



Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

1990-03

The development of a database management system for library loan management

Park, Seong Seung

Monterey, California. Naval Postgraduate School

<http://hdl.handle.net/10945/30707>

Downloaded from NPS Archive: Calhoun



Calhoun is a project of the Dudley Knox Library at NPS, furthering the precepts and goals of open government and government transparency. All information contained herein has been approved for release by the NPS Public Affairs Officer.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

DTIC FILE COPY

2

NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A226 249

DTIC
ELECTE
SEP 06 1990
D S D



THESIS

THE DEVELOPMENT OF A DATABASE
MANAGEMENT SYSTEM FOR
LIBRARY LOAN MANAGEMENT

by

Seong Seung Park
March, 1990

Thesis Advisor:

Myung W. Suh

Approved for public release; distribution is unlimited

90 09 05 022

REPORT DOCUMENTATION PAGE				
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release: distribution is unlimited.		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE				
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		5. MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b. OFFICE SYMBOL <i>(if applicable)</i> 32	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School		
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000		7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL <i>(if applicable)</i>	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)		10. SOURCE OF FUNDING NUMBERS		
		Program Element No	Project No	Task No
11. TITLE (Include Security Classification) THE DEVELOPMENT OF A DATABASE MANAGEMENT SYSTEM FOR LIBRARY LOAN MANGEMENT				
12. PERSONAL AUTHOR(S) Seong Seung Park				
13a. TYPE OF REPORT Master's Thesis	13b. TIME COVERED From To	14. DATE OF REPORT (year, month, day) March 1990	15. PAGE COUNT 172	
16. SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.				
17. COSATI CODES		18. SUBJECT TERMS (continue on reverse if necessary and identify by block number) Database System Design and Implementation		
FIELD	GROUP			SUBGROUP
19. ABSTRACT (continue on reverse if necessary and identify by block number) <p>This thesis deals with the procedures for and the issues in the analysis, design, and implementation of Library Loan Management System (LLMS). LLMS is a low-volume real-time transaction processing system intended for small or medium size libraries. It is designed to provide such library functions as library cataloging, patron registration, circulation, and reference services based on a relational database management system. We implemented prototype LLMS to run on IBM PC/AT or XT compatible microcomputer using dBASE IV. The developed prototype system has been documented in this thesis. We also discuss some issues in implementing LLMS in a networked environment.</p>				
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS REPORT <input type="checkbox"/> DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION Unclassified		
22a. NAME OF RESPONSIBLE INDIVIDUAL Myung W. Suh		22b. TELEPHONE (Include Area code) (408)646-2637	22c. OFFICE SYMBOL Code 54Su	

Approved for public release; distribution is unlimited.

THE DEVELOPMENT OF A DATABASE MANAGEMENT SYSTEM FOR
LIBRARY LOAN MANAGEMENT

by

Seong Seung Park
Captain, Korean Army
B. S., Kyung-Pook National University, 1982

Submitted in partial fulfillment of the requirements for
the degree of

MASTER OF SCIENCE IN TELECOMMUNICATIONS SYSTEMS
MANAGEMENT


from the

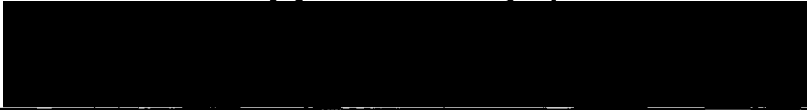
NAVAL POSTGRADUATE SCHOOL
March, 1990

Author:


Seong Seung Park

Approved by:


Myung W. Suh, Thesis Advisor


Gary K. Poock, Second Reader


David R. Whipple, Chairman,
Department of Administrative Science

ABSTRACT

This thesis deals with the procedures for and the issues in the analysis, design, and implementation of Library Loan Management System (LLMS). LLMS is a low-volume real-time transaction processing system intended for small or medium size libraries. It is designed to provide such library functions as library cataloging, patron registration, circulation, and reference services based on a relational database management system. We implemented prototype LLMS to run on IBM PC/AT or XT compatible microcomputer using dBASE IV. The developed prototype system has been documented in this thesis. We also discuss some issues in implementing LLMS in a networked environment. (KR)



Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

DTIC
COPY
INSPECTED
1

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	BACKGROUND	3
	A. LIBRARY MANAGEMENT	3
	1. Organizations	3
	2. Functions	4
	B. DATABASE SYSTEMS	5
	1. Database and Database Administrator	5
	2. Data Models	6
	3. Conceptual Data Models.....	8
	4. Information Architecture	9
	5. System Architecture.....	11
III.	RELATIONAL DATA MODEL.....	13
	A. RELATIONAL MODEL CONCEPTS.....	13
	1. Terminology	13
	2. Characteristics of Relations	14
	3. Key Attributes of a Relation	15
	4. Relational Database Schema	15
	B. RELATIONAL ALGEBRA	16
	1. Projection	17
	2. Selection	17
	3. Join	17
	4. Union	18

5.	Set Difference	18
6.	Cartesian Product	18
C.	RELATIONAL CALCULUS	18
IV.	SYSTEM ANALYSIS	20
A.	INTRODUCTION	20
B.	APPLICATION REQUIREMENT	21
1.	Cataloging Requirements	21
2.	Circulation Requirements	24
3.	Reference Service Requirements	26
V.	SYSTEM DESIGN	28
A.	STEPS IN CONCEPTUAL DESIGN	29
B.	INTRODUCTION TO SEMANTIC DATA MODEL (SDM)	30
C.	GENERAL PRINCIPLES OF DESIGNING SDM	32
D.	SDM DESIGN	32
E.	SDM DESIGN SUPPORT	33
1.	Domain Constraints	34
2.	Intra-relation Constraints	34
3.	Inter-relation Constraints	35
F.	NORMALIZATION	36
1.	Update Anomalies	37
2.	Functional Dependency	37
3.	Normal Forms	38
G.	DESIGNING LLMS WITH SDM	43
H.	E-R DIAGRAM REPRESENTATION	44
VI.	IMPLEMENTATION	47

A.	SOFTWARE REQUIREMENTS	48
1.	Features of dBASE IV	48
2.	Limits of dBASE IV	48
B.	HARDWARE REQUIREMENTS	49
C.	IMPLEMENTATION DESIGN	50
D.	PHYSICAL DESIGN	51
E.	FUNCTIONAL DESCRIPTION	51
1.	The LLMS Main Menu	52
2.	Getting Started	53
3.	Entering Data about New Book	55
4.	Patron Registration	55
5.	Circulation	59
VII.	ISSUES IN A NETWORKED ENVIRONMENT	66
A.	NETWORK COMMANDS AND FUNCTIONS	67
B.	DESIGNING A PROTECTION SCHEME	68
1.	Designing User Profiles	68
2.	Designing File Privilege Schemes	69
C.	FILE AND RECORD LOCKING	70
1.	File Locking	71
2.	Record Locking	71
VIII.	CONCLUSION	72
APPENDIX A.	SDM DESIGN	74
APPENDIX B.	SDM DOMAIN DEFINITION	78
APPENDIX C.	PROGRAM LIST	81
LIST OF REFERENCES	162
INITIAL DISTRIBUTION LIST	163

ACKNOWLEDGEMENT

I would like to give my appreciation to the Republic of Korea which has given the great opportunity for graduate course and physical support. Also, I thank a number of people who helped me. And, I greatly thank Lt Colonel D. S. Hong and Major I. S. Park for their encouragement.

I must thank my thesis adviser, Professor Myung W. Suh, for his eager help and encouragement. I think I could not complete this thesis without his advice and effort. I also would like to express thanks to my second reader, Professor Gary K. Pooch, for his guidance and comments.

I also appreciate my sponsor Jeffrey Carpenter and his family, Wendy and Jordon, who have helped me throughout entire graduate study.

Finally, I give a thanks to my parents, my lovely wife, Young Hee, and my bunnies, Jun Suh and Ye Jin (Judy) who was born during thesis work for their patience and encouragement.

I. INTRODUCTION

As we see in the 1980s, more information is being generated than ever before. Regardless of the state of the economy, the publishing output of the world increases continuously. Nowadays, the crisis facing libraries is the information explosion. In order to manage the huge amount of information, it is imperative to use the database technology for the library.

The database management systems(DBMS) provide us with the following capabilities:

- o Controlling redundancy.
- o Sharing of the database.
- o Restricting unauthorized access.
- o Providing multiple interfaces.
- o Representing complex relationships among data.
- o Enforcing integrity constraints.
- o Providing backup and recovery. [Ref. 1:p. 20]

Although there are several kinds of database models and design concepts, we need to choose the one which best fits the situation. In this thesis, the relational database model will be used to develop the database for the library loan management. Most DBMS experts recommend the relational data model over others which supports data independency and facilitates conceptual modeling.

The database management system developed in this thesis is primarily intended for use by medium and small libraries. These libraries typically do not have sufficient specialized staff, such as systems analyst or computer programmer, to develop the in-house expertise necessary to get the benefits of

automation. This thesis presents a guideline for these libraries to follow in developing an automated system based on database. Automation offers the possibility of expanding the level of service without a corresponding increase in staff that would be required under a manual system. Absolutely, the major impact of the library automation is likely to be new and improved patron services. The implementation of the library loan management is done by using dBASE IV from Ashton-Tate. Multiuser features allow dBASE IV to operate in a networked environment just as easily as in a standalone environment.

Chapter II covers the issues of the library management and gives a general overview of the database systems. Chapter III presents the general concept of the relational database model.

Chapter IV and V deal with the practical system analysis and relational database design for the library loan management. Chapter VI demonstrates the database implementation and functional description. Finally Chapter VII discusses the issues in a networked environment.

Chapter VIII addresses the conclusions and recommendations for this research. Appendix A, B, and C are included that show the design and the application programs.

II. BACKGROUND

A. LIBRARY MANAGEMENT

Libraries reflect the diversity and character of the communities they serve. Excellence in library service is not a simple matter of numbers. It lies in the fit between the library's roles and the needs and expectations of the community it serves.

In the perspective of the past several decades of the twentieth century, libraries in the world have undergone enormous change. With the acceleration of book production and the tremendous increase of information, libraries have long sought technological aids to facilitate and enhance their services.

1. Organizations

The administrative organization of any library must be considered in the light of the structure of the society it represents. If the society aims to do certain things, to achieve certain results, its library must also be organized with these definite purposes in mind.

Figure 1 shows typical functional organization of the library. This traditional type of functional organization provides separate departments of cataloging, circulation, and reference. Sometimes, circulation and reference departments are combined into a single department, because there is insufficient staff to cover the schedules of two separate departments. However, in the large library, there will probably be an acquisitions department in order to do selection and acquisition of materials. [Ref. 2:pp. 38-45]

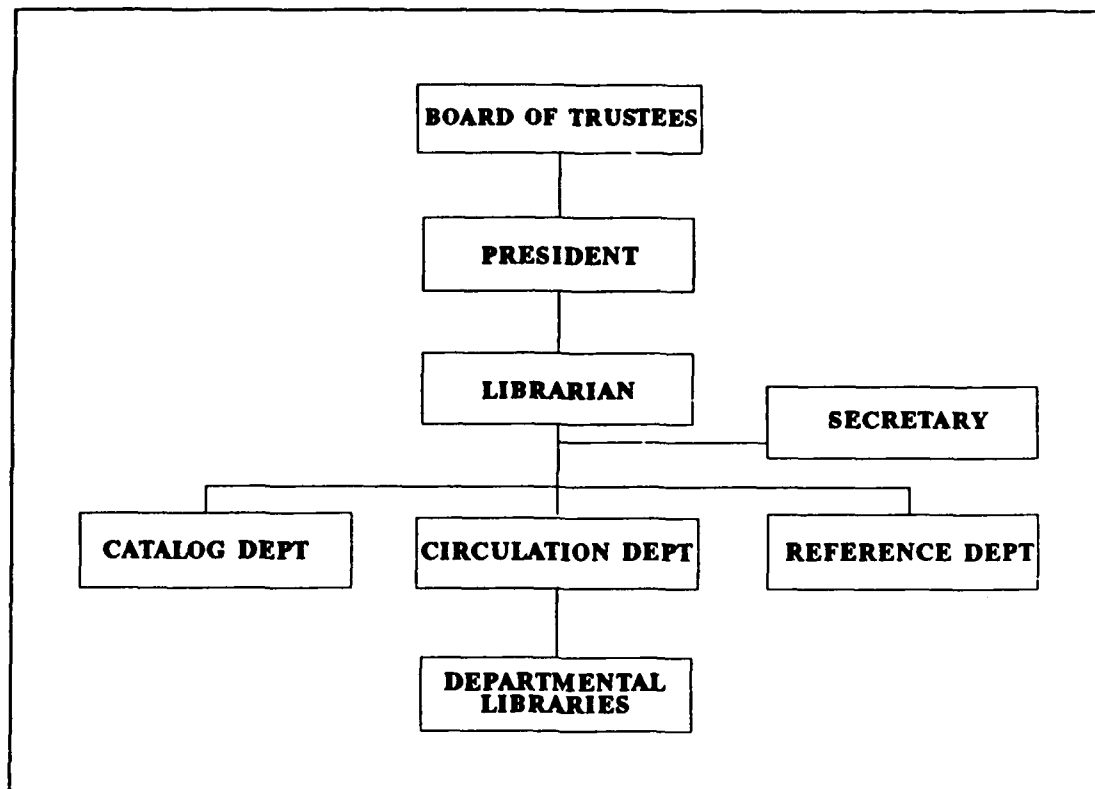


Figure 1. Typical Functional Organization.

2. Functions

No generalization about the functions the library must carry out as part of its role can be completely accurate, but the following list would seem to cover the principal duties:

a. Cataloging – to prepare index cards for all books, journals, and most other types of material added to the library. [Ref. 2:p. 52]

b. Classification – to determine where each new addition belongs in the library's scheme of book arrangement. [Ref. 2:p. 57]

c. Circulation Service – to lend library materials to the legitimate users of the library. [Ref. 2:p. 68]

d. Reference Service – to render full assistance to readers in using

the library and its contents. [Ref. 2:p. 90]

e. Selection – to bring valuable suggestions for augmenting the library's holdings. [Ref. 2:pp. 170–175]

f. Acquisition – to secure the selected books by purchase, gift, or exchange. [Ref. 2:p. 194]

B. DATABASE SYSTEMS

1. Database and Database Administrator

The collection of data representing information of interest is called the database. The database consists of objects, concepts, or event extensions that can be retrieved, modified, inserted, or deleted by users of the database management system.

The data stored in a database must be organized to promote ease in locating any desired piece of datum. Just as the books in a library are arranged on shelves, data in a computer must be organized and arranged well. The database administrator who is responsible for the organization of data in the database determines not only how data are stored in the database, but also what data are to be stored and who may access the data.

The database management system is a collection of computer software used to describe, insert, delete, modify, and retrieve data in the database. Thus, it helps database administrators manage the data in the database. As users insert, delete, and modify data, the data in the database changes. When designing a database, the database administrator cannot have the complete collection of data for the lifetime of the database. Thus, it is necessary for the database administrator to see the things more abstractly by identifying

classes of objects, concepts, and events called data types. For instance, the data instances CAR, TRAIN, and AIRPLANE can be abstracted to the data type VEHICLE. A data model is used to describe the types of data in the database and the relationships between the data types. [Ref. 3:pp. 1-2]

2. Data Models

Data model refers to the style of describing and manipulating data in a database. Data models differ in the style in which data objects and relationships between data objects are described. Many data models have been proposed and each is appropriate for a limited range of applications, but no one data model seems to be suitable for all applications. Here, we will introduce the most common three data models: hierarchical, network, and relational data model.

In the hierarchical data model, records of data are arranged in a hierarchy(Figure 2a). A superior-subordinate relationship may exist between records in this model. There are commands to traverse the relationships between the records, including READ and WRITE commands which transfer data between the program work space and the database. A record in this model may be directly subordinate to at most one other record.

In the network data model, records of data are arranged in a network of relationships(Figure 2b). So, a record may be directly subordinate to several other records in the network. Therefore, hierarchical data model can be thought to be a specific case of the network data model.

In the relational data model, no record can be subordinate to another record(Figure 2c). Special relational data model commands can be used to coordinate records and select records for retrieval and update.

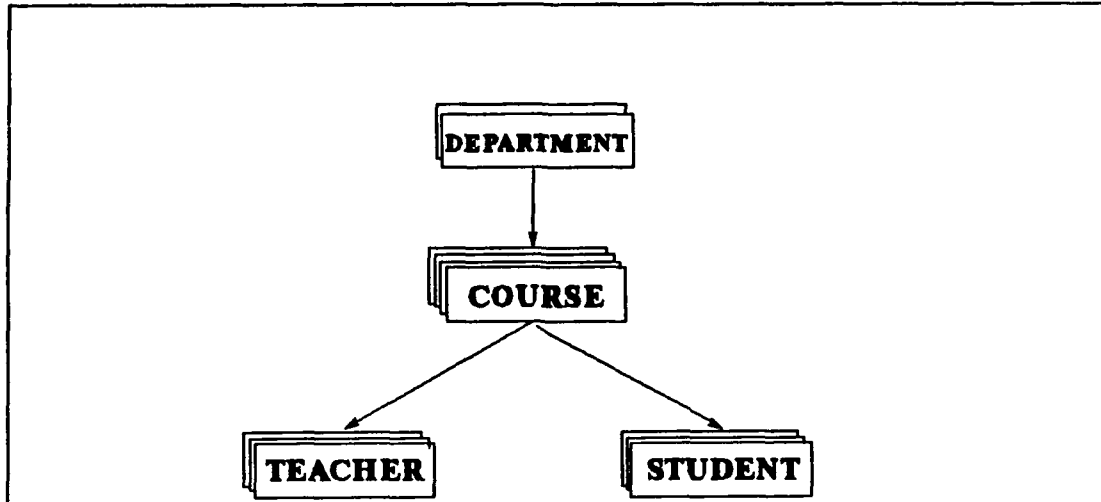


Figure 2a. Hierarchical Data Model

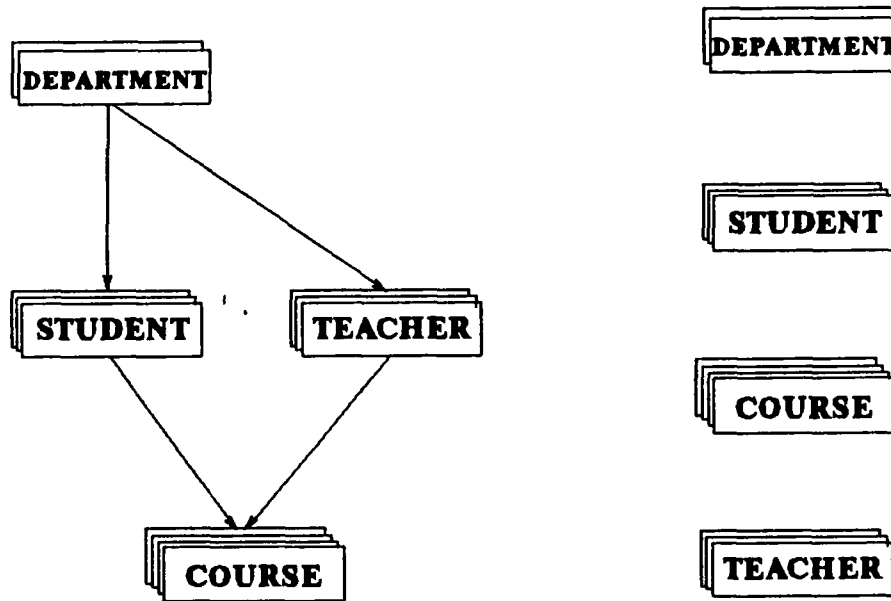


Fig 2b. Network Data Model

Fig 2c. Relational Data Model

Figure 2. Three Data Models.

In some database management systems, data manipulation commands can be entered directly into the computer by using a terminal and without being embedded into a programming language. Such languages are called query languages. Query language is an important factor when we select a database management system because some database management systems support more than one query language. Data models are used to describe schemas which in turn describe data in database. A database management system may have several schemas. [Ref. 3:pp. 2-4]

3. Conceptual Data Models

The data models, hierarchical, network and relational, have been used as the basis for database management systems(DBMS). These data models are too *low level* for adequate modeling of the real world and for producing conceptual schemas. This led to the development of external and conceptual data models. These conceptual data models are independent of the particular internal data model that will be or could be used. Two such models will be introduced here. [Ref. 4:p. 198]

a. *Semantic Data Model (SDM)*

The SDM provides a class of real world semantics which are important in data modeling. In SDM, a database is a collection of entities which may be objects, events, or names which are designators for objects or events. Entities are organized into classes, which are meaningful collections of relevant objects. Each class is either a base class which may be defined independent of other classes, or a non-base class which is defined in terms of other classes using interclass connections. The interclass connection may be subclass, superclass, restrict, subset, merge member, or extract missing

member. The classes are logically connected via interclass connections.

The entities and classes have attributes which relate them to other entities and they describe characteristics. An attribute has a semantic *kind*, which identifies the type of relationship the value of the attribute has with the entity. The value of the *kind* may be one of component, property participant, class determined component, property, or participant. [Ref. 5:pp. 3-4]

b. Entity-Relationship Model (E-R Model)

The real world is modeled in terms of entities, relationships, and attributes. The Entity-Relationship Model is a conceptual model and is based on that real world modeling. Entities and relationships can be represented diagrammatically by an E-R diagram. In E-R diagram the entity sets are represented by rectangular boxes and the relationships by diamonds. Rectangles and diamonds are linked with lines showing the types of the relationship. Attributes of each entity set are drawn in ellipses around their entity set rectangle. [Ref. 4:p. 198]

4. Information Architecture

The functions of data description have been defined thoroughly in the 1970s. A committee of the American National Standards Institute, ANSI/X3/SPARC, published an influential report that separated data description into the three interrelated levels as shown in Figure 3: the conceptual schema, the external schema, and the internal schema.

Data description functions have evolved from application programs and have been stratified into different types of data description called schemas. The types of schemas and their interrelationships are called an information

architecture.

The conceptual schema is a description of types of data useful to the user as a whole, while the external schema represents a masking of the conceptual schema and presents a description of the types of data and their interrelationships of interest to one or more applications. Many external schema can be defined for a single conceptual schema, each for use with a different set of applications. So, programs are insulated from many types of changes in data definition.

The internal schema ensures that data described in the conceptual schema are stored and accessed efficiently. Even though the revisions take place to the internal schema, no changes are required in the conceptual or external schemas. [Ref. 3:pp. 4-5]

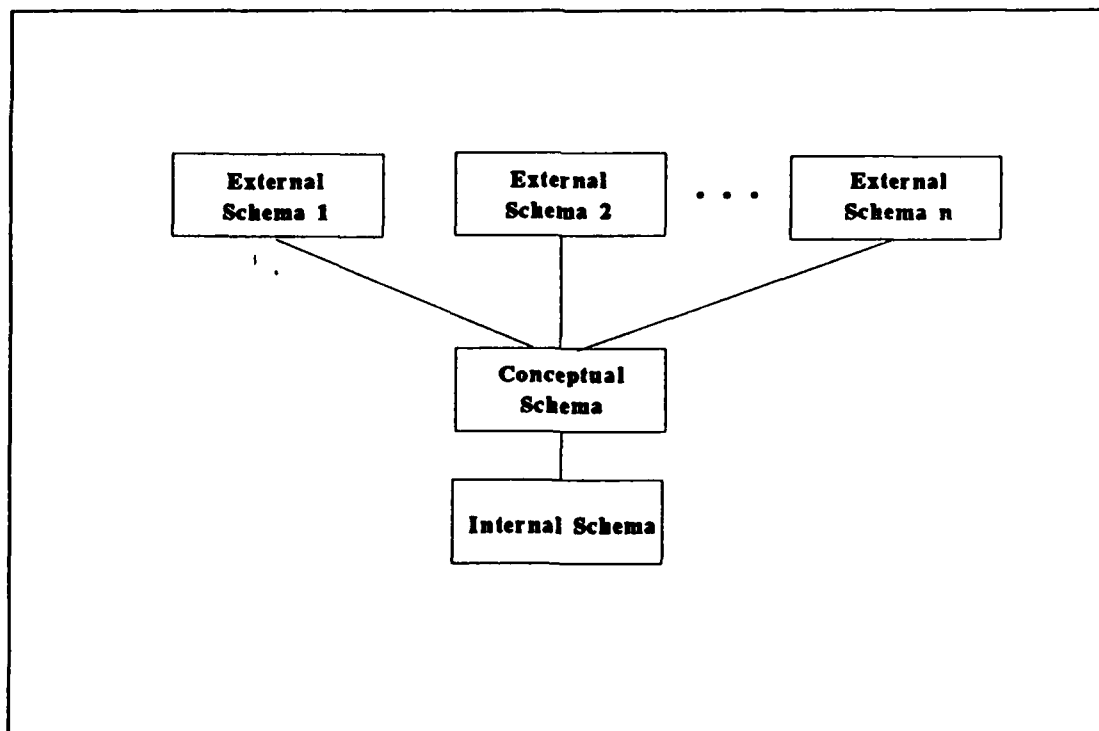


Figure 3. Information Architecture.

5. System Architecture

The various software processes and modules of a database management system and their interrelationship is called the system architecture. Database management systems are popular because they provide useful features for maintaining and accessing data. These features consist of multiple user interfaces, concurrency transparency, program data independence, and transaction atomicity. The user command translator, canonical command translator, and runtime support processor shown in Figure 4 are responsible for supporting these features:

a. Multiple user interfaces – Different database management system users may access the database by using different user interfaces. The user command translator enables multiple user interfaces.

b. Program data independence – When the database is restructured or reorganized, and the application program does not require modification, then it has program data independence. The canonical command translator supports this feature.

c. Concurrency transparency – This feature enables users to believe that each one has sole access to the database even though the DBMS can support several users simultaneously. The runtime support processor is responsible for accepting several requests, scheduling the processing of those requests, and returning the results.

d. Transaction atomicity – This means that each transaction is either completely executed with any modifications made to the database permanently recorded or the transaction is aborted with no permanent change to the database. The runtime support processor is also responsible for

transaction atomicity. [Ref. 3:pp. 5-7]

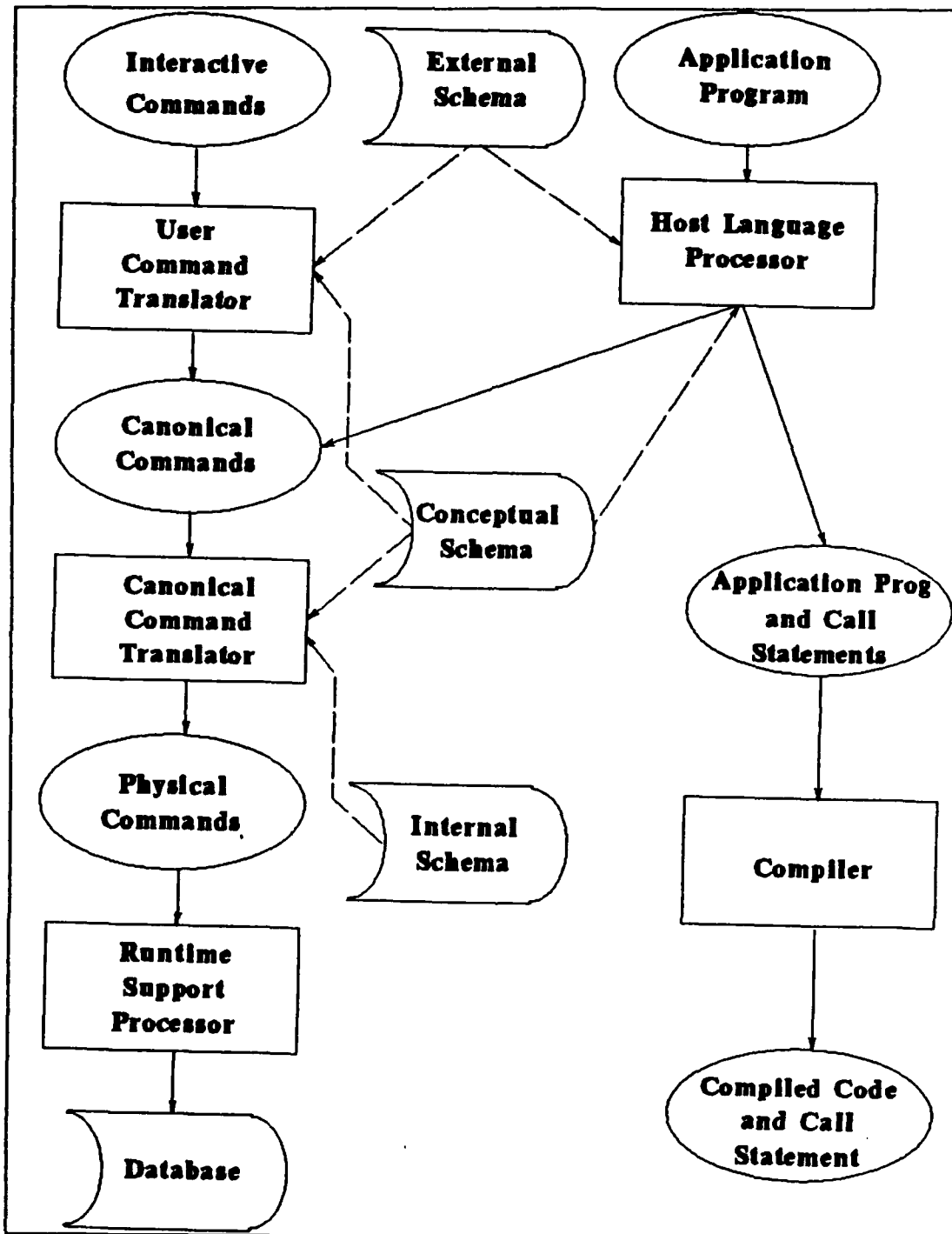


Figure 4. System Architecture.

III. RELATIONAL DATA MODEL

A. RELATIONAL MODEL CONCEPTS

The relational data model has the simplest and most uniform data structures and is the most formal in nature among the three data models presented in Chapter II. The relational model represents the data in a database as a collection of relations.

1. Terminology

When a relation is considered as a table of values, each row in the table represents a collection of related data values. These values can be interpreted as a fact describing an entity or a relationship instance.

A domain is a set of atomic values which are indivisible as far as the relational model is concerned. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn.

A relation schema R , denoted by $R(A_1, A_2, \dots, A_n)$, is a set of attributes $R = \{A_1, A_2, \dots, A_n\}$. Each attribute A_i is the name of a role played by some domain in the relation schema R . A relation schema is used to describe a relation; R is called the name of this relation. The degree of a relation is the number of attributes of its relation schema.

Figure 5 shows an example of a PATRON relation. In relational database terminology, a row is called a tuple, a column name is called an attribute, and the table is called a relation. The data type describing the types of values that can appear in each column is called a domain.

attributes				
PATRON	Name	Address	Phone	SMC
tuples	Jeff Carpenter	22 Mervine Mty	694-1121	1990
	Young H. Paek	1143 Carson SE	394-5380	1888
	Jung H. Park	1144 Carson SE	394-6193	1818
	Chong S. Ryu	333 NPS BOQ	655-1615	2039

Figure 5. The attributes and tuples of a relation PATRON.

2. Characteristics of Relations

The tuples in a relation do not have any particular order. The reason that tuple ordering is not part of a relation definition is that a relation attempts to represent facts at a logical or abstract level. Many logical orders can be specified on a relation. However, there is no preference for one logical ordering over another.

At a logical level, the order of attributes and attribute values are not really important as long as the correspondence between attributes and values is maintained, since the attribute name appears with its value.

Each value in a tuple is an atomic value which is not divisible into components within the framework of the relational model. Thus, composite and multivalued attributes are not allowed. This is called first normal form assumption. Multivalued attributes must be represented by separate relations, and composite attributes are represented only by their simple component attributes.

The relation schema can be interpreted as a declaration or a type of

assertion. Each tuple in the relation can be represented as a fact or a particular instance of the assertion. Some relations may represent facts about entities whereas other relations may represent facts about relationships. Thus, the relational model represents facts about both entities and relationships uniformly as relations. An alternative interpretation as a predicate is quite useful in the context of logic programming languages, such as PROLOG, because it allows the relational model to be used within these languages.

3. Key Attributes of a Relation

A relation is defined as a set of tuples. All elements of a set are distinct and all tuples in a relation must also be distinct. This means that no two tuples can have the same combination of values for all their attributes. Generally speaking, a relation schema may have more than one key. All these possible keys are called the candidate keys. It is common to designate one of the candidate keys as the primary key of the relation. The primary key is used to identify tuples in the relation. We usually underline the attributes that form the primary key of a relation schema. The primary key can be arbitrarily selected from several candidate keys. However, it is more preferable to choose a primary key with a small number of attributes.

4. Relational Database Schema

A relational database usually consists of many relations, with tuples in those relations related together in various ways. When we refer to a relational database, we implicitly include its schema and instance. Relational database schema can be defined as follows;

A relational database schema S is a set of relation schemas $S = \{R_1, R_2, \dots, R_m\}$ and a set of integrity constraints IC . A relational database instance DB of S is a set of relation instances $DB = \{r_1, r_2, \dots, r_m\}$ such

that each r_i is an instance of R_i and such that the r_i 's satisfy the integrity constraints specified in *IC*. [Ref. 1:p. 142]

Integrity constraints are specified on a database schema and are expected to hold on every database instance of that schema. Key constraints specify the candidate keys of each relation schema; candidate key values must be unique for every tuple in any relation instance of that relation schema. In addition to the key constraints, two other types of constraints are considered part of the relational model – entity integrity and referential integrity.

The entity integrity constraint states that no primary key value can be null. This is because we use the primary key value to identify individual tuples in a relation; having null values for the primary key implies that we cannot identify some tuples. For example, if two or more tuples had null for their primary keys, we might not be able to distinguish them.

Key constraints and entity integrity constraints are specified on individual relations. The referential integrity constraint is a constraint that is specified between two relations and is used to maintain the consistency among tuples of the two relations. Informally, the referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation. [Ref. 1:p. 143]

B. RELATIONAL ALGEBRA

The relational algebra is a collection of operations which are used to manipulate whole relations. These operations can be used to choose tuples from individual relations and to combine related tuples from several relations for the purpose of specifying a query on the database. The relational algebra operations include two groups. One group comprises operations developed

specifically for relational databases – PROJECTION, SELECTION, and JOIN operations. The other group consists of set operations from mathematical set theory, which are applicable because each relation is defined to be a set of tuples. These operations are UNION, SET DIFFERENCE, INTERSECTION, and CARTESIAN PRODUCT.

1. Projection

If R is a relation of arity k , we let $\pi_{i_1, i_2, \dots, i_m}(R)$, where the i_j 's are distinct integers in the range 1 to k , denote the projection of R onto components i_1, i_2, \dots, i_m , that is, the set of m -tuples $a_1 a_2 \dots a_m$ such that there is some k -tuple $b_1 b_2 \dots b_m$ in R for which $a_j = b_{i_j}$ for $j = 1, 2, \dots, m$. For example, $\pi_{B,D}(R)$ is performed by taking each tuple in R and the resulting relation has attribute B in its first column and attribute D in its second column.

2. Selection

Let F be a formula involving operands that are constants or component numbers, the arithmetic comparison operators $<, =, >, \leq, \neq,$ and \geq , and the logical operators \wedge (and), \vee (or), and \neg (not). Then $\sigma_F(R)$ is the set of tuples t in R such that when, for all i , we substitute the i th component of t for any occurrences of the number i in formula F , the formula F becomes true. For example, $\sigma_{\text{price}=100}(R)$ denotes the set of tuples in R whose price is equal to 100.

3. Join

The θ -join of R and S on columns i and j written $R \bowtie_{i\theta j} S$, where θ is an arithmetic comparison operator($=, <$, and so on), is shorthand for $\sigma_{i\theta(j)}(R \times S)$, if R is of arity r . That is, the θ -join of R and S is those

tuples in the Cartesian product of R and S such that the i th component of R stands in relation θ to the j th component of S . If θ is $=$, then the operation is called an equijoin.

4. Union

The union of relations R and S , denoted $R \cup S$, is the set of tuples that are in R or S or both. We only apply the union operator to relations of the same arity, so all tuples in the result have the same number of components.

5. Set Difference

The difference of relations R and S , denoted $R - S$, is the set of tuples in R but not in S . This operation also requires that the arities of R and S are the same.

6. Cartesian Product

Let R and S be relations of arity k_1 and k_2 , respectively. Then $R \times S$, the Cartesian product of R and S , is the set of (k_1+k_2) -tuples whose first k_1 components form a tuple in R and whose last k_2 components form a tuple in S .

C. RELATIONAL CALCULUS

Many commercial relational database languages are based on some aspects of relational calculus, including the QBE(Query By Example) and QUEL(Data definition and manipulation language for INGRES relational DBMS). The difference between relational algebra and relational calculus is that relational calculus allows us one declarative expression to specify a retrieval request, whereas relational algebra requires a sequence of operations. In other words,

relational calculus is considered to be a declarative or nonprocedural language because there is no description of how to evaluate a query.

The general expression of the relational calculus has two parts, a target list(tuple variable or domain variable) which consists of a list of the wanted elements, and a condition or formula which defines the wanted elements in terms of the relations from which they are to be retrieved. An expression of the relational calculus is of the form

$$\{ \text{Target list} \mid \text{Condition or Formula} \}$$

to be evaluated as a set of instances that satisfy the specified condition or formula.

The relational calculus is a formal language, based on the branch of mathematical logic called predicate calculus. There are two types of relational calculus. The tuple relational calculus uses tuple variables that range over relations, whereas the domain relational calculus uses domain variables that range over relations.

IV. SYSTEM ANALYSIS

A. INTRODUCTION

The process of identifying and analyzing the intended uses is called requirements collection and analysis. This process is important because it is the blueprint for database design and implementation. Defining requirements for a database has two major tasks. The first task is to identify and describe the objects that the users want to track and define their structure. The best way for this task is to derive the objects' structure by analyzing the application outputs.

The second task is to determine the functional components of each application that will use the database. This task consists of two phases. First, the logical structure of the database which is DBMS-independent is specified. Secondly, the transformation of this logical structure into a design that conforms to the limitations and peculiarities of a given DBMS product is made. This task is designed in parallel with the development of the logical database structure.

The purpose of a database application program is to enable the user to get the needed information about things that are important in his working environment. Each application may include display, update, and control mechanisms for controlling access and processing the database. The display mechanisms include facilities to present the data on a screen, on a printer, or send them to another device. With the update mechanisms the users can add new records, modify existing records, or delete unwanted ones. Finally, the

control mechanisms control the database processing and the accessed data. This means the application programs can include authorization routines to assign different rights to each user or routines which restrict the user's access and processing options. [Ref. 6:pp. 42-55]

After collecting and analyzing application requirements, they must be reviewed by the users before they will be used.

B. APPLICATION REQUIREMENT

The scope of this section is to state the user application requirements for further development of the conceptual or logical structure of the database system and the application programs by describing the main tasks and the different factors that affect the library activities.

The problem of the library can be divided into the following requirements categories:

- o Cataloging requirements.
- o Circulation requirements.
- o Reference service requirements.

Each category will be examined and analyzed in order to identify and describe the data that is required by the library.

1. Cataloging Requirements

When librarians confront the issue of cataloging, there are a number of topics which must be addressed. These include: What data elements or fields should be included in the bibliographic record? Does this library adhere to the MARC (Machine-Readable Cataloging) standard? What is nearly necessary for a specific type of library (academic, school, public, or special)?

For all books added to the library, librarians must prepare initial catalog record. Cataloging work combines both professional duties and a mass of routine detail. The following sections describe the cataloging work.

a. Classification

A realistic approach as to which system to adopt in a particular library requires distinguishing between only two - the Dewey decimal classification and the Library of Congress classification. While by far the largest number of public libraries still use the Dewey decimal classification, most academic libraries use the Library of Congress classification. The Library of Congress classification has more benefit that is practical to connect into a national network for remote, instantaneous access to a central bibliographical store.

Regardless of what classification scheme a library uses, it is good for the librarian to grasp clearly the purposes his classification scheme is trying to serve and to reflect his conviction in the policies which govern its application and interpretation in a particular library. Otherwise some critics will question him and the catalog librarians may have very severe problems. Among those policies the following may be considered:

(1) In classifying books newly added to the library, the catalog department should accept the Library of Congress numbers.

(2) Classification is not an exact science. In many cases there is no right way to classify a particular title. Classifiers, and even subject specialists, will frequently disagree about the exact number to assign a certain book. This is because books deal with subjects and parts of subjects in different and unpredictable ways. A book may not fit nicely into any of the categories of a rigid classification scheme.

(3) It is recognized that the classification numbers assigned to certain books, or groups of books, may become outmoded with the expansion of subject fields and the development of new areas of specialization. Logically such changes might be used as justification for continuous

reclassification of a library's holdings in many fields.

(4) Any classification scheme by itself will be inadequate as a guide to the contents of a library. Many books deal with more than one subject, yet a book can stand in only one place on the shelves. Classification must be supplemented by other approaches to books.

b. *Subject Headings*

There is a close relationship between classification and the subject headings in the card catalog. A reasonable organization of work may well provide that the person classifying books in the library should also be responsible for assigning subject headings. The official record of the subject headings used in the library and the standard published lists are the essential tools for assigning such headings and making subject reference cards. The official list may be in the form of the library's own authority file, with cross-references used, but this is expensive and may differ from the authority in other libraries. A simpler method is to mark a copy of a standard subject heading list, with a check for every subject heading when first used, and to indicate similarly every subject reference as it is made.

c. *Descriptive Cataloging*

Descriptive cataloging is the process which relates to the choice of entries and bibliographic description of books as distinguished from the processes of classification and assigning subject headings. It involves making a copy or process slip which, after the call number and subject headings are added, contains all the information needed for cataloging a title fully.

d. *Revision*

Changing contents in the catalog which are already made is an accepted part of cataloging routine in the optimistic struggle to make the catalog consistent, accurate, and up to date. The work of revision requires

both competence in all phases of cataloging and familiarity with the cataloging and classification policies of the particular library.

e. Preparation of Book Lists

Providing book lists helps the librarian determine which books are missing from a library's collection, and aids the user to find out the information which he needs. [Ref. 2:pp. 52-65]

2. Circulation Requirements

Once the library has acquired and cataloged its books, its subsequent obligations are circulation services and reference services. Circulation service is to lend the library's collections to the legitimate user of the library which is called *patron*. By far the greater part of the book stock is circulated for home use, and the use of some collections of the library must be restricted to the library building. These are reference books, bibliographical sets, and heavily used journals. Following features are required to control the circulation of library materials effectively.

a. Patron Management

Academic libraries make loans to faculty, students, employees and their families. A patron has a unique identification number. Patrons can borrow library materials and use the facilities of the library. The database system must maintain each patron's record by registering patron and by keeping track of any change on patron's data. Then the system can detect delinquent borrowers at time of check out and determine which patron is borrowing a certain book.

b. Circulation Policies

Most libraries have a handbook in which the regulations of the

library on loan periods, renewals, overdue charges, and the like are set forth. The normal charge period is two weeks, and sometimes books need not to be returned before the end of the term unless requested by another reader or for reserve. If in circulation, a book may be *reserved* and the patron requesting it will be notified when it is available. Whatever the period of circulation, the library reserves the right to recall a book for the use of another reader after it has been checked out for two weeks. Patrons other than students may be given special privileges with the understanding that all material charged out will be returned or renewed at the end of a term. In this case of special privileges, it is understood that any specific title may be recalled for the use of another borrower at the end of two weeks.

Fines and the failure to return books are another concerns of the circulation. Fines are imposed not to increase the library's income but to discourage the monopolization of books so that other readers may have a chance to use them also. There is no best way for impressing borrowers with the need for returning books promptly when they are due, even though there are some libraries which do not charge fines and seem well satisfied with their handling of overdues.

c. Charging Systems

A charging system must show what books are charged out to readers and when books charged out are due for return. And it is desirable that it should enable the library to check all the material a borrower has out, with reasonable speed and economy.

A manual charging system consists of filing, charging and discharging of records, and other routine tasks such as the handling of

overdues done by hand. This is the simplest and least expensive system in which two filing cards are employed: (1) a book card, which is arranged in the charge-out file by call number, to show who has a book out; and (2) a date due card, which is arranged in a second charge-out file to show daily what books are overdue. Some libraries prepare and insert two cards in the pocket of the book, one each for the purposes mentioned. [Ref. 2:pp. 68-84]

The move from a manual system to an automated system requires a radically different approach to the problem. The computer offers such an approach by storing, sorting, selecting, searching, and printing large volumes of data. A patron's ID, a book ID, and date due information are brought together in charging out a book, and processed automatically. The advantages of the system are that it relieves the borrower of filling out a charge card for each book he takes out, speeds up charging, relieves the staff of monotonous routine, and does not bury errors as a manual system may.

3. Reference Service Requirements

Reference service is to render full assistance to patrons in using the library and its contents. The function of the reference service is: (1) to provide answers to specific informational questions, (2) to give personal guidance in the use of the library-catalog, bibliographies, abstracts, and indexes, (3) to consult about term papers, theses, and so on, including methods of finding material and bibliographic form, and (4) to give formal instruction in the use of the library.

The library's usefulness in answering both factual and topical reference questions is the ability of the staff to see that the patron receives the information he wants or the knowledge of where to get it as quickly and

precisely as possible. The reference service must help patrons to find what they want. Often it is much easier to find the reference for a patron than to tell him how to go about finding it for himself. [Ref. 2:pp. 90-104]

V. SYSTEM DESIGN

Database design is the process of developing database structures from user requirements for the data. Since database design is a complex and difficult process, this process must be performed by an organized methodology. Teory and Fry (1982) have developed a general model for database design, defining four major steps in the design process. [Ref. 4:pp. 212-213] Conceptual design is the second step in database design, following requirement analysis.

During the conceptual design process the development team makes use of a data model that can support all the applications. This process can be accomplished by defining the entity classes, identifying the relationships among the entities, and transforming the entities into relations. Then the project team must review the established relations and apply the rules of normalization to those relations to find existing anomalies. When anomalies are found, the team modifies the design to eliminate them. [Ref. 6:pp. 210-213]

The conceptual data model is defined by the data themselves and is independent of the application programs, the database management system, computer hardware, or any other physical considerations.

There are two design approaches dealing with the conceptual design: top-down and bottom-up. The designer, in the first approach, builds the enterprise model and then adds details to it until a satisfactory conceptual design has been achieved. For that reason, this approach is also called entity analysis. In the second approach, the designer starts with a detailed

requirements analysis and proceeds to build each user's view separately. The conceptual schema is then formed by merging the relations from each user's view. The bottom-up approach is preferable since in the top-down approach there is no assurance that all user requirements have been represented in the conceptual schema. [Ref. 4:pp. 245-249]

A. STEPS IN CONCEPTUAL DESIGN

In the conceptual design, five major steps must be performed, even though the detailed steps are dependent on the approach or methodology of a user. These major steps are shown in Figure 6 and are described below:

1. **Data Modeling.** The process of identifying and structuring the relationships between data elements is called data modeling.

2. **View Integration.** In this step, the structured relationships from

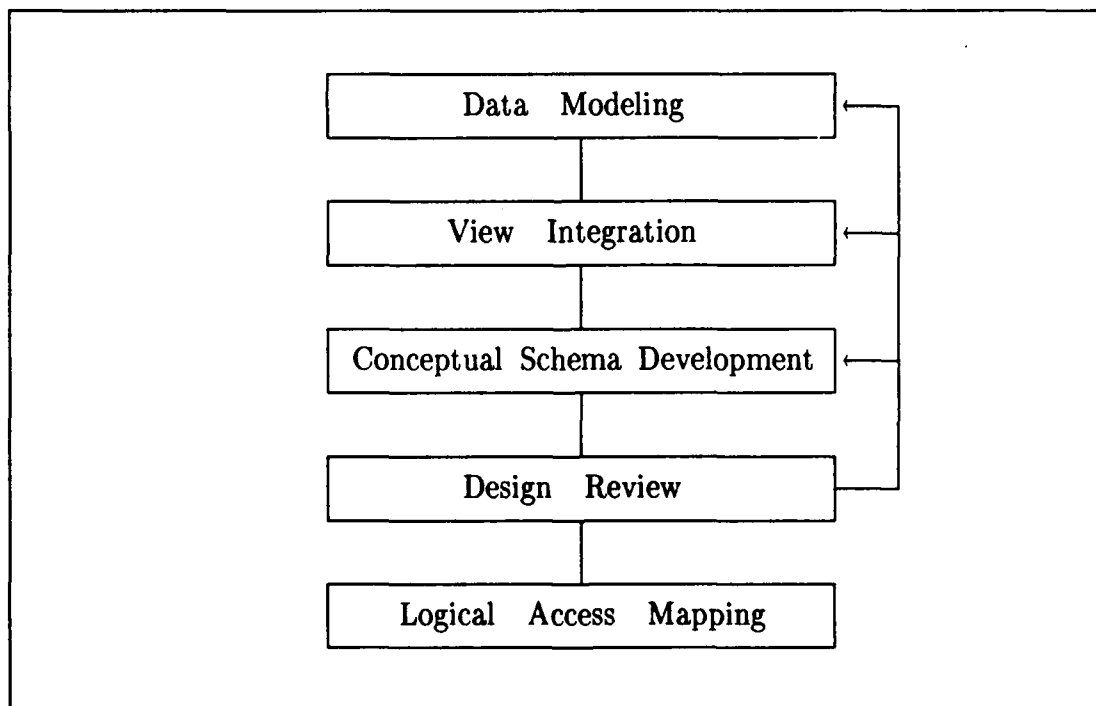


Figure 6. Steps in conceptual design.

the above step are merged together in a single set of relations. The result of this step is the conceptual data model expressed in the form of normalized relations.

3. Conceptual Schema Development. The conceptual schema development step takes the conceptual data model and transforms it into a graphical model for better understanding. Entity-relationship diagram will be used as a graphical model in this chapter.

4. Design Review. When the conceptual data model is developed, the managers and key users should evaluate it and suggest changes or improvements before the implementation design is attempted. The evaluation has two points of view: accuracy and completeness.

5. Logical Access Mapping. The last step in conceptual design is logical access mapping. In this step, diagrams showing the logical sequence of accessing the conceptual records are drawn. Logical access mapping may be considered part of conceptual designer a step in implementation design. [Ref. 4:pp. 249-253]

This chapter will cover these design steps. Data modeling will be performed by means of Semantic Data Model (SDM). The normalized relations from the SDM will be transformed into an Entity-Relationship graphical model. Then design review will be performed and logical access mapping will be part of the implementation.

B. INTRODUCTION TO SEMANTIC DATA MODEL (SDM)

The SDM was developed by Hammer and McLeod and can express a conceptual database design. The SDM allows the same information to be viewed in several ways. [Ref. 7:pp. 351-356]

The database designers that deal with SDM should define a conceptual structure for each of the real world structures. Each database is a model of some real world environment. The real world has some primitives; phenomena that can be represented by nouns as objects. Each object has properties. A property is a characteristic of the object and the collection of all possible

values of a property is called a property value set. A particular value from the property value set for a given property and object is called a fact. The last term that is associated with a database for the real world is the relation of the objects, called associations.

When we design or process a database, we are not working with the above described real world primitives, but we represent these primitives using conceptual terms. Database experts have defined a conceptual primitive for each of the real world primitives. An entity is a conceptual representation of an object having properties which are called attributes. The collection of all values that an attribute can have is called domain. Value is the representation of a fact. Finally, a relationship is the conceptual representation of an association. Figure 7 shows the equivalencies between real world and conceptual primitives. [Ref. 8:p. 209]

REAL WORLD PRIMITIVES	CONCEPTUAL PRIMITIVES
Object	Entity
Object Class	Entity Class
Property	Attribute
Property Value Set	Domain
Fact	Value
Association	Relationship

Figure 7. Real World and Conceptual Primitives.

C. GENERAL PRINCIPLES OF DESIGNING SDM

The following general principles of database organization underlie the design of SDM.

1. A database is to be viewed as a collection of entities that correspond to the actual objects in the application environment.
2. The entities in a database are organized into classes that are meaningful collections of entities.
3. The classes of a database are not, in general independent, but rather are logically related by means of interclass connections.
4. Database entities and classes have attributes that describe their characteristics and relate them to other database entities. An attribute value may be derived from other values in the database.
5. There are several primitive ways of defining interclass connections and derived attributes, corresponding to the most common types of information redundancy appearing in database applications. These facilities integrate multiple ways of viewing the same basic information and provide building blocks for describing complex attributes and interclass relationships. [Ref. 7:p. 355]

D. SDM DESIGN

SDM is a form to synthesize the various user's views and information requirements into database design using a data model form. Two such forms will be used for the database design: SDM and E-R diagram. The latter form will represent the normalized SDM design in diagrams for better understanding according to the conceptual schema development step.

Figure 8 shows the SDM description schema. This will be used to synthesize the entity classes resulting from the requirement analysis into a SDM form. The SDM design is presented in Appendix A.

Each attribute has a name, a description, a value class, and a set of

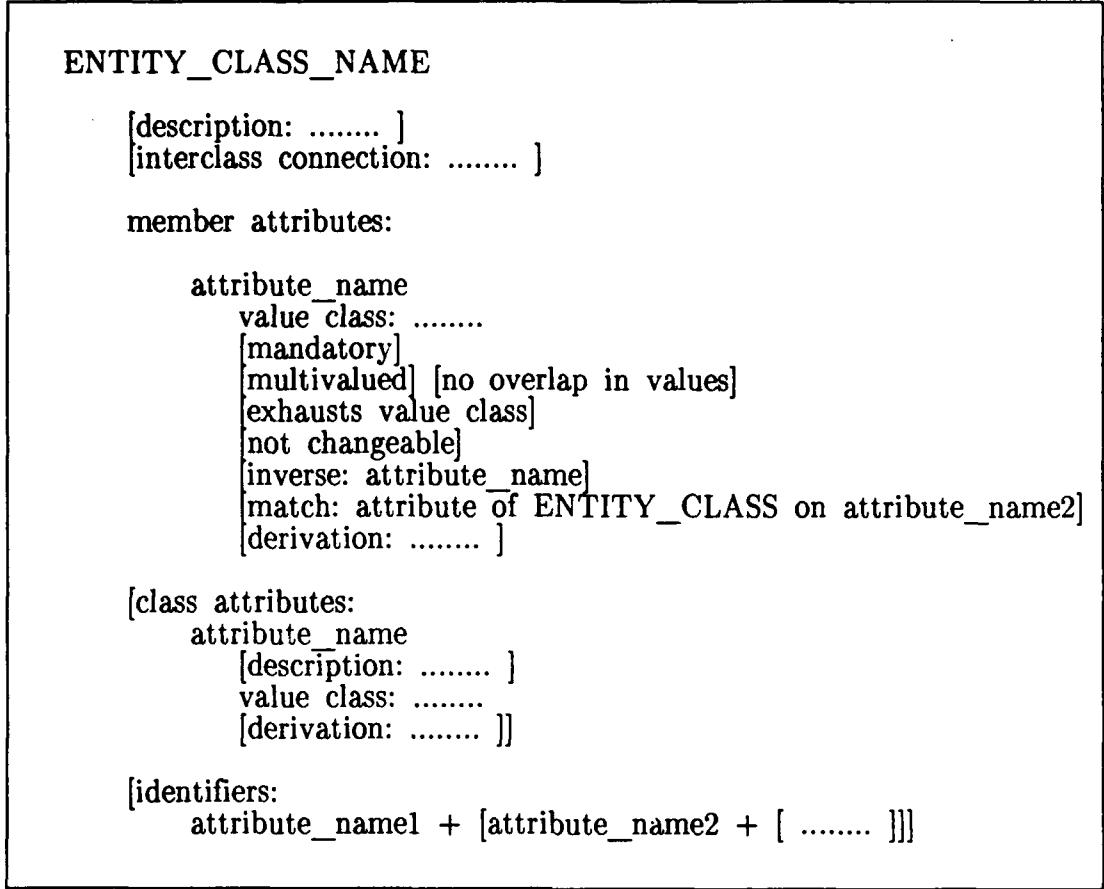


Figure 8. SDM Entity Class Description.

characteristics as shown in Figure 9. The name and the value class are mandatory in each attribute. The value class is the set of values that the attribute can have. In other words, it is another term for domain that is used by the SDM. [Ref. 8:p. 219] The value class and the set of characteristics that each attribute uses to represent the object properties form the constraints of the database design.

E. SDM DESIGN SUPPORT

Constraint is a rule about the data or its relationship to other data in the database. Three types of constraints may be expressed in the conceptual

<u>Descriptor</u>	<u>Status</u>	<u>Remarks</u>
Name	Mandatory	Initial capital letter
Description	Optional	Remarks about attribute
Value class	Mandatory	Domain of the attribute
<u>Descriptor Characteristics</u>		<u>Default Value</u>
Single or multivalued		Single
Value optional or mandatory		Optional
Changeable or not changeable		Changeable
Exhaustive or nonexhaustive		Nonexhaustive
Overlapping or nonoverlapping		Overlapping

Figure 9. Attribute Descriptors in SDM.

design: domain constraints, intra-relation constraints, and inter-relation constraints. These constraints will be used in the design of the LLMS. [Ref. 6:p. 369]

1. Domain Constraints

The domain constraints state the allowed data values that can be accepted by the value class of each attribute. The statement of allowed data values includes the data type(character, numeric, logical, date, and memo), the maximum length of data, a description of the allowable range of data values, or a discrete set of allowable values. The SDM domain definition for LLMS is given in Appendix B.

2. Intra-relation Constraints

An intra-relation constraint states that the characteristics of the data within a table. The set of characteristics that are used by the SDM design are: [Ref. 8:pp. 219-221]

a. Single or multivalued. The value of an attribute can be single or multivalued (like a repeating field).

b. Value optional or mandatory. An attribute can be specified as mandatory which means an accepted value must be inserted, that is, a null value is not allowed from the conceptual point of view. For instance, BOOK_NAME attribute in BOOKINFO entity class is specified as *mandatory*; this means that every book has a BOOK_NAME.

c. Changeable or not-changeable. An attribute can be not-changeable, meaning that the value of the attribute cannot be altered except to correct existing error. For example, PATRON_ID attribute in PATRONINFO entity class is specified as *not changeable*; since each patron has a unique identification number and it cannot be changed.

d. Exhaustive or non-exhaustive. Exhaustive means that every member of the value class of the attribute must be used.

e. Overlapping or non-overlapping. Overlapping means that a member of the value class of the attribute can be used more than one time.

These constraints indicate the characteristics of each attribute. In the SDM design phase, these constraints have been used. The attributes with no constraints are assumed to have the default value as shown on Figure 9.

3. Inter-relation Constraints

Inter-relation constraints state the relationship of data values between or among tables. SDM provides three facilities to express the inter-relation constraints:

a. Inverse. The inverse facility allows two entities to be contained within each other. Each entity class specifies the inverse with the other entity class. For that reason, the inverses are always specified in pairs. Although this is physically impossible, it is sufficient to state the relationship between two entities in this way for design purposes.

b. Matching. The SDM matching facility allows a member of one entity class to be matched with a member of another entity class. That means the value of an attribute in one of the members is moved to the other.

c. Derivations. The last SDM facility for the inter-relation constraints is the derivation. Derivation can be used to specify relationships among members in the same entity class. This means an attribute of an entity class can be defined as the derivation of some other attributes within

this class (e.g. by summation of them). [Ref. 9:pp. 49-58]

F. NORMALIZATION

The normalization process, as first proposed by Codd, takes a relation schema through a series of tests to certify whether or not it belongs to a certain normal form. Initially, Codd proposed three normal forms, which he called first, second, and third normal form. A stronger definition of third normal form (3NF) was proposed later by Boyce and Codd and is known as Boyce-Codd normal form. All these normal forms are based on the functional dependencies among the attributes of a relation. Later, a fourth normal form (4NF) and a fifth normal form (5NF) were proposed, based on the concepts of multivalued dependencies and joint dependencies, respectively. Each of the higher normal forms contains others in the lower ones, as shown in Figure 10.

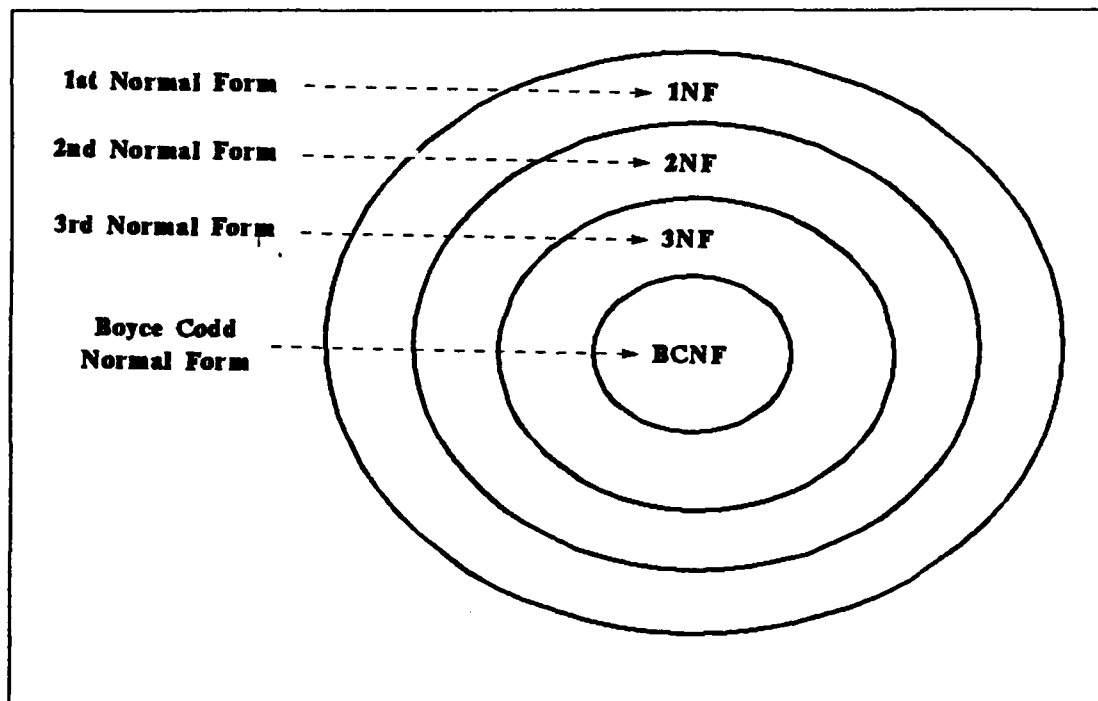


Figure 10. Relationship of Normal Forms.

Normalization of data can be looked on as a process during which unsatisfactory relation schemas are decomposed by breaking up their attributes into smaller relation schemas that possess desirable properties.

The serious problem with using the relations is the update anomalies. These can be classified into insertion anomalies, deletion anomalies, and modification anomalies. [Ref. 1:pp. 360-361]

1. Update Anomalies

a. Insertion anomalies can happen when we cannot insert a fact in one entity without inserting an additional fact in another entity.

b. Deletion anomalies can take place when deleting a fact from one entity results in the loss of a fact from another entity.

c. Modification anomalies. If we want to change a fact in one entity, we must update all the facts that are related with that entity. Otherwise, the database will become inconsistent.

2. Functional Dependency

A functional dependency is a constraint between two sets of attributes from the database. Suppose our relational database schema has n attributes A_1, A_2, \dots, A_n ; let us think of the whole database as being described by a single universal relation schema $R = \{ A_1, A_2, \dots, A_n \}$.

A functional dependency, denoted by $X \rightarrow Y$, between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuples that can form a relation instance of r of R . The constraint states that for any two tuples t_1 and t_2 in r such that $t_1[X] = t_2[X]$, we must also have $t_1[Y] = t_2[Y]$. This means that the values of the Y component of a tuple in r depend on, or are determined by, the values of the X component. We also

say that there is a functional dependency from X to Y or that Y is functionally dependent on X. In other words, attribute Y is functionally dependent on a set of attributes X if at every instance, each value of X has only one value of Y associated with it.

Attribute that functionally depends on the full composite key is said to be fully dependent on that key. Otherwise, attribute is said to be partially dependent on that key. And, a transitive dependency occurs when one non-key attribute is dependent on one or more non-key attributes. In other words, if $X \rightarrow Y$ and $Y \rightarrow Z$, then Z is functionally dependent on X. [Ref. 1:p. 365]

3. Normal Forms

Normalization theory is built around the concept of normal forms. A relation can be examined to be in one of the normal forms that the relational theorists have defined. Normally, a relation will be unnormalized, which means it may contain repeating groups whose presence creates serious access problems leading to reduction in data independence. A relation may contain nonprime attributes with partial and transitive dependency on the candidate keys. These undesirable associations are removed from a relation by normalization. The existence of normal forms is because any one of them does not eliminate all the anomalies, but only certain anomalies. For that reason, relational database designers tried to find normal form to eliminate all the anomalies. So far, the examination up to BCNF (Boyce-Codd Normal Form) is enough for a good database structure. Even though there have been introduced a couple of higher normal forms, they are not so good for application yet and so we will exclude those forms.

a. *First Normal Form (1NF)*

First normal form is defined to disallow multivalued attributes, composite attributes, and their combinations. It states that the domains of attributes must include only atomic (simple, indivisible) values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. Hence, 1NF disallows having a set of values, a tuple of values, or combination of both as an attribute value for a single tuple. In other words, 1NF disallows *relations within relations* or *relations as attributes of tuples*. The only attribute values permitted by 1NF are single atomic (or indivisible) values from a domain of such values. [Ref. 1:p. 373]

Consider the BOOKINFO relation schema shown in Figure 11. This is not in 1NF because AUTH_NAME is not an atomic attribute. It does not even qualify as a relation according to our definition. To normalize into 1NF relations, we break up its attributes into the two relations BOOK and AUTHORS shown in Figure 12. The idea is to remove the attribute AUTH_NAME that causes the relation not to be in 1NF and place it in a separate relation AUTHORS along with the primary key BOOK_NUMBER of

BOOKINFO		
<u>BOOK_NUMBER</u>	BOOK_NAME	AUTH_NAME
103511	Database system	Henry F. Korth Abraham Silberschatz
121234	Library Management	Robert D. Stueart John T. Eastlick
130056	Computer Network	Andrew S. Tanenbaum

Repeating Groups: AUTH_NAME

Figure 11. BOOKINFO relation schema that is not in 1NF.

BOOK. The primary key of this relation is the combination {BOOK_NUMBER, AUTH_NAME} as shown in Figure 12. A distinct tuple in AUTHORS exists for each author of a book. The AUTHORS attribute is removed from the BOOKINFO relation, decomposing the non-1NF relation into the two 1NF relations BOOK and AUTHORS.

BOOK		AUTHORS	
<u>BOOK_NUMBER</u>	<u>BOOK_NAME</u>	<u>BOOK_NUMBER</u>	<u>AUTH_NAME</u>
103511	Database system	103511	Henry F. Korth
121234	Library Management	103511	Abraham Silberschatz
130056	Computer Network	121234	Robert D. Stueart
		121234	John T. Eastlick
		130056	Andrew S. Tanenbaum

Figure 12. Decomposed relations that are in 1NF.

b. Second Normal Form (2NF)

The second normal form is based on the concept of a full functional dependency. A functional dependency $X \rightarrow Y$ is a full functional dependency if removal of any attribute from X means that the dependency does not hold any more. And a functional dependency $X \rightarrow Y$ is a partial dependency if there is some attribute that can be removed from X and the dependency will still hold.

For example, the CIRCULATION relation schema is not in 2NF since the non-key attributes BOOK_DATA, CIR_DATE, and CIR_TYPE are not fully functionally dependent on the key {CIR_NO, BOOK_NUMBER} as shown in Figure 13. After these partial dependencies are removed, new relation schemas in 2NF will be produced as shown in Figure 14.

dependency $X \rightarrow Y$ in a relation schema is a transitive dependency if there is a set of attributes Z that is not a subset of any key, and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold. [Ref. 1:pp. 376–380] R_CRPBK relation schema in Figure 14 has a transitive dependency. We say that the dependency of $PATRON_DATA$ on the key attribute $\{CIR_NO, BOOK_NUMBER\}$ is transitive via $PATRON_ID$ because both the dependencies $\{CIR_NO, BOOK_NUMBER\} \rightarrow PATRON_ID$ and $PATRON_ID \rightarrow PATRON_DATA$ hold, and $PATRON_ID$ is not a subset of the key of R_CRPBK . Intuitively, we can see that the dependency of $PATRON_DATA$ on $PATRON_ID$ is undesirable in R_CRPBK since $PATRON_ID$ is not a key of R_CRPBK . Therefore, we can normalize R_CRPBK by decomposing it into the two 3NF relation schemas shown in Figure 15.

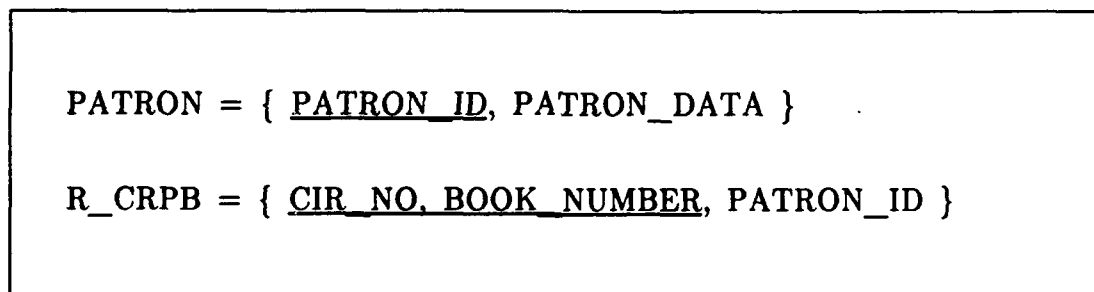


Figure 15. Decomposed relations that are in 3NF.

d. Boyce–Codd Normal Form (BCNF)

A relation schema R is in Boyce–Codd normal form if whenever a functional dependency $X \rightarrow Y$ holds in R , then X is a super key of R . The only difference between BCNF and 3NF is that condition (b) of 3NF, which allows Y to be nonprime if X is not a superkey, is absent from BCNF. Hence, BCNF is stronger (more restrictive) than 3NF, meaning that any

relation schema in BCNF is automatically in 3NF. [Ref. 1:p. 381] Another popular definition is that a relation is in BCNF if it is in 3NF and every determinant is a candidate key. A determinant is a set of attributes on which some other attributes are fully dependent. [Ref. 6:p. 144]

The relation schemas in Figure 15 are also in BCNF. They do not violate the above definitions. In general, it is best to have relation schemas in BCNF. If that is not possible, 3NF will do. However, 2NF and 1NF are not considered good relation schema designs. These normal forms were developed historically as stepping stones to 3NF and BCNF.

G. DESIGNING LLMS WITH SDM

We have stated the SDM design concepts and the normalization process. Now, we need to apply the normal forms to the SDM design. Then we can look at the relationships between the entity classes.

In relation schema design, a key functionally determines all non-key attributes. More than two keys may be possible, but only one of them can be primary key and the others are named candidate keys. Functional dependencies are used to find the key(s) of the relation. Functional dependencies which are not related to LLMS are not used to our database design.

Figure 16 demonstrates the relation schemas for LLMS. The entity BOOK contains three functional dependencies: BOOK_NUM \rightarrow AUTH_NAME, CALL_NUM \rightarrow BOOK_NAME, and ISBN_NUM \rightarrow BOOK_NAME. But, for the purpose of simplicity, We will allow only one author for each book and do not apply normal forms to the BOOK entity class. The entity PATRON has

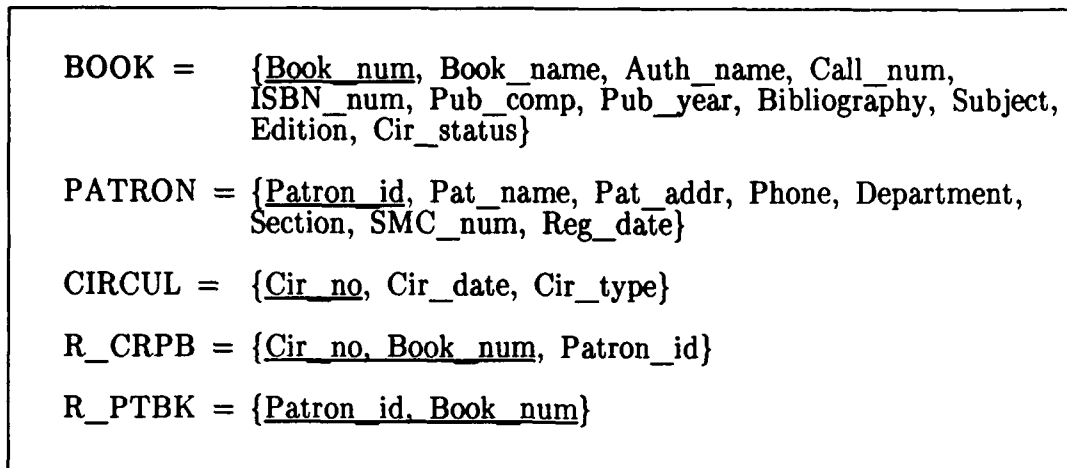


Figure 16. The relation schema design for LLMS.

two functional dependencies: PHONE \rightarrow PAT_NAME, PAT_ADDR and SMC_NUM \rightarrow PAT_NAME. These dependencies in the LLMS do not affect the whole transactions of the system, so we do not apply normal forms to the PATRON entity class.

H. E-R DIAGRAM REPRESENTATION

As with SDM, the E-R model is based on the real world objects and represents them as entities and relationships among these objects. The E-R diagram can represent the overall logical structure of a database by means of a graphical expression. The E-R diagram notations are shown in Figure 17. The normalized SDM design must be transformed into the E-R diagram. This diagram can show the entities, the attributes in them, the relationships between entities, and the cardinality ratio in relationships.

The E-R diagram for LLMS using the normalized SDM design is shown in Figure 18. This diagram is drawn for better understanding, as the conceptual schema development step in Figure 6 describes.

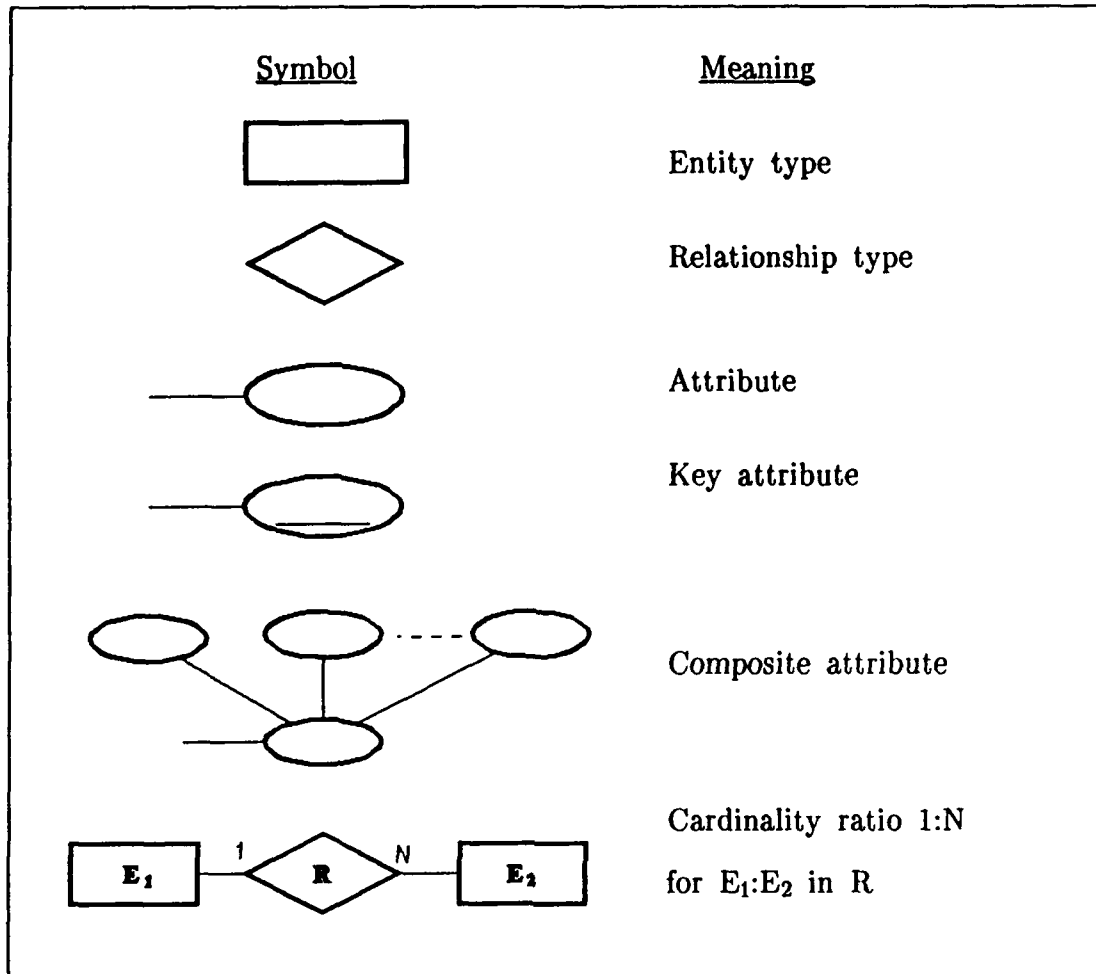


Figure 17. The E-R diagram notations. [Ref. 1:p.57]

The E-R modeling concepts can model a wide variety of database applications. However, some applications – especially newer ones such as databases for engineering design applications or for artificial intelligence applications – require require advanced modeling concepts if we want to model them with greater accuracy. [Ref. 1:pp. 409–452]

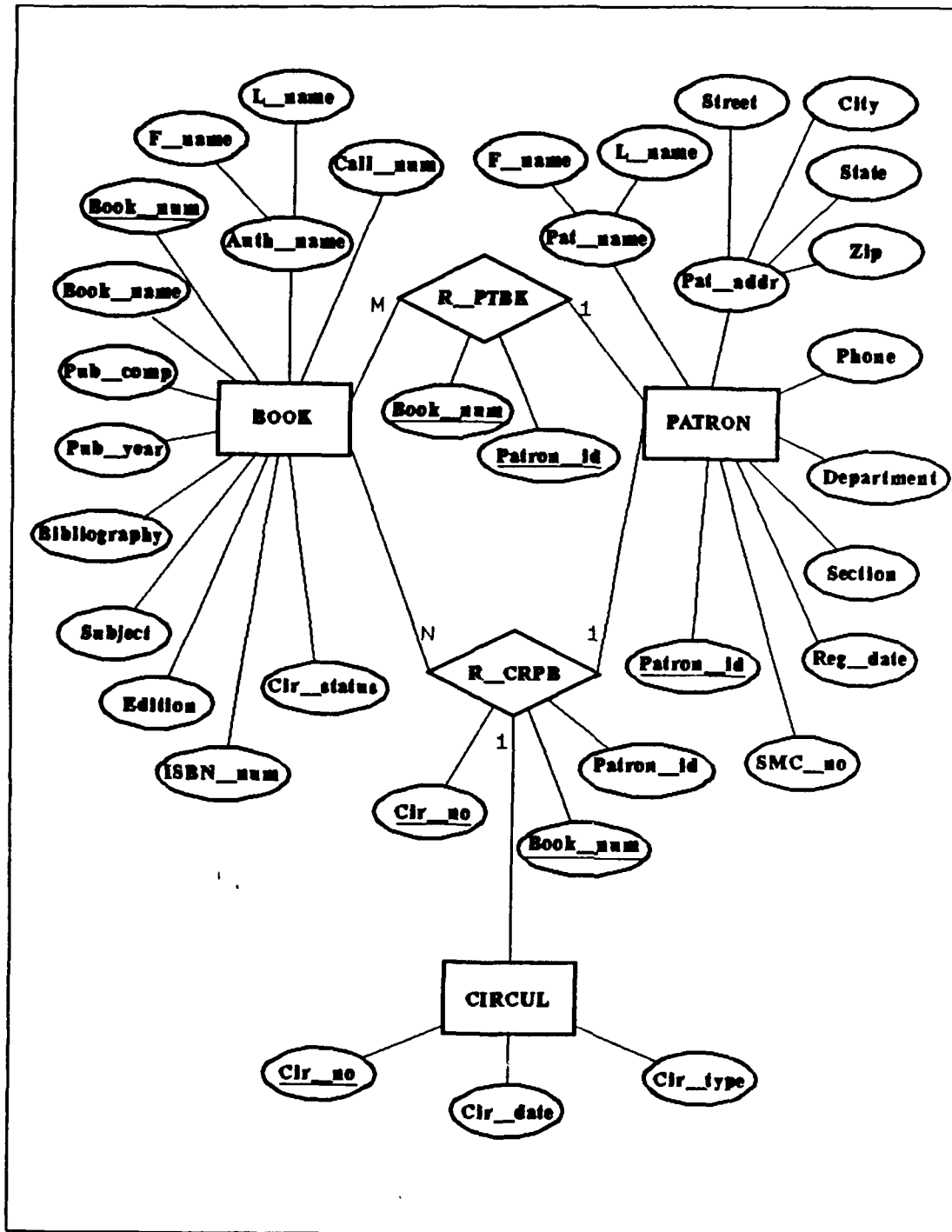


Figure 18. The E-R Diagram for LLMS.

VI. IMPLEMENTATION

After the conceptual design is completed, we can do the implementation design and the physical design. These two steps refine the conceptual design so that it can be implemented on the specific DBMS. Language statements in the DDL (Data Definition Language) and SDL (Storage Definition Language) of the selected DBMS are compiled and used to create the database schemas and database files. Then, the database can be loaded with the data. If data is available from an earlier computerized system, conversion may be required to reformat the data for loading into the database.

Now, the database transactions must be implemented. The specifications of transactions are examined, and corresponding program with embedded DML (Data Manipulation Language) is coded and tested. Once the transactions are ready and the data is loaded into the database, the design and implementation phase is set and then the operational phase begins.

The implementation design is the mapping of the conceptual design into a DBMS logical model. The physical design is the process of selecting the appropriate file structures, access methods and related factors. The major inputs to physical design are the logical structure from the implementation. Both designs must be done carefully, since they affect performance, security, and a number of other factors. [Ref. 4:p. 278]

The features of dBASE IV which will be used and its hardware requirements must be represented before the presentation of the implementation and physical design.

A. SOFTWARE REQUIREMENTS

The DOS version of dBASE IV from Ashton-Tate will be used to implement the LLMS (Library Loan Management System). dBASE IV uses a relational database model and is a database manager targeted for microcomputer use. It has a number of advanced features and generous limits. The following summary shows the features and limits of dBASE IV.

1. Features of dBASE IV

a. The Control Center. enables users to access all design screens and data, and displays the names of all files in a catalog.

b. Data Types. Characters, Numerics, and Logicals. In addition, Float, Date, and Memo field are useful for floating-point, date, and text manipulation.

c. Advanced Features. SQL programming module, QBE utility, and improved file-handling and application development capabilities.

d. Applications Generator. enables users to build applications of any complexity using pull-down menus, pop-up menus, and windows.

e. Automatic Compilation: maximizes execution speed.

2. Limits of dBASE IV

a. Each Database File

Number of records:	1 billion
Number of bytes:	2 billion
Record size(.DBF):	4,000 bytes
Number of fields:	255

b. Index File

Number of indexes per multiple index file:	47
Block size:	16,384 bytes(default: 2,048bytes)

c. Field sizes

Character fields:	254 bytes
-------------------	-----------

Date fields:	8 bytes
Logical fields:	1 byte
Type N(BCD) fields:	20 decimal digits
Type F(float) fields:	20 decimal digits
Field names:	10 characters

d. Arrays

Dimensions:	2
Total elements(columns × rows):	1,170

e. File Operations

Open files(all types):	99
Open database files:	10
Open memo files per active database:	1
Open index files per active database:	10
Open format files per active database:	1
Open procedure files per run:	1

f. Memory Variables

Default:	500
Maximum established in CONFIG.DB:	15,000

g. Word Wrap Editor

Number of lines:	32,000
Line lengths:	1,024

h. QBE

Joined files:	8
---------------	---

i. SQL

Tables in a join:	limited by available memory
Number of cursors:	10
Number of indexes per table:	47
SQL statement length:	1,024 characters

Available memory may limit the above maximum values. [Ref. 10:pp.

13-15]

B. HARDWARE REQUIREMENTS

dBASE IV is designed to run on IBM PC/XT, AT, or 100-percent compatible computers. In addition, it is designed to run on IBM PS/2 Models and COMPAQ Deskpro Models. We must have 640K of system RAM to run dBASE IV and a hard disk drive. dBASE IV works best on 286- or

386-based systems.

dBASE IV runs with PC DOS V2.0 or greater or COMPAQ DOS V3.31. Other versions of DOS may allow satisfactory operation depending on their compatibility with PC DOS.

Another requirement may be a printer with at least 80 columns and which can interface with the above computers. [Ref. 10:p. 11]

C. IMPLEMENTATION DESIGN

Implementation design starts with the conceptual data model and maps that model into a logical model that conforms to a particular database management system. The mapping from the conceptual data model to a logical model is the most important step in implementation design.

The conceptual data model may be expressed in the entity-relationship model, the semantic data model, the data structure diagram, or a set of relations in third normal form. On the other hand, most database management systems support one or more of the three logical models which were stated earlier (relational, network, and hierarchical model). Therefore, the mapping task may vary from trivial (such as when the conceptual and logical models are both relational) to complicated (such as mapping from an entity-relationship model to a hierarchical model). [Ref. 4:pp. 278-282]

The process of mapping to a logical data model depends on the form of the conceptual data model and the form of the logical model. In this thesis, mapping E-R diagram from the conceptual design onto a relational model is a straightforward process. Each entity type in the E-R diagram becomes a relation, and attributes of the entity are attributes of the relations. Each

relationship also becomes a relation.

D. PHYSICAL DESIGN

Physical design is the process of developing an efficient, implementable physical database structure. [Ref. 4:p. 299]

The full power of dBASE IV derives from its combination of a relational database, interactive query environment, and a powerful application development language. With dBASE IV, we can develop user-friendly applications that allow someone with limited computer experience to enter data and generate reports in a routine manner.

Each relation schema can be implemented as a separate database file(.DBF) and data in the file can be retrieved, updated, and deleted either by directly accessing the file from the user or by the application programs. Application programs(.PRG) may be coded manually or automatically (by using applications generator). Input forms(.FMT) for entering data can be produced easily by using database file and accessed by the application programs.

The LLMS is designed and implemented for user-friendly applications in which the user can easily understand the whole system actions. The application programs are listed in Appendix C. The following functional description will help users access the LLMS and enable them to enhance its capability and performance.

E. FUNCTIONAL DESCRIPTION

In order to run the LLMS (Library Loan Management System), you have

to put the following sequence of keystrokes at the DOS prompt:

```
DBASE↵
↵ (When you see the copyright screen)

* Now, you will see dBASE IV control center.
  Select "LLMS_SYS.CAT" catalog using menu bar.
  Highlight "LLMSMAIN.PRG" on the Applications Panel.
  ↵
  Highlight "Run the application" and ↵
  Highlight "Yes" and ↵

* If you are at the dot prompt,
  Type "DO LLMSMAIN" and ↵
```

Figure 19. Starting LLMS on dBASE IV

1. The LLMS Main Menu

Now, you will be at the LLMS main menu as shown in Figure 20. The LLMS is menu-driven. To perform it, all you have to do is make choices from the menus given on the screen. The main menu has four options on it. These options tell the LLMS which functions you want to work with.

```
LLMS Main Menu - [F10] for HELP,- [ESC] to exit
(1) Library Catalog
(2) Patron Registration
(3) Circulation
(4) Reference Service
```

Figure 20. The Library Loan Management System's Main Menu.

a. Library Catalog. This option allows you to enter data about new

books and to print the list of books.

b. Patron Registration. This option allows you to enter data about new patrons, change patron data, and print a list of patrons.

c. Circulation. This option is used to record the check-out of books, record the check-in of loaned books, find out whether a book available for loan, print a circulation summary, and print a list of overdue books.

d. Reference Service. This option enables you to search a book by using known information.

Selection from the menus is made by locating highlight at desired position using arrow keys (↓ and ↑) and by hitting [ENTER]. Then, you can get into the desired function.

2. Getting Started

The first thing you can get is a book list. To obtain it, choose option (1) library catalog from the LLMS main menu. The system presents you with the library catalog menu as shown in Figure 21. Select either option (2) print by item number or option (3) print by title. Then, the system will show you the information destination menu (Figure 22). You need to answer where you want the report information to go. The way to choose an option

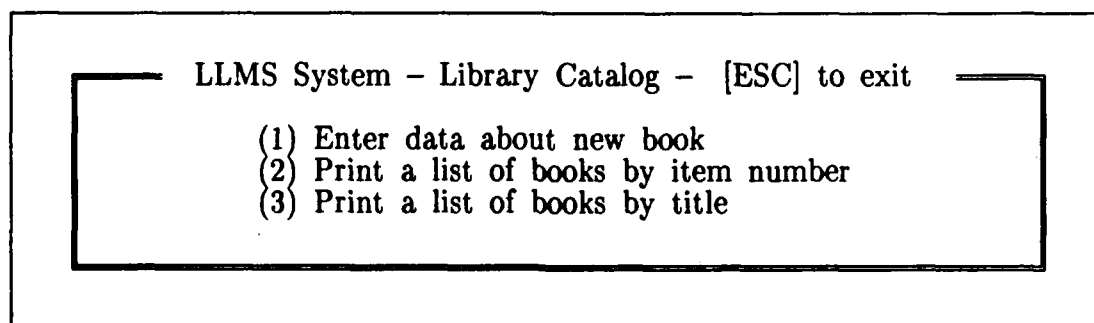


Figure 21. The Library Catalog Menu.

<div style="border: 1px solid black; padding: 2px; display: inline-block;"> PRINTER </div>	Select the report destination SCREEN FILE	
---	--	--

Figure 22. The Information Destination Menu.

from this kind of menu is to move the highlighted area with arrow keys until your choice is highlighted, and then press [ENTER]. If the report destination is printer, make sure that the power is on and that the printer has been loaded with continuous form paper. When the system finishes printing the book list like Figure 23, it will be back to the library catalog menu. If you want to go back to the LLMS main menu, press [ESC].

Library Loan Management System Book Item Inventory As of 04:11:56 on February 20, 1990			
Item_no	Title Author	Publisher	Page 1 Call_no ISBN
1	Telecommunications and the computer James Martin	Prentice-Hall, Inc.	TK5101.M326 0-13-902494-8
2	Complete reference for dBASE IV Douglas Hergert	Microsoft press	QA76.9.D3H484 1-55615-165-9
3	Library automation issues Dennis Reynolds	R.R.Bowker company	Z678.9.R.38 0-8352-1489-3
4	Computer networks Andrew Tanenbaum	Prentice-Hall, Inc.	TK5105.5.T36 0-13-162959-X
Total Books listed :		4	
End of Book list.			

Figure 23. An example of Book List.

Also, you can get a patron list in a similar way. Choose option (2) patron registration from the LLMS main menu, then you will be on the patron registration menu as shown in Figure 24. Now, you can print a patrons list by choosing option (3). After the report is finished printing (Figure 25), you will be asked to *Press any key to continue*. Then you will be returned to the patron registration menu. You can use [ESC] key to go back to the LLMS main menu.

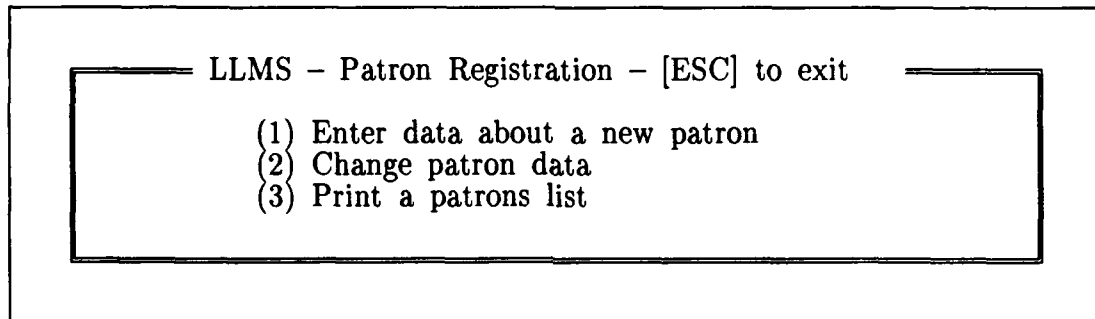


Figure 24. The Patron Registration Menu.

3. Entering Data about New Book

Now try putting data into the database. Record data for a new book by choosing option (1) library catalog from the LLMS main menu. Next select option (1) enter data about new book. To record new book information, fill in the blanks on the book item information form (Figure 26) by following the instructions in the screen.

4. Patron Registration

a. Entering Data for New Patron

New patron data can be recorded in a similar way. Choose option (2) patron registration from the LLMS main menu, and option (1) enter data about a new patron. At this time, you are required to fill out the

Library Loan Management System Patrons List As of February 20, 1990					Page 1
Name	Reg_date	Patron Id	Patron Address/Phone	Department Section/SMC_no	
Seong S. PARK	01/31/90	1	1143 Carson E Seaside, CA93955 (408)394-5380	Electrical Eng. Telecomm Mgnt. 1888	
George BUSH	02/01/90	2	1202 Sanpablo H Seaside, CA93955 (408)624-2709	Admin Science. Information Sys. 2345	
John WAYNE	03/15/90	3	64 Sloat Av. Carmel, CA93940 (412)646-2408	Operations Res. Operations Res. 1110	

Library Loan Management System
Patrons List
As of February 20, 1990

Total patrons listed : 3
Press any key to continue...

Figure 25. An example of Patrons List.

patron data form (Figure 27). After you have completed entering data for all the new patrons, leave the form blank and press [ESC]. The screen will be back to the patron registration menu.

b. Changing Patron Data

If you want to change any data about a patron except the patron identification number which cannot be changed, choose option (2) from the patron registration menu. You are required to answer the patron id whose data you want to change as shown in Figure 28. If you know the patron id, enter the number. If you do not have the patron id, you can get help by

Press [ESC] when done with this data

Library Loan Management System Book Item Information Form	
Enter the information in the spaces provided. To move from one space to another, press [Tab] or [Enter]. When you have finished with the form, press [Esc]. To record the information, press [A] to Add it to the database. If you want to change the information before you press [A], press [E] to Edit it. After you have added the information for all the books you have to record, press [Q] to Quit.	
All information is required -	
Book Item number: 5	
Book Title:	
Author: FirstName	LastName
Publisher:	Year:
Subject:	
Call Number:	ISBN:
Edition:	Page:

Figure 26. The Book Item Information Form.

Library Loan Management System Patron Data Form	
All information is required. To move from one space to the next, press [Enter] or [Up] or [Down]. When you finish filling in the form, press [Esc]. To return to the Patron Data Menu, leave the form blank and press [Esc].	
Patron_Id number: 4 (once entered, CANNOT be changed)	
First name:	Last name:
Department:	Section:
SMC number:	Registration date: 02/20/90
Street:	City:
State: Zip:	Phone:() -

Figure 27. The Patron Data Form.

```
Enter:
  [Esc] to return to the previous menu
  H for Help
  Enter Patron Id Number:
```

Figure 28. Request for Patron Id Number.

typing [H] and pressing [ENTER] as shown in Figure 29.

If you have the patron's last name, enter it. For instance, if you want to know the patron id of the patron whose last name is PARK, type "PARK" and press [ENTER]. Then you can see the patron id in Figure 30. If you don't have any information about patron, type "ALL" and press [ENTER]. Then you will get the patron id of all patrons. From Figure 30, you can save typing the patron id number by pressing [ESC] since it will be automatically sent to the screen in Figure 28.

```
ID Number Locator

Enter a keyword or Phrase: PARK.
Press [Esc] to exit HELP
```

Figure 29. The ID Number Locator for getting an unknown patron id.

```
Patron_Id      Patron Name
-----
      1      PARK, Seong S.

If PATRON is correct, press [Esc], else press [Enter]:
```

Figure 30. Response from the ID Number Locator.

The patron id number is either sent from the ID number locator or entered directly. After you press [ENTER], the patron data form with

PARK's data. Now you can move to any field on the form using [TAB] or [ENTER] key except the patron id number field. The patron id is used by the system to identify each patron. Also it will be used to identify which circulations each patron is associated with. Thus you are not allowed to change the patron id. In addition, the patron id will be produced by the system automatically. After you change all the information, then press [ESC]. The system will tell you to wait while the patron record is updated. Then it returns you to the patron data menu. You can press [ESC] to go back to the LLMS main menu.

5. Circulation

To perform the circulation of the book, choose option (3) circulation from the LLMS main menu. Then you will see the circulation menu (Figure 31). With this menu, you can record book loans and their return. Also, you can check to see if a particular book is available for check-out. And, you can get a list of all the circulations during any period you want to see. In addition, this function provides you a list of overdue books, which includes the information about the delinquent patrons.

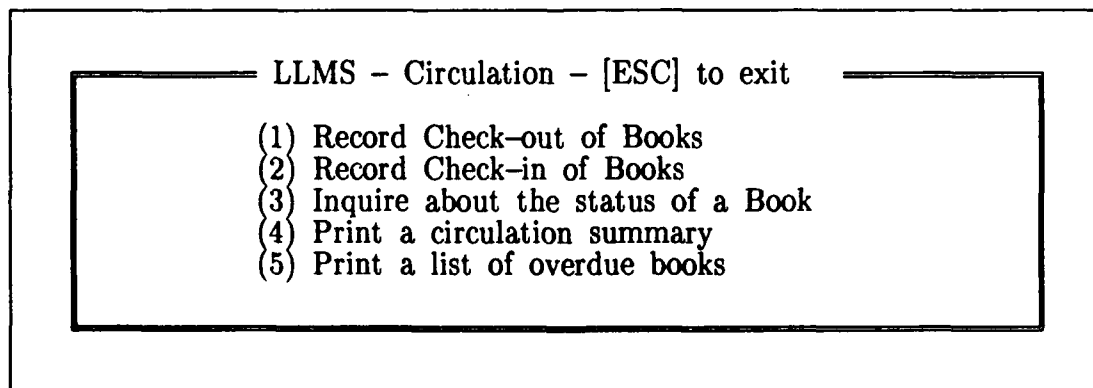


Figure 31. The Circulation Menu.

a. Recording Book Loans

Suppose Seong S. Park picks up *Complete reference for dBASE IV* and *Computer networks* for loan. He presents those books (or item number) to you. Then you can record the book loans by choosing option (1) record check-out of books. You will be asked to enter patron id number as shown Figure 28. This step works exactly like the one we saw in section 4.b. After you type Park's patron id number, the check-out form will be displayed on the screen (Figure 32). The system will provide the patron's name and address, and the circulation number and date.

To fill out the check-out form, simply type the item number of the book. Then system will present you the book title and the call number. When you type the item number for checked-out book or the invalid number (not on the file), system will display an error message in the book title field. You can borrow a maximum of four books. It is not possible to record the check-out of more than four books. When you are completed, press [ESC].

Library Loan Management System — Check-out procedure			
Patron number:	1	Check-out to:	Seong S. PARK
Circulation number:	5		1143 Carson E
Circulation date:	02/20/90		seaside,CA93955
Item Number	Book Title	Circulation type:	O
			Call_No
2	Complete reference for dBASE IV		QA76.9.D3H484
4	Computer networks		TK5105.5.T36
1 :	That book is already checked out!		
109921 :	Item_No is not found!		
Total Check-out:			2

Type the item number then [ENTER], [ESC] when done

Figure 32. The Check-out Form.

b. Inquiring about the Book Status

If a patron wants to check to see if *Complete reference for dBASE IV* is available for loan, then choose option (3) inquire about the status of a book from the circulation menu. You will be asked to enter item number. This step is also similar to the one we saw in section 4.b. After you type the item number and press [ENTER], the system tells us the information about the patron who loaned *Complete reference for dBASE IV* (Figure 33). If a book is available for loan, the system tells us that it is available.

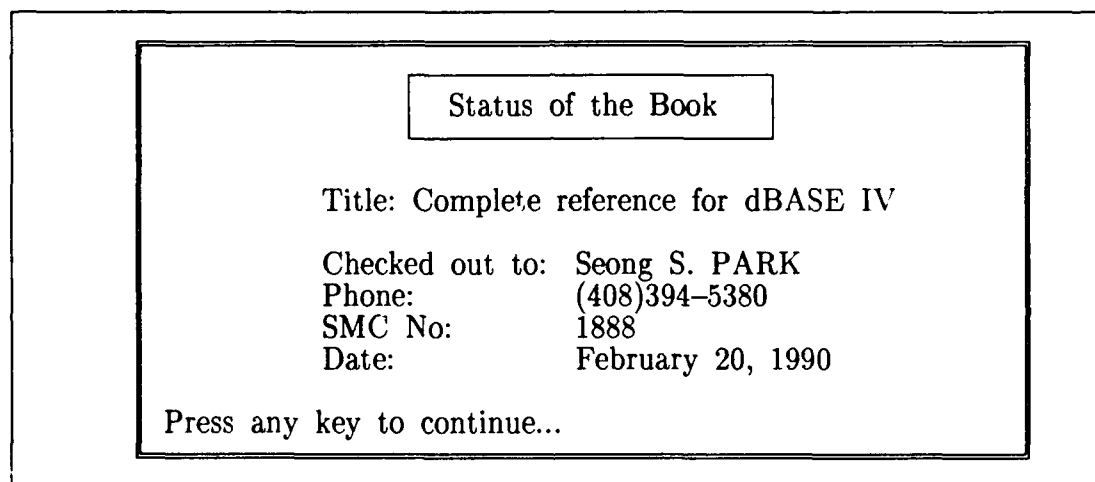


Figure 33. The Status of the Book.

c. Recording Book Returns

You can record the return of the loaned books by choosing option (2) record check-in of books from the circulation menu. Then you will be asked to enter patron id number. Again, you can get help by pressing [H] and [ENTER]. If you type Park's id number, the system will give you a list of all the unreturned books (Figure 34).

After pressing any key, you have to answer the question *Have all*

PARK, Seong S. has the following books out:		
Item_No	Title	Call_No
2	Complete reference for dBASE IV	QA76.9.D3H484
4	Computer networks	TK5105.5.T36

Press any key to continue...

Figure 34. The All Unreturned Books.

books been returned? (Y/N). If the patron brought all the books loaned, then press [Y]. If you press [N], the system will ask you to enter the item number of any book not checked in.

d. Printing a Circulation Summary

If you choose option (4) print a circulation summary from the circulation summary, then you can get the loan history during the period you specify. You will be prompted the following:

```
Please enter Beginning Report Date : / /
Please enter Ending Report Date   : / /
Format for Date input is MM/DD/YY
```

After you type the appropriate period and choose the report destination, the circulation summary is printed as shown in Figure 35. It will take more time than other reports.

e. Printing a Overdue List

You can get a list of overdue books by choosing option (5) from the circulation menu. The library does not allow its patrons to loan books for more than two weeks. After responding to the report destination menu, you will be given the list of books which are past due as shown in Figure 36.

Library Loan Management System
 Circulation Register
 From February 10, 1990 To February 20, 1990
 Printed at 04:39:22 on February 21, 1990

Page 1

Date: February 20, 1990
 Circulation Number: 4
 Patron Name: BUSH, George

Item_No	Title	Quantity
3	Library automation issues	1
Circulation total		1 book

Date: February 20, 1990
 Circulation Number: 5
 Patron Name: PARK, Seong S.

Item_No	Title	Quantity
2	Complete reference for dBASE IV	1
4	Computer Networks	1
Circulation total		2 books
Daily total		3 books
Summary Total		3 books

Figure 35. The Circulation Summary.

6. Reference Service

To get the reference service from this system, choose option (4) reference service from the LLMS main menu. Then you will see the reference service menu (Figure 37). With this menu, you can find bibliographic information for books held by the library. In addition, circulation information is also available for the users. Search strategies given by this menu are by item number, title, author, call number, ISBN, and subject. If you have only one piece of information, you can search by whatever is known. Now, let us take a look at the power of the search capabilities.

Library Loan Management System Overdue Book Report As of 01:48:57 on February 21, 1990		
Book Late Date is 02/06/90		
Patron Name: WAYNE, John		
Phone: (412)646-2408 SMC No: 1110		
Item_No	Title	Date Out
1	Telecommunications and the computer	February 2,1990
Patron Name: BUSH, George		
Phone: (408)624-2709 SMC No: 2345		
Item_No	Title	Date Out
16	Networking personal computers	February 5,1990

Figure 36. The Overdue Book List.

LLMS - Reference Service - [ESC] to exit	
(1)	Search by Item Number
(2)	Search by Title
(3)	Search by Author
(4)	Search by Call Number
(5)	Search by ISBN
(6)	Search by Subject

Figure 37. The Reference Service Menu.

All the search options are similar. So, We will just show option (2) search by title.

Assume you only know the first word of a book's title. Then, you can select option (2) to find the information. You will be asked to enter the title and you just type one word rather than entire title (Figure 38). If you

```
Enter:
[Esc] to return to previous menu
Enter Title : dBASE IV
```

Figure 38. Request for Search Entry.

know entire title, then search will be more efficient. The system will display a list of all books whose title contains words you typed (Figure 39). You can get the information by typing item number of the book you want to see (Figure 40).

<u>Book Listing</u>	
<u>Item_No</u>	<u>Title</u>
5	dBASE IV user's desktop companion
19	dBASE IV applications library
125	dBASE IV programming

Type Item_No: 5

Figure 39. The Book Listing from Search by Title.

<u>Book Information</u>	
Item_no	: 5
Author	: Alan Simpson
Title	: dBASE IV user's desktop companion
Edition	: 1st Edition
Subject	: Database systems
Publisher	: Sybex Inc.
Year	: 1989
Call_No	: QA76.9.D3.H677
ISBN	: 0-89588-523-9
STATUS	: AVAILABLE

Figure 40. The Book Information.

VII. ISSUES IN A NETWORKED ENVIRONMENT

When more than one user needs to access database over a local area network, the application must be designed to operate in multiuser mode. The dBASE IV commands to move into multiuser mode are straightforward. In multiuser mode, dBASE IV provides basic security functions to restrict access to defined area of the application and basic concurrency control functions to protect shared information. The file and record locking allows everyone to look at the same record while only one person can modify it. If someone makes changes to a record we are viewing, we see them right away.

We can implement dBASE IV applications on most area networks with dBASE IV Local Area Network (LAN) pack which allows five additional users to be added to a dBASE IV network installation. dBASE IV applications can operate on the following network configurations:

- IBM PC LAN program V1.2 or later including Token Ring(NETBIOS V1.0 or later)
- Novell SFT NetWare/286 TTS V2.10
- Ungermann-Bass Net/One PC V16.0 or 100% compatibles
- 3COM 3+ Share Software V1.3.1

If a multiuser application is to be built based on dBASE IV on one of these networks, we need to install the multiuser version of dBASE IV. The standard version does not support multiuser since its use on a network can lead to corrupted files and indexes.

The network level commands may be imbedded in applications regardless

of their environment. If the network level commands are used in a single-user version of dBASE IV, they are simply ignored. However, if the application runs under a multiuser version on a network, we will not have to modify our original code to fit it to the network. If we want to use the multiuser version on the same computer as a single-user version, we must install the programs in different directories to prevent conflicts. PATH setting should indicate which version an out-of-directory call from DOS will use.

If several users use the same computer or have access to the same database through a local area network, we need to protect the database. The following sections will discuss network commands and functions, and describe schemes for security and concurrency control (i.e., file and record locking).

A. NETWORK COMMANDS AND FUNCTIONS

There are several network commands in dBASE IV.

.CONVERT : Adds a field to a database for multiuser lock detection.

.DISPLAY USERS : Displays the names of all currently logged dBASE IV network users.

.LIST STATUS : Displays information about the current status of dBASE IV in the network.

.LOGOUT : Logs a user off of the network, allowing a new user to log on.

.PROTECT : Activates the dBASE IV file security system.

.RETRY : Returns to a calling program and executes the same line.

.SET ENCRYPTION : Determines whether protected files are encrypted when copied.

.SET EXCLUSIVE : Sets a file open attribute to either exclusive or shared mode.

.SET LOCK : Determines whether automatic record locking is activated.

.SET PRINTER : Selects a printer on the network.

.SET REFRESH : Determines the interval for checking multiuser database changes and updating user screens.

.SET RETRIES : Sets the number of times dBASE IV reexecutes a command before quitting.

.UNLOCK : Removes record and file locks. [Ref. 11:pp. 830-831]

The following network functions can be used when we develop applications that run on a network.

.ACCESS() : Returns the level of access for the last logged-in user.

.CHANGE() : Returns .T. if a value has been changed by a network user.

.FLOCK() : Locks a database file.

.LKSYS() : Returns the time, date, and user name for a locked file.

.LOCK() : Locks a database record.

.NETWORK() : Returns .T. if dBASE is currently installed on a network.

.RLOCK() : Same function as LOCK().

.USER() : Returns the log-in name of a network user. [Ref. 11:p. 808]

B. DESIGNING A PROTECTION SCHEME

When we use dBASE IV on a network, we need to protect data from unauthorized retrieval and alteration. This can be done by designing user profiles and a file privilege scheme for each protected database.

1. Designing User Profiles

A user profile consists of login name, password, group name, and an access level to each dBASE IV user on the network.

a. *Login Name*

A login name is the name the user enters when first logging on to dBASE IV. This is used for dBASE IV to identify each user. Up to eight characters are allowed, and uppercase and lowercase letters are equivalent.

b. *Password*

User password should be unique and confidential to the user. The user can select his own password, up to 16 characters long. The password will not be appear on the screen to prevent bystanders from seeing the user password.

c. *Group Name*

Group names enable one to organize users by application. Once a group name has been assigned to a database file, only users that are assigned to that group can access the database file. A group name can be up to 8 characters long.

d. *User Access Level*

Each user should also be assigned an access level. This level can be any value in range 1 (most privileges) to 8 (least privileges). This number corresponds to the file access privileges in the file privilege schemes. [Ref. 11:pp. 723-724]

2. *Designing File Privilege Schemes*

We need to design a privilege scheme for each protected database in the system. The purpose of designing a privilege scheme is basically to control who has access to what in the system. If we do not create a protection scheme for a database file, all users on the system will have

unrestricted access to that file.

a. Database File Group

Each protected database file needs to be assigned to a group. This group name should correspond to a group name in the user profiles. Only users who are assigned to the group will have access to the database file.

b. File Access Privilege

We can assign access privileges to four types of database operation; Read, Update, Extend, and Delete. To each of these access operations, we assign a privilege level, numbered 1 through 8, that corresponds to the access privilege levels assigned to users in their user profiles. For each operation, users whose access level is equal to or lower than the access number for that operation will be granted the privilege of performing the operation. That is the level 8 is the most restricted, and the level 1 is the least restricted.

c. Field Privilege

This privilege can be used to further refine the protection scheme by adding further restrictions to individual fields. We can assign any one of three access privileges to each field; FULL (read and write), R/O (read only), and NONE (neither read nor write). This privilege works in a similar way to the file privilege. However, field privileges take precedence over file privileges only when a file privilege is less restrictive. [Ref. 11:pp. 724-727]

C. FILE AND RECORD LOCKING

In a networked environment, dBASE IV uses file locking and record locking to maintain the integrity of data in the database. Files and records are locked on an as-needed basis only to ensure that all users have maximum

access to all databases. When a particular operation requires a file or record to be locked for one user, other user can view data in that file or record, but not change it. As soon as the operation that requires the lock is completed, dBASE IV automatically releases the lock.

1. File Locking

This operation guarantees the user to have exclusive use of the file, and other users only to view data. Exclusive use of a file is granted during execution of queries that perform calculations or updating. As soon as the operation is done, dBASE IV automatically unlocks the file.

2. Record Locking

Whenever a user is modifying an individual record in a database, that record is locked. Other users can still view the record, but they cannot make changes to it. As soon as the user moves on to another record, the lock is automatically released. [Ref. 11:pp. 753-754]

VIII. CONCLUSIONS

This thesis has focused on the development of Library Loan Management System (LLMS) based on a relational database system. Since the circulation and reference services are not easy work without automation, this system will be of great help for better quality of library service. The library is a reservoir of knowledge. The circulation frequency of the library reflects the prosperity of the society. However, active use of libraries in the Korean military is not common due to the low volume of collections and the lack of library staff. This system can alleviate such problems and lead to higher levels of usage of library services.

LLMS is a low-volume real-time transaction processing system intended for small or medium size libraries. It is designed to provide such library functions as library cataloging, patron registration, circulation, and reference services based on a relational database management system. We implemented prototype LLMS to run on IBM PC/AT or XT compatible microcomputer using dBASE IV.

In this thesis, we take a look at the basic concepts of database systems with emphasis on relational model. Then we address the requirements of library loan management systems based on which the analysis and design of LLMS are performed. Finally, a prototype LLMS is implemented and its functional description is documented.

This system can be used on any relational database system other than dBASE IV with trivial modification. However, if it is to be used on database

systems based on network model or hierarchical model, much implementation work may be required.

We also discuss some issues in implementing LLMS in a networked environment. A multiuser version of LLMS can be implemented in a future research.

APPENDIX A. SDM DESIGN

BOOKINFO

description: information of each book in the library.

member attributes:

BOOK_NUMBER

description: book ID number
value class: BOOK_NUM
mandatory
not changeable

BOOK_NAME

description: book's title
value class: TITLE
mandatory
not changeable

AUTH_NAME

description: author's name
subattributes:

FST_NAME

description: first name
value class: F_NAME

LST_NAME

description: last name
value class: L_NAME

mandatory
not changeable
multivalued

CALL_NUM

description: Library of Congress card number
value class: CALL_NO

ISBN_NUM

description: International Standard Book Number
value class: ISBN

PUB_COMP

description: publisher of the book
value class: PUBLISHER
mandatory

PUB_YEAR

description: year a book was published
value class: P_YEAR
mandatory

BIBLIOGRAPHY

description: total pages of the book
value class: BIB_PAGE

SUBJECT

description: subject headings of the book
value class: SUBJECTS

identifiers:

BOOK_NUMBER

PATRONINFO

description: information about each patron of the library.

member attributes:

PATRON_ID

description: patron ID number
value class: PAT_NUM
mandatory
not changeable

PATRON_NAME

description: patron name
subattributes:

PAT_FNAME

description: first name
value class: F_NAME

PAT_LNAME

description: last name
value class: L_NAME

mandatory
not changeable

PAT_ADDR

description: patron's address
subattributes:

STREET

value class: STREET

CITY

value class: CITY

STATE

value class: STATE

ZIP

value class: ZIP

PHONE

description: patron's phone number
value class: PHONES

DEPARTMENT

description: department in which patron is working
value class: DEPT

SECTION

description: section name to which patron belongs
value class: SECT

SMC_NUM

description: patron's mail box number
value class: SMC_NO

REG_DATE

description: date that a patron is registered
value class: DATES

identifiers:

PATRON_NUMBER

CIRCULATION

description: information about the circulation of the book

member attributes:

CIRCUL_NUM

description: serial number of the circulation
value class: CIR_NO
mandatory
not changeable

CIRCUL_DATE

description: date that the circulation happens
value class: DATES
mandatory
not changeable

CIRCUL_TYPE

description: type of the circulation
value class: CIR_TYPE
mandatory

PATRONINFO

description: information about the patron

value class: PATRON_INFO
mandatory

BOOKINFO

description: information about the book
value class: BOOK_INFO
mandatory

identifiers:

CIRCUL_NUM

APPENDIX B. SDM DOMAIN DEFINITION

BOOK_NUM

interclass connection: subclass of STRINGS where length is 6 characters.

TITLE

interclass connection: subclass of STRINGS where length is less than 50 characters where specified.

F_NAME

interclass connection: subclass of STRINGS where length is less than 15 characters where specified.

L_NAME

interclass connection: subclass of STRINGS where length is less than 15 characters where specified.

CALL_NO

interclass connection: subclass of STRINGS where length is less than 18 characters where specified.

ISBN

interclass connection: subclass of STRINGS where length is 13 characters.

PUBLISHER

interclass connection: subclass of STRINGS where length is less than 25 characters where specified.

P_YEAR

interclass connection:	subclass of STRINGS where format is number as YYYY.
BIB_PAGE	
interclass connection:	subclass of STRINGS where format is positive integer less than 9999.
SUBJECTS	
interclass connection:	subclass of STRINGS where length is less than 20 characters where specified.
PAT_NUM	
interclass connection:	subclass of STRINGS where length is 5 characters.
STREET	
interclass connection:	subclass of STRINGS where length is less than 30 characters where specified.
CITY	
interclass connection:	subclass of STRINGS where length is less than 15 characters where specified.
STATE	
interclass connection:	subclass of STRINGS where length is 2 characters.
ZIP	
interclass connection:	subclass of STRINGS where length is 5 characters.
PHONES	

interclass connection:	subclass of STRINGS where format is 13 characters as (ARA)LEC-CODE.
DEPT	
interclass connection:	subclass of STRINGS where length is less than 25 characters where specified.
SECT	
interclass connection:	subclass of STRINGS where length is less than 25 characters where specified.
SMC_NO	
interclass connection:	subclass of STRINGS where length is 4 characters.
DATES	
interclass connection:	subclass of STRINGS where format is number as MMDDYYYY.
CIR_NO	
interclass connection:	subclass of STRINGS where length is 10 characters.
CIR_TYPE	
interclass connection:	subclass of STRINGS where value is: 'O', 'I'.

APPENDIX C. PROGRAM LIST

```
*****
* Program Id       : LLMS0000                               *
* Program         : LLMSMAIN.PRG                           *
* Author          : Park, Seong Seung                       *
* Date            : 15 Oct 1989                             *
* Software        : dBASE IV                                *
* Description     : This program displays the main menu for the Library Loan *
*                  Management System and calls subprogram associated with *
*                  user response.                            *
*****
```

```
set talk off
set color of normal to w+/b
set color of highlight to w+/rb
set color of field to n/g
set color of box to n/bg
set escape on
set function 10 to "4"
set procedure to ProcLib1
```

```
* initialize variables
```

```
MyNum = 0
```

```
Option = 4
```

```
Last = 27
```

```
* display main menu until Last
```

```
do while MyNum # Last
```

```
Content1 = " Library Catalog "
```

```
Content2 = " Patron Registration "
```

```
Content3 = " Circulation "
```

```
Content4 = " Reference Service "
```

```
Opn1 = ""
```

```
Opn2 = ""
```

```
Opn3 = ""
```

```
Opn4 = ""
```

```
MenuTitle = " LLMS Main Menu - [F10] for HELP,- [ESC] to exit "
```

```
clear
```

```
* execute procedure HILIGHT to display menu
```

```
do HILIGHT with Option, MenuTitle
```

```
do case
```

```
case MyNum = 1
```

```
do CATALOG.PRG
```

```
case MyNum = 2
```

```
do PATRONS.PRG
```

```

    case MyNum = 3
      do CIRCULA.PRG
    case MyNum = 4
      do REFRNCE.PRG
    case MyNum = -9
      do LLMSHELP.PRG
  endcase
enddo

* close databases, procedures and exit the system
close all
MyNum = 0
set talk on
set status on
@ 12,25 say "Thank You for Using LLMS !!"
clear all
close databases
close proc
@ 21,2 say " "
wait
set color of normal to n/bg
set color of highlight to w+/rb
set color of messages to rg+/b
set color of titles to w+/n
set color of box to b+/w
return

```

```

*****
* Program Id       : LLMS0100                               *
* Program         : PROCLIB1.PRG                           *
* Author          : Park, Seong Seung                       *
* Date            : 20 Oct 1989                             *
* SoftWare       : dBASE IV                                 *
* Description     : This program is a collection of generic *
*                  procedures and functions that can be    *
*                  called by other programs to do some    *
*                  operations.                              *
*****

```

```

*—— Procedure to paint a menu with options that can be selected by hitting
* the return key or by entering a number. Up and down arrows move to
* next selection. Pass the actual number of menu options and the name
* of the menu to this procedure.
* Menu options should be stored as "opn"— null spaces and the text as
* "content" with appropriate subscripts. Insure you use different
* variable names in your program set choice as "private" as a minimal.

```

```

PROCEDURE hilight
PARAMETERS OptionNo, MenuTitle

```

```

clear
row = 6
set status off

```

```
set escape off
```

```
* display the menu title
```

```
@ 2,1 to 11,78 DOUBLE
```

```
@ 2,15 say MenuTitle
```

```
* paint the remainder of the menu options
```

```
@ 4,22
```

```
row = 6
```

```
do while row -5 <= OptionNo
```

```
  Num = str(row -5,1)
```

```
  Opn&Num = iif(val(Opn&Num) = 0, "("+str(row-5,1)+")"+Opn&Num,;  
  Opn&Num)
```

```
  @ row-1,22 say Opn&Num + Content&Num
```

```
  row = row + 1
```

```
enddo
```

```
* initialize memory variables
```

```
Opn = 1
```

```
Num = "1"
```

```
Sel = 0
```

```
set function 10 to "4"
```

```
* reverse video on option 1
```

```
@ 3,78 to 10,78 DOUBLE
```

```
@ 5,22 get Opn1
```

```
clear gets
```

```
* loop for selecting menu options
```

```
MyNum = 0
```

```
do while MyNum = 0
```

```
  Sel = 0
```

```
  * wait for a key to be pressed
```

```
  do while Sel = 0
```

```
    Sel = inkey()
```

```
  enddo
```

```
  * if arrow key is pressed
```

```
  if Sel = 24 .or. Sel = 5
```

```
    @ Opn+4,22 say Opn&Num
```

```
    Opn = iif(Sel=24,Opn+1,Opn-1)
```

```
    Opn = iif(Opn > OptionNo,1,Opn)
```

```
    Opn = iif(Opn < 1,OptionNo,Opn)
```

```
    Num = str(Opn,1)
```

```
    @ Opn+4,22 get Opn&Num
```

```
    clear gets
```

```
    loop
```

```
  endif
```

```

* if a number is pressed
if Sel >= 49 .and. Sel < 49 + Opn
    MyNum = Sel - 48
endif

* if return key is pressed
if Sel = 13
    MyNum = Opn
endif

* if F10 is pressed
if Sel = -9
    MyNum = -9
endif

* if [ESC] is pressed
if Sel = 27
    MyNum = 27
endif
enddo

* exit and return to main menu
RETURN

```

*—— Procedure to center any character string using any right margin
PROCEDURE Center

```

PARAMETERS TitleQ, RMargin
Padding = SPACE((RMargin/2) - LEN(TRIM(TitleQ))/2)
? Padding + TRIM(TitleQ)
RETURN

```

*—— Function to convert MM/DD/YY dates to proper format
FUNCTION PropDate
PARAMETERS ThisDate
SET TALK OFF
Month = CMONTH(ThisDate)
Day = LTRIM(STR(DAY(ThisDate),2,0))
Year = STR(YEAR(ThisDate),4,0)
ThisDate = Month + " " + Day + " , " + Year
RETURN ThisDate

*—— Procedure to respond to the escape key pressed within a program
* It is called from any program that allows [ESC] to be entered as
* an option. It is primarily for those programs that generate reports.
* This program will exit to main menu program.
PROCEDURE Goodbye

```
aborted = "Print Job is Aborted"
exiting = "Returning to Main Menu"
done = val(right(time(),2)) + 5
```

```
* quit printing and exit to main program
clear
@ 14,24 get aborted
@ 15,24 get exiting
```

```
* hold message for five seconds
do while val(right(time(),2)) # done
enddo
```

```
set print off
set alternate off
close all
```

```
* exit and return to main menu
RETURN TO MASTER
```

```
*****
* Program Id      : LLMS0200                                     *
* Program        : LLMSHELP.PRG                               *
* Author         : Park, Seong Seung                          *
* Date           : 16 Oct 1989                                 *
* Software       : dBASE IV                                    *
* Description    : This program defines the help menu called from *
*                 LMSMAIN.PRG (main menu).                    *
*****
```

```
set scoreboard off
clear
```

```
* declare private variables
Private TITLE1
Private TITLE2
```

```
* initialize variables
store "Library Loan Management System" to TITLE1
store "    Main Help Menu    " to TITLE2
```

```
* help screen header block
@ 0,0 to 24,79 DOUBLE
@ 2,20 say TITLE1
@ 3,20 say TITLE2
@ 5,1 to 5,78 DOUBLE
```

```
* help screen narrative
@ 7,5 say;
    'This system is menu driven. To enable the system, all you have to do'
```

- Ⓒ 8,5 say;
'is make choices from the menus which the system displays on the screen.'
- Ⓒ 9,5 say;
'The Main Menu has four options displayed. These options notify the '
- Ⓒ 10,5 say 'system which function you want to work with.'

* description of the options on Main Menu

- Ⓒ 13,5 say;
'1. Library Catalog : This option allows you to enter data about'
- Ⓒ 14,8 say 'new books and to print the list of books.'
- Ⓒ 15,8 say 'The choices are :'
- Ⓒ 16,11 say '- Enter data about book.'
- Ⓒ 17,11 say '- Print a list of books by item number.'
- Ⓒ 18,11 say '- Print a list of books by title.'
- Ⓒ 23,5 say 'Press any key to continue...'

wait ""

clear

* define new screen

- Ⓒ 0,0 to 24,79 DOUBLE
- Ⓒ 2,20 say TITLE1
- Ⓒ 3,20 say TITLE2
- Ⓒ 5,1 to 5,78 DOUBLE
- Ⓒ 6,5 say;
'2. Patron Registration : This option allows you to enter data about'
- Ⓒ 7,8 say;
'new patrons, change patron data, and print a list of patrons.'
- Ⓒ 8,8 say 'The choices are :'
- Ⓒ 9,11 say '- Enter data about a new patron.'
- Ⓒ 10,11 say '- Change patron data.'
- Ⓒ 11,11 say '- Print a patrons list.'

- Ⓒ 13,5 say;
'3. Circulation : Select this option to record the check-out of'
- Ⓒ 14,8 say 'books, record the check-in of loaned books, find out'
- Ⓒ 15,8 say 'whether a book available for loan, print a circulation'
- Ⓒ 16,8 say 'summary, and print a list of overdue books. The choices are:'
- Ⓒ 17,11 say '- Record Check-out of Books.'
- Ⓒ 18,11 say '- Record Check-in of Books.'
- Ⓒ 19,11 say '- Inquire about the status of a Book.'
- Ⓒ 20,11 say '- Print a circulation summary.'
- Ⓒ 21,11 say '- Print a list of overdue books.'
- Ⓒ 23,5 say 'Press any key to continue...'

wait ""

clear

* define new screen

- Ⓒ 0,0 to 24,79 DOUBLE
- Ⓒ 2,20 say TITLE1
- Ⓒ 3,20 say TITLE2
- Ⓒ 5,1 to 5,78 DOUBLE

```

© 7,5 say '4. Reference Service : Select this option to search a book'
© 8,8 say;
    'by item number, title, author, call number, ISBN, and subject.'
© 9,8 say 'The choices are : '
© 10,11 say '- Search by Item Number'
© 11,11 say '- Search by Title'
© 12,11 say '- Search by Author'
© 13,11 say '- Search by Call Number'
© 14,11 say '- Search by ISBN'
© 15,11 say '- Search by Subject'
© 17,5 say 'Making choices from the menu is easy to do.'
© 18,5 say 'Use the keyboard arrow keys to move the highlighted bar'
© 19,5 say 'and press [ENTER] for your choice.'
© 23,5 say 'Press any key to continue...'
wait ""
return

```

```

*****
* Program Id      : LLMS1000                                *
* Program        : CATALOG.PRG                            *
* Author         : Park, Seong Seung                       *
* Date          : 16 Oct 1989                              *
* Software       : dBASE IV                                *
* Description    : This program displays the book catalog menu and calls *
*                subprogram for entering new book data or listing book *
*                data.                                     *
*****

```

```

* initialize variables

```

```

BookOpt = 3

```

```

Last = 27

```

```

Private MyNum

```

```

MyNum = 0

```

```

* displays catalog menu until Last

```

```

do while MyNum # Last

```

```

    Content1 = " Enter data about new book"

```

```

    Content2 = " Print a list of books by item number "

```

```

    Content3 = " Print a list of books by title "

```

```

    Opn1 = ""

```

```

    Opn2 = ""

```

```

    Opn3 = ""

```

```

    BookTitle = " LLMS System - Library Catalog - [ESC] to exit "

```

```

    clear

```

```

* execute procedure HILIGHT to display

```

```

do HILIGHT with BookOpt, BookTitle

```

```

do case

```

```

    case MyNum = 1

```

```

        select A

```

```

        use BOOK order ITEM_NO

```



```

do BOOKDATA.PRG
close databases
case MyNum = 2
select A
use BOOK order ITEM_NO
do BOOKLIST.PRG
close databases
case MyNum = 3
select B
use BOOK order TITLE
do BOOKLIST.PRG
close databases
case MyNum = Last
exit
endcase
enddo

```

```

* exit the library catalog
return

```

```

*****
* Program Id       : LLMS1100                               *
* Program         : BOOKDATA.PRG                           *
* Author          : Park, Seong Seung                       *
* Date            : 20 Oct 1989                             *
* Software        : dBASE IV                               *
* Description     : This program accepts new book data and records data *
*                  into databases (BOOK)                   *
*****

```

```

goto top
no_rec = RECCOUNT()
end_edit = .F.
done = .F.

```

```

* execute book input until [ESC] is pressed
do while .not. done
BOOK_TITLE      = space(50)
BOOK_AT_L      = space(15)
BOOK_AT_F      = space(15)
BOOK_PUBLISHER = space(25)
BOOK_BIB_PAGE  = space(4)
BOOK_SUBJECT   = space(20)
BOOK_EDITION   = space(4)
BOOK_YEAR      = space(4)
BOOK_CALL_NO   = space(18)
BOOK_ISBN      = space(13)
BOOK_STATUS    = "I"
goto top

```

```

* this step generates item number automatically
* that is, record count plus one would be the new book id number
new_rec = RECCOUNT() + 1
BOOK_ITEM_NO = STR(new_rec, 6)

* Enter data until [ESC] is pressed
esckey = 0
do while esckey # 12
  clear gets
  set format to BOOKDATA.FMT
  read
  esckey = readkey()
enddo

* Decide what to do with the data entered
functor = 0
do while functor # 65 .and. functor # 69 .and. functor # 81 .and.;
  functor # 97 .and. functor # 101 .and. functor # 113
  functor = inkey()
enddo
end_edit = .F.
do while .not. end_edit
  do case

    * If EDIT, stay in the screen to correct until [ESC]
    case functor = 69 .or. functor = 101
      esckey = 0
      do while esckey # 12
        clear gets
        set format to BOOKDATA.FMT
        read
        esckey = readkey()
      enddo
      functor = 0
      do while functor # 65 .and. functor # 69 .and. functor # 81 .and.;
        functor # 97 .and. functor # 101 .and. functor # 113
        functor = inkey()
      enddo

    * If ADD, add data to the database and clear input format
    case functor = 65 .or. functor = 97
      append blank
      replace TITLE                with BOOK_TITLE
      replace AUTHOR_L             with BOOK_AT_L
      replace AUTHOR_F             with BOOK_AT_F
      replace PUBLISHER            with BOOK_PUBLISHER
      replace BIB_PAGE             with BOOK_BIB_PAGE
      replace SUBJECT              with BOOK_SUBJECT
      replace EDITION              with BOOK_EDITION
      replace P_YEAR               with BOOK_YEAR
      replace CALL_NO              with BOOK_CALL_NO
  endcase
enddo

```

```
replace ISBN          with BOOK_ISBN
replace STATUS        with BOOK_STATUS
replace ITEM_NO       with BOOK_ITEM_NO
end_edit = .T.
```

```
* If QUIT, finish the book data input
case functor = 81 .or. functor = 113
  done = .T.
  end_edit = .T.
```

```
endcase
enddo
enddo
```

```
* return to book catalog menu
return
```

```
*****
* Program Id      : LLMS1200
* Program        : BOOKLIST.PRG
* Author         : Park, Seong Seung
* Date          : 21 Oct 1989
* Software       : dBASE IV
* Description    : This program controls the output of a report to the
*                : printer, screen, or a file.
*****
```

```
clear
set escape off
```

```
* declare private variables
Private START_PSN
Private NO_SEL
Private OPTION
Private SELECTION
Private COLUMN
Private TEXT
```

```
* frame screen and print title
@ 2,0 to 4,79 DOUBLE
@ 2,23 say " Select the report destination"
```

```
* intialize variables
store "PRINTER SCREEN FILE" to TEXT
NO_SEL = .T.
OPTION = 1
START_PSN = "011121"
```

```
* wait until selection is made
do while NO_SEL
  @ 3,3 say TEXT
```

```

do case
  case OPTION = 1
    SELECTION = "PRINTER"
  case OPTION = 2
    SELECTION = "SCREEN"
  case OPTION = 3
    SELECTION = "FILE"
endcase
COLUMN = val(substr(START_PSN, OPTION * 2-1,2)) + 2

* print selection in reverse video
set color to bg+/gr+
@ 3,COLUMN SAY SELECTION

* get selection
I = 0
do while I = 0
  I = inkey()
enddo
do case
  case I = 4
    if OPTION = 3
      OPTION = 1
    else
      OPTION = OPTION + 1
    endif
  case I = 19
    if OPTION = 1
      OPTION = 3
    else
      OPTION = OPTION - 1
    endif

* execute selection made with [ENTER]
case I = 13
  NO_SEL = .F.
  do case
    case OPTION = 1
      clear
      do BOOKLISA.PRG
    case OPTION = 2
      clear
      do BOOKLISB.PRG
    case OPTION = 3
      clear
      do BOOKLISC.PRG
  endcase

* Execute selection made with letter key
case upper(CHR(I)) $ "PSF"

```

```

NO_SEL = .F.
do case
  case upper(CHR(I)) = "P"
    clear
    do BOOKLISA.PRG
  case upper(CHR(I)) = "S"
    clear
    do BOOKLISB.PRG
  case upper(CHR(I)) = "F"
    clear
    do BOOKLISC.PRG
  endcase
endcase

* return color to normal
set color of normal to w+/b
set color of highlight to w+/rb
set color of fields to n/g
set color of box to n/bg
enddo

* redirect output back to screen
set device to screen
on escape return
return

*****
* Program Id       : LLMS1210                *
* Program         : BOOKLISA.PRG            *
* Author          : Park, Seong Seung        *
* Date            : 25 Oct 1989              *
* Software        : dBASE IV                 *
* Description     : This program prints out  *
                  : the book list to the    *
                  : printer.                 *
*****

set print on
set console off
go top
set margin to 2

* declare private variables
Private TITLE1
Private TITLE2
Private TITLE3
Private TITLE4
Private TITLE5
Private LINE
Private LINE_COUNT
Private BOOK_COUNT
Private PAGE_NO

```

```

* initialize variables
LINE_COUNT = 4
PAGE_NO = 1
BOOK_COUNT = 0
TITLE1 = "Library Loan Management System"
TITLE2 = "Book Item Inventory"
TITLE3 = "As of " + TIME() + " on " + PROPDATE( DATE() )
TITLE4 = "Item_No" + " " + "Title" + SPACE(46) + "Call_No"
TITLE5 = SPACE(9) + "Author" + SPACE(21) + "Publisher" +
        SPACE(15) + "ISBN"
LINE = " _____ " +

```

```

* print screen heading
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
?
? SPACE(65) + "PAGE" + STR(PAGE_NO,3)
?
? TITLE4
? TITLE5
? LINE
?

```

```

* print the records
do while .not.EOF()
  if LINE_COUNT >= 25
    eject
    LINE_COUNT = 0
    PAGE_NO = PAGE_NO + 1
    ? SPACE(65) + "PAGE " + STR(PAGE_NO,3)
    ?
    ? TITLE4
    ? TITLE5
    ? LINE
    ?
  endif
  TFNAME = trim(AUTHOR_F)
  TLNAME = trim(AUTHOR_L)
  ? ITEM_NO + SPACE(2) + TITLE + SPACE(2) + CALL_NO
  ? SPACE(8) + TFNAME + " " + TLNAME +
    SPACE(27-LEN(TFNAME+TLNAME)) + trim(PUBLISHER) +
    SPACE(24-LEN(trim(PUBLISHER))) + ISBN
  SKIP
  LINE_COUNT = LINE_COUNT + 2
  BOOK_COUNT = BOOK_COUNT + 1
enddo
?
? LINE

```

```

?
? "          Total Books listed : " + STR(BOOK_COUNT,5)
? LINE

* return to menu
set print off
set console on
set margin to 0
return

*****
* Program Id      : LLMS1220 *
* Program        : BOOKLIB.PRG *
* Author         : Park, Seong Seung *
* Date           : 25 Oct 1989 *
* Software       : dBASE IV *
* Description    : This program prints out the book list to the screen. *
*****

set status off
set color to w+/b
clear
go top

* declare private variables
Private TITLE1
Private TITLE2
Private TITLE3
Private TITLE4
Private TITLE5
Private LINE
Private LINE_COUNT
Private BOOK_COUNT

* initialize variables
LINE_COUNT = 6
LINE_NO = 1
BOOK_COUNT = 0
TITLE1 = "Library Loan Management System"
TITLE2 = "Book Item Inventory"
TITLE3 = " As of " + TIME() + " on " + PROPDATE( DATE() )
TITLE4 = "Item No" + " " + "Title" + SPACE(46) + "Call No"
TITLE5 = SPACE(9) + "Author" + SPACE(21) + "Publisher" + SPACE(15) +
"ISBN"
LINE = " _____ " +;
" _____ " +;

* print screen heading
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80

```

```

?
? TITLE4
? TITLE5
? LINE
?

* print the records
do while .not.EOF()
  if LINE_COUNT >= 25
    ? LINE
    * Hold screen for viewing
    wait
    clear
    LINE_COUNT = 0
    ? TITLE4
    ? TITLE5
    ? LINE
  endif
  TFNAME = trim(AUTHOR_F)
  TLNAME = trim(AUTHOR_L)
  ? ITEM_NO + SPACE(2) + TITLE + SPACE(2) + CALL_NO
  ? SPACE(8) + TFNAME + " " + TLNAME +
    SPACE(27-LEN(TFNAME+TLNAME)) + trim(PUBLISHER) +
    SPACE(24-LEN(trim(PUBLISHER))) + ISBN
  SKIP
  LINE_COUNT = LINE_COUNT + 2
  BOOK_COUNT = BOOK_COUNT + 1
enddo

LINE_COUNT = LINE_COUNT + 3
?
@ LINE_COUNT,33 TO LINE_COUNT,38

? "      Total Books listed : " + STR(BOOK_COUNT,5)
LINE_COUNT = LINE_COUNT + 2
@ LINE_COUNT,33 TO LINE_COUNT,38 DOUBLE
?
? "                End of Book list."
?
? LINE
?

* hold last screen for viewing
wait
return

```

```

*****
* Program Id      : LLMS1230
* Program        : BOOKLISC.PRG
* Author         : Park, Seong seung
* Date           : 25 Oct 1989
*****

```



```

* Software      : dBASE IV
* Description   : This program sends a book list to a file(BOOKLIST.OUT). *
*****

```

```

set alternate to BOOKLIST.OUT
set alternate on
set console off
go top

```

```

* declare private variables

```

```

Private TITLE1
Private TITLE2
Private TITLE3
Private TITLE4
Private TITLE5
Private LINE
Private LINE_COUNT
Private PAGE_NO
Private BOOK_COUNT

```

```

* initialize variables

```

```

LINE_COUNT = 4
PAGE_NO = 1
BOOK_COUNT = 0
TITLE1 = "Library Loan Management System"
TITLE2 = "Book Item Inventory"
TITLE3 = "As of " + TIME() + " on " + PROPDATE(DATE())
TITLE4 = "Item_No" + " " + "Title" + SPACE(46) + "Call_No"
TITLE5 = SPACE(9) + "Author" + SPACE(21) + "Publisher" + SPACE(15) +
"ISBN"
LINE = " _____"

```

```

* print file heading

```

```

do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
?
? SPACE(65) + str(PAGE_NO,3)
?
? TITLE4
? TITLE5
? LINE
?

```

```

* print the records to the file

```

```

do while .not.EOF()
  if LINE_COUNT >= 25
    eject
    LINE_COUNT = 0

```

```

PAGE_NO = PAGE_NO + 1
? SPACE(65) + "PAGE " + STR(PAGE_NO,3)
?
? TITLE4
? TITLE5
? LINE
?
endif
TFNAME = trim(AUTHOR_F)
TLNAME = trim(AUTHOR_L)
? ITEM_NO + SPACE(2) + TITLE + SPACE(2) + CALL_NO
? SPACE(8) + TFNAME + " " + TLNAME +
  SPACE(27-LEN(TFNAME+TLNAME)) + trim(PUBLISHER) +
  SPACE(24-LEN(trim(PUBLISHER))) + ISBN
SKIP
LINE_COUNT = LINE_COUNT + 2
BOOK_COUNT = BOOK_COUNT + 1
enddo
?
? LINE
? "      Total Books listed : " + STR(BOOK_COUNT,5)
? LINE

* return to menu
set console on
set alternate off
close alternate
return

*****
* Program Id      : LLMS2000
* Program        : PATRONS.PRG
* Author         : Park, Seong Seung
* Date          : 17 Oct 1989
* Software       : dBASE IV
* Description    : This program displays the patron data menu and calls the
*                  subprogram for managing patrons.
*****

Sensor = 3
Last = 27
private MyNum
MyNum = 0

* displays patron data menu until Last
do while MyNum # Last
  Content1 = " Enter data about a new patron "
  Content2 = " Change patron data "
  Content3 = " Print a patrons list "
  Opn1 = ""
  Opn2 = ""

```

```

Opn3 = ""
PatronTitle = " LLMS - Patron Registration - [ESC] to exit "
clear
set procedure to PROCLIB1

* execute procedure HILIGHT to display menu
do HILIGHT with Sensor, PatronTitle
do case
  case MyNum = 1
    do PTRNDATA.PRG
  case MyNum = 2
    do PTRNCHNG.PRG
  case MyNum = 3
    do PTRNLIST.PRG
  case MyNum = Last
    exit
endcase
enddo

* close databases and exit the patron registration
close databases
return

*****
* Program Id      : LLMS2100 *
* Program        : PTRNDATA.PRG *
* Author         : Park, Seong Seung *
* Date           : 19 Oct 1989 *
* Software       : dBASE IV *
* Description    : This program will accepts new patron data and records data *
*                : into the database PATRON. This program *
*                : will run the format program PTRNDATA.FMT to get the *
*                : patron data input form. *
*                : And, here, a new patron_id number will be generated by *
*                : searching for the largest patron number and incrementing *
*                : by one. *
*****

select A
use PATRON order PATRON_ID

* initialize variables
done = .F.
goto top
no_rec = RECCOUNT()

* enter the new patrons
* execute until [ESC] is pressed
do while .not. done

```

* initialize memory variables

```
PATRON_LNAME = space(15)
PATRON_FNAME = space(15)
PATRON_STREET = space(30)
PATRON_DEPT = space(25)
PATRON_SECT = space(25)
PATRON_SMC = space(4)
PATRON_CITY = space(15)
PATRON_STATE = space(2)
PATRON_ZIP = space(5)
PATRON_PHONE = space(13)
```

find the largest PATRON_ID number

```
goto top
NO_REC = RECCOUNT()
ID_NEW = 00001
do while NO_REC # 0
  if val(PATRON_ID) > ID_NEW
    ID_NEW = val(PATRON_ID)
  endif
  NO_REC = NO_REC - 1
  skip
enddo
PATRON_NUM = str(ID_NEW + 1,5)
```

* set the registration date to the current date supplied by the computer
REGIST_DATE = DATE()

* enter the data. Exit only on [ESC]

```
LASTKEY = 0
do while LASTKEY # 12
  clear gets
  set format to PTRNDATA.FMT
  read
  LASTKEY = readkey()
enddo
```

* if data was entered, add it as a new PATRON

```
if PATRON_LNAME # " " .or. PATRON_FNAME # " "
  select A
  append blank
  replace A->L_NAME with upper(PATRON_LNAME)
  replace A->F_NAME with PATRON_FNAME
  replace A->DEPT with PATRON_DEPT
  replace A->SECTION with PATRON_SECT
  replace A->SMC_NO with PATRON_SMC
  replace A->STREET with PATRON_STREET
  replace A->CITY with PATRON_CITY
  replace A->STATE with PATRON_STATE
  replace A->ZIP with PATRON_ZIP
  replace A->PHONE with PATRON_PHONE
```

```

        replace A->PATRON_ID with PATRON_NUM
        replace A->REG_DATE with REGIST_DATE
    else
        * if data was not entered exit
        done = .T.
    endif
enddo
close databases

* return to the menu
return

*****
* Program Id      : LJMS2200
* Program        : PTRNCHNG.PRG
* Author         : Park, Seong Seung
* Date           : 19 Oct 1989
* Software       : dBASE IV
* Description    : This program makes the pointer in PATRON database to
*                 a specific record with PATRON_ID.
*****

clear
set proc to PTRNHELP.PRG
set escape off
select B
use PATRON order PATRON_ID

* initialize variables
patnum = space(5)
done = .F.

* get the PATRON_ID number
do while .not. done
    clear
    clear gets
    @ 15,20 say "Enter: "
    @ 16,22 say "[ESC] to return to the previous menu"
    @ 17,22 say "H for Help"
    @ 18,22 say "Enter Patron Id Number: " get patnum picture "!XXX9"
    read

    * handle the need for help
    if readkey() = 12
        close proc
        done = .T.
        return
    endif
    patnum = UPPER(patnum)
    if PATNUM = "H "
        do PTRNHELP.PRG

```

```

select B
use PATRON order PATRON_ID
else

    * find the patrons record
select B
set order to PATRON_ID
goto top
seek (transform(PATNUM,"99999"))
if found()
    DONE = .T.
else
    clear
    @ 10,25 say "Invalid PATRON_ID Number"
    @ 17,0 say ""
    wait
    clear
endif
endif
enddo

* call the program to change PATRON data
do PTRNCHDT.PRG
close databases
close proc

* exit and return
return

*****
* Program Id       : LLMS2210
* Program         : PTRNHELP.PRG
* Author          : Park, Seong Seung
* Date            : 20 Oct 1989
* Software        : dBASE IV
* Description     : This program provides the user with help for finding a
*                  patron id number. If a borrower forgets his or her patron
*                  id number, last name and a search will return the id
*                  number. Or operator can display a list of all patrons to
*                  help the patron.
*****

clear
set bell off
search = space(15)
allitems = space(15)
keyword = space(15)
dummy = .T.
title = " Patron Listing Continued"
close databases

```

```

* open databases and index files
select A
use PATRON order L_NAME
go top

* loop through the help prompt
do while dummy
  clear
  @ 5,20 say "ID Number Locator"
  @ 15,20 say "Enter a Keyword or Phrase: " get keyword
  @ 16,20 say "Press [ESC] to exit HELP"
  read
  if readkey() = 12
    clear
    close databases
    return
  endif
  search = upper(keyword)

  * execute the response from operator
  if search # space(15)
    if search # "ALL"
      seek search
      if found()
        do while upper(L_NAME) = search
          clear
          @ 8,15 say "Patron_Id      Patron Name"
          @ 9,15 say "_____+;"
          "
          ? space(15) + PATRON_ID + space(12) + trim(L_NAME) + ;
            " + trim(F_NAME)
          patnum = PATRON_ID
          @ 15,10 say "If PATRON correct, press [ESC], else press [Enter]:"
          read
          if readkey() = 12
            clear
            go top
            dummy = .F.
            exit
          endif
          skip
        enddo
      else
        @ 22,3
        wait " Keyword not found - Press any key to continue"
        search = space(15)
        keyword = space(15)
        clear
      endif
    endif
  endif
enddo

```

```

else
  clear
  set order to PATRON_ID
  go top
  row = 7
  col = 5
  @ 3,15 say "          Patron Listing          "
  @ 5,15 say " Patron_Id          Patron Name"
  @ 6,15 say " _____+;
  "
do while .not. EOF()
  ? space(15) + PATRON_ID + space(20) + trim(L_NAME) + ", " +;
  trim(F_NAME)
  if ( row >= 22 )
    ?
    wait
    clear
    ?
    ?
    ? TITLE
    row = 3
  endif
  skip
  row = row + 1
enddo
go top
?
?
?
wait
keyword = space(15)
clear
select A
go top
dummy = .F.
endif
endif
enddo

* close databases and return
close databases
return

*****
* Program Id      : LLMS2220
* Program        : PTRNCHDT.PRG
* Author         : Park, Seong Seung
* Date           : 21 Oct 1989
* Software       : dBASE IV
* Description    : This program allows to change data in PATRON file.
*
```

set escape off

* initialize variables

```
LASTKEY = 0
PATRON_LNAME = L_NAME
PATRON_FNAME = F_NAME
PATRON_DEPT = DEPT
PATRON_SECT = SECTION
PATRON_SMC = SMC_NO
PATRON_STREET = STREET
PATRON_CITY = CITY
PATRON_STATE = STATE
PATRON_ZIP = ZIP
PATRON_PHONE = PHONE
PATRON_NUM = PATRON_ID
REGIST_DATE = REG_DATE
```

* enter the changes

* loop until escape is pressed

```
do while LASTKEY # 12
  set format to PTRNDATA.FMT
  read
  LASTKEY = readkey()
enddo
```

* store the changes

```
replace B->L_NAME with PATRON_LNAME
replace B->F_NAME with PATRON_FNAME
replace B->DEPT with PATRON_DEPT
replace B->SECTION with PATRON_SECT
replace B->SMC_NO with PATRON_SMC
replace B->STREET with PATRON_STREET
replace B->CITY with PATRON_CITY
replace B->STATE with PATRON_STATE
replace B->ZIP with PATRON_ZIP
replace B->PHONE with PATRON_PHONE
replace B->PATRON_ID with PATRON_NUM
replace B->REG_DATE with REGIST_DATE
return
```

* get the patron id number

do while .not. done

clear

clear gets

© 15,20 say "Enter: "

© 16,22 say "[ESC] to return to the previous menu"

© 17,22 say "H for Help"

© 18,22 say "Enter Patron Id Number: " get patnum PICTURE "!XXXX"

```

* handle the need for help
if PATNUM = "H"
  do PTRNHELP.PRG
  close proc
  select B
else
  * find the patrons record
  goto top
  seek PATNUM

  if found()
    DONE = .T.
  else
    clear
    @ 10,25 say "Invalid Patron Id Number"
    @ 17,0 say ""
    wait
    clear
  endif
endif
enddo

```

```

* call the program to change Patron data
do PTRNCHDT.PRG
close databases

```

```

* exit and return
set procedure to PROCLIB1
return

```

```

*****
* Program Id      : LLMS2300
* Program        : PTRNLIST.PRG
* Author         : Park, Seong Seong
* Date           : 19 Oct 1989
* Software       : dBASE IV
* Description    : This program controls the output of a report to the printer,
*                  screen, or a file.
*****

```

```

clear
set escape off
use PATRON order PATRON_ID

```

```

* declare private variables
Private START_PSN
Private NO_SEL
Private OPTION
Private SELECTION
Private COLUMN
Private TEXT

```

```

* frame screen and print title
@ 2,1 to 4,78 DOUBLE
@ 2,23 SAY ' Select the report destination '

* initialize variables
store "PRINTER SCREEN FILE" to TEXT
NO_SEL = .T.
OPTION = 1
store '011121' to START_PSN

* select the output device
* loop until selection is made
do while NO_SEL

    * display selections
    @ 3,3 SAY TEXT

    * remember selection
    if OPTION = 1
        store "PRINTER" to SELECTION
    endif
    if OPTION = 2
        store "SCREEN" to SELECTION
    endif
    if OPTION = 3
        store "FILE" to SELECTION
    endif
    COLUMN = val(substr(START_PSN,OPTION * 2-1,2)) + 2

    * print selection in reverse video
    set color to bg+/gr+
    @ 3,COLUMN SAY SELECTION

    * get selection
    I = 0
    do while I = 0
        I = inkey()
    enddo
    do case

        * right arrow is pressed
        case I = 4
            if OPTION = 3
                OPTION = 1
            else
                OPTION = OPTION + 1
            endif

        * left arrow is pressed
        case I = 19

```

```

if OPTION = 1
  OPTION = 3
else
  OPTION = OPTION - 1
endif

* return key is pressed
case I = 13
  NO_SEL = .F.
  do case

    * output to the printer
    case OPTION = 1
      do PTRNLISA.PRG

    * display the list on the screen
    case OPTION = 2
      do PTRNLISB.PRG

    * save the list to a file
    case OPTION = 3
      do PTRNLISC.PRG
  endcase

* execute selection made with letter key
case upper(CHR(I)) $ "PSF"
  NO_SEL = .F.
  do case

    * a "P" for printer was entered
    case upper(CHR(I)) = "P"
      clear
      do PTRNLISA.PRG

    * a "S" for screen was entered
    case upper(CHR(i)) = "S"
      clear
      do PTRNLISB.PRG

    * a "F" for F, E was entered
    case upper(CHR(I)) = "F"
      clear
      do PTRNLISC.PRG
  endcase
endcase

* return color to normal
set color of normal to w+/b
set color of highlight to w+/rb
set color of fields to n/g
set color of box to n/bg

```

enddo

* restore initial parameters
set device to screen
on escape return

* exit and return to menu
close databases
return

```
*****  
* Program Id      : LLMS2310                               *  
* Program        : PTRNLISA.PRG                           *  
* Author         : Park, Seong Seung                       *  
* Date           : 22 Oct 1989                             *  
* Software       : dBASE IV                                *  
* Description    : This program prints out patrons list on the printer. *  
*                : The page heading and number will appear on each page. *  
*****
```

set print on
set console off
go top
clear

* declare private variables
Private TITLE1
Private TITLE2
Private TITLE3
Private TITLE4
Private TITLE5
Private TITLE6
Private PAGE_NO
Private LINE
Private LINE_COUNT

* initialize variables
LINE_COUNT = 10
PAGE_NO = 1
TITLE1 = "Library Loan Management System"
TITLE2 = "Patrons List"
TITLE3 = "As of " + prodate(date())
TITLE4 = "Total patrons listed: " + str(reccount(),4)
TITLE5 = space(32) + "Patron" + space(2) + "Patron"
TITLE6 = space(1) + "Name" + space(20) + "Date" + space(3) + "Id Num" + space(2) +
"Address / Phone" + space(4) + "Dept/Section/SMC-No"
LINE = " _____ " +
" _____ "

* print heading
@ 10,23 say "Printing the patrons list."

```

?
?
?
?
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
?
? SPACE(70) + str(PAGE_NO)
? TITLE5
? TITLE6
? LINE
?

* print the records
do while .not. EOF()

* the page is full if the line count is >= 47. When it is full eject
* a page, print the column heading on the new page and then continue
* to print the patrons list.
if LINE_COUNT >= 51
    eject
    ?
    ?
    ?
    LINE_COUNT = 6
    PAGE_NO = PAGE_NO + 1
    ? SPACE(70) + str(PAGE_NO)
    ? TITLE5
    ? TITLE6
    ? LINE
    ?
endif

* print the records on the printer
TFNAME = trim(F_NAME)
TLNAME = trim(L_NAME)
? " "+TFNAME+" "+TLNAME+SPACE(21 - LEN(TFNAME+TLNAME))+;
  DTOC(REG_DATE) + " "+PATRON_ID+ " " + trim(STREET) +;
  SPACE(20 - LEN(trim(STREET)))+trim(DEPT)
? SPACE(39)+trim(CITY)+", "+STATE+" "+ZIP +;
  SPACE(10-LEN(trim(CITY))) + trim(SECTION)
? SPACE(39) + PHONE + SPACE(7) + SMC_NO
SKIP
LINE_COUNT = LINE_COUNT + 3
enddo
? LINE

* print the report footer
if LINE_COUNT >= 47

```

```

EJECT
PAGE_NO = PAGE_NO + 1
? space(70) + str(PAGE_NO)
endif
?
?
?
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
do CENTER with TITLE4,80
eject

* restore initial conditions
set print off
set console on

* return to menu
return

*****
* Program Id       : LLMS2320                                     *
* Program         : PTRNLISB.PRG                                 *
* Author          : Park, Seong Seung                           *
* Date            : 22 Oct 1989                                  *
* Software        : dBASE IV                                     *
* Description     : This program prints out patrons list to the screen. *
*****

set color to w+/b
go top
clear

* declare private variables
Private TITLE1
Private TITLE2
Private TITLE3
Private TITLE4
Private TITLE5
Private TITLE6
Private PAGE_NO
Private LINE
Private LINE_COUNT

* initialize variables
LINE_COUNT = 10
PAGE_NO = 1
TITLE1 = "Library Loan Management System"
TITLE2 = "Patrons List"
TITLE3 = "As of " + prodate(date())

```

```

TITLE4 = "Total patrons listed: " + str(reccount(),4)
TITLE5 = space(32) + "Patron" + space(2) + "Patron"
TITLE6 = space(1)+"Name"+space(20)+"Date"+space(3)+"Id Num"+space(2)+;
        "Address / Phone"+space(4)+"Dept/Section/SMC-No"
LINE = " _____ " +;

```

```

* print screen heading
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
?
? SPACE(70) + str(PAGE_NO,7)
? TITLE5
? TITLE6
? LINE

```

```

* print the records
do while .not. EOF()

```

```

* the screen is full if the line count is >= to 10. When it is full stop
* and allow the operator to view the data then clear the screen and continue
if LINE_COUNT >= 18

```

```

    ? LINE
    * hold screen for viewing
    wait
    clear
    PAGE_NO = PAGE_NO + 1
    LINE_COUNT = 3
    ? SPACE(60) + str(PAGE_NO,7)
    ? TITLE5
    ? TITLE6
    ? LINE
endif

```

```

* print the records on the screen
TFNAME = trim(F_NAME)
TLNAME = trim(L_NAME)
?
? " "+TFNAME+" "+TLNAME + SPACE(21-LEN(TFNAME+TLNAME))+;
    DTOC(REG_DATE) + " " + PATRON_ID + " " + trim(STREET) +;
    SPACE(20-LEN(trim(STREET))) + trim(Dept)
? SPACE(39)+trim(CITY)+", "+STATE+" "+ZIP+;
    SPACE(10 - LEN(trim(CITY)))+trim(SECTION)
? SPACE(39) + PHONE + SPACE(7) + SMC_NO
SKIP
LINE_COUNT = LINE_COUNT + 3
enddo

```

```

* print the report footer

```



```

? LINE
?
if LINE_COUNT >= 10
    wait
    clear
    PAGE_NO = PAGE_NO + 1
    ? space(70) + str(PAGE_NO,7)
endif
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
do CENTER with TITLE4,80
?

* hold last screen for viewing
wait

* return to menu
return

*****
* Program Id      : LLMS2330
* Program        : PTRNLISC.PRG
* Author         : Park, Seong Seung
* Date          : 22 Oct 1989
* Software       : dBASE IV
* Description    : This program prints the patrons list to the file
*                : PTRNLIST.OUT.
*****

set alternate to PTRNLIST.OUT
set alternate on
set console on
go top

* declare private variables
Private TITLE1
Private TITLE2
Private TITLE3
Private TITLE4
Private TITLE5
Private TITLE6

* initialize variables
TITLE1 = "Library Loan Management System"
TITLE2 = "Patrons List"
TITLE3 = "As of " + proptime(date())
TITLE4 = "Total patrons listed: " + str(reccount(),4)
TITLE5 = space(32) + "Patron" + space(2) + "Patron"
TITLE6 = space(1)+"Name"+space(20)+"Date"+space(3)+"Id Num"+space(2)+;

```

```
"Address / Phone"+space(4)+"Dept/Section/SMC-No"
LINE = " _____ " +;
```

```
* print file heading
do CENTER with TITLE1,80
do CENTER WITH TITLE2,80
do CENTER with TITLE3,80
?
?
? TITLE5
? TITLE6
? LINE

* print the records to the file
do while .not. EOF()
  TFNAME = trim(F_NAME)
  TLNAME = trim(L_NAME)
  ? " "+TFNAME+" "+TLNAME+SPACE(21-LEN(TFNAME+TLNAME)) +;
  DTOC(REG_DATE)+ " "+PATRON_ID+" "+trim(STREET)+;
  SPACE(20 - LEN(trim(STREET)))+trim(DEPT)
  ? SPACE(39) + trim(CITY) + ", " + STATE + " " + ZIP +;
  SPACE(10 - LEN(trim(CITY))) + trim(SECTION)
  ? SPACE(39) + PHONE + SPACE(7) + SMC_NO
  SKIP
enddo
? LINE

* print report footer
?
?
do CENTER with TITLE1,80
do CENTER with TITLE2,80
do CENTER with TITLE3,80
?
do CENTER with TITLE4,80

* restore initial conditions
set console on
set alternate off
close alternate

* exit and return to menu
return
```

```
*****
* Program Id      : LLMS3000      *
* Program        : CIRCULA.PRG   *
* Author         : Park, Seong Seung *
* Date           : 18 Oct 1989   *
* Software       : dBASE IV      *
```

```

* Description      : This program displays the book circulation menu and calls
*                  the subprogram for managing circulation and relationship
*                  between patrons and books.
*****

```

```

set function 10 to "4"
set escape on

```

```

* initialize variables
Loan = 5      && number of options for HILIGHT procedure
Last = 27
Private MyNum && local variable
MyNum = 0

```

```

* displays circulation menu until Last
do while MyNum # Last
  Content1 = " Record Check-out of Books"
  Content2 = " Record Check-in of Books"
  Content3 = " Inquire about the status of a Book"
  Content4 = " Print a circulation summary "
  Content5 = " Print a list of overdue books"
  Opn1 = ""
  Opn2 = ""
  Opn3 = ""
  Opn4 = ""
  Opn5 = ""
  LoanTitle = " LLMS - Circulation - [ESC] to exit "
  clear

```

```

* execute procedure HILIGHT to display menu
set procedure to ProCLib1
do HILIGHT with Loan, LoanTitle
do case
  case MyNum = 1
    do CIRLOAN.PRG
    on escape
  case MyNum = 2
    do CIRRTUN.PRG
    on escape
  case MyNum = 3
    do CIRSTAT.PRG
    on escape
  case MyNum = 4
    do CIRSMRY.PRG
    on escape
  case MyNum = 5
    do CIROVDU.PRG
    on escape
  case MyNum = 27
    exit

```

```
endcase
enddo
```

```
* return to main menu
close databases
return
```

```
*****
* Program Id      : LLMS3100 *
* Program        : CIRLOAN.PRG *
* Author         : Park, Seong Seung *
* Date          : 22 Oct 1989 *
* Software       : dBASE IV *
* Description    : This program manages check-out procedures for LLMS *
*                : system. *
*                : It calls the program for getting patron id(CIRGETID.PRG). *
*                : The program handles a transaction for the entry of the *
*                : book items and the check out. *
*                : It will automatically update BOOK, CIRCUL, R_PTBK, *
*                : R_CRPB databases. *
*****
```

```
clear
set proc to CIRGETID.PRG
```

```
* initialize variables
```

```
linecount = 0
bookcount = 0
patnum = " "
Ctype = " "
loaned = "O"
maxof4 = .F.
cirdone = .F.
cirnum = 0
cirdate = date()
reserve = "R"
```

```
* prompt the user and return the patron's id number
do CIRGETID.PRG
if patnum # "ESCAPE"
```

```
    * open the database and index values
```

```
    select A
    use BOOK order ITEM_NO alias BO_OK
    select B
    use R_CRPB order ITEM_NO alias CIR_CRN
    select C
    use PATRON order PATRON_ID alias PAT_RON
```

```
    * get the last circulation number from CIRCUL
    select D
```

```

use CIRCUL
go bott
cirnum = VAL(CIR_NO)
go top

select C
go top
seek patnum
if found()
  * begin loop to retrieve patnum, update records and related
  oncemore = .T.
  do while oncemore
    clear
    cirnum = cirnum + 1
    @ 1,1 to 23,78 DOUBLE
    @ 1,12 say " Library Loan Management System — Check-out procedure"
    @ 2,1
    ? space(3) + "Patron number: " + PATRON_ID + space(5) +
      "Check-out to: " + trim(F_NAME) + " " + trim(L_NAME)
    ? " Circulation number: " + str(cirnum,10) + space(17) + STREET
    ? space(3) + "Circulation date : " + dtoc(date()) + space(18) +
      trim(CITY) + ", " + STATE + " " + ZIP
    ? space(3) + "Item Book"
    @ 7,3 say "Number Title"
    @ 7,60 say "Call_No"
    @ 8,3 say "_____"
    @ 8,10 say replicate("-",49)
    @ 8,60 say "_____"

    * print message if restricted patron
    clear gets
    set confirm on

    * error trap for incorrect input for circulation type
    do while .not. (Ctype$"OoRr")
      @ 6,23 say "Circulation Type: " get Ctype picture "!"
      @ 24,1 say "Type the item number then [ENTER], [ESC] when done"
      @ 2,1 to 22,1 DOUBLE
      @ 2,78 to 22,78 DOUBLE
      read
      if readkey() = 12
        close proc
        close all
        return
      endif
    enddo
    set confirm off

    * loop to retrieve book numbers, update, and totals
    Ctype = UPPER(Ctype)
    working = .T.

```

```

row = 9
clear gets
do while working
  ok = .F.

  * check for correct item number
  do while .not. ok
    bookno = space(6)
    @ row,3 get bookno picture "XXXXXX"
    read

    * check for escape key
    if readkey() = 12
      working = .F.
      exit
    endif (escape)
    @ 3,1 to 22,1 DOUBLE
    @ 3,78 to 22,78 DOUBLE

    * search BOOK database
    select A
    go top
    seek bookno
    do case
      case maxof4
        if bookcount = 4
          ok = .T.
          working = .F.
        endif
      case STATUS = "O"
        @ row,3 say bookno + ": That book is already checked out!"
        bookno = " "
        ok = .F.
      case .not. found()
        @ row,3 say bookno + ": Item_No is NOT found!"
        bookno = " "
        ok = .F.
      case found()
        @ row,10 get TITLE
        @ row,60 get call_no picture "XXXXXXXXXXXXXXXXXXXX"
        @ 3,78 to 22,78 DOUBLE
        @ 3,1 TO 22,1 DOUBLE
        clear gets
        recnum = recno()
        bookcount = bookcount + 1

        * begin updating the data files
        if Ctype = loaned
          replace STATUS with loaned

          * append the new record to the CIRCUL file

```

```

        if cirdone = .F.
            select D
            append blank
            replace CIR_DATE      with cirdate
            replace CIR_TYPE      with Ctype
            replace CIR_NO        with str(cirnum,10)
            cirdone = .T.
        endif

        * append new record to R_PTBK file
        select E
        use R_PTBK
        append blank
        replace PATRON_ID        with patnum
        replace ITEM_NO          with bookno

        * append new record to R_CRPB file
        select B
        append blank
        replace PATRON_ID        with patnum
        replace ITEM_NO          with bookno
        replace CIR_NO           with str(cirnum,10)

    else
        replace STATUS           with reserve
    endif
endcase

    row = row + 1
enddo (.not. ok)

if (row >= 19)
    @ 24,1
    ?
    row = 19
endif
enddo (working)

@ row + 1,1
? space(66) + " _____ "
?
? space(50) + "Total Check-out" + space(2) + str(bookcount,3)
? space(66) + " _____ "
@ 3,1 to 22,1 DOUBLE
@ 3,78 to 22,78 DOUBLE
oncemore = .F.
wait
enddo (oncemore)
else
clear
@15,5 say "Patron id NOT found - exit and get help"

```

```

        @17,5 say ""
        wait
    endif (if not found)

endif (escape pressed)

* close databases and return
close databases
return

*****
* Program Id       : LLMS3110
* Program          : CIRGETID.PRG
* Author           : Park, Seong Seung
* Date             : 23 Oct 1989
* Software         : dBase IV
* Description      : This program is the prompt procedure to get the patrons
*                  : identification number for Book Check-outs. It is called
*                  : from any program, this procedure will automatically return
*                  : the patrons identification number, if found. A help
*                  : procedure, PTRNHELP.PRO is provided if the poor
*                  : individual can't remember his identification number.
*****

clear
set proc to PTRNHELP.PRG
on escape return

* initialize variables
patnum = space(5)
done = .F.

* prompt the clerk for the patrons identification number
do while .not. done
    @ 15,20 say "Enter: "
    @ 16,22 say "ESC to return to the previous menu"
    @ 17,22 say "H for Help"
    @ 18,22 say "Enter Patron ID Number: " get patnum picture "!XXX9"
    read

    if readkey() = 12
        close proc
        patnum = "ESCAPE"
        done = .T.
    endif
    patnum = UPPER(patnum)
    if patnum = "H"
        do PTRNHELP.PRG
    else
        done = .T.
    endif
enddo

```



```

endif
enddo

* close databases and return
close databases
close proc
return

*****
* Program Id      : LLMS3200
* Program        : CIRRTUN.PRG
* Author         : Park, Seong Seung
* Date           : 24 Oct 1989
* Software       : dBASE IV
* Description     : This program handles check-in procedure for LLMS system.*
*                : It checks books in returned by the patrons and
*                : automatically updates databases. Operator can see a list
*                : of all books checked out by the patrons. Then, operator
*                : needs to answer if all books not returned. If all books
*                : are not returned, operator inputs books which is not
*                : returned. PTRNHELP.PRG give you helps for patron id.
*****

clear
close databases

* initialize procedures and variables
set proc to CIRGETID.PRG
bookcnt = 0
scncnt = 0
patnum = " "
YN = " "
gotdata = .T.
booknum = space(6)

* prompt the user and return the patron's id number
do CIRGETID.PRG
if patnum # "ESCAPE"

    * get the PATRON information from the PATRON file
    select A
    use PATRON order PATRON_ID alias PAT_RON
    go top
    seek patnum
    if found()

        * print the patron's information and item number header
        clear
        @ 2,1 to 20,78 DOUBLE
        @ 3,2 say trim(L_NAME) + ", " + trim(F_NAME) + " " + " has the "
        @ 4,2 say "following books out:"

```

```

@ 5,2 say "Item_No Title Call_No"
@ 6,2 say "_____";
"_____";
@ 22,2 say "Make a note of any book which has not been returned."

* open databases and index files
select B
use BOOK order ITEM_NO alias BO_OK

* JOIN (R_PT BK, BOOK)
select C
use R_PT BK order ITEM_NO alias PT_BK
set relation to ITEM_NO into BO_OK
row = 7

* main program loop
do while .T.
  go top
  do while .not. EOF()
    tempstatus = BO_OK->STATUS

    if tempstatus = "O" .or. tempstatus = "R"
      if PATRON_ID = patnum
        @ row,2 get BO_OK->ITEM_NO
        @ row,9 get BO_OK->TITLE
        @ row,60 get BO_OK->CALL_NO
        row = row + 1
        bookcnt = bookcnt + 1
      endif
    endif
    skip
  enddo

* no books checked out according to book file
if bookcnt = 0
  @ row,10 say "Patron has no books checked out. "
  @ row+2,10
  wait
  exit
endif
?
?
@ 22,1
wait
clear
row = 7
go top

* prompt to see if all the books are being turned in
do while .not. YN$"YyNn"
  @ 10,10 say "Have all books been returned? (Y/N) " get YN

```

```

    read
  enddo
  clear
  YN = UPPER(YN)

  * execute response
  do case
    * all books returned -- update book file
    case YN = "Y"
      go top
      * update BOOK file and delete R_PTBK record
      do while .not. EOF()
        recnum = recno()
        tempstatus = BO_OK->STATUS
        if tempstatus = "O" .and. PATRON_ID = patnum
          replace BO_OK->STATUS with "I"
          delete record recnum
        endif
        skip
      enddo
      go top
      booknum = space(6)
      @ row+6,15
      wait

    * all books not returned
    case YN = "N"
      go top

      * loop through for the books not being returned
      @ 2,1 to 20,78 DOUBLE
      @ row+1,2 say "Enter Item Number of the book not returned: "
      @ row+2,2 say "Or press [ENTER] to continue when done."
      @ row+3,2 say;
      "Item_No Title Call_No"
      @ row+4,2 say "_____";
      "_____";

      row = row + 5
      do while gotdata .and. scnct <= 5
        booknum = space(6)
        @ row,2 get booknum
        read
        if booknum # space(6)
          seek booknum
          if found()
            @ row,9 say BO_OK->TITLE
            @ row,60 say BO_OK->CALL_NO
            row = row + 1
            if BO_OK->STATUS = "O"
              replace BO_OK->STATUS with "H"
              booknum = space(6)
            endif
          endif
        endif
      enddo
    endcase
  enddo

```

```

        scncnt = scncnt + 1
    endif
    if BO_OK->STATUS = "I"
        @ row,2 say booknum +;
        ": This book is already checked in!"
        row = row + 1
        booknum = space(6)
    endif
    else
        @ row,2 say booknum + ": Item_No is Not found!"
        booknum = space(6)
        row = row + 1
    endif
    else
        gotdata = .F.
    endif
    go top
enddo
clear

* now finish the update with the books being returned
go top
do while .not. EOF()
    tempstatus = BO_OK->STATUS
    recnum = recno()
    if tempstatus = "H" .and. PATRON_ID # patnum
        replace BO_OK->STATUS with "O"
    else
        if tempstatus = "H" .and. PATRON_ID = patnum
            replace BO_OK->STATUS with "O"
        else
            if tempstatus = "O" .and. PATRON_ID = patnum
                replace BO_OK->STATUS with "I"
                delete record recnum
            endif
        endif
    endif
    skip
enddo
endcase
exit
enddo
select PT_BK
set relation to
pack
clear

* update completed
@ 17,5 say "Update completed -- press any key to continue."
wait ""

```

```

* the patron number was not found
else
  clear
  @ 10,15 say "Patron has no books checked out."
  @ 11,15 say "(or the patron number does not exist)."
  @ 13,15 say " "
  wait
endif
endif

```

```

* close all databases
close all
return

```

```

*****
* Program Id      : LLMS3300
* Program         : CIRSTAT.PRG
* Author          : Park, Seong Seung
* Date            : 24 Oct 1989
* Software        : dBASE IV
* Description     : This program shows the status of a book. If one of the
*                  patrons want to know if a book is available for loan, you
*                  are asked to enter book id number.
*                  If the book is checked out, information about the patron
*                  who checked out the book will be displayed.
*****

```

```

close databases
clear
set proc to CIRGTBNO.PRG
set escape off

```

```

* open databases
select A
use PATRON order PATRON_ID alias PAT_RON
select B
use BOOK order ITEM_NO alias BO_OK
select C
use R_CRPB order ITEM_NO alias CR_PB
select CR_PB
set relation to ITEM_NO into BO_OK

```

```

* initialize variables
tempstatus = " "
header = "Status of the Book"
available = "available"
booknum = " "
cirdate = ctod(" / / ")
enter = "[ENTER]"
got = .T.

```

```

* prompt the clerk for the Item number
do CIRGTBNO.PRG
clear
if booknum = "ESCAPE"
    close proc
    close databases
    return
endif

* searches for the Item number
select B
goto top
on error exit
do while .T. .or. .not. bof()
    if booknum # ITEM_NO
        skip + 1
    else
        exit
    endif
enddo
on error

* do the header
define window STATBNO from 5,10 to 18,70 double
activate window STATBNO
@ 0,19 to 2,42
@ 1,22 get header
?
?

* print the status if found
go top
seek booknum
if found()
    select C
    tempstatus = BO_OK->STATUS
    booktitle = BO_OK->TITLE
    booktitle = upper(booktitle)
    if tempstatus = "O"
        @ 3,1
        ? space(7) + "Title: " + booktitle
        set relation to

        * go into PATRON file, get PATRON info and print it
        set relation to PATRON_ID into PATRON
        goto bottom
        do while .not. BOF() .and. got
            if booknum = ITEM_NO
                fstname = PATRON->F_NAME
                lstname = PATRON->L_NAME
                ? space(7)+"Checked out to: "+trim(fstname)+" "+trim(lstname)
            endif
        enddo
    endif
endif

```

```

        ? space(7)+"Phone:          " + PAT_RON->PHONE
        ? space(7)+"SMC No:         " + PAT_RON->SMC_NO
        got = .F.
    else
        skip -1 in C
    endif
enddo
set relation to

* go into the CIRCULATION file and get the check out date
set procedure to ProcLib1
select E
use CIRCUL order CIR_NO alias CIR_CUL
select C
set relation to CIR_NO into CIR_CUL
cirdate = CIR_CUL->CIR_DATE
? space(7) + "Date:          " + proptime(cirdate)
?
wait
set relation to
endif

if tempstatus = "I"
    @ 4,1
    ? space(7) + "Title: " + booktitle
    ? space(7) + "Status: " + upper(available)
    @ 10,1
    wait
endif
else
    @ 4,1
    ? space(15) + " That Item number was not found."
    ? space(15) + " Please exit and retry with the "
    ? space(15) + " book item help."
    @ 10,1
    wait
endif

* erase the window
deactivate window STATBNO
release window STATBNO

* close databases
close proc
close database
return

*****
* Program Id       : LLMS3310
* Program         : CIRGTBNO.PRG
* Author          : Park, Seong Seung
*****

```

```

* Date           : 24 Oct 1989
* Software       : dBase IV
* Description    : This program prompts the user for a item number.
*                : It is called from any program that requires a book item
*                : number. This program will return a item number and calls
*                : a help program if a listing of all books in the club is
*                : needed.
*****

```

```

* set up initial parameters
clear
set proc to CIRCHELP.PRG

```

```

* initialize variables
booknum = space(6)

```

```

* display prompt to clerk
done = .F.
do while .not. done
  clear
  @ 15,20 say "Enter: "
  @ 16,22 say "[ESC] to return to previous menu"
  @ 17,22 say "Enter Book Item Number: " get booknum picture "!XXXXX"
  read
  if readkey() = 12
    booknum = "Escape"
    done = .T.
  endif
  booknum = upper(booknum)

  if booknum = "H"
    do CIRCHELP.PRG
    booknum = space(6)
  else
    done = .T.
  endif
enddo
on escape

```

```

* exit and return
return

```

```

*****
* Program Id    : LLMS3320
* Program      : CIRCHELP.PRG
* Author       : Park, Seong Seung
* Date        : 25 Oct 1989
* Software     : dBASE IV
* Description  : This program is the help program for the book status
*              : query. It produces a listing of all books in the library.
*****

```



```

* set up initial parameters
clear
set bell off
on escape return

* initialize variables
search = " "
allitems = space(3)
keyword = space(15)
linecount = 1

* display prompt to clerk
again = .T.
do while again
  @ 15,20 say "ID Number Locator"
  @ 16,20 say "Type 'ALL' for a list of all books" get allitems picture "!!!"
  @ 17,20 say "Press [ESC] to exit help"
  read

  * escape if escape key is pressed
  if readkey() = 12
    return
  endif
  allitems = upper(allitems)

  * display a list of all books
  do case
    case allitems = " "
      exit

    case allitems = "ALL"
      select B
      row = 7
      col = 5
      clear
      @ 3,35 say "Book Listing"
      @ 5,2 say "Item_No      Title                      Call_No"
      @ 6,2 say " _____ " +;
      " _____ "
      go top
      linecount = 7
      do while .not. EOF(2)
        ? space(2) + ITEM_NO + space(1) + TITLE + space(1) + CALL_NO
        skip
        linecount = linecount + 1
        if linecount = 2
          wait
          clear
          @ 5,15 say " Book List Continued"
          @ 6,15 say " _____ "

```

```

        linecount = 7
    endif
enddo
?
?
wait
clear
allitems = " "
go top
again = .F.
select C
endcase
enddo

* exit and return
return

*****
* Program Id       : LLMS3400
* Program         : CIRSMRY.PRG
* Author          : Park, Seong Seung
* Date            : 25 Oct 1989
* Software        : dBASE IV
* Description     : This program produces the circulation summary report.
*                  This program prompts the user for the beginning and ending
*                  dates. It checks only valid dates. A summary report can
*                  be generated for any period of time. The circulation number
*                  for the starting date is retrieved from the CIRCUL file
*                  and the majority of the searches are accomplished in the
*                  R_CRPB file with relations set into the BOOK file.
*                  It is assumed that all circulations are in circulation
*                  number, otherwise the datafiles and report information
*                  will be corrupted.
*****

clear
set talk off
set proc to ProcLib1
on escape do stoplist
set margin to 4
linecount = 1
pagecount = 1
pagelength = 53
done = .F.
ttotal = 0
dtotal = 0
gtotal = 0
patnum = " "
title1 = "Library Loan Management System"
title2 = "Circulation Register"
start = ctod(" / / ")

```

```

stop = ctod(" / / ")
tempdate = ctod(" / / ")

* query user for output destination
opt = " "
do CIRSMRYL with opt
clear

* open databases and index file
select A
use BOOK order ITEM_NO alias BO_OK
select B
use R_PTBK order ITEM_NO alias PT_BK
select C
use CIRCUL alias CR_CL

* set the relations
select B
set relation to ITEM_NO into BO_OK

* need begin and end dates for the report
do while .T.
  @ 5,5 say "Please enter Beginning Report Date: " get start
  @ 6,5 say "Please enter Ending Report Date : " get stop
  @ 8,5 say "Format for date input is MM/DD/YY"
  read
  if readkey() = 12
    set print off
    set alternate off
    close proc
    close all
    return
  endif

  * make sure starting date is before ending date
  * make sure a date range gets entered
  if dtoc(start) # " " .and. dtoc(stop) # " " .and. start <= stop
    exit
  endif

  if dtoc(start) = " " .or. dtoc(stop) = " "
    @ 10,5 say "You must enter a date range for the report – Please Reenter"
  endif
  if start >= stop .and. dtoc(start) # " " .and. dtoc(stop) # " "
    @ 10,5 clear
    @ 10,5 say "Ending date occurs before starting date – Please Reenter"
  endif
enddo
clear

```

```

* do the header
?
earlist = start
latest = stop
do CENTER with title1,80
do CENTER with title2,80
tempo1 = "From " + proptime(earlist) + " To " + proptime(latest)
do CENTER with tempo1,80
tempo2 = "Printed at " + time() + " on " + proptime(date())
do CENTER with tempo2,80
do CENTER with "Page" + str(pagecount,3),80
?
linecount = 7

* start main loop to retrieve data
select C
go top
* find starting record
do while .not. done .and. .not. EOF(3)
  if CIR_DATE >= start .and. CIR_DATE <= stop
    done = .T.
  else
    skip
  endif
enddo

* loop through circulation file
do while CIR_DATE >= start .and. CIR_DATE <= stop
  cirnum = CIR_NO
  tempdate = CIR_DATE
  do while CIR_DATE = tempdate
    select D
    use R_CRPB order CIR_NO alias CR_PB
    set relation to ITEM_NO into BO_OK
    seek cirnum

    * loop through to last date in the R_CRPB file
    rupdate = tempdate
    ? "Date: "+space(3)+ proptime(rupdate)
    * loop through R_CRPB for each circulation
    patnum = PATRON_ID
    ? "Circulation Number: "+ltrim(CIR_NO)
    select E
    use PATRON order PATRON_ID alias PATRON
    seek patnum
    ? "Patron: "+space(3)+trim(L_NAME)+"," +trim(F_NAME)
    ? "Item"
    ? "Number"+space(5)+"Title"+space(46)+" Quantity"
    ? "
    "_____";
  linecount = linecount + 6

```

```

select D

*loop through same circulation numbers
do while CIR_NO = cirnum .and. .not. EOF()
  ? BO_OK->ITEM_NO + " " + BO_OK->TITLE + space(5) + "1"
  ttotal = ttotal + 1
  linecount = linecount + 1

  * screen page
  if opt = "S" .or. opt = "A"
    if linecount >= 18
      ?
      wait
      clear
      linecount = 1
    endif
  endif

  if linecount >= pagelength .and. opt = "P"
    eject
    pagecount = pagecount + 1
    ? space(20) + "Circulation Summary Continued"
    ? space(28) + "Page" + str(pagecount,3)
    ?
    ?
    linecount = 4
  endif
  skip
enddo
cirnum = CIR_NO
go top
select C
go top
locate for CIR_NO = cirnum
? space(62) + "_____ "
if ttotal = 0
  bkstr = space(6)
else
  if ttotal = 1
    bkstr = " book "
  else
    bkstr = " books"
  endif
endif
? space(21) + "Circulation total" + space(23) + str(ttotal,5) + bkstr
?
?
linecount = linecount + 4
dtotal = dtotal + ttotal
ttotal = 0
enddo

```

```

if opt = "S" .or. opt = "A"
  if linecount >= 18
    ?
    wait
    clear
    linecount = 1
  endif
endif

if linecount >= pagelength .and. opt = "P"
  eject
  pagecount = pagecount + 1
  ? space(20) + "Circulation Summary Continued"
  ? space(28) + "Page" + str(pagecount,3)
  ?
  ?
  linecount = 4
endif

* finish up the totals and print
if dtotal = 0
  bkstr = space(6)
else
  if dtotal = 1
    bkstr = " book "
  else
    bkstr = " books"
  endif
endif
? space(21) + "Daily total" + space(29) + str(dtotal,5) + bkstr
? space(62) + " _____"
?
linecount = linecount + 3
gtotal = gtotal + dtotal
ttotal = 0
dtotal = 0
enddo
if gtotal = 0
  bkstr = space(6)
else
  if gtotal = 1
    bkstr = " book "
  else
    bkstr = " books"
  endif
endif
? space(21) + "Summary Total" + space(27) + str(gtotal,5) + bkstr
? space(62) + " _____"
if opt = "S" .or. opt = "A"
  ?

```

```
wait
endif
set print off
set alternate off
close all
return
```

```
*****
* Program Id      : LLMS3410
* Program        : CIRSMRYL.PRG
* Author         : Park, Seong Seung
* Date          : 27 Oct 1989
* Software       : dBASE IV
* Description    : This program directs the output of a report to the
*                : printer,screen, or a file.
*****
```

```
* set up initial parameters
procedure cirsmryl
parameters opt
clear
set talk off
```

```
* declare private variables
private start_psn
private no_sel
private option
private selection
private column
private text
private output_mode
```

```
* frame screen and print title
@ 2,0 to 4,79 DOUBLE
@ 2,23 say "Select the report destination"
```

```
* initialize variables
store "Printer Screen File" to text
no_sel = .T.
option = 1
store "011121" to start_psn
```

```
* loop until selection is made
do while no_sel
* display selections
@ 3,3 say text

*remember selection
if option = 1
store "Printer" to selection
```

```

endif
if option = 2
    store "Screen" to selection
endif
if option = 3
    store "File" to selection
endif
column = val(substr(start_psn,option*2-1,2)) +2

* print selection in reverse video
set color to bg+/gr+
@ 3,column say selection

* get selection
I = 0
do while I = 0
    I = inkey()
enddo
do case
    case I = 4
        if option = 3
            option = 1
        else
            option = option + 1
        endif
    case I = 19
        if option = 1
            option = 3
        else
            option = option - 1
        endif

* execute selection made with return key
case I = 13
    no_sel = .F.
    do case
        case option = 1
            set print on
            opt = "P"
            clear
        case option = 2
            set device to screen
            opt = "S"
            clear
        case option = 3
            set alternate to CIRSMRY.OUT
            set alternate on
            opt = "A"
            clear
    endcase

```



```

* execute selection made with letter key
case upper(CHR(I)) $ "PSF"
  no_sel = .F.
  do case
    case upper(CHR(I)) = "P"
      set print on
      opt = "P"
      clear
    case upper(CHR(I)) = "S"
      set device to screen
      opt = "S"
      clear
    case upper(CHR(I)) = "F"
      set alternate to CIRSMRY.OUT
      set alternate on
      opt = "A"
      clear
  endcase
endcase

* return color to normal
set color of normal to w+/b
set color of highlight to w+/rb
set color of fields to n/g
set color of box to n/bg
enddo

* redirect output back to the screen
set device to screen
close databases
return

*****
* Program Id      : LLMS3500 *
* Program        : CIROVDU.PRG *
* Author         : Park, Seong Seung *
* Date           : 26 Oct 1989 *
* Software       : dBASE IV *
* Description    : This program is the overdue book report for LLMS. *
*                : It returns all those books not returned based on todays *
*                : date minus 15 days, since books loaned two weeks ago are *
*                : not due in until closing. A search is made of the R_CRPB *
*                : file from the bottom up. Searching conditions are based *
*                : on the dates found in the CIRCUL file. The report may be *
*                : directed to a printer, screen, of a file. The output *
*                : prompts are provided in a external file. *
*****

* set up initial parameters
set margin to 2

```

```

* query the user for output destination – returns option
opt = 2
do CIROVDUL with opt

* open database and index files
select A
use BOOK order ITEM_NO alias BO_OK
select B
use R_PTBK order ITEM_NO alias PT_BK
select C
use PATRON order PATRON_ID alias PAT_RON
select D
use R_CRPB order ITEM_NO alias CR_PB

* set the relations
select BO_OK
set relation to ITEM_NO into PT_BK
select B
set relation to PATRON_ID into PAT_RON

* do the report header
clear
if opt # "P"
  if opt # "A"
    @ 1,1 to 6,78 DOUBLE
    @ 2,17 say " Library Loan Management System "
  endif
endif
? "
? "          Overdue Book Report"
? space(14) + "as of " + time() + " on " + cmonth(date()) + " " +;
  ltrim(str(DAY(DATE()),2)) + ", " + ltrim(str(YEAR(DATE()))))
if opt # "P"
  if opt # "A"
    @ 2,1 to 5,1 DOUBLE
  endif
endif
?
?

* initialize working variables
testcnt = 0
cirnum = " "
cirdate = ctod("/ / ")
latedate = ctod("/ / ")
tempstatus = " "
linecount = 7
pagelength = 55
pagecount = 1

* get the target date = today's date minus 15
latedate = date() - 15

```

```

? "          Book Late Date is " + dtoc(latedate)
?

* open the BOOK file
select A
go top

* loop through the book numbers in BOOK
do while .not. EOF([BO_OK])
  booknum = ITEM_NO
  seek booknum
  if STATUS = "O" .or. STATUS = "R"
    select CR_PB
    go bott
    do while .not. BOF([CR_PB])
      if ITEM_NO # booknum
        skip -1
      else
        patnum = PATRON_ID
        cirnum = CIR_NO
        exit
      endif
    enddo
    select E
    use CIRCUL order CIR_NO
    go top
    locate for CIR_NO = cirnum
    cirdate = CIR_DATE
    select A
    if cirdate <= latedate
      select C
      go top
      seek patnum

      * print patron's information
      ? " Patron Name: " + trim(PAT_RON->F_NAME) + space(1) +
        trim(PAT_RON->L_NAME)
      ? " Phone: " + PAT_RON->PHONE + space(5) + "SMC No: " +
        PAT_RON->SMC_NO

      * reopen BO_OK - pointer is already in position
      select A
      go top
      locate for ITEM_NO = booknum

      * print the book information
      ?
      ? "Item_No   Title                               Date Out"
      ? " _____"
      ? ITEM_NO + space(2) + TITLE + space(1) + cr. nth(cirdate) + " " +;

```

```

        ltrim(str(day(cirdate),2)) + "," + ltrim(str(year(cirdate)))
    ?
    linecount = linecount + 6
    testcnt = testcnt + 1

    * page the output to the printer
    if linecount >= pagelength .and. opt = "P"
        eject
        pagecount = pagecount + 1
        ?
        ?
        ? space(17) + "Overdue Report Continued"
        ? space(25) + "Page " + str(pagecount,3)
        ?
        linecount = 5
    endif

    * page the screen output
    if linecount > 20 .and. opt = "S"
        wait
        clear
        ?
        linecount = 1
    endif
endif
endif
select A
if .not. EOF([BO_OK])
    skip
endif
enddo
if testcnt = 0
    if opt = "A" .or. opt = "P"
        ?
        ?
        ?
        ?
        ? "                No Overdue Books"
    endif
    @ 10,27 say "No overdue Books."
    if opt = "A" .or. opt = "P"
        set console on
    endif
else
    if opt = "A" .or. opt = "P"
        set console on
    endif
endif
endif

* close databases
if opt = "A" .or. opt = "P"

```

```
eject
else
  wait
endif
set alternate off
set print off
set device to screen
close all
close proc
return
```

```
*****
* Program Id      : LLMS3510                      *
* Program        : CIROVDUL.PRG                  *
* Author         : Park, Seong Seung              *
* Date          : 27 Oct 1989                    *
* Software       : dBASE IV                       *
* Description    : This program directs the output of a report to the *
*                  printer,screen, or a file.     *
*****
```

```
* set up initial parameters
procedure cirovdul
parameters opt
clear
set talk off
```

```
* declare private variables
private start_psn
private no_sel
private option
private selection
private column
private text
private output_mode
```

```
* frame screen and print title
@ 2,0 to 4,79 DOUBLE
@ 2,23 say "Select the report destination"
```

```
* initialize variables
store "Printer Screen File" to text
no_sel = .T.
option = 1
store "011121" to start_psn
```

```
* loop until selection is made
do while no_sel
  * display selections
  @ 3,3 say text
```

```

*remember selection
if option = 1
  store "Printer" to selection
endif
if option = 2
  store "Screen" to selection
endif
if option = 3
  store "File" to selection
endif
column = val(substr(start_psn,option*2-1,2)) +2

* print selection in reverse video
set color to bg+/gr+
@ 3,column say selection

* get selection
I = 0
do while I = 0
  I = inkey()
enddo
do case
  case I = 4
    if option = 3
      option = 1
    else
      option = option + 1
    endif
  case I = 19
    if option = 1
      option = 3
    else
      option = option - 1
    endif

* execute selection made with return key
case I = 13
  no_sel = .F.
  do case
    case option = 1
      set print on
      opt = "P"
      clear
      @ 10,15 say "Printing the overdue list to the printer."
    case option = 2
      set device to screen
      opt = "S"
      clear
    case option = 3
      set alternate to CIROVDU.OUT
      set alternate on

```

```

        opt = "A"
        clear
        @ 10,15 say "Printing the overdue list to file CIROVDU.OUT."
    endcase

* execute selection made with letter key
case upper(CHR(I)) $ "PSF"
    no_sel = .F.
    do case
        case upper(CHR(I)) = "P"
            set print on
            opt = "P"
            clear
            @ 10,15 say "Printing the overdue list to the printer."
        case upper(CHR(I)) = "S"
            set device to screen
            opt = "S"
            clear
        case upper(CHR(I)) = "F"
            set alternate to CIROVDU.OUT
            set alternate on
            opt = "A"
            clear
            @ 10,15 say "Printing the overdue list to the file CIROVDU.OUT."
        endcase
    endcase
endcase

* return color to normal
set color of normal to w+/b
set color of highlight to w+/rb
set color of fields to n/g
set color of box to n/bg
enddo

* redirect output back to the screen
set device to screen
close databases
return

*****
* Program Id       : LLMS4000
* Program         : REFRNCE.PRG
* Author          : Park, Seong Seung
* Date            : 18 Oct 1989
* Software        : dBASE IV
* Description     : This program displays the reference service menu and
*                  calls subprograms for search request from user.
*****

set function 10 to "4"

```

```

set escape on

* initialize variables
Serch = 6      && number of options for HILIGHT procedure
Last = 27
Private MyNum  && local variable
MyNum = 0

* displays reference service menu until Last
do while MyNum # Last
  Content1 = " Search by Item Number"
  Content2 = " Search by Title"
  Content3 = " Search by Author"
  Content4 = " Search by Call Number"
  Content5 = " Search by ISBN"
  Content6 = " Search by Subject"
  Opn1 = ""
  Opn2 = ""
  Opn3 = ""
  Opn4 = ""
  Opn5 = ""
  Opn6 = ""
  SerchTitle = " LLMS – Reference Service – [ESC] to exit "
  clear

  * execute procedure HILIGHT to display menu
  set procedure to ProcLib1
  do HILIGHT with Serch, SerchTitle
  do case
    case MyNum = 1
      do REFITEM.PRG
      on escape
    case MyNum = 2
      do REFTITL.PRG
      on escape
    case MyNum = 3
      do REFAUTH.PRG
      on escape
    case MyNum = 4
      do REFCALL.PRG
      on escape
    case MyNum = 5
      do REFISBN.PRG
      on escape
    case MyNum = 6
      do REFSUBJ.PRG
      on escape
    case MyNum = 27
      exit
  endcase
enddo

```



```
* return to main menu
close databases
return
```

```
*****
* Program Id       : LLMS4100                               *
* Program         : REFITEM.PRG                             *
* Author          : Park, Seong Seung                       *
* Date            : 24 Oct 1989                             *
* Software        : dBASE IV                                *
* Description     : This program searches a book by Item Number. *
*                  You are asked to enter book item number. This program *
*                  reponses the information about the requested book. *
*****
```

```
clear
set proc to CIRGTBNO.PRG
set escape off
```

```
* open databases
select B
use BOOK order ITEM_NO
```

```
* initialize variables
booknum = " "
header = "BOOK INFORMATION"
```

```
* prompt the clerk for the Item number
do CIRGTBNO.PRG
clear
if booknum = "ESCAPE" .or. booknum = " "
    close proc
    close databases
    return
endif
```

```
* searches for the Item number
select B
goto top
on error exit
do while .T. .or. .not. bof()
    if booknum # ITEM_NO
        skip + 1
    else
        exit
    endif
enddo
on error
```

```
* do the header
```

```

define window STATBNO from 3,8 to 20,73 double
activate window STATBNO
@ 0,19 to 2,40
@ 1,22 get header
?

```

```

* print the status if found
seek booknum
if found()
  ? space(5) + "Item_No : " + ITEM_NO
  ? space(5) + "Author  : " + trim(AUTHOR_F) + " " + trim(AUTHOR_L)
  ? space(5) + "Title   : " + trim(TITLE)
  ? space(5) + "Edition : " + trim(EDITION) + " Edition"
  ? space(5) + "Subject : " + trim(SUBJECT)
  ? space(5) + "Publisher: " + trim(PUBLISHER)
  ? space(5) + "Year    : " + P_YEAR
  ? space(5) + "Call_No : " + trim(CALL_NO)
  ? space(5) + "ISBN   : " + trim(ISBN)
  if STATUS = "I"
    ? space(5) + "STATUS : AVAILABLE"
  else
    ? space(5) + "STATUS : Checked Out"
  endif
  ?
else
  @ 6,1
  ? space(15) + " That Item number was not found."
  ? space(15) + " Please exit and retry with the "
  ? space(15) + " book item help."
  ?
  ?
endif
wait

```

```

* erase the window
deactivate window STATBNO
release window STATBNO

```

```

* close databases
close database
return

```

```

*****
* Program Id       : LLMS4200 *
* Program         : REFTITL.PRG *
* Author          : Park, Seong Seung *
* Date            : 24 Oct 1989 *
* Software        : dBASE IV *
* Description     : This program searches a book by title. *
*                 : You are asked to enter book title. This program reponses *
*                 : the information about the requested book. *

```

```
clear
set escape off

* open databases
select B
use BOOK order ITEM_NO

* initialize variables
titlename = space(50)
header = "BOOK INFORMATION"
booknum = space(6)

* prompt the clerk for the TITLE
clear
@ 15,20 say "Enter: "
@ 16,22 say "[ESC] to return to previous menu"
@ 17,22 say "Enter Title : "
@ 18,22 get titlename
read
if readkey() = 12 .or. titlename = " "
    close databases
    return
endif
titlename = upper(titlename)
select B
clear
@ 3,15 say "          Book Listing "
@ 5,6 say " Item No          Title"
@ 6,6 say " _____" "+;
"
go top
linecount = 7
do while .not. EOF(2)
    if trim(titlename) = substr(upper(trim(TITLE)),1,len(trim(titlename)))
        ? space(6) + ITEM_NO + space(3) + TITLE
    endif
    skip
    linecount = linecount + 1
    if linecount >= 18
        @ 21,15 say "Type Item_No or 'G' for next" get booknum picture "!XXXXX"
        read
        clear
        if booknum # "G" .and. booknum # " "
            exit
        endif
        booknum = space(6)
        @ 5,15 say " Book List Continued"
        @ 6,15 say " _____"
        linecount = 7
    endif
enddo
```

```

endif
enddo
if booknum = "    "
    @ 21,15 say "Type Item_No" get booknum picture "!XXXXX"
    read
endif

* searches for the Item_No number
clear
select B
goto top
on error exit
do while .T. .or. .not. bof()
    if booknum # ITEM_NO
        skip + 1
    else
        exit
    endif
enddo
on error

* do the header
define window STATBNO from 3,8 to 20,73 double
activate window STATBNO
@ 0,19 to 2,40
@ 1,22 get header
?

* print the status if found
seek booknum
if found()
    ? space(5) + "Item_No :" + ITEM_NO
    ? space(5) + "Author  :" + trim(AUTHOR_F) + " " + trim(AUTHOR_L)
    ? space(5) + "Title   :" + trim(TITLE)
    ? space(5) + "Edition :" + trim(EDITION) + " Edition"
    ? space(5) + "Subject :" + trim(SUBJECT)
    ? space(5) + "Publisher:" + trim(PUBLISHER)
    ? space(5) + "Year    :" + P_YEAR
    ? space(5) + "Call_No :" + trim(CALL_NO)
    ? space(5) + "ISBN   :" + trim(ISBN)
    if STATUS = "I"
        ? space(5) + "STATUS  : AVAILABLE"
    else
        ? space(5) + "STATUS  : Checked Out"
    endif
    ?
else
    @ 6,1
    ? space(15) + " That Item number was not found."
    ? space(15) + " Please exit and retry with the "
    ? space(15) + " Item number help."
endif

```

```

    ?
    ?
endif
wait

* erase the window
deactivate window STATBNO
release window STATBNO

* close databases
close database
return

*****
* Program Id       : LLMS4300                      *
* Program          : REFAUTH.PRG                   *
* Author           : Park, Seong Seung             *
* Date             : 26 Oct 1989                   *
* Software         : dBASE IV                       *
* Description      : This program searches a book  *
                  : by author.                      *
                  : You are asked to enter author  *
                  : name. This program             *
                  : responds the information about  *
                  : the requested book.            *
*****

clear
set escape off

* open databases
select B
use BOOK order ITEM_NO

* initialize variables
authorname = space(30)
header = "BOOK INFORMATION"
booknum = space(6)

* prompt the clerk for the AUTHOR name
clear
@ 15,20 say "Enter: "
@ 16,22 say "[ESC] to return to previous menu"
@ 17,22 say "Enter Author Name(First name first): "
@ 18,22 get authorname
read
if readkey() = 12 .or. authorname = " "
    close databases
    return
endif
authorname = upper(authorname)
select B
clear
@ 3,15 say "      Book Listing "

```

```

@ 5,3 say " Item No           Title / Author"
@ 6,3 say " _____" +;
go top
linecount = 7
do while .not. EOF(2)
  tempname = upper(trim(AUTHOR_F) + " " + trim(AUTHOR_L))
  if trim(authurname) = substr(tempname,1,len(trim(authurname)))
    ? space(3) + ITEM_NO + " " + trim(TITLE) + " / " +;
    trim(AUTHOR_F) + " " + trim(AUTHOR_L)
  else
    if trim(authurname) =
      substr(upper(trim(AUTHOR_F)),1,len(trim(authurname))) .or.;
      trim(authurname) =
      substr(upper(trim(AUTHOR_L)),1,len(trim(authurname)))
    ? space(3) + ITEM_NO + " " + trim(TITLE) + " / " +;
      trim(AUTHOR_F) + " " + trim(AUTHOR_L)
    endif
  endif
  skip
  linecount = linecount + 1
  if linecount >= 18
    @ 21,15 say "Type Item_No or 'G' for next" get booknum picture "!XXXXX"
    read
    clear
    if booknum # "G" .and. booknum # " "
      exit
    endif
    booknum = space(6)
    @ 5,15 say " Book List Continued"
    @ 6,15 say " _____"
    linecount = 7
  endif
endif
enddo
if booknum = " "
  @ 21,15 say "Type Item_No" get booknum picture "!XXXXX"
  read
endif

* searches for the Item_No number
clear
select B
goto top
on error exit
do while .T. .or. .not. bof()
  if booknum # ITEM_NO
    skip + 1
  else
    exit
  endif
enddo

```

on error

```
* do the header
define window STATBNO from 3,8 to 20,73 double
activate window STATBNO
@ 0,19 to 2,40
@ 1,22 get header
?
```

```
* print the status if found
seek booknum
if found()
  ? space(5) + "Item_No :" + ITEM_NO
  ? space(5) + "Author  :" + trim(AUTHOR_F) + " " + trim(AUTHOR_L)
  ? space(5) + "Title   :" + trim(TITLE)
  ? space(5) + "Edition  :" + trim(EDITION) + " Edition"
  ? space(5) + "Subject  :" + trim(SUBJECT)
  ? space(5) + "Publisher:" + trim(PUBLISHER)
  ? space(5) + "Year    :" + P_YEAR
  ? space(5) + "Call_No  :" + trim(CALL_NO)
  ? space(5) + "ISBN    :" + trim(ISBN)
  if STATUS = "I"
    ? space(5) + "STATUS  : AVAILABLE"
  else
    ? space(5) + "STATUS  : Checked Out"
  endif
  ?
else
  @ 6,1
  ? space(15) + " That Item number was not found."
  ? space(15) + " Please exit and retry with the "
  ? space(15) + " Item number help."
  ?
  ?
endif
wait
```

```
* erase the window
deactivate window STATBNO
release window STATBNO
```

```
* close databases
close database
return
```

```
*****
* Program Id      : LLMS4400                      *
* Program        : REFCALL.PRG                    *
* Author         : Park, Seong Seung              *
* Date           : 24 Oct 1989                    *
* Software       : dBASE IV                       *
```

```

* Description      : This program searches a book by Call Number.      *
*                  You are asked to enter book call number. This program *
*                  reponses the information about the requested book.    *
*****

```

```

clear
set escape off

```

```

* open databases
select B
use BOOK order CALL_NO

```

```

* initialize variables
callnum = "      "
header = "BOOK INFORMATION"

```

```

* prompt the clerk for the Call number
done = .F.
do while .not. done

```

```

  clear
  @ 15,20 say "Enter: "
  @ 16,22 say "[ESC] to return to previous menu"
  @ 17,22 say "Enter Book Call Number: " get callnum picture "!!!!!!!!!!!!!!"
  read
  if readkey() = 12 .or. callnum = "      "
    callnum = "ESCAPE"
    done = .T.
  endif
  callnum = upper(callnum)

```

```

  if callnum = "H"
    clear
    search = " "
    allitems = space(3)
    linecount = 1
  
```

```

  * display prompt to clerk
  again = .T.
  do while again
    @ 15,20 say "ID Number Locator"
    @ 16,20 say "Type 'ALL' for a list of all books" get allitems picture "!!!"
    @ 17,20 say "Press [ESC] to exit help"
    read
    if readkey() = 12
      return
    endif
  
```

```

  * display a list of all books
  do case
    case allitems = " "
      exit
  
```



```

case allitems = "ALL"
  select B
  row = 7
  col = 5
  clear
  @ 3,15 say "          Book Listing "
  @ 5,6 say " Call_No          Title"
  @ 6,6 say " _____"
  "_____";
go top
linecount = 7
do while .not. EOF(2)
  ? space(6) + CALL_NO + space(2) + TITLE
  skip
  linecount = linecount + 1
  if linecount >= 20
    wait
    clear
    @ 5,15 say " Book List Continued"
    @ 6,15 say " _____"
    linecount = 7
  endif
enddo
?
wait
clear
allitems = " "
go top
again = .F.
endcase
enddo
callnum = space(18)
else
  done = .T.
endif
enddo
clear
if callnum = "ESCAPE"
  close databases
  return
endif

* searches for the Call number
select B
goto top
on error exit
do while .T. .or. .not. bof()
  if callnum # upper(CALL_NO)
    skip + 1
  else
    exit
  endif
enddo

```

```

endif
enddo
on error

* do the header
define window STATBNO from 3,8 to 20,73 double
activate window STATBNO
@ 0,19 to 2,40
@ 1,22 get header
?

* print the status if found
seek callnum
if found()
? space(5) + "Call_No :" + trim(CALL_NO)
? space(5) + "Item_No :" + ITEM_NO
? space(5) + "Author  :" + trim(AUTHOR_F) + " " + trim(AUTHOR_L)
? space(5) + "Title   :" + trim(TITLE)
? space(5) + "Edition  :" + trim(EDITION) + " Edition"
? space(5) + "Subject  :" + trim(SUBJECT)
? space(5) + "Publisher:" + trim(PUBLISHER)
? space(5) + "Year    :" + P_YEAR
? space(5) + "ISBN    :" + trim(ISBN)
if STATUS = "I"
? space(5) + "STATUS  : AVAILABLE"
else
? space(5) + "STATUS  : Checked Out"
endif
?
else
@ 6,1
? space(15) + " That Call number was not found."
? space(15) + " Please exit and retry with the "
? space(15) + " book Call_No help."
?
?
endif
wait

* erase the window
deactivate window STATBNO
release window STATBNO

* close databases
close database
return

*****
* Program Id      : LLMS4500
* Program        : REFISBN.PRG
* Author         : Park, Seong Seung
*****

```

```

* Date           : 24 Oct 1989
* Software       : dBASE IV
* Description    : This program searches a book by International Standard
*                : Book Number(ISBN). You are asked to enter book ISBN.
*                : This program reponses the information about the requested
*                : book.
*****

```

```

clear
set escape off

```

```

* open databases
select B
use BOOK order ISBN

```

```

* initialize variables
isbnnum = "      "
header = "BOOK INFORMATION"

```

```

* prompt the clerk for the ISBN number
done = .F.
do while .not. done

```

```

  clear
  @ 15,20 say "Enter: "
  @ 16,22 say "[ESC] to return to previous menu"
  @ 17,22 say "Enter ISBN Number: " get isbnnum picture "!!!!!!!!!!!!"
  read
  if readkey() = 12 .or. isbnnum = "      "
    isbnnum = "Escape"
    done = .T.
  endif
  isbnnum = upper(isbnnum)

```

```

if isbnnum = "H"
  clear
  search = " "
  allitems = space(3)
  linecount = 1

```

```

  * display prompt to clerk
  again = .T.
  do while again
    @ 15,20 say "ID Number Locator"
    @ 16,20 say "Type 'ALL' for a list of all books" get allitems picture "!!!"
    @ 17,20 say "Press [ESC] to exit help"
    read
    if readkey() = 12
      return
    endif

```

```

  * display a list of all books

```

```

do case
  case allitems = " "
    exit
  case allitems = "ALL"
    select B
    row = 7
    col = 5
    clear
    @ 3,15 say "          Book Listing "
    @ 5,6 say " ISBN          Title"
    @ 6,6 say " _____ "+";
    "
    go top
    linecount = 7
    do while .not. EOF(2)
      ? space(6) + ISBN + space(2) + TITLE
      skip
      linecount = linecount + 1
      if linecount >= 20
        wait
        clear
        @ 5,15 say " Book List Continued"
        @ 6,15 say " _____ "
        linecount = 7
      endif
    enddo
    ?
    wait
    clear
    allitems = " "
    go top
    again = .F.
  endcase
enddo
isbnnum = space(13)
else
  done = .T.
endif
enddo
clear
if isbnnum = "ESCAPE"
  close databases
  return
endif

* searches for the ISBN number
select B
goto top
on error exit
do while .T. .or. .not. bof()
  if isbnnum # upper(ISBN)

```

```

        skip + 1
    else
        exit
    endif
enddo
on error

* do the header
define window STATBNO from 3,8 to 20,73 double
activate window STATBNO
@ 0,19 to 2,40
@ 1,22 get header
?

* print the status if found
seek isbnnum
if found()
    ? space(5) + "ISBN      : " + trim(ISBN)
    ? space(5) + "Item_No  : " + ITEM_NO
    ? space(5) + "Author   : " + trim(AUTHOR_F) + " " + trim(AUTHOR_L)
    ? space(5) + "Title    : " + trim(TITLE)
    ? space(5) + "Edition  : " + trim(EDITION) + " Edition"
    ? space(5) + "Subject  : " + trim(SUBJECT)
    ? space(5) + "Publisher: " + trim(PUBLISHER)
    ? space(5) + "Year     : " + P_YEAR
    ? space(5) + "Call_No  : " + trim(CALL_NO)
    if STATUS = "I"
        ? space(5) + "STATUS  : AVAILABLE"
    else
        ? space(5) + "STATUS  : Checked Out"
    endif
    ?
else
    @ 6,1
    ? space(15) + " That ISBN number was not found."
    ? space(15) + " Please exit and retry with the "
    ? space(15) + " ISBN number help."
    ?
    ?
endif
wait

* erase the window
deactivate window STATBNO
release window STATBNO

* close databases
close database
return

```

```

*****
* Program Id      : LLMS4600
* Program        : REFSUBJ.PRG
* Author         : Park, Seong Seung
* Date           :26 Oct 1989
* Software       : dBASE IV
* Description    : This program searches a book by subject.
*                : You are asked to enter book subject. This program
*                : responses information about the requested book.
*****

```

```

clear
set escape off

```

```

* open databases
select B
use BOOK order ITEM_NO

```

```

* initialize variables
subjectname = space(20)
header = "BOOK INFORMATION"
booknum = space(6)

```

```

* prompt the clerk for the SUBJECT
clear
@ 15,20 say "Enter: "
@ 16,22 say "[ESC] to return to previous menu"
@ 17,22 say "Enter Subject : " get subjectname
read
if readkey() = 12 .or. subjectname = " "
    close databases
    return
endif

```

```

subjectname = upper(subjectname)
select B
clear
@ 3,15 say "          Book Listing "
@ 5,6 say " Item No          Title"
@ 6,6 say " _____"

```

```

go top
linecount = 7
do while .not. EOF(2)
    if trim(subjectname) =
        substr(upper(trim(SUBJECT)),1,len(trim(subjectname)))
        ? space(6) + ITEM_NO + space(3) + TITLE
    endif
    skip
    linecount = linecount + 1
    if linecount >= 18
        @ 21,15 say "Type Item_No or 'G' for next" get booknum picture "!XXXXX"
    endif
endif

```

```

    read
    clear
    if booknum # "G" .and. booknum # " "
        exit
    endif
    booknum = space(6)
    @ 5,15 say " Book List Continued"
    @ 6,15 say " _____ "
    linecount = 7
endif
enddo
if booknum = " "
    @ 21,15 say "Type Item_No" get booknum picture "!XXXXX"
    read
endif

* searches for the Item_No number
clear
select B
goto top
on error exit
do while .T. .or. .not. bof()
    if booknum # ITEM_NO
        skip + 1
    else
        exit
    endif
enddo
on error

* do the header
define window STATBNO from 3,8 to 20,73 double
activate window STATBNO
@ 0,19 to 2,40
@ 1,22 get header
?

* print the status if found
seek booknum
if found()
    ? space(5) + "Item_No :" + ITEM_NO
    ? space(5) + "Author  :" + trim(AUTHOR_F) + " " + trim(AUTHOR_L)
    ? space(5) + "Title   :" + trim(TITLE)
    ? space(5) + "Edition :" + trim(EDITION) + " Edition"
    ? space(5) + "Subject :" + trim(SUBJECT)
    ? space(5) + "Publisher:" + trim(PUBLISHER)
    ? space(5) + "Year    :" + P_YEAR
    ? space(5) + "Call_No :" + trim(CALL_NO)
    ? space(5) + "ISBN    :" + trim(ISBN)
    if STATUS = "I"
        ? space(5) + "STATUS  : AVAILABLE"
    endif
endif

```

```

else
    ? space(5) + "STATUS : Checked Out"
endif
?
else
    @ 6,1
    ? space(15) + " That Item number was not found."
    ? space(15) + " Please exit and retry with the "
    ? space(15) + " Item number help."
    ?
    ?
endif
wait

```

```

* erase the window
deactivate window STATBNO
release window STATBNO

```

```

* close databases
close database
return

```

```

*****
* Program      : BOOKDATA.FMT                      *
* Author       : Park, Seong Seung                 *
* Date         : 23 Oct 1989                       *
* Software     : dBASE IV                          *
* Description   : This program formats the file for new book data input. *
*****

```

```

clear
@ 0,0 say " Press [ESC] when done with this data"
@ 1,2 to 21,77 DOUBLE
@ 2,24 say "Library Loan Management System"
@ 3,26 say "Book Item Information Form"

@ 4,3 to 4,76 double
@ 5,3 say;
' Enter the information in the spaces provided. To move from one space to'
@ 6,3 say;
' another, press [TAB] or [ENTER]. When you have finished with the form,'
@ 7,3 say;
' press [ESC]. To record the information, press [A] to Add it to the data-'
@ 8,3 say;
' base. If you want to change the information before you have pressed [A],'
@ 9,3 say;
' press [E] to Edit it. After you have Added the information for all the'
@ 10,4 say 'books you have to record, press [Q] to Quit.'

@ 11,3 to 11,76 double
@ 12,4 say "All information is required —"

```



```

@ 13,4 say "Book Item number:"
@ 13,22 get book_item_no picture "XXXXXX"
@ 14,4 say "Book Title:"
@ 14,16 get book_title picture "!XXXXXXXXXXXXXXXXX"+;
"XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
@ 15,4 say "Author : FirstName" get book_at_f picture "!XXXXXXXXXXXXXXXXX"
@ 15,42 say "LastName" get book_at_l picture "!XXXXXXXXXXXXXXXXX"
@ 16,4 say "Publisher:" get book_publisher;
picture "!XXXXXXXXXXXXXXXXXXXXXXXXXXXX"
@ 16,42 say "Year:" get book_year picture "9999"
@ 17,4 say "Subject:" get book_subject picture "!XXXXXXXXXXXXXXXXXXXX"
@ 19,4 say "Call Number:" get book_call_no;
picture "XXXXXXXXXXXXXXXXXXXX"
@ 19,42 say "ISBN:" get book_isbn picture "XXXXXXXXXXXX"
@ 20,4 say "Edition:" get book_edition picture "9XXX"
@ 20,42 say "Page:" get book_bib_page picture "9999"

```

```

*****
* Program      : PTRNDATA.FMT *
* Author       : Park, Seong Seung *
* Date         : 24 Oct 1989 *
* Software     : dBASE IV *
* Description   : This program formats the file for patron data input. *
*****

```

```

clear
@ 1,0 to 21,79 double
@ 2,23 say "Library Loan Management System"
@ 3,30 say "Patron Data Form"
@ 5,1 to 5,78
@ 6,2 say;
" All information is required. To move from one space to the next, press"
@ 7,2 say;
"[ENTER] or [UP] or [DOWN]. When you have finished filling in the form,"
@ 8,2 say;
"press [ESC]. To return to the Patron Data Menu, leave the form blank"
@ 9,2 say "and press [ESC]."
@ 10,1 to 10,78
@ 11,2 say "Patron_Id number:"
@ 11,20 get patron_num picture "XXXXX"
@ 11,41 say "(once entered, CANNOT be changed)"
@ 13,2 say "First name:"
@ 13,14 get patron_fname picture "!XXXXXXXXXXXXXXXXX"
@ 13,41 say "Last name:"
@ 13,52 get patron_lname picture "!XXXXXXXXXXXXXXXXX"
@ 15,2 say "Department:"
@ 15,14 get patron_dept picture "!XXXXXXXXXXXXXXXXXXXXXXXXXXXX"
@ 15,41 say "Section:"
@ 15,50 get patron_sect picture "!XXXXXXXXXXXXXXXXXXXXXXXXXXXX"
@ 16,2 say "SMC Number:"

```

© 16,14 get patron_smc picture "9999"
© 16,41 say "Registration date:"
© 16,60 get regist_date
© 18,2 say "Street:"
© 18,10 get patron_street picture "!XXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
© 18,41 say "City:"
© 18,47 get patron_city picture "!XXXXXXXXXXXXXXXXXXXX"
© 19,2 say "State:"
© 19,9 get patron_state picture "!!"
© 19,22 say "Zip:"
© 19,27 get patron_zip picture "99999"
© 19,41 say "Phone:"
© 19,48 get patron_phone picture "(999)999-9999"

LIST OF REFERENCES

1. Ramez Elmasri, and Shamkant B. Navathe, *Fundamentals of Database Systems*, The Benjamin/Cummings Publishing Company, Inc., 1989.
2. Guy R. Lyle, *The Administration of the College Library*, 4th Edition, The H.W. Wilson Company, 1974.
3. James A. Larson, *Database Management*, IEEE Computer Society Press, 1987.
4. Fred R. McFadden, and Jeffrey A. Hoffer, *Database Management*, The Benjamin/Cummings Publishing Company, Inc., 1985.
5. Peter P. Chen, *Entity-Relationship Approach to Information Modeling and Analysis*, Elsevier Science Publishers B. V., 1983.
6. David M. Kroenke, and Kathleen A. Dolan, *Database Processing*, 3rd Edition, Science Research Associates, Inc., 1988.
7. Hammer M. and McLeod D., "Database Description with SDM: A Semantic Database Model", *ACM Transactions on Database Systems*, Vol. 6, No. 3, September 1981.
8. David Kroenke, *Database Processing*, 2nd Edition, Science Research Associate, Inc., 1983.
9. Dimitrios A. Elefsiniotis, *Development of a Personal Database System for Watch Scheduling on Hellenic Navy Ships*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1988.
10. Thomas W. Carlton, and David W. Solomon, *dBASE IV Applications Library*, 2nd Edition, Que Corporation, 1989.
11. Alan Simpson, *dBASE IV User's Desktop Companion*, SYBEX Inc., 1989.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
Cameron Station
Alexandria, VA 22304-6145 2
2. Library, Code 0142
Naval Postgraduate School
Monterey, CA 93943-5002 2
3. Professor Myung W. Suh, Code 54Su
Naval Postgraduate School
Monterey, CA 93943-5000 1
4. Professor Gary K. Poock, Code 55Pk
Naval Postgraduate School
Monterey, CA 93943-5000 1
5. Professor Dan C. Boger, Code 54Bo
Naval Postgraduate School
Monterey, CA 93943-5000 2
6. Major Yong Goo Hwang
SMC 2120 NPGS
Monterey, CA 93943 1
7. Captain Jae Doo Chung
SMC 1504 NPGS
Monterey, CA 93943 1
8. Major Myeong Hung Kang
SMC 2418 NPGS
Monterey, CA 93943 1
9. Captain Jung Hyun Park
SMC 1818 NPGS
Monterey, CA 93943 1
10. Captain Chong Soo Ryoo
SMC 2039 NPGS
Monterey, CA 93943 1
11. Captain Heung Taek Kim
SMC 1930 NPGS
Monterey, CA 93943 1

- | | | |
|-----|--|---|
| 12. | Captain Young je Jung
SMC 2240 NPGS
Monterey, CA 93943 | 1 |
| 12. | Captain Do Kyeong Ok
245-17, Sang-Dong, Nam-Gu,
Bucheon-City, Gyunggi-Do,
Republic of Korea, 422-030 | 1 |
| 13. | Captain In Sub Shin
518, Sanno-Ri, Kayagok-Myun,
Nonsan-Gun, Chung-Nam-Do,
Republic of Korea, 320-840 | 1 |
| 14. | Captain Il Joong Kim
Hanyang Apt 503, Dongbu-Dong,
Andong-City, Kyung-Pook-Do,
Republic of Korea, 760-050 | 1 |
| 15. | Library
Kyung-Pook National University,
Bokhyun-Dong, Buk-Gu, Taegu-City,
Republic of Korea, 702-020 | 1 |
| 16. | Library
Keum-Oh Institute of Technology,
180-1, Sinpyeong 1-Dong,
Kumi-City, Kyung-Pook-Do,
Republic of Korea, 641-000 | 1 |
| 17. | Library
P.O.Box 77, Gongneung-Dong,
Dobong-Gu, Seoul,
Republic of Korea, 132-240 | 1 |
| 18. | Army Central Library
Army Headquarter, Bunam-Ri,
Duma-Myun, Nonsan-Gun, Chung-Nam-Do,
Republic of Korea, 320-919 | 1 |
| 19. | Park, Seong Seung
625-113, 6-Tong 5-Ban,
Dobong-2-Dong, Dobong-Gu, Seoul,
Republic of Korea, 139-012 | 7 |