Teradata

Concepts and Facilities

C02-0001-04

Release 3.0

DBC/1012 Data Base Computer

RECEIVED JAN 1 3 1988

Concepts and Facilities

C02-0001-04

Release 3.0

Copyright, 1987, Teradata Corporation

All rights reserved. No part of this document may be copied or reproduced in any form or by any means without the prior written consent of Teradata. Teradata Corporation, 12945 Jefferson Boulevard, Los Angeles, CA 90066.

EFFECTIVE PAGES

This is the -04 version of the DBC/1012® Data Base Computer Concepts and Facilities. Revision levels of individual pages are given below.

Page	Revision
Title page	-04
Effective pages (iii)	-04
Preface (v through viii)	-04
Contents (ix through xii)	-04
1-1 through 1- 9	-04
2-1 through 2-13	-04
3-1 through 3-29	-04
4-1 through 4-13	-04
5-1 through 5-12	-04
6-1 through 6- 4	-04
Glossary G-1 through G-11	-04
Index I-1 through I-5	-04

DBC/1012 is a registered trademark of Teradata Corporation.

PREFACE

This preface describes the structure of the DBC/1012 Data Base Computer Concepts and Facilities, and tells you what you should know before you begin to read this manual.

This document is one of several that describe the DBC/1012 Data Base Computer. The complete set includes:

- DBC/1012 Data Base Computer Concepts and Facilities (document number C02-0001)

  The concepts document is written for senior executives, managers, and technical personnel. The document presents an overview of the DBC/1012 Data Base Computer System, addressing such topics as architecture, user facilities, system facilities, hardware and software structure, operating characteristics, and configuration specifications.

- DBC/1012 Data Base Computer User's Guide (document number C09-0001)

  The user's guide is written for the non-DP user. The guide presents a basic introduction to ITEQ™, addressing such topics as ITEQ and BTEQ, on-line edits, queries, print formats, and table creation and modification. It also explains macros, privileges, the use of DBC/SQL statements in a COBOL program, and the Data Dictionary/Directory.

- DBC/1012 Data Base Computer Primer (document number C09-0002)

  The Primer is written for new computer users. It teaches basic DBC/1012 query statements, offering hands-on examples for users to try at their own pace.

- DBC/1012 ITEQ Keypad Template (document number C99-0002)

  The template, which fits over the PF-key keypad of the standard 3270 terminal keyboard, shows the assignment of PF keys to ITEQ commands.

- DBC/1012 Data Base Computer Reference Manual (document number C03-0001)

  The reference manual is written for technical personnel. The manual presents the details of language syntax, DBC/SQL statements, ITEQ commands, BTEQ commands, and the Data Dictionary/Directory.

- DBC/1012 Data Base Computer Messages Reference Manual (document number C03-0002)

  The messages manual is written for all users. It lists and explains all error messages and return codes generated by the DBC/1012 Data Base Computer.

- DBC/1012 Data Base Computer Reference Cards (document numbers C04-0001, C04-0002, C04-0003)

  The reference cards are written for all users. There are three cards to a set. Each card is a multi-panel, fan-folded summary of language notation, syntax, and acceptable abbreviations. The first card lists DBC/SQL statements and Data Dictionary/ Directory views, the second card lists ITEQ and BTEQ commands, and the third card lists COBOL and PL/I Preprocessor statements.

ITEQ is a trademark of Teradata Corporation.

- DBC/1012 Data Base Computer Operator's Guide (document number C15-0001)

  The operator's guide is written for DBC/1012 operators. The guide presents features of the DBC/1012 and its console, as well as their operating procedures, programs, and status indicators.

- DBC/1012 Data Base Computer Utilities Reference Manual (document number C11-0001)

  The utilities manual is written for DBC/1012 operators and technical personnel. The manual presents the utilities that are used to load, dump, and restore data, initialize and configure a DBC/1012 system, and perform system maintenance.

- DBC/1012 Data Base Support Utilities Manual (document number B07-0031)

  The support utilities manual describes the utilities used by Teradata support personnel to format disks, add and delete AMPs, copy data from one AMP to another, initially load software, and rebuild user tables.

- DBC/1012 Data Base Computer System Manual (document number C10-0001)

  The system manual is writtem for system programmers, application programmers, and DBAs. The manual presents the many considerations and trade-offs for designing and querying DBC/1012 data bases and tables, as well as the details of performance, productivity, startup and shutdown, and software maintenance.

- DBC/1012 Data Base Computer Host Interface Manual (document number C12-0001)

  The host interface manual is written for programmers who use the Call-Level Interface Version 1 (CLIv1) rather than a language preprocessor to communicate with the DBC/1012 system. The manual presents the details of information flow, data structures, and the interface routines. The manual covers CLIv1 for hosts.

- DBC/1012 Data Base Computer Call-Level Interface Manual (document number C12-0006)

  The Call-Level Interface manual is written for programmers who use the Call-Level Interface Version 2 (CLIv2) rather than a language preprocessor to communicate with the DBC/1012 system. The manual presents the details of information flow, data structures, and the interface routines. The manual covers CLIv2 for workstations.

- DBC/1012 Data Base Computer COP Interface Manual (document number C12-0005)

  The COP interface manual is written for system programmers. The manual presents the details of information flow, data structures, and interface routines in the COP Interface software.

- DBC/1012 Data Base Computer Network Reference Manual (document number C03-0003)

  The network reference manual is written for network administrators. The manual gives an overview of the COP interface and the components involved. It also describes how to change the configuration of a DBC/1012 to add COPs and LANs and how to install and configure Teradata's workstation-resident software. The steps and an example are provided for IBM PCs and compatibles using using TCP/IP, IBM PCs and compatibles using ISO/OSI, and AT&T 3B2s using TCP/IP.

- DBC/1012 Data Base Computer Workstation User's Guide (document number C09-0003)

  The user's guide covers the use of BTEQ, showing examples of using BTEQ on-line, debugging and submitting BTEQ scripts, coverting from screen displays to reports sent to a print file, and using DBC/SQL macros. For each set of BTEQ commands, it describes when and how to use them and how the commands are related. It also describes each command in details.

- DBC/1012 Data Base Computer CICS Interface Manual (document number C12-0002)

  The CICS interface manual is written for programmers who access the resources of the DBC/1012 system through CICS.

- DBC/1012 Data Base Computer MVS/VM Host Software Manual (document number C13-0001)

  The MVS/VM host software manual is written for programmers who must understand the Teradata software that resides on the MVS or VM host. The manual describes SVC mode and cross memory services mode under MVS, and corresponding operation under VM.

- DBC/1012 Data Base Computer Planning Guide (document number C07-0001)

  The planning guide is written for personnel who are responsible for the hardware, software, and facility preparation for the DBC/1012. The guide presents physical planning issues and environmental characteristics, as well as software planning and installation considerations.

- DBC/1012 Data Base Computer MVS Software Installation Guide (document number C16-0001)

  The MVS software installation guide is written for technical personnel. The guide provides procedures for installing Teradata MVS interface software.

- DBC/1012 Data Base Computer Preprocessor Reference Manual (document number C03- 0005)

  The preprocessor manual is written for the COBOL and PL/I application programmer. This manual presents details of preprocessor use and includes examples in COBOL and PL/I.

- DBC/1012 Data Base Computer Glossary (document number G01-0001)

  This Glossary is intended for anyone who uses the Teradata DBC/1012 data base computer. This Glossary is a comprehensive document of terms, phrases, acronyms, etc., that apply to any hardware, software, or firmware matter that pertains to the Teradata DBC/1012 data base computer.

This document is the DBC/1012 Data Base Computer Concepts and Facilities manual. Its purpose is to present an overview of the Teradata DBC/1012 Data Base Computer.

The document has six chapters:

1. Chapter 1 presents an historical perspective of the data base problem as an introduction to DBC/1012 features and benefits.

2. Chapter 2 describes the overall architecture of the system.

3. Chapter 3 highlights features that enable an end user to define, and manipulate data.

4. Chapter 4 outlines facilities for controlling the operation and administration of the system.

5. Chapter 5 describes the structure of DBC/1012 hardware subsystems, the organization of system software, elements of host software, and file organization and structure.

6. Chapter 6 lists specifications for system operation, including host computer and environmental requirements, and gives configuration data for multiprocessor DBC/1012 systems.

This document contains a glossary of data base management terms as they relate to the DBC/1012. An index is also included for easy reference.

From time to time, the material in this document is revised. To help you keep track of the various revisions, we will provide you with the following information for each revision:

- The date of the revision
- The software release of the revision
- Change bars in the margin to indicate exactly what information has changed.

Chapter 1 is not highly technical. However, the reader of Chapters 2-6 will benefit from some background in computer technology and data base management.

W. Riefling
July 1987
Los Angeles, California

CONTENTS

Chapter	Page
CHAPTER 1 INTRODUCTION	1- 1
DBC/1012 FEATURES OVERVIEW	1- 1
Expanded Capabilities	1- 2
Ease of Use	1- 3
Performance	1- 3
Reliability and Availability	1- 3
Productivity	1- 3
ARCHITECTURAL OVERVIEW	1- 4
Communications Processor (COP)	1- 5
DBC/1012 Software	1- 6
RELATIONAL DATA BASE	1- 6
Table Operations	1- 7
Views	1- 8
Advantages of the Relational Model	1- 8
SUMMARY AND PREVIEW	1- 9
CHAPTER 2 ARCHITECTURE	2- 1
HOST SYSTEM COMMUNICATION INTERFACE (HSCI)	2- 1
INTERFACE PROCESSOR (IFP)	2- 3
PARALLEL STEPS	2- 6
INTERPROCESSOR BUS (YNET)	2- 6
ACCESS MODULE PROCESSOR (AMP)	2- 8
DISK STORAGE UNIT (DSU)	2- 9
COMMUNICATIONS PROCESSOR (COP)	2-11
SYSTEM CONSOLE WITH PRINTER	2-11
SUMMARY AND PREVIEW	2-13
CHAPTER 3 USER FACILITIES	3- 1
DBC/SQL	3- 1
Operations on Tables	3- 1
Data Types	3- 2
DBC/SQL Statements	3- 3
Names	3- 3
Keywords	3- 4
Delimiters	3- 4
Separators	3- 4
Constants	3- 4
Referencing Data in DBC/SQL	3- 5
DBC/SQL Expressions	3- 5
Arithmetic Operators	3- 6
Aggregate Operators	3- 6
Comparison Operators	3- 6
Logical Operators	3- 7
Partial String Matching Operators	3- 7
Set Operators	3- 7
Other Operators	3- 8
Arithmetic Functions	3- 8
Defining Data	3- 8

Creating Tables 3- 9
Adding Columns to a Table 3-11
Creating Indexes 3-11
Creating Views 3-12
Removing Tables, Indexes, and Views 3-13
Selecting Data 3-13
Making Simple Queries 3-14
Selecting Specific Columns 3-14
Selecting Specific Rows 3-14
Specifying Order 3-15
Defining Groups 3-16
Using More than One Table 3-17
Nesting Subqueries 3-17
Manipulating Data 3-18
Adding Rows to a Table 3-18
Changing Rows 3-19
Deleting Rows from a Table 3-19
Macros 3-19
Controlling Data 3-20
HELP Statement 3-20
INTERACTIVE TERADATA QUERY (ITEQ) FACILITY 3-21
Screen Format 3-22
Establishing a Session 3-23
Entering Statements and Commands 3-23
Formatting Data 3-24
Requesting Column Summaries 3-24
Changing Headings and Titles 3-24
Format Commands 3-24
BATCH TERADATA QUERY (BTEQ) FACILITY 3-27
DBC/1012 COORDINATED PRODUCTS 3-30
The DBC/1012 INTELLECT Interface 3-30
NOMAD2 Interface 3-30
PC/SQL-link Interface 3-31
FOCUS Interface 3-31
IDEAL Interface 3-31
THE DBC/1012 COP 3-32
DATA DICTIONARY/DIRECTORY 3-33
Information Levels 3-33
Querying the Dictionary/Directory 3-34
System Views 3-34
End User Views 3-34
Supervisory User Views 3-35
DBC/1012 Administrator Views 3-35
Recovery Control User Views 3-36
LANGUAGE PREPROCESSORS 3-37
CALL-LEVEL INTERFACE 3-38
SUMMARY AND PREVIEW 3-39

CHAPTER 4 SYSTEM FACILITIES 4- 1

SESSION PROTOCOL 4- 1
DATA PROTECTION 4- 1
Concurrency Control 4- 2
Transient Journal 4- 3
Permanent Journaling 4- 3
Redundant Data Storage 4- 4
Transaction Management 4- 4

Recovery 4- 5
USER SECURITY INTERFACE 4- 5
USER LOGON INTERFACE 4- 6
SPACE ALLOCATION AND ACCESS CONTROL 4- 6
Creating Data Bases and Users 4- 6
Granting and Revoking Privileges 4- 7
CREATE DATABASE Statement 4- 9
CREATE USER Statement 4- 9
GENERAL PURPOSE UTILITY PROGRAMS 4-10
Archiving and Restoring a Data Base 4-10
Bulk Loading Data from a Host 4-10
Fast Loading Data from a Host 4-10
RECONFIGURATION 4-10
SYSTEM MAINTENANCE 4-11
SYSTEM CONSOLE OPERATION 4-11
SYSTEM STATUS AND STATISTICS 4-12
System and Configuration Status 4-12
ResUseView View 4-12
ACCOUNTING 4-13
SUMMARY AND PREVIEW 4-13

CHAPTER 5 STRUCTURE 5- 1

HARDWARE SUBSYSTEMS 5- 1
Ynets 5- 1
Processor Modules 5- 2
Disk Storage Units 5- 6
Packaging 5- 6
SOFTWARE SYSTEMS 5- 6
DBC/1012 Software 5- 7
Teradata Operating System (TOS) 5- 7
Parser System 5- 8
Data Base System 5- 8
Session Control 5- 8
Dispatcher 5- 8
Data Base Manager 5- 8
Host-Resident Software 5- 9
Host System Communication Interface 5- 9
Teradata Director Program 5-10
User-to-TDP Communication 5-10
Call-Level Interface Library 5-10
Interactive TEradata Query Facility 5-10
Batch TEradata Query Facility 5-11
Language Preprocessors 5-11
Host-Resident Utility Programs 5-11
SUMMARY AND PREVIEW 5-11

CHAPTER 6 OPERATING AND CONFIGURATION SPECIFICATIONS 6- 1

OPERATING CHARACTERISTICS 6- 1
HOST REQUIREMENTS 6- 1
LAN-ATTACHED HOST REQUIREMENTS 6- 2
SYSTEM AND CONFIGURATION SPECIFICATIONS 6- 2
POWER SPECIFICATIONS 6- 3
ENVIRONMENTAL REQUIREMENTS 6- 3

GLOSSARY G- 1

INDEX X- 1

ILLUSTRATIONS

Figure **Page**

1-1 Basic DBC/1012 Configuration 1- 5
1-2 Expandability of DBC/1012 System 1- 6
1-3 Example Table in a Relational Data Base 1- 7
2-1 MVS Host Operating Environment 2- 2
2-2 VM/SP Host Operating Environment 2- 4
2-3 IFP 2- 5
2-4 IFP Routing of DBC/SQL Request Messages 2- 7
2-5 Ynet 2- 8
2-6 AMP 2- 9
2-7 Disk Storage Unit 2-10
2-8 Distribution of Primary and Fallback Data 2-11
2-9 Distribution of Data With Clustered AMPs 2-12
2-10 Workstation Operating Environment 2-13
3-1 Table of Employee Data 3- 9
3-2 Table of Department Data 3- 9
3-3 ITEQ Screen Format 3-22
3-4 Result of a Query 3-23
3-5 Column Totals 3-25
3-6 Changing Headings and Titles 3-26
3-7 Printed Report 3-27
3-8 BTEQ Report 3-29
3-9 COP Interface Components 3-32
4-1 Ownership Structure in a Data Base 4- 7
5-1 Basic Ynet Configuration 5- 2
5-2 Multi-Level Network Configuration 5- 3
5-3 DBC/1012 Hardware Configuration (2 parts) 5- 4

Table **Page**

3-1 End User Views 3-35
3-2 Supervisory User Views 3-35
3-3 DBC/1012 Administrator Views 3-36
3-4 Recovery Control Views 3-37

CHAPTER 1 INTRODUCTION

The DBC/1012 Data Base Computer is a complete database management system. The DBC/1012 attaches directly to the high speed I/O channels of mainframe computers, and to minicomputers and intelligent workstations via local area networks (LANs).

The DBC/1012 Data Base Computer is built on advances in data base management technology that have evolved since the late 1960s. This chapter discusses the problems of previous data base management systems and presents the features and benefits of the Teradata solution. There is a brief description of DBC/1012 architecture, along with a discussion of the advantages of a relational data base management system.

The Teradata DBC/1012 employs a unique architecture of multiple processors, software, and direct-access storage devices. Distinguished as the first system to provide high performance, full-function relational data base management capabilities. In fact, the DBC/1012 Data Base Computer breaks the central processing unit bottleneck by harnessing the cost-efficient power of multiple microprocessors operating in parallel.

This design represents a major departure from conventional processing techniques. The design enables the DBC/1012 system to sustain concurrent access to the relational data base from end-user interactive sessions, on-line transaction systems, and high-volume batch jobs. All data is accessed via a single, high-level, nonprocedural language. This flexibility and ease of use is combined with *parallel processing*, which maintains throughput that simply cannot be achieved by a conventional monolithic processor configuration.

The architecture provides a degree of *modularity* in disk storage capacity and processing power unavailable in conventional systems. Relational data bases stored by the DBC/1012 Data Base Computer may range from modestly sized data bases to systems of up to a terabyte (1,000,000,000,000 characters). As the size of a data base grows, additional processing power may be added in small increments. This modularity enables the DBC/1012 Data Base Computer to achieve performance levels unattainable by any conventional relational data base management system (DBMS).

The DBC/1012 Data Base Computer relieves the host of its heavy software DBMS burden. Moreover, it provides major improvements over conventional software approaches in such areas as price/performance, modularity, and data integrity. When attached to a LAN, the DBC/1012 can operate as a data base server in a networking environment. *Sharability* permits the user to simultaneously access data from a heterogenous group of hosts. The fully redundant hardware, software, power, and data storage (if desired) of the DBC/1012 system provide the necessary *availability* and *reliability* to ensure continuous operation of this major system component.

DBC/1012 FEATURES OVERVIEW

The Teradata DBC/1012 Data Base Computer solves problems inherent in conventional data base management systems while meeting new requirements of users. The DBC/1012 is a system of integrated hardware and software that relieves the attached host(s) of the software DBMS burden.

The DBC/1012 Data Base Computer may be attached to one or more "host" computers, from which users can access the data. A host computer may be any one of three types of computers: mainframes, minicomputers, or intelligent workstations (e.g., personal computers, microcomputers, etc.).

The DBC/1012 Data Base Computer uses a unique architecture of multiprocessors (hardware) and software that embodies major advances over software DBMS approaches. The following sections highlight the system's features, benefits, and approaches.

Expanded Capabilities

The DBC/1012 system provides all of the functions of the best software data base management systems, and more. Features that increase the functions of the DBC/1012 are

- A SINGLE, COMPREHENSIVE LANGUAGE for data definition, data manipulation, and report writing.
- DATA INTEGRITY AND CONSISTENCY during concurrent access by many users.
- DATA PROTECTION from failure of a host computer, failure of a DBC/1012 processor or Disk Storage Unit, or abnormal termination of an application program. Comprehensive system facilities automatically handle recovery and restart operations. Database back up features are also available (i.e., permanent journaling, transient journaling, etc.).
- SEVERAL OPERATING ENVIRONMENTS to enable users to use DBC/1012 data bases interactively, in batch mode, and in on-line transaction systems.
- GENERAL PURPOSE UTILITIES for dumping and restoring data bases and loading or extracting data at a host computer.
- DATA DICTIONARY/DIRECTORY to let the data base administrator control the use of data. This fully integrated, active dictionary/directory also lets users obtain information about DBC/1012 data bases.
- SECURITY FEATURES to protect the data of individual users as well as on a global basis. Every user has comprehensive control over access to the user's own data. Access by all users is controlled by password. No application code resides or runs in the DBC/1012 system, and data storage units are not removable.
- MODULAR DESIGN to permit growth that corresponds directly to user requirements. The system can accommodate up to 968 processors by adding increments as small as a single processor.
- SHARABILITY permits multiple users to simultaneously access data bases stored in a DBC/1012.

Ease of Use

The DBC/1012 system is easier to use than conventional software data base management systems. Features that make the DBC/1012 easy to use include

- A SIMPLE QUERY LANGUAGE to exercise the full range of system capabilities. A single language, DBC/SQL, is used for data definition, manipulation, and control. DBC/SQL is compatible with the IBM SQL data manipulation language. DBC/SQL statements may be used directly by interactive users or may be embedded into applications programs.
- RELATIONAL DATA STRUCTURES to simplify data organization and manipulation. Data is represented as one or more tables consisting of columns and rows.
- An INTERACTIVE FACILITY to provide access to the DBC/1012 system for non-technical users. ITEQ (Interactive TEradata Query facility) enables a user at a 3270 terminal attached to a mainframe to interact directly with the DBC/1012 Data Base Computer.

- A BATCH FACILITY to provide access to the DBC/1012 Data Base Computer in batch mode. BTEQ (Batch TEradata Query facility) also features advanced report-writing capabilities.

- EFFICIENT HOST COMMUNICATION SOFTWARE to relieve users of the burden of writing routines access the DBC/1012. Teradata Corporation provides a library of interface routines to accelerate the user's effective use of the system.

Performance

The DBC/1012 Data Base Computer provides a level of performance that cannot be achieved with software-only data base management systems. Its unique architecture provides superior response time in most application areas. Exceptional performance is achieved in applications involving complex queries against a large data base. Features specific to performance include

- The use of MICROPROCESSOR TECHNOLOGY to provide cost-effective processing power.

- PARALLEL PROCESSING to eliminate one-operation-at-a-time bottlenecks. Data is spread across a number of processors and disk storage units. Processing is performed asynchronously by a number of independent processors.

- PARALLEL ROLLBACK AND RECOVERY FUNCTIONS to restore a system to on-line operation with a complete and consistent data base in minimal time.

- SYSTEM EXPANDABILITY to bring increased processing power to bear on a growing data store.

Reliability and Availability

The DBC/1012 Data Base Computer addresses the critical requirements of high reliability and availability. This is done by combining multiple microprocessors with a system for protecting the data base from operating anomalies of the host system. The DBC/1012 is reliable and available to the user because of:

- DUPLICATED DATA STORE to provide continuous operation without user intervention. At the user's option, the DBC/1012 Data Base Computer automatically maintains a secondary copy of the data. This permits normal operations to continue if the primary copy of the data becomes temporarily unavailable.

- SYSTEM DISRUPTION IS MINIMAL thereby increasing system availability. The DBC/1012 system design incorporates redundant hardware and software components. This redundancy ensures that failure of a single component does not effect ongoing operation of the other components in the system.

- ISOLATION OF THE DATA BASE MANAGEMENT SYSTEM AND DATA operating system to protect the DBC/1012 system from the effects of a failure in any of the attached host(s).

Productivity

The DBC/1012 system lets an organization maximize its investment in human resources by making it easier for people to be more productive. Features that enhance productivity include

- A POWERFUL, EASY-TO-LEARN LANGUAGE which makes it easy to use the DBC/1012 system. Novice users are able to master the system in a short time, and sophisticated users can develop applications more quickly.
- SUPERIOR RELIABILITY to ensure availability of data whenever it is needed.
- HIGH PERFORMANCE to satisfy user data needs quickly.

These advantages can be attributed to the unique architecture of the DBC/1012 Data Base Computer.

ARCHITECTURAL OVERVIEW

The DBC/1012 Data Base Computer is a dedicated relational data base management system. Except for Host System Communication Interface software (provided by Teradata Corporation as part of the DBC/1012 system), all hardware, firmware, and software reside in an environment separate from the host computer. The DBC/1012 system is comprised of the following subsystems.

- Host System Communication Interface (HSCI)
- Interface Processor (IFP)
- Ynet™ (Interprocessor Bus)
- Access Module Processor (AMP)
- Disk Storage Unit (DSU)
- Systems Console with Printer
- Communications Processor (COP)

The interrelationship of these subsystems (except for the COP), in a basic system configuration, is shown in Figure 1-1. Each subsystem is outlined below and described in greater detail in Chapter 2.

The HSCI consists of a small library of service routines and an interface program. This software resides in the mainframe computer and enables users to conduct sessions with the DBC/1012 Data Base Computer. Sessions may be conducted in interactive, batch, or on-line transaction processing environments.

IFPs manage the traffic between the mainframe computer and the DBC/1012 Data Base Computer. IFPs translate requests from a host into internal commands, forward the commands over the Ynet to the AMPs, and coordinate the responses as they return over the Ynet from the AMPs. As this request traffic increases over time, it may be necessary to add IFPs to a DBC/1012 system.

The Ynet is an independent, intelligent bus that interconnects all IFPs, COPs, and AMPs and performs -- in hardware -- many of the complex tasks associated with multiprocessor management. To provide fail-safe operation, there are two fully operational Ynets in each DBC/1012 system. In contrast to an ordinary, passive bus, the Ynet is an array of active logic. The name "Ynet" is derived from the pictorial representation of this subsystem, a network of high-speed circuits that resemble upside-down Ys arranged in a lattice. AMPs receive the requests that are forwarded by the

---

Ynet is a trademark of Teradata Corporation.

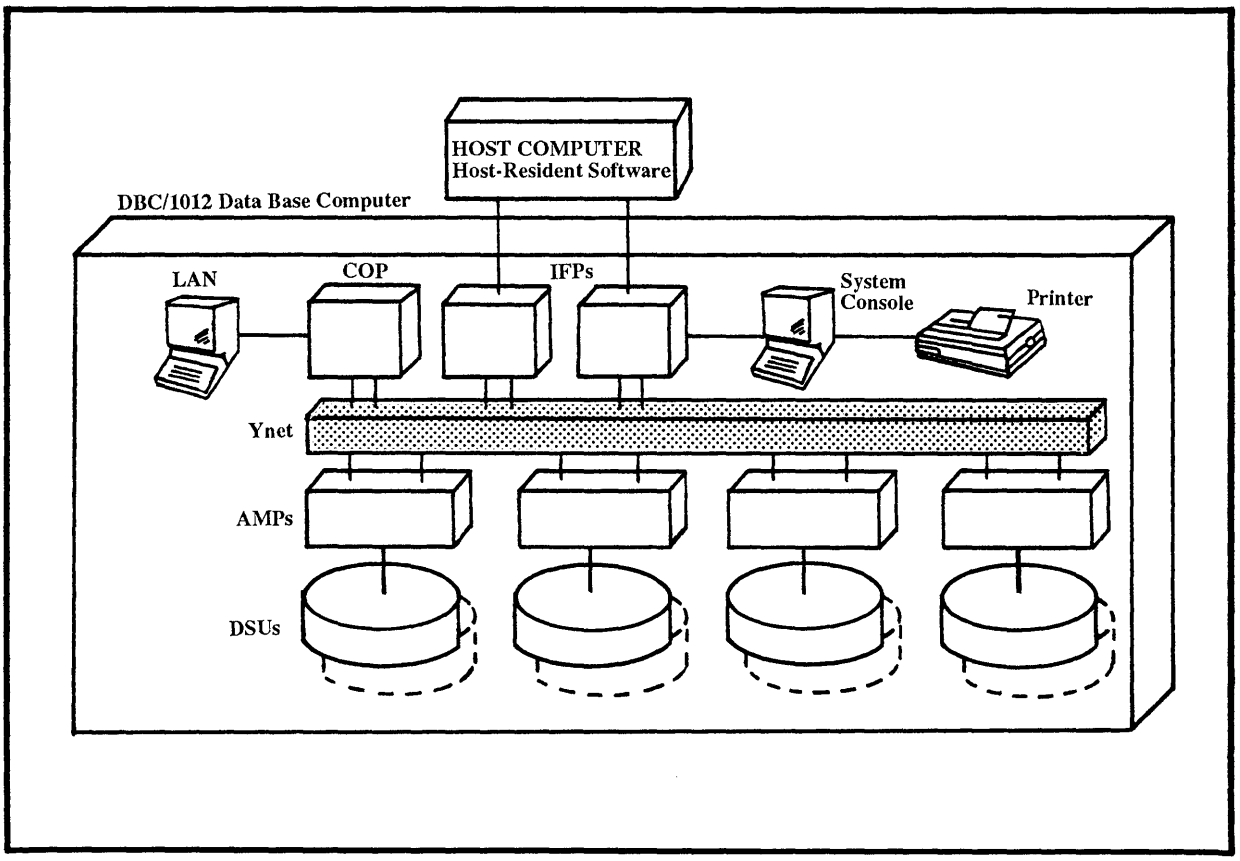

Figure 1-1. Basic DBC/1012 Configuration

IFPs and COPs, and perform the required data manipulations. The AMPs then send appropriate responses back over the Ynet to the IFPs and COPs, which return the responses to the original point of request. Data is distributed evenly across the DSUs of all AMPs. As a result, workload in the AMPs is balanced and data can be processed in parallel. When AMPs are added to a DBC/1012 system to increase storage capacity (or to increase performance) , data is automatically redistributed across the DSUs.

DSUs serve as the data storage medium for the system. Each DSU contains a portion of the total data stored on the DBC/1012 Data Base Computer. To increase space available for data storage, a second DSU may be attached to each AMP. The system console (an IBM or compatible personal computer) allows an operator to communicate with the DBC/1012 Data Base Computer in order to get reports of system status, current configuration, and performance. The console is also used to control system and diagnostic operations. A printer attached to the system console provides a printed record of everything that is displayed at the console.

As shown in Figure 1-2, the user can expand the DBC/1012 system to accommodate a number of mainframes, IFPs, and AMPs.

Communications Processor (COP)

A Communications Processor (COP) is a special IFP installed in a DBC/1012 that allows the user to access the DBC/1012 from a local area network (LAN).

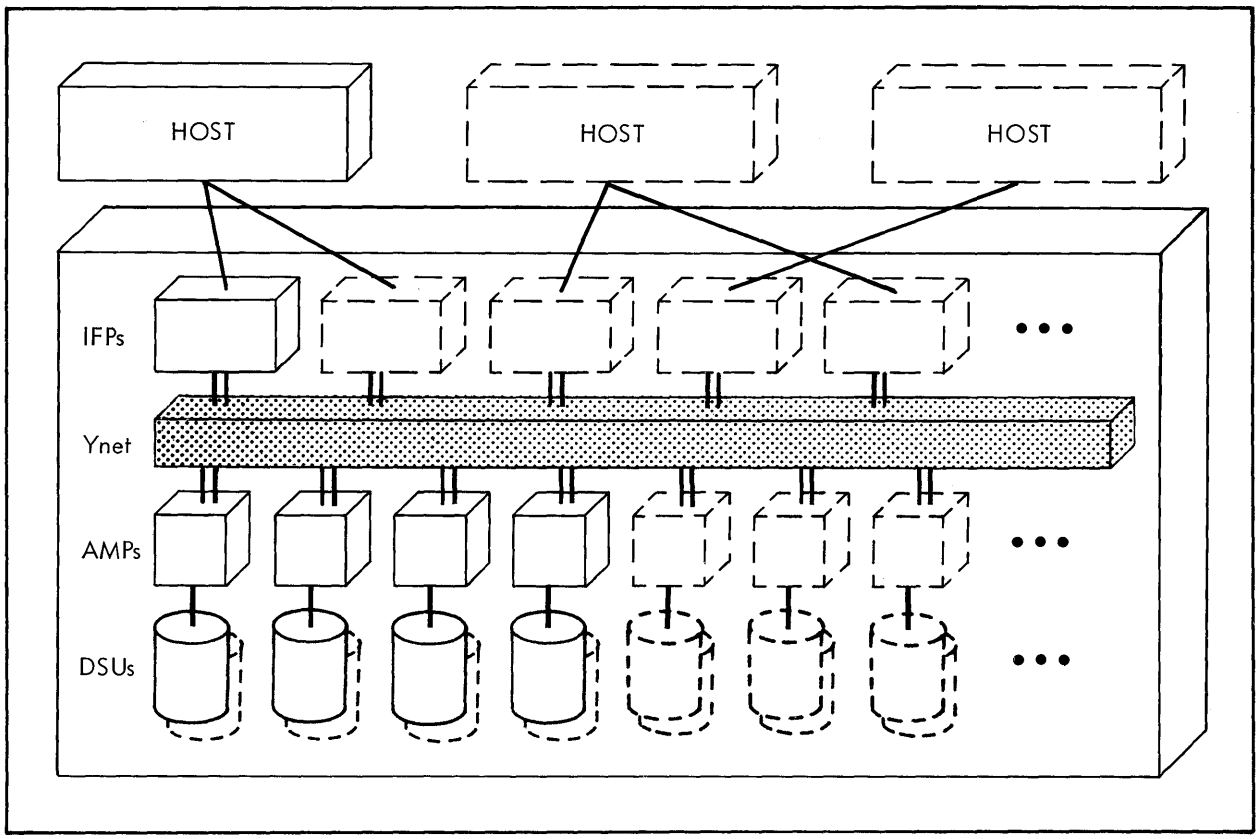

Figure 1-2. Expandability of DBC/1012 System

DBC/1012 Software

There are several software components, located in the DBC/1012 processors, that supplement the host-resident software and hardware subsystems. These software packages include

- The Teradata Operating System (TOS), a virtual-memory, multi-user, real-time operating system
- A Parser System, which supports DBC/SQL
- A Data Base System, which supports DBC/1012 data definition and manipulation functions

These software packages are described more fully in Chapter 5. While most users will not be aware of the inner workings of the DBC/1012 Data Base Computer, they will quickly become familiar with the terminology, capabilities, and advantages of a relational data base management system.

RELATIONAL DATA BASE

Data on the DBC/1012 Data Base Computer is stored in one or more relational tables. In a table, data can be thought of as arranged in vertical columns and horizontal rows. The Employee table in Figure 1-3 has five columns and nine rows.

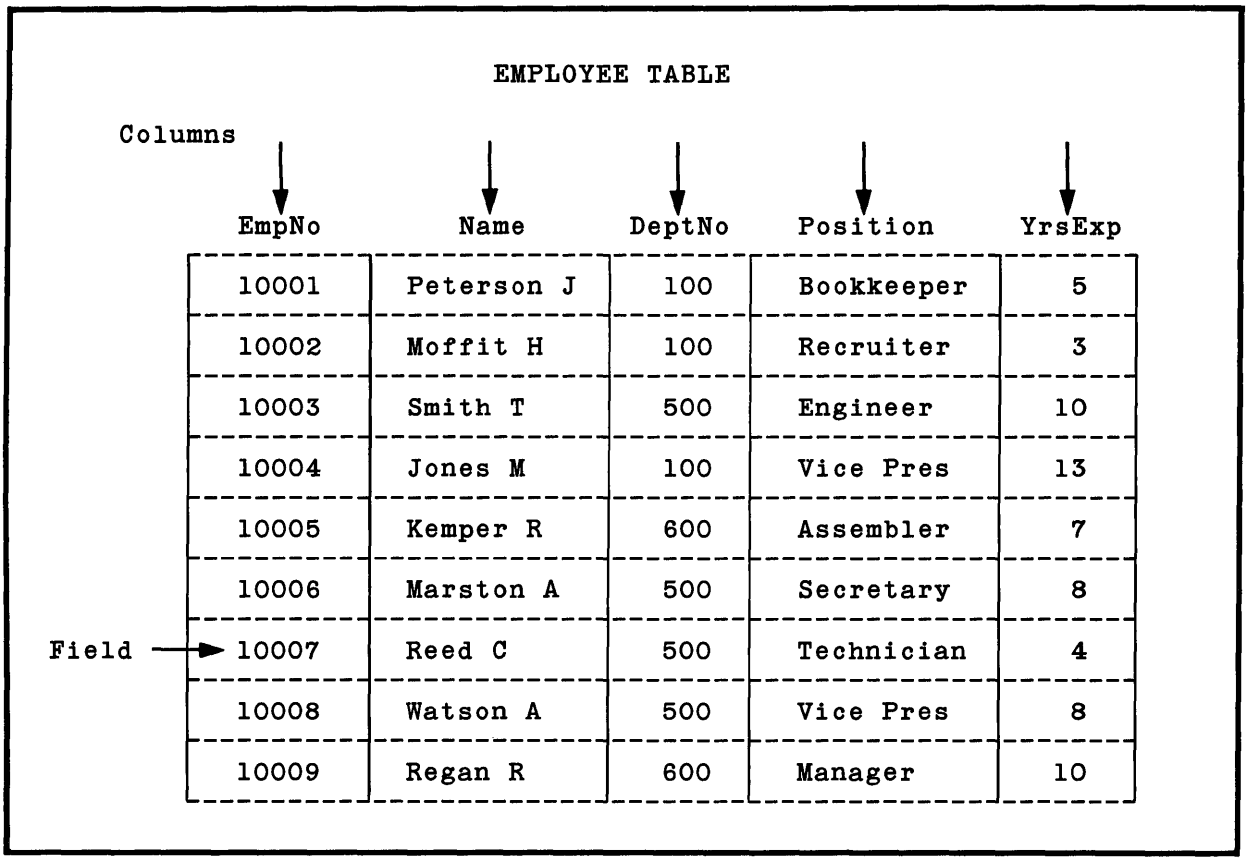

Figure 1-3. Example Table in a Relational Data Base

A row is an entry in the table. Each row is characterized by a number of attributes, called columns. For example, each row in the Employee table is characterized by an employee number (EmpNo), name (Name), department number (DeptNo), position (Position), and years of experience (YrsExp).

The intersection of a column and a row is called a "field". The term "field value" refers to the specific data value in a field. In the figure, the field value for row 3 of the Position column is "Engineer".

Table Operations

Data is defined in terms of tables and selected by operations on tables. A table is defined by giving it a name and identifying its columns by their names and characteristics. All fields in a column have the same characteristics. For example, all values in the YrsExp column in Figure 1-3 are integers.

A row is the smallest unit that can be inserted into or deleted from a table. An insert operation adds one or more rows to a table. A delete operation removes one or more rows from a table. In general, rows have no inherent order and can be selected in any order a user specifies. Similarly, columns have no inherent order and can be selected in any order the user specifies.

A field is the smallest unit of data that can be modified. An update operation replaces one or more field values, but never a part of a value, for one or more rows. Basic operations on tables are:

- Creating or deleting tables
- Adding or deleting columns
- Inserting or deleting rows
- Updating existing rows
- Updating existing columns
- Selecting data from existing rows.

A user can select individual columns (or all columns) from specific rows (or all rows) from one or more tables. A select operation that invoice data in more than one table is called a "join".

Views

A view is a "window" through which a user looks at selected portions of a table. The view is an apparent table derived from one or more real tables. Views can also be derived from other views, or combinations of views and tables.

A view looks like a stored table. It has rows and columns and, in general, can be used as if it were a table. It is an "apparent" table because it does not physically exist; that is, it does not take up any storage space. A view is typically used to simplify query requests, to limit a user's access to or manipulation of data, and to allow different users to look at the same data from a different perspective.

Advantages of the Relational Model

A relational data base management system has many practical advantages. In particular, a data base is easy to learn because there is only one simple data structure, the table. As a result, data is conceptually easy for a user to select, manipulate, and control using the nonprocedural DBC/SQL language. Using DBC/SQL, the user simply states the results desired, rather than a procedure for deriving the results.

For example, given the Employee table in Figure 1-3, the DBC/SQL statement

```
SELECT Name
FROM Employee
WHERE EmpNo = 10006;
```

produces the name of the employee whose Employee Number is 10006: Marston A.

In contrast to DBC/SQL, the data manipulation language in most software data base management systems is procedural. To select a single record, an application programmer must first know an access path to the desired record and then traverse the path, step by step, to reach it. Each step requires an explicit statement to describe it, and the navigation process must be repeated for each record to be selected.

This navigation requires a high level of user expertise, and productivity is relatively low. Also, as the data base grows in size and complexity, the navigation process originally selected may become cumbersome or ineffective.

Businesses continually store new data, generating new requirements for using and relating data in the process. Often these requirements necessitate structural changes in a data base. Structural changes in

a conventional, non-relational data base management system require changing the application program to reflect the new access paths. Thus, these structural changes are difficult to make.

In contrast, the relational data base management system of the DBC/1012 Data Base Computer facilitates changes in data structures and contents. This flexibility provides two long-term benefits to the DBC/1012 user: individual programs are safeguarded against obsolescence, and the system as a whole has a much longer, more productive life cycle.

SUMMARY AND PREVIEW

This chapter introduced the features of the DBC/1012 Data Base Computer. Subsequent chapters discuss these features in greater detail.

Chapter 2 describes the unique architecture of the DBC/1012 system.

Chapter 3 discusses user facilities for defining, and manipulating data; interacting with the DBC/1012 Data Base Computer from an on-line terminal; using the Data Dictionary/Directory to obtain information about DBC/1012 objects (e.g., tables) and using DBC/SQL within a COBOL application program.

Chapter 4 describes DBC/1012 system facilities for protecting data; allowing users to access DBC/1012 data; archiving data bases; maintaining and reconfiguring the system; and allocating system resources.

Chapter 5 details the structure of the DBC/1012 system, including hardware and software systems.

Chapter 6 presents the operating and configuration specifications of the DBC/1012 system.

CHAPTER 2 ARCHITECTURE

The DBC/1012 Data Base Computer uses multiple microprocessors, firmware, and software in its architecture. As discussed in Chapter 1, its software and hardware systems are composed of the following subsystems:

- Host System Communication Interface (HSCI)
- Interface Processor (IFP)
- Interprocessor Bus (Ynet)
- Access Module Processor (AMP)
- Disk Storage Unit (DSU)
- Communications Processor (COP)
- System Console with Printer

This chapter describes how each of these subsystems functions to process user requests for data. For a structural description of each subsystem, refer to Chapter 5.

HOST SYSTEM COMMUNICATION INTERFACE (HSCI)

The Host System Communication Interface (HSCI) supports user sessions with the DBC/1012 Data Base Computer in the following host system environments:

- MVS/XA (all releases); MVS/SP (release 1.3 and above)
- VM/SP (release 3 and above)

The HSCI consists of:

- Call-Level Interface (CLI)
- User-to-TDP Communication (UTC) routines:
  - MVS: SVC and Cross Memory Services (XMS)
  - VM: Inter-User Communication Vehicle (IUCV)
- Teradata Director Program (TDP)

User sessions can be conducted with the DBC/1012 in an interactive or a batch environment. In Figure 2-1, which illustrates the MVS host operating environment,

- One end user has direct, interactive access to data via the Interactive TEradata Query facility (ITEQ). ITEQ is an application program that operates under TSO (Time Sharing Option), the MVS time sharing facility.
- Other "users" of data in DBC/1012 data bases are an application program running in batch mode and an on-line transaction program operating under CICS (Customer Information Control System).
- The TDP, described below, operates as a separate "server" address space.

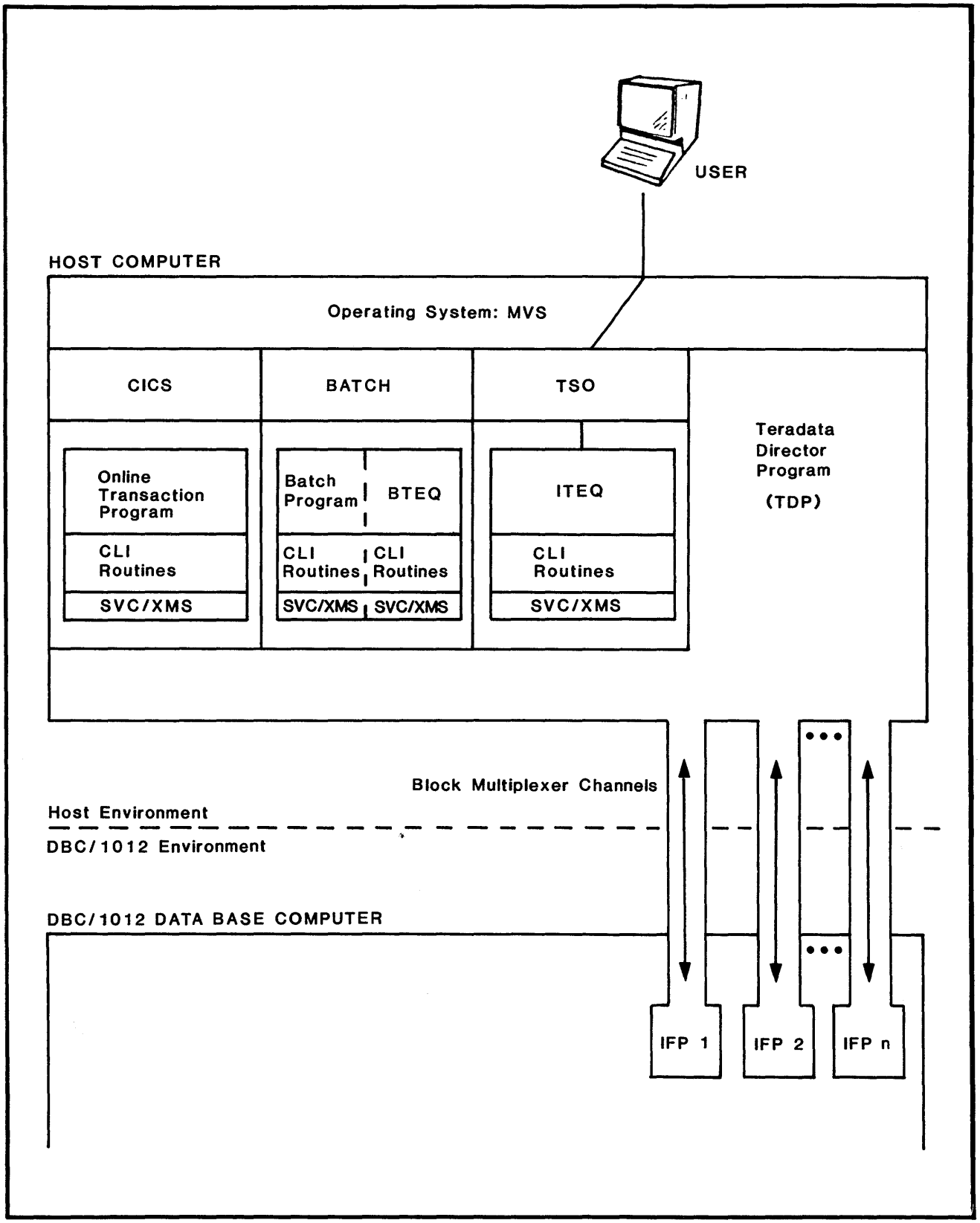

Figure 2-1. MVS Host Operating Environment

In Figure 2-2, which illustrates the VM/SP host operating environment,

- One end user has access to data via an on-line transaction program
- Another end user has interactive access to data via ITEQ
- Another user is operating in a batch environment
- The TDP, described below, operates as a separate "server" virtual machine.

Under VM, all users access data through the Conversational Monitor System (CMS), the VM time-sharing facility.

For users in either host environment, Call-Level Interface (CLI) service routines provide a logical, consistent protocol for communicating requests to the TDP and the DBC/1012, and for receiving responses. CLI routines are used in different ways, for example:

- They may be called by Teradata supplied software such as ITEQ or BTEQ as a result of DBC/SQL statements entered directly by a user.
- They may be called indirectly by user-written application programs written in high-level languages such as COBOL and PL/1.
- They may be called directly by programs written in Assembly language or any other language that has a "call" facility.

The Teradata Director Program (TDP) manages communication between users and the DBC/1012 Data Base Computer. The TDP runs in a dedicated address space in an MVS environment, and as a virtual machine in a VM environment.

The CLI routines and the TDP communicate via the User-to-TDP Communication (UTC) routines, which physically manage communication of requests between Teradata-supplied or user-supplied applications and the TDP.

A CLI routine converts a CLI call into a DBC/SQL request and communicates the request via a UTC routine to the TDP. The TDP, in turn, creates a request message and communicates it over a block multiplexer channel for processing by an Interface Processor (IFP).

CLI routines also receive responses to DBC/SQL requests from the DBC/1012 Data Base Computer via the TDP and return them to the originating application program (e.g., ITEQ). If a DBC/SQL request has a single result, the routines return one row of data. If a result contains a number of rows, the routines perform "deblocking" and serve the requestor with one row at a time.

DBC/SQL statements may be embedded in a COBOL program or a PL/1 program. The COBOL Preprocessor or the PL/1 Preprocessor, both provided by Teradata, converts embedded statements into calls to CLI routines and preprocessor support routines.

After the program is compiled, it may be link-edited with CLI routines to form an executable load module, or the routines may be dynamically loaded at runtime. Applications written in high-level languages that have a CALL statement (including COBOL and PL/1) may also call CLI routines directly (refer to Chapter 3).

INTERFACE PROCESSOR (IFP)

The Interface Processor (IFP), illustrated in Figure 2-3, manages the dialogue between a user session in a mainframe and the DBC/1012 Data Base Computer. When a DBC/1012 Data Base Computer

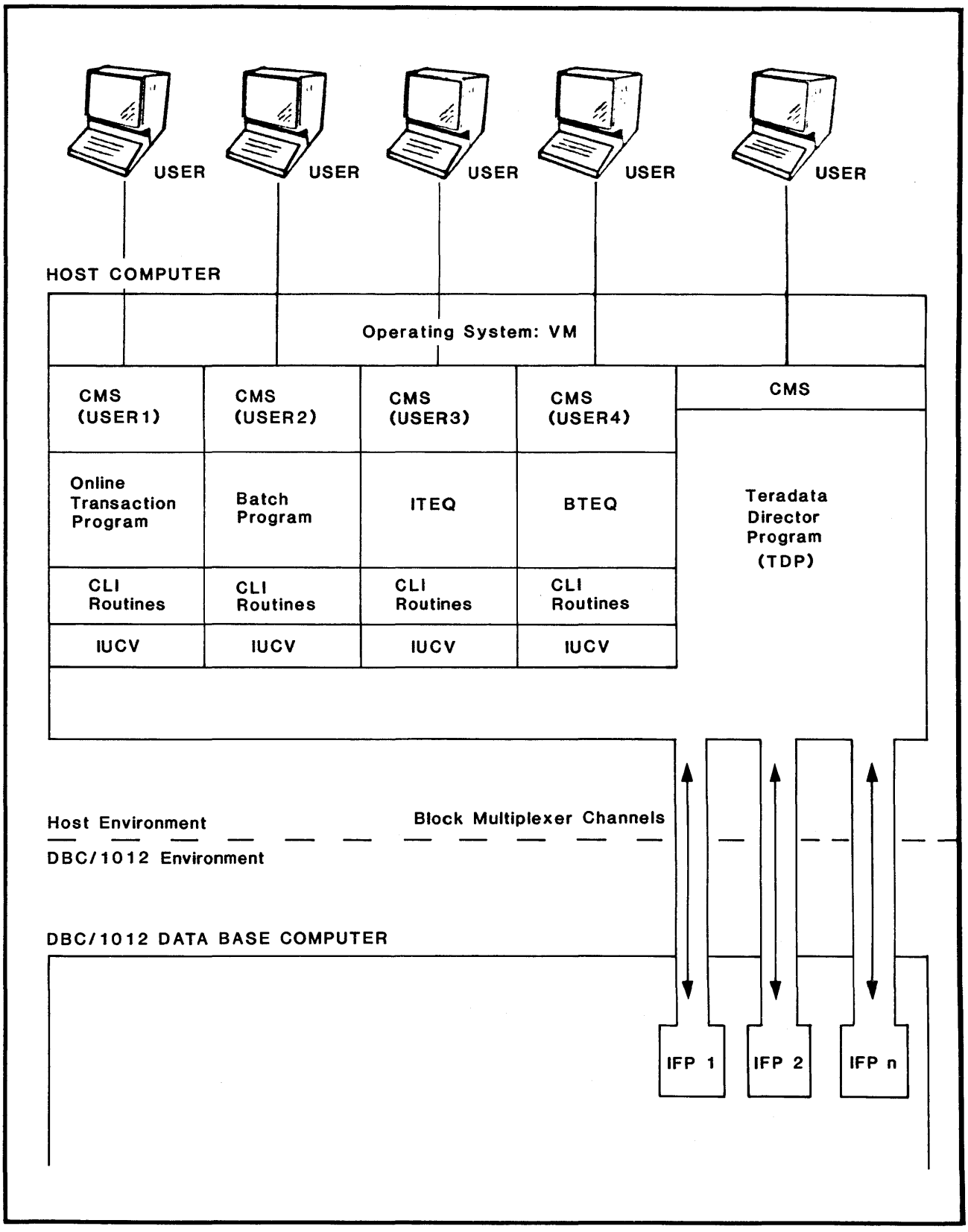

Figure 2-2. VM/SP Host Operating Environment

is attached to several mainframes, at least two IFPs are normally assigned to each mainframe in order to achieve redundancy and fail-safe operation.

An IFP has five components:

1. Session Control
2. Host Interface
3. Parser
4. Dispatcher
5. Ynet Interface

Session Control processes logon and logoff requests from a host and establishes sessions.

The Host Interface controls the exchange of messages between the TDP in a mainframe and the DBC/1012 Data Base Computer.

When an IFP receives a DBC/SQL request message from a TDP, the Parser interprets it, checks it for syntax, and evaluates it semantically. In processing the request message, the Parser refers to the Data Dictionary/Directory (system tables containing information about DBC/1012 data bases) to

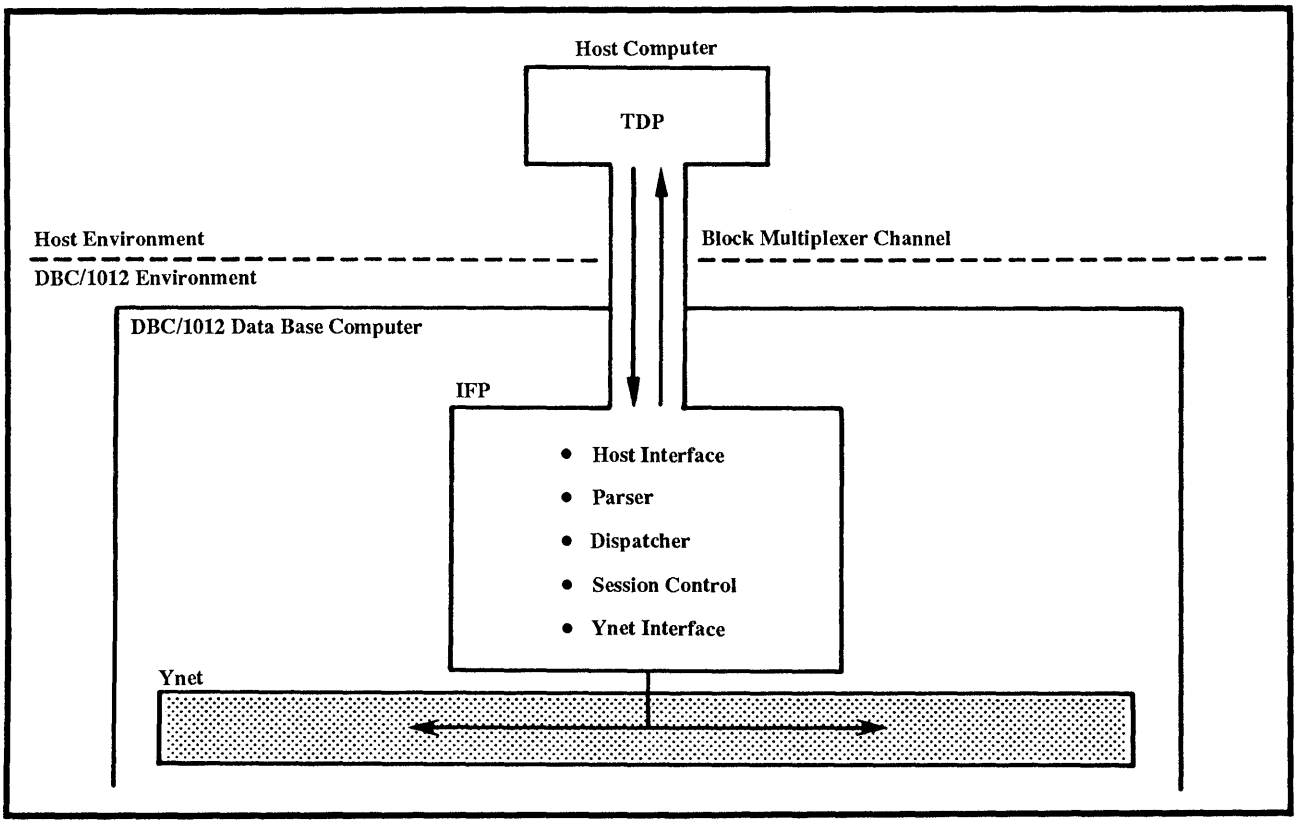

Figure 2-3. IFP

resolve symbolic data names and to make integrity checks. Finally, the Parser decodes the request into a series of work steps necessary for routing and resolving the request and passes them on to the Dispatcher.

The Dispatcher controls the sequence in which the steps are executed and passes the steps to the Ynet Interface.

The IFP Ynet Interface controls the transmission of messages over the Ynet to and from the AMPs. As noted below under Access Module Processor, when rows are inserted in a table, the data is automatically distributed evenly across all AMPs in the system. This distribution increases throughput by allowing table data to be accessed simultaneously on all AMPs. This principle is illustrated in Figure 2-4, where twelve rows of a table are distributed among DSUs attached to four AMPs. Depending on whether a request is for data in a single row (a "primary index" request) or data from many rows (an "all-AMP" request), the IFP either (2.4) transmits steps to a specific AMP (see IFP 1 in Figure 2-4) or it tells the Ynet to "broadcast" the steps to all AMPs (see IFP 2 in the figure). Also, in order to minimize unnecessary system overhead, the IFP can send a step to a subset of AMPs when the request asks for data that resides on more than one (but not all) of the AMPs.

For example, consider the following two DBC/SQL statements from a table of checking account information:

```
1.   SELECT * FROM Table_01 WHERE AcctNo = 129317 ;

2.   SELECT * FROM Table_01 WHERE AcctBal > 1000 ;
```

Assume that IFPs 1 and 2 receive requests 1 and 2, respectively; assume that data for account number 129317 is contained in table row R9, which is stored on AMP 1; and assume that information about all account balances is distributed evenly among the DSUs of all four AMPs. The IFP 1 Parser determines that its request is a primary-index retrieval, which calls for access and return of one specific row (record). IFP 1 then issues a message to the Ynet containing an appropriate read step and R9/AMP 1 routing information.

Having determined that an IFP2 request may require the retrieval of data that resides on any or all AMPs, it issues a message to the Ynet containing an appropriate read step that is broadcast to all four AMPs. Once results are received from the AMPs, the IFP immediately transmits the data back to the TDP over the block multiplexer channel.

PARALLEL STEPS

To enhance system performance, the DBC/1012 executes in parallel steps, whenever possible. Parallel steps can work with multistatement requests, macros, and single statements and can provide significant improvements in response time. For example, the response time of a multistatement request, that consists of four statements which can be independently executed, may be cut in half.

INTERPROCESSOR BUS (YNET)

As shown in Figure 2-5, dual interprocessor buses, or Ynets, connect the IFPs, COPs, and AMPs of the DBC/1012 system. The name "Ynet" is derived from the schematic representation of a network node, an upside-down letter Y (the "walking Y" in the Teradata logo).

Both Ynets operate concurrently to share the communication load. In contrast to an ordinary bus, which contains no logic, each Ynet is an array of active logic that provides routing and sorting functions. Each of the Ynets is independent of the other. This independence extends to physical

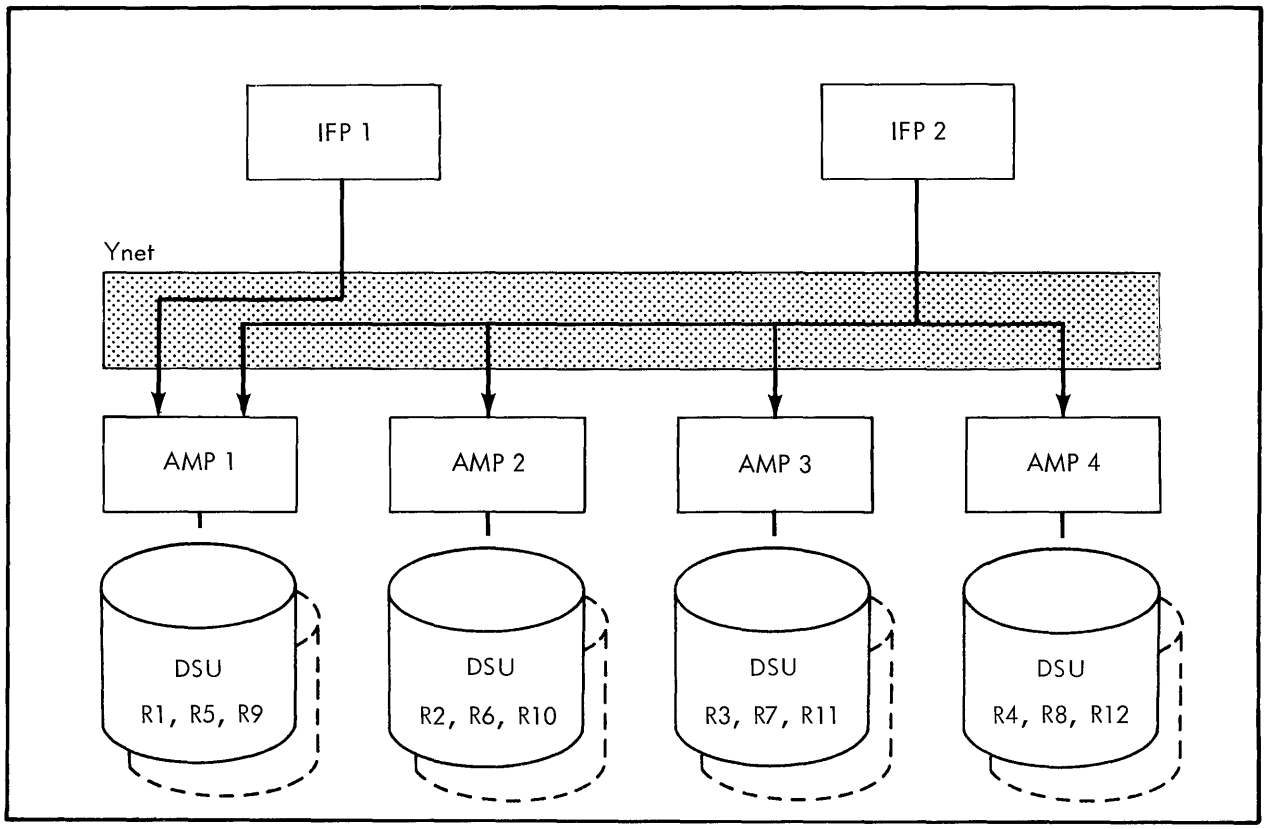

Figure 2-4. IFP Routing of DBC/SQL Request Messages

partitioning, electrical power, internal clocks, and interface logic. Each Ynet thus serves as backup for the other, ensuring continuous system operation in the event of a failure of either.

Each Ynet provides communication:

- Between single processors (e.g., between an IFP and an AMP).
- From a single processor to a group of processors (e.g., from one COP to all AMPs).
- From a group of processors to a specific processor (e.g., from all AMPs to one IFP).
- Between various processors to synchronize operations on data.

Request steps from an IFP or COP travel across the Ynet to the appropriate AMP(s). As data returned from the AMP(s) is merged back across the Ynet to the originating IFP or COP, it is automatically sorted into the sequence specified by the user's request. In this situation, the Ynet acts as an ultra-high-speed sort/merge processor.

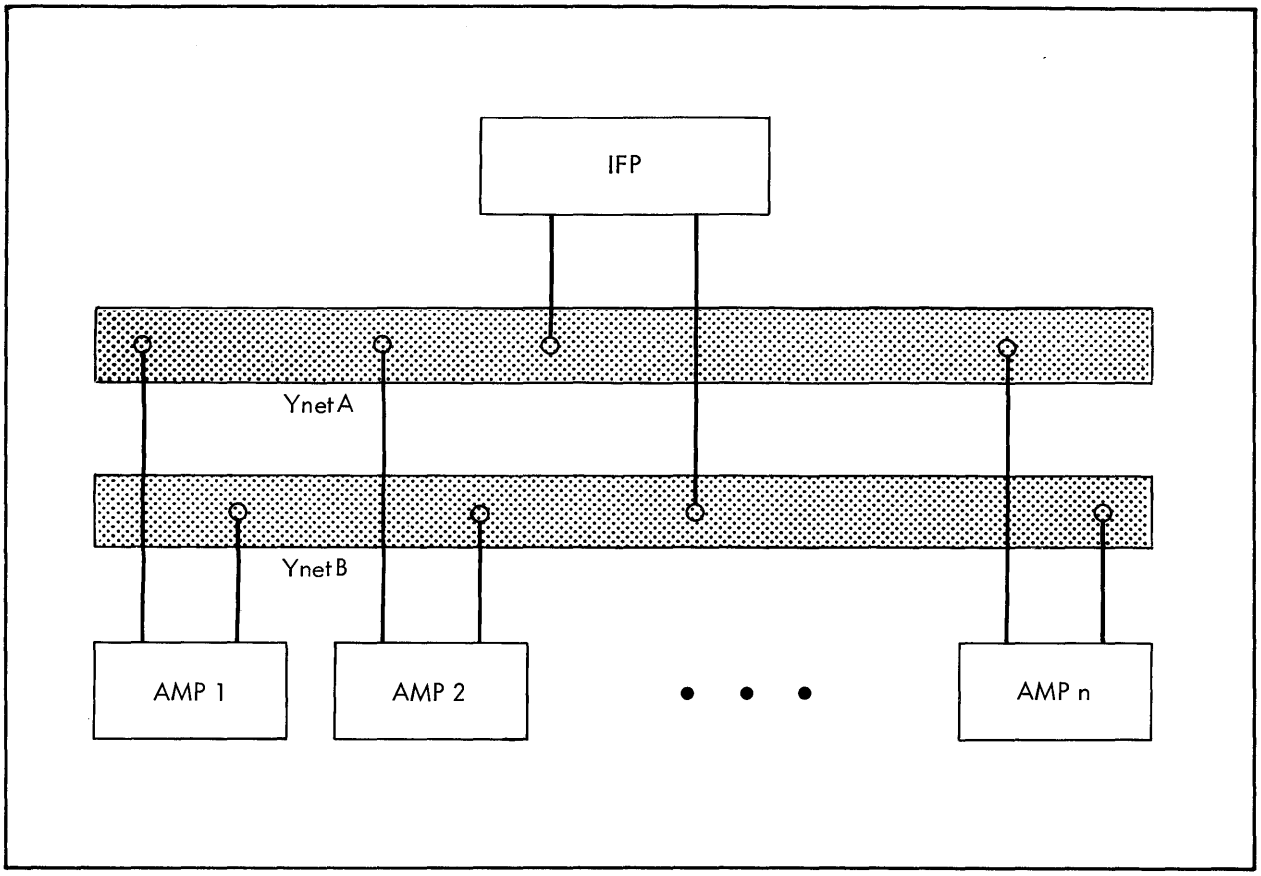

Figure 2-5. Ynet

ACCESS MODULE PROCESSOR (AMP)

The Access Module Processor (AMP) handles the actual data base functions of the DBC/1012 Data Base Computer. An AMP responds to IFP request steps received across the Ynet by selecting data from or storing data on its DSU(s). Figure 2-6 shows the components of the AMP. They are:

- Ynet Interface
- Data Base Manager
- Disk Interface

The Ynet Interface accepts messages from the IFP or COP, and returns responses. The interface also participates in system-wide synchronization of an operation that involves many AMPs.

The Data Base Manager services select, insert, delete, and update operations against existing tables. Other functions of the Data Base Manager include data definition, rollback and recovery, data base reorganization, and logging.

The Disk Interface passes commands to and receives responses from the attached disk storage unit(s).

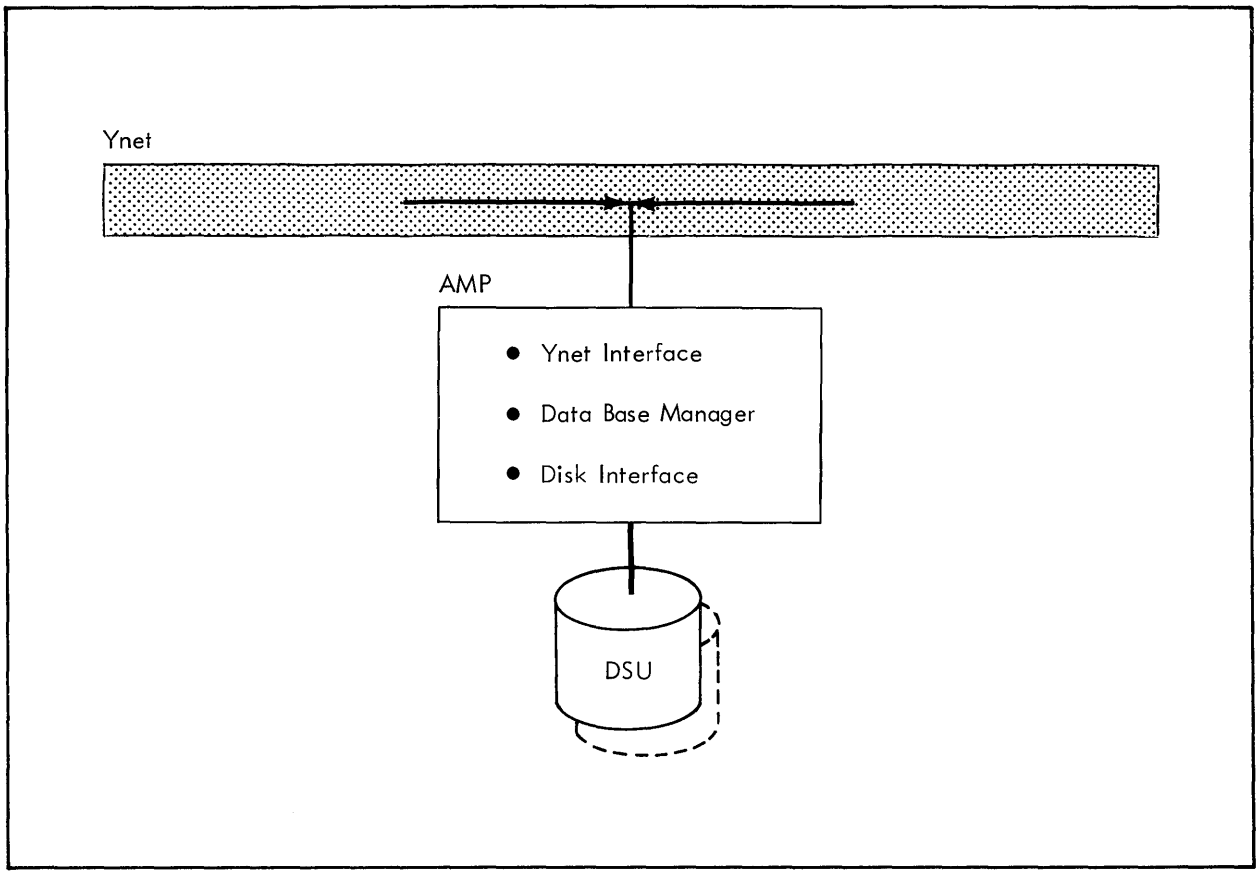

Figure 2-6. AMP

DISK STORAGE UNIT (DSU)

The Disk Storage Unit (DSU) is the data storage medium for the DBC/1012 Data Base Computer. Figure 2-7 shows an AMP and its DSU. As indicated in the figure, either one or two DSUs can be attached to an AMP.

The disk space on a DSU can be divided conceptually into three areas:

1. System area
2. Primary copy area
3. Fallback copy area (optional)

The system area stores system programs and tables. The primary copy area stores the primary copy of the data. The primary copy area is accessed under normal conditions. In general, as rows are inserted into a table, the data is evenly distributed among the DSUs of all AMPs. Processing a request for data from a table can thus be performed in parallel; that is, the data can be accessed simultaneously on all AMPs.

This simultaneous access greatly increases system throughput (imagine having many warehouse pickers rather than just one assembling a rush order). The result of this parallel multi-processing capability is

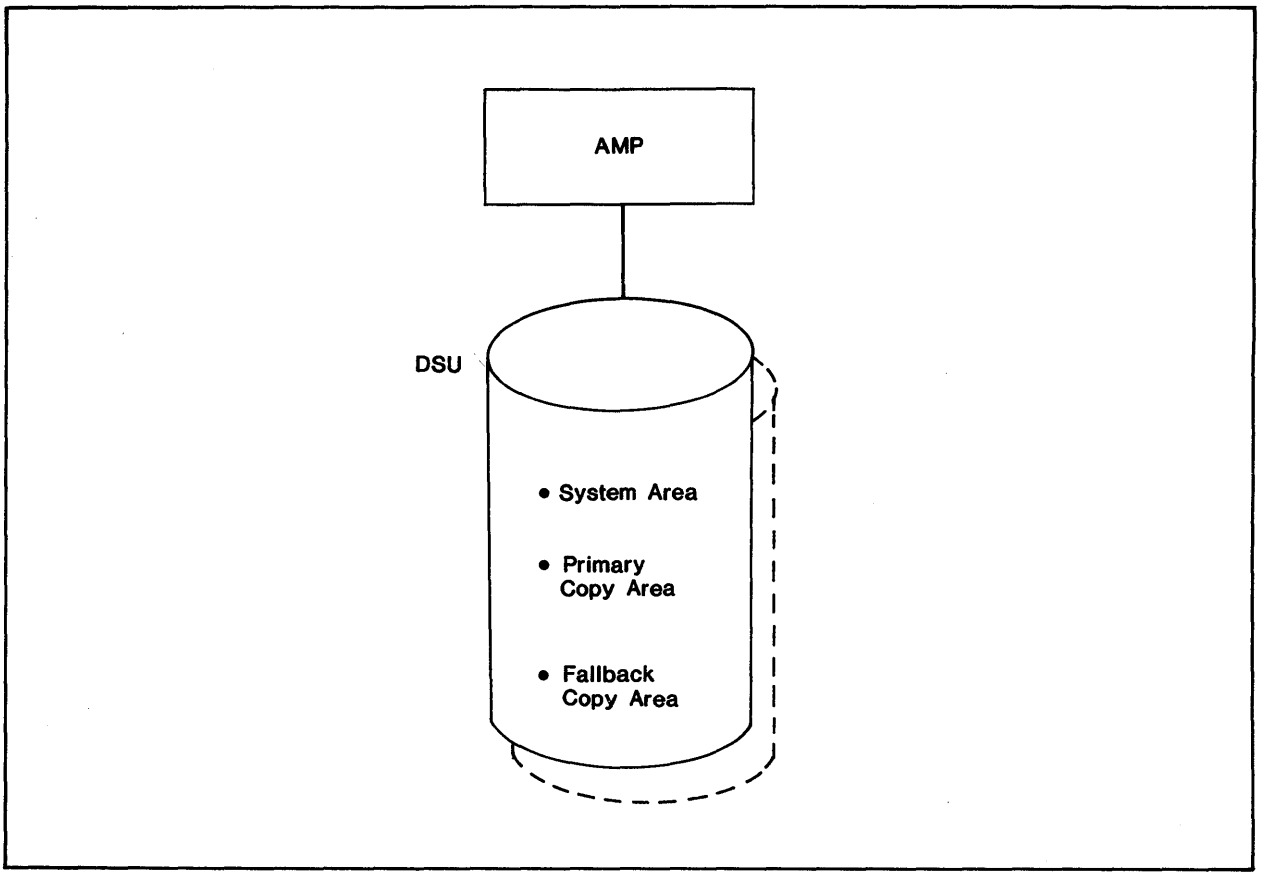

Figure 2-7. Disk Storage Unit

that, as DSUs and AMPs are added, storage capacity is increased while performance improves -- a unique situation for a DBMS.

The fallback copy area stores a secondary copy of data on other AMPs within the system. The fallback copy is accessed when the primary copy is temporarily unavailable.

When a data base is created, or any time after a table has been created, its creator can specify that a fallback copy be created. Like the primary copy of table data, this fallback copy is also distributed among other AMPs in the system. However, the fallback copy of any row is stored on a different AMP from that on which the primary copy is stored. This distribution ensures that, if an AMP fails, a fallback copy of any of its stored data remains available on other AMPs. For example, in the eight-AMP system illustrated in Figure 2-8, the primary copy of data on AMP 4 is distributed for fallback on AMPs 5, 6, and 7. If AMP 4 were to fail, its data could be accessed on these other AMPs.

On the other hand, if AMPs 4 and 7 were to fail simultaneously, there would be a loss of data. For this reason, the system administrator may provide additional data availability by "clustering" AMPs in groups of two to sixteen. Clustering increases the probability that all system data is available even if two or more AMPs fail simultaneously.

In the example shown in Figure 2-9, if AMPs were grouped in two clusters of four, the data on AMP 4 would be backed up on AMPs 1, 2, and 3 and the data on AMP 7 by AMPs 5, 6, and 8. If both AMPs 4 and 7 were now to fail simultaneously, all rows would still remain available.

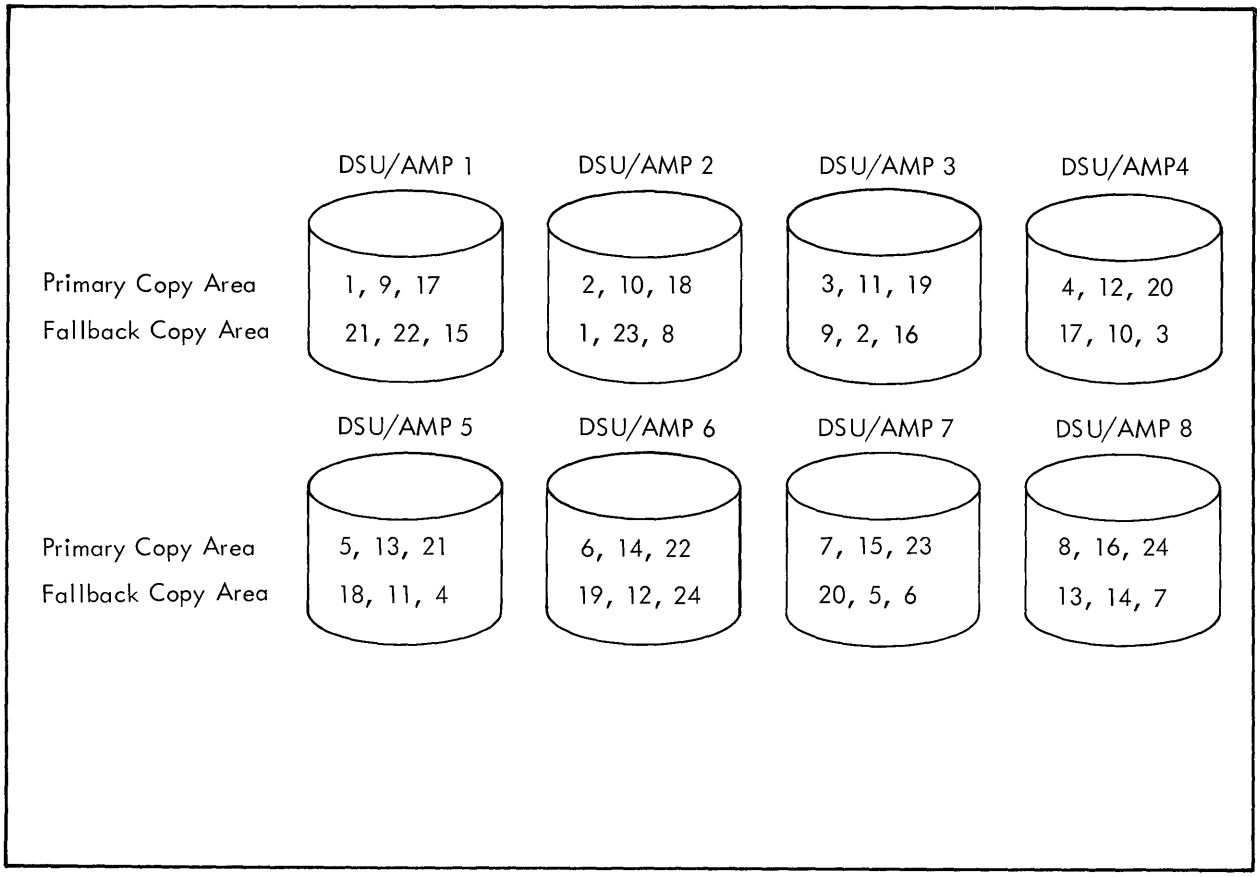

Figure 2-8. Distribution of Primary and Fallback Data

COMMUNICATIONS PROCESSOR (COP)

The COP enables users to access the DBC/1012 from a LAN. The COP functions very much like an IFP, performing session management, parsing, and dispatching functions. The COP, however, does this for a LAN rather than for a host. The same LAN may have more than one DBC/1012 connected to it, and a single DBC/1012 may be attached to more than one LAN. See Figure 2-10, Workstation Operating Environment for an illustration of the COP/LAN relationship.

SYSTEM CONSOLE WITH PRINTER

The System Console is an intelligent workstation (a personal computer) that can be connected directly to any IFP or AMP in a DBC/1012 system.

Through this connected processor, an operator can communicate with all processors in the system via the Ynet. The console consists of a keyboard and a screen (with a 24-line, 80-character display area), a diskette drive, and a printer.

The console can display system status, current configuration, and performance statistics.

The console printer is a bidirectional dot matrix printer that features a full 96-character ASCII set with upper- and lowercase characters. The printer provides a printed record of everything that is displayed at the console during a session with the DBC/1012.

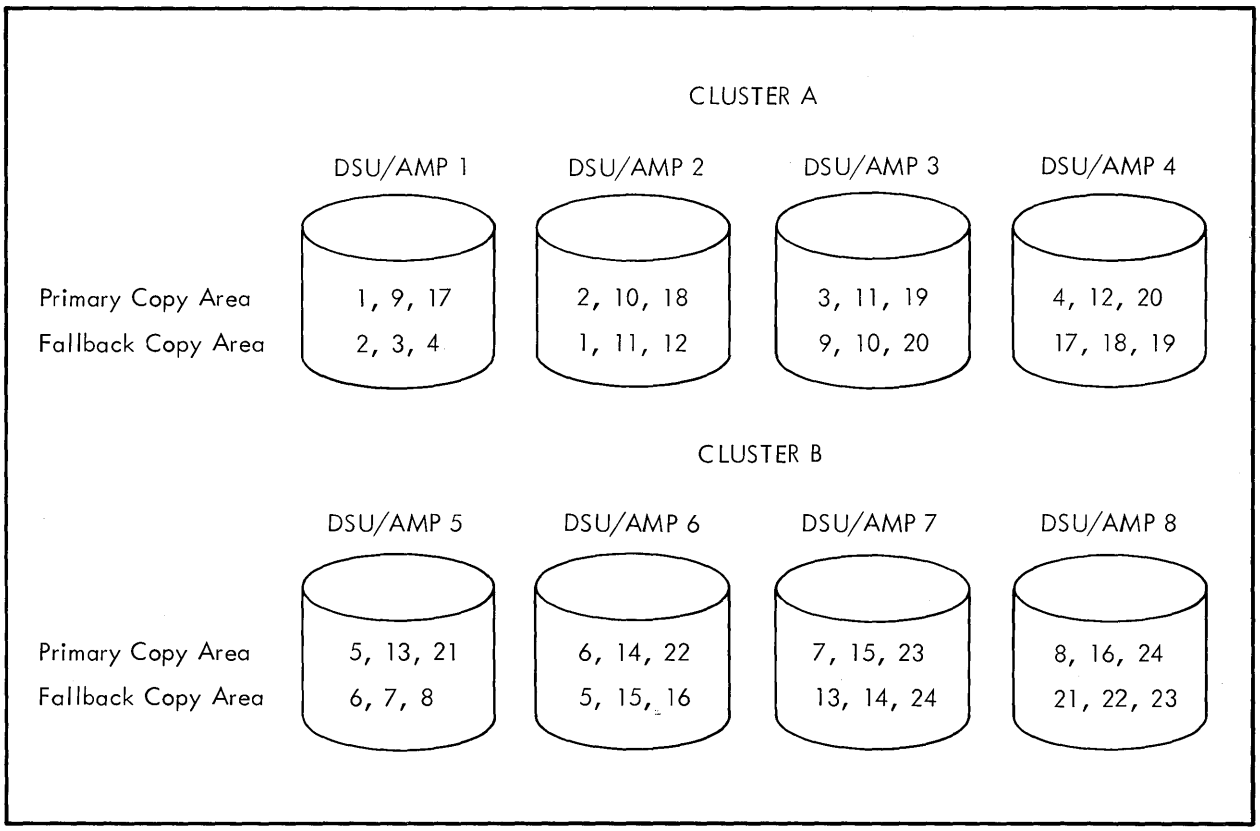

Figure 2-9. Distribution of Data With Clustered AMPs

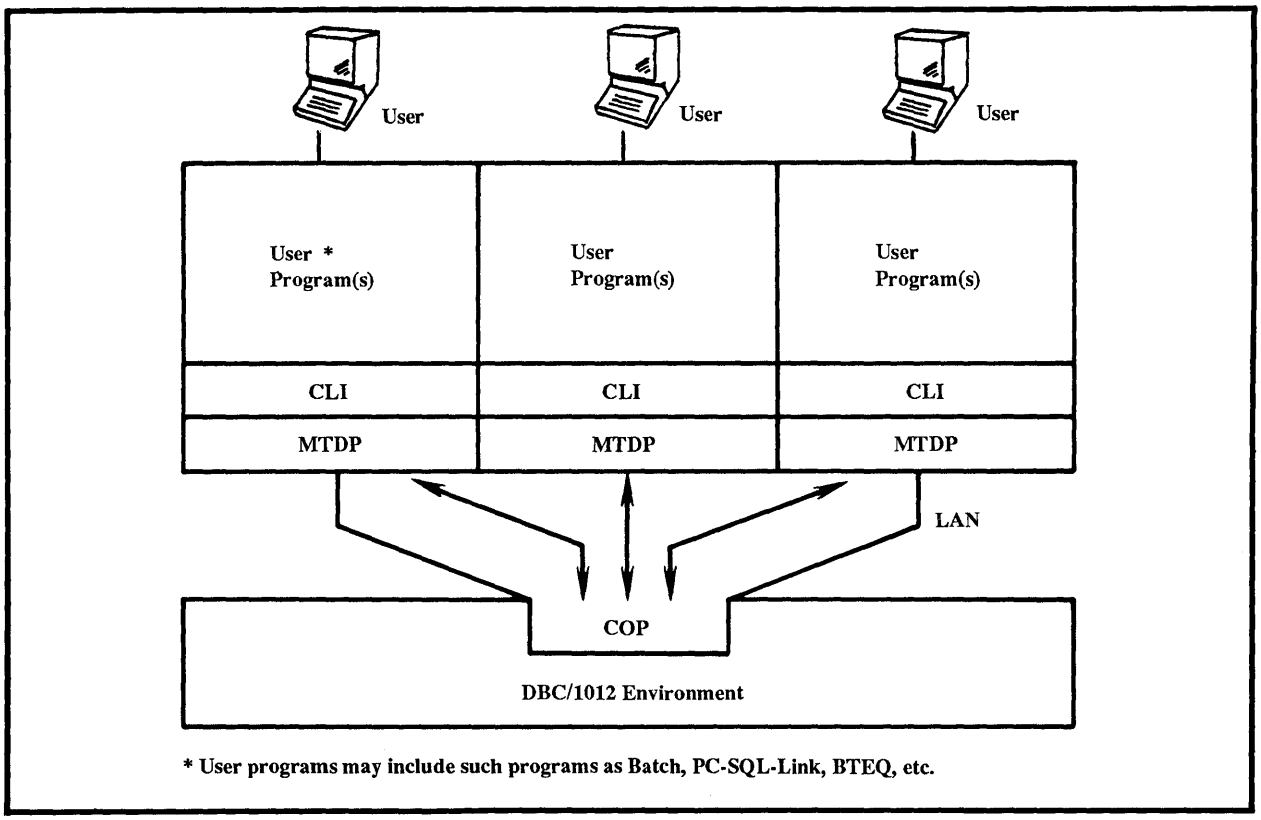

Figure 2-10. Workstation Operating Environment

SUMMARY AND PREVIEW

This chapter described the hardware and software subsystems that comprise the DBC/1012 Data Base Computer. The next chapter presents the facilities that allow an end user or application program to define, and manipulate data on the DBC/1012.

CHAPTER 3 USER FACILITIES

The DBC/1012 system provides the following facilities that allow an end user or application program to define and manipulate data:

- DBC/SQL, the Teradata query language, a single, unified, multipurpose data base language.
- ITEQ, the Interactive TEradata Query facility, which allows an 3270 end user (as opposed to an application program) to operate on data in a DBC/1012.
- BTEQ, the Batch TEradata Query facility, which allows a user to execute in batch a script that consists of a number of DBC/SQL statements and BTEQ commands.
- Coordinated products: Intellect, PC/SQL-link, NOMAD2, etc.
- Data Dictionary/Directory, a fully integrated, active dictionary/directory that gives users access to information about DBC/1012 objects through a series of system views.
- Language preprocessors, which allow DBC/SQL statements to be embedded in application programs written in COBOL or PL/1.
- Call-Level Interface, which enables application programs written in high-level languages that have a CALL statement to interact with the DBC/1012 Data Base Computer.

These facilities are discussed in this chapter.

DBC/SQL

The DBC/1012 system supports a single, easy-to-learn, English-like language, DBC/SQL. DBC/SQL can be used for:

- Defining data: creating and modifying tables and views.
- Manipulating data: selecting, inserting, deleting, and updating data.
- Using macros: storing and executing sequences of DBC/SQL statements.
- Controlling data: defining data bases and users, establishing access rights, and securing data.
- Creating archive copies of data bases.
- Loading data.

The last three points are described in Chapter 4. A complete presentation of all DBC/SQL capabilities is provided in the DBC/1012 Reference Manual.

Operations on Tables

As discussed in Chapter 1, basic operations on tables include: creating and deleting tables, adding new columns, selecting data, updating fields, inserting and deleting rows, copying data from one table to another, and creating "virtual" tables called views. With a select operation, a user can select data from entire tables, specific rows, specific columns (or specific rows and columns), from one or more tables.

Data Types

When a table is created, a "data type" attribute is specified for every column. DBC/SQL supports the following data types:

- INTEGER: a 32-bit signed, binary whole number that may vary in size from -2,147,483,648 to 2,147,483,647.
- SMALLINT: a 16-bit signed, binary whole number that may vary in range from -32,768 to 32,767.
- BYTEINT: an 8-bit signed, binary whole number that may vary in range from -128 to 127.
- DECIMAL [(n[,m])]: a packed decimal number of n digits, with m of these digits to the right of the decimal point. The precision value n may range from 1 to 15, the scale value m from 0 to n.
- FLOAT: a 64-bit floating point number in sign-and-magnitude form. This numeric data type is useful for representing values that have a large range, from (plus/minus) 4 x 10^{-307} to (plus/minus) 2 x 10^{308}.
- CHAR(n): a fixed-length character string of n characters. The maximum value for n is 32,000 characters.
- VARCHAR(n): a variable-length character string of maximum length n. The maximum value for n is 32000.
- LONG VARCHAR: the longest variable-length character string: VARCHAR(32000).
- BYTE(n): a fixed-length binary string of n bytes. The maximum value for n is 32,000.
- VARBYTE(n): a variable-length binary string of maximum value n. The maximum value for n is 32,000.
- DATE: a 32-bit integer that represents the date as YYMMDD. DBC/1012 users can request a date value to be formatted as a combination of characters (D=day, M=month, Y=year, B=blank) and special characters (/ , ' - . :) that represent a calendar date.

In addition to these data types, the user can specify the following:

- NULL: a value need not be specified for the field.
- NOT NULL: a value for the field must be included.
- COMPRESS: the space occupied by one or more columns of a table may be compressed to zero space for a given value.
- BETWEEN n AND n: a range constraint may be specified for the INTEGER SMALLINT, BYTEINT, DECIMAL, FLOAT, and DATE data types.
- CASESPECIFIC: specifies that data be stored as it is entered, and that data not be converted to uppercase during comparison operations.
- UPPERCASE: used when a field is defined in a CREATE TABLE statement to specify that data be stored in uppercase regardless of the case in which it is entered.

- FORMAT: used in a CREATE TABLE statement to specify the display format of column data. Formats may be specified for character, numeric, or DATE data.

- TITLE: used during table creation to specify a title for displayed or printed results other than the column name used by default.

- DEFAULT: supplies a value for a field when a value is not specified in an INSERT statement.

See the CREATE TABLE statement later in this chapter for an example of how different data types are used.

DBC/SQL Statements

Requests for operations on data base tables are expressed as DBC/SQL statements. Statements are structured according to rules and conventions of the DBC/SQL language syntax.

A DBC/SQL statement consists of a keyword verb, optional clauses introduced by keywords, and expressions. A statement is terminated by a semicolon. Structured use of keyword verbs and keyword clauses permits exact specification of any request.

For example, in the following DBC/SQL statement

```
SELECT Name, Salary, JobTitle
FROM Employee
WHERE DeptNo = 100;
```

SELECT is the keyword verb; FROM is a keyword that identifies the table that contains the information; WHERE is a keyword that introduces a conditional clause; and Name, Salary, JobTitle, and DeptNo = 100 are expressions.

Names

A name consists of a word used in a DBC/SQL statement to label and refer to a data base, table, column, or other entity. A name can contain up to 30 characters. Characters in a name can be chosen from uppercase and lowercase letters; digits 0 through 9; and special characters ($ #_). Words may not begin with a digit.

Examples of valid names are:

```
Employee
R2D2
DeptName
```

The following are examples of invalid names:

```
Department Name
1980_Population
current_population_of_Manhattan_Beach
```

The first is invalid because a name cannot contain a blank space. The second is invalid because a name cannot begin with a digit. The third is invalid because it is longer than 30 characters.

An important feature of DBC/SQL is its case-independence. Regardless of the case in which a name is defined (uppercase, lowercase, or a combination of the two), DBC/SQL recognizes the name as long as it is spelled correctly.

Keywords

Words having special significance in DBC/SQL statements are called "keywords". Keywords are reserved words and may not be used in names. Some keywords are:

```
SELECT      UPDATE      AND
DELETE      WHERE       TABLE
```

A keyword can be made part of a name by using the underscore character or other special character to connect it with another word.

Delimiters

Delimiters are special characters that are used to group, separate, and define items in DBC/SQL statements. The following characters are used as delimiters:

```
( )  ,  :  ;  '  "  @  .
```

For example, the statement

```
INSERT INTO Employee (EmpNo, Name, DeptNo)
VALUES (10015, 'Goldsmith H', 600);
```

contains four kinds of delimiters (parentheses, commas, apostrophes, and the semicolon).

Separators

Separators are used to separate words, constants, and delimiters in a DBC/SQL statement without changing the meaning of the statement. One or more blanks (spaces), a comment (explanatory text), and the semicolon terminator are common separators. For example, in the INSERT statement above, space and commas are used as separators.

Constants

Constants are numeric or character values of fixed length that are embedded in a DBC/SQL statement. Numeric constants may be expressed in integer form (1 to 15 digits with optional sign), decimal form (1 to 15 digits with decimal point and optional sign), or floating form (a signed integer or decimal constant, followed by the letter E, followed by an integer representing a power of ten).

Examples of numeric constants are:

Integer	Decimal	Floating
0	0.0	0E0
15	15.0	1.5E1
	.51	5.1E-1
687	687.00	687E0

Character string constants consist of one or more characters of any kind, including blanks, enclosed by matching apostrophes (' '). An apostrophe can be included within a character string constant if the apostrophe within the string is doubled.

The following are examples of valid character string constants:

```
'Terminator'    'Oh, well!'
'Don''t'        'Now is the time to do it.'
'Marston A'     'Los Angeles'
```

Referencing Data in DBC/SQL

The DBC/1012 Data Base Computer can accommodate many data bases, each consisting of one or more related tables. A table can have one or more columns.

In a DBC/SQL statement, each of these entities -- data base, table, column -- is identified by a unique name. That is, each table name within a data base and each column name within a table must be unique. To refer to a particular column without ambiguity, the user specifies the data base and table names in addition to the column name as follows:

```
databasename . tablename . columnname
```

The period is the delimiter between data base, table, and column names.

A column reference in this form is considered a fully qualified name. For example, if the table in Figure 1-3 were named Employee and were part of a data base named Personnel, the fully qualified name for the JobTitle column could be:

```
Personnel.Employee.JobTitle
```

To refer to all columns simultaneously, the user uses an asterisk (*) in place of a field name. For example, the statement

```
SELECT * FROM Employee;
```

retrieves all columns in the Employee table.

A fully qualified name is not always necessary to specify a column in a DBC/SQL statement. DBC/SQL permits omission of a data base name and table name that occur elsewhere in the statement if no ambiguity results. In addition, when a user logs on to the DBC/1012, a default data base name is established as the user's "user identification" (username).

Thus, unless the user specifies another default data base after logon, all statements in the session (other than fully qualified references) apply to the data base "username".

To improve clarity in the DBC/SQL examples in this chapter, all keywords are shown in uppercase letters, while name references to data are shown in lowercase letters with initial capitals This convention is illustrated in the SELECT and INSERT examples above.

DBC/SQL Expressions

Expressions in DBC/SQL statements allow a user to perform arithmetic and logical operations, to generate new values or Boolean results from constants and stored values, and to select results that meet certain criteria. An expression can consist of a column name, a constant, or a combination of column names and constants connected by operators. The simplest expression is a column name. More complex expressions can be formed by connecting column names or expressions with operators.

Arithmetic Operators

DBC/SQL supports the standard arithmetic operations of addition, subtraction, multiplication, and division. The following operators are used:

Operator	Meaning
+	addition
-	subtraction
*	multiplication
/	division
MOD	modulus
**	exponentiation

Arithmetic operations may involve a mix of numeric types -- integer, decimal, and floating-point.

The data type of the result of an arithmetic expression is a function of the data types of the two operands. Operands are converted to the result type before the operation is performed. For example, before an INTEGER value is added to a DECIMAL(5,2) value, the INTEGER value is converted to DECIMAL(15,2), the data type of the result.

Aggregate Operators

The following aggregate operators may be used to define a query result:

Operator	Meaning
AVERAGE	Provides the average of a set of values
COUNT	Provides a count of the members of a set
MAXIMUM	Provides the maximum value in a set of values
MINIMUM	Provides the minimum value in a set of values
SUM	Provides the sum of a set of values

An aggregate function computes the values in a group. The group may consist of either all the values in a particular column or a subset of column values.

Comparison Operators

Comparison operators compare numeric or character values to produce a logical (true or false) result. The following operators may be used:

Operator	Meaning
= or EQ	equal
> or GT	greater than
< or LT	less than
<= or LE	less than or equal to
>= or GE	greater than or equal to
BETWEEN...AND	range

Logical Operators

To combine logical expressions and generate compound conditions, the keywords AND, OR, and NOT can be used in the WHERE clause of a SELECT statement. The " ¬ " symbol may be used as an alternative for NOT. Parentheses define groupings and precedence. In the absence of these, rules of grouping and precedence apply.

Partial String Matching Operators

The LIKE operator is used in an expression to locate character strings that match portions of or complete strings of characters, as described in the following table:

```
Expression           Meaning
-----------------    ----------------------
LIKE charstring%     Begins with charstring
LIKE %charstring     Ends with charstring
LIKE %charstring%    Contains charstring
```

The result of these operations is either true or false. The NOT operator may be used with LIKE to form the negative of an expression. For example, NOT LIKE charstring%

means "does not begin with charstring".

Set Operators

The set operators are as follows:

- IN
- NOT IN
- UNION
- INTERSECT
- MINUS

These operators are used in a conditional expression. The operators test whether one or more values are within a defined set of values. Sets may be expressed as a list of constants or a single-column table. The conditional forms are:

```
value IN (constant, constant,...)
value NOT IN (constant, constant,...)
(valuel [,value2,... ]) IN (SELECT statement)
(valuel [,value2,... ]) NOT IN (SELECT statement)
```

In the first form, the value belongs to one of the set defined within the parentheses; in the second form, the value does not belong to one of the set. In the last two forms, the set is expressed as the result of a data base query operation.

Other Operators

DBC/SQL also provides a concatenation operator and string functions for working with character data.

Arithmetic Functions

DBC/SQL provides the following arithmetic functions that require numeric arguments (arg):

```
Function            Result
----------------    ----------------------------

ABS (arg)           Absolute value
EXP (arg)           Raises e to the power of arg
                    (e ** arg)
LOG (arg)           Base 10 logarithm
LN (arg)            Base e (natural) logarithm
SQRT (arg)          Square root
NULLIFZERO (arg)    Returns NULL if arg is zero
ZEROIFNULL (arg)    Returns zero is arg is NULL
```

(The "e" included in the Result for EXP and LN is the base value for natural logarithms, 2.71828182845904.)

Arithmetic functions have the following result data types:

```
Function    Result Data Type
----------  ----------------

ABS         Same as arg
EXP         FLOAT
LOG         FLOAT
LN          FLOAT
SQRT        FLOAT
NULLIFZERO  Same as arg
ZEROIFNULL  Same as arg
```

Defining Data

Within data bases, tables are used to contain data. DBC/SQL data definition statements enable a user to:

- Create tables
- Add and remove table columns
- Create secondary indexes on tables
- Create views of one or more tables, or combinations of views and tables
- Remove tables, indexes, and views

Tables are defined using the CREATE TABLE statement. CREATE TABLE specifies a table name, column names, and column attributes.

The tables shown in Figures 3-1 and 3-2 represent a sample data base named Personnel, which will be used in subsequent examples of DBC/SQL statements. The first table is named Employee and has six columns; the second table is named Department and has five columns.

Table: Employee

EmpNo	Name	DeptNo	JobTitle	Salary	YrsExp
10001	Peterson J	100	Payroll Ck	25,000.00	5
10002	Moffit H	100	Recruiter	35,000.00	3
10003	Leidner P	300	Secretary	23,000.00	13
10004	Smith T	500	Engineer	42,000.00	10
10005	Omura H	500	Programmer	40,000.00	8
10006	Kemper R	600	Assembler	29,000.00	7
10007	Aguilan J	600	Manager	45,000.00	11
10008	Phan A	300	Vice Pres	55,000.00	12
10009	Marston A	500	Secretary	22,000.00	12
10010	Reed C	500	Technician	30,000.00	4
10011	Chin M	100	Controller	38,000.00	11
10012	Watson L	500	Vice Pres	56,000.00	8
10013	Regan R	600	Purchaser	44,000.00	10

Figure 3-1. Table of Employee Data

Table: Department

DeptNo	DeptName	EmpCount	Loc	MgrNo
100	Administration	4	NYC	10004
300	Exec Office	1	NYC	10012
500	Engineering	5	ATL	10008
600	Manufacturing	3	CHI	10009

Figure 3-2. Table of Department Data

The following CREATE TABLE statement is used to create the Employee table:

```
CREATE TABLE  Employee, FALLBACK
(EmpNo INTEGER,
 Name  VARCHAR(12),
 DeptNo SMALLINT,
 JobTitle VARCHAR(12),
 Salary DECIMAL (8,2),
 YrsExp BYTEINT)
UNIQUE PRIMARY INDEX (EmpNo)
INDEX (Name);
```

The following CREATE TABLE statement is used to create the Department table:

```
CREATE TABLE  Department, FALLBACK
(DeptNo SMALLINT,
 DeptName VARCHAR(14),
 EmpCount INTEGER,
 Loc CHAR(3),
 MgrNo INTEGER)
UNIQUE PRIMARY INDEX (DeptNo);
```

Table column definitions follow the name of the table being created. Each column is identified by a name and one or more attributes. Attributes assigned to a column must include a data type that specifies how data is represented externally (for example, INTEGER, DECIMAL, DATE, etc.).

Optional attributes may be specified by the following phrases:

- FORMAT phrase

  Specifies how column data is to be displayed.

- TITLE phrase

  Specifies a display title for a column.

- DEFAULT phrase

  Specifies a constant value that is supplied automatically if a specific value is not supplied when the row is inserted.

- COMPRESS phrase

  Saves storage space on the DBC/1012 by compressing space in a table.

- NULL phrase

For the Employee table, the EmpNo column is defined as a primary index using the UNIQUE PRIMARY INDEX clause. Next, Name is defined as a secondary index using the INDEX clause. DeptNo is designated as a primary index for the Department table. A more detailed discussion of indexes is found below.

After a table is created, rows can be added to it using the INSERT statement, described below. Because the user has specified the FALLBACK option for both tables, a fallback copy of the rows of each table is distributed among the Disk Storage Units of the AMPs in the DBC/1012 Data Base Computer.

Adding Columns to a Table

The ALTER TABLE statement allows a user to change the structure of an existing table by adding or deleting columns and the attributes assigned to them. The statement may also be used to change the FALLBACK option for a table, and the FORMAT, TITLE, and DEFAULT attributes for existing columns.

To add a column called Budget to the Department table, the following statement is used:

```
ALTER TABLE Department ADD Budget (DECIMAL(9,2));
```

The Department table now looks like this:

Table: Department

DeptNo	DeptName	Loc	MgrNo	Budget
100	Administration	NYC	10004	
300	Exec Office	NYC	10012	
500	Engineering	ATL	10008	
600	Manufacturing	CHI	10009	

Note that the new column contains null values. Values are entered in the Budget column using the UPDATE statement, described below in the section "Changing Rows". To drop the Budget column, the following statement is entered:

```
ALTER TABLE Department DROP Budget;
```

Creating Indexes

In general, table rows have no inherent order: DBC/SQL allows a user to select rows in any order. To speed up selection of rows, however, one or more columns may be designated as indexes for the table.

A primary index must be defined during table creation. Secondary indexes may also be defined during table creation. Secondary indexes may, however, be defined after usage patterns have been determined for an existing table. In this case, the CREATE INDEX statement is used to add a secondary index on one or more columns.

For example, many queries of the Employee table may reference both the DeptNo and JobTitle columns (for example, SELECT Name FROM Employee WHERE DeptNo = 500 AND JobTitle = 'Engineer';). To establish a secondary index on these two columns, the following statement is used:

```
CREATE INDEX (DeptNo, JobTitle) ON Employee;
```

In the CREATE TABLE or CREATE INDEX statements, the UNIQUE keyword, for example,

```
CREATE UNIQUE INDEX (EmpNo) ON Employee;
```

specifies that no two rows in the table can have the same value for the index column.

Creating Views

A view is a "window" through which a user can see selected portions of a data base. A view may be used to display portions of one or more tables or combinations of views and tables, and to update, insert, and delete table rows.

Views look like stored tables; they have rows and columns and, in general, may be used as if they were tables. However, some table operations are not valid on views and some operations are restricted, depending on how the view is defined.

Views are typically used to simplify query requests for data from a larger table or from several tables, to limit users' access to or manipulation of table data, and to change users' perception of table data. Views allow a user to:

- Rename Tables or Columns

  Like a table, a view based on a table or tables may have its own unique name and arbitrary names for its columns.

- Reorder Table Columns

  The columns of a view may be defined in any order, independent of the tables on which the view is based.

- Select Table Columns

  A view may be defined to contain only specific columns. Access to the view limits other users' access to these columns only.

- Select Table Rows

  A view may include only certain rows selected from a table or tables.

- Join Table Columns

  A view may contain columns from more than one table. In this case, the rows of the view are the result of a join on the tables involved.

- Include Virtual Columns

  A virtual column may be defined for a view using arithmetic expressions to compute column data from the underlying tables.

A view is defined with the CREATE VIEW statement. In CREATE VIEW, the view and its columns are named and columns and rows defined. For example, a personnel clerk needs access to employee names, positions, and departments, but not to sensitive information such as salary. The statement

```
CREATE VIEW  Employee_Info
(Employee, JobTitle, Department)
AS SELECT Name, JobTitle, Department.DeptName
FROM Employee, Department
WHERE Employee.DeptNo = Department.DeptNo;
```

creates a view named Employee_Info that permits a user to select the following information from both Employee and Department tables:

```
Employee      JobTitle    Department
------------  ----------  -------------
Peterson J    Payroll Ck  Administration
Moffit H      Recruiter   Administration
Leidner P     Secretary   Exec Office
Smith T       Engineer    Engineering
Omura H       Programmer  Engineering
Kemper R      Assembler   Manufacturing
Aguilan J     Manager     Manufacturing
Phan A        Vice Pres   exec Office
Marston A     Secretary   Engineering
Reed C        Technician  Engineering
Chin M        Controller  Administration
Watson L      Vice Pres   Engineering
Regan R       Purchaser   Administration
```

A view may be modified using the REPLACE VIEW statement. For example, the statement

```
REPLACE VIEW  Employee_Info
(Number, Employee, Position, Department)
AS SELECT EmpNo, Name, DeptName
FROM Employee, Department
WHERE Employee.DeptNo = Department.DeptNo;
```

replaces Employee_Info with a view that includes an employee number column. If the Employee_Info view does not exist, the statement creates the new view.

Removing Tables, Indexes, and Views

When a table, secondary index, or view is no longer needed, it can be removed using a form of the DROP statement. For example, the statements

```
DROP TABLE Employee;
DROP INDEX (Name) ON Employee;
DROP VIEW Employee_Info;
```

delete the Employee table, the secondary index on the Name column in the Employee table, and the Employee_Info view.

Selecting Data

Selecting data is one of the most important operations of the DBC/1012 Data Base Computer. Using the DBC/SQL SELECT statement, the user can:

- List all or parts of a table
- Sort or sequence data
- Combine data from more than one table

- Perform arithmetic operations on the data
- Transform the result using aggregate operators, comparison operators, logical operators, etc.

Making Simple Queries

The simplest form of the SELECT statement can be used to obtain all the data from a table. For example,

```
SELECT * FROM Employee;
```

causes the entire Employee table (Figure 3-1) to be returned. The * denotes "all columns" of the table.

Selecting Specific Columns

Columns to be selected from a table are named in the SELECT statement. For example, to select only the names, salaries, and positions of employees, the following query is used:

```
SELECT Name, Salary, JobTitle
FROM Employee;
```

This query produces the following result:

Name	Salary	JobTitle
Peterson J	25,000.00	Payroll Ck
Moffit H	35,000.00	Recruiter
Leidner P	23,000.00	Secretary
Smith T	42,000.00	Engineer
Omura H	40,000.00	Programmer
Kemper R	29,000.00	Assembler
Aguilan J	45,000.00	Manager
Phan A	55,000.00	Vice Pres
Marston A	22,000.00	Secretary
Reed C	30,000.00	Technician
Chin M	38,000.00	Controller
Watson L	56,000.00	Vice Pres
Regan R	44,000.00	Purchaser

The order of the column names in the SELECT statement (rather than their order in the stored table) determines the left-to-right display of columns in the output. If no ORDER BY clause is entered, the rows are returned in a system-determined order. Details about the ORDER BY clause are provided below.

Selecting Specific Rows

The WHERE clause is used to select specific rows from a table. For example, to get the name, salary, and position (as in the example above) of employees in department 100 only, the query

```
SELECT Name, Salary, JobTitle
FROM Employee
WHERE  DeptNo = 100;
```

produces the following result:

```
Name              Salary  JobTitle
------------  ----------  ----------
Peterson J     25,000.00  Payroll Ck
Moffit H       35,000.00  Recruiter
Chin M         38,000.00  Controller
```

The WHERE clause can include various comparison and logical operators. Any column name can be used and compound selection criteria specified. For instance, the statement

```
SELECT Name, Salary
FROM Employee
WHERE DeptNo IN (100, 500) AND Salary > 40000;
```

produces a list of the names and salaries of employees in departments 100 and 500 who earn more than $40,000 per year, as follows:

```
Name              Salary
------------  ----------
Smith T        42,000.00
Omura H        40,000.00
Aguilan J      55,000.00
Watson L       56,000.00
Regan R        44,000.00
```

In the WHERE clause of the SELECT statement, the IN set operator is used in place of the = comparison operator to specify the condition:

```
WHERE DeptNo = 100 OR DeptNo = 500
```

Specifying Order

The sequence in which selected data is returned is specified using the ORDER BY clause. The following statement causes the name and years of experience for each employee in department 600 to be listed in ascending order of seniority:

```
SELECT Name, YrsExp
FROM Employee
WHERE DeptNo = 600
ORDER BY YrsExp;
```

The result of this query is as follows:

```
Name          YrsExp
------------  ------
Kemper R           6
Aguilan J          6
Regan R            6
```

A number of fields can be listed with the sort direction (ASC or DESC) specified for each in the ORDER BY clause. Ascending order (ASC) is the default. The statement

```
SELECT DeptNo, Name, YrsExp
FROM Employee
ORDER BY DeptNo, YrsExp DESC;
```

lists the names and years of experience of all employees in department number order with the most senior people first, as follows:

```
DeptNo  Name          YrsExp
------  ------------  ------
   100  Chin M            11
   100  Peterson J         5
   100  Moffit H           3
   300  Leidner P         13
   300  Phan A            12
   500  Marston A         12
   500  Smith T           10
   500  Omura H            8
   500  Watson L           8
   500  Reed C             4
   600  Aguilan J         11
   600  Regan R           10
   600  Kemper R           7
```

Defining Groups

A table may be divided into groups according to the values in one or more columns. The aggregate operators described above may then be applied to values in each group to provide summary information about the group. The GROUP BY clause is used to define the group. When a GROUP BY clause is present in a SELECT statement, each item in the statement must be a unique property of the group. The HAVING clause, similar to the WHERE clause, may be specified to restrict the groups that are to appear in the query result.

For example, the statement

```
SELECT DeptNo, MIN(Salary), MAX(Salary)
FROM Employee
GROUP BY DeptNo
HAVING SUM(Salary) > 170000;
```

gives the minimum salary and the maximum salary for every department (group) whose salaries total more than $170,000.

Using More than One Table

A query may return data from more than one table. Such a query causes a "join" of the tables. The WHERE clause specifies the relationship on which the tables are to be joined. For example, the Employee table and the Department table may be joined. To list the names and locations of all employees, enter the following SELECT statement

```
SELECT Name, Loc
FROM Employee, Department
WHERE Employee.DeptNo = Department.DeptNo;
```

Nesting Subqueries

To determine who employee Marston's department manager is, three separate queries might be used:

```
SELECT DeptNo FROM Employee WHERE Name = 'Marston A';

SELECT MgrNo FROM Department WHERE DeptNo = 500;

SELECT Name FROM Employee WHERE EmpNo = 10008;
```

The answer to the first query, 500, provides the critical parameter for the WHERE clause of the second query. The answer to the second query, 10008, provides the critical parameter for the third.

Using the subquery feature of DBC/SQL, this type of question can be answered with one statement. The result of the first query can be referenced in the WHERE clause of the second query, and the result of the second query referenced in the WHERE clause of the third. The queries can be entered as if the questions were being asked in the reverse order:

```
SELECT Name FROM Employee
WHERE EmpNo IN
   (SELECT MgrNo FROM Department
    WHERE DeptNo IN
       (SELECT DeptNo FROM Employee
        WHERE Name = 'Marston A'));
```

The same result can be obtained using one level, as follows:

```
SELECT Name FROM Employee
WHERE EmpNo IN
   (SELECT MgrNo FROM Department, Employee
    WHERE  Employee.Name = 'Marston A'
    AND Department.DeptNo = Employee.DeptNo);
```

Manipulating Data

Besides selecting data, common data manipulation operations in DBC/SQL include adding rows to tables, modifying existing rows, and deleting rows.

Adding Rows to a Table

The INSERT statement adds new rows to a table. Several forms of this command may be used.

In the first form, the columns that are to receive the values are listed separately, enclosed by parentheses. The values that are to be added to these columns also are listed separately in parentheses, in the same left-to-right order as the columns for which they are intended, and preceded by the keyword VALUES.

For example, the following statement adds a new employee to the Employee table:

```
INSERT INTO Employee (Name, EmpNo, DeptNo, YrsExp)
VALUES ('Clarkson B', 10014, 600, 3);
```

Assuming that Clarkson's salary and position were not known when the INSERT was made, it would be added to the end of the Employee table as shown below.

EmpNo	Name	DeptNo	JobTitle	Salary	YrsExp
10013	Watson L	500	Vice Pres	56,000.00	8
10014	Clarkson B	600			3

The second form of the INSERT statement allows the field values simply to be listed. The left-to-right order of the list must correspond to that of the table's columns as defined in the CREATE TABLE statement. The list must allow a position for each column, whether or not data is entered in all columns. The position of a column is indicated by a comma.

For example, the statement

```
INSERT INTO Employee
VALUES (10015, 'Goldsmith H', 600,,, 5);
```

adds yet another employee to the Employee table.

Again, it is assumed that position and salary were not known at the time of insert. Note the use of commas in the list to indicate that these fields are not present.

The third form of INSERT uses an embedded SELECT statement to insert values from one table into another. For example, the statement

```
INSERT INTO Promotion
SELECT Name, DeptNo, YrsExp FROM Employee
WHERE YrsExp > 10;
```

inserts into a previously created table, Promotion, Employee information for employees having more than ten years of experience with the organization. Column values must be provided in the order in which columns are defined in the CREATE TABLE statement for Promotion.

Changing Rows

The UPDATE statement allows a user to modify existing columns in one or more rows of a table. For example, to enter position and salary data for employees Clarkson and Goldsmith, inserted above in the Employee table, the following UPDATE statements are used:

```
UPDATE Employee
SET JobTitle = 'Inspector', Salary = 32000
WHERE EmpNo = 10014;

UPDATE Employee
SET JobTitle = 'Assembler', Salary = 25000
WHERE EmpNo = 10015;
```

As a result of the additions to the Manufacturing department, the Department table must be updated. The following statement increments the employee count for this department by 2:

```
UPDATE Department
SET EmpCount = EmpCount + 2
WHERE DeptName = 'Manufacturing';
```

The following statement

```
UPDATE Employee
SET Salary = Salary * 1.1
ALL;
```

gives every employee a 10 percent raise.

Deleting Rows from a Table

The DELETE statement deletes one or more rows from a table. As in the UPDATE statement, a WHERE clause determines which rows are affected. Employee Smith T has left the company. The following statements

```
DELETE FROM Employee
WHERE Name = 'Smith T';

UPDATE Department
SET EmpCount = EmpCount - 1
WHERE DeptNo = 500;
```

are used to update the two tables.

Macros

A sequence of DBC/SQL statements may be defined as a macro and executed as a single transaction in order to perform a complex task. Macros are typically used to reduce the number of characters that must be entered to specify a particular operation, thus saving the user time and decreasing the chance of error. Macros are particulary important for enforcing data integrity rules and providing additional data security.

Macros are defined using the CREATE MACRO statement. For example, the following statement can be used to define a macro for adding new employees to the Employee table and incrementing the EmpCount field in the Department table:

```
CREATE MACRO NewEmp (name (VARCHAR(12),
    number (INTEGER, NOT NULL),
    dept (INTEGER, DEFAULT 100))
AS (INSERT INTO Employee (Name, EmpNo, DeptNo)
    VALUES (:name, :number, :dept);
    UPDATE Department SET EmpCount=EmpCount+1
    WHERE  DeptNo=:dept;);
```

This macro contains parameters that are filled in each time it is executed. Only constants can be used as parameters. In the macro, references to the parameter name are prefixed by the : character.

In the following statement,

```
EXECUTE NewEmp ('Goldsmith H', 10015, 600);
```

the NewEmp macro is used to supply Name, EmpNo, and DeptNo parameters in order to add employee Goldsmith to the Manufacturing department.

A macro may be modified using the REPLACE MACRO statement. For example, the statement

```
REPLACE MACRO NewEmp (name (VARCHAR(12),
    number (INTEGER, NOT NULL),
    dept (INTEGER, DEFAULT 300))
AS (INSERT INTO Employee (Name, EmpNo, DeptNo)
    VALUES (:name, :number, :dept);
    UPDATE Department SET EmpCount=EmpCount+1
    WHERE  DeptNo=:dept;);
```

replaces NewEmp with a macro that changes the default for dept from 100 to 300. If the NewEmp macro does not exist, the statement creates the new macro.

A macro is deleted using the DROP MACRO statement. For example

```
DROP MACRO NewEmp;
```

removes NewEmp from the data base.

Controlling Data

DBC/SQL enables users to control access to their own data. The user who creates a table, view, or macro is known as its creator. The creator of a table or view can perform SELECT, INSERT, UPDATE, DELETE, MODIFY, and DROP operations on the table or view and create and remove indexes on a table.

In addition, the creator can grant some or all of these privileges to other users and thus share the table or view. Similarly, the creator of a macro can grant EXECUTE privileges to other users.

Using the DBC/SQL GRANT statement, a creator can grant to specific users the option of granting to additional users the privileges that they have received. The grantor of a privilege can revoke it at any time. Thus a table, view, or macro can be as secure as its creator chooses.

HELP Statement

The DBC/SQL HELP statement enables interactive users to obtain information about any data base, table, view, macro, column (field), or index. In addition, the HELP statement may be used to

provide information about the statistics that have been collected on a table by using the DBC/SQL HELP STATISTICS statement. For each of these categories, the HELP statement provides the following information:

- Data Base

  Information about all tables, views, and macros in the data base.

- Table or View

  Information about all columns in the table or view.

- Macro

  Information about all macro parameters.

- Column or Field

  Information about attributes of the specific column that contains the field.

- Index

  Information about all of the indexes that are defined for a table, or about a selected index.

- Statistics

  Statistical information about table columns, time and data of collection, and number of distinct values at the time of collection.

INTERACTIVE TERADATA QUERY (ITEQ) FACILITY

ITEQ is a facility that enables a user on a host to enter DBC/SQL statements directly to the DBC/1012 Data Base Computer from an on-line terminal attached to the host computer. ITEQ includes functions for:

- Controlling the Operation of the Terminal

  An ITEQ user can define the use of various program function keys on the terminal keyboard, as well as the size of the screen areas.

- Entering, Editing, and Executing DBC/SQL Statements

  An ITEQ user can compose and execute DBC/SQL statements from a terminal. If the result is not satisfactory, the user may modify the statement without rekeying it.

- Formatting Output and Writing Reports

  An ITEQ user can format the result of a query for display on a terminal screen or for printing on a hardcopy device. ITEQ commands allow a user to view a result that exceeds the size of a single display, horizontally or vertically.

- Executing Macros

  An ITEQ user can define and execute macros containing many DBC/SQL statements and ITEQ format commands.

ITEQ supports IBM 3270-compatible devices with a screen size of at least 24 lines of 80 characters each. ITEQ automatically formats the results of a SELECT statement for display on the terminal screen or for printing on a system's output device.

Screen Format

Figure 3-3 shows the format of an ITEQ display screen. The display screen is divided into three areas: display, input, and status.

The display area usually is used to display responses to ITEQ commands and DBC/SQL statements. However, the area also can be used to compose a lengthy DBC/SQL statement, to modify macros or views, or to correct a previously entered DBC/SQL statement.

The input area is where ITEQ commands and DBC/SQL statements usually are entered. The cursor, a small underline character or a box that moves as a user keys, is positioned to the area that is used for input.

The status area is used to display ITEQ status and system messages.

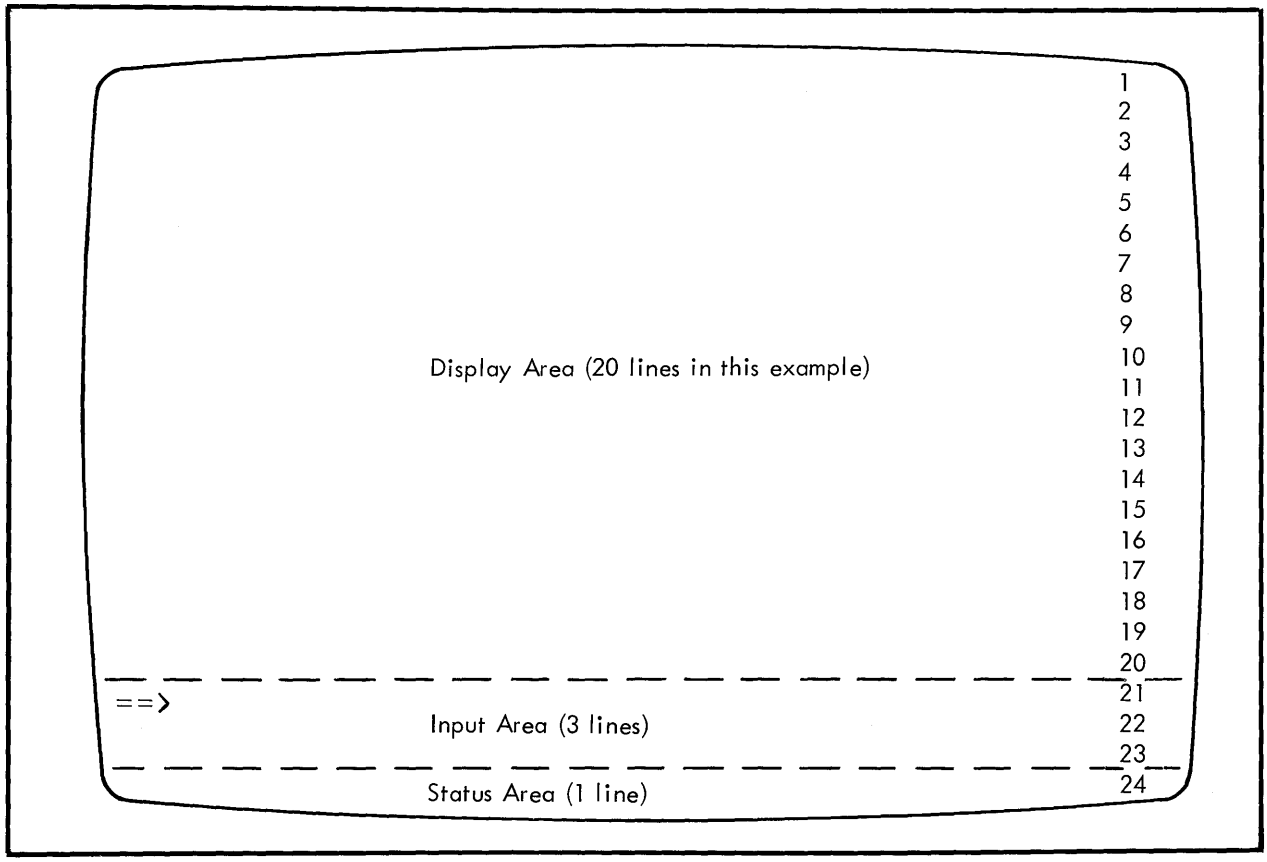

Figure 3-3. ITEQ Screen Format

Establishing a Session

To establish a session, a user enters the ITEQ LOGON command. After the logon has been accepted and the session established, the user may enter DBC/SQL statements or ITEQ commands.

Entering Statements and Commands

For entering DBC/SQL statements or ITEQ commands, the cursor is usually positioned after the arrow at the beginning of the input area and the statement or command is typed in. The DBC/SQL statement remains displayed in the input area during processing and display of the response.

If the statement is accepted, a message in the display area indicates the type of processing performed and shows the processing statistics and results, if any. If the statement is in error or the results are not satisfactory, the statement can be modified in the input area and resubmitted without rekeying the entire statement.

In Figure 3-4, a user has entered a simple query in the input area and received the result, complete with processing message and statistics, in the display area. The status area indicates that the response is complete and that the user may enter a new statement or command.

```
RETRIEVE COMPLETED. 13 RECORDS FOUND. 3 COLUMNS RETURNED.
   MAXIMUM LINE WIDTH IS 32 CHARACTERS.

DeptNo  Name             Salary
------  ------------  ----------
   100  Greene W       32,500.00
   100  Jones M        50,000.00
   100  Moffit H       35,000.00
   100  Peterson J     25,000.00
   300  Russell S      65,000.00
   500  Carter J       44,000.00
   500  Marston A      22,000.00
   500  Reed C         30,000.00
   500  Smith T        42,000.00
   500  Watson L       56,000.00
   600  Kemper R       29,000.00
   600  Newman P       28,600.00
   600  Regan R        44,000.00


==> SELECT DeptNo, Name, Salary FROM Employee
      ORDER BY DeptNo, Name;

                          *** END OF DATA.  READY FOR COMMAND.***
```

Figure 3-4. Result of a Query

Formatting Data

ITEQ and DBC/SQL together provide a powerful set of report writer features (although these features are not as comprehensive as those of BTEQ, described below). Data returned from a SELECT statement can be formatted for both a display terminal and a hardcopy device such as a line printer. Formatting includes centering, as well as the ability to:

- Show summary information (for example, totals and subtotals by groups).
- Specify a report title, which appears automatically, centered at the top of each display or printed page along with the current date and page number.
- Change column headings and column summary titles.
- Specify the format for numeric fields (for example, suppression of leading zeros, number of decimal places to appear, and so on).
- Suppress repeating column values.
- Establish printer page size (width and length).
- Cause a report file to be created, sent to a printer, or both.

Requesting Column Summaries

A user may want to specify both the summary information for a group and the data that comprises the group. This can be done by using the WITH clause in a SELECT statement. More than one WITH clause may be used to specify more than one level of summary data.

For example, in the following statement,

```
SELECT DeptNo, Name, Salary
FROM Employee
WHERE DeptNo IN (100, 500)
WITH SUM(Salary) BY DeptNo
ORDER BY Name;
```

the WITH clause is used to specify a salary total for departments 100 and 500. The result is shown in Figure 3-5.

In this statement, the WITH clause has the effect of ordering the data by department, so that "ORDER BY Department" is unnecessary.

Changing Headings and Titles

The TITLE phrase enables the user to define column headings and summary titles that are most appropriate for the report being prepared. In Figure 3-6, the heading for the DeptNo column is changed and a more meaningful title for the department salary totals is specified.

Format Commands

The following DBC/SQL statement requests a report of salary totals for every department, along with a grand total of salaries:

```
RETRIEVE COMPLETED. 9 RECORDS FOUND. 3 COLUMNS RETURNED.
   MAXIMUM LINE WIDTH IS 32 CHARACTERS.

DeptNo  Name             Salary
------  ------------  ----------
   100  Greene W       32,500.00
   100  Jones M        50,000.00
   100  Moffit H       35,000.00
   100  Peterson J     25,000.00
                      ----------
         SUM(Salary)  142,500.00

   500  Carter J       44,000.00
   500  Marston A      22,000.00
   500  Reed C         30,000.00
   500  Smith T        42,000.00
   500  Watson L       56,000.00
                      ----------
         SUM(Salary)  194,000.00

==> SELECT DeptNo, Name, Salary FROM Employee
     WHERE DeptNo IN (100, 500)
     WITH SUM(Salary) BY DeptNo  ORDER BY Name;
                          *** END OF DATA.  READY FOR COMMAND.***
```

Figure 3-5. Column Totals

```
SELECT DeptNo(TITLE 'Dept//Number'),Name,Salary FROM Employee
WITH SUM(Salary) (TITLE 'Dept TOTAL') BY DeptNo WITH SUM
(Salary) (TITLE 'TOTAL SALARIES***') ORDER BY Name;
```

To further define a format for a hardcopy of the report (shown in Figure 3-7), the following ITEQ commands set the format mode, suppress repetition of the values in the Dept Number column, and define a report title:

```
SET FORMAT ON;
SET SUPPRESS ON 1;
SET RTITLE 'DEPARTMENT SALARY REPORT';
```

```
RETRIEVE COMPLETED. 9 RECORDS FOUND. 3 COLUMNS RETURNED.
   MAXIMUM LINE WIDTH IS 32 CHARACTERS.
  Dept
Number  Name              Salary
------  ------------  ----------
   100  Greene W       32,500.00
   100  Jones M        50,000.00
   100  Moffit H       35,000.00
   100  Peterson J     25,000.00
                      ----------
           Dept TOTAL 142,500.00

   500  Carter J       44,000.00
   500  Marston A      22,000.00
   500  Reed C         30,000.00
   500  Smith T        42,000.00
   500  Watson L       56,000.00
                      ----------
           Dept TOTAL 194,000.00

==> SELECT DeptNo(TITLE 'Dept//Number'),Name,Salary FROM Employee
     WHERE DeptNo IN (100, 500) WITH SUM(Salary) (TITLE 'Dept
     TOTAL') BY DeptNo ORDER BY Name;
                          *** END OF DATA.  READY FOR COMMAND.***
```

Figure 3-6. Changing Headings and Titles

```
85/11/04          DEPARTMENT SALARY REPORT          PAGE   1

  Dept
Number  Name              Salary
------  ------------  ----------

   100  Greene W       32,500.00
        Jones M        50,000.00
        Moffit H       35,000.00
        Peterson J     25,000.00
                       ----------
           Dept TOTAL 142,500.00

   300  Russell S      65,000.00
                       ----------
           Dept TOTAL  65,000.00

   500  Carter J       44,000.00
        Marston A      22,000.00
        Reed C         30,000.00
        Smith T        42,000.00
        Watson L       56,000.00
                       ----------
           Dept TOTAL 194,000.00

   600  Kemper R       29,000.00
        Newman P       28,600.00

                       ----------
           Dept TOTAL 101,600.00
                       ----------
   TOTAL SALARIES***   503,100.00
```

Figure 3-7. Printed Report

BATCH TERADATA QUERY (BTEQ) FACILITY

The BTEQ program enables a user on a host or workstation to execute in batch mode (or in line-by-line interactive mode), a series of DBC/SQL statements and BTEQ commands. BTEQ features include:

- Use of more than one DBC/SQL statement per request
- Display of response time for each request
- Ability to read data from and write data to files maintained on the host computer

- Ability to repeat a request
- Use of more than one session for improved performance
- Ability to perform conditional tests and branching

In addition, BTEQ includes more comprehensive report formatting features than does ITEQ. These features include:

- Flexibility in defining report titles
- Ability to print footings and headings
- Ability to specify a page break when data changes in a given column (for example, the Dept Number column in Figure 3-7)
- Ability to skip one or more lines when data changes in a given column
- Ability to fold long lines in order to compress report width
- Flexibility in positioning summary titles
- Ability to specify column spacing, or to print a character (such as a vertical line) between column values

For example, the following BTEQ script is used to generate a report:

```
.LOGON someuser, somepassword
DATABASE PERSONNEL;
.SET FORMAT ON
.SET WIDTH 80
.SET HEADING 'Total Salary by Location, Department'
.SET FOOTING '&DATE &TIME  Confidential  Page&PAGE'
.SET SUPPRESS ON 1,2
  SELECT  Loc                  (TITLE 'Location')
         ,Department.DeptNo    (TITLE 'Dept.//No.')
         ,Name                 (TITLE 'Employee//Name')
         ,JobTitle             (TITLE 'JobTitle')
         ,Salary
         ,YrsExp               (TITLE 'Years//Experience')
    FROM  Department
         ,Employee
   WHERE  Loc IN ('NYC', 'ATL') AND
          Salary > 15000 AND
          Department.DeptNo=Employee.DeptNo
ORDER BY  Loc, Department.DeptNo, Name
    WITH  SUM(Salary) (TITLE 'Total for Department &2')
         ,SUM(YrsExp) (TITLE ' ', FORMAT 'zz9')
      BY  Loc,Department.DeptNo
    WITH  SUM(Salary) (TITLE 'Total for Location &1')
         ,SUM(YrsExp) (TITLE ' ', FORMAT 'zz9')
      BY  Loc
    WITH  SUM(Salary) (TITLE 'GRAND TOTAL')
         ,SUM(YrsExp) (TITLE ' ', FORMAT 'zz9')
         ;
.LOGOFF
```

The report is shown in Figure 3-8.

Total Salary by Location, Department

Location	Dept. No.	Employee Name	JobTitle	Salary	Years Experience
ATL	500	Carter J	Engineer	44,000.00	20
		Inglis C	Tech Writer	34,000.00	5
		Marston A	Secretary	22,000.00	8
		Omura H	Programmer	40,000.00	8
		Reed C	Technician	30,000.00	4
		Smith T	Engineer	42,000.00	10
		Watson L	Vice Pres	56,000.00	8
				---	---
		Total for Department 500		268,000.00	63
				---	---
		Total for Location ATL		268,000.00	63
NYC	100	Chin M	Controller	38,000.00	11
		Green W	Payroll Ck	32,500.00	15
		Jones M	Vice Pres	50,000.00	13
		Moffit H	Recruiter	35,000.00	3
		Peterson J	Payroll Ck	25,000.00	5
				---	---
		Total for Department 100		180,500.00	47
	300	Leidner P	Secretary	23,000.00	13
		Phan A	Vice Pres	55,000.00	12
		Russell S	President	65,000.00	25
				---	---
		Total for Department 300		143,000.00	50
	700	Brangel B	Salesperson	30,000.00	5
		Clements D	Salesperson	38,000.00	9
		Smith T	Manager	45,000.00	10
				---	---
		Total for Department 700		113,000.00	24
				---	---
		Total for Location NYC		436,500.00	121
				---	---
			GRAND TOTAL	704,500.00	184

85/05/29 11:41 Confidential Page 1

Figure 3-8. BTEQ Report

DBC/1012 COORDINATED PRODUCTS

A number of third-party software products run in conjunction with the DBC/1012, offering users alternate ways to manipulate table data:

1. INTELLECT[1] -- a natural language query facility
2. NOMAD2[2] -- a fourth-generation query language
3. PC/SQL-link[3] -- a PC-to-DBC/1012
4. FOCUS[4] -- a fourth-generation language
5. IDEAL[5] -- a fourth-generation language

The DBC/1012 INTELLECT Interface

The DBC/1012 INTELLECT Interface enables users to access data on the DBC/1012 using INTELLECT. INTELLECT is a natural-language query facility: users can query data using everyday English commands. The DBC/1012 INTELLECT Interface is available for both MVS and VM host systems.

The key component of INTELLECT's flexibility is its lexicon. The lexicon contains definitions of the tables that can be queried, along with English terms (synonyms, words, or phrases) to refer to them. Users can customize lexicons -- adding jargon and application-specific terms -- to suit their needs.

When a user enters an INTELLECT query, INTELLECT uses the lexicon to translate the query into DBC/SQL. From there, the DBC/1012 processes the query as if it had been submitted from any other user environment.

Special INTELLECT facilities, such as the Custom Format Option and the PGF Graphics Interface, can also be used via the DBC/1012.

NOMAD2 Interface

The DBC/1012 NOMAD2 Interface enables users to query data bases with the fourth-generation query language, NOMAD2. A data base may include data stored on the DBC/1012 or data stored in standard NOMAD2 files.

If the data being accessed resides on the DBC/1012, NOMAD2 generates the necessary DBC/SQL source statements and sends them to the DBC/1012 to be executed. The DBC/1012 then processes the query as if it were submitted from any other user environment. When the DBC/1012 returns the results back to the host, NOMAD2 resumes control and formats the data according to the user's command.

With the DBC/1012 NOMAD2 Interface, users can take advantage of NOMAD2 tools such as a comprehensive report writer, an interactive decision support tool, an extensive procedural language for application development, and a financial and statistical package.

---

(1) INTELLECT is a registered trademark of the Artificial Intelligence Corporation, Waltham, Massachusetts.

(2) NOMAD2 is a registered trademark of D&B Computing Services, a company of the Dun & Bradstreet Corporation.

(3) PC/SQL-link is a registered trademark of Micro Decisionware, Inc.

(4) FOCUS is a registered trademark of Information Builders, Inc.

(5) IDEAL is a registered trademark of Applied Data Research, Inc., Princeton, New Jersey

The DBC/1012 NOMAD2 Interface is available for both MVS and VM host systems.

PC/SQL-link Interface

PC/SQL-link is a menu-driven facility that accesses data on the DBC/1012 from an IBM personal computer or compatible running PC-DOS or MS-DOS 2.0 or higher. The user need not be familiar with the DBC/SQL language. There are several advantages to using PC/SQL-link:

- Statements can be developed off-line by non-SQL trained users via windows, menus, and prompts.
- A "free-form" mode allows DBC/SQL users to bypass menu selections.
- PC users may use application programs, such as Lotus(TM) and MultiPlan(TM) by automatically converting selected data to these PC data formats.

Each PC/SQL-link user owns a "local catalog" containing data dictionary information specific to the data to be accessed. The local catalog is stored on a PC storage device -- either a diskette or a hard disk. PC/SQL-link provides a set of menus to help users set up or refresh local catalogs.

The user is prompted through screens to choose source tables and to specify columns and conditions. At each step, the DBC/SQL statement being generated is displayed on the PC screen. This feature has the added benefit of teaching DBC/SQL syntax as the user proceeds. Query results are stored on PC data files. PC/SQL-link provides menu options for the user to view the file contents on the screen, to route the results to a printer, or to convert the data file to formats that can be used by other PC application programs. PC/SQL-link also provides menus for defining software and hardware configuration options such as modem equipment, host communication method, and printer options. The DBC/1012 PCSQL-link interface is available for a variety of host environments including MVS, VM, and GCOS (Honeywell).

FOCUS Interface

The FOCUS Interface is a fourth-generation language, a registered trademark of Information Builders, Inc. For details, contact a Teradata National Support representative, or your Teradata salesperson.

IDEAL Interface

The DBC/1012 IDEAL Interface enables users to quickly and easily develop online and batch production applications. IDEAL integrates the features of ADR/DATACOM/DB, the high performance relational data base; ADR/DATADICTIONARY, for centralized resource management; a fourth-generation language, screen painting, report writing, interactive developer's workstation, and facilities for managing the entire application life cycle.

An IDEAL application definition consists of two major classes of components:

- Fill-in-the-blank screens

  These are for general declarative, or nonprocedural, information, such as a description of the application, its inputs and outputs, reports, and panels.

- A high-level procedure language

  This language incorporates relational data management services, logic, and binding of the nonprocedural components.

THE DBC/1012 COP

To broaden the DBC/1012 interface capabilities, Teradata has introduced the COP (Communications Processor). The COP enables the DBC/1012 to communicate with a combination of mainframes, minicomputers, and PCs by means of networks. The COP supports devices attached to an Ethernet Local Area Network (LAN). Teradata expects that other types of networks will be supported in the future. Figure 3-9 shows the various components of the COP Interface.

The COP functions similarly to an IFP. Instead of the IFP receiving requests from the host via a block multiplexer channel, the COP receives requests via messages over a local area network. Network protocols define how messages are transported between the LAN-attached host(s) and the DBC/1012. The initial release of the COP Interface is currently available with two types of network protocols:

1. ISO/OSI
2. TCP/IP

Teradata will make other protocols available in the future. Other than the network software already obtained by the user, Teradata provides host software for each host that accesses the DBC/1012 via the COP.

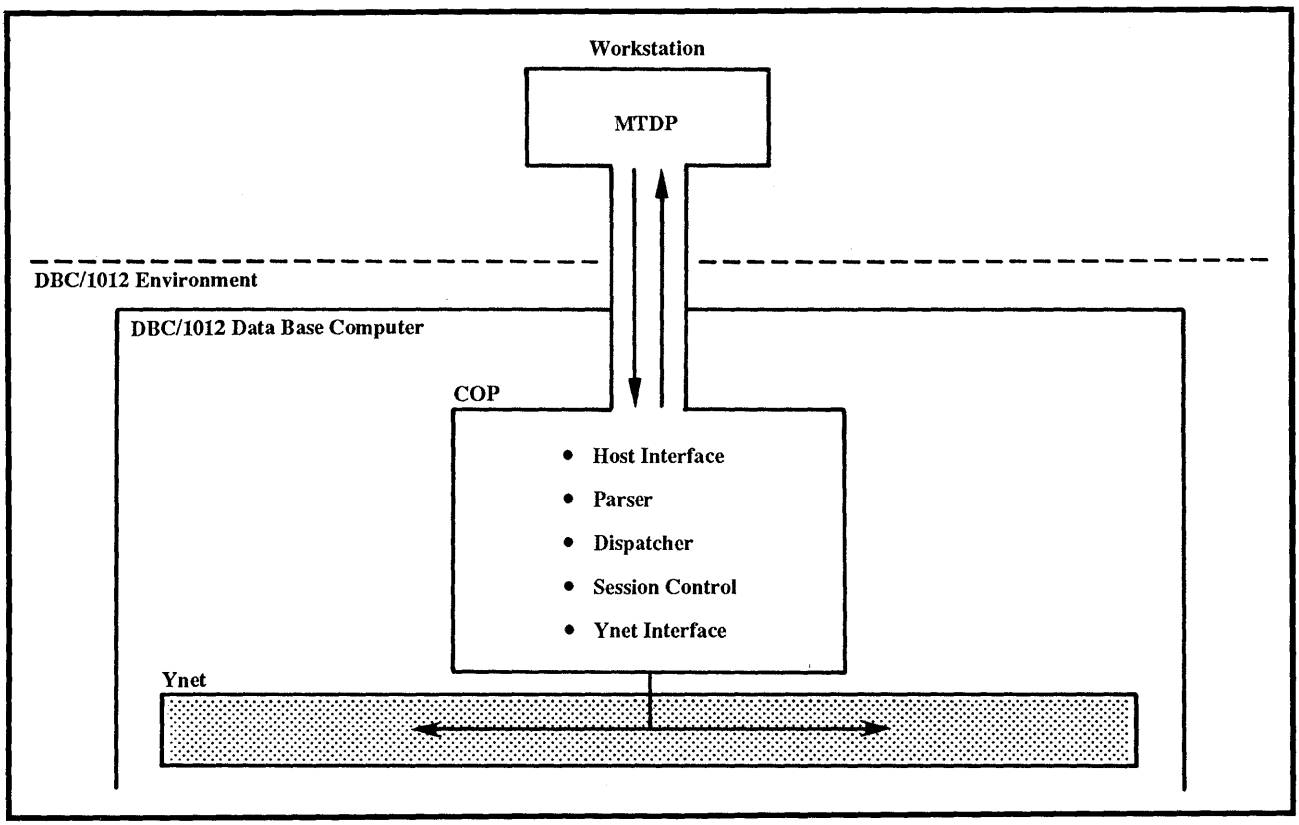

Figure 3-9. COP Interface Components

Host software includes BTEQ, Call Level Interface (CLI) version 2, the Micro Teradata Director Program (MTDP), and the Micro Operating System (MOSI).

The CLI, version 2, is involved with managing the blocking and deblocking of messages. The MTDP is called by the CLI to manage communications with the DBC/1012. The MOSI is a library in which the operating-system-dependent and communication-protocol-dependent services have been collected, for ease in porting the interface to a new operating system. Calls to MOSI procedures are mapped onto calls to the relevant operating system or protocol. Teradata currently provides software versions for

1. IBM PCs or PC-compatibles running MS-DOS or PC-DOS (version 3.1 or higher)

2. AT&T 3B2/300 and 400 computers running UNIX 5.2

Teradata intends to support other host environments.

Using CLI, programmers may also write special-purpose applications, if they are required at the user's site.

Refer to the DBC/1012 Data Base Computer COP Interface Manual, document number C12-0005, for further information on the above features. See also the DBC/1012 Data Base Computer Network Reference Manual, C03-0003, the DBC/1012 Data Base Computer Workstation User's Guide, C09-0003, and the DBC/1012 Data Base Computer Call-Level Interface Mnaual, C12-0006.

DATA DICTIONARY/DIRECTORY

The DBC/1012 system has a fully integrated, active Data Dictionary/Directory that lets users access information about DBC/1012 objects (e.g., tables). The Data Dictionary/Directory is composed of system tables and views that contain current definitions, control information, and general information about data bases, tables, columns, indexes, views, users, accounts, access rights, and macros. The Data Dictionary/Directory is automatically updated in response to DBC/SQL data definition operations. For example:

- When a user creates a table, the table name, column names, and column attributes are automatically added to the Data Dictionary/Directory and the table is identified with the user.

- When a table is dropped, its column definitions are dropped. All privileges associated with that table are also dropped.

Information Levels

Data Dictionary/Directory information is geared to meet the needs of three classes of DBC/1012 user:

1. End user, responsible for the user's own data bases. The end user needs to know what information is available, what form it is in, how to get it, and what access rights the user has granted to others.

2. Supervisory user, responsible for data bases and users. The supervisory user creates and organizes data bases, monitors use of space, defines new users, allocates control privileges, creates indexes, and performs archiving operations.

3. DBC/1012 administrator, responsible for operation and administration of the system. The administrator needs information about system performance, status and statistics, errors, and accounting.

Querying the Dictionary/Directory

Data Dictionary/Directory information is stored as system tables. A user may gain access to Data Dictionary/Directory tables by means of system-defined views.

NOTE: These tables and views may be queried via the DBC/SQL as described earlier in this chapter.

To select information about the Personnel data base, the user can enter the following statement:

```
SELECT Databasename, Creatorname, Ownername,
Permspace, Fallbackflag
FROM DBC.DataBases
WHERE Databasename='Personnel';
```

The result resembles the following example:

```
Databasename  Creatorname  Ownername  Permspace  Fallbackflag
------------  -----------  ---------  ---------  ------------
Personnel     Jones        Jones       1000000   Y
```

Definitions for each of the columns shown above are presented below.

- DatabaseName

  Returns the name of a data base as CHARACTER, FORMAT 'X (30)'

- CreatorName

  Returns the name of the user who created the data base as CHARACTER, FORMAT 'X (30)'

- OwnerName

  Returns the name of the data base owner as CHARACTER, FORMAT 'X (30)'

- Permspace

  Returns an integer indicating the total space allocated to the data base on all AMPs. FORMAT is 'ZZZ,ZZZ,ZZZ,ZZZ,ZZ9'.

- Fallbackflag

  Returns an F (FALLBACK) or N (none) to indicate if the data base is protected by the FALLBACK option. Data type is CHARACTER, FORMAT 'X'.

System Views

Data Dictionary/Directory views, intended for a variety of DBC/1012 users, are summarized in the subsections that follow.

End User Views

The Data Dictionary/Directory views summarized in Table 3-1 provide information about data bases, tables, and columns.

Table 3-1. End User Views

View	Contents
Columns	Attributes of columns and parameters of tables, views, and macros
DataBases	Characteristics of data bases
Journals	Information about journal-to-table mapping
Tables	Tables, views, and macros that have been created in a data base
UserGranted-Rights	Privileges that the user has granted to other users
UserRights	Privileges that the user has been granted on data bases, tables, views, and macros

Supervisory User Views

The Data Dictionary/Directory views summarized in Table 3-2 provide information about users and indexes.

Table 3-2. Supervisory User Views

View	Contents
AccountInfo	Accounts that are available to a given user
Indices	Kinds of indexes defined for a given table
Users	Information about users that are owned, or have been created, by a user

The concept of "ownership" of users and data bases is explained in Chapter 4, Space Allocation and Access Control.

DBC/1012 Administrator Views

Table 3-3 lists the Data Dictionary/Directory views that the DBC/1012 administrator may use to obtain information about errors, disk space usage, AMP processor usage, sessions, and logon and logoff activity.

Table 3-3. DBC/1012 Administrator Views

View Name	Contents
AllRights	Information about which users have what rights on what objects
AllSpace	AMP-by-AMP information about disk space usage (including spool) for any data base, table, or account
AMPusage	Information about AMP usage for each data base and account
Children	Names of data bases and users that are owned by a user
Delete-SecurityLog	Removes information logged for SecurityLog statements that is more than 30 days old
DiskSpace	AMP-by-AMP information about disk space usage (including spool) for each data base or account
ErrorLog	Log of system errors
LogOnOff	Record of all logon and logoff activity
ResUseView	Summarizes information about processor utilization that may be used in capacity planning
SecurityLog	Log of all statements entered that affect privileges (access rights)
SessionInfo	Information about users who are currently logged on
TableSize	AMP-by-AMP information about disk space usage (not including spool) for any data base, table, or account

Recovery Control User Views

The Recovery Control Catalog (RCC) consists of two systems tables, and this allows a user to access information about archive and recovery operations. The RCC consists of standard system tables that contain rows that define the logon user identification and the operations taken through the host utilities dump and restore.

Table 3-4. Recovery Control Views

View	Contents
Journals	Information about journal-to-table mapping
Events	In audit trail of all archive and recovery activity
Events_Configuration	Detail of all archive and recovery activity that did not effect all AMPs
Event_Media	Information about archive and recovery events that involve a removable media.

LANGUAGE PREPROCESSORS

Teradata offers two language preprocessors that facilitate development of application programs that access information stored in DBC/1012 data bases:

- COBOL Preprocessor
- PL/1 Preprocessor

The user embeds DBC/SQL statements directly in a COBOL or PL/1 source program, intermixing them with source language statements. The appropriate language preprocessor then converts the program to a form compatible with ANSI COBOL 1974 or IBM PL/1.

A language preprocessor permits full use of DBC/SQL data manipulation capabilities with a relational data base. These include using joins, nested subqueries, aggregate operations, creating objects, and so on. A language preprocessor provides the following benefits:

- Insulates the user from the interface between a host computer and the DBC/1012 Data Base Computer.
- Simplifies and eases the coding task by allowing the user to use a high-level DBC/SQL statement in situations that might require many source language statements.

A language preprocessor, running in a host computer as a batch job, processes source language programs. Before the source program is compiled, the language preprocessor scans the program for embedded DBC/SQL statements identified by a prefix delimiter, normally a ? character. The preprocessor replaces the DBC/SQL statement mainly with a call to a CLI routines that uses the DBC/SQL statement as a parameter.

The preprocessed source language program (which includes the original DBC/SQL statements as source language comments) is then compiled. The compiled program may be link-edited with the CLI routines or the routines may be dynamically loaded at runtime. A language preprocessor provides facilities for:

- Coupling DBC/SQL statements with program variables

- Determining the status and statistics of a DBC/SQL request
- Handling error conditions
- Selecting a number of rows into program variables
- Including source lines from another file
- Establishing and terminating a session

CALL-LEVEL INTERFACE

The Call-Level Interface (CLI) consists of service routines that allow application programs (written in high-level languages) with a CALL statement to communicate with the DBC/1012 Data Base Computer.

The application programs with CALL statements that may be used from a mainframe host are as follows:

- COBOL
- Pascal
- PL/I
- FORTRAN
- Assembler

The C application program with a CALL statement may be used from a workstation.

A CLI service routine takes requests via parameters. The CLI routine then communicates the request service routines (MVS) or the Inter-User Communication Vehicle (VM) to the Teradata Director Program (TDP). The TDP/MTDP manages the physical input and output of DBC/SQL requests between many users in the host or workstation and the DBC/1012 Data Base Computer. The CLI maintains a context for each session and each request.

CLI uses standard linkage conventions. An application program includes CALL statements that invoke the CLI at various entry points in order to:

- Provide logon and logoff capability
- Send and receive information.

The CLI may be used with any of several host software environments. For mainframes, the software environments are:

- MVS/XA; MVS/SP, release 1.3 or above VM/SP, release 3
- Teradata Director Program, release 1.1
- CICS, release 1.6 or above
- GCOS (Honeywell)

For workstations, the software environments are:

- MS-DOS/PC-DOS
- UNIX

For more details about workstation software, consult the Network Reference Manual.

SUMMARY AND PREVIEW

This chapter described DBC/1012 facilities that allow an end user or application program to define, and manipulate data. The next chapter discusses facilities for controlling the operation and administration of the DBC/1012.

CHAPTER 4 SYSTEM FACILITIES

Teradata provides numerous facilities to control the operation and administration of the DBC/1012 Data Base Computer. These facilities include:

- Session protocol
- Data protection
- User security interface
- Logon security exit
- Space allocation and access control
- General purpose utility programs for archiving and bulk-loading data
- Reconfiguration
- System maintenance
- System console operation
- System status and statistics
- Accounting

SESSION PROTOCOL

To communicate with the DBC/1012 Data Base Computer, an end user or application program must establish a session (the context for performing operations on a data base). To establish a session, the user logs on to the DBC/1012 system.

The logon procedure varies depending on the host, the operating system, and whether the "user" is an application program or a person at an interactive terminal communicating through ITEQ. Logon parameters may include an optional identifier for the Teradata Director Program in the host computer (tdpid), user identifier (username), password, and optional account number.

The session is established when the DBC/1012 Data Base Computer accepts the username, password, and account number. Subsequent DBC/SQL statements generated by the user and responses returned from the DBC/1012 Data Base Computer are identified by a session number. The session context also includes a default data base name. The default data base name is the same as the user name, as noted below under Creating Data Bases and Users.

When a session is ended, the context is discarded and no additional DBC/SQL statements are accepted from the user.

DATA PROTECTION

Comprehensive DBC/1012 data protection ensures the integrity of a data base during concurrent access by many users, after the failure of a host system or the DBC/1012 Data Base Computer, and following abnormal termination of an application program.

Data protection is accomplished by:

- A locking mechanism that ensures the consistency of data being accessed concurrently by many users.
- Journals that record before and after images of modified fields of a table and other information.
- Optional redundant storage of data on the DSUs of separate AMPs.
- The "transaction" concept that treats a sequential set of DBC/SQL statements in a session as a single unit of work. All statements must be completed or the transaction is aborted without having an effect on the data base.
- A recovery process that is invoked automatically whenever a system component fails.

In addition, the DBC/1012 system provides for re-creation of a data base from archived copies. This feature is discussed below under General Purpose Utility Programs.

Concurrency Control

The DBC/1012 system ensures that users who are concurrently trying to change the same data do not violate the consistency of shared data stored in the data base. In addition, where it is important to the user, the DBC/1012 provides a facility that ensures that the user will never see inconsistent data.

This concurrency control is implemented by "locking" the shared data. A user holding a lock is assured that data remains consistent. Any other user's request for access is held until the lock is released.

Locking is done automatically by the system. Locks are acquired during the processing of a request and released at completion of the request. In addition, the user may explicitly specify locks. If a request is aborted, locks are released immediately. The DBC/1012 system provides three levels of locks and three types of locks. Levels of locks are:

1. Data base
2. Table
3. Row

Types of locks are:

- Exclusive
- Write
- Read
- Access

An exclusive lock, applied only to a data base or table, is the most restrictive lock: all other users are locked out. Exclusive locks are rarely used and are generally only necessary when structural changes are being made to the data base.

A write lock enables a single user to modify data while locking out all other users except readers not concerned about data consistency. Until a write lock is released, no new read locks are allowed.

A read lock is used to ensure consistency during read (for example, SELECT) operations. Several users may hold read locks on a table, during which no modification of the table is permitted.

An access lock may be specified by a user who is not concerned about data consistency. Use of an access lock allows modifications of the table while the select operation is in progress.

Transient Journal

A transient journal is a sequential record of modifications made to data stored on the DBC/1012. A journal record includes identification of the transaction, and a copy of the data before modification.

Journaled information is used in two ways. First, if a transaction is not completed (that is, if it is aborted), transient journal entries are used to "roll back", or restore, the data base to its state before the transaction was begun. Second, journaled information other than the transient journal is used to bring a temporarily inoperative AMP and its disk into conformity with data on fallback AMPs, which was modified while the AMP was inoperative.

Permanent Journaling

The permanent journaling feature is another way to protect against losing data. The user specifies the permanent journaling feature at the time he creates the data base or at some later time. In either case, only one permanent journal table may be specified per data base. If the user does not specify permanent journaling, that feature will not go into effect.

The permanent journal table may be used by a specific data table or by any or all other tables in a data base. Also, a data table or data tables from one data base may use a permanent journal in another data base.

When specifying the permanent journaling feature, the options below are available for selection. These options are to

- log before-change images
- log after-change images
- perform single logging
- perform dual logging

A change image is a "picture" of a data table row before or after the row is changed. The picture is stored to the permanent journal table.

The single logging option provides one copy of an image (whether it is a before-change image or an after-change image). The dual logging option provides added protection because two copies of an image (before-image or an after-image) are made. Single logging is the default.

A checkpoint capability is provided as part of the permanent journaling feature. This capability allows the user to mark the permanent journal at specific points in time to aid in the process of recovering the tables being journaled.

To protect against the loss of data in the event of a site disaster, a copy or copies of DBC tables, the data tables, and permanent journals could be archived and restored to another DBC/1012 at another site. The user may archive as often as necessary.

To restore data, the user must first determine which permanent journal contains the data needed. The data may be in a current permanent journal (the journal that currently exists in the data base), or an archived permanent journal.

If the data is in the current permanent journal, a roll forward or roll backward may be performed at any time.

If the data is in an archived permanent journal, that journal must first be restored to the DBC/1012; then a roll forward or roll backward may be performed.

When using either the roll forward or roll backward statements, the user may roll everything in an entire journal forward or backward, or roll forward or backward to a specific checkpoint in the journal.

Redundant Data Storage

As described in Chapter 2 (see Figure 2-7), the rows of every table are distributed evenly across all the AMPs in a DBC/1012 Data Base Computer. At the user's option, each row of a table may be redundantly stored on the DSU of a separate AMP. In such cases, one AMP stores the primary copy of a row while another AMP stores a secondary or fallback copy. Select, insert, delete, and update requests are sent to the primary AMPs, which perform the operation specified; insert, update, and delete statements are also to the fallback AMPs. If a primary AMP is inoperative, a request is sent directly to the secondary AMP. When the primary AMP is again operative, it gathers from its fallback AMPs all updates that occurred while it was inoperative. The AMP then updates data on its DSU to conform with that on fallback DSUs and resumes its role as primary AMP.

To decrease the probability of data being lost when two or more AMPs fail simultaneously, AMPs can be clustered in groups of from two to sixteen. The fallback copy of a row is always stored in the same cluster as the primary copy. Thus, the DBC/1012 can tolerate more than one AMP failure without loss of data as long as two or more failures do not occur within the same cluster.

Transaction Management

The term "transaction" refers to one DBC/SQL statement or a sequence of DBC/SQL statements that are treated as a single unit of work. The DBC/1012 system manages transactions so as to maintain consistent data without unnecessarily locking resources. All statements within a transaction are performed or none of them are performed. If for any reason (for example, statement errors, deadlocks, access rights violations, table constraint violations) a statement cannot be completed successfully, the entire transaction is aborted. Any changes made to the data base up to that point are backed out and locks are released.

Transactions are implicit or explicit. Typical implicit transactions are multi-statement requests and macros. An explicit transaction (for example, a series of related data manipulation statements that operate on a data base) is bracketed by BEGIN TRANSACTION and END TRANSACTION statements. An explicit transaction might be defined for a preprocessor as follows:

```
?BEGIN TRANSACTION;

?DELETE FROM Employee
?WHERE Name='Smith T';

?UPDATE Department
?SET EmpCount=EmpCount-1
?WHERE DeptNo=500;

?END TRANSACTION;
```

If an error occurred during the processing of either the DELETE or UPDATE statements within the BEGIN TRANSACTION and END TRANSACTION statements, both Employee or Department tables would be restored to their state before the transaction.

Recovery

A primary function of DBC/1012 recovery is to restore tables to their state before an error occurred. There are three types of recovery:

1. Single transaction recovery. This goes into effect when a single transaction was aborted by the DBC/1012 Data Base Computer because of a user error or a deadlock timeout. In order to avoid a deadlock when two or more transactions are vying for the same lock(s), the most recent transaction(s) entered is aborted to resolve the potential deadlock.

2. DBC/1012 recovery. This goes into effect when there is a major DBC/1012 system error (e.g., failure of an IFP, AMP, or Ynet).

3. Host recovery. This goes into effect when the host computer or Teradata Director Program (TDP) fails.

When a transaction is aborted because of user error, timeout, or DBC/1012 recovery, all changes made to table data are backed out and rows are reconstructed from transient journal entries. If the abort is caused by user error, the user may resubmit a corrected transaction. If the abort is caused by timeout or DBC/1012 recovery, the user may later resubmit the same transaction. All IFPs, AMPs, COPs, and the TDP or MTDP participate in recovery. During recovery, each AMP first restores the tables on its DSUs to then pre-failure state. To accomplish this task, the AMP:

- Rebuilds its memory-resident index to data on the DSU
- Accesses transient journal entries to reconstruct its tables
- Accesses fallback AMPs to obtain updates that occurred while it was inoperative (applies only if the AMP is recovering from its own failure)
- Reports to a controlling AMP when it is again operational

After the AMPs become operational, the IFPs and COPs reload their session information and notify the host computer that recovery has been successfully completed.

If an AMP fails to come on-line during system recovery, the DBC/1012 Data Base Computer continues to process transactions from the host using fallback AMPs. When the AMP is ready to rejoin the system, the system is restarted. All AMPs remain in a quiescent state until the AMP has synchronized its data with that on the fallback AMPs.

USER SECURITY INTERFACE

On channel-connected hosts, any return code from the DBC/1012 that indicates an illegal or invalid attempt to access data is intercepted and passed to the TDP User Security Interface.

Under normal conditions, when it is notified of a security violation, the User Security Interface:

- Sets a return code indicating that an error message is to be issued to the host operator and written to the system log, and that a Security Violation SMF (System Management Facility) record is to be written.

- Returns control to the TDPSECUR module (which called the User Security Interface).

USER LOGON INTERFACE

The user may examine logon requests before they are sent to the DBC/1012 via the User Logon Exit Interface (TDPLGUX). Therefore, the user may either allow a logon request to continue, to deny the logon request, or to modify the logon string.

A logon request is first passed to TDPLGUX before the request is allowed to continue. The interface consists of a TDP task (TDPLGEX) that receives all logon requests and an exit.

When a TDPLGUX is enabled, a TDPLGEX first builds a user parameter list. This list consists of the logon string and length, the job name, the session number, and other pertinent information. This list is passed on to the TDPLGUX routine.

When the TDPLGUX receives this information, it may or may not allow the logon to proceed. Otherwise, the TDPLGUX may change the logon string.

If TDPLGUX returns with a return code of zero, the logon string is moved back to a data block and the logon is allowed to proceed. If a non-zero return code is returned, the logon request is denied and the violation is reported to the security interface (TDPSECUR).

SPACE ALLOCATION AND ACCESS CONTROL

In DBC/SQL, the term "data base" refers to a collection of related tables, views, and macros. A data base also contains an allotment of space from which users may create and maintain their own tables, views, macros, or other users or data bases.

Besides providing a logical grouping, data bases are the foundation for DBC/1012 space allocation and access control.

Creating Data Bases and Users

When first installed in an organization, the DBC/1012 Data Base Computer contains a single data base named "DBC", which is usually managed by a system administrator. It is from disk space belonging to DBC that the administrator assigns space to all other organization data bases. Therefore, DBC may be said to "own" these data bases. Consider the example below of how the system administrator may manage DBC. To protect the security of system tables within DBC that are used by DBC/1012 software, the system administrator usually creates a DBC/1012 administrator data base from DBC. The administrator assigns all DBC disk space not needed for these system tables to the new administrator data base.

The administrator allocates space from the administrator data base to the organization's data bases and users. For example, the administrator creates a Finance and Administration (F&A) department data base and creates Jones as a supervisory user (referred to in some organizations as a data base administrator or DBA) within F&A.

Jones is allocated space from F&A for Jones' own data and granted all privileges on the F&A data base, as discussed below. Jones allocates space from F&A to create the Personnel data base and other department data bases and user space allotments.

Thus, when a new data base is created or space is allotted to a user, disk space is assigned from the space belonging to an existing data base or user. This data base or user is then known as the "owner" of the new data base or user space. The owner permanently grants the new data base or the user a specified amount of space, which is subtracted from the unused space available to the owner. This hierarchical ownership structure is illustrated in Figure 4-1.

In the figure, the F&A department data base owns Personnel and all other department data bases. F&A also owns the space allotted to user Jones and all other users within the department. Because the DBC data base owns all space, however, DBC is the ultimate owner of all of the organization's data bases and user space.

This hierarchical ownership structure gives the owner of a data base or user space complete control over the security of owned data. The owner is free to dump the data base or to control access to its data by granting or revoking privileges on it.

Granting and Revoking Privileges

When a user creates a data base, the user is automatically granted all privileges on any table, view, macro, or other object in the data base. When a user creates an object in a data base on which the user has privileges, the user is automatically granted all privileges on that object.

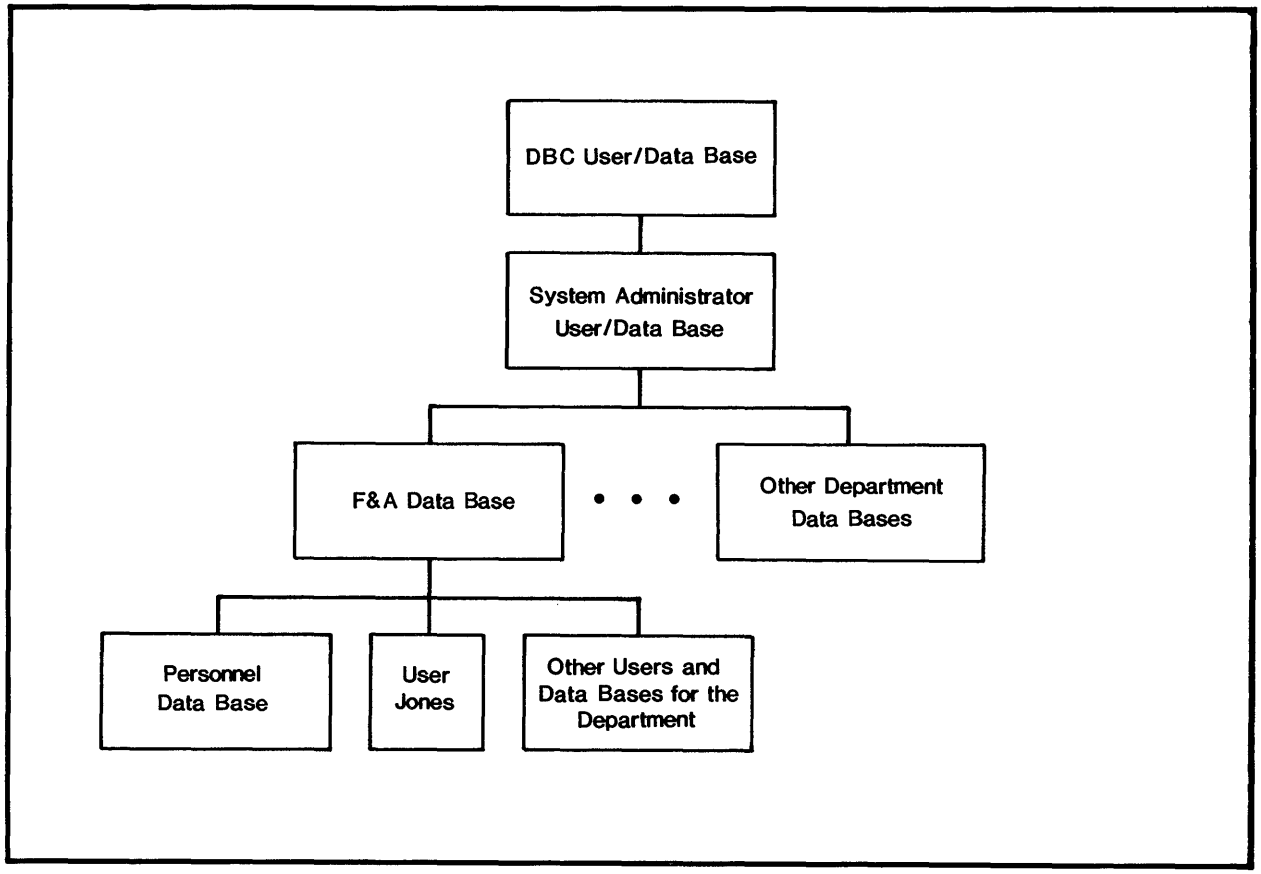

Figure 4-1. Ownership Structure in a Data Base

Some of the privileges that may be granted are:

- CREATE: To define users, data bases, and tables using the space of an existing data base; to define views and macros within data bases
- DROP: To delete objects defined by the CREATE statement
- SELECT: To obtain information from a table or view
- UPDATE: To modify rows in a table
- INSERT: To insert rows in a table
- DELETE: To remove rows from a table
- GRANT: To extend privileges to another user
- EXECUTE: To execute a macro

Privileges are explicitly controlled through GRANT and REVOKE statements. Both statements specify the privileges being granted or revoked, the users receiving or losing the privileges, and the objects on which the privileges are granted or revoked. For example, the statement:

```
GRANT ALL PRIVILEGES ON Personnel TO Peterson
WITH GRANT OPTION;
```

gives Peterson permission to grant the following operations in the Personnel data base:

```
CHECKPOINT  CREATE/ DROP/ DATABASE/ MACRO/ TABLE/ USER/
VIEW/ DELETE, EXECUTE, INSERT, SELECT, UPDATE,
REPLACE, COMMENT, DUMP, EXECUTE, GRANT, RESTORE
```

The user entering a GRANT statement must have a GRANT privilege on the object or on its owner, or must be the owner. Further, the user must have any privilege being granted.

If the following statement,

```
GRANT SELECT ON Employee Info TO Peterson;
```

were entered rather than the preceding statement, Peterson would instead be given a limited retrieval privilege on the Employee table via the Employee Info view created in Chapter 3.

Or, the statement,

```
GRANT CREATE TABLE ON Personnel TO Peterson;
```

would allow Peterson to create and control Peterson's own tables in the Personnel data base.

The syntax of the REVOKE statement is almost identical to that of the GRANT statement. For example, the statement,

```
REVOKE CREATE TABLE ON Personnel TO Peterson;
```

removes the CREATE TABLE privilege granted to Peterson.

CREATE DATABASE Statement

The CREATE DATABASE statement is used to create a data base in which tables, views, and macros can be defined. For example, the following statement:

```
CREATE DATABASE Personnel FROM F&A
AS PERMANENT = 10000000 BYTES, SPOOL = 100000000 bytes
FALLBACK, ACCOUNT = 'Administration';
```

might be used to create the Personnel data base in which the Employee and Department tables are defined in Chapter 3.

AS introduces a clause that specifies one or more data base parameters. Any parameters that are not specified are set by default. PERMANENT specifies the allocation of disk space to the new data base.

The FROM clause and the SPOOL, FALLBACK, and ACCOUNT parameters are optional. FROM introduces an owner data base whose disk space is allocated to create the new data base. If an owner data base is not specified, the data base of the user entering the statement is assumed. If an owner data base is specified, the user must have a CREATE DATABASE privilege on that data base or be its owner.

The SPOOL parameter specifies a space limit for spool files, which store the results of DBC/SQL statement processing until they can be examined by a user or application program. If not specified, the spool space allocated to the owner data base is used.

FALLBACK specifies that, in addition to the primary copy of the rows of all data base tables, a secondary (fallback) copy also is distributed among the DSUs of all AMPs in the DBC/1012 system. FALLBACK is the default.

ACCOUNT identifies for accounting purposes the department or budget responsible for accumulated disk space used by the new data base. If not specified, the account of the owner data base is used. The account id is also used to establish relative priorities of the sessions.

CREATE USER Statement

Users are defined with the CREATE USER statement. This statement authorizes a new user identification (username) for the DBC/1012 system and specifies a PASSWORD for user authentication (see Session Protocol, above).

Because a data base is automatically created for each user, the CREATE USER statement uses essentially the same parameters as the CREATE DATABASE statement, as shown in the following example:

```
CREATE USER Jones FROM F&A
AS PERMANENT = 100000, PASSWORD = Jan,
FALLBACK, SPOOL = 1000000,
ACCOUNT = 'Administration',
STARTUP = 'DATABASE F&A;';
```

The optional STARTUP clause specifies one or more DBC/SQL statements that may be executed automatically when the user establishes a session. In this example, a DATABASE statement automatically establishes F&A as a default data base.

The user executing this statement must have a CREATE USER privilege on the owner data base or be its owner. The new user is automatically granted all privileges on tables, views, and macros created in the disk space assigned to the user.

GENERAL PURPOSE UTILITY PROGRAMS

A number of utility programs are provided to enhance and support system operation. Provided here is an overview of the utilities provided with the system. For more details about these utilities, see the Utilities Reference Manual, C11-0001.

Utility programs are used to archive and restore a data base and to load large amounts of data into a data base. These programs reside on the host and are invoked by utility statements.

Archiving and Restoring a Data Base

The Dump and Restore Utility runs on the host system and is used to

- Archive (dump) a copy of a data base, individual table, or permanent journal from the DBC/1012 to a host-resident dataset.
- Restore a data base, individual table, or journal back to the DBC/1012 from an archive dataset on the host system.
- Place a synchronization (checkpoint) entry in a journal table for later use in recovery or archive activities.
- Recover a data base to a previous synchronization point by either rolling it back or forward with change images from a journal table.
- Deleting change image rows from a DBC resident journal table.

Bulk Loading Data from a Host

A bulk load program allows a user to perform batch select insert, update, and delete operations on an existing data base. The program moves large amounts of data from a user-supplied input file in a host system to a DBC/1012 data base.

The bulk load program sends messages containing data from the input file to the DBC/1012 Data Base Computer. Each message contains either an INSERT or UPDATE statement and data to be inserted or updated, or a DELETE statement that identifies rows to be deleted. As each message is received, a response indicates the success or failure of that load operation.

Fast Loading Data from a Host

A fast load program enables a user to load newly created tables with data from a host input file. This program is similar to bulk load, except that it loads data much faster, and does not include update and delete capabilities.

RECONFIGURATION

The primary purpose of reconfiguration is to automatically redistribute data on the DSUs when the configuration of a DBC/1012 system is changed (that is, when AMPs are added or removed). The distribution of data to AMPs is based on the primary index field(s) of the rows. Using these primary indexes, a "hashing" algorithm determines to which AMPs the data rows are sent. When AMPs are added to a system, some portion of data is moved from existing AMPs to the new AMPs. When AMPs are removed from a system, a reverse process redistributes data rows from the AMPs to be deleted to the remaining AMPs.

In addition, cluster assignments may be reassigned to enhance availability, or changed to accommodate the addition or deletion of AMPs. In these cases, fallback data is also redistributed.

When data is redistributed, the amount of data moved is kept to a minimum. When AMPs are added, primary data is moved only from existing AMPs to the new AMPs, not between existing AMPs. When AMPs are removed, data is moved only from those removed to those remaining, not between remaining AMPs. When cluster assignments are changed, fallback data is not moved between AMPs remaining in the same cluster.

After new processors are installed in a system and tested, the system is restarted. After restart, reconfiguration can take place at any time to make the new configuration fully functional. The original system can operate under the old configuration until that time.

SYSTEM MAINTENANCE

The DBC/1012 Data Base Computer provides the following maintenance facilities:

- Trouble log
- Automatic testing of processors during startup
- Isolation, diagnosis, and repair of off-line processor modules during on-line system operation
- Electrical readout of processor status

Hardware and software failures are recorded in a trouble log. The log, maintained as standard system tables, is accessed by operations personnel using DBC/SQL statements.

Following processor restart, a complete set of hardware tests is made. If any processor component is malfunctioning, the processor module is taken out of service. To trouble-shoot problems detected during startup hardware testing, customer engineers can isolate a processor module off-line and run diagnostics from a system console. Tests indicate whether each hardware element is functional, whether it can communicate with and detect a malfunction of other elements, and whether it can communicate over the various required paths (for example, an AMP with its DSU and with each Ynet).

While the DBC/1012 system is operating, the control panel of each processor cabinet contains a readout of status information for each processor module in the cabinet refer to Chapter 5, "Packaging".

SYSTEM CONSOLE OPERATION

The system console allows operations and support personnel to communicate directly with the DBC/1012 Data Base Computer. The console is an intelligent keyboard display terminal with diskette storage and a printer. The console is attached via a serial interface to a DBC/1012 processor, through which it can communicate with all processors in the system.

From the console, system status and performance data can be requested and displayed or printed. Certain system functions, such as disk rebuild and reconfiguration, also can be invoked. Off-line utility and diagnostic programs available on diskette can be loaded and executed on a processor under control of the console.

SYSTEM STATUS AND STATISTICS

In monitoring its own operation, the DBC/1012 maintains:

- Data on system and configuration status
- Workload and performance statistics for system monitoring and capacity planning

System and Configuration Status

At any time, the DBC/1012 system is in one of the following states, as displayed at the system console:

- Off-line

  The processor to which the console is attached (usually an AMP) or the entire DBC/1012 has been powered up off- line. The DBC/1012 cannot be accessed from the host or used for processing.

- Startup

  The system is undergoing a startup and is not ready to process requests.

- Logoff

  No new sessions may log on (logons are disabled), although sessions are still logged on.

- Logoff/Quiet

  Logons are disabled and no sessions are logged on (the system is quiescent).

- Logon

  Sessions may log on (logons are enabled) and some sessions are logged on.

- Logon/Quiet

  Logons are enabled, but no sessions are logged on.

- Reconfig

  The reconfiguration program is being run.

ResUseView View

The ResUseView view summarizes information about processor utilization that is collected during a sampling period. This information may be used in capacity planning. Capacity planning involves assessing the current DBC/1012 workload in order to plan for future system requirements.

```
                  { [ TheDate, TheTime, Number, Proc,     ] }
DBC.ResUseView    { [                                     ] }
                  { [ Secs, Hits, Cpu, Disk, Host, Chan   ] }
```

The resource subsystem may be used to sample resource information at 6-, 60-, or 600-second intervals. A row for each interval is inserted into the DBC.ResUsage table for subsequent analysis.

Logging of resource utilization information is enabled by the DBC/1012 system console command,

```
SET LOG period
```

where period specifies the period in seconds (6, 60, or 600).

ACCOUNTING

Accounting facilities provided by the host computer system may be used to monitor:

- DBC/1012 system use by user groups (accounts) by recording logon/logoff activity
- Attempted security violations

A DBC/1012 accounting facility also tracks, by username and account:

- AMP usage
- DSU logical input/output activity
- DSU disk space usage

SUMMARY AND PREVIEW

This chapter discussed facilities for controlling the operation and administration of the DBC/1012. The next chapter describes the structure of the DBC/1012, including the hardware subsystems, the organization of system software, and the elements of host software.

CHAPTER 5 STRUCTURE

This chapter describes the structure of the DBC/1012 Data Base Computer, including the hardware subsystems, the organization of system software, and the elements of host software.

HARDWARE SUBSYSTEMS

There are five basic hardware subsystems in the DBC/1012 system:

1. Ynets
2. IFPs and AMPs (COPs are optional)
3. Disk Storage Units (DSUs)
4. Packaging
5. Console

Ynets

All DBC/1012 processor modules communicate via dual Ynets. As described in Chapter 2, the Ynets differ from an ordinary, passive bus in that each is an array of active logic that supplies selection and sorting functions. In contrast, an ordinary bus contains no logic. The Ynets provide a routing mechanism for the system, passing messages to and from processor modules. Processor modules are interconnected through a hierarchical network of nodes. The basic Ynet configuration or "node module", depicted in Figure 5-1, consists of seven nodes connected to the Ynet interfaces of eight processor modules. Note that the figure shows only one of the two Ynets. Each processor module is also connected to the second Ynet.

Simply explained, work steps from an IFP or COP travel up the node module hierarchy and then down, directed toward a single AMP (as in a prime index request) or fanning out to a number of AMPs (as in a request for multiple rows). Contention logic (for sorting) is applied in the upward path; in the downward path, the nodes operate in simple broadcast mode. Because all nodes in a Ynet operate from a common clock, all communication within the Ynet is synchronous. Work steps are sent over the Ynet in the form of message blocks. Blocks contain control fields that carry information about the type of block and its destination. This information enables the Ynet to direct the blocks to the processor modules and to sort the blocks so that they are received in a specified sequence.

A Ynet can be expanded beyond eight processor modules using node expansion modules. A node expansion module consists of three nodes and connects to node modules or other node expansion modules. In a given DBC/1012 Data Base Computer, up to 968 processor modules can be connected through Ynet node expansion modules.

Figure 5-2 shows node expansion modules connecting eight 8-processor-module configurations. Both Ynets are shown in this figure.

Processor Modules

A processor module consists of the following printed circuit boards:

- Processor Board
  - Central Processing Unit (CPU)
  - Numeric Processing Unit (NPU)
  - I/O Processor (IOP)
- Memory Board
  - Cache
  - Random Access Memory (RAM)
  - Erasable Programmable Read-Only Memory (EPROM)
- Ynet A Board (Ynet A Interface Processor)
- Ynet B Board (Ynet B Interface Processor)

As can be seen in Figure 5-3, except for differences in the interfaces, IFPs and AMPs use the same basic circuit boards.

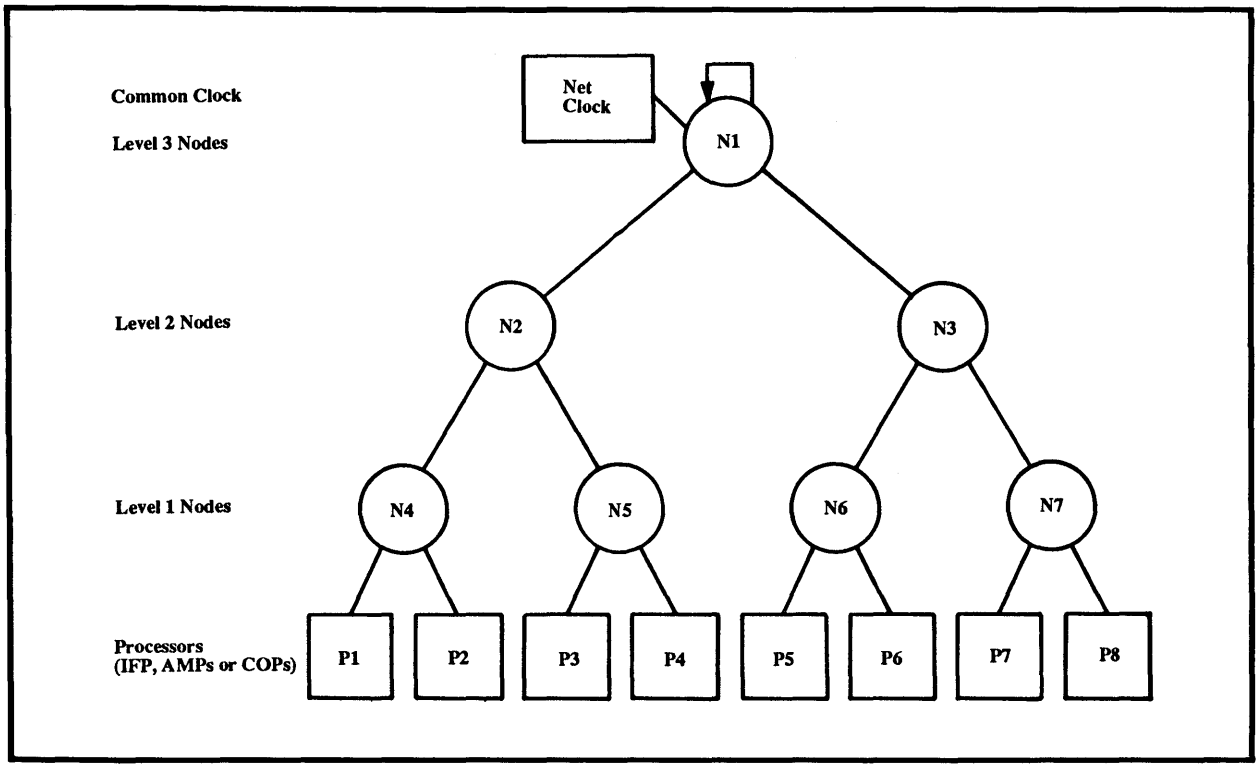

Figure 5-1. Basic Ynet Configuration

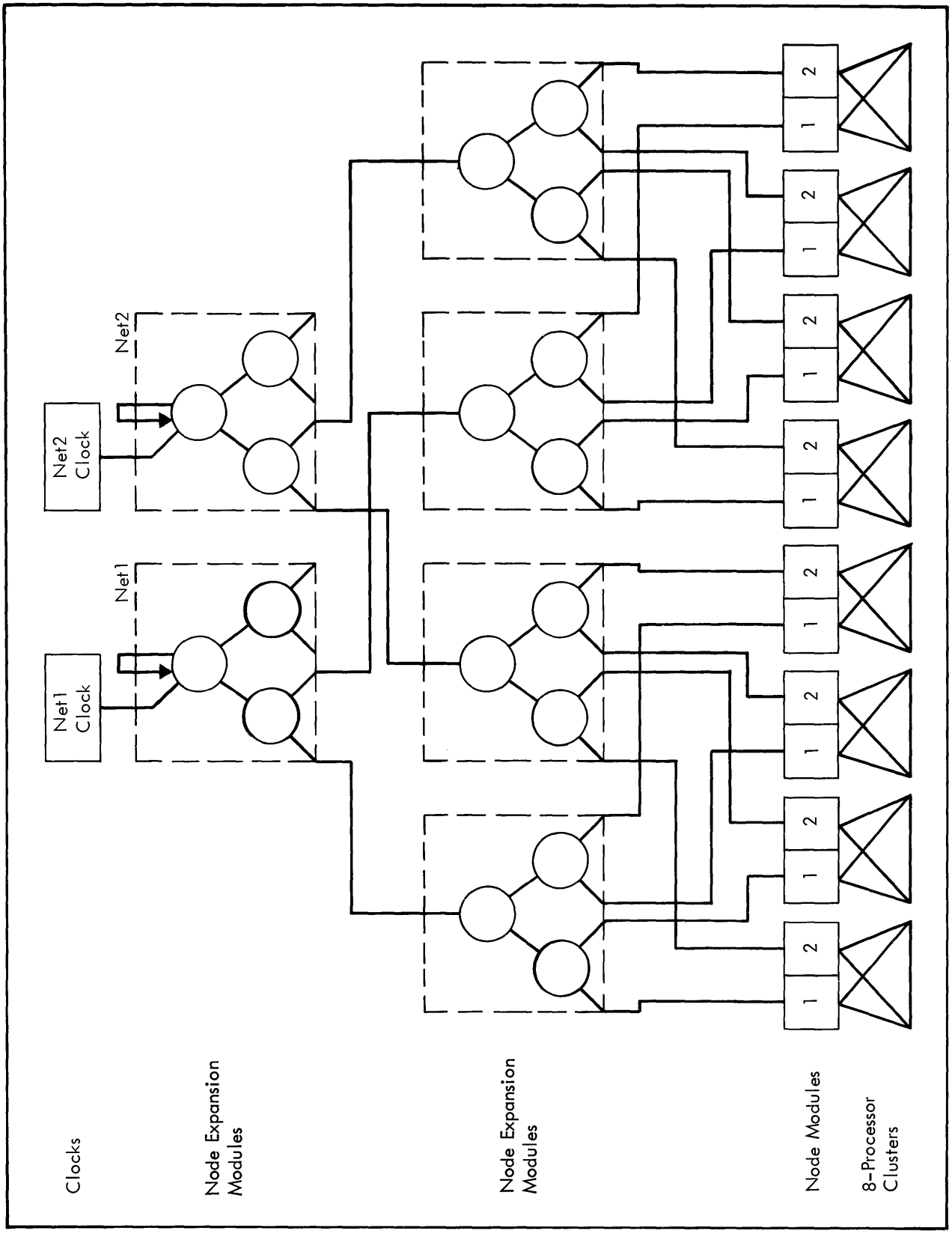

Figure 5-2. Multi-Level Network Configuration

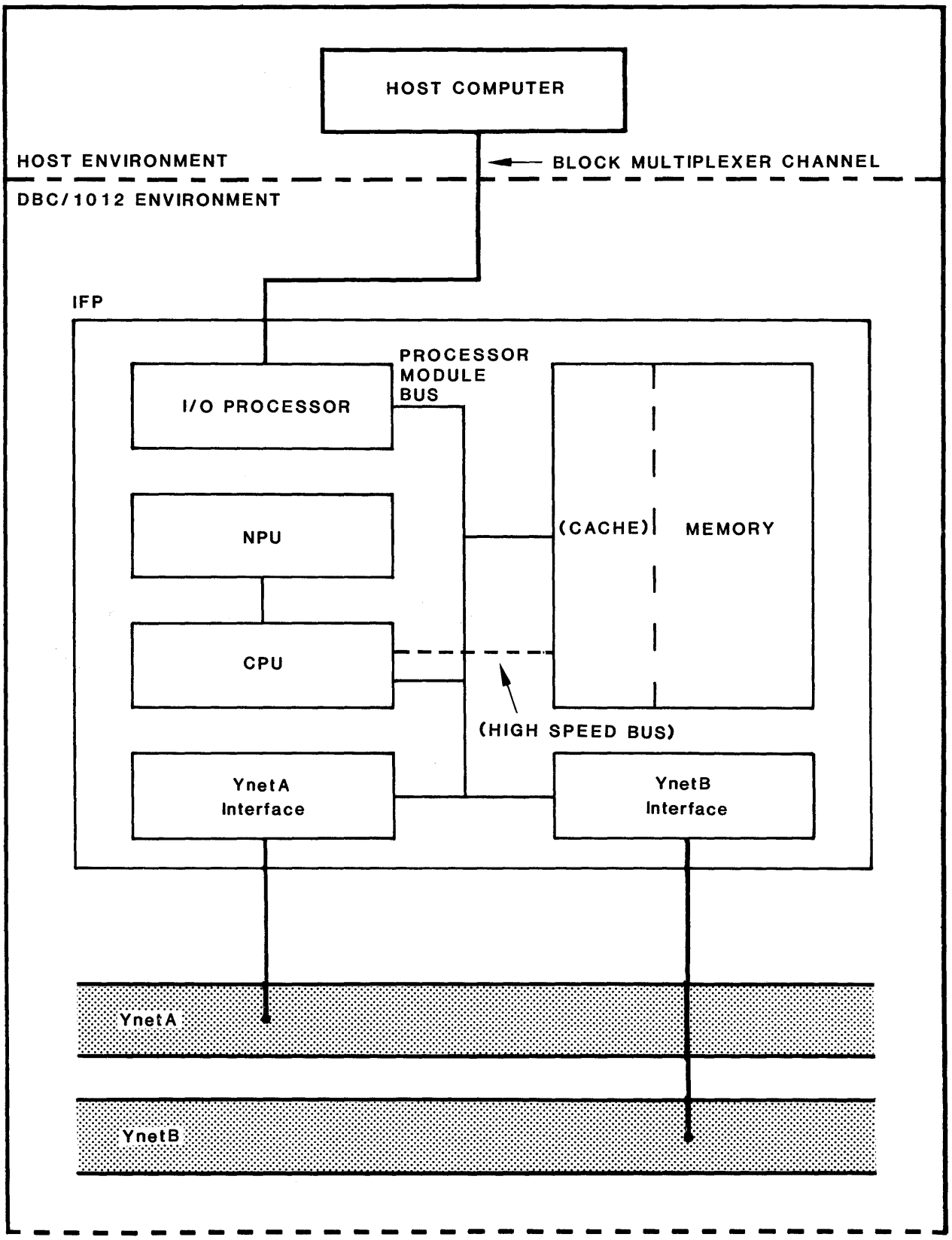

Figure 5-3. DBC/1012 Hardware Configuration (1 of 2)

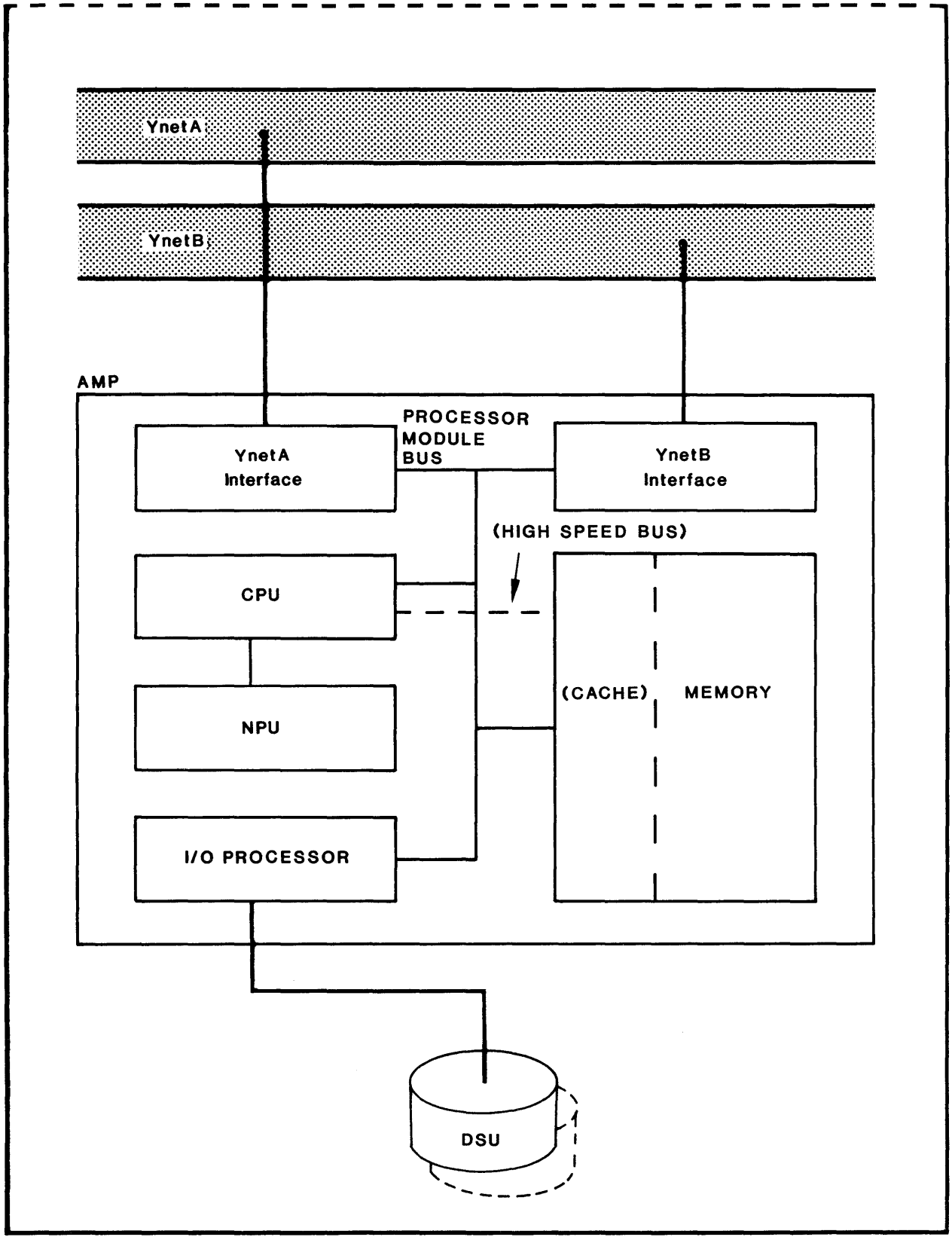

Figure 5-3. DBC/1012 Hardware Configuration (2 of 2)

The memory board contains main memory and a fast cache for performance. The memory contains 2M bytes of random access memory (RAM), optionally expandable to 4M bytes. Main memory includes error checking and correction (ECC) capability, which automatically corrects single-bit errors.

The memory board and the processor board communicate via a processor module bus. These boards also may communicate over a high-speed local bus.

The IOP in an IFP manages the interface to an IBM block multiplexer channel. The IOP in an AMP manages an industry-standard SMD interface to the DSUs.

The Ynet Interfaces control the transmission of messages to and from the Ynets. Each interface contains 32K bytes of high-speed random access memory (HSRAM), which is used to buffer the message blocks.

Disk Storage Units

Teradata provides any one of several possible DSUs. Each possible DSU varies in storage capacity and speed. For a list of currently available DSUs, see the DBC/1012 System Planning Guide, or get in touch with the National Support Center.

Packaging

DBC/1012 Data Base Computer components are housed in two types of hardware cabinets. One type of cabinet is designed to contain up to eight processor modules, the other type of cabinet may contain up to four DSUs. For measurements about currently available DSUs, see the DBC/1012 System Planning Guide, or get in touch with the National Support Center. A processor module cabinet also houses two basic Ynet node modules, one for each Ynet, and has slots reserved for two node expansion modules.

Cabinets provide for cooling as well as power supply and distribution. Front and rear doors can easily be removed to service components. Side panels also can be removed so that cabinets can be bolted together to form a continuous cabinet group.

In addition to cabinet identification on processor module and storage cabinets, the processor module cabinet's control panel contains an electrical readout to report status information. Color-coded indicators under program control report the status of each processor module in the cabinet. The indicators glow green to indicate operational status, amber to indicate service status, and red to indicate failure status.

This feature allows an operator to make an immediate visual evaluation of system status. If there is a problem with a processor module, it can be individually powered down for maintenance.

Each of the four printed circuit boards that comprise a processor module is keyed to prevent insertion into an incorrect slot within the cabinet. To aid in servicing, each board also has green, amber, and red lights to indicate operational, service, and fault status.

SOFTWARE SYSTEMS

Software falls into two categories:

1. DBC/1012 Software
   Provided by Teradata Corporation. Resides in DBC/1012 processor modules. Processes DBC/SQL requests that perform operations on the data stored in the DBC/1012 Data Base Computer.

2. Host-Resident Software
   Provided by Teradata Corporation and other software vendors supplying coordinated products. Resides on one or more of the host computers connected to a DBC/1012 Data Base Computer. Enables the host to communicate with the DBC/1012.

DBC/1012 Software

DBC/1012 software consists of the following systems:

- Teradata Operating System
- Parser System
- Data Base System

These systems are described briefly in the following paragraphs.

Teradata Operating System (TOS)

The Teradata Operating System (TOS) resides in each DBC/1012 processor module and is a virtual-memory, multi-user operating system. Some of the key TOS subsystems are:

- Scheduler

  Schedules tasks; handles calls to create, delete, start, stop, and control tasks; handles calls to start and stop timers and wait for timers to expire; performs runtime binding of program procedures; and handles basic interrupts.

- Ynet Driver

  Moves messages between a processor module's main memory and the Ynet interface module's high-speed buffer memory; handles interrupts from the two Ynet interface modules; and provides a standard method of accessing and using the unique Ynet features.

- Host Driver

  Resides in the IFP and provides control over the channel input/output processor module; causes messages to be transmitted from the message router to the host; and accepts messages from the host.

- Disk Driver

  Resides in the AMP and controls a disk drive attached to the Disk Controller.

- Message Subsystem

  Handles messages between addressable objects called "mailboxes" and tasks.

- Segment Subsystem

  Provides a uniform method of accessing segments independent of processor module type (IFP, AMP, or COP) and physical (disk) location of the segments; virtual memory management; memory protection; a method for allocating and deallocating disk space; and a locking mechanism for controlling concurrent access to segments.

- Console Command Interpreter Subsystem

  Resides in all DBC/1012 processor modules and drives a display area on the console; receives unsolicited messages and indicates their presence on a status area of the console; and accepts data from the input area of the console, interprets it, and acts on the interpretation.

Parser System

The Parser System supports DBC/SQL, the Teradata query language. Residing in each IFP, and COP the Parser System accepts a DBC/SQL statement from a user and transforms it into a series of low-level file operation steps to be processed by the Data Base Manager.

Data Base System

The Data Base System (DBS) consists of Session Control, Dispatcher, and Data Base Manager subsystems.

Session Control: Session Control resides in each IFP and COP. The major functions performed by Session Control are logon and logoff. Logon takes a textual request for session authorization, verifies it, and returns a yes or no answer. Logoff terminates any ongoing activity and deletes the session context.

Dispatcher: The Dispatcher subsystem resides in each IFP and COP and is composed of execution control and response control tasks. Execution control receives step definitions from the Parser System, transmits the definitions to an AMP(s) for processing, receives status responses from the AMP(s) as it processes the steps, and interacts with response control after the AMP(s) completes processing.

Response control is responsible for returning results of a given request to a user. After execution control has completed processing of the steps generated for a DBC/SQL request, a response is returned to the user who submitted the request.

Data Base Manager: The Data Base Manager (DBM) subsystem resides in each AMP. The DBM performs the following functions:

1. Receives steps generated by the Parser System.
2. Processes the steps by locking data base tables and creating, modifying, or deleting definitions of the tables; inserting, deleting, or modifying data rows within the tables; retrieving information from definitions and tables; and collecting accounting statistics.
3. Sends update messages to fallback AMPs.
4. Returns responses to the Dispatcher in the IFP.

A major function of the Data Base Manager is to provide the necessary transition from the logical organization of data to the physical organization of data on the DSUs. To perform this function, the manager views the disk storage available on each AMP as a number of logical contiguous cylinders, and each cylinder as a number of logical contiguous sectors.

Most of the time required to access data on disk is spent positioning the DSU read head on the desired cylinder (seek) and waiting for the desired sector to spin by the head (latency). Once the head is correctly positioned, the time required to read the sector (transfer) is relatively short.

In fact, when several contiguous sectors are read together, the average access time per sector is greatly reduced. Therefore, to improve performance, the Data Base Manager groups contiguous sectors into blocks that are read and written together.

Each row, identified by a row identifier, is stored in a block. Rows in the same table may be stored in the same block in ascending order of their row identifier, provided that their combined size does not exceed a maximum for blocks containing more than one row.

A row that exceeds the maximum is never split between blocks, but is stored in a new block. Each block is identified by the table identifier of the row(s) stored in the block and the row identifier of the first row in the block.

To remember where a row is stored on disk and to facilitate retrieval of rows, the Data Base Manager uses two levels of indexes: the master index and cylinder indexes.

Each AMP has a master index that contains a used cylinder descriptor list (UCDList). An entry in the UCDList specifies the cylinder number, the table identifier, and the row identifier of the first row stored in that cylinder. The UCDList is sorted in ascending order of table identifier and row identifier, so that a binary search can efficiently locate the cylinder where a specific row is stored.

Each cylinder has a cylinder index that contains a data block descriptors list (DBList). An entry in the DBList specifies the table identifier of the rows stored in a block, the row identifier of the first row in the block, and the disk address of the block. The DBList is sorted in ascending order of table identifier and row identifier of the first row in each block so that a binary search can efficiently locate the block where a specific row is stored.

In accessing a specific row, a binary search of the master index of an AMP determines on which cylinder the row is stored. Another binary search on the cylinder index of that cylinder then finds the block where the row is stored. Once the block is found, a sequential search through the rows in the block locates the desired row.

Host-Resident Software

Software residing in each host computer attached to a DBC/1012 Data Base Computer enables the computer to communicate with the DBC/1012. Host software, provided by Teradata Corporation, includes the following systems:

- Host System Communication Interface
- Interactive TEradata Query facility (ITEQ)
- Batch TEradata Query facility (BTEQ)
- Language preprocessor modules
- Host-resident utility programs
- Coordinated products Intellect

These systems are discussed in the following sections.

Host System Communication Interface

The Host System Communication Interface (HSCI) allows application software in a host to communicate with the DBC/1012 Data Base Computer via an input/output channel attached to an

IFP. The HSCI is composed of the Teradata Director Program (TDP), user-to-TDP communication techniques, and Call-Level Interface library.

Teradata Director Program: The Teradata Director Program (TDP) resides in a dedicated address space or virtual machine. The functions of the TDP include:

- Session establishment and termination
- Logging, verification, recovery, and restart
- Physical input to and output from IFPs and maintenance of queues
- Security

User-to-TDP Communication: User-to-TDP communication techniques enable application software in the host to communicate with the TDP. These techniques make all DBC/1012 functions available to a user application.

The user-to-TDP communication techniques are:

- SVC module (MVS)
- Cross Memory Services (MVS)
- Inter-User Communication Vehicle (VM)

Call-Level Interface Library: The Call-Level Interface (CLI) Library is an intermediate-level interface between the DBC/1012 system and an application program. CLI service routines are link-edited with an application program or dynamically loaded at runtime to:

- Provide a set of common functions for managing communications with the DBC/1012 Data Base Computer
- Provide a convenient interface for applications that do not use the COBOL or PL/1 Preprocessor
- Allow application programs written in languages that have a call statement (e.g., PL/1, Pascal, FORTRAN, and Assembler) to access data on the DBC/1012 Data Base Computer
- Support batch, TSO, CMS, and CICS environments

Interactive TEradata Query Facility

The Interactive TEradata Query facility (ITEQ) resides in the host under the MVS Time Sharing Option (TSO) or the VM Conversational Monitoring System (CMS). As discussed in Chapter 3, ITEQ provides a user-friendly interface for users of the DBC/1012 Data Base Computer and includes functions for:

- Terminal control
- Entering, editing, and executing DBC/SQL statements
- Formatting output and writing reports

- Requesting reference information

ITEQ uses TSO or CMS facilities for communicating with end users via IBM 3270 terminals or 3270-compatible display devices.

Batch TEradata Query Facility

The Batch TEradata Query facility (BTEQ) enables a user to execute a series of DBC/SQL requests in batch or line-by-line interactive mode. BTEQ provides functions for:

- Entering more than one DBC/SQL statements per request
- Reading data from, and writing data to host files
- Using more than one session for a job
- Formatting output and writing reports

BTEQ may also be used on-line under TSO or CMS.

Language Preprocessors

As discussed in Chapter 3, the COBOL Preprocessor or PL/1 Preprocessor accepts an input file containing DBC/SQL and preprocessor statements intermixed with source language statements, adds calls to procedures in the Call-Level Interface (CLI) library, and produces an output file containing a source language program acceptable by a COBOL compiler. A language preprocessor runs as a batch job on the host and operates independently of the DBC/1012 Data Base Computer.

Host-Resident Utility Programs

Host-resident utility programs allow a user to:

- Archive a data base in a file on the host and restore a data base from an archive file
- Perform batch select, insert, update, and delete operations on a data base
- Load data from an input file in a host system to newly created DBC/1012 tables

(See Chapter 4, General Purpose Utility Programs.)

The utility programs operate in both MVS and VM environments and accept utility control statements from any device, data set (MVS), or input file (VM) supported by either environment. The programs operate on tables within a single data base identified by a utility program statement.

The programs read and write data sets/files on the host system using the sequential access method. Data control blocks for input and output have predefined names (MVS ddnames or VM FILEDEFs). The job control language of the host operating system is used to associate these control blocks with the input or output media and to define the security to be applied to data sets/files.

SUMMARY AND PREVIEW

This chapter described the structure of the DBC/1012, including hardware subsystems, the organization of system software, and the elements of host software. The next chapter lists specifications for operation of a DBC/1012 system, including host computer and environmental requirements, and gives configuration data for systems with multiple processor modules.

CHAPTER 6 OPERATING AND CONFIGURATION SPECIFICATIONS

This chapter lists specifications for system operation, including host computer and environmental requirements, and gives configuration data for DBC/1012 Data Base Computers with a number of processors.

OPERATING CHARACTERISTICS

Operating characteristics for the DBC/1012 Data Base Computer are given below:

- Performance:

  Processing power increases in a linear fashion, depending on number of processors. Up to 968 processors (IFPs, AMPs, and COPs) per system

- Data Storage Capacity:

  Number of data bases: 32,000

  Number of tables per data base: 32,000

  Number of columns per table: 256

  Row length (bytes): 32,000

  Name size (characters): 30

  Number of fields per index: 16

  Primary indexes per table (clustered): 1

  Secondary indexes per table (nonclustered): 16

  Total data capacity: up to a terabyte (trillion bytes) depending on customer's DBC/1012 configuration. Check with a Teradata Sales department or the Teradata National Support Center.

HOST REQUIREMENTS

Requirements for a mainframe computer are given below:

- Hardware IBM 370 Models 148, 155 with DAT, 158, 168, 303X, 308X, 309X, 43XX, PCM (NASCO, Amdahl, Magnuson, CDC, IPL, etc.) with one or more available block multiplexer channels

- Operating System OS/VS-MVS Release 3.8 and above, including MVS/SP release 1 or 2 (SP 1.3 required for CICS support), and MVS/XP or VM/SP release 3 and above.

- TP Monitors CICS 1.6, and above TSO, and/or CMS

- Application Languages

  COBOL Preprocessor: ANSI COBOL, X3.23-1974

  PL/1 Preprocessor: IBM PL/1

Call-Level Interface: Any language with CALL statement (e.g., COBOL, FORTRAN, Pascal, PL/1, Assembler).

LAN-ATTACHED HOST REQUIREMENTS

The requirements for LAN-attached hosts are provided below.

- Hardware

  IBM PC (must be model XT or AT) or AT&T 3B2 (must be model 300 or 400 with at least 1 M byte)

- Software

  The IBM PC must run PC-DOS or MS-DOS at level 3.1 or 3.2. For the IBM PC to use TCP/IP protocol, it must have an Excelan EXOS3 205 Intelligent Ethernet Controller board. For the IBM PC to use ISO/OSI protocol, it must have the Intel OpenNet PCLink4 hardware/software package.

  The AT&T must run UNIX5, release 5.2. For the AT&T 3B2 to use the TCP/IP protocol, the network controller board must be a 3BNet feature card with the Enhanced TCP/IP WIN6 3B software package.

SYSTEM AND CONFIGURATION SPECIFICATIONS

System specifications and configuration data for typical DBC/1012 installations are given below.

- System Specifications:
  - Processor Cabinet:

    Capacity: up to 8 processors

    Dimensions: 27 inches wide by 32.5 inches (without doors and status panel) deep or 35.25 inches (with doors and status panel) deep by 60 inches high.

    Weight: 662 pounds (maximum)

  - Storage Cabinet:

    Capacity: maximum number ofg DSUs depends on DSU type.

    Dimensions: The same dimensions as those for the processor cabinet.

    Clearance: 4 feet, front and rear

    Weight: 1126 pounds (maximum)

  - Intercabinet aisle width: 56 inches (minimum)
- Ynet Cabling:
  - Between node modules and processors: backpanel wiring
  - Between node modules (or node expansion modules): 25-conductor pair, flat coaxial cable

- Power Cabling and Sequencing:

  An adjacent grouping of one processor cabinet and two disk cabinets can be accommodated on a single 30-amp AC branch circuit. A designated storage cabinet is plugged into the main outlet. That cabinet daisy-chains power to the other cabinets in the grouping.

- Cabinet Positioning and Floor Area:

  - System with 512 processors: minimum required floor area is 56 feet by 61 feet.
  - System with 968 processors: minimum required floor area is 77 feet by 84 feet.

POWER SPECIFICATIONS

Each cabinet has the following electrical service requirements:

- Voltage: 208VAC, Y-connected
- Voltage tolerance: +8, -10%
- Frequency: 50/60Hz
- Rated line current: 24A
- Wires: 4 plus ground
- Circuit ampacity: 30A per phase (tandem grouping of one processor cabinet and one or two storage cabinets)
- Power cord:
  - Length: 14.5ft
  - Plug type: NEMA L21-30P (supplied)
  - Receptacle type: NEMA L21-30R (not supplied; one receptacle is required for each tandem grouping)
- Power control: compatible with IBM power control requirements

ENVIRONMENTAL REQUIREMENTS

Each cabinet has the following environmental requirements:

- Temperature:
  - Air inlet: 10 degrees C to 32 degrees C (50 degrees F to 90 degrees F)
  - Maximum change: 10 degrees C per hour (18 degrees F per hour)
- Heat dissipation:
  - Processor cabinet: 2400 BTU per hour plus 750 BTU per hour per processor -- 8400 BTU per hour (maximum configuration)

- Storage cabinet: (Fujitsu) 890 BTU per hour plus 1800 BTU per hour per disk drive -- 8090 BTU per hour (maximum configuration); (CDC) 890 BTU per hour plus 887 BTU per hour per disk drive -- 7986 BTU per hour (maximum configuration)

- Relative humidity and other atmospheric conditions:
  - Air inlet: 20% to 80%
  - Change: 10% per hour (noncondensing)
  - Altitude: 10,000 feet (maximum)
  - Noise: 65dB (estimate for a tandem grouping of three fully configured cabinets)
  - Vibration: 0.2G (5Hz to 50Hz); 1.0G (50Hz to 500Hz)
  - Dust: 0.168mg/m^3 (Stearic Acid Standard)

GLOSSARY

Words that are italicized in the explanations are themselves defined in the glossary. For your convenience, frequently misunderstood terms are flagged with "cf" (compare) citations to alert you to similar terms for comparison. Parenthetical notes alert you to similar or contrasting terms. Finally, in accordance with other documents in the DBC/1012 series, all command or statement keywords, modifiers, and other reserved words are shown in all capital letters.

The following convention was observed in alphabetizing entries in the glossary; a blank sorts before any letter, and a dash sorts after any letter.

ABORT. [in ITEQ]
A command that immediately terminates any DBC/SQL statement in execution in the DBC/1012 and returns control to the user. [in DBC/SQL] A statement that aborts a transaction in progress backs out changes to the data base only if the conditional expression associated with the abort statement is true.

Access Module Processor. See *AMP module.*

administrator. [in Data Dictionary/Directory]
A user (or userid) responsible for allocating resources of the DBC/1012 to a community of users.

AMP module. [in hardware]
A complete Access Module Processor, consisting of an AMP processor/controller board, a memory board, two Ynet interface boards, a cable adapter board, and DC power supply.

application program. [in host software]
A program that performs a particular function or set of functions that the user desires to perform.

backout. [process]
The process by which changes to a data base area reversed after a transaction has been aborted so that the data base is restored to its state as of the beginning of the transaction. Cf: *ABORT, transaction.*

batch. [in host software]
Application programs that run in a background mode where their execution is not under the direct, moment-to-moment control of a user are said to run in batch. Batch programs or jobs are often run for an extensive period of time and may be part of a data processing operation that is routinely run every day, week, month, or year. See also *BTEQ.*

block. [in host software]
A collection of records, rows, or packets that is manipulated as a unit, typically for efficiency of execution.

block multiplexer channel. [in hardware]
One of the types of IBM channels. The IFP supports only the block multiplexer channel.

BTEQ. [software component] Batch TEradata Query.
A Teradata-supplied *host-resident application program* that enables a user to execute in batch mode a series of DBC/SQL requests. BTEQ can read from or write to host data sets and use more than one DBC/1012 *session.* BTEQ has more sophisticated report formatting features than has *ITEQ.*

buffer. [in host software]

An area in main memory used for constructing and processing messages.

byte. [in DBC/SQL]
A data type in which information is stored as a string of zero or more 8-bit elements without translation. Also, one such 8-bit element.

cabinet. [in hardware]
An enclosure that houses the various subsystems of the DBC/1012 for physical support, interconnection, and protection from environmental hazards. The DBC/1012 has two types of cabinets: processor cabinets and storage cabinets. See *processor cabinet, storage cabinet.*

Call-Level Interface. [software component]
A set of entry points provided by Teradata to facilitate low-level communication between programs running on the host and the DBC/1012. The CLI (Call-Level Interface) is used by Teradata-supplied application programs such as ITEQ, BTEQ, the COBOL Preprocessor (runtime environment), the Bulk Loader, and host utility programs. CLI is available to customer-written application programs coded in any language that supports a call statement and separate compilation of source modules.

channel. [in hardware]
The means by which an IBM central processor is attached to peripheral units; the path by which data is transferred between the host and the DBC/1012.

CICS. [in host software]
The Customer Information Control System, an IBM program product that acts as a supervisory or "monitor" program for application programs that are optimized for real-time interaction with users to perform relatively constrained information processing tasks. CICS runs under control of the MVS operating system (there is also a DOS version of CICS) and communicates with a network of terminals. Application programs written for CICS must use only CICS system services and must obey a number of other constraints imposed by the CICS environment.

CLI. See *Call-Level Interface.*

cluster. [in hardware]
A group of *AMP modules* (typically in different cabinets in a large system) that provides *fallback* capability (which see) for each module. A large system typically consists of many clusters of two to sixteen AMP modules each.

CMS. [in ITEQ] Conversational Monitor System.
A multi-user monitor subsystem that runs under the VM operating system.

COBOL Preprocessor. [software component]
A program that facilitates productive design, coding, and testing of user application programs written in COBOL that interact with the DBC/1012.

column. [in DBC/SQL]
In the relational model, data bases consist of one or more tables. In turn, each table consists of fields, organized into one or more columns by zero or more rows. All of the fields of a given column share the same attributes.

command. [in ITEQ]
Any operation that is processed by the ITEQ program, as distinguished from a DBC/SQL statement, which is sent to the DBC/1012 for processing.

console. [in hardware]

A subsystem consisting of a video display, a keyboard, diskette drives, a printer, and an electronics module that enables operator interaction with the DBC/1012. The console supports operator communications during *on-line* operation of the DBC/1012. The console also supports loading and control of diagnostic programs run by the field engineer in an *off-line* DBC/1012 system or *processor mudule.*

cylinder. [in hardware]
The tracks of a Disk Storage Unit (DSU) that can be accessed from a given position of a read/write head.

cylinder index. [in hardware]
For each cylinder of a Disk Storage Unit (DSU), a listing of descriptors for data blocks (DBList) stored on the cylinder. Each entry specifies the table identifier of the rows stored in the block, the row identifier of the first row in the block, and the starting address and size of the block.

data base. [in DBC/SQL]
A related set of tables that share a common space allocation and owner.

data block. See *block.*

data definition. [in DBC/SQL]
The statements and facilities that manipulate (create, modify, and delete) data base structures and the Data Dictionary/Directory information kept about these structures. Cf: *data manipulation.*

Data Dictionary/Directory. [in DBC/SQL]
The information automatically maintained by the DBC/1012 about all of the tables, views, macros, data bases, and users known to a DBC/1012 system, including information about ownership, space allocation, accounting, and access right relationships between those objects. Data Dictionary/Directory information is updated automatically during the processing of DBC/SQL data definition statements, and is used by the *Parser* to obtain information needed to process all DBC/SQL statements.

data manipulation. [in DBC/SQL]
The statements and facilities that change the information content of the data base. These statements include SELECT, INSERT, UPDATE, and DELETE.

DBC. Data Base Computer.

DBC/SQL Parser. [software component]
The DBC/SQL language processor, resident in the *IFP*, that translates DBC/SQL statements entered by a user into the *steps* that accomplish the user's intentions.

DBC/SQL statement. [in COBOL Preprocessor, PL/1 Preprocessor]
A statement in the DBC/SQL language that is processed by the DBC/1012.

DBMS. Data Base Management System.
Typically, a software package that runs on a host computer. Cf: *DBC.*

deadlock. [in DBC/SQL]
A condition in which two or more transactions are competing for locks on the same resources in such a way that none of the deadlocked transactions can make any progress without resources held by another transaction. The DBC/1012 detects deadlocks and resolves them by aborting one or more of the transactions that is causing the deadlock condition.

delimiter. [in DBC/SQL]

A punctuation mark or other special symbol that separates one clause in a DBC/SQL statement from another, or that separates one DBC/SQL statement from another.

diagnostic. [in hardware]
A program that exercises a hardware subsystem to verify that it is operating correctly, and if not, to provide information to assist the field engineer in isolating the failure to a field-replaceable unit.

Dispatcher. [software component]
A program that executes in each IFP to coordinate the flow of processing within the DBC/1012.

DSU. [in hardware] Disk Storage Unit.

dump. [operation]
A function provided to create an archival copy, typically on tape, of a data base, part of a data base, or a collection of data bases stored on the DBC/1012. Cf: *restore.*

dynamic. [in hardware]
A form of buffer or memory management that acquires buffers of varying sizes from the free space within the address space of a partition.

ECC. [in hardware] See *error correction*

end user. [in Data Dictionary/Directory]
An ordinary user of the DBC/1012, as opposed to a *supervisory user* or an *administrator.* An end user cannot create a subordinate user or data base, except within the end user's user space.

error correction. [in hardware]
Logic that uses additional memory bits to correct errors which can be caused when one or more bits of main or secondary storage become unreliable. Error correction logic improves the reliability of the system and the integrity of stored data.

failure. [in DBC/SQL]
Any condition that precludes complete processing of a DBC/SQL statement. Any failure will abort the current transaction.

fallback. [in DBC/SQL]
The ability of the DBC/1012 to maintain an extra copy of every row of a table, so that if one or more AMPs are down (but not more than one AMP per cluster), all requests against the table can still be satisfied.

field. [in DBC/SQL]
The basic unit of information stored in the DBC/1012. A field is either null or else has a single numeric or string value. See also *column, data base, row, table.*

firmware. [in hardware]
Programming that is permanently fixed into a subsystem, as opposed to a *software system,* which is replaced without altering a hardware configuration.

hashing. [in DBC/SQL]
A way of mapping data records to various physical storage areas. In the DBC/1012, hashing is used to determine at which AMP a given *row* is to be stored. The DBC/1012 hashing scheme operates in conjunction with the *Ynet.*

hierarchical. [generic term]

An organization of entities, such as data records, in which some "superior" or "parent" entities are related to one or more "subordinate" or "child" entities; also pertains to any data base management system that uses or describes information in a hierarchical form, such as IMS/VS. Cf: *inverted, network, relational.*

host. [generic term]
A general-purpose computer attached to a DBC/1012 that can execute *application programs* that access and manipulate information on the DBC/1012.

host-resident. [generic term]
Pertaining to a system or application program that executes on a *host* computer.

Host System Communication Interface. See HSCI.

host utility program. [software component]
One of several programs provided by Teradata that executes on the *host* computer to provide archiving of information from the DBC/1012 to tape and/or restoring archived information to the DBC/1012, bulk loading data to the DBC/1012 from an input file on the host, and fast loading data to the DBC/1012 from a host input file.

HSCI. [software component]
The Host System Communication Interface, which consists of the *Teradata Director Program, Call-Level Interface,* and *user-to-TDP communication techniques.* These HSCI components provide the means by which user- and Teradata-written *application programs* communicate with the DBC/1012.

HSRAM. [in hardware]
High Speed Random Access Memory. A part of the Ynet Interface that stores *packets* to be transmitted on or received from the *Ynet.*

IFP. [in hardware] Interface Processor. See *IFP module.*

IFP module. [in hardware]
A complete IFP processor module, consisting of an IFP board, memory board, two Ynet Interface boards, an IFP cable adapter board, associated cables, etc.

index. [in DBC/SQL]
A means of ordering and locating rows on disk for efficient access and processing. Cf: *primary index, secondary index, unique.*

inverted. [generic term]
A form or organization of records in a data base management system in which extensive use is made of *secondary index* capability to provide alternative access paths to records. Each secondary index is also known as an "inversion." Cf: *hierarchical, network, relational.*

ITEQ. [software component] Interactive TEradata Query.
A Teradata-supplied *host-resident application program* that provides a user at a 3270-series terminal the ability to directly use all of the query, data manipulation, and data definition capabilities of DBC/SQL without any additional programming. ITEQ includes extensive features for the presentation of data, control of the 3270 screen, and generation of reports.

join. [in DBC/SQL]
A select operation that combines information from two or more tables to produce a result.

key. [in DBC/SQL]

The value(s) of the *index field*(s) used to locate a *row* within the DBC/1012.

keyword. [in DBC/SQL]
A string of characters that has a special meaning in the DBC/SQL language. A keyword cannot be used as a *name.*

lock. [in DBC/SQL]
The right to use a *data base, table,* or *row* for a particular purpose (such as to read or write) with the assurance that other activities in the system cannot alter the object in a way that could affect the outcome of the activity that holds the lock. Users who do not require data consistency may use a lock specifically for access.

logical. [generic term]
Pertaining to an entity, record, or grouping of data that is treated as a unit by a software program, as opposed to an entity that is treated as a unit by hardware.

LOGOFF. [in ITEQ]
A command that terminates an ITEQ session with the DBC/1012. Any file or print data sets are closed, and control of the 3270 screen is returned to TSO. Cf: *LOGON*

LOGON. [in ITEQ]
A command that initiates an ITEQ session with the DBC/1012. If a startup string is present for the user, it will be executed by the LOGON command.

macro. [in DBC/SQL]
A set of DBC/SQL statements that are stored in the DBC/1012 and that can be executed by a single EXECUTE statement. Each macro execution is implicitly treated as a *transaction.*

memory. [in hardware]
A printed wiring board assembly that is part of every processor module and provides read-only and random access memory for the processor module. Cf: *processor module.*

memory board. [in hardware] See *memory.*

message. [in host software]
The basic unit of information interchange between an *application program* and the DBC/1012. An application program sends messages via a *session* to the DBC/1012 in half-duplex fashion, and must wait for a response message from the DBC/1012 before sending another message on the same session. Messages consist of one or more parcels, which are logical subdivisions of a message. Messages are sent in one or more *packets* on the *channel* between the *Teradata Director Program* and the *IFP.*

module. [in software maintenance]
A unit of software that typically performs one function or a set of closely related functions and is the smallest unit of software that can be replaced.

multitasking. [in host software]
The ability to share the resources of a computer, operating system, or address space among several tasks, or "threads" of execution, where the state of each task is, in general, independent of the state of other tasks.

MUX. [in hardware] Abbreviation for multiplexer.

MVS. [software component]
Multiple Virtual Storage. One of the primary operating systems (or "control programs") for medium and large IBM computers.

name. [in DBC/SQL]
A word supplied by the user that is used to refer to an object, such as a *column, data base, macro, table, user,* or *view.*

network. [generic term]
A method of organizing records in a data base management system in which relationships between one record and another are represented by pointers. A pointer is a part of a record that gives the address, typically on disk, at which the next related record can be found. The data base thus consists of a network of records and pointers. This organizational form is also known as a "plex" structure, a "navigational" data base, or a CODASYL model data base. Cf: *hierarchival, inverted, relational.*

null. [in DBC/SQL]
The absence of a value for a *field.*

off-line. [generic term]
A state of a system or component in which it is logically disconnected from its normal operating environment so that special functions such as service, maintenance, and/or diagnostics can be performed on the component or system in isolation from the rest of the environment.

on-line. [generic term]
Any state in which a system is available for users to enter ordinary requests, and in which all normal functions are provided.

operand. [generic term]
Values, given by constants or variables, that an *operator* acts upon to produce a result.

operator. [in DBC/SQL]
A symbol or *keyword* that specifies an operation to be done on the values of the *operands* (if any).

owner. [in DBC/SQL]
The user who has the ability to grant or revoke all access rights on a data base to and from other users. By default, the creator of the data base is the owner, but ownership can be transferred from one user to another by the GIVE statement.

packet. [in host software]
The smallest unit of data sent on the *channel* between the *TDP* and the *IFP.* A *message* consists of one or more packets. A packet is a purely physical division of a message. Packets are distinct from *parcels,* which are logical subdivisions of a message.

parameter. [in DBC/SQL]
A variable *name* in a *macro* for which an argument value is substituted when the macro is executed.

parcel. [in host software]
A logical *part* of a message. A parcel indicates what kind of information DBC/SQL statements, result rows, failure codes, etc.) the message contains.

Parser. See *DBC/SQL Parser.*

PL/1 Preprocessor. [software component]
A program that facilitates productive design, coding, and testing of user application programs written in PL/1 that interact with the DBC/1012.

Preprocessor. See *COBOL Preprocessor, PL/1 Preprocessor.*

primary index. [in DBC/SQL]
An *index* used to determine on which AMP a *row* is stored. It is also the means of locating a row on the AMP's disk. Access through the primary index is generally the most efficient means of locating a row.

prime key. [in DBC/SQL]
The set of values of a *primary index.*

privilege. [in DBC/SQL]
The right of a specified *user* to enter a specified DBC/SQL statement (such as CREATE, SELECT, GRANT, etc.) against a specified *data base, macro, table, user, or view.*

processor cabinet. [in hardware]
An enclosure that houses up to eight *processor modules.* The cabinet provides cooling, AC/DC power distribution, cabling, and so on. See also *storage cabinet.*

processor module. [in hardware] See *AMP module, IFP module.*

program. [generic term]
A unit of software that performs a set of operations to satisfy the needs of users or other programs. A program consists of one or more *modules.*

protocol. [generic term]
A set of rules that govern the communication between two or more entities, such as processors, programs, or systems, including the formats of messages that flow among the entities.

query. [in DBC/SQL]
A DBC/SQL statement, particularly a SELECT statement.

queue. [generic term]
A list of requests for the use of some resource of the system, such as processor time, memory, access to a peripheral device, or a lock on a resource.

RAM. [in hardware] See *random access memory.*

random access memory. [in hardware]
A quality of a memory device that allows data to be written in or read from the memory through direct locating, rather then locating through reference to other data in the memory.

recovery. [process] See *backout.*

relational. [generic term]
A data base management system in which complex data structures are represented as simple, two-dimensional tables consisting of columns and rows. Cf: *hierarchical, inverted, network.*

request. [in host software]
A *message* sent from an *application program* to the DBC/1012.

response. [in DBC/SQL]
The result (*success* or *failure*) generated when the DBC/1012 processes a DBC/SQL statement.

restart. [process]

The process by which *on-line* operation of the DBC/1012 is resumed following a system error condition such as a hardware failure, a software protocol failure, or loss and restoration of AC power.

restore. [software component]
A function provided by a *host utility program* that re-creates a data base on the DBC/1012 from archived dump tapes. Cf: *dump.*

result. [in DBC/SQL]
The information returned to the user to satisfy a request made of the DBC/1012. Results may include a return code, activity count, error message, warning message, title information, and/or rows from a *spool file.*

ROM. [in hardware] Read-Only Memory.
Storage for *firmware* on the *memory board.*

row. [in DBC/SQL]
The *fields,* whether *null* or not, that represent one entry under each *column* in a *table.* The row is the smallest unit of information operated on by *data manipulation statements.* Cf: *column, data base, field, table.*

secondary index. [in DBC/SQL]
An *index* on a *column* or group of columns other than those used for the *primary index.* A secondary index results in storage of extra information ordered on the secondary index columns, which can be used to more rapidly locate information in the DBC/1012.

separator. [in DBC/SQL]
A character or group of characters that separates *words* and special symbols in DBC/SQL. Blank and comments are the most common separators.

session. [in host software]
A logical connection between an *application program* on a host and the DBC/1012 that permit the application program to send one *request* to and receive one *response* from the DBC/1012 at a time.

software system. [in software]
A major class or package of software. Software systems are configured into specific host-DBC/1012 systems to satisfy customer requirements.

spool file. [in DBC/SQL]
A file on the DBC/1012 that holds the results of the processing of DBC/SQL statement(s) until they can be examined by the user or *application program.*

startup string. [in DBC/SQL]
One or more *DBC/SQL statements* that are executed automatically when a *user* performs a *LOGON.*

statement. [in COBOL Preprocessor, PL/1 Preprocessor]
A preprocessor statement, *DBC/SQL statement,* COBOL statement, or PL/1 statement.

statement. [in ITEQ] A *DBC/SQL statement.*

statement. [in DBC/SQL]
A request for processing on the DBC/1012 that consists of a *keyword* verb and optional phrases and operands and that is processed as a single entity.

status panel. [in hardware]

An assembly on the top front of the *processor cabinet* that visually displays information about the state of the *processor modules* and *Ynets* in the cabinet.

step. [in DBC/SQL]
A unit of work that does some or all of the processing of a single *DBC/SQL statement.* A step is created by the *DBC/SQL Parser* and sent to the *AMPs* by the *Dispatcher.* Step for a given statement are processed serially: processing of a step is not initiated by the Dispatcher until the previous step has been completed by all affected AMPs.

storage cabinet. [in hardware]
An enclosure housing up to four Disk Storage Units (DSUs). See also *processor cabinet.*

success. [in DBC/SQL]
A *parcel* that is returned from every *DBC/SQL statement* that executes to normal completion.

supervisory user. [in Data Dictionary/Directory]
A user who has been delegated authority by the administrator to further allocate DBC/1012 resources such as space and the ability to create, drop, and modify users within a subset of the overall user community. Cf: *end user, administrator.*

system console. [in hardware] See *console.*

system view. [in Data Dictionary/Directory]
A view that permits *end users, supervisory users,* and *administrators* to obtain appropriate information about *data bases, macros, tables, users, views,* and the relationships among them.

table. [in DBC/SQL]
A set of two or more *columns* with zero or more *rows* that consist of *fields* of related information. See also *data base.*

TDP. [software component] *Teradata Director Program.*

tera. [generic term]
A prefix that means "trillion" (1,000,000,000,000).

Teradata Director Program. [software component]
A program that manages communication between *application programs* and the DBC/1012. It is a part of the *Host System Communication* Interface.

title. [in DBC/SQL]
A string used as a column heading in a report. By default it is the name of the column, but a title can also be explicitly declared by a TITLE phrase.

TOS. [software component] Teradata Operating System.
The control program control program that executes in every *on-line* DBC/1012 *processor module.*

transaction. [in DBC/SQL]
A set of DBC/SQL statements starting with a BEGIN TRANSACTION statement and finishing with an END TRANSACTION statement that is performed as a unit. Either all of the statements are executed normally or else any changes made during the transaction are *backed out* and the remainder of the statements in the transaction are not executed.

TSO. [in ITEQ] Time Sharing Option.
A multi-user monitor subsystem that runs under the MVS operating system.

type. [in DBC/SQL]
An attribute of a *column* that specifies the representation of data values for fields in that column. DBC/SQL data types include numerics and strings.

unique. [in DBC/SQL]
A property of an *index* that specifies that no two rows of a table are allowed to have the same *key* value for that index. The default is non-unique, which permits duplicate keys.

update operation. [in DBC/SQL)
An operation that alters the contents of a data base, such as an INSERT, DELETE, or UPDATE *data manipulation* statement.

user. [in DBC/SQL]
A *data base* associated with a person who uses the DBC/1012. The data base is used to store the person's private information and to gain access to other DBC/1012 data bases.

user-to-TDP communication technique. [software component]
One of the ways in which application software communicates with the Teradata Director Program. Techniques include: SVC and commonly addressable utility routines (under MVS); IBM Cross Memory Services routines and routines that reside in common storage (under MVS); Inter-User Communication Vehicle (under VM).

view. [in DBC/SQL]
An alternate way of organizing and presenting information in the DBC/1012. A view, like a table, has *rows* and *columns.* However, the rows and columns of a view are not directly stored in the DBC/1012, but are derived from the rows and columns of tables (or other views) whenever the view is referenced.

virtual. [generic term]
A system resource that can be used by programs but that is not an actual hardware device in the system. A "virtual" resource is simulated by software and "real" hardware resources.

VM. [software component] Virtual Machine.
One of the primary operating systems (or "control programs") for medium and large IBM computers.

word. [in DBC/SQL]
One or more contiguous, nonblank, alphabetic, numeric characters ($, , #).

Ynet. [in hardware]
The interconnection network that provides for high-speed communications among the *processor modules.* of a DBC/1012 system.

INDEX

A

ABS function...see arithmetic functions
access lock...4-3
Access Module Processor...see AMP
ACCOUNT parameter...4-9
account number...4-1
accounting...3-23, 4-1, 4-9, 4-13, 5-8
administrator, data base...1-2, 2-10, 3-33, 4-6
after images...4-2
aggregate operators...3-6, 3-14, 3-16
ALTER TABLE statement...3-11
AMP...1-4, 1-5, 2-1, 2-6, 2-7, 2-8, 2-9, 2-10, 2-11, 4-5, 4-10 4-11, 4-12, 5-8
    clustering...2-10, 4-11
    Data Base Manager...2-8, 5-8,
    Disk Interface...2-8
    fallback...2-9, 2-10
    reconfiguration...4-10, 4-11, 4-12
    recovery...4-5
    synchronization...2-8
    Ynet Interface...2-8
apostrophe...3-4
application program...2-1, 2-3, 3-1, 3-31
arithmetic functions...3-8
    absolute value (ABS)...3-8
    base 10 logarithm...3-8
    exponentiation (EXP)...3-8
    natural logarithm...3-8
    NULLIFZERO...3-8
    result data types...3-8
    square root...3-8
    ZEROIFNULL...3-8
arithmetic operators...3-6
AS parameter...4-9

B

batch mode...2-1, 3-27, 5-11
Batch TEradata Query...see BTEQ
BEGIN/END TRANSACTION statements...4-2
BETWEEN n AND n ...3-2
    see also Data Types
block...5-9
block multiplexer channel...2-3
blocking messages...3-33
BTEQ...1-3 3-1, 3-27, 3-28, 3-29, 5-9, 5-11
    formatting data...3-27, 3-28
    report...3-27, 3-28
bulk load utility...4-10
Bus, Interprocessor...2-1
BYTE(n) data type...3-2
BYTEINT data type...3-2

C

cabinets...5-6, 6-3
Call-Level Interface...see CLI
capacity planning...4-12
CASESPECIFIC...3-2
change image...4-3
CHAR data type...3-2
character string constants...3-4, 3-5
checkpoint capability...4-3
CICS...2-1, 5-10
CLI...2-1, 2-3, 3-1, 3-38, 3-39, 5-11
    library...5-11
clustering...2-10, 4-11
COBOL Preprocessor...2-3, 3-38, 5-11
column...1-6, 1-7, 1-8, 3-2, 3-3, 3-5, 3-12, 3-14, 3-18
Communications Processor commands...see COP Interface
    entering... 3-23
    format... 3-24, 3-25
comparison operators...3-6
COMPRESS phrase...3-2, 3-10
configuration specifications...6-2
concurrency control...4-2, 4-3
Console Command Interpreter Subsystem...5-8
console printer...2-11
constants...3-4
COP Interface...1-4, 1-5, 2-1, 2-6, 2-7, 2-11, 3-32
    components...3-32
copy area...2-9
CREATE statement
    DATABASE...4-9
    INDEX...3-11
    MACRO...3-19, 3-20
    TABLE...3-9, 3-10
    USER...4-9
    VIEW...3-12
creator...3-20
creating data bases...4-6, 4-9
creating users..4-6, 4-9
cross-memory services...2-1
cursor...3-23
Customer Information Control System...see CICS
cylinder...5-8, 5-9
cylinder index... 5-9

D

data base administrator...1-2, 2-10, 3-33, 4-6
data base, entity...3-5
data base management systems...1-1
Data Base Manager...2-8, 5-8, 5-9
data base creator...3-20

data base owner...4-6, 4-7
data base ownership structure...4-6, 4-7
Data Base System...1-6, 5-8, 5-9
    Data Base Manager...5-8, 5-9
    Dispatcher...5-8
    Session Control...5-8
data definition...3-8
Data Dictionary/Directory...1-2, 2-5, 3-1, 3-33
    information levels...3-33
    querying...3-34
    views...3-34
        administrator...3-35
        end user...3-34
        supervisory user...3-35
        system...3-34
data protection...4-1, 4-2
data types...3-2, 3-6
    BYTE(n)...3-2
    BYTEINT...3-2
    CHAR...3-2
    DATE...3-2
    DECIMAL...3-2
    FLOAT...3-2
    INTEGER...3-2
    LONG VARCHAR...3-2
    SMALLINT...3-2
    VARBYTE(n)...3-2
    VARCHAR...3-2
DATE data type...3-2
DBC/SQL...1-3, 3-1, 3-37, 3-38,
    arithmetic...3-8
    data controls...3-20
    data definitions...3-8
    data references...3-5
    data types...3-2
    HELP...3-20
    expressions...3-5
    macros ...3-19
    manipulations...3-19
    operations...3-1
    selecting data...3-13
    statements...3-3
DBList...5-9
DBMS...see data base management systems
DECIMAL data type...3-2
DEFAULT...3-3
DEFAULT phrase...3-10
DELETE statement...3-19, 3-20, 4-10
delimiters...3-4
Department table...3-9, 3-10, 3-11
diagnostics...4-11
Disk Driver...5-7
Disk Interface...2-7
Disk Storage Unit...see DSU
disk space...2-9
Dispatcher...2-6, 5-8
display area...3-23
DROP statement
    INDEX...3-13
    MACRO...3-20
    TABLE...3-13
    VIEW...3-13
DSU...2-1, 2-6, 2-8, 2-9, 2-10
dual image...4-3
DUMP Utility...4-10

E

Employee table...1-11, 3-9, 3-10, 3-11, 3-17,
    3-18, 3-19, 3-20
Employee Info view...3-12
end user...3-34, 3-35
entering statements and commands..3-23
environmental requirements...6-4
error checking and correction...5-7
exclusive lock...4-2
EXECUTE command...3-20, 3-21, 3-24
EXP function...see arithmetic functions
expandability...1-4, 1-7
expressions...3-7

F

fallback copy...3-10
FALLBACK parameter...3-10, 3-11, 4-9
Fast Load utility...4-10
field...1-7
FLOAT data type...3-2
FOCUS Interface...3-30, 3-31
format commands...3-24, 3-25
format, screen...3-22, 3-23
FORMAT...3-3
FORMAT phrase...3-10
formatting data...3-24
FROM parameter...4-9

G

GRANT statement...4-8
granting privileges...3-24, 4-7, 4-8,
GROUP BY clause...3-16, 3-17

H

hardware subsystems...5-1
    Disk Storage Units...5-6
    Packaging...5-6
    Processor Modules...5-1, 5-2, 5-6
    Ynets...5-1, 5-2, 5-6
hardware testing...4-15
hashing...4-14
HAVING clause...3-16, 3-17

HELP statement...3-20, 3-21
high-speed random access memory...5-6
Host Driver...5-7
host relief...1-1
host requirements...6-2
Host System Communication Interface...see HSCI
host utilities...5-15
Host-Resident Software...2-1, 5-9
    BTEQ...5-11
    HSCI...5-9
    ITEQ...5-10
    language preprocessors...5-11
    utility programs...5-11
HSCI...2-1, 5-9

I

IDEAL Interface...3-30, 3-31
IFP...1-4, 1-5, 1-6, 2-1, 2-3, 2-5, 2-6, 2-7, 2-8, 3-32, 4-5, 5-8
    Dispatcher...2-6, 5-8
    Host Interface...2-1, 2-5
    Parser...2-6, 5-8
    recovery...4-5
    Session Control...2-5, 5-8
    Ynet Interface...2-6, 2-8
logging image...4-3
IN Operator...3-7
INDEX clause..3-10
index
        cylinder...5-8, 5-9
        master...5-8, 5-9
        primary...3-10, 3-11, 4-10, 6-1
        secondary...3-10, 3-11, 6-1
input area...3-23
INSERT statement...3-10, 3-18, 3-20, 4-10
inserting
    values...3-18
    rows...3-18
INTEGER data type...3-2
INTELLECT Interface...3-30
Interactive TEradata Query...see ITEQ
Interface Processor...see IFP
INTERSECT operator...3-7
ITEQ...1-2, 2-1, 2-3, 3-1, 3-21, 3-22, 3-23, 3-24, 3-25, 5-9
    display area...3-22
    formatting data...3-24, 3-25
    printed report...3-24, 3-25, 3-26, 3-27
    screen format...3-22
    status area...3-23

J

join...1-18, 3-12, 3-17
    see also WHERE statement

K

keywords...3-3, 3-4

L

LAN Interface...see COP Interface
language preprocessors...2-3, 3-37, 3-38, 5-11
    see also COBOL, PL/1 Preprocessor
LIKE Operator...3-7
locking data...4-2
    access lock...4-2, 4-3
    exclusive lock...4-2
    read lock...4-2, 4-3
    write lock...4-2
logarithm...see arithmetic functions
logging on
    ITEQ...3-23
logical operators...3-7
LONG VARCHAR data type...3-2

M

macro...3-19, 3-20
manipulation oparations...3-18
master index...5-9
memory board...5-2, 5-6
message block...5-1
Message Subsystem...5-7
MINUS operator...3-7
modularity...1-1
modulus...3-6
MS-DOS/PC-DOS...3-39

N

names...3-3, 3-5
nesting subqueries...3-17
netwark protocols...3-32
node expansion modules...5-1
NOMAD2 Interface...3-30, 3-31
NOT IN Operator...3-7
NOT NULL...3-2
NULL...3-6
NULLIFZERO...see arithmetic functions

O

operations, arithmetic...see arithmetic functions
operators
    aggregate...3-6
    arithmetic...3-6
    comparison...3-6
    logical...3-7

set...3-7
string matching...3-7
ORDER BY clause...3-15, 3-16
ASC...3-16
DESC...3-16
owner...4-6,4-7
see also creator
ownership, user and data base...3-30. 3-35, 4-6

P

packaging...5-6
parallel processing...1-1
Parser...1-6, 2-1, 2-5, 2-6, 5-8
password...4-1
path...1-8, 1-9
PC/SQL-link Interface...3-30, 3-31
permanent journaling...4-3, 4-4
PERMANENT parameter...4-9
Personnel data base...3-9, 3-10, 3-28, 3-34
picture...4-3
PL/I Preprocessor...2-3, 3-37, 3-38
planning...4-12
portable...3-33
power specifications...6-3
preprocessors...see language preprocessors
primary index...3-10, 3-11, 4-10 6-1
printer, console...2-11
Printed Report...3-24, 3-25, 3-28
privileges...4-7, 4-8
Processor Modules...5-1
circuit boards...5-2
processor utilization, obtaining information about...4-12, 4-13
product interfaces...3-30
FOCUS Interface...3-31
IDEAL Interface...3-31
INTELLECT Interface...3-30
NOMAD2 Interface...3-30, 3-31
PC/SQL-link...3-31
productivity...1-3, 1-4
protection...4-1, 4-2
protocol
network...3-32
session...4-1

R

random access memory...5-6
Range Constraint...see BETWEEN n AND n
read lock...4-2, 4-3
RCC...3-36, 3-37
reconfiguration...4-10, 4-11, 4-12
recovery...1-3, 4-5
Recovery Control Catalog...see RCC
recovery control user views...3-35
redundant storage...4-4,
relational data base...1-1, 1-2, 1-6, 1-8, 1-9
column...1-7, 1-8
field...1-7,
field value...1-7
row...1-7, 1-8
table...1-7, 1-8
view...1-8
reliability...1-1, 1-3
REPLACE statement
MACRO...3-20
VIEW...3-13
report formatting...3-28
report writer features...3-24
response control task...5-8
restore...4-3
RESTORE utility...4-10
ResUseView view...4-12, 4-13
REVOKE statement...4-7, 4-8
revoking privileges...4-7
rollback...1-3, 4-3
roll backward...4-4
roll forward...4-4
row...1-6, 1-7, 1-8, 3-12, 3-18, 3-19, 5-9

S

Scheduler...5-7
screen...3-21, 3-22, 3-23
secondary index...3-10, 3-11, 6-1
sectors...5-8, 5-9
Segment Subsystem...5-7
SELECT keyword...3-3, 3-4
SELECT statement...3-7, 3-13, 3-14,3-16, 3-17
set a report format...3-25
separators...3-4
Session Control...2-5, 5-8
sessions...3-21, 2-23, 4-1
set operators...3-7
sharability...1-1, 1-2
single image...4-3
single logging...4-3
software...5-7
Data Base System...5-8
Parser System...5-8
TOS...5-7
Software Systems...5-6, 5-7, 5-9
DBC/1012 Software...5-7, 5-8, 5-9
Host-Resident Software...5-9, 5-10, 5-11
sort direction...3-16
sorting...3-15, 3-16
space allocation and access control...4-6
SPOOL parameter...4-9
square root...see arithmetic functions
status area...3-22,
statements, entering...3-23
storage capacity...2-9, 2-10

string constants...3-4, 3-4
string matching operators...3-7
supervisory user views...3-35
synchronization...2-8
system accounting...4-13
system area...2-9
system console...2-11, 4-11, 4-12
    operation...4-11
system facilities...4-1
    data protection...4-1
    maintenance...4-11
    reconfiguration...4-10, 4-11
    session protocol...4-1
    space allocation and access control...4-6
    user logon interface...4-6
    user security interface...4-5
system specifications...6-2
system state...4-12
system views...3-1, 3-34

T

table...1-7, 1-8, 2-6, 3-1, 3-2, 3-3, 3-5, 3-8, 3-9, 3-10, 3-11, 3-17, 5-9
task...5-8
TDP...2-1, 2-3, 2-5, 2-6, 5-10
tdpid...4-1
Teradata Director Program...see TDP
Teradata logo...2-6
Teradata Operating System...see TOS
third-party products...3-30
Time Sharing Option...see TSO
TITLE...3-3
TITLE phrase...3-10, 3-24, 3-25
TOS...1-6, 5-7
    Console Command Interpreter Subsystem...5-8
    Disk Driver...5-7
    Host Driver...5-7
    Message Subsystem...5-7
    Scheduler...5-7
    Segment Subsystem...5-7
    Ynet Driver...5-7
transaction concept...4-2
transaction management...4-4, 4-5
transient journal...4-3
trouble log...4-11
TSO...2-2, 5-10, 5-11

U

UCDList...5-9
UNIQUE PRIMARY INDEX clause...3-10
UNION operator...3-7
UNIX...3-39
UPDATE statement...3-11, 3-19, 3-20, 4-10
UPPERCASE...3-2
used cylinder descriptor list...see UCDList
user identification...see username
user security interface...4-5
User-to-TDP Communication...see UTC
user views
    administrator...3-35
    end user...3-34, 3-35
    recovery control...3-36
    supervisory...3-35
UTC...2-1
username...4-1
utility programs...4-10, 5-11

V

VALUES keyword...3-18, 3-20
VARBYTE(n) data type...3-2
VARCHAR data type...3-2
view...1-10
    creating a view...3-12
    create statement...3-12
    DROP statement...3-13
    REPLACE statement...3-13
    ResUseView...4-12, 4-13
    user...3-34, 3-35, 3-36
    virtual table...3-1
    virtual machine...2-1, 2-3, 5-10

W

WHERE clause...3-14, 3-15, 3-17
    in a DELETE statement...3-19
WITH clause...3-24, 3-25
workstation...see COP Interface
write lock...4-2

Y

Ynet...1-4, 1-5, 2-1, 2-5, 2-6, 2-7, 2-8, 2-11, 4-5 5-1, 5-2, 5-6, 5-7
    basic configuration...5-2
    cabling...6-2
    Driver...5-7
    multi-level network configuration...5-3
    node expansion modules...5-1, 6-2

Z

ZEROIFNULL...see arithmetic functions