Bellcore

Bell Communications Research

SNS Database Admin Guide
BR 007-560-406
Issue 2, December 1991
SNS Release 2.0

Service Negotiation Support (SNS)

Database Administration Guide

Prepared by the
Information Management Services Division
Bellcore

Service Negotiation Support (SNS)
Database Administration Guide
CONTENTS

LIST OF FIGURES

LIST OF TABLES

Preface

This document is intended for database administrators (DBAs) of the Service Negotiation Support (SNS) processing layer building block (PLBB). It is assumed that the reader is familiar with the operation of an IMS (Information Management System) system and a DB2^{TM} system and understands basic Centrex and ISDN (Integrated Services Digital Network) terminology.

DB2 is a trademark of International Business Machines, Inc.

1. Introduction and Documentation Index

This document provides database administrators with guidelines for managing the SNS databases. It is the responsibility of the database administrator to assure the integrity and availability of the databases in an SNS environment.

The *Service Negotiation Support (SNS) Database Administration Guide* includes the following sections:

- Section 1 – "Introduction and Documentation Index"
- Section 2 – "Database Description" describes each of the SNS databases. The IMS databases are described in one subsection, and the DB2 database is discussed in another.
- Section 3 – "Database Naming Conventions" provides the suggested database naming conventions.
- Section 4 – "Database Sizing" describes the use of the database sizing model and identifies the factors that influence the sizing of the SNS databases.
- Section 5 – "Database Performance Monitoring and Tuning" specifies the criteria for monitoring the performance of the database management system, notes potential monitoring tools, and provides guidelines for reorganization.
- Section 6 – "Database Security" discusses database management system (DBMS) security. The SNS contract processing and maintenance transaction security strategy is discussed in the *Service Negotiation Support (SNS) System Administration Guide.*
- Section 7 – "Database Back-Up and Recovery" provides information relating to the physical protection of the SNS databases. This includes providing guidelines and procedures for establishing and maintaining the integrity of the databases.
- Section 8 – "Database Reporting Tools" provides information on how to generate database reports.
- Section 9 – "Database Repair" provides references to vendor tools that support database repair. No application-specific tools are provided.
- Appendix A – "Guide to Acronyms" lists and defines the major acronyms used in this document.

In addition to this document, there are several other Bellcore documents that support SNS. They are referenced throughout this document. They include:

1. *Service Negotiation Support (SNS) Contract Specifications*, BR 007-560-403, describes the communication between SNS and the contract originator.
2. *Service Negotiation Support (SNS) System Administration Guide*, BR 007-560-405, covers application installation, security, recovery, troubleshooting, and other topics.
3. *PREMIS/NMAG Application Guide,* BR 007-560-408, explains the procedures for working with the PREMIS/NMAG screens.
4. *PREMIS/NMAG Database Administration Guide,* BR 007-560-409, describes the PREMIS system marketing database area and provides the physical data specifications for the area and the records.
5. *PLATFORM Installation – Installation Planning Guide*, BR 752-105-050, explains the procedures for installing the Provisioning PLATFORM system.
6. *PLATFORM System Security Administration System Administration Guide*, BR 752-106-010, describes PLATFORM-based application security.
7. *PLATFORM System Generic Table System (GTS) System Administration Guide*, BR 752-106-070, describes the Generic Table System and explains the online capabilities of the GTS formats.
8. *PLATFORM System DD/RDAS Administration Guide*, BR 752-106-030, provides instructions on installing IMS databases.

The following IBM documents are referenced in this document:

1. *SQL Usage Guide*
2. *SQL Reference Guide*
3. *DB2 Database Administration Guide*
4. *DB2 Command and Utility Reference*

2. Database Description

The information required by SNS is managed in IMS and DB2 database tables. The IMS database tables used by SNS include tables required by the Provisioning PLATFORM system and the SNS Session and Rules databases. The Session and Rules databases are used by the SNS Negotiation Engine, which is the component of SNS that controls the ongoing negotiation sessions. A *negotiation session* consists of all data and activities corresponding to one instance of an SNS dialogue (e.g., a Network Element Inquiry). Control is defined by a set of *rules* stored in the SNS Rules database. These rules are grouped into sets called *states.*

Each negotiation session has its own set of *session data*, which is stored in the Session database. This includes the contracts and the *facts* that represent the actual events or occurrences that have taken place during a negotiation session. All of the session data for a specific negotiation session is removed at the completion of that session.

The SNS DB2 database relates and manages all of the data items necessary for service negotiation support and the TN selection process employed by SNS, as well as all SNS error, completion, and monitor notices.

The TNLIST database table organization can be categorized by the relationships between several types of data. This structure logically consists of the customer, group, list, range, and TN levels.

A *customer* is identified by an attribute known as tie code, and can be an owner of many groups. A group can represent either a Centrex or non-Centrex service. A Centrex group is usually identified by a main TN and maps to a specific TN list. A TN list is a group of TNs for either a Centrex or non-Centrex group, identified by an internal list identifier, containing many ranges of TNs. For regular Centrex, a group number and device identifier uniquely determine a TN list. For other types of services, a service type, optional product name, and network element (i.e., switching entity) combination uniquely determine a TN list. A TN range is a consecutive group of telephone numbers identified by the list on which they exist. Lists contain information about thresholds for replenishment when the supply of selectable numbers becomes low. TNs are the primary item inventoried in TNLIST and are also identified by the list on which they participate. The selectability status of TNs for Centrex customers is kept on an ongoing basis. Activity to the TNs for a customer's account has an effect on their selectability. The TNs for non-Centrex customers are not tracked by account after they have been selected. Hunt group information, including the hunting sequence, is also stored in TNLIST.

The Negotiation Reference database tables contain the following types of information:

- Feature data
- Network element data
- Feature package data
- Customer premises equipment (CPE) data.

The feature data tables provide information about the availability of features for specific switch types and generics and also indicate how each feature can be configured to provide service. The network element data tables identify the individual network elements (switches) supported by SNS and describe their characteristics. The feature package data tables provide information about BCC-tariffed packages of features for use with a specific product offering. The CPE tables provide information about individual CPE and the features and configurations that can be supported by each.

The Service Request Output (SRO) database tables relate and manage the data items necessary to track active SNS service requests and their related service orders and output messages. In addition, certain SRO database tables are populated by Bellcore with reference data that affect the internal processing of the SRO component of SNS.

The SNS Exception Handler database (SEH) table contains:

- Error, completion, and monitor narrative text
- The system action codes, which instruct SNS in a course of action as a result of the condition
- The relationships between calling module type (i.e., Negotiation versus Maintenance) and system action.

For example, Telephone Number selection via the negotiation process may need an exception message generated in an outgoing contract, which requires a database rollback and processing termination. This message is stored in the SEH table in fields containing the message text, the message code, the module type (negotiation), and the SNS action (terminate and rollback).

2.1 SNS-IMS Database

The following IMS databases exist in SNS:

1. YHALRDD – Performance logging database (PLATFORM system)
2. YHGIN*DD – Reference non-table database (PLATFORM system)
3. YHGTB*DD – Reference Table database (PLATFORM system)
4. YHSIX*DD – Security database (PLATFORM system)
5. YHTTB*DD – GTS (Generic Table System) database (PLATFORM system)
6. YHCES*DD – Session database
7. YHDAR*DD – Rules database
8. YHDDC*DD – PLATFORM meta-data database
9. YHSSD*DP – PLATFORM site source data dictionary database

The asterisk (*) in the database name represents the entity ID. The non-PLATFORM system databases (Session and Rules) are described in the following subsections.

2.1.1 Session Database

The Session database is a repository of session-specific local data. The SNS Negotiation Engine keeps track of the progress of a negotiation via the Session database. When a session is established for the first time, a unique session ID is created. This ID is used as a key to create the root segment in the Session database. No dependent segments are created at this time. When a session is temporarily suspended, states, facts, and contracts are stored in the states, facts, and contracts segments, respectively, under the root segment. When the session resumes, the session ID is used to retrieve the root segment. The dependent segments are also retrieved. This data is sufficient to enable session resumption. States and facts are updated as necessary. The Negotiation Engine does a "hold" on the root segment, so that only one transaction can update session data at a time. When the session terminates, all data associated with the session is removed from the database. The physical characteristics of the database follow:

DBMS: IMS full function
Database name: YHCES*DD (the asterisk represents the entity ID)
IMS organization: HDAM (Hierarchic Direct Access Method)
Access method: OSAM (Overflow Sequential Access Method)
Block size: 8192

The database segments are described below:

1. *Session characteristics segment*

   Formal name: SCHARS
   Parent segment: none
   Description: This segment holds characteristics of a session. It contains fields, such as session status, date and time of last session activity, rule base name and version, and the last contract ID.
   Length: 60
   Key name: SESSID
   Key description: Session ID number
   Key start: 1
   Key length: 14
   Search fields: none
   Pointers: 3 physical child, 1 physical twin
   Frequency: 1 segment per session

2. *State segment*

   Formal Name: STATE
   Parent segment: SCHARS
   Description: Each segment holds a state name.
   Length: 30
   Key name: no key
   Key start: no key
   Key length: no key
   Search fields: none
   Pointers: 1 physical twin
   Frequency: 1 segment per state. There may be many states for a session.

3. *Facts segment*

   Formal name: FACTS
   Parent segment: SCHARS
   Description: Block of text containing up to 500 bytes. For a given session, all the chunks taken together form the entire text of all facts for the session.
   Length: 500
   Key name: no key
   Key start: no key
   Key length: no key
   Search fields: none
   Pointers: 1 physical twin
   Frequency: variable, depending upon total text size

4. *Contract segment*

   Formal name: CONTRSEG
   Parent segment: SCHARS
   Description: Block of contract text. There may be multiple occurrences of the contract segment for a given key value. All of these segments taken together make up the entire text of the contract identified by the key.
   Length: 1000
   Key name: CONTRACT
   Key description: Concatenation of session ID, contract name, and contract ID
   Key start: 1
   Key length: 11
   Search fields: none
   Pointers: 1 physical twin
   Frequency: Variable, depending upon contract length

2.1.2 Rules Database

The Rules database is a repository for rules of the SNS Negotiation Engine process type. A rule is a string of characters consisting of a left-hand side (LHS) and a right-hand side (RHS). The LHS of a rule is evaluated against facts, data that represents run-time events, and contract data, the application data input to the Negotiation Engine during a session. The RHS contains the actions to take if the LHS is actually satisfied. These actions can include state transitions, adding facts and invoking application tasks. The term "firing" a rule implies that the LHS of the rule has been satisfied and the RHS has been processed. Rules are instrumental in driving a negotiation session forward. A logical collection of rules is called a grouping. Several groupings make up a rule set, identified by a rule set name. A rule set belongs to a process type, i.e., Negotiation Engine. Batch utilities exist to load the rules into the Rules database and unload them from the Rules database. The Negotiation Engine only reads the rules. In either case, rules are accessed at a grouping level.

The rules are stored in the database as follows. All the rules for a grouping are placed back-to-back in a buffer. They are divided into approximately 4k-byte blocks. The first block is inserted as a root segment. Succeeding blocks (if any) are inserted as unkeyed dependent segments under the root segment. The physical characteristics of the database follow:

DBMS: IMS full function
Database name: YHDAR*DD (the asterisk represents the entity ID)
IMS organization: HDAM
Access method: OSAM
Block size: 8192

The database segments are described below:

1. *Rule text segment*

   Formal name: RULESEG
   Parent segment: none
   Description: This segment holds the first 3930 bytes of rule text within a grouping.
   Length: variable, up to 4000 bytes
   Key name: RULEKEY
   Key description: concatenation of grouping, rule set name, and process type
   Key start: 5
   Key length: 43
   Search fields: none
   Pointers: 1 physical child, 1 physical twin
   Frequency: 1 segment per grouping per rule set per process type

2. *Overflow segment*

   Formal name: OVFLSEG
   Parent segment: RULESEG
   Description: overflow of rule text from the root segment
   Length: variable, maximum 4000 bytes
   Key name: no key
   Key start: no key
   Key length: no key
   Search fields: none
   Pointers: 1 physical twin
   Frequency: as many segments as are required to accommodate the entire rule text within the grouping

2.2 SNS DB2 Database

The following sections describe the components of the SNS DB2 database.

2.2.1 Overview

SNS contains a DB2 database for TN selection, SNS exception handling, negotiation reference, and service request output. The DB2 database management system uses the relational database model. DB2 databases are logical groupings of database tables. For the SNS implementation, all of the tables are in one database. However, there are no physical restrictions for data access across databases once privileges have been granted. DB2 relational tables are collections of data attributes or columns. There are no physical links (or database pointers) between tables, like those in network or hierarchical databases.

The notion of a database or system "entity" is supported through separate copies of the database tables. Each entity has its own database with the corresponding copy of database tables. The table names are identical, but the database names each contain one character for entity and version (or release) identification. A deliverable DB2-based application must undergo a process called the Application Plan Bind. This process must be performed locally using the DB2 subsystem on the SNS Multiple Virtual System (MVS) host machine.[1] The site must initiate the bind to verify specific internal table and other DB2 Object Identifiers (OBIDs) in the Catalog of the DB2 subsystem running on the BCC MVS system that is hosting the SNS application.

During the application bind process, a specific SNS database is bound to a single application copy and entity. This creates an environment in which there is *one* SNS application, *one* application plan, and *one* database with the corresponding database tables *per* entity. One or more "applications" may use the same Information Management System Program Specification Block (IMS PSB) name, although each has its own Transaction Code. This environment allows many entities to run concurrently against the same copy of DB2 without restriction. Entity partitioning requires that the SNS data stored in each entity is orthogonal (normalized) to every other entity, to ensure referential integrity across entities.[2]

[1] It is not possible to copy an Application Plan from one DB2 subsystem to a different DB2 subsystem. Alternatively, Data Base Request Modules (DBRMs) are delivered and used as input to the bind process.

[2] This is an alternative way of saying that Centrex groups and telephone numbers should not be replicated across TNLIST entities, preventing duplicate assignments of telephone numbers (TNs).

2.2.1.1 Association Tables

The relationships between two or more tables may be represented several ways. SNS implements relationships by storing pieces of key information from at least one of the tables in the other tables. As relationships become more complex (including the possibility of many to many relationships), this type of structure introduces data dependencies among the tables. Additionally, the likelihood of maintaining a large amount of redundant data increases as entire rows must be duplicated to model a given "instantiation" of the relationship. Traditionally, this situation has been handled in Hierarchical and Network Model DBMS's through the use of database pointers contained within each row or record to logically link the tables or record types together.

An equivalent structure for database pointers then becomes necessary when relating two or more tables within a relational DBMS. One method of supporting this relationship is to store the key column(s) for the object tables[3] in an association table.

The association table is composed mostly of key columns (the set of all the columns serves as its Primary Key). The keys for each of the related tables are called Foreign Keys for the association table. Each row of this table represents a particular occurrence of a relationship between the two "base" tables that are being related. This method normalizes the data for the two base tables, thereby removing redundant rows of nonkey data. It is also called the First Normal Form.

Minimizing record key lengths is a physical database design objective, which reduces the number of index pages searched on a query and minimizes the number of index page splits and merges on database inserts and deletes. Since most of the columns in an association table are key columns, the row lengths for this type of table are usually short. The maximum number of records that can be stored on a data page is set by DB2 at 127, leaving a substantial amount of unusable space remaining on each data page of the table unless some other nonkey attributes of the relationship are stored in the same record.

Access to the relationship data in the association table is accomplished by querying one object table, the association table, and finally the other object table(s). Association tables facilitate the implementation of a many-to-many relationship between object tables. SNS uses several association tables. All relationships in SNS are between no more than two object (base) tables.

[3] Database tables that are part of the relationship.

2.2.1.2 Indexes

Two important uses for indexes are to improve database access time and to reduce the number of pages searched on a multi-record request. DB2 allows the database administrator (the user who has been granted DBADM privileges) the ability to create indexes on any of the columns in a database table (columns must be either "not null" or "not null with default," however). The characteristics specified for SNS DB2 indexes include:

- Column Name(s) – The columns that participate in the index.
- Storage allocations – STOGRP, PRIQTY, and SECQTY keywords denoting the DB2 Storage Group name and the Primary and Secondary quantities for space allocations and extents (in kilobytes), respectively.
- Key attributes – Physical and logical attributes for the key column(s) including the CLUSTER (physical record sequence) and UNIQUE (nonredundant key values) qualifiers.
- Index usage attributes – Keywords ERASE, CLOSE, SUBPAGES, and BUFFERPOOL representing automatic index ERASE when dropped, Index data set CLOSE after access, number of SUBPAGES per index page (influences concurrency), and BUFFERPOOL from which to select page buffers.

The indexes used in SNS make use of the above qualifiers to ensure that there are no duplicate rows of Primary Key data for a table, that Primary Keys are used in ordering the physical placement of records in a table, and that physical access and concurrency properties of the SNS DB2 database are optimized.

The columns participating in SNS DB2 UNIQUE indexes ensure that no redundant Primary Key information is entered into the database. Primary access methods are chosen to maximize the number of queries that can be resolved using indexes. Choosing index columns is also related to query resolution. Some queries may be resolved using only the data present in the index (INDEX ONLY searches performed automatically by DB2). The CLUSTER qualifier represents the physical storage sequence of records for a particular table. CLUSTER indexes need not be UNIQUE. The rationale behind the decisions involving the selection of candidate columns for indexes and their storage sequence is included with each database table design under the sub-heading "Primary Access Method and Physical Organization."

The Storage Group, Primary and Secondary space allocation and extents (STOGRP, PRIQTY and SECQTY, respectively) are site-specific parameters. STOGRP represents the host volume(s) for the SNS DB2 database, while PRIQTY and SECQTY contain the quantity of pages to allocate.

2.2.1.3 Database Physical Characteristics

Database Objects

DB2 provides an object called a STOGRP (storage group) to identify the volume(s) for table data. The size of the SNS DB2 database is affected by the number of services it supports. For further information on sizing the SNS DB2 database, see Section 4, "Database Sizing."[4]

Datasets for tables are allocated through the use of TABLESPACEs. A TABLESPACE is stored as a Virtual Sequential Access Method (VSAM) data set defined by DB2 to contain a table according to the input specified in the SQL CREATE TABLESPACE command. The SNS DB2 database contains one table per tablespace (simple tablespace), while all tablespaces (and therefore tables) reside within the same STOGRP.

2.2.1.4 Database Environment

Database calls for Release 2.0 of SNS are received from Data Access Routines (DARs) as a part of the negotiation and maintenance processing. SNS uses the Negotiation Engine, which facilitates the modularity of SNS processing and database activities.

[4] Initially, SNS supports only Centrex services. As other types of services are supported, the database volume may increase dramatically. Currently, the TNLIST database is small in comparison to other systems, such as SWITCH.

2.2.2 Negotiation Reference Database Tables

The SNS DB2 database provides reference information needed to support the service negotiation process. These tables are collectively referred to as the Negotiation Reference database tables and contain the following types of information:

- Feature data
- Network element data
- Feature package data
- Customer premises equipment (CPE) data.

The feature data tables provide information about the availability of features for specific switch types and generics and also indicate how each feature can be configured to provide service. Specific tables in this category are the Feature Group, Feature Catalog, and Feature Name tables. The network element data tables identify the individual network elements (switches) supported by the SNS system and describe their characteristics. These tables include the Switch Parameter, Network Element, and Switch Feature Exception tables. The feature package data tables provide information about BCC-tariffed packages of features for use with a specific product offering. The Feature Package table is the only table in this category. The CPE tables provide information about individual CPE and the features and configurations that can be supported by each. These tables consist of the CPE Reference table and the CPE Information table.

The following sections present a description of the individual tables that comprise the Negotiation Reference database tables, along with information about their key and nonkey columns, primary access methods, and physical organization. Figure 2-1 illustrates the structure of the Negotiation Reference database tables and their relationships to one another.

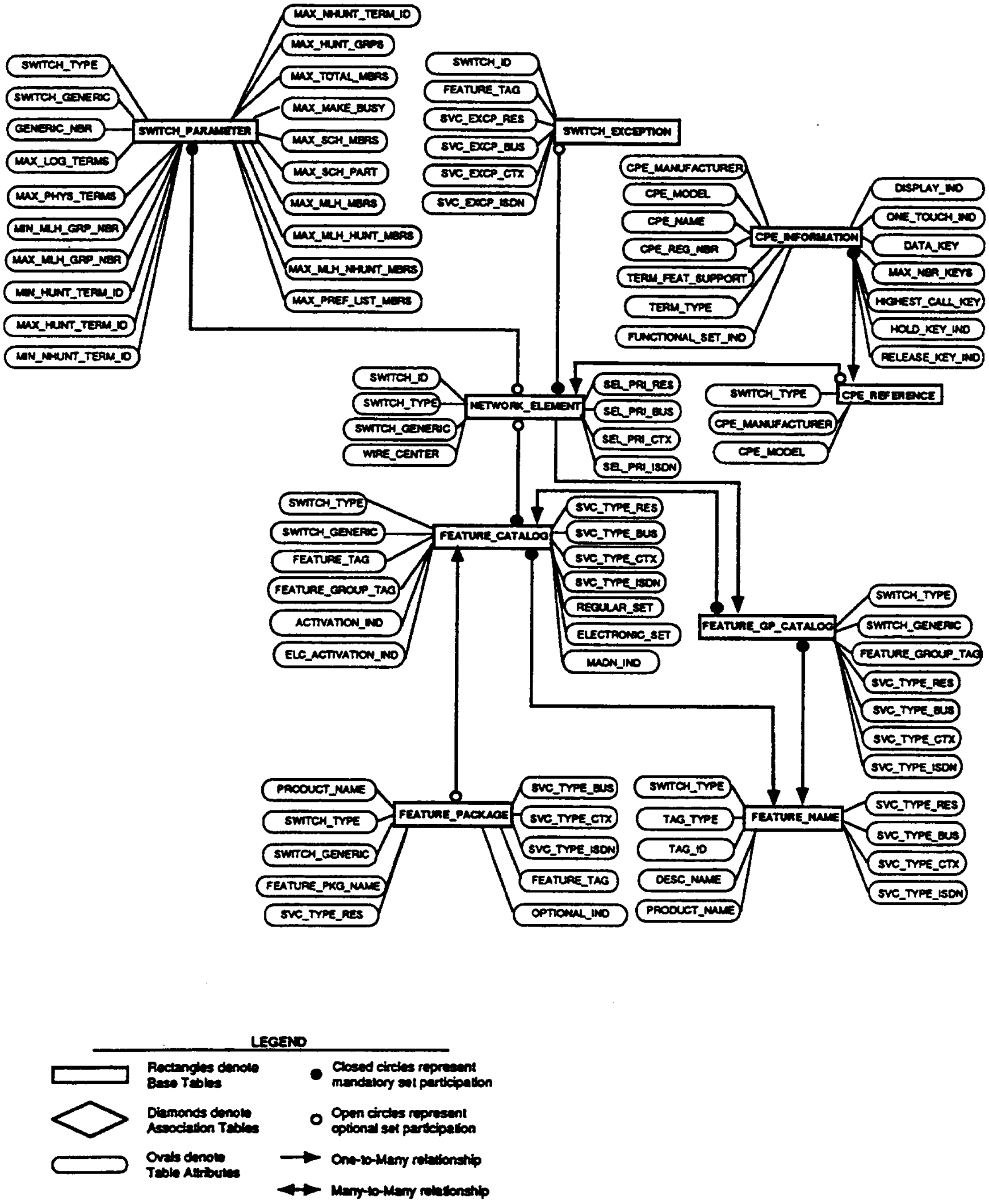

Figure 2-1. Negotiation Reference Database Tables

2.2.2.1 CPE Information Table

The CPE Information table contains information about specific customer premises equipment types. Each CPE is identified by its manufacturer and model number. Information in this table describes the CPE and indicates the types of services and features that the CPE can support. This table can be used during the service negotiation process to provide data about specific CPE to the negotiator and to ensure that the CPE to be used is compatible with the negotiated services.

2.2.2.1.1 Primary Access Method and Physical Organization

The CPE Information table is indexed on the CPE_MANUFACTURER and CPE_MODEL fields. This information uniquely identifies each individual CPE. The table is physically stored (clustered) in the ascending order of the key fields.

2.2.2.1.2 Expected Volume

There is one table row for each model of customer premises equipment in the table. Normally, only the more complex types of CPE (i.e., electronic sets) appear in this table. This could range from just a few sets, such as the Centrex electronic sets compatible with the DMS®-100 switch, to several hundred sets when SNS supports ISDN service.

2.2.2.1.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the CPE Information table. They include information on the columns, indexes,[5] and physical storage characteristics of the table.

DMS is a registered trademark of Northern Telecom Inc.

[5] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-1. CPE Information Table Design

Column Name	Column Size/ Column Type	Nulls	Description
CPE_MANUFACTURER*	CHAR(20)	Not null	Manufacturer name of the CPE
CPE_MODEL*	CHAR(20)	Not null	Model number of the CPE
CPE_NAME	CHAR(30)	Not null with default	Name of the CPE
CPE_REG_NBR	CHAR(17)	Not null with default	CPE registration number
TERM_FEAT_SUPPORT†	CHAR(14)	Not null with default	Standard code used to identify terminal characteristics and features supported
TERM_TYPE†	CHAR(1)	Not null with default	Standard code (A - E) that indicates terminal characteristic and bearer services supported (for 5ESS® only)
FUNCTIONAL_SET_IND†	CHAR(1)	Not null	Indicates if CPE is a functional set (Y) or a stimulus set (N).
DISPLAY_IND	CHAR(1)	Not null	Indicates if CPE can support features that require a display (Y or N).
ONE_TOUCH_IND	CHAR(1)	Not null	Indicates if CPE has speakerphone feature (Y or N).
DATA_KEY†	CHAR(3)	Not null with default	Identifies dedicated data key on CPE.

† Reserved for future use.

5ESS is a registered trademark of AT&T.

Table 2-1. CPE Information Table Design (cont'd)

Column Name	Column Size/ Column Type	Nulls	Description
MAX_NBR_KEYS	SMALL INT	Not null with default	Specifies number of keys on the CPE.
HIGHEST_CALL_KEY	SMALL INT	Not null with default	Specifies the maximum number of call appearances.
HOLD_KEY_IND	CHAR(1)	Not null	Identifies if CPE has a dedicated hold key (Y or N).
RELEASE_KEY_IND	CHAR(1)	Not null	Identifies if CPE has a dedicated release key (Y or N).

Table 2-2. CPE Information Table Characteristics

Table Name	YHNRTB10
Synonym(s)	CPE_INFORMATION
Database	YHSN*D2
Tablespace	YHNRSP10
Storage Group	YHSTOG**
Number of Columns	14
Row Length (bytes)	122
Row bytemap	[20][20][30][17][14][1][1][1][1][3][2][2][1][1]
Index Name	YHNRX10A
Index Type	CLUSTER, UNIQUE
Index Column(s)	CPE_MANUFACTURER, CPE_MODEL

2.2.2.2 CPE Reference Table

The CPE Reference table provides a mapping between each switch type (i.e., DMS or 5ESS) and its compatible customer premises equipment. This table can be accessed during the service negotiation process to identify the compatible CPE for a specific network element. Only the identification of the compatible CPE for a switch can be obtained from this table. Further information about an individual CPE or its characteristics is contained in the CPE Data table.

2.2.2.2.1 Primary Access Method and Physical Organization

The CPE Reference Table is indexed on the SWITCH_TYPE field, which provides access to all CPE names that the switch type supports. Since multiple CPE are compatible with a given switch type, the key field is not unique. The table is physically stored (clustered) in the ascending order of the key field to enable all CPE names for a switch type to be stored in adjacent rows. This allows for the most efficient access to the CPE reference data.

2.2.2.2.2 Expected Volume

For each switch type supported by SNS, one table row is required per specified compatible CPE.

2.2.2.2.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the CPE Reference table. They include information on the columns, indexes,[6] and physical storage characteristics of the table.

[6] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-3. CPE Reference Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_TYPE*	CHAR(15)	Not null	Switch type (i.e., 5ESS, DMS100)
CPE_MANUFACTURER	CHAR(20)	Not null	Manufacturer name of the CPE
CPE_MODEL	CHAR(20)	Not null	Model number of the CPE

Table 2-4. CPE Reference Table Characteristics

Table Name	YHNRTB15
Synonym(s)	CPE_REFERENCE
Database	YHSN*D2
Tablespace	YHNRSP15
Storage Group	YHSTOG**
Number of Columns	3
Row Length (bytes)	63
Row bytemap	[15][20][20]
Index Name	YHNRX15A
Index Type	CLUSTER, NON-UNIQUE
Index Column(s)	SWITCH_TYPE

2.2.2.3 Feature Catalog Table

The SNS Feature Catalog table defines the service features that exist for a specific switch type and generic. Initially the data contained in this table will be loaded at Bellcore and shipped to the BCCs. BCCs can use the catalog to define which specific service features they will offer.

The information in this table can be used during the service negotiation process to provide the negotiator with data on available feature offerings. Access to the table is by switch type and switch generic. Any exception to this catalog (about the availability of a feature at a specific switch) can be made by establishing an entry in the Switch Feature Exception table.

Each feature in the catalog is represented by a tag identifier. The tag identifier is the same tag that is referenced by the SNS negotiation contracts to represent the feature. The tag identifier selected to represent each feature is actually a COMMON LANGUAGE® tag used across multiple Bellcore systems. The associated long name description of the feature represented by each tag is contained in the Feature Name table.

In addition to identifying features, the catalog also provides information on how each feature can be configured for the switch type/generic. For example, the catalog indicates if the feature is buttonable, dial access, or both. It also indicates the service types (e.g., residence, Centrex, etc.) for which the feature is available, as well as the associated feature group. The feature group allows individual features to be classified into groups of related features.

2.2.2.3.1 Primary Access Method and Physical Organization

The Feature Catalog contains two indexes. The first index type is comprised of the SWITCH_TYPE, SWITCH_GENERIC, and FEATURE_GROUP_TAG data fields. This key combination provides access to all features in a specific feature group for a given switch type and generic. Since multiple features can exist for one feature group, this key combination is non-unique. The table is physically stored (clustered) in the ascending order of these key fields.

The second index type is comprised of the SWITCH_TYPE, SWITCH_GENERIC, and FEATURE_TAG data fields. This key combination

COMMON LANGUAGE is a registered trademark and CLCI, CLEI, CLFI, and CLLI are trademarks of Bellcore.

provides access to information on an individual feature basis. Since only one of each feature type can exist at a switch type and generic, the key combination is unique.

2.2.2.3.2 Expected Volume

One table row exists for each service feature that is supported as part of each switch type and generic. There are currently about 200 features defined per switch type/generic, and about eight switch type/generic combinations supported by SNS.

2.2.2.3.3 Design Considerations

The same feature tag can exist for many switch type/generic combinations. However, the data associated with each feature (i.e., supported service types) may need to be set differently across the various switch generics. As a result, the data is not solely dependent on the feature tag, but on a switch type/generic/feature tag basis. This fact has been reflected in the Feature Catalog table design.

2.2.2.3.4 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Feature Catalog table. They include information on the columns, indexes,[7] and physical storage characteristics of the table.

[7] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-5. Feature Catalog Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_TYPE*	CHAR(15)	Not null	Switch type (i.e., 5ESS, DMS100)
SWITCH_GENERIC*	CHAR(10)	Not null	Switch generic software release level
FEATURE_GROUP_TAG*	CHAR(8)	Not null	Feature Group identifies the group the feature belongs to.
FEATURE_TAG*	CHAR(8)	Not null	Feature tag identifies a feature supported by the switch type/generic.
ACTIVATION_IND	CHAR(1)	Not null	Feature activation indicator indicates if the customer's feature is buttonable (B), dial accessed (D), either (E) or does not apply (N).
ELC_ACTIVATION_IND	CHAR(1)	Not null	Electronic set activation indicator that indicates if the feature is buttonable (B), dial accessed (D), either (E) or does not apply (N).
SVC_TYPE_RES	CHAR(1)	Not null	Identifies if the feature supports residence service (Y or N).
SVC_TYPE_BUS	CHAR(1)	Not null	Identifies if the feature supports business service (Y or N).
SVC_TYPE_CTX	CHAR(1)	Not null	Identifies if the feature supports Centrex service (Y or N).
SVC_TYPE_ISDN†	CHAR(1)	Not null	Identifies if the feature supports ISDN service (Y or N).

† Reserved for future use.

Table 2-5. Feature Catalog Table Design (cont'd)

Column Name	Column Size/ Column Type	Nulls	Description
REGULAR_SET	CHAR(1)	Not null	Identifies if the feature is available (A) on a station set or unavailable (U) on a station set.
ELECTRONIC_SET	CHAR(1)	Not null	Identifies if the feature is a electronic business set feature (S), electronic business set subset feature (D), can only be assigned to the primary number (P), or unavailable as an electronic business set feature (U).
MADN_IND†	CHAR(1)	Not null	Indicates if the feature is a MADN feature (Y or N).

† Reserved for future use.

Table 2-6. Feature Catalog Table Characteristics

Table Name	YHNRTB30
Synonym(s)	FEATURE_CATALOG
Database	YHSN*D2
Tablespace	YHNRSP30
Storage Group	YHSTOG**
Number of Columns	13
Row Length (bytes)	58
Row bytemap	[15][10][8][8][1][1][1][1][1][1][1][1][1]
Index Name	YHNRX30A
Index Type	CLUSTER, NON-UNIQUE
Index Column(s)	SWITCH_TYPE SWITCH_GENERIC FEATURE_GROUP_TAG
Index Name	YHNRX30B
Index Type	NON-CLUSTERING, UNIQUE
Index Column(s)	SWITCH_TYPE SWITCH_GENERIC FEATURE_TAG

2.2.2.4 Feature Group Catalog Table

Feature offerings can be categorized into sets of similar features. For example, the features Call Forwarding-Variable, Call Forwarding-Don't Answer, and Call Forwarding-Busy Line are all various types of call forwarding. These sets of similar features are called Feature Groups within SNS. The Feature Group Catalog table defines the feature groups that exist for each specific switch type and generic.

During the service negotiation process, negotiators can obtain data on all available service features for the serving switch. However, to reduce the amount of data to be transmitted and displayed, the feature groups can optionally be used. This is accomplished by first obtaining the names of the supported feature groups, and then selecting the feature groups of interest. Detailed information can then be obtained on the service features available in the selected groups only, thus limiting the amount of feature data involved.

Each feature group in the catalog is represented by a tag identifier. The feature group tag is the same tag that is referenced by SNS contracts. The associated long name description of the feature group represented by each tag is obtained from the Feature Name table. Initially the data contained in this table will be loaded at Bellcore and shipped to the BCCs.

2.2.2.4.1 Primary Access Method and Physical Organization

The Feature Group Catalog table is indexed on the SWITCH_TYPE and SWITCH_GENERIC data fields. This key combination is non-unique, which will allow the retrieval of all feature groups for a switch type/generic. The table is physically stored (clustered) in the ascending order of the key fields.

2.2.2.4.2 Expected Volume

There is one table row for each feature group defined for each switch type and generic. It is expected that about eight switch type/generic combinations will initially exist within SNS, each with about two dozen feature groups.

2.2.2.4.3 Design Considerations

The same feature group can exist for many switch type/generic combinations. However, the data associated with each feature group (i.e., supported service types) may need to be set differently across the various switch generics. As a result, the data is not solely dependent on the feature group tag, but on a switch type/generic/feature group basis. This fact has been reflected in the Feature Group Catalog table design.

2.2.2.4.4 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Feature Group Catalog table. They include information on the columns, indexes,[8] and physical storage characteristics of the table.

Table 2-7. Feature Group Catalog Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_TYPE*	CHAR(15)	Not null	Switch type (i.e., 5ESS, DMS100)
SWITCH_GENERIC*	CHAR(10)	Not null	Switch generic software release level
FEATURE_GROUP_TAG	CHAR(8)	Not null	Feature group identifier
SVC_TYPE_RES	CHAR(1)	Not null	Identifies if the feature group supports residence service (Y or N).
SVC_TYPE_BUS	CHAR(1)	Not null	Identifies if the feature group supports business service (Y or N).
SVC_TYPE_CTX	CHAR(1)	Not null	Identifies if the feature group supports Centrex service (Y or N).
SVC_TYPE_ISDN†	CHAR(1)	Not null	Identifies if the feature group supports ISDN service (Y or N).

† Reserved for future use.

[8] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-8. Feature Group Catalog Table Characteristics

Table Name	YHNRTB35
Synonym(s)	FEATURE_GP_CATALOG
Database	YHSN*D2
Tablespace	YHNRSP35
Storage Group	YHSTOG**
Number of Columns	7
Row Length (bytes)	45
Row bytemap	[15][10][8][1][1][1][1]
Index Name	YHNRX35A
Index Type	CLUSTER, NON-UNIQUE
Index Column(s)	SWITCH_TYPE SWITCH_GENERIC

2.2.2.5 Feature Name Table

The SNS service negotiation contracts use tags to represent each individual feature and feature group. The Feature Name table provides the association between the tag and its long descriptive name. The descriptive names can be modified through the use of maintenance contracts. In addition to modifying the descriptive name for a tag, it is also possible to establish additional descriptive names on a service type or product basis. This is meant to support cases in which the feature descriptive name varies depending on the service type or product for which it is offered.

During the service negotiation process when data is requested on available features, this table is accessed to determine the associated feature descriptive name. The descriptive name is included as part of the SNS response contract so that meaningful feature names are available for display at the user terminal.

2.2.2.5.1 Primary Access Method and Physical Organization

The Feature Name table is indexed on the fields SWITCH_TYPE, TAG_TYPE, TAG_ID, SVC_TYPE_RES, SVC_TYPE_BUS, SVC_TYPE_CTX, SVC_TYPE_ISDN, and PRODUCT_NAME. Since different descriptive names are sometimes required for a feature on a service type or product name basis, these key columns together provide a unique identification for each table entry. The table is physically stored (clustered) in the ascending order of the key fields.

2.2.2.5.2 Expected Volume

There is at least one table row for each feature or feature group associated with a supported switch type. SNS initially supports two switch types, with several hundred feature/feature group descriptive names in each. Additional descriptive names can be established as needed by the BCCs for each feature. However, it is anticipated that only a few features will require multiple descriptive names.

2.2.2.5.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Feature Name table. They include information on the columns, indexes,[9] and physical storage characteristics of the table.

[9] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-9. Feature Name Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_TYPE*	CHAR(15)	Not null	Switch type (i.e., 5ESS, DMS100)
TAG_TYPE*	CHAR(1)	Not null	Indicates if the associated tag is for an individual feature (F) or a feature group (G).
TAG_ID*	CHAR(8)	Not null	The tag associated with an individual feature or a feature group
DESC_NAME	CHAR(50)	Not null	The long name of the associated feature or feature group
SVC_TYPE_RES*	CHAR(1)	Not null with default	Indicates if the associated feature name description is for residence service (Y or N).
SVC_TYPE_BUS*	CHAR(1)	Not null with default	Indicates if the associated feature name description is for business service (Y or N).
SVC_TYPE_CTX*	CHAR(1)	Not null with default	Indicates if the associated feature name description is for Centrex service (Y or N).
SVC_TYPE_ISDN*†	CHAR(1)	Not null with default	Indicates if the associated feature name description is for ISDN service (Y or N).
PRODUCT_NAME*	CHAR(30)	Not null with default	Identifies the associated feature name as associated with a specific product.

† Reserved for future use.

Table 2-10. Feature Name Table Characteristics

Table Name	YHNRTB40
Synonym(s)	FEATURE_NAME
Database	YHSN*D2
Tablespace	YHNRSP40
Storage Group	YHSTOG**
Number of Columns	9
Row Length (bytes)	116
Row bytemap	[15][1][8][50][1][1][1][1][30]
Index Name	YHNRX40A
Index Type	CLUSTER, UNIQUE
Index Column(s)	SWITCH_TYPE, TAG_TYPE, TAG_ID, SVC_TYPE_RES, SVC_TYPE_BUS, SVC_TYPE_CTX, SVC_TYPE_ISDN, PRODUCT_NAME

2.2.2.6 Feature Package Table

The Feature Package table defines general feature arrangements that are established and marketed by a BCC for use with a specific product offering. Each feature package is defined on a switch type and generic basis.

During the negotiation process, these general feature arrangements established by the BCC can be accessed from the Feature Package table. The feature packages correspond to the product offering selected by the customer. They are compatible with the switch type and generic and can be used as a basis for the negotiation.

2.2.2.6.1 Primary Access Method and Physical Organization

The Feature Package table is indexed on the PRODUCT_NAME, SWITCH_TYPE, SWITCH_GENERIC, and FEATURE_PKG_NAME data fields. This allows easy access to all feature package names for a specific product/switch type/generic combination, as well as to all individual features that comprise a specific feature package. Since many features can exist in a feature package, the key combination is non-unique. The table is physically stored (clustered) in the ascending order of the key fields to enable all features for a product to be stored in adjacent rows. This allows for the most efficient access to the Feature Package table.

2.2.2.6.2 Expected Volume

There is one table row for each feature in each BCC-defined feature package. It is expected that about a dozen features will comprise each package, with packages being defined on a product/switch type/generic basis. Use of this table by a BCC is entirely optional, in this way packages may be defined for only a small percentage of the products actually offered. It is estimated, however, that there may be several dozen product offerings with several packages defined for each.

2.2.2.6.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Feature Package table. They include information on the columns, indexes,[10] and physical storage characteristics of the table.

[10] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-11. Feature Package Table Design

Column Name	Column Size/ Column Type	Nulls	Description
PRODUCT_NAME*	CHAR(30)	Not null	Product name identifies the product being offered.
SWITCH_TYPE*	CHAR(15)	Not null	Switch type identifies the type of switch (e.g., 5ESS, DMS100).
SWITCH_GENERIC*	CHAR(10)	Not null	Switch generic indicates the software release level that has been applied on the switch.
SVC_TYPE_RES	CHAR(1)	Not null	Identifies if the feature package supports residence service.
SVC_TYPE_BUS	CHAR(1)	Not null	Identifies if the feature package supports business service.
SVC_TYPE_CTX	CHAR(1)	Not null	Identifies if the feature package supports Centrex service.
SVC_TYPE_ISDN†	CHAR(1)	Not null	Identifies if the feature package supports ISDN service.
FEATURE_PKG_NAME*	CHAR(15)	Not null	Feature Package name
FEATURE_TAG	CHAR(8)	Not null	Feature tag identifies each feature being offered through the feature package.
OPTIONAL_IND	CHAR(1)	Not null	Indicates if the feature for the feature package is offered as an option (Y or N).

† Reserved for future use.

Table 2-12. Feature Package Table Characteristics

Table Name	YHNRTB45
Synonym(s)	FEATURE_PACKAGE
Database	YHSN*D2
Tablespace	YHNRSP45
Storage Group	YHSTOG**
Number of Columns	10
Row Length (bytes)	91
Row bytemap	[30][15][10][1][1][1][1][15][8][1]
Index Name	YHNRX45A
Index Type	CLUSTER, NON-UNIQUE
Index Column(s)	PRODUCT_NAME SWITCH_TYPE SWITCH_GENERIC FEATURE_PKG_NAME

2.2.2.7 Network Element Table

This table contains descriptive information about each specific switch supported by SNS. This includes the switch type, switch generic, and associated wire center. Also included are switch selection priority values, which will not be used until a future SNS release.

Once the serving switch for a customer has been determined during the negotiation process, this table can be accessed. The switch type and generic are then determined and can be used to obtain supported service features from the SNS feature catalog. These features are defined on a switch type and generic basis. The switch type and generic are also used to access the Switch Parameter table, which contains various parameters needed to establish and verify negotiated services.

2.2.2.7.1 Primary Access Method and Physical Organization

The Network Element table is indexed on the SWITCH_ID field, which provides a unique identifier for each row. This allows access to data on an individual switch basis. The table is physically stored (clustered) in the ascending order of this key field.

2.2.2.7.2 Expected Volume

There is one table row for each switch supported by SNS.

2.2.2.7.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Network Element table. They include information on the columns, indexes,[11] and physical storage characteristics of the table.

[11] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-13. Network Element Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_ID*	CHAR(11)	Not null	Network element ID
SWITCH_TYPE	CHAR(15)	Not null	Network element type (i.e., 5ESS, DMS100)
SWITCH_GENERIC	CHAR(10)	Not null	Network element software release level
WIRE_CENTER	CHAR(8)	Not null	Wire center for the switch
SEL_PRI_RES†	SMALL INT	Not null with default	Identifies the selection priority of the switch for residence service.
SEL_PRI_BUS†	SMALL INT	Not null with default	Identifies the selection priority of the switch for business service.
SEL_PRI_CTX†	SMALL INT	Not null with default	Identifies the selection priority of the switch for Centrex service.
SEL_PRI_ISDN†	SMALL INT	Not null with default	Identifies the selection priority of the switch for ISDN service.

† Reserved for future use.

Table 2-14. Network Element Table Characteristics

Table Name	YHNRTB50
Synonym(s)	NETWORK_ELEMENT
Database	YHSN*D2
Tablespace	YHNRSP50
Storage Group	YHSTOG**
Number of Columns	8
Row Length (bytes)	60
Row bytemap	[11][15][10][8][2][2][2][2]
Index Name	YHNRX50A
Index Type	CLUSTER, UNIQUE
Index Column(s)	SWITCH_ID

2.2.2.8 Switch Feature Exception Table

The SNS feature catalog defines which service features are associated with network elements on a switch type and generic basis. It also defines whether the features are to be made generally available to customers by the BCC or not. Exceptions to this catalog information can be established on an individual switch basis. The function of the Switch Feature Exception table is to maintain this service feature exception data.

The table contains an entry for each switch ID/feature combination in which the feature availability differs from that specified in the catalog. Features not offered at a specific switch, even though they are generally available at other similar switches throughout the BCC, are recorded here. Similarly, features available at a specific switch are recorded here when the catalog specifies that they are not generally offered in the BCC. Feature exceptions at a switch can be established for a single service type (i.e., residence or business), all SNS service types, or any combination of service types. This information is used during the service negotiation process to validate that negotiated features can be supported at the serving switch.

2.2.2.8.1 Primary Access Method and Physical Organization

The Switch Feature Exception table is indexed on the SWITCH_ID field, which provides access to all features for the switch that differ from the SNS feature catalog. Since several feature exceptions may exist for a single switch, the key field is not unique. The table is physically stored (clustered) in the ascending order of the key field to enable all feature exceptions for a switch to be stored in adjacent rows. This allows for most efficient access to the feature exception data.

2.2.2.8.2 Expected Volume

There is one table row for each feature exception at each switch. It is currently anticipated that each switch will have only one or two feature exceptions.

2.2.2.8.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Switch Feature Exception table. They include information on the columns, indexes,[12] and physical storage characteristics of the table.

[12] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-15. Switch Feature Exception Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_ID*	CHAR(11)	Not null	Network element ID
FEATURE_TAG	CHAR(8)	Not null	Tag representing a specific service feature in the SNS feature catalog.
SVC_EXCP_RES	CHAR(1)	Not null	Indicates whether the feature is available by exception (Y) or not available by exception (N) for residence service. Value ' ' indicates no exception exists for this service type.
SVC_EXCP_BUS	CHAR(1)	Not null	Indicates whether the feature is available by exception (Y) or not available by exception (N) for business service. Value ' ' indicates no exception exists for this service type.
SVC_EXCP_CTX	CHAR(1)	Not null	Indicates whether the feature is available by exception (Y) or not available by exception (N) for Centrex service. Value service type.
SVC_EXCP_ISDN†	CHAR(1)	Not null	Indicates whether the feature is available by exception (Y) or not available by exception (N) for ISDN service. Value ' service type.

† Reserved for future use.

Table 2-16. Switch Feature Exception Table Characteristics

Table Name	YHNRTB55
Synonym(s)	SWITCH_EXCEPTION
Database	YHSN*D2
Tablespace	YHNRSP55
Storage Group	YHSTOG**
Number of Columns	6
Row Length (bytes)	31
Row bytemap	[11][8][1][1][1][1]
Index Name	YHNRX55A
Index Type	CLUSTER, NON-UNIQUE
Index Column(s)	SWITCH_ID

2.2.2.9 Switch Parameter Table

The Switch Parameter table contains information on various limits that exist on a switch type/generic basis. Currently, this includes the maximum number of supported physical and logical terminals for ISDN service, as well as a variety of parameters to support hunt group administration (both series completion and multi-line hunt). Not all of this information is currently used by SNS.

The data in this table can be used to ensure that the configured service does not exceed the capabilities of the switch (e.g., that the number of members requested in a series completion hunt group is not more than the switch can support) and is available to the service negotiator as part of the network element information. Not all of the table parameter columns apply for each switch type/generic. Columns that do not apply contain a default value.

2.2.2.9.1 Primary Access Method and Physical Organization

The Switch Parameter table is indexed on the SWITCH_TYPE and SWITCH_GENERIC fields. This provides a unique key combination for table access. The table is physically stored (clustered) in the ascending order of the key fields.

2.2.2.9.2 Expected Volume

There is one table row for each switch type and generic. Initially, about eight switch type/generic combinations will be supported in SNS.

2.2.2.9.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Switch Parameter table. They include information on the columns, indexes,[13] and physical storage characteristics of the table.

[13] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-17. Switch Parameter Table Design (1 of 3)

Column Name	Column Size/ Column Type	Nulls	Description
SWITCH_TYPE*	CHAR(15)	Not null	Switch type (i.e., 5ESS, DMS100)
SWITCH_GENERIC*	CHAR(10)	Not null	Switch software release level
GENERIC_NBR	SMALL INT	Not null	Internal code to enable switch generics to be accessed in sequential order.
MAX_LOG_TERMS†	SMALL INT	Not null	Maximum number of bearer services supported per line.
MAX_PHYS_TERMS†	SMALL INT	Not null	Maximum number of CPE devices supported per line.
MIN_MLH_GRP_NBR†	SMALL INT	Not null with default	Minimum multi-line hunt group ID number supported.
MAX_MLH_GRP_NBR†	SMALL INT	Not null with default	Maximum multi-line hunt group ID number supported.
MIN_HUNT_TERM_ID†	SMALL INT	Not null with default	Smallest multi-line hunt terminal ID number allowed.
MAX_HUNT_TERM_ID†	SMALL INT	Not null with default	Largest multi-line hunt terminal ID number allowed.
MIN_NHUNT_TERM_ID†	SMALL INT	Not null with default	Smallest multi-line non-hunt terminal ID number allowed.
MAX_NHUNT_TERM_ID†	SMALL INT	Not null with default	Largest multi-line non-hunt terminal ID number allowed.

† Reserved for future use.

Table 2-17. Switch Parameter Table Design (2 of 3)

Column Name	Column Size/ Column Type	Nulls	Description
MAX_HUNT_GRPS†	SMALL INT	Not null with default	Maximum number of hunt groups supported by the switch.
MAX_TOTAL_MBRS†	SMALL INT	Not null with default	Maximum total number of hunt group members supported by the switch.
MAX_MAKE_BUSY†	SMALL INT	Not null with default	Maximum number of hunt group members to which make-busy feature may be extended.
MAX_SCH_MBRS	SMALL INT	Not null with default	Maximum number of members allowed in one series completion hunt group.
MAX_SCH_PART	SMALL INT	Not null with default	Maximum number of series completion hunt groups in which a single TN may participate.
MAX_MLH_MBRS†	SMALL INT	Not null with default	Maximum number of members allowed in one multi-line hunt group.

† Reserved for future use.

Table 2-17. Switch Parameter Table Design (3 of 3)

Column Name	Column Size/ Column Type	Nulls	Description
MAX_MLH_HUNT_MBRS†	SMALL INT	Not null with default	Maximum number of hunt members allowed in one multi-line hunt group.
MAX_MLH_NHUNT_MBRS†	SMALL INT	Not null with default	Maximum number of non-hunt members allowed in one multi-line hunt group.
MAX_PREF_LIST_MBRS†	SMALL INT	Not null with default	Maximum number of members allowed in a preferential hunt list.

† Reserved for future use.

Table 2-18. Switch Parameter Table Characteristics

Table Name	YHNRTB60
Synonym(s)	SWITCH_PARAMETER
Database	YHSN*D2
Tablespace	YHNRSP60
Storage Group	YHSTOG**
Number of Columns	20
Row Length (bytes)	69
Row bytemap	[15] [10] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2] [2]
Index Name	YHNRX60A
Index Type	CLUSTER, UNIQUE
Index Column(s)	SWITCH_TYPE, SWITCH_GENERIC

2.2.3 SNS Exception Handler Database Table

The SNS Exception Handler (SEH) DB2 table relates and manages all of the error, warning, and completion messages and monitor notices used in SNS processing. The data in the SEH table contains a one-to-many relationship between error messages and logical callers. For example, a given error message may be appropriate for both Negotiation and Maintenance activity, but require different SNS processing for each. If the message is generated from Negotiation activity, it may be appropriate to terminate processing, while from Maintenance activity a return of control to the calling routine may be the proper action. This information is implemented in the SEH DB2 table by messages via an exception code and resulting SNS action code.

For service negotiation and maintenance activity, an SEH message is returned as a part of an SNS response contract. TNLIST uses SEH to generate monitor notices to a destination defined in the OH Destination table (usually a line printer), when TN selection activity depletes the number of available TNs below a BCC-defined threshold. For more information on the SNS Exception Handler, please refer to the section entitled "SNS Exception Handler" in the *Service Negotiation Support (SNS) System Administration Guide.*

The exception, warning, monitor notice, and completion messages issued by SNS are centrally located in the Exception Message DB2 table in the SNS DB2 database. The exception table follows the SNS DB2 naming conventions (see Section 3, "Database Naming Conventions," in this document), using the third and fourth characters to distinguish it from the TNLIST tables. The Exception Handler table name is YHENTB10. Figure 2-2 illustrates the structure of the Exception Handler database table. The following sections describe the database design for the Exception Handler table.

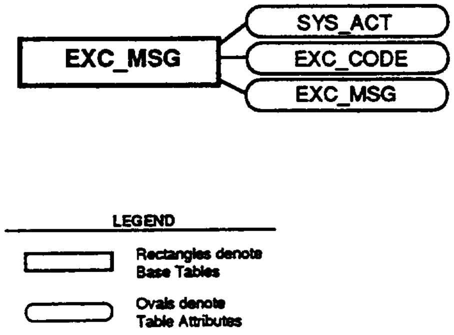

Figure 2-2. Exception Handler Database Table

2.2.3.1 SNS Exception Message Table

The Exception Message (EXC_MSG) table stores information used by the exception handler routines for displaying exception, warning, monitor notice, and status messages for all components of the SNS application. The exception handler routines provide an exception code that maps to an error message. Error messages are narratives used by SNS in all forms of processing and are retrieved by their individual exception codes.

2.2.3.1.1 Primary Access Method and Physical Organization

This table is indexed on the five-character exception code, which provides a unique identifier for each row. Through the use of this field, all necessary exception processing data is returned within one request. The table is physically stored (clustered) in the ascending order of the exception code.

2.2.3.1.2 Nonkey Data

The exception message field holds a two-hundred-character narrative containing the message text associated with the input exception code. Message text may contain space for variable parameters that are supplied at the time the exception handler is called. These variables are denoted by the C language data type formats used in the standard C library string manipulation functions (i.e., integers may be flagged in the message text with "%d", strings with "%s").

The SYS_ACT field contains a two-character code that instructs SNS on the proper action to take as a result of the input exception code. System actions currently include:

- Terminate processing
- Terminate processing and roll back any database changes
- Return control to the calling routine
- Return control to the calling routine and roll back any database changes.

For further information on the exception handler processing, see the *Service Negotiation Support (SNS) System Administration Guide.*

2.2.3.1.3 Expected Volume

There is one record for each SNS exception message, warning, or notifier as documented in the *Service Negotiation Support (SNS) System Administration Guide.*

2.2.3.1.4 Design Considerations

The capability to substitute parameters in the exception messages allows both message length and quantity to be minimized.

2.2.3.1.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Exception Message table. They include information on the columns, indexes,[14] and physical storage characteristics of the table.

[14] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-19. Exception Message Table Design

Column Name	Column Size/ Column Type	Nulls	Description
EXC_CODE*	CHAR(5)	Not null	Exception code
SYS_ACT	CHAR(2)	Not null with default	System action (i.e., terminate and rollback)
EXC_MSG	CHAR(200)	Not null	Exception message narrative

Table 2-20. Exception Message Table Characteristics

Table Name	YHENTB10
Synonym(s)	EXC_MSG
Database	YHSN*D2
Tablespace	YHENSP10
Storage Group	YHSTOG**
Number of Columns	3
Row Length (bytes)	215
Row bytemap	[5][2][200]
Index Name	YHENX10A
Index Type	CLUSTER, UNIQUE
Index Column(s)	EXC_CODE

2.2.4 Service Request Output Database Tables

The Service Request Output (SRO) database tables relate and manage all data related to the SNS service requests and their related service orders that are active in SNS. They are used primarily by the SRO component of SNS, which is designed to use the recapped view of negotiated service in a service specification (SVCSPC) contract to produce an output message in a configurable format. The current means of invoking SRO is to send an internally-defined contract to the SRO program using the Test Message Generator (TMG) of the Provisioning PLATFORM (PPLAT).

When SRO receives this contract, it retrieves a previously-issued SVCSPC contract from its database, processes it, and stores the resulting output message in its database. The Order Image and Order Status tables are populated by SNS with incoming SVCSPC contracts and their related status information. SRO uses these tables to retrieve SVCSPC contract information and to store output messages. Later, when an order completion (ORDCMP) contract is received by SNS, entries in the Order Image and Order Status tables related to the completed order are deleted. In addition, two reference tables, the SRO Control and Tag Type tables, are used to configure SRO to produce a particular type of output message (for example, they currently support the production of SNS FCIF). These tables are pre-populated by Bellcore.

NOTE

In order for SRO to be used to produce output messages, a complete view of the negotiated service must be given in the SVCSPC contract. SNS Release 2.0 supports only an abridged version of this contract.

Figure 2-3 illustrates the structure of the Service Request Output database tables, including the relationship between the tables. The following sections describe the tables involved in the SRO process.

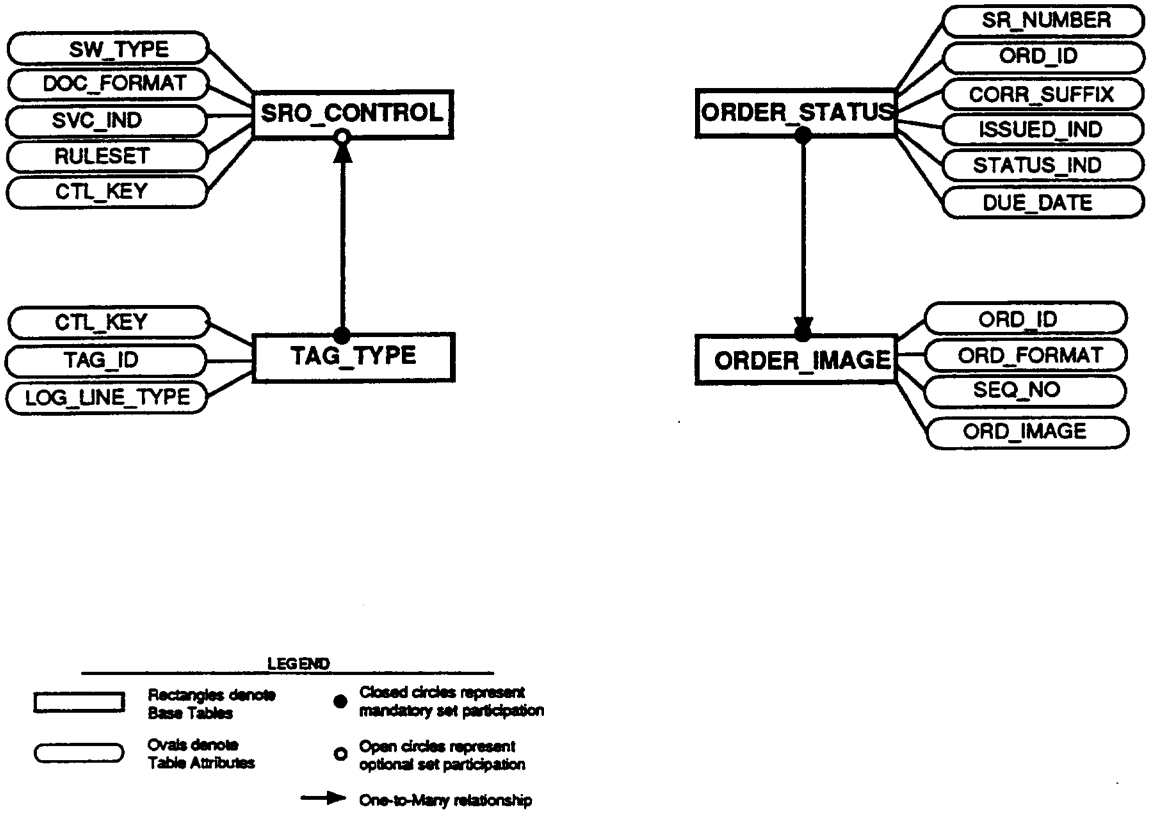

Figure 2-3. Service Request Output Database Tables

2.2.4.1 Order Image Table

When the service negotiation process is completed for a service request, the ULBB or another PLBB sends the summarized negotiated information in a SVCSPC contract to SNS. The contract data is organized into sections that reflect the way in which the negotiated service is partitioned into service orders (since a single service request may involve several orders). The information includes not only the circuits, telephone numbers, and hunt groups that appear on each order, but also the associated listing, billing, remarks, and identification data. This data, on a per-order basis, is then stored within SNS in the Order Image table. This table is the source from which the SRO process retrieves data to generate the corresponding output messages.

Output generated by the SRO process is also stored within this table. The output message format currently supported by SRO is a Bellcore-defined format known as SNS FCIF. These output messages are stored for two main reasons: first to enable SNS to easily produce additional copies if so requested by the BCC, and second to support correction pass generation of output messages. Since SNS does not track the individual changes requested as a result of a service correction, SRO regenerates the SNS FCIF message for all orders associated with an individual service request. However, orders for which the SNS FCIF has not changed must not have an SNS FCIF message reissued (presuming that the SNS FCIF message is issued to a destination for further processing). SRO can make this determination by comparing the information in the Order Image table with the regenerated SNS FCIF message. If the two are different, the Order Image table is updated with the latest version of the SNS FCIF. If the two are the same, the order's SNS FCIF output was not affected by the service corrections, and the regenerated SNS FCIF message is discarded.

Both the SVCSPC contract data and the SNS FCIF may exist for an order in this table simultaneously. If more order image data for a particular format exists than will fit into a single table row, additional table rows are used to hold the overflow information. A sequence number column is used to keep the overflow rows in their correct order.

2.2.4.1.1 Primary Access Method and Physical Organization

The Order Image table is indexed on the ORD_ID, ORD_FORMAT, and SEQ_NO fields. The ORD_ID field enables access to order data on an individual order basis. The ORD_FORMAT field is used to distinguish between the two types of order image data: negotiated order (SVCSPC contract) data and generated SNS FCIF message output. The SEQ_NO field is used to sequence an order image that overflows to multiple table rows. The key combination is unique, and the table is physically stored (clustered) in the ascending order of the three key fields.

2.2.4.1.2 Expected Volume

It is anticipated that two to three Order Image table rows will exist for each order processed by SRO. The average size for each row is currently unknown, but depends directly upon the amount of SVCSPC contract data associated with each order.

2.2.4.1.3 Design Considerations

Since the amount of image data associated with each order can vary considerably, a variable length column was defined. If a fixed length column had been chosen and the selected column size was too large, database space would have been wasted. Similarly, if the selected column size was too small, excess continuation rows would have resulted. The specification of a variable length column avoids these situations.

The maximum length specified for the variable length column represents almost an entire DB2 database page. This specification should minimize the number of orders that need to be split because they are too large for a single table row.

2.2.4.1.4 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Order Image table. They include information on the columns, indexes,[15] and physical storage characteristics of the table.

[15] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-21. Order Image Table Design

Column Name	Column Size/ Column Type	Nulls	Description
ORD_ID*	CHAR(13)	Not null	Service order type and number
ORD_FORMAT*	CHAR(1)	Not null	The format of the image entry: negotiated order format (n) or SNS FCIF order output format (f).
SEQ_NO*	SMALL INT	Not null	Sequence number. Maintains order image data in its correct sequence when an entry is split across multiple table rows.
ORD_IMAGE	VARCHAR(4000)	Not null	Order image data

Table 2-22. Order Image Table Characteristics

Table Name	YHGTTB20
Synonym(s)	ORDER_IMAGE
Database	YHSN*D2
Tablespace	YHGTSP20
Storage Group	YHSTOG**
Number of Columns	4
Row Length (bytes)	26 - 4026
Row bytemap	[13][1][2][4000]
Index Name	YHGTX20A
Index Type	CLUSTER, UNIQUE
Index Column(s)	ORD_ID, ORD_FORMAT, SEQ_NO

2.2.4.2 Order Status Table

Once the service negotiation process has been completed for a customer, SRO provides the capability to generate SNS FCIF output messages. Since these messages are generated on a per-order basis, multiple SNS FCIF messages may be associated with a single negotiated service request. The Order Status table provides the mapping between service request numbers and service order numbers to facilitate subsequent tracking of BCC service orders and SNS FCIF messages.

For each service order, this table indicates whether SRO issued an SNS FCIF message associated with the order, the corresponding order due date, and the current status (whether the order has been completed, cancelled, or corrected). If changes are made before the due date that result in the reissuance of a service order and a revised SVCSPC contract, SRO marks the order as corrected and reissues the modified SNS FCIF message associated with the order, if it was issued before. The Order Status table tracks the latest correction suffix associated with each order. Changes requested after an order has been completed require that an entirely new negotiation (i.e., service request) be initiated.

> **NOTE**
>
> SRO does not currently issue SNS FCIF messages to destinations, but always stores these messages in the Order Image table. The ISSUED_IND column in the Order Status table is designed to support the deferred issuing of output messages, pending user review.

2.2.4.2.1 Primary Access Method and Physical Organization

The Order Status table is indexed on the ORD_ID field. This allows access to the data for a specific order number when status information is required. The Order Status table also has a separate index on the SR_NUMBER field. This allows SNS to determine the status of associated orders when only the service request number is provided. The table is physically stored (clustered) in the ascending order of the ORD_ID field. The ORD_ID index is unique, while the SR_NUMBER index is not (since several orders may be associated with a single service request).

2.2.4.2.2 Expected Volume

There is one table entry for each order processed by SRO.

2.2.4.2.3 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Order Status table. They include information on the columns, indexes,[16] and physical storage characteristics of the table.

[16] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-23. Order Status Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SR_NUMBER*	CHAR(13)	Not null	The service request number associated with the service order
ORD_ID*	CHAR(13)	Not null	Service order type and number
CORR_SUFFIX	CHAR(1)	Not null with default	Identifies the correction suffix associated with the service order if correction pass processing has come through SNS.
ISSUED_IND	CHAR(1)	Not null with default	Indicates if SRO has actually issued the SNS FCIF message associated with the order.
STATUS_IND	CHAR(3)	Not null	Indicates which pass of an order has last been processed: pre, cor, can, or pcn.
DUE_DATE	DATE	Null	The scheduled due date for the service order.

Table 2-24. Order Status Table Characteristics

Table Name	YHGTTB30
Synonym(s)	ORDER_STATUS
Database	YHSN*D2
Tablespace	YHGTSP30
Storage Group	YHSTOG**
Number of Columns	6
Row Length (bytes)	44
Row bytemap	[13][13][1][1][3][4]
Index Name	YHGTX30A
Index Type	UNIQUE
Index Column(s)	ORD_ID
Index Name	YHGTX30B
Index Type	CLUSTER, NON-UNIQUE
Index Column(s)	SR_NUMBER

2.2.4.3 SRO Control Table

The SRO Control table provides the capability to determine the proper processing rules and data to use when formatting SNS output. When the SRO process is invoked, information on the serving switch type, desired output message format, and service being supported is passed as input data. This is in turn used as key data to determine the name of a set of SRO processing rules, and a foreign key into the Tag Type table.

The data in this table is loaded at Bellcore and shipped to the BCCs. The data represents reference information used by SRO to generate SNS FCIF output and is not modifiable by the BCC.

2.2.4.3.1 Primary Access Method and Physical Organization

The table is accessed by a combination of the SW_TYPE, DOC_FORMAT, and SVC_IND fields. These three fields provide a unique key for the table. The SW_TYPE (switch type) field describes the manufacturer and model of the intelligent controller. The DOC_FORMAT (document format) field corresponds to the type of output being produced by SRO, and the SVC_IND (service indicator) field represents the type of service to be supported by SRO's internal rules and tables (e.g., Centrex service).

2.2.4.3.2 Nonkey Data

The RULE_SET (rule set name) determines the set of processing rules SRO uses to generate the appropriate output message format. The CTL_KEY (control key) is a short identifier that relates a switch type, document format, and service indicator to a set of tags and logical line types used in SNS output generation.

2.2.4.3.3 Design Considerations

The use of a surrogate key (the CTL_KEY field) was selected because of the length of the primary key for this table. The combined length of the primary key is 15 bytes, comprised of the SW_TYPE, DOC_FORMAT, and SVC_IND fields. The use of an integer reduces the resulting foreign key, which is used in the relationship with the Tag Type table, to four bytes. This design saves eleven bytes per row in the Tag Type table.

2.2.4.3.4 Expected Volume

There is one SRO Control table record for each unique combination of switch type, document format, and service indicator. Current support includes only a handful of switches and a single output document type (SNS FCIF). Service indicators are used as a further qualifier on the selection of a rule set and supported tags/logical line types. Supported service indicators include residence, business, and Centrex. There should be at a maximum approximately sixty combinations of the above information in the SRO Control table. Twenty rows is a more reasonable estimate of the maximum number of table rows, given currently supported services.

2.2.4.3.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the SRO Control table. They include information on the columns, indexes,[17] and physical storage characteristics of the table.

[17] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-25. SRO Control Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SW_TYPE*	CHAR(5)	Not null	Switch type indicator (e.g., AT&T 5E)
DOC_FORMAT*	CHAR(5)	Not null	Document format (e.g., FCIF)
SVC_IND*	CHAR(5)	Not null	Service indicator (CTX, RES, etc.)
RULESET	CHAR(20)	Not null	Rule set name to use for SRO internal processing
CTL_KEY	INTEGER	Not null	Internal key to relate to Tag Type table

Table 2-26. SRO Control Table Characteristics

Table Name	YHGTTB35
Synonym(s)	SRO_CONTROL
Database	YHSN*D2
Tablespace	YHGTSP35
Storage Group	YHSTOG**
Number of Columns	5
Row Length (bytes)	47
Row bytemap	[5][5][5][20][4]
Index Name	YHGTX35A
Index Type	CLUSTERED, UNIQUE
Index Column(s)	SW_TYPE, DOC_FORMAT, SVC_IND

2.2.4.4 Tag Type Table

The SRO process of SNS takes negotiated SVCSPC contract data from the Order Image table and translates it via the SRO rule processor. The rule processor determines the FCIF logical line types to be associated with the formatted SNS FCIF output message. Once this is completed, SRO must then group and format the negotiated service tags with the various logical line types. This is accomplished using the Tag Type table. By accessing the table for each negotiated service tag, SRO can determine the associated logical line types on which the tag can appear. By applying each tag to the appropriate logical lines, the SNS FCIF output message can be generated.

The data in this table is loaded at Bellcore and shipped to the BCCs. The data represents reference information used by SRO to generate SNS FCIF output and is not modifiable by the BCC.

2.2.4.4.1 Primary Access Method and Physical Organization

The Tag Type table is indexed on the CTL_KEY and TAG_ID fields. The TAG_ID (tag identifier), which represents a specific feature or piece of data, enables access to the table on an individual tag basis. Multiple table entries for an individual tag, however, are required for cases in which the associated logical line information needs to be established differently. This could occur because of differences in a tag's usage between switch types or types of negotiated service (i.e., ISDN, Centrex). Multiple entries for a tag are distinguished from one another using the CTL_KEY (control key) field. The control key, whose value is determined from the SRO Control Table, represents the specific conditions under which the table entry for the tag is valid. Together, the key fields represent a unique index for this table, which is physically stored (clustered) in the ascending order of the key fields.

2.2.4.4.2 Expected Volume

At least one table row exists for each tag used in defining negotiated services. Multiple table rows exist for a tag when the usage can vary based on switch type or type of service. It is initially anticipated that about 500 such tags will exist, with an average row length of about 50 bytes.

2.2.4.4.3 Design Considerations

The number of logical line types with which a tag can be associated can vary widely. While 50 characters is sufficient to store the logical line types for most tags, a number of tags exist that require several hundred characters. As SNS grows in functionality, these numbers will continue to increase. As a result, a variable length column was chosen to hold the logical line data.

2.2.4.4.4 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Tag Type table. They include information on the columns, indexes,[18] and physical storage characteristics of the table.

[18] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-27. Tag Type Table Design

Column Name	Column Size/ Column Type	Nulls	Description
CTL_KEY*	INTEGER	Not null	Control key indicating under which conditions the associated table entry is valid. Values are determined via the SRO Control Table.
TAG_ID*	CHAR(8)	Not null	Tag identifier used in generating an SNS FCIF output message.
LOG_LINE_TYPE	VARCHAR(1000)	Not null	The associated SNS FCIF logical line types for which the tag can appear.

Table 2-28. Tag Type Table Characteristics

Table Name	YHGTTB40
Synonym(s)	TAG_TYPE
Database	YHSN*D2
Tablespace	YHGTSP40
Storage Group	YHSTOG**
Number of Columns	3
Row Length (bytes)	23 - 1022
Row bytemap	[4][8][1000]
Index Name	YHGTX40A
Index Type	CLUSTER, UNIQUE
Index Column(s)	CTL_KEY, TAG_ID

2.2.5 SNS TNLIST Database Tables

The SNS TNLIST DB2 tables relate and manage all of the data items necessary for the TN selection process employed by SNS. The database organization can be categorized by the relationships between several types of data. This structure logically consists of the customer, group, list, range, and TN levels.

A *customer* is identified by an attribute known as a tie code. A customer can be an owner of many groups. A *group* can represent either a Centrex or non-Centrex service. A Centrex group is usually identified by a main TN and maps to a specific TN list. Many non-Centrex groups can map to the same TN list. A TN list is a group of TNs for either a Centrex or non-Centrex group that is identified by an internal list identifier and may contain many ranges of TNs. For Centrex, a group number and device identifier uniquely determine a TN list. For non-Centrex services, a product name and device identifier (i.e., switching entity) combination uniquely determine a TN list. A TN range is a consecutive group of telephone numbers identified by the list on which the numbers exist. TNs are the primary item inventoried in TNLIST and are also identified by the list on which they participate. The selectability status of TNs for Centrex customers is kept on an ongoing basis. Activity to the TNs for a customer's account has an effect on their selectability. The TNs for non-Centrex customers are not tracked by account after they have been selected. (Figures 2-4 and 2-5 show the structure of the TNLIST database.)

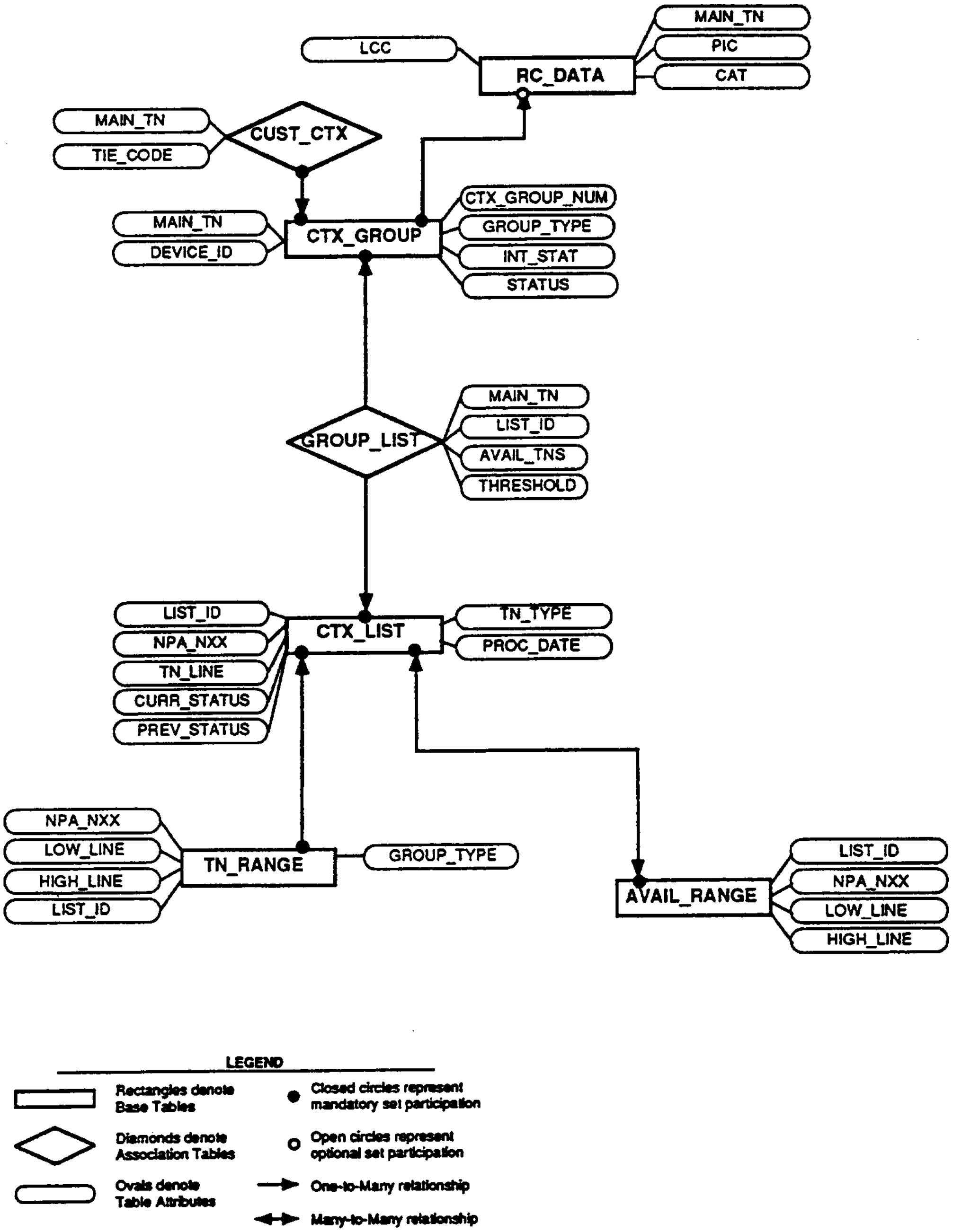

Figure 2-4. TNLIST Database Structure (1 of 2)

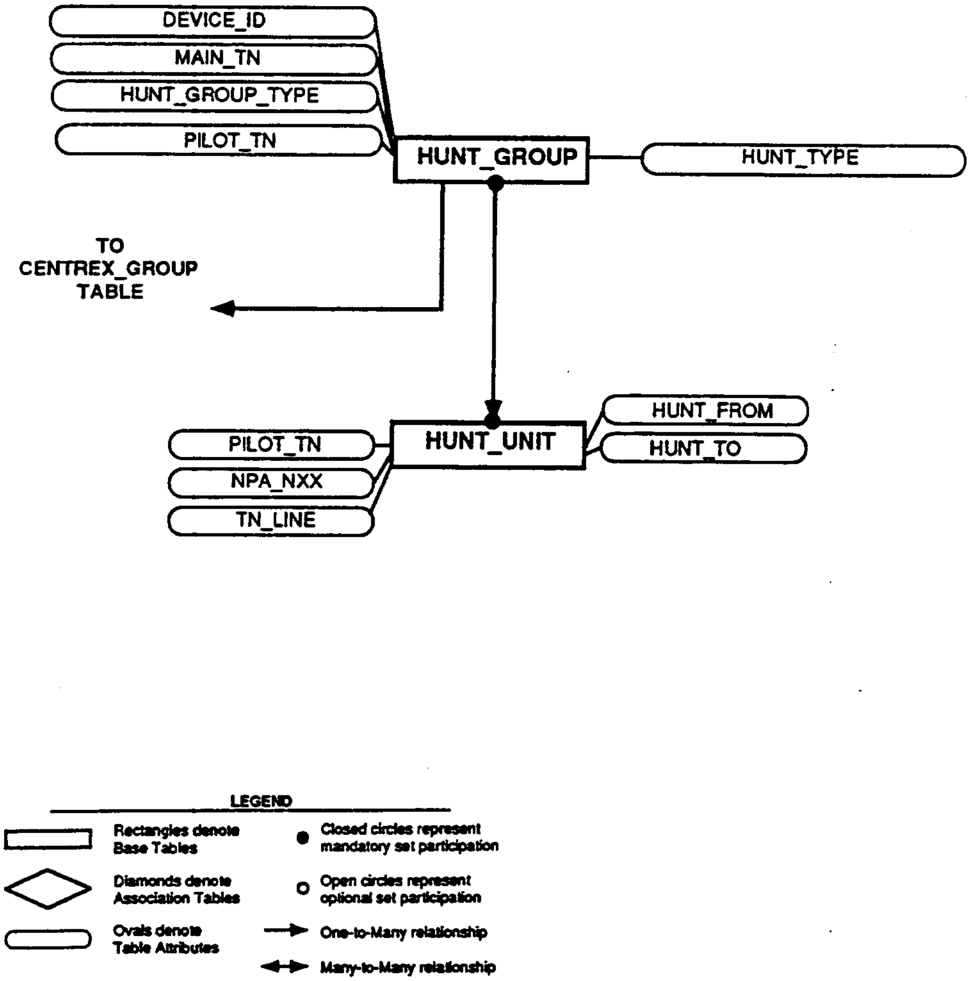
DEVICE_ID
MAIN_TN
HUNT_GROUP_TYPE
PILOT_TN
HUNT_GROUP
HUNT_TYPE
TO
CENTREX_GROUP
TABLE
HUNT_FROM
HUNT_TO
PILOT_TN
HUNT_UNIT
NPA_NXX
TN_LINE
LEGEND
Rectangles denote Base Tables
Diamonds denote Association Tables
Ovals denote Table Attributes
Closed circles represent mandatory set participation
Open circles represent optional set participation
One-to-Many relationship
Many-to-Many relationship

Figure 2-5. TNLIST Database Structure (2 of 2)

The logical data just described is partitioned into two pairs of tables, one for Centrex (pre-established) customers and one for non-Centrex or product-oriented customers. The latter type of customer is referred to as a "pool" type of customer, based on the fact that TNs are selected for them from a pool of selectable numbers that are not associated with a specific customer.

The following sections, which are organized alphabetically, present a description of each of the TNLIST tables, as well as information about their key and nonkey columns, primary access methods, basic design methodology, and physical organization.

2.2.5.1 Available Telephone Number Range Table

The Available Telephone Number Range table (AVAIL_RANGE) stores information about consecutive blocks of telephone numbers on a telephone number list basis. It is used by the SNS TNLIST telephone number selection processing.

2.2.5.1.1 Primary Access Method and Physical Organization

The table is physically stored in the ascending order of the telephone number list identifier, numbering plan area and exchange (NPA_NXX), and the low and high lines for the consecutive block. These fields also provide a unique key for the table. The table is accessed via a list identifier and the desired quantity of telephone numbers. SQL processing determines the appropriate record(s) to satisfy the request.

2.2.5.1.2 Nonkey Data

None.

2.2.5.1.3 Expected Volume

For each list of telephone numbers in the SNS TNLIST database, there are many blocks of available consecutive numbers. At the bottom of the list are consecutive blocks containing one number. However, the number of records should be significantly less than the number of records in the Centrex Telephone Number List or Pool Telephone Number List tables. Non-consecutive telephone number selection processing may try to maintain large blocks by choosing from the bottom of the list first. The volume of records in this table may increase if large blocks are split and blocks of one or two numbers are not selected.

2.2.5.1.4 Design Considerations

In selecting consecutive blocks of telephone numbers it is necessary to find the closest fit in terms of quantity to satisfy the request. For example, if ten consecutive telephone numbers are requested, a block of twelve would be selected over a block of twenty to satisfy the request. In this way, a request for twenty consecutive numbers could still be satisfied at a later time. The most straightforward way of processing requests for consecutive telephone numbers is to read from a list sorted according to decreasing quantities. If no block exists that is large enough to satisfy the request, the set of smaller blocks can be returned until the request is satisfied.

2.2.5.1.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Available Telephone Number Range table. They include information on the columns, indexes,[19] and physical storage characteristics of the table.

[19] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-29. Available TN Range Table Design

Column Name	Column Size/ Column Type	Nulls	Description
LIST_ID*	INTEGER	Not null	TN list identifier
NPA_NXX*	INTEGER	Not null	Numbering plan area and exchange
LOW_LINE*	SMALLINT	Not null	Low line in range
HIGH_LINE*	SMALLINT	Not null	High line in range

Table 2-30. Available TN Range Table Characteristics

Table Name	YHTLTB10
Synonym(s)	AVAIL_RANGE
Database	YHSN*D2
Tablespace	YHTLSP10
Storage Group	YHSTOG**
Number of Columns	4
Row Length (bytes)	20
Row bytemap	[4][4][2][2]
Index Name	YHTLX10A
Index Type	CLUSTER, UNIQUE
Index Column(s)	LIST_ID, NPA_NXX, LOW_LINE, HIGH_LINE

2.2.5.2 Centrex Group Table

The Centrex Group table (CTX_GROUP) keeps information associated with Centrex groups that have been established in TNLIST. Centrex groups are pre-established in TNLIST before SNS negotiation activities.

The Centrex Group table is used by transactions to determine the main TN for a Centrex group when a Centrex group number and switching device identifier (eleven character CLLI code) are input. This main TN can then be mapped to the correct list of TNs. This mapping is discussed in more detail in the description of the Group to TN List Association table. The Centrex Group table is also used to determine the default recent change information and intercept status for a Centrex group. Additionally, maintenance contract processing uses this table for validation.

2.2.5.2.1 Primary Access Method and Physical Organization

The table is indexed on the ten-digit main TN (MAIN_TN) for the Centrex group, which provides a unique identifier for each row. The table is also indexed on the DEVICE_ID and CTX_GROUP fields. The table is physically stored (clustered) in the ascending order of the main TNs. Either a main TN or the combination of a device identifier (eleven character CLLI) and Centrex group number may be used to uniquely identify a Centrex group.

2.2.5.2.2 Nonkey Data

The GROUP_TYPE and INT_STAT fields represent the Centrex group type (regular or non-Centrex) and the default intercept status for the group. Other data items associated with a Centrex group include default recent change information, stored in the Recent Change Data table. A group status field is also used internally to determine whether TNs exist for the group.

2.2.5.2.3 Expected Volume

There is one record for each Centrex group supported in SNS.

2.2.5.2.4 Design Considerations

This table provides high-level access into TNLIST data and normalizes the Centrex group information away from the TN records in the CTX_LIST table.

2.2.5.2.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Centrex Group table. They include information on the columns, indexes,[20] and physical storage characteristics of the table.

Table 2-31. Centrex Group Table Design

Column Name	Column Size/ Column Type	Nulls	Description
MAIN_TN*	CHAR(10)	Not null	Main TN for the Centrex group
DEVICE_ID*	CHAR(11)	Not null	Intelligent controller
CTX_GROUP_NUM*	CHAR(18)	Not null	Centrex group number
GROUP_TYPE	CHAR(2)	Not null	Centrex group type
INT_STAT	CHAR(3)	Not null with default	Default intercept status for group
STATUS	CHAR(2)	Not null with default	Status of group

[20] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name.

Table 2-32. Centrex Group Table Characteristics

Table Name	YHTLTB20
Synonym(s)	CTX_GROUP
Database	YHSN*D2
Tablespace	YHTLSP20
Storage Group	YHSTOG**
Number of Columns	6
Row Length (bytes)	54
Row bytemap	[10][11][18][2][3][2]
Index Name	YHTLX20A
Index Type	CLUSTER, UNIQUE
Index Column(s)	MAIN_TN
Index Name	YHTLX20B
Index Type	UNIQUE
Index Column(s)	DEVICE_ID, CTX_GROUP_NUM

2.2.5.3 Centrex Telephone Number List Table

The Centrex Telephone Number List table (CTX_LIST) stores all information about specific telephone numbers relevant to Centrex TN selection. These numbers are grouped according to TN lists. A unique list identifier is used to differentiate between lists. This numeric list ID is assigned by TNLIST when TNs are added to a Centrex group. The negotiator does not provide the list identifier (LIST_ID) as input for telephone number selection. Instead, the negotiator provides a main Centrex TN or Centrex group number and device identifier, and TNLIST processing maps these items to the correct TN list from which to choose.

One major role of the LIST_ID is to abstract the organization of TNs away from any single Centrex group and provide a structure that allows many Centrex groups to choose from a shared list of telephone numbers.

2.2.5.3.1 Primary Access Method and Physical Organization

The Centrex Telephone Number List table is organized in the ascending order of list identifier (LIST_ID), numbering plan area and exchange (NPA_NXX), and line (TN_LINE, the last four digits of a telephone number). The two access methods are:

1. LIST_ID (internal to TNLIST processing)
2. NPA_NXX and TN_LINE, as specified by the negotiator.

TN selection marks telephone numbers as unselectable (as soon as they are selected), so that they cannot be chosen by other SNS processes. TNs may be returned to the selectable state by an SNS negotiation process in the event of an order cancellation or TN return. Since the types of Centrex groups represented in this table correspond to pre-established customers, the TNs within them are tracked as long as they are associated with their specific groups.

2.2.5.3.2 Nonkey Data

The nonkey data for the Centrex Telephone Number List table is composed of the TN status[21], TN type, TN subtype, and processing date (PROC_DATE). The processing date is a date stamp that reflects

[21] The current, previous, and next selectability statuses of the TN are stored to allow for the restoration of a TN to its previous status in the event that an order negotiation session is cancelled, or the number is returned as unwanted during selection.

the last time that any application-related activity (excluding database back-ups or reorganizations) occurred to a specific TN.

2.2.5.3.3 Expected Volume

There should be one record for each Centrex telephone number supported in SNS.

2.2.5.3.4 Design Considerations

The records are stored by list identifier in ascending TN order to optimize the TN selection process.

2.2.5.3.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Centrex List table. They include information on the columns, indexes,[22] and physical storage characteristics of the table.

[22] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name.

Table 2-33. Centrex List Table Design

Column Name	Column Size/ Column Type	Nulls	Description
LIST_ID*	INTEGER	Not null	TN list identifier
NPA_NXX*	CHAR(6)	Not null	Numbering plan area and exchange
TN_LINE*	CHAR(4)	Not null	Line
CURR_STATUS	CHAR(2)	Not null with default	Current TN status
PREV_STATUS	CHAR(2)	Not null with default	Previous TN status
NEXT_STATUS†	CHAR(2)	Not null with default	Next TN status
TN_TYPE	CHAR(1)	Not null	Type of TN (Q for Centrex, x for res., etc.)
TN_SUBTYPE	CHAR(3)	Not null with default	Subtype of TN (PDN, VDN, etc.)
EFF_DATE†	CHAR(10)	Not null with default	Effective date for activity
PROC_DATE	DATE	Not null with default	Date stamp for most recent activity

† Reserved for future use.

Table 2-34. Centrex List Table Characteristics

Table Name	YHTLTB25
Synonym(s)	CTX_LIST
Database	YHSN*D2
Tablespace	YHTLSP25
Storage Group	YHSTOG**
Number of Columns	10
Row Length (bytes)	46
Row bytemap	[4][6][4][2][2][2][1][3][10][4]
Index Name	YHTLX25A
Index Type	CLUSTER, UNIQUE
Index Column(s)	LIST_ID, NPA_NXX, TN_LINE
Index Name	YHTLX25B
Index Type	NON-UNIQUE
Index Column(s)	LIST_ID, CURR_STATUS

2.2.5.4 Customer to Centrex Group Association Table

The Customer to Centrex Group Association table (CUST_CTX) relates the tie code information for a customer to the Centrex groups they own. This relationship is only kept for regular Centrex customers, since they are the only type of customers with tie codes assigned to them (i.e., not non-Centrex customers). A regular Centrex customer has only one tie code, regardless of the number of Centrex groups owned. Tie codes are unique across the TNLIST database tables.

2.2.5.4.1 Primary Access Method and Physical Organization

This association table is physically organized on the ascending sequence of main TNs and is accessible by both main TN and tie code. For activity related to all the groups for an account, the tie code and the related Centrex groups can be derived from an input main TN from any one of the related groups.

2.2.5.4.2 Nonkey Data

None.

2.2.5.4.3 Expected Volume

One record exists for each customer-to-Centrex-group combination.

2.2.5.4.4 Design Considerations

This table maps large accounts to their Centrex groups.

2.2.5.4.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Customer to Centrex Group Association table. They include information on the columns, indexes,[23] and physical storage characteristics of the table.

[23] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-35. Customer to Centrex Group Association Table Design

Column Name	Column Size/ Column Type	Nulls	Description
MAIN_TN*	CHAR(10)	Not null	Main TN for the Centrex group
TIE_CODE*	CHAR(11)	Not null	Tie code for the customer

Table 2-36. Customer to Centrex Group Association Table Characteristics

Table Name	YHTLTB30
Synonym(s)	CUST_CTX
Database	YHSN*D2
Tablespace	YHTLSP30
Storage Group	YHSTOG**
Number of Columns	2
Row Length (bytes)	29
Row bytemap	[10][11]
Index Name	YHTLX30A
Index Type	CLUSTER, UNIQUE
Index Column(s)	MAIN_TN
Index Name	YHTLX30B
Index Type	NON-UNIQUE
Index Column(s)	TIE_CODE

2.2.5.5 Group to TN List Association Table

The Group to TN List Association Table (GROUP_LIST) facilitates a many-to-many relationship between the group-level table (CTX_GROUP) and the list-level table (CTX_LIST). It is used by all TN selection transactions to correctly map the group-level information to the list-level information, as well as to monitor TN notices.

2.2.5.5.1 Primary Access Method and Physical Organization

For Centrex TN selection, a main TN is received as input, or the Centrex group table maps the Centrex group number and device ID to a main TN. This main TN is then used as the key into the GROUP_LIST table to derive the correct list from which to select telephone numbers. Lists are identified by an integer value internal to TNLIST called LIST_ID. The list identified according to the process described above has corresponding record data that exists in the CTX_LIST table.

For regular Centrex, there exists one Centrex group per TN list. When portable telephone numbers are eventually supported, Centrex groups on different devices may point to the same TN List, and thus share the same pool of TNs. The TN Range table may be augmented to reflect the set of valid switching entities for any given range. For non-Centrex groups, there exists one list per service type, optional product, and device combination.

The GROUP_LIST table is physically stored in the ascending order of main TNs.

2.2.5.5.2 Nonkey Data

The threshold information consists of the THRESHOLD and AVAIL_TNS fields. The THRESHOLD field is populated when a group is created or changed using a maintenance or negotiation transaction that affects the group. It contains the minimum allowable number of selectable TNs on the list before a notifier is issued to replenish the list. The AVAIL_TNs field is updated whenever TNs are selected from the list and reflects the remaining number of selectable TNs. The TN selection process generates monitor notices when the number of selectable TNs drops below the BCC-defined threshold.

2.2.5.5.3 Expected Volume

There is one Group to TN List Association table record for each TN list in SNS.

2.2.5.5.4 Design Considerations

The capability to monitor the number of selectable TNs for a list and automatically generate a notifier is best managed through this table since many non-Centrex group records might point to a single list.

2.2.5.5.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Group to TN List Association table. They include information on the columns, indexes,[24] and physical storage characteristics of the table.

Table 2-37. Group to TN List Association Table Design

Column Name	Column Size/ Column Type	Nulls	Description
MAIN_TN*	CHAR(10)	Not null	Main TN for the Centrex group
LIST_ID*	INTEGER	Not null	List identifier
AVAIL_TNS	SMALLINT	Not null with default	Number of available TNs
THRESHOLD	SMALLINT	Not null with default	Threshold of available TNs

[24] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-38. Group to TN List Association Table Characteristics

Table Name	YHTLTB40
Synonym(s)	GROUP_LIST
Database	YHSN*D2
Tablespace	YHTLSP40
Storage Group	YHSTOG**
Number of Columns	4
Row Length (bytes)	26
Row bytemap	[10][4][2][2]
Index Name	YHTLX40A
Index Type	CLUSTER, UNIQUE
Index Column(s)	MAIN_TN
Index Name	YHTLX40B
Index Type	NON-UNIQUE
Index Column(s)	LIST_ID

2.2.5.6 Hunt Group Table

The Hunt Group table (HUNT_GROUP) records information that is necessary for the creation, validation, and maintenance of hunting at a group level. This table relates terminal and line information stored in the Hunt Unit table to the device information stored in the Hunt Device table. It also tracks available terminal identifiers for hunt groups.

2.2.5.6.1 Primary Access Method and Physical Organization

Information in the Hunt Group table is physically stored in the ascending order of pilot TN. The pilot TN is the first number in the hunt sequence. The pilot TN provides a unique key for this table.

2.2.5.6.2 Nonkey Data

The hunt type is a one-character identifier representing the type of hunt group, currently limited to Series Completion.

2.2.5.6.3 Expected Volume

There is one hunt group record for each hunt group supported by SNS TNLIST. The maximum number of hunt groups per device is currently limited to around two-thousand, depending on device type.

2.2.5.6.4 Design Considerations

The hunt group level of information is needed to normalize many attributes out of the hunt unit level. This includes all of the nonkey data, as well as the relationships between a hunt group and its owners. The hunt group owners consist of both the device on which the group resides and the Centrex group or product for which the group is created.

2.2.5.6.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Hunt Group table. They include information on the columns, indexes,[25] and physical storage characteristics of the table.

[25] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-39. Hunt Group Table Design

Column Name	Column Size/ Column Type	Nulls	Description
PILOT_TN*	CHAR(10)	Not null	First TN in group
DEVICE_ID	CHAR(11)	Not null	Intelligent controller
MAIN_TN	CHAR(10)	Not null	Owning pool or Centrex group
HUNT_GROUP_TYPE	CHAR(1)	Not null	Hunt group type (SCH, MLH)
HUNT_GROUP_ID†	INTEGER	Not null	Hunt group identifier
PILOT_TERMINAL†	SMALLINT	Not null with default	First terminal in group
HUNT_TYPE	CHAR(1)	Not null	Regular, circular, UCD, etc.
NUM_HUNT_MBRS†	SMALLINT	Not null with default	Number of hunt members for MLH
NUM_NHUNT_MBRS†	SMALLINT	Not null with default	Number of nonhunt members
NEXT_HUNT_TERM†	SMALLINT	Not null with default	Next hunt term. number for MLH
NEXT_NHUNT_TERM†	SMALLINT	Not null with default	Next nonhunt term. number for MLH

† Reserved for future use.

Table 2-40. Hunt Group Table Characteristics

Table Name	YHTLTB50
Synonym(s)	HUNT_GROUP
Database	YHSN*D2
Tablespace	YHTLSP50
Storage Group	YHSTOG**
Number of Columns	11
Row Length (bytes)	55
Row bytemap	[10] [11] [10] [1] [4] [2] [1] [2] [2] [2] [2]
Index Name	YHTLX50A
Index Type	CLUSTER, UNIQUE
Index Column(s)	PILOT_TN

2.2.5.7 Hunt Unit Table

The Hunt Unit table (HUNT_UNIT) contains information about the individual lines and terminals that make up the hunt groups stored in SNS TNLIST. The lines and terminals are collectively referred to as hunt units in SNS TNLIST. These hunt units are grouped together by a hunt group identifier as well as their hunting pattern. Information is also stored for validation purposes on a hunt unit level.

2.2.5.7.1 Primary Access Method and Physical Organization

The records in the Hunt Unit table are physically stored in the ascending order of their pilot TN. A hunt unit is allowed to participate as a member of multiple hunt groups under certain conditions. Therefore, no unique identifier is defined for hunt unit records.

2.2.5.7.2 Nonkey Data

The numbering plan area, exchange, and four-digit line number are stored for all members of an SCH group. The hunting sequence is reflected as a linked list of telephone numbers for SCH. This list is implemented through the hunt-to and hunt-from fields.

2.2.5.7.3 Expected Volume

There is one hunt unit record for each member of each hunt group supported by SNS TNLIST. Additionally, members that participate in multiple hunt groups have a record for each. The number of hunt group members allowed is dependent on the type of owning device and level of switch software.

2.2.5.7.4 Design Considerations

The hunt unit level of information is at the bottom of the hunt hierarchy. This information is related to both the hunt group and hunt device levels. When creating or changing a hunt group, every member is deleted, and the new image of the group replaces the old. The added complexity and processing overhead associated with analyzing and modifying the old group information can thus be avoided.

2.2.5.7.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Hunt Unit table. They include information on the columns, indexes,[26] and physical storage characteristics of the table.

[26] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-41. Hunt Unit Table Design

Column Name	Column Size/ Column Type	Nulls	Description
PILOT_TN*	CHAR(10)	Not null	Pilot TN for hunt group
NPA_NXX	INTEGER	Not null with default	NPA and exchange
TN_LINE	SMALLINT	Not null with default	Four-digit line number
TERMINAL_ID†	SMALLINT	Not null with default	Terminal identifier
HUNT_FROM	INTEGER	Not null with default	Terminal ID or TN
HUNT_TO	INTEGER	Not null with default	Terminal ID or TN
HUNT_IND†	CHAR(2)	Not null with default	Indicates if unit hunts
NUM_GROUPS†	SMALLINT	Not null with default	Number of groups hunting to unit

† Reserved for future use.

Table 2-42. Hunt Unit Table Characteristics

Table Name	YHTLTB60
Synonym(s)	HUNT_UNIT
Database	YHSN*D2
Tablespace	YHTLSP60
Storage Group	YHSTOG**
Number of Columns	8
Row Length (bytes)	38
Row bytemap	[10][4][2][2][4][4][2][2]
Index Name	YHTLX60A
Index Type	CLUSTER
Index Column(s)	PILOT_TN

2.2.5.8 Pool Group Table

> **NOTE**
>
> The Pool Group table is reserved for future use in SNS.

The Pool Group table (POOL_GROUP) keeps information associated with non-Centrex groups that have been established in TNLIST.

It is assumed that a Pool Group is pre-established in TNLIST for non-Centrex customers prior to SNS negotiation activities for that group. However, the TNs in a pool group are not associated with any specific customer.

The Pool Group table is used by transactions to determine the main TN for a non-Centrex group when a service type, optional product, and device identifier (eleven-character CLLI code) are input. This main TN can then be mapped to the correct list of TNs. This mapping is discussed in more detail in the description of the Group to TN List Association table in this document. Maintenance contracts use this table for validation.

2.2.5.8.1 Primary Access Method and Physical Organization

The table is indexed on the SVC_TYPE, DEVICE_ID, and PRODUCT fields, which provide a unique identifier for each row. The table is also indexed on the ten-digit main TN (MAIN_TN) for a non-Centrex group. The table is physically stored (clustered) in the ascending order of the SVC_TYPE, DEVICE_ID, and PRODUCT. Either a main TN, or the combination of a service type, device identifier (eleven-character CLLI code) and optionally product are used to uniquely identify a non-Centrex group.

2.2.5.8.2 Nonkey Data

The GROUP_TYPE and INT_STAT fields represent the group type (i.e., non-Centrex) and the default intercept status for the group. A group status field is also used internally to determine whether TNs exist for the group.

2.2.5.8.3 Expected Volume

There is one Pool Group record for each non-Centrex group.

2.2.5.8.4 Design Considerations

This table provides high-level access into TNLIST data and normalizes the non-Centrex group information away from the TN records in the POOL_LIST table.

2.2.5.8.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Pool Group table. They include information on the columns, indexes,[27] and physical storage characteristics of the table.

[27] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-43. Pool Group Table Design

Column Name	Column Size/ Column Type	Nulls	Description
SVC_TYPE*	CHAR(15)	Not null	Service type for group (res., bus.)
DEVICE_ID*	CHAR(11)	Not null	Intelligent controller
PRODUCT*	CHAR(30)	Not null	Product name for the group
MAIN_TN*	CHAR(10)	Not null	Main TN for the group
GROUP_TYPE	CHAR(2)	Not null	Group type
INT_STAT	CHAR(3)	Not null with default	Default intercept status for group
STATUS	CHAR(2)	Not null with default	Status of group

Table 2-44. Pool Group Table Characteristics

Table Name	YHTLTB70
Synonym(s)	POOL_GROUP
Database	YHSN*D2
Tablespace	YHTLSP70
Storage Group	YHSTOG**
Number of Columns	7
Row Length (bytes)	81
Row bytemap	[15][11][30][10][2][3][2]
Index Name	YHTLX70A
Index Type	CLUSTER, UNIQUE
Index Column(s)	SVC_TYPE, DEVICE_ID, PRODUCT
Index Name	YHTLX70B
Index Type	UNIQUE
Index Column(s)	MAIN_TN

2.2.5.9 Pool Telephone Number List Table

> **NOTE**
>
> The Pool Telephone Number List table is reserved for future use in SNS.

The Pool Telephone Number List table (POOL_LIST) is a structure analogous to the Centrex Telephone Number List table. This table stores all information about specific telephone numbers relevant to pool TN selection. Pool TNs are not associated with a specific customer, regardless of their status (selectable or unselectable). These numbers are grouped according to TN lists. A unique list identifier is used to differentiate the lists. This numeric list ID is assigned by TNLIST when TNs are added to a non-Centrex group. The negotiator does not provide the LIST_ID as input for telephone number selection. Instead, the negotiator provides an existing TN or service type, optional product, and device identifier. TNLIST processing maps these items to the correct TN list.

Non-Centrex groups are pre-populated with an available pool of TNs to allow small, first- contact customers assignment of telephone numbers. However, the non- Centrex groups must also be pre-established for a given device, and the available pool replenished as it becomes low.

2.2.5.9.1 Primary Access Method and Physical Organization

The Pool Telephone Number List table is organized in the ascending order of list identifier (LIST_ID), numbering plan area and exchange (NPA_NXX), and line (TN_LINE), respectively. Access is accomplished via LIST_ID (internal to TNLIST processing) or by NPA_NXX and TN_LINE, as specified by the negotiator.

TN selection marks telephone numbers as unselectable (as soon as they are selected) so that they cannot be chosen by other SNS processes. TNs may be returned to the selectable state by an SNS negotiation process. Since the types of groups represented in this table do not correspond to pre-established customers, the TNs within them are not tracked with a specific customer.

2.2.5.9.2 Nonkey Data

The nonkey data for the Pool List table is composed of the telephone number status,[28] TN type, subtype (not currently used in TNLIST processing), and a

[28] The current and previous statuses of the TN are stored to allow for the restoration of a TN to its previous status in the event that an order negotiation session is cancelled or the number is returned as unwanted during selection.

processing date (PROC_DATE). The processing date is the same as described for the Centrex Telephone Number List table.

2.2.5.9.3 Expected Volume

Records in the POOL_LIST table are stored by list identifier in ascending TN order to optimize the TN selection process.

2.2.5.9.4 Design Considerations

The records are stored by list identifier in ascending TN order to optimize the TN selection process.

2.2.5.9.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Pool Telephone Number List table. They include information on the columns, indexes,[29] and physical storage characteristics of the table.

[29] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-45. Pool Telephone Number List Table Design

Column Name	Column Size/ Column Type	Nulls	Description
LIST_ID*	INTEGER	Not null	TN list identifier
NPA_NXX*	CHAR(6)	Not null	Numbering plan area and exchange
TN_LINE*	CHAR(4)	Not null	Line
CURR_STATUS*	CHAR(2)	Not null	Current TN status
PREV_STATUS	CHAR(2)	Not null with default	Previous TN status
NEXT_STATUS	CHAR(2)	Not null with default	Next TN status
TN_TYPE	CHAR(1)	Not null	Type of TN (Q for Centrex X for res.)
TN_SUBTYPE	CHAR(3)	Not null with default	Subtype of TN (PDN, VDN, etc.)
PROC_DATE	DATE	Not null with default	Date stamp for most recent activity

Table 2-46. Pool Telephone Number List Table Characteristics

Table Name	YHTLTB75
Synonym(s)	POOL_LIST
Database	YHSN*D2
Tablespace	YHTLSP75
Storage Group	YHSTOG**
Number of Columns	9
Row Length (bytes)	36
Record bytemap	[4][6][4][2][2][2][1][3][4]
Index Name	YHTLX75A
Index Type	CLUSTER, UNIQUE
Index Column(s)	LIST_ID, NPA_NXX, TN_LINE
Index Name	YHTLX75B
Index Type	NON-UNIQUE
Index Column(s)	LIST_ID, CURR_STATUS

2.2.5.10 Recent Change Data Table

The Recent Change Data table (RC_DATA) stores all of the recent change information for a Centrex group that is needed on a service order. Recent change information is composed of Centrex Access Treatments (CATs), Line Class Codes (LCCs), and Primary Interchange Carriers (PICs). For each group, there is one CAT code, one LCC code, and one PIC code.

2.2.5.10.1 Primary Access Method and Physical Organization

The Recent Change Data table provides access to data through the main telephone number for a group. The main telephone number is either input or derived from the Centrex Group table. The RC_DATA table is organized on the ascending sequence of main TNs. When a TN is selected and the default recent change data needs to be returned, all the rows in the RC_DATA table that contain the main TN for the object group[30] are retrieved.

2.2.5.10.2 Nonkey Data

The LCC and CAT fields contain data associated with a specific group. The LCC and CAT columns are single-valued. A primary interexchange carrier (PIC) is also stored for the Centrex group.

2.2.5.10.3 Expected Volume

There is one RC_DATA record per Centrex group.

2.2.5.10.4 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Recent Change Data table. They include information on the columns, indexes,[31] and physical storage characteristics of the table.

[30] The object group is the Centrex group from which the TNs were selected.

[31] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-47. Recent Change Table Design

Column Name	Column Size/ Column Type	Nulls	Description
MAIN_TN*	CHAR(10)	Not null	Main TN for the Centrex group
PIC	CHAR(4)	Not null with default	Primary interexchange carrier
CAT	CHAR(2)	Not null with default	CAT code
LCC	CHAR(3)	Not null with default	Line class code
CCF†	CHAR(70)	Not null with default	Customized call forwarding features

† Reserved for future use.

Table 2-48. Recent Change Table Characteristics

Table Name	YHTLTB90
Synonym(s)	RC_DATA
Database	YHSN*D2
Tablespace	YHTLSP90
Storage Group	YHSTOG**
Number of Columns	5
Row Length (bytes)	97
Row bytemap	[10][4][2][3][70]
Index Name	YHTLX90
Index Type	CLUSTER
Index Column(s)	MAIN_TN

2.2.5.11 Telephone Number Range Table

The Telephone Number Range table (TN_RANGE) contains information about all ranges of TNs in Centrex groups. At the time a group is created, TN ranges must be established. These ranges continue to be associated with the group until either the range or the group is deleted. When the input to the TN selection process is an existing TN for the requested group (as opposed to a main TN), the TN Range table is used to identify the proper telephone number list from which to choose telephone numbers.

2.2.5.11.1 Primary Access Method and Physical Organization

Access to the TN Range table is accomplished via either a full ten-digit telephone number or a list identifier. If it is necessary to determine the correct TN list for an existing telephone number, the ten-digit TN is used for access. When it is necessary to identify the TN ranges for a specific TN list, the list ID is used for access. The table is physically clustered by list in the ascending sequence of the low and high TNs for a range within an NPA and NXX.

2.2.5.11.2 Nonkey Data

The group type identifies the type of Centrex group that is associated with the range.

2.2.5.11.3 Expected Volume

The volume of TN range records varies depending on the block sizes of TNs assigned to Centrex groups. Each record corresponds to a range of consecutive TNs allocated to a group.

2.2.5.11.4 Design Considerations

This table facilitates efficient access to TN data in the CTX_LIST table for subsequent TN selection processing.

2.2.5.11.5 DB2 Table Data

The charts below summarize the important DB2 design aspects of the Telephone Number Range table. They include information on the columns, indexes,[32] and physical storage characteristics of the table.

[32] Columns that are part of an index are suffixed by an asterisk in the table. This asterisk is for notational purposes and does not appear in DB2 as a part of the column name. The asterisk in the database name indicates the position for an entity identifier and the double asterisks in the storage group should be substituted with a BCC-chosen storage group identifier.

Table 2-49. TN Range Table Design

Column Name	Column Size/ Column Type	Nulls	Description
NPA_NXX*	INTEGER	Not null	Numbering plan area and exchange
LOW_LINE*	SMALLINT	Not null	Low line in range
HIGH_LINE*	SMALLINT	Not null	High line in range
LIST_ID*	INTEGER	Not null	TN list identifier
GROUP_TYPE	CHAR(2)	Not null	Group type for the owning list

Table 2-50. TN Range Table Characteristics

Table Name	YHTLTB95
Synonym(s)	TN_RANGE
Database	YHSN*D2
Tablespace	YHTLSP95
Storage Group	YHSTOG**
Number of Columns	5
Row Length (bytes)	22
Record bytemap	[4][2][2][4][2]
Index Name	YHTLX95A
Index Type	CLUSTER, UNIQUE
Index Column(s)	NPA_NXX, LOW_LINE, HIGH_LINE
Index Name	YHTLX95B
Index Type	UNIQUE
Index Column(s)	LIST_ID, NPA_NXX, LOW_LINE, HIGH_LINE

3. Database Naming Conventions

The following subsections explain the suggested database naming conventions for the SNS IMS and DB2 databases.

3.1 SNS IMS Databases

An IMS database name is the same as its database definition (DBD) name. The name is of the general form "YHsstizz." The "YH" is the SNS product line. The "ss" is the SNS subsystem to which the database belongs. The "t" is a unique character identifying the database. The "i" is the entity code. The "zz" is "DD" for HDAM databases.

3.2 SNS DB2 Databases

There exists a hierarchy of physical objects that make up the SNS DB2 database. From top to bottom, this hierarchy consists of a DATABASE, TABLESPACEs, TABLEs, INDEXes and SYNONYMs. The STOGRP is not usually considered a physical object, but it is still necessary to create the DATABASE and TABLESPACEs. The remainder of the hierarchy defines the dependencies between objects and the order in which they must be created.

The objects in SNS DB2 databases follow IBM restrictions for identifier naming. (These restrictions may be found in the IBM manual, *SQL Reference Guide.*)

3.2.1 DATABASE and STOGRP Naming Conventions

The DATABASE and STOGRP names are referenced in the Data Definition Language (DDL) statements used to create the various physical objects. The delivered SNS runstreams do not specify actual names for Databases or Storage Groups. Instead, variable parameters are supplied that should be replaced by the BCC.

To remain consistent with the Bellcore Separate Product Line Identifier (SPLID) naming scheme, it is recommended that the first two characters of SNS DB2 Database and Stogroup names be set to "YH". This avoids the possibility of conflict with DB2 objects from other Bellcore Centrally Developed Systems (CDSs).

It is recommended that the character "D" appear somewhere in a Database name, and the character "G" appear somewhere in a Storage Group name. The remaining characters may use any naming convention employed by the BCC.

3.2.2 TABLESPACE Naming Conventions

SNS has several tablespaces whose names are eight characters in length and have the form "YHSSSPnn." The "YH" is the SNS product line. The "SS" is arbitrarily designated to signify the SNS subsystem (i.e., Exception Handler or TNLIST). The "SP" stands for "tableSPace." The "nn" is a two-digit number that uniquely identifies the tablespace.

3.2.3 TABLE Naming Conventions

Table names are eight characters and follow the form "YHSSTBnn." The "YH" is the SNS product line. The "SS" is arbitrarily designated to signify the SNS subsystem (i.e., Exception Handler or TNLIST). The "TB" stands for "TaBle." The "nn" is a two-digit number that uniquely identifies the table.

3.2.4 INDEX and SYNONYM Naming Conventions

Tables also have indexes defined for them. The names of these indexes are not known by an application programmer, since DB2 automatically decides which indexes to use for a given SQL statement. The index names are only used by a DBA for index maintenance. SNS DB2 indexes are named and numbered sequentially in the form "YHSSXnna" as above, where "SS" again stands for subsystem, "X" designates "indeX," "nn" matches the table number for which the index is created, and "a" is a single alphabetic character that uniquely identifies the index.

SYNONYMs are created for SNS DB2 tables to allow application programs alternative "English" names for referencing tables.

The naming convention in SNS for the SYNONYM (English name) for an association table borrows several characters from each of the base tables that are being related (e.g., in TNLIST, the association table relating the CTX_GROUP and CTX_LIST tables is named GROUP_LIST).

4. Database Sizing

To enable the BCCs to estimate the DASD requirements for their SNS databases, an SNS Database Sizing Model was developed. In addition to calculating database space allocations, the model also determines values for several related parameters needed during the SNS installation process to ensure proper database sizing. The model is considered to be a part of the general SNS product release.

The SNS Database Sizing Model is implemented as a spreadsheet under the LOTUS 1-2-3® software. It provides a user-friendly environment for performing the large volume of calculations involved. LOTUS 1-2-3 is available for a wide range of PC hardware. While the model is compatible with all LOTUS 1-2-3 releases, it is recommended that at least 512k bytes of PC memory be available for use.

The Database Sizing Model user should be familiar with the operation of LOTUS 1-2-3 before using the model. The model, however, is user friendly and requires only an introductory subset of the full LOTUS 1-2-3 capabilities. The model release consists of a single floppy diskette containing the spreadsheet. This section contains accompanying information describing the model and providing instructions for its use.

4.1 Database Sizing Model Overview

The model itself consists of five main sections. Each of these sections is described below.

1. Introduction – The introduction section consists of a title screen, followed by a brief overview of the model and then general instructions for its use.
2. User Data – This section consists of a set of questions to be answered by the BCC to provide information to the model for sizing the SNS databases. By supplying responses to the questions that reflect the SNS system as a whole, the user can obtain an estimate of overall DASD database requirements for the entire SNS system. However, SNS can be configured into partitions, called entities, based on BCC local criteria. Each entity has its own copy of the SNS databases, which contains data relevant to that partition only. In conjunction with the SNS installation process, the

1-2-3 is a registered trademark of Lotus Development Corporation.

questions in this section should be answered on a per-entity basis to obtain proper sizing parameter information for each set of entity databases.

3. Sizing Summary – This section summarizes the SNS database DASD requirements as calculated by the model. All calculations are based on the user data specified in the previous section. The summary consists of two parts. The first part summarizes sizing information for SNS DB2 databases, while the second part is a summary for SNS IMS databases. Separate DB2 and IMS summary screens exist since the sizing information required by each is different.
4. Reference Data – This section contains general reference information, such as database blocking factors and various database overhead factors used in formulas throughout the spreadsheet. This section is provided for general reference purposes only.
5. Sizing Calculations – This section of the model is where the detailed database sizing calculations are actually performed. A separate screen is devoted to each individual SNS IMS database or DB2 table. The spreadsheet formulas take into account not only the SNS data lengths, but also space for system overhead, fragmentation, free space, overflow and growth. This section is primarily provided for reference purposes only. It is possible, however, to use this section for updating various sizing parameters (i.e., amount of free space) on an individual database or table basis. The final calculation results for each database or table are summarized in the summary section of the model, as described previously.

4.2 User Instructions

The SNS Database Sizing Model is distributed on a diskette as a LOTUS 1-2-3 spreadsheet file named "SNSDBR#.WK1" (the "#" indicates the associated release of SNS). The PC that is being used to access the model must have LOTUS 1-2-3 installed on it. LOTUS installation instructions can be found in the *LOTUS 1-2-3 Reference Manual.* Users with a hard disk may first choose to copy the sizing model to a PC working directory prior to use. The model should be accessed via standard LOTUS 1-2-3 file retrieval procedures.

The model consists of a set of screens. The best technique for accessing these screens is by using the "PgDn" and "PgUp" keys to move forward or backward through the spreadsheet. Fields that support user input are highlighted and unprotected. All other fields in the model are protected and may not be changed. All user data may be specified as either integers or decimal numbers. Saving the spreadsheet with user updates is accomplished with standard LOTUS 1-2-3 commands.

The model is distributed with "default" values in all unprotected fields. The default values in the User Data section of the spreadsheet have been included for demonstration purposes and should be modified to reflect actual user information. The default values in the Sizing Calculation section represent recommended settings for various DB2 and IMS sizing parameters. Although these values can be updated, it is recommended that they be updated only with appropriate cause. Whenever a data field is updated by the user, the entire spreadsheet is immediately recalculated.

There are two primary steps in using this spreadsheet:

1. Provide the information requested in the User Data section.
2. Use the Sizing Summary information to update corresponding database sizing parameters and allocations in the SNS installation data files. Information about the various data files involved in the SNS installation process can be found in the *PLATFORM Installation — Installation Planning Guide.*

This process should be repeated for each SNS entity to be created.

Specific sizing information included in the Summary Section includes the following:

1. The space allocation, in kilobytes, required to support each SNS DB2 database. This information should be used to update the corresponding TABLESPACE creation runstreams for each database.
2. The space allocation, in tracks or cylinders, required to support each SNS IMS database. For databases indicated as OSAM, this information should be used in the corresponding database creation Job Control Language (JCL) runstreams. For databases indicated as VSAM, this information should be used in the corresponding cluster definition control cards.
3. The optimal number of root anchor points (RAPs) and the number of blocks required in the root addressable area (RAA) for each IMS database specified as using the HDAM access method. This information should be used to update the corresponding database definitions (DBDs).

All allocations calculated by the model are based on single density 3380 DASD devices. The sizing estimates reflect SNS database requirements only, and do not include storage needed for MVS, IMS, TSO,™ work space, etc.

TSO is a trademark of International Business Machines, Inc.

5. Database Performance Monitoring and Tuning

This section discusses performance monitoring and tuning of the IMS databases and reorganization of the SNS DB2 database.

5.1 SNS IMS Databases

Database performance monitoring and tuning go hand in hand. The most effective way to tune a database system is to evaluate the various performance indicators, decide which indicators are out of line, then adjust the system variables to remove the contention that is causing the indicators to be outside the given range or guideline.

The performance of an IMS system is dependent on many factors, including:

- Hardware configuration
- Operating system resources
- IMS resources
- Maintenance frequency
- Application program design

The selection of database performance variables directly relates to the effective use of system resources and design techniques built into the application software.

5.1.1 Database Control Block Source Distribution

To facilitate the task of database administration in the BCCs, database control block source code is available to the BCC DBA.

The following DBD parameters should be set by the BCC DBA:

- Number of root anchor points per block
- Number of blocks in the root addressable area

If necessary, the following parameters may also be changed:

- Free space factors
- Secondary data set groups

To ensure the highest level of support, all modifications to control block source must be logged and documented for future reference.

5.1.2 Database I/O Tuning Variables

Problems reported as poor IMS response can often be traced to the I/O subsystem. The factors that contribute to the performance of a database are:

- Data set placement
- Free space allocation
- Database buffer pool sizing
- Reorganization type and frequency

To perform the function of database tuning, the BCC DBA should be aware of the variables that can be adjusted to improve database performance in the BCC environment. A brief discussion of each follows.

5.1.2.1 Data Set Placement

Data set placement may be assisted by the hardware performance and capacity planning specialists. These individuals conduct a study of the overall hardware environment and make decisions based on channel, control unit, and device contention. The BCC DBA must be aware of basic factors that influence data set placement.

To achieve maximum performance, the data sets shipped with the SNS application should adhere to the following:

- Data sets should not be placed on channels shared by other CPUs; if this is not possible, the path to the device should be varied offline to the other CPUs.
- SNS databases should always be mounted as "Private" to eliminate contention with other applications.
- Avoid large amounts of allocated free space between data sets.
- Try to isolate SNS databases to control units that have no other activity (e.g., TSO).

Data set placement, like all performance factors, requires constant monitoring. The BCC DBA should use the Resource Measurement Facility (RMF) and IMS Monitor to keep track of the various performance indicators associated with data set placement. Problem indicators that require attention are:

- Over 30 percent channel or device utilization
- Maximum I/O IWAIT times of more than 300 milliseconds
- Average I/O IWAIT times of more than 60 milliseconds

5.1.2.2 Free Space Allocation

The space algorithm used by the Hierarchic Direct Access Methods attempts to place segments inserted after the initial load as close to its root as possible. When database segments are close to other related segments, the I/O time needed to retrieve a dependent segment is shorter. However, as the database grows and the available space decreases, dependent segments are stored increasingly further from their related parent and sibling segments. When this happens, performance problems can only be corrected by reorganizing the database.

The recommendation for the allocation of free space for HDAM databases calls for a 30 percent free space factor. This figure is based on the volatility of the data and the impact a database reorganization would have on daily operations. Since the specification of free space for HDAM databases can cause unexpected results, it is the recommendation of the designers to over-allocate the Root Addressable Area (RAA) and let the randomizer manage the free space utilization. A different free space factor can be used provided it is taken into account for database sizing.

5.1.2.3 Database Reorganization

Reorganizing a database involves changing either its storage allocation or structure. Reorganization processing with IMS is critical for the maintenance of performance and reclamation of fragmented space. The reorganization of databases is an expensive and time-consuming process; therefore, the reasons for and frequency of database reorganization are important factors to understand.

Databases are reorganized for many reasons, including:

- Remove records from HDAM overflow area
- Change the characteristics of the database record
- Recover fragmented free space
- Increase the space allocation
- Reduce the length of twin chains
- Modify the HDAM randomizing parameters

All of the above reasons for database reorganization have one main objective: to reduce the I/O access time required to retrieve database segments.

The frequency of database reorganization varies from one database to another and is highly dependent on the performance objectives established at each BCC location. As a general rule, reorganization should be performed when performance degrades to an unacceptable level because segments in a database record are stored across too many CIs or blocks or roots are out of their home block. This situation is usually detected by monitoring the VSAM catalog or the Volume Table of Contents (VTOC) to determine the presence of either CI or CA splits, the assignment of multiple extents, and excessive EXCP counts caused by long twin chains.

5.1.2.4 Database Buffer Pool Sizing

The objective of adjusting the IMS buffer pools is to reduce the I/O activity required to access the databases. In addition to reducing the I/O incidences, a reduction of potential channel and device contention and the CPU overhead associated with I/O activity is also gained. This is accomplished by varying the pool dimension until the search hit ratio is at its highest point.

5.1.2.5 Performance Measurement Tools

Several performance measurement tools may be used to provide the monitoring capability needed for tuning the variables that affect database performance. These tools, if available, should be used by the BCC DBA to detect and improve database performance. The following provides information regarding tools presently available.

1. Data Language Test (DFSDDLT0)

   On occasion, the need to predict processing time for certain BCC DBA actions (conversions, special runs, etc.) is required. This utility program can perform scans or a specified sequence of calls through databases to approximate run times. The results are not precise, but usually allow reasonable process estimations that have a factual basis.

   Extreme caution must be exercised when using the Data Language Test. The application may expect certain value sets in specific fields and may depend upon certain logical relationships between segments of multiple databases.

2. RMF Analysis

   RMFII (Resource Measurement Facility II) is an OS/VS feature that gathers and records information in order to evaluate system usage. The data collection routines gather several types of information including CPU time, storage usage, I/O configuration and activity, and system wait time.

3. Access Method Services (LISTCAT)

   This facility examines the VSAM catalog contents in order to report the characteristics of each catalog object. The information reported is number of entries, volume serial number, and device type of each volume that contains the cataloged objects, and the space allocation to VSAM data spaces. This information can be used as an aid in tuning VSAM data sets for improvement of I/O operations.

4. Database (DB) Monitor

   Database Monitor is a productivity aid that collects and reports performance-related data during execution of an IMS/VS batch run. A database call summary, database buffer pool and VSAM buffer subpools statistics, IMS-induced I/O wait, and CPU time are performance areas for which detailed reports are produced.

5. Data Communications Monitor

   Data Communications (DC) Monitor is a productivity aid that collects and reports performance-related data during the execution of the IMS/VS data communications process. The report formats and the nature of the information reported are similar to those of the DB monitor reports. System configuration, buffer pool statistics, database call summary, transaction queuing, and timing data by message region and program are performance areas for which detailed reports are produced.

6. IMS/VS Log Transaction Analysis Utility (DFSILTA0)

   The IMS/VS Log Transaction Analysis Utility collects information on each IMS/VS transaction based on records in the IMS/VS system log data set. Total response time, time on input queue, processing time, and time on the output queue are examples of the information reported by DFSILTA0.

7. Database Tools

   The Database Tools package is an IBM product that can be used to analyze reorganization needs. Through reports produced, a user may observe the extent of "overflow," the utilization of free space, the degree of synonym chaining, and the amount of total space utilization.

5.1.2.6 Implementation of Performance Recommendations

Analysis of the SNS application in the BCC environment will probably result in a list of recommended changes to the system. Some recommendations will be easier to implement than others, and some will have a greater impact than others. The more significant bottlenecks must be eliminated to have a noticeable performance improvement.

It is not practical to singularly implement changes and re-monitor to measure the effect, because of time constraints, etc. In general, recommended changes are interdependent on each other for effect.

Once a set of recommendations has been implemented, the DB Monitor and DC Monitor should once again be run and the key performance indicators should be checked again to see if there is a need to analyze the critical resources again.

5.2 SNS DB2 Database Reorganization

Several of the factors that indicate a need for database reorganization are common to both IMS and DB2 databases, including:

- Changing the physical layout of a database record
- Recovering fragmented free space

Additionally, there are reasons for reorganization that are specific to DB2, including:

- Changing the amounts of free space left on database pages or the number of free database pages to interleave
- Improving the Cluster Ratio (the adjacent placement of records with similar keys) for tables with clustering indexes
- Removing records from "Overflow" pages that require more I/Os to fetch a record

All of these factors surely influence the decision on when to reorganize the database. Regardless of any particular factor, the underlying reasons for database reorganization are application performance improvements and more efficient database space utilization. It is the responsibility of the SNS DBA to use all the data available to decide when reorganizations are necessary.

5.2.1 Reorganization Techniques

Each BCC has its own individualized performance objectives and reorganization strategies. Each BCC can therefore set the necessary performance thresholds below which database reorganizations must be performed.

Two tools that a BCC DBA may use in determining whether database inefficiencies are resulting in performance degradation include the RUNSTATS and STOSPACE utilities. The RUNSTATS utility gathers information about the distribution of key data and free space within a table or tablespace. The STOSPACE utility reports on the total space consumption of indexes and tablespaces. This data can be useful in determining if a tablespace is using a large number of secondary extents, which can degrade database performance. See the IBM *Command and Utility Reference* for detailed information on executing the RUNSTATS and STOSPACE utilities.

The decision on which tools to use and how often to run them is subjective, but in general it is a good idea to consider these techniques after a large quantity of update activity has occurred to the SNS DB2 tables.

After a reorganization is performed, the RUNSTATS utility should again be executed and the application plans rebound to allow the DB2 optimizer to make use of the newly generated database statistics in selecting the most efficient SQL access paths for SNS.

6. SNS DB2 Database Security

The DB2 Data Control Language (DCL) GRANT and REVOKE statements are used by SNS to ensure that only AUTH-IDs known to DB2 are allowed to access the SNS database tables. A RACF group name should be selected for the "owner" of the SNS database to ensure easy transitioning of responsibilities between DBA personnel.

Additional RACF groups may be defined for different classes of users, including maintenance users, service negotiators, utility users (Rule Load and Rule Print), and SPUFI users. By using RACF groups, the structure of SNS privileges in the DB2 catalog need not be altered when a USERID or LTERM change is required. The appropriate TSO USERID or LTERM may be added or removed from the RACF group, without issuing any further GRANT or REVOKE statements.

A skeleton for granting SNS DB2 privileges is delivered in the YHGRANT member. This data set should be edited to replace all the site-specific parameters, including RACF group names, TSO USERIDs, and LTERMS where applicable. Each GRANT statement has a comment indicating the potential users of the plans and objects that are affected.

The option exists to grant access to the SNS DB2 objects "TO PUBLIC." This avoids any maintenance of user-lists or RACF groups. However, if the TO PUBLIC option is used, DB2 does not deny access to the SNS DB2 objects to *anyone*. It is therefore strongly recommended to insert TSO USERIDs, LTERMS, or RACF group names in the skeletons where appropriate.

7. Database Back-Up and Recovery

This section discusses database back-up and recovery of the IMS and SNS DB2 databases.

7.1 Online Checkpoint/Restart

The primary technique that IMS/VS uses to record information that can be used to restart interrupted operation is checkpointing. This section describes methods of performing checkpoints and restarts.

7.1.1 Checkpoint

Using the status information recorded during a checkpoint, the content of the message queues and database changes can be restored. Checkpoints are an integral part of system shut-down and start-up.

1. Simple Checkpoints – Simple checkpoints provide a checkpoint, or synchronization, of the IMS/VS system. They are taken automatically by IMS/VS and may also be invoked at any time at the master terminal with the /CHECKPOINT command.

   The frequency of automatic simple checkpoints is determined at IMS/VS system definition time. This frequency is a factor in determining how long it will take to perform an emergency restart of the IMS/VS system. In the absence of any installation standards, a reasonable frequency to use is 30 minutes.

2. Termination Checkpoints – Termination checkpoints are used for orderly termination. They are normally invoked at the master terminal with the following commands:

   a. /CHECKPOINT FREEZE is used for immediate shut-down. I/O messages in transit are allowed to complete, after which communication ceases. Message regions terminate after current messages are processed. All messages waiting to be processed are retained in the message queues, and IMS/VS control information is written to the system log. After a message noting the day and time is transmitted to the master terminal, the IMS/VS control region is terminated. An emergency restart cannot be done from a FREEZE termination checkpoint.

b. /CHECKPOINT DUMPQ writes the contents of the message queues to the system log. An emergency restart can occur for the DUMPQ termination checkpoint. The log volume at this point is synchronized with the status of the IMS/VS queue data sets.

c. /CHECKPOINT PURGE is the most time-consuming shut-down procedure. Every message is processed. Any messages that could not be processed are written to the system log. Following a message to the master terminal, IMS/VS shut-down then takes place. An emergency restart can be done form a PURGE termination checkpoint.

7.1.2 Restart

Restart is driven and controlled by the status of the resources, as left from the previous normal or abnormal online system shut-down. This status is recorded in the Checkpoint-ID table, which is stored in the Restart Data Set. During restart, this table can be examined to determine the type of restart required, and the functions to be performed during restart processing. For a summary of recovery procedures for online IMS system failures, see Table 7-1.

1. Normal Restart – The /NRESTART command is used to bring up the IMS/VS system for the first time or after system definition has produced an updated nucleus (cold start), or to restart IMS/VS after termination checkpoint shut-down. All resources and conditions that existed in the control region at the time it was shut down will be restored if the warm start option of /NRESTART is used, and the action specified with the command matches the exact condition of the prior shut-down.

2. Emergency Restart – The Emergency Restart facilities of IMS/VS are used as an extension of the normal restart to restart IMS/VS after an IMS/VS, OS/VS hardware, or power failure. An emergency restart, using the /ERESTART command, requires a closed system log. The emergency restart process involves resetting each active region back to its last synchronization point, restoring the databases to that point, and restoring the message queues and SPAs to their condition at the time of failure. Each dependent message region active at the point of failure is rescheduled. BMP regions must be manually restarted.

Table 7-1. Online IMS System Failure Recovery Procedures

CONDITIONS	DFSULTR0	EMERGENCY RESTART	REINPUT LOST TRANSACTIONS
System Terminates, Buffers Written		X	
System Terminates with Dump, Buffers Not Written		X	
System Terminates with No Dump, Buffers Not Written		X	X
Log Tape I/O Error	X	X	X

7.2 IMS Database Back-Up and Recovery

This section provides information concerning the availability, use, and functional capabilities of facilities that are used to perform database back-up operations, database recovery operations, and other similar activities. These activities are necessary to maintain the integrity of the databases used in the SNS system.

7.2.1 Back-Up and Recovery Concepts

The BCCs have the option to use the IMS Database Recovery Control (DBRC) Facility to manage control over data set selection and sequencing for any of the following recovery techniques.

When a problem occurs that jeopardizes the integrity of an IMS database, the intent of the recovery process is to correct the error that caused the problem and restore the failed database to a point in time from which normal processing can proceed with the least impact on the user of the database. The IMS/VS back-up and recovery philosophy is based upon creating a copy of a database at a specified point in time and keeping a log of all changes made to the database subsequent to the database copy. Then, if the database is damaged, it can be recovered by restoring the database copy and applying the logged changes to the restored copy.

To be most effective, database utilities that provide back-up and create the input for the recovery process must be executed in an orderly and timely fashion. In addition, whenever database reorganization is performed, the database should be backed up after it is reloaded to preserve a physically changed database. The image copy utility performs that function. The log tapes produced by the system subsequent to the image copy contain the records necessary for forward recovery.

7.2.2 Recovery Methods

The IMS database recovery methods include image copy recovery, forward recovery, and database back-out.

7.2.2.1 Image Copy Recovery

The simplest recovery technique is to restore a database using an image copy. This, of course, represents the database structure prior to any processing steps that may have occurred subsequent to the copy taken.

7.2.2.2 Forward Recovery

When a database failure occurs, the database must be restored. Forward Recovery uses the latest back-up copy (Image Copy) of the database and applies subsequent changes to bring the database up to the point of failure.

The system log contains a record of all database updates, and the database change accumulation utility can be used to sort and accumulate database change records from the log data sets. The Database Recovery Utility can be used to perform recovery and it merges the change accumulation data set into the restored copy and then applies any unaccumulated log data set database changes.

7.2.2.3 Database Back-Out

Database back-out refers to restoring data back to a point in time prior to the processing that was performed upon that data. There are two types of back-out: batch (offline) and dynamic (online).

Batch back-out is performed by a utility and may be used to restore a database back to the first checkpoint of program activity or back to a specified checkpoint (for batch jobs only). Image copies should be taken after online activities have been completed (i.e., prior to batch activities starting) and after the batch activities have completed (i.e., prior to the next day's online activities).

Dynamic back-out, which is performed automatically by the online system (e.g., during emergency restart processing or MPP or BMP program abend), uses the dynamic log and the system log. The dynamic log is a work area, located in core, which contains before and after images of database records that were updated to facilitate dynamic back-out. Dynamic back-out restores the databases that were updated to the last synchronization point (GU to the I/O PCB or the last checkpoint).

7.2.3 Database Back-Up Techniques

This section examines database back-up tools and procedures.

7.2.3.1 The Log Tape

A major tool for back-up and recovery is the IMS system log. A system log volume, which may consist of multiple tape reels, is created during each execution of the IMS control region. It will also be created during batch IMS execution for the SNS system.

Information is placed on the log whenever a database modification is made. This information includes an identification of the database and the data set within that database, an identification of the modified record within the data set, and the contents of the database record before and after it is updated. This information is crucial to database back-up and recovery.

7.2.3.2 Log Integrity

Since the log is the primary tool for maintaining database integrity, mechanisms are provided for maintaining the integrity of the log itself.

1. Dual Logging – Dual logging is the term that describes duplicate logging of IMS change activity simultaneously to two IMS/VS System Log Tapes. In the event of read or write errors, the second copy of the System Log may be used. During log creation, this means that the system can continue using the alternate log data set. However, loss of one of the dual log data sets leaves the IMS/VS system in the same state as the single log operation, and the system should be shut down and restarted as soon as it is convenient in order to resume dual logging. Dual logging is an option that is strongly recommended for the SNS IMS environment.

2. System Log Recovery Utility (DFSULTR0) – The system log recovery utility produces a new and usable system log data set from one that contains unreadable data. For dual log input, the utility reads the primary log onto a new tape. When an error is encountered, it positions the

secondary log to the point of the error and continues by copying the secondary log until an error is encountered. This process continues until a new system log data set is created.

7.2.3.3 Image Copy (DFSUDMP0)

This utility is used to create a copy of a database. Copying is done on a data set basis, which provides the flexibility to copy a single data set within a database or, preferably, all data sets for the entire database. Multiple data sets and databases may be copied in one execution of the utility.

Image copies of databases should be taken as frequently as possible, dependent on update activity, to reduce recovery time in the event of a serious database problem. At a minimum, an image copy must be taken after each database reorganization.

7.2.3.4 IDCAMS

The IDCAMS utility is used to create a back-up copy of data sets with the following two commands:

1. EXPORT copies a VSAM cluster and its catalog entries. This copy can be used as a back-up or can be imported to another OS/VS1 or OS/VS2 system with VSAM facilities.
2. REPRO copies a VSAM cluster (without its catalog entry), a VSAM catalog, or a sequential data set. The copy can be used as a back-up.

7.2.3.5 Database Change Accumulation Utility (DFSUCUM0)

The change accumulation utility sorts and accumulates database change records from the system log data sets, sorting acceptable log records by database sequence, and combining change records that apply to the same physical block or control interval. The output is a sequential data set that can be used by the database recovery utility or as input to another iteration of change accumulation.

The change accumulation utility should be run on a timely basis, consistent with the number of log tapes produced by the IMS system. Database recovery is faster with accumulated change input than with log input only.

The change accumulation utility should use as input only those log tapes produced after an image copy. If multiple copies are produced between database reorganizations, and a change accumulation is done on all log tapes produced since reorganization, the change accumulation will contain unnecessary records. Also, if log tapes produced before and after a reorganization are input to the change accumulation, the resulting change accumulation file will be unusable and would destroy the database if used in a forward recovery. Care must be taken in scheduling the change accumulation utility and in selecting the appropriate files as input to the recovery process.

7.2.4 Summary

The time necessary to recover "broken" databases is of major concern and should be accomplished as quickly as possible.

To facilitate the recovery of SNS databases, the following general recommendations are made:

- The IMS image copy should be used to dump/restore the databases.
- Generation Data Groups should be used to provide multi-copy back-ups and control of database restoration.
- The SNS databases should be image-copied on a daily basis. If this is not feasible, a schedule should be implemented that will provide acceptable recovery time.

7.3 SNS DB2 Database Back-Up and Recovery

To save SNS data from loss and restore it to a useful state, the DB2 recovery process must be initiated. The discussion on recovery that follows should serve as a supplement to the IBM *Database Administration Guide*, which provides a comprehensive treatment of the subject.

7.3.1 DB2 Recovery Concepts

Recovering DB2 tablespaces is a straightforward process with the proper preparation. Full and incremental image copies must be regularly performed to ensure that both consistent and frequent synchronization points`exist from which to initiate the recovery process. Taking frequent back-ups of the SNS DB2 tablespaces minimizes the time involved should a recovery become necessary.

Recovering a tablespace involves three basic operations. First, the tablespace is restored to a previous point in time using a copy of the tablespace taken at that point. Next, incremental changes are applied to the tablespace from data sets that contain only changes to the tablespace that occurred from the point in time established after the first step. Again, these data sets must be created before the time of recovery. Finally, individual changes are applied to the tablespace from the DB2 Log.

After all three phases of the recovery process have been completed, the tablespace has been restored to the last point of consistency (point in the DB2 Log where all units of recovery have been committed or rolled back).

7.3.2 DB2 Recovery Methods

The recovery of DB2 tablespaces may involve several IBM database utilities, including QUIESCE, REPORT, COPY, and RECOVER. QUIESCE, which forces DB2 to take a checkpoint, is useful with some forms of recovery. REPORT may provide information beneficial in determining which DB2 tablespaces need to be recovered. COPY is the primary utility used to take both full and incremental image copies of the data contained in DB2 tablespaces, and RECOVER is the supplied utility to return the tablespace to usefulness.

There are also several methods by which a tablespace may be returned to a prior point of consistency. These include using combinations of QUIESCE with the DSN1COPY and COPY utilities and recovering the tablespace to a Log Relative Byte Address (RBA). However, the recommended strategy (and by far the simplest) involves a series of regularly scheduled full and incremental back-ups of all the SNS DB2 tablespaces using the COPY utility, and the execution of the "RECOVER TABLESPACE" command using the RECOVER utility. This process can be related back to the three phases of recovery described previously:

1. Use the DB2 Catalog to identify and restore the tablespace using the last full image copy.
2. Identify and apply all incremental image copies taken since the last full image copy.
3. Apply all Log records (forward recover) from the last incremental image copy to the most recent prior point of consistency.

These steps would be complicated if performed manually, but DB2 executes all these operations automatically when the RECOVER command is invoked. No user intervention is required unless DB2 cannot find all of the information it needs (or if portions of it are corrupted) to recover the tablespace. After the data in the tablespace is recovered, all of the corresponding indexes must be recovered using the "RECOVER INDEX" command of the RECOVER utility. See the IBM *Command and Utility Reference* for the complete syntax of the RECOVER command.

The above scenario follows the sequence of increasing expense, with respect to the time it takes to recover a tablespace. It is much more expensive to apply log records to a tablespace than to apply an incremental image copy for the same time interval. It is correspondingly more expensive to apply multiple incremental image copies than it is to apply a single full image copy.

However, the order of expense is reversed when you are backing up the tablespace. Therefore, in determining back-up frequency, a site must weigh the time it takes to complete incremental or full image copies against the time the corresponding process would save during a database recovery.

7.3.3 DB2 Database Back-Up Techniques

DB2 database back-up involves taking a "mirror image" view, or image copy, of the SNS DB2 data to establish a point of consistency for the recovery process. The image copy may be either full (a true mirror image of all the data in the tablespace is copied) or incremental (only the changes to the data since the last full or incremental copy are captured). It may therefore require substantially more processing to make a full copy than an incremental copy.

Taking an incremental image copy of a tablespace involves using the "COPY TABLESPACE" command with the "FULL NO" option. To make a full image copy the "FULL NO" option is omitted. The pattern of back-ups should include daily incremental back-ups of the SNS TNLIST DB2 database tables (tablespaces), with regular but less frequent full image copies (at least once a week is initially recommended, or when database update activity is high). The SNS Negotiation Reference and Exception Handler database tables are read-only by the SNS application, and, therefore, a full image copy after initial load is sufficient (unless they are subsequently tailored by the BCC).

If a site determines that there is little or no update activity against a tablespace, the back-up interval may be increased. Recovery and synchronization with other tablespaces is still possible. However, a longer back-up interval requires more forward recovery from the DB2 log than is needed when a suitable image copy exists. The correct choice of back-up interval ensures that the time necessary to save the database and execute the recovery process is kept to a minimum.

Periodic use of the MERGECOPY utility allows multiple incremental image copies to be collected with the last full image copy to create a new full image copy. The MERGECOPY utility works from the DB2 Log and image copy tapes and does not contend for SNS tablespace pages. This process is an alternative to taking full image copies when application availability is at a premium.

For a further description of the SNS tables, see Section 2.

7.3.4 Recovery of SNS DB2 Database Tables

It is necessary to know the semantic or logical relationships that exist among the SNS DB2 tablespaces in order to choose which tablespaces to recover. It is always possible to ensure the semantic integrity of the database by recovering all SNS DB2 tablespaces when any recovery is necessary. However, the SNS DB2 tablespaces may be divided according to subsystem when it comes to recovery. There are currently four SNS DB2 subsystems: Negotiation Reference, SNS Exception Handler, Service Request Output, and TNLIST.

The Exception Handler tablespace need not be recovered when an error occurs in a TNLIST tablespace, and vice versa. Similarly, there is no need to recover the TNLIST tablespaces when an error occurs in a negotiation reference data tablespace. However, the logical relationships between the SNS DB2 database tables of a particular subsystem require that all tablespaces for that subsystem be recovered. (For more information on the database tables and tablespaces comprising each SNS DB2 subsystem, see Section 2 in this document.)

For example, there are many key fields that "relate" one SNS TNLIST table to the next, and, therefore, the need to recover individual tablespaces cascades. The Customer to Centrex Group table is related to the Centrex Group table via the MAIN_TN attribute. The Centrex Group table is related to the Group List table again via the MAIN_TN attribute. The Group List table is related to the Centrex TN List table via the LIST_ID attribute. It is, therefore, unlikely that the recovery of any individual TNLIST tablespace would leave the database in a semantically correct state (although physically correct).

8. Database Reporting Tools

This section reviews the reporting tools available for the SNS IMS and SNS DB2 databases.

8.1 SNS IMS Databases

The Rule Unload Facility can be used to generate a report that lists all of the rules currently in the SNS Rules database. This facility is discussed in the "Verifying the Rules Database Load" section of the *Service Negotiation Support (SNS) System Administration Guide.* To generate the report, edit the Rule Unload Exec Deck (YHDAU01X in the JCL Library) and change the "A" control card value to "U". The control card should read as follows:

```
A U;
```

The resulting data set will contain a formatted rule listing. The Rule Unload Facility will not generate output to a printer. The data set containing the Rule listing must be manually printed.

8.2 SNS DB2 Databases

DB2 provides several methods for reporting on the information contained in its tables. The primary method available to an SNS DBA is the use of the SPUFI processor found in the DB2 Interactive (DB2I) panels. For complete documentation on the SPUFI processor and DB2I, see the IBM *SQL Usage Guide*.

Through SPUFI, an input file containing many SQL requests may be processed at once. The output goes to the SPUFI output data set. A data set named YHREPORT is delivered with the SNS application. This data set contains sample SQL requests that may be used to report on the SNS DB2 tables. The runstream contains instructions for replacing the various site-specific parameters so that queries can run properly. The runstream may be split into separate data sets, containing one report each, and they may be run individually. These cover various levels of detail.

9. Database Repair

This section highlights important aspects of database repair for the SNS IMS and SNS DB2 databases and references documents in which additional information can be found.

9.1 SNS IMS Databases

The IBM-supplied DFSDDLT0 program is capable of inserting, updating, and deleting segments from any IMS database. However, extreme caution must be exercised. The application may expect certain value sets in specific fields and may depend upon certain logical relationships between segments of multiple databases.

Bad pointers in IMS databases may be caused by failure to run database back-out, failure to perform emergency restart, or failure to include a log tape during recovery or back-out. A bad pointer can normally be corrected by recovery. In certain instances, bad pointers can be corrected via reorganization. Such instances are documented in Appendix F of the *IMS/VS Data Base Administration Guide.*

9.2 SNS DB2 Databases

SNS DB2 databases may be repaired using the IBM-supplied REPAIR utility. If a database utility or transaction indicates a failure with a table, the procedures for database repair outlined in the IBM *DB2 Database Administration Guide* should be followed.

There are many different kinds of corruption that may occur to a DB2 table or database page, including both physical and logical, which may each require a different strategy. Database repair should only be undertaken by a well-trained DB2 database administrator.

A. Appendix A — Guide to Acronyms

AIN	Advanced Intelligent Network
BAE	Bellcore Application Environment
BCC	Bellcore Client Company
CAT	Centrex Access Treatment
CCF	Customized Calling Feature
CMT	Communication Terminal Table
CPU	Central Processing Unit
DAR	Data Access Routines
DASD	Direct Access Store Device
DBA	Database Administrator
DBD	Database Definition
DBMS	Database Management System
DBRC	Database Recovery Control
DBRM	Database Request Modules
DCL	Data Control Language
DDL	Data Definition Language
DPS	Display Processing System
FC	Functional Component
FCIF	Flexible Computer Interface Form
FID	Field Identifier
GOT	Generic Output Translator
GTS	Generic Table System
HDAM	Hierarchic Direct Access Method
IMP	Immediate Message Poster
IMS	Information Management System
ISDN	Integrated Services Digital Network
I/O	Input/Output
JCL	Job Control Language
LCC	Line Class Code
LFACS	Loop Facilities Assignment and Control System
MVP	Multi-Variety Package
NMAG	Network Marketing Area Guide
ORT	Output Routing Table
PLBB	Processing Layer Building Block
PREMIS	Premises Information System
PSB	Program Specification Block
RAA	Root Addressable Area

RAP	Root Anchor Point
RBA	Relative Byte Address
RCU	Recent Change USOC
RMF	Resource Measurement Facility
SAC	Service Activation Controller
SNS	Service Negotiation Support
SOE	Standard Operating Environment
SOP	Service Order Processor
SQL	Structured Query Language
STOGRP	Storage Group
TCIS	Telecommunications Interface System
TN	Telephone Number
TNLIST	Telephone Number List
TTS	TIRKS Table System
ULBB	User Layer Building Block
USOC	Universal Service Order Code
VSAM	Virtual Sequential Access Method
VTOC	Volume Table of Contents