



NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

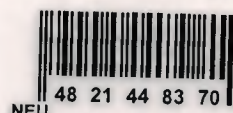
Library and Book Loaning System

**Graduation Project
COM- 400**

Student: Ahmed Hammad (980621)

Supervisor: Assist. Prof.Dr. Erdal ONURHAN

Nicosia - 2003





ACKNOWLEDGMENT

“First, I would like and foremost to thank Allah whom its accomplishment would not have been possible.

Second, I would like to thank my supervisor Assoc. Prof. Dr ERDAL ONURHAN for his invaluable advice and belief in my work and myself over the course of this Graduation Project.

Third, I am deeply indebted to my parents for their love and financial support. They have always encouraged me to pursue my interests and ambitions throughout life.

Especially my Father: Fahmy Hammad

Forth, I thank my family for their constant encouragement and support during the preparation of this project.

*Especially my brothers: Ashraf, Shakar and Kamal
Especial thanks for Mr. Muhammad Awais for his supports*

Last but not least I would also like to thank all of my friends especially Ahmed Asi ,

ABSTRACT

The purpose of this project is the development of library and book loaning system. The main problems which we have come across in library and book loaning system have been analyzed. The relationship between student information and book information, and using one function of relationship. The main structures and elements of database system for these problems are clarified. The operation principles of each blocks of the information system are modeled in Microsoft access programming language. The developed system allows to make registration book and students easily and decreasing time response of the system. Over the past decades people have transferred in maintaining records through paper and pen, and now we are evolving into the technology aria.

This project has taken a lot of time and effort to send out a very clear and simple program in Microsoft access concerning any library.

This system has been designed in a way that it would work more speedy than the normal record keeping system.

AKNOWLEDGMENT	i
ABSTRACT	ii
TABLE OF CONTENTS	iii
INTRODUCTION	1
CHAPTER ONE: WHAT IS DATABASE	2
1.1. Overview	2
1.2. Data Model	2
1.3. Database Processing	3
1.3.1 Database Recovery	3
1.4. A Simplified Database Structure	3
1.5. Difference between a DBMS and a Conventional File Processing System	5
1.5.1 Some Important Terms	6
1.6. What is a Relational Database?	8
1.6.1. Relational Databases versus Database Servers	9
1.7. Relationships and Joins	10
1.8. Primary and Foreign Keys	13
1.9. Database management on PC	15
1.10. Advantages of DBMS	17
1.11. Disadvantages of DBMS	18
CHAPTER TWO: DATABASE IN MICROSOFT ACCESS	18
2.1. Relational Databases	18
2.2. Relational Database Terminology	19
2.3. Database Capabilities	20
2.4. Main Functions of a Database	20
2.5. Microsoft Access as an RDBMS	20
2.6. Data Definition and Storage	20
2.7. Data Manipulation	24
2.8. Data Control	26
2.9. Microsoft Access as Something More	27
2.10. Developing Application Logic	28
2.11. Deciding to Move to Database Software	29
2.12. Reasons to Switch to a Database	31
2.13. The Architecture of Microsoft Access	32
2.14. The Major Objects In An Access Database	32
2.14.1. Table	32
2.14.2. Query	33
2.14.3. Form	33
2.14.4. Report	33
2.14.5. Macro	33
2.14.6. Module	33
2.15. The Uses of Microsoft Access	34
2.15.1. In a Small Business	35
2.15.2. In Contract Work	37
2.15.3. In a Large Corporation	39
2.16. Connection/disconnection data in a utility system	40
2.17. Workgroup Applications	41
2.18. Information Processing Systems	42
2.18.1. A Personal RDBMS	42

CHAPTER THEREE: LIBRARY AND BOOK LOANING PROGRAM	43
3.1. Introduction About Program	43
3.2. Program Tables	43
3.2.1. Password Table	43
3.2.2. Authentication Table	44
3.2.3. Book Table	44
3.2.4. Student Table	45
3.2.5. Date Table	46
3.2.6. Relationships	47
3.3. Entering to Program	47
3.3.1. Input Password Form	47
3.3.2. The Code of Input Password Form	48
3.3.3. Change Password Form	48
3.3.4. The Code of Change Password	49
3.4. The Main Form	51
3.4.1. The Code of Main Form	51
3.4.2. Book Loaning Form	51
3.4.3. The Code of Book Loaning	52
3.4.4. Student Information Form	53
3.4.5. Loaning Information Form	53
3.4.6. The Code of Student Information Form	54
3.4.7. The Code of Loaning Information Form	55
3.4.8. Return Book form	56
3.4.9. Code of Return Book form	56
3.4.10. Edit Student Information Form	57
3.4.11. The Code of Edit Student Information Form	57
3.5. Book Store Form	58
3.5.1. The Code of Book Store From	59
3.5.2. Add New Book Form	59
3.5.3. The Code of Add New Book from	60
3.5.4. Delete Book Form	61
3.5.5. The Code of Delete Book Form	62
3.5.6. Edit Book Information Form	63
3.5.7. The Code of Edit Information From	63
3.5.8. Search For Book From	64
3.5.9. The Code of Search For Book From	64
3.5.10. Search for Book by Author Name	65
3.5.11. The Code of Search for Book by Author Name	66
3.5.12. Search for Book by Book Name	66
3.5.13. The Code of Search by Book name Form	67
3.5.14. Search by Book Subject Form	68
3.5.15. The Code of Search by Book Subject From	69
CONCLUSION	76

INTRODUCTION

A database is a fully connected repository of information. "Fully Connected" is its one of the most important property. This property indicates that the pieces of information stored in the database are inter-connected either semantically or through some link. This connectivity allows the user to go from one piece of information to another piece in the same session. Thus, a user after accessing the job history of an employee can move to company history in the same session. In the study of the database we learn how the information pieces are created, how to link them together, how to navigate through them, how to access the desired piece of information, how to keep the information always consistent, and so on. In this project I will study all these in detail. To fully understand all these and to be able to use them require that a number basic concepts must be clearly understood. Database programs have been available for personal computers for a long time. Unfortunately, many of these programs have been either simple data storage managers that aren't suitable for building applications or complex application development systems that are difficult to learn and use. Even many computer-literate people have avoided the more complex database systems unless they have been handed a complete. By using microsoft access I create a program that could be used for automation circulation desk of any library. The reason behind this is to make the operation accurate, organized and faster which result in saving time and efforts.

CHAPTER 1

WHAT IS DATABASE?

1.1. Overview

In the simplest sense, a database is a collection of records and files that are organized for a particular purpose. On your computer system, you might keep the names and addresses of all your friends or customers. Perhaps you collect all the letters you write and organize them by recipient. You might have another set of files in which you keep all your financial data accounts payable and accounts receivable or your checkbook entries and balances. The word processor documents that you organize by topic are, in the broadest sense, one type of database. The spreadsheet files that you organize according to their uses are another type of database. If you're very organized, you can probably manage several hundred spreadsheets by using folders and subfolders. When you do this, you're the database manager. But what do you do when the problems you're trying to solve get too big? How can you easily collect information about all customers and their orders when the data might be stored in several document and spreadsheet files? How can you maintain links between the files when you enter new information? How do you ensure that data is being entered correctly? What if you need to share your information with many people but don't want two people to try updating the same data at the same time? Faced with these challenges, you need a database management system (DBMS).

1.2. Data Model

Data is stored in the database for processing. As mentioned before data processing is necessary to record and represent the facts. When the data volume is large, then for efficient processing a model is required. For example, if one has to manage thousands of SSN, phone numbers, and addresses, then it is helpful to organize them in some form. You may put SSN first, then phone numbers and, finally addresses. This means you have now a model, which also defines searching methods, new data inserting methods, and so on. So a model is a data representation template that defines rules for processing these data. We will study in great detail different kinds of model and their properties.

1.3. Database Processing

A database must always store facts so that a user always receives the correct information. It is possible that the same data might be modified by more than one user. For example, the husband may deposit some money and at the same time his wife may also deposit or withdraw some money from the same joint account. If these operations are not processed carefully, then the final value may not represent the fact. The database system uses complex mechanism, referred to as “Concurrency Control Mechanism (CCM)” to guarantee correctness (consistency) of database data. We will study these mechanisms in brief later.

1.3.1. Database Recovery

Database system may fail for a variety of reasons. If it does, then it must be recovered so that most recent processing is not lost. Database recovery is a complex process and we will study in brief database system recovery protocols.

1.4. A Simplified Database Structure

We now look at graphically what we discussed. We study a very simple database.

Part of the real world: Undergraduate Students.

Name of the database: Undergraduate Student Database

Entities: Students, Prerequisite, Grade report, Courses, and Section.

Table 1.1 Relation Data Model

Student				Prerequisite		Grade Report		
Name	Student No.	Class	Major	Course No.	Prerequisite	Sid	SecID	Grade
Smith	17	1	CS	CS431	CS352	1	A	A
Brown	8	2	Phy	CS470	CS352	2	A	B
						3	B	A
						4	C	C

Course				Section				
Course	Course No	Credit Hrs.	Dept.	Sec.	Course No.	Sem.	Yr	Instructor
Intro. To C	CS101	3	CS	B	CS101	F	98	Hines
Intro to OS	CS431	3	CS	A	CS431	W	99	Appie
Intro to DBS	CS470	3	CS	A	CS470	W	99	Kumar
Intro to DS	CS352	3	CS	A	CS352	S	97	Mullins

Information about students is routinely processed so data must be stored in a suitable data model. The model, which is used here, is called Relation Data Model, which will be discussed in detail later. Data values are stored in a tabular structure and the related semantics (i.e., meaning of each entity, attribute, etc.) is stored in database catalogue, which is referred to as “metadata”. These entities are modified by a set of system software modules. A partial list of operation a user can perform using the set of software modules are as follows. Note that all these activities can be performed in a single session.

Reads a row (group of fields: Course, Course No. Credit Hrs, Dept., etc.).

Modifies a field of a row.

Create a row with desired fields.

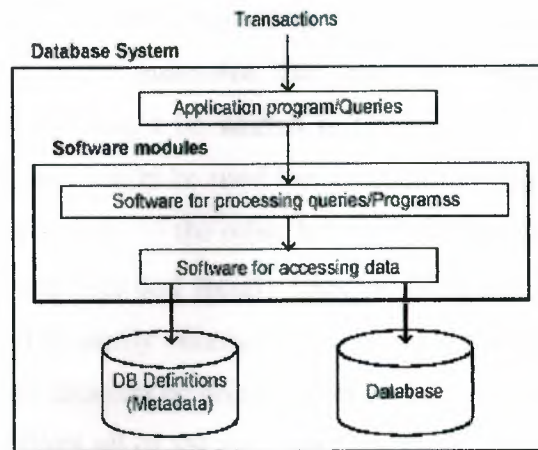
Create a file (set of rows).

Delete a file/record/field.

Print the desired information.

Store new information in a file, etc.

A user issues a query to the database and the database management system makes sure that the request is processed and granted. The entire processing requires a number of components (hardware and software) for completion. The following diagram briefly illustrates the components of a database management system.



A simplified Database Management System architecture

Figure 1.1 asimplified database manegment system architecture

1.5. Difference between a DBMS and a Conventional File Processing System

A conventional file processing system can also process data stored in these files, however, it has a number of limitations, which makes it unsuitable for processing and managing data efficiently.

SUPPOSE WE HAVE THE FOLLOWING SET OF FILES IN A UNIVERSITY:

- Faculty data files containing salary information. Processing software Payroll system
- Class data files containing class scheduling data and class timetable. Processing software Scheduling system
- Student data files containing grade report information. Processing software Grade posting system.

SUPPOSE A USER WANTS TO GET THE ANSWER OF THE FOLLOWING QUERY:

Get the salary paid to each instructor who teaches a class scheduled by the class scheduling system. The conventional file processing system treats each file separately. The payroll system processes only the faculty file, the class scheduling system processes only the class data, and the grade posting system processes only student data. To obtain the answer to the

query three programs must be executed separately in three independent sessions. In this scheme, unfortunately, there is no guarantee that data files would be compatible and consistent. The faculty data file might be written in COBOL binary format, whereas an incompatible PL/1 record format might be used for the class data file. In this situation one file must be converted to the format of the other before it can be processed. Furthermore, same data may exist in student files and faculty files, which could be inconsistent. These problems and limitations can be easily eliminated by database management system, where the faculty, class, and student data can be processed as an integrated whole. Since the files have been created by the DBMS all of the data are compatible. Such processing is called integrated processing and so the name of such system is Integrated Database Management System. The following diagram illustrates the way a DBMS handles the files.

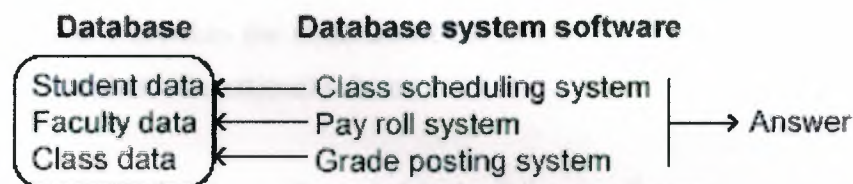


Figure 1.2 The illustration way DBMS handles the files

1.5.1. Some Important Terms

- **Structure:** Explains how related components are put together and a set of rules to manage them.
- **Instances (Occurrence):** When correct values are assigned to the structure, then an instance of the structure is created.

The following table illustrates the structure and its multiple occurrences.

Table 1.2 Illustrate the structure and its multiple occurrences

Structure		Instance 1		Instance 2		Instance 3	
Name	SSN	Name	SSN	Name	SSN	Name	SSN
		Stack	112233445	Mullins	100223344	Ram	100100100
		Appie	111222333	Hines	188990077	Hari	222333898
				Peter	599599599	Prasad	123412345
						Tony	119988777

• **Schema:** Structure of the database. So in database terminology the structure is a schema and the instance of a schema is a database.

Generally there are three types:

Internal schema: Describes how the data is going to stored on the disk.

Conceptual schema: Describes the structure of the database to the database designer.

External schema: Describes the structure of the database to end users. An end user gets the view of the database via external schema.

The three schemas relationship is illustrated in the following figure:

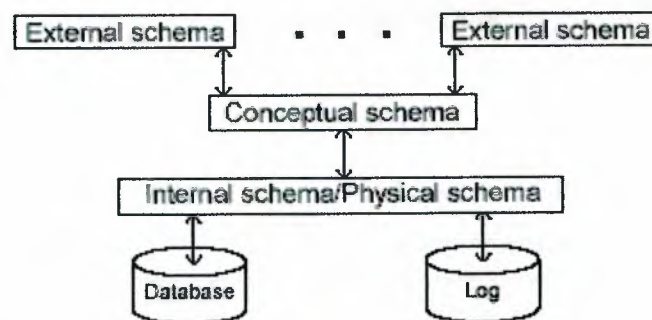


Figure 1.3 Three schemas relationship

Data Definition Language (DDL): A high level language to define data types. Every database system has its own DDL.

Data Manipulation Language (DML): A language to manipulate database.

Storage Definition Language (SDL): Describes how data will be stored on the database disks.

View Definition Language (VDL): Describes how view will be presented to an end user.

1.6. What is a Relational Database?

A relational database stores all its data inside tables, and nothing more. All operations on data are done on the tables themselves or produces another tables as the result. You never see anything except for tables. A table is a set of rows and columns. This is very important, because a set does not have any predefined sort order for its elements. Each row is a set of columns with only one value for each. All rows from the same table have the same set of columns, although some columns may have NULL values, i.e. the values for that rows was not initialized. Note that a NULL value for a string column is different from an empty string. You should think about a NULL value as an "unknown" value. The rows from a relational table is analogous to a record, and the columns to a field. There are two basic operations you can perform on a relational table. The first one is retrieving a subset of its columns. The second is retrieving a subset of its rows. Here are samples of the two operations:

```
SELECT NAME, E_MAIL FROM ADDR_BOOK
```

NAME	E_MAIL
Fernando Lozano	fsl@centroin.com.br
Bill Gates	bill@microsoft.com

```
SELECT * FROM ADDR_BOOK WHERE COMPANY = 'EDM2'
```

NAME	COMPANY	E_MAIL
Fernando Lozano	EDM2	fsl@centroin.com.br

You can also combine these two operations, as in:

```
SELECT NAME, E_MAIL FROM ADDR_BOOK WHERE COMPANY = 'EDM2'
```

NAME	E_MAIL
Fernando Lozano	fsl@centroin.com.br

You can also perform operations between two tables, treating them as sets: you can make cartesian product of the tables, you can get the intersection between two tables, you can add one table to another and so on. Later we'll present more details about these operations and how they can be useful.

1.6.1. Relational Databases versus Database Servers

Not all databases are relational, and not all relational databases are built on the client/server paradigm. But most of the time you'll want a relational database server, so it's important to clarify the distinction. Remember: a relational database manipulates only tables and the result of all operations are also tables. The tables are sets, which are themselves sets of rows and columns. You can view the database itself as a set of tables.

So a DBF file is not a relational database. You do not manipulate a DBF table as a set (you are always following an index) and you do not perform operation on tables that yield other tables as the result (you are just looping through records from one or more tables, even when you use the "SET RELATION" dBase statement). Most database file formats are not relational databases. Even the Btrieve server NLM is **not** a relational database, because you do not operate on sets tables or sets of tables. Conversely, a MDB file (from MS Access) is a relational database. Although you can open and manipulate a MDB file just like a DBF file, navigating through records and index, you can also perform all operations through a relational view of the database and using SQL statements. Actually, most non-relational databases are based on some "navigational" model: an hierarchy, a linked list, a B-Tree, etc. It's common to refer to these as ISAM (Indexed Sequential Access Method) Databases. Now let's see what is a database server: it's a specialized process that manages the database itself. The applications are clients to the database server and they never manipulate the database directly, but only make requests for the server to perform these operations. This allows the server to add many sophisticated features, such as transaction

processing, recovery, backup, access control and etc without increasing the complexity of every application. The server also reduces the risk of data file corruption, if only because only the server writes to the database (a crash on any client machine will not leave unflushed buffers).

A nice database server also takes advantage of the client/server architecture to lower network usage. If you open a DBF or MDB file stored on a file server you need to retrieve every record just to filter out which ones you really need. But if you connect to a database server, it filters out the unneeded records and send to the client only the data that really matters. Access is a relational database but it is not a database server. mSQL, SQL Anywhere, DB2, Oracle are both relational databases and database servers. The Btrieve NLM is a database server but it is not a relational database.

1.7. Relationships and Joins

Most set operations between tables are interesting but of limited use. After all, they will work as expected only when the tables have the same set of columns. The fun begins when you operate on tables that do NOT have the same set of columns. For example, see the table COMPANY:

Table 1.3 set of columns

NAME	URL
EDM2	http://www.edm2.com
Microsoft	http://www.microsoft.com

You want to establish a relationship between the tables COMPANY and ADDR_BOOK we've seen before. These tables have a common column, the name of the company. Even if each table has its own name for the column, we see that the data stored and its meaning is the same on both tables. So we could use this relationship to get a URL for each person on ADDR_BOOK. Here's the SQL statement:

```
SELECT ADDR_BOOK.NAME, COMPANY.URL  
FROM ADDR_BOOK, COMPANY
```

```
WHERE ADDR_BOOK.COMPANY = COMPANY.NAME
```

Maybe this example was not so useful, but the simple idea of establishing relationships between tables through column values is the basis of most commercial information systems today. This operation, matching rows from one table to another using one or more column value, is called a "join", more specifically an "inner join".

Let's go on. Imagine an order form from your preferred on-line shop site. The order itself has the name of the customer, the address of delivery and payment information. Besides, each order has one or more items which the customer has ordered and will be delivered together. On most databases, CHAR appends spaces to fill the column length, but VARCHAR does not] And to list all items from a particular order, say order no. 12345:

```
SELECT * FROM ORDER_ITEMS WHERE ORDER_NO = 12345
```

So we have two tables, ORDER and ORDER_NO and a relationship between these two tables on field ORDER_NO. The field ITEM_NO allows us to identify each item from the same order. But maybe your preferred site allows you to register so you do not have to retype the delivery address each time you shop. This leads us to a third table, CUSTOMER, and a relationship between ORDER and CUSTOMER. And to print the mailing label for each order, you'd use the following query:

```
SELECT ORDER.ORDER_NO, CUSTOMER.NAME, CUSTOMER.ADDRESS,  
CUSTOMER.CITY, CUSTOMER.ZIP_CODE, CUSTOMER.COUNTRY  
FROM ORDER, CUSTOMER  
WHERE ORDER.CUST_NO = CUSTOMER.CUST_NO
```

Note that a join need not match one row to only one other row. It can match one row to a set of rows from other table, as long as all rows match the join condition.

Some customers may have no order on the database (eg, their older orders were moved to an "historic" database). These customers will not show on the previous query, because they would not match the join condition. Well, it's unlikely you'll ever list labels for all orders,

but maybe you want the mailing labels for all orders entered at November, 20. This shows that a join operation can be combined with other restrictions:

```
SELECT ORDER.ORDER_NO, CUSTOMER.NAME, CUSTOMER.ADDRESS,  
CUSTOMER.CITY, CUSTOMER.ZIP_CODE, CUSTOMER.COUNTRY  
FROM ORDER, CUSTOMER  
WHERE ORDER.CUST_NO = CUSTOMER.CUST_NO  
AND ORDER.DATE_ENTERED = '1998-20-11'
```

Some databases may have different syntax for date values, so check your product documentation if this query does not work as expected. We could continue growing our database as we refine our application. You'd probably have a PRODUCTS table, a SHIPPING table for the shipping methods and costs, and so on. But before we go to another topic, let's see another example on how powerful relationships can be. Take your favorite web software archive. It probably has many categories on which each package is cataloged. These categories generally have sub categories, such as "Internet/browsers" or "Applications/Graphics/Converters". Such an hierarchical structure is easily implemented as a self-relationship. Note the use of a NULL value when the Category has no parent (it's a root category). This is generally better than use a special value such as zero or -1 because NULL will never be a valid value whatever the data type of the column. The application that browses through the software archive will start listing all rows at the root level:

```
SELECT ID, NAME from CATEGORY WHERE PARENT = NULL
```

And, when the user selects a category to enter (or expand) the application would list all subcategories from the selected one, say "Applications":

```
SELECT ID, NAME from CATEGORY WHERE PARENT = 3
```

The SELECT statement from the SQL language is very powerful. We have seen only the tip of the iceberg. Many books have been written only about SQL syntax and capabilities,

and a deeper exploration on the subject would miss the focus of this article. For example, you can sort the result, compute sums, means and other statistic functions, group the data by one or more column values and perform "outer joins". Please visit The mSQL PC Homepage (<http://www.blnet.com/msqlpc>) to get links for more information about the SQL query language.

1.8. Primary and Foreign Keys

Each time you have data inside a relational table, you need a way to identify each row stored into that table. For example, say Fernando Lozano has changed his e-mail address. How do I know the right row to update? Given the table ADDR_BOOK we've already been presented, I'd say:

```
UPDATE ADDR_BOOK SET E_MAIL = lozano@blnet.com
      WHERE NAME = 'Fernando Lozano'
```

So the column NAME identifies each row from ADDR_BOOK. Then, NAME is said to be the primary key from table ADDR_BOOK.

Well, name is not a good primary key. There's a possibility you'll have another person named 'Fernando Lozano'. Once I was searching through AltaVista and found more than a dozen personal home pages for 'Fernando Lozano', and none were mine! Besides, a woman can change her name when she marries. This leads to what constitutes a nice primary key: It should uniquely identify every possible data for the table, and its value should not change over time. Sometimes the data you want to store already has one or more columns suitable to be the primary key. If you have the social security number of your employees, this can be the primary key for table EMPLOYEE. You can have a composite primary key. For example, my software archive may have many versions of Netscape Communicator, but only one version 4.04 (I'd not have duplicate copies of any software on my archive). So the primary key for table SOFTWARE could be (NAME, VERSION). Maybe I'd add the field LANGUAGE if my archive contains both English, Portuguese and Germany versions of Communicator.

Even if you have a set of columns that meet the requirements for a primary key, you may want to create a "synthetic" primary key. Computers take much more time comparing

strings than integers. Shorter strings are much faster than long ones. A long string may lead to typing mistakes. Comparing two or three columns when searching for a record is slower than searching for only one columns. This way we end up with fields like CUST_NO and ORDER_NO. Another reason to create a syntethic primary key is the use of its value as a foreign key by another tables. Remember our order entry system: table ORDER_ITEM has a column named ORDER_NO that identifies the order that contains each item. ORDER_NO is a foreign key for table ORDER_ITEM. Some databases have an auto-increment data type to be used for syntethic primary keys. Other let you explicitly define a "sequence" as a stand alone entity or associated to a specific table. I prefer the sequences as they provide more control when importing/exporting data and make easier to insert related rows into other tables. When the primary key of a table is too long a string or composite by many columns, other related tables may spend more space storing its foreign keys than the actual data they were designed to store.

For example, a software database may have a table SOFTWARE with columns SOFT_NAME, VERSION_NAME, VERSION_NO, BUILD_NO, PLATFORM, LANGUAGE, RELEASE_DATE, SUPPLIER, SIZE_BYTES, DESCRIPTION and many other columns so you can build a nice software archive or a bug tracking database.

The six first fields uniquely identifies every software. You need all them. Think about Netscape Communicator Professional 4.04 for OS/2 Brazilian Portuguese. Its key would be ('Communicator', 'Professional', '4.04', '1.0', 'OS/2', 'PT_BR').

The '1.0' is needed so you know this is not one of the beta releases nor a bug fix for 4.04.

Our database may register many locations, or copies, for this software: an FTP site, a CD-ROM media, a ZIP drive, and so on. I want to keep track of all copies so I can quickly find one when I need and I can also release storage from obsolete software. I'd have a table COPIES which has a many-to-one relationship with SOFTWARE, that is, one SOFTWARE has one or more COPIES. This means every row from COPIES must store the value of the primary key of a row from SOFTWARE, all six columns. That's too much data for such a simple thing! A syntethic primary key will make our design easier to understand, easier to program and faster for the computer. The table COPIES would need only one column for its foreign key. Some people may even tell you to always create

synthetic keys for all tables. I'll not go so far, but use common sense when choosing the primary keys and foreign keys for your tables.

1.9. Database management on PC

Most commercial databases are relational database management systems (RDBMSs). The most common RDBMSs on desktop microcomputer systems are Microsoft's Access that is included with MS Office 97 or MS Office 2000 (professional versions), Borland's Paradox, and Lotus Development's Approach.

Data from several files (tables) are combined by a RDBMS through the fields (columns) that the files have in common. The name relational database management system implies that the software relates data in different files (tables) by common fields in those files.

Interrelated tables. A system that interrelates the files is critical for the type of information-retrieval task used in RDBMSs. DBMS-like packages that do not automatically interrelate files store information in flat files.

Interacting with a DBMS by users is through:

An easy-to-use retrieval/update facility that accompanies the database package. This interface lets users frame queries onscreen and interact with database applications, or

An applications program written in a programming language.

A database processing environment for a typical microcomputer system would resemble the following:

Data definition is the first step in creating a database.

Data definition involves specifying to the DBMS the properties of the data that are to go into the database.

Information specified is referred to as the file structure and includes the following type of information:

The name of each field.

The maximum length of each field.

The type of data (i.e. text, numeric) that each field will store.

In creating a file structure, you might specify fields in an inventory table as:

Table 1.4 Creating a file structure example

<u>Field name</u>	<u>Field type</u>	<u>Width</u>	<u>Decimals</u>
Product number	Text	4	N/A
Uncommitted stock	Numeric	10	0
On order?	Logical	1	N/A

In most DBMSs, arithmetic computations can be performed on data that have been declared as numeric but not as text or logical. The text, numeric (number), and logical yes/no designations are commonly called field descriptors. The data dictionary monitors the application environment and won't let you enter or use data in a conflicting way. For example you couldn't enter a ten character text field if the data dictionary limits the field to seven characters. Likewise, the data dictionary wouldn't let you add text field to a numeric field. Data manipulation refers to the process of creating data for a database or using the database in a hands-on fashion. Some data manipulation activities include: Creation of database data. File maintenance. Information retrieval (Query). Structured query language (SQL) is the most common standard for information retrieval. Query by example is usually used instead of writing SQL similar to:

1.10. Advantages of DBMS

Data Integration: Files are integrated so no separate sessions are required to process data stored in different files.

Minimizing Data Redundancy: In file processing the same information may be stored in more than one file. Such duplication (redundancy) is not required in DBMS.

Data Independency: In file processing system the program such as payroll package interacts directly with the data file. This requires that the program must contain the description of the format (record type, field type, file organization etc.) of the file it uses. This creates problem when a file is changed. For example if the ZIP field is expanded to nine digit, all programs that access a file containing ZIP code will need to be modified,

even if those programs do not manipulate ZIP code. This situation does not occur in aDBMS.

Physical Data Independency: It ensures that the physical organization of data files is transparent to the application programs. For example changes in the physical location such as disk to magnetic tape or to drum does not affect any application program.

Logical Data Independency: It ensures that the logical organization of data files is transparent to the application program. For example changes in a file type from index sequential to random, serial to inverted organization etc., does not affect the application programs.

Multiple Views of the Data: A database has a variety of users. Some may view the stored information differently than others. The registrar may view students as customers, a faculty may view students as nice person to have intellectual discussion. So the registrar may be interested to see a student's financial situation, a faculty may like to see students eagerness to learn a subject. These two users (registrar and faculty) will have a different set of information from the database regarding a student.

Better Data Management (Shareability and Access Flexibility): One department is responsible for maintaining the data structure, i.e. centralized management. The standard also makes sure that desired record can be accessed via many different routes. For example an employee record can be accessed via his/her social security number or via his/her name or via telephone number etc.

Reliability: Database system does not fail very often.

Integrity: Since a database represents a picture of a part of the real world at any time its contents must be consistent, i.e., tells the correct story. In a system controlled centrally it is easy to maintain data integrity. Therefore, at any time the database gives a correct picture.

1.11. Disadvantages of DBMS

Size Mainframe database management system programs tend to be very large, requiring large amounts of memory and data storage. Complexity In order to take advantage of the features provided by a DBMS, users, programmers, and system people must spend substantial effort in learning the new system.

CHAPTER 2

MICROSOFT ACCESS IS A DATABASE

2.1. Overview

If you're a serious user of a personal computer, you've probably been using word processing or spreadsheet applications to help you solve problems. You might have started a number of years ago with character-based products running under MS-DOS but subsequently upgraded to software that runs under the Microsoft Windows operating system. You might also own some database software, either as part of an integrated package such as Microsoft Works or as a separate program.

Database programs have been available for personal computers for a long time. Unfortunately, many of these programs have been either simple data storage managers that aren't suitable for building applications or complex application development systems that are difficult to learn and use. Even many computer-literate people have avoided the more complex database systems unless they have been handed a complete, custom-built database application. Microsoft Access, however, represents a significant turnaround in ease of use, and many people are drawn to it to create both simple databases and sophisticated database applications.

Now that Access is in its fourth release and has become an even more robust product in the second edition designed for 32-bit versions of Windows, perhaps it's time to take another look at how you work with your personal computer to get the job done. If you've previously shied away from database software because you felt you needed programming skills or because it would take you too much time to become a proficient user, you'll be pleasantly surprised at how easy it is to work with Access. But how do you decide whether you're ready to move up to a database system such as Access? To help you decide, let's take a look at the advantages of using database application development software.

2.1. Relational Databases

Nearly all modern database management systems store and handle information using the relational database management model. The term relational stems from the fact that each record in the database contains information related to a single subject and only that subject. Also, data about two classes of information (such as customers and orders) can

each record in the database contains information related to a single subject and only that subject. Also, data about two classes of information (such as customers and orders) can be manipulated as a single entity based on related data values. For example, it would be redundant to store customer name and address information with every order that the customer places. In a relational database system, the information about orders contains a field that stores data, such as a customer number, which can be used to connect each order with the appropriate customer information. In a relational database management system, sometimes called an RDBMS, the system manages all data in tables. Tables store information about a subject (such as customers or students) and have columns that contain the different kinds of information about the subject (for example, customers' or students' addresses) and rows that describe all the attributes of a single instance of the subject (for example, data on a specific customer or student). Even when you query the database (fetch information from one or more tables), the result is always something that looks like another table.

2.2. Relational Database Terminology

Relation: Information about a single subject such as customers, orders, students, or colleges. A relation is usually stored as a table in a relational database management system.

Attribute: A specific piece of information about a subject, such as the address for a customer or the dollar amount of a contract. An attribute is normally stored as a data column, or field, in a table.

Relationship: The way information in one relation is related to information in another relation. For example, customers have a one-to-many relationship with orders because one customer can place many orders, but any order belongs to only one customer. Students might have a many-to-many relationship with colleges because each student is interested in applying to multiple colleges, and each college receives applications from many students.

Join: The process of linking tables or queries on tables via their related data values. For example, customers might be joined to orders by matching customer ID in a customers table and an orders table.

You can also join information on related values from multiple tables or queries. For example, you can join student information with college application information to find out which students applied to which colleges. You can join employee information with contract information to find out which salesperson should receive a commission.

2.3. Database Capabilities

An RDBMS gives you complete control over how you define your data, work with it, and share it with others. The system also provides sophisticated features that make it easy to catalog and manage large amounts of data in many tables. An RDBMS has three main types of capabilities: data definition, data manipulation, and data control. All this functionality is contained in the powerful features of Microsoft Access. Let's take a look at how Access implements these capabilities and compare them to what you can do with spreadsheet or word processing programs.

2.4. Main Functions of a Database

Data definition: you can define what data will be stored in your database, the type of data (for example, numbers or characters), and how the data is related. In some cases, you can also define how the data should be formatted and how it should be validated.

Data manipulation: you can work with the data in many ways. You can select which data fields you want, filter the data, and sort it. You can join data with related information and summarize (total) the data. You can select a set of information and ask the RDBMS to update it, delete it, copy it to another table, or create a new table containing the data.

Data control: you can define who is allowed to read, update, or insert data. In many cases, you can also define how data can be shared and updated by multiple users.

2.5. Microsoft Access as an RDBMS

Microsoft Access is a fully functional RDBMS. It provides all the data definition, data manipulation, and data control features you need to manage large volumes of data.

2.6. Data Definition and Storage

While you're working with a document or a spreadsheet, you generally have complete freedom to define the contents of the document or each cell in the spreadsheet. Within a given page in a document, you might include paragraphs of text, a table, a chart, or multiple columns of data displayed with multiple fonts. Within a given column on a spreadsheet, you might have text data at the top to define column headers for printing or display, and you might have various numeric formats within the column, depending on the function of the row. You need this flexibility because your word processing document must be able to convey your message within the context of a printed page, and your spreadsheet must store the data you're analyzing as well as provide for calculation and presentation of the results.

This flexibility is great for solving relatively small, well-defined business problems. But a document becomes unwieldy when it extends beyond a few dozen pages, and a spreadsheet becomes difficult to manage when it contains more than a few hundred rows of information. As the amount of data grows, you might also find that you exceed the data storage limits of your word processing or spreadsheet program or of your computer system. If you design a document or spreadsheet to be used by others, it's difficult (if not impossible) to control how they will use the data or enter new data. For example, on a spreadsheet, even though one cell might need a date and another a currency value to make sense, a user might easily enter character data in error.

Some spreadsheet programs allow you to define a "database" area within a spreadsheet to help you manage the information you need to produce the desired result. However, you are still constrained by the basic storage limitations of the spreadsheet program, and you still don't have much control over what's entered in the rows and columns of the database area. Also, if you need to handle more than number and character data, you might find that your spreadsheet program doesn't understand such things as pictures or sounds.

An RDBMS allows you to define the kind of data you have and how the data should be stored. You can also usually define rules that the RDBMS can use to ensure the integrity of your data. In its simplest form, a validation rule might ensure that you can't accidentally store alphabetic characters in a field that should contain a number. Other rules might define valid values or ranges of values for your data. In the most sophisticated systems, you can define the relationship between collections of data

(usually tables or files) and ask the RDBMS to ensure that your data remains consistent. For example, you can have the system automatically check to ensure that every order entered is for a valid customer.

With Access, you have complete flexibility to define your data (as text, numbers, dates, times, currency, Internet links, pictures, sounds, documents, spreadsheets), to define how Access stores your data (string length, number precision, date/time precision), and to define what the data looks like when you display or print it. You can define simple or complex validation rules to ensure that only accurate values exist in your database. You can request that Access check for valid relationships between files or tables in your database.

Because Access is a state-of-the-art application for Microsoft Windows, you can use all the facilities of Dynamic Data Exchange (DDE), object linking and embedding (part of Microsoft's OLE technology), and ActiveX custom controls. DDE lets you execute functions and send data between Access and any other Windows-based application that supports DDE. You can also make DDE connections to other applications using macros or Microsoft Visual Basic for Applications (VBA). OLE is an advanced Windows capability that, in part, allows you to link objects to or embed objects in your Access database. Objects include pictures, graphs, spreadsheets, and documents from other Windows-based applications that also support OLE. Figure 1-1 shows embedded object data from the sample Northwind Traders database that ships with Access. You can see a product category record that not only has the typical name and descriptive information but also has a picture to visually describe each category. Access 97 can also act as an OLE Automation server, allowing you to open and manipulate Access database objects (such as tables, queries, and forms) from other Windows-based applications.

The screenshot shows a Microsoft Access form titled 'Categories'. It has a tabular layout with the following fields and data:

Category Information		Picture
Category Name:	Beverages	
Description:	Soft drinks, coffee, tea, beer, and ales	

Products in Category		
Product Name:	Chai	<input type="checkbox"/> Discontinued
Quantity Per Unit:	10 boxes x 30 bags	Unit Price: \$18.00
Product Name:	Chang	<input type="checkbox"/> Discontinued
Quantity Per Unit:	24 - 12 oz bottles	Unit Price: \$19.00

At the bottom, there is a record navigation bar showing 'Record: 14' and '1 of 3'.

Figure 2.1 The Categories form in the North wind Traders sample database.

Within your Access forms and reports, you can include ActiveX custom controls to enhance the operation of your application. ActiveX controls use OLE technology to provide sophisticated design objects that allow you to present complex data in a simpler, more graphical way. Most ActiveX controls provide a rich set of "actions" (called methods in object terminology) that you can call from a procedure and properties you can set to manage how the control looks and behaves. For example, you might want to let your user enter a date by selecting from a calendar picture. You could laboriously build a "calendar" form that has sets of boxes arranged in rows of seven columns and write lots of code to let the user "scroll" to the "next" or "previous" month and then click in a box to pick the date. Access 97 comes with a standard ActiveX calendar control that takes care of all the details for you. This control is used in the Contracts form in the Entertainment Schedule database that is included with this book. The form is illustrated in Figure 1-2 on the next page.

Figure 2.2 choosing a date using the ActiveX calendar control.

The user can type in contract start and end dates or click a down-arrow button to reveal the ActiveX calendar control. The user can choose a different month or year from the drop-down boxes on the control, and the control displays the appropriate month. When the user clicks on a date on the calendar control, the control passes the date back to the form to update the date field in the record. If you purchase the Office Developer Edition, you will have several additional ActiveX controls available to use in your applications. Many third-party software vendors have built libraries of ActiveX controls that you can purchase for use with Microsoft Access.

Access can also understand and use a wide variety of other data formats, including many other database file structures. You can import data to and export data from word processing files or spreadsheets. You can directly access and update Paradox, dBASE III, dBASE IV, Microsoft FoxPro, and other database files. You can also import data from these files into an Access table. In addition, Access can work with most popular databases that support the Open Database Connectivity (ODBC) standard, including Microsoft SQL Server, Oracle, DB2, and Rdb.

2.7. Data Manipulation

Working with data in a word processing or spreadsheet program is very different from working with data in an RDBMS. In a word processing document, you can include tabular data and perform a limited set of functions on the data in the document. You can also search for text strings in the original document and, with OLE, include tables, charts, or pictures from other applications. In a spreadsheet, some cells contain functions that determine the result you want, and in other cells you enter the data that

provides the source information for the functions. The data in a given spreadsheet serves one particular purpose, and it's cumbersome to use the same data to solve a different problem. You can link to data in another spreadsheet to solve a new problem, or you can use limited search capabilities to copy a selected subset of the data in one spreadsheet to use in problem-solving in another spreadsheet.

An RDBMS provides you with many ways to work with your data. You can, for example, search a single table for information or request a complex search across several related tables or files. You can update a single field or many records with a single command. You can write programs that use RDBMS facilities to read and update your data. Many systems provide data entry and report generation facilities.

Access uses the powerful SQL database language to process data in your tables. (SQL is an acronym for Structured Query Language.) Using SQL, you can define the set of information that you need to solve a particular problem, including data from perhaps many tables. But Access simplifies data manipulation tasks. You don't even have to understand SQL to get Access to work for you. Access uses the relationship definitions you provide to automatically link the tables you need. You can concentrate on how to solve information problems without having to worry about building a complex navigation system that links all the data structures in your database. Access also has an extremely simple yet powerful graphical query definition facility that you can use to specify the data you need to solve a problem. Using point and click, drag and drop, and a few keyboard strokes, you can build a complex query in a matter of seconds.

Figure 2.3 shows a complex query under construction in Access. You can find this query in the Entertainment Schedule sample database on the disc included with this book. Access displays field lists from selected tables in the upper part of the window; the lines between field lists indicate the automatic links that Access will use to solve the query.

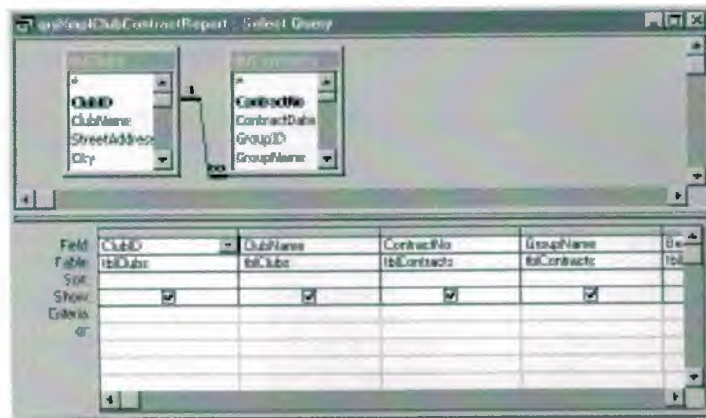


Figure 2.3 A query to retrieve club and contract information from the Entertainment Schedule database.

To create the query, you simply select the fields you want from the upper part of the window and drag them to the design grid in the lower part of the window. Choose a few options, type in any criteria, and you're ready to have Access select the information you want.

Figure 2.4 shows an example of an SQL statement that Access automatically creates from your specifications in the design grid. You don't need to be an expert to correctly construct the SQL syntax you need to solve your problem, but as you'll learn in Chapter 11, "Advanced Query DesignSQL," you can specify the SQL yourself for certain advanced types of queries. Figure 1-5 shows the result of running the query.

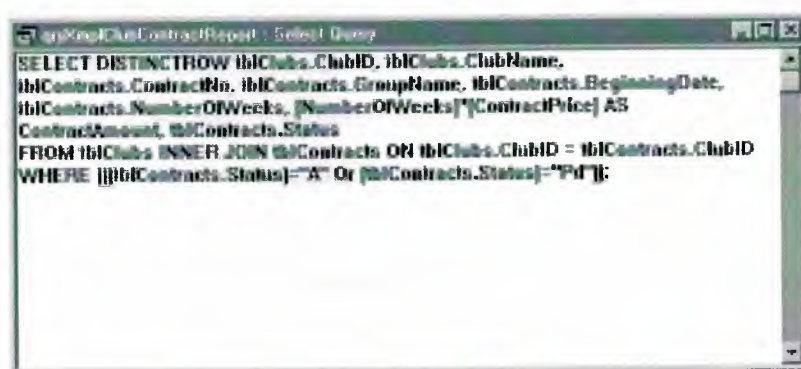


Figure 2.4 The SQL statement for a query to retrieve club and contract information.

The screenshot shows a Microsoft Access window titled 'mKeyClubContractReport - Select Query'. The window displays a table with the following data:

Club ID	Club Name	Contract	Group Name	Begin Date
000950	The Alligator Club	4688	The Belltones	Friday, July 12, 1996
000950	The Alligator Club	4695	Shaman's Apprentice	Friday, August 2, 1996
000950	The Alligator Club	4696	Bucky and the Fullere	Friday, August 9, 1996
000950	The Alligator Club	4698	Supertube	Friday, August 16, 1996
000950	The Alligator Club	4719	Supertube	Friday, October 11, 1996
000950	The Alligator Club	4720	Life Imitates Art	Friday, September 12, 1997
000950	The Alligator Club	4726	The Codpieces	Friday, October 25, 1996
000950	The Alligator Club	4734	Henry and Otis	Friday, November 8, 1996
000950	The Alligator Club	4735	Shaman's Apprentice	Friday, November 15, 1996
000950	The Alligator Club	4738	The Belltones	Friday, November 22, 1996
000950	The Alligator Club	4742	The Codpieces	Friday, November 29, 1996
000950	The Alligator Club	4743	The Codpieces	Friday, December 6, 1996
000950	The Alligator Club	4746	Internal Hemorrhage	Friday, December 20, 1996
000950	The Alligator Club	4756	The Belltones	Friday, January 10, 1997
000950	The Alligator Club	4762	Bucky and the Fullere	Friday, January 17, 1997
000950	The Alligator Club	4763	Internal Hemorrhage	Friday, January 24, 1997

The status bar at the bottom indicates 'Record: 1 of 270'.

Figure 2.5 A lists of clubs and their contracts.

2.8. Data Control

Spreadsheets and word processing documents are great for solving single-user problems, but they are difficult to use when more than one person needs to share the data. Spreadsheets are also useful for providing templates for simple data entry, but they don't do the job well if you need to perform complex data validation. For example, a spreadsheet works well as a template for an invoice for a small business with a single proprietor. But if the business expands and a number of salespeople are entering orders, you need a database. Likewise, a spreadsheet can assist employees with expense reports in a large business, but the data eventually must be captured and placed in a database for corporate accounting.

When you need to share your information with others, true relational database management systems allow you to make your information secure so that only authorized users can read or update your data. An RDBMS that is designed to allow data sharing also provides features to ensure that no two people can change the same data at the same time. The best systems also allow you to group changes (a series of changes is sometimes called a transaction) so that either all of the changes or none of the changes appear in your data. For example, while entering new order information for

a customer, you probably want to know that all items are recorded or, if you encounter an error, that none of the changes are saved. You probably also want to be sure that no one else can view any part of the order until you have entered all of it.

Access is designed to be used either as a stand-alone RDBMS on a single workstation or in a shared client-server mode across a network. Because you can share your Access data with other users, Access has excellent data security and data integrity features. You can define which users or groups of users can have access to objects (such as tables, forms, and queries) in your database. Access automatically provides locking mechanisms to ensure that no two people can update an object at the same time. Access also understands and honors the locking mechanisms of other database structures (such as Paradox, dBASE, and SQL databases) that you attach to your database. In addition, Access lets you create multiple copies of a "master" database through a process called replication. Several remote users can have their own copies of the database, and they can each use utilities built into Windows 95 and Access to periodically synchronize their copies.

2.9. Microsoft Access as Something More

Being able to define exactly what data you need, how it should be stored, and how you want to access it solves the data management part of the problem. However, you also need a simple way to automate all of the common tasks you want to perform. For example, each time you need to enter a new order, you don't want to have to run a query to search the Customers table, execute a command to open the Orders table, and then create a new record before you can enter the data for the order. And once you've entered the data for the new order, you don't want to have to worry about scanning the table that contains all your products to verify the order's sizes, colors, and prices.

Advanced word processing software lets you define templates and macros to automate document creation, but it's not designed to handle complex transaction processing. In a spreadsheet, you enter formulas that define what automatic calculations you want performed. If you're an advanced spreadsheet user, you might also create macros to help automate entering and validating data. If you're working with a lot of data, you've probably figured out how to use one spreadsheet as a "database" container and use references to selected portions of this data in your calculations.

Although you can build a fairly complex "application" using spreadsheets, you really don't have the debugging and application management tools you need to easily construct a robust data management application. Even something as simple as a wedding guest invitation and gift list is much easier to handle in a database. (See the Wedding List sample database included with this book.) Database systems are specifically designed for application development. They give you the data management and control tools that you need and also provide facilities to catalog the various parts of your application and manage their interrelationships. You also get a full programming language and debugging tools with a database system.

2.10. Developing Application Logic

When you want to build a more complex database application, you need a powerful relational database management system and an application development system to help you automate your tasks. Virtually all database systems include application development facilities to allow programmers or users of the system to define the procedures needed to automate the creation and manipulation of data. Unfortunately, many database application development systems require that you know a programming language, such as C or Xbase, to define procedures. Although these languages are very rich and powerful, you must have experience before you can use them properly. To really take advantage of some database systems, you must learn programming, hire a programmer, or buy a ready-made database application (which might not exactly suit your needs) from a software development company. Fortunately, Microsoft Access makes it easy to design and construct database applications without requiring that you know a programming language. Although you begin in Access by defining the relational tables and the fields in those tables that will contain your data, you will quickly branch out to defining actions on the data via forms, reports, macros, and VBA. You can use forms and reports to define how you want the data displayed and what additional calculations you want performed very much like spreadsheets. In this case, the format and calculation instructions (in the forms and reports) are separate from the data (in the tables), so you have complete flexibility to use your data in different ways without affecting the data. You simply define another form or report using the same data. When you want to automate actions in a simple application, Access provides a macro definition facility to make it easy to respond to events (such as clicking a button to open

Access, you store a single copy of the data in the tables you design. Perhaps one of the hardest concepts to grasp is that you store only your basic data in database tables. For example, in a database you would store the quantity of items ordered and the price of the items but you would not usually store the extended cost (a calculated value). You use a query, a form, or a report to define the quantity-times-price calculation. You can use the query facility to examine and extract the data in many ways. This allows you to keep only one copy of the basic data yet use it over and over to solve different problems. In a student tracking database, you might create one form to display individual students and the evaluation tests each student has taken. You can use a report defined on the same data to graph the test results for all students during specified time periods. You don't need a separate copy of the data to do this, and you can change either the form or the report independently, without destroying the structure of your database. You can also add new student or test information easily without having to worry about the impact on any of your forms or reports. You can do this because the data (tables) and the routines you define to operate on the data (queries, forms, reports, macros, or modules) are completely independent of each other. Any change you make to the data via one form is immediately reflected by Access in any other form or query that uses the same data. If you're wondering how you'll make the transition from word processing documents and spreadsheets to Access, you'll be pleased to find features in Access to help you out. You can use the import facilities to copy the data from your existing text or spreadsheet files. You'll find that Access supports most of the same functions you have used in your spreadsheets, so defining calculations in a form or a report will seem very familiar. Within the Help facility, the Office Assistant can suggest solutions quickly. Help also includes "how do I" topics that walk you through key tasks you need to learn to begin working with a database and "tell me about" and reference topics that enhance your knowledge. In addition, Access provides powerful wizard facilities to give you a jump start on moving your spreadsheet data to an Access database, such as the Import Spreadsheet Wizard to help you design database tables to store your old spreadsheet data. Take a long look at the kind of work you're doing today. The sidebar on the following page summarizes some of the key reasons why you might need to move to Access. Is the number of files starting to overwhelm you? Do you find yourself creating copies of old files when you need to answer new questions? Do others need to share the data and update it? Do you find yourself exceeding the limits of your current software or the memory on your system? If the answer to any of these is yes, you should

be solving your problems with a relational database management system like Microsoft Access.

2.12. Reasons to Switch to a Database

Reason 1: You have too many separate files or too much data in individual files. This makes it difficult to manage the data. Also, the data might exceed the limits of the software or the capacity of the system memory.

Reason 2: You have multiple uses for the data detailing transactions (invoices, for example), summary analysis (such as quarterly sales summaries), and "what if" scenarios. Therefore, you need to be able to look at the data in many different ways, but you find it difficult to create multiple "views" of the data.

Reason 3: You need to share data. For example, numerous people are entering and updating data and analyzing it. Only one person at a time can update a spreadsheet or a word processing document, but many people can simultaneously share and update a database table. Also, databases ensure that people reading the data see only committed updates.

Reason 4: You must control the data because different users access the data, because the data is used to run your business, and because the data is related (such as data for customers and orders). This means you must secure access to data and control data values, and you must ensure data consistency.

2.13. The Architecture of Microsoft Access

Microsoft Access calls anything that can have a name an object. Within an Access database, the main objects are tables, queries, forms, reports, macros, and modules. If you have worked with other database systems on desktop computers, you might have seen the term database used to refer to only those files in which you store data. In Access, however, a database also includes all the major objects related to the stored data; including objects you define to automate the use of your data.

2.14. The Major Objects In An Access Database

2.14.1. Table

An object you define and use to store data. Each table contains information about a particular subject, such as customers or orders. Tables contain fields (or columns) that

store different kinds of data, such as a name or an address, and records (or rows) that collect all the information about a particular instance of the subject, such as all the information about an entertainment group named The Belltones. You can define a primary key (one or more fields that have a unique value for each record) and one or more indexes on each table to help retrieve your data more quickly.

2.14.2. Query

An object that provides a custom view of data from one or more tables. In Access, you can use the graphical query by example (QBE) facility or you can write SQL statements to create your queries. You can define queries to select, update, insert, or delete data. You can also define queries that create new tables from data in one or more existing tables.

2.14.3. Form

An object designed primarily for data input or display or for control of application execution. You use forms to customize the presentation of data that your application extracts from queries or tables. You can also print forms. You can design a form to run a macro or a Visual Basic for Applications (VBA) procedure (see the sections on macros and modules below) in response to any of a number of events for example, to run a procedure when the value of data changes.

2.14.4. Report

An object designed for formatting, calculating, printing, and summarizing selected data. You can view a report on your screen before you print it.

2.14.5. Macro

An object that is a structured definition of one or more actions that you want Access to perform in response to a defined event. For example, you might design a macro that opens a second form in response to the selection of an item on a main form. You might have another macro that validates the content of a field whenever the value in the field changes. You can include simple conditions in macros to specify when one or more actions in the macro should be performed or skipped. You can use macros to open and execute queries, to open tables, or to print or view reports. You can also run other macros or VBA procedures from within a macro.

2.14.6. Module

An object containing custom procedures that you code using VBA. Modules provide a more discrete flow of actions and allow you to trap errors something you can't do with macros. Modules can be stand-alone objects containing functions that can be called from anywhere in your application, or they can be directly associated with a form or a report to respond to events on the associated form or report. You can extract with queries and display in reports or that you can display and update in forms. Notice that forms and reports can use data either directly from tables or from a filtered "view" of the data created by using queries. Queries can use VBA functions to provide customized calculations on data in your database. Access also has many built-in functions that allow you to summarize and format your data in queries. Events on forms and reports can "trigger" either macros or VBA procedures. What is an event? An event is any change in state of an Access object. For example, you can write macros or VBA procedures to respond to opening a form, closing a form, entering a new row on a form, or changing data either in the current record or in an individual control (an object on a form or report that contains data). You can even design a macro or a VBA procedure that responds to the user pressing individual keys on the keyboard when entering data!

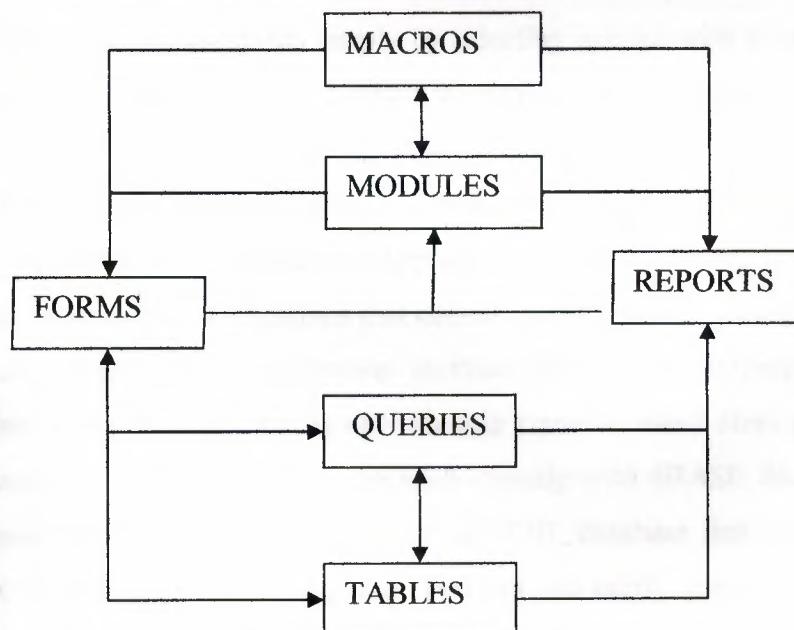


Figure 2.6 Main objects and their relationships in Access.

Using macros and modules you can change the flow of your application; open, filter, and change data in forms and reports; run queries; and build new tables. Using VBA, you can create, modify, and delete any Access object; manipulate data in your database row by row or column by column; and handle exceptional conditions. Using module code you can even call Windows Application Programming Interface (API) routines to extend your application beyond the built-in capabilities of Access.

2.15. The Uses of Microsoft Access

Microsoft Access has all the features of a classic relational database management system (RDBMS)—and more. Access is not only a powerful, flexible, and easy-to-use RDBMS, it is also a complete database application development facility. You can use Access to create and run under the Microsoft Windows operating system an application tailored to your data management needs. You can limit, select, and total your data by using queries. You can create forms for viewing and changing your data. You can also use Access to create simple or complex reports. Both forms and reports "inherit" the properties of the underlying table or query, so in most cases you need to define such things as formats and validation rules only once. Among the most powerful features of Access are the wizards that you can use to create tables and queries and to customize a wide variety of forms and reports simply by selecting options with your mouse. Access also includes wizards that help you analyze your table design, import spreadsheet or text data, improve database performance, or build and customize one of more than 20 types of applications using built-in templates. Access includes a comprehensive programming language, Microsoft Visual Basic for Applications (VBA), that you can use to create very robust "production" applications that can be shared by many users.

Finally, you get all of these development facilities not only for working with an Access database but also to attach to and work with data stored in many other popular formats. You can build an Access application to work directly with dBASE files, with Paradox and Microsoft FoxPro databases, and with any SQL database that supports the Open Database Connectivity (ODBC) standard. You can also easily import and export data as text, word processing files, or spreadsheet files. This chapter describes five scenarios, in which Access is used to meet the database and application development needs of the owner of a small business, a PC application developer or consultant, the marketing

department in a company, a management information systems (MIS) coordinator in a large corporation, and a home computer user.

2.15.1. In a Small Business

If you're the owner of a small business, you can use the simple yet powerful capabilities of Microsoft Access to manage the data you need to run your business. In addition, you can find dozens of third-party Access-based applications that will add to your productivity and make running your business much simpler. Because Access's application design facilities are so simple to use, you can be confident in creating your own applications or in customizing applications provided by others for your specific needs. Throughout much of the rest of this book, you'll read about the design and creation of a database application for an entertainment booking agency, RM Productions. This company is an actual small business in western Washington state, owned by a friend who not only handles the booking for many nightclub acts but also is an accomplished musician in his own right. RM Productions is like many small businesses. The owner, Ray McCann, realized many years ago that a personal computer system could potentially help him run his business more efficiently. At a minimum, he could keep a readily searchable list of all the clubs and groups that hired him as their booking agent, rather than depend on a manual card file. His first personal computer ran on something called CP/M (we won't ask him how long ago that was), and he eventually found a simple little database program that would let him keep track of the clubs and groups and print out contracts.

Even as his business grew, Ray hung on to his old CP/M database program. He even went so far as to install a special emulation card in his PC so that he could still run the program after he had upgraded to MS-DOS and Windows. As his entertainment booking business grew, Ray needed more sophisticated ways to send out mailings and keep track of clubs and groups that had open future dates. He got pretty creative dumping comma-separated data into an MS-DOS-based word processing program, but he quickly realized that he needed something better. Ray spent some time in early 1993 taking a look at several new Windows-based database systems. He settled on Microsoft Access. Even though he had virtually no database programming experience, it didn't take Ray long to learn how to import his club and group information into an Access database and create simple queries and mailing labels. Ray contacted me to help him expand his database to run his entire booking business. Figure 2-1 shows the central

contract booking form from that application. In this book, we'll explore many of the key elements of this database. A major portion of Ray's database application is on the sample disc included with this book.

The screenshot shows a Microsoft Access form titled "Contracts". It has a menu bar with "File", "Edit", "Format", "Tools", "Window", and "Help". Below the menu bar are buttons for "Add New", "Save", "Search...", "Cancel", "Print", and "Previous". The form contains several sections of data entry fields:

- Contract Info:** Contract # (4842), Status (Paid), Contract Date (8/22/98).
- Artist/Location:** Genre (Apes of Wrath), Artist (Lucifer's Lighthouse), Location (Lucifer's Lighthouse), Contact (Michael Griffin), Phone # ((208) 525-7447), Agent (), Style (Lounge - Dance).
- Financials:** Contract Price (\$250.00), Terms (), Commission % (10.0%).
- Scheduling:** Days/Times (Tuesday - Thursday 7:30pm to 12 Midnight, Friday - Saturday 8pm to 1:30 am), Start Date (Tuesday, July 02, 1998), End Date (Saturday, July 27, 1998), Weeks (Jul 1998), and a calendar grid showing the month of July.
- Member Info:** Member Name (), SSN (), and a section for "Additional Member No." with a grid for "Agent/Office".

At the bottom, it shows "Records: 14" and "1 of 200".

Figure 2.7 The Contracts data entry form in the Entertainment Schedule database.

The bottom line? It's true that Ray was already pretty comfortable working with a personal computer. But without Access, he would still be struggling with his outdated database and word processing system. Now he has the database he needs for his growing business. If you're a small business owner who understands that computers should be able to do more than word processing and spreadsheet programs can do, perhaps Access is for you too. You'll find a lot of computer consultants ready and able to put together an Access-based application for you in a short time and at low cost. Even if someone has constructed your database for you, you'll want to know more about Access so that you can take advantage of its native features.

2.15.2 In Contract Work

In today's highly competitive consulting marketplace, the developer who can deliver custom applications quickly and inexpensively will win the lion's share of the business. If you're a PC application developer or consultant, you'll find that the query, form, and report features of Microsoft Access allow you to create applications for your clients in record time. You can also take advantage of VBA, which is built into Access, to satisfy unique requirements and produce truly custom applications. If you've worked with

products such as Microsoft Visual Basic for Windows, you'll find the Access application development features very familiar. If you're a consultant building applications for a vertical market, you'll especially appreciate how Access makes it easy to build your core application and modify the application for each client's needs. You can create add-on features that you can price separately. Whether you're building a custom application from scratch or modifying an existing one, your clients will appreciate the fact that you can sit down with them and use Access to prototype the finished application so that they can see exactly what they'll get. You can scale your application to your clients' needs by taking advantage of the fact that Access can connect to and work with other database management systems. For smaller clients, you'll find the native Access database system more than adequate. For larger clients, you can connect your application to Microsoft SQL Server or other host databases without having to change any of the forms, reports, macros, or modules in your application. Imagine that your local high school's college counselors use a database to help seniors locate colleges of interest. Suppose the database system was built by you several years ago using an Xbase product. The high school would like to upgrade the system by converting it to run under the Microsoft Windows 95 operating system. The school would also like to connect the database system to the current student information that is kept in an SQL Server database. The school wants the new database system to help students locate colleges of interest, to track the application process, and to keep statistics on where students were accepted and where they enrolled.

Sounds as though Access might be a perfect solution. You can use the existing Xbase data or convert it easily to Access format. You can also connect the new application to the existing SQL Server student data. Adding search criteria for colleges and tracking student application data is easy. Figure 2-2 shows a search in progress in the College Counseling sample database included with this book. My son Michael built this database for his high school, the Overlake School in Redmond, Washington, as part of a senior research project. As you might guess, his contract "fee" was a passing grade!

As you might expect, Microsoft Corporation takes full advantage of its database tools to provide applications for its field marketing and sales force. I have worked on several application projects at Microsoft using Access. On one project, I built an Access database to manage a large catalog of Microsoft PowerPoint slides used for marketing presentations. Using this database, a sales representative could enter criteria for an upcoming presentation, including products of interest, type of audience, and maximum

might be interested in these books. If you own the Office Developer Edition, you can package a marketing database application so that even a computer user who doesn't own a copy of Microsoft Access can use the application.

2.15.3. In a Large Corporation

Today all companies recognize that one of the ways to remain competitive is to use computer-stored data for more than just day-to-day operations. Creative managers are constantly looking for ways to "turn data into information." As a result, companies no longer have "data processing" units; they have vast MIS departments charged with the care and feeding of the company's valuable computer-stored information.

Nearly all corporations start by building operational data processing systems. These systems collect and process the individual transactional data required to run the business on a day-to-day basis. Examples of transactional data include the following:

Checks cleared and money withdrawn and deposited in a banking demand deposit system Incoming inventory and items sold in a retail system Raw materials ordered and received and finished goods shipped in a manufacturing system Energy consumed, raw product delivered.

2.16. Connection/disconnection data in a utility system

These systems are relatively simple to design and implement with respect to the data input, the processes required on this data, and the data output. They are also easy to justify financially because they can reduce clerical tasks and, at the same time, handle rapidly growing volumes of data. (Imagine trying to post 10 million checking accounts manually.) After operational systems are in place and management becomes aware of the vast amounts of data being collected, management often begins to examine the data to better understand how the business interacts with its customers, suppliers, and competitors—to learn how to become more efficient and more competitive. Information processing in most MIS departments usually begins quite innocently as an extension of operational systems. In fact, some information processing almost always gets defined as part of an operational application system design. While interviewing users of a system during the systems analysis phase, the system designer usually hears requests such as, "When the monthly invoices are produced, I'd also like to see a report that tells me

which accounts are more than 90 days past due." Printing the invoices is not information processing. Producing the report is. On the surface, it would seem simple to answer a question about delinquent accounts, given the data about all accounts receivable. However, the operational system might require only 30 days of "current" data to get the job done. The first information request almost always begins to put demands on the data processing systems, and these demands far exceed the data and processing power needed to merely run the business. At some point, the MIS organization decides to reserve additional data storage and processing capability to meet the growing need for information. This need for information has led companies to build vast networks of departmental systems, which are in turn linked to desktop systems on employees' desks. As more and more data spreads through the corporation, the data becomes more difficult to manage, locate, and access, as on the following page makes clear. Multiple copies of the same data proliferate, and it becomes hard to determine who has the most current and accurate data.

Why do so many copies exist? Many copies of data exist because the vast majority of tools aren't designed to work with data in more than one format or to connect to data from multiple sources. Employees must resort to obtaining a copy of the data they want and then converting it to the format understood by their tool of choice.

One major strength of Microsoft Access in a corporate environment is the ability to link to a variety of database formats on the workstation, on database servers, or on host computers. A manager trying to solve a problem no longer has to figure out how to get copies of data from several different sources to plug into a spreadsheet-based graph for analysis. Using Access, the manager can connect directly to the source data, build a query to extract the necessary information, and create a report with an embedded graph—all with one tool. The ability to retrieve data from multiple sources, combined with ease of use, makes Access a powerful tool for creating information processing systems.

2.17. Workgroup Applications

Large corporations find Access especially well suited for creating the workstation portion of client-server applications. Unlike many other Windows-based client application development systems, Access uses its knowledge of the application data and structure to simplify the creation of forms and reports. Applications developed using Access can be made available to users at all levels of the corporation. And with Access

it's easy to design truly "user-friendly" applications that fully utilize the investment in employee workstations. Because Access can link to and share data in many different database formats, it's ideal for creating workgroup applications that maintain data on local departmental servers but need to periodically tap data from applications in other departments or upload data to corporate servers. For smaller workgroup applications, local data can be stored and shared across the workgroup using native Access database files. For larger applications, a true database server, such as SQL Server, can be used to store the data, with Access as the workstation client. When data must be shared with other workgroups or corporate servers, the Access-based application can use the ODBC standard to execute queries that read or update data that is stored in any of several database formats.

2.18. Information Processing Systems

Perhaps a more common use for Access in a corporate environment is as the front-end tool for information processing systems. Many corporations create Executive Information System (EIS) applications using Access so that knowledgeable executives can create their own "drill down" queries, graphs, and reports. MIS departments find that Access is also a great tool for creating the end-user interface for information processing applications. In recent years, Ray McCann's business has grown by leaps and bounds. He now keeps track of dozens of entertainment groups and more than 100 nightclubs. Although he doesn't run a large corporation, his information processing needs have grown to the point that he often needs sophisticated search tools to find the perfect match between entertainment groups and clubs. The Entertainment Schedule sample database now contains the sort of "drill down" search mechanism that you might implement in a corporate information processing system. The initial search screen generates a filtered list of groups (the Group Search Summary) when more than a few groups meet the criteria entered. A double-click on a specific group in the filtered list lets Ray display the full details for that group.

2.18.1. A Personal RDBMS

Last, but certainly not least, Microsoft Access is a great tool for managing personal information on your home computer. If you're one of the millions of PC users who has a home computer system that can run Microsoft Windows, you can use Access to help

you be more productive. You might want to build a database application to manage your investment portfolio. You can create a directory containing the addresses, birthdays, and anniversaries of all your friends. If you like to cook, a recipe database could be useful. Perhaps you'd like to keep track of your collection of movies or books. I have a friend who uses Access to keep track of his athletic training. The Database Wizard in Access shows you how to quickly build and customize an assortment of personal databases. When one of our daughters got married a couple of years ago, I created a small Access application to keep track of the wedding guest list. This was one of the very first Access databases that I created, and it's automated completely with macros.

CHAPTER 3

LIBRARY AND BOOK LOANING PROGRAM

3.1. Introduction About Program

The aim of this project was to create a program that could be used for automation circulation desk of any library. The reason behind this is to make the operation accurate, organized and faster which result in saving time and efforts. so I found that using MS_access will be useful to get this aim, this program will be easy to use by any employee, in the program all form are a pop up form that mean the is just one window will be shown at time, the program have tables that will be the base of any data the user will store or looking for it, the program help the user by showing him how to do the work of the system such as how to add new books and how the system used in searching for books or loaning book , the program have the enough security to be safety by using password for each employee and whit out this password the program will not be allowed.

3.2. Program Tables

Her is the tables and the data base that the program will use during doing the work that which designed to do it.

3.2.1. Password Table

This table have the employee ID and the employee password which will be use to access to the program, the employee ID is the primary key for this table which mean that this is no employee can have the same ID number.

	Field Name	Data Type	
Key	EmpID	Text	primary key
	password	Text	employee ID

Field Properties

Figure3.1 Passwords table design

3.2.3. Book Table

This table has the field need to store data about books such as book name, book author name, book subject, and book serial number. Give every book a serial number Is the main point in to store and search for this book.

BOOK : Table			
	Field Name	Data Type	
?	BOOKNAME	Text	BOOK NAME
	BOOKSERIALNO	Number	BOOK SERIAL NO
	AUTHORNAME	Text	AUTHOR NAME
▶	BOOKSUBJECT	Text	BOOK SUBJECT

Figure3.5 Book table design

BOOK : Table				
BOOKSUBJECT	AUTHORNA	BOOKSERIALNO	BOOKNAME	
ms_access programming	rick dobson	125662	ms_acces	+
network design	john vescaes	123663	network	+
				*

Figure3.6 Book table

3.2.4. Student Table

This table use to store information about the student whom borrow book from the library this table is related to the book table by the book serial number.

Table3 : Table			
	Field Name	Data Type	
	name	Text	name of student
	lastname	Text	
?	idno	Number	ID. number
▶	department	Text	
	class	Number	
?	serialno	Number	serial number

Figure3.7 Student table design

serialno	class	department	idno	lastname	name
125555	2	cis	980334	fahmy	kamal
124525	4	com	980621	hammad	ahmed
124881	3	ee	990366	ashor	hani
0	0		0		*

Figure3.8 Student table

3.2.5. Date Table

This table store the issue date and the return back book date this table is related to the Student information table by using student ID number field.

Field Name	Data Type
idno	Number
bookname	Text
issuedate	Date/Time
returndate	Date/Time

Figure3.9 Date table design

returndate	issuedate	bookname	idno
11/10/02	10/10/02	network	980621
10/05/02	05/05/02	ms_access	980256
			0 *

Figure3.10 Date table

3.2.6 Relationships

Here is the relationship form, which control relation between the tables

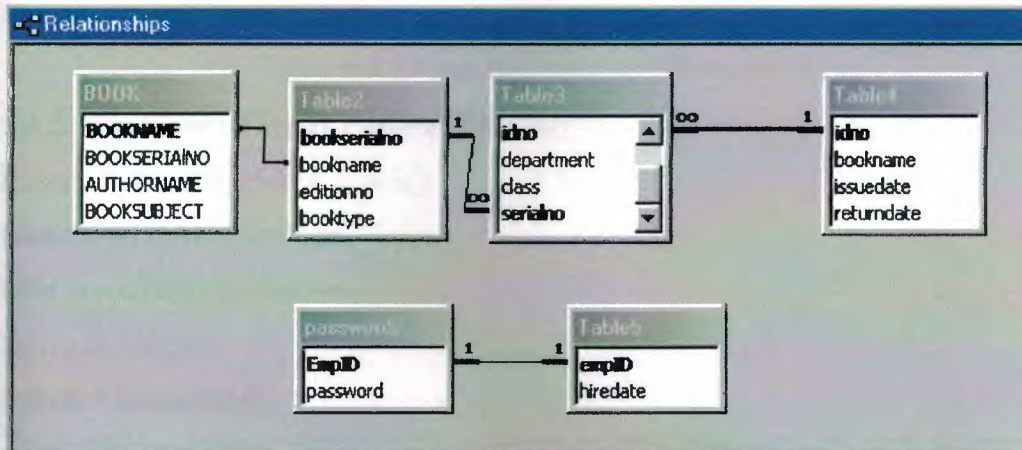


Figure3.11 Tables relation ship

3.3. Entering to Program

3.3.1. Input Password Form

The first from will be show be running the program is the password form which will ask for the employee ID. And employee password if both are conformed then the employee can use the program. A user enter into both text boxes on form and click let me in if the password matches the saved in the employee ID field the application presents a welcome message box. If it does not match the stored password for an employee ID, the user can try again or change the password.

The form titled 'frmInputPassword : Form' contains the following elements:

- Label: EMPLOYEE ID : followed by a text input box.
- Label: PASSWORD: followed by a text input box.
- Three buttons at the bottom: LET ME IN, CHANGE, and EXIT.

Figure3.12 Password form

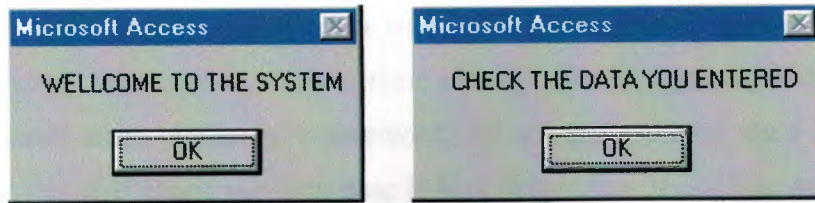


Figure3.13 message boxes of Password form

3.3.2. The Code of Input Password Form

```

Private Sub cmdLetMeIn_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "SELECT * FROM passwords WHERE EmpID='" & Me.txtEmpID & "'"
Set rs = db.OpenRecordset(s)
If rs.EOF And rs.BOF Then
    MsgBox " CHECK THE DATA YOU ENTERED "
Else
    If rs.Fields(1).Value = Me.txtPassword Then
        MsgBox "WELLCOME TO THE SYSTEM"
    End If
End If
End Sub

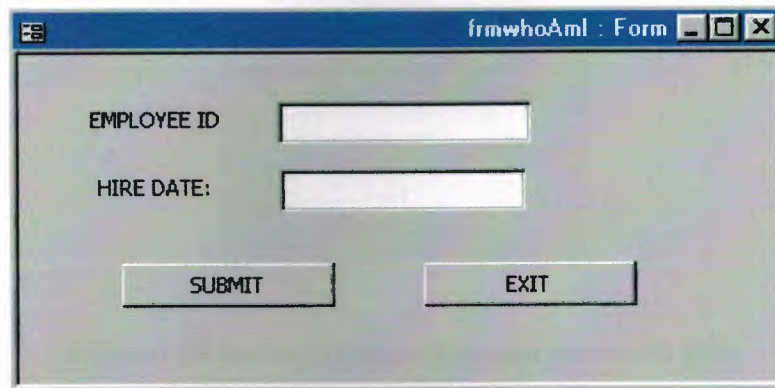
Private Sub cmdNewPassword_Click()
DoCmd.OpenForm "frmWhoAmI"
Forms("frmWhoAmI").txtEmpID = Me.txtEmpID
DoCmd.Close acForm, "frmInputPassword"
End Sub

Private Sub cmdExit_Click()
DoCmd.Close
End Sub

```

3.3.3. Change Password Form

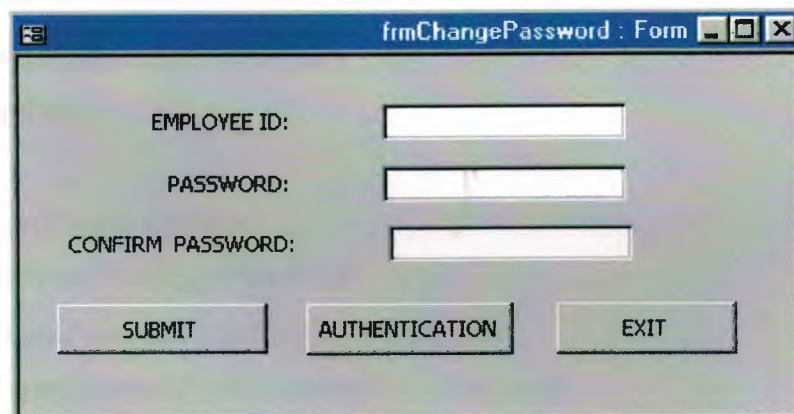
This form is appears when a user want to change the password for employee ID. This form merely asks user to confirm their identity. The system requires this confirm before it permits users to change a password. All users do is enter their hire date and click submit this will check the data base if it is if this date is confirm and then allow user to change the password



The screenshot shows a Windows-style window titled "frmwhoAml : Form". Inside the window, there are two text input fields. The first is labeled "EMPLOYEE ID" and the second is labeled "HIRE DATE:". Below these fields, there are two buttons: "SUBMIT" on the left and "EXIT" on the right.

Figure3.14 Who am i form

After the user confirms the hire date the change password will appear. The user have to enter all data needed from him to change password.



The screenshot shows a Windows-style window titled "frmChangePassword : Form". Inside the window, there are three text input fields. The first is labeled "EMPLOYEE ID:", the second is labeled "PASSWORD:", and the third is labeled "CONFIRM PASSWORD:". Below these fields, there are three buttons: "SUBMIT" on the left, "AUTHENTICATION" in the middle, and "EXIT" on the right.

Figure3.15 change password form

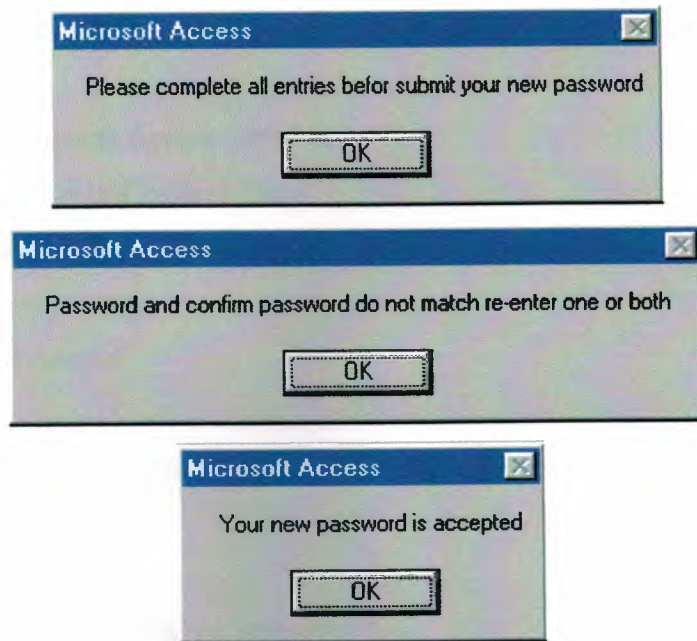


Figure3.16 message boxes of change password form

3.3.4. The Code of Change Password

```
Private Sub cmdExit_Click()
DoCmd.Close acForm, "frmwhoamI"
End Sub

Private Sub cmdSubmit_Click()
Dim ProcessMe As New myTestClass3
ProcessMe.whoAmI.CLng (txtEmpID), _
CDate(txtHireDate)
End Sub

Private Sub cmdLogin_Click()
DoCmd.OpenForm "frmInputPassword"
Forms("frmInputPassword").txtEmpID = txtEmpID
Forms("frmInputPassword").txtPassword = txtPassword
DoCmd.Close acForm, "frmChangePassword"
End Sub

Private Sub cmdSubmuit_Click()
Dim updatePW As New myTestClass3
If Me.filledCheck = False Then
```

```

MsgBox "Pleas complete all entries befor submitting your new password.",
vbInformation, _
"Programming Microsoft Access 2000"
ElseIf txtpasswprd <> txtConfirm Then
MsgBox "Password and Confirm Password donot math.Re-enter one or both.",
vbInformation, _
"Programming in Access"
Else
updatePW.NewPW txtID, txtPassword
End If
End Sub

Sub cpw(empid As Long, PW As String)
Dim cmd1 As command
Dim strSQL As String
Dim prm1 As ADODB.Parameter
Dim rst1 As ADODB.Recordset
Set cmd1 = New ADODB.command
cmd1.ActiveConnection = CurrentProject.Connection
strSQL = "parameters Secret Long;" & _
"select EmployeeID,Password from passwords where EmployeeID=secret"
cmd1.CommandText = strSQL
cmd1.CommandType = adCmdText
Set prm1 = cmd1.CreateParameter("secret", adInteger, adParamInput)
prm1.Value = empid
cmd1.Parameters.Append prm1
cmd1.Execute
Set rst1 = New ADODB.Recordset
rst1.Open cmd1
If rst1.Fields("Password") = PW Then
MsgBox "Welcome on in.", vbInformation, _
"programming Microsoft access 2000"
Else
MsgBox " check the data you entered.", vbCritical, _
"Programming Microsoft Access 2000"

```


End If

End Sub

3.4. The Main Form

This form divide the program for two part the first which deal whit the consumer and student, which showing by the command loaning, and the second which use to control the book store of the library.

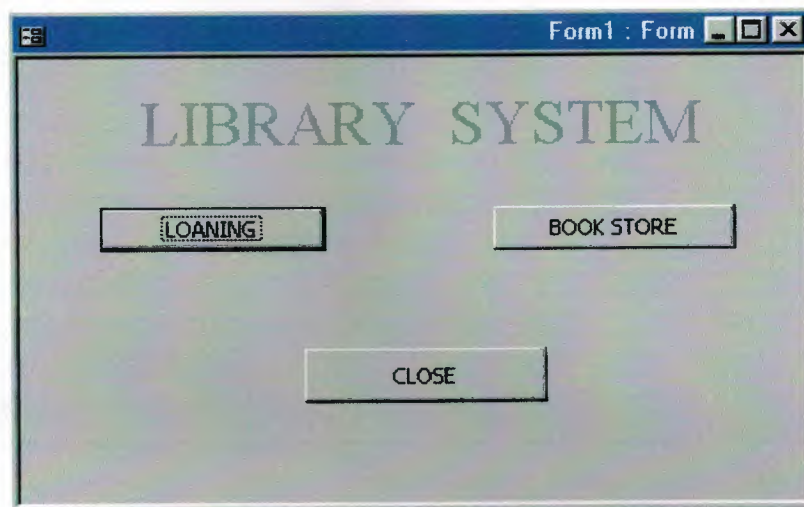


Figure3.17 main form

3.4.1. The Code of Main Form

```
Private Sub Command0_Click()
    DoCmd.OpenForm "FORM2"
    DoCmd.Close acForm, "form1"
End Sub

Private Sub Command1_Click()
    DoCmd.OpenForm "FORM3"
    DoCmd.Close acForm, "form1"
End Sub

Private Sub Command2_Click()
    DoCmd.Close acForm, "form1"
End Sub
```

3.4.2. Book Loaning Form

This form has the loaning operation control command, here in this form the user can Loaning a new book by press the new loaning command or return the book back to

the library by pressing return, and also the employee can change in the information or student loaded book by using edit information command, by pressing the back command this form will close and the main form will appear, there is just one form will be appear at time when any for is open it will close all other forms

The image shows a screenshot of a Windows-style application window titled "FORM2 : Form". The main content area has a light blue background with the text "BOOK LOANING" in a large, bold, serif font. Below the title, there are four rectangular buttons stacked vertically. The top button is labeled "LOANING NEW BOOK", the second "RETURN BOOK", the third "EDIT INFORMATION", and the bottom button "BACK". All buttons have a thin black border and a light gray fill.

Figure3.18 Book loaning form

3.4.3. The Code of Book Loaning

Option Compare Database

Private Sub command_Click()

DoCmd.OpenForm "form1"

DoCmd.Close acForm, "form2"

End Sub

Private Sub Command0_Click()

DoCmd.Close acForm, "form2"

End Sub

Private Sub Command1_Click()

DoCmd.OpenForm "form4"

DoCmd.Close acForm, "form2"

End Sub

Private Sub Command2_Click()

DoCmd.OpenForm "form5"

```
DoCmd.Close acForm, "form2"
```

```
End Sub
```

```
Private Sub Command3_Click()
```

```
DoCmd.OpenForm "form10"
```

```
DoCmd.Close acForm, "form2"
```

```
End Sub
```

3.4.4. Student Information Form

After pressing loaning new book form the student information form will be open this is the form, that have the information about the student whom loaning a book from library, the aim of this form to store all the data about the loaner and relate it book the data base, after entering the information and pressing add command the student information will be added to the student information table.

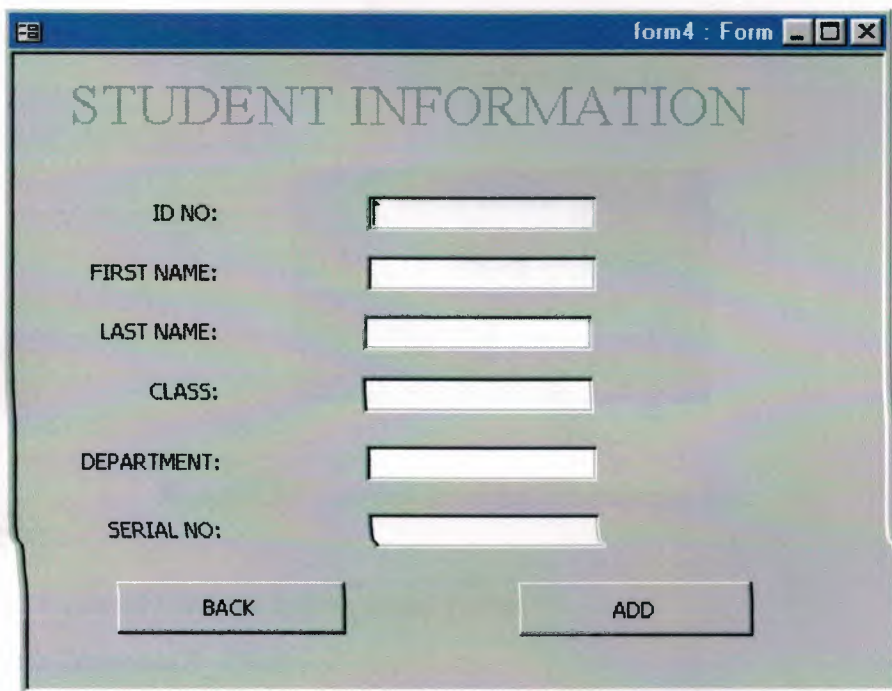
The image shows a screenshot of a Microsoft Access form titled "form4 : Form". The form has a light green background and a title bar. At the top, the text "STUDENT INFORMATION" is displayed in a large, serif font. Below this, there are six text input fields arranged vertically, each preceded by a label: "ID NO:", "FIRST NAME:", "LAST NAME:", "CLASS:", "DEPARTMENT:", and "SERIAL NO:". At the bottom of the form, there are two buttons: "BACK" on the left and "ADD" on the right.

Figure3.19 Student information form

3.4.5. Loaning Information Form

After pressing add loaning information form will be appear this form is related to the student information form data her the user have to enter the student ID for the student whom loaning book and the book name and also the issue data for loaning the

book, and the return back date of the book, after entering this data by press ok command this data will be store in the data base, then the system will ask if there any other student want to loaning a new book if the yes command pressed the system will go back to the student information form.

Figure3.20 Loaning information form



Figure3.21 Loaning Information massage box

3.4.6. The Code of Student Information Form

```
Private Sub Command5_Click()
    Call addnew
    DoCmd.OpenForm "form12"
    If MsgBox("is there any anther student? ", vbExclamation + vbYesNo) = vbYes Then
        Me.txtIDNO = ""
        Me.txtFIRSTNAME = ""
        Me.txtLASTNAME = ""
    End If
End Sub
```

```

Me.txtCLASS = ""
Me.txtDEPARTMENT = ""
Me.TXTserialno = ""
Else
DoCmd.Close acForm, "form4"
End If
End Sub
Sub addnew()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "SELECT * FROM Table3 where idno=" & Me.txtIDNO
Set rs = db.OpenRecordset(s)
If rs.EOF And rs.BOF Then
rs.addnew
rs.Fields("Name").Value = Me.txtFIRSTNAME
rs.Fields("lastname").Value = Me.txtLASTNAME
rs.Fields("class").Value = Me.txtCLASS
rs.Fields("Department").Value = Me.txtDEPARTMENT
rs.Fields("Serialno").Value = Me.TXTserialno
rs.Update
MsgBox "new book had loaed "
Else
MsgBox " This record is already present "
End If
DoCmd.OpenForm "form12"
End Sub
Private Sub Form_Load()
Me.txtIDNO = ""
Me.txtFIRSTNAME = ""
Me.txtLASTNAME = ""
Me.txtCLASS = ""
Me.txtDEPARTMENT = ""

```

```
Me.TXTserialno = ""
```

```
End Sub
```

3.4.7. The Code of Loaning Information Form

```
Private Sub Command4_Click()
```

```
Dim db As DAO.Database
```

```
Dim rs As DAO.Recordset
```

```
Dim s As String
```

```
Set db = CurrentDb()
```

```
s = "SELECT * FROM table4 WHERE idno=" & Me.txtIDNO
```

```
Set rs = db.OpenRecordset(s)
```

```
    If rs.EOF And rs.BOF Then
```

```
        rs.addnew
```

```
        rs.Fields("BOOKNAME").Value = Me.txtbookname
```

```
        rs.Fields("issudate").Value = Me.txtissuedate
```

```
        rs.Fields("returndate").Value = Me.txtreturndate
```

```
        rs.Update
```

```
        rs.MoveFirst
```

```
    Else
```

```
        MsgBox " This record is already present"
```

```
    End If
```

```
End Sub
```

3.4.8. Return Book form

By pressing the return book command this form will be appear this form will have all student whom loaning books from the library information, by pressing the command clear, system will realize that the book is back and this student information will be remove from the tables.

The image shows a Windows application window titled "form5 : Form". Inside the window, the title "RETURN A BOOK" is centered at the top. Below the title, there are six input fields arranged vertically. The first field is labeled "ID NO:" and has a small dropdown arrow on its left. The other five fields are labeled "NAME:", "LAST NAME:", "DEPARTMENT:", "CLASS:", and "SERIAL NO:". At the bottom of the form, there are two buttons: "BACK" on the left and "CLEAR" on the right.

Figure3.22 return book

3.4.9. Code of Return Book form

```

Private Sub Combo8_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select*from table3 where idno=" & Me.Combo8
Set rs = db.OpenRecordset(s)
Me.txtname = rs.Fields("name").Value
Me.txtLASTNAME = rs.Fields("lastname").Value
Me.txtDEPARTMENT = rs.Fields("department").Value
Me.txtCLASS = rs.Fields("class").Value
Me.TXTserialno = rs.Fields("serialno").Value
End Sub

Private Sub Command20_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()

```

```

s = "select*from table3 where idno='" & Me.Combo8 & "'"
Set rs = db.OpenRecordset(s)
rs.MoveLast
rs.Delete
rs.MoveFirst
End Sub
Private Sub Command21_Click()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, "form5"
End Sub

```

3.4.10. Edit Student Information Form

By pressing the edit information command this form will be appear this form will have all student whom loaning books from the library information, her you can change in data of student, by pressing the command edit, system will realize that the change of student information and will be save it to the tables.

The screenshot shows a Windows-style form window titled "form10 : Form". The form itself has a light green background and a title "EDIT STUDENT INFORMATION" in large, bold, blue letters. Below the title, there are six labels on the left, each followed by an input field on the right: "ID NO:" (with a small dropdown arrow on the left of the text box), "NAME:", "LAST NAME:", "DEPARTMENT:", "CLASS:", and "SERIAL NO:". At the bottom of the form, there are two rectangular buttons: "BACK" on the left and "EDIT" on the right.

Figure3.23 Edit Student Information Form

3.4.11. The Code of Edit Student Information Form

```
Private Sub Combc_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim a As String
Set db = CurrentDb()
s = "select*from table3 where idno=" & Me.Combc & ""
Set rs = db.OpenRecordset(s)
Me.txtname = rs.Fields("name").Value
Me.txtLASTNAME = rs.Fields("lastname").Value
Me.txtDEPARTMENT = rs.Fields("department").Value
Me.txtCLASS = rs.Fields("class").Value
Me.TXTserialno = rs.Fields("serialno").Value
End Sub

Private Sub Command14_Click()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
s = "table3"
Set db = CurrentDb()
Set rs = db.OpenRecordset(s)
rs.Edit
rs.Fields("name").Value = Me.txtname
rs.Fields("lastname").Value = Me.txtLASTNAME
rs.Fields("department").Value = Me.txtDEPARTMENT
rs.Fields("class").Value = Me.txtCLASS
rs.Fields("serialno").Value = Me.TXTserialno
rs.Update
End Sub

Private Sub Command15_Click()
DoCmd.OpenForm "form2"
DoCmd.Close acForm, "form10"
End Sub
```


3.5. Book Store Form

This is the second part of the main part of the program this form have many command, such as add new book to the library, delete book from library data base, edit book information, and search for book in the store.

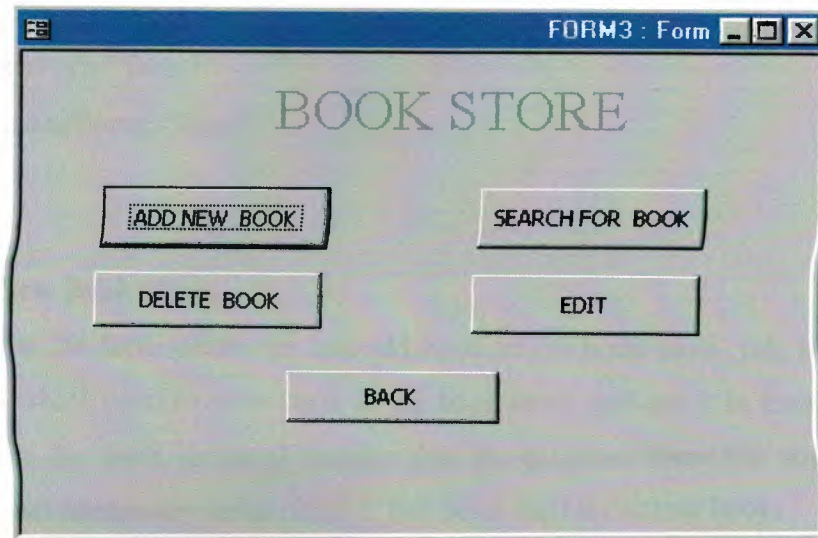


Figure3.24 Book Store Form

3.5.1. The Code of Book Store From

```
Private Sub Command0_Click()  
DoCmd.Close acForm, "form3"  
End Sub  
  
Private Sub Command1_Click()  
DoCmd.OpenForm "form6"  
DoCmd.Close acForm, "form3"  
End Sub  
  
Private Sub Command2_Click()  
DoCmd.OpenForm "form7"  
DoCmd.Close acForm, "form3"  
End Sub  
  
Private Sub Command3_Click()  
DoCmd.OpenForm "form8"  
DoCmd.Close acForm, "form3"
```

```

End Sub
Private Sub Command4_Click()
DoCmd.OpenForm "form9"
DoCmd.Close acForm, "form3"
End Sub
Private Sub Command5_Click()
DoCmd.OpenForm "form1"
DoCmd.Close acForm, "form3"
End Sub

```

3.5.2. Add New Book Form

Here is the form where we can add book to the book store, this form have the enough data which need to store book in the book store, and use it in loaning. Here the user will give the book its serial number that the program know the book by it. The system will not accept any serial number had been used for another book.

Figure3.24 Add new book form

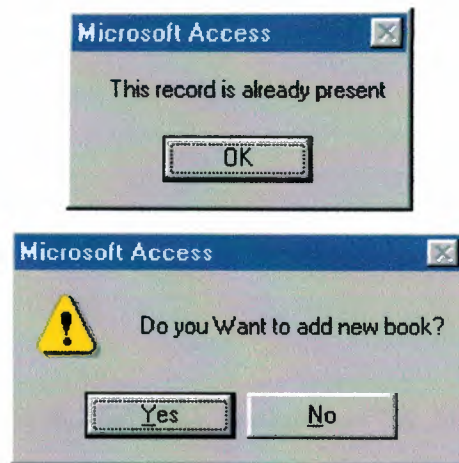


Figure3.25 Message boxes of add new book

3.5.3. The Code of Add New Book from

```

Private Sub Command8_Click()
Call add
DoCmd.OpenForm "FORM16"
If MsgBox("Do you Want to add new book? ", vbExclamation + vbYesNo) = vbYes
Then
Me.txtbookname = ""
Me.txtbookserialno = ""
Me.txtauthorname = ""
Me.txtBOOKSUBJECT = ""
Else
DoCmd.Close acForm, "form6"
End If
End Sub
Private Sub Command9_Click()
DoCmd.OpenForm "form3"
DoCmd.Close acForm, "form6"
End Sub
Private Sub Form_Load()
Me.txtbookname = ""
Me.txtbookserialno = ""
Me.txtauthorname = ""

```



```

Me.txtBOOKSUBJECT = ""
End Sub
Sub add()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "SELECT * FROM BOOK1 WHERE bookname='" & Me.txtbookname & "'"
Set rs = db.OpenRecordset(s)

If rs.EOF And rs.BOF Then
    rs.addnew
    rs.Fields("BOOKNAME").Value = Me.txtbookname
    rs.Fields("Bookserialno").Value = Me.txtbookserialno
    rs.Fields("Authorname").Value = Me.txtauthorname
    rs.Fields("Booksubject").Value = Me.txtBOOKSUBJECT
    rs.Update
    rs.MoveFirst
Else
    MsgBox " This record is already present "
End If
End Sub

```

3.5.4. Delete Book Form

In this form the user can delete book from the book store that base, her the information about the book will be remove data base, the book serial number can be used for another book

Figure3.26 Delete book form



Figure3.27 message box of delete book form

3.5.5. The Code of Delete Book Form

```

Private Sub Combs_AfterUpdate()
Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim a As String
Set db = CurrentDb()
s = "select*from book where BOOKNAME='" & Me.Combs & "'"
Set rs = db.OpenRecordset(s)
Me.txtauthname = rs.Fields("AUTHORNAME").Value
Me.txtBOOKSUBJECT = rs.Fields("booksubject").Value
Me.txtbookserialno = rs.Fields("BOOKSERIALNO").Value
End Sub
Private Sub Command0_Click()

```

```

Dim db As DAO.Database
Dim rs As DAO.Recordset
Dim s As String
Set db = CurrentDb()
s = "select*from BOOK where BOOKNAME='" & Me.Combs & "'"
Set rs = db.OpenRecordset(s)
rs.MoveLast
rs.Delete
rs.MoveFirst
End Sub
Private Sub Command9_Click()
DoCmd.OpenForm "form3"
DoCmd.Close acForm, "form7"
End Sub

```

3.5.6. Edit Book Information Form

In edit book information form the user can change any information about the book and store a new information about the book. by pressing edit the new information will be store in the table that the for related to the form.

Figure3.28 Edit Book Information Form

3.5.7. The Code of Edit Information From

```
Private Sub Combh_AfterUpdate()  
Dim db As DAO.Database  
Dim rs As DAO.Recordset  
Dim a As String  
Set db = CurrentDb()  
s = "select*from book where BOOKNAME='" & Me.Combh & "'"  
Set rs = db.OpenRecordset(s)  
Me.txtauthorname = rs.Fields("AUTHORNAME").Value  
Me.txtBOOKSUBJECT = rs.Fields("booksubject").Value  
Me.txtbookserialno = rs.Fields("BOOKSERIALNO").Value  
End Sub  
  
Private Sub Command0_Click()  
DoCmd.OpenForm "form3"  
DoCmd.Close acForm, "form9"  
End Sub  
  
Private Sub Command9_Click()  
Dim db As DAO.Database  
Dim rs As DAO.Recordset  
Dim s As String  
s = "book"  
Set db = CurrentDb()  
Set rs = db.OpenRecordset(s)  
rs.Edit  
rs.Fields("AUTHORNAME").Value = Me.txtauthorname  
rs.Fields("booksubject").Value = Me.txtBOOKSUBJECT  
rs.Fields("BOOKSERIALNO").Value = Me.txtbookserialno  
rs.Update  
MsgBox "the update has been done"  
End Sub
```

3.5.8. Search For Book From

This form is use for search for book in the book store, there are three type of searching for book, search for book by using author name, search for book by using book name, and search for book by book subject.

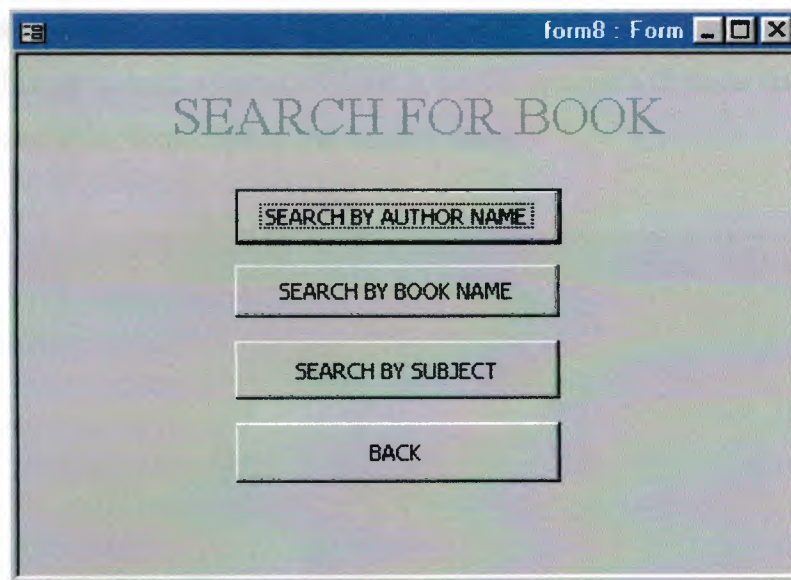


Figure3.29 Search For Book From

3.5.9. The Code of Search For Book From

```
Private Sub Command0_Click()  
DoCmd.OpenForm "form13"  
DoCmd.Close acForm, "form8"  
End Sub  
  
Private Sub Command1_Click()  
DoCmd.OpenForm "form17"  
DoCmd.Close acForm, "foRm8"  
End Sub  
  
Private Sub Command2_Click()  
DoCmd.OpenForm "form20"  
DoCmd.Close acForm, "form8"  
End Sub  
  
Private Sub Command3_Click()  
DoCmd.OpenForm "form3"
```

```
DoCmd.Close acForm, "form8"
```

```
End Sub
```

3.5.10. Search for Book by Author Name

This form is using for search for book by using book author name the user have to enter the author name and then press find command, the system will load all book for this author if there is such a name, if there is no the system will show the message box to say that there is no book such as the entered name.

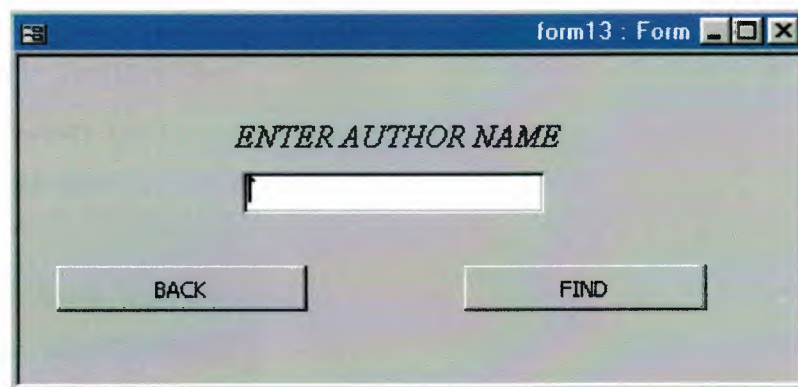


Figure3.30 Search for Book by Author Name

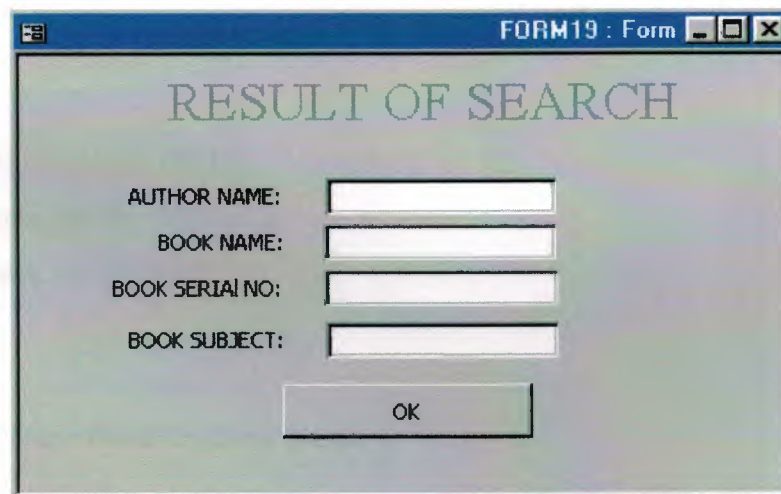


Figure3.31 Result of search form



Figure3.32 message box of Result of search form

3.5.11. The Code of Search for Book by Author Name

```

Private Sub Command3_Click()
For Each ctl In Screen.ActiveForm.Controls
If TypeOf ctl Is TextBox Then
If IsNull(ctl.Value) Then
MsgBox "enter name "
Else
DoCmd.OpenQuery "query3"
DoCmd.OpenForm "FORM19"
DoCmd.Close acQuery, "QUERY3"
Exit For
End If
End If
Next ctl
End Sub

Private Sub Command6_Click()
DoCmd.Close acForm, "FORM13"
DoCmd.OpenForm "FORM8"
End Sub

```

3.5.12. Search for Book by Book Name

This form using to search for book by using the book name when the user enter the book name the system will call the query which used for this aim.

Figure3.33 Search for book by Book Name

Figure3.34 Result of search form

Figure3.35 Message box of Search result for book by Book Name

3.5.13. The Code of Search by Book name Form

```
Private Sub Command2_Click()
```

```
For Each ctl In Screen.ActiveForm.Controls
```

```
If TypeOf ctl Is TextBox Then
```

```
If IsNull(ctl.Value) Then
```

```
MsgBox "enter name "
```

```
Else
```

```
DoCmd.OpenQuery "query2"
```

```
DoCmd.OpenForm "form18"
```

```
DoCmd.Close acQuery, "query2"
```

```
Exit For
```

```
End If
```

```
End If
```

```
Next ctl
```

```
End Sub
```

```
Private Sub Command3_Click()
```

```
DoCmd.Close acForm, "FORM17"
```

```
DoCmd.OpenForm "FORM8"
```

```
End Sub
```

3.5.14. Search by Book Subject Form

This is type of searching for book by using book subject, here the user enter the subject that he want to find a book about it, by pressing find command all book whit this subject will be list in the search result form.

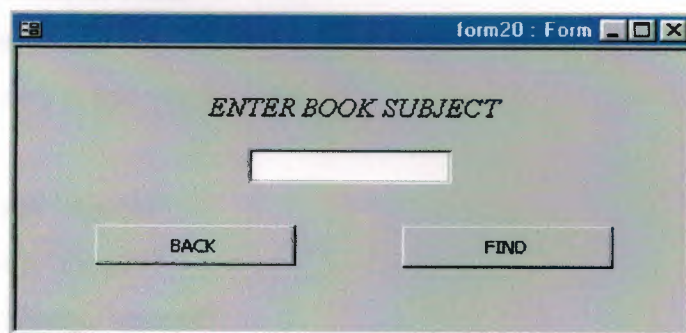
The image shows a screenshot of a Microsoft Access form window titled "form20 : Form". The form has a light blue background and a title bar with standard Windows window controls. In the center of the form, the text "ENTER BOOK SUBJECT" is displayed in a bold, italicized font. Below this text is a single-line text box for user input. At the bottom of the form, there are two rectangular buttons: one labeled "BACK" on the left and one labeled "FIND" on the right.

Figure3.36 Search by Book Subject Form

Figure3.37 Result of search form



Figure3.38 Message box of search result for book by book subject form

3.5.15. The Code of Search by Book Subject From

```
Private Sub Command2_Click()
For Each ctl In Screen.ActiveForm.Controls
If TypeOf ctl Is TextBox Then
If IsNull(ctl.Value) Then
MsgBox "enter subject that you look for it "
Else
DoCmd.OpenQuery "query4"
DoCmd.OpenForm "form21"
DoCmd.Close acQuery, "query4"
Exit For
End If
End If
```

Next ctl

End Sub

Private Sub Command3_Click()

DoCmd.Close acForm, "FORM20"

DoCmd.OpenForm "FORM8"

End Sub

CONCLUSION

In this project I learned a lot of thing that in the first time and even though not all of things I wanted to do in this project but this is mainly because of the lack of time and knowledge in programming with Microsoft access Programming. But we can say the Microsoft access database support is very extensive and complete. I have very high hopes on expanding the capability of this program in near future and from there I will take-off in mastering Microsoft access to design any project. I will try to take a lot of experience which is very important tool that I will need to take any obstacles being faced in the future.

In the graduation project the description of book, loaner registrations, loaning, loaning search are given. The structure of loaner information system is presented. Its main database modules are developed. The algorithms for loaning, loaner registration, book search, are presented. The implementation of loaning, loaner registration, book search, book loaning calculation problem in Microsoft access programming language are carried out.