files	70
Step 7—Edit PLT	70
Step 8—Update binder procedure to use new TBLBASE API	71
Step 9—Customize CICS startup JCL	71
Step 10—Confirm successful CICS Interface install	71
Step 11—Change system abends to task abends	72
Notes and restrictions.....	74
Level of CICS	74
CICS/TS CSD	74
FCT restriction for tableBASE libraries	74
9—Installing the IMS TM Interface product	75
Differences between Release 5.x and Version 6 affecting IMS	75
Steps to install the IMS/TM Interface product	76
Step 1—Modify default parameters (optional)	76
Step 2—Copy tableBASE load modules (optional)	77
Step 3—Check pre-load list	77
Step 4—Customize IMS startup JCL.....	77
Step 5—Confirm successful IMS Interface install	77
10—Installing the DB2 Stored Procedures Interface product	79
Steps for DB2 Stored Procedure support	79
Step 1—Modify default parameters (optional)	79
Step 2—Copy tableBASE load modules (optional)	80
Step 3—Customize startup JCL.....	80
11—Installing the tablesONLINE/CICS product	81
Steps to install tablesONLINE/CICS.....	81
Installing tablesONLINE/CICS for the first time	82
Step 1—Copy the tablesONLINE control libraries	82
Step 2—Customize CICS startup JCL.....	82
Step 3—Enable batch job submission from CICS—Part 1	83
Step 4—Create CICS definitions	83
Step 5—Confirm successful tablesONLINE install	84
Step 6—Enable batch job submission in CICS—part 2	84
Step 7—Change Fetch Generic Delimiter (optional)	84
Step 8—Confirm successful tablesONLINE/CICS install	85
Notes and restrictions.....	85
CICS/MRO or CICS/ISC	85

Converting Views	85
TWA size for tablesONLINE/CICS transactions	86
Upgrading tablesONLINE/CICS from Release 4.2.x, 5.0.x, or 5.1	86
Upgrade 1—Create new Version 6 TBACTLB library	86
Upgrade 2—Create new Version 6 TBDICLB library	87
Upgrade 3—Create new Version 6 TBAPPLB library	87
Upgrade 4—Update description tables	87
Upgrade 5—Update menu tables	88
Upgrade 6—Update message tables	88
Upgrade 7—Update help table	89
Upgrade 8—Update security module (optional)	89

12—Installing the tablesONLINE/ISPF product 91

Steps to install tablesONLINE/ISPF	91
Step 1—Change CLIST to variable blocked file (optional)	91
Step 2—Point TBALLOCS CLIST to newly installed TBSYSLB	92
Step 3—Point TBSKDEF SKELETON to newly installed load library	92
Step 4—Modify the primary ISPF menu to point to tablesONLINE	92
Step 5—Modify Help tutorial to point to tablesONLINE help	93
Step 6—Connect tablesONLINE to ISPF	93
Step 7—Confirm successful tablesONLINE/ISPF install	94
Notes and restrictions	94

Appendix A 95

tableBASE run-time options	95
ALLOW_TB_LIB_IO—Allow tableBASE library I/O	95
CICSJRNL—CICS Journal File ID	95
DATERTNX—Date-Sensitive Processing Found Code	95
FGDELIM—Fetch Generic Delimiter	96
HASH_HI_DEN_LIM—High Density Limit for Hash Indexes	96
HASH_LOW_DEN_LIM—Low Density Limit for Hash Indexes	97
LDS—Linear Data Set Use	97
LIBnn, ML—tableBASE Library List	97
LISTOPTIONS—List Parameter Options	98
LOCKTIMER—Lock Timer Wait Value	98
LOCKTIMEWTO—Lock Timer Message Wait Value	98
MAXNMTAB—Maximum Number of Tables	98
MTRETAIN—Retain Rows and Index Areas	99
MULTITASKING—Multitasking	100
MULTOPNX—Multiple Alternate Index	100
OVERRIDES—Allow Changes to Status Switches	100
SWITCHES—Status Switches	101

STROBE—Strobe Interval	103
SUPPRESS_DUMPS—Suppress tableBASE messages and dumps.....	104
TABLEWAITRC—Table open enqueue wait time.....	104
TABLEWAITWTO—Table open enqueue report time	104
TPVM—VTS Group Manager Use	105
TSR_ALGORITHM—Optimize TSR Usage.....	105
TSR_WARNING_FREQ—Frequency of TSR allocation warnings.....	105
TSR_WARNING_PCT—Percentage of TSR allocation for warning activation.....	106
TSRACCESS—Read Only (RO) or Read Write (RW) TSR access	106
TSRSIZE—tableSPACE Region Size	106
USEREXITS—User Exits	108
VTSFIRST, VTSLAST—VTS Search Sequence.....	108
VTSNAME—specifying the name of a VTS-TSR.....	109
VTSPREFIX—VTS prefix definition	109
ZEROROWS—Zero data table rows on de-allocation.....	110
Appendix B	111
Converting tableBASE libraries	111
Description of tableBASE library versions	111
Converting a tableBASE library	114
tableBASE version 5 library expansion limits.....	127
DK1TLCHK utility.....	129
Appendix C	131
TBOPT dataset coding.....	131
Appendix D	133
tableBASE LPA-eligible programs	133
Appendix E	135
Source modules.....	135
Appendix F	139
JCL members	139
Appendix G	145
Distribution datasets and libraries.....	145

Mandatory tableBASE datasets	145
Optional tableBASE datasets.....	146
Mandatory tableBASE libraries.....	146
Optional tableBASE libraries	147

Preface

This guide provides information on how to install the MVS versions of the tableBASE software, and its optional products for Release 6.1.1 of tableBASE.

Release 6.1.0 was the first release of tableBASE which will allow you to install the optional DataKinetics VTS Manager product. VTS Manager may be installed with an upgrade to a new modification level of tableBASE or on top of an existing tableBASE release, the former of which is recommended.

[Chapter 1 “Installation overview”](#) on page 17 will provide you with more details and a roadmap on which procedures should be followed to perform the installation.

Audience for this Guide

This guide and the *tableBASE Release Notes* are intended for use by system installers and/or system administrators responsible for installing the tableBASE Version 6 software. The person installing tableBASE should be familiar with a programming language, compile procedures, the linkage editor, and MVS JCL.

If you are upgrading to a new *modification level* of tableBASE from a previous *modification level* (without adding a new product or component) — for example, from Release 6.0.2 to Release 6.1.1, or from Release 6.0.3 to Release 6.1.1 — you should consult the relevant installation instructions in the *tableBASE Release Notes*, which will explain the most efficient way to accomplish your upgrade.

If you are installing a new *release* or *version* of tableBASE — for example, moving from Version 5 to Version 6 or installing Release 6.1.1 for the first time — you should continue following the steps and procedures in this guide. If you are unsure about the distinction we are making here between the terms *version* and *release*, see the [“Conventions used in this guide”](#) on page 13 below for more detail.

What is Covered in this Guide

This guide contains the installation procedure for the tableBASE base product (batch interface) as well as the procedures for the installation of the optional CICS and IMS TM interfaces, the VTS (Virtual Table Share) product, the VTS Manager product, and the tablesONLINE products for CICS and ISPF. Also covered in this guide are parameter settings, parameter delivered defaults, and overrides.

What is new in Version 6

Some of the highlights of Version 6 include:

- A simplified installation process. The process is similar for all products (batch, CICS, IMS, VTS, and VTS Manager) and there is a uniform treatment of the install options.
- A single load library that contains all Version 6 tableBASE modules, as well as the modules for all additionally licensed DataKinetics software products.

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.

All tableBASE version 6 modules have the same module names. Modules received for the installation should be used to entirely replace earlier Version 6 modules.

Glossary

The following terms are used throughout this guide:

Data Table	A Data Table is the actual raw data. Each Data Table has a table definition (DT-BLOCK) that is used to generate the Index for the Data Table.
Index	An Index is defined for each Data Table. A Data Table Index is generated dynamically when a table is opened or defined 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. Alternate Indexes are optional, and there is no limit to the number of Alternate Indexes a Data Table may have.
Delivered defaults	The defaults that are delivered with the product. Also known as <i>factory defaults</i> .
Installation defaults	The defaults set at installation time by an administrator, which may or may not be the same as the delivered defaults. Defined using the TBOPTGEN file. (These defaults may be overridden by an individual application using the TBOPT file.)

TSR	Table Space Region. A data space of up to 2G is used by tableBASE to house tables. The data space is owned by an application in the associated address space. The application uses tableBASE to access data within the tables.
Local TSR	As above.
Shared TSR	A VTS-TSR; see below.
Temporary Table	A temporary table exists only within a TSR, and is created by the DT command (or IA). It is never stored in a library. A temporary table can be distinguished from a library table using the GD command output—if found, a temporary table will show no dataset name.
Linked Table	A linked table (also known as a remote table) is created when a user issues a command to open a table that is already open in a VTS-TSR specified in the LIB-LIST. The table entry in the local TSR is linked to the existing open table in the VTS-TSR. No updates are allowed to a linked table.
Table Expansion	Dynamic allocation of space for tables in the TSR when the initial space allocated becomes insufficient.
Multitasking Batch	An MVS region that implements multitasking by attaching multiple Task Control Blocks (TCBs). This can include a batch job that attaches several subtasks or a transaction processing region like DB2 stored procedures that implements multitasking through multiple TCBs.
View	A tablesONLINE View provides the field, edit and display attributes for a Data Table with its Index. In releases previous to Version 6 (and Version 5) the View was referred to as a Field Definition Table (FDT).
Alternate Index View	A tablesONLINE Alternate Index View is identical to a View but applies to a Data Table when access is through an Alternate Index.
VTS	Virtual Table Share. The term “VTS” may be used to refer to the DataKinetics VTS product, which permits TSRs to be shared among applications. These shared TSRs are called VTS-TSRs.
DataKinetics VTS	The product that provides the shared VTS-TSR capability.
VTS-TSR	A Virtual Table Share (VTS) Table Space Region (TSR) is a shared TSR, and resides in a shared data space. Applications can access tables within a VTS-TSR, and use the information as if it were within their local TSRs.

VTS Agent	The DK1VAGNT program, which initializes VTS-TSRs in tableBASE, and then sits idle until the VTS-TSR is to be terminated. If VTS Manager is installed, VTS-TSRs are managed by VTS Group Managers, and the VTS Agent is not required, but is still available for transition purposes.
VTS Manager	A DataKinetics Ltd. product that extends the functionality of the VTS shared TSR capabilities.
TPM	An internal DataKinetics development term which refers to the VTS Manager top tier component. It will sometimes be found in diagrams which explain the architecture and internal processes of VTS Manager. Also occasionally refers to the VTS Manager itself.
VTS Group Manager	The middle tier of VTS Manager which manages VTS-TSRs.
TPVM	An internal DataKinetics development term which refers to the VTS Group Manager component of the VTS Manager product. It will sometimes be found in diagrams which explain the architecture and internal processes of VTS Manager.
Catalog	A catalog contains definitions of managed items in the next tier down in the hierarchy. Definition information about the VTS Group Manager is contained in the VTS Manager catalog, and VTS-TSR definition information is contained in the VTS Group Manager catalog. The catalog is contained within the LDS associated with the VTS Manager or VTS Group Manager.
Cataloged VTS	A VTS-TSR that is managed by a user-defined VTS Group Manager under VTS Manager, as opposed to the <i>compat</i> VTS Group Manager.
LDS	Linear DataSet used for VTS Manager.
Alias name	An alternate name for a VTS-TSR that can be created, assigned, and used in lieu of the name assigned at VTS-TSR definition.
VTS switch	The action of switching an alias name from one VTS-TSR to another. This is a feature of VTS Manager.
<i>compat</i> VTS Group Manager	The <i>compat</i> VTS Group Manager is the default VTS Group Manager that runs under VTS Manager.

Conventions used in this guide

This guide uses certain conventions to differentiate code and typed commands, to display the names of parameters, and to describe specific version and release levels of the tableBASE product.

Convention	Description
code examples and commands	Code examples and commands appear in this type of font: this is an example of the 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 IBM standards, the term <i>version</i> refers to a generation of a software product that has significant new code or new functionality. <i>Version</i> is a more general term than <i>release</i> . For example, <i>Version 6</i> includes <i>Release 6.1</i> and <i>Release 6.2</i> , and is equivalent to <i>Release 6.x</i> .
Release	Following IBM standards, the term <i>release</i> refers to a program or set of programs which represent a specific revision to the base version of a software product. For example, <i>Release 6.0</i> is a term that is used to identify the first release of <i>Version 6</i> . Subsequent releases made available under the Version 6 umbrella, such as <i>Release 6.1</i> , will provide additional revisions to the base product.
Modification Level	Following IBM standards, the term <i>modification level</i> refers to the application of specific program enhancements and error corrections to the release of a software product. For example, <i>Release 6.0.3</i> is at <i>modification level 3</i> , and <i>Release 6.1.0</i> is at <i>modification level 0</i> .
MVS	MVS is a generic term which is used when referring to z/OS and other related IBM operating systems.

Additional tableBASE references

This guide is one of a series that describe tableBASE and tablesONLINE:

- *tableBASE Release Notes*
- *tableBASE Installation Guide*
- *tableBASE Concepts and Facilities Guide*
- *tableBASE Batch Utilities Guide*
- *tableBASE Administration Guide*
- *tableBASE Programming Guide*
- *tableBASE Quick Reference Guide*
- *tablesONLINE/CICS User's Guide*
- *tablesONLINE/ISPF User's Guide*
- *tableBASE Library Bridge Manual, Release 5.B*

VTS Manager references

For more information about the DataKinetics VTS Manager product, consult the following manuals:

- *VTS Manager Concepts and Facilities Guide*
- *VTS Manager Administration Guide*

Customer Support

Customers with maintenance agreements can obtain 24-hour hotline support through a direct line to our Technical Support staff (+1-613-523-5588).

The DataKinetics customer support Service Level Agreement lists the guaranteed response times for each problem severity level. High priority critical and serious problems receive immediate attention from our Technical Support staff.

Before calling the DataKinetics Technical Support staff, be sure to consult the "Support" section of our web site (www.dkl.com) for all the latest information on our technical support protocol. Look in "Support" for our "FAQs", which provide answers to frequently asked questions posed by our users.

The "Customer Login" link supplied on our web site navigates to a special support section which is available to customers who have maintenance agreements with DataKinetics Ltd. A username and password are needed to access this section of the web site—consult your tableBASE administrator, or DataKinetics customer support, for this information.

Training and Professional Services

DataKinetics Ltd. is proud to be able to offer our customers the very best in training and technical consulting services. See the "Services" section of our web site for all the latest information regarding our offerings in this area.

About DataKinetics

DataKinetics®, the leader in enterprise IT optimization, enables the top data centers in the world to overcome today's mainframe throughput, performance, capacity and interconnectivity challenges. In business for over 30 years, DataKinetics products are powering nearly a billion mission-critical transactions per second every day for the world's largest banking, insurance, credit card, brokerage house and retail organizations.

1

Installation overview

The tableBASE software is available for z/OS environments. The base product consists of the tableBASE nucleus and the batch component.

Optional DataKinetics products are available to interface tableBASE software with the IBM transaction servers, IMS TM and CICS. Other available products include VTS (Virtual Table Share), VTS Manager, and tablesONLINE. The VTS software provides an environment to share tables across regions, significantly increasing performance. VTS Manager is a VTS-TSR management product, providing flexibility and a further increase in performance. The tablesONLINE software is a completely menu-driven online 3270-based editor for the tableBASE system.

This chapter covers the information needed to install the tableBASE software and its optional products. All sites receive the tableBASE batch/TSO component, which is the base product. Optional tableBASE products are:

- IMS TM Interface
- CICS Interface
- tablesONLINE/CICS
- tablesONLINE/ISPF
- VTS (Virtual Table Share)
- VTS Manager

Release 6.1.0 was the first release of tableBASE which allowed you to install the optional DataKinetics VTS Manager product.

VTS Manager may be installed with an upgrade to a new modification level of tableBASE or on top of an existing tableBASE release. The former approach is recommended, if you are running an existing release of tableBASE in your production environment. This will allow you to run both releases in parallel. If you choose to install the VTS Manager product on top of an existing tableBASE release, it will not require you to re-install the tableBASE base component and any of its other products. However, you will need to shut down your current programs and restart them.

Description of Installation Procedures

The procedures that follow will provide you with a roadmap of the installation process, depending on whether you are installing Release 6.1.1 with or without VTS Manager, or whether VTS Manager is being installed on top of an existing tableBASE release.

Below is a brief description of each installation procedure and when it should be used.

- [“Installation procedure for Release 6.1.1 without VTS Manager”](#) on page 19.

This installation procedure is for installing Release 6.1.1 of tableBASE without the optional VTS Manager product. Use this procedure if you are installing tableBASE for the first time. If you are upgrading from a previous release of tableBASE, a simpler upgrade procedure can be found in the *tableBASE Release Notes*.

- [“Installation procedure for Release 6.1.1 with VTS Manager”](#) on page 21.

This installation procedure is for installing Release 6.1.1 of tableBASE with the optional VTS Manager product. Use this procedure if you are installing tableBASE with the VTS Manager product for the first time or if you are upgrading to Release 6.1.1 and this is the first time you will be running VTS Manager.

If you are upgrading from a previous release of tableBASE which comprised of the VTS Manager product, a simpler upgrade procedure can be found in the *tableBASE Release Notes*.

- [“Installation procedure for VTS Manager on existing Release 6.1.1”](#) on page 23.

This installation procedure is for installing the VTS Manager product on an existing Release 6.1.1 of tableBASE. Use this procedure if you are already running Release 6.1.1 in your production environment and only wish to install VTS Manager.

This procedure will allow you to remain on Release 6.1.1 and run the new VTS Manager product. You will not be required to install the tableBASE base component or any other tableBASE products.

Installation procedure for Release 6.1.1 without VTS Manager

This installation procedure is for installing Release 6.1.1 of tableBASE without the optional VTS Manager product.

For more information on when you should use this procedure, refer to [“Description of Installation Procedures”](#) on page 18.

Your installation media contains the required load modules and datasets. The following describes the process of installing these files.

Step 1—Prepare for installation

Before you begin the installation process, you should understand the required tasks, and plan your dataset naming conventions. For more information, see [“Step 1: Prepare for tableBASE installation”](#) on page 25.

Step 2—Obtain installation files from installation media

Typically, the product is installed from the product CD:

- For CD installation, see [“Step 2: Obtain installation files from installation media”](#) on page 28.

Note: If you are downloading the product via FTP, please contact DataKinetics Technical Support: 1-613-523-5588.

Step 3—Install tableBASE PC Server (required)

Go to the tableBASE PC Server installation procedure: [“Steps to install the tableBASE PC Server”](#) on page 31.

Step 4—Install the tableBASE Batch Interface (required)

Go to the MVS Batch Interface installation procedure: [“Steps to install tableBASE Batch”](#) on page 48.

Step 5—Install the VTS product (optional)

Go to the tableBASE Installation procedure: [“Steps to install the VTS product”](#) on page 53.

Step 6—Install the tableBASE CICS Interface (optional)

Go to the optional interface installation procedure: [“Steps to install the CICS Interface product”](#) on page 67.

Step 7—Install the tableBASE IMS TM Interface (optional)

Go to the optional interface installation procedure: [“Steps to install the IMS/TM Interface product”](#) on page 76.

Step 8—Modify tableBASE parameters for DB2 stored procedures (optional)

Go to the optional interface installation procedure: [“Steps for DB2 Stored Procedure support”](#) on page 79.

Step 9—Install the tablesONLINE/CICS application (optional)

Go to the optional applications installation procedure: [“Steps to install tablesONLINE/CICS”](#) on page 81.

Step 10—Install the tablesONLINE/ISPF application (optional)

Go to the optional applications installation procedure: [“Steps to install tablesONLINE/ISPF”](#) on page 91.

Step 11—Post-installation

At this point, all installation steps have been completed.

Your next step is to perform configuration, initialization and administration tasks. See the *tableBASE Administration Guide* for further information.

Installation procedure for Release 6.1.1 with VTS Manager

This installation procedure is for installing Release 6.1.1 of tableBASE with the optional VTS Manager product.

For more information on when you should use this procedure, refer to [“Description of Installation Procedures”](#) on page 18.

Your installation media contains the required load modules and datasets. The following describes the process of installing these files.

Step 1—Prepare for installation

Before you begin the installation process, you should understand the required tasks, and plan your dataset naming conventions. For more information, see [“Step 1: Prepare for tableBASE installation”](#) on page 25.

Step 2—Obtain installation files from installation media

Typically, the product is installed from the product CD:

- For CD installation, see [“Step 2: Obtain installation files from installation media”](#) on page 28.

Note: If you are downloading the product via FTP, please contact DataKinetics Technical Support: 1-613-523-5588.

Step 3—Install tableBASE Servers for VTS Manager (required)

Go to the installation procedure: [“Steps to install the tableBASE Servers for VTS Manager”](#) on page 38.

Step 4—Install tableBASE Batch Interface (required)

Go to the MVS Batch Interface installation procedure: [“Steps to install tableBASE Batch”](#) on page 48.

Step 5—Install the VTS product (required)

Go to the tableBASE Installation procedure: [“Steps to install the VTS product”](#) on page 53.

Step 6— Install VTS Manager (required)

Go to the installation procedure: [“Steps to install VTS Manager”](#) on page 57.

Step 7—Install the tableBASE CICS Interface (optional)

Go to the optional interface installation procedure: [“Steps to install the CICS Interface product”](#) on page 67.

Step 8—Install the tableBASE IMS TM Interface (optional)

Go to the optional interface installation procedure: [“Steps to install the IMS/TM Interface product”](#) on page 76.

Step 9—Modify tableBASE parameters for DB2 stored procedures (optional)

Go to the optional interface installation procedure: [“Steps for DB2 Stored Procedure support”](#) on page 79.

Step 10—Install the tablesONLINE/CICS application (optional)

Go to the optional applications installation procedure: [“Steps to install tablesONLINE/CICS”](#) on page 81.

Step 11—Install the tablesONLINE/ISPF application (optional)

Go to the optional applications installation procedure: [“Steps to install tablesONLINE/ISPF”](#) on page 91.

Step 12—Post-installation

At this point, all installation steps have been completed.

Your next step is to perform configuration, initialization and administration tasks. See the *VTS Manager Administration Guide* and the *tableBASE Administration Guide* for further information.

Installation procedure for VTS Manager on existing Release

6.1.1

This installation procedure is for installing the VTS Manager product on an existing tableBASE release.

For more information on when you should use this procedure, refer to [“Description of Installation Procedures”](#) on page 18.

Your installation media contains the required load modules and datasets. The following describes the process of installing these files.

Step 1—Obtain installation files from installation media

Typically, the product is installed from the product CD:

- For CD installation, see [“Step 2: Obtain installation files from installation media”](#) on page 28.

Note: If you are downloading the product via FTP, please contact DataKinetics Technical Support: 1-613-523-5588.

Step 2—Install tableBASE Servers for VTS Manager (required)

Go to the installation procedure: [“Steps to install the tableBASE Servers for VTS Manager”](#) on page 38.

Step 3— Install VTS Manager (required)

Go to the installation procedure: [“Steps to install VTS Manager”](#) on page 57.

Step 4—Post-installation

At this point, all installation steps have been completed.

Your next step is to perform configuration, initialization and administration tasks. See the *VTS Manager Administration Guide* and the *tableBASE Administration Guide* for further information.

2

Preparing for Installation

To prepare for installing tableBASE and any of its associated products, perform the steps below.

Step 1: Prepare for tableBASE installation

Before starting the installation procedure, you should be aware of the essential and recommended planning steps — see the overview below, and the recommended preparations.

Installation overview

Here is an overview of the tableBASE installation process:

1. You will copy datasets from a CD or FTP site.
2. For each product that you have licensed, you will:
 - a. set site-specific system naming-convention defaults for:
 - i. load library
 - ii. tableBASE libraries
 - iii. sample libraries
 - iv. VTS Manager catalogs
 - v. VTS-TSR image datasets
 - b. tailor the product by performing some assemblies, link-edits, and moving of datasets
 - c. run a test procedure to verify a successful installation
 - d. update your user documentation to reflect local naming-convention defaults

Recommended preparations

Before starting the installation process, consider the following:

- **Dataset usage** — Most of the datasets on the distribution media are for access by all of your users, (load modules would fall into this category). However, some datasets are for use only by the system installer or system administrator. The following sections outline the various uses for the datasets that will be installed. Be aware that the distribution media does not segregate them: i.e., all JCL procedures are in one dataset. Therefore, if you want to prevent or limit access to any of these procedures, you may choose to place certain members on different libraries, perhaps with different levels of access.
- **Dataset naming conventions** — You will have to plan your site's actual naming conventions, taking into account such things as: site standards, RACF (or other) access rules, and required DASD and VSAM volume locations. If you are upgrading, you likely have already established naming conventions. However, it may still be valuable to consider using a unique naming convention for tableBASE, so that it can be distinguished from previous installations.

Note: tableBASE documentation, and any subsequently distributed service packs, refer to the datasets as we have named them. Thus, the administrator may be asked to apply a service pack to:

your.prefix.LOAD

You must be able to identify your equivalent of it in order to apply the service pack.

If you are upgrading to a new release you may have already established naming conventions. However, it may still be valuable to consider using a unique naming convention for this release so it can be distinguished from previous releases in your applications.

- **DASD volumes**—you may have to decide on specific DASD volumes to suit your site standards for installed software.
- **Authority considerations (for example, RACF)**—you may also need to control user access to the various tableBASE datasets. Keep in mind that the RACF restrictions on tableBASE libraries do not apply to tables once they are opened in a TSR. tableBASE provides other mechanisms to control access to tables in a TSR. For more information, see the *tableBASE Administration Guide*.
- **Applicability of special notes and restrictions to your site**—what are the special features of your site that need to be considered during the installation?

tableBASE Parameters

Tailor the tableBASE software for optimal performance in both batch and online environments by setting execution-time parameters and switches. The parameters are listed in [Appendix A on page 95](#). Chapters 3 through 9 provide details of their application

to a particular tableBASE facility. For usage details, see the Parameters index in the *tableBASE Administration Guide*.

Default execution-time parameters can be modified by changing the default values, or, by specifying them at execution time.

To modify the default values for a specific interface, change the TBOPTGEN macro operands in the appropriate member of the TBASE.SRC dataset, reassemble and relink it. Models for reassembling and relinking are supplied in the TBASE.CNTL partitioned data set. These procedures are described in the following chapters.

At execution time, the default values can be overridden by specifying them in a dataset defined by DDNAME TBOPT. For details about coding parameters in the TBOPT dataset, see the Parameters index in [Appendix A on page 95](#).

Enqueues

Integrity of tableBASE is achieved by issuing enqueue requests. The major enqueue name is TBLBASE. The scope of the enqueues is SYSTEMS.

An installation's modifications to Global Resource Serialization (GRS) PARMLIB controls must not prevent the propagation of the major enqueue name throughout the complex. Failure to adhere to this can result in the corruption of the datasets on DASD shared between several MVS images running tableBASE.

Since these requirements are beyond the scope of the tableBASE software, you must ensure that these conditions are met if the same datasets are to be shared among separate copies of the operating system.

Proceed to [“Step 2: Obtain installation files from installation media”](#) on page 28.

Step 2: Obtain installation files from installation media

This section assumes that you have already read “[Step 1: Prepare for tableBASE installation](#)” on page 25. It is highly recommended that you do this before proceeding.

The Install package can be delivered via CD, FTP or E-mail.

The Install package contains several datasets containing the licensed tableBASE products as well as jobs to allocate MVS datasets for transferring and receiving the contents on the mainframe.

In addition to the Install package, you will need the capability to:

- transfer files from the Install package to the mainframe
- use TSO and run jobs on a MVS system (z/OS)

1. Upload dataset allocation JCL and CLIST

Upload (using FTP) the JCL for an MVS dataset allocation job and a TSO RECEIVE CLIST:

- Upload files @@ALLOC and @@RECV to members of a CNTL dataset.

2. Allocate MVS datasets for uploading CD contents

Tailor and execute the allocation JCL (@@ALLOC) to create target datasets by type:

- Tailor the @@ALLOC JCL member by editing the JOB statement, selecting the standard disk unit (for example, “SYSDA”) and the desired prefix for the datasets. When it is executed, two sets of datasets will be allocated. Those of the form YourPrefix.xxxx are your final target datasets for tableBASE software. Those of the form YourPrefix.xxxx.XMIT are for intermediate datasets.
- Execute JCL @@ALLOC.

3. Upload remaining datasets

Upload (using FTP) the rest of the files on the CD to corresponding datasets on MVS:

- Select your favorite FTP program and transfer the files as binary (BIN or IMAGE) from the CD to the pre allocated datasets. The following is an example of two different types of datasets:

```
TBSUP.CUST.XXXXCD.CNTL.XMIT      TO  your.prefix.CNTL.XMIT
TBSUP.CUST.XXXXCD.MAINLIB        TO  your.prefix.MAINLIB
```

4. Reformat uploaded datasets to MVS Library format

The datasets with the suffix XMIT must be reformatted to MVS Library format using the TSO RECEIVE command:

- Tailor and execute the @@RECV CLIST uploaded in step 1 to reformat the datasets with XMIT suffix to PDSEs.

OR

- Use the following RECEIVE and DA commands to reformat each dataset; for example:
 - At the TSO prompt, or using ISPF option 6, enter:
RECEIVE INDA (your.prefix.CNTL.XMIT)
 - At the prompt enter:
DA('your.prefix.CNTL')
- When this is done the datasets with the XMIT suffix may be deleted.

Proceed to [“Step 3: Verify recommended security considerations”](#) on page 30.

Step 3: Verify recommended security considerations

After obtaining your tableBASE datasets from CD, the following datasets should be available depending on the products which your site has licensed. All datasets should be available for update by the installer(s) and those who apply product updates.

Note: Depending on the tableBASE products that your site has licensed, you may not have all of the following datasets.

The following datasets should be available for read to everyone, but include members which only the administrator would use:

```
your.prefix.CNTL
your.prefix.LOAD
your.prefix.TBSYSLB
```

If **your.prefix.LOAD** is not to be APF-authorized, the following dataset must be created and authorized for the PC Server jobs, the VTS Agent job and VTS Manager jobs as required:

```
your.prefix.AUTHLIB
```

The following datasets, if they are present, are application datasets and should be available for write to everyone as it is intended for update by users of tableBASE:

```
your.prefix.MAINLIB
your.prefix.TBACTLB
your.prefix.TBAPPLB
your.prefix.TBDICLB
```

If you are licensed for tablesONLINE/ISPF, read access is also required for the following datasets:

```
your.prefix.CLIST
your.prefix.MESSAGES
your.prefix.PANELS
your.prefix.SKELETON
your.prefix.TABLE
```

The following datasets are used only by the system installer or system administrator:

```
your.prefix.SRC
your.prefix.UPGRDLB
```

Note: To perform configuration, initialization and administration tasks, see the *tableBASE Administration Guide* and the *VTS Manager Administration Guide*.

3

Installing the tableBASE PC Server

This chapter describes the procedure for installing the tableBASE PC Server. If you are installing the VTS Manager product, you should not install this PC Server; instead install the tableBASE Servers for VTS Manager.

Version 6 introduces enhanced abend recovery mechanisms and other features which are implemented through Program Call (PC) instructions. The PC Server must be initiated, and run 24x7.

The PC Server must run from an APF-authorized library and should be running at all times, starting before the first use of tableBASE in the MVS image. It is recommended that it be set up as a started task and initiated at system initialization.

Note: Before you install this component, review the installation road-map in [Chapter 1 “Installation overview”](#) on page 17 to determine which set of installation procedures you should follow.

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

Steps to install the tableBASE PC Server

With all members and datasets properly acquired from the installation media, you can now proceed to install the tableBASE PC Server on your system. This will consist of several steps:

1. “[Step 1—Move load modules to authorized load library](#)”
2. “[Step 2—Modify PROC and Install in PROCLIB](#)”
3. “[Step 3—Shutdown the tableBASE Servers for VTS Manager](#)”
4. “[Step 4—Start the tableBASE PC Server](#)”

The JCL members for this step can be found in **your.prefix.CNTL**.

To proceed with the installation, perform the steps below.

Step 1—Move load modules to authorized load library

If `your.prefix.LOAD` is already authorized, skip this step.

- Modify the CNTL member AUTHLIB to conform to your specific environment (see [Figure 3-1](#)):

Specifically:

- dataset names in DD statements TBASE and AUTHLIB
- see notes below.

Moving load modules

```

/* INSERT YOUR JOB CARD HERE
/*
/******
/* COPIES TABLEBASE MODULES TO AUTHLIB
/******
/*
//IEBCOPY EXEC PGM=IEBCOPY
//TBASE DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD
//AUTHLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.AUTHLIB
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
COPY INDD=( (TBASE,R) ),OUTDD=AUTHLIB
S M=(DK1PCSRV,DK1TPCRM)
/*

```

Figure 3-1: Sample AUTHLIB JCL

Note: You may require the assistance of a systems programmer for this step.

Note: AUTHLIB contains member names required by the tableBASE PC Server that must reside in an authorized load library.

- Execute AUTHLIB.
(Confirm successful return code.)

Step 2—Modify PROC and Install in PROCLIB

The PROC shown in this section is intended to be used as a started task for the PC Server and should reside in PROCLIB. It is:

- DK1PCSRV

A. Modify DK1PCSRV PROC

- Modify the sample PROC DK1PCSRV to conform to your specific environment (see [Figure 3-2](#)):

Specifically:

- dataset name in STEPLIB statements
- see note, below.

Modify DK1PCSRV PROC

```
//DK1PCSRV PROC
//*
//*****
//*   RUNS THE TABLEBASE PC (PROGRAM CALL) SERVER
//*****
//*
//DK1PCSRV EXEC PGM=DK1PCSRV,TIME=1440
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.AUTHLIB
//SYSUDUMP DD SYSOUT=*
//*
```

Figure 3-2: Sample PROC DK1PCSRV JCL

Note: The dataset names in PROC DK1PCSRV(above), originate from those defined earlier:

- i. For DD STEPLIB, use the dataset name specified in the AUTHLIB statement of CNTL member AUTHLIB (see [“Step 1—Move load modules to authorized load library”](#) on page 32).

Caution: If you change the PROC name for DK1PCSRV, ensure that the same name is used to start up the PC Server in the next step.

Step 3—Shutdown the tableBASE Servers for VTS Manager

If you are currently running the Release 6.1.1 tableBASE Servers for VTS Manager in this environment, you must shut down the tasks that are running as well as the tableBASE Servers for VTS Manager in order to start up the Release 6.1.1 tableBASE PC Server.

If you are not already running tableBASE Release 6.1.1, please omit this step and proceed to [“Step 4—Start the tableBASE PC Server”](#) on page 35.

To shut down the tasks and tableBASE Servers for VTS Manager for Release 6.1.1, perform the steps below.

Shutdown tasks running under Release 6.1.1

Shutdown all applications, regions and VTS's that are running under tableBASE Release 6.1.1, i.e. all tasks that are pointing to the load modules for tableBASE Release 6.1.1.

To shut down VTS's that are running, use the following command:

```
P jobname
```

Shutdown the tableBASE Servers for VTS Manager

When all applications, regions and VTS's have successfully been shutdown, bring down the tableBASE Servers for VTS Manager using the following commands.

Shut down the *compat* VTS Group Manager Server:

```
P DK1TPVM
```

Shut down the VTS Manager Server:

```
P DK1TPM
```

Cleanup the tableBASE Release 6.1.1 Pointers

The pointers which were used under the existing installation must be cleaned up in order to start up the tableBASE PC Server.

- Modify the CNTL member, TERMTTPM, to conform to your specific environment (see [Figure 3-3](#)).

Specifically:

- modify dataset name in DD statement STEPLIB

Cleaning up the Release 6.1.1 Pointers

```
//TERMTPM JOB 0, 'TERMINATE TPM'  
//* USE FORCE TO REMOVE AND CLEANUP TPM/TPVM/VTS  
//*  
//TERM EXEC PGM=DK1PTERM, PARM='V611'  
//STEPLIB DD DSN=*YOUR.PREFIX*.TBASE.AUTHLIB, DISP=SHR  
//SYSABEND DD SYSOUT=*  
//DK1SNAP DD SYSOUT=*  
//DK1MSG DD SYSOUT=*
```

Figure 3-3: Sample TERMTPM JCL

- Execute TERMTPM.
(Confirm successful return code.)

Step 4—Start the tableBASE PC Server

- Issue the MVS command S DK1PCSRV
or, alternatively, create your own job to run PROC DK1PCSRV and submit it.
See [Figure 3-4](#) for sample job output.

Sample DK1PCSRV job output:

```
$HASP373 DK1PCSRV STARTED  
IEF403I DK1PCSRV - STARTED - TIME=11.11.15  
DK100800I tableBASE PC Server V611 available
```

Figure 3-4: Sample job output

4

Installing the tableBASE Servers for VTS Manager

This chapter describes the procedure for installing the tableBASE Servers for running the VTS Manager product. If you are not installing the VTS Manager product, you should not install these Servers; instead install the tableBASE PC Server.

Version 6 introduces enhanced abend recovery mechanisms and other features which are implemented through Program Call (PC) instructions. The tableBASE Servers for VTS Manager must be initiated and run 24x7.

The tableBASE Servers for VTS Manager consists of a minimum of two tasks which will be started up when the server job is run. These two tasks are for the VTS Manager Server and for the *compat* VTS Group Manager Server (the equivalent of the tableBASE PC Server) and are described in the procedures that follow.

These servers must run from an APF-authorized library and should be running at all times, starting before the first use of tableBASE in the MVS image. It is recommended that the server job be set up as a started task and initiated at system initialization.

Note: Before you install this component, review the installation road-map in [Chapter 1 “Installation overview”](#) on page 17 to determine which set of installation procedures you should follow.

For more information on the tableBASE Servers for VTS Manager, please see the *tableBASE Administration Guide* and the *VTS Manager Administration Guide*.

Steps to install the tableBASE Servers for VTS Manager

These Servers are required to run tableBASE Release 6.1.1 with VTS Manager.

With all members and datasets properly acquired from the installation media, you can now proceed to install the tableBASE Servers for VTS Manager on your system. This will consist of several steps:

1. “Step 1—Move load modules to authorized library”
2. “Step 2—Configure the VTS Manager catalog”
3. “Step 3—Configure the compat VTS Group Manager catalog”
4. “Step 4—Modify PROCs and install in PROCLIB”
5. “Step 5—Shutdown tableBASE PC Server for Release 6.1.1”
6. “Step 6—Start the tableBASE Servers for VTS Manager”

The JCL members for this step can be found in **your.prefix.CNTL**.

To proceed with the installation, perform the steps below.

Step 1—Move load modules to authorized library

If your `.prefix.LOAD` is already authorized, skip this step.

- Modify the CNTL member COPYAUTH to conform to your specific environment (see [Figure 4-1](#)).

Specifically:

- modify dataset names in DD statements TBASELOD and AUTHLIB
- delete the last S (SELECT) statement if you have previously installed Release 6.1.1
- see notes below.

Moving load modules

```
//COPYAUTH JOB 0, 'POPULATE AUTHLIB'
//*
/*****
/* COPY VTS Manager MODULES TO AUTHLIB
/*****
/*
//IEBCOPY EXEC PGM=IEBCOPY
//TBASELOD DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <=====
//AUTHLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.AUTHLIB <=====
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
COPY INDD=( (TBASELOD,R) ),OUTDD=AUTHLIB
* COPY VTS MANAGER TPM, TPVM, VTS MODULES
S M=(DK1P1MGR,DK1P1RM,DK1P2MGR,DK1P2RM,DK1V3MGR,DK1PTERM)
S M=(DK1P1PM,DK1PMSG,DK1P0000,DK1P1001,DK1PBASE,DK1V3JF)
*
* DELETE THE STATEMENT BELOW IF YOU HAVE PREVIOUSLY INSTALLED V611
*
S M=(DK1TPCRM) REMOVE IF V611 INSTALLED <=====
/*
```

Figure 4-1: Sample COPYAUTH JCL

Note: You may require the assistance of a systems programmer for this step.

Note: COPYAUTH contains member names required by the tableBASE Servers for VTS Manager and the VTS Manager product that must reside in an authorized load library.

Note: If you do not delete the last S (Select) statement in the COPYAUTH job before executing it, the job will complete with return code 4 because member DK1TPCRM will not be supplied if you have previously installed Release 6.1.1.

- Execute COPYAUTH.
(Confirm successful return code.)

Step 2—Configure the VTS Manager catalog

- Modify the CNTL member CNFGTPM to conform to your specific environment (see [Figure 4-2](#)).

Specifically:

- modify dataset names in the DEFCAT and CNFGCAT steps
- see notes, below.

Configuring the TPM catalog

```
//CNFGTPM JOB 0,'INIT TPM LDS'
//*****
//* TPM CATALOG INITIALIZATION
//*   DELETE AND DEFINE TPM CATALOGUE VSAM LDS
//*   CONFIGURE TPM CATALOGUE
//*****
//DEFCAT   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN   DD *
          DELETE *your.prefix*.TPMCAT CLUSTER
          DEFINE CLUSTER(NAME(*your.prefix*.TPMCAT) -
                        RECORDS(032) LINEAR SHR(3,3))
          SET MAXCC = 0
/*
//CNFGCAT   EXEC PGM=DK1P1TPC
//STEPLIB  DD DSN=*your.prefix*.TBASE.LOAD,DISP=SHR
//TPMCAT   DD DSN=*your.prefix*.TPMCAT,DISP=SHR
//SYSUDUMP DD SYSOUT=*
//DK1MSG   DD SYSOUT=*
//*
```

Figure 4-2: Sample CNFGTPM JCL

Note: STEPLIB must be a standard tableBASE load library.

Note: The dataset name defined in the DEFCAT step and specified in DD statement TPMCAT will be used again in “[A. Modify DK1TPM PROC](#)” on page 42.

- Execute CNFGTPM.
(Confirm successful return code.)

Step 3—Configure the *compat* VTS Group Manager catalog

- Modify the CNTL member CNFGTPVM to conform to your specific environment (see [Figure 4-3](#)).

Specifically:

- modify dataset name in CRTECAT step
- see note, below.

Configuring the *compat* catalog

```
//CNFGTPVM JOB 0, 'ALLOC TPVM CAT'
//*****
//*      ALLOCATE TPVM CATALOG:
//*      "COMPAT" CATALOGUE IS REQUIRED FOR STANDARD VTS AGENTS
//*****
//CRTECAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
          DELETE *your.prefix*.TPVMCAT.COMPAT CLUSTER
          SET MAXCC = 0
          DEFINE CLUSTER(NAME(*your.prefix*.TPVMCAT.COMPAT) -
                        RECORDS(032) LINEAR SHR(3,3))
/*
```

Figure 4-3: Sample CNFGTPVM JCL

Note: The dataset name specified in the DELETE and DEFINE statements will be used again in “[B. Modify DK1TPVM PROC](#)” on page 42.

Note: The mnemonic used for *VTS Group Manager* is *TPVM*.

- Execute CNFGTPVM.
(Confirm successful return code.)

Step 4—Modify PROCs and install in PROCLIB

The PROCs shown in this section are intended to be used as cataloged procedures for the PC Server, and should reside in PROCLIB. They are:

- DK1TPM <==== this is for the VTS Manager
- DK1TPVM <==== this is for the *compat* VTS Group Manager

A. Modify DK1TPM PROC

- Modify the sample PROC DK1TPM to conform to your specific environment (see [Figure 4-4](#)).

Specifically:

- modify dataset names in DD statements STEPLIB and TPMLDS
- see note, below.

Modify DK1TPM PROC

```
//DK1TPM PROC START=' '
//* VTS MANAGER PROC
//*   REQUIRED FOR ALL VTS MANAGER PROCESSING
//*   START WITH MVS 'S DK1TPM' AT SYSTEM INITIALIZATION
//*   USES TPMLDS01 CONFIGURED IN JOBSTREAM CNFGTPM
//STEP1 EXEC PGM=DK1P1MGR,TIME=1440,PARM=&START
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.AUTHLIB <=====
//TPMLDS01 DD DSN=*YOUR.PREFIX*.TPMCAT,DISP=SHR
//SYSABEND DD SYSOUT=*
//DK1SNAP DD SYSOUT=*
//DK1MSG DD SYSOUT=*
```

Figure 4-4: Sample PROC DK1TPM JCL

Note: The dataset names in PROC DK1TPM (above), originate from those defined earlier:

- i. For DD TPMLDS01, use the dataset name specified in the DELETE, DEFINE and DEFCAT statements of CNTL member CNFGTPM (see [“Step 2—Configure the VTS Manager catalog”](#) on page 40).
- ii. For DD STEPLIB, use the dataset name specified in the AUTHLIB statement of CNTL member COPYAUTH (see [“Step 1—Move load modules to authorized library”](#) on page 39).

Caution: If you change the PROC name for DK1TPM, ensure that the same PROC name is used to start up the TPM in [“Step 6—Start the tableBASE Servers for VTS Manager”](#) on page 45.

B. Modify DK1TPVM PROC

- Modify the sample PROC DK1TPVM to conform to your specific environment (see [Figure 4-5](#)).

Specifically:

- modify dataset name for LDSDSN in PROC statement
- modify dataset name in STEPLIB statement
- see note, below.

Modify DK1TPVM PROC

```
//DK1TPVM PROC LDSDSN='your.prefix.TPVMCAT.COMPAT'
//*****
//* VTS GROUP MANAGER PROC
//*     DEFAULTS TO COMPATIBILITY VTS GROUP MANAGER ("COMPAT")
//*****
//TPVM     EXEC PGM=DK1P2MGR,TIME=1440
//STEPLIB DD DSN=*your.prefix*.TBASE.AUTHLIB,DISP=SHR
//TPVMLDS DD DSN=&LDSDSN,DISP=SHR
//SYSABEND DD SYSOUT=*
//DK1SNAP DD SYSOUT=*
//DK1MSG  DD SYSOUT=*
```

Figure 4-5: Sample PROC DK1TPVM JCL

Note: The dataset names in PROC DK1TPVM (above), originate from those defined earlier:

- For LDSDSN in the PROC statement, use the dataset name specified in the DELETE and DEFINE statements of CNTL member CNFGTPVM (see “[Step 3—Configure the compat VTS Group Manager catalog](#)” on page 41).
- For DD STEPLIB, use the dataset name specified in the AUTHLIB statement of CNTL member COPYAUTH (see “[Step 1—Move load modules to authorized library](#)” on page 39).

Caution: Do not change the PROC name. This PROC name is used to start up the *compat* VTS Group Manager; if it is changed, the *compat* VTS Group Manager will not start up when the TPM is started.

C. Copy all modified PROCs to the system PROCLIB

- Copy the following PROCs to the system PROCLIB:
 - DK1TPM
 - DK1TPVM

Note: You may require the assistance of a systems programmer for this step.

Step 5—Shutdown tableBASE PC Server for Release 6.1.1

If you are currently running the tableBASE PC Server for Release 6.1.1 in this environment, you must shut down the tasks that are running as well as the tableBASE PC Server in order to start up the Release 6.1.1 tableBASE Servers for VTS Manager.

If you are not already running tableBASE Release 6.1.1, please omit this step and proceed to [“Step 6—Start the tableBASE Servers for VTS Manager”](#) on page 45.

To shut down the tasks and tableBASE PC Server for Release 6.1.1, perform the steps below.

Shutdown tasks running under Release 6.1.1

Shutdown all applications, regions and VTS-TSRs that are running under tableBASE Release 6.1.1, i.e. all tasks that are pointing to the load modules for tableBASE Release 6.1.1.

To shut down VTS-TSRs that are running, use the following command:

```
P jobname
```

Shutdown the tableBASE PC Server

When all applications, regions and VTS-TSRs have successfully been shutdown, bring down the tableBASE PC Server using the following command:

```
P DK1PCSRV
```

Cleanup the tableBASE Release 6.1.1 Pointers

The pointers which were used under the existing installation must be cleaned up in order to start up the tableBASE Servers for VTS Manager.

- Modify the CNTL member, TERMTTPM, to conform to your specific environment (see [Figure 4-6](#)).

Specifically:

- modify dataset name in DD statement STEPLIB

Cleaning up the Release 6.1.1 Pointers

```
//TERMTPM JOB 0, 'TERMINATE TPM'
//* USE FORCE TO REMOVE AND CLEANUP TPM/TPVM/VTS
//*
//TERM EXEC PGM=DK1PTERM, PARM='V611'
//STEPLIB DD DSN=*YOUR.PREFIX*.TBASE.AUTHLIB, DISP=SHR
//SYSABEND DD SYSOUT=*
//DK1SNAP DD SYSOUT=*
//DK1MSG DD SYSOUT=*
```

Figure 4-6: Sample TERMTPM JCL

- Execute TERMTPM.
(Confirm successful return code.)

Step 6—Start the tableBASE Servers for VTS Manager

- Issue the MVS command S DK1TPM
or, alternatively, create your own job to run PROC DK1TPM and submit it.

Two jobs should start up: DK1TPM for the VTS Manager and DK1TPVM for the *compat* VTS Group Manager. See [Figure 4-7](#) for sample JES messages that will be displayed on successful startup.

Sample DK1TPM JES Messages:

```
DK1P02001I: VTS Manager - Initializing Starting
DK1P02009I: TPM start parm used: PRISTINE
DK1P02133I: LXRES value being used = 00002A00
DK1P02175I: TPM is processing a request to DEFINE TPVM compat
DK1P02603I: VTS Manager is Licensed to
DK1P02601I: your company name
DK1P02604I: Your VTS Manager License expires on yyyy/mm/dd
DK1P02175I: TPM is processing a request to START TPVM compat
DK1P02044I: VTS Manager for TB V611 Initialization Completed
```

Sample DK1TPVM JES Messages:

```
DK1P02200I: Initializing TPVM COMPAT
DK1P02201I: TPVM COMPAT is running
```

Figure 4-7: tableBASE Servers for VTS Manager sample JES messages

5

Installing tableBASE MVS batch

This chapter describes basic steps required to install the tableBASE batch component.

The tableBASE MVS batch software is the tableBASE base product. Install this component first, then install any optional components and products.

Note: Before you install this component, review the installation road-map in [“Installation overview”](#) on page 17 to determine which set of installation procedures you should follow.

To install the tableBASE batch interface, perform the steps starting on the following page.

Steps to install tableBASE Batch

With all members and datasets properly acquired from the installation media, you can now proceed to install the batch component on your system. This will consist of eight sub-steps:

- Modify default parameters (optional)
- Copy tableBASE load modules
- Run tableBASE above or below the line.
- Convert tableBASE libraries to Version 6 format
- Point TBDRIVER CLIST to newly installed libraries
- Set linkage editor to use the Version 6 TBLBASE API
- Convert BDAM table libraries to VSAM (optional)
- Confirm successful install

1. Modify default parameters (optional)

If desired, modify the default parameters (see “[tableBASE Parameters](#)” on page 26) for operating tableBASE in a batch environment. If the default parameters are acceptable or you use the Override feature provided through the TBOPT dataset, no additional work is required in this step.

For parameter descriptions, see [Appendix A on page 95](#). The default values for this component are listed below and are also defined in:

your.prefix.SRC (DK1T1134)

Default batch parameters

```
TBOPTGEN BASE,  
    ALLOW_TB_LIB_IO=Y,  
    DATERTNX=N,  
    FGDELIM=' '*',  
    HASH_HI_DEN_LIM=900,  
    HASH_LOW_DEN_LIM=600,  
    LISTOPTIONS=N,  
    LOCKTIMER=0,  
    LOCKTIMEWTO=30,  
    MAXNMTAB=0,  
    MTRETAIN=N,  
    MULTITASKING=N,  
    OVERRIDES=YYYYYYYY,  
    STROBE=0,  
    SWITCHES=YNYNNNN,  
    TABLEWAITRC=0,  
    TABLEWAITWTO=30,  
    TSR_ALGORITHM=D,
```

```
TSR_WARNING_FREQ=1,  
TSR_WARNING_PCT=85,  
TSRSIZE=10M,  
USEREXITS=N,  
VTSFIRST=,  
VTSLAST=,  
VTSNAME=,  
VTSPREFIX=VTS: ,  
ZEROROWS=Y,  
ML=MAINLIB
```

Note: Three other parameters which are not listed above are LDS, TPVM and TSRACCESS. These parameters are only relevant if you have licensed the VTS Manager product. We recommend that you keep the defaults for these as delivered and use the Override feature provided through the TBOPT dataset.

Note: The Status Switch setting (SWITCHES) and Overrides (OVERRIDES) are unique to this interface.

To make changes to the installation parameters:

- Modify: **your.prefix.SRC (DK1T1134)**.
- Reassemble and link-edit the module into the tableBASE batch interface program DK1BBASE using the following as a model:
your.prefix.CNTL (ALT1134)

2. Copy tableBASE load modules

If it is necessary to copy the tableBASE load modules to another system library, use the IBM utility IEBCOPY to do so.

3. Run tableBASE interface above the line

For backward compatibility purposes, the tableBASE interface, DK1TCALL, and all its aliases, have been created to run in 24-bit mode. If you require it to run in 31-bit mode, you need to relink DK1TCALL and all its aliases.

To do this:

- Modify and execute job **your.prefix.CNTL (RLK31ANY)**.

Note: The above job will replace DK1TCALL and all its aliases in the library specified under the SYSLMOD DD. If you require the 24-bit version of DK1TCALL and its aliases, make a copy of it to another library before the relink.

Note: Aliases for DK1TCALL are TBLBASE, TBASE, TBCALL, TBBAFN01, TBROOT, MAINCALL, MAINROOT, TBCALLC, TBASEC, TBCALLI, TBROOTI, TBCALLV and TBASEV.

4. Convert tableBASE libraries to Version 6 format

Conversion of tableBASE libraries is a necessary step in running tableBASE version 6. Before beginning this step, see [“Converting tableBASE libraries”](#) on page 111.

Note: This step is not required if you are already running a prior release of Version 6.

If you have previously installed the Library Bridge and have converted your libraries to Bridge format, you have two options. You can convert your Bridge libraries to the Version 6 format or you can continue to use your Bridge libraries.

Note: Release 6.0.2 and above will be able to access your Bridge libraries. However, DataKinetics Ltd. cannot guarantee support of Bridge libraries in future versions of tableBASE. Any modifications or enhancements to tableBASE library processing will only be made to the Version 6 format libraries.

The following steps for converting tableBASE libraries to the Version 6 format applies to customers who are installing Version 6 for the first time and are not using the Library Bridge, and for customers who are using the Library Bridge and would like to convert their Bridge libraries.

Note: TBSYSLB must not be converted— a new version is supplied with every release- the new version must be used.

1. To convert your tableBASE libraries to Version 6 Transition Libraries, modify and execute one of the jobs CNVBTO6T (BDAM) or CNVVTO6T (VSAM) supplied in **your.prefix.CNTL** for each tableBASE library that needs to be converted.
2. To convert your tableBASE libraries to Version 6 libraries, modify and execute one of the jobs CNVBTO6 (BDAM) or CNVVTO6 (VSAM) supplied in **your.prefix.CNTL** for each tableBASE library that needs to be converted.

For more information on converting tableBASE libraries and samples of other conversion jobs, see [“Converting tableBASE libraries”](#) on page 111 and [“JCL members”](#) on page 139.

5. Point TBDR CLIST to newly installed libraries

- Customize the REXX TBDR CLIST that you use to access the TBDRIVER program through TSO. It is:

your.prefix.CLIST(TBDR)

- Make the following changes:

Original reference:

your.prefix.LOAD

your.prefix.TBSYSLB

Change to:

actual tableBASE load library Dataset Name

actual Dataset Name

6. Update binder procedure to use the Version 6 TBLBASE API

The **your.prefix.LOAD** library contains aliases MAINCALL, MAINROOT, TBASE, TBCALL and TBBAFN01 for prior tableBASE interfaces. These interfaces have been replaced in Version 6 but the aliases are retained for compatibility. Although with Version 6, the tableBASE interface module is renamed to DK1TCALL in Version 6, the TBLBASE alias is the standard interface name.

If you are upgrading from an earlier version of tableBASE and your applications called these earlier versions of the tableBASE interface programs, either statically or dynamically, they will continue to work without having to recompile or relink your applications.

Note: Aliases MAINEXEC for TBEXEC, TBLRT for TBACC and POWRFEAT for TBINDX are no longer supported.

If you wish to convert existing tableBASE batch applications to the TBLBASE API or to create new tableBASE applications using the TBLBASE API, then you must ensure that these applications use the following Version 6 API, and not a version in any other load library:

your.prefix.LOAD

Update the binder procedure accordingly.

7. Convert BDAM table libraries to VSAM (optional)

All tableBASE libraries that have been unloaded from the installation media have been defined with a file organization of BDAM. If you wish to convert these libraries to a VSAM file organization, customize and run the BDMTOVSM job (for each library to be converted) in:

your.prefix.CNTL

Notes:

- Not all libraries have to be converted to VSAM. You may have a mixture of BDAM and VSAM libraries.
- For VSAM library definition information, see the *tableBASE Administration Guide* (Chapter 12, Best Practices).

8. Confirm successful install

Note: Depending on whether or not you are installing VTS Manager, ensure that either the tableBASE PC Server or the tableBASE Servers for VTS Manager have been installed and are running on the same LPAR as this component. If neither of the Servers are running at the time that tableBASE software initializes, the job which issued the tableBASE call will abend and you will see the following error message in the JES LOG:

```
DK100201S tableBASE PC Server unavailable.
```

Modify and execute the batch validation job:

- Modify the job statements and names in:
your.prefix.CNTL (VALIDATE) .
- Execute this job. (Confirm success return code.)
Also, verify that your banner information is in the TBMSG output.

This job tests the command executor (TBDRIVER) and the tableBASE batch utility (TBEXEC). As the job leaves no permanent changes, it may be run at any time to confirm that the base product is operational.

6

Installing the VTS (Virtual Table Share) product

This chapter describes the Virtual Table Share (VTS) installation procedures.

Note: Before you install the DataKinetics VTS software product, review the information in [Chapter 1 “Installation overview”](#) on page 17.

Steps to install the VTS product

If you have licensed the DataKinetics VTS product, the installation media contains additional load modules and datasets. This guide describes the process of installing these additional files. For details about the implementation and operation of VTS, see the *tableBASE Administration Guide* and the *tableBASE Programming Guide*.

To proceed with the installation, perform the steps on the following pages.

Step 1—Modify default parameters (optional)

Modify the default parameters for operating the VTS Agent, if desired. If the default values are acceptable, or you wish to use the Override feature provided via the TBOPT dataset, no additional work is required in this step.

For parameter descriptions, see [Appendix A on page 95](#). The default values for this product are listed below and are also defined in:

your.prefix.SRC (DK1V1134) .

Default VTS parameters

```
TBOPTGEN VTS,  
  LISTOPTIONS=N,  
  MAXNMTAB=0,  
  MTRETAIN=N,  
  STROBE=0,  
  TSR_ALGORITHM=D,  
  TSR_WARNING_FREQ=1,  
  TSR_WARNING_PCT=85,  
  TSRSIZE=10M,  
  ZEROROWS=Y,  
  VTSNAME=DKL1
```

Note:

1. Three other parameters which are not listed above are LDS, TPVM and TSRACCESS. These parameters are only relevant if you have licensed the VTS Manager product. We recommend that you keep the defaults for these as delivered and use the Override feature provided through the TBOPT dataset.
2. In Version 5, the VTS product had its own settings for SWITCHES. In Version 6 the VTS product does not have its own SWITCHES settings; it inherits the settings from the interface corresponding to the application being run— Batch, IMS, CICS, DB2, etc. The default settings for the SWITCHES in Version 5 VTS was NNNNN, while the default settings for the Batch, IMS, CICS and DB2 interfaces differ from these values (refer to the installation chapters applicable to the interface). Therefore an application from Version 5 that may have not abended when the VTS-TSR was down (due to a 1072 error) may now abend in Version 6, unless the SWITCHES setting is changed from the default value to N. Other differing settings may cause other behavioral differences unless changed from their Version 5 default values.

To make changes to the installation parameters:

1. Modify **your.prefix.SRC (DK1V1134)** .
2. Reassemble DK1V1134 and link-edit into load module DK1VBASE. Use **your.prefix.CNTL (ALV1134)** as a model.

Step 2—Copy tableBASE load modules (optional)

The VTS Agent must be run from an APF-authorized library. If **your.prefix.LOAD** is not APF-authorized, then the member AUTHLIBV of **your.prefix.CNTL** can be used to copy modules to an authorized library (**your.prefix.AUTHLIB**). A system programmer should be able to confirm that an appropriate authorized load library is set up.

Step 3—Confirm successful VTS install

Note: Depending on whether or not you are installing VTS Manager, ensure that either the tableBASE PC Server or the tableBASE Servers for VTS Manager have been installed and are running on the same LPAR as this component. If neither of the Servers are running at the time that tableBASE software initializes, the VTS job which issued the tableBASE call will abend and you will see the following error message in the JES LOG:

```
DK100605E V611 Environment not ready; VTS stopping
```

To verify that VTS is installed correctly, perform the following steps:

1. Tailor the JCL for your environment.
2. Ensure that either the tableBASE PC Server or tableBASE Server for VTS Manager are up.
3. Start a VTS Agent (member VTSAGENT in **your.prefix.CNTL**).
 - Modify the jobstream VTSAGENT below, specifically:
 - jobname statement
 - dataset name in STEPLIB statement

Modify VTSAGENT

```
//VTSAGENT JOB 0, 'START VTS'
//*****
//*   STARTS A VTS AGENT NAMED DK1A
//*   TO STOP, ENTER MVS COMMAND "P VTSAGENT" (JOBNAME)
//*****
//VTS      EXEC PGM=DK1VAGNT,TIME=1440
//STEPLIB DD DISP=SHR,DSN=your.prefix.TBASE.AUTHLIB <=====
//SYSUDUMP DD SYSOUT=*
//TBDUMP  DD SYSOUT=*
//TBOPT   DD *
LISTOPTIONS=Y
VTSNAME=DK1A
TSREGION=50M
/*
```

- Execute VTSAGENT and confirm successful return code.
Also, check the JES log for the following message from the VTS job:

```
DK100600I tableBASE VTS DK1A initialized.
```

4. Access the sample VTS using **your.prefix.CNTL(VALIDVTS)**. Modify as required.
5. Stop the VTS Agent using MVS command:

```
"p jobname
```

7

Installing the VTS Manager product

This chapter describes the VTS Manager installation procedures. This product may only be installed with tableBASE Release 6.1.0 and above. If you are currently running tableBASE, you may also install VTS Manager by upgrading to Release 6.1.1.

Note: Before you install the VTS Manager product, review the road-map in [Chapter 1 “Installation overview”](#) on page 17 for the installation procedures to use.

Steps to install VTS Manager

If you have acquired a license for VTS Manager, you should have already installed the tableBASE Servers for VTS Manager - see [“Steps to install the tableBASE Servers for VTS Manager”](#) on page 38. You should also have installed the Batch and VTS interfaces, or be running these interfaces at Release 6.1.1, before proceeding to this step. For more details about the implementation and operation of VTS Manager, see the *VTS Manager Administration Guide*.

To perform the installation, follow the steps below:

- “Step 1—Modify PROCs and install in PROCLIB”
- “Step 2—Start the tableBASE Servers for VTS Manager”
- “Step 3—Validate the VTS Manager installation”

The JCL members for these steps can be found in **your.prefix.CNTL**.

Step 1—Modify PROCs and install in PROCLIB

The PROCs shown in this section are intended to be used as cataloged procedures, and should reside in PROCLIB. They are:

- DK1VTS <=== this allows you to start a VTS with R/W TSR access
- DK1VTSRO <=== this allows you to start a VTS with R/O TSR access

A. Modify DK1VTS PROC

- Modify the sample PROC DK1VTS to conform to your specific environment (see [Figure 7-1](#)).

Specifically:

- modify the dataset name in the STEPLIB statement (see note, below).

Modify DK1VTS PROC

```
//DK1VTS   PROC JPARM=COMPAT,LDSTSR=NULLFILE
//*****
//*   START UP A VTS WITH R/W ACCESS TO VTS-TSR
//*   DEFAULTS TO START UNDER VTS GROUP MANAGER "COMPAT"
//*****
//DK1VTS   EXEC PGM=DK1V3MGR,TIME=1440,PARM='&JPARM'
//STEPLIB  DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.AUTHLIB  <=====
//LDSTSR   DD DISP=OLD,DSN=&LDSTSR
//SYSABEND DD SYSOUT=*
//TBDUMP   DD SYSOUT=*
//DK1SNAP  DD SYSOUT=*
//DK1MSG   DD SYSOUT=*
```

Figure 7-1: Sample PROC DK1VTS JCL

Note: The dataset name in DD STEPLIB (above), originates from that defined earlier. Use the dataset name specified in the AUTHLIB statement of CNTL member COPYAUTH (see [“Step 1—Move load modules to authorized library”](#) on page 39).

Caution: The PROC name for DK1VTS is used in a subsequent validation procedure to ensure that your installation was successful. If you change the PROC name, ensure that the same name is used in the "PROC=" parameter of the DEFVTS job in [“Step 3—Validate the VTS Manager installation”](#) on page 60.

B. Modify DK1VTSRO PROC

- Modify the sample PROC DK1VTSRO to conform to your specific environment (see [Figure 7-2](#)).

Specifically:

- modify dataset name in the STEPLIB statement (see note, below).

Modify DK1VTSRO PROC

```
//DK1VTSRO PROC JPARM=COMPAT,LDSTSR=NULLFILE
//*****
//*   START UP A VTS WITH READ ONLY ACCESS TO VTS-TSR
//*   DEFAULTS TO START UNDER VTS GROUP MANAGER "COMPAT"
//*****
//DK1VTSRO EXEC PGM=DK1V3MGR, PARM=' &JPARM'
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.AUTHLIB <=====
//LDSTSR DD DISP=SHR,DSN=&LDSTSR
//SYSABEND DD SYSOUT=*
//TBDUMP DD SYSOUT=*
//DK1SNAP DD SYSOUT=*
//DK1MSG DD SYSOUT=*
```

Figure 7-2: Sample PROC DK1VTSRO JCL

Note: The dataset name in DD STEPLIB (above), originates from that defined earlier. Use the dataset name specified in the AUTHLIB statement of CNTL member COPYAUTH (see [“Step 1—Move load modules to authorized library”](#) on page 39).

Caution: The PROC name DK1VTSRO is used to define a VTS with R/O (read-only) access to the VTS-TSR. If you change the PROC name, ensure that you use the same PROC name when performing the DEFINE for the read-only VTS. For more details on how to use the DEFINE command, see the chapter on TPDRIVER in the *VTS Manager Administration Guide*.

C. Copy all modified PROCs to the system PROCLIB

- Copy the following PROCs to the system PROCLIB:
 - DK1VTS
 - DK1VTSRO

Note: You may require the assistance of a system programmer for this step.

Step 2—Start the tableBASE Servers for VTS Manager

- Start the tableBASE Servers for VTS Manager if they are not already running - see [“Step 6—Start the tableBASE Servers for VTS Manager”](#) on page 45.

Step 3—Validate the VTS Manager installation

The steps taken at the start of the installation process ([“Steps to install VTS Manager”](#) on page 57) provided you with the needed members and datasets that will be required for the testing of the VTS Manager installation.

The term *cataloged VTS* refers to a VTS-TSR that is managed by a VTS Group Manager, as opposed to being managed by a VTS Agent. A cataloged VTS is defined on a catalog that is managed by the VTS Group Manager. An LDS (Linear Dataset) is required to save the image of a cataloged VTS when the VTS is brought down.

The steps that follow validates the VTS Manager installation by defining, starting, shutting down and deleting a *cataloged VTS* on the *compat* VTS Group Manager.

1. Allocate the VTS-TSR Image dataset for a cataloged VTS

This step allocates the LDS that is used to save the VTS Image.

- Modify the CNTL member CNFGVTS (see [Figure 7-3](#)).

Specifically:

- modify dataset names in the DELETE and DEFINE statements.
(Note that this dataset is used in CNTL member DEFVTS—see [“Define a sample cataloged VTS”](#) on page 61.)

Modify CNFGVTS

```
//CNFGVTS JOB 0, 'ALLOC VTS-TSR LDS'
//*****
//* ALLOCATE VTS-TSR IMAGE DATASET:
//* RECORDS * 4K = TSRSIZE (OVERRIDES TSRSIZE IN VTS DEFINITION)
//* RECORDS VALUE WILL ROUND UP TO PHYSICAL TRACK SIZE
//* 2560 PAGES MAPS A 10M TSR
//*****
//CRTECAT EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
DELETE your.prefix.VTSIMAGE.DK1V TSA CLUSTER
SET MAXCC = 0
DEFINE CLUSTER (NAME (your.prefix.VTSIMAGE.DK1V TSA) -
RECORDS (2560) LINEAR SHR (3,3))
/*
```

Figure 7-3: Sample CNFGVTS JCL

- Execute CNFGVTS and confirm successful return code.

2. Define a sample cataloged VTS

This step defines the VTS on the *compat* VTS Group Manager catalog. These definitions can only be viewed using the SHOW command in the sample.

- Modify the CNTL member DEFVTS (see [Figure 7-5](#)).

Specifically:

- modify the dataset name in the STEPLIB statement
- modify the dataset name in parameter **LDS=** in the DEFINE statement.

Note that this dataset is allocated in CNTL member CNFGVTS—see [“Allocate the VTS-TSR Image dataset for a cataloged VTS”](#) on page 60.

Modify DEFVTS

```
//DEFVTS      JOB 0, 'DEFINE VTS'
//*****
//*          DEFINE A VTS (CATALOGUED) UNDER TPVM "COMPAT"
//*****
//DEFVTS      EXEC PGM=TPDRIVER
//STEPLIB     DD DISP=SHR,DSN=your.prefix.TBASE.LOAD
//SYSOUT      DD SYSOUT=*
//TPDRRPT    DD SYSOUT=*
//SYSUDUMP    DD SYSOUT=*
//SYSPRINT    DD SYSOUT=*
//DK1MSG     DD SYSOUT=*
//TPDRCNTL   DD *
*
*                                                    DEFINE VTS
DEFINE VTS,NAME=DK1VTSA,
          JOBNAME=DK1VTSA,PROC=DK1VTS,
          TSRSIZE=10M,
          MAXNMTAB=100,
          LDS=your.prefix.VTSIMAGE.DK1VTSA,
          DESC="SAMPLE VTS FOR INSTALLATION VERIFICATION"
*
*                                                    SHOW VTS DEFINITION
SHOW     VTS,NAME=DK1VTSA
```

Figure 7-4: Sample DEFVTS JCL

- Execute DEFVTS and confirm successful return code.

Note: Do not execute this step unless [“Allocate the VTS-TSR Image dataset for a cataloged VTS”](#) on page 60 completed successfully.

3. Start the sample cataloged VTS

This step starts the VTS that was defined in the previous step.

- Modify the CNTL member STARTVTS (see [Figure 7-6](#)).

Specifically:

- modify dataset name in the STEPLIB statement.

Modify STARTVTS

```
//STARTVTS JOB 0, 'START VTS'
//*****
//*      START A VTS (CATALOGUED) UNDER TPVM "COMPAT"
//*****
//STARTVTS EXEC PGM=TPDRIVER
//STEPLIB DD DISP=SHR,DSN=your.prefix.TBASE.LOAD
//SYSOUT DD SYSOUT=*
//TPDRRPT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DK1MSG DD SYSOUT=*
//TPDRCNTL DD *
*
*                                START VTS
START VTS,NAME=DK1VTSA
*
*                                SHOW VTS STATUS
SHOW VTS,NAME=DK1VTSA
```

Figure 7-5: Sample STARTVTS JCL

- Execute STARTVTS and confirm successful return code.

On successful completion of this job, started task DK1VTS should be running (see sample output in [Figure 7-6](#) below):

```
DK1P02400I:  INITIALIZING VTS DK1VTSA UNDER compat
DK100259I  TSR size reset to LDS size
DK100417I  TSR successfully mapped to LDS
DK1P02401I:  VTS DK1VTSA IS RUNNING UNDER compat
```

Figure 7-6: Sample DK1VTS output

Note: Do not execute this step unless “[Allocate the VTS-TSR Image dataset for a cataloged VTS](#)” on page 60 and “[Define a sample cataloged VTS](#)” on page 61 completed successfully.

4. Access the sample cataloged VTS

This step accesses the VTS that was started in the previous step.

- Once again, modify the CNTL member TESTVTS (see [Figure 7-7](#)).

Specifically:

- modify the job statement
- modify the dataset name in the STEPLIB statement.

Modify TESTVTS

```
//TESTVTS JOB 0, 'TEST VTS AGENT'
//*****
/*  VERIFY THAT THE INSTALLATION OF THE MVS VTS INTERFACE IS VALID
/*  MODIFY VTSNAME IN SS COMMAND TO VTS BEING TESTED
//*****
//TBDriver EXEC PGM=TBDriver
//STEPLIB DD DISP=SHR,DSN=your.prefix.TBASE.LOAD <=====
//TBOPT DD *
LISTOPTIONS=Y
TSRSize=40K
//SYSOUT DD SYSOUT=*
//TBTSRPT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//CMD DD *
**                DISPLAY CUSTOMER BANNER
BN
**                ACCESS VTS-TSR DK1V TSA
SS,DK1V TSA
**                DEFINE AND POPULATE TABLE
DT, TABLE1
IK, ,CCCCCCCCC
IK, ,RRRRRRRRR
IK, ,AAAAAAAAA
IK, ,ZZZZZZZZZ
IK, ,YYYYYYYYY
**                PRINT TABLE
PR, ,0
**                LIST OPEN TABLES AND TSR USAGE
LT
**                CLOSE TABLE
CL, TABLE1
**                ACCESS LOCAL TSR
SS,
/*
```

Figure 7-7: Sample TESTVTS JCL

- Execute TESTVTS and confirm successful return codes for all commands.

5. Shut down the sample cataloged VTS

This step shuts down the VTS that was started in “Start the sample cataloged VTS” on page 62.

- Modify the CNTL member SHUTVTS (see [Figure 7-8](#)).

Specifically:

- modify the dataset name in the STEPLIB statement.

Modify SHUTVTS

```
//SHUTVTS JOB 0, 'SHUTDOWN VTS'
//*****
//*      SHUTDOWN A VTS (CATALOGUED) UNDER TPVM "COMPAT"
//*****
//SHUTVTS EXEC PGM=TPDRIVER
//STEPLIB DD DISP=SHR,DSN=your.prefix.TBASE.LOAD
//SYSOUT DD SYSOUT=*
//TPDRRPT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DK1MSG DD SYSOUT=*
//TPDRCNTL DD *
*
*                                SHUTDOWN VTS
SHUTDOWN VTS,NAME=DK1VTSA
*
*                                SHOW VTS STATUS
SHOW VTS,NAME=DK1VTSA
```

Figure 7-8: Sample SHUTVTS JCL

- Execute SHUTVTS and confirm successful return code.

Upon successful completion of this step, the started task DK1VTS should no longer be running.

6. Delete the sample cataloged VTS

This step deletes the VTS that was defined in “[Define a sample cataloged VTS](#)” on page 61.

- Modify the CNTL member DELVTS (see [Figure 7-9](#)).

Specifically:

- modify the dataset name in the STEPLIB statement.

Modify DELVTS

```
//DELVTS      JOB 0, 'DELETE A VTS'
//*****
//*          START A VTS (CATALOGUED) UNDER TPVM "COMPAT"
//*****
//DELVTS      EXEC PGM=TPDRIVER
//STEPLIB     DD DISP=SHR,DSN=your.prefix.TBASE.LOAD
//SYSOUT      DD SYSOUT=*
//TPDRRPT     DD SYSOUT=*
//SYSUDUMP    DD SYSOUT=*
//SYSPRINT    DD SYSOUT=*
//DK1MSG      DD SYSOUT=*
//TPDRCNTL    DD *
*
*                               DELETE VTS
DELETE VTS,NAME=DK1VTS
*
*                               SHOW VTS STATUS
SHOW VTS,NAME=DK1VTS
```

Figure 7-9: Sample DELVTS JCL

- Execute DELVTS and confirm successful return code.

Note: Do not execute this step unless “[Shut down the sample cataloged VTS](#)” on page 64 completed successfully.

Proceed to install other components and products, as required.

8

Installing the CICS Interface product

If you have licensed the tableBASE CICS Interface product, the installation media contains additional load modules in the library:

`your.prefix.LOAD`

Note: Before you install the tableBASE CICS Interface software, review the information in [Chapter 1 “Installation overview”](#) on page 17.

Difference between Release 5.x and Version 6 affecting CICS

The difference between Release 5.x and all release levels of Version 6 are:

- No TBTS LIB is required
- tableBASE PC Server or tableBASE Servers for VTS Manager are required
- Threadsafe support is available with V601, Modification Level 7 and later.

Steps to install the CICS Interface product

To install the tableBASE CICS Interface, perform the following steps:

Step 1—Modify default parameters (optional)

Modify the default parameters for operating tableBASE in a CICS environment, if desired. If the default values are acceptable or you wish to use the Override feature provided via the TBOPT dataset, no additional work is necessary in this step.

For parameter descriptions, see [Appendix A on page 95](#). The default values for this component are listed below and are also defined in:

your.prefix.SRC (DK1T2734)

Default CICS parameters

```
TBOPTGEN CICS,
    ALLOW_TB_LIB_IO=Y,
    CICSJRNL=99,
    DATERTNX=N,
    FGDELIM=*,
    HASH_HI_DEN_LIM=900,
    HASH_LOW_DEN_LIM=600,
    LISTOPTIONS=N,
    LOCKTIMER=0,
    LOCKTIMEWTO=30,
    MAXNMTAB=0,
    MTRETAIN=N,
    OVERRIDES=YNYYYYYY,
    STROBE=0,
    SUPPRESS_DUMPS=NONE,
    SWITCHES=YNYYNNNN,
    TABLEWAITRC=0,
    TABLEWAITWTO=30,
    TSR_ALGORITHM=D,
    TSR_WARNING_FREQ=1,
    TSR_WARNING_PCT=85,
    TSR_SIZE=10M,
    USEREXITS=N,
    VTSFIRST=,
    VTSLAST=,
    VTSNAME=,
    VTSPREFIX=VTS:,
    ZEROROWS=Y,
    ML=MAINLIB
```

Note: An additional parameter which is not listed above is TPVM. This parameter is only relevant if you have licensed the VTS Manager product. We recommend that you keep the default for this parameter as delivered and use the Override feature provided through the TBOPT dataset.

Note: The Status Switch setting (SWITCHES) and Overrides (OVERRIDES) are unique to this interface.

To make changes to the installation parameters:

1. Modify **your.prefix.SRC (DK1T2734)**.
2. Reassemble and link-edit the module into the tableBASE CICS interface program DK1CBASE. Use **your.prefix.CNTL (ALT2734)** as a model:

Step 2—Copy tableBASE load modules (optional)

If it is necessary to copy the tableBASE load modules to another system library, use the IBM utility IEBCOPY to do so.

Step 3—Convert from BDAM table libraries to VSAM (optional)

If you are converting any of the tableBASE libraries from the installation media or any of your own tableBASE libraries from a BDAM to a VSAM file organization, see “[7. Convert BDAM table libraries to VSAM \(optional\)](#)” on page 52 for information to perform this task.

Note: Steps 4 and 5 refer to CICS System Definition (CSD) resource definitions for tableBASE libraries, programs and transactions.

Step 4—Create CICS definitions for tableBASE libraries

Create CICS definitions for each tableBASE library accessed under CICS:

- member DFHFCTBD in **your.prefix.SRC** contains macro instructions for defining BDAM files
- member TBASE60V in **your.prefix.SRC** contains CICS System Definition (CSD) resource definitions for defining VSAM files

In addition to MAINLIB, you may include an entry for each additional tableBASE table library to be referenced under CICS.

Define all tableBASE libraries in CICS as NON RECOVERABLE. This is accomplished with LOG=NO on the FCT for any tableBASE table library or RECOVERY(NONE) on the CSD. Additional information is contained under “[Notes and restrictions](#)” on page 74.

Note: Ensure that any tableBASE BDAM libraries are defined using the DFHFCT macros. BDAM datasets cannot be defined as resource definitions in the CSD dataset.

Step 5—Create CICS definitions for tableBASE programs and transactions

Create CICS definitions for each tableBASE program and transaction accessed under CICS.

If tableBASE is called dynamically with a CICS link, and autoinstall (SIT:PGAIPGM=ACTIVE) is not used then TBLBASE (and any other tableBASE CICS APIs such as TBCALLC, TBASEC and TBROOTC used in your application programs) must be defined in the CSD. If the tableBASE calls are through MVS, then MVS will resolve them and the CICS definitions are not required.

The member TBASE60 in **your.prefix.SRC** contains the CSD instructions for defining tableBASE programs and transactions to CICS.

Note: It is important that the modules marked resident remain marked resident; this is part of the tableBASE architecture and any changes will result in product failure. (These are DK1TNAME, DK1TNUCL, DK1TROTC and DK1CBASE.)

Note: Programs DK1TCIN and DK1TSHUT must be defined as quasi-reentrant on the CONCURRENCY attribute of the program resource definition.

Note: tableBASE programs must be defined in the CSD; they cannot be autoinstalled. This is because some tableBASE programs are loaded by the PLTPI program, DK1TCIN, using EXEC CICS SET...PHASEIN and this is not supported by autoinstall.

Step 6—Create CICS definitions for tableBASE journal files

Create the CICS definition for the tableBASE journal file specified in the customizing DK1T2734.

If you are using the CICS Transaction Server (TS) for z/OS, the CICS journal functions are performed in conjunction with the MVS logger. Allocate the necessary log streams on the MVS logger for the journal file specified in DK1T2734. Define the log stream for the journal file in CICS in the CSD. The log stream no longer needs to be defined in the CICS startup job.

The member DFHCSD99 in TBASE.SRC contains sample instructions for defining the MVS log stream for the CICS journal file.

Step 7—Edit PLT

If you have a PLTPI member in your CICS startup, edit the member in the Program List by entering DK1TCIN in the line after the entry for PROGRAM=DFHDELIM. If not, use the sample in member DFHPLT6T in **your.prefix.SRC** as a guide.

Similarly, if you have a PLTSD member in your CICS definition, edit the member in the Program List by entering DK1TSHUT in the line after the entry for PROGRAM=DFHDELIM. If not, use the sample in member DFHPLT6S in **your.prefix.SRC** as a guide.

During system startup this entry causes the tableBASE Resource Manager Interface to initialize. If the initialization is not performed or is not successful (required programs are not defined or not available), then all CICS tasks using TBLBASE or TBASEC will terminate with abend code AEY9.

Step 8—Update binder procedure to use new TBLBASE API

The **your.prefix.LOAD** library contains aliases TBASEC, TBCALLC and MAINROOT for the Version 6 TBLBASE interface program. With Version 6, although the tableBASE interface module is renamed to DK1TCALL, the TBLBASE alias remains the standard interface name.

If you are upgrading from an earlier version of tableBASE and your applications called these earlier versions of the tableBASE interface programs, either statically or dynamically, they will continue to work without having to recompile or relink your applications.

If you wish to convert existing tableBASE/CICS applications to the TBLBASE API or to create new tableBASE/CICS applications using the TBLBASE API, then you must ensure that these applications use the version of the API in the Version 6 **your.prefix.LOAD** and not a version in any other load library. Please update the CICS binder procedure accordingly.

Step 9—Customize CICS startup JCL

Customize your CICS start-up JCL (in the job TBASE.CNTL(CICSJCL)) to include:

- access to the tableBASE load library **your.prefix.LOAD** (or the load library to which the modules have been moved in the previous step)
- the tableBASE library **your.prefix.TBSYSLB** unloaded from the product media
- any other tableBASE libraries used by your online applications
- If the TBOPT dataset is to be used, see the CICS chapter in the *tableBASE Administration Guide*.

Note: All tableBASE libraries may be allocated with DISP=SHR.

To reference an existing tableBASE library, use this JCL as a pattern:

```
//DDNAME DD DSN=your.prefix.MAINLIB,DISP=SHR
```

Each tableBASE library is identified by a unique DDNAME. You may not associate more than one DDNAME with the same tableBASE library. Multiple libraries cannot be concatenated using a single DD statement. The tableBASE ML command is available for logically concatenating libraries.

Step 10—Confirm successful CICS Interface install

Note: Depending on whether or not you are installing VTS Manager, ensure that either the tableBASE PC Server or the tableBASE Servers for VTS Manager have been installed and are running on the same LPAR as this product. If neither of the Servers are running at the time that tableBASE software

initializes, the transaction which issued the tableBASE call will abend and you will see the following error message in the JES LOG:

```
DK100567E The tableBASE PC Server must be running for tableBASE to
initialize.
```

Once the CICS region has been successfully started, verify the installation of the CICS interface.

1. Sign on to CICS.
2. On a clear screen enter the transaction TBDR.
3. Press <ENTER>. The **tableBASE CICS Driver** screen is displayed. You are now able to enter tableBASE and DRIVER commands.
4. On the first line of the **tableBASE CICS Driver** screen, enter the command PR and the table name EXAMPLE.
5. Press <ENTER>. The data is displayed on the screen.

If:	Then:
the installation is successful	pressing <PF3> terminates the CICS DRIVER
the installation was unsuccessful	contact tableBASE Customer Support for help

For more details on the operation of the CICS DRIVER, see the *tableBASE Programming Guide*.

Step 11—Change system abends to task abends

This step is recommended for any CICS regions that access a tableBASE VTS-TSR. If the VTS-TSR region is cancelled or abends there is a chance that CICS tasks would fail. The failure is displayed (or listed) as an MVS S01D or S067abend. These two abends bring down the entire CICS region. They can be converted to Task (rather than System) abends by updating the CICS region System Recovery table (SRT).

1. Review the SRT information in the CICS documentation section of the IBM manual, Resource Definition Guide, to determine if it is possible to revise the SRT at your site.
2. Check if the abend codes are already in the SRT used by your CICS region's start-up parameters (SIT).

If the abend codes are present do not continue.

If the abend code is not present and you wish to add the codes then:

3. Alter the job DFHSRTTB in the library TBASE.CNTL to meet your installation requirement.
4. Submit the job to assemble and link the new SRT containing the entries S01D and S067.
5. Alter the start-up parameters (SIT) (see [Figure 8-1](#)) for each CICS region to use the revised SRT.

```
//DFHAUPLE EXEC DFHAUPLE,NAME=SDFHLOAD
//ASSEM.SYSUT1 DD *
        TITLE 'TABLEBASE CICS SYSTEM RECOVERY TABLE'
        PRINT NOGEN
        DFHSRT TYPE=INITIAL,SUFFIX=TB
        DFHSRT TYPE=SYSTEM,RECOVER=YES,          X
        ABCODE=(01D,067)
        DFHSRT TYPE=FINAL
        END
```

Figure 8-1: Sample job

Notes and restrictions

Level of CICS

Version 6 of the tableBASE software is compatible with the CICS Transaction Server for z/OS.

CICS/TS CSD

A sample input to update the CSD using the CICS resource definition program DFHCSDUP is supplied in **your.prefix.SRC** members TBASE60 and TBASE60V to define tableBASE resources to CICS.

Customize and use the source supplied in **your.prefix.SRC (TBASE60V)** as input to the DFHCSDUP utility to add the CSD VSAM tableBASE library file definitions for the CICS region where tableBASE is to be installed.

If you are using the CICS Transaction Server for z/OS, the sample in member DFHCSD99 of TBASE.SRC should also be customized as input to the DFHCSDUP utility. This will add the MVS log stream for the tableBASE/CICS journal file to the CSD for the CICS region where tableBASE is to be installed.

FCT restriction for tableBASE libraries

All tableBASE libraries in CICS must be defined as NON RECOVERABLE. This is accomplished with **LOG=NO** on the FCT or **RECOVERY (NONE)** on the CSD for any tableBASE library (these settings are defaults in CICS).

The tableBASE software has been designed to ensure the integrity of the libraries, therefore there is no need to use CICS back out and recovery mechanisms.

If CICS is permitted to perform back out and recovery on tableBASE libraries, a transaction interlock occurs. In addition, tableBASE libraries may be damaged during the CICS recovery process.

9

Installing the IMS TM Interface product

If you have licensed the tableBASE IMS TM product, the installation media contains the additional member DK1IBASE in **your.prefix.LOAD**.

Note: Before you install the tableBASE IMS TM Interface software, review the information in [Chapter 1 “Installation overview”](#) on page 17.

Differences between Release 5.x and Version 6 affecting IMS

The differences between Release 5.x and all release levels of Version 6 are:

- no TBTSLIB is required
- there is no need for various TBROOTIx (where x is a digit) modules; TSR size is determined by a site-configurable default or by parameter input through the TBOPT initialization file
- the Version 6 equivalent of DK1TROTb in Release 5.X TBROOTI module is reentrant and resides above the line
- start the tableBASE Program Call (PC) Server or the tableBASE Servers for VTS Manager, if you are licensed for VTS Manager, before running the IMS interface.

Steps to install the IMS/TM Interface product

To install the tableBASE IMS/TM interface software, perform the steps that follow.

Step 1—Modify default parameters (optional)

If desired, modify the default parameters for operating the tableBASE software in an IMS message processing region (MPR). If the default values are acceptable or you wish to use the Override feature provided via the TBOPT dataset, no additional work is necessary in this step.

For parameter descriptions, see [Appendix A on page 95](#). The default values for this component are listed below and are also defined in **your.prefix.SRC (DK1T1334)**.

Default IMS parameters

```
TBOPTGEN IMS,
    ALLOW_TB_LIB_IO=Y,
    DATERTNX=N,
    FGDELIM=*,
    HASH_HI_DEN_LIM=900,
    HASH_LOW_DEN_LIM=600,
    LISTOPTIONS=N,
    LOCKTIMER=0,
    LOCKTIMEWTO=30,
    MAXNMTAB=0,
    MTRETAIN=N,
    OVERRIDES=YYYYYYYY,
    STROBE=0,
    SWITCHES=NNYYNNNN,
    TABLEWAITRC=0,
    TABLEWAITWTO=30,
    TSR_ALGORITHM=D,
    TSR_WARNING_FREQ=1,
    TSR_WARNING_PCT=85,
    TSSIZE=10M,
    USEREXITS=N,
    VTSFIRST=,
    VTSLAST=,
    VTSNAME=,
    VTSPREFIX=VTS:,
    ZEROROWS=Y,
    ML=MAINLIB
```

Note: An additional parameter which is not listed above is TPVM. This parameter is only relevant if you have licensed the VTS Manager product. We recommend that you keep the default for this parameter as delivered and use the Override feature provided through the TBOPT dataset.

Note: The Status Switch settings (SWITCHES) and Overrides (OVERRIDES) are unique to this environment.

To make changes to the installation parameters:

1. Modify **your.prefix.SRC (DK1T1334)**.
2. Reassemble and link-edit the module into the tableBASE IMS interface program DK1IBASE using **your.prefix.CNTL (ALT1334)** as a model.

Step 2—Copy tableBASE load modules (optional)

If it is necessary to copy the tableBASE load modules to another system library, use the IBM utility IEBCOPY to do so. If reblocking is required use the COPYMOD control statement.

Step 3—Check pre-load list

At this point, you should ensure that your IMS pre-load list is updated for Version 6. See the IMS Chapter in the *tableBASE Administration Guide*.

Step 4—Customize IMS startup JCL

To use tableBASE Version 6 in an IMS environment, review the information in the *tableBASE Administration Guide* for the JCL for IMS/TM transactions (rules for DLI batch apply to BMPs and IMS Batch), pre-loading of tableBASE modules, restrictions, etc. Make the changes to your IMS environment that are listed in the *tableBASE Administration Guide*.

Step 5—Confirm successful IMS Interface install

Note: Depending on whether or not you are installing VTS Manager, ensure that either the tableBASE PC Server or the tableBASE Servers for VTS Manager have been installed and are running on the same LPAR as this product. If neither of the Servers are running at the time that tableBASE software initializes, the transaction which issued the tableBASE call will abend and you will see the following error message in the JES LOG:

```
DK100266S tableBASE PC Server unavailable.
```

To verify a successful install of the IMS MPR support:

1. Prepare or modify a simple IMS transaction to include a call to TBLBASE, for example, use an LS command to return the current status switch settings.
2. Prepare a TBOPT file with at least this parameter — LISTOPTIONS=Y.

3. Try your transaction.
4. Examine the MPR's JES Log using the System Display and Search Facility (SDSF) for various start-up messages from tableBASE system that show the active runtime options. This is an indication that tableBASE has initialized successfully and is ready for subsequent calls from your transactions.

10

Installing the DB2 Stored Procedures Interface product

Use the procedures described in this chapter to modify and transfer the DB2 Stored Procedures Interface.

Note: Before you install the DB2 Stored Procedures Interface software, review the information in [Chapter 1 “Installation overview”](#) on page 17.

Steps for DB2 Stored Procedure support

To install the tableBASE DB2 Stored Procedures Interface software, perform the steps that follow.

Step 1—Modify default parameters (optional)

Modify the default parameters for operating tableBASE in a DB2 Stored Procedures environment, if desired. If the default values are acceptable or you wish to use the Override feature provided via the TBOPT dataset, no additional work is necessary in this step.

For parameter descriptions, see [Appendix A on page 95](#). The default values for this component are listed below and are also defined in **your.prefix.SRC (DK1T1434)**.

Default DB2 parameters

```
TBOPTGEN DB2,  
    ALLOW_TB_LIB_IO=N,  
    DATERTNX=N,  
    FGDELIM=*,  
    HASH_HI_DEN_LIM=900,  
    HASH_LOW_DEN_LIM=600,  
    LISTOPTIONS=N,
```

```
LOCKTIMERC=0,  
LOCKTIMEWTO=30,  
MAXNMTAB=0,  
MTRETAIN=N,  
OVERRIDES=YYYYYYYY,  
SWITCHES=NNYYNNNN,  
TABLEWAITRC=0,  
TABLEWAITWTO=30,  
TSR_ALGORITHM=D,  
TSR_WARNING_FREQ=1,  
TSR_WARNING_PCT=85,  
TSRSIZE=10M,  
USEREXITS=N,  
VTSFIRST=,  
VTSLAST=,  
VTSNAME=,  
VTSPREFIX=VTS:,  
ZEROROWS=Y,  
ML=MAINLIB
```

Note: An additional parameter which is not listed above is TPVM. This parameter is only relevant if you have licensed the VTS Manager product. We recommend that you keep the default for this parameter as delivered and use the Override feature provided through the TBOPT dataset.

Note: The Status Switch setting (SWITCHES) and Overrides (OVERRIDES) are unique to this environment.

To make changes to the default parameters:

1. modify **your.prefix.SRC(DK1T1434)**.
2. Reassemble and link-edit the module into the tableBASE module DK1DBASE using **your.prefix.CNTL(ALT1434)** as a model.

Step 2—Copy tableBASE load modules (optional)

If it is necessary to copy the tableBASE load modules to another system library, use the IBM utility IEBCOPY to do so. If reblocking is required use the COPYMOD control statement.

Step 3—Customize startup JCL

To use tableBASE Version 6 in a DB2 Stored Procedures environment, review the information in the "DB2 Stored Procedures" chapter of the *tableBASE Administration Guide*. Make the recommended changes to the JCL for the SPAS environment, as described in that chapter.

11

Installing the tablesONLINE/ CICS product

This chapter describes the tablesONLINE/CICS installation procedures.

Note: Before you install the tablesONLINE/CICS software product, review the information in [Chapter 1 “Installation overview”](#) on page 17.

Steps to install tablesONLINE/CICS

If you have licensed the tablesONLINE/CICS product, the installation media contains additional load and other libraries.

Note: Before installing tablesONLINE/CICS, complete the installation of the CICS Interface, as described in [Chapter 8 “Installing the CICS Interface product”](#) on page 67. The resource definitions for tablesONLINE/CICS will have been installed as part of the process of installing the CICS Interface.

The steps involved in installing tablesONLINE/CICS vary, depending on whether you are a first-time user or you are upgrading from an earlier release. If you are upgrading from an earlier release, start with the steps [“Upgrading tablesONLINE/CICS from Release 4.2.x, 5.0.x, or 5.1”](#) on page 86 and when those are completed continue with [“Step 2—Customize CICS startup JCL”](#) on page 82.

Installing tablesONLINE/CICS for the first time

To proceed with the implementation of tablesONLINE/CICS in an MVS installation, perform the following steps:

Step 1—Copy the tablesONLINE control libraries

Copy the contents of the following tableBASE libraries to three new libraries—for an example of the JCL required to do so, see the member TBOL600 in **your.prefix.CNTL** (or TBOL600V if it is a VSAM library):

- **your.prefix.TBACTLB**
- **your.prefix.TBAPPLB**
- **your.prefix.TBDICLB**

This is necessary because some or all of the tables in the above libraries will be altered in the course of using tablesONLINE/CICS. Copying the libraries ensures that the original libraries will always be available for reference.

Step 2—Customize CICS startup JCL

Customize your CICS start up JCL to include the tablesONLINE/CICS table libraries created in Step 1:

- **your.prefix.CUSTOM.TBACTLB**
- **your.prefix.CUSTOM.TBAPPLB**
- **your.prefix.CUSTOM.TBDICLB**

and any other tableBASE libraries used by your online applications. See the member CICSTBOL in **your.prefix.CNTL**.

Note: All tableBASE libraries may be allocated as DISP=SHR.

Create CICS definitions for each tablesONLINE library that will be accessed under CICS:

- If you are using BDAM libraries, member DFHFCTOL in **your.prefix.SRC** contains macro instructions for defining BDAM files.
- If you are using VSAM libraries, member TBOL60V in **your.prefix.SRC** contains CICS System Definition (CSD) resource definitions for defining tablesONLINE/CICS specific VSAM files.

Step 3—Enable batch job submission from CICS—Part 1

To utilize the COBOL copybook generation, or the PRINT TABLE and VIEW features, you must ensure that the parameter SPOOL=YES is defined in the SIT.

Step 4—Create CICS definitions

Part 1: Perform only if you want mixed case character input

To enable tablesONLINE transactions to capture mixed case characters when editing tableBASE tables, the tablesONLINE transactions require the setting of UCTRAN to TRANID on the associated CICS terminals.

Terminal LU names and definition methods vary by installation. To assist you, the group DKLUCTRN contains templates for terminal definitions. A sample input to update the CSD using the CICS resource definition program DFHCSDUP is supplied in **your.prefix.SRC** member DKLUCTRN. You must modify these templates to suit the installation requirements.

Installations that use the auto-install facility of CICS to define terminals must make sure that the auto-install program will use the appropriate model definition. For more information on the CICS auto-install facility, please see IBM's *CICS Resource Definition Guide* and *CICS Customization Guide*.

Part 2: Create CICS definitions for the tablesONLINE transactions and libraries

Create CICS definitions for the tablesONLINE transactions and libraries that will be accessed under CICS.

Member TBOL60 in **your.prefix.SRC** provides the input to define tablesONLINE to CICS using the CICS resource definition program, DFHCSDUP.

If you are using VSAM libraries, member TBOL60V in **your.prefix.SRC** provides sample definitions for the tablesONLINE and CICS VSAM libraries.

If you are using BDAM libraries, member DFHFCTOL in **your.prefix.SRC** provides sample definitions for the tablesONLINE and CICS BDAM libraries.

Note: See “[TWA size for tablesONLINE/CICS transactions](#)” on page 86 for the TWA size to use for the CICS definitions in member TBOL60.

Note: If you will be using tablesONLINE in an MRO environment, please see the “[Notes and restrictions](#)” on page 85.

Step 5—Confirm successful tablesONLINE install

Use this procedure to verify the tablesONLINE installation.

1. Ensure that either the tableBASE PC Server or the tableBASE Servers for VTS Manager are running.
2. Sign on to CICS.
3. On a clear screen, enter the transaction TBOL. Press <ENTER> and a menu appears. If a menu appears the installation is successful. Press <PF3> to terminate the TBOL session. If no menu appears, please contact DataKinetics Technical Support: 1-613-523-5588.

For a detailed description of the use and operation of tablesONLINE/CICS, see the *tablesONLINE/CICS User's Guide*.

Step 6—Enable batch job submission in CICS—part 2

The generation of C and COBOL copybooks and the printing of tables and Views are driven from four tables, TBOLJCL1, TBOLJCL2, TBOLJCL3 and TBOLJCL4 in the **your.prefix.TBAPPLB** library.

In order to use these features, you must modify the STEPLIB DSN, JOBCLASS, and MSGCLASS, in tables TBOLJCL1, TBOLJCL2, TBOLJCL3, and TBOLJCL4 so that they conform to your installation standards. Detailed instructions for modifying tables will be found in the *tablesONLINE/CICS User's Guide*. For those unfamiliar with tablesONLINE, the screen sequence for locating the table editor is:

From the Administration Screen select:
D - DEVELOP APPLICATION; then select:
A - EDIT TABLE

Step 7—Change Fetch Generic Delimiter (optional)

This step is optional. If you wish to change the Fetch Generic Delimiter from the default asterisk (*) to some other character, you must:

- Log on onto tablesONLINE/CICS as an administrator and edit the TBOLCNST table, using option 9 (edit constants), on the tablesONLINE/CICS **Administrator** menu.
- Select the DEFAULT row in TBOLCNST and scroll down through the screens till you get to the screen with the FG DELIMITER field. Change the value and exit from edit using <PF3>.
- Exit completely from tablesONLINE/CICS by pressing <PF3> until you see the Confirmation Screen (message TB-5052W).

- Change the source module DK1T2734 (For more information, please see “[Step 1—Modify default parameters \(optional\)](#)” on page 67).

Step 8—Confirm successful tablesONLINE/CICS install

Once the above steps have been completed, the system is ready for validation by the tablesONLINE/CICS administrator. For information on tablesONLINE/CICS administration, please refer to the *tableBASE Administration Guide*.

Notes and restrictions

This section details the Notes and Restrictions that apply to Version 6.

CICS/MRO or CICS/ISC

The system table, TBOLMRO, is supplied in the tablesONLINE system library (TBSYSLB). This table is to be modified only if tablesONLINE is to be installed in more than one Application Owning Region (AOR) in the MRO/ISC complex. It contains information required to set up tablesONLINE to operate in CICS/MRO. This table is updated as part of the tablesONLINE installation process in a CICS/MRO environment.

TBOLMRO is a reserved table name which may not be used as a table name by a user in tableBASE/CICS. When installing tablesONLINE for CICS/MRO, customize and run the JCL supplied in TBASE.CNTL (TBOLMRO). Instructions for customizing this JCL are supplied as comments in the data set member (TBOLMRO). This job will update the TBOLMRO table with the required table items.

The existence of an item in the TBOLMRO table with the CICS APPLID of the AOR will determine if tablesONLINE has been installed to operate in a CICS/MRO complex. If the TBOLMRO table does not contain an item with the APPLID of the AOR, then tablesONLINE will operate as if it were installed in a stand-alone CICS region.

The first time tablesONLINE/CICS is invoked in an AOR, it will determine the CICS transaction IDs, in the operating CICS region, for all of its pseudo-conversational transactions by referring to the TBOLMRO table. This process is done only once, therefore, an error in setting up the TBOLMRO table will require the CICS AOR to be recycled.

A View and an empty Data Table are distributed so that tablesONLINE can also be used to edit the TBOLMRO table.

Converting Views

If you are upgrading from a release prior to Release 5.0, please note the following.

In Release 5.0, the internal structure of a View was changed. tablesONLINE will automatically convert Views from Releases prior to 5.0 to this new structure. However, at some sites, the requisite libraries may not be updateable when tablesONLINE is run. Therefore, we have developed a procedure to convert your pre-Release 5.0 Views to the format needed for the current version of tablesONLINE. This procedure is defined in Chapter 13 of the *tableBASE Administration Guide*.

TWA size for tablesONLINE/CICS transactions

The Transaction Work Area (TWA) size for tablesONLINE/CICS transactions is 5595 bytes in Version 6. If you are creating your CICS resource definitions using the TBOL60 member provided in **your.prefix.SRC**, these already contain the right TWA size for these transactions. If you are upgrading from a release prior to 5.1.0 of tablesONLINE/CICS and intend to use existing CICS resource definitions, you must ensure that the TWA size for these transactions is updated in your CICS definitions.

Upgrading tablesONLINE/CICS from Release 4.2.x, 5.0.x, or 5.1

Note: Before you install tablesONLINE/CICS you should review the information in [Chapter 1 “Installation overview”](#) on page 17.

The following steps are specific to upgrading from a prior release of tablesONLINE/CICS. It is important that all tableBASE libraries are either in the BRIDGE format or are converted to the Version 6 format before attempting the installation outlined below. tablesONLINE/CICS libraries cannot be in Bridge format. tablesONLINE/CICS Version 5 libraries are compatible with tableBASE Bridge loads. Note that tablesONLINE/CICS Version 4.2.x must be upgraded to Version 5; Version 5 can then be upgraded to Version 6. For more information, see [“4. Convert tableBASE libraries to Version 6 format”](#) on page 50.

Note: Before performing the following steps, ensure that the PC Server is running.

Upgrade 1—Create new Version 6 TBACTLB library

Create new Version 6 TBACTLB library.

Run job TBOL601 in **your.prefix.CNTL** (or TBOL601V if you are using VSAM). Modify the JCL in this member to conform to your installation standards.

This job will create a new library by copying the tables from your Release 4.2 or 5.x library and upgrading the TBOLACT table to the Version 6 level, if required.

Upgrade 2—Create new Version 6 TBDICLB library

Create new Version 6 TBDICLB library.

Run job TBOL602 in **your.prefix.CNTL** (or TBOL602V if you are using VSAM). Modify the JCL in this member to conform to your installation standards.

This job creates a new library by copying the tables from your Release 5.x library. If you have not used TBDICLB before, the instructions in job TBOL602 or TBOL 602V will direct you to copy the contents of the supplied **your.prefix.TBDICLB** to your new library.

Upgrade 3—Create new Version 6 TBAPPLB library

Create new Version 6 TBAPPLB library.

If you are upgrading from Release 4.0, run the job TBOL603 in **your.prefix.CNTL**. If you are upgrading from Release 5.x, run job TBOL604 or TBOL604V if you are using VSAM libraries in **your.prefix.CNTL**. Modify the JCL in either of these members to conform to your installation standards.

The job creates a new library by copying the tables from your Release 5.x library. The TBOLPROF table will be upgraded to Version 6 level.

In past installation procedures, temporary tables used for upgrading the TBAPPLB libraries and tables used in tableBASE educational programs were automatically downloaded to the customized TBAPPLB libraries of the customer. In order to simplify library usage and reduce storage requirements, included with each job are the DELETE control statements to remove these unnecessary tables.

Finally, this step produces a directory listing of all the tables remaining in the TBAPPLB library. This directory listing is used to prepare the next three steps: [Upgrade 4—Update description tables](#), [Upgrade 5—Update menu tables](#) and [Upgrade 6—Update message tables](#).

Note: If your CICS regions are currently running with tableBASE Bridge, before proceeding with the following upgrade steps, please contact tableBASE Support for clarification on the release level that you will be upgrading from.

Upgrade 4—Update description tables

The job defined in the UPDTDESC member in **your.prefix.CNTL** is required to update your xxxxDESC description tables. This job updates and also compares your xxxxDESC tables with the TBOLDESC table that was delivered in an earlier release. It ensures that any customizations that may have been made are reflected in the Version 6 xxxxDESC tables.

This job must be run once for each xxxxDESC table including TBOLDESC, where xxxx is usually the application name, but can be any four characters.

As part of the job, the program TBCOMP compares your versions of the xxxxDESC tables with the originally released TBOLDESC. It may return a condition code-- see [Table 11-1](#).

Table 11-1: Upgrade condition codes

Condition code	Meaning
00	No changes to the xxxxDESC table -- it is an unchanged copy of the originally released TBOLDESC.
08	This is normal and indicates that there are differences between the tables. These differences reflect the customizations at your site; they are then applied, via TBEXEC, to the newly created versions of the xxxxDESC tables.
16	Job stops with this condition code. Indicates that there were non-matching table definitions-- these are listed in the TBCOMP output.
n	Any other condition code is an error-- refer to the list of tableBASE return codes in the Appendix "tableBASE Messages" in the <i>tableBASE Programming Guide</i> .

Upgrade 5—Update menu tables

The job defined in the UPDTMENU member in **your.prefix.CNTL** is required to update your xxxxMENU menu tables. This job updates and also compares your xxxxMENU tables with the TBOLMENU table that was delivered in an earlier release. It ensures that any customizations that may have been made are reflected in the Version 6 xxxxMENU tables.

This job must be run once for each xxxxMENU table including TBOLMENU (where xxxx is usually the application name, but can be any four characters).

As part of the job, the program TBCOMP compares your versions of the xxxxMENU tables with the originally released TBOLMENU. It may return a condition code-- see [Table 11-1](#).

Upgrade 6—Update message tables

The job defined in the UPDTMSG member in **your.prefix.CNTL** is required to update your xxxxMSG messages tables. This job updates and also compares your

xxxxMSGs table with the TBOLMSGs table that was delivered in an earlier release. It ensures that any customizations that may have been made are reflected in the Version 6 xxxxMSGs tables.

This job must be run once for each xxxxMSGs table including TBOLMSGs, where xxxx is usually the application name, but can be any four characters.

As part of the job, the program TBCOMP compares your versions of the xxxxMSGs tables with the originally released TBOLMSGs. It may return a condition code-- see [Table 11-1](#).

Upgrade 7—Update help table

The job defined in the UPDTHELP member in **your.prefix.CNTL** is required to update your TBOLHELP help table. This job updates and also compares your TBOLHELP table with the TBOLHELP table that was delivered in an earlier release. It ensures that any customizations that may have been made are reflected in the Version 6 version.

Unlike the previous three jobs, this job is run only once.

As part of the job, the program TBCOMP compares your version of the TBOLHELP table with the the Version 6 TBOLHELP. It may return a condition code-- see [Table 11-1](#).

Upgrade 8—Update security module (optional)

If the module TBDKUSID of prior releases has never been customized skip this step and go to “[Step 2—Customize CICS startup JCL](#)” on page 82.

If, in a previous release of tablesONLINE/CICS, you have modified the module TBDKUSID, this modification needs to be reflected in the load module supplied with Version 6.

If you are upgrading from tablesONLINE Release 5.1, copy the existing Release 5.1 TBDKUSID into the Version 6 load library replacing the delivered version. If you are upgrading from a release earlier than 5.1; the recommended choice is to then reapply your changes to the new source version of TBDKUSID supplied with Version 6. Recompile and relink it for use with Version 6, replacing the TBDKUSID load module supplied with Version 6.

TBDKUSID will be executed according to the SECURITY IO IND field in the Constants table. Set SECURITY IO IND field as shown in [Table 11-2](#).

Table 11-2: Security I/O indicator settings

I	TBDKUSID to be executed at system startup.
O	TBDKUSID to be executed at system exit.
Y	TBDKUSID to be executed at both system startup and exit.

If you wish to continue using your old version of TBDKUSID, change the SECURITYPGM VERSION field in TBOLCNST from 6 to 4 for Release 4.2 or 5 for Release 5.0.1. Save the changes.

To edit TBOLCNST, select option 9 (edit constants), on the tablesONLINE **Administrator** menu. Then replace the TBDKUSID load module supplied with Version 6 with your Release 4.0 or 5.0 version of TBDKUSID (that is, first change TBOLCNST, then replace TBDKUSID).

Note: These steps must be performed in the order specified or you will not be able to go back into tablesONLINE/CICS.

Go to [“Step 2—Customize CICS startup JCL”](#) on page 82.

12

Installing the tablesONLINE/ ISPF product

This chapter describes the tablesONLINE/ISPF installation procedures.

Note: Before you install the tablesONLINE/ISPF software product, review the information in [Chapter 1 “Installation overview”](#) on page 17.

If you have licensed the tablesONLINE/ISPF product, the installation media contains additional load modules and ISPF datasets.

Steps to install tablesONLINE/ISPF

To proceed with the implementation of the tablesONLINE/ISPF product, perform the following steps:

Step 1—Change CLIST to variable blocked file (optional)

The CLIST library **your.prefix.CLIST** is shipped as a fixed blocked (FB) file. If you require a variable blocked file, do the following:

1. Rename **your.prefix.CLIST** to **your.prefix.CLISTFB**
2. Allocate a variable blocked file with LRECL of 255 with the Dataset Name **your.prefix.CLIST**
3. Copy **your.prefix.CLISTFB** to **your.prefix.CLIST**

TBASE.CLIST members use SYSDA as a generic name for disk storage. If this is not applicable at your installation, modify the members to suit your requirements. Members affected are listed in [Table 12-1](#).

Table 12-1: TBASE CLIST members

TBALLOC	TBDEFD	TBONLINE
TBALLOCS	TBDEFNIT	TBPRINT
TBBUILD	TBDEFPRT	TBPSWD
TBCHKDSN	TBDEFULD	TBRENAME
TBCLEAR	TBDELETE	TBRESET
TBCOPY	TBDIRECT	TBSUBDEF
TBCOPYFD	TBDRIVER	TBSUBHRD
TBDEFDEL	TBEXPAND	TBSYSDA
TBDEFINE	TBLOAD	TBUNLOAD

Step 2—Point TBALLOCS CLIST to newly installed TBSYSLB

1. Customize the member TBALLOCS in `your.prefix.CLIST(TBALLOCS)`.
2. Change the reference in the PROC statement from `your.prefix.TBSYSLB` to the actual Dataset Name.

Step 3—Point TBSKDEF SKELETON to newly installed load library

1. Customize the member TBSKDEF in `your.prefix.SKELETON(TBSKDEF)`.
2. Change the reference in the member from `your.prefix.LOAD` to the actual Dataset Name.

Step 4—Modify the primary ISPF menu to point to tablesONLINE

Access to tablesONLINE/ISPF is through a menu selection.

Modify the primary ISPF menu or a subordinate menu in order to allow access to tablesONLINE.

A sample of a primary ISPF menu is provided in:

```
your.prefix.PANELS(ISRTPRIM).
```

An entry W has been added in this panel to allow users to access tablesONLINE/ISPF. This selection option matches the user documentation.

After reviewing this sample, modify the primary ISPF or subordinate menu to include the tableBASE option.

Step 5—Modify Help tutorial to point to tablesONLINE help

Modify the HELP tutorial for the panel that is used to invoke tablesONLINE. A sample panel is provided in:

```
your.prefix.PANELS(ISRTHELP).
```

Step 6—Connect tablesONLINE to ISPF

In order for tablesONLINE/ISPF to be operational, allocate the datasets listed in [Table 12-2](#) to the appropriate files.

Table 12-2: ISPF datasets

Dataset name	Allocated to
<code>your.prefix.TBASE.LOAD</code>	ISPLLIB or STEPLIB
<code>your.prefix.TBASE.PANELS</code>	ISPPLIB
<code>your.prefix.TBASE.SKELETON</code>	ISPSLIB
<code>your.prefix.TBASE.MESSAGES</code>	ISPMLIB
<code>your.prefix.TBASE.TABLE</code>	ISPTLIB
<code>your.prefix.TBASE.CLIST</code>	SYSPROC

The tablesONLINE/ISPF COBOL programs are compiled with COBOL for z/OS. The LE runtime modules must be available to the ISPF environment through an allocation, through the LINKLIST, or through the LPA.

These allocations may be done in different ways depending on your ISPF environment:

- The CLIST `your.prefix.CLIST(TBONLINE)` may be customized. Execution of this CLIST as part of the user's TSO logon PROC will set the environment for the tablesONLINE option to be selected.
- The datasets may be allocated as part of the general TSO logon CLIST.
- The datasets may be allocated as part of the other dataset allocations in a TSO logon PROC.
- Remove any allocations based on previous versions of tableBASE.

Select the method best suited to your installation and carry out the steps needed to install, using the selected method. tableBASE does MVS LOADs under TBOL/ISPF. This

means that the STEPLIB in use at tableBASE initialization determines which tableBASE modules are loaded. Unless a dynamic STEPLIB product is available, multiple versions of tableBASE cannot be supported during an ISPF session.

Step 7—Confirm successful tablesONLINE/ISPF install

Ensure that the PC Server is running.

Use the example procedures in Appendix A of the *tablesONLINE/ISPF User's Guide* to confirm that everything is operational. This test will verify that tablesONLINE/ISPF is properly installed.

Notes and restrictions

The tablesONLINE/ISPF user must be aware of the following operating considerations and limitations:

1. If a screen is split into two sessions, tablesONLINE/ISPF may be run in only one of the sessions. The other session may be used for other ISPF functions.
2. When the full screen Browse Table facility is used, the user will need to know the ISPF BROWSE functions.
3. The tablesONLINE/ISPF limitations are:
 - a. maximum field size is 50 characters
 - b. maximum number of key fields is 10
 - c. numeric fields may not be longer than 16 digits
 - d. first character of table name must be upper case alphabetic or numeric
 - e. space on a user library will be required for Views.

Appendix A

tableBASE run-time options

This appendix lists the tableBASE run-time option parameters. Most occur in all interfaces. If you are running applications across multiple environments any changes to these options in one interface may need to be made in each installed interface. Options can be customized for your site at installation time, using the TBOPTGEN macro in DK1Txx34 modules. Applications can further override options on an individual basis, using the TBOPT dataset.

ALLOW_TB_LIB_IO—Allow tableBASE library I/O

This parameter allows access to tableBASE libraries. When set to Y, access to tableBASE libraries is allowed; when set to N, access to tableBASE libraries is not allowed. By default, the parameter is set to Y, with the exception of the DB2 Stored Procedures interface. While it is possible to set the parameter to Y for the DB2 Stored Procedures interface, it is not recommended as this could cause tableBASE to abend if more than 1 task in a DB2 WLM region access a tableBASE library at the same time.

CICSJRNL—CICS Journal File ID

This parameter pertains only to the CICS interface. It identifies the CICS Journal File onto which the TABLE SPACE Report records are to be written by the system, if strobe records are to be written. The delivered default value for the CICSJRNL parameter is 99.

This option applies to this region's users, regardless of which TSR they access.

DATERTNX—Date-Sensitive Processing Found Code

The DATERTNX parameter controls the value placed in the found code for a fetch by count (FC) operation on a date-sensitive table.

With an FC command, it is possible that the row retrieved is not within the date range. When this occurs, the found code is normally set to N to reflect the logical condition that the retrieved row is not a match. For compatibility with previous releases, the found code

returned under this condition can be made to be X. The default setting of DATERTNX=N provides this compatibility with previous releases.

If Date-Sensitive Processing is being introduced in a new application, or no FC commands are used in the application, it is strongly recommended that DATERTNX=Y be specified so the found code will never be set to X. The delivered default is DATERTNX=N for backwards compatibility.

This option applies to this region's users, regardless of which TSR they access.

FGDELIM—Fetch Generic Delimiter

This parameter identifies the character to denote the end of the high order part of the key used in the FG (Fetch Generic) command.

The delivered default value for FGDELIM is an asterisk (*), for example, FG,table-name,key*. We recommend that programmers use the key length override function in the tableBASE command area as this executes faster, is less error prone and allows an asterisk to be part of the key.

This option applies to this region's users, regardless of which TSR they access.

Note: If you are installing tablesONLINE/CICS and wish to change the Fetch Generic Delimiter, please see the tablesONLINE/CICS section of the *tableBASE Installation Guide*.

HASH_HI_DEN_LIM—High Density Limit for Hash Indexes

HASH_LOW_DEN_LIM and HASH_HI_DEN_LIM limit the density of the index for a hash table opened in Version 6. These values are designed to prevent performance problems which can occur when inappropriately high values are used when defining hash tables (can result in numerous new indexes being created). Other problems occur if the difference between low and high density values is too small. A ratio of 2/3 is now enforced: Low density may not be greater than 2/3 of high density.

HASH_HI_DEN_LIM=nnn must be between 100 and 900 (10% and 90%); the default is 900. We recommend the value be lowered to 500.

This option applies to this region's users, regardless of which TSR they access. (An exception to this is a situation where a Version 6, Release 6.0.3, or higher, application accesses a Release 6.0.2 or lower VTS-TSR—in this case the region's density limits are ignored.)

HASH_LOW_DEN_LIM—Low Density Limit for Hash Indexes

See above (“HASH_HI_DEN_LIM—High Density Limit for Hash Indexes”).

HASH_LOW_DEN_LIM=nnn must be between 10 and 600; the default is 600. We recommend the value be lowered to between 200 and 300.

This option applies to this region's users, regardless of which TSR they access. (An exception to this is a situation where a Version 6, Release 6.0.3, or higher, application accesses a Release 6.0.2 or lower VTS-TSR—in this case the region's density limits are ignored.)

LDS—Linear Data Set Use

Specifies whether the TSR defined in this jobstep will be mapped to a LDS (Linear Data Set). The LDS must be pre-defined.

In a batch or VTS environment, this option can be set to Y to allow a mapping of the local TSR or VTS-TSR to a LDS. If set to N, there is no mapping to a LDS.

The delivered default is *N*.

This option applies to the TSR created by this region/job and all users accessing it.

Note: This parameter applies only if you have licensed the optional VTS Manager product.

LIBnn, ML—tableBASE Library List

Specify in sequence, the names of the libraries (DDNAMEs) in the tableBASE Library List (LIB-LIST). In TBOPTGEN, this is specified as a comma delimited list under ML. In the TBOPT file, the library names are specified as LIBnn=lib_name. Up to 10 LIBnn may be specified, as long as the numbers are contiguous. For example, LIB01=MAINLIB, LIB02=TESTLIB.

A VTS-TSR may also be included in the search list by specifying the name of the VTS-TSR prefixed by "VTS:" in LIB-LIST (for example, VTS:DKL1), instead of a library DDNAME.

The delivered default is ML=MAINLIB which is equivalent to supplying a single parameter LIB01=MAINLIB. If no ML is set, tableBASE uses MAINLIB.

This option applies to this region's users, regardless of which TSR they access.

LISTOPTIONS—List Parameter Options

Determines whether or not to list execution time parameters. If LISTOPTIONS=Y is specified, all default parameters and parameters overridden by the TBOPT dataset are listed in the JESMSG LG. LISTOPTIONS=N suppresses this list. The delivered default is LISTOPTIONS=N.

LISTOPTIONS=X is a special setting for TBOPTGEN (DK1T1134). It is the equivalent of LISTOPTIONS=N if the TBOPT DD was not present in the jobstream, and the equivalent of LISTOPTIONS=Y if the TBOPT DD is present. LISTOPTIONS=X applies to TBOPTGEN (DK1T1134) only; it does not apply to TBOPT.

This option applies to this region's users, regardless of which TSR they access.

LOCKTIMERC—Lock Timer Wait Value

LOCKTIMERC=nnnnnn specifies the number of seconds (default 0) that tableBASE should wait for a lock. When the LOCKTIMERC interval has passed, RC=71 is returned. A value of LOCKTIMERC=0 specifies that the process will never time out.

The lock controlled by LOCKTIMERC is used internally by tableBASE to maintain table integrity in the TSR. It is unrelated to the table ENQUEUE that occurs when a table is opened for write (OW).

This option applies to this region's users, regardless of which TSR they access.

LOCKTIMEWTO—Lock Timer Message Wait Value

LOCKTIMEWTO=nnnnnn specifies the number of seconds (default 30) to wait before issuing messages that the process is waiting for a lock. A value of LOCKTIMEWTO=0 specifies that no warning message will be issued.

The lock controlled by LOCKTIMEWTO is used internally by tableBASE to maintain table integrity in the TSR. It is unrelated to the table ENQUEUE that occurs when a table is opened for write (OW).

When applicable, messages are generated to report that TSR space allocation has been blocked by other processes.

This option applies to this region's users, regardless of which TSR they access.

MAXNMTAB—Maximum Number of Tables

Determines the number of tables that can be opened in a given local TSR or VTS-TSR. (With the possibility of creating a TSR of up to 2G in size, there can be a great variation

in how many tables will actually be in a TSR at any time.) If the value of the MAXNMTAB is not set by the user (or is set to zero) then a default value is calculated.

If an attempt is made to open more tables than is indicated by MAXNMTAB, an error code of 20 will be given: The maximum number of tables has been exceeded. Check MAXNMTAB.

This option applies to the TSR created by this region/job and all users accessing it.

Note: For a VTS-TSR region, VTSNMTAB is retained for backwards compatibility and is treated as an equivalent.

User sets the value of MAXNMTAB

To optimize the memory required for system overhead, the user may choose to set the value of MAXNMTAB rather than use the default value. MAXNMTAB can be set to any integer, zero or greater. However, if MAXNMTAB is set to a value greater than the default for the given TSR size, the default is used and the user is notified with a warning message.

After the user sets the value of MAXNMTAB, memory is allocated and an index is created that contains enough entries to support the maximum number of tables indicated. Additional memory is allocated as tables are opened. This memory is reused as tables are closed and new tables opened. This approach keeps the system overhead in line with the number of tables that are opened.

The calculation of the default value of MAXNMTAB

If the user does not set the value MAXNMTAB or the value is zero, the tableBASE software calculates the default value based on the size of the TSR. Each table in the TSR occupies a minimum of 4K pages. The maximum possible number of tables is 51,455, for a 2G TSR; the default, when unspecified is one table for every K page in the TSR.

MTRETAIN—Retain Rows and Index Areas

MTRETAIN=Y|N determines whether the allocated rows and index areas are to be retained when an MT command is issued or whether they are to be reduced to the original allocation before table expansion. (It may be necessary to save the original allocations). The default is "N", which is current processing—not retained.

Also see [“ZEROROWS—Zero data table rows on de-allocation”](#) on page 110.

This option applies to the TSR created by this region/job and all users accessing it (applies only to Release 6.0.3 and above).

MULTITASKING—Multitasking

Specifies whether this region will allow multiple subtasks to access tableBASE. DB2 stored procedures are considered to be multitasking.

This option applies only to batch environments. All other environments are multitasking in their nature, and will not accept this option. The MULTITASKING option can be set to Y to allow multiple subtasks within a region to access tableBASE concurrently. Each subtask may have multiple concurrent subtasks of its own, with no limit to the level of multitasking. The delivered default is N (multitasking not enabled).

This option applies to this region's users, regardless of which TSR they access.

Note: If a batch application attempts to issue a tableBASE command from a second TCB with Multitasking=N, it will abend with an S0C3 and display message: "DKL00496E Multitasking requires TBOPT Multitasking=Y be set".

Note: Please call DataKinetics Technical Support for assistance when writing applications that access tableBASE in a multitasking environment.

MULTOPNX—Multiple Alternate Index

MULTOPNX is a flag indicating whether to allow Multiple Alternate Indexes to a Data Table to be open concurrently. Releases of tableBASE prior to Release 5.0 allowed only one Index to be open. Subsequent Indexes were invoked using the IA command. For more information about Alternate Indexes, see the *tableBASE Concepts and Facilities Guide (Indexes)* and the *tableBASE Programming Guide (various)*.

This option is maintained for compatibility with previous releases, and is not used in tableBASE Version 6. The only valid value for this flag is Y and this is the delivered default.

This option applies to this region's users, regardless of which TSR they access.

Note: For assistance with this default value, please contact tableBASE customer support.

OVERRIDES—Allow Changes to Status Switches

The Status Switches may be changed by the application with the use of the Change Status (CS) line command. Overriding individual Status Switches may be inhibited by the settings of the override controls. Overrides are specified as a string of y's and n's; for example: `yyynynn`. All eight positions must be specified.

The defaults for these values will depend upon the interface, as listed in [Table A-1](#).

Table A-1: Overrides default values

Position	Default	Override setting	Meaning
1	Allow Change To Abend On Errors	Y	The application is allowed to set the ABEND status switch.
		N	The application is not allowed to set the ABEND status switch.
2	Allow Change To Wait For Enqueued Tables	Y	The application is allowed to set the WAIT status switch.
		N	The application is not allowed to set the WAIT status switch.
3	Allow Change To Return Empty Rows In Hash Tables	Y	The application is allowed to set the HASH-EMPTYES-RETURNED status switch.
		N	The application is not allowed to set the HASH-EMPTYES-RETURNED status switch.
4	Allow Change To Permit Implicit Open Of Tables	Y	The application is allowed to set the IMPLICIT OPEN status switch.
		N	The application is not allowed to set the IMPLICIT OPEN status switch.
5	Allow Change To Trace tableBASE Commands	Y	The application is allowed to set the tableBASE TRACE status switch.
		N	The application is not allowed to set the tableBASE TRACE status switch.
6	Reserved for future use; default set to Y.		
7	Reserved for future use; default set to Y.		
8	Reserved for future use; default set to Y.		

These options apply to this region's users, regardless of which TSR they access.

SWITCHES—Status Switches

The Status Switches are a series of one-byte codes for altering the operation of the Application Programming Interface used by your programs when requesting a tableBASE service. Switches are specified as a string of y's and n's; for example: `yyynynnn`. All eight positions must be specified. For information about switch settings, see [Table A-2](#).

The switches are stored in the parameter module for the interface. As each batch job or online task begins, the run-time values will be set according to the values in the parameter module but may be changed using the values in the TBOPT file. For more information, see Parameters in the MVS Batch section of the *tableBASE Installation Guide*.

The application also may change the values with the CS, the Change Status command (unless changing a particular switch is suppressed via the override controls).

The defaults for these values depend upon the interface and are described in the *tableBASE Installation Guide* in the appropriate chapter:

- tableBASE MVS batch
- CICS Interface product
- IMS TM Interface product
- DB2 Stored Procedure Interface product.

Table A-2: Switches default values

Position	Default	Switch setting	Meaning
1	Abend on Errors	Y	Abend processing is to be performed on tableBASE errors 0001-0099 or 1001-1099. Errors 100 to 999 and errors greater than 1099 always abend.
		N	Abend processing is not to be performed on tableBASE errors 0001-0099 or 1001-1099. User programs will handle these return codes.
2	Wait for Enqueued Table	Y	tableBASE is to wait for tables that are enqueued
		N	tableBASE is not to wait for such enqueued tables. In this case tableBASE will terminate with a user 0072 abend if abend processing is enabled, or, will return an error code of 0072: TABLE UNAVAILABLE; NO WAIT in the command area if abend processing has been disabled.
		Note: Waiting in an online environment is normally discouraged. In CICS and IMS, the default is set to N	

Table A-2: Switches default values (Continued)

Position	Default	Switch setting	Meaning
3	Return Empty Rows in Hash Tables	Y	tableBASE is to return empty rows for hash tables
		N	tableBASE is to suppress the return of empty rows for hash tables
4	Allow Implicit Open Of Tables	Y	tableBASE is to automatically open tables for read on first access
		N	tableBASE is to suppress automatic opens; an explicit open command, OR, OW, or IA must be issued to open a table
		Note: We recommend setting Allow Implicit Open of Tables to N, especially in multi-user environments. This gives better control over which tables are opened.	
5	Trace tableBASE Commands	Y	tableBASE automatically records the last ten commands executed per thread. This is done for diagnostic purposes. Contact DKL for more information.
		N	tableBASE does not trace commands
6	Reserved for future use; specify as N.		
7	Reserved for future use; specify as N.		
8	Reserved for future use; specify as N.		

Note:

1. These apply to this region's users, regardless of which TSR they access.
2. The SWITCHES parameter does not apply to the VTS Interface product. For VTS, SWITCHES settings are inherited from the interface used for the applicable application.

STROBE—Strobe Interval

Specify a numeric value, from 0 to 2,147,483,647(2G – 1) to control the intervals at which Table Space statistics report records are to be generated. After the specified number of calls to tableBASE that access the TSR the strobe reporting routine is called.

The delivered default is 0.

A final strobe report will always be produced if the TBTSRPT DD statement is present (even if set to 0). In order to suppress strobe reporting, the TBTSRPT DD statement must be removed from the job. For strobe=0, no intermediate strobe report is produced.

This option applies to the TSR created by this region/job and all users accessing it.

Note: The numeric value must be specified without commas.

SUPPRESS_DUMPS—Suppress tableBASE messages and dumps

Specifies whether to suppress tableBASE messages and dumps when an abend occurs. This option applies only to the CICS environment. Values for this parameter can be NONE or SYMPATHETIC.

The following is a description of each option:

- NONE indicates that all tableBASE messages and dumps will be produced.
- SYMPATHETIC indicates tableBASE error messages and dumps (TBDUMP and tableBASE CICS dumps) will be suppressed if a transaction abends outside tableBASE code (i.e., in the associated application code, or in some other non-tableBASE code). Exceptions are if the tableBASE STATUS_SWITCH is set to Y or the Command_Area abend switch is set to Y and a tableBASE error is encountered—messages and dumps will not be suppressed in these scenarios. Any abends occurring within tableBASE code will still produce the tableBASE error messages and dumps.

The delivered default is SUPPRESS_DUMPS=NONE.

TABLEWAITRC—Table open enqueue wait time

Specify a value to indicate the number of seconds that a user will wait to obtain the MVS enqueue to open a table for read or write before timeout. If the enqueue is not obtained before the timeout, tableBASE will return code 72, or will abend, if the *Abend on Errors* switch is set to Y.

The delivered default is 0 (wait forever).

The Wait for Enqueued Table switch (see [Table A-2](#) on page 102) must be set to Y for this parameter to have any effect.

This option applies to this region's users, regardless of which TSR they access.

TABLEWAITWTO—Table open enqueue report time

Specify a value to indicate the elapsed time before a message will be generated to report that the enqueue has not yet been received.

The delivered default is 30 (0=no messages).

The Wait for Enqueued Table switch (see [Table A-2](#) on page 102) must be set to Y for this parameter to have any effect.

This option applies to this region's users, regardless of which TSR they access.

TPVM—VTS Group Manager Use

Specify a valid VTS Group Manager (TPVM) name to use for all accesses to VTS-TSRs in this jobstep. The name must be pre-defined, and can be up to eight characters in length.

The delivered default is *null* which maps to the *compat* VTS Group Manager.

This option applies to the TSR created by this region/job and all users accessing it.

Note: This parameter applies only if you have licensed the optional VTS Manager product.

TSR_ALGORITHM—Optimize TSR Usage

TSRAlgorithm=P|D allows the TSR Space Manager to determine if TSR space usage should be maximized or performance should be optimized when TSR space is allocated and deallocated for table usage. The TSR space manager is the tableBASE component responsible for tracking TSR space usage and allocating and deallocating TSR space when tables are opened, closed, expanded or reduced in size.

The following is a description of each option:

- P (Performance) indicates that the TSR will be optimized for performance. Space will be assigned to tables within the TSR so as to minimize CPU usage, which may result in a less than optimum use of space.
- D (Default) indicates that there will be no optimization, and there will be no messages regarding optimization. However, messages will be provided regarding current percentage of TSR capacity.

The delivered default is D.

This option applies to the TSR created by this region/job and all users accessing it.

TSR_WARNING_FREQ—Frequency of TSR allocation warnings

Specify a value to indicate the frequency of reports generated when the TSR allocation percentage (as defined by `TSR_WARNING_PCT`) is exceeded. 0 results in a message for every allocation over the specified percentage; 30 results in a wait of at least 30 seconds before repeating the message; 999 results in a wait of just over 15 minutes before repeating the message.

The delivered default is every one second (001).

This option applies to this region's users, regardless of which TSR they access. (Different regions accessing a VTS-TSR can set their own thresholds.)

TSR_WARNING_PCT—Percentage of TSR allocation for warning activation

Specify a value to indicate the percentage of TSR allocation allowed before a message will be generated to report that the allocation percentage has been exceeded. The message will be repeated on every allocation of this percentage, subject to the `TSR_WARNING_FREQ` parameter.

The delivered default is 85 (85%).

This option applies to this region's users, regardless of which TSR they access. (Different regions accessing a VTS-TSR can set their own thresholds.)

TSRACCESS—Read Only (RO) or Read Write (RW) TSR access

Specifies whether the local TSR or VTS-TSR defined in this jobstep will have Read-Only access or Read/Write access. This option applies to a batch or VTS environment and can be set to RO for Read-Only access; or RW for Read/Write access.

If RO is specified, then the LDS parameter must be set to Y.

The delivered default is *RW*.

This option applies to the TSR created by this region/job and all users accessing it.

Note: This parameter applies only if you have licensed the optional VTS Manager product.

TSRSIZE—tableSPACE Region Size

The `TSRSIZE` parameter is an integer representing the amount of storage to be used for the TABLE SPACE REGION (TSR). Version 6 uses Data Spaces for all TSRs, whether local or VTS. This ensures that tableSpace memory requirements do not affect other memory requirements in the region. The format for `TSRSIZE` is:

- nnnnnnnn = bytes
- nnnnnnnK = kilobytes
- nnnnM = megabytes
- nG = gigabytes

Value recommendations

The delivered default value for the TSRSIZE is 10M for all interfaces. The current minimum size of a TSR is 28 KB (but this may change in subsequent maintenance releases). The maximum size of a TSR is 2G (or 2048M or 2097152K). TSRSIZE only accepts up to eight digits, thus you cannot specify 2G as TSRSIZE=2147483648.

Note: If you specify a value of 0, the minimum size will be substituted; if you specify a value of between 1 and the minimum, an error code will be returned.

Other considerations

If you are upgrading from an earlier release, you must ensure you are allocating enough space for all tables open simultaneously in the region. A conservative TSR size for Version 6 would be the sum of the prior release TSR size and the size of TBTSLIB (the Rollout library from Release 5).

If you are upgrading from an earlier tableBASE release, you can determine the current TSR size in one of two ways:

1. use the LT command in a tableBASE driver program—batch program DK1TDRV / TBDRIVER, CICS transaction TBDR,
2. browse strobe reports produced by your currently-installed tableBASE release.

Failure to set your TSR size parameter appropriately can produce the following error:

Error 92 There is insufficient tableSPACE region available.
 Increase TSR size.

Notes:

1. TSREGION and TSR parameters are maintained for backwards compatibility and are treated as equivalents to TSRSIZE.
2. There is a possibility that your system programmers have established limits on the maximum size or the maximum number of Data Spaces. All local TSRs are allocated as SCOPE=SINGLE Data Spaces; VTS-TSRs are allocated as SCOPE=ALL Data Spaces.
3. For users upgrading to Version 6, batch now requires the use of the TSRSIZE parameter. It is no longer sufficient to use the REGION= parameter in the JOB, or any other method. You can use the same parameter value for TSRSIZE that you did for REGION. For instance, if the REGION value was 32M or less, the batch job could have getmained up to 40M of space to hold tableBASE tables (32M above the line, and 8M below). If the REGION value was larger than 32M, the specified value plus 8M should be sufficient. Otherwise, see the recommendations above.

This option applies to the TSR created by this region/job and all users accessing it.

USEREXITS—User Exits

Specifies whether User Exits are used. Setting the tableBASE execution-time parameter USEREXITS=Y causes the DK1TX072 module to be dynamically loaded during the initialization of tableBASE. For more information, see *tableBASE User Exits* in the *tableBASE Administration Guide*.

This option applies to this region's users, regardless of which TSR they access.

VTSFIRST, VTSLAST—VTS Search Sequence

VTSFIRST and VTSLAST are two parameters that can be used to control the sequence in which tableBASE searches libraries when DataKinetics VTS product is installed.

VTSFIRST is the name of the VTS-TSR that is to be searched for tables before searching any libraries listed in the tableBASE library-list. VTSLAST is the name of the VTS-TSR that is to be searched for tables after searching any libraries listed in the tableBASE library-list.

If you wish to override the installation default for this parameter, it is strongly recommended that you use the TBOPT DD statement described later in this chapter to do so. If you set the VTSFIRST parameter by tailoring the installation parameters at installation time, problems may occur. For example, any processes that operate on the library copy of a table that is open in VTS may abend with a tableBASE error code of 3 or may return unpredictable results.

Note: If VTSFIRST or VTSLAST is specified in the default parameters (DK1Txx34), setting VTSFIRST=0 or VTSLAST=0 in the TBOPT dataset for the job turns the respective parameters off.

The delivered defaults are VTSFIRST, VTSLAST set to null strings.

This option applies to this region's users, regardless of which TSR they access.

Note: These parameters apply only if you have licensed the optional DataKinetics VTS product. Do not change the delivered defaults supplied for these parameters if you have not licensed VTS.

VTSNAME—specifying the name of a VTS-TSR

The VTSNAME option specifies the name to be assigned to the VTS-TSR when it is initialized by the VTS Agent. The name is used by other regions to access the shared TSR.

Notes:

1. Starting in Release 6.0.3, VTSNAME can be 1 to 8 characters. The naming restrictions are identical to those for DDNAMEs in IBM JCL. VTSNAMEs of 1 to 4 characters created in prior releases are still supported.
2. Care must be taken when assigning VTSNAMEs, particularly if VTSNAMEs are to be used in the ML List— the combined length of the VTSPREFIX and the VTSNAME cannot exceed eight characters (see “[VTSPREFIX—VTS prefix definition](#)” below).
3. For compatibility with previous releases, where it was specified in the TBOPTV dataset of regions accessing tableBASE, VTSNAME can be used to specify the default VTS-TSR accessed by the TBCALLV and TBASEV interfaces. Note that the tableBASE options for applications using TBCALLV and TBASEV may have changed. See the "Environmental interfaces" chapter in the *tableBASE Programming Guide* for additional information.

This option applies to the TSR created by this region/job and all users accessing it.

VTSPREFIX—VTS prefix definition

The VTSPREFIX option specifies a prefix that will be used by tableBASE to denote a VTS name within the ML list.

The delivered default is VTS:. There can be 0-3 characters before the colon; the colon is mandatory. Example valid values:

- VTS:
- V:
- :

The maximum number of characters in total for the VTSPREFIX and the actual VTS name is eight. Therefore, a VTS name of eight characters could never be used in conjunction with the ML command. Example valid combinations:

- VTS:SHAR
- V:SHARE1
- :SHARE99

This option applies to this region's users, regardless of which TSR they access.

ZEROROWS—Zero data table rows on de-allocation

Related to the MTRETAIN parameter (see [“MTRETAIN—Retain Rows and Index Areas”](#) on page 99)—applies only when MTRETAIN = N.

ZEROROWS=Y|N determines whether the data rows area should be zeroed when it is deallocated. The default is Y, which is current processing. Note that the index area is never zeroed, except for hash indexes.

This option applies to the TSR created by this region/job and all users accessing it (applies only to Release 6.0.3 and above).

Appendix B

Converting tableBASE libraries

The tableBASE libraries for Version 6 are significantly different from those of previous releases. As a result, libraries formatted under tableBASE Version 4 and 5 can not be used with Version 6, and libraries formatted under Version 6 cannot be used with tableBASE Version 4 or 5.

Converting libraries to the Version 6 format can be done in one of two ways: use the utilities provided with Version 6 to convert your libraries, or use the utilities provided with Version 6 along with the Library Bridge. For a description of converting libraries using the Library Bridge, please consult the *tableBASE Library Bridge Manual, Release 5.B*.

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

- Created only by converting a Version 5 library
- Convert 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
- Can *NOT* be accessed by any tableBASE 6.0 releases prior to Release 6.0.2

Note: All releases of tableBASE 6.0.3 will continue to provide support for Bridge libraries. However, DataKinetics Ltd. cannot guarantee support in future releases of tableBASE beyond 6.0.3. 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, Version 6 directory structure, indexes are stored on the library, no paged tables
- Can only be accessed by tableBASE release 6.0.x (or later)

Note: Version 6 Transition Libraries must be converted using the DK1TCNV utility to Version 6 libraries, or to BRIDGE libraries (or to Version 5 libraries, if they have not been expanded), if so desired. The library and can be identified with the DK1TLCHK utility. For more information see [“Converting a tableBASE library”](#) on page 114.

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
- Have no fixed Block Size after conversion to Version 6 (this feature does not apply to libraries converted from previous library versions to the Version 6 format).
- Have Version 6 directory structure— no paged tables
- Provide better performance than Version 4 or Version 5 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.

Note: Any attempt to define different libraries in a single conversion job will not be successful.

Note: You cannot change the library block size during the conversion process. You must leave the block size at 3120.

Table B-1: Compatibility of library versions with tableBASE code

tableBASE version/release	Libraries it can access
tableBASE 5.x	V5
Library Bridge (tableBASE 5.B)	V5, V5 Bridge
tableBASE 6.0.3	V5 Bridge, V6Trans, V6

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

Converting a tableBASE library

Note: It is recommended that a backup copy of the library be made before beginning the conversion process.

The Library Conversion Utility DK1TCNV (**your.prefix.CNTL**) is used to convert tableBASE libraries between Version 5 (V5), Library Bridge (Bridge), Version 6 Transition (V6Trans), and Version 6 (V6).

Note: Version 6 libraries cannot be converted to any other library type.

This section identifies and describes DK1TCNV input parameters, shows a variety of conversion functions, describes a sample conversion and ends with example JCL for executing DK1TCNV.

Note: DK1TCNV does not require that source and target libraries be both BDAM or both VSAM, so it is possible to change between VSAM and BDAM while converting.

The DK1TCNV utility has free-format input using keywords that invoke four library conversion functions. The general format is:

```
COMMAND KEYWORD1=VALUE KEYWORD2=VALUE;
```

KEYWORD=VALUE is the representation of a keyword-value combination and may be repeated for each parameter of the command.

The following rules apply:

1. Each command sequence must start on a new line.
2. A command sequence must be terminated by a semicolon (;). Anything following a semicolon is ignored.
3. The command and each keyword-value combination must be followed by at least one blank.
4. Keyword-value combinations may appear in any order.
5. Many library conversion requests may be entered in the same input file.
6. Columns 73-80 of a statement are ignored.
7. An asterisk (*) in column one means that the line is a comment.

Keyword parameters

Input parameters for DK1TCNV are keywords with assigned values:

FROMLIB	Source library DDname.
TOLIB	Target library DDname.
TOFORMAT	Either V5, BRIDGE, V6 or V6TRANS.
TBLPREFIX	Four alphanumeric characters used when renaming invalid tables. DK1TCNV will rename any tables with invalid names (i.e. all blanks, hex zeroes or hex 'FF'). The new name will be the TBLPREFIX value followed by a numeric suffix starting from 0001. The numeric suffix is in character form, not binary. The default TBLPREFIX is V5BR.

Note: DK1TCNV automatically detects the format of the source library.

Conversion utility commands

There are four conversion utility commands:

- [CONVERT](#)
- [REVERT](#)
- [FIXLIB](#)
- [DIAGNOSE](#)

Note: The target library for any conversion must be a newly defined library. The source and target must be different libraries.

CONVERT

The tableBASE library is converted to the format specified with the TOFORMAT parameter, the version of the source library is detected automatically. There are four types of libraries: Version 5 (V5), Library Bridge (Bridge), Version 6 Transition (V6Trans) and Version 6 (V6). For more information on these library types, see [“Description of tableBASE library versions”](#) on page 111.

It is used to perform to do the following:

1. Convert from V5 to BRIDGE, V6TRANS or V6.
2. Convert from BRIDGE or V6TRANS to V6.
3. Convert from BRIDGE to V6TRANS.

4. Convert from V6TRANS to BRIDGE.

Examples:

```
CONVERT FROMLIB = MAINLIB TOLIB = TESTLIB TOFORMAT = BRIDGE;  
CONVERT FROMLIB = MAINLIB TOLIB = MAINLIB2 TOFORMAT = V6;  
CONVERT FROMLIB = OLDLIB TOLIB = JANLIB2 TOFORMAT = BRIDGE;
```

REVERT

Backward conversion from BRIDGE or V6TRANS to V5. The target library will be in V5 format.

Example:

```
REVERT FROMLIB = MAINLIB TOLIB = NEWLIB;
```

FIXLIB

Detect and fix V5 problems which conversion would also detect and fix. The source must be a Version 5 library. The target library remains in a Version 5 format.

Examples:

```
FIXLIB FROMLIB=MAINLIB TOLIB=PRODLIB;  
FIXLIB FROMLIB=OLDLIB TOLIB=MAINLIB2 TBLPREFIX=V5@@;
```

DIAGNOSE

Diagnose a tableBASE library.

Example:

```
DIAGNOSE FROMLIB = MAINLIB;
```

Completion codes

Completion codes listed in [Table B-2](#) are set to indicate whether problems were encountered. If any of the commands or parameters are invalid, none of the commands will be processed, errors will be displayed in TBREPORT and a completion code of 16 will be returned. If more than one completion code applies, the highest code will be returned and complete details will be found in TBREPORT.

Table B-2: DK1TCNV completion codes

Completion Code	Description
0	All library conversion commands were successfully processed.
4	A warning was issued for one or more library conversion commands, but the conversion(s) was successful.
16	One or more errors were encountered which prevented the library conversion from proceeding. Warnings may also have been issued.

Description of Conversion Process

Below is a sample output from a typical conversion job.

```

COMMAND VALIDATION
-----
  CMD |  COMMAND      |  FROMLIB  TOLIB      TOFORMAT  TBLPREFIX
-----
 0001 |  CONVERT        |  SRCLIB   TARGET      V6TRANS
-----
TOTAL NUMBER OF COMMANDS READ: 0001

DETECTED COMMANDS WITH THE FOLLOWING INFO/ERRORS
CMD 0001 : KEYWORD "TBLPREFIX" NOT SUPPLIED; DEFAULT VALUE "V5BR " USED

TCNV:PROCESSING COMMAND # 0001
  COMMAND = CONVERT
  FROMLIB = SRCLIB
  TOLIB   = TARGET
  TOFORMAT = V6TRANS
  TBLPREFIX = V5BR

DIAGNOSING SOURCE "SRCLIB "
LCHK:LIBRARY FORMAT IS "V5 "; LIBRARY DSN IS "TBTST.V5B0.V510LIB6
BLOCK X00000000: INVALID NAME " " FOUND IN DIR BLOCK
BLOCK X00000000: INVALID NAME " " FOUND IN DIR BLOCK
BLOCK X00000000: INVALID NAME ":TMPNAME" FOUND IN DIR BLOCK
BLOCK X00000031: INVALID NUMBER OF TABLE ENTRIES. EMPTY DIRECTORY BLOCK
BLOCK X0000005A: INVALID NUMBER OF TABLE ENTRIES. EMPTY DIRECTORY BLOCK
LCHK:TOTAL 00000005 WARNING(S) FOUND IN THE DIAGNOSED LIBRARY
LCHK:DIAGNOSTICS ENDED SUCCESSFULLY

T1182:STARTING CONVERSION FROM "SRCLIB " TO "TARGET " LIBRARY...

ALL TABLES HAVE BEEN COPIED TO TARGET LIBRARY

REPAIRING INVALID TABLE NAMES:
INVALID TBL NAME LOW-VALUES REPLACED WITH "V5BR0001"
INVALID TBL NAME HIGH-VALUES REPLACED WITH "V5BR0002"
INVALID TBL NAME ":TMPNAME" DELETED
INVALID NAME REPAIR ENDED SUCCESSFULLY

REPAIRING DIRECTORY BLOCKS
  EMPTY DIRECTORY BLOCK AT RBN 00000049 HAS BEEN FREED
  EMPTY DIRECTORY BLOCK AT RBN 00000090 HAS BEEN FREED
REPAIR ENDED SUCCESSFULLY

REORGANIZING TABLES IN LIBRARY
  TABLE NAME ORG MTHD INDEX SMC GEN CONVERT ACTION
  DBPR        D  B  P  R  1  SORTED
  HHPR        H  H  P  R  1  CHECKED
  HHTP1       H  H  T  P  1  CONVERTED TO INDEX=P,SMC=R
  V5BR0001    S  B  T  R  1  CHECKED
  V5BR0002    H  H  T  R  1  CHECKED
REORGANIZATION ENDED SUCCESSFULLY

T1182:CONVERSION PROCESS ENDED SUCCESSFULLY

DIAGNOSING TARGET "TARGET "
LCHK:LIBRARY FORMAT IS "V6TRANS "; LIBRARY DSN IS "TBTST.V5B0.V6LG
LCHK:DIAGNOSTICS ENDED SUCCESSFULLY

TCNV:CURRENT COMMAND SUCCESSFULLY EXECUTED

```

Figure B-1: Sample Conversion Output

The messages displayed in the output, as referenced by the numbers in the sample output, are described below. They represent the basic processing steps performed by the conversion program:

1. Identification of the command to be processed.
2. Identification of the table prefix to be used for renaming invalid table names.
3. Diagnosis of the source library to be converted to check the integrity of the internal structure.
4. Copying of all tables from the source library to the target library.
5. Repairing of invalid tables names in the target library using the default table name prefix.
6. Repairing of linkage to empty directory blocks in the target library.
7. Reorganization of tables in target library to make internal format changes between V5 and V6 libraries like block size changes, converting of paged tables to pointer tables and ensuring that the order of data is similar to that in V5.
8. Diagnosis of the converted target library to check the integrity of the internal structure.

JCL for creating BRIDGE libraries

Converting to BRIDGE libraries

Here is a sample JCL (BDAM) for the execution of DK1TCNV to create a BRIDGE library (can be found in **your.prefix.CNTL**).

CNVBTO5B (BDAM)

```

/* INSERT YOUR JOB CARD HERE
/*
/*****
/* CONVERTS A BDAM LIBRARY FROM EITHER RELEASE 5 OR V6 TRANSITION
/* TO BRIDGE FORMAT
/* CHANGE LINES TO YOUR DATASET NAMES <====
/*****
/* CREATE STEP ALLOCATES THE TARGET BDAM BRIDGE TABLEBASE LIBRARY
/*-----
//CREATE EXEC PGM=IEFBR14
//NEWLIB DD DSN=**YOUR.BRIDGE.TABLEBASE.LIBRARY**, <====
// SPACE=(3120,NNNNN),UNIT=SYSDA, <====
// DCB=(BLKSIZE=3120,LRECL=3120,RECFM=F,DSORG=PS),
// DISP=(,CATLG)
/*****
/* DK1TCNV STEP CONVERTS LIBRARY TO BRIDGE FORMAT
/* THE FORMAT OF YOUR "OLDLIB" IS AUTOMATICALLY DETECTED IN THE
/* CONVERSION PROCESS AND WILL ONLY ACCEPT EITHER RELEASE 5 OR
/* V6 TRANSITION LIBRARIES
/*-----
//DK1TCNV EXEC PGM=DK1TCNV
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <====
//OLDLIB DD DSN=**YOUR.OLD.TABLEBASE.LIBRARY**,DISP=OLD <====
//NEWLIB DD DSN=**YOUR.BRIDGE.TABLEBASE.LIBRARY**,DISP=OLD <====
//TBREPORT DD SYSOUT=*
//CNTLCARD DD *
CONVERT FROMLIB=OLDLIB TOLIB=NEWLIB TOFORMAT=BRIDGE;
/*

```

Converting to V6 transition libraries

Here are two samples of JCL (BDAM and VSAM) for the execution of DK1TCNV to create a V6 Transition library.

CNVBTO6T (BDAM)

```

/* INSERT YOUR JOB CARD HERE
/*
/******
/* CONVERTS A BDAM LIBRARY FROM EITHER RELEASE 5 OR BRIDGE TO
/* V6 TRANSITION FORMAT
/* CHANGE LINES TO YOUR DATASET NAMES <====
/******
/* CREATE STEP ALLOCATES THE TARGET BDAM V6 TRANSITION LIBRARY
/*-----
//CREATE EXEC PGM=IEFBR14
//NEWLIB DD DSN=**YOUR.V6TRANS.TABLEBASE.LIBRARY**, <====
// SPACE=(3120,NNNNN),UNIT=SYSDA, <====
// DCB=(BLKSIZE=3120,LRECL=3120,RECFM=F,DSORG=PS),
// DISP=(,CATLG)
/******
/* DK1TCNV STEP CONVERTS LIBRARY TO V6 TRANSITION
/* THE FORMAT OF YOUR "OLDLIB" IS AUTOMATICALLY DETECTED IN THE
/* CONVERSION PROCESS AND WILL ONLY ACCEPT EITHER RELEASE 5 OR BRI
/* LIBRARIES.
/*-----
//DK1TCNV EXEC PGM=DK1TCNV
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <====
//OLDLIB DD DSN=**YOUR.OLD.TABLEBASE.LIBRARY**,DISP=OLD <====
//NEWLIB DD DSN=**YOUR.V6TRANS.TABLEBASE.LIBRARY**,DISP=OLD <====
//TBREPORT DD SYSOUT=*
//CNTLCARD DD *
CONVERT FROMLIB=OLDLIB TOLIB=NEWLIB TOFORMAT=V6TRANS;
/*

```

CNVVTO6T (VSAM)

```

/* INSERT YOUR JOB CARD HERE
/*
/******
/* CONVERTS A VSAM LIBRARY FROM EITHER RELEASE 5 OR BRIDGE
/* TO V6 TRANSITION FORMAT
/* CHANGE LINES TO YOUR DATASET NAMES <====
/******

/* CREATE STEP ALLOCATES THE TARGET VSAM V6 TRANSITION LIBRARY
/*-----
//CREATE EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=*
//SYSIN DD *
DELETE (**YOUR.V6TRANS.TABLEBASE.LIBRARY**) CLUSTER <====
SET MAXCC=0
DEFINE CLUSTER (NAME(**YOUR.V6TRANS.TABLEBASE.LIBRARY**) - /*<==== */
RECSZ(3120 3120) REC(780) SHR(3) -
CISZ(3584) NUMBERED SPEED REUSE -
VOLUME(*****)) /* <==== */
/*
/******
/* DK1TCNV STEP CONVERTS LIBRARY TO V6 TRANSITION FORMAT
/* THE FORMAT OF YOUR "OLDLIB" IS AUTOMATICALLY DETECTED IN THE
/* CONVERSION PROCESS AND WILL ONLY ACCEPT EITHER RELEASE 5 OR BRIDGE
/* FORMAT LIBRARIES
/*-----
//DK1TCNV EXEC PGM=DK1TCNV
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <====
//OLDLIB DD DSN=**YOUR.OLD.TABLEBASE.LIBRARY**,DISP=OLD <====
//NEWLIB DD DSN=**YOUR.V6TRANS.TABLEBASE.LIBRARY**,DISP=OLD <====
//TBREPORT DD SYSOUT=*
//CNTLCARD DD *
CONVERT FROMLIB=OLDLIB TOLIB=NEWLIB TOFORMAT=V6TRANS;
/*

```

Converting to V6 libraries

Here are two samples of JCL (BDAM and VSAM) for the execution of DK1TCNV to create a V6 library.

CNVBTO6 (BDAM)

```

/* INSERT YOUR JOB CARD HERE
/*
/******
/* CONVERTS A BDAM LIBRARY FROM EITHER RELEASE 5, BRIDGE OR V6
/* TRANSITION TO RELEASE 6 FORMAT
/* CHANGE LINES TO YOUR DATASET NAMES <====
/******
/* CREATE STEP ALLOCATES THE TARGET BDAM RELEASE 6 TABLEBASE LIBRARY
/*-----
//CREATE EXEC PGM=IEFBR14
//NEWLIB DD DSN=**YOUR.V6.TABLEBASE.LIBRARY**, <====
// SPACE=(3120,NNNNNN),UNIT=SYSDA, <====
// DCB=(BLKSIZE=3120,LRECL=3120,RECFM=F,DSORG=PS),
// DISP=(,CATLG)
/******
/* DK1TCNV STEP CONVERTS LIBRARY TO RELEASE 6 FORMAT
/* THE FORMAT OF YOUR "OLDLIB" IS AUTOMATICALLY DETECTED IN THE
/* CONVERSION PROCESS AND WILL ONLY ACCEPT EITHER RELEASE 5, BRIDGE OR
/* V6 TRANSITION LIBRARIES
/*-----
//DK1TCNV EXEC PGM=DK1TCNV
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <====
//OLDLIB DD DSN=**YOUR.OLD.TABLEBASE.LIBRARY**,DISP=OLD <====
//NEWLIB DD DSN=**YOUR.V6.TABLEBASE.LIBRARY**,DISP=OLD <====
//TBREPORT DD SYSOUT=*
//CNTLCARD DD *
CONVERT FROMLIB=OLDLIB TOLIB=NEWLIB TOFORMAT=V6;
/*

```

CNVVTO6 (VSAM)

```

    /* INSERT YOUR JOB CARD HERE
    /*
    /******
    /* CONVERTS A VSAM LIBRARY FROM EITHER RELEASE 5, BRIDGE OR V6
    /* TRANSITION TO RELEASE 6 FORMAT
    /* CHANGE LINES TO YOUR DATASET NAMES <=====
    /******
    /* CREATE STEP ALLOCATES THE TARGET VSAM RELEASE 6 TABLEBASE LIBRARY
    /*-----
    //CREATE EXEC PGM=IDCAMS
    //SYSPRINT DD SYSOUT=*
    //SYSOUT DD SYSOUT=*
    //SYSIN DD *
    DELETE (**YOUR.V6.TABLEBASE.LIBRARY**) CLUSTER <=====
    SET MAXCC=0
    DEFINE CLUSTER (NAME (**YOUR.V6.TABLEBASE.LIBRARY**) - /* <===== */
    RECSZ(3120 3120) REC(780) SHR(3) -
    CISZ(3584) NUMBERED SPEED REUSE -
    VOLUME (*****) ) /* <===== */
    /*
    /******
    /* DK1TCNV STEP CONVERTS LIBRARY TO RELEASE 6
    /* THE FORMAT OF YOUR "OLDLIB" IS AUTOMATICALLY DETECTED IN THE
    /* CONVERSION PROCESS AND WILL ONLY ACCEPT EITHER RELEASE 5, BRIDGE OR
    /* V6 TRANSITION LIBRARIES
    /*-----
    //DK1TCNV EXEC PGM=DK1TCNV
    //STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <=====
    //OLDLIB DD DSN=**YOUR.OLD.TABLEBASE.LIBRARY**,DISP=OLD <=====
    //NEWLIB DD DSN=**YOUR.V6.TABLEBASE.LIBRARY**,DISP=OLD <=====
    //TBREPORT DD SYSOUT=*
    //CNTLCARD DD *
    CONVERT FROMLIB=OLDLIB TOLIB=NEWLIB TOFORMAT=V6;
    /*
  
```

Reverting to V5 libraries

Here is a sample JCL (BDAM) for the execution of DK1TCNV to revert back to a V5 library.

CNVBTO5 (BDAM)

```

/* INSERT YOUR JOB CARD HERE
/*
/*****
/* REVERTS A BDAM LIBRARY FROM EITHER BRIDGE OR V6 TRANSITION
/* BACK TO RELEASE 5
/* CHANGE LINES TO YOUR DATASET NAMES <====
/*****
/* CREATE STEP ALLOCATES THE TARGET BDAM RELEASE 5 TABLEBASE LIBRARY
/*-----
//CREATE EXEC PGM=IEFBR14
//NEWLIB DD DSN=**YOUR.V5.TABLEBASE.LIBRARY**, <====
// SPACE=(3120,NNNNNN),UNIT=SYSDA, <====
// DCB=(BLKSIZE=3120,LRECL=3120,RECFM=F,DSORG=PS),
// DISP=(,CATLG)
/*****
/* DK1TCNV STEP REVERTS LIBRARY TO RELEASE 5 FORMAT
/* THE FORMAT OF YOUR "OLDLIB" IS AUTOMATICALLY DETECTED IN THE
/* CONVERSION PROCESS AND WILL ONLY ACCEPT BRIDGE OR V6 TRANSITION
/* LIBRARIES.
/*-----
//DK1TCNV EXEC PGM=DK1TCNV
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD <====
//OLDLIB DD DSN=**YOUR.OLD.TABLEBASE.LIBRARY**,DISP=OLD <====
//NEWLIB DD DSN=**YOUR.V5.TABLEBASE.LIBRARY**,DISP=OLD <====
//TBREPORT DD SYSOUT=*
//CNTLCARD DD *
REVERT FROMLIB=OLDLIB TOLIB=NEWLIB;
/*

```

Virtual memory requirements

The DK1TCNV utility requires sufficient virtual memory (specified using the REGION parameter within DK1TCNV) to load the program and to read parts of the library into memory. The amount of virtual memory required can be calculated by using the following formulas and using the result that is largest:

Region size (in bytes) = Number-of-library-blocks * 16 + 3120 + 3120 + 512000 or

Region size (in bytes) = Largest-table-size * 2 + 512000
(where table-size = row-size * number-rows)

Conversion notes

The library conversion process introduces some changes:

- 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.

- Paged tables that are converted will no longer be SMC=Paged, Index=True but rather SMC=Resident, Index=Pointer. These tables remain Resident/Pointer/Hash even if the library is converted back to the Version 5 format.
- A larger TSR may be required for tables opened from a converted library which were converted from SMC=Paged. When such a table is opened, the entire table must now fit into the TSR and not just a single 3120-byte page at a time. This may be the largest impact in the migration to Version 6. Most installations however, never use Paged tables and if this is the case no change in TSR size is anticipated.
- Library expansion must be done using the corresponding version of TBEXEC. For example if a Version 5 library must be expanded, use the TBEXEC provided with Version 5. Note that the Library Bridge version of TBEXEC will create a Version 5 library, which must, in turn, be converted to Bridge. Version 5 expansion restrictions still apply (see “[tableBASE version 5 library expansion limits](#)” on page 127).
- In order to define a library with Version 5 format, run the TBEXEC using Version 5 loads. In order to define a library with Version 6 format, run the TBEXEC using Version 6 loads.

tableBASE version 5 library expansion limits

To find the expansion limit, start by finding the row into which your source library size would fall (see [Table B-3](#)). For example, if your source library was 100,000 blocks or 700 cylinders then you would be in the first row under the Lower Limit. The expansion limit would then be the number of blocks or cylinders listed under the Upper Limit for that row. In the example given, the expansion limit would be 168,000 blocks or 746 cylinders. This means that your target library must be less than 168,000 blocks or 746 cylinders.

Table B-3: Version 5 expansion limits

Lower limit 3120 byte BLOCKS	Upper limit 3120 byte BLOCKS	FSDs	BDAM CYLS Lower limit	BDAM CYLS Upper limit	VSAM CYLS Lower limit	VSAM CYLS Upper limit
8	168,000	7	1	746	1	861
168,001	192,000	8	747	853	862	984
192,001	216,000	9	854	960	985	1107
216,001	240,000	10	961	1066	1108	1230
240,001	264,000	11	1067	1173	1231	1353
264,001	288,000	12	1174	1280	1354	1476
288,001	312,000	13	1281	1386	1477	1600
312,001	336,000	14	1387	1493	1601	1723
336,001	360,000	15	1494	1600	1724	1846
360,001	384,000	16	1601	1706	1847	1969
384,001	408,000	17	1707	1813	1970	2092
408,001	432,000	18	1814	1920	2093	2215
432,001	456,000	19	1921	2026	2216	2338
456,001	480,000	20	2027	2133	2339	2461
480,001	504,000	21	2134	2240	2462	2584
504,001	528,000	22	2241	2346	2585	2707
528,001	552,000	23	2347	2453	2708	2830
552,001	576,000	24	2454	2560	2831	2953
576,001	600,000	25	2561	2666	2954	3076
600,001	624,000	26	2667	2773	3077	3200
624,001	648,000	27	2774	2880	3201	3323

Table B-3: Version 5 expansion limits (Continued)

Lower limit 3120 byte BLOCKS	Upper limit 3120 byte BLOCKS	FSDs	BDAM CYLS Lower limit	BDAM CYLS Upper limit	VSAM CYLS Lower limit	VSAM CYLS Upper limit
648,001	672,000	28	2881	2986	3324	3446
672,001	696,000	29	2987	3093	3447	3569
696,001	720,000	30	3094	3200	3570	3692
720,001	744,000	31	3201	3306	3693	3815
744,001	768,000	32	3307	3413	3816	3938
768,001	792,000	33	3414	3520	3939	4061
792,001	816,000	34	3521	3626	4062	4184
816,001	840,000	35	3627	3733	4185	4307
840,001	864,000	36	3734	3840	4308	4430
864,001	888,000	37	3841	3946	4431	4553
888,001	912,000	38	3947	4053	4554	4676
912,001	936,000	39	4054	4160	4677	4800
936,001	960,000	40	4161	4266	4801	4923
960,001	984,000	41	4267	*4369	4924	5046
984,001	1,006,000	42			5047	5169
1,006,001	1,030,000	43			5170	5292
1,030,001	1,054,000	44			5293	5415
1,054,001	1,078,000	45			5416	5538
1,078,001	1,102,000	46			5539	5661
1,102,001	1,126,000	47			5662	5784
1,126,001	1,150,000	48			5785	5907
1,150,001	1,174,000	49			5908	6030

Note: * For 3120 blocksize the maximum size BDAM = 983,025 blocks / 65,535 tracks / 4369 cyls.(BDAM size stored in half-word)

Limits for conversion back and forth between V5 and V6 (3120 blocksize libraries) is 836,550 blocks / 3718 cyls BDAM / 4290 cyls VSAM.

These restrictions to the expansion of tableBASE libraries do not apply to libraries in the Version 6 format. Once converted, Version 6 format libraries may be expanded. However, expansion will mean that they can no longer be converted back to the Version 5 format.

DK1TLCHK utility

The Library Version Identification Utility, DK1TLCHK, is a diagnostic tool used to identify the version number of, and any structural problems with a tableBASE library (see sample JCL in **your.prefix.CNTL**).

It may be used if a tableBASE application encounters an error or abend 0062 (version incompatibility error), or an S001 abend (non-compatible versions of tableBASE attempting to access a Version 6 or Bridge Library).

Compatible versions of code and libraries are described in [Table B-1](#) on page 113. Other combinations will produce errors. For example, Version 6 code trying to read a Version 5 library, will return error code 62; Version 5.0.2 or 5.1 code trying to read a Bridge or Version 6 library will abend with a S001.

The JCL to run DK1TLCHK is:

```
/*  INSERT YOUR JOB CARD HERE
/*
/******
/*  TABLEBASE LIBRARY IDENTIFICATION
/*  IDENTIFIES THE FORMAT OF THE TABLEBASE LIBRARY
/*  CHANGE LINES TO YOUR DATASET NAMES      <=====
/******
//LIBCHECK EXEC PGM=DK1TLCHK,PARM=CHECKLIB
//STEPLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.TBASE.LOAD      <=====
//TBREPORT DD SYSOUT=*
//CHECKLIB DD DISP=SHR,DSN=*YOUR.PREFIX*.MAINLIB      <=====
```


Appendix C

TBOPT dataset coding

The TBOPT dataset can be a sequential file, a member of a data set, or, for CICS, a VSAM dataset. The TBOPT dataset can be specified for all interfaces, including VTS. The dataset must contain fixed-length 80-byte records.

Note: The TBOPTV functionality has been integrated into TBOPT, allowing for a single source of run-time parameter input. TBOPTV is still maintained for backwards compatibility. If both TBOPT and TBOPTV are used, TBOPT is read first.

The data in TBOPT uses the same parameter names and values as are coded on the TBOPTGEN macro for the defaults, with the exception of LIB-LIST. TBOPT uses LIBNN to specify tableBASE libraries to update the tableBASE Library List.

Each parameter is entered on a single line in the dataset. The parameter may begin in any column. A line beginning with an asterisk (*) denotes a comment. Comments may also be added after the parameter value. A semicolon may be used to indicate line end. Comments may follow the semi-colon.

Although each region may have defined its own TBOPT dataset, all regions can share a sequential DASD dataset, and CICS regions can share a VSAM TBOPT dataset.

A sample TBOPT dataset for a batch region follows:

```
//TBOPT DD *  
* A leading asterisk denotes a comment  
ListOptions=Y  
TSRegion = 12M  
MAXNMTAB=500  
LIB01 = TESTLIB  
LIB02 = MAINLIB  
/*
```

Note: ListOptions=Y is handy for diagnostic purposes; TESTLIB is first for batch testing.

Note: With the exception of the LISTOPTIONS parameter, parameters must appear only once in the TBOPT file. The form KEYWORD=* indicates that the site default is to be used. The form KEYWORD=0 indicates that the default value of a parameter that takes a character string be nullified.

Appendix D

tableBASE LPA-eligible programs

Table D-1 lists modules that can be stored in the Link Pack Area:

Table D-1: LPA-eligible modules

DK1BBASE	TBDKACTN
DK1CBASE	TBDKAPHK
DK1DBASE	TBDKAPPL
DK1IBASE	TBDKEDIT
DK1TCIN	TBDKFDHK
DK1TCOBF	TBDKHELP
DK1TCRM	TBDKIDEN
DK1TDRVC	TBDKMENU
DK1TNAME	TBDKMSGR
DK1TNUCL	TBDKMSTK
DK1TPCRM	TBDKPFKS
DK1TROT B	TBDKSITE
DK1TROT C	TBDKUSID
DK1TRSTA	TBDKUTHK
DK1TSHUT	TBROOTC
DK1VAGNT	TBROOTV
DK1VBASE	

Note: Your system may have only some of the modules in the list.

Appendix E

Source modules

After the media unload, the tableBASE source modules reside in the dataset:

your.prefix.SRC (or whatever you name the source library)

Table E-1 lists the members of this dataset.

Table E-1: TBASE.SRC members

Member name	Contains
DFHCSD99	Sample CICS resource definition for defining tableBASE/ CICS journal for CICS Transaction Server for z/OS. Present if CICS interface is supplied.
DFHFCTBD	Entries for CICS File Control Table for BDAM libraries. Present if CICS interface is supplied.
DFHFCTOL	Entries for CICS File Control Table for BDAM libraries. Present if tablesONLINE/CICS interface is supplied.
DFHPLT6S	Entry for CICS Program Load Table for Termination. Present if CICS interface is supplied.
DFHPLT6T	Entry for CICS Program Load Table for Initialization. Present if CICS interface is supplied.
DK1EXDFN	Macro used to define a set of user exits.
DK1T1134	Default parameters for tableBASE to be run in batch.
DK1T1334	Default parameters for tableBASE to be run in IMS.
DK1T1434	Default parameters for tableBASE to be run in DB2.
DK1T2734	Default parameters for tableBASE/CICS. It is present if CICS interface is supplied.
DK1TX072	User exits table definition.

Table E-1: TBASE.SRC members (Continued)

Member name	Contains
DK1V1134	Default parameters for the tableBASE VTS Agent to run in a VTS environment. It is present if the VTS Agent is supplied.
DK1XUAFT	Sample user exit for each command (after execution).
DK1XUFOR	Sample user exit for each command (before execution).
DK1XUTIN	Sample user exit for CICS thread initialization.
DK1XUINT	Sample user exit invoked during region initialization.
DK1XUTER	Sample user exit at invoked during region termination.
DK1XUTTR	Sample user exit for CICS thread termination.
DKH	"C" data structures for tableBASE.
DKLUCTRN	Sample CICS resource definition table entries to enable mixed-case data entry on some tablesONLINE/CICS screens. It is present if tablesONLINE/CICS interface is supplied.
EXAMFLDS	Exit program for the example table to illustrate exit programming. this is a field level exit (or hook) program.
EXAMITMS	Exit program for the example tables to illustrate exit programming.
EXAMTBLS	Sample table exit program for table = example.
EXITISPF	This module maps the linkage section to the exits.
EXITPARM	COBOL copy book mapping the parameters passed to tablesONLINE, if tablesONLINE/CICS is supplied.
EXITWS	tablesONLINE working storage. Present if tablesONLINE/CICS is supplied.
TBAUDIT	Exit program for all on-line tables to track all update transactions that take place in tablesONLINE.
EXITPGMC	Exit program for the example table to illustrate exit programming.
TBASE60	CICS resource definition table entries for tableBASE/CICS. It is present if CICS interface is supplied.
TBASE60V	Entries for CICS File Control Table for VSAM libraries. It is present if CICS interface is supplied.

Table E-1: TBASE.SRC members (Continued)

Member name	Contains
TBDKUSID	This module is the source for the security exit that is installed with tablesONLINE/CICS. Refer to <i>tableBASE Administration Guide</i> for details. Present if tablesONLINE/CICS is supplied.
TBOL60	CICS resource definition table entries for tablesONLINE/CICS. Present if tablesONLINE/CICS interface is supplied.
TBOL60V	Entries for CICS File Control Table for VSAM libraries. Present if tablesONLINE/CICS interface is supplied.
TBOPTGEN	Macro used to specify installation defaults for interfaces.
TBPARMS	Sample COBOL copybooks for tableBASE parameters.

Note: Your system may have only some of the modules in the list.

Appendix F

JCL members

After the media unload, the tableBASE JCL Modules reside in the dataset:

`your.prefix.CNTL`

Table F-1 lists the members of this dataset.

Table F-1: JCL members

Member name	Contains
@@ALLOC	JCL to allocate tableBASE datasets from a distribution CD.
TC@LIST	JCL to list all dataset on the CD.
@@RECV	JCL to receive files back to mainframe into data sets.
ALT2734	To assemble and link DK1T2734 if it is customized. Present if the CICS interface is supplied.
ALT1134	To assemble and link DK1T1134 if it is customized.
ALT1334	To assemble and link DK1T1334 if it is customized.
ALT1434	To assemble and link DK1T1434 if it is customized.
ALTUSREX	To assemble and relink a user exit. The sample used is for DK1XUFOR).
ALV1134	To assemble and link DK1V1134. It is present if the VTS Agent is supplied.
ALTX072	To assemble and relink DL1T072.
AUTHLIB	Copy modules from TBASE.LOAD to an authorized library.
AUTHLIBV	Copy modules from TBASE.LOAD to an authorized library. Present if the VTS Agent is supplied.

Table F-1: JCL members (Continued)

Member name	Contains
BDMTOVSM	Sample JCL to convert a BDAM library to a VSAM library.
CICSJCL	JCL that is to be added to the CICS startup JCL. Present if the CICS interface is supplied.
CICSTBOL	JCL that is to be added to the CICS startup. Present only if tBOL/CICS is supplied.
CNVBDIAG (BDAM)	Sample JCL to run DK1TCNV to diagnose a Version 5, Bridge, Version 6 Transition, or Version 6 library.
CNVBFIX (BDAM)	Sample JCL to run DK1TCNV to fix a Version 5 library.
CNVBTO5 (BDAM)	Sample JCL to run DK1TCNV to convert a Bridge or Version 6 Transition library back to the Version 5 library format.
CNVBTO5B (BDAM)	Sample JCL to run DK1TCNV to convert a Version 5 or Version 6 Transition library to the Bridge library format.
CNVBTO6 (BDAM)	Sample JCL to run DK1TCNV to convert a Version 5, Version 6 Transition, or Bridge library to the Version 6 library format.
CNVBTO6T (BDAM)	Sample JCL to run DK1TCNV to convert a Version 5 or Bridge library to the Version 6 Transition library format.
CNVVTO6 (VSAM)	Sample JCL to run DK1TCNV to convert a Version 5, Version 6 Transition or Bridge library to the Version 6 library format.
CNVVTO6T (VSAM)	Sample JCL to run DK1TCNV to convert a Version 5 or Bridge library to the Version 6 Transition library format.
COPYAUTH	Sample JCL to COPY VTS Manager modules to AUTHLIB. Used only if the VTS Manager product is licensed.
DEFTBOPT	JCL to define and load TBOPT VSAM file. Present if CICS interface is supplied.
DFHSRTTB	JCL to assemble a CICS system recovery table.
DK1TCOBF	Sample JCL to run the batch utility DK1TCOBF (TBCOBF) that builds a copy book for a View.
DK1TDRV	Sample JCL to run the batch driver program DK1TDRV (TBDRIVER).

Table F-1: JCL members (Continued)

Member name	Contains
DK1TEXEC	Sample JCL to run the batch utility DK1TEXEC (TBEXEC).
DK1TLCHK	Sample JCL to execute a tableBASE library identification program.
DK1TPRTL	Sample JCL to run the DK1TPRTL(TBPRNTL) program to dump a tableBASE library to assist in problem diagnosis with a damaged library.
DK1TPRTM	Sample JCL to run the DK1TPRTM(TBPRNTM) program to dump a tableBASE library to assist in problem diagnosis with a damaged library.
DK1TPTBL	Sample JCL to run the program DK1TPTBL(TBPRINT), which prints formatted listings of tables using their corresponding tablesONLINE Views.
DK1TSTRB	Extracts the information from the tableBASE/CICS journal file and produces the Table Space Report.
DK1TVWPR	Sample JCL to run DK1TVWPR(TBDEFPR), a batch utility that prints Views.
PCSRVR	JCL to execute the tableBASE PC server.
RELINK	Sample job to relink application load modules to replace pre-V6 stubs with the V6 API stub.
RLK31ANY	Sample JCL to relink DK1TCALL and all its aliases (AMODE31, RMODE ANY), and replaces the load. This allows 31-bit TBLBASE operation.
TBCOMP	Sample JCL to run DK1TCOMP(TBCOMP), a batch utility to compare tables.
TBOL600	JCL to make a copy of TBACTLB, TBAPPLB and TBDICLB before customization. Present if tablesONLINE/CICS is supplied.
TBOL600V	For tableBASE VSAM libraries – JCL to make a copy of TBACTLB, TBAPPLB and TBDICLB before customization. Present if tablesONLINE/CICS is supplied.
TBOL601	JCL to upgrade TBOLACT table to Release 5.1 standards. Present if tablesONLINE/CICS is supplied.
TBOL601V	For tableBASE VSAM libraries – JCL to upgrade TBOLACT table to Release 5.1 standards. Present if tablesONLINE/CICS is supplied.

Table F-1: JCL members (Continued)

Member name	Contains
TBOL602	JCL to make a copy of TBDICLB before customization. Present if tablesONLINE/CICS is supplied.
TBOL602V	For tableBASE VSAM libraries – JCL to make a copy of TBDICLB before customization. Present if tablesONLINE/CICS is supplied.
TBOL603	JCL to upgrade TBOLPROF to Release 5.0 standards and to copy TBOLJCL1 and TBOLHELP from Release 5.0 libraries. Present if tablesONLINE/CICS is supplied.
TBOL604	JCL to upgrade TBOLPROF from Release 5.0 to Release 5.1 standards and copy TBOLJCL1, TBOLJCL2, TBOLJCL3 and TBOLHELP from Release 5.1 libraries. Present if tablesONLINE/CICS is supplied.
TBOL604V	For tableBASE VSAM libraries – JCL to upgrade TBOLPROF from Release 5.0 to Release 5.1 standards and copy TBOLJCL1, TBOLJCL2, TBOLJCL3 and TBOLHELP from Release 5.1 libraries. Present if tablesONLINE/CICS is supplied.
TBOLMRO	JCL to set up tablesONLINE in a CICS/MRO environment. Present if tablesONLINE/CICS is supplied.
TBVIEWCV	JCL to upgrade tablesONLINE/CICS Views to release 5.0 standards. Present if tablesONLINE/CICS is supplied.
UPDTDESC	JCL to upgrade XXXDESC tables to Release 5.0 standards. Present if tablesONLINE/CICS is supplied.
UPDTHELP	JCL to upgrade XXXXHELP tables to Release 5.0 standards. Present if tablesONLINE/CICS is supplied.
UPDTMENU	JCL to upgrade XXXXMENU tables to Release 5.0 standards. Present if tablesONLINE/CICS is supplied.
UPDTMSGS	JCL to upgrade XXXXMSGS tables to Release 5.0 standards. Present if tablesONLINE/CICS is supplied.
VALIDATE	JCL to verify that the installation of the MVS batch interface of tableBASE is complete and correct. The JCL illustrates the use of the utility program TBEXEC and the tableBASE batch command executor TBDRIVER. If the default parameters were modified, the verification jobs will also require modification.

Table F-1: JCL members (Continued)

Member name	Contains
VALIDVTS	JCL to verify that the installation of the VTS Agent of tableBASE is complete and correct. The JCL parallels the VALIDATE member, but executes against a VTS-TSR.
VTSAGENT	Sample JCL to start a VTS Agent. Present if VTS is supplied.

Note: Your system may have only some of the modules in the list.

Appendix G

Distribution datasets and libraries

This appendix lists the distribution datasets and libraries.

Mandatory tableBASE datasets

The installation media contains only the products that have been licensed. There are seven partitioned datasets that are always present on the distribution media, which are described below.

your.prefix.CNTL

This data set contains two types of members:

- JCL to install, customize, and test tableBASE and all the products that have been licensed.
- JCL to run tableBASE utility programs.

Both types of members may require customization to conform to your particular installation standards.

your.prefix.LOAD

This data set contains:

- all executable programs for tableBASE and other licensed DataKinetics products; and
- the tableBASE API, TBLBASE (an alias of DK1TCALL). This entry point is required to link-edit programs that call tableBASE.

your.prefix.SRC

This data set contains source code for programs that are used to customize the various installation parameters for use with tableBASE and other licensed DataKinetics products. Test programs are also included.

This data set also contains:

- COBOL copybooks for tableBASE parameter areas in your COBOL programs.
- Sample coding examples utilizing the tablesONLINE exit feature.
- Examples of batch programs utilizing various facilities of tableBASE.
- Material for use in tableBASE tutorials.

your.prefix.CLIST

This data set contains TSO CLISTs required by tablesONLINE/ISPF as well as CLISTs that enable you to run tableBASE utility programs in the TSO foreground.

Optional tableBASE datasets

DataKinetics Ltd. tailors the distribution media to include only the products licensed for your installation. Depending on the licensing arrangement, the distribution media may contain one or more of the following partitioned datasets.

your.prefix.MESSAGES

This data set contains messages used by tablesONLINE/ISPF.

your.prefix.PANELS

This data set contains the screens used by tablesONLINE/ISPF.

your.prefix.SKELETON

This data set contains a skeleton library used by tablesONLINE/ISPF. For details about the use of this library, see the *tablesONLINE/ISPF User's Guide*.

your.prefix.TABLE

This data set contains tables used by tablesONLINE/ISPF in its internal operation.

Mandatory tableBASE libraries

The installation media contains one or more of the following tableBASE libraries.

your.prefix.MAINLIB

This library contains Example tables used by various tableBASE components and products.

your.prefix.TBSYSLB

This library contains System tables used by various tableBASE components and products.

Note: The contents of this library must not be altered.

Optional tableBASE libraries

DataKinetics Ltd. tailors the distribution media to include only the products licensed for your installation. Depending on the licensing of tablesONLINE/CICS, the distribution media may also contain the following libraries.

your.prefix.TBAPPLB

This library is for use with tablesONLINE/CICS. It contains the starter set of Application Driving Tables for tablesONLINE/CICS. For information about modifying these tables, see the *tablesONLINE/CICS User's Guide*.

your.prefix.TBACTLB

This library contains tables that control security for tablesONLINE/CICS. These tables are typically maintained by the tableBASE administrator.

your.prefix.TBDICLB

This library is used by tablesONLINE/CICS. It contains Views that have corporate-wide applicability. Use this library to centralize those Views that are used by many different applications.

your.prefix.UPGRDLB

This library is used for tablesONLINE/CICS.

