1988

Development of a personnel database system for watch scheduling on Hellenic Navy ships

Elefsiniotis, Dimitrios A.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DEVELOPMENT OF A PERSONNEL DATABASE SYSTEM
FOR WATCH SCHEDULING ON HELLENIC
NAVY SHIPS

by

Dimitrios A. Elefsiniotis

September 1988

Thesis Co-Advisors: Samuel H. Parry
Vincent Y. Lum

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release;
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	Distribution is unlimited
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) Code 55PY	7a NAME OF MONITORING ORGANIZATION Naval Postgra uate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey,California 93943-5000
8a NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
DEVELOPMENT OF A PERSONNEL DATABASE SYSTEM FOR WATCH SCHEDULING ON HELLENIC NAVY SHIPS.

12 PERSONAL AUTHOR(S)
Elefsiniotis Dimitrios

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM _____ TO _____	1988 September	204

16 SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Goverment.

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	DatAbase system design and implementation; Decision support system

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This thesis represents the development of a database system for personnel management and watch scheduling on the Hellenic Navy ships. DBase III plus is used as a "Database Management System" to implement the main task,duties assignment in the administration office of a war ship. The implementation is a collection of algorithms that provide intelligent decision support to users of the system about the assignment duties. It is designed to run on an IBM PC XT-TURBO microcomputer.

The system is developed to support all the administrative tasks and activities and to provide real time decision information on the crew allocations.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL Prof. Samuel H. Parry ; Prof. V.Y.Lum	22b TELEPHONE (Include Area Code) (408) 646-2768 22c OFFICE SYMBOL Code 55PY

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

Approved for public release, distribution unlimited

Development of a Personnel Database System
for watch scheduling on Hellenic
Navy Ships.

by

Dimitrios Elefsiniotis
Lieutenant, Hellenic Navy
B.A, Hellenic Naval Academy, 1977

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

ABSTRACT

This thesis represents the development of a database system for personnel management and watch scheduling on the Hellenic Navy ships. DBase III plus is used as a "Database Management System" to implement the main task, duties assignment in the administration office of a war ship. The implementation is a collection of algorithms that provide intelligent decision support to users of the system about the assignment duties. It is designed to run on an IBM PC XT TURBO microcomputer.

The system is developed to support all the administrative tasks and activities and to provide real time decision information on the crew allocations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENT

I would like to express my gratitude to Professor Samuel H. Parry of the Operation Research department and Professor Vincent Y. Lum of the Computer Science department, for their enthusiastic guidance and endless supply of ideas.

I would like, also to thank my wife Helen and my son Alex for their understanding and encouragement, and for the sacrifices they have made during my graduate study.

I. INTRODUCTION

It is obvious that now is the computer era but it takes time for a society to accept computers. Many societies still resist their use, though computers have passed the test of time and are being accepted at a rapid pace in all aspects of life.

Hardware technologies have played vital roles in our ability to use electronic properties to store and process information. The software and data processing aspects, however, have not been developed at the same rate. One of the main reasons is the novelty and the complexity of the subject.

In the beginning, computers were used to process only numerical information, but the majority of our information is nonnumeric in nature and very little is known about its description and processing. [Ref. 1:pp. 1-2]

Around 1964 a new concept appeared in the computer literature to denote the organization of nonnumerical information. This concept is named "database". In the last ten years, many information systems have been developed for storing and retrieving nonnumerical information using the database concept.

These information systems were based on the extension of operating systems which were developed for processing

numerical information. The information systems support the organization's functions maintaining the data and assisting users to interpret theses data for making decision. For making these decisions data and knowledge are required. The needed data is stored in files, and the required knowledge for processing the data is encoded in the program. The general scheme of a database system is given in Figure 1. [Ref. 2:p. 3]

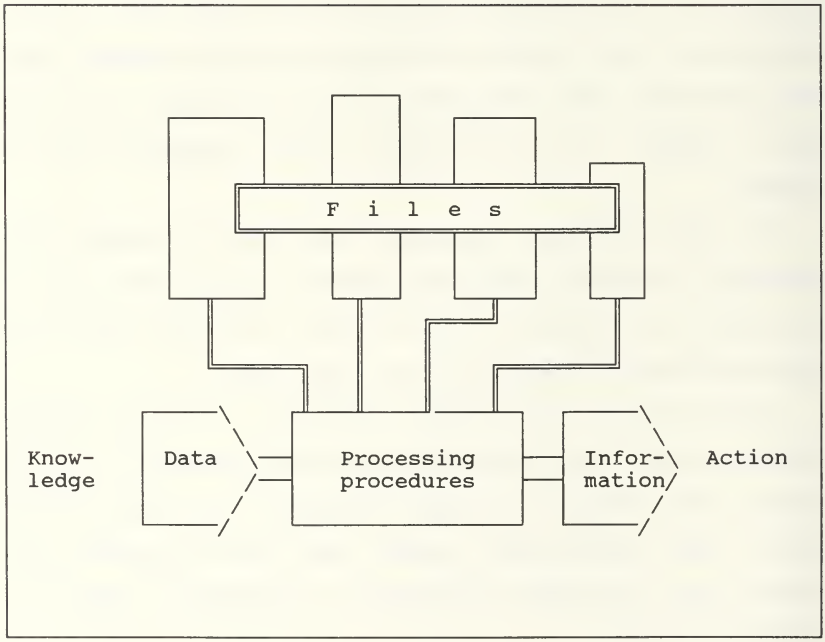


Figure 1. General database scheme.

There are three characteristics of files which distinguish them from other objects with which programs deal: [Ref. 2:p. 2]

1. "Persistence. Data written into a file persist after the program is finished. The data can be used later by the same or by other programs."

2. "Sharability. Data stored on external storage devices can be shared by multiple programs and by multiple users simultaneously."

3. "Size. Data volumes are typically greater than the capacity of the directly addressable memory of the computer."

The overall file system design process, beginning with user requirements, is the use and effectiveness of computer stored files since the best measure of a good design is the overall economic utility of the services provided by the file system. [Ref. 2:p. 2]

A. INTRODUCTION TO THE PROBLEM

Every war ship must be able to operate in different environments and under different situations. This means a variety of functions must be performed by the personnel on the ship. The personnel organization and the individual training based on the mission that each war ship should be able to perform depend on some standards that each country establishes.

The management of this personnel organization is a difficult and time-consuming job, in spite of the availability of the tables to define the duties of the crewmembers.

The difficulty becomes bigger when we have to assign additional or collateral duties since in the real time environment one or more jobs will be empty and we have to define which of them, if left unfilled, will not be a detriment to the smooth operation of the department and the ship.

In the Hellenic Navy only officers and non-commissioned officers serve on a permanent basis while seamen serve for a standard, short period of time. Crewmember changes are based on annual occurrences. During these annual reassignments, there is sometimes not enough time for a new crewmember to be trained in his new duties and these duties must be reassigned among the other crewmembers. Seamen changes are based on the time served. From time to time, projection tables must be assembled and sent to the Navy training centers for future needs of seamen. The new seamen must be trained first in order to assign them duties and then they must attend more training courses for efficient execution of their duties. Further, any type of leave (standard, on-sea, hospital, etc) can be permitted under certain circumstances, even during a training course or an exercise period. This depends on the crewmember duties and sometimes the service period in the Navy. Sometimes it is also necessary to reschedule the leave period. In this case, the rescheduled period must be avoided to overlap with a training course or an exercise period. Moreover, the total number of crewmembers on leave at any

instance of time must not exceed the established percentage amount by the Headquarters, and the duties of a crewmember on leave must be assigned among the other crewmembers when the leave period exceeds a certain number of days or when the crewmember's duties are important for the smooth operation of the department.

Although there are organization tables, general rules for assignments, and records of the characteristics of each crewmember, it is a very difficult and time-consuming task to assign duties to each crewmember. Thus, the assignment process involves a constant reexamination of these tables. Existing tables must be compared with the newly assembled information and reevaluation of the assignment tables requires a considerable amount of time. The Executive officer is responsible for these jobs and is assisted by the administration office staff.

The administration office, in addition, has several additional jobs. Daily duties on dock, punished crewmember lists, leaved crewmember lists, training crewmember lists, and many reports are some examples of those additional jobs. The administration office has also the responsibility to organize the crewmembers into groups according to "near base home area" for quick call back in case of emergency, to organize activities between the crewmembers on the ship or with the crewmembers of other ships, or to assign duties to personnel that come on board as visitors.

The purpose of this thesis is to design and implement the alternative of replacing the manual system in which crewmember records are used for supporting crewmember management by a computer database system that can implement the crewmember records, provide solutions to the assignment problems in real time, and help the administration office to perform the described additional jobs.

This thesis accordingly has the following objectives:

1. Present the design steps for a war ship personnel organization database system, the design criteria, and the elements of the database system which enable the designers to evaluate databases against these criteria.

2. Attempt to resolve conflicts and to make the system easy to use and helpful for making decisions by providing the necessary information.

3. Implement the designed database system which controls and executes the transactions written in Dbase III Plus on IBM PC XT TURBO.

This thesis provides the computerized personnel organization according to the Hellenic Navy standards for a medium size war ship. It can be applied with some small changes to any size of ship.

B. SUMMARY OF CHAPTERS

In this chapter, the general concepts of a database system for an application and an introduction to the administration office problem have been described. The following six chapters will include:

1. Background. Primitives concepts of a database system and data models are described in this chapter.

2. Applications. The administration office requirements will be stated in this chapter. The description of these requirements is important for the structure of the database system since they will constitute the blueprint of the design.

3. Design. Two data models will be used to show the designed database system:SDM and E-R model. The first model, SDM, will synthesize the application requirements according to a standard description, after applying the normalization in any existing anomalies the E-R diagram will present the final conceptual database system design.

4. Design support and constraints analysis. The SDM and E-R diagram will be reviwed in order to define the existing conflicts and constraints of the designed database system. The definition of these conflicts and constraints is important for the application programs.

5. Implementation. A part of the designed database system will be implemented using Dbase III plus to demonstrate the structure of the database system. Additional relations will be used in the implementation to provide friendliness to the users of the system.

6. Conclusions. The last chapter includes the conclusions of this thesis.

II. BACKGROUND

A database is a shared collection of interrelated data designed to meet the varied information needs of an organization. A database has two important properties : it is intergrated and it is shared. By integrated we mean that previously distinct data files have been logically organized to eliminate redundancy and to facilitate data access. By shared we mean that all qualified users in the organization have access to the same data for use in a variety of activities [Ref. 3:pp. 3-4]. Another simple definition of a database is that it is a collection of related data with a well defined structure. The data storage for a database is accomplished by the use of one or more files. A comprehensive database should contain all the information necessary to manage an enterprise. Less comprehensive data collections that support some part of an enterprise are also commonly called databases. [Ref. 2:p. 4]

The database approach offers a number of important and practical advantages to an organization. Reducing redundancy improves the consistency of data while reducing the waste in storage space. Sharing data often permits new data processing applications to be developed without having to create new data files. In general, less redundancy and greater sharing

lead to less confusion between organizational units and less time spent resolving errors and inconsistencies in reports. [Ref. 3:p. 3]

Databases can be implemented directly, using file management programs, or a database management system. However, using file management programs without a database management system (DBMS) will reduce the coverage and a level of detail which is relevant to the enterprise [Ref. 2:p. 3]. Thus, databases are implemented using some DBMS. If a suitable database management system is available, then much work can be saved. All files of one database must be accessible by the computer being used for processing. If the database is distributed over several computers, then its files must be accessible from any of the interconnected computers.

According to Everest (1976), "the database concept is rooted in an attitude of sharing common data resources releasing control of those data resources to a common responsible authority, and cooperating in the main tenancy of those shared data resources". [Ref. 3:p. 14]

A. DEFINITIONS FOR DATABASE TERMINOLOGY

Dealing with large quantities of data is difficult even for computers. For that reason, it is important to present some basic definitions and terminology before describing the general overview of a database system.

1. A "Database management system" (DBMS) is a collection of interrelated data and a set of application programs

to accesss that data. Therefore, a database management system includes data files and application programs to manage these files.

2. A "File" is a collection of similar records kept on computer storage devices. A file will have a name and a structure, or organization determined by a file access program.

3. A "Record" is a collection of related fields containing elemental data items.

4. A "Field" contain the bases values which comprise a record. The content of a field is provided to users or their programs on request. Fields may be related to each other because they describe some specific instance, an object, or an event. [Ref. 2:pp. 5-6]

5. A "Domain" is a semantic definition of the value, so that the meaning of the value is understood. Comparing or adding values from different domains is not meaningful.

6. A "Key" is a data item used to identify a record. There are two basic types of keys:
(a). A primary key is a data item that uniquely identifies a record and corresponds to the identifier of a real word entity.
(b). A secondary key is a data item that normally does not uniquely identify a record but identifies a number of records in a set that share the same property.

7. An "Object" is the description of an entity in the user 's work environment and is consisted of a named collection of properties. [Ref. 4:p. 90]
An "Entity" is an object that exists and is distinguishable from other objects. It may be concrete or it may be abstract. [Ref. 5:p. 21]

8. A "Data Definition Language" (DDL) is a vocabulary, and a grammar, that is used to describe a database to the DBMS.

9. A "Data Manipulation Language" (DML) is a sublanguage of a data language, consisting of a vocabulary set and a grammar, that can be used to manipulate the structures of the data set.

The terms File, Record, Field are similar to the terms Relation, Tuple, and Attribute which come from the field of mathematics and are alien to most people. Users, on the other hand, might prefer the terms Table, Row, and Column when refering to the structure of stored data. Relation, Row, and Attribute are used when discussing logical database design. [Ref. 4:p. 133]

Data are facts concerning people, objects, events, or other entities. Data can be historical or predictive; financial and quantitative ; qualitative and subjective; internal or external. This definition of data is given to distinguish it from the information which may include data that have been organized or prepared in a form that is suitable for decision making. [Ref. 3:p. 4]

B. OVERALL DATABASE SYSTEM STRUCTURE

A database system consists of a number of components. Each component is responsible for the overall system performance. The operating system of the computer supports the database system by providing some basic functions to it. This, the interface between the database and the operating system must be considered when designing a database system.The most important functional components in the database system are: [Ref. 5:pp. 15-17]

1. File manager. This manages the allocation of space storage and the data structure that used to represent this on disk.

2. Database manager. This manages the interface between the application programs and any query language that is supported by the system and low-level data stored in the database.

3. Query processor. A query statement is translated into low level database manager instructions.

4. DML precompiler. DML statements are converted to normal procedure calls in the host language.The precompiler, in order to produce the appropriate code, cooperates with the query processor.

5. DDL compiler. DDL statements are converted to a set of metadata tables. These tables are stored in the data dictionary.

These components and their connections are shown in Figure 2. [Ref. 5:p. 17]

In addition to functional components, a database system requires several data structures. These data structures are:

1. Data files. These files store the database data.

2. Data-dictionary. This is the information about the structure of the database. A good design of the data dictionary is necessary, since it is used heavily.

3. Indices. This is a facility that every database has for fast access to particular data.

C. ADVANTAGES OF DATABASE PROCESSING

A database has a number of important advantages compared to traditional file processing systems as follows: [Ref. 3:pp. 15-20]

1. Minimal Data Redundancy. Separate and redundant data files are integrated into a single, logical structure. Each data item occurrence is ideally recorded in only one place in the database. Sometimes multiple copies of the same data can be used for special requirements. However, in a database system, redundancy is controlled.

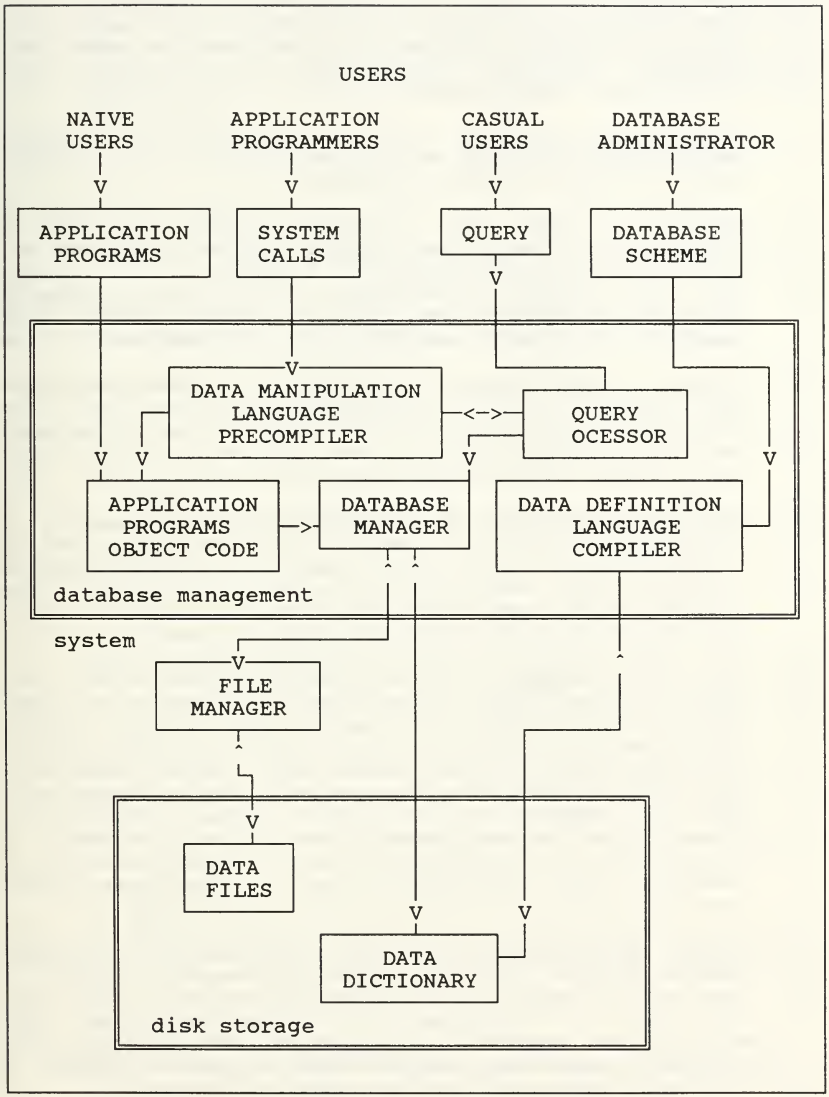


Figure 2. Database components.

2. Consistency of data. The opportunities for inconsistency are already reduced as a result of controlling data redundancy. The database system, in addition, enforces consistency by updating the appropriate data items when a change occurs.

3. Intergration of data. Data are organized with logical relationships defined between associated data entities, into a single, logical structure.

4. Sharing of data. All authorized users can share a database in a controlled manner. Each user has his own view of the database which is a subset of the conceptual data model. In this way, each user can make a decision or perform some function without having to be aware of the overall complexity of the database.

5. Ease of Application Development. Developing new applications is greatly reduced in cost and time. This is a major advantage of the database system. A programmer can code and debug a new application faster than using merely conventional data files.

6. Security. Data must be protected against accidental or intentional misuse or destruction. The DBMS provides mechanisms to assign,control, and remove the rights of access by any user to any data of the database. Data protection is very important since the amount of data shared and the number of users are increased.

7. Data independence. Data independence denotes independence or insulation of application programs or users from a wide variety of changes in the specific logical organization, physical organization, and storage considerations. Database technology tries to provide as much data independence as possible. Data independence has many degrees but there is no clear definition of these degrees since there is no industry standard to measure it. It should be stressed that there is no complete data indepedence since all possible changes cannot be predicted by an application program.

8. Access flexibility. Each item of data in a database system can be retrieved by multiple paths, giving a user more flexibility in locating and retrieving data than with a traditional file processing system. DBMS enhances the capability of parallel access of data and real-time query and update. Parallel access of data is extremely important in a database environment.

9. Reduced Program Maintenance. Maintenance refers to modifying or rewriting old programs to accept new data

formats, access methods, etc. In a database system, data are independent of the application programs that use the data and can be changed within limits without a change in the other factor. Therefore, program maintenance can be significantly reduced in a database environment.

D. DATABASE DESIGN

Database design is the process of developing database structures from user requirements for the data. The first step in the design process is the requirements analysis, which identifies user needs for data. The next step is the translation of these user requirements into first a conceptual, then a physical database design.

Teorey and Fry (1982) have developed a general model for database design, defining four major steps in the design process as shown in Figure 3: [Ref. 3:pp. 212-215]

1. Requirements formulation and analysis. The major task of this step is to identify and describe the required data for the building database system by the organization. Analysis of the requirements identifies not only what data are used, but how they are used. For that reason, this step has two major inputs: the user information requirements and the processing requirements. During the requirement analysis, the relationships among the objects and the repetition of each object is defined and the conflicts are at least recognized.

2. Conceptual design. In this step the various user views, information requirements and processing requirements are synthesized into a global database system. The design may be expressed in one of several forms described later in this chapter.
3. Implementation design. The purpose of implementation design is to map the conceptual data model into an internal model than can be processed by a particular DBMS. The conceptual data model is mapped into a data model: hierarchical, network, or relationl data model. It is then developed using the data description

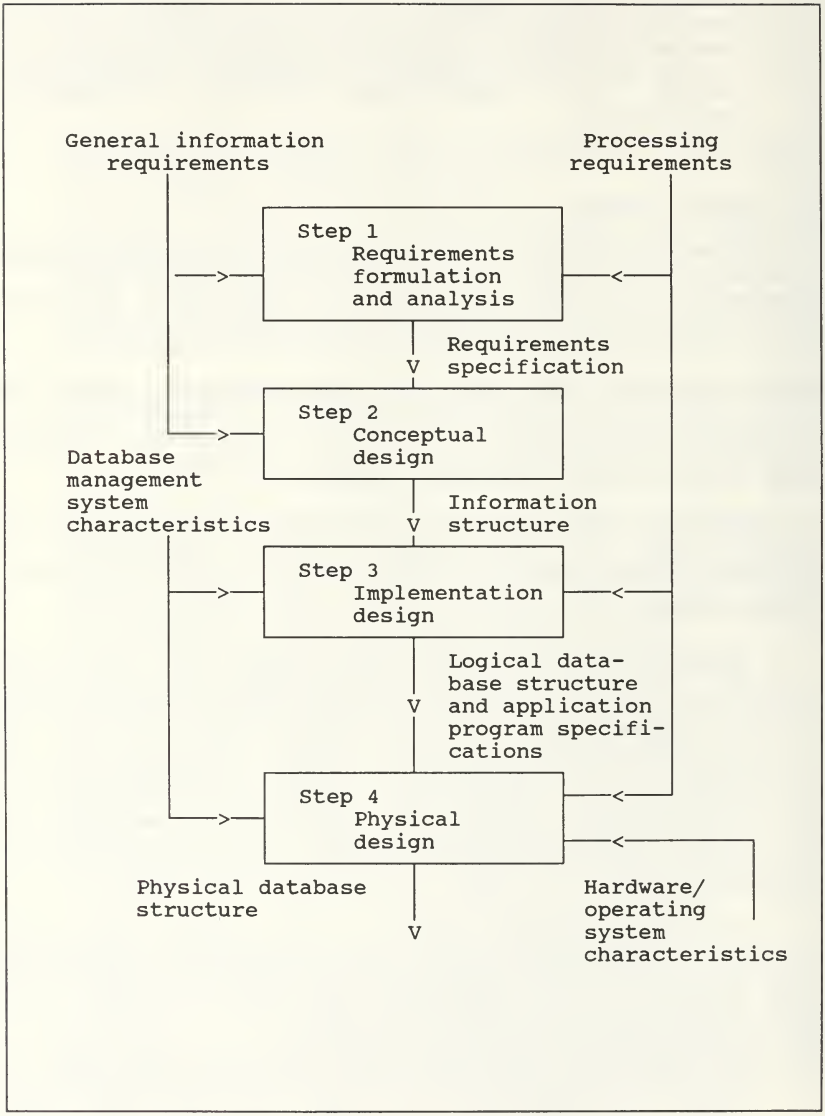


Figure 3. A general model for database design.

language for the specific DBMS. The construction of additional data structures and access paths to support the applications efficiently is also decided in this process. This step can be consideted as the intermediate step between logical and physical database design.

4. Physical design. This is the last step of database design. It includes designing stored record formats, selecting access methods, and deciding on physical factors. The steps in database design proceed in sequential fashion. However, there is much repetition and iterations among the steps in the design progresses since it may be discovered that there are gaps in the data definitions and they need additional requirement formulation and analysis.

The last two steps must be performed carefully, since they affect performance, integrity, security and a number of other factors that have a direct impact on user(s).

E. DATA MODELS

In the real world we deal with many types of information. This information must be stored economically and retrieved efficiently from the computer, in some form. This form enables one to represent data and their relationships about the real world in terms of a "data model". In other words, a data model is an abstract representation of the data about objects and their associations within an organization. A data model should be independant of a database management system.

There are three major data models that have been used in database systems. These data models are the "Hierarchical data model", the "Network data model", and the "Relational data model". Each of these data models have advantages and

limitations in their use and a short description is given below [Ref. 6:pp. 4-5].

1. Hierarchical data model

The hierarchical data model uses tree structure and the data are represented as a set of nested one-to-one and one-to-many relationships. In this data model a single occurrence of a record type has one parent and all of its children and the descendants.

Figure 4 shows a tree structure and how the data are organized in one-to-one and one-to-many relationships.

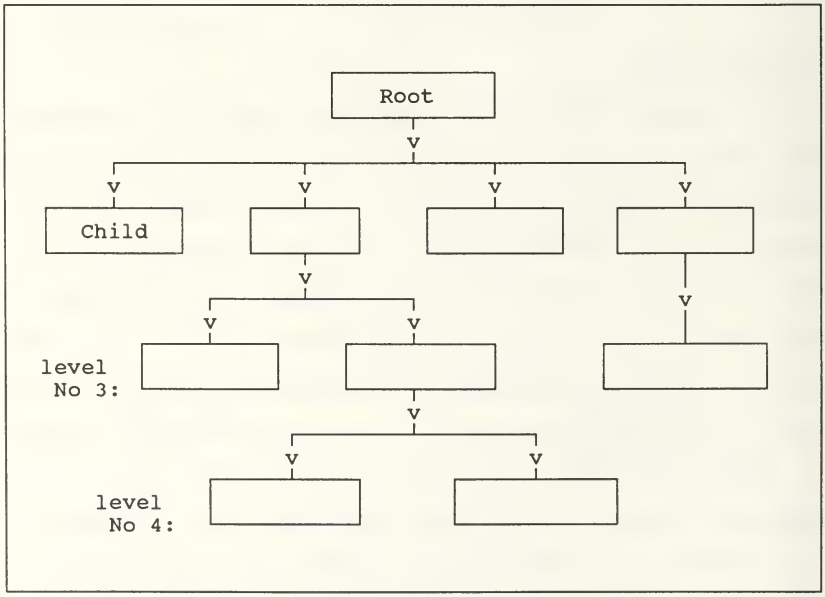


Figure 4. A Hierarchical data model.

The main advantage of this data model is that the tree structure, in general, is well understood, since it is widely used in many applications. The disadvantage of this model is the limitation of supporting many-to-many relationship easily, since a tree structure cannot support this kind of relationship directly. As a result, redundancy of data occurs [Ref 6:pp.91-100].

2. Network data model

The network data model represent its data by a set of record types and pairwise relationships between these record types. The connection link is an association between precisely two records. Therefore, relations that involve more than two record types are not directly permitted. The network data model is a graph data model and it can be considered as an extension of the hierarchical data model. The data structure of the network data model is shown in Figure 5.

The advantage of this model is that many relationship types can be easily depicted, and each relationship and record type is explicity stated [Ref. 3:p. 183].

3. Relational data model

In the relational model, the data are viewed as a collection of non-hierarchical time-varying relations. Operations or expressions of relational algebra can be extended to data manipulation. These operations can decompose a complex logical structure into a collection of simple

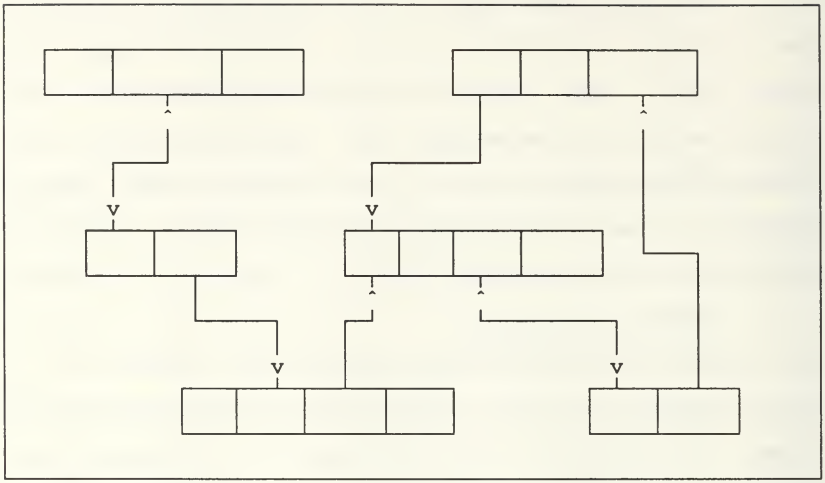


Figure 5. A Network data model.

relations and make accessing and updating data simple and fast.

The relational model is based on the mathematical concept of the set theoretic relations which is a subset of the Cartesian product of a list of domains. A domain is a set of values (e.g. character strings, character strings of length 15). A relation is any subset of the Cartesian product of one or more domains. These domains create two dimensional tables, the columns contain the values of the attributes and the rows are the tuples which form the elements of the relation. The columns are assigned by distinct names and the rows must be distinct, also. A relation is invariant under permutation of rows.

The ordering of the columns within a table is immaterial. The items of one column cannot be mixed with the items of another column. [Ref. 6:p.73]

In a relational model multiple columns can take values from the same domain. The degree of the relation is the number of columns in it. In a relation there exists one or more attributes which uniquely identifies every tuple. These attributes are called candidate keys. A relational data model is shown in Figure 6.

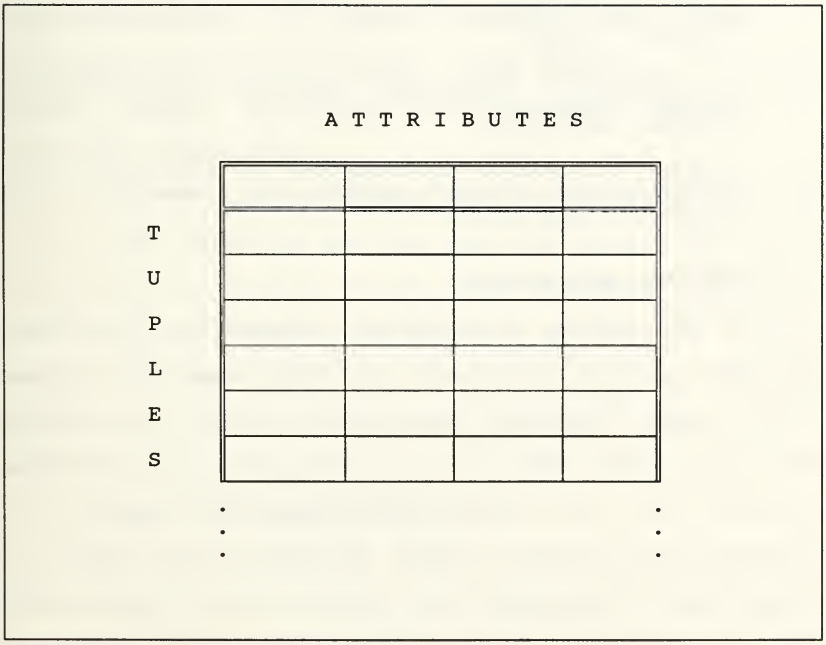


Figure 6. A Relational data model.

The relational data model is as rich as the complex network data model in its ability to represent directly, without much redundancy, a wide variety of relationship types. It is the choice of many database designers and users since the tables used are more understandable than the graphs or the trees and it is different from hierarchical and network data model in the following ways. [Ref. 3:pp. 183-187]

a. This model supports all the types of relationships (one-to-one, one-to-many, many-to-many).

b. High level programming languages have been developed specifically to access databases defined via the relational data model. These languages permit data to be manipulated as groups or files and not procedurally one record at a time.

c. The relational data model logically represents all relationships implicitly without looking at the internal data model.

d. Certain maintenance problems can be eliminated using "normalization theory" within the context of the relational data model.

F. CONCEPTUAL DATA MODELS

The data models, hierarchical, network and relational, have been used as the basis for data management systems (DBMS). These data models are too "low level" for adequate modeling of the real world and for producing conceptual schemas. This led to the development of external and conceptual data models. These conceptual data models are independent of the particular internal data model that will be or could be used. Two such data models will be introduced here. [Ref. 3:p. 198]

1. Semantic Data Model (SDM).

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

The entities and classes have attributes which relate them to other entities and they describe characteristics. An attribute has a semantic "Kind", which identifies the type of relationship the value of the attribute has with the entity. The value of the "Kind" may be one of component, property participant, class determined component, property, or participant. [Ref. 7:pp. 3-4]

2. Entity-Relatioship Model (E-R M).

The real world is modelled in terms of entities, relationship, and attributes. The Entity-Relationship model is a conceptual model and is based on that real world modeling. Entities and relationships can be represented

diagrammatically by an E-R diagram. In E-R diagram the entity sets are represented by rectangular boxes and the relationships by diamonds. Rectangulars and diamonds are linked with lines showing the types of the relationship. Attributes of each entity set are drawn in ellipses around their entity set rectangular. [Ref. 3:p.198]

III. APPLICATIONS

The developing and documenting of database application requirements are important because they are the blueprint for database design and implementation activities. [Ref. 4:p 87]

Defining requirements for a database and its applications has two major tasks. The first task is to identify and describe the objects that the users want to track and define their structure. The best way to do this is to work from the application outputs to derive the objects' structure. The second one is to determine the functional components of each application that will use the database. From these components, the users can obtain information from the database and keep it current. In other words the second task is to design the application programs.

The second task has two phases. First, the logical structure of the database (which means a DBMS-independent database design) is specified. Secondly, the transformation of this logical structure into a design that conforms to the limitations and peculiarities of a given DBMS product is made. The second task, application programs, is designed in parallel with the development of the logical database structure.

The purpose of a database application program is to enable the user to get the needed information about things that are important in his working environment. Each

application includes display, update, and control mechanisms for controlling access and processing the database. The display mechanisms include facilities to present the data on a hard copy printer, on a computer display screen, or send them to another device. With the update mechanisms the users can add new records, modify existing records, or delete unwanted ones. Finally, the control mechanisms control the database processing and the accessed data. This means the application programs can include authorization routines to assign different rights to each user or routines which restrict the users' access and processing options.

After developing and documenting database application requirements, they must be reviewed by the users before they will be used as input to the rest of the system development project. [Ref. 4:pp. 42-55]

A. DESCRIPTION OF CURRENT PROBLEM

A well described existing situation and the related problems are very important so that the application requirements for a new database system can be stated clearly, completely, and unambiguously. Also, it is much easier for the designer to understand the whole problem and to determine the structure of the database system. As discussed in the first chapter, every war ship must be well organized to be able to operate in different environments and under different situations. The crewmember on every war ship, as part of the

organization of the ship, should be organized such that the ship is able to perform its tasks and operate smoothly. The Executive Officer is responsible for the crewmember organization and this constitutes the mission of the administration office on the ship. In order for the administration office to complete its mission, two major tasks must be performed.

First, the crewmembers must be well trained for efficient execution of their duties and from time to time be trained again to bring them up to date. Second, additional or collateral duties must be assigned to other crewmembers since one or more jobs can be unfilled in the real time environment. The unfilled jobs may be created either from crewmembers on leave (standard, extra, hospital, etc), or from a shortage of the available number of crewmembers to fill the position in the ship, or because the departing crewmembers have to leave before the new crewmembers arrive to take over the duties. In the administration office, there are tables to define the duties and the training courses of each crewmember.

Even though these two tasks appear easy, they are difficult and time-consuming jobs for the administration office. The difficulty in these two tasks is that several different and sometimes unpredictable factors change the administration schedule. These factors will be described later in this chapter.

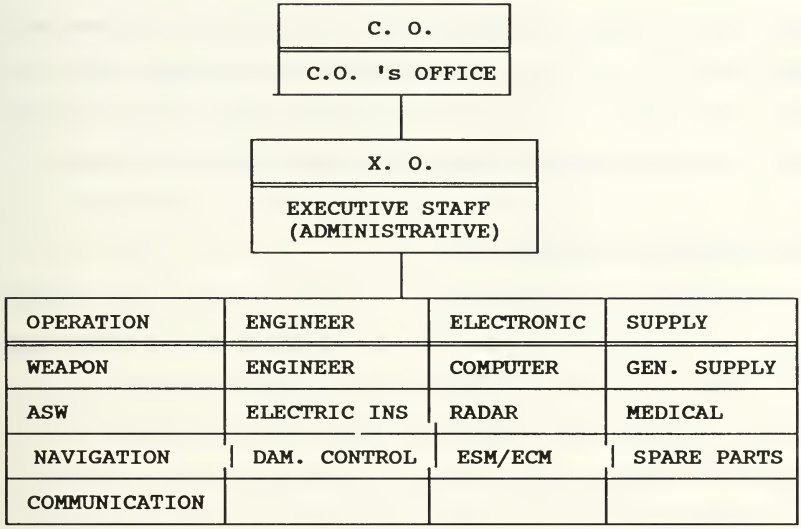
The crewmembers on a war ship are divided into three categories based on their ranks: officers, non-commissioned officers or petty officers, and the lower level crewmembers which will be referred to as seamen. Officers and non-commissioned officers (NCO) serve on a permanent basis, meaning an indefinite period of time, while the seamen serve for a fixed period of time. This is very important for developing the database system since the categories of officers and NCO's, and the category of seamen are under different regulations. These different regulations will be discussed later in this chapter.

From another point of view, the crewmembers are divided into four major categories: "Operations" department, "Engineering" department, "Electronic" department, and the "Supply" department. Each department has a number of subdepartments, and each subdepartment may have two or more sub-subdepartments in order to provide the necessary functions for the operation of the ship. This division of a war ship into departments and subdepartments is important since they form a hierarchy on the ship and makes the mission of the administration office simpler. Each department establishes its own training courses and leave schedules, and supports the administration office in assigning duties to its own crewmembers.

Duties under different ship environments are grouped together and duties between groups cannot be interchanged.

TABLE I.

CLASSIFICATION OF DEPARTMENTS AND SUBDEPARTMENTS



These groups are formed according to the "battle book" of the ship and each group is specified by a unique number with special meaning, named "duty number". The crewmember who is assigned with a duty number must be able to perform all his duties under all different ship environments. The administration office cannot change the group of these duties under specific "duty number" when assigning duties to new crewmembers. That means the administration office has to find the group of duties that best fit each person and his profile

according to previous knowledge, rank, and specialty in order to perform his duties efficiently.

The duty numbers are composed of five digits and each digit has a special meaning. The first digit represents the department, the second one represents the subdepartment, and the third digit the sub-subdepartment. The last two digits are the serial number of the hierarchy in the department.

B. APPLICATION REQUIREMENT

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

The problem of the administration can be described by stating the problem and the application requirements with respect to the use of the data needed to do the job, and by stating the problem according to how frequently each administration job is repeated and how urgently the outputs are needed in each case.

The first part helps the designer to understand and develop the logical structure for the data, the necessary objects (entities), and the needed application programs. The second part permits the designer to develop the database system and the necessary accesses to data efficiently as required by the

applications. Both parts are needed by a designer to find an appropriate solution.

The problem of the administration office can be divided into the following application requirements categories:

1. Daily application requirements.
2. Weekly application requirements.
3. Monthly application requirements.
4. On-demand application requirements.

Each one of these categories will be examined and analyzed in order to identify and describe the data that is required by the administration office.

1. Daily application requirements.

The daily jobs that the database system must accomplish for the administration office are scheduling leave, record punishments, and completing the "Daily duties on dock" report.

a. Leave

Four different types of leave exist in the Hellenic Navy and they fall into different categories. The types of leave and the categories that can participate in each of those leave categories are shown in Table II.

Standard leave is given by the administration office according to leave-tables that each department makes up in the beginning of every year, usually in January and after receiving the schedule of exercises for the current year from the Headquarters.

TABLE II.

TYPES OF LEAVE IN THE HELLENIC NAVY

	OFFICERS	N.C.O.	SEAMEN
STANDARD	X	X	X
EXTRA	X	X	X
ON-SEA			X
SPECIAL			X
DUTIES EXCEPTION	X	X	X
HOSPITAL	X	X	X
HOSTITAL LEAVE	X	X	X

Another type of leave is the extra leave. Its purpose is to provide the flexibility to the Executive Officer to allow any unscheduled short leave to crewmembers for reasonable problems or for special situations (e.g., tragedy in the family). The total extra leave days are not counted or summarized with any other types of leave.

On-sea leave is the type of leave that only a seamen can take. This type of leave is allowed by the Headquarters as an extension of the standard leave and only to seamen for which their total service period is on a ship. Special leave can be given only to seamen and under special circumstances.

Duties exceptions and hospital leave are two cases on which a crewmember can be away from the ship. Duty exceptions include several kinds of crewmember activities away from the ship by order. Although the above three cases are not actually leave, it is easier and very helpful for the administration office to include these with the other types of leave since the method of handling them is the same. Each department is responsible for making up the leave schedule tables with the permitted types of leave for its crewmembers in such a way that leave periods do not overlap with other similar crewmember duties.

In order to satisfy the administration requirements, the database system must be able to store the leave schedule made up by each department with all the necessary information about the crewmembers, such as the leave type, the leave period, the start date, etc. The database system must be able to check if a leave period overlaps with an exercise period or a training course. For that reason, information about the periods of exercises and training courses must be included in the database system.

Since creating or defining leave depends mainly on the ship's requirements and its policy, sometimes leave is permitted to occur in an exercise period or in a training course period. These exceptions must be accepted by the database design. In case a change of a person's leave is

necessary, the database system must help the user to determine a new leave period with the above described regulations.

Another regulation about leave is the assignment of the absent crewmember's duties to another crewmember with similar duties. The rules of this regulation are established by the Executive Officer and usually depend on the leave period and the significance of duties. So, the database must be able to support the administration office in making a decision whether an assignment is needed and which of the crewmembers will be assigned these duties.

b. Punishments

The second daily job that the database system must be able to do in order to work for the administration office is to keep records about the crewmember being punished.

Four different types of punishments exist in the Hellenic Navy. These four types are "Prison ashore", "Payback confinement", "Consecutive day confinement", and "Alternate day confinement". These types of punishments are meted out to any crewmember on the ship but actually only the seamen have to serve the punishment. The execution of these types of punishments for the other two categories, officers and N.C.O., is just the issue of a report of the punishment to the Headquarters.

Prison ashore means the crewmember under punishment still belongs to the ship but he stays ashore in a

prison for the punishment period. Payback confinement means the crewmember stays on the ship for the whole punishment period. Prison ashore and payback confinement days are added to service total period. This addition for extending service period is taken care of by the Headquarters through the monthly reports from the ship and not by the administration office. Consecutive day confinement is the same as the pay-back confinement type of punishment but does not count in the service period. In the last type of punishment, alternative day confinment, the days count every time that crewmember is permitted to go out.

The punishment data must be accepted by the database system for all the crewmembers on the ship and the daily punished crewmember lists that are used by the officers on duty need to include only the seamen.

c. Daily duties on dock.

The last daily task that the administration office has to do is to complete a report for the officer on duty.

There are differences between the operation of a war ship and the operation of a commercial organization where members arrive each morning, work for a specific period of time and leave until the next working day. A war ship is an organization which must operate on an "around the clock" basis. After the working hours, a number of officers, N.C.O. and seamen stay on the ship and they are called crewmembers

on duty. The senior officer on duty is responsible for the safety and proper operation of the ship. For ease of discussion, we shall name this officer the "Officer In Charge". He supervises and directs the operation of the ship in accordance with the orders of the Commanding Officer and other proper authorities, and executes the scheduled activities and jobs established by the Executive Officer. The administration office supports the efficient execution of the task of the Officer In Charge by publishing a "Daily duties on dock" report. This report includes mainly the Commanding Officer's orders and the activities and jobs defined by the Executive Officer.

The daily officers and N.C.O. on duty are usually scheduled monthly. Changes on those monthly duties lists can be made only with the approval of the Commanding Officer or the Executive Officer. The crewmembers of the ship, on the other hand, are divided into shifts with equal number of persons in each one. During exercises, two shifts or three shifts are required for the ship's operation and the number depend on the exercise requirements. A two-shift schedule is used for the seamen to define the period the seamen can leave the ship. This means that at any time about half of the total seamen stay on the ship. However, the Executive Officer has the authority to change the permissions of some seamen to leave the ship by defining another way such as a three-shift schedule. The monthly duties schedule for officers and

N.C.O., the exceptions of the seamen on duties, and the activities and jobs that the officers on duty have to perform must be stored as data in the database system.

The database system, in order to make up the "Daily duties on dock" report, must be able to do the above by date and to make up the daily seamen's stay on the ship according to the two-shift schedule or the alternatives established by the Executive Officer. It must also provide lists of the crewmembers' under punishment, crewmembers on leave, and the participating crewmembers on each training course for the specified date. Since the Officer In Charge has to know the total number of crewmembers on leave and its percentage, etc., the database system must calculate and provide these numbers for him.

Figure 7 shows a "Daily duties on dock" report that the administration office must provide to the Officer In Charge for the efficient execution of his duties. This report must be provided automatically by the database system daily.

2. Weekly application requirements.

The second category of application requirements is the "weekly application requirements". The jobs that are repeated in the administration office every week are training courses and planning for next week.

a. Training courses.

As previously mentioned, one of the major tasks of the administration office is the scheduling and the

DAILY DUTIES REPORT

DATE :

TO: (Officer on duties name)

INSTRUCTIONS TO OFFICER ON DUTIES

A. ON DUTIES :

(RANK) (SPECIALY) (FIRST NAME) (LAST NAME) (DUTIES)

B. STAY-IN SEAMEN :

(RANK) (SPECIALY) (FIRST NAME) (LAST NAME)

C. CREWMEMBERS :

ON LEAVE:
(RANK) (SPECIALY) (FIRST NAME) (LAST NAME)
(TYPE OF LEAVE)

IN PUNISHMENT :
(RANK) (SPECIALY) (FIRST NAME) (LAST NAME)
(TYPE OF PUNISHMENT)

IN TRAINING:
(RANK) SPECIALY) (FIRST NAME) (LAST NAME)
(TYPE OF COURSE)

D. STATISTICS :

OFFICER ON LEAVE :
TOTAL OFFICERS : PERCEN (%) :
N.C.O ON LEAVE :
TOTAL N.C.O : PERCEN (%) :
SEAMEN ON LEAVE :
TOTAL SEAMEN : PERCEN (%) :

E. ACTIVITIES / NOTES :

Approved by: (Executive Officer)

Figure 7. Daily duties on dock report.

execution of the training courses. Basic training courses for new crewmembers, advanced training courses for them and for crewmembers that have already attended these courses, as refreshers on the subjects, and some training courses ordered by the superior commanders are included in this category. The basic selection rules for the crewmembers to participate in a training course are their rank, specialty and the date they last attended the course. Most of these training courses are fixed and described in the "training courses" book and some of them are derived from other ship's requirements.

Each department submits its own training course schedules to the administration office every year. Then the administration office is responsible for making the final training course schedule for the ship and to complete the crewmembers' training lists before the execution of any training course.

In addition, some training courses may be ordered by superior commanders and these courses are usually scheduled after working hours. That means the execution of these courses is the responsibility of the officer in charge and so they must be included in the "Daily duties on dock" report.

Unpredictable factors sometimes require the changing of a scheduled training course. The job of rescheduling such a training course is difficult and

time-consuming. The difficulty is that the rescheduled period must not overlap with a period of an exercise or another training course and on the other hand, rescheduling a training course causes disturbances in other schedules. These disturbances have a propagating effect and iterations must be performed.

The database system must be able to store any necessary information about the training courses and relate them to the yearly training schedule given by each department of the ship and with the training courses ordered by the superior commanders each month. The database system must be able to retrieve any training course data and determine the participating crewmembers in each training course and, when necessary, modify appropriately the participatants' list. Finally, the database system must support its users in making decisions and help them schedule the leave and training courses when rescheduling courses become necessary.

b. Planning for next week.

The administration office, in order to make a successful job, must schedule and program its activities for at least one week in advance, taking as input the information from the yearly and monthly schedules and from other sources such as messages, etc. The next week schedule is very helpful for the Executive Officer since he can program easier the

activities and jobs of the ship. Figure 8 shows the form that the administration office completes for the X.O..

The database system must be able to satisfy the Executive Officer's and administration office's requirements by scheduling and reporting the next week's planing. This report must include the crewmembers' leave schedule, the scheduled training courses, and the scheduled exercises. Since during an exercise period, one or more superior commanders with their staff can get on the ship, their names and the periods of stay must be included in the above described database report.

3. Monthly application requirements.

The monthly application requirements includes submitted reports to Headquarters. These reports are officers' reports, N.C.O.s' reports, and seamen's reports. The reports on the seamen punishments are important, as mentioned above, since their punishment days for some type of punishments are counted in the total service period and not for some others.

The database system must be able to make these reports by collecting the appropriate data about each category. The needed data for all reports include rank, specialty, first name, last name, duties on dock, leave, punishments of the last month and the address, if a change of address request has been submitted.

WEEK:

SCHEDULE FOR NEXT WEEK

A. SCHEDULED LEAVE :

(RANK) (SPECIALY) (FIRST NAME) (LAST NAME)

B. SCHEDULES TRAINING COURSES :

(COURSE NAME) (START DATE) (END DATE) (PLACE)
(MAX NO OF PERSONS)

C. ACTIVITIES :

1. EXERCISE : (EXERCISE NAME) (START DATE) (PERIOD)
(AREA)

2. VISITORS : (TITLE) (RANK) (DATE ON SHIP)
(FIRST NAME) (LAST NAME)

D. CREWMEMBERS ON LEAVE :

(RANK) (SPECIALY) (FIRST NAME) (LASTE NAME)
(RETURNED DATE) (TYPE OF LEAVE)

E. NOTES :

Figure 8. Schedule for the next week.

4. On-demand application requirements.

The last category of the administration requirements, as described in the beginning of this section, includes the jobs that have no fixed frequency of occurance. This category includes the collection of data about a new crewmember or changing the file data of a crewmember already on board, completing emergency calling lists, assigning duties to a new crewmember and finally, furnishing any information to the superior commander and his staff about their responsibilities (duties) on the ship.

a. Crewmember status data.

All the administration office's activities and jobs are based on the crewmember status and professional data. Therefore, these data are very important for efficient performance of the database system and they must be handled carefully.

Figure 9 shows the report that must be filled by each new crewmember. The report in Figure 11 includes the hobbies that the new crewmember is interested in and the jobs that he did before he joined the Navy. The first part is helpful for planning recreation activities of the ship and the second one helps the administration office, especially for the seamen, in making decisions for the assignment of duties. The meaning of different types of addresses, in Figure 11, is that the administration office has to know all the necessary addresses and telephone numbers to reach every

CREWMEMBER REPORT

Please, complete all the qyestions. If something is unknown, let it blank but you must complete it as soon as possible. For any question ask the administrative staff.

A. CREWMEMBER STATUS

MILITARY I.D.	:	MARITAL STATUS	:
RANK	:	No OF CHILDREN	:
SPECIALTY	:	CLASS	:
FIRST NAME	:	CLASS No	:
LAST NAME	:	INDATE	:
BIRTHDAY	:	OUTDATE	:

B. ADDRESS

1. NEAR BASE AREA
   ADDRESS:
   AREA :
   TOWN : TELEPHONE:
2. ORIGINAL
   ADDRESS:
   AREA :
   TOWN : TELEPHONE:
3. RELATIVE
   ADDRESS:
   AREA :
   TOWN : TELEPHONE:
4. POLICE STATION
   ADDRESS:
   AREA :
   TOWN : TELEPHONE:

Figure 9. New crewmember form.

C. ACTIVITES/EDUCATION

1. HOBBY :

2. EDUCATION :

3. PR. OCCUPATION :

D. NOTES

DATE :

SIGNATURE :

Figure 9. New crewmember form (cont 'd).

crewmember in case of an emergency situation. The original address helps the Executive Officer to allow short leave when the ship visits a port nearby.

The database system nust be able to handle and process information related to these activites.

b. Duties.

The assignment of duties can be viewed as follows. First, as described above, a crewmember must be assigned the duties of another crewmember on leave. Second, a new crewmember gets duties of the crewmember on leave by order. Third, the new crewmember must be scheduled for

additional training if he is not ready to accept his duties. The second case is an easy job since the duties are just transfered from one crewmember to the other. But the third one is more difficult since another crewmember must be assigned as an intermediate person.

The database system must be able to provide any necessary information in order to help a user in making decisions for the assignment of duties.

c. Visitors.

The last job of the administration office is to provide any information to superior commanders and their staff about their responsibilities on the ship. Such information on responsibilities includes the positions in any type of alert, the number of life boats in case of abandon ship, etc.

These responsibilities are defined according to the rank and specialty of each officer and must be included in the database.

C. SUMMARY OF APPLICATIONS

In this chapter, the administration office requirements were described. These requirements will be used for database design and implementation for the next chapters. The described requirements have some conflicts and constraints and they will be discussed in a separate chapter after designing the logical structure of the database system.

An important thing for designing the database system is that the program outputs must be in a hierarchical order. This means that the crewmember name lists , especially the officer and N.C.O. categories, must be based on their rank and specialty, and, between the crewmembers with the same rank and specialty, the graduation year of the Naval Schools (class) and perhaps the ranking number in which the crewmember graduated in his class (class No) will be used for making a decision. The above criteria must be taken care of by the database system when creating output lists.

IV. DESIGN

The process of developing a database structure from application requirements for data is called database design.

Designing a database is a complex and difficult process. This process requires an organized approach or methodology. A general approach model with four major steps for database design has been developed by Teory and Fry. [Ref. 3:p. 213] The second step of that general model is the conceptual design as shown in Figure 3.

During the conceptual design process the development team makes use of a data model that can support all the applications. This process can be accomplished by defining the entity classes, identifying the relationships among the entities, and transforming the entities into relations. Then the project team must review the established relations and apply the rules of normalization to those relations to find existing anomalies. When anomalies are found the team modifies the design to eliminate them. [Ref. 4:pp. 210-213]

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

There are two design approaches dealing with the conceptual design: top-down and bottom-up. The designer, in the

first approach, builds the enterprise model and then adds detail to it until a satisfactory conceptual design has been achieved. For that reason, this approach is also called entity analysis. In the second approach, the designer starts with a detailed requirements analysis and proceeds to build each user's view separately. The conceptual scheme is then formed by merging the relations from each user's view. The bottom-up approach is preferable since in the top-down approach there is no assurance that all user requirements have been represented in the conceptual scheme. [Ref. 3:pp. 245-249]

A. STEPS IN CONCEPTUAL DESIGN

In the conceptual design, five major steps must be performed, even though the detailed steps are dependent on the approach or methodology of a user. These major steps are shown in Figure 10 and are described below: [Ref. 3:pp. 249-253]

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

2. View Intergration. In this step, the structured relations from the above step are merged together in a single set of relations. The result of this step is the conceptual data model expressed in the form of normalized relations.

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

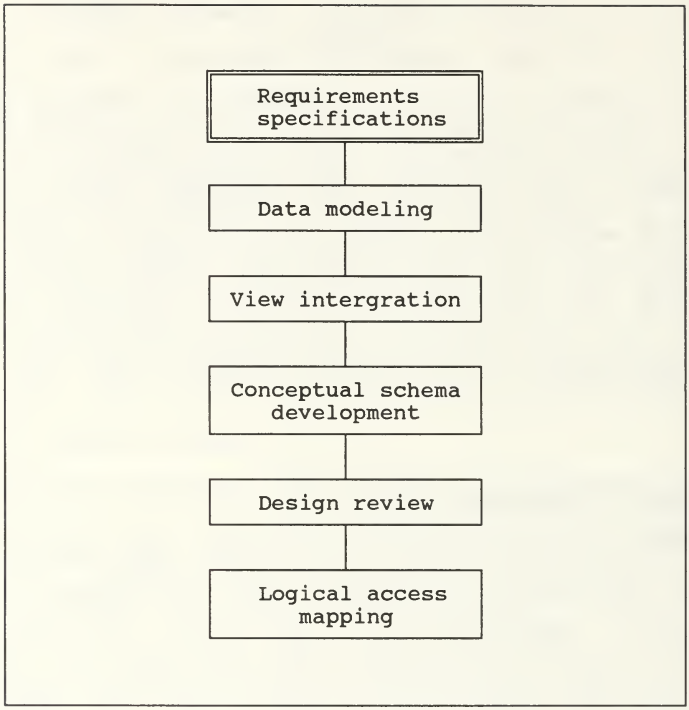


Figure 10. Steps in Conceptual design.

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

5. Logical Access Mapping. The last step in conceptual design is logical access mapping. In this step, diagrams showing the logical sequence of accessing the conceptual records are drawn. Logical access mapping may be considered part of conceptual designor a step in implementation design.

This chapter will follow the described conceptual design steps, starting with Semantic Data Model (SDM). Next the normalized relations from the SDM will be transformed into an Entity-Relationship (E-R) graphical model. The fourth step "Design Review" will be the subject of the next chapter. Finally, the last step will be part of the implementation chapter.

B. INTRODUCTION TO SEMANTIC DATA MODEL (SDM)

The semantic data model (SDM) was developed by Hammer and Mc Leod and can express a conceptual data base design. The SDM allows the same information to be viewed in several ways. [Ref. 8: 351-356]

The database designers that deal with SDM should define a conceptual structure for each of the real world structures. Each database is a model of some real world environment. The real world has some primitives; phenomena that can be represented by nouns as objects. Each object has properties. A property is a characteristic of the object and the collection of all possible values of a property is called a property value set. A particular value from the property value set for a given property and object is called a fact. The last term that is associated with a database for the real world is the relation of the objects, called associations.

When we design or process a database, we are not working with the above described real world primitives, but we

represent these primitives using conceptual terms. Database experts have defined a conceptual primitive for each of the real world primitives. An entity is a conceptual representation of an object having properties which are called attributes. The collection of all values that an attribute can have is called domain. Value is the representation of a fact. Finally, a relationship is the conceptual representation of an association.

Figure 11 shows the equivalencies between real world and conceptual primitives. [Ref. 9:p. 209]

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

Figure 11. Real world and Conceptual primitives.

Figure 11. Real world and Conceptual primitives.

C. GENERAL PRINCIPLES OF DESIGNING SDM

There are general principles of database organization to support the design of SDM as follows:

1. "A database is to be viewed as a collection of entities that correspond to the actual objects in the application environment."

2. "The entities in a database are organized into classes that are meaningful collections of entities."

3. "The classes of a database are not,in general independent, but rather are logically related by means of interclass connections."

4. "Database entities and classes have attributes that describe their characteristics and relate them to other database entities. An attribute value may be derived from other values in the database."

5. "There are several primitive ways of defining interclass connections and derived attributes, corresponding to the most common types of information redundancy appearing in database applications. These facilities integrate multiple ways of viewing the same basic information and provide building blocks for describing complex attributes and interclass relationships." [Ref. 8:pp. 355].

D. DESIGNING WITH SDM

SDM is a form to synthesize the various user's views and information requirements into database design using a data model form. Two such forms will be used for the database design: SDM and Entity-Relationship (E-R) diagram. The latter form will represent the normalized SDM design in diagrams for better understanding according to the third step of the conceptual design (conceptual schema development) as shown in Figure 10. The Relation database development will be used as

In the previous chapter, the requirements of the administration office were described. The description of the administration office requirements shows the needed entity classes for the database system design. These entity classes with any necessary information are synthesized into a SDM

```
ENTITY_CLASS_NAME

    [description: ..........]
    [interclass connection: ........]

    member attributes:

        Attribute_name
            value class: ..........
           [mandatory]
           [multivalued][no overlap in values]
           [exhausts value class]
           [not changeable]
           [inverse: Attribute_name]
           [match: Attribute of ENTITY_CLASS
                   on Attribute_name2]
           [derivation: ..........]

    [ class attributes:
           Attribute_name
               [description: ..........]
                value class: ..........
               [derivation: ..........]

    [ identifiers:
           Attribute_name1 + [Attribute_name2 + [...]]
```

Figure 12. SDM Entity class description.

form according to the SDM description scheme shown in Figure 12. The SDM design and its domain definition are given in Appendix A as Figure A-1 and Figure A-2.

Each attribute has a name, a description, a value class, and a set of characteristics as shown in Figure 13. The name and the value class are mandatory in each attribute. The value class is the set of values that the attribute can have. In other words, it is another term for domain that is used by the SDM [Ref. 9:p. 219]. The value class and the set of characteristics that each attribute uses to represent the object properties form the constraints of the database design.

Decsriptor	Status	Remarks
Name	Mandatory	Initial capital letter
Description	Optional	Remarks about attribute
Value class	Mandatory	Domain of the attribute

Descriptor Characteristics	Default Value
Single or multivalued	Single
Value optional or mandatory	Optional
Changeable or not changeable	Changeable
Exhaustive or nonexhaustive	Nonexhaustive
Overlapping or nonoverlapping	Overlapping

Figure 13. Attribute descriptors in SDM.

E. SDM DESIGN SUPPORT

Constraint is a rule about the data or its relationship to other data in the database. Three types of constraints may be expressed in the conceptual design: domain constraints, intra-relation constraints, and inter-relation constraints. These three types of constraints have been used in the administration SDM design. [Ref. 4:p 369]

1. Domain constraints.

The domain constraints state the allowed data values that can be accepted by the value class of each attribute. The statement of allowed data values includes the data type (character, numeric, logical, date, memo), the maximum length of data, a description of the allowable range of data values, or a discrete set of allowable values. [Ref. 4:p 370]. For example, the domain LE_NAMES is a discrete set of six different allowable values, the domain DATES is a data type of date in form MMDDYY, etc. The domain constraints of the designed SDM are given in Appendix A as Figure A-2.

2. Intra-relation constraints

An intra-relation constraint states the characteristics of the data within a table. The set of characteristics that are used by the SDM design are: [Ref. 9:pp. 219-221]

a. Single or multivalued. The value of an attribute can be single or multivalued (like a repeating field). For example the attribute RANK in TRAINING_COURSE_INFO relation is a multivalued relation since it can take more than one value.

b. Value optional or mandatory. An attribute can be specified as mandatory which means an accepted value must be

inserted, that is, a null value is not allowed from the conceptual point of view. For example, the CR_NUMBER attribute in CREW_PROF relation is specified as "mandatory"; this models the fact that every crewmembers has a CR_NUMBER (Military ID).

c. Changeable or not-changeable. An attribute can be not-changeable, meaning that the value of the attribute cannot be altered except to correct existing error. For example CR_NUMBER attribute in CREW_PROF relation is characterized as "not changeable" since each crewmember has a unique Military ID and it is never changed.

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

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

These intra-relation constraints have been used in the SDM design as shown in Appendix A Figure A-1, and they state the characteristics of each attribute. The attributes without any intra-relation constraints, are assumed to have the default value of the characteristics of Figure 13.

3. Inter-relation constraints

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

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

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

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

F. APPROACH TO RELATION DESIGN

Several approaches for organizing and manipulating a database have evolved in the past twenty years. One of them is the relational database model and this model will be used in the implementation design. This model organizes and stores the data in two-dimensional tables called relations. [Ref. 4:p. 131] The relational data model is rich in the ability to represent a wide variety of relationships without significant redundancy. However, these relationships are implicit in that they cannot represent explicitly the relationship between two relations. [Ref. 3:p. 186]

In the relational data model, certain restrictions are imposed as follows: [Ref. 4:pp. 132-133]

1. "Attributes are single valued; neither repeating groups nor arrays are allowed."
2. "Entries in any column are all of the same kind".
3. "Each attribute has a unique name and attribute positions are insignificant".
4. "No two tuples in a relation may be identical and the order of the tuples is also insignificant".

Any table that satisfies these four restrictions can be a relation. Changing data in a relation can have undesirable consequences, called modification anomalies. These anomalies

can be deletion anomaly, insertion anomaly, or inter-relation constraint. By insertion anomaly we mean any new insertion of data or updating existing data in the relation. For better understanding of the modification anomalies, an example of a relation shown in Figure 14 will be used in the following discussion. The example shows an entity class with the crewmember names and the training courses that they have attended. For this discussion, we shall assume that each tuple in the relation must be fully represented to be valid. [Ref. 4:pp. 134-136]

1. Deletion anomaly.

Deletion anomaly can happen when deleting a fact from one entity causes loss of a fact from another entity. This means that by deleting (e.g. the officer "Coleman Michael" with the Military ID "12654") in Figure 14 the data about the training course "8725" and its name "RADAR" will be lost.

2. Insertion anomaly.

Insertion anomaly can happen when we cannot insert a fact in one entity without inserting an additional fact in another entity. For example we can not insert the crewmember "Haley Christopher" with MIL_ID "19544" in the example without inserting the training course number that he has attended and its training name.

In the above anomalies the problem is solved by dividing the entity class CREW-TRAINING into two different entity classes (e.g. CREW and TRAINING) as Figure 15 shows.

CREW_TRAINING

Key : (MIL_ID, T_NUMBER)

MIL_ID	LAST_NAME	FIRST_NAME	T_NUMBER	T_NAME
18752	HARRISON	TOM	8725	RADAR
19654	COLEMAN	MICHAEL	3512	D. CON.
18927	MITCHELL	CARLOS	8725	RADAR
19522	PATTON	PAUL	4999	RADIO
18752	HARRISON	TOM	4999	RADIO

Figure 14. CREW_TRAINING relation with anomalies.

The two new relations are connected by including the key of one relation in the other. Separating a relation into two smaller relations results in another anomaly called an inter-relation constraint.

3. Inter-relation constraint.

This type of anomaly appears when we want to insert a crewmember into the CREW entity class but he has not as yet attend a training course.

The process of conceptual database design has two major goals; first, synthesizing the user requirements to construct the objects, and second, using enough robust

relations to avoid modification anomalies which result in inconsistencies. [Ref. 4:p. 133]

CREW

Key : MIL_ID

MIL_ID	LAST_NAME	FIRST_NAME	T_NUMBER
18752	HARRISON	TOM	8725
19654	COLEMAN	MICHEAL	3512
18927	MITCHELL	CARLOS	8725
19522	PATTON	PAUL	4999
18752	HARRISON	TOM	4999

TRAINING

Key : T_NUMBER

T_NUMBER	T_NAME
8725	RADAR
3512	D. CON.
4999	RADIO

Figure 15. CREW and TRAINING relations without anomalies.

The theoretical process of a well-designed relation is called normalization. This process is the examination of the relations using standard normal forms but avoids the anomalies. Before describing the normal forms and examining the designed relations, terms that are necessary for the normalization process will be described in the next sections. [Ref. 4 :p. 133]

G. FUNCTIONAL DEPENDENCY

Functional dependency (FD) is a term derived from mathematical theory and is a relationship between attributes. The functional dependencies between attributes in a relation do not involve equations. A functional dependency is represented by "X --> Y" and is read: X functionally determines Y or Y is functionally dependent on X. By this notation, each value of attribute (or set of attributes) X uniquely determines one and only one value of attribute (or set of attributes) Y. In other words, knowing the value of X is sufficient to determine the value of Y and there is no other value of Y for this particular value of X. Therefore, if the Y attribute is determined by the attribute X, then the relationship between the attributes Y and X is N:1. Since the Key of an entity class is a group of one or more attributes that uniquely identifies a record, it is obvious that all the other non-key attributes in the entity class are functionally dependent on the Key. [Ref 4 :pp 138-139]

To show the basic principles of functional dependencies, consider the sample CREW-TRAINING database in Figure 14. The Key in this relation is the composite attributes MIL_ID and T_NUMBER. According to the above paragraph all the non-key attributes are functionally dependent on the key. But the T_NAME attribute is also functionally dependent on the T_NUMBER since knowing a value of the attribute T_NUMBER, we can determine the value of the attribute T_NAME. Similarly, we can see that the LAST_NAME and FIRST_NAME are functionally dependent on the MIL_ID attribute. These functional dependencies are shown in Figure 16.

In the functional dependency theory, two other terms are important and they are related to the functional dependency concept: full functional dependency, and transitive dependency.

1. Full functional dependency.

An attribute (or set attributes) X fully determines an attribute (or set of attributes) Y if Y is functionally dependent on X and Y is not functionally dependent on any proper subset of X. For example, in the relation CREW_TRAINING in Figure 14 the attribute LAST_NAME is functionally dependent on the key (MIL_ID , T_NUMBER) but since this attribute is functionally dependent on the subset of the key (MIL_ID) we can say that LAST_NAME is not full functionally dependent on the key (Figure 16).

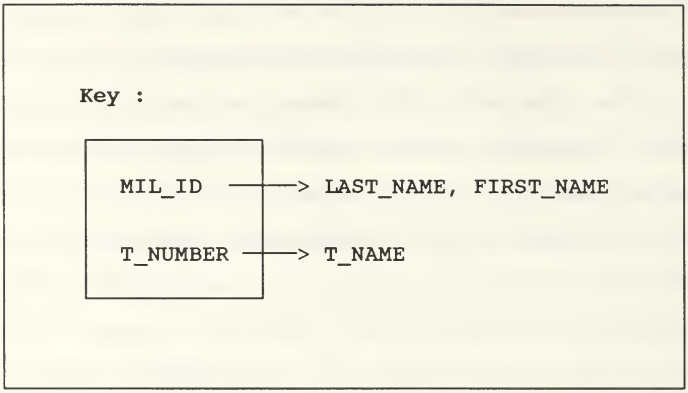


Figure 16. Functional Dependencies.

2. Transitive dependency.

If an attribute (or set of attributes) Z is functionally dependent on an attribute (or set of attributes) Y and this attribute Y is functionally dependent on another attribute (or set of attributes) X, then the attribute (or set of attributes) Z is functionally depended on the attributes) X. In other words, if X --> Y holds and Y --> Z holds then X --> Z holds. [Ref. 5:pp. 186]

H. NORMAL FORMS

As previously mentioned, normalization is the process for forming well-designed relations by grouping attributes together. This process is the examination of the relations to be in one of the normal forms that the relational theorists have defined. Each of the higher normal forms contains others in

the lower ones, as shown in Figure 17. This means that if a relation, for example, is in second normal form, then it is automatically in the first normal form. Therefore, the steps of normalization are standard and one normal form follows another. The existence of these normal forms is because any one of them does not eliminate all the anomalies, but only certain anomalies. For that reason, it was necessary for relational database designers to search for more and more normal forms to eliminate all the anomalies, but theoretically the examination of a designed database system up to BCNF normal form gives a sufficiently well-structured relational database. Although there are higher normal forms, their application in practice is not so well established and therefore they will not be discussed in this thesis as they are not used here.

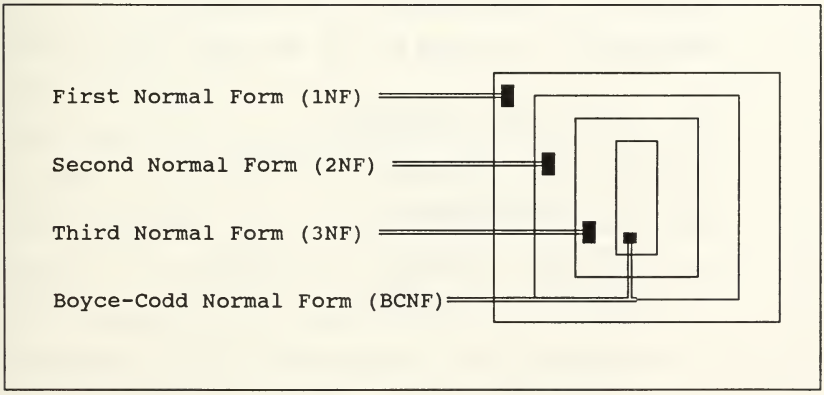


Figure 17. Relationship of normal forms.

Before we normalize the database design (SDM) to eliminate the existing anomalies, definitions and an example for each normal form will be given. In general, when determining whether a particular relation is in a specific normal form, the functional dependencies must be examined first.

1. First normal form (1NF)

"Any relation is in first normal form if the relation has no repeating groups in it". This means that all the attributes in the relation must be atomic and the records in

CREW_ADDRESS

LAST_NAME	ADDRESS	TELEPHONE
HARRISON	21 SECOND ST	372-5281
COLEMAN	12 FIFTH ST	372-4185
PATTON	12 FIFTH ST	372-4185
HARRISON	31 FIFTH ST	372-6409

Key : (LAST_NAME, TELEPHONE)

Candidate key : (LAST_NAME, ADDRESS)

Functional dependencies :

(LAST_NAME, TELEPHONE) --> ADDRESS

TELEPHONE --> ADDRESS

Figure 18. CREW_ADDRESS relation in not 2NF.

it must have the same set of attributes. But since this is the definition of a relation we can say that every normalized relation is in first normal form. [Ref. 4:p. 142]

2. Second normal form (2NF)

"A relation is in second normal form if all non-key attributes are dependent on all the attributes of the key". This means if the key is a single attribute then the relation is automatically in second normal form. If the key is a set of attributes then all the non-Key attributes must be fully functionally dependent on the key. [Ref. 4:p.142] For example, the CREW_ADDRESS relation is not in 2NF since the non-key attribute ADDRESS is not fully functionally dependent on the Key as Figure 18 shows. Therefore, CREW_ADDRESS relation may be decomposed to form two relations in second normal form as shown in Figure 19.

3. Third normal form (3NF)

"A relation is in third normal form if it is in second normal form and has no transitive dependencies" [Ref. 4:pp. 142-144]. The example in Figure 20 is not in 3NF since the key is the CR_NUMBER which determines the duty on dock (D_DOCK). Therefore, the CREW_DUTIES relation has a transitive dependency which creates anomalies. For example, what happens if we delete the crewmember with the CR_NUMBER "18-752"? We lose not only the fact that the assignment of this crewmember has been changed with the DUT_NUMBER but also the fact that this DUT_NUMBER corresponds to D_DOCK "Navigation".

CR_TEL

LAST_NAME	TELEPHONE
HARRISON	372-5281
COLEMAN	372-4185
PATTON	372-4185
HARRISON	372-6409

TEL_ADD

TELEPHONE	ADDRESS
372-5281	21 SECOND ST
372-4185	12 FIFTH ST
372-6409	31 FIFTH ST

Figure 19. Decomposed relations that are in 2NF.

CREW_DUTIES

Key : CR_NUMBER

CR_NUMBER	DUT_NUMBER	D_DOCK
18752	43147	NAVIGATION
19654	57439	COMMUNICATION
18927	63247	WEAPON

Functional dependencies:

CR_NUMBER --> DUT_NUMBER

DUT_NUMBER --> D_DOCK

Figure 20. CREW_DUTIES relation that it is not in 3NF.

CREW

Key : CR_NUMBER

CR_NUMBER	DUT_NUMBER
18752	43147
19654	57439
18927	63247

DUTIES

Key : DUT_NUMBER

DUT_NUMBER	D_DOCK
43147	NAVIGATION
57439	COMMUNICATION
63247	WEAPON

Figure 21. Decomposed relations that are in 3NF.

CR_DUTIES

LAST_NAME	DUT_NUMBER	DEPT
HARRISON	43147	ELECTRONIC
COLEMAN	57439	OPERATION
HARRISON	63247	SUPPLY

Key : (LAST_NAME, DEPT)

Candidate key : (LAST_NAME, DUT_NUMBER)

Functional dependencies :

DUT_NUMBER --> DEPT

Figure 22. CR_DUTIES relation that is not in BCNF.

CR_DUT

Key : LAST_NAME, DUT_NUMBER

LAST_NAME	DUT_NUMBER
HARRISON	43147
COLEMAN	57439
HARRISON	63247

DUT_DEPT

Key : DUT_NUMBER

DUT_NUMBER	DEPT
43147	ELECTRONIC
57439	OPERATION
63247	SUPPLY

Figure 23. Decomposed relations that are in BCNF.

To eliminate this anomaly, the transitive dependency must be removed by breaking the CREW_DUTIES relation into two relations: CREW and DUTIES as depicted in Figure 21.

4. Boyce-Codd normal form (BCNF)

"A relation is in BCNF if every determinant is a candidate key" [Ref. 4:pp. 144-145].

The example in Figure 22 is not in BCNF. The DUT_NUMBER attribute which is the determinant of the DEPT attribute is not a candidate Key. That means the relation CR_DUTIES has anomalies and must be decomposed into two relations having no anomalies. The decomposed relations CR_DUT and DUT_DEPT have no more anomalies and they are in BCNF as they are depicted in Figure 23. Although BCNF is very desirable, its existance is not assured nor is there any method to find

it even when it exists. Thus, in practice, 3NF is generally accepted as the solution.

I. APPLYING NORMAL FORMS TO SDM DESIGN

The administration office application requirements have been developed into the SDM design and constitutes Appendix A of this thesis. Before we apply the normal forms to the SDM design we have to determine the functional dependencies of each relation (entity class).

Recall that a Key is an arbitrary attribute or set of attributes that uniquely identifies each record, and the key functionally determines the entire row in a relation (i.e. all non-key attributes). In a relation, two or more keys may exist; however, only one is named primary key and the others are called candidate keys. In general, we need to consider the properties of functional dependencies in order to find the keys of the relation. The functional dependencies are applied under the database working environment. That means functional dependencies that are obvious or necessary for other working environments are not applied to our database system design. Figure 24 depicts the SDM designed relations with their primary keys, candidate keys, functional dependencies, notes about the relationship of each entity with other entities and a description of the subattributes used.

The entity CREW_PROF has two functional dependencies: CLASS -> RANK and EXIT_PERM -> DATE_IN. Both functional

dependencies imply anomalies and according to 3NF they must be split into two relations. However, for our design problem, this is not necessary. The entity LEAVE has functional dependencies between the attributes ST_LDATE, E_LDATE, and LPERIOD. Since these functional dependencies in the administration environment do not affect the structure of the designed database system, we do not apply the normal forms to the LEAVE entity class. Similarly for the PUNISHMENTS entity class we observe that the same functional dependencies exist as in the LEAVE entity class and for the same reasons we do not apply the normal forms.

In the database environment, a relationship is the method by which records in one relation can be associated with records in other relations. When considering a relationship that involves only two record types,this relationship is a binary relationship. A binary relationship is the main building block for constructing objects. There are three types of binary relationship: one-to-one, one-to-many, and many-to many.

1. <u>One-to-one(1:1).</u>

"An object relationship is one-to-one if Object A contains Object B as a single valued object property, and either Object B contains Object A as a single valued Object property or Object B does not contain Object A". The one-to-one relationship is represented by making the key of one relation an attribute of other relation. In this

relationship, the relation that will take the key of the other relation is insignificant and this key is called a foreign key. [Ref. 4:pp. 169-174]

2. One-to-many (1:M).

In a 1:M relationship, a record of one type is related to potentially many records of another type. The one-to-many relationship is represented by making the key of the parent relation an attribute of the child relation. [Ref. 4:pp. 174-178]

3. Many-to-many (M:N).

"In an M:N relationship, a record of one type corresponds to many records of the second type and a record of the second type corresponds to many records of the first type" [Ref. 4:pp. 178-182]. The many-to-many relationship is not represented as one-to-many and many-to-one relationships (i.e. by making the key of one relation an attribute in the other relation). Instead, creation of a third relation is necessary. This third relation is sometimes called an intersection relation because it represents the intersection of the two relations involved. The key for an intersection relation is always the combination of the parent keys.

The notes under each entity class in Figure 24 describe indirectly the type of the relationship between the involved entity classes. From these notes we can observe that some entities classes are related many-to-many, some of them one-to-many, and some one-to-one. The only entity class

```
CREW_PROF (Cr_number, Rank, Specialty, Name, Class
           Class_pos, Exit_perm, Date_in, Dut_num
           Cr_leave, Cr_punishment, Cr_training,
           On_duty )

     Key : Cr_number

     Candidate key : (Specialty, Class, Class_
                      pos)


     Functional dependencies : Class --> Rank
                               Exit_perm --> Date_in

     Notes : 1.  Name is combination of F_name,
                 L_name.
             2.  On_duty is combination of Dut_date,
                 Role.
             3.  Dut_num is a contained attribute
                 Dut_number of DUTIES; multivalued.
             4.  Cr_leave is a contained LEAVE tuple;
                 multivalued
             5.  Cr_punishment is a contained PUNI-
                 SHMENTS tuple; multivalued
             6.  Cr_training is a contained attribute
                 T_number of TRAINING_COURSES; multi-
                 valued


CREW_STATUS (Cr_num, Marital_status, Children, Bir-
             thdate, Indate, Outdate, Cr_town, Cr_
             hobbies, Educations, Previous_occupa-
             tions)

     Key: Cr_num

     Functional dependencies : None

     Notes : 1.  Cr_num is the same with Cr_number
                 of Crew_Prof.
             2.  Cr_town is a contained TOWN tuple;
                 multivalued.
             3.  Cr_hobbies is a contained attribute
                 H_num of HOBBY; multivalued.
```

Figure 24. Summary of the Conceptual design.

TOWN (Cr_names, T_type, T_add)

Key : (Cr_name, T_type)

Candidate Key : Telephone

Functional dependencies : None

Notes : 1. T_abb is combination of T_types Address, Area, Town_add, telephone.
2. Telephone CAN BE a Key but in our database system environment this is not usefull.

HOBBY (H_num, H_type, H_description, H_specialty, Cr_names)

Key : H_num

Candidate Key: (H_type, H_description, H_specialty)

Functional dependencies : None

Note : Cr_names is a contained attribute Cr_hobbies of CREW_STATUS; multivalued.

EDUCATION (E_num, Degree, E_type, School, Cr_names)

Key: E_num

Candidate Key : (Degree, E_type, School)

Functional dependencies : None

Note : Cr_names is a contained attribute Cr_educations of CREW_STATUS ; multivalued.

Figure 24. Summary of the Conceptual design (cont 'd).

```
PREVIOUS_OCCUPATION (Pr_num, Pr-type, Place, Cr_na-
                          mes)

    Key : Pr_num

    Candidate Key : (Pr_type, Place)

    Functional dependencies : None

    Note :  Cr_names is a contained attribute Pre-
            vious_occupations of CREW_STATUS ;
            multivalued.

LEAVE  (Le_name, L_data, Cr_names, Sub_number)

    Key : (Le_name, St_ldate)

    Candidate Key : (Le_name ,E_ldate)

    Functional dependencies :
              (St_ldate, E_ldate) --> Lperiod
              (St_ldate, Lperiod) --> E_ldate
              (Lperiod, E_ldate)  --> St_ldate

    Notes : 1.  L_data is combination of St_ldate,
                E_ldate, Lperiod, Ldestination,
                Ldescription.
            2.  Sub_number is a  contained attribute
                Dut_num of CREW_PROF.

PUNISHMENTS (P_name, P_data, Cr_names)

    Key : (P_name, St_pdate)

    Candidate Key: (P_name, E_pdate)

    Functional dependencies :
              (St_pdate, E_pdate) --> Pperiod
              (St_pdate, Pperiod) --> E_pdate
              (E_pdate, Pperiod) --> St_pdate

    Note :  P_data is combination of St_pdate, E_
            pdate, Pperiod.
```

Figure 24. Summary of the Conceptual design (cont 'd).

TRAINING_COURSE_INFO (T_number, T_name, T_description, T_place, T_crew)

Key : T_number

Candidate key : (T_name, T_description)

Note : T_crew is a combination of T_rank, T_specialty.

TRAINING_COURSES (T_numbers, T_course, Cr_names)

Key : (T_numbers, St_tdate)

Candidate key : (T_numbers, E_tdate)

Functional dependencies : None

Notes : 1. T_course is a combination of St_tdate, E_tdate, Max_no_of_person, Hours_per_day, St_hour
2. Cr_names is a contained attribute Cr_training of CREW_PROF ; multivalued

DUTIES (Dut_number, Crew_num, Vis_num, D_rank, D_specialty, Two_shifts, Three_shifts, D_dock, Abandon, Alert)

Key : Dut_number

Candidate Key : (Two_shifts)
(Three_shift)
(Alert)

Functional dependencies : None

Notes : 1. Two_shifts is a combination of Two_section, Two_pos, Two_duties.

Figure 24. Summary of the Conceptual design (cont 'd).

2. Three_shifts is a combination of Th_section, Th_pos, Th_duties.
3. Alert is a combination of Al_pos, Al_duties.
4. Crew_num is a contained attribute Dut_num of CREW_PROF; multivalued.
5. Vis_num is a contained attribute V_number of VISITORS; multivalued.

VISITORS (V_number, V_title, V_rank, V_name, V_period)

Key : (V_number, V_date)

Candidate key : (V_title, V_rank, V_name, V_date)

Functional dependencies : None

Note : V_name is a combination of V_fname, V_lname.

DAILY_ACTIVITIES (Act_num, Activities, Cr_names)

Key : Act_num

Functional dependencies : None

Note : Cr_names is a contained attribute On_duty of CREW_PROF; multivalued.

Figure 24. Summary of the Conceptual design (cont 'd).

that is not related to the other entity classes is the EXERCISES entity class. This entity is not related to others according to application requirements; it is used only to check leave periods and training courses periods to assure they do not conflict with an exercise, as this constitutes a problem for the application program. The relationships

MANY-TO-MANY		INTERSECTION
CREW_STATUS	HOBBY	CR_HOBBY
CREW_STATUS	EDUCATION	CR_EDUC
CREW_STATUS	PREVIOUS_OCCUPATION	CR_OCCUP
CREW_PROF	TRAINING_COURSES	TRAINING_CREW
CREW_PROF	DAILY_ACTIVITIES	CR_DAILY
CREW_PROF	DUTIES	CR_DUT

ONE-TO-MANY	
CREW_STATUS	TOWN
CREW_PROF	LEAVE
CREW_PROF	PUNISHMENTS

ONE-TO-ONE	
CREW_STATUS	CREW_PROF
DUTIES	VISITORS
TRAINING_COURSE_INFO	TRAINING_CREW
TRAINING_COURSE_INFO	TRAINING_COURSES

NO RELATIONSHIP
EXERCISES

Figure 25. SDM relationships.

between the entity classes as they are defined in the notes of each entity classes and the necessary intersection for many-to-many relationships are shown in Figure 25.

J. ENTITY-RELATIONSHIP MODEL (E-R MODEL)

As previously metioned, the third step of the conceptual design is the conceptual schema development. In this step the normalized SDM design must be transformed into the Entity-Relationship diagrams. The Entity-Relationship model was developed by Chen (1977) for external and conceptual levels. As with SDM, the E-R model is based on the real world objects and represents them as entities and relationships among these objects. The overall logical structure of a database is a graphical expression by an E-R diagram which uses the following components:

1. Double rectangles to represent entity sets.
2. Rectangles to represent attributes.
3. Diamonds to represent relationship sets.
4. Lines: Single to link attributes to entity sets; and double ones, to link entity sets to relationship sets.

Each of these components is labeled with its corresponding name. The different types of relationships are described on linked double lines. The one-to-many relationship is represented by the word "FROM" followed by the connected entity class name showing the relationship between parent and child. The many-to-many relationship is represented by the words "TO / FROM" followed by the connected entity classes

showing the intersection between these two entity classes. As previously mentioned, the intersection can contain the keys of the related two entity classes, plus some other attributes. These extra attributes may come from one or both of the related entity classes, or even some new ones to support the many-to-many relationship. These new attributes are represented by rectangles connected directly to the related diamonds.

The administration office E-R diagram using the normalized SDM design is shown in Appendix A Figure A-3. Therefore, the E-R diagram does not need any more normalization and is included for better understanding, as the "Conceptual scheme development" step in Figure 10 describes.

K. SUMMARY OF DESIGN

In this chapter, the administration office database system was designed using the SDM design model and E-R diagram to show the entities, the attributes in them, and the connected relationships.

The relational database model that will be used in the implementation chapter is represented by a collection of tables. These tables are the translation of Figure A-3 (E-R diagram) in Appendix A. For each entity set and for each relationship set in the database, there is a unique table which is assigned the name of the corresponding entity set, or relationship set from the E-R diagram. Each table has a

number of columns with unique names which, again, corresponds to the E-R diagram attributes sets. [Ref. 5: Chapter 2]

V. DESIGN SUPPORT AND CONFLICT ANALYSIS

A database system is supported by a number of programs, named "application programs". The purpose of these application programs is to process the user request for data. For this reason, the constraints and conflicts of the database system must be stated clearly and completely to help the designer to encode the application programs for the particular database system.

A. DESIGN SUPPORT

As previously mentioned, the SDM model is a collection of entities which are organized into classes. Each class has a number of attributes which represent the properties of the described object. Each attribute has a number of descriptors as shown in Figure 13, and they are used to represent the properties of the described object. The purpose of each descriptor that is used by an attribute has been stated. These descriptors form the constraints of the administration database system and they are included in the SDM design shown in Figure A-1 in Appendix A.

Figure 26 summarizes the constraints that will be used to encode the application programs in the next chapter.

ATTRIBUTES	CHARACTERISTICS	DOMAIN
CREW_PROF		
CR_NUMBER	MANDATORY NOT CHANGEABLE	CHAR(5)
RANK	MANDATORY	CHAR(10); VALUE SET IS: RANKS IN THE NAVY
SPECIALTY	MANDATORY NOT CHANGEABLE	CHAR(13); VALUE SET IS: SPECIALTIES IN THE NAVY
F_NAME	MANDATORY NOT CHANGEABLE	CHAR (12)
L_NAME	MANDATORY NOT CHANGEABLE	CHAR(15)
CLASS	MANDATORY NOT CHANGEABLE	POSITIVE INTEGER; RANGE 50-95
CLASS_POS	MANDATORY NOT CHANGEABLE	POSITIVE INTEGER; RANGE 1-50
EXIT_PEAM		POSITIVE INTEGER RANGE 1-5
DATE_IN		DATE
DUT_DATE		DATE
ROLE		CHAR(12)
CREW_STATUS		
MARITAL_STATUS		VALUE SET IS: 'MARRIED' 'UNMARRIED' 'DIVORCED'
CHILDREN		POSITIVE INTEGER RANGE 0-5
BIRTHDATE	MANDATORY NOT CHANGEABLE	DATE
INDATE	MANDATORY NOT CHANGEABLE	DATE
OUTDATE		DATE

Figure 26. Domain and intra-relation constraints.

ATTRIBUTES	CHARACTERISTICS	DOMAIN
TOWN		
T_TYPE	MANDATORY	VALUE SET IS: 'ORIGINAL' 'RELATIVE' 'NEAR BASE' 'POLICE'
ADDRESS		CHAR(20)
AREA		CHAR(12)
TOWN_ADD		CHAR(12)
TELEPHONE		CHAR(11)
HOBBY		
H_NUM	MANDATORY NOT CHANGEABLE	POSITIVE INTEGER RANGE 0-99
H_TYPE	MANDATORY NOT CHANGEABLE	VALUE SET IS: 'SOCCER' 'BASKETBALL' 'TENNIS' 'VOLLEYBALL' 'SKI' 'PING-PONG' 'MUSIC' 'SWIM' 'OTHER'
H_DESCRIPTION		CHAR(10)
H_SPECIALTY	MANDATORY	VALUE SET IS: 'LISTENING' 'PLAYING' 'WATCHING' 'DOING' 'OTHER'
EDUCATION		
E_NUM	MANDATORY NOT CHANGEABLE	POSITIVE INTEGER RANGE 0-99
DEGREE		CHAR(10)
E_TYPE		CHAR(10)
SCHOOL		CHAR(10)

Figure 26. Domain and intra-relation constraints (cont 'd).

ATTRIBUTES	CHARACTERISTICS	DOMAIN
PREVIOUS_OCCUPATION		
PR_NUM	MANDATORY NOT CHANGEABLE	POSITIVE INTEGER
PR_TYPE		CHAR(10)
PLACE		CHAR(10)
LEAVE		
LE_NAME	MANDATORY	VALUS SET IS: 'STANDARD' 'EXTRA' 'ON_SEA' 'SPECIAL' 'DUTY EXCEPTION' 'HOSPITAL LEAVE'
ST_LDATE	MANDATORY	DATE
E_LDATE		DATE
LPERIOD	MANDATORY	POSITIVE INTEGER RANGE 1-30
L_DESTINATION		CHAR(12)
L_DESCRIPTION		CHAR(10)
PUNISHMENTS		
P_NAME	MANDATORY	VALUE SET IS: 'PRISON ASHORE' 'PAYBACK CONFINEMENT' 'CONSECUTIVE DAY CONFINEMENT' 'ALTERNATE DAY CONFINEMENT'
ST_PDATE	MANDATORY	DATE
E_PDATE		DATE
PPERIOD	MANDATORY	POSITIVE INTEGER RANGE 1-30
TRAINING_COURSE_INFO		
T_NUMBER	MANDATORY NOT CHANGEABLE	POSITIVE INTEGER RANGE 0-9999
T_NAME	MANDATORY NOT CHANGEABLE	CHAR(10)

Figure 26. Domain and intra-relation constraints (cont 'd).

ATTRIBUTES	CHARACTERISTICS	DOMAIN
T_DESCRIPTION	NOT CHANGEABLE	CHAR(10)
T_PLACE		CHAR(10)
T_RANK		CHAR(10); VALUE SET IS: RANKS IN THE NAVY
T_SPECIALTY		CHAR(13); VALUE SET IS: SPECIALTIES IN THE NAVY
TRAINING_COURSES		
ST_TDATE	MANDATORY	DATE
E_TDATE	MANDATORY	DATE
MAX_NO_OF_ PERSONS	MANDATORY	POSITIVE INTEGER RANGE 5-20
HOURS_PER_DAY	MANDATORY	POSITIVE INTEGER RANGE 1-4
ST_HOUR	MANDATORY	CHAR(5) IN FORMAT HH.MM
DUTIES		
DUT_NUMBER	NOT CHANGEABLE	CHAR(5)
D_RANK	NOT CHANGEABLE	CHAR(10); VALUE SET IS: RANKS IN THE NAVY
D_SPECIALTY	NOT CHANGEABLE	CHAR(13); VALUE SET IS: SPECIALTIES IN THE NAVY
TWO_SECTION	NOT CHANGEABLE	VALUE SET IS: 'LEFT' 'RIGHT'
TWO_POS	NOT CHANGEABLE	CHAR(10)
TWO_DUTIES	NOT CHANGEABLE	CHAR(12)
TH_SECTION	NOT CHANGEABLE	VALUE SET IS: 'A' 'B' 'C'
TH_POS	NOT CHANGEABLE	CHAR(10)
TH_DUTIES	NOT CHANGEABLE	CHAR(12)
D_DOCK	NOT CHANGEABLE	CHAR(12)

Figure 26. Domain and intra-relation constraints (cont 'd).

ATTRIBUTES	CHARACTERISTICS	DOMAIN
ABANDON	NOT CHANGEABLE	CHAR(2)
AL_POS	NOT CHANGEABLE	CHAR(10)
AL_DUTIES	NOT CHANGEABLE	CHAR(12)
VISITORS		
V_TITLE	MANDATORY	CHAR(10)
V_RANK	MANDATORY	CHAR(10)
V_FNAME	MANDATORY	CHAR(12)
V_LNAME	MANDATORY	CHAR(15)
V_DATE	MANDATORY	DATE
V_PERIOD	MANDATORY	POSITIVE INTEGER RANGE 1-99
DAILY_ACTIVITIES		
ACT_NUM	MANDATORY	POSITIVE INTEGER RANGE 1-40
ACTIVITIES	MANDATORY	CHAR(20)
EXERCISES		
E_NAME	MANDATORY	CHAR(10)
ST_EDATE	MANDATORY	DATE
E_EDATE		DATE
EPERIOD	MANDATORY	POSITIVE INTEGER RANGE 1-30
AREA		CHAR(10)

Figure 26. Domain and intra-relation constraints (cont 'd).

B. CONFLICTS ANALYSIS

In Chapter III, the administration office application requirements were described and some of the conflicts were identified. These requirements have been divided into four categories according to how frequently each administration task is repeated. These categories will be examined again to establish more precisely the conflicts among the entity

classes. Each category may have one or more reports that the designed database system must provide for the administration office. These reports are actually a collection of specified data and do not involve any conflict among the entity classes. For that reason, the reports will not be examined.

1. Daily application requirements.

This category includes leave and punishments.

a. Leave

The conflicts of leave entity class are :

(1) Reschedule of a leave period must be made if the crewmember is punished for this period. Exception of this rule is when the crewmemebers must take a "special leave" or "hospital leave"

(2) A leave period must not overlap with an exercise period or a training course that the crewmember must attend. The executive officer has the authority to make an exception to this rule.

(3) Crewmember on leave must not be assigned with duties on dock for the leave period.

b. Punishments

Punishment conflicts are :

(1) Crewmembers with punishment type " PRISON ASHORE" must be considered as "out of board".

(2) Under punishment crewmember who makes use of the exception and takes "special leave " or hospital leave" must continue his punishment period after finishing the above exception leave type period.

2. Weekly application requirements

This category includes training-courses. The conflicts of training-courses are :

a. The selection of the participating crewmembers must be made based on their rank and specialties with the

predefined RANK and SPECIALTIES in the TRAINING_COURSE_INFO entity class and on the oldest date crewmembers participated if the MAX_NO_OF_PERSON is not yet completed or if capacity is exceeded.

b. The participating crew list must be able to be updated in case that, under the approval of the executive officer, one or more of these crewmembers may be on any type of leave during that period.

c. The training courses must not overlap with an exercise period.

3. Monthly application requirements

This category includes only reports

4. On-demand application requirements

This category includes crewmember status and professional data, duties ,visitors, and exercises.

a. Crewmember status and professional data.

The following conflicts belong to this class.

(1) The duties of a crewmember on leave by order must be assigned to another person before he leaves the ship.

(2) A new-comer crewmember may or may not be assigned with the duties of a particular crewmember.

(3) Every list must be shorten according to RANK_SPECIALTY_CLASS_CLASS_NO of the listed crewmembers.

b. Duties

The database system must not allow one or more duties to be unassigned.

c. Visitors and Exercises

These entity classes do not involve any conflict since the first entity class is related only to the duties entity class and the latter is not related to any other relation.

C. SUMMARY

In this Chapter, the constraints and conflicts of the designed database system for the administration office have been described. These constraints and conflicts with the entity classes of the E-R diagram compose the main sources for development of the application programs.

The chapter also gives the final step of the conceptual design of the database system for the administration office. The constraints and conflicts stated in the chapter must be solved by the application programs in accordance with the rules that govern the operation.

VI. IMPLEMENTATION

The conceptual design uses several data model to organize its data and it is completely independent of any database management system or any other software or hardware considerations. The next two steps following the conceptual design are implementation design and physical design. These two steps refine the conceptual design so that it can be implemented on the DBMS used by the administration office.

Implementation design is the mapping of the conceptual design into a DBMS logical model: hierarchical, network, or relational data model. Physical design is the process of selecting the appropriate file organizations, access methods, and related factors. The major inputs to physical design are the logical structure from implementation design. Both designs must be done carefully, since they affect performance, security, and a number of other factors. [Ref. 3:p. 278]

The features of the database management system which will be used and its hardware requirements will be represented before the presentation of the implementation and physical system.

A. SOFTWARE REQUIREMETS

Dbase III plus is the DBMS that will be used to implement the administration office designed database system. The Dbase III plus is a relational DBMS and is designed to run on IBM PC microcomputers. This DBMS has a number of important features as well as limitations described below. [Ref. 10 :pp. 17-24]

1. Features of dbase III plus

a. Dbase III provides the basic data types: characters, numerics, and logicals. In addition to them, it provides date and memo data types that are powerful tools for date and text managing.

b. The information is stored as disk files in thirteen specialized format each serving a specific dbase III plus need.

c. Intefacing with other software systems is allowd by dbase III plus.

d. The application programs are independent of changes in file structure.

e. Data can be easily updated, sorted, and indexed.

f. Reports in special formats are provided by dbase III plus.

2. Limitations of dbase III plus [Ref. 11:pp.1-2]

a. Database file:
   Number of records: 1 billion maximum
   Number of bytes: 2 billion maximum
   Record size: 4000 bytes

b. Field size:
   Character fields: 254 bytes maximum
   Data fiels: 8 bytes
   Logical fiels: 1 byte
   Memo fields: 5000 bytes maximum
   Numeric fields: 19 bytes maximum

c. File operations:
   15 open files of all types
   10 open database files
   7 open index files per active database file
   1 open format file per active database file.

d. File names can be up to 8 characters long and field names can be up to 10 characters long. These characters can be letters, numbers, or the underscore character (_) and no blank characters are allowed.

e. Active memory variables: 256.

All values may be limited by the computer hardware configurations.

B. HARDWARE REQUIREMENTS

Dbase III plus requires MS-DOS or PC DOS version 2.0 or later and can be operated on a 16-bit microcomputer IBM PC or IBM compatible.

The minimum computer memory required is 256K, more memory usually results in increased processing speed. The system should have one 360K floppy disk and a hard drive, or two 360K floppy disk drives. However, a hard disk drive of 5M or bigger is recommended for faster proccessing and large databases.

Another hardware requirement is any printer with a capacity at least 80 columns and which is able to interface with the above mentioned microcomputer. [Ref. 10:p. 23]

C. IMPLEMENTATION DESIGN

As mentioned above, one of the main tasks and time consuming jobs of the administration office is the assignment

of duties to each crewmember. For that reason, the CREW_PROF and DUTIES entity classes will be demonstrated as implementation of this thesis. These two entity classes are related with many-to-many relationship so a third relation is necessary to represent the intersection of the implemented entity classes as shown in E-R diagram (Appendix A Figure A-3). Since the dbase III plus is a relational DBMS ,tables according to relational data model theory must be created. The created tables must have one row per record occurance and one column per attribute. The intersection relation includes three attributes, two attributes repeat the primary key of each related entity class and an extra attribute that characterize the duties as "M" (main duty) or "S" secondary duties, because a crewmember may have been assigned more than one duty number.

In addition to these three relations, two extra relations will be used: PASSWORD and RANK_SPE. The first relation is used to increase the security of the system by giving permission only to authorized users who know the passwords. The second relation is used to stored the RANK, SPECIALTY, and their CODE number. The latter relation provides the database system more friendly to users. This means that users do not have to remember any code number when an insertion action takes place for a new crewmember. Instead, insertion of the crewmembers RANK and SPECIALTY can be done directly and the system is responsible for validating these data.

In case of a wrong insertion, the system automatically displays on the screen the acceptable set of values.

The CREW_PROF entity class will be tested with 50 different crewmember names and the DUTIES entity class with 53 different groups of duties (duty-numbers). The intersection CREW_DUT entity class will include 55 records with 3 crewmembers having two different duties and 2 crewmembers with the same duties (many-to-many relationship).

D. PHYSICAL DESIGN

Physical design is the process that takes the logical model created by the implementation design to develop an efficient, implementable physical database structure. Dbase III plus is a powerful DBMS that helps the designer to develop the necessary relations and supporting application programms so that the user can navigate through the database system easily and with minimal training or experience. [Ref. 3:p.294]

In dbase III plus, each relation is stored as a seperate file and the data are retrieved by the application programs. By using the dbase III plus facilities of indexing and join these data are retrieved according to administration requirements (i.e. lists are sorting according to RANK, SPECIALTY, CLASS, and CLASS_POS). Finally by joining the two basic relations CREW_PROF and DUTIES, the system can provide any

necessary requested information by the administration office.

The administration database system is developed under a "menu driven" system in which the user is led to answer the questions provided by the system and to go through the main menu. For better understanding of the various system actions provided by the application programs, a description of these actions is given in the following sections.

E. RUNNING THE SYSTEM

To start running the administration database system, the user has to call the "MAIN PROGRAM" by its name (DO MAIN). The program starts running by asking the user to insert his password. If the password is incorrect then the program provides the message "UNAUTHORIZED USER" and returns to the operating system (DOS). The use of passwords protects the database system for unauthorized users to get information from the system.

If the user of the system inserts the correct password, then the main program presents to the user the "MAIN MENU" with a set of choices, as shown in Figure 27. The choice "0" returns to the operating system and the choice "3" to dbase III plus at the "dot_prompt mode". The last option is for programmers when modifications must be done in the application programs or in the structure of the database system. The rest of the choices enables the main program to call the

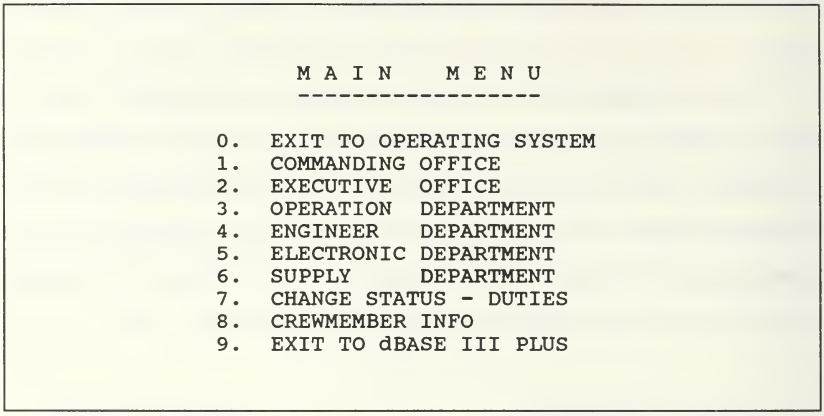


Figure 27. Main menu.

appropriate application programs which, in turn, presents their own submenu with a new set of choices. In this way, the user can navigate in the database system or get the desired action. In the designed implementation three maximum levels of submenus exist. The main menu choices can be divided into three categories; choices 1-6 which represent crewmembers duties in groups according to department and subdepartment, choice 7 which lets the user do the basic functions in the database system (i.e. delete, insert, update, or change duty), and choice 8 which presents crewmembers' lists in groups of the basic categories in the Navy (i.e. Officers, N.C.O., seamen) and duties of each crewmember. These three categories will be described in the next sections.

A small program named "DELAY" is called by the application programs to produce a small delay whenever it is necessary; otherwise, the inputs would not be viewable by the user.

F. CREWMEMBER_DUTY

The stored data in each relation can be viewed in many different ways, usually limited by the working enviroment. In the administration database system implementation, all the needed lists are supported.

The main menu choices 1 and 2 provide the crewmembers and their duties who work in the C.O.'s and the X.O.'s (administration office) offices, respectively. Choices 3-6 provide submenus of the subdepartments that belong to the selected department. Then, the user can choose one particular subdepartment or all of them in case he wants to get all the crewmembers' duties of the department as shown in Figure 28 (submenu for the "Operation department"). After selecting the department and subdepartment, a special procedure in the database system permits the user to get any combination of duties under different ship environments (i.e. Alert, two shifts, etc.) by typing a combination of the appropriate numbers in the submenu. In this way, the database system supports all the lists that the department and subdepartment officers may need.

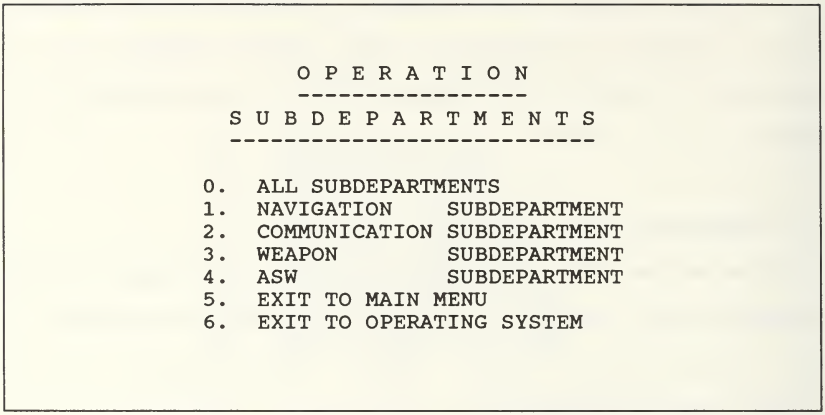


Figure 28. Operation department submenu.

G. BASIC FUNCTIONS

As previously mentioned, the basic functions in our working environment are insert new crewmember, delete a crewmember, update a crewmember, or change crewmembers' duties.

These functions are provided as a submenu by the database system when Choice 7 is selected in the main menu. This submenu is shown in Figure 29.

1. Insert new crewmember

When the "insert new crewmember" option has been selected by the user, the "newcrew" program is called. This program interacts with the user by providing the necessary fields that must be completed by him. The program checks each inserted data item for validity, and lets the user insert

again his data for a specific field in case of incorrect insertion. The program checks the database for the inserted CR_NUMBER to determine if this CR_NUMBER already exists.

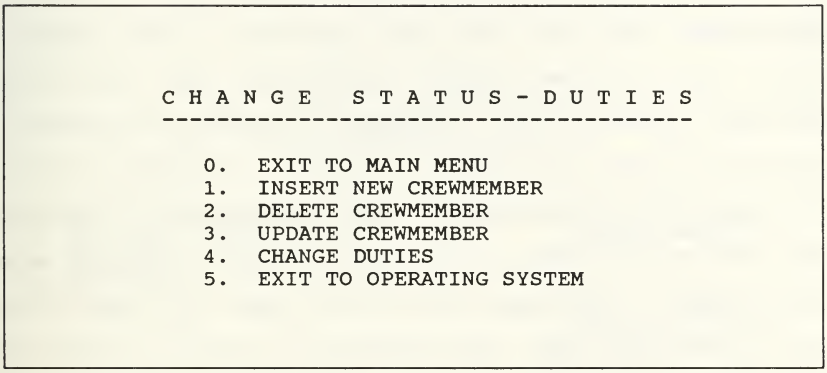


Figure 29. Change status-duties submenu.

Since the database system does not include any code, the user has to insert a valid rank and specialty in the corresponding fields asked by the system. In case of incorrect insertion, the system provides the acceptable ranks and specialties until a valid value is inserted. After all data have been inserted, the user is asked to review these data and makes any necessary corrections. This protects the database system from getting valid data but with errors. For example, the CR_NUMBER of "35652" will be typed as "35662" which is valid data but incorrect.

All crewmembers on the ship have to be assigned a group of duties for different environments even though the new crewmember is not ready to accept his duties. In that case, he shares his duties with one or more crewmembers. For this reason, the new crewmember must be assigned at least one duty-number after his data are inserted in the database system. The "inscrew" program supports the user for making decisions about the duty assignment by asking the necessary questions and providing a list of acceptable duties according to the crewmember's rank and specialty. The user can ask for more duties. The program is able to provide more duty lists of the same specialty of lower or higher rank that his current rank. The flowchart in Figure 30 shows how the "insecrew" program works.

2. Update crewmember

When the "update a crewmember" option has been selected by the user, the "upcrew" program is called. This program permits the user to update the fields RANK, EXIT_PERM, and DATE_IN since all the other fields in the CREW_PROF relation are not changeable according to SDM design. The program prompts the user to fill up these fields and again the RANK field is checked for valid insertion. In case of incorrect insertion,the program displays the acceptable rank values until a valid value is inserted.

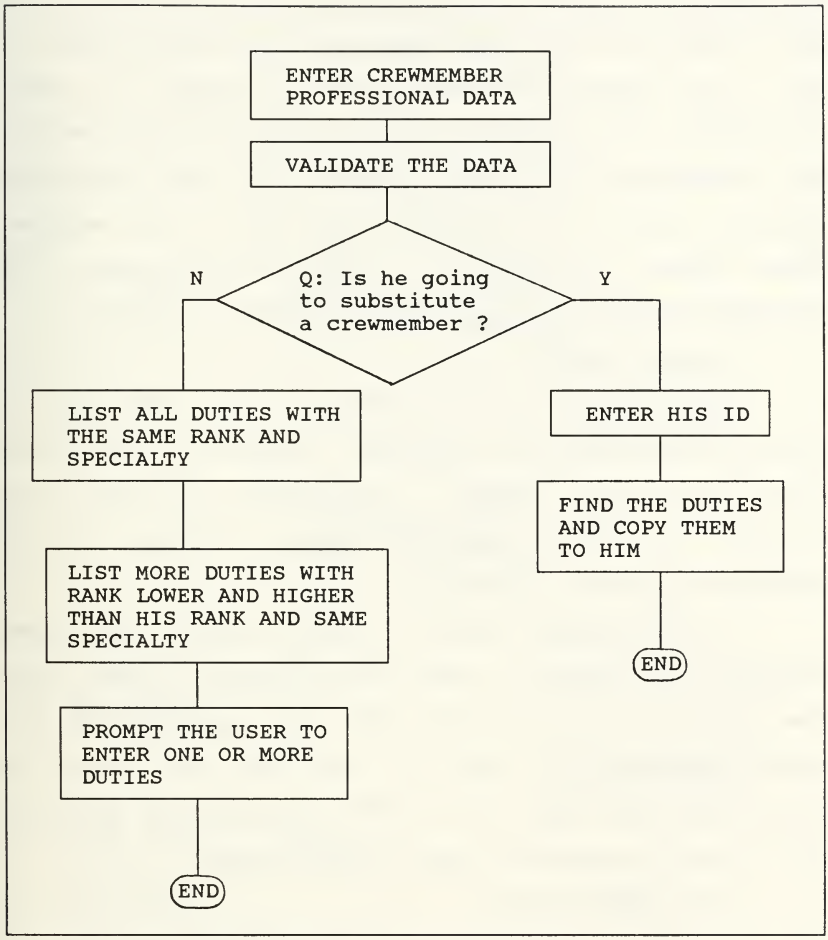


Figure 30. Flowchart of "insertion".

3. Delete crewmember

When the "delete a crewmember" option has been selected by the user, the "delcrew" program is called.

The programm prompts for the MIL ID to be inserted by the user. When the ID has been inserted, the program checks to see if his duties have been already assigned to other crewmembers. If its answer is "Yes", then the program deletes him. Otherwise, the program supports the user for making decisions about the crewmembers who have to be assigned these duties by providing a list of crewmembers. The flowchart for the "delcrew" program is shown in Figure 31.

4. Change duties

Many times, it is necessary for the administration office to reassign duties among the crewmembers. This option helps the user to perform a reassignment of duties. The flowchart of change duties in Figure 32 shows how the "chduties" program is developed.

This program displays a short submenu with three options: delete duties, assign duties, and all done. Since the program can be run for more than one crewmember, the last option terminates the user's job. The other two options are designed to help the user in making decisions by providing the necessary crewmember names or duty-numbers.

a. Delete duties.

When "delete duties" option has been selected the program prompts the user to insert the desired duty number. Then, the programs checks if this duty-number has been assigned to another crewmember. If the answer is "Yes", then it deletes the duty-number. Otherwise, the program

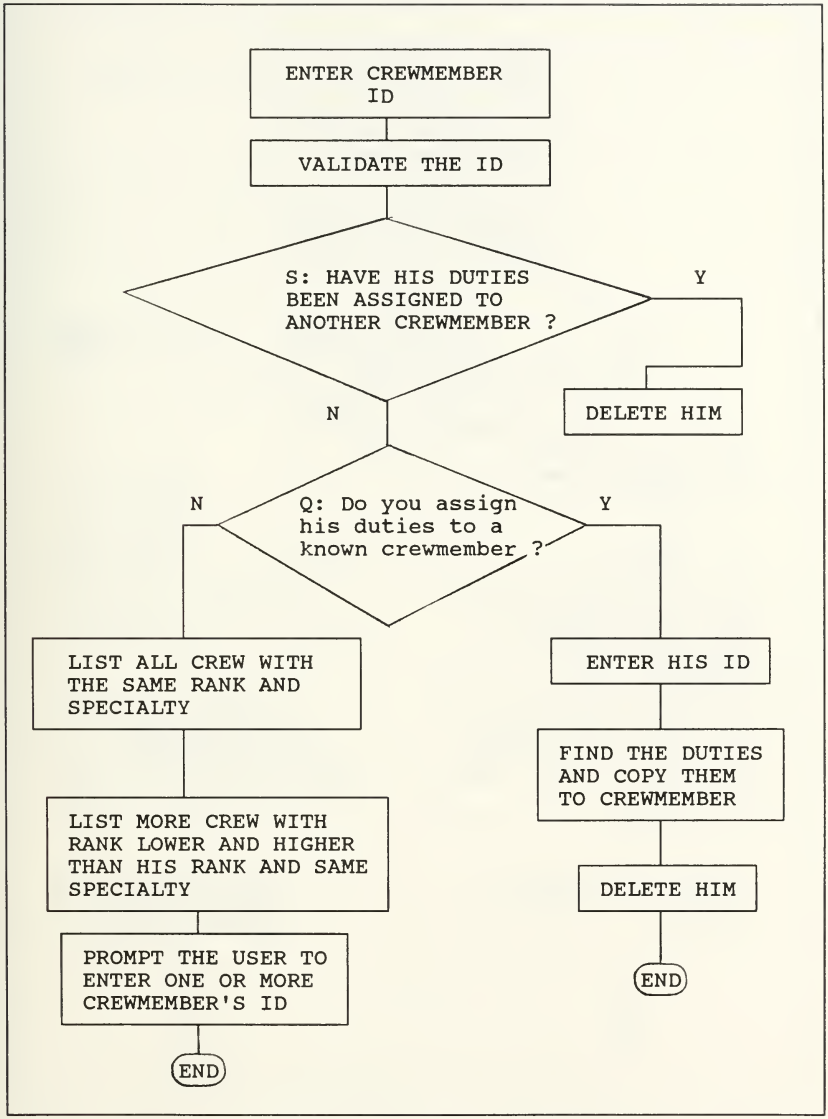


Figure 31. Flowchart of "deletion".

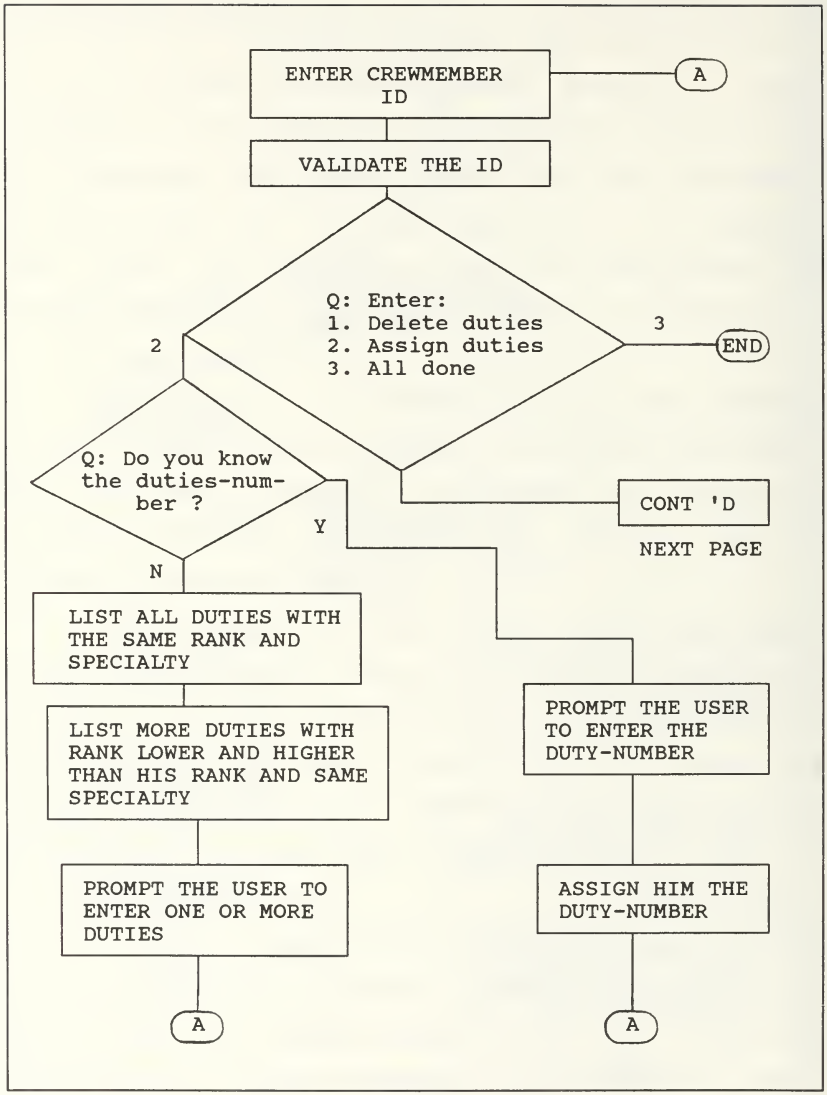


Figure 32. Flowchart of "change-duties".

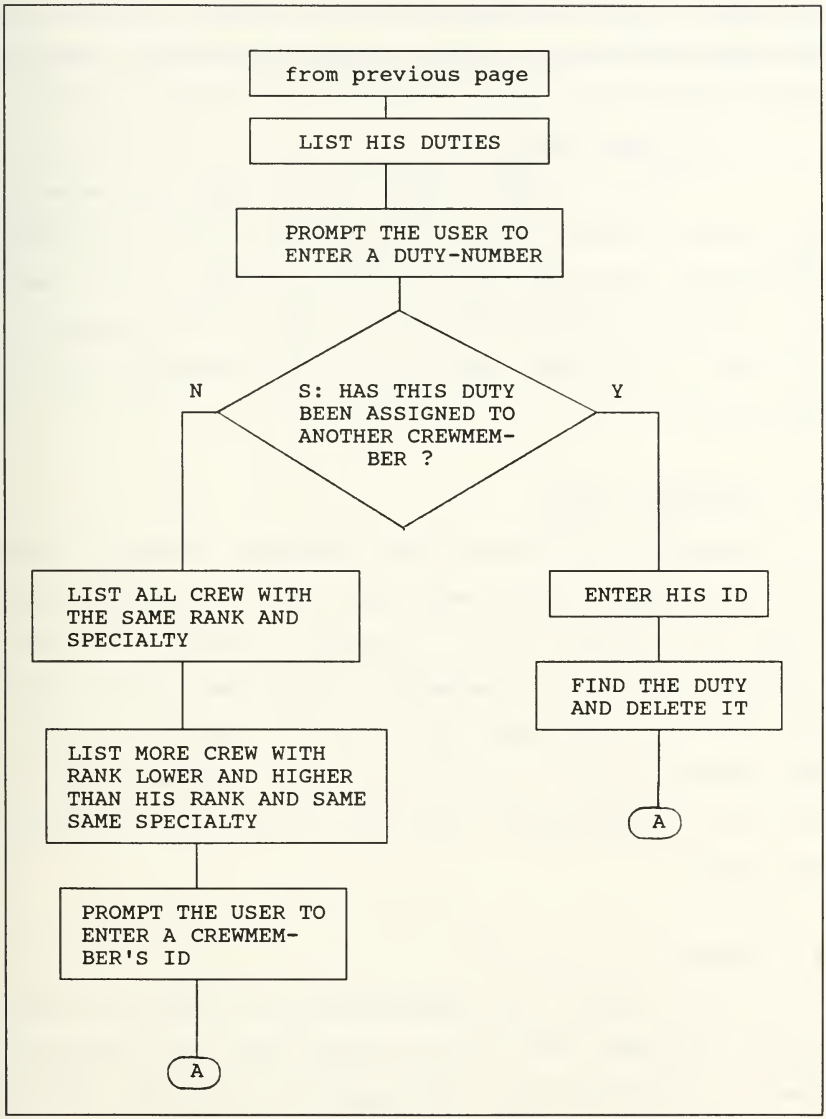


Figure 32. Flowchart of "change-duties" (cont 'd).

provides a list of crewmembers that can accept this duty-number, thus helping the user to make his decision. The user finally inserts the crewmember's ID of his choice.

b. Assign duties.

When "assign duties" option has been selected the program prompts the user to see if he has already decided the new duties. If the answer is "Yes", then it asks for the duty number. Otherwise, a list of duty numbers is provided to the user according to the crewmember rank and specialty.

H. CREWMEMBER LISTS

The option 8 of the main menu provides a submenu of lists that the database system must carry out for the user. These lists are officers, N.C.O., seamen, or duties of a particular crewmember. A crewmember may have been assigned with more than one duty-number but only one is named "main duty". The database system can provide all the crewmember's duties with their characteristic name (i.e. "main duty" or "secondary duty").

I. SUMMARY

In this chapter, the implementation and physical design of the CREW_PROF and DUTIES relations have been completed. These relations with the three supporting relations (CR_DUT, PASSWORD, and RANK_SPE) were implemented in an IBM computer

using dbase III plus. The application programs are given in Appendix B and some sample outputs (lists) are given in Appendix C.

VII. CONCLUSIONS

The purpose of this study was to develop a database system for personnel management and watch scheduling, implementable for a war ship. Three objectives of this thesis were stated in the first chapter; first, the presentation of design steps, the design criteria, and the evaluation of this design against the design criteria, second, the attempt to resolve conflicts and make the system easy to use and helpful for making decisions, and finally, the implentation of the designed database system using dbase III plus on IBM PC XT TURBO microcomputer.

The omission of some details describing the problem was necessary due to unclassified nature of this thesis. However, these details would not affect the database system design, and with some modifications or extensions, the design as presented in this thesis can be used for war ships of any size. At the same time, the design is presented in such a way to fit the standards used by most nations.

The database managent system has been implemented using dbase III plus. It is done with a high level programming language written for the IBM PC, based on the relational data model. The application programs supporting the designed database system has been developed with the goal of helping the users make decisions through a friendly user-interface.

The mechanism of "menu driven" approach was selected for this purpose, allowing users to use the system even without attending any training course or previous experience. Passwords were used to ensure that the system will be used by authorized users only.

APPENDIX A

The administration database system design is shown in this Appendix and includes the SDM design which is represented in Figure A-1, Figure A-2 shows the supporting the database system domain definitions, and finally, the Entity-Relationship diagram of the normalized SDM design is depicted in Figure A-3.

```
CREW_PROF

    description: Profesional information of each
                 crewmember on ship.

    member attributes:

        CR_NUMBER
            description: Crewmember's Military ID
            value class: CR_NUMBERS
            mandatory
            not changeable

         RANK
            description: Crewmember's rank
            value class: RANKS
            mandatory

        SPECIALTY
            description: Crewmember's specialty
            value class: SPECS
            mandatory
            not changeable

        NAME
            description: Crewmember's name

            subattributes:

                F_NAME
                    description: First name
                    value class: F_NAMES

                L_NAME
                    description: Last name
                    value class: L_NAMES
            mandatory
            not changeable

        CLASS
            description: Graduation year
            value class: NUMBER
            mandatory
            not changeable
```

Figure A-1. SDM design.

```
CLASS_POS
    description: Graduation number
    value class: NUMBER
    mandatory
    not changeable

EXIT_PERM
    description: Consecutive number of days
                 that is permitted to go
                 out of the ship after the
                 working hours
    value class: DIGIT

DATE_IN
    description: Calculated date according
                 to EXIT_PERM
    value class: DATES

DUT_NUM
    description: Code number of duties
                 on the ship
    value class: DUTIES
    inverse: CREW_NUM
    multivalued

CR_LEAVE
    description: Leave types and periods
    value class: LEAVE
    inverse: CR_NAMES
    multivalued

CR_PUNISHMENT
    description: Punishment types and periods
    value class: PUNISHMENTS
    inverse: CR_NAMES
    multivalued

CR_TRAINING
    description: Training courses that he has
                 attended
    value class: TRAINING_COURSES
    inverse: CR_NAMES
    multivalued
```

Figure A-1. SDM design (cont 'd).

```
        ON_DUTY
            description: Date and duty on dock
            value class: DAILY_ACTIVITIES
            inverse: ACT_NUM

            subattributes:

                DUT_DATE
                    description: Date of duty on dock
                    value class: DATES

                ROLE
                   description: Special duty on
                                dock after the wor-
                                king hours
                   value class: ROLES
        multivalued

    identifiers:
        CR_NUMBER


CREW_STATUS

    description: Status details about each crewmember
                 on the ship.

    member attributes:

        CR_NUM
            description: Crewmemmber's Military ID
            value class: CREW_PROF
            inverse: CR_NUMBER

        MARITAL_ STATUS
            description: Marital status of each crew-
                         member
            value class: CR_MAR

        CHILDREN
            description: Number of children
            value class: DIGIT
```

Figure A-1. SDM design (cont 'd).

```
        BIRTHDATE
            description: Date of his birth
            value class: DATE
            mandatory

        INDATE
            description: Date that the crewmember
                         came on the ship
            value class: DATES
            not changeable
            mandatory

        OUTDATE
            description: Date that the crewmember
                         is scheduled to leave by
                         order from the ship
            value class: DATES

        CR_TOWN
            description: Crewmember address
            value class: TOWN
            inverse: CR_NAMES
            multivalued

        CR_HOBBIES
            description: Hobbies preferences
            value class: HOBBY
            inverse: CR_NAMES
            multivalued

        EDUCATIONS
            description: The degree of education
                         of each crewmember
            value class: EDUCATION
            inverse: CR_NAMES
            multivalued

        PREVIOUS_OCCUPATIONS
            description: Occupations that the crew-
                         member did before he
                         joined in the Navy
            value class: PREVIOUS_OCCUPATION
            inverse: CR_NAMES
            multivalued

    identifiers:
        CR_NUM
```

Figure A-1. SDM design (cont 'd).

```
TOWN

    description: Crewmember 's address.

    member attributes:

        CR_NAMES
            description: Crewmember name
            value class: CREW_STATUS
            inverse: CR_TOWN

        T_TYPE
            description: Different types of crewmem-
                         ber address
            value class: T_TYPES
            mandatory

        T_ADD
            description: Address and telephone of
                         each crewmember

            Subattributes:

                ADDRESS
                    value class: ADDR

                AREA
                    value class: PLACES

                TOWN_ADD
                    value class: PLACES

                TELEPHONE
                    value class: PHONES


    identifiers:
        ADDRESS
```

Figure A-1. SDM design (cont 'd).

```
HOBBY

    description: Recreation activities.

    member attributes:

        H_NUM
            description: Hobby number
            value class: NUMBER
            mandatory
            not changeable

        H_TYPE
            description: Hobby type
            value class: H_TYPES
            mandatory
            not changeable

       H_DESCRIPTION
            description: Description of the hobby
                         type
            value class: DESCR

        H_SPECIALTY
            description: Specialty in the Hobby type
            value class: H_SPECS
            mandatory

        CR_NAMES
            description: Personnel names in the same
                         Hobby activities
            value class: CREW_STATUS
            inverse: CR_HOBBIES
            multivalued

    identifiers:
        H_NUM
```

Figure A-1. SDM design (cont 'd).

```
EDUCATION

    description: Education database of each crew-
                 member on the ship.

    member attributes:

        E_NUM
            description: Education number
            value class: NUMBER
            mandatory
            not changeable

        DEGREE
            description: The degree of the education
            value class: NAMES

        E_TYPE
            description: Description about each edu-
                         cation
            value class: NAMES

        SCHOOL
            description: Graduated school of each
                         education
            value class: DESCR

        CR_NAMES
            description: Crewmembers that have been
                         educated in a particular
                         education type
            value class: CREW_STATUS
            inverce: EDUCATIONS
            multivalued

    identifiers:
        E_NUM
```

Figure A-1. SDM design (cont 'd).

```
PREVIOUS_OCCUPATION

    description: Occupation before the crewmember
                 joined the Navy.

    member attributes:

        PR_NUM
            description: Number that specifies the
                         occupation
            value class: NUMBER
            mandatory
            not changeable

        PR_TYPE
            description: Description of the previous
                         occupation
            value class: NAMES

        PLACE
            description: The place of the previous
                         occupation
            value class: PLACES

        CR_NAMES
            description: Crewmembers that belong to a
                         particular occupation type
            value class: CREW_STATUS
            inverse: PREVIOUS_OCCUPATIONS
            multivalued

    identifiers:
        PR_NUM
```

Figure A-1. SDM design (cont 'd).

```
LEAVE

    description: Leave types and period schedule
                 for each crewmember.

    member attributes:

        LE_NAME
            description: Type of leave
            value class: LE_NAMES
            mandatory

        L_DATA
            description: Data about each type of
                         leave

            subattributes:

            ST_LDATE
                description: Start scheduled date
                value class: DATES
                mandatory

            E_LDATE
                description: End scheduled date
                value class: DATES

            LPERIOD
                description: Number of days on leave
                value class: NUMBER
                mandatory

            LDESTINATION
                description: Place to spend the leave
                value class: PLACES

            LDESCRIPTION
                description: Comments about the leave
                value class: DESCR
```

Figure A-1. SDM design (cont 'd).

```
            CR_NAMES
                description: Personnel names who take
                             leave
                value class: CREW_PROF
                inverse: CR_LEAVE

        SUB_NUMBER
            description: Person name who substi-
                         tutes for the leaving person
            value class: CREW_PROF
            inverse: DUT_NUM
            The leaving person concedes his du-
            ties number to the assigning person
            for leave period.


    identifiers:
        LE_NAME + ST_LDATE



PUNISHMENTS

    description: Punishment types and periods.

    member attributes:

        P_NAME
            description: Punishment type
            value class: P_NAMES
            mandatory

        P_DATA
            description: Details about each punish-
                         ment

            subattributes:

                ST_PDATE
                description: Start date of the puni-
                             shment
                value class: DATES
                mandatory
```

Figure A-1. SDM design (cont 'd).

```
            E_PDATE
                description: End date of the punishment
                value class: DATES

            PPERIOD
                description: Number of punished days
                value class: NUMBER
                mandatory

        CR_NAMES
            description: Punished crewmember name
            value class: CREW_PROF
            inverse: CR_PUNISHMENT

    identifiers:
        P_NAME + ST_PDATE



TRAINING_COURSE_INFO

    description: Information about the training
                 courses.

    member attributes:

        T_NUMBER
            description: The number of the training
                         course
            value class: T_NUMBERS
            mandatory
            not changeable

        T_NAME
            description: The name of the training
                         course
            value class: NAMES
            mandatory
            not changeable
```

Figure A-1. SDM design (cont 'd).

```
        T_DESCRIPTION
            description: Details about the subject
                         of the course
            value class: DESCR
            not changeable

        T_PLACE
            description: Place where the training
                         course will take place
            value class: PLACES

       T_CREW
            description: Ranks and specialties of
                         the participating crew-
                         member

            Subattributes:

            T_RANK
                description: Ranks of the training
                             crewmembers
                value class: RANKS

            T_SPECIALTY
                description: Specialties of the
                             training crewmembers
                value class: SPECS

        multivalued

    identifiers:
        T_NUMBER
```

Figure A-1. SDM design (cont 'd).

```
TRAINING_COURSES

    description: Schedule of the training courses.

    member attributes:

        T_NUMBERS
            description: Schedule of each
                         training course
            value class: TRAINING_COURSE_INFO
            inverse: T_NUMBER
            mandatory

        T_COURSE
            description: Details about each training
                         course

            subattribute:

                ST_TDATE
                    value class: DATES

                E_TDATE
                    value class: DATES

                MAX_NO_OF_PERSONS
                    value class: NUMBER

                HOURS_PER_DAY
                    value class: NUMBER

                ST_HOUR
                    value class: TIMES

            mandatory

        CR_NAMES
            description: Crewmembers who attend the
                         course
            value class: CREW_PROF
            inverse: CR_TRAINING
            multivalued

    identifiers:
        T_NUMBER + ST_TDATE
```

Figure A-1. SDM design (cont 'd).

```
DUTIES

    description: Crewmember duty and positions under
                 different situations on the ship.

    member attributes:

        DUT_NUMBER
            description: Code number of a set of
                         duties
            value class: D_NUMBERS
            not changeable

        CREW_NUM
            description: Crewmember's name who has
                         been assigned with a set
                         of duties
            value class: CREW_PROF
            inverse: DUT_NUM
            multivalued

        VIS_NUM
            description: Visitor's name who has
                         been assigned a particular
                         set of duties
            value class: VISITORS
            inverse: V_NUMBER

        D_RANK
            description: Duty rank
            value class: RANKS
            not changeable

        D_SPECIALTY
            description: Duty specialty
            value class: SPECS
            not changeable
```

Figure A-1. SDM design (cont 'd).

```
TWO_SHIFTS
    description: Two shifts duties

    subattributes:

        TWO_SECTION
            value class: T_SECTION

        TWO_POS
            value class: POSITION

        TWO_DUTIES
            value class: DUTIES

    not changeable

THREE_SHIFTS
    description: Three shifts duties

    subattributes:

        TH_SECTION
            value class: TH_SECTION

        TH_POS
            value class: POSITION

        TH_DUTIES
            value class: DUTIES

    not changeable

D_DOCK
    description: Duty on dock
    value class: DUTIES
    not changeable

ABANDON
    description: Life-boat number in ABANDON
                 SHIP situation
    value class: NUMBER
    not changeable
```

Figure A-1. SDM design (cont 'd).

```
        ALERT
            description: Duty and position in alert

            subattributes:

                AL_POS
                    value class: POSITION

                AL_DUTIES
                    value class: DUTIES

            not changeable

    identifiers:
        DUT_NUMBER



VISITORS

    description: Basic responsibilities for non
                 crewmember (visitors).

    member attributes:

        V_NUMBER
            description: Visitor's number
            value class: DUTIES
            inverse: VIS_NUM
            mandatory
            not changeable

        V_TITLE
            description: Visitor's title
            value class: V_TITLES
            mandatory

        V_RANK
            description: Visitor's rank
            value class: RANKS
            mandatory
```

Figure A-1. SDM design (cont 'd).

```
        V_NAME
            description: Visitor's FIRST and
                         LAST name

            subattributes:

                V_FNAME
                    value class: F_NAMES

                V_LNAME
                    value class: L_NAMES
            mandatory

        V_DATE
            description: Visitor's date on
                         the ship
            value class: DATES
            mandatory

        V_PERIOD
            description: Total days on the ship
            value class: NUMBER
            mandatory

    identifiers:
        V_NUMBER + V_DATE



DAILY_ACTIVITIES

    description: Database activities which are esta-
                 blished by the Executive Officer and
                 must be carried out by the officer on
                 duty.

    member attributes:

        ACT_NUM
            description: Code number for each activity
            value class: NUMBER
            mandatory
```

Figure A-1. SDM design (cont 'd).

```
        ACTIVITIES
            description: Activity description
            value class: ACTS
            mandatory

        CR_NAMES
            description: Officers on duties
            value class: CREW_PROF
            inverse: ON_DUTY
            multivalued

    identifiers:
        ACT_NUM



EXERCISES

    description: Scheduled exercises for the
                 current year.

    member attributes:

        E_NAME
            description: Exercise's name
            value class: E_NAMES
            mandatory

        ST_EDATE
            description: Start date of the
                         exercise
            value class: DATES
            mandatory

        E_EDATE
            description: End date
            value class: DATES
```

Figure A-1. SDM design (cont 'd).

```
        EPERIOD
            description: Exercise's period
            value class: NUMBER
            mandatory

        AREA
            description: Exercise's area
            value class: PLACES

    identifiers:
        E_NAME + ST_EDATE
```

Figure A-1. SDM design (cont 'd).

CR_NUMBERS

interclass connection: Subclass of STRINGS where length is 5 characters.

RANKS

interclass connection: Subclass of STRINGS where length is less than 10 characters where specified.

SPECS

interclass connection: Subclass of STRINGS where length is less than 13 characters where specified.

F_NAMES

interclass connection: Subclass of STRINGS where length is less than 12 characters where specified.

L_NAMES

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

NUMBER

interclass connection: Subclass of STRINGS where format is positive integer less than 99.

DATES

interclass connection: Subclass of STRINGS where format is number as MMDDYY

Figure A-2. SDM domain definition.

ROLES

interclass connection: Subclass of STRINGS where length is less than 12 characters where specified.

CR_MAR

interclass connection: Subclass of STRINGS where value is:
'MARRIED'
'UNMARRIED'
'DIVORCED'

DIGIT

interclass connection: Subclass of STRINGS where length is a positive number less than 9.

ADDR

interclass connection: Subclass of STRINGS where length is less than 20 characters where specified.

PLACES

interclass connection: Subclass of STRINGS where length is less than 12 characters where specified.

PHONES

interclass connection: Subclass of STRINGS where length is less than 11 characters where specified.

Figure A-2. SDM domain definition (cont 'd).

T_TYPES

interclass connection: Subclass of STRINGS where value is:
'ORIGINAL'
'RELATIVE'
'NEAR BASE'
'POLICE'

H_TYPES

interclass connection: Subclass of STRINGS where value is:
'SOCCER'
'BASKETBALL'
'TENNIS'
'VOLLEYBALL'
'SKI'
'PING-PONG'
'MUSIC'
'SWIM'
'OTHER'

DESCR

interclass connection: Subclass of STRINGS where length is less than 10 characters where specified.

H_SPECS

interclass connection: Subclass of STRINGS where value is:
'LISTENING'
'PLAYING'
'WATCHING'
'DOING'
'OTHER'

NAMES

interclass connection: Subclass of STRINGS where length is less than 10 characters where specified.

Figure A-2. SDM domain definition (cont 'd).

LE_NAMES

interclass connection: Subclass of STRINGS where value is:
'STANDARD'
'EXTRA'
'ON_SEA'
'SPECIAL'
'DUTY EXCEPTION'
'HOSPITAL LEAVE'

P_NAMES

interclass connection: Subclass of STRINGS where value is:
'PRISON ASHORE'
'PAYBACK CONFINEMENT'
'CONSECUTIVE DAY CONFINEMENT'
'ALTERNATE DAY CONFINEMENT'

T_NUMBERS

interclass connection: Subclass of STRINGS where format is positive number less than 9999.

TIMES

interclass connection: Subclass of STRINGS where format is HH.MM, HH is positive integer between 0 and 23, and MM positive integer between 0 and 59.

D_NUMBERS

interclass connection: Subclass of STRINGS where length is 5 characters.

POSITION

interclass connection: Subclass of STRINGS where length is less than 10 characters where specified.

Figure A-2. SDM domain definition (cont 'd).

T_SECTION

interclass connection: Subclass of STRINGS where value is:
'L' for Left section
'R' for Right section

DUTIES

interclass connection: Subclass of STRINGS where length is less than 12 characters where specified.

TH_SECTION

interclass connection: Subclass of STRINGS where value is:
'A'
'B'
'C'

V_TITLES

interclass connection: Subclass of STRINGS where length is less than 12 characters where specified.

ACTS

interclass connection: Subclass of STRINGS where length is less than 20 characters where specified.

E_NAMES

interclass connection: Subclass of STRINGS where length is less than 10 characters where specified.

Figure A-2. SDM domain definition (cont 'd).

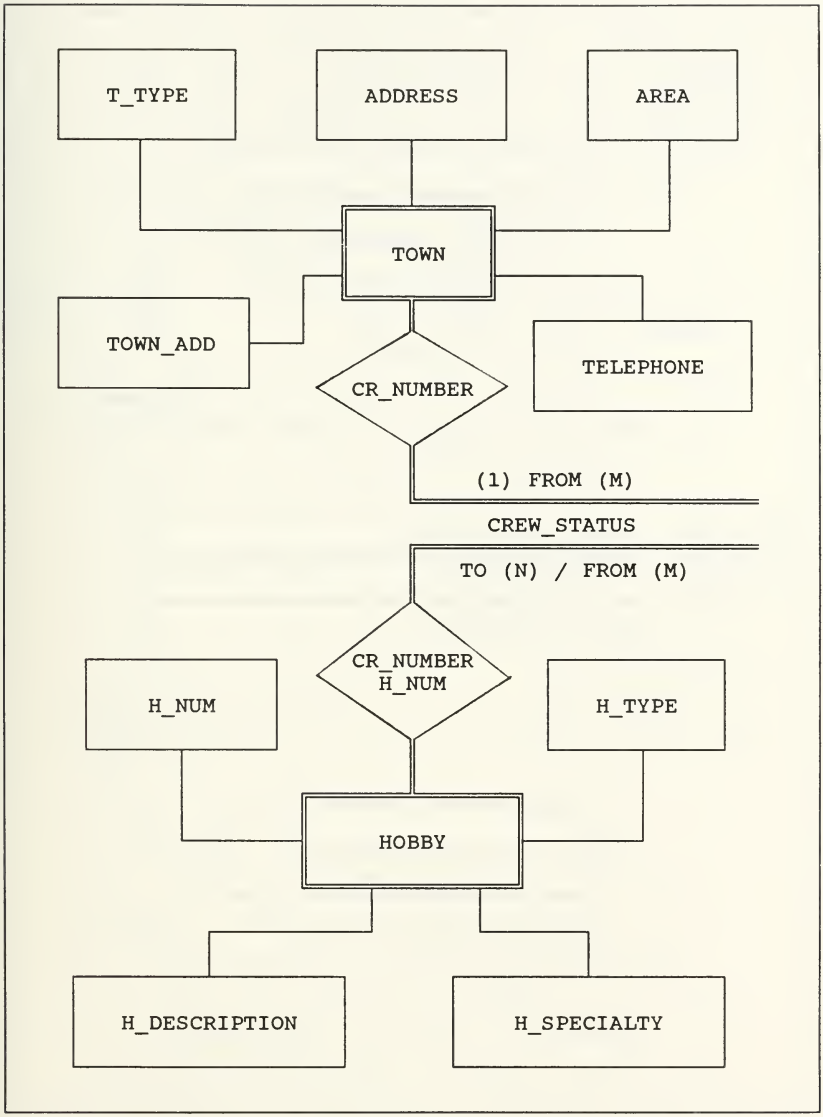


Figure A-3. E-R diagram.

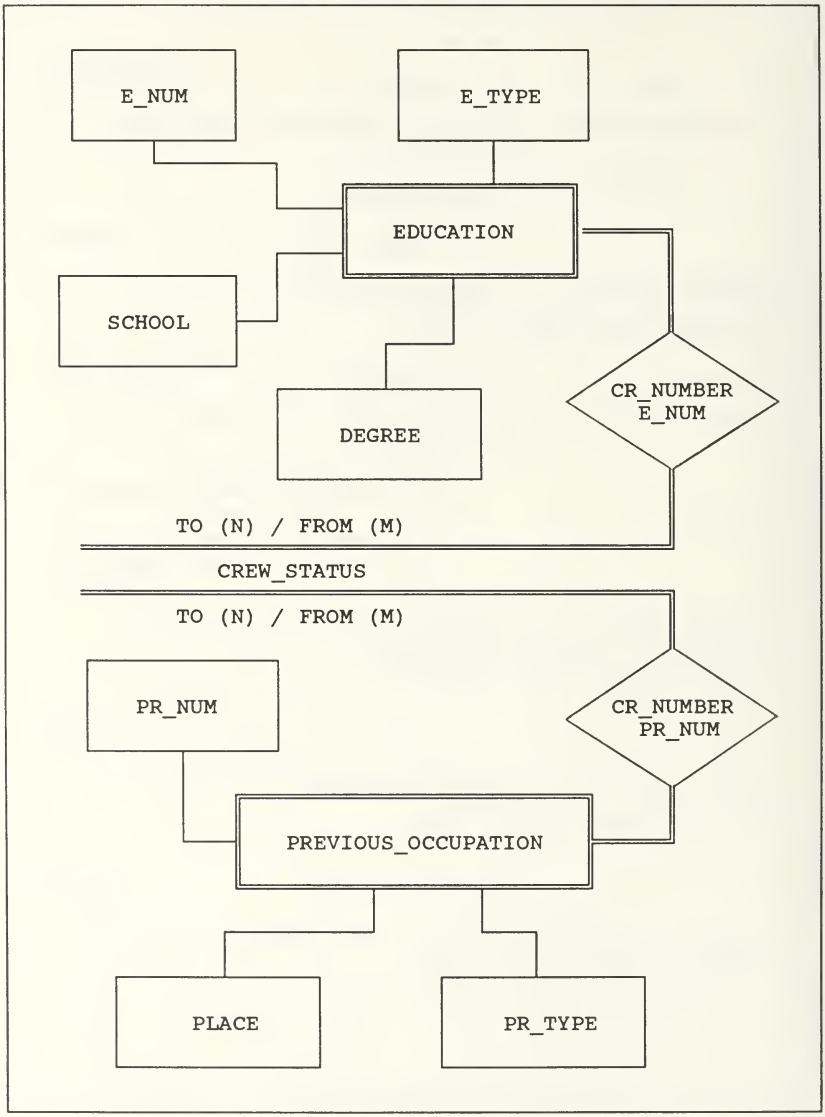


Figure A-3. E-R design (cont 'd).

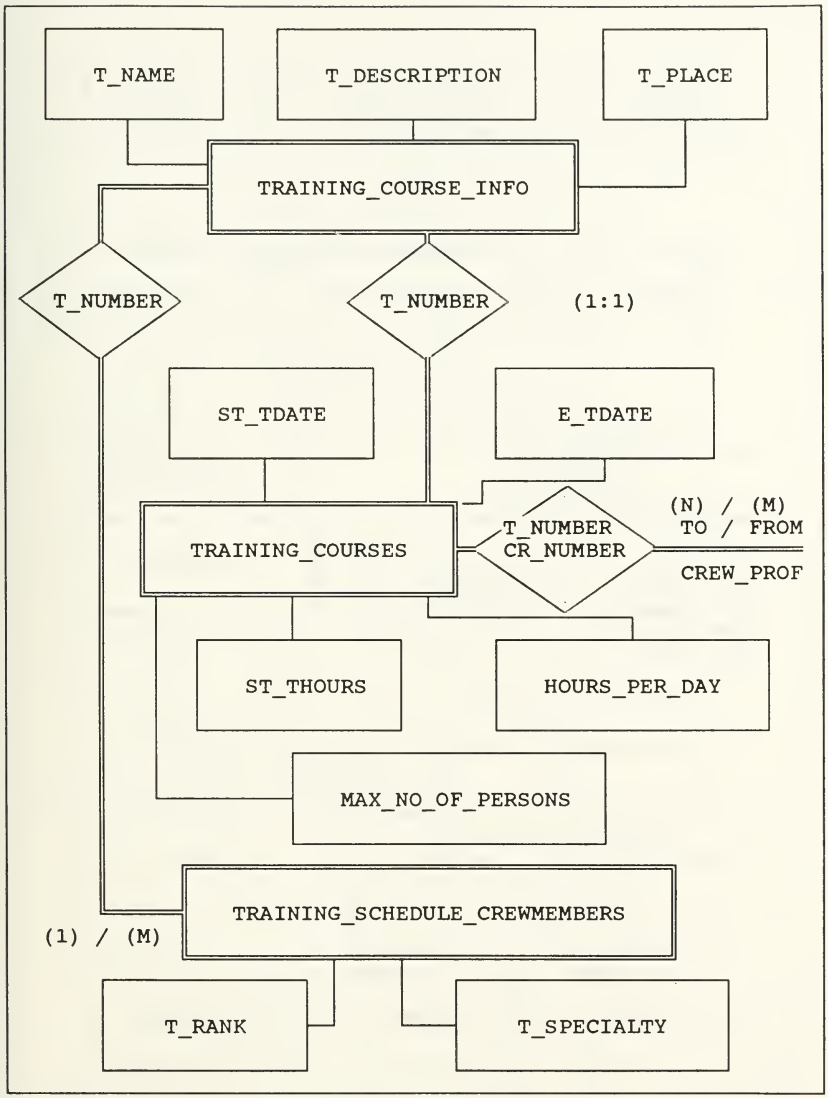


Figure A-3. E-R design (cont 'd).

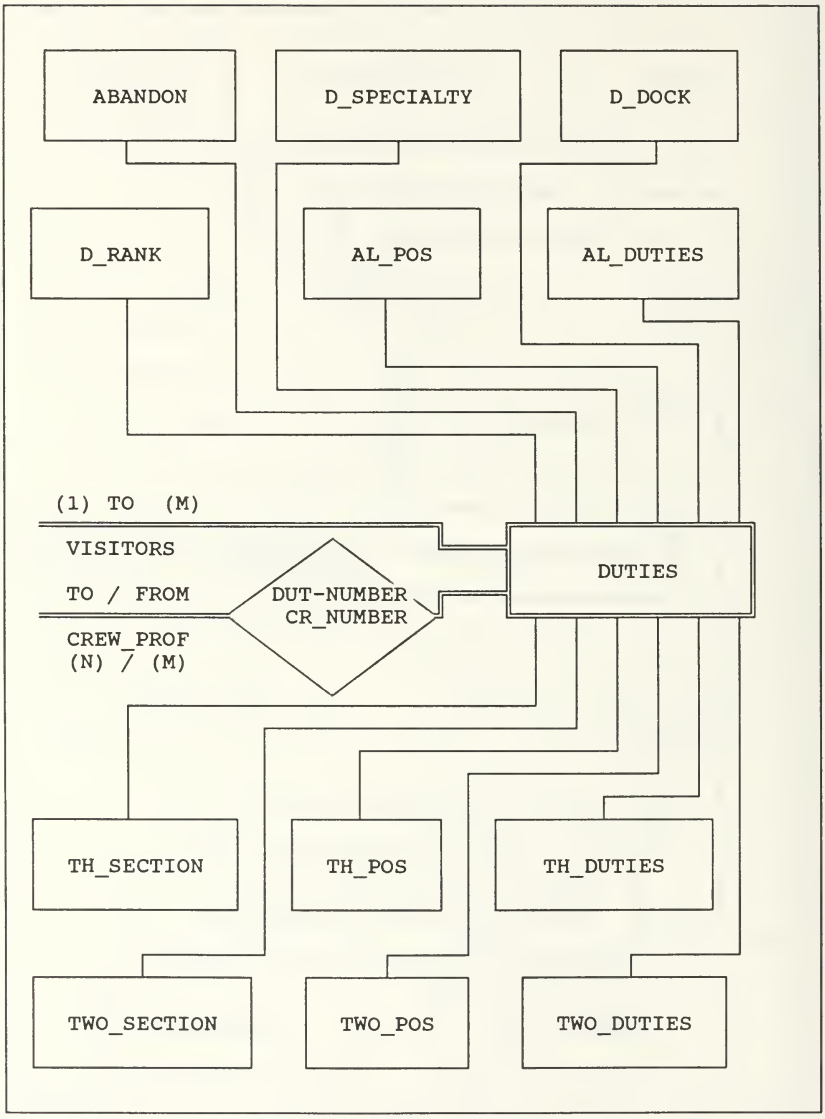


Figure 30. E-R design (cont 'd).

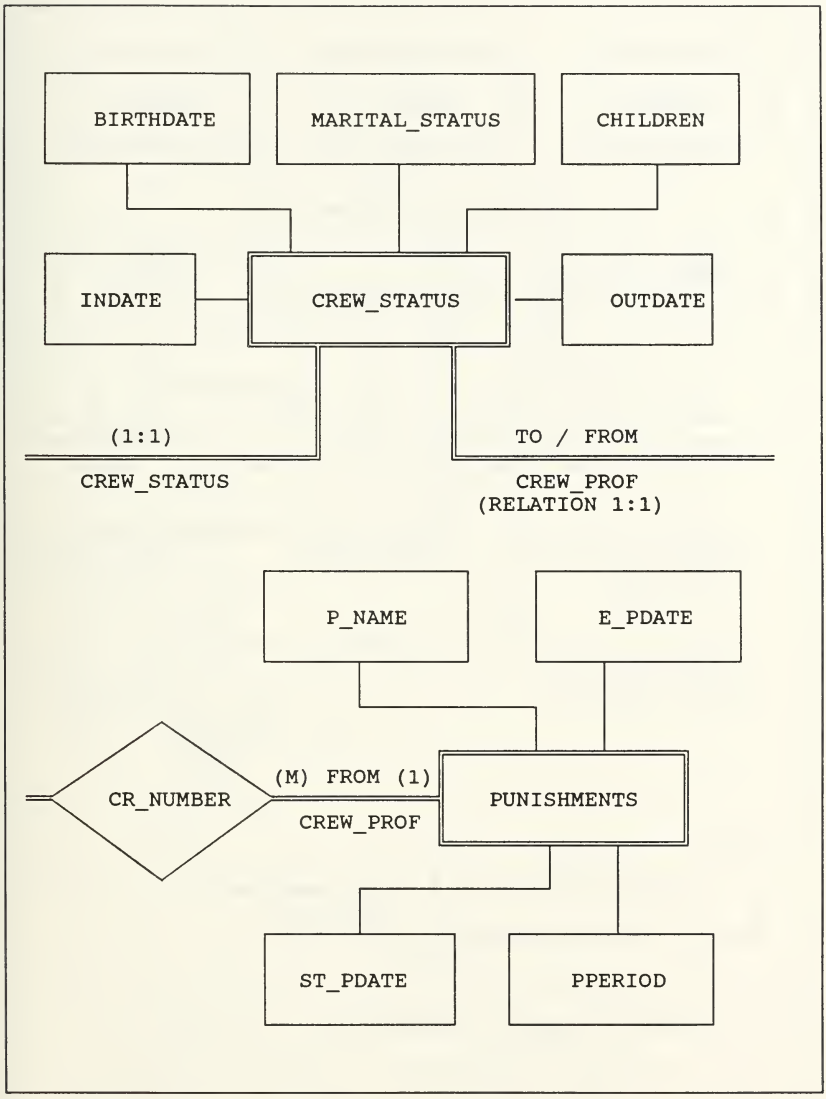


Figure A-3. E-R design (cont 'd).

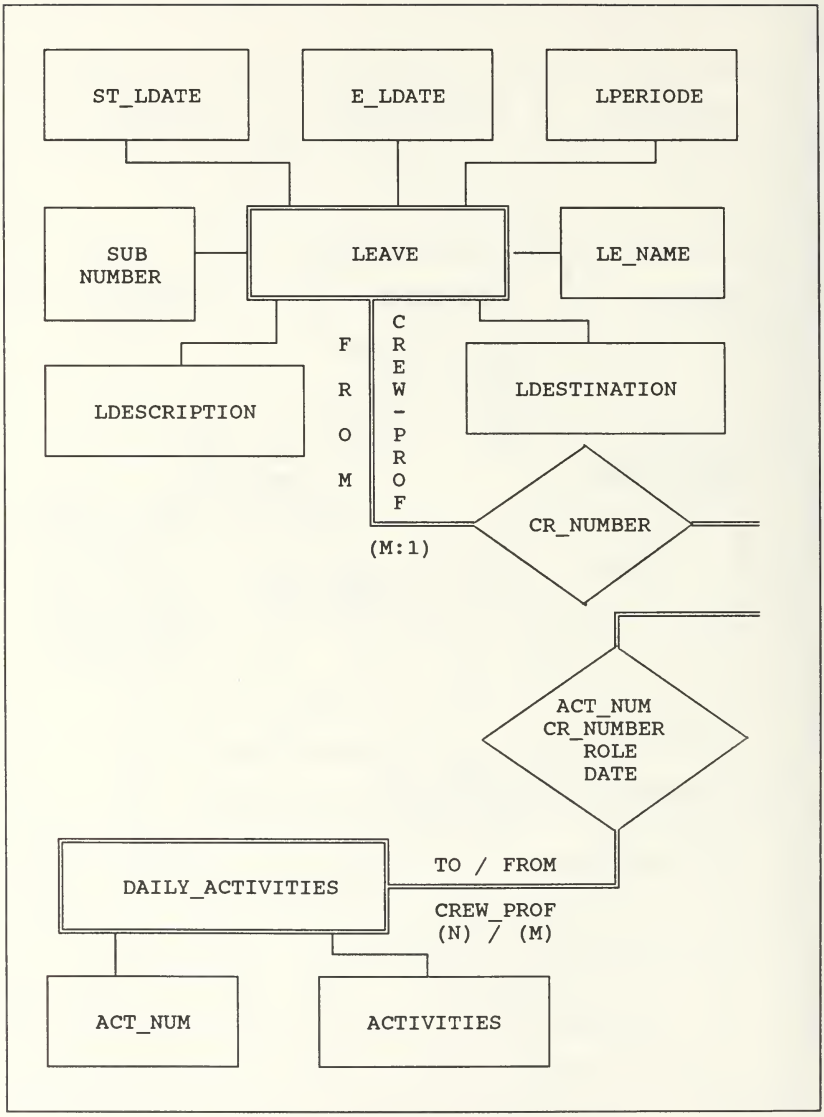


Figure A-3. E-R design (cont 'd).

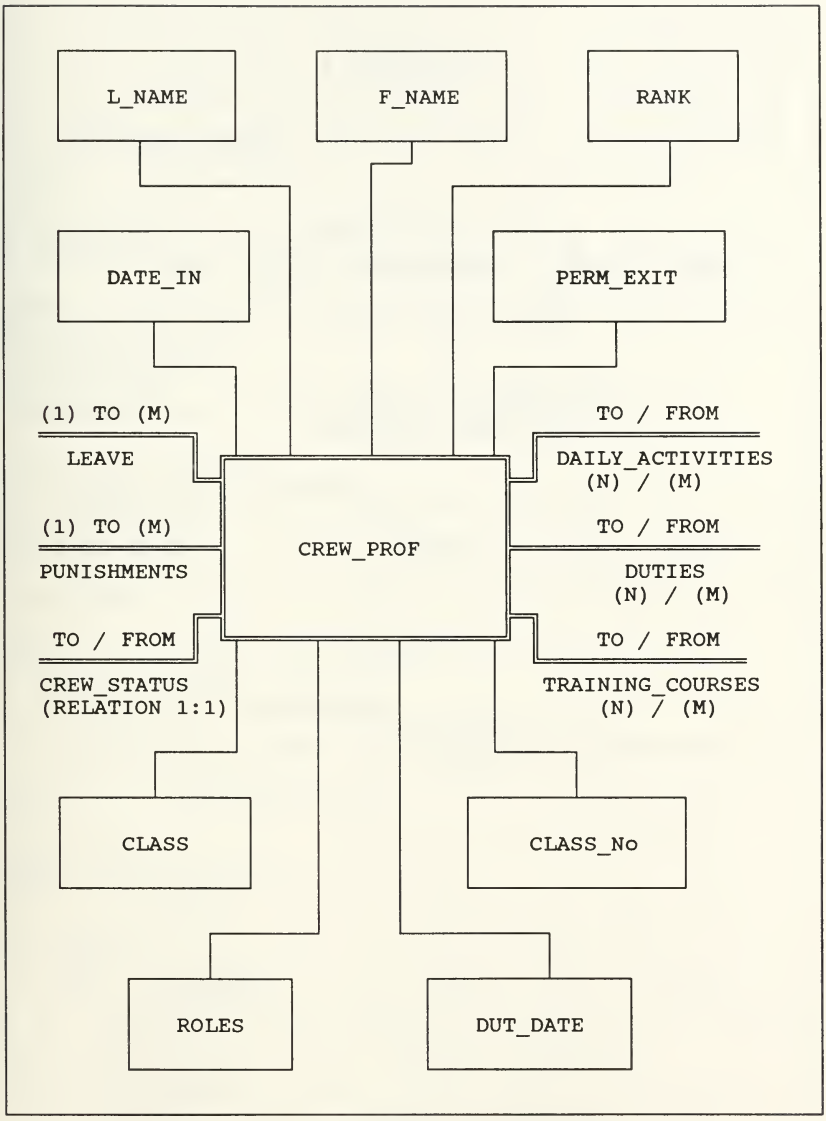


Figure A-3. E-R design (cont 'd).

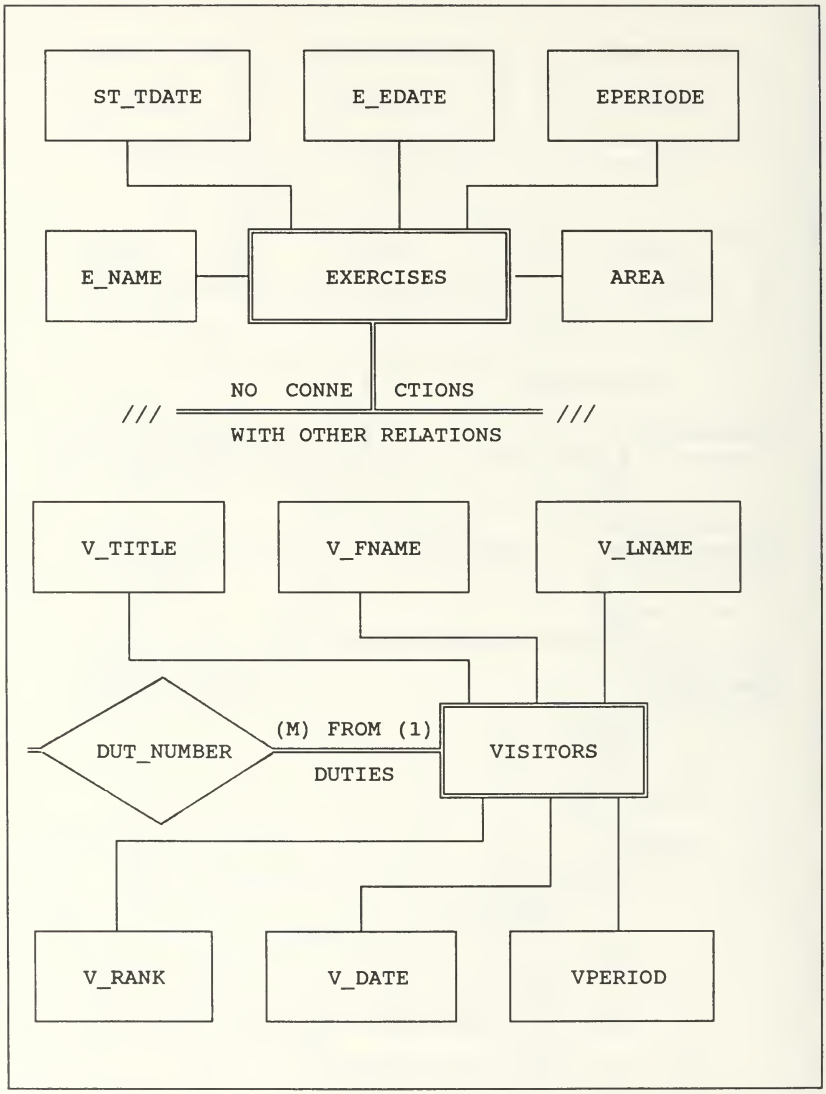


Figure A-3. E-R design (cont 'd).

APPENDIX B

A. MAIN-MENU PROGRAM

```
*  MAIN.PRG
*  MAIN PROGRAM FOR THE ADMINISTRATION DATABASE SYSTEM.

CLEAR
SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET BELL OFF
SET CONFIRM ON

STORE CHR(0) TO psw
@ 11,26 SAY 'ENTER PASSWORD ==> '
SET CONSOLE OFF
ACCEPT TO psw
SET CONSOLE ON

USE USERS

LOCATE FOR password = UPPER(psw)
IF EOF()
   SET COLOR TO W*
   @ 11,26 SAY '   UNAUTHORIZED USER   '
   DO delay
   SET COLOR TO W
   QUIT
ENDIF

CLOSE ALL
PUBLIC subdep,choice
STORE 0 TO subdep

DO WHILE .T.
   CLEAR
   STORE 9 TO choice
   @  1,15 SAY '* * * * * * * * * * * * * * * * * * * * * * *'
   @  2,15 SAY '*                                           *'
   @  3,15 SAY '*             M A I N   M E N U             *'
   @  4,15 SAY '*             -----------------             *'
   @  5,15 SAY '*                                           *'
   @  6,15 SAY '*         0. EXIT TO OPERATING SYSTEM       *'
   @  7,15 SAY '*         1. COMMANDING OFFICE              *'
   @  8,15 SAY '*         2. EXECUTIVE  OFFICE              *'
```

```
   @  9,15 SAY '*          3. OPERATION  DEPARTMENT          *'
   @ 10,15 SAY '*          4. ENGINEER   DEPARTMENT          *'
   @ 11,15 SAY '*          5. ELECTRONIC DEPARTMENT          *'
   @ 12,15 SAY '*          6. SUPPLY     DEPARTMENT          *'
   @ 13,15 SAY '*          7. CHANGE STATUS - DUTIES         *'
   @ 14,15 SAY '*          8. CREWMEMBER INFO                *'
   @ 15,15 SAY '*          9. EXIT TO dBASE III PLUS         *'
   @ 16,15 SAY '*                                            *'
   @ 17,15 SAY '* * * * * * * * * * * * * * * * * * * * * * *'


   SET COLOR TO W+
   @ 19,24 SAY 'ENTER YOUR CHOICE ==> ';
   GET choice PICTURE '9' RANGE 0,9
   READ
   SET COLOR TO W
   DO CASE
      CASE choice = 0
           CLEAR
           CLOSE ALL
           QUIT
      CASE choice = 1
           SET PROCEDURE TO dut_list
           DO options
      CASE choice = 2
           SET PROCEDURE TO dut_list
           DO options
      CASE choice = 3
           DO operat
      CASE choice = 4
           DO engine
      CASE choice = 5
           DO electr
      CASE choice = 6
           DO supply
      CASE choice = 7
           DO chstdut
      CASE choice = 8
           DO cr_list
      CASE choice = 9
           CLEAR
           CLOSE ALL
           RETURN
    ENDCASE
ENDDO
SET TALK ON
SET DELIMITER ON
SET HEADING ON
SET CONFIRM OFF

RETURN
```

B. DEPARTMENT AND SUBDEPARTMENT LISTS

```
*    PROGRAM TO LIST THE DUTIES ACCORDING TO SELECTED
*    SUBDEPARTMENT IN OPERATION DEPARTMENT.

CLEAR
SET PROCEDURE TO dut_list
SET CONFIRM ON

DO WHILE .T.

   STORE 0 TO subdep

   @  3,14 SAY '* * * * * * * * * * * * * * * * * * * * * * *'
   @  4,14 SAY '*                                           *'
   @  5,14 SAY '*            O P E R A T I O N              *'
   @  6,14 SAY '*            -----------------              *'
   @  7,14 SAY '*       S U B D E P A R T M E N T S         *'
   @  8,14 SAY '*       ---------------------------         *'
   @  9,14 SAY '*                                           *'
   @ 10,14 SAY '*     0. ALL SUBDEPARTMENTS                 *'
   @ 11,14 SAY '*     1. NAVIGATION     SUBDEPARTMENT       *'
   @ 12,14 SAY '*     2. COMMUNICATION SUBDEPARTMENT        *'
   @ 13,14 SAY '*     3. WEAPON         SUBDEPARTMENT       *'
   @ 14,14 SAY '*     4. ASW            SUBDEPARTMENT       *'
   @ 15,14 SAY '*     5. EXIT TO MAIN MENU                  *'
   @ 16,14 SAY '*     6. EXIT TO OPERATING SYSTEM           *'
   @ 17,14 SAY '*                                           *'
   @ 18,14 SAY '* * * * * * * * * * * * * * * * * * * * * * *'

   SET COLOR TO W+

   @ 21,22 SAY 'ENTER YOUR SELECTION ==> ';
   GET subdep PICTURE '9' RANGE 0,6
   READ
   SET COLOR TO W
   DO delay

   DO CASE

      CASE subdep = 5
           CLOSE ALL
           CLEAR
           RETURN
      CASE subdep = 6
           CLOSE ALL
           QUIT
      OTHERWISE
           DO options
           IF mexit
```

```
            CLOSE ALL
            CLEAR
            RETURN
         ENDIF

   ENDCASE

ENDDO

SET PROCEDURE TO
SET CONFIRM OFF

RETURN


* PROGRAM TO LIST THE DUTIES ACCORDING TO SELECTED
* SUBDEPARTMENT IN THE ENGINEER DEPARTMENT.

CLEAR
SET PROCEDURE TO dut_list
SET CONFIRM ON

DO WHILE .T.

   STORE 0 TO subdep

   @  3,14 SAY '* * * * * * * * * * * * * * * * * * * * *'
   @  4,14 SAY '*                                       *'
   @  5,14 SAY '*             E N G I N E E R           *'
   @  6,14 SAY '*             --------------            *'
   @  7,14 SAY '*       S U B D E P A R T M E N T S     *'
   @  8,14 SAY '*       --------------------------      *'
   @  9,14 SAY '*                                       *'
   @ 10,14 SAY '*       0. ALL SUBDEPARTMENTS           *'
   @ 11,14 SAY '*       1. ENGINEER      SUBDEPARTMENT  *'
   @ 12,14 SAY '*       2. ELECTRIC INST SUBDEPARTMENT  *'
   @ 13,14 SAY '*       3. DAM. CONTROL  SUBDEPARTMENT  *'
   @ 14,14 SAY '*       4. EXIT TO MAIN MENU            *'
   @ 15,14 SAY '*       5. EXIT TO OPERATING SYSTEM     *'
   @ 16,14 SAY '*                                       *'
   @ 17,14 SAY '* * * * * * * * * * * * * * * * * * * * *'

   SET COLOR TO W+

   @ 20,22 SAY 'ENTER YOUR SELECTION ==> ';
   GET subdep PICTURE '9' RANGE 0,5
   READ
   SET COLOR TO W
   DO delay
```

```
   DO CASE

      CASE subdep = 4
           CLOSE ALL
           CLEAR
           RETURN
      CASE subdep = 5
           CLOSE ALL
           QUIT
      OTHERWISE
           DO options

           IF mexit
              CLOSE ALL
              CLEAR
              RETURN
           ENDIF

   ENDCASE

ENDDO

SET PROCEDURE TO
SET CONFIRM OFF

RETURN



* PROGRAM TO LIST THE DUTIES ACCORDING TO SELECTED
* SUBDEPARTMENT IN THE ELECTRONIC DEPARTMENT.

CLEAR
SET PROCEDURE TO dut_list
SET CONFIRM ON

DO WHILE .T.

   STORE 0 TO subdep
   @  3,14 SAY '* * * * * * * * * * * * * * * * * * * * * *'
   @  4,14 SAY '*                                         *'
   @  5,14 SAY '*            E L E C T R O N I C          *'
   @  6,14 SAY '*            ------------------           *'
   @  7,14 SAY '*       S U B D E P A R T M E N T S       *'
   @  8,14 SAY '*       --------------------------        *'
   @  9,14 SAY '*                                         *'
   @ 10,14 SAY '*       0. ALL SUBDEPARTMENTS             *'
   @ 11,14 SAY '*       1. COMPUTER      SUBDEPARTMENT    *'
   @ 12,14 SAY '*       2. RADAR         SUBDEPARTMENT    *'
   @ 13,14 SAY '*       3. ESM / ECM     SUBDEPARTMENT    *'
   @ 14,14 SAY '*       4. EXIT TO MAIN MENU              *'
   @ 15,14 SAY '*       5. EXIT TO OPERATING SYSTEM       *'
```

```
   @ 16,14 SAY '*                                                *'
   @ 17,14 SAY '* * * * * * * * * * * * * * * * * * * * * * * * *'

   SET COLOR TO W+

   @ 20,22 SAY 'ENTER YOUR SELECTION ==> ';
   GET subdep PICTURE '9' RANGE 0,5
   READ
   SET COLOR TO W
   DO delay

   DO CASE

      CASE subdep = 4
           CLOSE ALL
           CLEAR
           RETURN
      CASE subdep = 5
           CLOSE ALL
           QUIT
      OTHERWISE
           DO options

           IF mexit
              CLOSE ALL
              CLEAR
              RETURN
           ENDIF

   ENDCASE

ENDDO

SET PROCEDURE TO
SET CONFIRM OFF

RETURN


* PROGRAM TO LIST THE DUTIES ACCORDING TO SELECTED
* SUBDEPARTMENT IN THE SUPPLY DEPARTMENT.

CLEAR
SET PROCEDURE TO dut_list
SET CONFIRM ON

DO WHILE .T.

   STORE 0 TO subdep
```

```
   @  3,14 SAY '* * * * * * * * * * * * * * * * * * * * * *'
   @  4,14 SAY '*                                          *'
   @  5,14 SAY '*                S U P P L Y               *'
   @  6,14 SAY '*                -----------               *'
   @  7,14 SAY '*        S U B D E P A R T M E N T S       *'
   @  8,14 SAY '*        --------------------------        *'
   @  9,14 SAY '*                                          *'
   @ 10,14 SAY '*       0. ALL SUBDEPARTMENTS              *'
   @ 11,14 SAY '*       1. GEN. SUPPLY   SUBDEPARTMENT     *'
   @ 12,14 SAY '*      2. MEDICAL       SUBDEPARTMENT     *'
   @ 13,14 SAY '*       3. SPARE PARTS   SUBDEPARTMENT     *'
   @ 14,14 SAY '*       4. EXIT TO MAIN MENU               *'
   @ 15,14 SAY '*       5. EXIT TO OPERATING SYSTEM        *'
   @ 16,14 SAY '*                                          *'
   @ 17,14 SAY '* * * * * * * * * * * * * * * * * * * * * *'

   SET COLOR TO W+

   @ 20,22 SAY 'ENTER YOUR SELECTION ==> ';
   GET subdep PICTURE '9' RANGE 0,5
   READ
   SET COLOR TO W
   DO delay

   DO CASE

      CASE subdep = 4
           CLOSE ALL
           CLEAR
           RETURN
      CASE subdep = 5
           CLOSE ALL
           QUIT
      OTHERWISE
           DO options

           IF mexit
              CLOSE ALL
              CLEAR
              RETURN
           ENDIF

   ENDCASE

ENDDO

SET PROCEDURE TO
SET CONFIRM OFF

RETURN
```

```
* Program....: DUT_LIST.PRG
* Notes......: PROCEDURE FOR DUTIES RETRIEVAL INFORMATION

* PROCEDURE     PURPOSE
* ---------     -------
*
* MESSAGE1      Display a note message about the MENU
selections.
* MESSAGE2      Display a warning message in case of wrong
*              selections.
* MENUOPTS      Display the MENU options for duties.
* REMBLANK      Removes the blanks e.t.c. from the selection.

* OPTIONS       Determines the selected fields to be printed.



PROCEDURE message1

CLEAR
TEXT
    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    *                      NOTES                              *
    *                      -----                              *
    *     This program helps you to retrieve information      *
    * by choosing one or more numbers from the below MENU.    *
    *     You may use comma (,), minus sign (-), or space     *
    * ( )(Blank) to seperate your entering numbers.           *
    *     Attention: Any other combination is not allowed.    *
    *                                          ---            *
ENDTEXT

RETURN


PROCEDURE message2

CLEAR
TEXT
    * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
    *                                                          *
    *                ? ? ?  WARNING  ? ? ?                     *
    *                       -------                            *
    *       * * * You entered invalid number sting * * *       *
    *          Only numbers from MENU list is allowed.         *
    *                                                          *
ENDTEXT

RETURN
```

```
PROCEDURE menuopts

TEXT
    *                                                       *
    *                  DUTIES SELECTION MENU                *
    *                  ---------------------                *
    *    0. EXIT                                            *
    *    1. ALERT         : POSITION AND DUTIES             *
    *    2. TWO SHIFTS    : SECTION, POSITION, AND DUTIES   *
    *    3. THREE SHIFTS  : SECTION, POSITION, AND DUTIES   *
    *    4. ABANDON       : LIFE BOAT NUMBER                *
    *    5. DOCK          : DUTIES ON DOCK                  *
    *    6. ALL           : ALL THE ABOVE                   *
    *                                                       *
    * * * * * * * * * * * * * * * * * * * * * * * * * * * *

 ENDTEXT

RETURN


PROCEDURE remblank

PARAMETERS lcount,lcode,lists

STORE .T. TO lbool

DO WHILE lbool

    STORE SUBSTR(lists,lcount,1) TO lcode
    STORE lcount + 1 TO lcount

    IF lcode = " " .OR. lcode = "," .OR. lcode = "-"
        STORE .T. TO lbool
    ELSE
        STORE .F. TO lbool
    ENDIF

ENDDO

RETURN


PROCEDURE options

PUBLIC mexit,seldep

SET EXACT OFF
DO message1
STORE .T. TO lrepeat
```

```
STORE .F. TO mexit

IF subdep <> 0
   STORE LTRIM(LTRIM(STR(choice)) + LTRIM(STR(subdep))) TO
seldep
ELSE
   STORE LTRIM(STR(choice)) TO seldep
ENDIF

DO WHILE lrepeat

    DO menuopts
    STORE CHR(0) TO lists
    ACCEPT '    ENTER YOUR CHOISE ==>  ' TO lists
    CLOSE DATABASES
    USE DUT_CREW INDEX DUT_CREW
    SET HEADING ON
    GO TOP
    STORE 1 TO lcount
    STORE CHR(0) TO lcode
    STORE .F. TO lrepeat

    DO WHILE lcount <= LEN(lists)

        DO remblank WITH lcount,lcode,lists
        CLEAR
        DO CASE
           CASE lcode = '0'
                STORE .T. TO mexit
                CLEAR
                RETURN
            CASE lcode = '1'
                 REPORT FORM TALARM FOR DUT_NUMBER = seldep;


                        TO DUTAL
            CASE lcode = '2'
                 REPORT FORM TWSHIFT FOR DUT_NUMBER =

                 seldep TO DUTTWO
            CASE lcode = '3'
                 REPORT FORM THRSH FOR DUT_NUMBER = seldep;
                        TO DUTTH
            CASE lcode = '4'
                 REPORT FORM TABAN FOR DUT_NUMBER = seldep;
                        TO DUTAB
            CASE lcode = '5'
                 REPORT FORM TDOCK FOR DUT_NUMBER = seldep;
                        TO DUTDOCK
            CASE lcode = '6'
                 DISPLAY CR_NUMBER,RANK,SPECIALTY,L_NAME;
                          F_NAME,DUT_NUMBER,MAIN;
```

```
                        AL_DUT,AL_POS,TW_SEC;
                        TW_DUT,TW_POS,TH_SEC;
                        TH_DUT,TH_POS,ABANDON;
                  D_DOCK FOR DUT_NUMBER = seldep

            OTHERWISE
                STORE .T. TO lrepeat
                STORE LEN(lists) + 1 TO lcount
                DO message2

        ENDCASE
        GO TOP
        WAIT
    ENDDO
    WAIT
    CLEAR
ENDDO

RETURN

* EOF : DUT_LIST.PRG
```

C. BASIC FUNCTIONS PROGRAMS

```
*  CHSTDUT.PRG
*  PROGRAM SHOWS THE MENU FOR INSERT, DELETE, UPDATE,
*  OR CHANGE DUTIES OF THE CREWMEMBERS ON THE SHIP.

SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET BELL OFF
SET CONFIRM ON
SET PROCEDURE TO changecr

SELECT 1
USE DUT_CREW INDEX DUT_CREW ALIAS DUT
SELECT 2
USE DUTIES INDEX DUTIES ALIAS DUTIES
SELECT 3
USE CR_DUT INDEX CR_DUT ALIAS CRDUT
SELECT 4
USE CREW_PR INDEX CREW_NUM, CREW_PR, CREW_RC ALIAS CREW
STORE .F. TO newjoin

DO WHILE .T.
   CLEAR
   STORE 0 TO chdata
   @  1,15 SAY '* * * * * * * * * * * * * * * * * * * * * *'
   @  2,15 SAY '*                                         *'
   @  3,15 SAY '*           CHANGE STATUS - DUTIES        *'
   @  4,15 SAY '*           ---------------------         *'
   @  5,15 SAY '*                                         *'
   @  6,15 SAY '*       0.  EXIT TO MAIN MENU             *'
   @  7,15 SAY '*       1.  INSERT NEW CREWMEMBER         *'
   @  8,15 SAY '*       2.  DELETE CREWMEMBER             *'
   @  9,15 SAY '*       3.  UPDATE CREWMEMBER DATA        *'
   @ 10,15 SAY '*       4.  CHANGE CREWMEMBER DUTIES      *'
   @ 11,15 SAY '*       5.  EXIT TO OPERATING SYSTEM      *'
   @ 12,15 SAY '*                                         *'
   @ 13,15 SAY '* * * * * * * * * * * * * * * * * * * * * *'


   SET COLOR TO W+
   @ 18,22 SAY 'ENTER YOUR CHOICE ==> ';
   GET chdata PICTURE '9' RANGE 0,5
   READ
   SET COLOR TO W

   DO CASE
```

```
      CASE chdata = 0
           CLEAR
           IF newjoin
              DO joindata
           ENDIF
           RELEASE ALL
           CLOSE ALL
           RETURN
      CASE chdata = 1
           DO newcrew
           STORE .T. TO newjoin
      CASE chdata = 2
           DO delcrew
           STORE .T. TO newjoin
      CASE chdata = 3
           DO upcrew
      CASE chdata = 4
           DO chduties
      CASE chdata = 5
           CLEAR
           IF newjoin
              DO joindata
           ENDIF
           RELEASE ALL
           CLOSE ALL
           QUIT

   ENDCASE

ENDDO

RETURN



*  NEWCREW.PRG
*  PROGRAM FOR INSERTING NEW CREWMEMBER INTO
*  THE DATABASE.

CLEAR
SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL ON
SET PROCEDURE TO chnew

PUBLIC rnk, spl, ans, crnum

STORE '          ' TO rnk
STORE '             ' TO spl
```

```
STORE .Y. TO ans
STORE '00000' TO crnum

SELECT 4

@  3,15 SAY 'INSERT NEW CREWMEMBER DATA'
@  4,15 SAY '--------------------------'
@  7,10 SAY 'MILITARY ID     : ';
GET crnum PICTURE '99999'
READ
GO TOP
LOCATE FOR CR_NUMBER = crnum
IF FOUND()
   DO WHILE .NOT. EOF()
      @ 17,5 SAY 'This MIL. ID exists ... Please, try again'


     @ 7,10 SAY 'MILITARY ID     : ';
      GET crnum PICTURE '99999'
      READ
      GO TOP
      LOCATE FOR CR_NUMBER = crnum
   ENDDO
ENDIF
@ 17,5 CLEAR TO 17,50
APPEND BLANK
REPLACE CR_NUMBER WITH crnum
@  8,10 SAY 'FIRST NAME      : ';
GET F_NAME PICTURE '@!A'
READ
@  9,10 SAY 'LAST NAME       : ';
GET L_NAME PICTURE '@!A'
READ
@ 10,10 SAY 'RANK            : ';
GET rnk PICTURE '@!'
READ
DO chrank
@ 11,10 SAY 'SPECIALTY       : ';
GET spl PICTURE '@!'
READ
DO chspl
GO TOP
LOCATE FOR CR_NUMBER = crnum
@ 12,10 SAY 'CLASS           : ';
GET CLASS PICTURE '99'
READ
@ 13,10 SAY 'CLASS No        : ';
GET CLASS_POS PICTURE '99'
READ
@ 14,10 SAY 'PERMITTION EXIT : ';
GET PERM_EXIT PICTURE '9' RANGE 0,5
READ
```

```
@ 15,10 SAY 'START DATE IN    : ';
GET DATE_IN RANGE CTOD("08/08/88"), CTOD("01/01/99")
READ

@ 18,14 SAY 'CHECK YUOR ENTERED DATA'
@ 19,18 SAY ' CONFIRM (Y/N)?';
GET ans PICTURE 'Y'
READ
IF ans
   DO again
ENDIF

SET PROCEDURE TO
CLEAR
SET PROCEDURE TO changecr

@ 5,16 SAY 'Is he going to substitute a crewmember ? (Y/N)';

GET ans PICTURE 'Y'
READ
IF ans
   DO rigcrew
   DO subst
ELSE
   DO newdut
ENDIF

SET PROCEDURE TO

RETURN



*  DELCREW.PRG
*  PROGRAM TO DELETE A LEAVING BY ORDER CREWMEMBER.

SET DELETED ON
SET TALK OFF
SET DELIMITER OFF
SET CONFIRM ON
SET BELL OFF
SET HEADING OFF

CLEAR

PUBLIC id, crnum, dutnum

SET PROCEDURE TO changecr
STORE .Y. TO ans
SELECT 4

DO rigcrew
```

```
DO WHILE EOF()
   SELECT 3
   GO TOP
   LOCATE FOR CR_NUMBER = id
   IF FOUND()
      STORE DUT_NUMBER TO dutnum
      GO TOP
      LOCATE FOR CR_NUMBER # id .AND. DUT_NUMBER = dutnum
      IF FOUND()
         DELETE ALL FOR CR_NUMBER = id .AND. DUT_NUMBER =
dutnum
         SELECT 1
         DELETE ALL FOR CR_NUMBER = id .AND. DUT_NUMBER =
dutnum
      ENDIF
   ENDIF
ENDDO

SELECT 3
GO TOP
LOCATE FOR CR_NUMBER = id
IF EOF()
   SELECT 4
   GO TOP
   LOCATE FOR CR_NUMBER = id
   @ 12,10 SAY 'SAY GOOD LUCK TO Mr. '+ L_NAME + F_NAME
   DO DELAY
   DELETE
   SET PROCEDURE TO
   RELEASE ALL
   RETURN
ELSE
   CLEAR
   @ 4,10 SAY 'HE HAS THE FOLLOWING DUTIES :'
   GO TOP
   DISPLAY ALL FOR CR_NUMBER = id
   @ 18,6 SAY 'Do you want to assign them to '
   @ 19,6 SAY 'a particular crewmember ? (Y/N) ';
   GET ans PICTURE 'Y'
   READ
   IF ans
      CLEAR
      SELECT 4
      STORE id TO crnum
      DO rigcrew
      DO subst
      SELECT 3
      DELETE ALL FOR CR_NUMBER = crnum
      SELECT 1
      DELETE ALL FOR CR_NUMBER = crnum
      SELECT 4
      CLEAR
```

```
      LOCATE FOR CR_NUMBER = crnum
      @ 4,8 SAY 'SAY GOOD LUCK TO Mr. '+ L_NAME + F_NAME
      DO delay
      DELETE
      RELEASE ALL
      SET PROCEDURE TO
      CLEAR
      RETURN
   ELSE
      SELECT 3
      DO WHILE .NOT. EOF()
         DISPLAY ALL FOR CR_NUMBER = id
         @ 18,10 SAY 'ENTER ONE DUTY-NUMBER ==> ';
         GET dutnum PICTURE '99999'
         READ
         DO delay
         SET PROCEDURE TO deletecr
         DO asscrew
         SELECT 3
      ENDDO
   ENDIF
ENDIF

RETURN



*  UPCREW.PRG
*  PROGRAM FOR UPDATING CREWMEMBER DATA INTO
*  THE DATABASE

CLEAR
SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL ON
SET PROCEDURE TO changecr

PUBLIC rnk,crnum

SELECT 4

STORE '          ' TO rnk
DO rigcrew
@  1,15 SAY 'UPDATE CREWMEMBER DATA'
@  2,15 SAY '----------------------'
@ 10,10 SAY 'RANK            : ';
GET rnk PICTURE '@!'
READ

SET PROCEDURE TO chnew
```

```
STORE id TO crnum
DO chrank
SELECT 4
LOCATE FOR CR_NUMBER = id
STORE 0 TO perm
STORE CTOD("08/08/88") TO datein
@ 14,10 SAY 'PERMITTION EXIT : ';
GET perm PICTURE '9' RANGE 0,5
READ
@ 15,10 SAY 'START DATE IN   : ';
GET datein RANGE CTOD("08/08/88"), CTOD("01/01/99")
READ
STORE perm TO PERM_EXIT
STORE DTOC(datein) TO DATE_IN
SELECT 1
GO TOP
LOCATE FOR CR_NUMBER = id
STORE perm TO PERM_EXIT
STORE DTOC(datein) TO DATE_IN

SET PROCEDURE TO
CLEAR

RETURN



*  CHDUTIES.PRG
*  PROGRAM THAT PERMITS REASSIGNMENT OF DUTIES
*  BETWEEN THE CREW ON THE SHIP.

SET DELETED ON
SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL OFF

PUBLIC id, rnk, spl

CLEAR
DO WHILE .T.
   DO rigcrew
   STORE 3 TO choice
   SELECT 4
   GO TOP
   LOCATE FOR CR_NUMBER = id
   STORE RANK TO rnk
   STORE SPECIALTY TO spl

   @ 6,20 SAY 'DO YOU WANT TO : '
   @ 7,20 SAY '1. Delete duties ? '
```

```
   @ 8,20 SAY '2. Assign duties ? '
   @ 9,20 SAY '3. All done ? '
   @ 11,17 SAY 'ENTER YOUR CHOICE ==> ';
   GET choice PICTURE '9' RANGE 1,3
   READ

   DO CASE
      CASE choice = 1
           SET PROCEDURE TO deletecr
           DO checkdut

      CASE choice = 2
           LOCATE FOR CR_NUMBER = id
           STORE RANK TO rnk
           STORE SPECIALTY TO spl
           SET PROCEDURE TO changecr
           DO newdut

      CASE choice = 3
           CLEAR
           RELEASE ALL
           SET PROCEDURE TO
           RETURN

   ENDCASE
ENDDO

RETURN



*  Program....: CHANGECR.PRG
*  Notes......: PROCEDURES TO ASSIGN DUTIES

*  PROCEDURE    PORPUSE
*  ---------    -------
*
*  RIGCREW      Provide assistance for the ented crewmember
*  SUBST        Assign duties
*  NEWDUT       Provide assistance for making decision in
*               assigning duties
*  DUTINS       Assign a specified duties
*  JOINDATA      Join the three basic databases when changes

*               occur


PROCEDURE  rigcrew

SET CONFIRM ON
SET BELL OFF
SET DELIMITER OFF
```

```
SET TALK OFF

PUBLIC id

CLEAR
STORE '00000' TO id
STORE .N. TO ans

DO WHILE .NOT. ans
   @ 4,15 SAY ' ENTER HIS MIL. ID : ';
   GET id PICTURE '99999'
   READ
   GO TOP
   LOCATE FOR CR_NUMBER = id
   IF FOUND()
      @ 6,10 SAY RANK +' '+SPECIALTY +' '+F_NAME +' '+L_NAME

      @ 8,10 SAY 'Is the rigth crewmember ? (Y/N) ';
      GET ans PICTURE 'Y'
      READ
      @ 6,10 CLEAR TO 6,60
      @ 8,10 CLEAR TO 8,60
   ELSE
      @ 10,10 SAY 'Wrong ID ; Please try again ...'
      DO delay
      @ 10,10 CLEAR TO 10,50
   ENDIF
ENDDO
@ 6,10 CLEAR TO 6,60
@ 8,10 CLEAR TO 8,60

RETURN



PROCEDURE subst

SELECT 3
GO TOP
LOCATE FOR CR_NUMBER = id
DO WHILE EOF()
   STORE DUT_NUMBER TO dutnum
   STORE MAIN TO exdut
   APPEND BLANK
   REPLACE CR_NUMBER WITH crnum, DUT_NUMBER WITH dutnum ;
           MAIN WITH exdut
   SELECT 3
   GO TOP
   LOCATE FOR CR_NUMBER = id .AND. DUT_NUMBER # dutnum
ENDDO

RETURN
```

```
PROCEDURE newdut

SET CONFIRM ON
STORE .Y. TO ans
CLEAR
@ 2,8 SAY 'Do you want assistance ? (Y/N) ';
GET ans PICTURE 'Y'
READ
SELECT DUTIES
IF ans
   @ 5,8 SAY 'Please enter one or more duty-numbers. '
   DO delay
   LIST ALL FOR D_RANK = rnk .AND. D_SPECIAL = spl
   STORE .Y. TO more
   @ 10,8 SAY 'Do you want more duties ? (Y/N) ';
   GET more PICTURE 'Y'
   READ
   IF more
      USE RANK_SPE INDEX RANK_COD
      GO TOP
      LOCATE FOR RANK = rnk
      STORE CODE TO rcod
      STORE VAL(rcod) TO rcod
      STORE INT(rcod)+1 TO rplus
      STORE rplus - 2 TO rminus
      STORE STR(rminus) TO rminus
      STORE STR(rplus) TO rplus
      GO TOP
      LOCATE FOR CODE = LTRIM(rminus)
      IF .NOT. EOF()
         STORE RANK TO rnkminus
         USE DUTIES INDEX DUTIES
         LIST ALL FOR D_RANK = rnkminus .AND. D_SPECIAL = spl

      ENDIF
      USE RANK_SPE INDEX RANK_COD
      GO TOP
      LOCATE FOR CODE = LTRIM(rplus)
      IF .NOT. EOF()
         STORE RANK TO rnkplus
         USE DUTIES INDEX DUTIES
         LIST ALL FOR D_RANK = rnkplus .AND. D_SPECIAL = spl

      ENDIF
   ENDIF
ENDIF

STORE .F. TO checkdig
DO WHILE  .NOT. checkdig
   STORE '                          ' TO dutnum
   @ 18,8 SAY 'Enter his duty-numbers ==> ';
   GET dutnum
```

```
   READ
   DO DELAY
   CLEAR
   STORE .T. TO maindut
   STORE .T. TO checkdig
   STORE 1 TO st, num
   DO WHILE st <= LEN(dutnum) .OR. checkdig
      STORE SUBSTR(dutnum,st,num) TO dutch
      IF dutch # ' ' .OR. dutch # '-' .OR. dutch # ','
         STORE 5 TO num
         STORE SUBSTR(dutnum,st,num) TO dutch
         STORE 1 TO lcode
         DO WHILE lcode <= LEN(dutch)
            STORE SUBSTR(dutch,lcode,1) TO dutcheck
            IF dutcheck < '0' .AND. dutcheck > '9'
               STORE .F. TO checkdig
               @ 5,10 SAY 'YOU ENTERED THE STRING : '+ dutnum

               @ 7,10 SAY ' Wrong entry; Please try again '
            ENDIF
            STORE lcode + 1 TO lcode
          ENDDO
          IF checkdig
             DO dutins
          ENDIF
          STORE st + num TO st
          STORE 1 TO num
       ELSE
          STORE st + 1 TO st
       ENDIF
   ENDDO
ENDDO

CLEAR
RETURN

PROCEDURE dutins


SELECT 3
APPEND BLANK
IF maindut
   STORE 'M' TO dutchar
   STORE .F. TO maindut
ELSE
   STORE 'S' TO dutchar
ENDIF
REPLACE CR_NUMBER WITH crnum, DUT_NUMBER WITH dutch;
        MAIN WITH dutchar

RETURN
```

```
PROCEDURE joindata

SET TALK OFF
SET BELL OFF
SET HEADING OFF

ERASE DUT_CREW.DBF
ERASE DUT_CREW.NDX
SELECT 3
JOIN WITH CREW TO TEMP
USE TEMP
INDEX ON DUT_NUMBER TO TEMP
JOIN WITH DUTIES TO DUT_CREW
USE DUT_CREW
INDEX ON RANK,SPECIALTY,CLASS,CLASS_POS TO DUT_CREW
ERASE TEMP.DBF
ERASE TEMP.NDX
SELECT 4

RETURN



*  Program....: DELETECR.PRG
*  Notes......: PROCEDURES FOR DELETE AND CHANGE
*               DUTIES PROGRAMS

*  PROCEDURE    PURPOSE
*  ---------    -------
*
*  ASSCREW      Provide assistance for making decision
*  CHECKDUT     Delete the specified duty-number after
*               assigning it to another crewmember



PROCEDURE asscrew

SET TALK OFF
SET CONFIRM ON
SET BELL OFF
SET DELIMITER OFF
SET HEADING OFF
SET PROCEDURE TO changecr

CLEAR
@ 2,8 SAY 'Do you want assistance ? (Y/N) ';
GET ans PICTURE 'Y'
READ
SELECT 4
GO TOP
LOCATE CR_NUMBER = id
```

```
STORE RANK TO rnk
STORE SPECIALTY TO spl
GO TOP
IF ans
   DISPLAY CR_NUMBER,RANK,SPECIALTY,L_NAME,F_NAME,DUT_NUMBER;

           FOR RANK = rnk .AND. SPECIALTY = spl
   STORE .Y. TO more
   @ 10,8 SAY 'Do you want more crewmembers? (Y/N) ';
   GET more PICTURE 'Y'
   READ
   IF more
      USE RANK_SPE INDEX RANK_COD
      GO TOP
      LOCATE FOR RANK = rnk
      STORE CODE TO rcod
      STORE VAL(rcod) TO rcod
      STORE INT(rcod) TO rcod
      STORE rplus - 2 TO rminus
      STORE STR(rminus) TO rminus
      STORE STR(rplus) TO rplus
      GO TOP
      LOCATE FOR CODE = LTRIM(rminus)
      IF .NOT. EOF()
         STORE RANK TO rnkminus
         SELECT 4
         LIST RANK+' '+SPECIALTY+' '+L_NAME+' '+F_NAME;
         FOR RANK = rnkminus .AND. SPECIALTY = spl
      ENDIF
      USE RANK_SPE INDEX RANK_COD
      GO TOP
      LOCATE FOR CODE = LTRIM(rplus)
      IF .NOT. EOF()
         STORE RANK TO rnkplus
         SELECT 4
         LIST RANK+' '+SPECIALTY+' '+L_NAME+' '+F_NAME;
         FOR RANK = rnkplus .AND. SPECIALTY = spl
      ENDIF
   ENDIF
ENDIF
GO TOP
STORE '00000' TO crnum
DO WHILE .NOT. FOUND()
   @ 18,10 SAY 'ENTER MIL. ID ==> ';
   GET crnum PICTURE '99999'
   READ
   IF EOF()
      @ 18,8 SAY 'WRONG ENTRY, PLEASE TRY AGAIN...'
      DO delay
   ENDIF
   @18,6 CLEAR TO 18,30
ENDDO
```

```
STORE id TO chnum
STORE crnum TO id
STORE chnum TO crnum

DO dutins

DELETE ALL FOR CR_NUMBER = crnum .AND. DUT_NUMBER = dutnum
STORE id TO chnum
STORE crnum TO id
STORE chnum TO crnum

RETURN



PROCEDURE checkdut

SET DELETED ON
SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL OFF
SET PROCEDURE TO changecr

PUBLC dutnum, newjoin

SELECT 3
GO TOP
STORE '00000' TO dutnum
LIST DUT_NUMBER, MAIN FOR CR_NUMBER = id
DO WHILE .NOT. FOUND()
   @ 18,10 SAY 'ENTER DUTY-NUMBER ==> ';
   GET dutnum PICTURE '99999'
   READ
   IF EOF()
      @ 17,6 SAY 'This DUTY-NUMBER does not exist '
      @ 18,10 SAY 'Please try again...'
      DO delay
      @ 17,6 CLEAR TO 17,30
   ENDIF
ENDDO

GO TOP
LOCATE FOR CR_NUMBER # id .AND. DUT_NUMBER = dutnum
IF EOF()
   DO asscrew
   STORE .F. TO newjoin
ELSE
   DELETE ALL FOR CR_NUMBER = id .AND. DUT_NUMBER = dutnum
   SELECT 1
```

```
   DELETE ALL FOR CR_NUMBER = id .AND. DUT_NUMBER = dutnum
ENDIF

RETURN


*  Program....:  CHNEW.PRG
*  Notes......:  PROCEDURES TO VALIDATE CREWMEMBER DATA

*  PROCEDURE     PORPUSE
*  ---------     -------
*
*  CHRANK        Validate the entered crewmember rank
*  CHSPL         Validate the entered crewmember specialty
*  AGAIN         Permit the review of entered data


PROCEDURE chrank

SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL ON

USE RANK_SPE INDEX RANK_SPE
STORE .N. TO ans

DO WHILE .T.
   GO TOP
   LOCATE FOR RANK = rnk
   IF FOUND()
      STORE CODE TO rncode
      IF ans
         STORE 1 TO X
         DO WHILE X < 18
            @ X+1,52 SAY '                '
            STORE X + 1 TO X
         ENDDO
      ENDIF
      USE CREW_PR INDEX CREW_NUM, CREW_PR, CREW_RC
      REPLACE RANK WITH rnk, RCODE WITH rncode;
              FOR CR_NUMBER = crnum
      RETURN
   ELSE
      SET COLOR TO W*
      @ 15,5 SAY 'WRONG RANK ENTRY; PLEASE TRY AGAIN'
      SET COLOR TO W
      IF .NOT. ans
         @ 19,7 SAY 'DO YOU NEED SOME HELP ? (Y/N)';
```

```
         GET ans PICTURE 'Y'
         READ
         @ 19,7 SAY '                                    '
         IF ans
            STORE 1 TO X
            DO WHILE X < 18
               GOTO RECORD X
               @ X+1,52 SAY RANK
               STORE X + 1 TO X
            ENDDO
         ENDIF
      ENDIF
   ENDIF
   @ 15,5 SAY '                                          '
   @ 19,7 SAY '                                  '
   @ 10,29 GET rnk PICTURE "@!"
   STORE LTRIM(rnk) TO rnk
   READ

ENDDO

RETURN



PROCEDURE chspl

SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL ON

USE RANK_SPE INDEX RANK_SPE
STORE .N. TO ans

DO WHILE .T.
   GO TOP
   LOCATE FOR SPECIALTY = spl
   IF FOUND()
      STORE CODE TO splcode
      IF ans
         STORE 1 TO X
         DO WHILE X < 18
            @ X+1,52 SAY '                       '
            STORE X + 1 TO X
         ENDDO
      ENDIF
      USE CREW_PR INDEX CREW_NUM, CREW_PR, CREW_RC
      REPLACE SPECIALTY WITH spl, SPCODE WITH splcode;
              FOR CR_NUMBER = crnum
      RETURN
```

```
   ELSE
      SET COLOR TO W*
      @ 17,5 SAY 'WRONG SPECIALTY ENTRY; PLEASE TRY AGAIN'
      SET COLOR TO W
      IF .NOT. ans
         @ 19,7 SAY 'DO YOU NEED SOME HELP ? (Y/N) ';
         GET ans PICTURE 'Y'
         READ
         @ 19,7 SAY '                                    '
         IF ans
            STORE 1 TO X
            GO TOP
            DO WHILE X < 18
               SKIP
               @ X+1,52 SAY SPECIALTY
               STORE X + 1 TO X
            ENDDO
         ENDIF
      ENDIF
  ENDIF
  @ 17,5 SAY '                                        '
  @ 19,5 SAY '                                        '
  @ 11,29 GET spl PICTURE "@!"
  READ

ENDDO

RETURN


PROCEDURE again

SET TALK OFF
SET DELIMITER OFF
SET HEADING OFF
SET CONFIRM ON
SET BELL OFF

@  3,15 SAY 'CORRECT YOUR NEW CREWMEMBER DATA'
@  4,15 SAY '--------------------------------'
@  7,10 SAY 'MILITARY ID     : ' ;
GET CR_NUMBER PICTURE '99999'
READ
@  8,10 SAY 'FIRST NAME      : ' ;
GET F_NAME PICTURE '@!A'
READ
@  9,10 SAY 'LAST NAME       : ' ;
GET L_NAME PICTURE '@!A'
READ
@ 10,10 SAY 'RANK            : ' ;
GET rnk PICTURE '@!'
```

```
READ
DO chrank
@ 11,10 SAY 'SPECIALTY       : '  ;
GET spl PICTURE '@!'
READ
DO chspl
GO TOP
LOCATE FOR CR_NUMBER = crnum
@ 12,10 SAY 'CLASS           : '  ;
GET CLASS PICTURE '99'
READ
@ 13,10 SAY 'CLASS No        : '  ;
GET CLASS_POS PICTURE '99'
READ
@ 14,10 SAY 'PERMITTION EXIT : '  ;
GET PERM_EXIT PICTURE '9'
READ
@ 15,10 SAY 'START DATE IN   : '  ;
GET DATE_IN RANGE CTOD("08/08/88"), CTOD("01/01/95")
READ

RETURN
```

D. CREWMEMBER LISTS

```
*  CR_LIST.PRG
*  PROGRAM THAT LIST CREWMEMBER'S DUTIES AND
*  CREATES RANK-CATEGORIES CREWMEMBER LISTS

CLEAR
USE CREW_PR INDEX CREW_PR
SET TALK OFF
SET CONFIRM ON
SET EXACT ON
SET BELL OFF
SET PROCEDURE TO cr_info

PUBLIC id

DO WHILE .T.

   STORE 0 TO infnum
   STORE '00000' TO id

   @  1,15 SAY '* * * * * * * * * * * * * * * * * * * * *'
   @  2,15 SAY '*                                       *'
   @  3,15 SAY '*           C R E W M E M B E R         *'
   @  4,15 SAY '*           ------------------          *'
   @  5,15 SAY '*                I N F O                *'
   @  6,15 SAY '*                -------                *'
   @  7,15 SAY '*                                       *'
   @  8,15 SAY '*        0. EXIT TO MAIN MENU           *'
   @  9,15 SAY '*        1. LIST OFFICERS STATUS        *'
   @ 10,15 SAY '*        2. LIST N.C.O.   STATUS        *'
   @ 11,15 SAY '*        3. LIST SEAMEN   STATUS        *'
   @ 12,15 SAY '*        4. LIST ALL CREWMEMBERS        *'
   @ 13,15 SAY '*        5. LIST STAY IN SEAMEN         *'
   @ 14,15 SAY '*        6. CREWMEMBER STATUS           *'
   @ 15,15 SAY '*        7. CREWMEMBER DUTIES           *'
   @ 16,15 SAY '*        8. EXIT TO OPERATING SYSTEM    *'
   @ 17,15 SAY '*                                       *'
   @ 18,15 SAY '* * * * * * * * * * * * * * * * * * * * *'


   SET COLOR TO W+

   @ 20,23 SAY 'ENTER YOUR SELECTION ==> ';
   GET infnum PICTURE '9' RANGE 0,7
   READ
   SET COLOR TO W

   CLEAR
   STORE .N. TO ok
```

```
IF  ( infnum > 0 ) .AND. ( infnum < 7 )
   @ 10,16 SAY 'DO YOU WANT TO SEND THE CREWMEMBER INFO '
   @ 12,25 SAY 'TO PRINTER ? (Y / N)' ;
   GET ok
   READ
ENDIF

IF ok
   SET COLOR TO W*
   @ 14,20 SAY 'PLEASE SWITCH ON THE PRINTER'
   SET PRINT ON
   SET COLOR TO W
   WAIT
ENDIF

CLEAR

DO CASE

   CASE infnum = 0
        CLOSE ALL
        CLEAR
        RELEASE ALL
        RETURN
   CASE infnum = 1
        REPORT FORM cr_off FOR RCODE < '11'
        IF .NOT. ok
            WAIT
   CASE infnum = 2
        REPORT FORM cr_nco FOR RCODE > '10'
               .AND. RCODE < '17'
        IF .NOT. ok
            WAIT
   CASE infnum = 3
        REPORT FORM cr_seam FOR RCODE > '16'
        IF .NOT. ok
            WAIT
   CASE infnum = 4
        REPORT FORM crew_pr
        IF .NOT. ok
            WAIT
   CASE infnum = 5
        DO stay_in
        IF .NOT. ok
            WAIT
   CASE infnum = 6
        DO cr_status
        IF .NOT. ok
            WAIT
   CASE infnum = 7
        DO cr_duties      CASE infnum = 8
        CLOSE ALL
```

```
            CLEAR
            QUIT

   ENDCASE

   SET PRINT OFF
   CLEAR

ENDDO

RETURN


*  Program....: CR_INFO.PRG
*  Notes......: PROCEDURES TO LIST CREWMEMBER DUTIES

*  PROCEDURE    PORPUSE
*  ---------    -------
*
*  CR_STATUS     Provide the status of a specified crewmember

*  CR_DUTIES     Provide the duties of a specified crewmember

*  STAY_IN      List the seamen who must stay on the ship


PROCEDURE  cr_status

SET CONFIRM ON
SET TALK OFF
SET BELL OFF
SET PRINTER ON
USE CREW_PR INDEX CREW_RC

DO WHILE .T.

   CLEAR

   GO TOP
   @ 2,20 SAY 'Enter the crewmember ID :';
   GET id PICTURE '99999'
   READ
   LOCATE FOR CR_NUMBER = id

   IF EOF ()
      SET COLOR TO W*
      @ 6,22 SAY ' WRONG ID PLEASE TRY AGAIN'
      DO delay
      STORE '00000' TO id
      SET COLOR TO W
   ELSE
```

```
      @  4,10 SAY '* * * * * * * * * * * * * * * * * * * * *'+;
                  '* * * * * * * * *'
      @  5,10 SAY '*                          ' + ;
                  '                                  *'
      @  6,10 SAY '*              MILITARY ID : '+ CR_NUMBER

+ ;
                  '                        *'
      @  7,10 SAY '*                       ' + ;
                  '                                *'
      @  8,10 SAY '* ----------------------------' + ;
                     '----------------------------*'
      @  9,10 SAY '*                                ' +;
                  '                        *'
      @ 10,10 SAY '*    FIRST NAME : '+ F_NAME + ;
                  'LAST NAME : ' +  L_NAME + ' *'
      @ 11,10 SAY '*                          ' + ;
                  '                             *'
      @ 12,10 SAY '*    RANK : '+ RANK + '        SPECIALTY
: ';
                  + SPECIALTY + '   *'
      @ 13,10 SAY '*                      ' + ;
                  '                               *'
      @ 14,10 SAY '* * * * * * * * * * * * * * *' + ;
                  ' * * * * * * * * * * * * * * * *'
      @ 22,10 SAY ' '

      CLOSE ALL
      RETURN

   ENDIF

ENDDO

RETURN


PROCEDURE cr_duties

SET CONFIRM ON
SET TALK OFF
SET BELL OFF

PUBLIC id, did

USE CREW_PR INDEX CREW_PR
CLEAR

DO WHILE .NOT. FOUND()
   GO TOP
   @ 2,20 SAY 'Enter the crewmember ID : ';
```

```
   GET id PICTURE '99999'
   READ
   LOCATE FOR CR_NUMBER = id

   IF EOF()
      SET COLOR TO W*
      @ 6,22 SAY 'WRONG ID PLEASE TRY AGAIN'
      DO DELAY
      STORE '00000' TO id
      SET COLOR TO W
      @ 6,22 SAY '                          '
   ENDIF

ENDDO

CLOSE ALL
USE DUT_CREW INDEX DUT_CREW
LOCATE FOR CR_NUMBER = id

IF EOF()
   @ 17,15 SAY ' HE DOES NOT HAVE DUTIES '
   @ 18,15 SAY ' ======================= '
   @ 21,15 SAY ' THE PROGRAM IS ABORDED '
   CLOSE ALL
   RETURN
ELSE
   DO WHILE .NOT. EOF()

      IF MAIN = 'M'
         CLEAR
         @ 1,32 SAY 'MAIN DUTIES'
         @ 2,32 SAY '-----------'
         DO maindut
         WAIT
      ELSE
         CLEAR
         @ 1,32 SAY 'EXTRA DUTIES'
         @ 2,32 SAY '------------'
         DO maindut
         WAIT
      ENDIF

      CONTINUE

   ENDDO
ENDIF

CLOSE ALL

RETURN
```

```
PROCEDURE stay_in


CLEAR
USE CREW_PR INDEX CREW_PR
SET TALK OFF
SET CONFIRM ON
SET EXACT ON
SET BELL OFF

REPORT FORM seam_in FOR DATE_IN = DATE()

GO TOP
LOCATE FOR RANK = 'COMMANDER' .AND. SPECIALTY = 'DECK'

IF DATE_IN # DATE()
   DO WHILE .NOT. EOF()
      REPLACE DATE_IN WITH DATE() + PERM_EXIT ;
              FOR DATE_IN = DATE() .AND. PERM_EXIT # 0
   ENDDO
 ENDIF


RETURN
```

APPENDIX C

A. CREWMEMBER STATUS LISTS

Page No. 1
09/09/88

LIST STATUS OF OFFICERS ON THE SHIP

MIL. ID	RANK	SPECIALTY	FIRST NAME	LAST NAME
20135	COMMANDER	DECK	MARK	KAPLAN
21512	LCDR.	DECK	BARRY	FOSTER
12948	LCDR.	ENGINNER	JOHN	RICHTER
24322	LT.	DECK	TIMOTHY	MALLORY
24139	LT.	DECK	ERNEST	KENNEDY
23458	LT.	DECK	JAMES	MARSHALL
24542	LT.	DECK	HOWARD	PATTISON
23154	LT.	DECK	JACK	PORTER
14751	LT.	ENGINEER	GERRY	MADSON
32522	LT.	SUPPLY	GEORGE	RAILTON
25312	LT. J. G.	DECK	ANDREW	MALLERICH
15671	LT. J. G.	ENGINEER	ALEX	PAYTON
15333	LT. J. G.	ENGINEER	CREG	RANKIN
43211	LT. J. G.	MEDICAL	CHRIS	MARSDEN
26212	ENSIGN	DECK	ARMAND	LANSON
26722	ENSIGN	DECK	DAVID	FINCH
26750	ENSIGN	DECK	ROBIN	KEELEX
26242	ENSIGN	DECK	MANDEL	RAMSDELL
16498	ENSIGN	ENGINEER	JOSEPH	PATERSON
16918	ENSIGN	ENGINEER	RICHARD	ROBINSON
16755	ENSIGN	ENGINEER	PATRICK	OLSON
39754	ENSIGN	SUPPLY	TOM	OLIVER

LIST STATUS OF N. C. O. ON THE SHIP

MIL. ID	RANK	SPECIALTY	FIRST NAME	LAST NAME
68122	M. CH. PO	RADAR USER	ANTHONY	RUTHERFORD
68455	M. CH. PO	NAVIGATOR	FRANC	SCAULTON
68514	M. CH. PO	COMMUNICATION	WILLIAM	WATSON
68123	M. CH. PO	RADIO	NICK	TOWBER
69322	S. CH. PO	RADAR USER	BEN	SCOTT
69345	S. CH. PO	RADAR USER	VINCENT	SEVERSON
69758	S. CH. PO	TORPEDO	STEPHEN	OHANIAN
75321	CH. PO	NAVIGATOR	RONALD	THORNGATE
75732	CH. PO	COMMUNICATION	PETER	WHISTON
75988	CH. PO	RADIO	LARRY	TRAWIS
75899	CH. PO	RADIO	BRUCE	WHERRY
79325	PO 1 CLASS	NAVIGATOR	ARTHUR	TRASON
80580	PO 1 CLASS	COMMUNICATION	JIM	HARPER
83215	PO 2 CLASS	COMMUNICATION	BILL	HARMAN
85512	PO 3 CLASS	RADAR USER	BOB	HARKINS

LIST STATUS OF SEAMEN ON THE SHIP

MIL. ID	RANK	SPECIALTY	FIRST NAME	LAST NAME
90402	SEAMAN	SUPPLY	ALLAN	PARKINS
90322	SEAMAN	RADAR USER	THOMAS	SANDERS
90777	SEAMAN	RADAR USER	TIM	RUSSEL
90425	SEAMAN	NAVIGATOR	ANDREW	SELBICKY
90345	SEAMAN	NAVIGATOR	RALPH	VINSON
90324	SEAMAN	NAVIGATOR	ALBERT	WHISLER
90520	SEAMAN	NAVIGATOR	ROY	ZIMMERMAN
90315	SEAMAN	NAVIGATOR	HARRY	HALE
90311	SEAMAN	NAVIGATOR	MARK	TIBBERY
90399	SEAMAN	RADIO	PAUL	WARERMAN
90983	SEAMAN	RADIO	DANIEL	SELONER
90288	SEAMAN	RADIO	CUNT	SIMPSON
90985	SEAMAN	TORPEDO	MICHAEL	PEARSON

LIST STATUS OF CREWMEMBERS ON THE SHIP

MIL. ID	RANK	SPECIALTY	FIRST NAME	LAST NAME
20135	COMMANDER	DECK	MARK	KAPLAN
21512	LCDR.	DECK	BARRY	FOSTER
12948	LCDR.	ENGINNER	JOHN	RICHTER
24322	LT.	DECK	TIMOTHY	MALLORY
24139	LT.	DECK	ERNEST	KENNEDY
23458	LT.	DECK	JAMES	MARSHALL
24542	LT.	DECK	HOWARD	PATTISON
23154	LT.	DECK	JACK	PORTER
14751	LT.	ENGINEER	GERRY	MADSON
32522	LT.	SUPPLY	GEORGE	RAILTON
25312	LT. J. G.	DECK	ANDREW	MALLERICH
15671	LT. J. G.	ENGINEER	ALEX	PAYTON
15333	LT. J. G.	ENGINEER	CREG	RANKIN
43211	LT. J. G.	MEDICAL	CHRIS	MARSDEN
26212	ENSIGN	DECK	ARMAND	LANSON
26722	ENSIGN	DECK	DAVID	FINCH
26750	ENSIGN	DECK	ROBIN	KEELEX
26242	ENSIGN	DECK	MANDEL	RAMSDELL
16498	ENSIGN	ENGINEER	JOSEPH	PATERSON
16918	ENSIGN	ENGINEER	RICHARD	ROBINSON
16755	ENSIGN	ENGINEER	PATRICK	OLSON
39754	ENSIGN	SUPPLY	TOM	OLIVER
68122	M. CH. PO	RADAR USER	ANTHONY	RUTHERFORD
68455	M. CH. PO	NAVIGATOR	FRANC	SCAULTON
68514	M. CH. PO	COMMUNICATION	WILLIAM	WATSON
68123	M. CH. PO	RADIO	NICK	TOWBER
69322	S. CH. PO	RADAR USER	BEN	SCOTT
69345	S. CH. PO	RADAR USER	VINCENT	SEVERSON
69758	S. CH. PO	TORPEDO	STEPHEN	OHANIAN
75321	CH. PO	NAVIGATOR	RONALD	THORNGATE
75732	CH. PO	COMMUNICATION	PETER	WHISTON
75988	CH. PO	RADIO	LARRY	TRAWIS
75899	CH. PO	RADIO	BRUCE	WHERRY
79325	PO 1 CLASS	NAVIGATOR	ARTHUR	TRASON
80580	PO 1 CLASS	COMMUNICATION	JIM	HARPER
83215	PO 2 CLASS	COMMUNICATION	BILL	HARMAN
85512	PO 3 CLASS	RADAR USER	BOB	HARKINS
90402	SEAMAN	SUPPLY	ALLAN	PARKINS
90322	SEAMAN	RADAR USER	THOMAS	SANDERS
90777	SEAMAN	RADAR USER	TIM	RUSSEL
90425	SEAMAN	NAVIGATOR	ANDREW	SELBICKY
90345	SEAMAN	NAVIGATOR	RALPH	VINSON

LIST STATUS OF CREWMEMBERS ON THE SHIP

MIL. ID	RANK	SPECIALTY	FIRST NAME	LAST NAME
90324	SEAMAN	NAVIGATOR	ALBERT	WHISLER
90520	SEAMAN	NAVIGATOR	ROY	ZIMMERMAN
90315	SEAMAN	NAVIGATOR	HARRY	HALE
90311	SEAMAN	NAVIGATOR	MARK	TIBBERY
90399	SEAMAN	RADIO	PAUL	WARERMAN
90983	SEAMAN	RADIO	DANIEL	SELONER
90288	SEAMAN	RADIO	CUNT	SIMPSON
90985	SEAMAN	TORPEDO	MICHAEL	PEARSON

B. DEPARTMENT DUTY LISTS

Page No. 1
09/09/88

ALARM DUTY-POSITION

RANK	LAST NAME	FIRST NAME	POSITION	DUTY
LT.	MALLORY	TIMOTHY	BRIDGE	WATCH ASS M.
CH. PO	TOWBER	NICK	RADIO ROOM	RADIO LEADER
M. CH. PO	TOWBER	NICK	RADIO ROOM	RADIO 3
CH. PO	WHISTON	PETER	BRIDGE	COM. ASS
CH. PO	TRAWIS	LARRY	RADIO ROOM	RADIO 1
CH. PO	WHERRY	BRUCE	RADIO ROOM	RADIO 2
PO 1 CLASS	HARPER	JIM	BRIDGE	HF COMMUN.
PO 2 CLASS	HARMAN	BILL	BRIDGE	UHF COMMUN.

TWO SHIFT DUTY-POSITION

RANK	LAST NAME	FIRST NAME	SEC	POSITION	DUTY
LT.	MALLORY	TIMOTHY	R	BRIDGE	WATCH ASS
M. CH. PO	TOWBER	NICK	L	RADIO	RADIO LEADER
M. CH. PO	TOWBER	NICK	R	RADIO	RADIO ASS
CH. PO	WHISTON	PETER	R	BRIDGE	COM. LEADER
CH. PO	TRAWIS	LARRY	R	RADIO	RADIO LEADER
CH. PO	WHERRY	BRUCE	L	RADIO	RADIO ASS
PO 1 CLASS	HARPER	JIM	L	BRIDGE	COM. ASS
PO 2 CLASS	HARMAN	BILL	R	BRIDGE	COM. ASS

THREE SHIFT DUTY-POSITION

RANK	LAST NAME	FIRST NAME	SEC	POSITION	DUTY
LT.	MALLORY	TIMOTHY	B	BRIDGE	WATCH OFF
LT.	MALLORY	TIMOTHY	C	BRIDGE	WATCH ASS
LT.	KENNEDY	ERNEST	C	BRIDGE	WATCH OFF
LT.	MARSHALL	JAMES	A	CIC	OPER. OFF
LT.	MARSHALL	JAMES	C	CIC	OPER. ASS
LT.	PATTISON	HOWARD	A	BRIDGE	WATCH OFF
LT.	PORTER	JACK	C	CIC	OPER. OFF
LT. J. G.	MALLERICH	ANDREW	B	CIC	OPER. OFF
ENSIGN	KEELEX	ROBIN	B	BRIDGE	WATCH ASS
ENSIGN	FINCH	DAVID	A	BRIDGE	WATCH ASS
ENSIGN	LANSON	ARMAND	A	CIC	AD
ENSIGN	RAMSDELL	MANDEL	C	CIC	AD
M. CH. PO	RUTHERFORD	ANTHONY	A	CIC	WATCH ASS
M. CH. PO	SCAULTON	FRANC	B	DAM.ST1	LEADER
M. CH. PO	TOWBER	NICK	A	RADIO	RADIO LEADER
M. CH. PO	TOWBER	NICK	A	RADIO	RADIO ASS
S. CH. PO	SCOTT	BEN	B	CIC	WATCH ASS
S. CH. PO	SEVERSON	VINCENT	C	CIC	WATCH ASS
S. CH. PO	OHANIAN	STEPHEN	A	TORPEDO ST	TOR. CONTR.
CH. PO	THORNGATE	RONALD	A	DAM. ST 1	LEADER
CH. PO	WHISTON	PETER	B	BRIDGE	COM. LEADER
CH. PO	TRAWIS	LARRY	B	RADIO	RADIO LEADER
CH. PO	WHERRY	BRUCE	C	RADIO	RADIO LEADER
PO 1 CLASS	TRASON	ARTHUR	C	DAM. ST 1	LEADER
PO 1 CLASS	HARPER	JIM	C	BRIDGE	COM. LEADER
PO 2 CLASS	HARMAN	BILL	A	BRIDGE	COM. ASS
PO 3 CLASS	HARKINS	BOB	B	CIC	SURF.RADAR
SEAMAN	SANDERS	THOMAS	C	CIC	SURF.RADAR
SEAMAN	TIBBERY	MARK	C	BRIDGE	OBSERVER
SEAMAN	VINSON	RALPH	B	BRIDGE	OBSERVER
SEAMAN	WHISLER	ALBERT	A	BRIDGE	HELMSMAN
SEAMAN	ZIMMERMAN	ROY	B	BRIDGE	HELMSMAN
SEAMAN	SELBICKY	ANDREW	A	BRIDGE	OBSERVER

DUTY ON DOCK

RANK	LAST NAME	FIRST NAME	DUTY
LT.	MALLORY	TIMOTHY	NAV. OFFICER
LT.	MALLORY	TIMOTHY	COM. OFFICER
LT.	KENNEDY	ERNEST	OPER. OFF
LT.	MARSHALL	JAMES	ASW OFFICER
LT.	MARSHALL	JAMES	ASW ASS
LT.	PATTISON	HOWARD	WEAPON OFF
LT.	PORTER	JACK	MIS. OFFICER
LT. J. G.	MALLERICH	ANDREW	CIC OFFICER
ENSIGN	KEELEX	ROBIN	OPER. ASS
ENSIGN	FINCH	DAVID	OPER. ASS
ENSIGN	LANSON	ARMAND	NAV. ASS
ENSIGN	RAMSDELL	MANDEL	WEAPON ASS
M. CH. PO	RUTHERFORD	ANTHONY	CIC ASS
M. CH. PO	SCAULTON	FRANC	NAV. LEADER
M. CH. PO	TOWBER	NICK	RADIO LEADER
M. CH. PO	TOWBER	NICK	RADIO TEAM
S. CH. PO	SCOTT	BEN	AC ASS
S. CH. PO	SEVERSON	VINCENT	AD ASS
S. CH. PO	OHANIAN	STEPHEN	TORPEDO
CH. PO	THORNGATE	RONALD	NAV. ASS
CH. PO	WHISTON	PETER	COM. ASS
CH. PO	TRAWIS	LARRY	RADIO TEAM
CH. PO	WHERRY	BRUCE	RADIO TEAM
PO 1 CLASS	TRASON	ARTHUR	NAV. TEAM
PO 1 CLASS	HARPER	JIM	COM. TEAM
PO 2 CLASS	HARMAN	BILL	COM. TEAM
PO 3 CLASS	HARKINS	BOB	CIC TEAM
SEAMAN	SANDERS	THOMAS	CIC TEAM
SEAMAN	TIBBERY	MARK	NAV. TEAM
SEAMAN	VINSON	RALPH	NAV. TEAM
SEAMAN	WHISLER	ALBERT	NAV. TEAM
SEAMAN	ZIMMERMAN	ROY	NAV. TEAM
SEAMAN	SELBICKY	ANDREW	NAV. TEAM

ABANDON SHIP

RANK	LAST NAME	FIRST NAME	No
LT.	MALLORY	TIMOTHY	1
LT.	MALLORY	TIMOTHY	14
LT.	KENNEDY	ERNEST	14
LT.	MARSHALL	JAMES	8
LT.	MARSHALL	JAMES	8
LT.	PATTISON	HOWARD	4
LT.	PORTER	JACK	4
LT. J. G.	MALLERICH	ANDREW	4
ENSIGN	KEELEX	ROBIN	7
ENSIGN	FINCH	DAVID	2
ENSIGN	LANSON	ARMAND	4
ENSIGN	RAMSDELL	MANDEL	14
M. CH. PO	RUTHERFORD	ANTHONY	14
M. CH. PO	SCAULTON	FRANC	5
M. CH. PO	TOWBER	NICK	14
M. CH. PO	TOWBER	NICK	5
S. CH. PO	SCOTT	BEN	4
S. CH. PO	SEVERSON	VINCENT	4
S. CH. PO	OHANIAN	STEPHEN	14
CH. PO	THORNGATE	RONALD	2
CH. PO	WHISTON	PETER	1
CH. PO	TRAWIS	LARRY	5
CH. PO	WHERRY	BRUCE	5
PO 1 CLASS	TRASON	ARTHUR	4
PO 1 CLASS	HARPER	JIM	14
PO 2 CLASS	HARMAN	BILL	1
PO 3 CLASS	HARKINS	BOB	4
SEAMAN	SANDERS	THOMAS	4
SEAMAN	TIBBERY	MARK	1
SEAMAN	VINSON	RALPH	10
SEAMAN	WHISLER	ALBERT	2
SEAMAN	ZIMMERMAN	ROY	2
SEAMAN	SELBICKY	ANDREW	7

LIST OF REFERENCES

1. Sakti P. Ghosh, Data Base Organization For Data Management (Second Edition), Academic Press, Inc., 1986.

2. Gio Wiederhold, File Organization For Data Base Design, Mc Graw-Hill Book Company, 1987.

3. Fred R. McFadden and Jeffrey A. Hoffer, Data Base Management, The Benjamin/Cummings Publishing Company, 1985.

4. David M. Kroenke and Kathleen A. Dolan, Data Base Processing (Third Edition), Science Research Associates, Inc., 1988.

5. Henry F. Korth and Abraham Silberschatz, Data Base System Concepts, Mc Graw-Hill, Inc., 1986.

6. Jeffrey D. Ullman, Principles Of Data Base Systems, Computer Science Press, 1980.

7. Peter P. Chen, Entity-Relatioship Approach To Information Modeling And Analysis, Elsevier Science Publishers B.V., 1983.

8. Hammer M. and Mc Leod D., "Database Description with SDM: A Semantic Database Model", ACM Transactions on Database Systems, Vol. 6, No 3, September 1981.

9. David Kroenke, Data Base Processing (Second Edition), Science Research Associate, Inc., 1983.

10. George Tsu-der Chou, dBase III plus Handbook (Second Edition), Que Corporation, 1986.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraguate School Monterey, California 93943-5000	2
3.	Computer Technology Curricular Office Code 37 Naval Postgraduate School Monterey,California 93943-5000	1
4.	Prof. S. H. PARRY, Code 55PY Department of Operations Research Naval Postgraduate School Monterey, California 93943-5000	1
5.	Prof. V. Y. LUM, Code 52LU Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	1
6.	Hellenic Navy General Staff GEN/B4 Sratopedo Papagou Holargos 15562 Athens, GREECE	5
7.	LT Elefsiniotis Dimitrios Hellenic Navy General Staff Stratopedo Papagou Holargos 15562 Athens, GREECE	2