
System 1032®

User's Guide

System 1032 Version 9.00

CompuServe
Data Technologies

1 Managing System 1032 Catalogs

Overview

This chapter describes the steps for creating System 1032 catalogs. A catalog is a dataset, database, or library. Specifically, this chapter tells you how you can:

- Create a dataset, both interactively and from a user definition file.
- Define datasets within a database, either interactively or from external description files.
- Create databases and libraries.

This chapter assumes that you are familiar with the System 1032 commands needed to create a simple dataset and to examine and update datasets. You should also be familiar with the System 1032 concepts described in Module 1.

To design datasets, first determine how their information should be organized: how many attributes will be in each dataset and what information will be in each one. The CREATE command sets up the dataset according to your dataset definition.

After you set up the dataset, you can then *populate* it with data. If your data is stored in a Record Management Services (RMS) file, use the LOAD command to transfer, move, or copy the data from the file into the System 1032 dataset. You can also populate a dataset by using the ADD command to type the data directly into the dataset.

Datasets are the basic building blocks of databases. This chapter describes the structure of a dataset, then outlines two ways you can create them. It also refers to the commands that you use to define the items in a dataset or database. In most cases, these commands are fully described in the *System 1032 Programmer's Reference Manual*.

Basic Dataset Structure

A System 1032 dataset is the basic unit for accessing and manipulating data. You can define a dataset independently or as part of a database. However, you can always access a dataset independently, even when it is defined as part of a database.

Figure 1-1 illustrates the contents of a dataset, showing, from top to bottom, the order of items as they are often defined.

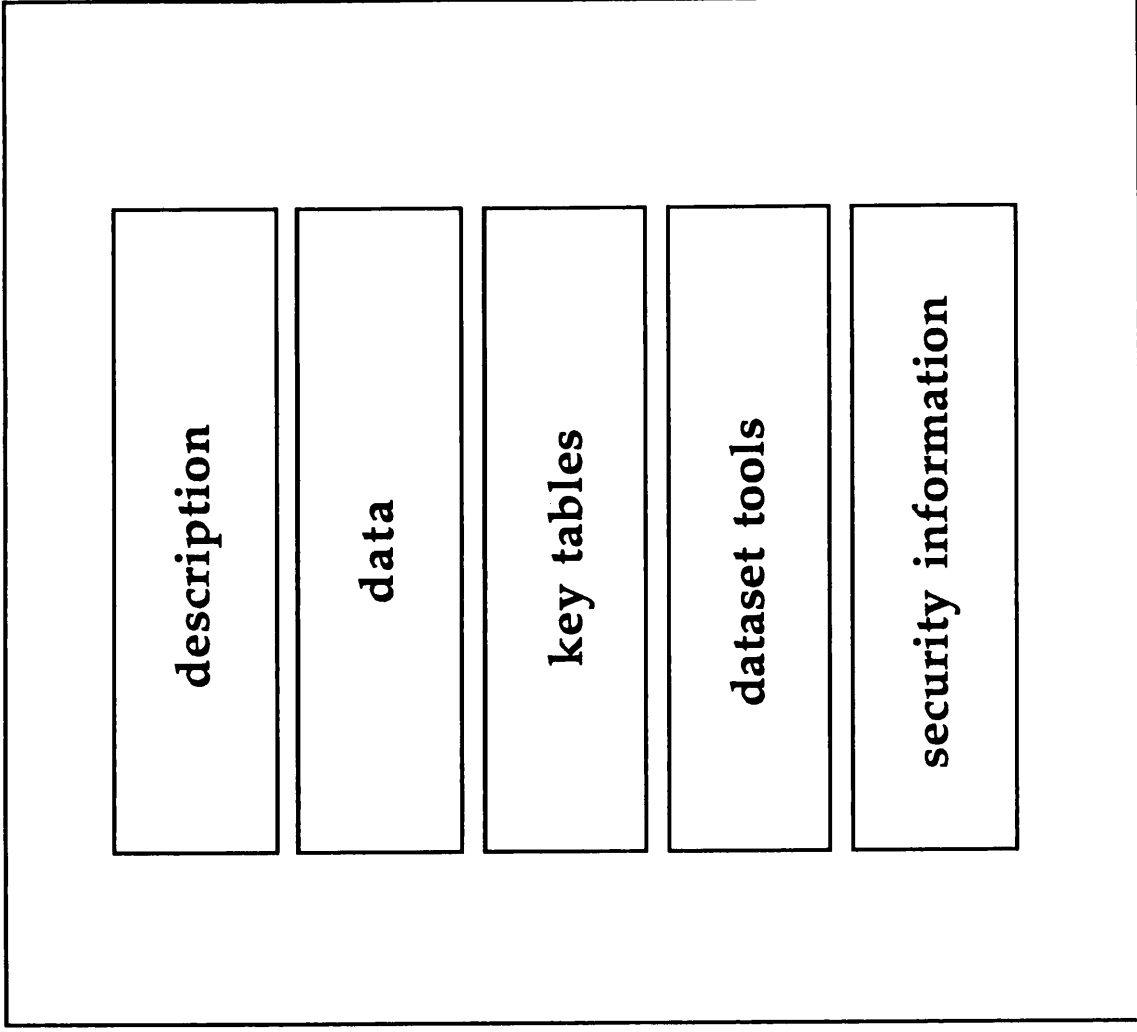


Figure 1-1 System 1032 Dataset

The dataset record template is a group of attributes that define the structure of the records in a dataset. Each attribute describes one item of data that occurs in every record in the dataset. The record template specifies what kinds of data can be stored in the dataset and how the user can manipulate the data. The template specifies the name, data type, and number of attributes in the dataset.

The records contain the data corresponding to the attributes defined in the record template. Each record is identified by a unique \$ID, a system attribute used internally as the record index.

System 1032 uses key tables to determine which records contain a particular value or a set of values. A key table contains all values for one keyed attribute and identifies the records (using the \$IDs) that contain each value. For example, System 1032 uses the key tables to find data to fill a selection set.

Creating Datasets

A System 1032 dataset may contain records, key tables, tools, and security information. When you define the dataset, you can specify the files used to store these items. If you do not specify individual files for these items, System 1032 stores all parts of the dataset in one file.

Dataset creation is a two-step process:

- Define the structure of the dataset.
- Load the data into the dataset.

After you set up the dataset with the CREATE DATASET command, you are ready to enter data into the dataset.

Use the INSERT and REMOVE commands to add or remove definitions of dataset tools, including forms, procedures, RDs, and variables. For more information, refer to the *System 1032 Programmer's Reference Manual*.

The Dataset Record Attributes

When you create a dataset, you are defining the record attributes, which are comprised of ATTRIBUTE commands. Each ATTRIBUTE command specifies the qualities of an attribute, such as its name, data type, external length, key status, print defaults, prompts, and comments.

You can also provide additional information about the attribute: load defaults, key defaults, RDs, procedures, forms, and variables.

In addition, you can also use the ATTRIBUTE command to describe the length or column position of external data in a data file.

Before you create record attributes, decide the following:

- How many attributes (that is, what kind of information) the dataset will have.
- What kind of data you already have.
- What data you want to add in the future.
- How you want to use the information.

While this chapter provides basic information about dataset creation, subsequent chapters describe more specialized operations. Table 1-1 describes the types of information you can define in a dataset, lists the related commands, and tells where in this module you can find detailed information. For more information, refer to the ATTRIBUTE command in the *System 1032 Programmer's Reference Manual*.

Table 1-1 Information Types Allowed in Datasets

Information Type	Related Commands	Command Description	Additional Information
Attribute definition	ATTRIBUTE	Specifies the qualities of an attribute, such as name, data type, and length.	Chapter 3-1
Dataset name and comment	DATASET	Specifies the dataset name and comment in a dataset definition.	Chapter 3-1
External data commands	RD, LOAD, APPEND, DUMP	These commands include record descriptors (RD), variables, procedures, and forms. Use the RD command to describe the format of an external record. System 1032 uses a record descriptor in the LOAD, APPEND, and DUMP commands and for communicating with the host language interface.	Chapter 3-2
Forms			
Journaling	JOURNAL	Establishes a journal file for the dataset to be used during recovery in case of dataset damage. You can specify a name for the journal file.	Chapter 3-1, 3-4
Keying attributes	KEY_DEFAULTS	Data definition command. KEY_DEFAULTS sets defaults used for key table creation and controls some efficiency parameters that you cannot control with KEY. In the KEY_DEFAULTS command, you can specify the files used to store the key tables.	Chapter 3-1
Loading default parameters	LOAD_DEFAULTS	Data definition command. LOAD_DEFAULTS sets the defaults for the LOAD, APPEND, DUMP, ADD, DM_ADD_REC, and DM_ADD_SET commands and controls some efficiency parameters that you cannot specify with those commands. In the LOAD_DEFAULTS command, you can specify the file used to store the records.	Chapter 3-2
Procedures, variables	PROCEDURE, VARIABLE	PROCEDURE defines a compiled or external procedure that can consist of PL1032 or other high-level language. VARIABLE defines a command repository in which you can temporarily place values.	Module 2
Record security	RECORD_SECURITY	Establishes defaults for reevaluating user access to records when the PREVENT RECORDS command is in effect.	Chapter 3-3.

Naming Attributes

Attribute names must begin with a letter and can include only letters, numbers, and underscores. Embedded spaces are not allowed. For example, if you want to use a name

with two or more words in it, separate the words with an underscore, such as LAST_NAME. The maximum length allowed is 127 characters.

Choose a name that reflects the content of the attribute, to help others using the dataset. For example, when a user gives the SHOW ATTRIBUTE command to see the attributes in the dataset, LAST_NAME is self-explanatory while ATTR3 is not. Specify a name that is unique for the dataset and avoid using System 1032 keywords as names. Also, use a short name to ensure more efficient data handling by key tables and forms.

An ATTRIBUTE command assigns one primary name to one attribute. This primary name is used as the default column title by the PRINT command and as the default prompt text by the ADD command.

You can also assign one or more secondary names to the attribute. While a primary name describes the attribute, the secondary name can be an abbreviation for the primary name. For example, LN is a useful abbreviation for LAST_NAME. You can use the primary and secondary names interchangeably. For example, use the OR option to assign one or more secondary names to the attribute.

```
DDL_ ATTRIBUTE FILM_TITLE OR FILM OR NAME TEXT 35
```

The attribute FILM_TITLE has two secondary names: FILM and NAME. TEXT is the data type and 35 characters is the assigned length of the attribute.

Defining Two or More Attributes at Once

You can define two or more attributes that share the same data type and qualities by using one ATTRIBUTE command.

To define more than one attribute, separate the names with a comma and repeat for any number of attributes.

```
DDL_ ATTRIBUTE SURNAME OR LAST_NAME, FIRST_NAME -  
DDL_ TEXT 15 KEYED
```

This command defines two separate attributes; the first attribute also has a secondary name.

Like all dataset qualities, all attributes defined with the ATTRIBUTE command share the same comment.

Note that you cannot define multiple attributes with one ATTRIBUTE command if the attribute has a group data type or is an element of a group (see the Language chapter of the *System 1032 Programmer's Reference Manual*).

Specifying Data Types

After you've named the attribute, you must specify its data type. The data type determines what kind of value the attribute can have, such as text characters (TEXT) or whole numbers (INTEGER). The data type determines what kind of operations you can perform with the values. For example, you can print a total for integers but not for text.

Attribute Qualities and Types

After you specify the basic data type, you can use any combination of attribute qualities in any order. The type qualities that you can use, such as MISSING and PRECISION,

are valid only for certain data types. You can use attribute qualities such as COMMENT and PROMPT for any attribute. Other qualities you can define, such as LENGTH and COLUMN, are discussed in later chapters.

If you are using one command to define more than one attribute, do not specify different type or attribute qualities for individual attributes.

The following sections define several common attribute qualities and types. For more information, see the Data Types chapter of the *System 1032 Programmer's Reference Manual*.

Associating a Comment with an Attribute

Use the COMMENT option to specify a comment that is permanently associated with the attribute. The SHOW ATTRIBUTE COMMENT command displays the attribute comment. This provides very useful information for the dataset users. The attribute comment can be any series of characters, up to the maximum of 32,767.

```
DDL_ ATTRIBUTE DIRECTOR_ID INTEGER KEYED COMMENT -  
DDL_ "Used for mapping to DIRECTORS"
```

Preventing the Insertion of Blank Data

If you want to prevent a user from entering a missing value during data entry, specify MISSING PROHIBIT in the ATTRIBUTE command along with the INITIALLY clause. In the following example, if the user presses <Return>, the initial value of TRUE will be entered into the dataset. If MISSING PROHIBIT were not part of the attribute definition, System 1032 would enter a MISSING value when the user pressed <Return>.

```
DDL_ ATTRIBUTE COLOR LOGICAL TITLE CENTERED "Film in/Color" -  
DDL_ FORMAT L PROMPT "Is film in color " INITIALLY TRUE -  
DDL_ MISSING PROHIBITED
```

Controlling Prompts for Data Entry

You can control whether or not the user is prompted for the attribute while entering data and if so, what prompt System 1032 uses. In addition, you can specify that the user must provide a value for the attribute.

If you design the attribute with a prompt, the prompt answers when the user types the ADD command, followed by a <Return>.

By default, if the ADD command does not specify an attribute and a value for the attribute, System 1032 prompts with the primary name of the attribute. The PROMPT option allows you to specify the prompt text that is used in place of the attribute primary name. PROMPT is also used in generating default forms, using the FIELD COPY command. A prompt can be any text constant up to 80 characters.

```
DDL_ ATTRIBUTE FILM_TITLE TEXT 35 PROMPT "Title of Film"
```

The above example defines the attribute FILM_TITLE and specifies a prompt.

The prompt "Title of Film" appears and the user must specify the film title. The user then types the title and presses <Return> to move to the next entry. If FILM_TITLE is defined as a DEMAND attribute, and the user enters no value, he or she receives a warning message and is re-prompted. If the FILM_TITLE is *not* a DEMAND attribute

and the user enters no value, System 1032 enters a missing attribute. However, if the attribute is defined with an initial value, that value appears at the prompt and is added if the user presses <Return>.

For attributes of all data types except logical, System 1032 prints a colon and a space after the prompt. For logical attributes, System 1032 prints a question mark instead of a colon. Keep this in mind when defining the prompt text for logical attributes.

If you want to require that the user enter a value while adding data, specify the DEMAND option. If the user types <Return> at the prompt, System 1032 displays the message "value is required" and repeats the prompt. If you specified SET MESSAGE NO PRINT, no message is displayed, but System 1032 displays the prompt each time a user presses <Return>.

```
DDL_ ATTRIBUTE FILM_TITLE TEXT 35 -  
DDL_ KEYED TITLE "FILM TITLE" PROMPT "TITLE of FILM" -  
DDL_ DEMAND
```

In this case, the user is prompted for a FILM_TITLE. The DEMAND option specifies that the user cannot add a FILMS record without typing the name of the film, which must comprise at least one character.

Keying Attributes for Rapid Access

This section describes the two operations that key attributes during dataset creation: the KEYED option and the KEY_DEFAULTS Data Description Language (DDL) command.

A key table is an index to records in the dataset with the same attribute values and corresponding indexes. It indicates which values are associated with which particular records. System 1032 uses key tables to locate records rapidly in a database. When you issue a FIND, DFIND, or MAP command, System 1032 uses the key table to point to records that contain a particular value or set of values.

Other commands rely on key tables. The SORT command can sort more efficiently if an attribute is keyed. The VALUES command also requires key tables to produce an efficient summary of the values for an attribute in a selection set. The JOIN command requires keyed attributes. The number of keyed attributes in a dataset also affects the efficiency of ADD, CHANGE, LOAD, and APPEND. Figure 1-2 illustrates a key table of the RELEASE_YEAR attribute in the FILMS datasets.

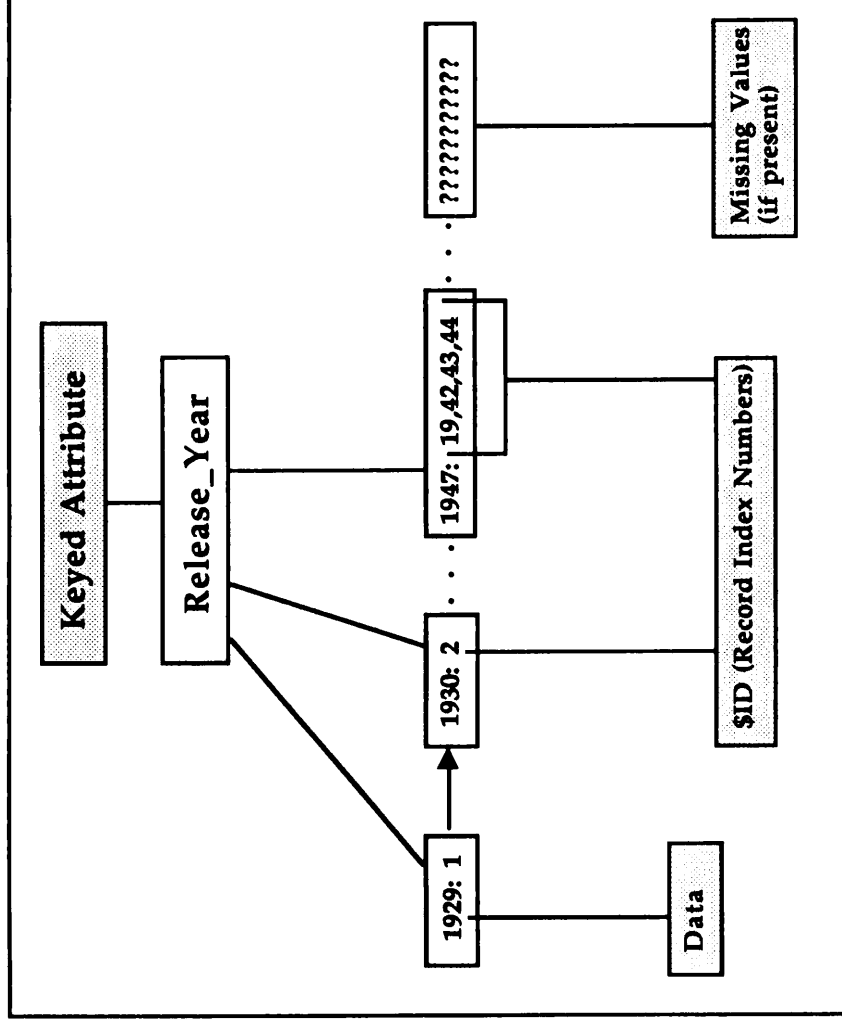


Figure 1-2 Typical Key Table

In the above figure, the RELEASE_YEAR attribute is keyed. The key table contains an index of all the records in which the names 1929, 1930, and 1947 appear. For example, if you search for all the records that contain 1947, System 1032 examines the key tables and selects record IDs 19, 42, 43, and 44. If there were missing values within this attribute, they would appear at the end.

A key table is part of a dataset. Depending on how you set up the dataset, the key table is stored in the database file (<filename>.DMB), the dataset file (<filename>.DMS) or in a special key file (<filename>.DMK). The amount of space taken up by a key table depends on:

- The number of records in the dataset.
- The number of different values in the particular attribute.
- The amount of null space built into the keys. For more information, refer to the KEY_DEFAULTS command in the *System 1032 Programmer's Reference Manual*.
- The length of the primary attribute name.
- The size of the attribute values.

You can control the keys for a dataset during both dataset creation and dataset maintenance. However, this chapter discusses only dataset creation. The chapter on maintenance tells how to use the two key commands that apply to dataset maintenance: KEY and UNKEY.