

**HP Computer Systems
Training Course**

TurboIMAGE/DBMS 3000 Instructor Workbook

Course No. 35053B



For Internal Use Only



**APPLICATION SUPPORT DIVISION
19320 Pruneridge Ave., Cupertino, CA 95014**

35053-90012 Printed in U.S.A. 10/85
Copyright (c) 1985 Hewlett-Packard Company

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD PROVIDES THIS MATERIAL "AS IS" AND MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS) IN CONNECTION WITH THE FURNISHING, PERFORMANCE OR USE OF THIS MATERIAL WHETHER BASED ON WARRANTY, CONTRACT, OR OTHER LEGAL THEORY.

Some states do not allow the exclusion of implied warranties or the limitation or exclusion of liability for incidental or consequential damages, so the above limitation and exclusions may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

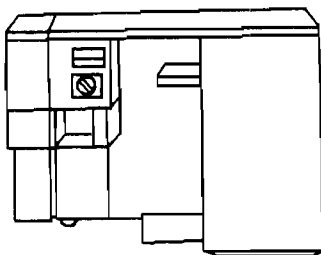
Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

This document contains proprietary information which is protected by copyright. All rights are reserved. No part of this document may be photocopied, reproduced or translated to another program language without the prior written consent of Hewlett-Packard Company.

Copyright (c) 1985 by HEWLETT-PACKARD COMPANY

Preface

LASER PRINTED WORKBOOK



This workbook was printed on an HP 2680 Laser Printer. The graphics were prepared on an HP graphics terminal using HPDRAW and were interfaced to the laser printer by TDP/3000, a Hewlett-Packard text processor.

Copyright © 1985
The information in this document is subject to change without notice. Hewlett-Packard is not responsible for errors or for any consequences arising from the use of the information contained herein. The appearance of trademarks in this document does not constitute an endorsement of those products.

Lab Setup Instructions

This lab setup document provides instructions for restoring the Lab Store tape for the TurboIMAGE/DBMS 3000 course.

1. Building the Account

Create an account IMAGE with the following capabilities: AM,AL,GL,ND,SF, IA,BA,PH,DS, and with user MGR having AM,GL,ND,SF,IA,BA,PH,DS capabilities. Create a LABS group (with 24,000 sectors) and a SOLUTION group (with 14,000 sectors).

The LABS and SOLUTION groups should have IA,BA,PH,DS capabilities. Access should be (R,L,X:ANY;W,A,S:GU,AL).

2. Restoring the Files

Restore all the files from the Lab Store tape as follows:

```
:HELLO MGR.SYS
:FILE T;DEV=TAPE;DEN=1600
:RESTORE *T;@.LABS.IMAGE,@.SOLUTION.IMAGE;SHOW
```

Files will be restored to the IMAGE account.

3. Installing the Labs

UDCs have been set up in ACCUDC.LABS to initiate the ITF (Interactive Training Facility) activities. These UDCs issue the following commands: :FILE COURSEIN= filename.LABS; :RUN CME.LABS. Set and release ACCUDC.LABS at the account level. Each of the ITF activities sends a printout to LP at the end of the activity. The formal file designator for this output is TRAINOUT. The file ITFDOC.LABS provides a description of the files relating to each ITF activity.

Create and load the REALTY data base in the LABS group in preparation for the lab activity in Module 2. Run DBSCHEMA, using LAB2SCHM.LABS for the DBSTEXT file; run DBUTIL,CREATE; run LOADDB.LABS to load data into the data base; and release REALTY.LABS.

Materials List

Instructor Materials:

Instructor Kit35053-60001
 (includes the following)

- Instructor Workbook
- Lab Store Tape
- Binder
- Spine Insert
- Tab Dividers
- MPE Software Pocket Guide

Overhead Slide Kit35009-60003
 (includes the following)

- Overhead Slide Set
- Binder
- Spine Insert

Student Materials:

Student Kit35053-60004
 (includes the following)

- Student Workbook
- Binder
- Spine Insert
- Tab Dividers
- MPE Software Pocket Guide

This lab series
 Laboratory
 1. Binder
 Create
 MPE Software
 SOLUTIONS
 THE LAB
 (BLACK AND WHITE)
 2. Reader
 Restore all the
 HELLO, I'M
 FILE FOLDER
 PRESENTATION
 File
 3. Reader
 GOOD
 The
 and
 the
 provided
 Good
 2
 2000

Table of Contents

MODULE 1 Data Management	1-1
Activity 1-1 Chalk Talk: Wonder Realty	1-10
Activity 1-2 Worksession: Data Base Models	1-11
Activity 1-3 Quiz: Basic Concepts	1-14
MODULE 2 TurboIMAGE Structure and Terminology	2-1
Activity 2-1 Chalk Talk: Master Data Sets	2-4
Activity 2-2 Lab: Introduction to the REALTY Data Base	2-9
Activity 2-3 Lab: Using QUERY/3000	2-18
MODULE 3 Review of Data Base Concepts	3-1
Activity 3-1 Quiz: Basic Data Base Concepts	3-2
Activity 3-2 Worksession: Data Base Design-Part One	3-5
MODULE 4 Data Definition Language	4-1
Activity 4-1 Lab: Data Base Creation-Part One	4-11
Activity 4-2 Lab: Data Base Creation-Part Two	4-18
Activity 4-3 Lab: Data Base Creation-Part Three	4-21
MODULE 5 Data Base Access	5-1
Activity 5-1 Worksession: Data Set Access	5-5
Activity 5-2a Lab: Accessing a Data Base	5-18
Activity 5-2b Lab: Accessing a Data Base with RPG	5-19
Activity 5-3a Lab: Data Retrieval	5-23
Activity 5-3b Lab: Data Retrieval with RPG	5-24
Activity 5-4a Lab: Data Modification	5-26
Activity 5-4b Lab: Data Modification with RPG	5-27
MODULE 6 Multiple User Considerations and Security	6-1
Activity 6-1 Chalk Talk: Concurrent Access Modes	6-4
Activity 6-2 Demo: Locking	6-6
Activity 6-3 Chalk Talk: Locking Strategies	6-11
Activity 6-4a Lab: Multiple User Access	6-13
Activity 6-4b Lab: Multiple User Access with RPG	6-14
Activity 6-5 Chalk Talk: Set and Item Access	6-17
Activity 6-6 Worksession: Security	6-18
Activity 6-7 Lab: Adding Security Features	6-19
MODULE 7 Review of Data Base Applications	7-1
Activity 7-1a Lab: Writing a Data Base Application	7-3
Activity 7-1b Lab: Optional Exercise	7-5

Table of Contents (cont'd)

MODULE 8 Utilities and Transaction Logging	8-1	
Activity 8-1 Chalk Talk:	DBUTIL Utility Program	8-3
Activity 8-2 Demo:	Utility Simulations	8-9
Activity 8-3 Chalk Talk:	Sequence of Operations	8-10
Activity 8-4 Worksession:	Data Base Restructuring	8-12
Activity 8-5 Demo:	Implementing Logging	8-16
Activity 8-6 Chalk Talk:	Logging Media	8-17
Activity 8-7 Chalk Talk:	DBRECOV >CONTROL Command	8-19
Activity 8-8 Demo:	Implementing Data Base Recovery	8-22
Activity 8-9 Worksession:	Logging and Recovery	8-23
MODULE 9 Performance and Design	9-1	
Activity 9-1 Chalk Talk:	Overview of DBMS Implementation	9-2
Activity 9-2 Chalk Talk:	Design Trade-offs	9-5
Activity 9-3 Chalk Talk:	Top-Down Design Approach	9-7
Activity 9-4 Chalk Talk:	Common Design Decisions	9-9
Activity 9-5 Chalk Talk:	Performance	9-13
Activity 9-6 Worksession:	Performance Considerations	9-16
Activity 9-7 Worksession:	Data Base Design-Parts Two & Three	9-17
MODULE 10 Data Base Tools	10-1	
APPENDIX A - Solutions	A-1	
APPENDIX B - Additional Information	B-1	
Schema Structure	B-1	
DBUTIL.PUB.SYS	B-2	
TurboIMAGE Utilities	B-4	
Lock Descriptor Examples	B-5	
APPENDIX C - Reference Material	C-1	
APPENDIX D - BASIC Applications	D-1	
APPENDIX E - Internal Structures of IMAGE/3000 and TurboIMAGE	E-1	
APPENDIX F - RPG & Locking	F-1	
APPENDIX G - Compile, Prep, and Execute Commands	G-1	

Overview

Course Description

The TurboIMAGE/DBMS 3000 customer training course was developed to provide data base programmers/designers/administrators with basic data base concepts in general and TurboIMAGE concepts in particular. The students will be able to create, access, and maintain a data base using TurboIMAGE procedures and utilities.

Course Goal & Student Learning Objectives

To present the components of the TurboIMAGE data base management system for data base programmers and data base administrators.

Upon completion of this course, the student will be able to:

- contrast IMAGE/3000 and TurboIMAGE.
- define data base and data base management system.
- identify three basic data base models.
- define TurboIMAGE data structures.
- identify the four components of TurboIMAGE.
- use QUERY/3000 to access a data base.
- utilize DBSCHEMA to create a root file.
- utilize DBUTIL to create, and maintain a data base.
- identify four data base access methods.
- utilize TurboIMAGE procedures in an application program.
- describe TurboIMAGE locking types and strategies.
- implement TurboIMAGE security via user class numbers.
- state the function of the TurboIMAGE utilities.
- identify the components of logging implementation.
- utilize recovery procedures.
- design a data base using performance and design guidelines.

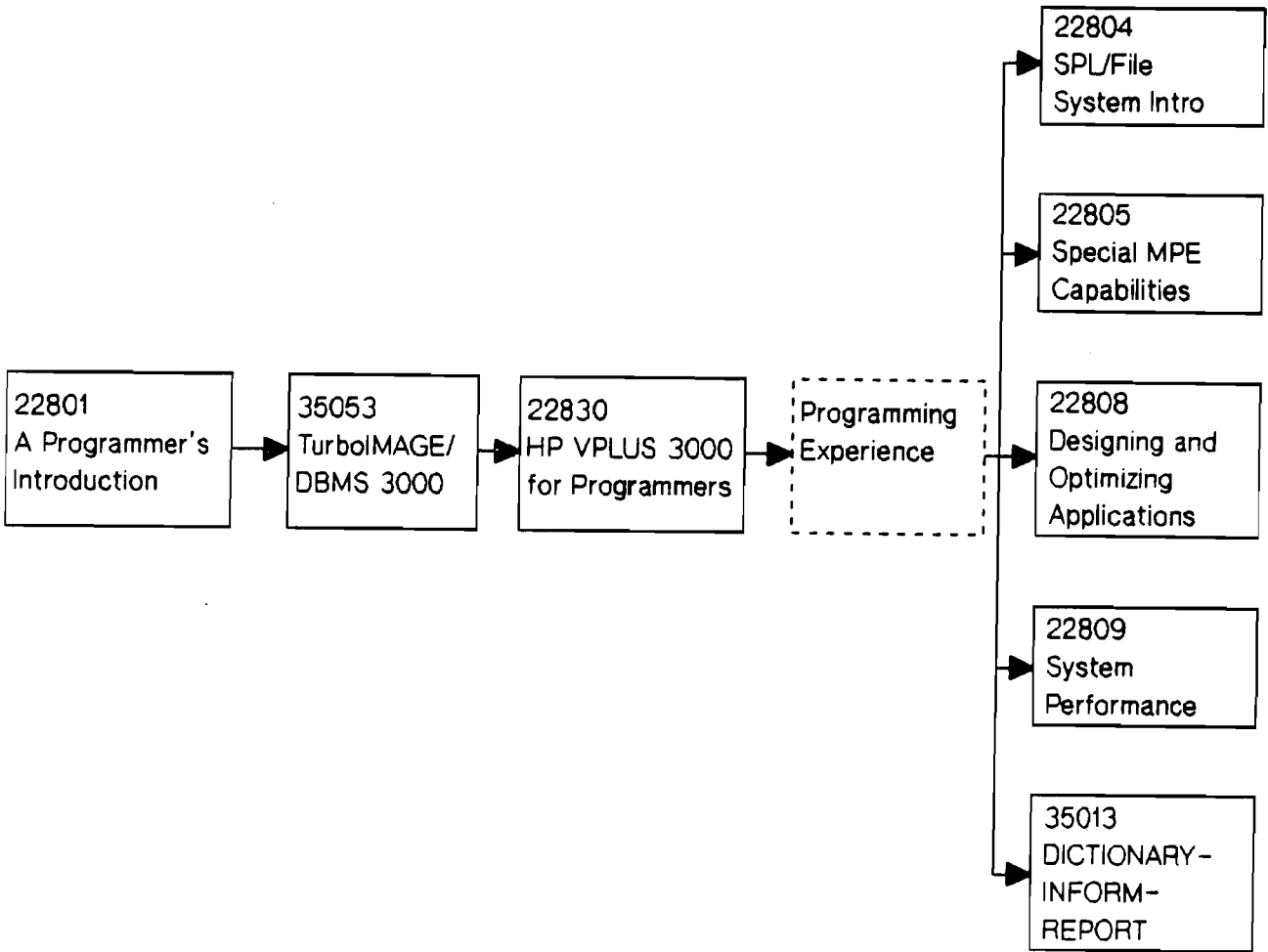
Overview (cont'd)

Reference Documentation:

- TurboIMAGE Data Base Management System Reference Manual (Part No. 32215-90050)
- QUERY Reference Manual (Part No. 30000-90042)
- Dictionary/3000 Reference Manual (Part No. 32244-90001)
- Report/3000 User's Guide (Part No. 32245-90001)
- Inform/3000 User's Guide (Part No. 32246-90001)
- Transact/3000 Reference Manual (Part No. 32247-90001)
- TurboIMAGE Profiler User's Guide (Part No. 36914-91001)
- BASIC/3000 Interpreter Manual (Part No. 30000-90026)
- RPG/3000 Reference Manual (Part No. 32104-90001)
- SPL Reference Manual (Part No. 30000-90024)
- FORTRAN Reference Manual (Part No. 30000-90040)
- COBOLII/3000 Reference Manual (Part No. 32233-90001)
- Pascal/3000 Reference Manual (Part No. 32106-90001)

Overview (cont'd)

Curriculum



Overview (cont'd)

Course Outline

O. Overview

- A. Course description
- B. Course goal and student learning objectives
- C. Curriculum
- D. Reference documentation
- E. Course outline
- F. Training agenda

I. Data Management

- A. Key features of TurboIMAGE
- B. HP 3000 environment for TurboIMAGE
- C. Definition of a data base management system
- D. Role of the data base administrator
- E. Data base models
 - 1. Hierarchical
 - 2. Relational
 - 3. Network
 - 4. TurboIMAGE
- F. File access

II. TurboIMAGE Structure and Terminology

- A. TurboIMAGE data structures
- B. TurboIMAGE terminology
- C. Media records
- D. Case study structure
- E. Components of TurboIMAGE
 - 1. Data definition language
 - 2. Data manipulation language
 - 3. Utilities
 - 4. QUERY/3000
 - a. HELP
 - b. Special characters
 - c. DEFINE
 - d. FORM
 - e. FIND
 - f. REPORT
 - g. LIST
 - h. ADD
 - i. REPLACE
 - j. DELETE
 - k. OUTPUT

III. Review of Data Base Concepts

- A. Review of the four components of TurboIMAGE
- B. Implementation of a data base management system

IV. Data Definition Language

- A. Data base creation process
- B. Schema components
 - 1. Data base name
 - 2. Password
 - 3. Item part
 - 4. Set part
 - 5. End part
- C. Data base limitations
- D. Schema processor

Overview (cont'd)

Course Outline (cont.)

IV. Data Definition Language (cont.)

- E. DBUTIL
 1. HELP
 2. EXIT
 3. CREATE
 4. ERASE
 5. PURGE

V. Data Base Access

- A. Data base access methods
 1. Serial
 2. Direct
 3. Calculated
 - a. synonyms
 - b. migrating secondaries
 4. Chained
- B. TurboIMAGE procedures
 1. Parameters
 2. Data types
- C. Procedure definitions and syntax
 1. DBOPEN
 2. DBINFO
 3. DBERROR
 4. DBEXPLAIN
 5. DBPUT
 6. DBCLOSE
 7. DBFIND
 8. DBGET
 9. DBUPDATE
 10. DBDELETE

VI. Multiple User Considerations and Security

- A. Concurrent access
- B. DBOPEN modes
- C. Controlling access
 1. DBLOCK
 2. DBUNLOCK
 3. Locking strategies
- D. Data base security

VII. Review of Data Base Applications

- A. Components of a data base application
- B. Writing a data base application

VIII. Utilities and Transaction Logging

- A. DBUTIL
- B. DBSTORE
- C. DBRESTOR
- D. DBUNLOAD
- E. DBLOAD
- F. DBCONV
- G. Data base restructuring
- H. Logging
- I. DBRECOV
- J. Recovery
 1. Roll-forward
 2. Roll-back
 3. ILR

Notes to the Instructor

Orientation/Philosophy

The TurboIMAGE/DBMS 3000 course provides the data base programmer/administrator with the necessary training to implement an IMAGE/3000 or TurboIMAGE data base management system. The overall course and each module have specific and measurable objectives. The course promotes interactive participation by the students and the instructor. There are labs, chalk talks, demonstrations, worksessions, and quizzes to reinforce the information being presented to the student.

Audience

This course is intended for data base programmers, data base designers, or data base administrators who are looking for the basic information necessary to design, create, access, and maintain an IMAGE/3000 or TurboIMAGE data base.

Instructor Profile

The instructor should have programming experience utilizing TurboIMAGE procedures and knowledge of data base design concepts. The instructor should also have reading knowledge of COBOL, FORTRAN, RPG, BASIC, SPL, and Pascal in order to assist the students in the lab activities.

How To Prepare & Teach

This Instructor Workbook will help you present the information in the course; it will not tell you everything to say. Read the information on the following pages and use it to help you be a more effective TurboIMAGE instructor.

1. Take time to familiarize yourself with the course content and organization. Note the following:
 - a. The course material is based on MPE V/E. The course covers TurboIMAGE as well as IMAGE/3000. Any topics that are exclusively TurboIMAGE are shaded in both the student workbook and the instructor workbook.
 - b. The text is organized by PURPOSE, KEY POINTS, PREPARATION, and TEACHING TIPS.
 - c. The PURPOSE and KEY POINTS are on the page adjacent to each slide. The TEACHING TIPS are also included on this page if room permits. Make sure that you cover each KEY POINT in as much detail as necessary for students to understand the point being made.
 - d. An overview is provided at the beginning of each module. This overview contains a list of the slides and activities in the module, grouped into lessons. An approximate time is given for each lesson. The time listed includes the suggested times for the activities in the lesson, and five minutes for each slide or text page in the lesson. Use this overview as a guide while teaching the module.
 - e. Many of the slides require the students to take notes as the slide is being discussed. Write key points on the overhead slide directly or use the chalkboard or a flip chart to highlight the main concepts of the slide.
 - f. The PREPARATION and remaining TEACHING TIPS are at the end of each module. Review and study this information before the class. Use this material to help prepare your lectures; be sure to bring out KEY POINTS during your preparations.
 - g. Add any activities that you feel will help enhance the existing material.
2. Review each slide individually. Read the PREPARATION materials with an eye to how you can enhance it with your own experiences. Remember, your own experience provides the best lecture material; use it to give lots of day-to-day, on-the-job, real-life situations that help reinforce the KEY POINTS.

Notes to the Instructor (cont'd)

3. Study each TEACHING TIP. Decide how you can use it to help students better understand the material. These tips may trigger ways for you to present/explain information or promote class discussion. Develop other teaching tips for yourself.
4. Do each LAB, WORKSESSION, CHALK TALK, DEMONSTRATION, and QUIZ. Try to anticipate questions and problems that students might have. The estimated activity completion time is provided on the instructor page for each activity. Use this as a guide only. Depending on the experience level of individual classes, you may want to adjust the amount of time spent on a given activity.
5. Make sure that you have all necessary equipment and materials available prior to the beginning of the class. Check that everything is in working order. Have one copy of each language reference manual (BASIC, SPL, COBOL, FORTRAN, Pascal, RPG) in the classroom for use during the programming labs. One TurboIMAGE reference manual should be provided for every two students. Encourage students to use the reference manuals whenever possible.
6. Have a supply of blank transparencies and pens available for students to use for their group presentations in Module 9. Prepare your own solution to this design lab on a transparency and present it to the class after the group preparations. Make copies of all the solutions and hand them out to the students.
7. Create and load the REALTY data base in the LABS group in preparation for the lab activity in Module 2. Run DBSCHEMA, using LAB2SCHM.LABS for the DBSTEXT file; run DBUTIL, CREATE; run LOADDB.LABS to load data into the data base; and release REALTY.LABS.
8. Create student logons prior to class. Home users to groups called: GROUP1, GROUP2, GROUP3, etc. The LABS group of the account must have PH and DS capabilities to support the ITF (Interactive Training Facility) activities.
9. Use blank flip chart sheets to write out the student objectives for each module in the course. Display the module objectives in class as you begin to teach each module. Check off each objective when it is met during the course. This will highlight the progress being made in class. Prepare these flip charts prior to the first day of class. An alternate method is to make an overhead transparency of the first page of each module. Then you can highlight the objectives before beginning the module, and review the objectives at the end of the module.
10. Slides 3.02 and 8.17 depict overviews of data base implementation and TurboIMAGE respectively. If available, use an HP2685 or HP2686 plotter to make poster size slides. Display these in the classroom at the appropriate times. These will help the students focus on the "big picture."
11. To make the lab activities more effective, assign students to lab teams according to the programming language to be used and the amount of programming experience. Be especially aware of any data base administrators in the class; assign them to work with more experienced programmers. It may be helpful to print several copies of the labs and solutions prior to class. Many students will want hard copies anyway, and you can save valuable class time by doing this ahead of time. To increase the students exposure to ideas, break up lab partners as much as possible when you assign students to groups for the design lab in Modules 3 and 9.
12. UDCs have been set up in ACCUDC.LABS to initiate the ITF (Interactive Training Facility) activities. These UDCs issue the following commands: :FILE COURSEIN=filename.LABS, :RUN CME.LABS. Be sure to set and release ACCUDC.LABS at the account level before the class begins. The file ITFDOC.LABS provides a description of the files relating to each ITF activity.
13. The most important aspect of presenting this course material is the interaction between students and instructor. Encourage students to participate by asking them to take notes, ask questions, and answer questions. Try to tailor your preparation to the individual concerns and interests of the class.

Notes to the Instructor (cont'd)

Overview of Activities

The following is a brief description of the types of activities in this course:

Chalk Talk - A chalk talk is designed to promote discussion between the instructor and the students.

Worksession - A worksession can be either a paper and pencil activity or an on-line, interactive activity. Worksessions provide an opportunity for students to implement skills introduced in the course.

Lab - A lab requires students to work directly on the computer, either entering code into a program, or running an existing program.

Demonstration - A demonstration is designed to illustrate a concept to the students.

Quiz - A quiz is designed to test the students' understanding of basic concepts.

ITF(Interactive Training Facility) - These activities are on-line interactive quizzes, demonstrations, and worksessions. Refer to the file ITFDOC.LABS for an explanation of all the files involved in these activities. Each activity is initiated by a UDC in ACCUDC.LABS. A copy of each activity is sent to device class LP at the end of each session. The formal file designator for this output is TRAINOUT.

The TurboIMAGE course contains a wide variety of activities. Part of your preparation for teaching this course should be completion of all the activities in the course. Since customer backgrounds and needs vary from class to class, it may be necessary to adjust the emphasis placed on individual activities.

If a class consists of a majority of programmers, the programming labs should be emphasized, and adequate time provided for completion of these labs. If there are many data base administrators in the class, the emphasis in the programming labs should be on what the TurboIMAGE procedures do, and what error conditions can result from the procedures, not on compiling and running the programs.

If time is a factor in completing all the activities, do not use class time for the paper and pencil activities. Suggest that students complete these activities (such as Activity 5-1) at home, or at a time during the class when they complete another activity early.

The most important point in preparing for the class is to plan, AHEAD OF TIME, how much emphasis you want to place on individual activities, and how much class time you will allow for completion of each activity. The times suggested in the instructor workbook allow for completion of all the material within the five days, allowing one hour for lunch, and thirty minutes for breaks throughout each day.

Notes to the Instructor (cont'd)

Activity List & Files

Chalk Talk 1-1	Wonder Realty
Worksession 1-2	Data Base Models
Quiz 1-3	Basic Concepts (ACCUDC.LABS, CME.LABS, IO103QP.LABS, QUIZ1.SOLUTION)
Chalk Talk 2-1	Master Data Sets
Lab 2-2	Introduction to the REALTY Data Base (LABDEMO.LABS, LAB2SCHM.LABS, LOADDB.LABS, LAB4DTA1.LABS, LAB4DTA2.LABS, LAB4DTA3.LABS, LAB4DTA4.LABS)
Lab 2-3	Using QUERY/3000 (REALTY.LABS)
Quiz 3-1	Basic Data Base Concepts (ACCUDC.LABS, CME.LABS, IO301QP.LABS, MOD3.SOLUTION)
Worksession 3-2	Data Base Design-Part One
Lab 4-1	Data Base Creation-Part One (MINISCHM.groupn)
Lab 4-2	Data Base Creation-Part Two (MINISCHM.groupn, REALTY.groupn)
Lab 4-3	Data Base Creation-Part Three (REALTY.groupn, LAB4SCHM.LABS, LOADDB.LABS, LAB4DTA1.LABS, LAB4DTA2.LABS, LAB4DTA3.LABS, LAB4DTA4.LABS, LOADDBJ.LABS)
Worksession 5-1	Data Set Access
Lab 5-2a	Accessing a Data Base (L5xxx2S.LABS, S5xxx2S.SOLUTION xxx=BAS, COB, FTN, PAS, SPL)
Lab 5-2b	Accessing a Data Base with RPG (L5RPG2S.LABS, RPGFORMS.LABS, S5RPG2S.SOLUTION)
Lab 5-3a	Data Retrieval (L5xxx3S.LABS, S5xxx3S.SOLUTION, xxx=BAS, COB, FTN, PAS, SPL)
Lab 5-3b	Data Retrieval with RPG (L5RPG3S.LABS, S5RPG3S.SOLUTION)
Lab 5-4a	Data Modification (L5xxx4S.LABS, S5xxx4S.SOLUTION, xxx=BAS, COB, FTN, PAS, SPL)
Lab 5-4b	Data Modification with RPG (L5RPG4S.LABS, S5RPG4S.SOLUTION, L5XPG4S.LABS, S5XPG4S.SOLUTION)
Chalk Talk 6-1	Concurrent Access Modes
Demo 6-2	Locking (ACCUDC.LABS, IO602DP.LABS, LOCKMOD6.SOLUTION)
Chalk Talk 6-3	Locking Strategies

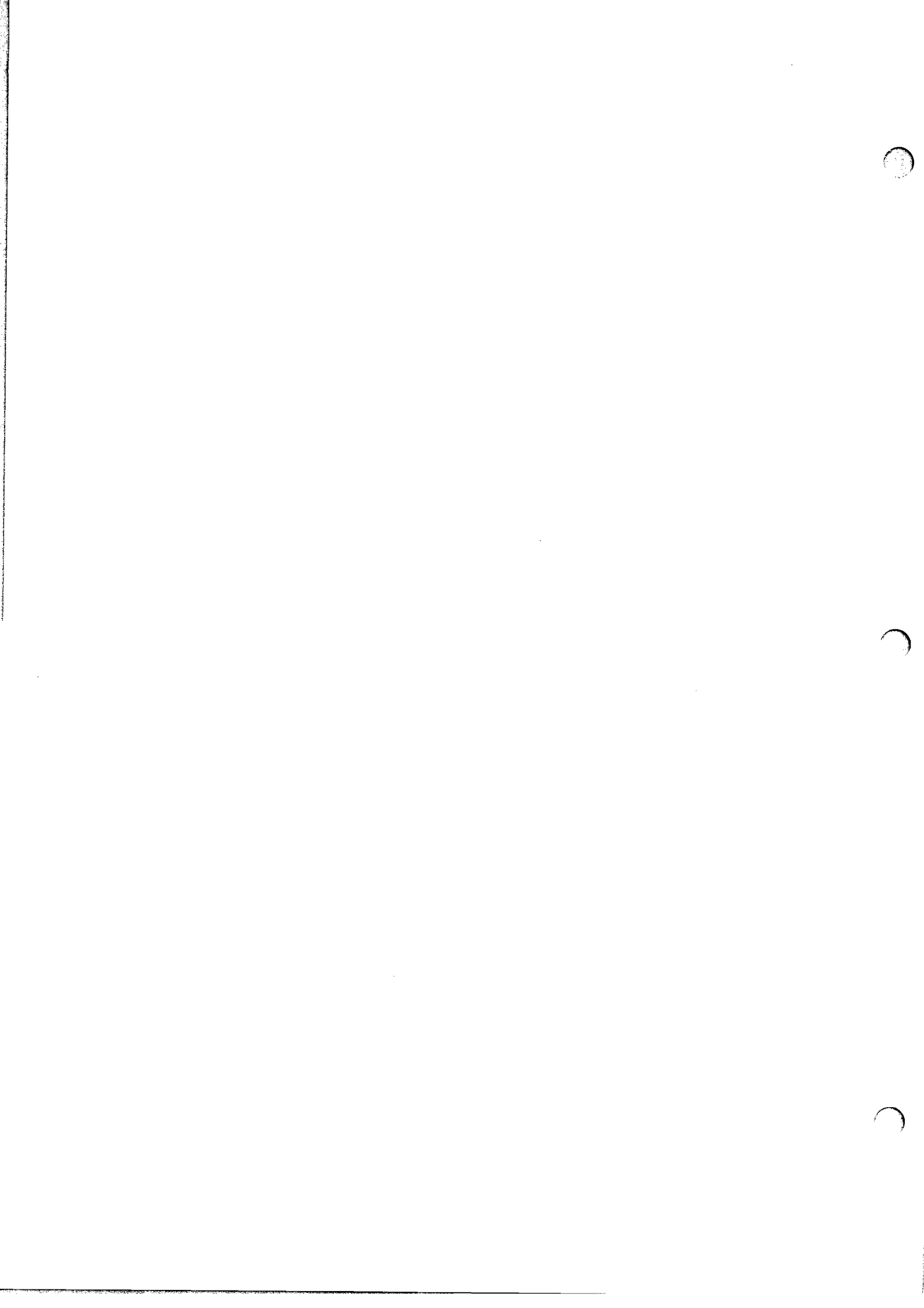
Notes to the Instructor (cont'd)

Activity List & Files (cont'd)

Lab 6-4a	Multiple User Access (L6xxx4S.LABS, S6xxx4S.SOLUTION, xxx=BAS, COB, FTN, PAS, SPL)
Lab 6-4b	Multiple User Access with RPG (L6RPG4S.LABS, S6RPG4S.SOLUTION)
Chalk Talk	Set and Item Access
Worksession 6-6	Security (ACCUDC.LABS; CME.LABS, I0606WP.LABS, SECMOD6.SOLUTION)
Lab 6-7	Adding Security Features (LAB4SCHM.groupn, LOADDDBJ.groupn, REALTY.groupn, S6xxx7S.SOLUTION)
Lab 7-1	Writing a Data Base Application (LEV1xxS.LABS, LEV2xxxS.LABS, LEV3xxxS.LABS, S7xxx1S.SOLUTION, xxx=BAS, COB, PAS, SPL)
Chalk Talk 8-1	DBUTIL Utility Program
Demo 8-2	Utility Simulations (ACCUDC.LABS, CME.LABS, I0802DP.LABS, MOD8LIST.SOLUTION)
Chalk Talk 8-3	Sequence of Operations
Worksession 8-4	Data Base Restructuring
Demo 8-5	Implementing Logging (ACCUDC.LABS, CME.LABS, I0805DP.LABS, MOD8LOG.SOLUTION)
Chalk Talk 8-6	Logging Media
Chalk Talk 8-7	DBRECOV >CONTROL Command
Demo 8-8	Implementing Data Base Recovery (ACCUDC.LABS, CME.LABS, I0808DP.LABS, MOD8REC.SOLUTION)
Worksession 8-9	Logging and Recovery
Chalk Talk 9-1	Overview of DBMS Implementation
Chalk Talk 9-2	Design Trade-offs
Chalk Talk 9-3	Top-Down Design Approach
Chalk Talk 9-4	Common Design Decisions
Chalk Talk 9-5	Performance
Quiz 9-6	Performance Considerations
Worksession 9-7	Data Base Design Parts Two & Three

Concluding Remarks

You play a key role in the success of TurboIMAGE. Hopefully, the Workbook and the Notes contained herein will help make your job easier and satisfying. It will hopefully help make your students perform their jobs more effectively and efficiently. All SEs who provided feedback during the development phase were extremely helpful. Any additional feedback is valuable and efforts to help improve this course are greatly appreciated; the INSTRUCTOR REVIEW SHEET is provided for this purpose. Good luck.



Module 1 Instructor Notes

□ Data Management

Overview of Module 1

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Introductions and Course Overview (30 minutes)

Lesson 1. Comparison of IMAGE/3000 and TurboIMAGE (15 minutes)

Slides: 1.01 - TurboIMAGE *
1.02 - HP 3000 Environment for TurboIMAGE
1.03 - How Does This Workbook Differentiate Between IMAGE/3000 and TurboIMAGE?

Lesson 2. Information Management (30 minutes)

Slides: 1.04 - What is Data Base? *
1.05 - Data Base Interaction *
1.06 - Before Data Base *
1.07 - After Data Base *
1.08 - Data Base Management System *
1.09 - Data Base Administrator

Lesson 3. Data Base Models (25 minutes)

Slides: 1.10 - Data Base Models *
1.11 - Hierarchical Model *
1.12 - Relational Model *
1.13 - Network Model *
1.14 - TurboIMAGE is a Network Structure *

Lesson 4. Applying Data Base Concepts to a Case Study (40 minutes)

Slide: 1.15 - Case Study *

Activity: Chalk Talk: Wonder Realty

Purpose: To discuss how data base concepts apply to Wonder Realty's objectives.

Activity: Worksession: Data Base Models

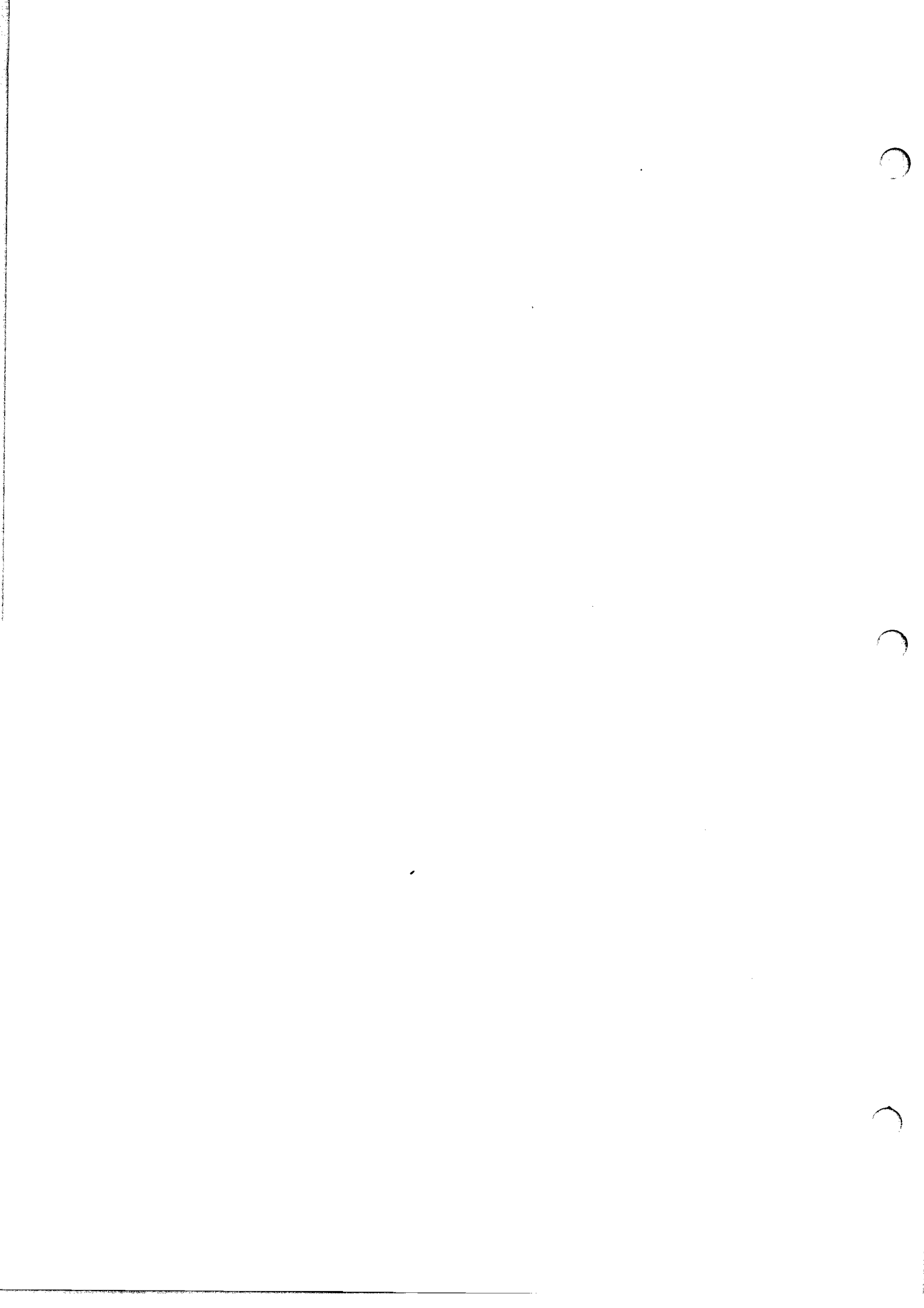
Purpose: To organize data into three data base models.

Lesson 5. Information Retrieval (35 minutes)

Slides: 1.16 - How is Information Retrieved?? *
1.17 - Accessing Files (serial and random) *
1.18 - Accessing Files (sequential by key) *

Activity: Quiz: Basic Concepts

Purpose: To test basic concepts of Module 1 via an on-line interactive quiz.



Module 1

Data Management

Introduction to the TurboIMAGE/DBMS 3000 Customer Training Course

Part 1. Instructor and student introductions (10 minutes)

Begin the class by introducing yourself and describing your position at Hewlett-Packard. Information on your educational background and job experiences at HP also may be of interest to the class.

Ask each student to introduce themselves by providing the following information:

- (1) Name
- (2) Company
- (3) Position (programmer, administrator, designer, etc.)
- (4) Programming languages used
- (5) Current projects involving data management
- (6) Objectives for taking this course

List the students' objectives on a flip chart and post it in the classroom. This will give you an opportunity to highlight the topics that will be covered in the course, by stating how each objective will be met by the end of the class. The student introductions also will give you the necessary information to form lab teams according to programming languages and job functions.

Part 2: Overview of Course (10 minutes)

Ask students to open the workbook to the Course Outline. Briefly highlight the topics in each module and the contents of each appendix. Also, point out the list of activities in the Table of Contents, the course goal and objectives, and the training curriculum. Provide the students with information about the training center (parking, restrooms, telephones, food), as well as the schedule for each day (starting/ending time, lunch, breaks).

Part 3: Introduction to the Reference Manual (10 minutes)

Ask students to open to the Table of Contents of the *TurboIMAGE Data Base Management System Reference Manual*. Emphasize the importance of using the reference manual throughout the course. Point out that specific references to sections of the manual are provided throughout the course. Highlight which sections of the manual will be used in each of the modules:

Section 1: Introduction
Module 1

Section 2: Data Base Structure and Protection
Module 2

Section 3: Defining a Data Base
Module 4

Section 4: Using the Data Base
Module 5, Module 6, Module 8

Section 5: Using the TurboIMAGE Library Procedures
Module 5, Module 6

Section 6: Host Language Access
All programming Labs

Section 7: Maintaining the Data Base
Module 8

Section 8: Using the Data Base Utilities
Module 8

(Continued on next page)

Module 1

Data Management

Part 3: Introduction to the Reference Manual (cont'd)

Section 9: Using a Remote Data Base
Module 10

Section 10: Internal Structures and Techniques
Module 2, Module 5, Appendix E

Appendix A: Error Messages
Module 4, Module 5, Module 6, Module 8

Appendix B: Results of Multiple Access
Module 6

Appendix C: Summary of Design Considerations
Module 9

Appendix D: Multiple RIN Special Capability
Module 6

Appendix E: TurboIMAGE Log Record Formats
Module 8

Appendix F: MPE Log Record Formats
Module 8

Appendix G: Recovery and Logging Quick Reference
Module 8

Appendix H: TurboIMAGE Conversion (DBCONV)
Module 8

Module 1-1

Data Management

Goal: To present the basic concepts of a data base and a data base management system, to provide basic information on access methods, and to identify differences between IMAGE/3000 and TurboIMAGE.

Objectives:

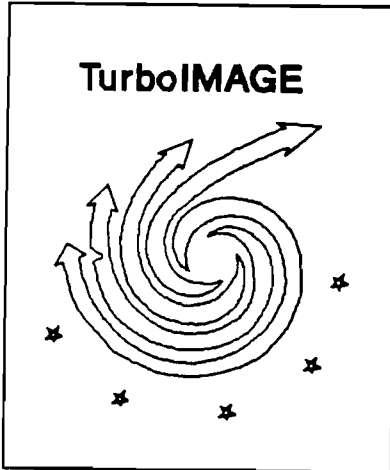
Upon completion of this module, you will be able to:

- contrast IMAGE/3000 and TurboIMAGE.
- define data base.
- define DBMS.
- identify advantages of using data base for information management.
- identify the three basic data base models.
- apply a specific case study to the data base models.
- contrast three basic file access methods.

TurboIMAGE

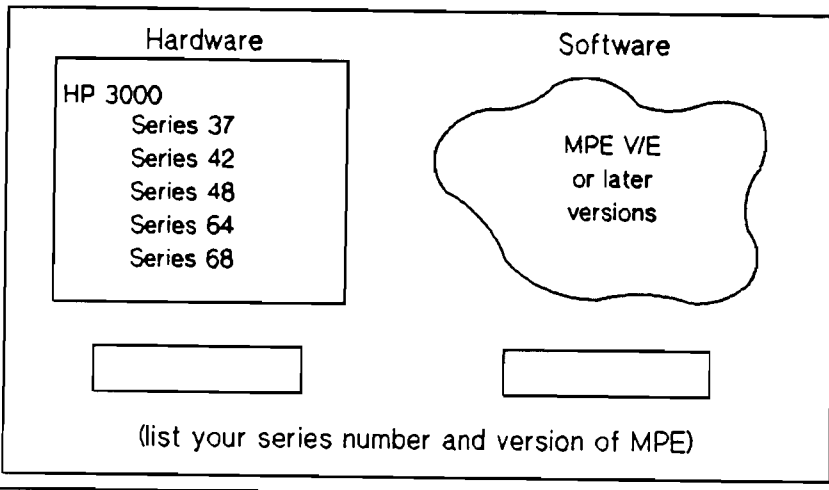
HP's Data Base Management System

TurboIMAGE is an enhanced version of IMAGE/3000.



- Higher Performance
- Relaxed Limitations
- Additional Options
- Easy Migration from IMAGE/3000

HP 3000 Environment for TurboIMAGE



Module 1-2 Instructor Notes: Slides 1.01/1.02

Data Management

Purpose: To introduce the key features of TurboIMAGE.

Key Points: *Check the end of this module for preparation information.*

- Over the years, the HP 3000 customer base has been expanding. Hardware and software configurations and limitations designed years ago are no longer adequate. This trend is evidenced by the continued improvements of HP hardware and the MPE operating system. TurboIMAGE is a response to the hardware and software improvements.
- The modifications provided through TurboIMAGE include:
 - increased performance
 - relaxed limitations allowing larger data bases
 - 100% forward compatible with IMAGE/3000 (requires a load/unload for use of capacity expansions)
 - increased throughput by enabling several data base operations to proceed concurrently (parallel processing)
 - easy migration of IMAGE/3000 data bases to TurboIMAGE data bases.

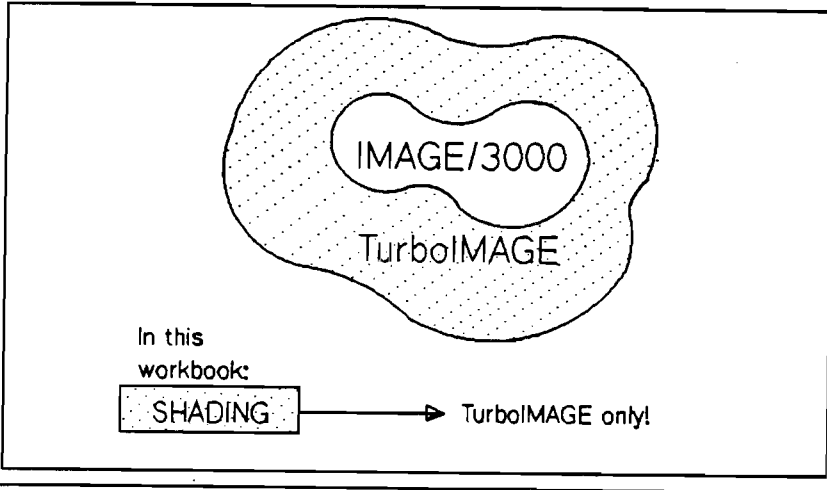
Purpose: To define the necessary HP 3000 environment for TurboIMAGE.

Key Points:

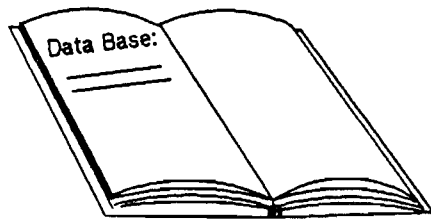
- TurboIMAGE is not available on Series I,II,III,30,33
- TurboIMAGE can run on Series 39,40,44 with a firmware update kit.
- TurboIMAGE can run on Series 64A and early Series 64B with a Diagnostic Control Unit (DCU) board.
- TurboIMAGE requires MPE V/E or later versions.
- TurboIMAGE has no dependency on disc cache.
- Affected software: QUERY(new version with TurboIMAGE)
Dictionary(update available)
Report(update available)
Inform(update available)
Transact(update available)
ADAGER(being updated)
BASIC(requires modification of programs that access the chain count in the status parameter array)
Any 3rd-party vendor software product that reads or modifies the root file, or is hard-coded to IMAGE/3000 limits, will no longer work on TurboIMAGE.

The Relationship Between IMAGE/3000 and TurboIMAGE

IMAGE/3000 is a subset of TurboIMAGE.



What is a Data Base?



:A collection of data organized for rapid search and retrieval.

Module 1-3 Instructor Notes: Slides 1.03/1.04

Data Management

Purpose: To differentiate between IMAGE and TurboIMAGE in the student workbook.

Key Points:

- Any text that is **shaded** is information that applies to TurboIMAGE ONLY!
- All other text applies to BOTH IMAGE and TurboIMAGE.

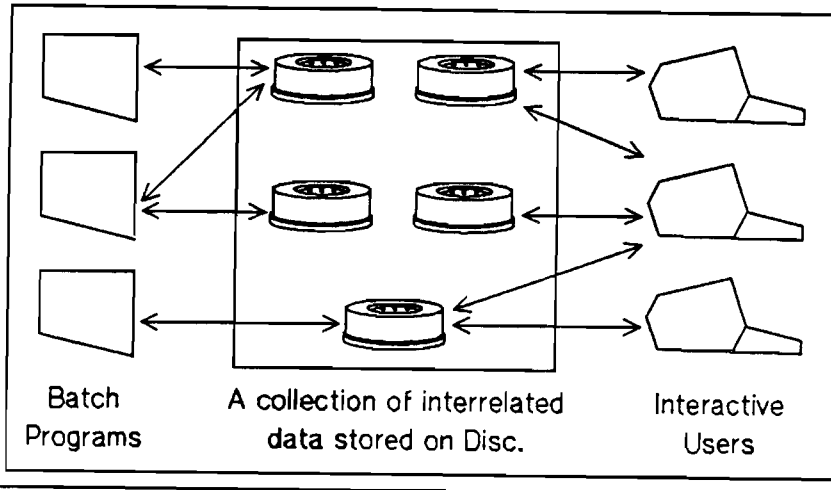
Purpose: To define the concept of a data base.

Key Points: *Check the end of this module for preparation information.*

- A data base is usually a large amount of data that is stored for long periods of time.
- The definition is not computer dependent. Therefore, a filing cabinet with files well organized and indexed would satisfy the definition.

If its concepts are clear in all minds the next several slides can be quickly glossed.

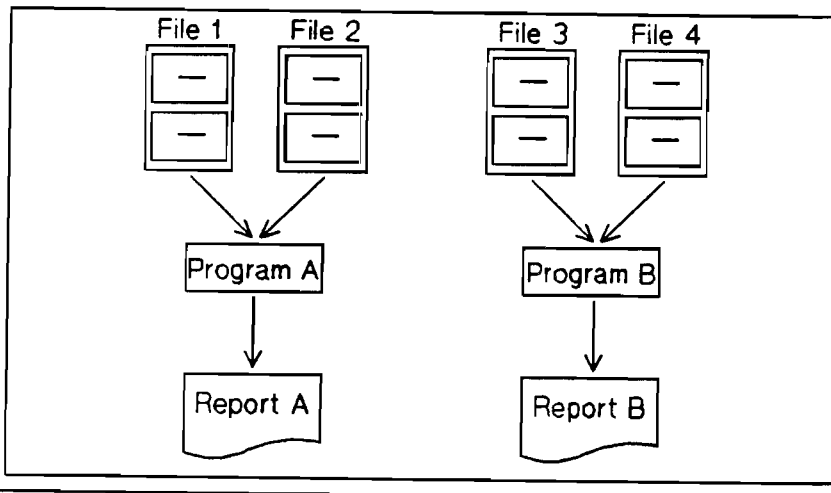
Data Base Interaction



T11 1.05

© 1985 Hewlett-Packard Company

Before Data Base



T11 1.06

© 1985 Hewlett-Packard Company

Module 1-4 Instructor Notes: Slides 1.05/1.06

Data Management

Purpose: To provide a conceptual understanding of the interaction between a data base and it's users.

Key Points: *Check the end of this module for preparation information.*

- A data base is a collection of interrelated data stored on a direct access storage device (e.g. magnetic disc).
- A data base is designed to be accessible to many users, both batch and interactive, all processing at the same time.

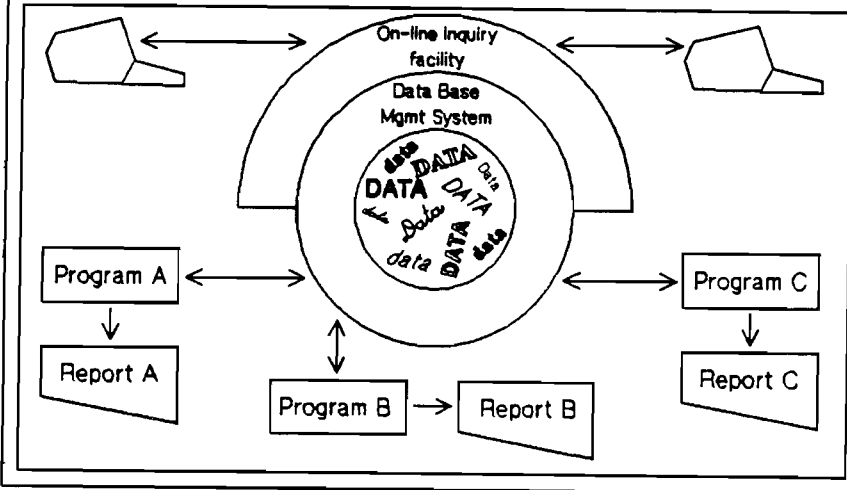
Purpose: To illustrate file-program dependence without a data base.

Key Points: *Check the end of this module for preparation information.*

- Programs are dependent on file form and content.
- Data redundancy is high. Each new program created may need data in different combinations. Therefore, a data item may occur in several different files.
- Program maintenance is costly. Since programs own their data files, any change in the format of the data file affects ALL the programs.
- Data may be inconsistent. There is no guarantee that updating fields in one file would also update the same fields in other files.
- Lack of control of data.
- Sharing data is difficult. Necessary data may be on several different files and may not always be available in the proper form.

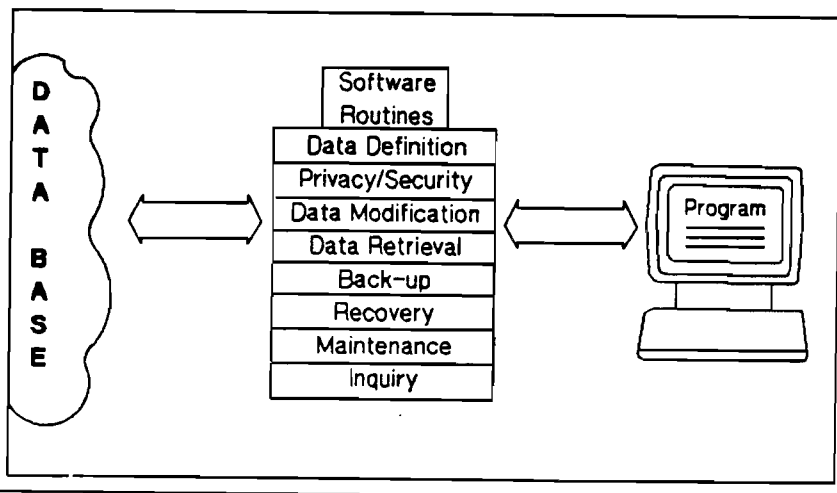
20 - 1000 (1-2)

After Data Base



The Data Base Management System is an interface between the data and application programs.

Data Base Management System



Module 1-5 Instructor Notes: Slides 1.07/1.08

Data Management

Purpose: To illustrate the data base approach to information management.

Key Points: *Check the end of this module for preparation information.*

- All data is merged into a single pool that allows sharing of data between programs, which makes separate files unnecessary. This centralized control over the data reduces redundancy, avoids inconsistencies, makes programs independent of the data, and provides alternate VIEWS of the data.
- It is possible to have an inquiry program that can act upon any data base.
- The necessary software to handle this pool of data is called a Data Base Management System (DBMS).

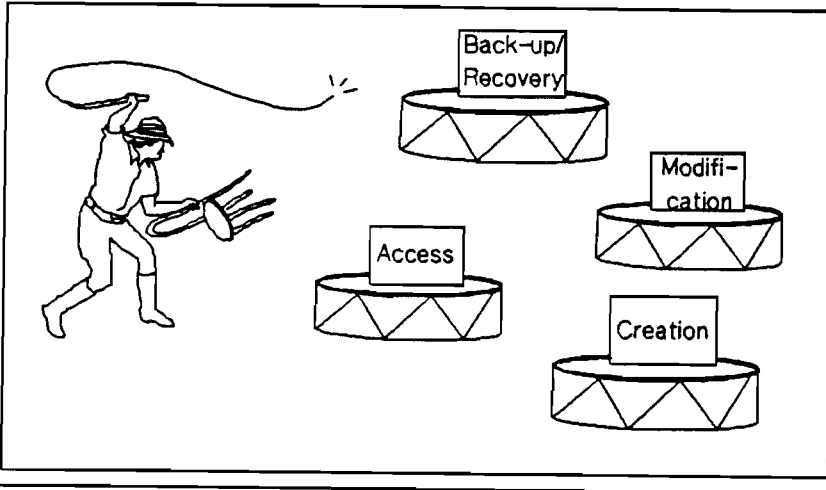
example: Views eg company, airline (1-17)

Purpose: To define a data base management system.

Key Points: *Check the end of this module for preparation information.*

- A DBMS is a collection of software routines that serves as the interface between application programs and the data base.
- A DBMS allows one or more users to use and/or modify the data in the data base.
- A DBMS allows the user to deal with the data in abstract terms, rather than in the way the computer stores the data.
- A DBMS usually consists of a DDL(data definition language), a DML(data manipulation language), and an inquiry facility. These will be discussed in more detail later.

Data Base Administrator



Data Base Models

- Hierarchical
- Relational
- Network

Module 1-6 Instructor Notes: Slides 1.09/1.10

Data Management

Purpose: To define the role of the data base administrator.

Key Points:

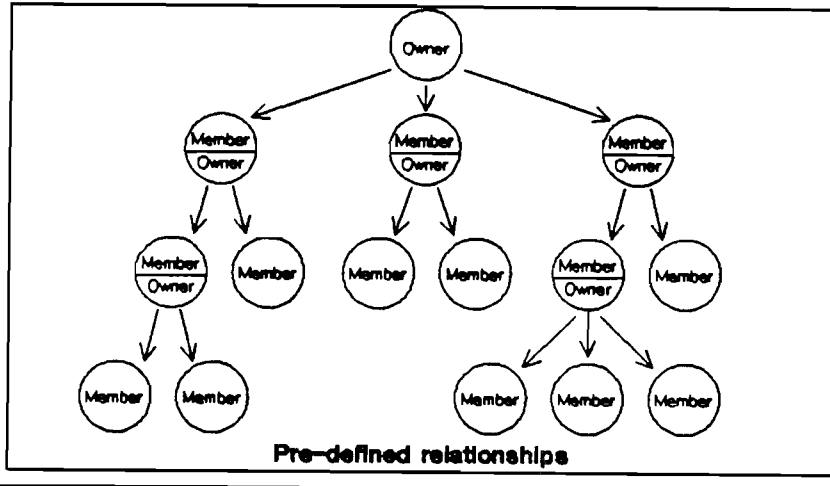
- A data base is handled by application programmers and users, but usually one high-level person is given responsibility for matters that deal with the data base as a whole. This person is the data base administrator (DBA).
- The responsibilities of a data base administrator include:
 - creation of the original description of the data base structure.
 - modification of the data base description when needed.
 - granting access to the data base or parts of it to various users.
 - backing up the data base and repairing damage to the data base caused by hardware or software failures.

Purpose: To introduce the three main data base models.

Key Points: *Check the end of this module for preparation information.*

- Three data base models serve as the basis for the vast majority of data base management systems on the market.
- Data bases are explained in terms of the structure they allow and the operations they allow on those structures.

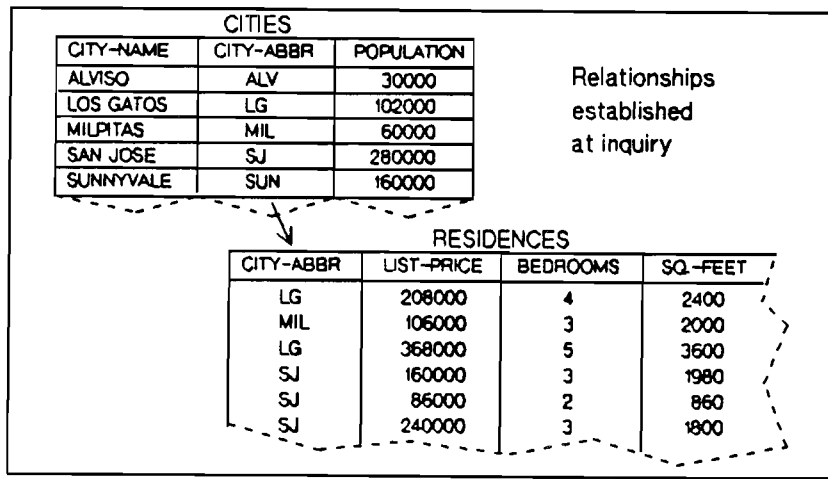
Hierarchical Model



T1 1.11

© 1985 Hewlett-Packard Company

Relational Model



T1 1.12

© 1985 Hewlett-Packard Company

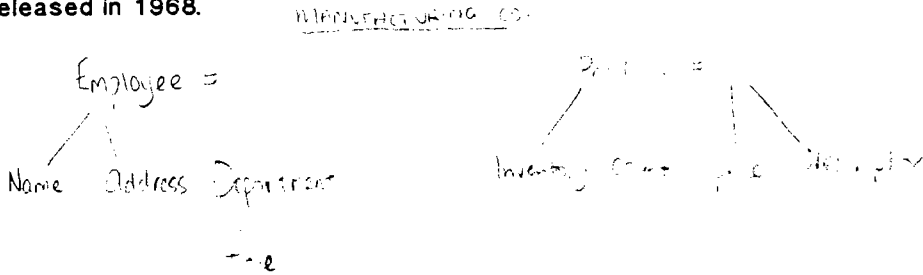
Module 1-7 Instructor Notes: Slides 1.11/1.12

□ Data Management

Purpose: To define the hierarchical data base model.

Key Points: *Check the end of this module for preparation information.*

- The hierarchical model is a "tree" structure in which a member can have only one owner.
- The hierarchical structure models one-to-many relationships such as one realtor-many properties or one department-many employees.
- IBM's hierarchical data base, called IMS, was one of the first such structures to be implemented, and was released in 1968.

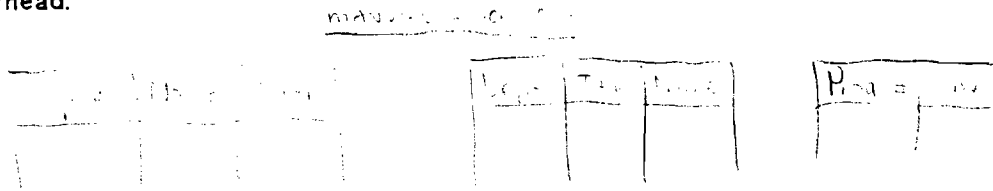


relationships defined at creation of data base

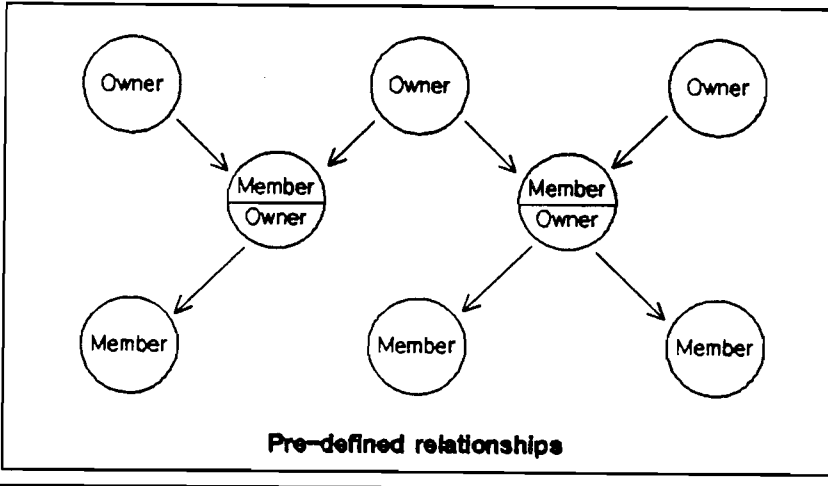
Purpose: To define the relational data base model.

Key Points: *Check the end of this module for preparation information.*

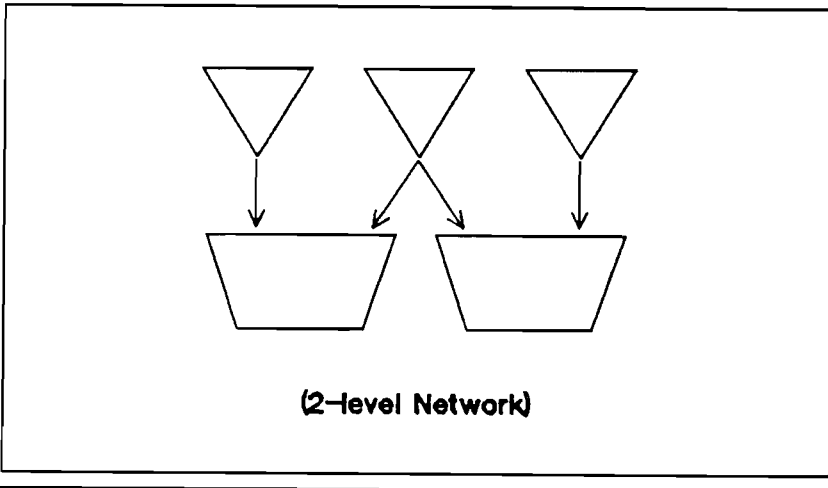
- Relational data bases are constructed using two dimensional tables containing related data items.
- The data is stored in table form and relations between tables are implemented via a common key item.
- Relational data bases are easy to use because the data is in a form easily understood by programmers and users alike.
- The major advantage of a relational data base is that the tables can be created as needed at inquiry time. However, this ability to create new relations causes a tremendous amount of overhead.



Network Model



TurboIMAGE is a Network Structure



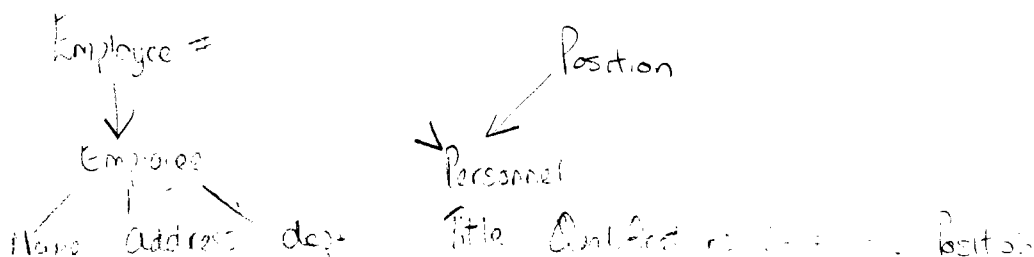
Module 1-8 Instructor Notes: Slides 1.13/1.14

□ Data Management

Purpose: To define the network data base model.

Key Points: *Check the end of this module for preparation information.*

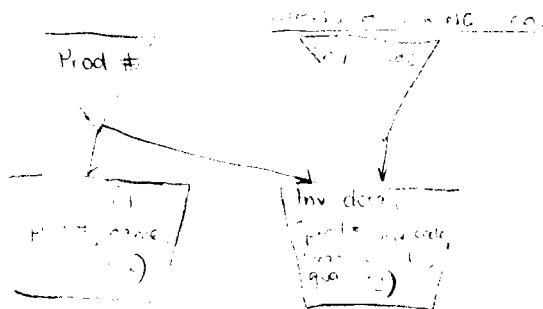
- The network model allows a member record to have more than one owner. Therefore, the network model is a superset of the hierarchical model.
- The network model allows more flexibility in accessing data and requires less system overhead for most inquiries.
- Simple network structures with pointers in only one direction are preferable because the overhead for chain maintenance is minimized.
- Both network and hierarchical models require relationships to be defined at data base creation time; these relationships are fixed thereafter. Other access requires either outside data manipulation (sorting) or restructuring the data base.



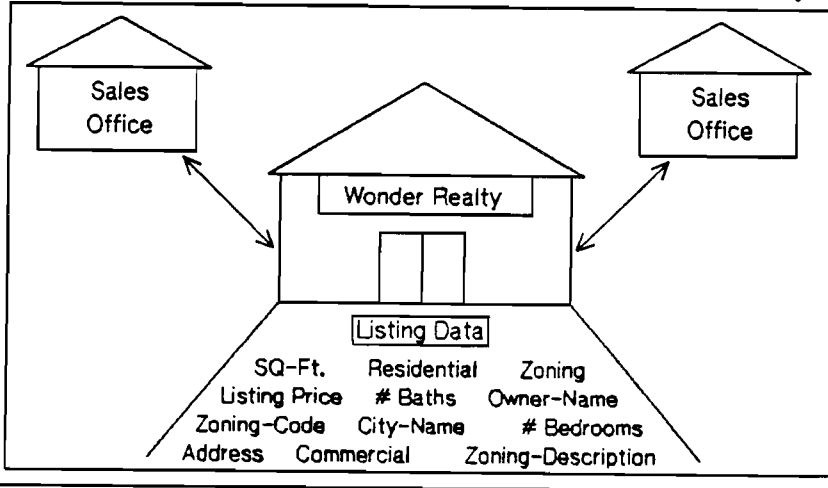
Purpose: To define the TurboIMAGE network model.

Key Points: *Check the end of this module for preparation information.*

- The TurboIMAGE structure is a two-level network structure.
- The triangles and trapezoids represent master and detail sets respectively and will be explained further in the next module.



Case Study



Module 1-9 Instructor Notes: Slide 1.15

Data Management

Purpose: To introduce a data base case study.

Key Points: *Check the end of this module for preparation information.*

- Wonder Realty handles residential and commercial properties.
- They want to computerize their listings to meet the following objectives:
 - minimize the sales cycle by quickly and accurately presenting buyers with properties that match their stated wants and needs.
 - implement the ability to list properties outside the local area.
 - enable remote offices to enter new listings as well as retrieve data on current listings.
 - expand the business to offices in areas serving major corporations' employee relocation needs.
 - increase revenues by selling their automation scheme to other real estate firms.

Module 1-10

Activity 1-1 Chalk Talk: Wonder Realty

Purpose: To discuss how data base concepts apply to Wonder Realty's objectives.

Directions: The Wonder Realty case study will be used throughout the course. Discuss with the instructor and the other members of the class, how a data base management system can help the company realize its objectives. List your ideas below.

- MINIMIZE SALES CYCLE

- REMOTE OFFICE ACCESS

- EXPAND BUSINESS

- INCREASE REVENUES

Module 1-10 Instructor Notes

Activity 1-1 Chalk Talk: Wonder Realty

Time: 15 minutes

Purpose: To discuss how data base concepts apply to Wonder Realty's objectives.

Notes: List specific characteristics of a data base management system that will help Wonder Realty to realize their objectives. (Emphasize that the students should list the information in their workbooks.)

Suggested responses to the various discussion items are provided below. Use these only if the students cannot provide responses encompassing the same concepts.

The next activity requires the students to think about the organization of data into a data base structure.

■ MINIMIZE SALES CYCLE

- *a DBMS will provide quick retrieval of property listings that match specific customer needs.*
- *new listings can be added or updated quickly and efficiently.*

■ REMOTE OFFICE ACCESS

- *using the inquiry facility, remote offices can access information about listings.*
- *new property listings from other areas can be added to the data base.*

■ EXPAND BUSINESS

- *by centralizing the data, the above two objectives are satisfied and the business can easily expand in quantity of listings and scope of listings.*

■ INCREASE REVENUES

- *if the business expands by providing better and faster service, the revenues will increase.*
- *if the data base management system is highly successful for Wonder Realty then perhaps they can sell it to other real estate companies.*



Module 1-11

Activity 1-2 Worksession: Data Base Models

Purpose: To organize data into three data base models.

Directions: The basic structures of the three data base models are presented for you. Organize the data items for Wonder Realty into each of the models.

You do not need to use all of the data items listed. You may also use items not listed if you feel they fit better into a particular structure. The goal is to see how the various structures can be used to organize data.

MANY DIFFERENT SOLUTIONS ARE POSSIBLE IN EACH CASE!

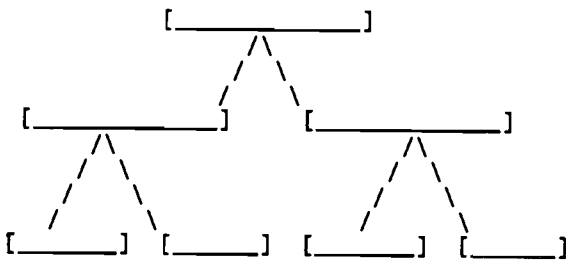
Specifications:

The data items used by Wonder Realty are:

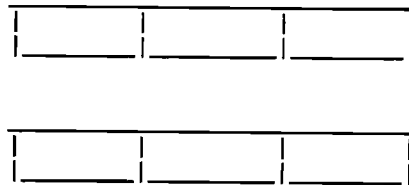
city name, address, owner's name, listing price, zoning code, zoning description, number of bedrooms, number of baths, number of square feet

Wonder Realty has residential and commercial listings.

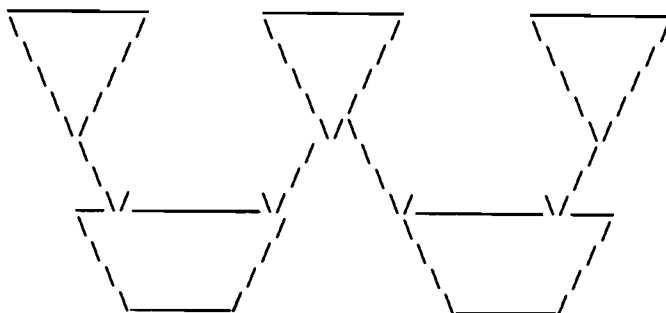
Hierarchical Structure



**Relational Structure
(2 possible tables)**



Network Structure (TurboIMAGE)



Solution: Refer to Appendix A for sample solutions.

Module 1-11 Instructor Notes

Activity 1-2 Worksession: Data Base Models

Time: 20 minutes

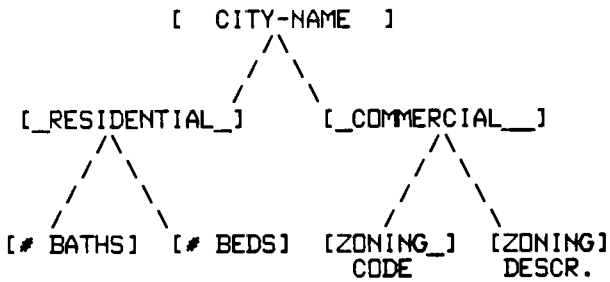
Purpose: To organize data into three data base models.

Directions:

1. The three basic data base structures are provided for the student.
2. Emphasize the fact that many solutions are possible.
3. Emphasize that the students can use any of the data items from the Wonder Realty data base.
4. Before the students begin the activity, point out that Wonder Realty has both RESIDENTIAL and COMMERCIAL listings. Ask the students to classify the list of data items into the two categories. This will help give them a start on the activity.

Solutions:

Hierarchical Structure

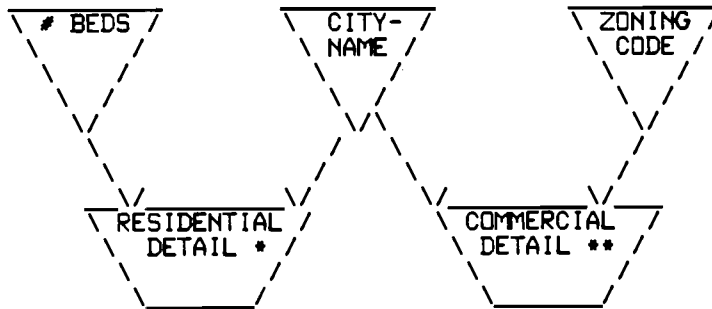


**Relational Structure
(2 possible tables)**

ZONING	ZONING	CITY-
DESCR.	CODE	NAME

CITY-	ADDRESS	LIST-
NAME		PRICE

Network Structure (TurboIMAGE)

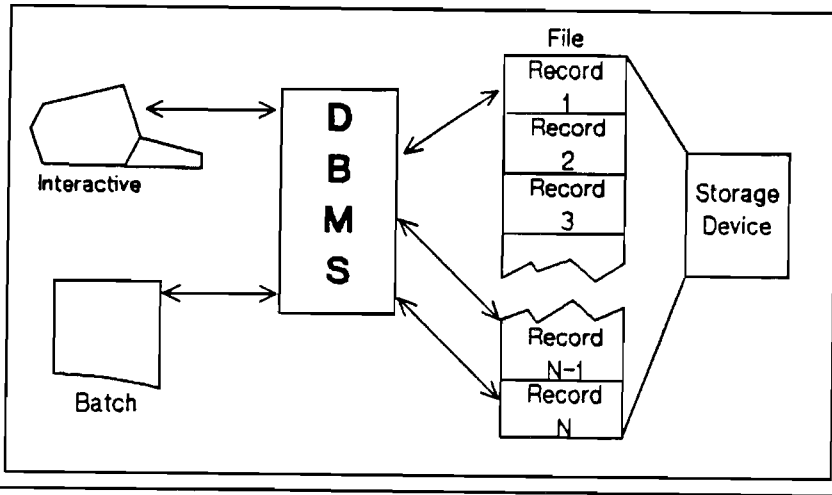


* city-name, address, owner's name, listing price, # of beds, # of baths, # of square feet.

** city-name, zoning code, zoning description, address, owner's name, # of square feet, listing price.

End this - LUNCH.

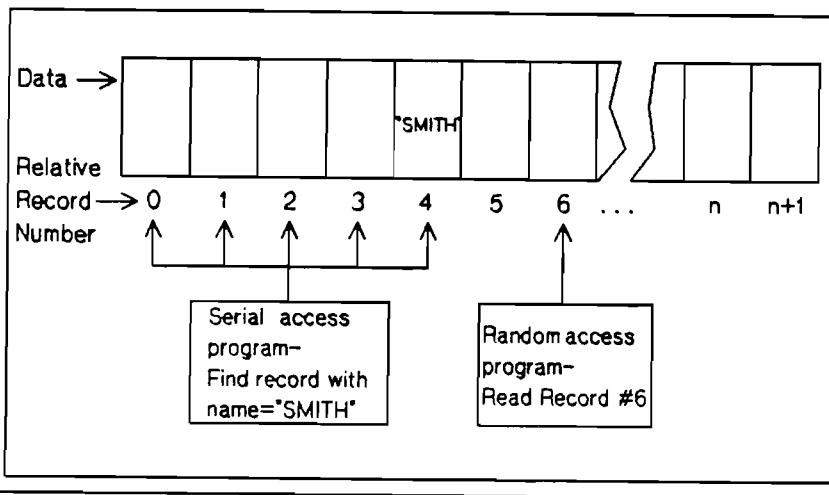
How is Information Retrieved?



T11 1.16

© 1985 Hewlett-Packard Company

Accessing Files



Random access is also known as directed access.

T11 1.17

© 1985 Hewlett-Packard Company

Module 1-12 Instructor Notes: Slides 1.16/1.17

Data Management

Purpose: To present an overview of methods of retrieving information from files.

Key Points: *Check the end of this module for preparation information.*

- Data items are stored in records with other associated data such that one record consists of many items of data that individually have little meaning. Many associated data items give meaning to each other, therefore, the concept of "information."
- Information is usually stored so it can be retrieved quickly. This gives rise to the fact that information is stored in files such that easy retrievals, additions and modifications can be performed.
- Accessing files is a method of information retrieval.
- Files may be stored on tape or disc.

Purpose: To introduce serial and random access methods.

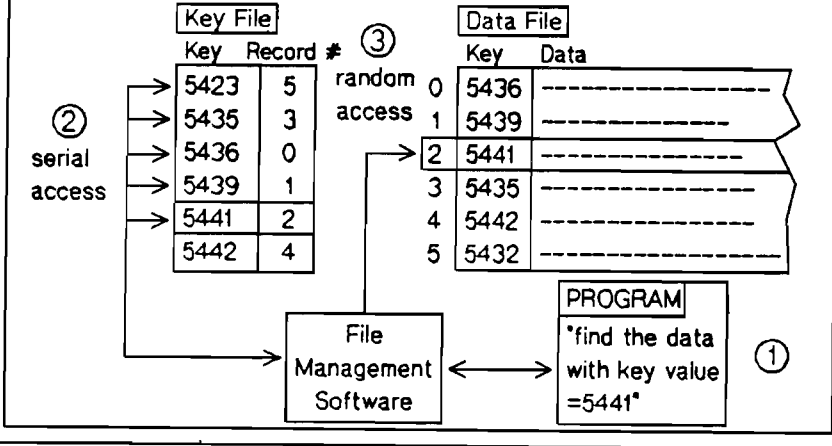
Key Points: *Check the end of this module for preparation information.*

- Serial access requires that each record is read one after the other in the order in which they occur in the file.
- Magnetic tape is an example of serial access media.
- Random or directed access selects a specific record in the file. To achieve this type of retrieval, the specific record's relative record number(position in the file) must be known.
- Most computer systems allow serial or random access through the file system.

Accessing Files



■ SEQUENTIAL BY KEY - combination of serial and random



Module 1-13 Instructor Notes: Slide 1.18

Data Management

Purpose: To define keyed sequential access.

Key Points: *Check the end of this module for preparation information.*

- Sequential access takes place in specialized files where access is gained through a "key."
- In these types of files, the records are sorted according to a key value. Then a key file is formed containing the key values and the relative record number of the associated full data record. This key file serves as an index to the data file.
- The key file and subsequent access to the data file are handled and maintained by the "file management software." Without such software, a programmer would need to write a program to access the key file sequentially and then use the relative record number to access the data file directly.
- Examples of file management software:
 - KSAM (Keyed Sequential Access Method-HP 3000)
 - ISAM (Indexed Sequential Access Method-IBM)
 - VSAM (Virtual Sequential Access Method-IBM)

Topic

→ files access

* sequential

* direct

* indexed

* computer-aid

⇒ Module 5

Module 1-14

Activity 1-3 Quiz: Basic Concepts

Purpose: To test basic concepts of Module 1 via an on-line interactive quiz.

Directions:

1. Type :QUIZ 1 to run the interactive session. Each student should run the quiz individually.
2. A copy of the quiz can be sent automatically to the line printer at the end of the session.

Module 1-14 Instructor Notes

Activity 1-3 Quiz: Basic Concepts

Time: 20 minutes

Purpose: To test basic concepts of Module 1.

Notes: This activity is an on-line quiz and simulation of basic concepts from Module 1. The quiz consists of eight questions. Informative responses will be provided for both correct and incorrect answers.

The student can request a listing of the quiz at the end of the session. However, only one listing per session will be printed. Therefore, encourage the students to run the session individually

A simulation is presented at the end of the quiz to give the students a method of determining whether IMAGE/3000 or TurboIMAGE is installed on a particular system.

Directions:

1. Be sure to set and release the UDC file ACCUDC.PUB at the account level. The students will initiate this session by typing the UDC :QUIZ1.
2. The questions and answers are in the file QUIZ1.SOLUTION.IMAGE. This file will be sent to device class LP upon request by the student at the end of the session. A copy of this file is listed below.

Solutions:

TurboIMAGE - Quiz - Module 1

1. Name the method of file access that requires knowledge of the relative record number.

INCORRECT RESPONSES: Serial - Serial access to a file will retrieve records in order of occurrence in the file. Therefore, the relative record number would not be needed to access a record in the file.

CORRECT RESPONSE: Random or Directed - Random or directed access means that a particular record in a file can be retrieved without reading each record preceding the desired one in the file. Therefore, in order to find one particular record in a file the address or relative record number would be necessary information to locate the record.

2. Name Hewlett-Packard's sequential file management software.

INCORRECT RESPONSES: ISAM or VSAM - IBM products.

CORRECT RESPONSE: KSAM or Keyed Sequential Access Method. Sequential access is a combination of serial and random access. It requires searching a key file that is in sorted order. The key file contains a KEY value and the relative record number of the complete data record.

Review section 4 objectives - check they're done.

Module 1-14 Instructor Notes

Activity 1-3 Quiz: Basic Concepts (cont'd)

3. Which data base model allows a member to have only one owner?

INCORRECT RESPONSES: Network - The network model allows a member to have more than one owner. Refer to slide 1.13 for a diagram of this structure.

Relational - The relational model is not a tree or graph structure. The data is structured into tables. Therefore, the terms member and owner have no meaning in this model. Refer to slide 1.12.

CORRECT RESPONSE: Hierarchical - The hierarchical data base model is a tree structure in which a member can have only one owner. Refer to slide 1.11 for a diagram of this structure.

4. Which data base model does TurboIMAGE represent?

INCORRECT RESPONSES: Relational, Hierarchical - These are two of the three basic data base models.

CORRECT RESPONSE: Network - TurboIMAGE is a two-level network structure. Refer to slide 1.13 for a diagram of this structure.

5. Which data base model creates data item relationships at inquiry time?

INCORRECT RESPONSES: Network, Hierarchical - The relationships represented in these two models are defined when the data base is created.

CORRECT RESPONSE: Relational - The relational data base model consists of two-dimensional tables. A given table displays relationships between data items. These relationships are determined at inquiry time. This provides a great deal of flexibility but also requires a greater amount of overhead than the other models.

6. Which of the following items is NOT a benefit of using a data base as opposed to other forms of data management: (1) reduced data redundancy, (2) program dependence on data, (3) reduced possibility of data inconsistencies.

INCORRECT RESPONSES: (1) - Because a data base is actually a pool of data that can be accessed by many users and programs, data redundancy is reduced.

(3) - The use of a data base for information management eliminates the necessity of data items appearing in several different files which are accessed by different programs. Therefore, inconsistencies caused by updating data items is avoided.

CORRECT RESPONSE: (2) - One of the main benefits of managing information via a data base is that programs are not dependent on specific data files.

Module 1-14 Instructor Notes

□ Activity 1-3 Quiz: Basic Concepts (cont'd)

7. Which of the following is NOT a responsibility of a data base administrator: (1) creating programs that access the data base, (2) repairing damage to the data base caused by software failures, (3) granting access to the data base to various users, (4) creating the original description of the data base structure.

INCORRECT RESPONSES:

- (2) - A data base administrator is responsible for backing up the data base and repairing damage to the data base caused by hardware and software failures.
- (3) - A data base administrator is responsible for granting to various users the authorization to access the data base or a part of the data base.
- (4) - A data base administrator is responsible for creating the original description of the data base structure.

CORRECT RESPONSE: (1) - A data base administrator usually does not program data base applications.

8. What is the minimum version of MPE that is needed to run TurboIMAGE?

CORRECT RESPONSE: V/E - MPE V/E is the minimum version necessary to run TurboIMAGE.

The next part of this quiz will demonstrate a quick, easy method for determining whether or not TurboIMAGE is installed on your system.

When an IMAGE/3000 or TurboIMAGE utility program is run, the banner that appears will indicate the version number of the product.

When the IMAGE/3000 utility DBUTIL is run, the following banner appears. If your system does NOT have TurboIMAGE installed, this number is preceded by the letter B. When the same utility program is run under TurboIMAGE, the version number is preceded by a C.

```
:RUN DBUTIL.PUB.SYS
```

```
HP32215B.04.30 IMAGE/3000: DBUTIL (C) COPYRIGHT HEWLETT-PACKARD CO.
```

```
:RUN DBUTIL.PUB.SYS
```

```
HP32215C.00.00 TurboIMAGE/3000: DBUTIL (C) HEWLETT-PACKARD CO. 1985
```

The preceding example illustrates a quick, easy method for determining which data base management system is installed on your system.

Module 1-15 Instructor Notes

Preparation Material: Slides 1.01, 1.04

Teaching Tips:

The next few slides are necessary since some students may not yet have TurboIMAGE on their systems or may not be able to run TurboIMAGE on their systems. Poll the class to determine how many have IMAGE/3000 and how many have TurboIMAGE. If you have students in both categories, it will be necessary to continually point out that there are a few differences between IMAGE/3000 and TurboIMAGE.

Some students may not know which of the two products is on their system. Slide 1.02 and the activity at the end of this module will provide the necessary information for determining which product is installed on a system.

Teaching Tips:

Give several examples of data bases and have students think about their own examples of data bases. Some suggested examples are:

1. An airline with information about passengers(which passengers have seats on which flights?), flights(who is the pilot,copilot on which flight?), aircraft(when was each aircraft last serviced?), etc.
2. A company with information about employees(name,social security#, address, length of service), stockholders(name, number of shares), inventory(item numbers, quantity), payroll(emp. ID, entity, salary).
3. A real estate company with information about listings(commercial, residential, listing number, address, price), agents(sales region, clients, sales figures), etc.
4. A library with information about books(title, author, subject), periodicals(title, date), patrons(books checked out, fines owed), etc.
5. A drug store chain with information about customers(doctor's name, prescription numbers, number of refills,etc.), drugs(generic name, commercial name, suggested quantity, etc.).

Module 1-16 Instructor Notes

Preparation Material: Slides 1.05/1.06

Teaching Tips:

Emphasize the fact that the data base is a COLLECTION of data stored in a form that is ACCESSIBLE to all necessary users. Without computers this might be a central filing system available to all persons needing information about the data base.

Point out that the term ACCESSIBLE means the ability for users to store, retrieve, add, update, and delete data from the data base.

Teaching Tips:

The purpose of this slide is to show the drawbacks to a traditional file organization of data. In the previous slide, the students were told that the objective of data management was to store, retrieve, add, delete, and update data. Discuss some of the problems that might arise from this file-program dependence scheme by looking at a simple example:

A company stores data on all its employees, including name, address, social security number, date hired, salary, manager's name, job title, job description, job location. The personnel department needs to generate reports including name, address, date hired, salary, and social security number. Managers need to generate reports including name, manager's name, job title, job description, job location, and salary. Therefore, different files and programs exist to generate these reports but data is duplicated in the files for the two programs. When a new employee arrives, BOTH sets of files must have the necessary information added. When an employee's salary changes, care must be taken to insure that the salary item is updated in BOTH sets of files.

This example (or any other you might use) should emphasize the key points of data management WITHOUT a data base management system.

Module 1-17 Instructor Notes

Preparation Material: Slides 1.07/1.08

Teaching Tips:

Emphasize how the data base form of data management solves the problems pointed out in the previous slide. Explain that this course will go into detail on the various levels shown on the slide: organizing and accessing the data, the data base management system as defined by IMAGE or TurboIMAGE, and the on-line inquiry facility (QUERY/3000).

One main advantage of a data base is that it allows different VIEWS of the data. A VIEW is an abstract model of a portion of the data base. In the example from the previous slide, the two views discussed were the personnel view and the management view. In the airline example mentioned earlier, one view might be that of the reservation service. A user with this view would be concerned with information about flights and passengers but not with information about aircraft and pilots. Other views might be defined as the dispatcher's view (aircraft, pilots, service) or the personnel view (employee information). Even though each of these views are concerned with different data items, a data base can accommodate these views without the possibility of data redundancy and/or inconsistency.

Ask students to think about the other examples of data bases that were discussed previously and state alternate views of each data base.

Teaching Tips:

This slide presents a more detailed definition of the functions of a data base management system. Emphasize the fact that these software routines represent an INTERFACE between the user's (batch and interactive) and the actual data. Each of the mentioned functions will be covered as the course progresses.

Module 1-18 Instructor Notes

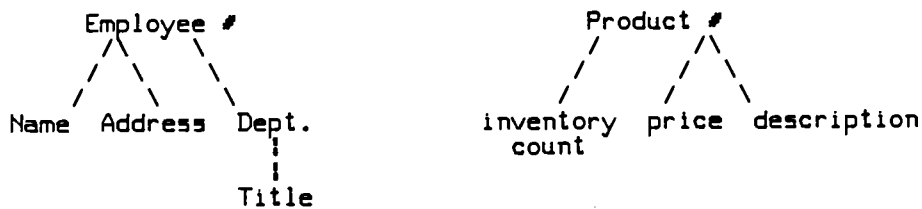
Preparation Material: Slides 1.10/1.11

Teaching Tips:

At this point, students should have a good understanding for the concepts involved in a data base management system. Therefore, more specific information will be presented in terms of implementing a data base management system. First, we will present three popular methods for structuring the data items. Point out that these are the three most widely used data base structures and each has advantages and disadvantages. Do not dwell on any one of these structures until the TurboIMAGE network structure is presented. The students should start thinking, however, as to how specific data might fit into each of these structures.

Teaching Tips:

Present the following example of a hierarchical structure of a data base for a manufacturing company. This sample structure consists of two trees. Point out that each member has only one owner and that all relationships are defined at data base creation time.



Write this example on the chalkboard or flip chart. Tell students that the first activity will require them to organize items of information into data base structures.

Module 1-19 Instructor Notes

Preparation Material: Slides 1.12/1.13

Teaching Tips:

Present the following example of a relational data base model for a manufacturing company. Point out that the tables shown below are created at inquiry time. Therefore, many possible tables can be formed.

Employee #	Name	Address	Dept.	Title	Name	Prod. #	Inv.

Write these examples on the chalkboard or flip chart.

Teaching Tips:

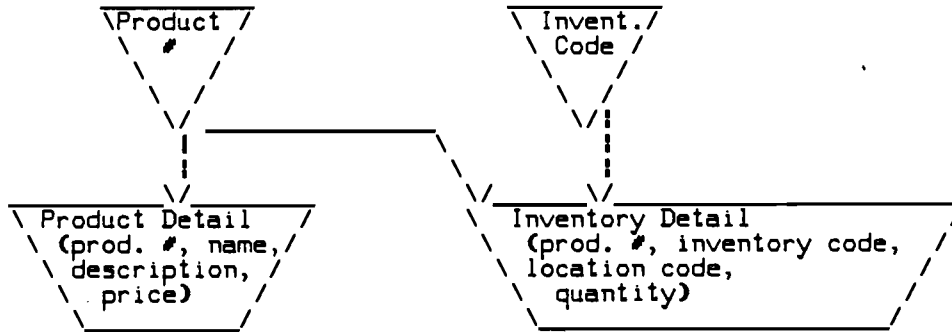
This slide represents a general network structure. The next slide presents the TurboIMAGE network structure. Therefore, do not give an example of a network structure until the TurboIMAGE structure is presented. However, point out on this slide that, unlike the hierarchical structure, the network structure allows a member to have more than one owner. Also emphasize that the relationships are determined at data base creation.

Module 1-20 Instructor Notes

□ Preparation Material: Slides 1.14/1.15

Teaching Tips:

Present the following example of a TurboIMAGE two-level network structure for a manufacturing company data base. Emphasize that a member (detail set) can have more than one owner (master set).



Write this example on the chalkboard or flip chart.

Teaching Tips:

Now that the structure of a data base has been discussed, it is time to present the case study that will be used as an example throughout the course. Encourage the students to consider other examples of data bases as the course progresses.

Discuss Wonder Realty's objectives with the class. The next two activities are intended to have the students analyze these objectives and decide how a DBMS might satisfy these objectives.

Module 1-21 Instructor Notes

Preparation Material: Slides 1.16/1.17

Teaching Tips:

Thus far we have discussed data base concepts and data base structures. Now we will look at ways that the data can be accessed from files. In this module, we look at an overview of information retrieval concepts. The students may already be familiar with these access methods, so do not dwell on specifics.

Teaching Tips:

This slide and the next slide provide a basic introduction to file access methods. Specific TurboIMAGE data base access methods will be presented in a later module. At this time, it is important that the students understand the basic concepts of file access methods.

Module 1-22 Instructor Notes

Preparation Material: Slide 1.18

Teaching Tips:

After discussing the methods of file access, tell the students that TurboIMAGE offers four types of data base access: sequential, direct, chained, calculated. These methods will be presented in detail in Module 5, Data Base Access.



Module 2 Instructor Notes

☐ TurboIMAGE Structure and Terminology

Overview of Module 2

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. Data Structures (45 minutes)

Slides: 2.01 - TurboIMAGE Data Structures
2.02 - TurboIMAGE Data Structures
2.03 - TurboIMAGE Data Sets
2.04 - Master Data Sets *

Activity: Chalk Talk: Master Data Sets

Purpose: To discuss the differences between manual and automatic master data sets.

Slides: 2.05 - Detail Data Sets *

Lesson 2. Terminology and Internal Structure (50 minutes)

Slides: 2.06 - TurboIMAGE Data Entry Terms *
2.07 - Data Set Relationships
2.08 - Media Record: Pointers + Data
2.09 - Bit Map
2.10 - Data Set Layouts
2.11 - The REALTY Data Base

Activity: Lab: Introduction to the REALTY Data Base

Purpose: To gain familiarity with the REALTY data base and the program that will be used in several labs throughout the course.

Lesson 3. Components of a DBMS (1 hour 25 minutes)

Slides: 2.12 - Components of TurboIMAGE *
2.13 - TurboIMAGE Overview
2.14 - QUERY/3000 *
2.15 - Getting Started with QUERY/3000
2.16 - QUERY Special Characters
2.17 - DEFINE and FORM Commands
2.18 - FIND Command
2.19 - REPORT and LIST Commands
2.20 - ADD Command *
2.21 - REPLACE and DELETE Commands *
2.22 - OUTPUT Command

Activity: Lab: Using QUERY/3000

Purpose: To formulate queries to the REALTY data base using QUERY/3000.



Module 2-1

TurboIMAGE Structure and Terminology

Goal: To present TurboIMAGE data structures, TurboIMAGE terminology, and the basic components of the TurboIMAGE DBMS.

Objectives:

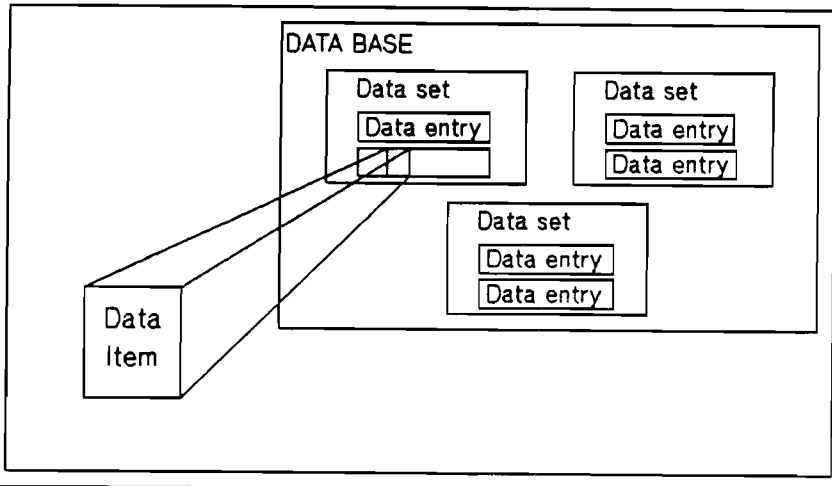
Upon completion of this module, you will be able to:

- define and give examples of data item, data entry, and data set.
- define and give examples of two types of master data sets and a detail data set.
- understand the concepts of media records and bit maps.
- identify the four basic components of TurboIMAGE.
- use QUERY/3000 to inquire about and modify a data base.

☐ TurboIMAGE Structure and Terminology

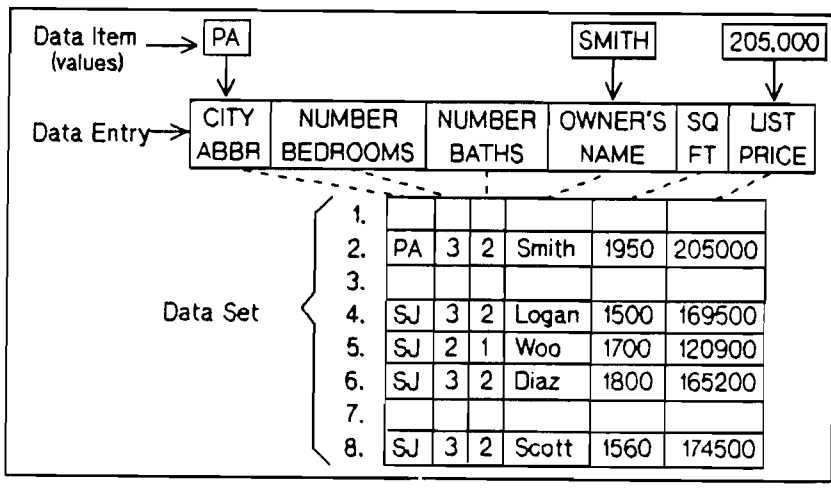
☐ Notes

TurboIMAGE Data Structures



DATA ITEM -- field
 DATA ENTRY --record
 DATA SET --file
 DATA BASE --collection of files

TurboIMAGE Data Structures



☐ TurboIMAGE Structure and Terminology

Purpose: To define the basic TurboIMAGE data structures.

Key Points:

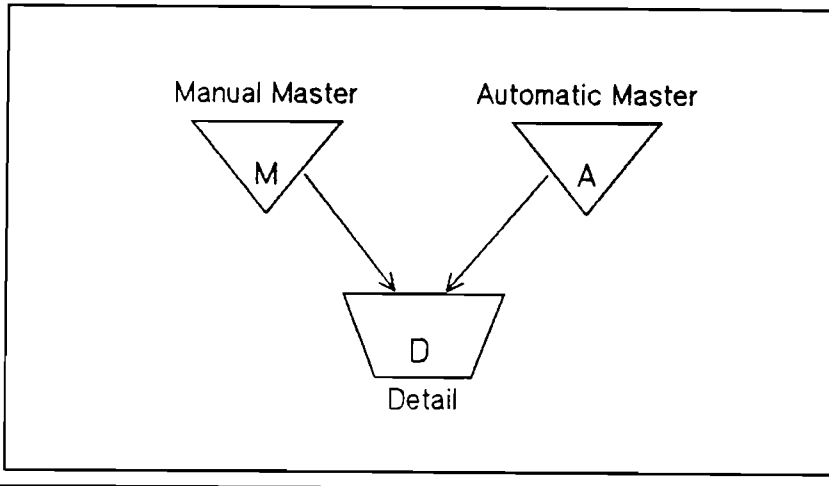
- A data item is the smallest unit of information in a data base structure. It is analogous to a field in an ordinary file structure.
- A group of associated data items make up a data entry. The data items are in a specified order in the data entry. A data entry is analogous to a record.
- A collection of entries make up a data set. Each entry in the data set occupies a specific location relative to the beginning of the file. A data set is an MPE privileged file. Standard MPE commands such as PURGE or RENAME will not work on privileged files. TurboIMAGE procedures access the data sets through the MPE file system.
- A collection of data sets (collection of files) make up a data base.

Purpose: To give examples of the basic TurboIMAGE data structures.

Key Points:

- The data items in a data set have names. In the slide, the data items are CITY-ABBR, NUMBER-BEDROOMS, NUMBER-BATHS, OWNERS-NAME, SQ-FT, and LIST-PRICE.
- When the above mentioned data items are listed together for a particular real estate listing, a data entry is formed.
- A set of these entries forms a data set.

TurboIMAGE Data Sets

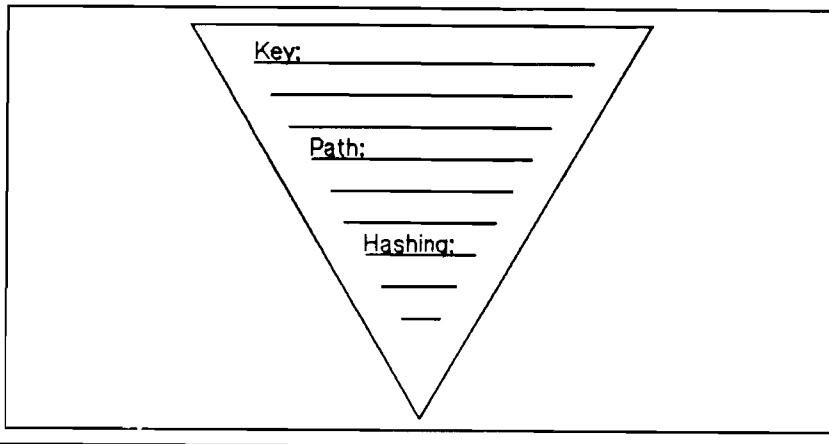


TI1 2.03

© 1985 Hewlett-Packard Company

Master Data Sets (Both Types)

- Serves as an index to a detail set.



- The terms KEY and SEARCH ITEM can be used interchangeably.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: Data Set Types and Relations and Section 10: Primary Address Calculation

TI1 2.04

© 1985 Hewlett-Packard Company

□ TurboIMAGE Structure and Terminology

Purpose: To define the three types of TurboIMAGE data sets.

Key Points:

- TurboIMAGE data sets fall into two main categories – master and detail.
- Two types of master data sets are used – manual and automatic.
- The data sets and the relationships between the data sets are represented in a block diagram. The block diagram consists of triangles, trapezoids, and arrows.
- Master data sets are represented by triangles.
- Detail data sets are represented by trapezoids.
- Master and detail data sets are connected by paths. The paths are represented by arrows.

Purpose: To present the basic characteristics of both types of master data sets.

Key Points: *Check the end of this module for preparation information.*

- A master data set serves as an index to the related detail data sets.
- A master data set contains unique search item values, or keys.
- A master data set can have up to 16 paths to detail data sets. In extreme cases, these paths could be to 16 different detail sets or to 1 detail set.
- The location of an entry in a master data set is determined by passing the entry's search item (key value) through an address calculation algorithm (hashing) whose result is a relative record location within the master data set.

See hashing example at end if necessary.

Block.

Module 2-4

Activity 2-1 Chalk Talk: Master Data Sets

Purpose: To discuss the differences between manual and automatic master data sets.

Directions:

List the characteristics and advantages of each type of master data set. Give appropriate examples of each type.

MANUAL

EXAMPLES

AUTOMATIC

EXAMPLES

Module 2-4 Instructor Notes

□ Activity 2-1 Chalk Talk: Master Data Sets

Time: 20 minutes

Purpose: To discuss the differences between manual and automatic master data sets.

Directions:

Sec 2-6 in IMAGE manual.

Discuss the differences between manual and automatic data sets and list the main points on the chalkboard. Instruct the students to take notes on the discussion in the student workbook. Ask the students to provide examples in both categories and to discuss the advantages and disadvantages of using the two types. Emphasize that the choice between the two types boils down to a choice between the control of data entry that is available through the use of manual masters and the time savings possible through the use of automatic masters.

MANUAL

EXAMPLES

- All entries must be explicitly (manually) added or deleted by the user program.
- Can contain data items in addition to the key value.
- A master entry **MUST** exist before a related detail entry with a matching key can be added.
- Key values of existing master entries serve as a table of legitimate search item values for all related detail data sets. Therefore, manual masters can be used to prevent the entry of invalid data in the related detail data sets.
- Can be stand alone (not related to detail sets). Key access to the master set is still available.

City names and abbreviations
- TurboIMAGE would allow addition of entries only if the city abbreviation already exists as an entry in a manual master.

Employee's name
- A legitimate name would have to exist in the manual master before data entries could be added to the associated detail set.

AUTOMATIC

EXAMPLES

- All entries are implicitly (automatically) added or deleted as related detail entries are added or deleted.
- Cannot be stand alone (must be related to at least one detail set).
- Must contain only one data item, the search item (key).
- Saves time when search item values are unpredictable or so numerous that manual addition and deletion of master entries is undesirable.

List-price
-possible values too numerous to add manually

Customer's P.O. number
-values may be too unpredictable to be maintained manually.

Date
-possible values too numerous to add manually.



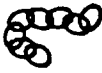
Detail Data Sets

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: Data Set Types and Relations

Search Item:

Sort Item:

TurboIMAGE Data Entry Terms

	Relative record number
	_____ _____
	Pointer
	_____ _____
	Chain
	_____ _____
1 2 3	Count
	_____ _____

□ TurboIMAGE Structure and Terminology

Purpose: To present the characteristics of a detail data set.

Key Points: *Check the end of this module for preparation information.*

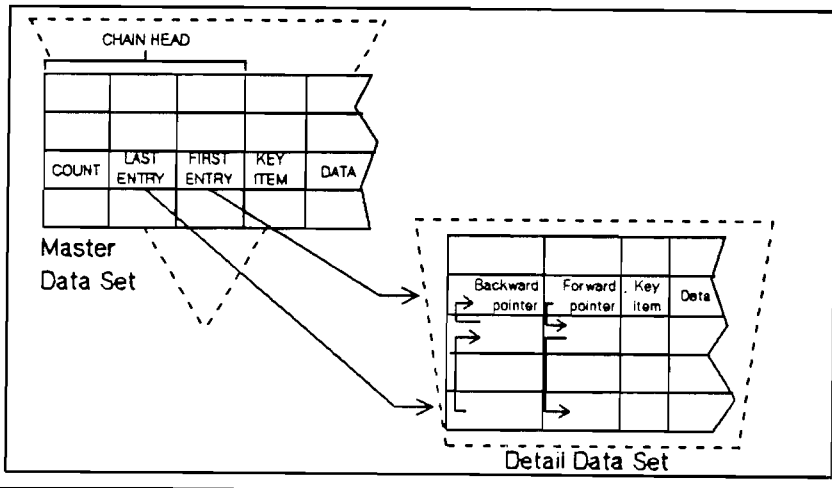
- Detail data sets record information about related events.
- Detail data sets allow retrieval of entries by a defined search item value when the detail set is related to a master set.
- Entries with duplicate search item values are chained together with pointers.
- Entries in a chain can be retrieved in sorted order according to a pre-defined sort item.
- Unlike a master data set which contains at most one search item, a detail data set can have up to 16 search items.
- A detail set can be related to up to 16 master data sets (depending on the number of search items defined).
- A detail set can be stand alone.

Purpose: To define terms used when referring to data and data sets.

Key Points: *Check the end of this module for preparation information.*

- The location of an entry in a data set relative to the beginning of the data set is called the 'relative record number.' This record number is also called a record address.
- The record address of one entry stored in another entry is called a 'pointer.' This pointer allows direct access from one entry to the other. Pointers are necessary when duplicate entries exist for a given key item. Slide 2.02 shows that four entries have a city abbreviation of SJ. These four entries would be linked together for fast access. This linkage is accomplished through the use of pointers.
- When entries are linked together via pointers a 'chain' is formed.
- The number of entries in a chain is called the 'count.' The count is an integer representing the number of entries in the chain. The count is zero if the chain is empty.

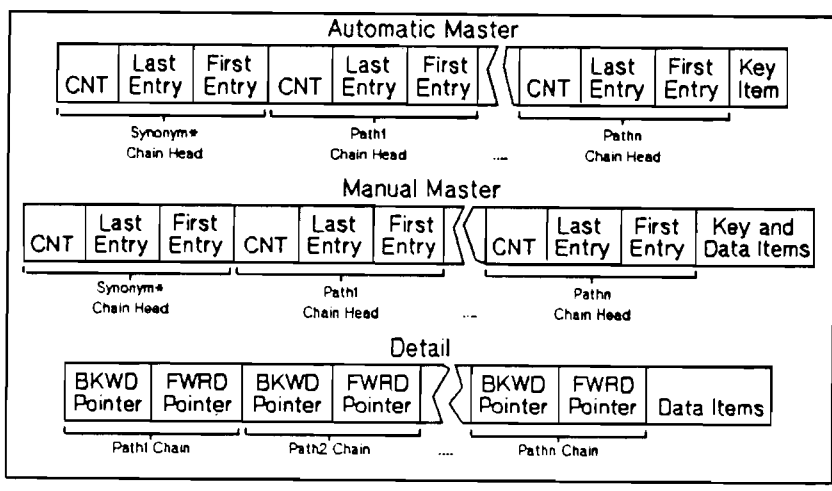
Data Set Relationships



T11 2.07

© 1985 Hewlett-Packard Company

Media Record: Pointers + Data



T11 2.08

© 1985 Hewlett-Packard Company

Pointers require 2 words. Count requires 1 word in IMAGE/3000, 2 words in TurboIMAGE. Chain head totals 5 words (IMAGE/3000), 6 words in TurboIMAGE. Detail chain pointer pairs are 4 words.

* Synonyms will be defined and discussed in Module 5. The count field for synonym chains is 1 word in both IMAGE/3000 and TurboIMAGE.

□ TurboIMAGE Structure and Terminology

Purpose: To define the relationship between master and detail data sets.

Key Points

- A master data set is related to a detail data set via pointers. These pointers are created and maintained by TurboIMAGE. The relationship is defined at data base creation.
- The relationship between master and detail sets is usually a one-to-many relationship (i.e. one master set related to many detail sets).
- The relationship between sets is called a 'path.' A path contains a chain for each unique search item value. For example, a path from a master containing city abbreviations to a detail set would contain a chain for each unique abbreviation (SJ,PA,LG,etc.). Each chain has a first and last member. Backward and forward pointers maintain the chain. The first member of a chain has a zero backward pointer and the last member of a chain has a zero forward pointer.
- As detail entries are added, their search item values are found in the master set. The master set contains information about the chain corresponding to the search item value. This information includes the location of the first and last entries in the chain and the count. This information is referred to as a 'chain head.'
- A stand alone detail set contains no pointers. Therefore, a stand alone detail set offers serial processing with low overhead and the advantage of TurboIMAGE security.
- The key item is part of the data and is not necessarily located immediately following the pointers.



Purpose: To examine the contents of a media record.

Key Points:

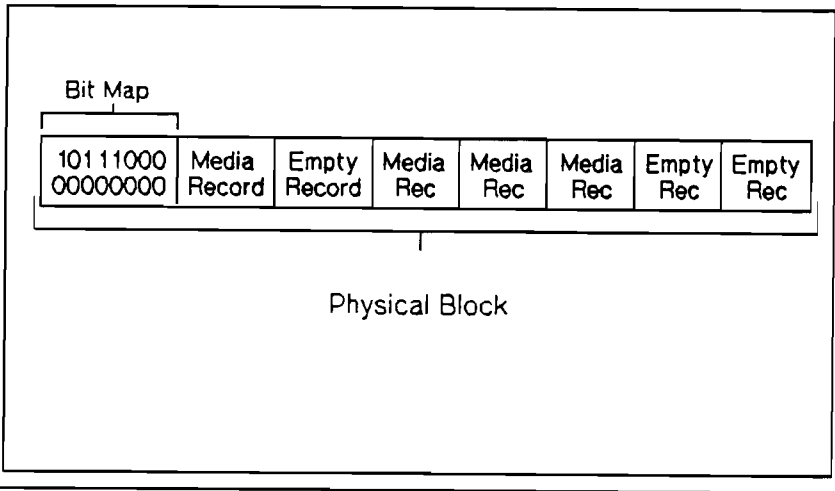
- TurboIMAGE transfers information to and from the disc in the form of a media record.
- A media record consists of data and pointers.
- For entries in detail sets, the media record consists of the entry preceded by chain pointers. The number of pointer pairs (forward and backward pointers) corresponds to the number of paths to the detail set.
- For entries in the master data sets, the key item is preceded by chain heads. The number of chain heads in the media record corresponds to the number of defined paths. A media record is the same for both automatic and manual masters with the exception that manual masters may contain data items as well as the key item.
- The first chain head in a master set media record is for synonym chains. This type of chain will be presented in detail in Module 5.

Describe in detail here the conversion from Image to Turbo - what actually happens in order to create the media records for point to point.

TurboIMAGE Structure and Terminology

Notes

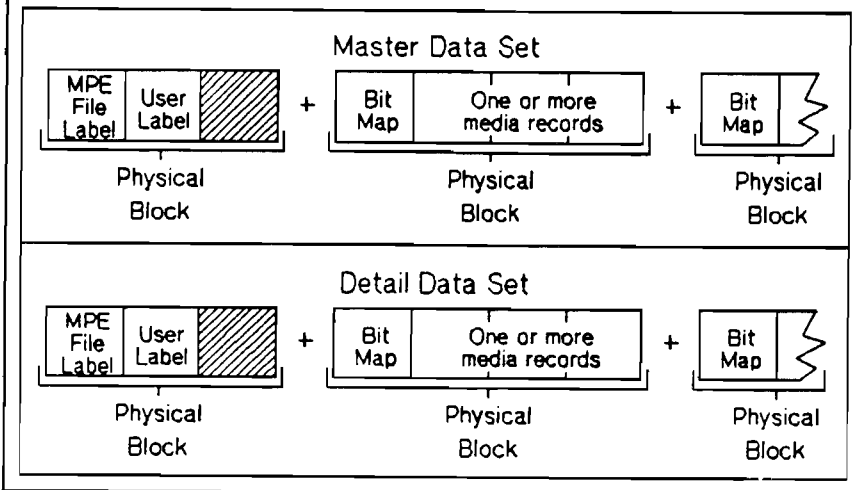
Bit Map



Bit=1 ----> media record occupies record space.
 Bit=0 ----> record space is empty.

This block contains 7 records. The bit map indicates that the 1st, 3rd, 4th, and 5th records are occupied by media records.

Data Set Layouts



Refer to *MPE Pocket Guide*, page 1-16, for the contents of the MPE file label.

□ TurboIMAGE Structure and Terminology

Purpose: To define a bit map and present the makeup of a physical block.

Key Points:

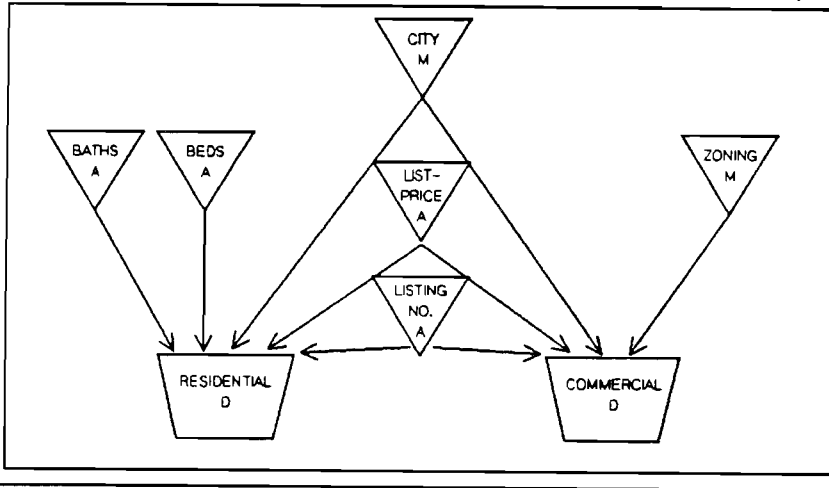
- A group of media records involved in a single disc transfer is called a 'block.'
- The first word(s) of each block contains a bit map. This bit map is used by TurboIMAGE to detect whether a particular media record of the block contains data or is empty.
- There is one bit for each record in the block. If the bit is 1, then the corresponding record contains a media record. If the bit is 0, then the corresponding record is empty.
- The maximum size of a block is controlled indirectly by the size of the items in a data entry and by a parameter (BLOCKMAX) specified at data base creation

Purpose: To present the physical structure of master and detail sets in terms of physical blocks.

Key Points:

- The first physical block contains the MPE file label and a user label containing information needed for dynamic storage allocation and de-allocation. The user label in a master set contains a free space counter, the data set capacity, and the data set end-of-file pointer. The user label in a detail set contains a free space counter, the data set end-of-file pointer, and a delete chain pointer which holds the address of the entry most recently deleted.
- When a detail data set is created, the end-of-file pointer and the delete chain pointer are both zero.
- The other physical blocks contain a bit map plus one or more media records.

The REALTY Data Base



☐ TurboIMAGE Structure and Terminology

Purpose: To present the REALTY data base structure.

Key Points:

- The REALTY data base has two detail data sets, residential and commercial.
- The only manual master data sets are city (names and abbreviations) and zoning codes. All other master data sets are automatic.
- City, list-price, and listing number have paths to two detail data sets.
- Residential detail set has five search item paths and commercial has four.

□ Activity 2-2 Lab: Introduction to the REALTY Data Base

Purpose: To gain familiarity with the REALTY data base and the program that will be used in several labs throughout the course.

Directions: *:FILE REALTY = REALTY.LABS just for this exercise*
After you will each have your own copies in your own groups

1. Run the program.
 - A. Run LABDEMO.LABS
 - B. Respond to the question "Which lab are you running?" with 2.
 - C. Use MANAGER (upper case) as the password.
2. Add an entry to the REALTY data base.
 - A. Indicate on the first screen that you want to work with the residential listing information. (NOTE: in screen mode use the ENTER key not the RETURN key.)
 - B. Enter the appropriate function to add an entry. Use the TAB key to move to the next field. (SHIFT TAB can be used to move to a previous field.)
 - C. Fill all the fields. Use SJ for city-abbreviation; list price should be in thousands of dollars; number of baths should be in the form x.xx (where .xx must be .00, .25, or .50.) Note that the LISTING-NR will be automatically provided.
3. Find and list all houses in San Jose. Note the use of 'E' to stop reading a chain.
 - A. Enter the appropriate function to find/list entries.
 - B. Specify SJ as city-abbreviation.
4. Find and list all houses with 3 bedrooms.
5. Change the phone number in the listing you entered.
 - A. Find the entry you added.
 - B. Update the entry by using the appropriate function and entering the new phone number.
 - C. Enter 'E' to stop chained read and find the entry again to check for a successful update.
6. If you have time, browse through the application.

Module 2-9 Instructor Notes

Activity 2-2 Lab: Introduction to the REALTY Data Base

Time: 20 minutes

Purpose: To gain familiarity with the REALTY data base and the program that will be used in several labs throughout the course.

Directions:

1. Create and load the REALTY data base in the LABS group.
 - Run DBSCHEMA, using LAB2SCHM.LABS for the DBSTEXT file.
 - Run DBUTIL,CREATE.
 - Run LOADDDB.LABS to load clean data into the data base.
 - Release REALTY.LABS.
2. Discuss the directions with the students before they begin working.
 - Emphasize the form required for entering number of baths.
 - List-price is in thousands of dollars, therefore, at most 4 digits need be entered.
3. Circulate the room to be sure that each student is able to execute the lab activities.
4. If some students finish early, suggest they try other activities with the application; for example, try to enter a property with a city abbreviation that does not exist in the master data set (such as XX).

check

Components of TurboIMAGE

- DDL _____

- DML _____

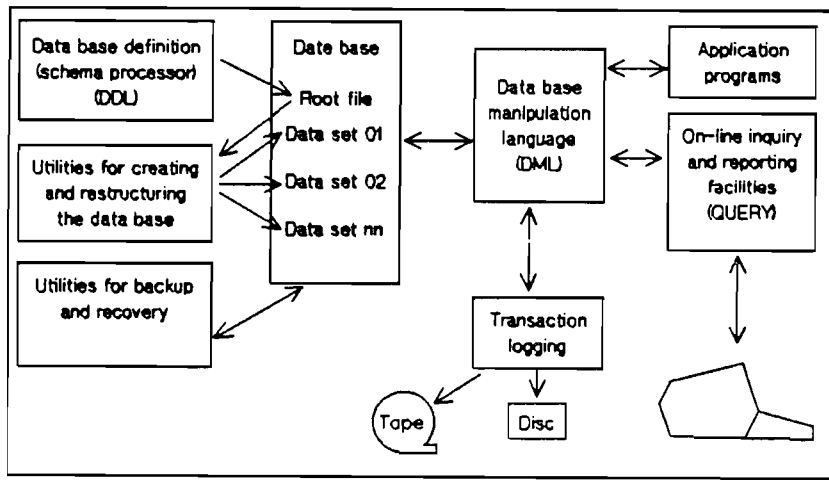
- Utilities _____

- QUERY/3000 _____

DDL - Data Definition Language

DML - Data Manipulation Language

TurboIMAGE Overview



□ TurboIMAGE Structure and Terminology

Purpose: To present the basic components of TurboIMAGE.

Key Points: *Check the end of this module for preparation information.*

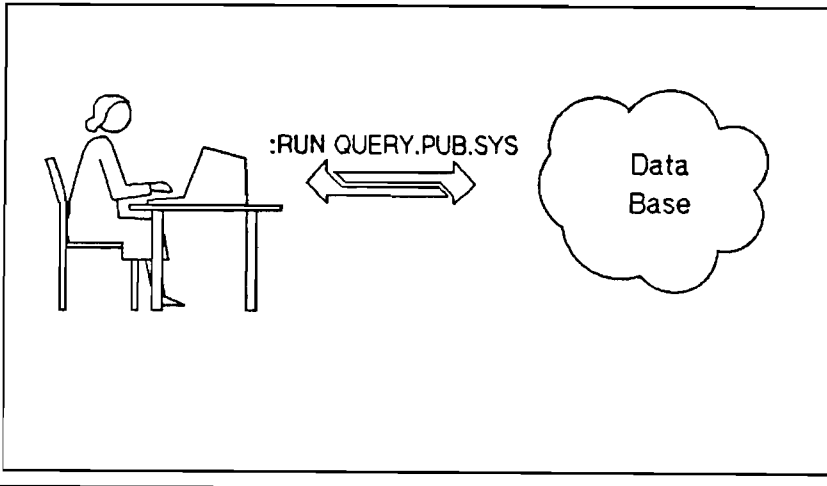
- Data Definition Language is used to define all aspects of the data base including data items, relationships, security, and capacities.
- Data Manipulation Language provides the interface between application programs and the data base. The DML consists of a set of library routines (procedures) that can be called from application programs.
- Utility programs create and maintain TurboIMAGE data bases. Utilities also exist for back-up, recovery, and migration from IMAGE/3000 to TurboIMAGE.
- QUERY/3000 is an on-line inquiry and reporting facility. QUERY/3000 allows display of the data base structure as well as inquiries and modifications of the data in the data base.

Purpose: To present an overview of the interconnections of the components of TurboIMAGE.

Key Points:

- The data base definition is performed by the schema processor.
- TurboIMAGE utilities are used to create and restructure the data base.
- The data base manipulation language is the interface between the data base and the application programs and/or QUERY.
- TurboIMAGE utilities are also used for transaction logging, back-up and recovery.

QUERY/3000



T11 2.14

© 1985 Hewlett-Packard Company

Getting Started with QUERY/3000

```
HEL P

:RUN QUERY.PUB.SYS
QUERY/3000 READY
>HELP (H)
...
>EXIT
```

1. Invoke QUERY
2. Enter HELP
3. Read the list of available QUERY commands.
4. Enter EXIT

T11 2.15

© 1985 Hewlett-Packard Company

Module 2-11 Instructor Notes: Slides 2.14/2.15

□ TurboIMAGE Structure and Terminology

Purpose: To introduce QUERY/3000.

Key Points: *Check the end of this module for preparation information.*

- QUERY/3000 is a powerful program development and debugging tool. It can be used to build test data as well as to interrogate the results of program tests.
- QUERY/3000 is also useful for low volume data base modification (additions, deletions, updates).
- QUERY/3000 is used for report generation.
- To use QUERY/3000, the relationships of the data base elements must be known. QUERY finds the data and performs the operations in response to commands using specified data set and data item names.
- QUERY/3000 is a command driven tool.

Purpose: To introduce the HELP command in QUERY/3000.

*On-line activities whenever possible
2-17 - 2-22. Get ready.*

Key Points:

- QUERY/3000 is invoked by typing - :RUN QUERY.PUB.SYS.
- The HELP command defines and describes QUERY commands. By typing HELP alone, a list of all the QUERY commands appear. To receive help on a specific QUERY command, type HELP *command name*.
- The QUERY command EXIT will end the subsystem and return control to MPE.

QUERY Special Characters

>	Prompt for QUERY command	Y ^c	(CNTL Y) - Terminates the QUERY command, prints <CONTROL Y>, and prompts for a new command
>>	Prompt for additional information		
&	Line continuation (Must be followed directly by C/R)	S ^c	(CNTL S) - Suspends output to terminal
		Q ^c	(CNTL Q) - Resumes output to terminal

□ TurboIMAGE Structure and Terminology

Purpose: To define special characters used in QUERY/3000.

Key Points:

- The symbol > is used to prompt for a QUERY command.
- >> indicates additional information is required.
- & is used to indicate that a command will be continued on the next line. (An example of this appears on slide 2.18 (the FIND command).
- Control-Y can be used to terminate a QUERY command. A prompt for a new command will appear.
- Control-S suspends output to the screen. This allows the user to read the information one screen at a time.
- Control-Q resumes output to the screen.

DEFINE and FORM Commands

> DEFINE (DEF)

DATA-BASE=>>
PASSWORD=>>
MODE=>>
DATA-SETS=>>
PROC-FILE=>>
OUTPUT=TERM
OUTPUT=>>

> FORM (FO)

FORM
FO SETS
FO ITEMS
FO PATHS
FO RESIDENTIAL

1. Enter DEFINE and the following parameters:
 - REALTY.LABS
 - MANAGER
 - 1
 - <CR>
 - <CR>
 - OUTPUT=TERM
 - <CR>
2. Try each of the FORM command examples.

□ TurboIMAGE Structure and Terminology

Purpose: To present the DEFINE and FORM commands.

Key Points:

- The DEFINE command identifies a particular data base and its attributes to QUERY.
- DEFINE may be used to initiate a QUERY session or to change specifications in the environment. When the command is entered, QUERY lists the current state of each parameter and prompts for any changes. A carriage return will maintain the current value. The only parameter that is initially set by QUERY is OUTPUT=TERM. This can be changed to OUTPUT=LP to redirect the output to device class LP. The formal file designator is QSLIST.
- The data base name, password, and mode must be entered.
- If carriage return is pressed for the data set parameter, all data sets will be searched for a particular data item. The user will be prompted for the set name if the item appears in multiple data sets.
- The PROC-FILE parameter is used for specifying a procedure file. This file may contain frequently used or complex commands.
- An alternative to the DEFINE command is to specify each parameter by an individual command. These commands are:

DATA-BASE=*data base name* or B=*data base name*
PASSWORD=*password*
MODE=*mode number* or M=*mode number*
DATA-SETS=*data set list* or S=*data set list*
PROC-FILE=*filename*
OUTPUT=TERM or LP

- The FORM command lists information about the data base currently being accessed. The command entered alone will list all information about the data base. More specific information can be accessed by specifying SETS, ITEMS, PATHS, or particular data set names or data item names.

TurboIMAGE Structure and Terminology

Notes

FIND Command

>FIND (F)

FIND ALL LIST-PRICE

ENTRIES QUALIFIED

FIND LIST-PRICE ILT 200

ENTRIES QUALIFIED

FIND LIST-PRICE ILT 200 AND &
NUMBER-BEDS EQ 03 AND &
NUMBER-BATHS >=02 AND &
CITY-ABBR IS SJ

ENTRIES QUALIFIED

1. Refer to *MPE Pocket Guide* Section IV for syntax and list of available operators, or type HELP FIND.
2. Enter DATA-SETS= RESIDENTIAL
3. Try each FIND example and record the number of entries.

REPORT and LIST Commands

>REPORT (R)

REPORT ALL

REPORT ALL,X

>LIST (L)

LIST RESIDENTIAL

LIST RESIDENTIAL FOR LIST-PRICE GT 200

1. Re-issue the various FIND commands from the previous slide.
2. After each example, report the results both with item names and without.
3. Try the LIST command examples.

□ TurboIMAGE Structure and Terminology

Purpose: To define and give examples of the FIND command.

Key Points:

- The FIND command locates entries in the data base according to data item values in the entries.
- Up to 50 logical relationships can be specified in one command.
- More than one value can be specified for comparison with a data item.
- The number of entries satisfying the logical relationship specified is reported.
- The FIND command consists of a data item name, a relational operator, and one or more values separated by commas.
- After the command is entered, QUERY may type the message: USING SERIAL READ. This means the data set must be searched in serial fashion without benefit of master set indexing. This search can (in some cases) consume a great deal of time. The search may be interrupted by pressing Control-Y. QUERY will display a search statistic (number of specified records found) and prompt the user for a decision to continue or abort the search.
- Once entries have been located via the FIND command they are available for listing, updating, or deleting. These entries will be available until QUERY terminates or another FIND command is entered.

Purpose: To present syntax and examples of the REPORT and LIST commands.

Key Points:

- The REPORT command is an extension of the FIND command in that it prints a report of the data entries located by the last FIND command.
- REPORT output can be directed to any desired output device through MPE :FILE commands and the QUERY OUTPUT command. The formal file designator is QSLIST.
- The command is used in two forms:
 - (1) REPORT ALL - returns the value of each item located by the FIND command. REPORT ALL, *any ASCII character* - returns the value of each item located by the FIND command with no accompanying item names. This is useful for creating output to be used on special forms such as labels.
 - (2) REPORT *report statements* END - returns the value of each item located by the FIND command in a specified format.
- The LIST command prints complete or partial entries from a single data set with automatic formatting and headings.
- The LIST command combines the functions of selectively locating and reporting on entries. It is totally independent of the FIND and REPORT commands.

TurboIMAGE Structure and Terminology

Notes

ADD Command

>ADD (AD)

```
ADD RESIDENTIAL
LISTING-NR      ==>>9X
CITY-ABBR      ==>>SJ
LIST-PRICE     ==>>____(K)
NUMBER-BEDS    ==>>__
NUMBER-BATHS   ==>>____
CURRENT-OWNER  ==>>_____
OWNERS-PHONENR ==>>_____
STREET ADDRESS ==>>_____
ZIP-CODE       ==>>_____
FORMAL-DINING-RM ==>>Y or N
SOLD-FLAG      ==>>Y or N
SQUARE FEET    ==>>_____
```

1. Add a sample entry using the LISTING-NR provided by the instructor and CITY-ABBR= SJ
2. Enter // when prompted for another entry.

Note: The number of dashes on the slide indicate the number of characters for each item.

TurboIMAGE Structure and Terminology

Purpose: To present syntax and examples of the ADD command.

Key Points: *Check the end of this module for preparation information.*

- The ADD command adds data entries to a manual master or a detail data set.
- A data set name must be provided and it must be the name of either a manual master or detail. Write access is required to complete the function.
- QUERY prompts for data item values. If a value contains invalid characters, QUERY issues a message and reprompts for the same data item name.
- Search and sort items must have values supplied. A carriage return can be entered for other data items to produce a null value for the item.
- When all data items in an entry have been responded to (null or otherwise), QUERY prompts for another entry. To terminate the ADD function, either enter // followed by RETURN or Control-Y. These termination responses can be entered at any time. However, if either is entered before responding to all data items in an entry, the entire entry will be discarded.

TurboIMAGE Structure and Terminology

Notes

REPLACE and DELETE Commands

>REPLACE (REPL)

```
REPLACE,  
NUMBER-BEDS='  ' ;  
ZIP-CODE='  ' ;  
SOLD-FLAG='  ' ;  
END
```

>DELETE (DEL)

1. Find and report the entry with LISTING-NR=9x.
2. Make the suggested updates. What happens?
3. Redo the example and update only ZIP-CODE and SOLD-FLAG.
4. Enter REPORT ALL and note the changes.
5. Now enter DELETE and answer YES (be sure you delete only the entry you added.)
6. Do a FIND LISTING-NR=9x to check if the entry has been deleted.

CAUTION: The DELETE command will delete ALL the entries retrieved by the most recent FIND command.

TurboIMAGE Structure and Terminology

Purpose: To present syntax and examples of the REPLACE and DELETE commands.

Key Points: *Check the end of this module for preparation information.*

- The command REPLACE will update values of the data items.
- The REPLACE command is an extension of the FIND command in that it operates on data entries selected by the previous FIND command.
- The REPLACE command requires all data entries to be from the same set.
- QUERY does not allow updating of search or sort items without totally deleting an entry and then adding it back in with the new values.
- The REPLACE command consists of a series of data item name/data item value pairs separated by semicolons.
- The DELETE command deletes data entries from a data set.
- The DELETE command is an extension of the FIND command in that it deletes those entries selected by the previous FIND.
- The DELETE command requires all data entries to be from the same set.
- When the DELETE command is issued, QUERY responds with a message: DELETE ALL RETRIEVED ENTRIES(YES OR NO)?. If yes is entered the entries selected will be deleted.
- QUERY disallows a master entry from being deleted if its search item value still exists in linked detail data sets.

TurboIMAGE Structure and Terminology

Notes

OUTPUT Command



>OUTPUT(OUT)

OUTPUT = { TERM }
 { LP }

1. Set OUTPUT=LP
2. Type HELP to receive a complete listing of QUERY/3000 commands.
3. Type EXIT
4. Pick up listing from line printer.

The formal file designator for QUERY output is QSLIST.

□ TurboIMAGE Structure and Terminology

Purpose: To present syntax and examples of the OUTPUT command.

Key Points:

- The OUTPUT command selects the output device for HELP, FORM, LIST, and REPORT commands.
- The default output device is \$STDLIST unless the OUTPUT command is used to change it.
- The parameter LP indicates that output is to be sent to the file designated QSLIST. QSLIST is equated to the MPE device class LP unless a :FILE equation is issued (:FILE QSLIST;DEV=*device type*.)
- QUERY will not output a record longer than 136 bytes even though the maximum record length for the device may exceed this limit.
- To change output device while in QUERY session mode, use the BREAK key, issue the :FILE command to redirect QSLIST, and return to QUERY with :RESUME.
- When output is directed to device class LP, the spoolfile is not closed until the EXIT command is issued or the output is reset to TERM. Therefore, emphasize to the students that they must type EXIT before they pick up the output for this activity.

Module 2-18

□ Activity 2-3 Lab: Using QUERY/3000

Purpose: To formulate queries to the REALTY data base using QUERY/3000.

Directions:

Step 1. Run QUERY.PUB.SYS.

Step 2. Define the same environment as on slide 2.17.

Step 3. Use QUERY to complete the following tasks:

1. How many data sets are there in the data base? _____
2. Give the name of one data set of each type.

3. How many data items are defined? _____
4. How many paths exist? _____
5. Name a sort item. _____ Which data set is it in? _____
6. List all the COMMERCIAL listings.
7. Add an entry to the COMMERCIAL data set with LISTING-NR=9x (number assigned by instructor), ZONING-CODE=R3, and CITY-ABBR=LA.
8. Find all entries in the COMMERCIAL data set with prices between \$150K and \$200K that are unoccupied and are larger than 1200 square feet (enter 00001200 for the SQUARE-FEET value.)
How many entries qualify? _____
9. Find the entry you added, change one item in the entry, then list the entry on the line printer.
10. Switch the output back to TERM mode.
11. Enter REPORT ALL to verify you found the correct entry (only the entry you created should be displayed.) Now delete this entry.
12. Exit QUERY.

Solution: After you have attempted the lab on your own, the solutions can be found in Appendix A.

Module 2-18 Instructor Notes

Activity 2-3 Lab: Using QUERY/3000

Time: 30 minutes

Purpose: To formulate queries to the REALTY data base using QUERY/3000.

Notes:

This lab requires the student to use the QUERY/3000 commands that have been presented in this module. The students will access the REALTY data base in the LABS group. The COMMERCIAL data set will be accessed instead of the RESIDENTIAL data set. Instruct the students to make note of the data types of each of the data item values in a COMMERCIAL entry before they attempt to add an entry. Remind students that entries can be added only if the search item value already exists in the appropriate manual master.

When the students try to update values in their new entry, remind them that search and sort items cannot be updated.

Instruct students to use the same listing number for number 7 as they used in the example on slide 2.20.

Solutions:

1. & 2. By entering FO SETS, a list of the data sets will appear. The type, item count, capacity, entry count, entry length, and blocking factor is listed for each set. There are 8 data sets.

The automatic masters are: LISTNR-MASTER, LIST-PRICE-MSTR, BATH-MASTER, BEDS-MASTER.

The manual masters are: ZONING-MASTER, CITY-MASTER.

The detail data sets are: RESIDENTIAL, COMMERCIAL.
3. Enter FO ITEMS to display a list of the data item names and the data types. There are 17 data items defined.
4. Enter FO PATHS to display path identifying information. There are 9 paths defined.
5. SQUARE-FEET is a sort item in the RESIDENTIAL set and the COMMERCIAL set. LISTING-NR, CITY-ABBR, LIST-PRICE, NUMBER-BEDS, and NUMBER-BATHS are search items in the RESIDENTIAL detail set. LISTING-NR, ZONING-CODE, LIST-PRICE, and CITY-ABBR are search items in the COMMERCIAL detail set.
6. >LIST COMMERCIAL

End Monday

(Continued on next page.)

Module 2-19 Instructor Notes

□ Activity 2-3 Lab: Using QUERY/3000 (cont'd)

7. >ADD COMMERCIAL
LISTING-NR =>>99
ZONING-CODE =>>R3
LIST-PRICE =>>_____
CITY-ABBR =>>LA
STREET-ADDRESS =>>_____
ZIP-CODE =>>_____
EXISTING-STRUCT =>> Y or N
OCCUPIED =>> Y or N
SOLD-FLAG =>> Y or N
SQUARE-FEET =>>_____
8. >FIND LIST-PRICE IB 150,200 AND &
>>OCCUPIED IS N AND &
>>SQUARE-FEET >00001200

1 ENTRY QUALIFIES
9. >FIND LISTING-NR=99
>REPLACE,
>> item name = " ";
>>END
>OUTPUT=LP
>REPORT ALL
10. >OUTPUT=TERM
11. >REPORT ALL
>DELETE
>>YES
12. >EXIT

Module 2--20 Instructor Notes

Preparation Material: Slides 2.04/2.05

Teaching Tips:

Use this slide to cover the basic concepts of a master data set. Have the students enter the basic characteristics on the slide in the student workbook. You can write the key points on the chalk board, a flip chart, or directly on the overhead slide.

Many students may not have previous experience with the concept of a hashing algorithm. A simple example may suffice to give them a conceptual idea of what hashing means.

Example: A hashing algorithm is designed to provide a record number that will distribute data evenly within the known capacity of the file.

Suppose a file has a capacity of 1000 records and you want to store purchase orders with numbers ranging from 2000 to 3000. A relative record number is assigned to the record by determining the remainder after the p.o. number is divided by 1000. This is called modulo arithmetic. If the p.o. number range is 2000 to 3000, no duplicate record numbers will occur. However, if the range increases, duplicates will occur. The concept of secondaries is covered in detail in Module 5.

Emphasize that this is a very simplistic example and that further discussion can be found in the *TurboIMAGE Data Base Management System Reference Manual*, Section 10: Primary Address Calculation.



Preparation:

Detail data sets record information about related events; for example, information about a particular property listing, information about all sales to the same account, information about an employee, etc.

Detail data sets allow retrieval of entries by a defined search item value such as a property listing number, an account number, an employee's social security number, etc. The values of a particular search item need not be unique. For example, in the REALTY data base, city abbreviation is a search item value and many entries have the city abbreviation SJ (San Jose). These entries with duplicate search item values are chained together with pointers.

Entries in a chain can be retrieved in sorted order according to a sort item. For example, the chain of entries with a particular city abbreviation could be retrieved in alphabetical order according to the owner's name. Sorted chains result in a great deal of overhead. During the design of a data base, a decision should be made as to whether sorting must be done while the data is being entered, or if a sorting routine within an application is sufficient.

A detail set can have up to 16 search items. For example, a detail set containing information about property listings may have listing number, city abbreviation, and number of bedrooms defined as search items. A detail set containing information about employees may have employee number, department number, and salary defined as search items.

The added complexity of multiple search items and sort items requires increased overhead. These issues are important ones to consider at the design phase of implementing a data base.

Module 2-21 Instructor Notes

Preparation Material: Slides 2.06, 2.12

Teaching Tips:

Use the example shown on slide 2.02 when defining the terms on this slide. Instruct students to take notes directly on the slide in the student workbook.

Teaching Tips:

As you explain each component, write, on the overhead slide, the basic definition. Instruct the students to record this information in the student workbook.

Point out that these components are fundamental to ANY data base management system.

Each of these components will be presented in detail in later modules as follows:

- DDL - Module 4
- DML - Module 5
- Utilities - Module 4 (DBSCHEMA, DBUTIL) and Module 8
- QUERY/3000 - Module 2

Module 2-22 Instructor Notes

Preparation Material: slides 2.14, 2.20

Teaching Tips:

The slides (2.15, 2.17-2.22) that follow give the student the opportunity to access and modify the REALTY data base using QUERY/3000. Students are given directions in the notes on each slide. You should present this information to the students by:

1. Introducing the command(s) on the slide.
2. Discussing syntax and uses of the command.
3. Having students step through the procedure in the student notes.
4. Discussing any problems or questions concerning the activity.

Allow the students 5-10 minutes to try each activity in the workbook. Ask the students to work ONLY on the command being discussed and to avoid working ahead in the workbook.

At the end of this group of slides, the students will do a lab activity involving the formulation of additional queries to the data base.

Teaching Tips:

The REALTY data base has 40 entries in the RESIDENTIAL data set plus the entries added by the students in lab 2-2. To avoid duplicate listing numbers, assign each student a number, starting with 90, to use in this example.

Module 2-23 Instructor Notes

Preparation Material: Slide 2.21

Teaching Tips:

The first step in the REPLACE example will illustrate that search items cannot be updated. Point out that sort items also cannot be updated.

Step 3 instructs the student to redo the REPLACE command without using a search item.

Module 3 Instructor Notes

Review of Data Base Concepts

Overview of Module 3

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Activity Quiz: Basic Data Base Concepts (20 minutes)

Purpose: To test basic concepts of Module 1 and Module 2 via an on-line interactive session.

Lesson 1. DBMS Implementation (40 minutes)

Slides: 3.01 - How do the Pieces Fit Together? *
 3.02 - Implementation Overview *

Activity Worksession: Data Base Design - Part One

Purpose: To design a data base.



Module 3-1

Review of Data Base Concepts

Goal: To test your understanding of the concepts presented thus far via an on-line interactive quiz, and to present an overview of the remainder of the course.

Objectives:

Upon completion of this module you will be able to:

- identify basic concepts of data base management and TurboIMAGE structure.
- identify the basic components of implementing a data base management system.
- begin the design phase of a data base implementation.

Module 3-2

□ Activity 3-1 Quiz: Basic Data Base Concepts

Purpose: To test basic concepts of Module 1 and Module 2.

Notes: This activity will test your understanding of the information presented in the first two modules. The activity is an on-line interactive quiz. You will be given the opportunity to request a listing of the questions and answers at the end of the session.

Directions:

1. Type :MOD3REVIEW to run the interactive session. Each student should run the quiz individually.
2. Instructions will be given at the beginning of the session.
3. Consult with the instructor if any of the concepts covered in the session requires further explanation.

Module 3-2 Instructor Notes

□ Activity 3-1 Quiz: Basic Data Base Concepts

Purpose: To test basic concepts of Module 1 and Module 2.

Time: 20 minutes

Directions:

1. Be sure to set and release the UDC file ACCUDC.PUB at the account level. The students will initiate this session by typing the UDC :MOD3REVIEW.
2. The questions and answers are in the file MOD3.SOLUTION. This file will be sent to device class LP upon request by the student at the end of the session. Only one copy per session will be listed. Therefore, encourage students to run the session individually. A copy of the file is listed below.

Solutions:

MODULE 3 - REVIEW OF DATA MANAGEMENT CONCEPTS AND TURBOIMAGE CONCEPTS

Directions: Each of the following items is a description of a concept covered in Module 1 or Module 2. Identify each concept.

1. A collection of software routines that serves as an interface between programs and a data base.
ANSWER: Data Base Management System (DBMS)
2. The TurboIMAGE data base model.
ANSWER: Network (2-level)
3. The address of a record used by a direct access method.
ANSWER: Relative record number
4. The letter preceding the version number of TurboIMAGE.
ANSWER: C
5. The smallest unit of data in the TurboIMAGE data base structure.
ANSWER: data item
6. A collection of related data items.
ANSWER: data entry
7. The part of a media record that indicates where the next entry (or previous entry) is located.
ANSWER: pointer
8. Entries with the same search item linked together.
ANSWER: chain

Module 3-2 Instructor Notes

Activity 3-1 Quiz: Basic Data Base Concepts (cont'd)

9. A data set that serves as an index to related data sets.

ANSWER: master

10. A master data set which contains key values and possibly data items and can exist as a stand alone set.

ANSWER: manual master

11. An address calculation algorithm.

ANSWER: hashing algorithm or hash function

12. The pre-defined relationship between master and detail data sets.

ANSWER: path

13. A unit of information used to transfer data and pointers to and from the disc.

ANSWER: media record

14. The first word(s) of a block of media records.

ANSWER: bit map

15. A language used to define all aspects of a data base including items, relationships, security, and capacities.

ANSWER: Data Definition Language (DDL)

16. TurboIMAGE's on-line inquiry and reporting facility.

ANSWER: QUERY/3000

17. A QUERY/3000 command that lists information about the sets, items, and paths of a data base.

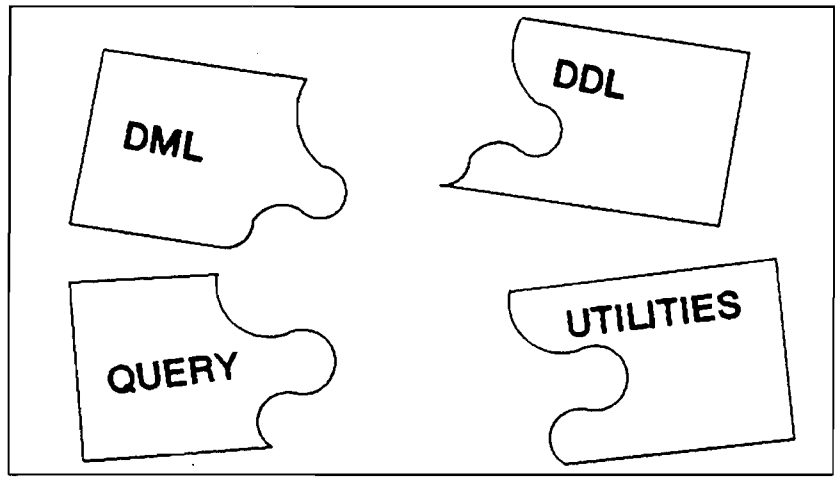
ANSWER: FORM

19. A QUERY/3000 command that combines the functions of locating and reporting entries and provides automatic formatting.

ANSWER: LIST



How Do The Pieces Fit Together?



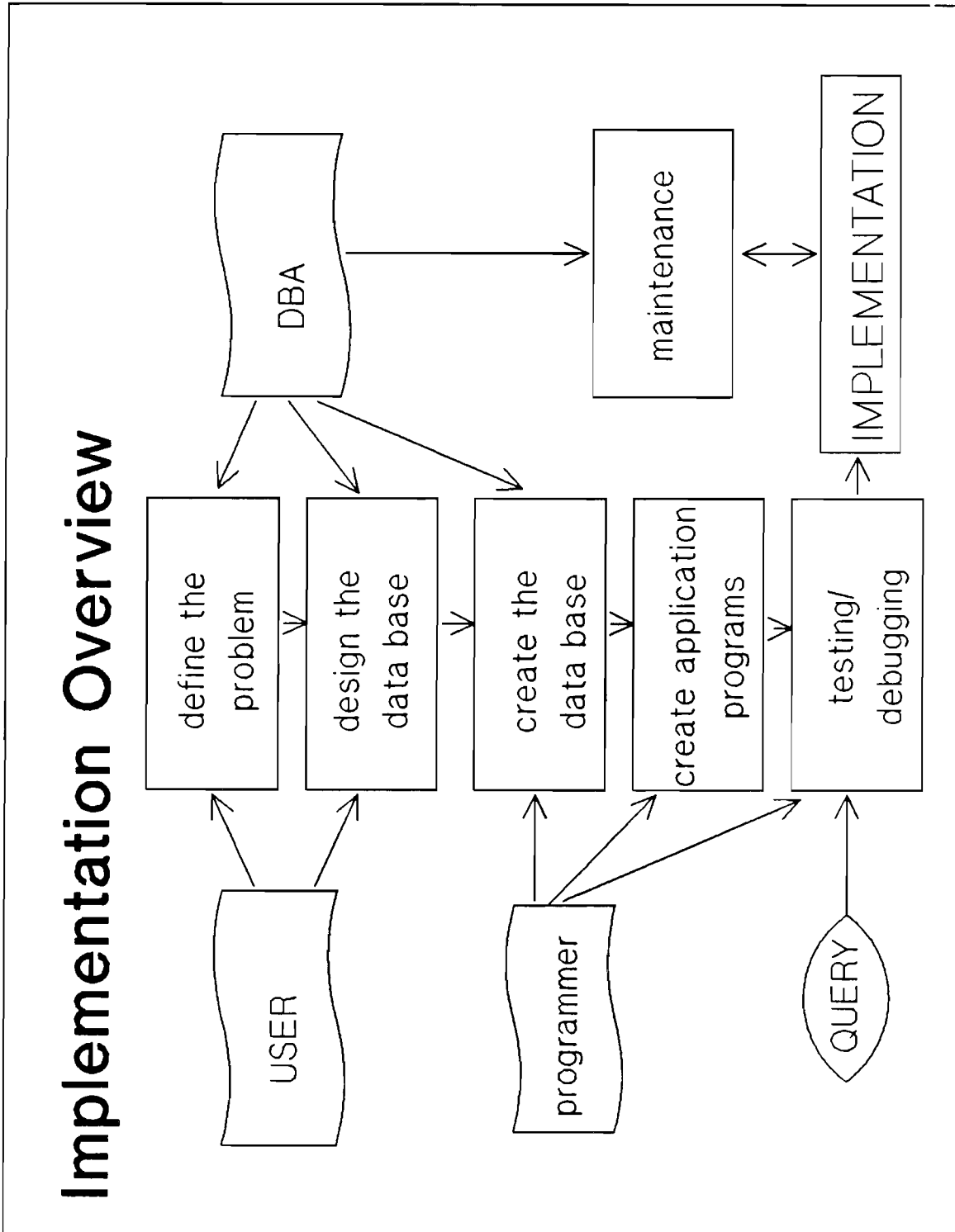
Module 3-3 Instructor Notes: Slide 3.01

Review of Data Base Concepts

Purpose: To review basic data base concepts and introduce upcoming topics.

Key Points: *Check the end of this module for preparation information.*

- A data base management system consists of a DDL(data definition language), a DML(data manipulation language), a set of utilities for creating and restructuring the data base, and an on-line inquiry and reporting facility, QUERY/3000.
- The TurboIMAGE DBMS is a two-level network structure. Information is represented by individual data items grouped in a logical manner to form data entries. A collection of data entries makes up a data set. Collections of data sets form a data base.
- QUERY/3000 is used to access and modify a data base. It is a useful programming tool for data base applications.



□ Review of Data Base Concepts

Purpose: To present an overview of the steps in implementing a data base management system.

Key Points: *Check the end of this module for preparation information.*

- Defining the problem is the most time consuming step. Management commitment and user involvement are essential at this step. Without either one of these, there is a good chance the application will be unsuccessful.
- The design team must decide what data is required by all the application projects that will share the data base.
- The design of the data base is described using the TurboIMAGE data base description language. The external description is called a schema.
- The external description of the data base is processed by a TurboIMAGE utility to produce an internal description called a root file. Another utility is then used to create the actual data base files.
- Once the data base is created the application development begins.
- The testing and debugging phase of an application can be facilitated by QUERY/3000.
- Once the implementation is in place, ongoing maintenance procedures must exist. These procedures should include making backup copies of the data base, use of utilities to recover and restructure the data base, and use of tools for performance tuning.

Module 3-5

□ Activity 3-2 Worksession: Data Base Design – Part One

Purpose: To design a data base.

Notes: This is part one of a three part design worksession. This activity will be accomplished in small groups of students. Part one defines an information management problem that is to be solved by designing an appropriate TurboIMAGE data base.

As the course progresses and you learn more about TurboIMAGE data bases, you will be able to enhance your design. Part two of this activity will be a group worksession to finalize the design. Part three will consist of presentations and discussions of the various designs.

There is no ONE correct solution to this activity. Data base design involves many trade-offs between data flexibility and system overhead.

Directions:

1. Your instructor will assign you to a group of 4 or 5 students.
2. Read the specifications of the problem listed below.
3. Begin the initial phase of the design process by discussing, with your group, a possible external structure of the data base. This discussion should address the following questions:
 - how many data sets should there be?
 - which data sets are master sets and which are detail sets?
 - should the master sets be manual or automatic?
 - how many paths should be defined?
 - which items should be search items?
 - which items, if any, should be sort items?
4. You will be able to provide final details to your design by the last day of the course. Your solution will include a complete block diagram for the specifications below. Keep this design problem in mind as the course addresses the REALTY case study implementation.

Specifications:

Parker Manufacturing Co. needs an order entry system that accesses a TurboIMAGE data base.

Data Items:	customer name	part number
	customer street address	part description
	customer city name	part unit price
	customer state	quantity of part ordered
	customer zip-code	current inventory of part
	customer account number	inventory location of part
	purchase order number	supplier name
	date of order	supplier address

All of these items should be included in your design. However, you may add additional data items if you feel they are necessary to provide a complete solution to the problem.

(Continued on next page.)

Module 3-6

Activity 3-2 Worksession: Data Base Design - Part One (cont'd)

Functions:

- locate an order by purchase order number.
- list all orders placed by a particular customer.
- retrieve information necessary to bill the customer.
- identify part numbers and supplier information for parts whose inventory is low.
- locate all suppliers of a given part.
- locate all parts supplied by a given supplier.

You may define additional functions if you feel they are necessary to provide a complete solution to the problem.

Solutions: One solution will be provided by each group on day 5.

Module 3-6 Instructor Notes

Activity 3-2 Worksession: Data Base Design - Part One

Purpose: To design a data base.

Time: 30 minutes

Notes: This is part one of a three part design worksession. This activity will be accomplished in small groups of students. Part one defines an information management problem that is to be solved by designing an appropriate TurboIMAGE data base. Part two will be a worksession to finalize the design. Part three will consist of presentations of the designs by one member of each group. Parts two and three will be done at the end of Module 9.

Directions:

1. Divide the class into groups of 4 or 5 students. Assign lab partners to different groups. This will expose students to a wider range of new ideas.
2. Allow the groups 30 minutes to discuss the specifications of the problem. A 60 minute worksession will be provided at the end of Module 9 to complete the design.
3. Explain that this activity is ongoing throughout the course, culminating in presentations of the solutions on the final day. The final solution will be a block diagram and a schema.
4. Emphasize strongly that no one correct solution exists. The intent of the activity is to share and discuss various solutions to the problem.

Module 3-7 Instructor Notes

Preparation Material: Slides 3.01/3.02

Teaching Tips:

Call on individual students to describe, in his/her own words, each of the following:

- data base
- DBMS
- DDL
- DML
- utilities
- QUERY
- TurboIMAGE network structure
- data item
- data entry
- data set

Teaching Tips:

This slide provides an overview of the rest of the course. As you discuss this slide, write the main components of implementing a DBMS on a flip chart. Post this chart in the classroom and refer back to it throughout the course. This will remind the students throughout the week of the BIG PICTURE as opposed to the many details involved in data base implementation.

Include the following information on the flip chart:

1. Define the problem.
 - Activity 3-2
2. Design the data base.
 - Module 4, 9
3. Create the data base.
 - Module 4
4. Create application programs.
 - Modules 5,6,7
5. Maintenance
 - Modules 8,9,10

Module 4 Instructor Notes

Data Definition Language

Overview of Module 4

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. The Schema (1 hour 20 minutes)

Slides: 4.01 - Overview of Data Base Creation
4.02 - Data Definition Language
4.03 - Data Base Name
4.04 - Password Part
4.05 - Item Part *

Text Page: Item Part Type Designators and Programming Languages

Slides: 4.06 - Set Part for Masters *
4.07 - Set Part for Details *
4.08 - End Statement
4.09 - IMAGE/3000 and TurboIMAGE Limits

Activity: Lab: Data Base Creation - Part One

Purpose: To create a data base schema structure.

Lesson 2. The Schema Processor (1 hour 5 minutes)

Slides: 4.10 - DBSCHEMA
4.11 - Root File
4.12 - DBSCHEMA Files
4.13 - Schema Processor Command Format
4.14 - Schema Processor \$CONTROL Command *
4.15 - \$TITLE, \$PAGE Commands
4.16 - DBSCHEMA Summary Table

Activity: Lab: Data Base Creation - Part Two

Purpose: To create a data base root file.

Lesson 3. Data Base Creation (50 minutes)

Slides: 4.17 - DBUTIL: A Utility to Manage the Data Base *
4.18 - DBUTIL Command: CREATE
4.19 - DBUTIL Commands: ERASE, PURGE
4.20 - Data Base Files *

Activity: Lab: Data Base Creation - Part Three



Module 4-1

Data Definition Language

Goal: To define and use the TurboIMAGE data definition language.

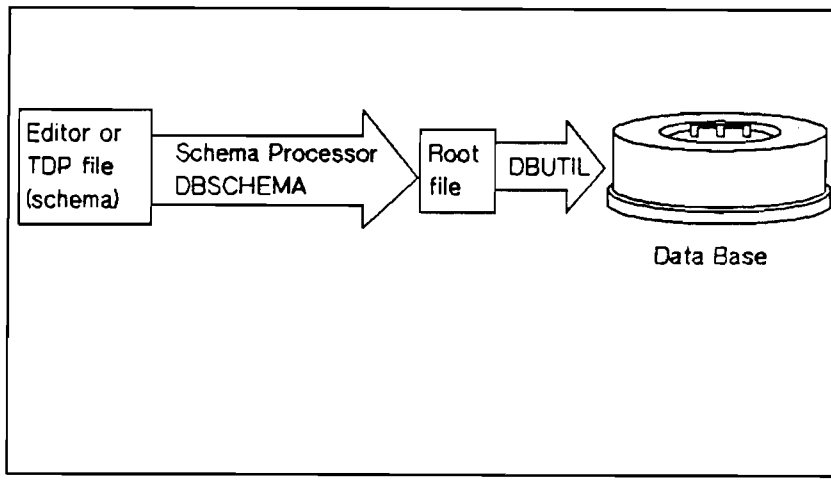
Objectives:

Upon completion of this module, you will be able to:

- define the schema structure.
- describe the components of each part of the schema structure.
- utilize DBSCHEMA to create a root file.
- utilize DBUTIL to create, erase, and purge a data base.

ie go through DDL portion and UTILITIES

Overview of Data Base Creation

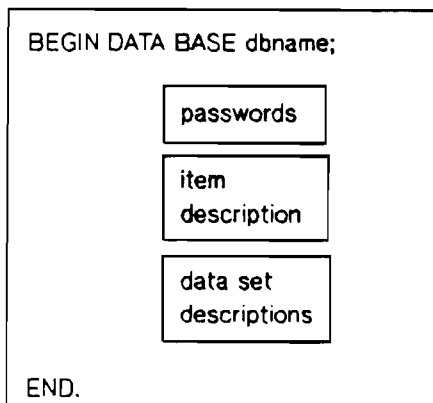


TI1 4.01

© 1985 Hewlett-Packard Company

Data Definition Language

The Schema



TI1 4.02

© 1985 Hewlett-Packard Company

- The schema is a description of the data base.
- Schema exists as a text editor MPE file.
- Schema is inputted to a Schema Processor.

□ Data Definition Language

Purpose: To present an overview of the data base creation process.

Key Points:

- The data base description, called a schema, can exist in the MPE system as an ASCII file on cards, magnetic tape, or as a catalogued disc file.
- The data base creator processes the schema using the TurboIMAGE Schema Processor which creates an internal description of the data base called a root file.
- DBUTIL, a TurboIMAGE utility program, builds the data base files according to requirements of the data base structure specified in the root file. The files contain no data initially.

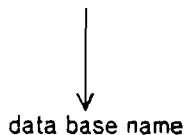
Purpose: To introduce the data definition language.

Key Points:

- Data definition language has a simple format and is easy to read, easy to understand, and easy to create.
- Once the data base has been designed, it must be described with the data definition language and processed by the Schema Processor to create the root file.
- The data base description, called a schema, can exist in the MPE system as an ASCII file on cards, magnetic tape, or as a catalogued disc file.
- Either EDITOR, TDP, or HPSLATE can be used to create the schema file.
- The overall schema structure has 5 parts: data base name, password part item part, set part, end.

Data Base Name

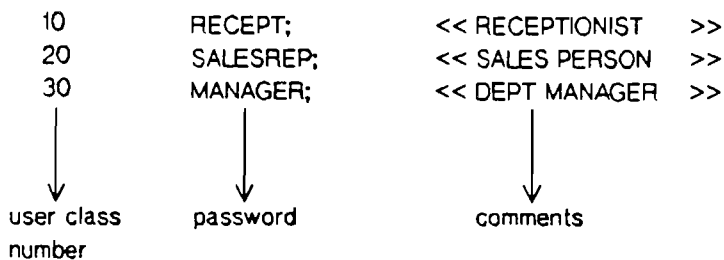
BEGIN DATA BASE REALTY;



- Data base name can be up to 6 alphanumeric characters, with first character alphabetic.

Password Part

PASSWORDS:



- Passwords restrict access to data items and data sets.
- User class number must be from 1 to 63 and unique within schema.
- Password can be up to 8 ASCII characters, with first character alphabetic.
- Additional information on data base security will be provided in Module 6.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: User Classes and Passwords and Section 3: PASSWORD PART.

Module 4-3 Instructor Notes: Slides 4.03/4.04

Data Definition Language

Purpose: To introduce the schema component Data Base Name.

Key Points:

- The data base name is used by TurboIMAGE to name all of the data sets the data base by attaching 2 numeric digits at the end (REALTY01 REALTY02, ... REALTY99, REALTYA0, ...REALTYA9, REALTYB0, ...REALTYB9, ...REALTYJ0, ...REALTYJ9)
- There exists a LANGUAGE parameter for a native language definition name or number of the data base. The default language is the US ASCII character set. Refer to Native Language Support Reference Manual for further information.

Eg Spanish, Japanese etc.

Purpose: To introduce the schema component Password Part.

Key Points:

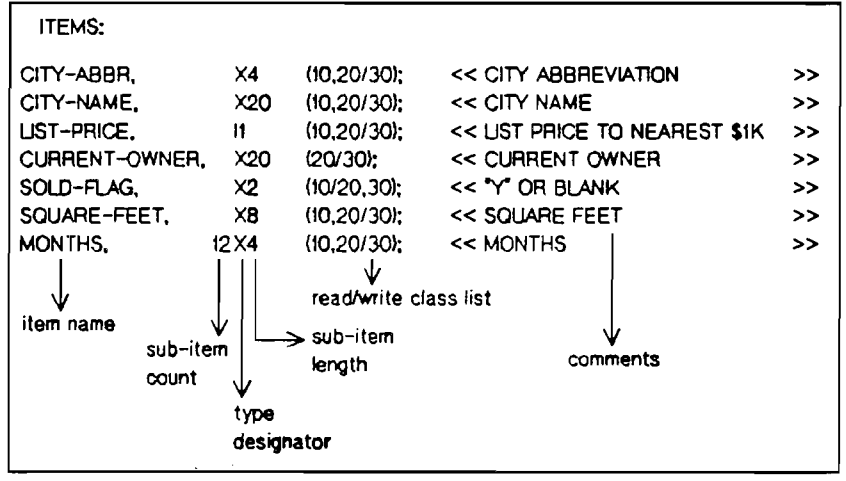
- The password component of the schema defines user classes and passwords.
- The password is what a user or program supplies to gain access to the base.
- The passwords are not upshifted by the Schema Processor. This results broader range of passwords. Everything else is upshifted by the Schema Processor.
- It is not an error to omit the password, however, the Schema Processor ignores lines containing only a user class number.
- <<comments>> may appear anywhere.
- User class numbers start at 1 on the password part of the schema, how user class numbers of 0 can be assigned to items and sets.
- When you open the data base you must provide a valid password to establish your user class number. If you do not provide one, you will be granted user class number 0. If you are the data base creator and supply a semicolon as a password, the number 64 is used to grant you unlimited data base access privileges.
- In-depth information on data base security is provided in Module 6.

At this point when you open the data base, the user class number is granted to you. If you do not provide a password, you will be granted user class number 0.

Data Definition Language

Notes

Item Part



- Defines each item in data base.
- Use the following two pages to record details on each of the components of the item part of a schema.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: Data Items and Section 3: Item Part and Type Designators Table.

Data Definition Language

Item Part

Item name:

- Item name can be up to 16 alphanumeric characters, with first character alphabetic.
- Only 255 item names can appear in one data base (1023 for TurboIMAGE).
-
-

Sub-item count:

- The appearance of a sub-item count denotes occurrences of a sub-item within an item, which makes the item a "compound" item.
- Sub-item count must be from 1 to 255.
-
-

Type designator:

- Type designator denotes how the item value is stored on disc.
- Word designators:
 - I - K -
 - J - R -
- Character designators:
 - U - Z -
 - X -
- Nibble designator:
 - P -
 -
 -

Data Definition Language

Sub-item length:

- Sub-item length denotes the length of each sub-item within an item.
- Sub-item length must be from 1 to 255 depending on the type designator.
- Each data item length must be an even number of bytes.
-
-

Read class list/write class list:

- Read class list/write class list specifies which user classes are permitted read and write access to the data item.
- User class numbers must be between 1 and 63, inclusive.
- User class 0 is assigned for an invalid password.
- User class 64 is assigned to the creator (password=;).
-
-

Module 4-6 Instructor Notes: Slide 4.05

□ Data Definition Language

Purpose: To introduce the schema component Item Part.

Key Points: *Check the end of this module for preparation information.*

- Item names:
 - Some languages only allow items to be 15 characters.
 - Some special characters are allowed.
 - One item name can be used for multiple data sets.
 - The fewer item names used, the less memory used.
 - Do not use different names for the same type item just to make it for a particular data set.
 - The order in which the items are listed is of no importance. How listing them alphabetically makes things easier (when you are look for a name).
 - Some shops recommend items be in alphabetical order to make them easier to find in the item list.
- Sub-item count:
 - If no sub-item count is specified, then the sub-item length is actually the item length.
 - If sub-item count is omitted 1 is implied.
 - Sub-item count is an array type item (COBOL "OCCURS").
- Type designator:
 - I: A signed binary integer in 2's complement form.
 - J: Same as I but QUERY allows only numbers conforming to specifications for COBOL COMPUTATIONAL data to be entered.
 - K: An absolute binary quantity.
 - R: A real (floating point) number. * Warning COBOL does not handle floats - use fixed data instead
 - U: An ASCII character string containing no lowercase alphabetic characters.
 - X: An unrestricted ASCII character string.
 - Z: A zone decimal format number.
 - P: A packed decimal number.
 - Data items of type U, X, or Z (byte type) must have a sub-item count and sub-item length whose product is an even number (X2, X4, X6).
 - Data items of type P (1/2 byte or 1 nibble) must have sub-item count and sub-item length whose product is a multiple of 4 (P4, P8, P12)
 - Not all of the data types can be used by all programming languages This especially important if a data base is going to be accessed by more than 1 programming language.

(Continued on next page.)

Module 4-6 Instructor Notes: Slide 4.05 (cont'd)

Data Definition Language

- Sub-item length:
 - The Schema Processor will give an error for any data item that is integral number of words.
 - A data item cannot exceed 2047 words in length. *2047 words*
- Read class list/write class list:
 - User class numbers are defined in password part.
 - Write access implies read access. (More on read/write lists in Module 6.)
 - User class numbers start at "1" on the password part of the schema, however, user class numbers of "0" can be assigned to items and set *2 more after but 0 is dead 0*



2000



Module 4-7

Data Definition Language

Item Part Type Designators and Programming Languages

	COBOL	FORTRAN	RPG	SPL	BASIC	PASCAL	LEN
							bytes
I	COMPUTATIONAL S9 to S9(4)	INTEGER	Binary	INTEGER	INTEGER	small INTEGER	2
I2	COMPUTATIONAL S9(5) to S9(9)	INTEGER*4	Binary	DOUBLE INTEGER		INTEGER	4
I4	COMPUTATIONAL S9(10) to S9(18)		Binary				8
J	COMPUTATIONAL S9 to S9(4)	INTEGER	Binary	INTEGER	INTEGER	small INTEGER	2
J2	COMPUTATIONAL S9(5) to S9(9)	INTEGER*4	Binary	DOUBLE INTEGER		INTEGER	4
J4	COMPUTATIONAL S9(10) to S9(18)		Binary				8
K1		LOGICAL		LOGICAL		BOOLEAN	2
R2		REAL		REAL	REAL	REAL	4
R4		DOUBLE PRECISION		LONG	LONG	LONGREAL	8
Un	DISPLAY PICTURE A(n)	CHARACTER*n	Charact.	BYTE	String	PACKED ARRAY	n
Xn	DISPLAY PICTURE X(n)	CHARACTER*n	Charact.	BYTE	String	PACKED ARRAY	n
Zn	DISPLAY PICTURE S9(n)		Charact.			PACKED ARRAY	n
Pn	S9(n-1)COMP-3		Numeric			PACKED ARRAY	n/2

* Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 3: Item Part.

Module 4-7 Instructor Notes

Data Definition Language

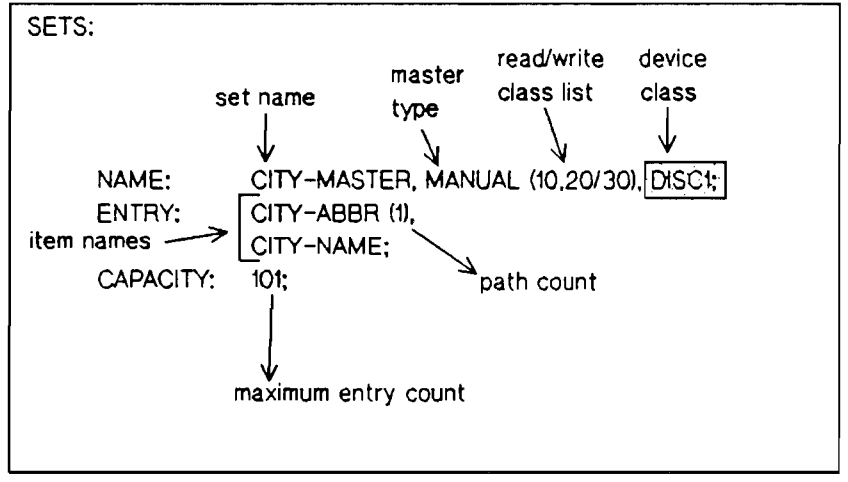
Purpose: To present the Item Part Type Designators and Programming Languages.

Key Points:

- Types J, P, or Z are commonly used for COBOL and RPG numerics, and I or R are used for the other languages.
- Type X is used for character type data.
- Use data types that can be used by all languages accessing the data base.

Arithmetic in certain languages, not always done in binary as some do, is done after translation to object code. eg. COBOL does arithmetic in decimal.

Set Part for Masters



- Path count indicates the number of paths to various detail data sets and also specifies the search item.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: Paths and Section 3: Set Part (MASTER SET).

Student Notes:

NAME:

set name _____

master data set type _____

read/write class list _____

device class _____

ENTRY:

item name _____

path count _____

CAPACITY:

maximum entry count _____

Module 4-8 Instructor Notes: Slides 4.06

□ Data Definition Language

Purpose: To present the schema component Set Part for master data sets.

Key Points: *Check the end of this module for preparation information.*

- The set part defines each master and detail data set in a data base.
- Read/write class list specifies which user classes are permitted read and write access to the data set.
- ✱ ■ **Device class parameter is used for data base placement onto specific devices.**
- ENTRY specifies which data items belong to the set.
- A master can have more than one path to a single detail set provided detail set has more than one search item specified as a link to that master.
- There can be up to 16 path counts. A path count of 16 does not necessarily equate to 16 different details. If path count is 0, then the master stand-alone set. A stand-alone master can still be accessed by key.
- Automatic masters will always have only 1 data item name in ENTRY (th search item).
- ✱ ✱ ■ A good design tip is to allow room for growth by keeping master sets at least 20% empty.

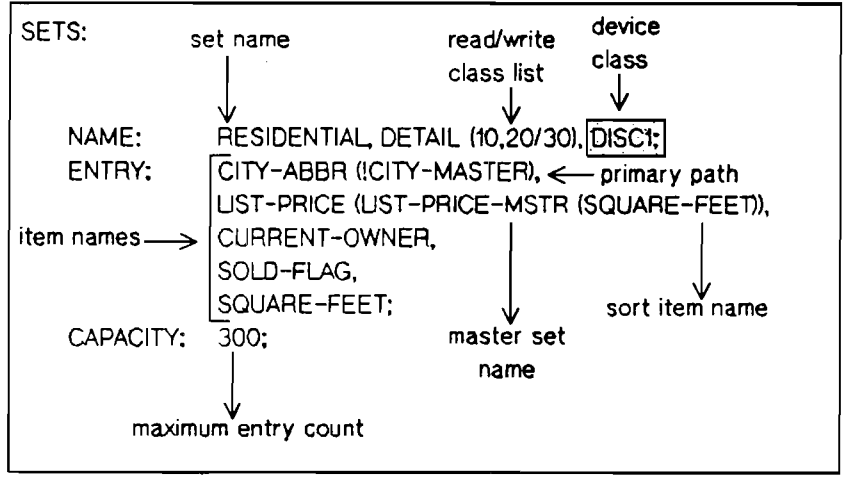
✱ Extremely useful in very large data base environments; TERADATA is a good example. In many cases, data sets are not guaranteed a physical or logical device. Extra effort is then required to manipulate using special facilities, frequently implemented as macros. Response to move set is slow.

✱ ✱ Talk about prime numbers on master sets — explain its reasons for hashing algorithm + the fact it will help for fewer synonyms (more much later in course)

Data Definition Language

Notes

Set Part for Details



- Master set name indicates a path from the specified master and also specifies a search item.
- Sort item name denotes a chain to be maintained logically in sorted sequence.
- ! denotes a primary path.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: Primary Paths, Sort Items and Section 3: Set Part (DETAIL).

TI 4.07

© 1985 Hewlett-Packard Company

Student Notes:

NAME:

set name _____
 detail data set _____
 read/write class list _____
 device class _____

ENTRY:

item name _____
 primary path _____
 master set name _____
 sort item name _____

CAPACITY:

maximum entry count _____

Module 4-9 Instructor Notes: Slides 4.07

Data Definition Language

Purpose: To present the schema component Set Part for detail data sets.

Key Points: *Check the end of this module for preparation information.*

- There can be up to 16 search items.
- A detail set can have more than one item linked to the same master set.
- If the "!" is not used, the first unsorted path becomes the primary path.
- The primary path should be chosen as the most frequently accessed path.
- Detail chains can be "logically" maintained in a sorted sequence.
- Sort item name must be an item in the current entry and must be type U, K, or X.
- ⊗ ■ Sort items should be placed as the last item in the entry to avoid additional overhead.
- A stand-alone detail must be accessed serially or directly.
- A detail's set definition must appear after the associated master's set definition.

⊗ *With respect to sort path overhead - consider carefully the trade off between serial and parallel access. 12-15-00*

End Statement

BEGIN DATA BASE dbname;

passwords

item
description

data set
descriptions

END.

IMAGE/3000 and TurboIMAGE Limits



Description	IMAGE/3000	TurboIMAGE
Data item names per data base	255	1023
Data item names per data entry (set)	127	255
Data sets per data base	99	199
Maximum entry size	4094 bytes	4094 bytes
Entries per data set	$2^{23} - 1$	$2^{31} - 1$
Entries per chain	65535	$2^{31} - 1$
Data item length	4094 bytes	4094 bytes
Lock area size	4K words	8K words

$2^{23} - 1$ is approximately 8 million.

$2^{31} - 1$ is approximately 2 billion.

Module 4-10 Instructor Notes: Slides 4.08/4.09

Data Definition Language

Purpose: To present the schema component End.

Key Points:

- The data base description must contain the BEGIN and the END statements. Do not forget the END statement and the period.
- The Schema Processor will report an error if the END component is missing.

It is "END" + full stop



Purpose: To present the limitations of IMAGE/3000 and TurboIMAGE.

Key Points:

- In order to take advantage of these new limits, an IMAGE/3000 data base must be converted to TurboIMAGE. A migration utility, DBCONV, is provided with TurboIMAGE. This utility is discussed in Module 8.
- These new limits are achieved via an increase in the count field from 1 word to 2 words (refer to slide 2.08).

Remember the record layout of media record

*2 words for count field
2 words for count field
2 words for count field*

BREAK

Module 4-11

□ Activity 4-1 Lab: Data Base Creation – Part One

Miniature REALTY Data Base – Part One

Purpose: To create a data base schema structure.

Notes: This is part one of a three part data base creation lab. This activity will be accomplished in student groups of two. Part One describes a miniature data base that needs to be created by designing an appropriate TurboIMAGE data base schema structure. As the module progresses and you learn more about the Data Definition Language, you will be able to create the required data base.

The following is a description of the three labs in this module:

- Activity 4-1 Create the schema in an editor file.
- Activity 4-2 Create the root file using the schema processor.
- Activity 4-3 Create the data base files using a TurboIMAGE utility.

Directions:

1. Work in groups of two students per terminal.
2. Read the specifications of the proposed data base listed below.
3. Create the schema file using the Data Definition Language and name your schema file MINISCHM.groupn. You can use EDITOR or TDP to do this. You can refer to Appendix B for the schema syntax.

Note: Even though passwords are not specified, the word "PASSWORDS:" must appear in the schema file.

Specifications:

Wonder Realty Company wants to create a "miniature" data base to list commercial properties only. It will consist of 3 data sets: listing-number master, zoning master, and commercial detail. The specifications for this data base are on the following pages.

Solutions: One possible solution is provided in Appendix A.

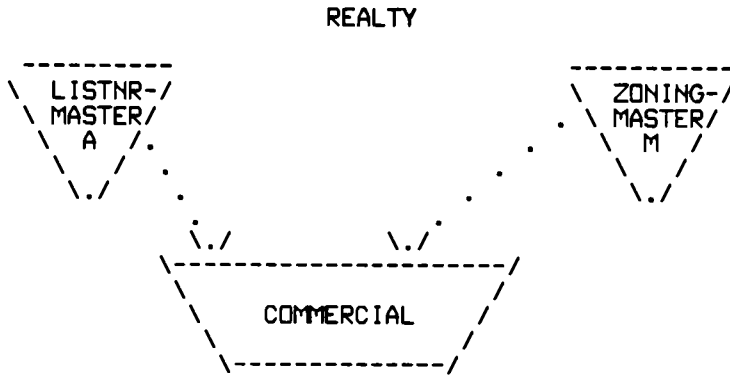
(Continued on next page.)

Module 4-12

Activity 4-1 Lab: Data Base Creation - Part One (cont'd)

Miniature REALTY Data Base - Part One

Specifications:



data set name

item name
item length
in bytes

LISTNR-MASTER

LISTING-NR
2

ZONING-MASTER

ZONING-CODE	ZONING-DESCRIP
4	20

COMMERCIAL

LISTING-NR	ZONING-CODE	LIST-PRICE	STREET-ADDRESS	SOLD-FLAG	SQUARE-FEET
2	4	2	30	2	8

Module 4-13

Activity 4-1 Lab: Data Base Creation - Part One (cont'd)

Miniature REALTY Data Base - Part One

Specifications:

DATA SET NAME	ITEM			KEY	PATH COUNT	CAPACITY	BLOCK-MAX**
	NAME	#	TYPE/DESIG				
LISTNR-MASTER	LISTING-NR	1	A I1	X	1	503	128
ZONING-MASTER	ZONING-CODE	2	M X4	X	1	31	128
	ZONING-DESCRIP	3	X20				
COMMERCIAL	LISTING-NR	1	D I1	X	2	322	768
	ZONING-CODE*	2	X4	X			
	LIST-PRICE	4	I1				
	STREET-ADDRESS	5	X30				
	SOLD-FLAG	6	X2				
	SQUARE-FEET	7	X8				

* primary path

** BLOCKMAX values will be used in Activity 4-2.

Module 4-13 Instructor Notes

□ Activity 4-1 Lab: Data Base Creation - Part One

Purpose: To design a data base schema structure.

Time: 30 minutes

Notes: Part one gives the specifications for the miniature REALTY data base. Students should be able to create a schema file (MINISCHM.groupn) of the required data base here.

Part two allows the students to add schema processor commands into their schema file. Then they will create the data base root file using DBSCHEMA.PUB.SYS.

Part three will be to create the data base structure using DBUTIL.PUB.SYS.

Directions:

1. Have students work in groups of two on this activity.
2. Explain that this activity is a three part lab ongoing throughout this module.
3. A solution is provided for the students in Appendix A. A copy is listed below.
4. Instruct students to use this exercise as a tool in designing the data base for PARKER MANUFACTURING.

Solution:

BEGIN DATA BASE REALTY;

PASSWORDS:

ITEMS:

```
LISTING-NR,          I  ;  << UNIQUE, SEQUENTIAL NUMBER >>
ZONING-CODE,         X4  ;  << ZONING CODE >>
ZONING-DESCRIP,     X20 ;  << ZONING DESCRIPTION >>
LIST-PRICE,         I   ;  << LIST PRICE TO NEAREST $1K >>
STREET-ADDRESS,    X30 ;  << STREET ADDRESS >>
SOLD-FLAG,          X2  ;  << "Y" OR BLANK >>
SQUARE-FEET,       X8  ;  << SQUARE FEET >>
```

SETS:

```
NAME:      LISTNR-MASTER, AUTOMATIC;
ENTRY:     LISTING-NR (1);
CAPACITY:  503;
```

```
NAME:      ZONING-MASTER, MANUAL;
ENTRY:     ZONING-CODE (1),
           ZONING-DESCRIP;
CAPACITY:  31;
```

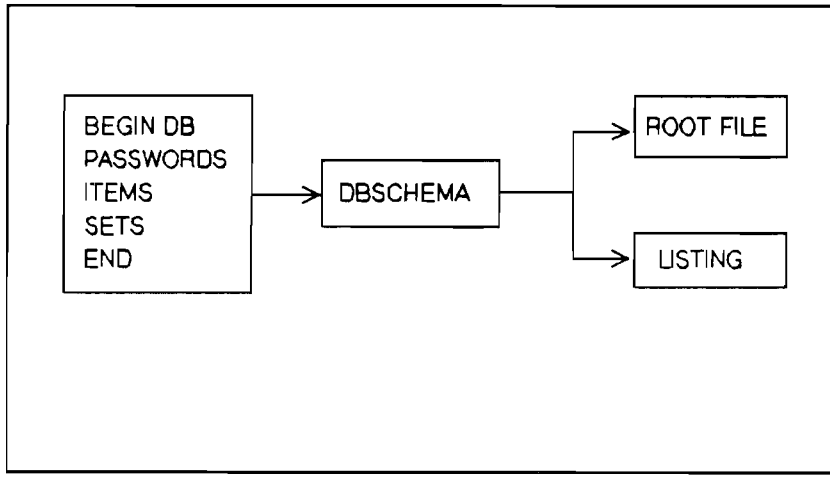
```
NAME:      COMMERCIAL, DETAIL;
ENTRY:     LISTING-NR (LISTNR-MASTER),
           ZONING-CODE (!ZONING-MASTER),
           LIST-PRICE,
           STREET-ADDRESS,
           SOLD-FLAG,
           SQUARE-FEET;
CAPACITY:  300;
```

END.

Data Definition Language

Notes

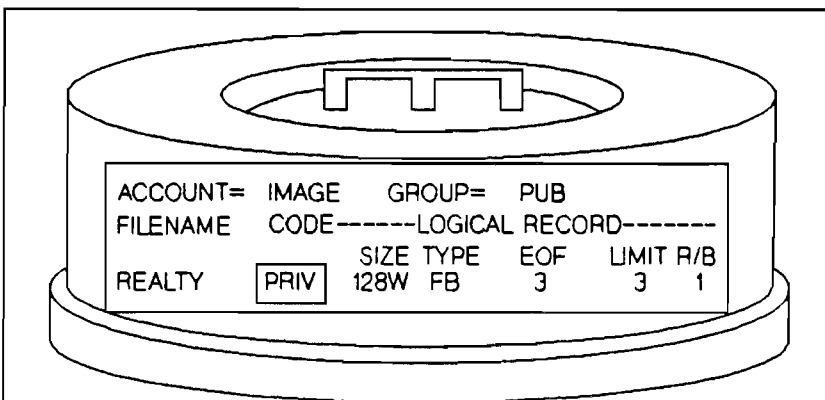
DBSCHEMA



- Processes SCHEMA textfile
- Produces output listing.
- Creates root file.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 3: Schema Processor Operation.

Root File



- An internal description of the Data Base.

Root file contains:

- creation date and time
- maintenance word
- passwords
- item table
- set table
- security for items and sets
- data set control blocks
- creator

Module 4-14 Instructor Notes: Slides 4.10/4.11

□ Data Definition Language

Purpose: To introduce DBSCHEMA.

Key Points:

- The Schema Processor is a program that accepts a textfile containing the schema as input, scans the schema, and if no errors are detected, optionally produces a root file (refer to slide 4.13).
- The Schema Processor prints a heading, an optional list of the schema and summary information on a listfile. *depends on \$CONTROL ~ a compiler options*
- The person who creates the root file is identified as the data base creator.

↑
the USER.ACCOUNT person - not physical person.

Purpose: To present root file contents.

Key Points:

- If a root file is requested, and the schema is error-free, it is created. Then, the root file is given the same name as the one specified for the data base in the schema, initialized, and saved as a catalogued disc file.
- The root file is a privileged file and cannot be accessed by MPE commands.
- The root file contains: creation date and time, maintenance word, passwords, item table (name, type, length, etc.), set table (name, type), security for items and sets, and data set control blocks (capacity, length, blocksize, counts, record definition, path tables)

Data Definition Language

Notes

DBSCHEMA Files

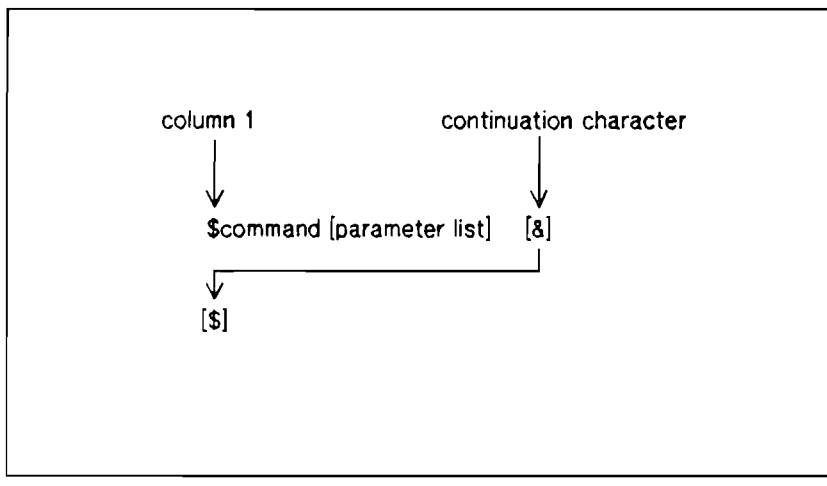
:RUN DBSCHEMA.PUB.SYS;PARM=n

PARM=	DBSTEXT	DBSLIST
1	:FILE	\$STDLIST
2	\$STDINX	:FILE
3	:FILE	:FILE
Omitted	\$STDINX	\$STDLIST

- Formal file designator of input textfile is DBSTEXT.
- Formal file designator of output listfile is DBSLIST.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 3: Schema Processor Operation.

Schema Processor Command Format



- Used to specify options while processing the schema.
- Can be used anywhere in schema.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 3: Schema Processor Commands.

Module 4-15 Instructor Notes: Slides 4.12/4.13

Data Definition Language

Purpose: To present DBSCHEMA.

Key Points:

- :FILE denotes that a file equation for DBSTEXT or DBSLIST must be issued.
- DBSCHEMA will not use the file equate unless the correct PARM value is used.
- It is necessary to know the formal file designators used by DBSCHEMA override the default files used (\$STDINX and \$STDLIST).
- \$STDINX is the standard job or session input device and \$STDLIST is the standard job or session output device. TurboIMAGE uses \$STDINX instead of \$STDIN, because of the special characters used in column 1 of the schema. For a description of these files, refer to *MPE Intrinsic Reference Manual*, Section 10: System-Defined Files.

Confirm everyone familiar with term "formal file designator"

Purpose: To present the format of Schema Processor commands.

Key Points:

- \$ must be in column 1.
- If a \$ command is continued with an ampersand (&), then the continuation line must also start with a \$ in column 1.

Just as in compilers — look up options in manual. 3-20

Most common use — Titles
Blocking factors — next slide anyway

Module 4-16 Instructor Notes: Slides 4.14/4.15

Data Definition Language

Purpose: To present Schema Processor command \$CONTROL.

Key Points: *Check the end of this module for preparation information.*

- The \$CONTROL command parameters are similar to the compiler commands.
- LIST causes all source records to output to the listfile. NOLIST can be used to suppress printing of the passwords. For security purposes NOLIST can be entered before passwords and then LIST entered after passwords in order to protect the passwords from appearing on the schema listing.
- ERRORS sets maximum errors to be encountered before DBSCHEMA aborts. Default value is 100.
- LINES sets the number of lines per page. Default value is 60 if list is a line printer and 32767 if it is not.
- ROOT causes the root file to be created if no errors are encountered.
- BLOCKMAX sets the maximum block size (in words) for the data sets. If it is specified once, that value will be used for all data sets. It can specified before any data set. This value affects buffer sizes and disc space used. Default value is 512, minimum value is 128, and maximum is 2048. Specify a value that is a multiple of 128.
- TABLE causes the summary table to output to the listfile.

Purpose: To present Schema Processor commands: \$TITLE, \$PAGE.

Key Points:

- Character strings are parameters for both \$TITLE and \$PAGE commands.

Aesthetics when preparing schema lists.

Data Definition Language

Notes

DBSCHEMA Summary Table



DATA SET NAME	TYPE	FLD CNT	PT CT	ENTR LGTH	MED REC	CAPACITY	BLK FAC	BLK LGTH	DISC SPACE
LISTNR-MASTER	A	1	1	1	11	503	11	122	48
ZONING-MASTER	M	2	1	12	22	31	5	111	9
COMMERCIAL	D	6	2	24	32	322	23	738	90
TOTAL DISC SECTORS INCLUDING ROOT: 156									

NUMBER OF ERROR MESSAGES: 0

ITEM NAME COUNT: 7 DATA SET COUNT: 3

ROOT LENGTH: 360 BUFFER LENGTH: 738 TRAILER LENGTH: 256

- For TurboIMAGE, the media record lengths for the master sets would be 12 and 23 because of the increased size of the count field for detail chain heads.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 3: Summary Information.

Data Definition Language

Purpose: To present the DBSCHEMA summary table.

Key Points:

- After the entire schema has been scanned, several types of summary information can be printed on the listfile (refer to slide 4.13, TABL parameter).
- If no errors are detected in the schema and if the TABLE option has been selected, the Schema Processor prints a table of summary information about the data sets.
- For the blocking factor, TurboIMAGE calculates a block length that will use the least number of total sectors for data sets.
- The media record lengths for the master sets would be 12 and 23 with TurboIMAGE. This results from the increase in the count field for detail chain heads.

Module 4-18

□ Activity 4-2 Lab: Data Base Creation - Part Two

Purpose: To create a data base root file.

Notes: This is part two of the three part data base creation lab for Module 4. This activity requires you to create the root file for the miniature REALTY data base using DBSCHEMA.PUB.SYS.

Directions:

1. Work on this lab with your partner from Activity 4.1.
2. Add schema processor commands to your schema file MINISCHM.groupn. You should include the following:
 - title for each major data base component
 - maximum block size for the data sets (Refer to Activity 4-1).
3. Run DBSCHEMA.PUB.SYS with appropriate file commands and parameters, to create the REALTY root file and to get an offline listing of your schema. (Note: If you enter DBSCHEMA without specifying PARM, you will be prompted for input. To exit DBSCHEMA, type : or :EOD to return to MPE. Then reissue the correct DBSCHEMA command.)

Specifications:

Files you will need for this activity are:

- MINISCHM.groupn - Data base schema for miniature REALTY data base.
- DBSCHEMA.PUB.SYS - HP utility program to create root file.
- REALTY.groupn - REALTY data base root file.

Solution: One possible solution is provided in Appendix A.

LUNCH TUESDAY

Module 4-18 Instructor Notes

Activity 4-2 Lab: Data Base Creation - Part Two

Time: 30 minutes

Purpose: To create a data base root file.

Notes: This is part two of the three part data base design lab for Module 4. At the end of this lab students should have created the miniature REALTY data base root file using DBSCHEMA.PUB.SYS.

Part three will be to create the data base structure using DBUTIL.PUB.SYS.

Directions:

1. Have students work in groups of two on this activity.
2. Instruct students to use this exercise as a tool in designing the data base for PARKER MANUFACTURING.
3. A solution is provided for students in Appendix A. A copy is listed below.

Solution:

Miniature REALTY Data Base - Part Two

PAGE 1 HEWLETT-PACKARD 32215B.04.30 IMAGE/3000: DBSCHEMA
TUE, DEC 18, 1984, 3:41 PM (C) HEWLETT-PACKARD CO. 1978

```
$TITLE 'WONDER REALTY DATA BASE'
```

```
BEGIN DATA BASE REALTY;
```

```
PASSWORDS:
```

```
ITEMS:
```

```
LISTING-NR,          I      ; << UNIQUE, SEQUENTIAL NUMBER >>  
ZONING-CODE,         X4      ; << ZONING CODE >>  
ZONING-DESCRIP,     X20     ; << ZONING DESCRIPTION >>  
LIST-PRICE,         I        ; << LIST PRICE TO NEAREST $1K >>  
STREET-ADDRESS,    X30     ; << STREET ADDRESS >>  
SOLD-FLAG,          X2      ; << 'Y' OR BLANK >>  
SQUARE-FEET,       X8      ; << SQUARE FEET >>
```

```
$TITLE 'MASTER DATA SETS'
```

```
SETS:
```

```
$CONTROL BLOCKMAX=128
```

```
NAME:      LISTNR-MASTER, AUTOMATIC;  
ENTRY:     LISTING-NR (1);  
CAPACITY:  503;
```

```
NAME:      ZONING-MASTER, MANUAL;  
ENTRY:     ZONING-CODE (1),  
           ZONING-DESCRIP;  
CAPACITY:  31;
```

Module 4-18 Instructor Notes

Activity 4-2 Lab: Data Base Creation - Part Two (cont'd)

\$TITLE 'DETAIL DATA SETS'

\$CONTROL BLOCKMAX=768

NAME: COMMERCIAL, DETAIL;
ENTRY: LISTING-NR (LISTNR-MASTER),
ZONING-CODE (!ZONING-MASTER),
LIST-PRICE,
STREET-ADDRESS,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

END.

LISTNR-MASTER	A	1	1	1	11	503	11	122	48
ZONING-MASTER	M	2	1	12	22	31	5	111	9
COMMERCIAL	D	6	2	24	32	322	23	738	90

TOTAL DISC SECTORS INCLUDING ROOT: 156

NUMBER OF ERROR MESSAGES: 0

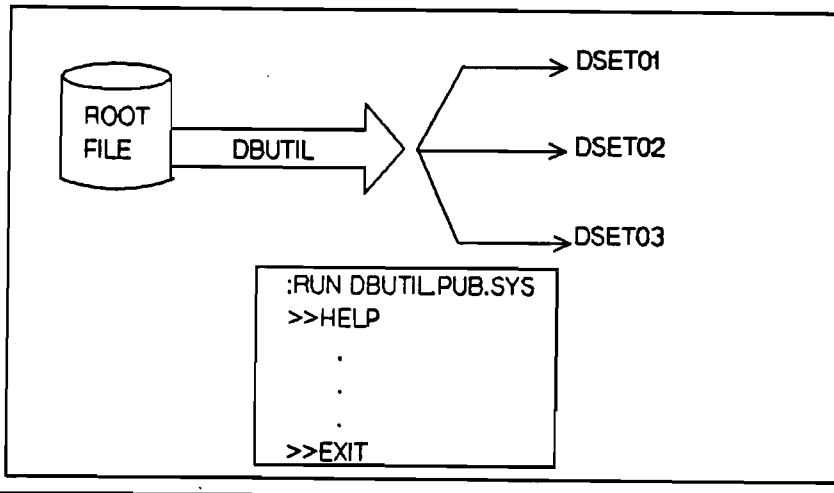
ITEM NAME COUNT: 7 DATA SET COUNT: 3

ROOT LENGTH: 360 BUFFER LENGTH: 738 TRAILER LENGTH: 256

ROOT FILE REALTY CREATED.



DBUTIL: A Utility to Manage the Data Base



Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 8: DBUTIL, HELP, and EXIT.

T11 4-17

© 1985 Hewlett-Packard Company

DBUTIL Commands: CREATE

```

:RUN DBUTIL.PUB.SYS
>>CREATE REALTY [/maint word]
  
```

- Allocates disc space for data sets.
- Initializes data sets to binary zeros.
- Saves data sets as privileged MPE files.
- Names data sets.
- Can assign maintenance word.

■ Data set names:

```

REALTY01
REALTY02
.
REALTY99
REALTYA0
.....
REALTYA9
REALTYB0
.....
REALTYB9
.....
REALTYJ0
.....
REALTYJ9
  
```

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 8: CREATE.

T11 4-18

© 1985 Hewlett-Packard Company

Module 4-19 Instructor Notes: Slides 4.17/4.18

Data Definition Language

Purpose: To introduce some features of DBUTIL and DBUTIL commands HELP and EXIT.

Key Points: *Check the end of this module for preparation information.*

- The DBUTIL program performs several different functions according to command that is entered.
- HELP displays all DBUTIL commands. HELP followed by a command name displays information on that command.
- EXIT terminates DBUTIL execution.
- DBUTIL can create, erase, or purge the data base.
- DBUTIL creates the data base in the user's account and group.

Go to terminals - run DBUTIL or HELP erase, purge and create.

Purpose: To introduce DBUTIL CREATE command.

Key Points:

- DBUTIL names the data sets using the name of the data base (or root file) plus 2 digits beginning with 01. TurboIMAGE data set names may be incremented up to J9.
- The data base name cannot be more than 6 characters because the file only allows a maximum of 8 characters for a file name. This naming convention is what places the limit of data sets per data base to 99 (199 for TurboIMAGE).
- The maintenance word is a type of password to access the data base with DBUTIL only. You must either be the creator of the data base or know maintenance word in order to access the data base via DBUTIL. A maintenance word is good security for a data base.
- The creator must log on with the same user name, group, and account as was used when the root file was created.

Data Definition Language

Notes

DBUTIL Commands: ERASE, PURGE

:RUN DBUTIL.PUB.SYS

>>ERASE REALTY [/maint word]

- Re-initializes all data sets on the data base to binary zeros.
- Data sets remain as allocated MPE files.
- Commonly used before reloading data back into data base.

>>PURGE REALTY [/maint word]

- Purges all data sets and root file.
- Commonly used when restoring or reconstructing a data base.

- The CREATE, ERASE, and PURGE commands can be executed as follows:

```
:RUN DBUTIL.PUB.SYS,CREATE
:RUN DBUTIL.PUB.SYS,PURGE
:RUN DBUTIL.PUB.SYS,ERASE
```

DBUTIL will then prompt you for the data base name.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 8: ERASE, PURGE.

Data Base Files



:LIST REALTY@.2

ACCOUNT = IMAGE

GROUP = PUB

FILENAME	CODE	-----LOGICAL RECORD-----					-----SPACE-----		
		SIZE	TYP	EOF	LIMIT	R/B	SECTORS	#X	MX
REALTY	PRV	128W	FB	3	3	1	9	1	1
REALTY01	PRV	128W	FB	28	28	1	30	1	1
REALTY02	PRV	128W	FB	21	21	1	23	1	1
REALTY03	PRV	640W	FB	13	13	1	70	1	1

:RUN QUERY.PUB.SYS

>FORM SETS

DATA BASE: REALTY

SETS	TYPE	ITEM COUNT	CAPACITY	ENTRY COUNT	ENTRY LENGTH	BLOCKING FACTOR
LIST-PRICE-MSTER	A	1	307	0	1	11
CITY-MASTER	M	2	101	0	12	5
RESIDENTIAL	D	5	312	0	18	24

Module 4-20 Instructor Notes: Slides 4.19/4.20

□ Data Definition Language

Purpose: To introduce DBUTIL commands: ERASE, PURGE.

Key Points:

- To use DBUTIL, ERASE you must be the data base creator or supply the correct maintenance word.
- Purging removes the files from the catalog and returns the disc space to the system.
- Only DBUTIL, PURGE can be used to purge data base files.
- If only the root file exists, DBUTIL, PURGE command must still be used.
- DBUTIL, PURGE will purge all files that have the data base name followed by 2 numeric digits, whether or not it is a data set; a warning is given if this occurs.

Clarify entry points to DBUTIL, their use

Purpose: To present data base files.

Key Points:

- The LISTF EOF and LIMIT values are identical, showing that all of the disc space is allocated.
- Run QUERY and use FORM SETS command to see the actual number of entries within each data set (0 at this time).
- REALTY01 corresponds to LIST-PRICE-MSTR, REALTY02 corresponds to CITY-MASTER, etc. in the same order that they were defined in the schema.
- The LISTF shows 28 records allocated, although the data set capacity is 307. This is because LISTF sees each block as a record (blocking factor \times EOF = capacity, or $11 \times 28 = 308$).

Module 4-21

□ Activity 4-3 Lab: Data Base Creation - Part Three

Purpose: To create data base files.

Notes: This is the last part of the three part data base creation lab for Module 4. This activity requires you to create the complete miniature data base structure using DBUTIL.PUB.SYS, CREATE. You will then purge the data base and create the entire REALTY data base that will be used throughout the course.

Directions: (Part 1)

1. Work on this lab with your partner from Activity 4.2.
2. Run DBUTIL.PUB.SYS,CREATE to create the remainder of your data base structure.
 - Once the root file is created do not forget to purge it if you need to recreate it.
3. Look at your data base files.
 - Perform a LISTF REALTY@,2.
 - Run QUERY.PUB.SYS.
Define the REALTY data base.
Use ";" for the password.
Use "1" for the mode.
Perform FORM SETS command.
Exit QUERY.

Directions: (Part 2)

1. Run DBUTIL.PUB.SYS,PURGE to purge the current miniature REALTY data base.
2. Text LAB4SCHM.LABS into the EDITOR and save it in your group.
3. Run DBSCHEMA.PUB.SYS with appropriate file commands and parameters to create the REALTY root file, and produce an offline listing. Use LAB4SCHM for your DBSTEXT file.
4. Run DBUTIL.PUB.SYS,CREATE to create the remainder of the data base structure.
5. Text LOADDDBJ.LABS into the EDITOR, modify username to your log-on, and save as LOADDDBJ in your group.
6. Stream LOADDDBJ to load initial data into your data base.

(Continued on next page.)

Module 4-22

Activity 4-3 Lab: Data Base Creation – Part Three

Specifications:

Files you will need for this activity are:

- DBSCHEMA.PUB.SYS - HP utility program to create root file.
- DBUTIL.PUB.SYS - HP utility program to create data sets.
- REALTY.groupn - REALTY data base.
- LAB4SCHM.LABS - Schema file for complete REALTY data base in EDITOR format.
- LOADDDB.LABS - IMAGE course utility program to load the complete REALTY data base.
- LAB4DTAn.LABS - Data files used by LOADDDB program. (n = 1 through 4)
- LOADDDBJ.LABS - Data base load job stream.

Module 4-22 Instructor Notes

Activity 4-3 Lab: Data Base Creation - Part Three

Purpose: To create data base files.

Time: 30 minutes

Notes: This is the last part of the three part data base creation lab for Module 4. At the end of this lab students should have created the complete miniature data base structure using DBUTIL.PUB.SYS, CREATE.

Directions:

1. Have students work in groups of two on this activity.
2. Inform students that the second data base structure, LAB4SCHM, will be used for the lab activities in Module 5.
3. A copy of LAB4SCHM is listed in Appendix A.

End

Module 4-23 Instructor Notes

Preparation Material: Slides 4.05/4.06

Teaching Tips:

Point out each of the 6 Item Part components: item name, sub-item count, type designator, sub-item length, read class list/write class list, comments. The page adjacent to the slide is provided to help students take class notes on this subject. Have students take notes in the space provided under each of the item part component headings.

Preparation:

Begin the presentation with the set part characteristics that are common to both master and detail definitions. Point out the need for the required words "SETS:" (specified once) and "NAME:" (once for each set definition).

The set part defines each data set in a data base. The data set name can be up to 16 alphanumeric characters, with the first character alphabetic. The set part also denotes the data set type. Just A, M, or D can be used to designate set type.

User class numbers are defined in password part. Write access implies read access.

* The device class parameter can be used to specify on which device the data set will reside. This option can be used to "spread" the data base and eliminate disc head contention for that device.

Path count must be an integer between 0 and 16, inclusive. The order the data items appear determines physical sequence in the entry. CAPACITY specifies the maximum number of entries in a data set.

It is not an easy task to change capacity of an established data set since all disc space is allocated at creation time.

Module 4-24 Instructor Notes

Preparation Material: Slides 4.07, 4.16

Preparation:

When a data base is reorganized (DBUNLOAD, DBLOAD), primary path chains are written into continuous areas on disc. This can greatly increase performance for chained reads. Therefore, the primary path should be chosen as the most frequently accessed path.

If no search items are specified, the detail is a stand-alone and must be accessed serially or directly (not chained). If a detail does have paths defined, then the set definition must appear after the associated master's set definition in the schema.

Maintaining a detail chain "logically" in sorted sequence, allows a user to read a particular detail chain in sequence. When a new entry is added with the same sort item value as an existing entry, TurboIMAGE will use the item values following the sort item in the entry as an "extended sort field," to position the new entry. Since this will add additional overhead, always place the sort item as the last item in the entry.

Teaching Tips:

Provide examples of uses of the various schema processor commands. \$CONTROL NOLIST and \$CONTROL LIST can be used immediately preceding and immediately following the password section to help increase security. \$CONTROL NOROOT is useful in the development process of a data base.

Module 4-25 Instructor Notes

Preparation Material: Slides 4.17, 4.20

Teaching Tips:

Have students log-on to their terminals and run DBUTIL.PUB.SYS and perform the HELP and EXIT commands. Also have them perform the HELP CREATE, HELP ERASE, HELP PURGE, and HELP EXIT commands. This module will cover the CREATE, ERASE, and PURGE commands. Other DBUTIL commands will be presented in Module 8.

Teaching Tips:

Students will get a chance to look at the data base files in Activity 4.



Module 5 Instructor Notes

Data Base Access

Overview of Module 5

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. Data Base Access Methods (1 hour)

Slides: 5.01 - Data Base Access Methods *
5.02 - Calculated Access and Synonyms *

Overlay 1 - First Primary Entry
Overlay 2 - First Secondary Entry
Overlay 3 - Second Secondary Entry
Overlay 4 - Primary Location Occupied

5.03 - Migrating Secondaries - Case One
5.04 - Migrating Secondaries - Case Two
5.05 - Chained Access*

Activity: Worksession: Data Set Access

Purpose: To implement the placement of entries in a master data set and a detail data set and to manipulate pointers to reflect migrating secondaries and the deletion of entries.

Lesson 2. Data Manipulation Language (25 minutes)

Slides: 5.06 - Data Manipulation Language *
5.07 - Calling a TurboIMAGE Procedure
5.08 - TurboIMAGE Parameters
5.09 - Status Parameter *

Text Page: Parameter Data Types

Lesson 3. TurboIMAGE Procedures (1 hour 30 minutes)

Slides: 5.10 - DBOPEN *
5.11 - Opening a Data Base with RPG
5.12 - TurboIMAGE and BASIC
5.13 - DBINFO
5.14 - DBERROR
5.15 - DBEXPLAIN
5.16 - DBPUT *
5.17 - Adding Entries with RPG
5.18 - DBCLOSE

Activity: Lab: Accessing a Data Base *

Purpose: To gain familiarity with the TurboIMAGE calls: DBOPEN, DBPUT, DBEXPLAIN, and DBCLOSE.

Module 5 Instructor Notes

Data Base Access

Overview of Module 5 (cont'd)

Lesson 4. Data Retrieval (1 hour 5 minutes)

Slides: 5.19 - DBFIND *
5.20 - DBGET *
5.21 - DBGET continued*
5.22 - RPG Data Retrieval

Activity: Lab: Data Retrieval

Purpose: To gain familiarity with the use of the DBFIND and DBGET intrinsics.

Lesson 5. Data Modification (55 minutes)

Slides: 5.23 - DBUPDATE
5.24 - DBDELETE

Activity: Lab: Data Modification

Purpose: To gain familiarity with the use of DBUPDATE and DBDELETE.

Module 5-1

Data Base Access

Goal: To present basic data base access methods and TurboIMAGE data manipulation procedures.

Objectives:

Upon completion of this module, you will be able to:

- identify the four basic data base access methods.
- define and give examples of primary and secondary entries.
- state two causes of migrating secondaries.
- identify the necessary parameter data types for TurboIMAGE procedures.
- use TurboIMAGE procedures in an application program.

ie Go through DML in detail.

Data Base Access

Notes

Data Base Access Methods

- Serial _____

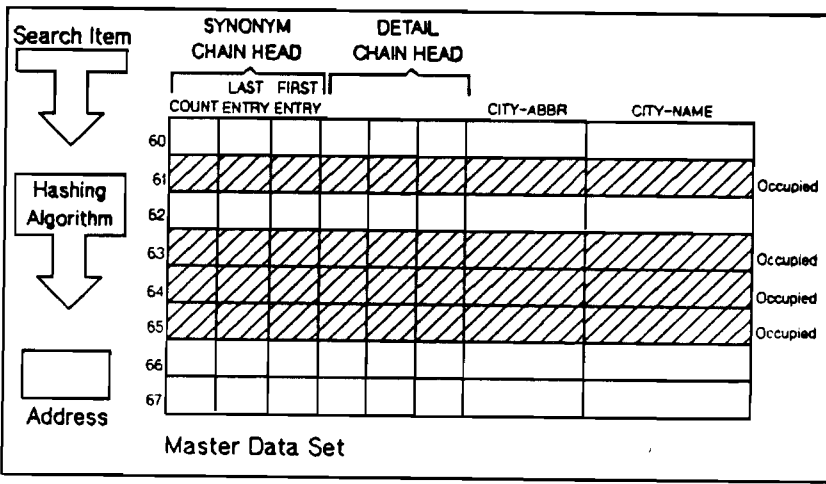
- Direct _____

- Calculated _____

- Chained _____

See ...
4 ...
4 ...
2 ...
1-12

Calculated Access and Synonyms



count=1 --> primary with no secondaries
 count=0 --> entry is a secondary
 count>1 --> number of total entries in synonym chain including the primary entry

For details on the hashing algorithm, refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 10: Primary Address Calculation.

Examples:

- STGA hashes to 62
- CAMP hashes to 62
- CUP hashes to 62
- SJ hashes to 66

Module 5-2 Instructor Notes: Slides 5.01/5.02

Data Base Access

Purpose: To define four methods of accessing a TurboIMAGE data base.

Key Points: *Check the end of this module for preparation information.*

- In the serial mode of access, TurboIMAGE reads sequentially through adjacent records until a particular data base entry is located. TurboIMAGE allows both forward and backward serial access. Serial access is most useful when most of the data in a data set is to be retrieved. The availability of serial access allows a data set to be used in the same way as an MPE file.
- Direct access can be used when an entry's record number is known. This type of access is used when records have already been located via other access methods and the record needs to be re-accessed. *This is NOT "keyed read"*
- Calculated access is used to retrieve an entry from a master data set by specifying a particular search item value. Calculated access is useful for retrieving a single entry. Calculated access is achieved by using a hashing algorithm. The exact technique of address calculation used by TurboIMAGE is explained in *TurboIMAGE Data Base Management System Reference Manual*, Section 10: Primary Address Calculation. The concepts of synonym chains and migrating secondaries are covered in slides 5.02 and 5.03. *This is "keyed read"*
- Chained access is used to retrieve the next entry in a chain. To perform this type of access to detail data set entries, the beginning of the chain must be located first. The application program specifies the search item value and TurboIMAGE locates the appropriate entry in the master data set. The entry contains pointers to the first and last entries in the chain and the count of the number of entries in the chain. TurboIMAGE allows both forward and backward chained access.

next 4 slides shown as overlays.

Purpose: To illustrate calculated access to a master data set and to present an example of a synonym chain.

Key Points: *Check the end of this module for preparation information.*

- Record addresses of master data set entries are determined by a hashing algorithm applied to search item values. A calculated address is called a primary address and an entry that resides at its primary address is called a primary entry.
- When a new entry with a unique search item value is assigned the same primary address as an existing primary entry, the entries are called synonyms. TurboIMAGE assigns the new entry a secondary address and the entry is known as a secondary entry.
- A primary entry together with all of its synonyms is called a synonym chain. Each synonym chain is maintained by a five word chain head in the media record of the primary entry and five word links in the media records of the secondary entries. A master data set can contain both a synonym chain head and multiple detail chain heads.

Data Base Access

Notes

Migrating Secondaries - Case One: A New Primary Entry

	Synonym Chain Head		Detail Chain Head			CITY-ABBR	CITY NAME	
	LAST	FIRST	LAST	FIRST	ENTRY			
	COUNT	ENTRY	ENTRY					
60	0	00	67	--	--	CAMP	CAMPBELL	Secondary
61								Occupied
62	3	67	60	--	--	STGA	SARATOGA	Primary
63								Occupied
64								Occupied
65								Occupied
66	1	00	00	--	--	SJ	SAN JOSE	New Primary
67	0	60	00	--	--	CUP	CUPERTINO	← migrated secondary

Master Data Set

1. SJ hashes to location 66.
2. The secondary occupying location 66 migrates to location 67.
3. The pointers are updated in locations 60 and 62.

Migrating Secondaries - Case Two: A Deleted Primary Entry

	Synonym Chain Head		Detail Chain Head			CITY-ABBR	CITY NAME	
	LAST	FIRST	LAST	FIRST	ENTRY			
	COUNT	ENTRY	ENTRY					
60	2	00	67	--	--	CAMP	CAMPBELL	Secondary migrates
61								Occupied
62	2	67	67	--	--	CAMP STGA	CAMPBELL SARATOGA	DELETE
63								Occupied
64								Occupied
65								Occupied
66	1	00	00	--	--	SJ	SAN JOSE	Primary
67	0	60	00	--	--	CUP	CUPERTINO	Secondary

Master Data Set

1. The primary entry STGA is deleted.
2. The first secondary in the synonym chain migrates to the primary location.
3. The pointers in location 62 and 67 are updated.
4. CAMP is now a primary with CUP as its only secondary.

□ Data Base Access

Purpose: To illustrate an example of a migrating secondary.

Key Points:

- The entry with the search item value SAN JOSE is placed in the location corresponding to its primary address calculation.
- The secondary that originally occupied this location MIGRATES to a new location.
- The pointers in the synonym chain head are updated to reflect this migration.
- This sample master data set now has two primary entries, one of which has two associated secondary entries.



Purpose: To illustrate an example of a migrating secondary.

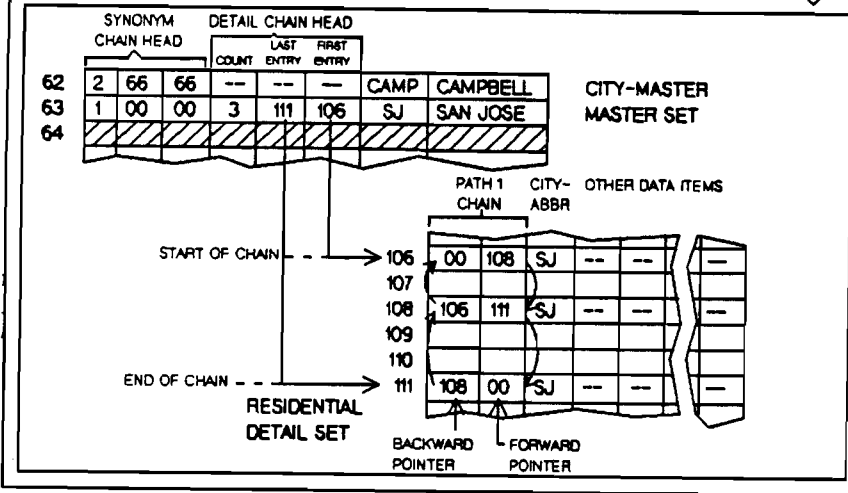
Key Points:

- If a primary entry is deleted from a master data set and the primary entry has associated secondary entries, the first secondary in the chain will migrate to the primary position.
- After the primary entry is deleted and the secondary entry migrates, the synonym chain head at location 62 has a count of 2, indicating 2 entries in the chain. The backward pointer at location 67 is updated to reflect the change in location of the other entry in the chain.

Data Base Access

Notes

Chained Access



Module 5-4 Instructor Notes: Slides 5.05

Data Base Access

Purpose: To illustrate an example of chained access.

Key Points: *Check the end of this module for preparation information.*

- A chain consists of several entries with the same search item value linked together via pointers.
- Both forward and backward pointers are used to maintain the chain.
- Chained access to master data sets retrieves entries in a synonym chain.
- Chained access to detail data sets retrieves data entries with the same search item value. This type of access is useful when retrieving information about related events, such as all residential listings in San Jose or all listings of houses with 3 bedrooms.
- When records are added to a chain they are added to the end of the logical chain, assuming the chain is not a sorted chain. Therefore, entries are chained in chronological order.
- When entries are added or deleted in a chain, the pointers and count field must be updated in the master chain head as well as the pointers in the detail data set.
- The first entry in a chain has a backward pointer equal to zero and the last entry in a chain has a forward pointer equal to zero.

Also, check on slide 5.06 for a diagram of how entries in the chain.

Module 5-5

□ Activity 5-1 Worksession: Data Base Access

Purpose: To implement the placement of entries in a master data set and a detail data set and to manipulate pointers to reflect migrating secondaries and the deletion of entries.

Directions: Trace the additions and deletions described below and fill in all necessary information in both the master data set and the detail data set. Your final answer should represent the contents of the two sets after all the additions and deletions have been completed. Several blank work copies of the data sets are provided on the following pages.

Notes:

- An entry cannot be placed in a detail data set until the key item value is placed in the master data set.
- All entries in the detail data set that have the same key item value are linked together using forward and backward pointers.
- Entries in a master data set that have the same calculated (hash) address are linked together in a synonym chain.
- Secondaries migrate if a primary entry is deleted or if a new key item value hashes to a location occupied by a secondary.

Specifications:

Part 1: Add entries to the master data set.

	CITY-ABBR	CITY-NAME	HASHED ADDRESS
1)	PA	Palo Alto	41
2)	SJ	San Jose	41 (use 40 as the secondary address)
3)	MV	Mountain View	40

Part 2: Add and delete entries to the detail data set. Be sure to enter the appropriate detail chain head information in the master set.

	LISTING NUMBER	CITY-ABBR	DETAIL SET ADDRESS	
4)	01	PA	161	<< ADD >>
5)	02	SJ	162	<< ADD >>
6)	03	SJ	163	<< ADD >>
7)	04	SJ	164	<< ADD >>
8)	05	MV	165	<< ADD >>
9)	06	SJ	166	<< ADD >>
10)	01	PA	<< DELETE detail entry and master entry. >>	
11)	03	SJ	<< DELETE single detail entry. >>	

Solutions: Solution diagrams are provided for various stages of the activity. These diagrams are located in Appendix A.

Module 5-6

Activity 5-1 Worksession: Data Base Access (cont'd)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	:	:	:	:	:	:			CITY- MASTER DATA SET
41	:	:	:	:	:	:			
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	:	:	:	:	:	:			

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS	
161	:	:	- - - - -	RESIDENTIAL DETAIL SET
162	:	:	- - - - -	
163	:	:	- - - - -	
164	:	:	- - - - -	
165	:	:	- - - - -	
166	:	:	- - - - -	

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	:	:	:	:	:	:			CITY- MASTER DATA SET
41	:	:	:	:	:	:			
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	:	:	:	:	:	:			

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS	
161	:	:	- - - - -	RESIDENTIAL DETAIL SET
162	:	:	- - - - -	
163	:	:	- - - - -	
164	:	:	- - - - -	
165	:	:	- - - - -	
166	:	:	- - - - -	

Module 5-7

Activity 5-1 Worksession: Data Base Access (cont'd)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	:	:	:	:	:	:	:	:	CITY- MASTER DATA SET
41	:	:	:	:	:	:	:	:	
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	:	:	:	:	:	:	:	:	

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS	
161	:	:	- - - - -	RESIDENTIAL DETAIL SET
162	:	:	- - - - -	
163	:	:	- - - - -	
164	:	:	- - - - -	
165	:	:	- - - - -	
166	:	:	- - - - -	

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	:	:	:	:	:	:	:	:	CITY- MASTER DATA SET
41	:	:	:	:	:	:	:	:	
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	:	:	:	:	:	:	:	:	

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS	
161	:	:	- - - - -	RESIDENTIAL DETAIL SET
162	:	:	- - - - -	
163	:	:	- - - - -	
164	:	:	- - - - -	
165	:	:	- - - - -	
166	:	:	- - - - -	

Module 5-7 Instructor Notes

Activity 5-1 Worksession: Data Base Access

Time: 30 minutes

Purpose: To implement the placement of entries in a master data set and a detail data set and to manipulate pointers to reflect migrating secondaries and the deletion of entries.

Notes: This activity will reinforce the concepts of calculated access, chained access, and migrating secondaries. This activity can be assigned as homework, if class time needs to be conserved.

Directions:

1. Have students work individually on this activity.
2. Circulate the room to offer assistance to students while they are working on the activity.
3. Solution diagrams are provided after every few steps of the activity. Students should consult these only if the final solution is incorrect. Then they can use the partial solution diagrams to trace their errors. Copies of the solution diagrams are listed below.
4. At the end of the activity, review the solutions and ask the following questions:

How many primary entries exist in the CITY-MASTER set?
Which entries were migrating secondaries? Why did each one migrate?
How many pointers were updated when the last deletion was made?

Break.

Module 5-7 Instructor Notes

□ Activity 5-1 Worksession: Data Base Access (cont'd)

Solution after 1) , 2) , and 3)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	:	:	:	MV	MOUNTAIN VIEW	CITY-MASTER
41	2	43	43	:	:	:	PA	PALO ALTO	DATA SET
42	XXX	XXX	XXX	XXX:	XXX:	XXX:	XX	XXXXXXXXXXXXXX	
43	0	00	00	:	:	:	SJ	SAN JOSE	(MIGRATED FROM 40)

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS	
161	:	:	- - - - -	
162	:	:	- - - - -	RESIDENTIAL DETAIL SET
163	:	:	- - - - -	
164	:	:	- - - - -	
165	:	:	- - - - -	
166	:	:	- - - - -	

Solution after Steps 4) through 9)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	1	165	165	MV	MOUNTAIN VIEW	CITY-MASTER
41	2	43	43	1	161	161	PA	PALO ALTO	DATA SET
42	XXX	XXX	XXX	XXX:	XXX:	XXX:	XX	XXXXXXXXXXXXXX	
43	0	00	00	4	166	162	SJ	SAN JOSE	

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS	
161	00 : 00	01 PA	- - - - -	
162	00 : 163	02 SJ	- - - - -	RESIDENTIAL DETAIL SET
163	162 : 164	03 SJ	- - - - -	
164	163 : 166	04 SJ	- - - - -	
165	00 : 00	05 MV	- - - - -	
166	164 : 00	06 SJ	- - - - -	

Module 5-7 Instructor Notes

Activity 5-1 Worksession: Data Base Access (cont'd)

Solution after 10)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	1	165	165	MV	MOUNTAIN VIEW	CITY-MASTER
41	1	00	00	4	166	162	SJ	SAN JOSE	DATA SET
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	ENTRY MIGRATED TO LOCATION 41								

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS
161	ENTRY DELETED		
162	00 : 163 02	SJ	- - - - -
163	162: 164 03	SJ	- - - - -
164	163: 166 04	SJ	- - - - -
165	00 : 00 05	MV	- - - - -
166	164: 00 06	SJ	- - - - -

RESIDENTIAL DETAIL SET

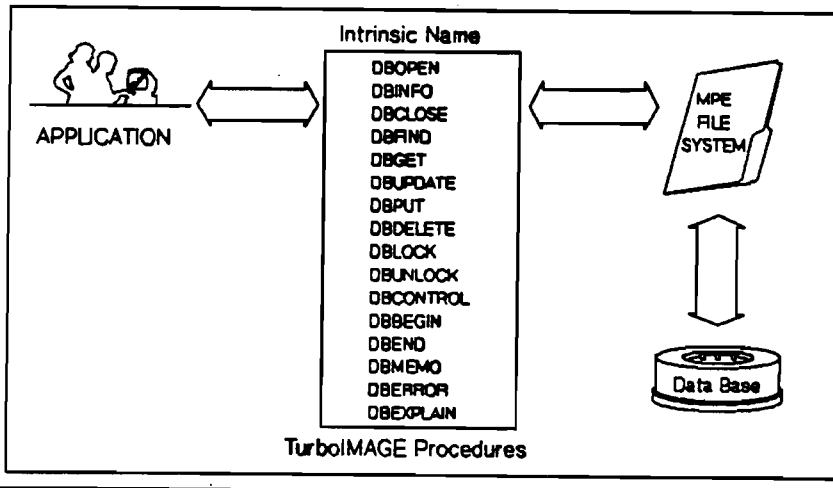
Solution after 11)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	1	165	165	MV	MOUNTAIN VIEW	CITY-MASTER
41	1	00	00	3	166	162	SJ	SAN JOSE	DATA SET
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	ENTRY MIGRATED TO LOCATION 41								

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS
161	DELETED		
162	00 : 164 02	SJ	- - - - -
163	DELETED		
164	162: 166 04	SJ	- - - - -
165	00 : 00 05	MV	- - - - -
166	164: 00 06	SJ	- - - - -

RESIDENTIAL DETAIL SET

Data Manipulation Language



For a description of each TurboIMAGE procedure, refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: Using the TurboIMAGE Library Procedures.

Calling a TurboIMAGE Procedure

- COBOL Call 'name' USING parameter, parameter, ... , parameter
- FORTRAN Call name (parameter, parameter, ... ,parameter)
- SPL Name (parameter, parameter, ... , parameter)
- BASIC Linenumber CALL Xname (parameter, parameter;... , parameter)
- PASCAL Name (parameter, parameter, ... , parameter)
- RPG accesses TurboIMAGE procedures indirectly

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: Using the TurboIMAGE Library Procedures and Section 6: Host Language Access

Module 5-8 Instructor Notes: Slides 5.06/5.07

Data Base Access

Purpose: To present TurboIMAGE's data manipulation language.

Key Points: *Check the end of this module for preparation information.*

- Data base files are privileged files and cannot be accessed directly through the file system without PM capability. TurboIMAGE provides a library of privileged but callable procedures to interface with the file system.
- The procedures are system intrinsics written in SPL. Each intrinsic, with the exceptions of DBERROR and DBEXPLAIN, are assigned an intrinsic number.
- The library of procedures provides the user application the ability to open the data base; read, update, and delete data; and interpret errors.

Purpose: To present syntax for TurboIMAGE procedure calls in various programming languages.

Key Points:

- All parameters are required when a call is made since a parameter's meaning is determined by its position. *even when the parameter is not used in a call.*
- The data types of the parameters vary. Data type declarations for specific languages will be presented on the next page.
- The specific parameters for each TurboIMAGE call will be discussed when the procedure is presented.
- All parameters must be passed by reference. (BASIC allows passing parameters by value.)
- All parameters must begin on word boundaries.
i.e. at levels or 77 levels in COBOL

Data Base Access

Notes

TurboIMAGE Parameters

- **base** Name of the data base
- **dset** Name of the data set
- **mode** An integer
- **status** Communication area
- **dummy** Unused parameter

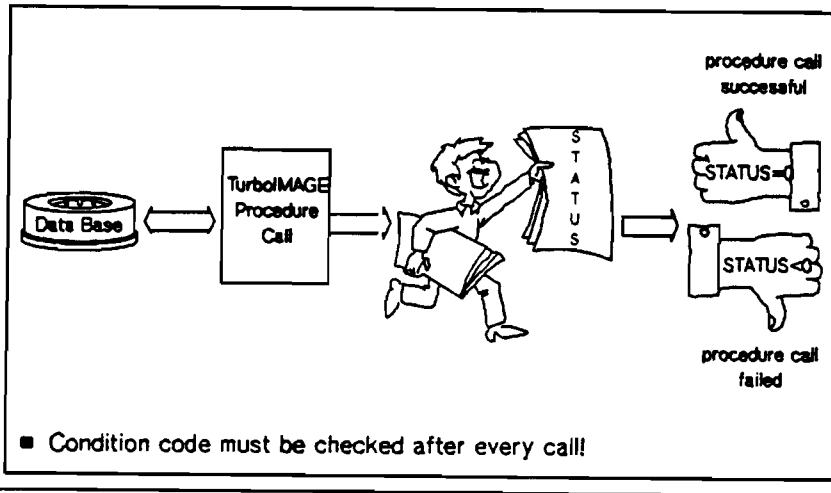
Data Base Access

Purpose: To present the most commonly used TurboIMAGE procedure parameters.

Key Points:

- Most of the TurboIMAGE procedure calls use the base, dset, and mode parameters. All the TurboIMAGE procedure calls use the status parameter.
- The various values of the mode parameter are specified in the description of individual procedures.
- Details about the status parameter are supplied in slide 5.09.
- Since TurboIMAGE uses positional parameters, all parameters must be specified. However, when using some procedures for a specific purpose, one of the parameters may be ignored even though it is listed in the call statement. A programmer may find it useful to set up a DUMMY variable to be used in these situations. This serves as a reminder that the value of the parameter has no effect on the procedure call.
- The other parameters that are used in various TurboIMAGE procedure calls will be listed and explained when the procedure is discussed.

Status Parameter



- STATUS parameter is a ten-word array.
- STATUS parameter is included in every TurboIMAGE call.
- A condition code is returned in the first word of the array.
 - condition code = 0 indicates successful completion of the call.
 - condition code < 0 indicates unsuccessful completion of the call.
 - condition code > 0 indicates an exception other than failure.
- The other information returned in the status array varies with each TurboIMAGE procedure.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Appendix A: Error Messages (Table A-5, Table A-6) or *MPE Software Pocket Guide*, Section IV: Intrinsic Exceptional Conditions

Module 5-10 Instructor Notes: Slide 5.09

Data Base Access

Purpose: To introduce the status parameter.

Key Points: *Check the end of this module for preparation information.*

- The status parameter is used in every TurboIMAGE procedure to return the outcome of the call.
- A program will NOT abort if a TurboIMAGE call fails. It is the responsibility of the programmer to check the status parameter after EACH call and continue appropriately. If the status parameter is not checked, the program will continue to execute unpredictably. It is possible to destroy the data integrity by ignoring the status parameter.
- The status parameter is a 10 word integer array. The condition code is always returned in word one. The use of the other words varies with the individual procedures.
- A successful procedure call will return a condition code of zero. A negative value indicates the procedure failed. A positive value indicates a nonfatal error. The nonfatal errors should be checked programmatically. The most common nonfatal error codes are presented with each procedure.

We check the breakup of the status array, and its uses in non fatal call checking and programming usefulness.

Data Base Access

Parameter Data Types

Refer to *MPE Software Pocket Guide*, Section IV: IMAGE Intrinsic. Note that all parameters are marked A, I, or DI. These letters represent array, integer, and double integer.

Below are examples of the data types for specific programming languages.

ARRAY (CHARACTER)	DATA SET NAME
COBOL	01 DSET-NAME PIC X(16) VALUE 'CITY-MASTER;'
BASIC	DIM D\${16} D\$ = 'CITY-MASTER;'
FORTRAN	DIMENSION DSETA(8) CHARACTER*16 DSETC EQUIVALENCE (DSETA(1),DSETC) DSETC = 'CITY-MASTER;'
PASCAL	DSET_TYPE = PACKED ARRAY[1..16] OF CHAR DSET_NAME:DSET_TYPE DSET_NAME:='CITY-MASTER;'
SPL	ARRAY DSETNAME(0:7) := 'CITY-MASTER;'
ARRAY (INTEGER)	STATUS ARRAY
COBOL	01 STATUS 05 COND-WORD PIC S9(4) COMP. 05 STAT1 PIC S9(4) COMP. 05 STAT2-3 PIC S9(9) COMP. 05 STAT4-5 PIC S9(9) COMP. 05 STAT6-7 PIC S9(9) COMP. 05 STAT8-9 PIC S9(9) COMP.
BASIC	INTEGER S[10]
FORTRAN	DIMENSION STATUS(10)
PASCAL	SINGLE_INTEGER = -32768..32767 STATUS = ARRAY[1..10] OF SINGLE_INTEGER
SPL	INTEGER ARRAY STATUS(0:9)

(Continued on next page.)

Module 5-12

Data Base Access

-----	-----
INTEGER	MODE
-----	-----
COBOL	01 MODE PIC 9999 COMP VALUE 1.
BASIC	INTEGER M M = 1
FORTRAN	MODE = 1
PASCAL	SINGLE_INTEGER = -32768..32767 MODE:SINGLE_INTEGER MODE:=1
SPL	INTEGER MODE:=1
-----	-----
DOUBLE INTEGER	RELATIVE RECORD NUMBER
-----	-----
COBOL	01 REC-NUM PIC 99(9) COMP.
BASIC	INTEGER R[2]
FORTRAN	INTEGER*4 RECNUM
PASCAL	REC_NUM: INTEGER
SPL	DOUBLE RECNUM

Module 5-12 Instructor Notes

Data Base Access

Purpose: To present parameter data types for specific programming languages.

Key Points:

- TurboIMAGE requires parameters to be at word addresses. Therefore, in FORTRAN, character strings are equivalenced to integer arrays.



Data Base Access

Notes

DBOPEN (base, password, mode, status)

Highlight the 2 bits spaces in front & its use.

EXAMPLE

```
base = "  REALTY;"
password = "MANAGER;"
mode = 3
      = 7
condition code → status (1)
user class number → status (2)
```

- DBOPEN initiates access to a data base.

Syntax:
COBOL CALL "DBOPEN"
using BASE,
PASSWORD,MODE,
STATUS.

FORTRAN CALL DBOPEN
(base,password,
mode,status)

SPL or Pascal DBOPEN(base,
password,
mode, status);

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBOPEN or *MPE Software Pocket Guide*, Section IV: IMAGE Intrinsic

Opening a Data Base with RPG

	7	15	19	26	32	53	60
F	CITY	IP	F	24	AM		
F						KIMAGE	REALTY32
F						KLEVEL	MANAGER
F						KDSNAMECITY-MASTER	
F						KSTATUSSTAT	

- KIMAGE spec defines the data base name and open mode
- KLEVEL spec defines the password
- KSTATUS spec defines a 6 element status array

Refer to *TurboIMAGE Data Base Management Reference Manual*, Section 6: RPG Examples or *RPG/3000 Compiler Reference and Applications Manual*, Section IV: Data Base Management

Module 5-13 Instructor Notes: Slides 5.10/5.11

Data Base Access

Purpose: To define and explain the DBOPEN procedure and its parameters.

Key Points: *Check the end of this module for preparation information.*

- The DBOPEN procedure initiates access to the data base. — opens ONLY root files. Sets are opened individually (using MPE FOPEN) the first time.
- The base parameter identifies the data base being accessed. any IO is performed on a set.
- The password parameter determines the user class number as defined in the schema. 8 bytes or 13"
- The mode parameter determines a user's access to the data base.
- The condition code of the DBOPEN call is returned in the first word of the status array. The second word in the status array contains the user class number.

Look up manual and show on board the modes of access and their concurrency rules. briefly only - security is in next module & will discuss in detail there.

Purpose: To present RPG specifications for accessing a TurboIMAGE data base.

Key Points:

- The F specification opens files and the file organization in column 32 indicates a TurboIMAGE file.
- The KSTATUS statement defines the status parameter as a 6 element numeric array with each element containing ten digits. File extension specifications do not need to redefine this array. If the status is not used in RPG, the program will abort when a positive error is encountered.

Data Base Access

Notes

TurboIMAGE and BASIC

- BIMAGE procedures simplify access to data base
- Format same as TurboIMAGE procedures but name preceded by an X
- Certain parameters can be passed by value

Example

10 Call XDBOPEN (base, password, mode, status)

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 6: BASIC Examples

DBINFO (base, ^Aqualifier, ^{I or A}mode, ^Istatus, ^Abuffer)

EXAMPLE

```
base = '___REALTY;'
qualifier = 'RESIDENTIAL;' _____
mode = 201 _____
condition code → status (I) _____
buffer _____
```

- DBINFO returns information about the data base structure.
- Modes of the form:
 - 1nn refer to data item information
 - 2nn refer to data set information
 - 3nn refer to path information
 - 4nn refer to logging
 - 5nn refer to subsystems

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBINFO

Data Base Access

Purpose: To present the BIMAGE procedures used to access a data base with BASIC.

Key Points:

- Access to TurboIMAGE data bases is simplified by using the BIMAGE interface procedures provided with TurboIMAGE.
- These procedures convert all parameter byte addresses to word addresses as required by TurboIMAGE.
- The password and mode parameters can be passed by value.

Purpose: To introduce the DBINFO intrinsic and to explain the necessary parameters.

Key Points:

- The DBINFO procedure returns information about the data base structure.
- The procedure accesses the root file.
- The base parameter must contain the base id returned by the DBOPEN.
- The qualifier parameter is an array containing a data set or data item name. This parameter is related to the mode parameter.
- One common use of DBINFO is to retrieve data item and data set numbers for subsequent TurboIMAGE calls. The number returned corresponds to the order in which the set or item is listed in the schema. The use of numbers instead of names is a more efficient method of passing parameters since TurboIMAGE always converts a name to the corresponding number before the call is accomplished.
- The buffer parameter is an array in which the requested information is returned.

DBERROR (status, buffer, length)

EXAMPLE

status (1) = 16 _____

THE DATA SET IS FULL → buffer

20 → length _____

- DBERROR returns messages that are useful while debugging application programs.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBERROR and Table 5-9: DBERROR Messages

DBEXPLAIN (status)

- Interprets status array
- Outputs multi-line message to \$STDLIST
- Sample output:
IMAGE RESULT AT %001057:CONDITION WORD = 16
DBPUT, MODE 1, ON #1 OF REALTY
THE DATA SET IS FULL
- Sample output:
IMAGE RESULT. CONDITION WORD = 5349
IMAGE CALL INFORMATION NOT AVAILABLE
UNRECOGNIZED CONDITION WORD: 5349
OCTAL DUMP OF STATUS ARRAY FOLLOWS:
012345 054321 011111 022222 033333
044444 055555 066666 077777 000000

- DBEXPLAIN output provides:
 - octal offset in user's code
 - condition code
 - TurboIMAGE procedure name
 - mode of the procedure
 - data set name or number
 - data base name
 - error message (if possible) or octal dump of status array

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBEXPLAIN

Module 5-15 Instructor Notes: Slides 5.14/5.15

□ Data Base Access

Purpose: To define the DBERROR intrinsic and its parameters.

Key Points:

- DBERROR returns messages that are useful while debugging application programs.
- The message returned is a one-sentence message that interprets the contents of the status array.
- TurboIMAGE returns the message to the buffer array parameter.
- The length parameter is set to the positive byte length of the message in the buffer array.
- The example on the slide illustrates the error message returned for an exceptional condition code of 16.

Often handy to use $\begin{cases} \text{DBEXPLAIN} \\ \text{DBERROR} \end{cases}$ for programmer and EDP explanations for program failures. Better to give users a less jargon-full message for things like no key existing for dataset set p.s. eg No key exists on PATH n of n
or Customer not on file.

Purpose: To define the DBEXPLAIN intrinsic and explain its parameters.

Key Points:

- The DBEXPLAIN procedure interprets the status array and outputs a multi-line message to \$STDLIST. The message interprets the results of a TurboIMAGE call.
- The only parameter is the status array. However, the base, qualifier, dset, and password parameters must be unchanged when a call is made to DBEXPLAIN since information is taken from them as well.
- If the status array contents are invalid or incomplete, an octal dump of the status array is displayed.

Call before OUT

explain numbers on set, item, path

Data Base Access

Notes

DBPUT (base, dset, mode, status, list, buffer)

EXAMPLE

```
base= " __REALTY;"
dset= "RESIDENTIAL;"
mode=1 _____
16 → status(1) _____
43 → status(1) _____
1xx → status(1) _____
list= "CITY-ABBR, CITY-NAME, SQ-FT;" _____
      _____
      = "@:" _____
      = "*:" _____
buffer _____
      _____
```

- DBPUT adds new entries to a manual master or a detail data set.
- BIMAGE procedures automatically pack list of BASIC variables or expressions into the buffer array.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBPUT

Adding Entries with RPG

F	7	15	19	26	30	53	60	66
F	CITY	UC	F	24R	4AM			A
F						KIMAGE REALTY3		
F						KLEVEL MANAGER		
F						KDSNAMECITY-MASTER		
F						KITEM CITY-ABBR		

- File addition constructs used
- KDSNAME spec defines the data set name
- No partial lists supported - RPG processes entire IMAGE entries (records) rather than individual items (fields)

Module 5-16 Instructor Notes: Slides 5.16/5.17

□ Data Base Access

Purpose: To define the DBPUT procedure and to explain its parameters.

Key Points: *Check the end of this module for preparation information.*

- The DBPUT procedure adds new entries to a manual master or a detail data set.
- The base parameter is the array used when opening the data base. It must contain the base id returned by the DBOPEN.
- The data set parameter can be 16 characters long. If it is less than 16, characters it must be terminated by a semicolon or a blank.
- The mode parameter is always equal to 1 for this procedure.
- The first word of the status parameter contains the condition code.
- The list parameter is an array containing an ordered set of data item names or numbers.
- The buffer parameter contains the values of the data items specified in the list parameter.

Condition/Code items must be specified in - ST param

IN2 INTERIM (data read)

Purpose: To present the RPG method of accessing data sets.

Key Points:

- RPG processes only entire entries. Therefore, no partial lists are supported. Single items cannot be read or modified using RPG.
- The KDSNAME statement declares the data set name. Only 15 characters are allowed for data set names when accessing through RPG.
- The A in column 66 denotes file addition.
- The KIMAGE statement defines the TurboIMAGE access mode by a blank in column 67. This mode is write or output file.

Data Base Access

Notes

DBCLOSE (^Abase, ^{I or A}dset, ^Imode, ^Astatus)



EXAMPLE

base = '___REALTY;'

dset → Dummy variable;

mode = 1 _____
 = 2 _____
 = 3 _____

condition code → status(1)

- DBCLOSE terminates access to a data base.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBCLOSE

Module 5-17 Instructor Notes: Slide 5.18

Data Base Access

Purpose: To present the DBCLOSE intrinsic and its parameters.

Key Points:

- The DBCLOSE intrinsic terminates access to a data base. The data set parameter is not used by the procedure but is still required.
- Mode 1 terminates access to the data base.
- Mode 2 closes the data set referenced in the 'dset' array.
- Mode 3 re-initializes the data set referenced in the 'dset' array but does not close the set.

ie resets file pointer to record # 1 or "FIRST" of set.

End Tuesday - pre/post Lab?

Module 5-18

□ Activity 5-2a Lab: Accessing a Data Base

Purpose: To gain familiarity with the TurboIMAGE calls: DBOPEN, DBPUT, DBEXPLAIN, and DBCLOSE.

Notes: In this lab you will modify a skeleton program to enable the program to add data to the REALTY data base. This application program utilizes HP's VPLUS/3000 as a terminal screen handler. The required subroutine calls to manage the entry screens have been coded for you. Under user control, you will be able to load either city codes into the master data set or individual property records into the residential detail data set. The REALTY data base files were created in your group in the last part of lab 4-3 in Module 4.

The files used in this lab are:

L5xxx2S.LABS	where xxx = BAS, COB, FTN, PAS, SPL
FASTFORM.LABS	VPLUS fast-compiled forms file
REALTY.groupn	the data base in your group
L5xxx2S.groupn	the completed source code file
L5xxx2P.groupn	the compiled/prepped program file

Directions:

1. Text L5xxx2S.LABS into EDITOR or TDP. (If using BASIC, enter the BASIC interpreter and issue a >GET command to bring in the BASIC program.)
2. Read the instructions at the beginning of the program.
3. Add code to open the REALTY data base for exclusive access. (Use password=";".)
4. Add code to write data to the CITY-MASTER and RESIDENTIAL data sets.
5. Add code to close the data base.
6. Add code to process TurboIMAGE procedure call errors and to check the status of the call after EVERY procedure call. No errors are acceptable for DBOPEN and DBCLOSE. Nonfatal errors are allowed for DBPUT (43, 1xx). Nonfatal errors result in an appropriate message being printed and a retry of input. Fatal errors result in a call to DBEXPLAIN and a programmatic abort.
7. After all code has been added, save the program in your group as L5xxx2S.
8. Compile L5xxx2S. (Refer to Appendix G for compile commands.)
9. When you have a clean compile, prep your program as follows:
:PREP \$OLDPASS,L5xxx2P;MAXDATA=20000
10. Run L5xxx2P.

Solution: The solution can be found in S5xxx2S.SOLUTION. Please refer to this after you have attempted the lab on your own.

Module 5-19

Activity 5-2b Lab: Accessing a Data Base with RPG

Purpose: To gain familiarity with the procedures in RPG needed to add entries to master and detail data sets.

Notes: In this lab you will create a batch file which will contain new records for the CITY-MASTER and RESIDENTIAL data sets. You will then modify a skeleton program to add this data to your data base. The REALTY data base files were created in your group in the last part of lab 4-3 in Module 4.

The files used in this lab are:

LSRPG2S.LABS	skeleton source program
RPGFORMS.LABS	VPLUS forms file used to create the batch file
ENTRY.PUB.SYS	VPLUS program used to create the batch file
REALTY.groupn	the data base in your group
LSRPG2S.groupn	the completed source code file
LSRPG2P.groupn	the compiled/prepped program file

Directions:

1. Run ENTRY.PUB.SYS. Enter RPGFORMS.LABS in response to the request for the name of the forms file. Enter BATCHIN in response to the request for the name of the batch file.
2. Select #1 on the menu. Press the enter key. (VPLUS operates in block mode. Therefore, use the enter key to transmit all screens to the system.)
3. Enter the following data:

ABBREVIATION	NAME
PO	POMONA
SPV	SPRING VALLEY
SF	SUFFERN
NY	NYACK
VC	VALLEY COTTAGE
PO	POMONA
PR	PEARL RIVER

4. After you have entered the previous records, press f6 to return to the main menu. Now select #2 and enter the following 4 records.

	RECORD 1	RECORD 2	RECORD 3	RECORD 4
NUMBER	51	52	53	54
CITY AB	NY	PO	MS	SPV
LIST PR.	140000	75000	32000	104000
# BEDS	6	2	3	4
# BATHS	4.00	1.50	1.00	2.50
OWNER	KEN BAKER	THERESE GARVEY	PAUL DINGMAN	DON SMITH
PHONE #	914-739-8918	914-354-2304	914-249-8901	914-353-5634
ADDRESS	45 RIVERVIEW	386 MT. IVY RD	98 MAIN ST.	12 OAK LANE
ZIP CODE	10945	10970	10952	10958
DINING RM	Y	Y	N	Y
SQ. FEET	2550	1350	1200	2000
SOLD	N	N	N	N

□ Activity 5-2b Lab: Accessing a Data Base with RPG (cont'd)

5. After you have entered the 4 records, press f6 to return to the main menu. Now select #3 to terminate the entry program. You now have a batch file to use with the program you will modify in the next step of this lab.
6. Text L5RPG2S.LABS into EDITOR or TDP. In place of the question marks, substitute the code necessary to make the program execute correctly. (You may use RISE.PUB.SYS to make the changes if you are familiar with that program.)

Although you are only adding records to the CITY-MASTER set, you must open it as an update chained file. You must check for the existence of a master entry before trying to add a record. If you do not and an entry exists with the same key as the record you are trying to add, your program will abort with a duplicate key error. Before adding records to the RESIDENTIAL detail set, you must check for the existence of a CITY-MASTER record with the key value of the record you are trying to add. If you do not and no master entry exists, your program will abort.
7. Keep the corrected program in your group as L5RPG2S.
8. Compile L5RPG2S.
9. When you have a clean compile, prep your program as follows:
:PREP \$OLDPASS,L5RPG2P;MAXDATA=20000
10. Run L5RPG2P. The program should add 7 records to the CITY-MASTER data set and 4 records to the RESIDENTIAL data set.

Solution: The solution to this lab can be found in S5RPG2S.SOLTUION. Please refer to this after you have attempted the lab on your own.

Module 5-20 Instructor Notes

Activity 5-2 Lab: Accessing a Data Base

Time: 45 minutes

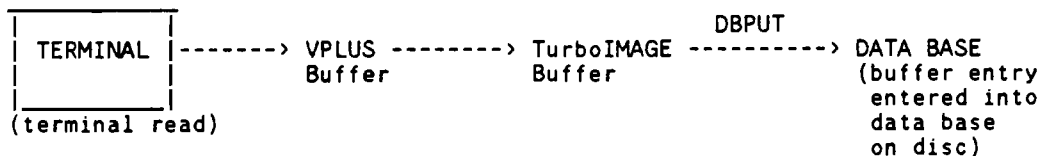
Purpose: To gain familiarity with the procedures to open and close a data base and to add entries to data sets.

Directions:

1. Students are to work in groups of two on this activity. Group the students according to the language they wish to use. If there are non-programmers in the class, group them with students with programming experience.
2. If any students wish to program in RPG, direct them to the separate lab instructions for RPG.
3. The instructions at the beginning of the skeleton program indicate the line numbers where code needs to be added. However, some students may want a hard copy of the lab. Print several copies of the labs and the solutions ahead of time. This will save time for the students when they actually start working on the lab.
4. Inform the students that they will be working on this program throughout Module 5 and Module 6. Also, in Module 7, they will have the opportunity to write an entire application program.
5. Try to group a programmer with a data base administrator. Emphasize that programmers need to concentrate on the syntax of the calls. Data base administrators need only concentrate on what function each call performs, and what the various error conditions are for each call.
6. Caution COBOL programmers NOT to end the DBCLOSE call with a period. A period will result in abnormal termination of the program.

Teaching Tips:

Draw the following diagram on the chalkboard to help students understand how the program interacts with the user and the data base.



Point out that this lab requires the student to handle the DBPUT call and NOT the VPLUS interface.

Data Base Access

Notes

DBFIND (base, dset, mode, status, item, argument)

EXAMPLE

```

base =  "___REALTY;"
dset =  "RESIDENTIAL;"
mode =  1
17 → status(1)  no master set
item =  "CITY-ABBR;"
argument = "SJ"
    
```

- DBFIND locates the detail chain head in the related master set for a subsequent access to a detail chain.
- Refer to slides 5.02 and 5.05 for review of chained and calculated access methods.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBFIND

DBGET (base, dset, mode, status, list, buffer, argument)

EXAMPLE

```

base =  "___REALTY;"
dset =  "RESIDENTIAL;"
mode =  1
2
3
4
5
6
7
8
    
```

- DBGET reads the data items of a specified entry.
- Refer to slide 5.01 for descriptions of the four basic access methods.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBGET

Data Base Access

Purpose: To present the DBFIND intrinsic and to explain its parameters.

Key Points: *Check the end of this module for preparation information.*

- DBFIND locates the detail chain head in the related master for subsequent access to a detail chain. This intrinsic must be called before any chained access to a detail data set.
- A call to DBFIND returns no data to the program.
- The data set parameter identifies the detail data set to be accessed.
- The mode parameter is always 1 for DBFIND.
- A common exceptional condition code is 17. This means that no master entry was found in the master data set.
- The item parameter identifies the search item name or number. If a name is used, 16 characters maximum are allowed.
- The argument parameter specifies the value of the search item contained in the list parameter.

Purpose: To present the DBGET intrinsic and to explain the available modes.

Key Points: *Check the end of this module for preparation information.*

- The DBGET intrinsic reads the data items of a specified entry.
- DBGET allows eight different modes of access.
- The modes of access incorporate the four basic access methods: serial, direct, chained, and calculated.

through each mode

NB DBGET 4 must get the rec addr from a previous DBGET, & use the value in the arg parm

Data Base Access

Notes

DBGET (cont'd)

15 → status(1) end of chain
 17 → status(1) no entry exists
 list = "CITY ABBR,CITY-NAME,SQ-FT;"
 = "@:"
 = "*:"
 = ":"
 buffer _____
 argument (mode 7 or 8)
 (mode 4)
 (mode 1,2,3,5 or 6)

Refer to *TurboIMAGE Data Base Management System Reference Manual, Section 5: DBGET*

RPG Data Retrieval



	7	15	19	25	30	53	60	67
F	RES	IC	F	90R	4AM			
F						KIMAGE REALTY3C		
F						KLEVEL MANAGER		
F						KDSNAMERESIDENTIAL		
F						KITEM	CITY-ABBR	

KIMAGE _____

 KITEM _____

 Multiple access modes _____

 Multiple paths _____

Refer to *TurboIMAGE Data Base Management System Reference Manual, Section 6: RPG Examples or RPG Reference Manual, Section IV: Group Fields, Input/Output Modes, and File Sharing Group Fields*

□ Data Base Access

Purpose: To present the status, list, buffer, and argument parameters of the DBGET intrinsic.

Key Points: *Check the end of this module for preparation information.*

- The status array returns the condition code in word 1. A condition code of 15 indicates the end of chain has been reached. When using DBGET with mode 5 (chained read) it is essential to check for condition code 15 after each call. A condition code of 17 indicates no entry exists.
- The status array also contains a double-word record number of the entry just read, a double-word count of the number of entries in a synonym chain, and double-word pointers to the preceding and next entries in the current chain.
- The list parameter contains a set of data item identifiers, either names or numbers. The list specified is saved as the current list. The current list can be used again by specifying "*" in the list parameter. If all the data items are desired, a list parameter of "@" can be used. A list parameter of ";" specifies a null list. A null list can be used to locate an entry before an update or delete.
- The buffer parameter will receive the values of the data items specified in the list parameter. The values are placed in the array in the same order as specified in the list parameter. If a null list was specified, no data will be transferred.
- If mode 4 is specified, the argument parameter contains a double-word record number of the entry to be read. With modes 7 or 8, the argument parameter contains a search item value. The argument parameter is ignored with modes 1, 2, 3, 5, or 6.
- Since DBOPEN does not open individual data sets, the first call to either DBFIND or DBGET will open the data set.

Purpose: To present RPG access modes.

Key Points:

- The KIMAGE statement defines the access mode (C) of chained forward. The KITEM statement defines the item name. The argument parameter is defined in the CHAIN calculation, which is an RPG statement typically used for accessing KSAM files by key value.
- DBGET modes 2 through 8 can be specified in the KIMAGE statement. Modes 5 and 6 give access to only one detail chain entry. Mode C will give forward chained access and mode R will give backward chained access. A KITEM statement is required for modes 5, 6, C, and R.
- If the same data set is to be accessed in more than one mode, an F specification is required for each. The file names used in the F spec can be equated with a :FILE command or the same KDSNAME statement can be used for each. The first method will cause RPG to open multiple logical files. The second method will open only one. The first method requires more overhead but may be useful if multiple sets of current record pointers are to be maintained for a single data set.
- When accessing a data set by multiple paths, code the KIMAGE statement with dummy data base names, then equate them to the actual data base name with a :FILE command. A separate DBOPEN is executed for each access path to the data base or data set.

Module 5-23

Activity 5-3a Lab: Data Retrieval

Purpose: To gain familiarity with the use of the DBFIND and DBGET intrinsics.

Notes: You have now added data to the data base. This lab will build on the previous program and teach you how to find and retrieve data from various data sets.

The files used in this lab are:

L5xxx3S.LABS	where xxx=BAS,COB,FTN,PAS,SPL
L5xxx3S.groupn	the completed source code file
L5xxx3P.groupn	the compiled/prepped program file

Directions:

1. Text L5xxx3S.LABS into EDITOR or TDP. (If using BASIC, enter the BASIC interpreter and issue a >GET command to bring in the BASIC program.)
2. Add code to GET data from the CITY-MASTER data set.
3. Add code to first FIND and then GET data from the RESIDENTIAL data set using CITY-ABBR as the search key. Use chained access to GET the entries.
4. Check the status after every procedure call. Allowable nonfatal errors are: DBFIND: 17 DBGET : 15,17 All other nonzero errors are fatal.
5. Save the source program in your group as L5xxx3S.
6. Compile L5xxx3S.
7. Prep the program with: PREP \$OLDPASS,L5xxx3P;MAXDATA=20000.
8. Run L5xxx3P.

Solution: The solution to this lab can be found in S5xxx3S.SOLUTION. Please refer to this after you have attempted the lab on your own.

Erick Han (Wednesday)

Module 5-24

Activity 5-3b Lab: Data Retrieval with RPG

Purpose: To gain familiarity with the use of RPG for data retrieval.

Notes: You have now added data to your data base. This lab will teach you how to find and retrieve data from your data base.

Directions:

1. Text L5RPG3S.LABS into EDITOR or TDP.
2. Make the necessary changes.
3. Save the source program in your group as L5RPG3S.
4. Compile L5RPG3S.
5. Prep your program with : PREP \$OLDPASS,L5RPG3P;MAXDATA=20000.
6. Issue the following file equations and run the program:

```
:FILE INPUT=$STDIN  
:FILE OUTPUT=$STDLIST
```

Solution: The solution to this lab can be found in S5RPG3S.SOLUTION. Please refer to this solution after you have attempted the lab on your own.

Module 5-24 Instructor Notes

Activity 5-3 Lab: Data Retrieval

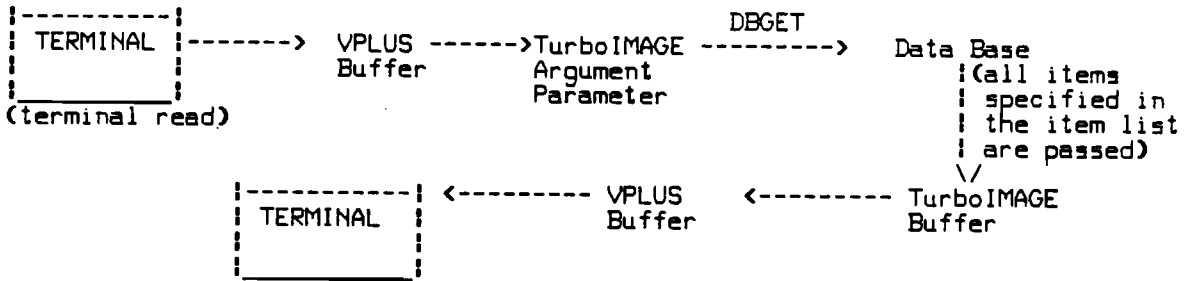
Purpose: To gain familiarity with the use of data retrieval procedures. Time: 45 minutes

Directions:

1. Students are to work in the same groups as in Lab 5-2.
2. There are separate lab instructions for RPG.
3. Circulate the room to offer assistance to students while they are working on the lab.

Teaching Tips:

Draw the following diagram on the chalkboard to help students understand the interaction between the buffers, the argument parameter, the list parameter, and the data base.





Data Base Access

Notes

DBUPDATE (base, dset, mode, status, list, buffer)

EXAMPLE

```

base = '__REALTY;'
DSET = 'RESIDENTIAL;'
mode = 1
17 → status(1) _____
41 → status(1) _____
42 → status(1) _____
list = 'CURRENT-OWNER,SOLD-FLAG;'
     = '@;'
     = '*;'
buffer _____
    
```

- DBUPDATE changes values of data items in the entry residing at the current record address of a specified data set.
- A call to DBUPDATE must be preceded by a call to DBGET.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBUPDATE

DBDELETE (base, dset, mode, status)



EXAMPLE

```

base = '__REALTY;'
dset = 'RESIDENTIAL;'
mode = 1
17 → status(1) _____
44 → status (1) chain not empty (master)
    
```

- DBDELETE deletes the current entry from a manual master or a detail data set.
- A call to DBDELETE must be preceded by a call to DBGET.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBDELETE

□ Data Base Access

Purpose: To present the DBUPDATE intrinsic and to explain its parameters.

Key Points:

- The DBUPDATE intrinsic modifies values of data items in the entry residing at the current record address of a specified data set. A call to DBUPDATE must be preceded by a call to DBGET.
- Search and sort items cannot be updated..
- The data base must be opened in access mode 1,2,3,or 4.
- The mode parameter is always 1 for a DBUPDATE call.
- A condition code of 17 indicates that no entry exists. A code of 41 indicates that a critical item (search or sort item) has been specified and the update cannot take place. A code of 42 indicates that the item has been defined as a read-only item and cannot be modified.
- The list parameter specifies the data items to be updated. Names, numbers, "@;", or "* ;" can be used in the list parameter. The user class established when DBOPEN is called must allow at least read access to all the items in the list parameter.
- The buffer parameter specifies the area containing the new data item values. The order and type of the variables in the buffer parameter must correspond to the list parameter.

Purpose: To present the DBDELETE intrinsic and to explain its parameters.

Key Points:

- The DBDELETE intrinsic deletes the current entry from a manual master or a detail data set. The call to DBDELETE must be preceded by a call to DBGET.
- The data base must be opened in access mode 1,3, or 4.
- The mode parameter is always 1 for DBDELETE.
- If a manual master entry is to be deleted, all chain heads must be empty. If detail entries still exist in the chain, a condition code of 44 will be returned.
- A detail chain is deleted by multiple DBGETs and DBDELETEs.
- An automatic master entry is deleted automatically when all detail chains are deleted.
- A condition code of 17 indicates that no entry exists.

18. INTERNALS OF DELETE ! (update chain head...)

Module 5-26

□ Activity 5-4a Lab: Data Modification

Purpose: To gain familiarity with the use of DBUPDATE and DBDELETE.

Notes: In labs 5-2 and 5-3 you learned the skills required to load and retrieve data. In this lab, you will learn the method of updating and deleting entries from the data base. Upon completion of this lab, you will have a complete, fundamental on-line data entry and retrieval program for use with the REALTY data base.

The files used in this program are:

L5xxx4S.LABS	where xxx=BAS,COB,FTN,PAS,SPL
L5xxx4S.groupn	the ocmpleted source code file
L5xxx4P.groupn	the compiled/prepped program file

Directions:

1. Text L5xxx4S.LABS into EDITOR or TDP. (If using BASIC, enter the BASIC interpreter and issue a >GET command to bring in the BASIC program.)
2. Add code to update and delete entries in the CITY-MASTER data set. The previous DB call must have been a DBGET. The DBGET call was coded in Activity 5-3.
3. Add code to update and delete entries in the RESIDENTIAL data set. You must first find the entry, then read down the detail chain to locate the proper entry.
4. Check the status after every procedure call. Allowable nonfatal errors are:
DBDELETE: 17,44
DBUPDATE: 17,41,42
All other nonzero errors are fatal.
5. Save the source program in your group as L5xxx4S.
6. Compile L5xxx4S.
7. Prep the program with: PREP \$OLDPASS,L5xxx4P;MAXDATA=20000.
8. Run L5xxx4P. When running the application, you must find an entry before updating it or deleting it.

Solution: The solution to this lab can be found in S5xxx4S.SOLUTION. Please refer to this after you have attempted the lab on your own.

Activity 5-4b Lab: Data Modification with RPG

Purpose: To gain familiarity with the use of data modification and deletion using RPG.

Notes: In labs 5-2 and 5-3 you learned the skills necessary to load and retrieve data. In this lab, you will learn the methods of updating and deleting entries from the data base.

This lab consists of two programs. The first program (L5RPG4S) will update selected entries in the RESIDENTIAL data set. The second program (L5XPG4S) will delete entries from the data base based on information you supply as input.

Directions: (Part 1)

1. Text L5RPG4S.LABS into EDITOR or TDP.
2. Make the necessary changes. The instructions in the program indicate which records are to be modified. Make note of this information.
3. Save the source program in your group as L5RPG4S
4. Compile L5RPG4S.
5. Prep the program with: PREP \$OLDPASS,L5RPG4P;MAXDATA=20000.
6. Run QUERY. List all records to be modified to the line printer. Save this listing for future reference.
7. Run L5RPG4P.
8. Run QUERY again and check that the updates have been completed properly.

Directions: (Part 2)

1. Text L5XPG4S.LABS into EDITOR or TDP.
2. Make the necessary changes.
3. Save the source program in your group as L5XPG4S.
4. Compile L5XPG4S.
5. Prep the program with: PREP \$OLDPASS,L5XPG4P;MAXDATA=20000.
6. Run QUERY. List all records in CITY-MASTER, RESIDENTIAL, and COMMERCIAL data sets to the line printer.
7. From the listings obtained above, select city codes to be deleted. Issue the following file equations before running the program:

```
FILE INPUT=$STDIN  
FILE OUTPUT=$STDLIST
```

8. Run QUERY again and check that the deletes have been completed properly.

Solutions: The solution to part 1 of this lab can be found in S5RPG4S.SOLUTION. The solution to part 2 can be found in S5XPG4S.SOLUTION. Please refer to these solutions after you have attempted the labs on your own.

Module 5-27 Instructor Notes

Activity 5-4 Lab: Data Modification

Time: 45 minutes

Purpose: To gain familiarity with the use of data modification and deletion procedures.

Directions:

1. Students are to continue working in the same groups.
2. There are separate lab instructions for RPG.
3. Circulate the room to offer assistance to students while they are working on the lab.

Module 5-28 Instructor Notes

Preparation Material: Slide 5.01

Preparation:

Serial and direct access methods were covered in slide 1.14. Examples of calculated and chained access are presented in slides 5.02-5.05.

The following is a summary of the access methods that can be used with the various types of data sets:

manual master - serial, direct, calculated, chained
automatic master - serial, direct, calculated, chained
detail - serial, direct, chained (chained access does not apply to stand-alone detail sets)

Instruct students to take notes directly on the slide in the student workbook.



Module 5-29 Instructor Notes

Preparation Material: Slide 5.02

Teaching Tips:

The student workbook contains a slide of an empty master data set. You have the same slide on an overhead as well as four slide overlays. Use these overlays to illustrate the process of entries being assigned primary and secondary locations within the set. Tell the students to take notes on the blank slide in the student workbook as you explain the process.

Write on the blank overhead slide if you feel more examples are needed.

The following is a summary of the four overlays:

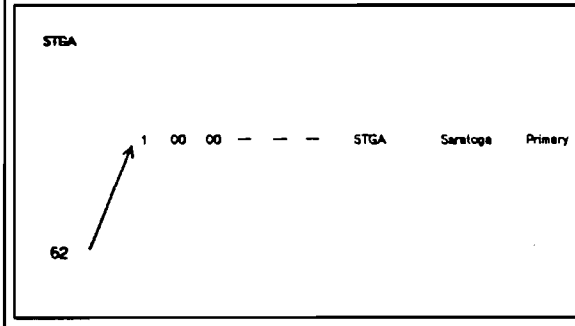
1. The entry with search item value SARATOGA (STGA) has a calculated address of 62. Since this location is empty, the entry is placed in the location as a primary entry. The synonym chain head count is 1, indicating that the entry is a primary with no secondaries. Both pointers are 0.
2. The entry with search item value CAMPBELL (CAMP) also has a calculated address of 62. This entry is placed in the closest empty location, 60, and it becomes part of a synonym chain. The synonym chain head count at location 62 now becomes 2, indicating a synonym chain of length 2. The pointers contain the address of the one secondary entry in the chain. The synonym chain head count for the secondary entry is 0, indicating that the entry is a secondary.
3. The entry with search item value CUPERTINO (CUP) also has a calculated address of 62. This entry becomes another secondary entry in the synonym chain. Now the synonym chain head count field at location 62 is 3, indicating a synonym chain of length 3. The pointer to the last entry in the chain is 66 and the pointer to the first entry in the chain is 60.
4. The entry with search item value SAN JOSE (SJ) has a calculated address of 66. However, location 66 is already occupied by a secondary entry. This situation is resolved by moving the CUPERTINO entry to a new location. This process is called migrating secondaries and is presented in detail in the next slide (5.03).

Copies of the four overlays are listed on the next page.

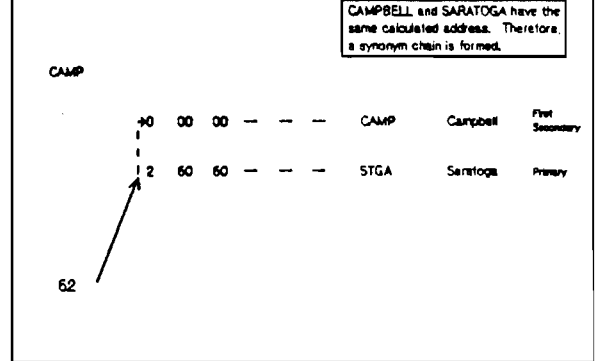
Module 5-30 Instructor Notes

Preparation Material: Slide 5.02 (cont'd)

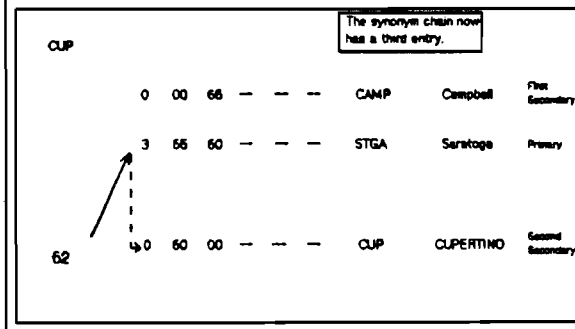
Calculated Access and Synonyms



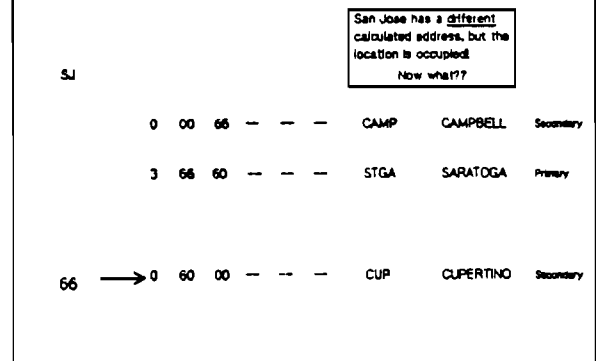
Calculated Access and Synonyms



Calculated Access and Synonyms



Calculated Access and Synonyms



Module 5-31 Instructor Notes

Preparation Material: Slides 5.05/5.06

Teaching Tips:

Present an example of how sorted entries are placed in a detail set. A simple example using LAST NAME, FIRST NAME, MIDDLE NAME will illustrate the process of maintaining entries in sorted order. This example will also highlight the amount of overhead that is necessary to maintain sorted chains. Also, highlight the concept of extended sort fields and point out that if an extended field is not necessary to maintain the desired order, place the sort field at the end of the entry.

Reinforce the necessity of carefully planning for sorted entries. To reduce overhead on implementing sorted chains: (1) make chains short, (2) pick naturally ascending items such as date, or P.O. #, (3) place sort item at end of entry.

Teaching Tips:

The students are now ready to access a data base via an application program. Before discussing this slide, outline on the chalkboard or flipchart the concepts and activities to be presented in the remainder of this module:

1. TurboIMAGE procedures and parameters
2. Opening a data base, adding data, interpreting errors
 - Activity 5-2 Lab: Data Base Access
3. Data retrieval procedures
 - Activity 5-3 Lab: Data Retrieval
4. Data modification procedures
 - Activity 5-4 Lab: Data Modification

Poll the class to determine which programming languages need to be emphasized. There are separate slides on BASIC and RPG. If no one in the class programs in either of these languages, the slides can be skipped.

Module 5-32 Instructor Notes

Preparation Material: Slides 5.09/5.10

Teaching Tips:

Emphasize the necessity of checking the condition code after each TurboIMAGE call. Also, highlight the additional information that can be obtained from the status area. As each of the TurboIMAGE procedures is introduced, highlight the specific information that is returned to the status area. Draw a sample layout of the status area on a flip chart, or refer the students to the reference manual for each individual procedure.

Preparation:

The DBOPEN procedure opens the root file only. Individual data sets are opened with subsequent procedure calls to the specific sets.

The base parameter must begin with two blanks. TurboIMAGE replaces the blanks with a two digit base id after a successful DBOPEN. This entire parameter is then used in other procedure calls. Therefore, the parameter must not be modified or redefined once the DBOPEN succeeds. The base parameter must end with a semicolon or a blank.

If the password is less than eight characters, it must be terminated by a semicolon or a blank. A semicolon alone can be used as a valid password for the creator only. The creator is the user, group, and account that created the root file. The semicolon password gives unlimited access to the creator. Security will be presented in more detail in Module 6.

The mode parameter may or may not allow concurrent access to the data base. Mode 3 allows exclusive modify access and mode 7 allows exclusive read access. Other modes and the allowed concurrent modes will be discussed in Module 6.

Module 5-33 Instructor Notes

Preparation Material: Slide 5.16

Preparation:

If an entry is added to a detail data set, the manual master must first contain the same search item value. The automatic master will be maintained automatically.

The new entry is linked into one chain for each search item, or path, defined according to the search item value. The entry is added to the end of a chain if no sort item is defined. The position of an entry in a sorted chain is determined by a backward search of the chain beginning at the last entry.

The following are common exceptional condition codes that can be returned in the first word of the status array:

- 16 - Data set full
- 43 - Duplicate search item
- 1xx - Missing chain head for path number xx
- 2xx - Full chain for path number xx
- 3xx - Full master for path number xx

If item names are passed in the list array, they must be left justified, separated by commas, and terminated by a semicolon or blank. If item numbers are passed, the parameter must be an integer array. The first word of the integer array must be an integer specifying the number of data items that follow. The item number is determined by the order of its appearance in the ITEMS section of the schema. Once a list is set up for a data set, it is saved and can be used again by specifying "*" for the next list parameter. Using this parameter or the item numbers saves some overhead. If all items are to be specified in the list parameter, a commercial at-sign can be used ("@:"). Any item that is not specified in the list parameter is filled with binary zeros. All search and sort items defined for an entry must be referenced in the list array. The number of words for each value in the buffer parameter must correspond to the number required by its type; for example, X6 must be 3 words long.

Module 5-34 Instructor Notes

Preparation Material: Slides 5.19/5.20

Preparation:

The DBFIND intrinsic sets up pointers to the first and last entries of a detail data set chain. This enables chained access to the data entries in the chain.

A condition code of 17 indicates that no chain exists with the specified search item value. Also, although a master set entry may exist with a specified search item value, the data set chain can be empty.

After a DBFIND call, the status array contains a double-word (IMAGE/3000 allocates two words in the array but uses only one word) count of the number of entries in the chain and two double-word pointers to the first and last entries in the chain. (Refer students to the reference manual for the status array layout for DBFIND.)

The item parameter can be a name or number. If the name is less than 16 characters, it must be terminated by a semicolon or a blank. The specified value of the item parameter defines the path to a detail data set chain.

The argument parameter value is used in calculated access to locate the desired chain head in the master data set.

When a data base is opened with a call to DBOPEN, the individual data sets are not opened. Thus, the first call to DBFIND on a specific data set will open that set. This results in extra overhead as compared to subsequent calls to the same data set.

Preparation:

The methods of accessing a data set are:

- 1 (re-read) The location of an entry just read is stored internally as the current record for the data set. This entry can be re-read by using DBGET with mode 1. The argument parameter is ignored.
- 2 (serial read) Read the next record following the current record. The argument parameter is ignored.
- 3 (backward serial read) Read the entry whose record address is less than the internally maintained current record address.
- 4 (directed read) Read the entry at the record address specified in the argument parameter.
- 5 (chained read) Read the next entry in the current chain. The argument parameter is ignored.
- 6 (backward chained read) Read the previous entry in the current chain. The argument parameter is ignored.
- 7 (calculated read) Read the entry with a search item value that matches the value specified in the argument parameter. The entry is located in the master data set specified in the data set parameter.
- 8 (primary calculated read) Read the entry occupying the primary address of a synonym chain using the search item value specified in the argument parameter. If the entry is not a primary entry, it is not read.

Module 5-35 Instructor Notes

Preparation Material: Slide 5.21

Preparation:

The double-word record address of an entry read by DBGET is returned to words 3 and 4 of the status array. (Refer students to the reference manual to see the status array layout for DBGET.)

Therefore, a DBGET MODE 4 must get the record address from a previous call to DBGET, and move the value into the argument parameter.

Module 6 Instructor Notes

Multiple User Considerations and Security

IMAGE Handbook needed today.
Locking strategy

Overview of Module 6

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. Concurrent Access (45 minutes)

Slides: 6.01 - Concurrent Access
6.02 - Access Path *
6.03 - DBOPEN Modes

Activity: Chalk Talk: Concurrent Access Modes

Purpose: To discuss possible open mode combinations.

Lesson 2. Controlling Concurrent Access (2 hours 5 minutes)

Slides: 6.04 - Controlling Access *

Activity: Demo: Locking

Purpose: To present an example of concurrent access to a data base via an on-line simulation.

Slides: 6.05 - Locking Types
6.06 - DBLOCK *
6.07 - Lock Descriptor Array
6.08 - RPG Locking *
6.09 - DBUNLOCK
6.10 - Data Entry Locking *
6.11 - Concurrent Locking *
6.12 - Logical Transaction *

Activity: Chalk Talk: Locking Strategies

Purpose: To discuss locking strategies.

Activity: Lab: Multiple User Access

Purpose: To gain familiarity with the use of DBLOCK and DBUNLOCK.

Lesson 3. Data Base Security (1 hour 35 minutes)

Slides: 6.13 - Data Base Security *
6.14 - TurboIMAGE Security *
6.15 - Security Flowchart

Activity: Chalk Talk: Set and Item Access

Purpose: To determine data base access in various situations.

Activity: Worksession: Security

Purpose: To apply the concepts of data base security in an on-line interactive worksession.

Activity: Lab: Adding Security Features

Purpose: To implement data set and data item security.



Module 6-1

Multiple User Considerations and Security

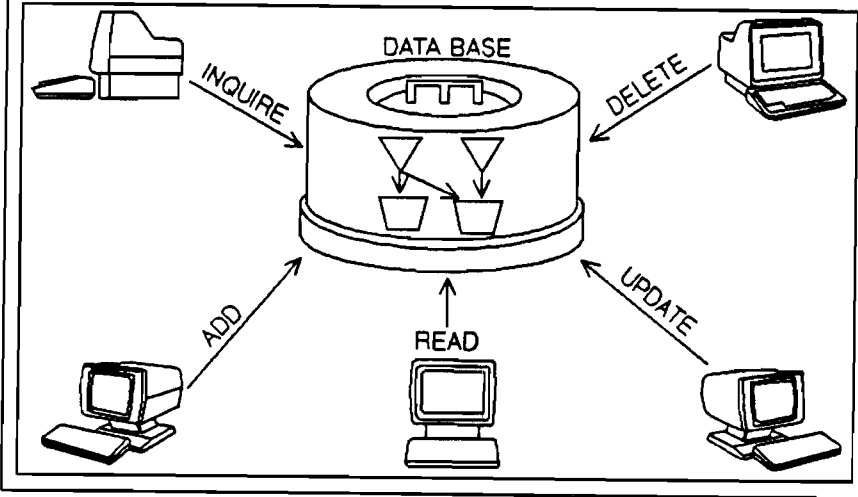
Goal: To present the concepts of concurrent data base access and data base security.

Objectives:

Upon completion of this module, you will be able to:

- define three data base access types.
- give examples of the eight DBOPEN modes.
- define the three locking levels and uses of each.
- define a logical transaction.
- implement the DBLOCK and DBUNLOCK intrinsics.
- state the levels in the hierarchy of security.
- implement TurboIMAGE security via user class numbers.

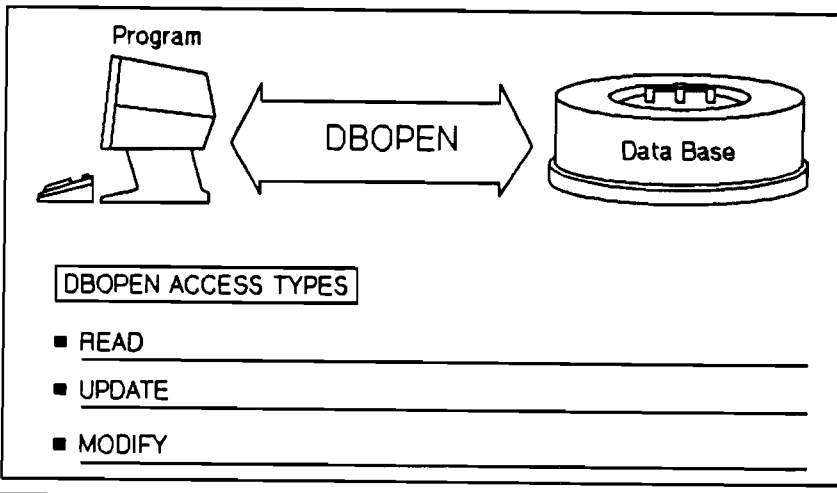
Concurrent Access



T11 6.01

© 1985 Hewlett-Packard Company

Access Path



T11 6.02

© 1985 Hewlett-Packard Company

□ Multiple User Considerations and Security

Purpose: To introduce the concept of concurrent access to a data base.

Key Points:

- Many users can access a centrally controlled data base concurrently via batch or interactive programs.
- Actual contact with the data is restricted to the TurboIMAGE procedures, providing control and security necessary for data base integrity.
- The concept of concurrent access to a TurboIMAGE data base is much the same as for MPE files.
- The data base must be allocated for concurrent or exclusive access at open time.
- If a data base is allocated for concurrent access, it must be controlled to prevent data inconsistencies.
- TurboIMAGE provides its own internal security which allows control of access to many files or subsets of files at one time.

Explain uses of multiple access for concurrent access
example at end of section 6.2

Purpose: To present the types of access available with DBOPEN.

Key Points: *Check the end of this module for preparation information.*

- The DBOPEN procedure forms a path from the user to the data base.
- The mode parameter of the DBOPEN procedure determines the type of access allowed and whether concurrent access is allowed.
- Read access allows the user to locate and read data entries. The procedures DBFIND and DBGET can be used with read access.
- Update access allows the user read access and, in addition, allows the user to change data item values (except search and sort items). The procedures DBFIND, DBGET, and DBUPDATE can be used with update access.
- Modify access allows the user update access and, in addition, allows the user to add and delete entries. Modifying the data base alters the data set structure. The procedures DBFIND, DBGET, DBUPDATE, DBPUT, and DBDELETE can be used with modify access.

DBOPEN Modes



First DBOPEN Mode	Access Type	Additional Concurrent Modes Allowed		
		modify	update	read-only
1	modify	1		5
2	update		2	6
3	modify	none	none	none
4	modify			6
5	read-only	1		5
6	read-only			6
		4	or	2
7	read-only	none		8
8	read-only		none	6
				plus
				8

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 4: Access Modes

- Mode 1 enforces locking.
- Mode 6 allows:
 - one mode 4 user
 - OR
 - multiple mode 2 users
 - OR
 - multiple mode 8 users

Module 6-3 Instructor Notes: Slide 6.03

Multiple User Considerations and Security

Purpose: To present the eight DBOPEN modes and the concurrent modes allowed with each.

Key Points:

- Two of the eight modes (3 and 7) produce exclusive access.
- The six shared modes work in pairs, where one is always a READ-ONLY mode. The possible pairs are: MODIFY plus READ-ONLY, UPDATE plus READ-ONLY, and READ-ONLY plus READ-ONLY.
- Modes 1 and 6 allow other users with the same mode to access the data base concurrently.
- DBOPEN modes should be planned during the design phase of the data base.
- The types of modes used depend on the types of concurrent activity required.

Most commonly see 1 and 5 for update, read/enquiry
in interactive multi-user environment but next page has chance for
examples of each situation

Lunch Wednesday

Module 6-4

Activity 6-1 Chalk Talk: Concurrent Access Modes

Purpose: To discuss possible open mode combinations.

Directions: Several types of data base access are listed below. Write down the DBOPEN modes that apply to each situation. Also list a practical situation that would require each type of data base access. Refer to the Wonder Realty data base or the Parker Manufacturing data base when discussing possible examples. For the last item, summarize the considerations for selecting an appropriate access mode for data base applications.

Concurrent modifiers -

Single modifier, concurrent readers -

No modifiers, multiple updaters -

Read-only (concurrent) -

Exclusive access -

Selecting an access mode -

Module 6-4 Instructor Notes

□ Activity 6-1 Chalk Talk: Concurrent Access Modes

Purpose: To discuss possible open mode combinations.

Time: 30 min.

Notes: Sample responses are supplied for each item below. Use these only if students are not able to supply appropriate examples.

At the end of the discussion, summarize by emphasizing the importance of coordinating concurrent access. The next slide presents the concept of locking.

Concurrent modifiers -

Mode 1 is a modify mode and allows concurrent mode 1 users and concurrent mode 5 (read-only) users. These modes allow the most flexibility and are most commonly used in data base applications.

Example: In the Wonder Realty data base, realtors can add, update, and delete property listings concurrently while others can inquire about the listings using a read-only mode.

Single modifier, concurrent readers -

Mode 4 is a modify mode and allows concurrent mode 6 (read-only) users.

Example: In the Parker Manufacturing data base, only one user is allowed to add, delete, or update the inventory data. However, many users can read the inventory information.

No modifiers, multiple updaters -

Mode 2 is an update mode and allows concurrent mode 2 users and concurrent mode 6 (read-only) users.

Example: In the Wonder Realty data base, an office clerk can update property listing information but cannot add or delete listings. Concurrently the data base information can be read by multiple users. The adds and deletes could then be performed at a later time by a different application.

Read-only (concurrent) -

Mode 6 and mode 8 are read-only modes. Mode 8 only allows concurrent readers. Therefore, a user with this access mode can be assured that the data base values are unchanging.

Example: In the Parker Manufacturing data base, many users may want to read and generate reports on inventory data. Therefore, it would be important that the data base is being accessed in read-only mode.

Exclusive access -

Mode 3 allows exclusive modify access and mode 7 allows exclusive read access. These modes allow no other concurrent access to the data base of any type.

Example: Wonder Realty has a program that allows the sales manager to update crucial information on house listings. So that no sales person reads inaccurate information while the data base is being updated, the program opens the data base in exclusive modify mode.

(Continued on next page.)

Module 6-4 Instructor Notes

Activity 6-1 Chalk Talk: Concurrent Access Modes (cont'd)

Selecting an access mode -

Use the least capability that will accomplish the task. For example, use a read-only mode if the program does not alter the data base in any way.

Allow concurrent users as much capability as is consistent with successful completion of the task. Allowing concurrent read-only access may be appropriate in many situations.

Programs that perform all data base operations should open with mode 1,3, or 4.

Programs that read and update data in existing entries but do not need to add or delete entries, should open the data base in mode 2.

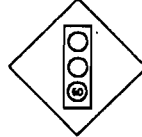
Programs that only locate and read or report on information should open in one of the read-only modes.

It is important that concurrent access modes are discussed during the design phase of a data base.

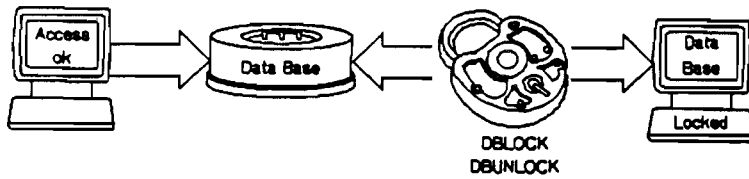
For more detail on selecting access modes, refer to TurboIMAGE Data Base Management System Reference Manual, Section 4: Concurrent Access Modes.



Controlling Access



■ Traffic signs control concurrent access to roadways



■ TurboIMAGE offers a logical signaling facility to control concurrent access

Refer to *TurboIMAGE Data Base Management System Reference Manual, Section 4: Using The Locking Facility*

Multiple User Considerations and Security

Purpose: To introduce the concept of controlling access to a data base via the TurboIMAGE locking facility.

Key Points: *Check the end of this module for preparation information.*

- Locking is a means of communication and control to be used by mutually cooperating users.
- Locking provides a method for protecting the logical integrity of shared data.
- The TurboIMAGE locking facility is a logical not a physical facility.
- TurboIMAGE provides two procedures, DBLOCK and DBUNLOCK, to control concurrent access.
- DBLOCK applies a logical lock to a data entry, a data set, or a data base.
- DBUNLOCK releases locks applied by DBLOCK.

Traffic light analogy

Module 6-6

Activity 6-2 Demo: Locking

Purpose: To present an example of concurrent access to a data base.

Notes: This is an on-line demonstration of two programs accessing the same data entry. Part 1 of the demonstration illustrates the consequences of concurrent access without locking. Part 2 repeats the example with locking. A copy of the demonstration can be sent automatically to the line printer at the end of the session.

Directions:

Type LOCKDEMO to run the session.

Break

Another example of not locking:

Prog 1

Find Chain Head for key

Read all entries in chain

Warning: data error - entry not found

Not true of course.

Prog 2

Find chain head

Read all entries in chain

Delete chain element

Module 6-6 Instructor Notes

Activity 6-2 Demo: Locking

Time: 20 min.

Purpose: To present an example of concurrent access to a data base.

Notes: Be sure to set and release the UDC file ACCUDC.PUB at the account level. The students will initiate this session by typing the UDC :LOCKDEMO.

A copy of this example is in the file LOCKMOD6.SOLUTION. This file will be sent to device class LP upon request by the student at the end of the session. A copy of this file is listed below.

Directions:

Allow the students 10 minutes to observe the demonstration. Discuss the example with the class and answer all questions. Ask the class what would happen if program A used locking but program B did not. Emphasize that locking is logical not physical. Therefore, a locking strategy must be observed by all users accessing the data base.

Present the following additional example of the consequences of ignoring locking procedures. Use the flip chart or chalk board to illustrate the example.

READING WITHOUT LOCKING

Program A performs a DBFIND to locate a detail chain. The program then reads the detail chain using chain DBGETs.

While program A is reading the detail chain, program B performs a DBGET and a DBDELETE on one entry in the chain.

Therefore, an entry is deleted before program A retrieves it. When program A performs the DBGET on the deleted entry, TurboIMAGE returns an error to the program.

READING WITH LOCKING

Before program A performs the DBFIND, a DBLOCK is issued on the detail chain. Then the DBFIND and subsequent chain DBGETs are performed without the possibility of an entry being deleted before it is retrieved. At the end of the chain, a DBUNLOCK is issued to release the lock on the detail chain.

Once the DBLOCK is issued by program A, program B must wait for access to the data. When program A issues the DBUNLOCK, program B then issues its own DBLOCK and proceeds to perform a DBGET, DBDELETE, and DBUNLOCK.

Both programs end without error.

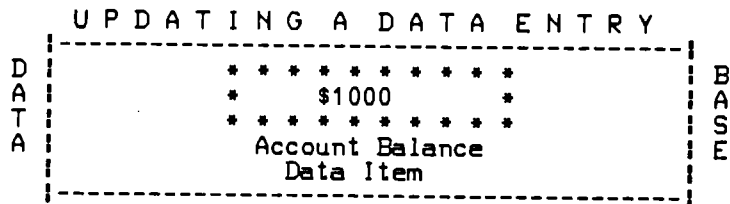
(Continued on next page.)

Module 6-6 Instructor Notes

Activity 6-2 Demo: Locking (cont'd)

Solution:

MODULE 6 - Demonstration: Locking



Two programs access and update this data item representing an account balance. One program adds deposits and the other program adds transfer payments. The sequence of operations performed by the two programs is listed below.

Program A	Program B
(1) DBGET (\$1000)	(2) DBGET (\$1000)
(3) ADD \$100 (\$1100)	(4) ADD \$200 (\$1200)
(5) DBUPDATE - The new value is entered in the data base.	(6) DBUPDATE - The new value is entered in the data base.

The final account balance is \$1200. What should it be?

ANSWER: The final balance should be \$1300. The balance should include the \$100 added by program A and the \$200 added by program B. The final balance is incorrect because the two programs access the same data item without consideration for concurrent access.

This example is repeated below using DBLOCK and DBUNLOCK to control the concurrent access.

Program A	Program B
(1) DBLOCK	(2) DBLOCK (WAITING)
(3) DBGET (\$1000)	(7) DBGET (\$1100)
(4) ADD \$100 (\$1100)	(8) ADD \$200 (\$1300)
(5) DBUPDATE - The new value is entered in the data base.	(9) DBUPDATE - The new value is entered in the data base.
(6) DBUNLOCK	(10) DBUNLOCK

Module 6-6 Instructor Notes

Activity 6-2 Demo: Locking (cont'd)

The final balance is \$1300.

These two examples illustrate the importance of using locking to protect data integrity.

The first example showed two programs getting the same data item in order to update its value. The first program to update the data base had its data item value overlayed by the second program's update. No error occurred and processing continued. However, the value of the data item was incorrect.

The second example showed that if the programs perform a call to DBLOCK prior to accessing the data item, only one program can access the data at a time. The other program must wait for the data item to be unlocked before its lock takes effect.

Multiple User Considerations and Security

Notes

Locking Types

Levels

- Data Base _____

- Data Set _____

- Data Entry (_____) _____

Control

- Conditional _____

- Unconditional _____

Refer to *TurboIMAGE Data Base Management System Reference Manual* Section 4: Using the Locking Facility

DBLOCK ^A (base, ^{A or I} qualifier, ^I mode, ^A status)

Example

```

base = * __REALTY:*
qualifier = _____
mode = 1 (data base) _____
      = 2 (data base) _____
      = 3 (data set)  _____
      = 4 (data set)  _____
      = 5 (data entry) _____
      = 6 (data entry) _____
condition code → status (1)
    
```

- DBLOCK applies a logical lock to a data base, one or more data sets, or one or more data entries.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBLOCK

Module 6-7 Instructor Notes: Slides 6.05/6.06

□ Multiple User Considerations and Security

Purpose: To present the various locking levels and control types.

Key Points:

- TurboIMAGE provides three locking levels: data base, data set, and data entry.
- A data base level lock prohibits other users from accessing any part of the data base.
- A data set level lock prohibits access by other users to the specified data set.
- A data entry lock covers only a specific entry.
- To modify (DBPUT, DBDELETE, or DBUPDATE) a data entry, a successful lock must cover the entry. This can be accomplished by issuing a lock at any of the three available levels.
- To modify a master data set, a data set or data base lock must be issued. A data entry lock will suffice if the master data set is only being updated. *→ original!*
- TurboIMAGE locks can be issued as conditional or unconditional. If an unconditional lock is requested, TurboIMAGE returns control to the calling program only after the lock has been granted. A conditional lock returns immediately. Therefore, the condition code must be examined to determine whether or not the lock has been applied.

Purpose: To present the DBLOCK intrinsic and its parameters.

Key Points: *Check the end of this module for preparation information.*

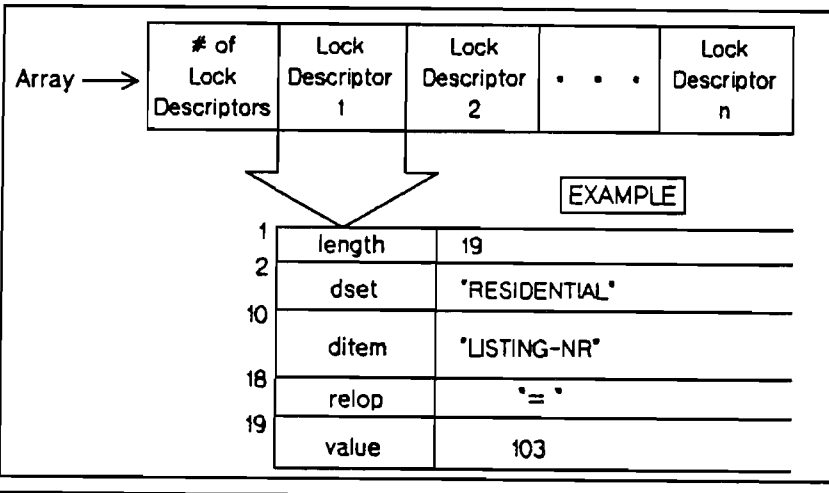
- The DBLOCK intrinsic applies a logical lock to a data base, one or more data sets, or one or more data entries.
- The base parameter is the name of the data base used when opening the data base. The base parameter must contain the base id returned by a call to DBOPEN.
- The value of the qualifier parameter depends on the DBLOCK mode.
- Modes 1, 3, and 5 are unconditional locking modes and modes 2, 4, and 6 are conditional locking modes.
- Modes 1 and 2 apply data base locks; modes 3 and 4 apply data set locks; and modes 5 and 6 apply data entry locks.
- The first word of the status array contains the condition code. The second word contains the number of lock descriptors successfully applied.

*See slide for exceptional condition codes: 20, 22, 23, 24, 25
also Ref. Page 5-47*

Multiple User Considerations and Security

Notes

Lock Descriptor Array



Refer to Appendix B for examples of lock descriptor arrays.

- length - 1 word
- dset - 8 words
- ditem - 8 words
- relop - 1 word
- value - depends on data item definition
- The possible relational operators are =, <=, >=.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBLOCK

RPG Locking

F	7	15	19	26	30	53	60	66
F	RES	IC	F	90R	4AM			
F						KIMAGE	REALTYLC	
F						KLEVL	MANAGER	
F						KDSNAMERESIDENTIAL		
F						KITEM	CITY-ABBR	

- KIMAGE spec defines locking mode

B	10	18	28	33	54	56	58
C		CITKEY	LOCK	RES	7374	75	
C	75	CITKEY	CHAINRES		8058		
C	75	CITKEY	UNLCKRES				76

- Data Entry Locking

RPG Locking Modes

- RPG Automatic
 - B data base for duration
 - S data set for duration
 - 1 data base per record
 - 9 data set per record
 - R record locking
- User-controlled
 - L locking enabled

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 6: RPG Examples or *RPG Reference Manual*, Section IV: Locking Modes. Also, refer to Appendix D for further details on RPG and locking.

□ Multiple User Considerations and Security

Purpose: To present the format of the lock descriptor array.

Key Points:

- The lock descriptor array applies only for locking modes 5 or 6.
- The first word of the array is an integer representing the number of lock descriptors.
- Each descriptor contains a data set name or number, a data item name or number, a relational operator, and a data item value.
- An at-sign (@) used for the data set name indicates the whole data base is to be locked. An at-sign used for the data item name indicates the whole data set is to be locked.
- The possible relational operators are: <=, >=, =.
- The value of the data item must be stored in the array in exactly the same way as it is stored in the data base.
- When using more than one lock descriptor and conditional locking (mode 6), the lock may be only partially successful. The locks are not executed in the order supplied by the user, therefore it is not predictable which locks are held and which are not after an unsuccessful DBLOCK. The second word of the status array indicates how many lock descriptors were successful. It is recommended that a DBUNLOCK be issued before trying the DBLOCK call again.

Get article on locking in IMAGE handbook and read excerpts. Provide copies for whoever wants them

Lock descriptors are coded by TurboIMAGE

Purpose: To present the RPG specifications for locking.

Key Points: *Check the end of this module for preparation information.*

- RPG provides for TurboIMAGE locking in a different manner than other languages. Its locking scheme must be declared when the data base is opened. A single alphanumeric character is placed in column 66 with the KIMAGE statement.
- The RPG locking modes replace the standard TurboIMAGE open mode of 1. All RPG programs accessing the data base must use one of these modes or mode 5 (shared read).
- Modes B and S lock for the entire program execution.
- Modes 1 and 9 lock the base/set whenever a record is accessed, but don't unlock until another record is accessed.
- Mode R locks and unlocks the record when it is accessed. A KITEM statement must be present for this mode.
- Mode L opens the data base with dynamic locking. This allows the program to control locking and unlocking. This mode should be used in a multiple user environment.

Multiple User Considerations and Security

Notes

A A or I I A

DBUNLOCK (base, dset, mode, status)

Example

base = ' __REALTY;'

dset → dummy variable _____

mode = 1 _____

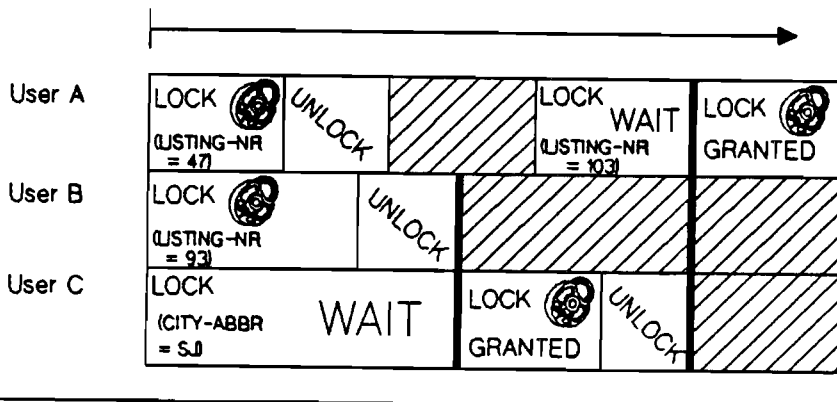
condition code → status (f)

- DBUNLOCK releases all currently held locks on a given access path.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 5: DBUNLOCK

Data Entry Locking

- This slides shows locking in the RESIDENTIAL data set.



- TurboIMAGE allows locking on only one data item per data set at a time.
- TurboIMAGE allows multiple values of the data item to be locked concurrently.

Multiple User Considerations and Security

Purpose: To present the DBUNLOCK intrinsic and its parameters.

Key Points:

- DBUNLOCK releases all currently held locks on a given access path. (An access path is established by each call to DBOPEN.)
- The data set parameter is not used but required by DBUNLOCK. A dummy variable can be used.
- The mode parameter is always equal to 1 for DBUNLOCK.
- The first word of the status array is the condition code. The second word contains the number of lock descriptors released by the call. Each data set lock or data base lock is counted as one descriptor.



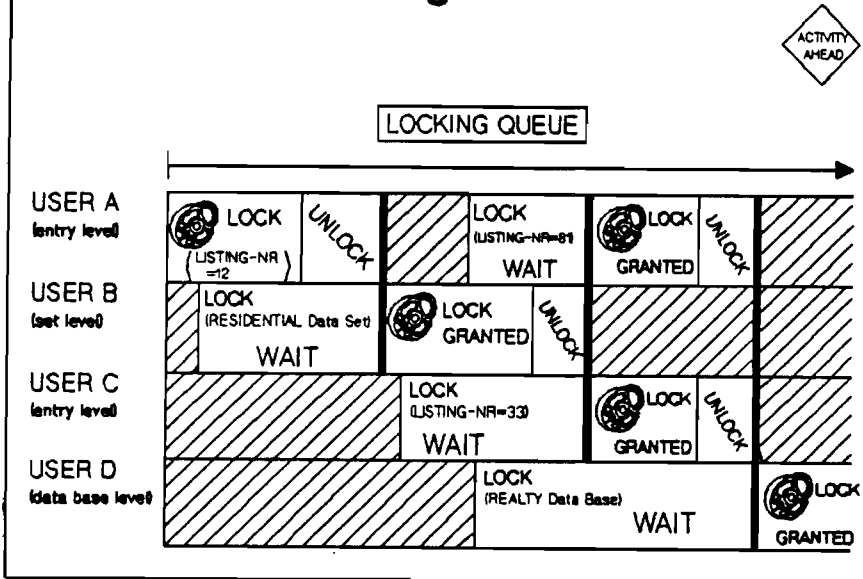
Purpose: To present the guidelines for data entry locking.

Key Points: *Check the end of this module for preparation material.*

- Many users can lock different values of a data item concurrently. A user requesting a lock on a different data item will wait until all previous locks against the data set are released.
- For efficiency, all processes locking data entries in a particular data set should lock using the same data item. This will decrease the probability that one process will have to wait until another process releases its locks.
- When locking entries in detail data sets, use the data item that is the primary path's key.
- When locking entries in master data sets, use the key data item.

... on files & different items

Concurrent Locking

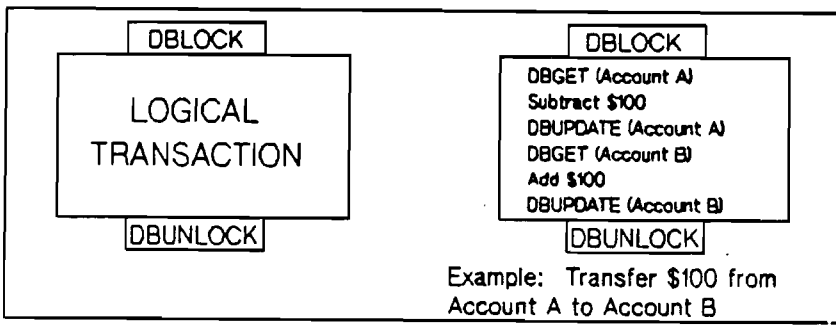


T1 6.11

© 1985 Hewlett-Packard Company

Logical Transaction

- A logical transaction is a basic unit of work performed against a data base.
- Data is logically consistent before and after the transaction, but not while the transaction is in progress.



T1 6.12

© 1985 Hewlett-Packard Company

□ Multiple User Considerations and Security

Purpose: To present an example of a locking queue that implements concurrent locking.

Key Points: *Check the end of this module for preparation information.*

- Concurrent locking at different levels can cause conflict. A data base lock cannot succeed until all locks of all levels are released. A data set lock cannot succeed until all data entries are released.
- All lock requests are handled by a first come, first served locking queue.

Purpose: To present the concept of a logical transaction.

Key Points: *Check the end of this module for preparation information.*

- The logical transaction not only covers calls that change data, but also calls that locate data.
- Locking protects a particular set of data from changes, and also signals other users of the changes.
- A transaction can consist of a single modification, or a set of several calls to TurboIMAGE intrinsics that lock, read, modify, and unlock information.
- Logical transactions transfer the data base from one consistent state to another.
- In the event of system failure and subsequent recovery, only complete logical transactions are re-executed to return the data base to a consistent state.

□ Activity 6-3 Chalk Talk: Locking Strategies

Purpose: To discuss locking strategies.

Notes: It is important to establish a locking strategy at system design time. Locking is related to a logical transaction. At the start of any transaction, locks should be established to cover: (1) all data entries you intend to change, and (2) all data entries which must NOT change during the transaction.

Directions: The following categories represent areas to consider while designing a locking strategy. Fill in the details in each section and relate the concepts to specific data base examples.

I. Locking Levels

TurboIMAGE needs more information to lock data entries than to lock the entire data base. Data entry locking provides the best performance; however, the extra programming effort required should be considered. At system design time, the advantages and disadvantages of low-level locking and extra programming effort should be weighed.

- A. High level locking

- B. Low-level locking
 - 1. Chain DBGET
 - 2. Serial DBGET
 - 3. Directed DBGET
 - 4. Update
 - 5. Add/delete in detail set
 - 6. Add/delete in master set

- C. User Dialogue

II. Multiple Calls to DBLOCK

(Continued on next page.)

Module 6-12

Activity 6-3 Chalk Talk: Locking Strategies (cont'd)

III. Locking Considerations

- A. DBOPEN mode 1 -
- B. DBOPEN mode 2, mode 4, mode 5, or mode 6 -
- C. DBOPEN mode 3 or mode 7 -
- D. DBOPEN mode 8 -

Note write up in TurboImage manual 4-16 on locking & strategies
Note to the article ex IMAGE handbook re locking strategies. read
it and provide copies to whoever wants it.

Talk about MR capability to enable multiple locks on same file set

Break

Module 6-12 Instructor Notes

□ Activity 6-3 Chalk Talk: Locking Strategies

Purpose: To discuss locking strategies.

Time: 20 min.

Directions: Discuss each of the following areas with the students. Instruct them to write the details in their workbooks. Ask students to provide their own examples before presenting the ones listed below.

I. Locking Levels

TurboIMAGE needs more information to lock data entries than to lock the entire data base. Data entry locking provides the best performance; however, the extra programming effort required should be considered. At system design time, the advantages and disadvantages of low-level locking and extra programming effort should be weighed.

A. High level locking

High level locking, modes 1-4, requires less overhead than mode 5 or mode 6. Use high level locking if transactions are brief. This depends on the time a process is holding resources others may need. 7 DBFINDs, DBGETs, or DBUPDATES are approximately equal to 2 DBPUTs or DBDELETES. Approximately 2 modifying calls or 7 non-modifying calls is considered brief. Also, the fewer the number of concurrent users, the less concurrency is a consideration, and higher level locks become more profitable.

B. Low level locking

A long transaction warrants the use of entry level locking. For example, a transaction that involves interactive dialogue can last several minutes. This is an unacceptable amount of time to stop all data base or data set activity. Since the length of transactions will vary, the longest transaction that is used frequently should guide the choice of a locking level.

1. Chain DBGET *Lock all entries in the chain.*

2. Serial DBGET *Lock the data set.*

3. Directed DBGET *This is not recommended in a shared environment. Lock the data set before determining which data entry is needed.*

4. Update *Lock the data entry before calling DBGET to read the data entry. Unlock after the update.*

5. Add/delete in detail set *Lock the data entry using the data item that was designated as the 'lock item' for the data set. No locks are necessary for the implicit additions or deletions that occur in any associated automatic master.*

6. Add/delete in master set *Lock the data set or data base. This is mandatory if the data base is opened with mode 1.*

C. User Dialogue - *Do not lock around user dialogue. Instead, do the following: GET, dialogue, LOCK, RE-READ (to be sure the entry is still there and is unchanged), UPDATE, UNLOCK. If it is necessary to lock around the dialogue, use data entry locking.*

(Continued on next page.)

Module 6-12 Instructor Notes

Activity 6-3 Chalk Talk: Locking Strategies (cont'd)

II. Multiple Calls to DBLOCK

TurboIMAGE does not allow multiple calls to DBLOCK by any process in a session or job unless the program has MR (Multiple RIN) capability. The DBLOCK procedure is similar to an MPE global RIN in that it may put a process into a waiting state and thus, can cause a deadlock to occur. Users whose programs have MR capability and issue multiple DBLOCK calls are responsible for deadlock prevention. Recovery from a deadlock requires a restart of the operating system. TurboIMAGE guarantees that deadlocks will never occur provided that no executing program that accesses the data base has MR capability.

III. Locking Considerations

- A. *DBOPEN mode 1 - TurboIMAGE requires a covering lock for all changes to data. Lock at the set level when modifying a master.*
- B. *DBOPEN mode 2, mode 4, mode 5, or mode 6 - To ensure that no modifications are in progress while reading data from the data base, place a lock on the data before starting. This will coordinate the reading and modifying sequences.*
- C. *DBOPEN mode 3 or mode 7 - These are exclusive access modes. Therefore, locking is unnecessary.*
- D. *DBOPEN mode 8 - This is a concurrent read-only mode. Therefore, locking is unnecessary.*

Module 6-13

Activity 6-4a Lab: Multiple User Access

Purpose: To gain familiarity with the use of DBLOCK and DBUNLOCK.

Notes: If only one user at a time accesses the data base, then the program completed in Lab 5-4 is entirely sufficient. However, in most real life applications the data base is accessed by multiple users simultaneously. To ensure that only one person attempts to modify the data base at one time, locking must be employed.

The files used in this lab are:

L6xxx4S.LABS
FASTFORM.LABS
REALTY.groupn
L6xxx4S.groupn
L6xxx4P.groupn

where xxx = BAS, COB, FTN, PAS, SPL
VPLUS fast-compiled forms file
the data base in your group
the completed source code file
the compiled/prepped program file

Directions:

1. Text L6xxx4S.LABS into EDITOR or TDP. (If using BASIC, enter the BASIC interpreter and issue a >GET command to bring in the BASIC program.)
2. Modify DBOPEN mode to allow simultaneous adds, deletes, updates, and reads.
3. Add data base locking at the data set level for CITY-MASTER and at the data item level for the RESIDENTIAL data set. Lock the RESIDENTIAL data set using CITY-ABBR as the lock item. (Note: You will be adding only a few of the locking calls.)
4. Save the source program in your group as L6xxx4S.
5. Compile L6xxx4S.
6. When you have a clean compile, prep your program as follows:
:PREP \$OLDPASS,L6xxx4P;MAXDATA=20000
7. Select another lab team to work with. Both teams need to log on as the same user and in the same group. Then, both teams run their own programs (:RUN L6xxx4P.groupn).
8. Both teams find the same house in the RESIDENTIAL information screen. Team #1 deletes the house, and then Team #2 tries to delete the same house. If the applications are coded correctly, Team #2 should get a message informing that the entry no longer exists.

Solution: The solution can be found in S6xxx4S.SOLUTION. Please refer to this after you have attempted the lab on your own.

Cobol lab - Check 88.1 - typo error in source code?

Module 6-14

Activity 6-4b Lab: Multiple User Access with RPG

Purpose: To gain familiarity with the use of TurboIMAGE locking with RPG.

Notes: In most real life applications, the data base is accessed by many users simultaneously. To ensure that only one user attempts to modify, add or delete data at one time, locking must be employed. RPG allows you to lock at the data base, data set, or data item level. Locking can apply to the entire RPG cycle or it can be dynamically stated in the calculation specifications. In this lab, you will use locking at the set level for the duration of the RPG cycle. For more information about locking through calculations, refer to Appendix D.

The files used in this lab are:

LAB6RPGS.LABS
REALTY.groupn
LAB6RPGS.groupn
LAB6RPGP.groupn

Skeleton source program
the data base in your group
the completed source code file
the compiled/prepped program file

Directions:

1. Text L6RPG4S.LABS into EDITOR or TDP.
2. Change the code for method of access.
3. Save the program in your group as LAB6RPGS.
4. Compile LAB6RPGS.
5. When you have a clean compile, prep your program as follows:
:PREP \$OLDPASS,LAB6RPGP;MAXDATA=20000
6. Issue the following file equations:
:FILE INPUT=\$STDIN
:FILE OUTPUT=\$STDLIST
7. Run LAB6RPGP.

Solution: The solution to this lab can be found in S6RPG4S SOLUTION. Please refer to this solution after you have attempted the lab on your own.

Module 6-14 Instructor Notes

Activity 6-4 Lab: Multiple User Access

Purpose: To gain familiarity with the use of TurboIMAGE locking.

Time: 45 min.

Notes: At this time, the REALTY data base the students are working with does not have passwords defined. Therefore, the data base can only be accessed by the creator. Students are instructed to log on to the same user and group and then run two programs that access the same data base.

Directions:

1. Have students work in groups of 2.
2. There are separate lab instructions for RPG.

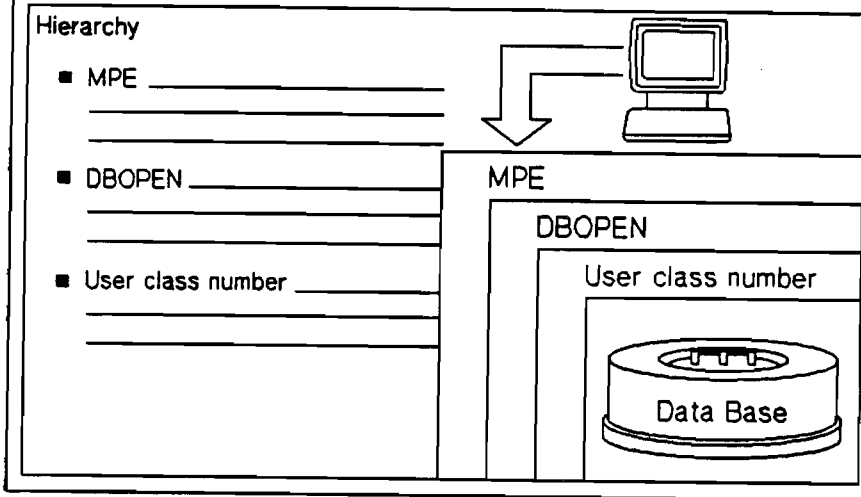


Multiple User Considerations and Security

Notes

Data Base Security

Refer to *TurboIMAGE Data Base Management System Reference Manual, Section 2: Protection of the Data Base*



T11 6.13

© 1985 Hewlett-Packard Company

TurboIMAGE Security

BEGIN DATA BASE SAMPLE.

PASSWORDS:

10	PASSWD1:
20	PASSWD2:
30	PASSWD3:

ITEMS:

ITEM1	I	(10,20/30)
ITEM2	X2	(10/20,30)
ITEM3	X4	(20/30)
ITEM4	I	(/20,30)

SETS:

NAME: SET1, MANUAL (10,20/30)

ENTRY: ITEM1 (I)

CAPACITY: 100;

NAME: SET2, AUTOMATIC (10/20,30)

ENTRY: ITEM2 (X)

CAPACITY: 200;

NAME: SET3, DETAIL (10,20/30)

ENTRY: ITEM1 (SET1), ITEM2 (SET2), ITEM3, ITEM4;

CAPACITY: 400;

END.

■ User class numbers

■ Read/write class lists

■ Set level _____

■ Item level _____

- (/) is called a null list and allows no access to the set, except the creator.
- The absence of a read/write list, and the parentheses and slash, allows read access to all user classes and write access to the creator only.

Refer to *TurboIMAGE Data Base Management System Reference Manual, Section 2: User Classes and Passwords*

T11 6.14

© 1985 Hewlett-Packard Company

Module 6-15 Instructor Notes: Slides 6.13/6.14

Multiple User Considerations and Security

Purpose: To present the hierarchy of data base security.

Key Points: *Check the end of this module for preparation information.*

- The purpose of data base security is to verify a user's right to use the data base.
- MPE and TurboIMAGE work together to provide privacy, security, and availability of data.
- MPE security includes logon password protection, file access restrictions, and privileged file protection.
- The DBOPEN procedure requires a password parameter that establishes the user class number. DBOPEN also requires a mode parameter that specifies modify, update, or read-only access.
- TurboIMAGE provides user class numbers that are specified at set and item levels. These user class numbers indicate modify, update, or read-only access to sets and items.

Store / Restore ASCII readable tapes including database files ∴ ARCHIVE security
SM,OP access to unauthorized users.
PM screws up the lot

Purpose: To present TurboIMAGE security as provided through user class numbers and read/write class lists.

Key Points: *Check the end of this module for preparation information.*

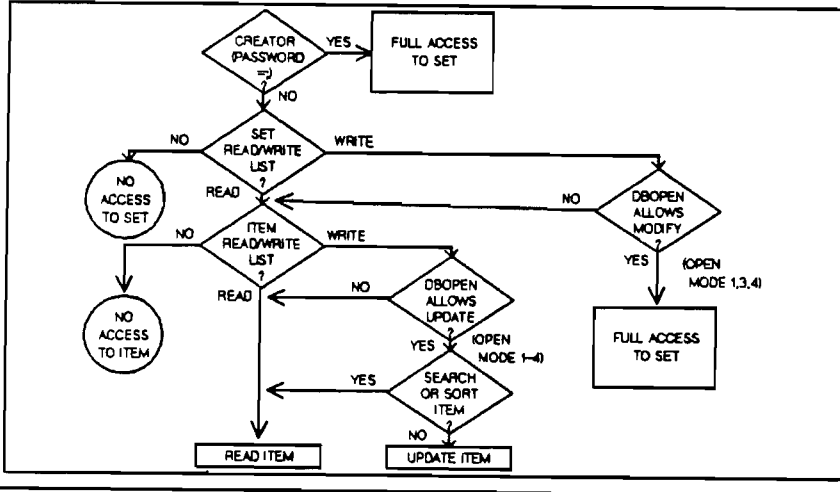
- TurboIMAGE allows the data base designer to control access to specific data sets and data items by defining up to 63 user classes and then associating the user classes with data sets and data items in read or write class lists.
- Each user class is associated with a password defined by the data base designer.
- Set level read/write class list grants permission to move on to the item level security.
- Item level security indicates read access or update access to individual data items. Search and sort items cannot be updated.

Creator - user class 04 and no password - '5'

Check out changes

passwords can be changed using SM -

Security Flowchart



Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 2: Granting a User Class Access to a Data Element.

Multiple User Considerations and Security

Purpose: To present a flowchart of the TurboIMAGE security checkpoints.

Key Points:

- If the user is the creator of the data base, all security checking is overridden and full access to all sets and items is provided.
- If a user needs to modify the data base, the data base must be opened in mode 1, 3, or 4 and the user's user class number must appear in the write class list at the set level.
- Write access at the set level allows full access to the data set within the constraints of the DBOPEN mode. Read access at the set level causes item level security to be checked. The indicated item level access type will be allowed within the constraints of the DBOPEN mode.
- Search and sort items cannot be updated.
- TurboIMAGE security is designed so that the data base designer must explicitly specify the user class number to allow a user to change the data base. Read access can be granted by default and can be denied by omitting the user class number from a read/write class list.

Module 6-17

Activity 6-5 Chalk Talk: Set and Item Access

Purpose: To determine data base access in various situations.

Notes: The type of data base access that is granted to a particular user depends on the DBOPEN mode and the user class number established from the password specified.

Directions: Refer to the schema on slide 6.14 and the security flowchart on slide 6.15. Each of the situations below specify the DBOPEN mode and the password. Determine the resulting access granted the user at the set level and the item level. Specify the access type as read, update, or modify.

SITUATION 1: DBOPEN mode = 5, password = PASSWD1

Resulting access: SET1 _____ ITEM1 _____ ITEM3 _____
SET2 _____ ITEM2 _____ ITEM4 _____
SET3 _____

SITUATION 2: DBOPEN mode = 1, password = PASSWD2

Resulting access: SET1 _____ ITEM1 _____ ITEM3 _____
SET2 _____ ITEM2 _____ ITEM4 _____
SET3 _____

SITUATION 3: DBOPEN mode = 1, password = PASSWD3

Resulting access: SET1 _____ ITEM1 _____ ITEM3 _____
SET2 _____ ITEM2 _____ ITEM4 _____
SET3 _____

SITUATION 4: DBOPEN mode = 5, password = PASSWD3

Resulting access: SET1 _____ ITEM1 _____ ITEM3 _____
SET2 _____ ITEM2 _____ ITEM4 _____
SET3 _____

SITUATION 5: DBOPEN mode = 2, password = PASSWD2

Resulting access: SET1 _____ ITEM1 _____ ITEM3 _____
SET2 _____ ITEM2 _____ ITEM4 _____
SET3 _____

SITUATION 6: Use the schema structure below for this section.

PASSWORDS: 40 PASSWDA; DBOPEN mode = 3, password = PASSWDA

ITEMS: SETA _____ ITEMSA _____
ITEMA 1 ; ITEMB _____
ITEMB X2 (/); ITEMC _____
ITEMC 1 (/40);

SETS: NAME: SETA, MANUAL;
ENTRY: ITEMSA (1),
ITEMB,
ITEMC;
CAPACITY: 200;

Module 6-17 Instructor Notes

□ Activity 6-5 Chalk Talk: Set and Item Access

Time: 30 min.

Purpose: To determine data base access in various situations.

Directions: Discuss each situation with the class. Review with the class the meanings of each type of access (read, update, modify). Call on individual students to supply answers in each situation, and to explain how the answers were determined.

SITUATION 1: DBOPEN mode = 5, password = PASSWD1

Resulting access: SET1 read ITEM1 read ITEM3 no access
SET2 read ITEM2 read ITEM4 no access
SET3 read

SITUATION 2: DBOPEN mode = 1, password = PASSWD2

Resulting access: SET1 read ITEM1 read ITEM3 read
SET2 modify ITEM2 update*ITEM4 update
SET3 read

SITUATION 3: DBOPEN mode = 1, password = PASSWD3

Resulting access: SET1 modify ITEM1 update ITEM3 update
SET2 modify ITEM2 update*ITEM4 update
SET3 modify

SITUATION 4: DBOPEN mode = 5, password = PASSWD3

Resulting access: SET1 read ITEM1 read ITEM3 read
SET2 read ITEM2 read ITEM4 read
SET3 read

SITUATION 5: DBOPEN mode = 2, password = PASSWD2

Resulting access: SET1 read ITEM1 read ITEM3 read
SET2 read ITEM2 update*ITEM4 update
SET3 read

* ITEM2 is defined as a search item value for this example. Therefore, only read access would be allowed.

SITUATION 6: Use the schema structure below for this section.

PASSWORDS: DBOPEN mode = 3, password = PASSWDA
40 PASSWDA;

ITEMS: SETA read ITEMA read
ITEMA I ; ITEMB no access
ITEMB X2 (/); ITEMC update
ITEMC I (/40);

SETS:
NAME: SETA, MANUAL;
ENTRY: ITEMA (1),
ITEMB,
ITEMC;
CAPACITY: 200;

End Worksheet

Module 6-18

Activity 6-6 Worksession: Security

Purpose: To apply the concepts of data base security in an on-line interactive worksession.

Notes: This is an on-line worksession. You will be provided with a sample set of user class numbers and two sample data sets. You must determine the access allowed to each user class at both the set and the item levels. A copy of the worksession can be sent automatically to the line printer at the end of the session.

Directions:

1. Type :SECURITY to activate the session.
2. Read the instructions at the beginning of the activity.
3. Complete the activity individually or with your lab partner.
4. Refer to the security flowchart on slide 6.15 if you have difficulty completing the exercise.

Module 6-18 Instructor Notes

Activity 6-6 Worksession: Security

Time: 20 mins.

Purpose: To apply the concepts of data base security in an on-line interactive worksession.

Directions:

1. Be sure to set and release the UDC file ACCUDC.PUB at the account level. The students will initiate this session by typing the UDC :SECURITY.
2. A copy of this worksession is in the file SECMOD6.SOLUTION. Copies of this file will be sent to device class LP upon request by the student at the end of the session. A copy of this file is listed below.

Solutions:

MODULE 6 - SECURITY EXERCISE

```

-----
          SECURITY EXERCISE
-----
PASSWORDS:          ACCESS TYPES:
  2 two;   6 six;   R - Read   M - Modify
  4 four;  8 eight; U - Update N - No access
-----
  
```

The data base was opened in Mode 3 (exclusive access). State the access allowed each user to the following sets and items. Enter R, U, M, or N in each box and press RETURN after each entry. Press RETURN now to display the first exercise.

Exercise #1	0	2	4	6	8	CR(creator)
Master DSETA (6,8/2,4)	N	M	M	R	R	M
ITEMA1 (/2,4)	N	U	U	N	N	U
KEYITEM (2/8)	N	R	R	N	R	R

Exercise #2	0	2	4	6	8	CR(creator)
Master DSETB (0,2,6/8)	R	R	N	R	M	M
KEYITEM(0,2,6,8) ITEMB1 (4,6,8)	N	N	N	R	R	R
ITEMB1(4,6/) KEYITEM (0,2,6,8)	R	R	N	U	U	U

← KEY ITEM (4,6)
ITEMB1

(Continue on next page.)

Module 6-18 Instructor Notes

Activity 6-6 Worksession: Security (cont'd)

Notes:

User class 0 is assigned when no password or an incorrect password is entered. No access is granted to user class 0 unless 0 is specifically listed in a read/write class list.

Modify access allows a user to add or delete entries. Therefore, modify access is granted at the set level only. The data base must be opened in mode 1, 3, or 4.

Update access allows a user to change the value of a data item. Therefore, update access is granted at the item level only.

Even though a user class has write access at the data item level, a KEY item cannot be updated.

If a user class number is listed in the write list at the set level, the user gains full access to the set. The item level security is bypassed. Likewise, if a user class number is not listed in the read/write class list at the set level, the user gains no access to the set. Again, the item level security is bypassed.

The creator of the data base gains full access to the data base by entering a password of ";". The set level and item level security is bypassed.

Module 6-20

Activity 6-7 Lab: Adding Security Features (cont'd)

8. Copy S6xxx7S.SOLUTION to your group: FCOPY FROM=S6xxx7S.SOLUTION;TO=S6xxx7S;NEW
9. Compile, prep, and run S6xxx7S.
10. You will be prompted for passwords for DBOPEN. Start with the manager password. Then rerun the program using the salesrep password and again with the receptionist password. Observe the differences in the access allowed in each situation. Did the receptionist get the owner's name, address, and phone? Were the salesrep and receptionist able to use the add, delete, or update functions?

Solution: The solution to this lab can be found in Appendix A. Please refer to this after you have attempted the lab on your own.

Break

Module 6-20 Instructor Notes

Activity 6-7 Lab: Adding Security Features

Purpose: To implement data set and data item security.

Time: 30 min.

Directions:

1. This lab is not available in RPG. The password is hard coded in the source program.
2. Have students work in groups of 2.

Module 6-21 Instructor Notes

Preparation Material: Slides 6.02, 6.04

Teaching Tips:

Write on the slide the TurboIMAGE procedures that can be used with each type of access. Emphasize that a call to DBOPEN provides ONE path from the user to the data base. Discuss the following example of the use of TWO access paths to a data base within one application.

A data base contains a SALES detail set and an INVENTORY detail set. An application program does a DBFIND and chain DBGETs on a particular customer account. Each record for this customer account number contains a product number. To find the inventory information on the product number, a DBFIND and DBGET must be performed for the INVENTORY detail set. Therefore, a second DBOPEN is issued to open a second path. After the inventory information is retrieved, the pointers in the SALES detail set from the first access path still exist and the chained read can continue where it left off.

Teaching Tips:

Use the traffic signal analogy to explain the concept of locking. Point out that traffic signals do not physically prevent collisions at intersections. A driver may choose to ignore the traffic signals and therefore suffer undesirable consequences. The same is true with data base locking. All users of a data base must observe some established locking strategy in order to protect the integrity of the data.

An on-line example follows this slide. The example demonstrates the consequences of ignoring concurrent access while updating a data item.

Module 6-22 Instructor Notes

□ Preparation Material: Slides 6.06, 6.08

Preparation:

The value of the qualifier parameter is ignored with modes 1 and 2. If mode 3 or 4 is used, the qualifier parameter contains the name or number of a data set. If mode 5 or 6 is used, the qualifier parameter is an array containing lock descriptors. The format of this array is presented in slide 6.07.

The first word of the status array returns the condition code. The following exceptional condition codes can appear:

20	Data base locked or contains locks	(modes 2,4,6)
22	Data set locked by another process	(modes 4,6)
23	Entries locked within set	(mode 4)
24	Item conflicts with current locks	(mode 6)
25	Entry or entries already locked	(mode 6)

Preparation:

In the data entry locking example on the slide, factor 1 is the key value and is used for data entry locking. Factor 2 is the FILE NAME of the data set for entry or set locking. The high indicator specifies conditional locking and is ON if the lock cannot be acquired. The low indicator is ON if the lock fails. The equal indicator is ON if the lock succeeds. If data base locking is done, the number of characters in the data base name must be specified in the result field length and the DATA BASE name in factor 2.

Refer to Appendix D for an article on RPG and locking.

Module 6-23 Instructor Notes

□ Preparation Material: Slides 6.10/6.11

Teaching Tips:

Use the example below to explain why locking LISTING-NR=47 and LISTING-NR=93 concurrently is allowed, but locking LISTING-NR=47 and CITY-ABBR=SJ is not allowed.

RESIDENTIAL DATA SET:

LISTING-NR	CITY-ABBR	NUM-BEDS . . .
10	SJ	2
11	LA	3
12	SJ	2
13	CUP	4

If a lock is issued on LISTING-NR=10 then that particular entry is locked. A concurrent request for a lock on CITY-ABBR=SJ would require that two entries be locked, one of which is already locked. Because of this potential conflict, TurboIMAGE allows locking on only one data item per data set at any given time. However, a concurrent request for a lock on LISTING-NR=11 would be allowed since no conflict would occur. It is recommended that one item per data set be designated as the LOCK ITEM. Comments can be added to the schema to indicate which item in each data set is designated as the LOCK ITEM.

Teaching Tips:

The slide illustrates a locking queue where lock requests are queued as a function of time. Trace the lock requests from left to right. Emphasize the following points:

- User A locks at the entry level (LISTING-NR=12)
- User B requests a lock on the RESIDENTIAL data set. The request is placed in the queue and the lock is granted when user A's lock is released.
- User A and user C request locks at the entry level using the same data item. Since the RESIDENTIAL data set is locked, these requests must wait. Both locks can exist concurrently and are granted as soon as the data set is unlocked.
- User D requests a lock on the REALTY data base. This lock must wait until all locks at all levels are released.

Module 6-24 Instructor Notes

Preparation Material: Slides 6.12/6.13

Teaching Tips:

Use the example on the slide to emphasize the two main points about logical transactions. Ask the students if the data base is logically consistent at the end of the DBUPDATE to Account A.

Stress the importance of identifying logical transactions in relationship to system failure and data base recovery. This will be discussed further in Module 8.

Preparation:

All TurboIMAGE files are privileged files. Therefore, no standard MPE command can access the data base files. In order to gain access to the data base, a user must have access to the group and account in which the data base resides.

The following are common external security considerations:

1. Store and sysdump tapes include all file labels including the maintenance word and all passwords of a TurboIMAGE data base. Sysdump also includes the system directory which contains all user, account, and group passwords. FCOPY can display these ASCII fields, and requires no special capability except access to tapes. Therefore, tapes should be archived securely.
2. MPE RESTORE can accidentally overwrite one or more TurboIMAGE files, thus making the data base inconsistent. The worst case is when the RESTORE has not been detected and processing has commenced on valid data. Care must be taken to control any tapes that may potentially restore data base files.
3. System managers distribute special user capabilities such as SM, AM, OP, and PM. OP capability allows the user to perform STORE or SYSDUMP on non-read access files such as data base files. PM capability allows the user to run privileged mode programs which override all security including privileged file codes.

Module 6-25 Instructor Notes

Preparation Material: Slide 6.14

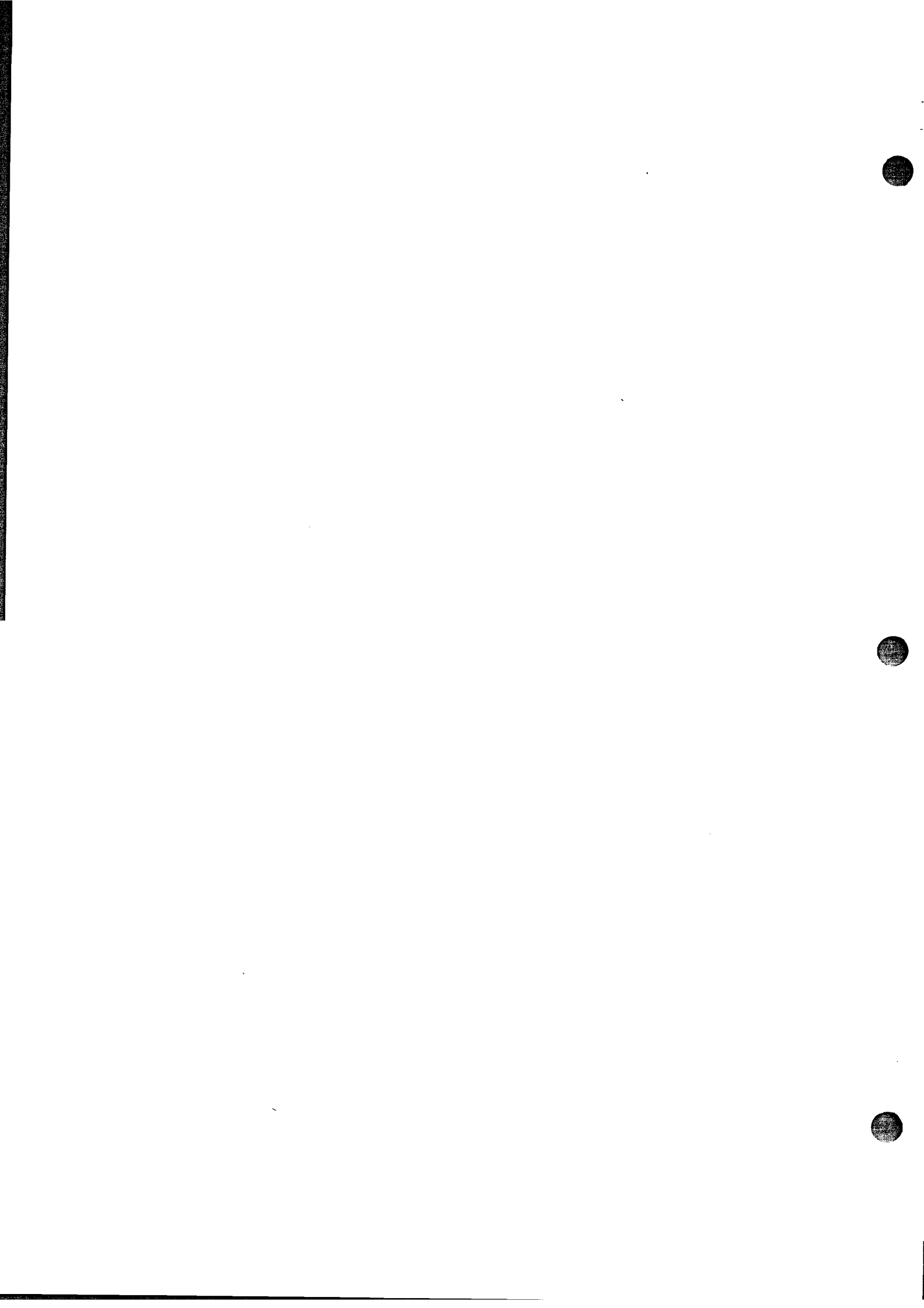
Preparation:

The data base creator is the user name and group that was used to create the root file. The creator (with password = :) is assigned user class number 64. This user class overrides all security. A missing or incorrect password results in the assignment of user class number 0. This allows the user no access to the data base unless user class 0 is specifically designated in the read/write class lists at the set or item level.

If a user's user class number is not on a read/write class list, then no access is allowed. A user class whose number is listed in the write class list implicitly has read access also. The absence of a read/write class list yields the same result as a read list containing all user class numbers and an empty write class list. A null list can be either of the following:

- (/) Both read and write class lists are null. No access is allowed at the set or item level (except creator).
- (10,30/) The write class list is null. Only read access is allowed for user class 10 and user class 30.

Passwords can be changed using DBUTIL (further details in Module 8) It is not necessary to re-create the data base. Maintenance words can be assigned to the data base using DBUTIL (further details in Module 8).



Module 7 Instructor Notes

Review of Data Base Applications

Overview of Module 7

* - indicates preparation and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. Data Base Applications (1 hour 45 minutes)

Slide: 7.01 - Components of a Data Base Application *

Activity: Lab: Writing a Data Base Application

Purpose: To write a data base application.

Activity: Optional Exercise

Purpose: To solve an application design problem involving master data sets.



Module 7-1

Review of Data Base Applications

Goal: To review the basic components of a data base application.

Objectives:

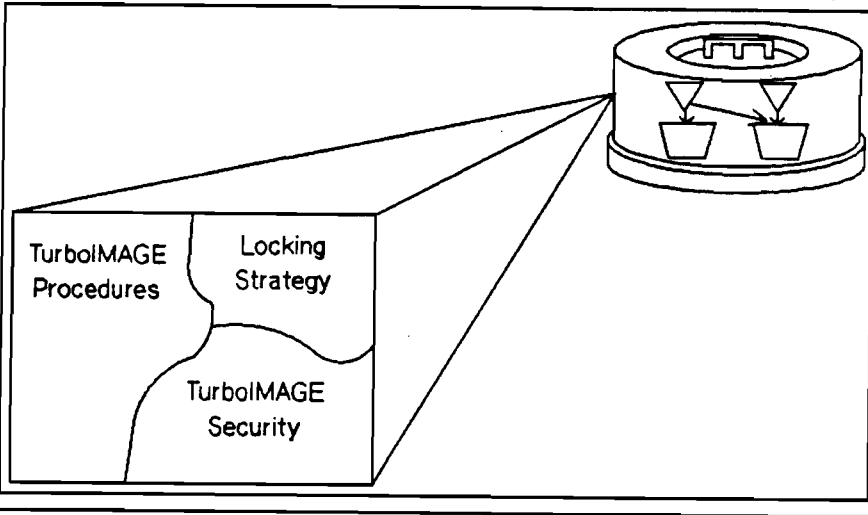
Upon completion of this module, you will be able to:

- state the three major components of a data base application.
- write a data base application program.

Review of Data Base Applications

Notes

Components of a Data Base Application



Module 7-2 Instructor Notes: Slide 7.01

Review of Data Base Applications

Purpose: To review the components of developing a data base application program.

Key Points:

- After a data base has been created by DBUTIL, application programs can be written to enter and use the data. Programs written in COBOL, FORTRAN, SPL, Pascal, or BASIC gain access to the data base through the TurboIMAGE procedures. RPG programs contain specifications used by the Report Program Generator to make calls to the TurboIMAGE procedures.
- A security plan is implemented by the data base designer. This plan includes the assignment of passwords and user class numbers. This plan is an important and necessary component in the overall implementation of a data base application.
- To ensure data integrity, a locking strategy must be designed and observed by all users of the data base.

Module 7-3

Activity 7-1a Lab: Writing a Data Base Application

Purpose: To write a data base application program.

Notes: This lab is designed to offer you the opportunity to write a complete data base application program. The program does not use VPLUS/3000 and is therefore a simpler application. The lab is offered at three levels depending upon the amount of programming required.

Specifications:

This program opens the REALTY data base and allows the user to add, delete, or update entries in the CITY-MASTER data set. The user is prompted for the desired function and then for the necessary values.

Level 1 of this activity requires only the substitution of TurboIMAGE parameters and error messages.

Level 2 requires coding of all TurboIMAGE calls and communications with the user.

Level 3 requires coding of all variable declarations, all TurboIMAGE calls, all communications with the user, and all logic to provide the correct flow of the program.

Directions:

1. Study the flowchart on the next page.
2. Read the description of the three levels listed above.
3. Text in the appropriate file and save it in your group under the same name.
4. Read the description of the program at the beginning of the file.
5. Enter all code necessary to make the program execute correctly. If you encounter difficulty, text in the next lower level and work from there.
6. Compile the source file.
7. When you have a clean compile, prep your program as follows: :PREP \$OLDPASS, LEV#xxxP
8. Run LEV#xxxP and verify that all functions of the program execute correctly.
9. When you complete this activity, try the optional exercise on the next page.

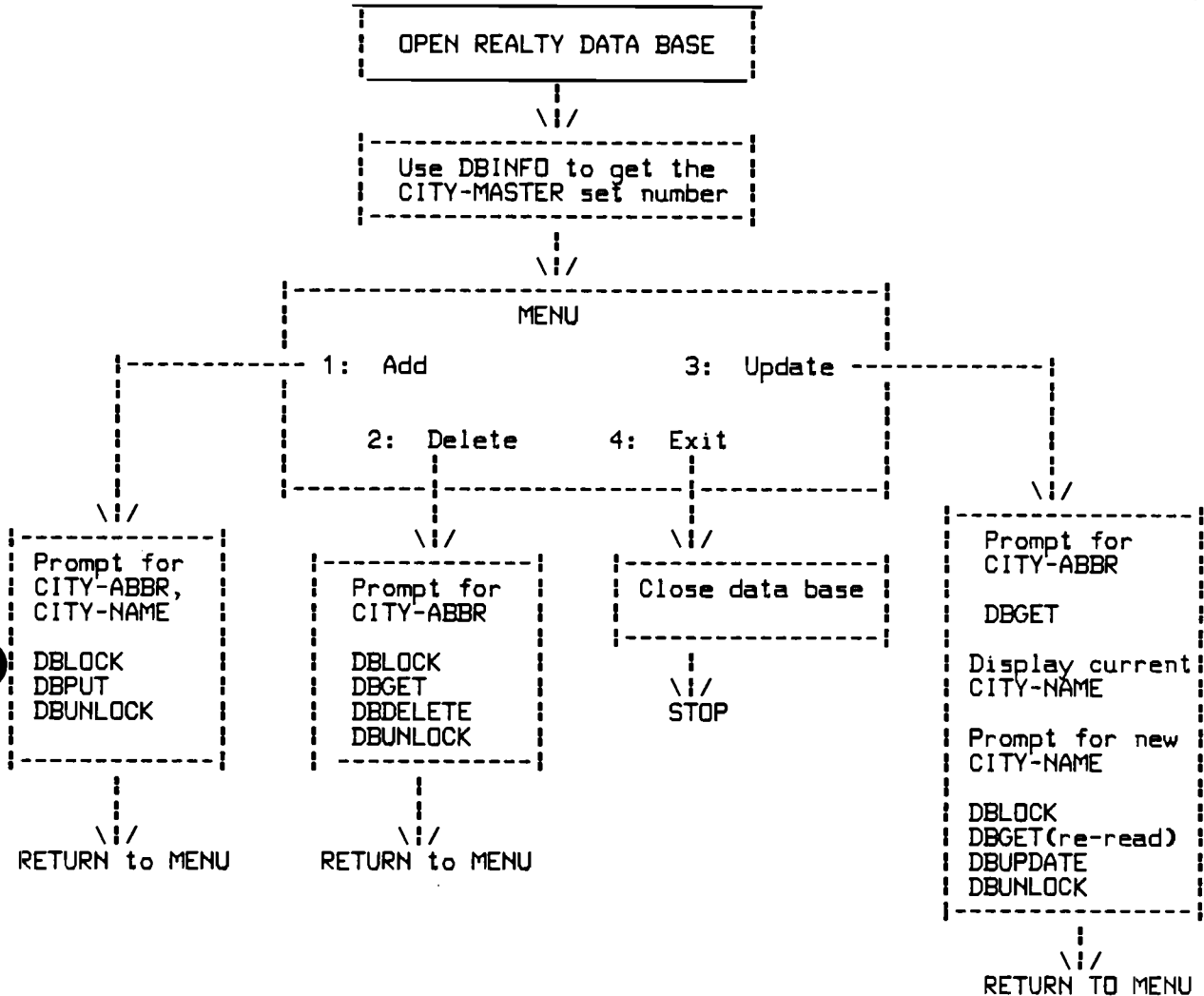
Solutions: The solution programs are: S7xxx 1S.SOLUTION

(Continued on next page.)

Module 7-4

Activity 7-1a Lab: Writing a Data Base Application

APPLICATION FLOWCHART



Lab Files

LEVEL 1

LEVEL 2

LEVEL 3

-----> (Most Challenging)

LEV1xxxS.LABS

LEV2xxxS.LABS

LEV3xxxS.LABS

[replace xxx with : COB - COBOL
FTN - FORTRAN
PAS - Pascal
BAS - BASIC
SPL - SPL
RPG - RPG]

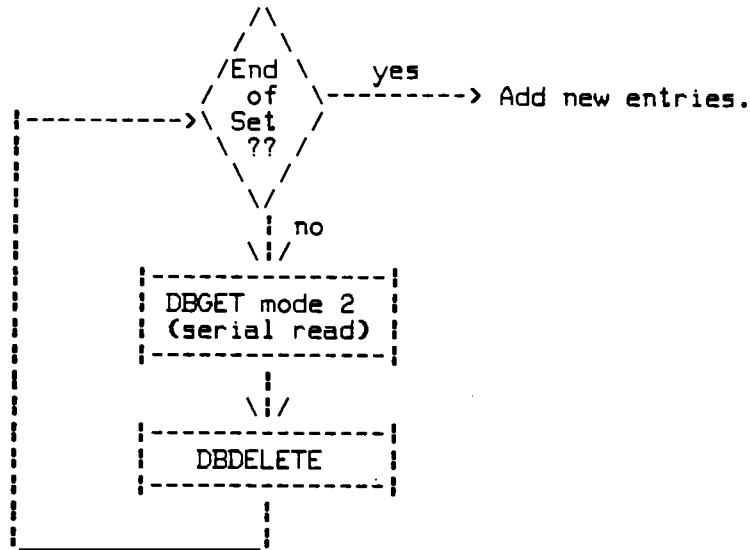
Module 7-5

□ Activity 7-1b Lab: Optional Exercise

Purpose: To solve an application design problem involving master data sets.

Specifications:

A program is written to delete all entries in a master data set and place new entries in the set. When the program is run, some of the original data is still in the set. The following flowchart represents the portion of the program designed to delete the entries in the master set.



Directions:

1. Read the situation described above.
2. Describe how it is possible to have entries left in the set after the series of DBDELETEs are completed.
3. Explain how you would modify the program so this situation does not occur again.

Solution: Three possible solutions are described in Appendix A.

break

Module 7-5 Instructor Notes

Activity 7-1 Lab: Writing a Data Base Application

Purpose: To write a data base application program. **Time:** 1 hour 30 minutes

Directions:

1. Students are to complete this activity in groups of two. Assign students to partners according to the language to be used and the level of programming experience.
2. Encourage students to begin at level 3. If level 3 is too difficult they can move down to level 2 and then to level 1 if necessary. Each group should have a completed program at the end of the activity.
3. Encourage students to attempt the optional exercise if they have time. This is a more challenging exercise. Do not expect all students to complete the exercise correctly. However, discuss the exercise and its solutions with the entire class.

Solutions:



SOLUTIONS TO OPTIONAL EXERCISE

The situation described in this exercise occurs because of migrating secondaries (Module 5). If an entry is deleted and it is a primary on a synonym chain then the first secondary migrates into the primary position. The program then moves on to the next record and misses the migrated entry. Three possible solutions are described below.

Solution # 1

After the DBDELETE succeeds, perform a DBGET mode 1 (re-read) to determine if an entry still exists in the same location. If no entry exists, continue with the serial read.

Solution # 2

After the DBGET mode 2, perform a sequence of DBDELETES until the status returns a condition code of 17 (no entry exists). The sequence of DBDELETES will continually delete entries in the same location. Therefore, all entries that migrate into the location will be deleted.

Solution # 3

After the call to DBGET, the status array (words 5 and 6) contains a count of the number of entries in the synonym chain (or 0 if the entry is a primary entry). This value can then be used to perform the exact number of DBDELETES necessary to delete all the entries in the synonym chain.

Module 7-6 Instructor Notes

Preparation Material: Slide 7.01

Teaching Tips:

Summarize the three components of application implementation that were covered in Modules 5 and 6. Review the uses of the TurboIMAGE procedures including DBLOCK and DBUNLOCK. Review the use of user classes to provide data base security.

This is an appropriate time to answer all questions students have regarding data base application programming. The activity that follows requires the student to create a simple application program. The activity is offered at three levels depending upon programming requirements.

Module 8 Instructor Notes

Utilities and Transaction Logging

Overview of Module 8

* - indicates preparation and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. Utilities (1 hour 20 minutes)

Slide: 8.01 - TurboIMAGE Utilities *

Activity: Chalk Talk: DBUTIL Utility Program

Purpose: To introduce the DBUTIL commands.

Text Page: Options for >>SET, >>ENABLE, and >>DISABLE

Slides: 8.02 - :RUN DBSTORE.PUB.SYS
8.03 - :RUN DBRESTOR.PUB.SYS
8.04 - :RUN DBUNLOAD.PUB.SYS *
8.05 - :RUN DBLOAD.PUB.SYS *
8.06 - :RUN DBCONV.PUB.SYS

Activity: Demo: Utility Simulations

Purpose: To see how the TurboIMAGE utilities work by using an on-line interactive simulation.

Lesson 2. Data Base Restructuring (25 minutes)

Activity: Chalk Talk: Sequence of Operations

Purpose: To identify all necessary steps that need to be performed to modify the structure of a data base.

Text Page: Supported and Unsupported Design Changes

Activity: Worksession: Data Base Restructuring

Purpose: To indicate the utilities and subsystems needed to accomplish a number of restructuring activities.

Lesson 3. Transaction Logging (1 hour 5 minutes)

Slides: 8.07 - Why Log?
8.08 - TurboIMAGE Logging
8.09 - Logical Transactions
8.10 - The Logging Cycle *
8.11 - Logging Considerations

Activity: Demo: Implementing Logging

Purpose: To implement TurboIMAGE logging in an on-line interactive session.

Activity: Chalk Talk: Logging Media

Purpose: To identify advantages and disadvantages of logging to disc and logging to tape.

Module 8 Instructor Notes

□ Utilities and Transaction Logging

Lesson 4. Data Base Recovery (1 hour 30 minutes)

Slides: 8.12 - DBRECOV
8.13 - :RUN DBRECOV.PUB.SYS

Activity: Chalk Talk: DBRECOV >CONTROL Command

Purpose: To present the options available with the DBRECOV >CONTROL command.

Slides: 8.14 - Roll-Forward Recovery
8.15 - Intrinsic Level Recovery (ILR)
8.16 - Roll-Back Recovery

Activity: Demo: Implementing Data Base Recovery

Purpose: To implement the Roll-Forward, Roll-Back, and ILR recovery methods.

Activity: Worksession: Logging and Recovery

Purpose: To review transaction logging and recovery options.

Slide: 8.17 - TurboIMAGE: The Whole Picture *

Module 8-1

□ Utilities and Transaction Logging

Goal: To introduce the utilities and recovery methods available with TurboIMAGE. To define the transaction logging process and its associated benefits.

Objectives:

Upon completion of this module, you will be able to:

- state the function of the TurboIMAGE utilities.
- construct a logical sequence of data base restructuring operations.
- understand the importance of logging and recovery procedures.
- identify the components of logging implementation.
- utilize recovery procedures.

TurboIMAGE Utilities



- DBUTIL - performs creation, deletion, and maintenance functions.
- DBSTORE - stores data base to tape or disc.
- DBRESTOR - copies data base to disc.
- DBUNLOAD - copies data to tape or disc.
- DBLOAD - loads data into data base.
- DBRECOV - recovers data base transactions from log file.
- **DBCONV** - converts IMAGE/3000 data base to TurboIMAGE data base.

Refer to *TurboIMAGE Data Base Management System Reference Manual*, Section 8: Using The Data Base Utilities, for details of each utility.

Module 8-2 Instructor Notes: Slide 8.01

Utilities and Transaction Logging

Purpose: To give an overview of the TurboIMAGE utilities that will be covered in this module.

Key Points:

- TurboIMAGE utility programs are used to create and initialize data base files, and to perform various maintenance functions.
- The utility programs can be run in either job or session mode. DBUTIL, DBSTORE, DBRESTOR, DBUNLOAD, and DBLOAD all require the user to be logged on under the group and account that contains the data base root file.
- The user must be the data base creator to execute DBUTIL >>CREATE to change or remove the maintenance word. To operate the other utility programs or enter other DBUTIL commands, the user must be able to supply the data base maintenance word.

Module 8-3

Activity 8-1 Chalk Talk: DBUTIL Utility Program

Purpose: To introduce the DBUTIL commands.

Directions: As your instructor describes the following DBUTIL commands, write a summary of its operation in the space provided.

>>HELP

>>CREATE

>>ERASE

>>MOVE

>>PURGE

>>DEACTIVATE

>>ACTIVATE

>>VERIFY

>>RELEASE

>>SECURE

>>SET

>>ENABLE

>>DISABLE

>>SHOW

>>EXIT

Module 8-3 Instructor Notes

□ Activity 8-1 Chalk Talk: DBUTIL Utility Program

Time: 15 minutes

Purpose: To introduce the DBUTIL utility program and its commands.

Directions: Summarize each of the commands listed below. Instruct students to take notes in the workbook.

- >>HELP will display all of the DBUTIL commands.
- >>CREATE creates and initializes a file to binary zeros for each data set in the data base. The data set file names are created by appending two digits to the root file name. A TurboIMAGE data base can have up to 199 data sets (99 in IMAGE/3000). The >>CREATE command obtains its information from the root file.
- >>ERASE re-initializes the data sets to binary zeros.
- The >>MOVE command is only available with TurboIMAGE. >>MOVE moves TurboIMAGE files to a specific device or class of devices. Using >>MOVE correctly will minimize disc head contention and improve overall data base performance.
- >>PURGE removes the root file and all referenced data sets from the MPE directory and returns the disc space to the system.
- >>DEACTIVATE, >>ACTIVATE, and >>VERIFY are used for accessing remote data bases over DSN/DS via the data-base-access file.
- >>RELEASE temporarily suspends file system security on the root file and data sets but leaves TurboIMAGE security intact. File security remains suspended until the creator issues a >>SECURE command.
- >>SET can be used to change or remove the data base maintenance word, and to specify the number of input/output buffers. The number of I/O buffers is dependent upon the current number of data base users.
- >>ENABLE enables the data base for access, dumping, logging, ILR, and recovery options. >>DISABLE disables access, dumping, logging, ILR, and/or recovery enabled by the >>ENABLE command.
- >>SHOW displays specific data base information.
- >>EXIT terminates DBUTIL execution.

General DBUTIL Comments:

- DBUTIL commands can be abbreviated to the first three characters. For example, >>DEACTIVATE can be abbreviated to >>DEA.
- >>CREATE, >>ERASE, and >>PURGE commands can be specified in the :RUN expression as entry points, thereby bypassing the command prompt. For example, :RUN DBUTIL.PUB.SYS.CREATE.
- The data base must be opened for exclusive access to perform any DBUTIL command except >>SHOW, >>HELP, or >>EXIT.

Module 8-4

□ Utilities and Transaction Logging

Options for >>SET, >>ENABLE, and >>DISABLE.

I. >>SET data base name [/maint word]

MAINT = maintenance word

* BUFFSPECS = number of buffers (from users/to users)
[,number buffers (from users/to users)] ...

LOGID = log identifier

PASSWORD class number = [password]

* Refer to Appendix E for information on buffer management.

II. >>ENABLE \ data base name [/maint word] FOR option [,option ...]
>>DISABLE /

ACCESS = user access (read/write/query) to the data base

AUTODEFER = automatic deferred output for the data base

DUMPING = dumps user stack and control blocks if TurboIMAGE aborts

ILR = Intrinsic Level Recovery

LOGGING = data base logging facility

RECOVERY = data base roll-forward recovery facility

ROLLBACK = data base roll-back recovery facility

Module 8-4 Instructor Notes

Utilities and Transaction Logging

Purpose: To review the >>SET, >>ENABLE, and >>DISABLE command options.

Note: ILR, LOGGING, RECOVERY, and ROLLBACK will be discussed in detail in the Transaction Logging and Data Base Recovery sections of this module.

I. >>SET data base name [/maint word]

- Only the data base creator can change or remove the maintenance word.
- BUFFSPECS from-users is the minimum number of concurrent users for which the preceding number of buffers should be allocated; the minimum number ranges between 1 and 120. To-users is the maximum number of concurrent users for which the preceding number of buffers should be allocated; the maximum number ranges between 1 and 120.
- The from-users/to-users ranges must be specified in increasing order. The ranges must not overlap and need not be consecutive. If a number of buffers is not specified for a particular number of users, the default number of buffers is used. (See Appendix E for a description on how the number of buffers is identified.) The minimum number of buffers allowed is 4 and the maximum is 255. To obtain the best performance, specify between 15 and 25 buffers.
- LOGID is an MPE log identifier obtained using the :GETLOG command. DBUTIL checks to insure that the log identifier is valid, then prompts for a password. Entry of the correct password will cause the log identifier to be stored in the root file and used whenever the logging capability is enabled.
- Classnumber is the user class number whose password is being changed; a number between 1 and 63.
- Password is the new password being assigned to a particular user class number. If password is omitted, any password previously assigned to that class is removed. Changing passwords dynamically aids in tightening security.

II. >>ENABLE \ data base name [/maint word] FOR option [,option ...]
>>DISABLE /

- Multi-user AUTODEFER defers the posting of TurboIMAGE buffers until they become full, allowing the program to run faster. The data base should be backed-up before running the program with AUTODEFER, as the buffers are not written to disc on a per transaction basis, potentially leaving the data base in a physically inconsistent state.
- DUMPING is used for development and debugging only. When enabled, the TurboIMAGE abort procedure copies the user's stack and the control blocks to a file if a TurboIMAGE procedure aborts.

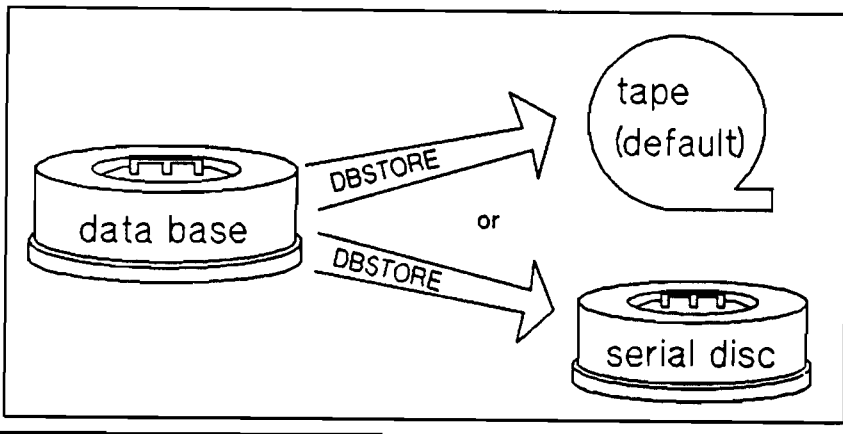
Appendix E

Utilities and Transaction Logging

Notes

:RUN DBSTORE.PUB.SYS

- Stores data base root file and data sets to tape or serial disc.



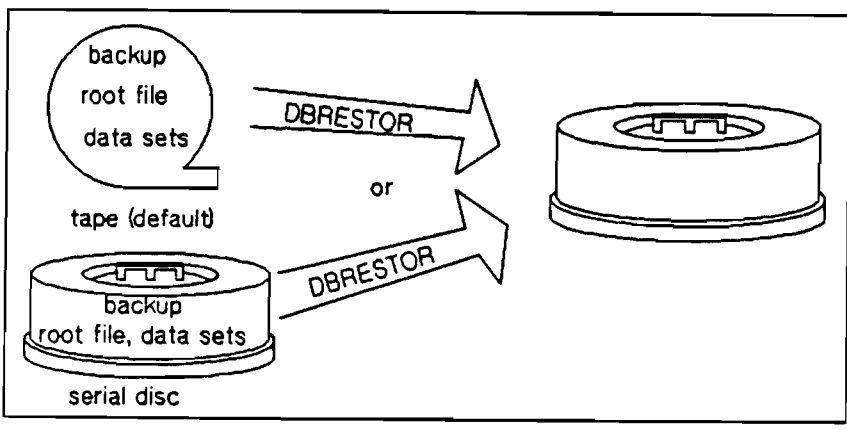
T1 8.02

© 1985 Hewlett-Packard Company

- DBSTORE sets the timestamp and store flag in the root file.
- DBSTORE format conforms to MPE :STORE.
- DBSTORE backup can be read by MPE :RESTORE.

:RUN DBRESTOR.PUB.SYS

- Restores data base root file and data sets from backup tape or serial disc volumes to disc.



T1 8.03

© 1985 Hewlett-Packard Company

- DBRESTOR restores the backup created by DBSTORE, MPE :STORE, or MPE :SYSDUMP.

Utilities and Transaction Logging

Purpose: To present the DBSTORE utility.

Key Points:

- The format of DBSTORE files is compatible with backup files created by MPE :STORE and MPE :SYSDUMP commands. DBSTORE differs from these commands in that it handles only TurboIMAGE data bases.
- DBSTORE must gain semi-exclusive access to the data base (users can open the data base in mode 6 or 8). If semi-exclusive access is not obtainable, DBSTORE terminates.
- If ILR is enabled at the time of the store, DBSTORE stores the ILR file associated with the data base on the same tape or serial disc. The log file will be restored automatically by DBRESTOR.
- DBSTORE updates a timestamp and store flag in the data base root file before storing the data base. The timestamp designates the date and time of the DBSTORE operation, and is used by DBRECOV to help identify the correspondence between logfiles and backup data bases. The store flag indicates that the data base has been stored, and is used by both DBRECOV and DBUTIL to help ensure the existence of a valid backup data base.
- Although MPE :STORE or MPE :SYSDUMP may be used to store data base files, neither updates the timestamp and store flag. It is recommended to use DBSTORE for the added protection it provides.

Purpose: To present the DBRESTOR utility.

Key Points:

- DBRESTOR restores the root file and data sets (along with the data) to disc. If ILR was enabled when DBSTORE was run, the ILR file was written on the same tape or serial disc as the root file and data sets. When the data base is restored with DBRESTOR, the ILR file is automatically restored along with the data base.

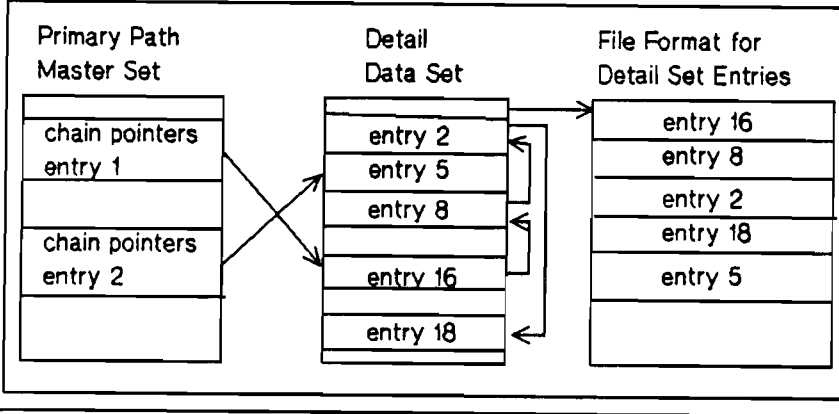
Utilities and Transaction Logging

Notes

**:RUN DBUNLOAD.PUB.SYS [,CHAINED]
[,SERIAL]**

- Writes data entries from each data set to tape or serial disc in a special format.

- In chained mode (default), detail data entries are copied by primary key.



Module 8-6 Instructor Notes: Slide 8.04

□ Utilities and Transaction Logging

Purpose: To show how the DBUNLOAD utility operates in chained mode.

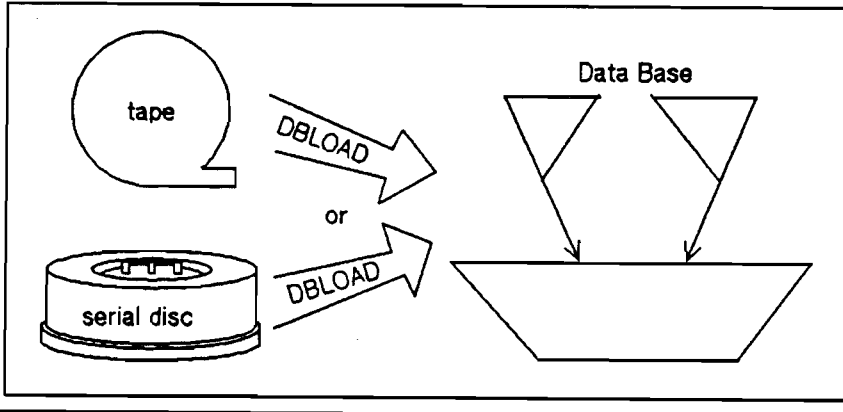
Key Points:

- DBUNLOAD copies data, not root file, chain information, or structure to tape or serial disc. DBUNLOAD requires exclusive access to the data base, and the default device class is TAPE.
- The data base must be unloaded with DBUNLOAD if modifications to the data base structure are to occur. For example, to increase data base capacity, unload the entries, purge the data base, change the schema and create a new root file, execute DBUTIL >>CREATE, and then reload the data entries from the volumes created by DBUNLOAD.
- In chained mode, DBUNLOAD copies all of the detail entries with the same primary path search item value to contiguous locations on the backup file. The ordering of the search item values from the primary path is based on the physical order of the matching value in the associated master data set. After the data base is reloaded, chained access along the primary path is more efficient.
- If a chained DBUNLOAD encounters a broken chain, it will unload all entries in the chain down to, but not including the break. It will then go to the end of the chain and unload all entries up to the break. DBUNLOAD will report any missing entries. If entries are missing, the user should unload the data base serially.
- In serial mode, DBUNLOAD copies the data entries serially in record number order. "Stand-alone" detail data sets are always unloaded serially.
- Using the file command, :FILE DBUNLOAD=\$NULL, will cause a simulated unloading of the data base. This provides information about data set chains without actually performing a DBUNLOAD.
- CONTROL-Y can be used when executing DBUNLOAD in session mode, to request the approximate number of entries in the current data set that have already been written.

⊛ DBUNLOAD to \$NULL still requires exclusive access to Data base

:RUN DBLOAD.PUB.SYS

- Loads data entries from DBUNLOAD volumes into data sets of the data base.



□ Utilities and Transaction Logging

Purpose: To present the DBLOAD intrinsic.

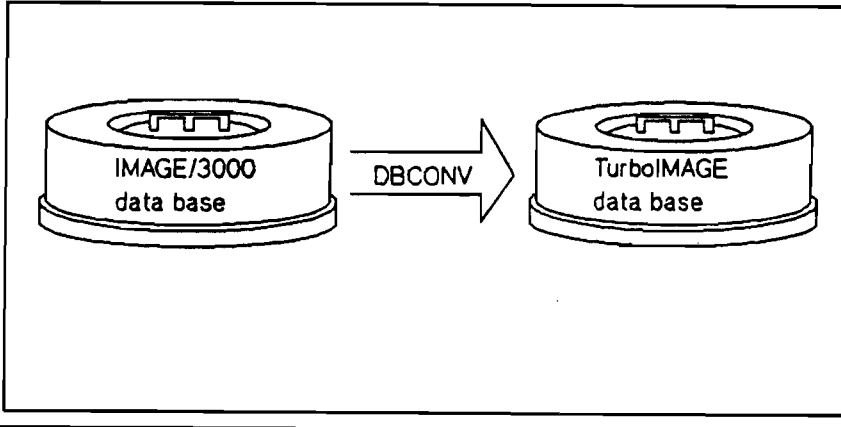
Key Points:

- DBLOAD is used for data base restructuring or for compacting chains along the primary path. DBLOAD requires exclusive access to the data base, and the default device is TAPE.
- DBLOAD loads onto disc those backup volumes produced by DBUNLOAD. The data base name on the volume must be exactly the same as the data base name, or root file name, in the current session or job's group and account. To reload the identical data into the data base, the DBUTIL >>ERASE command must be used prior to DBLOAD, unless the data base has been purged and re-created.
- DBLOAD reads each entry from the backup volume and puts it into the respective data set from which it was read by DBUNLOAD. If a data set in the receiving data base is an automatic master, no entries are directly put into it by DBLOAD, even though there are entries on the volume associated with the data set's number. Automatic master entries are created as needed, when entries are put into the related detail data sets.
- DBLOAD calls the DBPUT procedure to put the entries read from the backup volume into the appropriate data sets. If the data set entry is larger than the backup entry, the data is left-justified and is padded out to the maximum entry length with binary zeros. If the data entry is smaller than the backup entry, the backup volume record is truncated on the right and the truncated data is lost.
- The location of master set entries is based on their search item value which is hashed to an internal location. The detail data set entries are put into consecutive data set records with the appropriate new chain pointer information.
- CONTROL-Y can be used in session mode to request the approximate number of entries in the current data set that have already been copied.
- The data base is not logically or structurally complete on disc until the DBLOAD is complete. If an MPE or hardware crash occurs during a DBLOAD, the load will need to be performed again once the system comes back up. — because it uses an "output deferred" option which does not always write data & structural info back to disc when DBPUT returns to DBLOAD. The data base is flagged "bad" and not unflagged until DBLOAD is finished completely.

:RUN DBCONV.PUB.SYS



- Converts IMAGE/3000 data base to TurboIMAGE data base.



- Conversion from IMAGE/3000 to TurboIMAGE is necessary due to the new limitations and addition of new tables in the root file.

Module 8-8 Instructor Notes: Slides 8.06

Utilities and Transaction Logging

Purpose: To present the **DBCONV** utility.

Key Points:

- TurboIMAGE and IMAGE/3000 are not compatible. **DBCONV** must be used to convert IMAGE/3000 data bases to TurboIMAGE data bases, once TurboIMAGE is on your system.
- Detail sets do not need to be converted.
- It is highly recommended that a **DBSTORE** of the data base and **MPE :STORE** of the schema are performed prior to updating the system to TurboIMAGE.
- **DBCONV** will not run if **ILR** is enabled on the data base. If enabled, a **DBUTIL >>DISABLE** should be performed to disable **ILR** before TurboIMAGE is installed and the conversion program is run.
- If it is necessary to return to IMAGE/3000 after data bases have been converted to TurboIMAGE, a **DBCONV; BACKWARD** program is available. But, this backward conversion process is not possible if the new TurboIMAGE data base exceeds the IMAGE/3000 limits.
- **DBCONV** will terminate and print the message **DATA BASE IN USE** if it cannot gain exclusive access to the specified data base.
- During the conversion of a data base from IMAGE/3000 to TurboIMAGE format, there is a potential for an increase in the amount of disc space required. This occurs because **DBCONV** adds one word to each path count in a master media record, thereby expanding the limit on chain entries from 65K to 2 billion.
- During the conversion process, a certain amount of extra disc space is required by the **DBCONV** utility. This disc space requirement is **TEMPORARY** and is in addition to any potential increase in permanent disc space required by the TurboIMAGE data base.

not to go as large as the biggest master set

Lunch

Module 8-9

Activity 8-2 Demo: Utility Simulations

Purpose: To see how the TurboIMAGE utilities work by using an on-line interactive simulation.

Directions:

1. Type :UTILITY to run the interactive session.
2. Refer to Appendix B for syntax of the DBUTIL commands, or use the HELP command during the session.
3. Upon completion of the activity, you can request a copy be sent to the line printer.

Module 8-9 Instructor Notes

Activity 8-2 Demo: Utility Simulations

Time: 30 minutes

Purpose: To see how the TurboIMAGE utilities work by using an on-line interactive simulation.

Notes: This activity is an on-line simulation of the following TurboIMAGE utilities: DBUTIL, DBLOAD, DBUNLOAD, DBSTORE, DBRESTOR, and DBCONV. The DBUTIL simulation covers the following commands: HELP, SHOW, SET, ENABLE, MOVE, and EXIT.

Although the simulation requires students to use the REALTY data base as the data base of choice in answering the questions, please emphasize to the students that they are only simulating usage of the TurboIMAGE utilities; they are not actually affecting their own REALTY data base.

Informative responses will be provide for both correct and incorrect answers.

Each students can request a listing at the end of the activity. Encourage students to run the session individually.

The students can access the utility simulations in any order via the main menu provided in the session. Instruct students to run each of the utility simulations available in the activity.

Directions:

1. Make sure the UDC file ACCUDC.LABS has been set and released at the account level. The students will initiate this session by typing the UDC :UTILITY.
2. The questions and answers are in the file MOD8LIST.SOLUTION.IMAGE. This file will be sent to device class LP upon request of the student at the end of the session. A copy of this file is listed below.

Answers:

MODULE 8 - TurboIMAGE Utilities Simulation

LESSON 1: DBUTIL

1. Enter the command to initiate execution of the DBUTIL program.

ANSWER: :RUN DBUTIL.PUB.SYS

HP32215C.04.30 TurboIMAGE/3000: DBUTIL (C) COPYRIGHT HEWLETT-PACKARD CO. 1985

(Continued on next page.)

Module 8-9 Instructor Notes

□ Activity 8-2 Demo: Utility Simulations (cont'd)

2. Enter the command to display all the available DBUTIL commands.

ANSWER: >>HELP

Commands are:

HELP	CREATE	ERASE	PURGE	DEACTIVATE
ACTIVATE	VERIFY	SET	ENABLE	DISABLE
SHOW	EXIT	RELEASE	SECURE	*MOVE

Commands may be abbreviated to the first three letters. For help on a particular command type: 'HELP command name'.

* MOVE is not defined in IMAGE/3000.

3. Enter the command to display information about the REALTY data base.

ANSWER: >>SHOW REALTY ALL (>>SHO REALTY ALL)

For data base REALTY

maintenance word is not present.

Access is enabled.

*Autodefer is disabled.

Dumping is disabled.

* TurboIMAGE Only !

*Rollback recovery is disabled.

Logging is disabled.

Recovery is disabled.

ILR is disabled.

Data base last stored on FRI, MAR 29, 1985, 1:59 PM

Data base has not been modified since last store date.

Subsystem access is READ/WRITE.

Logid is not present.

Buffer specifications:

8(1/2),9(3/4),10(5/6),11(7/8),12(9/10),13(11/12),14(13/14),
15(15/16),16(17/18),17(19/120)

No other users are accessing the data base.

4. The SHOW REALTY ALL display indicated that no maintenance word exists for the REALTY data base.

Assign the maintenance word HP to the REALTY data base.

ANSWER: >>SET REALTY MAINT=HP

Maintenance word changed.

```
* * * * *
* The maintenance word for a data base must be supplied by *
* anyone other than the data base creator. Only the data *
* base creator can change or remove the maintenance word. *
* * * * *
```

(Continued on next page.)

Module 8-9 Instructor Notes

□ Activity 8-2 Demo: Utility Simulations (cont'd)

5. Verify that the maintenance word 'HP' has been assigned to the REALTY data base.

ANSWER: >>SHOW REALTY MAINT (>>SHO REALTY MAINT)

For data base REALTY

MAINTENANCE WORD: HP

6. The number of buffers available to users is determined by the number of users on the system. Type SHOW REALTY BUFFSPECS at the prompt to display the TurboIMAGE default buffer specifications.

ANSWER: SHOW REALTY BUFFSPECS

Buffer specifications:

8(1/2),9(3/4),10(5/6),11(7/8),12(9/10),13(11/12),14(13/14),
15(15/16),16(17/18),17(19/120)

The format is: # of buffers (range of users)

Example: 11(7/8) - 11 buffers will be allocated if there are 7 or 8 users accessing the data base.

7. If a varied number of users will be accessing a data base, the default buffer specifications result in expansions and contractions of the buffer area in memory. This means extra overhead. Therefore, with a wide range of users, the best performance can be achieved by allocating a constant number of buffers.

Use the SET command to allocate 25 buffers for a range of users from 1 to 120. (Type HELP SET to display the correct syntax.)

ANSWER: SET REALTY BUFFSPECS=25(1/120)

Buffer specifications:

25(1/120)

8. TurboIMAGE maintains flags indicating the state (enabled or disabled) of the logging, rollback, ILR, recovery, restart, subsystem access, autodefer, access, and dumping options.

Issue the DBUTIL command to display the state of all the flags.

ANSWER: >>SHOW REALTY FLAGS (>>SHO REALTY FLAGS)

For data base REALTY

Access is enabled.

*Autodefer is disabled.

*Rollback recovery is disabled.

Restart is disabled.

Dumping is disabled.

Logging is disabled.

Recovery is disabled.

ILR is disabled.

Subsystem access is READ/WRITE.

* TurboIMAGE only !

(Continued on next page.)

Module 8-9 Instructor Notes

□ Activity 8-2 Demo: Utility Simulations (cont'd)

9. Enable the REALTY data base for recovery.

Then verify that recovery is enabled.

ANSWER: >>ENABLE REALTY FOR RECOVERY (>>ENA REALTY FOR RECOVERY)

Recovery is enabled.

>>SHOW REALTY FLAGS (>>SHO REALTY FLAGS)

For data base REALTY

Access is enabled.
Autodefer is disabled.
Rollback recovery is disabled.
Restart is disabled.
Dumping is disabled.
Logging is disabled.
Recovery is enabled.
ILR is disabled.
Subsystem access is READ/WRITE.

10. To increase performance, the most frequently accessed data sets should reside on different disc devices.

Issue the DBUTIL command to show the device class names currently being used by the REALTY data base.

ANSWER: >>SHOW REALTY DEVICE (>>SHO REALTY DEVICE)

For data base REALTY

Data Set MPE file Name	Data Set Name	Device
REALTY01.GROUPN.IMAGE	Listnr-master	DISC1
REALTY02.GROUPN.IMAGE	List-price-mstr	DISC1
REALTY03.GROUPN.IMAGE	Bath-master	DISC1
REALTY04.GROUPN.IMAGE	Bed-master	DISC1
REALTY05.GROUPN.IMAGE	Zoning-master	DISC1
REALTY06.GROUPN.IMAGE	City-master	DISC1
REALTY07.GROUPN.IMAGE	Residential	DISC1
REALTY08.GROUPN.IMAGE	Commercial	DISC1

11. Use the MOVE command to move the RESIDENTIAL data set to device class DISC2.

Then verify that the data set resides on DISC2.

ANSWER: >>MOVE REALTY07 TO DISC2

Data base last stored on MON, MAR 4, 1985, 8:32 PM
Data base has been modified since last store date.
The data base should be backed up before doing a MOVE operation.
Do you still want to continue the MOVE operation (Y/N)? Y

(Continued on next page.)

Module 8-9 Instructor Notes

Activity 8-2 Demo: Utility Simulations (cont'd)

Starting file copy ...

... file copy completed.

Purging old copy of file "REALTY07" New copy of file "REALTY07" saved as a permanent file. File "REALTY07" moved to device DISC2.

```
*****
* It is recommended to do a DBSTORE of the data base prior *
* to performing a MOVE. This is a precaution against the *
* system failing during the MOVE operation. *
* *
* NOTE: THE MOVE command is defined in TurboIMAGE only! *
*****
```

Verify that REALTY07 resides on DISC2.

ANSWER: >>SHOW REALTY DEVICE (>>SHO REALTY DEVICE)

For data base REALTY

Data set MPE file Name	Data Set Name	Device
REALTY01.GROUPN.IMAGE	Listnr-master	DISC1
REALTY02.GROUPN.IMAGE	List-price-mstr	DISC1
REALTY03.GROUPN.IMAGE	Bath-master	DISC1
REALTY04.GROUPN.IMAGE	Bed-master	DISC1
REALTY05.GROUPN.IMAGE	Zoning-master	DISC1
REALTY06.GROUPN.IMAGE	City-master	DISC1
REALTY07.GROUPN.IMAGE	Residential	DISC2
REALTY08.GROUPN.IMAGE	Commercial	DISC1

12. Enter the command to exit DBUTIL.

ANSWER: >>EXIT
END OF PROGRAM

This concludes LESSON 1: DBUTIL. Please refer to the TurboIMAGE Reference Manual, Section 8: DBUTIL, or Appendix B, for complete details on the DBUTIL utility.

LESSON 2: DBSTORE

The DBSTORE program stores the data base root file and all data set files to a tape or serial disc in a format compatible with back-up files created by the MPE :STORE and :SYSDUMP commands.

Assume you want to create a back-up copy of your REALTY data base. How would you initiate execution of the DBSTORE program?

ANSWER: :RUN DBSTORE.PUB.SYS

WHICH DATA BASE? REALTY
DATA BASE STORED
END OF PROGRAM

LESSON 3: DBRESTOR

The DBRESTOR program copies a data base from the back-up volume(s) created by the DBSTORE program, or the MPE :STORE or :SYSDUMP commands, to disc.

(Continued on next page.)

Module 8-9 Instructor Notes

Activity 8-2 Demo: Utility Simulations (cont'd)

Assume you want to restore a back-up copy of your REALTY data base to disc. How would you initiate execution of the DBRESTOR program?

ANSWER: :RUN DBRESTOR.PUB.SYS

WHICH DATA BASE? REALTY

DATA BASE RESTORED END OF PROGRAM

LESSON 4: DBUNLOAD

The DBUNLOAD program copies data entries from each data set to tape or serial disc volumes specially formatted by MPE.

Assume you want to copy data from your REALTY data base to tape. How would you initiate execution of the DBUNLOAD program in chained mode?

ANSWER: :RUN DBUNLOAD.PUB.SYS (:RUN DBUNLOAD.PUB.SYS,CHAINED) (Note: chained is the default mode)

WHICH DATA BASE? REALTY

DATA SET 1: 50 ENTRIES EXPECTED, 50 ENTRIES PROCESSED.
DATA SET 2: 35 ENTRIES EXPECTED, 35 ENTRIES PROCESSED.
DATA SET 3: 2 ENTRIES EXPECTED, 2 ENTRIES PROCESSED.
DATA SET 4: 4 ENTRIES EXPECTED, 4 ENTRIES PROCESSED.
DATA SET 5: 6 ENTRIES EXPECTED, 6 ENTRIES PROCESSED.
DATA SET 6: 25 ENTRIES EXPECTED, 25 ENTRIES PROCESSED.
DATA SET 7: 40 ENTRIES EXPECTED, 40 ENTRIES PROCESSED.
DATA SET 8: 10 ENTRIES EXPECTED, 10 ENTRIES PROCESSED.
END OF VOLUME 1, 0 WRITE ERRORS RECOVERED
DATA BASE UNLOADED

END OF PROGRAM

LESSON 5: DBLOAD

The DBLOAD program loads data entries from the back-up volume(s) created by the DBUNLOAD program into data sets of the data base.

Assume you want to load data from a back-up copy of your REALTY data base to disc. How would you initiate execution of the DBLOAD program?

ANSWER: :RUN DBLOAD.PUB.SYS

WHICH DATA BASE? REALTY

DATA SET 1: AUTOMATIC MASTER
DATA SET 2: AUTOMATIC MASTER
DATA SET 3: AUTOMATIC MASTER
DATA SET 4: AUTOMATIC MASTER
DATA SET 5: 6 ENTRIES
DATA SET 6: 25 ENTRIES
DATA SET 7: 40 ENTRIES

(Continued on next page.)

Module 8-9 Instructor Notes

Activity 8-2 Demo: Utility Simulations (cont'd)

DATA SET 8: 10 ENTRIES

```
*****
* NOTE: Data sets 1,2,3, and 4 are automatic masters *
*       so 0 entries are copied; the entries are     *
*       created as related detail entries are copied *
*       to the data base.                             *
*****
```

LESSON 6: DBCONV

The DBCONV program converts IMAGE/3000 data sets to TurboIMAGE data sets.

Assume you want to convert your IMAGE/3000 REALTY data base to a TurboIMAGE data base. How would you initiate execution of the DBCONV program?

ANSWER: :RUN DBCONV.PUB.SYS

DATABASE NAME? REALTY

Database has not been modified since last DBSTORE. Continue (YES/NO)? YES

```
REALTY01 has been converted.
REALTY02 has been converted.
REALTY03 has been converted.
REALTY04 has been converted.
REALTY05 has been converted.
REALTY06 has been converted.
REALTY  has been converted.
```

Detail sets don't need to be converted.

Conversion is done !!!!

Module 8-10

Activity 8-3 Chalk Talk: Sequence of Operations

Purpose: To identify all necessary steps that need to be performed to modify the structure of a data base successfully.

Note: It is possible to make certain changes to the structure of an existing data base without having to write special programs to transfer data from the old data base to the new one.

Directions: As your instructor describes the steps necessary in data base restructuring, jot down a description and purpose of each step.

1.

2.

3.

4.

5.

6.

7.

Module 8-10 Instructor Notes

□ Activity 8-3 Chalk Talk: Sequence of Operations

Time: 10 minutes

Purpose: To identify the necessary steps in restructuring a data base.

Note: Draw a diagram on the chalkboard showing the flow from one step to the next.

The following steps are necessary:

1. :RUN EDITOR.PUB.SYS to modify the schema text. (This step does not have to be performed first, as long as it is performed prior to number 4 below.)
2. :RUN DBSTORE.PUB.SYS to store the original data base for insurance in case of error.
3. :RUN DBUNLOAD.PUB.SYS on the old data base, copying all the data entries to tape or serial disc.
4. :RUN DBUTIL.PUB.SYS,PURGE to purge the old data base.
5. :RUN DBSCHEMA.PUB.SYS to redefine the data base using the same data base name and create a new root file.
6. :RUN DBUTIL.PUB.SYS,CREATE to create and initialize the data sets of the new data base.
7. :RUN DBLOAD.PUB.SYS on the new data base using the tape or serial disc created in step 2 to put the old data into the new data base.

It is recommended to :RUN DBSTORE.PUB.SYS to store the data base after the restructuring is completed.

UB-Delta 2 !!!

Talk about DBCHANGE — HP Product — supported, available on ~~UB-Delta 2~~ and later
ADAGER Model 3.
DBCHANGE as alternatives to this long-winded effort

Get the book from Spectrum release on DBCHANGE.

□ Utilities and Transaction Logging**Supported and Unsupported Design Changes****I. Allowable Design Changes**

Any of the following schema changes, alone or combined, which are acceptable to the Schema Processor will always result in a successful transformation of the data base:

- Adding, changing, or deleting passwords and user class numbers.
- Changing a data item or data set name and all references to it.
- Changing data item or data set read and write class lists.
- Adding new data item definitions.
- Removing or changing definitions of unreferenced data items.
- Increasing data set capacities.
- Adding, deleting, or changing sort item designators.

II. Unsupported Design Changes

The following design changes are legitimate only in some circumstances and can result in data set discrepancies or lost data:

- Changing primary paths.
- Adding new data items to the original end of a data entry definition.
- Removing data items from the original end of a data entry definition.
- Changing an automatic master to a manual master or vice versa.
- Changing the native language definition for the data base.
- Adding or deleting a data set at the end of the schema.

Module 8-11 Instructor Notes

Utilities and Transaction Logging

Purpose: To identify the supported and unsupported design (structural) changes on a TurboIMAGE data base.

Key Points:

I. Allowable Design Changes

- Supported design changes are those sanctioned by standard DBLOAD capabilities.

II. Unsupported Design Changes

- Although DBLOAD does not prohibit making the identified unsupported design changes, there is no guarantee that the data will be consistent. DBLOAD and DBUNLOAD always handle full entries, without regard to item positions or lengths. Therefore, if the new data set's entry is defined with the items in a different order than the old data set, DBLOAD will not fail, but the data set contents may still be invalid.
- Performing unsupported design changes may cause loss of data. For example data will be lost if a data set's capacity has been reduced in the new data base to a number less than the number of that data set's entries on the tape or serial disc.

Module 8-12

Activity 8-4 Worksession: Data Base Restructuring

Purpose: To indicate the utilities and subsystems needed to accomplish a number of restructuring activities against a data base.

Directions: Choose, from the list below, each utility and subsystem needed to perform the following tasks. Assume your data base has not been recently backed up. Refer to the *TurboIMAGE Utilities* summary in Appendix B if necessary.

1. Increase the capacity of a data set and change the passwords.

Solution:

2. Your response time has been getting noticeably slower over the past week while adding, deleting and retrieving data entries from a detail data set that has a sorted chain as a primary path.

Solution:

3. Add a new item to the end of a data entry.

Solution:

4. Change the structure of your data base, for example: add a new manual master and change the size of a data item.

Solution:

5. Your data base was accidentally purged. However, your system manager backed up the whole system this morning.

Solution:

List of Options:

- A. :EDITOR (modify original schema)
- B. :RUN DBUTIL.PUB.SYS.CREATE
- C. :RUN DBUTIL.PUB.SYS.ERASE
- D. :RUN DBUTIL.PUB.SYS.PURGE
- E. :RUN DBSTORE.PUB.SYS
- F. :RUN DBRESTOR.PUB.SYS
- G. :RUN DBUNLOAD.PUB.SYS.SERIAL
- H. :RUN DBUNLOAD.PUB.SYS.CHAINED
- I. :RUN DBLOAD.PUB.SYS
- J. :RUN DBSCHEMA.PUB.SYS
- K. :RUN user written unload program
- L. :RUN user written load program

Module 8-12 Instructor Notes

Activity 8-4 Worksession: Data Base Restructuring

Time: 10 minutes

Purpose: To identify the utilities and subsystems needed to restructure a data base successfully.

Directions:

1. Students are to work individually while completing this activity.
2. When all students have completed this activity, discuss each question with the class.

Answers:

CASE 1: A, E, H, D, J, B, I, E

CASE 2: E, H, C, I, E

CASE 3: A, E, H, D, J, B, I, E

CASE 4: A, E, K, D, J, B, L, E

CASE 5: D, F

Book 2

WHY LOG?

Just imagine ...

Time: 4:00 P.M.

Day: Friday, end of the quarter.

... Your clerks have entered 5000 transactions since the last data base backup ...

and then ...

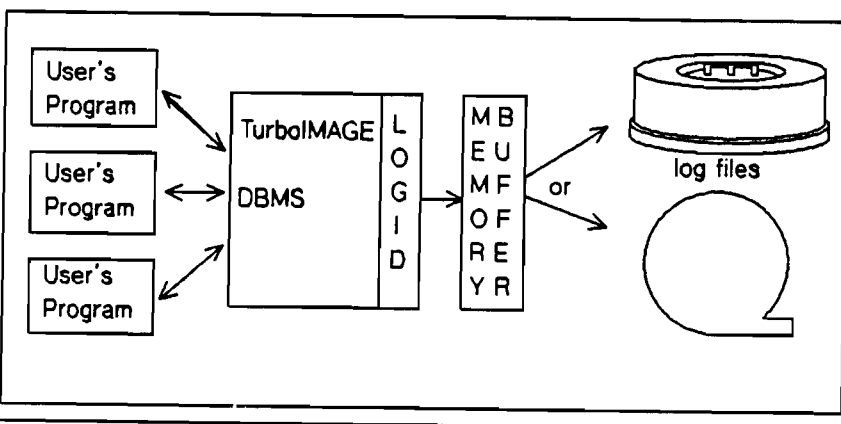
SYSTEM FAILURE

T1 8.07

© 1985 Hewlett-Packard Company

TurboIMAGE Logging

- All transactions that change the data base are logged.



T1 8.08

© 1985 Hewlett-Packard Company

- The LOGID links the data base with the associated log file.
- TurboIMAGE logging uses MPE user logging.

Refer to the *TurboIMAGE Data Base Management System Reference Manual*, Appendix E, for TurboIMAGE log record formats.

□ Utilities and Transaction Logging

Purpose: To show a situation that supports logging.

Key Points:

- A system failure has occurred. The data base is probably corrupt, logically, physically, or both. All 5000 entries may have to be re-entered into the system. Why log?!!
- Data is one of the most important assets of a company. It is usually the responsibility of the data base administrator to protect data. Logging offers insurance against hardware and software failures, and the necessity of re-entering data.

Purpose: To define logging and the logging process.

Key Points:

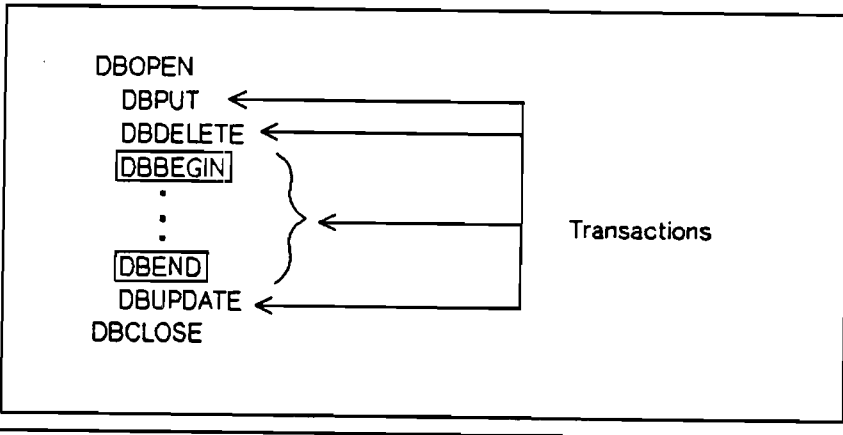
- TurboIMAGE logging is a facility used to recover logged transactions in case of lost or corrupted data.
- Logging does not place a heavy load on the system. Most system users will not even notice logging has been enabled. But, logging may be noticed by users whose programs modify a data base; the impact on response time depends on how tight the modification loop is. Additional memory will usually correct slow response time.
- Logging can be enabled without modification to user code. ** CLARIFY! Limited effectiveness because only physical integrity assured SEE NEXT SLIDE*
- Hard copies of log files are useful for auditing, debugging, and tuning. By analyzing the log files, statistics can be gathered to help answer performance-tuning questions. For example: logging allows user access to remote data bases to be easily monitored; large data bases can reduce backup frequency from daily to weekly; and, logging may be the only way to eliminate structural damage to the data base, if the time required for a DBUNLOAD and DBLOAD is out of the question.
- The log file can reside on either disc or tape.

Utilities and Transaction Logging

Notes

Logical Transactions

- DBBEGIN and DBEND define a logical transaction.

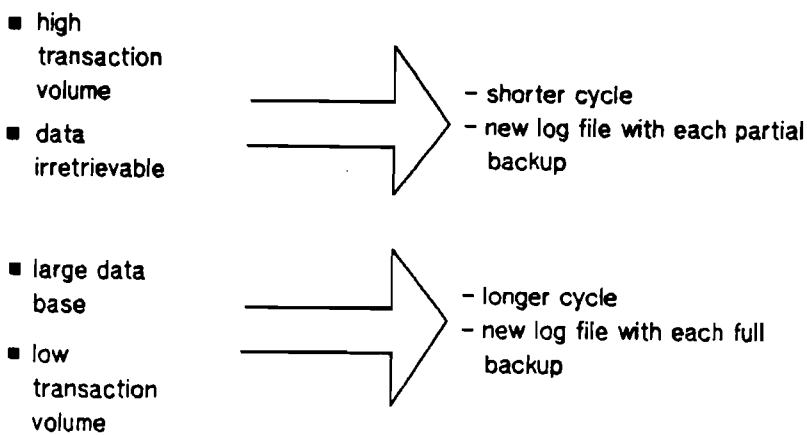


- Use DBMEMO to log user data to the log file.

Refer to the *TurboIMAGE Data Base Management Reference Manual*, Section 6: DBBEGIN, DBEND, and DBMEMO for syntax and further information.

The Logging Cycle

The logging cycle runs from backup to backup.



□ Utilities and Transaction Logging

Purpose: To define a logical transaction.

Key Points:

- A logical transaction is defined as a sequence of one or more procedure calls begun with a DBBEGIN and concluded with a DBEND. DBBEGIN and DBEND are used to surround a multiple-step transaction that would leave the data base inconsistent if a failure was to occur in the middle of the transaction; for example, add to sales, delete from inventory. If DBBEGIN and DBEND are not used, TurboIMAGE considers each DBPUT, DBDELETE, and DBUPDATE as a single logical transaction. * Continue from last slide - logical vs physical integrity
- Calls to DBOPEN, DBCLOSE, DBMEMO and log intrinsics built into TurboIMAGE procedures are also logged.
- While a transaction is executing, the data base could be in a temporarily inconsistent state. If the system fails while the data base is being modified, logical or structural damage can occur. A logical inconsistency might result if the failure occurs between modifications of a multiple-step transaction. Structural damage (such as broken chains) can result if the failure occurs during the execution of a TurboIMAGE procedure.

Purpose: To introduce the logging cycle and associated logging time estimates.

Key Points:

- How often a new log cycle is started depends on two things: how long it would take to process the log file should recovery need to take place, and the amount of time it takes to store the data base periodically.
- The more often the data base is stored, the smaller the log file and recovery time will be.
- The logging cycle heavily affects the operation of a data center. It must be well studied, well defined, and must have the backing of management.
- Data bases with a high transaction volume should have a new log file started with each partial backup.
- Low transaction volume data bases should have a new log file started with each full backup.

Length of recovery time less significant with ROLL BACK recovery

Logging Considerations



- Should be an integral part of computer systems operations.
- Never process after a system failure without recovering first.
- Must have management backing to be successful.
- Results in reliable data base integrity.
- Use DBINFO mode 401 to obtain information about current logging status.
- Always check status returned by DBOPEN, DBPUT, DBDELETE, DBUPDATE, DBBEGIN, DBEND, DBMEMO, and DBCLOSE.

Utilities and Transaction Logging

Purpose: To identify important logging considerations.

Key Points:

- DBINFO mode 401 provides the following information: the log identifier name, whether the data base is enabled for logging, whether a transaction is in progress, and the current transaction number. The amount of disc space left in the log file cannot be found with DBINFO mode 401.
- The DBUTIL >>SHOW command can be used to display the log identifier and the status of the flags for access, recovery, and logging.
- User programs should be written to identify TurboIMAGE logging errors, enabling an independent error file to be created and then causing the process to abort
- The overhead required by the logging operation is comparable on disc or tape.

In preparation for act'ing explain the need to initially allocate ALL extents in disc log file, otherwise there is a risk when writing to log and new extent is allocated there will be insufficient space to build new extent and log file will appear to be bad - logging will shut down & damage log file - may not be valid!

Module 8-16

Activity 8-5 Demo: Implementing Logging

Purpose: To implement TurboIMAGE logging in an on-line interactive session.

Directions:

1. Type :LOGGING to run the interactive session.
2. Refer to the *TurboIMAGE Data Base Management System Reference Manual*, Appendix G: Recovery and Logging Quick Reference, for assistance during the activity.
3. Upon completion of the activity, you can request a copy be sent to the line printer.

Module 8-16 Instructor Notes

Activity 8-5 Demo: Implementing Logging

Time: 20 minutes

Purpose: To implement TurboIMAGE logging in an on-line interactive session.

Notes: This activity is an on-line session requiring the student to set up a data base for TurboIMAGE logging, and then terminate the logging process.

Although the simulation requires students to use the REALTY data base as the data base involved in the logging process, please emphasize to the students that they are only simulating usage of the logging process; they are not actually affecting their own REALTY data base.

Informative responses will be provide for both correct and incorrect answers.

Each students can request a listing at the end of the activity. Encourage students to run the session individually.

Directions:

1. Make sure the UDC file ACCUDC.LABS has been set and released at the account level. The students will initiate this session by typing the UDC :LOGGING.
2. The questions and answers are in the file MOD8LOG.SOLUTION.IMAGE. This file will be sent to device class LP upon request of the student at the end of the session. A copy of this file is listed below.

Solutions:

Activity 8-5 Demo: Implementing Logging

Lesson I: Logging Installation

TurboIMAGE Logging, or Transaction Logging, insures logical integrity of a data base. When TurboIMAGE Logging is enabled, logical transaction are logged to a log file, built by the user, to be used in case of syste failure or abnormal termination.

A logical transaction is defined as a sequence of one or more modificati that transfer a data base from one consistent state to another.

In the case of a system failure or abnormal termination, the log file is used by DBRECOV, the TurboIMAGE recovery system, to bring the data base back to resemble its state before the interruption.

This lesson will lead you through the steps in setting up a data base fo TurboIMAGE Logging. Refer to the TurboIMAGE Reference Manual, Section 7 Logging Installation and Appendix G: Recovery and Logging Quick Refere

The 5 steps to install TurboIMAGE Logging are:

1. Acquire logging capability.
2. Acquire log identifier.
3. Build log file, if logging to disc.
4. Set log identifier into dat
5. Store a back-up copy.

(Continued on next page.)

Module 8-16 Instructor Notes

□ Activity 8-5 Demo: Implementing Logging (cont'd)

- Step 1. The person responsible for maintenance of a data base must have LG capability to implement logging. Both the account and the user must have LG capability. The ALTACCT command can be used to give LG capability at the account level:

```
:ALTACCT IMAGE;CAP=AL,GL,SF,ND,IA,BA,LG
```

Once this command has been issued, the ALTUSER command can be used to give LG capability to a user.

Enter the appropriate command to give LG capability to USER1. (Refer to the MPE Pocket Guide for syntax.)

```
:ALTUSER USER1;CAP=AL,GL,ND,SF,IA,BA,LG
```

As with the ALTACCT command, when using the ALTUSER command to add a capability, you must specify the existing capabilities along with the new one. Step 1 is now complete.

- Step 2. This step involves the definition of a system logging process to which log records are passed. This process is identified by a log identifier (logid). The logid is acquired from MPE using the GETLOG command as follows:

```
:GETLOG logid;PASS=password;LOG=logfile,devtype
```

where logid = an 8-character log identifier,
password = an 8-character password,
logfile = the MPE file to which log records are written,
devtype = TAPE or DISC.

Issue the GETLOG command using RLTYLOG as the logid, PASSLOG as the password, LGFILE as the logfile, and DISC as devtype.

```
:GETLOG RLTYLOG;PASS=PASSLOG;LOG=LGFILE,DISC
```

- Step 3. Since DISC was specified as the device type in the GETLOG command, a log file must be built. The BUILD command can be used as follows:

```
:BUILD logfile;CODE=LOG;DISC=numrec,numextents,initialloc
```

where logfile = the MPE file to which log records are written,
numrec = the maximum number of log records that can be written to the log file,
numextents = the maximum number of extents,
initialloc = the number of extents to be allocated initially to the file at the time it is opened.

Issue the BUILD command for the log file using 200000 as the number of records, 20 as the number of extents, and 20 as the number of extents initially allocated.

```
:BUILD LGFILE;CODE=LOG;DISC=200000,20,20
```

When building this file, it is important to ensure that the file is of sufficient size to prevent the end-of-file from being reached during the logging process. MPE does not automatically switch to a new disc file, but instead terminates the logging process when the log file is filled. This can result in a data base being left in an inconsistent state.

The command :SHOWLOGSTATUS can be used to determine when a log file is nearing capacity. The command :CHANGELOG can then be issued to open a new log file. These two commands will be presented in more detail in the lesson on logging maintenance.

(Continued on next page.)

Module 8-16 Instructor Notes

□ Activity 8-5 Demo: Implementing Logging (cont'd)

Step 4. The first three steps involved the MPE user logging system. Now the MPE logging system must be interfaced to TurboIMAGE by storing the log identifier and password into the data base root file. This is done through the DBUTIL >>SET command.

Issue the appropriate command to enter the LOGID into the REALTY data base.

```
:RUN DBUTIL.PUB.SYS
>>SET REALTY LOGID=RLTYLOG
PASSWORD? **PASSLOG**
LOGID: RLTYLOG IS VALID
      PASSWORD IS CORRECT
```

Now that the log identifier is set in the data base root file, we can enable the data base for logging, and proceed to Step 5 - storing a back-up copy. To do this, it is necessary to set three flags in the root file indicating the following:

1. the data base is enabled for logging
2. the data base is disabled for user access
3. the data base is enabled for recovery

The flags indicating items 2) and 3) are necessary to prepare the data base for Step 5 - storing a back-up copy.

Issue the command to enable the data base for logging.

```
>>ENABLE REALTY FOR LOGGING
WARNING: Data Base modified and not DBSTORED Logging is enabled
```

(NOTE: This warning should be interpreted as a prompt to store the data base.)

Before storing the data base, access should be disabled so that in the event of a system failure, the data base will be restored with access disabled. This will prevent users from modifying the data base before recovery is executed.

```
>>DISABLE REALTY FOR ACCESS
Access is disabled
```

Before the data base is stored, it must be enabled for recovery. Then, when the data base is restored, it will be ready for recovery.

```
>>ENABLE REALTY FOR RECOVERY
Recovery is enabled
```

What DBUTIL command can you issue to verify that the flags are set appropriately, and that the log identifier is set in the data base?

```
>>SHOW REALTY ALL
For data base REALTY
```

Maintenance word is present.

```
LOGID=RLTYLOG
ACCESS is disabled
AUTODEFER is disabled
ROLLBACK recovery is disabled
DUMPING is disabled
LOGGING is enabled
RECOVERY is enabled
ILR is disabled
```

(Continued on next page.)

Module 8-16 Instructor Notes

□ Activity 8-5 Demo: Implementing Logging (cont'd)

Data base last stored on MON, MAY 10, 1984, 1:09 PM
Data base has been modified since last store date.
Subsystem access is READ/WRITE.

LOGID: RLTYLOG is valid password is correct

Buffer specifications: 25(1/120)

No other users are accessing the data base

- Step 5. Once the flags and logid have been verified, the DBSTORE can be performed. Issue the command to EXIT the DBUTIL utility and the command to store a back-up copy of the data base.

```
>>EXIT  
END OF PROGRAM
```

```
:RUN DBSTORE.PUB.SYS  
WHICH DATA BASE? REALTY
```

```
DATA BASE STORED  
END OF PROGRAM
```

Lesson 2: Logging Maintenance

Logging maintenance is very important to the integrity of a data base. The data base administrator should determine a logging maintenance cycle for stopping the logging process, storing the data base, and then starting the logging process again.

This lesson will lead you through the steps necessary to properly maintain the logging process.

Part One) Starting the Logging Process

After a data base back-up copy has been stored as described in Lesson 1: Logging Installation, a logging process must be allocated to the log identifier. A logging process is an MPE system process responsible for buffering log records in memory. This process is initiated from the console with the :LOG command.

The syntax for the :LOG command is:

```
START - initiates a log process.  
:LOG logid, RESTART - appends log records to an old log file.  
STOP - terminates a log process.
```

Issue the command to start the log process.

```
:LOG RLTYLOG,START
```

(Continued on next page.)

Module 8-16 Instructor Notes

□ Activity 8-5 Demo: Implementing Logging (cont'd)

Part Two) Preparing the Data Base for Access

After installing the logging process, making a back-up copy of the data base, and initiating the logging process, the data base can be returned to a ready-for-access state.

Use DBUTIL to set the flags for access and recovery.

```
:RUN DBUTIL:PUB.SYS
>>ENABLE REALTY FOR ACCESS
  Access is enabled
>>DISABLE REALTY FOR RECOVERY
  Recovery is disabled
>>EXIT
  END OF PROGRAM
```

Part Three) Monitoring the Logging Process

Once a log process has been started, the MPE command :SHOWLOGSTATUS can be used to gain information about all active log processes.

```
:SHOWLOGSTATUS
```

LOGID	USERS	STATE	RECORDS	%USED	TOTAL
disclog	1	INACT	120	90%	400
tapelog	1	ACTIVE	100		
ritylog	1	INACT	35	5%	500

The ACTIVE state is displayed when a user process is logging information to the file. The INACTIVE state is displayed when a process is waiting for information from a user process.

The SHOWLOGSTATUS command is useful for determining when a disc log file is nearing capacity.

TurboIMAGE provides two features for ensuring a continuous MPE logging process, with the ability to change log file tape or disc files when they reach capacity, without stopping the user logging process. The order of the files will be kept in a log file set.

1. The CHANGELOG command.

```
:CHANGELOG logid;DEV=device
```

where logid - name of the currently active logging process. device - device name of the new log file.

For further details on the CHANGELOG command, refer to the TurboIMAGE Reference Manual, Section 7: Changelog Capability.

NOTE: The CHANGELOG command is not available with IMAGE/3000.

(Continued on next page.)

Module 8-16 Instructor Notes

□ Activity 8-5 Demo: Implementing Logging (cont'd)

2. The AUTO option of the GETLOG command.

```
:GETLOG RLTYLOG;PASS=PASSLOG;LOG=RLOG001,DISC,AUTO
```

When the disc log file becomes full, the AUTO option will initiate a CHANGELOG. A new log file will be created with the same name incremented by one in the last digit (RLOG002).

For further details on the AUTO option, refer to the TurboIMAGE Reference Manual, Section 7: Acquiring Log Identifier.

NOTE: The AUTO option of the GETLOG command is not available with IMAGE/3000.

Refer to the MPE Commands Reference Manual for other MPE user login commands, including:

```
:RELLOG      - removes a log identifier
:ALTLOG      - alters an existing log identifier
:LISTLOG     - lists the current log identifiers
```

Part Four) Stopping the Logging Process

At the end of the specified maintenance cycle, the maintenance steps are reversed as follows:

```
:LOG RLTYLOG,STOP
:RUN DBUTIL.PUB.SYS
>>DISABLE REALTY FOR ACCESS
  Access is disabled
>>ENABLE REALTY FOR RECOVERY
  Recovery is enabled
>>EXIT
END OF PROGRAM
```

```
:RUN DBSTORE.PUB.SYS
WHICH DATA BASE? REALTY
DATA BASE STOPED
END OF PROGRAM
:
```

Module 8-17

Activity 8-6 Chalk Talk: Logging Media

Purpose: To identify some of the advantages and disadvantages of logging to disc and logging to tape.

Directions: As your instructor describes the advantages and disadvantages of the two types of logging media, write down the associated key points.

I. Logging to Tape

A. Advantages:

B. Disadvantages:

II. Logging to Disc

A. Advantages:

B. Disadvantages:

Module 8-17 Instructor Notes

□ Activity 8-6 Chalk Talk: Logging Media

Time: 20 minutes

Purpose: To identify some of the advantages and disadvantages in logging to disc and logging to tape.

Directions: Discuss the advantages and disadvantages of logging to tape and logging to disc. Use this activity to summarize the logging process and to answer any questions on logging media.

I. Logging to Tape:

A. Advantages:

- *Logging to tape does not require any disc space.*
- *Tape is more secure than disc in terms of a hard or head crash.*

B. Disadvantages:

- *When the system reaches the end of a log tape, it asks the operator to mount another reel. If the operator ignores the request, the processes that require logging will suspend within a few minutes and logging will stop. Applications requiring logging will get a WRITELOG error and will terminate. No message is printed on the user's terminal.*
- *If the system operator responds to a new log tape request, the operator must be sure a new log tape has been mounted. The request for a new reel requires only that the tape drive be put on-line, there is no reply. Logging to the same tape overwrites the old tape (no security provision) wiping out the logged transactions.*
- *Logging to tape requires a dedicated tape drive for the entire day. The tape cannot be dismounted for another use. A reliable tape drive and a library of "good" tapes must be maintained.*
- *Logging to tape is more time consuming than logging to disc. After a system failure, the tape must be rewound and sequentially scanned until the end of file is detected. Remaining records (contents of the disc buffer file) can then be appended to the active log tape. When a failure occurs, hit RESET, LOAD, and ON-LINE buttons on the tape drive to write the EOF marker at the end of the tape.*
- *Logging to tape is subject to power failure complications.*

(Continued on next page.)

Module 8-17 Instructor Notes

Activity 8-6 Chalk Talk: Logging Media (cont'd)

II. Logging to Disc

A. Advantages:

- *Logging to disc does not require a dedicated tape drive.*
- *Four new MPE commands have been introduced with the TurboIMAGE MIT, to make logging to disc much simpler to carry out.*
- *The :CHANGELOG command allows a new log file to be created on disc when the current one becomes full. This provides continuous logging without interruption, as opposed to the current situation in which the logging process shuts down when the log file becomes full.*
- *The :GETLOG,AUTO command option can be used to set up the automatic switching of a disc log file when it becomes full, without user intervention. (The :CHANGELOG command can be re-issued by the user to accomplish this same task.)*
- *The :SHOWLOGSTATUS command can be used to determine whether or not a log process is running. :SHOWLOGSTATUS will display the percentage of records in the log file.*
- *The :LISTLOG command will list the current log file name.*

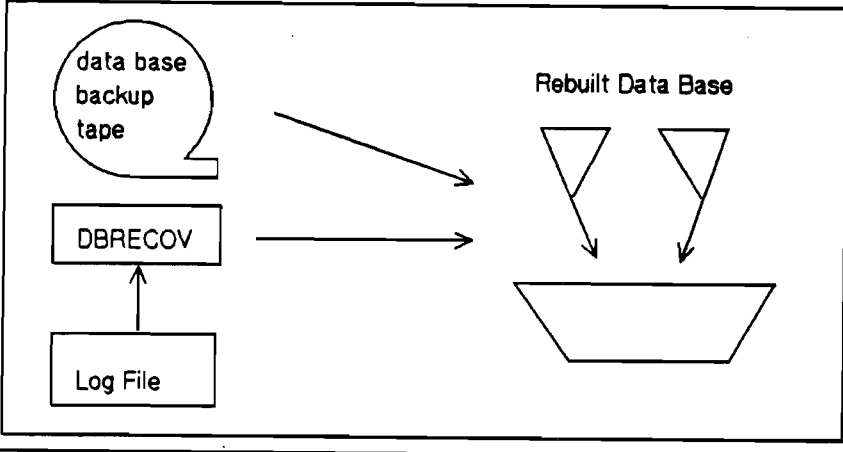
B. Disadvantages:

- *The disc log file must be created with CODE=LOG and with a record size of 128 words. The log file should be built with a capacity larger than anticipated, with all extents allocated.*
For example, :BUILD LOGFILE;DISC=100000,8,8;CODE=LOG.
- *The integrity of the log file may be no better than the current data base state, as disc files are susceptible to a hard crash.*
- *To maintain a good audit trail, at least 5 to 10 log files on store tapes should be kept.*
- *An active log file cannot be backed up.*

Broak.

DBRECOV

- Uses latest data base backup and log files to rebuild damaged data base.



:RUN DBRECOV.PUB.SYS



- >CONTROL - specifies options affecting execution.
- >EXIT - terminates DBRECOV.
- >FILE - requests user recovery files.
- >PRINT - prints pre-recovery information.
- >RECOVER - defines data base for roll-forward recovery.
- >ROLLBACK - defines data base for roll-back recovery.
- >RUN - initiates recovery process.

- >CONTROL options are covered in Activity 8-7.
- >RECOVER command is presented in slide 8.14.
- >ROLLBACK command is presented in slide 8.16.

□ Utilities and Transaction Logging

Purpose: To introduce the DBRECOV utility program.

Key Points:

- The DBRECOV program usually is executed after a data base backup copy has been restored by running DBRESTOR after a system failure.
- DBRECOV reads the log file containing records of all data base modifications and re-executes the transactions against the restored data base. The DBRECOV file facility enables individual users to be informed of the extent of recovery.
- DBRECOV checks the root file to determine that: the LOGID in the root file matches the log file; recovery is enabled for the data base; and, a STORE flag has been set by DBSTORE. DBRECOV also checks to ensure that the timestamp in the log file is greater than or equal to the timestamp of DBSTORE on the DBSTORE'd data base.

Purpose: To introduce the DBRECOV commands.

Key Points:

- >EXIT terminates DBRECOV without recovering any data base(s).
- >FILE routes log records to individual user files, providing the application program with information about the outcome of recovery. >FILE also serves as a useful tool for auditing previous entries.
- >PRINT prints data bases or files specified for recovery. >PRINT can be used as a check before actually initiating recovery with the >RUN command.
- >RUN actually initiates the recovery process. The recovery system opens the log file and validates the log identifier before recovery begins.

Module 8-19

□ Activity 8-7 Chalk Talk: DBRECOV>CONTROL Command

Purpose: To present the options available with the DBRECOV >CONTROL command.

Directions: As your instructor describes each of the command options write down key points. Refer to the *TurboIMAGE Data Base Management System Reference Manual*, Section 8: DBRECOV>CONTROL, for details of each option.

I. Default Conditions:

STAMP =

STORE =

ABORTS =

ERRORS = nnn

MODEX =

NOSTATS =

STOPTIME =

EOF = nnnn

II. Reverse Default Conditions:

NOSTAMP =

NOSTORE =

NOABORTS =

STATS =

MODE4 =

Module 8-19 Instructor Notes

Activity 8-7 Chalk Talk: DBRECOV>CONTROL Command

Time: 10 minutes

Purpose: To define the options available with the DBRECOV >CONTROL command.

I. Default Conditions:

STAMP = *timestamp*. STAMP must correspond to the log file.

STORE = *DBSTORE* flag set in the data base root file.

ABORTS = *failed transactions will be recovered*.

ERRORS = *nnnn*; *maximum non-fatal errors in the job*.

MODEX = *recovery executes in exclusive mode*.

NOSTATS = *tabulated information printed only if data base recovered*.

STOPTIME = *DBRECOV will not check log record timestamps that are greater than STOPTIME*.

EOF = *nnnn*; *DBRECOV will recover all log records*.

II. Reverse Default Conditions:

- *Additional >CONTROL commands are: NOSTAMP, NOSTORE, NOABORTS, STATS, and MODE4. MODE4 allows the recovery process to proceed in DBOPEN mode 4, allowing users in mode 6 to access (read only) the data base while recovery is in process. All additional commands reverse default conditions.*

III. General DBRECOV Information:

- *If the >CONTROL command is not called, the default conditions will apply.*
- *The >CONTROL command is used to override the default conditions.*
- *If a particular parameter is not specified within a >CONTROL command, the default condition remains in effect.*

Roll-forward Recovery

>RECOVER data base name [/maintenance word] [.group] [.account]

- uses backup copy and log file
- backup copy must match log file (DBSTORE timestamp and flag)
- data base must be enabled for recovery
- recovers only completed transactions
- one or more records could be lost if system buffers not posted to disc before system failure

T11 8.14

© 1985 Hewlett-Packard Company

Intrinsic Level Recovery (ILR)

- Guarantees structural integrity of data base in event of system failure.

DBOPEN

DBGET

← IMAGE/3000 undoes interrupted intrinsic

DBPUT ← System Failure

← TurboIMAGE completes interrupted intrinsic

DBDELETE

DBEND

- The most recent DBPUT and DBDELETE are logged to the ILR file.
- ILR, if enabled, is performed automatically.

T11 8.15

© 1985 Hewlett-Packard Company

Module 8-20 Instructor Notes: Slides 8.14/8.15

□ Utilities and Transaction Logging

Purpose: To present the advantages and disadvantages of Roll-Forward Recovery.

Key Points:

- The roll-forward recovery system can be executed to bring data bases back to a likeness of their state at the time of a hard system failure.
- The >RECOVER command does not initiate recovery, but makes several preparatory checks. The >RUN command actually initiates recovery.
- The >RECOVER command is used to: designate the name of the data base to be roll-forward recovered; open the data base root file; validate the LOGID and password with MPE; and, check the DBSTORE flag.
- The log identifier characteristics (name, password, log file name, and device type) must not have been altered since the log file was generated.
- The >RECOVER command will not be accepted if the LOGID is unknown to MPE. If the LOGID is known to MPE but specifies the wrong log file, this condition IS NOT SENSED at the time and >RECOVER will be accepted. DBRECOV will generate erroneous data in the data base if the data base is recovered with the wrong log file.
- The DBSTORE flag must be set, indicating that the data base has not been modified between restoration and roll-forward recovery. (This check can be overridden by the NOSTORE options of the >CONTROL command.)

Purpose: To define Intrinsic Level Recovery (ILR).

Key Points:

- If a data base is enabled for ILR, upon the first DBOPEN of the data base, the ILR file will be examined. If recovery is necessary, the buffers logged to the ILR file are posted and the interrupted intrinsic is completed for TurboIMAGE and undone for IMAGE/3000. In this way, the TurboIMAGE data base is brought back to a state as if the intrinsic had completed normally. Chains are reconstituted automatically so that the internal structure of the data base remains consistent.
- ILR and AUTODEFER cannot occur at the same time, since AUTODEFER does not insure that modified buffers will be posted in the event of a system failure.
- ILR logging is invisible to the user. Since ILR is only concerned with data base structure, only the most recent DBPUT or DBDELETE is noted in the ILR file. Recovery after a system failure requires no more overhead than a single DBPUT or DBDELETE. *3 extra I/O's per IMAGE call*
- ILR is enabled with the >>ENABLE command of the DBUTIL utility. *2 extra I/O's per TurboIMAGE call*
- TurboIMAGE does not allow a program abort to interrupt a DBPUT or DBDELETE, therefore ILR is not needed following an abnormal program termination (user program abort).

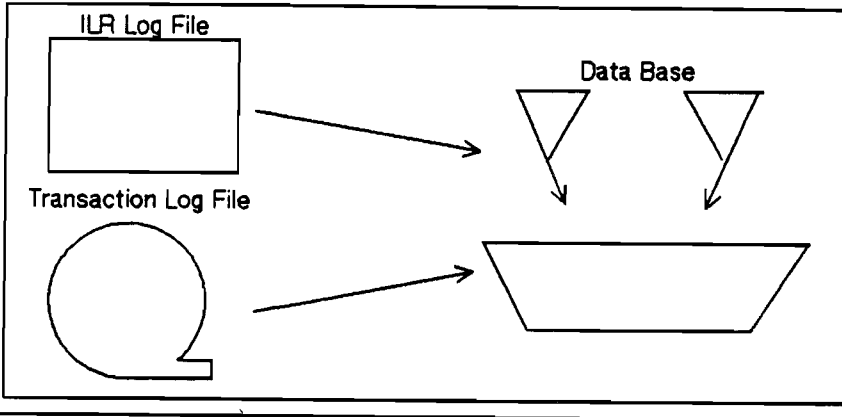
Talk now about overheads of logging versus ILR overheads. IO overhead

Roll-Back Recovery



>ROLLBACK data base name [/maintenance word][.group[.account]]

- uses CURRENT data base and log files



- Roll-back recovery is available with TurboIMAGE only.
- Roll-back does not require a backup, and is a faster method of recovery than roll-forward.
- With roll-back, all interrupted transactions are undone.
- ILR is required, and AUTODEFER is not allowed, when roll-back is enabled.

Module 8-21 Instructor Notes: Slide 8.16

Utilities and Transaction Logging

Purpose: To define Roll-Back Recovery.

Key Points:

- Roll-Back recovery guarantees the logical integrity of a data base in the event of a system failure. When roll-back is enabled for a data base and recovery is performed, any TRANSACTIONS that were interrupted by the system failure are undone. Therefore, the data base is brought back to the logically and physically consistent state it was in prior to the system failure.
- Roll-Back recovery makes use of the current state of the data base and user log files. Therefore, a backup copy of the data base at recovery time is not necessary, although regular backups are still recommended.
- Roll-Back is quicker than roll-forward, since undoing incomplete transactions is faster than redoing all completed transactions since the last backup. Additional time is saved since a DBRESTOR of an old copy of the data base is not required.
- Roll-Back recovery is enabled through DBUTIL. ILR is required when roll-back is used, and is automatically enabled when a data base is enabled for roll-back. However, the user is still responsible for enabling logging for the roll-back method to work.

ILR overhead reduced from 3 to 2 with Turbo but still costly. Trade and balance required on critical systems.

Module 8-22

Activity 8-8 Demo: Implementing Data Base Recovery

Purpose: To implement the Roll-Forward, Roll-Back, and ILR recovery methods.

Directions:

1. Type :RECOVERY to run the interactive session.
2. Refer to the *TurboIMAGE Data Base Management System Reference Manual*, Appendix G: Recovery and Logging Quick Reference, for assistance during the activity.
3. Upon the completion of the activity, you can request a copy be sent to line printer.

Notes:

Additional comments on ILR:

1. Logging the intrinsics increases the overhead on each DBPUT and DBDELETE.
2. After system failure, ILR requires no more overhead than a single DBPUT or DBDELETE.
3. Output deferred cannot be used with ILR.
4. ILR is intrinsic driven, therefore transaction locking is not necessary.
5. The RESTART option in user logging can be used.
6. ILR is automatic and transparent to the user.

Additional comments on Roll-Forward Recovery:

1. Roll-forward recovery provides structural and logical recovery.
2. Roll-forward recovery uses logging to recover both intrinsics and transactions.
3. Roll-forward recovery does not require ILR be enabled, however, enabling ILR is recommended to eliminate the possibility of broken chains.
4. Roll-forward recovery does not require logical transaction locking, however, it is recommended.

Additional comments on Roll-Back Recovery:

1. Roll-back recovery provides rapid recovery of data base integrity.
2. Logging is required, however, only the current log file is required to restore data base integrity. (Logging is enabled automatically.)
3. ILR is required to ensure correctness of physical data links. (ILR is enabled automatically.)
4. When roll-back is disabled, ILR and logging must be disabled manually.
5. Roll-back requires all multiple-intrinsic data base transactions execute independently, using logical transaction locking.

Module 8-22 Instructor Notes

Activity 8-8 Demo: Implementing Data Base Recovery

Time: 20 minutes

Purpose: To implement the Roll-Forward, **Roll-Back**, and ILR recovery methods.

Notes: This activity is an on-line session requiring the student to recover the REALTY data base using Roll-Forward, Roll-Back, and ILR.

Although the simulation requires students to use the REALTY data base as the data base involved in recovery, please emphasize to the students that they are only simulating usage of the recovery methods; they are not actually affecting their own REALTY data base.

Informative responses will be provide for both correct and incorrect answers.

Each students can request a listing at the end of the activity. Encourage students to run the session individually.

Directions:

1. Make sure the UDC file ACCUDC.LABS has been set and released at the account level. The students will initiate this session by typing the UDC :RECOVERY.
2. The questions and answers are in the file MOD8REC.SOLUTION.IMAGE. This file will be sent to device class LP upon request of the student at the end of the session. A copy of this file is listed below.

Solutions:

Activity 8-8 TurboIMAGE Recovery Options

Introduction to TurboIMAGE Recovery Options

TurboIMAGE provides two levels of data base recovery:

Intrinsic Level Recovery (ILR) and
Transaction Level Recovery (Roll-Forward and Roll-Back).

These recovery options ensure the physical and logical integrity of the data base following a system interruption.

The level of recovery is determined by the data base administrator and is based on available data base back-up and logging resources as well as user performance requirements.

Refer to the TurboIMAGE Reference Manual, Appendix G. Recovery and Logging Quick Reference for information about recovery options.

Lesson 1: Intrinsic Level Recovery (ILR)

Intrinsic Level Recovery provides recovery of intrinsics that were interrupted during execution. ILR is automatic and transparent to the user. When ILR is enabled, TurboIMAGE automatically logs each DBPUT and DBDELETE to an internal ILR file. Since ILR is only concerned with data base structure, only the most recent DBPUT/DBDELETE is noted in the ILR file.

(Continued on next page.)

Module 8-22 Instructor Notes

□ Activity 8-8 Demo: Implementing Data Base Recovery

ILR is enabled using the DBUTIL utility. Initiate execution of DBUTIL and enable the data base for ILR.

```
:RUN DBUTIL.PUB.SYS
>>ENABLE REALTY FOR ILR
```

When ILR is enabled, TurboIMAGE builds and initializes a privileged ILR file for the data base. It is given the same name as the root file appended by two ASCII zeros (REALTY00).

Enter a DBUTIL command to display the status of the data base flags to verify that ILR is enabled.

```
>>SHOW REALTY FLAGS
```

```
For data base REALTY
```

```
Access is enabled.
Autodefer is disabled.
Rollback recovery is disabled.
Restart is disabled.
Dumping is disabled.
Logging is disabled.
Recovery is disabled.
ILR is enabled.
Subsystem access is READ/WRITE
```

```
>>EXIT
END OF PROGRAM
```

Lesson 2: Roll-Forward Recovery

The roll-forward recovery option recovers the physical and logical integrity of a data base by suppressing any incomplete transactions. The data base is then restored to the state at which logging was enabled, and the transactions in the log file are re-executed.

The steps to implement roll-forward recovery are:

1. Restore the back-up copy of the data base.
2. Use roll-forward recovery on the data base.
3. Re-start logging and set data base flags.

Step 1) Requires that a back-up copy of the data base be restored. The back-up copy of the data base should be one that was stored before the start of the current logging cycle.

Before restoring the back-up copy, the current data base must be purged using DBUTIL.

Initiate execution of DBUTIL and purge the damaged data base.

```
:RUN DBUTIL.PUB.SYS
>>PURGE REALTY
  Data base REALTY has been PURGED
>>EXIT
END OF PROGRAM
```

Enter the command to restore the back-up copy of the data base.

```
:RUN DBRESTOR.PUB.SYS
WHICH DATA BASE? REALTY
DATA BASE RESTORED
END OF PROGRAM
```

(Continued on next page.)

Module 8-22 Instructor Notes

□ Activity 8-8 Demo: Implementing Data Base Recovery

Step 2) Use roll-forward recovery on the data base.

Assume that a hard system failure has occurred. After bringing up the operating system, the applicable log file media must be located to prepare for roll-forward recovery. If the log file resides on disc, TurboIMAGE will locate the log file by checking the beginning of the root file for the logid.

Enter the command to execute the DBRECOV program.

```
:RUN DBRECOV.PUB.SYS
```

Enter the DBRECOV command to recover the data base.

```
>RECOVER REALTY  
DATABASE REALTY LAST DBSTORED MON, MAY 20, 1985, 8:45 AM
```

The RECOVER command of DBRECOV does not initiate recovery. It checks data base flags and security to make sure the data base is in the correct state for recovery.

If all conditions are met, the RECOVER command is accepted.

Enter the command to actually initiate the recovery system.

```
>RUN  
>EXIT  
END OF PROGRAM
```

After the >RUN command is given, the DBRECOV program will recover the REALTY data base and then terminate.

Step 3) Re-start logging and set data base flags.

After recovery has completed, logging can either be restarted from the current log file, or the log file can be purged and a new log file built.

Enter the command to RESTART logging from the current log file.

```
:LOG RLTYLOG,RESTART
```

The data base can now be made available to users. Set the appropriate flags in DBUTIL to allow access to the data base and to disallow unintended recovery.

```
:RUN DBUTIL.PUB.SYS  
>>ENABLE REALTY FOR ACCESS  
Access is enabled  
>>DISABLE REALTY FOR RECOVERY  
Recovery is disabled  
>>EXIT  
END OF PROGRAM
```

(Continued on next page.)

Module 8-22 Instructor Notes

□ Activity 8-8 Demo: Implementing Data Base Recovery

Lesson 3: Roll-Back Recovery

Roll-Back Recovery provides rapid recovery of data base integrity follow a "soft" system crash, such as a system failure.

When invoked, the roll-back recovery feature will "roll-back", or undo, any incomplete data base transactions.

Roll-Back Recovery is available with TurboIMAGE only. The steps to implement the roll-back process are:

1. Enable the roll-back feature.
2. Use roll-back recovery on the data base.
3. Disable roll-back recovery.

Step 1) Enable the roll-back feature

Prior to enabling the data base for roll-back, the data base must be enabled for logging. The steps to set up logging were performed in Activity 8-5, and will not be performed here.

The roll-back feature is enabled through the DBUTIL program.

Execute the DBUTIL program and enable the REALTY data base for ROLLBACK.

```
:RUN DBUTIL.PUB.SYS
>>ENABLE REALTY FOR ROLLBACK
Rollback is enabled
```

To ensure correctness of the data base physical data links, ILR must be enabled when using roll-back. With ILR enabled, when a system failure occurs, TurboIMAGE will automatically reconstitute chains.

DBUTIL will automatically enable logging and ILR when roll-back is enabled.

Since roll-back recovery enables ILR, output deferred mode (AUTODEFER) cannot concurrently be enabled on the data base.

Enter a DBUTIL command to verify that the appropriate flags are set.

```
>>SHOW REALTY FLAGS
```

```
For data base REALTY
```

```
Access is enabled.
Autodefer is disabled.
Rollback recovery is enabled.
Logging is enabled.
Restart is disabled.
Dumping is disabled.
Recovery is disabled.
ILR is enabled.
Subsystem access is READ/WRITE.
```

```
>>EXIT
END OF PROGRAM
```

(Continued on next page.)

Module 8-22 Instructor Notes

Activity 8-8 Demo: Implementing Data Base Recovery

Step 2) Use roll-back recovery on the data base.

Assume a system failure occurred while modifications were being made to your REALTY data base. You were logging to disc at the time, therefore TurboIMAGE will locate automatically the applicable log file.

Enter the command to execute the DBRECOV program.

```
:RUN DBRECOV.PUB.SYS
```

Enter the command to recover the data base.

```
>ROLLBACK REALTY
```

Issue the CONTROL command which allows users read access to the data base while recovery is in process.

```
>CONTROL MODE4
```

Enter the command to initiate the recovery process.

```
>RUN
```

Exit the DBRECOV program.

```
>EXIT
```

Step 3) Disable Roll-back Recovery

To disable the roll-back feature, the data base must be in a quiet state, with no user activity in progress.

The roll-back feature should not be disabled if it must be used later. Disabling roll-back will reset the logging time stamp to the time of the next DBOPEN. Therefore, recovery cannot be performed with the current log file.

The roll-back feature is disabled through the DBUTIL program.

Execute the DBUTIL program and disable the REALTY data base for ROLLBACK.

```
:RUN DBUTIL.PUB.SYS
```

```
>>DISABLE REALTY FOR ROLLBACK
```

```
WARNING:ROLLBACK time stamp will be erased.
```

```
Please type Y to confirm your disable command>> Y
```

```
Rollback is disabled
```

The WARNING message serves as a reminder to you. If you will be using roll-back at a later time, a back-up of the data base should be performed immediately after disabling roll-back. If roll-back is not going to be used any longer, disregard the WARNING.

Entering a Y at the WARNING prompt, will disable roll-back. If Y is not entered, then the DISABLE command is not performed.

Module 8-23

Activity 8-9 Worksession: Logging and Recovery

Purpose: To review transaction logging and recovery options.

Directions: Answer each of the following questions. Refer to your workbook or the TurboIMAGE Reference Manual if necessary.

1. Which of the following media can you log to when using transaction logging: tape, system domain disc, serial disc, or private volume disc?
2. TurboIMAGE logging uses MPE user logging. T F
3. All TurboIMAGE calls can produce a log record. T F
4. Which two TurboIMAGE procedures define a logical transaction?
5. Without procedures that define a transaction, no logging takes place. T F
6. List the steps to initialize TurboIMAGE logging the first time.
7. When building a log file on disc, which name is used?
8. When starting the log process with the LOG command, which LOGID is used?
9. What should be the status of the flags in the root file in order to process data base applications?
10. What should be the status of the flags in the root file to back-up the data base?
11. Why should the flags be set this way before back-up?
12. How can the status of the flags in the root file be displayed?
13. What method of back-up is recommended?

(Continued on next page.)

Module 8-24

Activity 8-9 Worksession: Logging and Recovery

14. Why should the backup be performed in this manner?
15. How would one tell that a disc log file is approaching capacity?
16. What two TurboIMAGE options are available to ensure a continuous logging process?
17. When logging to tape, what happens when the reel fills up?
18. What is the basic difference between roll-forward and roll-back recovery options?
19. Which is faster - roll-forward or roll-back? Why?
20. The data base should be purged using DBUTIL before DBRESTOR. T F
21. What should be the status of the flags for recovery?
22. How does DBRECOV check for the correct log file?
23. What command from DBRECOV executes recovery?
24. Does ILR maintain data base logical or structural integrity?
25. What is logged to the ILR log file?

Break

Module 8-24 Instructor Notes

Activity 8-9 Worksession: Logging and Recovery

Purpose: To review the transaction logging and recovery options.

Time: 30 minutes

Directions:

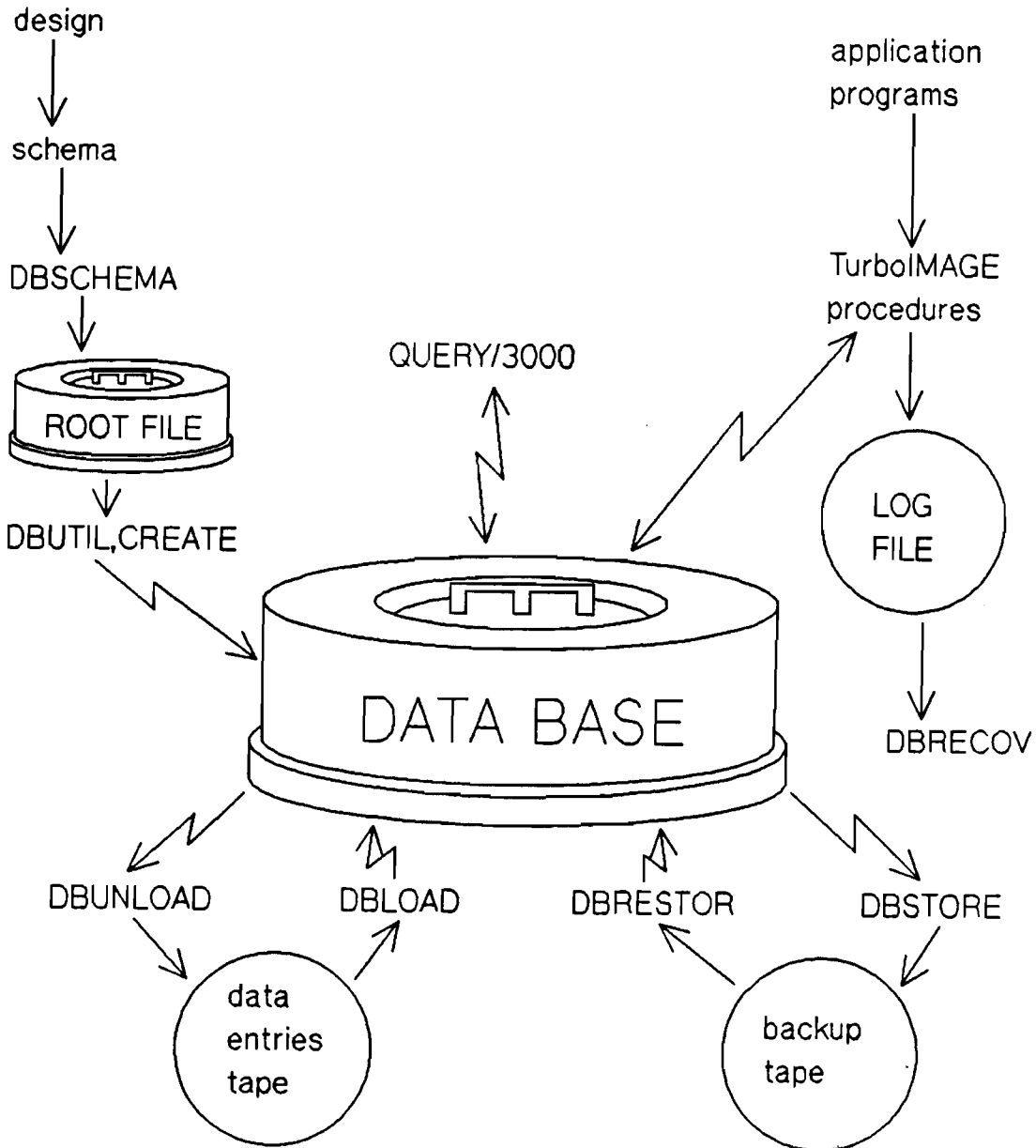
1. Have each student complete this activity individually.
2. When all students have completed the activity, go over the answers with the class.

Answers:

1. tape, system domain disc, private volume disc, serial disc
2. True
3. False
4. DBBEGIN and DBEND
5. False *Like capability on Acct, users*
6. ✓ GETLOG, BUILD logfile, DBUTIL - SET LOGID, DBUTIL - SET FLAGS, DBSTORE
7. Same as GETLOG command log file name
8. Same as GETLOG command logid
9. Enable for access, disable for recovery
10. Enable for recovery, disable for access
11. Because flags will be set correctly for recovery after DBRESTOR
12. >>SHOW REALTY ALL or >>SHOW REALTY FLAGS
13. DBSTORE
14. DBSTORE sets both the timestamp and the STORE flag in the root file
15. LISTF, 2 or :SHOWLOGSTATUS
16. :CHANGELOG or :GETLOG,AUTO
17. A message appears on the system console requesting another tape be mounted.
18. Roll-forward completes interrupted transactions and roll-back undoes the transactions.
19. Roll-back is faster since it does not require the use of DBSTORE and DBRESTOR. The transactions in the log file are not re-executed.
20. True
21. Enabled for recovery, disabled for access, enabled for logging.
22. Timestamp in log file must be greater than or equal to that of the root file.
23. >RUN
24. structural integrity
25. The most recent DBPUT or DBDELETE



TurboIMAGE: The Whole Picture



Module 8-25 Instructor Notes: Slide 8.17

Utilities and Transaction Logging

Purpose: To present an overview of all components of TurboIMAGE.

Key Points:

- The design of a data base is accomplished by a data base design team. Details of the design phase are presented in Module 9.
- The person that creates the rootfile is designated as the data base creator. This person can bypass data base security when accessing the data base.
- Application programmers and users create and use the application programs. They can also access the data base using QUERY/3000.
- The data base creator, or other user with the data base maintenance word, can perform restructuring (DBLOAD, DBUNLOAD), backup (DBRESTOR, DBSTORE), and recovery (DBRECOV) functions.

Module 8-26 Instructor Notes

Preparation Material: Slides 8.01, 8.04

Teaching Tips:

Mention each TurboIMAGE utility and describe its main function. Syntax and peculiarities of the utilities will be covered on the following slides.

Teaching Tips:

Go over the slide diagram in detail. Describe how DBUNLOAD operates in chain mode.

Draw an example on the chalkboard, showing DBUNLOAD in serial mode. Draw a number of data sets, each having a few entries. Draw an empty file, into which the entries will be loaded serially. Show how the entries are loaded serially into the empty file.

Module 8-27 Instructor Notes

Preparation Material: Slide 8.05

Teaching Tips:

Ask students to identify possible data base structural changes; i.e. adding items, changing capacities, etc.

Module 8-28 Instructor Notes

Preparation Material: Page 8-10

Teaching Tips:

Review the lists of supported and unsupported design changes. Encourage students to question or add to the lists as they see fit.

Module 8-29 Instructor Notes

Preparation Material: Slides 8.10, 8.17

Preparation:

For example, suppose the data base is maintained on a daily cycle. This means that at the beginning of each day, the log process is initiated from the console with the :LOG command and flags are set by the data base administrator. At the end of the day, the console operator stops the log process and the administrator resets the flags for storage of the backup data base.

Teaching Tips:

Use this slide to review all components of TurboIMAGE. Point out each of the functions illustrated on the slide. Be sure that all questions are answered at this point.



Module 9 Instructor Notes

□ Performance and Design

Overview of Module 9

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. DBMS Implementation (1 hour 5 minutes)

Activity: Chalk Talk: Overview of DBMS Implementation

Purpose: To review concepts in implementing a data base management system and to review what is needed to write a data base application program.

Slides: 9.01 - TurboIMAGE Design Considerations *
9.02 - Design Team *
9.03 - Types of Designs *
9.04 - Define and Prioritize Global Concerns *

Activity: Chalk Talk: Design Trade-offs

Purpose: To review data base design trade-offs.

Lesson 2. Design Approach (1 hour 20 minutes)

Slide: 9.05 - Top-Down Design Approach

Activity: Chalk Talk: Top-Down Design Approach

Purpose: To apply a top-down design approach to a given residential listing application.

Activity: Chalk Talk: Common Design Decisions

Purpose: To present common design decisions that occur during data base design.

Lesson 3. Performance (2 hours 50 minutes)

Slide: 9.06 - TurboIMAGE Performance

Activity: Chalk Talk: Performance

Purpose: To present TurboIMAGE impact on resources.

Slide: 9.07 - Efficient Programming Techniques *

Activity: Quiz: Performance Considerations

Purpose: To review performance considerations.

Activity: Worksession: Data Base Design - Parts Two & Three

Purpose: To design a data base for Parker Manufacturing.

Text Pages: TurboIMAGE Performance and Design Guidelines



Module 9-1

Performance and Design

Goal: To present design tips and performance considerations for producing a data base application system.

Objectives:

Upon completion of this module, you will be able to:

- design a data base.
- develop and present a block diagram and schema for a data base design.
- evaluate data base designs based on design guidelines.



Module 9-2

Activity 9-1 Chalk Talk: Overview of DBMS Implementation

Purpose: To review concepts in implementing a data base management system and to review what is needed to write a data base application program.

- Implementing a data base management system involves:

- Creating a data base application program involves:

A data management problem was defined in Activity 3-2. By the end of this module, you will:

1. design a data base for Parker Manufacturing.
2. develop and present a block diagram and schema for the design (one design per group).
3. evaluate the other proposed designs based on design guidelines presented in this module.

Module 9-2 Instructor Notes

□ Activity 9-1 Chalk Talk: Overview of DBMS Implementation

Time: 30 min.

Purpose: To review concepts in implementing a data base management system and to review what is needed to write a data base application program.

Directions:

Use slide 3.02 to review the steps in implementing a data base management system. Ask students to describe each step, and field any questions they may have. Then use slide 7.01 to review the steps in creating a data base application program. Ask students to describe the considerations in creating a data base application program.

■ Implementing a data base management system involves:

1. problem definition
2. data base design
3. data base definition
4. data base creation
5. create application programs
6. testing/debugging
- 7a. maintenance (restructuring)
- 7b. maintenance (backup and recovery)

■ Creating a data base application program involves:

Refer to slide 7.01 and review each of the steps involved.

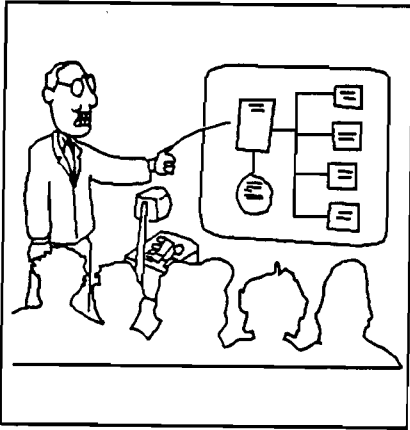
- TurboIMAGE procedures
- locking strategy to control concurrent access
- security

Review the WONDER REALTY data base design created in Module 4.

End Thursday.

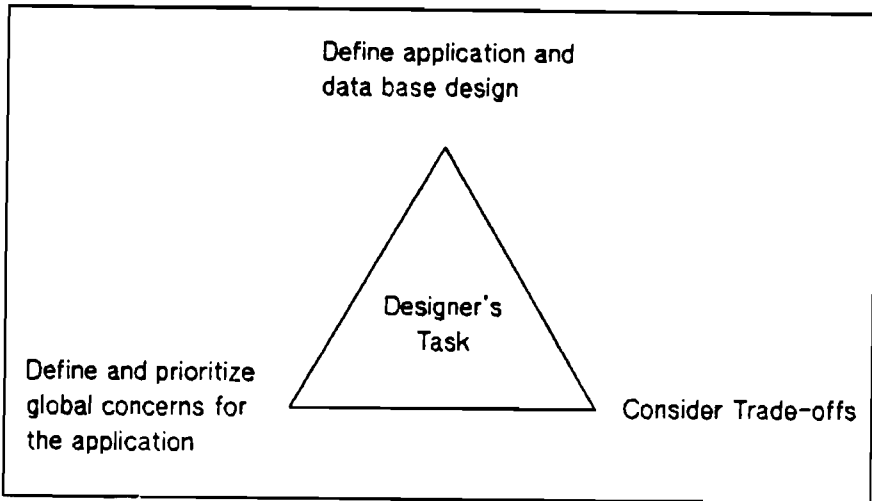
TurboIMAGE Design Considerations

How does a data base design come about?



- design team
- applications and data base designs
- considerations
- design approach
- common design decisions

Design Team



This slide greatly simplifies the design process. The three activities shown all take place at the same time.

Module 9-3 Instructor Notes: Slides 9.01/9.02

Performance and Design

Purpose: To introduce the topic of data base design.

Key Points:

- When designing a data base, a number of things should be taken into account, including flexibility, security, development time, etc.
- Performance is not always the first concern in data base design.

Purpose: To present the designer's tasks as a design team member.

Key Points:

- Defining the application can consist of endless interviews with end-users, specification writing and rewriting, flowcharting, data base diagraming, etc.
- More work done in designing the data base, according to an organization's priorities, will result in easier and quicker development.
- Well-designed applications usually are easier to maintain and more adaptable to changes.

Think about future directions, expected growth. Is it sensible to include items in your specs now rather than in six months or 12 months when you know you'll need them. eg Payroll or A/P and CENTEX.

Types of Designs

Application Design = "logical" design = WHAT

Data Base Design = "physical" design = HOW

"Logical" design: defines current and future requirements, functions to be performed, data elements, security, keys, and relationships.

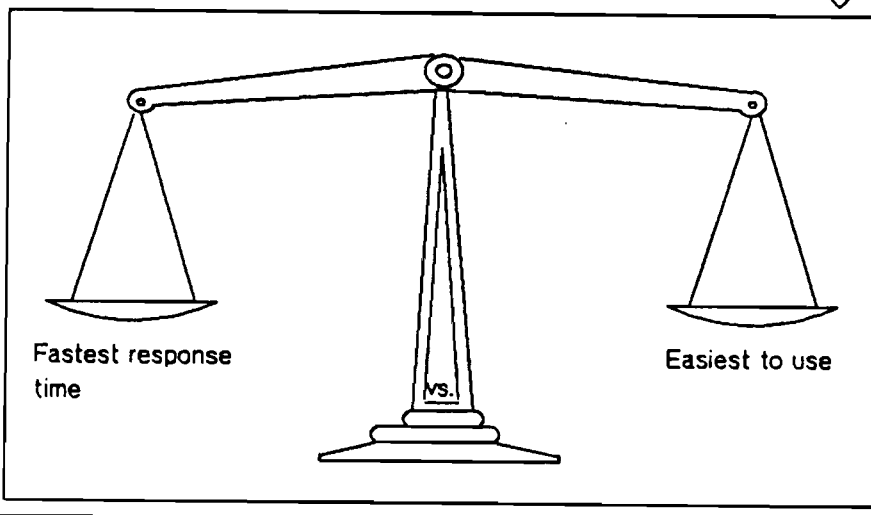
"Physical" design: provides structures and access methods to implement the application, performance, backup and recovery, and security.

Define and Prioritize Global Concerns



Consider trade-offs. Typical design concerns:

- flexibility and complexity of end-user needs
- security
- performance
- development time
- data independence
- disc storage
- maintenance time
- backup and recovery time



Module 9-4 Instructor Notes: Slides 9.03/9.04

Performance and Design

Purpose: To present the differences between the logical and physical data base design.

Key Points:

- Design of the actual data base is only one part of the whole design activity.
- Application Design is WHAT you are going to do.
- Data Base Design is HOW you are going to do it.
- For purposes of this discussion, tell class to assume that most of the application design is complete with the exception of, perhaps, selection of keys identifying all relationships between data elements. Discussion will center on these two activities as well as data base design itself.

Purpose: To present typical design concerns when designing a data base application.

Key Points:

- The designer should list an application's design concerns in order of priority. This makes it easier during the design process to make a decision about which strategy should be employed.
- Note that performance is only one of many design considerations.
- The concerns listed represent conflicts; their trade-offs must be considered. For example to design for performance requires that end users give up some of their requirements to get fast response time.
- The reverse may also be true: the most flexible, easiest to use, "do-everything" application, is often the slowest.

Module 9-5

Activity 9-2 Chalk Talk: Design Trade-offs

Purpose: To review data base design trade-offs.

Directions: List some examples of trade-offs in designing a data base.

Trade-off examples:

1. flexibility vs. performance

Module 9-5 Instructor Notes

□ Activity 9-2 Chalk Talk: Design Trade-offs

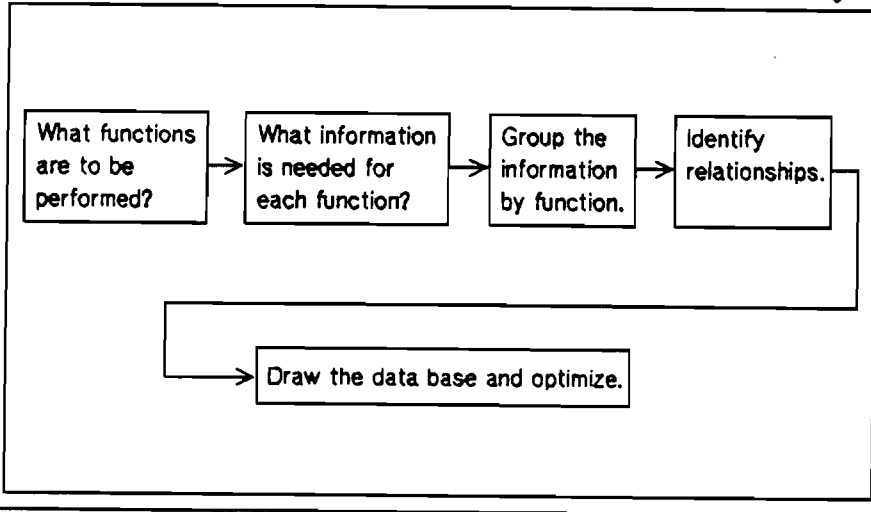
Time: 15 min.

Purpose: To review data base design trade-offs.

Directions: Give class time to write down examples of design trade-offs in their workbooks. Then, ask students for some of their examples; write these on a flip chart or blackboard.

- Trade-off examples:
 - flexibility vs. performance
 - development time vs. flexibility/complexity
 - development time vs. ease of use
 - data independence vs. development time
 - security vs. ease of use
 - performance vs. ease of use
 - performance vs. data independence
 - program maintenance vs. ease of use
- Ask students to mention other trade-offs.

Top-Down Design Approach



Module 9-8

Activity 9-3 Chalk Talk: Top-Down Design Approach

6. Draw the data base.

First pass:

Group items into detail data sets and identify the key items. Add the master data sets, specifying automatic or manual.

Notes:

Second pass:

Now, adjust your drawing to reflect set definitions. Optimize to save resources, resolve problems, and meet design concerns.

Notes:

Module 9-8 Instructor Notes

Activity 9-3 Chalk Talk: Top-Down Design Approach

Time: 45 min.

Purpose: To apply a top-down design approach to a given residential listing application..

Directions: Use the blackboard to go over this case study example. Relate this to the upcoming design lab. Refer to the actual REALTY data base (slide 2.11); emphasize that this is not the only possible design. Encourage students to discuss their own ideas on designing this data base. List several alternatives at each step of the design.

1. Define all the functions needed:

- a. identify each residence
- b. identify each residence by city
- c. identify each residence by price
- d. identify each residence by number of bedrooms
- e. identify each residence by number of bathrooms
- f. identify each residence by unique key
- g. identify each city

2. List the information needed to accomplish the above functions.

listing #
bedrooms
bathrooms
dining room
square feet
price
sold
current owner
street address
city name
zip
phone

(Continued on next page).

Module 9-6 Instructor Notes: Slide 9.05

Performance and Design

Purpose: To present a top-down design approach for designing a data base.

Key Points:

- The structured approach works best with complex projects. The following chalk talk shows an example of a case study design and the steps involved.
- This is only one approach that might be taken. In many cases, people use "seat of the pants" approach to designing their data bases.

Break

Module 9-7

Activity 9-3 Chalk Talk: Top-Down Design Approach

Purpose: To apply a top-down design approach to a given residential listing application.

Notes: Assume that you've already defined the current and future functions to be performed, and identified your design concerns.

Specifications: Provide residential listings by city, price, # bedrooms, # bathrooms, and a unique key.

Directions: Use the space to take notes as your instructor presents a case study example.

1. Define all the functions needed:
 - a. identify each residence
 - b. identify each residence by city
 - c. identify each residence by price
 - d. identify each residence by number of bedrooms
 - e. identify each residence by number of bathrooms
 - f. identify each residence by unique key
 - g. identify each city
2. List the information needed to accomplish the above functions.
3. Define items into functional entities.
4. Select keys. Be sure there is at least one key to identify a particular occurrence of an entity.
5. Define the relationships.

Notes:

(Continued on next page)

Module 9-8 Instructor Notes

Activity 9-3 Chalk Talk: Top-Down Design Approach

3. Define items into functional entities.

All these items are needed to describe a residence. In addition, we've decided to store information about each city for which we have listings. So, change CITY NAME in (2.) to a CITY ABBREVIATION. Add a new relationship of CITY ABBREVIATION with CITY NAME, POPULATION, DAYS OF WARM WEATHER, etc.

```
listing #
# bedrooms
# bathrooms
dining room
square feet
price
sold
current owner
street address
CITY ABBREVIATION
zip
phone
CITY ABBREVIATION
CITY NAME
POPULATION
DAYS OF WARM WEATHER
```

4. Select keys. Be sure there is at least one key to identify a particular occurrence of an entity.

Keys will be listing #, # bedrooms, # bathrooms, price, city abbreviation.

```
* listing #
* # bedrooms
* # bathrooms
dining room
square feet
* price
sold
current owner
street address
* city abbreviation
zip
phone
* city abbreviation
city name
population
days of warm weather
```

5. Define the relationships.

```
(detail data set)
* listing # <----- * listing # (automatic master)
* # bedrooms <----- * # bedrooms (automatic master)
* # bathrooms <----- * # bathrooms (automatic master)
dining room
square feet
* price <----- * price (automatic master)
sold
current owner
street address
* city abbr. <----- * city abbr. -----> * (detail data set)
zip (automatic master) * city name
phone population
days of warm
weather
```

(Continued on next page.)

Module 9-8 Instructor Notes

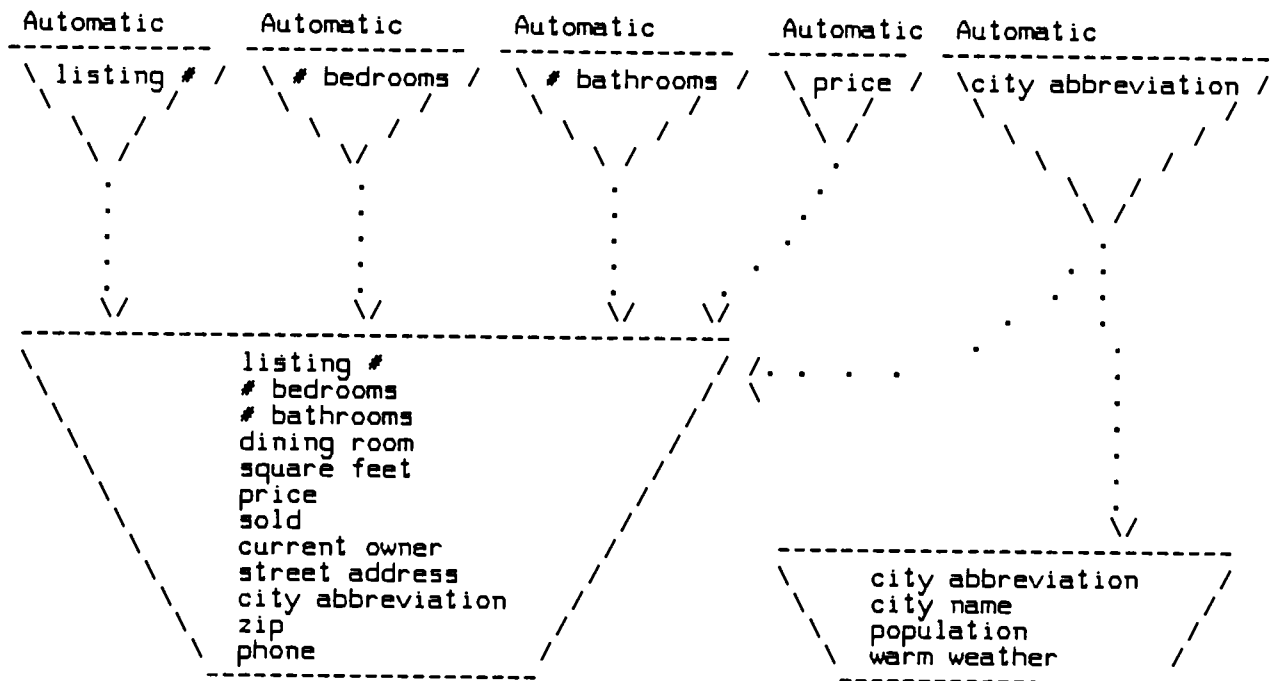
Activity 9-3 Chalk Talk: Top-Down Design Approach

6. Draw the data base.

Each entity or relationship becomes a detail data set and each key is put into a master.

First pass:

Group items into detail data sets and identify the key items. Add the master data sets, specifying automatic or manual.



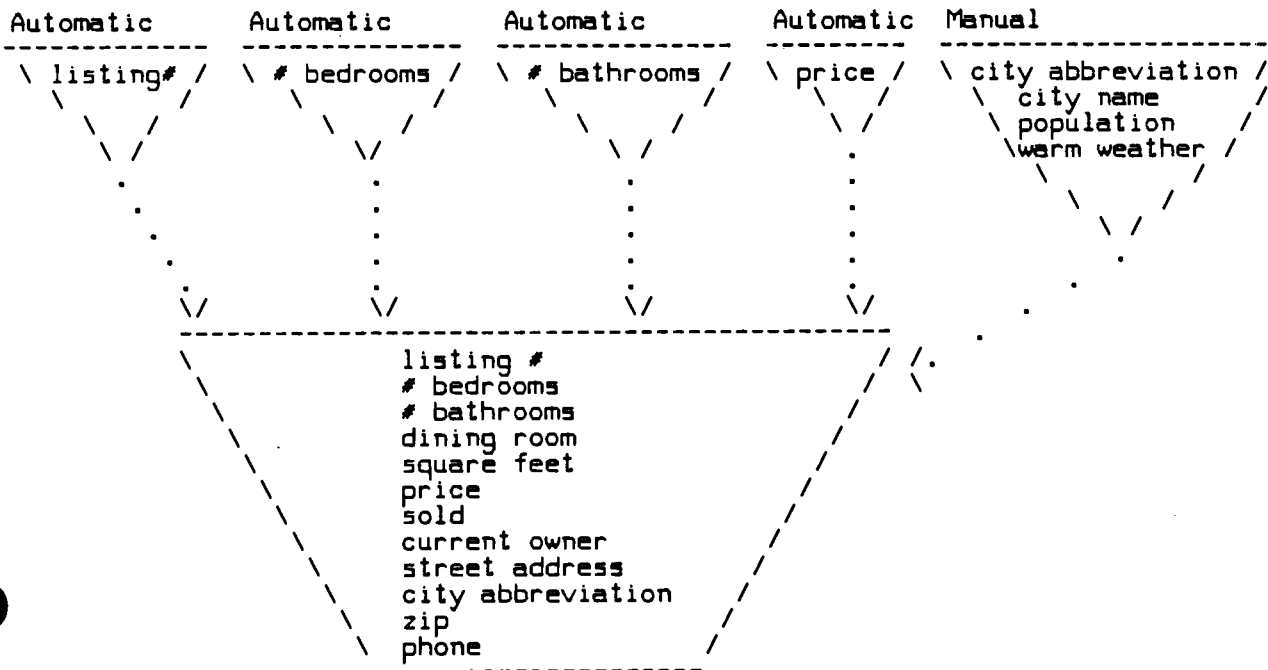
(Continued on next page.)

Module 9-8 Instructor Notes

□ Activity 9-3 Chalk Talk: Top-Down Design Approach

Second pass:

Now, adjust your drawing to reflect set definitions. Optimize to save resources, resolve problems, and meet design concerns.



We now have a simplified version of the case study. Refer to the drawing on the previous page and point out that a detail set can provide a link between two masters (this is much easier to see with the data base). Emphasize that this redrawing can happen several times. Each time, review the design concerns and function list to make sure they are satisfied. Also be sure that the relationships are allowed in TurboIMAGE.

Note: Point out to the class that designers may not always use the top-down approach to data base design. Some data base designs are accomplished using the "seat-of-the-pants" method. This means that the designer starts with the set of data items and builds the design from the bottom up as opposed to starting with the functions of the data base. Emphasize that data base design is more of an art than a science, and many factors affect the design method as well as the design itself.

Module 9-9

Activity 9-4 Chalk Talk: Common Design Decisions

Purpose: To present common design decisions that occur during data base design.

Notes: The following list represents some of the design decisions that need to be made.

Directions:

1. Use the space provided to take notes as your instructor highlights each topic.
2. Suggest other design decisions you think should be included.

Key selection

- Designers often overuse keys and paths.

Sets and paths

- Consider relationships.

Data sets: master vs. detail

- Detail sets usually store multiple occurrences of information.
- Master sets usually hold unique values (keys) for fast retrieval.

Master data sets: manual vs. automatic

- Manual masters are appropriate when you need to validate key values.
- Automatic masters are appropriate when key values are unpredictable or too numerous to enter manually.

(Continued on next page.)

Module 9-10

Activity 9-4 Chalk Talk: Common Design Decisions

Capacity

- Set detail data set capacity to current need plus reasonable growth.
- Set master data set capacity to current need, plus reasonable growth, plus 20% free space.

Single vs. multiple users

- Consider performance impact, open modes, locking levels.
- Must any function have exclusive access to the data base?

(Continued on next page.)

Module 9-11

Activity 9-4 Chalk Talk: Common Design Decisions

QUERY for production

- Consider performance impact, time and money to develop report in another language, etc.
- QUERY is more commonly used for production than it should be.
- QUERY is recommended for program debugging.

Single vs. multiple data bases

- Consider complexity of programming, memory usage, backup and recovery, etc.
- Performance can also be a consideration here as well. In many cases, two or more data bases perform better than one.

Interactive vs. batch

- Consider users' needs for current information, performance impact, etc.
- Identify which applications are primarily for reporting.

Back-up and recovery

- Consider time to back-up, performance impact of logging, etc.
- Consider how critical the data is to the operation of the business.

Module 9-11 Instructor Notes

□ Activity 9-4 Chalk Talk: Common Design Decisions

Time: 30 min.

Purpose: To present common design decisions that occur during data base design.

Directions: Use the list, which identifies design decisions that must be made, to generate discussion. Ask students for other design decisions they think are important, or ones they haven't been able to resolve.

Key selection

- Designers often overuse keys and paths. Emphasize the performance impact of overhead inherent in having many key fields in a detail set.
- Data items should be made into keys only for frequent searches or for those that must be fast when they are done.
- How do users look up data?
- What other keys are needed for fast or frequent access?

Sets and paths

- Consider relationships. The type of relationship between data items should determine which sets they belong to, and what relationships to build between sets.
- How is each data item related to every other data item?
- What does it take to describe an entity or to accomplish a function?

Data sets: master vs. detail

- Detail sets usually store multiple occurrences of information.
- Master sets usually hold unique values (keys) for fast retrieval.
- Master sets can contain the information that describes the key.
- Master sets can hold information about the search item. In this manner, they serve as a special kind of detail set.

Master data sets: manual vs. automatic

- Manual masters are appropriate when you need to validate key values. They are also good when you need to include more data about the key.
- Automatic masters are appropriate when key values are unpredictable or too numerous to enter manually.

(Continued on next page.)

Module 9-11 Instructor Notes

Activity 9-4 Chalk Talk: Common Design Decisions

Capacity

- Set detail data set capacity to current need plus reasonable growth. "Reasonable growth" depends on how often you wish to "grow" the data sets by going through the DBUNLOAD/DBLOAD process. If you can afford the time for this process every six months, then estimate the number of adds to each set for that period. If you have ADAGER, this is less of a concern.
- Set master data set capacity to current need, plus reasonable growth, plus 20% free space. Use prime number.
- There is no need to set the capacity at its eventual maximum if the growth rate is slow and you need to save on disc space and backup time. Remember that TurboIMAGE reserves all space required by the "CAPACITY:" statement in the schema.
- What is the current capacity required to store all records?
- How much and how fast will this figure grow?
- Consider disc space, hashing optimization, time to back up, etc.

Single vs. multiple users

- Consider performance impact, open modes, locking levels, etc.
- Functions that require exclusive use of the data base should be scheduled so they don't hold up other on-line users (i.e. evening, lunchtime).
- Locking designs should be carefully examined. Locking can be avoided if readers and updaters always work on different sets of the data base or on different entries.
- There can exist applications where it is not critical that the reader's data be protected. Be aware of pointer problems when doing chained reads.
- Adds, deletes, and updates can also be scheduled not to conflict with the more common read operations, so that the reporting programs can be written without locking.
- Must any function have exclusive access to the data base?
- Must readers be protected from changes during their transactions?
- Must updaters be protected from other updaters?

(Continued on next page.)

Module 9-11 Instructor Notes

□ Activity 9-4 Chalk Talk: Common Design Decisions

QUERY for production

- Consider performance impact, time and money to develop report in another language, etc.
- QUERY is more commonly used for production than it should be. QUERY can be a performance problem.
- Data types may not be useable with QUERY (i.e. inputting data which is packed).
- QUERY is recommended for program debugging and data base fix-ups.
- QUERY can be used for ad-hoc reports, but once these reports move from ad-hoc to monthly/weekly/daily functions they should be re-designed using a more efficient, less generalized tool.
- QUERY has no built-in validation function (beyond numeric and length checks) and no audit-trail (unless TurboIMAGE logging is used). Most programs will have these functions included.
- How often will this report be executed and against which other functions?
- Does the function require an audit trail or validation function?

Single vs. multiple data bases

- Consider complexity of programming, memory usage, backup and recovery, etc.
- Performance can also be a consideration here as well. In many cases, two or more data bases perform better than one. TurboIMAGE helps some.
- This decision is often a tough one. Major considerations are the ability to split data functionally without creating a gigantic redundancy and coordination problem; memory limitations; logging coordination problems; ability of programmers to code the more complex programs; etc.
- Are TurboIMAGE item or set limits reducing application flexibility?
- Are data items logically categorized by function with little overlap?

(Continued on next page.)

Module 9-11 Instructor Notes

Activity 9-4 Chalk Talk: Common Design Decisions

Interactive vs. batch

- Consider users' needs for current information, performance impact, etc.
- The tendency is to make everything an on-line transaction. They may not be appropriate and, in fact, in the case of adds and deletes to detail sets, may degrade performance significantly.
- If performance is a high-priority design concern, the designer should carefully scrutinize the data base design to optimize adds and deletes and should consider performing some activities in batch jobs in the D or E queues to preserve on-line response times. (This assumes default queue base and limit settings.)
- If users must have up-to-the-minute current data, batch is not an alternative.
- Is this primarily a reporting or update application?
- How current must information be?
- Must changes to key values be made on-line?

Backup and recovery

- Consider time to backup, performance impact of logging, recovery, etc.
- This design consideration is often "tacked on" at the end of the design process. It should be an integral part of the whole exercise.
- Not every data base needs to be logged. Not all data is critical to the daily functions of a business, and the performance price of logging should be considered for non-critical data bases.
- Some manual recovery can be as easy and as fast as TurboIMAGE recovery.
- How critical is this data to the operation of the business?
- How often must the data be available to users? (hours per day, days per week)
- How much data can we afford to lose in case of a catastrophic failure?
- How long can we be down for recovery?

Break



Performance and Design

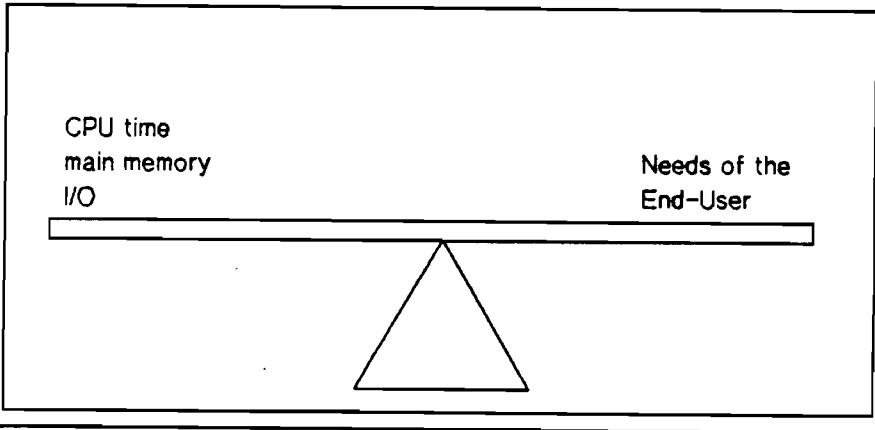
Notes

TurboIMAGE Performance



The goal of designing a TurboIMAGE data base is to minimize the use of CPU, Memory, and I/O resources while meeting the needs of the end-user.

What resources affect performance?



□ Performance and Design

Purpose: To present TurboIMAGE performance considerations and efficient programming techniques.

Key Points:

- It is not just the use we want to minimize, but contention and wait-time also. An example is a CPU not fully utilized, because processes are waiting for disc I/O to complete.
- Performance is only one consideration in designing a data base application. The application must still meet users' requirements (or some subset of them). It is important, however, to understand the performance implications imposed by a data base design.
- These are not the only resources to be considered. Other resources are disc and system tables.

Module 9-13

□ Activity 9-5 Chalk Talk: Performance

Purpose: To present TurboIMAGE impact on resources.

Notes: The following list represents some of the prime resource considerations when measuring performance. This is intended to serve as a guide for designers who are trying to increase performance. This will expose designers to how much their designs and programs cost in terms of system resources. Whether the performance being measured is on-line response time or batch throughput, these considerations should be reviewed for each application.

Directions: As your instructor goes over each of these considerations, use up or down arrows (or NA if not applicable) under the appropriate column to indicate how resources are affected.

	CPU	Memory	I/O
Items, sets, and paths	/\	/\	/\
- more items	_____	_____	_____
- more sets	_____	_____	_____
- more paths	_____	_____	_____
Data base application types			
- read and update only	_____	_____	_____
- add and delete	_____	_____	_____
Paths (for adds and deletes to the detail sets only)			
- many masters related to one detail set	_____	_____	_____
- many details related to one master set	_____	_____	_____
Synonym chains			
- byte-type keys (X, U, Z, P)	_____	_____	_____
- capacity = prime #	_____	_____	_____
- 20% - 30% free space	_____	_____	_____
Detail chains (for chained reads)			
- many adds and deletes	_____	_____	_____
- select primary path = most accessed chain	_____	_____	_____

(Continued on next page.)

Activity 9-5 Chalk Talk: Performance (cont'd)

	CPU	Memory	I/O
Sorted chains (for adds)			
- many sorted chains	_____	_____	_____
- short chains	_____	_____	_____
- pre-sorted transactions	_____	_____	_____
- sort field at EOR	_____	_____	_____
Locking			
- locking employed	_____	_____	_____
Blocksize			
- SERIAL or CHAINED access	_____	_____	_____
- DIRECTED or CALCULATED access	_____	_____	_____
Buffers			
- default buffers with frequent DBOPENS and DBCLOSEs	_____	_____	_____
- stable buffer allocation for range of users	_____	_____	_____
File placement			
- files placed on same disc	_____	_____	_____
- files placed on many discs	_____	_____	_____
Multiple data bases			
- more data bases with increased complexity	_____	_____	_____
Backup and recovery			
- TurboIMAGE logging	_____	_____	_____

Module 9-14 Instructor Notes

□ Activity 9-5 Chalk Talk: Performance

Purpose: To present TurboIMAGE impact on resources.

Time: 30 min.

Directions:

1. Go over each of these considerations with students. Review in as much detail as necessary to insure the students are comfortable with the information.

Items, sets, and paths

The more items, sets, and paths included in data bases, the more complex they become. They tend to use more CPU and memory (larger control blocks and stacks) while generating more requests for I/O than simpler designs.

	CPU	Memory	I/O
- more items	∧	∧	∧
- more sets	∧	∧	∧
- more paths	∧	∧	∧

Data base application types

TurboIMAGE is optimized (at least in I/O terms) for inquiry and update; but not for adds and deletes. DBFINDs, DBGETs, & DBUPDATES are less costly than DBPUTs & DBDELETES. If performance is important, the more you do to lessen the I/O costs of adds and deletes, the better.

There are many solutions to this potential problem. Often, if on-line response time for inquiries must be fast, add and delete transactions can be stored in a file or set to be processed when there is little or no on-line activity. The trade-off is that users won't have immediate access to this information. Another alternative for deletions is to set a delete flag by using DBUPDATE calls and do the DBDELETES later.

	CPU	Memory	I/O
- read and update only	∨		∨
- add and delete	∧		∧

(Continued on next page.)

Module 9-14 Instructor Notes

□ Activity 9-5 Chalk Talk: Performance (cont'd)

Paths (for adds and deletes to the detail sets only)

Many masters related to one detail set is much more expensive than the reverse case when adding and deleting detail records. However, if the key fields provided by the masters are needed for fast inquiry, the designer may be willing to pay the price. Exception: If detail records are loaded infrequently and are very stable most of the time (few or no DBPUTs & DBDELETes on-line), the many masters design would be fine. It would provide fast access paths for inquiries. The many masters design would take up more disc space for chain heads and pointers than the many details design.

	CPU	Memory	I/O
- many masters related to one detail set			∧
- many details related to one master set			∨

Synonym chains

All the work that TurboIMAGE does to maintain synonym chains, migrate secondaries, and look up master records reside on these chains. Whatever can be done to minimize synonyms will improve performance. Reducing the number of synonyms will result in shorter synonym chains. If an I, J, K, or R key is used, be sure to distribute the key values by pre-hashing if necessary. The hashing algorithm for integer keys is very poor at randomizing values that are not inherently random (i.e. 101, 102, 201, 202, etc.). Exception: Suppose key values for 1,000 master records are integers from 1 to 1,000. Set the capacity to 1,000 and there will be no synonyms. The relative record number is equal to the remainder obtained when the integer value is divided by the capacity of the set. Refer to the *TurboIMAGE Reference Manual*, Section 10: Primary Address Calculation, for further details.

	CPU	Memory	I/O
- byte-type keys (X, U, Z, P)			∨
- capacity = prime #			∨
- 20% - 30% free space			∨

Detail chains (for chained reads)

Many adds and deletes give increased disorganization, which can be corrected by selecting the primary path carefully. Designers should use the path on which they want the fastest chain reads. Regular DBUNLOAD/DBLOAD will place members of the chain together in physical blocks. As records are deleted and added this organization will gradually be destroyed.

	CPU	Memory	I/O
- many adds and deletes			∧
- select primary path = most accessed chain			∨

(Continued on next page.)

Module 9-14 Instructor Notes

□ Activity 9-5 Chalk Talk: Performance (cont'd)

Sorted chains (for adds)

If the sorted chains are very stable with only an occasional DBPUT, the price of adding the sort field can be smaller than the advantage of guaranteed order in the chain. If the order of adds is naturally in sorted order, the sort field designation need not be required (i.e. sort by the current date or clock time). If the chains or sets are to be read in batch mode, it is better to read and then sort with MPE sort facilities.

	CPU	Memory	I/O
- many sorted chains			∧
- short chains			∨
- pre-sorted transactions			∨
- sort field at EOR			∨

Locking

With locking employed, the data integrity goes up but the CPU use also goes up and the data base availability goes down. Data integrity, if it is important, will cost in terms of CPU and availability to the application program. There is no need to add locking if the application owns the data base exclusively or if it is opened in read mode. It is only when an updater logs on that results become unpredictable and locking may be needed.

Locking at the lowest possible locking level increases data base availability, but it also increases programming complexity. Lower level locking allows more concurrent use of data bases and perceived performance is improved. They can be more difficult to program, but are absolutely essential in a multi-user environment. Entry-level locking used with a common lock item will increase data base availability. This is also true for entry-level locking with a pre-sorted descriptor. The lock that best protects the user's transaction must be employed. Locks should not be placed around user dialogue.

	CPU	Memory	I/O
- locking employed			∧

Blocksize

SERIAL - if there are few inactive records, a larger block size will improve performance. CHAINED - if this is the primary path and you've maintained organization with DBUNLOADS/DBLOADS, larger block sizes will create less physical I/Os. DIRECTED - smaller is better. CALCULATED - smaller is better, except when there are many synonyms; then a larger blocksize would bring in more synonym chain members with one physical I/O.

	CPU	Memory	I/O
- SERIAL or CHAINED access		∧	∨
- DIRECTED or CALCULATED access		∨	∨

(Continued on next page.)

Module 9-14 Instructor Notes

Activity 9-5 Chalk Talk: Performance (cont'd)

Buffers

Using the default buffers with frequent DBOPENS & DBCLOSEs causes a lot of swap-out, swap-in to expand and contract the Control Blocks. Using a stable buffer allocation for a range of users will reduce the Control Block expansions and contractions.

	CPU	Memory	I/O
- default buffers with frequent DBOPENS and DBCLOSEs			/\
- stable buffer allocation for range of users		/\	\

File placement

Files placed on the same disc increases head contention. Files spread across many discs reduces head contention. Use :RESTORE with the :DEV= parameter to direct files to a particular disc; can also use DBUTIL,MOVE or specify disc class in schema.

	CPU	Memory	I/O
- files placed on same disc			/\
- files placed on many discs			\

Multiple data bases

Splitting the data for an application into multiple data bases can have both negative and positive effects. More data bases increase complexity, CPU, memory, and I/O. However, more data bases also increase flexibility and decreases contention and recovery time. At recovery time, you can restore one data base making it available to users while the other data bases are being recovered. One large data base would be unavailable longer.

Disadvantages are: increased use of all resources, increased complexity of programming, more redundant items, more disc space, logging coordination problems, and perhaps, deadlock trouble with MR.

Data bases should be split functionally with little overlap or redundancy. Locking schemes should be designed to minimize wait time and eliminate deadlocks.

	CPU	Memory	I/O
- more data bases with increased complexity	/\	/\	/\

Backup and recovery

Frequent backups make the data base often unavailable, but recovery is easier and faster and the chance of losing data is reduced. TurboIMAGE logging increases recovery time, CPU, memory, and I/O but it also reduces the chance of losing data, and data base availability.

The decision to implement logging should take into account the effect of lost data, how long it would take to recover manually, and the performance impact of the extra I/O.

	CPU	Memory	I/O
- TurboIMAGE logging	/\	/\	/\

Efficient Programming Techniques



DBOPEN - Minimize the number of DBOPENs to reduce use of all resources.

DBCLOSE - Use Mode 3 to reset pointers to 0 and avoid overhead of other modes.

DBGET - Use DBINFO procedure calls to retrieve data item and data set numbers.

DBUPDATE - Use partial LIST to reduce security check.

DBPUT - DBPUTs and DBDELETEs to master sets can cause migrating secondaries.

□ Performance and Design

Purpose: To present efficient programming techniques with TurboIMAGE procedures.

Key Points:

- **DBOPEN** - Use open mode that protects well, but allows as many concurrent users as possible.
- **DBGET** - **LIST = @** reduces item-by-item checking. **LIST = *** on second and later calls avoids list processing and security check. Pass numbers instead of names to avoid Item Table searches. Use partial LIST to retrieve a few items from a large set to avoid security check.
- **DBUPDATE** - Use **LIST = @** only if changing most non-key items or if changes are unpredictable. Ensure key is unchanged. Ensure write access to all items that may be changed. Write access at set level is faster.
- **DBPUT** - **LIST = @** reduces item-by-item checking. **LIST = *** on second and later calls avoids list processing and security check. Ensure all keys are present.

Module 9-16

Activity 9-6 Quiz: Performance Considerations

Purpose: To review performance considerations.

Directions: Answer each question by circling true or false. Your instructor will review the answers at the end of the quiz.

1. A large blocksize is the best choice for a data set that is usually accessed in serial mode (DBGET mode 2 or 3). T F
2. The entire data base should be kept on one disc drive in order to reduce head contention. T F
3. The capacity for any data set should only be great enough to hold the current records so that disc space is not wasted. T F
4. The number of synonyms in a master set can usually be reduced by choosing a prime number capacity for that set. T F
5. When many add (DBPUT) and delete (DBDELETE) transactions are required for a detail data set, performance can be improved by increasing the number of paths it has to master sets. T F
6. In general, more items, sets, and paths in your data base means more system resources are required to run your application. T F
7. When you design a data base application to perform well on the HP 3000, your only goal should be to satisfy every request of the end-users. T F
8. One advantage of data base logging is that it does not use any system resources. T F
9. A "*" in the LIST parameter of a TurboIMAGE procedure call can be used to minimize overhead. T F
10. An application using data entry locking always will have better performance than the same application using data set locking. T F

Module 9-16 Instructor Notes

□ Activity 9-6 Quiz: Performance Considerations

Time: 10 min.

Purpose: To review performance considerations.

Directions: After students have completed this activity, review the answers with them.

ANSWERS:

1. True.
2. False.
3. False.
4. True.
5. False.
6. True.
7. False.
8. False.
9. True.
10. False.

Platby

Module 9-17

Activity 9-7 Worksession: Data Base Design

Purpose: To design a data base for Parker Manufacturing.

Notes: Use the TurboIMAGE Performance And Design Guidelines at the end of this module as a guide for designing your data base and for evaluating other designs.

Directions:

Part 2:

1. Work in the same group from Part 1 of this activity. Use the concepts presented in this module to produce a data base design for Parker Manufacturing.
2. Draw a block diagram and schema on the blank transparencies provided by your instructor.

Part 3:

3. Select one member of your team to present and discuss your design to the class.
4. As the other teams present their designs, offer constructive suggestions.
5. Your instructor will provide you with copies of all designs presented in class.

Activity 3-2 for outlines

Module 9-17 Instructor Notes

□ Activity 9-7 Worksession: Data Base Design

Purpose: To design a data base for Parker Manufacturing.

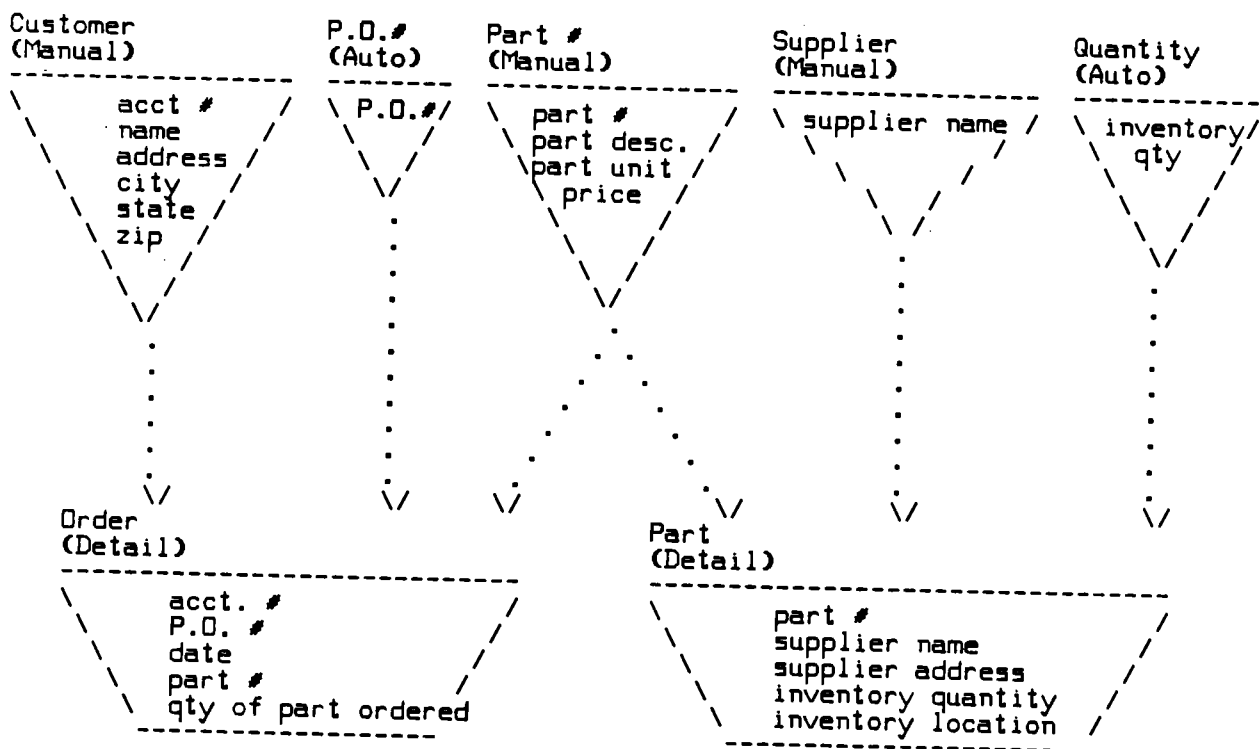
Time: 2 hours

Notes: Students will be using the TurboIMAGE Performance And Design Guidelines at the end of this module to guide their data base design and evaluation of other designs.

Directions:

1. Hand out 2 transparencies and one pen to each team.
2. Circulate around the room to offer assistance to the teams while they are working.
3. When all teams are finished supervise the presentations. Caution students to offer constructive criticism to the designs.
4. Present your own design solution (or the sample solution listed below) to the class.
5. Make copies of all designs and hand out to students.

One possible solution:



TurboIMAGE Performance And Design Guidelines

I. Data Base Creation

- A. Run DBSCHEMA to create the root file.
- B. Run DBUTIL,CREATE to create the data sets.
- C. Run DBUTIL and set the buffer specifications according to the applications requirements. (>SET BUFFSPECS=)
- D. Decide which disc each data set will be placed on, and then place them accordingly.
 - For best performance, separate masters and related details, and spread the heavily accessed data sets over the available devices (STORE then RESTORE with DEV= parameter, or use DBUTIL,MOVE with TurboIMAGE).
- E. If using TurboIMAGE transaction logging to disc, do not put the transaction log file on the same discs as the data base.

II. Security

- A. A single function's data should be in as few data sets as possible.
- B. Reviewing the security reminds us of what the individual functions will be doing.
- C. There should be a password for every user type or function expected on the data base.
- D. Batch processing should also have separate passwords.
- E. For each password; give the minimum access necessary (for security reasons) at the highest level possible (for performance reasons).
- F. Use item level security only when necessary.

(Continued on next page.)

□ Performance and Design**TurboIMAGE Performance and Design Guidelines (cont'd)****III. Capacities**

- A. TurboIMAGE allocates all disc space when you create the data base.
- B. Changing the capacity normally requires a DBUNLOAD/DBLOAD which can't be done at a moment's notice.
- C. Capacity of each data set should be specified as realistic as possible so that unused disc space is not wasted.
- D. Master data sets should be specified to be at least 20% greater than the needed capacity to reduce synonym chains (this is not required for detail data sets). Do not use capacities of 2**n.
- E. In detail data sets, TurboIMAGE will take the capacity specified and change it slightly to make most efficient use of disc space, based on the blocking factor.
- F. For detail data sets there are no performance considerations for capacity; just pick a capacity large enough so that you don't force an unscheduled DBUNLOAD/DBLOAD.
- G. Check for potential problems when you are near to a limit. Current limits are:
 - 1. data item names per data base : 1024 (255 with IMAGE/3000)
 - 2. data item names per data set : 255 (127 with IMAGE/3000)
 - 3. data sets per data base : 199 (99 with IMAGE/3000)
 - 4. detail data sets per master : 16
 - 5. search items (keys) per detail : 16
 - 6. maximum entry size : 4096 bytes
 - 7. maximum # of entries per data set: 2 billion (8 million with IMAGE/3000)
 - 8. maximum # of entries per chain : 2 billion (65k with IMAGE/3000)

IV. Schema

- A. Use set level write access.
- B. Use dummy set/items to allow for future changes.
- C. Automatic masters are easier to use but have less control.
- D. Use stand-alone masters for look-up tables.
- E. Use stand-alone details to benefit from TurboIMAGE security, QUERY/3000, procedures, and logging.
- F. Consider blocksize.
- G. Put items most used first in schema (faster conversion of names to numbers).

(Continued on next page.)

TurboIMAGE Performance and Design Guidelines (cont'd)

V. BUFFSPECS

- A. Set to a constant for a wide range of users to avoid control block expansions and contractions.
- B. Use more for updating.
- C. Use output deferred in stand-alone environment.

VI. Block Size (DBSCHEMA)

- A. The block size can affect efficient disc space utilization, increase/decrease number of disc accesses and system/program execution time.
- B. If record is large or if using sequential or chained reads, use a larger block size.
 - 1. fewer I/Os for sequential or chained reads
 - 2. takes less time to chase/place synonyms
- C. Use the \$CONTROL BLOCKMAX= command to vary the maximum allowable block size for all data sets.
- D. To set a different maximum block size for each data set, put a \$CONTROL BLOCKMAX statement before each set definition in the schema.
- E. In general, the block size selected by TurboIMAGE is a good choice (default = 512 words, minimum = 128 words, maximum = 2048 words).
- F. If using serial/chained access, select large block sizes to reduce disc accesses by placing more records in a block per access.
- G. If most activity on the data base is random, select smaller block sizes and more buffers (BUFFSPECS).
- H. TurboIMAGE uses the largest block size in the data base to determine what the buffer size will be. Having one data set with a very large block size will force larger buffers and waste main memory.

VII. Synonyms

- A. Use ASCII-type search keys or integers with random values.
- B. Keep master data set at most 75% full to get a better distribution over the data set and therefore reduce the number of synonyms.
- C. Making master data set capacity equal to a prime number will generally yield fewer synonyms.
- D. Two pass loading of masters is recommended.

(Continued on next page.)

□ Performance and Design**TurboIMAGE Performance and Design Guidelines (cont'd)****VIII. Key Selection**

- A. For best performance, master data sets should have as few synonym chains as possible. Factors affecting this are the data type of the key, the key values, and the capacities selected for the master data sets.
- B. The data type you select for each key determines what type of algorithm TurboIMAGE will use to calculate primary addresses.
- C. For master data sets with keys of data type U, X, Z, or P, TurboIMAGE uses a hashing algorithm that approximates a uniform distribution of primary addresses. To achieve the minimum number of synonyms the capacity of the data set should be a prime number, and should be large enough to be at most 75% full.
- D. For master data sets with keys of data type I, J, K, or R, TurboIMAGE uses modulo arithmetic, based on the data set capacity, to calculate the primary address for the key. By selecting the right capacity, you can prevent groups of key values from generating the same primary addresses.

IX. Paths

- A. DBFIND changes path (current record in detail = 0); use multiple DBOPENS to maintain multiple paths to the same data set.
- B. Number of paths specified in a data base should be kept to a minimum for optimum performance.
- C. The cost of a path is mostly in pointer maintenance and overhead (DBPUT/DBDELETE).
- D. Ask these questions:
 - 1. How much will it be used?
 - 2. Will it mean faster access?
 - 3. How often are adds and deletes done?
 - 4. Will the key value ever change?
 - 5. Will the longest chain be less than 10% of the total number of entries in the detail data set?
- E. The general rule is the simpler the better. For good performance, you want just as many paths as you really need, no more. You should have a good reason for every path in the data base.



(Continued on next page.)

Performance and Design**TurboIMAGE Performance and Design Guidelines (cont'd)****X. Primary Path**

- A. Select the most frequently accessed path in a detail data set as a primary path.
- B. Select a path that is accessed frequently, by interactive users, and whose average chain length is fairly long.
- C. Always choose a primary path for each detail data set. By default, TurboIMAGE selects the first unsorted path specified in the schema, which may not be the most active path. If all paths are sorted, TurboIMAGE selects the first sorted path.
- D. Doing DBUNLOAD/DBLOADs periodically can significantly improve performance of long chains.

XI. Sorted Chains

- A. Limit the use of sorted chains to paths with relatively short chains to reduce the time required to add/delete entries.
- B. Extended sort fields are useful for multi-level sorts, but should not use DBUPDATE on these fields.
- C. Pre-sort before adding.
- D. Updating a sort field requires a DBDELETE/DBPUT.
- E. When adding a sorted chain, TurboIMAGE begins at the end of that chain and follows the backward pointer until it finds the logical insertion point and then modifies the pointers on either side of the new entry.
- F. To improve the load time of a detail data set with a sorted chain, use the sort item to pre-sort the input data before loading the data base. The load will be as efficient as adding to a detail data set with no sorted chain.
- G. Sorted chains were designed for the processing convenience of interactive programs.
- H. Use the external sorting capabilities of SORT/3000, or through COBOL, for reports.
- I. The difference between a sorted and an unsorted chain is the method used to do the DBPUT. The cost of a DBPUT to an unsorted chain is fairly constant, but for a sorted chain it depends on where the entry will be put.
- J. DBUNLOAD copies no pointer information. Chains of detail data sets are created by DBLOAD when the file is reloaded. Therefore, the order of entries in the chains is often changed.
- K. Good reasons to use sorted chains:
 - 1. When you must maintain an order.
 - 2. When you seldom (or never) add.
 - 3. When the sort values are always ascending.
 - 4. When the chain is short.
 - 5. When an interactive process requires it.

(Continued on next page.)

□ Performance and Design**TurboIMAGE Performance and Design Guidelines (cont'd)****XII. List Parameter In DBGET**

- A. Partial
 1. names hashed to get item numbers
 2. security checks done
 3. data formatted in control block trailer and then passed to stack
 4. use if few items needed from entry
 5. use small number of items in a large record to bypass checking security on all items
- B. "@"
 1. no hashing of names (numbers found in DBCB data set control block)
 2. security checks done
 3. data transferred directly to stack
 4. bypasses list processing overhead
- C. "*"
 1. no look-up of numbers
 2. no security checks on reads
 3. if *=@ data transferred directly to stack
 4. bypasses security checking/list processing overhead
- D. "0" (null or blank list)
 1. only points to record; no data transferred
 2. checks bit map to ensure record exists
- E. Use item number or set number to bypass "item table" search for valid item name.
- F. Put frequently accessed items at beginning of ITEM part in schema because of the serial search through the table.
- G. Mode 2/3 (serial/backward serial reads).
 - large data set capacity with a large amount of unused space and/or large blocks contribute to processing overhead
- H. Mode 5/6 (chained/backward chained reads).
 1. use large blocks for chains to decrease disc accesses
 2. use periodic reloads to increase performance
- I. Mode 7/8 (calculated/primary calculated reads).
 - uses synonym chains

(Continued on next page.)

TurboIMAGE Performance and Design Guidelines (cont'd)

XIII. DBOPEN

- A. Use the least access mode capability that will accomplish the task.
- B. Keep number of buffers consistent over a range of users.
- C. Set buffers to maximum for a data base reload and consider output deferred mode of operation if using mode 3.

XIV. DBCLOSE

- A. Use mode 2 to close a data set if short of stack space and intend not to re-open.
- B. Mode 3 resets pointers and does not close data set.

XV. DBUPDATE

- A. List
 - 1. never bypasses list processing; must insure search items are not changed and check write access on all changed items
 - 2. use only what you want to update
- B. DBGET - DBUPDATE sequence - establish list parameter with DBGET and use "*" in DBUPDATE

XVI. DBPUT

- A. DBPUT to a detail with an automatic master is more time consuming than to a manual master.
- B. DBPUT to a manual master is low on overhead if no synonyms.
- C. DBPUT to a detail with an automatic master is high on overhead due to chain maintenance in master.
- D. If an automatic master has 0 or 1 record in the detail, there will be a lot of adds and deletes done to the master.

XVII. DBDELETE

- A. Delete entries with a flag field or run a batch job at night.
- B. Has same overhead as DBPUT.
- C. Has low overhead on masters with no synonyms.

(Continued on next page.)

TurboIMAGE Performance and Design Guidelines (cont'd)

XVIII. DBUNLOAD/DBLOAD

- A. Periodically use DBUNLOAD/DBLOAD to reduce data base fragmentation.
- B. DBUNLOAD/DBLOAD ensures that chains are placed in the same block, as data can be fragmented through adds/deletes on highly volatile data sets and response time can be increased through reduction of disc accesses.
- C. Use DBUNLOAD,CHAINED to reorganize chains and improve performance (fewer I/Os).
- D. DBLOAD runs faster if buffspecs set high (50(1/2)).

XIX. QUERY/3000

- A. Debugging aid.
- B. Not for heavy updating (no data edits).
- C. QSLIST for restructuring.
 - 1. delete sets
 - 2. change item data types
 - 3. change item length

(Continued on next page.)

Performance and Design**TurboIMAGE Performance and Design Guidelines (cont'd)****XX. Shared Access/Locking**

- A. Lock around a transaction.
- B. Keep lock as short as possible.
- C. Set locks OK for ≤ 8 calls.
- D. MR - lock in pre-defined sequence and unlock in reverse order.
- E. Lock at same level; use same lock item.
- F. Use MODE1 DBGET (re-read) in a lock after operator action.
- G. For serial or directed reads - lock set/base;
for chained or calculated reads - lock entries;
for adds/deletes in master - must lock set/base.
- H. When updating lock item, must lock on both values.
- I. No need to lock in modes 3,7,8.
- J. Program complexity increases with lower level locking.
- K. With many users accessing a given data base, locking at the data entry level gives the best performance because it increases concurrency.
- L. All programs concurrently accessing the data base should generally lock at the same level.
- M. For lengthy transactions (more than 8 TurboIMAGE calls) a user should lock at the data entry level.
- N. If user dialog takes place while locking is in effect data entry level locking should be used.
- O. For chained DBGET calls, lock all data entries in the chain using a single lock descriptor.
- P. For serial DBGET calls, lock the data set.
- Q. For updates to a data entry (DBUPDATE), lock the data entry before calling DBGET to read the data entry and unlock after the update.
- R. For directed reads (DBGET), lock the data set before determining which data entry is needed.
- S. When adding an entry to a detail data set, use any lock that covers the data entry, but it is preferable to use the data entry that is the "lock item" for the data set.
- T. When adding to or deleting from a master data set (DBPUT and DBDELETE), lock the data set or data base.
- U. Multiple locks require MR capability.
- V. With multiple locks use conditional locking on second lock to avoid possible process deadlocks.

(Continued on next page.)

□ Performance and Design**TurboIMAGE Performance and Design Guidelines (cont'd)**

- W. With multiple locks, lock at the same level.
- X. With multiple locks, lock in a predefined sequence.

XXI. Programming Tips

- A. Keep code segments to around 4K words.
- B. Keep code localized.
- C. Use stacks efficiently (re-use data areas).
- D. Use item/set numbers instead of names.
- E. Close data set when not needed.
- F. Use DBGET mode 1, if serially reading a master and deleting all entries.
- G. Use DBCLOSE mode 3 before serial reads.
- H. Do not ignore errors; always check the TurboIMAGE condition codes.
- I. Consider data types and language used.

XXII. Disc Drive Contention

- A. Use the :RESTORE command with the DEV= option against a STORE tape to spread individual data sets across disc drives thus reducing disc contention. In TurboIMAGE use DBUTIL,MOVE. Once done MPE knows the disc where the data set was stored and will restore it to that same disc drive. Can also specify disc in schema.

XXIII. Transactions

- A. Get a feeling for the performance of the data base application by defining the different logical transactions that will be used, and exercise them.
- B. Walk through each type of transaction and see how it works with the data base design.
- C. Watch out for transactions that do a large number of DBPUTs and DBDELETES.
- D. To be most efficient, locking and recovery strategies should be completed, and the necessary intrinsic calls included in the transactions.
- E. If there are any areas of concern, consider testing the performance of the transactions against the data base design.
 - 1. DBDRIVER
 - 2. TurboIMAGE Profiler
 - 3. Can also use logging

(Continued on next page.)

TurboIMAGE Performance and Design Guidelines (cont'd)

XXIV. Data Base Maintenance

- A. Analyze the time required to maintain the data base (unload/load).
- B. Procedures should be developed to detect potential performance problems (e.g. long synonym chains in master data sets).
 - DICTDBA - Dictionary/3000 data base audit utility that reports on the usage statistics and checks the internal linkages of a TurboIMAGE data base.
- C. Is the data still correct?
- D. Is the structure still correct?
- E. Structural changes to data bases are a natural part of the maintenance of a data base application.
 - 1. DBUNLOAD/DBLOAD
 - 2. custom programs
 - 3. DICTDBU/DICTDBL
 - 4. ADAGER/3000

Module 9-29 Instructor Notes

Preparation Material: Slides 9.01/9.02

Teaching Tips:

This lesson will provide students with information on how to design TurboIMAGE data base applications. Poll the students before you start this lesson as to their interests, and tailor the presentation to their capabilities and enthusiasm; that is, are the students programmers, designers, or consultants (i.e. they will only be reviewing or supporting data bases already in place).

Teaching Tips:

- Ask these questions to generate some discussions:
- Who is on a team?
- What decisions does the team make?
- How often does the team need to communicate?
- What are the current practices in your shop?

Module 9-30 Instructor Notes

Preparation Material: Slides 9.03/9.04

Teaching Tips:

Reference the specifications from Activity 3.2. These represent components of the application design phase.

Teaching Tips:

Review each concern with students and ask which ones apply to their own applications and which are high priority concerns. Also ask what other concerns they can think of. Most of the concerns listed are self-explanatory. For clarification:

- data independence - refers to the need to avoid changing programs when the physical characteristics of the data base are changed.
- maintenance time - refers to ease of restructuring data bases and ease of debugging and documenting systems.
- backup and recovery time - usually deals with the amount of time the data base would be unavailable due to these activities and how easy it is to recover lost data.

Reference the relational data base model (Module 1) as an example of flexibility gained at the expense of performance.

Module 9-31 Instructor Notes

□ Preparation Material: Slide 9.07

Preparation:

Highlight the points on the slide; then refer students to the TurboIMAGE Performance and Design Guidelines at the end of this module.

Emphasize that no unnecessary DBOPENS or DBCLOSEs should be done. A common mistake occurs in the application that begins by displaying a menu from which the user chooses a transaction to be done. All transactions access the same data base, but the program does a DBOPEN at the beginning of each transaction type and a DBCLOSE before returning to the menu. This generates a lot of needless work on MPE's part and should be avoided. Open the data base once and only close it when the user exits the application program.

With DBGET, item and set numbers are faster than names as they avoid the table searches. The fastest overall method is the "@" on the first read followed by the "*" for each successive read. An item name or number list followed by the "*" can be even faster if only read access is given at the set level.

Point out that the "@" list reduces data independence of programs, as well as the self-documenting nature of listing all the item names. If the structure of the data entry changes, all programs that read it with an "@" list must be changed.

With DBUPDATE, the partial list is fastest and should be used unless all non-key (or non-sort) fields are updated. DBUPDATE will always do the security check for each changed data item (unless write access is given at the set level). This makes the "@" list expensive here.



Module 10 Instructor Notes

Data Base Tools

Overview of Module 10

* - indicates preparation material and/or teaching tips are included at the end of the module.

Note: Times for lessons are approximate and include activity times.

Lesson 1. Remote Data Base Access (25 minutes)

Slide: 10.01 - Remote Data Base Access *

Text Page: Remote Data Base Access Methods

Lesson 2. TurboIMAGE Profiler (10 minutes)

Slides: 10.02 - TurboIMAGE Profiler *
10.03 - User Interface Program

Lesson 3. RAPID/3000 (30 minutes)

Slides: 10.04 - RAPID/3000 *
10.05 - DICTIONARY/3000
10.06 - DICTIONARY/3000 Example
10.07 - TRANSACT/3000
10.08 - REPORT/3000 *

Text Page: INFORM/3000

Lesson 4. ADAGER/3000 (5 minutes)

Slides: 10.09 - ADAGER/3000



Module 10-1

Data Base Tools

Goal: To present an overview of data base tools.

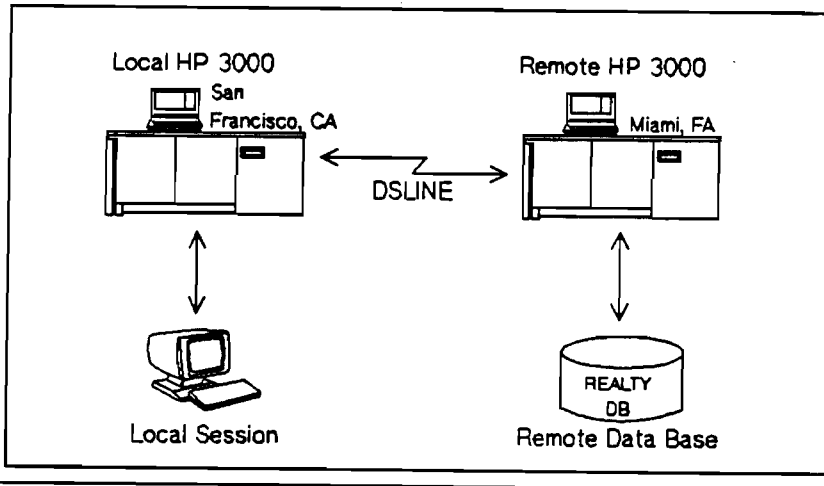
Objectives:

Upon completion of this module, you will be able to:

- describe five methods of remote data base access.
- describe the function of TurboIMAGE Profiler.
- identify the four RAPID/3000 products.
- state several functions of ADAGER/3000.

Remote Data Base Access

Refer to *DS/3000 Reference Manual* and *TurboIMAGE Data Base Management System Reference Manual*, Section 10: Using a Remote Data Base.



Data Base Tools

Purpose: To present five methods of remote data base access.

Key Points: *Check the end of this module for preparation information.*

- Wonder Realty has offices across the U.S., each with its own HP 3000 running a data base application. Printed reports from the different areas are routed via the U.S. mail. However, the agents need information on listings more quickly to help clients relocate. Therefore, a distributed systems network (DSN) and remote data base access (RDBA) can be implemented to solve the problem.
- The five methods of RDBA are: remote commands, remote file access, programmatic use of the COMMAND intrinsic, RDBA file, and PTOP.
- A DS link consists of a local HP 3000 and an established communications link with a remote HP 3000. A session initiated on the local HP 3000 is a local session and a session initiated on the remote HP 3000 is a remote session. There are various ways of opening the communications link and initiating the remote session.
- Accessing a remote data base increases response time and system overhead.
- Follow the 80/20 rule: do 80% of the data base accesses locally and no more than 20% on the remote HP 3000.



Module 10-3

□ Data Base Tools

Remote Data Base Access Methods

Remote Data Base Access (RDBA) is allowed between IMAGE/3000 and TurboIMAGE systems in both directions, providing the IMAGE/3000 limits are not exceeded with the TurboIMAGE data base. If the limits have been exceeded, RDBA is only allowed from TurboIMAGE to IMAGE/3000, and NOT from IMAGE/3000 to TurboIMAGE.

Refer to the *TurboIMAGE Data Base Management System Reference Manual*, Section 9: Using a Remote Data Base, for complete details on remote data base access.

Method 1: Remote Commands

```
:HELLO SF.WONDER           << initiate a local session       >>
:DSLIN OUTSIDE              << device class name for DS line  >>
:REMOTE HELLO MIAMI.WONDER  << initiate a remote session     >>
:REMOTE RUN DBAPPL          << run data base application     >>
(:REMOTE RUN QUERY.PUB.SYS is an alternate command.)
```

Refer to *DS/3000 Reference Manual*, Sections 1-3 for a detailed description of this method.

Method 2: Remote File Access

```
:HELLO SF.WONDER           << initiate a local session       >>
:DSLIN OUTSIDE              << open communications link     >>
:REMOTE HELLO MIAMI.WONDER  << initiate a remote session     >>
:FILE REALTY;DEV=OUTSIDE#DISC
                             << The FILE equation specifies   >>
                             << which data base is to be    >>
                             << accessed, and on which remote >>
                             << system and device. A local   >>
                             << application program can now >>
                             << access a remote data base. >>
:RUN DBAPPL                 << run the local program.       >>
```

Method 3: Programmatic Use of 'COMMAND'

This method is basically the same as Method 2, except the commands are executed programmatically.

```
:HELLO SF.WONDER           << initiate a local session       >>
:RUN CMDPGM                << The local application contains >>
                             << a call to the MPE COMMAND   >>
                             << intrinsic. COMMAND intrinsic >>
                             << is called for each of the   >>
                             << following commands:        >>
                                COMMAND 'DSLIN OUTSIDE'
                                COMMAND 'REMOTE HELLO MIAMI.WONDER'
                                COMMAND 'FILE REALTY;DEV=OUTSIDE#DISC'
                                .
                                .
                                . program code
```

Refer to the *MPE Intrinsic Reference Manual*, Section 2: Intrinsic Descriptions, for the details and the syntax of the COMMAND intrinsic.

Data Base Tools**Method 4: RDBA File**

This method involves creating a special file that provides TurboIMAGE with the necessary information to establish a communications link and a remote session. The file specifies the remote data base name so that the necessary TurboIMAGE intrinsics can be executed on the remote system. This file must be activated by using: :RUN DBUTIL.PUB.SYS, >>ACTIVATE *rdba file name*.

The following commands are issued:

```
:HELLO SF.WONDER
:RUN RDBALIST      << This program opens the RDBA File >>
```

The RDBA File consists of:

```
Record #1: FILE REALTY;DEV=OUTSIDE#DISC
Record #2: DSLINE OUTSIDE
Record #3: MIAMI.WONDER=HELLO
      .
      .
      . etc.
```

Method 5: PTOP(Program-to-Program)

```
:HELLO SF.WONDER
:DSLIN OUTSIDE
:REMOTE HELLO MIAMI.WONDER
:RUN MASTER      << This program creates and controls
                    a slave program on the remote system.
                    The slave program executes the
                    TurboIMAGE procedures and passes the
                    data back to the master program to be
                    displayed. >>
```

Module 10-4 Instructor Notes

Data Base Tools

Purpose: To present five methods of remote data base access.

Key Points:

- Method 1 establishes a communications link with the remote system where the data base resides. An application program that resides on the remote system is used to access the data base.
- Method 2 establishes a communications link with the remote system. A file equation is then used to specify the remote data base. Then a local application is run to access the data base.
- Method 3 performs the same steps as Method 2, except the commands are specified programmatically using the COMMAND intrinsic.
- Method 4 uses a special file that contains the necessary commands to establish the communications link and to initiate a remote session.
- Method 5 illustrates a program that creates and controls another program on the remote system.

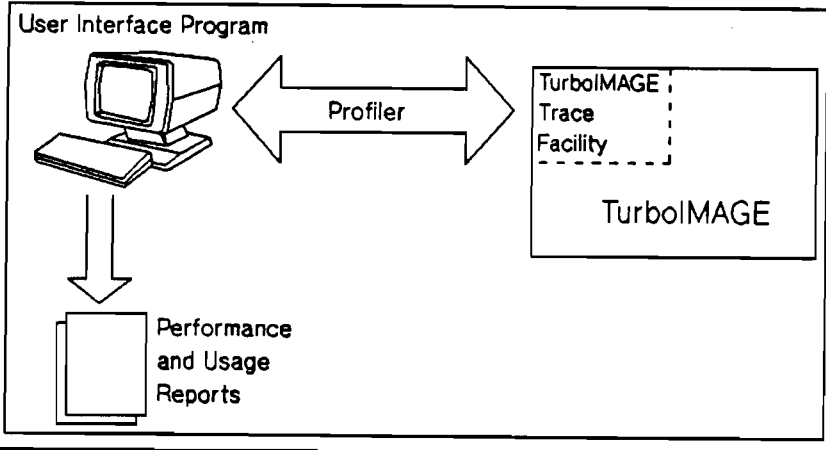


Data Base Tools

Notes

TurboIMAGE Profiler

- A data base diagnostic analysis tool.



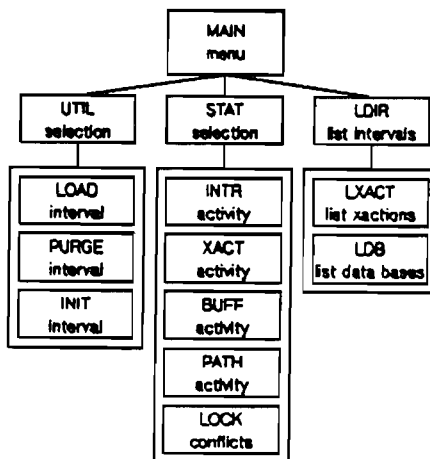
Refer to *TurboIMAGE Profiler User's Guide*

- TurboIMAGE Profiler reads trace data that is recorded by the TurboIMAGE trace facility, and produces various types of reports.
- TurboIMAGE Profiler is a powerful tool for fine tuning existing data bases as well as designing new ones.
- TurboIMAGE Profiler will NOT work with IMAGE/3000 data bases.

User Interface Program

- A hierarchy of VPLUS/3000 forms

- Three groups:
 - Utilities Selection
 - Statistics Selection
 - Directory Listings



□ Data Base Tools

Purpose: To present an overview of TurboIMAGE Profiler.

Key Points:

- TurboIMAGE Profiler is a data base and application program diagnostic analysis tool that is designed to work directly with TurboIMAGE data bases.
- TurboIMAGE Profiler is designed for use by data base administrators or programmers who are responsible for maintaining and interfacing with TurboIMAGE data bases.
- TurboIMAGE Profiler reads trace data that is recorded by the TurboIMAGE Trace Facility and produces various types of reports. Reports can be generated in the following areas: intrinsic activity, transaction activity, buffer effectiveness, path effectiveness, and lock conflict analysis.
- TurboIMAGE Profiler is a powerful tool for fine tuning existing data bases and designing new ones.
- Since TurboIMAGE Profiler uses trace data provided by the TurboIMAGE Trace Facility, users must have TurboIMAGE installed on their systems. TurboIMAGE Profiler will NOT work on iMAGE/3000 data bases.
- TurboIMAGE Profiler has a friendly user interface called User Interface Program.
- TurboIMAGE Profiler is packaged with three Dictionary/3000 utilities, DICTDBA, DICTDBU, and DICTDBL. This package of products is called DATABASE TOOLS.

Purpose: To present the hierarchy of the User Interface Program.

Key Points:

- The User Interface Program consists of a hierarchy of VPLUS/3000 screens. The user steps through the forms to construct desired reports.
- The Utilities Selection group of forms allows the user to manage trace interval files.
- The Statistics Selection group of forms allows the user to request reports in each of the five areas: intrinsic activity, transaction activity, buffer activity, path activity, and lock conflicts.
- The Directory Listings group of forms allows the user to monitor existing trace intervals.

Rapid/3000

- Dictionary/3000: Defines data and provides a data directory.
- Transact/3000: High-level programming language.
- Report/3000: Programmer's report writer.
- Inform/3000: End-user's report writer.

Customer Training Courses available:

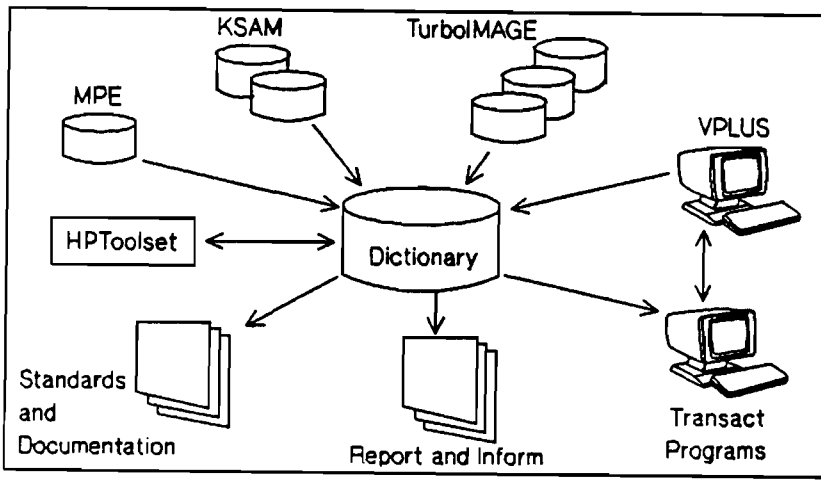
Dictionary-Inform-Report/3000 (35013C)

Transact/3000 (35014B)

Advanced Transact/3000 (51402A)

Dictionary/3000

- Stores data about data.



□ Data Base Tools

Purpose: To provide an overview of the Rapid/3000 products.

Key Points: *Check the end of this module for preparation information.*

- The Rapid/3000 products interface with TurboIMAGE data bases to make data base access simpler, faster, and more flexible.
- Dictionary/3000 defines data and provides a data directory.
- Transact/3000 is a high-level programming language.
- Report/3000 is a programmer's report writer.
- Inform/3000 is an end-user's report writer.

Purpose: To present an overview of Dictionary/3000.

Key Points:

- Dictionary/3000 provides a data dictionary for TurboIMAGE data bases, KSAM and MPE files, VPLUS forms files, programs, and reports.
- Dictionary/3000 uses TurboIMAGE data bases to store "data about data."
- Dictionary/3000 provides data definition and access information for Transact/3000, Report/3000, and Inform/3000.
- Dictionary/3000 provides special utilities for use with TurboIMAGE data bases. These utilities can create new data bases, backload existing data bases into a dictionary, unload and load data bases to disc or tape with more restructuring capability, and audit chain lengths and synonyms.
- Dictionary/3000 provides utilities to extract dictionary definitions as COBOL ENVIRONMENT and DATA division declarations and as Pascal TYPE and VAR declarations.

Data Base Tools

Notes

Dictionary/3000 Example

Dictionary/3000 HP32244A.02.00 - ICI Hewlett-Packard Co. 1985

ELEMENT: TYPE: SIZE: DEC: LENGTH: COUNT: RESPONSIBILITY:
 CITY ABBR X 4 0 4 1 MGMT

LONG NAME: CITY ABBREVIATION
 HEADING TEXT: CITY
 ENTRY TEXT: PLEASE ENTER CITY ABBREVIATION
 EDIT MASK:

MEASUREMENT UNITS
 DATE CREATED: 85/04/22 BY MGR
 DATE CHANGED: 85/04/22 BY MGR

DESCRIPTION:
 FOUR CHARACTER CITY ABBREVIATION

FILE:	TYPE: ELEMENT (ALIAS):
CITY- MASTER	MAST CITY-ABBR
RESIDENTIAL	DETL CITY-ABBR
	CHAIN MASTER SET: ICITY-MASTER
COMMERCIAL	DETL CITY ABBR
	CHAIN MASTER SET: CITY-MASTER

1 RECORD FOUND

Transact/3000

■ Fourth-generation language

DELETE CITY-MASTER:

Transact/3000

```

300-DELETE-MASTER.
MOVE "CITY-MASTER;" TO DSET-NAME.
CALL "DBGET" USING BASE DSET-NAME MODE7
STATUS-AREA ITEM-UST BUFFER ARG-VALUE.
IF C-W NOT = 0
GO- TO 900-ERROR.
CALL "DBDELETE" USING BASE DSET-NAME MODE1
STATUS-AREA.
IF C-W NOT = 0
GO TO 900-ENTER.
STOP RUN.
399-EXIT
900-ERROR.
CALL "DBEXPLAIN" USING STATUS-AREA.
STOP RUN.
    
```

COBOL II

Module 10-7 Instructor Notes: Slides 10.06/10.07

Data Base Tools

Purpose: To provide an example of Dictionary/3000.

Key Points:

- The slide presents the information stored in Dictionary/3000 about the data item CITY-ABBR in the REALTY data base.
- The element section of the report lists the type and size of the data item.
- The description section of the report lists the data sets that contain the data item CITY-ABBR.

Purpose: To introduce and give an example of Transact/3000.

Key Points:

- Transact/3000 is a fourth-generation language used for programming transaction-oriented applications.
- Transact/3000 contains built-in procedures for TurboIMAGE and VPLUS/3000 calls.
- Transact/3000 can be used with Dictionary/3000 to eliminate data definitions.
- Transact/3000 is excellent for speedy program development.

Report/3000

■ Report-writing language

```
REPORT HOUSE;
SELECT LISTING-NR;
OPTION NOHEAD;
REPORT TITLE "RESIDENTIAL LISTING",COL=4;
DETAIL "LISTING NUMBER:",LINE=3:LISTING-NR,SPACE=1;
      "LIST PRICE:",LINE=1,COL=1:LIST-PRICE,SPACE=1;
      "# BEDROOMS:",LINE=1:NUMBER-BEDS,SPACE=1;
      "# BATHS:",LINE=1:NUMBER-BATHS,SPACE=1;
      "DINING ROOM?":LINE=1:FORMAL-DINING-RM,SPACE=1;
      "SOLD?":LINE=1:SOLD-FLAG,SPACE=1;
      "SQUARE FEET:",LINE=1:SQUARE-FEET,SPACE=1;
      "CITY:",LINE=1:CITY-ABBR,SPACE=1;
END;
```

Source
File

RESIDENTIAL LISTING

```
LISTING NUMBER: 21
LIST PRICE: $162K
# BEDROOMS: 3
# BATHS: 2.00
DINING ROOM? N
SOLD? N
SQUARE FEET: 1245
CITY: MH

LISTING NUMBER: 22
LIST PRICE $229K
# BEDROOMS: 4
# BATHS: 3.00
DINING ROOM? Y
SOLD? N
SQUARE FEET: 1439
CITY: MH
:
:
```

Report
Output

Data Base Tools

Purpose: To introduce and give an example of Report/3000.

Key Points:

- Report/3000 is a report-writing language that allows complex access and formatting.
- Report/3000 can generate reports from multiple sets, data bases, MPE and KSAM files.
- Report/3000 allows a "relational view" of the data. The programmer can define new relationships between data items, sets, or MPE files that do not currently exist.

Module 10-9

Data Base Tools

INFORM/3000

I. MENU

DATA NAMES IN DATA SET RESIDENTIAL

1: LISTING-NR	5: NUMBER-BATHS	9: ZIP-CODE
2: CITY-ABBR	6: CURRENT-OWNER	10: FORMAL-DINING-RM
3: LIST-PRICE	7: OWNERS-PHONENR	11: SOLD-FLAG
4: NUMBER-BEDS	8: STREET-ADDRESS	12: SQUARE-FEET

TYPE NUMBER(S) FOR DATA NAME(S):

TO INCLUDE IN REPORT> 1,3,8,2
TO SORT BY> 3,1
TO SUBDIVIDE BY>
TO GRAND TOTAL>
FOR SELECTION CRITERIA> 3

REPORT TITLE> RESIDENTIAL LISTING REPORT

II. REPORT

MON, JUL 26, 1985, 4:15 PM

RESIDENTIAL LISTING REPORT

LISTING NO.	PRICE	ADDRESS	CITY
29	\$161K	5433 DOVER DR	SC
33	\$162K	5600 PARTER RD	SJ
21	\$162K	534 BLACKFORD RD	MH
5	\$167K	8876 HARTE AVE	CAMB
32	\$168K	5566 ULTIMA DR	SJ
.	.	.	.
.	.	.	.
.	.	.	.

Module 10-9 Instructor Notes

Data Base Tools

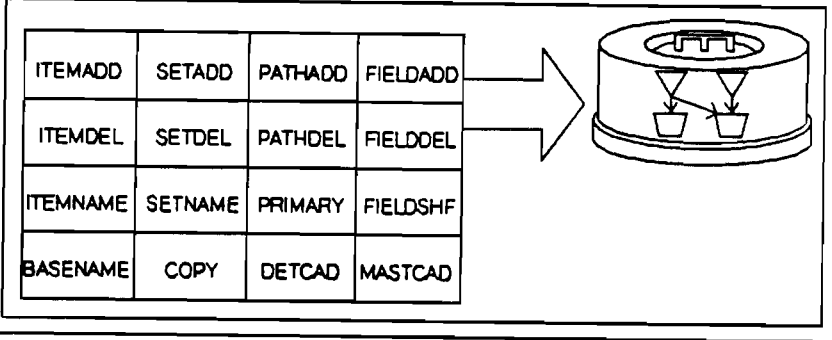
Purpose: To introduce and give an example of Inform/3000.

Key Points:

- Inform/3000 is a fast and easy report generator that is menu driven.
- Simple reports can be quickly defined, saved, modified, and printed.
- Inform/3000 is easy to use for non-programmers.
- Inform/3000 must operate in conjunction with Dictionary/3000.
- Reports can be produced from TurboIMAGE, KSAM, and MPE files.
- Due to the simplicity of Inform/3000, it has more limitations than Report/3000 in terms of formatting and "relational views" of data. However, it can often meet end-users' ad-hoc reporting requirements.
- Inform/3000 reports can be converted to Report/3000 format and then edited.

ADAGER/3000

- A set of modules.
- Allows structure transformations without RELOADs or schema recompilations.
- Supported by HP.



ADAGER/3000 is a product of ADAGER S.A.

Data Base Tools

Purpose: To present an overview of ADAGER/3000.

Key Points:

- ADAGER/3000 evolved from a user-developed software system.
- ADAGER/3000 accomplishes transformations of a data base structure without reloading the entire data base. ADAGER/3000 deals only with the structure that must be transformed and leaves everything else untouched.
- ADAGER/3000 contains modules that: add, delete, or rename items, sets, and paths; change capacities of data sets; create a copy of an existing data base; and move data sets to specified disc logical units.
- ADAGER/3000 is a product of ADAGER S.A. but is fully supported by Hewlett-Packard Co.

Module 10-11 Instructor Notes

Preparation Material: Slides 10.01/10.02

Teaching Tips:

This section is an overview of remote data base access. Present the Wonder Realty situation as an example of why remote data base access is needed.

The text page that follows covers the five methods of RDBA. Emphasize that the *DS/3000 Reference Manual* should be read prior to attempting remote data base access.

Teaching Tips:

Emphasize to students that slides 10.02 and 10.03 provide an introduction to TurboIMAGE Profiler. This course does not provide training on TurboIMAGE Profiler. The user's guide is provided with the product.

Module 10-12 Instructor Notes

Preparation Material: Slides 10.04, 10.08

Teaching Tips:

Write, on the slide, a brief definition of each product.

The RAPID/3000 products are introduced briefly in this module. For training in the use of the RAPID/3000 products the following customer training courses are available:

Dictionary-Inform-Report/3000 (35013C) - This is a four-day course that introduces the application programmer or data base administrator to all aspects of the three products.

Transact/3000 (35014B) - This is a five-day course for application programmers.

Advanced Transact/3000 (51402A) - This is a five-day course for programmers with at least six months experience with Transact/3000.

Teaching Tips:

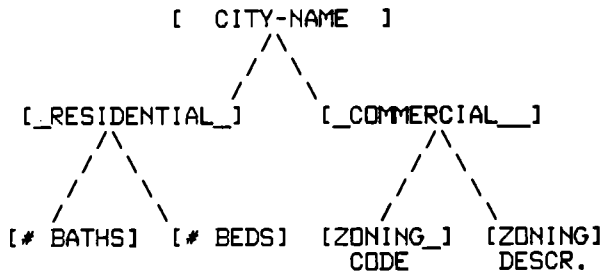
Point out that this slide shows a very simple example of a REPORT/3000 report and does not fully illustrate the capabilities of REPORT/3000.



□ Solution: Activity 1-2

Worksession: Data Base Models

Hierarchical Structure

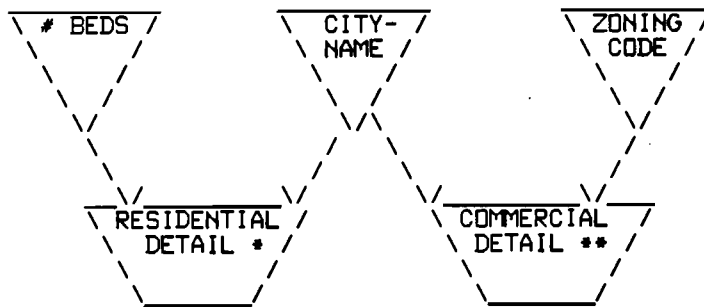


Relational Structure
(2 possible tables)

ZONING	ZONING	CITY-
DESCR.	CODE	NAME

CITY-	ADDRESS	LIST-
NAME		PRICE

Network Structure (TurboIMAGE)



* city-name, address, owner's name, listing price, # of beds, # of baths, # of square feet.

** city-name, zoning code, zoning description, address, owner's name, # of square feet, listing price.

Module Appendix A-2

□ Solution: Activity 2-3

Lab 3 -- Solution

1. & 2. By entering `FO SETS`, a list of the data sets will appear. The type, item count, capacity, entry count, entry length, and blocking factor is listed for each set. There are 8 data sets.

The automatic masters are: `LISTNR-MASTER`, `LIST-PRICE-MSTR`, `BATH-MASTER`, `BEDS-MASTER`.

The manual masters are: `ZONING-MASTER`, `CITY-MASTER`.

The detail data sets are: `RESIDENTIAL`, `COMMERCIAL`.

3. Enter `FO ITEMS` to display a list of the data item names and the data types. There are 17 data items defined.
4. Enter `FO PATHS` to display path identifying information. There are 9 paths defined.
5. `SQUARE-FEET` is a sort item in the `RESIDENTIAL` set and the `COMMERCIAL` set. `LISTING-NR`, `CITY-ABBR`, `LIST-PRICE`, `NUMBER-BEDS`, and `NUMBER-BATHS` are search items in the `RESIDENTIAL` detail set. `LISTING-NR`, `ZONING-CODE`, `LIST-PRICE`, and `CITY-ABBR` are search items in the `COMMERCIAL` detail set.
6. `>LIST COMMERCIAL`
7. `>ADD COMMERCIAL`
- | | |
|------------------------------|-------------------------------|
| <code>LISTING-NR</code> | <code>=>>99</code> |
| <code>ZONING-CODE</code> | <code>=>>R3</code> |
| <code>LIST-PRICE</code> | <code>=>>_____</code> |
| <code>CITY-ABBR</code> | <code>=>>LA</code> |
| <code>STREET-ADDRESS</code> | <code>=>>_____</code> |
| <code>ZIP-CODE</code> | <code>=>>_____</code> |
| <code>EXISTING-STRUCT</code> | <code>=>> Y or N</code> |
| <code>OCCUPIED</code> | <code>=>> Y or N</code> |
| <code>SOLD-FLAG</code> | <code>=>> Y or N</code> |
| <code>SQUARE-FEET</code> | <code>=>>_____</code> |
8. `>FIND LIST-PRICE IB 150,200 AND &`
`>>OCCUPIED IS N AND &`
`>>SQUARE-FEET >00001200`
- 1 ENTRY QUALIFIES
9. `>FIND LISTING-NR=99`
`>REPLACE,`
`>> item name = " ";`
`>>END`
`>OUTPUT=LP`
`>REPORT ALL`
10. `>OUTPUT=TERM`
11. `>REPORT ALL`
`>DELETE`
`>>YES`
12. `>EXIT`

Module Appendix A-3

□ Solution: Activity 4-1

Miniature REALTY Data Base - Part One

BEGIN DATA BASE REALTY;

PASSWORDS:

ITEMS:

LISTING-NR,	I	;	<< UNIQUE, SEQUENTIAL NUMBER >>
ZONING-CODE,	X4	;	<< ZONING CODE >>
ZONING-DESCRIP,	X20	;	<< ZONING DESCRIPTION >>
LIST-PRICE,	I	;	<< LIST PRICE TO NEAREST \$1K >>
STREET-ADDRESS,	X30	;	<< STREET ADDRESS >>
SOLD-FLAG,	X2	;	<< 'Y' OR BLANK >>
SQUARE-FEET,	X8	;	<< SQUARE FEET >>

SETS:

NAME: LISTNR-MASTER, AUTOMATIC;
ENTRY: LISTING-NR (1);
CAPACITY: 503;

NAME: ZONING-MASTER, MANUAL;
ENTRY: ZONING-CODE (1),
ZONING-DESCRIP;
CAPACITY: 31;

NAME: COMMERCIAL, DETAIL;
ENTRY: LISTING-NR (LISTNR-MASTER),
ZONING-CODE (!ZONING-MASTER),
LIST-PRICE,
STREET-ADDRESS,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

END.

Module Appendix A-4

Solution: Activity 4-2

Miniature REALTY Data Base - Part Two

PAGE 1 HEWLETT-PACKARD 32215B.04.30 IMAGE/3000: DBSCHEMA
TUE, DEC 18, 1984, 3:41 PM (C) HEWLETT-PACKARD CO. 1978

\$TITLE 'WONDER REALTY DATA BASE'

BEGIN DATA BASE REALTY;

PASSWORDS:

ITEMS:

LISTING-NR,	I	; << UNIQUE, SEQUENTIAL NUMBER >>
ZONING-CODE,	X4	; << ZONING CODE >>
ZONING-DESCRIP,	X20	; << ZONING DESCRIPTION >>
LIST-PRICE,	I	; << LIST PRICE TO NEAREST \$1K >>
STREET-ADDRESS,	X30	; << STREET ADDRESS >>
SOLD-FLAG,	X2	; << 'Y' OR BLANK >>
SQUARE-FEET,	X8	; << SQUARE FEET >>

\$TITLE 'MASTER DATA SETS'

SETS:

\$CONTROL BLOCKMAX=128

NAME: LISTNR-MASTER, AUTOMATIC;
ENTRY: LISTING-NR (1);
CAPACITY: 503;

NAME: ZONING-MASTER, MANUAL;
ENTRY: ZONING-CODE (1),
ZONING-DESCRIP;
CAPACITY: 31;

\$TITLE 'DETAIL DATA SETS'

\$CONTROL BLOCKMAX=768

NAME: COMMERCIAL, DETAIL;
ENTRY: LISTING-NR (LISTNR-MASTER),
ZONING-CODE (!ZONING-MASTER),
LIST-PRICE,
STREET-ADDRESS,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

END.

Module Appendix A-5

Solution: Activity 4-2

Miniature REALTY Data Base - Part Two (cont.)

LISTNR-MASTER	A	1	1	1	11	503	11	122	48
ZONING-MASTER	M	2	1	12	22	31	5	111	9
COMMERCIAL	D	6	2	24	32	322	23	738	90

TOTAL DISC SECTORS INCLUDING ROOT: 156

NUMBER OF ERROR MESSAGES: 0

ITEM NAME COUNT: 7 DATA SET COUNT: 3

ROOT LENGTH: 360 BUFFER LENGTH: 738 TRAILER LENGTH: 256

ROOT FILE REALTY CREATED.

Module Appendix A-6

□ Solution: Activity 4-3

\$TITLE 'WONDER REALTY DATA BASE'

BEGIN DATA BASE REALTY;

PASSWORDS:

ITEMS:

LISTING-NR,	I	; << UNIQUE, SEQUENTIAL NUMBER >>
NUMBER-BEDS,	X2	; << NUMBER OF BEDROOMS >>
CITY-ABBR,	X4	; << CITY ABBREVIATION >>
CITY-NAME,	X20	; << CITY NAME >>
ZONING-CODE,	X4	; << ZONING CODE >>
ZONING-DESCRIP,	X20	; << ZONING DESCRIPTION >>
NUMBER-BATHS,	X4	; << NUMBER OF BATHS (d.dd) >>
LIST-PRICE,	I	; << LIST PRICE TO NEAREST \$1K >>
CURRENT-OWNER,	X20	; << CURRENT OWNER >>
OWNERS-PHONENR,	X10	; << OWNERS PHONE NUMBER >>
STREET-ADDRESS,	X30	; << STREET ADDRESS >>
ZIP-CODE,	X6	; << ZIP CODE (5 DIGIT) >>
FORMAL-DINING-RM,	X2	; << "Y" OR BLANK >>
SOLD-FLAG,	X2	; << "Y" OR BLANK >>
SQUARE-FEET,	X8	; << SQUARE FEET >>
EXISTING-STRUCT,	X2	; << "Y" OR BLANK >>
OCCUPIED,	X2	; << "Y" OR BLANK >>

\$TITLE 'MASTER DATA SETS'

SETS:

\$CONTROL BLOCKMAX=128

NAME: LISTNR-MASTER, AUTOMATIC ;
ENTRY: LISTING-NR(2);
CAPACITY: 503;

NAME: LIST-PRICE-MSTR, AUTOMATIC ;
ENTRY: LIST-PRICE(2);
CAPACITY: 307;

NAME: BATH-MASTER, AUTOMATIC ;
ENTRY: NUMBER-BATHS(1);
CAPACITY: 31;

NAME: BEDS-MASTER, AUTOMATIC ;
ENTRY: NUMBER-BEDS (1);
CAPACITY: 31;

NAME: ZONING-MASTER, MANUAL ;
ENTRY: ZONING-CODE (1),
ZONING-DESCRIP;
CAPACITY: 31;

NAME: CITY-MASTER, MANUAL ;
ENTRY: CITY-ABBR (2),
CITY-NAME;
CAPACITY: 101;

\$TITLE 'DETAIL DATA SETS'

\$CONTROL BLOCKMAX=640

Module Appendix A-7

Solution: Activity 4-3

\$TITLE "DETAIL DATA SETS"

\$CONTROL BLOCKMAX=640

NAME: RESIDENTIAL, DETAIL ;
ENTRY: LISTING-NR (LISTNR-MASTER),
CITY-ABBR (!CITY-MASTER),
LIST-PRICE (LIST-PRICE-MSTR (SQUARE-FEET)),
NUMBER-BEDS (BEDS-MASTER),
NUMBER-BATHS (BATH-MASTER),
CURRENT-OWNER,
OWNERS-PHONENR,
STREET-ADDRESS,
ZIP-CODE,
FORMAL-DINING-RM,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

\$CONTROL BLOCKMAX=768

NAME: COMMERCIAL, DETAIL ;
ENTRY: LISTING-NR (LISTNR-MASTER),
ZONING-CODE (!ZONING-MASTER),
LIST-PRICE (LIST-PRICE-MSTR (SQUARE-FEET)),
CITY-ABBR (CITY-MASTER),
STREET-ADDRESS,
ZIP-CODE,
EXISTING-STRUCT,
OCCUPIED,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

END.

Module Appendix A-8

□ Solution: Activity 5-1

Solution after 1), 2), and 3)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	:	:	:	MV	MOUNTAIN VIEW	CITY-MASTER
41	2	43	43	:	:	:	PA	PALO ALTO	
42	XXX	XXX	XXX	XXX:	XXX:	XXX	XX	XXXXXXXXXXXXXX	DATA SET
43	0	00	00	:	:	:	SJ	SAN JOSE	(MIGRATED FROM 40)

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS
161	:	:	- - - - -
162	:	:	- - - - -
163	:	:	- - - - -
164	:	:	- - - - -
165	:	:	- - - - -
166	:	:	- - - - -

RESIDENTIAL DETAIL SET

Solution after Steps 4) through 9)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	1	165	165	MV	MOUNTAIN VIEW	CITY-MASTER
41	2	43	43	1	161	161	PA	PALO ALTO	
42	XXX	XXX	XXX	XXX:	XXX:	XXX	XX	XXXXXXXXXXXXXX	DATA SET
43	0	00	00	4	166	162	SJ	SAN JOSE	

LISTING NR	PATH 1	CITY ABBR	OTHER DATA ITEMS
161	00 : 00	01 PA	- - - - -
162	00 : 163	02 SJ	- - - - -
163	162 : 164	03 SJ	- - - - -
164	163 : 166	04 SJ	- - - - -
165	00 : 00	05 MV	- - - - -
166	164 : 00	06 SJ	- - - - -

RESIDENTIAL DETAIL SET

Module Appendix A-9

□ Solution: Activity 5-1 (cont'd)

Solution after 10)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	1	165	165	MV	MOUNTAIN VIEW	CITY-MASTER DATA SET
41	1	00	00	4	166	162	SJ	SAN JOSE	
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	ENTRY MIGRATED TO LOCATION 41								

LISTING NR	PATH 1		CITY ABBR	OTHER DATA ITEMS	
	/	\			
161	ENTRY DELETED				
162	00	163	02	SJ	RESIDENTIAL DETAIL SET
163	162	164	03	SJ	
164	163	166	04	SJ	
165	00	00	05	MV	
166	164	00	06	SJ	

Solution after 11)

	SYNONYM CHAIN HEAD			DETAIL SET CHAIN HEAD			CITY ABBR	CITY-NAME	
40	1	00	00	1	165	165	MV	MOUNTAIN VIEW	CITY-MASTER DATA SET
41	1	00	00	3	166	162	SJ	SAN JOSE	
42	XXX	XXX	XXX	XXX	XXX	XXX	XX	XXXXXXXXXXXXXX	
43	ENTRY MIGRATED TO LOCATION 41								

LISTING NR	PATH 1		CITY ABBR	OTHER DATA ITEMS	
	/	\			
161	DELETED				
162	00	164	02	SJ	RESIDENTIAL DETAIL SET
163	DELETED				
164	162	166	04	SJ	
165	00	00	05	MV	
166	164	00	06	SJ	

Module Appendix A-10

□ Solution: Activity 6-7

\$TITLE 'WONDER REALTY DATA BASE'

BEGIN DATA BASE REALTY;

PASSWORDS:

10	RECEPT;	<< RECEPTIONIST	>>
20	SALESREP;	<< SALES PERSON	>>
30	MANAGER;	<< SALES MANAGER	>>

ITEMS:

LISTING-NR,	I	(10,20/30);	<< UNIQUE, SEQUENTIAL NUMBER>>	>>
NUMBER-BEDS,	X2	(10,20/30);	<< NUMBER OF BEDROOMS	>>
CITY-ABBR,	X4	(10,20/30);	<< CITY ABBREVIATION	>>
CITY-NAME,	X20	(10,20/30);	<< CITY NAME	>>
ZONING-CODE,	X4	(10,20/30);	<< ZONING CODE	>>
ZONING-DESCRIP,	X20	(10,20/30);	<< ZONING DESCRIPTION	>>
NUMBER-BATHS,	X4	(10,20/30);	<< NUMBER OF BATHS (d.dd)	>>
LIST-PRICE,	I	(10,20/30);	<< LIST PRICE TO NEAREST \$1K>>	>>
CURRENT-OWNER,	X20	(20/30);	<< CURRENT OWNER	>>
OWNERS-PHONENR,	X10	(20/30);	<< OWNERS PHONE NUMBER	>>
STREET-ADDRESS,	X30	(20/30);	<< STREET ADDRESS	>>
ZIP-CODE,	X6	(10,20/30);	<< ZIP CODE (5 DIGIT)	>>
FORMAL-DINING-RM,	X2	(10,20/30);	<< "Y" OR BLANK	>>
SOLD-FLAG,	X2	(10,20/30);	<< "Y" OR BLANK	>>
SQUARE-FEET,	X8	(10,20/30);	<< SQUARE FEET	>>
EXISTING-STRUCT,	X2	(10,20/30);	<< "Y" OR BLANK	>>
OCCUPIED,	X2	(10,20/30);	<< "Y" OR BLANK	>>

\$TITLE 'MASTER DATA SETS'

SETS:

\$CONTROL BLOCKMAX=128

NAME: LISTNR-MASTER, AUTOMATIC (10,20/30);
ENTRY: LISTING-NR(2);
CAPACITY: 503;

NAME: LIST-PRICE-MSTR, AUTOMATIC (10,20/30);
ENTRY: LIST-PRICE(2);
CAPACITY: 307;

NAME: BATH-MASTER, AUTOMATIC (10,20/30);
ENTRY: NUMBER-BATHS(1);
CAPACITY: 31;

NAME: BEDS-MASTER, AUTOMATIC (10,20/30);
ENTRY: NUMBER-BEDS (1);
CAPACITY: 31;

NAME: ZONING-MASTER, MANUAL (10,20/30);
ENTRY: ZONING-CODE (1),
ZONING-DESCRIP;
CAPACITY: 31;

NAME: CITY-MASTER, MANUAL (10,20/30);
ENTRY: CITY-ABBR (2),
CITY-NAME;
CAPACITY: 101;

Module Appendix A-11

Solution: Activity 6-7 (cont'd)

\$TITLE 'DETAIL DATA SETS'

\$CONTROL BLOCKMAX=640

NAME: RESIDENTIAL, DETAIL (10,20/30);
ENTRY: LISTING-NR (LISTNR-MASTER),
CITY-ABBR (!CITY-MASTER),
LIST-PRICE (LIST-PRICE-MSTR (SQUARE-FEET)),
NUMBER-BEDS (BEDS-MASTER),
NUMBER-BATHS (BATH-MASTER),
CURRENT-OWNER,
OWNERS-PHONENR,
STREET-ADDRESS,
ZIP-CODE,
FORMAL-DINING-RM,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

\$CONTROL BLOCKMAX=768

NAME: COMMERCIAL, DETAIL (10,20/30);
ENTRY: LISTING-NR (LISTNR-MASTER),
ZONING-CODE (!ZONING-MASTER),
LIST-PRICE (LIST-PRICE-MSTR (SQUARE-FEET)),
CITY-ABBR (CITY-MASTER),
STREET-ADDRESS,
ZIP-CODE,
EXISTING-STRUCT,
OCCUPIED,
SOLD-FLAG,
SQUARE-FEET;
CAPACITY: 300;

END.

Module Appendix A-12

Solution: Activity 7-1b

Solutions to Optional Exercise

The situation described in this exercise occurs because of migrating secondaries (Module 5). If an entry is deleted and it is a primary on a synonym chain then the first secondary migrates into the primary position. The program then moves on to the next record and misses the migrated entry. Three possible solutions are described below.

Solution #1

After the DBDELETE succeeds, perform a DBGET mode 1 (re-read) to determine if an entry still exists in the same location. If no entry exists, continue with the serial read.

Solution #2

After the DBGET mode 2, perform a sequence of DBDELETES until the status returns a condition code of 17 (no entry exists). The sequence of DBDELETES will continually delete entries in the same location. Therefore, all entries that migrate into the location will be deleted.

Solution #3

After the call to DBGET, the status array (words 5 and 6) contains a count of the number of entries in the synonym chain (or 0 if the entry is a primary entry). This value can then be used to perform the exact number of DBDELETES necessary to delete all the entries in the synonym chain.

Module A-13

Solution: Activity 8-4

CASE 1: A, E, H, D, J, B, I

CASE 2: E, H, C, I

CASE 3: A, E, H, D, J, B, I

CASE 4: A, E, K, D, J, B, L

CASE 5: D, F

Module Appendix A-14

Solution: Activity 8-9

1. Tape, system domain disc, private volume disc, serial disc
2. True
3. False
4. DBBEGIN and DBEND
5. False
6. GETLOG, BUILD LOGFILE, DBUTIL - SET LOGID, DBUTIL - SET FLAGS, DBSTORE
7. Same as GETLOG command log file name
8. Same as GETLOG command logid
9. Enable for access, disable for recovery
10. Enable for recovery, disable for access
11. Because flags will be set correctly for recovery after DBRESTOR
12. >>SHOW REALTY ALL or >>SHOW REALTY FLAGS
13. DBSTORE
14. DBSTORE sets both the timestamp and the STORE flag in the root file
15. LISTF,2 or :SHOWLOGSTATUS
16. :CHANGELOG or :GETLOG,AUTO
17. A message appears on the system console requesting another tape be mounted.
18. Roll-forward completes interrupted transactions and roll-back undoes the transactions.
19. Roll-back is faster since it does not require the use of DBSTORE and DBRESTOR. The transactions in the log file are not re-executed.
20. True
21. Enabled for recovery, disabled for access, enabled for logging.
22. Timestamp in log file must be greater than or equal to that of the root file.
23. >RUN
24. structural integrity
25. The most recent DBPUT or DBDELETE

Module Appendix B-1

□ Schema Structure

Schema Structure

```
$CONTROL LIST, ERRORS=nnnn, LINES=nnnnn, ROOT, BLOCKMAX=nnnn,&  
$      TABLE
```

```
BEGIN DATA BASE data base name;
```

```
PASSWORDS:
```

```
    user class number [password];          << comments >>  
    .  
    user class number [password];          << comments >>
```

```
ITEMS:
```

```
    item name, [sub-item count] type designator [sub-item length]  
    [(read class list/write class list)];
```

```
SETS: (master data sets)
```

```
                {Manual  }  
                {M      }  
{NAME:}      set name, {Automatic} [(read class list/write class list)]  
{N:  }      {A      }  
                [, device class];
```

```
{ENTRY:}      item name [(path count)].  
{E:  }      .  
                item name [(path count)];
```

```
{CAPACITY:}   maximum entry count;  
{C:  }      }
```

```
: (detail data sets)
```

```
{NAME:}      set name, {DETAIL} [(read class list/write class list)]  
{N:  }      {D      }  
                [, device class];
```

```
{ENTRY:}      item name [(!] master set name [(sort item name)]].  
{E:  }      .  
                item name [(master set name [(sort item name)]]];
```

```
{CAPACITY:}   maximum entry count;  
{C:  }      }
```

```
END.
```

- The required words are shown in uppercase.
- Punctuations such as ":" after required words and ";" to terminate statements are required.
- The ENTRY syntax varies with the set type.
- << comments >> can contain any characters and can appear anywhere in the schema except embedded in another comment. They are included in the schema listing, but are otherwise ignored by the Schema Processor program.
- A user can issue a BLOCKMAX parameter but it can be overridden by TurboIMAGE if it is not an optimum blocking size.
- ! indicates the primary path. Only one primary path is allowed per detail set. The primary path can be defined by any item in the set.

Module Appendix B-2

□ DBUTIL.PUB.SYS

DBUTIL.PUB.SYS Utility Program

```
:RUN DBUTIL.PUB.SYS
>>ACT[IVATE] data-base-access file name
>>CRE[ATE] data base name[/maintenance word]
>>DEA[CTIVATE] data-base-access file name
>>DIS[ABLE] data base name[/maintenance word] FOR option[, option..]
>>ENA[BLE] data base name[/maintenance word] FOR option [,option..]
>>ERA[SE] data base name[/maintenance word]
>>EXI[IT]
>>HEL[P] [commandname]
>>MOV[E] TurboIMAGE file name TO device
>>PUR[GE] data base name[/maintenance word]
>>REL[EASE] data base name
>>SEC[URE] data base name
>>SET data base name[/maintenance word]
    {MAINT=maintenance word
    {BUFFSPECS=num buffers [from-users/to-users]
    {      [,num buffers[from-users/to-users]]...
    {LOGID=log identifier
    {PASSWORD classnum=[password]
    {SUBSYSTEM={NONE}
    {      {READ}
    {      {RW }
    {LANGUAGE=language id
>>SHO[W] data base name[/maintenance word]
    MAINT      }
    ALL        }
    BUFFSPECS }
    LANGUAGE   }
    LOCKS      } [OFFLINE]
    USERS      }
    LOGID      }
    FLAGS      }
    PASSWORDS }
    DEVICE     }
>>VER[IFY] data-base-access file name
    {CREATE}
* can also be :RUN DBUTIL.PUB.SYS, {ERASE }
    {PURGE }
```

Module Appendix B-3

□ TurboIMAGE Utilities

DBUTIL

ACTIVATE	Prepares a data-base-access file used when accessing a remote data base.
CREATE	Creates and initializes a data base file for each data set.
DEACTIVATE	Deactivates a data-base-access file.
DISABLE	Disables logging, roll-forward recovery, roll-back recovery, ILR, autodefer, access and dumping options.
ENABLE	Enables logging, roll-forward recovery, roll-back recovery, ILR, autodefer, access and dumping options.
EXIT	Terminates DBUTIL program execution.
HELP	Provides description of all other DBUTIL commands.
MOVE	Moves TurboIMAGE across devices.
PURGE	Purges entire data base including root file and data set files. Used before restoring a stored data base and before creating a new, restructured data base.
RELEASE	Suspends security provisions for the root file and data set files.
SECURE	Restores security provisions suspended by RELEASE command.
SET	Changes or removes the maintenance word, specifies number of buffers to be used, stores log identifier and password into root file. Changes the Native Language of the data base.
SHOW	Used to display information about data base maintenance.
VERIFY	Used to determine whether a data-base-access file is activated or deactivated.

Module Appendix B-4

TurboIMAGE Utilities (cont'd)

The TurboIMAGE utility programs are used to create and initialize the data base files and perform various maintenance functions. Here is a brief summary of the utility routines and their functions.

PROGRAM	COMMANDS	FUNCTION
<u>DBC</u> ONV		Converts IMAGE/3000 data base root file and all master data sets to TurboIMAGE format.
DBLOAD		Loads data entries copied by DBUNLOAD back into data sets.
DBRECOV	CONTROL	Controls various options which affect execution of DBRECOV.
	EXIT	Terminates DBRECOV without re-executing any transactions.
	FILE	Routes log records to individual user files and returns information about recovery.
	PRINT	Prints information about data bases or user files specified for recovery.
	RECOVER	Designates name of data base to be roll-forward recovered.
	<u>ROLLBACK</u>	Defines name of data base to be roll-back recovered.
	RUN	Initiates recovery process.
DBRESTOR		Copies the data base to disc from magnetic tape or serial disc volumes created by DBSTORE, or the MPE :STORE or :SYSDUMP commands.
DBSTORE		Copies entire data base including root file to magnetic tape or serial disc volumes.
DBUNLOAD		Copies data entries to specially formatted magnetic tape or serial disc volumes; arranges entries in each data set so that chained access along the primary path is more efficient.

□ Lock Descriptor Examples

LOCK DESCRIPTOR ARRAY EXAMPLES:

COBOL

```

01 LOCK-DESC-ARRAY
05 NUM-ARRAYS PIC S9(4) COMP VALUE +2.
05 LOCK-DESC-1.
10 DESC-LEN PIC S9(4) COMP VALUE +19.
10 DSET PIC X(16) VALUE "RESIDENTIAL".
10 DITEM PIC X(16) VALUE "LISTING-NR".
10 RELOP PIC X(2) VALUE "==".
10 LOCK-VAL PIC S9(4) COMP VALUE +103.
05 LOCK-DESC-2.

```

FORTRAN

```

INTEGER NUM,LENGTH,VALUE,LOCKDESC(20)
CHARACTER*2 RELOP
CHARACTER*16 DSETNAME, ITEMNAME

EQUIVALENCE (LOCKDESC(1),NUM), (LOCKDESC(2),LENGTH),
             (LOCKDESC(3),DSETNAME), (LOCKDESC(11), ITEMNAME),
             (LOCKDESC(19),RELOP), (LOCKDESC(20),VALUE)

NUM = 1
LENGTH=19
DSETNAME="RESIDENTIAL;    "
ITEMNAME="LISTING-NR;    "
RELOP==" "
VALUE=103

```

Pascal

```

TYPE
SMALL_INT = -32768..32767;
WORD2 = PACKED ARRAY[1..2] OF CHAR;
WORD16= PACKED ARRAY[1..16] OF CHAR;
LOCKTYPE=RECORD
    NUM: SMALL_INT;
    DESCRIPTOR: RECORD
        LEN: SMALL_INT;
        DSETNAME:WORD16;
        ITEMNAME:WORD16;
        RELOP:WORD2;
        VALUE:SMALL_INT;
    END; (* DESCRIPTOR *)
END; (* LOCKTYPE *)

```

VAR

```

LOCKDESC:LOCKTYPE;

WITH LOCKDESC DO
BEGIN
    NUM:=1;
    DESCRIPTOR.LEN:=19;
    DESCRIPTOR.DSETNAME:='RESIDENTIAL;    ' ;
    DESCRIPTOR.ITEMNAME:='LISTING-NR;    ' ;
    DESCRIPTOR.RELOP:='= ' ;
    DESCRIPTOR.VALUE:=103;
END;

```

Module Appendix B-6

Lock Descriptor Examples

SPL

```
ARRAY LOCKDESC(0:20);  
BYTE ARRAY LOCK'DESC(*)=LOCKDESC;
```

```
LOCKDESC:=1;  
LOCKDESC:=19;
```

```
MOVE LOCK'DESC(4):='RESIDENTIAL;    '';  
MOVE LOCK'DESC(20):='LISTING-NR;    '';  
MOVE LOCK'DESC(36):='= '';  
MOVE LOCK'DESC(38):=103;
```

BASIC

```
INTEGER I1,I2  
DIM I3$(37)
```

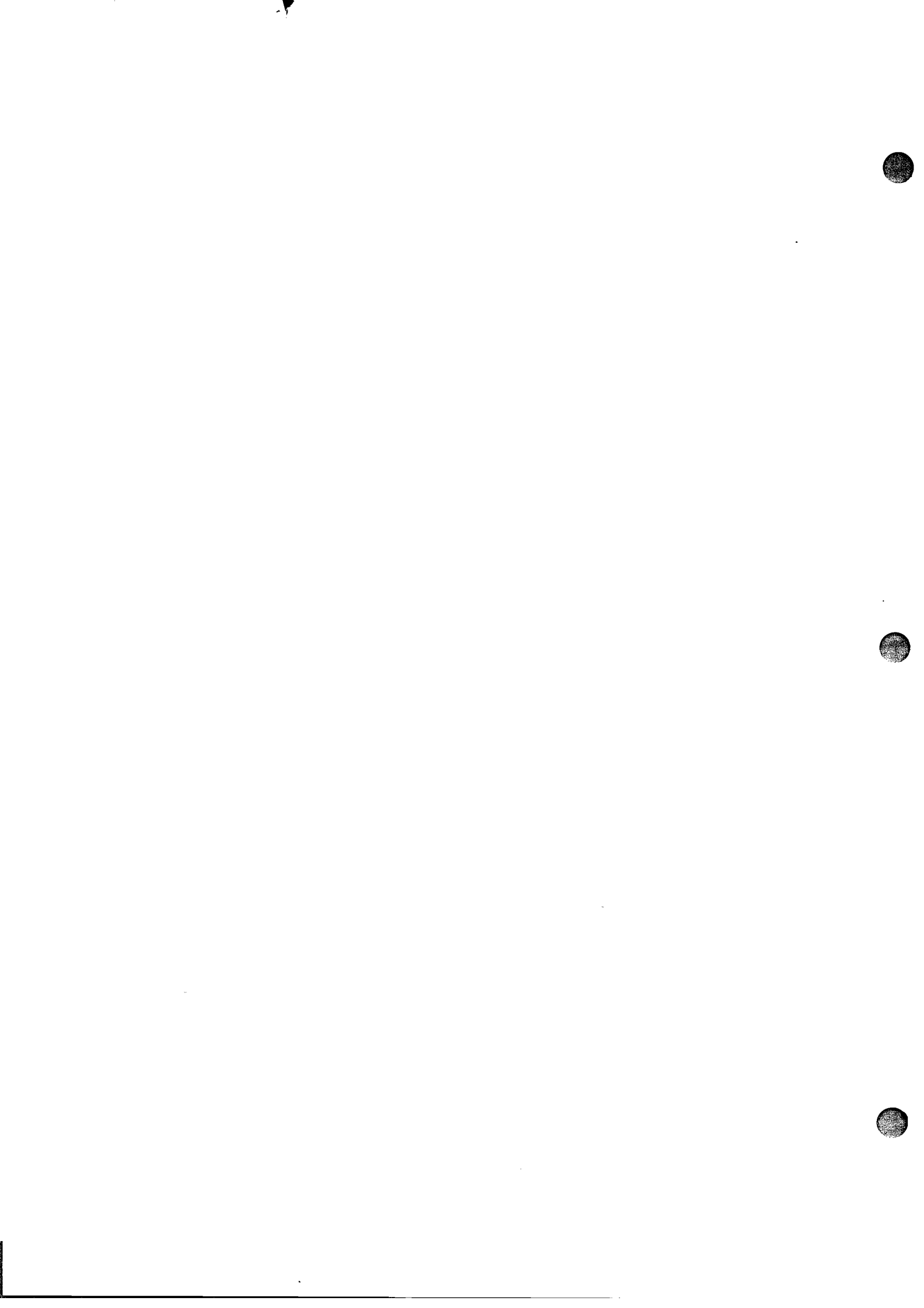
```
I1=1  
I2=19  
I3$='RESIDENTIAL;    LISTING-NR;    = 103'
```

Module Appendix C-1

Reference Material

Reading/Reference Material on Data Base Management

- *TurboIMAGE Data Base Management Reference Manual*
- *QUERY/3000 Reference Manual*
- "An Introduction to Data Base Systems-2nd Edition", C.J. Date, Addison-Wesley, Menlo Park, CA
- "Principles of Data Base Management", James Martin, Prentice Hall Inc., Englewood Cliffs, New Jersey 07632
- "Computer Data Base Organization, 2nd Edition", James Martin, Prentice Hall, Inc., Englewood Cliffs, New Jersey 07632
- "Data Base: Structured Techniques for Design, Performance, and Management", S. Atre, John Wiley & Sons
- "Principles of Database Systems", Jeffrey D. Ullman, Computer Science Press, Inc., Rockville, Maryland 20850
- "The IMAGE/3000 Handbook, Green, Rego, White, Greer, Heidner, WORDWARE, Seattle, Washington



□ BASIC Applications**BASIC Applications**

If an existing IMAGE/3000 application, written in BASIC, accesses and uses the count field in the status array, the application will not run correctly with TurboIMAGE. The reason for this is that TurboIMAGE uses two words in the status array, 5 and 6, for the count of entries in a detail chain. Since BASIC does not have a double integer data type, the double-word count cannot be used correctly without modification to the program.

One solution is to re-code the application so it does not actually use the count. If this is an unacceptable solution, a sample work-around is presented below.

```
10 LONG C
20 INTEGER S(10)
   :
   :
500 REM   The two-word count resides in S(5) and S(6).
501 REM   Each of these quantities is adjusted and then
502 REM   combined into a single LONG variable.

510 C = 0

520 REM   If word 5 is zero, the count is not sufficiently
521 REM   large to use two words. Therefore, S(5) will not
522 REM   affect the count. If word 5 is not zero, its value
523 REM   must be adjusted to represent the correct value.

530 IF S(5) <> 0 THEN C = 65536L0 * S(5)

540 REM   If the high-order bit of word 6 is 1, then BASIC
541 REM   will interpret S(6) as a negative quantity.
542 REM   Therefore, the value must be adjusted to represent
543 REM   the correct positive value. If the high-order bit
544 REM   is 0, then BASIC interprets S(6) as a positive
545 REM   quantity and no adjustment is required.

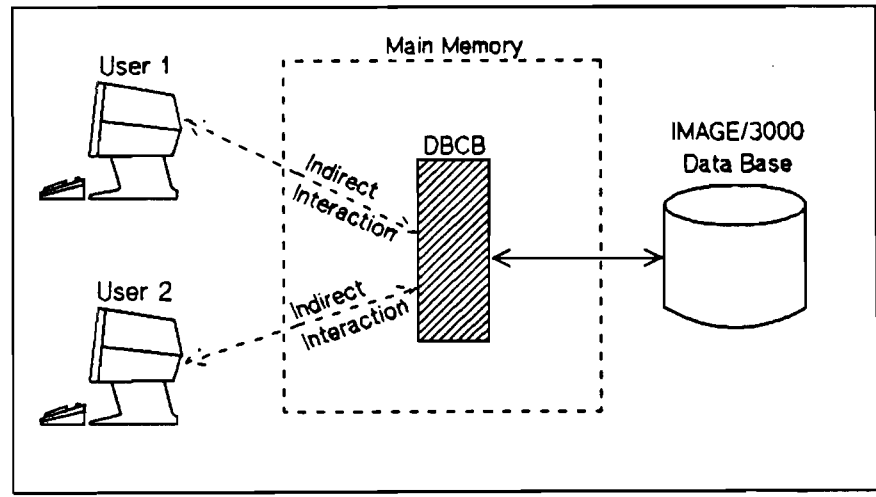
550 IF S(6) < 0 THEN C = C + 65536L0 - S(6)
560 ELSE C = C + S(6)

670 REM   C now contains the correct count of the number of
671 REM   entries in a detail chain.
```

Note: The above does not apply to the count of entries in a synonym chain. TurboIMAGE and IMAGE/3000 both use a one-word count for synonym chains.



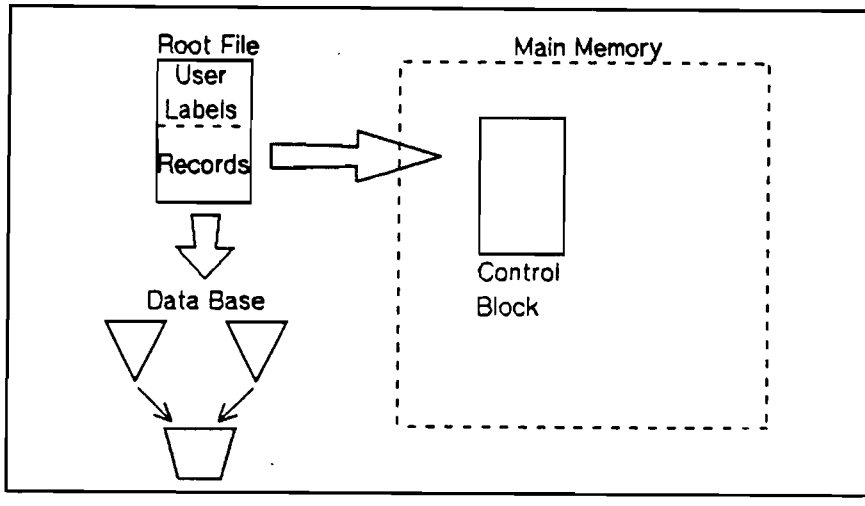
Data Base Control Block (DBC B)



Student Notes:

- Every user must go through the DBC B to access the data base
- One DBC B is created per open data base, regardless of the number of concurrent users.
- The DBC B is created by IMAGE/3000 when the first user opens a data base. It is released when the last user closes a data base.
- The DBC B contains global declarations, root file information, pointers, and buffers.

Root File

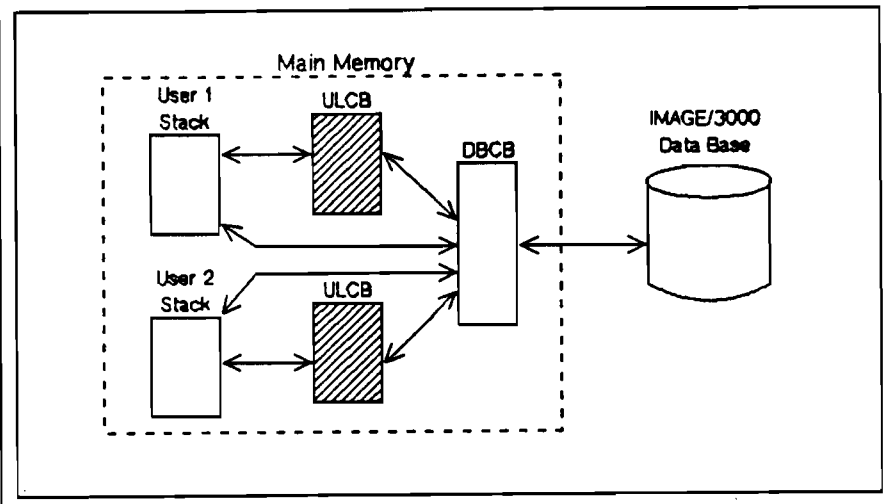


Student Notes:

The root file:

- contains the data base schema in machine-readable form.
- defines the data base structure.
- defines the control block layout.
- contains user defined labels used for security definitions.
- contains records used for item and set definitions.

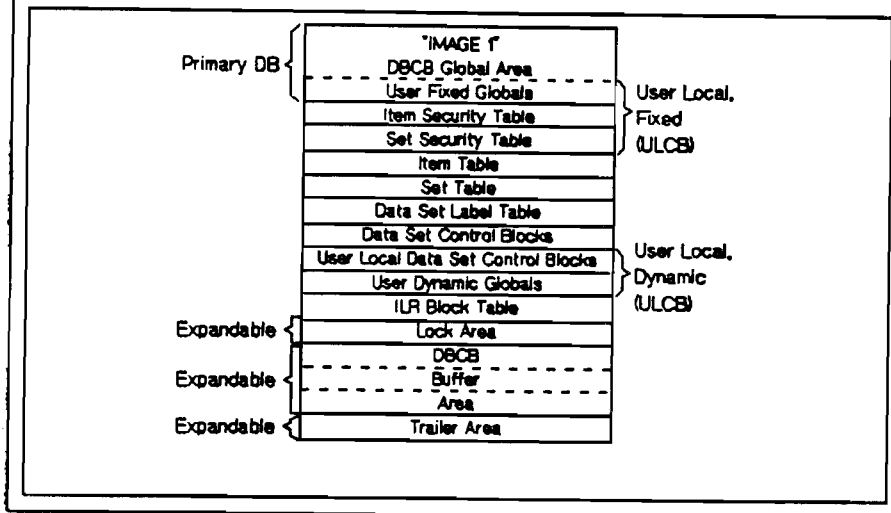
User Local Control Block (ULCB)



Student Notes:

- A new ULCB is created when any process successfully calls DBOPEN. It is released when the process issues a DBCLOSE.
- The ULCB establishes a new access path to the data base.
- The ULCB contains information local to each access path.
- One ULCB is created per user.

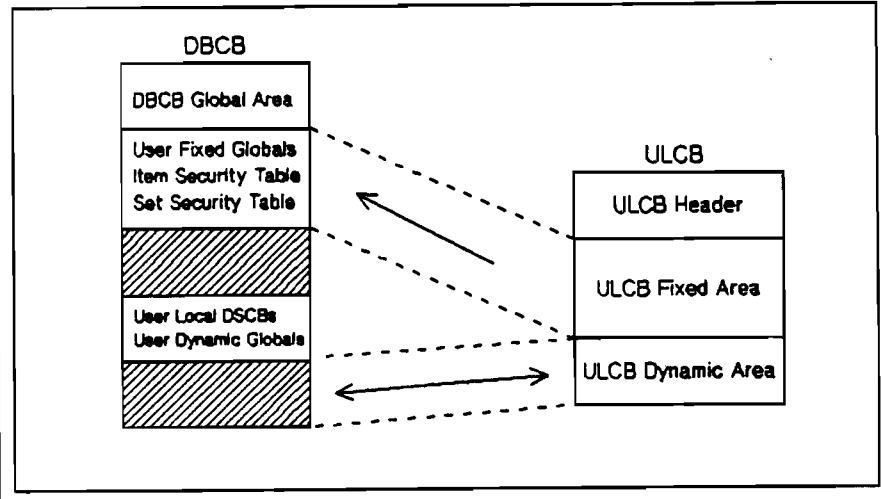
DBCBC Layout



Student Notes:

- **Primary DB:**
 - contains pointers to all portions of the data base.
 - contains global flags set by calls.
- **User Local Fixed:**
 - contains values which differ from user to user but are fixed over time for any one user.
- **User Local Dynamic:**
 - contains values which differ from user to user and differ over time for any one user.
- **Expandable areas:**
 - may dynamically change in size once the data base is open.
- **Size - maximum of 31K words.**

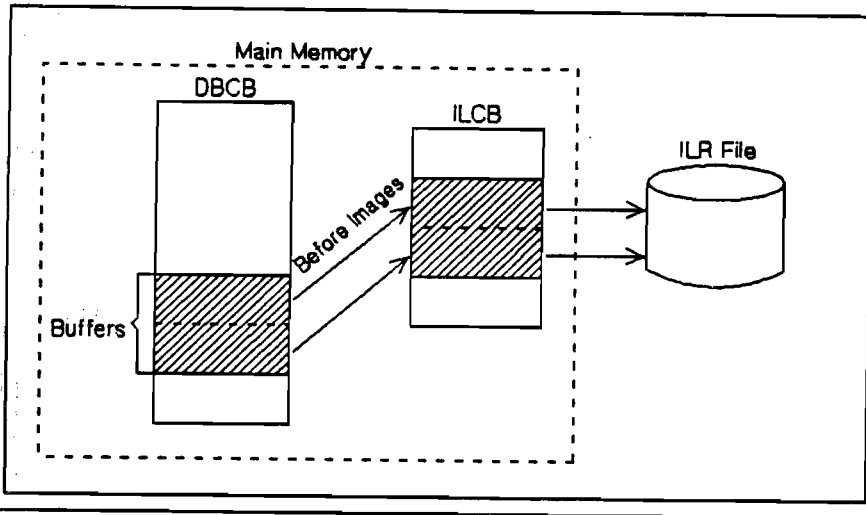
Interaction of DBCB and ULCB



Student Notes:

- ULCB header contains:
 - segment type ID.
 - DST # of ULCB.
 - Base ID (open count, DBCB DST #).

ILR Control Block (ILCB)



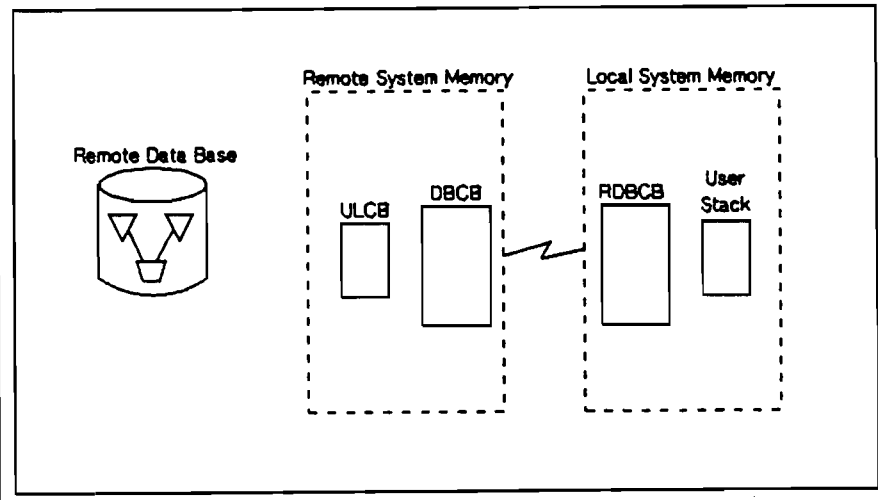
T11 E.08

© 1985 Hewlett-Packard Company

Student Notes:

- The ILCB is the intermediate staging area for buffers to be modified by DBPUT and DBDELETE.
- The ILCB is allocated upon the first DBOPEN of the data base.

Remote Data Base Access Control Block (RDBCBC)



T11 E.07

© 1985 Hewlett-Packard Company

Student Notes:

- One RDBCBC exists per remote DBOPEN.
- The ULCB and DBCB reside on the remote system (where the data base resides).
- Parsing and security checking are done on the local system. The call is then passed to the remote system for processing.

Internal Structure of IMAGE/3000

Notes

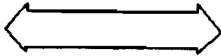
Comparison of IMAGE/3000 and TurboIMAGE Control Blocks

IMAGE/3000

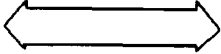
■ DBCB



■ ULCB



■ RDBCBC



TurboIMAGE

■ DBG (Data Base Globals)
■ DBB (Data Base Buffers)

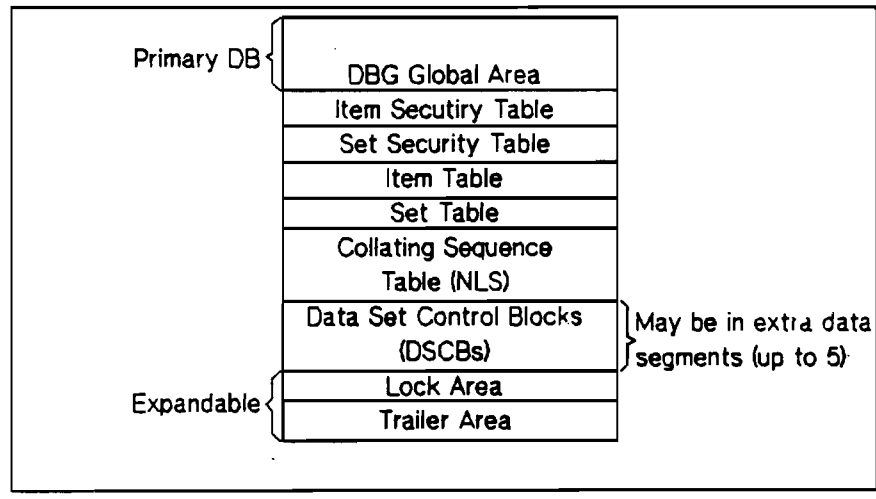
■ DBU (Data Base User)

■ DBR (Data Base Remote)

Student Notes:

- TurboIMAGE uses from 1 to 6 more data segments per open data base than IMAGE/3000. Therefore, a larger DST may have to be configured.

DBG Layout



Student Notes:

- **DBG Global Area contains:**
 - segment type ID.
 - pointers to each area and flags.
 - item/set map tables (used to hash item/set name to locate item/set in item/set table).
 - counts and lengths.
 - DBG wait field (used to lock DBG).
 - User Fixed Globals (used to construct DBU globals during DBOPEN and moved to DBU).
 - file number of root file.
- **Item/Set Tables, Item/Set Security Tables, DSCBs, Lock Area, Trailer contain:**
 - same information as is stored in these tables by IMAGE/3000.
 - DSCBs may be stored in extra data segments
- **Collating Sequence Table:**
 - is used for Native Language Support.
 - contains ASCII sequence according to the Language-ID of the data base in the root file.

DBB Layout

DBB Global Area
Buffer Management Globals
Logging Globals
NLS Collating Sequence Table
Data Set Globals Table
Data Set Access Table
Data Set Label Table
ILR Block Table
Data Buffer Area
Trailer Area

Student Notes:

- DBB Global Area contains:
 - segment type ID.
 - pointers to each area and flags.
 - 4 word wait fields for Global Lock and Busy Flag (used if waiting for individual buffer(s)).
 - counts and lengths.
- Buffer Management Globals contains buffer counts, number of I/Os pending, and tracing information.
- Logging Management Globals contain loggin and ILR flags.
- Data Set Globals Table contains capacity, block length, media record length, blocking factor, field and path information, set type for each data set.
- Data Set Access Table contains:
 - file numbers (wait I/O and nowait I/O).
 - dirty label flag.
 - pointers to buffers pending I/O.
- Data Set Label Table and ILR Block Table are the same as in IMAGE/3000.
- Collating Sequence Table is added to eliminate ExchangeDB to DBG.
- Buffer Area contains:
 - buffer headers and buffers.
 - buffer headers containing 4 word wait fields for individual locks.
 - dirty flag, block number, set number, and release priority.

DBU Layout

DBU Global Area
User Block Data Set Table
Item Security Table (1 word/item)
Set Security Table (1 word/item)
User Local Data Set Table (12 words/set)
Item List Table (1 list/set)
Trailer Area

Student Notes:

- DBU Global Area contains:
 - segment type ID.
 - pointers to each area & flags.
 - DBG, DBB, DST numbers.
 - update modify flags.
 - flags, pointers, counts.
- User Block Data Set Table contains:
 - capacity, block length, media record length, blocking factor, field and path information, set type
 - for each data set.
- Item/Set Security Tables are the same as in IMAGE/3000.
- User Local Data Set Table contains:
 - current record, forward and backward pointers.
 - current path and chain length.
- Item List Table contains the current list array for each data set.
- The Trailer Area is a work area for the DBU.

Internal Structure of TurboIMAGE

Notes

DBR Layout

DBR Global Area
Item Security Table
Set Security Table
Item Table
Set Table
Last List Size Table
Trailer Area

Student Notes:

- The DBR layout is the same as for the IMAGE/3000 RDBC.B.

Module Appendix E-13

Buffer Management

Data Set Buffers

- Data set buffers are used for data transfer to and from disc.
- Each buffer is as large as the largest data block + 10 words.
- The default allocation of buffers is calculated as follows:
 - MAXPATHS = maximum number of paths into any data set + 3.
 - Default allocation = (MAXPATHS + 3) or 8, whichever is greater.
 - 1 additional buffer is allocated for each additional 2 users, up to 20 users.
- Record level locking can cause buffer thrashing.
- The default allocation can be overridden by the DBUTIL >>SET command.
- The number of buffers should be kept constant if there will be a wide range of users. This will eliminate contractions/expansions to the DBB(DBCB) during DBOPEN/DBCLOSE.

Calculating Maximum Number of Buffers

IMAGE/3000

$$MXB = [30400 - 121 - 43s - 2(TFC + TPC)] / (BL + 9)$$

TurboIMAGE

$$MXB = (27570 - 27s - 4MB) / (BL + 13)$$

where:

i - total count of data items
s - number of data sets
TPC - total path count
TFC - total field count
BL - buffer length

MB = maximum number of buffers modified by DBPUT or DBDELETE

$$= 4A + 3M + 1$$

A = number of automatic masters linked to a detail
M = number of manual masters linked to a detail

This formula is calculated for EACH detail set in the data base. MB is equal to the maximum of these values.

Buffer Management

Buffer Management

A new buffer management scheme has been implemented for TurboIMAGE, and is designed to allow greater concurrency amount users, while continuing to ensure data base integrity. As a result of the restructuring of IMAGE/3000, the Data Base Control Block (DBCBC) has been divided into two control blocks, the Data Base Globals Control Block (DBG), and the Data Base Buffer Area Control Block (DBB). There is one DBG and one DBB per data base. TurboIMAGE dynamically allocates the number of buffers within the DBB, depending on the number of users accessing the data base.

The buffer management scheme utilizes a modified Least Recently Used (LRU) replacement algorithm together with an MPE file system capability that allows background I/O requests. Under normal conditions, TurboIMAGE will finish all I/Os before terminating any intrinsic. When operating in AUTODEFER mode, however, dirty buffers are kept within the DBB until space requirements dictate that a flush occur.

The buffer management scheme also implements an internal two-level locking mechanism to control multiple user access with the DBB. To modify a buffer, the process must first obtain a global buffer access lock within the DBB. Once this is held, the process can then issue a request for a buffer access lock. The buffer access lock prevents other users from gaining access to and modifying the buffer. Both the global and buffer locks allow only exclusive access. The buffer management scheme and restructured control blocks increase the number of users in TurboIMAGE, since a process only holds the DBB rather than the entire DBCB (IMAGE/3000).

□ RPG & Locking

RPG & Locking

A variety of user-controlled file locking capabilities have been implemented. This enhancement allows you to perform conditional or unconditional locking and unlocking when you want to on IMAGE, KSAM, or MPE files. For an IMAGE file, you may perform locks at the data base, data set, or record level. For a KSAM and MPE file, you may only lock at the file level.

An unconditional lock is a lock request on a data object such that if the data object is already locked by another process, the lock request will wait in a queue of all lock requests for the data object until it is its turn to lock. All the while the process which requested the unconditional lock is suspended until the lock request is granted. The time the process is suspended may seriously degrade performance in an interactive processing environment. To overcome this, conditional locks should be performed.

A conditional lock is a lock request on a data object such that if the object is already locked by another process, the lock will fail and a resulting indicator will turn on to inform you of the situation. The process is not suspended, so it may now proceed to perhaps process a different data object, returning later when this data object is available.

To perform all locking and unlocking yourself, you use the new operations LOCK and UNLOCK. In the discussion below, every use of LOCK also refers to UNLOCK.

Locking an IMAGE Data Base:

To lock an IMAGE data base, you specify "LOCK" in the calculation operation field, a file name in Factor 2, a data base name in the Result Field, and Result Indicators.

The file name specified in Factor 2 must be a file name given in your File Specifications which describes the IMAGE data base to be locked with K extensions.

The data base name specified in the Result Field must be the data base name given in the KIMAGE specifications for the file in Factor 2. The data base name here in the Result Field is the data base you wish to lock. It must be a literal string, not a variable. THE NUMBER OF CHARACTERS IN THE DATA BASE NAME MUST BE SPECIFIED IN THE RESULT LENGTH FIELD (col. 49-51).

The KIMAGE specifications for the IMAGE data base must also specify "L" (enable locking) as the IMAGE open mode in column 66.

Result Indicators must be specified for this operation. The High Indicator is optional, but one of either the Low or Equal Indicators is required. The presence of the High Indicator specifies conditional locking, whereas its absence specifies unconditional. The Result Indicators return the following status information:

Result Indicator ON Status Information

- High > lock failed - already locked by another process (conditional)
- Low < lock failed - not opened with locking enabled or need MR CAP
- Equal = successful lock

Locking an IMAGE Data Set

To lock an IMAGE data set, you specify "LOCK" in the calculation operation field, a filename in Factor 2, and Result Indicators. The Result Field must be blank.

The file name specified in Factor 2 is the name of the data set to be locked as described in the File Description Specifications. Furthermore the KIMAGE specifications for the data base which contains the data set must specify "L" (enable locking) as the IMAGE open mode in column 66.

Result Indicators must be specified, and their individual purposes are the same as explained above in Locking an IMAGE data base.

Module Appendix F-2

RPG & Locking

Locking an IMAGE Record

To lock an IMAGE record, you specify "LOCK" in the calculation operation field, a search key in Factor 1, a data set file name in Factor 2, and Result Indicators. The Result Field must be blank.

The key specified in Factor 1 is used as the search key for the records which contain that key. All records with items that equal the search key will be locked.

The file name in Factor 2 is the name of the data set you wish to search for records to lock. This file must be described in the File Specification with KIMAGE and KITEM extensions. The KIMAGE specifications must specify "L" as the IMAGE open mode in column 66 to enable locking. KITEM specify the IMAGE item field to be used against the search key.

Result Indicators are required and are the same as explained above in the first section on Locking an IMAGE Data Base.

Locking a KSAM or MPE File

To lock a KSAM file or MPE file, you specify "LOCK" in the calculation operation field, the file name in Factor 2, and Result Indicators.

The file name in Factor 2 must be a name given in the File Description Specifications. If the file is described there as a KSAM file, then this is a KSAM lock operation, else it is an MPE file lock.

The File Specifications for the file to be locked must specify a NOLOCK extension to enable locking. Furthermore, Result Indicators are required and are the same as explained above in the first section on IMAGE data base locking.

Programming Cautions

1. Multiple RIN Capability

If you wish to request more than one outstanding lock, you must have MR special capability. See the IMAGE Reference Manual for a discussion on MR CAP.

2. Dsname'd Files

If you request RPG to perform all locking for you on a cycle by cycle basis, you cannot do your own locking in Calculation Spec.

New Error Messages

ERROR 681T - LOW OR EQUAL RESULT INDICATOR MUST BE SPECIFIED FOR THIS OPERATION

ERROR 682T - IMAGE FILE LOCKING MODE IS NOT L TO ENABLE LOCKING ONLY

ERROR 683T - FILE NOT DESCRIBED WITH NOLOCK TO ENABLE LOCKING

ERROR 684T - DB NAME IN RESULT FIELD MUST BE SAME NAME SPECIFIED IN IMAGE SPEC FOR FILE IN FACTOR 2

Summary Table of LOCK/UNLCK

Type of Lock	Factor 1	Oper	Factor 2	Result Fld
IMAGE Record	key	LOCK	filename	blank
IMAGE DataSet	blank	LOCK	filename	blank
IMAGE DataBase	blank	LOCK	filename	data base
KSAM file	blank	LOCK	filename	blank
MPE file	blank	LOCK	filename	blank

Result Indicator ON Status Information

High	>	lock failed - already locked by another process (conditional)
Low	<	lock failed - not opened with locking enabled or need MR CAP
Equal	=	successful lock

EXAMPLES

FILE DESCRIPTION SPECIFICATIONS

```

FINFILE      IP F      80      DISC
FKSDATA1    IC F      72R14AI  51 DISC
F
F
KIMAGE EMP1 L5
KITEM CITY
    
```

CALCULATION SPECIFICATIONS

```

** IMAGE DATA BASE LEVEL LOCKING
**
C          LOCK KSDATA1 EMP1 4 737475
C 75      CHAINKSDATA1 8058
C 75      UNLCKKSDATA1 EMP1 4 76
    
```

Source:

HP 3000Communicator, MPE III 2028, Issue number 25, 7/80
 Page 29, #5. "Enhancements to RPG/3000"

[Redacted]

[Redacted]

Initial
Initial

[Faint, illegible text]

[Faint, illegible text]

[Faint, illegible text]

[Faint, illegible text]

[Faint, illegible text]

[Faint, illegible text]

Module Appendix G-1

□ Compile, Prepare, and Execute Commands

BASIC

```
:BASIC [commandfile] [, [inputfile] [, listfile]]
:BASICOMP [commandfile] [, [uslfile] [, listfile]]
:BASICPREP [commandfile] [, [progfile] [, listfile]]
:BASICGO [commandfile] [, listfile]
```

COBOL II/3000

```
:COBOLII [textfile] [, [uslfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring] [;WKSP=wksfilename]
:COBOLPREP [textfile] [, [progfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:COBOLGO [textfile] [, [listfile] [, [masterfile] [, newfile]]]
    [;INFO=quotedstring]
```

FORTRAN

```
:FORTRAN [textfile] [, [uslfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:FORTPREP [textfile] [, [progfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:FORTGO [textfile] [, [listfile] [, [masterfile] [, newfile]]]
    [;INFO=quotedstring]
```

FORTRAN/77

```
:FTN [textfile] [, [uslfile] [, [listfile]]] [;INFO="text"]
:FTNPREP [textfile] [, [progfile] [, [listfile]]] [;INFO="text"]
:FTNGO [textfile] [, [listfile]] [;INFO="text"]
```

Pascal/3000

```
:PASCAL [textfile] [, [uslfile] [, [listfile]]] [;INFO=quotedstring]
:PASCALPREP [textfile] [, [progfile] [, [listfile]]] [;INFO=quotedstring]
:PASCALGO [textfile] [, [listfile]] [;INFO=quotedstring]
```

RPG/3000

```
:RPG [textfile] [, [uslfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:RPGPREP [textfile] [, [progfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:RPGGO [textfile] [, [listfile] [, [masterfile] [, newfile]]] [;INFO=quotedstring]
```

SPL

```
:SPL [textfile] [, [uslfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:SPLPREP [textfile] [, [progfile] [, [listfile] [, [masterfile] [, newfile]]]]
    [;INFO=quotedstring]
:SPLGO [textfile] [, [listfile] [, [masterfile] [, newfile]]] [;INFO=quotedstring]
```

TRANSACTION/3000

```
:RUN TRANCOMP.PUB.SYS
:RUN TRANSACT.PUB.SYS
```

Note: Check with your system manager to see if language UDCs have been created on your system.

Module Appendix G-2

Compile, Prepare, and Execute Commands

Where

Commandfile	BASIC subsystem input; original source of all commands and statements of BASIC program.
Inputfile	BASIC program input; contains data input to BASIC through INPUT, ENTER, and LINPUT statements.
Listfile	Actual designator of file on which program listing is written.
Masterfile	Actual designator of master file which is merged against textfile to produce composite source.
Newfile	Actual designator of merged textfile and masterfile.
Progfile	Actual designator of program file on which prepared program segments are written.
Quotedstring	A string of ASCII characters delimited by a matching set of 's or "s. This feature is used for specifying compiler options, which will appear in front of the source listing.
Text	A specification of initial compiler options.
Textfile	Actual designator of input file from which source program is read.
Usrfile	Actual designator of user subprogram library (USL) file on which object program is written.
Wkspfilename	Actual designator of the file to be used as an HPToolset workspace.

Language Reference Manuals

BASIC Compiler Reference Manual (32103-90001)	01-3,3-2-4-1
BASIC Interpreter Manual (30000-90026)	01-3,3-2-4-2
COBOLII/3000 Reference Manual (32233-90001)	01-3,3-2-4-3
FORTRAN Reference Manual (30000-90040)	01-3,3-2-4-4
FORTRAN/77 Reference Manual (5957-4685)	01-3,3-2-4-5
Pascal/3000 Reference Manual (32106-90001)	01-3,3-2-4-6
RPG/3000 Reference Manual (32104-90001)	01-3,3-2-4-7
SPL Reference Manual (30000-90024)	01-3,3-2-4-8
TRANSACTION/3000 Reference Manual (32247-90001)	01-3,3-2-4-9

Compile, Prepare, and Execute Commands

- access 1-12, 1-13, 5-2, 5-5, 5-18, 6-2, 6-7, 7-1
 - calculated 5-2
 - chained 5-2, 5-4
 - concurrent 6-2, 6-4, 6-5, 6-13
 - direct 1-12, 5-2
 - random 1-12
 - sequential by key 1-13
 - serial 1-12, 5-2
- ACTIVATE (DBUTIL) 8-3, B-2, B-3
- Adager/3000 10-10
- ADD (QUERY) 2-15
- address calculation 2-3, 5-2
- argument parameter 5-21, 5-22
- automatic master 2-3, 2-4

- back-up 8-5, 8-14, 8-18
- backward pointer 2-6, 5-4
- base parameter 5-9, 5-13, 5-14, 5-16, 5-17, 5-21, 5-25, 6-7, 6-9
- BASIC 5-8, 5-14, D-1
- batch 1-4, 1-12
- BIMAGE procedures 5-14
- bit map 2-7
- block 2-7
- buffer management 8-4, 8-13, E-13
- buffer parameter 5-14 to 5-16, 5-21, 5-22, 5-25

- calculated access 5-2
- capacity 4-8, 4-9
- chain 2-5, 5-4
- chain head 2-6, 5-2 to 5-4
- chained access 5-2, 5-4
- COBOL 5-8, 5-13
- components of TurboIMAGE 2-10
- concurrent access 6-2, 6-4, 6-5, 6-13
- condition code 5-10, 5-13, 5-14, 5-17, 6-7, 6-9
- conditional locking 6-7
- CONTROL (DBRECOV) 8-18, 8-19, B-4
- count 2-5, 2-6, 5-2, 5-3
- CREATE (DBUTIL) 4-19, 8-3, B-2, B-3

- data base 1-3 to 1-5, 2-2
- data base application 7-2 to 7-4
- data base creation 3-4, 4-2, 4-11, 4-18, 4-21
- data base models 1-6, 1-11, A-1
 - hierarchical 1-6, 1-7
 - network 1-6, 1-8
 - relational 1-6, 1-7
 - TurboIMAGE 1-8
- data base restructuring 8-10 to 8-12
- data entry 2-2
- data item 2-2
- data set 2-2, 2-3
 - automatic master 2-3, 2-4
 - detail 2-3, 2-5 to 2-7
 - layout 2-7
 - manual master 2-3
 - master 2-3, 2-4, 2-6, 2-7, 7-5
- DBA (Data Base Administrator) 1-6, 3-4
- DBB (Data Base Buffer control block) E-8, E-10
- DBBEGIN 5-8, 8-14, 8-15
- DBC (Data Base Control Block) E-1, E-4, E-5

Index

DBCLOSE 5-8,5-17,8-14,8-15,9-15
DBCONV 8-2,8-8,B-4
DBDELETE 5-8,5-25,8-14,8-15,8-20
DBEND 5-8,8-14,8-15,8-20
DBERROR 5-8,5-15
DBEXPLAIN 5-8,5-15
DBFIND 5-8,5-21
DBG (Data Base Globals control block) E-8,E-9
DBGGET 5-8,5-21,5-22,7-5,8-20,9-15
DBINFO 5-8,5-14,8-15
DBLOAD 8-2,8-7,8-12,B-4
DBLOCK 5-8,6-5,6-7,6-10
DBMEMO 5-8,8-15
DBMS(Data Base Management System) 1-2,1-5,1-12
DBOPEN 5-8,5-13,8-15,8-20,9-15
DBOPEN access types 6-2
DBOPEN modes 6-3,6-12,6-17
DBPUT 5-8,5-16,8-14,8-15,8-20,9-15
DBR (Data Base Remote control block) E-8,E-12
DBRECOV 8-2,8-18,B-4
- CONTROL command 8-18,8-19
- EXIT command 8-18
- FILE command 8-18
- PRINT command 8-18
- RECOVER command 8-18
- ROLLBACK command 8-18,8-21
- RUN command 8-18
DBRESTOR 8-2,8-5,8-12,B-4
DBSCHEMA 4-2,4-14,4-15,4-17,8-12
DBSTORE 8-2,8-5,8-12,8-20,B-4
DBU (Data Base User control block) E-8,E-11
DBUNLOAD 8-2,8-6,8-12,B-4
DBUNLOCK 5-8,6-5,6-9,6-10
DBUPDATE 5-8,5-25,8-14,8-15,9-15
DBUTIL 4-2,8-2 to 8-4,8-12,B-2
- ACTIVATE command 8-3,B-2,B-3
- CREATE command 4-19,8-3,8-12,B-2,B-3
- DEACTIVATE command 8-3,B-2,B-3
- DISABLE command 8-3,8-4,B-2,B-3
- ENABLE command 8-3,8-4,B-2,B-3
- ERASE command 4-20,8-3,8-12,B-2,B-3
- EXIT command 4-19,8-3,B-2,B-3
- HELP command 4-19,8-3,B-2,B-3
- MOVE command 8-3,B-2,B-3
- PURGE command 4-20,8-3,8-12,B-2,B-3
- RELEASE command 8-3,B-2,B-3
- SECURE command 8-3,B-2,B-3
- SET command 8-3,8-4,B-2,B-3
- SHOW command 8-3,B-2,B-3
- VERIFY command 8-3,B-2,B-3
DDL(Data Definition Language) 2-10,3-3,4-2
DEACTIVATE (DBUTIL) 8-3,B-2,B-3
DEFINE (QUERY) 2-13
DELETE (QUERY) 2-16
design 3-4,3-5,8-10,8-11,9-3 to 9-5,9-17 to 9-28
- considerations 9-3,9-4,9-6,9-7
- decisions 9-3,9-8 to 9-11
- team 9-3
- trade-offs 9-3,9-5
- top-down 9-6 to 9-8
- types of 9-4
detail data set 2-3,2-5,4-9
device class 4-8,4-9

5-8,5-17,8-14,8-15,9-15
8-2,8-8,B-4
5-8,5-25,8-14,8-15,8-20
5-8,8-14,8-15,8-20
5-8,5-15
5-8,5-15
5-8,5-21
E-8,E-9
5-8,5-21,5-22,7-5,8-20,9-15
5-8,5-14,8-15
8-2,8-7,8-12,B-4
5-8,6-5,6-7,6-10
5-8,8-15
1-2,1-5,1-12
5-8,5-13,8-15,8-20,9-15
6-2
6-3,6-12,6-17
5-8,5-16,8-14,8-15,8-20,9-15
E-8,E-12
8-2,8-18,B-4
8-18,8-19
8-18
8-18
8-18
8-18
8-18,8-21
8-18
8-2,8-5,8-12,B-4
4-2,4-14,4-15,4-17,8-12
8-2,8-5,8-12,8-20,B-4
E-8,E-11
8-2,8-6,8-12,B-4
5-8,6-5,6-9,6-10
5-8,5-25,8-14,8-15,9-15
4-2,8-2 to 8-4,8-12,B-2
8-3,B-2,B-3
4-19,8-3,8-12,B-2,B-3
8-3,B-2,B-3
8-3,8-4,B-2,B-3
8-3,8-4,B-2,B-3
4-20,8-3,8-12,B-2,B-3
4-19,8-3,B-2,B-3
4-19,8-3,B-2,B-3
8-3,B-2,B-3
4-20,8-3,8-12,B-2,B-3
8-3,B-2,B-3
8-3,B-2,B-3
8-3,8-4,B-2,B-3
8-3,B-2,B-3
8-3,B-2,B-3
2-10,3-3,4-2
8-3,B-2,B-3
2-13
2-16
3-4,3-5,8-10,8-11,9-3 to 9-5,9-17 to 9-28
9-3,9-4,9-6,9-7
9-3,9-8 to 9-11
9-3
9-3,9-5
9-6 to 9-8
9-4
2-3,2-5,4-9
4-8,4-9
E-8,E-12
8-2,8-18,B-4
8-18,8-19
8-18
8-18
8-18
8-18
8-18,8-21
8-18
8-2,8-5,8-12,B-4
4-2,4-14,4-15,4-17,8-12
8-2,8-5,8-12,8-20,B-4
E-8,E-11
8-2,8-6,8-12,B-4
5-8,6-5,6-9,6-10
5-8,5-25,8-14,8-15,9-15
4-2,8-2 to 8-4,8-12,B-2
8-3,B-2,B-3
4-19,8-3,8-12,B-2,B-3
8-3,B-2,B-3
8-3,8-4,B-2,B-3
8-3,8-4,B-2,B-3
4-20,8-3,8-12,B-2,B-3
4-19,8-3,B-2,B-3
4-19,8-3,B-2,B-3
8-3,B-2,B-3
4-20,8-3,8-12,B-2,B-3
8-3,B-2,B-3
8-3,B-2,B-3
8-3,8-4,B-2,B-3
8-3,B-2,B-3
8-3,B-2,B-3
2-10,3-3,4-2
8-3,B-2,B-3
2-13
2-16
3-4,3-5,8-10,8-11,9-3 to 9-5,9-17 to 9-28
9-3,9-4,9-6,9-7
9-3,9-8 to 9-11
9-3
9-3,9-5
9-6 to 9-8
9-4
2-3,2-5,4-9
4-8,4-9

Dictionary/3000 10-6,10-7
 direct access 1-12,5-2
 DISABLE (DBUTIL) 8-3,8-4,B-2,B-3
 DML(Data Manipulation Language) 2-10,3-3,5-8
 dset parameter 5-9,5-16,5-17,5-21,5-25,6-9
 dummy parameter 5-9,5-17

ENABLE (DBUTIL) 8-3,8-4,B-2,B-3
 - ACCESS option 8-4
 - AUTODEFER option 8-4
 - DUMPING option 8-4
 - ILR option 8-4
 - LOGGING option 8-4
 - RECOVERY option 8-4
 - ROLLBACK option 8-4

ERASE (DBUTIL) 4-20,8-3,B-2,B-3
 EXIT (DBRECOV) 8-18,B-4
 EXIT (DBUTIL) 4-19,8-3,B-2,B-3
 EXIT (QUERY) 2-11

FILE (DBRECOV) 8-18,B-4
 FIND (QUERY) 2-14
 FORM (QUERY) 2-13
 FORTRAN 5-8,5-13
 forward pointer 2-6,5-4

hashing 2-3,5-2
 HELP (DBUTIL) 4-19,8-3,B-2,B-3
 HELP (QUERY) 2-11
 hierarchical model 1-6,1-7

ILCB (ILR Control Block) E-6
 ILR (Intrinsic Level Recovery) 8-4,8-20,8-21
 IMAGE/3000 control blocks E-1,E-2,E-4 to E-8
 - DBCB E-1,E-4,E-5,E-8
 - ILCB E-6
 - ULCB E-2,E-5,E-8
 - RDBC B E-7,E-8
 implementation 3-4,9-2
 Inform/3000 10-6,10-9
 interactive 1-4,1-12
 item name 4-4,4-5,4-8,4-9,5-21

key 2-3,2-6

LIST (QUERY) 2-14
 list parameter 5-16,5-21,5-25
 lock descriptor array 6-8,B-5
 locking 6-5,6-6,6-8 to 6-10,7-2
 - conditional 6-7
 - data base 6-7
 - data entry 6-7,6-9
 - data set 6-7
 - strategy 6-11
 - unconditional 6-7
 log file 8-13,8-18,8-21
 logging 2-10,8-4,8-13 to 8-17,8-23
 logical transaction 6-10,8-14
 LOGID 8-4,8-13

DBRECOV 8-2,8-18,B-4
 - CONTROL command 8-18
 - EXIT command 8-18
 - FILE command 8-18
 - PRINT command 8-18
 - RECOVER command 8-18
 - ROLLBACK command 8-18
 - RUN command 8-18
 DBRECOV 8-2,8-18,B-4
 DBSCHEMA 4-2,4-14,4-15,4-17,4-18
 DBSTORE 8-2,8-12,8-13,8-14
 DBU (Data Base User Control Block) 8-2,8-11
 DBUNLOAD 8-2,8-12,8-13,8-14
 DBUNLOCK 8-2,8-12,8-13,8-14
 DBUPDATE 8-2,8-12,8-13,8-14
 DBUTIL 4-2,8-3 to 8-4,8-12,8-13
 - ACTIVATE command 8-3,8-12,8-13
 - CREATE command 4-19,8-3,8-12,8-13
 - DEACTIVATE command 8-3,8-12,8-13
 - DISABLE command 8-3,8-4,8-12,8-13
 - ENABLE command 8-3,8-4,8-12,8-13
 - ERASE command 4-20,8-3,8-12,8-13
 - EXIT command 4-19,8-3,8-12,8-13
 - HELP command 4-19,8-3,8-12,8-13
 - MOVE command 8-3,8-12,8-13
 - PURGE command 4-20,8-3,8-12,8-13
 - RELEASE command 8-3,8-12,8-13
 - SECURE command 8-3,8-12,8-13
 - SET command 8-3,8-4,8-12,8-13
 - SHOW command 8-3,8-12,8-13
 - VERIFY command 8-3,8-12,8-13
 DDL(Data Definition Language) 2-10,3-3,5-8
 DEACTIVATE (DBUTIL) 8-3,8-12,8-13
 DEFINE (QUERY) 2-14
 DELETE (QUERY) 2-14
 design 3-4,8-12,8-13,8-14,8-15,8-16,8-17,8-18,8-19
 - considerations 8-12,8-13,8-14
 - decisions 8-12,8-13,8-14
 - team 8-13
 - trade-offs 8-13,8-14
 - top-down 8-12 to 8-13
 - types of 8-14
 detail data set 2-3,2-4,2-5
 device class 4-8,4-9

Index

manual master 2-3,2-4
master data set 2-3,2-4,4-8,7-5
media record 2-7
migrating secondaries 5-3
migration 1-2,8-8
mode 5-9,5-13,5-14,5-16,5-17,5-21,5-25,6-7,6-9
modify access 6-2,6-3,6-5
MOVE (DBUTIL) 8-3,8-2,8-3

network model 1-6,1-8

OUTPUT (QUERY) 2-17

parameters 5-9 to 5-11
Pascal 5-8,5-13
password 5-13,8-4
path 2-3
path count 4-8
performance 9-12 to 9-16,9-18 to 9-28
pointer 2-5,2-6
primary entry 5-3
primary path 4-9,8-1
PRINT (DBRECOV) 8-18,8-4
Profiler, TurboIMAGE 10-5
PURGE (DBUTIL) 4-20,8-3,8-2,8-3

qualifier parameter 5-14,6-7
QUERY/3000 2-10 to 2-12,2-18,3-3,3-4
- ADD command 2-15
- DEFINE command 2-13
- DELETE command 2-16
- EXIT command 2-11
- FIND command 2-14
- FORM command 2-13,4-20
- HELP command 2-11
- LIST command 2-14
- OUTPUT command 2-17
- REPLACE command 2-16
- REPORT command 2-14
- special characters 2-12

random access 1-12
Rapid/3000 10-6
RDBC (Remote Data Base Control Block) E-7
read access 6-2,6-3,6-5
read class list 4-4,4-6,4-8,4-9,6-15,6-16
RECOVER (DBRECOV) 8-18,8-20,8-4
recovery 8-4,8-18,8-20,8-22,8-23
- roll-back 8-21
- roll-forward 8-20
relational model 1-6,1-7
relationship between IMAGE/3000 and TurboIMAGE 1-2,1-3,4-10,8-8,E-8,E-13
relative record number 2-5
RELEASE (DBUTIL) 8-3,8-2,8-3
remote data base access 10-2,10-3
REPLACE (QUERY) 2-16
REPORT (QUERY) 2-14

- Report/3000 10-6,10-8
- ROLLBACK (DBRECOV) 8-4,8-18,8-21,B-4
- root file 4-2,4-14,4-19,8-5,E-2
- RPG 5-8,5-13,5-16,5-19,5-22,5-24,5-27,F-1 to F-3
 - adding entries 5-16
 - locking 6-8,6-14,F-1 to F-3
 - opening a data base 5-13
 - retrieving entries 5-22
- RUN (DBRECOV) 8-18,B-4

- schema 4-2,4-14,B-1
 - items 4-2,4-4
 - name 4-2,4-3
 - passwords 4-2,4-3
 - sets 4-2,4-8
- schema processor 4-2,4-15
 - \$CONTROL command 4-16
 - \$TITLE command 4-16
 - \$PAGE command 4-16
- search item 2-3,2-5,5-2
- secondary entry 5-3
- SECURE (DBUTIL) 8-3,8-2,B-3
- security 6-15,6-16,5-18,6-19,7-2
- sequential by key 1-13
- serial access 1-12,5-2
- SET (DBUTIL) 8-3,8-4,B-2,B-3
 - BUFFSPECS 8-4
 - LOGID 8-4
 - MAINT 8-4
 - PASSWORD 8-4
- set name 4-8,4-9
- SHOW (DBUTIL) 8-3,B-2,B-3
- sort item 2-5,4-9
- SPL 5-8,5-13
- status parameter 5-9,5-10,5-13 to 5-17,5-21,5-25,6-7,6-9
- sub-item length 4-4 to 4-6
- synonym 2-6,5-2,5-4
- Transact/3000 10-6,10-7
- TurboIMAGE control blocks E-8 to E-12
 - DBB E-8,E-10
 - DBG E-8,E-9
 - DBR E-8,E-12
 - DBU E-8,E-11
- TurboIMAGE limits 4-10
- TurboIMAGE overview 2-10,8-25
- TurboIMAGE parameters 5-9,5-10
- TurboIMAGE procedures 5-8,7-2
- TurboIMAGE Profiler 10-5
- type designator 4-4,4-5,4-7

- ULCB (User Local Control Block) E-3,E-5
- unconditional locking 6-7
- update access 6-2,6-3,6-5
- user class numbers 4-3,5,13,6-15
- User Interface Program (TurboIMAGE Profiler) 10-5
- utilities 2-10,3-3,8-2,8-9,B-3,B-4

- VERIFY (DBUTIL) 8-3,B-2,B-3

- Wonder Realty 1-9,1-10,2-8,2-9,A-3,A-4,A-6,A-10
- write class list 4-4,4-6,4-8,4-9,6-15,6-16
- XDBOPEN 5-14

LOCATION _____ DATE _____

QUESTION

What did you learn from this course prior to this? _____
What are the main points of the course? _____
What are the main points of the course? _____

What are the main points of the course? _____
What are the main points of the course? _____

What are the main points of the course? _____
What are the main points of the course? _____

What are the main points of the course? _____
What are the main points of the course? _____

EXCELLENT GOOD FAIR POOR

_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____

What are the main points of the course? _____

Instructor Review Sheet

COURSE NAME _____ LOCATION _____ DATE _____

COURSE ADMINISTRATION:

1. How many times had you taught this course prior to this? _____
2. Were the course materials (slides, workbooks, etc.) prepared by you or somebody else? _____
3. How many students did you have in the class? _____
4. Was the setup procedure clear? _____ How could it be simplified?
Improved? _____

5. Did the course content meet the needs of the audience? If not, what did you have to add, supplement, delete or modify?

6. How many hours did you spend in preparation to teach? _____
7. Could the prep time be reduced? _____
How? _____

OVERHEAD SLIDES:

1. Were any of the slides in error? _____
If yes, approximately how many? _____
2. Were the slides plotted in color? _____
If NO, how were they prepared? _____
3. Rate the overall effectiveness of the slides.

QUALITY	EXCELLENT	GOOD	FAIR	POOR
a. Colors	_____	_____	_____	_____
b. Clarity	_____	_____	_____	_____
c. Logical Structure	_____	_____	_____	_____
d. Content accuracy	_____	_____	_____	_____
e. Aesthetic appearance	_____	_____	_____	_____
4. How would you improve the slides?

Instructor Review Sheet (cont'd)

Instructor Review Sheet (cont'd)

233

STUDENT WORKBOOK:

1. Were the student workbooks printed on the HP2680 Laser Printer System?
If NO, how were they printed?
2. Were the workbooks bound with special Customer Training binder covers?
If NO, how were they bound?
3. Rate the overall effectiveness of the student workbook?

QUALITY	EXCELLENT	GOOD	FAIR	POOR
a. Ease of use	<u> </u>	<u> </u>	<u> </u>	<u> </u>
b. Content accuracy	<u> </u>	<u> </u>	<u> </u>	<u> </u>
c. Completeness	<u> </u>	<u> </u>	<u> </u>	<u> </u>
d. Logical structure	<u> </u>	<u> </u>	<u> </u>	<u> </u>
4. How would you improve the workbook?

INSTRUCTOR NOTES:

1. How did you use your instructor notes while teaching (choose one)
 - a. Interleaved the instructor notes with a student workbook.
 - b. Used the instructor notes separately from the workbook.
 - c. Transferred the information from the instructor notes to the student workbook and taught from the latter.
 - d. Other: Please explain.
2. Rate the overall effectiveness of the instructor notes

QUALITY	EXCELLENT	GOOD	FAIR	POOR
a. Clarity	<u> </u>	<u> </u>	<u> </u>	<u> </u>
b. Technical accuracy	<u> </u>	<u> </u>	<u> </u>	<u> </u>
c. Logical structure	<u> </u>	<u> </u>	<u> </u>	<u> </u>
d. Teaching techniques	<u> </u>	<u> </u>	<u> </u>	<u> </u>
e. Completeness	<u> </u>	<u> </u>	<u> </u>	<u> </u>
3. Did you add any notes? If so, please attach copies of the notes you added.
4. Did the notes follow the material in a logical and coordinated fashion?

STUDENT WORKBOOK

LAB EXERCISES:

1. Did you personally test each of the labs prior to the class? _____
 If not, did someone else at your location? _____
2. Rate the overall effectiveness of the lab instructions.
 QUALITY EXCELLENT GOOD FAIR POOR
 a. Clarity _____ _____ _____ _____
 b. Technical accuracy _____ _____ _____ _____
 c. Logical structure _____ _____ _____ _____
 d. Completeness _____ _____ _____ _____
3. Rate the overall effectiveness of the lab exercises.
 QUALITY EXCELLENT GOOD FAIR POOR
 a. Met the teaching objectives _____ _____ _____ _____
 b. Challenged the students _____ _____ _____ _____
 c. Lab Documentation _____ _____ _____ _____
 d. Solutions worked as provided _____ _____ _____ _____
4. Did the material presented in class provide a firm basis for completing the lab exercises? _____
5. What additional labs did you provide? _____
6. Did the solutions work as provided? _____

ADDITIONAL COMMENTS OR SUGGESTIONS:

Lab 2 - Should include file creation ~~REALITY~~ REALITY-LABS
 Worksession 6-6 (Security) - Error in instructor Exercise # 2 notes - Key + item revised.
 2:08 slide PATHØ; 2:13 Date Base; 88.1 LABb Prog (color) typo

MAIL TO:

Please send this Review Sheet along with copies of slides, notes, workbook pages, etc., marked with corrections or comments to:

John Beach
 SE Customer Training
 Hewlett-Packard Company
 19320 Pruneridge Avenue
 Cupertino, CA. 95014

INSTRUCTOR _____ DATE _____

