NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

PHARMACY MANAGEMENT SYSTEM

Graduation Project
COM-400

Student: Niyazi Cinskızan (991702)

Supervisor: Assoc Prof. Dr Rahib Abiyev

Nicosia - 2005

ACKNOWLEDGEMENTS

First of all I would like to thank Assoc Prof. Dr Rahib Abiyev for his endless and untiring support and help and his persistence, in the course of the preparation of this project.

Under his guidance, I have overcome many difficulties that I faced during the various stages of the preparation of this project.

Finally, I would like to thank my family, especially my mother who name is Ms. Nurhan Cinskızan.Their love and guidance saw me through doubtul times. Their never ending belief in me and their encouragement has been a crucial and a very strong pillar that has held me together.

ABSTRACT

As the information age has effected every aspect of our life, the need for computerizing many information systems has raised.

Once of the important branches that are effected by information revolution is the computer programming languages.

This project is concerned about using compueter program in Pharmacy management system . It is written using Borland Delphi 6 programming language and used MySQL Database language for databases. Delphi is one easy programming languages.

This project is accomlete Pharmacy management program, that covers all services needed in most Pharmacy, such as computer related information,madicine, goods and many other Pharmacy management related services.

Before coming to this point, this project has gone through some important steps;

- First one was the requirements definition for which I had to go to some Pharmacy and study their systems.
- The second steps were designing the system and sofware that is intended to serve an integrated Pharmacy management system.
- The final steps was the implementation of the design on the computer using Delphi Language.

With this project I solved following problems;

- Medicine registration for saved data and stock count.
- Warehouse registration for buying medicine.
- Foundation registraion for selling medicine.
- Medicine selling part for medicine selling.
- Invoices part for search prescriptions.
- Account part for learn account.
- Dictionary parts for emergency.

TABLE OF CONTENTS

LIST OF TABLE

LIST OF FIGURES

INTRODUCTION

As a Pharmacy program is necessary for all pharmacies, in the project it was aimed to write a program considering the problems that we were faced till today in pharmacies. The main structure of the program was designed to apply to the medicine stock control and sales control. The program is user friendly and very simply adapted to the different stock programs with simple changes.Using the enormous advantages of Delphi program gives the chance to update this code in future due to pharmacy needs. In the following chapters the main structures and menus of the program are explained in details and finally the source code of the program is presented.

In chapter one, I summarize to pharmacy manager system information shortly.

In chapter two I brief to database desing with mysql. How to create database and how to work?.

In chapter three, I would like pripare to follow charts of pharmacy manager system.

In chapter four, I summarize to development of program modules of pharmacy manager system.

CHAPTER ONE
PHARMACY MANAGEMENT SYSTEM

1.1.Pharmacy Manager Program Main Structure

In all pharmacies need for stock program which suitable for their system and facilities of that organisation. Accordingly a common program that directly responds to registration formalities in all of these type organisations can not be prepared easily.

In this chapter it was tried to explain some of common features for pharmacy very likely adaptable and usefull to stock control.

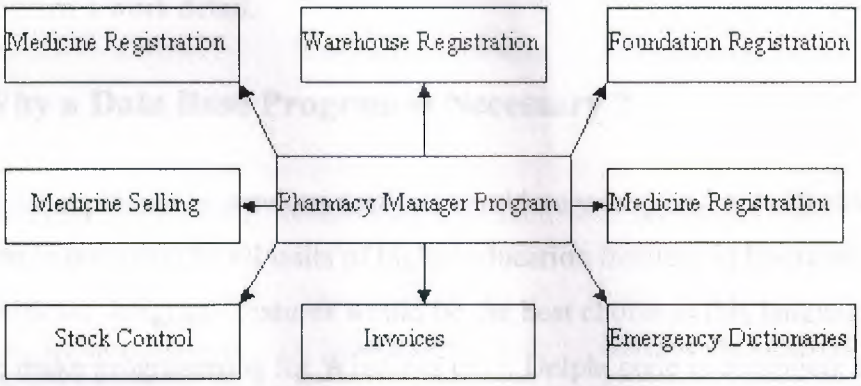


Figure1.1. Main Structures of Pharmacy Manager Program

- Medicine registration for saved data and stock count.
- Warehouse registration for buying medicine.
- Foundation registraion for selling medicine.
- Medicine selling part for medicine selling.
- Invoices part for search prescriptions.
- Account part for learn account.
- Dictionary parts for emergency.

1.2.Explanation of Main Steps in Pharmacy Manager Program

In pharmacies the first step is registeration of medicine. After the register of medicine we should buy it from warehouse. When we buy the medicine, program saves the our dapt for buy. When we want to pay our dept, program says us our dept.

When the our customer come, firstly program want his or her name, surname and prescription number. When we write these information, we can start to sell medicine which customer want. After that program automatically decrease the medicine stock and increase the our cash money.

Basicly I said the pharmacy manager program working idea. Next chapters we will see program's work detail.

1.3.Why a Data Base Program is Necessary ?

Doing all explained in previous sections would very long and not effective if a proper program is not using by all units of higher education institute in hormany. Delphi with its significant language features would be the best choise as this language is contains tolls to make programming for Windows easy. Delphi code is compiled; therefore, the compiled code runs quick. It is object oriented so objects keep the simple, organised and protected. It is very easy to read and well structured. One can actually have their code easily&efficiently proofed by a third party.

Delphi's editor works and the component system is efficient and easy to use. Delphi is single platform. While this may not be strength in some arguments, it means that (if yau are develping for windows) the tools are very mature&uniquely suited to the job at hand. Delphi is very fast. Not only is the generated code very tight, but the compiler is orders of magnitude faster then most compilers. Considering all of these advantages it can be very easy to create such a program for pharmacy .

CHAPTER TWO

DATABASE DESIGN USING BY MYSQL

2.1.Why is the computer necessary in our life

Computer software has become a driving force; it is a powerful force that set Decision-making and serves as a basis for modern investigation and problem solving. Computers have become a key factor that gives products and services that modern look, its embedded in systems of all kinds; medical, industrial, military, entertainment, even office-based products.

A Computer system in a service management record can promise better speed and efficiency with almost no change of effors.

2.2.How to develop a database application

The steps involved in database application development any relational data base application there are always the same basic steps to follow. Mysql is a relational data base management system because all data is stored in an Mysql data base in the form of simple tables. Another name for a table is relation.

The steps of Mysql database design like this

- Database design
- Tables design
- Forms design
- Query design

2.3.Relational database

DBMS(Database Management System) has established themselves as one of the primary means for data storage for information based systems ranging from large business applications to simple pc based programs. However a relational database management system (RDBMS) is the system used to work with data management operations more than 15 years, and still improving, providing more sophisticated

storage, retriaval systems.Relational database management systems provides organisations with ability to handle huge ammount of data and changing it into meaningful information.

2.4.The facilities of Mysql

Mysql is relational DBMS(Database Management System) with all the features necessary to develop and use a data base application.The facilities it offers can be found on most modern relational DBMS and all versions of Mysql.

- Tables are where all the data is stored. They are usually linked by relationships.
- Queries are the way you extract data from the database
- Forms are the method used for input and display of database data.
- Reports are used to display nicely formatted data on paper.

2.5.Delphi and Mysql

Mysql is the DBMS(Database Management System) Delphi and Mysql in developing data base applications is that for non-trivial database applications, Delphi offers more flexibility to the developper then the Delphi comes with Mysql. Mysql database using Delphi program code and setting properties.

First method of linking Delphi forms to Mysql databases called the data control. The data control is a simple Delphi control that you drag on to a Delphi form to link it to your choosen database. The data can be displayed and updated using tied text boxes, list boxes, combo boxes, and grids.

2.5.1.DAO(Data Access objects)

The DAO approach to database programming often requires more code ,but like SQL compared to the Qury Design View,offers greater control to the database programmer over what's going on his/her application.

Data Access Objects are things like databases,recordsets,table and query definitions, and fields,Rather than tying a recor set to a data control when we use DAO we shall allow our programs to create and manipulate recordsets.

2.5.2.ADO(Active X Data Objects)

The ADO programming is in principle very similar to DAO programming but cointains some new commands.ADO is Microsoft's new approach to database programming which aims to give the programmer a more consistent way of connecting to a broad range of different types of data source.

2.6.The Application of Mysql

Mysql is begin used as the development tool, and the application is going to be a single user application, which means its going to be installed on one machine, this application however may be used by more than one user on many computers sharing the same tables by using simple advancements.
After the new button, for a new database, after having specified the database name and path as above, you will be confronted with the following window.

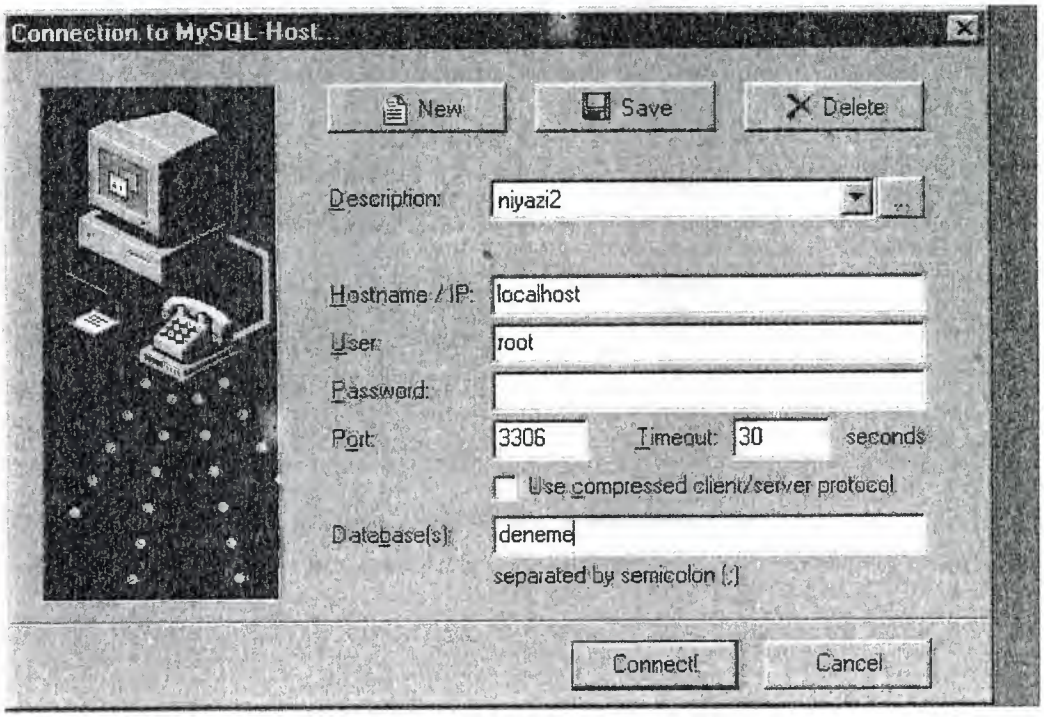


Figure2.1. The window of database

This window shows that there are notables in database yet.Click Connect button.

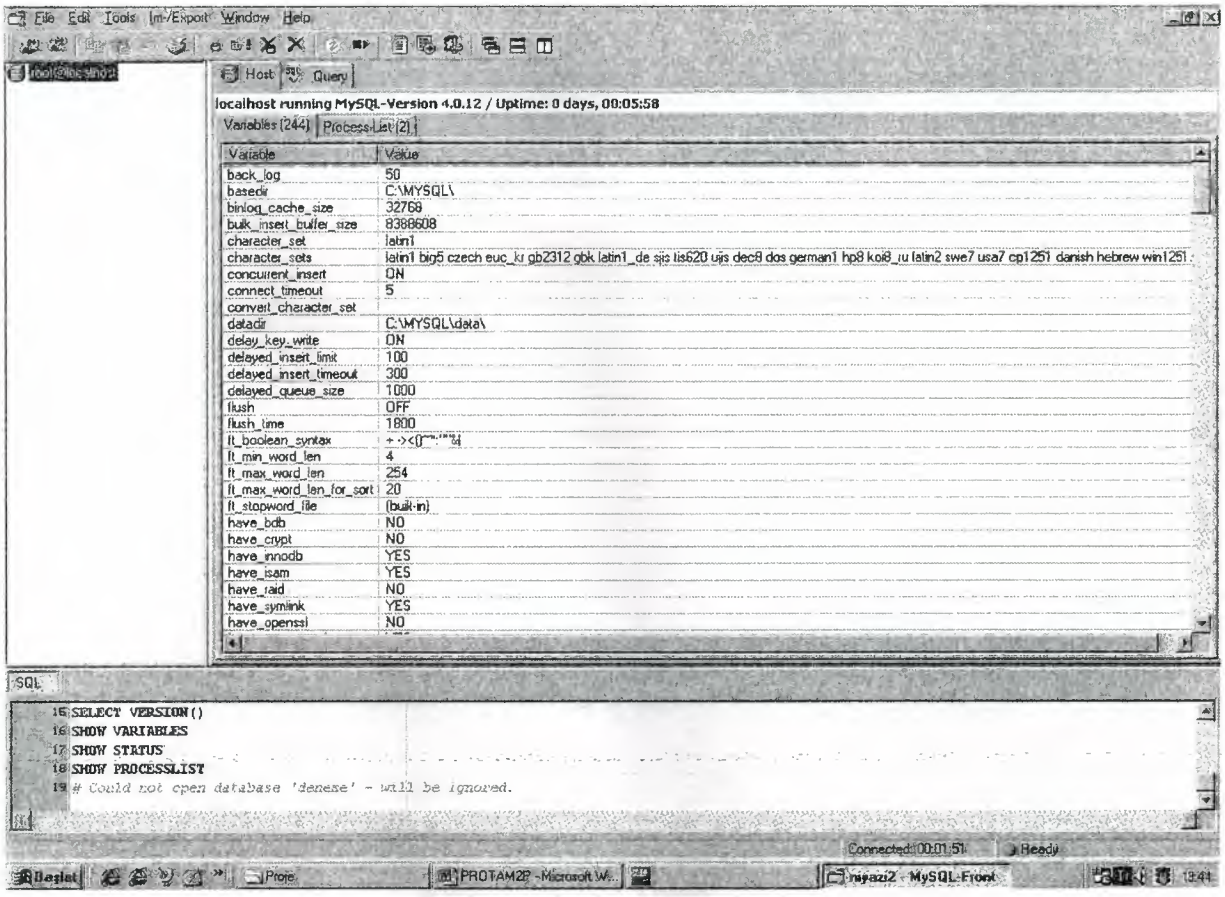


Figure 2.2. The window is type of table design

When we right click to root@localhost we see the the create database line. If we choose it, Mysql ask to us database name. After the write the name, our database being exist.

2.6.1. Tables Design

When we right click to right side, we see the create table line.

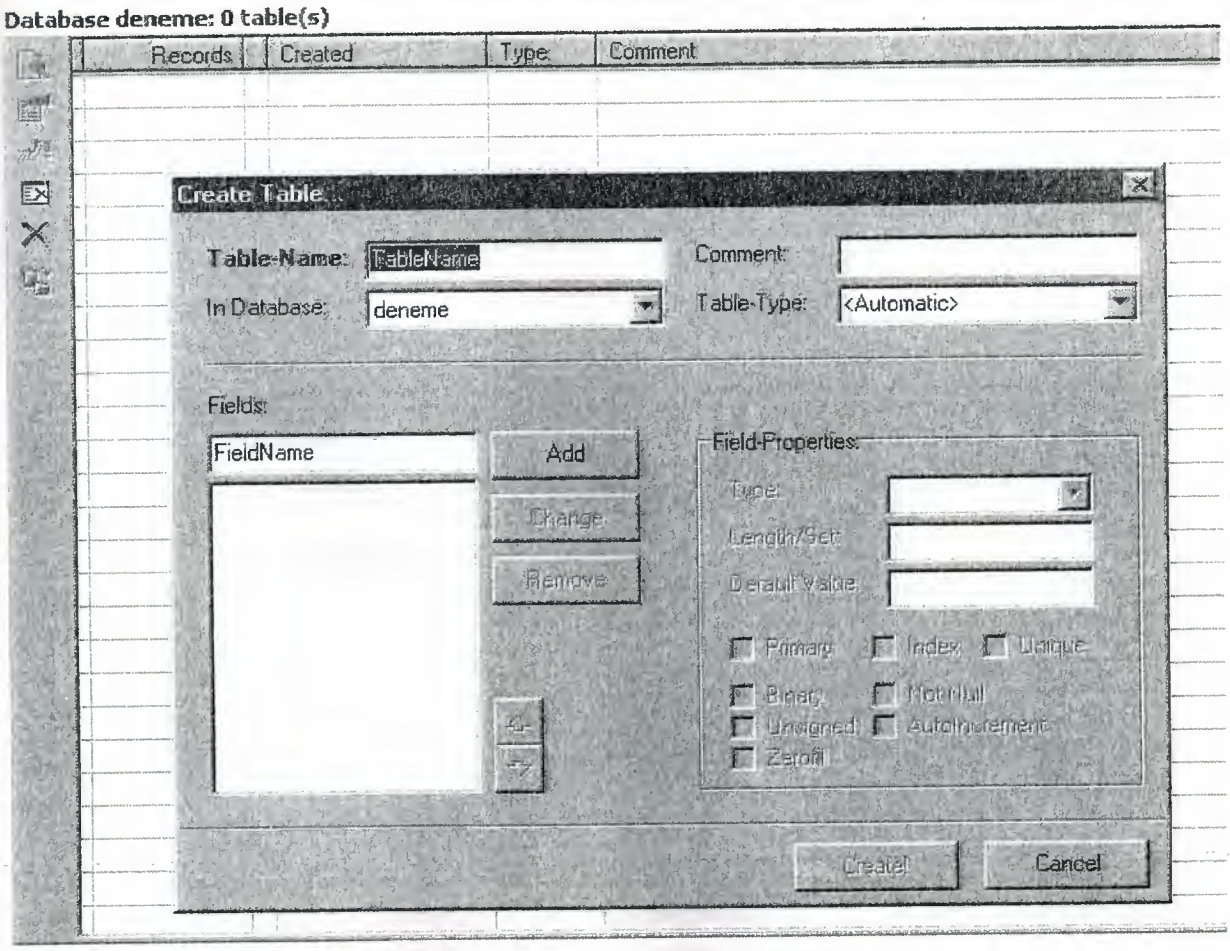


Figure 2.3 The Table

As you see, from these form, we decide the table name, fields name and fields types.

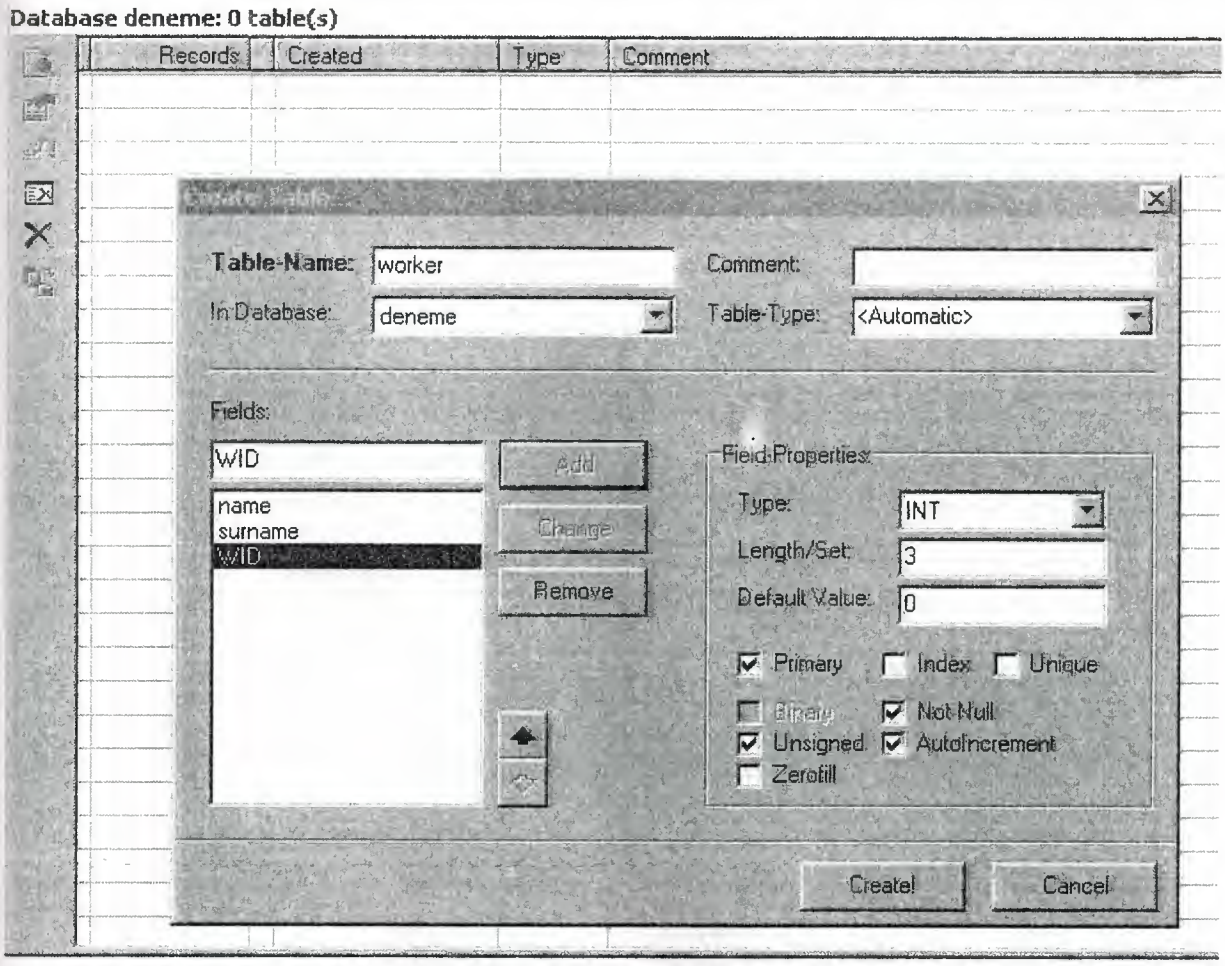


Figure 2.4 Add field to table

Finally we can see our table;

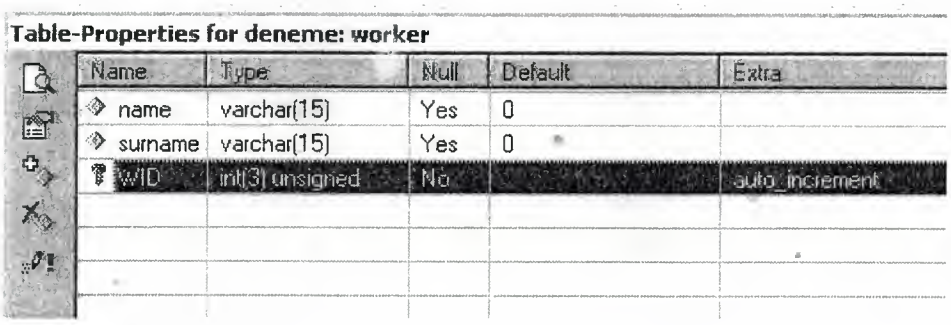


Figure 2.5 The Table

2.7. Defining Relationship Between the Tables

The structure and relation between tables are given in figure 2.6

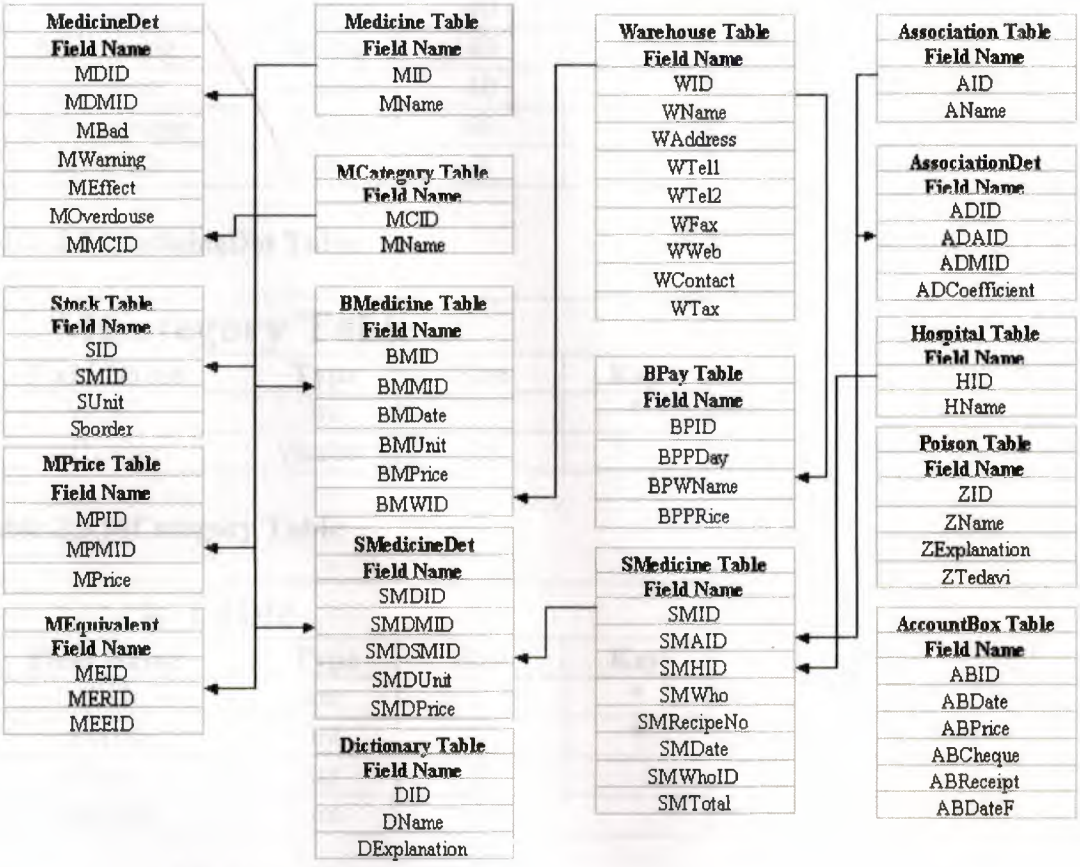


Figure 2.6 Relationship Between Tables

2.8. Database Structure

Program's database includes seventeen tables. Some tables are given below.

Medicine Table			
Field Name	**Type**	**Size**	**Key**
MID	int	5	*
MName	varchar	30	

Table 2.1 Medicine Table

MedicineDet Table			
Field Name	**Type**	**Size**	**Key**
MDID	int	5	*
MDMID	int	5	*
MBad	varchar	40	
MWarning	varchar	40	
MEffect	varchar	40	
MOverdouse	varchar	40	
MMCID	int	2	

Table 2.2 MedicineDet Table

MCategory Table			
Field Name	**Type**	**Size**	**Key**
MCID	int	2	*
MName	varchar	30	

Table 2.3 MCategory Table

Stock Table			
Field Name	**Type**	**Size**	**Key**
SID	int	5	*
SMID	int	5	*
SUnit	int	5	
Sborder	int	5	

Table 2.4 Stock Table

MPrice Table			
Field Name	**Type**	**Size**	**Key**
MPID	int	5	*
MPMID	int	5	*
MPrice	int	12	

Table 2.5 MPrice Table

BMedicine Table			
Field Name	**Type**	**Size**	**Key**
BMID	int	10	*
BMMID	int	5	*
BMDate	date		
BMUnit	int	5	
BMPrice	int	12	
BMWID	int	3	*

Table 2.6 BMedicine Table

SMedicine Table			
Field Name	**Type**	**Size**	**Key**
SMID	int	15	*
SMAID	int	2	*
SMHID	int	3	*
SMWho	varchar	40	
SMRecipeNo	varchar	15	
SMDate	Date		
SMWhoID	varchar	15	
SMTotal	int	15	

Table 2.7 SMedicine Table

SMedicineDet Table			
Field Name	**Type**	**Size**	**Key**
SMDID	int	20	*
SMDMID	int	5	*
SMDSMID	int	15	*
SMDUnit	int	5	
SMDPrice	int	12	

Table 2.8 SMedicineDet Table

Association Table			
Field Name	**Type**	**Size**	**Key**
AID	int	2	*
AName	varchar	20	

Table 2.9 Association Table

AssociationDet Table			
Field Name	**Type**	**Size**	**Key**
ADID	int	15	*
ADAID	int	2	*
ADMID	int	5	*
ADCoefficient	float		

Table 2.10 AssociationDet Table

Hospital Table			
Field Name	**Type**	**Size**	**Key**
HID	int	3	*
HName	varchar	35	

Table 2.11 Hospital Table

Warehouse Table			
Field Name	**Type**	**Size**	**Key**
WID	int	5	*
WName	varchar	35	
WAddress	varchar	70	
WTel1	int	15	
WTel2	int	15	
WFax	int	15	
WWeb	varchar	35	
WContact	varchar	35	
WTax	varchar	20	

Table 2.12 Warehouse Table

MEquivalent Table			
Field Name	**Type**	**Size**	**Key**
MEID	int	15	*
MERID	int	5	
MEEID	Tinyint	5	

Table 2.13 MEquivalent Table

Poison Table			
Field Name	**Type**	**Size**	**Key**
ZID	int	5	*
ZName	varchar	35	
ZExplanation	varchar	100	
ZTedavi	varchar	150	

Table 2.14 Poison Table

Dictionary Table			
Field Name	**Type**	**Size**	**Key**
DID	int	5	*
DName	varchar	35	
DExplanation	varchar	100	

Table 2.15 Dictionary Table

AccountBox Table			
Field Name	**Type**	**Size**	**Key**
ABID	int	15	*
ABDate	Date		
ABPrice	int	15	
ABCheque	char	1	
ABReceipt	char	1	
ABDateF	date		

Table 2.16 AccountBox Table

BPay Table			
Field Name	**Type**	**Size**	**Key**
BPID	int	10	*
BPPDay	Date		
BPWName	varchar	35	
BPPRice	int	15	

Table 2.17 BPay Table

2.9. Working with SQL

SQL standards of structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingress etc. Altough most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "Select", "Insert", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

2.9.1. Table Basics

A relational database system contains one or more objects called tables. The data or information for the database are stored in these tables. Tables are uniquely identified

by their names and are comprised of columns and rows. Columns contain the column name, data type, and an other attributes for the column. Rows contain the records or data for the columns. Here is a sample table called "weather".
City, state, high and low are the columns. The rows contain the data for this table:
Weather
City state high low
Phoenix Arizona 105 90
Tucson Arizona 101 92
Flagstaff Arizona 88 69
San Diego California 77 60
Albuquerque New Mexico 80 72

2.9.2. Selecting Data

The select statement is used to query the database and retrive selected data that match the criteria that you specify. Here is the format of a simple select statement:

Select "column1"[,"column2".etc] from "tablename"["where condition"];
[]=optional

The column names that follow the select keyword determine which columns will be returned in the results. You can select as many column names that you'd like, or you can use a "*" to select all columns.
The table name that follows the keyword from specifies the table that will be queried to retrieve the desired results.
The where clause (optinal) specifies which data values or rows will be returned or displayed, based on the criteria described after the keyword where.

2.9.3. Like

The like pattern matching operator can also be used in the conditional selection of the where clause. Like is very powerful operator that allows you to select only rows that are "like" what you specify. The percent sign "%" can be used as a wild card to match

any possible character that might appear before or after the characters specified. For example:

Select first, last, city form empinfo where first LIKE 'Er%';

This SQL statement will match any first names that start with 'Er'. Strings must be in single quotes or you can specify,

Select first, last from empinfo where last LIKE '%s';

This statement will match any last names that end in a 's'.

Select * form empinfo where first='Eric';

This will only select rows where the first name equals 'Eric' exactly.

2.9.4. Updating Records

The update statement is used to update or change records that match a specified criteria. This is accomplished by carefully constructing a where clause.

Update "tablename" set "columnname" = "newvalue" [, "nextcolumn" = "newvalue2"...] where "columnname" OPERATOR "value" [and/or "column" OPERATOR "value"];
[]=optional

Example: update phone_book set area_code=623 where prefix=979;

2.9.5. Deleting Records

The delete statement is used to delete records or rows from the table.

Delete from "tablename" where "columnname" OPERATOR "value" [and/or "column" OPERATOR "value"];

[]=optional

To delete an entire record/row from a table, enter "delete from" followed by the table name, followed by the where clause which contains the conditions to delete. If you leave off the where clause, all records will be deleted.

2.9.6. Drop a Table

The drop table command is used to delete a table and all rows in the table. The delete an entire table including all of its rows, issue the drop table command followed by the table name. Drop table is different from deleting all of the records in the table. Deleting all of the records in the table leaves the table including column and constraint information. Dropping the table removes the table defination as well as all of its rows.

Drop table "tablename";

Example: Drop table employee;

CHAPTER THREE

FLOW-CHARTS OF PROGRAM MODULES

3.1.Flow-Chart of Main Program

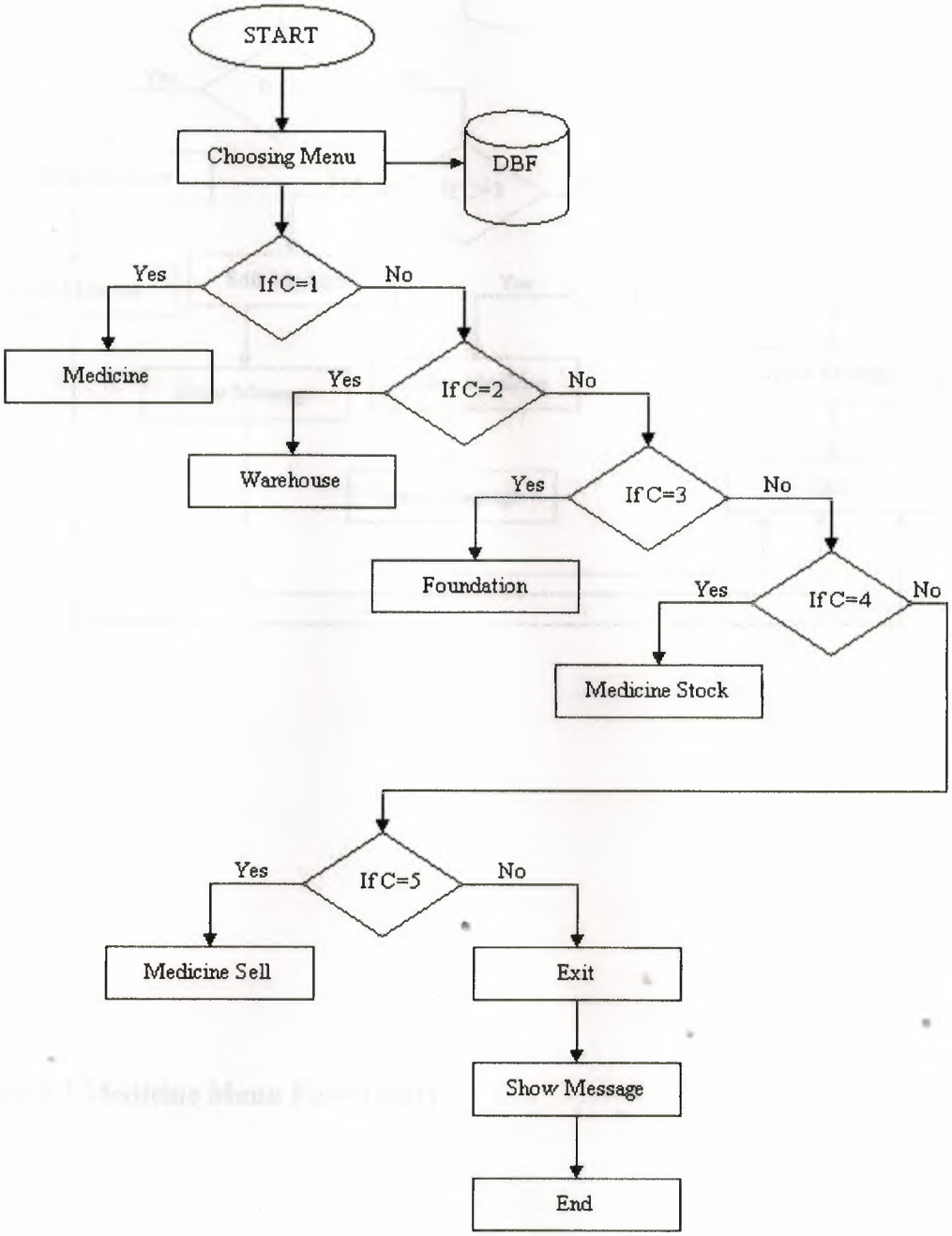


Figure 3.1 Main Menu Flow-Chart

3.2.Flow-Chart of Medicine Registration

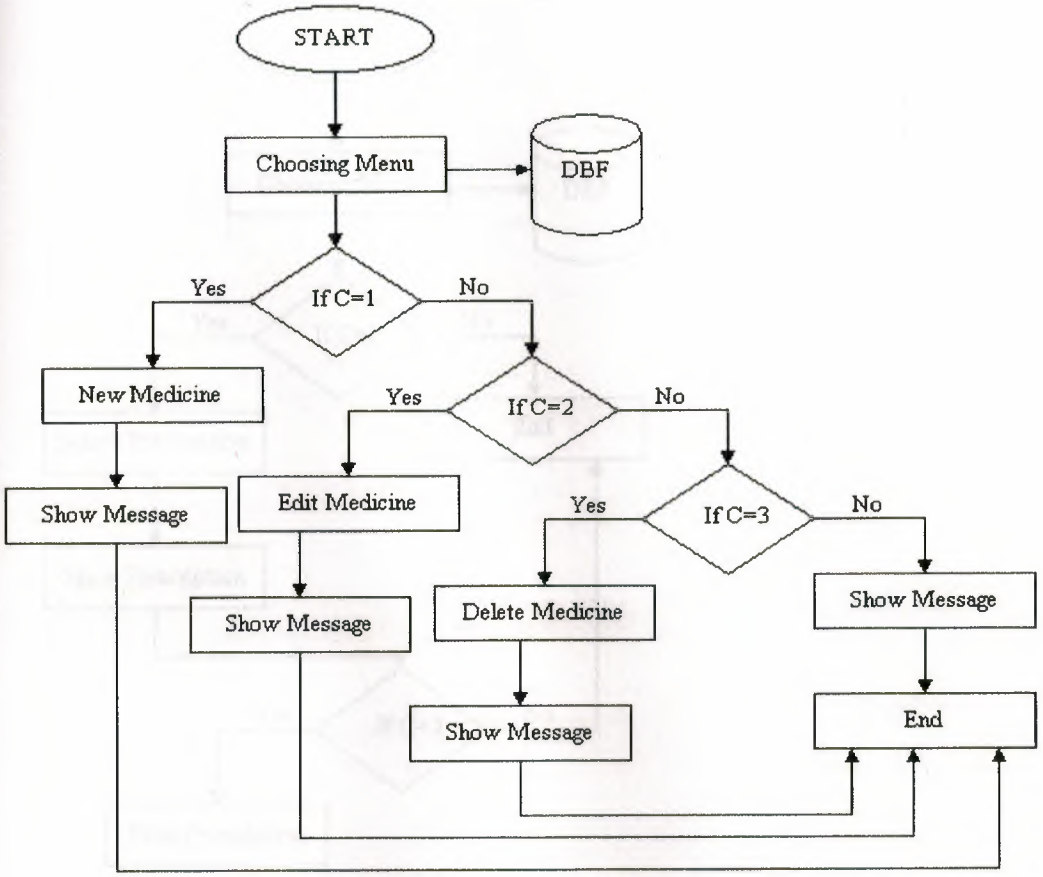


Figure 3.2 Medicine Menu Flow-Chart

3.3.Flow-Chart of Prescription Search

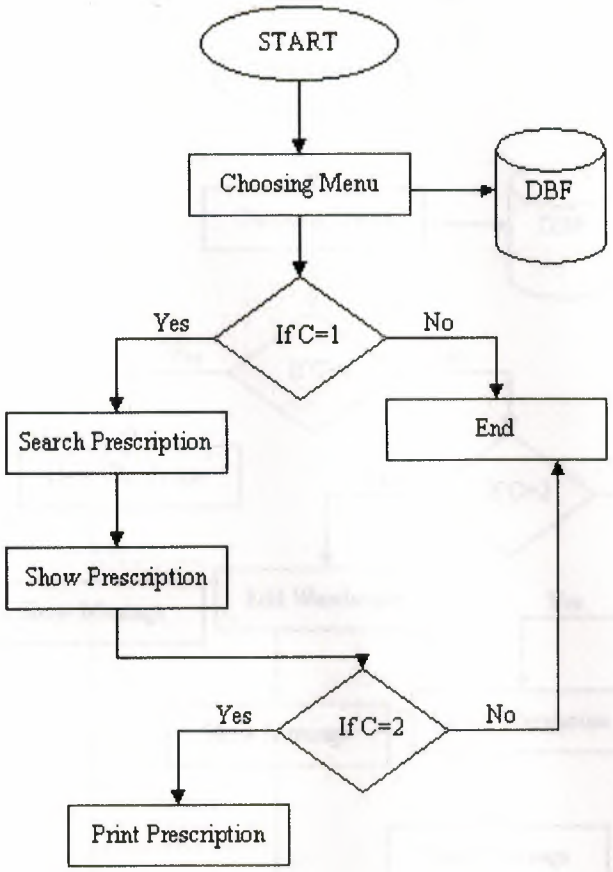


Figure 3.3 Prescription Search Flow-Chart

3.4.Flow-Chart of Warehouse Registration

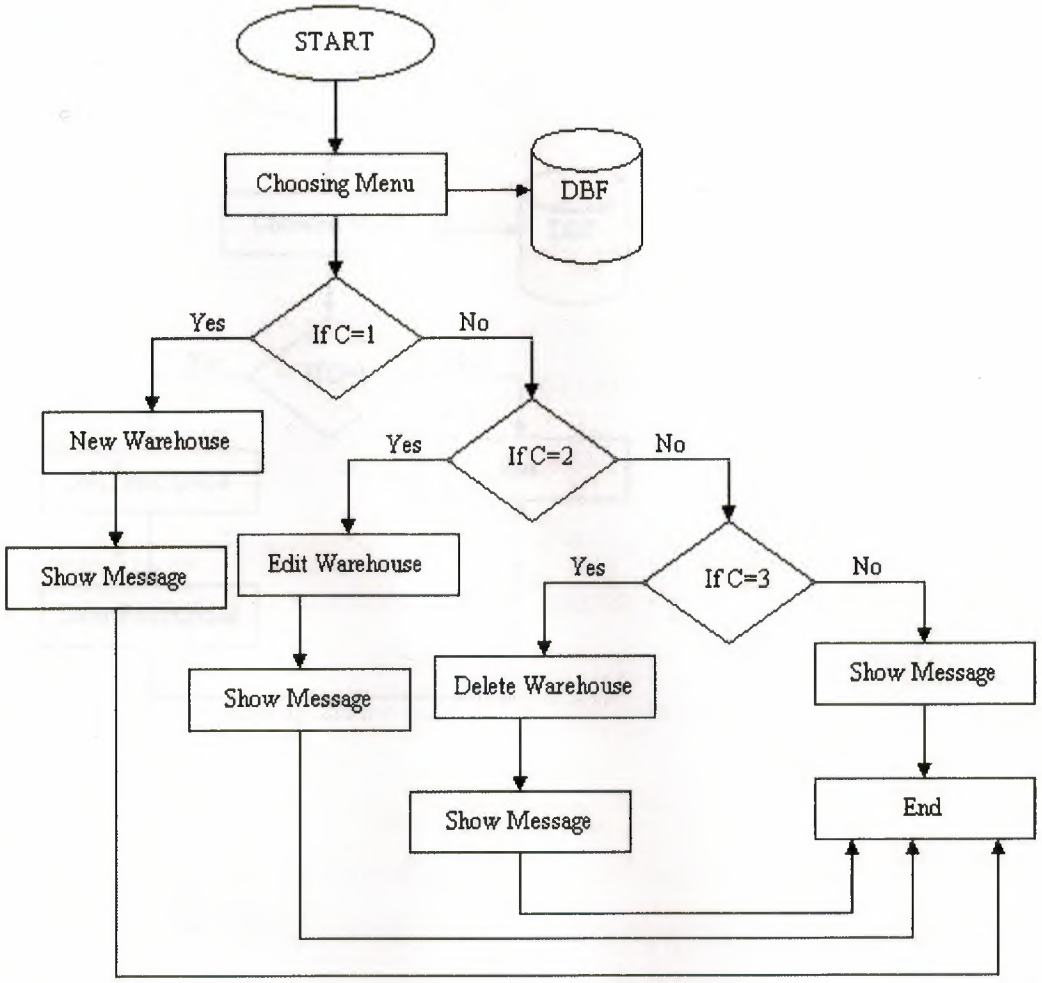


Figure 3.4 Warehouse Registration Flow-Chart

3.5.Flow-Chart of Medicine Selling

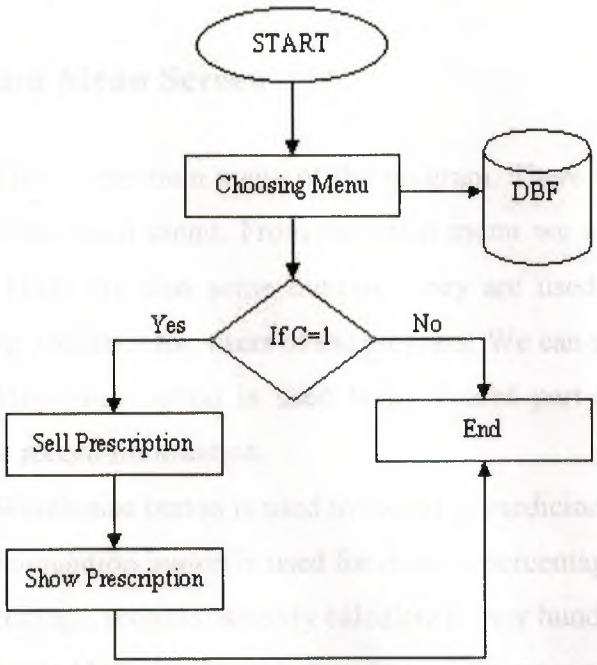


Figure 3.5 Medicine Selling Flow-Chart

CHAPTER FOUR
DEVELOPMENT OF PROGRAM MODULES OF PHARMACY PROGRAM

4.1. Main Menu Screen

This is the main menu of the program. There is also some sub menus on the top and under of the main menu. From the main menu we can go sub programs by using this sub menus. There are also some buttons. They are used to go to the sub programs. They are providing facilities for users of the program. We can see all sub programs on the main menu.

Medicines button is used to go record part of the program. In the part we enter medicine record information.

Warehouse button is used to record of medicines warehouses.

Foundation button is used for medine percentage about to foundation. If we don't give any percentage, program directly calculate it over hundred.

Enter Medicine Stock button is used for enter medicine stock.

Invoices button used for search prescriptions.

Account button is used for see account book.

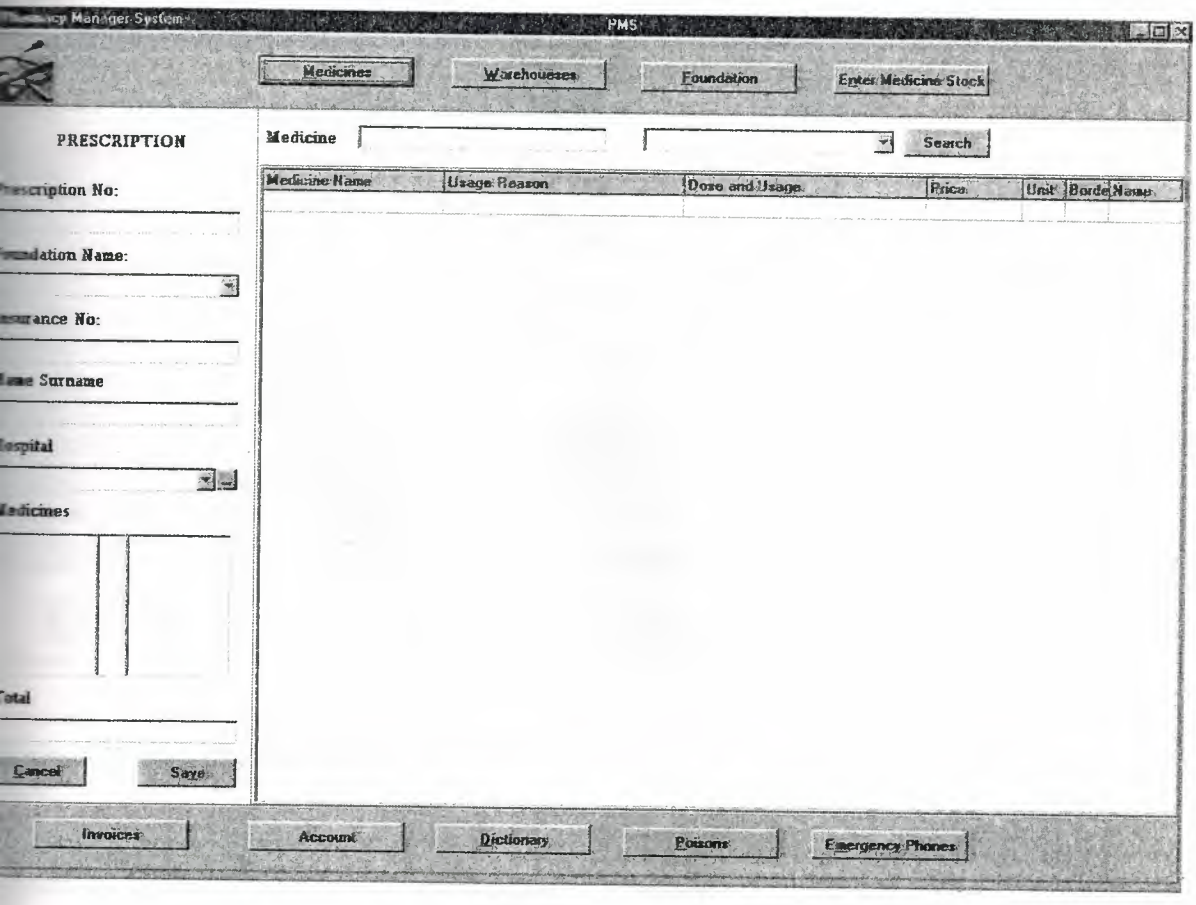
Dictionary and poison buttons are medical dictionary.

Search button used for search of database about medicines which we have in stock.

Save and Cancel buttons used for save or cancel the prescription.

Emergency Phones Button is used for emergency numbers.

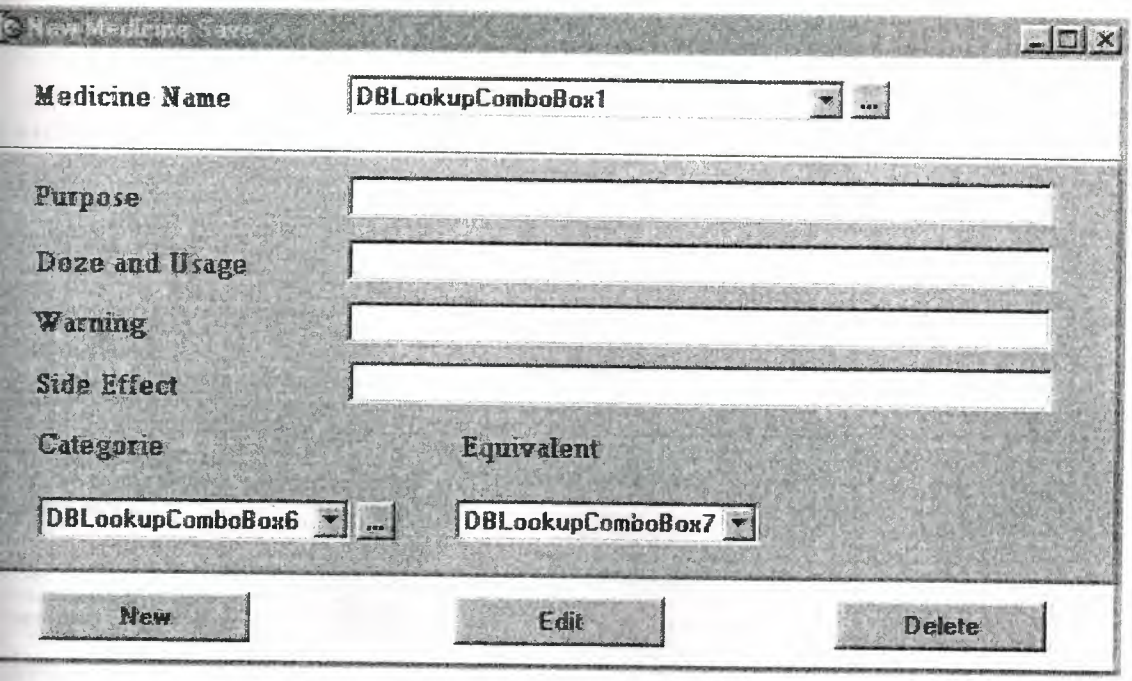
The form and codes of the main menu is following down.



gure 4.1. Main Menu

2. Record Of Medicine Screen

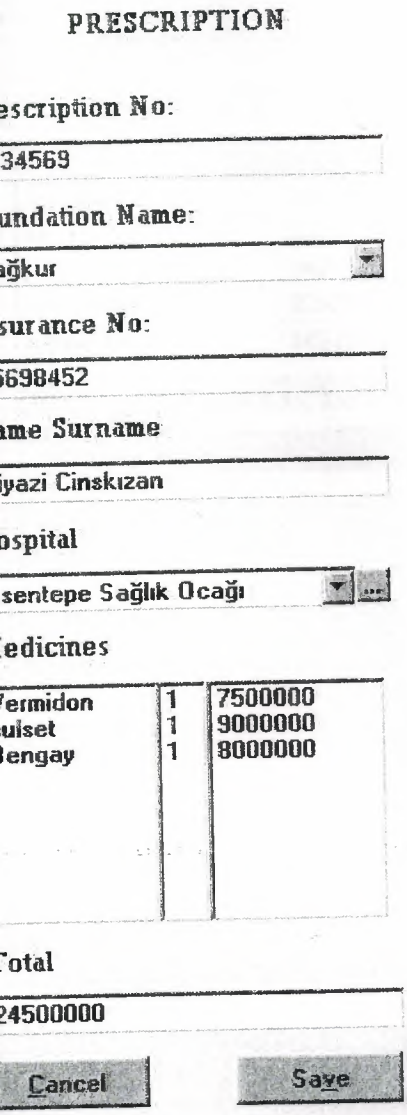
ction of showing the type of record. You can select to type of record with using record of edicine secreen .The type of record are searching, deleting, adding, finding, editining with s screen.



igure 4.2. Record of Medicine

3. Medicine Selling

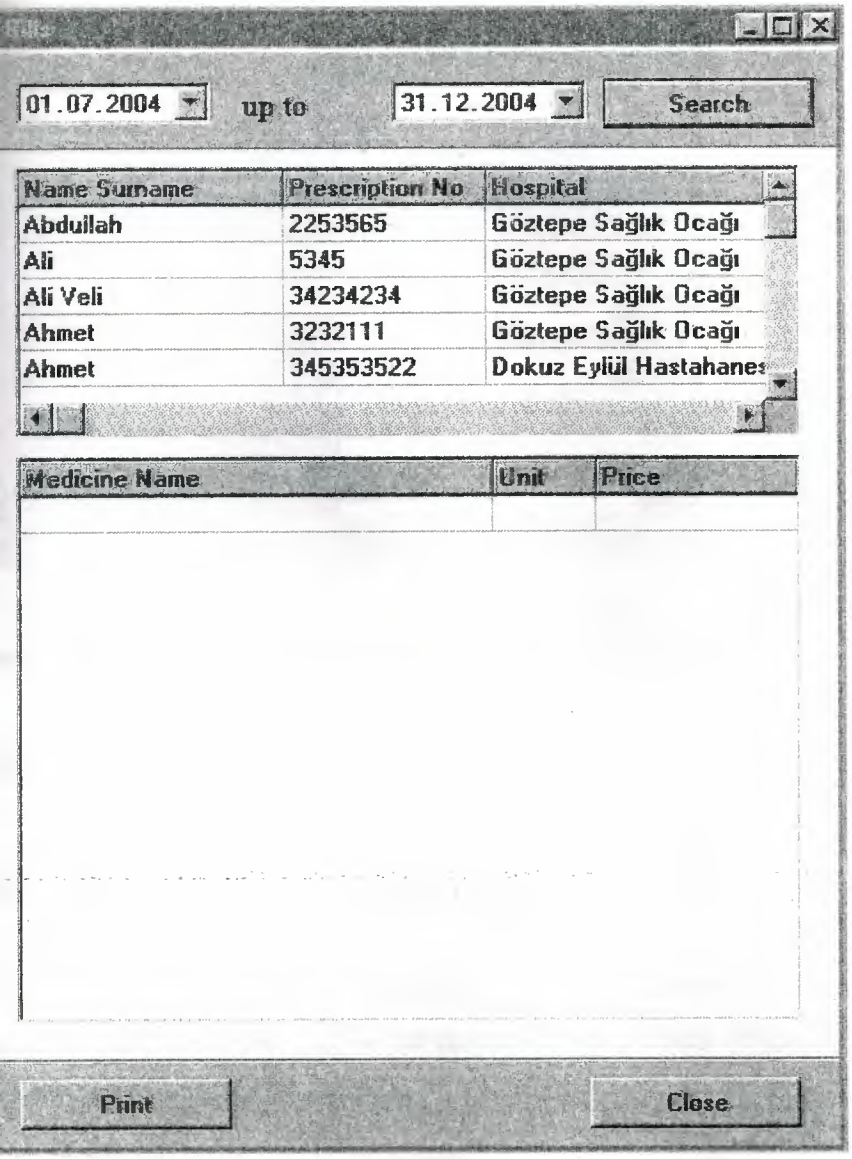
hink this screen is so important screen because of you sale your medicines from your stock. isy to use this scren. After salling the medicine the medicine will dicrease from the stock. s you know medicine selling being from main manu. Firstly we should write prescription formation. After that we can search the database for medicine which patient want. If we ve it, directly we add it to prescription. If we havent medicine on our store, we can offer the edicine which is the equal to medicine. After all of them we can sell medicines.



igure 4.3. Prescription

.4. Prescription Report Screen

he report of all sold medicine by prescription. Also we can print prescription which we want rom these screan.



gure 4.4. Prescription Search

5. Enter the Stock

r buying any medicine firstly sould we save its record to our database from medicine form. ter that we can chouse medicine, medicine stock border, buyed unit, buyed place, buyed te and buying and selling price. Also we can see medicine how many we have.

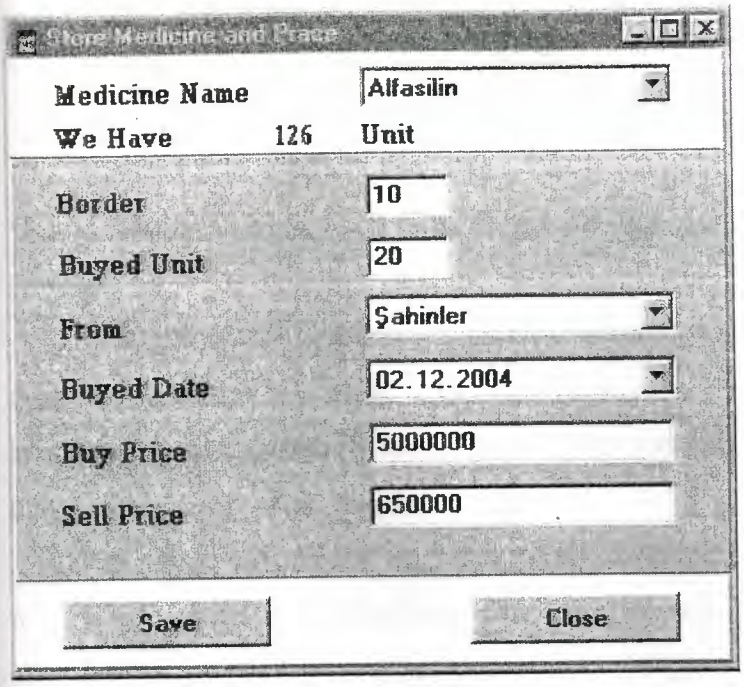


Figure 4.5. Stock Form

4.6. Record Of Warehouse Screen

With this form, we can save, edit and deleting to the warehouse record. When we need to buy any medicine, we should call to the medicine warehouses. When we buy medicine we should save warehouse name for our dept.

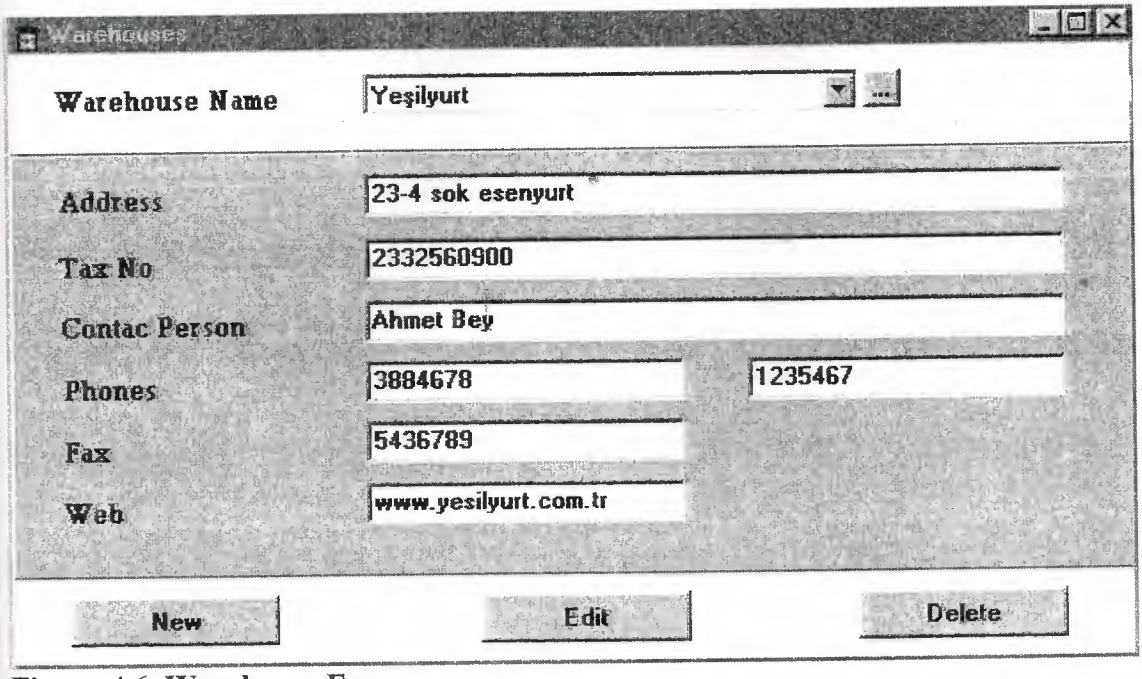


Figure 4.6. Warehouse Form

CONCLUSSION

Delphi is an easy program to grasp. This cause is why its been decided to use this program.

Delphi is a Microsoft Windows programming Language. Delphi is a distinctly different language providing powerfull features such as graphical user interfaces, even handling, access to the Win32 API, object-oriented features, error handling, structured programming, and much more.

In this project, built medicine database program. It is easy to use and it can be use most kind of drugstore. Delphi used for write this program and MySql database used for keep all databases.

In this study, the main aim to put accross is that this program can be operated by some one who has never used it before.

In this program there is also menus to makethe writting much simpler, It containing windows menus and also a facility to prepare reports.

REFERANCES

1-) Ihsan Karagülle ; Zeydin Pala(1999).Delphi 6. Istanbul. Türkmen press.

2-) Prof. Dr. Mithat Uysal (1999). Development Of The Software with Delphi 6.0. Istanbul. Beta Press.

3-) Jeff Duntemann ; Jim Mischel ; Don Taylor (1995). Delphi Programming Explorer. Coriolis Group Press.

4-)Neil Rubenking (1996). Delphi Programming Problem Solver. IDG Books Press.

5-) Hilal Pharmacy, Girne

6-) Macit Pharmacy, Lefkosa

APPENDIX

Main Form

```
unit ufrmMain;
interface
private
   { Private declarations }
 public
   Gelenilac:string;
   Satilanilac:string;
 end;
var
 Main: TMain;
implementation
uses uDBMain, uDMQuery, ufrmNMedicine, ufrmNMSave, ufrmWarehouse,
 ufrmNWarehouse, ufrmNCategory, ufrmStock, ufrmMSell, ufrmAssociation,
 ufrmAdd, ufrmBill, ufrmGeneralReports, UTools, ufrmDictionary, ufrmKasa,
 urfmPoison, ufrmPhone;
{$R *.dfm}
procedure TMain.BitBtn1Click(Sender: TObject);
begin
NMedicine.ShowModal;
end;
procedure TMain.BitBtn2Click(Sender: TObject);
begin
Warehouse.ShowModal;
end;
procedure TMain.BitBtn4Click(Sender: TObject);
begin
Stock.ShowModal;
```

```
dbgrid2.Refresh;
 With DMQuery.QSelectMedicine do
 begin
 Close;
  Sql.Clear;
  Sql.Add('Select      MPrice.*,      Stock.*,MCategory.MCName,      Medicinedet.MEffect,
Medicine.*, Medicinedet.MOverdouse from Medicinedet ');
   Sql.Add('INNER JOIN Medicine ON (Medicine.MID=Medicinedet.MDMID) ');
   Sql.Add('INNER JOIN MPrice ON (MPrice.MPMID=Medicinedet.MDMID) ');
   Sql.Add('INNER JOIN Stock ON (Stock.SMID=Medicinedet.MDMID) ');
   Sql.Add('INNER JOIN MCategory ON (MCategory.MCID=Medicinedet.MMCID) ');
   Sql.Add(' Where Medicine.MID =:Gelen order by Medicine.MName');
   ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
   Open;
  end;
    DBMain.tblMedicine.Refresh;
    DBMain.tblMedicinedet.Refresh;
    DBMain.tblMPrice.Refresh;
    DBMain.tblStock.Refresh;
    DBMain.tblMCategory.Refresh;
    dbgrid2.DataSource:=dmquery.dsQSelectMedicine;
    edit1.Text:='';
 Except
  end;
 end;
procedure TMain.BitBtn9Click(Sender: TObject);
begin
 Try
  if Trim(edit1.Text)='' then
   begin
    dbgrid2.Refresh;
    With DMQuery.QSelectMedicine do
      begin
       Close;
```

```
      Sql.Clear;
      Sql.Add('Select MPrice.*, Stock.*,MCategory.MCName, Medicine.*,
Medicinedet.MEffect, Medicinedet.MOverdouse from Medicinedet ');
      Sql.Add('INNER JOIN Medicine ON (Medicine.MID=Medicinedet.MDMID) ');
      Sql.Add('INNER JOIN MPrice ON (MPrice.MPMID=Medicinedet.MDMID) ');
      Sql.Add('INNER JOIN Stock ON (Stock.SMID=Medicinedet.MDMID) ');
      Sql.Add('INNER JOIN MCategory ON (MCategory.MCID=Medicinedet.MMCID) ');
      Sql.Add(' order by Medicine.MName ');
      Open;
     end;
     DBMain.tblMedicine.Refresh;
     DBMain.tblMedicinedet.Refresh;
     DBMain.tblMPrice.Refresh;
     DBMain.tblStock.Refresh;
     DBMain.tblMCategory.Refresh;
     dbgrid2.DataSource:=dmquery.dsQSelectMedicine;
     edit2.Text:='';
     edit3.Text:='';
     edit4.Text:='';
     edit5.Text:='';
     DBLookupComboBox1.KeyValue:=0;
     DBLookupComboBox2.KeyValue:=0;
     DBLookupComboBox3.KeyValue:=0;
  End
    Else
  begin
   dbgrid2.Refresh;
  With DMQuery.QSelectMedicine do
  begin
  Close;
   Sql.Clear;
   Sql.Add('Select Medicine.*, MPrice.*, Stock.*,MCategory.MCName,
Medicinedet.MEffect, Medicinedet.MOverdouse from Medicinedet ');
   Sql.Add('INNER JOIN Medicine ON (Medicine.MID=Medicinedet.MDMID) ');
```

```
Sql.Add('INNER JOIN MPrice ON (MPrice.MPMID=Medicinedet.MDMID) ');
Sql.Add('INNER JOIN Stock ON (Stock.SMID=Medicinedet.MDMID) ');
Sql.Add('INNER JOIN MCategory ON (MCategory.MCID=Medicinedet.MMCID) ');
Sql.Add(' Where Medicine.MName like:Gelen1 order by Medicine.MName ');
ParamByName('Gelen1').AsString := (Trim(edit1.Text) + '%');
Open;
end;
  DBMain.tblMedicine.Refresh;
  DBMain.tblMedicinedet.Refresh;
  DBMain.tblMPrice.Refresh;
  DBMain.tblStock.Refresh;
  DBMain.tblMCategory.Refresh;
  dbgrid2.DataSource:=dmquery.dsQSelectMedicine;
  edit2.Text:='';
  edit3.Text:='';
  edit4.Text:='';
  edit5.Text:='';
  DBLookupComboBox1.KeyValue:=0;
  DBLookupComboBox2.KeyValue:=0;
  DBLookupComboBox3.KeyValue:=0;
 end;
Except
 end;
end;
procedure TMain.DBGrid2DblClick(Sender: TObject);
begin
Gelenilac  :=  dbgrid2.Columns.Grid.Fields[7].AsString;
MSell.ShowModal;
end;
procedure TMain.FormActivate(Sender: TObject);
begin
DBGrid2.Refresh;
end;
procedure TMain.BitBtn3Click(Sender: TObject);
```

```
begin
Association.Showmodal;
end;
procedure TMain.Edit5Click(Sender: TObject);
var
i,a:integer;
c:real;
begin
i:=listbox1.Items.Count;
c:=0;
For a:=0 to i-1 do
begin
 With DMQuery.QSelect1 do
    begin
     Close;
     Sql.Clear;
     Sql.Add('SELECT MID From Medicine ');
     Sql.Add(' Where MName=:Gelen ');
     ParamByName('Gelen').AsString := Listbox1.Items[a];
     Open;
    end;
 With DMQuery.QSelect2 do
    begin
     Close;
     Sql.Clear;
     Sql.Add('SELECT ADCoefficient From AssociationDet ');
     Sql.Add(' Where (ADAID=:Gelen3) and (ADMID=:Gelen4) ');
     ParamByName('Gelen3').AsInteger := DBLookupComboBox2.KeyValue;
     ParamByName('Gelen4').AsInteger                                    :=
DMQuery.QSelect1.FieldByName('MID').AsInteger;
     Open;
    end;
if dmquery.QSelect2.RecordCount>0 then
 begin
```

```
c:=(strtofloat(Listbox3.Items[a])*DMQuery.QSelect2.FieldByName('ADCoefficient').asfloat)
+c;
 end
else
 begin
  c:=strtofloat(Listbox3.Items[a])+c;
 end;
end;
Edit5.Text := floattostr(c);
end;
procedure TMain.BitBtn11Click(Sender: TObject);
var i,a:integer;
begin
i:=listbox1.Items.Count;
edit2.Text:='';
edit3.Text:='';
edit4.Text:='';
edit5.Text:='';
DBLookupComboBox1.KeyValue:=0;
DBLookupComboBox2.KeyValue:=0;
DBLookupComboBox3.KeyValue:=0;
For a:=0 to i-1 do
begin
Listbox1.Items[a]:='';
Listbox2.Items[a]:='';
Listbox3.Items[a]:='';
 end;
 end;
 procedure TMain.BitBtn10Click(Sender: TObject);
 var LastID,i,a,SUnitM:integer;
 begin
 if (edit2.Text='') or (edit4.Text='') then
 begin
```

```
Application.Messagebox('Enter the Prescription No and Foundation Name!','Warning
:',mb_Ok+MB_ICONINFORMATION);
exit;
end;
if edit5.Text='' then
begin
exit;
end;
With DMQuery.Qinsert do
      Begin
       Close;
       Sql.Clear;
       Sql.Add                                    ('insert                                                    into
SMedicine(SMWho,SMAID,SMRecipeNo,SMHID,SMWhoID,SMTotal,SMDate)');
                                                                                                    ('Values
       Sql.Add
(:MWho,:MAID,:MRecipeNo,:MHospital,:MWhoID,:MTotal,:SMDate)');
       ParamByName('MWho').AsString      := Edit4.Text;
       ParamByName('MAID').AsInteger     := DBLookupComboBox2.KeyValue;
       ParamByName('MRecipeNo').AsString := Edit2.Text;
       ParamByName('MHospital').AsInteger:= DBLookupComboBox3.KeyValue;;
       ParamByName('MWhoID').AsString    := Edit3.Text;
       ParamByName('MTotal').AsInteger   := strtoint(Edit5.text);
       ParamByName('SMDate').AsDate      := (Date);
       Execsql;
      End;
  With DMQuery.QSelect do
    begin
     Close;
     Sql.Clear;
     Sql.Add('SELECT MAX(SMID) AS LastID From SMedicine ');
     Open;
     LastID := FieldByName('LastID').AsInteger;
    end;
  i:=listbox1.Items.Count;
```

```
For a:=0 to i-1 do
begin
With DMQuery.QSelect1 do
   begin
    Close;
    Sql.Clear;
    Sql.Add('SELECT MID From Medicine ');
    Sql.Add(' Where MName=:Gelen ');
    ParamByName('Gelen').AsString := Listbox1.Items[a];
    Open;
   end;
 With DMQuery.QSelect2 do
    begin
     Close;
     Sql.Clear;
     Sql.Add('SELECT SUnit From Stock ');
     Sql.Add(' Where SMID=:Gelen ');
     ParamByName('Gelen').AsInteger                                    :=
 DMQuery.QSelect1.FieldByName('MID').AsInteger;
     Open;
    end;
    SUnitM:=((DMQuery.QSelect2.FieldByName('SUnit').AsInteger)-1);
 With DMQuery.QSelect3 do
     begin
      Close;
      Sql.Clear;
      Sql.Add('SELECT ADCoefficient From AssociationDet ');
      Sql.Add(' Where (ADAID=:Gelen3) and (ADMID=:Gelen4) ');
      ParamByName('Gelen3').AsInteger := DBLookupComboBox2.KeyValue;
      ParamByName('Gelen4').AsInteger                                  :=
  DMQuery.QSelect1.FieldByName('MID').AsInteger;
      Open;
     end;
  With DMQuery.QEdit do
```

```
begin
 Close;
  Sql.Clear;
  Sql.Add('Update Stock Set SUnit=:SUnitM ');
  Sql.Add(' WHERE SMID=:Hangiilac ');
  ParamByName('SUnitM').AsInteger        := SUnitM;
                                                                          :=
  ParamByName('Hangiilac').AsInteger
DMQuery.QSelect1.FieldByName('MID').AsInteger;
   ExecSql;
   dbMain.tblStock.Refresh;
  end;
With DMQuery.Qinsert1 do
      Begin
       Close;
       Sql.Clear;
       Sql.Add ('insert into SMedicineDet(SMDMID,SMDUnit,SMDPrice,SMDSMID)');
       Sql.Add ('Values (:MDMID,:MDUnit,:MDPrice,:MDSMID)');
                                                                          :=
       ParamByName('MDMID').AsInteger
DMQuery.QSelect1.FieldByName('MID').AsInteger;
       ParamByName('MDUnit').AsInteger   := strtoint(Listbox2.Items[a]);
                                                                          :=
       ParamByName('MDPrice').AsInteger
strtoint(floattostr(strtofloat(Listbox3.Items[a])*DMQuery.QSelect3.FieldByName('ADCoeffi
cient').asfloat));
       ParamByName('MDSMID').AsInteger   := LastID;
       Execsql;
      End;
 Listbox1.Items[a]:='';
 Listbox2.Items[a]:='';
 Listbox3.Items[a]:='';
 end;
 edit2.Text:='';
 edit3.Text:='';
 edit4.Text:='';
 edit5.Text:='';
```

```
BLookupComboBox1.KeyValue:=0;
BLookupComboBox2.KeyValue:=0;
BLookupComboBox3.KeyValue:=0;
d;
ocedure TMain.ListBox1Click(Sender: TObject);
r a:integer;
gin
=ListBox1.ItemIndex;
stbox1.Items[a]:='';
stbox2.Items[a]:='';
stbox3.Items[a]:='';
d;
ocedure TMain.ListBox2Click(Sender: TObject);
r a:integer;
gin
=ListBox1.ItemIndex;
stbox1.Items[a]:='';
stbox2.Items[a]:='';
stbox3.Items[a]:='';
d;
ocedure TMain.ListBox3Click(Sender: TObject);
r a:integer;
gin
=ListBox1.ItemIndex;
stbox1.Items[a]:='';
stbox2.Items[a]:='';
stbox3.Items[a]:='';
d;
ocedure TMain.BitBtn12Click(Sender: TObject);
gin
dd.ShowModal;
d;
ocedure TMain.BitBtn13Click(Sender: TObject);
gin
```

```
Bill.ShowModal;
end;
procedure TMain.FormCloseQuery(Sender: TObject; var CanClose: Boolean);
begin
if (Application.MessageBox('Yazılımdan Çıkmak istediğine Emin misin?',
                'Uyarı:',MB_YESNO+MB_ICONQUESTION)=idYes) then
                 begin
                  Application.Terminate;
                 end
                  Else
                   begin
                     SYSutils.Abort;
                     Exit;
                   end;
end;
procedure TMain.BitBtn6Click(Sender: TObject);
begin
Dictionary.ShowModal;
end;
procedure TMain.BitBtn5Click(Sender: TObject);
begin
Kasa.ShowModal;
end;
procedure TMain.BitBtn7Click(Sender: TObject);
begin
Poison.ShowModal;
end;
procedure TMain.BitBtn8Click(Sender: TObject);
begin
 Phone.ShowModal;
end;
end.
```

NMedicine Form

```
unit ufrmNMedicine;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Buttons, DBCtrls;
type
  TNMedicine = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    Label10: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    BitBtn1: TBitBtn;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    DBLookupComboBox2: TDBLookupComboBox;
    DBLookupComboBox3: TDBLookupComboBox;
    DBLookupComboBox4: TDBLookupComboBox;
    DBLookupComboBox5: TDBLookupComboBox;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    Label9: TLabel;
    BitBtn5: TBitBtn;
    Panel5: TPanel;
    Label12: TLabel;
```

```
  Label15: TLabel;
  Label16: TLabel;
  Label17: TLabel;
  Edit5: TEdit;
  Edit6: TEdit;
  Edit7: TEdit;
  Edit8: TEdit;
  DBLookupComboBox6: TDBLookupComboBox;
  DBLookupComboBox7: TDBLookupComboBox;
  BitBtn6: TBitBtn;
  Label11: TLabel;
  Label20: TLabel;
  procedure BitBtn1Click(Sender: TObject);
  procedure BitBtn2Click(Sender: TObject);
  procedure DBLookupComboBox1Click(Sender: TObject);
  procedure BitBtn3Click(Sender: TObject);
  procedure BitBtn4Click(Sender: TObject);
  procedure BitBtn6Click(Sender: TObject);
  procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
 private
  { Private declarations }
 public
  { Public declarations }
 end;
var
 NMedicine: TNMedicine;
implementation
uses uDBMain, uDMQuery, ufrmNMSave, ufrmNCategory;
{$R *.dfm}
procedure TNMedicine.BitBtn1Click(Sender: TObject);
begin
NMSave.Showmodal;
end;
procedure TNMedicine.BitBtn2Click(Sender: TObject);
```

```
'ar HangiKayit:String;
egin
Iangi Kayit := label9.Caption;
f HangiKayit='' Then
egin
Application.Messagebox('Select The Medicine                    ','Warning
',mb_Ok+MB_ICONINFORMATION);
Exit;
end
Else
Try
With DMQuery.QSelect do
  begin
  Close;
   Sql.Clear;
   Sql.Add('Select * from Medicinedet ');
   Sql.Add(' WHERE Medicinedet.MDMID =:Gelen ');
   ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
   Open;
   end;
   if dmquery.QSelect.RecordCount>0 then
    begin
    Application.Messagebox('Information saved before, For edit medicine select
edit','Warning :',mb_Ok+MB_ICONINFORMATION);
     Exit;
     end;
      With DMQuery.Qinsert do
       Begin
        Close;
        Sql.Clear;
        Sql.Add                    ('insert                    into
Medicinedet(MDMID,MBad,MWarning,MEffect,MOverdouse,MMCID)');
        Sql.Add ('Values (:DMID,:Bad,:Warning,:Effect,:Overdouse,:MCID)');
        ParamByName('DMID').AsInteger :=DBLookupComboBox1.KeyValue;
```

```
    ParamByName('Bad').Asstring := Trim(Edit8.Text);
    ParamByName('Warning').Asstring := Trim(Edit7.Text);
    ParamByName('Effect').Asstring := Trim(Edit5.Text);
    ParamByName('Overdouse').Asstring := Trim(Edit6.Text);
    ParamByName('MCID').AsInteger := DBLookupComboBox6.KeyValue;
    Execsql;
   End;
  dbMain.tblMedicinedet.Refresh;
  if DBLookupComboBox7.KeyValue>0 then
  begin
   With DMQuery.Qinsert1 do
    Begin
     Close;
     Sql.Clear;
     Sql.Add ('insert into MEquivalent(MERID,MEEID)');
     Sql.Add ('Values (:ERID,:EEID)');
     ParamByName('ERID').AsInteger := DBLookupComboBox1.KeyValue;
     ParamByName('EEID').AsInteger := DBLookupComboBox7.KeyValue;
     Execsql;
    End;
   end;
 Application.Messagebox('Medicine                                    Saved','Warning
:',mb_Ok+MB_ICONINFORMATION);
  Except
  End;
DBLookupComboBox1.KeyValue:=0;
DBLookupComboBox6.KeyValue:=0;
DBLookupComboBox7.KeyValue:=0;
Edit5.Text:='';
Edit6.Text:='';
Edit7.Text:='';
Edit8.Text:='';
Label20.Caption :='';
Label11.Caption:='';
```

```
d;
ocedure TNMedicine.DBLookupComboBox1Click(Sender: TObject);
gin
bel9.Caption:=DBLookupComboBox1.KeyValue;
Medicine.Refresh;
it5.Text:='';
it6.Text:='';
it7.Text:='';
it8.Text:='';
bel20.Caption :='';
bel11.Caption:='';
BLookupComboBox6.KeyValue:=0;
BLookupComboBox7.KeyValue:=0;
y
ith DMQuery.QSelect do
begin
 Close;
 Sql.Clear;
 Sql.Add('Select * from Medicinedet ');
 Sql.Add(' WHERE Medicinedet.MDMID =:Gelen ');
 ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
 Open;
end;
ith DMQuery.QSelect1 do
 begin
 Close;
 Sql.Clear;
 Sql.Add('Select MCName from MCategory ');
 Sql.Add(' WHERE MCategory.MCID =:Gelen1 ');
 ParamByName('Gelen1').AsInteger                                    :=
MQuery.QSelect.FieldByName('MMCID').AsInteger;
 Open;
 end;
ith DMQuery.QSelect2 do
```

```
  begin
  Close;
   Sql.Clear;
   Sql.Add('Select * from MEquivalent ');
   Sql.Add(' WHERE MERID =:Gelen1 ');
   ParamByName('Gelen1').AsInteger                                    :=
DMQuery.QSelect.FieldByName('MDMID').AsInteger;
   Open;
   end;
With DMQuery.QSelect3 do
  begin
  Close;
   Sql.Clear;
   Sql.Add('Select MName from Medicine ');
   Sql.Add(' WHERE MID =:Gelen2 ');
   ParamByName('Gelen2').AsInteger                                    :=
DMQuery.QSelect2.FieldByName('MEEID').AsInteger;
   Open;
   end;
   if DMQuery.QSelect.RecordCount>0 then
    Begin
     Edit5.Text:=DMQuery.QSelect.FieldByName('MEffect').AsString;
     Edit6.Text:=DMQuery.QSelect.FieldByName('MOverdouse').AsString;
     Edit7.Text:=DMQuery.QSelect.FieldByName('MWarning').AsString;
     Edit8.Text:=DMQuery.QSelect.FieldByName('MBad').AsString;
     Label20.Caption:=DMQuery.QSelect1.FieldByName('MCName').AsString;
     LAbel11.Caption:=DMQuery.QSelect3.FieldByName('MName').AsString;
    end;
 Except
   end;
  end;
procedure TNMedicine.BitBtn3Click(Sender: TObject);
Var HangiKayit:String;
begin
```

```
HangiKayit := Label9.Caption;
if HangiKayit='' Then
begin
Application.Messagebox('Select the Medicine ','Warning
:',mb_Ok+MB_ICONINFORMATION);
Exit;
end;
With DMQuery.QSelect do
  begin
   Close;
   Sql.Clear;
   Sql.Add('Select * from Medicinedet ');
   Sql.Add(' WHERE Medicinedet.MDMID =:Gelen ');
   ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
   Open;
  end;
if DMQuery.QSelect.RecordCount>0 then
begin
With DMQuery.QEdit do
   begin
    Close;
    Sql.Clear;
    Sql.Add('Update Medicinedet Set MEffect=:Effect, MOverdouse=:Overdouse,
MWarning=:Warning, MBad=:Bad, MMCID=:MCID ');
    Sql.Add(' WHERE MDMID=:Hangiilac ');
    ParamByName('Effect').AsString      := Trim(Edit5.Text);
    ParamByName('Overdouse').AsString   := Trim(Edit6.Text);
    ParamByName('Warning').AsString     := Trim(Edit7.Text);
    ParamByName('Bad').AsString         := Trim(Edit8.Text);
    ParamByName('Hangiilac').AsInteger  := strToint(label9.caption);
    if DBLookupComboBox6.KeyValue>0 then
    begin
     ParamByName('MCID').AsInteger      := DBLookupComboBox6.KeyValue;
    end;
```

```
    ExecSql;
    dbMain.tblMedicinedet.Refresh;
   end;
 if label11.Caption='' then
 begin
  With DMQuery.Qinsert do
       Begin
        Close;
        Sql.Clear;
        Sql.Add ('insert into MEquivalent(MERID,MEEID)');
        Sql.Add ('Values (:ERID,:EEID)');
        ParamByName('ERID').AsInteger := DBLookupComboBox1.KeyValue;
        ParamByName('EEID').AsInteger := DBLookupComboBox7.KeyValue;
        Execsql;
        dbMain.tblMEquivalent.Refresh;
       End;
  end
  else
   With DMQuery.QEdit1 do
    begin
     Close;
     Sql.Clear;
     Sql.Add('Update MEquivalent Set MEEID=:EEID');
     Sql.Add(' WHERE MERID=:ERID ');
     ParamByName('ERID').AsInteger    := DBLookupComboBox1.KeyValue;
     ParamByName('EEID').AsInteger    := DBLookupComboBox7.KeyValue;
     ExecSql;
     dbMain.tblMEquivalent.Refresh;
    end;
    Application.Messagebox('MedicineEdited','Warning
:',mb_Ok+MB_ICONINFORMATION);
    Edit5.Text:='';
    Edit6.Text:='';
    Edit7.Text:='';
```

```
    Edit8.Text:='';
    Label20.Caption :='';
    Label11.Caption:='';
    DBLookupComboBox1.KeyValue:=0;
    DBLookupComboBox6.KeyValue:=0;
    DBLookupComboBox7.KeyValue:=0;
   end
  else
  Application.Messagebox('Medicine cant edit because its not saved normally','Warning
:',mb_Ok+MB_ICONINFORMATION);
  exit;
  end;
procedure TNMedicine.BitBtn4Click(Sender: TObject);
Var HangiKayit:String;
   c:Word;
begin
HangiKayit := Label9.Caption;
if HangiKayit='' Then
begin
Application.Messagebox('Select Medicine','Warning :',mb_Ok+MB_ICONINFORMATION);
Exit;
end;
c:=Application.Messagebox('Are you sure for delete the
medicine?','Warning:',MB_YesNo+MB_ICONINFORMATION);
case c of
IDYES : begin
With DMQuery.QDelete do
 begin
  Close;
  Sql.Clear;
  Sql.Add('Delete From Medicinedet ');
  Sql.Add(' WHERE MDMID=:Gelenilac ');
  ParamByName('Gelenilac').AsInteger := strtoint(label9.Caption);
  ExecSql;
```

```
  End;
With DMQuery.QDelete1 do
 begin
  Close;
  Sql.Clear;
  Sql.Add('Delete From MEquivalent ');
  Sql.Add(' WHERE MERID=:Gelenilac ');
  ParamByName('Gelenilac').AsInteger := strtoint(label9.Caption);
  ExecSql;
  End;
  Application.Messagebox('You delete to medicine
informations!','Warning:',MB_OK+MB_ICONINFORMATION);
  end;
IDNO : Exit;
end;
Edit5.Text:='';
Edit6.Text:='';
Edit7.Text:='';
Edit8.Text:='';
Label20.Caption :='';
Label11.Caption:='';
DBLookupComboBox1.KeyValue:=0;
end;
procedure TNMedicine.BitBtn6Click(Sender: TObject);
begin
NCategory.Showmodal;
end;
procedure TNMedicine.FormCloseQuery(Sender: TObject;
  var CanClose: Boolean);
begin
 Edit5.Text:='';
 Edit6.Text:='';
 Edit7.Text:='';
 Edit8.Text:='';
```

```
Edit7.Text:='';
Edit8.Text:='';
Label20.Caption :='';
Label11.Caption:='';
DBLookupComboBox1.KeyValue:=0;
DBLookupComboBox6.KeyValue:=0;
DBLookupComboBox7.KeyValue:=0;
end;
end.
```

NMedicine Form

```
unit ufrmBill;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, ComCtrls, Buttons, Grids, DBGrids;
type
  TBill = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    BitBtn2: TBitBtn;
    BitBtn4: TBitBtn;
    DateTimePicker1: TDateTimePicker;
    DateTimePicker2: TDateTimePicker;
    Label10: TLabel;
    DBGrid2: TDBGrid;
    DBGrid1: TDBGrid;
    BitBtn1: TBitBtn;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
```

```
 public
   { Public declarations }
 end;
var
 Bill: TBill;
implementation
uses uDBMain, uDMQuery, ufrmMain, UTools, ufrmGeneralReports;
{$R *.dfm}
procedure TBill.BitBtn1Click(Sender: TObject);
begin
With DMQuery.QSelect do
   begin
   Close;
    Sql.Clear;
    Sql.Add('Select Hospital.*, SMedicine.* from SMedicine ');
    Sql.Add('INNER JOIN Hospital ON (Hospital.HID=SMedicine.SMHID) ');
    Sql.Add(' Where (SMedicine.SMDate>=:Gelen) and (SMedicine.SMDate<=:Gelen1)');
    ParamByName('Gelen').AsDate  := DateTimePicker1.DateTime;
    ParamByName('Gelen1').AsDate := DateTimePicker2.DateTime;
    Open;
   end;
    DBMain.tblSMedicine.Refresh;
    DBMain.tblHospital.Refresh;
    dbgrid2.DataSource := dmquery.dsQSelect;
end;
procedure TBill.DBGrid2DblClick(Sender: TObject);
var Gelenilac:String;
begin
if dbgrid2.Columns.Grid.Fields[3].AsString='' then
 begin
  exit;
 end;
Gelenilac  :=  dbgrid2.Columns.Grid.Fields[3].AsString;
With DMQuery.QSelect1 do
```

```
 begin
 Close;
  Sql.Clear;
  Sql.Add('Select Medicine.MName, SMedicineDet.* from SMedicineDet ');
  Sql.Add('INNER JOIN Medicine ON (Medicine.MID=SMedicineDet.SMDMID) ');
  Sql.Add(' Where SMedicineDet.SMDSMID=:Gelen ');
  ParamByName('Gelen').AsInteger  := strtoint(Gelenilac);
  Open;
  end;
   DBMain.tblSMedicineDet.Refresh;
   DBMain.tblMedicine.Refresh;
   dbgrid1.DataSource := dmquery.dsQSelect1;
end;
procedure TBill.BitBtn2Click(Sender: TObject);
var Gelenilac:string;
begin
Gelenilac := dbgrid2.Columns.Grid.Fields[3].AsString;
if Gelenilac='' then
begin
Application.Messagebox('Select to Rescription!','Warning
:',mb_Ok+MB_ICONINFORMATION);
exit;
end;
frmGeneralReports.richreport.Clear;
frmGeneralReports.richreport.Lines.Add(format('%-20s%-10s%-15s%-6s%-10s%-13s%-
10s%-10s%-15s%-15s%',
['Name','Prescription No','Medicine Name','Unit','Price','Date','Total','Insurance
No','Foundation','Hospital'])) ;
frmGeneralReports.richreport.Lines.Add(format('%-20s%-10s%-15s%-6s%-10s%-13s%-
10s%-10s%-15s%-15s%',
['_________','_________','_________','_____','_____','_____','_____','_____','____
_____'])) ;
Try
With DMQuery.QSelect2 do
```

```
  begin
  Close;
   Sql.Clear;
   Sql.Add('Select Association.*, Hospital.*, SMedicine.*, Medicine.MName,
SMedicineDet.* from SMedicineDet ');
   Sql.Add('INNER JOIN Medicine ON (Medicine.MID=SMedicineDet.SMDMID) ');
   Sql.Add('INNER JOIN Hospital ON (Hospital.HID=SMedicine.SMHID) ');
   Sql.Add('INNER JOIN Association ON (Association.AID=SMedicine.SMAID) ');
   Sql.Add('INNER JOIN SMedicine ON (SMedicine.SMID=SMedicineDet.SMDSMID) ');
   Sql.Add(' Where SMedicineDet.SMDSMID=:Gelen ');
   ParamByName('Gelen').AsInteger  := strtoint(Gelenilac);
   Open;
   end;
    DBMain.tblSMedicineDet.Refresh;
    DBMain.tblMedicine.Refresh;
    DBMain.tblSMedicine.Refresh;
    DBMain.tblAssociation.Refresh;
    DBMain.tblHospital.Refresh;
 if DMQuery.QSelect2.RecordCount<1 then
 begin
 frmGeneralReports.richreport.Lines.Add('Warning!');
 frmGeneralReports.ShowModal;
 SysUtils.Abort;
 exit;
 end;
 frmGeneralReports.richreport.Lines.Add(format('%-20s%-10s%-15s%-6s%-10s%-13s%-
10s%-10s%-15s%-15s%',
                        [DMQuery.QSelect2.FieldByName('SMWho').AsString ,
                         DMQuery.QSelect2.FieldByName('SMRecipeNo').AsString,
                         DMQuery.QSelect2.FieldByName('MName').AsString,
                         inttostr(DMQuery.QSelect2.FieldByName('SMDUnit').AsInteger),
                         inttostr(DMQuery.QSelect2.FieldByName('SMDPrice').AsInteger),
                        datetostr(DMQuery.QSelect2.FieldByName('SMDate').AsDateTime),
                         intToStr(DMQuery.QSelect2.FieldByName('SMTotal').AsInteger),
```

```
                datetostr(DMQuery.QSelect2.FieldByName('SMDate').AsDateTime),
                 intToStr(DMQuery.QSelect2.FieldByName('SMTotal').AsInteger),
                 DMQuery.QSelect2.FieldByName('SMWhoID').AsString,
                 DMQuery.QSelect2.FieldByName('AName').AsString,
                 DMQuery.QSelect2.FieldByName('HName').AsString,
                "]));


Except
 SysUtils.Abort;
 Exit;
end;
 frmGeneralReports.ShowModal;
end;
procedure TBill.BitBtn4Click(Sender: TObject);
begin
Close;
end;
procedure TBill.FormCreate(Sender: TObject);
begin
 DateTimePicker1.DateTime := Now-1;
 DateTimePicker2.DateTime := Now;
end;
end.
```

Stock Form

```
unit ufrmStock;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DBCtrls, StdCtrls, ExtCtrls, Buttons, ComCtrls;
  TStock = class(TForm)
```

```
  Panel3: TPanel;
  Panel4: TPanel;
  Label10: TLabel;
  DBLookupComboBox1: TDBLookupComboBox;
  BitBtn2: TBitBtn;
  BitBtn3: TBitBtn;
  Label12: TLabel;
  Edit5: TEdit;
  Label2: TLabel;
  DBLookupComboBox6: TDBLookupComboBox;
  Label5: TLabel;
  DateTimePicker1: TDateTimePicker;
  Label1: TLabel;
  Edit1: TEdit;
  Edit2: TEdit;
  Label6: TLabel;
  Label7: TLabel;
  Edit3: TEdit;
  Label3: TLabel;
  Label4: TLabel;
  Label8: TLabel;
  procedure DBLookupComboBox1Click(Sender: TObject);
  procedure BitBtn2Click(Sender: TObject);
  procedure BitBtn3Click(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure Edit1KeyPress(Sender: TObject; var Key: Char);
  procedure Edit2KeyPress(Sender: TObject; var Key: Char);
  procedure Edit5KeyPress(Sender: TObject; var Key: Char);
  procedure Edit3KeyPress(Sender: TObject; var Key: Char);
private
  { Private declarations }
public
  { Public declarations }
end;
```

```
var
  Stock: TStock;
implementation
uses uDBMain, uDMQuery, ufrmMain, UTools;
{$R *.dfm}
procedure TStock.DBLookupComboBox1Click(Sender: TObject);
begin
With DMQuery.QSelect do
   begin
   Close;
    Sql.Clear;
    Sql.Add('Select * from Stock ');
    Sql.Add(' WHERE Stock.SMID =:Gelen ');
    ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
    Open;
    end;
    Label8.Caption:=inttostr(DMQuery.QSelect.FieldByName('SUnit').AsInteger);
    edit5.Text    :=inttostr(DMQuery.QSelect.FieldByName('SBorder').AsInteger);
With DMQuery.QSelect1 do
   begin
   Close;
    Sql.Clear;
    Sql.Add('Select * from MPrice ');
    Sql.Add(' WHERE MPrice.MPMID =:Gelen1 ');
    ParamByName('Gelen1').AsInteger := DBLookupComboBox1.KeyValue;
    Open;
    end;
    Edit2.Text    :=inttostr(DMQuery.QSelect1.FieldByName('MPPrice').AsInteger);
With DMQuery.QSelect2 do
   begin
   Close;
    Sql.Clear;
    Sql.Add('Select * from BMedicine ');
    Sql.Add(' WHERE BMedicine.BMMID =:Gelen2 ');
```

```
    ParamByName('Gelen2').AsInteger := DBLookupComboBox1.KeyValue;
    Open;
    end;
    Edit1.Text    :=inttostr(DMQuery.QSelect2.FieldByName('BMPrice').AsInteger);
 end;
procedure TStock.BitBtn2Click(Sender: TObject);
Var adet:integer;
begin
With DMQuery.QSelect do
   begin
   Close;
    Sql.Clear;
    Sql.Add('Select * from Stock ');
    Sql.Add(' WHERE Stock.SMID =:Gelen ');
    ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
    Open;
    adet:=DMQuery.QSelect.FieldByName('SUnit').AsInteger;
    end;
if DMQuery.QSelect.RecordCount<1 then
 Begin
  adet:=adet+strtoint(edit3.Text);
  With DMQuery.Qinsert do
       Begin
        Close;
        Sql.Clear;
        Sql.Add ('insert into Stock(SMID,SUnit,SBorder)');
        Sql.Add ('Values (:MID,:Unit,:Border)');
        ParamByName('MID').AsInteger   :=DBLookupComboBox1.KeyValue;
        ParamByName('Unit').AsInteger  := adet;
        ParamByName('Border').AsInteger:= strtoint(Trim(Edit5.Text));
        Execsql;
        dbMain.tblStock.Refresh;
       End;
      end
```

```
else
 begin
  adet:=adet+strtoint(edit3.Text);
  With DMQuery.QEdit do
    begin
     Close;
     Sql.Clear;
     Sql.Add('Update Stock Set SUnit=:Unit, SBorder=:Border ');
     Sql.Add(' WHERE SMID=:Hangiilac ');
     ParamByName('Unit').AsInteger        := adet;
     ParamByName('Border').AsInteger      := strtoint(Trim(Edit5.Text));
     ParamByName('Hangiilac').AsInteger   := DBLookupComboBox1.KeyValue;
     ExecSql;
     dbMain.tblStock.Refresh;
 end;
 With DMQuery.QSelect1 do
    begin
    Close;
     Sql.Clear;
     Sql.Add('Select * from MPrice ');
     Sql.Add(' WHERE MPrice.MPMID =:Gelen ');
     ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
     Open;
     end;
 end;
 if DMQuery.QSelect1.RecordCount<1 then
  Begin
    With DMQuery.Qinsert1 do
        Begin
         Close;
         Sql.Clear;
         Sql.Add ('insert into MPrice(MPMID,MPPrice)');
         Sql.Add ('Values (:MID,:Price)');
         ParamByName('MID').AsInteger   :=DBLookupComboBox1.KeyValue;
```

```
      ParamByName('Price').AsInteger :=strtoint(Edit2.text);
      Execsql;
      dbMain.tblMPrice.Refresh;
     End;
   end
else
 Begin
  With DMQuery.QEdit1 do
   begin
    Close;
    Sql.Clear;
    Sql.Add('Update MPrice Set MPPrice=:Price ');
    Sql.Add(' WHERE MPMID=:Hangiilac ');
    ParamByName('Price').AsInteger      := strtoint(Trim(Edit2.Text));
    ParamByName('Hangiilac').AsInteger  := DBLookupComboBox1.KeyValue;
    ExecSql;
    dbMain.tblMPrice.Refresh;
 end;
end;
  With DMQuery.Qinsert2 do
      Begin
       Close;
       Sql.Clear;
       Sql.Add ('insert into BMedicine(BMMID,BMDate,BMUnit,BMPrice,BMWID)');
       Sql.Add ('Values (:MMID,:MDate,:MUnit,:MPrice,:MWID)');
       ParamByName('MMID').AsInteger  :=DBLookupComboBox1.KeyValue;
       ParamByName('MDate').AsDate    :=DateTimePicker1.Date;
       ParamByName('MUnit').AsInteger :=strtoint(Edit3.Text);
       ParamByName('MPrice').AsInteger :=(strtoint(Edit1.text)*strtoint(Edit3.Text));
       ParamByName('MWID').AsInteger  :=DBLookupComboBox6.KeyValue;
       Execsql;
       dbMain.tblBMedicine.Refresh;
      End;
     DBLookupComboBox1.KeyValue:=0;
```

```
    DBLookupComboBox6.KeyValue:=0;
    Edit5.Text:='';
    Edit3.Text:='';
    Edit1.Text:='';
    Edit2.Text:='';
    Label8.Caption:='';
end;
procedure TStock.BitBtn3Click(Sender: TObject);
begin
DBLookupComboBox1.KeyValue:=0;
DBLookupComboBox6.KeyValue:=0;
Label8.Caption:='';
Edit5.Text:='';
Edit3.Text:='';
Edit1.Text:='';
Edit2.Text:='';
Stock.Close;
end;
procedure TStock.FormCreate(Sender: TObject);
begin
DateTimePicker1.DateTime := Now;
end;
procedure TStock.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
Key:=Rakam(Key) ;
end;
procedure TStock.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
Key:=Rakam(Key) ;
end;
procedure TStock.Edit5KeyPress(Sender: TObject; var Key: Char);
begin
Key:=Rakam(Key) ;
end;
```

```
Key:=Rakam(Key) ;
end;
procedure TStock.Edit3KeyPress(Sender: TObject; var Key: Char);
begin
Key:=Rakam(Key) ;
end;
end.
```

Warehouse Form

```
unit ufrmWarehouse;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, DBCtrls, ExtCtrls;
type
  TWarehouse = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Label10: TLabel;
    DBLookupComboBox1: TDBLookupComboBox;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    Label12: TLabel;
    Edit5: TEdit;
    Edit6: TEdit;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label1: TLabel;
```

```
  Label2: TLabel;
  Edit3: TEdit;
  Label9: TLabel;
  procedure BitBtn1Click(Sender: TObject);
  procedure DBLookupComboBox1Click(Sender: TObject);
  procedure BitBtn2Click(Sender: TObject);
  procedure BitBtn3Click(Sender: TObject);
  procedure BitBtn4Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;
var
 Warehouse: TWarehouse;
implementation
uses uDBMain, uDMQuery, ufrmMain, ufrmNWarehouse;
{$R *.dfm}
procedure TWarehouse.BitBtn1Click(Sender: TObject);
begin
NWarehouse.ShowModal;
end;
procedure TWarehouse.DBLookupComboBox1Click(Sender: TObject);
begin
Label9.Caption:=DBLookupComboBox1.KeyValue;
Warehouse.Refresh;
Edit5.Text:='';
Edit6.Text:='';
Edit7.Text:='';
Edit8.Text:='';
Edit2.Text:='';
Edit1.Text:='';
Edit3.Text:='';
 Try
```

```
With DMQuery.QSelect do
  begin
   Close;
   Sql.Clear;
   Sql.Add('Select * from Warehouse ');
   Sql.Add(' WHERE Warehouse.WID =:Gelen ');
   ParamByName('Gelen').AsInteger := DBLookupComboBox1.KeyValue;
   Open;
  end;
if DMQuery.QSelect.RecordCount>0 then
    Begin
     Edit5.Text:=DMQuery.QSelect.FieldByName('WAddress').AsString;
     Edit6.Text:=DMQuery.QSelect.FieldByName('WTax').AsString;
     Edit7.Text:=DMQuery.QSelect.FieldByName('WContact').AsString;
     Edit8.Text:=inttostr(DMQuery.QSelect.FieldByName('WTel1').AsInteger);
     Edit2.Text:=inttostr(DMQuery.QSelect.FieldByName('WTel2').AsInteger);
     Edit1.Text:=inttostr(DMQuery.QSelect.FieldByName('WFax').AsInteger);
     Edit3.Text:=DMQuery.QSelect.FieldByName('WWeb').AsString;
    end;
   Except
  end;
end;
procedure TWarehouse.BitBtn2Click(Sender: TObject);
Var HangiKayit:String;
begin
HangiKayit := label9.Caption;
if HangiKayit='' Then
begin
Application.Messagebox('Select Warehouse','Warning
:',mb_Ok+MB_ICONINFORMATION);
Exit;
end
Else
Try
```

```
    With DMQuery.QEdit do
     begin
      Close;
      Sql.Clear;
      Sql.Add('Update Warehouse Set WAddress=:Address, WTel1=:Tel1, WTel2=:Tel2,
WFax=:Fax, WWeb=:Web, WTax=:Tax, WContact=:Contact ');
      Sql.Add(' WHERE WID=:Gelen1 ');
      ParamByName('Address').AsString   := Trim(Edit5.Text);
      ParamByName('Tel1').AsInteger     := strtoint(Trim(Edit8.Text));
      ParamByName('Tel2').AsInteger     := strtoint(Trim(Edit2.Text));
      ParamByName('Fax').AsInteger      := strtoint(Trim(Edit1.Text));
      ParamByName('Web').AsString       := Trim(Edit3.Text);
      ParamByName('Tax').AsString       := Trim(Edit6.Text);
      ParamByName('Contact').AsString   := Trim(Edit7.Text);
      ParamByName('Gelen1').AsInteger   := strToint(label9.caption);
      ExecSql;
      dbMain.tblWarehouse.Refresh;
     end;
    Application.Messagebox('Warehouse Saved','Warning
:',mb_Ok+MB_ICONINFORMATION);
  Except
  End;
Edit5.Text:='';
Edit6.Text:='';
Edit7.Text:='';
Edit8.Text:='';
Edit1.Text:='';
Edit2.Text:='';
Edit3.Text:='';
end;
procedure TWarehouse.BitBtn3Click(Sender: TObject);
Var HangiKayit:String;
begin
HangiKayit := label9.Caption;
```

```
if HangiKayit='' Then
begin
Application.Messagebox('Select Warehouse','Warning
:',mb_Ok+MB_ICONINFORMATION);
Exit;
end
Else
Try
  With DMQuery.QEdit do
   begin
      Close;
      Sql.Clear;
      Sql.Add('Update Warehouse Set WAddress=:Address, WTel1=:Tel1, WTel2=:Tel2,
WFax=:Fax, WWeb=:Web, WTax=:Tax, WContact=:Contact ');
      Sql.Add(' WHERE WID=:Gelen1 ');
      ParamByName('Address').AsString   := Trim(Edit5.Text);
      ParamByName('Tel1').AsInteger     := strtoint(Trim(Edit8.Text));
      ParamByName('Tel2').AsInteger     := strtoint(Trim(Edit2.Text));
      ParamByName('Fax').AsInteger      := strtoint(Trim(Edit1.Text));
      ParamByName('Web').AsString       := Trim(Edit3.Text);
      ParamByName('Tax').AsString       := Trim(Edit6.Text);
      ParamByName('Contact').AsString   := Trim(Edit7.Text);
      ParamByName('Gelen1').AsInteger   := strToint(label9.caption);
      ExecSql;
      dbMain.tblWarehouse.Refresh;
     end;
   Application.Messagebox('Warehouse Edited','Warning
:',mb_Ok+MB_ICONINFORMATION);
  Except
  End;
end;
procedure TWarehouse.BitBtn4Click(Sender: TObject);
Var HangiKayit:String;
   c:Word;
```

```
begin
HangiKayit := Label9.Caption;
if HangiKayit='' Then
begin
Application.Messagebox('Select Warehouse','Warning
:',mb_Ok+MB_ICONINFORMATION);
Exit;
end;
c:=Application.Messagebox('Are you sure for delete to
warehouse?','Warning:',MB_YesNo+MB_ICONINFORMATION);
case c of
IDYES : begin
With DMQuery.QDelete do
 begin
  Close;
  Sql.Clear;
  Sql.Add('Delete From Warehouse ');
  Sql.Add(' WHERE WID=:Gelendepo ');
  ParamByName('Gelendepo').AsInteger := strtoint(label9.Caption);
  ExecSql;
  dbMain.tblWarehouse.Refresh;
  End;
  Application.Messagebox('Warehouse
Deleted','Warning:',MB_OK+MB_ICONINFORMATION);
  end;
IDNO : Exit;
end;
Edit2.Text:='';
Edit3.Text:='';
end;
end.
```