# Metadata Management for Large Statistical Databases

John L. McCarthy

Lawrence Berkeley Laboratory
University of California
Berkeley, California 94720

**Abstract**

Data description or *metadata* presents a significant database management challenge, particularly for scientific and statistical databases. Ideally, we would like to access and manipulate data and metadata using the same DBMS tools, but there are few systems that even begin to provide such integrated capabilities. This paper outlines a framework for more integrated metadata management by synthesizing ideas from statistical analysis, bibliographic retrieval, data dictionary, and database management systems. Drawing on experience and examples from a large statistical database project, the paper discusses and analyzes:

- general types and uses of data about data
- special types of metadata for statistical databases
- metadata structure and characteristics
- principles and requirements for metadata management

## 1. Introduction

As databases continue to grow in number, size, and complexity, database management researchers, system implementors, and users have recognized a need for more detailed data description or *metadata* to provide systematic information for end-users, database administrators, application programs, and database management software. They have also noted the desirability of integrating metadata facilities, such as data dictionaries, with database management systems [CODD82, CURT81, MEYE81]. An "active," integrated data dictionary could provide information such as attribute names and characteristics, security requirements, etc., for the DBMS as well as for users and application programs. The DBMS could in turn manage metadata by treating the data dictionary as a database.

Scientific and statistical databases [TEIT77, CHAN81, BORA82, SHOS82] share this need for integrated metadata management. In addition, they require metadata that are not yet found even in specialized data dictionary systems. They also need manipulation and analysis routines that automatically use and produce *self-describing* data files [BECK78, BURN82]. Self-describing data files facilitate use of data with external statistical analysis programs. They are essential in an interactive environment where the output of any routine can immediately serve as input to another.

At present, few data management systems provide even rudimentary integrated facilities for metadata management. Even fewer provide facilities for statistical metadata. As a result, most scientific and statistical users still rely primarily on special purpose programs and statistical analysis packages, which have only limited data and metadata management capabilities [MCCA79].

Most relational systems treat metadata as data, and relational operations produce metadata as well as data. But metadata in relational systems are currently quite limited [CODD82, KILO81]. With the notable exception of SPIRES [SCHR75], most other systems that support a richer variety of metadata do so via separate and less flexible facilities for metadata management.

This paper addresses general issues of metadata management as well as the special problems of scientific and statistical metadata. Section 2 sets the context by summarizing some central metadata issues of an experimental system for large statistical databases. Section 3 briefly reviews what the concept of *metadata* includes, the major *objects* to which metadata pertains, and the general *types* of metadata that characterize most databases. Section 4 discusses statistical metadata, and Section 5 summarizes the *uses* to which various types of metadata can be put. Section 6 compares *data structures, types of access*, and *update patterns* that typically characterize metadata in contrast to data. Based on the analysis of sections 3 through 6, sections 7 and 8 propose some general principles plus *structural, definitional*, and *functional* requirements for integrated metadata management.

## 2. Metadata Issues and the SEEDIS Project

SEEDIS, an interactive Social, Economic, Environmental, and Demographic Information System at the Lawrence Berkeley Laboratory [MCCA82B, COMP82], illustrates some of the significant metadata management problems that arise for large scientific and statistical databases. With databases containing 2.5 billion data values for a million different data elements,[1] and covering over eighty types of geographic entities, metadata is an important part of SEEDIS.

As with most large databases, SEEDIS users need a variety of printed and on-line metadata to find out what data and tools are available, to understand and interpret the data, to specify retrieval requests, and to automatically label output displays. SEEDIS database administrators need metadata tools to describe new data, to map logical to physical storage locations, and to provide links between related entities, attributes, and databases.

[1] Data elements are attributes or variables such as the number of unemployed persons, or individual cells of an aggregate summary table such as population by age, race, and sex. Data values are the numeric values of data elements for individual instances of entities such as states, countries, or households.

SEEDIS data management software as well as analysis and display programs require information on physical structure and file locations, data element types, etc. to validate input, to maintain data integrity and independence, and to carry out high-level user and DBA requests.

In SEEDIS, as in advanced statistical systems, much of the metadata is located with the data in self-describing files. Manipulation and analysis routines that automatically use and produce self-describing files free the user from having to re-specify data parameters, labels, and the like as s/he proceeds from creation of a working dataset through various phases of analysis [BECK78, MERR81, BURN82]. Tools that use and produce self-describing data files are essential in an interactive environment, where one needs to track and document the manipulation and analysis process through many intermediate steps.

SEEDIS includes substantial metadata facilities. But they are not sufficiently integrated with data management or with one another. SEEDIS users and programs must create, access, and maintain various types of metadata (e.g., database schemas, data dictionaries, and system documentation) independently, and they cannot use the same tools to query, retrieve, and update both data and metadata.

Experience with SEEDIS suggests that integrated metadata management is difficult because metadata differ significantly from data, particularly statistical data. It is also difficult because SEEDIS, like more general data management systems, lacks sufficiently general data structures, definition language, and manipulation tools to manage metadata as well as data. In order to overcome these limitations, we need a better understanding of metadata, its characteristics, and its uses. We need to analyze metadata entities, attributes, and relationships as database design problems in their own right. We need to examine the basic characteristics of these metadata entities and attributes in terms of typical data types, structures, access, and update patterns. Only then can we begin to suggest specific strategies for integrating metadata and data management.

## 3. What is metadata?

Metadata is data about data -- that is, systematic descriptive information about data content and organization that can be retrieved, manipulated, and displayed in various ways. Metadata may be simple and unstructured, such as a typewritten narrative describing a data tape, or structured and complex, such as an active machine-readable DBMS dictionary used to control multiple databases.

The distinction between metadata and data is not always a clear one. For example, consider a table of population counts by age, race, and sex. In one sense, age, race, and sex categories are data values that characterized the individuals summarized in the table. In another sense, they are metadata that serve as labels for cross- product cells of a three dimensional table. From a relational perspective, they may be composite keys of a single "population" attribute. As Smith and Smith have noted,

> *relationship, entity, component, category, attribute, and instance are just different interpretations of the same abstract objects.* [SMIT78]

Nevertheless, these less abstract concepts are useful to help differentiate and structure metadata information at a logical level. Since researchers and database system developers have not yet agreed on standard terminology, and since some readers may not be thoroughly familiar with such concepts, the following two subsections briefly summarize major types of objects to which metadata may pertain, and general types of information that metadata may include.

### 3.1. What Objects Does Metadata Describe?

Metadata can pertain to many different types of objects and logical levels of data abstraction. Current database management, data dictionary, and statistical analysis systems commonly include at least some minimal metadata for the following types of metadata entities, among others:

*Entities (Record-types, Summary Levels)*

Entities are objects, events, or relationships to which data pertain. [CHEN76] (Hence we refer to the term *entity* itself as one type of *metadata entity -- i.e., a type of object to which metadata pertains)*

*Attributes (Data Elements, Data Items, Fields, Variables)*

Elementary attributes are individual atoms of information within a database, such as population counts, book titles, or machine part codes. Complex attributes are sets of two or more elementary attributes that always occur together in a unit. Vectors (e.g., time series), multi-dimensional arrays (e.g., tables), and other ordered sets are complex attributes that frequently occur in statistical data.

*Category Sets (Dimensions, Value Label Sets)*

Category sets are structured lists of names and associated information which pertain to a particular dimensions or categorical attributes. Such lists may or may not be mutually exclusive and exhaustive. For example, a simple category set called "sex" might consist of the two categories "male" and "female". Most statistical packages provide facilities for category sets; most data management systems do not [HAMM82, BURE80, DATA81].

*Databases (Files, Datasets) and Database Collections*

All the instances for all the attributes of one or more types of entities that are all related or treated together (either logically or physically) are usually referred to as a database. Database collections are groupings of databases.

### 3.2. What Information Does Metadata Contain?

In theory, if not in practice, each type of metadata entity described above may have various different metadata attributes. Some important categories of metadata attributes for both general and statistical databases are as follows:

*Names and Aliases*

Names are unique identifiers for metadata entities. Aliases or synonyms are alternative identifiers that can be used in place of main names.

*Labeling and Descriptive Information*

Description to supplement names and aliases may include a short label, one-line title, unlimited textual description, subject index terms, footnotes, and special remarks.

*Data Derivation and Quality*

Information about the derivation and quality of particular fields or attributes may include creation and modification procedures and history, source citations, and reliability estimates.

*Security Specifications*

Security specifications designate what users or programs can read, add, delete, or modify different types of information.

*Logical Structure Description*

Structural information may include multiple logical views for different users and descriptions of complex data types, such as arrays. For statistical databases, it is important that such representation extend to *access and manipulation* of entire structural units as objects (e.g., vectors, matrices, and non-homogeneous data structures such as analysis of variance tables) [BECK78, MCCA79, KLEN81].

*Access Path and and Linkage Specification*

In addition to structural characteristics of individual databases and attributes, metadata can also contain descriptions of how different entities are linked, (e.g., which attributes determine the unique key of a particular record-type, or what access path represents a particular many-to-many relationship).

*Processing Procedures*

Processing procedures for individual fields or attributes include simple recoding, transformation, and checking of data on input found in most data management and statistical packages. SPIRES provides an even more powerful and extensive set of standard data item manipulation functions for input, output, indexing and other operations [SCHR75, SPIR82].

*Usage Information*

Usage information may include which programs access, update, or control which metadata entities, and vice-versa, summary usage statistics such as number of accesses, etc.

*Physical Characteristics*

To facilitate data independence, physical characteristics such as disk dataset names, block sizes, compression algorithms, etc., need to be identified as separate types of metadata.

## 4. Statistical Metadata

In addition to the basic types of metadata outlined above, scientific and statistical databases require special types of metadata to describe statistical characteristics, and to provide information for data manipulation and analysis software. Statistical systems, not surprisingly, provide for more statistical metadata than do database management systems. But there are no standards for such metadata, and few statistical systems use it to the extent that they could.

Statistical metadata typically includes default specifications at the database level (e.g., global default missing data codes), which apply to all data elements, as well as detailed specifications at the individual element level. As in artificial intelligence languages such as LISP, individual attributes inherit the global default values (such as missing data codes) or can over-ride them with element level equivalents.

Since statistical characteristics may not be as familiar to readers of this paper as other types of metadata, this section enumerates some specific examples, with references to systems that include such information and papers that discuss them in more detail. Many statistical analysis programs make active use of certain types of metadata for case selection, type-checking, error calculation, etc., and such programs will probably use an increasing variety of metadata in the future.

### 4.1. Statistical Metadata Expressions

Many types of statistical metadata can be described by an arithmetic or logical *expression*, which may contain names of other attributes -- e.g., weight = 1000 * (area_size/population_size). If they are kept in a standard form, database systems and analysis programs, as well as people, can read and use such expressions for a variety of purposes. Some specific examples of statistical metadata that can take the general form of expressions include:

*weighting expression*

weighting factor to be applied to individual instance values to take into account such factors as scaling of values to save storage space, or disproportionate stratified sampling in survey data [NIE75, BUHL79, SAS79, INST73, ROBI80].

*error expression*

estimated error for individual data values. In the case of sample data, this might be a relatively complicated mathematical formula involving other data elements [SPAR82, KLEN81].

*aggregation or disaggregation expression*

an equation to be used for aggregating or disaggregating data to a different level of analysis (e.g., from census tracts to air quality districts) [MERR82].

*suppression expression*

specification of a particular form of data suppression (e.g., when there are fewer than 15 cases) to protect individual or corporate privacy [DATA81B].

*variable creation history expression*

transcript of arithmetic and logical operations used to create derived or "virtual" attributes [STEW78, SHAN80].

### 4.2. Other Statistical Metadata

Other types of statistical metadata do not take the form of expressions, but can be used in an active way by statistical analysis routines if they are expressed in standard, controlled forms. Some specific examples include the following:

*measurement units*

a restricted vocabulary keyword identifying the units of measurement (e.g., gallons, miles, count of persons, etc.), which can be used by computational routines to provide automatic conversion (e.g. miles to kilometers), and extended type-checking when performing operations involving different attributes [SPAR82].

*missing data specification*

a valid value range or list of missing data codes used to select valid cases and to flag conditional processing procedures in certain statistical routines. [NIE75, BUHL79, SAS79, ROBI80].

*data quality indicator*

a categorical variable indicating relative reliability of data values, which can be used for weighting, case selection, and other purposes, [STEW78, KLEN81].

*universe*

in aggregate census and other types of summary data, different attributes within a single data record may pertain to different underlying universes or populations of entities (e.g., "total population," or "Hispanics over 18") [DATA81B] -- not to be confused with the entity or record-type to which the aggregated *summary* data pertain (e.g., a *geographic* unit such as tract, county, or state) *value label (category) set*

a group of descriptions, labels, and codes, each of which is associated with a particular attribute value or range of values (e.g., "0-6999 = low" "7000-19999 = medium" "over 20000 = high"), [NIE75, BUHL79, SAS79, ROBI80, INST73]. In certain cases, such information can be used to aid automatic data conversion between different category sets or from one logical structure to another.

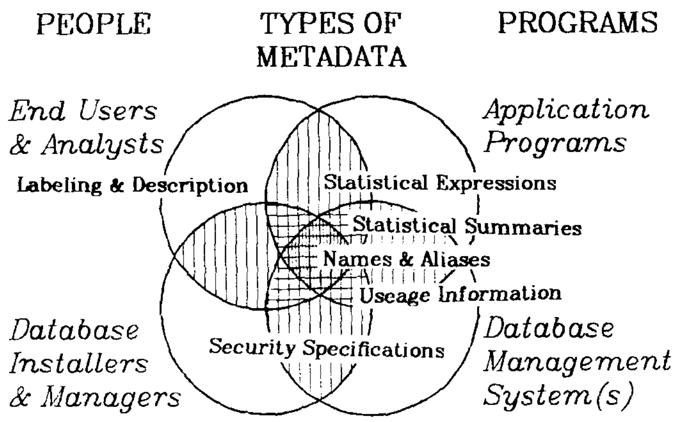
*statistical summary data*

univariate statistical summaries for an attribute (e.g., median, selected percentiles, range, count of non-missing data values, count of unique values), and multivariate summaries of relationships (e.g., correlation coefficients). If data are not relatively static, this involves non-trivial update considerations [ROWE82, BORA82].

## 5. How Is Metadata Used?

The primary purposes of metadata, whether used by people or programs, are to *locate*, *define* and *control* the data to which they pertain. The various classes of metadata outlined above can be used in a variety of different but overlapping ways by end users, database administrators, application programs and database systems. Some types of metadata such as physical and security specifications usually are available only to database administrators and the database system itself to help support data independence and appropriate privacy constraints for application programs and end users. Other types of metadata such as textual descriptions and footnotes are primarily for the convenience of end users. Still other metadata such as category sets and measurement units are used by all four types of "users." Exhibit 1 pictures some of the major categories of metadata discussed thus far, pictured in various sectors of overlapping circles which represent different types of use.

**Exhibit 1: Types and Uses of Metadata**



We can also identify several different *functional* ways in which metadata are used, as follows:

*Data Definition*

A central use of metadata is data definition. Metadata attributes (e.g., data element name) can be used to describe not only internal data files, but also external data to be loaded or data prepared for export to other systems.

Two parallel versions of data definition may be needed, just as we have source code and compiled versions of computer programs. SPIRES, for example, maintains file definitions as (1) text database entries, for ease of reading and updating by database administrators and end users, which are in turn compiled into (2) machine code tables for efficient use by database system and application programs [SCHR75].

*Documentation*

If it is sufficiently well differentiated and properly structured, both DBMS and application programs can use computer-readable metadata to generate a full range of both printed and on-line documentation, such as database dictionaries; over-all indexes of attributes, entities, databases, and names; thesauri of restricted vocabulary subject index terms.

*Data Selection*

Both human-readable and compiled versions of metadata can aid various types of data selection. Users may employ metadata entity and attribute names plus attribute values to select data while *browsing* an on-line database dictionary or via a direct query, such as

find counties where state = cal and pop > 50000

*Data Manipulation*

As noted above, data manipulation routines can use metadata not only for identifying attributes, categories, and category sets, but also as an important part of computational algorithms.

*Data Display*

Another common use of metadata is to provide standard default labeling for different types of data and analysis output. Metadata for display may also include ancillary information, such as map outlines and coordinates for mapping software.

## 6. Metadata Characteristics

Metadata differs substantially from data, particularly statistical data, in terms of its typical data types, structures, access patterns, and update requirements. These differences help explain why integration of data and metadata management has been difficult, and why integration has been minimal to date.

In many respects, metadata resembles bibliographic information. Metadata are primarily *textual* and thus require corresponding textual functions, whereas statistical data are primarily *numeric*, and require numeric functions. Statistical data require *regular data structures* such as vectors, matrices, and arrays, whereas metadata require more flexible *open-ended hierarchical data structures*. Statistical access patterns tend to involve large numbers of entity instances (i.e., data records) for a relatively small number of attributes [TEIT77]. Metadata access typically involves accessing a high proportion of the metadata attributes information for a relatively small number instances of particular metadata entities (i.e., specific databases, data elements, etc.) at one time. Such access may be via an *index*, if available.

*Metadata Data Types*

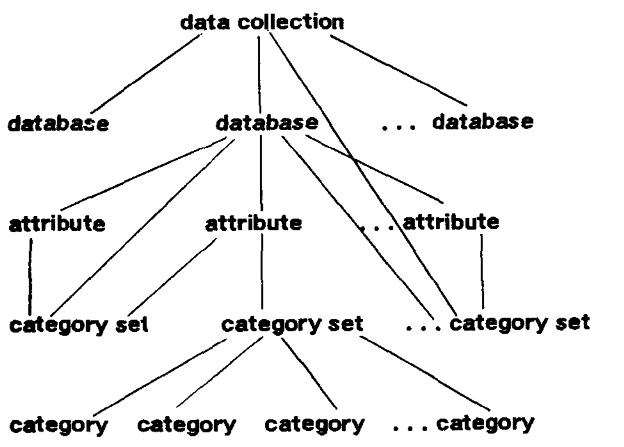
Although metadata is mainly textual, statistical metadata also includes numeric constants and ranges as well as expressions containing arithmetic operators, functions, and attribute names. Many types of data, including statistical, are fixed length. Most metadata, on the other hand, is quite variable in length. Some metadata may run to a page or more of text. Efficient representation and use of metadata thus requires not only most of the standard, fixed-length data types, but also extended or complex data types such as variable length text.

*Metadata Structure*

Metadata frequently contains *variable length lists* of multiply occurring values, such as subject index terms, aliases for main names, and category set values. Similarly, whole clusters of metadata information such as attribute, category set, and category metadata, are *themselves multiply occurring data structures.*

Exhibit 2 pictures metadata as an acyclic hierarchical structure with multiply occuring clusters of information. Each node represents a metadata entity and the cluster of information associated with that entity. Each line or arc represents an explicit linkage or association between different metadata entities.

**Exhibit 2: Hierarchical Metadata Structure**



Each node or metadata structure can contain or reference both simple metadata attributes and further metadata structures pictured below it. For example, the database node includes both simple database level information plus additional clusters of information for each of the attributes in the database. If certain attributes are arrays or contain categorical values they may reference additional clusters of information for one or more category sets.

Metadata definition may specify that certain types of nodes automatically inherit certain higher level metadata attributes, while other linkages must be explicit. A metadata definition might specify that attributes automatically inherit database missing data codes, while category sets can be referenced explicitly at any level that is not higher than the one where they are defined. For example, the category set at the left side of exhibit 2 is defined for a database and referenced by two of its attributes. The category set at the right is defined globally for the entire data collection, and it is referenced by one database and one attribute.

Exhibit 3 shows an example data definition fragment using a hierarchical metadata specification currently under development for SEEDIS [MCCA82A]. This data definition contains repeating occurrences of aliases, subject terms, attribute structures, category set structures, and category structures. Note that the category set specification in the last line simply references a category set defined in the previous table.

**Exhibit 3: Example Data Definition Fragment**

```
database = census80_stf1
subject = population
subject =  social characteristics
subject = United States
attribute = fips_state_code
        alias = scode
        alias = sc
attribute = state_name
        alias = state
attribute = table12
        structure = matrix
        universe = persons
        label = Total Population by Race and Sex
        subject = population
        array_size = 4*2
        category_set = race
        subject = race
                category = white
                category = black
                category = indian
                        label = American Indian, Eskimo,
                                and Aleut
                category = asian_pi
                        label = Asian & Pacific Islander
        category_set = sex
                category = male
                category = female
attribute = table12
        structure = matrix
        label = Hispanic Population by Race
        array_size = 4
        category_set = race
```

*Logical Linkages in Metadata*

Many types of metadata, particularly statistical metadata may include names of other metadata attributes which can serve as *keys* or *pointers* to more extensive information stored elsewhere. For example, statistical expressions may include names of data and metadata elements; footnote numbers associated with individual data element descriptions may point to a common set of footnote text; dimension or category set names may refer to standard category set structures defined outside a particular database; some metadata, such as sources or references, may point to published materials that are not even machine-readable.

In addition to these explicit linkages, most statistical systems include facilities for automatic inheritance of certain global metadata attributes (such as missing data codes) from the database to the data element level. Inherited metadata usually can be overridden at the data element level.

*Metadata Access Patterns*

For large databases, users typically need relatively large amounts of metadata to help them browse and select data. They usually extract subsets of data, with correspondingly smaller amounts of metadata, to work with intensively. They may return to the main data archive from time to time for additional variables and metadata information [BURN82].

*Metadata Updates*

Metadata updates may involve addition or modification of the *types* of metadata that are recognized by the system as well as changes to existing metadata *values.* For statistical archives where data tend to be relatively static, *metadata value* updates are usually either:

- addition of whole sets of metadata values for new databases as they are added, or
- small changes to supplement or correct existing metadata (e.g., changing the physical location of a particular file or adding a new alias for an attribute name).

Small, incremental updates to the *types* of metadata the system recognizes will occur from time to time as application designers and end users think of new metadata they need and new ways to use existing metadata information.

Metadata thus does not require facilities for rapid or concurrent updates, but it does require facilities for easy *addition* of new metadata and new types of metadata. A new type of metadata may only need to be available for new databases as they are added so this should not necessarily require extensive reloading of either data or metadata. Finally, database administrators may need to change the logical, physical, or linkage *structure* of metadata from time to time to improve efficiency or simplify use.

## 7. Metadata Management Principles and Requirements

As the preceding discussion has noted, data and metadata differ in many respects. When we consider metadata as data, however, their DBMS requirements overlap to a considerable degree. Given the metadata entities, attributes, users, and characteristics we have examined, data and metadata should be able to share most DBMS facilities, provided they are sufficiently general. Shared DBMS facilities should include the following:

data definition language
manipulation procedures
security specifications
input processing
output formatting
query facilities

General shared facilities usually require more resources for initial design and implementation, but separate facilities increase the ultimate burdens of maintenance and understanding for everyone from system designers through end users.

Contrary to the ways it is often treated, data description requires even more detailed and systematic standardization, differentiation, and management than data. Metadata needs to be sufficiently simple that humans can read and understand it, yet structured so that programs can parse and compile it. Since the kinds of data and metadata that need to be accommodated will continue to grow in ways that cannot fully be anticipated, basic DBMS facilities must be as flexible and *extensible* as possible. Database designers should be able to add new types and structures of metadata, while database administrators should be able to add and revise metadata values quickly and easily -- both without necessarily reloading other metadata or data.

Given these general goals of *integration, standardization, simplicity*, and *extensibility*, along with our preceding analysis, the sections below outline specific *structural*, *definitional*, and *functional* requirements for integrated management of data and metadata.

### 7.1. Complex Data Types and Structures

While various representations of data and metadata may differ in their particulars, they share a pervasive need for common structural features that apply to all levels of definitional elements. The basic data types and structures required for integrated management of data and metadata are:

- variable length text attributes -- which may range in size from null to several pages
- multi-valued attributes -- in which occurrences may range from zero to arbitrarily large numbers, without prespecifying the number of occurrences
- array attributes -- including vectors and matrices
- named hierarchical structures -- composed of any combination of the above types and/or other nested hierarchical structures (which may themselves occur from zero to an arbitrary number of times)

To describe data and metadata in a comprehensive way, data definition language requirements begin to approach those of modern programming languages that support abstract data structuring and complex data types. Metadata and statistical data share a common need for complex data types currently found in few systems [BECK78, KLEN81].

To make data types and structures as extensible as possible, the underlying physical implementation should support addition of new optional attributes and structures to the schema and entry of corresponding values to new data or metadata records without necessarily reloading old ones.[2]

### 7.2. Definitional Requirements

The most important general requirements for data and metadata definition are to provide many highly differentiated, standardized metadata components rather than a few more general ones, and to provide mechanisms for adding new metadata components easily as new needs arise. The more specific each piece of information is, the easier it is for both humans and programs to use in various ways. Integrated metadata management requires a general data definition language capable of defining all the metadata types, entities, and attributes outlined above. The language should include a set of "metadata primitives" for defining and extending the language itself.

*Simple Data Definition Language*

Both data and metadata should be defined in terms of a single extensible, non-procedural *data definition language* (ddl). The basic grammar of such a language should be simple, such as "name = value" assignments for each individual piece of metadata information, which humans can understand and programs can parse relatively easily. If it seems desirable to have more structured forms in addition (e.g., lists of items separated by commas, or ranges delimited by colons), those can be identified as special *named structures* and standard processing procedures employed to automatically break them down into the more basic form.

*Common Name Structures*

It also is desirable to use common names, particularly for descriptive information such as "alias," "note," etc., across different types of metadata entities such as databases, attributes, category sets, and categories. In such cases there need to be ways to define and reference different hierarchically nested metadata items

[2] For example, in SPIRES one can do this by including a variable length "optional element bit mask" as a part of every data record and intra-record structure; each bit indicates the presence or absence of a particular data element in a given data record, and the bit string can simply be lengthened as new elements are defined [SPIR73].

whose simple names are not unique. (e.g., using a structure name concatenation operator such as the "@" in "attribute@note").

*Named Lists*

Many types of metadata consist of named *lists* of names, codes, or numeric values such as category value sets. In addition to supporting such named structures, the definition language should provide standard processing procedures for automatic table-lookup, input validation, recoding, and mechanisms to facilitate automatic conversion between related lists. For example, we might want to specify how a list of 26 racial categories can be grouped into another list of three basic racial categories.

*Indexing Specifications*

Since metadata needs to be *indexed* in various ways (see section 8.3 below), there is also a requirement for non-procedural definition of index contents and structure, what values should be passed to which indexes, and what standard processing procedures should be invoked to do so.

*Structure Specification*

It is important for statistical databases that the data definition language support specification of category sets, vectors, matrices, etc., as well as the various types of statistical characteristics outlined in section 4. To cite just one example, census databases in SEEDIS are comprised primarily of aggregate summary data arranged as multi-dimensional tables (e.g., total population by age, race, and sex for different geographic levels such as tracts, counties, places, etc.). Each census file contains hundreds of such tables with an average of twenty to fifty cells per table. Representing such data as simple single-valued data elements corresponding to individual cells of each table would entail prohibitive overhead in terms of metadata or extra record keys (not to mention loss of binding information) When such data are represented as *array attributes*, metadata is required only for the sum of the cardinalities of each dimension, rather than the product. Data storage locations can be calculated from the metadata. This storage scheme is much more economical and compact, but it requires more complex metadata and more sophisticated data management software to use that metadata.

*Metadata Definition Primitives*

As with other types of languages, it is desirable from the standpoint of users and software maintenance to define the data definition language recursively, using a subset of metadata attributes as "metadata primitives." While space does not permit discussion of what might constitute a necessary and sufficient set of such primitives, exhibit 4 suggests how a metadata definition fragment might define the metadata elements "element," "alias," and "type" in terms of the primitive data elements "structure," "element," "type," "occurrence," and "alias."

**Exhibit 4: Metadata Definition Fragment**

```
structure = data_element_group
     element = element
          alias = attribute
          alias = data element
          type = char
          occurrence = 1
     element = alias
          type = char
          occurrence = optional
     element = type
          type = char
          occurrence = 1
```

Another way to make the data definition language extensible would be to define it in terms of a separate set of primitive operators, such as those of Kreps' Semantic Core Model [KREP81].

## 8. Metadata Functional Requirements

The types of data structures and data definition language features outlined above are not in themselves sufficient for integrated management of data and metadata. Database designers, administrators, and users also need certain *functional* facilities to use metadata effectively. The following subsections outline a few of the more important ones.

### 8.1. Automatic Indexing

Since one of the most important purposes of metadata is to help *locate* information, there is a major functional requirement for automatic mechanisms to create, maintain, and use metadata indexes. Some important types of objects that might be indexed include:

- all non-trivial *words* from descriptive metadata items at all levels (i.e., name, title, subject, description, etc.);
- *names and aliases* of all databases, attributes, and each type of entity for which data is available;
- names of all the specific *entity instances* for which data is available;
- *dates* covered by particular databases, attributes, etc.;
- restricted vocabulary *keyword subject terms* linked to specific databases, attributes, category sets, etc.

"Free text" word indexes help users maximize recall (i.e., the proportion of relevant items retrieved), while more restricted indexes help users maximize precision (i.e., the proportion of retrieved items that are relevant).

*Thesauri*

Keyword indexes often are organized in the form of a thesaurus to help users navigate through a hierarchically structured set of restricted vocabulary subject indexing terms. In a thesaurus, each entry contains a structured list of broader, narrower, and related terms (and sometimes others such as "used for," "see," etc.) as well as pointers to the records containing those terms. Each broader, narrower, and related term may itself have broader, narrower, and related terms. [CAHN80] describes an innovative on-line system, which dynamically arranges target terms with related terms plus several levels of broader and narrower terms on a video display terminal. Although most thesaurus development to date has pertained to textual data, some recent efforts have been made to extend thesaurus tools to descriptions of numeric data [CHAN81]. Thesaurus indexing facilities should be able to automatically generate cross references -- e.g., so that addition of a "broader term" attribute for a particular subject term will automatically generate the corresponding "narrower term" cross reference.

*Index Generation and Maintenance*

Indexing facilities should also be extensible -- i.e., it should be possible to define and create new indexes or modify old ones without updating data or existing indexes. In order to build and maintain comprehensive indexes, relevant DBMS facilities must support indexes which can reference different attributes and record-types in different databases. Although metadata indexing requires considerable generality, index maintenance does not have to be dynamic -- as it does for some transaction-oriented systems. Overnight updating of indexes is perfectly acceptable because metadata is relatively static.

## 8.2. Query Facilities

Users need to be able to search for metadata information using the same high level query language used to select subsets of data. The query language should permit users to specify what type of metadata entity (e.g., data elements, types of entities, or databases) as well as which index or metadata attribute will be the target of a particular query (e.g., "find *elements* where *word* = unemployment").

*Preliminary Results of Index Searches*

When the user only specifies the name of a particular index (e.g., "find *subject* = housing"), the system should respond with the number and name of target metadata entities located prior to returning metadata for those entities -- for example:

```
  5 databases (db)
 67 tables (tb)
 22 time series (ts)
283 simple data elements (de)
```

Note that this implies a single index referencing different types of metadata records (databases, tables, etc.).

*Range and String Operators*

In addition to the standard arithmetic and logical operators for equality and inequality, the query language should also support range searching (e.g. "between ...and," "from...to," etc.). and text operators such as:

- prefix (e.g., all words and terms beginning with the string "cong").
- suffix (e.g., all items with the suffix "ment")
- string (all words and terms containing the string "ener" anywhere within them)
- word (all terms containing a particular word surrounded by blanks or punctuation)
- having (all items that contain a particular sequence of characters; not necessarily contiguous, but separated by blanks)
- with (all items that contain a particular sequence of characters, which do not have to be contiguous or in the same word - this is useful for codes, etc.).

Another powerful approach to string searching might be to use metacharacters (e.g., "*[ ]") as in the UNIX and software tools regular expression notation (see chapter 5 in [KERN76]) in addition to the operators suggested above.

*Intermediate Query Results*

Users should be able to save and retrieve intermediate query results by name at any stage. As in bibliographic systems, the query language should permit further subsetting or expansion of an intermediate search result based on the immediately preceding query as well as by boolean operations on named and saved search result sets. Boolean operators (and, or) and parentheses should be legitimate components of any primary or secondary queries in order to facilitate specification of complex criteria and minimize ambiguity.

*Other Query Navigation Aids*

Two other commands found in some bibliographic retrieval systems would also be useful for data and metadata searching. *Backup* returns to the previous result when a query has inadvertently reduced the number of hits below the number desired. *History* prints a review of the command sequence that produced the current result set.

## 8.3. Browsing Capabilities

Users like to browse indexes, attribute values, etc. in order to familiarize themselves with the data or metadata. It helps them verify their understanding of what to expect. Some systems such as SEEDIS and DIALOG [DIAL79] provide data and metadata browsing facilities, including the ability to mark items for selection during the course of browsing. Users should be able to browse through data as well as metadata records, indexes, thesauri, commands, etc., beginning at any specified point and displaying 10 to 20 entries at a time. They should be able to specify whether they wish to see sequential entries or a systematic sample of entries from throughout a specified file or index.

## 8.4. Manipulation Facilities

Most data management systems provide data manipulation facilities. But few of those facilities use pertinent metadata information. This is especially important for statistical data manipulation, where greater use of metadata could relieve users and application programs from considerable burdens of type checking, error calculation, and output labeling. At minimum, integrated DBMS manipulation functions should all automatically use and produce data description as well as data, as do most relational systems, the SEEDIS codata tools [MERR81], system S [BECK78], and P-STAT [BUHL79]. In addition, they should be able to use statistical characteristics as described in section 4.2 to do automatic type checking, conversion, etc. for statistical data and metadata.

## 8.5. Output Display Control

Most bibliographic retrieval systems provide several alternative output formats that users can specify in order to control the amount of information they see (e.g., only titles, authors and titles, full citation, etc.). Similarly, a good metadata subsystem should enable users to select the amount and format of metadata information to be displayed at any time. For example, a user might want to display only short names while scanning a large number of potential data items, but full paragraph descriptions for a smaller selected subset of items. In addition, users should be able to request display of non-standard subsets of metadata in a standard namelist format (i.e., variable name = <value> for each specified metadata item). [WONG82] describes an experimental system that would enable users to explore database schemas and frame database queries using interactive graphic symbols and displays.

*Sequencing Order*

Whenever sets of metadata or data have been retrieved, users should be able to control the order of presentation of the information by means of a simple sequencing command, specifying one or more attributes to be used as sort keys.

*Non-Procedural Formatting Language*

While not essential, another very useful facility that extends control over output display is a generalized, non-procedural formatting language to define input and output of data and metadata. [SPIR80] describes such a language and its uses.

### 8.6. Documentation and User Aids

Users need on-line help and explanatory facilities to help them find and use metadata and associated tools to retrieve and manipulate data. Database designers and administrators need facilities to organize and maintain such information about data, metadata, and associated tools. One effective and extensible way of structuring explanatory information is to consider information on each command tool (e.g., browse, show, etc.) and each aspect of metadata (e.g., databases, types of entities, etc.) as a separate explanatory entity. Each such entity could comprise a "documentation database" record, which might be composed in turn of at least three differentiated kinds of information (with hypothetical commands that could access each separately shown in parentheses for illustration):

- descriptive text ("explain <term>")
- command syntax ("syntax <term>")
- a brief example ("example <term>")

## 9. Conclusions

This paper has examined the major *types*, *uses* and *characteristics* of metadata, as well as types of *objects* to which metadata pertains. It has outlined a number of specific types of metadata and logical data structures that frequently arise for statistical data. It has explored the similarities and differences between data and metadata -- especially for large statistical databases. Both metadata and much statistical data have more complex structures than most current data management systems are capable of handling in a natural and flexible way.

Given these types and characteristics of metadata and statistical data, and the uses that we want to make of them, the paper has proposed some basic *structural*, *definitional*, and *functional* requirements for integrated metadata management. We cannot fully anticipate future needs for data and metadata. But we can and should make data structures, definition language, manipulation facilities, and other tools easily extensible. Doing so will facilitate integration and gradual evolution towards a richer variety of metadata than can currently be found in any one system.

The considerations discussed above are guiding design and implementation of metadata enhancements in SEEDIS. Many of the issues raised in this paper are also fruitful areas for further research, including recursive metadata definition primitives, category set relationships, standard item processing procedures, and greater use of metadata in statistical analysis.

**Acknowledgments**

This work was supported by the Employment and Training Administration, U.S. Department of Labor, and the Applied Mathematical Sciences Research Program of the Office of Energy Research of the U.S. Department of Energy, under Contract No. DE-AC03-76SF00098. The author wishes to thank Fred Gey, Roy Hammond, Paula Hawthorn, Deane Merrill, and Arie Shoshani for their encouragement and valuable suggestions.

## References

| | |
|---|---|
| **BECK78** | Becker, R.A., and J.M. Chambers. "Design and Implementation of the S System for Interactive Data Analysis," *Proc. I.E.E.E. Compsac78, (1978)*, p. 626-629. |
| **BORA82** | Boral, H., D.J. DeWitt, and D. Bates. "A Framework for Research in Database Management for Statistical Analysis," Computer Sciences Technical Report #465. Madison: University of Wisconsin, Computer Sciences Department, February, 1982. |
| **BUHL79** | Buhler, S. and R. Buhler. *P-STAT 78 User's Manual*. Princeton, NJ: P -STAT, Inc., P.O. Box 285, 1979. |
| **BURE80** | Bureau of Labor Statistics, *Table Producing Language System, Version 5*, Washington, DC, July, 1980. |
| **BURN82** | Burnett, R. A., and Thomas, J. J., "Data Management Support for Statistical Data Editing and Subset Selection," in H.T.K. Wong, ed., *Proceedings of the First LBL Workshop on Statistical Database Management*, LBL-13851, March, 1982. |
| **CAHN80** | Cahn, D. F. and Murano, C. V., "Interactive Computer Graphics Displays for Hierarchical Data Structures," LBL-10247, Paper presented at the American Society for Information Science Mid-year Meeting. Pittsburgh, 1980. |
| **CHAN81** | Chan, P. and A. Shoshani, "SUBJECT: A Directory Driven System for Organizing and Accessing Large Statistical Databases," Proceedings, 7th VLDB, Cannes, France (ACM Order No. 471810), September, 1982. |
| **CHEN76** | Chen, P.P., "The Entity Relationship Model: Toward a Unified View of Data," *1 ACM Transactions on Database Systems 1*, (March, 1976), p. 9-37. |
| **CODD82** | Codd, E.F., "Relational Database: A Practical Foundation for Productivity," *25 Comm. ACM* 2, (February, 1982), p. 109-117. |
| **COMP82** | Computer Science and Mathematics Department, "SEEDIS Release Notes version 1.3," February, 1982; version 1.4 (preliminary), February, 1982. |
| **CURT81** | Curtice, R.M., "DATA DICTIONARIES: An Assessment of Current Practice and Problems," *Proceedings*, 7th VLDB, Cannes, France (AC Order No. 471810), September, 1982. |
| **DATA81A** | Data User Services Division, Bureau of the Census, *Census Software Package (CENSPAC) Preliminary User Reference Manual*, Washington, DC, 1981. |

DATA81B Data User Services Division, Bureau of the Census, *Census of Population and Housing, 1980: Summary Tape File 1 Technical Documentation*, Washington, DC, 1981.

DIAL79 DIALOG Retrieval Service, *Guide to DIALOG Searching* Palo Alto CA: Lockheed Data Services, 1979.

HAMM82 Hammond, R., "Metadata in the RAPID DBMS" in H.T.K. Wong, ed., *Proceedings of the First LBL Workshop on Statistical Database Management*, LBL-13851, March, 1982.

INST73 Institute for Social Research, *OSIRIS III*, 6 vols, Ann Arbor: Institute for Social Research, University of Michigan, 1973.

KERN76 Kernighan, B.W., and P.J. Plauger, *Software Tools*, Reading, Mass: Addison-Wesley, 1976.

KILO81 Kilov, Kn.I., and I.A. Popova, "Meta-Database Architecture for Relational DBMS" 12 *Sigmod Record 1*, 18-25 [7 Programmirovanie 1 (1981) UDR 681.3], October, 1981.

KLEN81 Klensin, J. C. and D. B. Yntema, "Beyond the package: A new approach to behavioral science computing," *20 Social Science Information 4/5*, (1981), p. 787-815.

KREP81 Kreps, P., "Views and Relativism in a Database Model," *Proceedings* of the Pingree Park Workshop on Data Abstraction, Databases, and Conceptual Modeling, ACM SIGMOD/SIGPLAN/SIGART, January, 1981.

MCCA79 McCarthy, J. L., "Data Characteristics, Application Requirements, and Database Management Tools: Matching Statistical User's Needs and Systems Capabilities," *Proceedings* of Computer Science and Statistics: 12th Annual Symposium on the Interface, University of Waterloo, Ontario, Canada, (May, 1979), p. 37-44.

MCCA82A McCarthy, J. L., "Enhancements to the Codata Data Definition Language" Lawrence Berkeley Laboratory, LBL-14083, February, 1982.

MCCA82B McCarthy, J. L., et al., "The SEEDIS Project: A Summary Overview," Lawrence Berkeley Laboratory, PUB-424, April, 1982

MERR81 Merrill, D., "CODATA Users' Manual," Lawrence Berkeley Laboratory, LBID-021, revised October, 1981.

MERR82 Merrill, D., "Problems in Spatial Data Analysis" in Proceedings of the Annual SAS User Group International Conference (San Francisco), LBL-14047, February, 1982.

MEYE81 Meyer, M.E. et al., eds, "Information Resource Dictionary System (IRDS) Requirements Document," ANSI/X3/H4 IRDS Technical Committee DRAFT Version 3.0, September, 1981.

NIE75 Nie, N.H., et al., *SPSS: Statistical Package for the Social Sciences* 2nd ed, McGraw Hill: New York, 1975.

ROBI80 Robinson, B. et al., *Scientific Information Retrieval User Manual* SIR, Inc., P.O. Box 1404: Evanston, IL, 1980.

ROWE82 Rowe, N.C., "Rule-based Statistical Calculations on a 'Database Abstract'," in H.T.K. Wong, ed., *Proceedings of the First LBL Workshop on Statistical Database Management*, LBL-13851, March, 1982.

SAS79 SAS Institute, Inc. *SAS User's Guide 1979 Edition* Raleigh NC, 1979.

SCHR75 Schroeder, J.R., W.C. Kiefer, R.L. Guertin, W.J. Berman, "Stanford's Generalized Database System" in D.S. Kerr, ed., *Proceedings of the International Conference on Very Large Data Bases* (Framingham, Massachusetts), September, 1975.

SHAN80 Shanks, M., "UNIX Software for Survey Analysis," University of California, Berkeley, Survey Research Center, 1980.

SHOS82 Shoshani, A., "Statistical Databases: Characteristics, Problems, and Some Solutions," this *Proceedings* (VLDB), Mexico City, 1982.

SMIT78 Smith, John M. and Diane C.P. Smith, "Principles of Database Conceptual Design," *Proceedings of the NYU Symposium on Database Design*, (1978), p. 35-49.

SPAR82 Sparr, T.M., "Units and Accuracy in Statistical Databases," in H.T.K. Wong, ed., *Proceedings of the First LBL Workshop on Statistical Database Management*, LBL-13851, March, 1982.

SPIR73 SPIRES Staff, *Design of SPIRES II*, Vol. 1 2nd ed. revised, SCIP/Campus Computing Facility, Stanford University, July, 1973.

SPIR80 SPIRES Staff, *SPIRES Formats Language*, SCIP/Campus Computing Facility, Stanford University, January, 1980.

SPIR82 SPIRES Staff, *SPIRES System Procs*, Center for Information Technology, Stanford University, January, 1982.

STEW78 Stewart, D. and M. Seda, *Data Processing in the National Health Insurance Study*, RAND Corporation, 1978.

TEIT77 Teitel, R.F., "Relational Database Models and Social Science Computing," *Proceedings* of Computer Science and Statistics: Tenth Annual Symposium on the Interface, Gaithersburg, MD: National Bureau of Standards, (April, 1977), p. 165-177.

WONG82 Wong, H.K.T. and I. Kuo, "A Graphical User Interface for Database Exploration," this *Proceedings* (VLDB), Mexico City, 1982.