NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

NAVAL POSTGRADUATE SCHOOL
Monterey, California

DTIC
ELECTE
JUL 3 1985
B

THESIS

THE DEVELOPMENT OF A STANDARD DATABASE FOR
REPUBLIC OF KOREA ARMY'S LOGISTICS
SUPPORT SYSTEM

by

Lee, Hee Young
and
Song, Wha Dal

March 1985

Thesis Advisors: Carl Jones
                                    Norman R. Lyons

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A155 931	3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle) The Development of a Standard Database for Republic of Korea Army's Logistics Support System	5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1985
	6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s) Lee, Hee Young and Song, Wha Dal	8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943	12. REPORT DATE March 1985
	13. NUMBER OF PAGES 119

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

logistics database

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In order to effectively command and control the logistics of the
ROK Army, the commander must know the status of his resources
accurately and in a timely manner. The database processing can
increase productivity, enable work to be done more effectively,
and increase combat capability.
This thesis presents a sample database systems for inventory
status of the ROK Army 2nd Logistics Support Command with
relational model   Continued

DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE
1 JAN 73

S N 0102-LF-014-6601

1 SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT (Continued)

Database design is a two-phased process, and here are examined
both logical and physical database design processes. These
processes are an iterative process to get optimal design. Normal
forms can be applied to decrease inefficiency of the relational
database model in the system design process.
A sample database using dBASE II is implemented with IBM PC, and
is designed for the user who does not have computer experience.

Accession For

NTIS GRA&I	✔	
DTIC TAB	☐	
Unannounced	☐	
Justification		

Distribution/

Availability Codes

Dist	Avail and/or Special
A-1	

The Development of a Standard Database
for
Republic of Korea Army's Logistics Support System

by

Lee, Hee Young
Major, Republic of Korea Army
B.A., Korea Military Academy, 1974

and

Song, Wha Dal
Major, Republic of Korea Army
B.S., Korea Military Academy, 1976

Submitted in partial fulfillment of the
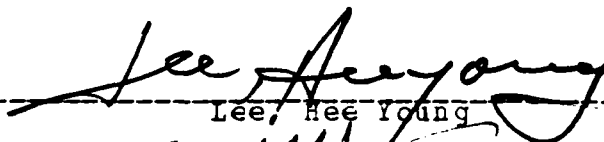requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1985

Authors: _____
Lee, Hee Young
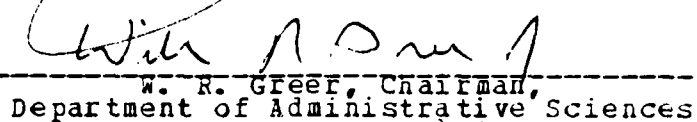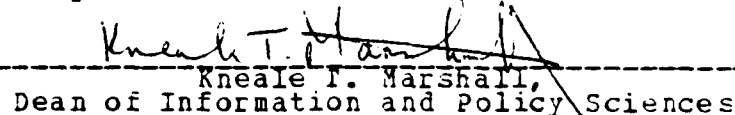
_____
Song, Wha Dal

Approved by: _____
Norman R. Lyons, Thesis Advisor

_____
Carl Jones, Co-Advisor

_____
W. R. Greer, Chairman,
Department of Administrative Sciences

_____
Kneale T. Marshall,
Dean of Information and Policy Sciences

3

ABSTRACT

In order to effectively command and control the logistics of the ROK army, the commander must know the status of his resources accurately and in a timely manner. The database processing can increase productivity, enable work to be done more effectively, and increase combat capability.

This thesis presents a sample database systems for inventory status of the ROK Army 2nd Logistics Support Command with relational model.

Database design is a two-phased process, and here are examined both logical and physical database design processes. These processes are an iterative process to get optimal design. Normal forms can be applied to decrease inefficiency of the relational database model in the system design process.

A sample database using dBASE II is implemented with IBM PC, and is designed for the user who does not have computer experience.

TABLE OF CONTENTS

6

LIST OF FIGURES

8

I. INTRODUCTION

A great deal of data and information is disseminated daily, and it betters everyone's lives. To keep track of this information, it is necessary to memorize it. It is very hard for everybody to remember all this data and information, so computers were invented. Using the computer, information systems can be constructed to gather, arrange and calculate all important data and information in magnetic disk or tape files.

Information systems can distribute all the necessary data to people who need it whenever they want it. This type of computer system is called a DATABASE system. It is an integrated collection of stored files that contain data used to operate an organization. Database systems also provide reliable information to users when they need it and maintain current available data.

From the 1970s, DB systems were considered an esoteric subject, of interest only to the largest corporations with the largest computers. Today, DB systems are becoming an information systems standard. DB processing has grown significantly in the computer science area and also in management of certain organizations.

An important consideration in database development is to store data in such a way that it can be used for a wide variety of applications and can be changed quickly and easily. To achieve the flexibility of data usage, three aspects of DB design and implementation are important. First, the data should be independent of each other and functionally dependent on the key value. Second, it should be possible to interrogate for user's requirements using application programs or the DBMS itself. Third, these data

9

items should provide useful information for decision makers to analyze, to investigation, to plan and to manage in a certain organization.

It is very difficult to develop DB system which perform in an optimal fashion. There are many different ways in which data can be structured and each has its own advantages and disadvantages. Different users want to use different data/information. It is hardly possible to satisfy all of the users with one type of data organization.

The normal form concepts of relational database will be used to develop an intelligence database, because the relational database management system supports independence better then other models and is easier to implement.

Chapter II addresses the background, which relates to the database system development for the Korean army's logistics support system requirement s. Chapter III defines fundamental concepts of database, and include a general overview of a DB system and its protection. Chapter IV discusses an introduction to database design, both logical and physical, and describes database models and selection that are useful for Korean army logistics support systems. Chapter V presents a relational database design for the Korean army 2nd logistics support command, which includes relational normal forms and the characteristics of the relational database. Chapter VI addresses the implementation of a logistics system in the Korean army using the relational DBMS product DBASE II. Finally, Chapter VII presents conclusion and recommendations based on the research presented in the thesis.

II. BACKGROUND

A. OVERVIEW

The republic of Korea(ROK) army uses the general staff
system of the U.S.A field Armies, corps, and divisions,
namely, G-1, personnel; G-2, intelligence; G-3, operations
and training; and G-4, logistics.

The Korean army is the largest organization in Korea. In
national security, the position of the Korean army is very
important. It stands face to face with communist north Korea
on the 155 mile long DMZ. In order to strengthen the war
potential of the Korean army, it is imperative tnat the
management of the logistics support system be performed very
efficiently.

However, manually managing all army logistics system is
a very tedious, complex, and time-consuming job. The army
logistics system deals with approximately 200,000 items. In
order to reduce time, overhead and the national defense
expenditure, and increase the war power, the army needs a
computerized management information system for army logis-
tics system management. Top-level Korean army officers are
very interested in a computer database system to maintain
the status of all items handled by the army logistics
system.

The Korean army installed its first computer system in
the data processing center(DPC) of the logistics command in
1973. The center, however, did not become fully operational
until 1974.

11

The DBMS also has features to provide security over data; the tools provided ensure that only authorized data are accessed. Also, the DBMS controls concurrent processing and includes features to provide backup and recovery.

The final type of program involved in database processing is the operating system. This set of programs controls the computer's resources. The DBMS sends requests for input/output services to operating system. All programs are controled by the operating system.

3. Data

According to standard usage in the computer industry, BITS are grouped into BYTES or CHARACTERS, CHARACTERS are grouped into FIELDS, and FIELDS are grouped into RECORDS. A collection of records is call a FILE. A DATABASE is more than a collection of files; It is a collection of integrated files. Another way of saying this is that a database is a collection of files and relationships among records in those files.

Database records can be accessed sequentially within a file, randomly by value of field, or by relationship to other records.

A key is a field that is used to identify a record. For database processing, key can be unique or nonunique.

In a database system a variety of views of the data are defined. One is called the "schema" or "conceptual view". This is complete logical view of data. The term "logical" means the data as it would be presented to the human. The schema describes all of the data in the database.

Another type of view called a "subschema", or external view defines a subset of the schema to be seen by a given application program or user.

A third view of the data ia called the "internal view", or sometimes, the physical view. This is the form of

25

The utility programs are provided by either the DBMS or the hardware vendor. Theses programs provide a wide variety of services. Query/update utilities provide generalized retrieval and update of the database.
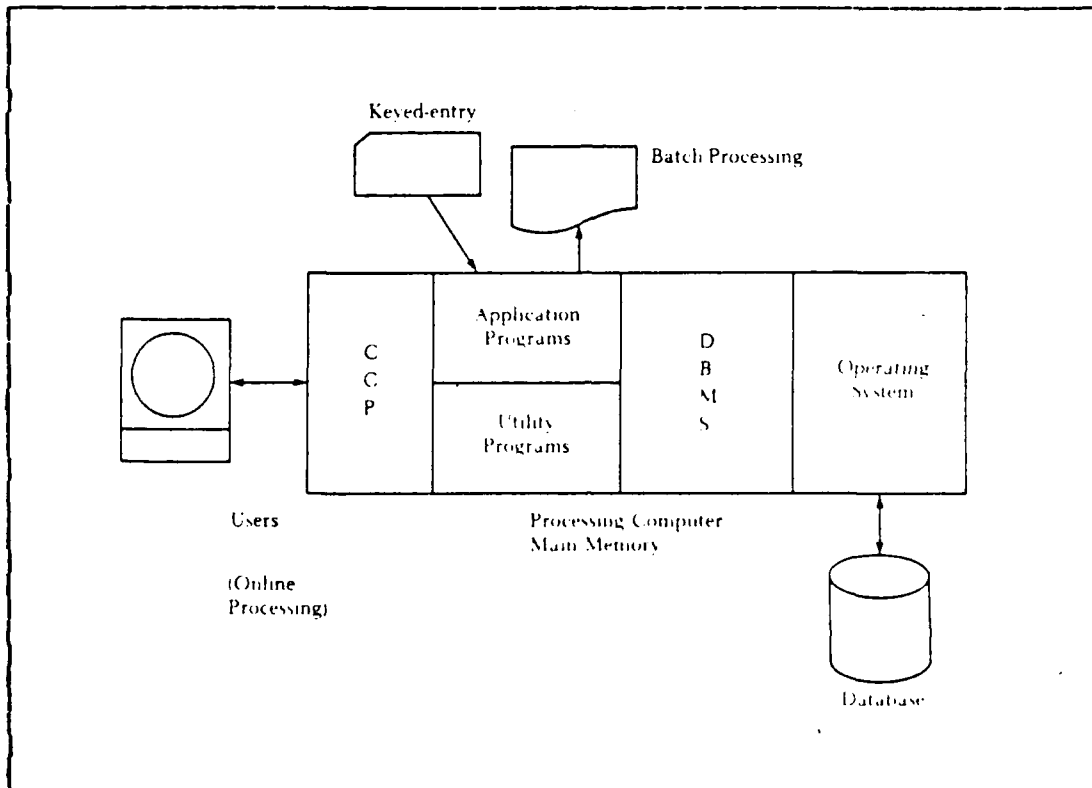
Figure 3.3   Programs Involved in Typical Database Processing

For normal   processing,   the DBMS receives   data and stores it for subsequent processing.   This system acts as a sophisticated data   librarian.   The DBMS   allows application programs and utilities a wide variety of access storages. It also enables these   programs to have different   views of the same data so that applications can use data in a format that is familiar and useful to them.

processing can be performed simultaneously with applications
processing.
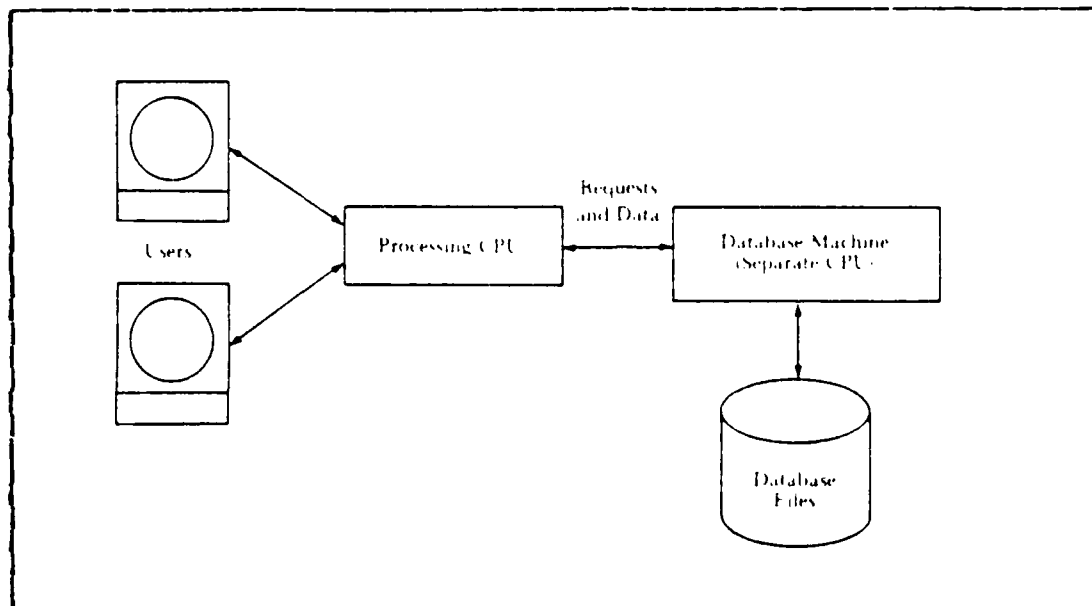
Figure 3.2   Schematic of Processing with Database Machine

2. Programs

There are several types of programs which are used
in database processing systems. Figure 3.3 shows the approx-
imate relationships of the major types. Online processing
requests or transactions are provided by users at terminals.
The requests are sent to the processing computer over commu-
nications lines.

The requests are received and routed by the
Communications Control Program (CCP). It provides communica-
tion error checking and correction, coordinates terminal
activity, routes messages to the correct next destination,
formats messages for various types of terminal equipment,
and performs other communication-oriented tasks.

23

are represented, what physical sequence the stored records are in, etc.

The conceptual and internal mapping defines the correspondence between the data model and the stored database; it specifies how conceptual records and fields map into their stored counterparts. If the structure of the stored database is changed, that is, if a change is made to the storage structure definition, The conceptual and internal mapping must be changed accordingly, so that the conceptual schema may remain invariant.

D. COMPONENTS OF A BUSINESS DATABASE SYSTEM

A business database system is a collection of five components that interact to satisfy business needs. The five components are hardware, programs, data, people, and procedures [Ref. 3].

1. Hardware

A database system does not require a special type of hardware. And it can be used in mainframes, minis and micros. Database processing, however, dose involve special programs and overhead data. Thus database applications often require more more hardware: more memory, a faster CPU, and more direct access storage.

Database machines are special-purpose computers that perform database processing functions. Also, Hsiao defines a database machine as "specialized hardware supporting basic DBMS functions found in most contemporary software database management systems" [Ref. 4]. As shown in figure3.2, the computer processing the application program sends requests for service and data over a channel to the database machine. The machine processes the requests and sends results, data, or messages back to the main computer. thus database

22
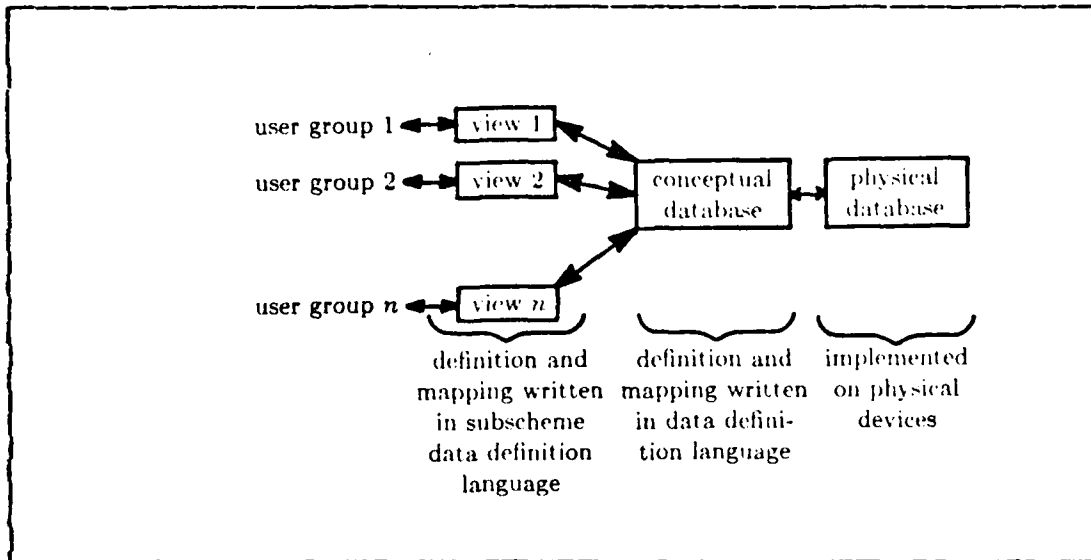
Figure 3.1   The Three Levels of the Architecture

additicn, there must be a definition of the mapping between
the external schema and the undering conceptual schema.

The conceptual model is a representation of the entire
information content of the database, again in a form that is
somewhat abstract in comparison with the way in which the
data is physically stored. The conceptual model is defined
by means of the conceptual schema, which includes defini-
tions of each of the various types of conceptual records.
This model is a view of the total database content, and the
conceptual schema is a definition of this view. The defini-
tion in the conceptual schema is intended to include a great
many additional features, such as the authorization checks
and validation procedures.

The internal model is a very low-level representation of
the entire database; It consists of multiple occurrences of
multiple types of internal records. The internal model is
described by means of the internal schema, which not only
defines the various types of stored record but also fields

21

environment is more complex for personnel who must manage
the system and data. Large amounts of data in many different
formats can be interrelated in the database. Third, backup
and recovery are more difficult, because of increased
complexity and because databases are often processed by
several users concurrently. Fourth, the system is more vuln-
erable to failure, because all data are centralized and
under one system. Another disadvantage of DBMS is likely to
be slower and more expensive than a file system "tuned" to a
particular application.

C. AN ARCHITECTURE FOR A DATABASE SYSTEM

The aim of presenting architecture is to provide a
framework which is useful for describing general concepts
and the structure of individual systems. But every database
system can not be neatly matched to this particular
framework.

The architecture is divided into three general levels:
internal, conceptual and external in figure 3.1 [Ref. 2,5].
Broadly speaking, the internal is the one closest to phys-
ical storage, the one concerned with the way in which the
data are actually stored; the external level is the one
concerned with the way in which the data are viewed by indi-
vidual users; and the conceptual level is a "level of indi-
rection" between the other two.

Next, the various components of the system will be exam-
ined. The users are either application programmers or remote
terminal users of any degree of sophistication. Each user
has a language at his disposal. It will be a conventional
programming language, such as COBOL, PL/1, PASCAL, etc.

Each external model is defined by means of an external
schema, which consists of descriptions of each the various
types of external records in that external model. In

20

sharp contrast to the situation that prevails in most enterprises today, where typically each application has its own private files so that the operational data is widely dispersed, and there is little or no attempt to control it in a systematic way [Ref. 2].

The database processing have many advantages and also many disadvantages [Ref. 3]. First, the advantage of database processing enables more information to be produced from a given amount of data. Second, the elimination or reduction of data duplication saves file space, and to some extent, can reduce processing requirements. The most serious problem of data duplication is that it can lead to a lack of data integrity. A common result of this is conflicting results. Third, creation of program/data independence dose not create problems when a file is changed. It means that the application concerned do not depend on any one particular storage structure and access storatge. The fourth advantage is better data management. When data are centralized in database, one department can specialize in the maintenance of data. That department can specify data standards and ensure that all data adhere to the standards. When someone has a data requirement, he or she can contact one department instead of many file maintenance groups. There is only one DBMS processing a shared database, and improvements made to the database or to the DBMS will benefit many users. The other advantages of database processings allow query languages for easy "one-shot" programs and make it easier to retrieve sophisticated information in a DBMS environment.

A major disadvantage of database processing is that it can be expensive. The DBMS may occupy much main memory, and software purchase costs are high. Once the database is implemented, operating and administrative costs for some systems will be higher. Also, more sophisticated computer personnel are required to operate a DBMS. Second, a DBMS

19

III. BASIC CONCEPT OF DATABASE

A. WHAT IS A DATABASE?

A much-publicized but impracticable idea of a database says that a corporation keeps all its processable items of data in a large storage in which diversity of data users can be accessed. The storage in which all the data are kept may be in one location or multiple locations, the latter possibly interconnected by telecommunications. Programs for a variety of applications have access to the data.

A database may be defined as a collection of interrelated data stored together without harmful or unnecessary redundancy to serve multiple applications; the data are stored so that they are independent of programs which use the data; a common and controlled approach is used in adding new data and in modifying and retrieving existing data within the data base. The data is structured so as to provide a foundation for future application development. One system is said to contain a collection of data bases that are entirely separate in structure [Ref. 1].

Also according to [Ref. 2], the definition of database is a collection of stored operational data used by the application systems of some particular enterprise (e.g. manufacturing companies, banks, hospitals, etc.). And database systems are nothing more than a computer-based record keeping system: that is, a system whose overall purpose is to record and maintain information.

B. ADVANTAGES AND DISADVANTAGE

A database is needed to provide the enterprise with centralized control of its operational data. This is in

management, storage management, Army equipment management, automatic return management, etc.

This ADP system generates many different reports which are made by result from running computer system. These reports are produced periodically as required for higher officers who work in the Army logistics field.

these reports show by the title, the generation time, the number of copies, contents of each report, their use, etc. These are also designed to provide up-to-date accurate status data for selected items or units.

.

Recently, higher managers have recognized the need for the standardization of hardware and the unification of application softwares One department, software developing department, that directly manages to develop application systems and programs was formed.

D. ADP SUPPORT

The ADPC(Automatic Data Processing Center) within the logistics structure provides significant support. In order to effectively command and control any operations, the commander must have adequate visibility.

The use of automatic data processing(ADP) systems has significantly increased the commander's visibility and has had an effect on logistics operations.

The ADPC dedicated to the logistics operations supports its own internal functions such as stock controls within its area of responsibility and a routine report for higher command.

In order to produce many different reports which higher level managers need, this system uses file systems which include several files such as all item's master file, storage by depot file, due-in file, due-out file, fund resource file, fund ceiling control file, material ceiling control file, demand file,item's location file, OST file, Army equipment file, etc.

Most are basic files among all files. Some are transaction files or sort files which are a relatively temporary files containing data about transactions and sorting of working activities.

From these files, army logistics ADP system controls asset management that includes requisition and issue, due-in, due-out management, report and documentations management, fund and material management, requirement

16

must have a capability to provide reliable information with efficient processing. this is complicated by the fact that the application systems use several different file system.

The problem of the file system are as follows [Ref. 15],

First, there is a high level of redundancy. There are several of the same kind of data items among personnel system, payroll system, PX system, inventory system, military medical system, etc.

These common data items are updated independently in each file system. It is very hard to maintain the accuracy of a common data item on different file systems. Furthermore, the number of files for application will be more and more.

Second, the file systems are inflexible requests for information from a wide range of users are impossible to answer within given time. Even though the file systems contain data items for producing information to be provided, information can not be provided relating to those data items. The data can not be processed without reconstruction. Although millions have been paid for computer system, the information can not be obtained when it is needed.

Third, it can be expensive to make changes to a file system. According to the requests of users, a file system can be changed or modified. Sometimes the modifications are difficult because the applications were not adequately documented for other programs. As time goes on, the problem becomes worse because more programs are created or modified. And, whenever a file is changed, programs for that file system have to be changed or modified. Additionally, individually developed file systems and non-standardized hardware systems do not help to achieve data communications with each other.

The army logistics computer system is the largest system among the different types of computer system. The logistics computer system, can be divided hierarchically following different level computer system: department of computer system control in army headquarters, logistics information system center in logistics command, and data processing center in logistics support command which is located three different place to support units which are in same area.

Further down at the division level including corps, management of the logistics operations is accomplished monitoring the operational readiness of weapon system.

They use several languages, COBOL, Assembly language, and FORTRAN. 53% of total applications software is COBOL, 44% is Assembly language, and 3% is FORTRAN.

Nowadays, Assembly language tends not to be used to program. The percentage of COBOL will increase. They did not introduce more advanced higher level languages like PASCAL and Database languages.

Today computer hardware system consists of IBM370, UNIVAC90/30 and 1100 series. These computers are not on-line systems, but rather run batch jobs.

Applications systems are operated daily, weekly, monthly, and yearly depending upon the different reports. The files of the applications consist of indexed sequential access method(ISAM), sequential access method(SAM) or sequentially fixed-length records.

At present, many files of records are used in ROK army without database techniques. These files contain limited data items that personnel managers require. Several file systems provide information to be used for doing logistics management by spooling, time sharing, and virtual techniques.

In order to provide logistics managers who want to use information as soon as possible, ROK army logistics systems

Intermediate echelon provides the major interface between the wholesale and direct support/user echelon. It includes units in the field which provide general support supply, maintenance, transportation, facilities and services.

Direct support/user echelon includes fields units which provide direct support supply, maintenance, transportation and services. Users include the combat, combat support, and combat service support units utilizing the services and equipment which are the responsibilities of the logisticians.

C. CURRENT LOGISTICS SITUATION IN KOREA

The first computer introduced in Korea was the IBM 360/40 (64 KB) which came from the U.S in March 1967. Its purpose was to survey the entire population of Korea. The Korean army installed its first computer system in 1972 to organize the military personnel system. Next year, another computer was installed for the logistics system that were mentioned before.

The Korean army used the computer relatively early. Several computer centers were installed by the ROK army. There are several types of computer centers. The type of computer center is determined by the purpose of use - education, personnel, logistics, intelligence, finance and national security, etc. All the computer centers are directly controlled by the staff of ROK army headquarters. There is one integrated software development center which is located in headquarter of ROK army.

Each of the computer centers has different hardware systems, and applications with file systems have been individually designed, developed and operated by the different operating systems.

B. LOGISTICS STRUCTURE

The primary mission of logistics is to insure the operation of weapons on the battlefield. Logistics encompasses a broad spectrum of functions and responsibilities which are required in order that the ultimate objective can be achieved.

Basically, logistics can be described as an effort to develop and maintain maximum combat power through the support of weapon systems.

Just as the army itself is a composite defense system, the system which keeps it supplied and operational is a composite of material, .personnel and facilities, processes and organizations, and different levels and varieties of activities, all in motion together and all merging in the common and basic objective of meeting the requirements of the forces.

In considering how to manage for army logistics, five categories can be listed, These are facilities management, finance management, material supply management, service management, and personnel management. Logistics usually deal with material supply management that includes the following principal functions : requirement, procurement(acquisition), storage and distribution, maintenance while in storage, and salvage of supplies including the determination of quantities of supplies.

There are three major echelons of logistics support which are determined by types of work done at each echelon.
   * Wholesale echelon
   * Intermediate echelon
   * Direct support/user echelon

Wholesale echelon includes depots, maintenance points, plants and factories associated with special army activities retained under army headquarters.

12

the data as it appears to a particular processing computer.
It describes how data is physically arranged and how it is
allocated to files.

4. Peoyle

Clientele are the people for whom the system is
developed. The clientele of an airline reservation system
are the people who take flights. The clientele of a payroll
system are employees. Clientele do not usually have an
active role in database system development or use.

Users are people who employ the system to satisfy a
business need. The users of an airline reservation system
are the clerks, the users of the payroll system are payroll
administrators, clerks, and business managers.

Operations personnel run the computer and associated
equipment. Typically, the operations department includes
machine operators, data control personnel, and data entry
people.

Systems development personnel design and implement
the database system. The determine requirements, specify
alternatives, design the five components of the system. and
message systems implementation. The design of the database
structure or schema is an important function of these
people.

The final category of people in database applica-
tions is database administration (DBA) personnel. A database
is a shared resource. The function of the DBA staff is to
serve as a protector of the database and as a focal point
for resolving user's conflicts. The DBA should be a repre-
sentative of the community as a whole, and not of any
particular user or group of users.

5. Procedures

Both users and the operations staff need documented procedures for normal conditions. The users need to know how to sign on the system, how to use the terminals, how to provide data, and so fourth. They also need procedures that ensure they do not interfere with one another.

File systems fail at some point, and when a database system fails, both users and operations personnel need procedures describing what to do. These procedures are especially important for database processing because so many applications are dependent on the database.

Database management procedures are needed for DBA and others because every business is a dynamic activity, and business needs will change.

E. DATABASE PROTECTION

1. Security

The subject of database security, the protection of the database against unauthorized use, has many different aspects and approaches. First, it is necessary to protect against both undesired modification and/or destruction of data and against unauthorized reading of data. Three techniques are described below [Ref. 5] :

a. User identification --- The most common schema to identify users is a password known only to to the system and the individual.

b. Physical protection --- A high security system needs better identification that a password, such as personal recognition of the user by a guard.

c. Maintenance and transmittal of rights --- The system needs to maintain a list of rights enjoyed by each user on each protected portion of the database.

27

2. Integrity Preservation

The term "integrity" is used in database contexts
with the meaning of accuracy, correctness, or validity
[Ref. 6].   This aspect concerns nonmalicious errors and
their prevention. The problem of integrity is the problem of
ensuring that the data in the database is accurate.   Invalid
updates may be caused by errors in data entry, by mistakes
on the part of the operator or the application programmer,
by system failures, even by deliberate falsification. The
DBMS can help detect some programming bugs, such as a proce-
dure that inserts a record with the same values in the key
fields as a record that already exists in the database.

IV. INTRODUCTION TO DATABASE DESIGN

A. INTRODUCTION

A database is the interface between people and machines. The nature of these two components is utterly different. People are imprecise and intuitive, and their thinking is fuzzy. Machines are precise and predictable, and their processing is exact. The difficulty is to develop a database design which meets the needs of the people who will use it.

Database design is both art and science. Dealing with people, understanding what they want today, predicting what they will want tomorrow, differentiating between individual needs and community needs, and making appropriate design tradeoffs are artistic tasks. To accomplish these tasks, there are principles and tools, but these must be used in conjunction with intuition and guided by experience.

Database design is a two-phased process [Ref. 3]. First, one examines the user's requirements and build a conceptual database structure that is a model of the organization. This phase of database design is often called "logoical database design". Once the logical database design is completed, this design is formulated in terms of a particular DBMS. Usually, compromises must be made. For example, the DBMS may not be able to express relationships precisely as the users see them. The process of formulating the logical design in terms of DBMS facilities is called "physical database design".

This chapter will introduce the two-phased process of database design. And then it will survey important design tools called database models.

29

B. LOGICAL DATABASE DESIGN

Database design is an intuitive and artistic process. There is no algorithm for it. Typically, database design is an iterative process; during each iteration, the goal is to get closer to an acceptable design. Thus a design will be developed and then reviewed. Defects in the design will be identified, and the design will be done. This process is repeated until the development team and users can find no major defects.

Figure 4.1 illustrates the flow of work in a typical database design project. User requirements are studied and a logical database design is developed. The preliminary design of database processing programs is produced. Next, the logical database and the preliminary program designs are used to develop the physical database design and and the detailed program specifications. Finally, both of these are input to the implementation phase of the project.

1. Outputs

A logical database design specifies the logical format of the database. The RECORDS to be maintained, their contents, and RELATIONSHIPS among those records are specified. Industry uses various terms for this design. It is called the schema, the conceptual schema, the logical schema, and This thesis will use the term logical schema.

   a. Logical database records --- To specify logical records, the designer must determine the level of detail of the database model. If the model highly aggregated and generalized, there will be few records. If the model is detailed, there will be many records. The database designer must examine the requirements to determine how coarse or how fine the database model should be.

30
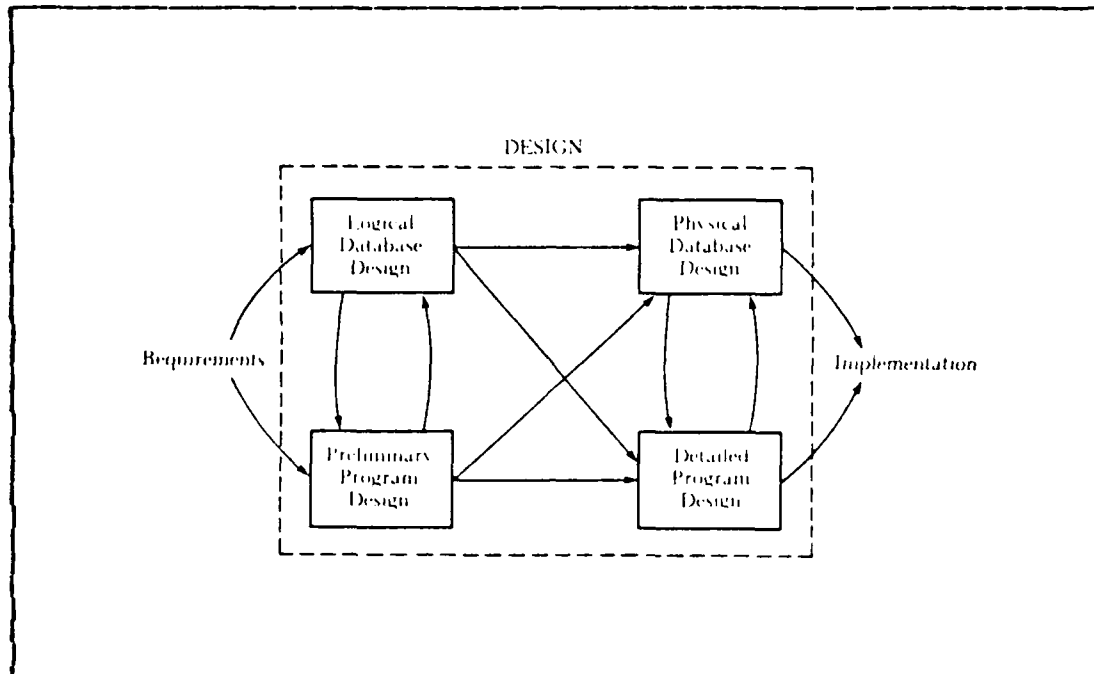
Figure 4.1   Database and Program Design Flow

The contents of these records are specified during logical design. Names of fields and their format must be determined. The designer must specify how much data will be maintained. Also, the format of data items is specified.

As the requirements are evaluated and the design progresses, CONSTRAINTS on data items will be identified. These constraints are limitations on the values that database data can have. Three types of constraints are common. FIELD constraints limit the values that a given data item can have. INTRARECORD constraints limit values between fields within a given record. INTERRECORD constraints limit values between fields in different records.

b. Logical database design relationships --- The essence of database is the representation of record

relationships. These relationships are specified during the logical design. The designer has to (a) determine one-to-one, one-to-many, and many-to-many relationships, (b) study the application environment, (c) examine the requirements, and (d) identify necessary relationships.

2. Inputs

The inputs to logical database design are the system requirements and the project plan. Requirements are determined by interviews with users, and that must be approved by both users and management. The project plan describes the system environment, the development plan, and constraints and limitations on the system design.

The requirement can be expressed in the form of data flow diagrams, policy statements, and the data dictionary. Contents of the data dictionary can be transformed into the logical and user views. Policy statements can be used to develop the descriptions of logical database processing. The requirements can be used to verify the completeness of the logical design.

3. Procedures for Logical Database Design

Many techniques have been defined for logical database design. Some are completely intuitive and others involve specific procedures for processing the data dictionary. The major steps in the logical design process are described below.

a. Identify data to be stored --- First, the data dictionary is processed, and that which is to be stored is identified and segregated. This is necessary because the data dictionary will contain descriptions of reports, and will screens, and input documents that will not be part of the database.

32

b. Consolidate and clarify data names --- One task is to identify synonyms, to decide on standard names for synonyms, and to record aliases (when synonyms cannot be eliminated, they are recorded as alternate names, or aliases).

Another task related to terminology is to ensure that data items having the same name are truly the same. If not, unique data item names will need to be developed.

c. Develop the logical schema --- This is developed by defining records and relationships. Records are defined by determining the data items they will contain. The design team examines the data flow diagrams and data dictionary, applies intuition to the business of setting up the new system, and determines that certain records will need to exist.

Some of these files may need to be combined, and others may need to be separated. Another problem regarding record definition concerns implied data. A data item is implied when it is needed to meet a requirement.

The second step in developing the logical schema is to determine relationships among database records. We want to model the way the users see the relationships. Generally, relationships are identified intuitively. At this point, the design team must discriminate between theoretical and useful relationships. A theoretical relationships can exist logically, but may never be needed in practice.

d. Define processing --- The requirements are examined to determine how the database should be manipulated to produce required results. The processing definitions can be developed in several ways. One method is to describe transactions and data to be modified. Another method is to develop structure charts of the programs that will access the database.

33

e. Design review -- The final stage of logical database design is a review. The logical schema and user views are examined in light of the requirements and program descriptions. Every attempt is made to identify omissions, unworkable aspects, or other flaws in the design. Typically, a panel of independent data processing people is convened for this review. Documentation of the logical schema, user views and program description is examined by the panel and oral presentations are evaluated. The purpose of the review is to identify flaws, not to solve them.

C. PHYSICAL DATABASE DESIGN

The second stage of database design -physical design is a stage of transformation. The logical schema is transformed into the particular data constructs that are available with the DBMS to be used. Whereas the logical design is DBMS-independent, the physical design is very much DBMS-dependent.

Detailed specifications of the database structure are produced. These specifications will be used during the database implementation to write source statements that define the database structure to the DBMS. These statements will be compiled by the DBMS and the object form of the database structure will be stored within the database. See Figure 4.2.

1. Output of Physical Database Design

Specific constructs vary widely from one DBMS to another. In general, two major specifications are produced. First, the physical specification of the logical schema is defined. This specification is called the PHYSICAL SCHEMA. This schema is a transformation of the logical schema into

34
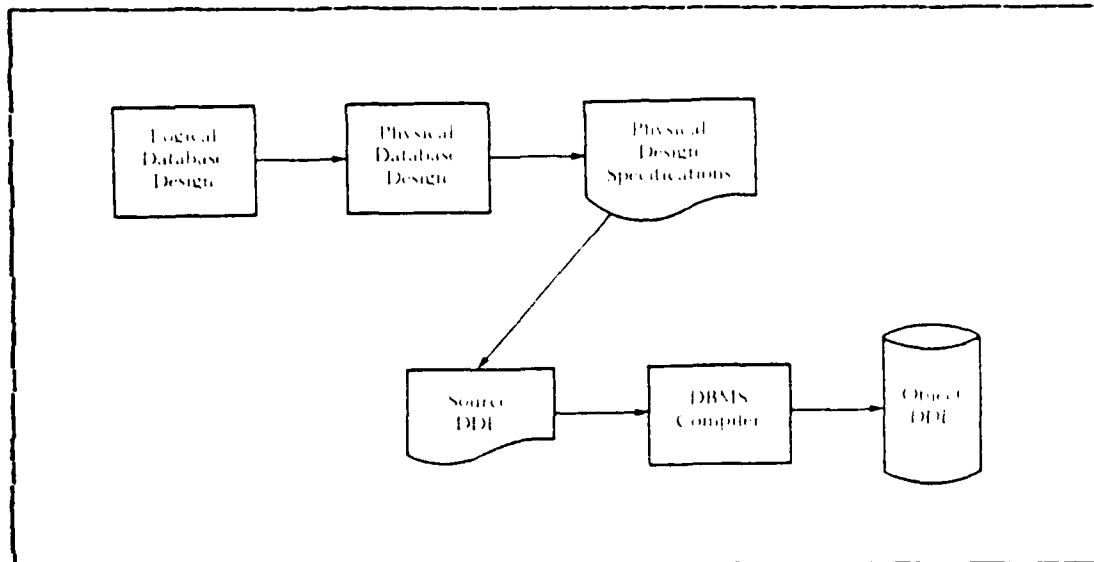
Figure 4.2   Role of Physical Design

the data modeling constructs available with the DBMS to be
used. Second, user views are defined.

a. Physical schema --- Figure 4.3a lists generic items
that are defined in a physical schema design.   The
contents of each record must be defined, and the name and
format of each field of each record specified.
Constraints from the logical database design are trans-
formed into criteria for field descriptions.   Keys of
database records need to be identified, and overhead
structures for supporting the keys defined.

Record relaticnships are also defined in the physical
design.   Limitations in the DBMS may necessitate that
record relationships be changed from what the user
wanted.   A   many-to-many   relationship may   need   to be
changed to a simple network.

b. User view --- Most users will need to view only a
portion of the database, the logical database design must
specify which user groups will view which portions of
database.

35

User views are generally a subset of the schema. Records or relationships may be omitted from a view; field may be omitted or rearranged. Also, the names of records, fields, or relationships may be changed. This flexibility allow users to employ terminology that is familiar and useful to them. Figure 4.3b lists items to be defined for user views during the physical design.

```
Name of physical schema
Names of records
Format of records
Names of fields
Format of fieds            Names of user view
Constraints                Names of records
Names of keys                (aliases)
Supporting overhead data   Format of records
          structures       Names(aliases) of
Format of record                        fields
          relationships    Format of fields
========================   ==================
a. Contents of physical    b. Contents of user
          schema                          views
```

Figure 4.3   Results of Physical Database Design

2.   Inputs to Physical Database Design

The inputs to the physical database design are the outputs of the logical database design, the system require-ments, and the preliminary design of programs. These were already described in the previous section.

3.   Physical Database Design Process

This is produced by transforming the logical design into a physical design. The specific outputs vary from one DBMS to another. It is impossible to describe this process, other than very generally, without first discussing the

36

physical database design specific DBMS features. The next
chapter will contain further discussion.

D. DATABASE MODELS

A database model is a vocabulary for describing the
structure and processing of a database [Ref. 3]. Database
models are an important database design tool. They can use
both logical and physical database design much as flowcharts
or pseudocode are used for program design. And database
models are used to categorize DBMS products. This section,
discusses the components of database models, the three
commercial database models, and survey of six important
models.

1. Components of Database Model

Database models have two major components. The Data
Definition Language(DDL) is a vocabulary for defining the
structure of the database. The DDL must include terms of
defining records, fields, keys and relationships. Also, the
DDL should provide a facility for expressing a variety of
user views. Ideally, the model will also provide a method
for expressing database constraints.

Data Manipulation Language(DML) is the second compo-
nent of database model. The DML is a vocabulary for
describing the processing of the database. Two types of DML
exist. Procedual DML is a language for describing actions to
be performed on the database. procedual DML obtains a
desired result by specifying operations to be performed. For
procedual DML, facilities are needed to define the data to
be operated on and to express the actions to be taken. Both
data items and relationships can be accessed or modified.

Nonprocedual DML is a language for describing the
data that is wanted without describing how to obtain it. The

user simply states what is wanted, not how to get the results. The DBMS is given the job of determining how to get the result. Nonprocedual DML is descriptive, not prescriptive.

2. Three Commercial Database Models

Database systems can be conveniently categorized according to several approach. The best know approaches are the relational, the hierarchical, and the network approach models. These have been used in the great bulk of database systems.

A model is hierarchical if its only data structure is a hierarchy (tree). With this model, all networks must first be decomposed to trees before they can be represented. And data are represented as a set of nested one-to-many and one-to-one relationships. This model is a special case of network model. So, many-to-many relationships are not directly supported and there is data redundancy.

A model is network if its data structures are both trees and simple networks. This model represents data as a set of record types and pairwise relationships between record types. Only complex networks need to be decomposed before they are represented.

The distinction between these two models has become unimportant. The hierarchical data model has become too narrow and the network data model to broad [Ref. 3]. The relational model will be described next section.

3. Overview of Prominent Database Models for Design

Figure 4.4 portrays six common and useful database models. The models are arranged on a continuum. Models on the left-hand side of this figure tend to be oriented to humans and human meaning, whereas those on the right-hand side are more oriented toward machines and machine specifications [Ref. 3].
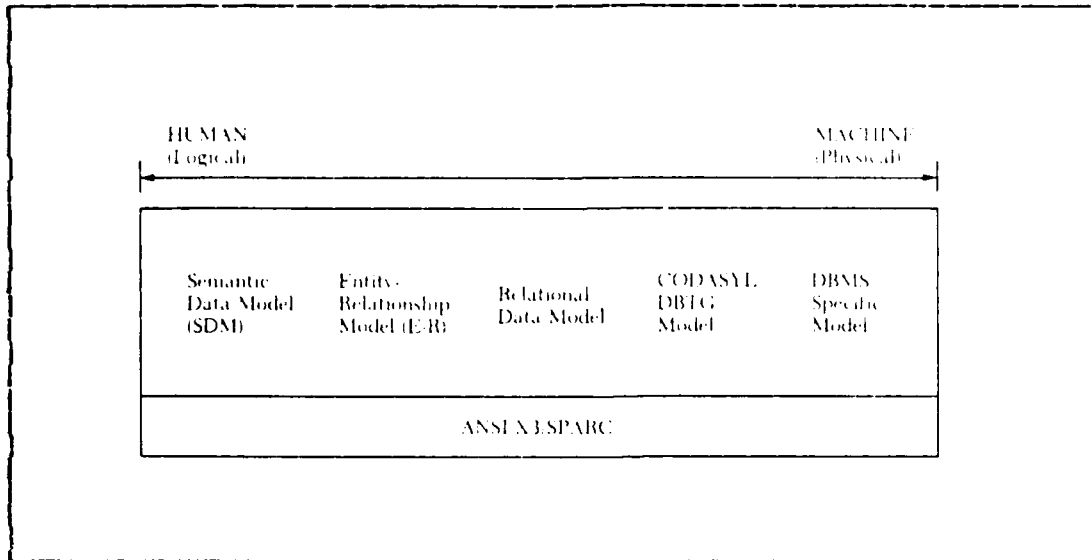
Figure 4.4   Relationship of Six Important Data Models

a.   Relational data model --- The relational database
model is near the midpoint of the human/machine continuum
in figure 4.4, because it has both logical and physical
characteristics. The relational model is logical in that
data are represented in a format familiar to humans; the
relational model is unconcerned with how the data are
represented in computer files. On the other hand, this is
more physical than SDM (semantic data model) or the
E_R(entity relationship)model.   Database that have been
designed according to the relational need not be trans-
formed into some other format before implementation. Thus
the relational model can be used for both logical and
physical database design.

A relation is simply a flat file.   The rows of the
relation are the file records. Rows are sometimes called
tuples of the relation. The field of the relation (in the
columns) are sometimes called the attributes of the rela-
tion. The significance of the relational model is not

39

a. various reporting facilities such as cross-reference reports, changes effecting reports, error-reports, etc;

b. various retrieval capabilities such as keywording, indexing, and online or batch querying;

c. common language to control, retrieve and update the data dictionary;

d. validation and redundancy - checking capabilities

e. security safeguards to control access to the data dictionary

f. data description generation

The examples of data dictionary for 2nd Logistics Support Command are shown on Appendix B.

D. RELATIONAL NORMAL FORMS

1. Anomalies

Some database design are better than others. A design that meets the user's needs is better than one that does not, but there are other criteria as well. Normal forms are (a) rules for assigning fields to files(relations) in a relational DBMS, (b) guidelines to prevent users from trying to place data together that does not belong together, (c) and are useful guides even if one is not using a DBMS.

With some relations changing data can have unexpected consequences. These consequences are called "modification" anomalies and are not desirable. When a fact is deleted, facts about two entities with one deletion are lost. This characteristic is called a "deletion anomaly" and is considered undesirable.

Also, when facts are gained about two entities with one insertion, a fact can not be inserted about one until we have an additional fact about another entity has been obtained.This characteristic is called an "insertion anomaly". These anomalies can be eliminated by creating two new relations via projection.

53

Record structure is detailed-design of record rela-
tionship. A recod is created, many-to-many relationship
between records need another record which is called "inter-
section record" to match their relationship(e.g. IUQ record
in Appendix A).

A trends toward integrated file structures has
resulted in the grouping of all data elements relevant to
the management and operations section of a user organiza-
tion. The emerging database concept requires placing all
relevant data in one database in a consistent and standard-
ized manner, and providing selective inquiry and extraction
capabilities designed to meet a wide variety of information
requests. Therefore, record structure must be well aggre-
gated and organized in order to achieve the goals of this
system.

4. Data Dictionary

Management of a database is usually a complex
process. It requires the database administrator to keep
track of all the database and user view definitions as well
as their use.

Data dictionaries have been developed to aid the
database administrator in this task. The generation of the
data dictionary which documents functions, data classes,
allowable values , formats, and their interrelationship
should be initiated at this point [Ref. 8].

Individual DBMS have their own methods for defining
data descriptions. Each has a repository for the database
description, a language facility to process that descrip-
tion, and a mechanism to input that description to the DBMS.
A comprehensive dictionary will include cross-reference
information such as which programs use which pieces of data,
which departments require which reports, and so on. The
general objectives of a data dictionary are to provide
[Ref. 9].

52
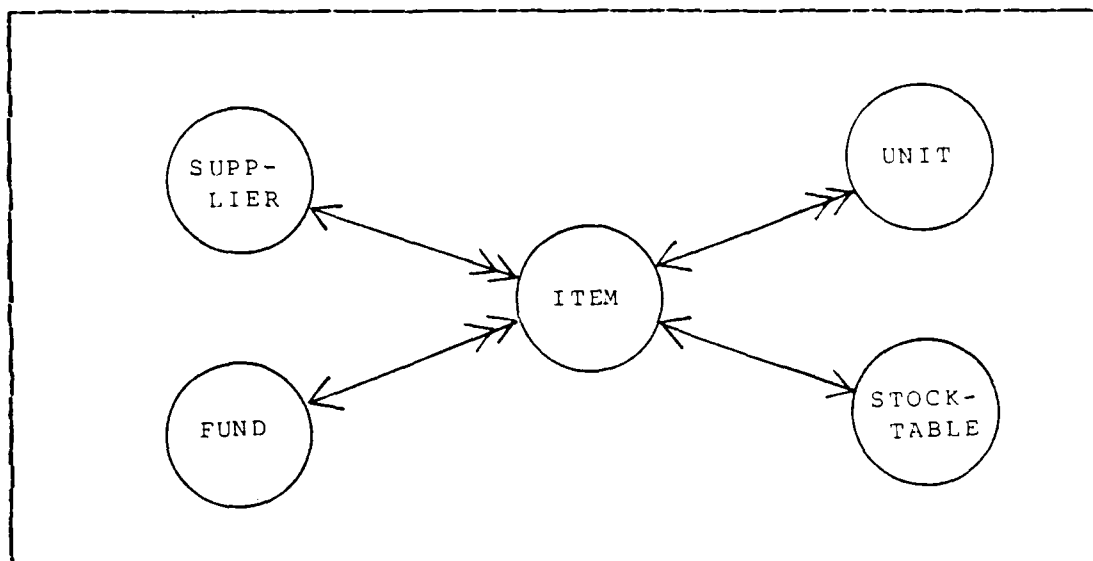
Figure 5.4   Bachman Diagram of 2nd Logistics Support Command

that have been defined.   A relationships may exist among
three or four or more records.

At this point the design team must discriminate
between theoretical and useful relationships. A theoretical
relationship can exist logically, but may never be needed in
practice.

Figure 5.4 represents a situation wher one supplier
supplies many items,   and one item is supplied by only one
supplier. One team could be handled by many units, and one
unit have many items.

3. Record Atructure

In Appendix A, each record structure represents a
view of the subschema/schema. This record structure shows a
relation among attributes, key attribute which is underlined
to identify each record and relation between record(entity)
and attributes. Also full-name of attributes are described
to identify each attribute.

a. gain familiarity with the area of the organization to be modeled

b. determine the information requirements of the organization without regard to constraints other than the way in which the organization does business;

c. represent these requirements via same formal modeling technique

The main purpose of requirements analysis is to understand the user's needs. Subsequent steps of the schema design process can transform these needs to subschemas according to the relational data model.

Information requirements are collected from users at all levels in the organization. From top management, information on the goals and objectives of the organization can be obtained, along with strategies and methods for managing the implementation of the strategies. Middle management provides data about required response time, reliability, security, and privacy, etc. Finally, operations management provides more specific information, such as names, sizes, number of occurrences, integrity constraints, reliability, security and privacy of data [Ref. 7].

2. Record Relationships

The essence of database is the representation of record relationships. The relationships can be specified in a variety of ways. Figure 5.4 shows one technique, called a data structure diagram, or DSD (also called a Bachman diagram). This method was used, because this is a simple methods to represent overall records structures. The single/double arrow notation is used to express relationships among records (one-to-one, one-to-many, many-to-many relationships). The DSD only shows the relationships among records.

The relationships are identified intuitively. The design team considers potential relationships among records

50

g. Join --- the join operation is a combination of the product, selection, and(possibly) projection operations. The join of two operations, say A and B, operates as follow: first, the product of a times B is formed. Then, selection is done to eliminate some tuples(the criteria for the selection are specified as part of the join). Then, (optionally) duplicate attributes are removed with projection.

C. SCHEMA DESIGN

A relational database is specified by a relational schema which consists of one or more relational subschemas. A relational subschema is a listing of a relation name and its corresponding attributes. Figure 5.3 represents an example of a relational schema for 2nd Logistics Support Command's database system.

```
ITEM ( SN, NM, UI, UP, OST, QTY, QTYOH, S:ID)
UNIT ( U:ID, U:NAME, PHONE, ADDR, ZIP)
```

Figure 5.3   An Example of a Relational Schema

1. Requirement Analysis

The first step of schema design is requirements analysis. This step consists of a high-level analysis of the function of an organization. The functions of the 2nd Logistics Support Command given in Chapter II are an example of requirements analysis. The purpose of this step is to

operations manipulate relations to form new relations. For example, the operation +(or union) combines the tuples of one relation with tuples of another relation. The result is a third relation [Ref. 1,5].

a. Union --- The union of two relations is formed by combining the tuples from one relation with those of a second relation to produce a third. Duplicate tuples are eliminated. For this operation to make sense, each relation must have the same number of attributes, and the attributes in corresponding columns must come from the same domain.

b. Difference --- The difference of two relations is a third relation containing tuples which occur in the first relation but not in the second.

c. Intersection --- The intersection of two relations is a third relation containing common tuples.

d. Product --- the product of two relations(sometimes called the cartesian product)is the concatenation of every tuple of one relation with every tuple of a second relation. The product of relation A(having m tuples) and relation B(having n tuples) has m times n tuples.

e. Project --- Projection is an operation that selects specified attributes from a relation. The result of projection is a new relation having the selected attributes. In other words, projection picks columns out of relation. Projection can also be used to change the other of attributes in a relation.

f. Selection --- Whereas the projection operator takes a vertical subset(columns) of a relation, the selection operator takes a horizontal subset(rows). Projection identifies attributes to be included in the new relation ; selection identifies to be included in the new relation.

1. Categories of Relational DML

"Relational algebra", one of the strategies, defines operators that work cn relations (akin to the operators +,-, etc., in high school algebra). Relations can be manipulated using these operators to achieve a desired result. Relational algebra is hard to use, partly because it is procedual.

Relational calculus is a second strategy for manipulating relations. Relational calculus is nonprocedual. It is language for expressing what we want without expressing how to get it. As integration in calculus has a variable, relational calculus has a similar variable. For "tuple relational calculus", the variable ranges over the tuples of a relation. For "domain relational calculus, the variable ranges over the values of a domain.

"Transform-oriented" languages are a class of nonprocedual languages that use relations to transform input data into desired outputs. These languages provide easy-to-use structures for expressing what is desired in terms of what is known. SQUARE, SEQUEL, and SQL are all transform-oriented languages.

The fourth category of relational DML is "graphic". Systems based on this technology provide the user with a picture of the structure of a relation. The user fills in an example of what is wanted, and the system responds with actual data in that format. Query-by-example(QBE) is an example of this process.

2. Relational Algebra

Here, only relational algebra is briefly described. Relational algebra is a far from the algebra operations like +, -, *, and / operated on numeric quantities. For relational algebra, the variables are relations, and the

47

3. Keys

A key can be considered an attribute or a set of attributes which uniquely identify each entity in an entity set. It is necessary to be able to identify each tuple in a relation by values of its attributes. For example, the value {100, M16A1RIFLE, EA, 100.00} constitutes a unique identifier. It will be unique because duplicate tuples are not allowed. And attribute SN {100} of the ITEM relation has the property that each ITEM tuple contains a district SN value. This value may be used to distinguish that tuple from all others in the relation. SN is said to be the primary key for ITEM.

A relational schema will have more than one attribute or combination of attributes that are unique. For the relation figure 5.2, attributes SN and nomenclature may both be unique. If so, they are called "candidate keys". In the design of the database, one of them will be chosen as a primary key.

When an attribute in one direction is a key of another relation, the attribute is called a "foreign key". The term means that the attribute is a key, but in a foreign relation.

B. RELATIONAL DATA MANIPULATION

Having described the processing of relations in a general and intuitive manner. However, to process relations with a computer, it is necessary to present a clear, unambiguous language for manipulat the data. Four different strategies for relational data manipulation have been proposed [Ref. 1].

46

2. Domains and Attributes

Each attribute has a domain, which the set of values that the attribute can have. That is, "M16A1RIFLE" is a value of attribute nomenclature. An attribute is the property of an entity which which associates a value from a domain with each entity. The domain of stock number (SN) is all positive integers less than 999.

```
 ┌──────────────────────────────────────────────────────────────────┐
 │  ┌──────────────────────────────────────────────────────────────┐ │
 │  │     Col.1        Col.2     ....................             │ │
 │  │    ─────────────────────────────────────────────────────────  │
 │  │    STOCK-│   NOMENCLATURE  │UNIT-OF│  UNIT-OF  │            │ │
 │  │    NUMBER│                 │-ISSUE │  -PRICE   │            │ │
 │  │    ═══════════════════════════════════════════════════      │ │
 │  │     100  │   M16A1 Rifle   │  ea   │  100.00   │            │ │
 │  │    ──────────────────────────────────────────────          │ │
 │  │     102  │   105m Motor    │  ea   │ 2000.00   │            │ │
 │  │    ──────────────────────────────────────────────          │ │
 │  │     105  │  M16 Ammunition │  box  │  200.00   │            │ │
 │  │    ──────────────────────────────────────────────          │ │
 │  │     200  │ 105m ammunition │  box  │   40.00   │            │ │
 │  │    ──────────────────────────────────────────────          │ │
 │  │     201  │    MO - gas     │  dm   │  220.00   │            │ │
 │  │    ──────────────────────────────────────────────          │ │
 │  └──────────────────────────────────────────────────────────────┘ │
 └──────────────────────────────────────────────────────────────────┘
```
row 1 — STOCK-NUMBER / NOMENCLATURE / UNIT-OF-ISSUE / UNIT-OF-PRICE
row 2 — 100, M16A1 Rifle, ea, 100.00

Figure 5.2   Item Relation

A relation of degree n has n domains, not all of which need be unique. For example, consider age and age of spouse attributes, where the domains of the two attributes are the same, that is, than are integers from 1 to 100. To differentiate between attributes that have the same domain, each is given a unique attribute name, like age, spouse-age.

Figure 5.2 is example, or occurrences. The generalized format, ITEM(SN, nomenclature, unit-of-issue, unit-price), is called the relation structure, and is what most people mean when they use the term "relation". If we add constraints on allowable data values to the relation structure, we have a relation schema [Ref. 3].

45

```
        -------------
        |  2nd LSC  |
        -------------
     T--------------|--------------T
 ----------    -----------    -----------
 | ORD Bn |__  |  QM Bn  |__  | Amm. Bn|__
 ----------  T -----------  T -----------  T
   |_____|T   |_____|T    |_____|T
```
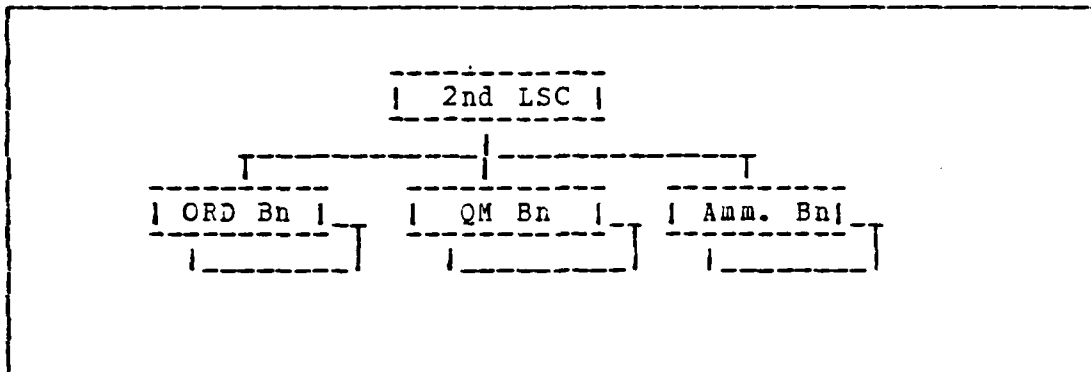
Figure 5.1   Organization of 2nd Logistics Support Command

A. STRUCTURE OF A RELATIONAL MODEL

1. Relations

The data structuring total used by the relational database model is a relation which is simply a two-dimensional table that has several properties. First, the entries in the table are single-valued; neither repeating groups nor arrays is allowed. Second, the entries in any column are all of the same kind. For example, one column may contain nomenclatures, and another unit-prices. Further, each column has a unique name and the order of the columns is immaterial. Columns of a relation are referred to as attributes. And no two rows in the table are identical and the order of the rows is significant. Figure 5.2 portrays a relation.

Each row of the relation is called a tuple. If the relation has n columns, then each row is referred to as an n-tuple. Also, a relation that has n columns or n attributes is said to be of degree n. The relation in figure 5.2 is of degree 4, and each row is a 4-tuple.

V. RELATIONAL DATABASE DESIGN

Database design is one of the most important steps in the development of computerized system. size and complexity combine to make this task disproportionately time comsuming and expensive.

Developing a database is an evolutionary process with the objective being an "idealized database". This is information that contains all the necessary data about all facets of an organization's operations and from which can be extracted instantaneously, in any form desired, information in response to inquiries in any format.

There are many ways in which a database can be designed. Here, we will describe a design theory and application for 2nd Logistics Support Command.

The data processing center of the 2nd Logistics Support Command for the Korea Army logistics systems has the responsibility to analyze, develop and maintain the computer based logistics system modeling to support keeping of track all inventory of each echelon and supply system such as procurement for the army.

This thesis documents the construction of a sample computer based database system for logistics support system in order to keep track of all information and the inventory status of each item in each subordinate battalions which are Ordnance Battalion, Quarter-master Battalion, and Ammunition Battalion Figure 5.1.
Each Battalion has two units, so there are total six units. And each Battalion handles two items at least(actually more then hundred). The general background was already described in Chapter 2.

43

although most of the core concepts of the model are
defined and agreed upon, there are many not-agreed-on
variants of the core concepts. These variants create
confusion and lead to a dilemma.

e. DBMS-specific models --- There are over one hundred
different commercial DBMS products. The DBMS are some-
times categorized in terms of their underlying data
model. A DBMS is considered a relational system if it
conforms, in essence, to the relational data model.
Alternately, A DBMS is considered to be a CODASYL system
if it conforms, in essence, to the CODASYL DBTG data
model. A third category of DBMS is other. If a DBMS does
not conform to one of the above two data models, then it
has its own, unique data model. There are many systems
that fall into the other category.

f. ANSI/X3/SPARC data model --- The
ANSI/X3/SPARC (American National Standards Institute /
Committee X3 / Standards Planning and Requirements
(sub)-Committee) data model does support a variety of
different data models in figure 4.4. This model is a
model for DBMS design rather than for database design.
This have the external, conceptual, and internal schema.

rules, the designer has a good deal of latitude and flexibility.

c. Entity - Relationship model --- The entity-relationship model(E-R model) is primarily a logical database model, although it has some aspects of a physical model as well. As its name implies, the E-R model is explicit about relationship. Unlike SDM, in the E-R model both entities and relationships are considered to be different constructs. Entities are grouped into entity sets, and relationships are grouped into relationship sets.

An entity-relationship diagram is a graphical portrayal of entities and their relationships. It is useful to summarize the information in a design. It supports the representation of more general relationships.

d. CODASYL DBTG model --- The CODASYL DBTG(Conference on Data System Languages, Database Task Group) data model was developed by the same group that formulated COBOL during the late 1960s and is the oldest of the data models. The DBTG model is a physical database model. There are constructs for defining physical characteristics of data, for describing where data should be located, for instructing the DBMS regarding what data structures to use for implementing record relationships, and other similar physical characteristics.

A DBTG schema is the collection of all records and relationships. A subschema is a subset and reordering of records and relationships in the schema. Unlike the relational model, relationships become fixed when they are defined in the schema.

Several reasons account for the lukewarm response that the CODASYL model has received, including the fact that it has a decidely COBOL flavor to it. Finally,

41

that data are arranged in relations but that relation-
ships are concerned to be implied by data values.

The principle advantage of carrying relationships in
data is flexibility. Relationships need not be predefined
[Ref. 2,3,4]. Further discussion of this will be in the
next chapter.

b. Semantic data model --- The word semantic means
meaning. The semantic data model provides a vocabulary
for expressing the meaning as well as the structure of
database data. As such, SDM is useful for logical data-
base design and documentation. SDM provides a precise
documentation and communication medium for database
users. In particular, a new user of a large and complex
database should find its SDM schema of use in determining
what information is contained in the database. Also, SDM
provides the basis for a variety of high level semantics-
based user interfaces to a database.

SDM has been designed to satisfy a number of criteria
that are not met by contemporary database models. The
chief advantage of SDM is that it provides a facility for
expressing meaning about the data in the database.

Another advantage of SDM is that it allows data to be
described in context. Users see data from different
perspectives. They see it relative to their field of
operation. SDM allows relative data definition.

A third advantage of SDM is that constraints on data-
base data cam be defined. For example, if a given item is
not changeable, SDM allows this fact to be stated. With
other data models, such constraints are not part of the
schema description and are documented separately.

SDM is like pseudocode, but instead of describing the
structure of programs as pseudocode does, SDM describes
the structure of data. Like pseudocode, SDM has certain
structures and rules, and within those structures and

40

And, when a fact is updated, the changing of one fact requires the research for all tuples containing this facts. This character is called an "updating anomaly".

2. <u>Normal Forms</u>

There are seven kind of normal forms which are shown in figure 5.5 [Ref. 1]. Usually, third normal form can be a design goals.

First Normal Form (1NF)
Second Normal Form (2NF)
Third Normal Form (3NF)
Boyce-Codd Normal Form (BCNF)
Fourth Normal Form (4NF)
Fifth Normal Form (5NF)
*Domain Key Normal Form (DK NF)
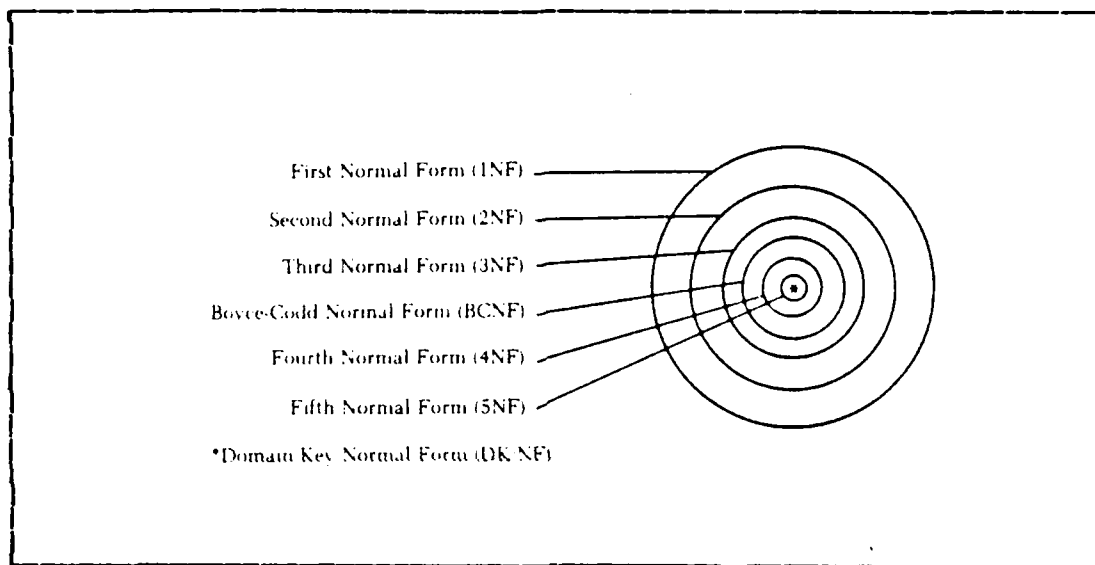
Figure 5.5   Relationship of Normal Forms

a. First normal form --- first normal form is the starting point: all relations are in first normal form. Relations in first normal form have modification anomalies. Some of these anomalies can be eliminated by putting the relation in second normal form. Second, third, and boyce-codd normal forms all address anomalies caused by inappropriate "functional dependencies".

A functional dependency is a relationship between attributes. Attribute Y is said to be functionally

54

dependent on attribute X if the value of X determines the
value of Y. For example, suppose that the serial number of
an item is known, the nomenclature of the item can be
determined.

An attribute is a 'determinant"if it occurs on the
left-hand side of a function dependency. Determinants may
or may not unique.

b. Second normal form --- The company relation in figure
5.6 has modification anomalies. If the tuple for SN 300 is
deleted, the fact that DAE-WHA's item costs $2500 is lost.
Also, can not be entered a company until an item is
procured.

The problem with this relation is that it has a depen-
dency involving only part of the key. The key is the
combination(SN, company), but the relation contains a
dependency, company --> price for an item. The modifica-
tion anomalies could be eliminated if the nonkey attri-
bute, price, were dependent on all of the key, not just
part of it. This leads that a relation is in second normal
form if all nonkey attributes are dependent on all of the
key.

Company can be decomposed via projection to form to
relations in second normal form. The relations to be
formed are COMPANIES{SN,COMPANY}, and COMPANIES{COMPANY,
PRICE}.

c. Third normal form --- Unfortunately, relations in
second normal form also have anomalies. Consider the
supplier relations in figure 5.7a. The key is SN, and the
functional dependencies are SN --> COMPANY and COMPANY -->
CITY. This means that a SN(item) supplied by only one
company since company determines city. Since SN determines
company and since company determines city, indirectly, SN
--> CITY(the item is supplied in the city). Thus this
relation is in second normal form(both city and company

55

```
|------------------------------------------------------|
| SN  |        COMPANY        |     PRICE     |
|=====|======================|===============|
| 100 |     Pung - san       |     1000      |
|-----|----------------------|---------------|
| 100 |     Dae - woo        |     2000      |
|-----|----------------------|---------------|
| 200 |     Dae - woo        |     2000      |
|-----|----------------------|---------------|
| 200 |     Remington        |     3000      |
|-----|----------------------|---------------|
| 300 |     Dae - wha        |     2500      |
|------------------------------------------------------|
```

Figure 5.6   Relation with a Two-Attribute Key

are determined by SN). However,   it has anomalies. If
fourth tuple in the relation is deleted, not only is the
fact lost that SN 400 is made by Remington, the fact that
the Remington company is in Detroit is also lost.

So, second normal form is not enough.   To eliminate
these anomalies, the transitive dependencies(X-->Y, Y-->Z)
must be eliminated.   This leads to a definition of third
normal form: A relation is in third normal form if it is
in second normal form and if it has no transitive
dependencies.

The supplier relation can be divided by projection
into two relations in third normal form.   The relations
SN-SUP(SN, company) and SUP-CITY(COMPANY, CITY) in Figure
5,7b are examples. Also Appendix A, also gives examples.
Record structure show the third normal form of 2nd LSC
relation.

d. Other normal form --- Also, even relations in third
normal form can have anomalies. This situation leads to
the definition of bcyce-codd normal form(BCNF). A relation
is in BCNF if every determinant is a candidate key which
two or more attributes or attribute collections can be a

56

```
    SN-SUP(SN, COMPANY)        SUP-CITY(COMPANY, CITY)
   -------------------------   ---------------------------
   |  SN |    COMPANY     |    |  COMPANY   |    CITY     |
   |=====================| |   |=========================|
   | 100 |   Pung-San    |    |  Pung-San  |   Ma-San    |
   |---------------------|    |-------------------------|
   | 200 |   Dae-Woo     |    |  Dae-Woo   | Chang-Won   |
   |---------------------|    |-------------------------|
   | 300 |   Dae-Woo     |    | Remington  |  Detroite   |
   |---------------------|    ---------------------------
   | 400 |   Remington   |
   |---------------------|
   | 500 |   Pung-San    |
   -----------------------
```
b. Relations Eliminating the Transitive Dependency

SUPPLIER(SN, COMPANY, CITY)
```
   -----------------------------------------------------------
   | SN  |        COMPANY       |        CITY              |
   |=========================================================|
   | 100 |     Pung - San       |      Ma - San            |
   |---------------------------------------------------------|
   | 200 |     Dae - Woo        |    Chang - Won           |
   |---------------------------------------------------------|
   | 300 |     Dae - Woo        |    Chang - Won           |
   |---------------------------------------------------------|
   | 400 |     Remington        |      Dtroite             |
   |---------------------------------------------------------|
   | 500 |     Pung - San       |      Ma - San            |
   -----------------------------------------------------------
```
a. Remington with Transitive Dependency

Figure 5.7   Elimination of Transitive Dependency

key. Relations in BCNF have no anomalies regarding func-
tional dependencies, but anormalies can arise from situ-
aticns other that functional dependencies.

Formally, multivalued dependency is defined as
follow; in relation R(X,Y,Z), X-->Y if each X value is asso-
ciated with a set of y values in a way that does not depend
on the Z values.

A relation is fourth normal form if it is in BCNF
and has no multivalued dependencies, or if it is in BCNF and
all multivalued dependencies are also function dependencies.
This means that if a relation has multivalued dependencies
and it is in forth normal form, then the multivalued depen-
dencies have a single value.

A relation is in fifth normal form if and only if every join dependency in relation is implied by the candidate keys of the relation.

A relation is in domain/key normal form if every constraint on the relation is a logical consequence of the definition of keys and domains. A constraints is any rule on static values of attributes that is precise enough that can evaluate. DK/NF means that if one can find a way to define keys and domains such that all constraints will be satisfied when the key and domain definitions are satisfied, then modification anomalies are impossible. If one can put a relation in DK/NF, then it is guaranteed that there be no anomalies. But there is no way to convert a relaticn to DK/NF automatically, nor is it even known which relations can be converted to DK/NF.

E. RELATIONAL DATABASE DESIGN CRITERIA

There are several different kind of criteria for producing an effective relational database design. Barri and co-workers have identified three relational criteria [Ref. 10],

* Representation : The final structure must correctly represent the original specification.
* Specification : The original specification are divided into relations that specify certain conditions.
* Redundancy : The final structure must not contain any redundant information.

Also, D. Kroenke has provided the three following design criteria which are elimination of modification anomalies, relation independence, and ease of use [Ref. 1].

1. Elimination of modification anomalies : If relaticns can be put into DK/NF, then no modification anomalies can occur. Thus DK/NF becomes a design objective, and relations that are in DK/NF are usually preferred.

58

Not all relations can be put into DK/NF, as described earlier. This occurs when there are constraints that cannot be expressed as logical consequences of keys and domains. An example described by Fagin [Ref. 10] is a relation having the following constraints; the relation must never have fewer than three tuples. There is no way to express this constraint in terms of domain and keys. Thus it has a modification anomaly. In fact, this strange relation has a deletion anomaly but no insertion anomaly.

When relations cannot be transformed into DK/NF, the constraint cannot be expressed in term of domains and keys must be inserted into application programs. This is undesirable because the constraint is hidden.

2. Relation independence : Two relations are independent if modifications can be made to one without regard for the other. The greater the independence, The better. However, independence in not always achievable. For example, interrelation constraints are a form of relation dependence. To eliminate this dependence, the relations can be joined together. The joined relation, however, may have modification anomalies.

Here the conflict in design goals is seen. To eliminate modification anomalies, relations are split; but in so doing, interrelation dependencies are created. In this case it is necessary to choose the least of the evils, based on the requirements of the application.

Projections that generate false data upon joining are called "loss projections". Projections that do not cause false data to be generated on joining are called "nonloss projections". If the projections do

not capture the essence of the functional dependencies, then it could be cause of loss projections.

3. Ease of use : A third criterion for a relational design is ease of use. as far as possible, we strive to structure the relations so that they are familiar and seen natural to users. Sometimes this goal conflicts with the elimination of anomalies or with independence.

VI. IMPLEMENTATION

A. RELATIONAL DATABASE MANAGEMENT SYSTEM

The relational model as the theoretical basis was introduced in the previous chapter. An important aspect of the theory is normal relations. Normal relations process many desirable structual properties. The criteria of normal relations are subsequently applied in data analysis to realize logical structures that satisfy normal relation properties.

This chapter describes the relational model as an implementation model that is supported by a DBMS. Any relations produced during data analysis can be implemented directly on the DBMS.

Because of its tabular interface, the relational model makes an attractive implementation model. It is receptive to two types of environment [Ref. 11]

1. the traditional data processing environment, where databases are set up by professional computer programmers on behalf of database users;

2. environments in which nonprogrammer users set up their own databases;

The relational model provides the same advantage in both types of environment. Its natural interface simplifies the design and use of the database. This is particularly so if a language with powerful selective capabilities can be provided by the DBMS. Such languages can reduce program development time and hence are attractive in commercial data-processing environments. They are also attractive to nonprogrammer users, allowing them to use the database without resorting to computer-oriented procedual languages.

61

However, there are problems in relational model implementations. A powerful language such as relational algebra or SQL is necessary to realize the full potential of a relational DBMS. These languages can be expensive to implement and use.

1. Relational Characteristics

What characteristics must a DBMS have to be considered a relational product? In his lecture, E.F. Codd [Ref. 12] defined a relational DBMS as one in which data is defined in tables and processed by using SELECT, PROJECT and unrestricted JOIN operations, or their equivalent. Codd called a system having these characteristics Minimally Relational [Ref. 1].

SELECT, PROJECT, and JOIN will be used in next section. The SELECT obtains rows of the table according to criteria on raw contents. PROJECT obtains columns of a table by column name. Finally, JOIN brings two relations together based on the relationship between two columns having the same domain.

Some DBMS products specify that only columns can be used as JOIN criteria. For examples, a DBMS may require the columns used as JOIN criteria to be indexed. This implies the undesirable situation of restricting user activity because of physical data representation. To the nonspecialist user, this restriction appears arbitrary. To eliminate this situation, Codd specifies that a minimally relational system must have unrestricted JOINS. This means that any column can be used as criteria for the join.

2. Commercial Relational DBMS

There are currently many commercial DBMS products that claim to be relational. Some are more relational in

62

name than in actuality. Criteria can be used to access whether or not a product is truly a relational product. Specifically, the DBMS should model data as tables, and it should support SELECT, PROJECT, and unrestricted JOIN operations.

Relational DBMS can be divided into three groups. One group is based on the data language SQL, one on the data language QUEL, and a third contains systems falling into neither of the other two categories [Ref. 1].

Three major SQL-based DBMS products are SQL/DS, system R, and ORACLE. System R is a research system developed by IBM for the study of relational technology. ORACLE is vended by Relational Software Incorporated. Originally, ORACLE was developed for operation on Digital Equipment Corporation PDP minicomputers. Since its origin, ORACLE has been converted to operate on IBM mainframes as well. ORACLE's user interface is based on SEQUEL II, an earlier version of SQL. According to RSI, ORACLE will soon be compatible with the current version of SQL. QUEL is a data language like SQL. (Just like COBOL and PL/I are alternative programming languages, SQL and QUEL are alternative data languages.) QUEL is based on tuple relational calculus. QUEL is nonprocedual and allows the user to process data without concern for physical data structures.

There are many other relational DBMS. Figure 6.1 lists some of the major systems as of late 1982. There is also a microcomputer relational product:--- dBASE II is used to implement 2nd LSC system is an example of a relational (or tabular) DBMS that restricts join operations. The join columns must be indexed.

```
     SQL-Based Systems
          SQL/DS, IBM
          ORACLE, Relational Software, Inc.
          System R, IBM
     QUEL- Based Systems
          INGRES, Relational Technology, Inc.
          IDM 500, Britton-Lee, Inc.
     Other Relational Systems
          MRDS/LINUS, Honeywell
          dBASE II, Ashton-Tate
          NOMAD, National Computer Sharing Services
```

Figure 6.1   Relational DBMS Products and Vender

B. IMPLEMENTATION USING DBASE II

The 2nd Logistics Support Command System has been imple-
mented using dBASE II relational DBMS.   As a word processor
allows one to manipulate characters, words, sentences and
pages to create a document that fits one's needs, dBASE II
allows one to work with fields, records, and files to manage
data in just the desired manner

To provide the necessary power, dBASE II provides many
data management facilities. Among these are an interactive
query language, a report writer to create tabular reports,
and a powerful programming language that allows a knowledge-
able person to adapt dBASE II to the needs of those who are
less familiar with computers.

The basic operations necessary to implement and under-
stand the 2nd LSC System in Appendices are given. If a more
detailed explanation is desired,   reference to dBASE II
user's guideis recommended [Ref. 13,14].

First, to create a file, CREATE command is used.

```
. CREATE
ENTER FILENAME: ITEM
ENTER RECORD STRUCTURE AS FOLLOWS:
   FIELD   NAME,TYPE,WITH,DECIMAL PLACES
   001     SN,C,3
   002     NOMENCLATURE,C,10
   003     QUANTITY,C,10
   004     <CR>
INPUT DATA NOW? N
```

A file called ITEM.DBF has now been created on the disk.
To select a file to work with by giving the command USE

```
. USE ITEM
```

To add the information to the file,   use the APPEND
command.   APPEND lets the user move the cursor to any field,
and enter or change the information.

```
.  APPEND

RECORD 00001
SN            : 423
NOMENCLATURE: M16A1RIFLE
QUANTITY    : 222

RECORD 00002
SN            : 234
NOMENCLATURE: 105MOTOR
QUANTITY    : 150

RECORD 00003
SN            : <CR>
```

Now necessary information has been added. To list a data
file's contents on the screen, LIST command is used.

```
.  LIST

00001  423   M16A1RIFLE    222
00002  234   105MOTOR      150

.  LIST FOR SN = "423"
00001  423   M16A1RIFLE    222
```

To sort a data file

```
. SORT ON SN TO NEWSN
SORT COMPLETE
. USE NEWSN
. LIST SN,NOMENCLATURE

00001  234 105MOTOR
00002  423  M16A1RIFLE
```

NEWSN is given as new file name for sorted file. If a specific item, and not the whole list, is wanted, then the FIND command can be used to display on the screen.

```
. FIND 105MOTOR
. DISPLAY

00001  234   105MOTOR  150
```

```
    VARIABLE NAME: PHONE
    FORMAT: ALPHANUMERIC
    WIDTH: 15
    ALLCWABLE VALUE: ALL TELEPHONE NUMBER
    DESCRIPTION: TELEPHONE NUMBER OF SUPPLIER

IND FILE ---------
    VARIABLE NAME: F:ID
    FORMAT: CHARACTER
    WIDTH: 5
    ALLCWABLE VALUE: ABSTRACTED FUND SOURCE NAME, UNIQUE
    DESCRIPTION: REPRESENT SOURCE OF FUND IN ABSTRACTED
                    TYPE

    VARIAELE NAME: F:SOURCE
    FORMAT: CHARACTER
    WIDTH: 10
    ALLCWABLE VALUE: SOURCE OF FUND
    DESCRIPTION: IDENTIFY THE SOURCE OF FUND BY FULL NAME

TOCKTABLE FILE ---------------
    VARIABLE NAME: SN    * THE SAME IN THE ITEM FILE

    VARIAELE NAME: REQOBJ
    FORMAT: ALPHANUMERIC
    WIDTH: 10
    ALLOWABLE VALUE: SELECTED LEVEL, ONLY DIGIT
    DESCRIPTION: REPRESENT REGUIRED QUANTITY ON THE STOCK

    VARIABLE NAME: SAFLVL
    FORMAT: ALPHANUMERIC
    WIDTH: 10
    ALLCWABLE VALUE: SELECTED QUANTITY BY DIGIT
    DESCRIPTION: REPRESENT SAFETY LEVEL OF ITEM  BY UNIT

    VARIABLE NAME: RDRPNT
    FORMAT: ALPHANUMERIC
    WIDTH: 10
```

ALLOWABLE VALUE: ALL ALPHANUMERIC
DESCRIPTION: THE ADDRESS OF EACH UNIT

VARIABLE NAME: CITY
FORMAT: CHARACTER
WIDTH: 10
ALLOWABLE VALUE: ALL CITY'S NAME
DESCRIPTION: CITY NAME WHERE UNIT IS LOCATED

VARIABLE NAME: ZIP
FORMAT: ALPHANUMERIC
WIDTH: 5
ALLOWABLE VALUE: ONLY DIGIT
DESCRIPTION: REPRESENT OF LOCATION BY DIGIT

SUPPLIER FILE --------------
VARIABLE NAME: S:ID   * THE SAME IN THE ITEM FILE

VARIABLE NAME: CCUNTRY
FORMAT: ALPHANUMERIC
WIDTH: 12
ALLOWABLE VALUE: ALL COUNTRY'S COMMON NAME
DESCRIPTION: THE COUNTRY NAME WHICH ITEM WAS MADE

VARIABLE NAME: CCMPANY
FORMAT: CHARACTER
WIDTH: 15
ALLOWABLE VALUE: ALL COMPANY NAME WHICH ABSTRACTED
DESCRIPTION: TO IDENTIFY THE COMPANY WHICH MADE THE
             ITEM

VARIABLE NAME: LCC
FORMAT: CHARACTER
WIDTH: 15
ALLOWABLE VALUE: NAME OF AREA
DESCRIPTION: THIS IS TO KNOW EASILY THE LOCATION OF
             COMPANY FOR EVERYBODY

80

```
VARIABLE NAME:QTYOH
FORMAT: NUMERIC
WIDTH: 10
ALLOWABLE VALUE: ALL NUMERIC, NUMBERS OF ITEM ON HAND
DESCRIPTION: THIS WILL REPRESENT QUANTITY OF ON HAND
             BY ITEM.

VARIABLE NAME: S:ID
FORMAT: CHARACTER
WIDTH: 10
ALLOWABLE VALUE:SELECTED SUPPLIER'S INDIVIDUAL UNIQUE
                NAME
DESCPIPTION: IDENTIFICATION OF SUPPLIER'CODE
```

UNIT FILE ----------

```
VARIABLE NAME: U:ID
FORMAT: ALPHANUMERIC
WIDTH: 4
ALLCWABLE VALUE: ONLY FOUR DIGITS
DESCRIPTION: USED FOR SECRET TO THE UNIT NAME
             MUST BE UNIQUE

VARIABLE NAME: U:NAME
FORMAT: ALPHNUMERIC
WIDTH: 5
ALLCWABLE VALUE: SELECTED UNIT NAME "99XXX"
DESCRIPTION: TO IDENTIFY UNIT SIZE and FUNCTIONS

VARIABLE NAME: PHONE
FORMAT: ALPHANUMERIC
WIDTH: 15
ALLOWABLE VALUE: ALL TELE PHONE NUMBER. DIGIT and "-"
DESCRIPTION: EACH UNIT'S TELEPHONE NUMBER

VARIABLE NAME: ADDR
FORMAT: ALPHANUMERIC
WIDTH: 22
```

APPENDIX B
DATA DICTIONARY

ITEM FILE ---------

    VARIABLE NAME: SN
    FORMAT: ALPHANUMERIC
    WIDTH: 15
    ALLOWABLE VALUE: SELECTED ITEM, ALL ALPHANUMERIC
    DESCRIPTION: STOCK NUMBER OF EVERY ITEM

    VARIABLE NAME: NM
    FORMAT: ALPHANUMERIC
    WIDTH: 15
    ALLOWABLE VALUE: ALL ALPHANUMERIC
    DESCRIPTION: NAME OF THE ITEM

    VARIABLE NAME: UI
    FORMAT: CHARACTER
    WIDTH: 2
    ALLOWABLE VALUE: "EA","BX"
    DESCRIPTION: THIS IS UNIT OF ISSUE, TWO VALUES ARE ALL

    VARIABLE NAME: UP
    FORMAT: NUMERIC
    WIDTH: 10
    DECIMAL: 2
    ALLOWABLE VALUE: ALL NUMBERES
    DESCRIPTION: UNIT PRICE, REPRESENTING UNIT IS "DOLLAR"

    VARIABLE NAME: CST
    FORMAT: ALPHANUMERIC
    WIDTH: 2
    ALLOWABLE VALUE: ALPHANUMERIC, NUMBER OF DAY
    DESCRIPTION: HOW LONG IT WILL TAKE AFTER ORDER

```
        CITY -- name of city      ZIP -- ZIP code

SUPPLIER(S:ID,COUNTRY,COMPANY,LOC,PHONE)
          ----

     S:ID -- supplier identification
     COUNTRY -- supplier country
     COMPANY -- supplier company
     LOC -- location of copany
     PHONE -- phone number

FUND(F:ID,F:SOURCE)
     ----

     F:ID -- fund identification
     F:SOURCE -- fund source

STOCKTABLE(SN,REQOBJ,SAFLVL,RDRPNT)
           --

     SN -- stock number
     REQOBJ -- request objective quantity
     SAFLVL -- safety stock level
     RDRPNT -- reorder point

IUQ(SN,U:ID,QTY) ----- intersection record
    -- ----

     SN -- stock number
     U:ID -- unit identification
     QTY -- quantity on hand of each item by each unit
```

APPENDIX A
RLATIONSHIPS AND RECORD STRUCTURE

1. RELATIONSHIP DIAGRAM
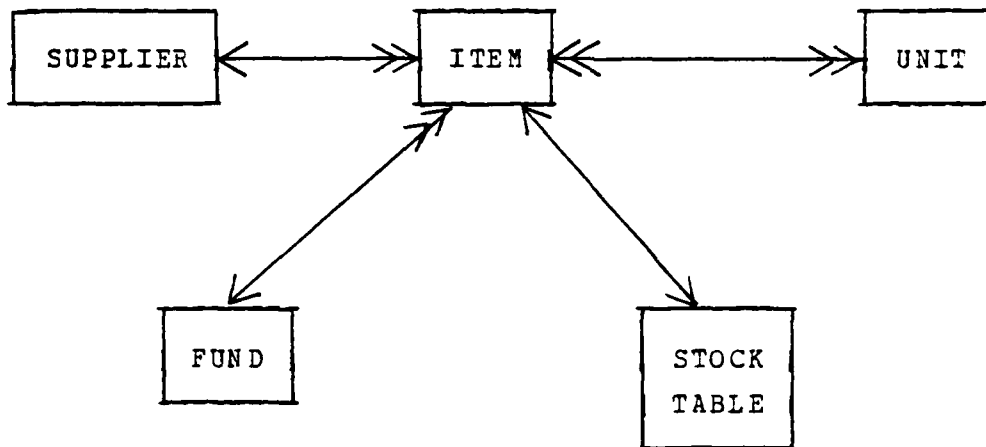
2. RECORD STRUCTURE

ITEM(SN,NM,UI,UP,OST,QTYOH,S:ID)

    --

    SN -- stock number

    NM -- nomenclature

    UI -- unit of issue

    UP -- unit price

    OST -- order shipping time

    QTYCH -- quantity on hand

    S:ID -- supplier identification

UNIT(U:ID,U:NAME,PHONE,ADDR,CITY,ZIP)

    U:ID -- unit identification

    U:NAME -- unit name

    PHCNE -- phone number

    ADDR -- address cf unit

This database can serve as a prototype for ROK Army database management systems. The question, What type of DBMS should be installed for each unit level of mainframe? is for further research.

VII. CONCLUSION

This thesis has focused on the Korean Army 2nd Logistics Support systems and inventory status. However its findings are applicable to all departments of the Korean military logistics support system.

The developed sample database presented here is based on a relational database model and a computerized six logistics support battalions of 2nd LSC for logistics personnel, and it may very well form the basis of the total Korean Army management system.

The army logistics system is very complex and deals with about 200 thousand items. To manually manage all army logistics system is a very tedious, time consuming job and would not prove effective to increase war power. Thus, the Army needs a computerized logistic management system

Database processing can increase end-user productivity, decrease staff, enable work to be done more efficiently, provide information effectively and timely, and increase combat capability.

A database is the interface between people and machines. Database design is a two-phased process. This thesis examined both logical and physical database design process, and this process is an iterative process to get closer to an acceptable and optimal design. Normal forms can be applied to decrease inefficiency of the relational database model in the system-design process.

Implementation of a sample database using dBASE II resulted in a more effective and timely presentation of all required logistics information. This DBMS in Appendix is developed for end-users who are working in the ROK Army 2nd LSC who do not have experience with database systems or computers.

To use more than one data file, dBASE II reserves two
areas of memory for data file. If it is necessary to use
another data file at the same time. SELECT SECONDARY must
be used while work is being done in the PRIMARY area of
memory. This will open the other area of memory. SELECT
PRIMARY moves the user back to the PRIMARY area.

```
. SELECT PRIMARY
. USE ITEM
. LIST

00001    423        M16A1RIFLE      222
00002    234        155MOTOR        140

. SELECT SECONDARY
. USE NEWSN
. LIST NOMENCLATURE

00001    NOMENCLATURE
00002    M16A1RIFLE

. SELECT PRIMARY
```

When it is necessary to know the internal structure of
a data file, the command LIST STRUCTURE which displays this
information can be used.

```
. LIST STRUCTURE

STRUCTURE FOR FILE: ITEM.DBF
NUMBER OF RECORDS:  00002
DATE OF LAST UPDATE: 3/10/85
PRIMARY USE DATABASE
FLD        NAME            TYPE   WIDTH   DEC
001        SN              C      003
002        NOMENCLATURE    C      010
003        QUANTITY        N      010
** TOTAL **                       00023
```

The record is still in the file, and has been marked
with an asterisk for deletion later. dBASE II follows the
safe method of removing records. They are first marked with
the DELETE command, and then removed with the PACK command.
This way it is possible to reconsider RECALL the record
before packing it.

```
. RECALL
00001 RECALL (S)

.DISPLAY
00001     423      M16A1RIFLE     222

. COPY TO TEST1
. USE TEST1
. 1
. DELETE
00001   DELETION (S)
. PACK
PACK COMPLETE, 00002 RECORD COPIED

. LIST
00001   234      155MOTOR        140
```

If the contents of data file is changed, the EDIT
command can be used.   EDIT allows full screen operation.

```
. LIST

00001     423        N16A1RIFIE       222
00002     234        105MOTOR         150

. EDIT 2
RECORD   00002
SN              :234
NCMENCLATURE :155MOTOR
QUANTITY      :140

. LIST
00001     423        M16A1RIFIE       222
00002     234        155MOTOR         140
```

To delete records from a data file

```
. USE ITEM
. LIST

00001     423        M16A1RIFIE       222
00002     234        155MOTOR         140

. 1
. DELETE

00001   DELETION (S)

.DISPLAY

00001   *423         M16A1RIFLE       222
```

This REPORT specification is the saved on the disk as
NEWSN.FRM. It may be printed at any time by repeating the
REPORT command.

```
.REPORT FROM NEWSN

    PAGE  NO. 00001
    3/10/85

                        EQUIPMENT STATUS REPORT

    SN          NOMENCLATURE    QUANTITY
    234         105MOTOR        150
    423         M16A1RIFLE      222

. REPORT FROM NEWSN FOR SN="423"

    PAGE  NO. 00001
    3/10/85

                        EQUIPMENT STATUS REPORT

    SN          NOMENCLATURE    QUANTITY
    423         M16A1RIFLE      222
```

dBASE II has a useful command REPORT. Designing a REPORT is similar to CREATEing a data file. The report can include totals of numeric field.

```
.USE ITEM
.REPORT FROM NEWSN
ENTER OPTIONS,  M=LEFT MARGIN,  L=lines/PAGE,  W=PAGE
WIDTH
PAGE CPTIONS? (Y/N) Y
ENTER PAGE HEADING: EQUIPMENT STATUS REPORT
DCUBLE SPACE REPCRT? (Y/N) N
ARE TCTAL REQUIRED? (Y/N) N
CCL      WIDTH, CCNTENTS
001           3, SN
ENTER HEADING: SN
002       10, NOMENCLATURE
ENTER HEADING: NCMENCLATURE
003       10, QUANTITY
ENTER HEADING: QUANTITY
004    <CR>
```

ALLOWABLE VALUE: SELECTED QUANTITY OF ITEM. ONLY DIGIT
DESCRIPTION: THE POINT WHICH MUST ORDER IF BELOW THAN
                THIS

IUQ FILE ---------
        VARIABLE NAME: SN   * THE SAME IN THE ITEM FILE

        VARIABLE NAME: U:ID * THE SAME IN THE UNIT FILE

        VARIABLE NAME: QTY
        FORMAT: NUMERIC
        WIDTH: 10
        ALLOWABLE VALUE: QUANTITY OF EACH ITEM BY UNIT
        DESCRIPTION: IDENTIFY QUANTITY OF EACH ITEM ON HAND

```
                        ┌───────────┐
                        │ MAIN MENU │
                        └─────┬─────┘
        ┌──────────┬─────────┼─────────┬──────────┐
   ┌────┴───┐  ┌───┴───┐ ┌───┴────┐ ┌──┴────┐ ┌───┴──┐
   │ SIGNON │  │  ADD  │ │ UPDATE │ │ QUERY │ │ HELP │
   └────────┘  └───────┘ └────────┘ └───────┘ └──────┘
```

ADD	UPDATE	QUERY
ITEM	ITEM	ITEM
UNIT	UNIT	UNIT
SUPPLIER	SUPPLIER	SUPPLIER
FUND	FUND	FUND
STOCKTB	STOCKTB	STOCKTB
IUQ	UPQTY	QTYOH
		MINQTY

```
        ┌──────────┐        ┌──────────┐
        │  DELETE  │        │  REPORT  │
        └────┬─────┘        └────┬─────┘
```

DELETE	REPORT
ITEM	BN-QTY
UNIT	BNASSET
SUPPLIER	TOTALQTY
FUND	
STOCKTB	
IUQ	

83
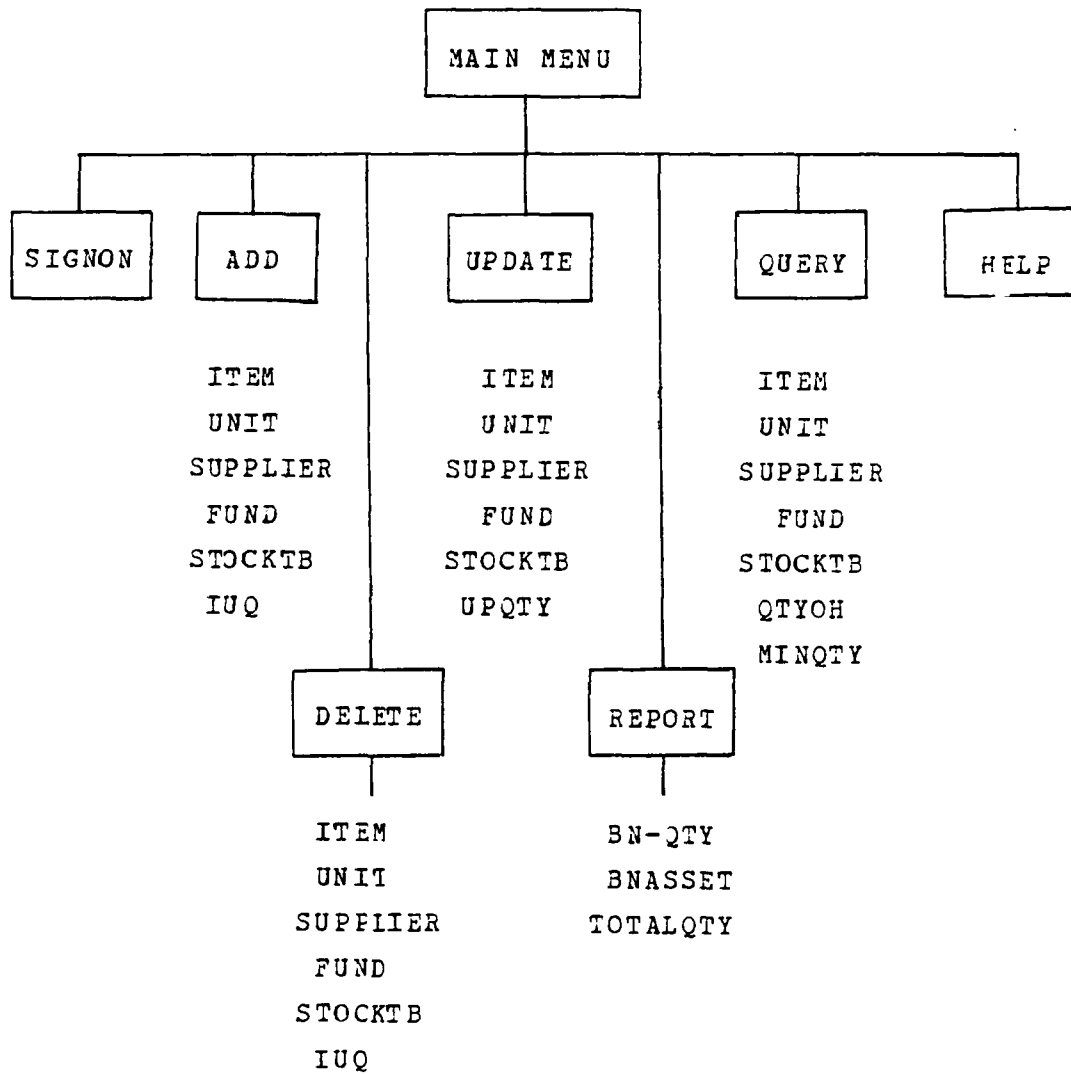
APPENDIX D
USER MANUAL

This manual is to use the IBM personel computer. First, You have to set the machine,   If the power turned off,then you need "Turn on" the power swich,   and put the main chassis's switch(red color) to the "0" position,and then do the following

1. Insert the DBASE11 floppy disk in the disk slot that is on the left(You can see the letters on the disk)

2. insert the IS4183 class disk in to the right slot and close the drive slot by pushing down.

3. Now turn on the machine by locating the switch position "1".

4. Turn on the moniter(The T.V.set) by rotating the switch on its face from "0" to "1".

5. Turn on the printer by locating the switch either on the side or back of it and moving it to the "1" position.

6. Wait for the machine to prompt you for the date on the screen.   You   will see "CURRENT DATE IS TUE 1-01-1980 ENTER NEW DATE".   Key the first month digit, next date and year. Then strike Return Key.

7. Then the system will give you the next message: "CURRENT TIME IS 0:00:15.65 ENTER THE TIME:" The same way with date, key the current "hour", "minute", and "second", then strike return key.

8. Then you will see the letter "A>".   Now you are ready to start up DBASE-II software.   Type "CEASE" right after that letter, and then strike return key. Tere are some messages, and last oh them Dot(.) will apear on the screen.

84

9. Type "SET DEFAULT TO B" and return key, then Dot will be appear again. Now you are ready to use the 2nd Logistics command database system.

10. Type "DO MAIN" and return key. The signon message will apear. If you ready then key the any character(this command are in the message). Then menu menu message will be on the screen.

11. You have to follow the command message which appear on the screen. If there is "waitting" word near the cusor on the screen do not need to strike return key. If not, strike the return key after put the selected menu or other words.

12. All kind of main function of this system are specified on the first message(main menu). Put the mian menu-number which you want, then tere will be another message for menu on the screen. Select menu-number which you want and follow the command message on the screen. This system is menu type.

13. All done which you wanted, you can stop by selecting menu "0". Also you can stop by keying "Esc" on the keyboard and type "QUIT"(start after Dot).

14. If you are familiar with this system, you can directly (do not use main menu and menu program) execute which your wanted procedure by the fcllowing command: "DO PROGRAM NAME". But if you are not famliar with, don't try this!

APPENDIX E
RECORD FORMAT

```
*******************ITEM.FMT   12/8/84*********************
*AUTHER : W.D. SONG
*PURPOSE : This program will format for item file.
*CALLED BY : ADD,UPDATE
*PROGRAM CALL : NONE
*LOCAL VARIABLE : NCNE
*FILES USED : ITEM.IBF
@ 1,0 SAY"ENTER THE SN " GET SN PICTURE "!!!!!!!!!!!!!!!"
@ 2,0 SAY"ENTER THE NM " GET NM PICTURE "!!!!!!!!!!!!!!!"
@ 3,0 SAY"ENTER THE UI " GET UI PICTURE "!!"
@ 4,0 SAY"ENTER THE UP " GET UP PICTURE "99999999.99"
@ 5,0 SAY"ENTER THE CST" GET OST PICTURE "!!"
@ 6,0 SAY"ENTER THE QTYOH " GET QTYOH PICTURE "9999999999"
@ 7,0 SAY"ENTER THE SUPPLIER ID"GET S:ID PICTURE
        "!!!!!!!!!!!"


*******************UNIT.FMT   12/7/84*********************
*AUTHOR:W.D.SONG
*PURPOSE: This program will format for unit file.
*CALLED BY:ADD,UPDATE
*PROGARMS CALL:
*LOCAL VARIABLE:
*FILES USED: UINT.DBF
*to clear the screen
@ 1,0 SAY"ENTER THE UNIT ID" GET U:ID PICTURE "!!!!"
@ 2,0 SAY"ENTER THE UNIT NAME" GET U:NAME PICTURE "!!!!!"
@ 3,0 SAY"ENTER THE PHONE NUMBER" GET PHONE PICTURE
        "!!!!!!!!!!!!!!!!"
@ 4,0 SAY"ENTER THE ADDRESS" GET ADDR PICTURE
```

```
                    "!!!!!!!!!!!!!!!!!!!!!!!!!!!"
@ 5,0 SAY"ENTER THE CITY" GET CITY PICTURE "!!!!!!!!!!"
@ 6,0 SAY"ENTER THE ZIP CODE" GET ZIP PICTURE"!!!!!"


********************SUPPLIER.FMT   12/7/84*****************
*AUTHOR:W.D.SONG
*PURPOSE: This program will format supplier file.
*CALLED BY:ADD.UPDATE
*PROGRAMS CALL:
*LOCAL VARIABLE:
*FILES USED: SUPPLIER.DBF
@ 1,0 SAY "ENTER THE SUPPLIER ID" GET    S:ID PICTURE
            "!!!!!!!!!!!"
@ 2,0 SAY "ENTER THE COUNTRY" GET COUNTRY PICTURE
            "!!!!!!!!!!!!!"
@ 3,0 SAY "ENTER THE COMPANY" GET COMPANY PICTURE
            "!!!!!!!!!!!!!!!!"
@ 4,0 SAY "ENTER THE LOCATION" GET LOC PICTURE
            "!!!!!!!!!!!!!!!"
@ 5,0 SAY "ENTER THE PHONE NUMBER" GET PHONE PICTURE
            "!!!!!!!!!!!!!!!!"


****************STOCKTABLE.FMT   12/8/84*******************
*AUTHOR:H.Y.LEE
*PURPOSE:This  program will  provide  to  format the  stock
table.
*CALLED BY:ADD.UPDATE
*PROGRAMS CALL:
*LOCAL VARIABLE:
*FILES USED:STOCKTABLE.DBF
@ 1,0 SAY "ENTER THE STOCK NUMBER" GET SN PICTURE
            "!!!!!!!!!!!!!!!!"
@ 2,0 SAY "ENTER THE REQUEST OBJECTIVE" GET REQOBJ PICTURE
            "!!!!!!!!!!!"
```

87

```
@ 3,0 SAY "ENTER THE SAFETY LEVEL" GET SAFLVL PICTURE
        "!!!!!!!!!!!"
@ 4,0 SAY "ENTER THE REODER POINT" GET RDRPNT PICTURE
        "!!!!!!!!!!!"


*******************IUQ.FMT   12/9/84*********************
*AUTHOR:W.D.SONG
*PURPOSE: This program will provide to format the
*         intersection file of thestock number and unit
*         and also provide on hand quantity.
*CALLED BY: ADD,UPDATE
*PROGRAMS CALL:
*LOCAL VARIABLES:
*FILES USED: IUQ.DBF
@ 1,0 SAY "ENTER THE STOCK NUMBER" GET SN PICTURE
        "!!!!!!!!!!!!!!!!"
@ 2,0 SAY "ENTER THE UNIT ID" GET U:ID PICTURE "!!!!"
@ 3,0 SAY "ENTER THE QTY OFEACH ITEM BY UNIT" GET QTY
        PICTUR: "9999999999"


******************FUND.FMT   12/8/84*********************
*AUTHOR: H.Y.LEE
*PURPOSE: This program will provide to format the fund
          file.
*CALLED BY:ADD,UPDATE
*PROGRAMS CALL:
*LOCAL VARIABLE:
*FILES USED:FUND.DBF
@ 1.0 SAY "ENTER THE FUND ID" GET F:ID PICTURE "!!!!!"
@ 2,0 SAY "ENTER THE FUND SOURCE" GET F:SOURCE PICTURE
        "!!!!!!!!!!!"
```

APPENDIX F
DATA LIST

```
A>DBASE
SET DEFAULT TO B
. USE ITEM
. LIST
00001  123456789123456 M16A1RIFLE      EA      111.22 14      3616666 PUNG-SAN
00002  123456789234567 105M-MOTOR      EA     2222.00 21          180 DAE-WOO
00003  123456789456789 M16AMMUNITION   BX      205.66 14      2377777 PUNG-SAN
00004  123456789567891 105MAMMUNITION  BX       40.40 21       172345 RMT
00005  123456789789123 COMBAT-SHOSE    BX      555.88 7          5700 DAE-WHA
00006  123456789912345 MO-GAS          DM      220.00 3       1764800 KYUNG-IN
.


    USE UNIT
. LIST
00001  1978 15ORD 22-33-1234      KYUNG-GI YANG-JU 232    DUK-JUNG    12312
00002  1258 20ORD 12-33-1222      KYUNG-GI YANG-JU 384    IL-DONG     12378
00003  5274 51AM  22-44-5454      KYUNG-GI AN-SUNG 343    AN-SUNG     17734
00004  5334 55AMM 12-12-2345      KANG-WON CHUL-WON 377   CHUL-WON    64422
00005  8756 11QTR 22-22-6666      KYUNG-GI YANG-JU 778    TAE-GAEWON  12333
00006  8976 15QTR 11-55-3456      KANG-WON WON-JU 994     WON-JU      88834
.


    USE SUPPLIER
. LIST
00001  PUNG-SAN    KOREA      PUNG-SAN AMM-CO MA-SAN        22-66-2345
00002  DAE-WOO     KOREA      DAE-WOO CORP.   CHANG-WON     55-33-7865
00003  RMT         U.S.A.     REMINGTON       DETROITE      203-444-7777
00004  DAE-WHA     KOREA      DAE-WHA CORP.   DAE-JEON      44-77-7896
00005  KYUNG-IN    KOREA      KYUNG-IN ENERGE IN-CHUN       23-45-3673
.


    USE STOCKTAB
. LIST
00001  123456789123456 2000000    1000000    1200000
00002  123456789234567 100        50         70
00003  123456789456789 2000000    1000000    1200000
00004  123456789567891 200000     100000     120000
00005  123456789789123 3000       1500       2000
00006  123456789912345 1000000    500000     700000
00007  222222222222222 300        150        200
.


    USE FUND
. LIST
00001  KI   R.O.K.
00002  FY   U.S.A.
.
```

```
USE IUQ
. LIST
00001    123456789123456  1978          1000000
00002    123456789123456  1258          1950000
00003    123456789234567  1978              100
00004    123456789234567  1258               80
00005    123456789456769  5274           777777
00006    123456789567891  5678            12345
00007    123456789567891  5334           160000
00008    123456789789123  8756             2800
00009    123456789789123  8970             2900
00010    123456789912345  8756           900000
00011    123456789912345  8970           864000
.
```

90

```
***********************MAIN.PRG   12/9/84********************
*AUTHOR: H. Y. LEE
*PURPOSE: This program is to display the entire menu for
*          selecting major function by user.
*CALLED BY:none
*PROGRAMS CALL: SIGNCN,ADD,DELETE,UPDATE,QUERY,REPORT,HELP.
*LOCAL VARIABLE: MAJORMENU,MOREA,ANS
SET TALK OFF
* clear screen and display sign on message
ERASE
DO SIGNCN
* Set up a loop for the user to select which major
* function to be done.
STORE T TO MOREA
*to set a main loop
DO WHILE MOREA
@ 2,2
  TEXT
      These are major menu for the logistic database system.
      And allows you to add,delete,update,query,report, and
      and help. Please choose a number for the function
      which you want to perform.

          0  quit the operation.
          1  add new item or any data.
          2  delete any data.
          3  update for transaction data.
          4  query some information.
          5  produce periodic reports.
          6  need help.
```

91

```
     ENDTEXT
  ACCEPT "Please enter the selected number here "
          TO MAJORMENU
* call the selected case statement function.
     DO CASE
          CASE   MAJORMENU ="0"
                 USE
                 ERASE
                 RELEASE ALL
                 RETURN
          CASE   MAJORMENU ="1"
                 DO ADD
          CASE   MAJORMENU ="2"
                 DO DELETE
          CASE   MAJORMENU ="3"
                 DO UPDATE
          CASE   MAJORMENU ="4"
                 DO QUERY
          CASE   MAJORMENU ="5"
                 DO REPORT
          CASE   MAJORMENU ="6"
                 DO HELP
          OTHERWISE
          @ 10,5 SAY "Please, check the entered menu and"
          @ 11,5 SAY "If not 0---6 then try again"
     ENDCASE
* Loop back again
ACCEPT "Do you want another menu (y/n)? " TO ANS
IF !(ANS) = "Y"
     STORE T TO MOREA
ELSE
     STORE F TO MOREA
ENDIF
ENDDO
QUIT
```

92

```
*******************SIGNON.PRG   12/9/84*******************
*AUTHOR: H. Y. LEE
*PURPOSE:This program is to display the system anouncement.
*CALLED BY: Main
*PROGRAMS CALL: none
*LOCAL VARIABLE: none
ERASE
?
?
?
?
?
?
?"                    LOGISTICS SUPPORT SYSTEM"
?
?"              FOR 2nd LOGISTICS   SUPPORT COMMAND"
?
?
?
?
?
?
?
?
?
?
?
?
?"**********   press any key if you ready   **********"
?
SET CONSOLE OFF
WAIT
SET CONSOLE ON
```

```
********************ADD.PRG   12/9/84********************
*AUTHOR:H. Y. LEE
*PURPOSE: This will add new records which the user want to
          the current file.
*CALLED BY:main.
*PROGRAMS CALL: additem,addunit,addsupplier,addfund,
                addstocktable,addiuq,help.
*LOCALVARIABLE: addmenu.
*FILES USED: none
SET TALK OFF
*to clear the screen
ERASE
DO WHILE T
 @ 2,2
    TEXT
        This is the add menu. It permits you to add new
        records which youwant. Please enter the number
        of the function you wish to perform.

            0  quit
            1  add item record
            2  add unit record
            3  add supplier record
            4  add fund record
            5  add stock table record
            6  add item unit quantity record
            7  need help.

        Enter the selected number
    ENDTEXT
    WAIT TO ADDMENU
    ?" "
    ?" "
*case statement to perform the user's choice.
```

```
DO CASE
    CASE ADDMENU = "0"
        USE
        ERASE
        RELEASE ALL
        RETURN
    CASE ADDMENU = "1"
        DO ADDITEM
    CASE ADDMENU = "2"
        DO ADDUNIT
    CASE ADDMENU = "3"
        DO ADDSUPPLIER
    CASE ADDMENU = "4"
        DO FUND
    CASE ADDMENU = "5"
        DO ADDST
    CASE ADDMENU = "6"
        DO ADDIUQ
    CASE ADDMENU = "7"
        DO HELP
    OTHERWISE
        @ 10,5 SAY "pLEASE CHECK YOUR CHOCE!"
  ENDCASE
*loop back again


******************ADDITEM.PRG   12/9/84*****************
*AUTHOR: H.Y.LEE
*PURPOSE: This program will add new item record to current
*         item file.
*CALLED BY: ADD
*PROGRAMS CALL: ITEM.FMT
*LOCAL VARIABLES: ANSW,OK.
*FILES USED: ITEM.DBF
USE ITEM
```

AD-A155 931   THE DEVELOPMENT OF A STANDARD DATABASE FOR REPUBLIC OF   2/2
               KOREA ARMY'S LOGISTICS SUPPORT SYSTEM(U) NAVAL
               POSTGRADUATE SCHOOL MONTEREY CA   H Y LEE ET AL. MAR 85
UNCLASSIFIED   F/G 9/2   NL

1·0

2·8

2·5

5·0

3·15

2·2

3·5

1·1

4·0

2·0

4·5

1·8

1·25

1·4

1·6
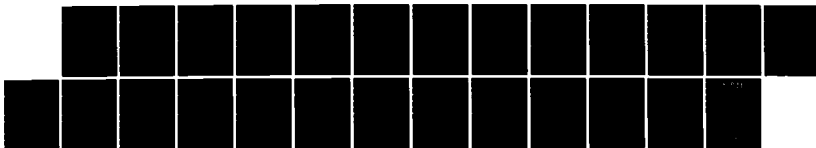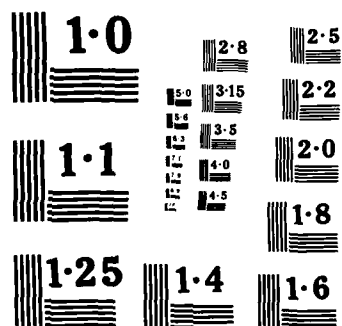
NATIONAL BUREAU OF STANDARDS
MICROCOPY RESOLUTION TEST CHART

```
*set the loop
STORE  T  TO ANSW
DO  WHILE ANSW
   APPEND BLANK
   STORE  F  TO  OK
   DO WHILE .NOT. OK
      ERASE
      SET FORMAT TO ITEM.FMT
      READ
      SET FORMAT TO SCREEN
      @ 15,2 SAY "iS IT CORRECT?(Y/N)"  GET  OK
      READ
   ENDDO
   @ 18,2 SAY"Do you want to add another record(Y/N)?"
   GET ANSW
   READ
ENDDO
*to return to the called routine
USE
ERASE
RELEASE ALL
RETURN


********************ADDUNIT,PRG  12/5/84********************
*AUTHOR : H.Y.LEE
*PURPOSE : This program will add new UNIT record
*CALLED BY : ADD
*PROGRAMS CALL : UNIT.FMT
*FILES USED : UNIT.LBF
*LOCAL VARIABLES : ANSWER, OK.
SET TALK OFF
USE UNIT
STORE T TO ANSWER
DO WHILE ANSWER
```

```
     APPEND BLANK
     STORE F TO OK
     DO WHILE .NOT. OK
          ERASE
          SET FORMAT TO UNIT
          READ
          SET FORMAT TO SCREEN
          @ 21,0 SAY "Is it correct(y/n)?" TO OK
          READ
     ENDDO
     @ 22,0 SAY "Do you want to add another record(y/n)?"
     TO ANSWER
     READ
ENDDC
USE
ERASE
RELEASE ALL
RETURN


**************ADDSUPPLIER.PRG   12/9/84*****************
*AUTHOR: H.Y.LEE
*PURPOSE: This program will add new supplier record.
*CALLED BY: ADD.PRG
*PROGRAMS CALL: SUPPLIER.FMT
*LOCAL VARIABLES:ANSWER,OK
*FILES USED: SUPPLIER.DBF
SET TALK OFF
USE SUPPLIER
STORE T TO ANSWER
*SET LOCP TO ADD RECCRD
DO WHILE ANSWER
  APPEND BLANK
  STORE F TO OK
  DO WHILE .NOT. OK
```

```
            SET FORMAT TO SUPPLIER.FMT
            READ
            SET FORMAT TO SCREEN
            @ 15,2 SAY "Is it correct(y/n)?" GET OK
            READ
        ENDDO
        @ 19,2 SAY "Do you want to add another record(Y/N)?"
        GET ANSWER
        READ
    ENDDO
    USE
    ERASE                                    .
    RELEASE ALL
    RETURN


*******************ADDFUND.PRG   12/7/84*****************
*AUTHOR: H.Y.LEE
*PURPOSE: This program will add new fund record
*CALLED BY: ADD
*PROGRAMS CALL: FUND.FMT
*LOCAL VARIABLES: ANSWER.OK
*FILES USED: FUND.DBF
SET TALK OFF
USE FUND
*SET THE LOOP TO ADD
STORE T TO ANSWER
DO WHILE ANSWER
    APPEND BLANK
    STORE F TO OK
    DO WHILE .NOT. OK
        SET FORMAT TO FUND.FMT
        READ
        SET FORMAT TC SCREEN
        @ 14,2 SAY "Is it correct(y/n)?" GET OK
```

98

```
        READ
      ENDDO
      a 19,2 SAY "Do you want to add another record(y/n)"
      GET ANSWER
      READ
ENDDO
USE
ERASE
RELEASE ALL
RETURN


*****************ADDST.PRG   12/6/84********************
*AUTHOR: W.D.SONG
*PURPOSE: This program will add new stock table record.
*CALLED BY:ADD
*PROGRAMS CALL:ST.FMT
*LOCAL VARIABLE: ANSWER,OK
SET TALK OFF
USE STOCKTABLE
*set loop to add
STORE T TO ANSWER
DO WHILE ANSWER
    APPEND BLANK
    STORE F TO OK
    DO WHILE .NOT. OK
    SET FORMAT TO STOCKTABLE.FMT
    READ
    SET FORMAT TO SCREEN
    a 15,2 SAY"Is it correct(y/n)?" GET  OK
    READ
    ENDDO
    a 19,2 SAY "Do you want to add another(y/n)?"
    GET ANSWER
    READ
```

99

```
ENDDO
USE
ERASE
RELEASE ALL
RETURN


******************ADDIUQ.PRG 12/7/84********************
*AUTHER:W.D.SONG
*PURPOSE: This program will add new item unit quantity
*record.
*CALLED BY: ADD
*PROGRAMS CALL: IUQ.FMT
*LOCAL VARIABLES: ANSWER,OK
*FILES USED: IUQ.DBF
SET TALK OFF
*set loop to add
STORE T TO ANSWER
DO WHILE ANSWER
    APPEND BLANK
    STORE F TO OK
    DO WHILE .NOT. OK
        SET FORMAT TO IUQ.FMT
        READ
        SET FORMAT TO SCREEN
        @ 15,2 SAY"Is it correct(y/n)?" GET OK
        READ
    ENDDO
    @ 19,2 SAY " Do you want to add another record(y/n)?"
    GET ANSWER
    READ
ENDDO
USE
ERASE
RELEASE ALL
```

```
RETURN


*****************DELETE.PRG   12/9/84*******************
*AUTHOR: H.Y.LEE
*PURPOSE: This program will delete records which the user
*          want.
*CALLED BY: MAIN.PRG
*PROGRAMS CALL:
*LOCAL VARIABLES: MORE1,MORE2,TEMP1,TEMP2,TEMP3
*FILES USED: ALL FILES
SET TALK OFF
*set the loop to perform DELETE
STORE T TO MORE1
DO WHILE MORE1
    ERASE
    @ 2,2 SAY "Which file do you want to delete?
             Select one!"
    @ 3,2 SAY "FILE NAME TO SELECT: ITEM,UNIT,SUPPLIER,FUND
                STOCKTABLE,IUQ"
    ACCEPT"Be careful spelling! and put here the selected
           name" TO TEMP1
    USE &TEMP1
    DISPLAY ALL
    *to choose the record number which want to delete.
    ?"which record do you want to delete?  e.g. 11 "
    ACCEPT "Please put selected number here:" TO TEMP2
    GOTO VAL(TEMP2)
    DISPLAY
    STORE " " TO TEMP3
    @ 21,0 SAY"Are you sure to delete this record(y/n) "
             GET TEMP3
    READ
       IF !(TEMP3) = "Y"
           DELETE
```

101

```
          ENDIF
     *If the user need more to delete
     STORE " " TO MORE2
     @ 22,0 SAY"Do you want delete another ? (y/n)"
               GET MORE2
     READ
        IF !(MORE2) = "Y"
           STORE T TO MORE1
        ELSE
           STORE F TO MORE1
        ENDIF
 ENDDO
 PACK
 USE
 RELEASE ALL
 ERASE
 RETURN


 ****************UPDATE.PRG   12/9/84********************
 *AUTHOR:H.Y.LEE
 *PURPOSE: This program will update(change) each record
           with a new data.
 *CALLED BY: MAIN.PRG
 *PROGRAMS CALL:UPQTY.PRG
 *LOCAL VARIABLE:TITIE,RNUM,ANSWER,MORE
 *FILES USED: ALL FILES
 SET TALK OFF
 *set the loop to execute DELETE
 STORE T TO MORE
 DO WHILE MORE
     ERASE
     @ 2,2
        TEXT
           Which file do you want to update?
```

102

```
            file name : ITEM, UNIT, SUPPLIER, FUND,
                        STOCKTABLE,IUQ
             *Please be careful with spelling!
     ENDTEXT
     ACCEPT"PUT HERE THE SELECTED FILE NAME :" TO TITLE
     *to clear screen
     ERASE
     USE &TITLE
     DISPLAY ALL
     ACCEPT "Put here RECORD NUMBER which you want to
             update:" TO RNUM
     GOTO VAL(RNUM)
     *to set up moderate file format to UPDATE(CHANGE)
      the data.
         IF TITLE ="ITEM"
             SET FORMAT TO ITEM.FMT
         ENDIF
         IF TITLE ="UNIT"
             SET FORMAT TO UNIT.FMT
         ENDIF
         IF TITLE ="SUPPLIER"
             SET FORMAT TO SUPPLIER.FMT
         ENDIF
         IF TITLE = "FUND"
             SET FORMAT TO FUND.FMT
         ENDIF
         IF TITLE = "STOCKTABLE"
             SET FORMAT TO STOCKTABLE.FMT
         ENDIF
         IF TITLE ="IUQ"
             SET FORMAT TO IUQ.FMT
         ENDIF
     READ
     SET FORMAT TO SCREEN
     ACCEPT   "Is this correct(y/n):" TO  ANSWER
```

```
      * to update the quantity on hand
      DO UPQTY
      IF !(ANSWER) = "Y"
          STORE F TO MORE
      ELSE
   STORE T TO MORE
      ENDIF
ENDDO
ERASE
RELEASE ALL
RETURN


*********************REPORT.PRG   12/9/84*****************
*AUTHOR:H.Y.LEE
*PURPOSE: This program will select report menu to produce.
*CALLED BY:MAIN.PRG
*PROGRAMS CALL:BNQTY.PRG, BNASSET.PRG, TOTQTY.PRG
*LOCAL VARIABLES:MORE, ANSWER, NEED
*FILES USED:none
SET TALK OFF
*CLEAR SCREEN
ERASE
STORE T TO MORE
DO WHILE MORE
    @ 2,2
       TEXT
       This is the menu to make reports. Please choose one.

           0   quit
           1   on hand quantity of item by each battalion
           2   status of asset of item by each battalion
           3   total quantity by each item
           4   need help
       ENDTEXT
       ACCEPT "Enter your selection here:"  TO ANSWER
```

104

```
* CASE statement to perform the selection
  DC CASE
        CASE  ANSWER = "0"
              USE
              ERASE
              RELEASE ALL
              RETURN
        CASE  ANSWER = "1"
              DO BNQTY
        CASE  ANSWER = "2"
              DO BNASSET
        CASE  ANSWER = "3"
              DO TOTQTY
        OTHERWISE
              @ 3,3 SAY "PLEASE CHECK YOUR CHOICE!"
  ENDCASE
  ACCEPT "Do you want another REPORT(Y/N)?" TO NEED
  IF !(NEED) = "Y"
        STORE T TC MORE
  ELSE
        STORE F TC MORE
  ENDIF
*loop back again
ENDDO
ERASE
RELEASE ALL
RETURN


********************BNQTY.PRG   12/9/84********************
*AUTHOR:H.Y.LEE
*PURPOSE: This program will produce report which includes
*         on hand quantityof item by each battalion.
*CALLED BY: REPORT.PRG
*PROGRAMS CALL:none
```

```
*LOCAL VARIABLE:TEMP1, TEMP2,PRINTER,RET,CONT
*FILES USED:ITEM.DBF, IUQ.DBF, UNIT.DBF
* set loop
STORE T TO RET
DO WHILE RET
    SET TALK OFF
    *to print or not
    ACCEPT "Do you want to print out(y/n)?" TO PRINTER
    IF !(PRINTER) = "Y"
         SET PRINT CN
    ENDIF
    ERASE
    *to make HEADING
    ?
    ?"          STATUS OF ITEM QUANTITY"
    ?"          ========================"
    ?
    ?
    ?
    ?"UNIT STOCK-NUMBER NOMENCLATURE QUANTITY   UI"
    ?"------------------------------------------------ "
    *This routine will make the needed data
     USE IUQ
     SELECT SECONDARY
     USE UNIT
     JOIN TO TEMP1 FOR P.U:ID = S.U:ID   FIELD U:NAME,SN,QTY
     USE TEMP1
     SELECT SECONDARY
     USE ITEM
     JOIN TO TEMP2 FOR P.SN = S.SN   FIELD U:NAME,SN,NM,QTY
                                                     UP
     USE TEMP2
     LIST
    *back to the called routine
     ACCEPT"   Do you want to return to called routine
```

```
                    (y/n) ?" TO CONT
    IF !(CONT) = "Y"
         STORE F TO RET
    ELSE
         STORE T TO RET
    ENDIF
   *to release printer
    SET PRINT OFF
ENDDO
   USE
   ERASE
   RELEASE ALL
   RETURN



****************BNASSET.PRG   12/6/84 ******************
*AUTHOR:W.L.SONG
*purpose:This will  produce report which include ASSET of
         each item by unit.
*CALLED BY: REPORT.PRG
*PROGRAMS CALL:
*LOCAL VARIABLE:MORE,TEMP1,TEMP2,ATOT,PRINTER,ANSWER
*FILES USED:UNIT.DBF, IUQ.DBF, ITEM.DBF
SET TALK OFF
*set the loop
STORE T TO MORE
DO WHILE MORE
   * set up print
    ACCEPT"Do you want to print(y/n)?" TO PRINTER
    IF !(PRINTER) ="Y"
         SET PRINT ON
    ENDIF
    ERASE
   * make heading
    ?
```

```
?"              ASSET STATUS OF ITEMS BY BATTALION"
?"              ======================================"
?
?
?
?"UNIT   NOMENCLATURE   QUANTITY   UNIT-PRICE    SUB-TOTAL"
?"------------------------------------------------------------"
*next routine produce neceassary colum and make total price
USE IUQ
SELECT SECONDARY
USE UNIT
JOIN TO TEMP1 FOR P.U:ID = S.U:ID   FIELD U:NAME,SN,QTY
USE TEMP1
SELECT SECONDARY
USE ITEM
JOIN TO TEMP2 FOR P.SN = S.SN   FIELD U:NAME,SN,NM,QTY,UP
USE TEMP2
STORE 0 TC ATOT
   DO WHILE .NOT. EOF
        ? U:NAME,NM,QTY,UP,QTY*UP
        STORE ATOT+QTY*UP TO ATOT
        SKIP
   ENDDO
?"THE TOTAL ASSET IS:--------------------------->",ATOT
*query for continuing this report again or not
ACCEPT"Do you want to make again(y/n)?" TO ANSWER
IF !(ANSWER) = "Y"
   STORE T TO MORE
ELSE
   STORE F TO MORE
ENDIF
*to release printer
SET PRINT OFF
ENDDO
*the end of program
```

108

```
USE
ERASE
RELEASE ALL
RETURN


******************TOTQTY.PRG   12/8/84******************
*AUTHOR:W.D.SONG
*PURPOSE: This program will provide a report to user the
*          total quantityon hand of each item.
*CALLED BY: REPORT.PRG
*PROGRAMS CALL:
*LOCAL VARIABLE:MORE,PRINTER,ANSWER
*FILES USED:ITEM.DBF
SET TALK OFF
*set loop
STORE T TO MORE
DO WHILE MORE
    * to make print
     ACCEPT "Do you want to print out(y/n)?" TO PRINTER
    IF  !(PRINTER) = "Y"
       SET PRINT ON
    INDIF
    * clean the screen
    ERASE
    *to make heading
    ?
    ?"     THE STATUS OF QUANTITY ON HAND BY EACH ITEM"
    ?"     ========================================="
    ?
    ?
    ?
    ?" STOCK-NUMBER  NOMENCLATURE  UNIT-PRICE   QUANTITY"
    ?"------------------------------------------------------"
    *produce each data list by item which indexed by stock
```

```
  *number
   USE ITEM
   INDEX ON SN TO ITEMSN
   USE ITEM INDEX ITEMSN
   LIST SN,NM,UP,QTYCH
  *to return to the called routine
   ACCEPT "Do you want again(y/n)?"  TO ANSWER
   IF !(ANSWER) = "Y"
       STORE T TO MCRE
   ELSE
       STORE F TO MORE
   ENDIF
  *to release the printer
   SET PRINT OFF
ENDDO
*the end of program
USE
ERASE
RELEASE
RETURN


*********************QUERY.PRG  12/8/84*******************
*AUTHOR: H.Y.LEE
*PURPOSE: provide the status of each file and requested
*         by user.
*CALLED BY: MAIN.PRG
*PROGRAMS CALL: MINQTY.PRG
*LOCAL VARIABLES :MORE, ANSWER,QMENU
*FILES USED:ALL FILES
SET TALK OFF
*Set up a loop for the user to select query which the user
 want.
STORE T TO MORE
DO WHILE MORE
```

```
ERASE
@ 2,2
TEXT
    This is quiry menu which prepared for you in this
    systems.
        0   quit
        1   item list(stock-number, nomencleture, unit-issue,
            unit-price,order-shipping-time, quantity-on-hand,
            supplier-id)
        2   unit(unitname, phone, address, city, zip)
        3   supplier(country, company, location, phone)
        4   fund(source of fund)
        5   stock-table(request-objective, safety-level,
                        reorder-point)
        6   quantity of item by each unit
        7   the item list which on hand quantity is less
            than reorder point
    ENDTEXT
ACCEPT " Enter the selection here !:" TO QMENU
*Execute the selected query  case function.
ERASE
DO CASE
        CASE QMENU = "0"
                USE
                ERASE
                RELEASE AIL
                RETURN
        CASE QMENU = "1"
                USE ITEM
                LIST
        CASE QMENU = "2"
                USE UNIT
                LIST
        CASE QMENU = "3"
                USE SUPPLIER
```

```
                    LIST
          CASE QMENU = "4"
                USE FUND
                LIST
          CASE QMENU = "5"
                USE STOCKTABLE
                LIST
          CASE QMENU = "6"
                USE IUQ
                LIST
          CASE QMENU = "7"
                DO MINQTY
          OTHERWISE
                ?"Please check the selected number!"
      ENDCASE
* to continue or not?
ACCEPT"Do you want to another query(y/n)?" TO ANSWER
      IF !(ANSWER) = "Y"
            STORE T TO MCRE
      ELSE
            STORE F TO MORE
      ENDIF
ENDDO
USE
ERASE
RELEASE ALL
RETURN


*******************MINQTY.PRG 12/7/84*******************
*AUTHOR:W.D.SONG
*PURPOSE: This program will produce the list of item which
*         on hand quantity is less than reorder point.
*CALLED BY: QUERY.PRG
*PROGRAMS CALL: none
```

112

```
*LOCAL VARIABLE: WHEN,TEMP1,TEMP2,ANSWER,PRINTER
*FILES USED: IUQ.DBF, STOCKTABLE.DBF, ITEM.DBF
SET TALK OFF
*set up loop
STORE T TO WHEN
DO WHILE WHEN
    *set printer
    ACCEPT "Do you want to print out(y/n)?" TO PRINTER
    IF !(PRINTER) = "Y"
        SET PRINT ON
    ENDIF
*set the routine to find answer
USE IUQ
SELECT SECONDARY
USE STOCKTABLE
JOIN TO TEMP1 FOR P.SN = S.SN FIELD  P.SN,U:ID, QTY,
                        RDRPNT
USE TEMP1
SELECT SECONDARY
USE ITEM
JOIN TO TEMP2 FOR P.SN = S.SN  FIELD P.SN,NM,U:ID,QTY,
                        RDRPNT
USE TEMP2
sort on stock number
INDEX ON SN TO TEMP2SN
USE TEMP2 INDEX TEMP2SN
DO WHILE .NOT. EOF
    DISPLAY SN,NM,U:ID,QTY,RDRPNT FOR QTY < VAL(RDRPNT)
    SKIP
ENDDO
*to return after query
ACCEPT"Do you want to return to called routine(y/n)?"
        TO ANSWER
IF !(ANSWER) = "Y"
   STORE F TO WHEN
```

```
ELSE
   STORE T TO WHEN
ENDIF
ENDDO
USE
ERASE
RELEASE
RETURN


*********************HELP.PRG   12/7/84******************
*AUTHOR: W.D.SONG
*PURPOSE: This program will provide to user needed
*         information to use thismore effectively.
*CALLED BY:ALL THE PROGRAMS
*to clear screen
ERASE
?
?
?
?
?
?"        Please   ask   to :   Maj.    H.Y.  LEE"
?"                               Maj.    W.D.  SONG"
?
?
?
?
?"                               Press any key to continue!"
?
?
SET CONSCLE OFF
WAIT
SET CONSCLE ON
*************************************************************
```

APPENDIX H
REPORT STATUS EXAMPLES

STATUS OF ITEM QUANTITY
==========================

UNIT	STOCK-NUMBER	NOMENCLATURE	QUANTITY	UNIT-ISSUE
00001	15ORD 123456789123456	M16A1RIFLE	1666666	111.22
00002	20ORD 123456789123456	M16A1RIFLE	1950000	111.22
00003	15ORD 123456789234567	105M-MOTOR	100	2222.00
00004	20ORD 123456789234567	105M-MOTOR	80	2222.00
00005	51AMM 123456789456789	M16AMMUNITION	777777	205.66
00006	55AMM 123456789567891	105MAMMUNITION	160000	40.40
00007	11QTR 123456789789123	COMBAT-SHOSE	2800	555.88
00008	15QTR 123456789789123	COMBAT-SHOSE	2900	555.88
00009	11QTR 123456789912345	MO-GAS	900000	220.00
00010	15QTR 123456789912345	MO-GAS	864800	220.00

Do you want to return to called routine(y/n)?:Y

ASSET STATUS OF ITEMS BY BATTALION
===================================

UNIT	NOMENCLATURE	QUANTITY	UNIT-PRICE	SUB-TOTAL
15ORD	M16A1RIFLE	1666666	111.22	185366592.50
20ORD	M16A1RIFLE	1950000	111.22	216879000.00
15ORD	105M-MOTOR	100	2222.00	222200.00
20ORD	105M-MOTOR	80	2222.00	177760.00
51AMM	M16AMMUNITION	777777	205.66	159957617.80
55AMM	105MAMMUNITION	160000	40.40	6464000.00
11QTR	COMBAT-SHOSE	2800	555.88	1556464.00
15QTR	COMBAT-SHOSE	2900	555.88	1612052.00
11QTR	MO-GAS	900000	220.00	198000000.00
15QTR	MO-GAS	864800	220.00	190256000.00

THE TOTAL ASSET IS:------------------------------> 960491686.30

Do you want to make again(y/n)?:N

115

APPENDIX I
QUERY EXAMPLES

These are major menu for the logistic database system. And allows you
to add,delete, update, query, report, and help. Please choose a number
for the function which you want to perform.

    0 quit the operation.
    1 add new item or any data.
    2 delete any data.
    3 update for transaction data.
    4 query some information.
    5 produce periodic reports.
    6 need help.
Please enter the selected number here :4

This is quiry menu which prepared for you in this systems.
    0 quit
    1 item list(stock-number, nomencleture, unit-issue,unit-price,
              order-shipping-time, quantity-on-hand, supplier-id)
    2 unit(unitname, phone, address, city, zip)
    3 supplier(country, company, location, phone)
    4 fund(source of fund)
    5 stock-table(request-objective, safety-level, reorder-point)
    6 quantity of item by each unit
    7 the item list which on hand quantity is less than reorder point
Enter the selection here 1::3

```
00001  PUNG-SAN   KOREA      PUNG-SAN AMM-CO  MA-SAN       22-66-2345
00002  DAE-WOO    KOREA      DAE-WOO CORP.    CHANG-WON    55-33-7865
00003  RMT        U.S.A.     REMINGTON        DETROITE     233-444-7777
00004  DAE-WHA    KOREA      DAE-WHA CORP.    DAE-JEON     44-77-7336
00005  KYUNG-IN   KOREA      KYUNG-IN ENERGE  IN-CHUN      23-45-3673
```
Do you want to another query(y/n)?:Y

This is quiry menu which prepared for you in this systems.
    0 quit
    1 item list(stock-number, nomencleture, unit-issue,unit-price,
              order-shipping-time, quantity-on-hand, supplier-id)
    2 unit(unitname, phone, address, city, zip)
    3 supplier(country, company, location, phone)
    4 fund(source of fund)
    5 stock-table(request-objective, safety-level, reorder-point)
    6 quantity of item by each unit
    7 the item list which on hand quantity is less than reorder point
Enter the selection here 1::7

Do you want to print out(y/n)?:Y

```
00005  123456789456789 MIGAMMUNITION   5274     777777 120000
00006  123456789567891 1U5MAMMUNITION  5678      12345 120000
```
Do you want to return to called routine(y/n)?:Y

Do you want to another query(y/n)?:N
Do you want another menu (y/n)? :N
*** END RUN   dBASE II   ***

116

LIST OF REFERENCES

1. Kronke, D., _Database Processing_, Science Research Associate Inc., Chicago, Tronto, 1983.

2. Date, C. J., _An Introduction to Database Systems_, Third Edition, Addition-Wesley Publishing Inc., 1981.

3. Martin, J., _Computer Data-base Organization_, Prince-Hall Inc., Englewood Cliffs, N.J., 1977.

4. Hsiao, D. K., "Data Base Machines Are Coming, Data Base Machines Are Com ing!", _IEEE Computer Society_, Computer 12, No. 3, March 1979.

5. Ullman, I. D., _Principles of Database Systems_, Computer Science Press Inc., Rockvill Maryland 20850, 1982.

6. Date, C. J., _An Introduction to Database Systems_, The Systems Programming Series, Volume II, Addition-Wesley Publishing Com pany Inc., 1983.

7. Tsichritzis, D. C. and Lochovsky, F. H., _Data Models_, Printice-Hall Inc., Inglewood Cliffs, N.J., 07632, 1982.

8. BCS(1977), "The British Computer Society Data Dictionary Systems Working Party Record.", _ACM SIGMOD Record 9(4)_, PP. 2-24.

9. _The Design of an Integrated Data Dictionary System_, Requirements and Logical Structures Proceedings, N.Y., May 1978, Edited b y S.B.Yao, S.B.Navathe, J.L.Wellon, and T.L.Kunil, PP. 56-71.

10. Berri, C., Bernstein, P.A., and Goodman, N., 1978, "A Sophiscate's Introduction to Database Normalization Theory" _Proceedings of the Fourth International Conference on Very Large Data base_, West Berlin, PP. 113-124.

11. Hawryszkiewycz, I.T., _Database Analysis and Design_, Science Research Associates Inc., 1984.

12. Codd, E.F., "Relational Database: A Practical Foundation for Productivity". In _Communications of the ACM_, Vol. 25, No. 2, Feb. 1982.

13.   Adam, B. Green, dBASE II User's Guide, SoftwareBank
       Inc., Arlington, MA 02174, 1983.

14.   Adam, B. Green, Advanced dBASE II User's Guide,
       Prentice Hall Inc., Englewood Cliffs, N.J. 07632,
       1984.

15.   Martin, J., An End-User's Guide to Database,
       Prentice-Hall Inc., 1981

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Department Chairman, Code 54 Dept. of Administrative Sciences Naval Postgraduate School Monterey, California 93943	2
4.	Professor Norman R. Lyons Code 54Lb Naval Postgraduate School Monterey, CA 93943	1
5.	Professor Carl Jones Code 54Js Naval Postgraduate School Monterey, CA93943	1
6.	Central Computer Center Army Headquarters, 140-01 Seoul, Republic of Korea	1
7.	Department of Logistics Management Army Headquarters, 140-01 Seoul, Republic of Korea	1
8.	Lee, Hee Young 16-5 Cheon-ho Dong Kang-dong Ku Seoul, Republic of Korea	8
9.	Song, Wha Dal 494-4 Sin-lim 4 Dong Kawn-ak Ku Seoul, Repulic of Korea	8
10.	Lee, Jang Hun SMC 1306 Naval Postgaduate School Monterey, CA 93943	1
11.	Jeong, Jee Ho SMC 1440 Naval Postgraduate School Monterey, CA 93943	1
12.	Park, In Seop SMC 2886 Naval Postgraduate School Monterey, CA 93943	1

END

FILMED

8-85

DTIC