ASG
Software Solutions

ASG-DataManager™ Siemens UDS Interface

Version 2.5
Publication Number: DMR0200-25-SUDS
Publication Date: January 1986

The information contained herein is the confidential and proprietary information of Allen Systems Group, Inc. Unauthorized use of this information and disclosure to third parties is expressly prohibited. This technical publication may not be reproduced in whole or in part, by any means, without the express written consent of Allen Systems Group, Inc.

© 1998-2002 Allen Systems Group, Inc. All rights reserved.
All names and products contained herein are the trademarks or registered trademarks of their respective holders.

ASG Worldwide Headquarters Naples Florida USA | asg.com | info@asg.com
1333 Third Avenue South, Naples, Florida 34102 USA Tel: 239.435.2200 Fax: 239.263.3692 Toll Free: 800.932.5536 (USA only)

ASG Documentation/Product Enhancement Fax Form

Please FAX comments regarding ASG products and/or documentation to (239) 263-3692.

Company Name	Telephone Number	Site ID	Contact name

Product Name/Publication	Version #	Publication Date
Product:		
Publication:		
Tape VOLSER:		

Enhancement Request:

© 2002 Allen Systems Group, Inc.
All names and products are trademarks or registered trademarks of their respective holders.

ASG Support Numbers

ASG provides support throughout the world to resolve questions or problems regarding installation, operation, or use of our products. We provide all levels of support during normal business hours and emergency support during non-business hours. To expedite response time, please follow these procedures.

Please have this information ready:

- Product name, version number, and release number
- List of any fixes currently applied
- Any alphanumeric error codes or messages written precisely or displayed
- A description of the specific steps that immediately preceded the problem
- The severity code (ASG Support uses an escalated severity system to prioritize service to our clients. The severity codes and their meanings are listed below.)
- Verify whether you received an ASG Service Pack for this product. It may include information to help you resolve questions regarding installation of this ASG product. The Service Pack instructions are in a text file on the distribution media included with the Service Pack.

If You Receive a Voice Mail Message:

1. Follow the instructions to report a production-down or critical problem.
2. Leave a detailed message including your name and phone number. A Support representative will be paged and will return your call as soon as possible.
3. Please have the information described above ready for when you are contacted by the Support representative.

Severity Codes and Expected Support Response Times

Severity	Meaning	Expected Support Response Time
1	Production down, critical situation	Within 30 minutes
2	Major component of product disabled	Within 2 hours
3	Problem with the product, but customer has work-around solution	Within 4 hours
4	"How-to" questions and enhancement requests	Within 4 hours

ASG provides software products that run in a number of third-party vendor environments. Support for all non-ASG products is the responsibility of the respective vendor. In the event a vendor discontinues support for a hardware and/or software product, ASG cannot be held responsible for problems arising from the use of that unsupported version.

Business Hours Support

Your Location	Phone	Fax	E-mail
United States and Canada	800.354.3578	239.263.2883	support@asg.com
Australia	61.2.9460.0411	61.2.9460.0280	support.au@asg.com
England	44.1727.736305	44.1727.812018	support.uk@asg.com
France	33.141.028590	33.141.028589	support.fr@asg.com
Germany	49.89.45716.222	49.89.45716.400	support.de@asg.com
Singapore	65.6332.2922	65.6337.7228	support.sg@asg.com
All other countries:	1.239.435.2200		support@asg.com

Non-Business Hours - Emergency Support

Your Location	Phone	Your Location	Phone
United States and Canada	800.354.3578		
Asia	65.6332.2922	**Japan/Telecom**	0041.800.9932.5536
Australia	0011.800.9932.5536	**Netherlands**	00.800.3354.3578
Denmark	00.800.9932.5536	**New Zealand**	00.800.9932.5536
France	00.800.3354.3578	**Singapore**	001.800.3354.3578
Germany	00.800.3354.3578	**South Korea**	001.800.9932.5536
Hong Kong	001.800.9932.5536	**Sweden/Telia**	009.800.9932.5536
Ireland	00.800.9932.5536	**Switzerland**	00.800.9932.5536
Israel/Bezeq	014.800.9932.5536	**Thailand**	001.800.9932.5536
Japan/IDC	0061.800.9932.5536	**United Kingdom**	00.800.9932.5536
		All other countries	1.239.435.2200

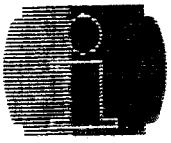
ASG Web Site

Visit http://www.asg.com, ASG's World Wide Web site.

Submit all product and documentation suggestions to ASG's product management team at http://www.asg.com/asp/emailproductsuggestions.asp.

If you do not have access to the web, FAX your suggestions to product management at (239) 263-3692. Please include your name, company, work phone, e-mail ID, and the name of the ASG product you are using. For documentation suggestions include the publication number located on the publication's front cover.

THE MANAGER FAMILY



MANAGER FAMILY – MSP brings you a unique offering – the MANAGER Family of dictionary driven software products. MANAGER Products and their dictionaries may be integrated to maximize the return on your software and on your dictionary investment.

USER CONFIGURED – Every MANAGER Product consists of a nucleus with optional additional facilities. You select the options you need, and MSP supplies your chosen configuration.

WORLD-WIDE SUPPORT – All MANAGER Products are backed by comprehensive support, documentation, education and user group activities on a world-wide basis.

CONTROLMANAGER™

The dictionary driven End User Facility for the MANAGER Family of products

DATAMANAGER®

The dictionary driven Data and Information Resource Management system, winner of the ICP $50 Million Award

DESIGNMANAGER™

The dictionary driven Interactive Data and Enterprise Modeling system for logical and physical database design

METHODMANAGER™

The dictionary driven Information Engineering Methodology which automatically navigates you and your users through the systems life cycle

PROJECTMANAGER®

The dictionary driven Project Resource Management and Budgetary Control system

SOURCEMANAGER™

The dictionary driven On-line Application Development system

TESTMANAGER®

The Program/Structure Testing and Test Data Generation system

The MSP MANAGER FAMILY – designed for the future; implemented today.

MSP TRADEMARKS

The following are trademarks of Management Systems and Programming Ltd:

- CONTROLMANAGER
- DATAMANAGER® (registered as a United States trademark at the U.S. Patent and Trademark Office. Registered in Canada. Registered in the United Kingdom.)
- DESIGNMANAGER
- METHODMANAGER
- PROJECTMANAGER® (registered as a United States trademark at the U.S. Patent and Trademark Office. Registered in Canada.)
- SOURCEMANAGER
- TESTMANAGER® (registered as a United States trademark at the U.S. Patent and Trademark Office. Registered in Canada.)
- TEXTMANAGER
- InfoBank
- InfoDictionary
- InfoSystem
- InfoView.

TRADEMARK ACKNOWLEDGEMENTS

The following trademarks may appear in any one or more MSP Publications:

- IBM is a registered trademark of International Business Machines Corporation
- The LIBRARIAN is a registered trademark of Applied Data Research Inc
- PANVALET is a trademark of Pansophic Systems Inc
- SYSTEM 2000 is a registered trademark of SAS Institute Inc.

PREFACE

This manual is one of a series describing DATAMANAGER, the dictionary driven data and information resource management system developed by MSP for use on Siemens 7.5xx and 7.7xx series with the operating system BS2000. This manual describes DATAMANAGER's facility for interfacing with the Siemens Universal Database System (SUDS), a facility which enables the user to include SUDS database data definitions in the data dictionary.

MSP provides Full Support for MANAGER Products in a Siemens BS2000 environment, where the release of BS2000 in use is supported by Siemens at the date of publication of this Preface. Particular MANAGER Products selectable units interface with certain Siemens software products that run within a BS2000 environment and/or with other vendor software: the Release Levels at which these products interface with MANAGER Products and the support provided by MSP are documented in Appendix 2 to the MANAGER Products Installation in Siemens Environments manual.

This edition relates to Release 5.0.0 and subsequent releases of DATAMANAGER, and to Release 1.0.0 and subsequent releases of CONTROLMANAGER.

It is assumed that the reader has a knowledge of DATAMANAGER to the extent covered by the User's Guide, and is familiar with SUDS.

Chapter 1 of this manual summarizes the interface between DATAMANAGER and SUDS.

Chapter 2 discusses briefly the concept of SUDS databases and illustrates how they can be defined to DATAMANAGER.

Chapter 3 gives the specifications of the DATAMANAGER data definition statements for SUDS databases.

Chapter 4 describes the interface between SUDS and the DATAMANAGER Source Language Generation facility.

Chapter 5 specifies the correspondences that relate SUDS data description and storage structure specifications to DATAMANAGER data definitions.

The notation used in the specification of DATAMANAGER statements is described on page viii.

To assist you to make full use of this manual, the Contents table commencing on page vi is supported by a combined keyword index and usage index under the heading 'Usage Directory' at the back of the manual. The Usage Directory provides a means of accessing information by word occurrence or by function.

For the storage and job control requirements for installing and running DATAMANAGER with the SUDS interface, reference should be made to the MANAGER Products Installation in Siemens Environments manual.

A range of manuals is available covering the MANAGER Family of Products. Details of the manuals and other documentation available are published every six months (at the end of June and the end of December) in the MSP Documentation Bulletin, which is distributed to all Users.

CONTENTS

PREFACE		v
NOTATION FOR STATEMENT FORMATS		viii
CHAPTER 1	DATAMANAGER SUDS INTERFACE FACILITIES	1-1
CHAPTER 2	SIEMENS UDS DATABASES AND DATAMANAGER	2-1
2.1	SUDS Concepts and Definitions	2-1
2.2	Example of a SUDS Database Defined Using DATAMANAGER Member Types	2-4
2.3	Further Considerations	2-10
2.3.1	Introduction	2-10
2.3.2	Dynamic Sets and Temporary Areas	2-10
2.3.3	Privacy Locks and Privacy Keys	2-10
CHAPTER 3	DATAMANAGER DATA DEFINITIONS FOR SIEMENS UDS DATABASES	3-1
3.1	Introduction	3-1
3.2	The SUDS-DATABASE Data Definition Statement	3-2
3.3	The SUDS-AREA Data Definition Statement	3-4
3.4	The SUDS-RECORD Data Definition Statement	3-5
3.5	The SUDS-SET Data Definition Statement	3-10
3.6	The SUDS-VIEW Data Definition Statement	3-18
3.7	The SUDS-SUBSCHEMA Data Definition Statement	3-21
3.8	System, Program and Module Data Definition Statements for a SUDS Environment	3-24
3.8.1	Introduction	3-24
3.8.2	Specification of the PROCESSES Clause	3-24
CHAPTER 4	SUDS SOURCE LANGUAGE GENERATION FROM DATAMANAGER	4-1
4.1	Introduction	4-1
4.2	Specification of the PRODUCE Command for SUDS Source Language Generation	4-2
CHAPTER 5	SUDS-DATAMANAGER CORRESPONDENCE TABLES	5-1
5.1	Introduction	5-1
5.2	Schema DDL Correspondence Table	5-2
5.3	Schema SSL Correspondence Table	5-5
5.4	Subschema DDL Correspondence Table	5-7
USAGE DIRECTORY		UD-1
AMENDMENT RECORD		AR-1

LIST OF FIGURES

(Figures are numbered within Chapters).

Figure 1.1	Hierarchy Relating SUDS-specific and Basic DATAMANAGER Member Types	1-2
Figure 2.1	Example of a SUDS Schema for a Mail Order Company	2-5
Figure 3.1	Format of the SUDS-SET Data Definition Statement	3-6
Figure 3.2	Format of the SUDS-RECORD Data Definition Statement	3-11
Figure 3.3	Format of a Content Declaration in a SUDS-RECORD or SUDS-VIEW Data Definition or in the Data Definition of a GROUP Directly or Indirectly CONTAINed in a SUDS-RECORD or SUDS-VIEW	3-12

NOTATION FOR STATEMENT FORMATS

In all MANAGER Products manuals, the following notation is used in the specification of statement formats (for commands and member definition statements):

- all words printed in capitals are statement identifiers or keywords that must be present in full or truncated form in the circumstances stated in the statement specification. The extent beyond which a word must not be truncated is indicated by underlining of the characters that must be retained.

- all words printed in lower case are variables for which the user must substitute a value consistent with the specification

- material enclosed in square brackets [ ] is an option which may be included or omitted as required

- braces { } indicate that a choice must be made of one of the options enclosed within them

- three periods or full stops ... indicates that the material they immediately follow may be repeated. Where ... immediately follows a closing square bracket or brace, the material that can be repeated is bounded by that square bracket or brace and the corresponding opening square bracket or brace. If material can be repeated only a limited number of times the repetition permitted is stated in the specification.

- other punctuation marks and symbols must be coded as shown, subject to the implications of any square brackets or braces enclosing them; except that where a single quote, ', is shown, a double quote, ", can alternatively be used, provided that the opening and closing quotes of any pair of quotes are the same character (single quote or double quote).

CHAPTER 1 DATAMANAGER SUDS INTERFACE FACILITIES

This manual is intended for users of the DATAMANAGER Siemens Universal Database System (SUDS) Interface facility. In a SUDS environment, the interface provides you with the following capabilities:

- the ability to define SUDS databases to DATAMANAGER, to hold the definitions in a data dictionary, and to process them using standard MANAGER Products commands

- the ability to define processing views of SUDS databases, to hold the definitions of these views in a dictionary, and to process them using MANAGER Products commands.

These capabilities are provided by means of six additional (SUDS-specific) DATAMANAGER member types and a SUDS-specific version of the PROCESSES clause in the DATAMANAGER SYSTEM, PROGRAM, and MODULE member type data definitions. Four of the additional member types are required to define SUDS databases, including:

- SUDS-DATABASE
- SUDS-AREA
- SUDS-SET
- SUDS-RECORD.

SUDS-AREA and SUDS-RECORD members are used to define the physical content of a database. The logical characteristics are defined by SUDS-SET members. In the hierarchy of member types, SUDS-AREA and SUDS-SET appear (at the same level) between SUDS-DATABASE and SUDS-RECORD.

The other additional member types, that is:

- SUDS-SUBSCHEMA and
- SUDS-VIEW,

are used, together with the PROCESSES clause of the DATAMANAGER member types indicated above, to provide processing views of a SUDS database. The SUDS-VIEW member type appears between SUDS-SUBSCHEMA and SUDS-RECORD in the member type hierarchy. The complete hierarchy of relevant member types in a SUDS environment is pictured in figure 1.1.

The syntax of the DATAMANAGER data definition statements for the SUDS-specific member types is given in Chapter 3 together with the specifications for the PROCESSES clause of the SYSTEM, PROGRAM, and MODULE data definitions when used to obtain processing views of a SUDS database. An example illustrating the use of these data definition statements is given in Chapter 2 following a brief conceptual discussion of SUDS databases.

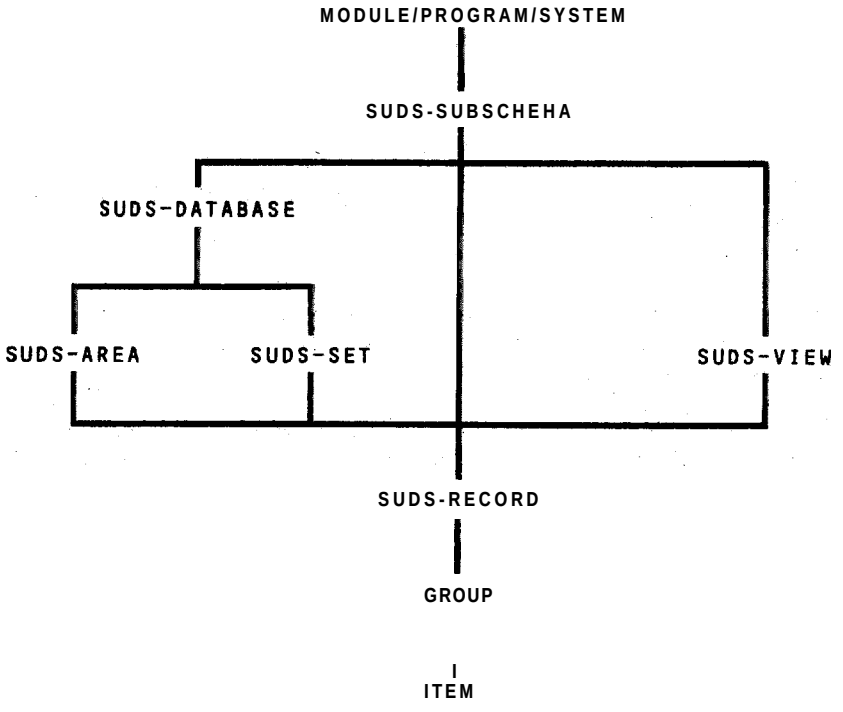


Figure 1.1 Hierarchy Relating SUDS-Specific and Basic DATAMANAGER Member Types

So that the SUDS-specific dictionary members can be processed by DATAMANAGER in the same way as other dictionary members, the member type keywords:

- SUDS-AREAS
- SUDS-DATABASES
- SUDS-RECORDS
- SUDS-SETS
- SUDS-SUBSCHEHAS
- SUDS-VIEWS

have been added to the list of member type keywords available in the following DATAMANAGER commands:

- BULK
- GLOSSARY
- LIST
- PERFORM
- REPORT
- WHICH.

These selection keywords are also available in certain Controller's private commands. An additional set of SUDS-specific keywords are available for use with the GLOSSARY command. They are listed in the table at the end of this chapter.

In addition, the following keywords are available for use with the VIA clause of the DATAMANAGER interrogation commands, WHICH and WHAT:

- ACCESSES
- AREAS
- DATABASE-KEYS
- IN
- KEYS
- LOCATED-IN
- MEMBER
- OF
- OWNER
- PLACED-IN
- PLACEHEMT-SET
- POPULATION-FOR
- PROCEDURE
- PROCESSES-SUDS
- RECORD
- RECORDS
- SEARCH-KEYS
- SETS
- SORT-KEYS
- WITHIN

The SORT-KEYS and KEYS keywords listed above are described in section 3.2 of the DATAMANAGER User's Guide; when used with the DATAMANAGER SUDS Interface facility, they refer to clauses that are specific to SUDS as well as those described in the User's Guide. The other keywords listed above are additional to those described in the User's Guide.

The following points should be noted regarding the use of the keywords in the WHICH and WHAT commands:

- aninterrogation that includes VIA RECORDS refers to the RECORDS clause of a SUDS-SUBSCHEMA data definition statement

- an interrogation that includes VIA RECORD or VIA MEMBER refers to either:

  - the RECORD clause of a SUDS-VIEW data definition statement, or
  - the MEMBER clause of a SUDS-SET data definition statement

- an interrogation that includes VIA AREAS refers to either:

  - the AREAS clause of a SUDS-DATABASE data definition statement, or
  - the AREAS clause of a SUDS-SUBSCHEMA data definition statement

- an interrogation that includes VIA KEYS refers to one of the following:

  - the DIRECT KEY clause of a SUDS-RECORD data definition statement
  - the CALC USING clause of a SUDS-RECORD data definition statement
  - the KEYS clause of a GROUP data definition statement

- an interrogation that includes VIA SEARCH-KEYS refers to either:

  - a SEARCH KEY clause of a SUDS-RECORD data definition statement, or
  - a SEARCH KEY clause of a SUDS-SET data definition statement

- an interrogation that includes VIA SORT-KEYS refers to either:

  - the SORT KEY clause of a SUDS-SET data definition statement, or
  - the SORT-KEY clause of a FILE data definition statement

- an interrogation that includes VIA PLACED-IN refers to either:

  - a PLACED-IN clause of a SUDS-RECORD data definition statement, or
  - a PLACED clause of a SUDS-SET data definition statement.

The ability to generate SUDS Schema and Subschema Data Description Language (DDL) specifications and Storage Structure Language (SSL) specifications from data dictionary members requires the use of the Source Language Generation facility, which is described in a separate manual. The Source Language Generation manual describes the basic version of the facility which can output data descriptions in COBOL, PL/I or Assembler, and/or record layouts. The enhancements to the facility which enable it to generate SUDS-specific output are discussed in Chapter 4 of this manual.

Keywords Allowed for SUDS Member Types in the Command GLOSSARY FOR member-type GIVING clauses

SUDS-SUBSCHEMAS	SUDS-DATABASES	SUDS-SETS	SUDS-RECORDS	SUDS-VIEWS	SUDS-AREAS
ACCESSES AREA-SELECTION AREAS PRIVACY-KEY PRIVACY-LOCK RECORD-SELECTION RECORDS SET-SELECTION SETS	AREAS PRIVACY-LOCK SETS	CALC CHAIN DEFINED-KEY DETACHED DUPLICATES IN INCREASE INDEX INDEX- REORGANIZATION INDEXED IS KEY LIST MEMBER MEMBER-NAME MEMBERSHIP- ESTABLISHMENT MEMBERSHIP- TERMINATION MODE NAME OF ORDER OWNER OWNER-LINKAGE PLACED POINTER-ARRAY POPULATION PROCEDURE REORGANIZATION REPEATED-KEY SEARCH SELECTION SELECTION-METHOD SET-TYPE SORT SORT-KEY-DIRECTION SORT-METHOD SORTED TYPE USING WITHIN	MEA-ID CALC CONTAINS DATABASE-KEYS DIRECT-KEY DUPLICATES- FOR ID IN INDEX INDEX- REORGANIZATION ITEM-COMPRESSION KEY LOCATED-IN LOCATION-MODE NAME OF PLACED-IN PLACEMENT-SET POPULATION PROCEDURE REPEATED-KEY SEARCH SIZE TRANSLATION-TABLE TYPE USING WITHIN	CONTAINS RECORD	MEA-TYPE

CHAPTER 2 SIEMENS UDS DATABASES AND DATAMANAGER

2.1 SUDS CONCEPTS AND DEFINITIONS

The DATAMANAGER Siemens Universal Database System (SUDS) Interface facility enables you to define a SUDS database completely using four of the SUDS-specific DATAMANAGER member types given in Chapter 1. The other two member types given can be used with the SUDS-specific version of the PROCESSES clause (see section 3.8) to define processing views of a SUDS database for particular user applications. The syntax for the additional member types and the PROCESSES clause appears in Chapter 3. Use of these member types to define a SUDS database is illusttated in the example in section 2.2. The remainder of this section contains a brief discussion of some of the SUDS terminology and data structures.

In SUDS terminology, a database is a collection of 'record types'. Each record type in turn is a collection of named items of data (and/or named groups of such items) that together characterize the entities represented by the record type, for example, the employees of a company or its departments. A data item is the smallest namable unit of data in a SUDS database. The value taken on by a data item is frequently referred to as the item content. A record type is the smallest unit of data, also namable, which can be accessed via a unique identifier. A specified value of a unique identifier is sufficient to to determine a single value of each data item of the record type.

A 'record occurrence' (or simply a 'record') is a single instance of a record type, that is, a (compatible) set of values, one each, for the data items comprising the record type. It is the set of item values determined by a given value of a unique identifier of the record type.

Every record type has at least one unique identifier, the 'data base key' for that record type. The data base key of a particular record type is a single data item which need not be contained in the record type. It may be a data item belonging to another record type or it may be in no record type at all. Values of the data base key for a record type are four-byte fields each consisting of the reference number and the sequence of an occurrence of this record type. Data base key values are either defined by the database designer or automatically assigned by SUDS.

Users can define additional keys, each consisting of one or more data items, to provide additional access paths to the records of a record type. These can be defined either as unique or non-unique identifiers by specifying, respectively, whether or not duplicate key values are allowed; that is, whether the same key value is permitted in more than one record. One of these keys can be designated for reasons of efficiency as a 'primary' key. A primary key is usually defined as a unique identifier. The remaining keys are designated as 'secondary' keys, and usually are defined to permit access to a collection of records (in some instances none) with a given key value.

The record types of a SUDS database can be grouped into both physical and logical subdivisions. The physical subdivisions, called 'realms' or 'areas', are generated as files when the database is created. The records of certain record types can be assigned to more than one realm. This facility for separating data physically was incorporated for reasons of:

- data privacy: selected data can be restricted to certain realms with access restricted to specified groups of users
- data security: the effects of hardware and software errors can be restricted to relatively few realms; more frequently updated data can be stored in realms separate from data which is seldom updated
- concurrent access protection: data which would otherwise often be accessed concurrently can be stored in different realms and the realms assigned to different disk drives; this reduces the likelihood of concurrent updating and tends to reduce access time
- optimization of resource allocation and performance: users' applications can help determine the allocation of data to realms so that programs need access only some existing realms. Thus, resources need not be wasted on realms which are rarely used.

The logical subdivisions of a SUDS database are called 'sets'. A set consists of two record types and a relationship which links them, as follows:

- one record type is designated as the 'owner' record type and the other as the 'member' record type

- a relationship is specified from the owner record type to the member record type in the form of a one-to-one or a one-to-many association. If one-to-one, then to each occurrence of the owner record type there corresponds exactly one occurrence of the member record type. If one-to-many, then each owner record can determine zero, one, or more occurrences of the member record type.

A set relationship is not permitted in which the association is many-to-one or many-to-many from the owner record type to the member record type. That is, a given occurrence of a member record type can 'belong' to only a single owner record in a given set. Such associations are handled as follows:

- a many-to-one association from one record type to another is handled as a one-to-many association by appropriate selection of the owner record type

- a many-to-many association between two record types is handled by interposing a third 'auxiliary' record type between the given record types such that each is one-to-many to the auxiliary. Thus each is identified as the owner record type in a separate set in which the auxiliary record type is the member record type. (This device is discussed in more detail in the Siemens BS2000 UDS Design and Definition User's Guide.)

A 'set occurrence' is a single instance of a set, consisting of a single owner record type occurrence and any number of occurrences of the corresponding member record type (possibly none, in which case the set occurrence is called empty).

Access to a particular owner record of a set will result in access to the corresponding member record(s), as determined by the set relationship. For example, in a typical company, a one-to-many association would be defined for the set relating a DEPARTMENT record type as owner with an EMPLOYEE record type as member, or for the set relating EMPLOYEE as owner record type with CHILD as member record type. In the latter case. a specified owner record for an employee without children would define an empty set occurrence. (Note that we could not define a set with EMPLOYEE as owner record type and DEPARTMENT as member record type because the relationship would be many-to-one.)

As a consequence of the set concept, SUDS can be classified as a structured network database system, that is, as a CODASYL type of database system.

As in the case of record types, user-defined keys can be specified to provide additional primary and/or secondary access to the records of a set occurrence.

The rules that govern both the physical and logical characterization of a database are called, collectively, the 'schema'. Thus, a SUDS schema is the description of the record types, realms, and sets comprising a SUDS database.

Once a database schema has been defined, a logical restructuring of a part (or all) of the database can be defined for purposes of a particular user application or class of applications. Such a restructuring is called a 'subschema'. A subschema description includes specification of those database realms, sets, record types, data items, and keys that may be accessed for processing by an application program. This is achieved in the DATAMANAGER SUDS Interface via the SUDS-VIEW and SUDS-SUBSCHEMA member types, where the latter is used to specify a subset of a particular record type for processing in the particular application. Thus, the SUDS-SUBSCHEMA restructures a subset of the database down to the level of record type, and the SUDS-VIEW restructures a **subset** of a record type to the data item level. As indicated in Chapter I, these DATAMANAGER member types are accessed for specific applications from SYSTEM, PROGRAM, and MODULE member types via the PROCESSES clause (see section 3.8).

The correspondence between the SUDS data types and structures discussed above and the DATAMANAGER SUDS-specific member types is as follows:

SUDS Data Type	**DATAMANAGER Member Type**
Database; Schema	SUDS-DATABASE
Realm (or Area)	SUDS-AREA
Set	SUDS-SET
Record Type	SUDS-RECORD
Subschema	SUDS-SUBSCHEMA; SUDS-VIEW
Group of Items	GROUP
Data Item	ITEM

An example of a SUDS database illustrating the above definitions and the corresponding DATAMANAGER member definitions appears in section 2.2.

2.2 EXAMPLE OF A SUDS DATABASE DEFINED USING DATAMANAGER MEMBER TYPES

Consider the SUDS database schema shown in figure 2.1. The database elements include the following (see section 2.3.2, section 3.4, and the Siemens BS2000 UDS Design and Definition User's Guide for discussion of temporary areas, dynamic sets, and the SYSTEM record type):

- the nine areas listed (as realms) in figure 2.1 plus the temporary area SEARCH-REALM
- the seventeen sets indicated (as labeled arrows) in figure 2.1 plus the dynamic set RESULT-SET
- the fifteen record types indicated (as labeled boxes) in figure 2.1 including the symbolic SUDS owner record type SYSTEM.

The schema pictured in' figure 2.1 also appears as a figure in the Siemens manuals, BS2000 UDS Design and Definition User's Guide, Chapter 5, and in BS2000 UDS Entwerfen und Definieren Benutzerhandbuch , Kapitel 5.

To describe this schema using DATAMANAGER member types, you would define the following members:

- one SUDS-DATABASE member specifying all the above areas and sets
- ten SUDS-AREA members
- eighteen SUDS-SET members
- fourteen SUDS-RECORD members.

Each SUDS-RECORD definition lists the data items and/or the groups of data items that comprise the record type and specifies the areas in which records of this record type can be stored. Data items and groups of data items are defined, respectively, in standard DATAMANAGER ITEMs and GROUPs.

To define the processing views which may be taken of the database from application programs requires subschema descriptions as discussed in section 2.1. They are used to restrict the record types required for an application and the data items required in each record type. A DATAMANAGER SUDS-VIEW statement specifies, for a given record type, which of its data items (and/or groups of data items) may be accessed in a processing view (via a subschema) by a program. A DATAMANAGER SUDS-SUBSCHEMA statement specifies, for a particular database schema, which areas (realms), sets, and record types (and/or SUDS-VIEWs of record types) can be accessed in a processing view. A program processing view of a SUDS database is defined in DATAMANAGER via the PROCESSES clause of a SYSTEM, PROGRAM, or MODULE dictionary member. Any number of SUDS-SUBSCHEMA members may be listed in a single PROCESSES clause.

The DATAMANAGER data definition statements given below illustrate the use of SUDS-specific DATAMANAGER members to describe a part of the schema, as follows:

- the area PURCHASE-ORDER-REALM shown in figure 2.1 with its three record types and the sets that link them

- two of the record types (ARTICLE and ARTICLE-DESCR) belonging to the CLOTHING, HOUSEHOLD-GOODS, SPORTS-ARTICLES, FOOD, LEISURE, and STATIONERY areas

- the record types, CUSTOMER and CUSTOMER-ORDER-ITEM, from the area CUSTOMER-ORDER-REALM

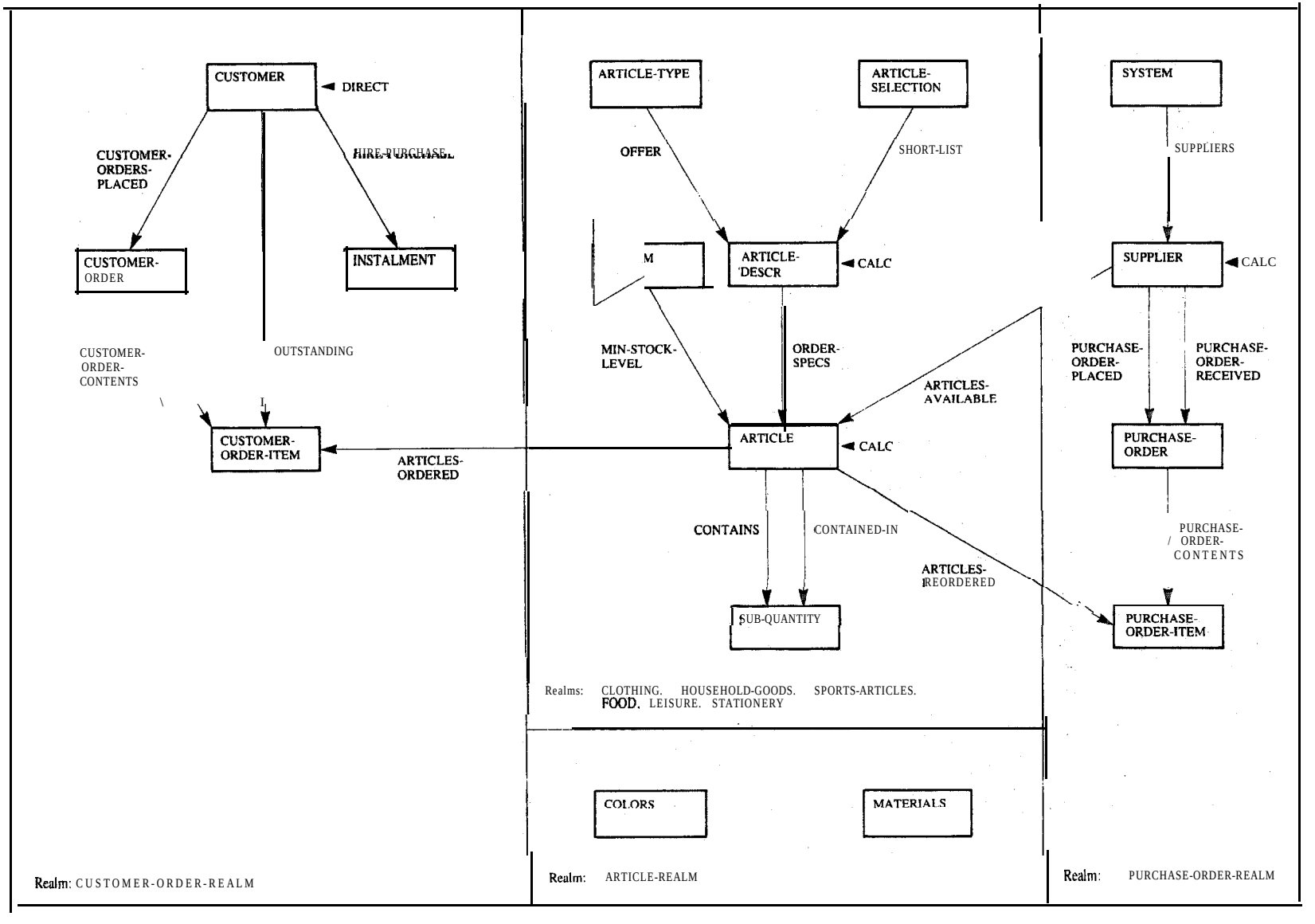


Figure 2.1 Example of a SUDS Schema for a Mail Order Company

- the SYSTEM set SUPPLIERS, the dynamic set RESULT-SET, and two of the sets that link record types from different areas

- a subschema to be used in application programs to access records of a number of the above record types (or SUDS-VIEWS thereof).

The required ITEMs and GROUPS are specified in the CONTAINS clauses of the SUDS-RECORDS and SUDS-VIEWS. Data definitions are included below for some of these ITEMs and GROUPS in order to illustrate the DATAMANAGER syntax used (see sections 5.2 and 5.4) in generating, via the PRODUCE command, SUDS record element definitions in the schema and subschema Data Description Languages. The DATAMANAGER ADD command is used in the example to enter members in the currently open dictionary.

Sample DATAMANAGER SUDS-specific Member Definitions

```
ADD  MAIL-ORDERS;
SUDS-DATABASE
AREAS  CUSTOMER-ORDER-REALM,PURCHASE-ORDER-REALM,  CLOTHING,
       HOUSEHOLD-GOODS,  SPORTS-ARTICLES,  FOOD.  LEISURE,
       STATIONERY, ARTICLE-REALM, SEARCH--REALM
SETS PURCHASE-ORDER-SPECS. MIN-STOCK-LEVEL. ARTICLES-ORDERED.
       ARTICLES-REORDERED,ARTICLES-AVAILABLE,  SUPPLIERS,
       PURCHASE-ORDER-PLACED,   PURCHASE-ORDER-RECEIVED,
       PURCHASE-ORDER-CONTENTS,   CUSTOMER-ORDERS-PLACED,
       CUSTOMER-ORDER-CONTENTS,  OUTSTANDING,  HIRE-PURCHASE,
       OFFER,  SHORT-LIST,  CONTAINS,  CONTAINED-IN,  RESULT-SET
PRIVACY-LOCK 'SHIP-KEY'
DESCRIPTION 'A SAMPLE SUDS SCHEMA'
NOTE 'EXAMPLE EXTRACTED FROM SIEMENS BS2000'
     'UDS DESIGN AND DEFINITION USER"S GUIDE'
;

ADD  PURCHASE-ORDER-REALM;
SUDS-AREA
;

ADD  SEARCH-REALM;
SUDS-AREA
TEMPORARY
;

ADD  SUPPLIER;
SUDS-RECORD
WITHIN  PURCHASE-ORDER-REALM
LOCATION-MODE  CALC
       USING  SUPPL-NO,  SUPPL-NAME
       DUPLICATES  DISALLOWED
CONTAINS  SUPPL-NO,  SUPPL-NAME,  SUPPL-PCODE,  SUPPL-TOWN,
       SUPPL-STREET,  SUPPL-STREET-NO,  SUPPL-TEL,  SUPPL-POBOX,
       SUPPL-TELEX
POPULATION  200  FOR  PURCHASE-ORDER-REALM
;

ADD  PURCHASE-ORDER;
SUDS-RECORD
WITHIN  PURCHASE-ORDER-REALM
CONTAINS  PURCH-ORD-NO,  PURCH-ORD-YEAR,  PURCH-ORD-MONTH,
       PURCH-ORD-DAY
TRANSLATION-TABLE SIZE 200
;

ADD  PURCHASE-ORDER-ITEM;
SUDS-RECORD
WITHIN  PURCHASE-ORDER-REALM
CONTAINS  PURCH-ORD-ITEM-NO,  PURCH-ORD-QUANTITY
TRANSLATION-TABLE SIZE 500
;
```

```
ADD  ARTICLE;
SUDS-RECORD
WITHIN CLOTHING  ANDHOUSEHOLD-GOODS,  SPORTS-ARTICLES,  FOOD,
       LEISURE,STATIONERY
       AREA-ID   REALM-SELECTION-4
LOCATION-MODE  CALC
       USING  ART-NO,  COL-NO.  SIZE
       DUPLICATES  DISALLOWED
SEARCH  KEY  ART-NAME
       USING  CALC  NAME  SEARCH-TAB-ARTICLE-2
       DUPLICATES  ALLOWED
SEARCH  KEY  ART-NO-AVAIL,  COL-NO-AVAIL,  SIZE
       USING  CALC  NAME  SEARCH-TAB-ARTICLE-l
       DUPLICATES  DISALLOWED
CONTAINS  ART-NO,  COL-NO,  ART-NAME,  ART-NO-AVAIL,  COL-NO-AVAIL,
       SIZE, PRICE,  INSTAL-PRICE,  MAX-STOCK,MIN-STOCK,
       CURR-STOCK,  STATISTICS,  NOT-AVAIL-CODE
TRANSLATION-TABLE  SIZE  600
       LOCATED-IN  ARTICLE-REALM
POPULATION  400  FOR  CLOTHING,100  FOR  HOUSEHOLD-GOODS,
       400  FOR  SPORTS-ARTICLES.150  FOR  FOOD,
       400  FOR  LEISURE,  250  FOR  STATIONERY
INDEX  SEARCH-TAB-ARTICLE-l
       PLACED-IN  ARTICLE-REALM
INDEX  SEARCH-TAB-ARTICLE-2
       PLACED-IN  ARTICLE-REALM
NOTE  'THIS  RECORD  LINKS  PURCHASE-ORDER-REALM  AND'
      'CUSTOMER-ORDER-REALM'  TO  OTHER  AREAS  OF  SCHEMA'
;

ADD  ARTICLE-DESCR;
SUDS-RECORD
WITHIN  CLOTHINGANDHOUSEHOLD-GOODS,  SPORTS-ARTICLES,  FOOD,
       LEISURE,  STATIONERY
       AREA-ID  REALM-SELECTION-3
LOCATION-MODE  CALC
       USING  ART-NAME
       DUPLICATES  ALLOWED
CONTAINS  ART-NO,  ART-NAME,  (4)  MATERIAL,  LENGTH-FIELD,
       ART-INFO
TRANSLATION-TABLE  SIZE  300
       LOCATED-IN  ARTICLE-REALM
POPULATION  200  FOR  CLOTHING,100  FOR  HOUSEHOLD-GOODS,
       200  FOR  SPORTS-ARTICLES,  70  FOR  FOOD,
       200  FOR  LEISURE,  100  FOR  STATIONERY


ADD  CUSTOMER-ORDER-ITEM;
SUDS-RECORD
WITHIN  CUSTOMER-ORDER-REALM
CONTAINS  CUST-ORD-ITEM-NO,  CUST-ORD-QUANTITY.
       PAY-INSTAL-CODE,  CUST-ORD-ITEM-STATUS
TRANSLATION-TABLE  SIZE  1000
;

ADD  CUSTOMER;
SUDS-RECORD
WITHIN  CUSTOMER-ORDER-REALM
LOCATION-MODE  DIRECT
       KEY  CUST-NO  IN  CUSTOMER
CONTAINS  CUST-NO,  CUST-NAME,  CUST-FIRST-NAME
DATABASE-KEYS'  CUST-NO
TRANSLATION-TABLE  SIZE  100
;

ADD  SUPPLIERS;
SUDS-SET
ORDER  SORTED  INDEXED  DEFINED-KEY
       DUPLICATES  DISALLOWED
MEMBER  SUPPLIER  MANDATORY  AUTOMATIC
       SORT  KEY  SUPPL-NAME,  SUPPL-NO  ASCENDING
```

```
ADD   ARTICLES-AVAILABLE;
SUDS-SET
ORDER  SORTED  INDEXED  DEFINED-KEY
       DUPLICATES ALLOWED
OWNER  SUPPLIER
MEMBER ARTICLE MANDATORY AUTOMATIC
       SORT  KEY  ART-NAME  ASCENDING
       SEARCH  KEY  NOT-AVAIL-CODE  USING  INDEX
       NAME  SEARCH-TAB-ART-AVAIL  DUPLICATES  ALLOWED
       SELECTION  CURRENT
MODE  POINTER-ARRAY  ATTACHED
POPULATION  500
REORGANIZATION  5
INDEX   SEARCH-TAB-ART-AVAIL
       PLACED DETACHED
       TYPE DB-KEY-LIST
LINKED-OWNER
;

ADD  ARTICLES-REORDERED;
SUDS-SET
ORDER LAST
OWNER ARTICLE
MEMBER  PURCHASE-ORDER-ITEM  MANDATORY  AUTOMATIC
       SELECTION  OWNER
LINKED-OWNER
;

ADD  PURCHASE-ORDER-PLACED;
SUDS-SET
ORDER LAST
OWNER  SUPPLIER
MEMBER PURCHASE-ORDER MANDATORY AUTOMATIC
       SELECTION  CURRENT
;

ADD   PURCHASE-ORDER-RECEIVED;
SUDS-SET
ORDER  FIRST
OWNER  SUPPLIER
MEMBER PURCHASE-ORDER MANDATORY AUTOMATIC
       SELECTION  CURRENT
;

ADD  PURCHASE-ORDER-CONTENTS;
SUDS-SET
ORDER NEXT
OWNER PURCHASE-ORDER
MEMBER  PURCHASE-ORDER-ITEM  MANDATORY  AUTOMATIC
       SELECTION  CURRENT
MODE  LIST  DETACHED  PHYSICAL
POPULATION  20
;

ADD  RESULT-SET;
SUDS-SET
DYNAMIC
ORDER  IMMATERIAL
;

ADD  ART-NO;
ITEM
HELD-AS PICTURE '9(6)'


ADD  ART-NAME;
ITEM
HELD-AS  CHARACTER  40
;

ADD  PERCENT;
ITEM
HELD-AS PICTURE '99'
```

```
ADD MAT-CODE;
ITEM
HELD-AS PICTURE 'X'


ADD PRICE;
ITEM
HELD-AS  DECIMAL-PACKED  5.2
;

ADD MAX-STOCK;
ITEM
HELD-AS  DECIMAL-PACKED  15


ADD CUST-NO;
ITEM
HELD-AS BINARY 9


ADD MATERIAL;
GROUP
CONTAINS  PERCENT,MAT-CODE
;

ADD SUPPL-ADDRESS;
GROUP
CONTAINS  SUPPL-PCODE,  SUPPL-TOWN,  SUPPL-STREET,
SUPPL-STREET-NO
;

ADD PURCH-ORD-DATE;
GROUP
CONTAINS  PURCH-ORD-YEAR,  PURCH-ORD-MONTH,  PURCH-ORD-DAY
;

ADD STOCK;
GROUP
CDNTAINS  MAX-STOCK,MIN-STOCK,  CURR-STOCK
;

ADD  SUPPLIER-VU;
SUDS-VIEW
RECORD  SUPPLIER
CONTAINS  SUPPL-NO,  SUPPL-NAME,  SUPPL-ADDRESS,  SUPPL-TEL,
       SUPPL-POBOX,  SUPPL-TELEX
;

ADD PURCHASE-ORDER-VU;
SUDS-VIEW
RECORD PURCHASE-ORDER
CONTAINS  PURCH-ORD-NO,  PURCH-ORD-DATE


ADD  ARTICLE-VU;
SUDS-VIEW
RECORD ARTICLE
CONTAINS  ART-NAME,  ART-NO-AVAIL,  COL-NO-AVAIL,  SIZE,  STOCK,
       NOT-AVAIL-CODE


ADD ORDERS;
SUDS-SUBSCHEMA
ACCESSES  MAIL-ORDERS
PRIVACY-KEY 'SHIP-KEY'
AREAS  PURCHASE-ORDER-REALM,  CLOTHING,
       HOUSEHOLD-GOODS,  SPORTS-ARTICLES,  FOOD,  LEISURE,
       STATIONERY,  ARTICLE-REALM,  SEARCH-REALM
RECORDS  PURCHASE-ORDER-ITEM,  SUPPLIER-VU,  PURCHASE-ORDER-VU,
       ARTICLE-VU
SETS  ARTICLES-AVAILABLE,  ARTICLES-REORDERED,  SUPPLIERS,
       PURCHASE-ORDER-PLACED,  PURCHASE-ORDER-RECEIVED,
       PURCHASE-ORDER-CONTENTS,  RESULT-SET
;
```

```
ADD HUD-ORDERS;,
MODULE:
PROCESSES SUDS SUBSCHEMAS ORDERS
;
```

2.3 FURTHER CONSIDERATIONS

2.3.1 Introduction

In the following sections, certain aspects of DATAMANAGER SUDS-specific data definitions-are. discussed. They are included as an extension to the remarks given in the Chapter 3 data. definition statement specifications. The topics, covered are:

- dynamic sets and. temporary areas
- privacy locks and keys.

Further discussion of these topics can be found in the Siemens BS2000 UDS Design and Definition User's Guide;

2.3.2 Dynamic Sets and Temporary Areas-

Within SUDS there is the facility to define dynamic sets and temporary realms (areas);, DATAMANAGER provides keywords (DYNAMIC and- TEMPORARY) which support. this facility in the SUDS-SET and SUDS-AREA data definition syntax-.

A dynamic set is a logical' grouping of records (all. of the same record type) generated by a program during its execution; the dynamic set itself is defined as a series of pointers to these records. Once generated; a dynamic set is stored in a temporary area. Each concurrent user is given a separate temporary area, which is cleared- when the program-has completed its run. In this way, several users of a database can access different records simultaneously, using the same subschema. The member records of a dynamic set can be of any type, but all records grouped together at any given time within a dynamic- set must- be of the same type.

A- dynamic set is a SYSTEM set (defined by the SUDS clause, OWNER IS SYSTEM; see remark 2 of section 3.4 and the Siemens. BS2000 UDS Design and Definition User's Guide) consisting of only a single set occurrence.

A temporary area must. be defined if the schema being described contains any dynamic sets (or if SUDS will have to generate a dynamic set automatically when certain Data Manipulation Language statements. are entered; see the Siemens User's Guide). No more, than one temporary area may be defined in a given schema description,

2.3.3 Privacy Locks and Privacy Keys

The SUDS privacy lock and privacy key facility provides two levels of security. One is designed to prevent unauthorized access to a database via a subschema. the other to prevent unauthorized compilation of a program attempting to use a subschema in such a way. This is achieved in SUDS by specifying up to three passwords (privacy locks) in the schema (database) definition and in the subschema definition. For a program to be able to access the schema (necessarily via a subschema), at least one of the passwords specified in the schema must be entered (as a privacy key) in the subschema. To ensure compilation of any (COBOL) program calling a subschema which has one or more passwords specified as privacy locks. at least one of these passwords must be coded as a privacy key in the program's Identification Division.

These facilities are defined to DATAMANAGER by means of:

- the PRIVACY-LOCK clause of the SUDS-DATABASE data definition, and
- the PRIVACY-LOCK and PRIVACY-KEY clauses of the SUDS-SUBSCHEMA data definition.

CHAPTER 3 DATAMANAGER DATA DEFINITIONS FOR SIEMENS UDS DATABASES

3.1 INTRODUCTION

To enable a Siemens Universal Database Systems (SUDS) environment to be fully reflected in a dictionary maintained by DATAMANAGER, the DATAMANAGER SUDS Interface provides:

- the following six additional SUDS-specific member types:
  - SUDS-DATABASE
  - SUDS-AREA
  - SUDS-SET
  - SUDS-RECORD
  - SUDS-VIEW
  - SUDS-SUBSCHEMA
- an extension to the basic DATAMANAGER MODULE, PROGRAM and SYSTEM data definition statements, to reflect the processing view of the database. See section 3.8.

As in the case of the basic DATAMANAGER member types, members of each of the above member types are held in a dictionary as datadefinitions. They can be entered into a dictionary using data definition statements (see section 4.1 of the DATAMANAGER User's Guide for a general discussion of DATAMANAGER data definitions and data definition statements), each preceded by a DATAMANAGER INSERT, ADD, or REPLACE command. On-line users with the CONTROLMANAGER Extended Interactive Facility (selectable unit CMR-FE01) installed can make use of the MSP full screen updating capabilities to add, replace, or modify dictionary data definitions. Data definition statement specifications for the above SUDS-specific member types are given in sections 3.2 to 3.7. General rules that apply to the coding of all data definition statements appear in Chapter 2 of the DATAMANAGER User's Guide.

3.2 THE SUDS-DATABASE DATA DEFINITION STATEMENT

The SUDS-DATABASE data definition statement allows you to define a SUDS schema to DATAMANAGER:

Format

```
SUDS-OATABASE
AREAS  area-name  C,area-name]...
[SETS  set-name  [,set-name]...]
[PRIVACY-LOCK  'password'  [,'password'  [,'password']]]
[common  clauses1
{;}
{.}
```

where

area-name is the name of a member that is a SUDS-AREA

set-name is the name of a member that is a SUDS-SET

password is a character string of up to ten printable characters

common clauses are any of the following clauses as defined in section 4.3 of the DATAMANAGER User's Guide:

```
-ACCESS-AUTHORITY          - FREQUENCY
- ADMINISTRATIVE DATA      -NOTE
- ALIAS                    -OBSOLETE-DATE
- CATALOGUE                - QUERY
- COMMENT                  -SECURITY-CLASSIFICATION
-DESCRIPTION               - SEE
-EFFECTIVE-DATE
```

Remarks

1 The SUDS-DATABASE data definition enables you to define a Siemens UDS schema to DATAMANAGER. In so doing, you must comply with the rules for schema definition given in the Siemens BS2000 UDS Design and Definition User's Guide.

2 The AREAS clause enables you to specify all of the areas (realms) within the database. The maximum number of areas permitted is 240.

3 The SETS clause enables you to specify all of the sets within the database and thus indirectly (via the SUDS-SET members referenced) all record types that are owners and members of these sets. A maximum of 254 sets are allowed within a database.

4 The PRIVACY-LOCK clause enables you to specify a maximum of three passwords for the database. Any SUDS-SUBSCHEMA member that accesses the database must be defined with a PRIVACY-KEY clause containing a password that matches one of the passwords specified in the SUDS-DATABASE PRIVACY-LOCK clause.

5 Common clauses, listed under Format above, can be present in any type of data definition statement; they are therefore documented separately, in section 4.3 of the DATAMANAGER User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any other clause identifier or other keyword available in the data definition syntax for this member type.

6 Clauses can be declared in any order, provided that subordinate clauses are not separated from the other elements of the clause of which they form a part.

7 A record containing the SUDS-DATABASE data definition statement can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 of the DATAMANAGER User's Guide or Chapter 5 of the CONTROLMANAGER User's Guide) and an encoded record can subsequently be generated and inserted into the data entries data set; If, when the encoded record is generated, any member whose name appears in the data definition statement has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as:

- a dummy SUDS-AREA if the member's name appears in the AREAS clause
- a dummy SUDS-SET if the member's name appears in the SETS clause
- a dummy ITEM if the member's name appears in a SEE clause.

Example

An example is given in section 2.2.

3.3 THE SUDS-AREA DATA DEFINITION STATEMENT

The SUDS-AREA data definition statement enables you to define a subset of the records contained in a database as an area.

Format

```
SUDS-AREA
[TEMPORARY]
[common clauses1
{;}
.}
```

where common clauses are any of the following clauses, as defined in section 4.3 of the DATAMANAGER User's Guide:

```
-ACCESS-AUTHORITY          - FREQUENCY
- ADMININSTRATIVE-DATA     - NOTE
- ALIAS                    -OBSOLETE-DATE
- CATALOGUE                - QUERY
- COMMENT                  -SECURITY-CLASSIFICATION
- DESCRIPTION              - SEE
-EFFECTIVE-DATE
```

Remarks

1 The SUDS-AREA data definition statement enables you to define a Siemens UDS area (realm) to DATAMANAGER. In so doing, you must comply with the rules given in the Siemens BS2000 UDS Design and Definition User's Guide for defining a realm.

2 If the keyword TEMPORARY is specified, a temporary area is defined that is used to store dynamic sets (see section 2.3.1).

3 Common clauses, listed under Format above, can be present in any type of data definition statement; they are therefore defined separately, in section 4.3 of the DATAMANAGER User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any other clause identifier or other keyword available in the data definition syntax for this member type.

4 Clauses can be declared in any order, provided that subordinate clauses are not separated from the other elements of the clause of which they form a part.

5 A record containing the SUDS-AREA data definition statement can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 of the DATAMANAGER User's Guide or Chapter 5 of the CONTROLMANAGER User's Guide) and an encoded record can subsequently be generated and inserted into the data entries data set, If, when the encoded record is generated, any member whose name appears in a SEE clause has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as a dummy ITEM.

Example

An example is given in section 2.2.

3.4 THE SUDS-SET DATA DEFINITION STATEMENT

The SUDS-SET data definition statement enables you to define a SUDS set to DATAMANAGER, consisting of an owner record type, a member record type, and the access paths linking their occurrences.

Format

See Figure 3.1, where

table-name is the name of a sort key or search key table or a hash area, and may be up to 30 printable characters in length

record-name is the name of a member that is a SUDS-RECORD

item-name is the name of a member that is an ITEM

module-name is the name of a member that is a MODULE

id-name-l is the name of a SUDS identifier or an ITEM member that is specified in the LOCATION-MODE clause of the SUDS-RECORD member named in the SUDS-SET OWNER clause

id-name-2 is the name of a local SUDS alias

area-name is the name of a member that is a SUDS-AREA

j is a non-negative integer, being the average number of member records expected to be in a set occurrence when initially loading the database. The default value of j is 0.

k is a positive integer, being the average additional number of records expected when extending the set occurrences subsequent to initial database loading. The default value of k is 1.

m is an integer from 1 to 20 (with a default value of 2), being the number of table blocks to be searched (by SUDS) in determining the extent to which the table describing the connection of records in a set occurrence (that is, pointer array, list, or sort key table) can be reorganized subsequent to initial database loading

n is an integer from 1 to 20 (with a default value of 2), being the extent to which a search key or sort key table defined for this set can be reorganized

common clauses are any of the following clauses as defined in section 4.3 of the DATAMANAGER User's Guide:

```
-ACCESS-AUTHORITY          - FREQUENCY
-ADMINISTRATIVE-DATA       - NOTE
- ALIAS                    - OBSOLETE-DATE
- CATALOGUE                - QUERY
- COMMENT                  -SECURITY-CLASSIFICATION
- DESCRIPTION              - SEE
-EFFECTIVE-DATE
```

```
SUDS-SET

[DYNAMIC]

ORDER  { IRST
         LAST
         NEXT
         PRIOR
         IMMATERIAL
         SORTED  [INDEXED [NAME table-name]] { DATABASE-KEY
                                               DEFINED-KEY DUPLICATES { ALLOWED
                                                                        DISALLOWED } } }

[OWNER  record-name]

[MEMBER  record-name  [ { MANDATORY } 1 [ { AUTOMATIC } 1
                        { OPTIONAL }     { MANUAL }

                        [SORT  KEY  item-name  C, item-name]...  { ASCENDING }]
                                                                 { DESCENDING }

                        [SEARCH  KEY  item-name  [, item-name]...

                                      USING { CALC [PROCEDURE  module-name1] }
                                            { INDEX                          }

                                      [NAME table-name] DUPLICATES { ALLOWED    } I...
                                                                   { DISALLOWED }

                        [SELECTION { CURRENT                                    } 11
                                   { OWNER [NAME  id-name-1  IS id-name-2]...   }

[MODE  { CHAIN [LINKED-PRIOR]                                                  }  1
       { { POINTER-ARRAY } { ATTACHED                                          }
       { { LIST          } { DETACHED  [WITHIN  area-name]  [PHYSICAL]         } }

[POPULATION  j  [INCREASE k]]

[REORGANIZATION  ml

[INDEX  table-name  [PLACED { ATTACHED                    } 1
                            { DETACHED [IN  area-name]    }

                    [TYPE { DB-KEY-LIST                                        } I1...
                          { REPEATED-KEY   [INDEX-REORGANIZATION   nl          }

CLINKED-OWNER1

[common  clauses1

{ ; }
{ . }
```

Figure 3.1 Format of the SUDS-SET Data Definition Statement

Remarks

1 The SUDS-SET data definition statement enables you to define a Siemens UDS set to DATAMANAGBR. In so doing, you must comply with the rules given in the Siemens BS2000 UDS Design and Definition User's Guide for defining a set.

2 The DYNAMIC keyword identifies the set as a dynamic set (see section 2.3.2). Any SUDS schema containing this set must also contain exactly one temporary area (see section 2.3.2). If DYNAMIC is specified, the DATAMANAGER software will accept other entries in the statement (if syntactically correct). However, when the PRODUCE command of the Source Language Generation facility is used to generate SUDS schema specifications, such entries are ignored and the following (SUDS) clauses are assumed:

```
SET IS DYNAMIC
ORDER IS IMMATERIAL
OWNER IS SYSTEM
MODE IS POINTER-ARRAY DETACHED WITHIN realm-name
```

3 The ORDER clause defines the point at which records are to be inserted or retrieved within a set occurrence. Subordinate keywords have the same meaning as in the Siemens BS2000 UDS Design and Detinition User's Guide. If SORTED is specified, a sort key table name may optionally be included defining an additional direct access path via primary key.

4 The OWNER clause specifies a SUDS-RECORD member as the owner record type for the SUDS-SET being defined. If the OWNER clause is omitted, then. when the PRODUCE command of the Source Language Generation facility is used to generate SUDS schema specifications, this set is assumed to be a SYSTEM set (see the Siemens BS2000 UDS Design and Definition User's Guide).

5 The MEMBER clause specifies a SUDS-RECORD member as the member record type for the SUDS-SET being defined. A MEMBER clause must be present unless the SUDS-SET, is a dynamic set.

6 The keywords, MANDATORY, OPTIONAL, AUTOMATIC. and MANUAL, available in the MEMBERS clause are as defined in the Siemens BS2000 UDS Design and Definition User's Guide.

7 If neither MANDATORY nor OPTIONAL is stated in a MEMBER clause, then, when the PRODUCE command of the Source Language Generation facility is used to generate SUDS schema specifications, OPTIONAL membership termination is assumed.

8 If neither AUTOMATIC nor MANUAL is stated in a MEMBER clause, then, when the PRODUCE command of the Source Language Generation facility is used to generate SUDS schema specifications, AUTOMATIC membership establishment is assumed.

9 The SORT KEY sub-clause defines a data item or a collection of data items of the member record type as a primary key. The member records within an occurrence of the set are sorted in either ASCENDING or DESCENDING order of the key values.

10 The SEARCH KEY sub-clause provides additional direct access to the member records of a set occurrence by defining a collection of one or more data items in the member record type as a secondary key. Key values are used to access member records within a set occurrence by means of a search key table (USING INDEX) or a hash routine (USING CALC). The PROCEDURE subclause is used to indicate that the hash routine is provided by the user (via module-name). Omission of the PROCEDURE sub-clause indicates that the standard SUDS hash routine is to be used. The optional NAME clause assigns a name to the search key table or the SUDS storage area (the hash area) to be accessed by the hash routine. For a more detailed discussion of hash routines and hash areas, see the Siemens BS2000 UDS Design and Definition User's Guide. More than one secondary key can be defined for a set by specification of additional SEARCH KEY clauses.

11 The SELECTION sub-clause defines the method of selecting the desired set occurrences. If CURRENT is specified, the record last referenced within a set identifies the set occurrence. If OWNER is specified, owner records of specific set occurrences are identified by data base key values or primary key values. This option requires that a unique identifier be defined in the SUDS-RECORD data definition for the owner record type, as either a data base key by means of the LOCATION-MODE DIRECT clause or a primary key via the LOCATION-MODE CALC clause with DUPLICATES DISALLOWED. If the OWNER option has been selected, the NAME sub-clause permits creation of an additional identifier to hold key values. If a key consisting of several items has been defined (for the owner record type) by LOCATION-MODE CALC, the NAME sub-clause is repeated an appropriate number of times to create one substitute item for every item in the key.

12 The MODE clause controls the placement of the records of a set occurrence and their connection, from owner record to member record(s). Subordinate keywords and the options they define are described in the Siemens BS2000 UDS Design and Definition User's Guide (defaults are also given).

13 The POPULATION clause specifies the average number of member records expected per set occurrence for initial loading and optionally the average increase expected when extending the size of a set occurrence. This specification is used by SUDS to determine the storage space required for occurrences of this set. The POPULATION clause should be present with j greater than zero if a PLACEMENT-SET clause (naming the current set) appears in the SUDS-RECORD data definition for the member record type of this set.

14 The REORGANIZATION clause is used to determine the extent of the permitted reorganization within a pointer array (MODE POINTER-ARRAY), a list (MODE LIST), or a sort key table (MODE CHAIN and ORDER SORTED with INDEXED omitted).

15 An INDEX clause can be used to specify the location and type of a search key table named in the MEMBER clause (USING INDEX) or a sort key table named in the ORDER clause (SORTED INDEXED). An option (the INDEX-REORGANIZATION clause) is included which can be used to determine the extent to which the table may be reorganized. Alternatively, an INDEX clause can be used to specify the location (but not the type) of a hash area named in a SEARCH KEY clause (USING CALC). More than one INDEX clause can be specified, as required.

16 The LINKED-OWNER keyword is used to include in each member record of the set an additional pointer, which points to its owner record.

17 Common clauses, listed under Format above, can be present in any type of data definition statement; they are therefore documented separately, in section 4.3 of the DATAMANAGER User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any other clause identifier or other keyword available in the data definition syntax for this member type

18 Clauses can be declared in any order, provided that subordinate clauses are not separated from the other elements of the clause of which they form a part.

19 A record containing the SUDS-SET data definition statement can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 of the DATAMANAGER User's Guide or Chapter 5 of the CONTROLMANAGER User's Guide) and an encoded record can subsequently be generated and inserted into the data entries data set. If, when the encoded record is generated, any member whose name appears in the data definition statement has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as:

- a dummy SUDS-RECORD if the member's name appears as record-name in the OWNER clause or the MEMBER clause

- a dummy ITEM if the member's name appears as item-name in the SORT KEY sub-clause or a SEARCH KEY sub-clause of the MEMBER clause, or if it appears in a SEE clause

a dummy MODULE if the member's name appears as module-name in the USING sub-clause subordinate to a SEARCH KEY sub-clause

- a dummy SUDS-AREA if the member's name appears as area-name in the MODE clause or an INDEX clause.

Example

An example is given in section 2.2.

3.5 THE SUDS-RECORD DATA DEFINITION STATEMENT

The SUDS-RECORD data definition statement enables you to define a SUDS record type to DATAMANAGER.

Format

See figure 3.2, where

area-name is the name of a member that is a SUDS-AREA

identifier-l is the name of a SUDS data item (automatically generated by SUDS) used to hold the name of any of the specified areas when the area is used to store an occurrence of this record type

identifier-2 is the name of a SUDS data item (automatically generated by SUDS) used to hold data base key values (see section 2.1). The data item is not contained in any record type.

item-name is the name of a member that is an ITEM

record-name is the name of a member that is a SUDS-RECORD

module-name is the name of a member that is a MODULE

table-name is the name of a search key table or hash area and may be up to 30 printable characters in length

content declares an item, a subordinate group or an array, in the format shown in figure 3.3 on page 3-12

where

item-name is the name of a member that is an ITEM

version is an unsigned integer in the range 1 to 15, being a number that indicates which version of the specified item is relevant to this definition. The version is within the HELD-AS form or within a defaulted form, as indicated in remark 12 below. If version is omitted, a default value of I is assumed.

group-name is the name of a member that is a GROUP

```
SUDS-RECORD

WITHIN area-name [AND area-name [,area-name]... AREA-ID identifier-11

[LOCATION-MODE { DIRECT { ID identifier-2                                  }
                         { KEY item-name { IN } record-name }
                                         { OF }
               { CALC [PROCEDURE module-name] USING item-name [,item-name]...
                                                DUPLICATES { ALLOWED    }
                                                           { DISALLOWED }

[SEARCH KEY item-name [,item-name]...

          USING { CALC [PROCEDURE module-name] }
                { INDEX                        }

          [NAME table-name] DUPLICATES { ALLOWED    } I...
                                       { DISALLOWED }

CONTAINS content [,content]...

[DATABASE-KEYS item-name [,item-name]...]

[TRANSLATION-TABLE [SIZE jl [LOCATED-IN area-name11

[POPULATION k FOR area-name [,k FOR area-name]...]

[PLACEMENT-SET set-name]

[INDEX table-name [PLACED-IN area-name]

                [TYPE { DB-KEY-LIST                          } Il...
                      { REPEATED-KEY [INDEX-REORGANIZATION l }

[COMPRESSION]

[common clauses1

{;}
{.}
```

Figure 3.2 Format of the SUDS-RECORD Data Definition Statement

```
[(n)] { item-name  [version] } [ALIGNED] [KNOWN-AS Local-name1
      { group-name              }
```

Note

group-name must be preceded by (n) when declared in a SUDS-RECORD CONTAINS clause or if declared in that of a GROUP itself **CONTAINed,** directly or indirectly, in a SUDS-RECORD.

Figure 3.3 Format of a Content Declaration in a SUDS-RECORD or SUDS-VIEW Data Definition or in the Data Definition of a GROUP Directly or Indirectly **CONTAINed** in a SUDS-RECORD or SUDS-VIEW

local-name is a name, conforming to the rules for member names stated in section 2.4 of the DATAMANAGER User's Guide, that can be used instead of the name or alias of the contained member. when record layouts or source language data descriptions are generated from this data definition by the DATAMANAGER Source Language Generation facility. local-name is not separately recorded in the data dictionary (that is, no dummy data entries record and no index record is created for local-name when the data definition in which it appears is encoded) so local-name cannot be interrogated and can be the same as another name, an alias or a catalog classification in the data dictionary. local-name is the name by which the contained member is known only within. the SUDS record type defined by this data definition.

n is a positive integer greater than or equal to two, being the number of times item-name or group-name occurs in the array

j is a positive integer, being the expected number of records of this record type. SUDS uses this number to calculate the size in blocks (with a default value of one block) to be reserved for the Data Base Key Translation Table and for any hash areas used to access records via secondary keys.

k is a positive integer, being the number of records to be stored in the indicated area. SUDS uses this number to calculate the size in blocks (with a default value of one block) to be reserved for a hash area used to access records via primary key.

set-name is the name of a member that is a SUDS-SET

m is an integer from 1 to 20 (with a default value of 2). being the extent to which table-name can be reorganized when table-name is a search key table for this record type

common clauses are any of the following clauses as defined in section 4.3 of the DATAMANAGER User's Guide:

-ACCESS-AUTHORITY	- FREQUENCY
- ADMINISTRATIVE-DATA	- NOTE
- ALIAS	-OBSOLETE-DATE
- CATALOGUE	- QUERY
- COMMENT	-SECURITY-CLASSIFICATION
- DESCRIPTION	- SEE
-EFFECTIVE-DATE	

Remarks

1 The SUDS-RECORD data definition statement permits you to define a Siemens UDS record type to DATAMANAGER. In so doing, you must comply with the rules given in the Siemens UDS Design and Definition User's Guide for defining a record type.

2 The WITHIN clause is mandatory; it designates one or more SUDS-AREA s (realms) for use in storing records of the currently defined record type. No area named in the WITHIN clause may be a temporary area (see section 2.3.2). If more than one area is named in the WITHIN clause, then the AREA-ID clause must also be present. Conversely, it must not appear if only one area is named.

3 The LOCATION-MODE clause determines the mode of access to the records of this record type, either via data base key values or via primary key values (see section 2.1). The former mode of access is obtained by specifying a SUDS data item (via LOCATION-MODE DIRECT) which is to hold data base key values. If the item is named in the DIRECT KEY item-name sub-clause, it is contained in the specified record type (not necessarily the record type currently being defined). If named in the DIRECT ID identifier-2 sub-clause, the named item is not contained in any record type. If the keyword DIRECT is not specified, then data base key values are supplied automatically by SUDS. Access to records via primary key is obtained (LOCATION-MODE CALC) by specifying a collection of one or more data items (all from the currently defined record type) as primary key. Values of the primary key are converted to the locations of records by a user-provided 'hash routine' (PROCEDURE module-name) or, by default, a SUDS-provided hash routine. For a more detailed discussion of SUDS hash routines and hash areas, see the Siemens BS2000 UDS Design and Definition User's Guide.

4 The SEARCH KEY clause specifies a collection of one or more data items (all from the currently defined record type) as a secondary key to the record type (see section 2.1). Secondary key values provide additional direct access to records of this record type by means of either a search key table (USING INDEX) or a hash routine (USING CALC). The PROCEDURE sub-clause is used to indicate that the hash routine is provided by the user (via module-name). Omission of the PROCEDURE sub-clause indicates that the standard SUDS hash routine is to be used. Optionally a NAME sub-clause may be specified providing a name for the search key table or hash area. More than one secondary key can be defined by specifying additional SEARCH KEY clauses.

'5 The contents of the record type are specified in the CONTAINS clause. The CONTAINS keyword is followed by a list of the elements of the record type. Each record element is defined by a content declaration, where content is as defined under 'Format' above. A record element must not be defined as a non-repeating group, as indicated in the note to figure 3.3. If two or more record elements are declared, each except the last must be followed by a comma which can optionally be followed by spaces.

6 It is not meaningful to include the keyword ALIGNED in a content declaration that declares an array of groups; the data definition of a group so specified determines the alignment or non-alignment of the directly CONTAINed items.

7 ALIGNED must be specified in a content declaration that declares a binary item. An indirectly CONTAINed binary item must be declared as ALIGNED in the data definition of the group in which it is directly CONTAINed. For any other item so specified, either directly or indirectly, the ALIGNED keyword is not applicable in the content declaration in which it is declared and, if specified, is ignored.

8 ALIGNED is the equivalent of COBOL SYNCHRONIZED or PL/I ALIGNED. It means that any binary item declared (as an individual item or as elements of an array) in a content declaration is aligned to a half word or full word boundary, thus:

- binary items having a length of four decimal digits or less occupy a complete half word

- binary items having a length of from five to nine decimal digits occupy a full word.

9 In a content declaration, the ALIGNED keyword and the KNOWN-AS clause can, if applicable, be declared in either possible order; but neither may precede any of the other elements of the content declaration.

10 Any direct or indirect reference from the CONTAINS clause to an item is assumed to be to the HELD-AS form of that item. If the item has no HELD-AS form, default assumptions are made as to the relevant form of the item, in the order: DEFAULTED-AS, ENTERED-AS, REPORTED-AS. The form first encountered in this order is taken as the defaulted form, and version is applied within that form as stated under Format above.

11 The DATABASE-KEYS clause indicates which of the data items (perhaps none) of the currently defined record type is a data base key of this or another record type (every record type must have one and only one data base key, not necessarily contained in the record type; see section 2.1).

12 The TRANSLATION-TABLE clause determines, for this record type, the size and location of the Data Base Key Translation Table (DBTT) and the size of the hash area for each secondary key defined in a SEARCH KEY clause (USING CALC). If the SIZE sub-clause is omitted, a default of one block is assumed for the DBTT and one block for each hash area. If the LOCATED-IN clause is omitted, the DBTT is located in the first area named in the WITHIN clause. If the LOCATED-IN clause is present, the named area must not be a temporary area (see section 2.3.2). A more detailed discussion of SUDS Data Base Key Translation Tables appears.in the Siemens BS2000 UDS Design and Definition User's Guide.

13 If a primary key has been specified (via LOCATION-MODE CALC) for the record type being defined, then the POPULATION clause is used to determine the size and location (k FOR area-name) of the required hash area(s). If the POPULATION clause is omitted, one block is reserved for a hash area in each of the SUDS areas named in the WITHIN clause.

14 The PLACEMENT-SET clause is used to ensure that, within occurrences of the named set (set-name), any member records of the record type currently defined are stored in the vicinity of their corresponding owner records, provided that:

- the record type being defined is specified as an AUTOMATIC member in the SUDS-SET data definition (see section 3.4) of set-name

- each SUDS-AREA named in the WITHIN clause for this record type must also be specified in the WITHIN clause (of the SUDS-RECORD data definition) for the record type specified as owner in set-name's data definition

- a POPULATION j clause must be included in set-name's data definition for which the integer j is greater than zero.

The desired effect of a PLACEMENT-SET clause is not achieved (during SUDS processing) if any of the following are true:

- the record type here defined is a member record type in any set (including set-name) whose SUDS-SET data definition includes a MODE LIST specification

- the owner record type in (the set) set-name is a member record type in any (other) set whose data definition includes a MODE LIST specification

- the data definition for the owner record type (in set-name) includes a LOCATION-MODE CALC or a PLACEMENT-SET specification.

15 Each INDEX clause present must include (as table-name) the name of a search key table or hash area which also appears as table-name in a SEARCH KEY clause for this record type. For a search key table (SEARCH KEY USING INDEX), the INDEX clause defines the location and type of table and the permissible extent of reorganization. For a hash area (SEARCH KEY USING CALC), the INDEX clause defines only the location. If the PLACED-IN clause is omitted, the search key table or hash area is placed in the first SUDS-AREA named in the WITHIN clause for this record type.

16 If the COMPRESSION keyword is specified, records of this record type are stored in compressed form. The keyword should not be specified if the record type contains any variable length items or if it is a member record type of any set whose SUDS-SET data definition includes a MODE LIST specification.

17 Common clauses, listed under Format above, can be present in any type of data definition statement; they are therefore documented separately, in section 4.3 of the DATAMANAGER User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any other clause identifier or other keyword available in the data definition syntax for this member type.

18 Clauses can be declared in any order, provided that subordinate clauses are not separated from the other elements of the clause of which they form a part.

19 A record containing the SUDS-RECORD data definition statement can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 in the DATAMANAGER User's Guide or Chapter 5 in the CONTROLMANAGER User's Guide) and an encoded record can subsequently be generated and inserted into the data entries data set. If, when the encoded record is generated, any member whose name appears in the data definition statement has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as:

- a dummy SUDS-AREA if the member's name appears in the WITHIN clause, the TRANSLATION-TABLE clause, the POPULATION clause, or an INDEX clause

- a dummy ITEM if the member's name appears in the CONTAINS clause, the DATABASE-KEYS clause, or a SEE clause, or if it appears as item-name in the LOCATION-MODE clause or a SEARCH KEY clause

- a dummy SUDS-RECORD if the member's name appears as record-name in the LOCATION-MODE clause
- a dummy MODULE if the member's name appears as module-name in the CALC sub-clause of either the LOCATION-MODE clause or a SEARCH KEY clause
- a dummy SUDS-SET if the member's name appears in the PLACEMENT-SET clause.

Example

An example is given in section 2.2.

3.6 THE SUDS-VIEW DATA DEFINITION STATEMENT

The SUDS-VIEW data definition statement enables you to define to DATAMANAGER a user's processing view of a SUDS record type for use in a SUDS subschema.

Format

SUDS-VIEW

RECORD record-name

CONTAINS content [,content]...

[common clauses1

{;/.}

where

record-name is the name of a member that is a SUDS-RECORD

content declares an item, a subordinate group or an array, in the format shown in figure 3.3 on page 3-12

where

item-name is the name of a member that is an ITEM

version is an unsigned integer in the range 1 to 15, being a number that indicates which version of the specified item is relevant to this definition. The version is within the HELD-AS form or within a defaulted form, just as indicated for version in remark 12 of the SUDS-RECORD data definition statement specification. If version is omitted, a default value of 1 is assumed.

group-name is the name of a member that is a GROUP

local-name is a name, conforming to the rules for member names stated in section 2.4 of the DATAMANAGER User's Guide, that can be used instead of the name or alias of the contained member, when record layouts or source language data descriptions are generated from this data definition by the DATAMANAGER Source Language Generation facility. local-name is not separately recorded in the data dictionary (that is, no dummy data entries record and no index record is created for local-name when the data definition in which it appears is encoded) so local-name cannot be interrogated and can be the same as another name, an alias or a catalog classification in the data dictionary. local-name is the name by which the contained member is known only within the view defined by this data definition.

n is a positive integer greater than or equal to two, being the number of times item-name or group-name occurs in the array

common clauses are any of the following clauses as defined in section 4.3 of the DATAMANAGER User's Guide:

-ACCESS-AUTHORITY	- FREQUENCY
- ADMINISTRATIVE-DATA	- NOTE
- ALIAS	-OBSOLETE-DATE
- CATALOGUE	- QUERY
- COMMENT	- SECURITY-CLASSIFICATION
- DESCRIPTION	- SEE
-EFFECTIVE-DATE	

Remarks

The SUDS-VIEW data definition statement defines a subset of a SUDS record type's data items to DATAMANAGER as a user's processing view (of the record type) for use in a SUDS subschema. In combination with the SUDS-SUBSCHEMA data definition statement (see section 3.7), appropriate use of the SUDS-VIEW data definition statement permits you to define a Siemens UDS subschema to DATAMANAGER. In so doing, you must comply with the rules given in the Siemens BS2000 UDS Design and Definition User's Guide for defining a subschema.

2 The RECORD record-name clause identifies the record type whose processing view is currently defined.

3 The CONTAINS clause identifies the elements of record-name which comprise the current processing view. The same restrictions and conditions that apply to each element of the record type (see remarks 5 to 10 of section 3.5) apply in the same way to those elements that are also CONTAINed in the current processing view. It should be noted, however, that a non-repeating group can be declared in the SUDS-VIEW CONTAINS clause (not permissible in the SUDS-RECORD CONTAINS clause of record-name; see the note in figure 3.3). This must be a GROUP member defined as a collection of (directly or indirectly) CONTAINED elements of record-name. This is illustrated in the example of section 2.2, where the non-repeating group STOCK, declared in ARTICLE-VU, itself CONTAINS record elements of ARTICLE.

4 Common clauses, listed under Format above, can be present in any type of data definition statement; they are therefore documented separately, in section 4.3 of the DATAMANAGER User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any other clause identifier or other keyword available in the data definition syntax for this member type.

5 Clauses can be declared in any order, provided that subordinate clauses are not separated from the other elements of the clause of which they form a part.

6 A record containing the SUDS-VIEW data definition statement can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 of the DATAMANAGER User's Guide or Chapter 5 of the CONTROLMANAGER User's Guide) and an encoded record can subsequently be generated and inserted into the data entries data set. If, when the encoded record is generated, any member whose name appears in the data definition statement has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as:

- a dummy SUDS-RECORD if the member's name appears in the RECORD clause

- a dummy ITEM if the member's name appears in the CONTAINS clause or a SEE clause.

Example

An example is given in section 2.2.

3.7 THE SUDS-SUBSCHEMA DATA DEFINITION STATEMENT

The SUDS-SUBSCHEMA data definition statement enables you to define a SUDS subschema to DATAMANAGER down to the level of record type.

Format

```
SUDS-SUBSCHEMA

ACCESSES  database-name

[PRIVACY-LOCK  'password'  [,'password' [,'password']]]

[PRIVACY-KEY  'password'1

{ALL-AREAS                                  }
{AREAS  area-name  C,area-name]....         }

{ALL-RECORDS                                                    }
{RECORDS {record-name} [,{record-name}]...                      }
{        {view-name  }   {view-name  }                          }

[{ALL-SETS                                  }|
{SETS  set-name  [,set-name]...             }

[common  c  lausesl

{;}
{.}
```

where

database-name is the name of a member that is a SUDS-DATABASE

password is a character string of up to ten printable characters

area-name is the name of a member that is a SUDS-AREA

record-name is the name of a member that is a SUDS-RECORD

view-name is the name of a member that is a SUDS-VIEW

common clauses are any of the following clauses as defined in section 4.3 of the DATAMANAGER User's Guide:

-ACCESS-AUTHORITY	- FREQUENCY
- ADMINISTRATIVE DATA	- NOTE
- ALIAS	-OBSOLETE-DATE
- CATALOGUE	- QUERY
- COMMENT	-SECURITY-CLASSIFICATION
- DESCRIPTION	-SEE
-EFFECTIVE-DATE	

Remarks

1 The SUDS-SUBSCHEMA data definition statement defines a user's processing view of a SUDS schema to DATAMANAGER down to the level of record type. In combination with the SUDS-VIEW data definition statement (used as required; see remark 8 below and section 2. I), the SUDS-SUBSCHEMA data definition statement permits you to define a Siemens UDS subschema to DATAMANAGER. In so doing, you must comply with the rules given in the Siemens BS2000 UDS Design and Definition User's Guide for defining a subschema.

2 The ACCESSES clause is mandatory. It identifies the SUDS schema (via the indicated SUDS-DATABASE member) from which the currently defined subschema is derived.

3 The PRIVACY-LOCK clause enables you to specify up to three passwords for the subschema currently being defined. To ensure compilation of a (COBOL) program which calls this subschema, one of these passwords must be coded as a privacy key in the program's Identification Division, See section 2.3.3 for further discussion of privacy locks and privacy keys.

4 The PRIVACY-KEY clause is used to specify a password permitting access to database-name if its data definition contains a PRIVACY-LOCK clause in which the password appears.

5 The ALL-AREAS keyword, if present, indicates that all SUDS-AREA s specified in the data definition of database-name are available for processing via the currently defined subschema.

6 The AREAS clause specifies a selection of SUDS-AREA s available for processing via the currently defined subschema. Each SUDS-AREA specified must also be named in the AREAS clause of database-name's data definition.

7 All SUDS-AREAs which are required for the record types and sets of the currently defined subschema (see remarks 8, 9, 11, and 12 below) and which are also accessible to users of the subschema (see section 2.1 on data privacy) should be made available for processing by means of the ALL-AREAS keyword or the AREAS clause. Either ALL-AREAS or the AREAS clause must appear in a SUDS-SUBSCHEMA data definition.

8 ALL-RECORDS. if present, indicates that every record type belonging to any of database-name's SUDS-AREAs is available for processing via the currently defined subschema.

9 The RECORDS clause specifies a selection of database-name's record types (or their processing views; see section 3.6) that are available for processing via the subschema.

10 Every record type (or its processing view) belonging to a set of the currently defined subschema (see remarks 11 and 12 below) should be made available for processing by means of the ALL-RECORDS keyword or the RECORDS clause. Either ALL-RECORDS or the RECORDS clause must appear in a SUDS-SUBSCHEMA data definition.

11 ALL-SETS, if present, indicates that all SUDS-SET s specified in the data definition of database-name are available for processing via the currently defined subschema.

12 The SETS clause specifies a selection of SUDS-SET s available for processing via the currently defined subschema. Each SUDS-SET specified must also be named in the SETS clause of database-name's data definition.

13 SUDS-SUBSCHEMA data definitions are permissible in which neither the ALL-SETS keyword nor the SETS clause is specified.

14 Common clauses, listed under Format above, can'be present in any type of data definition statement; they are therefore documented separately, in section 4.3 of the DATAMANAGER User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any dther clause identifier or other keyword available in the data definition syntax for this member type.

15 Clauses can be declared in any order, provided that shbordinate clauses are not stparated from the other elements of the clause of which they form a part.

16 A record containing the SUDS-SUBSCHEMA data definition statement can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 of the DATAMANAGER User's Guide or Chapter 5 of the CONTROLMANAGER User's Guide) and an encoded record can subsequently be generated and inserted into the data entries data set. If. when the encoded record is generated, any member whose name appears in the data definition statement has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as:

- a dummy SUDS-DATABASE if the member's name appears in the ACCESSES clause
- a dummy SUDS-AREA if the member's name appears in the AREAS clause
- a dummy SUDS-RECORD if the member's name appears in the RECORDS clause
- a dummy SUDS-SET if the member's name appears in the SETS clause
- a dummy ITEM if the member's name appears in a SEE clause.

Example

An example is given in section 2.2.

3.8 SYSTEM, PROGRAM AND MODULE DATA DEFINITON STATEMENTS FOR A SUDS ENVIRONMENT

3.8.1 Introduction

The data definition statements for DATAMANAGER SYSTEM, PROGRAM and MODULE members acting on conventional files are described in the DATAMANAGER User's Guide. For the SUDS Interface, a further clause, the PROCESSES clause, is included in the format of these statements, to specify which processing views of the database (that is, which SUDS subschemas) are relevant to the member. The PROCESSES clause is defined in section 3.8.2. For a full specification of the SYSTEM, PROGRAM and MODULE data definition statements in a SUDS environment, therefore, section 3.8.2 must be read in conjunction with section 4.2 of the User's Guide.

3.8.2 Specification of the PROCESSES Clause

Format

```
PROCESSES SUDS SUBSCHEMAS subschema-name [,subschema-name]...
```

where

subschema-name is the name of a member that is a SUDS-SUBSCHEMA

Remarks

1 The PROCESSES clause is used to specify the names of the SUDS-SUBSCHEMA members to which this SYSTEM, PROGRAM or MODULE relates.

2 A record containing the data definition of the SYSTEM, PROGRAM or MODULE that includes the PROCESSES clause can be inserted into the dictionary's source data set by a suitable command (see Chapter 3 of the DATAMANAGER User's Guide or Chapter.5 of the CONTROLMANAGER User's Guide) and an encoded record can subsequently bc generated and inserted into the data entries data set. If, when the encoded record is generated, any member whose name appears in the data definition statement has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is created as a dummy SUDS-SUBSCHEMA.

Example

An example is given in section 2.2.

CHAPTER 4 SUDS SOURCE LANGUAGE GENERATION FROM DATAMANAGER

4.1 INTRODUCTION

The DATAMANAGER Source Language Generation facility can be used to produce Siemens Universal Database System (SUDS) statements of the following types from encoded data definitions held in a DATAMANAGER data dictionary:

- Schema Storage Structure Language (SSL)
- Schema Data Description Language (DDL)
- Subschema Data Description Language (DDL).

Generation of these statements is achieved by use of the PRODUCE command.

The basic form of the command, which can generate record layouts or COBOL, PL/I or Assembler data descriptions for conventional tile environments, is described in the separate DATAMANAGER manual entitled Source Language Generation. Users should refer to that manual for a general description of source language generation *and* of the PRODUCE command. The record layouts, COBOL, PL/I or Assembler data descriptions can be produced from SUDS-RECORD and SUDS-VIEW members if required.

The version of the command used to produce SUDS Storage Structure Language or Data Description Language is described in section 4.2 below. SUDS source language generation requires the SUDS Generation Facility selectable unit DMR-SL10. This facility runs in Batch mode or Line mode (that is the pseudo-interactive environments ICCF, TSO/ISPF, ROSCOE). The COBOL Source Language Generation facility, selectable unit DMR-SL1, is also required.

Certain parts of the output generated when producing SUDS Data Description Language can be tailored by means of the DATAMANAGER installation macro DGCOB. This macro is documented in the Source Language Generation manual, Appendix 1.2.

In the specifications in this chapter, any MSP-defined conditions or values that can be tailored by the Controller are annotated "(unless tailored, see xxxx)", where xxxx is the relevant keyword of the macro DGCOB.

4.2 SPECIFICATION OF THE PRODUCE COMMAND FOR SUDS SOURCE LANGUAGE GENERATION

Format

```
PRODUCE UDS-DB {{SSL    } FROM  database-name
                {SCHEMA}
               {SUBSCHEMA FROM  subschema-name}

[{ONTO filename}] [{PRINT    }]
 {NOGENERATION} {NOPRINT  }
 {NO-GENERATION} {NO-PRINT}

[{WITH-ALIAS} [{number     }]] {;}
 {ALIAS     } {alias-type}]]  {.}
```

where

database-name is the name of an encoded SUDS-DATABASE member

subschema-name is the name of an encoded SUDS-SUBSCHEMA member.

tile-name is the name of the output source library data set. It is the logical file name (ddname or dtfname) used in job control statements to indicate the external data set name (physical file name) of the file to which generated program source data descriptions are to be written.

Remarks

1 The first two elements of the command must be PRODUCE and UDS-DB, in that order. Other elements present in the command must be in the order shown above under Format.

2 If SSL is stated in the command, the Schema Storage Structure Language specifications are generated from the SUDS-DATABASE member named in the FROM clause; and from the SUDS-SET and SUDS-RECORD members associated with the database. The contents of the DATAMANAGER definition of the database will include SUDS-SETs and SUDS-AREAs; the SUDS-AREAs will be linked to SUDS-RECORD members by a 'WITHIN' clause in each record. SUDS Schema Storage Language specifications will be generated from data regarding physical storage, given in the SUDS-SET and SUDS-RECORD DATAMANAGER definitions.

3 If SCHEMA is stated in the command, the Schema Data Description Language specifications are generated from the SUDS-DATABASE member named in the FROM clause; and from the SUDS-AREA, SUDS-SET, and SUDS-RECORD members directly and indirectly defined in the SUDS-DATABASE member. The SUDS specifications generated include a list of SUDS-AREA names; a set of record layouts of SUDS-RECORDS defined as 'WITHIN' those areas, generated from DATAMANAGER SUDS-RECORD definitions as indicated in the Schema DDL Correspondence Table (section 5.2); and a list of SUDS-SET definitions generated from DATAMANAGER SUDS-SET definitions, as indicated in the Schema DDL Correspondence Table.

4 If SUBSCHEMA is stated in the command, the Subschema Data Description Language specifications are generated from the SUDS-SUBSCHEMA member named in the FROM clause, and from any SUDS-VIEW members specified in the SUDS-SUBSCHEMA member. The SUDS specifications generated include a list of SUDS-AREA names and a set of record layouts of SUDS-RECORDS or of records defined in DATAMANAGER definitions of SUDS-VIEW members. (See the Subschema DDL Correspondence Table, section 5.4.)

5 Only one database-name or subschema-name can be declared in the FROM clause.

6 If the database-name or subschema-name specified in the FROM clause:

- is not encoded, or
- is not present in the data dictionary, or
- is of a member type that is not valid in the context, or
- is protected against access by the user (see remark 7),

then a message is output and no generation takes place.

7 Acceptance of the PRODUCE command is subject to access security levels. For details of the use of security levels, see the specification of the PROTECT command in the DATAMANAGER User's Guide, and the description of the security system in section 1.7 of the User's Guide.

If the database or subschema member named in the FROM clause has an access security level higher than the user's (general or specific) security level, the command is rejected. If the command can be accepted in respect of that member, but a reference path from it includes a protected member with an access level higher than the user's security level, the reference path supplies the names of members up to, but not beyond, the last member to which the user has access. The PRODUCE command follows the reference path to its end, in order to determine the total storage space required. For each member beyond the last member in the reference path to which the user has access, a filler name is generated and an error message is output, indicating that an attempt has been made to access a member whose protection level is too high.

8 If the ONTO clause is present in the PRODUCE command, the Storage Structure Language or Data Description Language specifications or statements are written to the file defined in the ONTO clause.

9 If the ONTO clause is omitted and the NOGENERATION (or NO-GENERATION) keyword is also omitted, a default of ONTO GENLIB is assumed (unless file-name is tailored: see DDNAME in the Source Language Generation manual).

10 If NOGENERATION or NO-GENERATION is stated in the command, no output file is written: but the generation of Storage Structure Language and Data Description Language, and their output to a printer or terminal, is not inhibited.

11 The keywords PRINT and NOPRINT or NO-PRINT relate to the printing or display of generated source language data descriptions. They have no effect on the output of record layouts or of messages.

12 The Storage Structure Language and the Data Description Language are printed as they are produced, unless NOPRINT or NO-PRINT is stated in the command.

13 The keywords WITH-ALIAS and ALIAS are equivalent.

14 The ALIAS (or WITH-ALIAS) clause only operates on members of the SUDS-AREA type. It is there so that you can generate data set names of a maximum length of 54 characters.

15 If a WITH-ALIAS (or ALIAS) clause is present in the command, generated data names are based wherever possible on aliases taken from the relevant SUDS-AREAs' ALIAS clauses, in accordance with the rules stated in remarks 16 and 17, instead of on the SUDS-AREA s' names.

16 If a WITH-ALIAS or ALIAS clause specifies a number, n, each generated data name is based if possible on the nth general alias in the SUDS-AREA's ALIAS clause.

17 If a WITH-ALIAS (or ALIAS) clause is operative, and a SUDS-AREA from which generation is taking place has no alias of the specified number, a message is output and the data name generated in respect of that area is based on its DATAMANAGER member name.

Examples

The following examples are based on the Sample DATAMANAGER SUDS-specific Member Definitions given in section 2.2, and the SUDS-DATAMANAGER correspondence Tables given in Chapter 5.

Example1

```
PRODUCE SSL FROM MAIL-ORDERS;
```

This generates the SSL specifications of every record type and set in the MAIL-ORDER database:

```
STORAGE STRUCTURE OF SCHEMA-.HAIL-ORDERS.
RECORD NAME IS PURCHASE-ORDER
      DATABASE-KEY-TRANSLATION-TABLE IS 200.
RECORD NAME IS SUPPLIER
      POPULATION IS 20.
            :
SET NAME IS PURCHASE-ORDER-CONTENTS
      MODE IS LIST DETACHED WITH PHYSICAL LINK
      POPULATION IS 20.
SET NAME IS MIN-STOCK-LEVEL
      MODE IS CHAIN LINKED TO PRIOR.
            :
SET NAME Ii ORDERED-ARTICLES
      MEMBER IS PHYSICALLY LINKED TO OWNER.
```

Note

The contents of the MAIL-ORDERS database (see section 2.2) are examined via the DATAMANAGER SUDS-DATABASE definition. There are lists of AREAS and SETS, and a PRIVACY-LOCK.

Within each area are RECORDs. A record is linked to an area by a 'WITHIN' clause in the DATAMANAGER SUDS-RECORD definition. The SUDS area PURCHASE-ORDER-REALM is linked to the SUDS records PURCHASE-ORDER and SUPPLIER in this way. From the list of areas in the DATAMANAGER SUDS-DATABASE definition, a list of records is generated, with their physical clauses, as above.

The DATAMANAGER SUDS-DATABASE definition is used directly to generate SUDS sets, as indicated in the Correspondence Tables (Chapter 5).

Example 2

```
PRODUCE SCHEMA FROM NAIL-ORDERS;
```

The SUDS Schema Data Description Language will be generated as:

```
SCHEMA NAME IS MAIL-ORDERS.
PRIVACY LOCK FOR COPY IS SHIP-KEY.
AREA NAME IS CUSTOMER-ORDER-REALM.
AREA NAME IS PURCHASE-ORDER-REALM.
          :
          .
AREA NAME IS SEARCH-REALM
      AREA IS TEMPORARY.
RECORD NAME IS CUSTOMER
      LOCATION MODE IS DIRECT CUST-NO OF CUSTOMER
      WITHIN CUSTOHER-ORDER-RLM.
01    CUST-NAME                TYPE IS CHARACTER 30.
01    CUST-F-NAME              TYPE IS CHARACTER 30.
01    CUST-NO                  TYPE IS DATABASE-KEY.
RECORD NAME IS ARTICLE-TYPE
      WITHIN CLOTHING, HOUSEHOLD-GOODS, SPORTS-ARTICLE,
          FOOD, LEISURE, STATIONERY AREA-ID IS
          RLH-SELECTION-1
      SEARCH KEY IS ART-NAME USING CALC
          NAME IS SEARCH-TAB-ARTICLE-TYPE DUPLICATES ARE
          ALLOWED.
01    ART-NAME                 TYPE IS CHARACTER 25.

          :
SET NAHE IS PURCH-ORD-CONTENTS
      ORDER IS NEXT
      OWNER IS PURCHASE-ORDER.
MEMBER IS PURCH-QRD-ITEM MANDATORY AUTOMATIC
      SET OCCURRENCE SELECTION IS THRU CURRENT OF SET.
```

CHAPTER 5 SUDS-DATAMANAGER CORRESPONDENCE TABLES

5.1 INTRODUCTION

This chapter contains tables that relate the SUDS Schema and Subschema Data Description Language (DDL) and Storage Structure Language (SSL) specifications to the corresponding DATAMANAGER data definition statement specifications.

Correspondence Between SUDS Schema Data Description Language (DDL) and DATAMANAGER Data Definition Statements	
SUDS Schema DDL Syntax	**DATAMANAGER Syntax**
(Schema Description) SCHEMA NAME is schema name PRIVACY LOCK FOR COPY IS literal OR Literal OR literal	 database-name. SUDS-DATABASE PRIVACY-LOCK 'password','password', 'password'
(Area Description) AREA NAME IS realm-name AREA IS TEMPORARY	 area-name. SUDS-AREA TEMPORARY
(Record Description) RECORD NAME IS record-name LOCATION MODE IS DIRECT item-name {IN / OF} record-name identifier CALC hash-routine USING item-name DUPLICATES ARE [NOT] ALLOWED WITHIN realm-name C,realm-name [,realm-name]... AREA-ID IS identifier] SEARCH KEY IS item-name [,item-name]... USING {CALC hash-routine / INDEX} NAME IS name DUPLICATES ARE [NOT] ALLOWED record-element-clause... where each record-element-clause describes an item, a vector, or a repeating group	 record-name. SUDS-RECORD LOCATION HODE DIRECT KEY item-name {IN / OF} record-name ID identifier CALC PROCEDURE module-name USING item-name DUPLICATES {ALLOWED / DISALLOWED} WITHIN area-name [AND area-name [,area-name]... AREA-ID identifier] SEARCH KEY item-name [,item-name]... USING {CALC PROCEDURE module-name / INDEX} NAME table-name DUPLICATES {ALLOWED / DISALLOWED} CONTAINS content [,content]...
(Item Description) item-name PICTURE IS mask-string TYPE IS FIXED REAL BINARY {15 / 31}	 item-name. ITEM PICTURE 'picture' BINARY {4 / 9}

(continued)

(continued)

Correspondence Between SUDS Schema Data Description Language (DDL) and DATAMANAGER Data Definition Statements

SUDS Schema DDL Syntax	DATAMANAGER Syntax
TYPE IS FIXED REAL DECIMAL n, m where $0 \leq m \leq n \leq 18$, $1 \leq n$	{DECIMAL-PACKED / PACKED-DECIMAL} r.m **where r = n-m**
TYPE IS CHARACTER n	CHARACTER n
TYPE IS DATABASE-KEY	BINARY 9 **item-name must also appear in the SUDS-RECORD DATABASE-KEYS clause**
(Vector Description) **same as for Item Description** **plus** OCCURS n TIMES	**(n) item-name must appear in the SUDS-RECORD CONTAINS clause**
(Repeating Group Description) group-name OCCURS n TIMES	group-name. GROUP **(n) group-name must appear in the SUDS-RECORD CONTAINS clause**
(Set Description) SET NAME IS set-name	set-name. SUDS-SET
SET IS DYNAMIC	DYNAMIC
ORDER IS LAST FIRST NEXT PRIOR IMMATERIAL SORTED INDEXED NAME IS name BY {DATABASE-KEY / DEFINED KEYS} DUPLICATES ARE [NOT] ALLOWED	ORDER LAST FIRST NEXT PRIOR IMMATERIAL SORTED INDEXED NAME table-name {DATABASE-KEY / DEFINED-KEY} DUPLICATES {ALLOWED / DISALLOWED}
OWNER IS {record-name! / SYSTEM}	OWNER record-name
MEMBER IS record-name MANDATORY OPTIONAL AUTOMATIC MANUAL	MEMBER record-name MANDATORY OPTIONAL AUTOMATIC MANUAL

(continued)

(continued)

Correspondence Between SUDS Schema Data Description Language (DDL) and DATAMANAGER Data Definition Statements	
SUDS Schema DDL Syntax	DATAMANAGER Syntax
- - - DESCENDING KEY IS item-name [,item-name]...	ASCENDING DESCENDING SORT KEY item-name [,item-name]...
SEARCH KEY IS item-name [,item-name]... USING {CALC hash-routine / INDEX} NAME IS name DUPLICATES ARE [NOT] ALLOWED	SEARCH KEY item-name [,item-name]... USING {CALC PROCEDURE module-name / INDEX} NAME table-name DUPLICATES {ALLOWED / DISALLOWED}
SET OCCURRENCE SELECTION IS THRU CURRENT OF SET LOCATION MODE OF OWNER ALIAS FOR {item-name / identifier-1} IS identifier-2	SELECTION CURRENT OWNER NAME id-name-1 IS id-name-2

5.3 SCHEMA SSL CORRESPONDENCE TABLE

Correspondence Between SUDS Schema Storage Structure Language (SSL) and DATAMANAGER Data Definition Statements

SUDS Schema SSL Syntax	DATAMANAGER Syntax
(Schema Description)	
STORAGE STRUCTURE OF SCHEMA schema-name	database-name. SUDS-DATABASE
(Record Description)	
RECORD-NAME is record-name	record-name. SUDS-RECORD
DATABASE-KEY-TRANSLATION-TABLE IS integer WITHIN realm-name	TRANSLATION-TABLE SIZE j LOCATED-IN area-name
POPULATION IS integer WITHIN realm-name	POPULATION k FOR area-name
PLACEMENT OPTIMIZATION FOR SET set-name	PLACEMENT-SET set-name
INDEX NAME IS name PLACING IS WITHIN realm-name	INDEX table-name PLACED-IN area-name
TYPE IS DATABASE-KEY-LIST REPEATED-KEY DYNAMIC REORGANIZATION SPANS integer PAGES	TYPE DB-KEY-LIST REPEATED-KEY REORGANIZATION m
COMPRESSION FOR ALL ITEMS	COMPRESSION
(Set Description)	
SET-NAME IS set-name	set-name. SUDS-SET
POPULATION IS integer-l INCREASE IS integer-2	POPULATION j INCREASE k
MODE IS CHAIN LINKED TO PRIOR POINTER-ARRAY LIST ATTACHED TO OWNER DETACHED WITHIN realm-name WITH PHYSICAL LINK	MODE CHAIN LINKED-PRIOR POINTER-ARRAY LIST ATTACHED DETACHED WITHIN area-name PHYSICAL
DYNAMIC REORGANIZATION SPANS integer PAGES	PAGES m
INDEX NAME IS name PLACING IS ATTACHED TO OWNER DETACHED WITHIN realm-name	INDEX table-name PLACED ATTACHED DETACHED IN area-name

(continued)

(continued)

Correspondence Between SUDS Schema Storage Structure Language (SSL) and DATAMANAGER Data Definition Statements	
SUDS Schema SSL Syntax	DATAMANAGER Syntax
TYPE IS DATABASE-KEY-LIST REPEATED-KEY DYNAMIC REORGANIZATION SPANS integer PAGES MEMBER IS PHYSICALLY LINKED TO OWNER	TYPE DE-KEY-LIST REPEATED-KEY REORGANIZATION n LINKED-OWNER

5.4 SUBSCHEMA DDL CORRESPONDENCE TABLE

Correspondence Between SUDS Subschema Data Description Language (DDL) and DATAMANAGER Data Definition Statements

SUDS Subschema DDL Syntax	DATAMANAGER Syntax
(Subschema Description)	
IDENTIFICATION DIVISION.	
SUBSCHEMA NAME IS subschema-name	subschema-name. SUDS-SUBSCHEMA
OF SCHEMA NAME schema-name	ACCESSES database-name
PRIVACY LOCK FOR COMPILE IS literal OR literal OR literal	PRIVACY-LOCK 'password','password', 'password'
PRIVACY KEY FOR COPY IS Literal	PRIVACY-KEY 'password'
DATA DIVISION.	
AREA SECTION.	
COPY ALL AREAS COPY realm-name [,realm-name]...	ALL-AREAS AREAS area-name [,area-name]...
RECORD SECTION.	
COPY ALL RECORDS COPY record-name C,record-name]...	ALL-RECORDS RECORDS {record-name / view-name} [,{record-name / view-name}]...
record-clause...	
where each record-clause is a view of a distinct record type, record-name-l, not specified in the COPY clause	
	(SUDS-VIEW Description)
	view-name. SUDS-VIEW
record-name-l	RECORD record-name-l
record-element-clause...	CONTAINS content [,content]... (see Note 1)
SET SECTION.	
COPY ALL SETS COPY set-name C,set-name]...	ALL-SETS SETS set-name [,set-name]...

Notes

1 It is not necessary for the user of the DATAMANAGER SUDS Interface to enter COBOL for the elements of a record type which are required for a subschema. The user need only reference the corresponding SUDS-RECORD data definition in a SUDS-VIEW data definition and enter an appropriate 'content' declaration for each of its elements required in the subschema.

USAGE DIRECTORY

A

Alias 4-2/4
Area 2-2/3, 3-4, 3-13, 3-15/16, 4-2/4
Assembler 4-1

B

BULK Command 1-3

C

Circumstances when source language is not generated. 4-3
COBOL Source Language Generation facility. 4-1
Commands to process SUDS member types. 1-1, 1-3

D

Data base key 2-1, 3-8, 3-13/15
Data Base Key Translation Table. 3-13, 3-15
Database Name. 4-2/3
Data Description Language. 1-4, 4-1/3, 5-1, 5-2/4, 5-7
DDL: see Data Description Language
Ddname, dftname. 4-2
DMR-SLI 4-1
DMR-SL10 4-1
Dynamic set. 2-6, 2-10, 3-4, 3-7

E

Example of a SUDS database. 2-4/9
Examples of SUDS source language generation. 4-4/5

F

Facilities offered by the interface: summary. 1-1
File name 4-2
FROM clause 4-2/3

G

GLOSSARY command. 1-3, 1-5

H

Hash area 3-8, 3-10, 3-13/16
Hash routine. 3-8, 3-14
Hierarchy of member types. 1-1/2

I

ICCF 4-1

L

LIST command. 1-3

M

Macro DGCOB. 4-1
Member record type. 2-2/3, 3-5, 3-7/8, 3-15/16

Member types
- hierarchy 1-1/2
- keywords in commands. 1-3
- summary 1-I

MODULE members. 1-1, 2-3, 3-1, 3-24

N

NOGENERATION keyword 4-2/3
NOPRINT keyword 4-2/3

O

ONTO clause. 4-2/3
Output source library data set. 4-2
Owner record type. 2-2/3, 3-5, 3-7/8, 3-15/16

P

PERFORM command. 1-3
PL/I 4-1
Primary key. 2-1, 2-3, 3-7/8, 3-13/15
Privacy key. 2-10/11, 3-2, 3-21/22
Privacy lock. 2-10/11, 3-2, 3-21/22
PROCESSES clause. 11, 2-1, 2-3, 3-1, 3-24
PRODUCE command. 3-7, 4-1/3
PROGRAM member. 1-1, 2-3, 3-1, 3-24
PRINT keyword. 4-2/3
PROTECT command. 4-3
Pseudo interactive environments. 4-1

R

Realm 2-2/3, 3-4, 3-13
see also Area
Record layout. 4-1/2
Record type. 2-1/3, 3-10, 3-13/16, 3-18/19
REPORT command. 1-3
ROSCOE 4-1

S

Schema 2-3, 2-4/6, 2-10, 3-2, 3-21, 4-1/2, 5-1/6
Schema syntax. 5-2/6
Secondary key 2-1, 2-3, 3-8, 3-13/15
Security 4-3
Set. 2-2/3, 3-5, 3-7/8
Source Language Generation 1-4, 3-7, 3-10, 3-13, 3-18/19, 4-1/5
SSL: see Storage Structure Language
Storage space determination. 4-2/3
Storage Structure Language. 1-4, 4-1/3, 5-1, 5-5/6
Subschema. 2-3, 2-4, 2-6, 2-9, 2-10/11, 3-18/19, 3-21/22, 4-1, 4-3, 5-1, 5-7
Subschema name. 4-2/3
Subschema syntax. 5-7
SUDS-AREA 1-1/2, 2-3, 3-1, 34, 4-2/3
SUDS-DATABASE 1-1/3, 2-3, 2-4, 2-6, 2-11, 3-1/3, 4-2
SUDS database. 1-1, 2-1/3, 2-4, 4-3
SUDS-RECORD 1-1/4, 2-3, 3-1, 3-10/17, 3-18/19, 4-1/2
SUDS schema DDL 1-4, 4-1/3, 4-5, 5-1, 5-2/4
SUDS schema SSL. 1-4, 4-1/4, 5-1, 5-5/6
SUDS-SET.. . . . . 1-1/4, 2-3, 3-1, 3-5/9, 4-2
SUDS-SUBSCHEMA 1-1/3, 2-3, 2-4, 2-9, 2-11, 3-1, 3-19, 3-21/23, 3-24, 4-2

SUDS subschema DDL 1-4, 5-1, 5-7
SUDS-VIEW 1-1/3, 2-3, 3-1, 3-18/20, 4-1/2
SYSTEM members. 1-1, 2-3, 3-1, 3-24

T

Temporary area.. ..... 24, 2-10, 3-4, 3-7. 3-13, 3-15, 4-5
TSO/ISPF 4-1

U

Unique identifiers. 2-1, 3-8

W

WHAT command l-3
WHICH command l-3
WITH-ALIAS clause. 4-3/4

AMENDMENT RECORD

This manual when issued should contain 66 pages (33 sheets). comprising:

Preliminary pages	8 pages (4 sheets)
Chapter 1	6 pages (3 sheets)
Chapter 2	12 pages (6 sheets)
Chapter 3	24 pages (12 sheets)
Chapter 4	2 pages (1 sheet)
Chapter 5	8 pages (4 sheets)
Usage Directory	4 pages (2 sheets)
Amendment Record	2 pages (1 sheet)

(Blank pages occurring at the end of chapters or appendices are included in the page counts.)

Please check that all pages are present immediately on receipt. If any pages are missing, please inform your local MSP Product Supplier without delay.

If any updates to the information contained in this manual become necessary, additional or replacement pages will be issued. Such pages will be accompanied by a front sheet, the amendment list, detailing the action required to update the manual. On completion of the updating, the amendment list should be filed behind this page. An entry should be made in the table overleaf to record the action taken.

Each page of this manual is annotated (immediately above the page reference) with the publication date of the page. Update pages will be similarly annotated. It is thus always possible to check that the latest version of a page is held.

Amendment Record

Amendment List Number	Date of Incorporation	Initials	Notes
1	09.97	[illegible]	

MSP Amendment List

DATAMANAGER

Siemens UDS Interface

Amendment List 1

Date of issue: 05.85

In the event of any query on this amendment list, please telephone your local MSP Product Support office at the number given on the card at the front of your MANAGER Products binder.

© Copyright Management Systems and Programming Limited 1985.
All rights reserved.

The amendments listed below should be made to the above-named publication.

Where re-issued pages contain information which is materially different from information in the replaced pages, the new or changed information is indicated by vertical lines in the inner margins of the pages. Complete new sections or appendices are not so marked. If a page is again re-issued, any marginal lines from previous amendments are removed.

	Remove and destroy pages:	Insert the new pages:
1	iii to vi	iii to vi
	4-1, 4-2	4-1 to 4-6
	UD-1 to UD-4	UD-1 to UD-4

2 Update page AR-2 to record the incorporation of Amendment List 1.

3 File this page behind page AR-2.