APPENDIX

1

IDMS Essentials

Introduction

This chapter introduces SAS System users to Computer Associates' Integrated Database Management System (CA-IDMS). It focuses on the terms and concepts that help you access CA-IDMS files with SAS/ACCESS software.

If you want more information on a CA-IDMS concept or term, see the books listed in "CA-IDMS Documentation" at the end of this chapter.

Data Dictionaries and the DDS

CA-IDMS enables you to build one or more databases using a data dictionary. A *data dictionary* is itself a CA-IDMS database that contains all the data and system definitions for one or more databases.

A data dictionary is divided logically into areas. The information is organized into entity types, which correspond to the main data processing components, such as elements, records, files, programs, and users. Data dictionaries monitor most aspects of the database environment, from tracking the status of terminals, systems, and users to being a central resource of information about the system and providing security. Some large information systems use multiple dictionaries; for example, a system may have one dictionary for each division of a company.

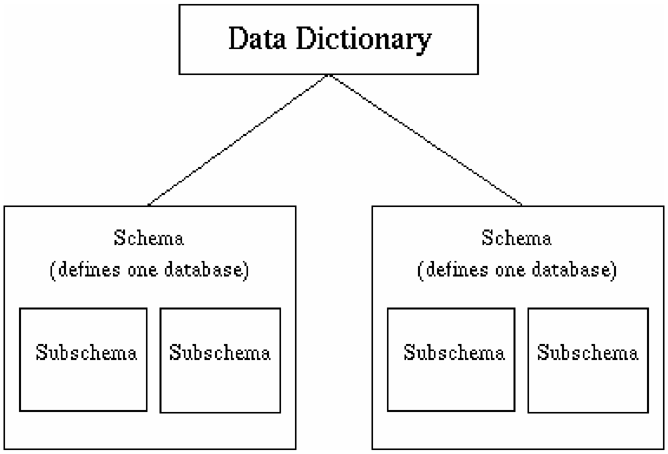
A *database administrator* (DBA) manages and maintains the data dictionaries and the entire CA-IDMS system. DBA duties often include programming systems, managing resources, monitoring the system's performance, and overseeing its security. The DBA has a key role in the SAS/ACCESS interface to CA-IDMS, which is explained in more detail in this chapter.

Within a CA-IDMS data dictionary are the definitions for a database's schema and subschema. A *schema* describes the contents and structure of a single database, including all of the records and sets that are necessary to define its data elements and data relationships.

A *subschema* is a subset of a schema that is used by programs at runtime. It consists of all the data elements, records, sets, and areas that are defined in the schema or a subset thereof. It includes database records and can include logical records as well as logical-record paths (defined below). The DBA defines logical records and their paths in the subschema before application programs are coded and executed.

Figure A1.1 on page 86 illustrates the relationships among the data dictionary, schemas, and subschemas.

Figure A1.1 Data Dictionary, Schemas, and Subschemas



CA-IDMS provides two operating environments, or modes, for accessing data dictionaries and databases. In the *central version*, multiple concurrently executing programs access the database(s) through one shared copy of the database management system (DBMS). The central version controls concurrent updating of the database by multiple users in order to maintain database integrity.

In *local mode*, one program at a time accesses the database through a dedicated copy of the DBMS. You cannot run local mode against a database at the same time that the central version is accessing it.

A *Distributed Database System (DDS)* distributes data storage and processing functions among several systems. These systems can execute on one or more machines and at one or more sites. Each system is a node in the DDS configuration. A central version specifies which node within the DDS system to access.

Networks and Sets

Each CA-IDMS database consists of database records that are grouped into record types. A *record type* consists of the record's name, all of its elements, and the elements' attributes, such as data types and sizes. These record types are linked together through different logical groups called sets. Sets are defined to the schema.

A *set* is a logical relationship established between two or more named record types. One record type is the owner of the set and the other record types are members. Record types can belong to more than one set, so a record type can be both an owner of one set and a member of another. That same record type can also be a member of more than one set. These sets and their interweaving relationships make up a *network* and give CA-IDMS its network capabilities.

To move through the database, each record type contains pointers to other record types in its set or sets. *Pointers* identify the next record in the set and link the records together in a chain. There are three kinds of pointers:

Next pointer (required pointer)
: points to the next record type in the set, regardless of whether the record type is an owner or a member of the set.

Prior pointer
: works the same way as the Next pointer except that it points to the prior record type.

Owner pointer
: points from a member record type to the owner record type.

Through these pointers, a program can navigate through the network and travel a specific path through one or many sets.

The database administrator is responsible for defining record types and sets in the schema.

CA-IDMS Documentation

You may find the following Computer Associates' Release 12.0 CA-IDMS documentation helpful while you are using the SAS/ACCESS interface to CA-IDMS. Refer to these manuals for information on your CA-IDMS system and DML application programming.

- □ *System Operations*
- □ *Database Administration*
- □ *Security Administration*
- □ *System Generation*
- □ *Features Summary*
- □ *Messages and Codes*
- □ *System Tasks and Operator Commands*
- □ *Utilities*
- □ *Database Design*
- □ *Quick Reference*
- □ *Programming Quick Reference*
- □ *Master Index*
- □ *Glossary*
- □ *Navigational DML Programming*

The correct bibliographic citation for this manual is as follows: SAS Institute Inc., *SAS/ ACCESS® Interface to CA-IDMS Software: Reference, Version 8*, Cary, NC: SAS Institute Inc., 1999. pp. 104.

SAS/ACCESS® Interface to CA-IDMS Software: Reference, Version 8

ISBN 1–58025–547–7

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st printing, October 1999

The Institute is a private company devoted to the support and further development of its software and related services.