FIPS PUB 124

FEDERAL INFORMATION PROCESSING STANDARDS PUBLICATION

1986 SEPTEMBER 30

U.S. DEPARTMENT OF COMMERCE/National Bureau of Standards

GUIDELINE ON FUNCTIONAL SPECIFICATIONS FOR DATABASE MANAGEMENT SYSTEMS

CATEGORY: SOFTWARE
SUBCATEGORY: DATA MANAGEMENT APPLICATIONS

FIPS PUB 124

U.S. DEPARTMENT OF COMMERCE, Malcolm Baldrige, *Secretary*
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, *Director*

Foreword

The Federal Information Processing Standards Publication Series of the National Bureau of Standards is the official publication relating to standards, guidelines, and documents adopted and promulgated under the provisions of Public Law 89-306 (Brooks Act) and under Part 6 of Title 15, Code of Federal Regulations. These legislative and executive mandates have given the Secretary of Commerce important responsibilities for improving the utilization and management of computers and automatic data processing in the Federal Government. To carry out the Secretary's responsibilities, the NBS, through its Institute for Computer Sciences and Technology, provides leadership, technical guidance, and coordination of Government efforts in the development of standards, guidelines and documents in these areas.

Comments concerning Federal Information Processing Standards Publications are welcomed and should be addressed to the Director, Institute for Computer Sciences and Technology, National Bureau of Standards, Gaithersburg, MD 20899.

James H. Burrows, *Director*
Institute for Computer Sciences and Technology

Abstract

This Guideline assists the data processing manager in the specifications of database management functions. In this Guideline is a framework for gathering and incorporating an appropriate set of data management functions into a request for proposals document. The emphasis is on the logical separation of the database management functional specifications, the relationship among the logical categories, and the recommended set of sources.

Key words: database management systems; DBMS; DBMS functional specifications; data model specifications; Federal Information Processing Standards Publication; global data factors; hardware and software constraints.

Natl. Bur. Stand. (U.S.) Fed. Info. Process. Stand. Publ. (FIPS PUB) 124, 30 pages
(1986)
CODEN:FIPPAT

For sale by the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161.

Federal Information Processing Standards Publication 124

1986 September 30

Announcing the Guideline on

FUNCTIONAL SPECIFICATIONS FOR DATABASE MANAGEMENT SYSTEMS

Federal Information Processing Standards Publications are issued by the National Bureau of Standards pursuant to the Federal Property and Administrative Services Act of 1949, as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973), and Part 6 of Title 15 Code of Federal Regulations (CFR).

1. Name of Guideline. Guideline on Functional Specifications for Database Management Systems (FIPS PUB 124).

2. Category of Guideline. Software, Data Management Applications.

3. Explanation. This Guideline assists Federal data processing managers in developing requests for proposals for database management systems based on functional specifications.

4. Approving Authority. U.S. Department of Commerce, National Bureau of Standards (Institute for Computer Sciences and Technology).

5. Maintenance Agency. U.S. Department of Commerce, National Bureau of Standards (Institute for Computer Sciences and Technology).

6. Cross Index.

a. Federal Information Processing Standards Publication (FIPS PUB) 110, Guideline for Choosing a Data Management Approach.

b. National Bureau of Standards IR 85-3164, "A Technical Overview of the Information Resource Dictionary System."

c. National Bureau of Standards IR 85-3165, "The Information Resource Dictionary System Command Language."

d. American National Standard X3.133-1986, Network Database Language (NDL).

e. (draft proposed) American National Standard Database Language SQL.

7. Applicability. This Guideline is intended as a basic reference for Federal data processing managers who are responsible for the planning, evaluation, and selection of agency database management systems.

8. Implementation. This Guideline should be consulted when Federal agencies are developing requests for proposals for new or replacement database management systems.

9. Specifications. Federal Information Processing Standards Publication 124 (FIPS PUB 124), Guideline on Functional Specifications for Database Management Systems.

10. Qualifications. This Guideline represents recommended good practices for identifying and selecting a database management system for Federal Government applications and in developing the documents needed to acquire new systems. In applying this Guideline, it is important to bear in mind that database management technology is rapidly evolving to take advantage of new hardware and software capabilities, and that each Federal agency must take into consideration its own specific circumstances when applying or referencing this publication.

10. Where to Obtain Copies of the Guideline. Copies of this publication are for sale by the National Technical Information Service, U.S. Department of Commerce, Springfield, VA 22161. When ordering, refer to Federal Information Processing Standards Publication 124 (FIPSPUB124), and title. When microfiche is desired, this should be specified. Payment may be made by check, money order, or NTIS deposit account.

Federal Information Processing Standards Publication 124

1986 September 30

GUIDELINE ON FUNCTIONAL SPECIFICATIONS FOR DATABASE MANAGEMENT SYSTEMS

Contents

Figures

Tables

EXECUTIVE SUMMARY

This document provides a set of guidelines for developing a "request for proposals" for a database management system. The guidelines recommend that the "request for proposals" should be organized into four logical areas which are identified as follows:

- Hardware and Software Constraints,
- Global Data Factors,
- Data Model Specifications, and
- Other Specifications.

The database management system functional specifications in one logical category can influence those in another. These **dependency relationships** are identified to serve as a "checks and balances" that can be used by data processing managers to help verify that thorough consideration has been given to each database management system functional specification included in the "request for proposals."

References for the technical content of a "request for proposals" are identified for each of the logical areas. The list of references is not all-inclusive, but serves as a starting point for the data processing manager who has the task of writing the "request for proposals" for a data base management system.

The final chapter discusses a methodology for writing the "request for proposals." The methodology uses the following three stage process:

Stage 1. Identify Requirements,
Stage 2. Prepare Representation, and
Stage 3. Prepare "Request for Proposals."

Each stage is comprised of a series of tasks. The stages and their respective tasks are sequentially presented in a recommended order of implementation.

An example of using the database management system functional specification categories to write a "request for proposals" is included in Appendix A. This example includes a limited set of database management system functional specifications; however, its content is full enough to emphasize the use of the logical categories in the preparation of a "request for proposals."

An outline of the terminology that is used for a network database management system [ASC-85a], a relational database management system [ASC-85b], and a data dictionary system [ASC-84] is included in Appendix B. This outline can serve as a quick reference tool for the proposed standard terminology.

1. INTRODUCTION

The abundance of commercially available database management system (DBMS) software packages has created a growing acquisition problem for data processing (DP) managers. The claims made by vendors about functional capabilities of their software packages compound the problem. Confronted with this situation, the DP manager is often without clear and unbiased guidance on the functional specifications necessary for acquiring a DBMS.

The Accredited Standards Committee (ASC) X3H2, a technical committee on databases, developed standard specifications for the syntax and semantics of interfaces to a DBMS [ASC-85a, ASC-85b]. Specifications were developed for defining and accessing both network and relational structured databases. However, these specifications do not address some of the facilities that might be required in operational database environments. In such cases, DP managers have to rely on promotional literature by vendors, DBMS surveys in computer magazines, and users of DBMS software with the desired additional features.

1.1 Scope of Guidelines

This document was prepared to assist DP managers with the acquisition of database management systems. It provides a set of guidelines specifying procedural methods for identifying and preparing the content of an RFP pertaining to DBMS functional specifications. Contracting regulations, requirements, restrictions, and incentives that are required in Federal purchases for open competition are outside the purview of this document. These guidelines represent only steps that can be used to prepare the technical specifications for the operating characteristics of a DBMS as they pertain to user application requirements.

1.2 Definitions

The functional specifications for database management systems can be placed into four logical categories as shown in table 1.

Table 1. *DBMS functional specification categories*

DBMS FUNCTIONAL SPECIFICATIONS					
Hardware and Software Constraints	Global Data Factors	Data Model Specifications		Other Specifications	
		Standards References	Implementation Enhancements	Standards References	Non-standards References

Hardware and Software. The "hardware and software constraints" category documents the computer resource environment of the organization. It lists the characteristics of that environment (i.e., model of computer hardware, version of operating system, available memory, available storage, and terminal types). Specifications in this category functionally define the prerequisites that a vendor's DBMS software must satisfy to operate in the target computing environment. An example of stating the hardware and software constraints in an RFP is shown in table 2.

Table 2. *Example of specifying "hardware and software" requirements*

Hardware (mandatory). The DBMS should be capable of running on a Brand X minicomputer with 8 mega bytes of internal memory and a total of 600,000 blocks (512 bytes per physical block) of external storage that is distributed across three disk packs.
Operating System (mandatory). The DBMS must currently run under Brand X operating system (Release 2, version 2).

Global Data Factors. The "global data factors" category includes basic data requirements of the applications environment. These requirements are a direct product of the analysis that should be performed on any existing or planned user applications. Such an analysis helps to define a list of basic requirements that an application environment will place on the DBMS software. FIPS PUB 110 [NBS-84] states that the following global data factors are derived from user requirements: data sharing, data usage pattern, data volume, data structures, staffing, and conversion. *Data sharing* emphasizes the need for the candidate DBMS to have the capability of handling concurrent data usage. *Data usage pattern* identifies such DBMS functional capabilities as ad hoc queries, and interactive modes of data processing. *Data volume* functionally defines the storage requirements. *Data structure* identifies the level of complexity among related data files and the dynamic capability that is required by a candidate DBMS to handle the relationships. *Staffing* identifies from user experiences the functional capabilities of a DBMS that are necessary to handle the varying levels of user sophistication. *Conversion* identifies the level of data processing that is functionally required to migrate data from one DBMS environment to another. The example in table 3 illustrates how to state the global data factors requirements in an RFP.

Table 3. *Example of specifying "global data factors" requirements*

Data sharing (mandatory). The DBMS must allow multiple logical views to be defined on the same data. As a result, it is imperative that concurrent access to the data be controlled by the DBMS.
Data Usage Pattern (desired). The nature of the applications demand a mean access time of 5 seconds on a retrieval query of 1000 database records and 10 seconds on an update. Access to the database will be 80% interactive and 20% batch.
Data Volume (mandatory). The DBMS must not have a logical limit on the number of records that can be assigned to a database file. The size of a database file should be limited only by the amount of available physical storage.
Data Structures (mandatory). The DBMS must adhere to the standard specifications that are outlined for defining relational database structures. This should allow database files to be viewed as tables comprised of rows and columns.
Staffing (mandatory). The DBMS must allow user-interfacing through screen forms, a SQL query language, a programming capability within the DBMS kernel, and the programming languages C and FORTRAN.
Conversion (mandatory). The DBMS must have a bulk loading facility that can handle populating a database from a plain ASCII file in which occurrences of attribute values are separated by commas or positioning.

Data Model Specifications. The "data model specifications" category lists the minimal set of data definition and data manipulation operators that identify a standard data model. In the case of the network and relational models, this minimal set of operators has been defined by the ASC X3H2 Technical Committee on Database.

Specifications in this section are important since a DBMS supports a data model and is an implementation of that data model [GALL-84]. Thus, a DBMS provides for transformation of the logical data structures of a data model to the physical storage structures of a particular hardware environment. An example of the RFP wording for this logical category is shown in table 4:

Table 4. *Example of specifying "data model" requirements*

Schema Definition Language (mandatory). The DBMS must follow the syntax and general rules that have been specified by ASC X3H2 as a minimum. Thus, the DBMS must allow defining schema definitions which contain a schema authorization clause, table definition, column definition, unique constraint definition, view definition, and privilege definition.
Module Language and Data Manipulation Language (mandatory). The DBMS must follow the syntax and general rules that have been specified by ASC X3H2 as a minimum.

Other Specifications. The "other specifications" category includes a projected set of utilities to enhance the usability of a DBMS that is designed to the specifications of a specified data model. Thus, the specifications in this category emphasize features that go beyond those unique to a specific data model. These include such enhancements as dictionaries, report writers, statistical packages, graphics, and libraries of special data processing functions. Some other features to specify include concurrency control, backup, and restart, as well as dumping and loading facilities. Table 5 is an example of the possible wording in an RFP for some of the DBMS features that are grouped in this logical category.

Table 5. *Example of specifying "other" requirements*

Integrity (mandatory). The DBMS must allow checking occurrences of attribute values against a specified set of allowed values. The DBMS must also allow the specification of assertions and triggers.
Database Transporting (mandatory). The DBMS must have a facility that can selectively dump portions or the entire database (i.e., the defined schema and the associated data occurrences). The dumped file must be loadable by the DBMS. The DBMS must also allow copying database record occurrences to standard files for information interchange.
Schema Modifications (mandatory). The DBMS must have facilities that allow modifications to the schema definition without the user having to explicitly dump the database.
Response Time (mandatory). The DBMS must support 4,800 transactions per hour from 24 on-line concurrent users with response time averaging two seconds for 95% of the time and up to a maximum of four seconds for the remaining 5% of the time.
Report Writer (desired). The DBMS should directly interface with a report writer facility that allows users to easily select, retrieve and print data in an on-line mode. Output should optionally be directed to either a printer or a terminal.
Restart and Recovery (mandatory). The DBMS must have a comprehensive and reliable recovery system that uses either the rollback approach, in which invalid or incomplete transactions and database images are backed up; and the shadowing approach with journaling (or transaction recording) and recovery by reapplying transactions against a previous version of the database. These facilities should also accommodate selected recovery for specific files, records or logical records.

2. RELATIONSHIPS AMONG THE LOGICAL CATEGORIES

The DBMS functional specifications in one logical category can influence those in another. These "dependency relationships" are illustrated in figure 1. The double headed arrows indicate that the decided features of a logical area will affect those of a related logical area and vice versa. The single headed arrows are unidirectional for any yielded results. Each relationship will be discussed below.

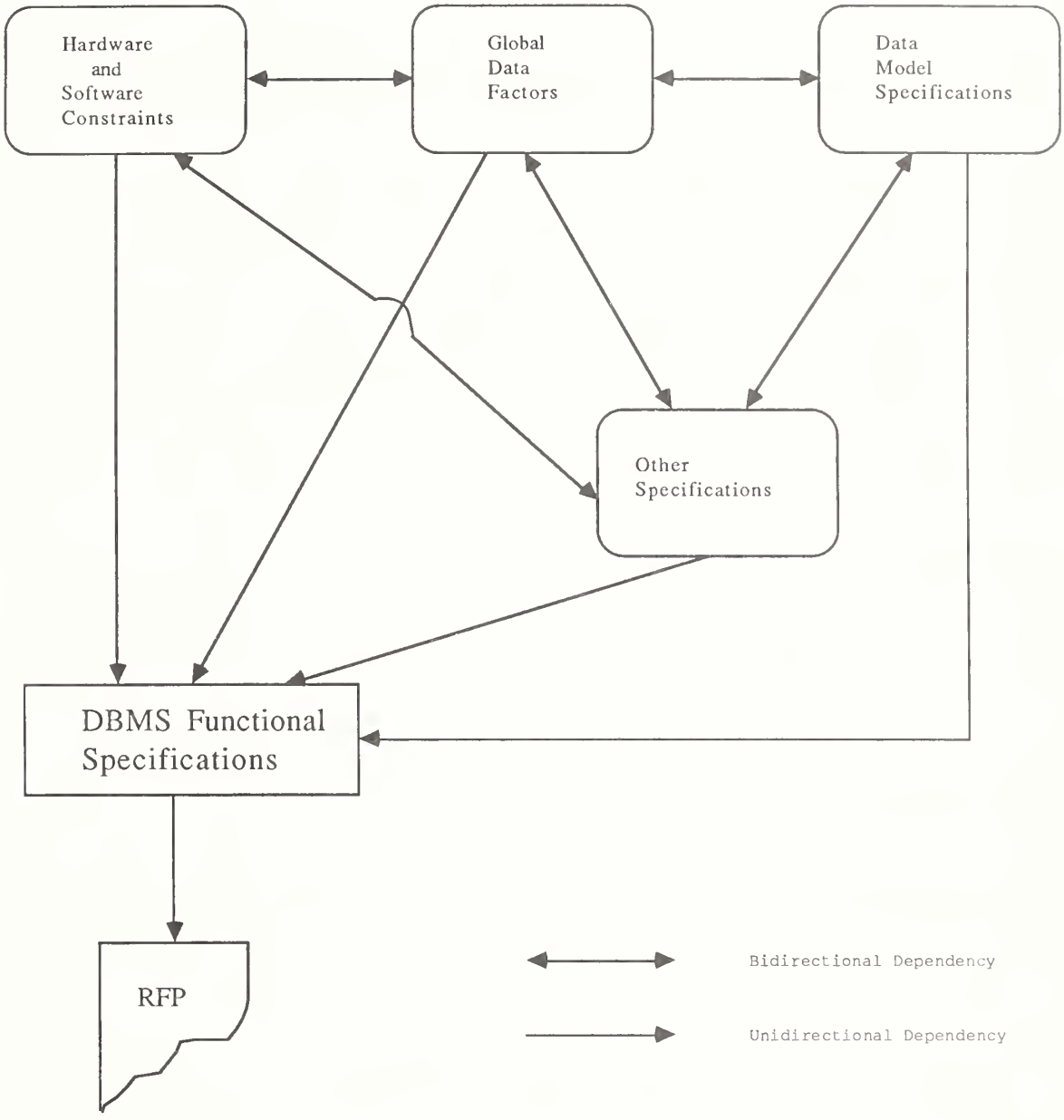

Figure 1. *Relationship among DBMS functional specification areas*

2.1 Global Data Factors versus Hardware and Software Constraints

Global data factors could be affected by certain hardware and software constraints. For instance, the **data volume** (a "global data factor") is affected by the **amount of physical disk storage** (a "hardware constraint"). If the amount of available disk storage is insufficient, more disk storage will have to be acquired or the volume of data will have to be reduced.

The **conversion factor** is another global data factor that is affected by hardware and software constraints. The conversion factors are concerned with the **amount of internal memory** (a "hardware constraint"), the **version of the operating system** (a "software constraint"), and the **space requirement for the software overhead** (both "hardware and software constraints"). For example, the amount of internal memory on the existing (or targeted) computer has to be large enough to handle the executable module of a DBMS, the operating system, and any on-line and background activities.

Since DBMS software is generally designed to run under a specific version of an operating system, RFP specifications should include the type and version of the existing operating system on the targeted computing hardware.

In addition to the data storage requirement, sufficient disk storage will be required for the DBMS software and any user application software. The disk storage requirement for the DBMS software is generally fixed; however, the disk storage requirement for the user application software will grow over a period of time.

2.2 Global Data Factors versus Data Model Specifications

The analysis process of user application requirements identifies the data structure and data usage pattern. These data characteristics are the key global data factors for selecting data model specifications. For example, the data usage pattern will indicate whether the data structure will be dynamic or remain stable. Generally, a **data model** (a "data model specification") is designed to efficiently handle either a dynamic or stable **data structure** (a "global data factor"), but not both. Thus, in most cases, it would not be cost effective to install a dynamic data structure under a data model designed to handle stable data structures.

2.3 Global Data Factors versus Other Specifications

Data sharing, staffing, and conversion are the global data factors that establish a relationship with the "other specifications" category. **Data sharing** (a "global data factor") indicates the need to specify such DBMS features as **concurrency control, integrity constraints, and security features** (three "other specifications"). **Staffing** (a "global factor") focuses on the levels of user sophistication which requires specifying the **different forms of user interfaces** (such "other specifications" as: query languages, programming languages, screen forms, and menus). **Conversion** (a "global data factor") specifies the required capabilities for such "other DBMS features" as: **bulk loading** from an ASCII file, **dumping** an ASCII file, and **backup and recovery** facilities (both manual and automatic).

2.4 Hardware and Software Constraints versus Other Specifications

Analysis of the user application requirements may reveal the need for **screen painting and graphics capabilities** (two "other specifications"). If this is the case, the **type of terminals** (a "hardware constraint") that are currently available for user interface with the DBMS could be incompatible with the **graphic signals** (an "other specifications") that are supported by the DBMS. Generally, screen painting and graphics capabilities will require a specific set of terminal types which are supported by the DBMS software. If the existing terminals are not included in the set of supported terminals, terminals from that set will have to be purchased; or the appropriate screen painting and graphics interfaces between the existing terminals and the DBMS will have to be developed by the purchaser. Thus, it is important to identify the types of terminals that are currently available for handling screen painting and graphics capabilities. This will indicate to the vendor of the DBMS software the need to include your existing terminals (if technically possible) in the supported set of terminals.

Another area of relationship between "hardware and software constraints" and "other specifications" is the need for the DBMS software to support **telecommunication capabilities** (an "other specification"). Specifications should identify the **required communication capabilities** (such "hardware constraints" as: modem connections, hard-wired connections, or both).

2.5 Data Models versus Other Specifications

Referential integrity (an "other specification") is inherent in the designs of the hierarchical and network data models. Therefore, the need to specify controls for referential integrity is not as great for these models as for the relational model which is not yet designed to handle this type of integrity constraint.

3. SOURCES FOR FUNCTIONAL SPECIFICATIONS FOR DATABASE MANAGEMENT SYSTEMS

There are many information sources that can be used to help define DBMS functional specifications. In this chapter, a subset of those sources is recommended for each DBMS functional specification category. These sources are shown by category in table 6.

Table 6. *Sources for DBMS functional specifications*

Hardware and Software Constraints
- Computer hardware and software manuals
- System Administrator
- System Log Book

Global Data Factors
- Guideline for Choosing a Data Management Approach [NBS-84]
- Evaluating Data Base Management Systems [KING-81]

Data Model Specifications
- Draft Proposed American National Standard on Database Language NDL [ASC-85a]
- Draft Proposed American National Standard on Database Language SQL [ASC-85b]

Other Specifications
- Standards Committee Specifications
  - ASC Draft Proposed Information Resource Dictionary System [ASC-84]
  - A Technical Overview of the Information Resource Dictionary System [GOLD-85a]
  - Using the Information Resource Dictionary System Command Language [GOLD-85b]
- Trade Journals (Commercial Publications)

3.1 Hardware and Software Constraints

The manuals that come with computer hardware and associated operating systems are appropriate information sources for defining hardware and software constraints when writing an RFP on DBMS functional specifications. They should specify the model of the computing hardware and the version of the operating system. This will automatically eliminate those DBMS packages that are not designed to run on the specified hardware, the operating system, or both.

Other information sources for defining the hardware and software constraints are the system administrator and the system administrator's maintenance log book. The system administrator should be helpful in defining any necessary technical details. Additionally, the maintenance log book should have recorded in it any peripheral modifications that have occurred to the hardware and operating system, such as: the initial amount of main memory, any additions of main memory, the amount of free disk space, and the variety of terminals.

3.2 Global Data Factors

Although, there are many information sources that the reader can reference when defining "global data factors," only two are identified in this section as a starting point. FIPS PUB 110 [NBS-84], as the first source, defines a framework for assessing whether a DBMS is the most appropriate tool for handling an organization's data management requirements. The assessment is made on the basis of the specified global data factors.

The second source from among the many is the chapter on "Understanding the Characteristics of Data" [KING-81]. In this chapter, the author attempts to separate the characteristics that are most likely to be attached to the data from those attached to the application.

3.3 Data Model Specifications

The initial information sources (if applicable) that should be referenced for data model specifications are the proposed standard database languages by the ASC X3H2 Technical Committee on Databases. Currently, standard languages have been proposed for the network model [ASC-85a] and the relational model [ASC-85b]. In these documents, the syntax and semantics of interfaces to a network DBMS are called NDL (Network Database Language) and SQL (Structured Query Language) for a Relational DBMS.

When a network or relational DBMS is required, the standard features for the NDL and SQL languages should be used as the nucleus of any functional specifications in an RFP. Such standards define a minimal set of functions that will constitute portability of database definition and application modules among conforming systems. Implementation specific features are not part of the standards specification. These features will be discussed in the section below.

3.4 Other Specifications

Many DBMS functions are not specified in the draft proposed standard database languages. These functions are generally implementation dependent and are targeted at enhancing the usability of a DBMS. For example, the draft proposed standard NDL language does not address:

a. An access control facility for granting access and operational privileges to specific users.
b. Additional integrity control capabilities for specifying more complex integrity constraints on the database.
c. A facility to import and export schema definitions.
d. A database unload facility for copying record and set populations to standard files for information interchange.
e. A schema database for making schema and subschema information available to accessing applications.
f. A schema manipulation language for creating, modifying or deleting portions of the schema or subschemas.
g. Interfaces to a data dictionary.
h. Application program pre-processing facilities for producing separate standard database modules and standard language programs.
i. A data storage definition language for defining physical storage structures and physical access methods.
j. Database procedures for user specified assertions and triggers.
k. A natural language query facility for ad hoc access to the database.
l. Report generator facilities for producing output tables and charts.
m. Graphics capabilities for direct database interface with standard graphics systems.
n. A distributed database facility for defining and accessing data at different nodes in a communications network.

Functional specifications for some of these issues are being considered as future enhancements to the currently proposed standard NDL language. The issue on interfaces to data dictionaries is an excellent initial candidate for standards functional specification. This is attributed to the fact that the ASC X3H4 Technical Committee has already produced a draft proposed standard which specifies the functional characteristics of a data dictionary system [ASC-84]. However, the functional specifications for interfacing the data dictionary to a DBMS have not been addressed. Other information sources on the functional characteristics of a data dictionary system include [GOLD-85a] and [GOLD-85b]. A recommended set of information sources for the characteristics of the other DBMS functions that have not been addressed by a standards committee are: publications on DBMS product evaluations, DBMS journals, DBMS textbooks, the 1978 CODASYL Journal of Development (for network DBMS's), and current DBMS users.

4. STANDARDS ON FUNCTIONAL SPECIFICATIONS FOR DATABASE MANAGEMENT SYSTEMS

As stated in the previous chapters, ASC X3H2 has proposed standard DBMS functional specifications for interfaces to: network [ASC-85a] and relational [ASC-85b] DBMS's. It is not in the scope of this guideline to discuss or to duplicate the content of these documents, but to emphasize the importance of appropriately referring to them in an RFP. That is, the "detailed" content of a standards document should not be included in an RFP—it should only be referenced. For example, an RFP that is written for the purchase of a relational DBMS should reference the standards document [ASC-85b]. The specifications in this document should be stated as the minimum acceptance criteria for any DBMS product claiming to be an implementation of the relational data model. See Appendix B for an outline of the standards specifications on the network and relational data models.

In addition to the standards committee work on data models, there has also been standards committee work on functional specifications for data dictionary systems [ASC-84]. The standards efforts on data dictionary systems is mentioned as a related DBMS functional specification. The standards committee work in this area will be helpful when specifying data dictionary terminology and any such functional capabilities in the composition of an RFP. However, specifications for interfacing data dictionary systems to database management systems are not currently available from the standards committee X3H4. This is an area in which the DP manager will have to rely on vendor competence and user references. See Appendix B for an outline of the standards functional specifications for a dictionary system.

5. A METHODOLOGY FOR DETERMINING DBMS FUNCTIONAL SPECIFICATIONS

The methodology for determining DBMS functional specifications can be divided into three stages as shown in table 7. These stages are listed as follows:

Stage 1. Identify Requirements,
Stage 2. Prepare Representation, and
Stage 3. Prepare RFP.

Stage 1—Identify Requirements. Each stage has a set of tasks that has to be performed. In stage one, there are three tasks to be completed. The first of these tasks is collecting information on the hardware and software constraints. This involves referencing such sources as: the system manuals for the hardware and software, the system administrator, and the system administrator's log book.

The second task is the analysis of user application requirements and collecting information on the characteristics of any associated data. There are many sources that can be referenced as a guide on performing this task; however, in this document, FIPS PUB 110 and [KING-81] are the referred sources.

Table 7. *The stages for collecting and documenting DBMS functional specifications in an RFP*

STAGE	ACTIVITY	TASKS
1	identify requirements	• collect information on hardware and software constraints • collect information on user application requirements • convert information on user application requirements to DBMS terminology
2	prepare representation	• survey marketplace • study standards specifications
3	prepare RFP	• document hardware and software constraints • document (if applicable) standards functions for selected data model: – data definition – data manipulation languages – data model languages • document any additional functions that are application specific: – security and integrity – backup/recovery – report generation – telecommunication – graphics – spreadsheeting – concurrency control – user friendly interfaces • menus • screens • document the need for user references

The third task requires using the information acquired from the second task to define the user application requirements in terms of: user interfaces (i.e., query languages, programming languages, menus and screens, report generating, bulk loading, and communication links), data security, data integrity, and graphics. The best source for this task is the involvement of the user community.

Stage 2—Prepare Representation. The first task in this stage is performing a marketplace survey of existing technology. This will be useful in determining a set of features that are needed to satisfy application requirements. There are many publications that will be useful for this task. Some examples are:

* AUERBACH Information Management Series,
* ComputerWorld,
* Datamation,
* Data Pro 70 the EDP Buyer's Bible,
* Datapro Management of Microcomputer Systems, and
* Software Digest · Newsletter.

The second task is studying existing standards specifications to formulate a minimal set of DBMS functions which should be used as the nucleus of any evaluation criteria for candidate DBMS software. This minimal set of DBMS functions can be matched against those identified by the second task. By matching these two sets of DBMS functions, a different set will be identified which will list the required extensions to the standards specifications. As previously stated, standards committees have specified DBMS functions for only the network model, relational model, and data dictionary. The list of documents on the standards specifications for DBMS functions are as follows:

* ASC NDL Database Language [ASC-85a],
* ASC SQL Database Language [ASC-85b],
* ASC Information Resource Dictionary System [ASC-84],
* A Technical Overview of the Information Resource Dictionary [GOLD-85a], and
* Using the Information Resource Dictionary System Command Language [GOLD-85b].

Stage 3—Prepare RFP. The first task in this stage is documenting any hardware and software constraints that currently exist in the user computing environment. The source for this effort should be the information collected in task 1 of stage 1. An example of the wording that could be used in an RFP is shown in Appendix A.

The second task involves documenting (if applicable) the level of conformity to standards specifications. For example, network or relational DBMS software would be required to adhere to the ASC proposed standard specifications for the languages on data definitions, data manipulations, and data modules. The standards specifications should be used as the core for the DBMS functional specifications. As such, the standards specifications will serve as the reference point from which the layers of enhancement features can be decided.

The third task is the documentation of any additional functions that are application specific but are not included in the ASC draft proposed standard specifications. This will include such areas as:

* security,
* integrity,
* backup and recovery,
* telecommunication,
* graphics,
* spreadsheets,
* reports,
* concurrency control, and
* user friendly interfaces (i.e. menus and screens).

Specifications for these additional functions will be the most difficult task in the writing process of the RFP since there are no standards specifications for the majority of these functions. The difficulty will be deciding on a median point between being too specific or too general.

If the specifications are **too specific**, the scope of the RFP will be limited to a small number of vendors. Thus, the degree of competitive bidding will be very low. An example of such specific specifications is biasing the desired additional features toward those of a particular vendor's database management software. This is an easy trap to fall into but it should be avoided.

At the other end of the spectrum, the scope of the RFP can be written at **too general** a level. This will make the evaluation process very time consuming and difficult since a large number of vendors will be able to claim compliance. An example of over generalizing is the following "specification phrase": The DBMS software must be capable of generating reports. This statement should include some specifics such as: the desired form of interfaces and the desired output controls.

The **median point** is a set of specifications that are detailed but not tailored to any specific DBMS product. That is, the functional operations of a (**desired** or **mandatory**) feature should be clearly defined so that the set of complying vendors will be functionally comparable. Without any standards specifications, the terminology needed for the necessary details will often vary from reference to reference. This problem can be alleviated by appending to the RFP a set of definitions for any DBMS functional terms that are not part of the standards specifications.

The fourth task is to include in the RFP document a request for a list of users of the proposed DBMS software. The list of users will aid the evaluation and selection process. The information gathered from these users will be helpful in verifying whether the DBMS is functionally conforming to the specifications in its manuals; in other words, "it does what it is supposed to do." Also, it helps to determine whether the implementation extensions are full enough to satisfy the user application requirements.

REFERENCES AND ADDITIONAL READINGS

[ASC-84] Accredited Standards Committee, X3H4, Technical Committee on Data Dictionary, Draft Proposed American National Standard on Information Resource Dictionary System, IRDS, December 1984.

[ASC-85a] Accredited Standards Committee, X3H2, Technical Committee on Database, Draft Proposed American National Standard on Database Language, NDL, October 1985.

[ASC-85b] Accredited Standards Committee, X3H2, Technical Committee on Database, Draft Proposed American National Standard on Database Language, SQL, October 1985.

[ATRE-86] Atre, Shaku, Mini and Mainframe DBMS—The Seven-step Solution, ComputerWorld, March 31, 1986, pp. 35-48.

[GALL-84] Gallagher, L. J. and Draper, J. M., Guide on Data Models in the Selection and Use of Database Management Systems, NBS Special Publication 500-108, January 1984.

[GOLD-85a] Goldfine, A. and Konig, P., A Technical Overview of the Information Resource Dictionary System, NBSIR 85-3164, April 1985.

[GOLD-85b] Goldfine, A., Using the Information Resource Dictionary System Command Language, NBSIR 85-3165, April 1985.

[KING-81] King, J. M., Evaluating Data Base Management Systems, Van Nostrand-Reinhold, New York, 1981.

[NBS-84] NBS, Guideline for Choosing a Data Management Approach, FIPS PUB 110, December 1984.

[SIBL-82] Sibley, E. H., A Functional Specification of the Relational DBMS, NBS-GCR-82-372, February 1982.

[SHEP-85] Sheppard, C. L., Guide for Selecting Microcomputer Data Management Software, NBS Special Publication 500-131, October 1985.

APPENDIX A

(An **EXAMPLE** of DBMS Functional Specifications by Logical Category)

1. Hardware and Software Constraints

This section describes the current computing environment.

Hardware (mandatory). The DBMS should be capable of running on a Brand X minicomputer with 8 mega bytes of internal memory and a total of 600,000 blocks (512 bytes per physical block) of external storage that is distributed across three disk packs.

Operating System (mandatory). The DBMS must currently run under Brand X operating system (Release 2, version 2).

2. Global Data Factors

This section describes the user application requirements in terms of DBMS functional specifications.

Data Sharing (mandatory). The DBMS must allow multiple logical views to be defined on the same data. As a result, it is imperative that concurrent access to the data be controlled by the DBMS.

Data Usage Pattern (desired). The nature of the applications demand a mean access time of 5 seconds on a retrieval query of 1000 database records and 10 seconds on an update. Access to the database will be 80% interactive and 20% batch.

Data Volume (mandatory). The DBMS must not have a logical limit on the number of records that can be assigned to a database file. The size of a database file should be limited only by the amount of available physical storage.

Data Structures (mandatory). The DBMS must adhere to the standard specifications that are outlined for defining relational database structures in section 3. This should allow database files to be viewed as tables comprised of rows and columns.

Staffing (mandatory). The DBMS must allow user-interfacing through screen forms, a SQL query language, a programming capability within the DBMS kernel, and the programming languages C and FORTRAN.

Conversion (mandatory). The DBMS must have a bulk loading facility that can handle populating a database from a plain ASCII file in which occurrences of attribute values are separated by commas or positioning.

3. Data Model Specifications

This section identifies the standard relational DBMS functions that a vendor's DBMS product must adhere to.

Schema Definition Language (mandatory). The DBMS must follow the syntax and general rules that have been specified by ASC X3H2 as a minimum. Thus, the DBMS must allow defining schema definitions which contain a schema authorization clause, table definition, column definition, unique constraint definition, view definition, and privilege definition.

Module Language and Data Manipulation Language (mandatory). The DBMS must follow the syntax and general rules that have been specified by ASC X3H2 as a minimum.

4. Other Specifications

This section identifies those user application requirement functions that are not covered in the ASC X3H2 standard functional specifications.

Integrity (mandatory). The DBMS must allow checking occurrences of attribute values against a specified set of allowed values. The DBMS must also allow the specification of assertions and triggers.

Database Transporting (mandatory). The DBMS must have a facility that can selectively dump portions or the entire database (i.e., the defined schema and the associated data occurrences). The dumped file must be loadable by the DBMS. The DBMS must also allow copying database record occurrences to standard files for information interchange.

Schema Modifications (mandatory). The DBMS must have facilities that allow modifications to the schema definition without the user having to explicitly dump the database.

Response Time (mandatory). The DBMS must support 4,800 transactions per hour from 24 on-line concurrent users with response time averaging two seconds for 95% of the time and up to a maximum of four seconds for the remaining 5% of the time.

Report Writer (desired). The DBMS should directly interface with a report writer facility that allows users to easily select, retrieve and print data in an on-line mode. Output should optionally be directed to either a printer or a terminal.

Restart and Recovery (mandatory). The DBMS must have a comprehensive and reliable recovery system that uses either the rollback approach, in which invalid or incomplete transactions and database images are backed up; and the shadowing approach with journaling (or transaction recording) and recovery by reapplying transactions against a previous version of the database. These facilities should also accommodate selected recovery for specific files, records or logical records.

APPENDIX B

(An Outline of the Standards DBMS Functional Specifications)

The outline of the standards specifications is meant to serve as a **quick reference tool** for what is standard and to show some of the enhancements that have been made to these specifications. This is achieved through the use of tables. Each table contains a set of features and their associated specifications. The specifications for these features are categorized into **standards specification** and **implementation extensions**. In some cases, a specification (under the "standard" column) will appear later in the same table as a feature. This is necessary to traverse the alternative branches in the syntactical specifications for the standards interface languages.

The "implementation extension" column is included in the tables to illustrate that the standards specifications, in some cases, are insufficient for the actual implementation of data base management software. To illustrate this fact, some features from current implementations in the marketplace were extracted to highlight those areas of the standards specification that have been extended by vendors. The blank specification cells indicate those areas of the standards specification which have proven to be sufficient in current implementations.

There is also an outline on the functional characteristics of a data dictionary system as proposed by the ASC X3H4 Technical Committee. The discussion will be general and will not attempt to discuss any specifics which can be found in the draft proposed document [ASC-84].

I. Network Data Model

The standards functional specifications for the network data model are shown in tables 8 through 11. As shown in these tables, the standards committee categorizes the functional specifications into four areas: schema definition language, subschema definition language, module language, and data manipulation language.

A. Schema Definition Language

The function of the schema definition language is to declare the structure and integrity constraints of a network database.

Standards Specification. The standards specify that the syntax for the schema definition language as shown in table 8 should contain a schema name clause, a collection of record type descriptions, and a collection of set type descriptions.

The **schema name clause** identifies the name which will be assigned an associated collection of record and set type descriptions.

The **record type descriptions** contain a record name clause, a record uniqueness clause, component types, and a record check clause. The **record name clause** contains the name of a record type; and the **uniqueness clause** designates the component identifiers of that record type that are to serve as a uniqueness constraint on record occurrences.

The **component type clause** contains the component name, its data type, and possible occurs and default clauses. A **data type** can be one of three types: character string, exact numeric, or approximate numeric.

The **record check clause** confirms the validity condition for occurrences of a record type.

The **set type** contains the set name, owner, order, and member clauses. The **set name clause** assigns a name to the set type; while, the **owner clause** specifies the record name that is to be the owner of that set type.

The **member clause** consists of: (1) a **member name clause** which specifies the record name of a member record type of a set type; (2) an **insertion clause** used to define the insertion characteristics of a member record type of a set type to be automatic, structural, and manual; (3) a **retention clause** used to define the retention characteristics of a member record type of a set type to be fixed, mandatory, and optional; (4) a **member uniqueness clause** which specifies the uniqueness constraint for member records of each occurrence of a set type; (5) a **key clause** used to define the sort control key for a member record type of a sorted set type; and, (6) a **member check clause** which specifies a validity condition on member records of each occurrence of a set type.

The **order clause** specifies the ordering of member records in a set. The standard order options are first, last, next, prior, default, and sorted order.

Table 8. *Standards functional specifications for network schema definition languages*

Schema Definition Language: NETWORK		
	Specifications	
Features	*Standard*	*Implementation Extensions*
schema	• schema name clause • record type • set type	• area clause
schema name clause	• schema name	• call clause • access-control clause
record type	• record name clause • record uniqueness clause • component type • record check clause	
record name clause	• record name	
record uniqueness clause	• component identifier	
component type	• component name clause • data type • occurs clause • default clause	• conversion clause • source clause • result clause • call clause • check clause • access-control clause
component name clause	• component name	
data type	• character string type • exact numeric type • approximate numeric type	• bit type • implementor-name
occurs clause	• extends	• depending on clause
default clause	• literal	
record check clause	• condition	
set type	• set name clause • owner clause • order clause • member clause	• call clause • access-control clause
set name clause	• set name	
owner clause	• record name • system	
order clause	• order option – FIRST – LAST – NEXT – PRIOR – DEFAULT – SORTED ORDER	
member clause	• member record name clause • insertion clause • retention clause • member uniqueness clause • key clause • member check clause	• selection clause • call clause • access-control clause
member record name clause	• record name	

Table 8. *Standards functional specifications for network schema definition languages (continued)*

NETWORK Schema Definition Language		
	Specifications	
Features	*Standard*	*Implementation Extensions*
insertion clause	• insertion mode – automatic – structural – manual	
retention clause	• fixed • mandatory • optional	
member uniqueness clause	• component identifier	
key clause	• key item • key duplicates	
member check clause	• condition	
component identifier	• dot style component identifier • of style component identifier	

Implementation Extensions. Standards specifications for the schema definition language do not address: an access control facility for granting access and operational privileges to specific users; additional integrity control capabilities for specifying more complex integrity constraints on the database; and a data storage definition language for defining physical storage structures and physical access methods. These DBMS functional capabilities are implementation dependent.

ACCESS CONTROL can be implemented as part of the schema name, component type, set type, and member clauses (see table 8). These are some of the levels at which implementations have addressed controlling data access on a user-by-user basis.

Complex integrity controls can be handled in implementations that allow CHECK CLAUSES to be included as part of the component type clause. The DEPENDING ON clause which can be implemented as part of the component type clause is another way of controlling complex integrity constraints.

Data storage is handled through implementations of an AREA CLAUSE. This clause would control paging, buffering and the amount of physical work space available for the data base.

B. Subschema Definition Language

The function of the subschema definition language is for declaring a user view of a database.

Standards Specification. The syntax for the subschema definition language as shown in table 9 must contain a subschema name, a collection of record views, and a collection of set views.

A **record view** contains a record renamed clause, a record view name clause and a component view clause. The **record renamed clause** is optional. If the record renamed clause is specified, the **record view name clause** can be used to rename the schema record name type. If the record renamed clause is omitted, the record view name clause must specify the same record name as the schema record name type.

The **component view clause** is comprised of an optional component renamed clause and a required component view name clause. If the **component renamed clause** is specified, the **component view name clause** can declare a name for a data element identifier that is different from the associated schema identifier name. If the component renamed clause is omitted, the component view name clause must declare the same identifier name as declared in the schema.

Table 9. *Standards functional specifications for network subschema definition languages*

NETWORK Subschema Definition Language		
	Specifications	
Features	*Standard*	*Implementation Extensions*
subschema	• subschema name clause • record view • set view	• area view
subschema name clause	• dot style subschema • of style subschema clause	• device-media clause
record view	• record renamed • record view name • component view • ALL	• access-control clause
component view	• component renamed • component view name	• access-control clause
set view	• set renamed • set view name	• access-control clause

The **set view clause** contains an optional set renamed clause and a set view name clause. If the **set renamed clause** is specified then the **set view name clause** can be used to declare a different set name than that in the schema definition; however, if the set renamed clause is omitted, the set view name must declare the same set name as specified in the schema definition.

Implementation Extensions. The limitations in the standards specification for the subschema definition language coincide with those for the schema definition language. Thus, implementations have extended the standards specification for the subschema definition language to include the capability to specify access control, storage control, and device types.

C. Module Language and Data Manipulation Language

The combination of these languages is used for declaring the database procedures and executable statements of a specific database application. Both languages serve as extensions to such programming languages as FORTRAN and COBOL. The module language binds a programming language executable module to a database. And, the data manipulation language statements are interpreted as calls from that executable module to the kernel of a data base management system which executes the requested database operations on the database specified by the module language. The standards specifications for the syntax of the module and data manipulation languages are shown in tables 10 and 11, respectively.

Standards Specification. As shown in table 10, the standards document specifies that a module should contain a module name clause, a language clause, a subschema specification, an optional temporary set specifications, and one or more procedures. A single module is associated with an application program during its execution, and an application program may be associated with at most one module. The associated module is assigned a name through the **module name clause**.

The **language clause** identifies the programming language that will be used to write the application source code. The standards document specifies the programming languages: COBOL, FORTRAN, PASCAL and PL.1.

The **subschema specification** of a module specifies the schema that defines the records and sets that can be referenced by the module.

The **temporary set specifications** of a module defines temporary set types that may be referenced by the module.

The **procedure**, as shown in table 10, consists of a procedure name, a sequence of parameter declarations, and a sequence of NDL statements. The **procedure name** names the procedure.

Table 10. *Standards functional specifications for network module languages*

NETWORK Module Language		
	Specifications	
Features	*Standard*	*Implementation Extensions*
module	• module name clause • language clause • subschema specification • temporary set specification • procedure	• access-control clause • realm clause
module name clause	• module name	
language clause	• COBOL • FORTRAN • PASCAL • PL1	
subschema specification	• dot style subschema specification • of style subschema specification	
temporary set specification	• temporary set specification • set view name	
procedure	• procedure name • parameter declaration • NDL statement	

A **parameter declaration** contains a parameter name, a data type and an optional occurs clause.

An **NDL statement** is any one of the data manipulation language statements (shown in table 11): commit, connect, disconnect, erase, find, get, modify, nullify cursor, ready, reconnect, rollback, store, and test.

Implementation Extensions. Since a module is dependent on a subschema, implementations have to extend the standards specification for a module with functional capabilities that will handle access and storage controls. Current implementations of data manipulation languages do not contain any extensions.

II. Relational Data Model

The standards functional specifications for the relational data model are shown in tables 12 through 14. As shown, the standards committee categorizes the functional specifications into three categories: schema definition language, module language, and data manipulation language.

A. Schema Definition Language

As stated for the network model, the function of the schema definition language is to declare the structures and integrity constraints of a relational database. The difference is in the underlying data model and the syntactical composition.

Standards Specification. As shown in table 12, the schema definition language for the relational data model is comprised of a schema authorization clause, table definition, column definition, unique constraint definition, view definition, and privilege definition.

The function of the **schema authorization clause** is to protect the database from unauthorized use.

The **table definition** defines the basic structure for the relational data model This structural definition consists of a table name and one or more column definitions. Each column has an ordinal position within the table.

The **table name** names a table.

Table 11. *Standards functional specifications for network data manipulation languages*

NETWORK

Data Manipulation Language

	Specifications	
Features	*Standard*	*Implementation Extensions*
commit statement	• COMMIT • FINISH	
connect statement	• database key identifier • set view name	
disconnect statement	• database key identifier • set view name	
erase statement	• database key identifier • cascade specification	
find statement	• find specification • find intent • find cursor disposition	
get statement	• to parameter move	
modify statement	• to database move	
nullify cursor statement	• database key identifier	
ready statement	• record view name • share specification • access intent	
reconnect statement	• database key identifier • set view name	
rollback statement	• ROLLBACK • FINISH	
store statement	• to database move • store retention	
test database key equal statement	• database key identifier	
test database key null statement	• database key identifier	
test set empty statement	• set view name	
test set membership statement	• set view name • database key identifier	
database key identifier	• session • record view name • set view name	
component view identifier	• dot style component view ID • of style component	
parameter identifier	• parameter name • subscripts	
to parameter move	• record view name • to parameter move clause	
to database move	• record view name • to database move clause	

Table 12. *Standards functional specifications for relational schema definition languages*

RELATIONAL		
Schema Definition Language		
	Specifications	
Features	*Standard*	*Implementation Extensions*
schema	• schema authorization clause • table definition • view definition • privilege definition	• index definition • synonym definition • cluster definition • partition definition • space definition
table definition	• table name • column definitions • constraint definitions	• space allocation
column definition	• column name • data type • not null • unique	• default
data type	• character • numeric • decimal • integer • small init • float • real • double precision	• long (or text) • date • time
constraint definition	• unique column list	• referential integrity
view definition	• table name • view column list • query specification • check option	
query specification	• select list • table expression	
table expression	• from clause • where clause • group by clause • having clause	• connect by clause • start with clause
privilege definition	• privileges • table name • grantee • with grant option	
privileges	• all privileges • select • insert • delete • update	• alter • index • connect • resource • DBA • references

The **column definition** names a column and specifies a description for that column. The **named column** is a column of a named table from a table definition, and its description assigns a data type along with an indication as to whether the column is constrained to contain only non-null values. The description of a character string column specifies its length attribute, and the description of an exact numeric column specifies the precision and scale of its numbers.

The **unique constraint definition** specifies a list of column names in a table such that no two rows are duplicates with respect to the designated columns. The column definition for each column name in the column list must specify NOT NULL.

The **view definition** specifies selected columns and the data values within them. These columns and data values are selected from one or more tables.

The **privilege definition** defines the database access privileges that are available to specified users. The privileges are defined through the use of the GRANT command. The set of privileges involve such action operators as SELECT, INSERT, DELETE and UPDATE. These operations are granted on a specified column list and the privilege to use them can be extended to the PUBLIC or identified users.

Implementation Extensions. As shown in table 12, there are implementation extensions in almost all of the standards feature categories. The extensions to the schema feature include such capabilities as defining indexes, synonyms, clusters, partitions and space allocations. The index definition establishes a more rapid access path for queries. The synonym definition assigns a shorter schema name to the initially created schema name which is generally prefixed with the owner name. The cluster definition improves system performance by taking logically related rows from separate tables and storing them together on the same disk page. The partition definition controls the distribution of data across separate disk drives (physical devices) to balance the I/O load. The space definition controls the initial storage allocated for tables, their growth rate, and their maximum growth.

The extension to the table definition is the space allocation which is determined by the associated space definition.

The "default" extension for the column definition is an automatic "data value" assignment for a column. For example, if a record is inserted without a value for that column, the default value will be assigned to that column.

The extension to the constraint definition has to do with referential integrity. This capability is not readily available in the commercial DBMS market place; however, it is being considered by the standards committee and is expected to pick up steam in the commercial market as well. The intent of referential integrity is to declare and maintain relations between different record types, based on matching column values. A major concern that supports the need for referential integrity is to prevent deletions from corrupting a database.

There are also such extended privileges as alter, index, connect, resource, DBA, and reference. The alter privilege controls modifying the table definition. The index privilege allows defining a more rapid access path to the table. The connect privilege establishes a user of the database. The resource privilege allows a user to create tables in the database. The DBA privilege gives a user the authority to specify privileges for other users. The reference privilege controls the references that can be made by one user against another user's tables.

Extensions to the "data type" feature include such data types as: long (text), date, and time. The long data type is generally used for text strings that are longer than the maximum string length for the character data type.

The "table expression" feature is extended by the "connect by" and "start with" clauses. These clauses are used together to manipulate the relational database in tree structure fashion.

B. Module Language and Data Manipulation Language

Together, the module and data manipulation languages are used to declare database procedures and executable statements for a specific application. Both languages serve as extensions to such programming languages as FORTRAN and COBOL. The module language binds a programming language executable module to a database. And, the data manipulation language statements are interpreted as calls from that executable module to the kernel of a data base management system which executes the requested database operations on the database specified by the module language. The standards specifications for the syntax of the module and data manipulation languages are shown in tables 13 and 14, respectively.

Standards Specification. As shown in table 13, the module language is comprised of a module name clause, a language clause, a module authorization clause, cursor declarations, and procedures.

The **module name clause** names a module. The module name must be different from the module name of any other module in the same environment where the concept of environment is implementation-defined.

The **language clause** should specify one of four possible languages: COBOL, FORTRAN, PASCAL, or PL1.

The **module authorization clause** should contain the authorized database access identifier.

The **cursor declaration** should define a cursor name and any associated specifications. And, for each declared cursor in a module, there must be exactly one procedure in that module that contains an open statement that specifies the cursor name declared in the declare cursor statement.

Table 13 shows that a **procedure** is comprised of a procedure name, parameter declaration, and SQL statement.

Table 14 shows that the **data manipulation language** is comprised of the list of statements: close, commit, declare cursor, delete, fetch, insert, open, rollback, select, and update.

Implementation Extensions. There are no widely implemented extensions to the module and data manipulation languages. The "language clause" and "SQL statement" are examples of areas in which extensions have been implemented for the module language category. For the language clause, the programming languages C, Assembly, and Ada are being used in addition to those specified by the X3H2 standards committee. A "lock statement" is implemented on some systems as an additional SQL statement.

An extension to the data manipulation language category is the capability to perform "two phase" Commit operations.

Additionally, some implementations have extended the select statement with such clause as: connect by, start with, and repeating subqueries.

Table 13. *Standards functional specifications for relational module languages*

RELATIONAL		
Module Language		
	Specifications	
Features	*Standard*	*Implementation Extensions*
module	• module name clause • language clause • module authorization clause • cursor declaration • procedure	
module name clause	• module name	
language clause	• COBOL • FORTRAN • PASCAL • PL1	• C • Assembler Language • Ada
module authorization clause	• authorization identifier	
cursor declaration	• cursor name • cursor specification	
procedure	• procedure name • parameter declaration • SQL statement	
parameter declaration	• parameter name and data type • SQLCODE parameter	
SQL statement	• close statement • commit statement • declare cursor • delete statement: positioned • delete statement: searched • fetch statement • insert statement • open statement • rollback statement • select statement • update statement: positioned • update statement: searched • whenever statement	• lock statement

Table 14. *Standards functional specifications for relational data manipulation languages*

RELATIONAL

Data Manipulation

	Specifications	
Features	*Standard*	*Implementation Extensions*
close statement	• cursor name	• two phase
commit statement	• commit work	
declare cursor	• cursor name • cursor specification	
delete statement: positioned	• table name • cursor name	
delete statement: searched	• table name • search condition	
fetch statement	• cursor name • fetch parameter list	
insert statement	• table name • insert column list • insert value list • query specification	
open statement	• cursor name	
rollback statement	• rollback work	
select statement	• select list • select parameter list • order by clause • subquery • correlated subquery • group by • having clause	• connect by clause • start with clause • repeating subquery
update statement: positioned	• table name • cursor name • set clause: searched	
update statement: searched	• table name • set clause: searched • search condition	

III. Data Dictionary Systems

In 1980, both the Accredited Standards Committee (ASC) X3H4 and the National Bureau of Standards (NBS) of the United States Department of Commerce initiated efforts to develop standards for dictionary software. From these initiated efforts, a draft proposed standard has been developed for an "Information Resource Dictionary System" (IRDS). The documented specifications were based on the analysis of relevant literature and existing commercial and Federally-developed dictionary systems. Features and capabilities in the current generation of dictionary systems and those projected by technology trends were included in the specifications.

A. Current Features

The draft proposed standard includes several feature categories. A list of these categories is shown in table 15.

Table 15. *Capabilities of the IRDS*

CURRENT FEATURES

- command language and panel interface facilities
- dictionary maintenance
- dictionary output
- schema maintenance
- schema output
- entity-list facility
- IRD-IRD interface facility
- view facility
- versioning facility
- basic functional schema
- IRDS security facility
- extensible life-cycle-phase facility
- procedure facility
- application program (call) interface

The **command language and panel interface facilities** are the proposed forms of user interface. The Command Language supports user interaction with the IRDS in both batch and interactive modes. The panel Interface provides the IRDS user with a set of logical screens (or panels).

The **dictionary maintenance category** contains those functions that should be available for updating the contents of a dictionary. The standards specify the syntax and rules for such operators as add, modify, delete, and copy. These operators are used to operate on entities and relationships which are contained in the content of a dictionary.

The **dictionary output category** comprises those functions that are used to query the dictionary. The standards specify three output functions: general, impact-of-change, and syntax. The general output function produces output on IRD entities, their associated relationships, and the attributes of these entities and relationships. The impact-of-change function, reports on those entities that might be affected in some manner by a change to a specified entity. The syntax output function produces output on selected entities in the same format as that used to create the entities using the Command Language Interface.

The **schema maintenance category** contains those functions that should be available for making changes to the schema structure. Thus, a user should be able to manipulate and redefine the schema by adding, modifying, and deleting meta-entities and meta-relationships.

The **schema output category** contains those functions that produce generalized output on the contents of the schema (i.e., the meta-entities and meta-relationships).

The **entity-list facility** is used to create and manipulate lists of access-names based on user-specified selection criteria. These "entity-lists" serve as input to output functions and certain maintenance functions.

The **IRD-IRD interface facility** controls the movement of data from one IRD to another. That is, if an organization has two or more standard conforming dictionaries, an interface facility should be available to select and transport some or all of the entities and relationships (along with their attributes) from one dictionary to another.

The **view facility** is used to define a logical subset of a dictionary. Thus, a view defines an environment in which a user works with an IRD. A view should be sharable by many users. And, a user may also have access to many views.

The **versioning facility** assigns to each entity in the IRD a version-identifier (by default, if not explicitly specified). The use of this facility is optional. A complete version-identifier is composed of two parts—a variation-name and a revision-number. The existence of a variation-name is optional—only those entities that have been explicitly assigned variation-names have them. All entities have revision-numbers—a default mechanism allows their specification to be optional.

The **basic functional schema** defines a "starter set" of entity-types, relationship-types, attribute-types, and other schema descriptors.

The **security facility** builds on the view concept to control access to IRD and IRD schema content and functionality. "Global" security allows access control at the command and entity-type level. "Entity-level" security controls access to individual entities in the IRD.

The **extensible life-cycle-phase facility** allows an organization to define life-cycle-phases that correspond to the methodology used by the organization, provides facilities to assign each IRDS entity to one of the defined phases, and enforces integrity rules controlling the movement of entities from one phase to another. Each life-cycle-phase is represented as a meta-entity in the IRD schema.

The **procedure facility** provides a mechanism for defining and executing procedures on the IRDS. IRDS procedures may contain IRDS commands and flow-control or assignment statements, as well as substitutable parts whose values are set at execution time.

The **application program (call) interface** provides an interface from a standard programming language to the IRDS. The interface is accomplished by using the CALL feature of the programming language. The IRDS is thus treated by the application as a subroutine.

B. Planned Extensions

NBS and the ASC X3H4 have identified a number of areas in which current standards specifications will be extended. These are listed in table 16.

A **generalized external software interface** will be developed to: (1) enable external software to directly access the IRD to obtain data declarations for standard programs and languages, and (2) enable the IRDS to extract data descriptions from programs written in standard programming languages or from schemas and subschemas written in standard database languages.

Support for network directory functions will be incorporated into the IRD schema. These facilities will document what exists in the network, what the dependencies are between processes and data in the network, and where in the network the processes and data reside. Functional support must also exist to control traffic management within the network.

The **data management support module** will provide additional support for: (1) standardization of data elements; and (2) the location of data in an IRD when a user does not know the appropriate access-name or descriptive-name. Support for data element standardization must occur throughout the standardization life cycle.

The **integration of Life-Cycle-Phase and Quality-Indicator facilities** is another area of future development. This combination could then be used to determine the "suitability" of moving entities to another phase. Furthermore, there is a desire to treat assemblages of processes and data as a structure so that they could be managed.

The **n-ary module** is needed to support certain complex environments such as those involving control flow, or some aspects of programming and database languages structure semantics.

There is also a **desire to integrate external software packages with IRDS data**. For example, "user exits" allowing insertion of organization-defined procedures into command streams. The IRDS would recognize these non-IRDS procedures and would take appropriate action to execute the inserted commands. This facility will only be appropriate if an organization has the Core Command Language Interface. Other examples include interfaces to a text editor, a report writer, and graphics software.

Table 16. *Planned extensions to the current standards specifications for IRDS*

PLANNED EXTENSIONS
• generalized external software interface
• support for network directory functions
• data management support module
• integration of life-cycle-phase and quality-indicator facilities
• N-ary relationship module

*U.S. GOVERNMENT PRINTING OFFICE: 1987 — 181-076/60004

NBS Technical Publications

Periodical

Journal of Research—The Journal of Research of the National Bureau of Standards reports NBS research and development in those disciplines of the physical and engineering sciences in which the Bureau is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Bureau's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NBS under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NBS publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NBS publications—FIPS and NBSIR's—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NBS Interagency Reports (NBSIR)—A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.